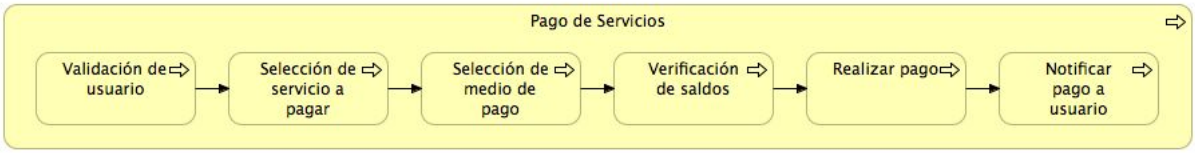
Modelado y Validación de Arquitectura

De la descripción del siguiente escenario, modelar e implementar una solución utilizando una aproximación orientada a servicios, basándose en los principios de diseño de servicios, patrones y estrategias de diseño.

Escenario
El Banco ABC está realizando varios proyectos de actualización tecnológica los cuales le permiten ofrecer sus productos financieros de manera más ágil y de ésta forma responder a nuevas necesidades del mercado.

El Banco acaba de firmar una alianza estratégica con diferentes proveedores de servicios públicos (Agua, Gas, Luz, Telefonía) o también llamados **convenios**, para permitir a los clientes del banco a través de los diferentes canales de servicio (Cajeros Automáticos, Cajero de Oficina, Teléfono, Portal Web, Aplicación Móvil) permitir el pago de los mismos.

El proceso de vista al usuario se puede definir de la siguiente manera a través de las siguientes tareas:



Cada uno de los proveedores de servicios públicos ofrece los mecanismos de interacción tecnológica necesarios para que el Banco pueda ejecutar las acciones de pago.

Cada proveedor utiliza tecnologías diferentes para ofrecer sus servicios.

El Banco ABC quiere tener la posibilidad de adicionar nuevos convenios con otros proveedores de servicios de manera ágil, o incluso la posibilidad de terminar/eliminar los convenios existentes sin que esto represente indisponibilidad del servicio.

Se llegó a un acuerdo de las capacidades/primitivas básicas que se deben soportar para cada convenio:

- Consulta de saldo a Pagar
- Pago del Servicio
- Compensación de pago (Opcional)

Principalmente el banco necesita un conjunto de servicios que representen sus necesidades internas de negocio, lo cual les permite desacoplar los servicios de los proveedores y así no depender de sus detalles.

La definición de los servicios se encuentra aquí:
https://github.com/germansua/UJaveriana-AES-ModVal/tree/master/workshops

Criterios de Aceptación/Calificación
Los siguientes son los puntos que se van a calificar, cada item se califica con 1 punto:

- Proyecto publicado en GitHub, junto con su documentación
- Justificaciones de arquitectura sobre cada una de las decisiones tomadas. E.g. Patrones, estilo de realización de servicios, trade-off, estilos de arquitectura, etc.
- Generación de los artefactos de servicios identificados para el inventario y blueprint. Se debe tener en cuenta que cada artefacto (esquemas, contratos y políticas) deben ser independientes. Cada artefacto se debe diseñar bajo la premisa de "*Contrato Primero*". Se puede utilizar RAML/OpenAPI/JSON Schema, Thrift IDL, GRPC, … (No se admiten servicios SOAP)
- Creación del inventario de servicio y registro de los mismos
- Creación de un servicio basado en el patrón **Intermediate Routing** que realice enrutamiento basado en datos, para poder enrutar las peticiones al tipo de servicio adecuados dependiendo del convenio
- Implementar los servicios que soportan el proceso
- Realizar la composición de los servicios usando cualquiera de las aproximaciones de composición (basado en Orquestación o Coreografía, se puede utilizar FUSE, Camel Kafka, ActiveMQ, RabbitMQ pero está prohibido utilizar BPEL y BPMS)
- Generar servicios que sean autocontenidos e independientemente desplegables en Docker
- Implementar un servicio que modele el patrón de API Proxy
- El API Proxy debe ser capaz de trabajar con diferentes representaciones de los datos, tanto para los request como para los response. De esta manera se pueden tener requests/responses XML/JSON