IX JORNADAS DE CIENCIA Y TECNOLOGÍA
DE FACULTADES DE INGENIERÍA DEL NOA
Santiago del Estero, 3 y 4 de Octubre de 2013

Arquitecturas avanzadas de software aplicadas en dos casos de estudio del ámbito gubernamental

Marcelo A. Castro[1], Víctor D. Sánchez Rivero[1], José H. Farfán[1], Andrea G. Cándido[1], Luis A. Vargas[1], Daniel Castro[1], Elizabeth D. R. Reinoso[1], María C. Aparicio[1], Fabiana R. Aragón[1], Liliana B. Cazón[1] & José V. Zapana[1]

(1) Investigación + Desarrollo en Gobierno Electrónico / Facultad de Ingeniería /Universidad Nacional de Jujuy.
mcastro@fi.unju.edu.ar, vdsanchezrivero@fi.unju.edu.ar, jhfarfan@fi.unju.edu.ar, agcandido@fhycs.unju.edu.ar, lavargas@fi.unju.edu.ar, ddcastro@fi.unju.edu.ar, edrreinoso@fi.unju.edu.ar, mcaparicio@fi.unju.edu.ar, fraragon@fi.unju.edu.ar, lbcazon@fce.unju.edu.ar, jvzapana@ fi.unju.edu.ar

RESUMEN: En el presente trabajo se desarrollaron dos aplicaciones de Arquitecturas Orientadas a Servicios (SOA) en el ámbito de la Administración Pública. En primer lugar se realizó una propuesta de aplicación de la arquitectura Modelo-Vista-Controlador (MVC) sobre un diseño arquitectónico existente en un organismo público de nivel municipal, lo cual incluyó la pertinencia de la aplicación de MVC al caso de estudio elegido, formalizando una propuesta de cambio arquitectónico; para la cual se describió su arquitectura, se detallaron algunas de sus funcionalidades y características más importantes de implementación, utilizando un esquema basado en el portafolio de servidores de aplicación Oracle. Finalmente se desarrolló un esquema de aplicación de Servicios Web SOAP/REST para utilizar funcionalidades existentes montados sobre una arquitectura cliente/servidor típica, con la que cuenta cualquier área de sistemas de la Administración Pública que tenga la responsabilidad de ejecutar aplicaciones transversales críticas como lo son los Sistemas de Administración Financiera, Liquidación de Haberes, Expedientes y verticales como Impuestos, Catastro, Obras Públicas e Historia Clínica, entre otras.

INTRODUCCIÓN

Los Sistemas de Información Distribuidos (SID) han experimentado un gran desarrollo en los últimos tiempos, ya que se considera esencial brindar servicios no solo a los sectores internos de una empresa -quizás localizados en un mismo edificio- sino que también a aquellos miembros, geográficamente, alejados de la casa matriz, por ejemplo. Y, si es necesario, también ofrecer servicios a componentes externos a la organización.

Coulouris definió a los sistemas distribuidos como "sistemas en los que los componentes hardware y/o software existentes en una red de computadoras, se comunican y coordinan sus acciones mediante el intercambio de mensajes" (Coulouris et al., 2001).

La extraordinaria evolución de la Web y la gran variedad de servicios insatisfechos, como por ejemplo, los servicios que una organización gubernamental (de cualquier orden) debiera brindar al ciudadano común –tema que los responsables de las administraciones públicas tratan de abordar a través de diferentes procesos de solución, siendo Gobierno Electrónico (GE) uno de ellos– hizo que se estudien e investiguen diferentes arquitecturas, como alternativas, para dar respuesta al problema.

En tal sentido y de acuerdo a los lineamientos generales establecidos por el Plan Nacional de Gobierno Electrónico y Planes Sectoriales de Gobierno Electrónico, los entes gubernamentales partícipes, como casos de estudio, adhieren al espíritu de tal normativa incorporando herramientas tecnológicas que aumenten la transparencia de los actos de gobierno y den rápida respuesta a las necesidades y requerimientos de la población.

Teniendo en cuenta lo anterior, los SID se diseñan en capas, las cuales son conceptuales y son vistas lógicas de la arquitectura, y pueden ser clasificados en una de las siguientes categorías: arquitecturas de una, dos o más capas, las cuales se sustentan en patrones de diseño estándar.

Como resultado de la investigación de las características de los patrones, se decide aplicar, a través de un caso de estudio en un organismo gubernamental, el Patrón MVC, que pertenece al grupo de los patrones de diseño creacionales.
Simultáneamente, en la actualidad, se está empleando SOA (Arquitecturas Orientadas a Servicios), como medio para implementar sistemas distribuidos.
Sin embargo, la implementación eficiente de estos sistemas en la Web, acarrea ciertos desafíos o necesidades tales como el rendimiento, la experiencia de los usuarios, la reusabilidad y la compatibilidad de los servicios. Estas necesidades dieron origen a los Servicios Web, que en definitiva brindan mecanismos estándar de interconexión entre los usuarios y la información residente en servidores.
En resumen, estos servicios extienden las características de SOA, sobre todo en lo que respecta a interoperabilidad y reusabilidad (Los Santos, 2009).
Es importante señalar que en ambos casos de estudio, tanto el centro de cómputos a nivel provincial como el municipal, se parte de una plataforma tecnológica existente. Es decir, se aprovechan los recursos informáticos con que ya cuentan las mencionadas reparticiones gubernamentales, particularmente en la utilización de las herramientas para migrar a MVC que ofrece Oracle, Gestor de Bases de Datos con el que cuenta desde hace más de 10 años el municipio, objeto de estudio del presente trabajo de investigación (Ver apartados "Arquitectura Actual" y "Ejemplo de Arquitectura Existente").

EL PATRÓN ARQUITECTÓNICO MVC

Al hacer referencia al tipo de Arquitectura de Software que se implementará en un sistema, uno de los aspectos principales es decidir que patrones de arquitectura y diseño de software se utilizarán.
El Patrón Arquitectónico MVC es uno de los tantos patrones de arquitectura de software existentes en el mercado; y es empleado, generalmente, para el desarrollo de aplicaciones Web que manejan gran cantidad de datos y transacciones complejas; debido a que este patrón facilita la funcionalidad, mantenibilidad y escalabilidad, del sistema, de forma simple y sencilla y a la vez impide mezclar lenguajes de programación en el mismo código (Bahit, 2011).
El MVC es el encargado de separar, en una aplicación, la lógica de la interfaz del usuario en tres capas: la capa de datos, la capa de interfaz y la capa lógica; facilitando la programación en diferentes capas, de manera paralela e independiente.
Los niveles de abstracción (Bahit, 2011) se pueden describir de la siguiente forma:
Modelo: representa la lógica. Es el encargado de acceder de forma directa a los datos actuando como intermediario con la base de datos.
Vista: es la encargada de mostrar la información al usuario de forma gráfica, es decir se encarga de la interfaz.
Controlador: es el intermediario entre la vista y el modelo. Controla las interacciones del usuario solicitando los datos al modelo y entregándolos a la vista para que ésta lo presente al usuario.

ARQUITECTURA ORIENTADA A SERVICIOS

La Arquitectura Orientada a Servicios (SOA) no es un concepto fácil de definir. Diversos autores afirman que es un enfoque para diseñar y construir soluciones de negocios; tal que permitan, a las organizaciones, unir sus objetivos con la infraestructura de las Tecnologías de la Información y la Comunicación (TIC).
Microsoft Corporation define SOA como: "una estrategia general de organización de los elementos de IT (Information Technologies), de forma que una colección abigarrada de sistemas distribuidos y aplicaciones complejas se pueda transformar en una red de recursos integrados, simplificada y sumamente flexible" (Microsoft, 2006).
Diego Marsili establece que "SOA no se trata de software o de un lenguaje de programación, SOA es un marco de trabajo conceptual que permite a las organizaciones unir los objetivos de negocio con la infraestructura de TIC integrando los datos y la lógica de negocio de sus sistemas separados" (Marsili, 2007).
De acuerdo a un trabajo desarrollado anteriormente por el equipo de investigación IDGE (Investigación + Desarrollo en Gobierno Electrónico) se debe distinguir falsos supuestos y verdades sobre SOA, donde, básicamente, se insiste que no es una tecnología ni una metodología, además no asegura una adecuada articulación entre TIC y las necesidades del negocio, no requiere una tecnología completa y la revisión de los procesos del negocio, debe ser gradual y construida sobre las aplicaciones actuales mediante procesos iterativos e incrementales; y es un medio para la entrega de soluciones, no un objetivo. Se aclara también que la manera más habitual de implementarla es a través de servicios web, que no significa que estos servicios requieran una arquitectura SOA, y viceversa (Castro et al., 2012).

El manejo ineficiente de la información es el origen principal de muchas deficiencias, debido especialmente a la imposibilidad de poder presentar la información de manera sencilla y coherente, esto repercute principalmente en un aspecto fundamental, para Gobierno Electrónico como por ejemplo, lo es la interoperabilidad.

Si nos referimos a organizaciones gubernamentales, IDGE mencionaba en el año 2009, la importancia y los beneficios de la interoperabilidad entre las distintas aplicaciones: obtener e integrar la información en tiempo real, consolidar las operaciones transaccionales, permitir que la información sea compartida por diversas reparticiones, obtener una única Base de Datos, eliminar las diversas interfaces que se encuentran en las distintas unidades de organización, estandarizar y homogeneizar las distintas herramientas de acceso a la información, facilitar los procesos de reingeniería y documentación, mejorar la gestión de los flujos de trabajo (workflow) y finalmente aumentar la productividad disminuyendo la duplicación de los esfuerzos de los distintos actores (Castro et al., 2009).

SERVICIOS WEB

Un Servicio Web es una tecnología que utiliza diversos estándares y protocolos para intercambiar datos entre aplicaciones (Wikipedia, 2013). Estas aplicaciones de software se pueden programar en distintos lenguajes de programación y acceder a distintas bases de datos a la vez, ya sea desde una red local o desde Internet por medio de computadoras o cualquier otro dispositivo que sea compatible con dicha aplicación.

Sin embargo, se puede decir que el rendimiento de una aplicación web distribuida es más lenta que una aplicación cliente/servidor y que al utilizar el protocolo HTTP, es necesario fortalecer las medidas de seguridad mediante el uso de firewall y sistemas de auditorías (Hansen, 2007).

Los Servicios Web han ido evolucionando y se pueden distinguir distintas generaciones, aunque en este trabajo utilizaremos SOAP y REST:

- SOAP (Simple Object Access Protocol): este protocolo permite a un WS (Web Service) comunicar dos objetos, de diferentes procesos, mediante XML; es muy recomendable para entornos formales y donde las funcionalidades de interfaz y datos estén claramente definidas.
- REST (REpresentacional State Transfer): arquitectura recomendada para sistemas distribuidos ideales como los entornos Web, debido a que trabaja con estándares HTTP y XML para transmitir datos sin tener que utilizar una capa de transporte extra como lo hacen los Servicios Web basados en SOAP.

PROPUESTA DE ADAPTACIÓN DE UN MVC SOBRE UN MODELO ARQUITECTÓNICO EXISTENTE

A continuación se detallará el caso de estudio, perteneciente a un municipio con aproximadamente 300 mil habitantes, en el que se desarrollan diferentes funcionalidades como por ejemplo Cobro y Gestión de Servicios tales como impuestos, infracciones, solicitud de libre deuda, administración de personal, entre otras.

ARQUITECTURA ACTUAL

El modelo arquitectónico que se encuentra actualmente en producción y que se puede apreciar en la Fig. 1, es un esquema clásico Cliente/Servidor estándar basado en los siguientes componentes:

- Un servidor de Base de datos ORACLE soportado por el sistema operativo Linux.
- Un servidor de aplicaciones con sistema operativo Windows Server 2003.
- Tres nodos ubicados físicamente fuera de la casa central, los cuales se encuentran conectados mediante fibra óptica a los servidores

Los nodos funcionan como sucursales o lugares de cobro y gestión de servicios para los ciudadanos, en distintos barrios de la ciudad. Cada nodo cuenta con una red interna conectada, a su vez, a la casa central; accediendo a los servidores de aplicación y base de datos.

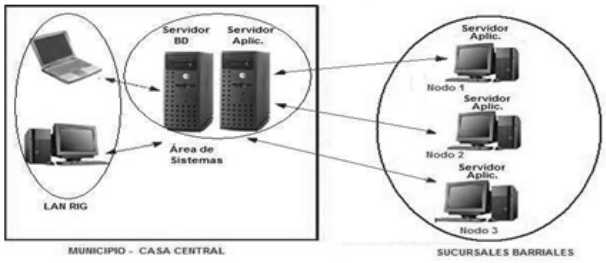

Figura 1. Arquitectura actual.

PROPUESTA DE ARQUITECTURA MVC

En la actualidad se hace necesario desarrollar e implementar arquitecturas de software que ayudan a integrar diversas plataformas con múltiples tecnologías, debido a que las empresas demandan soluciones integrales que se encuentran directamente relacionadas con sus constantes necesidades de cambio (Pérez, 2004).

El área encargada de llevar a cabo dicha integración de los diferentes sistemas, comúnmente se denomina Middleware. Este software puede ser clasificado en:

- Remote Procedure Call – RPC middleware, es el encargado de facilitar la realización del llamado de procedimientos remotos como si fuesen locales.
- Object Request Broker – ORP middleware, tiene como función particular distribuir y compartir los objetos de una aplicación a través de redes heterogéneas.
- Message-Oriented Middleware – MOM middleware; proporciona, a aplicaciones distribuidas, la posibilidad de comunicar e intercambiar información a través del envío y recepción de mensajes.

En este caso de estudio se adaptará una solución propuesta por Esahu Alfonso Pérez Morales, en el documento "Arquitectura MOM (Message-Oriented Middleware) basada en el portafolio de servidores de aplicación Oracle" (Pérez, 2004), el cual se adecúa a las necesidades del municipio aunque requiere de ciertas características particulares del organismo público. Es importante señalar que podría utilizarse cualquier otra herramienta disponible en el mercado que permita adaptarse a los requerimientos del caso de estudio.

Este modelo establece ciertas condiciones de implementación como, por ejemplo, un esquema de comunicación asíncrono.

El municipio necesita enviar y recibir información de las sucursales barriales; en consecuencia se puede tomar el modelo que poseerá un entorno centralizado y un entorno distribuido; este último será replicado en cada una de las sucursales barriales.

Las características y funciones del esquema se pueden resumir en:

- Entorno Centralizado: tiene como función principal soportar el procesamiento de todas las aplicaciones que se ejecutan en las sucursales barriales, en consecuencia deberá poseer tecnología de clustering y alta disponibilidad, soportando balanceo de cargas de trabajo.
- Entorno Distribuido: en este entorno se ubicarán las aplicaciones que utilizarán la arquitectura MOM, de manera distribuida.

Teniendo en cuenta el esquema anterior, en la Fig. 2 se puede observar la propuesta de utilización del portafolio de servidores de aplicación Oracle, el cual soporta perfectamente la solución que se desea implementar en el municipio y está conformado por:

- Oracle WebLogic Server.
- Oracle GlassFish.

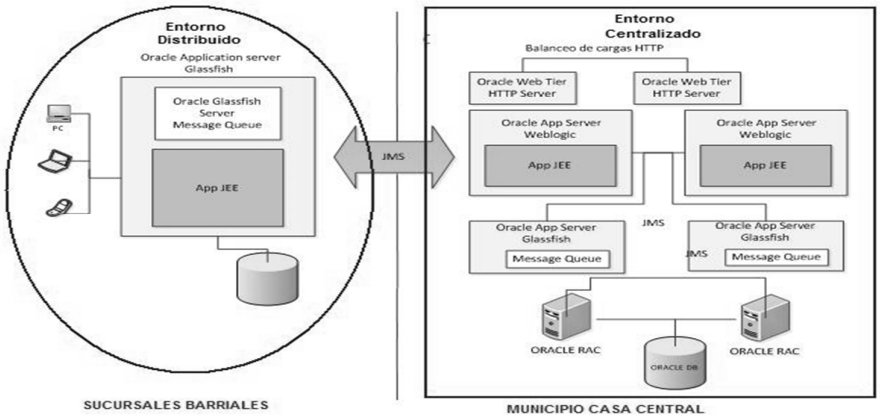

Figura 2. Arquitectura propuesta basada en el portafolio de servidores de aplicación Oracle.

Aquí se puede visualizar que en el ambiente remoto o distribuido, es decir una sucursal barrial, se requerirán únicamente las prestaciones de un servidor de aplicaciones en modo stand-alone, para los cuales se destinará Oracle GlassFish. El servicio que utiliza este último y a través del cual se implementa la especificación JMS de Java, es OpenMQ; sobre el cual se depositarán los mensajes de la información que se genere en la sucursal barrial y que deban de ser transmitidos hacia la Casa Central del Municipio.

En el caso del entorno centralizado, constará de una arquitectura multicapa, con la posibilidad de gestionar balanceo de cargas de trabajo y alta disponibilidad sobre las distintas aplicaciones invocadas.
A través de una arquitectura multicapa se distribuye el procesamiento sobre cada uno de los niveles, en donde el primer nivel tiene como finalidad filtrar las peticiones del protocolo http, a través de la capa de WebTier.
El segundo nivel realiza el procesamiento de las aplicaciones JEE, a través de los servidores de aplicación Oracle WebLogic. Oracle WebLogic Server es el servidor de aplicaciones estratégico de Oracle para la plataforma de Fusion Middleware. Es importante señalar que es en este lugar, donde estarán contenidas las aplicaciones de los diferentes servicios que brinda el Municipio a los ciudadanos, y las mismas pueden combinar diferentes arquitecturas tales como aplicaciones web y aplicaciones móviles.
En particular y pensando en las aplicaciones móviles, se sugiere que las mismas consuman servicios web para su ejecución, logrando de este modo un menor costo de procesamiento y mayor performance en los diferentes dispositivos. Todas las aplicaciones que se encuentren desplegadas en los servidores JEE, tendrán una arquitectura MVC tal como se muestra en la Fig. 3; logrando, de este modo, una integración de la arquitectura MVC dentro de la arquitectura MOM.

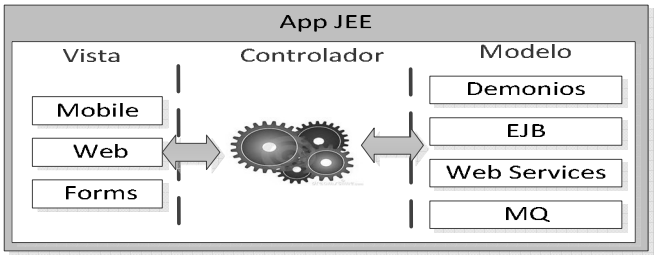

Figura 3. Arquitectura MVC de las aplicaciones integradas en la arquitectura MOM.

En el tercer nivel se utiliza Oracle GlassFish como administrador de cola de mensajes para atender las transacciones provenientes de las sucursales barriales. La funcionalidad que permite efectuar la integración con los entornos distribuidos, es el Bridge Service de Oracle MQ, a través del JMS Bridge Service.

APLICANDO SERVICIOS WEB SOAP Y REST EN FUNCIONALIDADES EXISTENTES

A continuación se describirá la arquitectura actual con la que cuenta, generalmente, cualquier área de sistemas en el ámbito gubernamental y que tenga la responsabilidad de ejecutar aplicaciones transversales críticas como lo son los Sistemas de Administración Financiera (SIAF), Sistema de Administración de Personal (SIAP) -que incluye el proceso de Liquidación de Haberes, Expedientes- y aplicaciones verticales como Impuestos, Catastro, Obras Públicas, Historia Clínica, entre otras. En general, en nuestro país esta situación se da en la práctica a nivel provincial, dónde existe un organismo que aglutina las funciones de ejecución de las aplicaciones anteriormente mencionadas, los denominados Centros de Cómputos Provinciales (CCP).

EJEMPLO DE ARQUITECTURA EXISTENTE

En general los CCP cuentan con mayoría de aplicaciones Cliente/Servidor (dos capas); para este trabajo se tomaron aplicaciones codificadas en Visual Basic 6.0, generadas con una herramienta C.A.S.E. Sin embargo, en la mayoría de los CCP se decidió iniciar un proceso de migración, de algunas aplicaciones, a entorno web. La plataforma tecnológica se encuentra basada en procesadores RISC, con clustering, alta disponibilidad y soportados por el sistema operativo AIX. Con respecto al Administrador de Bases de Datos se utiliza Informix.
En relación a la conectividad se utiliza, a modo de ejemplo, un paquete de internet denominado Integra que permite velocidades de hasta 100 mbps a través de fibra óptica, creándose una red privada virtual (VPN) para el acceso a los usuarios de los distintos sistemas, que se encuentran en las reparticiones fuera del ámbito del CCP (Ver Fig. 4).
Tomando como base la arquitectura existente, a continuación se realizará una propuesta sobre utilización de Servicios Web SOAP.

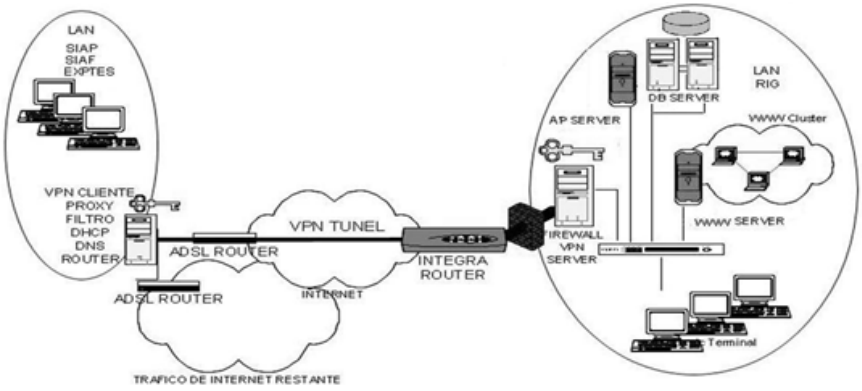

Figura 4. Arquitectura Existente.

PROPUESTA PARA LA UTILIZACIÓN DE SERVICIOS WEB SOAP

Se propone utilizar Servicios Web SOAP para acceder a las funcionalidades que no se encuentran desarrolladas para entorno web. Como caso de estudio se utilizará del Sistema de Liquidación de Sueldos, el cual se ejecuta para todas las reparticiones pertenecientes al Poder Ejecutivo en el CCP. Sin embargo, si una determinada repartición quisiera ejecutar su propia liquidación de haberes, no lo podría hacer; en consecuencia se propone que a través de Servicios Web SOAP se permitan ejecutar las funcionalidades, que se encuentran desarrolladas en Visual Basic 6.0 y de esta forma permitir a las distintas reparticiones, liquidar sus sueldos.

Para lo cual se hace necesario efectuar una invocación de las siguientes funcionalidades que se encuentran alojadas en el Application Server y codificadas en Visual Basic 6.0:

- Ingresar y Controlar Novedades.
- Procesar Liquidación.
- Generar Planillas y Recibos de Sueldos.
- Generar Listados de Retenciones, Embargos y Gremios.
- Procesar Órdenes de Pago y Generar Acreditación.

En la Fig. 5 se puede apreciar el esquema propuesto, basado en la arquitectura actual y descripta en el apartado anterior.

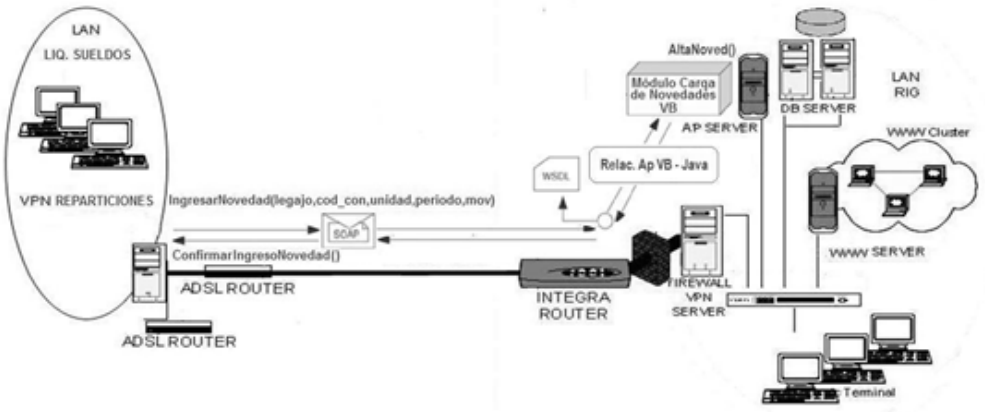

Figura 5. Utilización de funcionalidades a través de Servicios Web SOAP.

Finalmente se deberá efectuar un procedimiento que permita mapear las solicitudes de Servicios Web SOAP con las funcionalidades desarrolladas en Visual Basic 6.0. Una manera de implementar dicho procedimiento podría ser que a partir de una tabla, se permitiera identificar la funcionalidad requerida a partir del servicio invocado por el cliente (Tabla 1). Así lo demuestra su predecesor, el protocolo XML-RPC. Estas tecnologías lograron un éxito limitado antes de ser adaptadas, y se debe a que este tipo de tecnologías, las basadas en modelos RPC (Remote Procedure Call), son más adecuadas para entornos aislados.

Tabla 1. Relación Web Service – Application VB.

Web Service (Client Side)	Param (input)	Application VB (Server Side)	Param (output)
ingresarNovedad()	legajo, cod_con, unidad, periodo, mov	AltaNoved()	confirma
ejecutarLiquida()	fecha,tipo	Liquida()	sinError
procesaAcred()	fecha,nroliq	Acredita()	cantEmpleados,montoTotal

¿POR QUÉ UTILIZAR SERVICIOS WEB BASADOS EN SOAP Y NO IMPLEMENTAR LA SOLUCIÓN A TRAVÉS DE REST?

Al tratarse de aplicaciones que se encuentran en un entorno y dominio específico, en el que las operaciones no sufrirán cambios significativos, la implementación de los Servicios Web basados en SOAP puede resultar muy conveniente. Sin embargo, podría cuestionarse la conveniencia de utilizar dicha solución, si la cantidad de datos a procesar en la liquidación de sueldos es significativa, lo cual induciría a creer que REST resultaría más oportuna.

Sumado a lo anterior, se encuentra el aspecto relacionado a la cantidad de usuarios que manejarían la aplicación, que en este caso de estudio son aproximadamente 3 usuarios por repartición; siendo 82 las reparticiones que utilizarían el sistema. Se estaría trabajando con aproximadamente 250 usuarios, lo cual no representa un número significativo al momento de cambiar la interfaz. Sin embargo, si se tiene en cuenta que se sumarían los usuarios del Sistema Integrado de Administración Financiera (SIAF) o simplemente el sistema de Expedientes, dicho número se multiplicaría por 4, con lo cual se podría concluir que REST resultaría más conveniente.

En la arquitectura actual con la que cuenta el CCP, se puede observar la utilización de Firewalls; hay que tener en cuenta que REST utiliza exclusivamente HTTP como medio de transporte y SOAP añade FTP y fundamentalmente JMS, lo cual puede resultar más conveniente desde el punto de vista de la seguridad, por el uso de la especificación WS-Security.

También puede pensarse que SOAP puede resultar interesante al momento de simplificar el diseño de la interfaz, a partir del ocultamiento de la complejidad del sistema. No obstante la ventaja fundamental, respecto a REST, es que permite la utilización de lenguajes de alto nivel para llamar y para implementar el servicio, que es lo que mejor se adapta a nuestros requerimientos para la solución planteada.

Aunque SOAP puede resultar más complejo al momento de la implementación, además de utilizar más recursos, posee la ventaja de ser fuertemente acoplado lo cual permite ser probado y depurado antes de poner en funcionamiento la aplicación (Erl, 2008).

Otro aspecto importante a tener en cuenta, para el caso de estudio del presente trabajo, es que aunque si bien SOAP no permite la asincronía pura, logra emularla a partir de la utilización del transporte asíncrono lo cual permitirá correr la liquidación (proceso largo y complejo) del lado del servidor y liberar al cliente, hasta que la misma finalice.

Otra característica que representa una ventaja de SOAP, es la fiabilidad en la comunicación, es decir el envío, por única vez, del mensaje.

También se puede señalar que SOAP posee algunas características competitivas respecto de REST, como por ejemplo la conveniencia en la utilización de WSDL, un modelo extensible que permite que los mensajes sean claramente comprensibles, logrando formalidad en el proceso de comunicación.

Finalmente se puede observar que, para el caso de estudio expuesto en el presente trabajo, SOAP se adapta mejor ya que se trata de una tecnología óptima para la integración punto a punto de sistemas internos.

CONCLUSIONES

La gran mayoría de los organismos públicos, en sus distintos niveles jurisdiccionales, poseen distintas plataformas que en la generalidad de los casos no se encuentran integradas. En el presente trabajo se desarrolló una propuesta de implementación de un esquema basado en el portafolio de servidores de aplicación de Oracle, que permiten crear ambientes que se adapten a diferentes requerimientos, tecnológicos y económicos, tanto para las organizaciones públicas como privadas.

A partir de la implementación de esta propuesta se podrán obtener múltiples ventajas operativas sobre las funcionalidades existentes. Como ejemplo podemos citar que los servicios de gestión y cobranza del municipio bajo estudio, podrán ser implementados utilizando tecnologías web y mobile, como así también es posible la interacción con otros servicios de pago, mediante la implementación de Servicios Web.

Con la implementación de esta arquitectura se provee una forma segura y capaz de solucionar las necesidades actuales del municipio, pero fundamentalmente también la de los usuarios, destinatarios finales de toda solución de Gobierno Electrónico.

Con la implementación de MVC, todas las aplicaciones desplegadas, en los servidores, contarán con una separación, a nivel de capas de software de las aplicaciones, lo que permitirá gozar de las ventajas que brinda este tipo de arquitectura.

Por otra parte la utilización de SOA y particularmente de los Servicios Web resulta un desafío interesante pero a la vez muy productivo para implementar soluciones de este tipo en el ámbito de la Administración Pública. Esto se encuentra fuertemente ligado a la gran cantidad de plataformas y arquitecturas de software que poseen los gobiernos en los tres niveles jurisdiccionales: municipal, provincial y nacional. Dicha multiplicidad, en reiteradas ocasiones, atenta contra objetivos y estándares a los que apunta la formalización de los procesos de Gobierno Electrónico, como es el caso de la interoperabilidad.

La utilización de arquitecturas avanzadas de software, y especialmente SOA, en el ámbito gubernamental permitirá obtener una innumerable cantidad de ventajas que van desde la actualización de aplicaciones que se encontraban basadas en arquitecturas de dos capas para pasar a arquitecturas multicapas con pequeños cambios en el diseño arquitectónico a nivel de software, hasta la implementación de Servicios Web (SOAP o REST) dependiendo de los entornos de las aplicaciones que se encuentran en producción.

Finalmente se puede concluir, que aunque si bien SOA puede resultar una herramienta interesante para aumentar los servicios que brinda un organismo público a los ciudadanos, se debe tener en cuenta ciertos aspectos propios del ámbito gubernamental, lo cual requiere tomar los recaudos necesarios para evitar inconvenientes que están relacionados con el tiempo de desarrollo, la curva de aprendizaje, la relación entre los recursos afectados y los costos demandados por la implementación de SOA.

REFERENCIAS Y BIBLIOGRAFÍA

Bahit, E., "POO y MVC en PHP", 2011. Disponible en: http://www.bubok.es/libros/205199/POO-y-MVC-en-PHP. Última visita: Abril de 2013.

Castro, M., J. Farfán, V. Sánchez Rivero, A. Cándido, "Las aplicaciones transversales en Gobierno electrónico". Actas de las V Jornadas de Investigaciones en Facultades de Ingeniería del NOA; Tomo I. Salta, 2009.

Castro, M., V. Sánchez Rivero, E. Reinoso, M. Aparicio, F. Aragón, L. Cazón, "Servicios digitales comunes reutilizables para gobierno electrónico". Actas del 6º Simposio de Informática en el Estado y 41º JAIIO; La Plata, Argentina, 2012.

Coulouris G., J. Dollimore & T. Kindberg, "Distributed Systems Concepts and Design". Third Edition. Ed. Addison-Wesley, 2001.

Erl, T., "SOA Principles of Service Design". Ed. Prentice Hall, 2008.

Hansen, M., "SOA Java Using Web Services". Ed. Prentice Hall, 2007.

Los Santos Aransay, A., "Revisión de los Servicios Web SOAP/REST: Características y Rendimiento", 2009. Disponible en: http://www.albertolsa.com/?p=187. Última visita: Abril de 2013.

Marsili, D., "¿Qué es SOA, la arquitectura orientada a servicios?", 2007. Disponible en: http://www.iprofesional.com/notas/46399-Que-es -SOA-laarquitectura-orientada- a-servicios.html. Última visita: Abril de 2013.

Pérez Morales, E. A., "Arquitectura MOM (Message-Oriented Middleware) basada en el portafolio de servidores de aplicación Oracle", 2004. Disponible en: http://www.oracle.com/technetwork/es/articles/soa/arquitectura-mom-integracion-server-426198-esa.html. Última visita: Abril de 2013

Wikipedia, "Servicio Web", 2013. Disponible en http://es.wikipedia.org/wiki/Servicio_web. Última visita: Abril 2013.