CONECTA - Gateway bidireccional de integración de servicios

Table of Contents

1. Presentación

CONECTA surge como una necesidad de centralizar y estandarizar las conexiones entre múltiples servicios externos y aplicaciones internas de una organización. Actualmente, cada integración se realiza de manera individual, generando una complejidad creciente, dificultad para escalar, mantener y auditar.

El objetivo del sistema es crear una puerta de entrada/salida única para estas integraciones, permitiendo:

- Conexiones bidireccionales entre orígenes externos y destinos internos.
- Configuración dinámica de rutas de integración.
- Seguridad basada en tokens JWT.
- Auditoría de cada mensaje enviado o recibido por el sistema.

- Escalabilidad para múltiples orígenes/destinos simultáneos.

2. Requisitos

2.1. Must Have

- CONECTA debe permitir la integración bidireccional entre servicios externos e internos vía HTTP con mensajes JSON.
- Autenticación de servicios externos mediante tokens JWT firmados.
- Registro de auditoría para cada mensaje entrante y saliente (payload, timestamp, origen, destino, estado).
- Configuración dinámica de rutas entre orígenes y destinos.
- Alta disponibilidad y tolerancia a fallos.
- Validación de esquema (JSON Schema) de los mensajes.

2.2. Should Have

- UI administrativa para gestionar rutas, tokens y logs de auditoría.
- Envío de alertas ante errores de entrega o fallos de autenticación.
- Encriptación de los datos sensibles en tránsito y en reposo.

2.3. Could Have

- Compatibilidad futura con eventos asíncronos (Webhooks, Kafka).
- Simulación/prueba de rutas antes de activarlas.
- Estadísticas de uso y performance de cada integración.

2.4. Won’t Have (por ahora)

- Soporte para protocolos distintos a HTTP.
- Motor de transformación compleja de mensajes (ETL avanzada).

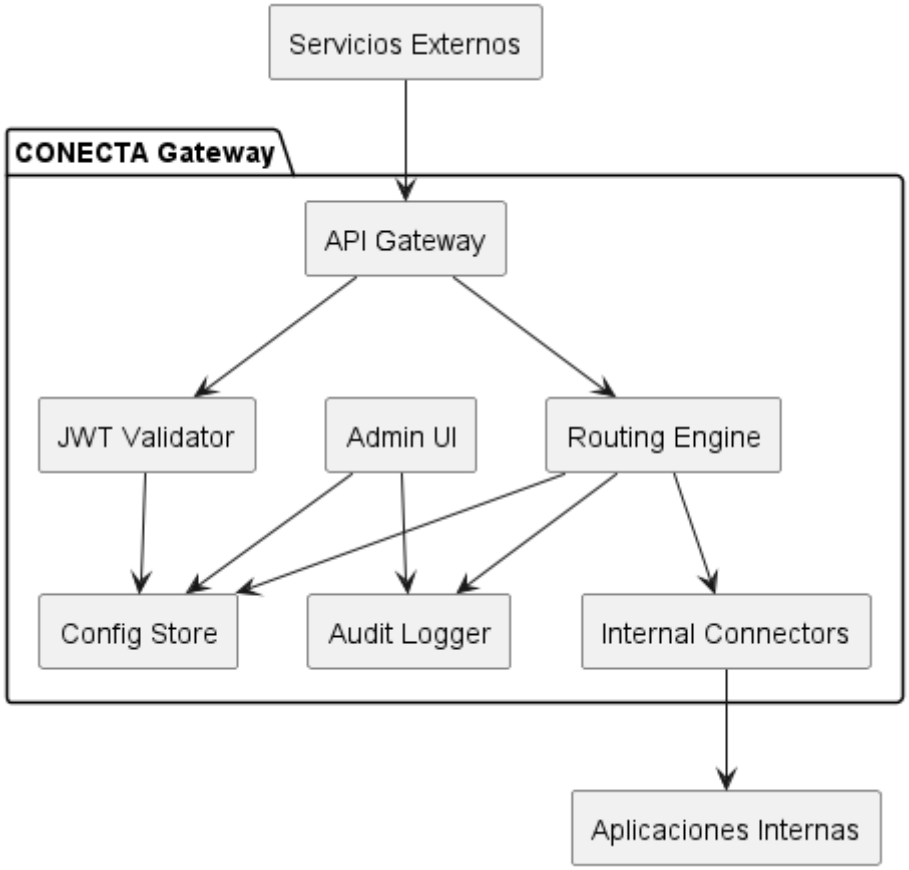
3. Metodología

3.1. Arquitectura General

CONECTA se compone de los siguientes bloques principales:

- **API Gateway**: Punto de entrada para todos los servicios externos. Maneja autenticación JWT y delega las solicitudes.

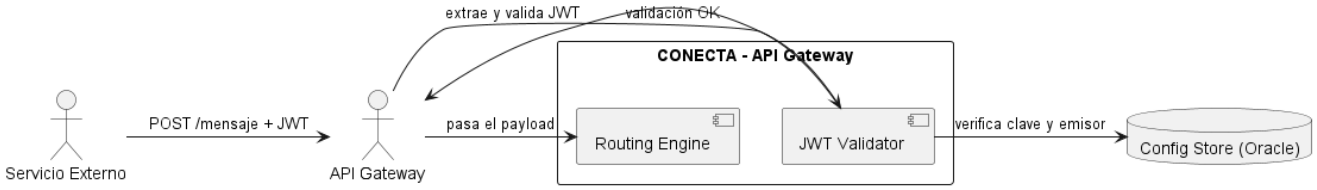
- **Routing Engine**: Motor interno que decide el destino de cada mensaje en función de la configuración.
- **Internal Connectors**: Módulos encargados de conectar con las aplicaciones internas.
- **Audit Logger**: Servicio que registra todos los intercambios (request/response) en una base de datos de auditoría.
- **Admin UI**: Interfaz para configurar rutas, gestionar tokens y revisar logs.
- **Config Store**: Base de datos donde se almacenan definiciones de rutas, tokens válidos, esquemas, etc.



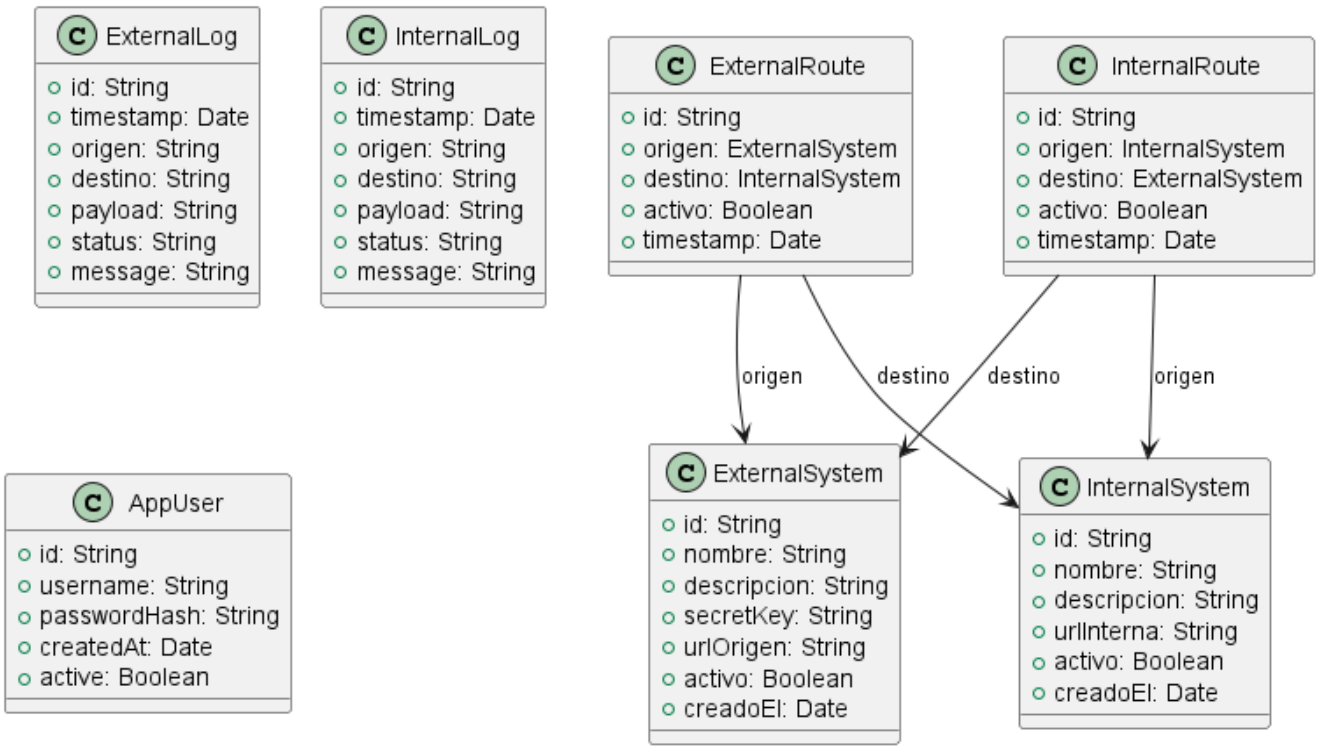
3.2. Autenticación JWT

- Cada sistema externo se identifica mediante un token JWT firmado previamente con una clave compartida (HS256).
- El API Gateway de CONECTA valida el JWT en cada solicitud entrante, revisando:
- Firma (clave secreta válida)
- Fecha de expiración (`exp`)
- ID del emisor (`iss`) o sujeto (`sub`)
- El JWT no contiene permisos, solo identidad del sistema.

3.2.1. Flujo de Autenticación



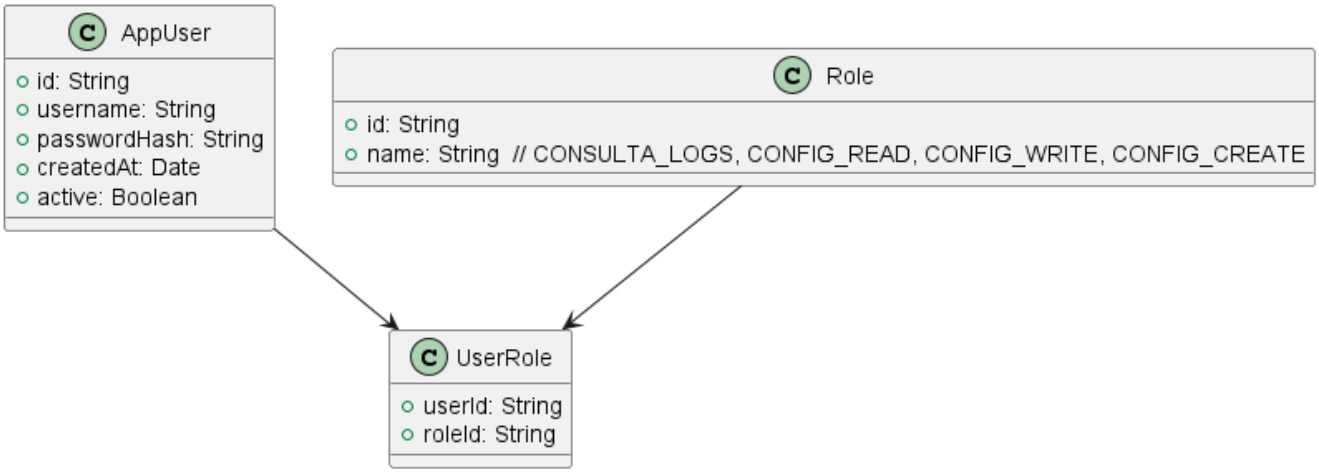
3.3. Modelo de Dominio (UML)



3.4. Roles definidos

- CONSULTA_LOGS: Puede visualizar registros de auditoría.
- CONFIG_READ: Puede ver rutas y sistemas.
- CONFIG_CREATE: Puede crear nuevas rutas o sistemas.
- CONFIG_WRITE: Puede modificar rutas o sistemas existentes.

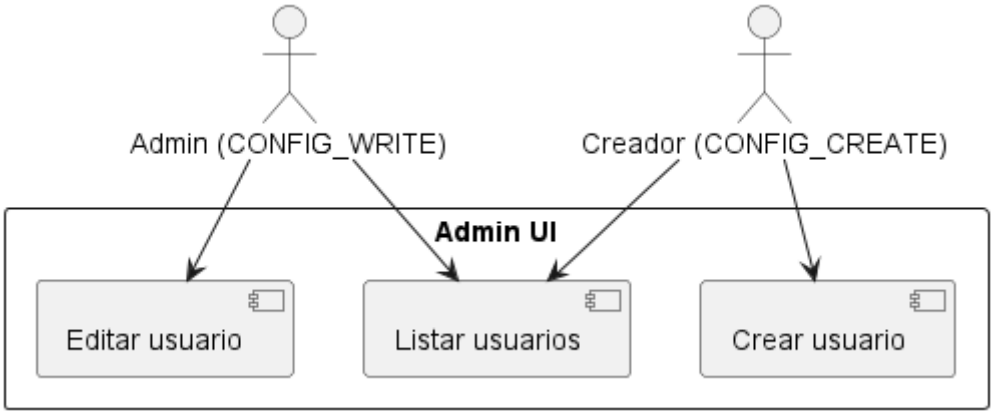
3.5. Seguridad y Control de Acceso



3.6. Gestión de Roles vía UI

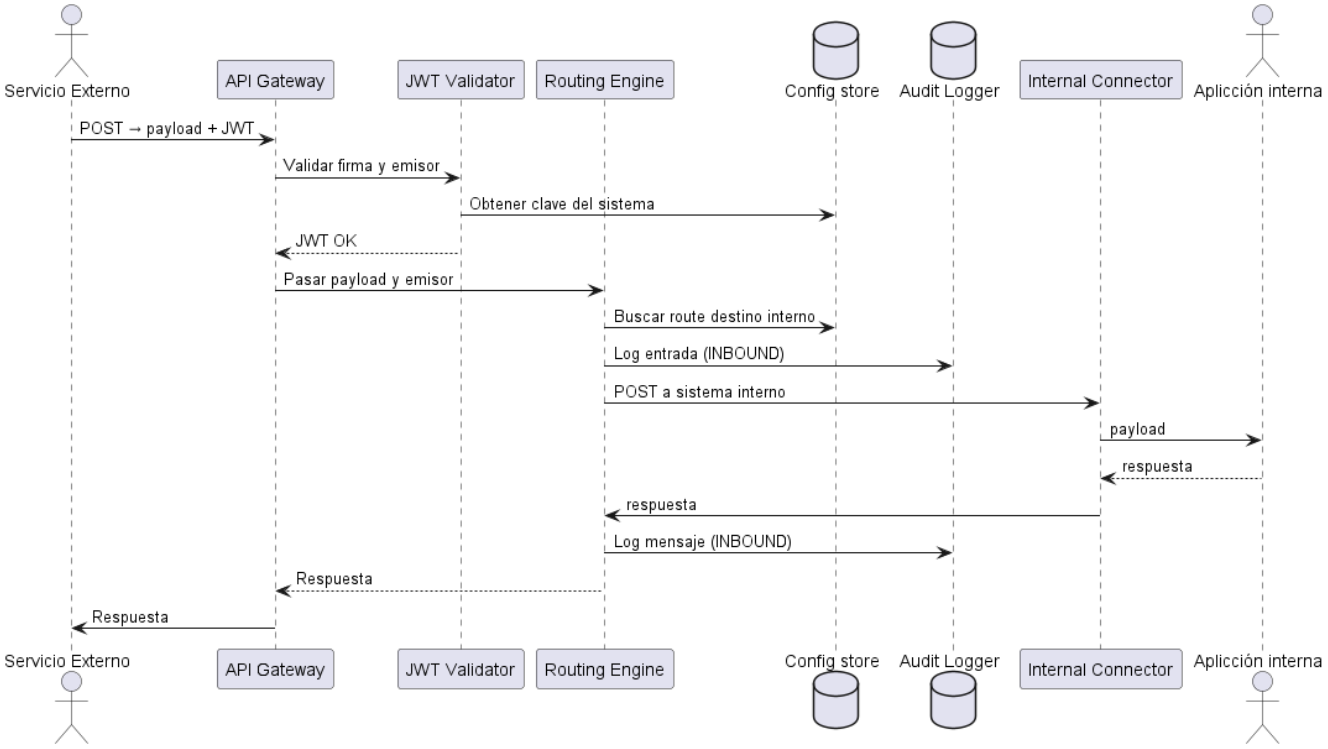
- Los usuarios con permiso `CONFIG_CREATE` podrán:
- Crear nuevos usuarios.
- Asignar roles iniciales.
- Los usuarios con permiso `CONFIG_WRITE` podrán:
- Modificar roles de usuarios existentes.
- Desactivar usuarios.

La UI administrativa incluirá: - Pantalla de listado de usuarios. - Formulario de creación de usuario con asignación de roles. - Edición de roles y estado activo.

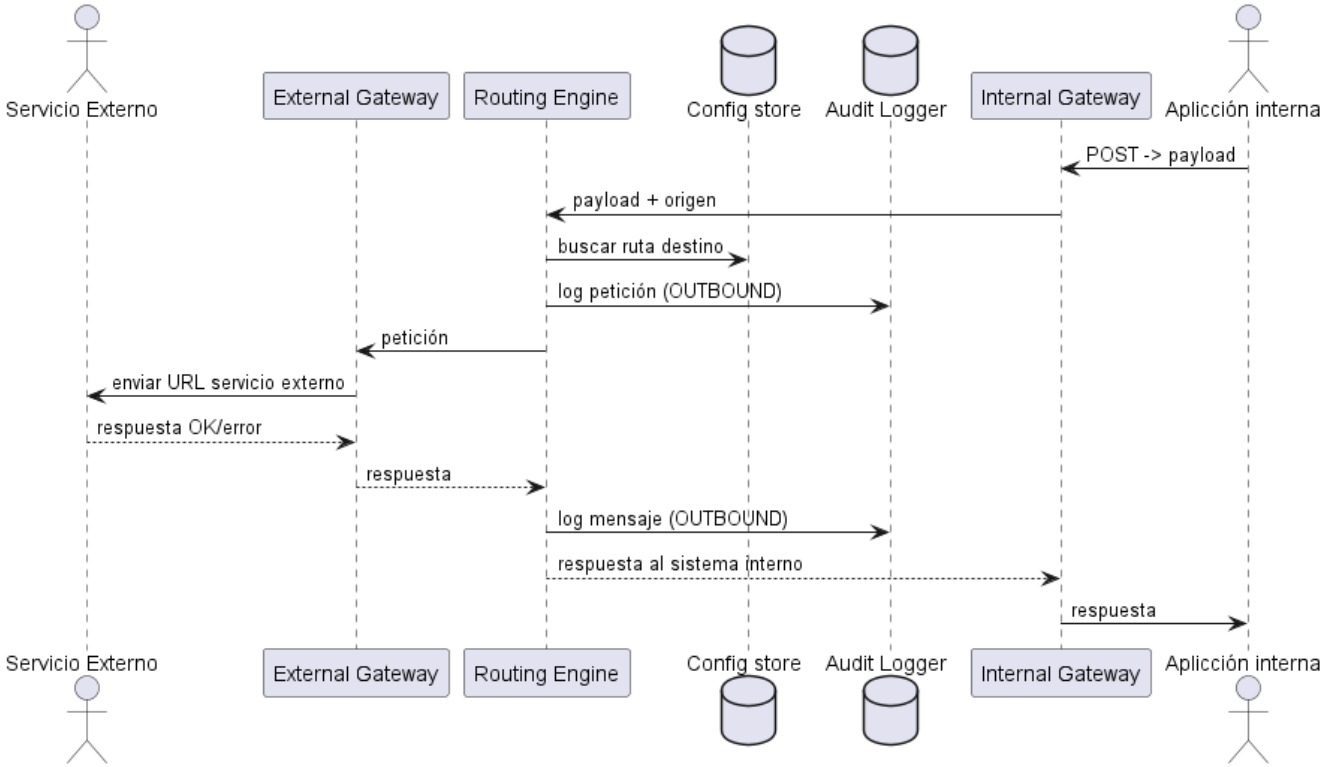


3.7. Flujo de Mensajes Entrantes (Externo → Interno)

3.7.1. Mensaje entrante



3.7.2. Mensaje saliente



4. Implementacion

4.1. Backend (Spring Boot)

1. Crear un proyecto Spring Boot con los siguientes módulos:
    - `conecta-api`: Exposición de endpoints para sistemas externos e internos.

    - `conecta-core`: Lógica de ruteo, validación, auditoría.
    - `conecta-auth`: Validación de tokens JWT y roles de usuarios.
    - `conecta-admin`: API REST para gestionar sistemas, rutas y usuarios.
2. Configurar Oracle como base de datos principal usando Spring Data JPA.
3. Implementar validación JWT usando HS256 con claves por sistema externo.
4. Implementar timeout global de 10s para todas las llamadas HTTP salientes.
5. Auditar todas las operaciones en las tablas `EXTERNAL_LOG` e `INTERNAL_LOG`.

4.2. Frontend (Angular)

1. Crear un proyecto Angular separado llamado `conecta-admin-ui`.
2. Funcionalidades de la UI:
    - Gestión de sistemas internos y externos.
    - Gestión de rutas entrantes (`routes_external`) y salientes (`routes_internal`).
    - Visualización de logs con filtros por fecha, estado, sistema.
    - Gestión de usuarios y asignación de roles (`CONSULTA_LOGS`, `CONFIG_*`).
3. Login con usuario y contraseña con JWT recibido desde backend.

4.3. Seguridad

1. Backend:
    - Login de usuarios administrativos usando contraseña con hash (BCrypt).
    - Generación de token JWT para sesiones de UI.
    - Verificación de roles por endpoint en base a `UserRole`.
2. Frontend:
    - Guardas de ruta según rol.
    - Almacenamiento seguro del JWT en memoria (no en localStorage).

4.4. Configuración y despliegue

1. Propiedades de configuración externa en `application.yml`:
    - `jwt.secret=<clave por defecto>` para usuarios internos.
    - `timeout.http=10000` (10s).
2. Deploy en servidor con contenedor (Tomcat o Spring Boot embebido).
3. Servir UI como app Angular estática o vía nginx aparte.

5. Milestones

1. Diseño técnico detallado finalizado
    - Confirmación de modelo de dominio
    - Diagrama de arquitectura validado
    - Esquemas base de Oracle definidos
2. Configuración inicial del entorno
    - Repositorio con estructura de proyecto Spring Boot y Angular
    - Entorno de base de datos Oracle accesible
    - CI básico para backend y frontend
3. Desarrollo MVP
    - Validación JWT en gateway
    - Ruteo básico entrante/saliente sin UI
    - Registro de logs en Oracle
4. UI Administrativa
    - Login y gestión de usuarios/roles
    - Gestión de sistemas y rutas
    - Visualización de logs
5. Pruebas y verificación funcional
    - Test de integraciones entrantes y salientes
    - Pruebas con sistemas reales o simulados
    - Validación de timeout y errores
6. Despliegue en entorno de producción
    - Configuración de variables sensibles y claves
    - Seguridad de acceso a UI
    - Backup inicial y monitoreo básico