

**UNIVERSIDAD POLITÉCNICA SALESIANA  
SEDE QUITO**

**CARRERA: INGENIERÍA DE SISTEMAS**

**Trabajo de titulación previo a la obtención del título de:  
INGENIERAS DE SISTEMAS**

**TEMA:**

**ANÁLISIS Y CREACIÓN DE VISUALIZADORES PROTOTIPO DE  
DIVERSAS LIBRERÍAS DE CÓDIGO ABIERTO CON DATOS  
GEOGRÁFICOS DE LA INSPECTORÍA SALESIANA; A TRAVÉS DE UNA  
CONEXIÓN CON LA BASE DE DATOS POSTGIS**

**AUTORAS:**

**ESCUDERO NOGALES MERY JESSICA  
GUATAPI CRIOLLO ESPERANZA NATALI**

**DIRECTOR:**

**GUSTAVO ERNESTO NAVAS RUILOVA**

**Quito, abril de 2015**

**DECLARATORIA DE RESPONSABILIDAD Y AUTORIZACIÓN DE USO  
DEL TRABAJO DE TITULACIÓN**

Nosotras, autorizamos a la Universidad Politécnica Salesiana la publicación total o parcial de este trabajo de titulación su reproducción sin fines de lucro.

Además, declaramos que los conceptos, análisis desarrollados y las conclusiones del presente trabajo son de exclusiva responsabilidad de las autoras.

Quito, abril 2015

---

Mery Jessica Escudero Nogales

CI. 172222625-3

---

Esperanza Natali Guatapi Criollo

CI. 180439264-3

## **DEDICATORIA**

Dedico este trabajo a Dios, por haberme dado la vida. A mis padres Ramiro, Margoth por demostrarme su cariño y apoyo incondicional. A mis hermanos Maribel, Klever y mi sobrino Emerson, por compartir momentos significativos conmigo. A Israel, quien con su amor y apoyo me alentó constantemente para culminar este trabajo. A Taty que me ha brindado su valiosa amistad durante toda la carrera. A Naty que gracias a su apoyo hemos logrado culminar este proyecto.

Jessica Escudero Nogales.

Dedico este trabajo a Dios primeramente por darme la vida, sabiduría, y guiar mi camino día a día, a mis padres por su amor y apoyo que gracias a su sacrificio eh llegado a culminar un sueño más en mi vida, a mis hermanos porque siempre estaban apoyándome moral y económicamente, a mi compañera Jessica Escudero que a pesar de las dificultades para el desarrollo del proyecto hemos logrado finalizar.

Natali Guatapi Criollo

## **AGRADECIMIENTOS**

A la Universidad, porque en sus aulas, recibimos el conocimiento intelectual y humano de cada uno de los maestros, quienes siempre estuvieron dispuestos a ayudarnos con nuestras inquietudes en especial al Ing. Rodrigo Tufiño.

A nuestro tutor Ing. Gustavo Navas quien con sus conocimientos y consejos ha sido una guía para concluir con éxito este proyecto.

## ÍNDICE

INTRODUCCIÓN .....	1
CAPÍTULO 1 .....	3
MARCO TEÓRICO.....	3
1.1 Objetivos .....	3
1.1.1 Objetivo general .....	3
1.1.2 Objetivos específicos .....	3
1.2 Resultados esperados .....	4
1.3 Alcance.....	4
1.2 Marco teórico .....	4
1.4.1 Georeferenciación .....	4
1.4.2 Visualizadores .....	5
1.4.3 Sistemas de información geográfica .....	5
1.4.4 Librerías de código abierto para el procesamiento de información geográfica o geoprocesamiento.....	7
1.4.4.1 Librería GeoPHP .....	7
1.4.4.2 Librería GeoTools .....	8
1.4.5 Servidores Web de datos geoespaciales .....	9
1.4.5.1 GeoServer.....	10
1.4.6 Bases de datos. ....	12
1.4.6.1 PostgreSQL .....	12
1.4.6.2 PostGIS .....	13
1.4.6.2.1 Funcionamiento de PostGIS.....	14
1.4.6.3 Geodatabase .....	14
1.4.7 Lenguajes de programación, herramientas y servidores .....	16
1.4.8 Metodología de desarrollo de software .....	17
1.4.8.1 Metodología SCRUM .....	18
CAPÍTULO 2 .....	22
ANÁLISIS .....	22
2.1 Especificaciones de la Metodología Scrum para el proyecto.....	22
2.1.1 Roles.....	22
2.1.2 Identificación de los requisitos funcionales .....	23
2.1.3 Planes de lanzamiento .....	28

2.1.4	Identificación de los requerimientos no funcionales de los visualizadores .....	30
2.1.5	Distribución, revisión y ajustes de los estándares del producto.....	31
2.1.6	Sprint.....	31
2.1.7	Cierre.....	34
CAPÍTULO 3 .....		35
DISEÑO Y CONSTRUCCIÓN .....		35
3.1	Diseño de prototipos .....	35
3.2	Herramientas para el desarrollo de los prototipos.....	43
3.3	Construcción de prototipos .....	44
3.3.1	Configuración de GeoServer.....	44
3.3.2	Desarrollo del visualizador prototipo GeoServer en la plataforma Netbeans 8.0.2.....	49
3.3.3	Visualizador GeoPHP .....	51
3.3.3.1	Capa del modelo.....	52
3.3.4	Construcción de visualizador prototipo GeoTools.....	53
CAPÍTULO 4.....		58
IMPLEMENTACIÓN Y PRUEBAS .....		58
4.1	Implementación en el servidor CIMA.....	58
4.1.1	Programas.....	58
4.1.1.1	Requerimientos para la implementación de los visualizadores prototipos VisGeo(GeoServer, GeoTools,GeoPHP).....	58
4.1.2	Restaurar base de datos .....	59
4.1.3	Levantamiento de los visores prototipo VisGeo en el servidor. ....	59
4.1.3.1	Visor GeoPHP .....	59
4.1.3.2	Visor GeoServer.....	60
4.1.3.3	Visor de GeoTools .....	60
4.2	Pruebas .....	61
4.2.1	Plan de pruebas .....	62
4.2.1.1	Casos de pruebas .....	63
4.2.1.1.1	Pruebas desarrolladas al prototipo GeoPHP .....	63
4.2.1.1.2	Pruebas desarrolladas al prototipo GeoServer .....	65
4.2.1.1.3	Pruebas desarrolladas al prototipo GeoTools.....	66
4.3	Manual de usuario .....	68
4.3.1	Manual de usuario para GeoPHP y Geo Tools .....	68

4.3.2 Manual de usuario para GeoServer .....	74
CONCLUSIONES .....	79
RECOMENDACIONES .....	81
LISTA DE DEFERENCIAS .....	82
ANEXOS .....	86

## ÍNDICE DE FIGURAS

<i>Figura 1.</i> Ejemplo del Visualizador GeoServer.....	5
<i>Figura 2.</i> Sistema de Información Geográfica.....	6
<i>Figura 3.</i> Integración GeoPHP con PostGIS. ....	8
<i>Figura 4.</i> Stock de librerías de GeoTools. ....	9
<i>Figura 5.</i> Arquitectura de un GeoServer.....	11
<i>Figura 6.</i> Plataforma Administrativa de GeoServer .....	12
<i>Figura 7.</i> PostGIS .....	13
<i>Figura 8.</i> Repositorio de una Geodatabase .....	15
<i>Figura 9.</i> Estructura central de Scrum. ....	19
<i>Figura 10.</i> Diagrama de casos de uso para los visualizadores GeoPHP y GeoTools	37
<i>Figura 11.</i> Diagrama de casos de uso para el visualizadores GeoServer .....	38
<i>Figura 12.</i> Diagrama de actividades para los prototipos GeoPHP y GeoTools.....	40
<i>Figura 13.</i> Diagrama de actividades para el prototipo GeoServer.....	42
<i>Figura 14.</i> Accediendo a PostGIS .....	45
<i>Figura 15.</i> Accediendo a PostGIS .....	45
<i>Figura 16.</i> Agregar nuevo.....	46
<i>Figura 17.</i> Seleccionar nuevo origen .....	46
<i>Figura 18.</i> Configurar nueva vista SQL .....	47
<i>Figura 19.</i> Insertar variables por defecto .....	48
<i>Figura 20.</i> Insertar el SRS .....	48
<i>Figura 21.</i> Observar la capa insertada .....	49
<i>Figura 22.</i> Clase myscripts.js.....	49
<i>Figura 23.</i> Inserción de código en la clase landing.php .....	50
<i>Figura 24.</i> Creación de la clase clase myscripts.js .....	50
<i>Figura 25.</i> Ejecución del prototipo GeoPHP .....	59
<i>Figura 26.</i> Ejecución del prototipo GeoServer .....	60
<i>Figura 27.</i> Ejecución de los prototipos GeoTools .....	61
<i>Figura 28.</i> Creación de plan de pruebas .....	62
<i>Figura 29.</i> Creación de plan de pruebas .....	63
<i>Figura 30.</i> Guión grabado al prototipo GeoPHP .....	63
<i>Figura 31.</i> Gráfico de resultados del prototipo GeoPHP.....	64
<i>Figura 32.</i> Informe agregado del prototipo GeoPHP.....	64

<i>Figura 33.</i> Guión grabado al prototipo GeoServer .....	65
<i>Figura 34.</i> Gráfico de resultados del prototipo GeoServer.....	65
<i>Figura 35.</i> Informe agregado del prototipo GeoServer .....	66
<i>Figura 36.</i> Guión grabado prototipo GeoTools .....	66
<i>Figura 37.</i> Gráfico de resultados del prototipo GeoTools .....	67
<i>Figura 38.</i> Informe agregado del prototipo GeoTools.....	67
<i>Figura 39.</i> Página de inicio GeoTools y GeoPHP .....	69
<i>Figura 40.</i> Seleccionar casa salesiana.....	69
<i>Figura 41.</i> Seleccionar tipo de obra.....	70
<i>Figura 42.</i> Localización en el mapa de la obra.....	70
<i>Figura 43.</i> Seleccionar tipo de mapa MapQuest.....	71
<i>Figura 44.</i> Seleccionar tipo de mapa Watercolor .....	71
<i>Figura 45.</i> Seleccionar tipo de mapa Sat .....	72
<i>Figura 46.</i> Maximizar mapa. ....	72
<i>Figura 47.</i> Minimizar mapa. ....	73
<i>Figura 48.</i> Pantalla completa. ....	73
<i>Figura 49.</i> Página de inicio Geo Server.....	74
<i>Figura 50.</i> Seleccionar casa salesiana.....	74
<i>Figura 51.</i> Seleccionar tipo de obra.....	75
<i>Figura 52.</i> Localización en el mapa la obra.....	75
<i>Figura 53.</i> Seleccionar tipo de mapa MapQuest.....	76
<i>Figura 54.</i> Seleccionar tipo de mapa Watercolor .....	76
<i>Figura 55.</i> Seleccionar tipo de mapa Sat .....	77
<i>Figura 56.</i> Maximizar mapa. ....	77
<i>Figura 57.</i> Minimizar mapa. ....	78
<i>Figura 58.</i> Pantalla completa. ....	78

## ÍNDICE DE TABLAS

Tabla 1. <i>Roles del proyecto</i> .....	22
Tabla 2. <i>Requerimientos funcionales del producto</i> .....	23
Tabla 3. <i>Historias de usuario</i> .....	24
Tabla 4. <i>Iteraciones realizadas en los prototipos</i> .....	29
Tabla 5. <i>Cronograma de trabajo por iteraciones</i> .....	32
Tabla 6. <i>Usabilidad de los componentes para el visor web</i> .....	50
Tabla 7. <i>Usabilidad de las librerías de GeoPHP en el visor web</i> .....	52
Tabla 8. <i>Usabilidad de las librerías de GeoTools en el visor web</i> .....	55

## ÍNDICE DE ANEXOS

Anexo 1. Tipo de metadatos .....	86
Anexo 2. Principales métodos de la librería GeoPHP.....	87
Anexo 3. Descripción de las librerías que conforman GeoTools.....	90

## **RESUMEN**

Los visualizadores prototipos son sitios web que brindan información sobre las Obras Salesianas que se encuentran en todo el país, al escoger la ubicación de la casa salesiana permite visualizar la cantidad de obras de forma numérica y seleccionar el tipo de obra que se requiere consultar para visualizar en el mapa. Al presionar en el mapa la obra salesiana escogida da como resultado la información de la misma con su respectiva posición e imagen.

Además los visores de GeoPHP y GeoTools permiten visualizar la información geográfica. También se puede maximizar, minimizar y realizar pantalla completa al mapa y a su vez cambiar los estilos de los mismos.

Este proyecto está realizado con la base de datos PostGIS que permite el manejo de datos geográficos, para llevar a cabo su desarrollo se utilizó la metodología Scrum.

## **ABSTRACT**

The displays prototypes are websites that provide information about the Salesian Works that are found throughout the country, choosing the location of the Salesian house to visualize the amount of work numerically and select the type of work required to display query on the map. By clicking on the map the Salesian chosen results in the information the same with their respective position and image.

Furthermore viewers GeoPHP and GeoTools to visualize geographic information. You can also maximize, minimize and make full screen map and in turn change the styles of them.

This project was made with PostGIS database that allows the management of geographic data, to perform the Scrum development methodology was used.

## INTRODUCCIÓN

Hoy en día la tecnología se ha desarrollado de acuerdo a los requerimientos de los usuarios. Cada vez son más las necesidades de procesamiento de datos e información en tiempo real con diversos fines como la toma de decisiones, el análisis y comparación de metadatos, la visualización de datos geográficos, entre otros. Debido a este último, se han creado diversas librerías de código abierto para georeferenciación y conjuntamente con los visualizadores web geográficos, se han convertido en un instrumento fundamental para la búsqueda y visualización de datos geográficos en tiempo real. Un visor web geográfico es una herramienta para la visualización de mapas interactivos y tiene como principal uso la toma de decisiones.

La georeferenciación es el proceso en el que se define la localización de un objeto en un sistema de coordenadas y datos determinados. Los visualizadores se han desarrollado con la finalidad de representar mediante mapas la ubicación que el usuario desee observar.

En los últimos años los visualizadores conjuntamente con las librerías de código abierto han tenido un crecimiento y desarrollo importantes en la búsqueda, localización de objetos y zonas en mapas. Algunas de las librerías más usadas para este fin son: Google maps, Google Earth, Open Layers, entre otras.

Debido a la necesidad de visualizar los datos geográficos desde la inspección Salesiana y procesarlos en tiempo real haciendo uso de las nuevas librerías de código abierto, se decide realizar un estudio sobre las diferentes librerías y visualizadores geográficos para procesar esta información y así determinar cuál de todas sería la más efectiva para el trabajo. Se hace necesario entonces, investigar y analizar las librerías de código abierto para desarrollar proyectos que tengan vinculación con la georeferenciación. La creación de los visualizadores prototipo con las librerías de código abierto permite visualizar con respuestas cada vez más efectivas y datos reales.

Por lo cual se realizó en el capítulo uno un análisis de las librerías GeoTools, GeoPHP y el servidor Geoserver. Además, la descripción de la base de datos postgis, servidores web y lenguajes de programación. El capítulo dos consiste en el análisis para la creación de los visualizadores prototipos GeoTools, GeoPHP y Geoserver en base a la metodología Scrum.

En el capítulo tres se desarrolló y se construyó los visualizadores prototipos Geoserver, GeoTools y GeoPHP. Más adelante en el capítulo cuatro consta la implementación en el servidor CIMA y las pruebas realizadas con la herramienta JMeter.

Finalmente se realizó las conclusiones y recomendaciones del trabajo realizado.

# CAPÍTULO 1

## MARCO TEÓRICO

### 1.1 Objetivos

#### 1.1.1 Objetivo general

Analizar y crear visualizadores prototipo de diversas librerías de código abierto con datos geográficos de la Inspectoría Salesiana, a través de una conexión con la base de datos PostGIS.

#### 1.1.2 Objetivos específicos

Para dar cumplimiento al objetivo general planteado y guiar el proceso de investigación del presente proyecto se plantean como objetivos específicos los siguientes:

- Investigar el componente PostGIS y su funcionamiento.
- Crear visualizadores prototipos con las librerías de georeferenciación como son GeoTools, GeoPHP, Geodatabase y GeoServer con los datos geográficos de la Inspectoría Salesiana; a través de una conexión con la base de datos PostGIS.
- Realizar pruebas de los prototipos con la finalidad de detectar y corregir errores de manera eficiente para los usuarios.
- Generar manuales de usuario de los visualizadores prototipo de georeferenciación como son GeoTools, GeoPHP, Geodatabase y GeoServer con los datos geográficos de la Inspectoría Salesiana; a través de una conexión con la base de datos PostGIS.
- Implementar los prototipos desarrollados en el servidor del CIMA UPS.

El presente proyecto de investigación y desarrollo se basa en la creación de visualizadores prototipo y conexión con la base de datos PostGIS utilizando librerías de código abierto como son: GeoServer, que es un servidor de software de código abierto escrito en Java que permite a los usuarios compartir y editar datos geoespaciales; GeoTools: librería Java que proporciona acceso a datos en distintos formatos; Geodatabase: son herramientas de software dedicadas a administrar y

gestionar grandes bases de datos; GeoPHP: es una librería PHP de código abierto para hacer las operaciones de geometría.

## **1.2 Resultados esperados**

La finalidad de la creación de los prototipos es levantar la información geográfica con la librería propuesta; la elaboración de visualizadores con datos de la Inspectoría Salesiana y la evaluación de funcionamiento y eficiencia.

## **1.3 Alcance**

Con el siguiente proyecto de tesis se pretende investigar las diversas librerías de código abierto como son GeoTools y GeoPHP; servidores web como es GeoServer y modelos de datos como Geodatabase para así crear prototipos de visualizadores con los datos geográficos de la Inspectoría Salesiana a través de una conexión con la base de datos PostGIS.

La propuesta del presente proyecto al realizar un análisis de las diferentes librerías de código abierto antes mencionadas se procederá a realizar manuales de usuario de los prototipos y un cuadro de las librerías empleadas, además de analizar qué librería es la más óptima a utilizar para lograr un uso más eficiente y orientar a los desarrolladores a emplear dicha librería de código abierto.

## **1.2 Marco teórico**

### **1.4.1 Georeferenciación**

Primeramente, antes de entrar en detalles técnicos se deben definir conceptos claves de la presente investigación como la Georeferenciación.

Georeferenciación, es el proceso mediante el cual se añade la descripción geográfica a cualquier tipo de objeto digital y además permite agregar coordenadas geográficas como latitud y longitud a su base de datos para que pueda ser visualizada en un mapa.

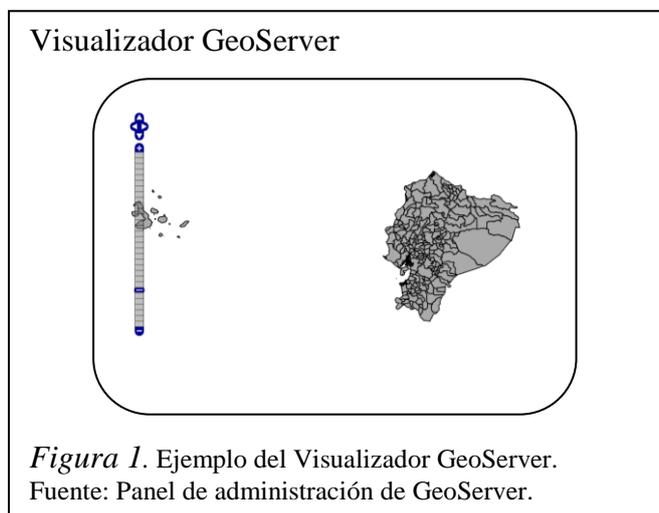
La Georeferenciación se puede realizar conociendo tres o más puntos del objeto que se quiere manipular y a estos puntos se los conoce como gpc (ground control point), ya que ayudan a colocar con precisión el mapa sobre la superficie terrestre tal y como se conoce en la actualidad (Roset & N, 2012).

### 1.4.2 Visualizadores

Uno de los objetivos del presente trabajo de investigación es la realización de visualizadores que permitan al usuario navegar, explorar, visualizar y compartir información usando nuevas librerías de código abierto como GeoTools y GeoPHP; un servidor como GeoServer y un modelo de datos como Geodatabase, que están diseñados para trabajar con información organizada de base de datos y georeferenciada. Por lo tanto los visualizadores creados serán utilizados para mapear y analizar eventos que ocurren en un área geográfica.

Un Visualizador de Información Geográfica es una aplicación que permite observar y consultar datos geográficos en tiempo real, además permite construir un mapa propio de acuerdo a las necesidades del usuario y elegir el camino más óptimo para llegar al destino.

Mediante el visualizador de datos geográficos se puede realizar consultas interactivas de la información entre el usuario y el sistema. A continuación se presenta la imagen de un Visualizador.

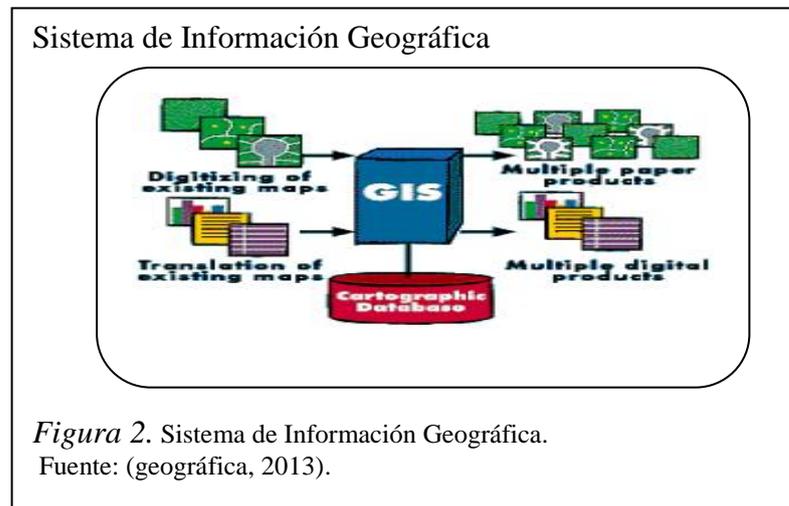


### 1.4.3 Sistemas de información geográfica

Un sistema de información geográfica (GIS) es un sistema de hardware, software y procedimientos diseñados para soportar la captura, administración, manipulación,

análisis, modelamiento y graficación de datos u objetos referenciados especialmente, para resolver problemas complejos de planificación y de gestión (geográfica, 2013).

A continuación se presenta un modelo de los sistemas de información geográfica:



Los sistemas de información geográfica se estructuran en diferentes conjuntos de información como son:

**Mapas interactivos:** permiten dar respuesta a cuestiones concretas, y presentan un resultado de dichas respuestas.

**Datos Geográficos:** en la base de datos se incluye información vectorial y raster, modelos digitales del terreno, redes lineales, información procedente de estudios topográficos, topologías y atributos.

**Modelos de datos:** incorpora, al igual que otros sistemas de información, reglas de comportamiento e integridad de la información.

**Metadatos:** son los datos que describen la información geográfica, que permite al usuario organizar, realizar búsquedas y acceder a información geográfica compartida, también describen varios atributos de los objetos de información que les proveen significado, contexto y organización a los mismos (Jack Dangermond, 2013).

Los metadatos se pueden clasificar en tres amplias categorías las cuales están en el Anexo 1.

#### **1.4.4 Librerías de código abierto para el procesamiento de información geográfica o geoprocésamiento.**

Para llevar a cabo el geoprocésamiento de la información se hacen uso de librerías de código abierto en muchos casos y que proporcionan métodos para la manipulación geoespacial como lo son GEOS, JTS, GeoTools y GeoPHP. La librería JTS *Topology Suite*, es una librería escrita completamente en Java y que permite trabajar con geometrías incorporando una gran colección de algoritmos espaciales en segunda dimensión. El rendimiento de esta tecnología es elevado permitiendo el trabajo y la utilización de sus algoritmos en tiempo de ejecución. La librería GEOS por su parte, es una versión en el lenguaje C++ de la librería JTS. GeoPHP, es una alternativa a GEOS en PHP, aunque si la librería GEOS está instalada entonces GeoPHP usará sus funciones para mejorar la eficiencia. En el presente trabajo solo se analizarán a fondo las librerías GeoTools y GeoPHP.

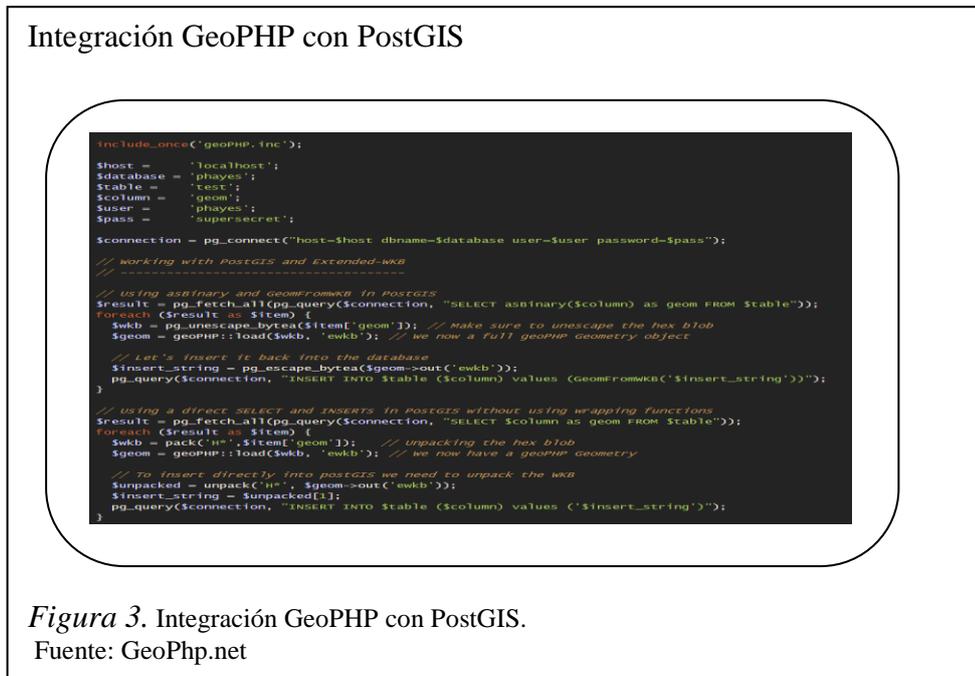
##### **1.4.4.1 Librería GeoPHP**

GeoPHP es una librería nativa de PHP de código abierto que es utilizada para realizar operaciones geométricas. Está escrita 100% en PHP y puede leerse y escribirse en una gran variedad de formatos como son: WKT, EWKT, WKB, EWKB, GeoJSON, KML, GPX y GeoRSS. Trabaja con puntos, líneas, polígonos, colecciones geométricas, etc.

WKT(Well-Known Text): es una representación de geometrías que está diseñada para intercambiar datos geométricos en formato ASCII. Está diseñada para describir objetos espaciales en forma vectorial.

WKB(Well-Known Binary): es una representación de valores geométricos que está definida por la especificación OpenGis. WKB se utiliza para intercambiar datos como cadenas binarias representadas por valores BLOB que contienen información geométrica WKB. (GeoPHP, 2014).

GeoPHP, a través de su adaptador de EWKB, tiene una buena integración con PostGIS, además proporciona una clase estática que contiene las funciones de beneficios útiles y como característica tiene que todos los métodos deben ser llamados estáticamente. A continuación se presenta un fragmento de código de ejemplo de la integración de esta librería con PostGIS.



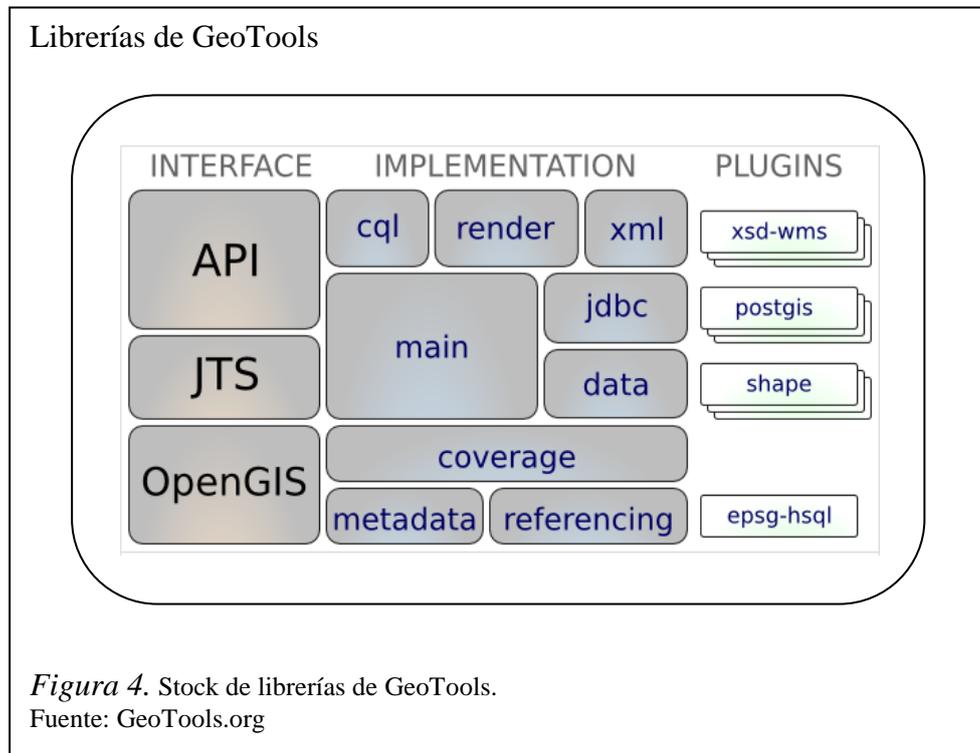
GeoPHP mantiene una extensión PHP de GEOS que proporciona a las aplicaciones que hacen su uso, de un aumento en el rendimiento cuando está instalado GEOS en el mismo servidor. GeoPHP utiliza los métodos más comunes que se muestran en el Anexo2.

#### 1.4.4.2 Librería GeoTools

GeoTools es una librería de código abierto desarrollada en Java que provee métodos estándares para la manipulación de datos geoespaciales, como por ejemplo en la implementación de los Sistemas de Información Geográfica (GIS, por sus siglas en inglés). La librería GeoTools implementa las especificaciones del Open Geospatial Consortium OGC de la misma forma en que estas fueron desarrolladas originalmente (GeoTools, 2014).

GeoTools maneja base de datos espaciales que es compatible con PostgreSQL/PostGIS. Además está compuesto por un gran número de módulos y componentes. Su estructura es en forma de pila de forma tal que cada sub-librería o

archivo se construyen sobre los conceptos implementados en los archivos .jars, como se indica en la siguiente figura.



En el Anexo 3 se presentan algunas de las librerías y métodos usados en GeoTools.

#### 1.4.5 Servidores Web de datos geospaciales

Un servidor web o servidor HTTP, es un programa que procesa cualquier aplicación del lado del servidor realizando conexiones bidireccionales o unidireccionales, síncronas o asíncronas con un cliente. Genera una respuesta, independientemente de las aplicaciones o lenguajes empleados, del lado del cliente.

Un servidor web se mantiene constantemente a la espera de nuevas peticiones de sus clientes o usuarios de internet y se encarga de dar respuesta a cada petición suya.

El servidor de datos espaciales se encarga de devolver información a los clientes sobre todo lo que se relacione con los datos espaciales y el sistema de información geográfica. Existen varios ejemplos y tipos de servidores web de datos geospaciales,

algunos de ellos son: MapServer, deegree, QGIS Server y GeoServer, entre otros (Morales, 2012).

En el presente trabajo de investigación se centrará únicamente en el análisis a fondo de las características del servidor GeoServer por ser el que se empleará en la institución.

#### **1.4.5.1 GeoServer**

GeoServer es un servidor de código abierto de datos geográficos basado en Java que permite a los usuarios ver y editar los datos espaciales. Trabaja con los estándares de la Open Geospatial Consortium (OGC) y permite una gran flexibilidad para la creación de mapas y para compartir información. Permite crear mapas en una gran cantidad de formatos e integra la librería de mapeo OpenLayers lo que permite la generación de mapas forma más rápida y fácil. GeoServer puede mostrar datos en muchas de las aplicaciones de mapeo como son: Google Maps, Google Earth, Microsoft Virtual Earth. (GeoServer, 2014).

GeoServer es software libre, desarrollado y probado por un gran número de personas y organizaciones alrededor del mundo. El hecho de ser de código abierto le da una ventaja extra, ya que los componentes de mejoramiento y solución de bugs en la comunidad de código abierto son mejores en cuanto a la rapidez en comparación a otras soluciones informáticas en el mundo. (Senplades, 2013) (Instituto Geográfico Militar del Ecuador, 2013).

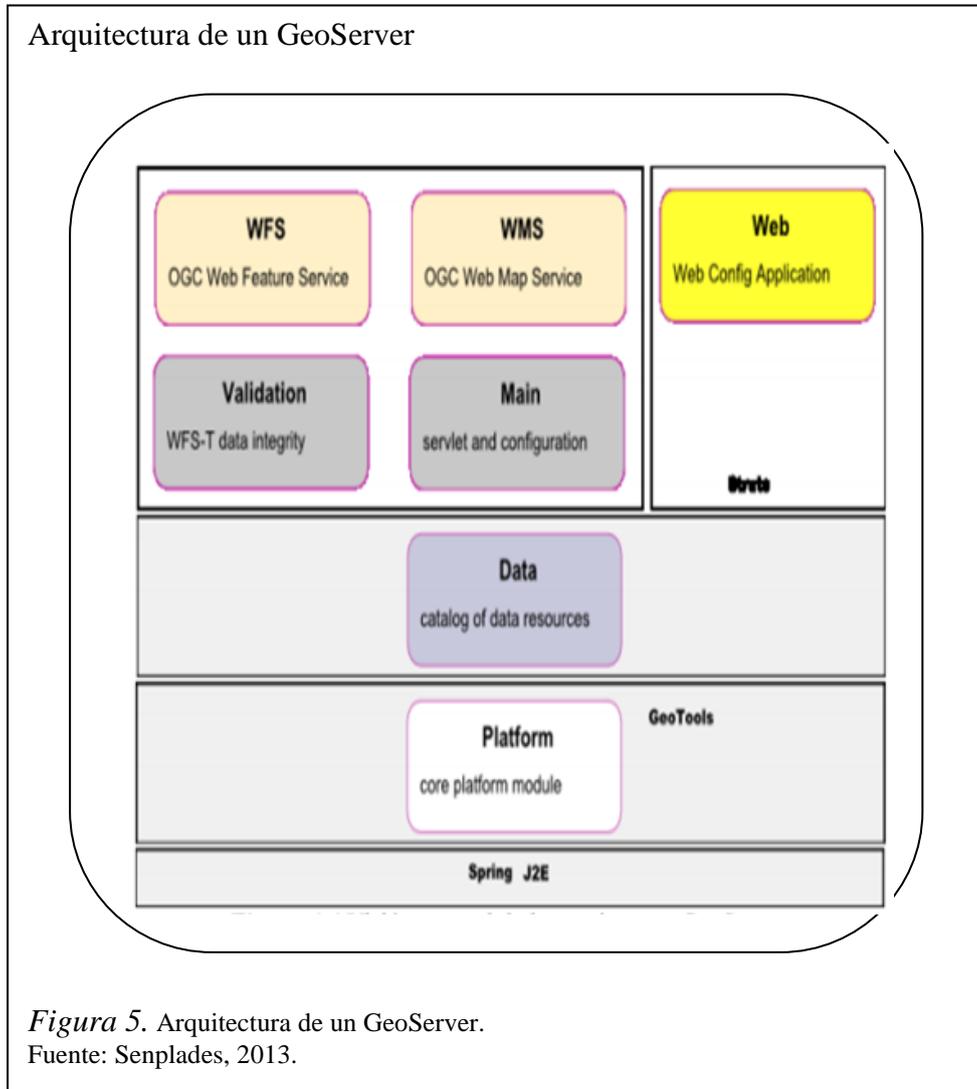
Es importante mencionar que GeoServer puede ejecutarse en cualquiera de los sistemas operativos como son Windows, Linux o Mac; de la misma forma posee una interfaz web muy útil para el manejo de la configuración, constituido por ventanas y pestañas.

La estructura de GeoServer contiene los siguientes niveles:

- Workspace o Namespace: espacio de trabajo que permite hacer agrupaciones lógicas de los elementos, aquí se almacenan: DataStores (conexión a BBDD, postgis etc.), FeatureTypes, estos son los fenómenos modelados en las distintas fuentes de datos y que se pueden utilizar como origen para definir

capas en un mapa de esta forma se obtiene la dirección de los wms openlayers.

A continuación se representa la arquitectura de un GeoServer:



*Figura 5.* Arquitectura de un GeoServer.

Fuente: Senplades, 2013.

Finalmente se presenta una imagen del entorno de un GeoServer, su interfaz administrativa:

## Plataforma Administrativa de GeoServer



*Figura 6.* Plataforma Administrativa de GeoServer  
Elaborado por: Jessica Escudero & Natali Guatapi

### 1.4.6 Bases de datos.

#### 1.4.6.1 PostgreSQL

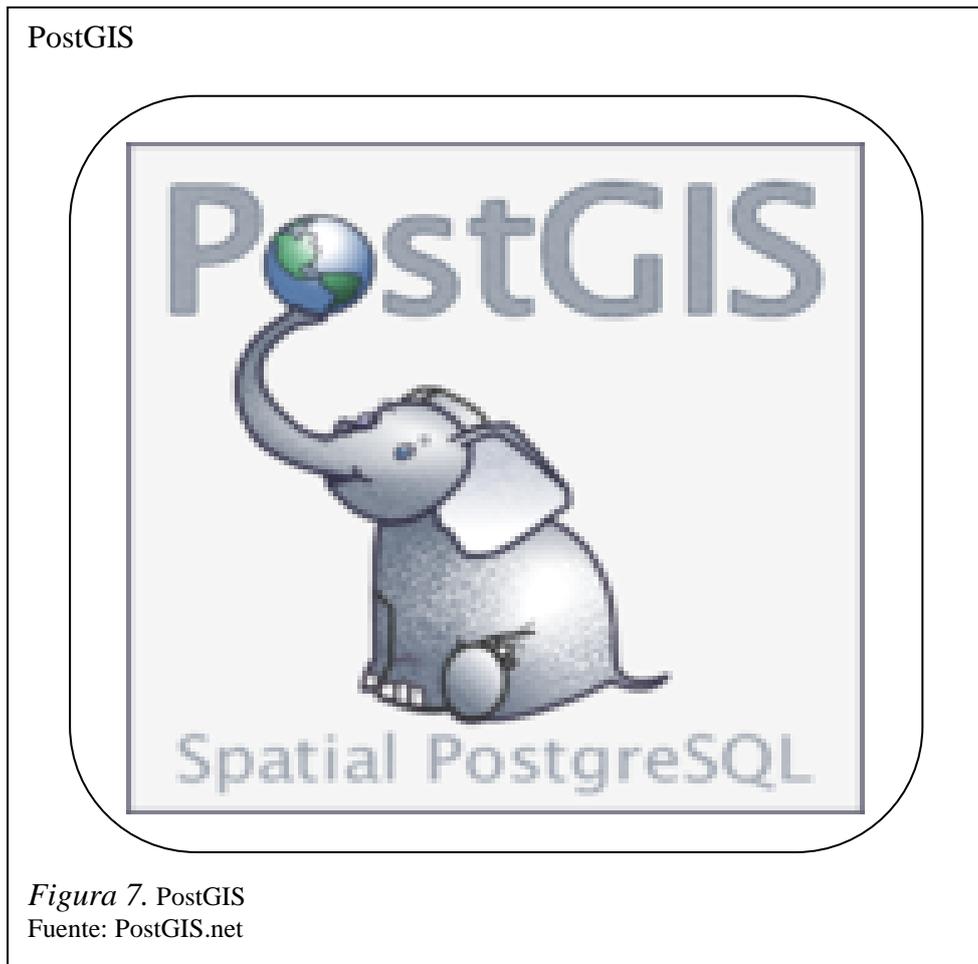
PostgreSQL es una base de datos objeto – relacional de código abierto compatible con SQL, es un potente motor que trabaja bajo la licencia BSD, por lo que se considera una arquitectura confiable, exacta e íntegra. Es capaz de correr en cualquiera de los sistemas operativos como son Windows, Linux, Unix.

Con PostgreSQL se puede generar y gestionar bases de datos, crear usuarios, mantener el servidor y todas las tareas relacionadas con la administración de este tipo de información.

PostgreSQL utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando (Martinez, 2010).

### 1.4.6.2 PostGIS

Es una extensión de base de datos espaciales que se puede utilizar con PostgreSQL, convirtiéndola en una base de datos espacial para su utilización en un sistema de información geográfica. Almacena objetos GIS como puntos, líneas, polígonos, multilíneas, multipuntos, y colecciones geométricas (Martín, 2013).



Adicionalmente el componente PostGIS adiciona a PostgreSQL funciones, operadores y mejoras en los índices que pueden aplicarse a los objetos GIS que aporta, donde la combinación de ambos es una solución perfecta para el almacenamiento, gestión y mantenimiento de los datos. Estas funciones, operadores e índices hacen de PostgreSQL una base de datos más eficiente haciéndola más rápida y con mejores características para manejar el sistema de datos espaciales (PostGIS, 2014).

Debido a que esta herramienta está construida sobre PostgreSQL, hereda todas sus características, así como sus estándares de código libre. Algunas de estas características son:

- Herramienta libre con licencia GNU.
- Es aplicada a los datos, índices y funciones espaciales.
- Importa y exporta datos mediante varias herramientas conversoras.
- Supera al software propietario en estabilidad y rapidez.
- Hoy en día es el gestor de base de datos de código abierto más utilizado.
- Es elegida por muchas empresas para asegurar su correcto funcionamiento en las bases de datos.

#### **1.4.6.2.1 Funcionamiento de PostGIS**

- **Arquitectura Cliente-Servidor**

PostGIS generalmente trabaja como servidor en un sistema cliente-servidor, de la manera en que el cliente realiza una petición al servidor y obtiene una respuesta. Estas peticiones se realizan en lenguaje SQL y la respuesta es generalmente una tabla de datos procedente de la base de datos generada por este gestor.

- **Creación de una base de datos espacial**

Este componente organiza su trabajo en base de datos separadas por lo que cada una trabaja independientemente con sus propias tablas y usuarios. Es por ello que al conectarse con una base de datos PostGIS se debe definir a qué base de datos se quiere acceder.

- **Consultas simples y espaciales**

Para seleccionar datos de una tabla PostGIS se pueden aplicar todas las operaciones SQL comunes y para realizar las consultas espaciales se utilizan las funciones con carácter espacial que se encuentran en la sección documental de esta herramienta.

#### **1.4.6.3 Geodatabase**

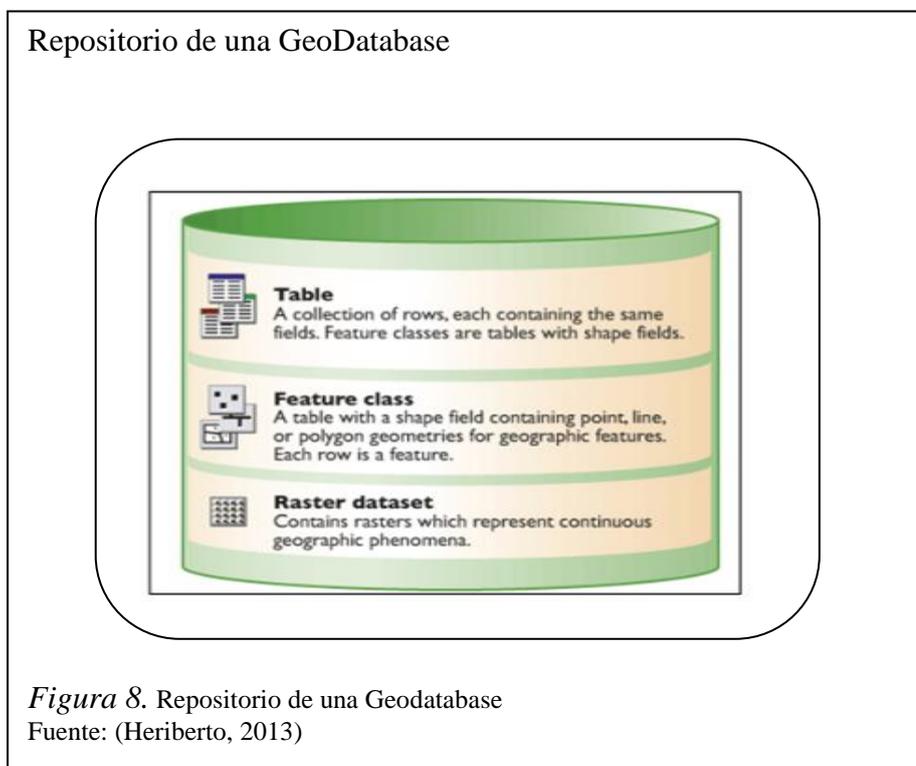
Geodatabase es un modelo que brindan herramientas de software dedicadas a administrar, gestionar y permitir el almacenamiento físico de datos espaciales. Estos datos son almacenados en archivos dentro de un Sistema Gestor de Base de Datos (Microsoft Access, Oracle, Microsoft SQL Server, IBM DB2).

Geodatabase es más robusto y extensible al tipo de datos en comparación con shapefiles y coberturas, asimismo permite definir los elementos geográficos de modo que sean creados más cerca al mundo real y es compatible con todos los diferentes elementos de datos GIS.

Geodatabase permite albergar en su interior clases de identidad como Puntos, Líneas, polígonos, tablas y agrupar en una colección de clases de entidad llamada dataset de entidades esto admite separar por grupos los datos espaciales con un mismo sistema de coordenadas; además tiene la habilidad de almacenar dos tipos de información en una sola base de datos estos tipos son:

- File Geodatabase: tiene una capacidad mayor a 2GB que permite conectarse con diferentes bases de datos o puede trabajar con un servidor.
- Personal Geodatabase: se usa para aplicaciones pequeñas individuales. (Llopis, 2008)

Geodatabase es aquella que agrupa la información geográfica, atributos y relaciones topológicas en un solo repositorio como esta en la siguiente figura:



*Figura 8.* Repositorio de una Geodatabase  
Fuente: (Heriberto, 2013)

El primer paso en la operación de una Geodatabase es generar los conjuntos de datos de “dataset” de las diferentes clases de entidades, los del tipo raster y las tablas.

#### **1.4.7 Lenguajes de programación, herramientas y servidores**

A continuación se detallan los lenguajes de programación, herramientas y servidores que serán utilizados en la implementación del sistema.

**JAVA:** es un lenguaje de programación orientado a objetos en el cual las aplicaciones son compiladas a bytecode que puede correr en cualquier máquina virtual. Es simple y portátil sobre diferentes plataformas y sistemas operativos, ya sea a nivel de código fuente como a nivel de código binario (Java, 2006).

**PHP:** es un lenguaje de script interpretado en el lado del servidor utilizado para la generación de páginas web dinámicas, similar al ASP de Microsoft o el JSP de Sun, embebido en páginas HTML y ejecutado en el servidor. La mayor parte de su sintaxis ha sido tomada de C, Java y Perl con algunas características específicas de sí mismo. La meta del lenguaje es permitir rápidamente a los desarrolladores la generación dinámica de páginas (PHP, 2014).

**XML:** es un método flexible para crear formatos de información comunes. Una vez creado el formato de la información, tanto el formato como los datos se pueden compartir vía web o con otros programas y bases de datos. El lenguaje XML es parecido al lenguaje HTML (Lenguaje de marcado de hipertexto) ya que tanto XML como HTML contienen etiquetas de marcado para describir el contenido de una página o un archivo. Un archivo XML se puede procesar meramente como datos en un programa o almacenarse con datos parecidos en otro equipo (Microsoft, 2010).

**HTML:** es el código con el que se escriben las páginas web fundamentalmente e indica básicamente dónde colocar cada texto, cada imagen o cada video y la forma de visualización que tendrá cada uno de estos elementos al ser colocados en la página. Otra característica fundamental del lenguaje es el uso de etiquetas con la forma < >, donde cada etiqueta tiene un uso específico dentro de la página (Desarrollo-Web, 2002).

**JSP:** el lenguaje Java Server Pages, por su acrónimo en inglés es una tecnología JAVA que permite generar contenido dinámico para la web en forma de documentos HTML, XML, entre otros formatos. Permite la utilización de código JAVA mediante scripts, además de que es posible utilizar algunas funciones JSP predefinidas mediante etiquetas. JSP se puede ejecutar en cualquiera de los sistemas operativos y servidores web más populares como son Apache, Netscape o Microsoft IIS. Las páginas JSP son compilados en Servlets por lo que actúan como una puerta a todos los servicios de Java del servidor y librerías Java para aplicaciones HTTP.

**MAVEN:** es una herramienta de software para la gestión y construcción de proyectos Java.

Maven utiliza Project Object Model (POM) para describir el proyecto de software a construir, sus dependencias de otros módulos y componentes externos. Viene con objetivos predefinidos para realizar ciertas tareas claramente definidas, como la compilación del código y su empaquetado.

Esta herramienta está construida usando una arquitectura basada en plugins y permite que utilice cualquier aplicación controlable a través de la entrada estándar.

**APACHE:** “es un servidor HTTP de código fuente abierto y licenciamiento libre que funciona en Linux, sistemas operativos derivados de Unix, Windows, y otras plataformas.” (Dueñas, 2015)

**APACHE TOMCAT:** es una aplicación de software de código abierto de las tecnologías Java Servlet y JavaServer, es desarrollado en un entorno abierto y participativo. (Foundation, 2015)

#### **1.4.8 Metodología de desarrollo de software**

Una metodología de desarrollo de software se refiere al frameworks que es usado para estructurar, planear y controlar el proceso de desarrollo en sistemas de información. Estos frameworks son a menudo vinculados a algún tipo de organización, que además desarrolla, apoya el uso y promueve la metodología.

Las metodologías se basan en una combinación de los modelos de proceso genéricos como son: cascada, evolutivo, incremental, espiral, iterativo, entre otros. Adicionalmente una metodología debe definir con precisión los artefactos, roles y actividades involucrados junto a las prácticas y técnicas recomendadas, guías de adaptación de la metodología al proyecto, guías para uso de herramientas de apoyo y otros.

La comparación y/o clasificación de metodologías no es una tarea sencilla debido a la diversidad de propuestas y diferencias en el grado de detalle, información disponible y alcance de cada una de ellas. Es por ello que para la selección de la metodología a desarrollar en la presente investigación se hizo un estudio de los diferentes modelos o frameworks de metodologías existentes y se determinó hacer uso del modelo ágil (Autores, 2015).

Entre las metodologías ágiles para el desarrollo de software estudiadas destaca la metodología SCRUM, finalmente la seleccionada para el presente proyecto.

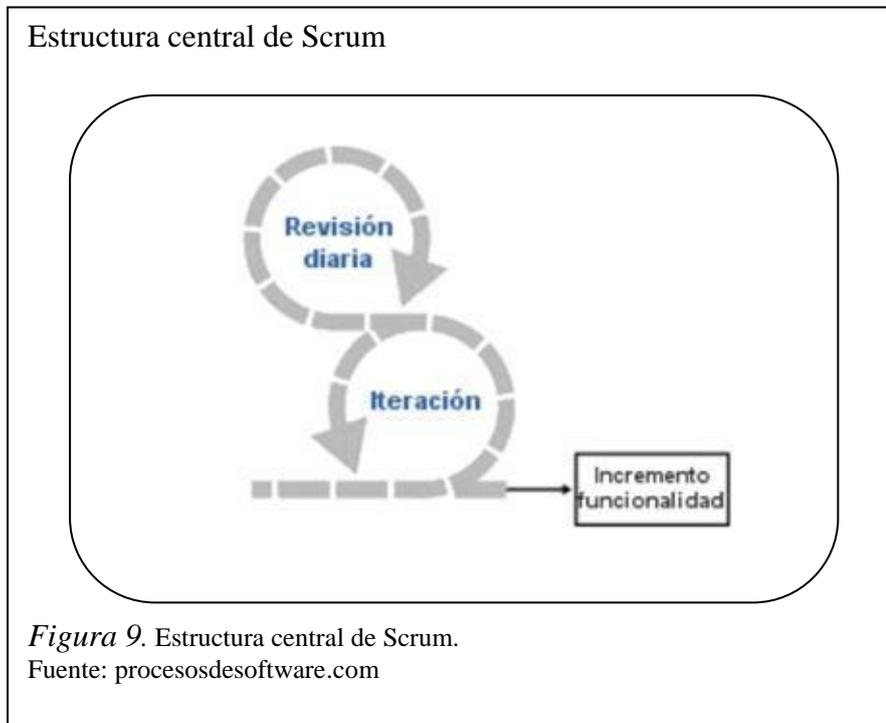
#### **1.4.8.1 Metodología SCRUM**

Scrum es un proceso en el que se aplican de manera regular un conjunto de mejores prácticas para trabajar colaborativamente en equipo y así obtener el mejor resultado posible en un proyecto. Estas prácticas se apoyan unas a otras y su selección tiene origen en un estudio de la manera de trabajar de equipos altamente productivos.

La característica fundamental de Scrum son las entregas parciales y regulares del producto, donde desde un principio se priorizan las de mayor relevancia o importancia para el desarrollo. Es por esta característica que Scrum está especialmente indicado para proyectos en entornos complejos, donde se necesita obtener resultados pronto; en los que los requisitos son cambiantes o poco definidos; donde la innovación, la competitividad, la flexibilidad y la productividad son fundamentales (IBM, 2015).

Otra característica de Scrum es que es utilizado en ciertas situaciones problemáticas del desarrollo como: en los casos en los que no se está entregando al cliente lo que necesita; cuando las entregas se alargan demasiado, los costes se disparan o la

calidad no es aceptable; cuando se necesita capacidad de reacción ante la competencia; cuando la moral de los equipos es baja y la rotación alta; cuando es necesario identificar y solucionar ineficiencias sistemáticamente o cuando se quiere trabajar utilizando un proceso especializado en el desarrollo de producto. A continuación se presenta el modelo central de trabajo de Scrum.



En Scrum un proyecto se ejecuta en bloques temporales cortos y fijos por iteraciones (o sprint), donde cada iteración tiene que proporcionar un resultado completo o sea el incremento de producto final que sea susceptible de ser entregado con el mínimo esfuerzo al cliente cuando lo solicite. El proceso parte de la lista de objetivos o requisitos del producto. Esta lista es definida por el cliente, donde debe priorizar los requisitos en dependencia del valor que aportan al sistema para de esta forma distribuirlos en las iteraciones y entregas.

### **Actividades en SCRUM**

Las tres actividades básicas en Scrum son:

La actividad de Planificación de la iteración se realiza el primer día de cada iteración y se compone de dos partes. La primera de ellas es la selección de los requisitos, donde es presentada por el cliente la lista de requisitos prioritarios del producto. La segunda parte es cuando se planifica la iteración, en la cual se elaboran las tareas que

deberán ser cumplidas por el equipo para desarrollar los requisitos, las mismas que son auto asignadas por los miembros del equipo.

Durante la actividad Ejecución de la iteración se realizan reuniones diarias de sincronización. En estas reuniones se inspecciona el trabajo de cada miembro del equipo y es donde se analizan las dependencias entre las tareas de cada miembro, el progreso que se está dando en función de los objetivos, y los obstáculos que puedan impedir el cumplimiento de los objetivos.

Finalmente la actividad de Inspección de la iteración se realiza el último día de la iteración y se lleva a cabo la reunión de revisión de la iteración. Esta actividad se compone de dos partes: la demostración y la retrospectiva. En la demostración es donde el equipo presenta a su cliente los requisitos completados en la iteración. En la segunda parte de retrospectiva, es donde el equipo analiza cómo ha sido su desempeño durante la iteración y se identifican en base a ello los problemas que puedan impedir continuar progresando en la construcción del sistema, mejorando así su rendimiento y productividad de manera continua.

### **Artefactos de Scrum**

En Scrum existen tres artefactos principales: la lista de objetivos/requisitos priorizada (product Backlog), la lista de tareas de la iteración (sprint Backlog) y los gráficos de trabajo pendiente (Burndown Chart).

El Product Backlog representa la visión y expectativas del cliente respecto a los objetivos y entregas del producto o proyecto. El cliente es el responsable de crear y gestionar la lista de requisitos con la ayuda del Facilitador y del equipo. Esta lista permite involucrar al cliente en la dirección de los resultados del producto o proyecto.

La lista de tareas se elabora en la reunión de planificación de cada iteración como el plan para dar cumplimiento a los objetivos de la iteración. La lista permite además revisar las tareas donde se presentan problemas al equipo para avanzar y de esta forma tomar las medidas pertinentes.

Para resumir, esta metodología facilita el manejo para administrar el desarrollo de software adaptable, fundamental para construir la funcionalidad de mayor valor para el cliente y se basa en un conjunto de valores, principios y prácticas. Scrum permite, acceder, mejorar y realizar mantenimiento de un sistema existente o la producción de nuevos prototipos. Se optó a escoger Scrum debido a que es una metodología ágil para el desarrollo de proyectos y este no es un proyecto que se reconozcan los requisitos de manera inmediata ni fija.

## CAPÍTULO 2

### ANÁLISIS

#### 2.1 Especificaciones de la Metodología Scrum para el proyecto

Durante el desarrollo de este capítulo se definieron los roles de la metodología Scrum entre los más importantes se distingue los siguientes: Scrum Master, Producto Owner y development team. Seguidamente se definen los objetivos/requerimientos priorizados del proyecto y finalmente se definirán los artefactos y las actividades a realizar para el desarrollo del proyecto.

##### 2.1.1 Roles

En primer lugar se definirá los roles que se manejarán en el proyecto, los mismos son descritos a continuación:

Tabla 1. *Roles del proyecto*

Roles/personas	Product Owner	Scrum Master	Developerment Team
Personas encargadas	Ing. Gustavo Navas	Jessica Escudero, Natali Guatapi	Jessica Escudero, Natali Guatapi

Nota. Roles de las personas encargadas para la elaboración del proyecto.

Elaborado por: Jessica Escudero & Natali Guatapi.

#### Dueño del producto (Product Owner)

En el presente trabajo será el encargado de mantener una relación directa con el equipo de desarrollo, es quien entiende las necesidades y prioridades necesarias para el producto. Trabaja en conjunto con el Scrum master y el equipo de desarrollo para hacer llegar y entender las necesidades y prioridades de los usuarios. Es también el encargado de manejar los criterios de aceptación para las características del producto.

### **Facilitador (Scrum master)**

El facilitador estará representado por el Ing. Navas quien deberá guiar al equipo de desarrollo en el cumplimiento de los objetivos, eliminando los obstáculos que puedan surgir durante el desarrollo e incidirá en la selección de materiales y tecnologías a utilizar.

### **Equipo**

El equipo de desarrollo estará integrado por Jessica Escudero y Natali Guatapi, quienes se encargarán de identificar los objetivos y tareas individuales en cada iteración o sprint y deberán responder por su cumplimiento al final de cada iteración.

### **2.1.2 Identificación de los requisitos funcionales**

A continuación se presenta la lista de requerimientos funcionales del producto.

Tabla 2. *Requerimientos funcionales del producto*

No.	Requisito Funcional	Prioridad
1	Seleccionar casa salesiana	1
2	Seleccionar tipo de obra de acuerdo a la casa salesiana.	1
3	Consultar obra.	1
4	Definir cantidad de obras encontradas de manera numérica de acuerdo a la casa salesiana señalada.	2
5	Seleccionar estilos de mapas.	1
6	Maximizar mapa	3
7	Minimizar mapas	3
8	Marcar en el mapa la obra salesiana buscada	1
9	Seleccionar obra en mapa	2
10	Brindar posición de la obra seleccionada	2
11	Brindar información de la obra seleccionada	2
12	Brindar información geográfica de la obra seleccionada mediante los visualizadores GeoTools y GeoPHP	2 3
13	Colocar pantalla completa del mapa	3
14	Salir de la pantalla completa del mapa	3

15	Seleccionar área	2
----	------------------	---

Nota. Requerimientos Funcionales con cada una de las prioridades.

Elaborado por: Jessica Escudero & Natali Guatapi.

Luego de presentarse la lista de requisitos funcionales del producto se procede a su especificación a través de historias de usuario. Antes de continuar se debe aclarar que para esta investigación, una historia de usuario representa un requerimiento de software escrito en una o dos frases y utilizando el lenguaje común del usuario.

En la presente investigación ha quedado asignado un objetivo a cada historia de usuario. A continuación se detallan las especificaciones de estas.

Tabla 3. *Historias de usuario*

<b>Historias de Usuario</b>	
<b>ID</b>	HU1
<b>Título</b>	Seleccionar casa salesiana
<b>Descripción</b>	En este caso de uso se pretende que el usuario seleccione la casa salesiana que necesita buscar en el mapa para determinar sus características.
<b>Dependencias</b>	Ninguna.
<b>Prioridad</b>	Alta
<b>Visualizadores</b>	GeoServer, GeoTools, GeoPHP
<b>ID</b>	HU2
<b>Título</b>	Seleccionar tipo de obra.
<b>Descripción</b>	En este caso de uso el usuario cuando selecciona la casa salesiana anteriormente, automáticamente se despliegan los tipos de obras que pertenecen a la misma donde debe marcar cuál es la que desea que sea buscada en el mapa para determinar sus características.
<b>Dependencias</b>	Caso de uso 1
<b>Prioridad</b>	Alta
<b>Visualizadores</b>	GeoServer, GeoTools, GeoPHP
<b>ID</b>	HU3

<b>Título</b>	Consultar obra.
<b>Descripción</b>	En este caso de uso se pretende que el usuario una vez seleccionado la casa salesiana y el tipo de obra presione consultar para que el sistema haga la búsqueda de la misma en el mapa.
<b>Dependencias</b>	Ninguna
<b>Prioridad</b>	Alta
<b>Visualizadores</b>	GeoServer, GeoTools, GeoPHP
<b>ID</b>	HU4
<b>Título</b>	Definir cantidad de obras encontradas de manera numérica.
<b>Descripción</b>	En este caso de uso se pretende que el usuario observe de manera numérica la cantidad de obras encontradas de acuerdo a la casa salesiana señalada anteriormente.
<b>Dependencias</b>	Caso de uso 1
<b>Prioridad</b>	Media
<b>Visualizadores</b>	GeoServer, GeoTools, GeoPHP
<b>ID</b>	HU5
<b>Título</b>	Seleccionar estilos de mapas.
<b>Descripción</b>	En este caso de uso se pretende que el usuario seleccione el estilo de mapa que desea utilizar para realizar la búsqueda.
<b>Dependencias</b>	Ninguna
<b>Prioridad</b>	Alta
<b>Visualizadores</b>	GeoServer, GeoTools, GeoPHP
<b>ID</b>	HU6
<b>Título</b>	Maximizar mapa
<b>Descripción</b>	En este caso de uso se pretende que el usuario una vez seleccionado el mapa con que desea realizar la búsqueda lo maximice en caso de desee observar lo buscado en un radio de acción mayor.
<b>Dependencias</b>	Ninguna
<b>Prioridad</b>	Baja

<b>Visualizadores</b>	GeoServer, GeoTools, GeoPHP
<b>ID</b>	HU7
<b>Título</b>	Minimizar mapas
<b>Descripción</b>	En este caso de uso se pretende que el usuario una vez seleccionado el mapa con que desea realizar la búsqueda lo minimice en caso de desee observar lo buscado en un radio de acción menor.
<b>Dependencias</b>	Ninguna
<b>Prioridad</b>	Baja
<b>Visualizadores</b>	GeoServer, GeoTools, GeoPHP
<b>ID</b>	HU8
<b>Título</b>	Mostrar en el mapa la obra buscada de acuerdo a la casa salesiana señalada.
<b>Descripción</b>	En este caso de uso se pretende que el usuario una vez seleccionado la casa salesiana y el tipo de obra que desea buscar presione consultar para que el sistema realice la búsqueda. Al realizar esta operación se marcará en el mapa la obra seleccionada de acuerdo a la casa salesiana señalada.
<b>Dependencias</b>	Caso de uso 3
<b>Prioridad</b>	Alta
<b>Visualizadores</b>	GeoServer, GeoTools, GeoPHP
<b>ID</b>	HU9
<b>Título</b>	Seleccionar obra en mapa
<b>Descripción</b>	En este caso de uso se pretende que el usuario una vez mostrado en el mapa la obra buscada de acuerdo a la casa salesiana señalada, debe seleccionarla para posteriormente visualizar su información
<b>Dependencias</b>	Caso de uso 8
<b>Prioridad</b>	Media
<b>Visualizadores</b>	GeoServer, GeoTools, GeoPHP

<b>ID</b>	HU10
<b>Título</b>	Brindar posición de la obra seleccionada
<b>Descripción</b>	En este caso de uso se pretende que el usuario una vez seleccionada el mapa la obra buscada, de acuerdo a la casa salesiana señalada, debe brindar la posición exacta de la misma.
<b>Dependencias</b>	Caso de uso 9
<b>Prioridad</b>	Media
<b>Visualizadores</b>	GeoServer, GeoTools, GeoPHP
<b>ID</b>	HU11
<b>Título</b>	Brindar información de la obra seleccionada.
<b>Descripción</b>	En este caso de uso se pretende que el usuario una vez seleccionado la casa salesiana y el tipo de obra se brinde en la interfaz del sistema la información de la misma en función del horario, nombre del lugar, servicios ofrecidos, dirección, página web disponible y responsable del lugar
<b>Dependencias</b>	Caso de uso 9
<b>Prioridad</b>	Media
<b>Visualizadores</b>	GeoServer, GeoTools, GeoPHP
<b>ID</b>	HU12
<b>Título</b>	Brindar información geográfica de la obra seleccionada.
<b>Descripción</b>	El usuario una vez seleccionado la casa salesiana y el tipo de obra se visualiza en la interfaz del sistema la información geográfica en función del área que ocupa y su dimensión.
<b>Dependencias</b>	Caso de uso 9
<b>Prioridad</b>	Media
<b>Visualizadores</b>	GeoPHP, GeoTools (En el visualizador prototipo de GeoServer no se implementó librerías ya que es un servidor de mapas y se trabaja por capas)
<b>ID</b>	HU13
<b>Título</b>	Colocar pantalla completa del mapa

<b>Descripción</b>	En este caso de uso se pretende que el usuario al trabajar en el mapa seleccionado, utilice la opción de colocar la pantalla completa del mismo para solamente tenerlo en la interfaz del sistema
<b>Dependencias</b>	Ninguna
<b>Prioridad</b>	Baja
<b>Visualizadores</b>	GeoServer, GeoTools, GeoPHP
<b>ID</b>	HU14
<b>Título</b>	Salir de la pantalla completa del mapa
<b>Descripción</b>	En este caso de uso se pretende que el usuario una vez entrado en la pantalla completa del sistema desee salir para seguir con la interfaz anterior.
<b>Dependencias</b>	Caso de uso 13
<b>Prioridad</b>	Baja
<b>Visualizadores</b>	GeoServer, GeoTools, GeoPHP
<b>ID</b>	HU15
<b>Título</b>	Seleccionar área
<b>Descripción</b>	En este caso de uso se pretende que el usuario seleccione el área en el mapa para determinar sus características.
<b>Dependencias</b>	Caso de uso 8
<b>Prioridad</b>	Alta
<b>Visualizadores</b>	GeoTools, GeoPHP

Nota. Especificaciones de los requisitos funcionales mediante historias de usuario.

Elaborado por: Jessica Escudero & Natali Guatapi.

### 2.1.3 Planes de lanzamiento

Como primera actividad a desarrollar en la metodología Scrum se definen las iteraciones que tendrá el desarrollo del presente proyecto. El desarrollo del proyecto contará con 4 iteraciones donde se definirán objetivos por cada iteración y tareas

específicas para cada miembro del equipo. A continuación se detallan las iteraciones los objetivos y el responsable de dar cumplimiento:

Tabla 4. *Iteraciones realizadas en los prototipos*

Iteración	Objetivos presentes	Responsable
<b>1<sup>ra</sup> Iteración</b>	Seleccionar casa salesiana	Jessica Escudero
	Seleccionar tipo de obra.	Natali Guatapi
	Consultar obra.	Natali Guatapi
	Definir cantidad de obras encontradas de manera numérica.	Jessica Escudero
<b>2<sup>da</sup> Iteración</b>	Seleccionar estilos de mapas.	Natali Guatapi y Jessica Escudero
	Maximizar mapa	Jessica Escudero
	Minimizar mapas	Natali Guatapi
	Seleccionar obra en el mapa	Jessica Escudero
	Seleccionar área perteneciente a la obra elegida	Natali Guatapi
<b>3<sup>ra</sup> Iteración</b>	Brindar posición de la obra seleccionada	Jessica Escudero
	Brindar información de la obra seleccionada	Natali Guatapi
	Brindar información geográfica de la obra	Jessica Escudero y Natali Guatapi

	seleccionada para los prototipos Geo PHP y GeoTools	
4 <sup>ta</sup> Iteración	Colocar pantalla completa del mapa	Jessica Escudero y Natali Guatapi
	Salir de la pantalla completa del mapa	Jessica Escudero y Natali Guatapi
	Marcar en el mapa la obra buscada de acuerdo a la casa salesiana señalada.	Jessica Escudero y Natali Guatapi

Nota. Se definen las iteraciones para el desarrollo del proyecto.

Elaborado por: Jessica Escudero & Natali Guatapi.

#### 2.1.4 Identificación de los requerimientos no funcionales de los visualizadores

Primeramente se debe decir que los requerimientos no funcionales describen aspectos del sistema que son visibles al usuario pero que no incluyen una relación directa con el comportamiento funcional del mismo. A continuación se presenta la lista de requerimientos no funcionales definidos para el presente proyecto.

**RNF1** Tiempos de respuesta: los visualizadores no excederán de los 2 minutos para cargar la información solicitada por el usuario.

**RNF2** Entorno de ejecución: está determinado por el propósito del visualizador prototipo y el uso de la librería, modelo de base de datos o servidor que esté en uso.

**RNF3** Confiabilidad: los datos devueltos por los visualizadores son totalmente confiables y pueden ser constatados según su veracidad.

**RNF4** Seguridad: el sistema estará protegido contra ataques de hackers y programas maliciosos ya que los datos devueltos por los visualizadores son en muchos casos confidenciales. Solo los usuarios con los permisos requeridos podrán acceder al visualizador y a la información que devuelven.

**RNF5 Recursos consumidos:** los recursos que consumirán los visualizadores en su gestión de información geográfica deberán ser los menores para no colapsar el sistema.

**RNF6 Mantenibilidad:** los visualizadores son de fácil configuración y adaptación a nuevos escenarios o entornos. Ello permite realizar cambios oportunos en las funcionalidades de los visualizadores para obtener un mejor rendimiento de estos.

**RNF7 Usabilidad:** mediante el uso del ratón o mouse, el usuario podrá interactuar con el visualizador para acercar, alejar o girar las imágenes obtenidas de este. Así como para acceder a las diferentes opciones y características que ofrece el sistema podrá usar el ratón o el teclado.

### **2.1.5 Distribución, revisión y ajustes de los estándares del producto**

En la presente propuesta se analizarán los estándares presentes en los visualizadores que se desarrollarán pertenecientes a la Open Geospatial Consortium OGC. Estos estándares son el Keyhole Markup Language (KML) y el Wep Map Service (WMS). Estos estándares son utilizados para el trabajo directo con los mapas.

#### **WMS**

El Servicio de publicación de mapas (WMS) produce mapas de forma dinámica a partir de información geográfica vectorial o raster presentando la información como imágenes digitales susceptibles de ser visualizadas en pantalla. La visualización de la imagen suele ser en formato raster: PNG, GIF o JPEG y ocasionalmente se representan como información vectorial en formato Scalable Vector Graphics (SVG) o Web Computer Graphics Metafile (WebCGM).

### **2.1.6 Sprint**

Luego de haber analizado toda la información pertinente a los requisitos y el uso de estándares en el proyecto y teniendo toda la información requerida para organizar el trabajo se procede a analizar las iteraciones o sprint que se realizarán en el presente proyecto.

Como se ha visto, el proyecto ha quedado distribuido en 4 iteraciones o sprints, de las cuales las dos primeras son las de mayor importancia pues se les dará cumplimiento a los objetivos de mayor prioridad y que deben cumplirse a toda costa por constituir la base fundamental del proyecto. Las dos últimas iteraciones serán importantes pero no tan necesarias para el proyecto. A continuación se muestra una tabla con el cronograma de trabajo por iteraciones.

Tabla 5. *Cronograma de trabajo por iteraciones*

Iteraciones (sprint)	Actividades x iteración	Fechas de cumplimiento
<b>Iteración 1</b>	Reunión de planificación de la Iteración 1	15/12/2014 Máximo 4 horas.
	Ejecución de la Iteración 1	16/12/2014 – 04/01/2015
	Reunión diaria de sincronización.	1 x cada día. Máximo 1 horas.
	Reunión de revisión de la iteración 1	04/01/2015 Máximo 3 horas.
	Reunión de presentación de los resultados de la Iteración (Sprint Review).	06/01/2015 Máximo 2 horas.
	<b>Iteración 2</b>	Reunión de planificación de la Iteración 2
Ejecución de la Iteración		06/01/2015 – 23/01/2015
Reunión diaria de sincronización		1 x cada día. Máximo 1 horas.
Reunión de revisión de la Iteración 2		23/01/2015 Máximo de 4 horas.
Reunión de presentación de los resultados de la Iteración 2 (2 <sup>da</sup> Sprint Review).		26/01/2015 Máximo 4 horas.
<b>Iteración 3</b>	Reunión de planificación de la Iteración 3	26/01/2015 Máximo 4 horas.
	Ejecución de la Iteración	26/01/2015 – 06/02/2015
	Reunión diaria de	1 x cada día

	sincronización	
	Reunión de revisión de la Iteración 3	06/02/2015 Máximo 3 horas.
	Reunión de presentación de los resultados de la Iteración 3 (3 <sup>ra</sup> Sprint Review)	09/02/2015 Máximo 3 horas.
<b>Iteración 4</b>	Reunión de planificación de la Iteración 4.	09/02/2015 Máximo 2 horas
	Ejecución de la Iteración	10/02/2015 – 20/02/2015
	Reunión diaria de sincronización.	1 x cada día
	Reunión de revisión de la Iteración 4.	23/02/2015 Máximo de 2 horas.
	Reunión de presentación de los resultados de la Iteración 4 (4 <sup>ta</sup> Sprint Review).	23/02/2015 Máximo de 4 horas.
<b>Revisión y Pruebas</b>	Revisión del cumplimiento de los objetivos del proyecto y elaboración del informe final.	23/02/2015 – 28/02/2015
<b>Cierre</b>	Presentación de los resultados del proyecto.	02/03/2015

Nota. Cronograma de actividades para la elaboración del proyecto.

Elaborado por: Jessica Escudero & Natali Guatapi.

A modo de explicación se debe añadir que el cronograma elaborado está sujeto a cambios de última hora como suele suceder en los proyectos de este tipo desarrollados con Scrum debido a replanteamiento de los objetivos por el cliente, dificultades en el entendimiento de estos objetivos por el equipo de desarrollo así como factores externos que pueda afectar el buen desenvolvimiento del equipo.

Destacar además que se le brindará un mayor margen de tiempo para el desarrollo de las 2 primeras iteraciones por considerarlas las de mayor peso en la presente investigación y donde puedan surgir las mayores dificultades en el desarrollo. Las

dos últimas iteraciones tienen un menor margen de tiempo para su desarrollo debido a que no necesitan de mayor tiempo para ejecutarse.

### **2.1.7 Cierre**

Al finalizar las iteraciones y la implementación de todos los objetivos propuestos por el cliente, luego de haberse realizado los cambios al proyecto y a los objetivos que puedan surgir durante el desarrollo y tener el proyecto listo para su entrega final se realizará una reunión de aceptación con el cliente que definirá la entrega del producto final. A partir de este momento solo quedará la implementación del proyecto en el entorno real y la elaboración y entrega de toda la documentación generada y de los manuales de usuario del o de los visualizadores que sean aprobados para su uso.

## **CAPÍTULO 3**

### **DISEÑO Y CONSTRUCCIÓN**

#### **3.1 Diseño de prototipos**

Los prototipos son aquellos que son desarrollados para representar un sistema informático creado para ser utilizado por los usuarios finales.

Con el fin de diseñar los prototipos desarrollados se definieron los casos de uso del sistema los cuales se muestran a continuación:

#### **Casos de usos del sistema para los prototipos GeoPHP y GeoTools:**

Diagrama de casos de uso para:

- Seleccionar casa salesiana
- Seleccionar tipo de obra
- Consultar obra.
- Visualizar numéricamente cantidad de obras encontradas
- Mostrar en el mapa la obra seleccionada
- Seleccionar obra en el mapa
- Seleccionar estilo de mapa.
- Seleccionar área
- Maximizar mapa
- Minimizar mapa
- Visualizar posición de la obra
- Visualizar información de la obra
- Colocar pantalla completa
- Salir de pantalla completa
- Visualizar información geográfica

#### **Casos de usos del sistema para los prototipos GeoServer**

Diagrama de casos de uso para GeoServer:

- Seleccionar casa salesiana
- Seleccionar tipo de obra

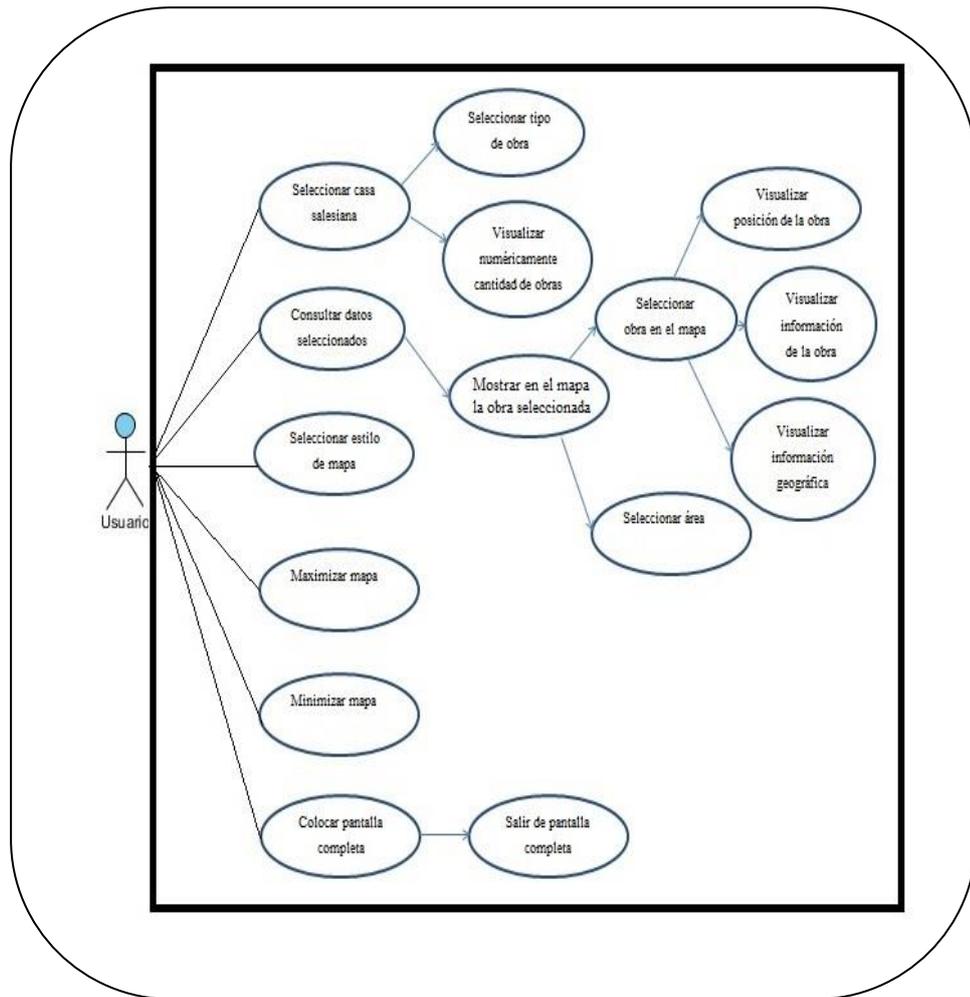
- Consultar obra.
- Visualizar numéricamente cantidad de obras encontradas
- Mostrar en el mapa la obra seleccionada
- Seleccionar obra en el mapa
- Seleccionar estilo de mapa.
- Maximizar mapa
- Minimizar mapa
- Visualizar posición de la obra
- Visualizar información de la obra
- Colocar pantalla completa
- Salir de pantalla completa

Para el diseño de los prototipos fue necesario desarrollar como complemento diagramas que determinaron el completo funcionamiento del sistema como es el caso de los diagramas de caso de uso.

Estos diagramas definen una notación gráfica para representar casos de usos mediante el lenguaje modelado unificado (UML).

A continuación se muestra el diagrama de casos de uso para los visualizadores GeoPHP y GeoTools, donde su actor es el usuario.

## Diagrama de casos de uso para los visualizadores GeoPHP y GeoTools



*Figura 10.* Diagrama de casos de uso para los visualizadores GeoPHP y GeoTools  
Elaborado por: Natali Guatapi & Jessica Escudero

- Usuario

**Descripción:** visualización de las obras salesianas

**Actores:** usuario

**Precondiciones:** acceder al visualizador

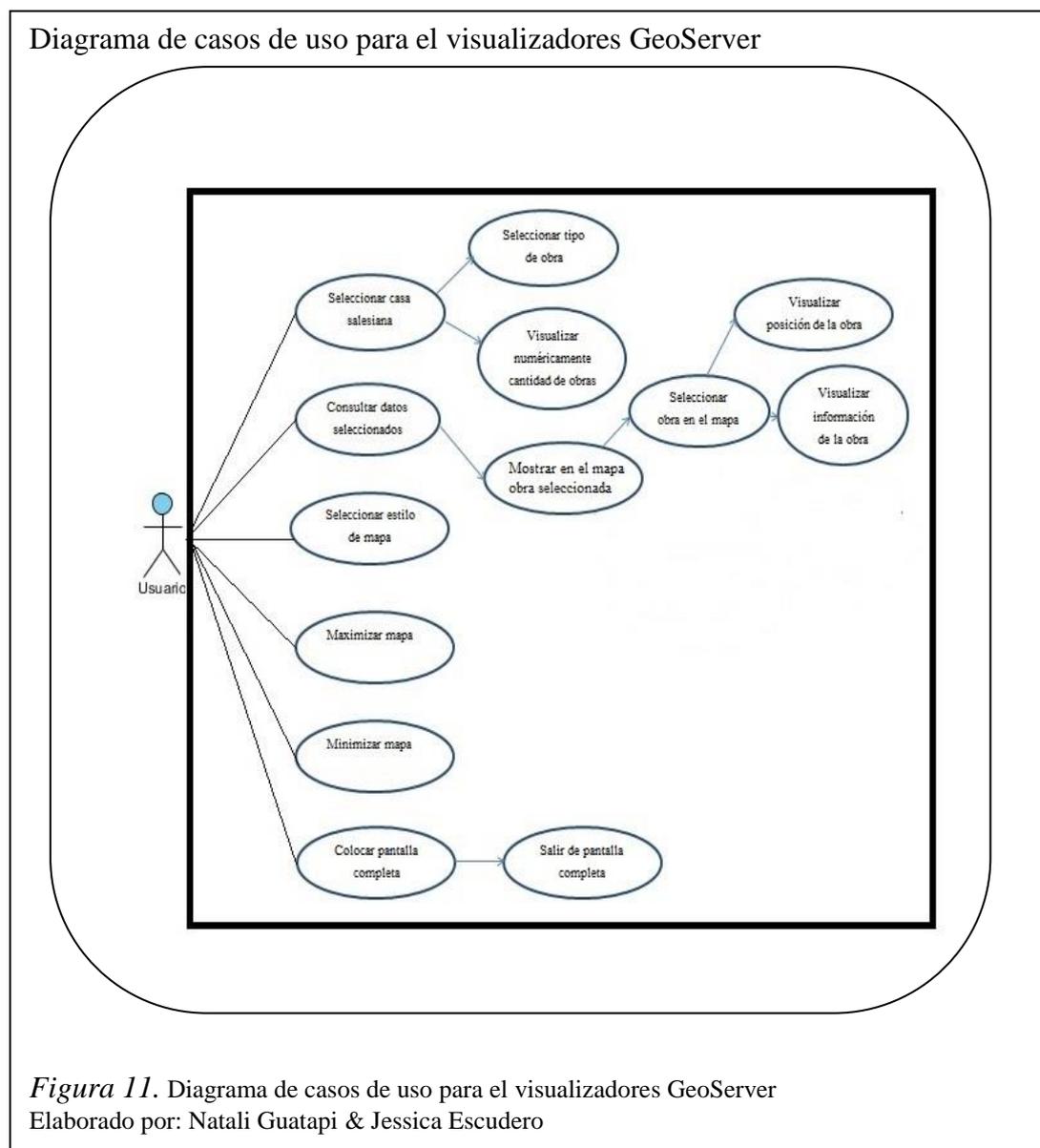
**Flujo normal:** el usuario mediante el combobox selecciona la casa salesiana y el tipo de obra, obteniendo mediante la consulta la obra correspondiente. A su vez, al seleccionar se puede obtener la información de la misma y además su respectiva información geográfica en los visualizadores de GeoTools y GeoPHP. Además

presionando sobre la obra por segunda vez se marca el área de influencia correspondiente a la obra de la cual se muestra la respectiva información.

**Flujo Alternativo:** el usuario puede maximizar y minimizar el mapa, colocar el producto en pantalla completa, seleccionar el tipo de mapa con que se desea trabajar y seleccionar el área que pertenece a la obra.

**Postcondiciones:** visualizar la obra seleccionada con su respectiva información.

Seguidamente se muestra el diagrama de casos de uso para el visualizador GeoServer, donde su actor es el usuario.



*Figura 11.* Diagrama de casos de uso para el visualizadores GeoServer  
Elaborado por: Natali Guatapi & Jessica Escudero

- Usuario

**Descripción:** visualización de las obras salesianas

**Actores:** usuario

**Precondiciones:** acceder al visualizador

**Flujo normal:** el usuario mediante el combobox selecciona la casa salesiana y el tipo de obra, obteniendo mediante la consulta la obra correspondiente. A su vez, al seleccionar una de ellas se puede obtener la información de la misma y además su respectiva información geográfica en los visualizadores de GeoTools y GeoPHP.

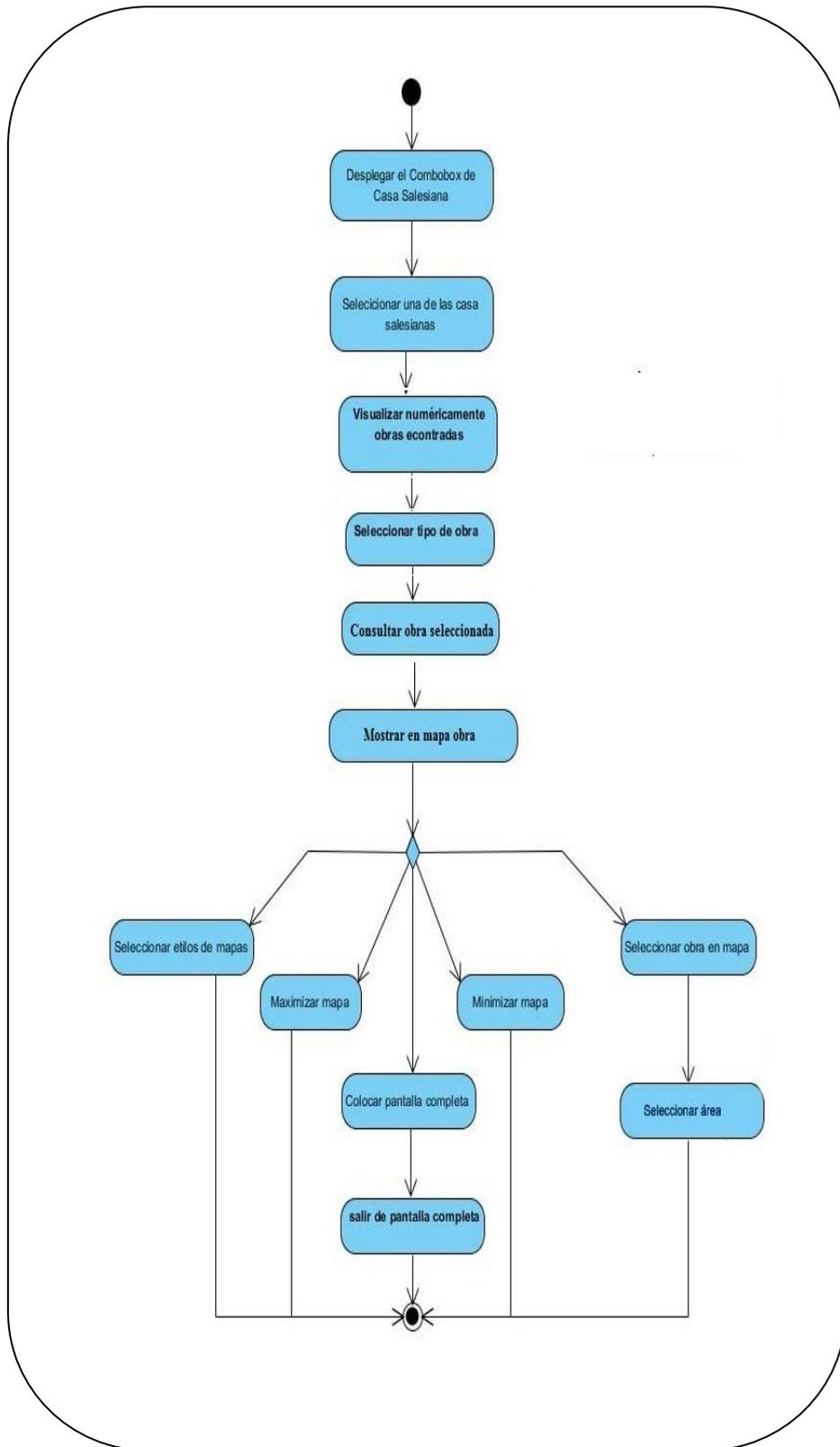
**Flujo Alternativo:** el usuario puede maximizar y minimizar el mapa, colocar el producto en pantalla completa y seleccionar el tipo de mapa con que se desea trabajar.

**Postcondiciones:** visualizar la obra seleccionada con su respectiva información.

El diagrama de actividad permite describir cómo un sistema implementa su funcionamiento mediante un conjunto de acciones que generan el proceso a describir integrado por acciones y flechas. Se puede decir además que los mismos modelan el comportamiento dinámico de un procedimiento, transacción o caso de uso, haciendo énfasis en el proceso que se lleva a cabo.

En esta investigación se desarrolló un diagrama de actividades que engloba todo el conjunto de acciones que desarrollan los prototipos GeoPHP, GeoServer y GeoTools. El mismo se muestra a continuación:

Diagrama de actividades para los prototipos GeoPHP y GeoTools.



*Figura 12.* Diagrama de actividades para los prototipos GeoPHP y GeoTools.  
Elaborado por: Natali Guatapi & Jessica Escudero

- Usuario

**Descripción:** visualización de las obras salesianas

**Actores:** usuario

**Precondiciones:** acceder al visualizador

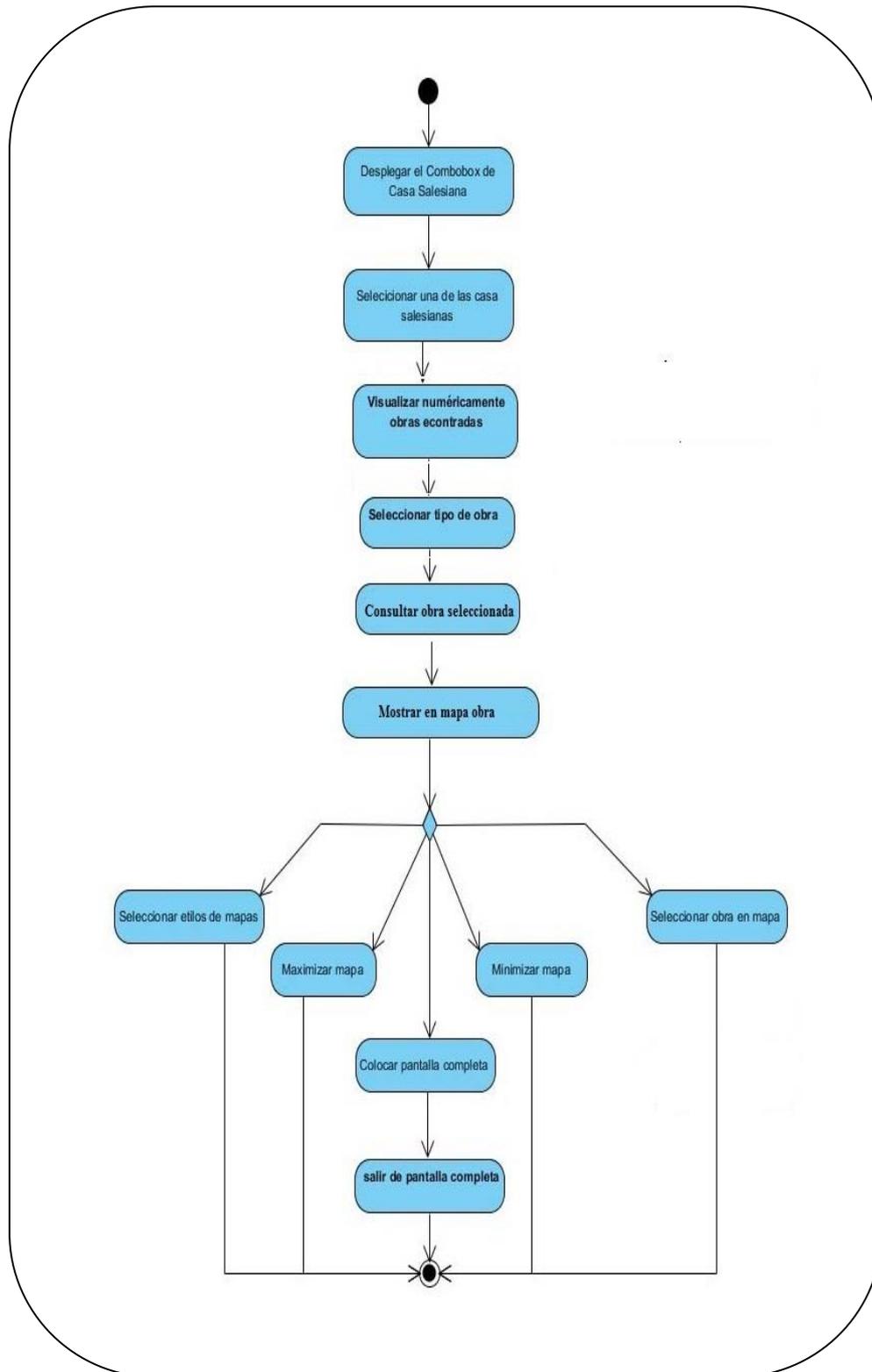
**Flujo normal:** el usuario mediante el combobox selecciona la casa salesiana y el tipo de obra, obteniendo mediante la consulta la obra correspondiente. El sistema muestra numéricamente la cantidad de obras encontradas de acuerdo a la casa salesiana.

**Flujo Alternativo:** el usuario puede maximizar y minimizar el mapa, colocar el producto en pantalla completa, seleccionar el tipo de mapa con que se desea trabajar, seleccionar la obra en el mapa que desea visualizar su información y además su respectiva información geográfica en los visualizadores de GeoTools y GeoPHP, así como mostrar su área.

**Postcondiciones:** visualizar la obra seleccionada con su respectiva información.

A continuación se muestra el diagrama de actividades para el visualizador GeoServer

## Diagrama de actividades para el prototipo GeoServer



*Figura 13.* Diagrama de actividades para el prototipo GeoServer  
Elaborado por: Natali Guatapi & Jessica Escudero

- Usuario

**Descripción:** visualización de las obras salesianas

**Actores:** usuario

**Precondiciones:** acceder al visualizador

**Flujo normal:** el usuario mediante el combobox selecciona la casa salesiana y el tipo de obra, obteniendo mediante la consulta la obra correspondiente. El sistema muestra numéricamente la cantidad de obras encontradas de acuerdo a la casa salesiana.

**Flujo Alternativo:** el usuario puede maximizar y minimizar el mapa, colocar el producto en pantalla completa, seleccionar el tipo de mapa con que se desea trabajar, seleccionar la obra en el mapa que desea visualizar su información y además su respectiva información geográfica en el visualizador GeoServer.

**Postcondiciones:** visualizar la obra seleccionada con su respectiva información.

### 3.2 Herramientas para el desarrollo de los prototipos

Para el tratamiento de los prototipos se utilizaron diversas herramientas que en su conjunto permitieron que los mismos se desarrollaran con éxito.

La plataforma NetBeans 8.0.2 permite que las aplicaciones sean desarrolladas a partir de un conjunto de componentes de software llamados módulos, quienes son archivos Java que contiene sus clases escritas para interactuar con las APIs de NetBeans y un archivo especial (manifest file) que lo identifica como módulo.

Como servidor web se utilizó Apache2 y Apache Tomcat 7.0.50, el cual posee tecnología Open Source. Este servidor es libre y se podría decir que es uno de los servidores más populares y utilizados a nivel mundial. Es flexible, robusto, potente, rápido, eficiente y continuamente actualizado.

Otras de las características que lo distinguen que es modular, puede ser adaptado a diferentes entornos y necesidades, con los diferentes módulos de apoyo que proporciona, para el desarrollo de módulos específicos, y además es extensible, pues gracias a ser modular se han desarrollado diversas extensiones entre las que destaca PHP 5.4.0, un lenguaje de programación del lado del servidor.

El servidor GeoServer 2.3.5 se utilizó como intermediario de conexión con la base de datos. Su procedimiento es de manera sencilla, nos permite crear capas en la cual se realiza la configuración y construcción de las vistas para realizar consultas a la base de datos.

Como sistema de gestión de base de datos se utilizó PostGIS 1.5.5 quien es una extensión de Postgree SQL9.1.1.

Se utilizó el sistema operativo Centos 6.5 que posee la característica de ser de código abierto, basado en la distribución Red Hat Enterprise Linux.

### **3.3 Construcción de prototipos**

#### **3.3.1 Configuración de GeoServer**

Previo a la configuración del servidor GeoServer se necesita restaurar la base de datos de las obras Salesianas de la Universidad.

Configurar el servidor GeoServer para agregar y realizar la conexión del origen de datos PostGIS y a su vez generar vistas las mismas que permite obtener los datos mediante una Url requeridos para la aplicación que se desarrolló en el lenguaje PHP y JavaScript.

Pasos para la configuración de GeoServer:

1. Instalar el GeoServer.
2. Acceder al sitio de administración del GeoServer.

<http://localhost:8080/GeoServer/web>

Autenticación:

- user: admin
- pass: geoserver

3. Crear nuevo origen de datos:

3.1. El primer paso es acceder al vínculo que se llama Almacenes de datos ubicado en el menú izquierdo de la página. Señalado en la siguiente imagen con el círculo rojo.

3.2. Acceder al vínculo PostGIS señalado con el círculo verde de la

*Figura 14.*

## Accediendo a PostGIS



Figura 14. Accediendo a PostGIS

Elaborado por: Natali Guatapi & Jessica Escudero

### 4. Rellenar los campos correspondientes al nuevo origen:

- 4.1. Espacio de trabajo: “topp”.
- 4.2. Nombre del origen de datos: “Postgisdb”.
- 4.3. Database: “base\_Tesis” (en este caso es el nombre de la base de datos que se restauró en postgresQL).
- 4.4. Nombre de usuario: “postgres” (el nombre de usuario que se usa para acceder a la base de datos que se creó en postgreeSQL).
- 4.5. Password:”postgres” (el password para acceder a la base de datos).
- 4.6. El resto de los campos se quedan los que están por defecto.
- 4.7. Pulsar el botón de Guardar.

## Rellenando campos de PostGIS

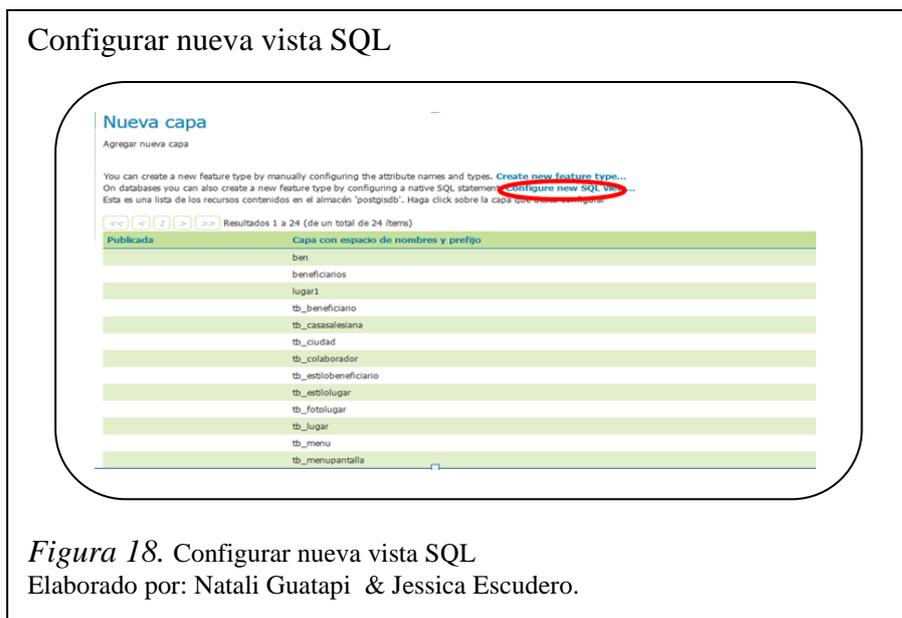


Figura 15. Accediendo a PostGIS

Elaborado por: Natali Guatapi & Jessica Escudero.



## 5.4 Configurar nueva vista SQL, señalada con el círculo en rojo en la *Figura 18*.



*Figura 18.* Configurar nueva vista SQL  
Elaborado por: Natali Guatapi & Jessica Escudero.

## 5.5 Configurar vistas:

### 5.5.1 Nombre: *ObrasViewFullProperties*

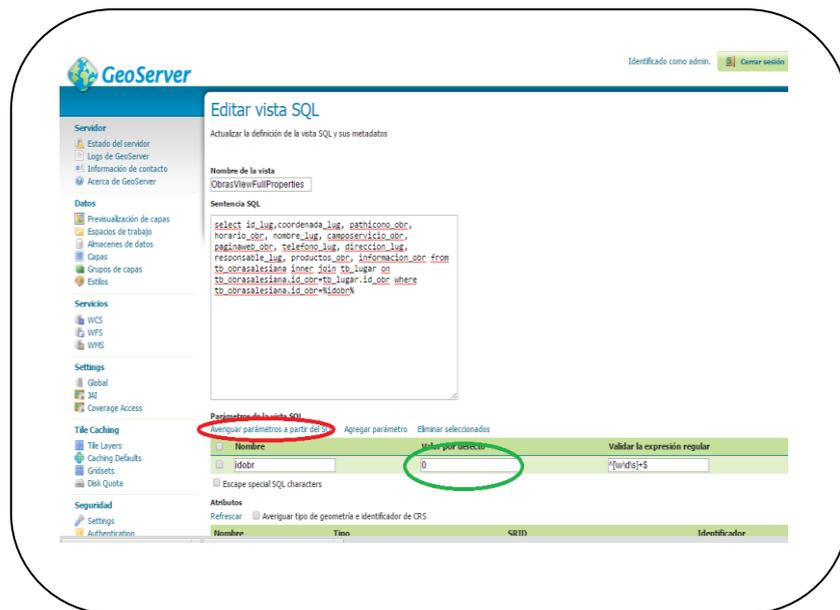
5.5.2 Consulta SQL: 

```
select id_lug, coordenada_lug, pathicono_obr, horario_obr, nombre_lug, camposervicio_obr, paginaweb_obr, telefono_lug, direccion_lug, responsable_lug, productos_obr, informacion_obr from tb_obrasalesiana inner join tb_lugar on tb_obrasalesiana.id_obr=tb_lugar.id_obr where tb_obrasalesiana.id_obr=%idobr%
```

5.5.3 Pulsar el vínculo de “averiguar parámetros a partir del SQL” señalado con el círculo en rojo en la *Figura 19*.

5.5.4 Insertar los valores por defecto de la para la variable idobr. Como se señala en el círculo en verde *Figura 19*.

## Variables por defecto



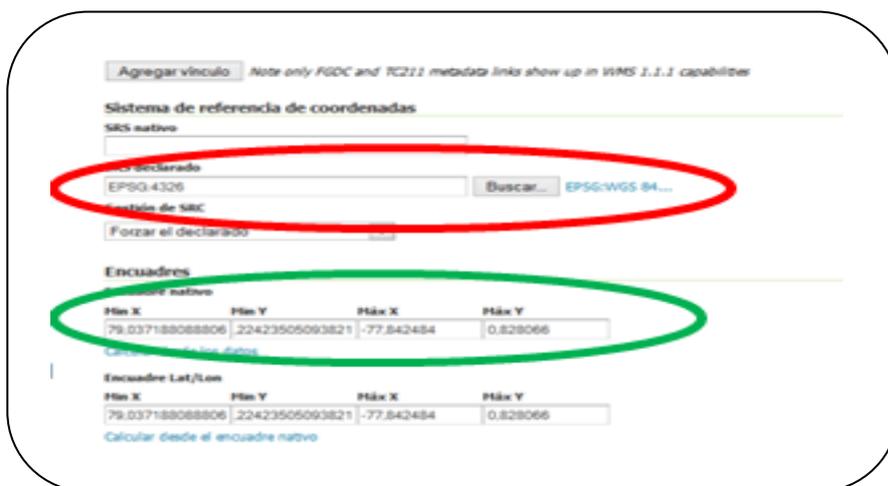
*Figura 19.* Insertar variables por defecto  
Elaborado por: Natali Guatapi & Jessica Escudero.

5.5.5 Pulsar el botón guardar

5.5.6 Insertar el SRS declarado con el valor EPSG: 4326. Señalado con el círculo en rojo en la *Figura. 20*.

5.5.7 Pulsar el vínculo Calcular desde los datos. Señalado con el círculo en verde en la *Figura. 20*.

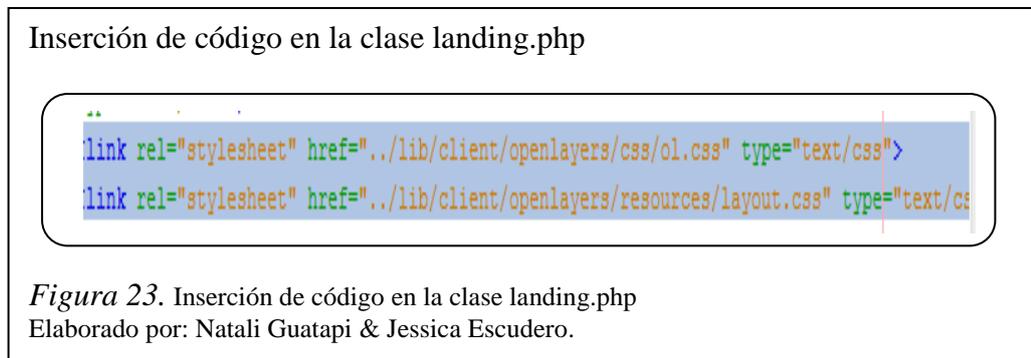
## Insertar el SRS



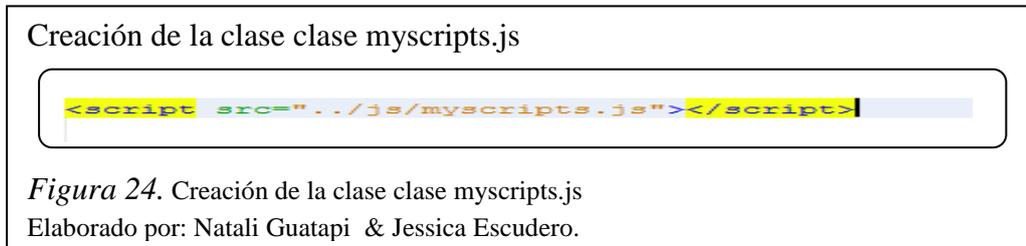
*Figura 20.* Insertar el SRS  
Elaborado por: Natali Guatapi & Jessica Escudero.



4- Para la visualizar el mapa hay que incluir el siguiente código en la clase landing.php.



5- Para conectarse con el servidor GeoServer se creó una clase myscripts.js el cual es importado de la siguiente manera en landing.php.



En este visualizador no se utiliza librerías por lo cual se está trabajando por medio de capas en las cuales se crearon vistas.

Componentes para la realización del visualizador GeoServer.

Tabla 6. Usabilidad de los componentes para el visor web

Componentes	Descripción	Observación
<b>Servidor Geoserver</b>	Es un servidor de mapas	Se utilizó para la conexión de la base de datos postgis, y la creación de las capas con sus respectivas vistas.
<b>OpenLayers</b>	Es una librería de JavaScript que permite previsualizar los mapas de diferentes estilos y con	Se utilizó para el diseño de la aplicación pa visualizar los estilos OpenStreerMap,

	sus respectivas herramientas.	MapQuest, WaterColor, Sat, maximizar , minimar, acercar, pantalla completa mapa.
<b>Php</b>	Lenguaje de programación de código abierto.	Se utilizó para el desarrollo del visualizador.

Nota. Lista de componentes utilizados en el visualizador GeoServer.  
Elaborado por: Natali Guatapi & Jessica Escudero

### 3.3.3 Visualizador GeoPHP

Para el desarrollo del visualizador se crea un proyecto web en PHP.

Descargar la librería de GeoPHP e insertar a proyecto el mismo que es utilizado para previsualizar los datos geográficos del visualizador que está conectado a la base de datos postgres.

Se crea una clase PGConnection.php que permite la conexión con la base de datos postgresSQL-postgis esta contiene el siguiente código.

```
public function create_connection($host, $database, $port, $user,
$password){ $this->dbconnection = pg_connect("host=$host
dbname=$database port=$port user=$user password=$password"); }
```

Una vez que tenga conexión con la base de datos se hace referencia a la librería de GeoPHP en la clase GeoPHP.inc esta me permite gestionar las librerías de la siguiente manera.

```
include_once("lib/geometry/Geometry.class.php"); // Abstract class
include_once("lib/geometry/Point.class.php");
include_once("lib/geometry/Collection.class.php"); // Abstract class
include_once("lib/geometry/LineString.class.php");
include_once("lib/geometry/MultiPoint.class.php");
include_once("lib/geometry/Polygon.class.php");
include_once("lib/geometry/MultiLineString.class.php");
include_once("lib/geometry/MultiPolygon.class.php");
include_once("lib/geometry/GeometryCollection.class.php");
```

get\_features\_collection: este método permite obtener una estructura FeaturesCollection: la que se utiliza como punto de unión con la librería OpenLayers a través de la cual se crea una capa para visualizar las geometrías en nuestro caso. El mismo recibe un arreglo con la información espacial obtenida mediante una consulta a la BD.

`get_geometry_properties`: mediante este método se puede obtener un arreglo con las propiedades de una geometría utilizando la librería GeoPHP

`get_geometry_json`: este método permite obtener la estructura JSON de una geometría, este método es muy importante también para la comunicación con la librería OpenLayers3.

### 3.3.3.1 Capa del modelo

En este proyecto se utiliza como gestor de base de datos PostgreSQL y para añadir soporte de objetos gráficos a la misma se utilizó el módulo PostGIS. Esto permitió almacenar la información geográfica en una columna de tipo geometry.

La base de datos relacional cuenta con una tabla `tb_casasalesiana` en la que se almacena toda la información de las Casas Salesianas de importancia para la investigación. Entre los campos más importantes almacenados en esta tabla se encuentran la información particular de esta distribuida en los campos `nombre_cas`, `direccion_cas`, `telefono_cas`, `correo_cas` y `director_cas`. También en la base de datos aparece la tabla `tb_obrasalesiana` la cual almacena la información de las obras de importancia, los principales campos son `denominacion_obr`, `camposervicio_obr`, `productos_obr`, `horario_obr` y `pagina web_obr`. Además se encuentran los campos `id_tobr` e `id_cas` los cuales permiten saber de qué tipo es la obra y a que casa salesiana pertenece. Estos dos últimos campos son los más importantes para realizar las consultas. Por último aparece también la tabla `tb_tipoobra` que almacena los tipos de obras existentes.

Tabla 7. *Usabilidad de las librerías de GeoPHP en el visor web*

Métodos Librería	Propósito	Usabilidad en el proyecto
<b>Área</b>	Visualizar el área donde está colocada la obra.	Se utiliza en la obtención de los datos del área.
<b>Centroid</b>	El centroide devuelve el valor numérico de las coordenadas del punto donde está la obra	Se utiliza para obtener el centro de las obras.

	seleccionada	
<b>Length</b>	Devuelve la longitud de esta curva.	Devuelve la dimensión de la curva del área señalada.
<b>Y</b>	Devuelve el valor del centroide de Y.	Se utiliza para obtener el valor de Y en el centro de las obras
<b>X</b>	Devuelve el valor del centroide de X.	Se utiliza para obtener el valor de X en el centro de las obras
<b>GeometryType</b>	Crea una instancia dentro de una clase en el desarrollo del prototipo.	Se utiliza para llamar a la librería en una instancia.

Nota. Librerías utilizadas para la elaboración del visualizador GeoPHP  
Elaborado por: Natali Guatapi & Jessica Escudero

### 3.3.4 Construcción de visualizador prototipo GeoTools

Para realizar el desarrollo del visualizador prototipo de GeoTools se utilizó el frameworks Spring MVC (Modelo-Vista-Controlador) en la parte del servidor para el desarrollo del visualizador web que utiliza GeoTools en su capa de servicios; y el framework OpenLayers en la parte del cliente para trabajo con capas, mapas y la visualidad de la aplicación. Se utilizaron además otras librerías clientes JavaScript como JQuery y el framework Bootstrap para obtener una visualidad adecuada.

La descripción de clases y métodos desarrollados en la parte del servidor es la siguiente (el lenguaje de programación utilizado fue Java):

#### **Capa de modelos:**

En este proyecto se utilizó como gestor de base de datos PostgreSQL y para añadir soporte de objetos gráficos a la misma se utilizó el módulo PostGIS. Esto permitió almacenar la información geográfica en una columna de tipo geometry.

Clase GeometryProperties: modelo que se encarga de almacenar todos los atributos extraídos de la base de datos relacionados con la geometría y otros atributos auxiliares que guardan las operaciones efectuadas a la geometría.

Clase MyGeometry: modelo auxiliar y simple de geometría que se encarga de sustituir a una geometría del paquete de GeoTools. Esta clase es solamente utilizada en la respuesta json del servidor relacionada a una petición ajax de consulta de obras).

Clase GetGeometry: servicio que se encarga de devolver una geometría dado un String en formato wkt.

### **Capa de servicios:**

Clase GetDataFromPostGIS: permite la conexión con la base de datos PostGIS en la cual hay que importar las siguientes librerías para la conexión de la misma.

```
import org.geotools.data.DataStore;
import org.geotools.data.DataStoreFinder;
params=new HashMap<String, Object>();
    params.put("dbtype", "postgis");
    params.put( "host", "localhost");
    params.put( "port", 5432);
    params.put( "schema", "public");
    params.put( "database", "base_Tesis");
    params.put( "user", "postgres");
    params.put( "passwd", "postgres");
```

Métodos:

- List<GeometryProperties> GetLayers (String casa\_salesiana, String tipo\_obra): accede a la base de datos y devuelve la lista de geometrías que corresponden con los parámetros especificados.
- Void GetInitialData() :accede a la base de datos; y con esa respuesta construye los HashMaps referentes a las casas salesianas y los tipos de obras.
- Geometry getGeometryByWkt(String the\_geom): devuelve una geometría dado un String en formato wkt.

### **Capa de controladores:**

Clase HomeController: único controlador de peticiones de la aplicación web.

Métodos:

- ShowHome: muestra la vista por defecto del visor que se encuentra en el fichero home.jsp

- **GetGeometries:** método que devuelve una respuesta json con las obras asociadas a los parámetros casa salesiana y tipo de obra, además de sus geometrías y el resultado de otras operaciones efectuadas a ellas. Este método es accedido por una petición Ajax.

Tabla 8. *Usabilidad de las librerías de GeoTools en el visor web*

Módulo	Propósito	Usabilidad en el proyecto
<b>Gt-render</b>	Implementa un engine de rendering Java2D para dibujar un mapa	Ninguna, pues la aplicación es web y este render es para ser usado en aplicaciones de escritorio
<b>Gt-jdbc</b>	Permite el acceso a bases de datos espaciales	Se utiliza en la conexión con la base de datos espacial que contiene la información geográfica de la aplicación (específicamente el artefacto gt-jdbc-PostGIS)
<b>Gt-data</b>	Permite el acceso a datos espaciales	Se utilizan las implementaciones de Features, DataStore, entre otras, en la conexión con la base de datos espacial y el manejo de geometrías
<b>Gt-xml</b>	Implementación de formatos espaciales xml	Ninguna, pues no se trabaja con ninguna estructura xml
<b>Gt-cql</b>	Implementación del Lenguaje de Consulta Común para la creación de filtros (CQL)	Se utilizan algunos de estos filtros en las consultas de acceso a la

		base de datos
<b>Gt-main</b>	Implementación de filtros, features, etc ...	Ninguna, pues se usan las implementaciones de filtros, features,.. de los módulos gt-data y gt-jdbc
<b>Gt-api</b>	Definición de interfaces para el trabajo con información espacial	Se utiliza, pues define interfaces que son implementadas en los otros módulos de GeoTools que si forman parte de la aplicación
<b>Jts</b>	Definición e implementación de geometrías	Se utiliza en la obtención y manipulación de geometrías
<b>Gt-coverage</b>	Permite el acceso a información raster	Ninguna, pues no hay información raster en la aplicación
<b>Gt-referencing</b>	Permite la localización y transformación de coordenadas	Ninguna, pues no hay transformación de coordenadas en la aplicación
<b>Gt-metadata</b>	Permite la identificación y descripción de datos	Ninguna, pues no es requisito de la aplicación la generación de metadatos
<b>Gt-opengis</b>	Definición de interfaces para conceptos espaciales comunes	Se utilizan implementaciones de features en la conexión con la base de datos

Nota. Lista de librerías de Geotools.  
Elaborado por: Natali Guatapi & Jessica Escudero

## **CAPÍTULO 4**

### **IMPLEMENTACIÓN Y PRUEBAS**

#### **4.1 Implementación en el servidor CIMA**

La Universidad Politécnica Salesiana cuenta con el Centro de Investigación en Modelamiento Ambiental (CIMA), el cual es el encargado de dar respuestas científicas a las necesidades de gestión ambiental, sobre los efectos contemporáneos del ambiente y a los recientes cambios en el clima regional con información confiable a corto, mediano y largo plazo, vinculando a la Universidad Politécnica Salesiana con los sectores gubernamentales de toma de decisión y planificación, convirtiéndose en referente nacional en cuanto al estudio del ambiente, generando información accesible, confiable y oportuna que permita la generación de planes a nivel local, nacional e internacional para su cuidado y preservación.

Este centro también tiene la función de archivar todas las investigaciones que se realizan en la universidad, es por ello que la misma será guardada en él y se implementará en el servidor los tres prototipos desarrollados para que los estudiantes y usuarios interesados puedan utilizarlo en función de sus necesidades.

#### **4.1.1 Programas**

Después de haber realizado el diseño de los prototipos, así como la configuración necesaria, adecuada y óptima para su funcionamiento, los mismos están listos para ser implementados en la organización CIMA. Para ello en el servidor deben estar instalados programas necesarios para que las aplicaciones puedan ser utilizadas sin ningún problema. Los mismos son nombrados a continuación:

##### **4.1.1.1 Requerimientos para la implementación de los visualizadores prototipos VisGeo(GeoServer, GeoTools,GeoPHP)**

- Centos 6.5
- Apache2
- Apache Tomcat 7.0.50
- Php 5.4
- Postgres 9.1.1

- Postgis 1.5.
- Geoserver 2.3.5
- Java 1.7.0.75

#### 4.1.2 Restaurar base de datos

Crear base de datos base\_Tesis en postgres con extensión a Postgis.

```
Psql -U postgres -W
Créate Database base_Tesis;
```

Activar las funciones postgis

```
Psql -U postgres -d base_Tesis -f postgis.sql
```

Activar las funciones Spatial\_ref\_sys.sql

```
Psql -U postgres -d base_Tesis -f Spatial_ref_sys.sql
```

Restaurar la base de datos mediante un respaldo tipo sql

```
Psql -U postgres -d base_Tesis -f basetesis.sql
```

Actualización de java de la versión 1.6 a 1.7.0.75

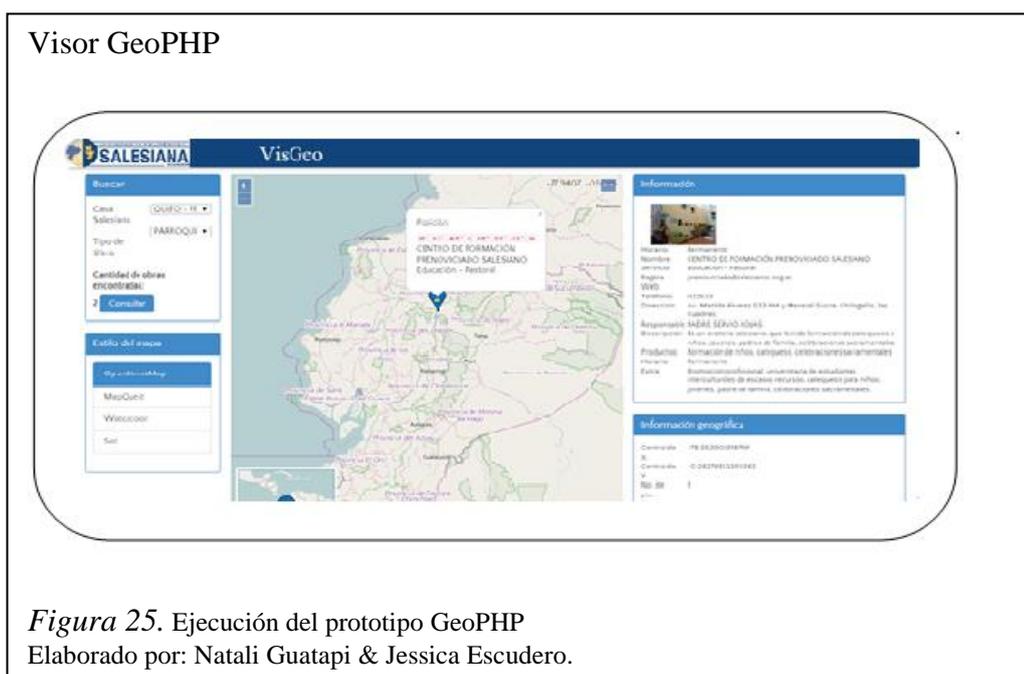
```
Yum install java
```

#### 4.1.3 Levantamiento de los visores prototipo VisGeo en el servidor.

##### 4.1.3.1 Visor GeoPHP

Ubicar el proyecto VisorGeoPHP en el servidor Web Apache en la siguiente dirección var/www/html/ y dar permisos con el siguiente sentencia.

```
Chmod +x var/www/html/ VisorGeoPHP
Service httpd restart
```



*Figura 25.* Ejecución del prototipo GeoPHP  
Elaborado por: Natali Guatapi & Jessica Escudero.

### 4.1.3.2 Visor GeoServer

Ubicar el proyecto VisorGeoServer en el servidor Web Apache en la siguiente dirección var/www/html y dar permisos con la siguiente sentencia y reiniciar el servidor.

```
Chmod +x var/www/html/ VisorGeoPHP  
Service httpd restart
```

Configurar el Geoserver que se encuentra en el servidor de tomcat7 en la dirección opt/tomcat7/webapps/ y reiniciar el servidor.



*Figura 26.* Ejecución del prototipo GeoServer  
Elaborado por: Natali Guatapi & Jessica Escudero.

### 4.1.3.3 Visor de GeoTools

Ubicar el proyecto GeoVisualizer en el servidor Web Tomcat en la siguiente dirección var/www/html y dar permisos con la siguiente sentencia y reiniciar el servidor.

```
Chmod +x opt/tomcat7/webapps/GeoVisualizer.war  
Opt/tomcat7/bin/startup.sh
```



## 4.2 Pruebas

Las pruebas que se realizarán para determinar el buen funcionamiento del sistema son las pruebas de rendimiento, quienes evalúan el desempeño del módulo de asignación de espacios físicos y las funciones que desempeña. Estas pruebas, realizadas sobre computadoras, redes, software u otros dispositivos, son utilizados para determinar la velocidad y eficiencia de los mismos.

Este procedimiento puede involucrar tanto tests cuantitativos, por ejemplo, medir tiempos de respuesta o cantidad en millones de líneas de código, como tests cualitativos, en los cuales se evalúa fiabilidad, escalabilidad e interoperabilidad.

Las limitaciones en los tiempos de respuesta de los prototipos desarrollados (GeoTools, GeoServer y GeoPHP) a la hora de realizar estas pruebas, según el criterio de la investigación están dados por:

- El servidor testeado se encuentra en la misma red en la cual se realizaron las pruebas.
- Velocidad de conexión del servidor.
- Velocidad de conexión del cliente.
- Tiempo en el cual el navegador web tarda para dibujar la página (tiempo muy pequeño).
- Rendimiento de la red en el momento de la prueba.

Las pruebas de rendimiento pueden ser realizadas a través de herramientas que proveen pruebas de estrés, que permiten determinar la estabilidad del sistema.

Para la realización de las pruebas se utilizó el programa Jmeter, herramienta open source muy completa, implementada en Java que permite realizar test de comportamiento funcional. También se puede utilizar para realizar pruebas de estrés, por ejemplo, en un servidor y poner a prueba su rendimiento.

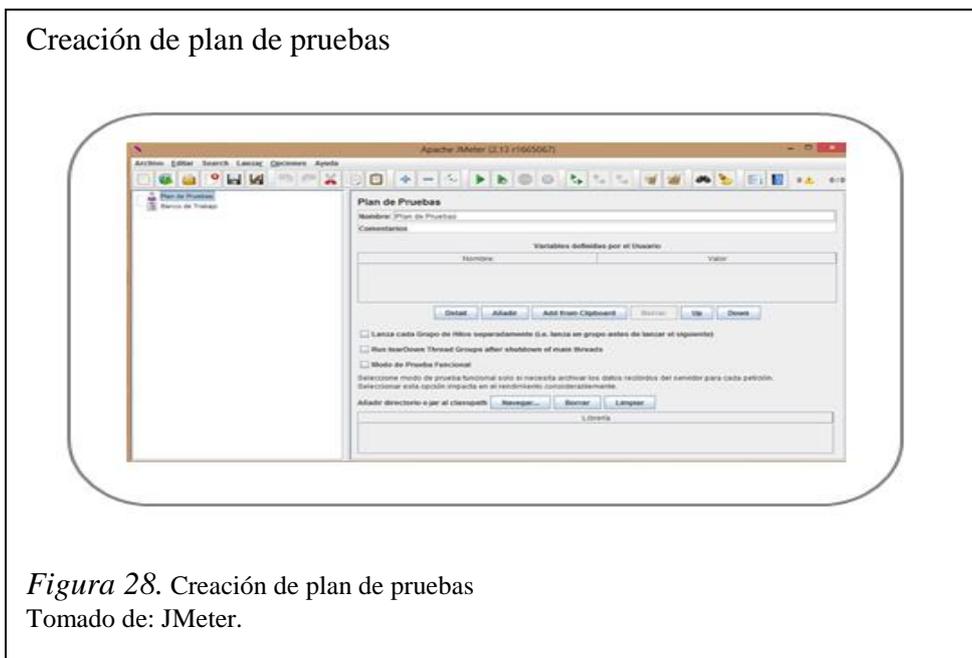
#### 4.2.1 Plan de pruebas

Las pruebas se realizaron sobre una computadora con las propiedades siguientes:

- Servidor: 3 GB de RAM, 32 bits
- Procesador: Inter Core 2Duo E6550 a 2.33 GHz

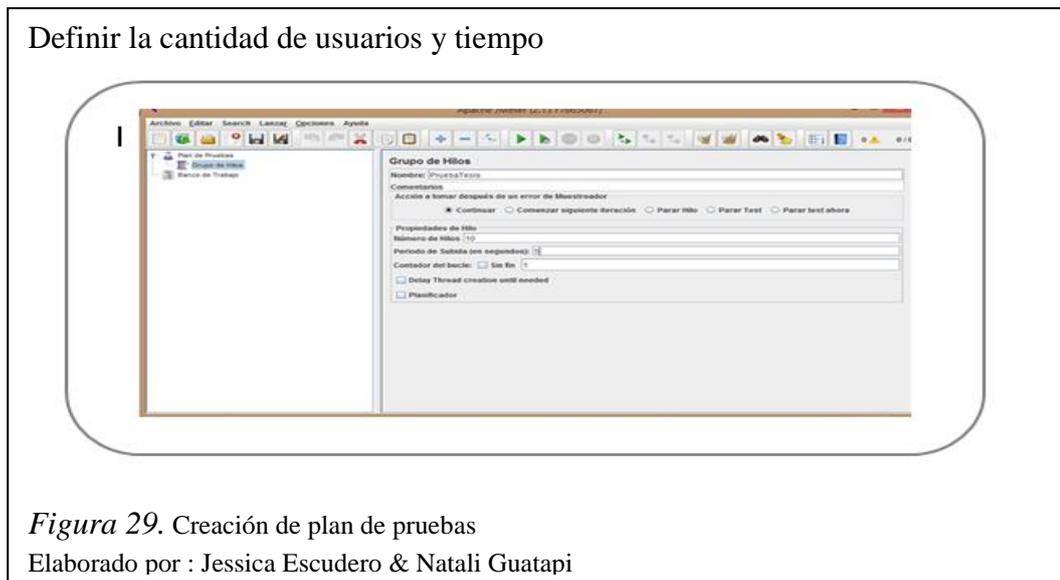
Se define el plan de pruebas con la herramienta jmeter, obteniéndose la imagen mostrada en la *Figura 28*.

En la parte izquierda se define el plan de pruebas que se va a realizar, mientras que en la parte derecha nos permite editar, a partir del ingreso de valores, lo que tenemos en la sección izquierda.



Sobre el Test plan o plan de pruebas se definen la cantidad de usuarios que se supone que navegan por los prototipos creados al unísono y el tiempo en segundo que se

demoran para hacer esta función, de la manera siguiente: Botón derecho sobre Test plan-Add-Thread Group, lo cual quedaría como se muestra en la *Figura 29*.

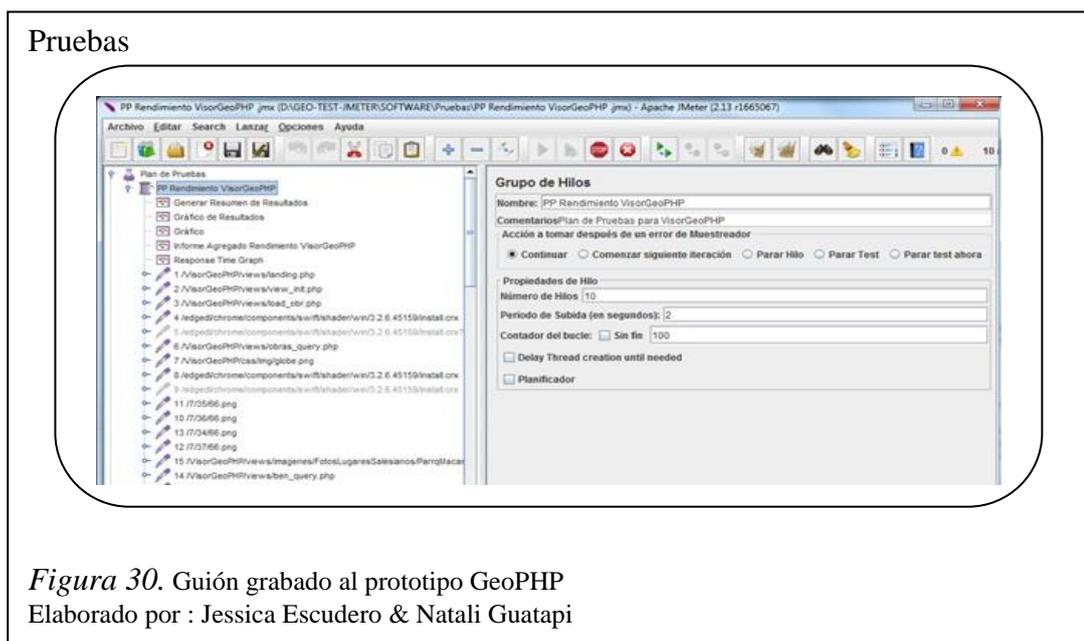


#### 4.2.1.1 Casos de pruebas

Después de configurar el Jmeter se desarrollarán tres casos de pruebas para cada uno de los prototipos desarrollados (GeoPHP, GeoServer y GeoTool):

- Guión grabado
- Gráfico de resultados
- Informe agregado

##### 4.2.1.1.1 Pruebas desarrolladas al prototipo GeoPHP







## Pruebas

**Informe Agregado**

Nombre: Informe Agregado

Comentarios

Escribir todos los datos a Archivo

Nombre de archivo:     Escribir en Log Sólo Errores  Exito

Etiqueta	# Muestras	Media	Mediana	90% Line	95% Line	99% Line	Mín	Más	% Error	Rendimiento	Kbsec
1 NisorGeoServ.	1000	541	541	597	600	608	6	625	0.00%	2.78sec	20.4
2 NisorGeoServ.	1000	535	539	595	596	605	3	620	0.00%	2.78sec	19.7
3 NisorGeoServ.	1000	495	555	613	629	664	14	689	0.00%	2.78sec	3.9
4 NisorGeoServ.	1000	36	30	60	76	110	16	189	0.00%	2.78sec	1.5
5 NisorGeoServ.	1000	30	24	54	67	97	12	133	0.00%	2.78sec	3.4
6 NisorGeoServ.	1000	28	23	52	67	100	11	154	0.00%	2.78sec	1.7
8 NisorGeoServ.	1000	9	7	17	25	36	3	512	0.00%	2.78sec	371.9
7 NisorGeoServ.	1000	55	28	71	111	590	16	827	0.00%	2.78sec	52.0
9 NisorGeoServ.	1000	460	549	613	630	674	16	751	0.00%	2.78sec	4.2
10 NisorGeoServ.	1000	30	23	55	71	111	12	158	0.00%	2.78sec	3.2
12 NisorGeoServ.	1000	29	23	50	63	101	12	221	0.00%	2.78sec	2.3
11 NisorGeoServ.	1000	27	22	49	65	98	11	160	0.00%	2.78sec	1.3
13 NisorGeoServ.	1000	7	5	12	17	31	3	333	0.00%	2.78sec	247.2
14 NisorGeoServ.	1000	421	542	611	626	659	12	706	0.00%	2.78sec	4.0
16 NisorGeoServ.	1000	28	23	49	63	97	12	188	0.00%	2.78sec	2.4
15 NisorGeoServ.	1000	27	22	47	62	103	11	153	0.00%	2.78sec	1.4
17 NisorGeoServ.	1000	8	6	14	20	40	3	394	0.00%	2.78sec	314.9
16 NisorGeoServ.	1000	33	28	53	66	104	16	213	0.00%	2.78sec	1.0
19 NisorGeoServ.	1000	29	23	49	67	116	12	178	0.00%	2.78sec	3.5
20 NisorGeoServ.	1000	28	22	50	65	98	11	129	0.00%	2.78sec	1.7
21 NisorGeoServ.	1000	29	23	50	62	104	12	155	0.00%	2.78sec	2.4
22 NisorGeoServ.	1000	9	5	16	24	42	3	514	0.00%	2.78sec	291.0
23 NisorGeoServ.	1000	33	29	58	72	100	16	140	0.00%	2.78sec	1.6
24 NisorGeoServ.	1000	29	23	54	68	96	12	148	0.00%	2.78sec	3.1
26 NisorGeoServ.	1000	28	23	50	63	107	11	180	0.00%	2.78sec	2.6
25 NisorGeoServ.	1000	560	559	615	626	661	16	711	0.00%	2.78sec	20.9
Total	26000	136	26	59	59	623	3	791	0.00%	69.29sec	1375.2

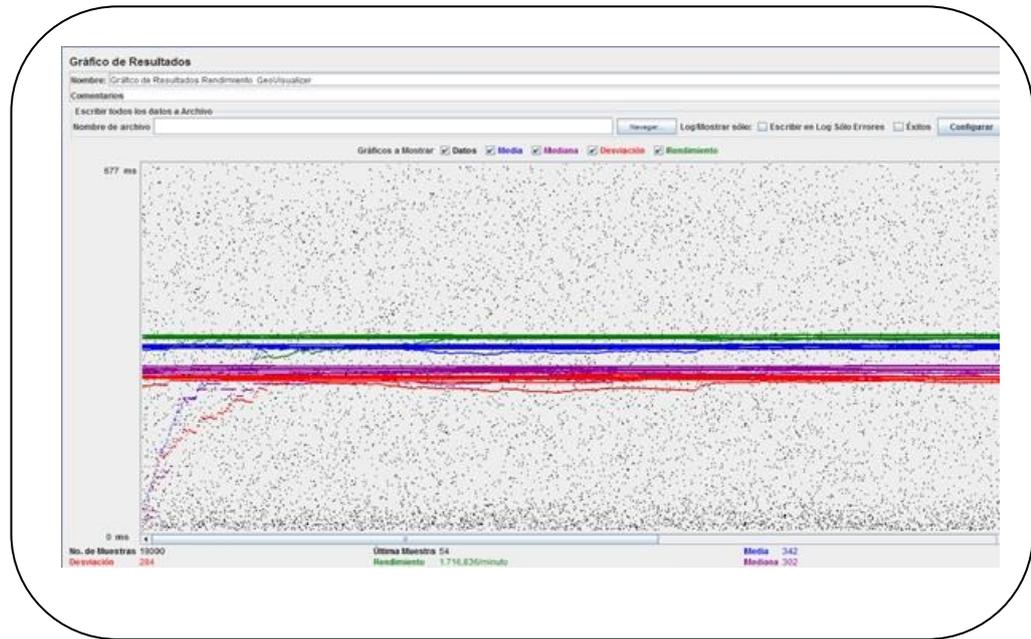
**Figura 35.** Informe agregado del prototipo GeoServer  
Elaborado por : Jessica Escudero & Natali Guatapi

### 4.2.1.1.3 Pruebas desarrolladas al prototipo GeoTools

## Pruebas

**Figura 36.** Guión grabado prototipo GeoTools  
Elaborado por : Jessica Escudero & Natali Guatapi.

## Pruebas



*Figura 37.* Gráfico de resultados del prototipo GeoTools  
Elaborado por : Jessica Escudero & Natali Guatapi

## Pruebas

Informe Agregado

Nombre: Informe Agregado Rendimiento GeoVisualizer

Comentarios

Escribir todos los datos a Archivo

Nombre de archivo

Log/Mostrar sólo:  Escribir en Log Sólo Errores  Éxitos  Configurar

Etiqueta	# Muestras	Media	Mediana	90% Line	95% Line	99% Line	Min	Máx	% Error	Rendimiento	Kb/sec
27 /GeoVisu...	1000	30	23	58	75	116	6	349	0.00%	1.5/sec	45.9
28 /GeoVisu...	1000	417	388	712	793	979	65	1540	0.00%	1.5/sec	.9
29 /GeoVisu...	1000	447	420	744	835	1134	65	4234	0.00%	1.5/sec	.6
31 /GeoVisu...	1000	38	28	72	96	197	5	670	0.00%	1.5/sec	186.3
30 /GeoVisu...	1000	425	395	715	788	1019	63	2855	0.00%	1.5/sec	5.3
32 /GeoVisu...	1000	444	404	749	839	1068	73	2908	0.00%	1.5/sec	.6
34 /GeoVisu...	1000	76	29	120	145	193	4	630	0.00%	1.5/sec	153.0
33 /GeoVisu...	1000	417	387	715	795	998	68	2166	0.00%	1.5/sec	5.2
35 /GeoVisu...	1000	420	395	710	800	971	61	3711	0.00%	1.5/sec	1.1
36 /GeoVisu...	1000	433	395	732	840	1209	57	3646	0.00%	1.5/sec	.6
38 /GeoVisu...	1000	40	28	79	100	143	5	561	0.00%	1.5/sec	204.0
37 /GeoVisu...	1000	526	501	901	1022	1342	51	1684	0.00%	1.5/sec	.8
39 /GeoVisu...	1000	426	384	721	819	1114	48	2442	0.00%	1.5/sec	.6
40 /GeoVisu...	1000	431	392	732	823	1084	48	3041	0.00%	1.5/sec	.7
41 /GeoVisu...	1000	477	555	625	647	688	4	842	0.00%	1.5/sec	151.8
42 /GeoVisu...	1000	410	375	719	834	1051	49	1976	0.00%	1.5/sec	.9
43 /GeoVisu...	1000	432	391	741	836	1152	50	3045	0.00%	1.5/sec	.6
45 /GeoVisu...	1000	75	54	152	192	375	10	521	0.00%	1.5/sec	426.7
44 /GeoVisu...	1000	532	498	941	1065	1386	52	2924	0.00%	1.5/sec	1.0
Total	19000	342	302	701	808	1081	4	4234	0.00%	28.6/sec	1179.9

*Figura 38.* Informe agregado del prototipo GeoTools  
Elaborado por : Jessica Escudero & Natali Guatapi

Con las pruebas de rendimiento desarrolladas utilizando la herramienta Jmeter se demostró que no se detectaron errores en ninguno de los prototipos desarrollados a partir del guión creado. Se estimaron las pruebas con 10 usuarios conectados al mismo tiempo, con un guión de cinco búsquedas, realizando 100 iteraciones por cada uno de ellos, lo cual equivale a cinco mil búsquedas de obras salesianas.

La cantidad de request o pedidos del Jmeter a la aplicación depende de la arquitectura y la implementación de la misma. De los tres prototipos creados, tuvo mayor rendimiento GeoTools, quien fue implementado con el lenguaje de programación Java, obteniéndose como promedio de respuesta del servidor 342 ms por pedido.

### **4.3 Manual de usuario**

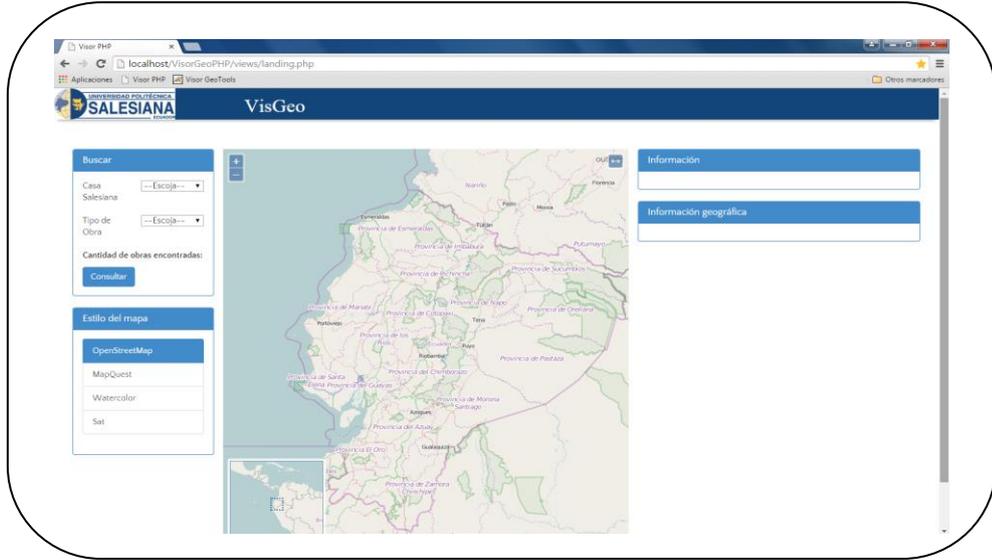
Los prototipos desarrollados en esta investigación tienen por finalidad que los usuarios puedan realizar la búsqueda de datos geográficos de la Inspectoría Salesiana.

Con este manual se podrán utilizar las funcionalidades básicas de los mismos sin afrontar ningún problema. A continuación se expondrán los pasos para su utilización:

#### **4.3.1 Manual de usuario para GeoPHP y Geo Tools**

1. Abrir página de inicio. Acceder a la siguiente dirección <http://ide.ups.edu.ec/VisorGeoPHP/> para el visor prototipo de GeoPHP o/y <http://ide.ups.edu.ec:8081/GeoVisualizer> para el visor prototipo de Geotools.

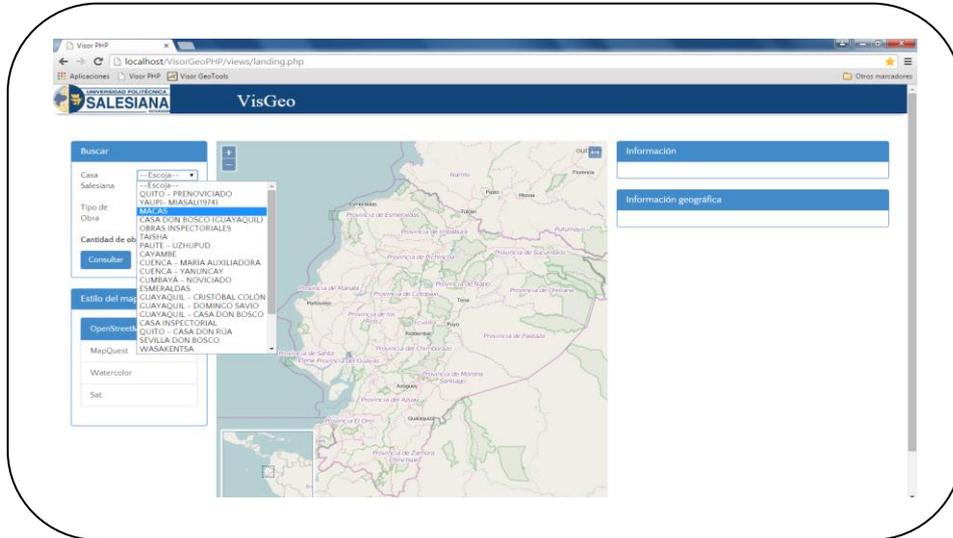
## Manual de usuario



*Figura 39.* Página de inicio GeoTools y GeoPHP  
Elaborado por : Jessica Escudero & Natali Guatani

2. Seleccionar casa salesiana desplegando el combobox casa salesiana.

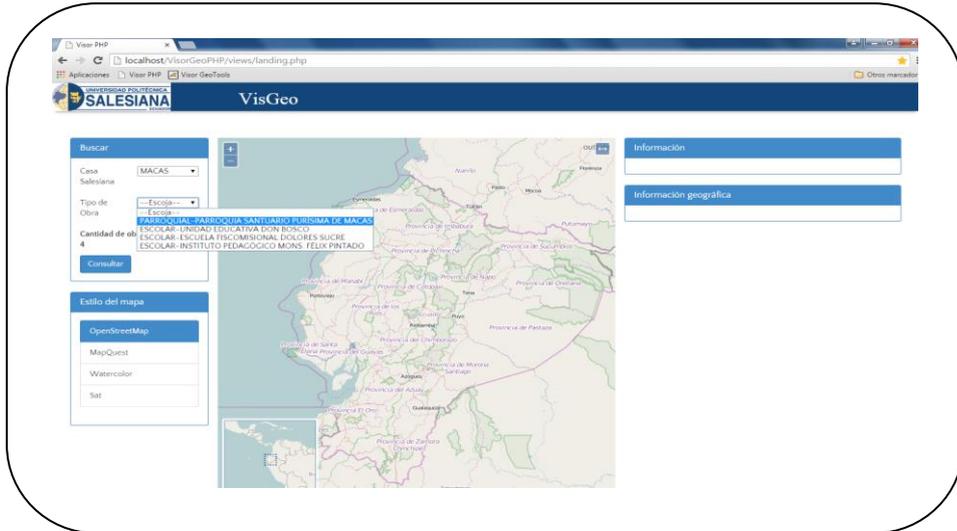
## Manual de usuario



*Figura 40.* Seleccionar casa salesiana  
Elaborado por : Jessica Escudero & Natali Guatani

3. Seleccionar tipo de obra las cuales ya están definidas para la casa salesiana seleccionada y presionar consultar para ver el resultado en el mapa.

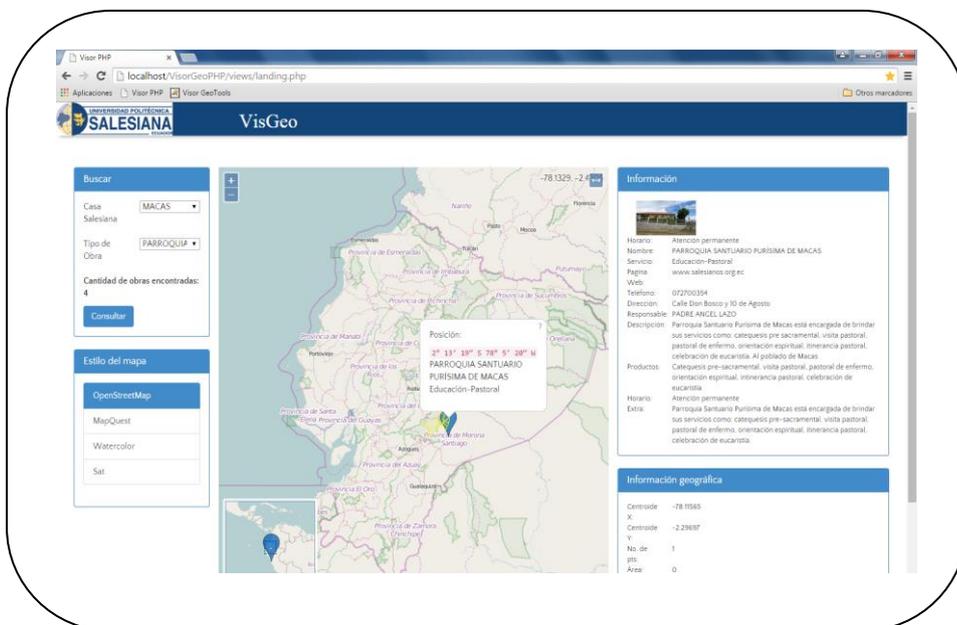
## Manual de usuario



*Figura 41.* Seleccionar tipo de obra  
Elaborado por : Jessica Escudero & Natali Guatapi

4. Localización en el mapa de la obra buscada de acuerdo a la casa salesiana seleccionada. A su derecha se encontrará la información y la información geográfica al seleccionar sobre ella el área en el mapa.

## Manual de usuario



*Figura 42.* Localización en el mapa de la obra  
Elaborado por : Jessica Escudero & Natali Guatapi

## 5. Selección de tipo de mapa.

Para indicar un tipo de mapa solo se tiene que presionar el tipo deseado

### Manual de usuario

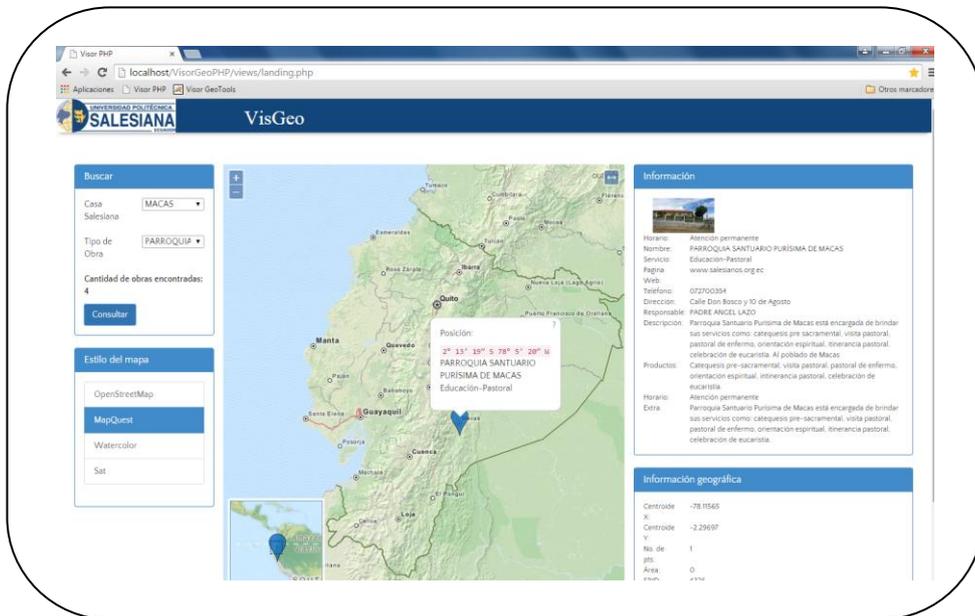


Figura 43. Seleccionar tipo de mapa MapQuest  
Elaborado por : Jessica Escudero & Natali Guatapi

### Manual de usuario

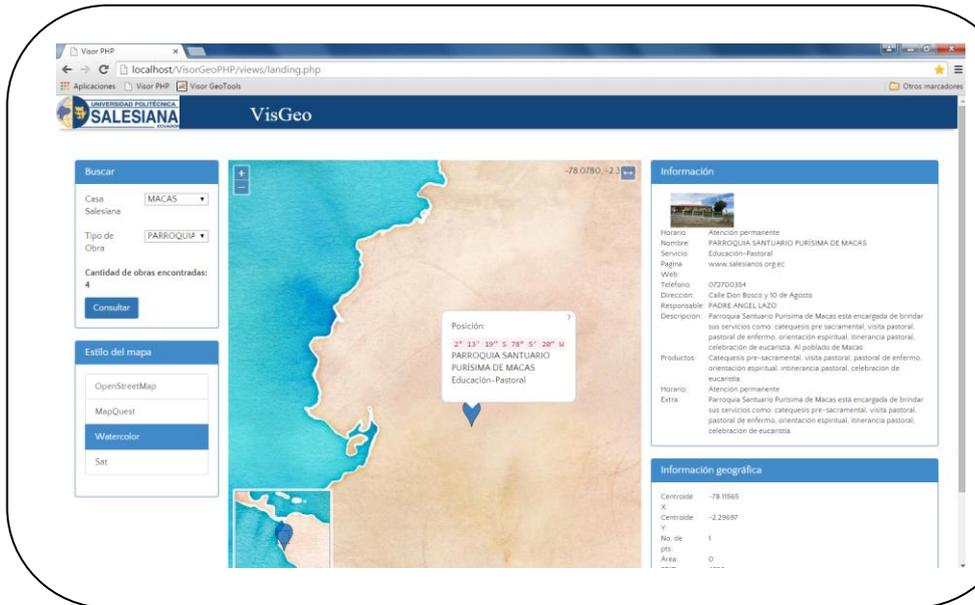


Figura 44. Seleccionar tipo de mapa Watercolor  
Elaborado por : Jessica Escudero & Natali Guatapi

## Manual de usuario

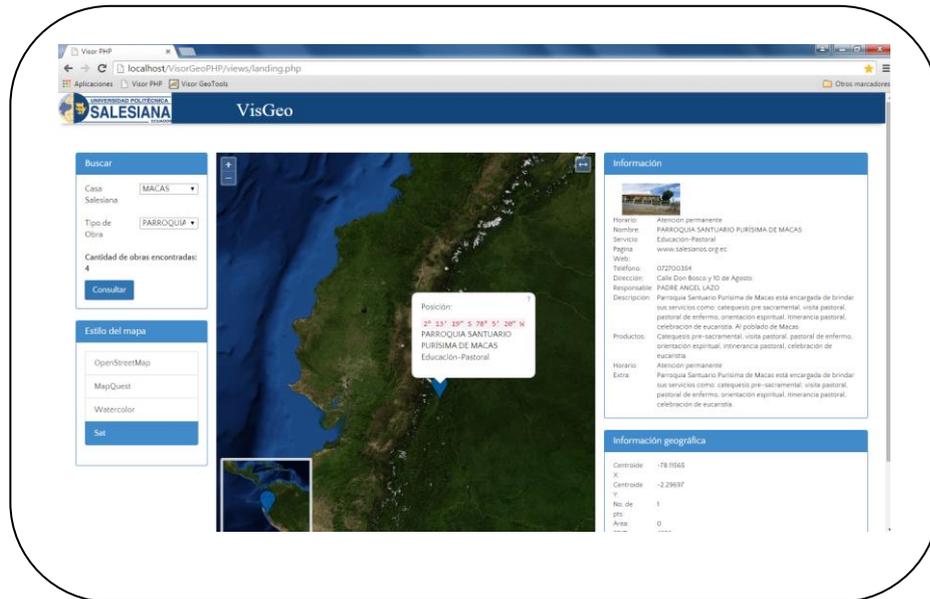


Figura 45. Seleccionar tipo de mapa Sat

Elaborado por : Jessica Escudero & Natali Guatapi

## 6. Maximizar mapa.

Dar clic en  para Maximizar mapa.

## Manual de usuario

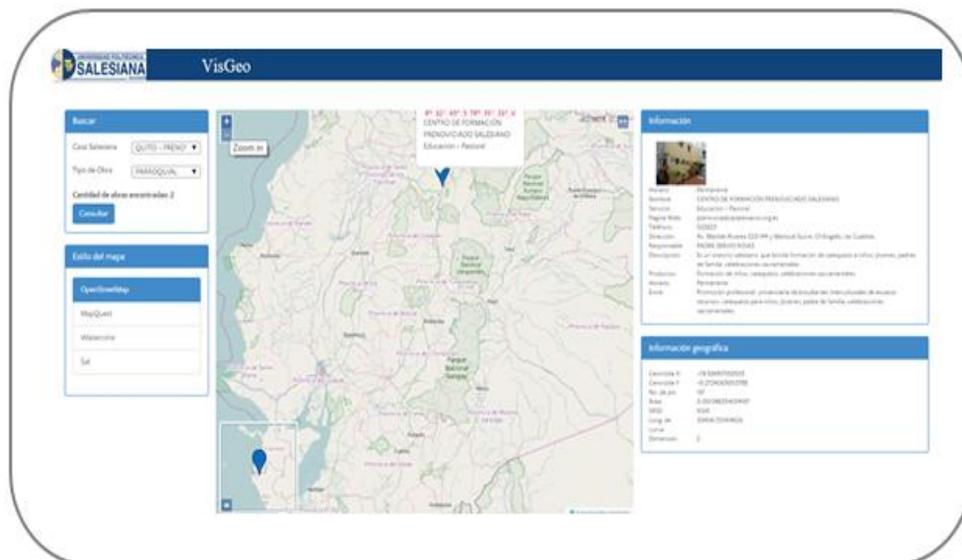


Figura 46. Maximizar mapa.

Elaborado por : Jessica Escudero & Natali Guatapi

7. Dar clic en para Minimizar mapa

Manual de usuario

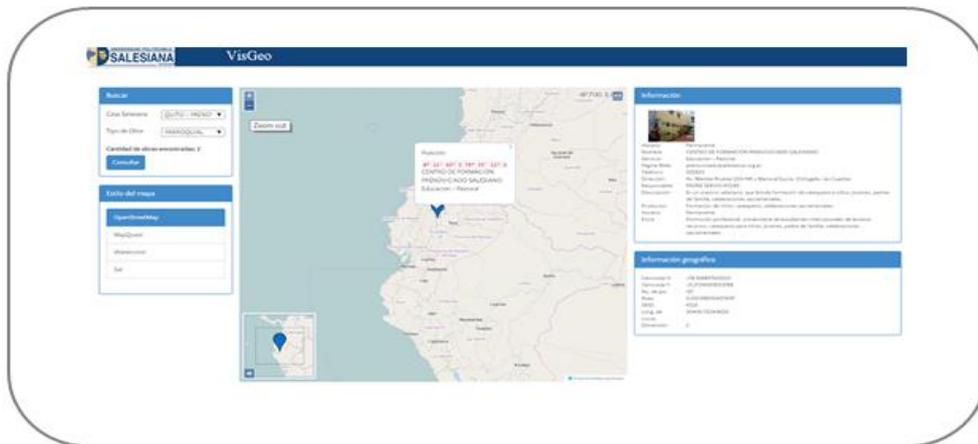


Figura 47. Minimizar mapa.  
Elaborado por : Jessica Escudero & Natali Guatapi

8. Dar clic en Toggel Foll-screen  que se encuentra en la parte superior para pantalla completa del mapa con la que solamente se observa en el prototipo el mapa

Manual de usuario

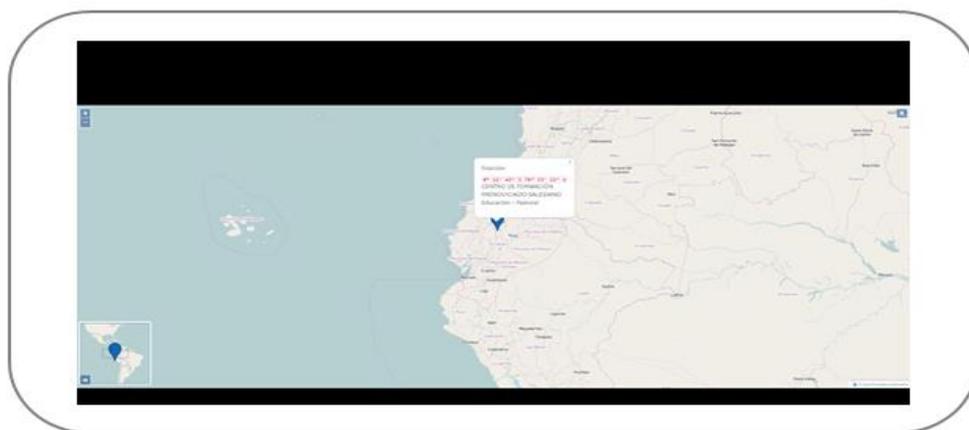


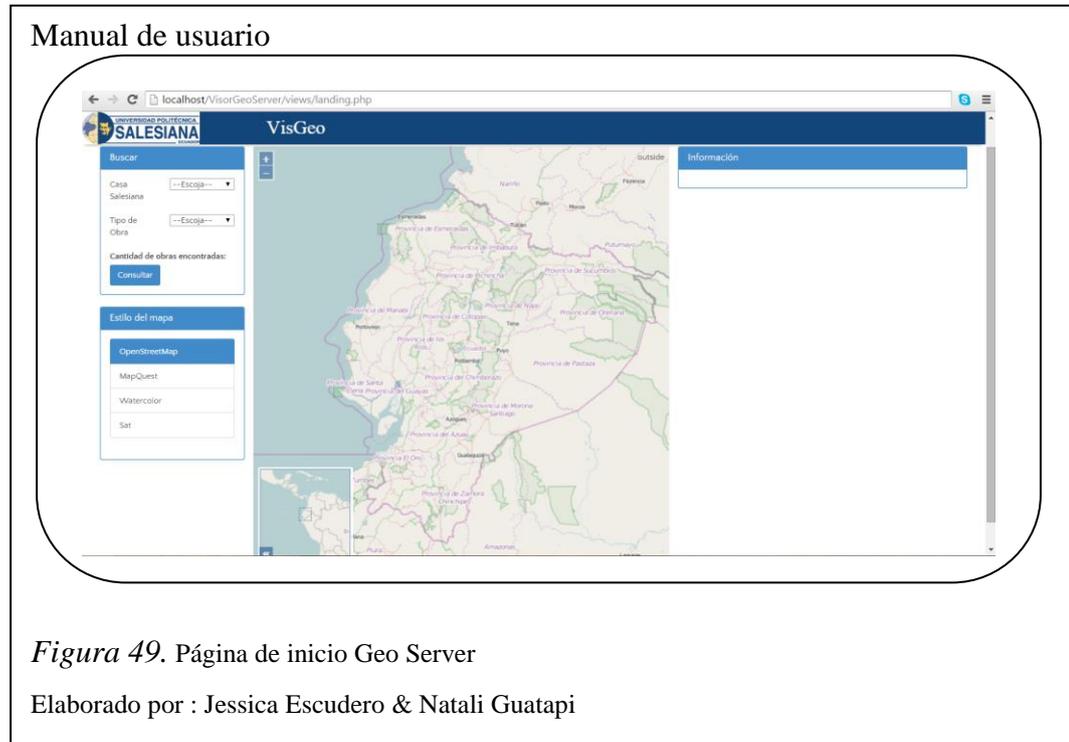
Figura 48. Pantalla completa.  
Elaborado por : Jessica Escudero & Natali Guatapi

9. Presionar la tecla ESC para salir de la pantalla completa del mapa.

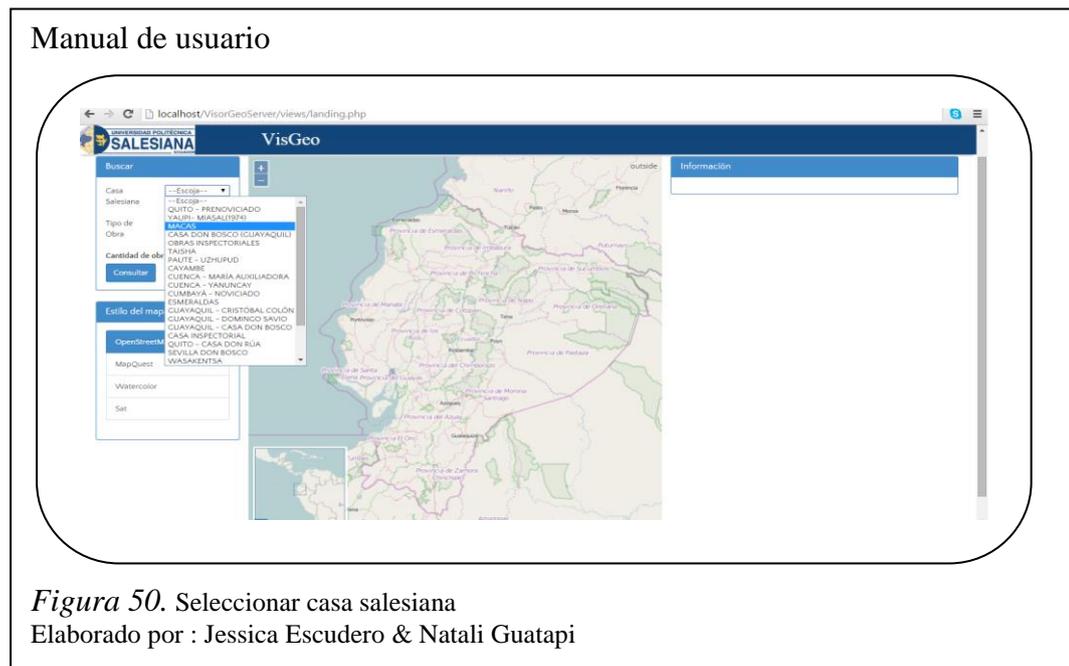
### 4.3.2 Manual de usuario para GeoServer

#### 1. Abrir pantalla de inicio

Acceder a la siguiente dirección <http://ide.ups.edu.ec/VisorGeoServer/> para el visualizador prototipo de GeoServer.

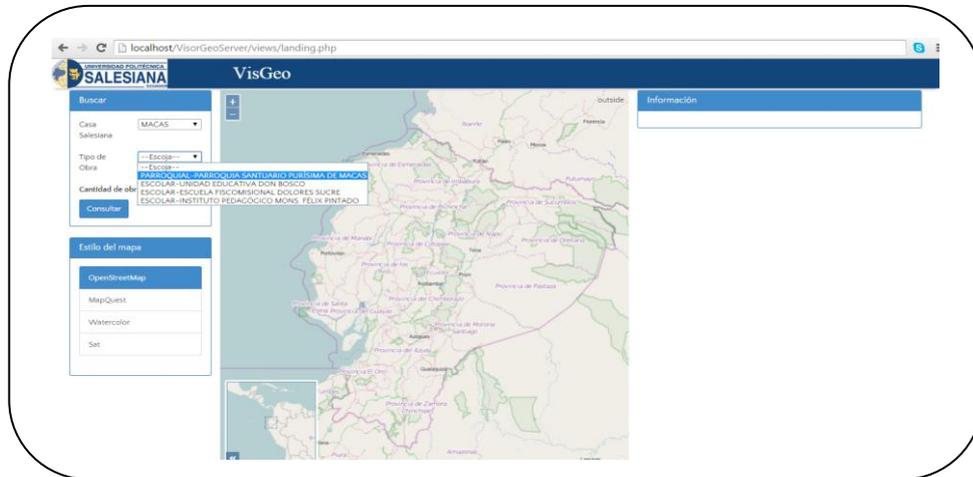


#### 2. Seleccionar casa salesiana desplegando el combobox casa salesiana.



3. Seleccionar tipo de obra las cuales ya están definidas para la casa salesiana seleccionada y presionar consulta para ver el resultado en el mapa.

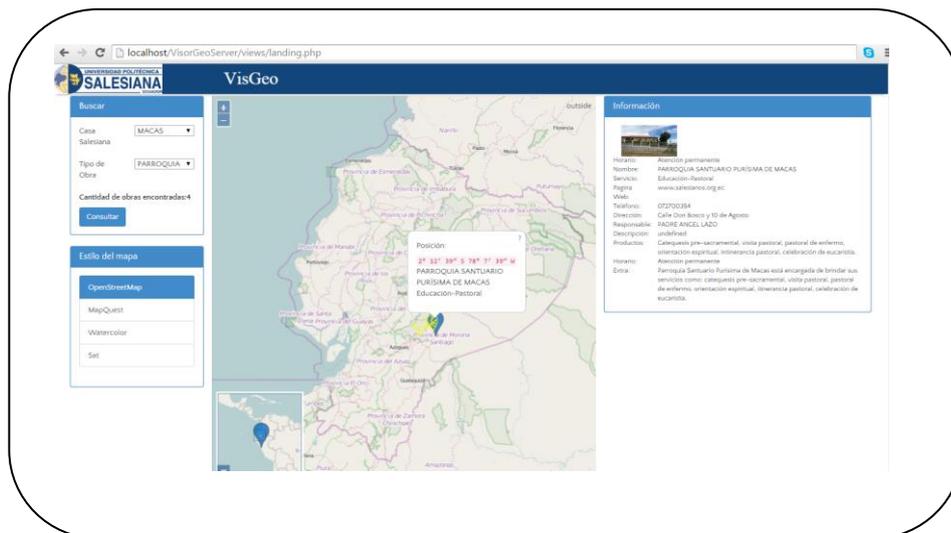
### Manual de usuario



*Figura 51.* Seleccionar tipo de obra  
Elaborado por : Jessica Escudero & Natali Guatapi

4. Localización en el mapa de la obra buscada de acuerdo a la casa salesiana seleccionada. Dando clic sobre ella a su derecha nos permite visualizar la posición y la información de la misma.

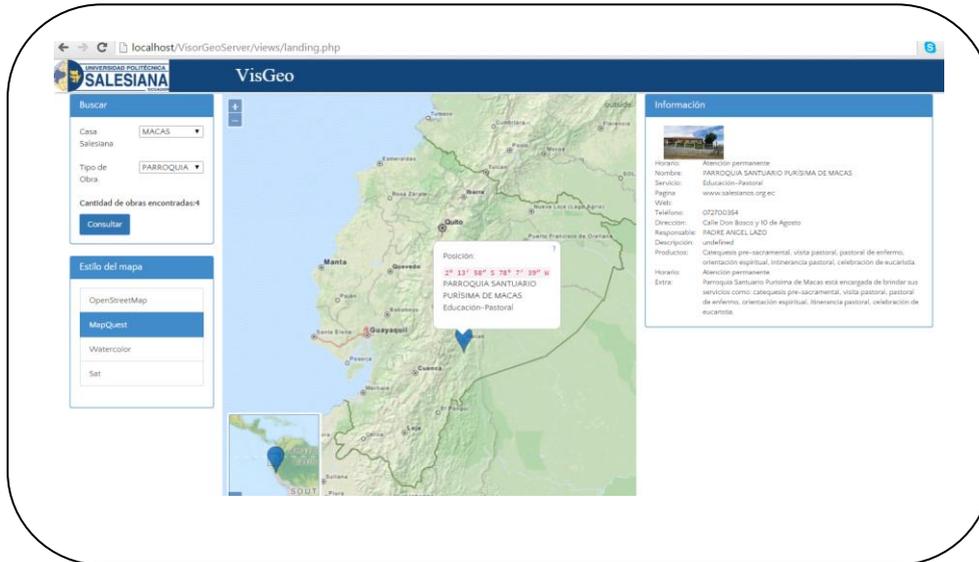
### Manual de usuario



*Figura 52.* Localización en el mapa la obra  
Elaborado por : Jessica Escudero & Natali Guatapi

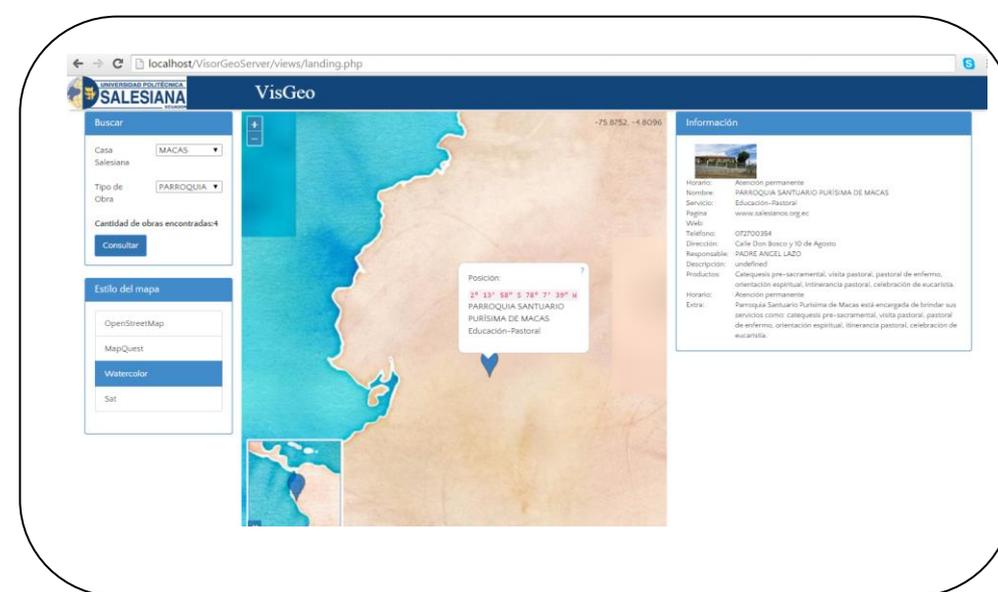
## 5. Selección de tipo de mapa.

### Manual de usuario



*Figura 53.* Seleccionar tipo de mapa MapQuest  
Elaborado por : Jessica Escudero & Natali Guatapi

### Manual de usuario



*Figura 54.* Seleccionar tipo de mapa Watercolor  
Elaborado por : Jessica Escudero & Natali Guatapi

## Manual de usuario

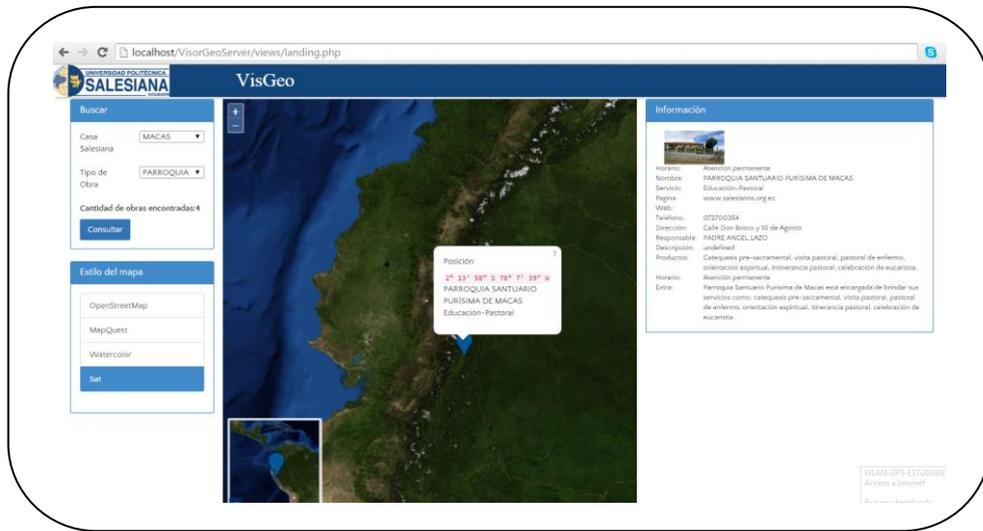


Figura 55. Seleccionar tipo de mapa Sat  
Elaborado por : Jessica Escudero & Natali Guatapi

## 6. Maximizar mapa

Dar clic en  para Maximizar mapa.

## Manual de usuario

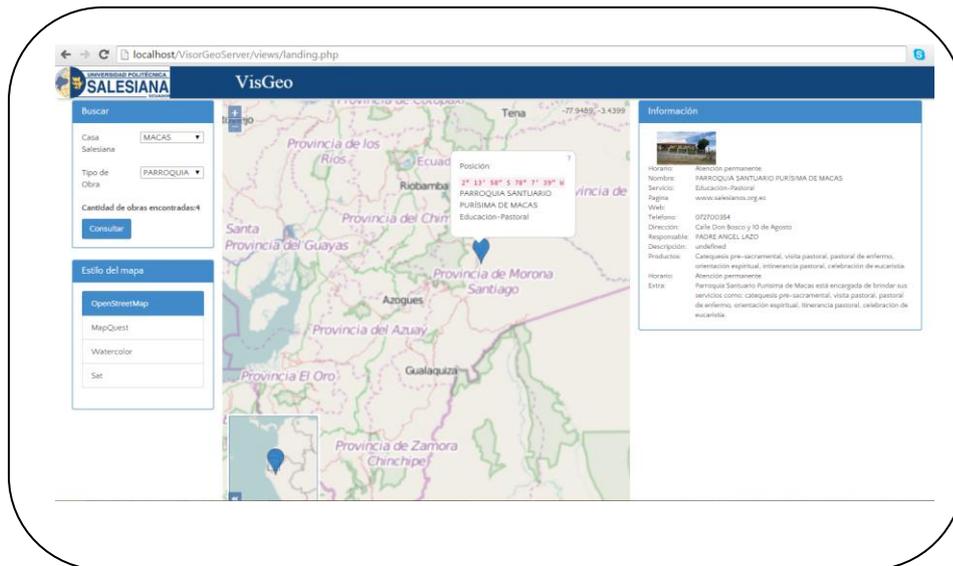


Figura 56. Maximizar mapa.  
Elaborado por : Jessica Escudero & Natali Guatapi

7. Dar clic en para Minimizar mapa

Manual de usuario

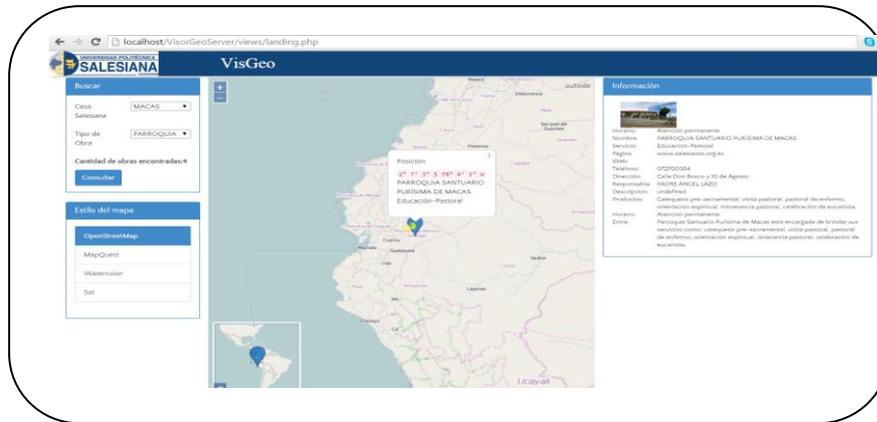


Figura 57. Minimizar mapa.  
Elaborado por : Jessica Escudero & Natali Guatapi

8. Dar clic en Toggele Foll-screen  que se encuentra en la parte superior para pantalla completa del mapa con la que solamente se observa en el prototipo el mapa.

Manual de usuario

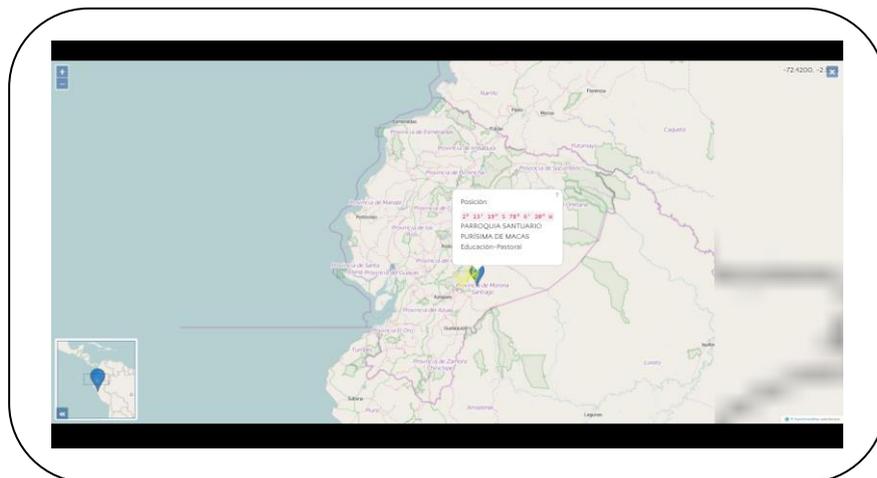


Figura 58. Pantalla completa.  
Elaborado por : Jessica Escudero & Natali Guatapi

9. Presionar la tecla ESC para salir de la pantalla completa del mapa.

## CONCLUSIONES

- Se determinó que para incorporar información geográfica en el gestor de base de datos PostgreSQL, se necesita instalar el componente Postgis que contiene un esquema denominado topology la misma que contiene todas las funciones y tablas asociadas con este módulo para manejar información georeferenciada.
- Geodatabase y Geoserver no son librerías de código abierto; ya que GeoServer es un servidor de mapas y Geodatabase permite administrar datos espaciales que se encuentran almacenados dentro de Postgis que es un gestor de base de datos.
- El visualizador de GeoPHP es netamente realizado en el lenguaje de PHP el mismo que maneja clases y métodos estáticos, mientras que GeoTools es una librería 100% java con sub-librerías específicas para web que permite la visualización de los contenidos geoespaciales.
- GeoTools y GeoPHP son librerías de código abierto que contienen sub-librerías de geometría los cuales permiten obtener la información geográfica con la visualización del cálculo del área, centroide y dimensión además existen más funcionalidades que se encuentran en el Anexo 2 y 3.
- Para visualizar los datos en el estilo del prototipo “GeoServer” es necesario copiar la Url de las vistas creadas en el servidor de mapas Geoserver previa conexión a la base de datos PostGis para el satisfactorio manejo de la información.
- De acuerdo con los resultados previstos por la herramienta Jmeter se pudo determinar que las pruebas de rendimiento con 10 usuarios concurrentes en un servidor con las características de 3 GB de RAM, 32 bits, procesador Inter Core 2Duo E6550 a 2.33 GHz es aceptable con una tasa de error de cero como se muestra en las Fig. 32,35,38. En función de estos resultados se

puede proyectar que el servidor puede incrementar sus propiedades en cuanto al procesador en 4 núcleos reales y 4 Gb de RAM.

- En función de los objetivos se desarrolló los respectivos manuales de usuario para los visualizadores prototipos GeoServer, GeoTools y GeoPHP, que ayudan al manejo de los mismos.
- En función de los objetivos planteados se implementaron los tres visualizadores prototipos GeoServer, GeoTools y GeoPHP, en el servidor del CIMA UPS satisfactoriamente una vez superado los inconvenientes de las versiones del servidor.

## RECOMENDACIONES

- Utilizar los manuales de usuario como guía para el adecuado manejo de los visualizadores prototipos GeoServer, GeoPHP y GeoTools.
- Durante la construcción de los prototipos de visualizadores GeoTools y GeoPHP se utilizó únicamente las sub-librerías de geometría que se encuentran en las *Tablas 7 y 8* por lo que es recomendable que se emplee y experimenten las demás funcionalidades que tiene cada una de estas librerías.
- Se recomienda verificar, actualizar si es necesario las versiones de las herramientas que se requiere al implementar en el servidor CIMA y así evitar futuros inconvenientes al momento de la implementación.
- Utilizar la librería de GeoPHP ya que trabaja con el lenguaje PHP el mismo que es muy útil para diseñar de forma rápida aplicaciones Web.
- Se sugiere utilizar la librería de openLayer ya que permite la manipulación y visualización de los mapas y estilo de los mismos.

## LISTA DE DEFERENCIAS

- Alegsa, S. F. (1998). Diccionario de informática. Recuperado el 4 de 10 de 2013, de Diccionario de informática: <http://www.alegsa.com.ar/Dic/biblioteca.php>
- Alegsa, Santa Fé, Argentina. (1998). Obtenido de <http://www.alegsa.com.ar/Dic/biblioteca.php>
- Alvarez, C. (2014). ¿Qué es Maven? Genbeta, 10.
- Avelaño, R. N. (2010). Bibliotecas estáticas y dinámicas. Veracruz, México: Instituto Tecnológico Superior de Coatzacoalcos.
- Clarís, P. (2013). Softeng Software Engineers. Recuperado el 31 de agosto de 2013, de Softeng Software Engineers: <http://www.softeng.es/es-es/empresa/metodologias-de-trabajo/metodologia-scrum/proceso-roles-de-scrum.html>
- Deolivera, J. (09 de Octubre de 2007). Jira Codehaus. Recuperado el 10 de Enero de 2015, de <http://jira.codehaus.org/browse/GEOS-1240>
- Desarrollo-Web. (2002). Desarrollo Web. Recuperado el 8 de Enero de 2015, de <http://www.desarrolloweb.com/articulos/711.php>
- Dueñas, J. B. (12 de febrero de 2015). Alcance Libre. Recuperado el 25 de febrero de 2015, de Configuración básica de Apache: <http://www.alcance Libre.org/staticpages/index.php/como-apache>
- F.González, V. (s.f.). Sigte. Recuperado el 6 de 11 de 2013, de Sigte: [http://www.sigte.udg.edu/jornadassiglibre2013/uploads/articulos\\_13/a32.pdf](http://www.sigte.udg.edu/jornadassiglibre2013/uploads/articulos_13/a32.pdf)
- Fernández, L., & Lara, P. (2010). Generación de casos de prueba a partir de especificaciones UML. Celia Gutiérrez: Universidad Europea de Madrid.
- Font, O. (2010). Generación automática de visualizadores web. I Jornada Ibérica de infraestructuras de datos Espaciales, 1.
- Foundation, T. A. (2015). Apache Tomcat. Recuperado el 1 de Marzo de 2015, de apache Tomcat: <http://tomcat.apache.org/>

- Garza, H. (2008). Rastreo Satelital. Recuperado el 02 de octubre de 2013, de Rastreo Satelital: <http://www.hergarza.com.mx/Vikingo.html>
- geográfica, s. d. (2013). Técnicas básicas para estudios de biodiversidad. Recuperado el 10 de septiembre de 2013, de Técnicas básicas para estudios de biodiversidad: [http://www.gbif.es/ficheros/Guion\\_SIG.pdf](http://www.gbif.es/ficheros/Guion_SIG.pdf)
- GeoPHP. (2014). GeoPHP.net. Recuperado el 6 de Enero de 2015, de <https://geophp.net/>
- GeoServer. (2014). GeoServer.org. Recuperado el 7 de Enero de 2015, de <http://geoserver.org/>
- GeoTools. (2014). GeoTools.org. Recuperado el 6 de Enero de 2015, de <http://geotools.org/about.html>
- Geotools, C. (2012). OsGeoLive. Recuperado el 5 de 11 de 2013, de OsGeoLive: [http://live.osgeo.org/es/overview/geotools\\_overview.html](http://live.osgeo.org/es/overview/geotools_overview.html)
- Heriberto, A. (2013). Tesis de grado. En A. Heriberto, Análisis y diseño de un Sistema de Información Geográfica GIS (pág. 142). quito.
- IBM. (2015). IBM. Rational Team Concert for Scrum Projects. Recuperado el 8 de Enero de 2015, de [https://www.ibm.com/developerworks/community/wikis/home?lang=en#!/wiki/Rational%20Team%20Concert%20for%20Scrum%20Projects/page/SCRUM%20como%20metodolog%C3%ADa?section=\\_ftn1](https://www.ibm.com/developerworks/community/wikis/home?lang=en#!/wiki/Rational%20Team%20Concert%20for%20Scrum%20Projects/page/SCRUM%20como%20metodolog%C3%ADa?section=_ftn1)
- Instituto Geográfico Militar del Ecuador. (2013). Instituto Geográfico Militar. Recuperado el diez de octubre de 2013, de Instituto Geográfico Militar: <http://www.geoportaligm.gob.ec/portal/index.php/descargas/software-de-descargas/>
- Java. (2006). Java. Recuperado el 16 de octubre de 2013, de Java: [http://www.java.com/es/download/faq/whatis\\_java.xml](http://www.java.com/es/download/faq/whatis_java.xml)
- Jucrap. (9 de abril de 2011). Analista de sistemas y organización de las mismas. Recuperado el 25 de septiembre de 2013, de Analista de sistemas y

organización de las mismas: <http://jucrap.blogspot.com/2011/04/modelo-scrum.html>

Letham, L. (2001). GPS fácil Uso del sistema de posicionamiento global. En L. Letham, GPS Made Easy (pág. 284). España: Paidotribo.

Llopis, J. P. (2008). Sistemas de información geográfica aplicados a la gestión del territorio. En J. P. Llopis, Entrada, manejo, análisis y salida de datos espaciales (pág. 310). Alicante: Club Universitario.

López-Vázquez, M. A.-P. (2012). Fundamentos de las Infraestructuras de Datos Espaciales (IDE). Dinama: BibliotecaOnline SL.

Martín, M. (2013). Manual PostGis. Recuperado el 12 de octubre de 2013, de Manual PotGis: <http://postgis.refractory.net/documentation/postgis-spanish.pdf>

Martinez, R. (2010). PostgreSQL. Recuperado el 23 de octubre de 2013, de PostgreSQL: [http://www.postgresql.org/es/sobre\\_postgresql](http://www.postgresql.org/es/sobre_postgresql)

Microsoft. (2010). Microsoft Developer Network. Recuperado el 7 de Enero de 2015, de <http://msdn.microsoft.com/es-es/library/cc507701.aspx>

Modelo original de Scrum para desarrollo de software. (05 de marzo de 2013). Recuperado el 27 de septiembre de 2013, de Modelo original de Scrum para desarrollo de software: [http://www.scrummanager.net/bok/index.php?title=Modelo\\_original\\_de\\_Scrum\\_para\\_desarrollo\\_de\\_software](http://www.scrummanager.net/bok/index.php?title=Modelo_original_de_Scrum_para_desarrollo_de_software)

Morales, A. (26 de Septiembre de 2012). MappingGIS.com. Recuperado el 7 de Enero de 2015, de <http://mappinggis.com/2012/09/aplicaciones-gis-open-source/>

Pérez Lamancha, B. (2008). Proceso de Testing funcional independiente. Motevideo: Universidad de la república, Uruguay.

Phayes. (03 de marzo de 2012). Geophp. Recuperado el 14 de julio de 2013, de Geophp: <https://drupal.org/project/geophp>

- PHP. (2014). PHP.net. Recuperado el 7 de Enero de 2015, de <http://php.net/>
- PostGIS. (2014). PostGis.net. Recuperado el 7 de Enero de 2015, de <http://postgis.net/features>
- Roset, R., & N, R. (2012). Georeferenciación de mapas antiguos con herramientas de código abierto. Revista catalana de geografia, <http://www.rcg.cat/articles.php?id=237>.
- Senplades. (2013). Geoserver. Recuperado el 6 de julio de 2013, de Geoserver: [http://www.geoportaligm.gob.ec/portal/?wpfb\\_dl=25](http://www.geoportaligm.gob.ec/portal/?wpfb_dl=25)
- Trigas, M. (s.f.). Gestión de Proyectos Informáticos. En G. d. Informáticos, Metodología Scrum (pág. 55). Recuperado el 15 de noviembre de 2013, de <http://openaccess.uoc.edu/webapps/o2/bitstream/10609/17885/1/mtrigasTFC0612memoria.pdf>
- Urrutia, J. (2006). Cartografía orientación y GPS. En J. Urrutia, Cartografía orientación y GPS (pág. 19). España: etor-ostoa s.i.
- Zapata, C. (26 de Noviembre de 2011). Mantenimiento de una computadora. Obtenido de <http://mantenimientosdeunapc.blogspot.com/2011/11/que-es-xampp-y-para-que-sirve.html>

## ANEXOS

### **Anexo 1.** Tipo de metadatos

- **Descriptivos:** Permiten la descripción e identificación de recursos de información en el nivel local facilitando la búsqueda y la recuperación, en el nivel web permite a los usuarios descubrir recursos.
- **Estructurales:** Facilitan la navegación y presentación de recursos electrónicos ya que proporcionan información sobre la estructura interna de los recursos, incluyendo página, sección, capítulo, numeración, índices, y tabla de contenidos; también describen la relación entre los materiales.
- **Administrativos:** Facilitan la gestión y procesamiento de las colecciones digitales tanto a corto como a largo plazo incluyen datos técnicos sobre la creación y el control de calidad; incluyen gestión de derechos y requisitos de control de acceso y utilización; información sobre acción de preservación.

## Anexo 2. Principales métodos de la librería GeoPHP

Método	Descripción	Devuelve
<b>Out</b>	Emite la geometría en el formato de adaptador especificado. Los formatos disponibles son <i>wkt</i> , <i>wkb</i> , <i>json</i> , <i>KML</i> , <i>GPX</i> , <i>google_geocode</i>	String
<b>Área</b>	El área de este polígono (o GeometryCollection), tal como se mide en el sistema de referencia espacial de la geometría	Float
<b>Boundary</b>	Devuelve el cierre de la frontera combinatoria de este objeto geométrico.	LinearRing
<b>Envelope</b>	El cuadro de límite mínimo para esta geometría, regresó como Geometría.	Polygon
<b>getBBox</b>	El cuadro de límite mínimo para esta geometría, devuelve como una matriz.	Array
<b>Centroid</b>	El centroide matemática para esta geometría como un punto. Para polígonos, el resultado no está garantizado a ser interior.	Point
<b>Length</b>	La longitud de esta curva en su referencia espacial asociado.	Float
<b>greatCircleLength</b>	La longitud de esta curva en la tierra, devuelve metros.	Float
<b>haversineLength</b>	La longitud de la curva .	Float
<b>Y</b>	El valor de la coordenada y de este punto.	Float

<b>X</b>	El valor de la coordenada x de este punto.	Float
<b>numGeometries</b>	El número de geometrías de componentes en esta colección	Integer
<b>geometryN</b>	Devuelve la geometría de N en esta colección. Tenga en cuenta que el índice comienza en 1.	Geometry
<b>isRing</b>	Devuelve 1 (TRUE) si esta curva es cerrada () y esta IsSimple Curva ().	Boolean
<b>isClosed</b>	Devuelve 1 (TRUE) si la curva es cerrada. StartPoint () == EndPoint ().	Boolean
<b>getComponents</b>	Obtener todos los componentes sub-geometría de la geometría	Array of geometries
<b>numPoints</b>	El número de puntos en este LineString	Integer
<b>pointN</b>	Devuelve el Punto N especificado en esta cadena lineal. Tenga en cuenta que el índice comienza en 1.	Point
<b>exteriorRing</b>	Devuelve el anillo exterior de este polígono.	LineString
<b>numInteriorRings</b>	Devuelve el número de anillos interiores en este polígono.	Integer
<b>interiorRingN</b>	Devuelve el anillo interior de enésima de este polígono como una cadena lineal. Tenga en cuenta que el índice comienza en 1.	LineString
<b>Dimensión</b>	La dimensión inherente de este objeto geométrico. En las colecciones no homogéneas, esto devolverá la mayor dimensión topológica de los objetos contenidos.	Integer

<b>geometryType</b>	Devuelve el nombre del subtipo crear instancias de la geometría de la que este objeto geométrico es un miembro instanciable. El nombre del subtipo de la geometría se devuelve como una cadena.	String
<b>SRID</b>	Devuelve el ID de sistema de referencia espacial para este objeto geométrico.	Integer
<b>setSRID</b>	Establezca el ID de sistema de referencia espacial para este objeto geométrico.	NULL
<b>asArray</b>	Obtener la geometría dada como un conjunto de componentes (recursivo)	Array
<b>getGeoInterface</b>	Obtener el geometryType y coordina en forma de matriz	Array
<b>isEmpty</b>	TRUE si esta geometría no contiene vértices	Boolean

Tomado de: [GeoPHP.net](http://GeoPHP.net)

### Anexo 3. Descripción de las librerías que conforman GeoTools

<b>Librería</b>	<b>Descripción</b>
<b>gt-render</b>	Implementación de la renderización del mapa usando Java 2D.
<b>gt-jdbc</b>	Adquisición de datos espaciales desde bases de datos relacionales (ej: PostGIS, Mysql, Oracle).
<b>gt-data</b>	Adquisición de datos espaciales desde fuentes diversas (ej: Shapefiles).
<b>gt-xml</b>	Implementación de datos espaciales en formato XML (XSD, WMS).
<b>gt-cql</b>	Implementación del CQL (Common Query Language)
<b>gt-main</b>	Clases base de manipulación de datos espaciales (Feature, Filter, ...)
<b>gt-api</b>	Definición de interfaces básicas para trabajar con datos espaciales.
<b>Jts</b>	Clases para la manipulación de geometrías.
<b>gt-coverage</b>	Acceso a datos ráster.
<b>gt-referencing</b>	Localización de coordenadas y transformación de las mismas.
<b>gt-metadata</b>	Identificación y descripción de datos espaciales.
<b>gt-opengis</b>	Interfaces para conceptos espaciales comunes.

Tomado de: [GeoTools.org](http://GeoTools.org)