Arquitectura SOA para la integración entre software libre y software propietario en entornos mixtos.

Alejandro Guinea de Salas[(1)], Sergio Jorrín Abellán[(2)]

[(1)]Director de Geograma S.L. C/Castillo de Lantarón 8 bajo. 01007 Vitoria-Gasteiz, Alava, alejandro@geograma.com
[(2)]Responsable de informática de Geograma S.L. C/Castillo de Lantarón 8 bajo. 01007 Vitoria-Gasteiz, Alava, sjorrin@geograma.com

RESUMEN

Tanto los sistemas basados en software propietario como en software libre tienen ventajas e inconvenientes que hacen que una empresa o institución no pueda elegir, al menos a corto plazo, en una solución global basada en un solo tipo de software. Consecuencia de esto, lo más habitual es encontrarnos con entornos mixtos en lo que la política de software se refiere. Se trata por tanto de lograr una coexistencia de ambos tipos de software, de tal manera que se consigan las máximas ventajas y los mínimos inconvenientes de cada una de las dos opciones de software, que faciliten la posterior migración, si se estima conveniente, a un entorno totalmente libre.

La ponencia expone un punto de vista práctico a la integración entre sistemas libres y propietarios, desde una perspectiva de arquitectura orientada a servicios (SOA), que logra aprovechar las ventajas de las diferentes opciones de software. La arquitectura mencionada se ha aplicado con éxito en diferentes proyectos de implantación de sistemas de información geográfica y proyectos empresariales de carácter interno realizados por la empresa Geograma.

SOA está resultando un paradigma para la integración de sistemas empresariales, que no sólo integra diferentes tipos de software sino que además es independiente de la plataforma, el sistema operativo y el dispositivo. Esta integración es especialmente interesante desde el punto de vista de los sistemas de información geográfica, tradicionalmente aislada de los sistemas corporativos. Los servicios web OGC son una evidencia palpable de la presencia cada vez mayor de arquitecturas basadas en servicios.

Implantar un sistema de información geográfica, basado en software libre, propietario o mixto, desde una perspectiva basada en servicios, asegura una escalabilidad, flexibilidad sin precedentes y, desde un punto de vista empresarial, una mejora del ROI.

Palabras clave: *Arquitectura SOA, sistemas de información geográfica, integración software libre, interoperabilidad.*

INTRODUCCIÓN

En las últimas décadas, los departamentos de IT de las empresas han construido infraestructuras que soportan en gran medida la operaciones de sus empresas y sus clientes. El camino para llegar hasta este punto no ha sido fácil. Se ha aprendido de los errores y aciertos de la industria. El resultado de este proceso, ha sido la creación y mantenimiento de un número considerable de aplicaciones internas, cada una responsable de sus propias tareas.

El negocio cada vez exige crear aplicaciones más complejas, con menos tiempo y presupuesto que antes. Crear estas aplicaciones, requiere en muchos casos de funcionalidades ya antes implementadas como parte de otros sistemas. En este punto los arquitectos de soluciones tienen dos opciones:

1. Tratar de reutilizar la funcionalidad ya implementada en otros sistemas. Una labor difícil de realizar, debido a que estos no fueron diseñados para integrarse o sobre plataformas y/o tecnologías incompatibles entre ellas. Incluso en el caso de encontrarse la manera técnica de realizar la conexión, se debe asumir el riesgo de alterar un sistema en producción que esta funcionando sin problemas.

2. Re-implementar la funcionalidad requerida ("reinventar la rueda"). Aunque implica mas tiempo de desarrollo, es la mas fácil y segura, aunque no las más acertada a largo plazo porque trae como resultado:

- Funcionalidad replicada en varios aplicativos
- Dificultad de migración de los sistemas internos. Al haber múltiples conexiones desde sistemas que dependen de estos para su funcionamiento.
- Al no haber una estrategia de integración de aplicaciones, se generan múltiples puntos de fallo, que pueden detener la operación de todos los sistemas muy fácilmente.
- Un modelo así, por lo general no escala muy bien.
- El inconveniente final es una pobre respuesta al cambio. Las aplicaciones siguen siendo concebidas desde un principio como islas independientes.

COMO SOA NOS PUEDE AYUDAR

Una estrategia de aplicaciones empresariales debe facilitar su integración. Además que debe motivar la construcción de servicios, mas que de aplicaciones. Estos servicios se encargarían de exponer una funcionalidad bien definida a la aplicación que la requiera.

De esta manera, una aplicación final simplemente orquesta la ejecución de un conjunto de estos servicios, añade su lógica particular y le presenta una interfaz al usuario final.

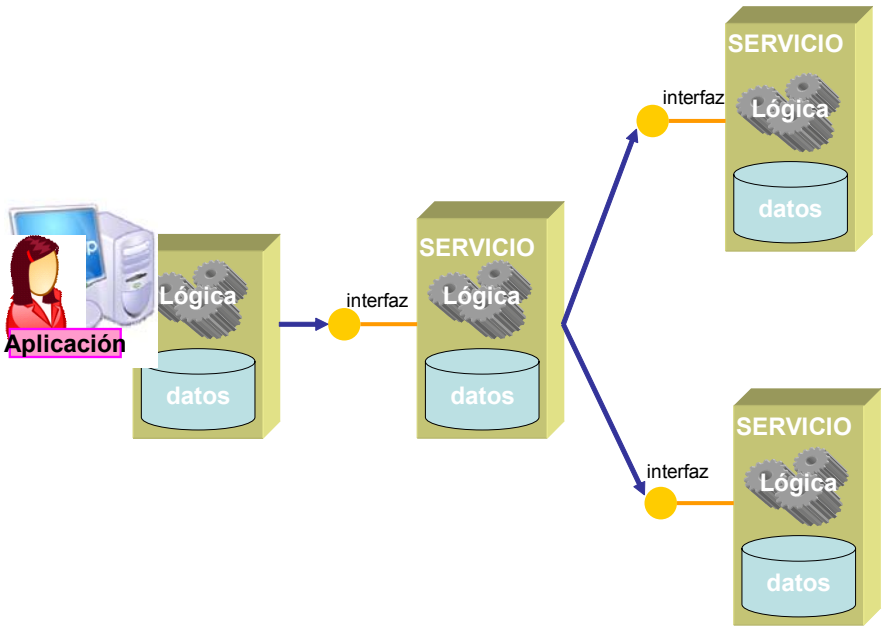

Figura 1. Un sistema sencillo basado en servicios. Una aplicación además de implementar sus propios componentes de negocio y datos, también puede reutilizar la funcionalidad de servicios existentes en la red empresarial.

Exponer procesos de negocio como servicios es la clave a la flexibilidad de la arquitectura. Esto permite que otras piezas de funcionalidad (incluso también implementadas como servicios) hagan uso de otros servicios de manera natural, sin importar su ubicación física. Así un sistema evoluciona con la adición de nuevos servicios y con su mejora continua. Donde cada servicio evoluciona de una manera independiente. La Arquitectura Orientada a Servicios (SOA) resultante, define los servicios de los cuales estará compuesto el sistema, sus interacciones, y con que tecnologías serán implementados.

VISIÓN INTERNA DE LOS SERVICIOS

Un servicio debe ser una aplicación completamente autónoma e independiente. A pesar de esto, no es una isla, porque expone una interfaz de llamado basada en mensajes, capaz de ser accedida a través de la red. Generalmente, los servicios incluyen tanto lógica de negocio como manejo de estado (datos), relevantes a la solución del problema para el cual fueron diseñados. La manipulación del estado es gobernada por las reglas de negocio.

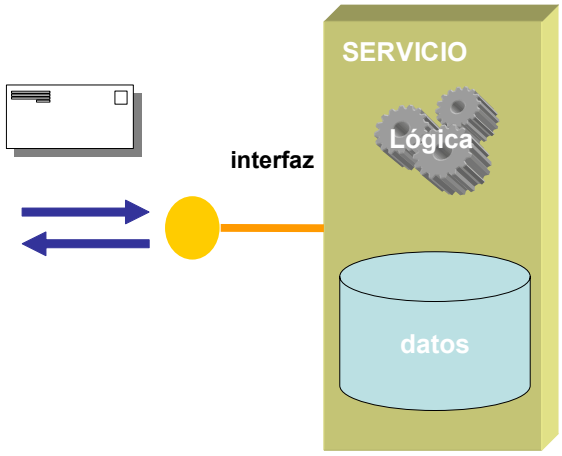

Figura 2: Visión interna de los servicios. Un servicio funciona como una aplicación independiente. Teniendo sus propias reglas de negocio, datos, procedimientos de administración y operación. Expone toda su funcionalidad utilizando una interfaz basada en mensajes. Y por lo tanto carece de una interfaz de usuario.

La comunicación hacia y desde el servicio, se realiza utilizando mensajes y no llamadas a métodos. Estos mensajes deben contener o referenciar toda la información necesaria para entenderlo. La idea es que haya el mínimo posible de llamadas entre el cliente y el servicio.

Un servicio es la evolución en complejidad de un componente distribuido, y se diferencian en:

- Mucho menos acoplados con sus aplicaciones cliente que los componentes.
- Menor granularidad que los componentes.
- No son diseñados e implementados necesariamente como parte de una aplicación *end-to-end*.
- Son controlados y administrados de manera independiente.
- Expone su funcionalidad a través de protocolos abiertos e independientes de plataforma. Incluso arriesgando el rendimiento y consumo de recursos.
- Son transparentes de su localización en la red, de esta manera garantizan escalabilidad y tolerancia a fallos.
- Tienen sus propias políticas de escalabilidad, seguridad, tolerancia a fallos, manejo de excepciones, configuración, etc.

INCONVENIENTES A TENER EN CUENTA EN EL DISEÑO SOA

La implementación ideal de un servicio exige resolver algunos inconvenientes técnicos inherentes a su modelo:

- Los tiempos de llamado no son despreciables, gracias a la comunicación de la red, tamaño de los mensajes, etc. Esto necesariamente implica la utilización de mensajería confiable.

- La respuesta del servicio se ve afectada directamente por aspectos externos como problemas en la red, configuración, etc. Estos se deben tener en cuenta en el diseño, desarrollándose los mecanismos de contingencia que eviten la parada de las aplicaciones y servicios que dependen de él.

- Debe manejar comunicaciones no seguras, mensajes impredecibles, reintentos, mensajes fuera de secuencia, etc.

Según lo anterior, se puede ver que la construcción de un servicio es una tarea mucho mas complicada que la de un simple componente distribuido.

El servicio debe publicar una interfaz (por ejemplo, utilizando WSDL o *proxies*) fácilmente localizable en la red. Esta interfaz debe servir como un contrato de servicio. Donde se describen cada una de las funciones que provee, e incluso los niveles de prestación de servicio (SLA). Esta interfaz se debe documentar de forma clara, de manera que sea muy fácil implementar una conexión.

Un diseño exitoso de una arquitectura basada en servicios debe estar basado en una plataforma de mensajería segura, que aísle de la implementación funcional muchos de los problemas anteriormente mencionados. Algunas de las responsabilidades de un mecanismo así, incluyen:

- Entrega garantizada de mensajes

- Enrutamiento de peticiones a un servicio disponible.

- Seguridad del contenido de los mensajes

- *QoS* (quality of service) Calidad del Servicio

- Escenarios fuera-de-línea

Cuantas más de las características mencionadas soporte la tecnología elegida, menos problemas tendrá la solución final en operación.

Una comunicación basada en mensajes por lo general implica que no existen sesiones. Por lo tanto, los clientes no guardan estado en el servicio (mayor escalabilidad), y la autenticación se debe dar a nivel de cada mensaje.

Por ultimo, los servicios deben ser diseñados para controlar internamente la transaccionalidad de sus propias operaciones. No es recomendable que una transacción traspase los límites de un servicio.

ENTERPRISE SERVICE BUS

La intersección de la arquitectura orientada a servicios con la integración de aplicaciones y el modelado de procesos de negocio, dan lugar a un nuevo producto

denominado Enterprise Service Bus (ESB). Es un concepto antiguo, pero que adquiere fuerza de nuevo gracias a la facilidad y bajo coste con el que se puede implantar actualmente debido a los avances tecnológicos que se han producido.

El ESB es un elemento de software, un middleware, una infraestructura basada en estándares, que proporciona servicios para la construcción de arquitecturas más complejas basadas en eventos y en un motor de mensajería (el BUS)

El ESB permite la integración de aplicaciones de forma rápida, directa y basada en estándares. Es una suite de productos independientes de la infraestructura que facilita el procesado, la transformación de datos, el enrutamiento y la orquestación de procesos usando Web Services.

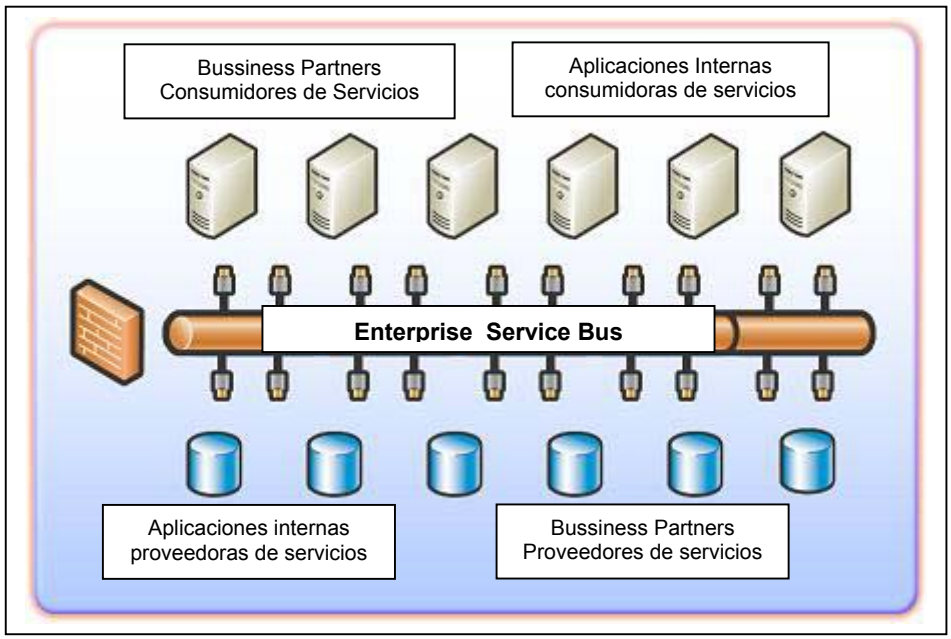

Figura 3: Esquema del Enterprise Service Bus

El ESB está integrado por aplicaciones proveedoras de servicios. Cada una de las aplicaciones puede ser independiente de las demás. Esta posibilidad de integración es independiente del modo de licencia del software, libre o propietario. Esta arquitectura nos permite, por ejemplo, a través del ESB utilizar con una herramienta de software propietario unos servicios desarrollados sobre plataformas de software libre.

El ESB puede hacer uso de la infraestructura existente de servidores de aplicaciones, transportes, aplicaciones y datos por lo que típicamente resulta en un ROI de un orden de magnitud mejor que los enfoques tradicionales de integración de aplicaciones.

APLICACIÓN A LOS SISTEMAS DE INFORMACIÓN GEOGRÁFICA

Una vez definida la arquitectura SOA, y comprobado que por definición es capaz de integrar diferentes herramientas a través del ESB, independientemente de la licencia del software (libre o propietario), vamos a aplicar un ejemplo de implantación SOA en el entorno de los sistemas de información geográfica. La arquitectura presenta tantas ventajas para la problemática habitual de los SIG que ningún proyecto de información geográfica debería comenzar sin al menos evaluar seriamente la opción de enfocarlo desde una perspectiva orientada a servicios.

WMS, WFS son siglas que cada vez están más presentes en el mundo GIS, especialmente cuando se refiere a Internet. Son protocolos basados en XML y http. Como éstos protocolos, otros muchos definidos por el OGC (Open Geospatial Consortium) establecen las reglas para el intercambio de información y funcionalidad geográficas. Sin embargo, estos protocolos no suelen estar presentes en proyectos de implantación de GIS, como no sea como un protocolo previsto para exponer información por Internet, pero no como estrategia interna. Para la estrategia interna, se estudian modelos en los que las partes cliente-servidor están íntimamente ligadas, de tal manera que a menudo es inconcebible utilizar distintos fabricantes de software, sea libre o propietario, para cada una de las partes.

Los protocolos definidos por OGC no son los únicos estándares que se pueden valorar para definir geoservicios. La definición de un Web Service basado en SOAP es un estándar ampliamente extendido y aceptado, y para determinadas problemáticas, sobre todo cuando OGC no solucione una necesidad concreta, es una opción muy válida. Esto es, podemos definir un servicio Web con funcionalidad geográfica estándar cuando OGC no solucione una lógica de negocio determinada.

Se trata, en resumen, de aportar al ESB corporativo una serie de servicios con funcionalidad geográfica que permitan su integración con las aplicaciones, internas o externas, de la empresa o institución. Estos servicios pueden ser estándares geográficos (WMS, WFS) o estándares de más bajo nivel (Web Services SOAP). La política de OGC es que ambos vayan convergiendo, pero en las versiones actuales todavía esta convergencia no se da.

Los servicios Web OGC tienen las siguientes ventajas:

- Correcta definición. No en vano existe un importante grupo de expertos detrás de ellos.

- Existencia de aplicaciones que los consumen y los ofrecen, que varían en función de la extensión del estándar. Por ejemplo, WMS está muy extendido pero otros protocolos tienen grandes carencias en este sentido. Esta existencia de aplicaciones, muchas veces gratuitas y con código abierto, pueden facilitar los desarrollos.

- Acoplamiento débil, que posibilita construir sistemas incrementalmente, y facilita el desarrollo de aplicaciones

En cuanto a los inconvenientes de los GeoServicios OGC, podemos citar:

- Están orientados a personal experto en mapas y no a personal técnico. Los protocolos son relativamente complejos, y es necesario un aprendizaje importante.

- Tienen limitaciones de funcionalidad. Los estándares publicados son los que son. Aunque existen multitud de proyectos, no todos están operativos. Los cambios de versiones van eliminando esta carencia.

- No contemplan el aspecto de la seguridad. Delegan esta funcionalidad en otros niveles lógicos. Autenticación, confidencialidad, autorización y no repudio son niveles de seguridad importantes en los servicios, que se deben implementar al margen de los estándares.

- Lentitud en su tramitación. Los estándares tienen una tramitación lenta, que además no asegura que se conviertan en estándares de hecho. Para cuando los estándares son publicados y existe una comunidad de usuarios crítica, es posible que se hayan quedado obsoletos. Esto impide su utilización para proyectos con alto nivel de investigación o innovación. Un ejemplo de esta desventaja es el soporte para SOAP y WSDL del estándar WMS, todavía en discusión.

Si se opta por utilizar (servir y consumir) los servicios propietarios (no OGC) que ofrecen las grandes marcas fabricantes de software, su principal limitación es precisamente uno de los cuatro puntos en los que se basa Tim Ewald (Microsoft) como importantes a tener en cuenta para "...vivir en el mundo nuevo de los servicios Web: Todo se basa en un acoplamiento débil. En ello se basa el éxito de la Web y lo que hace interesantes los servicios Web..."

A la complejidad de los mismos se añade el fuerte acoplamiento, y que los servicios aportados al ESB están controlados por el fabricante, lo que no lo hacen recomendable para una arquitectura SOA salvo en casos de gran complejidad funcional y alto número de usuarios y/o dispositivos.

La mejor opción es el desarrollo de un ESB corporativo, que ofrezca servicios que se acoplen débilmente con las aplicaciones geográficas, que encapsulen la complejidad y de gran flexibilidad al estar controlados por la propia empresa o institución. Esto no impide utilizar los estándares ya creados, con lo que se aprovechan multitud de sus ventajas, según el caso concreto que se necesita resolver.

CASO PRÁCTICO

Una empresa o institución necesita resolver la siguiente funcionalidad geográfica: A la hora de realizar un informe con un procesador de textos, los técnicos desean incorporar un plano de situación dada una población concreta.

Dada la complejidad de un servidor se mapas no se plantea su desarrollo y se utiliza un servidor de mapas de software libre para exponer la funcionalidad necesaria en la red según el protocolo WMS.

Se desarrolla un servicio Web SOAP que dada una población determinada, consulta a una base de datos corporativa (a través de un servicio también expuesto en el ESB), y obtiene el nombre normalizado y las coordenadas de la población en

cuestión; Una vez obtenidas las coordenadas accede al servicio WMS para obtener tres mapas a diferentes escalas del entorno de la población indicada.

Se desarrolla una aplicación en el entorno del procesador de textos, que accede a un servicio Web normalizado que únicamente pasa la población y recibe tres imágenes JPG con un tamaño determinado para insertarlas directamente en el informe.

El servicio Web SOAP se documentará abundantemente, y permitirá otros desarrollos que necesiten la misma funcionalidad, sin necesitar conocimientos de GIS. Por ejemplo, el mismo caso sería directamente aplicable en dispositivos tipo PDA, envíos por correo electrónico, inserción en planos y cualquier aplicación específica del negocio, permaneciendo la parte de los Web Services (ESB) y el servidor de mapas invariable.

Si se diera el caso de necesitar sustituir o actualizar el servidor de mapas, el cambio se podría realizar de manera totalmente transparente para las aplicaciones desarrolladas. Incluso en el supuesto de que cambiara la versión o la forma de acceder al servidor WMS, o cambiando de protocolo de acceso a los mapas, ya que la capa lógica con el servicio que se ha introducido en el ESB se encargaría de hacer transparentes estas modificaciones para las aplicaciones.

La arquitectura planteada proporciona gran flexibilidad. Por ejemplo, estableciendo unos protocolos de autenticación, se puede acceder al servicio Web desde fuera de la organización de manera sencilla (casa, dispositivos móviles, etc.) Se puede dar el caso de decidir externalizar el servidor de mapas, y/o conectarse a otros servidores con otro tipo de información, o información de otros ámbitos. Estas opciones están disponibles sin necesidad de realizar ningún cambio, facilitando la adaptación a los futuros cambios. Además de esto, para implementar los servicios se ha aprovechado totalmente la infraestructura de servidores de la empresa, sin que haya sido necesario adquirir ningún elemento nuevo para establecer el ESB.

Geograma lleva aplicando este modelo en el problema geográfico desde hace varios años, a través de la plataforma SOA www.geoservicios.com. La arquitectura está resultando de una flexibilidad e integración sin precedentes, integrando software libre, propietario, entornos Web, entornos desktop, dispositivos móviles y diferentes lenguajes de programación. Los servicios Web están integrados en el ESB de corporaciones muy diversas, que abarcan sectores como la banca, telefonía o las administraciones públicas, resultando una solución de integración útil y de rápido despliegue incluso por personal no especializado en sistemas de información geográfica.

CONCLUSIONES

Indiscutiblemente, SOA es el paradigma actual en cuanto a arquitectura de software se refiere, como lo demuestra la apuesta de todas las casas de software y la rapidez con la que se está implantando en las empresas. Las ventajas de la arquitectura son aplicables directamente a los sistemas de información geográfica. Tal es su potencia, que los responsables de los departamentos que soportan los sistemas de información geográfica no se deben plantear si se va a implantar, sino cómo se va a implantar una arquitectura orientada a servicios.

REFERENCIAS

- Microsoft development network (MSDN) http://www.microsoft.com/spanish/msdn
- IBM http://ww.ibm.com/soa/es
- Wikipedia http://es.wikipedia.org