

Evaluación Practica Integración de Sistemas

Integrante: José Sánchez

Evidencias:

```
PS C:\examen-practico-integracion\evaluacion-practica-agrotech> mvn exec:java
[INFO] Scanning for projects...
[INFO]
[INFO] -----< com.agrotech:evaluacion-practica-agrotech >-----
[INFO] Building evaluacion-practica-agrotech 1.0-SNAPSHOT
[INFO]    from pom.xml
[INFO] -----[ jar ]-----
[INFO]
[INFO] --- exec:3.1.0:java (default-cli) @ evaluacion-practica-agrotech ---
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
Starting Camel (MainApp). Press Ctrl+C to stop.
oct 30, 2025 9:29:41 P. M. io.undertow.Undertow start
INFO: starting server: Undertow - 2.2.21.Final
oct 30, 2025 9:29:42 P. M. org.xnio.Xnio <clinit>
INFO: XNIO version 3.8.7.Final
oct 30, 2025 9:29:42 P. M. org.xnio.nio.NioXnio <clinit>
INFO: XNIO NIO Implementation Version 3.8.7.Final
oct 30, 2025 9:29:42 P. M. org.jboss.threads.Version <clinit>
INFO: JBoss Threads version 3.1.0.Final
```

Respuestas de las preguntas:

1. ¿Qué patrón aplicaste en cada fase del flujo y por qué?

File Transfer para la ingestión batch (leer [sensores.csv](#), mover a processed y procesar filas) porque desacopla productor/consumidor y es simple para lotes; Shared Database para intercambio de estado entre AgroAnalyzer y FieldControl (ambos leen/escriben en lecturas) por rapidez de implementación; RPC simulado (direct request-reply en Camel) para modelar llamadas síncronas cuando el cliente necesita respuesta inmediata.

2. ¿Qué riesgos observas al usar una base de datos compartida?

Riesgos de DB compartida: conlleva acoplamiento fuerte entre servicios, problemas de gobernanza y migraciones, riesgo de contención y single point of failure, dificultades para escalar y gestionar permisos/seguridad, y mayor complejidad para probar y evolucionar sin romper consumidores.

3. ¿Cómo ayuda el RPC simulado a representar un flujo síncrono?

RPC simulado y flujo síncrono: al usar rutas `direct` y `requestBody` en Camel se modela exactamente la semántica `request-reply` —el cliente bloquea hasta recibir respuesta o `timeout`— lo que permite probar comportamiento síncrono (`timeouts`, errores) sin necesidad de infraestructura remota, aunque no reproduce latencias/particiones de red reales.

4. ¿Qué limitaciones tienen los patrones clásicos frente a arquitecturas modernas?

Limitaciones de patrones clásicos vs modernos: enfoques como `File Transfer` o `Shared DB` son fáciles de implementar pero limitan escalabilidad, resiliencia y despliegue independiente; las arquitecturas modernas (`event-driven`, `DB-per-service`, mensajería) favorecen desacoplo, tolerancia a fallos y evolución independiente a costa de mayor complejidad operacional.