Geo-Almacén de datos geográficos

M.A. Manso-Callejo, E. Castaneda

Grupo de Investigación MERCATOR
Universidad Politécnica de Madrid (UPM)
m.manso@upm.es, emecas@ieee.org

Resumen

La publicación y el uso compartido de los datos geográficos obtenidos y compartidos por usuarios nóveles o neófitos (Neo Geografía) bajo estándares Geográficos (OGC, ISO TC211) suponen una barrera infranqueable: necesidad de infraestructura (hardware, software y comunicaciones). Algunas iniciativas como *OpenStreetMap* (OSM), *ikiMap, Tinymap, TargetMap* o *GeoNode* hacen posible que cualquier usuario de un modo voluntario pueda compartir sus datos. Cada una de ellas adolece de ciertas limitaciones que les impide cubrir el espectro de necesidades y escalar cuando el número de usuarios del sistema crece. En este documento se presenta una solución en desarrollo, basada en *Open Source*, que trata de solventar algunas de las limitaciones detectadas, concretamente la escalabilidad de las soluciones basadas en *GeoServer* y *MapServer*, así como otras propias de la variedad de formatos de datos que se pueden compartir.

Palabras clave: Geo-Almacén, OGC, *Volunteered Geographic Information* (VGI), WMS, WFS, WCS, CS-W.

1 Introducción

La Información Geográfica (IG) cada vez toma mayor relevancia en los procesos de toma de decisiones, la gestión de los recursos naturales, la implantación de nuevas infraestructuras, etc. llegando algunos autores [1] a cuantificar el número de actividades en las que las decisiones políticas hacen uso de la misma. La evolución y la historia de la IG hoy, antes Cartografía, ha restringido su uso al plano político e ingenieril, sin embargo gracias a la evolución de las tecnologías de la información y las comunicaciones, está motivando la aparición de movimientos sociales partidarios de que los datos estén al alcance de los ciudadanos, que los propios ciudadanos puedan crear y compartir aquellos

datos que no se encuentran accesibles o disponibles en la red. Estos movimientos sociales enmarcados en el concepto de Neo-Geografía o Neo-Cartografía [2], tienen su fundamento en un movimiento previo relacionado con el software (*Open Source*).
Algunos autores de referencia en el mundo de la IG, como Michael GoodChild [3] bautizaron este fenómeno como información Geográfica Voluntaria (VGI) en 2007. Desde entonces se han descrito varios ejemplos de VGI en la literatura, como casos de uso e inspiración para estudiar cómo las Infraestructuras de Datos Espaciales (IDE) pueden hacerse eco para tratar de mejorar las relaciones de colaboración inter o intra organizaciones. Este es el caso de los estudios que están desarrollando Grus, et al. [4], o Giorgiadou y Stoter [5] definiendo modelos conceptuales para estudiar la colaboración en los entornos IDE o el uso de la información en los gobiernos.

Desde el punto de vista del movimiento VGI, el caso de éxito más descrito es *OpenStreetMap*, en el que los ciudadanos de modo libre y voluntario deciden capturar cierta información mediante unos receptores GPS, anotar nombres de viales y compartirlo con el resto de la comunidad de usuarios [6]. Otros usuarios simplemente se limitan a digitalizar los viales sobre una imagen aérea como las proporcionadas por *Google Maps*™.

En este trabajo, se analiza las capacidades de algunas iniciativas, usualmente privadas, que proporcionan herramientas para que los ciudadanos puedan compartir los datos geográficos que generen, o que tengan disponibles. Cada iniciativa persigue unos fines concretos, y en muchos casos, ofrecen su infraestructura para que puedan compartir estos datos, visualizarlos conjuntamente con otros, pero limitan bien la capacidad de interoperar con ellos o la de acceder y descargarlos para usarlos en local en un ordenador de sobremesa. Los principales inconvenientes de estas aplicaciones residen en las soluciones adoptadas: limitando el nº de formatos de datos que un usuario puede compartir, limitando la capacidad de explotar estos datos, la carencia de información que describa los datos (metadatos) o el uso de estándares geoespaciales. Una sola solución supera parcialmente estas limitaciones (GeoNode), si bien adolece de dos problemas importantes bajo nuestro punto de vista: escalabilidad y limitación del nº de formatos de datos que se pueden compartir. La solución que aquí se presenta, primero trata de enmarcar el problema y proponer una arquitectura de componentes software de tipo *Open Source*, sobre los que desarrollar un conjunto de servicios, y *wrappers* de servicios, que permitan cumplir con las expectativas de lo que hemos denominado Geo-Almacén de datos geográficos.

2. Trabajos relacionados

Entre los trabajos relacionados se encuentra ikiMap (http://www.ikimap.com). Esta iniciativa permite compartir datos, exclusivamente vectoriales en formatos KML (plano o comprimido), Shapefiles y GPX. El usuario puede dibujar sus propios mapas con las herramientas que el sitio ofrece. Se gestiona la autoría de los datos mediante un conjunto de políticas de usuarios y existe la posibilidad de comunicar el uso fraudulento de los datos (problemas con autoría, licencia, etc.). Cuando se publican datos se han de etiquetar/catalogar y describir narrativamente de modo resumido. El usuario que publica los datos decide si éstos se pueden descargar o no. Los datos almacenados en el servidor se obtienen usando un servicio intermediario al que se le indica el número de la capa, y se obtiene un documento KML o KMZ. Permite describir cada elemento de una capa con sus atributos, enlazarlo con páginas web, etc. La herramienta disponible permite editar (añadir y borrar) elementos.

Otra iniciativa identificada es Tinymap (tinymap.net). Ésta permite construir tus mapas, como una aplicación de *MashUp* sobre Google, generando marcas de tipo puntual o líneas. A cada elemento editado se le puede asignar un icono y un texto descriptivo. Cuando se ordena publicar la información, la herramienta facilita un URL que identifica y señala el mapa generado. También permite descargar los datos en formato KML. Para poder crear un mapa no es necesario estar registrado en la plataforma, sin embargo para editar o borrar un elemento sí es necesario. Este hecho genera una *paradoja*, ¿Cómo borrar un elemento que se creó sin estar registrado?

TargetMap (http://www.targetmap.com/) es una aplicación web que permite generar mapas y compartirles con el resto de usuarios. A diferencia de los anteriores, esta plataforma ofrece las bases geométricas de las divisiones administrativas para que el usuario aporte los datos que pretende representar, es decir funciona como un servicio *Table Joint Service* (TJS), aunque no estandarizado. El formato usual soportado para representar los datos es una hoja de cálculo *Excel*, en cuya primera hoja se encuentren los datos, de tal forma que la primera fila contenga los nombres de las columnas.

El principal antecedente de Geo-Almacén identificado y relacionado con nuestra propuesta es geoNode (http://www.geonode.org) -anteriormente CAPRA. Este

proyecto *Open Source* permite que los usuarios se den alta y se identifiquen ante el sistema. Una vez identificados puedan publicar mediante servicios OGC (WMS, WFS y WCS) sus datos almacenados en ficheros (*dataStore: Shapefile, coverageStore: GeoTiff*). Permite y obliga a definir un conjunto mínimo de metadatos para describir los datos, publicándolo en un servicio de catálogo estandarizado (OGC-CSW). Mediante una simple interfaz de consulta se pueden localizar datos disponibles que responden a la consulta (cliente de catálogo). Con la información obtenida, se muestran los metadatos mínimos (*brief*) y se facilita el acceso a los datos para su descarga en una variada lista de formatos. También se puede acceder al conjunto de metadatos creado (un perfil propio de metadatos) para aprender más sobre los datos. Finalmente, si el usuario así lo desea, puede añadir la capa de datos localizada a un visor cartográfico (cliente de servicios WMS).

3 Definición y justificación del Geo-Almacén

El Geo-Almacén que proponemos en este trabajo, pretende que los usuarios puedan publicar sus datos, ofreciendo la mayor variedad de formatos posible, y facilitar la creación de sus metadatos.

3.1 Metodología utilizada

Para alcanzar el objetivo identificado en el párrafo anterior, se han analizado las prestaciones de los principales proyectos *Open Source* maduros y con soporte internacional que permitan publicar servicios OGC (WMS. WFS, WCS).

Los dos proyectos más relevantes son GeoServer y MapServer. La primera dificultad encontrada es usar un criterio común para comparar dos tecnologías tan diferentes: el primero es una aplicación web (*servlet*) desarrollado en Java y el segundo es un CGI desarrollado en C++. En el primer caso se despliega en un servidor de aplicaciones Java (p.e. Jetty, Tomcat, etc.) y el segundo sobre un servidor de páginas web (p.e. Apache HTTP server, IIS, etc.).

Para comparar las prestaciones y posibles limitaciones de las tecnologías en su estado actual se ha diseñado un experimento consistente en publicar un mismo conjunto de datos como un número creciente de capas en ambos casos de estudio. Así para desplegar 1.000, 10.000, 50.000 y 100.000 capas en GeoServer se ha utilizado una pequeña utilidad de desarrollo propio que explota la interfaz REST

de GeoServer. Para desplegar el mismo número de capas en MapServer se ha diseñado una utilidad que genera el archivo *MapFile* con la información necesaria.

Una vez publicados los datos en ambos casos, se ha medido el tiempo necesario para poner en marcha los servicios web con GeoServer y MapServer (*Servlet init*). De forma paralela, se ha medido el tiempo consumido por los servicios para responder a la consulta *GetCapabilities* El resultado de estas medidas se muestra en las Figura 1 para GeoServer y Figura 2 para MapServer.

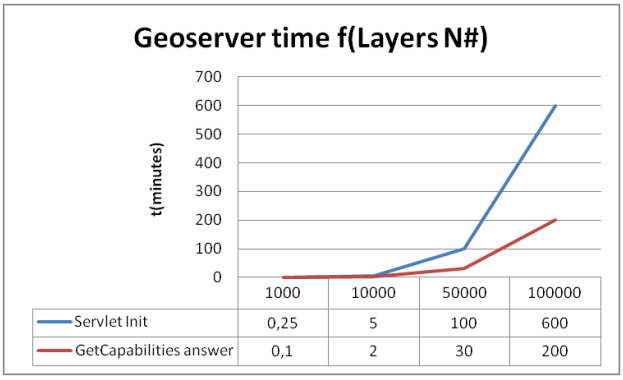

Figura 1: Tiempo requerido por GeoServer para iniciar la aplicación web y para las peticiones GetCapabilities dependiendo del nº de capas

Otro de los factores que limita la escalabilidad de los servicios es la cantidad de memoria RAM necesaria para poder operar. En este sentido se ha consultado la memoria que consumen GeoServer y MapServer en función del nº de capas publicadas. El resultado se presenta en la figura 3 se muestra la evolución de la memoria ocupada con GeoServer. En MapServer, la memoria necesaria es constante, y despreciable, ya que no ocupa de modo permanente la memoria RAM.

También se ha analizado la variedad de formatos de almacenamiento para la información geográfica soportados en ambos casos.

Así en GeoServer los *dataStore* para formatos vectoriales solo soportan: Shapefile, Property file, H2 database y spatialLite database. Los coverageStore, para formatos ráster, solo soporta: GeoTiff, georreference image (png, jpeg, tif) y mosaico de imágenes. GeoServer admite más formatos de almacenamiento para los datos y las coberturas a través de las librerías GDALe ImageIO-ext, por ejemplo: DTED, Hdr, ERDASImg, JP2MrSID, MrSID y NTIF. Por el contrario MapServer

soporta a través de las librerías GDAL/OGR (120 formatos ráster y 49 vectoriales –incluidas las bases de datos).

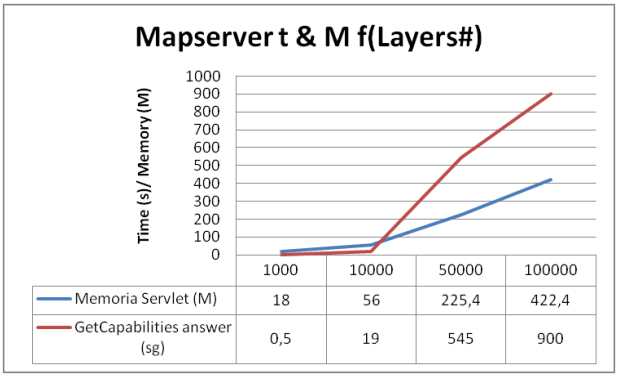

Figura 2: Memoria y tiempo requerido por MapServer para procesar una petición GetCapabilities en función del nº de capas

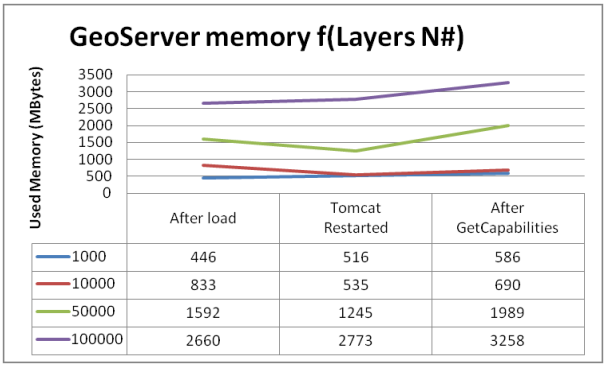

Figura 3: Memoria usada por GeoServer en función del nº de capas

3.2. Análisis y discusión de los resultados

A la vista de la Figura 1, GeoServer no sería apto en explotación cuando el nº de capas sea mayor de 1000 ya que el tiempo de inicio del servicio o de respuesta a la consulta *GetCapabilities* crece de forma lineal con el número de capas. Esta misma conclusión surge del análisis de la cantidad de memoria requerida por el sistema (Figura 3) cuando el nº de capas publicadas sea mayor de 10.000.
A la vista de la Figura 2, MapServer no sería apto en explotación cuando el nº de capas sea mayor de 10.000 al dispararse el tiempo de respuesta de las peticiones

individuales *GetCapabilities* o *GetMap* en términos lineales con gran pendiente. La memoria necesaria, también crece de forma lineal con el nº de capas, si bien la pendiente es mucho menor.

A la vista del número y la variedad de formatos soportados por MapServer y GeoServer, se puede decir que es más adecuado MapServer.

Desde el punto de vista de las necesidades de un Geo-Almacén que permita a los usuarios compartir y publicar su IG, no parece tener mucho sentido que el servicio esté replicado o balanceado entre varios servidores, porque sería necesaria una capa intermedia de software que analizara en qué servidor se encuentra una capa o conjunto de datos para su gestión y aparecerían problemas de sincronización.

Una solución intermedia, podría consistir en almacenar toda la información de configuración de los servidores (datos + *mapFile* o archivos XML de configuración de GeoServer) en un disco compartido y poner a trabajar varios servidores con la misma configuración e información. Esta solución, a priori factible, puede incurrir en algunos problemas de concurrencia al dar de alta o consultar los datos. Analizada esta situación o necesidad en los foros de GeoServer, se ha llegado a la conclusión de que es posible, está prevista una migración de esta configuración desde archivos XML a una base de datos (*GeoSolutions*: BBDD *Hibernate*), pero no está disponible actualmente.

4 Solución propuesta

La solución que se propone y se presenta en este documento, trata de alcanzar los siguientes objetivos: (1) homogeneizar tecnologías de desarrollo en un entorno de aplicaciones web con Java, (2) ampliar la variedad de formatos de datos publicables y (3) ampliar la riqueza y cantidad de metadatos que se generen de forma automática sobre los datos.

En cuanto a la base tecnológica que da soporte a la solución propuesta, se fundamenta en las librerías GDAL/OGR y MapScript (MapServer) desarrolladas en C++, explotadas mediante el SWIG para Java.

La idea fundamental de la propuesta desarrollada consiste en almacenar la información de las capas publicadas en una base de datos relacional (tablas), de modo que la aplicación web que se desarrolle (*Servlet* basado en *MapScript*)

genere al vuelo el archivo *mapFile* de configuración de MapServer con las capas demandadas. Si alguna de las capas o estilos demandados no existe se genera la correspondiente excepción, emulando el funcionamiento del servicio.

Para la generación automática de los metadatos, se propone el uso de las librerías GDAL/OGR, complementando su funcionalidad con otra información que pueda ser recuperada de los datos (archivos) mediante otras librerías.

Las tecnologías utilizadas en el desarrollo de un prototipo de aplicación web que implementa la solución propuesta son las siguientes: el *framework* Apache Click en su versión 2.3.0, el *framework* Apache Cayenne en su versión 3.0, el modulo Spring Security versión 2.0.4 parte del *framework* Spring en su versión 2.5.6, para algunas funcionalidades especiales del sistema se han manejado la API JavaMail en su versión 1.4.4 junto a la librería *commons-email* versión 1.2, y la librería Recaptcha Java en su versión 0.0.7 para interactuar con la API reCAPTCHA.

La aplicación está implementada usando el Modelo Vista Controlador (MVC) soportado mediante el *framework* Spring MVC. Spring utiliza un gestor de peticiones (DispatcherServlet) para dirigir las solicitudes a los objetos que controlan la lógica (Controller), para posteriormente pasar los resultados Model y View a la capa de la prestación. De esta manera se puede llevar a cabo la integración acoplando para cada elemento del patrón que cumplan respectivamente las funciones de Vista y Controlador dentro de la arquitectura propuesta. Esto se hace de dos formas: mediante las páginas implementadas usando el *framework* Click, y mediante los servicios, o componentes de datos, desarrollados con Cayenne,

Para la gestión de usuarios, credenciales, y el manejo de sesiones se adoptó el *framework* Spring Security, que permite personalizar y ajustar según las necesidades puntuales de la aplicación las funciones de autenticación y control de acceso, ya que proporciona las directivas necesarias para configurar las operaciones más comunes y personalizar en acceso a las secciones privadas y públicas de la aplicación.

La base de datos se compone de dos partes: (1) en una parte se maneja la información propia del prototipo, las tablas necesarias para la gestión de usuarios y el almacenamiento de los datos que los usuarios desean compartir, (2) de otra parte las tablas correspondientes al catalogo de metadatos utilizado (GeoNetwork). Ambas partes pueden estar soportadas por distintos gestores de base de datos y

ubicadas en diferentes servidores o pueden coexistir en un mismo gestor y servidor. Este hecho facilita la distribución de carga de trabajo en función de las necesidades y volumen de usuarios.

5 Estado actual de los desarrollos y Trabajos Futuros

En el prototipo de Geo-Almacén se han desarrollado los módulos que cubren las funcionalidades que se enumeran a continuación:

- Alta automática de usuarios, con envío de correo de verificación y mecanismo para evitar la generación automática de usuarios (reCaptcha).
- Agregar nuevos datos geográficos al almacén, ya sean ráster o vectoriales, formados por uno o varios archivos comprimidos en uno.
- Generación automática de metadatos para ambos formatos de datos: ráster y vector.
- Empaquetado de los metadatos en un archivo XML con el formato definido en la norma ISO19139.
- Alta de usuarios en GeoNetwork.
- Alta de metadatos en el catálogo CSW (GeoNetwork)
- Acceder al perfil de usuario para actualizar los datos de usuario y listar los datos publicados.
- El servlet que atiende las peticiones de los servicios WMS, WFS y WCS basado en MapScript y construyendo al vuelo el archivo *mapFile* de configuración.

Entre los trabajos por realizar en los próximos meses destacan las siguientes funcionalidades:

- Crear una interfaz de consulta al catálogo que liste y facilite el acceso a los datos publicados por los distintos usuarios.
- Integrar un cliente web (*WMS client*) que permita superponer las capas de datos localizadas mediante las búsquedas en el catálogo.
- Enriquecer los metadatos generados automáticamente con metadatos que se puedan derivar o calcular a partir de la información extraída de los datos.

Además de las funcionalidades enumeradas, se pretende someter el prototipo a diferentes pruebas de esfuerzo para analizar el comportamiento de la solución diseñada y desarrollada ante distintos niveles de carga (nº de usuarios, nº de capas publicadas, etc.).

Agradeciemientos

Este trabajo ha sido financiado parcialmente por el proyecto CENIT España Virtual, subvencionado por el CDTI dentro del programa Ingenio 2010 a través del Centro Nacional de Información Geográfica (CNIG).

Referencias

[1] Franklin, C., and Hane, P. “An introduction to GIS: linking maps to databases,” Database. Vol. 15, No. 2 April, 1992

[2] Turner, A. "Introduction to Neogeography", O'Reilly Short Cuts.

[3] Goodchild, M. "Citizens as sensors: the world of volunteered geography". *GeoJournal* 69 (4): 211–221

[4] Grus, L. Castelein, W. Crompvoets, J. Overduin, T. Loenen, B. Groenestijn, A. Rajabifard, A. Bregt, A. "An assessment view to evaluate whether Spatial Data Infrastructures meet their goals", Computers, Environment and Urban Systems 35 (3). - p. 217 - 229

[5] Georgiadou,Y. y Stoter, J. "Studying the use of geo-information in government – A conceptual framework". Computers, Environment and Urban Systems Nº 34, pp. 70-78

[6] Haklay, M. y Weber, P. "OpenStreetMap: User-Generated Street Maps", IEEE PERVASIVE COMPUTING, 7 (4) 12 - 18