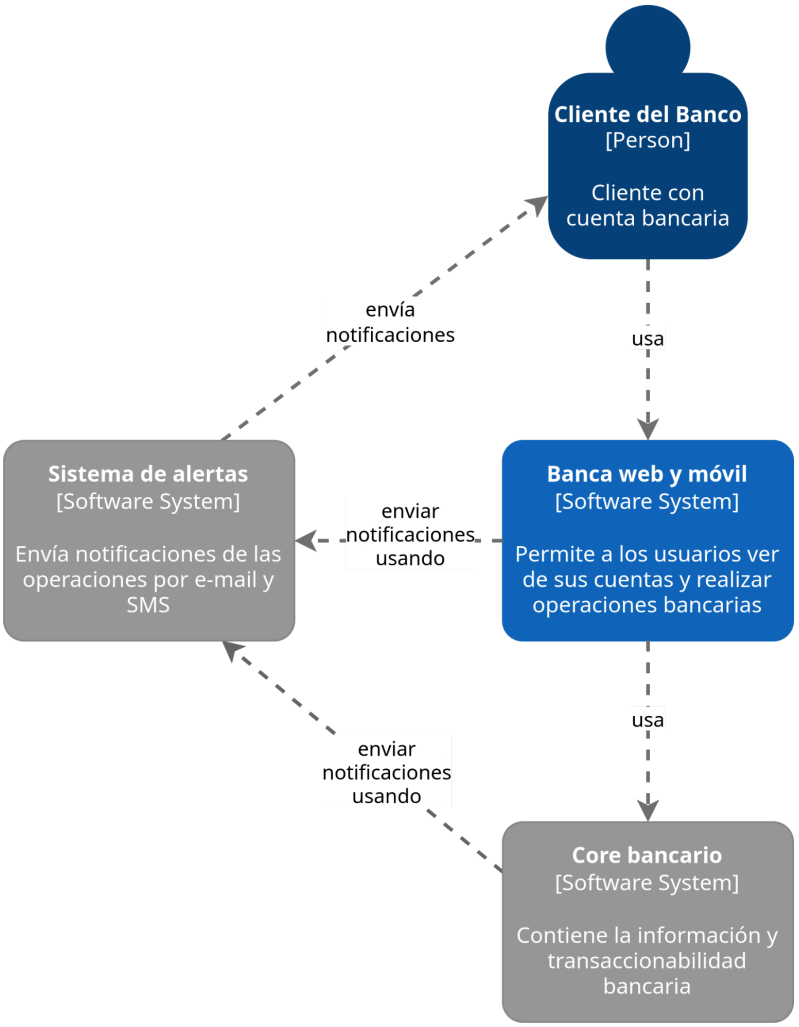
Reto de Arquitectura DEVSU

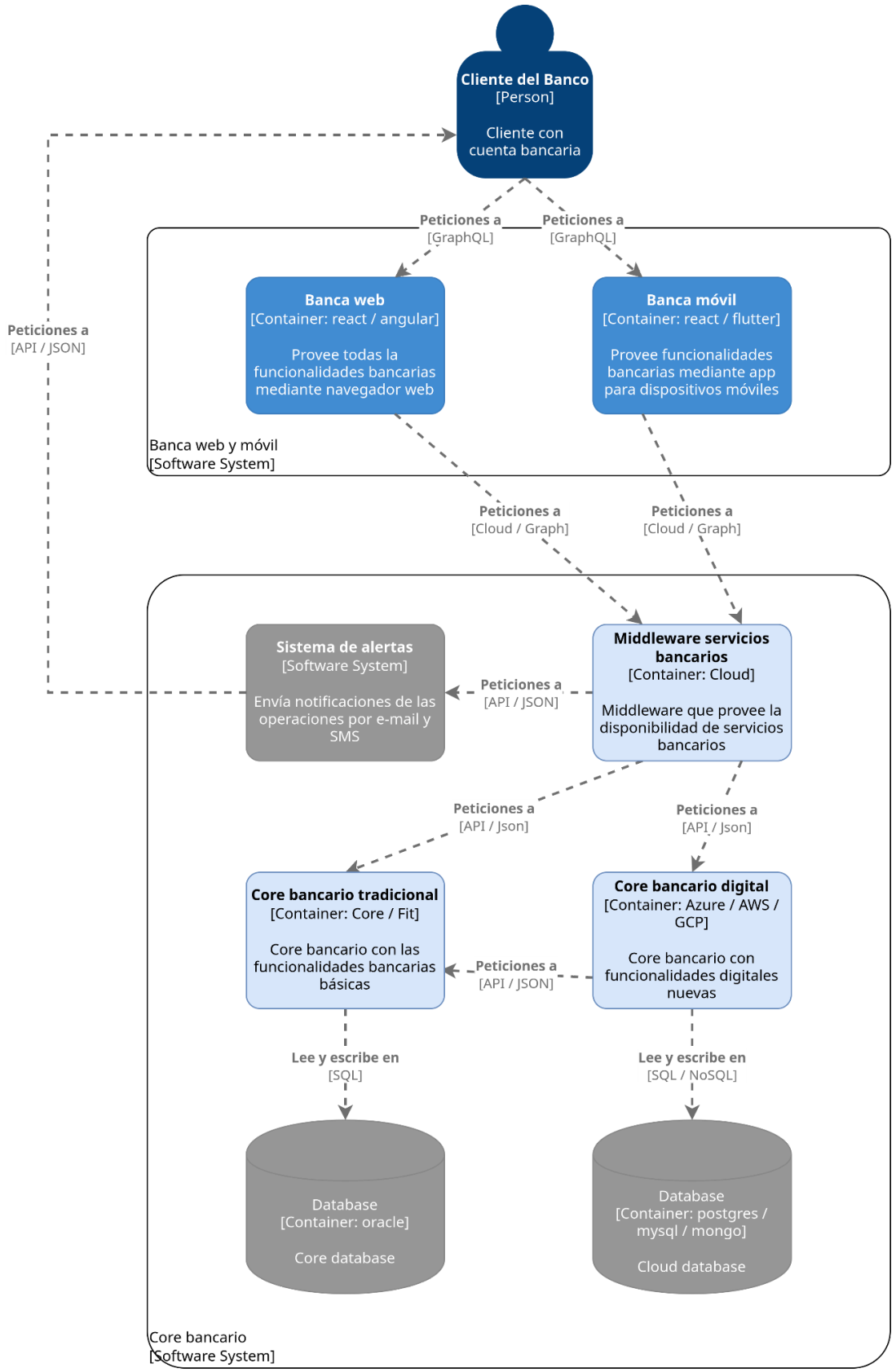
Realizado por: Ing. Carlos Andrés Ordóñez Parra
C.I.: 0104159215
Fecha: 2024/08/17

1. Diseñar una Arquitectura de Integración de Alto Nivel

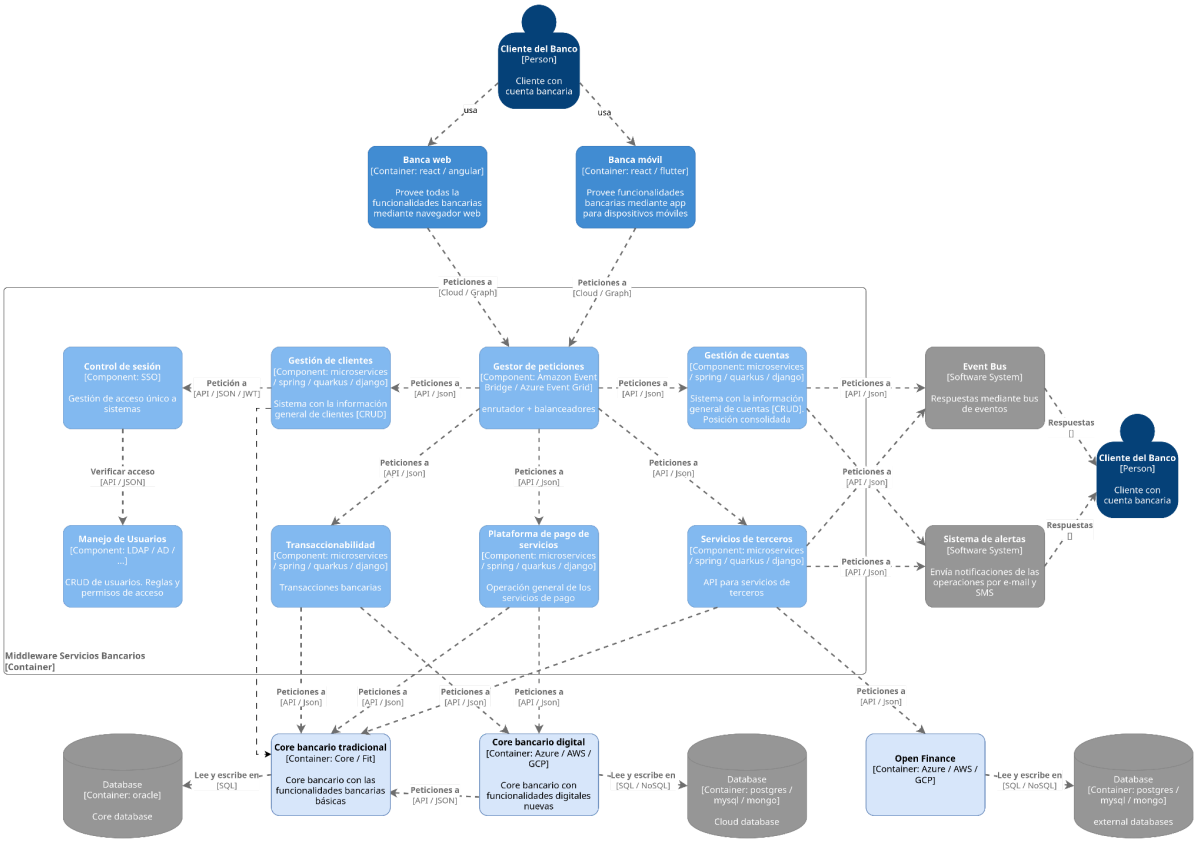
C4 Contexto



C4 Contenedores



C4 Componentes



2. Especificar patrones de integración y tecnologías a utilizar

Encuentro la necesidad de crear un **middleware** de los servicios ofrecidos, de preferencia sobre una plataforma cloud, que contenga:

- API gateway
- Sincronización de datos, tratando de que todos los servicios en el middleware hablen el mismo "idioma"
- SOLID, para establecer servicios estándar, flexibles y de fácil mantenimiento, para su uso desde cualquier servicio
- Bajo acoplamiento y alta cohesión
- Creación de orquestadores para ejecución de servicios con múltiples conexiones a otros servicios, para que sean realizados de forma organizada y ordenada
- Creación de adaptadores, encargados de la transformación de datos y sincronizarlos a los servicios del middleware
-

3. Abordar requisitos de seguridad, cumplimiento normativo, ley orgánica de protección de datos personales.

- ☐ Ley Orgánica de Economía Popular y Solidaria y del Sector Financiero Popular y Solidario: Regula aspectos relacionados con las instituciones financieras populares y solidarias, así como los derechos y deberes de sus miembros y clientes.
- ☐ Ley Orgánica para la Regulación de Crédito y la Protección del Cliente Bancario: Establece los derechos de los clientes bancarios y las obligaciones de las instituciones financieras, incluyendo transparencia en la información y condiciones contractuales.
- ☐ Código Orgánico Monetario y Financiero: Es una ley clave que regula todo el sistema financiero en Ecuador, incluyendo aspectos de supervisión, regulación y control de instituciones financieras.
- ☐ Normativa de Prevención de Lavado de Activos y Financiamiento del Terrorismo: Establece medidas para prevenir que el sistema financiero sea utilizado para el lavado de dinero y el financiamiento de actividades ilícitas. Los bancos deben realizar la debida diligencia del cliente para verificar la identidad y el origen de los fondos.
- ☐ Reglamento de la Superintendencia de Bancos y Seguros: Contiene directrices específicas que los bancos deben seguir al abrir cuentas, incluyendo la documentación necesaria y los procesos de verificación.
- ☐ Reglamento para la Aplicación de la Ley Orgánica de Protección de Datos Personales: Asegura que los bancos manejen la información personal de los clientes de manera adecuada y segura, protegiendo su privacidad.

4. Proponer estrategia para garantizar alta disponibilidad y recuperación ante desastres.

Para garantizar alta disponibilidad y recuperación ante desastres, se necesita una estrategia que abarque redundancia, resiliencia, monitoreo y respuesta rápida a incidentes. Aquí propongo una estrategia (como ejemplo usando AWS, vale recalcar que Google Cloud y Azure disponen de herramientas y funcionalidades similares):

Arquitectura de Alta Disponibilidad (HA)

Diseño Multi-AZ (Availability Zones):

- Implementar instancias de aplicación y base de datos en varias zonas de disponibilidad (AZ) dentro de una misma región.
- Usar Elastic Load Balancers (ELB) para distribuir automáticamente el tráfico entre múltiples instancias en diferentes AZs.

Auto Scaling:

- Configurar grupos de autoescalado (Auto Scaling Groups) para que se lancen nuevas instancias automáticamente si la demanda aumenta o si una instancia falla.

Amazon RDS Multi-AZ Deployment:

- Utilizar Amazon RDS con la opción de despliegue Multi-AZ para las bases de datos. Esto crea una réplica en una AZ diferente y permite conmutación por error automática en caso de fallo en la AZ principal.

Almacenamiento Duradero:

- Usar Amazon S3 con almacenamiento estándar para datos críticos.

Estrategia de Recuperación Ante Desastres (DR)

Backups Automatizados y Replicación:

- Configurar copias de seguridad automáticas diarias de bases de datos con Amazon RDS y habilitar la replicación transaccional a una región secundaria.
- Realizar copias de seguridad regulares de EBS, S3 y otros datos críticos y almacenarlos en otra región utilizando AWS Backup.

Plan de Recuperación de Región Cruzada (Cross-Region Recovery):

- Diseñar una arquitectura de recuperación en una región secundaria. Usar Amazon Route 53 para la conmutación por error global. Configurar políticas de conmutación por error que apunten a la región secundaria si la primaria no responde.
- Desplegar instancias mínimas en la región secundaria y activar el escalado automático en caso de un evento de desastre.

Infraestructura como Código (IaC):

- Utilizar herramientas como AWS CloudFormation o Terraform para definir y gestionar la infraestructura como código.

Pruebas Regulares de DR:

- Realizar pruebas regulares de la estrategia de recuperación ante desastres para asegurarte de que todo funcione como se espera.

Monitoreo y Alertas

AWS CloudWatch:

- Implementar un monitoreo completo utilizando Amazon CloudWatch para vigilar el rendimiento, la salud de las instancias, y otros indicadores clave. Configurar alarmas para notificar a los equipos en caso de problemas potenciales.

AWS Config y GuardDuty:

- Usar AWS Config para monitorear los cambios en la configuración y GuardDuty para la detección de amenazas.

Logging Centralizado:

- Implementar un sistema de logging centralizado con AWS CloudTrail y CloudWatch Logs para recolectar y analizar logs de todos los componentes de tu sistema.

Seguridad

Protección DDoS:

- Implementar AWS Shield y AWS WAF para proteger la infraestructura contra ataques DDoS y otras amenazas en la capa de aplicación.

5. Proponer estrategia de integración multicore.

Aquí tienes una estrategia simple y de alto nivel para la integración multicore de sistemas bancarios en la nube:

- ☐ Identificación y Clasificación de Tareas
  - ☐ División de Procesos Bancarios
  - ☐ Clasificación por Prioridad
- ☐ Asignación Inteligente de Núcleos
  - ☐ Optimización de Localidad de Datos
  - ☐ Afinidad de Núcleo
- ☐ Escalabilidad y Autoescalado

    - ☐ Escalado Automático en la Nube
    - ☐ Balanceadores de Carga
- ☐ Sincronización y Consistencia de Datos
    - ☐ Gestión de Transacciones Concurrentes
    - ☐ Uso de Primitivas de Sincronización
- ☐ Monitoreo y Optimización Continua
    - ☐ Monitoreo de Rendimiento
    - ☐ Ajustes Dinámicos
- ☐ Seguridad Multicore
    - ☐ Segmentación de Tareas Sensibles
    - ☐ Gestión de Accesos y Permisos
- ☐ Pruebas y Validación
    - ☐ Pruebas de Carga y Estrés
    - ☐ Validación Continua
- ☐ Documentación y Capacitación
    - ☐ Documentación Detallada
    - ☐ Capacitación del Equipo

6. Delinear un enfoque para la gestión de identidad y acceso a todos los sistemas.

Establecer el SSO de la empresa para gestionar permisos y políticas de acceso a los recursos (por ejemplo un protocolo estándar de autenticación OAuth).

En caso de no disponerlo, se puede establecer el uso del disponible en la plataforma cloud y su integración a la gestión de permisos internos de la institución.

7. Diseñar una estrategia de API internas y externas, bajo estándares de industria respecto a la mensajería.

Se debe establecer un estándar para la comunicación entre servicios, para lo que se puede volver a mencionar

- SOLID, para establecer servicios estándar, flexibles y de fácil mantenimiento, para su uso desde cualquier servicio
- Bajo acoplamiento y alta cohesión
- Creación de adaptadores, encargados de la transformación de datos y sincronizarlos a un "idioma" general para todas las API

8. Proponer un modelo de gobierno de APIs y microservicios.

Se debe establecer un plan que contenga:

- ☐ Establecimiento de Estándares de Diseño
- ☐ Políticas de Seguridad
- ☐ Ciclo de Vida de las APIs
- ☐ Monitoreo y Métricas

- ☐ Documentación y Soporte
- ☐ Gestión de Cambios
- ☐ Reutilización y Catalogación
- ☐ Cumplimiento y Auditoría
- ☐ Gobernanza y Roles
- ☐ Comunicación y Capacitación

9. Proponer un plan de migración gradual que minimice el riesgo operativo.

Preparación
- Inventario de APIs: Identificar todas las APIs y servicios actuales.
- Definición de Objetivos: Establecer objetivos claros para la migración, como mejorar el rendimiento o agregar nuevas funcionalidades.
- Plan de Comunicación: Informar a los stakeholders y usuarios sobre el plan y cronograma de migración.

Planificación
- Evaluación de Riesgos: Identificar y evaluar los riesgos potenciales y establecer estrategias de mitigación.
- Diseño de Nuevas APIs: Rediseñar o adaptar las APIs para la nueva plataforma, asegurando la compatibilidad con los sistemas existentes.

Implementación Gradual
- Configuración de Entorno: Preparar el entorno de prueba y producción para los nuevos servicios.
- Migración Piloto:
    - Selección de servicios: Escoger un grupo pequeño de APIs menos críticas para migrar primero.
    - Pruebas exhaustivas: Realizar pruebas completas en el entorno piloto.
    - Monitoreo Intensivo: Supervisar el desempeño y el impacto en tiempo real.

Migración Progresiva
- Fase 1: Migrar APIs no críticas y supervisar el impacto en el sistema.
- Fase 2: Migrar APIs críticas por fases, asegurando que cada fase funcione correctamente antes de proceder.
- Fase 3: Completar la migración de todas las APIs restantes.

Validación y Ajustes
- Monitoreo Post-Migración: Supervisar el desempeño y la estabilidad de las APIs migradas.
- Retroalimentación: Recoger retroalimentación de los usuarios y realizar ajustes necesarios.

Desactivación de la Infraestructura Antigua
- Desactivación Gradual: Desactivar las APIs y servicios antiguos de manera controlada.
- Revisión Final: Asegurar que todas las funcionalidades estén funcionando como se espera en la nueva plataforma.

Documentación y Capacitación
- Actualización de Documentación: Asegurar que la documentación esté actualizada con los cambios realizados.

- Capacitación: Ofrecer capacitación a los desarrolladores y usuarios sobre las nuevas APIs y cualquier cambio en los procesos.