

Integración GIS-PGE

Ingeniería en Computación

Facultad de Ingeniería de la Universidad de la República Oriental del Uruguay

Mayo - 2015

Integrantes:

Luciana Canales

Maximiliano Felix

Alejandro Remiro

Tutora: Raquel Sosa.

Resumen

Gobierno electrónico es la aplicación de tecnologías de la información y la comunicación y en particular la internet para lograr una mejor gestión para los ciudadanos. Es por esto que se crean las Plataformas de Gobierno Electrónico (PGE), como es el caso de Uruguay, que cuenta con la Agencia para el Desarrollo del Gobierno Electrónico y la Sociedad de la Información y del Conocimiento (AGESIC). AGESIC desarrolla su PGE para facilitar la integración de los servicios ofrecidos por los organismos, brindándolos a los ciudadanos a través de internet.

Los datos de interés gubernamental brindados a los ciudadanos a través de la internet puede ser muy variado y en particular se encuentran los servicios que brindan información geográfica, como ser los padrones de un departamento, datos meteorológicos georeferenciados, información de suelos, entre otros.

La tesis de maestría de Raquel Sosa, Integración de Servicios Geográficos en Plataformas de Gobierno Electrónico Agosto 2011, muestra un estudio realizado sobre la integración de servicios geográficos con la PGE en Uruguay. Allí se explica la problemática que existe en la actualidad para dicha integración debido a que la PGE da soporte para web services basados en el estándar SOAP y no así para web services basados en el estándar REST. Los servicios de información geográfica generalmente se basan en los protocolos WMS y WFS y a su vez estos se basan en REST y es por este motivo que la PGE no ofrece la posibilidad de acceder a datos geográficos ofrecidos por los diferentes organismos. Lo que plantea la tesis es una arquitectura abstracta que permite resolver dicha problemática.

Este proyecto tiene como objetivo desarrollar la solución que se propone en la tesis de maestría. Dicha solución identifica cinco escenarios de integración de información geográfica en el marco de gobierno electrónico tomando como referencia la arquitectura de Uruguay y se propone la inserción de Componentes de Transformación de Protocolos (CTP).

Para desarrollar la solución propuesta en la tesis de maestría se estudian los escenarios de aplicación para la integración de web services geográficos en la PGE, la arquitectura teórica propuesta y sus mecanismos de transformaciones. Como parte del análisis se consideran las alternativas de arquitecturas concretas y el diseño de los componentes subyacentes. Luego se implementa un prototipo de los CTPs que permiten cubrir la mayor parte de los escenarios analizados. Y por último se desarrolla un caso de estudio para validar lo implementado.

Contenido:

1	Introducción	7
1.1	Motivación y Contexto.....	7
1.2	Objetivos	7
1.3	Resultados esperados	8
1.4	Resultados alcanzados.....	8
1.5	Organización del documento	8
2	Marco de trabajo.....	11
2.1	Web Services Geográficos.....	11
2.1.1	WMS.....	11
2.1.2	WFS	11
2.2	Estándares SOAP	12
2.2.1	SOAP.....	12
2.2.2	REST.....	12
2.2.3	WS-SECURITY	13
2.2.4	WS-ADDRESSING.....	14
2.2.5	MTOM	15
2.3	Tecnologías y productos estudiados.....	15
2.3.1	ESB	15
2.3.2	Servidores de mapas	17
2.4	Plataforma de eGob de Uruguay.....	18
2.4.1	Plataforma de Interoperabilidad	19
3	Análisis y arquitectura	25
3.1	Escenarios propuestos en la tesis.....	25
3.1.1	Público general accediendo a información geográfica	25
3.1.2	Público especializado consultando información.....	25
3.1.3	Instituciones colaborando en la creación de información geográfica	25
3.1.4	Instituciones colaborando en trámites	26
3.1.5	Público generando información geográfica	26
3.1.6	Resumen análisis de escenarios.....	26
3.2	Solución propuesta en la tesis.....	27
3.3	Casos de uso.....	29

3.3.1	Actores	30
3.3.2	CU: Público general consultando información.	30
3.3.3	CU: Instituciones colaborando en trámites	30
3.4	Decisiones de arquitectura	31
3.4.1	Mapeo de las direcciones del servicio.....	31
3.4.2	Opciones de despliegue	32
4	Diseño e implementación	37
4.1	Arquitectura	37
4.1.1	Diagramas de actividad	37
4.2	Implementación, diagrama de componentes, productos utilizados	38
4.2.1	CTP RestConnector.....	39
4.2.2	CTP SoapConnector.....	40
4.2.3	Configuración de los ctps y pge	41
4.2.4	Productos utilizados.....	43
4.3	Problemas técnicos encontrados	44
4.3.1	JBossESB y deploy de web services.....	44
4.3.2	GetCapabilities y URLs declaradas	44
4.3.3	GetCapabilities en MapServer	44
4.3.4	GetFeatureInfo y MapServer	44
4.3.5	Problemas con la implementación del STS.....	44
5	Caso de Estudio	47
5.1	Marco de trabajo	47
5.2	Descripción.....	47
5.3	Aplicación.....	47
5.3.1	Escenario 1	47
5.3.2	Escenario 2	48
5.3.3	Escenario 4	49
6	Conclusiones.....	51
6.1	Gestión de proyecto	52
7	Trabajo a Futuro	55
8	Referencias.....	57

Anexo 1: Documento de Arquitectura de Software

Anexo 2: Documento de Casos de Uso

1 Introducción

1.1 Motivación y Contexto

Con el paso del tiempo los avances en la tecnología han logrado que las mismas sean implantadas prácticamente en todos los ámbitos de la sociedad y en particular en las gestiones del gobierno. Es por esto que surge la necesidad de crear lo que se denomina gobierno electrónico.

Varias son las definiciones que se han propuesto a lo largo del tiempo para describir el concepto "gobierno electrónico", como ser la OCDE (Organización para la Cooperación y el Desarrollo Económicos) que lo define como *"El uso de las tecnologías de la información y comunicación (TIC's), particularmente la Internet, como una herramienta para alcanzar un mejor gobierno"* [14].

Hoy en día existen varios países que han incorporado este concepto en las actividades del gobierno, como es el caso de Uruguay. Con el objetivo de mejorar los servicios brindados a los ciudadanos uruguayos se crea el organismo AGESIC (Agencia para el Desarrollo del Gobierno de Gestión Electrónica y la Sociedad de la Información y del Conocimiento) [13].

Para lograr los objetivos propuesto AGESIC cuenta con la Plataforma de Gobierno Electrónico (PGE) [1] que facilita la integración de los servicios ofrecidos por los organismos, brindándolos a los ciudadanos a través de internet. Las plataformas de gobierno electrónico generalmente se basan en una Arquitectura Orientada a Servicios (SOA).

Los servicios de información geográfica de a poco se han ido integrando a las operaciones del Estado, lo cual implica que las administraciones para responder a estas necesidades se planteen la integración de servicios geográficos en las plataformas de gobierno electrónico.

A pesar de los avances en las TIC's la integración entre los servicios de información geográfica y las plataformas de gobierno electrónico presentan algunas dificultades debido a que las plataformas se basan generalmente en Web Services y SOAP lo cual presenta incompatibilidades en los GIS y sus servicios vía internet [7]. En particular en Uruguay, la PGE brinda soporte a servicios basados en el estándar SOAP [11] y los servicios geográficos provistos por los organismos se basan en el estándar REST [8].

Para resolver dicha problemática en la tesis de maestría de Raquel Sosa [7] se identifican cinco escenarios de integración de información geográfica en el marco de gobierno electrónico tomando como referencia la arquitectura de Uruguay y se propone la inserción de Componentes de Transformación de Protocolos (CTP). Dichos escenarios van desde consultas de información por parte de público general y público especializado hasta modificación y generación de información geográfica.

1.2 Objetivos

El objetivo principal que plantea el proyecto es diseñar e implementar la arquitectura propuesta para la integración de Web Services Geográficos en Plataformas de Gobierno

Electrónico, tomando en cuenta los escenarios de aplicación propuestos en la tesis de maestría de Raquel Sosa.

Para lograr dicho objetivo se plantean los siguientes puntos:

- Estudiar los escenarios de aplicación para la integración de Web Services Geográficos en Plataformas de Gobierno Electrónico, la arquitectura propuesta y sus mecanismos de transformaciones.
- Diseñar e implementar un prototipo de los mecanismos de transformaciones para cubrir la mayor parte de los escenarios analizados.
- Desarrollar un caso de ejemplo que muestre los mecanismos implementados.

1.3 Resultados esperados

Como primer hito se espera tener una documentación completa con el estudio y diseño de la solución.

También se espera tener un prototipo de la arquitectura para Integración de Web Services Geográficos en Plataformas de Gobierno Electrónico. El mismo contará con los mecanismos de transformaciones implementados para cubrir la mayor cantidad de escenarios de aplicación posibles.

Por otro lado, se espera poder contar con un caso de estudio que permita validar el prototipo implementado.

1.4 Resultados alcanzados

Se logra implementar los componentes de transformación que permiten el acceso a los Web Services Geográficos a través de la Plataforma de Gobierno Electrónico. Dichos componentes dan soporte de lectura y escritura, quedando así implícito el cubrimiento de los escenarios que van desde consultas hasta modificación y generación de información geográfica.

Para validar los componentes de transformación implementados se desarrolla un caso de estudio que permite realizar consultas y modificaciones sobre datos geográficos.

Por último se consigue tener una documentación detallada con análisis, diseño e implementación de la solución.

1.5 Organización del documento

El presente documento se estructura en siete secciones más la sección de referencias.

En la sección dos se hace una breve reseña de los estándares y tecnologías que fueron necesarios conocer y aprender previo al desarrollo del proyecto. En las secciones tres y cuatro se detallan las decisiones propuestas de arquitectura y diseño respectivamente. En la sección cinco se desarrolla un caso de estudio que tiene como objetivo validar las decisiones de arquitectura y diseño planteadas en la sección anterior. En la sección seis se concluye sobre el

todo el proceso de desarrollo del proyecto. Y por último, en la sección siete se proponen trabajos a futuros.

2 Marco de trabajo

A continuación se presenta una breve descripción de los principales estándares mencionados en el documento y que son necesarios para la resolución del problema planteado.

2.1 Web Services Geográficos

Open Geospatial Consortium (OGC) [3] es la organización que propone los estándares de Web Services Geográficos, los cuales están agrupados bajo el nombre OpenGIS Web Services (OWS).

Las siguientes descripciones de los protocolos Web Map Service (WMS) y Web Feature Service (WFS) se arman a partir de la conjunción de [4], [5], [6], y [7].

2.1.1 WMS

WMS [4] es un estándar que define un protocolo para obtener mapas dinámicos, de datos referenciados espacialmente a partir de información geográfica distribuida. El mapa que se obtiene para representar la información geográfica es un archivo de imagen en formato PNG, GIF o JPEG, o en ocasiones como elementos gráficos basados en las especificaciones Scalable Vector Graphics (SVG) o Web Computer Graphics Metafile (WebCGM).

WMS define tres operaciones para publicar datos geográficos:

getCapabilities: Devuelve los metadatos del servicio, tales como, formatos que soporta, datos que posee, información de los valores admitidos de los parámetros, etc.

getMap: Devuelve un mapa geográfico a partir de ciertas capas.

getFeatureInfo: Operación opcional que devuelve información acerca de las características particulares mostradas en el mapa.

2.1.2 WFS

WFS [5] es un estándar que define un protocolo para consultar y modificar información geográfica codificada en Geography Markup Language (GML).

WFS define seis operaciones para consultar y modificar datos geográficos:

getCapabilities: Devuelve información del servicio.

describeFeatureType: Describe la estructura de cualquier tipo de entidad soportado por el servicio.

getFeature: Devuelve los datos geográficos resultado de la consulta realizada por el cliente (analogía con SELECT).

getGmlObject: Devuelve cualquier elemento GML a través de su identificador único.

transaction: Operación que permite modificar las características de los datos geográficos, tales como, darlos de alta, modificarlos y darlos de baja.

lockFeature: Operación opcional que permite bloquear una entidad al momento de realizar una transacción.

En base a las operaciones ofrecidas los servicios WFS se clasifican de la siguiente manera:

WFS Básico: Estos servicios son únicamente de lectura y ofrecen las operaciones GetCapabilities, DescribeFeatureType y GetFeature.

WFS con XLink: Ofrecen las operaciones de un WFS básico más la operación GetGmlObject. Con XLink significa que el servidor puede solicitar datos extra a otros servidores.

WFS Transaccional: Ofrecen las operaciones de un WFS básico más la operación Transaction.

WFS Transaccional con XLink: Ofrecen las operaciones de un servicio WFS Transaccional más la operación GetGmlObject.

2.2 Estándares SOAP

2.2.1 SOAP

SOAP(Simple Object Access Protocol) es un protocolo para la comunicación entre aplicaciones por medio de internet. Es independiente de la plataforma y el lenguaje y define un formato para envío y recepción de mensajes basado en XML [11].

Se publican servicios como RPC (remote call procedure) y la comunicación funciona por medio de un request enviado por el cliente a lo que se devuelve una respuesta del servidor.

Se utiliza el lenguaje WSDL (Web Service Description Language) para describir los servicios que se ofrecen, así como el formato de la entrada y la salida de cada servicio. Un web service SOAP tiene una especificación asociada escrita en este lenguaje que está basado en XML [12].

2.2.2 REST

Una API REST es un tipo de arquitectura de desarrollo web basada en el estándar HTTP y que permite interconectar sistemas hypermedia distribuidos.

A diferencia de SOAP no está orientado a servicios (RPC) sino a recursos. Un recurso es cualquier información que pueda ser almacenada, una entidad que representa un concepto de negocio que a su vez puede ser accedido por otro sistema. Cada recurso posee un identificador único que lo distingue de otros recursos. Algunos recursos son estáticos, o sea, en cualquier momento en que sean examinados luego de su creación siempre les va a corresponder el mismo resultado. Otros cambian a medida que pasa el tiempo [8].

La comunicación es stateless, cada recurso posee un estado interno, este estado no puede ser accedido desde el exterior. Cada pedido hecho del cliente al servidor debe tener toda la información necesaria para que el request pueda ser comprendido. Y no se puede sacar ventaja de tener un contexto guardado para cada sesión. La información de las sesiones son almacenadas enteramente en el cliente. Esto también mejora la visibilidad, confiabilidad y escalabilidad del servicio [8].

Un servicio REST permite las siguientes 4 operaciones sobre un recurso :

get: Para obtener información sobre un recurso.

post: Para crear una nueva instancia del recurso.

put: Para modificar un recurso existente.

delete: Para eliminar un recurso.

Se puede entender como una aplicación CRUD (Create, Read, Update and Delete), pero a diferencia de estas, un servicio REST le puede asignar otro comportamiento a una de estas operaciones estándar.

2.2.3 WS-SECURITY

Es una extensión de los estándares de Web Services SOAP, publicado y mantenido por OASIS, encargado de brindar seguridad extremo a extremo a nivel del mensaje SOAP. Características similares pueden obtenerse utilizando TLS, aunque tiene el problema de que el mensaje no puede utilizarse para ruteo intermedio, debido que TLS encripta todo el mensaje http.

WSS brinda dos servicios básicos, confidencialidad e integridad. Los cuales pueden alcanzarse utilizando diferentes métodos, XML Signature y XML Encryption. Estos se basan e integran con diferentes tecnologías de seguridad, como tokens de seguridad, certificados X.509 y Kerberos, entre otras.

Provee ciertas ventajas sobre la opción de TLS, como por ejemplo se puede cifrar todo o parte del mensaje, lo cual permite utilizar las partes no críticas para ruteo o controles de acceso.

Se integra con otros estándares de seguridad como WS-SecureConversation y WS-Trust.

También soporta la integración con SAML utilizado para proveer SingleSign on a lo largo de internet, permitiendo la interacción entre Proveedores de identidad y proveedores de servicio que confían en los mismos.

2.2.3.1 WS-TRUST

WS-Trust define extensiones a WS-Security que proveen lo siguiente [18]:

- Métodos para emisión, renovación y validación de tokens de seguridad.
- Gestión de relaciones de confianza entre los agentes.

Esta especificación define el modelo Web Services Trust Model. Esto define un proceso que permite restringir a que las peticiones que llegan a un servicio cumplan con una serie de especificaciones. Si quien hace la solicitud no cumple con todos los requerimientos necesarios, entonces debe pedir el acceso a un servidor STS (Security Token Service). Esto último se considera que es una relación de confianza que existe entre el STS y el servicio al que se quiere acceder [19].

Web Services Trust Model define los siguientes mecanismos para verificar relaciones de confianza [18]:

- Fixed trust roots: El destinatario tiene definido un set fijo de relaciones de confianza. Cuando lleguen pedidos va a evaluar si contiene tokens emitidos por alguna de esas relaciones.
- Trust hierarchies: Existe una jerarquía de confianza y el servicio destinatario solo tiene el root de esa jerarquía. Por lo tanto cuando llegue una petición, este deberá verificar que pertenece a la jerarquía de confianza.
- Authentication service: Existe un servicio de autenticación con el cual el destinatario mantiene una relación de confianza. Cuando llega un token, el destinatario lo enviará a este servicio de autenticación, este último determinará si la autenticación es válida o no.

2.2.4 WS-ADDRESSING

WS-Addressing es un estándar para agregar información de direccionamiento en mensajes SOAP. Específicamente define elementos XML para identificar endpoints y asegurar la identificación punto a punto de los mismos en mensajes SOAP [16].

Básicamente se definen dos tipos de elementos que se incluyen en mensajes SOAP:

Endpoint References [16]: Identifican el punto donde son dirigidos los mensajes y especifica las siguientes propiedades:

- wsa:Address: Una dirección (URI) que identifica el endpoint. Puede ser una dirección de red o una dirección lógica.
- wsa:ReferenceProperties: Propiedades que identifican al recurso transportado.
- wsa:ReferenceParameters: Parámetros asociados al endpoint para facilitar la interacción.
- wsa:PortType: Identificador (en formato QName) para el portType del recurso transportado.
- wsa:ServiceName: Es opcional e identifica la descripción del servicio que contiene un WSDL con la descripción del endpoint que se hace referencia.
- wsa:Policy: Especifica una política (policy) que es relevante para la interacción con el endpoint.

Message Information Headers [16]: Cabezales que contienen información sobre la identificación del mensaje. Definen las siguientes propiedades:

- wsa:to : Destino del mensaje.
- wsa:from: Origen del mensaje (emisor).
- wsa:replyTo: Contiene el endpoint al cual dirigir una respuesta.
- wsa:faultTo: Especifica el endpoint al cual se dirigen los fallos.
- wsa:action: Contiene un identificador para las semánticas implicadas en el mensaje. (una URI indicando una entrada, salida o mensaje de error).
- wsa:messageID: Contiene una URI que identifica el mensaje en espacio y tiempo. No debe haber dos mensajes con el mismo identificador.
- wsa:relatesTo: Contiene un par de valores que indican como este mensaje se relaciona con otro mensaje.

2.2.5 MTOM

Es un estándar de la W3C para la transmisión de datos binarios de forma óptima, como lo indica su nombre: Message Transmission Optimization Mechanism [13].

Debido a que XML no provee una forma de incluir datos binarios, previo al surgimiento de MTOM la única forma de enviar o recibir este tipo de datos era transformarlo a Base64.

Base64 es una técnica que transforma datos binarios en una tira de caracteres ASCII de forma reversible. Esta solución que todavía se utiliza para algunos contextos. El problema de base64 es que hace crecer el tamaño de los datos, porque para la transformación convierte 6 bits de información en su correspondiente carácter ASCII de 8 bits, agregando un total de 25% extra de información. A priori puede parecer poco, pero para archivos grandes, de Gigabytes, un 25% es mucho.

MTOM soluciona este problema haciendo uso de los pedidos multi-parte de http. Funciona agregando en el body del mensaje SOAP “punteros” hacia las partes del mensaje http donde vienen los datos binarios sin codificar. De esta manera no agrega el overhead de Base64.

2.3 Tecnologías y productos estudiados.

2.3.1 ESB

Enterprise Service Bus [26] es un modelo de arquitectura de software usado para implementar comunicación entre sistemas orientados a servicios. Generalmente se basa en plataformas que implementan y potencian sistemas de mensajería, definiendo puntos de entrada, principalmente web services SOAP. Además agregan servicios dentro del bus como seguridad y confidencialidad, así como también ruteo de mensajes según condiciones del mensaje o del contexto. También proveen servicios de transformación de datos, para adecuar los mensajes a diferentes sistemas interactuando los cuales agregan o quitan información.

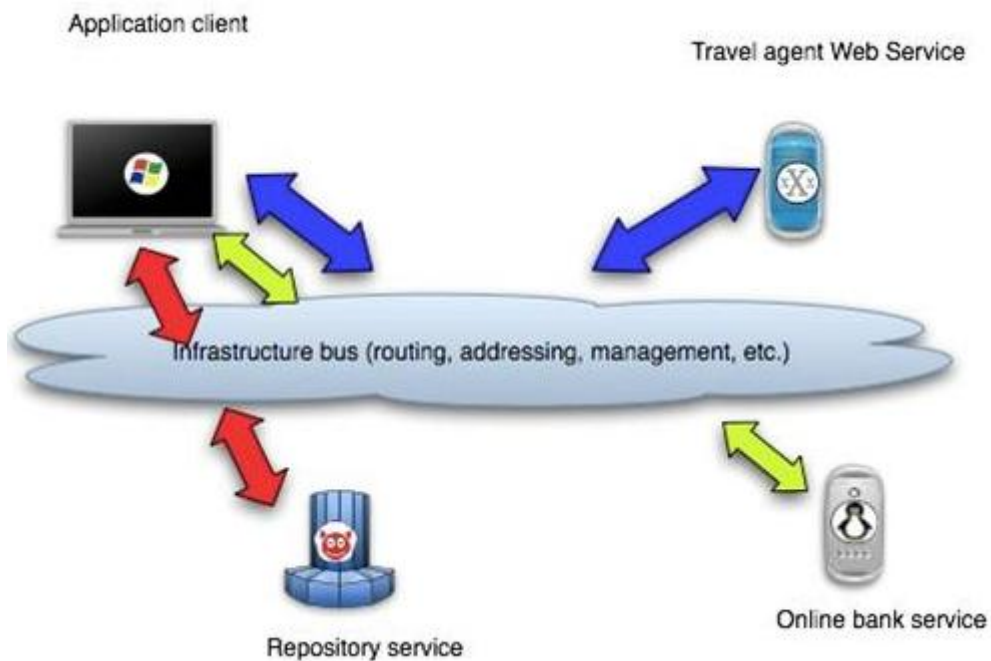


Figura 2.1 – Esquema de ESB¹

2.3.1.1 JBOSS ESB

Jboss Esb [21] es un producto de Red Hat y su comunidad, una implementación de la arquitectura ESB en Java, basada también en el servidor de aplicaciones JBoss de la misma compañía. Se basa en estándares, tanto de Java como de la Web para brindar conectividad y comunicación con muchos otros tipos de tecnologías. En la siguiente figura vemos un esquema de todas las tecnologías de comunicación soportadas por este producto.

¹ Imagen tomada de [21].

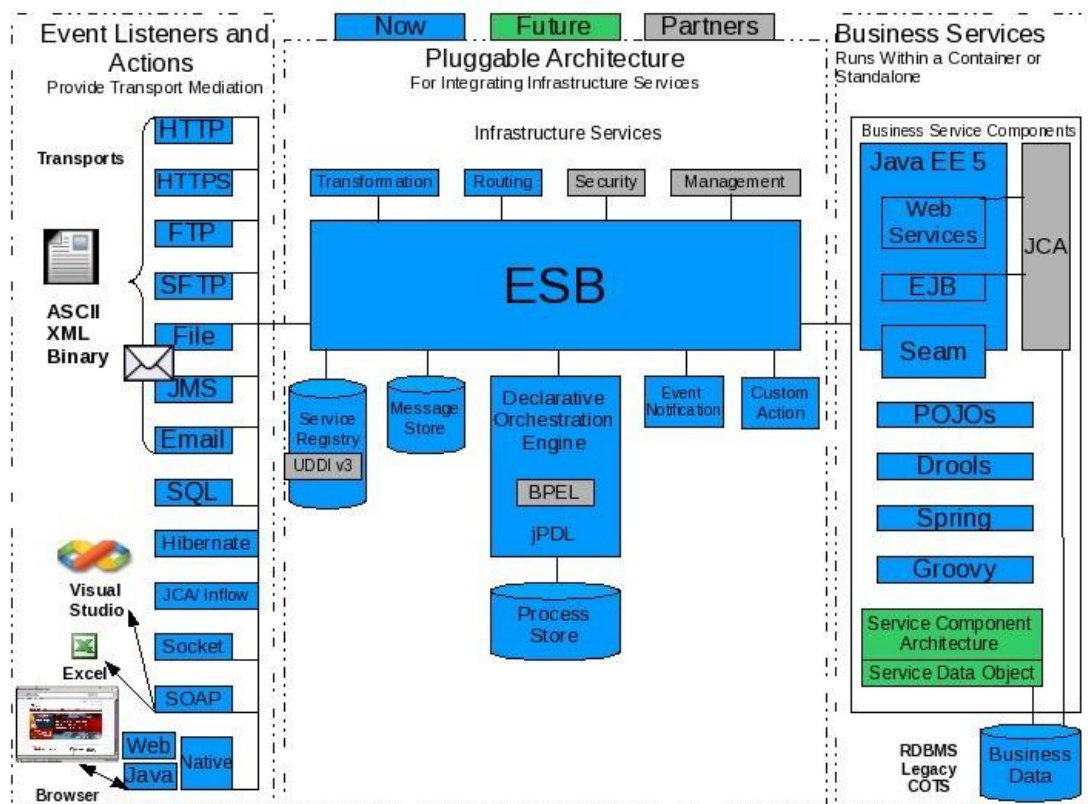


Figura 2.2 – Esquema de ESB²

Básicamente en Jboss ESB un servicio define un conjunto de Listeners de los cuales quiere estar al tanto cuando llegan mensajes, luego el servicio es notificado de que un mensaje ha llegado y define para éste, filtros o procesadores que trabajarán con la información contenida en él. Por último el servicio ruteará el mensaje hacia algún conector de salida para invocar otro servicio fuera del ESB o dará una respuesta a quien inició la comunicación en primera instancia.

2.3.2 Servidores de mapas

2.3.2.1 GEOSERVER

GeoServer es un servidor de mapas de código abierto que permite a los usuarios compartir y editar datos geoespaciales. Está diseñado para la interoperabilidad, publica los datos de cualquier fuente importante de datos espaciales usando estándares abiertos [9].

Es la implementación de referencia de los estándares Open Geospatial Consortium (OGC) Web Feature Service (WFS) y Web Coverage Service (WCS). Soporta WMS, WFS y WFS-Transaccional [9].

Posee las siguientes características [10]:

² Imagen tomada de [21].

- Es el más adecuado para los desarrolladores java ya que ofrece facilidad para interoperar con aplicaciones escritas en esa plataforma.
- Soporta WFS y WFS-T.
- Como servidor soporta una variedad de formatos de almacenamiento (PostGis, Oracle spatial, Mysql, GeoTiff) tanto vectorial como raster.
- Reproyección al vuelo.
- WMS Tiling cache (Usa GeoWebCache como cliente de tiles WMS).

2.3.2.2 MAPSERVER

Es una plataforma de código abierto para publicar información geográfica y aplicaciones interactivas de mapas en la web [17].

El proyecto es administrado y gestionado por el MapServer Project Steering Committee (PSC), el cual está autorizado por OSGeo. Mientras tanto las mejoras y mantenimientos del proyecto son realizadas por usuarios y programadores en todo el mundo [17].

Básicamente MapServer es una aplicación CGI alojada en un servidor web. Cuando llega un pedido a MapServer, se usa información que fue enviada en la URL y el archivo de configuración (Map file, ver abajo) para crear una imagen del mapa solicitado [17].

Una aplicación de MapServer consiste en los siguientes elementos [17].

Map file: Es un archivo de texto que sirve como configuración de una aplicación MapServer. Aquí se define el área del mapa, donde están los datos, las capas del mapa incluyendo datasources, proyecciones y símbolos.

Información geográfica: MapServer puede utilizar varios tipos de datasource. Por defecto se utiliza ESRI Shape Format pero muchos otros formatos son soportados por la herramienta.

Páginas HTML: Son la interface entre el usuario y la aplicación de mapas. MapServer puede ser invocado para colocar un mapa estático en un html. Para que sea interactivo la imagen se debe colocar en un formulario para poder hacer varias invocaciones MapServer y que este devuelva las distintas imágenes.

MapServer CGI: Un archivo ejecutable que recibe pedidos y devuelve imágenes o datos.

Web/HTTP server: Puede ser cualquier servidor web que aloje las paginas html con mapas que son solicitadas por el usuario.

2.4 Plataforma de eGob de Uruguay

El gobierno electrónico o gobierno en red radica en la idea de la construcción de una Administración Pública enfocada en el ciudadano, siempre accesible y más cercana y que hace uso intensivo de las Tecnologías de la Información y las Comunicaciones (TIC) [1].

La plataforma de gobierno electrónico es un facilitador para el desarrollo de servicios y tramites en línea que provee servicios transversales y herramientas comunes a los Organismos del Estado así como también servicios a personas, empresas y organismos. Es un medio que facilita la interoperabilidad y el intercambio de información entre Organismos. Es el contexto tecnológico y legal que permite asegurar que la información intercambiada cumpla con los requisitos legales y tecnológicos predefinidos [1].



Figura 2.3 - Plataforma de Gobierno Electrónico y Otros Servicios³

2.4.1 Plataforma de Interoperabilidad

*"La Plataforma de Interoperabilidad (PDI) forma parte de la Plataforma de Gobierno Electrónico (PGE) de AGESIC y tiene como objetivo general facilitar y promover la implementación de servicios de Gobierno Electrónico en Uruguay. Para esto, la PDI brinda mecanismos que apuntan a simplificar la integración entre los organismos del Estado y a posibilitar un mejor aprovechamiento de sus activos."*⁴

³ Imagen tomada de [1].

⁴ Cita tomada de [2].

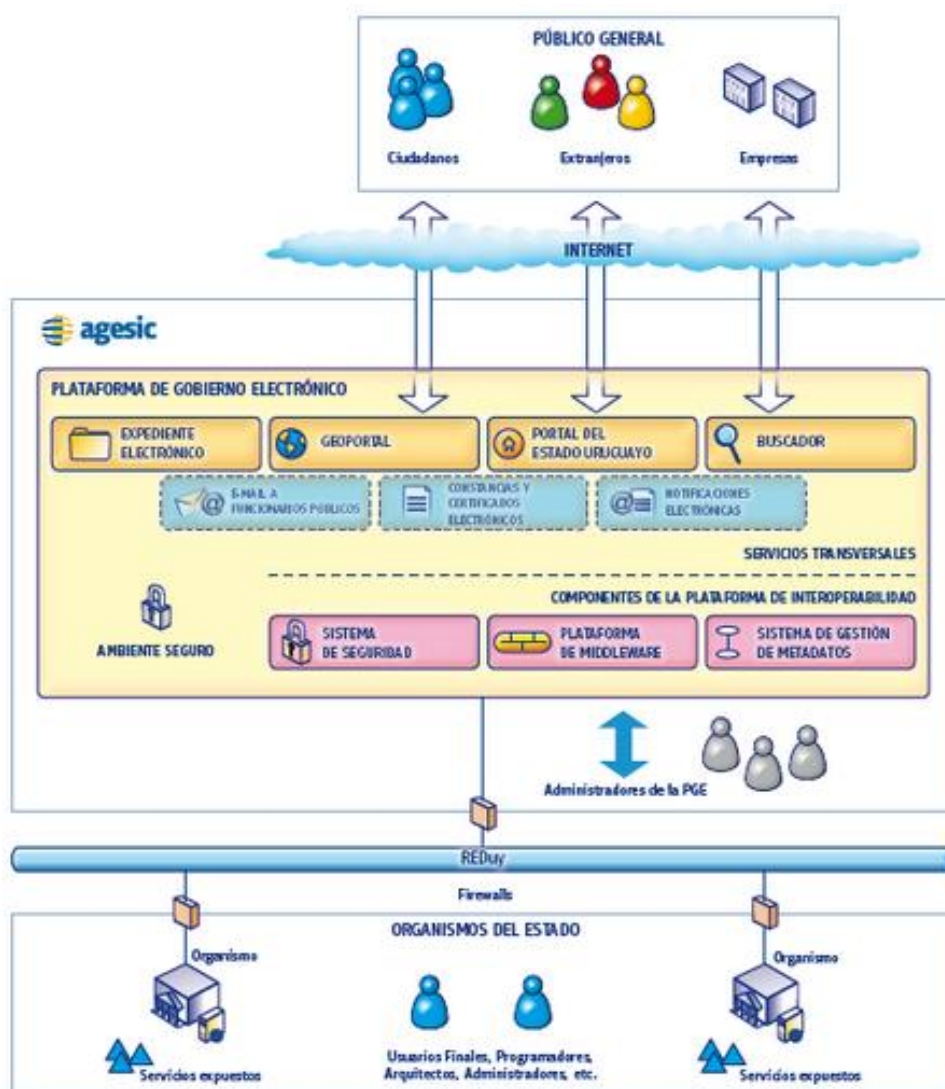


Figura 2.4 - Plataforma de Interoperabilidad ⁵

La PDI está basada en una arquitectura orientada a servicios (SOA) y compuesta por un sistema de control de acceso, un sistema de gestión de metadatos y una plataforma de middleware [2].

Sistema de control de acceso: Es el punto de entrada a la plataforma y provee mecanismos de autenticación y autorización para el consumo de servicios basados en XML [2].

Sistema de gestión de metadatos: Provee especificación de alto nivel de los conceptos relativos a de los servicios públicos [2].

Plataforma de middleware: Cuenta con mecanismos para facilitar el desarrollo, despliegue e integración de servicios y aplicaciones. Está integrado por dos ESB, uno de tecnología Microsoft y otro de tecnología Java, con el fin de obtener lo mejor de ambos y ampliar el espectro de posibilidades en cuanto a los métodos de conexión. Los organismos pueden

⁵ Imagen tomada de [2].

utilizar esta plataforma para publicar y descubrir servicios, así como utilizar las diferentes capacidades de mediación, las cuales permiten desacoplar clientes y servicios [2].

2.4.1.1 Plataforma de Middleware

Uno de los principales componentes de la Plataforma de Interoperabilidad de la PGE es la Plataforma de Middleware [20].

A continuación se describe la plataforma de middleware mediante el siguiente ejemplo:

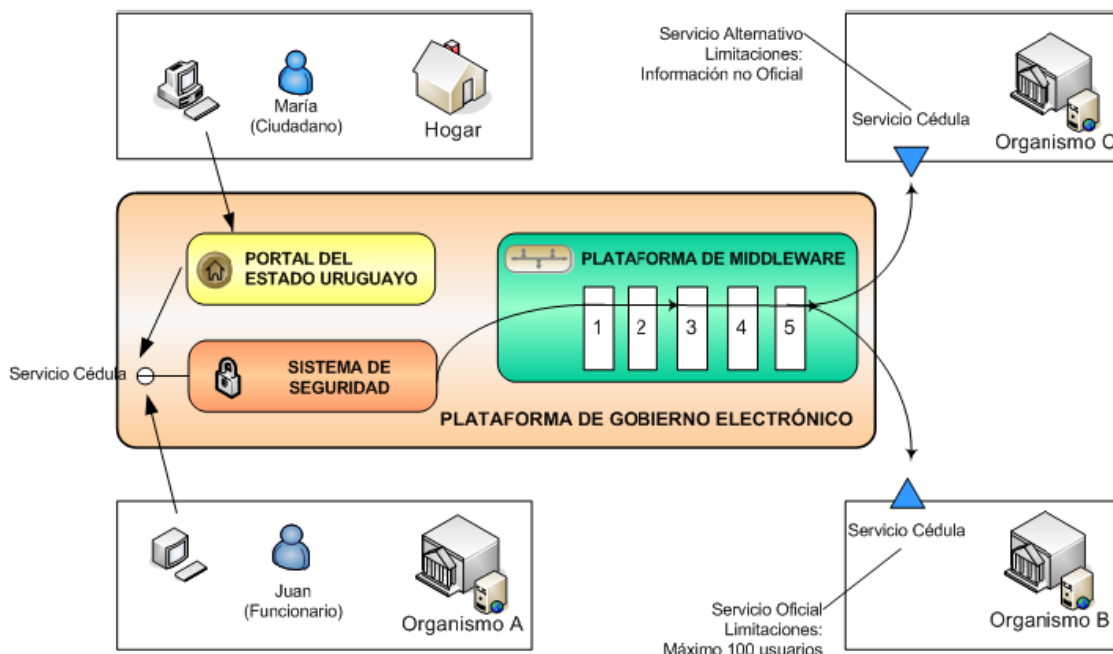


Figura 2.5 - Ejemplo de funcionamiento de la plataforma de middleware⁶

Los mensajes enviados por los usuarios solicitando el Servicio de Cédula (indicado en la figura) cuando llegan a la PGE primeramente pasan por los controles de seguridad y luego son enviados a la plataforma de middleware, la cual realiza las siguientes acciones:

- 1) Verificación sintáctica: Validaciones de integridad tales como verificación de nulos, estructuras de datos incompletas o errores en tipos de datos. En caso de error, el mensaje es rechazado y se notifica al cliente los motivos.
- 2) Verificación de políticas de seguridad: Validaciones para constatar que se cumple con las restricciones de seguridad definidas en la Ley 18.331 de Protección de Datos Personales y acción de Habeas Actas así como también las políticas definidas por la PGE. En caso de error, el mensaje es rechazado y se notifica al cliente los motivos.

⁶ Imagen tomada de [20].

3) Elección del destino del mensaje: En base a la política de direccionamiento de mensajes de la PGE y según ejemplo indicado en la figura se define que *"Siempre se enviará el mensaje al servicio del organismo B si hay menos de 100 pedidos concurrentes pendientes. En caso contrario, se redirigirán los pedidos al servicio del organismo C."*⁷

4) Transformación de datos: Según especificaciones del servicio puede que sea necesario transformar el pedido. En el ejemplo, el servicio del organismo C requiere que la cédula de identidad contenga puntos y dígito verificador.

5) Envío del mensaje al servicio: Se envía el mensaje al servicio destino.

2.4.1.1.1 Componentes de la Plataforma de Middleware



Figura 2.6 - Plataforma de Middleware⁸

Entornos de ejecución: Generalmente las aplicaciones y servicios de la PGE se alojan en los propios organismos, pero para el caso en que los organismos no cuenten con la infraestructura de hardware o software necesaria para sus servicios la plataforma de middleware ofrece entornos de ejecución basados en tecnologías de middleware tales como servidores de aplicaciones, entre otros.

Por otro lado, los entornos de ejecución también se utilizan para servicios, componentes o aplicaciones que brindan funcionalidades comunes o utilitarias. Actualmente en la PGE existe un servicio "Timestamp" provisto por AGESIC, el cual provee la fecha y hora actual.

Las principales plataformas para el desarrollo de aplicaciones empresariales proporcionadas por la plataforma de middleware son: la plataforma .NET de Microsoft y la plataforma Java Enterprise Edition (Java EE). Esta última se provee a través del JBoss Enterprise SOA Platform.

Registros de servicios: Provee funcionalidades para que los organismos publiquen, describan, busquen y descubran servicios en la PGE.

⁷ Cita tomada de [20].

⁸ Imagen tomada de [20].




	Proveedor	Servicio	Categorías	WSDL
		Certificado de Nacidos Vivos	salud	http://[ip-wsdl]/nacidosVivos?wsdl
		Timestamp	general	http://[ip-wsdl]/timeStamp?wsdl

Figura 2.7 - Directorio de Servicios⁹

Actualmente el registro de servicios se maneja de forma interna a AGESIC, pero se planea brindar un registro UDDI para que los organismos puedan buscar y descubrir servicios de acuerdo a distintos criterios.

Productos Enterprise Service Bus: Proveen mecanismos que pueden ser utilizados por los organismos para el consumo y provisión de servicios. La plataforma de middleware cuenta con los siguientes productos de tipo ESB: JBoss ESB y Microsoft Biztalk Server complementado con Biztalk ESB Toolkit.

2.4.1.2 Sistema de Seguridad

El sistema de seguridad es el segundo componente más importante de la Plataforma de Interoperabilidad [20].

A continuación se describe el sistema de seguridad mediante el siguiente ejemplo:

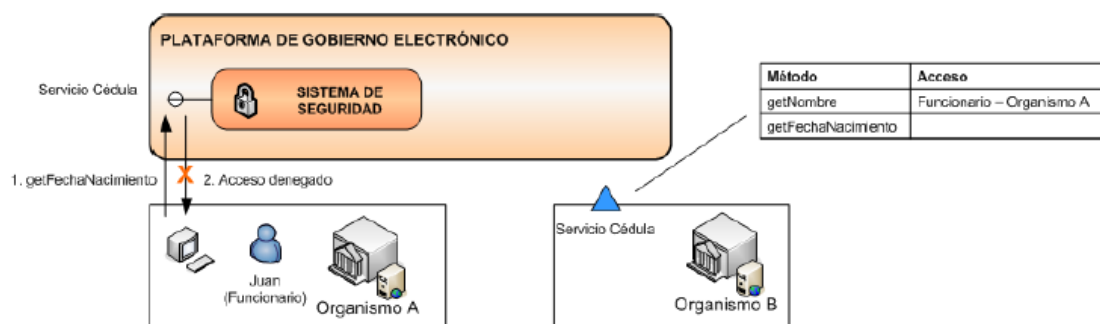


Figura 2.8 - Ejemplo de funcionamiento del Sistema de Seguridad¹⁰

Cuando un organismo publica un servicio, este debe especificar quien tiene acceso a cada método del mismo ya que el control de acceso en la plataforma se realiza a nivel de métodos.

En el ejemplo indicando en la figura el organismo B provee el servicio de Cédula con los métodos getNombre y getFechaNacimiento y brinda a los funcionarios del organismo A acceso únicamente al método getNombre. De esta forma si el organismo A intenta acceder al método getFechaNacimiento, este será rechazado por la PGE. Esto permite que el organismo B se olvide de la seguridad delegando el control de acceso al Sistema de Seguridad de la PGE.

⁹ Imagen tomada de [20].

¹⁰ Imagen tomada de [20].

2.4.1.2.1 Componentes del Sistema de Seguridad

El Sistema de Seguridad se divide en los siguientes tres componentes:

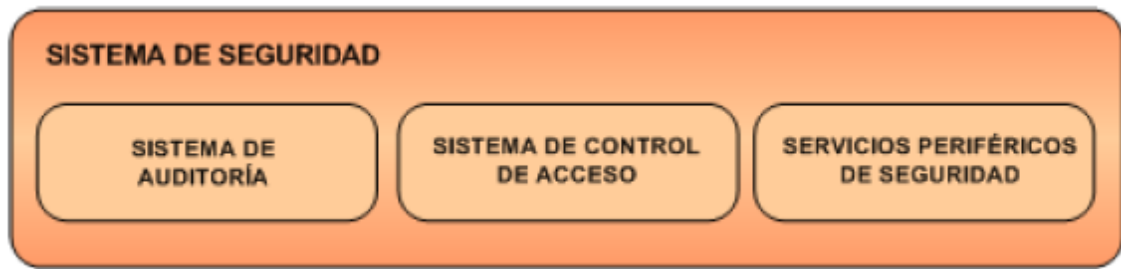


Figura 2.9 - Sistema de Seguridad¹¹

Sistema de Auditoría: Provee las herramientas necesarias para realizar auditorías de seguridad sobre la PGE. Recolecta información y realiza análisis y reportes de auditoría.

Sistema de Control de Acceso: Provee los mecanismos necesarios para aplicar las políticas de control de acceso sobre los servicios publicados y las aplicaciones disponibles en la PGE.

Servicios Periféricos de Seguridad: Provee los mecanismos necesarios para facilitar al organismo el acceso seguro a la PGE.

¹¹ Imagen tomada de [20].

3 Análisis y arquitectura

A continuación se presenta el análisis realizado sobre la tesis de maestría de Raquel Sosa y las decisiones de arquitectura que se tomaron para la resolución del proyecto.

3.1 Escenarios propuestos en la tesis

En la tesis se presentan y analizan cinco escenarios, donde una tecnología como la desarrollada sería útil. A continuación se describen brevemente dichos escenarios.

3.1.1 Público general accediendo a información geográfica

Este escenario está pensado para acceder a información geográfica que sea publicada solamente para consultas ya que es de interés general. Dicha información podrá ser accedida por tanto por organismos como por los ciudadanos o público en general.

Como se trata de información pública, este escenario no requiere mecanismos de seguridad pero si debe asegurar la disponibilidad del servicio.

3.1.2 Público especializado consultando información

En este escenario un organismo brinda un servicio específico para que la información pueda ser consultada por otro organismo. En este caso la información solo está disponible para usuarios especiales pertenecientes al organismo cliente y no para el público en general.

Este escenario tiene como principal requerimiento la disponibilidad del servicio. Ya que esta misma se debe asegurar en fin de que el organismo que consume la información pueda cumplir con sus funciones. Además de la disponibilidad también requiere de mecanismos de seguridad para el control de acceso para asegurar que quienes accedan al servicio sean los organismos clientes autorizados.

En este escenario se requiere realizar acuerdos entre el organismo cliente y el organismo proveedor logrando así un acceso seguro a la información geográfica.

3.1.3 Instituciones colaborando en la creación de información geográfica

Este escenario contempla el caso en que un organismo modifica o genera la información geográfica que es mantenida por otro organismo. Esto se da en organismos que trabajan con un conjunto de datos pero no tienen la responsabilidad de mantener esos datos.

Este escenario requiere un fuerte control de seguridad ya que se debe permitir el acceso solo a funcionarios específicos del organismo colaborador. También es necesario validar la información recibida por parte del organismo que mantiene la misma.

3.1.4 Instituciones colaborando en trámites

En este escenario dos o más organismos colaboran para la realización de trámites públicos. Los organismos participantes deben compartir datos para poder brindar el servicio a los usuarios. Este proceso le evita a un usuario tener que presentarse en los distintos organismos para recoger la información que necesita para hacer el trámite.

Entre los problemas a enfrentar en este escenario está generar una buena coordinación institucional. A su vez el proceso se hace más complejo cuando en un trámite participan varios organismos.

3.1.5 Público generando información geográfica

En este escenario un usuario público puede aportar información geográfica a un organismo.

Entre los desafíos que plantea este escenario está el de brindar mecanismos amigables para que los usuarios puedan enviar la información fácilmente. También es importante auditar los datos que son enviados de manera de filtrar solo los datos confiables.

3.1.6 Resumen análisis de escenarios

En el siguiente cuadro se resumen las características de los escenarios descriptos.

Item	Escenario I	Escenario II	Escenario III	Escenario IV	Escenario V
Plataforma	Localización	Localización, seguridad	Localización, seguridad	Localización, seguridad, orquestación	seguridad
Protocolos	GIS: WMS, WFS, CSW PGE: ninguno	GIS: WMS, WFS, CSW PGE: WS-Security, WS-Trust	GIS: WMS, WFS-T, CSW PGE: WS-Security, WS-Trust	GIS: WMS, WFS-T, CSW PGE: WS-Security, WS-Trust, WS-Coordination	GIS: WMS, WFS-T, CSW PGE: WS-Security, WS-Trust
Operaciones	Solo Lectura	Solo Lectura	Lectura, escritura	Lectura, escritura	Lectura, escritura
Actores	Público en General, Funcionarios	Funcionarios de organismo cliente	Funcionarios de organismos	Funcionarios de organismos	Public en general
Seguridad	No requiere	Control de Acceso	Control de Acceso	Control de Acceso	Control de Acceso
Software	Cliente web y cliente GIS desktop	cliente GIS desktop adaptado	Cliente GIS desktop adaptado	Aplicaciones de negocio	Software Comunidad Web
Coordinación Institucional	Solo acuerdo de uso de la PGE	Acuerdo cliente-proveedor	Acuerdos con organismos proveedores	Acuerdos entre organismos	Herramientas de la comunidad

Figura 3.1 – Cuadro resumen de características de los escenarios¹²

¹² Imagen tomada de [7].

3.2 Solución propuesta en la tesis

Para la propuesta de la solución se recopilaban las características comunes a todos los escenarios. Teniendo en cuenta los escenarios II, III y IV ya que el resto de los escenarios quedan cubiertos implícitamente.

Se encuentran como puntos críticos el ruteo de los pedidos y el manejo de asincronismo y seguridad. En la PGE se utiliza ws-addressing para el ruteo de pedidos y asincronismo y la seguridad se resuelve mediante mecanismos de ws-security.

La estrategia se basa en utilizar transformaciones sobre los pedidos de Web Services Geográficos para poder integrar los mismos con la PGE. Primero se transforman los pedidos en paquetes SOAP y se les agrega información para poder utilizar los mecanismos de ruteo y seguridad de la PGE. Luego, cuando la PGE debe enviar el pedido al servidor de mapas, este será transformado nuevamente a un formato REST.

Para esto se define un componente llamado Componente de Transformación de Protocolos (CTP). El mismo se encargará de realizar todas las transformaciones sobre los mensajes para poder lograr que las llamadas WS-GIS utilicen la PGE.

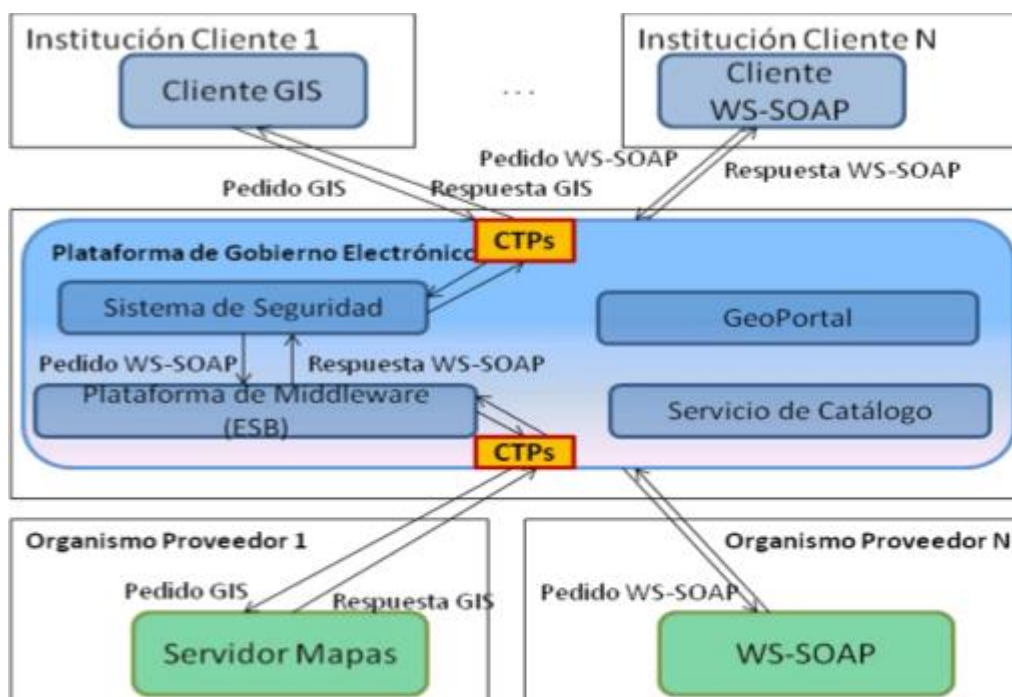


Figura 3.2 – Arquitectura del componente de transformación de protocolos¹³

Los pedidos pueden viajar dentro de la PGE mediante flujos, los cuales usan los mecanismos habituales de la PGE para tratar los pedidos WS-SOAP. Cada escenario seguirá flujos distintos ya que los requerimientos de seguridad o de ruteo de cada uno son distintos. Dependiendo

¹³ Imagen tomada de [7].

del flujo también se usarán diferentes mecanismos de transformación. Por esto último es que se definen distintas transformaciones de acuerdo a aspectos particulares de cada flujo.

A continuación se definen las transformaciones asociadas a cada flujo [7].

Flujos de ruteo:

Transformación	Flujo relacionado	Función
T1: Req WS-GIS \Rightarrow SOAP-Req	Flujo de Ruteo	Genera un paquete SOAP a partir de un pedido WS-GIS.
T1inv: SOAP-Req \Rightarrow Req WS-GIS	Flujo de Ruteo	Arma el pedido WS-GIS original a partir de un paquete SOAP.
T2: respWS-GIS \Rightarrow SOAP-Resp	Flujo de Ruteo	Empaqueta en un paquete SOAP con una respuesta del WS-GIS.
T2inv: SOAP \Rightarrow respWS-GIS	Flujo de Ruteo	Desempaqueta la respuesta del WS-GIS en un paquete SOAP.

Figura 3.3 – Transformaciones para flujos de ruteo¹⁴

Flujos de asincronismo:

Transformación	Flujo relacionado	Función
T3: SOAP-Req \Rightarrow T3SOAP-Req	Flujos con Asincronismo	Toma un paquete SOAP y le agrega información para soportar invocaciones asincrónicas. Se utiliza compuesta con T1.
T3inv: T3SOAP-Req \Rightarrow SOAP-Req	Flujos con Asincronismo	Transforma un paquete SOAP guardando en el contexto la información para el retorno de la respuesta. Se utiliza compuesta con T1inv.
T4: SOAP-Resp \Rightarrow T4SOAP-Resp	Flujos con Asincronismo	Toma un paquete de respuesta SOAP y le agrega información de direccionamiento asincrónico. Se utiliza compuesta con T2.
T4inv: T4SOAP-Resp \Rightarrow SOAP-Resp	Flujos con Asincronismo	Transforma un paquete SOAP en otro quitando la información de direccionamiento. Se utiliza compuesta con T2inv.

Figura 3.4 – Transformaciones para flujos de asincronismo¹⁵

¹⁴ Imagen tomada de [7].

¹⁵ Imagen tomada de [7].

Flujos de seguridad:

Transformación	Flujo relacionado	Función
T5: SOAP-Req \Rightarrow T5SOAP-Req	Flujos con Seguridad - Sesión	Transforma un paquete SOAP en otro agregando la información del usuario autenticado. Se utiliza compuesta con T1.
T5inv: T5SOAP-Req \Rightarrow SOAP-Req	Flujos con Seguridad - Sesión	Transforma un paquete SOAP en otro retirando la información del usuario autenticado. Se utiliza compuesta con T1inv.
T6: SOAP-Resp \Rightarrow T6SOAP-Resp	Flujos con Seguridad - Sesión	Transforma un paquete SOAP en otro agregando información de sesión. Se utiliza compuesta con T2.
T6inv: T6SOAP-Resp \Rightarrow SOAP-Resp	Flujos con Seguridad - Sesión	Transforma un paquete SOAP en otro retirando la información de sesión. Se utiliza compuesta con T2inv.
T7: SOAP-Req \Rightarrow T7SOAP-Req	Flujos con Seguridad - Autorización	Transforma un paquete SOAP en otro agregando la información del perfil de usuario. Se utiliza compuesta con T1.
T7inv: T7SOAP-Req \Rightarrow SOAP-Req	Flujos con Seguridad - Autorización	Transforma un paquete SOAP en otro retirando la información del perfil de usuario. Se utiliza compuesta con T1inv.
T8: SOAP-Resp \Rightarrow T8SOAP-Resp	Flujos con Seguridad - Autorización	Transforma un paquete SOAP en otro agregando la información del perfil de usuario en respuesta. Se utiliza compuesta con T2.
T8inv: T8SOAP-Resp \Rightarrow SOAP-Resp	Flujos con Seguridad - Autorización	Transforma un paquete SOAP en otro retirando la información del perfil de usuario en la respuesta. Se utiliza compuesta con T2inv.

Figura 3.5 – Transformaciones para flujos de seguridad¹⁶

3.3 Casos de uso

En esta sección se da una descripción de los casos de uso más críticos, elegidos especialmente porque obligan a tomar definiciones arquitectónicas específicas para poder soportar dichas interacciones en la plataforma. También se presenta una definición de los actores que interactúan en los mencionados casos de uso.

¹⁶ Imagen tomada de [7].

3.3.1 Actores

Organismo proveedor: Es una entidad que contiene datos geográficos y ofrece servicios tanto de consulta como de actualización.

Organismo cliente: Es una entidad que consulta y/o actualiza datos geográficos de un organismo proveedor.

Público general: Es una persona con un perfil público que quiere actualizar o consultar información geográfica.

STS: Encargado de la autenticación. Provee tokens de seguridad que luego la PGE se encarga de verificar.

PGE: Es el componente que se encarga de procesar todos los pedidos y controlar que sean correctos así como también maneja los mecanismos de seguridad y de balanceo de carga.

3.3.2 CU: Público general consultando información.

En este escenario se debe permitir a un usuario público consultar la información geográfica ofrecida por otro organismo.

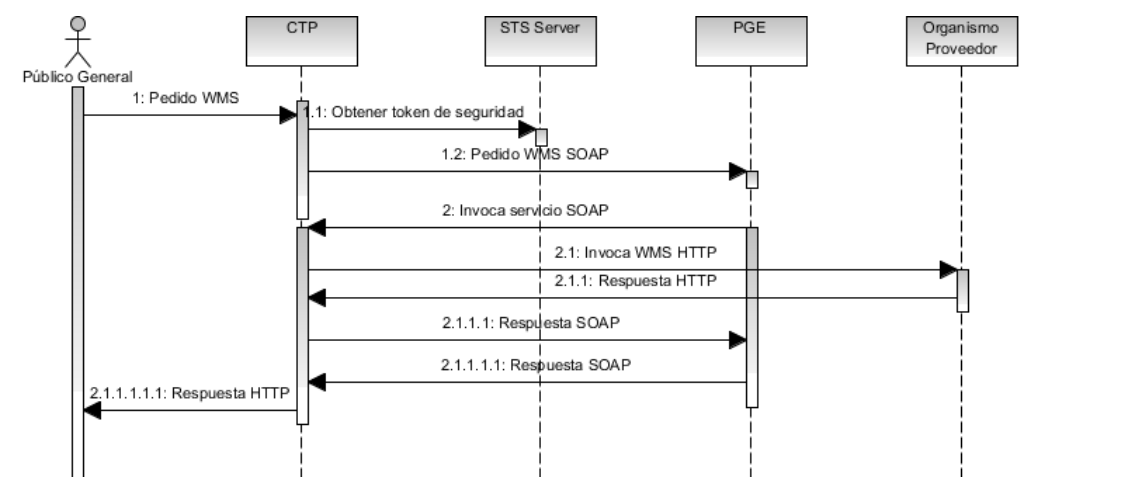


Figura 3.6 – Diagrama de secuencia para el caso de uso de consulta pública

3.3.3 CU: Instituciones colaborando en trámites

Este caso se da cuando un organismo solicita información de otro organismo para realizar trámites solicitados por un usuario público.

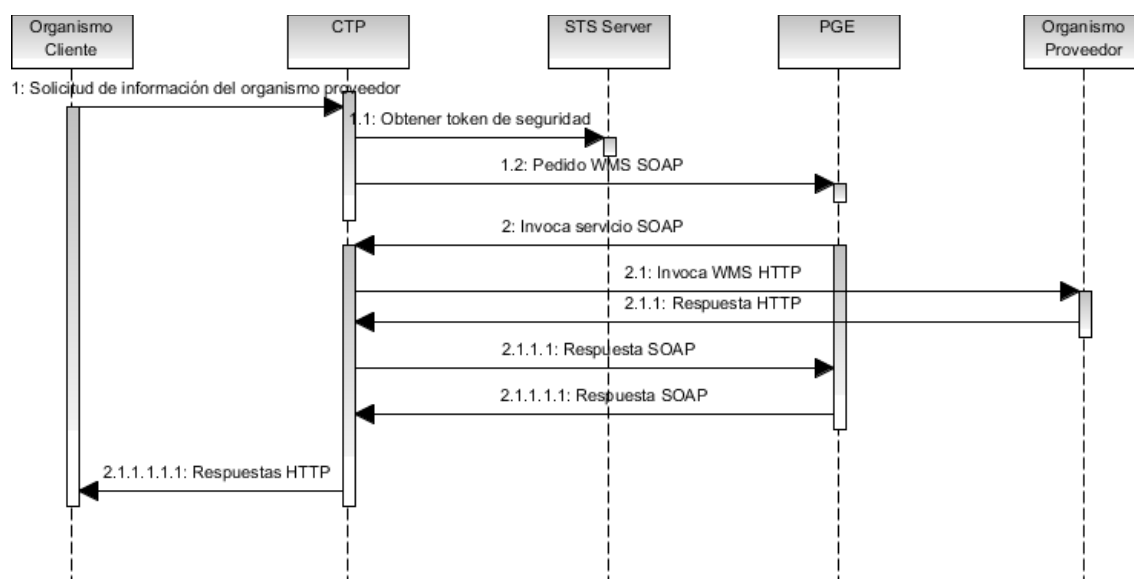


Figura 3.7 – Diagrama de secuencia para el caso de uso de instituciones colaborando en trámites

3.4 Decisiones de arquitectura

Esta sección describe algunas de las situaciones y problemas que fomentaron ciertas decisiones arquitectónicas para resolverlos Durante el desarrollo del proyecto.

3.4.1 Mapeo de las direcciones del servicio

Uno de los grandes problemas enfrentados en el proyecto, el cual obligó a tomar ciertas decisiones arquitectónicas, fue resolver el mapeo de direcciones lógicas y físicas relacionadas a cualquier servicio expuesto en la PGE.

Por defecto existen tres direcciones para cualquier servicio en la PGE las cuales se detallan a continuación.

- **Dirección lógica del servicio:** es una cadena de caracteres en formato URL que identifica al servicio en la PGE, o sea es único.
- **Dirección física del servicio:** está dado por la URL, incluyendo IP y puerto del servidor donde está alojado físicamente el servicio. Esta dirección es por lo general interna a la REDuy y solo conocida por la PGE.
- **Dirección física del proxy del servicio:** URL del proxy por donde los clientes invocan el servicio en la PGE, esta dirección es la publicada.

A las direcciones ya mencionadas se agregan dos más:

- **Dirección física del CTP de entrada:** URL donde el cliente GIS hace la invocación REST para pedir la información geográfica.
- **Dirección física del CTP de salida:** URL donde la PGE reenvía el pedido SOAP, el cual es transformado a REST y luego enviado al servidor de mapas. Esta dirección en este caso

viene a sustituir para la PGE la dirección física del servicio, pero esta no deja de existir porque es la dirección del servidor de mapas y debe ser conocida por el CTP de salida.

Dadas las pautas el problema es el siguiente. El cliente GIS debe conocer entonces solo la dirección del CTP de entrada, hasta acá no cambia nada en cuanto a la arquitectura. El CTP de entrada debe conocer la dirección física del proxy del servicio y la dirección lógica también, para poder hacer las peticiones a la PGE, esto implica configuración en el CTP. Pero luego se tiene para la salida de la PGE dos direcciones físicas, la del CTP salida y la del servidor de mapas. Esto no es soportado por la PGE, cuya relación entre dirección lógica y física es uno a uno para cualquier servicio.

Hay varias formas de solucionar este problema, una de ellas es modificar la PGE, lo cual fue descartado por las premisas del proyecto. Las otras opciones son diseñar una arquitectura acorde, esto último es explicado a continuación con una breve descripción de los pasos dados en el modelado para llegar a la solución final, la cual es descrita en profundidad en la sección 4.

3.4.2 Opciones de despliegue

En la etapa de diseño se fueron manejando diferentes opciones para el despliegue de los componentes. Varias de estas opciones fueron descartadas por los motivos expuestos más adelante. Otras fueron cambiando hasta llegar a la opción elegida, la cual parece ser la que más se adapta a soportar la solución propuesta y tiene buenos niveles de adaptabilidad y baja complejidad de implantación.

3.4.2.1 CTP monolítico

En cierto momento se manejó la posibilidad de construir un CTP único totalmente externo a la PGE, que se encargara de recibir los pedidos de los clientes, encaminarlos a la PGE y esta luego de procesarlos los devolvería al CTP que haría el pedido al servidor de mapas correspondiente.

Como ventaja tiene que es un componente centralizado y manejable por un solo ente público, además acepta toda la responsabilidad de disponibilidad de los servicios deslindando de esta a los proveedores con infraestructuras débiles.

Las grandes contras por las que fue descartado es que se convierte en un único punto de falla susceptible a ataques a todos los servicios geográficos. Además necesita una infraestructura importante para no convertirse en un cuello de botella para la performance. Todo esto sumado a que la complejidad que alcanza la configuración de la plataforma la hace costosa de extender y mantener.

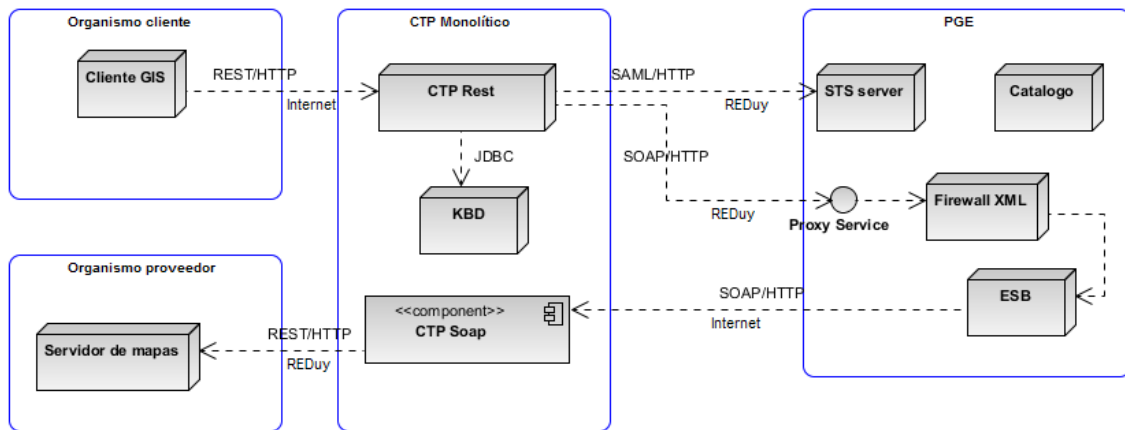


Figura 3.8 – Diagrama de arquitectura utilizando CTP Monolítico

3.4.2.2 CTPs externos

Surge la necesidad de separar las tareas de los traductores, naciendo así el CTP RestConnector, el cual recibe los pedidos de los clientes y los encamina a la PGE. Y el CTP SoapConnector que enmascara el servidor de mapas recibiendo los pedidos SOAP provenientes de la PGE y los transforma a un pedido REST nuevamente.

No hay restricciones donde podría estar alojado el CTP RestConnector (RC), lo que elimina el único punto de falla, ya que podría haber varias instancias ejecutando al mismo tiempo, especialmente en los organismos clientes. Aún así puede convertirse en un cuello de botella y la carga de configuración de todos los servicios GIS sigue siendo grande.

Los CTP SoapConnector (SC) se alojan en los proveedores al igual que cualquier otro servicio ya expuesto a través de la PGE. Acá surge un problema interesante con el manejo de las direcciones lógicas y físicas de los servicios en cuestión que es explicado anteriormente en el punto 3.4.1.

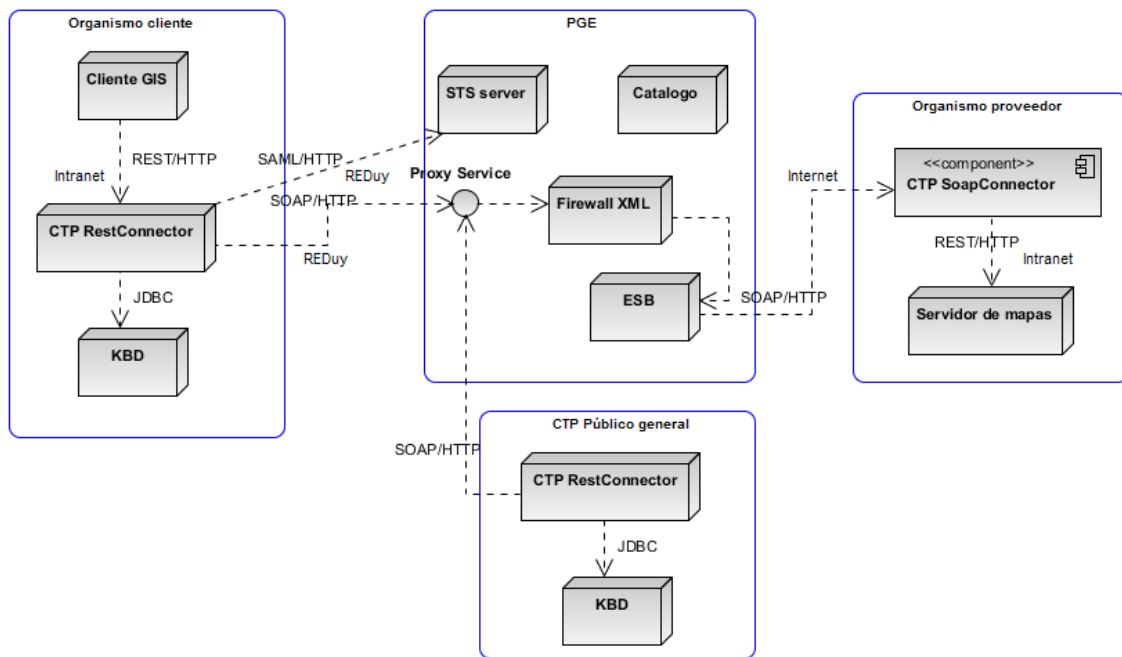


Figura 3.9 – Diagrama de arquitectura utilizando CTPs Externos

3.4.2.3 CTPs dentro de la PGE

Por motivo de dotar a los CTP de una infraestructura poderosa, eliminando así los problemas de cuello de botella y problemas de seguridad, también se exploró la opción que los CTP estuvieran dentro de la PGE. Esto implicaría modificar la PGE para incluir un punto de entrada del tipo REST. Lo cual es descartado porque la premisa del proyecto es no modificar y adaptarse a las soluciones existentes, para facilitar la rápida adopción. Eso incluye los clientes GIS, los servidores de mapas y la PGE.

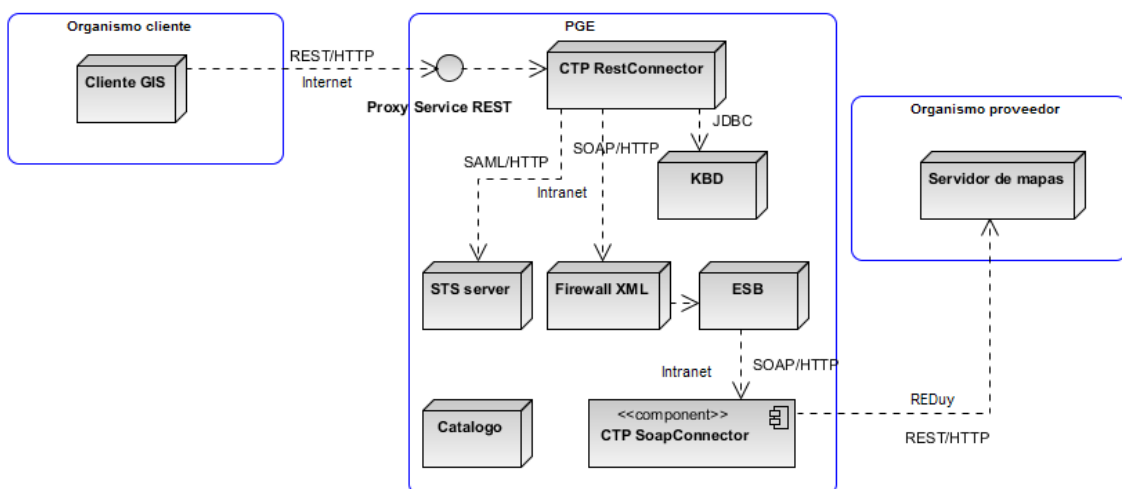


Figura 3.10 – Diagrama de arquitectura utilizando CTPs dentro de la PGE

3.4.2.4 CTPs en distribución ampliada

La arquitectura finalmente propuesta toma los criterios antes planteados, minimiza los problemas encontrados y reduce los riesgos ya mencionados. Hace uso extensivo de los recursos disponibles en la PGE.

Primero se plantea instanciar un servidor CTP SC por cada servidor de mapas a proteger en la PGE. Esto elimina uno de los mapeos de direcciones del servicio, lo que evita modificar la PGE. Además estos CTP SC estarán hospedados en los Execution Environment provistos por la PGE, lo cual elimina el problema de infraestructuras pobres en los entes.

Segundo, cada organismo que quiera consumir servicios geográficos deberá hostear su propio CTP RC, configurando solo los servicios que requiera para sus tareas. Permite reducir la configuración a su mínima expresión y responsabiliza al organismo cliente de los permisos que da sobre los servicios.

Estos CTP RC no deben estar accesibles por el público general, solo personal autorizado dentro del organismo.

Por último para aquellos organismos proveedores, que deseen exponer información pública, deberán hostear su propio CTP RC, el cual contendrá configurado solo los servicios propios.

Esta solución permite además que si un CTP RC de información pública es atacado, el acceso a esa información sigue estando disponible para los organismos clientes ya que acceden desde sus propios CTP RC.

Esta opción de despliegue plantea dos escenarios que pueden convivir para un mismo servicio GIS. Según el nivel de servicio que se quiera brindar se pueden usar una u otra de las siguientes arquitecturas o ambas.

3.4.2.4.1 Escenario: servicio expuesto a organismos clientes

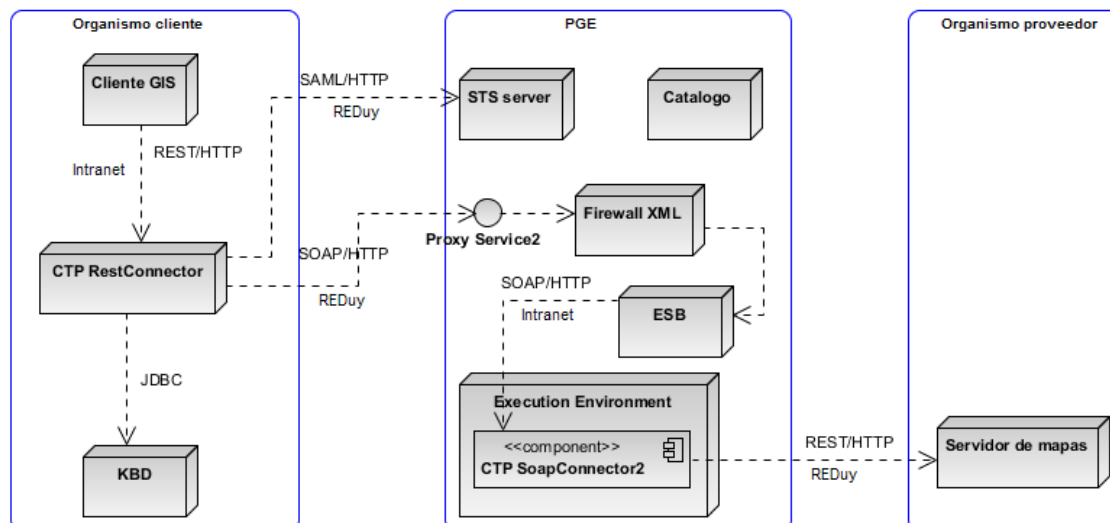


Figura 3.11 – Diagrama de escenario, servicio expuesto a organismos clientes

3.4.2.4.2 Escenario: servicio expuesto a público general

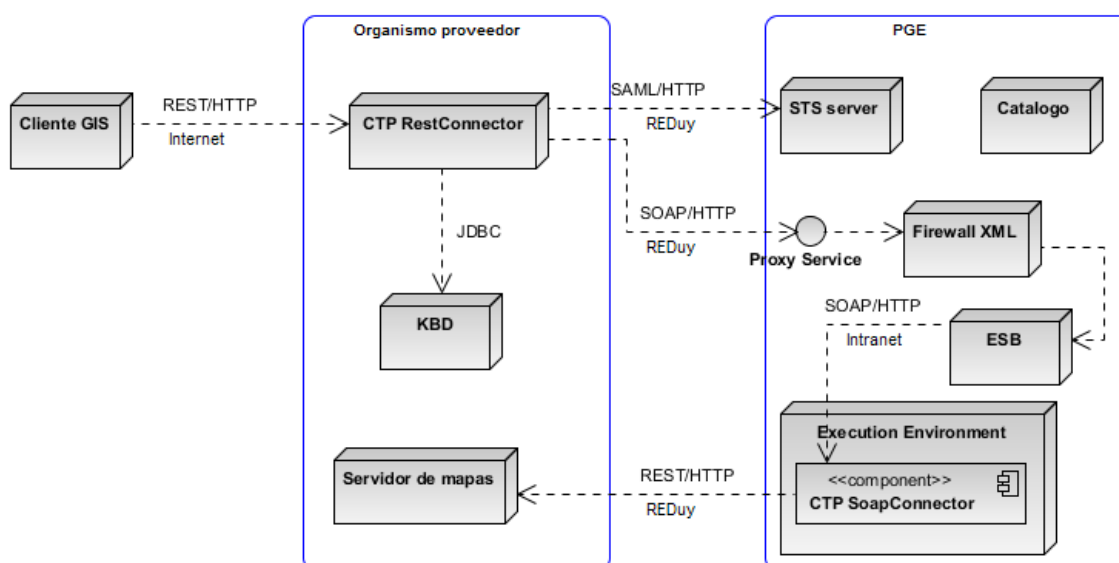


Figura 3.12 – Diagrama de escenario, servicio expuesto a público general

Para una descripción detallada ver documento anexo de arquitectura.

4 Diseño e implementación

A continuación se presentan las decisiones de diseño e implementación que se tomaron para la resolución del proyecto.

4.1 Arquitectura

Tomando en cuenta la solución propuesta por Raquel Sosa en su tesis de maestría [7] se tiene un conjunto de 5 subsistemas independientes que interactúan para resolver los casos de uso. Con respecto a la figura 6, de izquierda a derecha se tiene: El cliente GIS, el CTP de entrada llamado RestConnector, la PGE propiamente dicha, el CTP de salida llamado SoapConnector y el servidor de mapas que se quiere enmascarar.



Figura 4.1 – Diagrama de arquitectura

Cliente GIS: Cualquier programa utilizado para consumir datos geográficos utilizando protocolos WMS y WFS. Puede ser desde una aplicación web usando OpenLayers[28] hasta Gvsig.

CTP RestConnector: es un sub sistema encargado de recibir pedidos REST especificados según los protocolos WMS y WFS. Transforma estos pedidos a formato SOAP para que atraviesen la PGE. Y los encamina hacia ella. También según la configuración agregará información a los pedidos que requieran seguridad y otros datos exigidos por la PGE.

PGE: Es la plataforma de gobierno electrónico del estado uruguayo [1].

CTP SoapConnector: Este sistema estará registrado en la PGE como un proveedor de servicios, al cual se encaminarán los pedidos WMS y WFS previamente transformados por el CTP RestConnector. Su función es volver a componer el pedido REST para enviarlo al servidor de mapas configurado, y transformar la respuesta del mismo en un mensaje SOAP para que recorra el camino inverso. La idea es que exista un SoapConnector por cada servicio GIS expuesto a través de la PGE.

Servidor de mapas: Es un servidor que soporte los protocolos WMS y WFS. En principio habrá soporte para GeoServer [5] y MapServer [6].

4.1.1 Diagramas de actividad

Los siguientes diagramas de actividad ilustran el flujo de datos de los

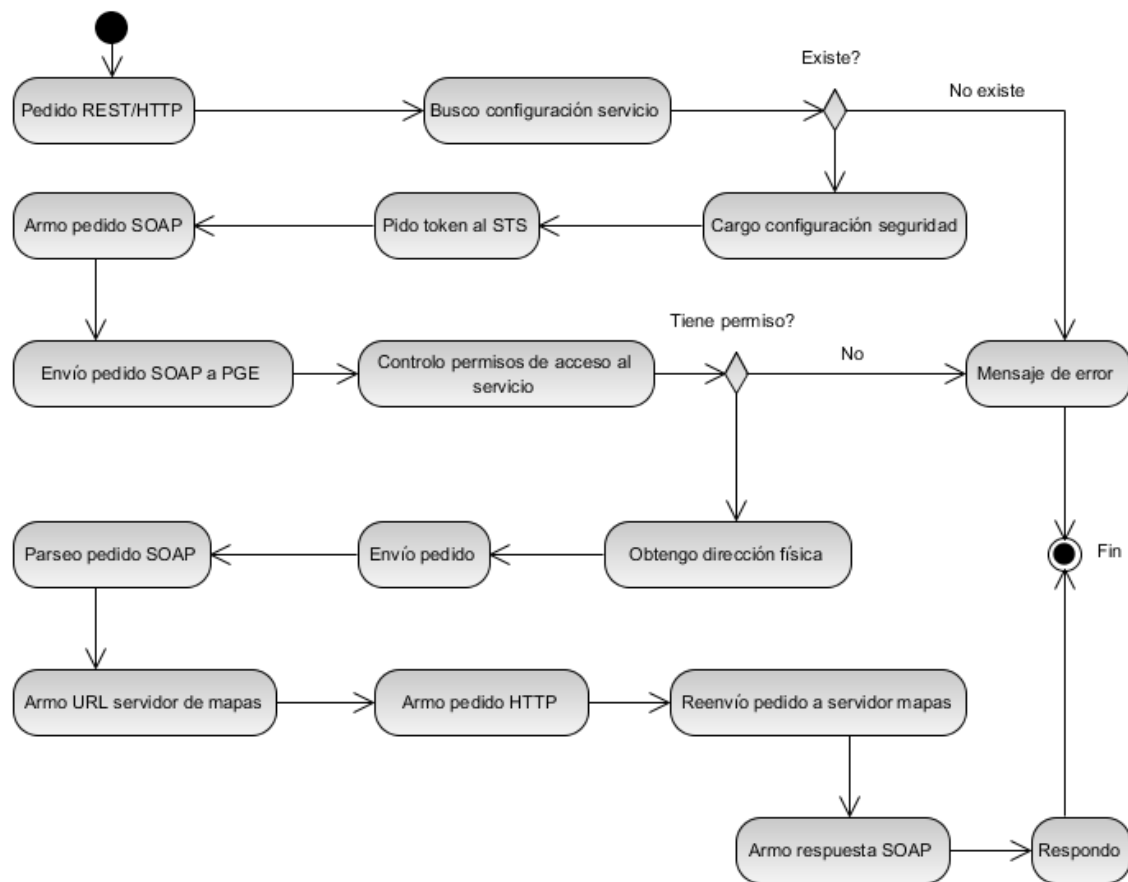


Figura 4.2 – Diagrama de actividad

El flujo de datos a la inversa es automático debido a las tecnologías utilizadas que una vez terminada la invocación arman la respuesta para devolver al cliente.

4.2 Implementación, diagrama de componentes, productos utilizados

En esta sección se describe cada uno de los subsistemas, debido a que son el objeto de ese proyecto se describen los CTPs y como se utiliza una versión reducida y minimalista de la PGE implementada específicamente para simular la PGE real en el contexto de este proyecto también se incluirán sus detalles en el siguiente apartado.

4.2.1 CTP RestConnector

Diagrama de Componentes.

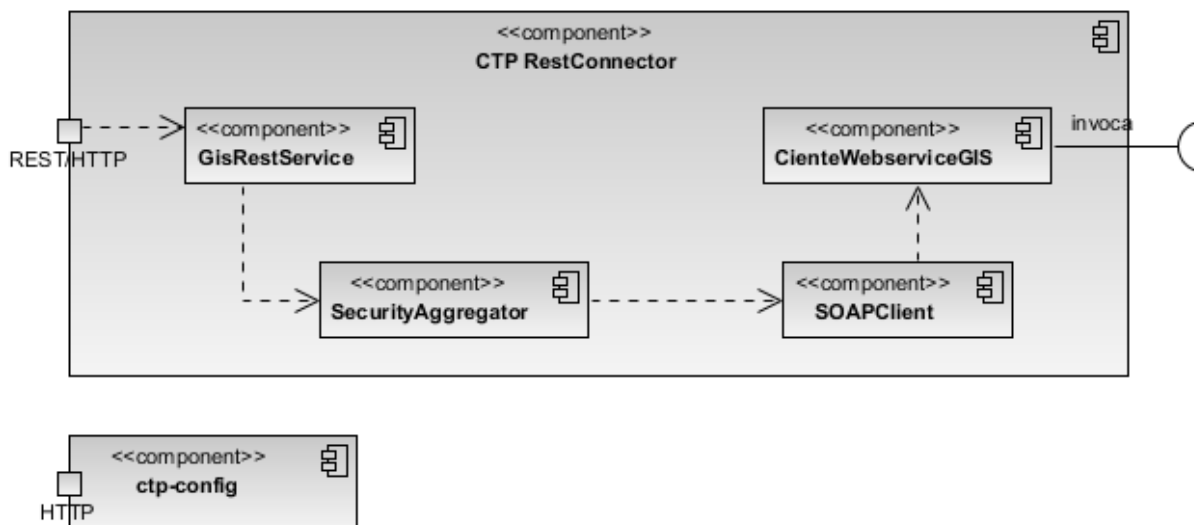


Figura 4.3 – Diagrama de componentes

Descripción.

El RestConnector está compuesto de tres filtros con responsabilidades bien separadas que van agregando información al mensaje ESB para finalmente transformar el pedido de REST en un mensaje SOAP y un cliente de web service SOAP generado en algún framework de web services [7] que lo envía hacia la PGE.

El punto de entrada es un Getway HTTP cuya implementación es provista por la plataforma ESB. Este recibe las conexiones HTTP en una URI particular del servidor y lo transforma en un mensaje ESB que se transmite al primer filtro y se va pasando entre estos procesándolo como sea debido.

El primer filtro es el GisRestService, este se encarga de parsear el pedido WMS o WFS para averiguar qué servicio geográfico expuesto a través de la PGE se quiere invocar. Para esto toma datos del pedido y consulta una base de conocimiento. Carga toda la información necesaria y delega al siguiente filtro.

El SecurityAggregator es el encargado de resolver las necesidades de autenticación y autorización que el servicio de la PGE requiera. Con la información proporcionada por el filtro anterior conformará el token de seguridad SAML que se requiere para atravesar la PGE.

Finalmente el SOAPClient utiliza un cliente de webservice generado a partir del webservice SOAP que provee el CTP SOAPConnector, para invocar el servicio geográfico con los datos necesarios incluyendo los parámetros del pedido original y los agregados por la PGE. Con la diferencia de que el endpoint invocado no es el propio SoapConnctor sino el proxy del servicio expuesto por la PGE.

Al recibir la respuesta se coloca en el mensaje ESB lo enviado por el proveedor y el servidor ESB se encarga automáticamente de convertir eso en una respuesta HTTP adecuada, ya que se ha terminado de invocar los filtros configurados para ese servicio.

Por otro lado se proyecta una aplicación web llamada ctp-config que será implementada en Grails. La funcionalidad de esta aplicación es brindar una interfaz de usuario agradable para configurar los diferentes servicios que pueden ser accedidos desde esa instancia del CTP y la seguridad que cada uno requiera. La información se guarda en una base de datos que será accedida tanto por el CTP como por la aplicación de configuración.

4.2.2 CTP SoapConnector

Diagrama de Componentes.

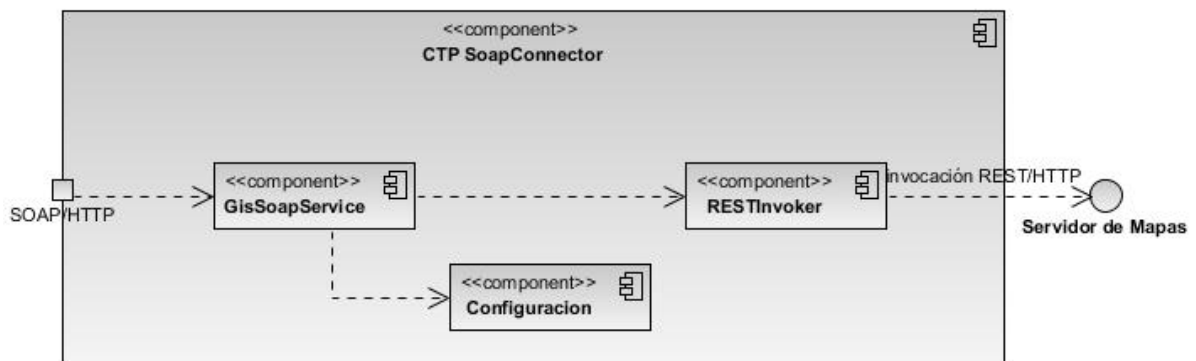


Figura 4.4 – Diagrama de componentes

Descripción.

El SoapConnector también está implementado en la tecnología ESB pero es bastante diferente al anterior. En este caso se configura un servicio cuyo punto de acceso en un procesador SOAP provisto por el ESB, el mismo recibe pedidos HTTP que contengan un mensaje SOAP, lo procesa e invoca a la clase que lo implementa, en este caso el componente GisSoapService, el cual tiene declarados como métodos todos los definidos por los estándares WMS y WFS. Este componente recibe toda la información y utiliza la configuración para armar la URL del servidor de mapas configurado. Dado que se utilizará una instancia de este sub sistema por cada servicio geográfico, es decir por cada servidor de mapas que se quiera exponer, solo se configura un servidor y su URL, IP, puerto y la URI. Esta URI puede variar según el protocolo y según la implementación del servidor, por ejemplo existen variantes entre GeoServer y MapServer.

El componente RestInvoker es el encargado de armar el pedido HTTP correspondiente, basado en los parámetros del pedido original y la URL provista por GisSoapService.

Para la respuesta, simplemente se retorna lo enviado por el servidor de mapas, los componentes provistos por ESB que implementan el webservice se encargan de convertirlo en

un mensaje SOAP Response. Solo se debe tener en cuenta que para el método GetMap la respuesta es una imagen en formato binario y no un XML como en todos los demás métodos.

4.2.3 Configuración de los ctps y pge

Visto toda la información que se debe manejar en los CTP, esta sección describe el modelo de datos utilizado para el CTP RC, para soportar la configuración de diferentes servicios y sus propiedades de seguridad. También se describe un modelo básico que se utilizó para el simulador de la PGE creado en el proyecto.

4.2.3.1 CTP RestConnector

El modelo de datos a continuación permite configurar servicios y sus propiedades de seguridad para que las peticiones puedan atravesar la plataforma en la forma debida.

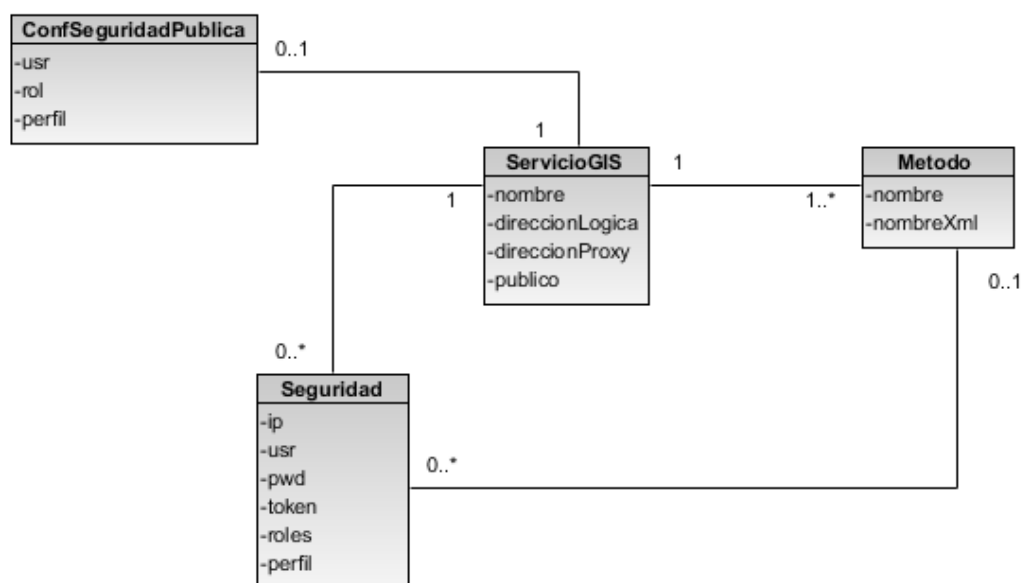


Figura 4.5 – Modelo de datos del CTP RestConnector

Este modelo es guardado en una base de datos relacional estándar y se ha construido una aplicación web sencilla para configurarlo, la cual sería una interfaz de usuario que utilizaría el administrador del CTP, para cualquiera de sus modalidades, dentro de un organismo o expuesto en internet. Tanto la base de datos como la aplicación son locales a cada instancia del CTP RC. Esto quiere decir que no son centralizados.

4.2.3.2 CTP SoapConnector

Este caso es mucho más sencillo, ya que cada instancia del CTP SC invoca un único servidor de mapas, existe un archivo de configuración donde se definirán los datos de conexión.

Nombre de la propiedad	Valor de ejemplo
url	http://localhost:8081/geoserver
Propwms	/integraciongispge/wms?
Propows	/integraciongispge/ows?
Propwfs	/integraciongispge/wfs?

La propiedad "url" indica la ubicación del servidor, y luego es concatenada con una de las otras propiedades para completar la URL a donde se enviará el pedido.

Nótese que el ejemplo es para Geoserver, para Mapserver, las propiedades son las mismas, pero cambia el valor debido a la forma de las URLs en este servidor.

4.2.3.3 PGE

Para el simulador se eligió un modelo sencillo que soporte lo básico de la configuración de servicios. La idea es simular el comportamiento en un caso de éxito o casos de falta de configuración o invocaciones con datos incorrectos.

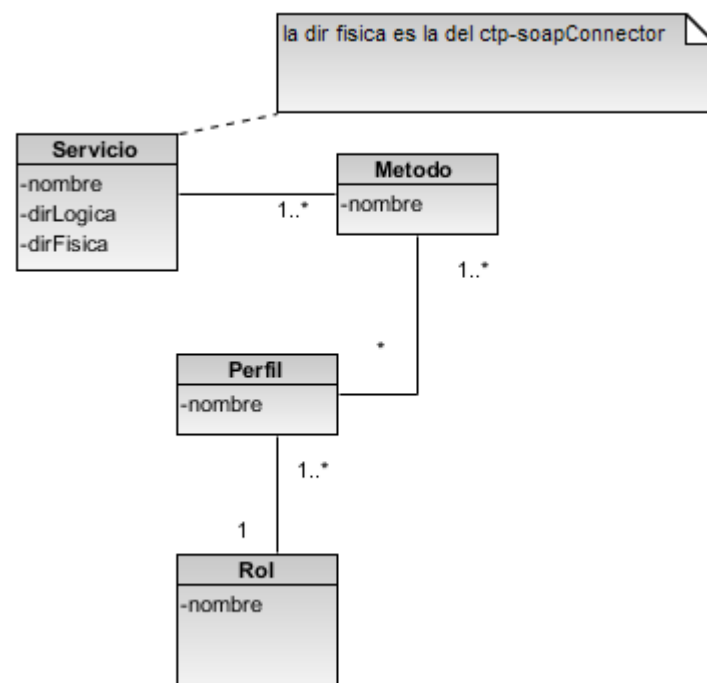


Figura 4.6 – Modelo de datos de la PGE

También es guardado en una base relacional, pero en este caso es solo por simplicidad. No se pretende usabilidad a futuro como en el CTP ya que es solo un simulador.

4.2.3.4 Servidores de mapas.

Se planteó como objetivo que la solución pudiera funcionar con los servidores de mapas Geoserver y Mapserver.

Geoserver fue elegido por ser la implementación de referencia del estándar. Soportar esta implementación significa también adaptarse mejor a cualquier otra implementación que siga el estándar.

Se optó por Mapserver ya que es de los servidores de mapas más antiguo de código abierto y además porque al ser una implementación con muchas peculiaridades que salen del estándar esto ayudaría a que la solución pudiera ser mas adaptable a otros servidores de mapas que no cumplan con el estándar.

4.2.4 Productos utilizados

La siguiente tabla resume los productos de software utilizados para implementar los subsistemas de la solución.

Sistema/Componente	Producto
Servidor de middleware ESB	JBoss ESB 4.9
Librería de web services cliente y Endpoint	JBoss WS Nativo
CTP-RestConnector	Aplicación ESB
CTP-SoapConnector	Aplicación ESB
Simulador PGE	Aplicación ESB
STS Server	STS Server del proyecto de grado [23]
Cliente STS	Cliente del proyecto de grado [23]
Interfaz configuración CTP RestConnector	Aplicación web Grails 2.4.3
Interfaz configuración Simulador PGE	Aplicación web Grails 2.4.3
Base de datos geográfica	PostgreSQL 9.3/Postgis 2.1.3
Base de datos relacional	PostgreSQL 9.3
Servidor de mapas	Geoserver 2.5 y Mapserver 3.0.6
Java	JDK 1.7.0_51
Librería de interfaz GIS	Openlayers 2.13
Software GIS de escritorio	Quantum GIS 2.6.1 y gvSIG 1.12.0

4.3 Problemas técnicos encontrados

4.3.1 JBossESB y deploy de web services

La última versión disponible de JBossESB es la 4.12, pero esta versión tiene problemas para exponer web services, lo cual impedía el correcto funcionamiento del proyecto CTP SoapConnector. Luego de una investigación y varias pruebas se encontró que es un bug conocido y planificado para la versión 4.13 que no tiene fecha de salida especificada. Por tal motivo se volvió a versiones anteriores hasta encontrar una en la que funcionara, esta fue la versión 4.9.

4.3.2 GetCapabilities y URLs declaradas

El XML respuesta del método GetCapabilities tiene no solo información de las capas provistas y otros metadatos, sino que para cada método de los estándares WMS, WFS, OWS, etc, retorna una URL absoluta al servidor. Para los clientes GIS esto es fundamental y les permite operar fácilmente contra el servidor de mapas. Pero en el contexto del proyecto, donde el servidor de mapas está protegido y preferentemente oculto del acceso externo, exponer las URLs es un problema de seguridad. Además de impedir el correcto funcionamiento del cliente, debido a que este intentará conectarse con el servidor y no tendrá acceso a ese servidor directamente.

La solución fue reemplazar las URLs del servidor de mapas por la URL del servidor CTP RC. Este procedimiento se lleva a cabo en el CTP SC, con la URL que el CTP cliente le envía como parámetro. Lo que obliga además que el CTP RC conozca su ubicación en la red, o sea como se accede a él, aunque esto último no es un problema y es totalmente configurable.

4.3.3 GetCapabilities en MapServer

Uno de los problemas que limita el soporte a MapServer es el largo máximo impuesto al archivo retornado por el método GetCapabilities. Los archivos CGI, imponen una restricción de tamaño máximo a los archivos de 4 KB. Esta restricción imposibilitó obtener el archivo completo y trabajar con él.

4.3.4 GetFeatureInfo y MapServer

Se encontró realmente compleja y con una curva de aprendizaje muy empinada la configuración de MapServer en general. Pero a esto se le suma que para habilitar el soporte del método GetFeatureInfo, este servidor exige implementar una template de alta complejidad especialmente si se quiere soportar un grupo de capas con diferentes atributos. No es imposible realizar tal tarea, pero parece estar fuera del alcance del proyecto.

4.3.5 Problemas con la implementación del STS

Para la implementación del conector se debe trabajar con un servicio de STS. De esta tecnología no existe ningún producto open source para poder utilizar en las pruebas.

Uno de los primeros intentos fue seguir un manual de Oracle, basado en tecnologías propias para generar un servidor STS [24]. Esta alternativa no fue exitosa debido a la complejidad del problema y la incompatibilidad de esas tecnologías con las de JBoss que se habían seleccionado para realizar el resto de la implementación. A su vez la implementación de tal servicio se encuentra fuera del alcance de este proyecto por lo que se decidió utilizar el trabajo de un proyecto de grado anterior. El proyecto es Orquestación de servicios en la Plataforma de Interoperabilidad de Gobierno Electrónico [23].

En la versión obtenida de este proyecto se encontraron errores en la configuración de Spring [25], lo cual hubo que solucionar para poder utilizarlo. Por otro lado se encontró el token de seguridad generado por el STS estaba incompleto, no devolvía la información de usuario y roles que exigía la documentación de la PGE. Por tanto hubo que completar la implementación para utilizar el servicio siguiendo la especificación.

Para consumir el servicio STS se pensó en principio utilizar la librería cliente-java, que es la ofrecida por AGESIC pero se encontró que esta no tiene compatibilidad con java 7. Por esto último es que se decidió utilizar el cliente STS del proyecto mencionado.

5 Caso de Estudio

A continuación se presenta el caso de estudio desarrollado que permite validar los componentes de transformación implementados.

5.1 Marco de trabajo

La Dirección Nacional de Catastro [22] que es controlada por el Ministerio de Economía y Finanzas, brinda información oficial de los bienes inmuebles de Uruguay. Su misión es *"Diseñar, realizar, conservar y administrar el catastro de los bienes inmuebles, atendiendo a sus características geométricas, físicas, económicas y jurídicas orientado a un uso multifinalitario y sirviendo de instrumento para la planificación económica y social del territorio nacional"*. En otras palabras Catastro es el responsable de generar y mantener la información geográfica de los padrones del Uruguay.

La Dirección General de Registros (DGR) [27] que es controlada por el Ministerio de Educación y Cultura, mantiene información registral. La *"función de los Registros es la publicidad de los actos y negocios jurídicos que la ley determina como trascendentes, lo cual se concreta en dos aspectos fundamentales: la inscripción de esos actos y la información de los mismos a quien lo solicita"*. En resumen DGR es responsable de mantener la información referente a los titulares de los padrones de bienes inmuebles, entre otros datos.

5.2 Descripción

Dentro del marco de trabajo presentado en la sección 5.1, el siguiente caso de estudio pretende validar el prototipo implementado haciendo referencia a los padrones de Montevideo Rural y su registro.

Hoy en día al no contar con un organismo que centralice la información se pueden producir inconsistencias de datos. Catastro divide o unifica padrones generando nuevos datos los cuales deberían notificarse instantáneamente a DGR. Esto hace que la información que se obtiene en DGR puede que no sea válida ya que puede no estar actualizada. Es por este motivo que se propone el desarrollo de un servicio geográfico que ofrezca soporte de lectura y escritura sobre los padrones rurales de Montevideo a diferentes organismos y/o público general.

5.3 Aplicación

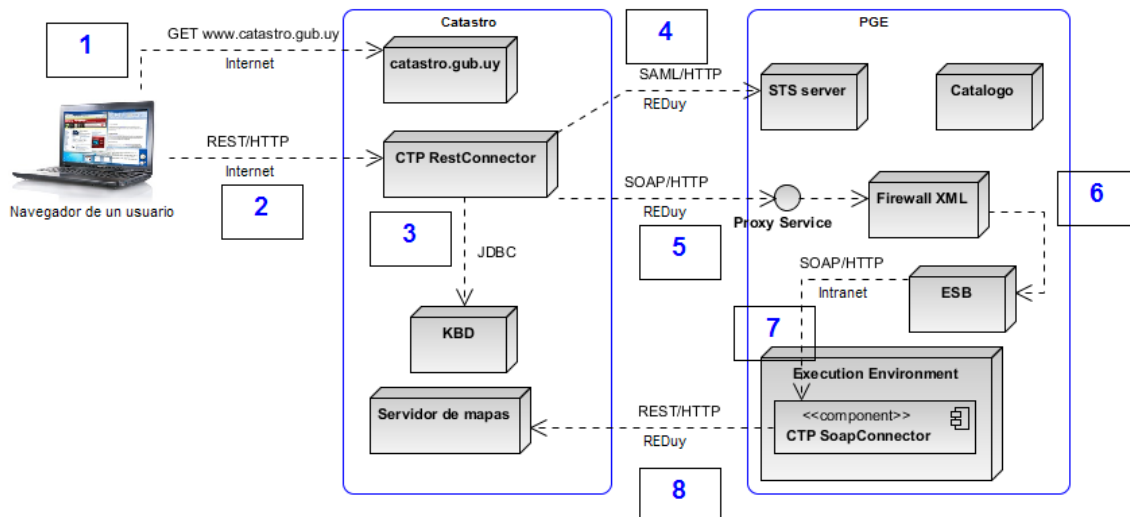
A continuación se detalla como el caso de estudio aplica a tres escenarios descritos en la tesis de maestría.

5.3.1 Escenario 1

Los padrones rurales pueden ser considerados de interés general por tal motivo dicha información geográfica podrá ser consultada tanto por organismos como por los ciudadanos o público en general.

Como ejemplo, DGR podrá acceder al servicio geográfico y así obtener información actual sobre los padrones rurales de Montevideo. Por otro lado, el público general como ser un escribano, podrá acceder a los datos sin necesidad de solicitársela al organismo.

Se plantea el siguiente diagrama:



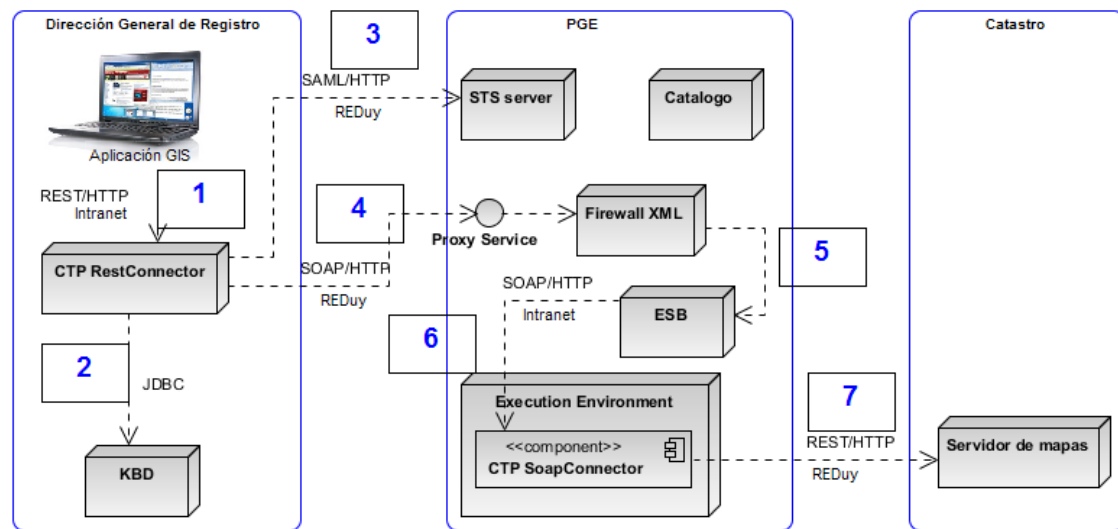
En primera instancia el usuario entra al sitio web del proveedor (1). Se carga una página con alguna librería para utilización de servicios geográficos, en este caso OpenLayers. Esta inicia la descarga del mapa directamente contra el CTP RC del proveedor (2) y sigue el flujo mostrado pasando por la PGE (5) hasta llegar al servidor de mapas que se encuentra alojado en la infraestructura del proveedor (8).

5.3.2 Escenario 2

Algunos organismos pueden brindar un servicio específico de consulta para otro u otros organismos.

En este caso de estudio Catastro brinda un servicio para que la información pueda ser consultada por la DGR. En este caso la información solo está disponible para usuarios especiales pertenecientes a la DGR y no para el público en general.

Se plantea el siguiente diagrama:



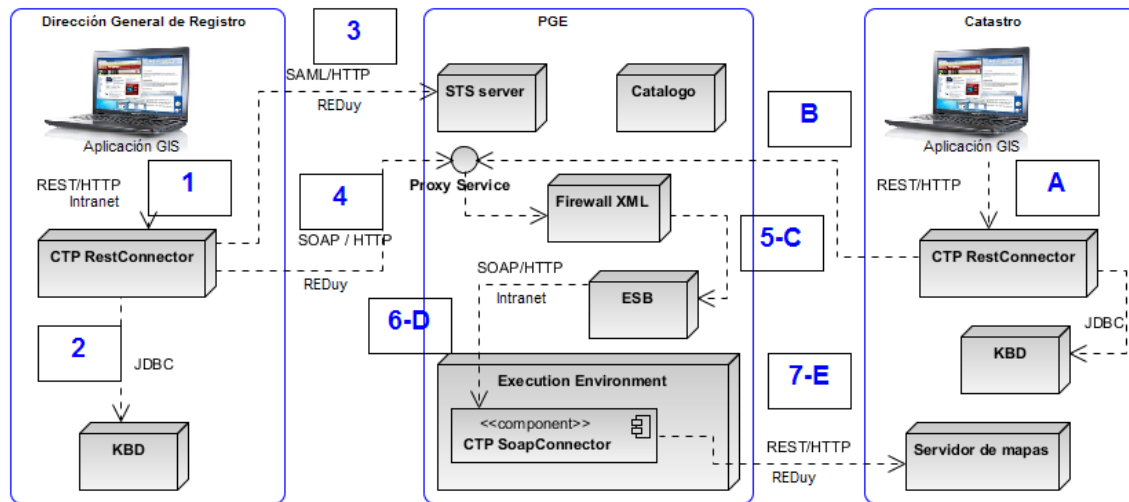
En este caso una aplicación de escritorio (1) es utilizada para acceder a la información geográfica, toda comunicación se realizará dentro de la infraestructura del organismo donde se encuentra el usuario. Este CTP RC estará configurado para acceder a los servicios geográficos que necesiten sus funcionarios para cumplir con sus tareas y nada más (2). Luego se inicia el flujo usual, pasando por los componentes de la PGE (4) para que se realicen los controles de acceso correspondientes para finalmente llegar al servidor de mapas alojado en el organismo que es dueño de la información (7).

5.3.3 Escenario 4

Existen ciertos trámites que requieren que los usuarios deban presentarse en los distintos organismos para recoger la información que necesitan. Por este motivo se plantea, en el contexto del caso de estudio, la colaboración de dos o más organismos para la realización de trámites públicos.

Catastro es el responsable de mantener los padrones rurales los cuales podrán ser visualizados al instante de su modificación por DGR, quitando así la necesidad que tienen los escribanos hoy en día de movilizarse a los diferentes organismos para obtener información actualizada.

Se plantea el siguiente diagrama:



Este escenario tiene dos organismos (DGR y Catastro) con sus respectivos CTP-RC que utilizan servicios geográficos a través de la PGE, pero solo uno tiene el servidor de mapas ya que es el encargado de mantener la información (Catastro). Igual que el escenario anterior se utiliza software de escritorio que permite consultar como modificar la información según los permisos que tenga cada usuario, en ese servicio configurado en la PGE. Por otro lado se puede notar que un usuario dentro de Catastro podría conectarse directamente al servidor de mapas, lo cual tiene sus ventajas y también desventajas, pero la opción mostrada sirve para enfatizar las diferentes alternativas brindadas por la solución. En el diagrama anterior se marcan con números el flujo de datos de un usuario en DGR y con letras el flujo desde Catastro, como ambos están utilizando el mismo servicio pero eventualmente con diferentes permisos comparten muchos de los componentes que interactúan en el flujo normal.

6 Conclusiones

Una de las primeras conclusiones que se desprenden del trabajo es que se llegó a una implementación funcional de la propuesta teórica de la Tesis de Raquel Sosa. Se logra realizar la interacción entre los diferentes sistemas mencionados y se aporta una arquitectura interesante que ataca diferentes problemas de una posible puesta en producción. Brinda además diferentes opciones de escalabilidad contemplando los diversos contextos y realidades de los entes públicos que se involucran en la creación y manejo de la información geográfica en el Uruguay.

Uno de los puntos fuertes de la solución es que se eligieron tecnologías que son o bien estándares de la industria o los mismos involucrados en los sistemas de la PGE, esto fue pensado para facilitar una posible puesta en producción del producto. Cabe destacar además que no se imponen restricciones sobre la PGE, sino que la utiliza tal cual los requerimientos de la misma para otros servicios, convirtiendo este producto completamente compatible con la arquitectura existente.

Las dificultades sorteadas se pueden separar en dos categorías, por un lado las tecnológicas relacionadas al estado del arte de las tecnologías utilizadas y las relacionadas con el diseño. En este último llevó mucho tiempo resolver el mapeo de direcciones para acceder a la PGE, pero a grandes rasgos ninguna de las dificultades evitó que se alcanzaran los objetivos planteados. Por un lado gracias al preciso análisis brindado en la Tesis de Raquel Sosa y por otro lado porque en muchos casos esas dificultades mostraron debilidades que finalmente fortalecieron la calidad de la solución.

Respecto al avance del proyecto es digno de remarcar que el inicio temprano de la implementación, realizando prototipos que atravesaran todas las capas y sistemas, permitió encontrar fallas y futuros problemas en el diseño, que fueron corregidos en etapas tempranas, esto alargó un poco las etapas de diseño e implementación pero mitigaron riesgos que a priori estaban ocultos.

Como conclusión final cabe destacar que la calidad obtenida en la solución se considera muy buena así como también la experiencia adquirida en tecnologías de información geográfica y plataformas de gobierno electrónico.

6.1 Gestión de proyecto

La gestión del proyecto se divide en cuatro etapas y una quinta etapa transversal a las anteriores:

Estado del arte: En esta etapa se realiza el estudio de la tesis de maestría de Raquel Sosa. El estudio de la PGE y de los estándares y tecnologías necesarios para el desarrollo del proyecto.

Análisis: En esta etapa se analiza la integración de la PGE con los servicios GIS mediante los escenarios de integración definidos en la tesis de maestría y la aplicación de los CTPs.

Diseño: Diseño de la arquitectura propuesta e implementación de un prototipo para evaluar su viabilidad. Diseño de un caso de estudio que aplica los escenarios sobre el desarrollo a implementar.

Implementación: Implementación de los CTPs y de un simulador de PGE, necesario para la integración. Implementación del caso de estudio.

Documentación: Esta actividad trata de la documentación generada durante todo el proyecto.

A continuación se muestran 2 gráficas de Gantt. Una con las fechas planificadas y la otra con fechas reales.

Hubo tareas que llevaron más tiempo del esperado por lo que algunas fechas se prolongaron más de lo planificado.

Entre éstos esta la comunicación con un servidor STS donde en primera instancia se subestimó la complejidad de implementar un servidor STS. Luego se decidió adaptar una solución de STS brindada por otro grupo de proyecto [23] pero a la misma hubo que hacerle correcciones con lo que se consumió más tiempo del esperado.

También la versión de JbossESB elegida al principio de la implementación presentaba limitaciones para exponer web services por lo que se tuvo que invertir tiempo investigando este problema por el cual se decidió finalmente cambiar a una versión anterior más estable.

El soporte para Mapserver resultó difícil de implementar y testear dada la alta complejidad que tiene Mapserver en su configuración.

Planificado

	Abr-2014	May-2014	Jun-2014	Jul-2014	Ago-2014	Sep-2014	Oct-2014	Nov-2014
Estudio sobre Web Services y Plataformas de Gobierno Electrónico								
Estudio sobre Información Geográfica, sus estándares y uso.								

Análisis de la Arquitectura Propuesta									
Propuestas de Implementación.									
Implementación de Prototipo									
Validación de Prototipo									
Documentación									

Real

	Abr-2014	Mayo-2014	Jun-2014	Jul-2014	Ago-2014	Sep-2014	Oct-2014	Nov-2014	Dic-2014	Ene-2015	Feb-2015	Mar-2015	Abr-2015
Estudio sobre Web Services y Plataformas de Gobierno Electrónico													
Estudio sobre Información Geográfica, sus estándares y uso.													
Análisis de la Arquitectura Propuesta													
Propuestas de Implementación.													
Implementación de Prototipo													
Validación de Prototipo													
Documentación													

Figura 6.1 – Se muestran los diagramas de gestión de proyecto planificado y el real.

7 Trabajo a Futuro

Dentro de los ítems que no entraron en el alcance del proyecto o que debido a problemas de diferente índole debieron ser excluidos durante la marcha se encuentran los siguientes puntos:

- Se encontraron problemas en la interacción con MapServer. Algunos métodos no funcionaron como era debido o fue muy complejo darles soporte. Aunque todos estos problemas estarían relacionados directamente y únicamente con la configuración del servidor. La puesta en marcha de dicho software no era parte esencial del alcance y aunque reducido, se demostró que con el mismo código, con la salvedad del manejo de la URL, se puede comunicar en los protocolos WMS y WFS en ambos servidores. Con un poco de trabajo de configuración directamente sobre este producto GIS se puede tener el sistema CTP funcionando, además de requerir testing para la validación de los resultados.
- Uno de los escenarios de estudio explicados en la tesis de Raquel Sosa, implica flujos de datos con asincronismo entre el CTP RestConnector y la PGE. El soporte para este tipo de interacción debe ser implementado en el CTP, modificando mediante cabeceras de WS-Addressing, el pedido SOAP para indicar a donde debe encaminarse la respuesta. Para este proyecto en particular se sumaban las dificultades inherentes al comunicación asincrónica, el desarrollo de dicha comunicación en el simulador básico implementado de la PGE y la falta de documentación de cómo ataca este tipo de pedidos la PGE, ya que no hay ni ejemplos ni casos de éxito para estudiar. Lo cual hacían bastante difícil la implementación. Por otro lado este no es considerado uno de los casos críticos para los objetivos del proyecto.
- Debido a la escasez de tiempo el testeo del soporte de WFS no ha sido posible, por tanto es importante terminar con esta tarea para poder decir que se soporta dicho estándar.
- Como es sabido el OGC está en proceso de finalización del estándar WFS en su versión 2.0. Estos incluyen cambios dignos de analizar, entre ellos la definición de una interfaz SOAP, esto último también está propuesto para la versión 1.3.1 de WMS. Lo cual podría reducir, modificar o hasta eliminar varias de la funcionalidades provistas por la plataforma.

8 Referencias

1. Plataforma de Gobierno Electrónico. *AGESIC*. [En línea] [Visitado: 05 de 05 de 2014.] http://www.agesic.gub.uy/innovaportal/v/452/1/agesic/plataforma_de_gobierno_electronico.html?menuderecho=5.
2. Plataforma de Interoperabilidad. *AGESIC*. [En línea] [Visitado: 05 de 05 de 2014.] http://agesic.gub.uy/innovaportal/v/1710/1/agesic/plataforma_de_interoperabilidad.html?menuderecho=5.
3. Open Geospatial Consortium. *OGC*. [En línea] [Visitado: 02 de 05 de 2014.] <http://www.opengeospatial.org/>.
4. Estándar WMS. *OGC*. [En línea] [Visitado: 02 de 05 de 2014.] <http://www.opengeospatial.org/standards/wms>.
5. Estándar WFS. *OGC*. [En línea] [Visitado: 02 de 05 de 2014.] <http://www.opengeospatial.org/standards/wfs>.
6. Notas del curso Taller de Sistemas de Información Geográficos Empresariales. *Facultad de Ingeniería de la Universidad de la República Oriental del Uruguay*. [En línea] [Visitado: 02 de 05 de 2014.] <http://www.fing.edu.uy/inco/cursos/tsi/TSIG/clases2012/WebServicesGeograficos2012.pdf>.
7. **Sosa, Raquel**. Integración de Servicios Geográficos en Plataformas de Gobierno Electrónico. *Facultad de Ingeniería de la Universidad de la República Oriental del Uruguay*. [En línea] [Visitado: 02 de 05 de 2014.] http://www.fing.edu.uy/~raquels/TesisRaquelSosa_vf_1.2.pdf.
8. **Fielding, Roy Thomas**. DISSERTATION. [En línea] [Visitado: 09 de 05 de 2014.] http://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm.
9. GeoServer. [En línea] [Visitado: 09 de 05 de 2014.] <http://geoserver.org/display/GEOS/Welcome>.
10. OSGeo Live. [En línea] [Visitado: 09 de 05 de 2014.] http://live.osgeo.org/es/overview/geoserver_overview.html.
11. SOAP Introduction. *W3Schools*. [En línea] [Visitado: 09 de 05 de 2014.] http://www.w3schools.com/webservices/ws_soap_intro.asp.
12. Introduction to WSDL. *W3Schools*. [En línea] [Visitado: 09 de 05 de 2014.] http://www.w3schools.com/webservices/ws_wSDL_intro.asp.
13. Agencia de Gobierno Electrónico y Sociedad de la Información. *AGESIC*. [En línea] [Visitado: 05 de 05 de 2014.] <http://www.agesic.gub.uy/>.
14. **Naser, Alejandra**. Gobierno Electrónico y Gestión Pública. *CEPAL (Comisión Económica para América Latina y el Caribe)*. [En línea] [Visitado: 28 de 09 de 2014.] http://www.cepal.org/ilpes/noticias/paginas/5/39255/gobierno_electronico_anaser.pdf.

15. **POLICY BRIEF, OECD.** The e-government imperative: main findings. Marzo 2003. [En línea] [Visitado: 20 de 10 de 2014.] <http://unpan1.un.org/intradoc/groups/public/documents/APCITY/UNPAN015120.pdf>.
16. W3Schools. *Web Services Addressing*. [En línea] 10 de 08 de 2004. [Visitado: 23 de 10 de 2014.] <http://www.w3.org/Submission/ws-addressing>.
17. MapServer. [En línea] [Visitado: 15 de 08 de 2014.] <http://mapserver.org/>.
18. OASIS. *WS-Trust 1.4*. [En línea] [Visitado: 08 de 05 de 2014.] <http://docs.oasis-open.org/ws-sx/ws-trust/v1.4/ws-trust.html>.
19. Marco de Desarrollo de la Junta de Andalucía. *Conceptos de seguridad en los servicios WEB*. [En línea] [Visitado: 08 de 05 de 2014.] <http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/211#WS-Trust>.
20. **PGE.** Capítulo III. *Descripción Técnica de la Plataforma de Gobierno Electrónico*. [En línea] [Visitado: 27 de 02 de 2015.] http://www.agesic.gub.uy/innovaportal/file/1454/1/capitulo_3.pdf.
21. JBoss ESB. [En línea] [Visitado: 10 de 05 de 2014.] <http://jbossesb.jboss.org/>.
22. Dirección Nacional de Catastro. [En línea] [Visitado: 04 de 03 de 2015.] <http://catastro.mef.gub.uy/>.
23. **Steffen, Martín y Penna, Emilio.** Orquestación de servicios en la PGE. Agostos 2013. [Citado el: 07 del 03 de 2015]
24. Manual de Oracle para creación de servidor STS. [En línea] [Visitado: 14 de 03 de 2015] https://metro.java.net/2.0/guide/Configuring_A_Secure-Token_Service__STS_.html
25. Spring framework [En línea] [Visitado: 14 de 03 de 2015] <http://projects.spring.io/spring-framework/>
26. **David Chappel, O'Reilly Media.** *Enterprise Service Bus: Theory in Practice*. Junio 2004. ISBN 0596006756.
27. Dirección General de Registros. [En línea] [Visitado: 21 de 03 de 2015.] <http://www.dgr.gub.uy/>.
28. OpenLayers [En línea] [Visitado: 16 de 05 de 2015] <http://openlayers.org/>