Análisis de Transiciones en el Ciclo de Vida MLOps

Grupo F: Monitoring System, Orchestrator y Metadata Store

Grupo F

Asignatura: MLOps & AI BI

19 de febrero de 2026

Alcance

Transiciones asignadas:

1. **Trigger de Reentrenamiento:**

$$\text{Orchestrator} \dashrightarrow \text{Data Retrieval}$$

2. **Gestión de Metadatos:** Flujos desde/hacia el **ML Metadata Store** conectados con:
   - Data Retrieval & Validation
   - Model Training & Validation
   - Model Registry

Contexto: Somos responsables de la "inteligencia" (cuándo actuar) y la "memoria" (qué ha ocurrido) del sistema CT (Continuous Training).

Transición 1: Monitoring System → Orchestrator→ Data retrieval

Identificación del Flujo

Tipo: Metadata (Monitoring System/ad-hoc/cron → Orchestrator).
Tipo: Trigger (Orchestrator→ Data retrieval)

- **Justificación:** No se transfieren datasets ni modelos. Es una señal de evento basada en una condición lógica.
- **Descripción Técnica:**
  - El sistema de monitorización evalúa métricas en tiempo real (drift, accuracy, latencia).
  - **Condición de activación:** Una métrica cruza un umbral predefinido (ej. *Accuracy* $< 0{,}85$).
  - **Dato transferido:** Payload ligero (JSON) con ID de alerta y timestamp.

Transición 1: Implementación Real

Herramientas Propuestas:

- **Monitor:** Prometheus + Alertmanager.
- **Orquestador:** Apache Airflow.
- **Mecanismo:** Webhook de Alertmanager a la API REST de Airflow para activar un DAG.

Análisis de Fallos:

- Fallo: "Tormenta de alertas" (Falsos positivos provocan reentrenamientos infinitos).
- **Mitigación:** Implementar *Cooldown periods* (ej. máx 1 reentreno cada 24h) y debouncing en las alertas.

Transición 2: Pipeline $\leftrightarrow$ ML Metadata Store

Identificación del Flujo

Tipo: Movimiento de Metadatos.

- **Justificación:** Registro de la traza de ejecución, no del objeto binario.
- **Descripción Técnica:**
  - Cada componente (Training, Validation) reporta su estado.
  - **Información transferida:** Hiperparámetros, métricas de evaluación ($F1 - score, RMSE$), URIs de los artefactos en S3, versión del código (Git hash).
  - **Formato:** Esquemas relacionales o documentos JSON estructurados.

Transición 2: Implementación Real

Herramienta Propuesta:

- **MLflow Tracking Server** (con backend en PostgreSQL/MySQL).
- *Razón:* Estándar de facto, permite registrar parámetros, métricas y artefactos de forma centralizada y agnóstica al lenguaje.

Análisis de Fallos:

- Fallo: Caída del Metadata Store (Timeout/Down).
- **Consecuencia:** Pérdida de linaje. Modelos "huérfanos" sin trazabilidad.
- **Mitigación:** Configuración de Alta Disponibilidad (HA) en la base de datos y caché local en el cliente para reintentos (retries con exponential backoff).

Metadatos vs. Datos

¿Quién genera/consume metadatos?

- **Generan:** Data Retrieval, Validation, Model Training/Validation.
- **Consumen:** Model Registry, Orchestrator.

Diferencia Práctica

- **Flujo de Datos:** Mueve el *contenido* (GB/TB). Requiere alto ancho de banda (I/O intensivo). Ej: CSVs, Imágenes, Pesos del modelo.
- **Flujo de Metadatos:** Mueve el *contexto* (KB). Es descriptivo. Ej: "¿Dónde está el CSV?", "¿Qué precisión tuvo?", "¿Quién lo ejecutó?".

Escenario de Fallo: 3:00 AM

Escenario: El trigger de monitorización falla y no avisa al orquestador.

¿Qué ocurre?

- El pipeline NO arranca.
- El modelo en producción sigue degradándose (Drift).
- El negocio recibe predicciones erróneas silenciosamente.

¿Cómo se entera el equipo?

- **Dead Man's Switch:** Una alerta externa que salta si "NO" se ha recibido un evento de reentrenamiento en X días.
- **Monitorización Sintética:** PagerDuty/OpsGenie comprobando salud del Orchestrator.

Cuello de Botella y SPOF

Punto Único de Fallo (SPOF): ML Metadata Store.

- **Por qué:** Es el nexo central. Si cae, no podemos registrar nuevos modelos, ni validar los existentes, ni auditar el sistema. El pipeline queda "ciego".

Solución de Arquitectura:

- Base de datos gestionada (RDS/Cloud SQL) Multi-AZ (Multi-zona de disponibilidad).
- Replicación de lectura para consultas de análisis, dejando la instancia principal solo para escritura del pipeline.
- Backups automáticos "Point-in-time".

Diagrama Parcial (Monitorización y Metadatos)

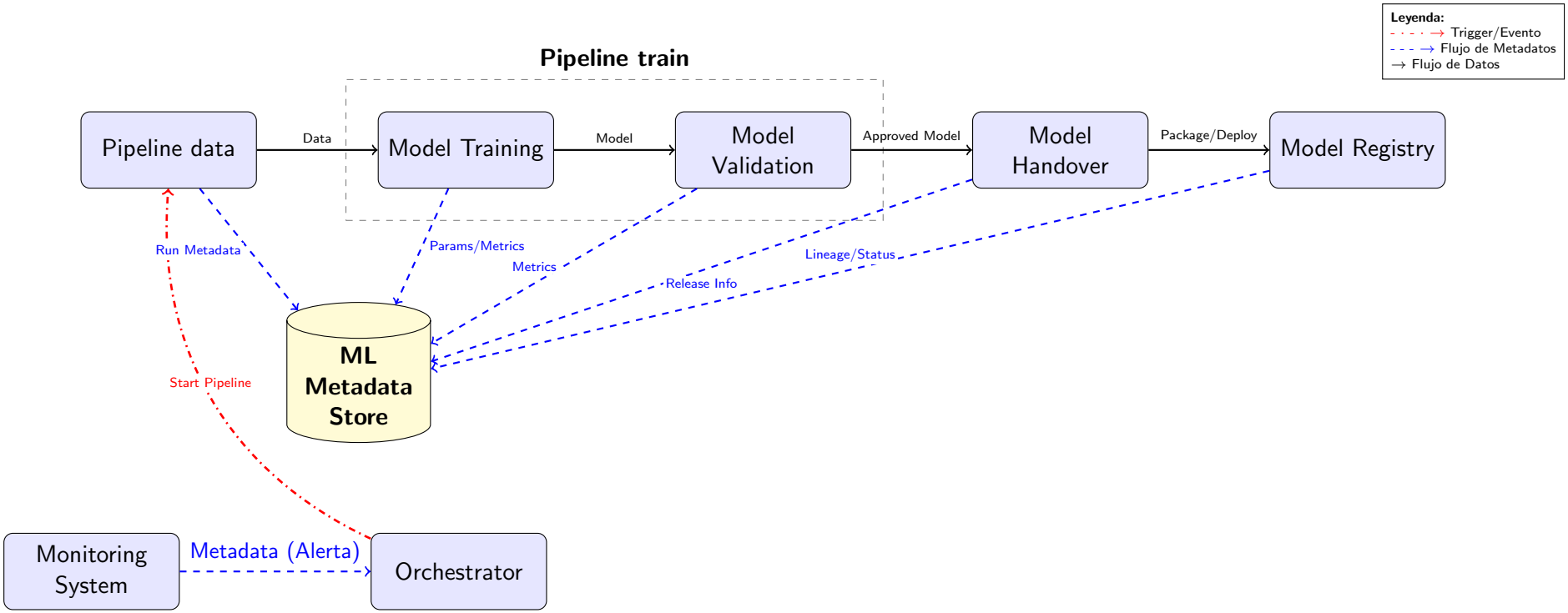

Tabla de Transiciones

Transición	Tipo	Dato Transferido	Herramienta Propuesta	Modo de Fallo	Mitigación
Orchestrator → Data retrieval	**Trigger** (Evento)	Payload JSON (ID Alerta, Severidad, Timestamp). Sin datos pesados.	**Prometheus** (Alertmanager) → **Airflow** (API Trigger DAG).	**Alertas Falsas:** Falsos positivos activan reentrenamientos continuos.	Implementar **Cooldown periods** y agrupación de alertas (debouncing).
Monitoring Sys./ad-hoc/cron → Orchestrator	**Metadatos** (Evento)	Payload JSON (ID Alerta, Severidad, Timestamp).	**Prometheus** (Alertmanager) → **Airflow** (API Trigger DAG).	**Fatiga de Alertas:** Falsos positivos activan reentrenamientos continuos.	Implementar **Cooldown periods** y agrupación de alertas (debouncing).
Pipeline (Training/Val) ↔ ML Metadata Store	**Metadatos**	Hiperparámetros, Métricas ($F1, AUC$), URI de artefactos, Git Hash.	**MLflow Tracking** (Backend SQL + Artifact Store S3).	**Pérdida de Linaje:** Si la BD cae, se generan modelos "huérfanos"sin traza.	Alta Disponibilidad (HA) en la BD y caché local en clientes (Fail-safe).
Metadata Store → Model Registry	**Metadatos**	Estado de validación ("Staging Ready") basado en comparación de métricas.	**MLflow Registry** / AWS SageMaker Model Registry.	**Inconsistencia:** Promoción de modelos sin validación completa por error de lectura.	Transacciones atómicas: Bloquear promoción si faltan metadatos clave.
Pipeline → Model training/validation → handover → registry.	**Datos**	Model/training data	**MLflow Registry** / AWS SageMaker Model Registry.	**Inconsistencia:** Promoción de modelos sin validación completa por error de lectura.	Transacciones atómicas: Bloquear promoción si faltan metadatos clave.

Resumen de Transiciones

Transición	**Tipo**	**Dato**	**Herramienta**	**Mitigación**
Monitor/ad-hoc/cron → Orch.	Metadata	Alert Payload	Prometheus → Airflow	Cooldown / Debounce
Monitor → Orch.	Trigger	Alert Payload	Prometheus → Airflow	Cooldown / Debounce
Pipeline → Meta.	Metadata	Metrics/Params	MLflow Tracking	Retries / Local Cache
Meta. → Registry	Metadata	Model Lineage	MLflow Registry	Atomic Transactions
Pipeline → Model training/validation → handover → registry	Data	Model/training data	MLflow Tracking	