

INSTITUTO DE COMPUTACIÓN
FACULTAD DE INGENIERÍA
UNIVERSIDAD DE LA REPÚBLICA

Plataforma de Integración de Información Geográfica sobre JBossESB

Implementación de una ESB-Based Enterprise
Integration Platform (GeoEEIP)

Autores

Ignacio Assandri Punschke
Martín Villero Sosa
Nicolás Lasarte Bonifacio

Tutores

Bruno Rienzi
Raquel Sosa

Usuario

Laura González

Resumen

En la actualidad se ha incrementado notoriamente el uso de información geográfica para diversos fines y por diferentes tipos de usuarios. Esta información es procesada y analizada a través de los Sistemas de Información Geográfica (GIS). Los GIS se encargan de manipular dicha información permitiendo el almacenamiento, análisis y presentación de datos geográficos, a través de protocolos estándares o propietarios. Las organizaciones han ido incorporando los GIS como parte de sus sistemas de información, dando lugar al concepto de “Enterprise GIS”. Si bien existe una tendencia a incorporar estándares en los GIS para el almacenamiento, procesamiento e intercambio de datos no existe un mecanismo estándar que permita integrar Web Services Geográficos con datos de negocio y funcionalidades de sistemas empresariales.

Con el fin de abordar esta problemática, el artículo “Towards an ESB-Based Enterprise Integration Platform for Geospatial Web Services” propone una plataforma de integración que, basándose en las capacidades de mediación nativas de un Enterprise Service Bus (ESB), provee mecanismos de integración orientados a servicios geográficos. Un ESB es una plataforma de integración basada en estándares, la cual proporciona capacidades de mediación, para abordar heterogeneidades entre las aplicaciones con respecto a los protocolos de comunicación y formatos de mensaje, entre otros. Sin embargo, la plataforma propuesta está planteada a nivel conceptual y no cuenta actualmente con una implementación que permita, por ejemplo, evaluar su factibilidad técnica.

Este trabajo propone una solución que implementa dicha plataforma utilizando el producto JBossESB y brindando a su vez una herramienta configurable y extensible que permite minimizar los tiempos de desarrollo asociados a la integración. Esta implementación brinda una solución a dos de los desafíos más usuales en la integración de sistemas con estas características: en primer lugar, el enriquecimiento de los datos geográficos con datos empresariales disponibles a través de Web Services y en segundo lugar, el acceso a Web Services geográficos mediante estándares tradicionales de Web Services (específicamente, acceso mediante el protocolo SOAP a Web Services que utilizan el estilo REST). A su vez, la plataforma brinda una base extensible para incorporar nuevas funcionalidades que permitan solucionar otros problemas de integración y mediación que se presenten dentro del mismo contexto.

Índice

Resumen	1
1. Introducción	4
1.1. Contexto y Motivación	4
1.2. Objetivos y resultados esperados.....	5
1.3. Aportes del proyecto	6
1.4. Organización del documento	6
2. Estado del arte.....	8
2.1. Tecnologías de Información Geográfica.....	8
2.1.1. Sistemas de información geográfica	8
2.1.2. Web Services Geográficos	8
2.1.3. GeoServer	13
2.2. Sistemas empresariales	13
2.2.1. SOA	14
2.2.2. Middleware	14
2.2.3. JMS.....	14
2.2.4. Web Services	15
2.2.5. ESB	18
2.3. Soluciones analizadas	22
2.3.1. Integración de servicios Geográficos en Plataformas de Gobierno Electrónico	22
2.3.2. Integración de servicios geográficos en sistemas empresariales.....	23
2.3.3. WFS en ESB con fuente externa y caché	24
2.3.4. WMS SOAP con ESB.....	25
2.3.5. Integración de sistemas geográficos y sistemas desarrollados sobre SAP ERP.....	25
2.4. Resumen	27
3. Análisis.....	30
3.1. Análisis del documento de definición de GeoEEIP	30
3.2. Componentes identificados.....	31
3.2.1. Componentes Básicos de integración	31
3.2.2. Componentes Complejos de integración	35
3.3. JBossESB	37
3.3.1. Mensajes y Servicios.....	37
3.3.2. Componentes disponibles	39
3.4. Alternativas de arquitecturas evaluadas.....	42

3.4.1. GeoEEIP Abstracto	43
3.4.2. GeoEEIP Mixto	45
3.4.3. Selección de la arquitectura	47
3.5. Aplicación de Configuración	48
4. Solución implementada.....	50
4.1. Arquitectura de alto nivel.....	50
4.2. Componentes básicos.....	52
4.3. Componentes complejos	58
4.4. Aplicación de configuración	62
4.5. Herramientas utilizadas.....	65
5. Caso de estudio	66
5.1. Análisis.....	66
5.1.1. Objetivo	66
5.1.2. Aspectos técnicos	66
5.2. Escenario de prueba	67
5.3. Componentes Configurados	70
5.4. Comparativa con documento de referencia.....	71
5.5. Resultados obtenidos	72
5.6. Funcionalidades no comprendidas en el caso de estudio	72
6. Conclusiones y trabajo futuro	76
6.1. Gestión del Proyecto	76
6.1.1. Etapas del proyecto	76
6.1.2. Análisis de estimación y desvío	77
6.2. Conclusiones.....	78
6.3. Trabajo a futuro.....	79
Referencias	82
Apéndice 1. Otros Componentes identificados.....	84
Apéndice 2. Otros Componentes implementados	86
Apéndice 3. Configuración para mapeo de datos	90

1. Introducción

Este informe describe el proyecto denominado “Plataforma de Integración de Información Geográfica sobre JBossESB”, el cual implementa la plataforma propuesta de integración de servicios geográficos y empresariales presentada en el artículo “Towards an ESB-Based Enterprise Integration Platform for Geospatial Web Services”[1].

Este capítulo contiene el contexto y la motivación del proyecto, los objetivos planteados, los aportes del proyecto y por último la organización del resto del documento.

1.1. Contexto y Motivación

Las organizaciones para exponer servicios que presentan información geográfica, utilizan en el escenario más sencillo, una única fuente de datos, que por lo general es una base de datos geoespacial. Esta información es publicada a través de un Internet Map Server (IMS), también conocido como *servidor de mapas*, el cual se encarga de consultar la base de datos geoespacial. Para presentar a los usuarios la información geográfica se utilizan visores de mapas, estos se comunican con el IMS a través de Web Services geográficos para obtener la información y luego se encargan de desplegarla en pantalla. Sobre este escenario se desea agregar información empresarial (posiblemente de varias fuentes) a la ya provista por la base de datos geoespacial. [1]

Un ejemplo de dicho escenario se plantea a continuación. Se consideran dos empresas, las cuales están interesadas en tener un mapa que contenga todas las direcciones en un país y así referenciar las direcciones con la información de sus clientes, para poder realizar un análisis de la información en el mapa. La información de los clientes puede ser obtenida accediendo a uno o varios Web Services (WS). Como se muestra en la Figura 1 la Empresa A accede al WS - 1 y la Empresa B accede al WS - 2 y WS - 3 para obtener información de los clientes. Por otro lado la información geográfica es obtenida por parte de las empresas A y B a través de consultas al IMS - 1 e IMS - 2 respectivamente. De esta forma para asociar la información de los clientes (información empresarial) con sus direcciones (información geográfica) se incluye un componente de integración, el cual se encarga de generar la respuesta enriquecida para los visores de mapas. Estos componentes se muestran en la Figura 1 como Integrador A, e Integrador B.

El ejemplo planteado implementa soluciones particulares para cada empresa, con el fin de resolver la integración de datos y servicios. La ausencia de mecanismos estandarizados complejiza el desarrollo de dichas soluciones, generando altos costos de desarrollo y careciendo de reusabilidad. Para abordar estas problemáticas en los últimos años se han propuesto varias soluciones teóricas. En [1] se propone una solución basada en mecanismos de integración orientados a servicios geográficos sobre un ESB. Los mecanismos de integración definidos en [1] encapsulan de forma genérica las problemáticas comunes de integración de servicios empresariales en un contexto georreferenciado, utilizando patrones de mediación soportados en un ESB.

Un ESB (Enterprise Service Bus) es una plataforma de integración, utilizada como base para resolver las señaladas limitaciones de los sistemas actuales. Como primer acercamiento, un ESB es una plataforma de integración basada en estándares que combina mensajería, Web Services, transformación de datos y ruteo inteligente para conectar y coordinar de forma fiable la integración de diversas aplicaciones. Un ESB provee una capa intermedia, con lógica reusable de integración y comunicación, que ayuda a tratar las diferencias

entre aplicaciones relacionadas con: protocolos de comunicación, formato de mensajes y calidad de servicio (QoS) entre otros.[2]

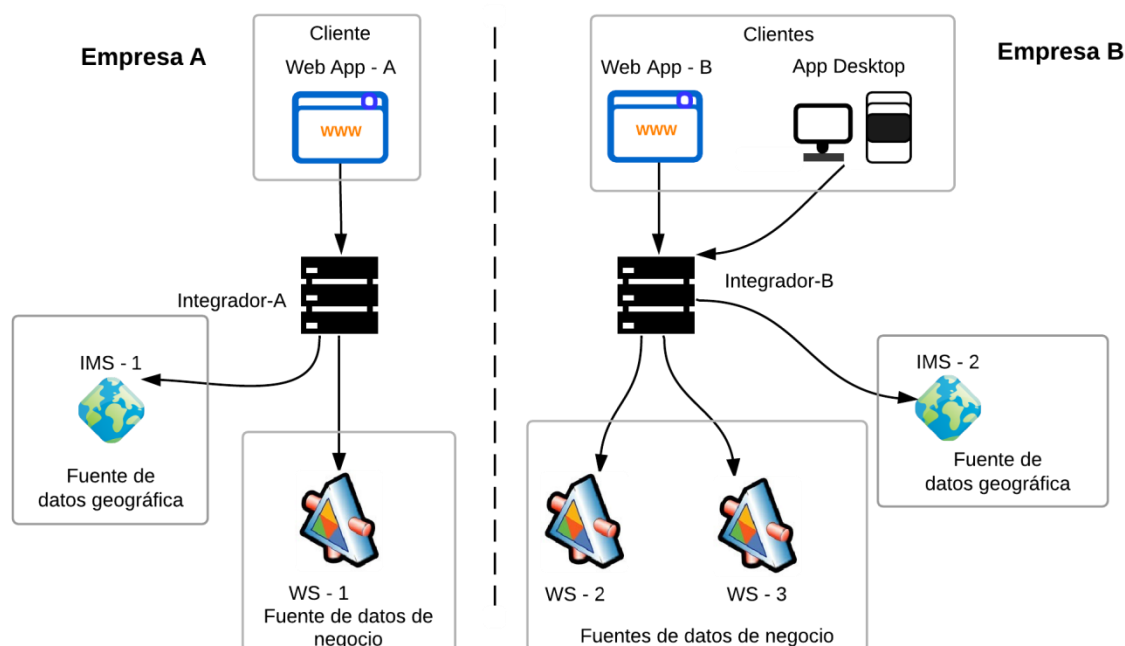


Figura 1 – Integraciones no reutilizables

De acuerdo con la solución planteada en [1] y las posibilidades que brinda un ESB, resulta interesante desarrollar una plataforma de integración basada en este último, que extienda sus capacidades básicas implementando los mecanismos de integración orientados a servicios geográficos definidos en [1] y dejando planteada una base para posibles extensiones de la misma.

1.2. Objetivos y resultados esperados

Partiendo del problema antes descrito, el objetivo general del proyecto es: realizar una implementación base de la plataforma presentada en el artículo “Towards an ESB-Based Enterprise Integration Platform for Geospatial Web Services”[1] sobre un producto de ESB específico.

Considerando esto se plantean los siguientes objetivos específicos:

- Investigación de las experiencias, propuestas y tecnologías relacionadas con la integración de sistemas de información geográficos con sistemas empresariales.
- Análisis, diseño e implementación de la plataforma de integración GeoEEIP sobre JBossESB.
- Implementación de una aplicación Web, para la configuración y ensamblado de los mecanismos implementados para la plataforma GeoEEIP.
- Desarrollo de un caso de estudio, que demuestre el uso de la plataforma GeoEEIP en una integración entre datos de un sistema de información geográfico y datos empresariales.

1.3. Aportes del proyecto

Los principales aportes son:

- Documento destinado a usuarios de JBossESB, con casos de prueba de patrones de mediación simples y extensión de acciones sobre JBossESB.
- Análisis, diseño e implementación de la plataforma de integración GeoEEIP sobre JBossESB, la cual permite solucionar problemas habituales de integración de servicios geográficos con servicios empresariales e incorporar a futuro nuevos mecanismos de integración y mediación.
- Implementación de una aplicación de configuración que permita generar componentes de integración de la plataforma GeoEEIP, con el fin de facilitar el uso de la misma. La aplicación está dirigida a usuarios que no cuenten con conocimientos avanzados en JBossESB.
- Desarrollo de un caso de estudio para evaluar los procesos de integración, definidos en el trabajo base [1].

1.4. Organización del documento

El resto de documento se organiza de la siguiente manera. En el Capítulo 2 se presenta una revisión del conocimiento existente y soluciones a problemáticas similares. En el Capítulo 3 se realiza un análisis de la problemática planteada, componentes detectados y una evaluación de las arquitecturas propuestas. En el Capítulo 4 se describe la solución implementada, especificando la integración de los componentes de la plataforma GeoEEIP y los componentes de JBossESB. Así mismo se describe la aplicación de configuración de los componentes de la plataforma. En el Capítulo 5 se muestra el uso de la solución en un caso de estudio. Por último en Capítulo 6 se presentan las conclusiones de la tesis y posibles trabajos a futuro.

2. Estado del arte

Este capítulo trata las tecnologías y conceptos básicos para entender la problemática del proyecto. Para esto se divide en cuatro grandes puntos. La sección 2.1 trata sobre las Tecnologías de la Información Geográfica, dentro de la cual se ven los conceptos fundamentales, como Sistema de Información Geográfica y Web Services Geográficos (en particular WMS y WFS). En la sección 2.2 se presentan los sistemas empresariales, los cuales en el marco del proyecto actúan como fuentes de enriquecimiento. Para poder entender cómo funcionan y cómo se adaptan a las necesidades del mercado empresarial actual, se introducen los conceptos de: Service Oriented Architecture (SOA), Middleware, Java Message Service (JMS), Web Services SOAP, Web Services REST, comparativas de SOAP y REST y por último Enterprise Service Bus (concepto base para el desarrollo del proyecto). En la sección 2.3 se presentan los principales aportes de la investigación y el análisis de soluciones existentes en el mercado, referentes a la integración de los dos puntos anteriores. Por último en la sección 2.4 se plantea un resumen de los conceptos, tecnologías e investigaciones vistas en el capítulo.

2.1. Tecnologías de Información Geográfica.

Las Tecnologías de Información Geográfica (TIG) son herramientas básicas en la investigación, la planificación y la gestión de la información geográfica. En las siguientes secciones se presentan conceptos básicos en los cuales se basan estas tecnologías.

2.1.1. Sistemas de información geográfica

Según [1], “tradicionalmente, un sistema de información geográfica (GIS), ha sido definido como un sistema que integra hardware, software y datos para capturar, administrar, analizar y visualizar todo tipo de información geo-referenciada”. Un ejemplo de este tipo de definición es la brindada por ESRI[3]. Los GIS han tenido su auge entre una amplia gama de usuarios en los últimos años. Para esto basta con observar cómo funcionaban los GIS hace un par de décadas, siendo estos orientados a expertos y poco integrados en las organizaciones. Contrariamente, en la actualidad cada vez más tecnologías utilizadas en las empresas, como son los servidores de aplicaciones y las bases de datos relacionales, han incorporado componentes GIS para proveer o interactuar con información geográfica [1].

2.1.2. Web Services Geográficos

Los Web Services geográficos que se presentan, son estándares definidos por Open Geospatial Consortium¹ (OGC), los cuales son llamados OWS. Estos son definidos en base a estándares de Internet siendo algunos ejemplos: HTTP, XML, MIME.

OGC es un consorcio, el cual tiene como objetivo definir estándares abiertos para los sistemas de información geográfica. Estos estándares permiten a los desarrolladores brindar Información Geográfica para ser utilizada aplicaciones de todo tipo. [4]

2.1.2.1. WMS

WMS (Web Map Service) provee una interfaz HTTP para realizar peticiones de imágenes de mapas geo-referenciados, de una o más bases de datos geoespaciales[5]. Los parámetros de la petición incluyen las capas a ser consultadas, el área de interés a ser obtenida y la transparencia (con el fin de poder combinar dichas capas). La respuesta es obtenida en un formato de imagen estándar (ejs.: JPG, PNG, GIF), o formato

¹ <http://www.opengeospatial.org/>

de gráficos vectoriales (ejs: SVG, WebCGM). El hecho de usar formatos como GIF o PNG en la respuesta, permite que por medio de la transparencia se combinen las diferentes capas obtenidas, resultando particularmente interesante cuando se obtienen capas de diferentes servidores. Cabe destacar que el estándar define dos clases, una para WMS básico y otro para Queryable WMS (WMS para consultas). WMS básico debe cumplir con los servicios básicos de las operaciones GetCapabilities y GetMap. Por otro lado WMS para consultas debe soportar las mismas funcionalidades que el básico, más la operación GetFeatureInfo[6]. Dichas operaciones se presentan a continuación.

Operaciones de WMS

GetCapabilities

Es mandatoria, y obtiene la metadata con la información del servicio que se provee y de los parámetros válidos y sus valores. Los parámetros que se destacan de la petición son:

- SERVICE, que indica el protocolo al cual hace referencia (WFS/WMS).
- VERSION, para indicar la versión del protocolo usado.
- REQUEST, indica la operación.

A continuación se plantea un ejemplo de petición, en la cual se especifica el servicio (WMS), la versión del servicio (1.1.1) y la operación (GetCapabilities). La misma retorna la metadata con la información del servicio.

<http://geoweb.montevideo.gub.uy/geoserver/ide/wms?SERVICE=WMS&VERSION=1.1.1&REQUEST=GetCapabilities>

GetMap

Esta operación es mandatoria y devuelve un mapa ante su invocación. El mapa es un binario, cuyo valor viene por defecto en Base64 como respuesta a la petición. De los parámetros se destacan:

- SERVICE, que indica el protocolo al cual hace referencia (WMS).
- VERSION, para indicar la versión del protocolo usado.
- REQUEST, indica la operación.
- LAYERS, indica las capas a retornar separadas por coma.
- SRS, especifica el sistema de coordenadas.
- BBOX, especifica el cuadro delimitador (*bounding box*) representado por cuatro valores separados por coma, el mínimo de X, mínimo de Y, máximo de X y máximo de Y, para una región en el SRS especificado.
- FORMAT, define el formato de la respuesta.

A continuación se plantea un ejemplo de petición, el cual retorna un mapa en forma de imagen. En este caso la imagen de la capa solicitada (*ide:ide_depto_p*) corresponde al departamento de Montevideo. En dicha petición se puede apreciar el servidor de mapas al cual se le solicita la operación (*geoweb.montevideo.gub.uy/geoserver*), el servicio (WMS), la versión del servicio (1.1.1), la operación (GetMap), la capa (*ide:ide_depto_p*), el *bounding box* (551994,6133494,589199,6159798), las dimensiones de la imagen en pixels (512x361), el sistema de coordenadas para Uruguay (EPSG:32721) y por último el formato de la respuesta (*image/png*).

La Figura 2 muestra el resultado en el formato PNG especificado:

http://geoweb.montevideo.gub.uy/geoserver/ide/wms?service=WMS&version=1.1.1&request=GetMap&layers=ide:ide_depto_p&bbox=551994,6133494,589199,6159798&width=512&height=361&srs=EPSG:32721&format=image/png



Figura 2 - Capa del departamento de Montevideo

GetFeatureInfo

Esta operación es opcional y es utilizada para consultar datos de una o varias capas, utilizando como parámetros de filtro: QUERY_LAYERS, X, Y. De los parámetros se destacan:

- SERVICE, que indica el protocolo al cual hace referencia (WMS).
- VERSION, para indicar la versión del protocolo usado.
- REQUEST, indica la operación.
- QUERY_LAYERS, indica las capas consultadas separadas por coma.
- LAYERS, indica separadas por coma, las capas a ser desplegadas.
- SRS, especifica el sistema de coordenadas.
- BBOX, especifica el *bounding box* representado por cuatro valores separados por coma, el mínimo de X, mínimo de Y, máximo de X y máximo de Y, para una región en el SRS especificado.
- INFO_FORMAT, define el formato de la respuesta.
- X, es la coordenada X de la capa, en pixeles (medida desde el vértice superior izquierdo).
- Y, es la coordenada Y de la capa, en pixeles (medida desde el vértice superior izquierdo).

A continuación se plantea un ejemplo de petición, en la cual se solicita información sobre un punto del mapa, determinado por los parámetros x e y. Se especifica la operación (*GetFeatureInfo*), el *bounding box* (BBOX), el servicio (WMS), el formato de respuesta (*application/vnd.ogc.gml*), la capa a ser consultada (*ide:ide_depto_p*), la capa a ser desplegada (*ide:ide_depto_p*), las dimensiones del mapa en pixeles (512x361), el sistema de coordenadas para Uruguay (EPSG:32721) y la versión del servicio (1.1.1). La respuesta brinda información adicional sobre el punto seleccionado y es devuelta en el formato especificado por el parámetro INFO_FORMAT.

http://geoweb.montevideo.gub.uy/geoserver/ide/wms?request=GetFeatureInfo&bbox=533391,6120414,607801,6172879&service=wms&info_format=application/vnd.ogc.gml&query_layers=ide:ide_depto_p&Layers=ide:ide_depto_p&width=512&height=361&srs=EPSG:32721&version=1.1.1&x=283&y=187

Es importante tener en cuenta que este protocolo no posee la capacidad de identificar ni seleccionar entidades geográficas individuales a partir del mapa, por lo que en lugar de utilizar un identificador como parámetro del pedido, utiliza un par de coordenadas (X e Y), la serie de capas sobre las cuales se realiza la consulta, el área visible del mapa (BBOX) y el SRS, entre otros. En la Figura 3 se muestra un ejemplo de consulta sobre las capas de RIOS, RUTAS y PUENTES. Al hacer clic en un punto específico del mapa, se pueden apreciar los datos obtenidos de estas tres capas.

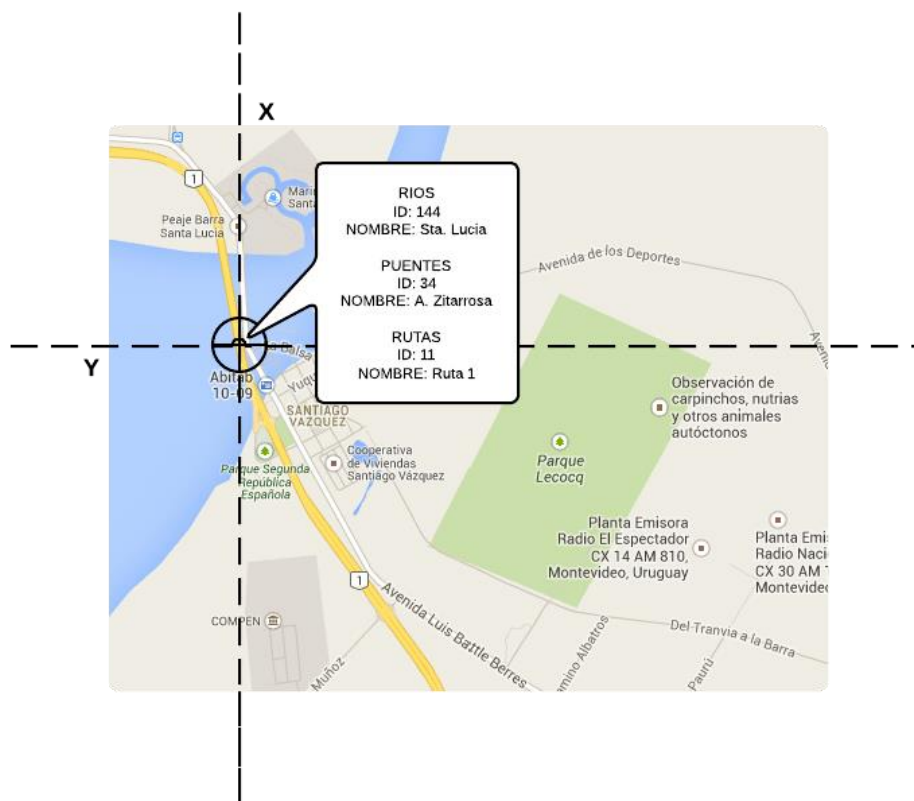


Figura 3 – Ejemplo de consulta GetFeatureInfo

2.1.2.2. WFS

Web Feature Service define un protocolo para consultar y modificar información geográfica en GML (una extensión de XML). A diferencia de WMS, el cual obtiene datos binarios de la información (a excepción de la operación GetFeatureInfo), WFS accede a la información geográfica de una forma detallada a nivel de entidades geográficas y sus propiedades. Esta respuesta puede ser una representación vectorial de la capa, en lugar de una imagen [7]. La versión 1.1.0 define las especificaciones que se describen a continuación en esta sección.

Dentro de WFS existen los siguientes tipos:

- WFS Básico, en la cual el servidor debe implementar las siguientes operaciones: *GetCapabilities*, *DescribeFeatureType* y *GetFeature*.
- WFS Transaccional, debe cumplir con las especificaciones de WFS básico y adicionalmente implementar la operación *Transaction*.

Operaciones de WFS:

GetCapabilities

Es mandatoria y obtiene la metadata con la información del servicio que se provee y de los parámetros válidos y sus valores. De sus atributos se destacan:

- SERVICE, que indica el protocolo al cual hace referencia (WFS).
- VERSION, para indicar la versión del protocolo usado.
- REQUEST, indica la operación.

A continuación se muestra un ejemplo de petición, en la cual se especifica el servicio (*WMS*), la versión del servicio (*1.1.0*) y la operación (*GetCapabilities*).

<http://geoweb.montevideo.gub.uy/geoserver/ows?service=wfs&version=1.1.0&request=GetCapabilities>

DescribeFeatureType

Solicita un esquema de descripción de un tipo de entidad geográfica. Dicha descripción define la codificación de las instancias de los objetos en las operaciones de consulta o en las transacciones. Para su invocación se pasa una lista separada por comas de los tipos de objetos a describir y como resultado se obtiene la definición de todos los tipos de objetos listados en la petición, en el formato especificado. De los parámetros se destacan:

- SERVICE, que indica el protocolo al cual hace referencia (WFS).
- VERSION, para indicar la versión del protocolo usado.
- REQUEST, indica la operación.
- TYPENAME, indica el nombre de la entidad geográfica a describir.

A continuación se muestra un ejemplo de petición, en la cual se especifica el servicio (*WFS*), la versión del servicio (*1.1.0*), la operación (*DescribeFeatureType*) y el nombre de la entidad geográfica a describir (*ide:ide_depto_p*).

http://geoweb.montevideo.gub.uy/geoserver/wfs?service=wfs&version=1.1.0&request=describefeaturetype&typename=ide:ide_depto_p

GetFeature

Obtiene una lista de objetos geográficos correspondientes a una capa. Esto se logra pasando por parámetros la lista con los nombres de los tipos que se consultan (se puede ver cómo una analogía a una clausula FROM del SELECT de SQL), y ciertas condiciones en las entidades de la respuesta (análogo a la cláusula WHERE de SQL).

La respuesta está condicionada a la cantidad de expresiones de consultas de la operación. En caso de contar con un única expresión de consulta (*single query*), el servidor responde con un elemento *wfs:FeatureCollection* conteniendo cero o más *wfs:member*, cada uno de éstos conteniendo o referenciando a un elemento del set de resultado de la petición.

Los parámetros más destacados son:

- SERVICE, que indica el protocolo al cual hace referencia (WFS).
- VERSION, para indicar la versión del protocolo usado.
- REQUEST, indica la operación.
- TYPENAME, indica el nombre de la entidad geográfica a consultar.

A continuación se plantea un ejemplo de petición, en la cual se especifica la operación (*GetFeature*), el servicio (WFS), la versión del servicio (1.1.0) y el nombre de la entidad geográfica a consultar (*ide:ide_depto_p*).

http://geoweb.montevideo.gub.uy/geoserver/wfs?request=GetFeature&service=wfs&version=1.1.0&typeName=ide:ide_depto_p

GetGMLObject

Permite obtener un elemento GML a través de su identificador “gml:id” único.

LockFeature

El objetivo de LockFeature es proveer un mecanismo de bloqueo a largo plazo para asegurar consistencia.

Transaction

Define una serie de operaciones Insert, Delete y Update, utilizadas para manipular los objetos del esquema utilizado. Las operaciones finalizan con un Commit o un Rollback.

2.1.3. GeoServer

GeoServer es un IMS que permite a los usuarios compartir y editar datos geoespaciales utilizando estándares de OGC. GeoServer es un producto open-source, desarrollado en Java por lo que el despliegue de la aplicación se realiza sobre cualquier servidor de aplicaciones que cumpla con la especificación de J2EE.[8]

A continuación se destacan algunas de las características de GeoServer:

- Implementa Web Feature Service (WFS) y Web Map Service (WMS)
- Soporte de formatos de respuesta JPEG, GIF, PNG, SVG y GML.
- Soporte completo de SLD, como definiciones del usuario (POST y GET), y como uso de configuración de estilos.

2.2. Sistemas empresariales

Según [9], “los sistemas de información empresarial constituyen el conjunto de recursos de la empresa que sirven como soporte para el proceso básico de captación, transformación y comunicación de la información.”. Estos sistemas pueden estar compuestos por simples aplicaciones empresariales que administren las operaciones y activos de una empresa, o por una combinación de aplicaciones intra o inter empresariales. Este último caso genera un esquema más complejo de interacción entre las aplicaciones dando lugar a variadas formas de brindar servicios y mecanismos de integración. Por ejemplo los modelos arquitectónicos Business to Business (B2B) o Enterprise Application Integration (EAI). El modelo B2B define transaccionalidad comercial entre partes, o intercambio de información. Por otra parte EAI refiere a una arquitectura en la cual se combinan tecnologías con el fin de integrar diversas aplicaciones empresariales [10]. Resulta interesante presentar las tecnologías más relevantes en que se basan estas arquitecturas, para entender la forma en que las aplicaciones empresariales se integran en la actualidad.

2.2.1. SOA

SOA es un patrón de diseño y arquitectura de software basado en un conjunto de piezas de software que brindan funcionalidades a modo de servicios. Estos servicios están orientados al uso por parte de usuarios finales, aplicaciones u otros servicios a través de interfaces públicas, que puedan ser descubiertas bajo demanda.

Un servicio se puede ver como una unidad funcional del conjunto que conforma una aplicación basada en SOA. Es así que este paradigma busca organizar y utilizar estas funcionalidades distribuidas, las cuales pueden estar bajo el control de los diferentes individuos involucrados e implementadas en variadas tecnologías, en una manera global y de bajo acoplamiento.

Se dice que las interacciones en una arquitectura SOA siguen el paradigma “find, bind and invoke”. Esto es porque los servicios son publicados en un registro público que luego las partes consumidoras utilizan para localizar, asociar a una funcionalidad utilizando el contrato provisto por el registro y finalmente consumir los servicios. [11]

2.2.2. Middleware

Se entiende por *middleware* el componente que colabora en la conexión de aplicaciones. Es un software que interconecta aplicaciones, accediendo a sistemas de menor nivel (Sistemas Operativos, Manejadores de Base de Datos, Servicio de Red). Este brinda interfaces de alto nivel para las aplicaciones, dando la idea de procesos locales.[12]

Los tipos de middleware se pueden separar en dos tipos, aquellos que comunican dos sistemas y los que comunican múltiples sistemas. Dentro de los que comunican dos sistemas tenemos a los Drivers para Manejadores de Bases de Datos, los Remote Procedure Call (RPC) y los Web Services (los cuales son utilizados en el desarrollo de este trabajo y son explicados en la sección 2.2.4). Los tipos de middleware que comunican múltiples sistemas, abarcan los Message Oriented Middleware² (MOM), Object Request Brokers (ORB), Integration Brokers y los Enterprise Service Bus (ESB) descriptos en la sección 2.2.5.

2.2.3. JMS

Java Message Service (JMS) es el estándar para la mensajería en Java EE. Se trata de una Interfaz de Programación de Aplicaciones (API) que provee una forma unificada de acceder a un MOM desde Java, evitando el uso de APIs propietarias.

Los componentes principales de la arquitectura son:

- Proveedores JMS que implementan la interfaz JMS y proveen funcionalidades de administración y control, por ejemplo la implementación de J2EE a partir de la versión 1.3 tiene un proveedor JMS integrado.
- Mensajes los cuales son objetos que comunican información entre clientes JMS.
- Clientes JMS desarrollados en JAVA que proveen y consumen mensajes.

² http://www.edwardcurry.org/publications/curry_MfC_MOM_04.pdf

JMS cuenta con dos modelos de comunicación [11]:

- **Punto a punto (PTP)**
Los productos o aplicaciones que siguen este modelo están contruidos según el concepto de colas, productores y consumidores de mensajes JMS.
Cada mensaje creado es direccionado a una cola específica, en tanto los receptores extraen los mensajes de una cola específica. Las colas se encargan de almacenar los mensajes enviados a los receptores hasta que sean consumidos o expiren.
Este modelo se aplica cuando hay un interés de ambas partes de que el mensaje sea entregado y procesado correctamente.
- **Productor- Suscriptor**
En este modelo, los productores direccionan sus mensajes a un tópico. Tanto productores como consumidores suelen ser anónimos y se pueden suscribir o publicar contenido de forma dinámica. El proveedor JMS se encarga de distribuir los mensajes que llegan a un tópico desde los diferentes productores y a los diferentes consumidores. A diferencia del modelo PTP, cada mensaje tiene múltiples consumidores y existe una dependencia de tiempos entre consumidores y productores, es decir un consumidor solo puede consumir mensajes enviados al tópico si previamente se suscribió al mismo.

2.2.4. Web Services

A partir de la definición de Web Service que brinda la W3C (World Wide Web Consortium), un Web Service es un software que permite la interoperabilidad “máquina-a-máquina”, a través de la red. Cuenta con una interfaz en formato procesable por una computadora, la cual está contenida en un archivo WSDL (definido en 2.2.4.1.1). Esto da lugar a que otros sistemas interactúan con el Web Service, utilizando la definición contenida en el WSDL y usando mensajes SOAP (ver sección 2.2.4.1) que típicamente hacen uso de HTTP con una serialización de XML, en conjunto con otros estándares relacionados con las tecnologías Web.[13]

En la definición anterior solo se abarca el concepto de Web Service SOAP, pero como se presenta a continuación dentro de los Web Services la W3C identifica dos grandes clases:

- Web Services que cumplen con la especificación REST (Representational State Transfer) que tienen como propósito primario manipular representaciones de recursos o datos en formato XML utilizando un conjunto de operaciones “Stateless” (operaciones que no cuentan con información del estado de ejecución del servicio) [13].
- Por otro lado los Web Services que exponen un conjunto arbitrario de operaciones definidas por cada uno, utilizando el estándar WSDL (Web Service Definition Language). Generalmente estos servicios intercambian mensajes con la aplicación mediante HTTP (protocolo de transporte más usual). Dichos mensajes tienen formato XML y específicamente cumplen con la codificación dada por SOAP (Simple Object Access Protocol) [13].

2.2.4.1. Web Services SOAP

Los Web Services SOAP utilizan como protocolo de transporte de mensajes el estándar SOAP [14] y para describir su interfaz, forma de acceso y ubicación, utilizan el lenguaje WSDL.

SOAP es un protocolo simple orientado a intercambiar información en un ambiente distribuido y descentralizado que utiliza mensajes XML. Define mecanismos para utilizar distintos protocolos de

transporte para el envío de mensajes. Además especifica un modelo de procesamiento que indica cómo se deben procesar los mensajes y una forma de adjuntar datos no-XML a los mensajes.

Un mensaje SOAP está compuesto por tres partes: como elemento principal se encuentra el SOAP Envelope el cual deberá tener el *namespace*³ especificado en el protocolo y contendrá como elementos hijos un nodo SOAP Header opcional y un nodo SOAP Body (ver Figura 4). El nodo Header se utiliza como modo de añadir características adicionales al mensaje como pueden ser variables de seguridad, enrutamiento o codificación y el nodo Body es el contenedor de la información a transmitir por el servicio. La Figura 5 muestra un ejemplo de un mensaje SOAP en formato XML, identificando cómo se expresa el Header y Body del mensaje.

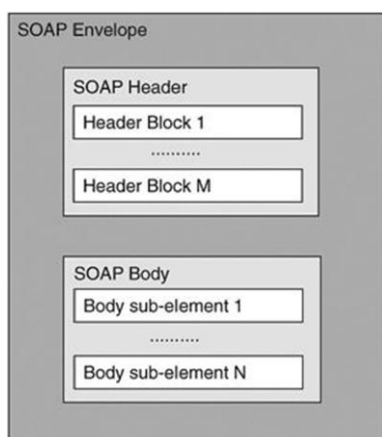


Figura 4 – Mensaje SOAP (estructura) [15]

```
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope">
  <env:Header>
    <pns:qualityOfService xmlns:pns="http://example.org/qos">
      <pns:priority>3</pns:priority>
      <pns:timestamp>2004-02-25T01:00:00-03:00</pns:timestamp>
      <pns:persist>true</pns:persist>
    </pns:qualityOfService>
  </env:Header>
  <env:Body>
    <bmns:businessPO xmlns:env="http://example.org/po">
      <bmns:description>Widgets</bmns:description>
      <bmns:quantity>100</bmns:quantity>
      <bmns:price>20.5</bmns:price>
    </bmns:businessPO>
  </env:Body>
</env:Envelope>
```

Header

Body

Envelope

Figura 5 - Mensaje SOAP (ejemplo)[15]

Web Service Definition Language (WSDL) [16] es un lenguaje de descripción para los Web Services con el que se proporciona al cliente toda la información necesaria para poder interactuar con el servicio (interfaz, forma de acceso y ubicación).

Los documentos WSDL definen los servicios como colecciones de puntos finales de red o puertos. En WSDL, la definición abstracta de los Endpoints (interfaz implementada por los puntos de acceso a un servicio) y de mensajes se separa de la implementación concreta de la red o de los enlaces del formato de datos. Esto permite la reutilización de las siguientes definiciones abstractas: *mensajes*, que son descripciones abstractas

³ http://www.w3schools.com/xml/xml_namespaces.asp

de los datos que se están intercambiando y *tipos de puerto*, que son colecciones abstractas de operaciones. Las especificaciones concretas del protocolo y del formato de datos para un tipo de puerto determinado constituyen un enlace reutilizable. Un puerto se define por la asociación de una dirección de red y un enlace reutilizable, mientras que una colección de puertos define un servicio.

2.2.4.2. Web Services REST

REST[17] identifica una clase de Web Services diferentes a los definidos por SOAP. La sigla REST, que significa Representational State Transfer, es un estilo de arquitectura que encapsula la noción de “Espacio de información compartida”. Este estilo provee un conjunto de restricciones que se enfocan en brindar mejoras en la escalabilidad, seguridad, eficiencia e independencia de los componentes en las aplicaciones WEB. Las URIs actúan como identificadores de recursos y los métodos HTTP como verbos que especifican operaciones sobre los mismos.

Las características comprendidas por REST son[18]:

- Arquitectura Cliente/Servidor
- Protocolos “Stateless” (sin estado)
- Uso de Caché
- Interfaces uniformes
- Sistema en Capas
- Código a demanda (Code-On-Demand)

En la Figura 6 se plantea en el contexto de empleados, cómo éstos pueden ser manipulados mediante REST a través de los diferentes métodos (Verbos HTTP) que este provee.

Verbo HTTP / URI	Significado en términos de CRUD
POST emps	Crear un nuevo empleado a partir de los datos de la solicitud.
GET emps	Leer una lista de todos los empleados.
GET emps?id=27	Leer el empleado 27.
PUT emps	Actualizar la lista de empleados con los datos de la solicitud.
DELETE emps	Eliminar la lista de empleados.
DELETE emps?id=27	Eliminar el empleado 27.

Figura 6 - REST, ejemplo de operaciones[15]

2.2.4.3. SOAP Vs. REST

Existe un gran debate en cuáles son las ventajas y desventajas que brindan estas dos familias de Web Services, pero su mayor diferencia es el objetivo primario con el cual fueron concebidas. En primer lugar los servicios REST buscan la simplicidad con el objetivo de brindar y transformar únicamente datos. Por otro lado, los servicios SOAP buscan dar acceso a lógica de aplicación, brindando grandes posibilidades de extender y/o customizar los componentes de su arquitectura de acuerdo a las necesidades de seguridad, confidencialidad, eficiencia, etc.

De esta manera tenemos que SOAP es más apropiado para integración de sistemas heterogéneos con requerimientos empresariales (escenarios tipo B2B y EAI). Por otro lado, REST está orientado a aplicaciones Web con gran cantidad de clientes y desconocidos (permite escalar fácilmente en cantidad de clientes)[15].

2.2.5. ESB

Según D. A. Chappell, “un ESB es una plataforma de integración basada en estándares que combina mensajería, Web Services, transformación de datos, enrutamiento inteligente, para conectar y coordinar diversas aplicaciones empresariales con integridad transaccional” [17].

La integración con un ESB implica que las aplicaciones no se comuniquen directamente, sino enviando mensajes a través del ESB, el cual puede realizar distintas operaciones de mediación sobre dichos mensajes, permitiendo asegurar que las aplicaciones se puedan conectar correctamente.

En la Figura 7 se presenta un ejemplo del panorama habitual en las interacciones de servicios empresariales, con una arquitectura donde los diferentes endpoints se conectan punto a punto (arquitectura *spaghetti*), siendo este un ejemplo donde no se utiliza un ESB como mediador.

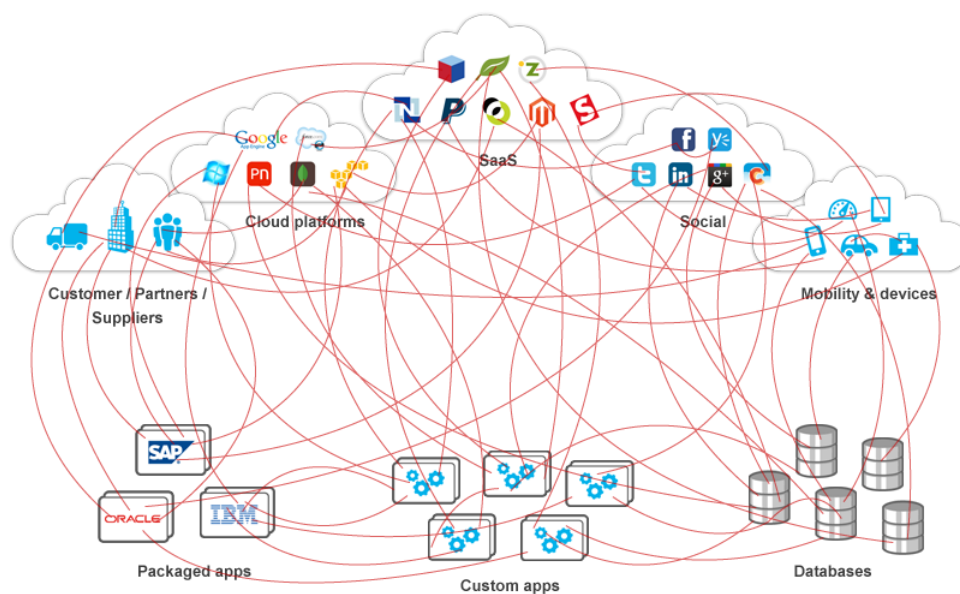


Figura 7 - Arquitectura Spaghetti[19]

Mediante el uso de un ESB se logra un desacoplamiento, sumando todas las funcionalidades que brinda para la interconexión (ver Figura 8).

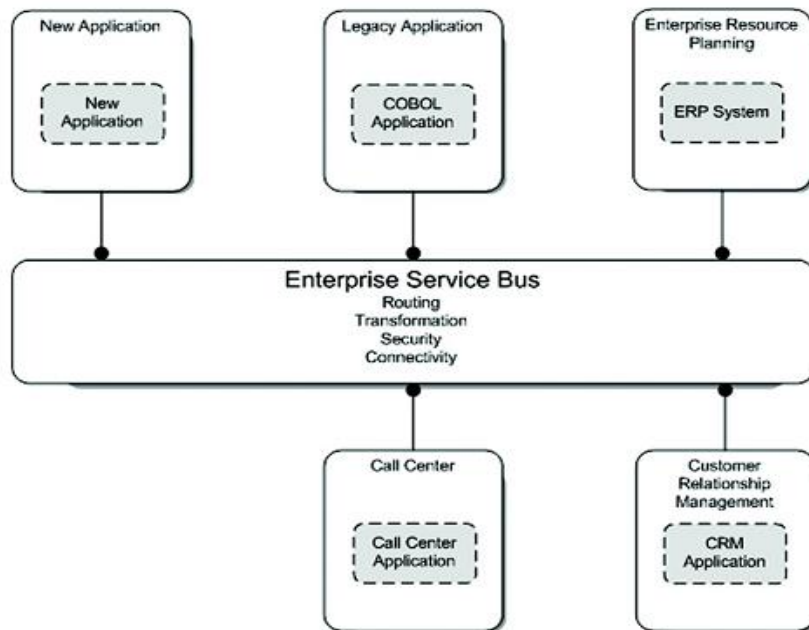


Figura 8 - Uso de ESB para integración de aplicaciones[20]

Los mensajes que fluyen a través del ESB son en general mensajes XML, permitiendo que los datos que son producidos y consumidos por un gran número de aplicaciones puedan coexistir en gran variedad de formatos y esquemas de representación. El ESB permite separar la lógica de integración y comunicación y además entre la lógica de negocio implementada por las aplicaciones, promoviendo el bajo acoplamiento entre las aplicaciones. [21][12]

2.2.5.1. Capacidades

A Continuación se presentan algunas de las capacidades de ESB.

- **Conectividad / Adaptadores**

Estos conectores son los que permiten al ESB interactuar con diferentes protocolos de comunicación, actuando como servidor o cliente. A su vez permiten la conversión de protocolo (utilizando dos adaptadores). Esto se puede ver en la Figura 9, en la cual una aplicación cliente envía un mensaje JMS al ESB, luego el “Adaptador JMS” despacha el mensaje al “Adaptador Sistema Archivos”, el cual se encarga de enviar mensajes a la aplicación legada en un formato legible por la misma.

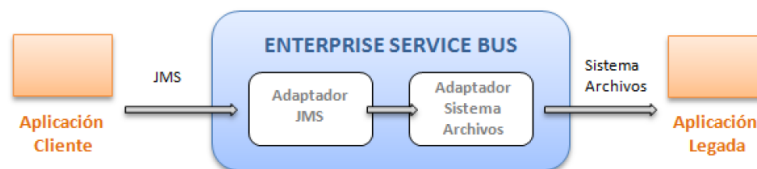


Figura 9 - Ejemplo de conectividad[22]

- **Transformación de Mensajes**

Los ESBs incluyen capacidades de transformación de mensajes, posibilitando que aplicaciones que utilizan distintos formatos o modelos de datos puedan comunicarse. En general, los ESB soportan transformaciones a través de XSLT, incluyendo también herramientas para transformar mensajes no XML (ej. Smooks). Esta capacidad podría utilizarse, por ejemplo, para dar soporte a diferentes versiones de un servicio, como en el caso para transformar SOAP 1.1 a SOAP 1.2 y viceversa. Por otro lado la Figura

10 muestra un caso donde se da soporte a un sistema legado, al transformar un mensaje SOAP de la aplicación cliente en un mensaje con un formato específico del sistema legado.

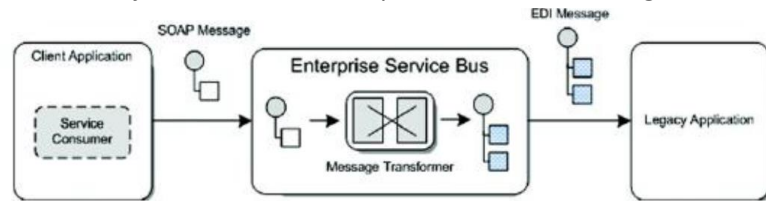


Figura 10 - Ejemplo de transformación[22]

- **Ruteo Inteligente**

Permite direccionar un mensaje de forma inteligente y determinar el destinatario (ver Figura 11). La lógica de ruteo está condicionada a factores como son el contenido del mensaje, o datos en el entorno de ejecución. Además los ESB pueden utilizar herramientas para realizar el ruteo (un ejemplo de esto es el motor de reglas Drools⁴)



Figura 11 - Ruteo inteligente[22]

- **Flujos de Mediación**

Combina una secuencia de operaciones de mediación a ejecutar en un mensaje, para generar un flujo de operaciones. La Figura 12 muestra como la aplicación A se comunica con las aplicaciones B y C. Esto se realiza mediante un ruteo en el ESB, el cual determina si el mensaje enviado va directamente a la aplicación B, o si es previamente transformado (dentro del ESB), para luego ser enviado a la aplicación C. La mediación queda determinada por los componentes de ruteo y transformación que se recuadran en la imagen.

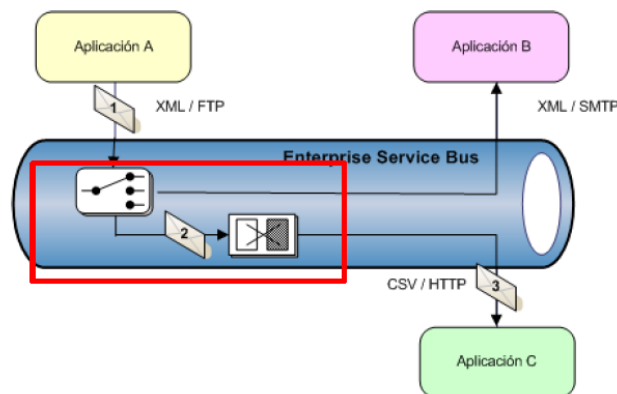


Figura 12 - Flujo de Mediación[22]

- **Mensajería Asincrónica**

Generalmente dada por el uso de un MOM incluido en el ESB.

⁴ Drools - <http://www.jboss.org/drools>

2.2.5.2. Patrones de Mediación

Los patrones de mediación son un punto crítico en el funcionamiento del ESB. Si bien las operaciones de mediación no están limitadas a estos patrones, se torna de gran utilidad para el desarrollador, el hecho de contar con estos patrones dentro de la implementación de ESB seleccionada. Mediante estos se pueden generar diversos tipos de flujos, cumpliendo con una amplia gama de necesidades de integración.[22]

- **Ruteo**

- Ruteo basado en contenido (Content-based router):

- Determina el destino del mensaje basándose en el contenido del mismo.

- Ruteo basado en contexto (Context-based router):

- Determina el destino en base a alguna propiedad del contexto de ejecución del mensaje.

- Balanceo de carga con Ruteo (Load-balancing router):

- La estrategia de balanceo de carga determina el destino.

- Divisor (Splitter):

- Este componente divide el mensaje entrante en varios mensajes de salida individuales, cada uno dirigido a un destinatario específico.

- Lista de Receptores (Recipient List):

- Clona el mensaje y lo envía a cada uno de los servicios de ESB definidos como destinos.

- Agregador (Aggregator):

- Se encarga de consolidar en un solo mensaje, varios mensajes que le llegan identificados unívocamente como parte del mismo conjunto. Utilizando un filtro Stateful, que recolecte y almacene los mensajes individuales hasta completar un cierto conjunto. Luego el agregador publica un mensaje a partir de los mensajes individuales que recolectó. Para la consolidación en un único mensaje se utiliza una condición de completitud y un algoritmo de agregación (transformación de múltiples mensajes en uno sólo).

- Filtrado de Mensajes (Message Filter):

- Elimina los mensajes que no cumplen con una cierta regla. La lógica de ruteo debe contener dicha regla.

- Ruteo basado en Itinerario (Routing Slip)

- El ruteo basado en itinerario determina el destino de un mensaje en base a un itinerario determinado El itinerario es metadata que se incluye en el mensaje y especifica al ESB el camino que debe seguir el dicho mensaje.

- **Transformaciones**

- Transformación de Contenido:

- Transforma el contenido del mensaje de acuerdo a lógica de transformación previamente definida para todos los mensajes que pasen por este componente.

Transformación de Modelos de Datos:

Adapta el contenido de un mensaje a un formato de salida diferente al original. El mensaje resultante se adapta a un nuevo esquema de definición sin cambiar el formato del mismo.

Transformación de Formato de Datos:

Adapta el contenido del mensaje, pero en este caso pasa de un formato de datos a otro. Un ejemplo de esto puede ser de JSON a XML.

Enriquecimiento de Contenido:

Enriquece el contenido, agregando información requerida por algún servicio posterior o el endpoint destino.

Filtrado de Contenido:

Quita información del mensaje original.

- **Administración**

Wiretap:

Envía una copia de los mensajes a otro destino que se encargue del monitoreo y análisis.

2.3. Soluciones analizadas

Se analizaron diferentes soluciones en la integración de información geográfica con fuentes de datos empresariales con el objetivo de conocer otros trabajos dentro del área e identificar posibles aportes al proyecto. En cada sección se desarrolla el contexto, solución y posibles aportes de cada integración.

Dentro de los documentos analizados, destacamos:

- “Integración de Servicios Geográficos en Plataformas de Gobierno Electrónico” [19]
- “Integración de Información geográfica en sistemas empresariales” [20]
- “WFS en ESB con Fuente Externa y Caché” [21]
- “WMS SOAP con ESB” [23]
- “SAP /GIS Integration Case Studies & Techniques” [24]

2.3.1. Integración de servicios Geográficos en Plataformas de Gobierno Electrónico

Contexto

En el referido documento [19] se realiza el análisis en detenimiento de los escenarios de integración de Servicios Geográficos en el marco de la Plataforma de Gobierno Electrónico⁵ (PGE) de AGESIC⁶.

⁵ http://www.agesic.gub.uy/innovaportal/v/452/1/agesic/plataforma_de_gobierno_electronico.html

⁶ <http://www.agesic.gub.uy>

Entre los escenarios planteados, el primero “Consulta de información pública”, tiene como objetivo brindar servicios de información geográfica de acceso público a través de la PGE. Este trabajo tiene grandes similitudes con la realidad del presente trabajo, ya que delega la interacción con los servicios geográficos y externos a una plataforma de Middleware. La consulta de información geográfica se puede realizar a través de distintos clientes (visualizador Web - navegador, aplicación *desktop*) hacia la plataforma mencionada.

Solución implementada

De la arquitectura presentada se destacan los siguientes actores: el llamado Organismo Proveedor de la Información, la PGE como mediadora y varios clientes que pueden ser tanto instituciones como público en general.

Lo que se destaca de esta solución, es el papel de PGE para brindar un acceso único a través de la misma, hacia el proveedor de datos geográficos (utilizando los estándares WMS y WFS).

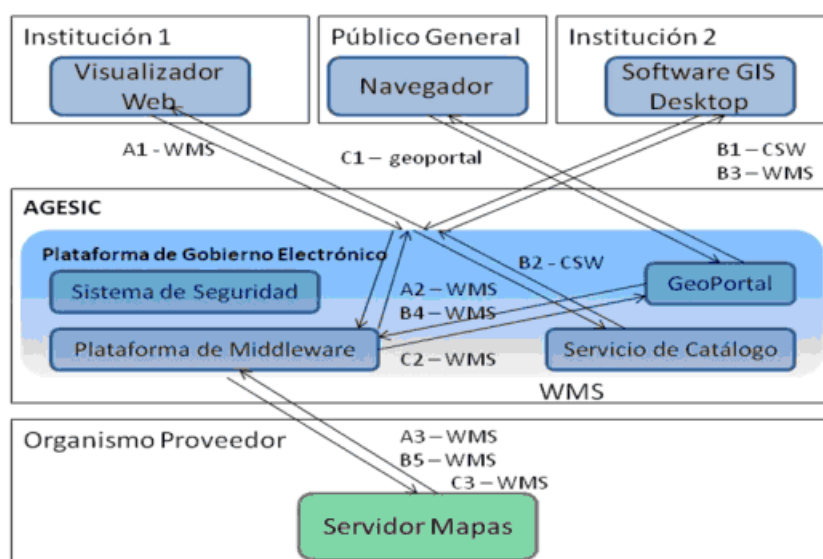


Figura 13 - Escenario de Consulta de información geográfica a través de PGE [23]

En la Figura 13 se observa la división en capas de la arquitectura y cómo los clientes interactúan con la PGE, para acceder a los datos geográficos.

Aportes

Del escenario planteado en la Figura 13, se obtuvo un ejemplo práctico de la manera en que se puede utilizar un middleware para el acceso a datos geográficos, siendo esto una de las partes fundamentales para el desarrollo de la plataforma.

2.3.2. Integración de servicios geográficos en sistemas empresariales

Contexto

Se analizó el Informe de proyecto “Integración de Información Geográfica en Sistemas Empresariales”, que tal como refiere su título tiene como objetivo principal la integración de servicios geográficos con información empresarial obtenida a través de Web Services SOAP.

Este trabajo surge en parte por la necesidad del Banco de Previsión Social⁷ (BPS) de contar con una aplicación en la cual se brinde al usuario la posibilidad de seleccionar el local de cobro de su preferencia. En dicha aplicación se debe contar con la información geográfica enriquecida.

El objetivo es crear una herramienta gráfica destinada a desarrolladores, que permita configurar las integraciones entre datos geográficos y datos de negocio. Además contar con la posibilidad de publicar a través de servicios geográficos estándar los datos integrados previamente definidos. Por último desarrollar un caso de estudio que utilice la solución para resolver el problema planteado y que sirva de base para construir aplicaciones de características similares. [20]

Solución implementada

Se presenta en su solución el concepto de datos integrados, como entidad contenedora de información tanto geográfica como empresarial.

El sistema final recibe peticiones WMS. En caso de operaciones como GetMap en las cuales no existe un posible enriquecimiento, se consulta solamente el servicio geográfico devolviendo un objeto integrado que solo posee la información original de la petición WMS al IMS. Por el contrario, en una operación como GetFeatureInfo se realiza una consulta al IMS y otra al WS destinado a enriquecer la información (tarea que lleva a cabo un componente llamado Mediador). Esta información es unificada devolviendo la entidad compuesta (el módulo encargado es el Integrador). Para obtener la información enriquecida se utilizó la unificación de identificadores, donde a un identificador de un objeto de capa geográfica (entidad geográfica) le corresponde un identificador de objeto de negocio.

Aportes

Este informe presenta una solución a las mismas problemáticas de integración, que se plantean para el desarrollo de GeoEEIP. En la solución planteada se optó por desarrollar componentes específicos para resolver el enriquecimiento en el contexto de los datos de BPS. Como conclusión se detectó la necesidad de generalizar los métodos de integración y enriquecimiento, brindando flexibilidad en el uso de la plataforma para distintos tipos y formatos de datos.

Otro aporte de gran utilidad es el análisis realizado en el informe sobre el caso de estudio de los locales de cobro de BPS, ya que es el mismo escenario a utilizar para el caso de estudio de la plataforma GeoEEIP.

2.3.3. WFS en ESB con fuente externa y caché

Contexto

El problema planteado consiste en analizar el uso de un ESB para integrar datos de negocio con un servidor de mapas y mejorar la performance de las consultas mediante el uso del caché.

El principal objetivo de este trabajo es brindar un servicio WFS enriquecido, a partir de información geográfica y empresarial (combinada). El segundo objetivo contempla el uso de caché, de forma de mejorar la performance y evitar los problemas relacionados con la no disponibilidad de alguno de los servicios. Ambos servicios se integran dentro de un ESB. [24]

⁷ <http://www.bps.gub.uy/>

Solución implementada

Se provee de una solución en la cual se cuenta con un cliente que realiza las peticiones al ESB, quien las direcciona según sean consultas WFS o WMS. En el caso de operaciones que no sean GetFeature, el ESB obtiene la respuesta directamente del IMS y la reenvía al cliente.

Si la operación es GetFeature, entra en juego el caché, obteniendo la respuesta del mismo en caso de ser posible, de lo contrario se accede al IMS y se guarda la respuesta en el caché. Luego esta respuesta es parseada extrayendo el identificador del elemento geo-referenciado, para junto con la respuesta completa realizar la consulta a los datos de negocio y realizar así el enriquecimiento. El mensaje devuelto es el XML retornado por el IMS con el agregado de negocio.

Aportes

En este proyecto se utiliza el patrón de ruteo basando en contenido “Content Based Router” de ESB, para direccionar las consultas dentro del mismo, en particular utilizando expresiones regulares para distinguir los distintos tipos de pedidos. Esta forma de direccionamiento nos sirve de ejemplo para dirigir el mensaje según las distintas peticiones a servicios geográficos.

2.3.4. WMS SOAP con ESB

Contexto

El documento presenta un análisis de la problemática referente a no contar con SOAP para la comunicación con los Web Services geográficos. Para trabajar con ellos se utilizan servicios REST con WMS. En este contexto la investigación se centra en la traducción SOAP de las solicitudes WMS y viceversa. Además se realiza un prototipo utilizando las funcionalidades del ESB. [25]

Solución implementada

El prototipo muestra una interacción de componentes que resuelve el problema, convirtiendo un pedido WMS en un pedido SOAP (GateWay dentro de JBossESB). Luego este pedido es derivado a otro componente según el tipo de operación solicitada, utilizando un Content Based Router para interpretar el pedido. Los componentes implementados permiten direccionar la petición directamente al IMS sin la necesidad de un componente intermediario. Previamente se direcciona el mensaje a un componente encargado de la transformación de SOAP a WMS. El siguiente paso en el flujo de datos, luego de la consulta al IMS es enviarlo a través del ESB para ser transformado a SOAP nuevamente, para así ser retornado como salida del ESB.

Aportes

El uso del patrón Aggregator en JBossESB para reunir el conjunto de respuestas relacionadas y enviar el mensaje correspondiente al cliente, resulta útil en los casos que se requiera reunir las respuestas de los Web Services Empresariales (para enriquecimiento con múltiples fuentes de datos).

2.3.5. Integración de sistemas geográficos y sistemas desarrollados sobre SAP ERP

Contexto

En este documento analizado [26] se plantea la necesidad de enriquecer los procesos de administración de recursos de los sistemas Enterprise Resource Planning (ERP) con la información y herramientas geográficas de los sistemas GIS. En particular se destaca cómo puede mejorar la toma de decisiones y el análisis de datos en el ámbito de las compañías petroleras de Arabia Saudita. En este escenario, mapas y datos georreferenciados juegan un importante papel, por ejemplo: manejo de tierras, selección del sitio de

trabajo, diseño en el tendido de cañerías, actividades de construcción, mantenimiento de instalaciones, respuesta a emergencias y muchos más.

Frente a esta necesidad es que dos compañías, SAP y ESRI de tecnologías ERP y GIS respectivamente, identificaron un conjunto de opciones y métodos de integración entre sus sistemas principales. Este conjunto está formado por 5 interfaces: conectores SAP RFC, conectores desarrollados por terceros, conector genérico para GIS de SAP, componentes “Enterprise Application Integrators” (EAI) de SAP, EAI desarrollados por terceros y soluciones de integración de terceros. [26]

Solución implementada

Los métodos de integración mencionados comprenden:

A. Conectores SAP RFC

Los conectores RFC (Remote Function Calls) permiten realizar llamadas remotas entre SAP ERP y cualquier sistema externo a SAP como lo es el sistema GIS de ESRI. Dentro de SAP se encuentran lo que llaman BAPIs que son conjuntos de “Remote Function Methods” (RFM), estos son métodos asociados a métodos de objetos de negocio del sistema SAP y se encuentran públicos para uso por parte de los desarrolladores. El conjunto básico de RFC puede ser utilizado para integrar SAP y GIS, para crear nuevas BAPI’s más complejas.

B. Conectores de terceros

Los conectores propietarios se pueden desarrollar utilizando lo que se conoce en SAP como “Control Brokers” que son componentes que tienen como objetivo solucionar la comunicación a “legacy backends” a nivel de la aplicación de ERP.

C. Conector genérico GIS de SAP

El componente SAP GBC (GIS Business Connector) es un middleware pasivo que media entre una solución SAP y un sistema ESRI GIS. Este mecanismo se encuentra disponible en SAP a partir de la versión 4.5.

D. EAI de terceros

EAI es un modelo arquitectónico, que tiene como objetivo la integración de datos entre sistemas mediante un conjunto de patrones, tanto para el acceso a datos como para la integración. SAP cuenta con una plataforma de EAI, pero en los casos de que los procesos no se basan en estándares y necesitan desarrollo específico es que se utiliza este tipo de solución.

E. SAP Exchange Infrastructure (EAI)

Esta es la plataforma de SAP que implementa el modelo EAI muy similar al caso anterior ya que facilita el intercambio de información entre los dos sistemas. SAP XI es considerado un intermediario de integración por mediar entre entidades con diferentes requisitos, como puede ser protocolos y formatos. Esta plataforma reduce en cierta medida los costos de desarrollo, por proveer un repositorio común para las interfaces a utilizar en la integración.

F. Soluciones de integración de terceros

Existen productos que integran directamente los sistemas SAP con ESRI GIS como puede ser los programas de la empresa IMPRESS [27] o el componente SICAD-APX de la empresa AED-SICAD [28]

que puede ser útil en casos que el tiempo de desarrollo sea una limitante importante y no el costo de estas soluciones privadas.

Aportes

El documento realiza un análisis sobre la opción a implementar según cada escenario, centrándose en tiempos y costos. Para luego dar paso a un conjunto de casos de estudio o experiencias en las que fueron utilizados algunos de los métodos planteados y así demostrar los beneficios de integrar tecnologías GIS con procesos de negocio.

Dos de estas experiencias a destacar son:

- Sistemas que utilizan la representación de objetos de SAP (recursos, instalaciones y áreas de trabajo) sobre mapas de GIS enriquecidos con información adicional, para el diseño de protocolos de manejo, almacenaje y desecho de materiales peligrosos, analizando las propiedades del terreno, para así utilizar los trayectos más cortos y óptimos para el traslado.
- Una aplicación que utiliza GIS orientada al mantenimiento de cañerías que se utiliza desde dispositivos móviles y desde la cual es posible crear objetos de negocio en el sistema SAP para llevar el seguimiento de inspecciones, el estado de las líneas inspeccionadas y la creación de solicitudes de mantenimiento.

2.4. Resumen

A modo de resumen de los conceptos, tecnologías e investigaciones vistas en el capítulo, se presenta el marco en el cual se realiza la integración de datos geográficos y empresariales.

En esta sección se plantea la integración en términos de enriquecer datos de sistemas geográficos con datos de sistemas empresariales. En las subsecciones anteriores se presentaron los conceptos involucrados en las integraciones.

Las aplicaciones que utilizan los sistemas de información geográficos, cada vez más tienen como objetivo agregar datos de negocio relevantes para el usuario, de forma de brindarle mayor información relacionada con la geolocalización. A su vez, las aplicaciones de negocio buscan darle a sus datos empresariales un contexto georreferenciado, para obtener mejores herramientas de análisis de los mismos. Para esto es necesario integrar la información que brindan los servicios geográficos, con la información de negocio.

Cabe destacar que los servicios geográficos no cuentan con mecanismos de integración propios y que resulta inviable almacenar los datos de negocio en las bases de datos geográficas. Esto se debe a que se generaría una redundancia de datos al momento de usarlos en más de un sistema geográfico diferente. Otra razón es que los servicios geográficos sólo deben encargarse de datos geográficos, así como los datos de negocio deben estar en fuentes de datos de negocios (por ejemplo a través de Web Services), ya que dicha información en la mayoría de los casos existe y es parte del sistema empresarial de una organización o de organizaciones independientes (como puede suceder en una SOA).

Ante esta problemática planteada, las soluciones vistas (sección 2.3) buscan utilizar los sistemas existentes especializados en la interconexión de servicios. Sin embargo, se observa que se encuentran de algún modo limitadas, en algunos casos por su implementación y en otro por haber sido diseñadas para tecnologías y escenarios específicos.

En el contexto presentando se enmarca la solución del presente trabajo como la implementación de una solución genérica a la integración de servicios geográficos con servicios empresariales. En las siguientes secciones se desarrolla el análisis diseño e implementación de la solución planteada.

3. Análisis

En la presente sección se presenta un análisis de la problemática planteada. Esta problemática, refiere a la plataforma de integración basada en un ESB (GeoEEIP), la cual apoyándose en sus capacidades de mediación provee mecanismos de integración geográficos para abordar desafíos comunes que surgen al integrar sistemas empresariales tradicionales con Web Services Geográficos. En la sección 3.1 se realiza el análisis del documento de definición de GeoEEIP, basándose en [1]. La sección 3.2 describe los componentes identificados, explicando cómo se identifican y la descripción de cada uno de ellos. En la sección 3.3 se describe el producto JBossESB (implementación de ESB a utilizar en la plataforma). La sección 3.4 presenta las alternativas de arquitecturas evaluadas. En la sección 3.5 se analiza la aplicación de configuración que complementa el desarrollo de la plataforma.

3.1. Análisis del documento de definición de GeoEEIP

En el artículo en el cual se basa este trabajo [1], se plantea la importancia de aprovechar los sistemas de información geográfica debido al auge vertiginoso que han experimentado, tanto para el análisis de datos por parte de usuarios expertos como para la visualización por parte del público general.

A partir de este análisis se plantea el desarrollo de una plataforma de integración basada en ESB la cual oficie de mediadora entre un sistema de negocio y los sistemas geográficos existentes. Esta solución está conformada por la especificación de un grupo de componentes de integración, los cuales se agrupan para resolver desafíos cotidianos a la hora de posibilitar el acceso a la información de negocio georreferenciada desde cualquier aplicación.

Los componentes se dividen en dos grandes categorías. La primera corresponde a los Componentes Básicos, los cuales se construyen a partir de componentes de mediación del ESB, para dar lugar a funcionalidades geo-orientadas o soluciones reutilizables en la integración de OWS con sistemas empresariales. La segunda categoría corresponde a los Componentes Complejos, los cuales representan mecanismos de mediación contruidos a partir de los Componentes Básicos y los de ESB. Los Componentes Complejos proveen soluciones reutilizables de alto nivel para integrar OWS con aplicaciones empresariales. En la Figura 14 se muestra el diseño, indicando las dependencias entre los diferentes tipos de componentes.

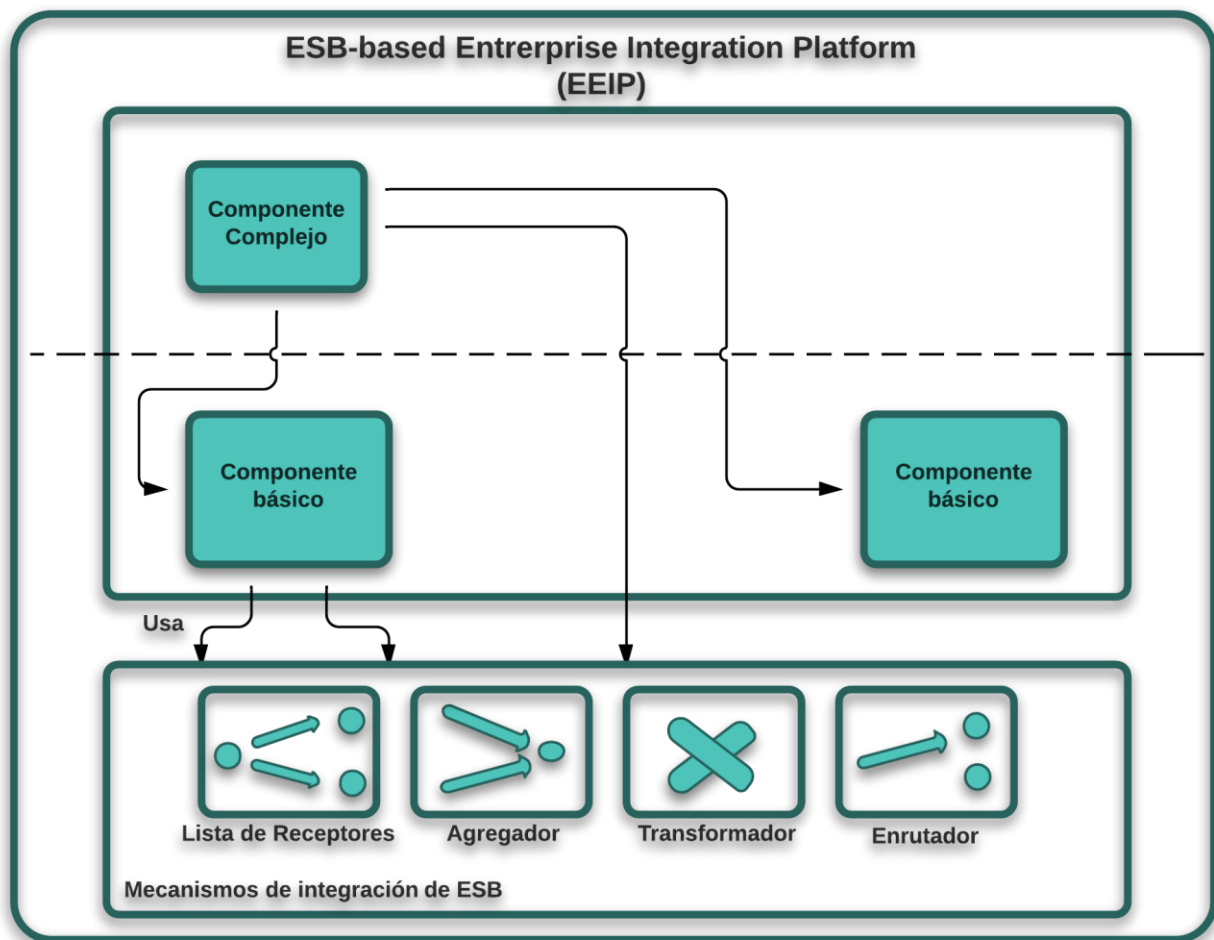


Figura 14 - GeoEEIP[1]

3.2. Componentes identificados

En esta sección se profundiza en cada uno de los mecanismos definidos en la plataforma de integración planteada en el artículo [1]. Para cada uno de estos mecanismos se analizan las funcionalidades requeridas y se identifican el o los componentes necesarios para su implementación.

En el análisis de cada funcionalidad se identifican los pasos en el procesamiento de un Mensaje ESB, y cada paso se traduce a un componente en la plataforma GeoEEIP a desarrollar. Los componentes se presentan indicando una descripción de su funcionalidad y un diagrama que muestra el flujo del Mensaje ESB.

3.2.1. Componentes Básicos de integración

Los Componentes Básicos de integración son mecanismos construidos a partir de los patrones de mediación presentados en la sección 2.2.5.2, por lo que proveen soluciones reusables en la integración de OWS con sistemas empresariales. A continuación se muestra el análisis de alguno de estos Componentes Básicos de integración. El resto de los Componentes Básicos se encuentran analizados en el Apéndice 1.

GeoEntryPoint

Mecanismo que brinda un punto de entrada a la plataforma GeoEEIP para el procesamiento de una petición HTTP, de uno de los protocolos posibles (WMS, WFS o WMS en formato SOAP). Como resultado genera un Mensaje ESB con la información obtenida en la petición correspondiente, con el formato esperado por la plataforma. El componente se configura con el protocolo a utilizar.

En la Figura 15 se muestra el flujo de procesamiento de una petición, en el “Punto de Entrada HTTP” se publica el punto de acceso para el protocolo configurado y luego en “Crea el Mensaje ESB” se crea el Mensaje ESB con la información de la petición de entrada procesada para su posterior uso en la plataforma GeoEEIP.

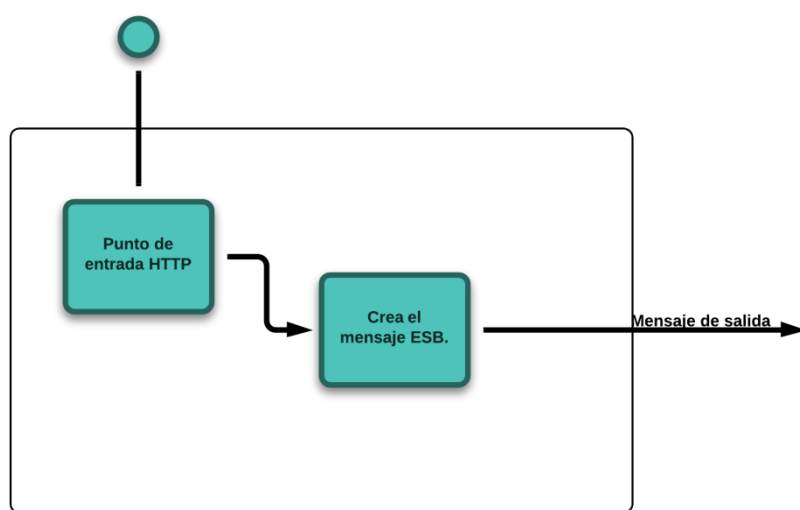


Figura 15 - GeoEntryPoint

GeoProxy

Mecanismo que permite obtener a través del IMS, la información geográfica correspondiente a la petición que se tiene en el mensaje de entrada, el resultado es almacenado en el mismo mensaje para su siguiente procesamiento. El componente se configura con la URL del IMS a consultar.

En la Figura 16 se muestra el procesamiento del Mensaje ESB en el GeoProxy. En la etapa “Lee el Mensaje ESB” se prepara la petición REST con los parámetros almacenados en el Mensaje. Luego se realiza la solicitud al IMS y por último se almacena la respuesta en el Mensaje.

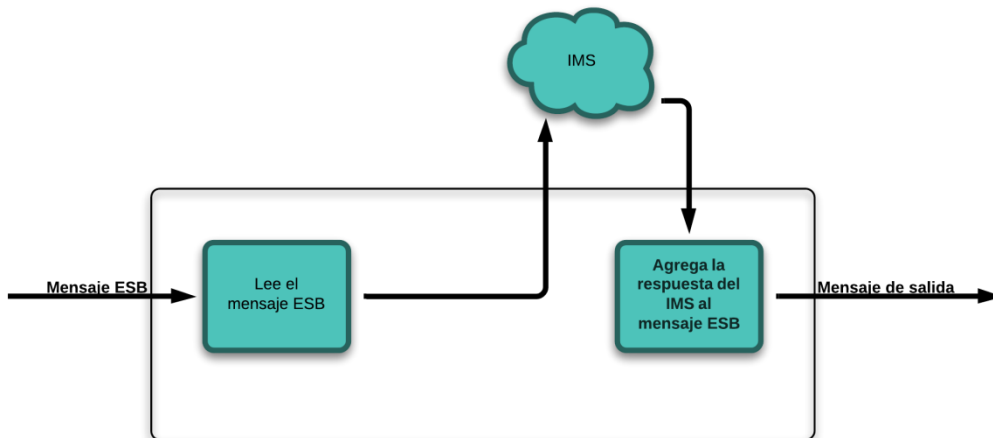


Figura 16 - GeoProxy

GeoRouter

Mecanismo encargado de direccionar un Mensaje ESB dependiendo de su contenido. Es decir que a partir del Mensaje y la configuración se identifica el siguiente paso. El procesamiento del Mensaje ESB se realiza utilizando el patrón de mediación “Ruteo Basado en Contenido” descrito en la sección 2.2.5.2. La Figura 17 presenta un escenario para el cual el componente tiene N direccionamientos posibles, según el contenido del mensaje.

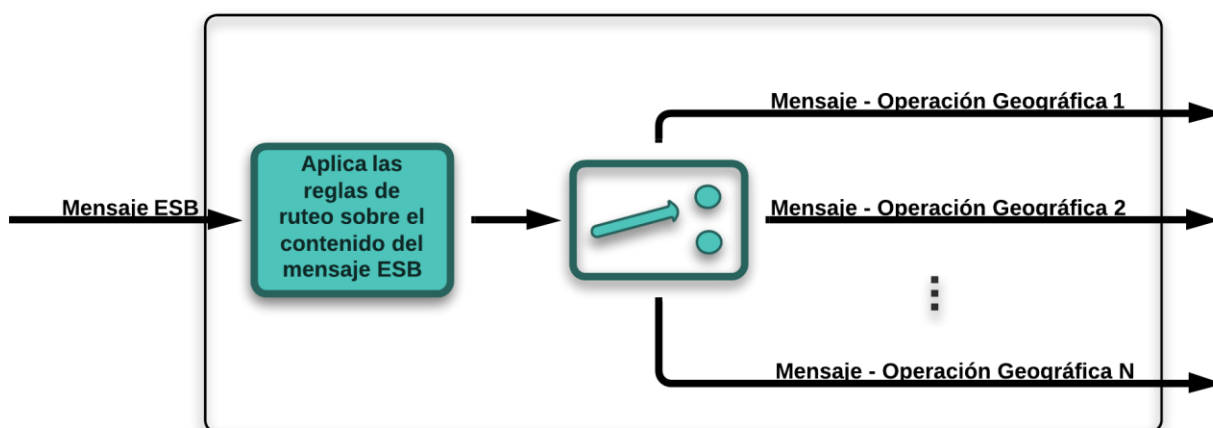


Figura 17 - GeoRouter

GeoFeatureInfoEnricher

Mecanismo encargado de enriquecer el resultado de la operación de WMS GetFeatureInfo con la información de negocio (resultado de la consulta a Web Services empresariales). En el procesamiento del mensaje se utilizan los patrones de mediación Lista de Receptores, Agregador y Transformador descritos en la sección 2.2.5.2.

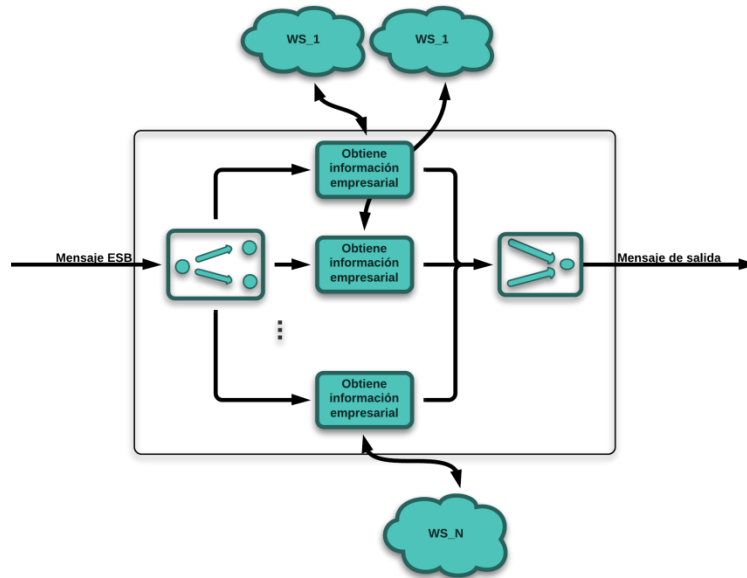


Figura 18 - GeoFeatureInfoEnricher

El mecanismo tiene como precondition que el Mensaje debe tener guardada la respuesta de GetFeatureInfo previo a la ejecución del mecanismo. Según se muestra en la Figura 18 el Mensaje ESB es procesado inicialmente aplicando el patrón Lista de Receptores, luego cada Receptor se encarga de consultar el Web Service Empresarial configurado. El procesamiento continúa aplicando el patrón Agregador, el cual se encarga de reagrupar los Mensajes en único Mensaje ESB aplicando el algoritmo de agregación. El algoritmo de agregación realiza el enriquecimiento de la información geográfica con la respuesta de los Web Services empresariales. A modo de ejemplo el enriquecimiento puede ser: agregar a la información de calles de la respuesta del GetFeatureInfo, los datos del tráfico proveniente de la respuesta a la consulta al WS empresarial.

SOAP2WMSTranslator

Mecanismo que realiza la conversión de un mensaje SOAP en una petición WMS (REST). Este mecanismo toma el mensaje SOAP del Mensaje ESB, aplica una transformación para obtener los parámetros de la solicitud y por último carga los parámetros en el mensaje de salida, este flujo se puede ver en la Figura 19.

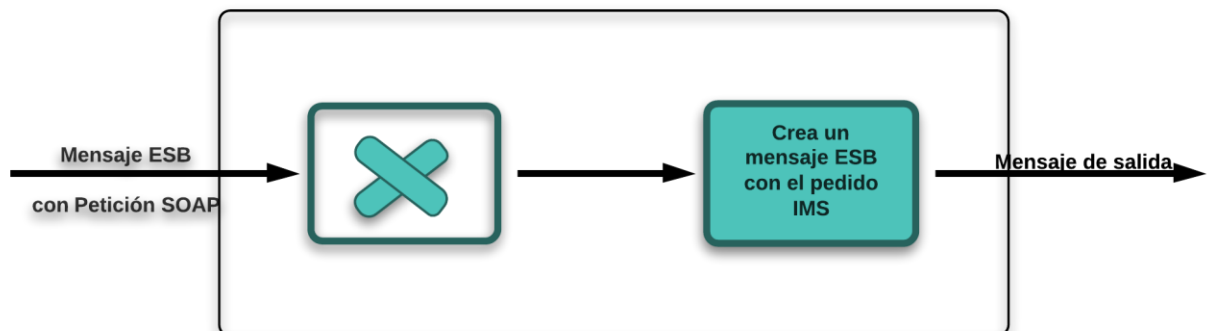


Figura 19 - SOAP2WMSTranslator

3.2.2. Componentes Complejos de integración

Los Componentes Complejos de integración son mecanismos de mediación, contruidos a partir de los Componentes Básicos y los patrones de mediación presentados en la sección 2.2.5.2, permitiendo así un alto nivel de reusabilidad en las soluciones de integración de los OWS con sistemas empresariales. A continuación se muestra el análisis de alguno de estos Componentes Complejos de integración, el resto de los Componentes Complejos se encuentran analizados en el Apéndice 1.

WMS Enricher

El mecanismo del WMS Enricher establece una solución completa a una de las problemáticas desarrollada en este trabajo para el caso de integración de respuestas WMS con datos empresariales. En la Figura 20 se tiene una vista a alto nivel del WMS Enricher, que muestra el uso de los Componentes Básicos para conformar el mecanismo.

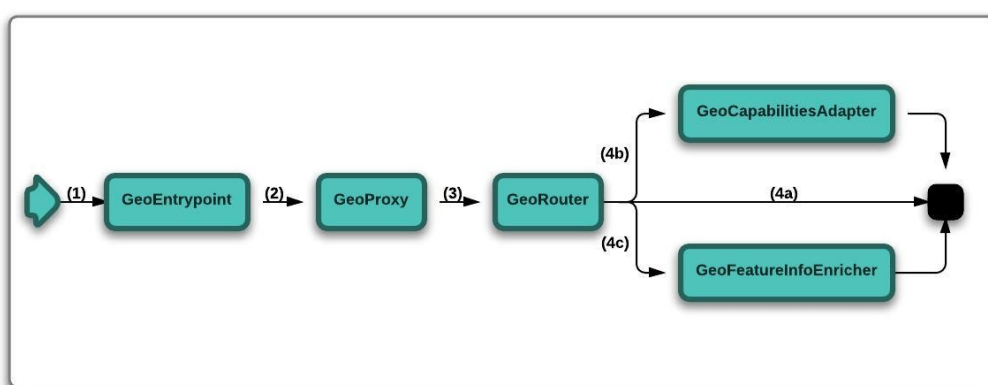


Figura 20 - WMS Enricher[1]

El cliente WMS en lugar de comunicarse directamente con el IMS, envía las solicitudes a GeoEEIP y cuando éste recibe la solicitud crea un mensaje ESB conteniéndola. El mensaje se procesa en secuencia según se muestra en la Figura 20 a través de los componentes GeoEntryPoint, GeoProxy y GeoRouter (descritos en la sección 3.2.1). En el GeoRouter se direcciona el procesamiento del mensaje según la operación consultada, GetCapabilities o GetFeatureInfo al componente GeoCapabilitiesAdapter o GeoFeatureInfoEnricher (descritos en la sección 3.2.1) respectivamente, si la operación WMS es GetMap no se requiere procesamiento adicional.

WFS Enricher

El mecanismo del WFS Enricher establece una solución completa a una de las problemáticas desarrollada en este trabajo para el caso de integración de respuestas WFS con datos empresariales. En la Figura 21 se tiene una vista a alto nivel de WFS Enricher, que muestra el uso de los Componentes Básicos para conformar el mecanismo.

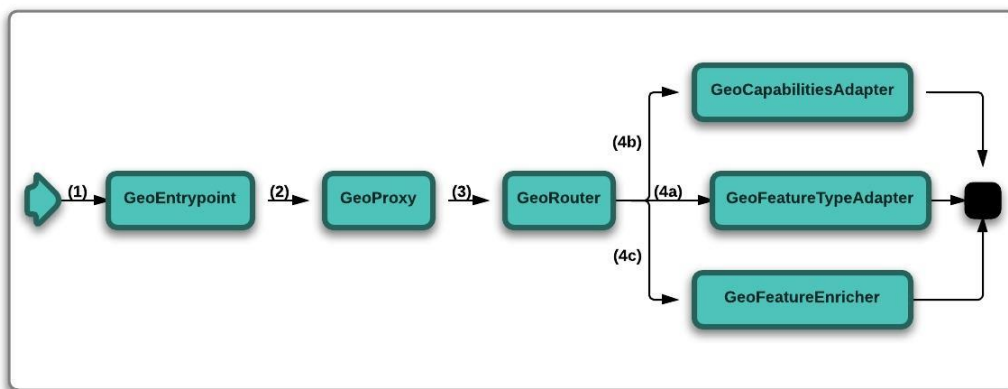


Figura 21 - WFS Enricher[1]

Al igual que el cliente WMS, para WFS el cliente realiza las peticiones a través de GeoEEIP y cuando éste recibe una solicitud crea un mensaje ESB conteniendo dicha solicitud. El Mensaje se procesa en los componentes GeoEntryPoint, GeoProxy y GeoRouter (descritos en la sección 3.2.1), como se muestra en la Figura 21. En el GeoRouter se direcciona el procesamiento del mensaje según la operación consultada, sea GetCapabilities, DescribeFeatureType o GetFeature al componente GeoCapabilitiesAdapter, GeoFeatureTypeAdapter o GeoFeatureInfoEnricher (descritos en la sección 3.2.1) respectivamente.

SOAP-WMS Wrapper

El mecanismo de SOAP-WMS Wrapper permite publicar operaciones de WMS utilizando los estándares de la W3C para Web Services, SOAP y WSDL. Con este mecanismo se permite integrar WMS con sistemas que tienen esos estándares como restricción. Por ejemplo si un sistema implementa orquestación de servicios web utilizando WS-BPEL⁸, la invocación a los servicios se puede realizar únicamente con mensajes SOAP. Por lo que el mecanismo SOAP-WMS permite que los procesos de WS-BPEL utilicen WMS.

El mecanismo SOAP-WMS Wrapper publica una versión SOAP de WMS según lo especificado en “OWS 1.2 SOAP Experiment Report”[29]. En la Figura 22 se muestra a alto nivel la estructura del mecanismo. Inicialmente el componente GeoEntryPoint recibe la petición SOAP realizada al Web Service publicado mediante mecanismos de ESB, el procesamiento continúa en los componentes SOAP2WMSTranslator, GeoProxy y GeoRouter descritos en la sección 3.2.1. El GeoRouter direcciona el mensaje según la operación WMS. Si la operación es GetMap el Mensaje ESB se direcciona al componente WMS2SOAPBinaryTranslator; en otro caso, se direcciona a WMS2SOAPTextTranslator.

⁸ <http://docs.oasis-open.org/wsbpel/2.0/wsbpel-v2.0.pdf>

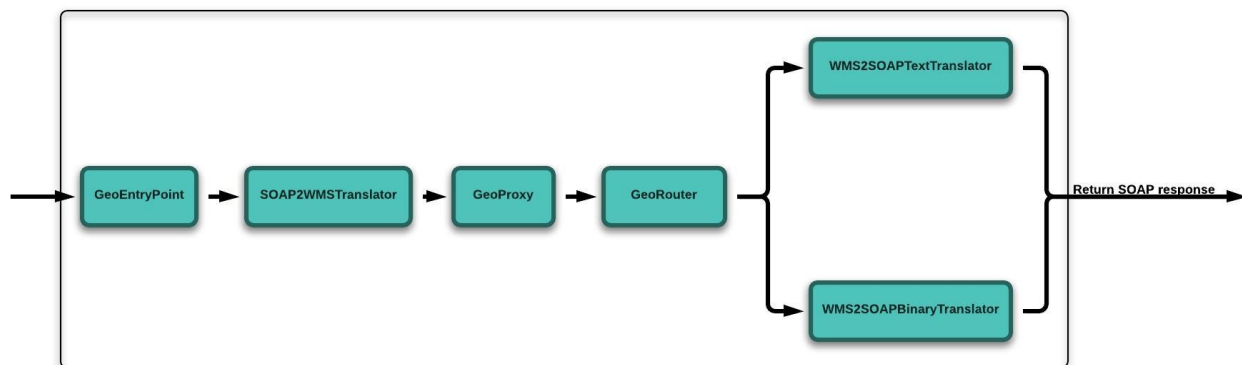


Figura 22 - SOAP-WMS Wrapper

3.3. JBossESB

JBossESB es una implementación de código abierto de un Enterprise Service Bus provista por JBoss de RedHat⁹. JBossESB brinda la interacción entre sistemas dispares abstrayendo sus diferencias y tratando a cada uno como un servicio lógico en el ESB. Los servicios encapsulan la lógica de integración, mientras que los mensajes son la forma en la que los servicios se comunican dentro de JBossESB.

En las siguientes secciones se detallan los Mensajes y Servicios de JBossESB y los distintos componentes disponibles dentro del producto.

3.3.1. Mensajes y Servicios

El Mensaje es la forma de encapsular la información relevante dentro del ESB, como ser: identificación, ruteo, seguridad e información de negocio, entre otros, permitiendo de esta manera comunicar de forma agrupada la información a ser utilizada por los componentes de JBossESB.

Los Mensajes de JBossESB están implementados mediante la clase `org.jboss.soa.esb.message.Message` y sus compontes principales son:

- Cabezal (Header): Información de la identidad y dirección de ruteo del Mensaje.
- Contexto (Context): Información de contexto para la entrega del Mensaje como por ejemplo información de seguridad.
- Cuerpo (Body): Carga útil del Mensaje y adicionales requeridos en el contrato del servicio.
- Adjunto (Attachment): Información adicional que puede ser referenciada como carga útil.
- Propiedades (Properties): Información relacionada con la entrega del Mensaje, por lo general datos específicos de transporte (por ejemplo nombre de la cola JMS original).

Un servicio es una implementación de lógica de negocio publicada para los clientes según un contrato previamente definido. El servicio permite brindar un contrato de servicio abstracto con: bajo acoplamiento, reusable y auto-contenido. Los servicios que se adhieren a este criterio adquieren las capacidades de desarrollo y escalado sin afectar a los usuarios del servicio. Por lo que, los usuarios no tienen que preocuparse de qué implementación del servicio va a ser invocada, ni donde está ubicado.

⁹ <http://www.redhat.com/>

En JBossESB los servicios se identifican por una categoría y un nombre; un servicio se compone de una lista de acciones (pipeline de acciones) que procesan los mensajes de forma secuencial como se ve en la Figura 23. En la implementación de un servicio, se divide la funcionalidad en un conjunto de acciones y cada una se implementa de forma independiente. Esto permite no sólo reutilizar los servicios, sino también las acciones implementadas, ya que éstas pueden utilizarse en múltiples servicios.

JBossESB incluye un conjunto de acciones reutilizables y permite construir acciones específicas. Las acciones provistas por JBossESB brindan funcionalidades tales como: transformación, ruteo y soporte a Web Services, entre otras[30].

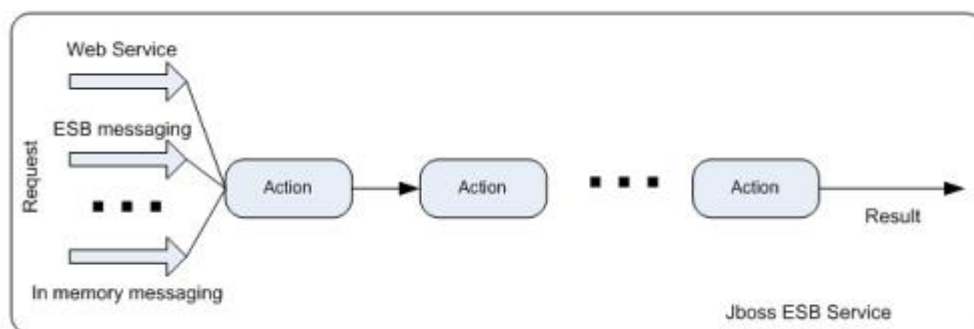


Figura 23 – Servicio JBossESB

Para simplificar el desarrollo existen dos clases abstractas (clases base), que provee JBossESB. Cada una define los métodos básicos necesarios para una acción así como el comportamiento básico de éstas. Se excluye de la definición de comportamiento al método responsable de la lógica específica de procesamiento (método “process”). Estas clases son `AbstractActionPipelineProcessor` y `AbstractActionLifecycle`, ambas pertenecientes al paquete `org.jboss.soa.esb.actions`. En la Figura 24 se ve un ejemplo de cómo se utiliza, apreciando los métodos básicos del ciclo de vida de una acción. En primera instancia el método “initialise” se encarga de inicializar el contexto de ejecución para la acción (variables y dependencias), luego el método “process” el cual ejecuta la lógica específica de la acción al momento de ser invocada y por último el método “destroy” encargado de restaurar el contexto de la acción al estado previo a la instanciación.

JBossESB especifica que el constructor para una acción recibe como parámetro un objeto “ConfigTree”. Éste corresponde a la estructura utilizada para representar la configuración de JBossESB para dicha acción. Por defecto se encuentra precargado al momento de la instanciación, con los valores definidos en el archivo de configuración de servicios (`jboss-esb.xml`).

```

public class ActionXXXProcessor extends AbstractActionPipelineProcessor {
    public ActionXXXProcessor(final ConfigTree config) {
        // extract configuration
    }

    public void initialise() throws ActionLifecycleException {
        // Initialize resources...
    }

    public Message process(final Message message) throws ActionProcessingException {
        // Process messages in a stateless fashion...
    }

    public void destroy() throws ActionLifecycleException {
        // Cleanup resources...
    }
}

```

Figura 24 - Acciones base de JBossESB

3.3.2. Componentes disponibles

Como se mencionó JBoss ESB ofrece distintas acciones nativas para procesar los Mensajes, en esta sección se presentan las más destacadas en el marco del proyecto. Estas acciones están disponibles en el paquete `org.jboss.soa.esb.actions` de la librería `jbossesb.jar`.

3.3.2.1. Ruteo Estático

El ruteo estático es implementado por la acción `org.jboss.soa.esb.actions.StaticRouter`. Los Mensajes que procesa esta acción son reenviados a todos los destinos configurados en tiempo de diseño[30]. Un ejemplo de su configuración se muestra en la Figura 25. En la misma se tiene un elemento “action” que representa la acción de JBossESB, declarando la clase que la implementa (incluyendo su paquete). Dentro del elemento “action” se encuentra el elemento “property”, con el atributo “name” conteniendo el valor “destinations”. Esta propiedad declara los destinos del ruteo, con la restricción de que deben ser servicios del JBossESB. Los elementos “route-to” son los que especifican los servicios de destino.

```

<action name="routeAction" class="org.jboss.soa.esb.actions.StaticRouter">
    <property name="destinations">
        <route-to service-category="source1" service-name="enterprise_source1" />
        <route-to service-category="source2" service-name="enterprise_source2" />
        <route-to service-category="source3" service-name="enterprise_source3" />
    </property>
</action>

```

Figura 25 – Configuración de Static Router

3.3.2.2. Ruteo basado en contenido

El ruteo basado en contenido está implementado en la acción `ContentBasedRouter` del paquete `org.jboss.soa.esb.actions`. El siguiente servicio por el cual va a continuar el Mensaje se decide cuando se lo procesa. JBossESB soporta varias formas de realizar el ruteo basado en contenido, las reglas de ruteo se

puede definir mediante Xpath¹⁰, expresiones regulares, Drools¹¹. Seguido se muestra un ejemplo de configuración. [30]

```
<action class="org.jboss.soa.esb.actions.ContentBasedRouter" name="contentRouter">
  <property name="cbrAlias" value="Regex" />
  <property name="ruleSet" value="/regex-georouter-rules.properties" />
  <property name="object-paths">
    <object-path esb="body.query_string" />
  </property>

  <property name="destinations">
    <route-to service-category="service-category1" service-name="service-name1" />
    <route-to service-category="service-category2" service-name="service-name2" />
    <route-to service-category="service-category3" service-name="service-name3" />
  </property>
</action>
```

Figura 26 – Configuración Ruteo Basado en Contenido

Los parámetros de configuración son:

- `cbrAlias`, indica el formato en que se define las reglas de ruteo (Xpath, expresiones regulares, Drools).
- `ruleSet`, indica la ruta del archivo de ruteo con las reglas en el formato indicado.
- `object-paths`, lista de referencias a campos del mensaje, utilizado como entrada en la evaluación de la reglas.
- `destinations`, al igual que en el ruteo estático indica los posibles servicios destino del Mensaje.

En la Figura 26 se plantea un ejemplo con expresión regular, el cual tiene como “ruleset” el archivo especificado, conteniendo las expresiones regulares para cada regla. El “object_path” corresponde a la propiedad “query_string” del cuerpo del mensaje ESB. Por último los destinos declarados son los servicios “service-name1”, “service-name2” y “service-name3”.

3.3.2.3. Agregador

El patrón Agregador es implementado por la acción `org.jboss.soa.esb.actions.Aggregator`. Los Mensajes que van llegando se agregan como adjuntos de un único Mensaje, para luego ser procesados por el pipeline de acciones. En la Figura 27 se muestra un ejemplo de configuración de la acción.

```
<action class="org.jboss.soa.esb.actions.Aggregator" name="Aggregator">
  <property name="timeoutInMillies" value="600000" />
</action>
```

Figura 27 – Configuración de Agregador

¹⁰ <http://www.w3schools.com/XPath/>

¹¹ <http://www.jboss.org/drools/>

La propiedad “`timeoutInMillies`” indica el tiempo de espera en milisegundos antes de que la espera de las respuestas se cancele por time-out[30].

3.3.2.4. Cliente Web Service SOAP

JBossESB provee mecanismos nativos para consumir y exponer Web Services, con las acciones `SOAPProcessor`, `SOAPClient` y `SOAPProxy`. Se destaca la acción de `SOAPClient` dado que resulta de utilidad en el desarrollo de este trabajo. Esta acción se basa en la herramienta de software libre SoapUI, que permite realizar invocaciones SOAP utilizando un enfoque de plantillas. Las librerías de SoapUI generan una plantilla basada en el WSDL de la operación requerida, que luego es cargada con los parámetros del Mensaje ESB. La implementación se realiza mediante la clase `org.jboss.soa.esb.actions.soap.SOAPClient`[30]. En la Figura 28 se presenta un ejemplo de configuración de la acción. En la misma la propiedad “`wsdl`” indica el WSDL del servicio que contiene la operación y la propiedad “`SOAPAction`” indica la operación a ejecutar.

```
<action class="org.jboss.soa.esb.actions.soap.SOAPClient" name="Weather">
  <property name="wsdl" value="http://wsf.cdyne.com/WeatherWS/Weather.asmx?WSDL" />
  <property name="SOAPAction" value="http://wsf.cdyne.com/WeatherWS/Weather.asmx/GetWeatherInformation" />
</action>
```

Figura 28 – Configuración de SOAPClient

3.3.2.5. Cliente HTTP

Esta acción reenvía el mensaje como URL para un procesamiento posterior. Es implementada en la clase `org.jboss.soa.esb.actions.routing.http.HttpRouter` en la Figura 29 se presenta un ejemplo de configuración, del cual se destacan:

- `endpointUrl`, indica la URL donde se va a reenviar el mensaje.
- `method`, indica el tipo de petición (GET o POST).
- `content-Type` indica el formato del mensaje.
- `responseType` indica el formato de respuesta.

```
<action name="httprouter" class="org.jboss.soa.esb.actions.routing.http.HttpRouter">
  <property name="endpointUrl" value="http://localhost:8080/rest/tutorial/helloworld"/>
  <property name="method" value="GET" />
  <property name="Content-Type" value="text/xml"/>
  <property name="responseType" value="STRING" />
  <property name="headers">
    <header name="h1" value="value"/>
  </property>
</action>
```

Figura 29 Configuración HttpRouter

3.3.2.6. Smooks

Smooks es una librería que permite realizar diferentes tipos de transformaciones al contenido de un Mensaje basándose en la lógica del patrón *Visitor*. Los tipos de transformación de Mensaje incluyen:

Transformación basada en plantilla, validación de Mensaje, enriquecimiento de Mensaje con información de la base de datos, entre otros.

En particular, la transformación basada en plantillas provee la opción de usar transformaciones en lenguaje XSL o FreeMarker¹², de esta forma permite utilizar distintos lenguajes en el enriquecimiento de la información. En la Figura 30 se presenta un ejemplo de configuración utilizando Smooks, con una plantilla de transformaciones [31]. En ésta configuración se destacan los siguientes atributos:

- `get-payload-location`, indica en qué campo del cuerpo del mensaje se encuentra la información a transformar.
- `set-payload-location`, indica en qué campo del cuerpo del mensaje se devuelve la información transformada.
- `smooksConfig`, indica la ruta al XML que contiene a la plantilla de transformación.

```
<action name="transform" class="org.jboss.soa.esb.smooks.SmooksAction">
  <property name="smooksConfig" value="smooks-res1.xml" />
  <property name="get-payload-location" value="input" />
  <property name="set-payload-location" value="output_transformed" />
</action>
```

Figura 30 - Configuración SmooksAction

Teniendo en cuenta la funcionalidades provistas por Smooks y que tiene licenciamiento LGPL¹³ (asegura el uso libre para todos sus usuarios), se toma la decisión de utilizar esta librería en la implementación de la plataforma.

3.4. Alternativas de arquitecturas evaluadas

En esta sección se plantean dos arquitecturas como solución del sistema a implementar: GeoEEIP Abstracto, GeoEEIP Mixto. Estas surgen como resultado del análisis de la arquitectura del JBossESB y la interacción de componentes identificados en la sección anterior.

Las siguientes subsecciones se dividen en: los objetivos específicos para cada una de las arquitecturas, una breve descripción a alto nivel de cómo sería su funcionamiento, puntos a destacar de las mismas y por último la arquitectura seleccionada.

¹² <http://freemarker.org/>

¹³ <http://www.gnu.org/licenses/lgpl.txt>

3.4.1. GeoEEIP Abstracto

Objetivos

Los objetivos de esta arquitectura son:

- Crear un único artefacto que resuelva un conjunto de problemáticas de integración conocido. Además debe posibilitar ser extendido agregando soluciones a nuevas problemáticas.
- Brindar una solución común para diferentes clientes. De forma que éstos puedan hacer uso del mismo artefacto para las problemáticas específicas de cada uno.

Funcionamiento

En el siguiente diagrama (Figura 31) se presenta la interacción de clientes con la plataforma, mostrando la comunicación entre componentes.

En esta arquitectura cada componente se implementa como un servicio ESB único, los cuales tienen dos etapas de procesamiento. En la primera se encuentra la lógica particular de procesamiento del mensaje y en la segunda la lógica de control de flujo. Esta división corresponde con la aplicación del patrón de mediación Itinerario (ver sección 2.2.5.2).

Del punto anterior se desprende que la manera de determinar el flujo del mensaje es a partir de metadata incluida en el mensaje, determinando así el siguiente servicio o componente a ser procesado. Este comportamiento se muestra en la Figura 31. También se aprecian los ClientEndpoints, que permiten a los clientes interactuar con GeoEEIP. Esto busca publicar una interfaz HTTP, redirigiendo las peticiones. Mediante las peticiones, el GeoEntrypoint puede determinar el cliente y generar la metadata correspondiente para la integración.

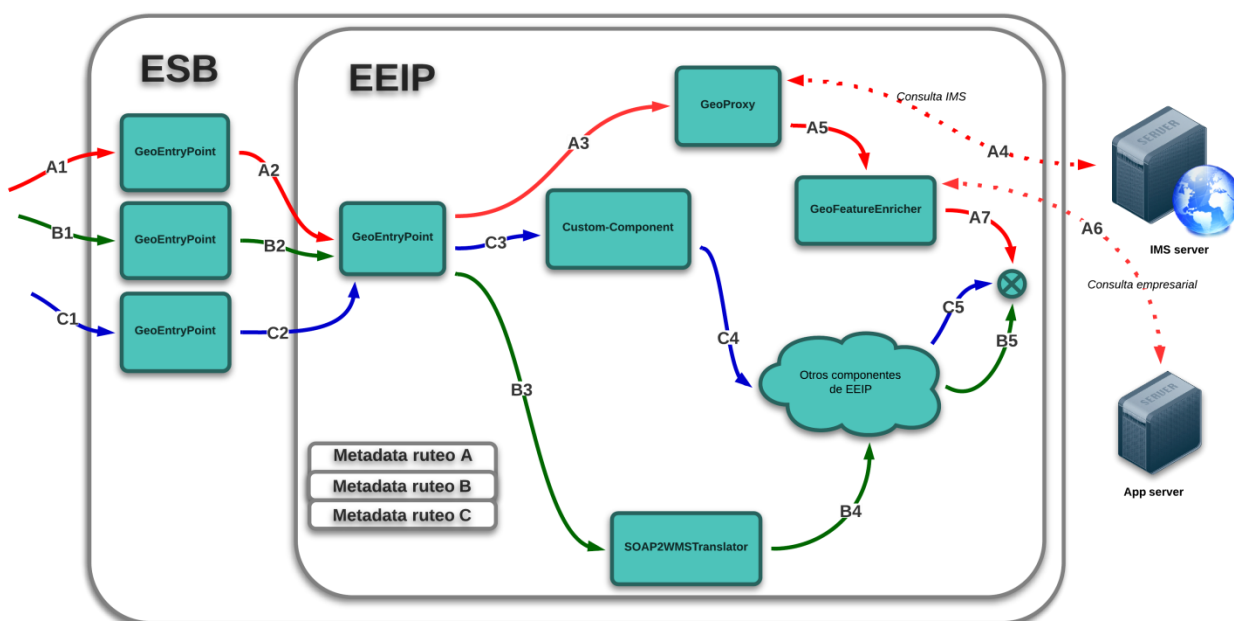


Figura 31 - Diagrama de flujo - GeoEEIP Abstracto

La forma de implementar la extensibilidad, es mediante contratos que definan la estructura de nuevos componentes y herencias para extender la lógica de procesamiento existente. Los nuevos flujos se definen agregando nueva metadata al mensaje y nuevas reglas de ruteo al GeoEEIP, mediante algún mecanismo de configuración.

Puntos a destacar

Algunos puntos a destacar son:

- Se cuenta con un único *deploy* de la plataforma GeoEEIP que atiende a todos los posibles clientes.
- Esta arquitectura delega la responsabilidad al cliente de generar un ClientEndpoint, para officiar de punto de entrada público.
- El mensaje cuenta con dos tipos de datos, los parámetros para la integración y la metada de enrutamiento.
- Tanto una extensión de comportamiento, como un cambio en el ruteo, da lugar a un alto costo en desarrollo. Esto se debe a que impacta en la metada del mensaje y las reglas de ruteo, contemplando los componentes ya en uso.

La Figura 32 presenta el diagrama de diseño de la solución GeoEEIP Abstracto, en el cual se plantean los componentes Básicos, Complejos, nativos de JBossESB y las dependencias entre los mismos. En dicha figura se muestran las dependencias entre los componentes, las cuales se presentan de forma jerárquica. En esta los clientes (end-points) tienen dependencia directa sobre Componentes Complejos y estos a su vez sobre los Componentes Básicos y nativos de JBossESB. Por último los Componentes Básicos tienen dependencia solamente sobre los componentes nativos de JBossESB. Las dependencias entre componentes de la jerarquía se da por composición, a modo de ejemplo los Componentes Complejos están compuestos por Componentes Básicos.

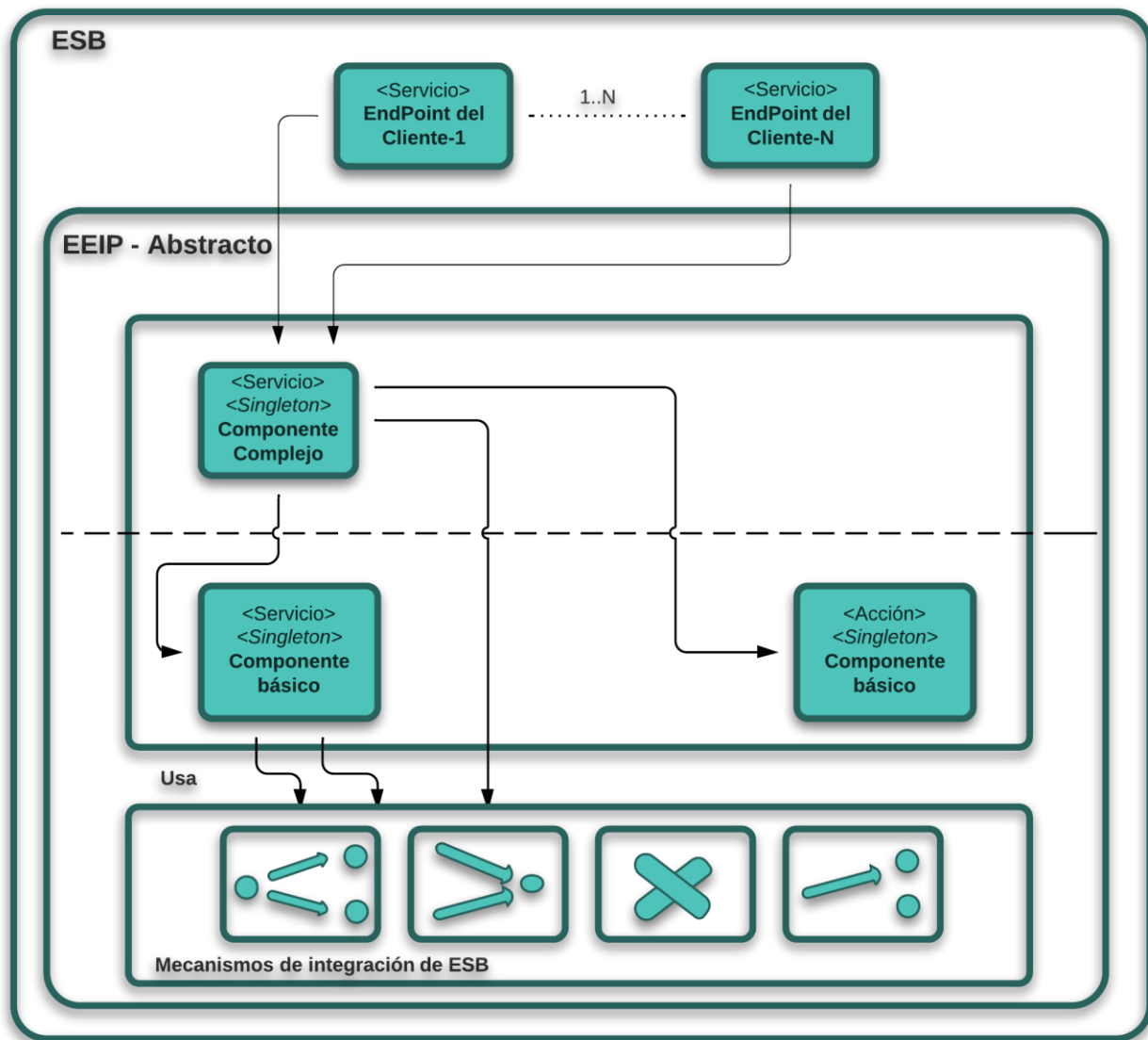


Figura 32 - Diagrama de diseño - GeoEEIP Abstracto

3.4.2. GeoEEIP Mixto

Objetivos

Los objetivos de esta arquitectura son:

- Crear una solución que genere artefactos para cada problemática de integración específica de un cliente.
- El conjunto de componentes de esta arquitectura debe ser extensible, posibilitando generar soluciones a medida utilizando nuevos componentes, la reutilización de los ya existentes o extensiones de los mismos.
- Generar los Componentes Básicos como acciones de JBossESB, para permitir la utilización de los mismos de manera flexible, reduciendo las interacciones entre elementos de ESB (acciones o servicios).

Funcionamiento

En esta arquitectura los Componentes Básicos se implementan como acciones individuales siempre que su lógica lo permita y como servicio en caso contrario. Los Componentes Complejos se implementan como servicios que contienen los Componentes Básicos y las acciones o servicios de JBossESB.

La lógica de flujo es particular de cada componente y es implementada con las herramientas provistas por JBossESB. Además es definida estáticamente en la configuración de cada componente.

De esta manera, como se aprecia en la Figura 33, cada cliente puede contar con su propia solución para una problemática de integración. Ya que genera un artefacto con la configuración y lógica para su escenario específico.

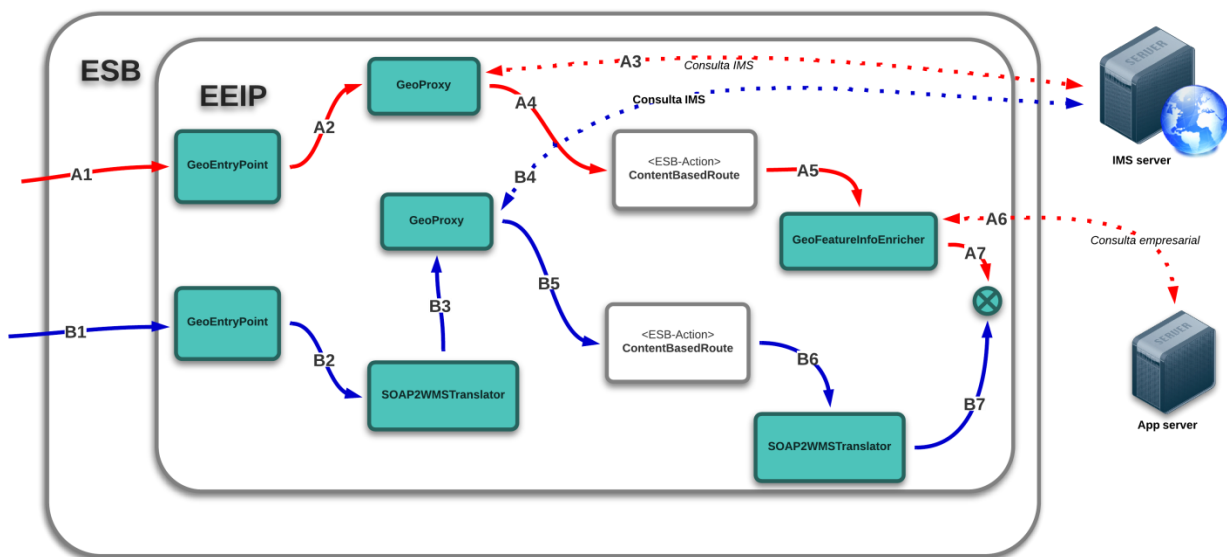


Figura 33 - Diagrama de flujo - GeoEEIP Mixto

Puntos a destacar

Algunos puntos a destacar son:

- Esta arquitectura es extensible. Esto se debe a que se pueden generar nuevos componentes, que cubran nuevas problemáticas, sin tener impacto sobre la plataforma (componentes existentes).
- Los artefactos generados son adaptables, ya que se puede modificar la composición de los elementos del mismo. Esto se realiza mediante la utilización de los provistos por GeoEEIP o generando nuevos.

La Figura 34 presenta el diagrama de diseño de la solución GeoEEIP Mixto, en el cual se plantean todos los Componentes Básicos, los Complejos y los nativos de JBossESB. Además se definen las fronteras de GeoEEIP y el ESB. La dependencia al igual que el GeoEEIP Abstracto, se da de forma jerárquica y por composición. Los Componentes Complejos están compuestos por Componentes Básicos y nativos de JBossESB y a su vez los Componentes Básicos están compuestos por componentes nativos de JBossESB.

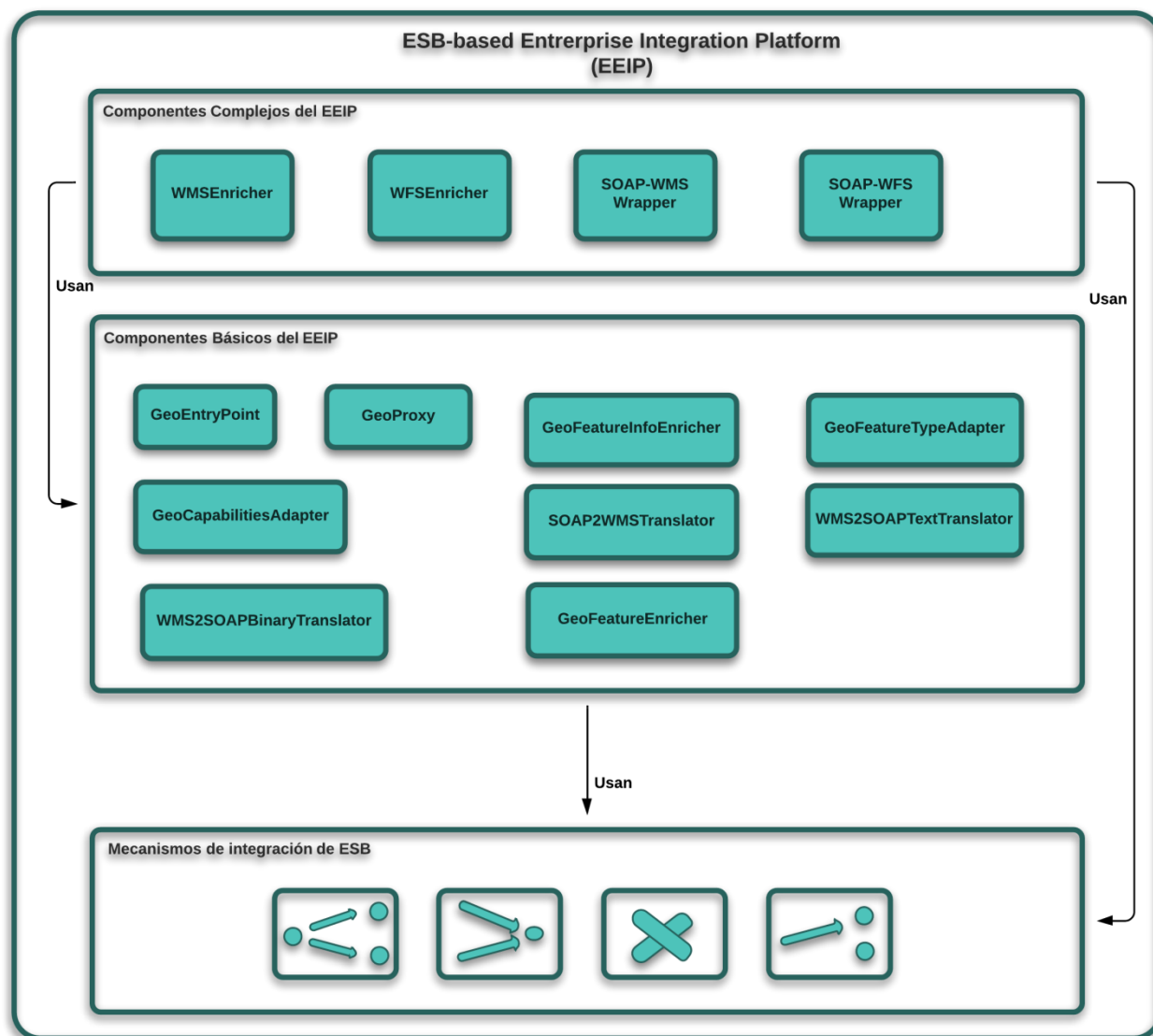


Figura 34 - Diagrama de diseño - GeoEEIP Mixto

3.4.3. Selección de la arquitectura

A partir de los puntos destacados de cada una de las arquitecturas, se obtiene la siguiente tabla comparativa.

	GeoEEIP Abstracto	GeoEEIP Mixto
Deploy	Único <i>deploy</i>	Multiple <i>deploy</i>
Extensibilidad	Alto costo en cambio de lógica impacta metadata y formato del mensaje	Bajo costo en cambio de lógica (cambio del componente)
Acoplamiento	Se necesita un ClientEndpoint desarrollado por el cliente.	No se necesita ningún desarrollo por parte del cliente.

Tabla 1 - Comparativa de arquitecturas

Luego de este análisis y una reunión con el usuario de la plataforma, se llegó a la conclusión de que la arquitectura a implementar fuera la de GeoEEIP Mixto. Principalmente debido a que el usuario requiere menos conocimiento técnico de la implementación de la plataforma para su utilización.

3.5. Aplicación de Configuración

En esta sección se realiza un análisis de las interfaces de usuario necesarias para la generación de los artefactos para el *deploy* en JBossESB, o de los archivos de configuración de un componente básico específico. Dichas interfaces estarán contenidas en una aplicación web que permita al desarrollador generar dichos artefactos de forma sencilla y abstrayéndolo de conocimientos avanzados de ESB.

Interfaz de configuración de Componente Básico

Interfaz de configuración que permita generar los archivos de configuración de ESB, de modo de facilitar el desarrollo. Mediante esta interfaz el usuario puede obtener el XML de configuración de JBossESB ingresando los valores de los parámetros de configuración. Y de esta manera utilizar la configuración obtenida en el desarrollo de nuevos componentes de integración.

Interfaz de configuración de Componente Complejo

Interfaz de configuración que permita generar los archivos para el *deploy* sobre JBossESB correspondientes a los Componentes Complejos. Mediante esta interfaz el usuario puede obtener los archivos de instalación de un Componente Complejo mediante un formulario que contiene los parámetros de configuración necesarios.

4. Solución implementada

En el presente capítulo se describe la solución implementada, la cual corresponde con la solución GeoEEIP Mixto vista en la sección 3.4.2. En la Sección 4.1 se describe la arquitectura utilizada y las decisiones de diseño en el marco de JBossESB, luego en las secciones 4.2 y 4.3 se presentan las decisiones de diseño e implementación de los Componentes Básicos y Complejos más interesantes por su complejidad. Por último, en la sección 4.4 se presenta la aplicación Web de configuración, indicando sus funcionalidades y la interacción para generar el archivo para el *deploy* en JBoss.

4.1. Arquitectura de alto nivel

La forma típica en que JBossESB presenta sus patrones de mediación es encapsulándolos en servicios, los cuales son configurados en el archivo `jboss-esb.xml`. En este archivo, se definen las propiedades de cada servicio, dando lugar a una estructura que permite configurar para cada servicio un pipeline de acciones a ser ejecutado, siendo éste el flujo de procesamiento.

Teniendo como objetivo que la funcionalidad de cada Componente Básico de GeoEEIP estuviese lo más encapsulada posible y que pueda ser reutilizada en el contexto de otro flujo dentro de JBossESB, se decidió implementar los Componentes Básicos como acciones. Esta decisión permite también que cada Componente definido en la solución teórica presentada en “Towards an ESB-Based Enterprise Integration Platform for Geospatial Web Services”[1] se refleje con mayor fidelidad en cada componente implementado. Existen excepciones donde no es posible implementar un componente como una acción, debido a características de JBossESB y de la funcionalidad a implementar. Por ejemplo JBossESB requiere que los destinos posibles de un ruteo sean servicios, en estos casos un componente a utilizar como destino se diseña como servicio.

La manera de desarrollar un componente como acción implica crear una clase que herede de `AbstractActionPipelineProcessor` (ver 3.4) o extender de una específica de JBossESB (la cual a su vez hereda de ésta). En ambas se puede desarrollar la lógica necesaria para el componente, que a su vez puede instanciar una o más acciones que cuenten con esta posibilidad. En la Figura 35 se aprecia la herencia descrita.

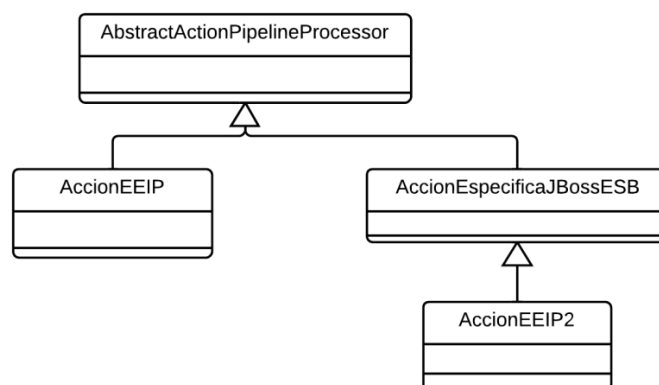


Figura 35 - Diagrama de implementación de acciones

En la Figura 36 se muestra la arquitectura de alto nivel, la cual presenta cada componente de la plataforma, además se demuestra a modo de ejemplo (por medio de las líneas punteadas) cómo el componente WMSEnricher se relaciona con los componentes que utiliza.

El estilo arquitectónico que utiliza la plataforma GeoEEIP, está definido por el de JBossESB. De esta manera se respetan la composición e interacción de componentes, necesarios para integrar la plataforma al ESB.

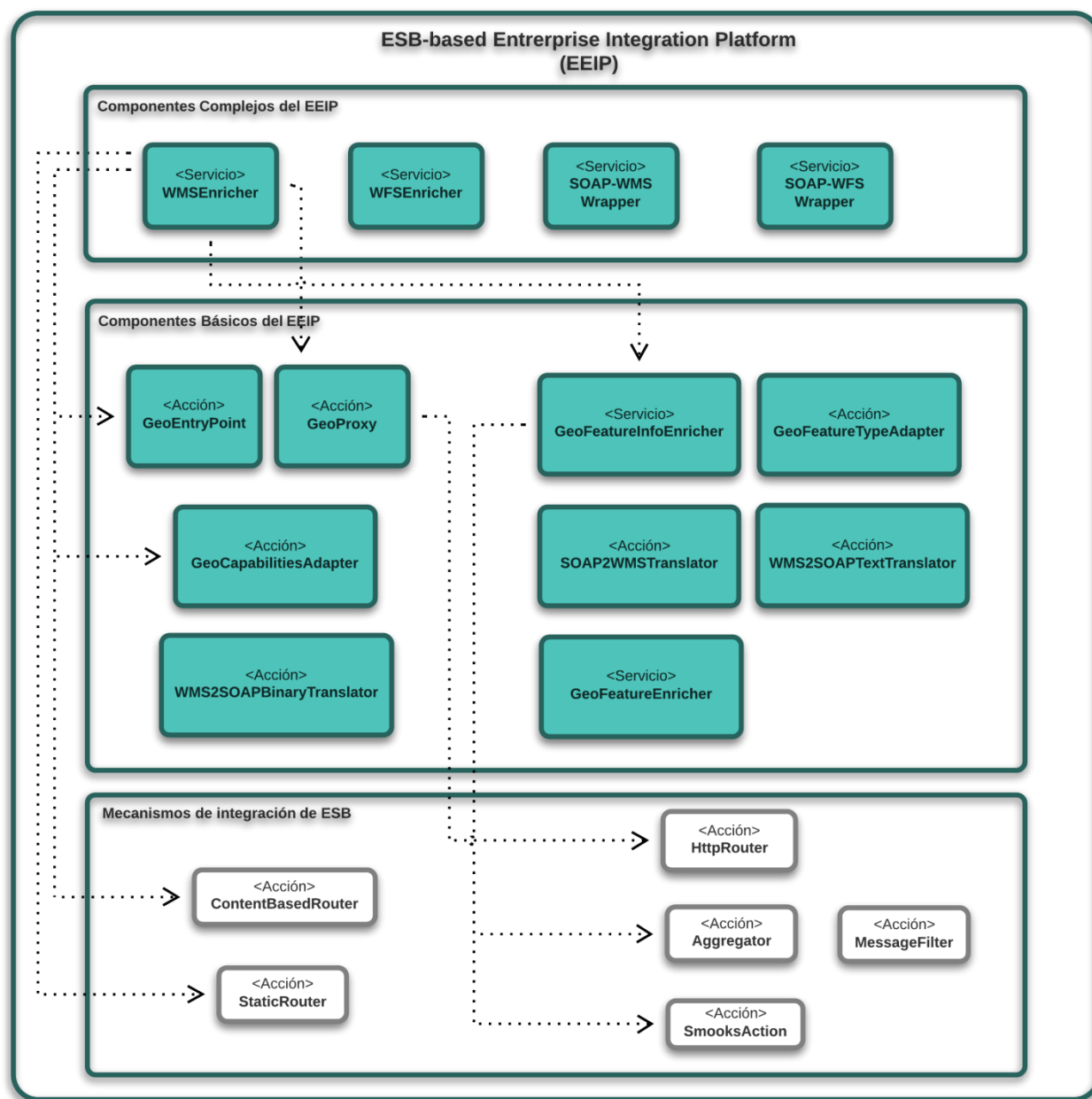


Figura 36 - Arquitectura - GeoEEIP

4.2. Componentes básicos

A continuación se analizan desde el punto de vista técnico y de implementación, los Componentes Básicos vistos en la sección 3.2.1. En el Apéndice 2 se realiza este análisis para el resto de los Componentes Básicos. Como parte de dicho análisis, para cada uno de los componentes se presentan las precondiciones y las post condiciones (primera tabla) para la invocación al componente. Adicionalmente se presenta una segunda tabla en la cual se especifican: el formato de entrada para el mensaje ESB en términos de sus parámetros, la configuración del componente en la cual se especifican y describen los atributos definidos y, por último, se muestra el formato del mensaje ESB de salida en términos de sus parámetros.

GeoEntryPoint

Componente que recibe un mensaje ESB conteniendo la petición HTTP recibida por GeoEEIP. Luego se encarga de guardar en el mensaje los siguientes datos: la dirección del servicio (LOCAL_ADDRESS), los parámetros de la consulta como texto (QUERY_STRING¹⁴) y un diccionario conteniendo dichos parámetros(QUERY_PARAMS).

La Figura 37 muestra el diagrama de clases para el GeoEntryPoint, en el cual se ve la herencia de AbstractActionPipelineProcessor (componente de JBossESB). Adicionalmente se especifican los atributos de entrada/salida.

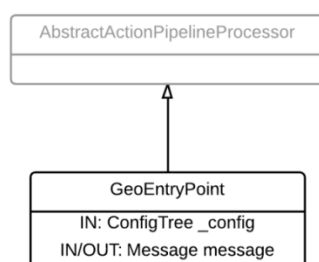


Figura 37 - Diagrama de clases – GeoEntryPoint (Rest)

Condiciones

Pre	Post
Petición WMS/WFS	Se genera un mensaje con la información recibida en el cuerpo del mensaje. Entre estos datos se identifican: URL, nombre de la operación WMS/WFS y parámetros de consulta.

Parámetros Requeridos

Nombre de parámetro	Descripción
Formato de mensaje de entrada	
<i>HTTP_REQUEST</i>	Petición recibida por el HTTPEntryPoint de ESB para el caso de una petición WMS/WFS REST.
Configuración del componente	
Formato de mensaje de salida	
<i>QUERY_STRING</i>	Fragmento de la petición IMS REST original con los parámetros de consulta.
<i>QUERY_PARAMS</i>	Diccionario de parámetros generado a partir del parámetro QUERY_STRING.

Tabla 2 - GeoEntryPoint REST Pre-Post condiciones y Parámetros

¹⁴ QueryString es un segmento de una URI, su definición se encuentra en <http://tools.ietf.org/html/rfc3986#section-3>

GeoEntryPointSOAP

Componente que recibe un Mensaje ESB conteniendo la petición SOAP recibida por la plataforma. Luego se encarga de guardar en el Mensaje el cuerpo del mensaje SOAP.

La Figura 38 muestra el diagrama de clases para el GeoEntryPointSOAP, en el cual se ve la herencia de AbstractActionPipelineProcessor (componente de JBossESB). Adicionalmente se especifican los atributos de entrada/salida.

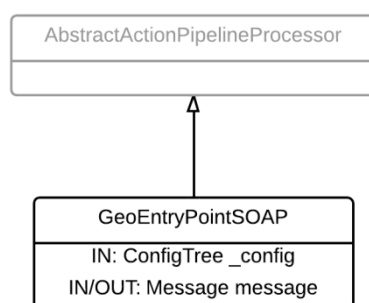


Figura 38 - Diagrama de clases - GeoEntryPoint (SOAP)

Condiciones

Pre	Post
Petición SOAP con información para invocar operación WMS	Se genera un mensaje ESB con el cuerpo del mensaje SOAP.

Parámetros Requeridos

Nombre de parámetro	Descripción
Formato de mensaje de entrada	
<i>SOAP_REQUEST</i>	Petición recibida por el HTTPEntryPoint de ESB para el caso de una petición SOAP.
Configuración del componente	
Formato de mensaje de salida	
<i>SOAP_MESSAGE</i>	Mensaje SOAP recibido por el HTTPEntryPoint.

Tabla 3 - GeoEntryPoint SOAP Pre-Post condiciones y Parámetros

En la Figura 39 se muestra un ejemplo de petición SOAP para WMS, recibida por el GeoEntryPointSOAP. En la misma se puede apreciar el cuerpo del mensaje SOAP, respetando la definición obtenida del estudio “OWS 1.2 SOAP Experiment Report”[29] de la OGC en el caso del método GetMap.

```

<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:eeip="http://www.fing.edu.uy/eeip" xmlns:sld="http://www.opengis.net/sld">
  <soapenv:Header/>
  <soapenv:Body>
    <eeip:GetMap>
      <eeip:version>1.1.1</eeip:version>
      <eeip:service>vms</eeip:service>
      <eeip:exceptions>application_vnd_ogc_se_xml</eeip:exceptions>
      <eeip:Map>
        <eeip:BoundingBox decimal="." cs="," ts=" " >574444.519082,6135153.872524,578556.052348,6137548.758795</eeip:BoundingBox>
        <eeip:Crs>EPSG:32721</eeip:Crs>
      </eeip:Map>
      <eeip:Image>
        <eeip:Height>330</eeip:Height>
        <eeip:Width>684</eeip:Width>
        <!--<eeip:Format>application/openlayers</eeip:Format>-->
        <eeip:Format>image/png</eeip:Format>
        <!--Optional:-->
        <eeip:Transparent>false</eeip:Transparent>
        <!--Optional:-->
        <eeip:BGColor>0xFFFFFFFF</eeip:BGColor>
      </eeip:Image>
      <sld:StyledLayerDescriptor>
        <sld:NamedLayer>
          <se:Name>eeip:manzanas</se:Name>
        </sld:NamedLayer>
      </sld:StyledLayerDescriptor>
    </eeip:GetMap>
  </soapenv:Body>
</soapenv:Envelope>

```

Figura 39 - Petición SOAP al GeoEEIP

GeoProxy

Componente que tiene como responsabilidad crear la URL de consulta al IMS a partir de la propiedad QUERY_STRING del Mensaje ESB y el dominio que se obtiene de la propiedad IMS_NAME proveniente de la configuración.

Luego se instancia la acción de ESB HttpRouter, para la cual se configuran las siguientes propiedades: "endpointUrl" conteniendo el valor de la URL obtenida y "method" con valor "GET", que especifica el método HTTP a utilizar. Luego de esto se invoca su método de procesamiento, el cual realiza la consulta al IMS y retorna la respuesta para ser almacenada en el mensaje original, en la propiedad IMS_RESPONSE. Adicionalmente se almacena en el mensaje ESB la URL del IMS invocada.

La Figura 40 muestra el diagrama de clases para el componente GeoProxy, en el cual se refleja la herencia de AbstractActionPipelineProcessor y la dependencia con HTTPRouter (componentes de JBossESB). Adicionalmente se especifican los atributos de entrada/salida del componente.

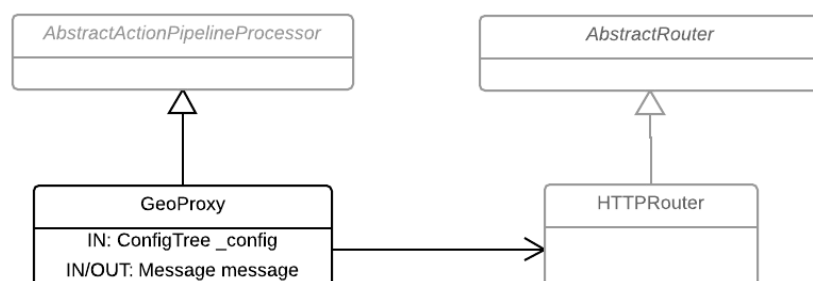


Figura 40 - Diagrama de clases – GeoProxy

Condiciones

Pre	Post
Mensaje ESB conteniendo la información de petición WMS/WFS.	Se agrega al mensaje ESB el retorno de la operación IMS definida por la petición y la configuración del componente.
Componente configurado con información necesaria del IMS a invocar.	

Parámetros Requeridos

Nombre de parámetro	Descripción
Formato de mensaje de entrada	
<i>QUERY_STRING</i>	Fragmento de la petición IMS REST original con los parámetros de consulta.
<i>QUERY_PARAMS</i>	Diccionario de parámetros generado a partir del parámetro <i>QUERY_STRING</i> .
Configuración del componente	
<i>IMS_NAME</i>	URL base del servicio IMS a consultar.
Formato de mensaje de salida	
<i>IMS_ADDRESS</i>	URL base del servicio IMS consultado.
<i>IMS_RESPONSE</i>	Respuesta de la consulta IMS en el formato solicitado.

Tabla 4 - GeoProxy Pre-Post condiciones y Parámetros

GeoRouter

Componente encargado de direccionar el Mensaje ESB a partir de la petición WFS o WMS a un Servicio ESB configurado.

En la implementación de este componente, se utilizó la acción *ContentBasedRouter* nativa de JBossESB (ver sección 3.3.2.3). El Servicio ESB a direccionar se obtiene de aplicar expresiones regulares a la propiedad del Mensaje ESB *QUERY_STRING*. La acción *ContentBasedRouter* tiene como restricción que el destino del Mensaje debe ser un Servicio ESB, por lo que no se puede agregar acciones luego de esta acción, condicionando los componentes a utilizar luego de éste.

Condiciones

Pre	Post
Mensaje ESB conteniendo la operación del IMS en el parámetro <i>QUERY_STRING</i>	Se direcciona el Mensaje ESB hacia el Servicio ESB correspondiente a la operación WMS/WFS.

Tabla 5 - GeoRouter Pre-Post condiciones

GeoFeatureInfoEnricher

Es el componente encargado de obtener los datos empresariales y luego enriquecer con éstos la información del IMS almacenada previamente en el mensaje ESB (mediante transformaciones XSLT). Este componente se implementa como un servicio, el cual está compuesto por otros servicios como se aprecia en la Figura 41. En dicha Figura se presentan tres etapas de procesamiento: en primera instancia un servicio que oficia de punto de entrada (*GeoFeatureInfoEnricher*) para luego direccionar a los servicios empresariales mediante ruteo estático. Estos servicios que componen la segunda etapa son los encargados de consultar la fuente empresarial correspondiente para luego dar paso a la tercera etapa. Esta última etapa es la más compleja debido a que involucra una mayor cantidad de acciones y es la encargada de ejecutar las transformaciones de enriquecimiento. En primer lugar se obtienen los datos empresariales a través de la acción de JBossESB *Aggregator*, que luego de haber recibido todas las respuestas esperadas da paso a la siguiente acción (*GeoFeatureInfoAggregator*). Esta acción se encarga de seleccionar una de las respuestas desde la colección obtenida anteriormente, para luego generar un mensaje conteniendo los datos empresariales y los datos georreferenciados, finalizando con la ejecución de la transformación XSLT configurada en la acción que ejecuta *SmooksAction* (ver sección 3.3.2.6). Esto se realiza por cada fuente empresarial consultada.

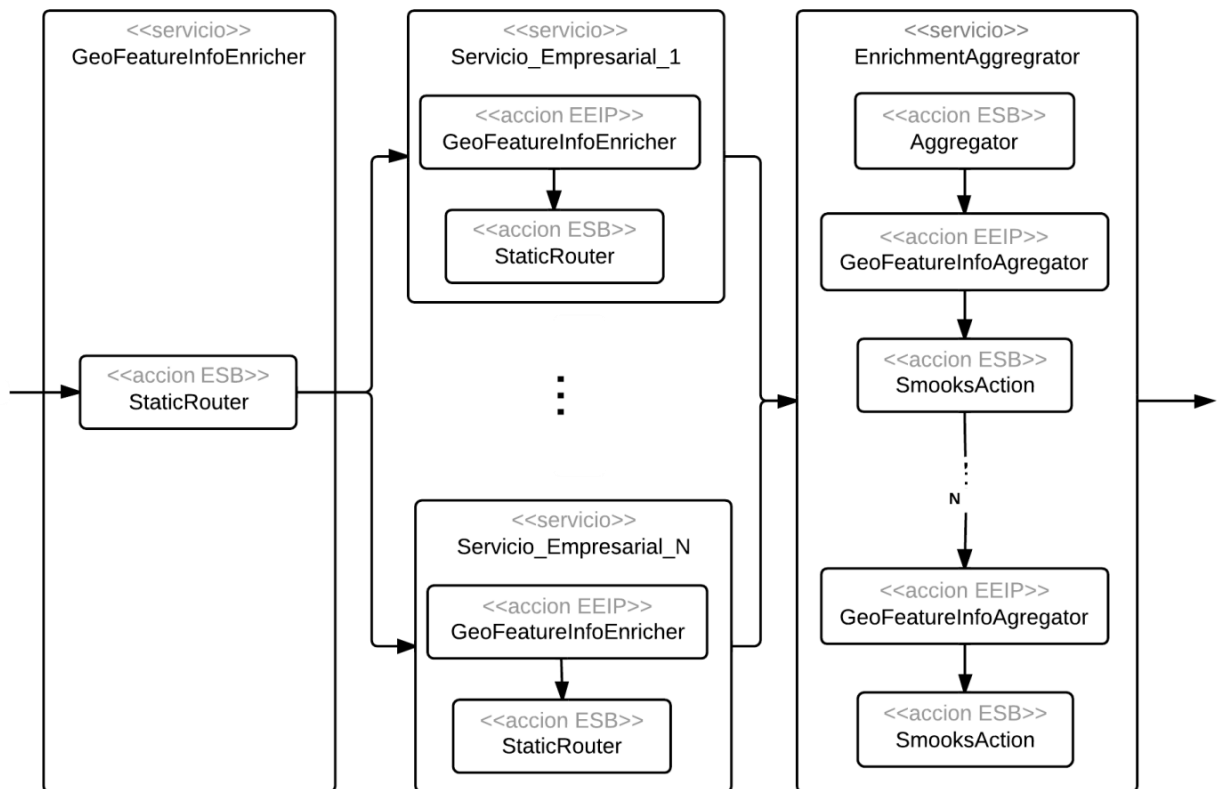


Figura 41 - Diagrama de flujo GeoFeatureInfoEnricher

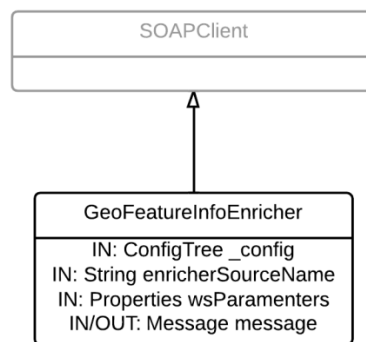


Figura 42 – GeoFeatureInfoEnricher

Condiciones

Pre	Post
Mensaje ESB que contiene el retorno de la operación GetFeatureInfo de un WMS.	Retorna el Mensaje ESB con la respuesta de la operación GetFeatureInfo enriquecido con los datos obtenidos de las fuentes externas mediante transformaciones XSLT.

Parámetros Requeridos

Nombre de parámetro	Descripción
Formato de mensaje de entrada	
<i>IMS_RESPONSE</i>	Respuesta de la consulta IMS en formato XML.
Configuración Fuente Empresarial	
<i>WSDL_URL(*)</i>	URL al WSDL de Web Services donde están publicadas las operaciones que se utilizan en el enriquecimiento.
<i>SOAP_OPT(*)</i>	Operación que retorna los datos empresariales a utilizar en el enriquecimiento.
<i>smooksConfig(*)</i>	Ruta al archivo que especifica la transformación que realiza el enriquecimiento.
Formato de mensaje de salida	
<i>IMS_RESPONSE</i>	Respuesta de la consulta IMS enriquecida en formato XML.

Tabla 6 - GeoFeatureInfoEnricher Pre-Post condiciones y parámetros

Como se aprecia en la Figura 41 este componente tendrá tantos Servicios Empresariales y enriquecimientos, como fuentes de datos a integrar con la respuesta IMS. Esto quiere decir que la configuración relacionada con estos pasos, destacada con (*), se repite por cada una de las Fuentes Empresariales a consultar.

SOAP2WMSTranslator

Este componente se encarga de transformar la petición SOAP que se encuentra en el mensaje ESB a una petición WMS, para luego guardarla en el mensaje.

La Figura 43 muestra el diagrama de clases para el SOAP2WMSTranslator, en el cual se ve la herencia de AbstractActionPipelineProcessor (componente de JBossESB). Adicionalmente se especifican los atributos de entrada/salida.

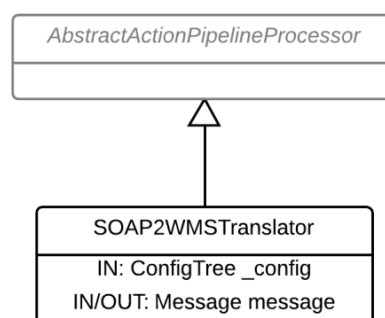


Figura 43 - WMS2SOAPTextTranslator

Condiciones

Pre	Post
Mensaje ESB que contiene una petición SOAP en su cuerpo.	Se genera un Mensaje ESB con la información obtenida a partir de la petición SOAP.

Parámetros Requeridos

Nombre de parámetro	Descripción
Formato de mensaje de entrada	
<i>SOAP_MESSAGE</i>	Petición SOAP con el formato definido en WSDL provisto por GeoEEIP para WMS.
Configuración del componente	
Formato de mensaje de salida	
<i>QUERY_STRING</i>	Fragmento de la petición IMS REST original con los parámetros de consulta.

Tabla 7 - SOAP2WMSTranslator Pre-Post condiciones y parámetros

4.3. Componentes complejos

A continuación se analizan desde el punto de vista técnico y su implementación, los Componentes Complejos vistos en la sección 3.2.2, el análisis del resto de los Componentes Básicos se puede ver en el Apéndice 2. Como parte de dicho análisis, para cada uno de los componentes se presentan las precondiciones y las post condiciones (primera tabla) para la invocación al componente. Adicionalmente se presenta una segunda tabla en la cual se especifican: el formato de entrada para el mensaje ESB en términos de sus parámetros, la configuración del componente en la cual se especifican y describen los atributos definidos y por último se muestra el formato del mensaje ESB de salida en términos de sus parámetros.

WMS Enricher

Este componente permite tener acceso a un IMS para obtener datos georreferenciados y a la vez enriquecer los mismos con datos provenientes de una o más fuentes empresariales. En la Figura 44 se muestran los Componentes Básicos que conforman el WMS Enricher. Los componentes GeoEntryPoint, GeoProxy y GeoRouter (vistos en la sección 4.2) se implementan en un mismo servicio. Luego, el GeoRouter direcciona hacia el componente correspondiente de acuerdo a la operación invocada por el usuario. En el caso de GetCapabilities se direcciona al componente GeoCapabilitiesAdapter¹⁵. Si la operación es GetFeatureInfo se direcciona al componente GeoFeatureInfoEnricher¹⁶, donde se enriquecen los datos georreferenciados según la configuración especificada. Y por último en el caso de que la operación solicitada sea GetMap, los datos obtenidos del IMS son retornados directamente.

¹⁵ Ver Apéndice 1.

¹⁶ En la figura se muestra el componente como un único servicio a manera de simplificar el esquema, por más detalles ver Sección 4.2.

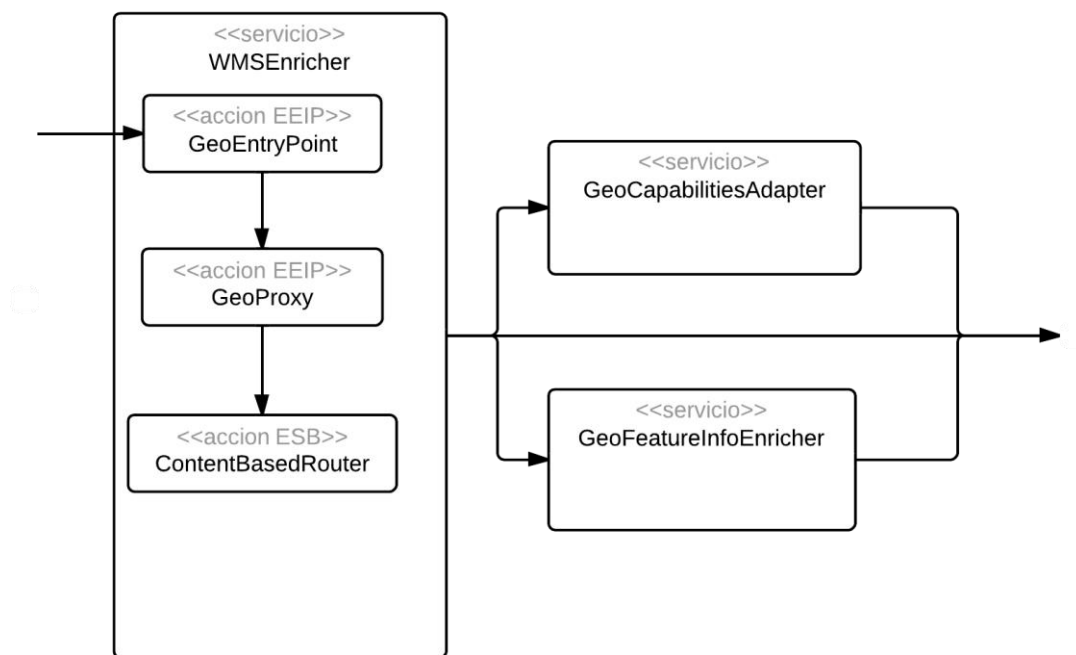


Figura 44 - Diagrama WMS Enricher

Condiciones

Pre	Post
Petición WMS	Petición WMS, conteniendo la información solicitada enriquecida según el caso.

Parámetros Requeridos

Nombre de parámetro	Descripción
Formato de mensaje de entrada	
<i>HTTP_REQUEST</i>	Consulta recibida por el GeoEntryPoint.
Configuración Fuente Empresarial	
<i>IMS_NAME</i>	URL base del servicio IMS a consultar.
<i>WSDL_URL(*)</i>	URL al WSDL de Web Services donde están publicadas las operaciones que se utilizan en el enriquecimiento.
<i>SOAP_OPT(*)</i>	Operación que se utiliza para el enriquecimiento.
<i>smooksConfig(*)</i>	Ruta al archivo que especifica la transformación que realiza el enriquecimiento.
Formato de mensaje de salida	
<i>HTTP_RESPONSE</i>	Respuesta IMS enriquecida.

Tabla 8 - WMSEnricher Pre-Post condiciones y parámetros

Nota: La configuración destacada con (*), se repite por cada una de las Fuentes Empresariales a consultar (ver WMSFeatureInfoEnricher, sección 4.2).

WFS Enricher

Este componente es similar al anterior, se diferencia en alguno de los componentes utilizados luego del GeoRouter, ya que se utilizan los correspondientes a las operaciones provistas por el protocolo WFS.

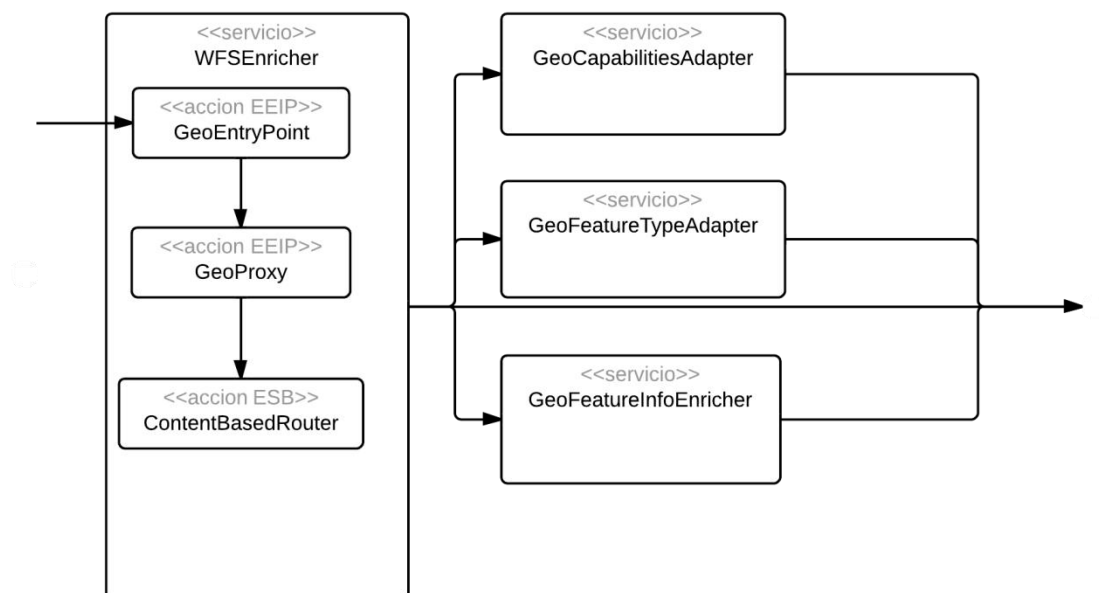


Figura 45 - Diagrama WFS Enricher

Condiciones

Pre	Post
Petición WFS	Respuesta WFS, conteniendo la información solicitada enriquecida según el caso.

Parámetros Requeridos

Nombre de parámetro	Descripción
Formato de mensaje de entrada	
<i>HTTP_REQUEST</i>	Consulta WFS recibida por el GeoEntryPoint.
Configuración Fuente Empresarial	
<i>IMS_NAME</i>	URL base del servicio IMS a consultar.
<i>WSDL_URL(*)</i>	URL al WSDL de Web Services donde están publicadas las operaciones que se utilizan en el enriquecimiento.
<i>SOAP_OPT(*)</i>	Operación que se utiliza para el enriquecimiento.
<i>smooksConfig(*)</i>	Ruta al archivo que especifica la transformación que realiza el enriquecimiento.
Formato de mensaje de salida	
<i>HTTP_RESPONSE</i>	Respuesta IMS enriquecida.

Tabla 9 - WFSEnricher Pre-Post condiciones y parámetros

Nota: La configuración destacada con (*), se repite por cada una de las Fuentes Empresariales a consultar (ver WFSFeatureInfoEnricher, Apéndice 1).

SOAP-WMS Wrapper

Este componente brinda la posibilidad de acceso a un servidor WMS mediante el protocolo SOAP, utilizando la definición WSDL planteada en el estudio “OWS 1.2 SOAP Experiment Report” [29]. El componente se divide en dos pasos como muestra la Figura 46. El primer paso es el encargado de publicar el Web Service SOAP con el WSDL anteriormente mencionado mediante la acción “ContractPublisher”. Luego se utiliza la acción de ESB “MessageFilter” para realizar la redirección al servicio correspondiente a la operación solicitada en el mensaje SOAP recibido. En el segundo paso los servicios para cada operación son similares, primero cuenta con tres componentes ya vistos en la sección anterior: una acción “GeoEntryPoint” que obtiene la petición recibida, una acción “SOAP2WMSTranslator” encargada de traducir la petición SOAP a una WMS para luego en la siguiente acción “GeoProxy” obtener los datos georreferenciados. Por último cuenta con una acción encargada de generar la respuesta SOAP con los datos obtenidos con el formato acorde para cada operación WMS. Es aquí que se puede apreciar la mayor diferencia entre los tres servicios representados, ya que para el caso de GetCapabilities y GetFeature se utiliza una acción “WMS2SOAPTextTranslator” y para el retorno de imágenes resultado de la operación GetMap se utiliza la acción “WMS2SOAPBinaryTranslator”.

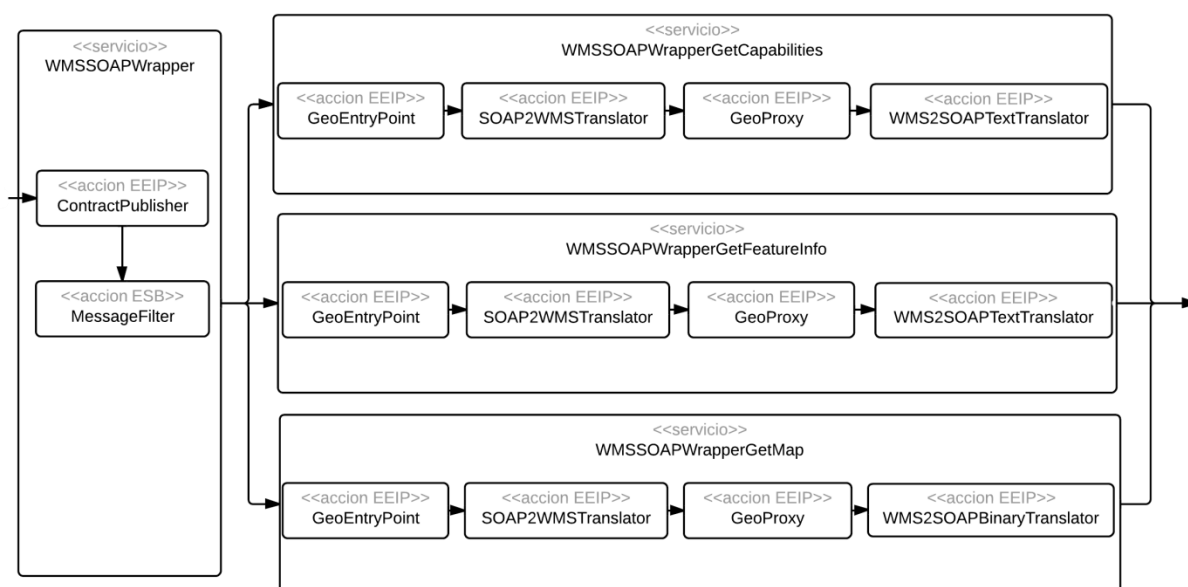


Figura 46 - Diagrama SOAP Wrapper

Condiciones

Pre	Post
Petición SOAP con información para invocar operación IMS	Respuesta SOAP, conteniendo en su cuerpo la respuesta WMS.

Parámetros Requeridos

Nombre de parámetro	Descripción
Formato de mensaje de entrada	
<i>SOAP_REQUEST</i>	Consulta SOAP recibida por el GeoEntryPoint.
Configuración del componente	
<i>IMS_NAME</i>	URL base del servicio IMS a consultar.
Formato de mensaje de salida	
<i>SOAP_RESPONSE</i>	Respuesta SOAP.

Tabla 10 – SOAP-WMS Wrapper Pre-Post condiciones y parámetros

4.4. Aplicación de configuración

Se desarrolló una aplicación Web, mediante la cual facilitarle al usuario la utilización de la plataforma GeoEEIP. Esta cuenta con dos funcionalidades principales, en primer lugar el usuario puede especificar la configuración de cada componente básico a través de los formularios disponibles y obtener el XML correspondiente a dicha configuración para ser utilizado en JBossESB. En segundo lugar el usuario puede configurar y obtener un archivo para el *deploy* de cualquiera de los Componentes Complejos desarrollados. El archivo de *deploy* contiene todo (clases, archivos de configuración, etc.) lo necesario para que utilizar el componente.

Configuración de Componentes Básicos

La configuración de los Componentes Básicos se realiza en formularios acordes con los parámetros que cuenta cada uno. En la Figura 47 se aprecia un ejemplo de estos formularios. El formulario corresponde a la configuración del componente GeoProxy, el cual tiene como parámetro el campo “imsName” en el que su valor es la ruta al IMS a consultar (ver sección 4.2). Luego de especificar los valores deseados para cada campo y presionar el botón “Generar componente” el usuario obtendrá el XML correspondiente a la configuración de JBossESB para dicho componente (esto se ve en la Figura 48).

Configurador EEIP

Componentes	Componente GeoProxy
-----Simple----- GeoProxy GeoRouter GeoCapabilitiesAdapter GeoFeatureTypeAdapter GeoFeatureInfoEnricher	URL IMS: Generar componente
-----Complejo----- GeoEnricher WMS SOAP Wrapper	

Figura 47 - Formulario configuración GeoProxy

Configurador EEIP

Componentes	Componente GeoProxy
-----Simple----- GeoProxy GeoRouter GeoCapabilitiesAdapter GeoFeatureTypeAdapter GeoFeatureInfoEnricher	URL IMS: http://localhost:8081/geo: Generar componente
-----Complejo----- GeoEnricher WMS SOAP Wrapper	<pre><action class="eeip.mechanisms.basic.GeoProxy" name="GeoProxy"><property name="imsName" value="http://localhost:8081/geoserver/wms"/></action></pre>

Figura 48- Formulario configuración GeoProxy resultado

Configuración de Componentes Complejos

La configuración de los Componentes Complejos se realiza mediante formularios similares a los utilizados para los Componentes Básicos. La principal diferencia se encuentra en el resultado obtenido, ya que el usuario obtiene un archivo para el *deploy*, el cual contiene todos los archivos binarios y de configuración necesarios para que el componente ejecute sobre JBossESB.

A continuación se detalla la configuración del componente GeoEnricher ya que para este se definieron archivos de configuración para alguno de sus componentes y para la generación de archivos necesarios en su funcionamiento. Por ejemplo el archivo de configuración de mapeo de datos para la generación de las transformaciones XSLT necesarias para el enriquecimiento de los datos.

En la Figura 49 se aprecia el formulario de configuración del componente GeoEnricher, el cual se divide en dos partes: parámetros generales del componente y parámetros de fuentes de datos a utilizar. Esta división existe ya que el usuario tiene la posibilidad de configurar tantas fuentes de datos como desee, desde las cuales obtener la información para enriquecer los datos georreferenciados.

Configuración básica:

- Nombre – Nombre que identifica el componente dentro del contexto de JBossESB.
- Descripción – Descripción del componente dentro del contexto de JBossESB.
- URL IMS – URL del IMS a consultar.
- Protocolo – Tipo de protocolo (WMS o WFS).

Configuración fuente de datos:

- Nombre – Nombre que identifica a la fuente de datos internamente en el contexto de JBossESB.
- WSDL – URL de publicación del archivo WSDL que identifica el Web Service a consultar.
- Método SOAP – Nombre del método del Web Service a invocar.
- Esquema del Tipo de Datos – Nombre del archivo XSD correspondiente al tipo de dato con el que se enriquecerán los datos geo referenciados.
- Configuración Xpath para Tipo de Datos – Nombre de archivo de configuración, el contenido de este archivo es una pareja de rutas XPATH, donde el primer registro corresponde con el nodo a copiar desde el esquema del tipo de datos de negocio y el segundo el nodo del tipo de dato georreferenciado donde se agregara el resultado del primer registro.
- Configuración Xpath de Parámetros WS – Nombre del archivo de configuración de los parámetros a utilizar en el método del Web Service a consultar.

Cada registro tiene el formato

<<SERVICE_NAME>>.<<PARAM_NAME>>=<<XPATH_DATA_IMS>>

SERVICE_NAME – Nombre del método del Web Service consultado

PARAM_NAME – Nombre del parámetro

XPATH_DATA_IMS – Ruta XPATH al valor a utilizar en la respuesta IMS

- Configuración para Mapeo de Datos – Nombre del archivo de configuración para el mapeo de datos, el cual será utilizado por la plataforma GeoEEIP para la generación de la transformación XSLT que enriquecerá un tipo de dato georreferenciado con un tipo de dato empresarial. Entre los tipos de mapeos se cuenta con identificación directa, identificadores de datos georreferenciados compuestos por dos o más datos empresariales e identificación indirecta basada en una tabla de correlación de valores. El formato detallado del mismo se encuentra en el Apéndice 3.

Configurador EEIP

Componentes

-----Simples-----

GeoProxy

GeoRouter

GeoCapabilitiesAdapter

GeoFeatureTypeAdapter

GeoFeatureInfoEnricher

-----Complejos-----

GeoEnricher

WMS SOAP Wrapper

Componente GeoEnricher

Nombre :

Descripción :

URL IMS :

Protocolo :

Crear una nueva fuente

Nombre : *

WSDL : *

Método SOAP : *

Esquema del Tipo de Datos (XSD) : *

Configuración Xpath para Tipo de Datos : *

Configuración Xpath de Parametros WS : *

Configuración para Mapeo de Datos : *

Nombre	WSDL	Método SOAP	Esquema del Tipo de Datos	Configuración Xpath para Tipo de Datos	Configuración para Mapeo de Datos	Operation
No records found.						

Ilustración 49 - Formulario configuración GeoEnricher

A continuación se presentan ejemplos de valores de los parámetros necesarios para la configuración de un enriquecimiento WMS y WFS utilizando una fuente empresarial.

Parámetro	Valor
Nombre	WMSEnricher
Descripción	Este es un ejemplo de uso del componente WMS Enricher
URL IMS	http://localhost:8081/geoserver/wms
Protocolo	WMS

Tabla 11 - Configuración Básica WMS

Parámetro	Valor
Nombre	FuenteBPS
WSDL	http://localhost:8080/BPSEnterpriseDataA/BPSEnterpriseWSA?wsdl
Método SOAP	infoLocal
Esquema del Tipo de Datos	**No aplica**
Configuración Xpath para Tipo de Datos	**No aplica**
Configuración Xpath de Parámetros WS	ws_query_params_xpath_FuenteBPS.properties
Configuración para Mapeo de Datos	MapConfig_infoLocal.xml

Tabla 12 - Configuración Fuente de Datos WMS

Parámetro	Valor
Nombre	WFSEnricher
Descripción	Este es un ejemplo de uso del componente WFS Enricher
URL IMS	http://localhost:8081/geoserver/wfs
Protocolo	WFS

Tabla 13 - Configuración Básica WFS

Parámetro	Valor
Nombre	FuenteEspacioLibre
WSDL	http://localhost:8080/BPSEnterpriseDataA/BPSEnterpriseWSB?wsdl
Método SOAP	infoEspacioLibre
Esquema del Tipo de Datos	InfoEspacioLibre.xsd
Configuración Xpath para Tipo de Datos	InfoEspacioLibre.properties
Configuración Xpath de Parámetros WS	ws_query_params_xpath_InfoEspacioLibre.properties
Configuración para Mapeo de Datos	testMapConfig_InfoEspacioLibre_WFS.xml

Tabla 14 - Configuración Fuente de Datos WFS

4.5. Herramientas utilizadas

A continuación se presentan las herramientas utilizadas en el proyecto.

Versionado

Para el versionado de fuentes y documentación se utilizó SVN. Este servicio está provisto por Assembla¹⁷.

Servidores

El servidor de aplicación utilizado es JBoss Application Server 6.1.0¹⁸. El mismo corre el servidor JBossESB 4.12¹⁹. Adicionalmente se utilizó el servidor WEB Apache Tomcat 7.0.50, en la realización del Caso de Estudio. El servidor de mapas utilizado (IMS), es GeoServer 2.4.2²⁰, el cual accede al motor de base de datos PostgreSQL 9.3.4²¹. Para publicar las capas en GeoServer se utiliza postGIS 2.0²².

IDE de desarrollo

Se utilizó Eclipse Kepler, en conjunto con los plugins JBoss ESB Tools 1.5.200.Final²³ y Maven 2.2.1.

Desarrollo WEB

Para el desarrollo de la Aplicación de Configuración se utilizó PrimeFaces 4.0²⁴, como plataforma de desarrollo y Bootstrap 1.0.10. En tanto para el Caso de Estudio se utilizó OpenLayers 2.12 para la visualización del mapa e invocación de Web Services geográficos y JQuery 1.10.2 para la manipulación de los datos geográficos y el HTML.

Diagramas

Los diagramas utilizados en la documentación, fueron creados mediante la herramienta Lucid Chart.

Verificación

Para la realización de pruebas del componente SOAP-WMS Wrapper se utilizó SoapUI 4.6.0.

¹⁷ <https://www.assembla.com>

¹⁸ <http://download.jboss.org/jbossas/6.1/jboss-as-distribution-6.1.0.Final.zip>

¹⁹ <http://download.jboss.org/jbossesb/4.12/binary/jbossesb-server-4.12.zip>

²⁰ <http://blog.geoserver.org/2013/11/19/geoserver-2-4-2-released/>

²¹ <http://www.postgresql.org/>

²² <http://postgis.net/>

²³ <http://tools.jboss.org/downloads/jbosstools/kepler/4.1.2.Final.html>

²⁴ <http://www.primefaces.org/>

5. Caso de estudio

En este capítulo se presenta un caso de estudio, que tiene por objetivo desarrollar una aplicación para la elección de un local de cobro del BPS²⁵. Este caso de estudio sirve como base para futuras aplicaciones web que deban integrar datos geográficos con datos de negocio, mediante la utilización de la plataforma GeoEEIP. A su vez sirve como caso de prueba de GeoEEIP, para poder detectar posibles desajustes en la integración. En la sección 5.1 se presenta un análisis del caso de estudio, evaluando objetivo, aspectos técnicos y el escenario de prueba. En la sección 5.2 se describe el escenario en el que se prueba el sistema, describiendo las funcionalidades utilizadas. En la sección 5.3 se presentan los componentes integración generados a partir de la aplicación de configuración y los Web Services empresariales utilizados. En la sección 5.4 se realiza una comparativa con la solución planteada en “Integración de Información Geográfica en Sistemas Empresariales”[32]. En la sección 5.5 se expresan los resultados obtenidos. Por último en la sección 5.6 se plantean las funcionalidades que quedan por fuera del caso de estudio.

5.1. Análisis

En esta sección se realiza un análisis del caso de estudio, evaluando objetivo, aspectos técnicos y el escenario de prueba.

5.1.1. Objetivo

A partir del trabajo “Integración de Información Geográfica en Sistemas Empresariales”[32], analizado en la sección 2.3.2, surge la realización de este caso de estudio, el cual implementa un prototipo para la selección de un local de cobro del BPS, asociado con el despliegue de información de negocio georreferenciada. Para su implementación se requiere desplegar un mapa de Montevideo con los puntos que representan los locales de cobro, a partir de los cuales el usuario será capaz de seleccionar uno en particular y así visualizar dicha información de negocio georreferenciada.

5.1.2. Aspectos técnicos

La aplicación web estará implementada mediante el uso de OpenLayers²⁶ y JQuery²⁷, como tecnologías base para el front-end. La información geográfica será obtenida a través del IMS utilizado, el cual consta de cuatro capas: una para las manzanas de Montevideo (capa base a desplegar), otra para las calles, otra para los servicios y la última correspondiente a los espacios libres. Estas capas pertenecen a la Intendencia de Montevideo y son provistas como material para la realización del caso de estudio. El mapa a desplegar debe contener las cuatro capas mencionadas y la capacidad de acceder a servicios WMS y WFS de GeoServer, a través de GeoEEIP. Las capas de manzanas, calles y servicios serán consultadas a través de WMS, mientras que la capa de espacios libres, será consultada mediante WFS.

Para las capas consultadas mediante WMS, se crean estilos particulares SLD²⁸, de forma de obtener una visualización apropiada para el usuario final. Estos estilos son configurados en GeoServer, mientras que para la capa WFS se define el estilo en la consulta (de forma programática), debido a que el uso de los archivos SLD desde GeoServer para especificar la apariencia de una capa, se limita a WMS.

²⁵ BPS: <http://www.bps.gub.uy/>

²⁶ OpenLayers: <http://openlayers.org/>

²⁷ JQuery: <http://jquery.com/>

²⁸ SLD (Styled Layer Descriptor): <http://www.opengeospatial.org/standards/sld>

Los datos empresariales del enriquecimiento, se brindan a través de Web Services provistos por los estudiantes, permitiendo simular datos de negocio que tengan valor para el caso de estudio. Los Web Services empresariales se desarrollarán mediante las herramientas de JBoss Tools disponibles para Eclipse (IDE de desarrollo).

5.2. Escenario de prueba

El usuario accede al sistema, visualizando el mapa con los locales de cobro de BPS para el departamento de Montevideo. Para acceder a la información correspondiente al local de su preferencia, deberá seleccionar dicho local en el mapa. El sistema debe desplegar la información geográfica, enriquecida con información de negocio. Para esto se utilizan las operaciones: GetMap que devuelve las capas a utilizar, GetFeatureInfo que se ejecuta cuando el usuario selecciona un local de cobro, GetFeature ejecutada para obtener la capa de espacios libres (WFS) y GetCapabilities, que muestra al usuario las capacidades del servicio que se está utilizando.

Inicialmente se despliega el mapa de Montevideo, conteniendo las capas de manzanas, calles y servicios. La capa correspondiente a manzanas se toma como capa base, luego se despliega la capa de calles y por último la de servicios que identifica a los locales de cobro. Estas capas son el resultado de la ejecución de la operación GetMap, utilizando el Componente Complejo WMS Enricher.

En el caso en que un usuario seleccione un local de cobro, se desplegará la información de negocio georreferenciada a través de una petición GetFeatureInfo, la cual será direccionada al GeoEEIP y el mismo se encarga de llamar a su mecanismo (componente Complejo) WMS Enricher, para retornar la respuesta IMS enriquecida.

El usuario es capaz de seleccionar los espacios libres y de esta manera visualizar la información enriquecida referente a la capa, a través de una ventana emergente sobre el mapa (similar a la visualización de datos sobre un local de cobro). Para esta funcionalidad el cliente ejecuta una consulta sobre los datos obtenidos al momento de desplegar la capa vectorial de espacios (capa WFS). Para desplegar esta capa se ejecuta la operación GetFeature de WFS, dicha petición es dirigida al GeoEEIP quien se encarga de resolverla a través de su componente Complejo WFS Enricher. El componente complejo devuelve la respuesta enriquecida de forma transparente para el cliente.

En la Figura 50 se presenta un diagrama que muestra las interacciones de componentes a alto nivel para un enriquecimiento. Un cliente OpenLayers realiza una petición IMS dirigida al GeoEEIP, el cual consulta al IMS definido en configuración, con los datos de la consulta. El resultado es guardado internamente en el GeoEEIP (dentro del mensaje ESB), para luego consultar a la/las fuentes empresariales y con estos resultados realizar el enriquecimiento de los datos geográficos obtenidos en el paso 2. Este enriquecimiento varía según el tipo de petición ejecutada. Para el caso de GetFeature se llama al Componte complejo WFS Enricher, mientras que para GetFeatureInfo se llama al Complejo WMS Enricher. Una vez que se tiene una respuesta geográfica enriquecida, se retorna el resultado al cliente, el cual la interpreta como una respuesta de un Web Service geográfico.

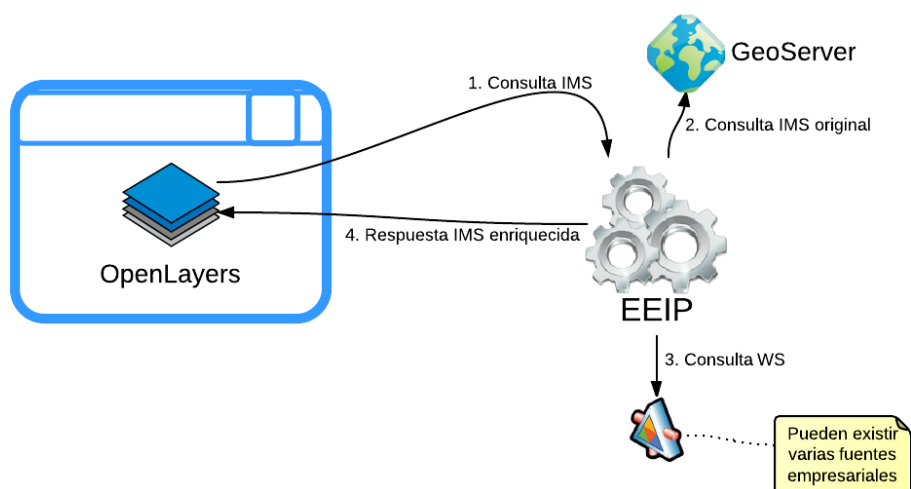


Figura 50 - Esquema de Interacción (Caso de Estudio - enriquecimiento)

Para lograr que las respuestas tengan los formatos correspondientes (WMS y WFS), fue necesario un análisis de dichos formatos. Para el caso de WMS el resultado de la fuente empresarial es añadido a la respuesta de IMS en formato XML (en particular GML), según lo que se especifique en la configuración del enriquecimiento. Para WFS esto no es posible debido a que el formato de la respuesta exige que el nombre de los nodos agregados tenga el prefijo utilizado en la capa a enriquecer.

En la Figura 51 se presenta un ejemplo de GetFeatureInfo (información de local de cobro) y la vista que genera el cliente.



Figura 51 - Caso de estudio (mapa)

A continuación se presenta un fragmento de la respuesta del IMS (Figura 52) y su transformación (Figura 53) para el caso anterior, enmarcando los datos agregados luego del enriquecimiento:

```
<?xml version="1.0" encoding="UTF-8"?>
<wfs:FeatureCollection xmlns:wfs="http://www.opengis.net/wfs" xmlns="http://www.opengis.net/wfs"
  xmlns:gml="http://www.opengis.net/gml" xmlns:eeip="http://www.fing.edu.uy"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.opengis.net/wfs http://localhost:8081/geoserver/schemas/wfs/1.0.0/WFS-basic.xsd http://www.fing.edu.uy
    http://localhost:8081/geoserver/wfs?service=WFS&version=1.0.0&request=DescribeFeatureType&typeName=eeip%3Aserv_comerciales">
  <gml:boundedBy>
    <gml:null>unknown</gml:null>
  </gml:boundedBy>
  <gml:featureMember>
    <eeip:serv_comerciales fid="serv_comerciales.17">
      <eeip:gid>17</eeip:gid>
      <eeip:tipo>local_cobro</eeip:tipo>
      <eeip:calle>Calle SUCRE</eeip:calle>
      <eeip:pta>2694.0</eeip:pta>
      <eeip:esq1>Calle CAVIA</eeip:esq1>
      <eeip:esq2>Calle TIMBO</eeip:esq2>
      <eeip:canal1>SANTANDER</eeip:canal1>
      <eeip:nbre>SANTANDER</eeip:nbre>
      <eeip:barrio>Pocitos</eeip:barrio>
      <eeip:geom>
        <gml:Point srsName="http://www.opengis.net/gml/srs/epsg.xml#32721">
          <gml:coordinates decimal="." cs="," ts=" ">577016,6136774</gml:coordinates>
        </gml:Point>
      </eeip:geom>
    </eeip:serv_comerciales>
  </gml:featureMember>
</wfs:FeatureCollection>
```

Figura 52 - Respuesta GetFeatureInfo (GML)

```
<?xml version="1.0" encoding="UTF-8"?>
<wfs:FeatureCollection xmlns:wfs="http://www.opengis.net/wfs" xmlns="http://www.opengis.net/wfs"
  xmlns:gml="http://www.opengis.net/gml" xmlns:eeip="http://www.fing.edu.uy"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.opengis.net/wfs http://localhost:8081/geoserver/schemas/wfs/1.0.0/WFS-basic.xsd http://www.fing.edu.uy
    http://localhost:8081/geoserver/wfs?service=WFS&version=1.0.0&request=DescribeFeatureType&typeName=eeip%3Aserv_comerciales">
  <gml:boundedBy>
    <gml:null>unknown</gml:null>
  </gml:boundedBy>
  <gml:featureMember>
    <eeip:serv_comerciales fid="serv_comerciales.17">
      <eeip:gid>17</eeip:gid>
      <eeip:tipo>local_cobro</eeip:tipo>
      <eeip:calle>Calle SUCRE</eeip:calle>
      <eeip:pta>2694.0</eeip:pta>
      <eeip:esq1>Calle CAVIA</eeip:esq1>
      <eeip:esq2>Calle TIMBO</eeip:esq2>
      <eeip:canal1>SANTANDER</eeip:canal1>
      <eeip:nbre>SANTANDER</eeip:nbre>
      <eeip:barrio>Pocitos</eeip:barrio>
      <eeip:geom>
        <gml:Point srsName="http://www.opengis.net/gml/srs/epsg.xml#32721">
          <gml:coordinates decimal="." cs="," ts=" ">577016,6136774</gml:coordinates>
        </gml:Point>
      </eeip:geom>
      <eeip:InfoLocalPago xmlns="" id="17" servicios="DHL BANRED PAGO DE PASIVIDADES">
        <eeip:horario dia="Lunes a viernes" hora="9:00hs a 18:00hs"/>
      </eeip:InfoLocalPago>
    </eeip:serv_comerciales>
  </gml:featureMember>
</wfs:FeatureCollection>
```

Figura 53- Respuesta GetFeatureInfo enriquecida (GML)

Como se puede apreciar, los datos del enriquecimiento que se muestran en la Figura 53 corresponden con parte de los datos desplegado por el cliente en la Figura 51. El resto de los datos de la ventana emergente, son datos geográficos que se encuentran en el GML de respuesta.

Otra funcionalidad extra, es la de poder visualizar las capacidades del servicio (GetCapabilities). Si bien este punto no brinda valor al usuario final del BPS, permite mediante el caso de estudio evaluar otra de las operaciones que es soportada por GeoEEIP, dando visibilidad al funcionamiento del mismo. La información

desplegada por GetCapabilities, se presenta de forma amigable para el usuario, teniendo de todos modos la posibilidad de visualizar en XML la respuesta completa del IMS. La Figura 54 muestra la visualización de esta respuesta:



Figura 54 - Caso de estudio (GetCapabilities)

5.3. Componentes Configurados

Para la configuración del caso de estudio se determina la creación de dos componentes complejos, los cuales surgen a partir del análisis, estos son: WMS Enricher y WFS Enricher. Para ambos se utilizó como fuente de datos geográfica un GeoServer local, el cual fue previamente configurado con las capas y los estilos necesarios para su utilización. El uso de las fuentes empresariales determinó la generación de varias instancias del caso de estudio. El primero, y más simple, involucra una sola fuente de datos empresariales a través de un Web Service SOAP. El segundo tiene una fuente, igual que el primero, pero en este caso debe recibir más de un parámetro como entrada a la consulta del WS. El tercer caso que involucra más de una fuente empresarial, en particular se implementó con dos Web Services SOAP. Por último, el cuarto caso implica una tabla de correlación (ver Apéndice 3) para el mapeo de datos geográficos con datos empresariales.

Los Servicios empresariales necesarios, se desarrollaron como un único Web Service, que expone las operaciones necesarias. Estas operaciones son: infoLocal, la cual retorna los datos de un local a partir de un identificador; infoTransporte, retorna información de las opciones de transporte a partir del identificador y el tipo de local; infoEspaciosLibres, la cual retorna información acerca del espacio libre consultado a partir del identificador. El identificador utilizado depende de cada operación, en infoLocal se utiliza el obtenido de la entidad geográfica, mientras que en infoEspaciosLibres es necesaria una función de correlación, ya que el valor de la entidad geográfica es diferente del valor del identificador de negocio requerido por el servicio.

La Figura 55 presenta un diagrama de deployment el cual abarca los casos generados. Y muestra cuatro dispositivos. El primero corresponde al dispositivo del usuario que desea consultar la información de un local de cobro, realizándolo a través de un browser (navegador web) en un dispositivo para tal fin. Por otro lado está el dispositivo del administrador, el cual debe contar con un browser para poder acceder a la aplicación web de configuración y a través de ésta generar los paquetes a ser desplegados en el servidor JBossESB.

Luego se tiene un Apache Tomcat (Servidor Web), el cual va a contener el GeoServer con los datos geográficos a ser consultados. Por último está el servidor de aplicación JBoss, que contiene al JBossESB. El ESB tiene desplegados los paquetes para los diferentes casos de GeoEEIP. A su vez dentro del servidor web que forma parte de la distribución de JBoss, se encuentra la aplicación web de configuración.

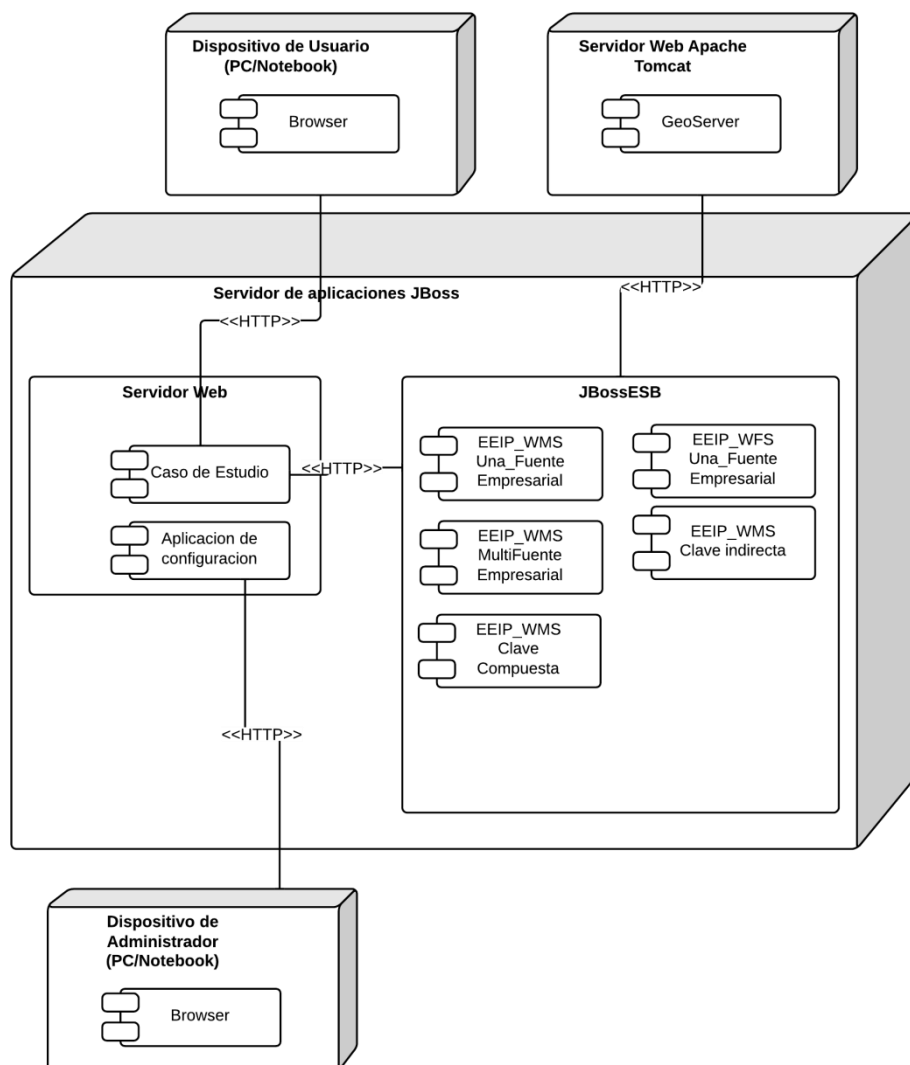


Figura 55 - Deploy Caso de estudio

5.4. Comparativa con documento de referencia

Cabe destacar que el caso de estudio difiere del realizado en “Integración de Información Geográfica en Sistemas Empresariales”[32] en dos puntos fundamentales. El primero tiene que ver con la forma de resolver la problemática de una solución que abarque la mayor cantidad de escenarios posibles de integración entre los datos geográficos y los datos empresariales. Esto lo brinda la utilización de GeoEEIP como plataforma de integración. Si bien se desconoce la implementación del caso de estudio planteado en el documento de referencia [32], ambos buscan una solución común para el usuario final y ésta es poder mostrar en un mapa información empresarial y geográfica sobre un local de cobro del BPS. La forma en que el documento de referencia plantea el enriquecimiento, implica un mapeo entre datos empresariales y datos geográficos

mediante la correlación dada por un identificador específico en los datos de negocio, con otro identificador específico en los datos geográficos. En contrapartida, la forma en que GeoEEIP hace frente a la implementación del caso de estudio, involucra la utilización del configurador, en el cual se define la correlación de datos geográficos con datos empresariales, permitiendo abarcar la mayor cantidad de casos posibles y sin la necesidad de recodificar (se basa en la configuración).

El otro punto fundamental que los diferencia es el uso de un ESB, ya que por motivos de complejidad no fue incorporado en [32]. Para el caso de GeoEEIP es la base de su implementación y de las prestaciones que éste provee.

5.5. Resultados obtenidos

El caso de estudio permite ver en operación las funcionalidades principales de GeoEEIP como integrador, permitiendo mostrar a usuarios finales datos geográficos integrados con datos empresariales de manera transparente. Está fuera del alcance del caso de estudio abarcar todas las operaciones posibles, pero se considera un conjunto suficiente que permita la ejecución de los mecanismos de integración que provee GeoEEIP. En particular se prueban los componentes complejos WMS Enricher y WFS Enricher.

Para el componente WMS Enricher se prueban las operaciones:

- GetFeatureInfo. Debe realizar enriquecimiento e involucra los siguientes componentes básicos: GeoEntryPoint, GeoProxy y GeoFeatureInfoEnricher.
- GetMap. Debe retornar la capa solicitada (binario), en este caso no se realiza enriquecimiento y los componentes básicos involucrados son: GeoEntryPoint y GeoProxy.
- GetCapabilities. Debe retornar la información acerca del servicio WMS y no se realiza enriquecimiento. Los componentes básicos involucrados son: GeoEntryPoint y GeoProxy.

Para el componente WFS Enricher se prueban las operaciones:

- GetFeature. Debe realizar enriquecimiento e involucra los componentes básicos: GeoEntryPoint, GeoProxy y GeoFeatureEnricher.

5.6. Funcionalidades no comprendidas en el caso de estudio

El componente SOAP – WMS no está comprendido dentro del caso de estudio, debido al alcance y requerimientos del mismo. Para poder hacer pruebas sobre el componente, que permitieran ver su funcionamiento, se utilizó la aplicación SoapUI²⁹. Esto permitió validar la invocación de las operaciones de WMS a través de SOAP, con las especificaciones presentadas en “OWS 1.2 SOAP Experiment Report” [29].

En la Figura 56 se ve un ejemplo de invocación, la cual es dirigida al componente WMS SOAP Wrapper, invocando a la operación GetFeatureInfo. El panel central se divide en dos partes, a la izquierda se aprecia la petición en formato SOAP y a la derecha la respuesta del IMS a dicha petición.

²⁹ <http://www.soapui.org/>

En la Figura 57 se ve el mensaje SOAP para un llamado a la operación GetMap. En dicha Figura se aprecian los campos necesarios para una petición REST, expresados en los campos del cuerpo del mensaje SOAP. La Figura 58 muestra la respuesta de la misma, en la cual la etiqueta GetMapResult contiene el binario de la imagen en formato Base64.

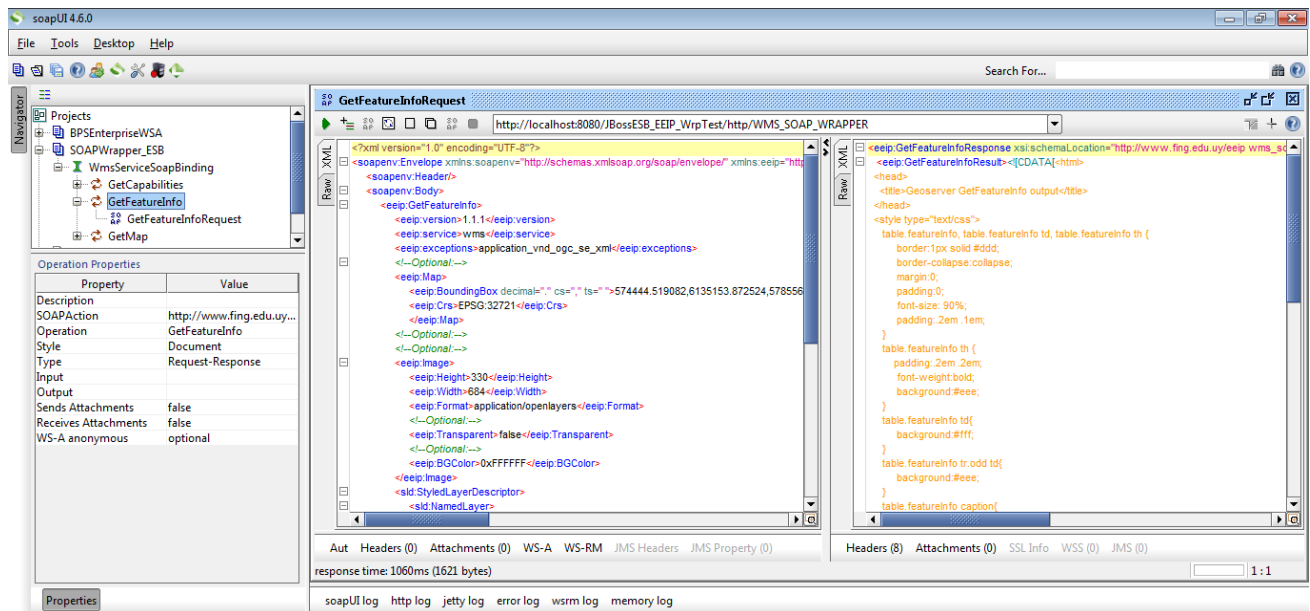


Figura 56 - soapUI, GetFeatureInfo SOAP

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:eeip="http://www.fing.edu.uy/eeip"
  xmlns:sld="http://www.opengis.net/sld" xmlns:se="http://www.opengis.net/se" xmlns:xlink="http://www.w3.org/1999/xlink"
  xmlns:ogc="http://www.opengis.net/ogc" xmlns:gml="http://www.opengis.net/gml">
  <soapenv:Header/>
  <soapenv:Body>
    <eeip:GetMap>
      <eeip:version>1.1.1</eeip:version>
      <eeip:service>wms</eeip:service>
      <eeip:exceptions>application/vnd.ogc.se_xml</eeip:exceptions>
      <eeip:Map>
        <eeip:BoundingBox decimal="." cs=" " ts=" " >-179.99978348919961,
          -89.99982838943765,180.0000000000001,83.63381093402974</eeip:BoundingBox>
        <eeip:Crs>EPSG:4326</eeip:Crs>
      </eeip:Map>
      <eeip:Image>
        <eeip:Height>330</eeip:Height>
        <eeip:Width>684</eeip:Width>
        <eeip:Format>application/openlayers</eeip:Format>
      </eeip:Image>
      <sld:StyledLayerDescriptor>
        <sld:NamedLayer>
          <se:Name>ne:ne_10m_admin_0_countries</se:Name>
        </sld:NamedLayer>
      </sld:StyledLayerDescriptor>
    </eeip:GetMap>
  </soapenv:Body>
</soapenv:Envelope>
```

Figura 57 - Mensaje SOAP - GetMap

```

<eeip:GetMapResponse xsi:schemaLocation="http://www.fing.edu.uy/eeip wms_schemas.xsd"
  xmlns:eeip="http://www.fing.edu.uy/eeip" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <eeip:GetMapResult>PD94bWwgdmVyc2lvdj0iMS4wIiBlbmNvZGluZz0iVVR
  |GLTgiIH0YW5kYWxvbU9Im5vIj8+PCFET0NUWVBFIFNlcnZpY2VFeGN+</eeip:GetMapResult>
</eeip:GetMapResponse>

```

Figura 58 - Respuesta a GetMap

En la Figura 59 se ve el mensaje SOAP para un llamado a la operación GetCapabilities. En dicha Figura se aprecian los campos necesarios para una petición REST, expresados en los campos del cuerpo del mensaje SOAP. La Figura 60 muestra la respuesta de la misma, la cual aparece simplificada, mostrando la definición del servicio. Todos los campos de esta respuesta están especificados en la sección 2.1.2.1.

```

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:eeip="http://www.fing.edu.uy/eeip">
  <soapenv:Header/>
  <soapenv:Body>
    <eeip:GetCapabilities>
      <eeip:version>1.1.1</eeip:version>
      <eeip:service>wms</eeip:service>
      <eeip:exceptions>application_vnd_ogc_se_xml</eeip:exceptions>
      <eeip:style>full</eeip:style>
    </eeip:GetCapabilities>
  </soapenv:Body>
</soapenv:Envelope>

```

Figura 59 - Mensaje SOAP – GetCapabilities

```

<!DOCTYPE WMT_MS_Capabilities SYSTEM "http://localhost:8081/geoserver/schemas/wms/1.1.1/WMT_MS_Capabilities.dtd">
<WMT_MS_Capabilities version="1.1.1" updateSequence="212">
  <Service>
    <Name>OGC:WMS</Name>
    <Title>GeoServer Web Map Service</Title>
    <Abstract></Abstract>
    <KeywordList>
      <Keyword>WFS</Keyword>
      <Keyword>WMS</Keyword>
      <Keyword>GEOSERVER</Keyword>
    </KeywordList>
    <OnlineResource xlink:type="simple" xlink:href="http://geoserver.sourceforge.net/html/index.php"
      xmlns:xlink="http://www.w3.org/1999/xlink"/>
    <ContactInformation>
      <ContactPersonPrimary>
        <ContactPerson>Proyecto de grado</ContactPerson>
        <ContactOrganization>Proyecto de grado</ContactOrganization>
      </ContactPersonPrimary>
      <ContactPosition>Chief geographer</ContactPosition>
      <ContactAddress>
        <AddressType>Work</AddressType>
        <Address/>
        <City>Montevideo</City>
        <StateOrProvince/>
        <PostCode/>
        <Country>Uruguay</Country>
      </ContactAddress>
      <ContactVoiceTelephone/>
      <ContactFacsimileTelephone/>
      <ContactElectronicMailAddress>pg@gmail.com</ContactElectronicMailAddress>
    </ContactInformation>
    <Fees>NONE</Fees>
    <AccessConstraints>NONE</AccessConstraints>
  </Service>

```

Figura 60 - Respuesta a GetCapabilities

6. Conclusiones y trabajo futuro

En este capítulo se presentan las conclusiones del trabajo realizado y los posibles trabajos a futuro. En la sección 6.1 se describe la gestión del proyecto, presentando la metodología de trabajo y el análisis de estimación y desvío. En la sección 6.2 se presentan las conclusiones y por último en la sección 6.3 se presentan los posibles trabajos a futuro.

6.1. Gestión del Proyecto

En esta sección se describe la metodología de trabajo, contemplando la estimación de tiempos inicial, hitos definidos y sus desvíos.

6.1.1. Etapas del proyecto

De modo de organizar el trabajo a realizar, se definieron etapas, cada una con sus particularidades descritas a continuación. Se estima que se tuvo una carga una carga de 20hs promedio por semana, durante la realización del proyecto (14 meses).

- **Estudio de tecnologías base**

En esta etapa cada integrante investigó una tecnología generando un reporte de la información recabada. Se llevaron a cabo reuniones de grupo semanales de seguimiento e intercambio de conocimientos adquiridos.

- **Estudio de propuesta de Plataforma de Integración para Web Services Geográficos.**

Se investigaron diversas soluciones de integración de información geográfica y de negocio, tanto en el ámbito académico como comercial. El objetivo de esta investigación fue analizar la reutilización de las ideas vistas en estas propuestas, para la implementación que se debía realizar en el proyecto.

- **Instalación del ambiente de desarrollo y pruebas de concepto**

En esta etapa se investigaron las plataformas necesarias para el desarrollo del proyecto y se definieron versiones a utilizar. Se evaluaron distintas herramientas de desarrollo de las cuales algunas se descartaron. Finalmente se obtuvo un ambiente completo y funcional para las tareas de desarrollo.

- **Documentación**

Para esta etapa se generó el documento de Estado del Arte (incluido en este documento), adicionalmente se realizó un análisis de requerimientos, el planteo de cronograma a seguir y la preparación de una presentación intermedia para poder realizar una puesta a punto con los tutores.

- **Implementación de GeoEEIP y aplicación de Configuración**

En esta etapa se realizó la implementación de los componentes básicos que conforman la solución. En el transcurso de la misma, se realizaron reuniones diarias.

- **Caso de estudio**

Utilizando la plataforma implementada, se desarrolló una aplicación para la elección de un local de cobro del BPS, como caso de prueba para esta plataforma. De esta manera se verificó el funcionamiento de las principales operaciones de la plataforma.

- **Informe Final**

En esta etapa se elaboró la documentación final del proyecto, la cual comenzó en paralelo con la etapa de implementación. Esta documentación comprende el documento final, apéndices y anexos.

- **Preparación de la presentación final**

Se crea una presentación para plantear los conceptos relacionados y demostrar el funcionamiento de la plataforma.

6.1.2. Análisis de estimación y desvío

En esta sección se muestra una gráfica comparativa entre tiempos estimados para cada tarea, y el tiempo de realización. Esto permite ver de forma clara la desviación acumulada.

	2013										2014					
Tareas	Abr	May	Jun	Jul	Ago	Set	Oct	Nov	Dic	Ene	Feb	Mar	Abr	May	Jun	
Estudio de tecnologías base																
Estudio de propuesta de Plataforma de Integración para Web Services Geográficos																
Instalación del ambiente de desarrollo y pruebas de concepto.																
Estado del Arte																
Implementación de GeoEEIP																
Implementación del Configurador																
Caso de estudio																
Informe Final																
Preparación de la presentación final																

Tabla 15 - Tiempo estimado y desvios

En la primera y segunda etapa no existe desvío debido a que se tratan de tareas acotadas en el tiempo, para las cuales estaba bien definido el período a dedicar a cada una.

Para la instalación del ambiente de desarrollo al igual que las pruebas de concepto, se dedicó un tiempo considerablemente mayor al estimado. Esto se debe a las siguientes razones:

- Compatibilidad de versiones para las tecnologías de desarrollo.
- Evaluación del IDE y herramientas de desarrollo complementarias
- Curva de aprendizaje de JBossESB, debido a la falta de detalle en la documentación y ejemplos. Si bien existen una guía de usuario y otra de desarrollo, ambas se basan en los ejemplos, sin detallar los componentes.

En la creación del documento de Estado del Arte y análisis de soluciones existentes se dedicó un tiempo fijo, determinado previamente, en el cual se trató de abarcar la mayor cantidad posible de soluciones existentes y de las tecnologías a utilizar en el proyecto.

La implementación del GeoEEIP tuvo un gran desfasaje debido a la falta de conocimiento de las tecnologías de desarrollo, en particular de JBossESB. Si bien en la primera etapa el grupo se familiarizó con la tecnología, los conocimientos adquiridos no abarcaron los detalles de implementación necesarios para dominarla. Esto se vio reflejado en una subestimación.

Los tiempos de desarrollo de la aplicación de configuración se vieron afectados por la falta de conocimiento sobre las tecnologías web a utilizar en el mismo que no estuvieron en la etapa de Estado del Arte. Además las funcionalidades requeridas para el mismo resultaron mucho más complejas que lo estimado en un principio y debían ser adaptadas acorde a la evolución de la Plataforma.

El caso de estudio partió de la base del documento de referencia y fue evolucionando para poder probar la mayor cantidad posible de funcionalidades de GeoEEIP. Parte de los tiempos de desfasaje corresponden a los ajustes, los cuales fueron surgiendo en conjunto con los tutores y el usuario. Otro factor que incidió en los tiempos fue que no se previó el estudio de las tecnologías utilizadas en la etapa de Estado del Arte.

La realización del documento final, presentó un desfasaje debido a la falta de experiencia en este tipo de documentación, por parte de los estudiantes. Esto generó re trabajo por parte de todos los involucrados, tanto los estudiantes como tutores.

6.2. Conclusiones

Este proyecto consistió en la implementación de la plataforma de integración de servicios geográficos y empresariales GeoEEIP, propuesta presentada en el artículo[1]. Ésta permite contar con soluciones genéricas a problemáticas de integración usuales, vistas en la Sección 1.1.

La plataforma se desarrolló sobre JBossESB y por ende hace uso de sus mecanismos de mediación. Ésta está orientada a desarrolladores de aplicaciones que requieran datos georreferenciados, abstrayéndolos de conocimientos avanzados en lo que a JBossESB se refiere. Esto se logra mediante mecanismos de configuración que se desarrollaron para el uso amigable de la plataforma. Adicionalmente se cuenta con todos los componentes generados para la solución, siendo posible la reutilización de los mismos en desarrollos sobre JBossESB.

Los principales aportes de este proyecto son:

- La documentación que se generó a partir de la investigación de los ejemplos provistos por JBossESB.
- Implementación de la plataforma, la cual demuestra la viabilidad de la solución planteada en el artículo[1].
- Implementación de aplicación de Configuración que facilita el uso de la plataforma.
- El caso de estudio desarrollado muestra la aplicación de los procesos de integración dentro de la plataforma GeoEEIP

La solución obtenida facilita la tarea del desarrollador al momento de integrar servicios geográficos con servicios empresariales. Esto se observó en la práctica con el desarrollo del caso de estudio. Ésta integración realizada mediante la aplicación de configuración, resultó sencilla con sólo contar con la especificación de los datos a utilizar.

Dentro de las tecnologías base del proyecto, Web Services geográficos y JBossESB, se observó que han tenido una evolución importante y que cumplieron satisfactoriamente con los objetivos en el contexto del proyecto. Los Web Services geográficos tienen posibilidades de mejoras, las cuales se ven reflejadas mediante el surgimiento de nuevas versiones y nuevas operaciones. Por otra parte JBossESB en su versión 4.12, está limitada a JBoss AS 6. Además, esta implementación de ESB está discontinuada, por lo que para

continuar con el desarrollo de la plataforma, sería conveniente considerar una implementación más actualizada.

Como conclusión final se demuestra la factibilidad técnica de la solución teórica planteada en [1], abriendo posibilidades de estudio tanto en la performance de la solución, como la evaluación de otras implementaciones de ESB en la búsqueda de un camino más eficiente.

6.3. Trabajo a futuro

En esta Sección se presentan los posibles trabajos a futuro, tanto en las mejoras de la plataforma desarrollada, como en el posible uso de la misma.

Caso de estudio

Resulta interesante presentar como caso de estudio, proveer la funcionalidad que brinda el componente SOAP-WMS Wrapper de la plataforma GeoEEIP a la solución presentada en el proyecto “Proyecto TOPO: Análisis y Propuesta para el desarrollo de un nuevo SIG Empresarial en DINAMIGE”[33]. En esta solución se establece el uso de SOAP como protocolo de comunicación entre los componentes, excepto para el caso de la comunicación con los Web Services geográficos, para los cuales se debe utilizar REST. Para hacer frente a esta limitación, se propone el uso de GeoEEIP, en particular el componente SOAP-WMS Wrapper, de manera que solo este componente sea el encargado de la comunicación REST con el IMS. Esto permite a los componentes que lo invoquen, comunicarse mediante SOAP. De esta forma se mantienen las capacidades de SOAP (ver Sección 2.2.4.1) para toda la solución.

Mejoras sobre GeoEEIP

A continuación se presentan las posibles mejoras que se pueden realizar al GeoEEIP. Las mismas surgen de limitaciones conocidas o aspectos que debido al alcance del proyecto o de requerimientos del mismo, no están contemplados en la solución implementada.

Mejoras a nivel de la plataforma:

- Actualmente no se está realizando un manejo de excepciones dentro de JBossESB. En este aspecto se pueden identificar dos grandes tipos de excepciones, por un lado las generadas a través del código propio de GeoEEIP (errores de configuración o posibles bugs) y las que surgen por la interacción con el IMS. En las primeras se puede mostrar al usuario un mensaje acorde con el tipo de error, mientras que para las que surgen por interacción con el IMS sería deseable responder con el mensaje devuelto por el IMS.
- Agregar seguridad a nivel de los servicios de JBossESB. Este tema no forma parte de los requerimientos del proyecto, ya que se encuentra fuera del alcance. JBossESB permite la autenticación para la ejecución de servicios, este tema se puede profundizar en la guía “Services_Guide”[34] de JBossESB.
- Utilizar alguna herramienta que permita realizar pruebas de performance. Además definir y ejecutar casos de pruebas a modo de Testing Funcional.

- Realizar la implementación del GeoEEIP sobre otro producto ESB. Este punto fue mencionado anteriormente en las conclusiones ya que podría ser un punto de simplificación para una nueva implementación y así comparar si existe alguna que se adecue mejor a la plataforma GeoEEIP.
- Agregar otras operaciones para los protocolos utilizados.
- Soportar integración de sistemas de información geográficos que utilicen otros protocolos como pueden ser: WFS 2.0. y WMTS³⁰.

³⁰ http://portal.opengeospatial.org/files/?artifact_id=35326

Referencias

- [1] Bruno Rienzi, Laura González, and Raúl Ruggia, “Towards an ESB-Based Enterprise Integration Platform for Geospatial Web Services.” .
- [2] LINS, “Enterprise Service Bus.” Facultad de Ingenieria - INCO - Curso Introduccion al Middelware, 2013.
- [3] “Overview | Geographic Information Systems,” *What is GIS?* [Online]. Available: http://www.esri.com/what-is-gis/overview#overview_panel. [Accessed: 18-Jan-2014].
- [4] “About OGC | OGC(R).” [Online]. Available: <http://www.opengeospatial.org/ogc>. [Accessed: 30-Nov-2013].
- [5] “Web Map Service | OGC(R).” [Online]. Available: <http://www.opengeospatial.org/standards/wms>. [Accessed: 30-Nov-2013].
- [6] “Web Service Common | OGC(R).” [Online]. Available: <http://www.opengeospatial.org/standards/common>. [Accessed: 10-Aug-2013].
- [7] “Web Feature Service | OGC(R).” [Online]. Available: <http://www.opengeospatial.org/standards/wfs>. [Accessed: 30-Nov-2013].
- [8] “Welcome - GeoServer.” [Online]. Available: <http://geoserver.org/display/GEOS/Welcome>. [Accessed: 30-Nov-2013].
- [9] “Master Executive Suply Chain Management » Sistemas de Información Empresarial.” [Online]. Available: <http://www.eoi.es/blogs/scm/2013/02/21/sistemas-de-informacion-empresarial/>. [Accessed: 15-Mar-2014].
- [10] L. Gonzalez, “Clase 2 - Taller de Sistemas de Información Geográficos Empresariales.” INCO - Facultad de Ingeniería – Montevideo, Uruguay, 2012.
- [11] L. Gonzalez, “Arquitectura Orientada a Servicios.” Lins, 2009.
- [12] L. Gonzalez, “Introducción al middleware.” INCO - Facultad de Ingeniería – Montevideo, Uruguay, 2013.
- [13] “Web Services Architecture.” [Online]. Available: <http://www.w3.org/TR/ws-arch/>. [Accessed: 07-Aug-2013].
- [14] “SOAP Version 1.2 Part 1: Messaging Framework (Second Edition).” [Online]. Available: <http://www.w3.org/TR/soap12-part1/>. [Accessed: 10-Aug-2013].
- [15] LINS, “Web Services.” Facultad de Ingenieria - INCO - Curso Introduccion al Middelware, 2013.
- [16] “Web Service Definition Language (WSDL).” [Online]. Available: <http://www.w3.org/TR/wsdl>. [Accessed: 10-Aug-2013].
- [17] “Fielding Dissertation: CHAPTER 5: Representational State Transfer (REST).” [Online]. Available: http://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm. [Accessed: 10-Aug-2013].
- [18] “Architectural Principles of the World Wide Web.” [Online]. Available: <http://www.w3.org/TR/2002/WD-webarch-20020830/#rest-constraints>. [Accessed: 10-Aug-2013].
- [19] “Meet Mule - Mule 3.4.X Documentation.” [Online]. Available: <http://www.mulesoft.org/documentation/display/34x/Meet+Mule>. [Accessed: 06-Dec-2013].
- [20] T. Rademakers and J. Dirksen, *Open Source Enterprise Services Buses in Action*. Manning Publications Company, 2008.
- [21] D. A. Chappell, *Enterprise service bus*. Beijing; Cambridge: O’Reilly, 2004.
- [22] LINS, “Enterprise Service Bus.” Facultad de Ingenieria - INCO - Curso Introduccion al Middelware, 2013.
- [23] Raquel Sosa, “Integración de Servicios Geográficos en Plataformas de Gobierno Electrónico,” UNIVERSIDAD DE LA REPÚBLICA - FACULTAD DE INGENIERÍA, 2011.
- [24] Emilio Penna, Victoria Perez, and Martín Steffen, “WFS en ESB con Fuente Externa y Cache.” .

- [25] Pablo Garcia, Jorge Laborde, and Martin Rodriguez, “WMS SOAP con ESB.” .
- [26] Mostafa Abou-Ghanem and Khalid A. Arfaj, “SAP/GIS Integration Case Studies & Techniques.” .
- [27] “Home: Integration SAP Primavera GIS.” [Online]. Available: <http://www.impress.com/>. [Accessed: 09-Aug-2013].
- [28] “HOME - AED-SICAD_com.” [Online]. Available: <http://www.aed-sicad.com/>. [Accessed: 09-Aug-2013].
- [29] J. Sonnet and C. Savage, “OWS 1.2 SOAP Experiment Report.” Ionic Software s.a., 15-Jan-2003.
- [30] “Programmers Guide.” [Online]. Available: http://docs.jboss.org/jbossesb/docs/4.12/manuals/html/Programmers_Guide/index.html. [Accessed: 28-Nov-2013].
- [31] “V1.5:Smooks v1.5 User Guide - Smooks.” [Online]. Available: http://www.smooks.org/mediawiki/index.php?title=V1.5:Smooks_v1.5_User_Guide. [Accessed: 20-Jan-2014].
- [32] Juan Herman, Rodrigo Mercader, and Claudio Perrone, “Integración de Información Geográfica en Sistemas Empresariales,” Instituto de Computación - Facultad de Ingeniería - Universidad de la República, 2010.
- [33] Javier Minhondo, “Proyecto TOPO: Análisis y Propuesta para el desarrollo de un nuevo SIG Empresarial en DINAMIGE.” Diciembre-2011.
- [34] “Services Guide.” [Online]. Available: http://docs.jboss.org/jbossesb/docs/4.12/manuals/html/Services_Guide/index.html#d0e2752. [Accessed: 16-Apr-2014].

Apéndice 1. Otros Componentes identificados

En esta Sección se profundizan los mecanismos definidos en la plataforma de integración planteada en el artículo[1], que no se desarrollaron en la Sección 3.2. Estos son: GeoCapabilitiesAdapter; GeoFeatureTypeAdapter; GeoFeatureEnricher; WMS2SOAPTTextTranslator y WMS2SOAPBinaryTranslator. Para cada uno de estos mecanismos se analizan las funcionalidades requeridas y se identifican el o los componentes necesarios para su implementación.

GeoCapabilitiesAdapter

Mecanismo que adapta la respuesta de GetCapabilities realizando la conversión de las URLs del IMS a las URLs de GeoEEIP. Este mecanismo es necesario porque la respuesta de la operación GetCapabilities contiene las URLs del IMS (a través del cual se ejecutó la operación). Entonces para evitar que el cliente invoque las operaciones GetFeatureInfo o GetMap directamente sobre el IMS, se deben reemplazar las URLs del IMS por URLs de GeoEEIP. De esta forma el IMS es transparente para los clientes de GeoEEIP. Este procesamiento se puede ver en la Figura 61.

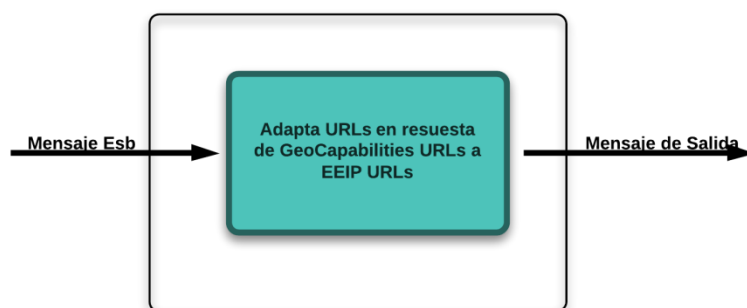


Figura 61 – GeoCapabilitiesAdapter

GeoFeatureTypeAdapter

Mecanismo que enriquece el esquema de XML retornado por la operación de WFS "DescribeFeatureType", con el esquema del Web Services empresarial. Para lograr este enriquecimiento el componente se debe configurar con el esquema del Web Services empresarial y las rutas de los nodos a enriquecer. En la Figura 62 se muestra el componente con el flujo a alto nivel.

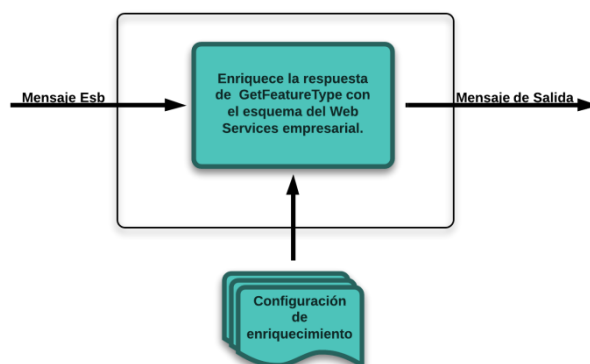


Figura 62 - GeoFeatureTypeAdapter

GeoFeatureEnricher

Mecanismo encargado enriquecer el resultado de la operación de WFS “GetFeature” con la información de negocio (resultado de la consulta a Web Services empresariales). El procesamiento del mensaje se realiza de la misma forma que en el caso del enriquecimiento de la operación WMS GetFeatureInfo (Sección 4.2).

WMS2SOAPTextTranslator y WMS2SOAPBinaryTranslator

Mecanismos que procesan una respuesta WMS, generando como resultado un mensaje SOAP conteniendo la respuesta WMS. Para establecer el formato en que se registra la repuesta WMS en el mensaje SOAP, se sigue el formato especificado en “OWS 1.2 SOAP Experiment Report”[29] .

WMS2SOAPTextTranslator aplica a respuestas que contienen únicamente texto, por ejemplo para las operaciones GetFeatureInfo y GetCapabilities. Por otro lado WMS2SOAPBinaryTranslator aplica a respuestas que contienen datos binarios, es decir, la operación GetMap. En la Figura 63 se muestra el flujo del Mensaje ESB a alto nivel.

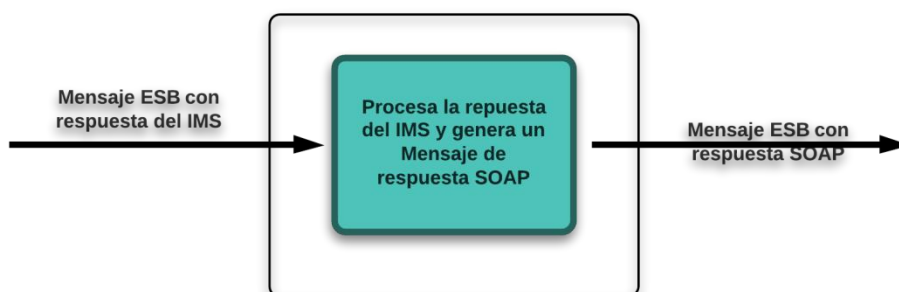


Figura 63 - WMS2SOAPTextTranslator/ WMS2SOAPBinaryTranslator

Apéndice 2. Otros Componentes implementados

A continuación se analizan desde el punto de vista técnico y su implementación, los componentes vistos en el Apéndice 1, el análisis del resto de los componentes se puede ver en las Secciones 4.2 y 4.3.

GeoCapabilitiesAdapter

Este componente procesa el Mensaje ESB en la acción GeoCapabilitiesAdapter (ver Figura 64) que extiende de la acción abstracta AbstractActionPipelineProcessor de JBossESB descrita en la Sección 3.3.1. En el método “process” que implementa la acción GeoCapabilitiesAdapter se realiza la adaptación de las URL de la repuesta a la consulta GetCapabilities que viene en el campo IMS_RESPONSE del cuerpo del mensaje ESB, en particular se reemplaza las URL IMS_ADDRESS por el parámetro LOCAL_ADDRESS. Estos parámetros son parte de la configuración según se indica en la tabla de Parámetros Requeridos.

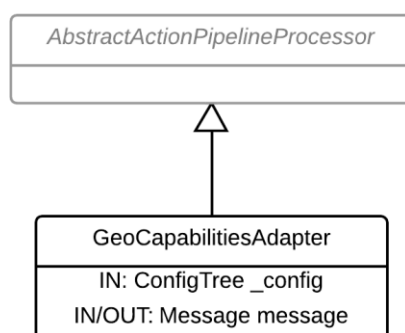


Figura 64 – Diagrama de clases GeoCapabilitiesAdapter

Condiciones

Pre	Post
Mensaje ESB conteniendo la respuesta WMS/WFS para el resultado de la operación GetCapabilities.	Se retorna el mensaje ESB con las rutas retornadas por el GetCapabilities, adaptadas para que sus URL's sean las correspondientes en GeoEEIP.

Parámetros Requeridos

Nombre de parámetro	Descripción
Formato de mensaje de entrada	
<i>IMS_RESPONSE</i>	Respuesta de la consulta IMS en el formato solicitado.
<i>LOCAL_ADDRESS</i>	URL base del servicio creado mediante el GeoEEIP y publicado en ESB.
<i>IMS_ADDRESS</i>	URL base del servicio IMS consultado.
Configuración del componente	
Formato de mensaje de salida	
<i>IMS_RESPONSE</i>	Respuesta de la consulta IMS en el formato solicitado, reemplazando las URL's de las respuestas por la correspondiente en el ESB.

Tabla 16 - GeoCapabilitiesAdapter Pre-Post condiciones y parámetros

GeoFeatureTypeAdapter

El componente procesa el Mensaje ESB en la acción GeoFeatureTypeAdapter (ver Figura 65), que extiende de la acción abstracta AbstractActionPipelineProcessor de JBossESB que se describe en la sección 3.3.1. En el método “process” que implementa la acción GeoFeatureTypeAdapter se realiza la transformación del campo IMS_RESPONSE (que se encuentra en el cuerpo del Mensaje) agregando nodos del esquema del Web Services Empresarial, indicado en el campo de la configuración FEATURE_TYPE_ADAPTER_SCHEMA. Los

XPATH de los nodos a agregar y la ubicación en el esquema de respuesta (DescribeFeatureType) se indican en el archivo de configuración que referencia en la propiedad FEATURE_TYPE_ADAPTER_XPATH.

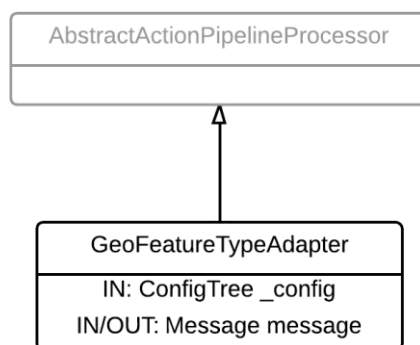


Figura 65 –Diagrama de Clases de el componente GeoFeatureTypeAdapter

Condiciones

Pre	Post
Mensaje ESB que contiene la respuesta WFS de la operación DescribeFeatureType.	Se retorna el mensaje ESB con la información de la respuesta del DescribeFeatureType enriquecido con los datos de definición de las propiedades a agregar en el enriquecimiento.

Parámetros Requeridos

Nombre de parámetro	Descripción
Formato de mensaje de entrada	
<i>IMS_RESPONSE</i>	Respuesta de la consulta IMS en formato XML.
Configuración del componente	
<i>FEATURE_TYPE_ADAPTER_SCHEMA</i>	Ruta del archivo esquema (XSD) de los datos empresariales.
<i>FEATURE_TYPE_ADAPTER_XPATHS</i>	Ruta del archivo que contiene la configuración para el enriquecimiento de la respuesta IMS. Este estará conformado por parejas de xpath con formato fuente-destino.
Formato de mensaje de salida	
<i>IMS_RESPONSE</i>	Respuesta de DescribeFeatureType enriquecida en formato XML.

Tabla 17 - GeoFeatureTypeAdapter Pre-Post condiciones y parámetros

GeoFeatureEnricher

Este componente se implementa de igual forma que el componente GeoFeatureInfoEnricher descrito en la Sección 4.2.

WMS2SOAPTextTranslator y WMS2SOAPBinaryTranslator

Los componentes WMS2SOAPTextTranslator y WMS2SOAPBinaryTranslator procesan el Mensaje ESB de forma similar, adaptando las respuestas WMS para ser retornadas en un mensaje SOAP (ver Figura 66). Ambas acciones extienden de la acción abstracta AbstractActionPipelineProcessor de JBossESB que se describe en la sección 3.4.1. En el procesamiento del mensaje se inserta la respuesta de WMS en el XML con formato SOAP del Mensaje resultante. Para el caso del componente WMS2SOAPBinaryTranslator los binarios son codificados en base 64 para poder ser interpretado por el cliente Web.

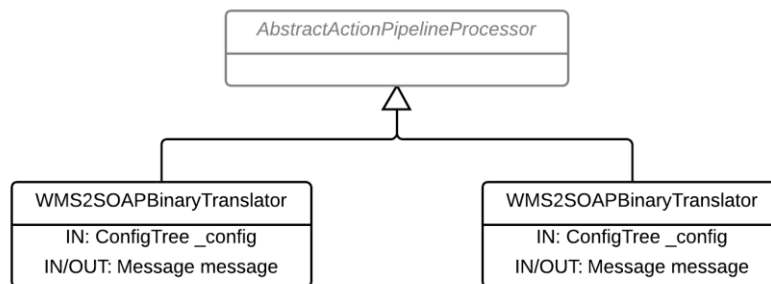


Figura 66 – Diagrama de Clases de WMS2SOAPTextTranslator y WMS2SOAPBinaryTranslator

Condiciones

Pre	Post
Mensaje ESB con el resultado de una operación WMS.	Se genera el SOAP response correspondiente para el retorno de los datos.

Parámetros Requeridos

Nombre de parámetro	Descripción
Formato de mensaje de entrada	
<i>IMS_RESPONSE</i>	Respuesta WMS en el formato solicitado.
<i>QUERY_PARAMS</i>	Diccionario de parámetros generado a partir del parámetro QUERY_STRING.
Configuración del componente	
Formato de mensaje de salida	
<i>IMS_SOAP_RESPONSE</i>	Respuesta WMS incluida en un mensaje de retorno con formato SOAP.

Tabla 18 - WMS2SOAPTextTranslator y WMS2SOAPBinaryTranslator Pre-Post condiciones y parámetros

Apéndice 3. Configuración para mapeo de datos

En este apéndice se describe el archivo de configuración para el mapeo de datos utilizado por GeoEEIP y los posibles casos de uso (tipos de mapeo).

El archivo de configuración utilizado por la Web de Configuración, se utiliza para la generación de la transformación XSLT que enriquece un tipo de dato contenido en la respuesta IMS con un tipo de dato de negocio obtenido desde una fuente de datos (Web Service). El objetivo del archivo es especificar la localización de los datos identificadores de entidades en cada una de las respuestas a utilizar en el enriquecimiento. El formato de esta configuración en formato XML está definido por el archivo de esquema XSD `EnrichmentDataMapping.xsd`, el cual se encuentra junto con el código del Framework desarrollado, a continuación se detalla su utilización.

En esta configuración se pueden realizar tres tipos de mapeo. Mapeo directo, donde la entidad georreferenciada y la entidad de negocio comparten el mismo valor identificadorio. Mapeo compuesto, donde el dato identificadorio de la entidad de negocio está compuesta por dos datos de la entidad georreferenciada. Y por último mapeo indirecto, donde un valor de la entidad georreferenciada se relaciona con un valor diferente en la entidad de negocio, donde el tipo no necesariamente es el mismo.

En la Figura 67 se puede ver un ejemplo del uso de esta configuración, el mismo corresponde al mapeo más sencillo, el mapeo directo. En el mismo se puede apreciar que la configuración se divide en tres tipos. En primer lugar se encuentra la definición de los namespaces utilizados tanto en el XML respuesta del IMS como el retornado por la consulta a la fuente de datos de negocio. Luego se encuentra la definición de los identificadores de datos destino, esto quiere decir la manera de identificar la entidad a enriquecer en la respuesta IMS. Los datos utilizados son: “`destiny-path`” este campo contiene el nombre del nodo contenedor del dato identificadorio, “`destiny-key`” este campo compuesto por tipo de clave (“`type`”) y el nombre de la misma (“`path`”), los tipos de clave disponibles se verán en detalle más adelante en conjunto con la manera de configurar cada tipo de mapeo disponible. Por último se define el identificador del dato fuente de negocio que enriquecerá la respuesta IMS, esta configuración es análoga a la anteriormente vista con la diferencia que los nombres de los campos son: “`source-path`” y “`source-key`”.

```

<?xml version="1.0" encoding="UTF-8"?>
<enrichment-data-mappings xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="../EnrichmentDataMapping.xsd">
    <enrichment-data-mapping>
        <map>
            <!-- NameSpaces utilizados en los datos geo referenciado y datos de negocio -->
            <namespaces>
                <namespace>
                    <name>eeip</name>
                    <uri>http://www.fing.edu.uy</uri>
                </namespace>
                <namespace>
                    <name>bdta</name>
                    <uri>http://www.fing.edu.uy/bdta</uri>
                </namespace>
            </namespaces>
            <!-- Datos necesarios para identificar una entidad en los datos geo referenciados -->
            <destiny-path>eeip:manzanas</destiny-path>
            <destiny-key>
                <key-search-criteria>
                    <type>element</type>
                    <path>eeip:gid</path>
                </key-search-criteria>
            </destiny-key>
            <!-- Datos necesarios para identificar una entidad en los datos de negocio-->
            <source-path>bdta:InfoManzana</source-path>
            <source-key>
                <key-search-criteria>
                    <type>attribute</type>
                    <path>id</path>
                </key-search-criteria>
            </source-key>
        </map>
    </enrichment-data-mapping>
</enrichment-data-mappings>

```

Figura 67 - Ejemplo de configuración de mapeo de datos

Los diferentes tipos de mapeos son configurados mediante variaciones en la definición de los “key-search-criteria”. A continuación se detalla la manera de configurar los tres tipos de mapeo disponibles: simple, indirecto y compuesto. En cada uno de los casos veremos los posibles valores que puede utilizar el campo “type”: “element”, “attribute”, “composition” y “correlation”.

Mapeo directo

Este tipo de mapeo se utiliza en el caso que la clave utilizada para identificar una entidad utilizada por el IMS es igual a la clave utilizada por la fuente de datos empresarial. Este es el caso visto en la Figura 68, aquí se puede apreciar el uso del tipo “element” y “attribute”. El primero especifica que el valor de la clave se encuentra en el nodo XML que tiene como nombre el valor del campo “path” (o sea “eeip:gid”) y que es hijo del nodo referenciado anteriormente (en este caso por “destiny-path”). El siguiente tipo significa que el valor de la clave es el valor del atributo “id” perteneciente al nodo especificado (“source-path” en el ejemplo). A continuación se ve en la Figura 68 un ejemplo de los datos mapeados, a la izquierda los datos georreferenciados y a la derecha los datos de negocio.

```

<gml:featureMember>
  <eeip:manzanas fid="manzanas.1">
    <eeip:gid>21661999</eeip:gid>
    <eeip:geom>
      <gml:MultiPolygon srsName="http://www.opengis.net/gml/srs/epsg.xml#32721">
        <gml:polygonMember>
          <gml:Polygon>
            <gml:outerBoundaryIs>
              <gml:LinearRing>
                <gml:coordinates xmlns:gml="http://www.opengis.net/gml"
                  577039.26026748,6135377.82650333 577011.96642427,6135371
                />
              </gml:LinearRing>
            </gml:outerBoundaryIs>
          </gml:Polygon>
        </gml:polygonMember>
      </gml:MultiPolygon>
    </eeip:geom>
  </eeip:manzanas>
</gml:featureMember>
  
```

```

<ns2:infoCobroBP82Response xmlns:ns2="http://webservices.samples.jboss.org/">
  <bdta:InfoManzana id="21661999" servicios="DHL Cobro">
    <horario dia="Lunes a viernes" hora="9:00hs a 18:00hs"/>
  </bdta:InfoManzana>
</ns2:infoCobroBP82Response>
  
```

Figura 68 - Ejemplo de datos con mapeo directo

Mapeo indirecto

Este es el caso en que un valor clave de una entidad georreferenciada se asocia a un único valor clave de una entidad de negocio. Para realizar este mapeo se utiliza una tabla de correlación la cual se define en la configuración xml (por lo genera este tipo de procesamiento se realiza involucrando alguna fuente de datos y servicios web, esto se simplifico por entenderse fuera del objetivo del proyecto).

La configuración necesaria para este mapeo solo se diferencia a la antes vista en el nodo “destiny-key”, en la Figura 69 se puede ver un ejemplo de configuración de este nodo. En primer lugar el valor del nodo “type” toma el valor correspondiente (“correlation”), además se agrega el nodo “correlation_table” en el cual se definen las parejas de valores relacionados. Esta tabla puede contener tantos registros de correlación como se deseen. En la Figura 70 se pueden apreciar los datos mapeados por esta correlación.

```

<!-- Datos necesarios para identificar una entidad en los datos geo referenciados -->
<destiny-path>eeip:manzanas</destiny-path>
<destiny-key>
  <key-search-criteria>
    <type>correlation</type>
    <path>eeip:gid</path>
    <correlation_table destiny_type="element">
      <correlation geo_id="21661998" bussines_id="1" />
      <correlation geo_id="21661999" bussines_id="2" />
      <correlation geo_id="21662000" bussines_id="3" />
      <correlation geo_id="21662001" bussines_id="4" />
      <correlation geo_id="21662002" bussines_id="5" />
      <correlation geo_id="21662003" bussines_id="6" />
      <correlation geo_id="21662004" bussines_id="7" />
      <correlation geo_id="21662005" bussines_id="8" />
      <correlation geo_id="21662006" bussines_id="9" />
      <correlation geo_id="21662007" bussines_id="10" />
      <correlation geo_id="21662008" bussines_id="11" />
    </correlation_table>
  </key-search-criteria>
</destiny-key>

```

Figura 69 - Configuración de mapeo indirecto

```

<gml:featureMember>
  <eeip:manzanas fid="manzanas.1">
    <eeip:gid>21661999</eeip:gid>
    <eeip:geom>
      <gml:MultiPolygon srsName="http://www.opengis.net/gml/srs/epsq.xml#32721">
        <gml:polygonMember>
          <gml:Polygon>
            <gml:outerBoundaryIs>
              <gml:LinearRing>
                <gml:coordinates xmlns:gml="http://www.opengis.net/gml"
                  577039.26026748,6135377.82650333 577011.96642427,6135371
                />
              </gml:LinearRing>
            </gml:outerBoundaryIs>
          </gml:Polygon>
        </gml:polygonMember>
      </gml:MultiPolygon>
    </eeip:geom>
  </eeip:manzanas>
</gml:featureMember>

```

```

<ns2:infoCobroBP82Response xmlns:ns2="http://webservices.samples.jboss.org/">
  <bdta:InfoManzana id="2" servicios="DHL Cobro">
    <horario dia="Lunes a viernes" hora="9:00hs a 18:00hs"/>
  </bdta:InfoManzana>
</ns2:infoCobroBP82Response>

```

Figura 70 - Ejemplo de datos con mapeo indirecto

Mapeo compuesto

Este último mapeo desarrollado es uno de los casos habituales donde la clave utilizada por las entidades de negocio se corresponden con la concatenación de campos de la entidad georreferenciada (nótese que los campos involucrados no requieren que identifiquen a la entidad georreferenciada de forma única). Para esto se define el nodo “destiny-key” con el ultimo tipo de mapeo “composition” como se puede ver en el la Figura 66. En este caso se concatenan los valores de los nodos especificados en los nodos “path” y “composition_path” utilizando como conector el carácter especificado en el nodo “composition_separator” (en el ejemplo de la Figura 71 se utiliza el infra guion). En caso de no haber sido definido se toma como concatenación de string simple.

```

<!-- Datos necesarios para identificar una entidad en los datos geo referenciados -->
<destiny-path>eeip:manzanas</destiny-path>
<destiny-key>
  <key-search-criteria>
    <type>composition</type>
    <path>eeip:gid</path>
    <composition_path>eeip:carpeta_ca</composition_path>
    <composition_separator>_</composition_separator>
  </key-search-criteria>
</destiny-key>
<!-- Datos necesarios para identificar una entidad en los datos geo referenciados -->
<source-path>bdta:InfoCalles</source-path>
<source-key>
  <key-search-criteria>
    <type>attribute</type>
    <path>id</path>
  </key-search-criteria>
</source-key>

```

Figura 71 - Configuración de mapeo

<pre> <gml:featureMember> <eeip:manzanas fid="manzanas.31"> <eeip:gid>31</eeip:gid> <eeip:carpeta_ca>1155</eeip:carpeta_ca> <eeip:geom> <gml:MultiPolygon srsName="http://www.opengis.net/gml/srs/epsg.xml#32721"> <gml:polygonMember> <gml:Polygon> <gml:outerBoundaryIs> <gml:LinearRing> <gml:coordinates xmlns:gml="http://www.opengis.net/gml" 567537.12407724,6137939.08142604 567537.28155696,613792. </gml:coordinates> </gml:LinearRing> </gml:outerBoundaryIs> </gml:Polygon> </gml:polygonMember> </gml:MultiPolygon> </eeip:geom> </eeip:manzanas> </gml:featureMember> </pre>	<pre> <ns2:infoCobroBPS2Response xmlns:ns2="http://webservices.samples.jboss.org/"> <bdta:InfoManzana id="31_1155" servicios="DHL Cobro"> <horario dia="Lunes a viernes" hora="9:00hs a 18:00hs"/> </bdta:InfoManzana> </ns2:infoCobroBPS2Response> </pre>
---	--

Figura 72 - Ejemplo de datos con mapeo compuesto

INSTITUTO DE COMPUTACIÓN
FACULTAD DE INGENIERÍA
UNIVERSIDAD DE LA REPÚBLICA

Plataforma de Integración de Información Geográfica sobre JBossESB

Anexos

Autores

Ignacio Assandri Punschke
Martín Villero Sosa
Nicolás Lasarte Bonifacio

Tutores

Bruno Rienzi
Raquel Sosa

Usuario

Laura González

1. Uso de Aplicación de Configuración

La plataforma GeoEEIP desarrollada en el proyecto de grado, cuenta con una aplicación de configuración. La cual tiene como objetivo facilitar el uso de la plataforma a usuarios que no necesitan conocimientos avanzados de JBossESB. Y además cuenta con la opción de generar los XML correspondientes de configuración en JBossESB para facilitar el uso de la plataforma por parte de desarrolladores que quieran extender la plataforma.

1.1. Configuración de Componentes Complejos

La aplicación de configuración cuenta con formularios¹ que permiten configurar y generar los Componentes Complejos desarrollados en GeoEEIP: SOAP WMS Wrapper y GeoEnricher (WMS o WFS). El resultado obtenido es: la generación de los paquetes para el *deploy* en JBossESB de dichos componentes. Estos archivos cuentan con los archivos necesarios para su funcionamiento, como archivos generados tales de configuración, transformaciones XSLT necesarias, etc.

La aplicación de Configuración es una aplicación Web que a partir de los datos introducidos en el formulario correspondiente genera los archivos necesarios para el funcionamiento del Componente Complejo, para luego crear un archivo de *deploy* del mismo. Para esto se basa en el uso de tres carpetas en la máquina virtual que acompaña esta documentación (esto se realizó para simplificar el desarrollo en el contexto del proyecto de grado, pero teniendo en cuenta la posibilidad de migrar dicha aplicación a un escenario real). Las carpetas mencionadas son:

- [HOME]\EEIP\deployFiles – Esta carpeta debe contener los archivos fuente compilados de la plataforma así como el archivo de configuración de destinos del Componente GeoRouter.
- [HOME]\EEIP\configInput – Esta carpeta debe contener los archivos de configuración referido en el formulario (por ejemplo archivo de configuración para el Mapeo de Datos). Luego de generado el componente, en esta carpeta se encontrará el contenido del archivo de *deploy* (código compilado, archivos generados, configuración JBossESB, etc).
- [HOME]\EEIP\configOutput – Esta carpeta oficia como repositorio de los archivos de *deploy* generados.

¹ Ver Sección 4.4 del documento “Implementación de un ESB-Based Enterprise Integration Platform (GeoEEIP)”

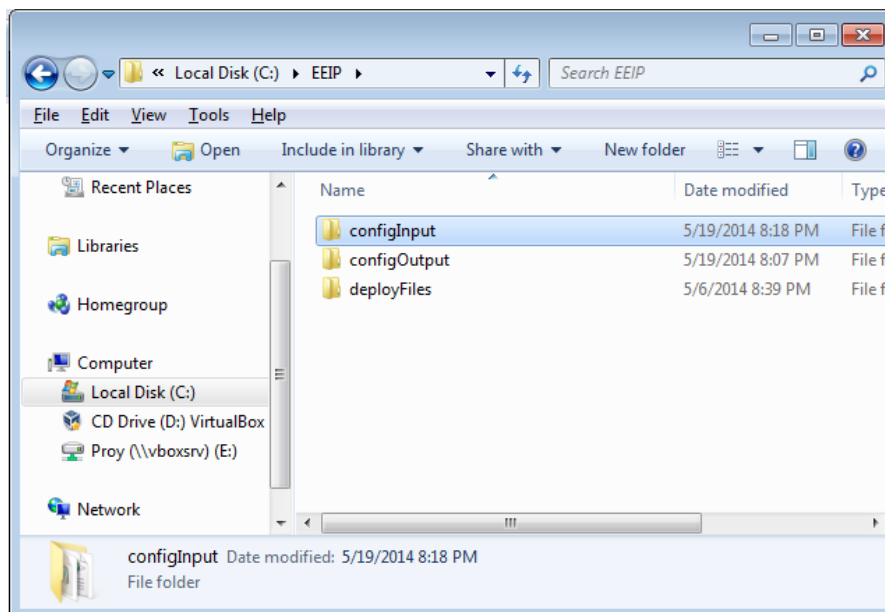


Figura 1 - Carpetas para generación de Configuración

1.2. Configuración de Componentes Básicos

La aplicación de configuración cuenta con la funcionalidad de generar los XML de configuración de JBossESB para usuarios técnicos que deseen realizar desarrollo en utilizando la plataforma GeoEEIP o extender la misma. Esto se realiza mediante los formularios² previamente mencionados, en los mismos se encuentra un campo el cual presenta la configuración generada. Para los casos de Componentes Básicos que generan archivos, por ejemplo GeoFeatureEnricher que genera la transformación XSLT para el enriquecimiento de datos, estos utilizan la misma estructura de carpetas descrita en la sección anterior.

² Ver Sección 4.4 del documento “Implementación de un ESB-Based Enterprise Integration Platform (GeoEEIP)”

2. Extensión de la plataforma

2.1. Anexo extensión de del Plataforma GeoEEIP

En el presente anexo se brindan los principales lineamientos para desarrolladores que requieran extender las funcionalidades de la plataforma, con el fin de obtener una solución que se ajuste a su problemática. La extensión de la plataforma está orientada a usuarios con conocimiento de JBossESB.

2.2. Generalidades

Se deja disponible a los usuarios la librería GeoEEIP, mediante la cual se puede extender la plataforma. Esta extensión debe seguir las pre y post condiciones expresadas en la Sección 4 del documento “Implementación de un ESB-Based Enterprise Integration Platform (GeoEEIP)”.

2.3. Extensión de la Plataforma

La plataforma puede ser extendida agregando nuevos Componentes a la misma. Para agregar Componentes se deben seguir las reglas planteadas por las precondiciones de los ya existentes.

Para extender los componentes existentes, se debe tener en cuenta como estos están implementados, ya que estos pueden ser acciones o servicios de JBossESB. Los componentes implementados como acciones, se extienden heredando de la clase correspondiente a dicha acción. Los componentes implementados como servicios, se extienden mediante la extensión de sus acciones y/o modificando su flujo de procesamiento.

A continuación se presenta cada uno de los Componente, con los aspectos relevantes para su extensión.

GeoEntryPoint

Tipo Acción
Paquete eeip.mechanisms.basic
Clases GeoEntryPoint

Configuración

```
<action class="eeip.mechanisms.basic.GeoEntryPoint" name="GeoEntryPoint" />
```

GeoProxy

Tipo Acción
Paquete eeip.mechanisms.basic
Clases GeoProxy

Configuración

```
<action class="eeip.mechanisms.basic.GeoProxy" name="GeoProxy">
  <property name="imsName" value="http://localhost:8081/geoserver/wms"/>
</action>
```

GeoRouter

Tipo Acción
Paquete org.jboss.soa.esb.actions
Clases ContentBaseRouter

Configuración

```
<action class="org.jboss.soa.esb.actions.ContentBasedRouter" name="GeoRouter">
  <property name="cbrAlias" value="Regex"/>
  <property name="ruleSet" value="/regex-georouter-rules.properties"/>
  <property name="ruleReload" value="true"/>
  <property name="object-paths">
    <object-path esb="properties.query_string"/>
  </property>
</action>
```

```

        <property name="destinations">
            <route-to destination-name="geoIdentity" service-category="geoIdentity" service-
name="geoIdentity_"/>
            <route-to destination-name="geoCapabilitiesAdapter" service-
category="geoCapabilitiesAdapter" service-name="geoCapabilitiesAdapter_"/>
            <route-to destination-name="geoFeatureInfoEnricher" service-
category="geoFeatureInfoEnricher" service-name="geoFeatureInfoEnricher_"/>
        </property>
    </action>

```

GeoCapabilitiesAdapter

Tipo Acción
Paquete eeip.mechanisms.basic
Clases GeoCapabilitiesAdapter
Configuración

```

<action class="eeip.mechanisms.basic.GeoCapabilitiesAdapter"
name="GeoCapabilitiesAdapter"/>

```

GeoFeatureTypeEnricher

Tipo Acción
Paquete eeip.mechanisms.basic
Clases GeoFeatureTypeAdapter
Configuración

```

<action
class="eeip.mechanisms.basic.GeoFeatureTypeAdapter" name="GeoFeatureTypeAdapter_TypeAdapterEjem
plo">
    <property name="featureTypeAdapterSchema" value="dataType.xsd"/>
    <property name="featureTypeAdapterXpaths" value="dataTypePaths.config"/>
</action>

```

GeoFeatureInfoEnricher

Tipo Servicio
Clases org.jboss.soa.esb.actions.StaticRouter
org.jboss.soa.esb.actions.Aggregator
org.jboss.soa.esb.smooks.SmooksAction
eeip.mechanisms.basic.RestoreMessage
eeip.mechanisms.basic.GeoFeatureInfoAggregator
eeip.mechanisms.basic.GeoFeatureInfoEnricher

Configuración

```

<jbossesb
xmlns="http://anonsvn.labs.jboss.com/labs/jbossesb/trunk/product/etc/schemas/xml/jbossesb-
1.3.1.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:schemaLocation="http://anonsvn.labs.jboss.com/labs/jbossesb/trunk/product/etc/schem
as/xml/jbossesb-
1.3.1.xsd http://anonsvn.jboss.org/repos/labs/labs/jbossesb/trunk/product/etc/schemas/xml/jbosses
b-1.3.1.xsd">
    <providers>
        <jms-provider connection-factory="ConnectionFactory" name="SOAPWrapperMQ">
            <jms-bus busid="route_to_geoFeatureInfoEnricher">
                <jms-message-filter dest-
name="queue/queue_route_to_getFeatureInfoEnricher_" dest-type="QUEUE"/>
            </jms-bus>
            <jms-bus busid="route_to_specificGeoFeatureInfoEnricher__EjemploFuente">
                <jms-message-filter dest-
name="queue/queue_route_to_getSpecificFeatureInfoEnricher__EjemploFuente" dest-type="QUEUE"/>
            </jms-bus>

```

```

        <jms-bus busid="AggregatorListenQueue">
            <jms-message-filter dest-
name="queue/queue_route_to_getAggregatorListenQueue_" dest-type="QUEUE"/>
        </jms-bus>
    </jms-provider>
</providers>
<services>
    <service category="geoFeatureInfoEnricher" description="route to geo Feature Info"
name="geoFeatureInfoEnricher_">
        <listeners>
            <jms-listener busidref="route_to_geoFeatureInfoEnricher"
name="route_to_geoFeatureInfoEnricher"/>
        </listeners>
        <actions mep="RequestResponse">
            <action class="org.jboss.soa.esb.actions.StaticRouter"
name="routeAction">
                <property name="destinations">
                    <route-to service-category="enricherSource" service-
name="EjemploFuente_"/>
                </property>
            </action>
        </actions>
    </service>
    <service category="enricherSource" description="Static route to EjemploFuente"
name="EjemploFuente_">
        <listeners>
            <jms-listener
busidref="route_to_specificGeoFeatureInfoEnricher__EjemploFuente"
name="route_to_specificGeoFeatureInfoEnricher__EjemploFuente"/>
        </listeners>
        <actions mep="RequestResponse">
            <action class="eeip.mechanisms.basic.GeoFeatureInfoEnricher"
name="GeoFeatureInfoEnricher">
                <property name="sourceName" value="EjemploFuente"/>
                <property name="wsdl" value="http://sample?wsdl"/>
                <property name="SOAPAction" value="MetodoSOAPEjemplo"/>
                <property name="ws_query_params_xpath" value=""/>
                <property name="get-payload-location"
value="ws_parameters_in_message"/>
            </action>
            <action class="org.jboss.soa.esb.actions.StaticRouter"
name="routeToAggregator">
                <property name="destinations">
                    <route-to destination-name="aggregatorService" service-
category="Aggregation" service-name="EnrichmentAggregador_"/>
                </property>
            </action>
        </actions>
    </service>
    <service category="Aggregation" description="Aggregates messages"
name="EnrichmentAggregador_">
        <listeners>
            <jms-listener busidref="AggregatorListenQueue"
name="AggregatorListenQueue"/>
        </listeners>
        <actions mep="RequestResponse">
            <action class="org.jboss.soa.esb.actions.Aggregator"
name="Aggregator">
                <property name="timeoutInMillies" value="600000"/>
            </action>
            <action class="eeip.mechanisms.basic.GeoFeatureInfoAgregador"
name="EjemploFuente_assemble">
                <property name="sourceName" value="EjemploFuente"/>
            </action>
            <action class="org.jboss.soa.esb.smooks.SmooksAction"
name="EjemploFuente_transform">

```

```

res_EjemploFuente.xml"/>
    <property name="smooksConfig" value="smooks-
    <property name="get-payload-location" value="ims_response"/>
    <property name="set-payload-location" value="ims_response"/>
    <property name="mappedContextObjects"
value="java.lang.String"/>
    </action>
    <action class="eeip.mechanisms.basic.RestoreMessage"
name="RestoreMessage"/>
    </actions>
  </service>
</services>
</jbossesb>

```

3. Instalación

A continuación se plantean los pasos a seguir, para instalar un ambiente funcional que permita utilizar la Plataforma GeoEEIP. En el mismo se consideran dados el IMS y los Web Services de negocio.

1. Instalar JBossAS³ 6.1.0 Final
2. Instala JBossESB⁴ 4.12
3. Disponer de un IMS
4. Disponer de los Web Services a utilizar como fuentes empresariales
5. Desplegar la aplicación de Configuración
6. Crear carpetas necesarias para la configuración
7. Extraer el archivo “GeoEEIP.rar” en la carpeta “home”.
8. Seguir los pasos del Configurador descritos en este Anexo, para generar los archivos de *deploy*.

Los archivos necesarios para la instalación, se encuentran en la distribución de la Plataforma GeoEEIP.

³ <http://www.scribd.com/doc/76108813/Jboss-6-1-0-Install>

⁴ **JBoss ESB Beginner's Guide,**

Release Date : January 2012

ISBN : 1849516588

ISBN 13 : 9781849516587

Author(s) : Len DiMaggio, Kevin Conner, Magesh Kumar B, Tom Cunningham