Programa de Desarrollo de las Ciencias Básicas (PEDECIBA), Área Informática

Instituto de Computación, Facultad de Ingeniería, Universidad de la República

Plataforma Específica de Dominio basada en Enterprise Service Bus para Integración de Servicios Geoespaciales

Tesis de Maestría en Informática

Ing. Bruno Rienzi Barrella

Director de Tesis: **Dr. Ing. Raúl Ruggia**

Directores Académicos: **Dra. Ing. Adriana Marotta** y **Dr. Ing. Raúl Ruggia**

Mayo, 2016

Montevideo, Uruguay

Resumen

Los Sistemas de Información Geográfica (GIS) se han convertido en los últimos años en sistemas casi omnipresentes, como resultado de varios factores, entre los que se destacan la abundancia de datos georreferenciados, la alta disponibilidad de dispositivos de geolocalización y la incorporación de tecnologías Web, dando origen a los llamados Web GIS. Un Web GIS debe ser capaz de responder a escenarios en donde un gran número de usuarios utilizan alguna funcionalidad que debe ser resuelta por un GIS, desde la simple visualización de un mapa dinámico hasta la aplicación de complejos modelos matemáticos.

Para viabilizar este tipo de sistemas dentro de un marco de interoperabilidad, el Open Geospatial Consortium (OGC) ha definido una arquitectura orientada al procesamiento e intercambio de información geográfica distribuida, basada en su propio conjunto de estándares de Web services geoespaciales. Las soluciones existentes que utilizan estos estándares funcionan bien en escenarios intra-organizacionales y cuando toda la información (geográfica y no geográfica) se encuentra en la misma fuente de datos. Sin embargo, estas soluciones no son útiles cuando esa información debe vincularse con otra información de negocio que se encuentra en otros sistemas tanto dentro como fuera de la organización. A su vez, los estándares OGC dejan algunos aspectos sin definir, por lo que deben ser resueltos por cada aplicación mediante soluciones a medida. Dentro de estos aspectos se encuentran los mecanismos de seguridad, transaccionalidad y orquestación de servicios, que son indispensables en aplicaciones de gran porte, como las que encontramos dentro de las Plataformas de Gobierno Electrónico. A nivel técnico, los aspectos mencionados son cubiertos adecuadamente por los llamados estándares avanzados de Web services (WS-*), pero estos tampoco pueden ser aprovechados directamente por los Web services de OGC debido a la incompatibilidad entre los protocolos que utilizan.

Una alternativa ampliamente utilizada para solucionar problemas de integración e interoperabilidad de sistemas de mediano y gran porte son las llamadas tecnologías de *middleware*, dentro de las que se destaca el Enterprise Service Bus (ESB). Un ESB, permite resolver, entre otros, diferencias en protocolos de comunicación, formatos de mensajes, y modelos de interacción, proveyendo mecanismos que implementan patrones de mediación, como por ejemplo, transformaciones y ruteo inteligente de mensajes. Si bien la introducción de un ESB podría resultar ideal para compensar algunas de las limitaciones que poseen los Web GIS actuales, sería una ardua labor que cada organización deba implementar en un ESB los mecanismos necesarios para satisfacer sus requerimientos de integración. Como solución a esta misma problemática, pero aplicadas en otros dominios, han surgido recientemente las llamadas plataformas específicas de dominio, que no son genéricas como un ESB pero pueden implementarse a partir de estos.

En esta tesis se especifica una Plataforma Específica de Dominio para el dominio GIS, basada en la utilización de un ESB, en el contexto de Gobierno Electrónico y de Servicios Geoespaciales. En términos generales, la solución propone interceptar los mensajes que ingresan al ESB y procesarlos a través de flujos de mediación, configurables, reusables y específicos para la interacción con Web services geoespaciales, para brindar un valor agregado a los mismos, en términos de enriquecimiento de mensajes, adaptación de protocolos y mecanismos de seguridad, entre otros.

La plataforma propuesta se especifica formalmente con el método Event-B y su factibilidad técnica se verifica en base a la implementación de un prototipo y la realización de pruebas de performance, en las que se evalúan particularmente indicadores de tiempo de respuesta.

Palabras clave: plataforma específica de dominio, enterprise service bus, sistema de información geográfica, Web services geoespaciales, middleware.

Tabla de contenido

1 Introducción

Los Sistemas de Información Geográfica (GIS) han sido definidos tradicionalmente como sistemas que integran hardware, software y datos para capturar, gestionar, analizar y desplegar todo tipo de información georreferenciada [1][2]. Actualmente, los GIS dan soporte a aplicaciones de muy diverso porte, desde pequeñas aplicaciones para teléfonos celulares hasta complejos sistemas distribuidos que trascienden los límites de una sola organización. Dentro de ese amplio espectro, esta tesis se centra en el uso de GIS en contextos de escala empresarial y gobierno electrónico. En particular, se proponen soluciones a problemáticas comunes en dichos contextos, como es la necesidad de integrar servicios geoespaciales con sistemas empresariales tradicionales.

En este capítulo se presenta el contexto y motivación de la problemática abordada, los objetivos planteados, la solución propuesta, los aportes de la tesis y cómo se organiza el resto del documento.

1.1 Contexto y Motivación

Mirando unas pocas décadas hacia atrás, los GIS eran principalmente implementados en torno a aplicaciones de escritorio que trabajaban con datos almacenados en archivos. Este tipo de aplicaciones eran utilizadas casi exclusivamente por expertos en algún área relacionada con la información geográfica. En los últimos tiempos, sin embargo, han convergido algunos fenómenos que han llevado a un cambio sustantivo de ese panorama. Podemos señalar tres de esos fenómenos que han actuado como catalizadores para transformar esos GIS tradicionales a la forma en la que los conocemos hoy. En primer lugar, se ha comprendido más que nunca que la ubicación, de las misma forma que el tiempo, es una dimensión ubicua que comparten casi todos los datos. En segundo lugar, se ha generalizado el uso de herramientas de geolocalización (ej. GPS), lo que ha colocado a los mapas en un sinnúmero de aplicaciones y dispositivos. En tercer lugar, los GIS tradicionales se han nutrido del desarrollo de Internet y la World Wide Web, dando origen a lo que se conoce como Web GIS o Internet GIS [3].

Con este auge de la información geográfica, un Web GIS debe ser capaz de responder a escenarios en donde un gran número de usuarios, con diferentes niveles de experticia, utilizan alguna funcionalidad que debe ser resuelta por un GIS, desde la simple visualización de un mapa dinámico hasta la aplicación de complejos modelos matemáticos. Para viabilizar este tipo de sistemas dentro de un marco de interoperabilidad, el Open Geospatial Consortium (OGC) ha definido una arquitectura orientada al procesamiento e intercambio de información geográfica distribuida, basada en su propio conjunto de estándares de Web services (los Web services generalmente implementan el concepto abstracto de servicio). La existencia de estos estándares, ha permitido, por ejemplo, que las organizaciones incorporen bases de datos geográficas, servidores de mapas y clientes de mapas (visualizadores), que al seguir los estándares OGC, pueden provenir de diferentes proveedores. Estas soluciones funcionan bien en escenarios intra-organizacionales y cuando una misma base de datos geográfica posee toda la información que necesitan los usuarios. Sin embargo, estas soluciones no son útiles cuando esa información debe vincularse con otra información de negocio que se encuentra en otros sistemas tanto dentro como fuera de la organización (ej. una base de datos de clientes manejada por un sistema de CRM[1], datos que provienen de Web services de terceros, etc.). En tales casos, las soluciones existentes no incluyen mecanismos orientados a facilitar dicha integración.

Si bien existen los estándares OGC y varias implementaciones de los mismos (a través de los mencionados servidores de mapas), estos no solucionan muchos de los problemas que surgen cuando se intenta integrar los GIS con sistemas empresariales tradicionales. En primer lugar, los estándares OGC dejan algunos aspectos sin definir, por lo que deben ser definidos e implementados a nivel de cada aplicación mediante soluciones a medida. Dentro de estos aspectos, es notoria la omisión de mecanismos de seguridad, transaccionalidad, orquestación de

[1] *Customer Relationship Manager* (CRM) hace referencia a un modelo de gestión de clientes y al software que permite informatizarlo

servicios, entre otros. En segundo lugar, el desarrollo divergente de los estándares OGC y los Web services definidos por W3C ha llevado a una incompatibilidad a nivel de protocolos y lenguajes utilizados por los mismos. En particular, los Web services de OGC (OWS) no utilizan la serialización en mensajes SOAP. Este hecho, además de otras consecuencias, implica que los OWS no puedan utilizar los llamados estándares avanzados de Web services o WS-*. El poder aplicar estos estándares a los OWS permitiría subsanar las carencias mencionadas anteriormente.

Paralelamente al desarrollo acelerado de los Web GIS de los últimos años, también las llamadas tecnologías de middleware han tenido un fuerte crecimiento, proveyendo las herramientas necesarias para resolver la integración e interoperabilidad de sistemas de mediano y gran porte. Entre los sistemas que se benefician por el uso de tecnologías de middleware, son de particular interés para este trabajo las Plataformas de Gobierno Electrónico (PGE) [4]. Estas plataformas, que se caracterizan por integrar un conjunto heterogéneo de servicios y aplicaciones de diversos organismos gubernamentales, generalmente se implementan siguiendo una Arquitectura Orientada a Servicios (SOA).

Dentro de las tecnologías de middleware más recientes, se destaca el Enterprise Service Bus (ESB) como uno de los productos más completos y más apegados a los estándares actuales de la industria. Un ESB, permite resolver, entre otros, diferencias en protocolos de comunicación, formatos de mensajes, y modelos de interacción. Para lograr esto, el ESB pone a disposición del integrador, un gran número de mecanismos que implementan patrones de mediación, como por ejemplo, transformaciones y ruteo inteligente de mensajes.

Si bien la introducción de un ESB podría resultar ideal para compensar algunas de las limitaciones que poseen los Web GIS actuales, sería una ardua labor que cada organización deba implementar en un ESB los mecanismos necesarios para satisfacer sus requerimientos de integración. Para subsanar esto, en algunos dominios han aparecido las llamadas plataformas específicas de dominio [5], que al no ser genéricas, facilitan la tarea de integración de sistemas dentro de un dominio predeterminado, las que a su vez pueden implementarse sobre un ESB. En base a esto, resulta de interés contar con una plataforma de este tipo que permita trabajar en el dominio GIS.

Por otro lado, los avances tecnológicos que se han mencionado no han sido acompañados con un desarrollo comparable en los modelos conceptuales. En este sentido, se ha constatado la necesidad de contar, en primer término, con una conceptualización de un marco de trabajo que defina las características fundamentales de una plataforma de integración de servicios geoespaciales con sistemas empresariales tradicionales, orientada a aplicaciones de mediano y gran porte, en general, y a aplicaciones de gobierno electrónico en particular. Esta conceptualización a nivel informal permitiría avanzar un paso más y expresar la misma en un modelo formal, de manera de contar con un marco lógico que sea totalmente independiente de la tecnología elegida para implementar la plataforma de integración, así como de las particularidades tecnológicas de los servicios y sistemas que integre.

Estas temáticas, que han sido poco abordadas en trabajos relacionados en el área, presentan un amplio conjunto de problemas a resolver y dejan abiertas varias líneas de investigación posibles, algunas de las cuales se exploran y profundizan en este trabajo.

1.2 Objetivos

El objetivo general de esta tesis es la definición de una Plataforma Empresarial basada en ESB, que facilite integrar servicios geoespaciales con servicios de negocio de uso general. Para cumplir con ese objetivo general, se plantean los siguientes objetivos específicos:

1. Definir un marco de trabajo para la integración de servicios geoespaciales en un contexto de gobierno electrónico (*e-gov*), identificando requerimientos asociados al dominio GIS, en particular de compatibilización de protocolos, enriquecimiento de información geográfica con información de negocio y mecanismos de autenticación y autorización.

2. Especificar una Plataforma Específica de Dominio para el dominio GIS, en base a los requerimientos identificados y que permita satisfacerlos en escenarios de uso típicos, en base a componentes configurables y reusables, de mayor nivel de especificidad que los que se encuentran en una plataforma de middleware de uso general.
3. Evaluar la factibilidad técnica de la solución propuesta en base al desarrollo de prototipos y la experimentación con los mismos.

1.3 Enfoque de la Propuesta de Solución

Para lograr los objetivos planteados, este trabajo se enfoca en realizar un análisis del contexto, los escenarios y requerimientos sobre los que debe plantearse la solución, especificar una plataforma específica de dominio para el dominio GIS que resuelva esos requerimientos, y prototipar esta plataforma para evaluar la factibilidad técnica y posibles limitaciones de la misma.

La plataforma específica de dominio que propone este trabajo es una Plataforma Empresarial de Integración de Servicios Geoespaciales basada en ESB (GeoEEIP), que actúa como un intermediario entre clientes y servidores, dentro de los cuales algunos pertenecen al dominio GIS y otros son sistemas tradicionales de otro dominio.

En términos generales, la función de la Plataforma es interceptar los pedidos que llegan desde los clientes, aplicar algún mecanismo de integración geoespacial (uno o varios de una lista preconfigurada) y rutear los pedidos a los servidores finales. Con las respuestas de dichos servidores se consolida una respuesta "integrada" que la Plataforma devuelve al cliente. Esta respuesta "integrada" posee un cierto valor agregado que no podría obtenerse consultando únicamente a los servidores finales sin agregar una lógica compleja ya sea a los clientes o a los servidores finales.

En la Figura 1 se muestran los principales tipos de clientes y servidores con los que interactúa la Plataforma, en un contexto típico de gobierno electrónico. Por un lado se encuentran los componentes típicos de un Web GIS, los clientes de servicios geoespaciales y los servidores de mapas (que interactúan mediantes los OWS), y por otro, los clientes de servicios de negocio y los sistemas empresariales (que interactúan mediantes Web services SOAP generalmente). Además de estos componentes, pueden aparecen otros más, orientados a proporcionar mecanismos de seguridad (ej. autenticación) a las interacciones que pasan por la Plataforma.

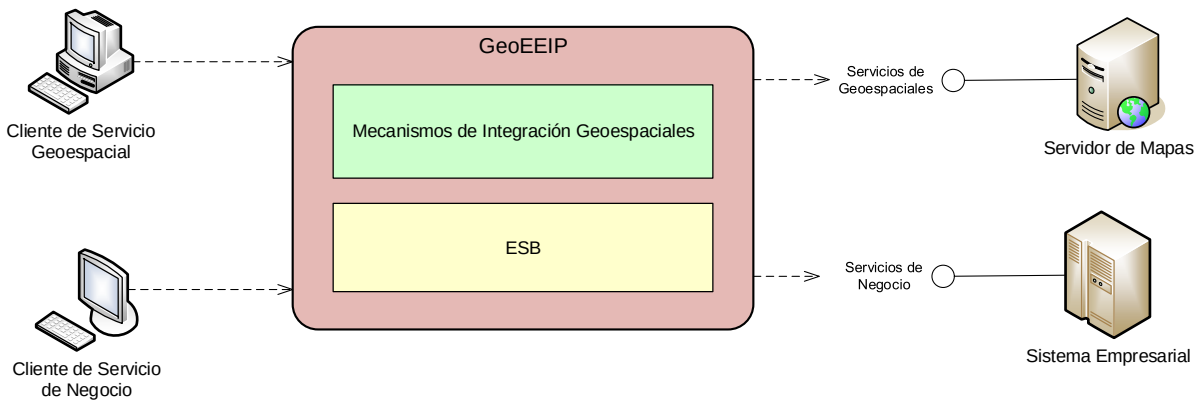


Figura 1 - Plataforma Empresarial de Integración de Servicios Geoespaciales basada en ESB

La metodología seguida en este trabajo consiste en modelar el sistema en diferentes niveles de abstracción, y pasar de un nivel a otro a través del concepto de refinamiento, siguiendo un método formal que permite realizar una construcción de este tipo. En primera instancia se define la Plataforma a un nivel de abstracción que muestra las principales funciones de integración (qué se hace), pero sin otros detalles, en particular sobre las características tecnológicas de la plataforma. Luego, mediante un proceso de refinamiento, se definen nuevos niveles, en donde

cada nuevo nivel es más concreto que el anterior. Por ejemplo, en la Figura 1 se muestra la Plataforma luego de realizar un primer refinamiento, en donde la misma se refina mediante dos componentes: el ESB y un conjunto de Mecanismos de Integración Geoespaciales. Estos mecanismos, son los que hacen que esta sea una plataforma específica de dominio y no una plataforma de middleware de uso general.

Una propuesta inicial de esta plataforma [6] fue presentada en "Fifth International Conference on Advanced Geographic Information Systems, Applications, and Services" (GEOProcessing 2013).

1.4 Aportes

Los principales aportes del trabajo realizado en esta tesis son los siguientes:

- El relevamiento y estudio de un amplio conjunto de tecnologías que se relacionan entre sí, pero que han evolucionado en forma paralela dentro de ciertos nichos, como son las de los GIS, por un lado, y las de las Plataformas de Gobierno Electrónico Orientadas a Servicios y basadas en Enterprise Service Bus, por el otro. Dicho estudio permitió detectar dificultades de integración, tanto a nivel de productos como de estándares, así como la ausencia de trabajos académicos que ataquen la problemática de integración de servicios geoespaciales dentro de las plataformas mencionadas.
- El diseño y especificación de una Plataforma Empresarial para la Integración de Servicios Geográficos, la cual permite utilizar los patrones de mediación y estándares presentes en los ESBs, para dar soporte a un conjunto extensible y combinable de requerimientos avanzados que amplían las capacidades originales de estos servicios, y que permiten atacar las dificultades de integración detectadas. Esta plataforma se enmarca dentro de las Plataformas Específicas de Dominio, al proveer un conjunto de *building blocks* específicos para el entorno GIS, que se definen como mecanismos configurables y reusables que evitan la necesidad de desarrollos ad hoc.
- La conceptualización de la plataforma en base a un modelo formal de máquinas abstractas y niveles de refinamiento, lo que permitió realizar una especificación formal en el modelo Event-B, el cual permite una especificación en refinamientos sucesivos.
- El análisis de la factibilidad técnica a través de la prototipación utilizando el producto JBoss ESB y de la realización de pruebas de performance. Concretamente, se analizan indicadores de tiempo de respuesta. comparando una solución tradicional de integración con el uso de la plataforma.

1.5 Organización del Documento

El resto de este documento se organiza de la siguiente manera.

- En el Capítulo 2 se presenta un resumen del conocimiento existente relevante para la realización de este trabajo, comenzando con conceptos generales necesarios para la comprensión del trabajo para luego investigar otros trabajos relacionados dentro del área.
- En el Capítulo 3 se analiza el contexto de aplicación en el cual se enfocará la propuesta, realizando una definición de marco de trabajo y planteando los requerimientos del sistema.
- En el Capítulo 4 se define la propuesta de solución, que consiste en la especificación de una Plataforma Empresarial para la integración de Servicios Geoespaciales, que permita satisfacer los requerimientos planteados dentro del marco de trabajo en el que se aplica.
- En el Capítulo 5 se detalla la implementación de un prototipo de la plataforma, con el fin de experimentar sobre sus características técnicas en base a la utilización de productos específicos.
- En el Capítulo 6 se presentan las conclusiones de la tesis, resumiendo el trabajo realizado y las contribuciones que este deja, y se plantean algunas reflexiones y posibilidades de trabajo a futuro.

2 Conocimiento Existente

En este capítulo se presenta un resumen del conocimiento existente que se considera relevante para comprender el punto de partida sobre el que se formula esta tesis.

En primer lugar, en la Sección 2.1 se describen conceptos generales provenientes de las áreas principales en las que se ubica este trabajo. Dentro del área de GIS, además de presentar las definiciones básicas, se introducen los conceptos de Web services geoespaciales e infraestructuras de datos espaciales. Dentro del área de Tecnologías de Middleware, se profundiza sobre los estándares de Web services de uso general y el enterprise service bus. Además, se describen otros conceptos importantes para entender el contexto de aplicación, como son las arquitecturas orientadas a servicios y las plataformas de gobierno electrónico.

En segundo lugar, en la Sección 2.2, se investigan algunos trabajos relacionados con el contexto y los objetivos planteados.

Finalmente, en la Sección 2.3, se realiza una síntesis del capítulo, reflexionando sobre las limitaciones y carencias detectadas en las soluciones existentes a la problemática que aborda este trabajo.

2.1 Conceptos Relevantes para este Trabajo

2.1.1 Organismos de Estandarización.

En este documento se recurrirá en forma frecuente a estándares que son regulados por alguno de los organismos que se describen a continuación.

- Open Geospatial Consortium (OGC)[2]. Esta organización, fundada en 1994, lidera el desarrollo de estándares para intercambio de información geográfica, en particular en lo relativo a Web services geoespaciales. Algunos de dichos estándares, como WMS, WFS, etc., serán desarrollados más adelante en este documento.
- International Organization for Standardization (ISO)[3]. Esta organización, fundada en 1947, está encargada de promover el desarrollo de normas internacionales relativas a la industria y el comercio, entre otros. El comité técnico ISO/TC 211 fue formado dentro de ISO con el fin de cubrir las áreas de información geográfica digital y geomática. Los estándares y especificaciones preparados por ISO/TC 211 pertenecen a la familia de estándares 19100. Fruto de la colaboración entre OGC e ISO/TC 211, algunos estándares de OGC son también estándares ISO, como WMS y GML.
- The World Wide Web Consortium (W3C)[4]. Esta organización, fundada en 1994 por Tim Berners-Lee, se encarga de promover estándares relativos a la Web, entre los que se encuentran: HTML, XML, XHTML, CSS, XML Schema, SOAP, WSDL, SVG, GeoRSS, etc.
- Internet Engineering Task Force (IETF)[5]. Esta organización, fundada en 1986, tiene como objetivo promover el desarrollo de los estándares de Internet, en particular a nivel de transporte, ruteo, seguridad, etc. Entre sus estándares que se mencionan en este documento, se encuentran HTTP, URL, MIME, TCP/IP, etc. Los estándares de IETF se denominan *RFC* (*request for comments*).

[2] http://www.opengeospatial.org/

[3] http://www.iso.org

[4] http://www.w3.org/

[5] http://www.ietf.org/

2.1.2 Sistemas de Información Geográfica

Un *Sistema de Información Geográfica* (*SIG*, o *GIS*, en inglés) puede ser visto como un sistema de información cuya particularidad radica en que maneja información geográfica. La *Información Geográfica (IG)* es definida en como la información que puede relacionarse con localizaciones en la superficie de la Tierra, es decir, información *georreferenciada* [7].

Existen numerosas definiciones de GIS en la bibliografía, la mayoría muy similares entre sí. En este trabajo nos basaremos en la definición de Worboys y Duckham: "Un Sistema de Información Geográfica es un sistema de información basado en computadoras que posibilita la captura, modelado, almacenamiento, intercambio, manipulación, consulta, análisis y presentación de información georreferenciada" [8].

Cabe mencionar que la sigla *GIS* también es utilizada con frecuencia (aunque en este trabajo no se usará con ese significado) para referirse a la *ciencia* de la información geográfica (*geographic information science*), la cual emerge como un nuevo campo de estudio interdisciplinario con bases en la geografía, la cartografía, la ciencia de la computación, la matemática, etc. y que se dedica al estudio de las teorías y conceptos subyacentes de la IG. [9][10]

Los GIS tradicionales, no han sido ajenos al desarrollo explosivo que ha experimentado Internet y la World Wide Web en los últimos años, dando como resultado aplicaciones GIS que son accesibles desde la Web. Este tipo de GIS se conoce como *Web GIS* y un elemento fundamental para su desarrollo han sido los Web services, como se verá en la próxima sección. Un Web GIS puede definirse como "cualquier GIS que utiliza tecnología Web para comunicar sus componentes" [3]. En un sentido más amplio, cuando un GIS utiliza servicios de Internet que van más allá de la Web, se habla de *Internet GIS* [10].

Algunas de las características principales de los Web GIS, de acuerdo a [3], son:

- HTTP es el principal protocolo utilizado.
- La arquitectura más simple es la de dos capas físicas, que consiste en un servidor y varios clientes.
- Muchas aplicaciones poseen arquitecturas de tres capas físicas, en donde se agrega a la anterior una capa de datos.
- Con la proliferación de *mashups*[6] basados en mapas, aparecen aplicaciones con más de tres capas físicas.
- Las aplicaciones Web GIS y los GIS de escritorio (*desktop GIS*) tienen un número creciente de funcionalidades comunes.

Históricamente, los GIS fueron introducidos en las organizaciones como parte de un proyecto a término específico, por lo que todos los componentes tecnológicos y los datos eran ensamblados para lo que durara el proyecto, sin prever el reuso del software, los datos o el conocimiento técnico adquirido [11]. A medida que el interés en la información geográfica fue creciendo, los GIS dejaron de ocupar esos pequeños nichos y los proyectos aislados empezaron a amalgamarse para formar un único GIS de alcance departamental, con equipos estables de expertos en el área. Esto favoreció la compartición de datos, el reuso de recursos y adopción de estándares.

La expansión de los GIS no se detuvo en el nivel departamental, sino que a medida que las organizaciones tomaron mayor conocimiento de sus capacidades, estas comenzaron a organizar algunos de sus activos, sus procesos y sus workflows en función de un GIS [11]. De esta manera, el GIS se posiciona como un sistema de información que abarca toda la empresa (*enterprise-wide*), de la misma forma que lo hace una solución de ERP (Enterprise Resource Planning) o de CRM (Customer Relationships Management), hasta ser hoy en día información imprescindible para

[6] Un *mashup* es una aplicación Web que combina y da valor agregado a contenido accesible en línea, utilizando generalmente varias APIs públicas, como la de Google Maps, Twitter, etc. [12]

muchas organizaciones a nivel corporativo. Es por esto que ESRI[7], una de las empresas líderes en el área, ha introducido el concepto de *GIS Empresarial* (Enterprise GIS), también llamado, a veces, *GIS Coorporativo* (*Corporate GIS*). Los GIS empresariales comparten muchas características con los Web GIS.

En la Tabla 1 se comparan algunos aspectos de los GIS tradicionales y los GIS empresariales.

Tabla 1- Comparación GIS tradicional vs. empresarial

	GIS Tradicional	GIS Empresarial
Almacenamiento principal	Archivos	Base de Datos
Cantidad de Usuarios	Un usuario por aplicación	Muchos usuarios por aplicación. Posibilidad de aplicaciones públicas de uso global.
Perfil del Usuario	Especialistas en GIS	Profesionales de diferentes áreas de la organización (incluyendo especialistas en GIS) y público en general
Tipos de aplicaciones	De escritorio	Web, móvil, de escritorio
Actualización de aplicaciones y datos	En cada estación de trabajo	Centralizada (se hace una vez en el servidor)
Formatos y protocolos principales	Propietarios	Estándares
Plataformas soportadas	Depende de la aplicación (ej. cierta aplicación puede correr sólo en Linux)	Para las aplicaciones Web, todas las plataformas. Para las demás aplicaciones (móviles o de escritorio), depende de la aplicación.

2.1.3 Infraestructuras de Datos Espaciales

Una *Infraestructura de Datos Espaciales* (*IDE*, o *SDI*, en inglés) es un ensamblaje funcional y bien conectado de tecnologías, personas y políticas para permitir el uso y publicación de información geográfica [2]. Debido a su naturaleza compleja (volumen, costos, participantes, normativas, etc.), las IDEs suelen tener alcance nacional y funcionar dentro de la órbita estatal. Este es el caso de la National Spatial Data Infrastructure[8] (NSDI) de Estados Unidos o la IDEuy de Uruguay. En el caso europeo, existen IDEs nacionales, como la IDE de España[9], que se circunscriben a la directiva INSPIRE[10] (2007) que regula el desarrollo de las IDEs europeas, con el fin de consolidar una única IDE europea para el año 2019.

Los elementos necesarios para construir una IDE son los siguientes [1]:

- Datos, y en particular, información geográfica. Deben aplicarse las restricciones de acceso y uso que establezca su propietario.
- Hardware y software de base.
- Metadatos, tanto de los datos (ej. autor, atributos de calidad, restricciones, etc.) como de los servicios (tipo de servicio, versión, disponibilidad, tarifas, etc.)

[7] http://www.esri.com

[8] https://www.fgdc.gov/nsdi/nsdi.html

[9] http://www.idee.es/

[10] http://inspire.ec.europa.eu/

- Tecnologías.
- Estándares, tanto de datos y metadatos (ej. familia de estándares ISO 19100) como de servicios (ej. estándares OGC).
- Acuerdos entre productores y consumidores de IG, tendientes a no duplicar ni esfuerzos ni gastos.
- Personal, que mantiene y hace funcionar los recursos informáticos (hardware y software).
- Marco organizativo y político.

Los GIS empresariales y las IDEs ofrecen el marco de trabajo de mayor relevancia para las temáticas abordadas en este trabajo.

2.1.4 Web services

La definición de *Web service* (*servicio Web*, en español) ha evolucionado a través de los años. El World Wide Web Consortium[11] (W3C), en su documento "Web services Description Requirements" de 2002 [13], realiza la siguiente definición: *"Un Web Service es una aplicación de software identificada por una URI, cuyas interfaces y formas de acceso pueden ser definidas, descriptas y descubiertas como artefactos XML, y soporta la interacción directa con otros componentes de software utilizando mensajes basados en XML que se intercambian a través de protocolos basados en Internet."*

Otra definición de W3C, pero posterior a la citada, proveniente del documento "World Services Architecture" de 2004 [14], define un Web service como "*un sistema de software designado para respaldar la interacción interoperable máquina-máquina a través de una red. Posee una interfaz descripta en un formato procesable por máquina (específicamente WSDL). Otros sistemas interactúan con el Web service en la manera prescripta por su descripción usando mensajes SOAP, típicamente transmitidos usando HTPP con serialización XML en conjunción con otros estándares relacionados con la Web."*

La primera definición plantea una clara dependencia entre los Web services y el lenguaje XML, en tanto que la segunda definición, incorpora los conceptos de SOAP y WSDL. El problema con estas definiciones, es que no tienen en cuenta los cambios considerables que han experimentado las tecnologías de Web services en los últimos años, que hacen que SOAP ya no sea la única alternativa existente para implementarlos, y que tampoco se necesite utilizar XML como único mecanismo de serialización de los mensajes.

En [3] encontramos una definición más simple, pero a su vez, más inclusiva: *"Un Web service es un programa que corre en un servidor Web y que expone interfaces programáticas a otros programas en la Web."*

2.1.4.1 Web services SOAP

SOAP[12] [15] es un protocolo para el intercambio de información estructurada en formato XML. Los mensajes SOAP poseen un elemento XML denominado *Envelope* que contiene un encabezado (*Header*) y un cuerpo (*Body*). El formato también define un mecanismo, denominado SOAP *faults*, para indicar y comunicar problemas que ocurran al procesar el mensaje.

La sección de encabezado de un mensaje SOAP es extensible y puede contener varios encabezados que se utilizan para especificar distintos tipos de información, por ejemplo, a nivel de seguridad y comportamiento transaccional [16].

[11] http://www.w3.org

[12] El significado original del acrónimo era *Simple Object Access Protocol*, pero se eliminó de la especificación en 2007.

La sección del cuerpo de un mensaje SOAP aloja el contenido del mensaje. En un contexto de Web services, este contenido se encuentra en formato XML y se ajusta a lo especificado en la descripción del Web Service que se invoca. Esta descripción, se especifica mediante el lenguaje WSDL (Web Service Descriptor Language) [17], que es un lenguaje basado en XML, para describir Web services y permitir la interacción con ellos.

Existe un tercer estándar, relacionado con los anteriores, que se denomina UDDI[13] (Universal Description, Discovery and Integration) que permite registrar y buscar Web services en directorios. En primer lugar, un proveedor de servicio utiliza UDDI para almacenar la descripción del servicio, la ubicación del servicio y las interfaces para acceder al servicio, entre otros datos. A través de estos y otros elementos, los consumidores de servicios pueden realizar búsquedas de Web services, para posteriormente ubicar al servicio y conectarse para utilizarlo. UDDI define un Web Service SOAP en sí mismo, brindando una interfaz que permite a los proveedores y consumidores realizar las tareas mencionadas sobre el directorio.

2.1.4.2 Web services REST

REST (Representational State Transfer) es un estilo de arquitectura de software introducido por Fielding en el año 2000. Este estilo está diseñado para aprovechar al máximo el protocolo HTTP (Fielding también fue uno de sus principales autores), reduciendo la complejidad de los sistemas y promoviendo su escalabilidad. [3]

Los Web services REST (también llamados RESTful) son aquellos que se adhieren a este estilo, aunque en la realidad muchos Web services que dicen ser REST no cumplen estrictamente con todos sus principios.

Los principios que debe cumplir un Web service REST son:

1. Los recursos se identifican mediante URIs. Todo es un recurso y cada recurso se identifica por una URI. Los recursos y sus URIs son organizados en una jerarquía intuitiva, predecible y fácil de comprender, sin necesidad de documentación detallada.
2. Los recursos se manipulan a través de su representación. Un recurso puede tener múltiples representaciones (formatos), tales como JSON, XML, PNG, SVG, etc. Los clientes pueden indicar en qué representación reciben o envían un recurso.
3. Se utilizan los verbos de HTTP para operaciones CRUD (crear, leer, actualizar, eliminar). Post crea un recurso, Get lee un recurso, Put actualiza o crea un recurso, Delete elimina un recurso.
4. Sin estado. Cada ciclo de pedido y respuesta es de tipo “stateless” (sin estado) o libre de contexto. Esto permite que el servidor no mantenga información de sesión para realizar una operación, mejorando la escalabilidad, robustez y performance, mediante el uso de clusters de servidores para realizar balanceo de carga y mecanismos de cache en servidores y navegadores.
5. Representaciones auto-descriptivas. Cada solicitud de un cliente y respuesta del servidor es un mensaje. Los mensajes deben ser auto-descriptivos, por lo que cada mensaje debe tener toda la información necesaria para completar la información.
6. Hypermedia como el motor del estado de la aplicación (Hypermedia As The Engine of Application State, HATEOAS). El cliente comienza por obtener la representación de un recurso mediante una URI fija conocida, y dentro de esta obtiene la hipermedia generada dinámicamente por el servidor (ej. los hyperlinks) que le permiten navegar hacia otros recursos, cambiando de estado.

Los Web services REST pueden definirse, sin ser demasiado estrictos, como Web services que trasmiten datos sobre HTTP sin una capa de mensajes adicional como SOAP. En la mayoría de las implementaciones, todos los pedidos se realizan mediante una URL y todos los parámetros se agregan a la URL. Las respuestas vienen típicamente en formato JSON o XML. [3]

[13] https://www.oasis-open.org/committees/uddi-spec/

En las implementaciones más comunes de Web services REST, todas las solicitudes y sus parámetros se envían en la URL. No se definen formatos de respuesta, pero los más comunes son JSON y XML (sin SOAP encapsulado).

2.1.4.3 Estándares avanzados WS-*

Los estándares de Web services descriptos (SOAP, WSDL, etc.), resuelven adecuadamente los requerimientos básicos para permitir la comunicación entre aplicaciones siguiendo un patrón de tipo RPC (remote procedure call) adaptado a las características de la Web. Sin embargo, estos no brindan *per se* ninguna característica para requerimientos empresariales avanzados como ser seguridad, transaccionalidad, orquestación, asincronismo, mensajería confiable, etc. [16] Es por esto que han aparecido con posterioridad diferentes propuestas, que se denominan en forma colectiva como WS-*, con la finalidad de dotar a los Web services de esas capacidades faltantes. Estos estándares avanzados han sido propuestos por diferentes empresas y organizaciones, y poseen al día de hoy diferentes grados de madurez, estandarización y adopción por parte de la industria. A continuación se resumen sólo algunos de los que se consideran más importantes o relevantes para este trabajo.

WS-BPEL

WS-BPEL[14] provee un lenguaje XML estándar para expresar procesos de negocio que consisten de funciones definidas a través de interfaces WSDL. WS-BPEL soporta tanto procesos de corta duración, como procesos de larga duración (procesos que deben esperar en cierto punto hasta que ocurra algún evento). [18]

En tiempo de diseño, los analistas de negocio utilizan herramientas de desarrollo o modelado para construir procesos de negocio. Estos procesos quedan especificados a través de WS-BPEL. Luego los implementadores conectan cada paso del proceso con implementaciones de servicio específicas. Por otro lado, en tiempo de ejecución los motores WS-BPEL controlan la ejecución de los procesos, invocando los servicios requeridos para implementarlos.

WS-Security

La especificación WS-Security[15], describe la forma de asegurar los servicios Web en el nivel de los mensajes, en lugar de en el nivel del protocolo de transferencia o en el de la conexión. Para ello, tiene como objetivo principal describir la forma de firmar y de encriptar mensajes de tipo SOAP. Las soluciones en el nivel de transporte actuales, como SSL/TLS, proporcionan un sólido cifrado y autenticación de datos punto a punto, aunque presentan limitaciones cuando un servicio intermedio debe procesar o examinar un mensaje. Por ejemplo, un gran número de organizaciones implementan un firewall que realiza un filtrado en el nivel de la aplicación para examinar el tráfico antes de pasarlo a una red interna.

Si un mensaje debe pasar a través de varios puntos para llegar a su destino, cada punto intermedio debe reenviarlo a través de una nueva conexión SSL. En este modelo, el mensaje original del cliente no está protegido mediante cifrado puesto que atraviesa servidores intermedios y para cada nueva conexión SSL que se establece se realizan operaciones de cifrado adicionales que requieren una gran cantidad de programación.

El estándar WS-Security se basa en estándares y certificaciones digitales para dotar a los mensajes SOAP de los criterios de seguridad necesarios. Se definen cabeceras y usa XML Signature para el manejo de firmas en el mensaje. La encriptación de la información la realiza mediante XML Encryption haciendo uso del intercambio de credenciales de los clientes. [18]

[14] http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html

[15] https://docs.oasis-open.org/wss/v1.1/wss-v1.1-spec-os-SOAPMessageSecurity.pdf

Otros estándares avanzados

El estándar WS-Addressing[16] se enfoca en proveer mecanismos para direccionar mensajes en forma independiente del protocolo de transporte, de manera que un sistema de mensajería puede trasmitir los mensajes a través de redes que incluyen nodos de procesamiento, firewalls, gateways, etc. Dentro de este estándar, se define, por ejemplo, un elemento "wsa:to" que permite especificar el destino del mensaje, en tanto que el elemento "wsa:action" permite especificar su semántica.

También existen estándares para garantizar transaccionalidad en las operaciones realizadas a través de Web services. Dentro de estos encontramos WS-AtomicTransaction [19] y WS-BusinessActivity [20]. El primero es utilizado en plataformas basadas en SOA para garantizar la consistencia de los datos mediante 2PC (2 phase commit). El segundo es utilizado para asegurar la consistencia en procesos de negocio de larga duración, en los que es inaceptable realizar un bloqueo de los recursos como lo exige el 2PC. En este caso, el *rollback* no existe como tal sino que utiliza una operación de compensación que "deshace" los cambios cuando la transacción es abortada.

2.1.4.4 Comparación SOAP vs. REST

Como se ha visto en las secciones anteriores, los Web services REST y SOAP poseen grandes diferencias, por lo que uno u otro puede ser más apropiado de acuerdo al tipo de aplicación [21]. Un ejemplo bien conocido, es el de los Web services de e-commerce de Amazon[17], que ofrecen la misma funcionalidad tanto con SOAP como con REST, y que registran un 85% de uso de REST contra un 15% de uso de SOAP[18].

REST es llamado "la línea de comandos de la Web," ya que alcanza con poner la URL en el navegador (como si de una página Web se tratara) y el resultado aparece inmediatamente. [3]

Tabla 2 se muestra cómo se comparan los Web services SOAP y REST de acuerdo a ciertos puntos.

Tabla 2 – Comparación SOAP vs. REST

	WS SOAP	WS REST
Operaciones	Dependientes del negocio	Métodos HTTP. También operaciones dependientes del negocio con método Get.
Definición de interfaz	WDSL (estándar W3C)	Por documentación. Propuestas no estandarizadas (WADL, RSDL)
Servicio de descubrimiento	UDDI (estándar W3C)	No
Protocolo de transporte	HTTP y otros	HTTP
Eficiencia	Baja (mensajes grandes, *parsing* lento)	Alta (mensajes livianos)
Seguridad, Transaccionalidad, Orquestación, etc.	Robusta. Estándares WS-*	Poco robusta o inexistente.
Soporte en middleware empresarial y SOA	Amplio	Escaso

[16] https://www.w3.org/TR/2006/REC-ws-addr-core-20060509/

[17] http://www.amazon.com

[18] http://archive.oreilly.com/pub/wlg/3005

Mercado	Infraestructuras de IT grandes, B2B	Infraestructuras de IT de pequeñas a grandes, mass market
Desarrolladores	Profesionales	Profesionales y no profesionales

2.1.5 Web services geoespaciales

Una categoría aparte de los estándares de Web services la componen los llamados Web services geoespaciales. Se trata de Web services especializados en el manejo de IG, que en su mayoría han sido definidos y estandarizados por OGC, por lo que se los conoce como OGC Web services u OWS. El diseño de estos estándares sigue los lineamientos establecidos en la Arquitectura Abstracta [22] y en la especificación de características comunes para todos ellos [23][24]. Dentro de las características comunes de todos los OWS podemos señalar las siguientes:

- Existen operaciones obligatorias y pueden existir operaciones opcionales.
- Definen la operación GetCapabilities que permite consultar los metadatos del servicio (conjunto de datos que ofrece, operaciones implementadas, valores soportados para ciertos parámetros, etc.)
- Están basados en los estándares más ubicuos de Internet, como ser HTTP y tipos MIME.
- Las operaciones son, en su gran mayoría, sincrónicas.
- No utilizan SOAP ni WDSL, aunque pueden implementarlos.

En las próximas secciones se profundiza en algunos de los estándares más difundidos y relevantes para este trabajo.

2.1.5.1 Web Map Service (WMS)

WMS [25] es un estándar OGC/ISO (ISO 19128) que permite obtener mapas dinámicos a partir de un conjunto de capas geográficas[19]. Un mapa para WMS es una representación de IG en un formato de imagen digital, como ser PNG, GIF, JPEG, etc.

OGC ha definido varias versiones de este estándar desde su primera versión del año 2000, siendo la más reciente la versión 1.3 de 2006. Se trata del primer OWS en ser aprobado y el más sencillo de todos, por lo que también es el más usado e implementado a nivel de productos, tanto comerciales como libres.

La especificación diferencia entre dos tipos de servicios WMS:

- WMS Básico, que soporta las operaciones obligatorias GetCapabilities y GetMap.
- WMS Consultable, que además de las anteriores soporta la operación opcional GetFeatureInfo.

Las operaciones de WMS son las siguientes:

- GetCapabilities. Esta operación permite obtener las capacidades del servicio en formato XML. Concretamente, obtiene los metadatos que describe el contenido de la información que provee el servicio, así como los valores admitidos de los parámetros con los que se realizan las solicitudes. Dentro de los metadatos, se informa de todas las capas geográficas (*layers*) que publica el servicio, y de cada capa se indica, entre otros:

[19] Se entiende por *capa geográfica* (*geographical layer*) a una colección lógica de entidades geográficas de un mismo tipo (ej. capa de lagos)

- los sistema de referencia soportados (CRS, *coordinate reference system*), basados en el estándar ISO 19111:2007[20]
- el *bounding box* (límite rectangular) de la capa, tanto en latitud/longitud (CRS:84) como en el CRS nativo de la capa, es decir, como efectivamente están almacenadas las coordenadas,
- los estilos de representación definidos para esa capa junto con la URL de la leyenda que explica el estilo (ej. un estilo puede definir que las avenidas se pintan en color rojo y las demás calles en color amarillo)
- si está marcada como *queryable*, indica que la capa es consultable (mediante GetFeatureInfo)

También en los metadatos se informa de las operaciones soportadas (GetCapabilities, GetMap, GetFeatureInfo), las URLs de cada operación (para los métodos Get y Post), y los formatos de respuesta.

- GetMap. Es la operación principal de WMS ya que es la que permite obtener el mapa. Dentro de los parámetros del pedido se encuentran, entre otros:

  - las capas que se desean dibujar y el estilo para cada capa
  - el CRS del mapa
  - el *bounding box* del mapa en coordenadas del CRS especificado
  - el formato de respuesta (ej. "image/png")
  - el ancho y alto del mapa en pixels de pantalla

  Cabe mencionar que todos los parámetros proporcionados en este pedido deben pertenecer a los valores admitidos que se indican en la respuesta del GetCapabilities, de lo contrario se generará una excepción.

- GetFeatureInfo. Esta operación brinda información descriptiva (alfanumérica) asociada a un punto del mapa. Los parámetros que se deben proporcionar son:

  - las capas que se desean consultar (deben ser de tipo *queryable*)
  - el CRS del mapa
  - el *bounding box* del mapa en coordenadas del CRS especificado
  - el formato de respuesta (ej. "text/html")
  - el ancho y alto del mapa en pixels de pantalla
  - el x e y del punto relativo a la imagen del mapa (en pixels)

  El resultado generalmente se obtiene en formato HTML o GML. La operación GetFeatureInfo está pensada como implementación de la funcionalidad *Identificar* que es común en todas las herramientas GIS. Con esta funcionalidad, el usuario hace clic en un punto del mapa y a continuación se muestra la información asociada (nombres, descripciones, datos numéricos, etc.).

Es interesante detenerse en algunos aspectos de esta operación que se mencionan a continuación. Dado que un cliente WMS no posee la capacidad de seleccionar entidades geográficas individuales dentro del mapa, ya que lo único que recibe del servidor es una imagen comprimida, es necesario que envíe toda la información del contexto, es decir, del mapa mostrado (prácticamente todos los parámetros del GetMap), más las coordenadas donde se hizo clic (relativas a la imagen). De esta manera, el servidor puede traducir esas coordenadas expresadas en pixeles a coordenadas cartográficas (en el CRS correspondiente) y así identificar el punto geográfico. A continuación, el servidor debe realizar operaciones espaciales con las capas consultadas para determinar qué entidad de cada capa

[20] http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=41126

se encuentra más cerca del punto o contiene al punto, dependiendo del *tipo geométrico*[21] de la capa, y así devolver la información descriptiva de dichas entidades.

En la Figura 2 se muestra un ejemplo de consulta sobre las capas de Ríos (polígonos), Rutas (líneas) y Puentes (puntos). Al hacer clic en un punto específico del mapa, se pueden apreciar los datos que se obtienen de una entidad de cada capa.

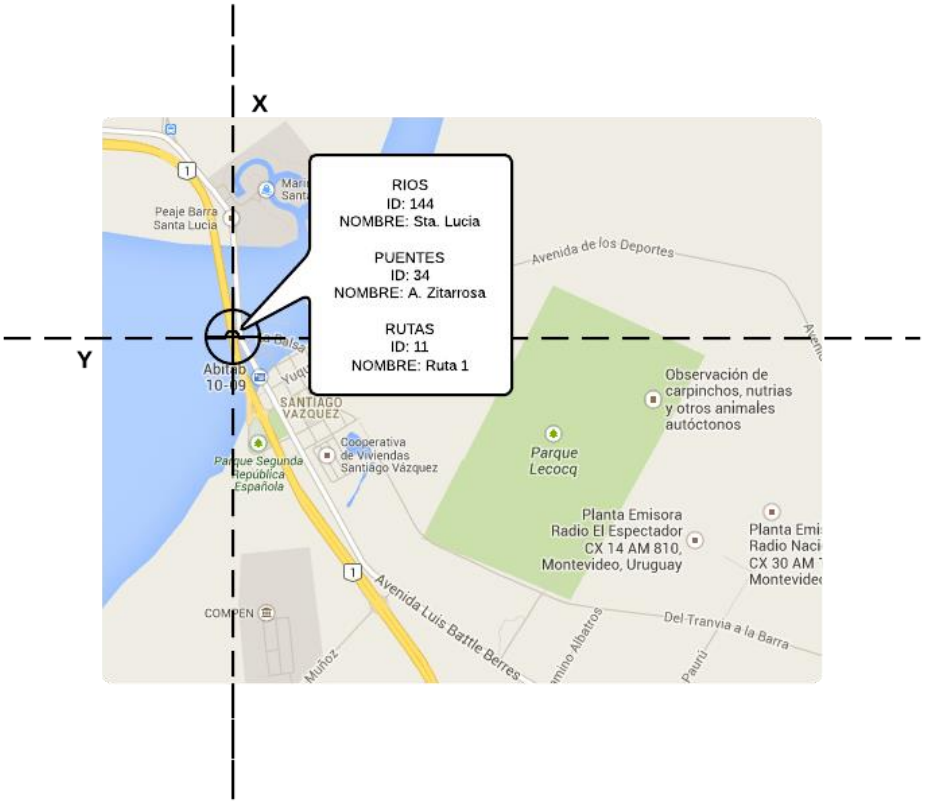


Figura 2- Consulta GetFeatureInfo

2.1.5.2 Web Feature Service (WFS)

Web Feature Service [26] es un estándar OGC que 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:

[21] Las capas geográficas vectoriales poseen entidades de un mismo tipo geométrico, por lo que se denominan capas *de puntos*, *de líneas* o *de polígonos* (en los casos más simples), según el tipo geométrico de sus entidades.

- **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*.

Las operaciones de WFS son las siguientes:

- **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[22]. 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&typenam e=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

[22] Se entiende por entidad geográfica (*geographical feature*) a todo objeto físico de interés geográfico que posee una figura geométrica georreferenciada y un conjunto de atributos descriptivos (identificación, nombre común, etc.)

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:id e_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.

Existe una versión más reciente de este estándar, la 2.0.0 [27], que a su vez es un estándar ISO (ISO 19142). La descripción se centra, sin embargo, en la versión anterior, por considerarse que en este momento es la versión más utilizada para publicar este tipo de servicios y mejor soportada a nivel de productos.

2.1.5.3 Otros OWS

Web Map Tile Service (WMTS)

WMTS [28] surgió en 2010 como un estándar alternativo a WMS orientado a mejorar la performance y escalabilidad de éste. A diferencia de WMS, en donde el cliente solicita un mapa completo en cada invocación de la operación GetMap, en WMTS el mapa es dividido como una matriz de teselas o *tiles*. Una tesela es un representación gráfica rectangular de cierta información geográfica, que cubre una extensión espacial contigua y que puede ser identificada mediante la matriz (*tile matrix*) que la contiene y un número de fila y de columna. A partir de un mapa, el servidor construye varios conjuntos de matrices de teselas (*tile matrix set*), en donde cada conjunto posee un sistema de referencia espacial (CRS) fijo, y cada matriz, una escala fija.

La gran ventaja de WMTS sobre WMS es que las matrices de teselas pueden estar pre-generadas en el servidor, para un conjunto acotado de escalas y sistemas de referencia.

Las operaciones específicas de WMTS son GetTile y GetFeatureInfo, que reciben parámetros muy similares a los de GetMap y GetFeatureInfo de WMS.

Catalogue Service – Web (CSW)

CSW es un perfil del estándar Catalogue Service de OGC [29] que permite descubrir y consultar metadatos sobre información geográfica y servicios geoespaciales. El catálogo está compuesto por un conjunto de registros (*records*) de metadatos. Los registros se codifican en XML siguiendo algún estándar de metadatos como el Dublín Core, ISO 19139 o FGDC. Este estándar define un conjunto de operaciones, entre las que se encuentran operaciones de consulta (GetRecords, GetRecordsById, etc.) y operaciones de creación, eliminación y actualización de datos (Transaction y Harvest).

La operación *GetRecords* permite hacer una consulta sobre el catálogo, solicitando que el servicio retorne todos los registros que cumplen con ciertas condiciones. Dentro de estas condiciones, se pueden especificar tanto condiciones alfanuméricas (ej. que en el título aparezca la palabra "Montevideo" o que la fecha de los metadatos pertenezca al año 2015) como condiciones espaciales (ej. que lo metadatos correspondan a una extensión geográfica comprendida dentro de otra extensión pasada como parámetro).

La operación *Transaction* es análoga a la operación homónima de WFS vista anteriormente, pero en este caso se aplica para la creación, eliminación y actualización de registros de metadatos. La operación *Harvest*, por otro lado, permite que un servicio CSW actualice su catálogo consultando periódicamente a otros servicios CSW, realizando invocaciones sincrónicas o asincrónicas de esta operación.

Web Processing Service (WPS)

WPS [30] es un estándar OGC que permite publicar procesos que trabajan con IG (llamados muchas veces, "geoprocesos") que son invocados por los clientes pero se ejecutan en el servidor. Los procesos posibles son cálculos, algoritmos y modelos que se aplican sobre IG, como ser: cálculos geoestadísticos, reproyecciones de capas geográficas, modelos estadísticos territoriales de prospectiva, modelos meteorológicos, etc. Estos procesos pueden ser muy simples o muy complejos, pueden utilizar IG de cualquier tipo (ej. vectorial, raster, etc.), pueden ejecutarse en forma sincrónica o asincrónica y pueden utilizar datos propios del servidor o accesibles en forma remota. Los procesos concretos que ejecuta un servicio WPS no están predeterminados, si no que cualquier proceso que ejecute en un servidor podría, en principio, exponerse a través de una interfaz WPS. Las lista de procesos y sus descripciones existentes en cada instancia se obtienen mediante el método *GetCapabilities*.

El estándar especifica una interfaz genérica en base a dos operaciones propias, *DescribeProcess* y *Execute*. La primera operación permite describir un proceso, con sus parámetros de entrada y formatos admitidos. La segunda operación permite que el cliente solicite la ejecución de un determinado proceso, especificando los valores de los parámetros y tipo de salida.

2.1.5.4 Web APIs para GIS

Hoy en día es muy común encontrar aplicaciones Web que muestran mapas. Podría pensarse que estas aplicaciones consumen algunos de los Web services geoespaciales vistos anteriormente. Sin embargo, muchas (tal vez, la mayoría) de estas no invocan Web services directamente, sino que utilizan alguna Web *API* (*Application Programming Interface*) geoespacial como Google Maps[23], Bing Maps[24], ArcGIS APIs[25] u OpenLayers[26]. Estas APIs permiten que los desarrolladores implementen aplicaciones Web con funcionalidades GIS en forma más rápida y sencilla [3].

Como los Web services sólo realizan funciones del lado del servidor, usualmente dejan una gran parte del trabajo de desarrollo de la aplicación al lado del browser (despliegue del mapa, control del mouse, cálculo de parámetros del mapa, parsing de respuestas, etc.) [3]. Las Web APIs corren del lado del browser y se encargan de manejar tanto la interacción con el usuario (por ejemplo, qué evento se dispara cuando el usuario mueve el puntero del mouse sobre una determinada zona del mapa) como la interacción con los Web services (por ejemplo, enviar un pedido y recibir la respuesta de un WMTS). Las APIs encapsulan la creación de un pedido y el parsing de la

[23] https://developers.google.com/maps/

[24] http://www.microsoft.com/maps/

[25] https://developers.arcgis.com/javascript/

[26] http://openlayers.org/

respuesta mediante construcciones del lenguaje de alto nivel en que están programadas (típicamente JavaScript, Flex o Silverlight).

En el caso de Google Maps y Microsoft Bing Maps, sus APIs acceden a cartografía propia de Google y Microsoft respectivamente, encapsulando Web services REST. En el caso de las ArcGIS APIs, éstas permiten acceder a cartografía publicada en cualquier servidor de mapas ArcGIS Server, así como cartografía de Google y Microsoft, a través de Web services REST. En el caso de OpenLayers, no está asociada a ningún proveedor particular, por lo que además de acceder a la cartografía de Google, Microsoft, OpenStreetMap[27] y otros, pueden acceder a cualquier capa geográfica publicada mediante WMS, WMTS o WFS.

2.1.5.5 Control de Acceso y GeoXACML

Un aspecto importante del acceso a la información geográfica tiene que ver con la seguridad que proporciona el sistema (ej. un GIS empresarial) para determinar quién puede hacer qué con qué dato, lo que se denomina, el control de acceso. Este consta de dos procesos: determinar y verificar la identidad del usuario, llamado proceso de autenticación, y restringir qué información es accesible para un usuario y determinar qué puede hacer con ella, llamado proceso de autorización. El conjunto de reglas de seguridad que otorgan derecho de acceso para la protección de los recursos se conoce también como políticas de control de acceso.

Dentro del incipiente trabajo que ha realizado OGC para dotar de seguridad a los servicios geoespaciales, el único estándar finalizado se llama GeoXACML [31], el cual es una extensión del estándar XACML (versión 2.0) de OASIS. XACML define un lenguaje de definición de políticas de control de acceso (basado en XML) y una arquitectura de referencia.

Las políticas son definidas mediante un *PolicySet* (o una única *Policy*). Un *PolicySet* puede contener otros *PolicySet* como también puede contener más de una Policy. Cada Policy, a su vez, puede contener una o varios *Rule*. Dado que para una misma consulta podrían aplicar más de una política o regla, llegando a contradicciones, existen los llamados algoritmos de combinación de políticas y reglas que permiten llegar a una única decisión. El conjunto de políticas es llamado Repositorio de Políticas.

El lenguaje de pedido/respuesta especifica la interacción entre los componentes de la arquitectura de referencia (Figura 1), definiendo dos tipos de mensaje XML, *Request* y *Response*). El *Request* es simplemente un conjunto de atributos que contienen información del pedido, conteniendo un *Subject*, con los atributos correspondientes a la entidad que intenta realizar una acción sobre los recursos, *Resource* que permite identificar los recursos involucrados, *Action*, que identifica la acción a realizar, y *Enviroment*, que contiene información de contexto y es la única opcional. El Response es la representación del resultado de una evaluación. Cada resultado contiene los siguientes datos: *Decision* (*Permit*, *Deny*, *Indeterminate* o *Not Applicable*), *Status* (si la evaluación falló o no) y de manera opcional uno o más *Obligation* (Acciones que debe tomar el PEP antes de permitir o denegar el acceso).

En la Figura 3 se muestra la arquitectura de referencia. En primera instancia el Usuario realiza un pedido para realizar una acción sobre uno o más Recursos. En esta arquitectura, el usuario no accede directamente a los recursos, sino que se agregan dos intermediarios. El PEP (Policy Enforcement Point) recibe el pedido, y se lo envía al PDP (Policy Decision Point) utilizando el lenguaje descrito anteriormente. El PDP, basándose en el pedido y las políticas existentes en el Repositorio de Políticas, evalúa si la acción puede realizarse o no. En caso afirmativo el PEP envía el pedido al servidor de Recursos y la respuesta de éste será devuelta al usuario, en forma transparente. En caso negativo el PEP retornará un mensaje con una respuesta en blanco.

GeoXACML mantiene esta arquitectura y sólo cambia en lo que a definición de políticas respecta, por lo tanto las modificaciones se producen únicamente a nivel del PDP. Las extensiones introducidas tienen que ver con la

[27] http://www.openstreetmap.org

incorporación de tipos de datos geoespaciales, como *Point*, *LineString*, *Polygon*, etc. (tomados de GML) y operaciones que utilizan estos tipos de datos, como *Touches*, *Crosses*, *Contains*, etc.

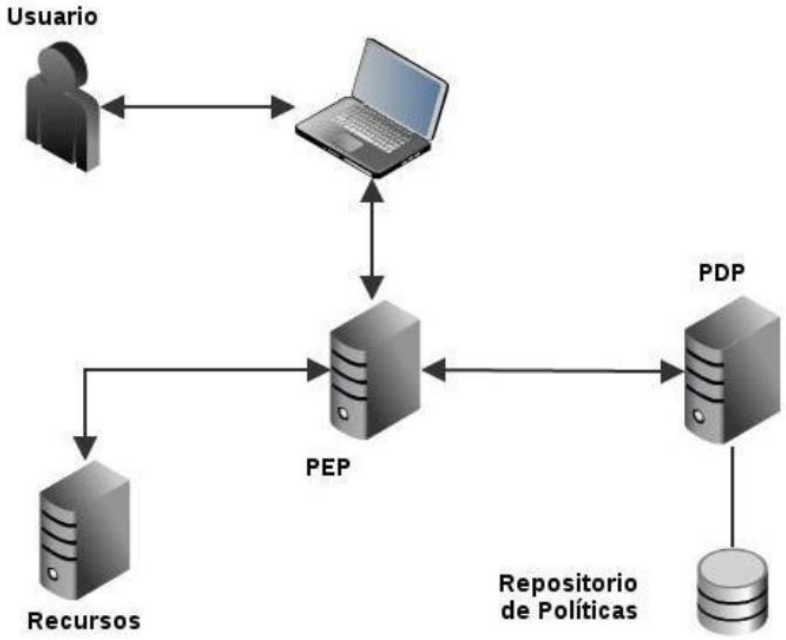


Figura 3 - Arquitectura de Referencia de XACML

2.1.6 Enterprise Service Bus: Middleware para Arquitecturas Orientadas a Servicos

En el contexto de esta tesis, surge claramente la relación entre el concepto de SOA y el concepto de IDE. Como se afirma en [1], una IDE es una aproximación a SOA, en la que las interfaces estandarizadas de servicios geoespaciales son la clave para que estos puedan interoperar.

En los últimos años se ha visto el surgimiento y afianzamiento de varias tendencias tecnológicas, como los Web services, las Arquitecturas Orientadas a Servicios, el middleware de Integración de Aplicaciones Empresariales (EAI) y el B2B (Business-to-business). Estas tecnologías han tratado de mejorar los resultados e incrementado el valor de los procesos de negocios integrados. El ESB es un nuevo acercamiento a la integración de aplicaciones, que toma ideas de algunas de esas tecnologías, pero que promete una infraestructura de integración altamente distribuida y débilmente acoplada. [32]

Un Enterprise Service Bus (ESB), según Chappel [32] 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 confiable, la interacción de un gran número de aplicaciones diversas a través de empresas extendidas y con integridad transaccional."* El concepto de *"empresas extendidas"* refiere a organizaciones y sus socios de negocio, a los cuales separan límites tanto de negocio como físicos. En este contexto, incluso las aplicaciones que pertenecen a una misma corporación pueden estar separadas geográficamente, por *firewalls* corporativos y por políticas de seguridad interdepartamentales.

En un ESB, aplicaciones y servicios dirigidos por eventos se encuentran ligados pero débilmente acoplados, lo que les permite operar independientemente pero aportando valor a una función de negocio más amplia. El ESB es la columna vertebral de la implementación de una SOA. [32]

Si bien se define habitualmente al ESB como una plataforma, también es considerado por algunos autores, dentro de la Arquitectura de Software, como un patrón o estilo arquitectónico, basado en el patrón *Bus* (*de Mensajes*) y

orientado a la comunicación e integración de sistemas [33] [34] [35]. Por otro lado, existen productos de software que se rotulan como *ESB*, y que ofrecen un conjunto de funcionalidades en común y otras que los diferencian. En el ámbito del software comercial, muchos de estos productos tienen su base en soluciones de EAI anteriores. [34]

Para poder soportar la variedad de patrones de interacción que se requieren en una SOA (ej. pedido/respuesta, publicar/subscribir, eventos), el ESB debe manejar en una sola plataforma los principales patrones de integración empresarial: [33]

- Arquitectura orientada a servicios (SOA): las aplicaciones se comunican invocando servicios reusables y con interfaces explícitas y bien definidas. La ejecución de un servicio puede resolverse internamente mediante el envío de mensajes o la generación de eventos.
- Arquitectura dirigida por mensajes (*message-driven architectures*): las aplicaciones envían y reciben mensajes con el ESB como intermediario.
- Arquitectura dirigida por eventos (event-driven architectures): las aplicaciones generan y consumen eventos sin visibilidad directa entre ellas.

Las capacidades que debe brindar un ESB varían según los diversos autores o proveedores de software que se consulten. En la Tabla 3, basada en [33], se muestra un conjunto de capacidades categorizadas que debe proveer un ESB. En [33] y [34] también se identifican algunas capacidades dentro de estas que se consideran la base mínima que debe proveer una solución de integración para ser considerada un ESB. En las siguientes secciones se detallan algunos de los patrones de mediación que se consideran más relevantes para este trabajo y que están integrados en todos los productos de ESB.

Tabla 3 -Capacidades de un ESB

Categoría	Capacidades	Comentarios
Comunicación	- Servicios Virtuales - Ruteo - Direccionamiento (addressing) - Estilos de mensajería (pedido/respuesta, publicar/subscribir) - Eventos - Protocolos de transporte (ej. HTTP, HTTPS, etc.)	Provee transparencia de ubicación y soporta sustitución de servicios.
Integración	- Adaptadores para aplicaciones y sistemas legados - Mapeo de servicios - Transformación de protocolos - Enriquecimiento de datos - Conectividad con servidores de aplicaciones - Conectividad con EAI - Conectividad con bases de datos	Soporta integración en ambientes heterogéneos y sustitución de servicios.
Interacción de Servicios	- Definición de interfaz de servicio (ej. WSDL)	Soporta principios de SOA, separación de servicios y su implementación.

	- Modelos de mensajería para servicios (ej. SOAP, REST, etc.) - Sustitución de implementación de servicios - Directorio y descubrimiento de servicios.	
Calidad de Servicio	- Transacciones (ej. transacciones atómicas, compensaciones, WS-Transaction, etc.) - Entrega asegurada (ej. WS-ReliableMessaging)	
Seguridad	- Autenticación - Autorización - No repudio - Confidencialidad - Estándares de seguridad	
Nivel de Servicio	- Performance - Throughput - Disponibilidad - Mediciones continuas especificadas por acuerdos o contratos	
Procesamiento de mensajes	- Lógica de codificación - Lógica basada en contenido - Transformaciones de datos y mensajes - Agregación de mensajes y servicios - Validación - Intermediación - Mapeo de objetos - *Store and forward*	
Administración y autonomía	- Administración - Registro de servicios - *Logging* - Medición - Monitoreo - Integración con sistemas de administración - Auto-monitoreo y auto-administración	Provee un punto de control para direccionamiento y nombramiento de servicios.
Modelado	- Soporte para diversos modelos de objetos	

	- Herramientas de desarrollo e instalación	
Inteligencia de Infraestructura	- Reglas de negocio - Comportamiento basado en políticas para seguridad, calidad y nivel de servicio (ej. WS-Policy) - Reconocimiento de patrones	

2.1.6.1 Patrones utilizados en los ESB.

En base a la experiencia práctica con productos tipo ESB, se han documentado distintos patrones que describen la utilización del ESB a varios niveles. Estos patrones son denominados por algunos autores como Enterprise Integration Patterns [36].

En particular, en esta sección se presentan un conjunto de patrones de conectividad, que apuntan a especificar estilos de integración generales utilizando un ESB, y un conjunto de patrones de mediación, que especifican operaciones (o secuencias de operaciones) de mediación comúnmente utilizadas para procesar mensajes en el ESB.

Servicios Virtuales

En [37] se realiza una clasificación de los patrones de conectividad que deberían formar parte de un ESB. Dentro de estos interesan particularmente los patrones de servicios virtuales, que son los que permiten lograr el acoplamiento débil entre servicios que debe existir en una SOA, agregando niveles de indirección.

Dentro de los patrones de servicios virtuales, los autores identifican varios casos, dentro de los cuales el más básico, y que generalmente sirve como ejemplo de lo que es un servicio virtual, es el Simple Service Proxy. Este patrón permite publicar un servicio existente dentro del ESB, por lo que las solicitudes llegan al servicio virtual del ESB en primera instancia y recién en determinado punto este las envía al servicio real. De esta manera, se tiene un punto de mediación en donde se pueden realizar operaciones de registro, seguridad, manejo de errores y alertas, administración de tráfico, cargos por uso del servicio, mediciones etc.

Dos de las características importantes de este patrón son que da soporte a la gobernanza en SOA, cuando se asegura que todos los servicios son accedidos únicamente a través del ESB, y que oculta la estructura de servicios interna de una organización

Ruteo

El ruteo o ruteo inteligente[28] es un patrón de diseño de SOA [38] que es implementado por los ESB y que consiste en determinar el destino de un mensaje en forma dinámica, de acuerdo a ciertos factores. De acuerdo a qué factores se tienen en cuenta, se habla de: ruteo basado en contenido, ruteo basado en contexto, ruteo basado en itinerario, etc.

En [36] se especifican diez patrones de ruteo, que se muestran en la Figura 4.

[28] http://soapatterns.org/design_patterns/intermediate_routing

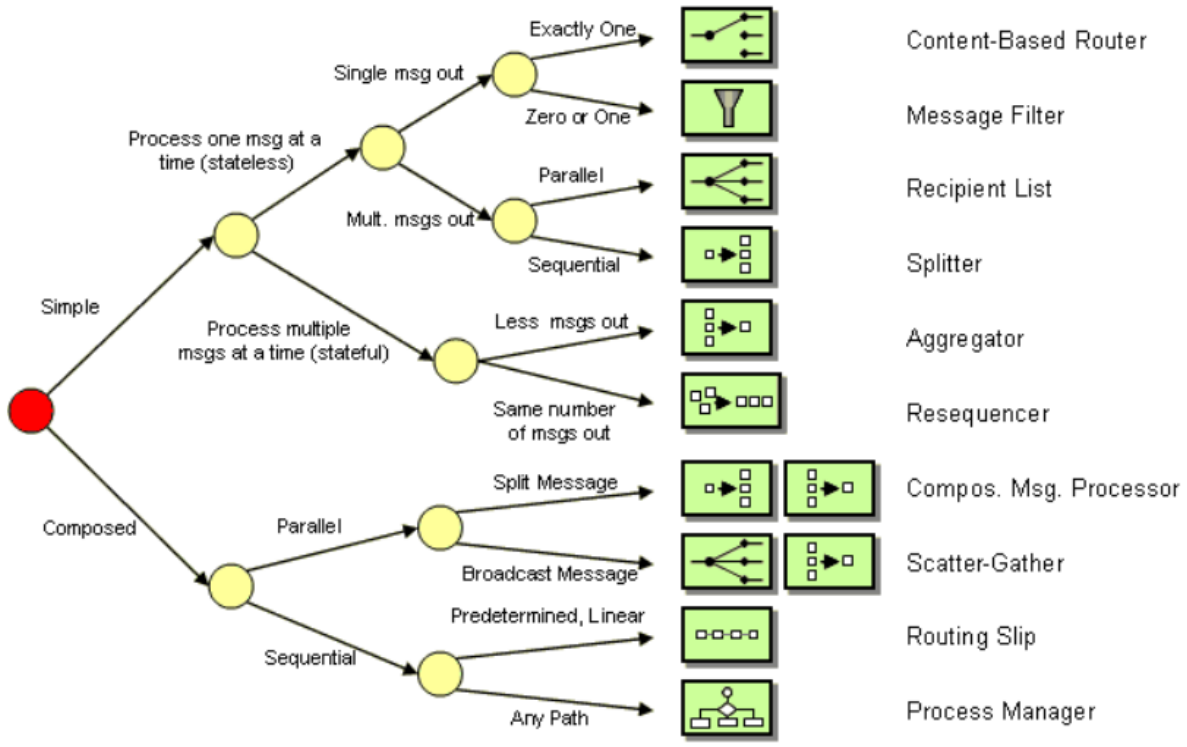


Figura 4 – Patrones de ruteo[29] [36]

El ruteo basado en contenido (*CBR*, *content-based routing*) es uno de los tipos de ruteo más nombrado y utilizado en los ESB. En el caso de mensajes XML, la lógica de ruteo puede evaluar el contenido utilizando XPath[30] y/o XQuery[31]. Además, también es común el uso de motores de reglas como Drools[32].

Transformación de mensajes y datos.

Dentro de los patrones referentes al procesamiento de mensajes, se encuentran los de transformación de mensajes y datos.

Un caso particular de transformación es el enriquecimiento de mensajes y datos. En [36] se define el patrón Content Enricher, que se muestra en la Figura 5. Este patrón utiliza información contenida en el mensaje de entrada (ej. identificadores) para obtener datos de una fuente externa. Luego que se obtienen esos datos adicionales, el Content Enricher los agrega al mensaje. Los datos originales que venían en el mensaje pueden mantenerse o no, dependiendo de los requerimientos específicos de la aplicación.

[29] http://www.enterpriseintegrationpatterns.com/patterns/messaging/MessageRoutingIntro.html

[30] http://www.w3.org/TR/xpath20/

[31] http://www.w3.org/TR/xquery-30

[32] http://www.drools.org/

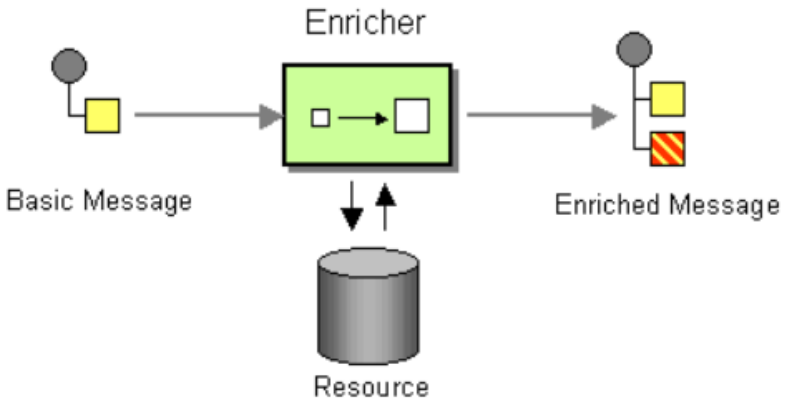


Figura 5 – Patrón Content Enricher[33] [36]

Otro patrón de transformación de interés definido en [36] es el Envelope Wrapper. Este patrón permite adaptar un mensaje que llega con un determinado *envelope* o "envoltorio," de manera que pueda ser enviado a un destino que requiere o no ciertos encabezados. El *envelope* generalmente consiste en un encabezado (*header*) antes de la carga útil (datos de aplicación) y opcionalmente un tráiler luego de esta, necesario para transmitir el mensaje dentro de cierto protocolo. La adaptación consiste en agregar un determinado *envelope* (lo hace el *Wrapper*) o en eliminarlo (lo hace el *Unwrapper*), como se muestra en la Figura 6.

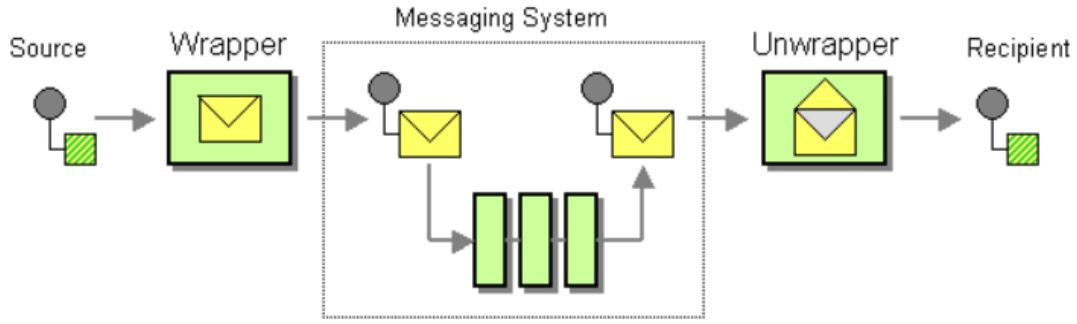


Figura 6 – Patrón Envelope Wrapper[34] [36]

2.1.7 Plataformas de Gobierno Electrónico.

Las Plataformas de Gobierno Electrónico son el soporte tecnológico para los servicios de Gobierno Electrónico, entendiéndose por Gobierno Electrónico al conjunto de actividades que realiza un gobierno a través del uso de las Tecnologías de Información y Comunicación (TIC) [39]. Esto se realiza para optimizar la gestión del Estado y brindar sus servicios en forma más efectiva a los ciudadanos.

Las iniciativas de Gobierno Electrónico han surgido desde los comienzos de este siglo, con un grado de avance diferente según los países y marcadas según el desarrollo de las TICs. Sin embargo, lograr que los servicios públicos se brinden a través de Internet en forma efectiva y completa, depende de varios factores, en donde las TICs son

[33] http://www.enterpriseintegrationpatterns.com/patterns/messaging/DataEnricher.html

[34] http://www.enterpriseintegrationpatterns.com/patterns/messaging/EnvelopeWrapper.html

uno de ellos, pero en donde también intervienen significativamente los aspectos organizacionales, legales y políticos.

En Uruguay la AGESIC[35] es la agencia responsable de la Plataforma de Gobierno Electrónico (PGEUy) [4]. Dicha plataforma opera sobre una red de alta velocidad que interconecta los diferentes organismos. A través de dicha red se logra la interconexión física de los organismos públicos con alta velocidad y disponibilidad de las conexiones, con conexiones que van de los 10 Mbps a los 100 Mbps. En la Figura 7 se muestra la arquitectura de la PGEUy y sus conexiones a través de la REDuy con los diferentes organismos.

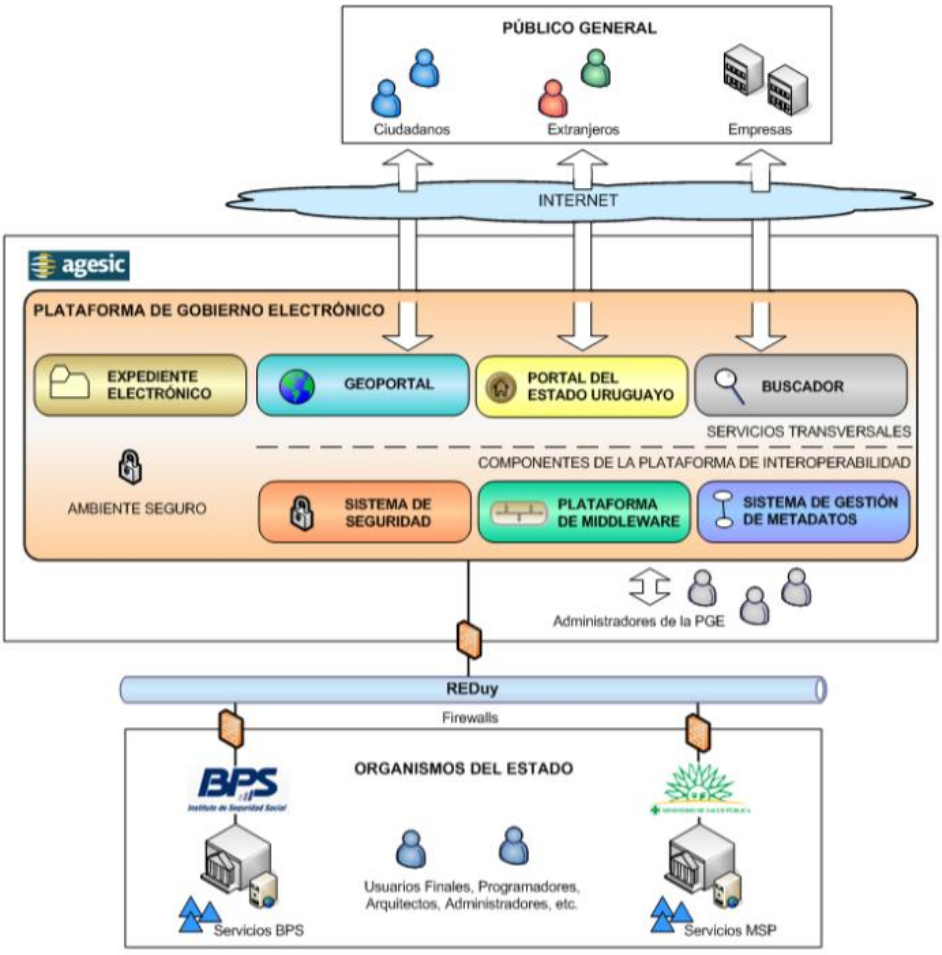


Figura 7 – Principales Componentes y Actores de la PGEuy [4]

Dentro de la plataforma se destacan los Componentes Base: el Sistema de Seguridad, la Plataforma de Middleware y el Sistema de Gestión de Metadatos. La plataforma permite interconectar sistemas de distintos organismos usando sus Componentes Base y a su vez brinda Aplicaciones para usuarios finales. Las aplicaciones actuales son: el Sistema de Expediente Electrónico, el Portal del Estado Uruguayo, el Buscador y el GeoPortal. La Plataforma de Middleware provee mecanismos que facilitan el desarrollo, despliegue e integración de servicios y aplicaciones, permitiendo la interoperabilidad entre los organismos del Estado uruguayo. Estos mecanismos brindan a su vez,

[35] http://www.agesic.gub.uy/

la infraestructura base para la implementación de la SOA a nivel del Estado. En la Figura 8 se detallan los grandes componentes de esta plataforma, destacándose la presencia de un ESB.

Figura 8 – Plataforma de Middleware de la PGEUy [4]

2.2 Trabajos Relacionados

A continuación se realiza un resumen de los trabajos dentro del área de investigación que se han identificado por guardar relación con el tema de esta tesis.

2.2.1 Plataformas de Integración Específicas del Dominio.

Si bien existen plataformas, basadas en tecnologías de middleware como el ESB y arquitecturas estilo SOA, que permiten construir soluciones a desafíos de integración complejos, dichas plataformas son demasiado amplias y generales como para adaptarse fácilmente a un dominio concreto. Es por esto que surgen las denominadas *plataformas específicas del dominio*, que son plataformas de middleware provistas de constructores y servicios adaptados a las necesidades de un dominio particular, con las abstracciones necesarias para el uso de patrones y algoritmos recurrentes [5].

Como ejemplo de estas plataformas, podemos encontrar aplicaciones a los dominios de la Salud y la Bioinformática, que se resumen a continuación:

2.2.1.1 Open Health Integration Platform

Open eHealth Integration Platform[36] (IPF) es un middleware tipo plataforma específico al dominio salud, con el propósito de proveer los mecanismos necesarios para la integración de aplicaciones informáticas en un contexto de salud. Las características más relevantes de esta Plataforma son:

- Soporte a los perfiles de integración en salud IHE: un conjunto de componentes que permiten la creación de interfaces especificadas en los perfiles IHE.
- Procesamiento de mensajes HL7: Un DSL específico para el procesamiento, manipulación y validación de mensajes HL7
- Soporte CDA: Un DSL específico para el procesamiento y manipulación de documentos CDA.
- DSL para EAI: Un DSL específico para la definición y uso de diferentes patrones de integración empresarial.
- Soporte QoS: soluciones para requerimientos no funcionales específicos a mensajería transaccional, administración de flujos, balanceo de carga y alta disponibilidad.

[36] http://www.openehealth.org/display/ipf2/IPF+Introduction

2.2.1.2 Plataforma de Integración de Servicios Bioinformáticos

En [40] y [41] se propone una Plataforma de Integración de Servicios Bioinformáticos, conectando herramientas de workflow y servicios bioinformáticos para el cumplimiento de requerimientos de integración avanzados, como ser las interacciones asincrónicas.

La plataforma permite el uso de herramientas locales para la composición de servicios sincrónicos y complementa sus capacidades incorporando interacciones asincrónicas y transformación de formatos. La interacción de las aplicaciones locales con la Plataforma se realiza a través de mensajes, los cuales serán direccionados a los servicios destino utilizando mecanismos de ruteo, conectores y protocolos específicos (Web services SOAP, R, etc). Por otro lado, las respuestas podrán ser entregadas mediante mecanismos de callback o notificaciones. En la Figura 9 se muestra la intermediación que realiza la plataforma en el software Taverna[37] y los Web services de los proveedores.

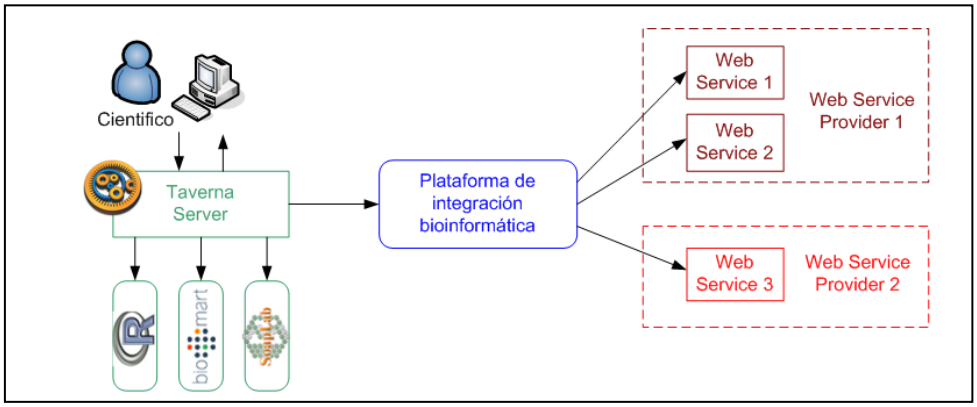
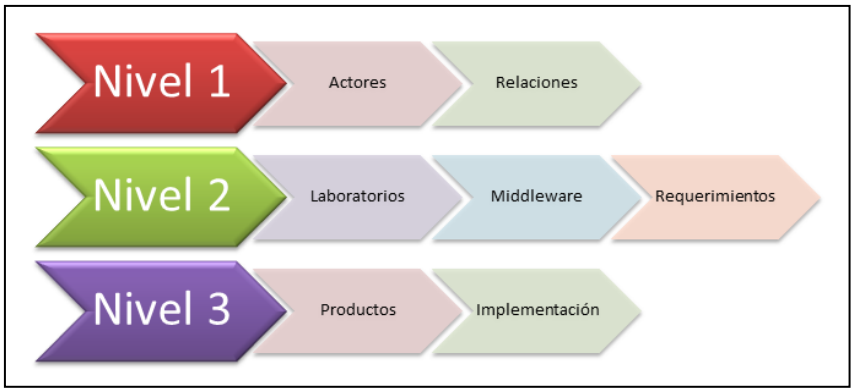


Figura 9 – Plataforma de Integración de Servicios Bioinformáticos [41]

La especificación de la Plataforma de Integración está dada en diferentes niveles de refinamiento (Figura 10), donde en un primer nivel de especificación, se presentan de forma abstracta los actores y las relaciones que existen entre ellos, sin tener en cuenta tecnologías, requerimientos ni contextos de uso (tipos de laboratorios). En un segundo nivel de refinamiento, se toman en cuenta estos elementos y se los combinan entre sí, definiendo Plataformas de Integración específicas, mientras que en un tercer nivel, se toman en cuenta aspectos técnicos relativos a la implementación y el uso de productos.



[37] https://taverna.incubator.apache.org/

Figura 10 - Niveles de refinamiento y dimensiones de refinamiento por nivel [41]

Todas las Plataformas propuestas se basan en patrones de mensajería e integración comúnmente soportados por las tecnologías de middleware utilizadas (Colas de mensajes, Web services, ESB), por lo que brinda una solución genérica y factible de aplicarse en la mayoría de estos productos.

2.2.2 Integración de Servicios Geoespaciales en Plataformas de Gobierno Electrónico.

Dentro de la problemática general de integración de servicios geográficos con otros tipos de sistemas, tiene especial importancia por su complejidad técnica, la integración de los mismos en plataformas de gobierno electrónico. A continuación se describen algunos trabajos e iniciativas, comenzando por una propuesta aplicable en primera instancia para el gobierno electrónico en Uruguay, y finalizando por un relevamiento de las iniciativas y trabajos en desarrollo dentro algunos de los organismos más importantes del área en Estados Unidos.

2.2.2.1 Integración de Servicios Geográficos en la Plataforma del Gobierno Electrónico de Uruguay.

En [42] encontramos un trabajo relevante de integración de servicios geoespaciales que se focaliza en la PGE del estado uruguayo [4]. En dicha tesis de maestría, se realiza una identificación de escenarios de integración, basándose en diferentes tipos de servicios ofrecidos y de clientes consumidores, así como en las características de los procesos involucrados.

En síntesis, los aportes de ese trabajo pueden resumirse de la siguiente manera:

- Identificación y descripción de escenarios de integración de servicios geográficos en una plataforma de gobierno electrónico, basándose en diferentes tipos de servicios ofrecidos y de clientes consumidores, así como en características de los procesos que pueden realizar organismos participantes de la plataforma de gobierno electrónico.
- Análisis sobre la compatibilidad de Web services Geográficos y Plataformas de Gobierno Electrónico, tomando como base los escenarios identificados. Este análisis identifica en detalle los elementos de incompatibilidad entre los protocolos involucrados, sentando las bases para definir una propuesta de solución a este problema.
- Propuesta de solución para mejorar la compatibilidad entre las plataformas de gobierno electrónico y los servicios de información geográfica, a través de mecanismos basados en transformaciones de paquetes de datos. Esta propuesta de solución trata de aprovechar al máximo los sistemas y plataformas existentes, teniendo en cuenta que se trata de infraestructura ya establecida y que funciona adecuadamente en sus propios contextos de uso.

Resultan especialmente relevantes los escenarios identificados, que se resumen a continuación.

Consulta de Información Pública

En este escenario, existen organismos que generan información de interés público, que puede ser utilizada tanto por otros organismos como por los ciudadanos o público en general. Este escenario está pensado para ofrecer al público IG básica general (ej. infraestructura vial, información catastral, etc.). Si bien esta información estará disponible para el público en general, también puede ser de utilidad para funcionarios de diferentes organismos del estado. La información puede ser accedida desde visualizadores Web o desde aplicaciones GIS de escritorio para realizar cruzamiento con información propia del usuario. A nivel de protocolos, esta información puede ser brindada mediante WMS o WFS de sólo lectura.

Servicios para Públicos Específicos

En este escenario, un organismo especializado en determinado tipo de IG brinda servicios específicos para otro organismo de acuerdo a los requerimientos de este último. Este caso se presenta cuando algunos organismos generan información que es de particular interés para otros organismos. Esto evitaría que el cliente tenga que

asumir la tarea de generarlos, cuando ya hay un organismo experto en la generación de dicha información. Dicha información se puede publicar tanto por WMS como WFS de sólo lectura ya que será accedida solamente para consulta. Se deberán proveer mecanismos de seguridad para control de acceso.

Colaboración para la Generación de Información Geográfica

Este escenario se basa en que un organismo central tiene acuerdos con otros organismos para colaborar en la actualización de determinada información. Para esto el organismo central brinda un servicio que permite a los organismos colaboradores actualizar la información. Esto se puede hacer a través de un servicio WFS transaccional y requiere seguridad adicional para validar que quienes actualizan la información son los habilitados para hacerlo. A nivel del organismo central se deberá ir validando la actualización de la información y consolidándola en nuevas versiones del conjunto de datos. Como ejemplo se menciona el caso de las rutas nacionales, que son responsabilidad del Ministerio de Transporte y Obras Públicas, y para las que se podría contar con la colaboración de las Intendencias para mantener actualizada la información sobre su estado.

Colaboración para la Realización de Procesos Administrativos

Este escenario requiere que entre los organismos existan acuerdos de datos a intercambiar, teniendo presente el proceso (o los procesos) en que está involucrado el intercambio. Tradicionalmente el ciudadano consultaba información (en un formulario) en un organismo y lo llevaba a otro para realizar un trámite. Esto se puede hacer más eficientemente si el segundo organismo consulta online dicha información, ahorrándole tiempo y costos al ciudadano.

En el caso más general, puede ocurrir que un proceso requiera la intervención de varios organismos en determinado orden, por lo que se deberá resolver nuevamente el problema del contexto del trámite y la coordinación del orden de intervención de los organismos. Otro problema que surge es que puede darse intercambio de información sensible, por lo que se requieren mecanismos de seguridad específicos, tanto para el acceso a las consultas, como para la transmisión de la información. Mientras que en el escenario anterior el objeto de la colaboración era la gestión de información geográfica, en este escenario se apunta a mejorar la gestión de procesos en general que utilizan información geográfica, siendo por lo tanto de aplicación mucho más amplia.

Dado que los procesos administrativos generalmente son gestionados desde sistemas de expediente, como ser los BPMS[38] (Business Process Management Suite), este escenario requerirá la integración del intercambio de información geográfica en dichos sistemas. Los usuarios destinatarios de este escenario son los funcionarios de los organismos involucrados en el proceso administrativo.

Público generando Información Geográfica

Este escenario, que se relaciona con el concepto de Volunteered Geographic Information (VGI), requiere que el organismo interesado en generar una comunidad para la generación colaborativa de datos geográficos brinde un sitio completo para esta comunidad. El usuario registrado en la comunidad podrá ingresar información, generalmente de granularidad muy fina (una ocurrencia de algún fenómeno en un punto), y consultar la información ingresada por otros miembros de la comunidad. Esto escenario permite tener información voluntaria de los ciudadanos que de otra forma sería muy difícil o costosa de obtener siguiendo los procedimientos tradicionales de generación de información geográfica.

[38] BPMS es un conjunto de herramientas que permiten administrar procesos de negocio durante todo su ciclo de vida (diseño, ejecución, monitoreo, análisis)

2.2.2.2 *Integración de GIS en plataformas de gobierno electrónico en Estados Unidos*

La *Environmental Protection Agency*[39] (EPA), que es la agencia de protección ambiental de Estados Unidos, por estar organizada fuertemente alrededor de condiciones y eventos físicos que son específicos de la ubicación geográfica, ha liderado históricamente en el área GIS de ese país. La EPA viene desarrollando, a partir de 2012, una plataforma llamada EPA Geospatial Platform [43], que consiste en un framework compartido de tecnología y gobernanza, que comprende a una comunidad de expertos y un conjunto de herramientas, datos y servicios geoespaciales para coordinar y consolidar actividades de mapeo. En la Figura 11 puede verse el modelo conceptual de dicha plataforma.

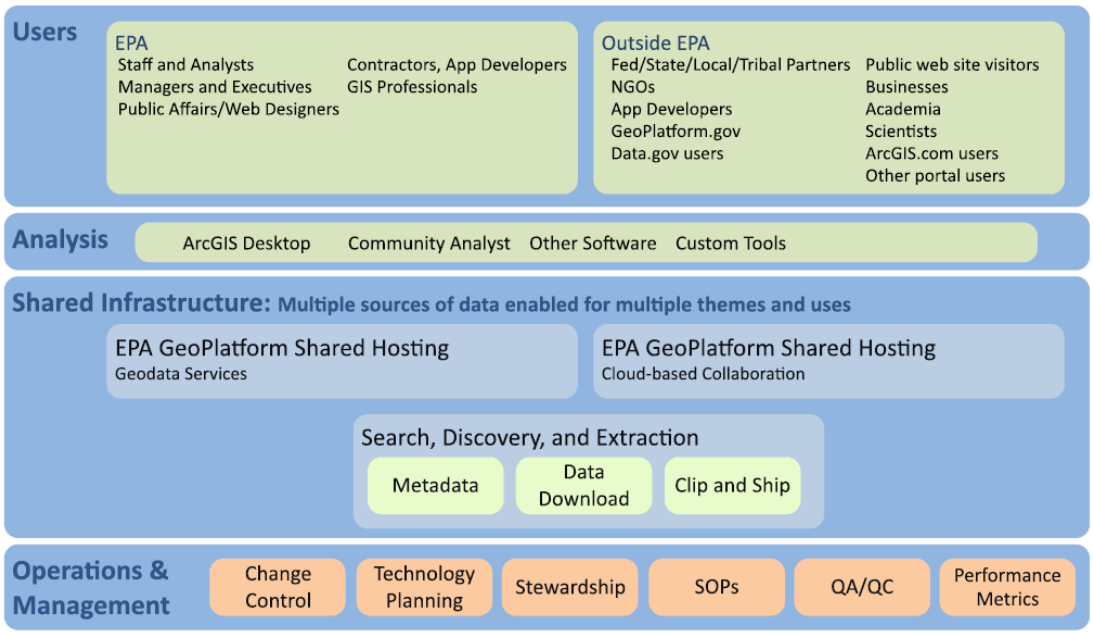


Figura 11 – Modelo conceptual de la EPA Geospatial Platform [43]

La EPA Geospatial Platform es utilizada principalmente por usuarios internos en la actualidad, pero está planeado que entre el 2016 y el 2017 esta se encuentra completamente operativa, brindando servicios al público y a otras agencias del gobierno, como la *National Oceanographic and Atmospheric Administration* (NOAA), el *U.S. Geological Survey* (USGS), la *National Aeronautical and Space Administration* (NASA), etc. La EPA Geospatial Platform ha sido tomada como base para el desarrollo de la *National Geospatial Platform* que es gestionada por el Ministerio del Interior y debe interoperar con las plataformas gestionadas por cada agencia de gobierno.

En el plan estratégico definido en [43], se menciona que la EPA está adoptando una arquitectura tipo SOA para todas sus plataformas de servicios, con la integración de un conjunto de herramientas que incluyen un ESB, un BPMS (Business Process Management Suite) y un motor de reglas, entre otros. No se menciona en dicho documento como impacta esta nueva arquitectura en la Geospatial Platform.

En [44] se presenta un conjunto de lineamientos o "línea de negocio" que busca impulsar un enfoque coordinado entre las diferentes agencias gubernamentales para la producción, el mantenimiento y el uso de IG. Esta iniciativa, que surge directamente de la presidencia de Estados Unidos, complementa los lineamientos establecidos por otros órganos, como la NSDI, mencionada anteriormente, cuyo desarrollo es coordinado por el FGDC (Federal Geographic Data Committee). En este documento se propone una arquitectura de muy alto nivel, en donde los proveedores publican sus servicios y los consumidores los encuentran utilizando sus sistemas de soporte de

[39] http://www3.epa.gov/

decisiones, para visualizar y manipular los datos geoespaciales y sus representaciones en mapas. El acceso "mediado" (*mediated access*) permite resolver el problema de heterogeneidad entre productores y consumidores a través del procesamiento, transformación, enriquecimiento e integración de datos. El acceso directo también se permite para datos y servicios geoespaciales cuando la mediación no se considera necesaria.

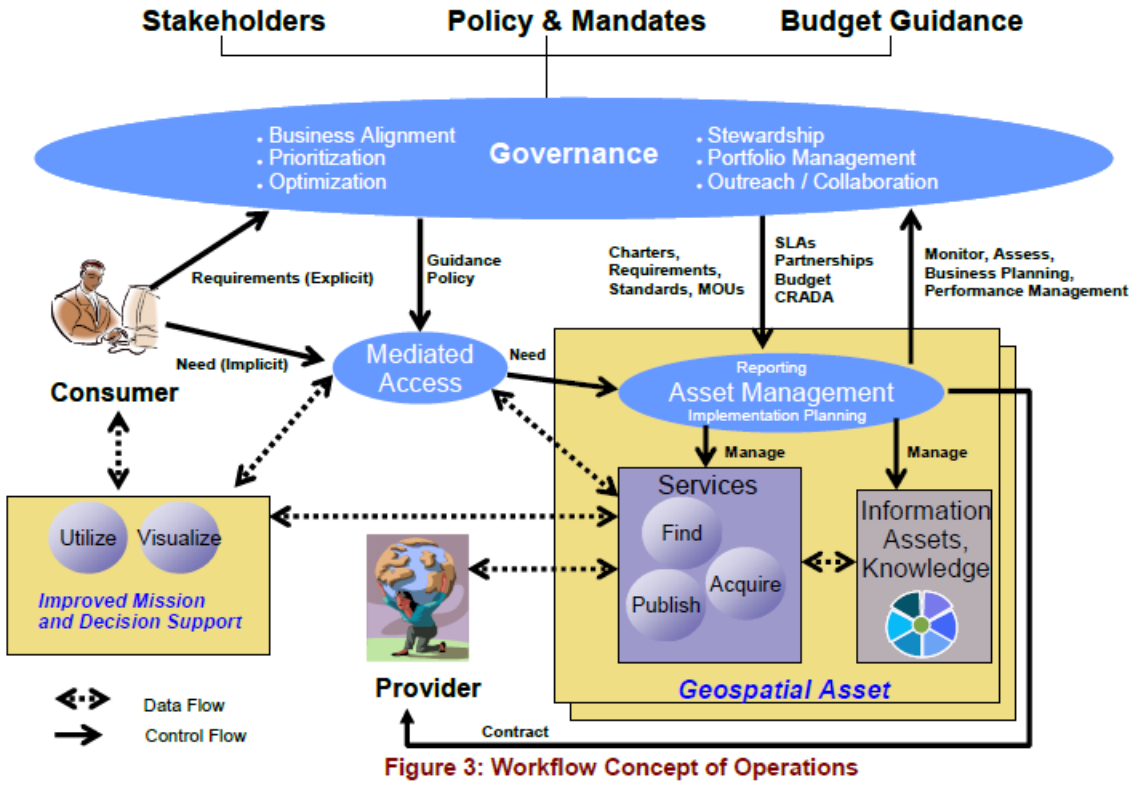


Figura 12 – Workflow de acceso a recursos geoespaciales [44]

2.2.3 Otros Trabajos Relacionados.

Dentro de los otros trabajos estudiados encontrados en la literatura sobre el tema, la mayoría atacan la problemática de integrar IG o servicios geoespaciales con servicios, aplicaciones o datos de un negocio en particular. Muchos de estos trabajos presentan soluciones concretas para un negocio específico (ej. aplicaciones municipales)

En [45] se analizan diferentes escenarios de integración, que permitirían ofrecer al usuario final una visión integrada de la IG y la información de negocio. Esos escenarios son:

- Integración de las bases empresariales con las bases de datos geoespaciales. En este caso se utiliza algún mecanismo propio de los manejadores de bases de datos para ofrecer una vista integrada de la IG y la información de negocio.
- Integración a nivel de la interfaz de usuario. En este caso el cliente es el encargado de consumir la IG y la información de negocio de las diversas fuentes e integrarla para mostrarla al usuario.
- Integración de servicios empresariales con servicio geoespaciales. En este caso existe un componente de middleware que realiza la integración en la capa de negocio.

En [46] se realiza un estudio de diferentes estilos arquitectónicos empleados para integrar GIS con sistemas empresariales, tomando como referencia tanto propuestas académicas como casos de negocio. Las arquitecturas

estudiadas son: SOA, ESB, Arquitectura Dirigida por Eventos (EDA) y Cloud Computing. Para cada caso el autor analiza las fortalezas y debilidades.

En el caso particular de una arquitectura de tipo ESB, se destaca como fortalezas:

- su capacidad de combinar los estilos de integración que brindan tanto una SOA como una EDA
- es útil tanto en escenarios de integración intra-organizacional (Entreprise Application Integration) como inter-organizacional (Business-to-business)
- aprovecha la existencia de varios estándares muy difundidos y promete incorporar otros que se encuentran en evolución y maduración.
- permite agregar servicios geoespaciales al bus de servicios en forma transparente para el consumidor.

Como desventajas se menciona la preocupación por el uso de grandes volúmenes de datos (típicos en aplicaciones que usan IG) y cuestiones de gobernanza y administración de la plataforma que pueden resultar complejas.

En [47] se realiza una propuesta de integración de servicios geoespaciales en una plataforma estilo SOA para aplicaciones de una municipalidad. La implementación se realiza utilizando un ESB, como se muestra en la Figura 13.

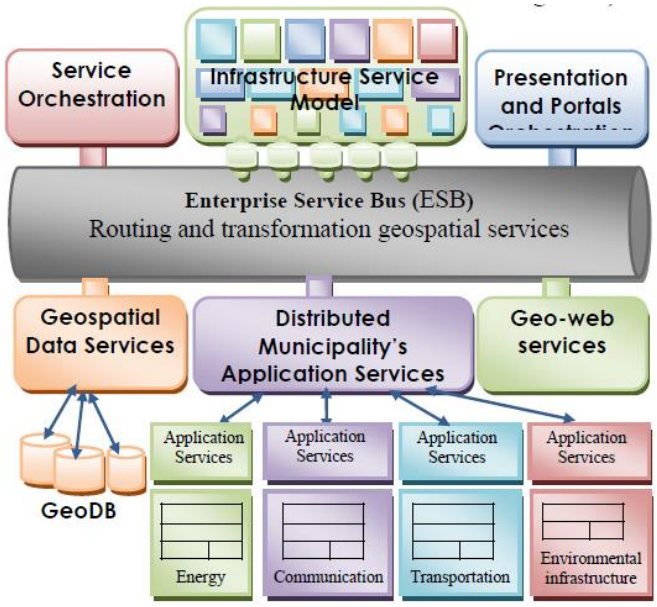


Figura 13 – Propuesta de integración de servicios municipales con servicios geoespaciales usando un ESB [47]

En [48] también se propone una arquitectura estilo SOA para la integración de GIS con otros sistemas, y se analizan casos de éxito en la administración pública de Dinamarca. En la Figura 14 puede verse como los diferentes tipos de servicios geoespaciales son utilizados por diversos clientes, en particular por un ESB.

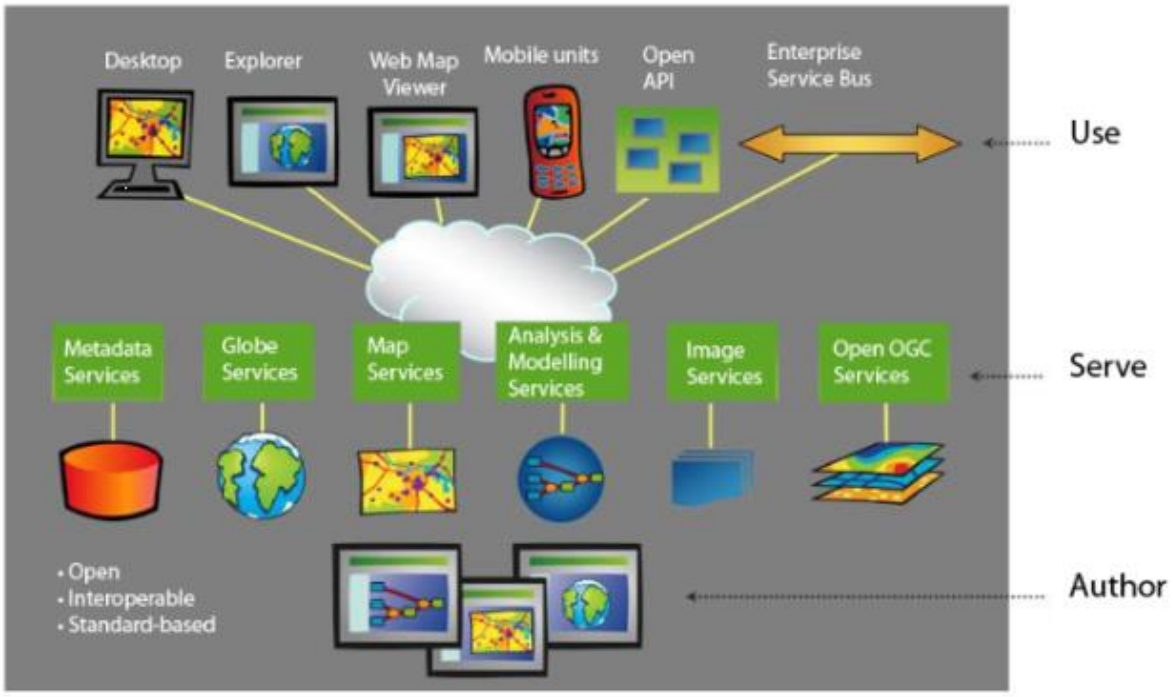


Figura 14 – Propuesta de integración de GIS en una SOA [48]

2.3 Síntesis.

En este capítulo se presentó un resumen del conocimiento existente en los temas más relevantes para este trabajo. Dentro de los trabajos relacionados, existen propuestas o proyectos en los que se plantea la utilización de un ESB como middleware de integración de servicios geoespaciales. Sin embargo, en ninguno de esos trabajos se plantea la construcción de una solución de mayor nivel, como ser una plataforma específica de dominio específica para GIS.

A continuación se realizan algunas reflexiones sobre las limitaciones y carencias detectadas dentro de las tecnologías, estándares y soluciones existentes.

2.3.1 Heterogeneidad en tecnologías de Web services.

Como se ha visto en este capítulo, los estándares de Web services basados en especificaciones de W3C y los estándares de Web services basados en especificaciones de OGC han surgido y evolucionado en paralelo. Esto ha llevado a que existan varias diferencias entre ellos, tanto a nivel formal y tecnológico como incluso en los objetivos para los que fueron desarrollados. A nuestro criterio, la diferencia más importante entre ellos radica en que los Web services W3C son de propósito general, por lo que pueden utilizarse en cualquier dominio, en tanto que los Web services OGC son específicos de un dominio (el de las aplicaciones GIS).

En la Figura 15 puede observarse gráficamente cómo se diferencian los tipos de Web services que se han analizado en este capítulo, según las dimensiones de Facilidad de Desarrollo y Genericidad. Por un lado, los OWS son los más fáciles de desarrollar, ya que, al poseer un conjunto fijo de operaciones, existen muchas herramientas (ej. servidores de mapas) que los implementan y el trabajo del desarrollador se centra en la configuración y no en la programación de los mismos. Por otro lado, tanto los Web services SOAP como REST deben ser implementados para cada aplicación, programando las operaciones en algún lenguaje de alto nivel (ej. Java, C#, etc.), por lo que su desarrollo no es tan sencillo, pero a su vez, son completamente genéricos y pueden utilizarse incluso también en aplicaciones GIS.

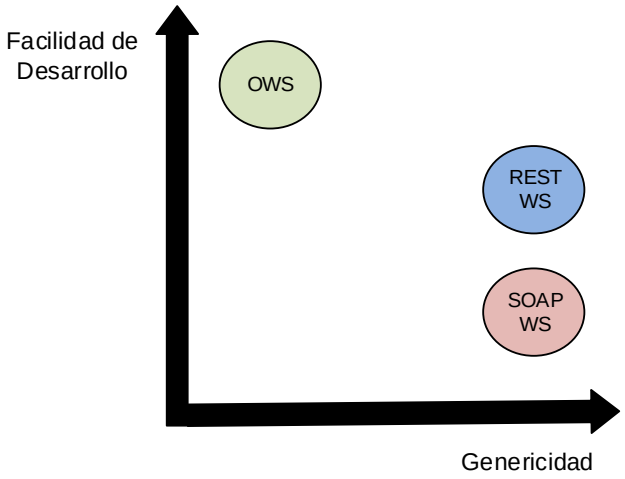


Figura 15 – Genericidad vs Facilidad de Desarrollo en Web services

Otro punto importante que se puede contrastar, basándonos en [49], es el de Complejidad de Aplicación contra Facilidad de Desarrollo. En este caso vemos que los Web services SOAP, al poder aprovechar las características avanzadas que les proporcionan los WS-* (seguridad, orquestación, etc.), son capaces de cumplir con los requerimientos de aplicaciones más complejas que los otros dos tipos de Web services.

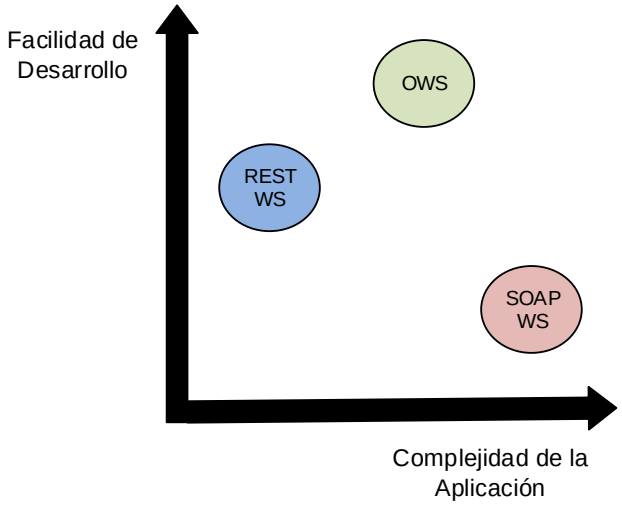


Figura 16 – Complejidad de la Aplicación vs Facilidad de Desarrollo en Web services

2.3.2 Carencias en mecanismos de integración entre la información geográfica y la información de negocio.

Con el auge que ha experimentado la producción y explotación de la IG en los últimos años, ha surgido la necesidad de integrar la misma con todo tipo de información general, y en particular, con información empresarial. Sin embargo, las herramientas GIS aún no soportan mecanismos de integración que permitan trabajar con IG e información de negocio provenientes de distintas fuentes en forma transparente. Generalmente, la integración se realiza mediante algún desarrollo específico para una aplicación particular, siguiendo alguno de los enfoques presentados en [45].

Específicamente, en el área de los servicios geoespaciales, se publican servicios que trabajan únicamente con capas de IG que son almacenadas en bases de datos geoespaciales (siguiendo algún estándar, como el Simple

Features Standard de OGC) o en archivos en formato *shapefile*[40], por citar dos de las opciones más utilizadas para datos vectoriales..

2.3.3 Carencias en mecanismos de seguridad integrados a los Web services geoespaciales.

Las especificaciones de los OWS no hacen mención a aspectos de seguridad. En el caso de los OWS más básicos y populares, como WMS, esta carencia no ha traído mayores inconvenientes ya que se utiliza generalmente para brindar servicios públicos o en geoportales públicos. Sin embargo, a medida que los geoservicios se vuelven más sofisticados y diversos, las organizaciones empiezan a ponerse más restrictivas en el acceso que brindan a su información. Es claro que ninguna organización desea publicar un servicio WFS transaccional en Internet sin asegurarse de que éste esté protegido al menos por un mecanismo de autenticación de los usuarios. Resulta una desventaja, por lo tanto, que dicho mecanismo tenga que ser proporcionado por cada aplicación, como una capa de seguridad extra que se agrega antes de realizar el pedido, a nivel del servidor de mapas, con parámetros dependientes del servidor que se agregan a los pedidos.

A la fecha, el único estándar definido por OGC relativo a la seguridad es GeoXACML, pero este no ha sido incorporado a los servidores de mapas, por lo que su adopción viene siendo muy lenta.

2.3.4 Limitaciones en herramientas de GIS empresarial.

El estado de las herramientas actuales y tecnologías no facilitan la implementación de plataformas de servicios que sean capaces de integrar servicios empresariales y geoespaciales en forma abierta, transparentes, basada en SOA.

2.3.5 Carencias en la implementación plataformas abiertas para integrar servicios empresariales y geoespaciales.

En la actualidad no existen plataformas abiertas en las que se pueda integrar un conjunto cualquiera de servicios geoespaciales y servicios de negocio, sin tener que implementar el algoritmo de integración caso a caso. En particular, no existen plataformas específicas de dominio como las relevadas anteriormente para el área GIS.

[40] http://www.esri.com/library/whitepapers/pdfs/shapefile.pdf

3 Análisis del Contexto de Aplicación

En esta sección se realiza un análisis de las características principales que presenta el contexto de aplicación para el que se propone la solución que será presentada en detalle en el Capítulo 4. En este sentido, se comienza describiendo el Marco de trabajo y luego los Requerimientos identificados para una plataforma de integración específica del dominio GIS empresarial.

3.1 Marco de Trabajo

Dentro del marco de trabajo que se toma como punto de partida, es importante describir algunas de sus características:

1. **Existe una infraestructura de tipo empresarial con componentes geográficos y no geográficos**. En particular, interesan las PGEs, las SOAs, las IDEs y los GIS empresariales. Dado que estas categorías no son ortogonales (una PGE puede tener una arquitectura estilo SOA, una IDE puede implementarse sobre una PGE, etc.), una infraestructura que encaje dentro de este marco de trabajo podría compartir más de una de estas denominaciones.
2. **Existen varios organismos que son consumidores y/o proveedores de servicios, tanto geográficos como no geográficos**. Dentro de los organismos se contemplan tanto públicos (caso típico de una PGE) como privados.
3. **Los organismos utilizan las tecnologías de Web services más comunes en cada caso.** En el caso de los servicios geoespaciales, interesan aquellos que son estándares (OWS) y que han demostrado un alto nivel de utilización. En el caso de los Web services de negocio, interesan particularmente los Web services SOAP y los estándares WS-* por ser los más robustos y utilizados en comunicaciones B2B y dentro de los ambientes que se describen en los puntos anteriores.

En las siguientes secciones se analizan los principales elementos que componen el marco de trabajo: los servicios, sus proveedores y sus consumidores. Asimismo se presentan escenarios habituales dentro del marco de trabajo en el cual se desarrolla esta tesis.

3.1.1 Categorización de Servicios Geoespaciales

Como hemos visto en la Sección 2.1.5 del capítulo anterior, existe una cantidad considerable de estándares de Web services geoespaciales, que sirven a diferentes aplicaciones dentro del dominio GIS. En este trabajo, si bien es necesario detenerse en particularidades de algunos de los estándares más utilizados (caso WMS o WFS), también interesa poder tener una visión más abstracta de los mismos, mediante una clasificación que permita encontrar características comunes entre ellos. Es decir, esta clasificación permite centrarse en *qué* hace un servicio geoespacial y no en el *cómo* lo hace.

Dentro de este trabajo, los servicios geoespaciales o *geoservicios* van a ser clasificados como sigue. Esta clasificación se basa en los criterios encontrados en la literatura ([1] y [3]), pero no necesariamente adhiere a la terminología exacta de ninguna de esas dos fuentes.

3.1.1.1 Servicios de Mapas

Los servicios de mapas [3], también conocidos como servicios de visualización [1], son aquellos que generan un mapa en forma dinámica, en base a ciertos parámetros que envía el cliente. Dentro de los parámetros, los más importantes son la extensión geográfica del mapa, las capas que lo componen y el formato de salida. La salida principal de estos servicios es un archivo binario, típicamente en un formato de imagen liviano como JPEG, PNG o GIF.

El cliente no requiere de una gran capacidad de procesamiento ni de una API compleja para interpretar (o *parsear*) el archivo, ya que cualquier navegador o cliente liviano puede desplegar fácilmente una imagen de esas características.

Además del mapa en sí, estos servicios pueden obtener información alfanumérica, como ser los atributos descriptivos asociados a una o varias geometrías. Sin embargo, la capacidad de consulta que tienen los servicios de mapas es limitada.

Los servicios de mapas nunca permiten modificar IG.

En resumen, podemos concluir que las principales características son:

- Devuelven un mapa como un archivo binario
- Son de sólo lectura
- Tienen capacidad básica de consulta de datos
- Sólo requieren clientes muy ligeros
- Son los más utilizados en la actualidad
- Son servicios de mapas: WMS y WMTS.
- Existen servicios de mapas que no son OWS y que se acceden mediante APIs propietarias (ej. Google Maps).

3.1.1.2 Servicios de Datos

Los servicios de datos [3], a diferencia de los anteriores, permiten manipular los datos "crudos." Así como los servicios de mapas trabajan principalmente a nivel de capa geográfica, los servicios de datos trabajan al nivel de los elementos atómicos que componen esas capas. Por ejemplo, si se trata de una capa vectorial (una capa de ríos), los elementos que la componen son los objetos geográficos o *features* (cada uno de los ríos). Los servicios de datos permiten realizar operaciones *CRUD* (*create*, *read*, *update*, *delete*) sobre los elementos de una capa, aunque en algunos casos existen versiones más básicas de los mismos que sólo permiten realizar consultas y no modificaciones.

En un caso de uso típico de un servicio de datos, el usuario visualiza una determinada capa vectorial y puede realizar modificaciones sobre alguna entidad geográfica, tanto a nivel de su geometría como a nivel de sus atributos descriptivos. Por ejemplo, supongamos que el usuario se encuentra modificando una capa de polígonos que representan los padrones de una ciudad, con el fin de realizar una fusión de dos padrones que pasarán a ser un único padrón con un nuevo número (ver Figura 17). Las aplicaciones que permiten realizar este tipo de edición, tanto Web como de escritorio, ponen a disposición del usuario herramientas típicas de aplicaciones de dibujo o CAD (Computer-Aided Design). De esta manera, el usuario podría dibujar un nuevo polígono que sea la unión de otros dos, guardarlo con un nuevo número de padrón y eliminar los dos padrones anteriores. La aplicación deberá en ese caso llevar un registro de las acciones realizadas sobre la capa para traducirlas en invocaciones al servicio de datos cuando el usuario quiera guardar los cambios.

Por ejemplo, si utilizamos WFS, que es un servicio de datos, para llevar a cabo el caso de uso anterior, debería invocarse la operación *Transaction* con un *create* y dos *delete*, posiblemente con la invocación previa de dos operaciones LockFeature sobre los padrones a eliminar para evitar problemas de modificaciones concurrentes.

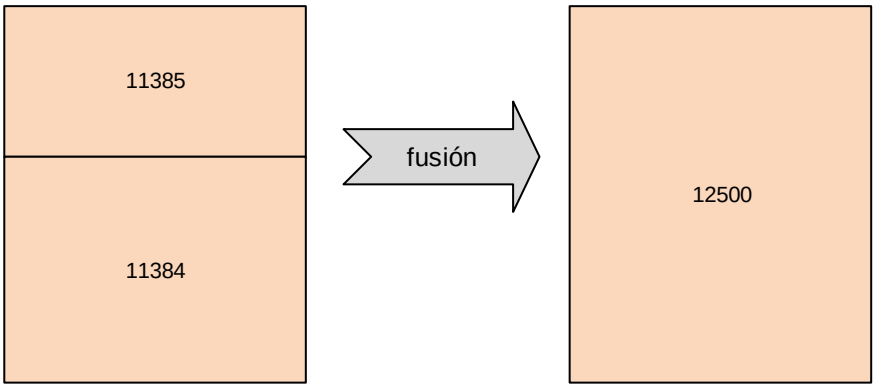


Figura 17 – Utilización de un servicio de datos para fusinar dos padrones

Los servicios de datos de OGC envían generalmente la información en formato XML, por lo que esta debe ser *parseada* en el cliente.

En resumen, podemos concluir que las principales características son:

- Permiten consultas avanzadas para diferentes modelos de datos.
- Permiten operaciones CRUD.
- Generalmente intercambian mensajes en formato XML.
- Son servicios de datos: WFS (datos vectoriales), WCS (datos *raster* o coberturas), SOS (redes de sensores).

Cabe mencionar que algunos servicios de datos también pueden utilizarse para la visualización de los datos en forma de mapa [3]. En este caso, es el cliente el encargado de construir la imagen digital a partir de los datos de cada entidad geográfica (geometría y otros atributos relevantes para la representación del mismo).

3.1.1.3 Servicios de Metadatos

Los servicios de metadatos [3] son los que permiten describir y consultar metadatos que refieren tanto a datos como a servicios. Como se mencionó en el Capítulo 2 (ver Sección 2.1.5), todos los OWS poseen una operación GetCapabilities que permite obtener las capacidades del servidor, mediante un conjunto de metadatos de diversa índole. De esta manera, todos los OWS (WMS, WFS, etc.) poseen capacidades básicas para publicar y permitir la búsqueda de metadatos. Sin embargo, esta única operación no es suficiente para permitir una gestión más sofisticada de los metadatos, por lo que se han desarrollado servicios específicos para la publicación y administración de metadatos. Dentro de los estándares OGC para este tipo de servicios se encuentran los Catalogue Services [29] y su perfil específico para la Web, CSW (ver Sección 2.1.5.3).

3.1.1.4 Servicios de Procesos

Los servicios de procesos tienen como finalidad la de compartir en la Web los cálculos, modelos y algoritmos que tradicionalmente sólo se encontraban en aplicaciones GIS de escritorio (ej. ArcGIS, gvSIG, QGIS, etc.)

El disponer de este tipo de servicios tiene dos ventajas principales [1]:

- Desarrollo de procesos complejos con alto requerimientos de cómputo. En los casos en que el cuello de botella se encuentre en el tiempo de ejecución de un proceso, ejecutarlo remotamente puede compensar el tiempo (y costo) que lleve la transmisión de datos (la entrada y salida del proceso), y de esta manera ser una alternativa preferible a la ejecución local de los procesos.

- **Ejecución distribuida de procesos.** Muchos procesos pueden dividirse en subprocesos y repartirse en varias máquinas. De esta manera puede realizarse una orquestación de servicios de procesamiento, en donde se aprovecha el procesamiento en paralelo, siendo esto transparente para el cliente.

Dentro de los OWS, el único estándar que se encuentra dentro de esta categoría es WPS (ver Sección 2.1.5.3).

3.1.2 Consumidores y Proveedores de Servicios

A continuación se propone una posible clasificación de los grandes componentes que interactúan en un GIS empresarial. Por un lado, se encuentran los diversos clientes, tanto los que trabajan fundamentalmente con IG como los que consumen habitualmente servicios de negocio (sin IG involucrada). Por otro lado, los diferentes proveedores de servicios que son accedidos por dichos clientes.

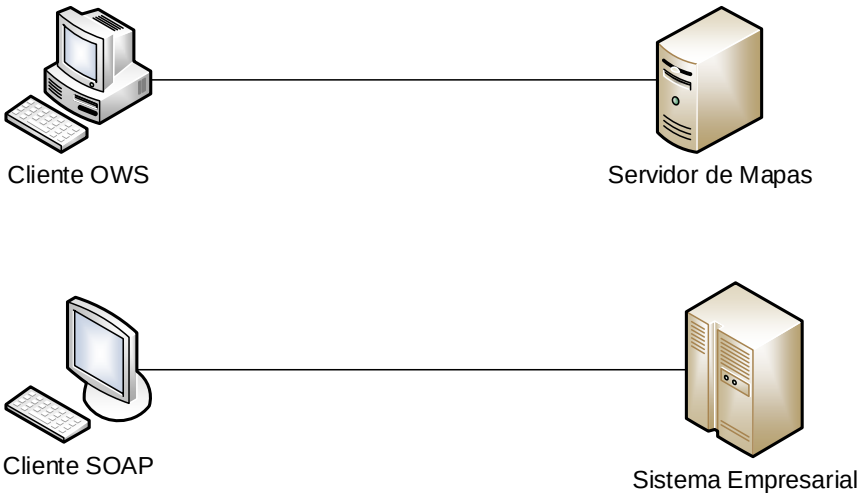


Figura 18 – Consumidores y proveedores de servicios en un GIS empresarial

Dentro de los consumidores, distinguimos dos tipos bien diferenciados:

Clientes OWS. Este tipo de clientes son aquellos que consumen geoservicios en base a estándares OWS. Caen dentro de esta clase, por ejemplo, los visualizadores que utilizan servicios de mapas y de metadatos, y los editores de IG, tanto Web como de escritorio, que utilizan, además de los anteriores, servicios de datos y de procesos.

Clientes SOAP. Este tipo de clientes son aquellos que utilizan Web services SOAP para el intercambio de mensajes. Generalmente son aplicaciones desarrolladas a medida para un determinado negocio.

Es importante destacar que los clientes OWS, por basarse en estándares, pueden ser aplicaciones de tipo COTS[41] o aplicaciones libres descargables, que no se desarrollan a medida, sino que cubren la gran mayoría de las funcionalidades típicas de un GIS, sin proveer capacidades de comunicación con sistemas empresariales. Dentro

[41] COTS (Comercial off-the-shelf) se aplica en el caso de software para aplicaciones que se venden al público en general y que no se desarrollan para un cliente específico.

de esta categoría, podemos mencionar algunos visualizadores/editores Web populares como p.mapper[42], Geomajas[43] o Mapbender[44], así como editores de escritorio tales como QGIS[45], gvSIG[46] o ArcGIS[47].

Dentro de los proveedores, distinguimos dos tipos bien diferenciados:

Servidores de Mapas. Los servidores de mapas son los encargados de publicar los geoservicios, tanto servicios de mapas como servicios de datos, e incluso servicios de procesos en algunos casos.

Sistemas Empresariales. Dentro de los sistemas empresariales que pueden encontrarse en una organización, interesan particularmente aquellos que exponen una interfaz que puede ser accedida por un cliente, tales como un servidor de aplicaciones que publica Web services SOAP.

En principio, quedan por fuera de esta clasificación los Web services REST que no sean geográficos, ya que se considera que no están tan desarrollados dentro del ámbito empresarial y de gobierno electrónico, que ha sido definido anteriormente como marco de trabajo en el que se debe desarrollar la solución. De todas formas, es perfectamente factible realizar una extensión de este marco de trabajo que también incluya a los Web services REST dentro de los sistemas empresariales.

En la Figura 18 se muestran las interacciones más comunes que se dan en la actualidad, en donde los clientes OWS interactúan directamente con los servidores de mapas y los clientes SOAP interactúan con los sistemas empresariales.

3.1.3 Escenarios

A continuación se describen tres escenarios de uso que son habituales dentro de este marco de trabajo. Estos escenarios surgen de plantear diferentes combinaciones entre consumidores (clientes OWS y SOAP) y proveedores de los servicios (servidores de mapas y sistemas empresariales).

Existe un caso que no es considerado dentro de estos escenarios, que es el de los clientes SOAP que se comunican únicamente con sistemas empresariales para consumir servicios de negocio, ya que ese escenario no plantea ningún tipo de interacción con los servicios geoespaciales, y por lo tanto no es de interés para este trabajo.

3.1.3.1 Escenario 1: Clientes OWS que requieren Servicios Geoespaciales

En este escenario (Figura 19), los clientes consumen únicamente servicios geoespaciales. Los más comunes son los servicios de mapas y de datos, aunque el escenario contempla también los servicios de metadatos y de procesamiento. Todos los servicios son publicados según estándares OWS.

---

[42] http://www.pmapper.net/

[43] http://www.geomajas.org/

[44] http://mapbender3.org/

[45] http://www.qgis.org

[46] http://www.gvsig.org

[47] https://www.arcgis.com

En relación a los escenarios que se presentan en [42], vemos que este escenario abarca los casos de "consulta de información pública" y "servicios para públicos específicos" (Sección 2.2.2.1), que involucran tanto servicios de mapas (WMS) como de datos (WFS básico), así como también el escenario de "colaboración para la generación de información geográfica," que involucra únicamente servicios de datos (WFS transaccional).

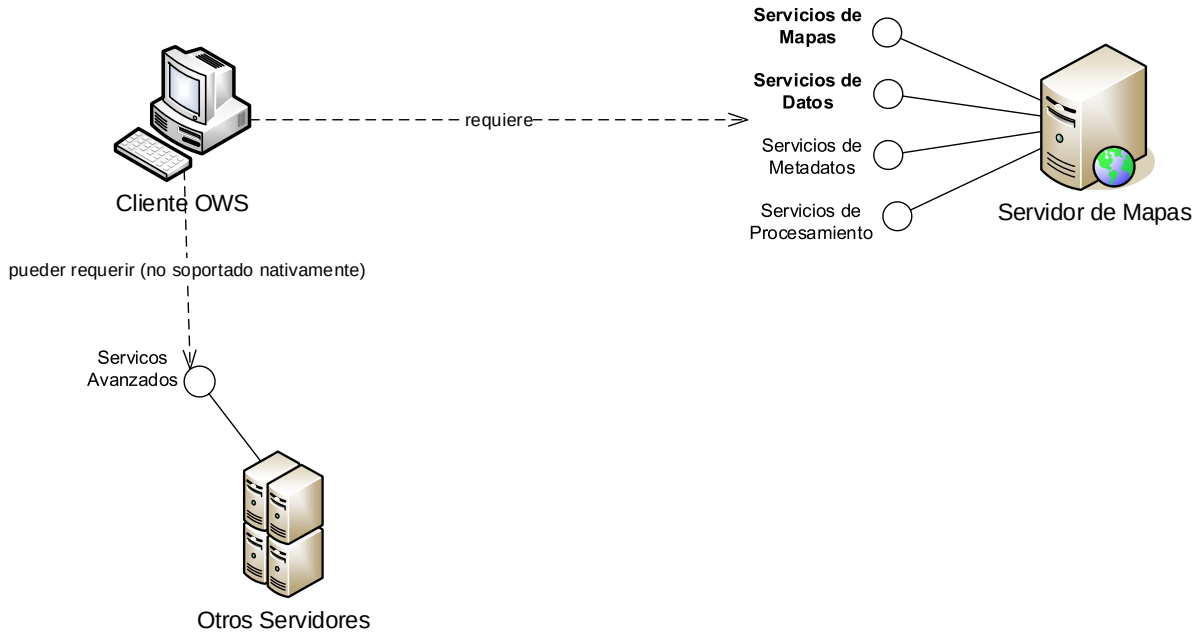


Figura 19 – Escenario 1

Este escenario puede ser desarrollado en base a las herramientas actuales con que se cuenta en un GIS empresarial, es decir, un cliente OWS que consume servicios de un servidor de mapas (Sección 3.1.2). Sin embargo, pueden existir requerimientos funcionales avanzados o requerimientos no funcionales que no puedan ser provistos por tales herramientas.

Como ejemplo de este escenario, podemos pensar en un visualizador de mapas Web que realiza consultas sobre entidades geográficas mediante el servicio WFS del servidor mapas. Estas consultas podrían estar limitadas de acuerdo a reglas impuestas por un servicio de autorización, que filtra los resultados devueltos al cliente (servicios avanzados).

3.1.3.2 Escenario 2: Clientes OWS que requieren Servicios Geoespaciales y Servicios de Negocio.

En este escenario (Figura 20), los clientes requieren consumir tanto servicios geoespaciales como servicios de negocio. Los servicios geoespaciales pueden ser consumidos de la misma forma que en el caso anterior, pero para consumir los servicios de negocio no existe ningún mecanismo estándar. Los clientes OWS esperan contar con una interfaz OWS del lado del servidor, por lo que, para hacer posible este escenario, es necesario contar con un mecanismo que soporte el enriquecimiento de la información geográfica con la información de negocio, sin alterar la interfaz del lado del servidor. Dado que se trata de un cliente OWS que trabaja con IG, la información de negocio debe ser integrada a la geográfica y no en sentido inverso.

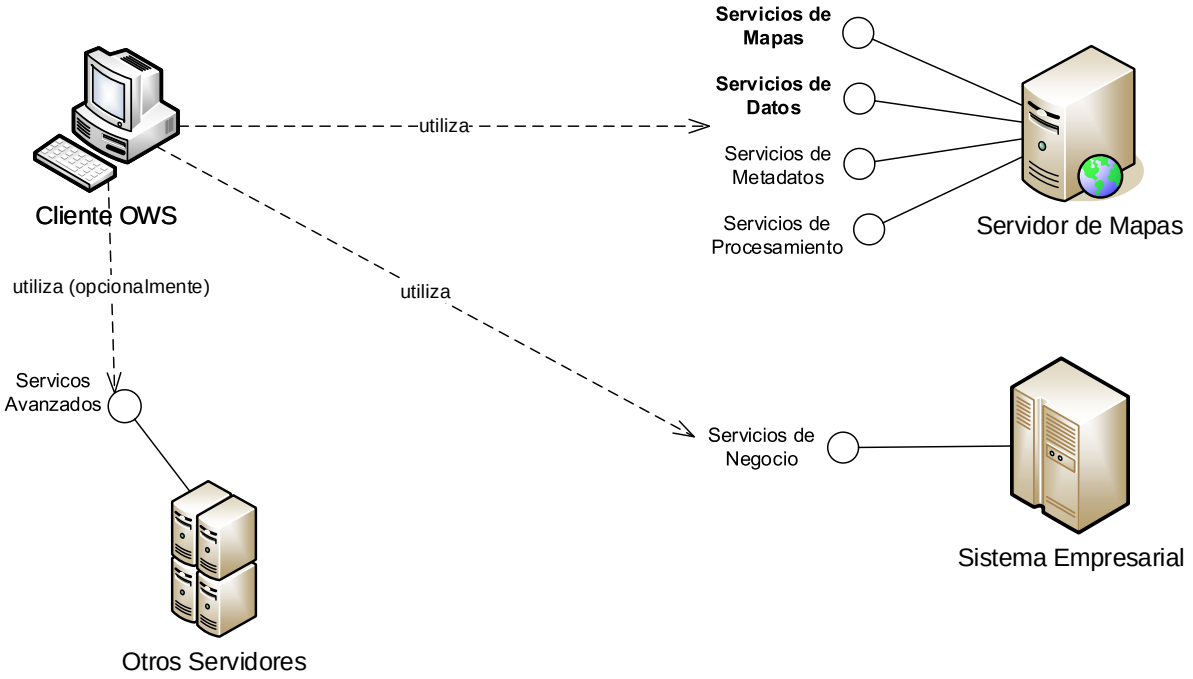


Figura 20 – Escenario 2

Como ejemplo de este escenario, podemos pensar en un visualizador de mapas Web que obtiene capas geográficas mediante el servicio WMS del servidor de mapas. Existen datos de negocio asociados a estas capas, que se obtienen mediante un Web service SOAP enviando el nombre de la capa y el identificador de una entidad de la misma. Otros servicios avanzados podrían limitar los resultados devueltos, como en el Escenario 1.

3.1.3.3 Escenario 3: Clientes SOAP que requieren Servicios de Negocio y Servicios Geográficos.

En este escenario (Figura 21), los clientes requieren consumir tanto servicios de negocio como servicios geoespaciales. Los servicios de negocio pueden ser consumidos de la misma forma que en el caso anterior, pero para consumir los servicios geoespaciales, surge el problema de incompatibilidad de protocolos, ya que los servidores de mapas no exponen generalmente interfaces SOAP de los servicios geoespaciales.

Un caso típico es el que encontramos en [42], en el escenario "colaboración para la realización de procesos administrativos." Los proceso administrativos que allí se mencionan, que involucran varios organismos comunicándose a través de Web services, son manejados en muchos casos por motores de orquestación de servicios basados en WS-BPEL, los que soportan nativamente la invocación de operaciones de Web services SOAP. Sin embargo, puede haber procesos en donde se necesite invocar un geoservicio que es proporcionado por un servidor de mapas. En este caso, al no contarse con capacidad de invocar servicios tipo REST, es necesario realizar una adaptación del geoservicio para que sea invocable desde estos motores.

Este escenario no solo abarca los procesos administrativos inter-organizacionales planteados en [42], sino que también contempla los procesos intra-organizacionales. En cualquier caso, los clientes SOAP podrían utilizar motores de ejecución de procesos de negocio basado en BPMN.

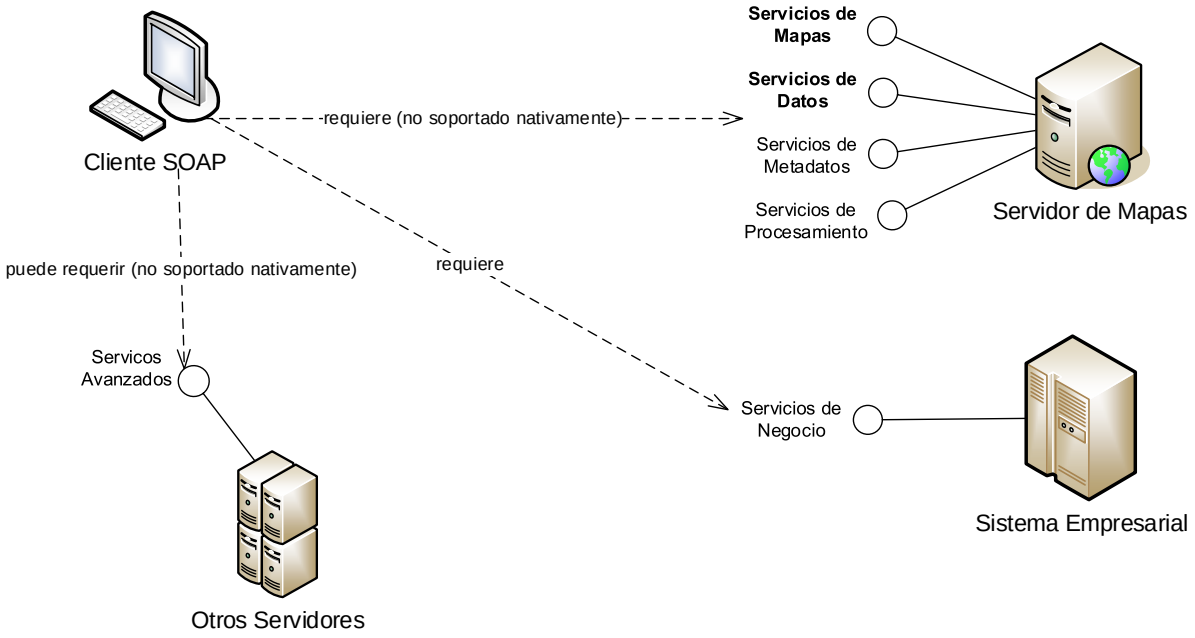


Figura 21 – Escenario 3

Como ejemplo de este escenario, puede pensarse en un motor externo de ejecución de servicios que utilice WS-BPEL para ejecutar una orquestación de Web services SOAP como parte de un proceso de negocio. En dicha orquestación, se obtienen primero los clientes que residen en una determinada área, realizando una consulta mediante WFS, y luego se solicita la facturación anual total correspondiente a esos clientes, mediante un Web service SOAP de negocio.

3.2 Requerimientos

En esta sección se analizan los requerimientos, tanto funcionales como no funcionales, que surgen de los escenarios analizados, y que por lo tanto sería deseable que fueran soportados por una plataforma de integración específica del dominio GIS.

3.2.1 Requerimientos funcionales

En base a lo antes expuesto, surge la necesidad de contar con un nuevo componente en el GIS empresarial (Componente de Integración) que permita solucionar las limitaciones mencionadas anteriormente y que a su vez, brinde un grado de extensibilidad suficiente para seguir dando respuesta a los desafíos que se planteen a corto y mediano plazo como resultado de las tendencias que ya se vislumbran en este momento.

Dentro de este trabajo, se pone el foco en construir una solución que cumpla con los siguientes requerimientos funcionales:

3.2.1.1 Enriquecer Información Geográfica con Información de Negocio.

La IG es un común denominador de gran valor en numerosos procesos de negocio [48]. Dado que la mayor parte de la información puede ser localizada geográficamente, la combinación de la IG (dónde están las cosas) y la información descriptiva (qué son las cosas) permite ver la información de negocio desde una perspectiva completamente nueva. [48]

Sin embargo, los GIS empresariales se apoyan generalmente en bases de datos geoespaciales que no poseen datos específicos de negocio, por lo que las aplicaciones que deben integrar datos de negocio en un mapa deben hacer una integración por fuera del servidor de mapas.

Existen diferentes alternativas para realizar dicha integración, por ejemplo las analizadas en [45]. En esta tesis se propone que la integración se realice a través de un componente de middleware, que es una de las alternativas que se analizan en dicho artículo.

En base a los diferentes tipos de servicios que hemos analizado, podemos refinar este requerimiento en diferentes casos, dentro de los que destacaremos los siguientes:

1. Enriquecimiento de Servicios de Mapas con Servicios de Negocio
2. Enriquecimiento de Servicios de Datos con Servicios de Negocio

3.2.1.2 Transformar Servicios Geoespaciales para Integrar en Procesos de Negocio

Como se describe en el Escenario 3, existen sistemas que se comunican naturalmente con Web services SOAP, como los motores WS-BPEL. La posibilidad de consumir Web services geoespaciales desde estos motores, brindaría una potencialidad que no ha sido explotada en la actualidad.

Surge entonces la necesidad de contar con un mecanismo que transforme los Web services geoespaciales en Web services SOAP, para de esta forma poder incluir los geoservicios en plataformas de tipo SOA que usen tecnologías avanzadas de Web services.

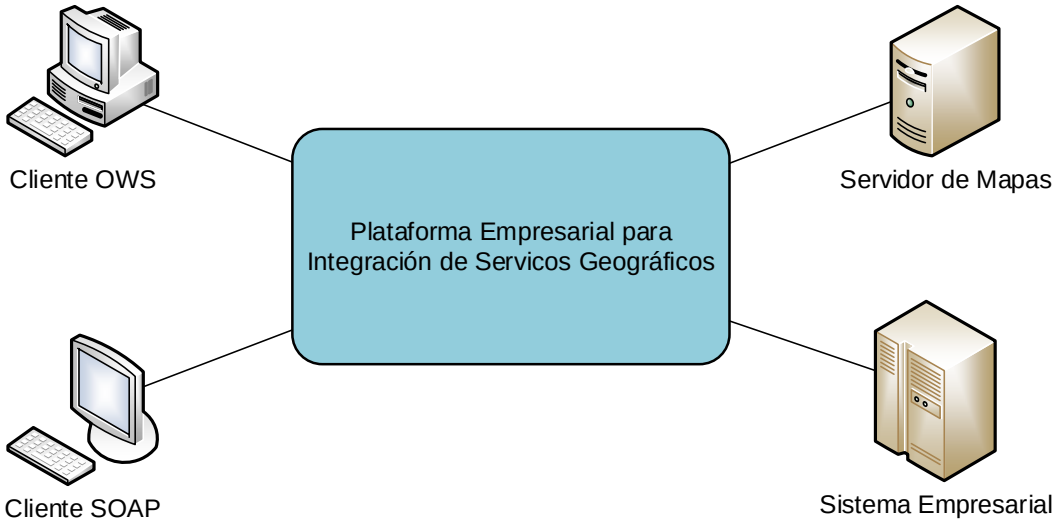


Figura 22 – Plataforma entre consumidores y proveedores

En la Figura 22 puede observarse cómo la plataforma actúa como intermediario entre los consumidores y los proveedores de servicios.

3.2.2 Requerimientos no funcionales

Además de los requerimientos funcionales, existen también requerimientos no funcionales que se consideran importantes dentro del marco de trabajo presentado. Una primera versión de esta plataforma debería cumplir con los siguientes requerimientos.

3.2.2.1 Maximizar el uso de estándares.

Se plantea como un requerimiento importante que la solución a desarrollar esté basada en estándares, tanto como sea posible. Dentro de los estándares son de particular importancia:

- Estándares OGC para servicios geoespaciales
- Estándares W3C y OASIS para servicios de negocio
- XML como formato preferido de intercambio
- Estándares basados en XML para realizar operaciones de mediación.

3.2.2.2 Proveer componentes reusables.

La reusabilidad de los componentes que formen parte de la solución es un requerimiento de diseño que permitirá que un mismo componente pueda ser usado en más de un escenario.

Es particularmente deseable que un mismo componente pueda aprovechar las similitudes que existen entre los diferentes geoservicios, especialmente los que corresponden a una misma categoría como se analizó en la Sección 3.1.1.

3.2.2.3 Permitir la parametrización de la solución sin requerir programación en lenguajes de alto nivel.

La solución debe permitir la realización de los requerimientos funcionales en base a la parametrización de los componentes, sin requerir que se escriba código en lenguajes de alto nivel.

3.2.2.4 Cubrir dos niveles de seguridad: autenticación y autorización.

Es una preocupación habitual en el despliegue de geoservicios dentro de una IDE o de un GIS empresarial, que no se cuenten con elementos de seguridad integrados en el diseño de los OWS. Por este motivo, es un requisito contar con mecanismos de seguridad que permitan imponer controles y restricciones sobre el uso de los mismos. Para esto es necesario contar, como mínimo, con algún mecanismo de autenticación y autorización que sea proporcionado por la propia plataforma, subsanando de esta manera esta carencia que presentan los estándares mencionados.

3.2.2.5 Mejorar la disponibilidad de servicios

Es un caso habitual dentro de una IDE, que la PGE sobre la que es desplegada posea una infraestructura con un poder de cómputo y un nivel de conectividad significativamente superior a la que ofrecen las organizaciones que publican sus servicios dentro de ella. En particular, utilizando la terminología de la arquitectura de la IDEuy, los nodos periféricos suelen ser muy heterogéneos, ya que cada uno depende de los recursos con que cuente la institución que los alberga. Es importante por lo tanto, poder contar dentro de la plataforma de integración, con mecanismos que puedan mejorar la calidad de servicio, y dentro de ésta, la disponibilidad de los mismos.

3.2.2.6 Mejorar el tiempo de respuesta de los servicios en los escenarios que sea posible.

Por los mismos motivos expuestos en el punto anterior, interesa mejorar el tiempo de respuesta de los servicios, de manera que el cliente que solicite un servicio a la plataforma perciba un tiempo de respuesta mejor y menos variable que el que se obtendría consumiendo directamente los servicios de cada proveedor.

3.2.2.7 No incrementar los tiempos de respuesta en forma significativa en los escenarios que sea posible.

En ciertos escenarios no será posible mejorar los tiempos de respuesta, ya que la plataforma brindará servicios con valor agregado que no sería posible obtener consumiendo directamente los servicios de cada proveedor. En

estos casos, que deberán ser analizados detalladamente, no se debería incurrir en un aumento superior al 50% sobre los tiempos de respuesta sin utilizar la plataforma.

3.2.2.8 Permitir la extensibilidad de la solución para incorporar características avanzadas.

Se busca que la solución esté diseñada para poder incorporar características avanzadas a los geoservicios. Dentro de estas, se identificaron como las más aplicables en este contexto la transaccionalidad, la mensajería confiable y el asincronismo. Estas características se encuentran disponibles en las plataformas empresariales y de gobierno electrónico a través de estándares como WS-Transaction o WS-ReliableMessaging que se aplican a Web services SOAP.

4 Propuesta de Solución

La propuesta central de este trabajo consiste en el diseño de una Plataforma Empresarial de Integración de Servicios Geoespaciales basada en ESB (GeoEEIP). El rol de la plataforma es el de actuar como un intermediario o *broker* entre clientes y servidores.

Al tratarse de una plataforma específica de dominio, es posible integrar en la misma un conjunto de mecanismos que den solución a requerimientos específicos, tanto funcionales (Sección 3.2.1) como no funcionales (Sección 3.2.2), dando soporte a los escenarios 1, 2 y 3 analizados anteriormente (Sección 3.1.3). Los mecanismos son, por lo tanto, lo suficientemente específicos como para satisfacer dichos requerimientos sin demandar un tiempo prolongado de adaptación y/o desarrollo, pero a su vez, los suficientemente genéricos como para no estar atados a las particularidades de un determinado consumidor o proveedor.

A grandes rasgos, el funcionamiento de la plataforma es como sigue. Cuando un cliente envía una solicitud a un servidor, esta es interceptada por la plataforma, se aplican los mecanismos de integración geoespaciales, y se rutean las solicitudes necesarias a los servidores finales. Con las respuestas de los servidores finales y los mecanismos de integración se forma una nueva respuesta "integrada" que se devuelve al cliente. Esta respuesta "integrada" posee un cierto valor agregado que no podría obtenerse consultando únicamente a los servidores finales sin agregar una lógica compleja ya sea a los clientes o a los servidores finales.

4.1 Arquitectura de Referencia

La Figura 23 presenta una vista de alto nivel de la plataforma GeoEEIP y los sistemas externos con los que interactúa. La misma puede verse como la suma entre un producto de ESB existente y una extensión compuesta por Mecanismos de Integración Geoespaciales.

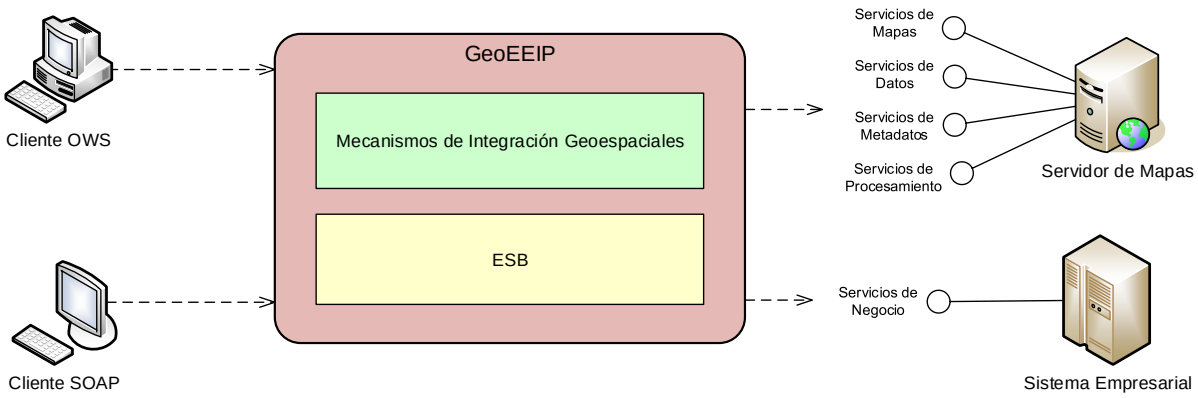


Figura 23 – Arquitectura de Referencia de la GeoEEIP

4.2 Solución Basada en Niveles de Refinamiento.

En la sección anterior se muestra la arquitectura de referencia de la plataforma, como una solución basada en ESB a la que se le agregan un conjunto de mecanismos de integración específicos del dominio. Sin embargo, la solución puede plantearse en una manera más abstracta, en base a diferentes niveles de abstracción de la plataforma, en donde se parte de un nivel inicial (nivel 0) y se van realizando sucesivos refinamientos que producen plataformas más concretas (Figura 24).

En el nivel 0, la plataforma es una caja negra que no brinda información sobre sus componentes internos, pero que queda definida parcialmente por los componentes con los que interactúa (consumidores y proveedores de servicios) y por los requerimientos que debe satisfacer.

En el nivel 1, la plataforma refina el nivel 0, incluyendo un componente de middleware concreto, el ESB, así como un conjunto de mecanismos de integración geoespaciales, que caracterizan a la GeoEEIP como una plataforma específica de dominio.

En el nivel 2, la plataforma refina el nivel 1, separando los mecanismos de integración geoespaciales en 2 capas (básico y compuestos) como se explicará a continuación.

Existe un último nivel, que es el nivel de implementación, en donde el ESB es instanciado por un producto concreto, y los mecanismos se definen en base a las particularidades del mismo, utilizando patrones de integración empresariales (Sección 2.1.6.1)

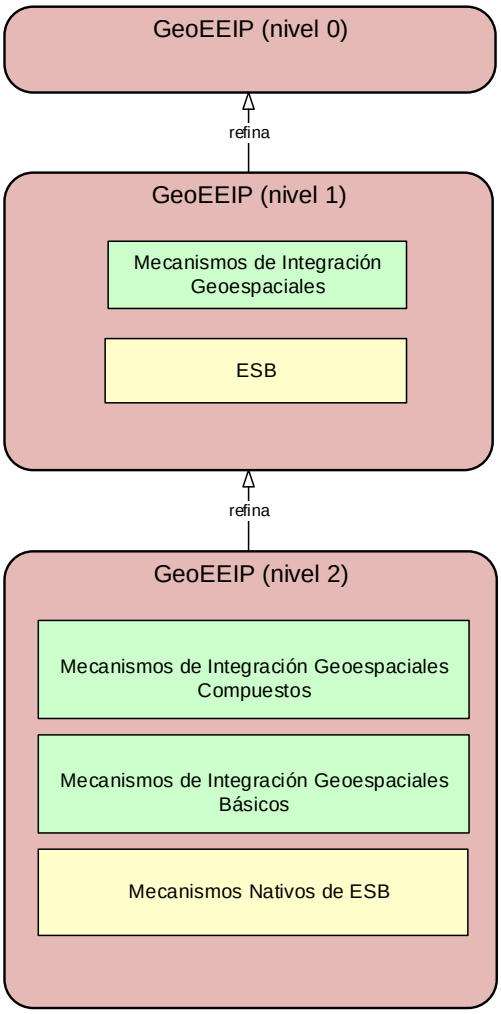


Figura 24 – Refinamiento en niveles de abstracción

Dentro de la plataforma, podemos identificar tres capas lógicas que agrupan los mecanismos con un nivel similar de abstracción.

La Capa 1 (Mecanismos Nativos de ESB) está formada por los mecanismos de mediación que son provistos por el mismo ESB. Estos mecanismos (ruteo, transformaciones, splitter, etc.) están disponibles en todos los principales productos de ESB existentes y proveen soluciones reusables para atacar requerimientos de integración y comunicación generales (Sección 2.1.6).

La Capa 2 (Mecanismos de Integración Geoespaciales Básicos) y la Capa 3 (Mecanismos de Integración Geoespaciales Compuestos) están formadas por los mecanismos de integración geoespaciales, que no forman parte de un ESB, sino que deben ser agregados a éste para construir la plataforma GeoEEIP.

La Capa 3 está compuesta por los Mecanismos Geoespaciales Compuestos. Cada uno de estos mecanismos, da una solución completa a un cierto requerimiento de integración de alto nivel, como los que se han analizado en la Sección 3.2.1.

La Capa 2 está compuesta por los Mecanismos Geoespaciales Básicos. Cada uno de estos mecanismos permite solucionar un requerimiento de integración en forma parcial. Como puede apreciarse en la arquitectura (Figura 25), los mecanismos de la Capa 3 pueden usar tanto mecanismos de la Capa 2 como de la Capa 1 para implementar su estrategia de integración. También es importante notar que los mecanismos de la Capa 2 pueden ser utilizados por diferentes mecanismos de la Capa 3, lo que exige que estos mecanismos sean diseñados con el nivel de abstracción y la granularidad necesaria para favorecer el reuso.

Cabe señalar que, en el caso de los mecanismos de integración geoespaciales (Capas 2 y 3), no existen dependencias entre los mecanismos de una misma capa. Por otro lado, en el caso de los mecanismos incorporados de ESB, las dependencias pueden ocurrir en el caso de patrones de mediación compuestos que utilizan otros patrones básicos. Éste es el caso del patrón Composed Message Processor presentado en [36], que realiza una composición de los patrones básicos Splitter, Router y Aggregator.

La existencia de mecanismos de mediación básicos y compuestos a nivel del ESB, podría llevar a pensar en dos niveles dentro de la Capa 1. Sin embargo, mecanismos compuestos como el mencionado u otros no se encuentran implementados en su totalidad en la mayoría de los productos de ESB, a diferencia de los mecanismos básicos, que cuentan con amplio soporte en estos productos y por lo tanto, son los mecanismos sobre los que se apoya esta solución.

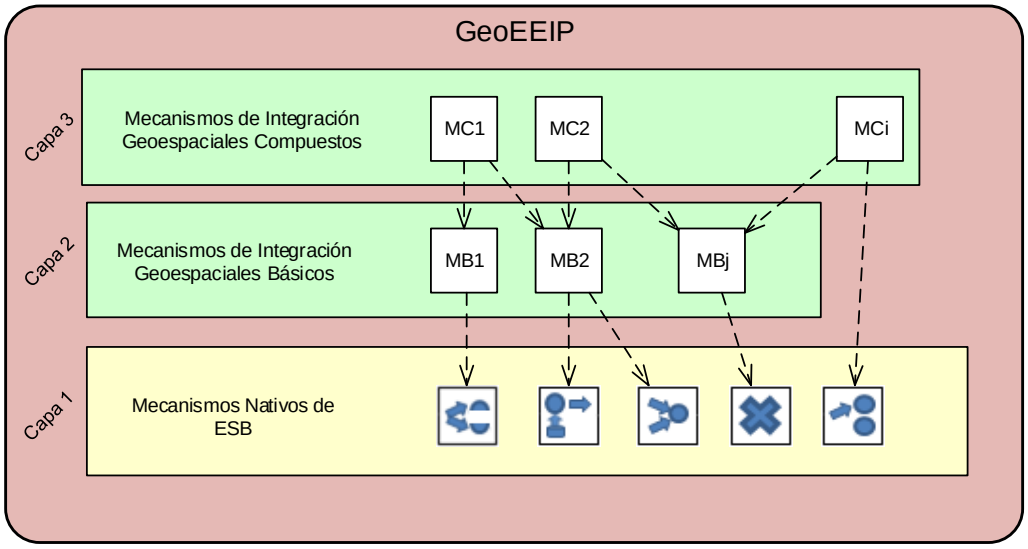


Figura 25 - GeoEEIP - Arquitectura lógica de 3 capas

En la siguiente sección de este capítulo, se detallan los mecanismos geoespaciales que se proponen inicialmente para integrar la GeoEEIP. Dicho conjunto de mecanismos es extensible, mediante la incorporación de nuevos

mecanismos que cumplan con la filosofía de diseño de la plataforma, por lo que deben ser tomados como un punto de partida lo suficientemente rico como para poder satisfacer los requerimientos definidos en la Sección 3.2.

4.3 Mecanismos de Integración Geoespaciales.

En esta sección se especifican mediante un diseño de alto nivel los mecanismos geoespaciales, siguiendo un enfoque *top-down*, comenzando con los mecanismos compuestos y siguiendo con los básicos. Los mecanismos compuestos son los que permiten definir las capacidades de cada instancia de la GeoEEIP, en tanto que los mecanismos básicos están orientados a descomponer la solución total en soluciones parciales, que puedan ser reutilizadas, y que a su vez sean *building blocks* reutilizables y más específicos que los que proporciona un determinado producto de ESB.

En la Tabla 4 se muestra la dependencia existente entre los mecanismo compuestos que cumplen requerimientos de enriquecimiento (ej. WMS Enricher) y los mecanismos básicos que son utilizados por estos (ej. GeoEntryPoiny).

Tabla 4 – Matriz de dependencias entre los mecanismos compuestos y básicos (sólo enriquecimiento)

Básico\Compuesto	**WMS Enricher**	**WMTS Enricher**	**WFS Enricher**	**WFS-T Enricher**
GeoEntryPoint	[]	[]	[]	[]
GeoProxy	[]	[]	[]	[]
GeoRouter	[]	[]	[]	[]
CapabilitiesAdapter	[]	[]	[]	[]
FeatureInfoAdapter	[]	[]		
FeatureTypeAdapter			[]	[]
FeatureAdapter			[]	[]
TransactionAdapter				[]

En la Tabla 5 se muestra la dependencia existente entre los mecanismo compuestos que cumplen requerimientos de adaptación (ej. SOAP-WMS Wrapper) y los mecanismos básicos que son utilizados por estos (ej. FeatureInfoAdapter).

Tabla 5 – Matriz de dependencias entre los mecanismos compuestos y básicos (sólo adaptación)

Básico\Compuesto	**SOAP-WMS Wrapper**	**SOAP-WMTS Wrapper**	**SOAP-WFS Wrapper**
GeoEntryPoint	[]	[]	[]
GeoProxy	[]	[]	[]
GeoRouter	[]	[]	[]
CapabilitiesAdapter	[]	[]	[]
FeatureInfoAdapter	[]	[]	
FeatureTypeAdapter			[]
FeatureAdapter			[]
TransactionAdapter			

Como puede apreciarse en estas tablas, es posible alcanzar un alto grado de reutilización de los mecanismos básicos. Por ejemplo, el mecanismo básico GeoRouter es utilizado por todos los mecanismos compuestos.

4.3.1 Mecanismos Geoespaciales Compuestos

Los mecanismos de integración compuestos (Tabla 6) son mecanismos de mediación de alto nivel que permiten resolver un determinado requerimiento como los analizados en la Sección 3.2.1 en base a un estándar OWS concreto (ej. WMS). A continuación se describen estos mecanismos.

Tabla 6 – Mecanismos Geoespaciales Compuestos según categoría funcional

Enriquecimiento de Servicios de Mapas con Servicios de Negocio	Enriquecimiento de Servicios de Datos con Servicios de Negocio	Adaptación de Servicios Geoespaciales
WMS Enricher WMTS Enricher	Basic WFS Enricher Transactional WFS Enricher	SOAP-WMS Wrapper SOAP-WMTS Wrapper SOAP-WFS Wrapper

4.3.1.1 WMS Enricher

El mecanismo WMS Enricher permite enriquecer la información descriptiva que se puede obtener mediante un servicio WMS con información de negocio proveniente de uno o varios servicios de negocio. En la Tabla 7 se hace un resumen de este mecanismo.

Tabla 7 – Ficha técnica de WMS Enricher

WMS Enricher	
Categoría funcional	Enriquecimiento de Servicios de Mapas con Servicios de Negocio
Motivación	Una organización posee un servidor de mapas que es accedido desde clientes externos a través de WMS. La organización desea complementar la IG que se publica mediante este servicio con información de negocio que es accesible mediante un Web service SOAP de negocio.
Restricciones	Dado que los clientes WMS son externos, no existe la posibilidad de que la organización los modifique para integrar la información de negocio, por lo que el mecanismo de integración debe ser transparente para el cliente.
Esbozo de solución	Se utiliza un mecanismo de mediación dentro de la GeoEEIP que consulta al servidor de mapas y enriquece las respuestas GetFeatureInfo con información de negocio a través de un Web service SOAP. Este mecanismo debe analizar cada solicitud WMS y, según cual sea la operación, debe generar los pedidos al servidor de mapas y al Web service de negocio para luego agregar las respuestas y enviar una respuesta enriquecida al cliente.
Ejemplo de aplicación	El Ministerio de Turismo posee un servidor de mapas que publica por WMS una capa de puntos de interés o *POIs* (hoteles, restaurantes, museos, etc.). Una empresa privada posee un Web service SOAP que brinda la cantidad de visitantes que hubo en el mes anterior por cada punto de interés. El ministerio desea que esa información se agregue como un atributo descriptivo más a la información que ya se muestra en el mapa.

En la Figura 26 se muestra el diseño a alto nivel del WMS Enricher, en base al uso de mecanismos básicos. Los mecanismos básicos serán explicados con mayor detalle en la Sección 4.3.2.

El funcionamiento del WMS Enricher es como sigue:

1. Un cliente envía un pedido WMS a un servicio virtual en el GeoEEIP como si se estuviera comunicando con un servidor de mapas.

2. GeoEntryPoint recibe el pedido HTTP y crea un nuevo mensaje de ESB conteniendo la dirección del servicio, el *query string*[48] del pedido original y un diccionario con las claves y los valores extraídos del *query string*.
3. GeoProxy recibe el mensaje ESB y envía al servidor de mapas el pedido WMS, utilizando el *query string* recibido. Cuando se recibe la respuesta del servidor, ésta se guarda en el mensaje ESB junto con la URL completa del pedido.
4. GeoRouter recibe el mensaje ESB y analiza el nombre de la operación invocada (GetCapabilites, GetMap o GetFeatureInfo) para determinar la siguiente acción.
5. Si la operación es GetMap, como es el resultado de la misma (un mapa) no necesita ser modificado, no se realiza ninguna otra acción.
6. Si la operación es GetCapabilities, entra en acción GeoCapabilitiesAdapter, que se encarga de adaptar ciertos parámetros de la respuesta, para que la intervención de la plataforma siga siendo transparente para el cliente.
7. Si la operación es GeoFeatureInfo, entra en acción GeoFeatureInfoEnricher, que se encarga de realizar el enriquecimiento de la respuesta.

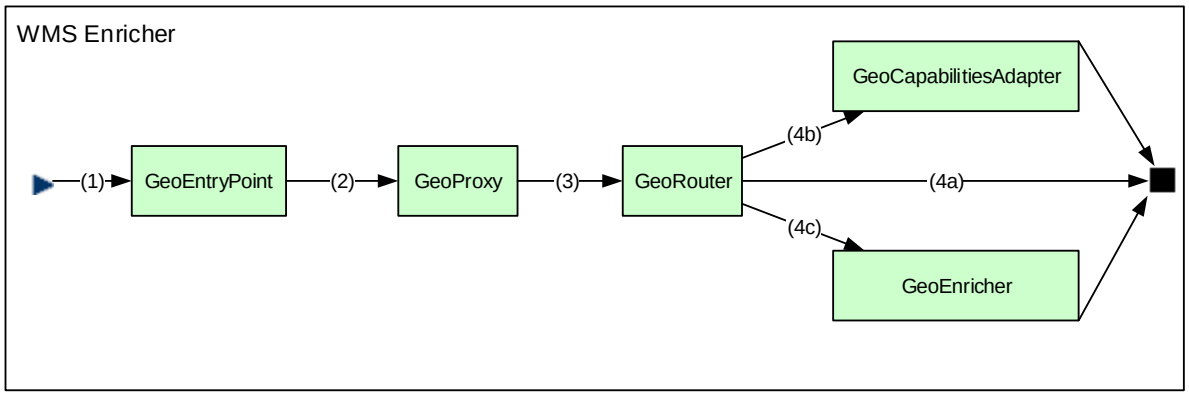


Figura 26 – Flujo de mediación de WMS Enricher

4.3.1.2 WMTS Enricher

El mecanismo WMTS Enricher permite enriquecer la información descriptiva que se puede obtener mediante un servicio WMTS con información de negocio proveniente de uno o varios servicios de negocio. Este mecanismo es análogo al anterior. En la Tabla 8 se hace un resumen de este mecanismo.

[48] Se llama *query string* al segmento de una URI (o URL) que contiene información no jerárquica de un recurso y que comienza luego del signo "?" como una sucesión de pares "clave=valor" (http://tools.ietf.org/html/rfc3986#section-3.4)

Tabla 8 – Ficha técnica de WMTS Enricher

WMTS Enricher	
Categoría funcional	Enriquecimiento de Servicios de Mapas con Servicios de Negocio
Motivación	Una organización posee un servidor de mapas que es accedido desde clientes externos a través de WMTS. La organización desea complementar la IG que se publica mediante este servicio con información de negocio que es accesible mediante un Web service SOAP de negocio.
Restricciones	Dado que los clientes WMTS son externos, no existe la posibilidad de que la organización los modifique para integrar la información de negocio, por lo que el mecanismo de integración debe ser transparente para el cliente.
Esbozo de solución	Se utiliza un mecanismo de mediación dentro de la GeoEEIP que consulta al servidor de mapas y enriquece las respuestas GetFeatureInfo con información de negocio a través de un Web service SOAP. Este mecanismo debe analizar cada solicitud WMS y, según cual sea la operación, debe generar los pedidos al servidor de mapas y al Web service de negocio para luego agregar las respuestas y enviar una respuesta enriquecida al cliente.
Ejemplo de aplicación	El Ministerio de Turismo posee un servidor de mapas que publica por WMS una capa de puntos de interés o *POIs* (hoteles, restaurantes, museos, etc.). Una empresa privada posee un Web service SOAP que brinda la cantidad de visitantes que hubo en el mes anterior por cada punto de interés. El ministerio desea que esa información se agregue como un atributo descriptivo más a la información que ya se muestra en el mapa.

En la Figura 27 se muestra el diseño a alto nivel de WMTS Enricher, en base al uso de mecanismos básicos. Como puede observarse, el diseño del mecanismo es análogo al de WMS Enricher, ya que utiliza los mismos mecanismos básicos y con la misma lógica de control, variando simplemente en la configuración del mismo.

El funcionamiento de WMTS Enricher es como sigue:

1. Un cliente envía un pedido WMTS a un servicio virtual (Sección 2.1.6.1) en el GeoEEIP como si se estuviera comunicando con un servidor de mapas.
2. GeoEntryPoint recibe el pedido HTTP y crea un nuevo mensaje de ESB conteniendo la dirección del servicio, el *query string* del pedido original y un diccionario con las claves y los valores extraídos del *query string*.
3. GeoProxy recibe el mensaje ESB y envía al servidor de mapas el pedido WMS, utilizando el *query string* recibido. Cuando se recibe la respuesta del servidor, ésta se guarda en el mensaje ESB junto con la URL completa del pedido.
4. GeoRouter recibe el mensaje ESB y analiza el nombre de la operación invocada (GetCapabilites, GetTile o GetFeatureInfo) para determinar la siguiente acción.
5. Si la operación es GetTile, como es el resultado de la misma (un mapa) no necesita ser modificado, no se realiza ninguna otra acción.
6. Si la operación es GetCapabilities, entra en acción GeoCapabilitiesAdapter, que se encarga de adaptar ciertos parámetros de la respuesta, para que la intervención de la plataforma siga siendo transparente para el cliente.
7. Si la operación es GeoFeatureInfo, entra en acción GeoEnricher, que se encarga de realizar el enriquecimiento de la respuesta.

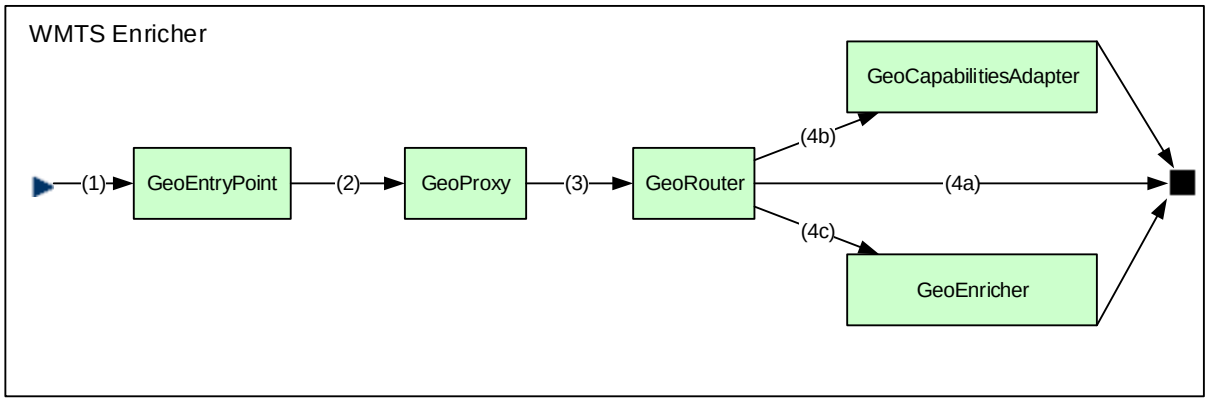


Figura 27 – Flujo de mediación de WMTS Enricher

4.3.1.3 Basic WFS Enricher

El mecanismo Basic WFS Enricher permite enriquecer la información descriptiva que se puede obtener mediante un servicio WFS básico (sólo-lectura) con información de negocio proveniente de uno o varios servicios de negocio. Este mecanismo es análogo al anterior. En la Tabla 9 se hace un resumen de este mecanismo.

Tabla 9 – Ficha técnica de Basic WFS Enricher

Basic WFS Enricher	
Categoría funcional	Enriquecimiento de Servicios de Datos con Servicios de Negocio
Motivación	Una organización posee un servidor de mapas que es accedido desde clientes externos a través de WFS. La organización desea complementar la IG que se publica mediante este servicio con información de negocio que es accesible mediante un Web service SOAP de negocio.
Restricciones	Dado que los clientes WFS son externos, no existe la posibilidad de que la organización los modifique para integrar la información de negocio, por lo que el mecanismo de integración debe ser transparente para el cliente.
Esbozo de solución	Se utiliza un mecanismo de mediación dentro de la GeoEEIP que consulta al servidor de mapas y enriquece las respuestas GetFeature con información de negocio a través de un Web service SOAP. Este mecanismo debe analizar cada solicitud WFS y, según cual sea la operación, debe generar los pedidos al servidor de mapas y al Web service de negocio para luego agregar las respuestas y enviar una respuesta enriquecida al cliente.
Ejemplo de aplicación	El Ministerio de Turismo posee un servidor de mapas que publica por WMS una capa de puntos de interés o *POIs* (hoteles, restaurantes, museos, etc.). Una empresa privada posee un Web service SOAP que brinda la cantidad de visitantes que hubo en el mes anterior por cada punto de interés. El ministerio desea que esa información se agregue como un atributo descriptivo más a la información que ya se muestra en el mapa.

En la Figura 28 se muestra el diseño a alto nivel de Basic WFS Enricher, en base al uso de mecanismos básicos.

El flujo de mediación de Basic WFS Enricher es como sigue:

1. Un cliente envía un pedido WFS a un servicio virtual en el GeoEEIP como si se estuviera comunicando con un servidor de mapas.

2. GeoEntryPoint recibe el pedido HTTP y crea un nuevo mensaje de ESB conteniendo la dirección del servicio, el *query string* del pedido original y un diccionario con las claves y los valores extraídos del *query string*.
3. GeoProxy recibe el mensaje ESB y envía al servidor de mapas el pedido WFS, utilizando el *query string* recibido. Cuando se recibe la respuesta del servidor, ésta se guarda en el mensaje ESB junto con la URL completa del pedido.
4. GeoRouter recibe el mensaje ESB y analiza el nombre de la operación invocada (GetCapabilites, DescribeFeatureType o GetFeature) para determinar la siguiente acción.
5. Si la operación es GetCapabilities, entra en acción GeoCapabilitiesAdapter, que se encarga de adaptar ciertos parámetros de la respuesta, para que la intervención de la plataforma siga siendo transparente para el cliente.
6. Si la operación es DescribeFeatureType, entra en acción GeoFeatureTypeAdapter, que se encarga adaptar la definición del tipo de entidad (esquema), para que la intervención de la plataforma siga siendo transparente para el cliente.
7. Si la operación es GetFeature, entra en acción GeoFeatureEnricher, que se encarga de realizar el enriquecimiento de la respuesta.

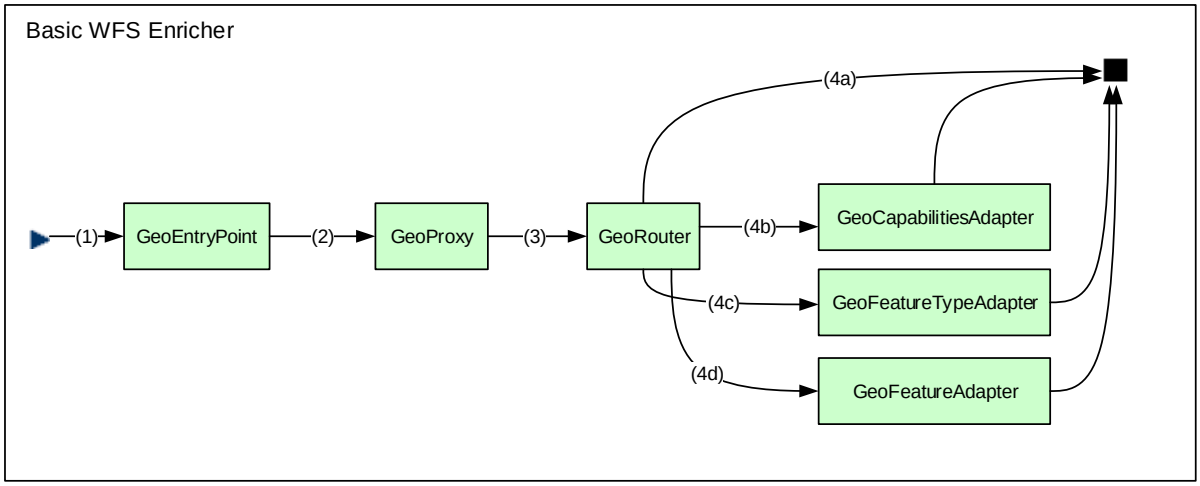


Figura 28 – Flujo de mediación de Basic WFS Enricher

4.3.1.4 SOAP-WMS Wrapper

El mecanismo SOAP-WMS Wrapper es un mecanismo que permite publicar servicios WMS siguiendo los estándares W3C (SOAP y WSDL). De esta manera, un servicio WMS puede ser invocado desde un cliente SOAP como si se tratase de cualquier otro Web service de negocio. En la Tabla 10 se hace un resumen de este mecanismo.

Tabla 10 – Ficha técnica de SOAP-WMS Wrapper

SOAP-WMS Wrapper	
Categoría funcional	Adaptación de Servicios Geoespaciales
Motivación	Una organización necesita incorporar un servidor de mapas en su infraestructura existente de Web services, de manera de poder acceder a la información geográfica y los geoservicios desde sus aplicaciones y procesos de negocio.
Restricciones	Dado que los procesos y aplicaciones de negocio ya consumen Web services SOAP y que se utilizan además estándares avanzados WS-*, no es posible o recomendable la publicación de geoservicios WMS en su forma REST.
Esbozo de solución	Un flujo de mediación que encapsula WMS utilizando SOAP se agrega a la GeoEEIP. Este flujo se encarga de manejar el pedido SOAP-WMS, transformarlo a WMS estándar, enviarlo al servidor de mapas, recibir la respuesta, transformarla a SOAP-WMS y devolverla al cliente.
Ejemplo de aplicación	El Ministerio de Industria, Energía y Minería ha implementado un BPMS, que utiliza Web services SOAP, para manejar sus procesos de negocios. Dentro de estos procesos se encuentra el otorgamiento de permisos para prospección, exploración y explotación minera. En un primer esfuerzo por incorporar funcionalidades geográficas en el sistema, se definió que el analista tenga la capacidad de visualizar, como parte integrada del proceso de negocio, el área sobre la cual se solicita un permiso junto con otras capas necesarias para el análisis, como por ejemplo la capa de áreas protegidas del país (donde la minería está prohibida). Esto puede ser resuelto por WMS.

En la Figura 29 se muestra el diseño a alto nivel de SOAP-WMS Wrapper, en base al uso de mecanismos básicos.

El flujo de mediación es como sigue:

1. Un cliente envía un pedido SOAP-WMS a un servicio virtual en la GeoEEIP.
2. GeoEntryPoint recibe el pedido HTTP y crea un nuevo mensaje de ESB conteniendo la dirección del servicio, el *query string* del pedido original y un diccionario con las claves y los valores extraídos del *query string* (1).
3. SOAP2WMSTranslator recibe el mensaje ESB, realiza la transformación del pedido SOAP-WMS a un pedido WMS estándar y guarda el nuevo pedido en el mensaje ESB (2).
4. GeoProxy recibe el mensaje ESB y envía al servidor de mapas el pedido WMS, utilizando el *query string* recibido. Cuando se recibe la respuesta del servidor, ésta se guarda en el mensaje ESB junto con la URL completa del pedido (3).
5. GeoRouter recibe el mensaje ESB y analiza el nombre de la operación invocada (GetCapabilites, GetTile o GetFeatureInfo) para determinar la siguiente acción (4).
6. Si la operación es GetCapabilities o GetFeatureInfo, que devuelven la respuesta en formato texto, entra en acción WMS2SOAPTextTranslator, que se encarga de decodificar la respuesta y crear un paquete SOAP para devolver al cliente (5a).
7. Si la operación es GetMap, que devuelven la respuesta en formato texto, entra en acción WMS2SOAPBinaryTranslator, que se encarga de decodificar la respuesta y crear un paquete SOAP para devolver al cliente (5b).

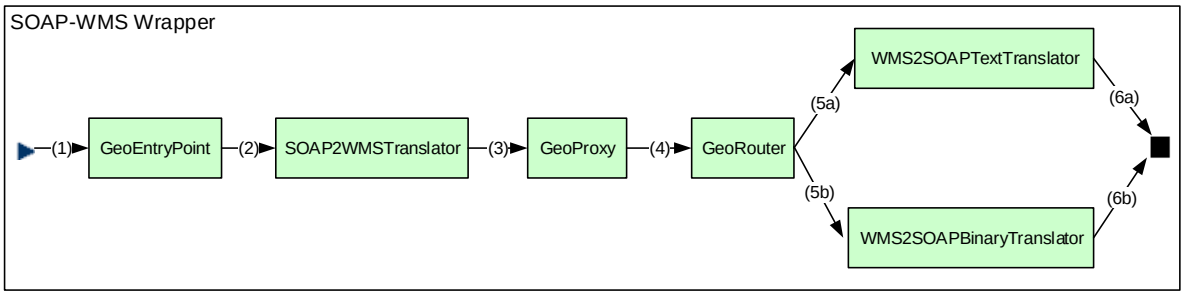


Figura 29 – Flujo de mediación de SOAP-WMS Wrapper

4.3.1.5 SOAP-WMTS Wrapper

El mecanismo SOAP-WMTS Wrapper es un mecanismo que permite publicar servicios WMTS siguiendo los estándares W3C (SOAP y WSDL). De esta manera, un servicio WMTS puede ser invocado desde un cliente SOAP como si se tratase de cualquier otro Web service de negocio. En la Tabla 11 se hace un resumen de este mecanismo.

Tabla 11 – Ficha técnica de SOAP-WMTS Wrapper

SOAP-WMS Wrapper	
Categoría funcional	Adaptación de Servicios Geoespaciales
Motivación	Una organización necesita incorporar un servidor de mapas en su infraestructura existente de Web services, de manera de poder acceder a la información geográfica y los geoservicios desde sus aplicaciones y procesos de negocio.
Restricciones	Dado que los procesos y aplicaciones de negocio ya consumen Web services SOAP y que se utilizan además estándares avanzados WS-*, no es posible o recomendable la publicación de geoservicios WMTS en su forma REST.
Esbozo de solución	Un flujo de mediación que encapsula WMTS utilizando SOAP se agrega a la GeoEEIP. Este flujo se encarga de manejar el pedido SOAP-WMTS, transformarlo a WMTS estándar, enviarlo al servidor de mapas, recibir la respuesta, transformarla a SOAP-WMTS y devolverla al cliente.
Ejemplo de aplicación	El Ministerio de Industria, Energía y Minería ha implementado un BPMS, que utiliza Web services SOAP, para manejar sus procesos de negocios. Dentro de estos procesos se encuentra el otorgamiento de permisos para prospección, exploración y explotación minera. En un primer esfuerzo por incorporar funcionalidades geográficas en el sistema, se definió que el analista tenga la capacidad de visualizar, como parte integrada del proceso de negocio, el área sobre la cual se solicita un permiso junto con otras capas necesarias para el análisis, como por ejemplo la capa de áreas protegidas del país (donde la minería está prohibida). Esto puede ser resuelto por WMTS.

4.3.1.6 SOAP-WFS Wrapper

El mecanismo SOAP-WFS Wrapper es un mecanismo que permite publicar servicios WFS siguiendo los estándares W3C (SOAP y WSDL). De esta manera, un servicio WFS puede ser invocado desde un cliente SOAP como si se tratase de cualquier otro Web service de negocio.

Tabla 12 – Ficha técnica de SOAP-WFS Wrapper

SOAP-WFS Wrapper	
Categoría funcional	Adaptación de Servicios Geoespaciales
Motivación	Una organización necesita incorporar un servidor de mapas en su infraestructura existente de Web services, de manera de poder acceder a la información geográfica y a los geoservicios desde sus aplicaciones y procesos de negocio.
Restricciones	Dado que los procesos y aplicaciones de negocio ya consumen Web services SOAP y que se utilizan además estándares avanzados WS-*, no es posible o recomendable la publicación de geoservicios WFS en su forma REST.
Esbozo de solución	Un flujo de mediación que encapsula WFS utilizando SOAP se agrega a la GeoEEIP. Este flujo se encarga de manejar el pedido SOAP-WFS, transformarlo a WFS estándar, enviarlo al servidor de mapas, recibir la respuesta, transformarla a SOAP-WFS y devolverla al cliente.
Ejemplo de aplicación	El Ministerio de Industria, Energía y Minería ha implementado un BPMS, basado en Web services SOAP, para manejar sus procesos de negocios. Dentro de estos procesos se encuentra el otorgamiento de permisos para prospección, exploración y explotación minera. Una actividad dentro de este proceso consiste en analizar si el área sobre la que se solicita un permiso cumple con ciertas restricciones (ej. no tiene intersecciones con áreas protegidas o con permisos ya otorgados). Esto puede ser resuelto mediante WFS.

4.3.2 Mecanismos Geoespaciales Básicos

Los mecanismos geoespaciales básicos permiten solucionar un requerimiento de integración en forma parcial, utilizando su propia lógica así como los mecanismos de mediación provistos por el ESB.

4.3.2.1 GeoEntryPoint

GeoEntryPoint brinda un punto de entrada a la plataforma GeoEEIP para el procesamiento inicial de una petición HTTP.

4.3.2.2 GeoProxy

GeoProxy brinda la capacidad de invocar las operaciones sobre un servidor de mapas.

4.3.2.3 GeoRouter

GeoRouter sigue el patrón CBR para direccionar un mensaje de acuerdo a la operación geoespacial que fue invocada en la petición original.

4.3.2.4 GeoFeatureInfoEnricher

GeoFeatureInfoEnricher utiliza transformaciones de mensajes para agregar información proveniente de servicios de negocio en la respuesta de una operación GetFeatureInfo.

4.3.2.5 GeoCapabilitiesAdapter

GeoCapabilitiesAdapter utiliza transformaciones de mensajes para adaptar algunos parámetros de la respuesta de una operación GetCapabilities. Un uso típico de este mecanismo consiste en sustituir las URLs de las operaciones de un servidor de mapas externo por las URLs publicadas por la GeoEEIP.

4.3.2.6 GeoFeatureTypeAdapter

GeoCapabilitiesAdapter utiliza transformaciones de mensajes para adaptar el esquema XML de un tipo de entidad geográfica (*feature type*) de acuerdo al enriquecimiento de datos que se haya realizado para dicha entidad, generando nuevos atributos que deben ser reflejados en su esquema para que este siga siendo válido.

4.3.2.7 GeoFeatureEnricher

GeoFeatureEnricher utiliza transformaciones de mensajes para agregar información proveniente de servicios de negocio en la respuesta de una operación GetFeature.

4.3.2.8 SOAP2WMSTextTranslator

SOAP2WMSTextTranslator realiza un *parseo* de un mensaje SOAP que contiene la invocación de una operación WMS y con los parámetros extraídos construye un mensaje WMS en formato estándar.

4.3.2.9 WMS2SOAPTextTranslator

WMS2SOAPTextTranslator construye un mensaje SOAP a partir de los parámetros extraídos de un mensaje WMS de tipo texto (XML, HTML, etc.) en formato estándar.

4.3.2.10 WMS2SOAPBinaryTranslator

WMS2SOAPBinaryTranslator construye un mensaje SOAP a partir de los parámetros extraídos de un mensaje WMS de tipo binario (PNG, JPEG, etc.) en formato estándar.

4.3.3 Extensiones no funcionales de los mecanismos.

Los mecanismos presentados anteriormente permiten responder a las necesidades funcionales que se han considerado primordiales dentro del contexto de aplicación analizado. Sin embargo, existen otras necesidades, también recurrentes en dicho contexto, que se circunscriben dentro de los llamados *cross-cutting concerns*[49], que son en esencia no funcionales, y que por lo tanto deberían poder combinarse con los mecanismos vistos hasta aquí. En las siguientes secciones se proponen dos mecanismos de mediación, que responden a requerimientos no funcionales, como la mejora de la performance o la seguridad. En el diseño de estos componentes, se tiene en cuenta que puedan combinarse con los mecanismo ya vistos. De esta manera, por ejemplo, podría configurarse la plataforma para que en un caso ofrezca servicios que incluyan autenticación, autorización, caché, transformación y enriquecimientos de mensajes. En la Figura 30 se muestra cómo interactúan los mecanismos funcionales y no funcionales dentro de la GeoEEIP, así como los componentes externos. Cuando un mensaje llega a la plataforma, a través de alguno de los servicios virtuales publicados, este pasa por un Punto de Entrada que se encarga de extraer los parámetros necesarios del pedido para los siguientes mecanismos, creando un mensaje ESB. A continuación, se acciona el Ruteo Inicial, que en base a la configuración existente, determina por qué mecanismos debe pasar el mensaje a continuación (Autenticación, Autorización, etc.) Los siguientes mecanismos que pueden

[49] https://msdn.microsoft.com/en-us/library/ee658105.aspx

accionarse, en forma opcional, son el de Autenticación, que se comunica con un Security Token Service (STS) para validar el token de seguridad recibido en el pedido, el de Autorización, que se comunica con un Policy Decision Point (PDP) para interrogarlo sobre los permisos que tiene el usuario autenticado de realizar el pedido, el de Caché para consultar si el pedido tiene una respuesta guardada, y en caso que no la tenga, accionar uno o varios Mecanismos Geoespaciales Compuestos de tipo funcional (WMSEnricher, SOAP2WMSWrapper, etc.)

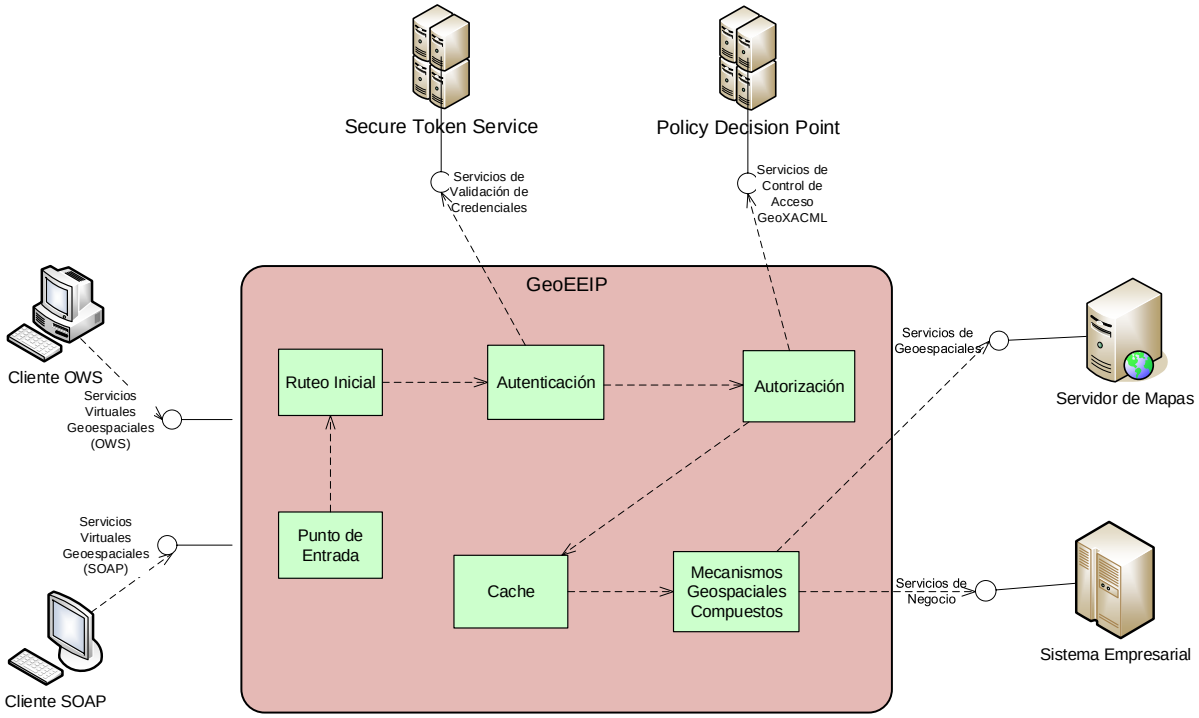


Figura 30 – Composición de varios tipos de mecanismos

4.3.3.1 Cache Mediator

El patrón de mediación con Caché presentado en [50], se plantea como una solución a los problemas de performance que son típicos en una SOA, ubicando este mecanismo en el middleware de mensajería, entre los productores y los consumidores de servicios. Los autores hacen hincapié en los siguientes problemas:

- La serialización mediante XML que se utilizan en los Web services genera un payload mucho más grande que el que utilizan otras tecnologías de invocación remota, lo que genera más tráfico en la red
- El empaquetado y *parsing* de XML es una tarea que consume tiempo adicional
- En una SOA, los consumidores y proveedores se encuentran muy débilmente acoplados, pueden traspasar la frontera de una organización y hallarse geográficamente muy alejados.
- La utilización de patrones de mediación a nivel del middleware (ruteos, transformaciones, logueos, etc.) degradan aún más la performance.

Como puede verse, la propuesta encaja perfectamente en el contexto de aplicación de este trabajo, por lo que se toma como referencia para el diseño de un mecanismo que pueda responder al requerimiento no funcional explicado en 3.2.2.5.

En la Figura 31 pueden verse los componentes que forman este patrón y su interacción que se describe resumidamente a continuación. Inicialmente, el consumidor envía el pedido al ESB (1 o a) que es interceptado por el Cache Request Mediation, el cual genera una clave con el pedido y busca el par (clave, respuesta) en el caché. Si se encuentra la respuesta (lo que se llama un *hit* del cache), esta se retorna al consumidor (b). En caso contrario

(un *miss* del caché), se envía el pedido al proveedor (2) y se genera una nueva entrada en el caché con la clave generada. Cuando el proveedor retorna la respuesta, se invoca al Cache Response Mediation (3), el que se ocupa de actualizar el caché completando el par de la clave ya creada y devolviendo la respuesta al consumidor (4).

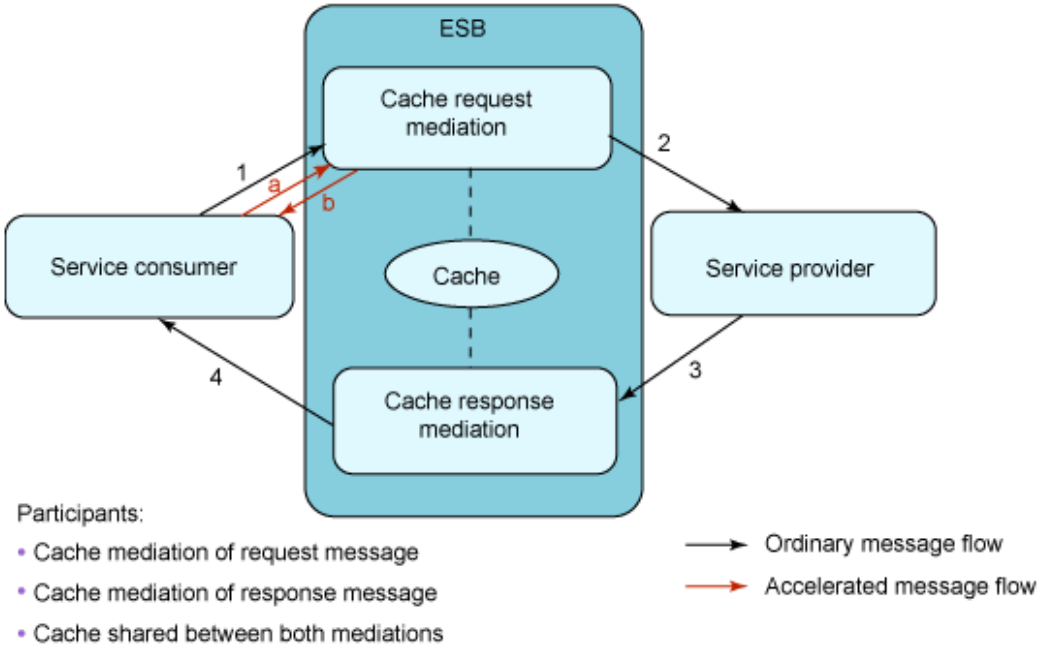


Figura 31 – Estructura del patrón de mediación Caché [50]

Dentro de la GeoEEIP, se consideraron tres alternativas para la incorporación de este patrón, siguiendo la discusión brindada en [51] sobre el diseño de un *gateway de adaptación*. La primera alternativa consiste en utilizar la facilidad de los interceptores o filtros que vienen incorporados a los ESB para capturar todos los mensajes que llegan al ESB y hacerlos pasar por una mediación de Caché. La segunda alternativa es modificar los servicios virtuales para que incluyan como primera operación la mediación de caché (en el caso que se configure con esta característica) y utilicen los mecanismos de integración geoespaciales como servicios. La tercera alternativa es crear un único servicio virtual que reciba todos los pedidos y que aplique la mediación de Caché, para luego invocar al servicio real de acuerdo al destino del mensaje.

Según [51], la primera alternativa tiene el problema de ser poco genérica ya que está atada a cada producto de ESB, y la segunda alternativa tiene el problema de que debe incorporarse a cada servicio virtual por separado, por lo que se inclina por la tercera solución. Sin embargo, vemos que ésta necesita conocer el destino de cada mensaje, utilizando por ejemplo WS-Addressing. Ya que los Web services geoespaciales no cuentan con esta posibilidad, sería imposible distinguir en un único servicio virtual qué mensajes son enviados a qué proveedor. Por lo tanto, se opta por la segunda alternativa.

En la Figura 32 puede observarse cómo se integra la mediación de Caché con un mecanismo de integración geoespacial. En esta alternativa de diseño, no es necesario modificar el diseño interno del mecanismo, sino que en vez de publicar el mecanismo de integración (ej, WMS Enricher) como un servicio virtual, se genera otro servicio virtual, que invoca al Cache Mediator y dentro este se configura como destino un mecanismo de integración, en lugar de un servicio real (siguiendo el ejemplo, esto sería un WMS Enricher with Cache).

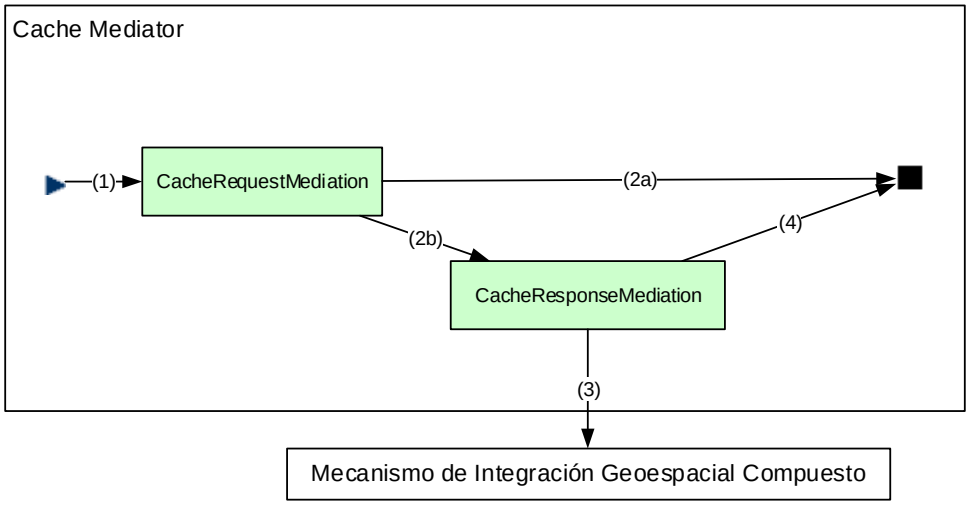


Figura 32 - Mecanismo de Integración Geoespacial con Caché

4.3.3.2 OWSAuthorization

OWSAuthorization en un mecanismo que permite brindar la capacidad de autorización basada en estándares (en este caso, mediante GeoXACML) y que a su vez pueda ser combinado con los mecanismos de integración, incluyendo las variantes con Caché. Se trata del componte PEP visto en la arquitectura de referencia de GeoXACML (Sección 2.1.5.5) que es integrado a la GeoEEIP. En la Figura 33 se muestra el flujo de mediación de este mecanismo. Se considera que el PDP es un servicio externo, posiblemente fuera de la GeoEEIP. Como puede observarse, el OWSAuthorization está construido en base a mecanismos geoespaciales básicos, al igual que los mecanismos de integración compuestos, pero no se considera un mecanismo de integración por cumplir una función distinta a la que cumplen estos.

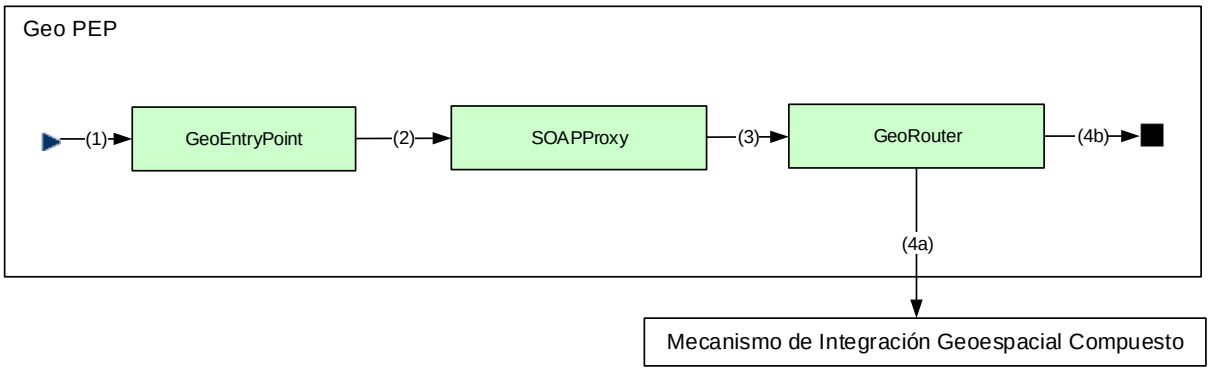


Figura 33 – Mecanismo de Integración Geoespacial con Control de Acceso

A continuación se describe el funcionamiento de este mecanismo:

- Un cliente envía un pedido a un servicio virtual en el GeoEEIP como si se estuviera comunicando con un servidor de mapas.

- GeoEntryPoint (1) recibe el pedido HTTP y crea un nuevo mensaje de ESB conteniendo la dirección del servicio, el *query string*[50] del pedido original y un diccionario con las claves y los valores extraídos del *query string*.
- SOAPProxy (2) recibe el mensaje ESB, crea un mensaje SOAP con el elemento Request según GeoXACML y lo envía al servidor PDP, utilizando el *query string* recibido. Cuando se recibe la respuesta del servidor, ésta se guarda en el mensaje ESB junto con la URL completa del pedido.
- GeoRouter (3) recibe el mensaje ESB y analiza el tipo de respuesta recibida (Permit, Deny, Indeterminate, Not Applicable) para determinar la siguiente acción. Si la decisión es Permit (4a), se invoca la operación original del mensaje en el mecanismo de integración configurado.
- En cualquiera de los otros casos (4b), se devuelve una respuesta vacía al cliente.

4.3.3.3 Autenticación

El mecanismo propuesto de autenticación está basado en la utilización de un Security Token Service (STS) externo. Se trata de un proveedor de identidad responsable de emitir *tokens* de seguridad, como parte de un sistema de identidad basada en notificaciones. El STS está encargado de proveer un token SAML, que luego debe ser enviado en cada solicitud por parte del cliente, así como de validarlo.

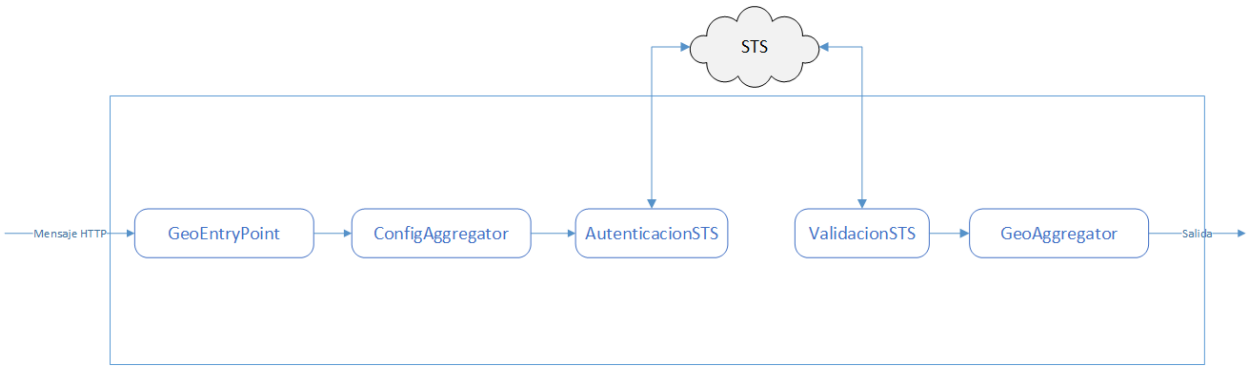


Figura 34 - Mecanismo de autenticación

4.4 Especificación Formal de la Plataforma

A continuación se presenta una especificación formal de la plataforma GeoEEIP. Para dicha especificación se utiliza Event-B[51], que es un método formal para el modelado y análisis de sistemas. Este método tiene varias características destacables, como el uso de la teoría de conjuntos como notación para el modelado, el concepto de *refinamiento*, que permite representar sistemas con diferentes niveles de abstracción, y la posibilidad de realizar pruebas matemáticas para verificar la consistencia entre niveles de refinamiento. En este trabajo se utilizará Event-B solamente con el fin de especificar la plataforma y no para probar propiedades de la misma.

El concepto más básico en Event-B es el modelo. Un modelo se compone de *máquinas abstractas* (abstract machine) y *contextos*. Una máquina describe los aspectos dinámicos (comportamiento) del modelo, en tanto que un contexto describe sus aspectos estáticos. Existen ciertas relaciones entre estos componentes: una máquina

[50] Se llama *query string* al segmento de una URI (o URL) que contiene información no jerárquica de un recurso y que comienza luego del signo "?" como una sucesión de pares "clave=valor" (http://tools.ietf.org/html/rfc3986#section-3.4)

[51] http://www.event-b.org/

puede refinar otra máquina, un contexto puede extender otro contexto y una máquina puede ver un contexto. [52]

Una *machine* se especifica mediante un nombre único y una serie de cláusulas que se describen a continuación.

- REFINES: Indica si la máquina actual refina otra máquina, especificando su nombre.
- SEES: Lista los contextos que son visibles para la máquina actual, permitiendo de esta manera referenciar los conjuntos y constantes definidos en esos contextos.
- VARIABLES: Lista las variables definidas en esta máquina. El estado de una máquina está dado por el estado de todas estas variables.
- INVARIANTS: Lista los predicados que deben ser verdaderos en todos los estados de la máquina.
- EVENTS: Lista los eventos que cambian el estado del modelo y le asignan nuevos valores a las variables. Cada evento se compone de una o más guardas (grd) y una o varias acciones (act). Cada guarda define una condición necesaria para que el evento se dispare. Cuando el AND lógico entre todas las condiciones es verdadero, se ejecutan todas las acciones, las que asignan nuevos valores a algunas de las variables.

Un *context* se especifica mediante un nombre único y una serie de cláusulas que se describen a continuación.

- EXTENDS: Indica si el contexto actual extiende a otros contextos. El contexto actual hereda las constantes y axiomas de los contextos que extiende.
- SETS: Define los tipos de datos como conjuntos.
- CONSTANTS: Lista las constantes que son definidas por este contexto. Cada constante tiene un nombre único.
- AXIOMS: Especifica el tipo de cada constante y los predicados que estas deben cumplir.

A continuación se proporciona la definición de la plataforma GeoEEIP en Event-B en su nivel de abstracción inicial (GeoEEIP_level0_mac), que corresponde al sistema presentado en la Figura 22. El código está acompañado por comentarios en color verde (luego del signo ">") que describen cada línea. Para la especificación se siguieron lineamientos encontrados en [53] y en [54].

En la Figura 35 se observa la primera parte de la definición de la máquina abstracta, en donde se definen los componentes del sistema (GeoEEIP, OWSClient, OWSService, etc.) así como las relaciones entre ellos (ConsumesOWS, ConsumeSOAP, etc.) dentro de la declaración de variables.

```
MACHINE
GeoEEIP_level0_mac > máquina abstracta que especifica la plataforma en el nivel 0.
VARIABLES
GeoEEIP > representa la plataforma
OWSClient > representa los clientes OWS
SOAPClient >  representa los clientes SOAP
OWSService > representa los servicios OWS
SOAPService  >  representa los servicios SOAP (servicios de negocio).
Message > representa los mensajes que se intercambian entre participantes
RequestMessage > representa los pedidos
ResponseMessage > representa las respuestas
Participant    > representa  los participantes que interactúan mediante la plataforma
Client  >  representa los clientes de la plataforma
```

ConsumesOWS > representa la relación entre GeoEEIP y los OWSService (la plataforma consume servicios geoespaciales).

ConsumesSOAP > representa la relación entre GeoEEIP y los SOAPService (la plataforma consume servicios de negocio).

GeoEEIPSendsMessage > representa una relación entre GeoEEIP y Message (la plataforma envía mensajes)

GeoEEIPReceivesMessage > representa una relación entre GeoEEIP y Message (la plataforma recibe mensajes)

OWSSendsResponse > representa la relación entre los OWSService y los ResponseMessage (un servicio geoespacial envía un mensaje de respuesta)

SOAPSendsResponse > representa la relación entre los SOAPService y los ResponseMessage (un servicio de negocio envía un mensaje de respuesta)

To > representa una relación entre un Message y un Participant (los mensajes tienen un destinatario)

From > representa una relación entre un Message y un Participant (los mensajes tienen un emisor)

Uses > representa una relación entre un Client y la GeoEEIP

ClientSendsRequest > representa una relación entre un Client y un RequestMessage (un cliente envía un pedido)

Figura 35 – Especificación en Event-B de componentes del sistema y sus relaciones

En la Figura 36 se muestra la declaración de invariantes de la máquina abstracta. En esta parte se definen los dominios de las variables anteriormente declaradas, así como otros invariantes (ej. los mensajes de pedido y los de respuesta son conjuntos disjuntos).

INVARIANTS

Message_type : Message ∈ ℙ(Message_SET)

RequestMessage_type : RequestMessage ∈ ℙ(Message)

ResponseMessage_type : ResponseMessage ∈ ℙ(Message)

Participant_type : Participant ∈ ℙ(Participant_SET)

Client_type : Client ∈ ℙ (Participant)

To_type : To ∈ Message → Participant

From_type : From ∈ Message ↔ Participant

ClientSendsRequest_type : ClientSendsRequest ∈ Client ↔ RequestMessage

GeoEEIP_type : GeoEEIP ∈ (Participant)

OWSClient_type : OWSClient ∈ ℙ(Client)

SOAPClient_type : SOAPClient ∈ ℙ(Client)

OWSService_type : OWSService ∈ ℙ(Participant)

SOAPService_type : SOAPService ∈ ℙ(Participant)

ConsumesOWS_type : ConsumesOWS ∈ GeoEEIP ↔ OWSService

ConsumesSOAP_type : ConsumesSOAP ∈ GeoEEIP ↔ SOAPService

GeoEEIPSendsMessage_type : GeoEEIPSendsMessage ∈ GeoEEIP ↔ Message

GeoEEIPReceivesMessage_type : GeoEEIPReceivesMessage ∈ GeoEEIP ↔ Message

OWSSendsResponse_type : OWSSendsResponse ∈ OWSService ↔ ResponseMessage

SOAPSendsResponse_type : SOAPSendsResponse ∈SOAPService ↔ ResponseMessage

Uses_type : Uses ∈Client ↔ GeoEEIP

Invariant1 : RequestMessage ∩ ResponseMessage = ∅

Figura 36 - Invariantes de GeoEEIP_level0_mac

En la Figura 37 se muestra la especificación de eventos. Esta especificación comienza con la inicialización de las variables (todos los conjuntos se hacen vacíos) y luego se especifica cada uno de los eventos del sistema. A continuación se describen los eventos:

- **owsSrvAdded**: Modela el cambio de estado cuando se agrega un nuevo servicio OWS a la plataforma
- **soapSrvAdded**: Modela el cambio de estado cuando se agrega un nuevo servicio SOAP a la plataforma
- **requestReceived:** Modela el cambio de estado cuando la plataforma recibe un mensaje de un cliente
- **owsResponseReceived:** Modela el cambio de estado cuando la plataforma recibe una respuesta de un servicio OWS
- **soapResponseReceived**: Modela el cambio de estado cuando la plataforma recibe una respuesta de un servicio SOAP

Los eventos **processRequest**, **processOWSResponse** y **processSOAPResponse** especifican el procesamiento que se realiza en la plataforma para procesar los pedidos de los clientes y las respuestas de los servicios, respectivamente. Estos eventos serán refinados en el siguiente nivel.

EVENTS

INITIALISATION:

BEGIN

> NOTA: En esta sección se inicializan todas las variables con el conjunto vacío

END

owsSrvAdded:

ANY

self

s

WHERE

s_type : s ∈OWSService

self_type : self ∈GeoEEIP

owsSrvAdded_Guard1 : self ↦s ∉ ConsumesOWS

THEN

owsSrvAdded_Action1 : ConsumesOWS ≔ConsumesOWS ∪ {self ↦s}

END

soapSrvAdded:

ANY

self

s

WHERE

s_type : s ∈SOAPService

self_type : self ∈GeoEEIP

soapSrvAdded_Guard1 : self ↦s ∉ConsumesSOAP

THEN

soapSrvAdded_Action1 : ConsumesSOAP ≔ConsumesSOAP ∩ {self ↦s}

END

requestReceived:

ANY

self

m

c

WHERE

m_type : m ∈RequestMessage

c_type : c ∈Client

self_type : self ∈GeoEEIP

requestReceived_Guard1 : To(m) = self

requestReceived_Guard2 : From(m) = c

requestReceived_Guard3 : c ↦m ∈ClientSendsRequest

THEN

requestReceived_Action1 : GeoEEIPReceivesMessage ≔ GeoEEIPReceivesMessage ∪{self ↦m}

requestReceived_Action2 : ClientSendsRequest :∈lientSendsRequest ∖{c ↦m}

requestReceived_Action3 : Uses ≔Uses ∪{c ↦self}

END

owsResponseReceived:

ANY

self

m

s

WHERE

m_type : m ∈ResponseMessage

s_type : s ∈OWSService

self_type : self ∈GeoEEIP

owsResponseReceived_Guard1 : s ↦m ∈OWSSendsResponse

owsResponseReceived_Guard2 : From(m) = s

owsResponseReceived_Guard3 : To(m) = self

THEN

owsResponseReceived_Action1 : GeoEEIPReceivesMessage ≔GeoEEIPReceivesMessage ∪{self ↦m}

END

soapResponseReceived:

ANY

self

m

s

WHERE

m_type : m ∈ResponseMessage

s_type : s ∈SOAPService

self_type : self ∈GeoEEIP

soapResponseReceived_Guard1 : s ↦m ∈ SOAPSendsResponse

soapResponseReceived_Guard2 : From(m) = s

soapResponseReceived_Guard3 : To (m) = self

THEN

soapResponseReceived_Action1 : GeoEEIPReceivesMessage ≔ GeoEEIPReceivesMessage ∪{self ↦m}

END

processRequest:

ANY

self

WHERE

self_type:self ∈GeoEEIP

END

processOWSResponse:

ANY

self

WHERE

self_type:self ∈GeoEEIP

END

processSOAPResponse:

ANY

self

WHERE

self_type:self ∈GeoEEIP

END

END

Figura 37 - Eventos de GeoEEIP_level0_mac

Luego de definida la máquina abstracta en su nivel 0, podemos realizar el primer refinamiento para obtener la máquina en su nivel 1 (Figura 38). El proceso de refinamiento permite extender la lista de variables, suprimir variables, refinar eventos abstractos por versiones concretas y agregar nuevos eventos, entre otras cosas. Como se observa en la Figura 24, en el nivel 1 se agregan los conceptos de ESB y Mecanismos de Integración Geoespaciales. En la Figura 38 se muestra la primera parte de la definición de la máquina en el nivel 1. Aquí se especifica que la GeoEEIP_level1_mac refina la GeoEEIP_level0_mac. También se indica que la máquina tiene visibilidad sobre un nuevo contexto, GeoEEIP_level1_ctx, que se muestra en la Figura 39.

Dentro de la definición de variables, se agrega la que representa a los mecanismos de integración geoespaciales (GeoMech), la que representa los tipos de servicios, como ser WMS, WFS, etc. (ServiceType) y la que representa los protocolos (Protocol), como se explicará más adelante.

MACHINE

GeoEEIP_level1_mac > máquina abstracta que especifica la plataforma en el nivel 1.

REFINES

GeoEEIP_level0_mac

SEES

GeoEEIP_level1_ctx > contexto (se agrega en este nivel)

VARIABLES

GeoMech > representa los mecanismos de integración geoespaciales (se agrega en este nivel)

ServiceType > representa los tipos de servicio (se agrega en este nivel)

Protocol > representa los protocolos (se agrega en este nivel)

SupportsSrvType ›representa una relación entre GeoMech y ServiceType (un mecanismo soporta un conjunto de tipos de servicio (WMS, WFS, etc.) y recíprocamente

SupportsProtocol ›representa una relación entre GeoMech y Protocol (un mecanismo soporta un protocolo (OWS o SOAP) y recíprocamente.

GeoMechProcessesReq > representa una relación entre GeoMech y RequestMessage (un mecanismo está procesando un pedido)

GeoMechProcessesRsp representa una relación entre GeoMech y RequestMessage (un mecanismo está procesando una respuesta)

Features › representa una relación entre GeoEEIP y GeoMech (la plataforma posee un conjunto de mecanismos geoespaciales)

> NOTA: El resto de la definición de variables es igual que en GeoEEIP_level0_mac

Figura 38 – Primera parte de la especificación de GeoEEIP_level1_mac

El contexto GeoEEIP_level1_ctx permite definir el conjunto de los protocolos soportados para los Web services en la plataforma (SupportedProtocols), que podrán ser OWS (basados en HTTP) o SOAP.

CONTEXT

GeoEEIP_level1_ctx

SETS

SupportedProtocols

CONSTANTS

SOAP

OWS

AXIOMS

SOAP_type: SOAP ∈SupportedProtocols

OWS_type: OWS ∈SupportedProtocols

enumerationOf_SupportedProtocols: partition(SupportedProtocols,{SOAP},{OWS})

```
END
```

Figura 39 – Definción del contexto GeoEEIP_level1_ctx

En la Figura 40 se muestra la declaración de invariantes de la máquina abstracta en el nivel 1. Se omiten las declaraciones idénticas a la del nivel anterior.

```
INVARIANTS
        GeoMech_type:     GeoMech ∈ ℙ(GeoMech_SET)
        ServiceType_type: ServiceType ∈ ℙ(ServiceType_SET)
        Protocol_type:    Protocol ∈ ℙ(Protocol_SET)
        SupportsSrvType_type:     SupportsSrvType ∈ GeoMech <<->> ServiceType
        SupportsProtocol_type:    SupportsProtocol ∈ GeoMech <<->> Protocol
        GeoMechProcessesReq_type:       GeoMechProcessesReq ∈ GeoMech ↔ RequestMessage
        GeoMechProcessesRsp_type:       GeoMechProcessesRsp ∈ GeoMech ↔ ResponseMessage
        Features_type:    Features ∈ GeoEEIP ↔ GeoMech
        HasType_type:     HasType ∈ Message → ServiceType
        GeoMechProcessesRsp.Injective:    GeoMechProcessesRsp~ ∈ ResponseMessage ⇸ GeoMech
        GeoMechProcessesReq.Injective:    GeoMechProcessesReq~ ∈ RequestMessage ⇸ GeoMech
> NOTA: El resto de la definición de invariantes es igual que en GeoEEIP_level0_mac
```

Figura 40 – Invariantes de GeoEEIP_level1_mac

En la Figura 41 se muestra la especificación de eventos (se omiten los eventos que son idénticos a los del nivel anterior). En la especificación de este nivel, se refinan los eventos processRequest, processOWSResponse y processSOAPResponse.

```
EVENTS
> NOTA: El resto de la definición de eventos es igual que en GeoEEIP_level0_mac
BEGIN
processRequest:
REFINES
        processRequest
ANY
self
m
gm
WHERE
        self_type: self ∈ GeoEEIP
        m_type:   m ∈ RequestMessage
        gm_type: gm ∈ GeoMech
```

self_type:self ∈GeoEEIP

processRequest_Guard1: (self ↦m)∈GeoEEIPReceivesMessage

processRequest_Guard2: HasType(m)∈SupportsSrvType(gm)

processRequest_Guard3: From(m)∈Client

THEN

processRequest _Action1 : GeoMechProcessesRequest :=GeoMechProcessesRequest ∪{gm ↦m}

processRequest _Action2 : GeoEEIPReceivesMessage :=GeoEEIPReceivesMessage \{ self ↦m}

END

processOWSResponse:

REFINES

processOWSResponse

ANY

self

m

gm

WHERE

self_type:self ∈GeoEEIP

m_type: m ∈ResponseMessage

gm_type: gm ∈GeoMech

processResponse_Guard1: (self↦m) ∈GeoEEIPReceivesMessage

processResponse_Guard2: HasType(m)∈SupportsSrvType(gm)

processResponse_Guard3: From(m)∈OWSService

THEN

processOWSResponse_Action1 : GeoMechProcessesOWSResponse :=GeoMechProcessesOWSResponse ∪{gm ↦m}

processOWSResponse_Action2 : GeoEEIPReceivesMessage :=GeoEEIPReceivesMessage \{ self ↦m}

END

processSOAPResponse:

REFINES

processSOAPResponse

ANY

self

m

gm

WHERE

self_type:self ∈GeoEEIP

m_type: m ∈ResponseMessage

gm_type: gm ∈GeoMech

processResponse_Guard1: (self↦m) ∈GeoEEIPReceivesMessage

processResponse_Guard2: HasType(m)∈SupportsSrvType(gm)

```
        processResponse_Guard3:    From(m)∈SOAPService
THEN
processSOAPResponse _Action1  :  GeoMechProcessesSOAPResponse :=GeoMechProcessesSOAPResponse ∪{gm ↦m}
processSOAPResponse _Action2  :  GeoEEIPReceivesMessage := GeoEEIPReceivesMessage \{ self ↦m}
END
END
```

Figura 41 – Eventos de GeoEEIP_level1_mac

5 Implementación y Experimentación

En el Capítulo 4 se realizó una propuesta de solución de la problemática abordada en este trabajo, en base a la definición de una plataforma de integración dependiente del dominio basada en ESB (GeoEEIP), que permite integrar servicios geográficos dentro de los escenarios planteados anteriormente. En este capítulo, se describe una implementación concreta de dicha plataforma, que permite validar la propuesta utilizando herramientas específicas y avanzar en la evaluación de la factibilidad operativa de la misma.

La primera consideración relativa a la implementación de la GeoEEIP fue la elección de un producto de ESB. En este sentido, se aprovechó la experiencia del Laboratorio de Integración de Sistemas[52] (LINS) del Instituto de Computación, que como grupo de investigación en el área posee una vasta experiencia en la utilización de productos de ESB. El producto elegido fue JBoss ESB en su versión 4.12.

5.1 Implementación de los mecanismos.

Los mecanismos abstractos que fueron definidos en la Sección 4.3 deben ser implementados y/o configurados utilizando los mecanismos concretos que provee un producto ESB específico. En el caso de JBoss ESB, todo puede ser especificado en términos de "servicios" y "mensajes," en sintonía con los principios SOA.

Un servicio en JBoss ESB se define como una lista de acciones (o "pipeline" de acciones) que procesan un mensaje ESB en forma secuencial. Un servicio puede definir un conjunto de "listeners" que rutean mensajes hacia su *pipeline* de acciones.

Los patrones de mediación típicos de un ESB (Sección 2.1.6) son encapsulados en JBoss ESB como acciones, las cuales son referenciadas dentro de los servicios que las utilizan. Para definir los servicios se utiliza un único archivo de configuración escrito en XML (jboss-esb.xml).

Una decisión de implementación que se presenta comúnmente cuando se utiliza JBoss ESB es si un patrón de mediación debe ser implementado como un servicio o como una acción. La ventaja de implementar acciones es que estas ofrecen un mayor grado de reutilización al poder ser utilizadas en varios servicios. Por otro lado, existen algunas restricciones que hacen que sea necesario implementar ciertos mecanismos como servicios. En particular, todo mecanismo que sea el destino de un Router debe ser implementado como servicio, ya que no se permiten acciones como destino de un ruteo.

La forma de implementar un mecanismo como una acción es utilizando la herencia del lenguaje Java a partir de la clase base abstracta AbstractActionPipelineProcessor.

5.1.1 Arquitectura de Implementación.

En base a la arquitectura lógica planteada en la Sección 4.2 (Figura 25), se llegó a una implementación sobre JBossESB que corresponde a la arquitectura de la Figura 42. Como puede observarse, cada una de los capas está compuesta por los mecanismos implementados, de acuerdo a su nivel de generalidad. En la Capa 3 se ubican los mecanismos geoespaciales compuestos, en la Capa 2 los mecanismos geoespaciales básicos, y en la capa 1 los mecanismos incorporados de JBossESB.

A partir de la Figura 42 podemos notar lo siguiente:

- Todos los mecanismos geoespaciales compuestos fueron implementados como servicios.

[52] http://www.fing.edu.uy/inco/grupos/lins/

- Fue posible mantener la división lógica entre mecanismos compuestos y básicos a nivel de implementación.
- Algunos mecanismos básicos fueron implementados como servicios, otros como acciones. En algunos casos, los mecanismos, dada su complejidad, fueron implementados como una combinación de servicios y acciones.
- Se utilizaron varios patrones de mediación provistos por JBossESB (los mecanismos incorporados), lo que permitió simplificar la implementación de los mecanismos geoespaciales.

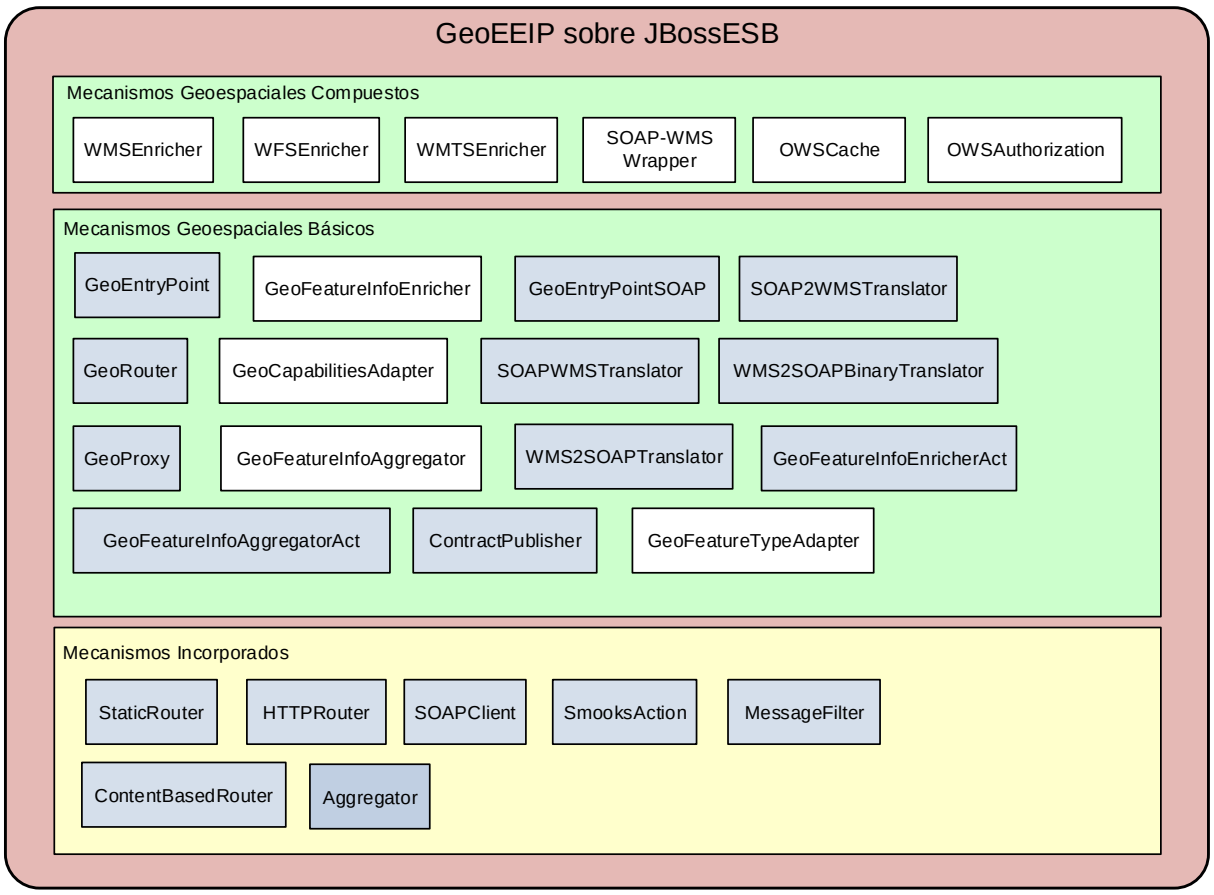


Figura 42 – Arquitectura de implementación en base a servicios (blanco) y acciones (gris) de JBossESB.

Es importante notar que en el caso de sustituir el producto de ESB, la capa de mecanismos que cambiaría completamente sería la de los mecanismos incorporados (para utilizar los mecanismos equivalentes del otro producto) y no así las capas superiores (las de los mecanismos geoespaciales). En este último caso, la implementación sobre otro producto de ESB implicaría adaptar los mecanismos geoespaciales de acuerdo a la arquitectura interna de ese ESB, pero manteniendo la misma lógica con que fueron implementados sobre JBossESB.

En las siguientes secciones, se profundizan los detalles de implementación de los mecanismos.

5.1.2 Mecanismos incorporados de JBossESB

Estos son los mecanismos que cada producto ESB debe proporcionar y que implementan los patrones de mediación de uso general. En el caso de JBossESB, como se mencionó anteriormente, estos mecanismos se implementan mediante *acciones*, que son clases Java que extienden la clase AbstractActionPipelineProcessor. En

esta sección se describen brevemente, en base a [55] y [56], las acciones que son utilizadas por los mecanismos geoespaciales. Estas acciones se muestran en la Figura 42 formando la capa inferior (en color amarillo).

5.1.2.1 StaticRouter

Esta acción se encarga de direccionar un mensaje a un conjunto de servicios, independientemente del contenido del mensaje u otra variable.

5.1.2.2 HTTPRouter

Esta acción se encarga de invocar un endpoint HTTP cualquiera desde el pipeline, especificando la URL, el método (GET o POST), el tipo de respuesta (texto o binario), etc.

5.1.2.3 SOAPClient

Esta acción permite consumir Web services SOAP externos al ESB.

5.1.2.4 SmooksAction

Esta acción permite utilizar todas las funcionalidades provistas por el framework Smooks[53]. Este framework realiza diferentes tipos de procesamiento sobre mensajes en formato XML y no XML (CSV, código Java, EDI, etc.) y tiene la particularidad de poder trabajar sobre fragmentos específicos de mensajes, que podrían ser de gran tamaño. Dentro de los tipos de procesamiento que puede realizar, su uso más común es como motor de transformaciones utilizando *templates* (XSLT[54] o FreeMarker[55]). Además de esto, puede realizar división (splitting) de mensajes, ruteo, enriquecimiento, validación basada en reglas complejas, persistencia utilizando frameworks de ORM (ej. Hibernate), asociación (binding) con objetos Java, etc.

5.1.2.5 MessageFilter

Esta acción es un tipo específico de ruteo basado en contenido (extiende la acción ContentBasedRouter) y se encarga de descartar los mensajes que no cumplen con cierta condición.

5.1.2.6 ContentBasedRouter

Esta acción se encarga de determinar los servicios interesados en recibir una copia del mensaje de acuerdo a la aplicación de un conjunto de reglas, que pueden estar especificadas en XPath o en Drools[56] Rule Language (DRL). Es la implementación por defecto en JBossESB del rutero basado en contenido explicado en la Sección 2.1.6.1 .

5.1.2.7 Aggregator

Esta acción permite unificar un conjunto de mensajes creando un mensaje unificado.

---

53 http://www.smooks.org

54 http://www.w3.org/TR/xslt20/

55 http://freemarker.incubator.apache.org/

56 http://www.drools.org/

5.1.3 Mecanismos Geoespaciales Básicos

A continuación se presentan los detalles de implementación de los mecanismos geoespaciales básicos dentro del prototipo desarrollado en JBoss ESB.

5.1.3.1 GeoEntryPoint

Este componente implementa el mecanismo especificado en 4.3.2.1 como una acción. GeoEntryPoint brinda un punto de entrada a la plataforma GeoEEIP para el procesamiento inicial de una petición HTTP.

Tabla 13 – Especifiación de GeoEntryPoint

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

5.1.3.2 GeoEntryPointSOAP

Este componente implementa una variación del mecanismo especificado en 4.3.2.1 como una acción, con parámetros específicos que lo diferencian de GeoEntryPoint. GeoEntryPointSOAP brinda un punto de entrada a la plataforma GeoEEIP para el procesamiento inicial de un mensaje SOAP. Se implementó como una acción.

Tabla 14 – Especifiación de GeoEntryPointSOAP

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

5.1.3.3 GeoProxy

Este componente implementa el mecanismo especificado en 4.3.2.2 como una acción. GeoProxy tiene como responsabilidad construir el pedido que se enviará al servidor OWS y realizar la invocación. El pedido se construye con los parámetros que se reciben en la propiedad QUERY_STRING del mensaje ESB y la URL del servidor OWS que obtiene de la propiedad IMS_NAME de la configuración.

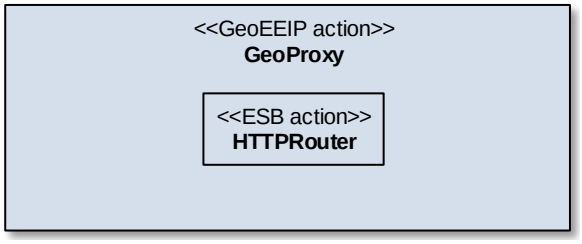


Figura 43 – Componentes de GeoProxy

En el siguiente paso 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.

GeoProxy es una acción que especializa AbstractActionPipelineProcessor y que utiliza HTTPRouter.

Tabla 15 – Especifiación de 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	
QUERY_STRING	Fragmento de la petición IMS REST original con los parámetros de consulta.
QUERY_PARAMS	Diccionario de parámetros generado a partir del parámetro QUERY_STRING.
Configuración del componente	
IMS_NAME	URL base del servicio IMS a consultar.
Formato de mensaje de salida	
IMS_ADDRESS	URL base del servicio IMS consultado.
IMS_RESPONSE	Respuesta de la consulta IMS en el formato solicitado.

5.1.3.4 GeoRouter

El mecanismo especificado en 4.3.2.3 puede ser implementado en JBossESB mediante la acción ContentBasedRouter, por lo que no se implementó una acción con el nombre GeoRouter, sino que simplemente se utilizó ContentBasedRouter desde los otros mecanismos. El Servicio ESB a direccionar se obtiene de aplicar expresiones regulares a la propiedad QUERY_STRING del mensaje ESB. La acción ContentBasedRouter tiene como restricción que el destino del mensaje debe ser un servicio ESB y no otra acción.

5.1.3.5 GeoFeatureInfoEnricher

Este componente implementa el mecanismo especificado en 4.3.2.4. Su función es la de enriquecer las respuestas de la operación GetFeatureInfo (WMS y WMTS).

Dado que se trata de un mecanismo de mayor complejidad que otros mecanismos básicos, para la implementación fue necesario generar un conjunto de servicios y acciones, como se muestra en la Figura 44.

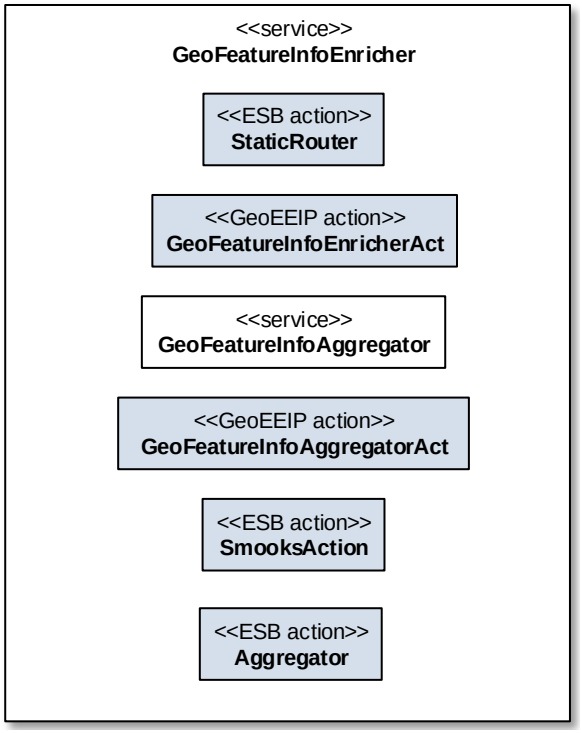


Figura 44 - Componentes de GeoFeatureInfoEnricher

En la Figura 45 puede observarse el flujo de mediación que ocurre dentro del servicio GeoFeatureInfoEnricher que actúa como punto de entrada. Dado que el enriquecimiento de una capa geográfica puede hacerse consultando a más de una fuente de negocio, se crea un servicio dentro del ESB por cada una de estas fuentes, y se emplea una acción StaticRouter para rutear los mensajes hacia cada uno de esos servicios.

Finalmente, en el servicio GeoFeatureInfoAggregator se realiza lo siguiente. En primer lugar se obtienen los datos empresariales a través de la acción de Aggregator, que luego de haber recibido todas las respuestas esperadas da paso a la 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. Esto se realiza por cada fuente empresarial consultada.

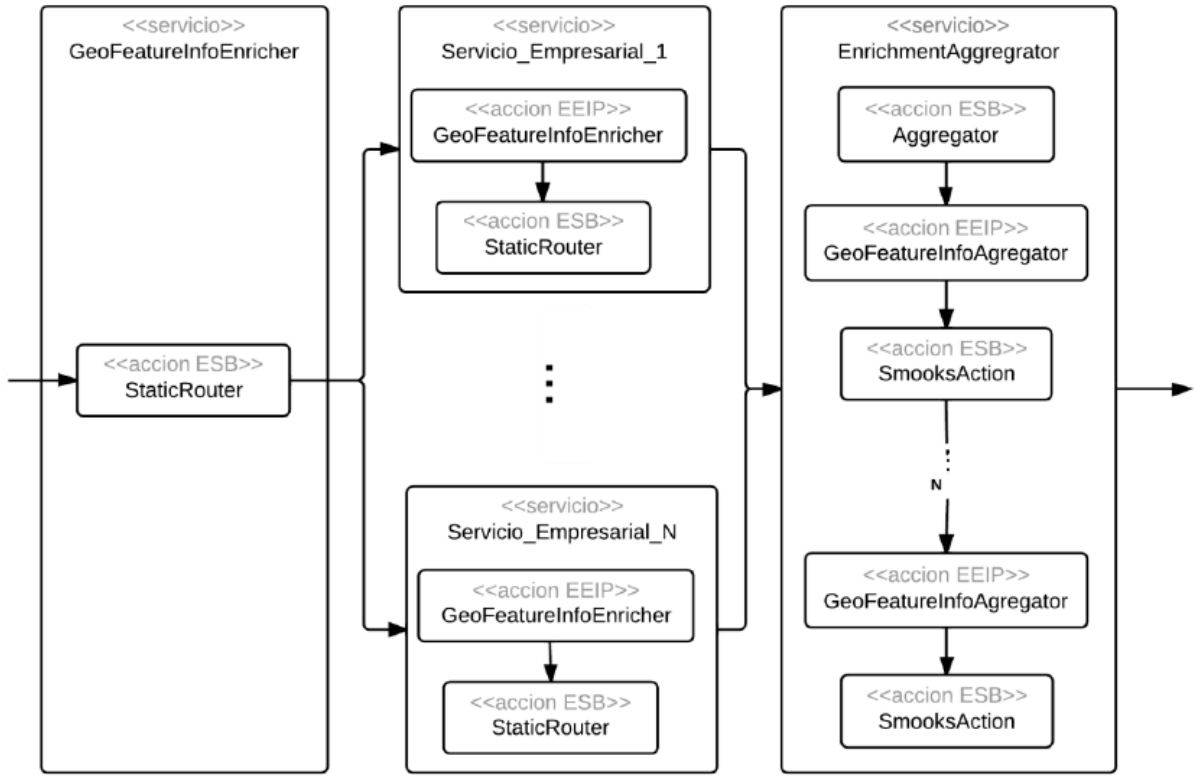


Figura 45 – Flujo de mediación de GeoFeatureInfoEnricher.

Tabla 16 – Especifiación de GeoFeatureInfoEnricher

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

5.1.3.6 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. Se implementó como una acción que especializa a AbstractActionPipelineProcessor de JBoss ESB.

Tabla 17 – Especifiación de SOAP2WMSTranslator

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

5.1.4 Mecanismos Geoespaciales Compuestos

A continuación se detallan los mecanismos geoespaciales compuestos que han sido implementados en la GeoEEIP, en base a los mecanismos básicos y los mecanismos nativos vistos en la sección anterior.

5.1.4.1 WMSEnricher

Este componente implementa el mecanismo especificado en 4.3.1.1, mediante la definición del servicio WMSEnricher. Este servicio utiliza las acciones GeoEntryPoint, GeoRouter y GeoProxy, y los servicios GeoFeatureInfoEnricher y GeoCapabilitiesAdapter, siguiendo la lógica de mediación definida anteriormente (Figura 26). Los componentes de los que depende directamente este servicio se resumen en la Figura 46.

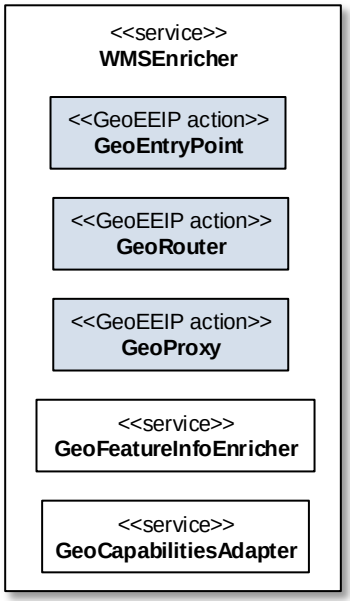


Figura 46 – Componentes de los que depende directamente WMSEnricher

En la Tabla 18 se muestran los parámetros que deben ser configurados para el servicio WMSEnricher. Los parámetros marcados con (*) pueden tener una multiplicidad mayor a uno en el caso que la respuesta sea enriquecida consultando a más de una fuente empresarial (servicio SOAP).

Tabla 18 – Parámetros de WMSEnricher

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

5.1.4.2 WMTSEnricher

Este componente implementa el mecanismo especificado en 4.3.1.2 mediante la definición del servicio WMTSEnricher. Este servicio utiliza las acciones GeoEntryPoint, GeoRouter y GeoProxy, y los servicios GeoFeatureInfoEnricher y GeoCapabilitiesAdapter, siguiendo la lógica de mediación definida anteriormente (Figura 27). Los componentes de los que depende directamente este servicio se resumen en la Figura 47. Puede notarse que se reutilizan los mismos componentes básicos que usa WMSEnricher, cambiando simplemente la lógica de ruteo en GeoRouter y la transformación XSLT en GeoCapabilitiesAdapter.

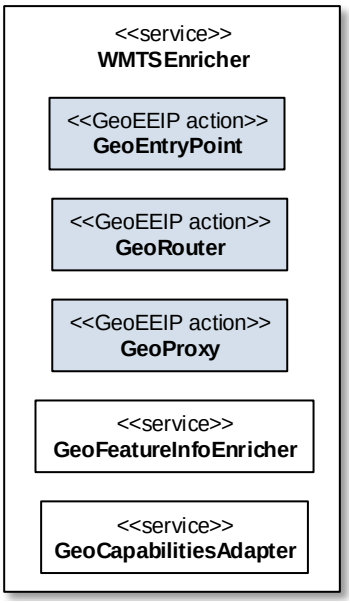


Figura 47 – Componentes de los que depende WMTSEnricher

En la Tabla 19 se muestran los parámetros que deben ser configurados para el servicio WMTSEnricher. Los parámetros marcados con (*) pueden tener una multiplicidad mayor a uno en el caso que la respuesta sea enriquecida consultando a más de una fuente empresarial (servicio SOAP).

Tabla 19 – Parámetros de WMTSEnricher

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

5.1.4.3 WFS Enricher

Este componente implementa el mecanismo especificado en 4.3.1.3, mediante la definición del servicio WFSEnricher. Este servicio utiliza las acciones GeoEntryPoint, GeoRouter y GeoProxy, y los servicios GeoFeatureInfoEnricher y GeoCapabilitiesAdapter, siguiendo la lógica de mediación definida anteriormente (Figura 28). Los componentes de los que depende directamente este servicio se resumen en la Figura 48.

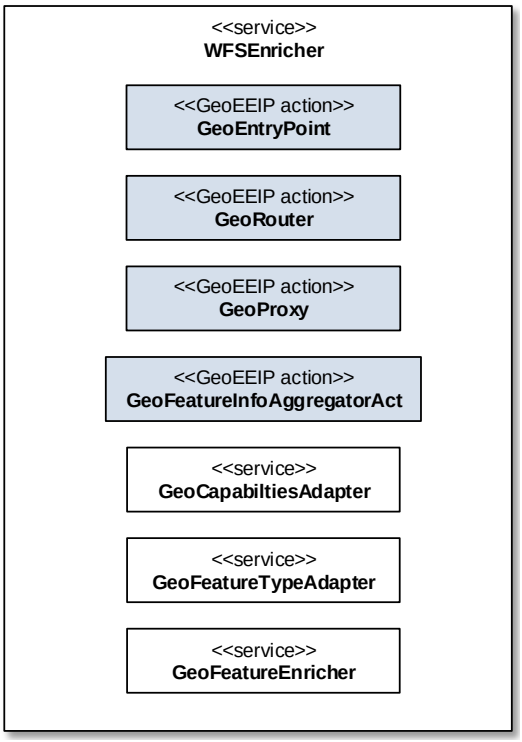


Figura 48 – Componentes de los que depende directamente WFSEnricher

En la Tabla 20 se muestran los parámetros que deben ser configurados para el servicio WMSEnricher. Los parámetros marcados con (*) pueden tener una multiplicidad mayor a uno en el caso que la respuesta sea enriquecida consultando a más de una fuente empresarial (servicio SOAP).

Tabla 20 – Parámetros de WFS Enricher

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

5.1.4.4 SOAP-WMS Wrapper

Este componente implementa el mecanismo especificado en 4.3.1.4. Su función es la empaquetar y desempaquetar los pedidos y las respuestas de las operaciones WMS en mensajes SOAP. Los componentes de los que depende directamente este servicio se resumen en la Figura 49.

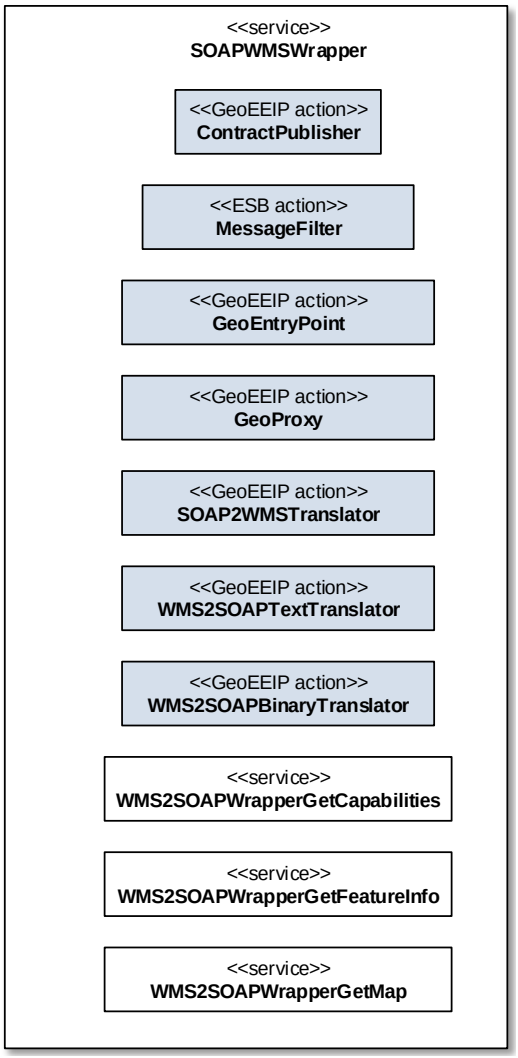


Figura 49 – Componentes de los que depende SOAP-WMSWrapper

Este componente brinda la posibilidad de acceso a un servidor WMS mediante el protocolo SOAP, utilizando la definición WSDL planteada en el estudio [57]. El componente se divide en dos pasos como muestra la Figura 50. 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."

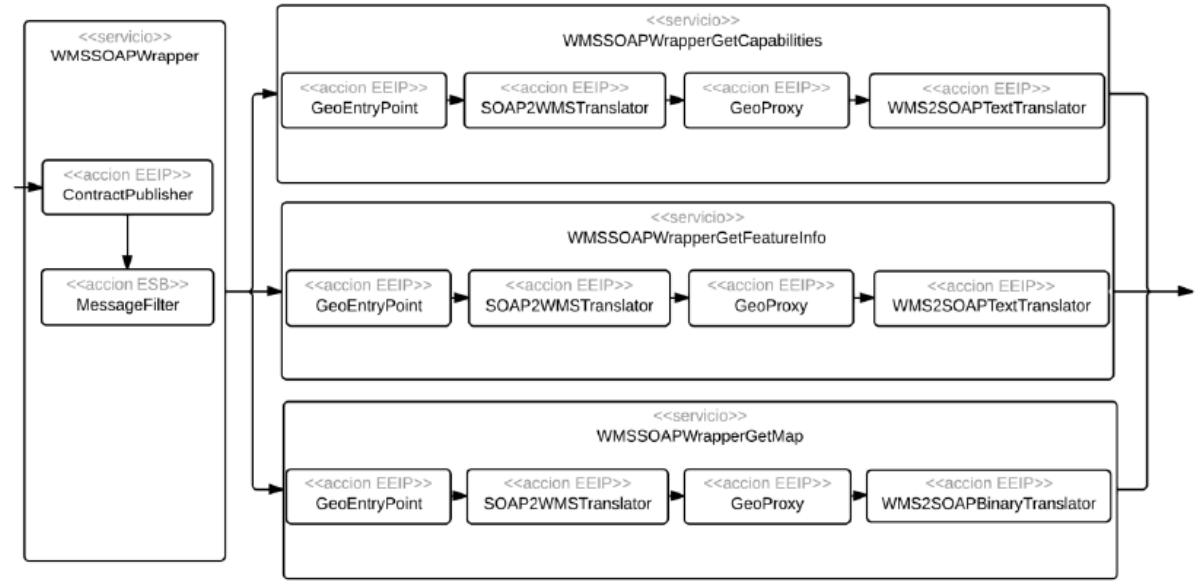


Figura 50 – Flujo de meadiación de SOAP-WMS Wrapper

En la Tabla 21 se muestran los parámetros que deben ser configurados para el servicio SOAP-WMS Wrapper.

Tabla 21 – Parámetros de SOAP-WMS Wrapper

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

5.2 Pruebas del Prototipo

En esta sección de describen las pruebas realizadas sobre el prototipo implementado. Estas pruebas permitieron evaluar la factibilidad técnica por un lado, y el desempeño de una plataforma de este tipo en base a invocaciones que se aproximan a las que pueden ocurrir en un ambiente real.

5.2.1 Ambiente de pruebas

Para la realización de las pruebas se trabajó en un ambiente experimental. El propósito de la ejecución de las pruebas en este ambiente no es el de medir valores absolutos de performance que se asemejen a los valores reales en una PGE, sino simplemente poder evaluar en forma relativa el impacto de introducir esta plataforma entre los consumidores y los proveedores de servicios.

El ambiente de pruebas se conforma de tres PCs: uno actuando como consumidor de servicios, otro actuando como proveedor de servicios, y otro actuando como plataforma de integración. Las características más relevantes del hardware de estos tres equipos se detallan en la Tabla 22. Cabe destacar que tanto la Plataforma como los Proveedores deberían ejecutar en servidores especializados en un ambiente real, y no en computadores personales. En la Figura 51 se muestra la topología de la red de este ambiente de pruebas.

Tabla 22 - Características del hardware de pruebas

Características	PC Consumidor	PC Plataforma	PC Proveedor
Software de virtualización	No	VirtualBox 5.0.x sobre Windows 8.1 64 bits	No
Sistema operativo	Windows 8.1 64 bits	Windows 7 Ultimate 32 bits (guest)	Windows 8.1 64 bits
Java Virtual Machine	JDK 8 64 bits	JDK 6 64 bits	JDK 8 64 bits
CPU	Intel Core i7 4510U Dual Core @2.6 Ghz	AMD A10-7800 Quad Core @3.5 Ghz	Intel Core i5 480M Dual Core @ 2.6 GHz
RAM	8 GB	16 GB	6 GB
Almacenamiento	SSD 540/520 MB/s	HDD 7200 RPM	HDD 7200 RPM
Interfaz de Red	802.3 @ 1000 Mbps	802.3 @ 1000 Mbps	802.11n @ 144 Mbps

En el PC Consumidor se ejecuta el software necesario para que actúe como Cliente OWS y SOAP. Este software consiste en un browser Firefox 40.x que ejecuta el cliente del Caso de Estudio utilizando código Javascript y la API OpenLayers, así como las herramientas JMeter[57] 2.13 y SOAP UI[58] 4.6.

En el PC Proveedor se ejecuta el software necesario para que actúe como Servidor de Mapas y como Sistema Empresarial. Este software consiste en el servidor GeoServer 2.4.x para publicar los OWS y en el servidor de aplicaciones JBossAS 6.1 para publicar los WS SOAP.

En el PC Plataforma se ejecuta el software de la plataforma GeoEEIP sobre JBoss ESB 4.12 y JBoss AS 6.1.

Los tres equipos se conectan a través de un router. El PC Consumidor y el PC Plataforma tienen conexiones cableadas Ethernet, en tanto que el PC Proveedor posee una conexión WiFi. Esto permite simular situaciones en donde la conexión entre el consumidor y la plataforma es mucho más veloz que entre el consumidor y el proveedor (requerimiento 3.2.2.6).

[57] http://jmeter.apache.org/

[58] http://www.soapui.org/

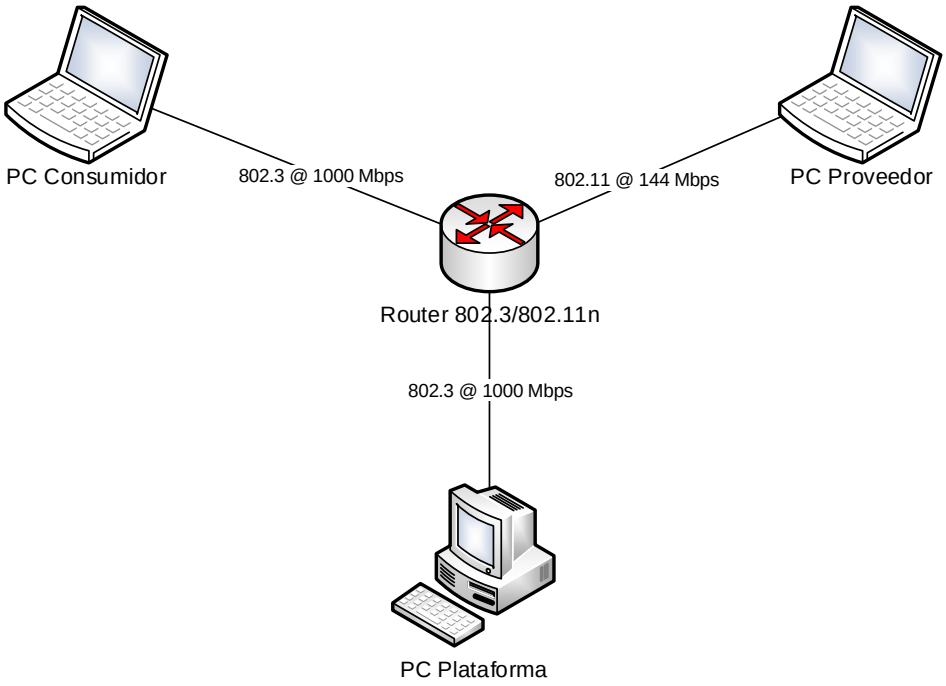


Figura 51 – Topología de la LAN de pruebas

5.2.2 Pruebas de Performance

En esta sección de detallan los planes de pruebas de performance ejecutados para cada mecanismo de la GeoEEIP. Se eligió como métrica de performance el tiempo de respuesta, que se define como el tiempo que transcurre desde que se envía un pedido hasta que se recibe la respuesta. El tiempo de respuesta es una suma de tres latencias:

Tiempo de Respuesta = Latencia de Cliente + Latencia de Red + Latencia de Servidor

La Latencia de Cliente es la suma de los tiempos que transcurren mientras el cliente procesa y envía el pedido más los tiempos que transcurren mientras procesa y recibe la respuesta. La Latencia de Red es la suma de los tiempos que lleva transmitir el pedido desde el cliente hasta el servidor y la respuesta desde el servidor hasta el cliente a través de la red que los conecta. La Latencia de Servidor es la suma de los tiempos que transcurren mientras el servidor recibe y procesa el pedido más los tiempos que transcurren mientras procesa y envía la respuesta.

En el caso más general, las pruebas consistieron en invocaciones HTTP GET utilizando la herramienta JMeter. Esta herramienta permite simular un número configurable de clientes concurrentes, cada uno ejecutando en su propio hilo o *thread*, que realizan las invocaciones al servidor. Se optó por configurar todas estas pruebas con un número de 100 clientes concurrentes. La herramienta se encarga de registrar el tiempo de respuesta de cada invocación y luego calcular diferentes indicadores, como el tiempo mínimo (tiempo de respuesta de la invocación que tardó menos), tiempo máximo (tiempo de respuesta de la invocación que tardó más), media de los tiempos, promedio de los tiempo, etc.

5.2.2.1 WMS Enricher

En la Tabla 23 se detallan las diferentes pruebas que se llevaron a cabo para medir y comparar los tiempos de respuesta de diferentes invocaciones a todas las operaciones de WMS.

Para probar la operación GetMap se utilizaron los siguientes parámetros:

SERVICE=wms&VERSION=1.3.0&REQUEST=GetMap&LAYERS=eeip%3Amanzanas,eeip%3Aesp_libres,eeip%3Aejes,eeip%3Aserv_comerciales&TRANSPARENT=TRUE&FORMAT=image%2Fpng&
&STYLES=&SRS=EPSG%3A32721&BBOX=578857.96977642,6137067.1159952,580055.41291205,6138264.5591308&WIDTH=256&HEIGHT=256

Para probar la operación GetCapabilities se utilizaron los siguientes parámetros:

SERVICE=wms&VERSION=1.3.0&REQUEST=GetCapabilities

Para probar la operación GetFeatureInfo se utilizaron los siguientes parámetros:

SERVICE=wms&VERSION=1.3.0&REQUEST=GetFeatureInfo&LAYERS=eeip%3Aserv_comerciales&QUERY_LAYERS=eeip%3Aserv_comerciales&STYLES
&BBOX=574443.440207%2C6135228.71272%2C578554.973473%2C6137623.598991&FEATURE_COUNT=10&HEIGHT=512&WIDTH=879&FORMAT=image%2Fpng&INFO_FORMAT=application%2Fvnd.ogc.gml&SRS=EPSG%3A32721&X=695&Y=53&BUFFER=25

Tabla 23 - Plan de pruebas para WMS Enricher

Referencia	Operación	Descripción
P-01-01	GetMap	Realiza un GetMap desde el consumidor hacia el proveedor, sin utilizar la plataforma.
P-01-02	GetMap	Realiza un GetMap desde el consumidor hacia el proveedor, pasando por la plataforma (la cual no introduce modificaciones en la respuesta).
P-01-03	GetCapabilities	Realiza un GetCapabilities desde el consumidor hacia el proveedor, sin utilizar la plataforma.
P-01-04	GetCapabilities	Realiza un GetCapabilities desde el consumidor hacia el proveedor, pasando por la plataforma, la cual introduce modificaciones en la respuesta.
P-01-05	GetFeatureInfo	Realiza un GetFeatureInfo desde el consumidor hacia el proveedor, sin utilizar la plataforma. No se realiza enriquecimiento de la respuesta.
P-01-06	GetFeatureInfo	Realiza un GetFeatureInfo desde el consumidor hacia el proveedor, pasando por la plataforma, la cual enriquece la respuesta mediante un servicio de negocio.
P-01-07	GetFeatureInfo	Realiza un GetFeatureInfo desde el consumidor hacia el proveedor. Realiza una invocación a un servicio de negocio. Realiza enriquecimiento mediante una transformación XSLT en el cliente. No se utiliza la plataforma.

En el caso de las prueba P-01-07 de la operación GetFeatureInfo, se implementó un enriquecimiento *ad hoc* de la respuesta en el cliente. Esto permite simular una solución en donde el cliente es responsable de realizar una invocación a un OWS (WMS en este caso) y otra a un WS SOAP, para luego integrar los datos de negocio con la información geográfica en el propio cliente.

En la Tabla 24 se registran los principales resultados obtenidos en estas pruebas. En este caso pueden sacarse las siguientes conclusiones:

- En el par de pruebas (P-01-01, P-01-02), en donde se invoca un GetMap sin realizar ninguna transformación en la respuesta, no hay diferencias significativas en los tiempos entre pasar por la plataforma o no. Es decir, la plataforma no presenta *overhead* cuando actúa como un pasamano.
- En los pares de pruebas (P-01-03, P-01-04) y (P-01-05, P-01-06), hay un *overhead* significativo cuando se utiliza la plataforma. Sin embargo, en los casos que no se utiliza la misma, las respuesta WMS y la

respuesta SOAP no se integran, es decir, no se realiza una transformación para crear un documento XML o HTML integrado, sino que simplemente se obtienen los resultados y quedan disponibles en el cliente para que este los despliegue al usuario, por lo que hay un procesamiento desigual. En la prueba P-01-07, en donde efectivamente se realiza una transformación en el cliente mediante código Javascript, los tiempos superan ampliamente a los registrados en la plataforma. En este caso se interpreta que la ventaja radica en la superioridad del hardware del servidor con respecto a la del cliente, lo que es altamente factible en un escenario real con servidores especializados y clientes heterogéneos.

Tabla 24 - Resultados del plan de pruebas para WMS Enricher

Referencia	# Muestras	Min	Max	Media
P-01-01	100	752	7948	5170
P-01-02	100	319	11609	6108
P-01-03	100	18	857	217
P-01-04	100	163	5648	3728
P-01-05	100	9	221	13
P-01-06	100	137	4693	2727
P-01-07	10	6693	8230	7589

Además de estas pruebas, se realizó una prueba de cumplimiento con la norma de la directiva INSPIRE que regula las IDEs europeas (Sección 2.1.3). Para el caso de WMS, se aplica la norma para servicios de visualización (que hemos llamado "servicios de mapas" en este documento) [58].

En dicha norma se establece, como único requerimiento para un servicio de visualización en cuanto a tiempo de respuesta, que "para una imagen de 470 kilobytes (por ejemplo, 800x600 pixels con una profundidad de color de 8 bits), el tiempo de respuesta para enviar la respuesta inicial de una petición GetMap de un servicio de visualización deberá ser como máximo de 5 segundos en una situación normal." Se aclara luego que como "situación normal" se refiere a períodos fuera de la carga pico, que deberían darse en un 90% de los casos.

En cuanto al procedimiento de testing, se especifican los siguientes criterios prácticos:

- Se deben enviar al menos 10 pedidos por hora
- El pedido debe ser de una sola capa geográfica por vez
- La imagen devuelta será de 800x600 pixels con una profundidad de color de 8 bits
- Se considerará el tiempo en devolver la imagen completa y no solo la "respuesta inicial" (entendido como el primer byte de la imagen)
- Se tomarán el 90% de los casos con mejor tiempo de respuesta.

Para realizar esta prueba, se reformuló el pedido GetMap para solicitar un única capa con un tamaño de imagen de 800x600 pixels.

SERVICE=wms&VERSION=1.3.0&REQUEST=GetMap&LAYERS=eeip%3Amanzanas
&TRANSPARENT=TRUE&FORMAT=image%2Fpng&
&STYLES=&SRS=EPSG%3A32721&BBOX=578857.96977642,6137067.1159952,580055.41291205,6138264.5591308&WIDTH=800&HEIGHT=600

Tabla 25 - Plan de pruebas de tiempo de respuesta para INSPIRE

Referencia	Operación	Descripción
P-01-08	GetMap	Realiza un GetMap desde el consumidor hacia el proveedor, sin utilizar la plataforma, siguiendo el procedimiento de prueba indicado por INSPIRE.
P-01-09	GetMap	Realiza un GetMap consumidor hacia el proveedor, pasando por la plataforma (la cual no introduce modificaciones en la respuesta), siguiendo el procedimiento de prueba indicado por INSPIRE.

Como puede verse en la Tabla 26, luego de descartadas las 10 muestras con los tiempos más altos y calculado el valor máximo, se llegó a que el tiempo de respuesta máximo es inferior a 2 segundos invocando directamente al servidor de mapas, e inferior a 3 segundos invocando a la plataforma. De esta manera vemos que la plataforma no produce una degradación de la performance que impida cumplir con la normativa INSPIRE.

Tabla 26 – Resultados del plan de pruebas de tiempo de respuesta para INSPIRE

Referencia	# Muestras Totales	# Muestras Consideradas	Max
P-01-08	100	90	1987
P-01-09	100	90	2945

5.2.2.2 Basic WFS Enricher

En la Tabla 27 se detallan las diferentes pruebas que se llevaron a cabo para medir y comparar los tiempos de respuesta de diferentes invocaciones a todas las operaciones de WFS Básico.

Para probar la operación GetFeature se utilizaron los siguientes parámetros:

SERVICE=wfs&VERSION=1.1.0&REQUEST=GetFeature& &TYPENAME=eeip%3Aesp_libres

Para probar la operación GetCapabilities se utilizaron los siguientes parámetros:

SERVICE=wfs&VERSION =1.1.0&REQUEST=GetCapabilities

Para probar la operación DescribeFeatureType se utilizaron los siguientes parámetros:

SERVICE=wfs&VERSION=1.1.0&REQUEST=DescribeFeatureType&TYPENAME = eeip%3Aesp_libres

Tabla 27 - Plan de pruebas para Basic WFS Enricher

Referencia	Operación	Descripción
P-02-01	GetCapabilities	Realiza un GetCapabilities desde el consumidor hacia el proveedor, sin utilizar la GeoEEIP.
P-02-02	GetCapabilities	Realiza un GetCapabilities desde el consumidor hacia el proveedor, pasando por la GeoEEIP, la cual introduce modificaciones en la respuesta.
P-02-03	DescribeFeatureType	Realiza un DescribeFeatureType desde el cliente Web hacia el servidor de mapas del provvedor, sin utilizar la GeoEEIP.
P-02-04	DescribeFeatureType	Realiza un DescribeFeatureType desde el cliente Web hacia el servidor de mapas del proveedor, pasando por la GeoEEIP, la cual introduce modificaciones en la respuesta.
P-02-05	GetFeature	Realiza un GetFeature desde el cliente Web hacia el servidor de mapas del provvedor, sin utilizar la GeoEEIP. No se realiza enriquecimiento de la respuesta.
P-02-06	GetFeature	Realiza un GetFeature desde el cliente Web hacia el servidor de mapas del proveedor, pasando por la GeoEEIP, la cual enriquece la respuesta mediante un servicio de negocio.
P-02-07	GetFeature	Realiza un GetFeature desde el cliente Web hacia el servidor de mapas del proveedor. Realiza una invocación a un servicio de negocio. Realiza enriquecimiento mediante una transformación XSLT en el cliente. No se utiliza la plataforma.

En el caso de las prueba P-01-07 de la operación GetFeatureInfo, se implementó un enriquecimiento *ad hoc* de la respuesta en el cliente. Esto permite simular una solución en donde el cliente es responsable de realizar una invocación a un OWS (WMS en este caso) y otra a un WS SOAP, para luego integrar los datos de negocio con la información geográfica en propio cliente.

En la Tabla 28 se registran los principales resultados obtenidos en estas pruebas.

Tabla 28 - Resultados del plan de pruebas para WMS Enricher

Referencia	# Muestras	Min	Max	Media
P-02-01	100	9	835	396
P-02-02	100	175	2814	1654
P-02-03	100	14	690	165
P-02-04	100	228	3122	2120
P-02-05	100	867	23473	11904
P-02-06	100	9483	12542	11117
P-02-07	10	9572	14808	12809

6 Conclusiones

En este capítulo se presentan las conclusiones de este trabajo. En primer lugar, se realiza un resumen del trabajo realizado (Sección 6.1), para luego profundizar en las principales contribuciones (Sección 6.2) del mismo dentro del área y realizar algunas reflexiones a las que se llegó con el trabajo finalizado (Sección 6.3). Por último, se comentan posibles trabajos a futuro (Sección 6.4).

6.1 Resumen

Esta tesis se enfoca en la temática de integración de servicios geoespaciales con otros servicios provenientes de sistemas empresariales tradicionales.

En los últimos años, los GIS se han extendido significativamente, siendo hoy una pieza clave en un gran número de aplicaciones, entre las que se destacan las que utilizan tecnologías Web (Web GIS) y empresariales (Enterprise GIS). Los GIS actuales pueden responder a escenarios en donde un gran número de usuarios utilizan aplicaciones que permiten desde la visualización de un mapa dinámico hasta la aplicación de complejos modelos matemáticos.

Con el objetivo de facilitar el desarrollo de sistemas y aplicaciones interoperables basados en información geográfica, se han definido un conjunto de estándares, dentro de los cuales, resultan particularmente interesantes los relativos a Web services geoespaciales. Sin embargo, existen ciertas dificultades para integrar estos Web services en plataformas que ya utilizan servicios de otro tipo, en particular las plataformas de gobierno electrónico. En concreto, los Web services geoespaciales presentan problemas de incompatibilidad a nivel de protocolos y lenguajes con los Web services de negocio (basados en estándares W3C), así como limitaciones en aspectos relativos a la seguridad, transaccionalidad, enriquecimiento con información de negocio, orquestación de servicios, etc.

Por otro lado existen, dentro de las tecnologías de *middleware* actuales, soluciones que permiten atacar este tipo de problemática en base a la aplicación de patrones de integración empresariales. En particular, el Enterprise Service Bus permite la aplicación de una amplia gama de dichos patrones en base a estándares de la industria, por lo que resulta una alternativa de solución natural en contextos en donde su complejidad no sea una limitante.

Si bien con la introducción de un ESB se podrían resolver los problemas planteados, cada organización que se enfrente a los mismos debería implementar su propia solución en base a los mecanismos provistos por el producto ESB elegido. Esto plantea la necesidad de contar con una plataforma específica de dominio para el dominio GIS, que facilite la utilización de mecanismos orientados a la integración de servicios geoespaciales con sistemas empresariales tradicionales, partiendo de la base que existen requerimientos de integración que son comunes a muchas organizaciones (ej. enriquecer la respuesta de un servicio geoespacial con información de negocio proveniente de otro origen).

Como solución general a los problemas plantados, en esta tesis se propone la definición de una Plataforma Empresarial basada en ESB, que facilite integrar servicios geoespaciales con sistemas empresariales tradicionales orientados a servicios.

Dentro de los objetivos específicos planteados, el primer objetivo consistía en definir un marco de trabajo para la integración de servicios geoespaciales en un contexto de gobierno electrónico. En ese sentido, en el Capítulo 3 se realizó un amplio análisis de contexto en el que es aplicable una plataforma como la propuesta. Como parte de este análisis, se categorizaron los principales tipos de servicios geoespaciales, se identificaron los agentes típicos (proveedores y consumidores) en una plataforma de gobierno electrónico que integra dichos servicios, se identificaron grandes escenarios de uso en los que interactúan esos agentes, y se plantearon un primer conjunto de requerimientos tanto funcionales como no funcionales que se aplican a esos escenarios.

Como segundo objetivo se planteó la especificación de una Plataforma Específica de Dominio para el dominio GIS. Esta especificación se realiza en el Capítulo 4, comenzando por la definición de una arquitectura de referencia

para la plataforma y de un primer grupo de mecanismos geoespaciales que resuelvan los requerimientos planteados. A su vez, la plataforma fue definida en forma más general siguiendo la metodología de refinamientos sucesivos, y llegando a una formalización de la misma con el método Event-B.

El tercer objetivo planteado consistía en evaluar la factibilidad técnica de la propuesta y experimentar sobre ciertas propiedades de la misma. Para cumplir con este objetivo, se realizó una implementación de la plataforma sobre JBoss ESB y se realizaron pruebas automatizadas para evaluar el impacto en la performance de una plataforma de este tipo, comparándola con otras soluciones tradicionales de integración.

6.2 Contribuciones

Entendemos que los principales aportes que surgen de esta tesis son los siguientes:

- El relevamiento y estudio sistemático de un amplio conjunto de tecnologías que se relacionan entre sí, pero que han evolucionado en forma paralela dentro de ciertos nichos, como son las de los GIS, por un lado, y las de las Plataformas de Gobierno Electrónico Orientadas a Servicios y basadas en Enterprise Service Bus, por el otro. Dicho estudio permitió detectar ciertas carencias y limitaciones existentes, tanto a nivel de productos como de estándares, así como la ausencia de trabajos académicos que ataquen la problemática de integración de servicios geoespaciales dentro de las plataformas mencionadas.
- El diseño y especificación de una Plataforma Empresarial para la Integración de Servicios Geográficos, la cual permite utilizar los patrones de mediación y estándares presentes en los ESBs, para dar soporte a un conjunto extensible y combinable de requerimientos avanzados que amplían las capacidades originales de estos servicios. Esta plataforma se enmarca dentro de las Plataformas Específicas de Dominio, al proveer un conjunto de *building blocks* específicos para el entorno GIS, que se definen como mecanismos configurables y reusables que evitan la necesidad de desarrollos ad hoc.
- La definición de un marco de trabajo para la conceptualización de la plataforma en base a máquinas abstractas y niveles de refinamiento, lo que permitió realizar una especificación formal con el método formal Event-B.

6.3 Reflexiones

El trabajo realizado permitió reflexionar acerca de algunos aspectos sobre los que no se tenía pleno conocimiento al inicio del mismo.

Por un lado, la solución propuesta define un conjunto de mecanismos geoespaciales que son independientes del producto ESB utilizado pero que deben instanciarse sobre cada producto específico, en base a las particularidades de cada uno. Para el caso de JBossESB, las decisiones de implementación consistieron en realizar un mapeo entre los mecanismos y los constructores específicos provistos por esta herramienta (servicios y acciones) así como la elección de las acciones que implementaban los patrones de mediación necesarios. Se desprende de esto que la implementación de la plataforma sobre otro producto de ESB no es un tarea trivial, ya que no existe una uniformización entre los productos a nivel los patrones de mediación provistos (la capa de mecanismos integrados de ESB dentro de la Arquitectura de Referencia de la Figura 25). Es decir, no se cuenta con un modelo conceptual de patrones de mediación que permitan que los diversos productos cumplan en forma total o parcial con el mismo.

Esta incompatibilidad, abarca incluso a los ESB que trabajan sobre la misma tecnología de base (ej. Java EE), contrasta con la de los servidores de aplicaciones Java EE, en los cuales puede realizarse un *deployment* de una misma aplicación sobre productos de diferentes proveedores (Wildfly, TomEE, WebSphere AS, Glassfish, etc.)

En otro orden, este trabajo también permitió evaluar la aplicabilidad del método formal Event-B para la formalización de sistemas como el que se propone en esta tesis. Si bien no se contaba con experiencia dentro de esta metodología, pudieron comprobarse algunos aspectos positivos. En primer lugar, la capacidad de manejar el concepto de evento, que se aplica en forma natural a sistemas basados en el intercambio de mensajes, como son las plataformas de middleware. En segundo lugar, el concepto de máquina abstracta y la relación de refinamiento

entre máquinas abstractas, que permite trabajar en diferentes niveles de abstracción, lo que permite controlar la complejidad de la especificación en cada nivel.

Si bien se destacan estos puntos sobre la correcta elección de Event-B para la especificación de una propuesta de este tipo, es necesario señalar que como todo método formal este posee sus dificultades (ej. curva de aprendizaje del método y herramientas relacionadas) y conlleva un esfuerzo considerable.

6.4 Trabajo a Futuro

A partir del trabajo realizado, se han identificado algunas líneas de trabajo que se abren a futuro. Estas líneas pueden pensarse en base a tres grandes grupos: (i) extensión de la plataforma, que se enfoca en aumentar la capacidad funcional de la solución tal como ha sido definida, (ii) trabajos de profundización de la solución que quedaron fuera del alcance de este trabajo, y (iii) evolución de la plataforma para adaptarse a los cambios tecnológicos más relevantes en el corto y mediano plazo.

6.4.1 Evolución de SOAP-WFS Wrapper para soportar la última versión de WFS.

El mecanismo definido en 4.3.1.6 permite convertir los mensajes SOAP con pedidos y respuestas WFS a su forma estándar. Este mecanismo ha sido definido para poder interactuar con un servicio WFS Básico, según la definición vista en la Sección 2.1.5.2. A partir de la versión 2.0 del estándar, que recientemente ha sido incorporada a los servidores de mapas, las operaciones de consulta han sido ampliadas y se profundizó en la posibilidad de implementación utilizando mensajes SOAP, por lo que esta actualización debería ser trasladada al mecanismo, que actualmente soporta la versión anterior (1.1).

6.4.2 Extensión de SOAP-WFS Wrapper para WFS Transaccional.

A partir de la versión 2.0 del estándar, la capacidad transaccional, para crear, modificar y eliminar objetos geográficos, ha sido considerablemente mejorada, por lo que se puede prever que esta capacidad será más utilizada en el futuro de lo que ha sido hasta la fecha. Resulta interesante, por lo tanto, extender el mecanismo en cuestión con las operaciones de WFS Transaccional (*Transaction*, *LockFeature*, etc.)

6.4.3 Diseño de mecanismos para soportar otros OWS.

Los mecanismos diseñados en este trabajo, se enfocaron en los OWS más utilizados en la actualidad. Por un lado WMS y WMTS como estándares para brindar servicios de mapas, y por otro lado, WFS para servicios de datos. Existen otros OWS (Sección 2.1.5.3) que, si bien no son de uso general, tienen un gran potencial dentro de cierto tipo de aplicaciones específicas. Algunos de los que resultan particularmente interesantes a futuro son WCS, WPS y SOS. En el caso de WCS, que trabaja con datos *raster*, sería natural que su utilización acompañe, en la próxima década, la mayor disponibilidad que herramientas dentro de los GIS empresariales que están apareciendo, como son las bases de datos geográficas con soporte para rasters, cuando anteriormente sólo soportaban datos vectoriales. En el caso de WPS, este está orientado a proporcionar a clientes livianos, ciertos geoprocesos que tradicionalmente se ejecutaban en aplicaciones de escritorio sobre datos locales. Con la disponibilidad de los datos en forma remota, es esperable que los geoprocesos que trabajen sobre esos datos también se comiencen a ejecutar en forma remota. En el caso de SOS, este estándar está orientado a la publicación de mediciones provenientes de redes de sensores, las que están teniendo un impulso particular en estos momentos con la aparición de sensores de bajo costo para el público general y su incorporación en dispositivos de uso masivo, con los teléfonos móviles. En todos estos casos, sería útil contar con mecanismos tipo Wrapper, similares a los presentados, para poder integrar estos servicios en escenarios de uso de Web services SOAP, así como poder integrarlos a la plataforma para brindarles características avanzadas (ej. autenticación y control de acceso).

6.4.4 Implementación sobre otros productos de ESB.

La implementación realizada sobre JBoss ESB sirvió como prueba de concepto, factibilidad y desempeño de la plataforma. Sin embargo, este producto no ha evolucionado recientemente, estando previsto su reemplazo a corto plazo por un nuevo producto, llamado Switchyard, dentro de la JBoss SOA Platform. Sería interesante por lo tanto, implementar la plataforma sobre este nuevo producto, como forma de actualizar la solución y evaluar los impactos de reeimplementación sobre un producto diferente, teniendo en cuenta las consideraciones expresadas en la Sección 6.3.

6.4.5 Ampliación de la especificación formal y pruebas formales.

En este trabajo se ha elaborado una especificación formal inicial de la platafroma utilizando Event-B, pero la especificación completa en el nivel menos abstracto ha quedado fuera del alcance, por lo que sería deseable continuar el trabajo en ese sentido. A su vez, una línea de trabajo a futuro podría ser la realización de demostraciones formales en base a esta especificación. Como ejemplo, se podría probar formalmente que la respuesta a una operación GetMap realizada directamente contra un servidor de mapas por medio de WMS es igual a la respuesta que se obtiene luego de utilizar el mecanismo SOAP-WMS Wrapper para extraer el pedido de un mensaje SOAP y convertirlo a WMS. Este tipo de pruebas, a diferencia de las que se realizan mediante técnicas de *testing*, requieren de un nivel de experticia en el área y de un tiempo de elaboración muy extenso, por lo que suelen aplicarse únicamente para probar las propiedades más críticas de un sistema, trabajando en forma conjunta con investigadores especializados en el área de métodos formales.

6.4.6 Ampliación de las pruebas del prototipo

Las pruebas realizadas sobre el prototipo se han centrado en la performance y en particular en el tiempo de respuesta. Existen otras pruebas de performance importantes que debería realizarse en un futuro, como pruebas de carga y estrés para evaluar la escalabilidad. Interesa particularmente medir el comportamiento del sistema tomando como parámetros el número de invocaciones y el tamaño de los mensajes, ya que se ha observado que el uso más intensivo que se realiza en la plataforma es el que involucra transformaciones de mensajes.

Referencias

[1] C. M. López-Vázquez, M. A. Bernabé-Poveda, y U. P. M. Press, "Fundamentos de las Infraestructuras de Datos Espaciales," 1st edition. UPM Press, 2012.

[2] S. Shekhar y H. Xiong, "Encyclopedia of GIS," 1 edition. New York: Springer, 2007.

[3] P. Fu y J. Sun, "Web GIS: Principles and Applications." Redlands, Calif, ESRI Press, 2010.

[4] AGESIC, "Descripción Técnica de la Plataforma de Gobierno Electrónico." [En línea]. Disponible en: http://www.agesic.gub.uy/innovaportal/file/1454/1/capitulo_3.pdf

[5] G. Edwards y N. Medvidovic, "A Methodology and Framework for Creating Domain-Specific Development Infrastructures», en 23rd IEEE/ACM International Conference on Automated Software Engineering, 2008. ASE 2008, 2008, pp. 168-177.

[6] B. Rienzi, L. González y R. Ruggia, "Towards an ESB-Based Enterprise Integration Platform for Geospatial Web Services," presentado en The Fifth International Conference on Advanced Geographic Information Systems, Applications, and Services - GEOProcessing 2013, Niza, Francia, 2013. [En línea].Disponible en:

http://portal.opengeospatial.org/files/?artifact_id=20555. [Accedido: 27-ene-2016].

[7] S. Fazal, "GIS Basics. New Age International," 2008.

[8] M. F. Worboys y M. Duckham, "GIS: A Computing Perspective," 2nd edition. Boca Raton, Fla: CRC Press, 2004.

[9] F. Harvey, "A Primer of GIS: Fundamental Geographic and Cartographic Concepts." Guilford Press, 2010.

[10] Z.-R. Peng y M.-H. Tsou, "Internet GIS: Distributed Geographic Information Services for the Internet and Wireless Networks." John Wiley & Sons, 2003.

[11] P. A. Longley, M. Goodchild, D. J. Maguire, y D. W. Rhind, "Geographic Information Systems and Science," 3rd edition. Hoboken, NJ: Wiley, 2010.

[12] A. Soylu, F. Mödritscher, F. Wild, P. D. Causmaecker, y P. Desmet, "Mashups by orchestration and widget-based personal environments: Key challenges, solution strategies, and an application," Program: electronic library and information systems, vol. 46, n.º 4, pp. 383-428, 2012.

[13] "Web Services Description Requirements." [En línea]. Disponible en:

http://www.w3.org/TR/ws-desc-reqs/#nonNormDefs. [Accedido: 19-jun-2014].

[14] "Web Services Architecture." [En línea]. Disponible en:

http://www.w3.org/TR/ws-arch/#whatis. [Accedido: 19-jun-2014].

[15] N. Mitra, Y. Lafon, "SOAP Version 1.2 Part 0: Primer," 2nd edition, World Wide Web Consortium (W3C), 2007. [En línea]. Disponible en:

https://www.w3.org/TR/soap12-part0/ [Accedido: 25-feb-2016].

[16] M. Papazoglou, "Web Services: Principles and Technology," 1st ed. Prentice Hall, 2007.

[17] E. Christensen, F. Curbera, G. Meredith, S. Weerawarana, "Web Services Description Language (WSDL) 1.1," World Wide Web Consortium (W3C), 2001. [En línea]. Disponible en:

https://www.w3.org/TR/wsdl [Accedido: 25-feb-2016].

[18] S. Weerawarana, F. Curbera, F. Leymann, T. Storey, y D. F. Ferguson, "Web Services Platform Architecture: SOAP, WSDL, WS-Policy, WS-Addressing, WS-BPEL, WS-Reliable Messaging, and More", 1st edition. Upper Saddle River, NJ: Prentice Hall, 2005.

[19] E. Newcomer e I. Robinson, "Web services atomic transaction (WS-atomic transaction) version 1.2," 2009.

[20] E. Newcomer e I. Robinson, "Web services business activity (WS-businessactivity) version 1.2," 2009.

[21] C. Pautasso, O. Zimmermann, y F. Leymann, "Restful Web Services vs. 'Big' Web Services: Making the Right Architectural Decision," presentado en Proceedings of the 17th International Conference on World Wide Web, New York, NY, USA, 2008, pp. 805–814.

[22] G. Percivall, "The OpenGIS Abstract Specification. Topic 12: OpenGIS Service Architecture," OGC 02-112, version 4.3, 2001. [En línea]. Disponible en:

http://portal.opengeospatial.org/files/?artifact_id=1221. [Accedido: 20-jun-2014].

[23] A. Whiteside y J. Greenwood, "OGC Web Services Common Standard," OGC 06-121r9, version 2.0.0, 2010. [En línea]. Disponible en:

http://portal.opengeospatial.org/files/?artifact_id=38867. [Accedido: 22-jun-2014].

[24] A. Whiteside, "OpenGIS Web Services Architecture Description," OGC 05-042r2, version 0.1.0, 2005. [En línea]. Disponible en:

http://portal.opengeospatial.org/files/?artifact_id=13140. [Accedido: 22-jun-2014].

[25] J. de la Beaujardiere, “OpenGIS Web Map Server Implementation Specification,” OGC 06-042, version 1.3, 2006.

[26] P. Panagiotis y A. Vretanos, “Web Feature Service Implementation Specification”, OGC 04-094, version 1.1.0, 2005. [En línea]. Disponible en:

http://portal.opengeospatial.org/files/?artifact_id=8339

[27] P. Panagiotis y A. Vretanos, “OpenGIS Web Feature Service 2.0 Interface Standard”, OGC 09-025r1, version 2.0.0, 2010. [En línea]. Disponible en:

http://portal.opengeospatial.org/files/?artifact_id=39967

[28] J. Masó, K. Pomakis, N. Juliá, “OpenGIS Web Map Tile Service Implementation Standard,” OGC 07-057r7, version 1.0.0, 2010. [En línea]. Disponible en:

http://portal.opengeospatial.org/files/?artifact_id=35326. [Accedido: 19-jun-2014].

[29] D. Nebert, A. Whiteside, P. Vretanos, “OpenGIS Catalogue Services Specification,” OGC 07-006r1, version 2.0.2, 2007. [En línea]. Disponible en:

http://portal.opengeospatial.org/files/?artifact_id=20555. [Accedido: 19-jun-2014].

[30] P. Schut, “OpenGIS Web Processing Service,” OGC 05-007r7, version 1.0.0, 2007. [En línea]. Disponible en:

http://portal.opengeospatial.org/files/?artifact_id=24151. [Accedido: 23-jun-2014].

[31] A. Matheus y J. Herrmann, “Geospatial eXtensible Access Control Markup Language (GeoXACML),” OGC 11-007, version 1.0.1, 2011. [En línea]. Disponible en:

http://portal.opengeospatial.org/files/?artifact_id=42734. [Accedido: 23-jun-2014].

[32] D. A. Chappell, “Enterprise Service Bus,” 1st edition, Sebastopol, Calif: O’Reilly Media, 2004.

[33] I. B. M. Redbooks, “Patterns: Implementing an Soa Using an Enterprise Service Bus.” Research Triangle Park, NC: IBM, 2004.

[34] T. Rademakers y J. Dirksen, “Open-Source ESBs in Action,” 1st edition. Greenwich, CT: Manning Publications, 2008.

[35] M. P. & P. Team, “Microsoft® Application Architecture Guide,” 2nd edition. Redmond, Wash.: Microsoft Press, 2009.

[36] G. Hohpe y B. Woolf, “Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions.” Addison-Wesley Professional, 2003.

[37] H. Wylie y P. Lambros, “Enterprise Connectivity Patterns: Implementing integration solutions with IBM's Enterprise Service Bus products,” 10-Mar-2009. [En línea]. Disponible en:

http://www.ibm.com/developerworks/library/ws-enterpriseconnectivitypatterns/index.html. [Accedido: 15-ene-2015].

[38] T. Erl, “SOA Design Patterns,” 1st edition. Upper Saddle River, NJ: Prentice Hall, 2009.

[39] L. González, R. Ruggia, J. Abin, G. Llambías, R. Sosa, B. Rienzi, D. Bello, F. Álvarez, “A service-oriented integration platform to support a joined-up e-government approach: the uruguayan experience,” Advancing Democracy, Government and Governance (pp. 140-154). Springer Berlin Heidelberg, 2012.

[40] G. Llambías, L. González, y R. Ruggia, “Towards an Integration Platform for Bioinformatics Services,” presentado en The 1st Workshop on Pervasive Analytical Service Clouds for the Enterprise and Beyond, 2013.

[41] G. Llambías, “Hacia una Plataforma de Integración de Servicios Bioinformáticos», PEDECIBA Informática, Instituto de Computación – Facultad de Ingeniería – Universidad de la República, 2013.

[42] R. Sosa, “Integración de Servicios Geográficos en Plataformas de Gobierno Electrónico,” PEDECIBA Informática, Instituto de Computación – Facultad de Ingeniería – Universidad de la República, 2011.

[43] “EPA Information Resources Management Strategic Plan”, United States Environmental Protection Agency, 2015.

[44] “Geospatial Line of Business. Common Solutions and Target Architecture”, Exectuive Office of the President of the United States, 2007.

[45] B. Rienzi, R. Sosa, P. Foti y L. González, “Benefits and challenges of using geographic information systems to enhance social security services,” presentado en 6th International Policy and Research Conference on Social Security, 2010.

[46] P. Jaarinen, “Architectures of Integrating GIS with Enterprise Systems,” Master thesis, Faculty of Engineering and Architecture, School of Science and Technology, Aalto Unversity, 2010.

[47] F. Samadzadegan, S. Saeedi, A. Alvand, y M. Hasanlou, “Enterprise GIS for Municipalities - A Service Oriented Approach,” The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, vol. XXXVII, pp. 1133-1136, 2008.

[48] U. Skjelbo, J. J. Jensen y L. J. Jørgensen, “Geographic Information in Enterprise Architecture - Principles supporting e-Government business functions,” Informi GIS AS, 2009.

[49] ESRI, “Geospatial Service-Oriented Architecture (SOA) ,” ESRI White Paper, 2007.

[50] T. Fang Rao et al, “Cache Mediation Pattern Specification: An Overview,” [En línea]. Disponible en: http://www.ibm.com/developerworks/webservices/library/ws-soa-cachemed/index.html. [Accedido: 16-ene-2015].

[51] L. González, “Plataforma ESB Adaptativa para Sistemas Basados en Servicios,” PEDECIBA Informática, Instituto de Computación – Facultad de Ingeniería – Universidad de la República, 2011.

[52] J. R. Abrial, “Modeling in Event-B: System and Software Engineering,” 1 edition. Cambridge ; New York: Cambridge University Press, 2010.

[53] I. Tounsi, M. Hadj Kacem, A. M. Hadj Kacem, K. Drira, “Modeling and refinement SOA design patterns with Event-B method,” 2012

[54] S. Padiar, “A Study In The Use Of Event-B For System Development From A Software Engineering Viewpoint,” 2012

[55] JBoss ESB Development Team, “JBoss ESB 4.12 Programmers Guide.” [En línea]. Disponible en: http://docs.jboss.org/jbossesb/docs/4.12/manuals/html/Programmers_Guide/index.html#section-Aggregator

[56] L. DiMaggio, K. Conner, M. B. Kumar, T. Cunningham, “JBoss ESB Beginner’s Guide”. [En línea]. Disponible en: http://www.amazon.com/JBoss-ESB-Beginners-Guide-DiMaggio/dp/1849516588. [Accedido: 09-nov-2015].

[57] J. Sonnet y C. Savage, “OWS 1.2 SOAP Experiment,” OGc 03-014, versión 0.8, 2003. [En línea]. Disponible en: http://portal.opengeospatial.org/files/?artifact_id=1337. [Accedido: 15-ene-2015].

[58] INSPIRE, “TechnicalGuidance ViewServices v3.1.” [En línea]. Disponible en: http://inspire.ec.europa.eu/documents/Network_Services/TechnicalGuidance_ViewServices_v3.1.pdf[Accedido: 23-set-2015].

Apéndice 1. Especificación formal

Especificación completa de GeoEEIP_level0_mac

MACHINE

GeoEEIP_level0_mac > máquina abstracta que especifica la plataforma en el nivel 0.

VARIABLES

GeoEEIP > representa la plataforma

OWSClient > representa los clientes OWS

SOAPClient > representa los clientes SOAP

OWSService > representa los servicios OWS

SOAPService > representa los servicios SOAP (servicios de negocio).

Message > representa los mensajes que se intercambian entre participantes

RequestMessage > representa los pedidos

ResponseMessage > representa las respuestas

Participant > representa los participantes que interactúan mediante la plataforma

Client > representa los clientes de la plataforma

ConsumesOWS > representa la relación entre GeoEEIP y los OWSService (la plataforma consume servicios geoespaciales).

ConsumesSOAP > representa la relación entre GeoEEIP y los SOAPService (la plataforma consume servicios de negocio).

GeoEEIPSendsMessage > representa una relación entre GeoEEIP y Message (la plataforma envía mensajes)

GeoEEIPReceivesMessage > representa una relación entre GeoEEIP y Message (la plataforma recibe mensajes)

OWSSendsResponse > representa la relación entre los OWSService y los ResponseMessage (un servicio geoespacial envía un mensaje de respuesta)

SOAPSendsResponse > representa la relación entre los SOAPService y los ResponseMessage (un servicio de negocio envía un mensaje de respuesta)

To > representa una relación entre un Message y un Participant (los mensajes tienen un destinatario)

From > representa una relación entre un Message y un Participant (los mensajes tienen un emisor)

Uses > representa una relación entre un Client y la GeoEEIP

ClientSendsRequest > representa una relación entre un Client y un RequestMessage (un cliente envía un pedido)

INVARIANTS

Message_type : Message ∈ ℙ(Message_SET)

RequestMessage_type : RequestMessage ∈ ℙ(Message)

ResponseMessage_type : ResponseMessage ∈ ℙ(Message)

Participant_type : Participant ∈ ℙ(Participant_SET)

Client_type : Client ∈ ℙ(Participant)

To_type : To ∈ Message → Participant

From_type : From ∈ Message ↔ Participant

ClientSendsRequest_type : ClientSendsRequest ∈Client ↔ RequestMessage

GeoEEIP_type : GeoEEIP ∈ℙ(Participant)

OWSClient_type : OWSClient ∈ℙ(Client)

SOAPClient_type : SOAPClient ∈ℙ(Client)

OWSService_type : OWSService ∈ℙ(Participant)

SOAPService_type : SOAPService ∈ℙ(Participant)

ConsumesOWS_type : ConsumesOWS ∈GeoEEIP ⇸ OWSService

ConsumesSOAP_type : ConsumesSOAP ∈GeoEEIP ↔ SOAPService

GeoEEIPSendsMessage_type : GeoEEIPSendsMessage ∈GeoEEIP ↔ Message

GeoEEIPReceivesMessage_type : GeoEEIPReceivesMessage ∈GeoEEIP ↔ Message

OWSSendsResponse_type : OWSSendsResponse ∈OWSService ↔ ResponseMessage

SOAPSendsResponse_type : SOAPSendsResponse ∈SOAPService ↔ ResponseMessage

Uses_type : Uses ∈Client ↔ GeoEEIP

Invariant1 : RequestMessage ∩ ResponseMessage = ∅

EVENTS

INITIALISATION:

BEGIN

GeoEEIP_init : GeoEEIP :=∅

OWSClient_init : OWSClient :=∅

SOAPClient_init : SOAPClient :=∅

OWSService_init : OWSService :=∅

SOAPService_init : SOAPService :=∅

Message_init : Message :=∅

RequestMessage_init : RequestMessage :=∅

ResponseMessage_init : ResponseMessage :=∅

Participant_init : Participant :=∅

Client_init : Client :=∅

ConsumesOWS_init : ConsumesOWS :=∅

ConsumesSOAP_init : ConsumesSOAP :=∅

GeoEEIPSendsMessage_init : GeoEEIPSendsMessage :=∅

GeoEEIPReceivesMessage_init : GeoEEIPReceivesMessage :=∅

OWSSendsResponse_init : OWSSendsResponse :=∅

SOAPSendsResponse_init : SOAPSendsResponse :=∅

To_init : To :=∅

From_init : From :=∅

Uses_init : Uses :=∅

ClientSendsRequest_init : ClientSendsRequest :=∅

END

owsSrvAdded:

ANY

self

s

WHERE

s_type : s ∈ OWSService

self_type : self ∈ GeoEEIP

owsSrvAdded_Guard1 : self ↦ s ∉ ConsumesOWS

THEN

owsSrvAdded_Action1 : ConsumesOWS := ConsumesOWS ∪ {self ↦ s}

END

soapSrvAdded:

ANY

self

s

WHERE

s_type : s ∈ SOAPService

self_type : self ∈ GeoEEIP

soapSrvAdded_Guard1 : self ↦ s ∉ ConsumesSOAP

THEN

soapSrvAdded_Action1 : ConsumesSOAP := ConsumesSOAP ∩ {self ↦ s}

END

requestReceived:

ANY

self

m

c

WHERE

m_type : m ∈ RequestMessage

c_type : c ∈ Client

self_type : self ∈ GeoEEIP

requestReceived_Guard1 : To(m) = self

requestReceived_Guard2 : From(m) = c

requestReceived_Guard3 : c ↦ m ∈ ClientSendsRequest

THEN

requestReceived_Action1 : GeoEEIPReceivesMessage := GeoEEIPReceivesMessage ∪ {self ↦ m}

requestReceived_Action2 : ClientSendsRequest :∈ ClientSendsRequest \{c ↦ m}

requestReceived_Action3 : Uses :∈ Uses ∪{c ↦ self}

END

owsResponseReceived:

ANY

self

m

s

WHERE

m_type : m ∈ ResponseMessage

s_type : s ∈ OWSService

self_type : self ∈ GeoEEIP

owsResponseReceived_Guard1 : s ↦ m ∈ OWSSendsResponse

owsResponseReceived_Guard2 : From(m) = s

owsResponseReceived_Guard3 : To(m) = self

THEN

owsResponseReceived_Action1 : GeoEEIPReceivesMessage :∈ GeoEEIPReceivesMessage ∪{self ↦ m}

END

soapResponseReceived:

ANY

self

m

s

WHERE

m_type : m ∈ ResponseMessage

s_type : s ∈ SOAPService

self_type : self ∈ GeoEEIP

soapResponseReceived_Guard1 : s ↦ m ∈ SOAPSendsResponse

soapResponseReceived_Guard2 : From(m) = s

soapResponseReceived_Guard3 : To (m) = self

THEN

soapResponseReceived_Action1 : GeoEEIPReceivesMessage :∈ GeoEEIPReceivesMessage ∪{self ↦ m}

END

END

Especificación completa de GeoEEIP_level0_mac

MACHINE

GeoEEIP_level1_mac > máquina abstracta que especifica la plataforma en el nivel 1.

REFINES

GeoEEIP_level0_mac

SEES

GeoEEIP_level1_ctx > contexto (se agrega en este nivel)

VARIABLES

GeoMech > representa los mecanismos de integración geoespaciales (se agrega en este nivel)

ServiceType > representa los tipos de servicio (se agrega en este nivel)

Protocol > representa los protocolos (se agrega en este nivel)

SupportsSrvType ›representa una relación entre GeoMech y ServiceType (un mecanismo soporta un conjunto de tipos de servicio (WMS, WFS, etc.) y recíprocamente

SupportsProtocol ›representa una relación entre GeoMech y Protocol (un mecanismo soporta un protocolo (OWS o SOAP) y recíprocamente.

GeoMechProcessesReq > representa una relación entre GeoMech y RequestMessage (un mecanismo está procesando un pedido)

GeoMechProcessesRsp representa una relación entre GeoMech y RequestMessage (un mecanismo está procesando una respuesta)

Features › representa una relación entre GeoEEIP y GeoMech (la plataforma posee un conjunto de mecanismos geoespaciales)

GeoEEIP > representa la plataforma

OWSClient > representa los clientes OWS

SOAPClient > representa los clientes SOAP

OWSService > representa los servicios OWS

SOAPService > representa los servicios SOAP (servicios de negocio).

Message > representa los mensajes que se intercambian entre participantes

RequestMessage > representa los pedidos

ResponseMessage > representa las respuestas

Participant > representa los participantes que interactúan mediante la plataforma

Client > representa los clientes de la plataforma

ConsumesOWS > representa la relación entre GeoEEIP y los OWSService (la plataforma consume servicios geoespaciales).

ConsumesSOAP > representa la relación entre GeoEEIP y los SOAPService (la plataforma consume servicios de negocio).

GeoEEIPSendsMessage > representa una relación entre GeoEEIP y Message (la plataforma envía mensajes)

GeoEEIPReceivesMessage > representa una relación entre GeoEEIP y Message (la plataforma recibe mensajes)

OWSSendsResponse > representa la relación entre los OWSService y los ResponseMessage (un servicio geoespacial envía un mensaje de respuesta)

SOAPSendsResponse > representa la relación entre los SOAPService y los ResponseMessage (un servicio de negocio envía un mensaje de respuesta)

To > representa una relación entre un Message y un Participant (los mensajes tienen un destinatario)

From > representa una relación entre un Message y un Participant (los mensajes tienen un emisor)

Uses > representa una relación entre un Client y la GeoEEIP

ClientSendsRequest > representa una relación entre un Client y un RequestMessage (un cliente envía un pedido)

INVARIANTS

GeoMech_type: GeoMech ∈ ℙ(GeoMech_SET)

ServiceType_type: ServiceType ∈ ℙ(ServiceType_SET)

Protocol_type: Protocol ∈ ℙ(Protocol_SET)

SupportsSrvType_type: SupportsSrvType ∈ GeoMech <<->> ServiceType

SupportsProtocol_type: SupportsProtocol ∈ GeoMech <<->> Protocol

GeoMechProcessesReq_type: GeoMechProcessesReq ∈ GeoMech ↔ RequestMessage

GeoMechProcessesRsp_type: GeoMechProcessesRsp ∈ GeoMech ↔ ResponseMessage

Features_type: Features ∈ GeoEEIP ↔ GeoMech

HasType_type: HasType ∈ Message → ServiceType

GeoMechProcessesRsp.Injective: GeoMechProcessesRsp~ ∈ ResponseMessage ⤔ GeoMech

GeoMechProcessesReq.Injective: GeoMechProcessesReq~ ∈ RequestMessage ⤔ GeoMech

Message_type : Message ∈ ℙ(Message_SET)

RequestMessage_type : RequestMessage ∈ ℙ(Message)

ResponseMessage_type : ResponseMessage ∈ ℙ(Message)

Participant_type : Participant ∈ ℙ(Participant_SET)

Client_type : Client ∈ ℙ(Participant)

To_type : To ∈ Message → Participant

From_type : From ∈ Message ↔ Participant

ClientSendsRequest_type : ClientSendsRequest ∈ Client ↔ RequestMessage

GeoEEIP_type : GeoEEIP ∈ ℙ(Participant)

OWSClient_type : OWSClient ∈ ℙ(Client)

SOAPClient_type : SOAPClient ∈ ℙ(Client)

OWSService_type : OWSService ∈ ℙ(Participant)

SOAPService_type : SOAPService ∈ ℙ(Participant)

ConsumesOWS_type : ConsumesOWS ∈ GeoEEIP ⇸ OWSService

ConsumesSOAP_type : ConsumesSOAP ∈ GeoEEIP ↔ SOAPService

GeoEEIPSendsMessage_type : GeoEEIPSendsMessage ∈ GeoEEIP ↔ Message

GeoEEIPReceivesMessage_type : GeoEEIPReceivesMessage ∈ GeoEEIP ↔ Message

OWSSendsResponse_type : OWSSendsResponse ∈ OWSService ↔ ResponseMessage

SOAPSendsResponse_type : SOAPSendsResponse ∈ SOAPService ↔ ResponseMessage

Uses_type : Uses ∈ Client ↔ GeoEEIP

Invariant1 : RequestMessage ∩ ResponseMessage = ∅

EVENTS

INITIALISATION:

BEGIN

soapResponseReceived:

ANY

self

m

s

WHERE

m_type : m ∈ ResponseMessage

s_type : s ∈ SOAPService

self_type : self ∈ GeoEEIP

soapResponseReceived_Guard1 : s ↦ m ∈ SOAPSendsResponse

soapResponseReceived_Guard2 : From(m) = s

soapResponseReceived_Guard3 : To (m) = self

THEN

soapResponseReceived_Action1 : GeoEEIPReceivesMessage := GeoEEIPReceivesMessage ∪ {self ↦ m}

END

owsSrvAdded:

ANY

self

s

WHERE

s_type : s ∈ OWSService

self_type : self ∈ GeoEEIP

owsSrvAdded_Guard1 : self ↦ s ∉ ConsumesOWS

THEN

owsSrvAdded_Action1 : ConsumesOWS := ConsumesOWS ∪ {self ↦ s}

END

soapSrvAdded:

ANY

self

s

WHERE

s_type : s ∈ SOAPService

self_type : self ∈ GeoEEIP

soapSrvAdded_Guard1 : self ↦ s ∉ ConsumesSOAP

THEN

soapSrvAdded_Action1 : ConsumesSOAP := ConsumesSOAP ∩ {self ↦ s}

END

processRequest:

REFINES

processRequest

ANY

self

m

gm

WHERE

self_type: self ∈ GeoEEIP

m_type: m ∈ RequestMessage

gm_type: gm ∈ GeoMech

self_type: self ∈ GeoEEIP

processRequest_Guard1: (self ↦ m) ∈ GeoEEIPReceivesMessage

processRequest_Guard2: HasType(m) ∈ SupportsSrvType(gm)

processRequest_Guard3: From(m) ∈ Client

THEN

processRequest _Action1 : GeoMechProcessesRequest := GeoMechProcessesRequest ∪ {gm ↦ m}

processRequest _Action2 : GeoEEIPReceivesMessage := GeoEEIPReceivesMessage \ { self ↦ m}

END

processOWSResponse:

REFINES

processOWSResponse

ANY

self

m

gm

WHERE

self_type: self ∈ GeoEEIP

m_type: m ∈ ResponseMessage

gm_type: gm ∈ GeoMech

processResponse_Guard1: (self ↦ m) ∈ GeoEEIPReceivesMessage

processResponse_Guard2: HasType(m) ∈ SupportsSrvType(gm)

processResponse_Guard3: From(m)∈OWSService

THEN

processOWSResponse_Action1 : GeoMechProcessesOWSResponse :=GeoMechProcessesOWSResponse ∪{gm ↦ m}

processOWSResponse_Action2 : GeoEEIPReceivesMessage :=GeoEEIPReceivesMessage \{ self ↦m}

END

processSOAPResponse:

REFINES

processSOAPResponse

ANY

self

m

gm

WHERE

self_type: self ∈GeoEEIP

m_type: m ∈ResponseMessage

gm_type: gm ∈GeoMech

processResponse_Guard1: (self↦m) ∈GeoEEIPReceivesMessage

processResponse_Guard2: HasType(m)∈SupportsSrvType(gm)

processResponse_Guard3: From(m)∈SOAPService

THEN

processSOAPResponse _Action1 : GeoMechProcessesSOAPResponse :=GeoMechProcessesSOAPResponse ∪{gm ↦m}

processSOAPResponse _Action2 : GeoEEIPReceivesMessage :=GeoEEIPReceivesMessage \{ self ↦m}

END

END

Apéndice 2. Glosario

- **Capa geográfica**

  Colección lógica de entidades geográficas de un mismo tipo (ej. capa de lagos)

- **CRS, Coordinate Reference System**

  También llamado *SRS* (*Spatial Reference System*). Es cualquier sistema basado en coordenadas que permita definir la ubicación de una entidad geográfica, tanto en forma global como regional o local.

- **ESB, Enterprise Service Bus**

  Plataforma de integración basada en estándares que combina mensajería, Web services, transformación de datos y ruteo inteligente para conectar y coordinar, de forma confiable, la interacción de un gran número de aplicaciones.

- **GIS, Geographical Information System**

  Sistema de información que integra hardware, software y datos para capturar, gestionar, analizar y desplegar todo tipo de información georreferenciada.

- **OGC, Open Geospatial Consortium**

  Consorcio internacional que lidera el desarrollo de estándares para intercambio de información geográfica, en particular en lo relativo a Web services geoespaciales.

- **OWS, OGC Web Service**

  Todo estándar de Web services geoespaciales especificado por OGC. Ejemplos: WFS, WMS, WMTS, etc.

- **SOAP**

  Estándar W3C que define un protocolo para el intercambio de información estructurada en formato XML, utilizado en Web services empresariales.

- **WFS, Web Feature Service**

  Estándar OGC/ISO que define un protocolo para consultar, modificar y eliminar información geográfica.

- **WMS, Web Map Service**

  Estándar OGC/ISO que permite obtener mapas dinámicos a partir de un conjunto de capas geográficas.

- **WMTS, Web Map Tile Service**

  Estándar OGC que permite obtener mapas dinámicos a partir de un conjunto de capas geográficas, dividiendo cada mapa en una matriz de teselas.