FACULTAD DE INGENIERIA
Universidad de Buenos Aires

Aprendizaje de máquina II

Carrera de Especialización en Inteligencia Artificial

Agenda

- Formas de servir las predicciones de un modelo de ML
- Despliegue mediante APIs usando FastAPI
- Despliegue mediante aplicaciones web usando Gradio
- Despliegue batch usando Airflow

¿Cómo servir las predicciones de un modelo de ML?

Formas de servir las predicciones de un modelo de ML

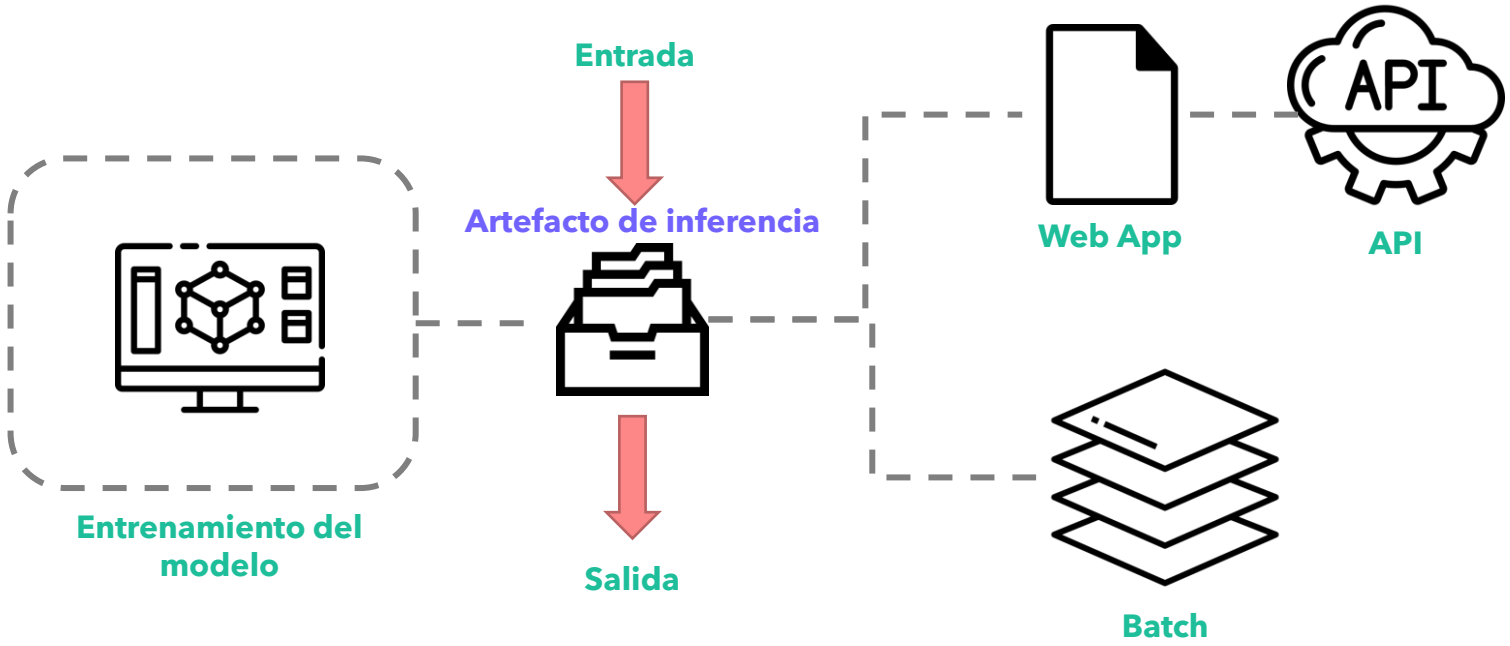
Al momento de servir las predicciones generadas por el modelo para el usuario final, existen varias alternativas.

Cuál es la correcta, depende de los **requerimientos del problema** como, por ejemplo:

- **Tiempo** de respuesta
- **Costos** destinados al proyecto
- **Integración** con otras herramientas de la compañía
- **Infraestructura** disponible

Despliegue de modelos

Algunas de las alternativas posibles para desplegar un modelo son las siguientes:



On-line

Se puede implementar mediante APIs o aplicaciones web. El modelo realiza predicciones bajo demanda y debe tener alta disponibilidad.

Batch

Se puede implementar mediante orquestadores como Airflow, Azure Data Factory (ADF), etc. El modelo realiza predicciones periódicas para un “paquete” de datos

Algunos ejemplos

On-line

Detección de fraude para transacciones con tarjetas de crédito, chatGPT, etc.

Batch

Sistemas de recomendación que me indican qué productos debo vender en el día de hoy, modelos de scoring crediticio, etc.

Fundamentos de las APIs y Aplicaciones WEB

“En términos sencillos, una API es un intermediario que **permite que dos aplicaciones se comuniquen** y compartan información de manera estructurada. Define los **métodos**, **formatos de datos** y **reglas de interacción** que deben seguir las aplicaciones para comunicarse entre sí.”

Fundamentos de las APIs

¿Qué es una API?

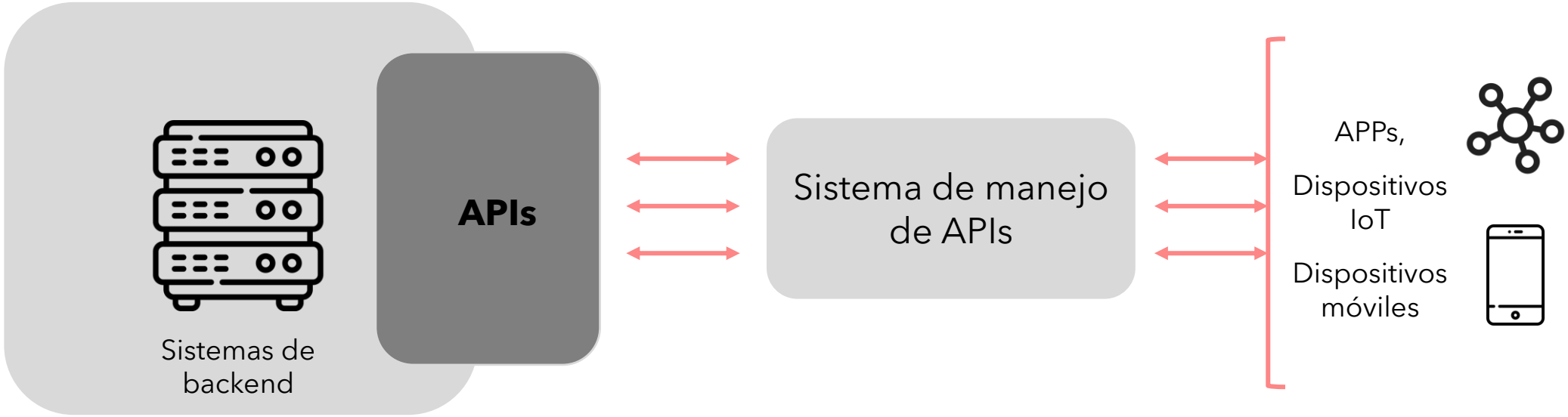
Una API proporciona una forma estandarizada y documentada de acceder a las funcionalidades y datos de una aplicación o servicio.

Esto permite que los desarrolladores puedan utilizar las capacidades de una aplicación o servicio sin necesidad de conocer los detalles internos de su implementación.

→ *Podemos decir entonces que una API es una abstracción de la aplicación que permite la simplificación de la integración.*

La forma mediante la cual los usuarios realizan consultas a la API es mediante el protocolo HTTP.

Fundamentos de las APIs

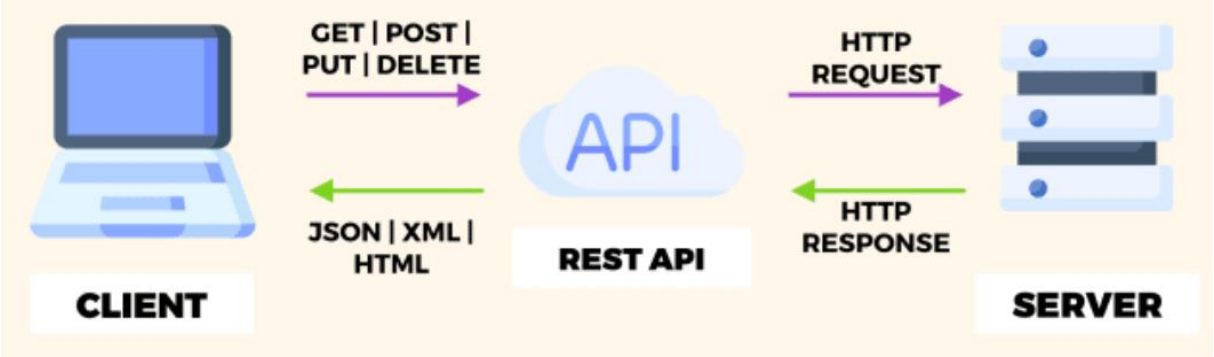


¿Qué es el protocolo HTTP?

¿Qué es el protocolo HTTP?

HTTP (Hypertext Transfer Protocol) es un **protocolo de comunicación** utilizado para la transferencia de información en la web.
HTTP funciona según un modelo **cliente-servidor**, donde un cliente (como un navegador web) realiza solicitudes a un servidor, y el servidor responde a esas solicitudes con los datos solicitados.

Para poder utilizar una API es necesario conocer los **métodos** HTTP y los **códigos de estado** que nos puede devolver.



Métodos y códigos de estado HTTP

Método	Función
GET	Recuperar un recurso existente. Extraer recursos del backend/servidor/API.
POST	Crear/guardar un nuevo recurso. Subir un archivo, cargar datos, etc.
PUT	Actualizar un recurso existente. Editar un usuario, modificar un registro de la base de datos, etc.
DELETE	Eliminar un recurso, registros de la base de datos, usuarios, etc.

Código	Función
2xx	Operación exitosa.
3xx	Redirección.
4xx	Error del cliente.
5xx	Error del servidor

Implementación de APIs en Python

Para implementar APIs en Python hay múltiples herramientas como:

Flask

Django

FastAPI

FastAPI utiliza **pydantic** para la validación del tipo de datos, utiliza **swagger** para la generación automática de documentación, entre otras.

Implementación de APIs en Python

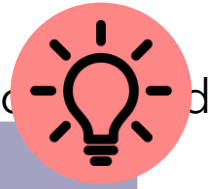
Para implementar APIs en Python hay múltiples herramientas como:

Flask

Django

FastAPI

FastAPI utiliza **pydantic** para la validación de datos, utiliza **swagger** para la generación



Pydantic es una biblioteca de validación y serialización de datos en Python. Está diseñada para facilitar la validación y manipulación de datos en aplicaciones Python, especialmente en el contexto de API web y modelos de datos.

Implementación de APIs en Python

Para implementar APIs en Python hay múltiples herramientas como:

Flask

Django

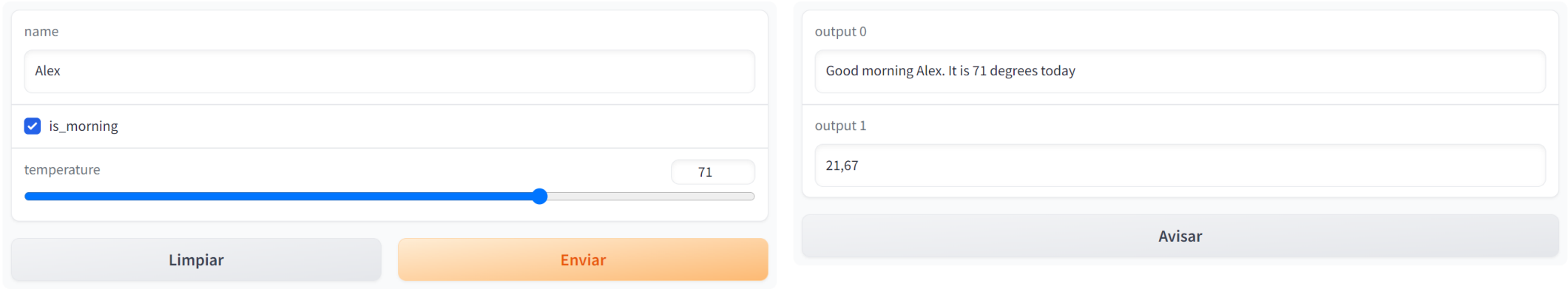
FastAPI

FastAPI utiliza **pydantic** para la validación del tipo de datos, utiliza **swagger** p eneración automática de documentación, e

Swagger es un conjunto de herramientas para diseñar, construir y documentar APIs de manera fácil y colaborativa.

Creación de aplicaciones web

Un usuario de negocio rara vez va a querer ejecutar código para obtener algún resultado. Al contrario, necesita tener disponibles las predicciones en entornos visuales y sencillos.
Para ello podemos utilizar gradio para desarrollar una interfaz web sencilla que nos permita interactuar al usuario con el modelo.

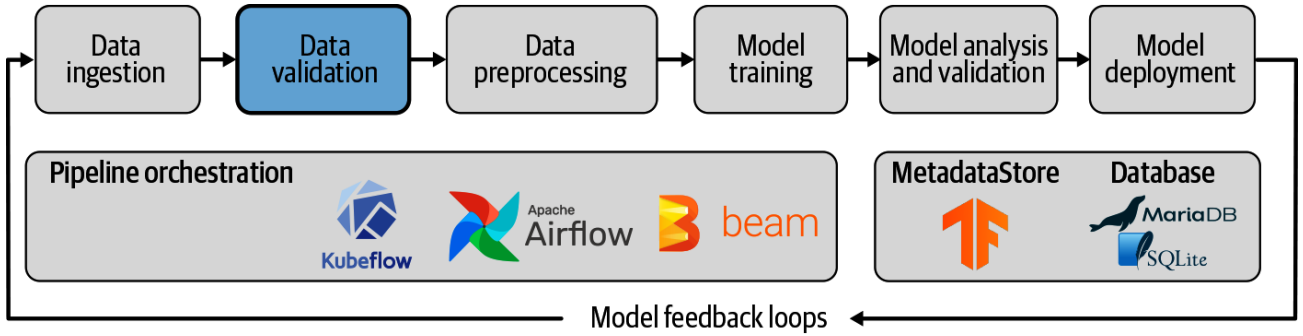


Orquestadores

¿Por qué utilizar Airflow?

Supongamos que estamos trabajando en un pipeline de ingesta de datos o en un pipeline de inferencia para un modelo de machine learning. ¿Qué pasa si alguno de los procesos no se ejecuta correctamente? ¿Cuándo se van a ejecutar esos procesos? ¿Quién los va a ejecutar?

Un orquestador como Airflow puede resolver todas esas tareas de una manera centralizada.



¿Qué es Airflow?

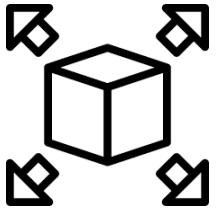
Airflow es una plataforma **open-source** que nos permite programar nuestros propios pipelines, definir un periodo de ejecución y monitorearlos.
Dentro de los beneficios de utilizar Airflow podemos mencionar:

- Para programar los pipelines se utiliza Python, lo que provee al entorno de cierta dinámica.
- Es muy escalable, podemos ejecutar tantas tareas como queramos (lo que permita nuestro poder de cómputo).
- Brinda una interfaz de usuario muy amigable.
- Se pueden incorporar plug-ins creados por el usuario de manera muy sencilla.

Python

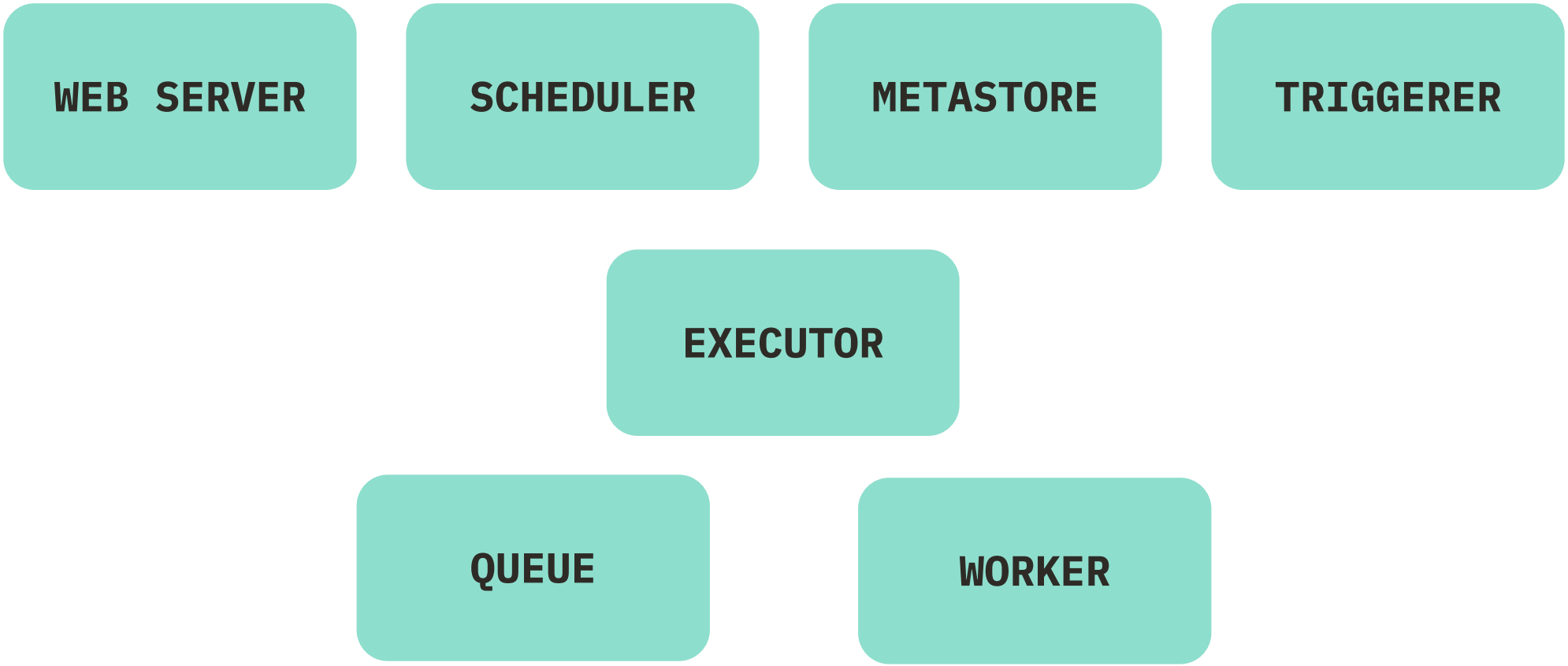
Escalable

Interfaz de usuario

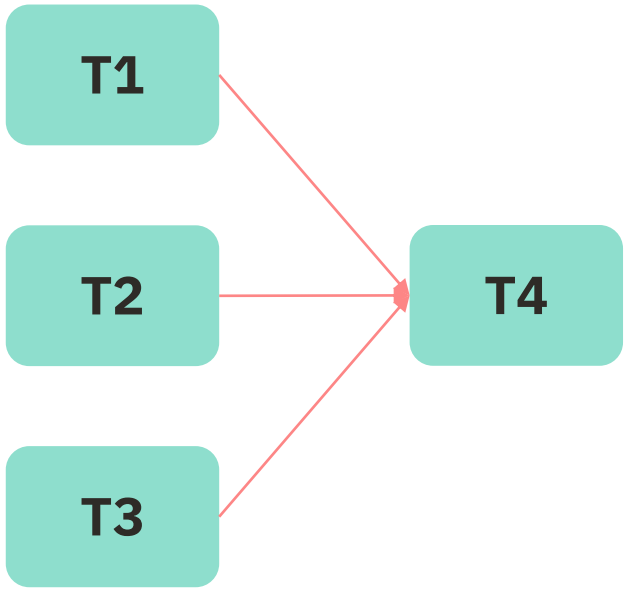


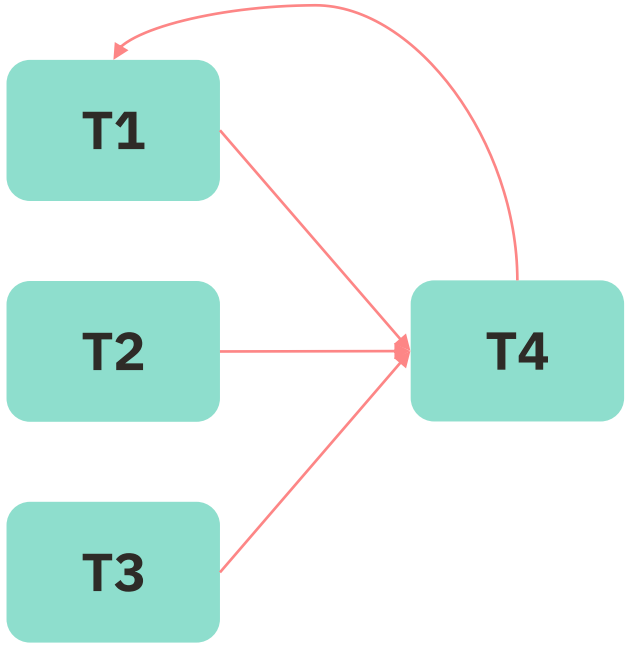
Extensible

Componentes principales



Conceptos principales: DAG





Conceptos principales: Operator

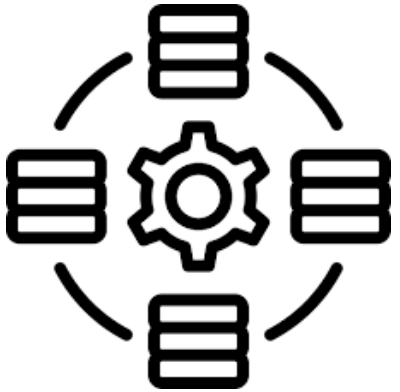
El operator es una tarea dentro de Airflow, una manera de encapsular código.

Existen tres tipos de operadores:

- **Action operators**: ejecutan alguna tarea, como por ejemplo existen python operators, bash operators, etc.
- **Transfer operators**: permiten transferir información de un punto a otro. Por ejemplo, de una BBDD mysql a redshift
- **Sensor operators**: este tipo de operadores nos permiten esperar a que suceda algún evento para avanzar a la siguiente tarea.

¿Qué no es Airflow?

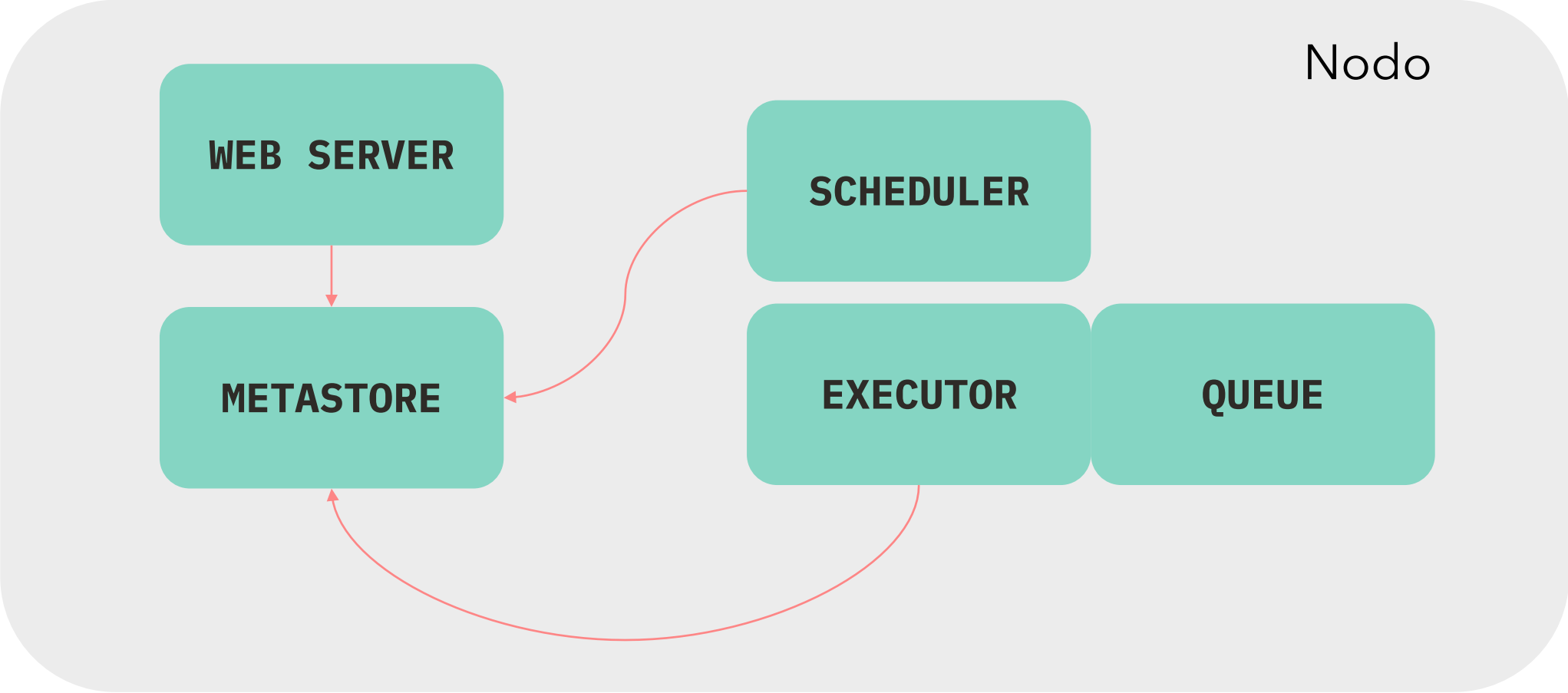
Streaming



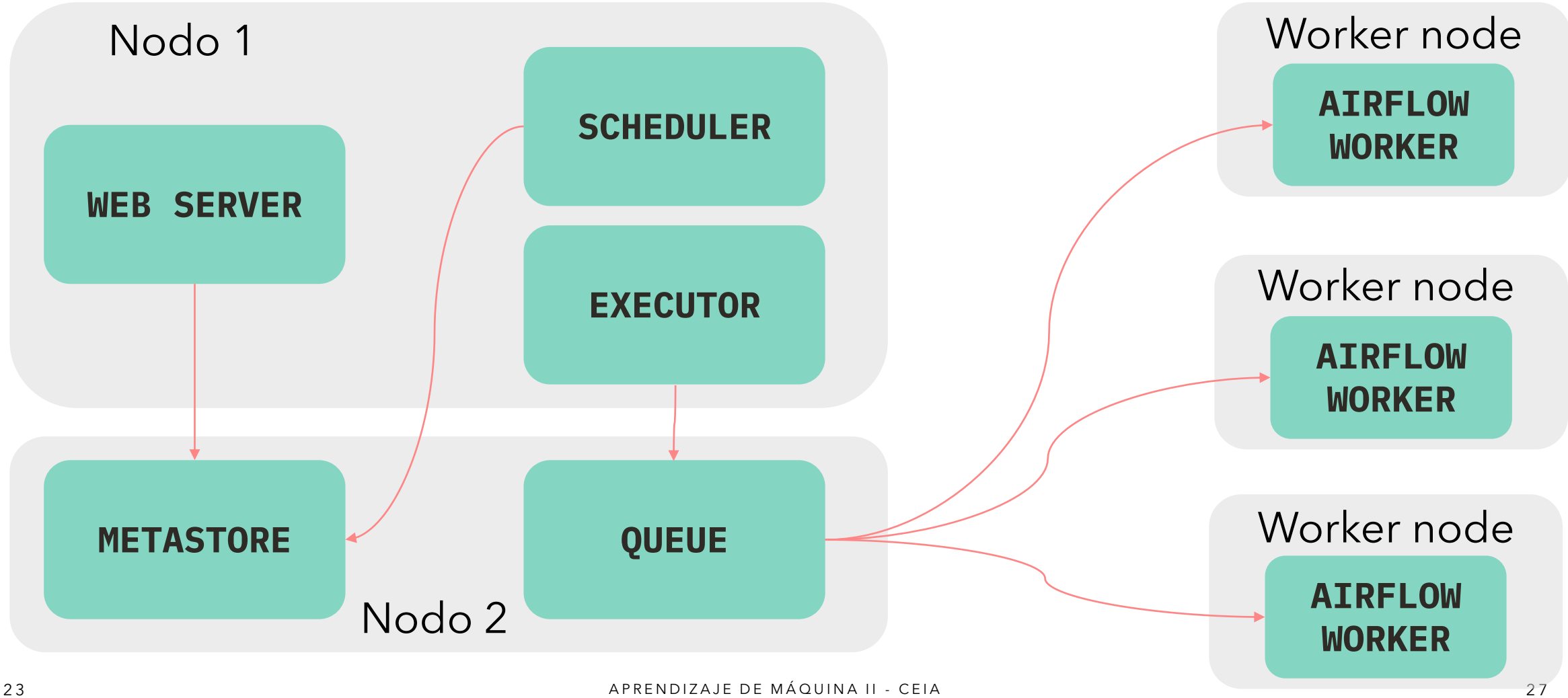
Procesamiento de datos

Airflow no es una plataforma para stremear datos ni tampoco un framework para procesar datos.

Arquitectura de Airflow



Arquitectura de Airflow



¿Cómo funciona Airflow?