



Ejercicio autónomo – Análisis distribuido de mantenimiento en una flota robótica industrial

Objetivo de la tarea

Con este ejercicio vas a aplicar todo lo aprendido en los ejemplos anteriores para construir un análisis completo distribuido que procese los registros de múltiples brazos robóticos.

El objetivo es:

- **Leer múltiples archivos de logs** generados por distintas unidades robóticas.
- **Filtrar automáticamente eventos críticos** definidos por los técnicos.
- Agregar los **datos** por tipo de suceso técnico y generar **estadísticas** de frecuencia.
- Observar la **ejecución distribuida** en la interfaz de Spark.
- **Reflexionar** sobre el **diseño, escalabilidad y utilidad** del sistema en un entorno realista.

Conexión con los ejemplos anteriores

Este ejercicio retoma los conceptos clave trabajados en:

- **Ejemplo 1:** generación y transformación de datos simulados (calibración).
- **Ejemplo 2:** filtrado y paralelismo aplicado a valores anómalos.
- **Ejemplo 3:** procesamiento realista de múltiples archivos con filtrado, agrupación y reducción distribuida (MapReduce).

Ahora, tú tendrás que diseñar y ejecutar todo el flujo de análisis con autonomía, basándote en la experiencia previa.

Contexto técnico

Cada brazo robótico de la planta genera diariamente un archivo de informe (.txt) con eventos registrados durante su funcionamiento. Estos eventos pueden incluir:

- ALERTA_SOBRECALENTAMIENTO
- REINICIO_SISTEMA



- CALIBRACION_AUTOMATICA
- CARGA_EXCESIVA
- PAUSA_NO_PROGRAMADA

Estos informes se recogen cada día en una carpeta compartida. El sistema central de análisis debe procesarlos todos y generar un resumen distribuido de frecuencia de eventos.

Requisitos del ejercicio

Tu tarea consiste en:

1. Crear al menos **cuatro archivos** `.txt` con eventos realistas en una carpeta `registros/``.
2. **Preparar el entorno** de Spark con Docker, montando esa carpeta como ``/app``.
3. Desarrollar el **código en `spark-shell`** para que:
 - **Lea** todos los archivos `.txt` de la carpeta.
 - **Filtre** las líneas que contengan solo los eventos técnicos clave.
 - **Extraiga** el nombre del evento (como clave).
 - Aplique una **reducción distribuida** para contar cuántas veces aparece cada evento.
 - Muestre el **resultado** en consola con ``collect()``.
4. Acceder a ``http://localhost:4040`` y analizar cómo Spark ha distribuido y ejecutado el trabajo.

Consejos

- Usa ``textFile()`` para leer todos los archivos.
- Utiliza ``filter``, ``map`` y ``reduceByKey`` como base del proceso.
- El ``collect()`` debe ejecutarse al final para lanzar el trabajo.
- Observa en Spark UI cuántas etapas se generan y si hay **shuffle**.

OJO:

La parte técnica del código puede contar con apoyo de herramientas de IA, siempre que comprendas su funcionamiento.

Las respuestas reflexivas y la interpretación del trabajo deben ser redactadas personalmente, sin ayuda de IA.



Estructura sugerida

```
val registros = spark.sparkContext.textFile("/app/registros/*.txt")
```

```
// filtrar líneas que contienen eventos clave
```

```
// extraer claves y contarlas
```

```
// mostrar el resultado con collect()
```

Análisis personal obligatorio

Redacta un breve informe (máx. 1 página) respondiendo con tus propias palabras a las siguientes cuestiones (ver apartado siguiente “Entrega” donde se detalla los contenidos a entregar)


1. ¿Qué pasos seguiste para construir tu solución y por qué?
2. ¿En qué parte tuviste dificultades y cómo las resolviste?
3. ¿Qué crees que está ocurriendo “por dentro” cuando Spark ejecuta tu código?
4. ¿Qué aprendiste sobre el middleware distribuido que no conocías antes?

OJO: Este análisis debe reflejar tu experiencia personal. Se corregirá con especial atención para comprobar la comprensión auténtica del ejercicio.

Entrega

Para completar correctamente este ejercicio, debes **subir a Moodle un único archivo comprimido (.zip)** que contenga los siguientes elementos obligatorios:

Contenido del archivo .zip que debes entregar:

1. **Informe personal** (.pdf o .docx)
2. Documento con las respuestas al apartado “ Análisis personal obligatorio”, redactado con tus propias palabras.
3. **Capturas de pantalla** (formato .png o .jpg)
 - Pantalla del terminal con el resultado del comando `collect()`.
 - Pantalla del navegador mostrando la interfaz de Spark UI (<http://localhost:4040>), concretamente:
 - Pestaña “**Jobs**”: mostrando el Job ejecutado.
 - Pestaña “**Stages**”: donde se vean las etapas generadas.
 - (Opcional pero recomendable) Pestaña “**Executors**” con estadísticas generales.



4. **Código utilizado**
5. Archivo `.scala`, `.txt` o `.docx` con el código que escribiste paso a paso (sin copiarlo de la IA directamente, debe reflejar lo que comprendiste y aplicaste).
6. **Carpeta registros/ con los archivos de entrada**
7. Archivos `.txt` que utilizaste como entrada en la práctica, con los sucesos generados. Deben ser realistas y distintos entre sí.

Formato y entrega final

- Comprime todos los elementos anteriores en un único archivo `.zip`.
- El nombre del archivo debe seguir este formato:
- `EjercicioFinalSpark_NombreApellido.zip`
- Sube ese archivo en el espacio correspondiente de la tarea en Moodle.

NOTA: No se evaluarán entregas incompletas, sin evidencias o con respuestas genéricas que no reflejen comprensión personal del ejercicio.

Rúbrica de evaluación

El siguiente esquema se utilizará para evaluar tu entrega. Los apartados marcados con reflejan aspectos que deben resolverse de forma personal y sin ayuda de IA.

| Criterio | Excelente (2 pts) | Aceptable (1 pto) | Insuficiente (0 pts) |
|--|---|--|---|
| Preparación técnica (entorno y código) | Ejecuta correctamente, incluso con apoyo puntual de IA o ejemplos. | Ejecuta con errores leves, requiere ajustes. | No ejecuta o el código no es funcional. |
| Análisis personal del proceso | Redacción clara y auténtica sobre su proceso y comprensión. | Explicación básica o con partes genéricas. | Texto copiado o sin argumentación real. |
| Interpretación de Spark UI | Identifica correctamente Jobs, Stages y explica el paralelismo observado. | Comprensión parcial o con errores leves. | No interpreta o no accede a Spark UI. |
| Reflexión técnica final | Contesta con profundidad, conecta teoría y práctica. | Respuestas incompletas o muy generales. | Respuestas vacías, erróneas o copiadas. |



Universitat d'Alacant
Universidad de Alicante

Puntuación total: /8 puntos

- 7-8 puntos: Dominio claro del ejercicio y comprensión real.
- 5-6 puntos: Competencia suficiente con aspectos a mejorar.
- 0-4 puntos: Requiere refuerzo conceptual o repetición del ejercicio.