Reto Técnico Diagrama 4C

1. Diagrama de Contexto

En el diagrama de Contexto divido por capas los sistemas, donde pongo énfasis en la parte normativa de actores, canales y servicios Bancarios.

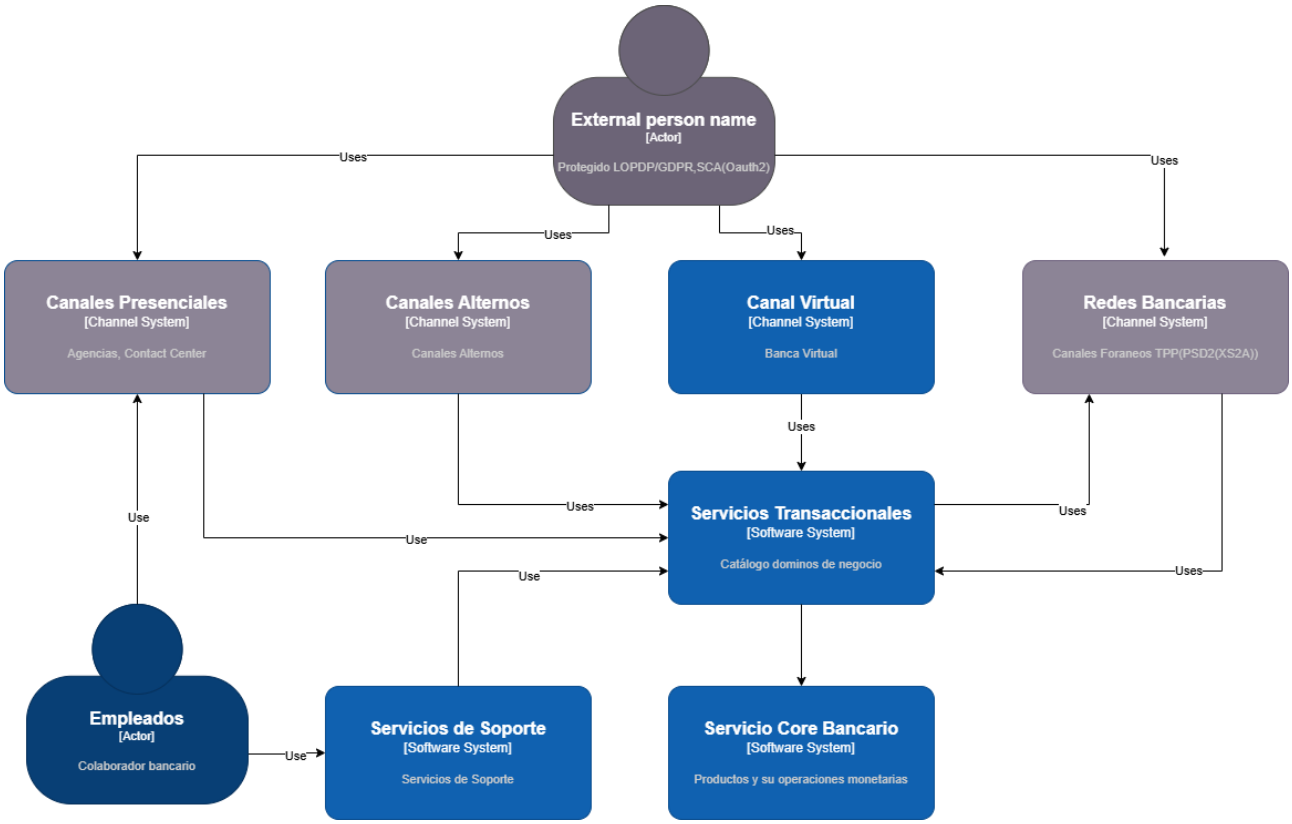

Diagrama de Contexto

1.1. Actores

1.1.1. Cliente

Quién es dueño de los productos bancarios y realiza las operaciones bancarias usando los servicios bancarios.

Normativas

Normativa LOPDP (Ley Orgánica de Protección de Datos Personales) y normativa GDPR (General Data Protection Regulation) para proteger su información personal, mediante SCA (Strong Customer Authentication) específicamente con OAuth2 (Open Authorization).

1.2. Canales Bancarios

1.2.1. Canal Virtual

Banca Mobile y Web (Banca Internet).

1.2.2. Canales Alternos

Canales electrónico: Atm propios, PoS Propios, Ivr, Voicebot, Chatbot, Kioskos, etc.

Normativas

PCI-DSS (Payment Card Industry Data Security Standard).

1.2.3. Redes Bancarias

Redes de servicios bancarios asociados como son: Franquicias de tarjetas de crédito, Remesadoras, Transferencias Swift, Sistema Interbancario del BCE, B2B (Businesses to Businesses) con distintas plataformas de cobro o servicios de cobro a instituciones, Fintech como son billeteras móviles y pagos en línea.

Normativas

Mormativa PSD2 (Payment Services Directive 2) para proteger al consumidor de transacciones no autorizadas por TPP(Third-Party Provider) con normativa (XS2A (Access to Accounts)) PCI-DSS (Payment Card Industry Data Security Standard)

1.2.4. Canales Presenciales

Agencias y Contact Center donde el cliente tiene una interrelación personalizada.
(Este canal no se lo tratará ya que no se requiere para el ejercicio)

1.3. Servicios Bancarios

1.3.1. Servicios Transaccionales

Son las distintas interfaces que exponen servicios bancarios que se pueden agrupar en dominios de negocio.

1.3.2. Servicios de Soporte

Son las plataformas de soporte a los productos y servicios financieros.

1.3.3. Servicio Core Banking

Es el sistema central de los productos bancarios, y todas las operaciones transacionales sobre los mismos.

2. Diagrama de Contenedores

En el diagrama de Contenedores pongo énfasis en los productos y servicios bancarios.

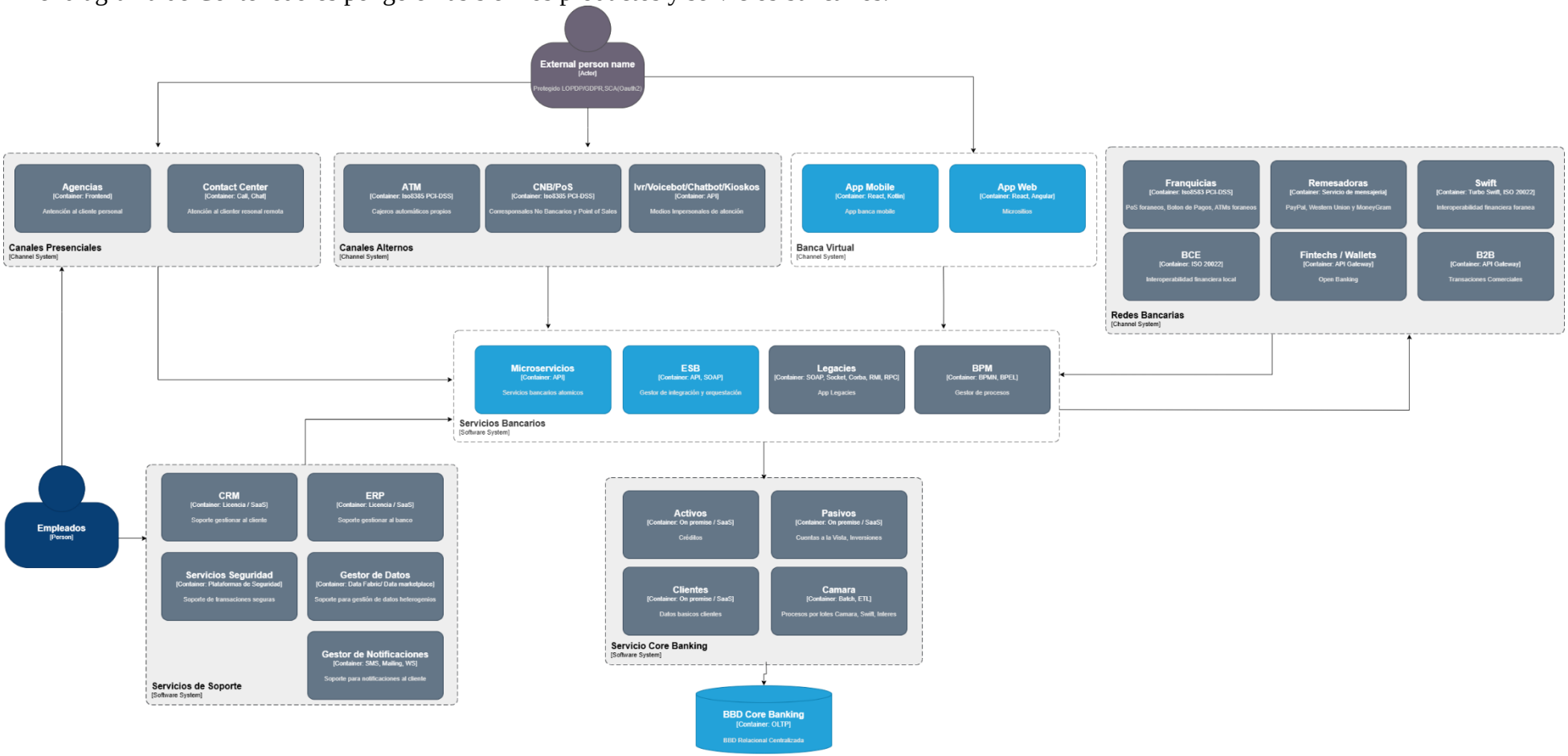

Diagrama de Contenedores

1.1. Banca Virtual

1.1.1. Banca Mobile

Se tiene los servicios de transferencias,

Dominios

Onboarding, gestión de cuentas, préstamos y créditos, inversiones, transferencias, pagos, consulta de movimientos,

1.1.2. Banca Web (Internet)

Se tiene los servicios de transferencias,

Dominios

Gestión de cuentas, préstamos y créditos, inversiones, transferencias, pagos, consulta de movimientos.

1.2. Servicios Bancarios

Capa intermedia de Integración entre servicios transaccionales, servicios de soporte y redes externas.

1.2.1. Microservicios

Se tiene los servicios atómicos, que cumplen puntualmente un propósito con el mínimo de acoplamiento y alta cohesión con toros servicios en una arquitectura de llamadas por eventos, cada microservicios garantiza las propiedades ACID de su propia data.
Generalmente se desarrollan microservicios cuando se crea un nuevo servicio bancario.

1.2.2. Enterprise Service Bus

Es una capa de integración donde se orquesta diferentes tipos de servicios (BPEL) con distintos tipos de protocolos, CISC, RPC, COM, RMI, CORBA, SOAP, API.
Una de sus principales características es que maneja a través de MQ, colas de mensajes para transacciones asíncronas.

1.2.3. App Legacies

Hay muchas aplicaciones, y plataformas que conviven con servicios modernos, que no se han podido migrar por su confiabilidad y/o su complejidad. Pero que todavía son usadas y necesitan integrarse con otros legacies o micros servicios modernos.

1.2.4. BPM

El Gestión de Procesos de Negocio crea flujos BPMN donde se automatiza ciertos procesos de negocio. El BPM se lo puede considerar una aplicación legacy ya que RPA, ML, IA y plataformas Low Code, están remplazando.

1.3. Servicios de Soporte

Capa intermedia de servicios no transaccionales que pan soporte a las transacciones o a servicios no transaccionales.

1.3.1. Servicios de Seguridad

Posiblemente el servicio de no transaccional más importante, que permite garantizar la confiabilidad, integridad y disponibilidad de la información. Donde existen servicios como el de autenticación, validación biométrica, control, monitoreo de aplicaciones, etc

1.3.2. CRM

El gestión de relaciones con clientes es una pieza clave para el negocio, para campañas de productos y servicios bancarios conde de clusteriza a los clientes para brindar un mejor servicio de calidad.

1.3.3. Gestor de Notificaciones

La gestión de notificación es muy importante para que entre otras cosas se informe a tiempo sobre una suplantación de identidad o una transacción fraudulenta.

3. Diagrama de Componentes

En el diagrama de Componentes donde pongo énfasis en las justificaciones técnicas.

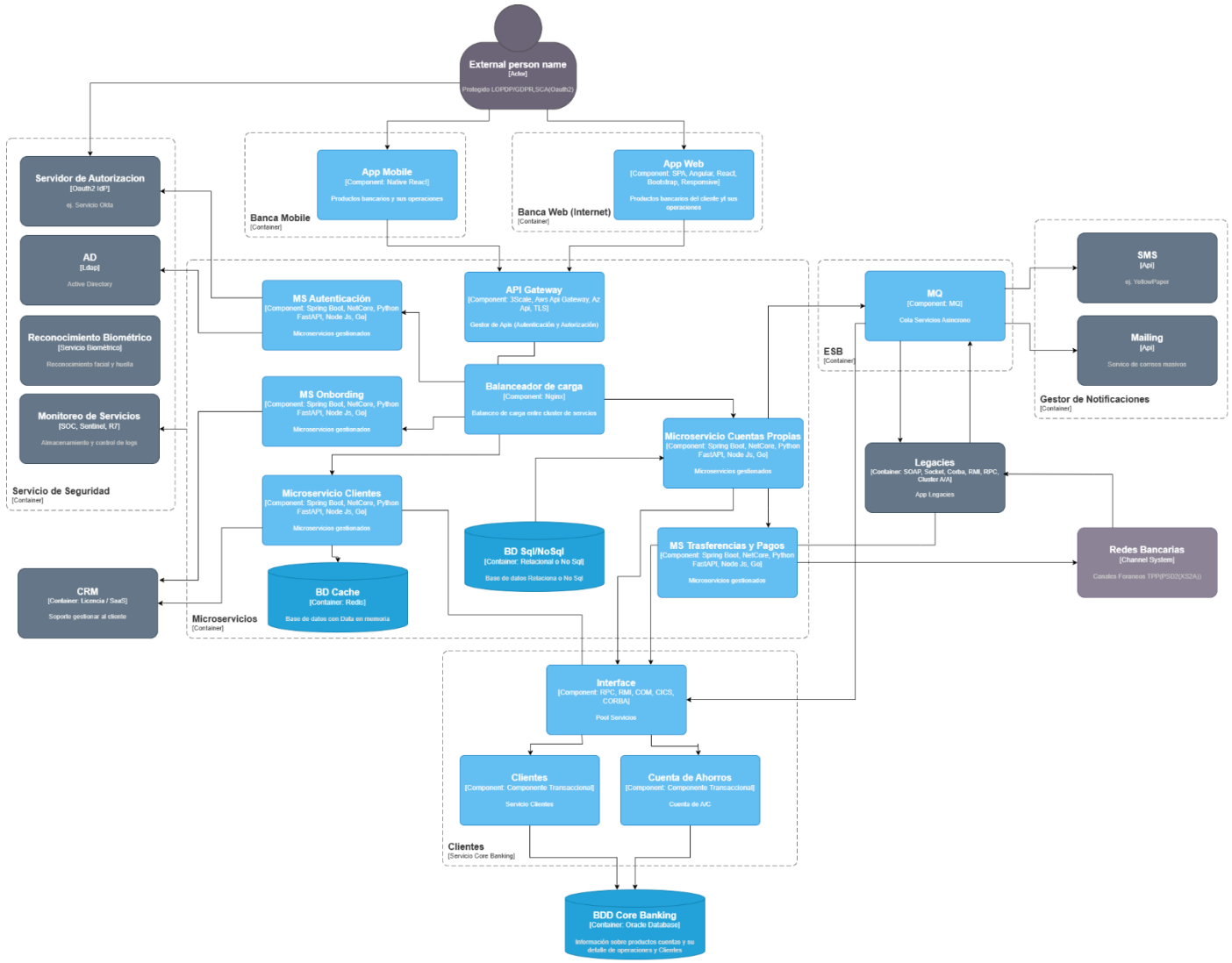

Diagrama de Componentes

1.1. Servicios SaaS

Se recomienda para servicios que no son parte del giro de negocio o donde la información no es confidencial usar servicios SaaS, en el ejercicio planteado se puede contratar servicios de validación biométrico, de notificación masiva de correos y mensajes de texto.

1.2. Tecnologías Front-end

1.2.1. App Mobile

La aplicación mobile, recomiendo Native React ya que es independiente de la plataforma sirve tanto para Android, iOS, Windows, macOS, Linux, para cualquier navegador Web.
La alta disponibilidad se la puede manejar mediante un Api Gateway distribuido en 2 regiones diferentes. También existen mecanismos de negocio como el Stand-In para manejo de cupos, para usuarios que necesitan hacer transacciones fuera de línea.

1.1.1. App Web (Internet)

Se usaría tecnología SPA, responsive web design, con una arquitectura micrositio se puede usar varios tiposde librerías para sacar el mejor provecho de cada una: React, Angular, Vue,js, Django.
La alta disponibilidad se la maneja con sitios en al menos 2 regiones, con micrositios que manejen servicios elásticos.

1.3. Tecnologías Middleware

1.3.1. Micro servicios

Los micro servicios deben exponer API Rest, se puede construir con frameworks livianos como son Spring Boot, Micronaut con Kotlin, Go, en lo particular me gusta Fast API con Python.
El acceso a datos se puede usar una arquitectura CQRS (Command Query Responsibility Segregation) para mejorar los tiempos
Crear una arquitectura orientada a evento, para mantener desacoplada la interacciones.
Usar mecanismos asíncronos para llamadas a servicios de baja latencia en el tiempo de respuesta de la transacción financiera.
La alta disponibilidad se puede manejar con balanceadores tipo Nginx y microservicios gestionados con crecimiento elástico automático.

1.4. Back-End

1.4.1. Base de Datos Relacionales

Base de datos ligeras como PostgrSql, MySql, para el core bancario puede ser Oracle o SqlServer con arquitectura Mirror para la alta disponibilidad,

1.4.2. Base de Datos in Memory

Redis es la más popular, para datos frecuentes de Clientes.

1.1.1. Base de Datos NoSql

Base de Datos Mongo DB

1.1.2. ETLs

Manejo de datos con Az Data Factory y AWS Data Pipe Line

1.1.3. Gestion de Datos

Exposición de datos Snowflake, Data Fabric

1.1.4. Gestion de Datos de Auditoría

Sentinel y Data Monitor de Azure es la opción que más conozco.

1.1.5. Core Banking

La alta disponibilidad para un Core Banking on Premise monolitico se puede manejar con un esquema Activo/Pasivo.