Digitalización de la Granja

Integración de Aplicaciones

Autores: Josu Abad, Darío Pérez, Jesús Arranz Navas
Asignatura: Arquitectura e Integración de Aplicaciones

1. Introducción

Anteriormente habíamos planteado TuGranjita como una granja digitalizada: ubicaciones, sensores, animales, procesos… Ahora, el proyecto da un paso lógico hacia una arquitectura más parecida a lo que usan las empresas reales: dividir el sistema en pequeños servicios independientes.

En esta fase hemos creado dos módulos separados:

- **CRM**, que representa la parte administrativa de la granja (proveedores, distribuidores, servicios externos…).

- **IoT**, que simula los sensores repartidos por diferentes zonas de TuGranjita y las lecturas que generan.

Cada servicio funciona por su cuenta, expone su propia API y valida la información que devuelve. La idea es que la granja ya no es un único programa, sino un conjunto de piezas que se comunican entre ellas, igual que en una arquitectura moderna.

2. Servicio CRM

2.1 Qué aporta al proyecto

El CRM es la parte administrativa. En el día a día de una granja real hay proveedores de pienso, distribuidores, veterinarios, empresas de mantenimiento, etc., por lo que lo hemos añadido como un microservicio aparte.

2.2 Datos que gestiona

Los datos están en el fichero `clientes.json`. Tenemos 25 entidades distintas para poder probar paginación y búsqueda. Cada una tiene:

- id
- nombre
- dirección
- nif
- correo
- teléfono
- y otros datos administrativos

Son valores variados para que se pueda comprobar que las búsquedas funcionan y que el esquema detecta errores.

2.3 Endpoints desarrollados

El CRM expone dos operaciones:

- **GET /clientes**: listado con paginación, búsqueda por texto y filtro por ubicación.
- **GET /clientes/{id}**: devuelve un cliente concreto.

La búsqueda funciona tanto por nombre como por correo, y la respuesta sigue la estructura que exige la práctica: total, página actual, tamaño y lista de resultados.

3. Servicio IoT

3.1 Origen y papel del módulo IoT en la arquitectura

En el diseño general de TuGranjita, la parte de sensorización siempre ha sido un componente clave para monitorizar distintas zonas de la granja. En esta fase del proyecto se ha decidido aislar esa funcionalidad y convertirla en un servicio independiente, siguiendo el enfoque habitual de los sistemas IoT reales.

Este servicio se encarga exclusivamente de los sensores y de las lecturas que generan, permitiendo que el resto de la arquitectura pueda consumir estos datos sin depender de cómo se producen internamente.

3.2 Datos simulados

Para este servicio tenemos dos ficheros:

- *sensores.json*
- *lecturas.json*

Los sensores tienen un identificador, un tipo y una ubicación, igual que en el modelo anterior (temperatura en el almacén, humedad en un establo, etc.). Las lecturas simulan los datos que estos sensores generan.

3.3 Endpoints implementados

El IoT expone dos operaciones muy sencillas:

- **GET /sensores**: lista todos los sensores.
- **GET /lecturas**: devuelve todas las lecturas, y admite filtros por id de sensor.

El servicio valida cada dato con su schema correspondiente para evitar inconsistencias, igual que el CRM.

4. Validación con JSON Schema

4.1 Importancia de la validación en una arquitectura de microservicios

En este proyecto cada módulo funciona de manera independiente, y eso implica que ninguno puede dar por hecho que los datos de otro servicio llegarán siempre en buen estado. Por ese motivo, tanto el CRM como el IoT validan su propia información antes de enviarla.

La idea es que cada microservicio sea responsable de garantizar que lo que publica sigue un formato estable y coherente, evitando errores que puedan propagarse al resto del sistema.

4.2 Validación en el servicio CRM

En el caso del CRM, la validación se utiliza para comprobar que toda la información administrativa es consistente. El schema revisa cosas básicas como que los correos tengan un formato válido, los teléfonos estén dentro de un rango razonable o las direcciones sean texto.

Si algún dato no cumple con las reglas definidas en el schema, el servicio devuelve un error claro en lugar de enviar datos que podrían causar problemas en fases posteriores. De esta forma, el CRM actúa como un filtro que garantiza la calidad de la información administrativa de TuGranjita.

4.3 Validación en el servicio IoT

El servicio IoT sigue la misma filosofía, pero aplicada a sensores y lecturas. Su schema asegura que los sensores están bien identificados, que los tipos coinciden con lo esperado y que las lecturas contienen todos los campos necesarios.

Esto evita situaciones típicas como lecturas incompletas, valores incoherentes o sensores sin identificador. Aunque los datos sean simulados, el sistema se comporta como lo haría un entorno real donde cada módulo protege su propio contrato de datos, contribuyendo a que toda la arquitectura sea más robusta.

5. Conclusión

El resultado final son dos microservicios completamente independientes, cada uno con sus datos, sus validaciones y su API. Aunque todavía no se integran entre ellos, ya están preparados para hacerlo en fases posteriores del proyecto.

Este paso ha servido para convertir TuGranjita en algo mucho más modular y cercano a un sistema real, donde cada parte de la granja (administrativa y sensores) está separada y puede evolucionar por su cuenta.