**TALLER 1 BANCO ABC**

**SERGIO ANDRES GRACIA CARDENAS**

**JEISON ANDRES CAMACHO CACAIS**

**EDGAR ANCIZAR ORTIZ GONZALEZ**

**LUIS FELIPE VANEGAS PINEDA**

**JORGE ALBERTO PUENTES NIÑO**

**PONTIFICIA UNIVERSIDAD JAVERIANA**

**ESPECIALIZACIÓN EN ARQUITECTURA EMPRESARIAL DE SOFTWARE**

**PATRONES**

**BOGOTÁ**

**2016**

**TALLER 1 BANCO ABC**

**SERGIO ANDRES GRACIA CARDENAS**

**JEISON ANDRES CAMACHO CACAIS**

**EDGAR ANCIZAR ORTIZ GONZALEZ**

**LUIS FELIPE VANEGAS PINEDA**

**JORGE ALBERTO PUENTES NIÑO**

**TRABAJO PRESENTADO A:**

**German Alonso Suarez Guerrero**

**PONTIFICIA UNIVERSIDAD JAVERIANA**

**ESPECIALIZACIÓN EN ARQUITECTURA EMPRESARIAL DE SOFTWARE**

**PATRONES**

**BOGOTÁ**

**2016**

Solución Taller 1 Propuesta servicio Banco ABC

1. Modelo de la solución entregado en ArchiMate se puede verificar en archivo **ArchiTalle1.archimate,** Los patrones aplicados se detallan más adelante para garantizar que la prestación del servicio no se vea interrumpida para crear un nuevo tipo de factura es necesario balancear carga y tener más de un servidor con el fin de hacer el cambio del micro servicio en una máquina, mientras la otra esta respondiendo las solicitudes y luego se hace lo contrario, se debe tener en cuenta que los servicios se modelaron pensando en el bajo acoplamiento y al incluir un nuevo proveedor de servicios públicos este no se ve afectado. Solo se ve afectado el incluir una referencia al servicio para poder ser invocado.
2. Inventario de servicio es de tipo **empresarial** ya que los servicios propuestos están disponibles para toda la organización sin necesidad de tener un dominio especifico, el inventario se encuentra en el archivo **InventarioWSTaller1.xlsx**
3. Patrones aplicados
4. **Diseño estandarizado de contrato de servicios:**

Este patrón es implementado en la solución propuesta ya que las entidades comunes ya sea mensajes Request o mensajes Response son reutilizados esto lo podemos ver en los servicios:

-Realizar Pago

-Compensación

-NotificarCompensación

En los cuales el mensaje de Request es el mismo y su contrato reutiliza los mensajes.

Este concepto también es aplicado en los mensajes de Response que utilizan el mismo concepto y resultado:

* AutenticarUsuario
* VerificarSaldo
* RealizarPago
* Compensacion
* NotificacionCompensacion
* CrearConvenio
* EliminarConvenio

Esto se aplicó con el fin de no tener que hacer transformaciones en los consumos de los servicios.

1. **Diseño de bajo acoplamiento**

Ya que cada servicio u operación se encarga de realizar su propia tarea lo cual permite que cualquier otra aplicación o servicio pueda consumir otro servicio sin necesidad de generar dependencias.

Así como se hizo en el servicio **Compensacion** ya que el llamado se hace sobre la compensación y el internamente determina si se debe notificar al proveedor o si realiza la compensación de una vez.

1. **Diseño de reusabilidad de servicios**

De los servicios construidos podemos decir que algunos pueden ser reusables en la compañía y/o utilizados en otros procesos internos y externos a la organización como lo son:

* AutenticarUsuario
* VerificarSaldo
* RealizarPago
* Compensacion

Estos servicios pueden ser utilizados por la compañía ya que no están atados a algún proceso en específico.

1. **Diseño de servicio sin estado**

Todos los servicios construidos no mantienen estados, el que se encarga de mantener la información que se necesite es el consumidor o cliente en su forma de implementación SOAP se reciben unos parámetros y se retorna un mensaje con el resultado del proceso sin mantener algún estado para consumo de otros WEB Services.

1. **Diseño de descubrimiento de servicios**

Se realiza el inventario en el cual están registrados los servicios con su nombre, funcionalidad y con su respectivo WSDL y ruta de descubrimiento para permitir el re uso dentro de la organización.

1. **Diseño de composición de servicio**

La composición de servicio se implementó en los servicio **ConsultaPagoServicio**, **RealizarPago y Compensación** ya que para el cliente solo se exponen y consumen los 3 servicios mencionados anteriormente pero dentro de cada uno de ellos se verifica el tipo de factura y se re direcciona a los servicios del proveedor ya sea REST o SOAP.

1. La generación de artefactos se puede evidenciar en la carpeta **Artefactos** en la cual se evidencia todos los contratos WSDL creados antes de realizar la implementación de los mismos, estos están creados con el formato XML.
2. El Demo se realizó en plataforma .Net, Los servicios tanto del proveedor como los servicios creados para la solución se crearon en Java y se exponen en un servidor con las siguientes URLS:

http://104.197.5.244:8086/ConsultaSaldoSrv/services/ConsultaSaldo?wsdl

http://104.197.5.244:8086/CompensaIISrv/services/CompensaII?wsdl

http://104.197.5.244:8086/SeguridadSrv/services/Seguridad?wsdl

http://104.197.5.244:8086/RealizarPagoSrv/services/RealizarPago?wsdl

http://104.197.5.244:8086/NotificarCompensacionSrv/services/NotificarCompensacion?wsdl

http://104.197.5.244:8086/AdministrarConveniosSrv/services/AdministrarConvenios?wsdl

http://104.197.5.244:8086/NotificarUsuarioSrv/services/NotificarUsuario?wsdl

http://104.197.5.244:8086/OpcionesPagoSrv/services/OpcionesPago?wsdl

http://104.197.5.244:8086/VerificarSaldo/services/VerificarSaldo?wsdl

Ver archivo Adjunto Manual Dumi entrega.docx en el cual se detalla cómo está construido el Dumi y como se implementaron los Web Services orientados por el negocio,

1. La dirección de GITHUB en la cual se encuentran todos los documentos del proyecto solución es la siguiente:

<https://github.com/Sergini46/EntregaTaller1BancoABC>