WMS SOAP con ESB

Luciana Canales
lucanales@gmail.com

Gabriel Centurión
gccomputos@gmail.com

Maximiliano Felix
maxi.felix@gmail.com

RESUMEN

Las tecnologías GIS han adquirido mayor popularidad hoy en día y son usadas en gran cantidad de aplicaciones que permiten a los usuarios crear consultas interactivas, analizar la información espacial, editar datos, mapas y presentar los resultados de todas estas operaciones.

En este artículo se presenta un trabajo de investigación y desarrollo del estándar WMS implementado en SOAP.

Como resultado principal se obtiene el aprendizaje sobre tecnologías GIS, así como también, el conocimiento de las herramientas que permiten la integración de los sistemas de información geográfica.

Palabras Clave

ESB, GeoServer, OpenLayers, GetFeatureInfo, WMS, SOAP, REST.

1. INTRODUCCIÓN

Actualmente las tecnologías GIS permiten a los usuarios crear consultas interactivas, analizar la información espacial, editar datos y mapas. Es por este motivo que surge el interés en la estandarización del formato de los datos y de las normas de transferencia.

Como objetivo del proyecto se busca analizar el estándar de WMS en versión SOAP y alternativas de implementación utilizando funcionalidades del ESB.

La solución propuesta se compone de tres proyectos, client-wms, wms-rest y wms-soap. El primero hace las peticiones http solicitando los mapas que desea visualizar. El segundo recibe dichas peticiones y se comunica con el tercero para solicitar la información requerida. Tanto el segundo como el tercero interactúan con el servidor de mapas dependiendo de la información requerida.

El presente trabajo se realizó como proyecto de laboratorio de la edición 2011 de la asignatura Taller de Sistemas de Información Geográficos Empresariales.

El resto del documento se organiza por secciones de la siguiente manera: Marco Conceptual, Descripción del Problema, Solución Planteada, Arquitectura del Sistema, Implementación, Evaluación de la Solución, Desarrollo del Proyecto, Conclusiones y Trabajo a Futuro y Referencias.

2. MARCO CONCEPTUAL

En esta sección se describen varios conceptos que puedan ser necesarios para el entendimiento del artículo.

2.1 WMS

WMS (Web Map Service) define un protocolo para obtener mapas dinámicos a partir de información geográfica distribuida presentando la información como imágenes digitales susceptibles de ser visualizadas en pantalla.

2.2 ESB

ESB (Enterprise Service Bus) es una arquitectura de software que proporciona servicios fundamentales para la construcción de arquitecturas complejas a través de un sistema de mensajes (el bus) basado en las normas y que responde a eventos.

2.3 GeoServer

GeoServer es un servidor de código abierto escrito en Java que permite a los usuarios compartir y editar datos geoespaciales. Está diseñado para la interoperabilidad y publica información de cualquier gran fuente de datos del espacio usando estándares abiertos, por ejemplo WMS.

2.4 OpenLayers

OpenLayers es una librería Javascript de código abierto que permite acceder, manipular y visualizar mapas en páginas web. Su es diseño es orientado a objetos y proporciona un API que permite la creación de clientes web para acceder y manipular información geográfica proveniente de múltiples fuentes de datos, utilizando el estándar WMS.

2.5 PostGres - PostGis

PostGres es un motor de base de datos de código abierto, muy utilizado en todo el mundo. Este motor de base de datos, tiene una extensión para dar soporte a datos espaciales llama PostGis, que será en su conjunto la base de datos espacial a utilizar.

3. DESCRIPCIÓN DEL PROBLEMA

El problema planteado consiste en implementar la versión SOAP del estándar WMS analizando alternativas de implementación y utilizando funcionalidades del ESB.

La motivación para dicha propuesta, es analizar de qué manera se pueden integrar los estándares de sistemas de información geográfica, con los estándares y prácticas más comunes de los sistemas de información empresariales. Dado que en el mundo empresarial lo mas utilizado son herramientas como ESB y que la forma de exponer servicios de red es con el estándar SOAP. Analizar y probar como estas tecnologías son capaces de integrarse y utilizarse también en el ambiente de las tecnologías de información geográfica.

El siguiente esquema ilustra de manera representativa la solución que se desea alcanzar.

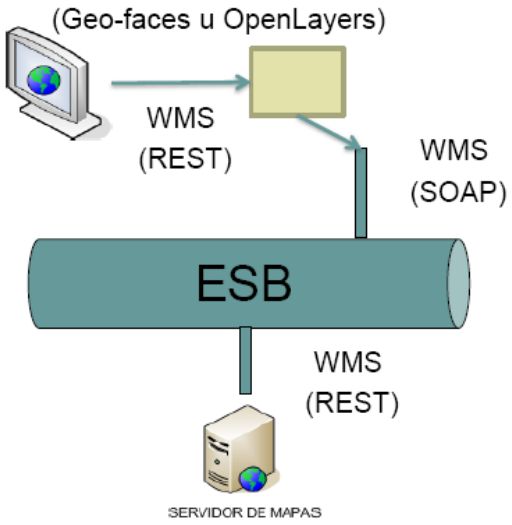


Figura 3.1 – Esquema del prototipo

Componentes del esquema:

- Cliente utiliza WMS para comunicarse con el servidor.
- Servidor realiza la transformación a SOAP e invoca WMS-SOAP.
- ESB expone WMS-SOAP y realiza la transformación SOAP-REST.
- ESB invoca WMS en el Servidor de Mapas.

Se desea exponer y consumir de manera SOAP y a través de un servidor ESB, la operación GetFeatureInfo, definida en el estándar WMS.

Dada la particularidad de que los clientes existentes en la actualidad que implementan el estándar WMS, solo soportan la versión REST del mismo, es necesario brindar ambas posibilidades, o sea, se requiere brindar la operación de forma SOAP, pero el cliente la utilizará en forma REST, por tanto se debe implementar también un transformador que pueda transferir pedidos REST en SOAP, y también las respuestas a los mismos en forma inversa.

4. SOLUCIÓN PLANTEADA

Se puede ver en la figura 4.1, como se desplegará la infraestructura para la prueba, se contará con un navegador que hará el papel del cliente, interactuando directamente con el servidor JBoss ESB, el cual brindará varios servicios, de los cuales el navegador sólo utilizará dos, el client-wms, que será nuestra interfaz de usuario web usando OpenLayers y el wms-rest, que utilizará OpenLayers para obtener los mapas y la información brindada por la operación GetFeatureInfo. Por otro lado está el servidor GeoServer, que será accedido solamente por el ESB, y la base de datos PostGis, que servirá los datos de mapas consumidos por el servidor de mapas.

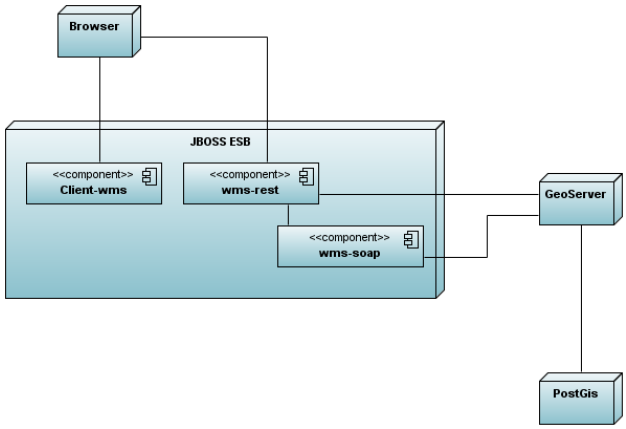


Figura 4.1 – Diagrama de despliegue

Se muestra en el diagrama la infraestructura de los distintos nodos que componen la prueba.

Se plantea dividir la solución en tres proyectos que serán servidos todos por el JBoss ESB, el primero llamado client-wms, es un simple war que sirve una página web que tiene el cliente OpenLayers que será usado desde un navegador. Esta página simplemente cargará las opciones e inicializará el OpenLayers, para que la herramienta pueda consumir los mapas, a través del ESB.

Por otro lado se implementará el proyecto wms-rest, el cual contará con tres servicios ESB, dos de los cuales son puntos de entrada al sistema.

Un servicio llamado "rest" brinda una comunicación directa con el GeoServer, transformando la petición http de entrada y sus parámetros, en una petición REST al servidor de mapas.

El servicio "router" también es accedido a través de peticiones http, este es el servicio que dependiendo del tipo de operación requerida enviará el mensaje hacia el servicio rest o al cliente SOAP utilizando el motor de reglas Drools. Define una regla que ejecuta para todas las peticiones, verifica el parámetro REQUEST, si el mismo tiene el valor "GetFeatureInfo", seguirá

el camino al cliente SOAP, de lo contrario seguirá el camino al servicio rest.

El último el servicio del proyecto wms-rest, es el cliente SOAP. Este servicio utiliza una de las acciones predefinidas del ESB, un cliente para consumir webservices SOAP, al cual fácilmente se le indica la ruta al wsdl a utilizar, donde buscar los parámetros, donde dejar la respuesta, con respecto al mensaje ESB, y que operación ejecutar, todo lo demás queda a cargo de la funcionalidad provista.

El tercer y último proyecto es la implementación SOAP de la operación GetFeatureInfo del estándar WMS. El nombre de este proyecto es wms-soap, el cual tiene un único servicio del ESB llamado WmsSoap, utilizando como herramienta para exponer el webservice jbossws, luego una de las acciones brindadas por el mismo JBoss ESB permite exponer como servicio de ESB el webservice mencionado.

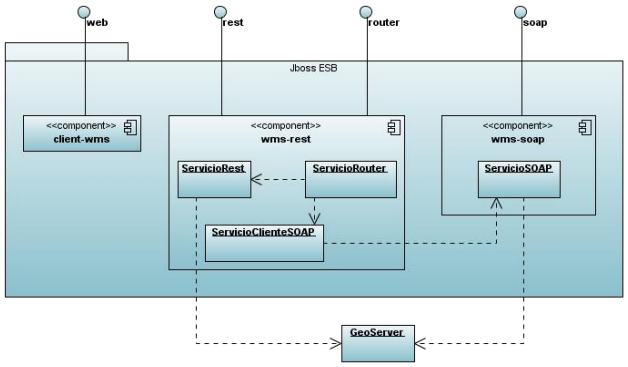


Figura 4.2 – Servicios dentro del ESB

El diagrama indica como se distribuyen los distintos servicios dentro del ESB, y su interacción.

5. IMPLEMENTACIÓN

Respecto a la implementación de los distintos proyectos se puede resaltar del proyecto wms-soap que se utilizó la acción provista por el ESB SOAPProcesor, para cual se tuvo que construir el archivo esb de forma diferente al proyecto wms-rest, más adelante en la sección de problemas encontrados se especifica con mas profundidad el asunto.

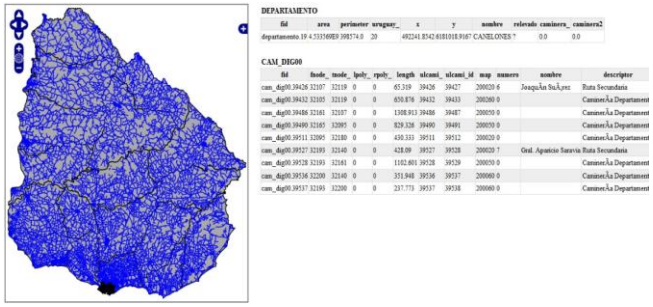
Sobre la forma en que se transforma la petición REST en SOAP para la operación GetFeatureInfo, cabe destacar que es un único método que recibe todos los parámetros definidos para la misma, aunque pueden omitirse los opcionales. Como el resultado de la operación es un xml también, este es colocado tal como viene en el xml de la respuesta SOAP. Luego el cliente SOAP lo extrae para ser devuelto al cliente en la respuesta http.

En lo que refiere al proyecto wms-rest se pueden destacar varios puntos, el primero sobre el que se hace mención es que el servicio llamado "rest" es completo, esto significa que a través del mismo funcionan todas las operaciones de WMS que son utilizadas por OpenLayers, se puede utilizando este servicio, dar soporte total a WMS. Por otro lado es interesante aclarar que para distinguir una operación de otra, se busca por el valor del atributo REQUEST, para tomar la decisión de seguir por el servicio rest o ir al servicio SOAP, esto se hace utilizando un regla de Drools, la cual tiene un primer paso que es obtener la petición http, guardarla en la memoria del motor de ejecución de reglas para que las siguiente reglas puedan acceder a la misma.

Sobre el cliente web que brinda la interfaz con OpenLayers, se puede resaltar que se le agregaron dos parámetros los cuales permiten cierta flexibilidad al momento de desplegar los datos y hacer las pruebas. El primer parámetro se llama layer, y corresponde al nombre de la capa o grupo de capas del servidor que se quiere visualizar en el mapa. El segundo parámetro es opcional y se llama "s" el mismo soporta dos valores: "rest" o "router", e indica al OpenLayers que servicio usar del servidor, "rest" para ir directamente al GeoServer a través del ESB utilizando llamadas REST. Y "router" utiliza en servicio Router del ESB el cual dependiendo de la operación pedida sigue un camino REST o SOAP.

Se debió conservar ambas posibilidades, la de utilizar REST o SOAP porque el cliente OpenLayers, solo utiliza REST, por tanto es dentro del ESB donde se toma la decisión de que servicio utilizar.

El producto final se puede observar desde un navegador y el resultado que se obtiene es el siguiente:



5.1 Productos y Herramientas

En esta sección se describen los productos y herramientas que se evaluaron y/o utilizaron, presentando brevemente una evaluación de los mismos.

Tabla 1. Evaluación de Productos y Herramientas

Producto	Puntos Fuertes	Puntos Débiles	Evaluación General
Gvsig 1.11	Permite visualizar de forma rápida, tanto directamente desde los shapefiles, como desde la base de datos	Más limitaciones que la base respecto de los nombres de tablas y columnas. Si hay un error al importar una tabla de	Parece una herramienta interesante especialmente si se quiere visualizar información en forma rápida. No se utilizó en

	geográfica.	la base, hay que reiniciar el programa, no se recupera, ya que no muestra las tablas de la misma.	gran medida por lo cual no es posible realizar una evaluación más crítica.
PostGres 9.0.4 Postgis 1.52	Buen manejador de base de datos en general. postgis, tiene fácil instalación. Importación de mapas desde shapefiles es muy buena. Buena y completa interfaz de usuario.	No se pudo utilizar la última versión liberada porque faltaba soporte para sistemas operativos de 64 bits.	Muy buena.
JBoss ESB Server 4.9	Buena cantidad de ejemplos muy completos y variados. Flexibilidad. Sencillez de uso una vez aprendidos los conceptos que maneja.	No provee un cliente para peticiones http. Curva de aprendizaje lenta. Documentación no es muy profunda, solo sirve para una leve introducción.	
OpenLayers 2.10	Fácil utilización. Buena documentación. Flexibilidad tiene muchas opciones personalizables. Forma de uso orientada a objetos. Soporte integrado para GetFeatureInfo.	Solo tiene soporte para servicios WMS Rest.	Buena herramienta para consumir mapas y mostrar mas información utilizando como cliente un navegador.
WMS 1.3	Fácil uso. Completo en los problemas que resuelve.	Solo sirve para consultar información, no define métodos para editar la información.	
GeoServer 2.1.0	Muy buena interfaz de usuario, intuitivo. Rápida curva de aprendizaje. Excelente integración con postgis. Multiplatafo rma, no requiere instalación. Soporte total para WMS.	Pequeño problema con el repositorio físico de espacios de trabajo. A veces según la sesión genera urls, que en sesiones subsiguientes son inválidas.	Buena herramienta para servir mapas, a priori nos parece mejor que MapServer.
JBoss Develper Studio 4.0.0	Totalmente integrado con eclipse, ahorra tener que instalar el plugin jboss Tools entre otros. Todas las bondades de Eclipse como IDE.	Tiene un problema en la definición de servidores debido que no permite para ESB 4.9 definir un jboss 4.2.x, siendo que estos dos productos se distribuyen juntos.	Muy buena en general.
Drools 5.1.0	Flexibilidad.	No se encontró rápidamente buena documentación introductoria.	No utilizamos mucho esta herramienta como para llegar a una conclusión determinante pero parece estar muy interesante, debido a su integración con ESB.

5.2 Problemas Encontrados

- Cuando se utilizó la aplicación GvSig, se tuvieron problemas para importar los shapefiles provistos tanto en el mismo programa como en la base de datos, en algunos caso los problemas pudieron resolverse modificando los

nombres de algunas tablas y columnas, pero hubo un caso que no pudo ser utilizado.

- Tratando de acceder al GeoServer a través del ESB utilizando un ruteador provisto llamado http-ROUTER. El mismo si bien hace la invocación http a cualquier url no pasa los parámetros de la petición original, fue entonces que con una investigación profunda llegamos a la conclusión de que ese router solo sirve para invocaciones internas al ESB, y asume que los parámetros van a ser pasados en el mensaje. Lo cual confunde porque permite especificarle cualquier url, pero solo sirve para invocaciones dentro del ESB. Este problema fue resuelto implementando un cliente http propio, que permite hacer peticiones GET o POST, con o sin parámetros, ya que el ESB no provee soporte para tal funcionalidad.
- En el proyecto wms-soap, el cual brinda el webservices SOAP que soporta la operación WMS GetFeatureInfo, para que funcione correctamente, como un webservice se deben tener ciertos recaudos. Debido a limitaciones de la plataforma JBoss ESB Server, no pasa así en el JBoss Aplication Server con ESB como servicio, todo webservice para que sea tratado como tal, por el deployer jbossws debe estar incluido dentro del archivo generado ".esb", para que esto funcione correctamente debe primero generarse un archivo ".war" el cual defina en conjunto con las anotaciones de webservice, un servicio web (web.xml). Este war debe ser incluido dentro el ".esb" del proyecto en la raíz del mismo. Esta es la única forma en la cual se logró que funcionara. Para esto hubo que construir un archivo Build.xml de Ant, porque el IDE no tiene soporte para esta acción. Por lo expresado anteriormente para este webservice no se pudo usar EJB 3, debió utilizarse un POJO con anotaciones. Existen otros métodos que provee el ESB para exponer webservices pero no se pudo hacerlos funcionar de manera sencilla así que fueron descartados para esta prueba.
- Un extraño comportamiento en el IDE no permite sincronizar un proyecto ESB 4.9 dentro de un JBoss Server 4.2.x por razones de compatibilidad entre ambos productos, según el mensaje que se muestra, lo curioso es que el JBoss ESB Server es un Aplication Server 4.2.3 que viene con la versión 4.9 de la plataforma ESB. La única solución encontrada fue mentirle al IDE y definir el servidor como un JBoss 5.1, aunque esta versión de servidor no sea totalmente compatible con ESB según el fabricante.

6. EVALUACIÓN DE LA SOLUCIÓN

Se considera que la solución alcanzada cumple con las expectativas deseadas.

Como punto fuerte se puede afirmar que desde el punto de vista de los requerimientos, la aplicación cumple con lo exigido lo cual es planteado en el punto 3, "Descripción del Problema".

Cabe destacar que la solución planteada logra crear componentes distribuidos los cuales podrían ser reutilizados en otros proyectos.

Estos componentes tienen como inconvenientes que algunas urls están hard code, por ende para la reutilización de los mismos se debería pasar estas rutas a un archivo de configuración o ser pasadas como parámetros. Pero salvando lo mencionado anteriormente es un punto fuerte que tan solo cambiando configuración de puede extender para utilizar cualquier servidor de mapas que soporte el estándar WMS.

Así mismo es posible de forma rápida extender el proyecto para soportar todas las operaciones WMS usando SOAP, ya que se debe implementar un webservice que enmascare la petición y la reenvíe al estilo REST, y extender la regla Drools para que sepa que encaminar hacia SOAP y que seguir por la vía REST.

El uso del ESB, sin embargo, tiene sus ventajas y desventajas, vemos como ventajas, todas las funcionalidades provistas que son totalmente reutilizables, por ejemplo, exponer y consumir webservices, el ruteo como forma de tomar decisiones. Sin el ESB estas funcionalidades hay que implementarlas una a una, y quedaría todo mucha mas ligado a la solución específica del problema. Por otro lado la lenta curva de aprendizaje que tiene el ESB, retrasa bastante el poder entender como conjugar de forma correcta todas las funcionalidades provistas. En conclusión, para programadores experimentados, la gran ventaja del ESB, es la reusabilidad, pero el tiempo invertido será similar a implementar todo fuera del ESB, como en un JBoss Aplication Server, o directamente en un Tomcat.

7. DESARROLLO DEL PROYECTO

Comenzando el desarrollo se dedicó dos semanas a definir herramientas, testeando las mismas mediante pequeños prototipos y pruebas para además de determinar cuales utilizar, creando así una base de conocimiento.

Ciertas herramientas no fueron opcionales, como ser Postgres con Postgis, pero incluso en ese caso hubo que probar la instalación de Postgres tanto en la última versión como en la penúltima, puesto que el equipo contaba con sistemas de 32 y 64 bits, no existiendo soporte para 64 bits en ciertas versiones beta. Además para el caso de PostGis había más de una forma de instalarlo e incluso cierta incompatibilidad entre versiones y plataformas. En definitiva para este punto se utilizo Postgressql 9.0 y a través de la aplicación StackBuilder 3.0.0 se agregó la extensión de PostGis.

En cuanto al bus empresarial Jboss ESB fue otra tecnología que no era optativa. En este caso más allá de las versiones las opciones eran instalar JbossESB dentro de un JbossAS o utilizar una versión de Jboss ESB Server, con todo lo relacionado al bus empresarial incorporado. Se optó por la segunda opción puesto que entre otras razones no tenía el costo de cargas en aplicaciones innecesarias, como ser un manejador de ejb etc.

Como IDE de desarrollo se optó por Jboss Developer Studio, una herramienta basada en Eclipse IDE que se integra aceptablemente a Jboss ESB.

En cuanto al servidor de mapas en un principio se planteó utilizar MapServer como servidor de mapas. Pero en las primeras pruebas se notó que la curva de aprendizaje para realizar una simple exposición no era despreciable, por lo que se optó por realizar una prueba en paralelo con GeoServer.

Rápidamente GeoServer demostró ser más intuitivo y mejor soportado a nivel de documentación, por lo que se definió utilizar este servidor de mapas.

Para la realización de cliente web se utilizó la librería javascript OpenLayers, los ejemplos prestados por GeoServer fueron suficiente para la realización del Cliente.

Luego de las primeras dos semanas que llevó lo anterior se comenzó con la realización del proyecto propiamente dicho. Por un lado se reconfiguró la BD con el sistema de coordenadas para la región planteada para evitar cualquier inconveniente. Además se comenzó con el proyecto que permitiría a través del ESB el acceso a GeoServer consumiendo REST. En paralelo se desarrolló el cliente que ejecutará sobre ese proyecto. Esto llevo dos semanas más. Sobre las siguientes dos semanas se desarrolló el proyecto que expone el WS SOAP, que a su vez transforma la petición en REST para luego consumir el servicio anteriormente desarrollado. Ya en el final utilizamos las dos ultimas semanas para la realización de la documentación, presentación, armado de la maquina virtual y otros entregables. Completando de esta manera un total de ocho semanas.

8. CONCLUSIONES Y TRABAJO A FUTURO

Se concluye que es posible transformar los servicios REST de WMS a servicios WS SOAP. Que para dicha transformación Jbossesb nos brinda una plataforma orientada a servicios la cual nos permitió desplegar todos los servicios necesarios para realizar la transformación como se planteó en los requerimientos.

Como trabajo a futuro queda pendiente la realización de la transformación de las operaciones GetCapabilities y GetMap, en las cuales se distingue que en la primera es un simple pasaje de un xml, pero en la segundo lo retornado es una imagen y hay que tener en cuenta eso, aunque sabiendo que el cuerpo del mensaje es un arreglo de bytes no parece inconveniente a priori tratar imágenes dentro del ESB.

Además podría ser interesante utilizar alguno de los motores de transformación de xml provistos por el ESB para transformar las peticiones SOAP en REST y viceversa. También quedan por probar los distintos métodos que provee el ESB para exponer webservices y consumirlos.

9. REFERENCIAS

[1] Laboratorio de Integración de Sistemas. Taller de Sistemas de Información Geográficos Empresariales – Trabajos Obligatorios. Año 2011. http://www.fing.edu.uy/inco/cursos/tsi/TSIG/clases2011/ProyectosTSIG2011.pdf [Consulta: Mayo 2011]

[2] http://www.fing.edu.uy/inco/cursos/tsi/TSIG/clases2011/WebServicesGeograficos2011.pdf

[3] http://en.wikipedia.org/wiki/Enterprise_service_bus

[4] http://www.jboss.org/jbossesb

[5] http://openlayers.ingemoral.es/manualOpenLayers.html

[6] http://www.fing.edu.uy/inco/cursos/tsi/TSIG/clases2011/Visualizadores.pdf

[7] http://es.wikipedia.org/wiki/GeoServer

[8] http://docs.jboss.org/jbossesb/docs/4.9/manuals/html/Programmers_Guide/index.html