Università degli studi di Bologna
Facoltà di Ingegneria

Presentazione su

Valutazione di strumenti per l'osservabilità di Apache Kafka con OpenTelemetry

Alessio Giorgetti

Anno accademico 2023/2024

Cloud Computing

Il **cloud computing** consiste nella **fornitura di servizi** di computing, quali software, database, server e reti, tramite connessione **Internet**.

- **Elaborazione** su una macchina **in remoto**.
- **Delocalizzazione** risorse **hardware**
- **Servizi pay-per-use**

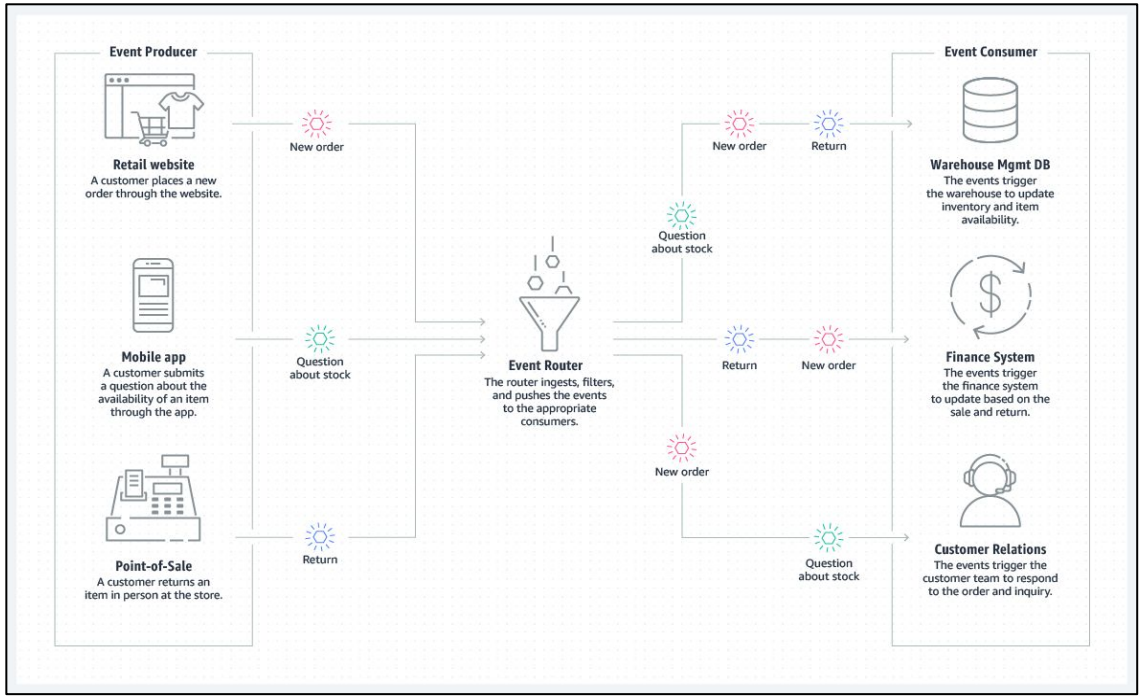
L'utente usufruisce servizi **complessi e dispendiosi** senza la necessità di possedere hardware avanzati e personale specializzato.

Architetture a eventi

I **flussi di dati** sono la base per le **architetture basate sugli eventi**

Un evento è una **registrazione** di qualsiasi **avvenimento o cambiamento** nello **stato hardware o software** di un sistema.

- Il sistema **acquisisce, comunica, elabora** eventi.
- I servizi **disaccoppiati** e dunque **asincroni**



Osservabilità

Osservabilità è la capacità di **ricavare lo stato interno** di un sistema software partendo **dai suoi output esterni** (es logs, metriche, tracciature…).

Si **utilizzano strumenti** che **generano, raccolgono, gestiscono e inviano** ad un servizio di **back-end** un **flusso costante di dati** sulle prestazioni di un'applicazione distribuita.

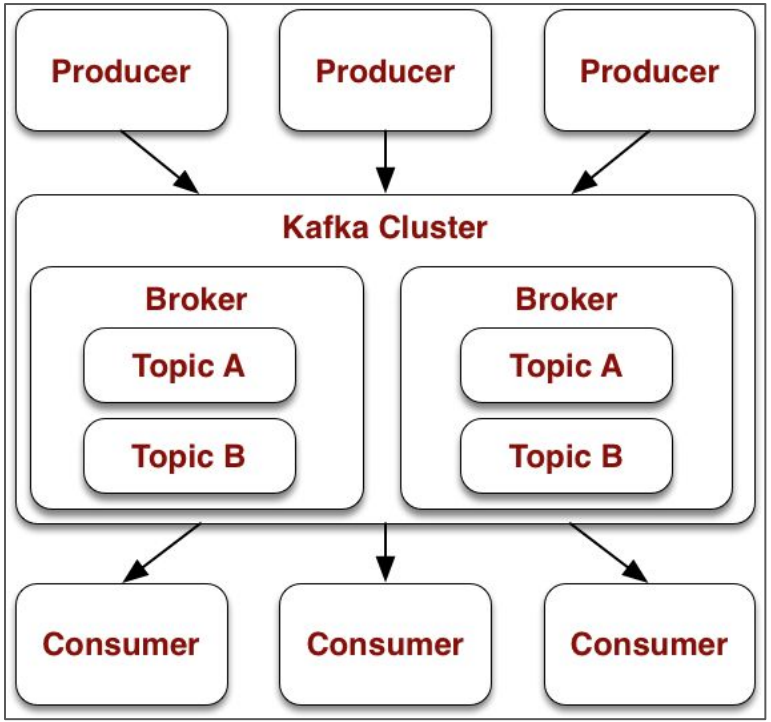
Un **framework standard come OpenTelemetry** permette agli sviluppatori **la comprensione del comportamento e delle performance** delle loro **applicazioni cloud**.

Apache Kafka

Kafka è una piattaforma di scambio di messaggi distribuita con modello di tipo **publish-subscribe**.

i dati sono:

- **archiviati sul broker** in ordine
- **letti** in modo **deterministico**
- **distribuiti tra i broker** del sistema



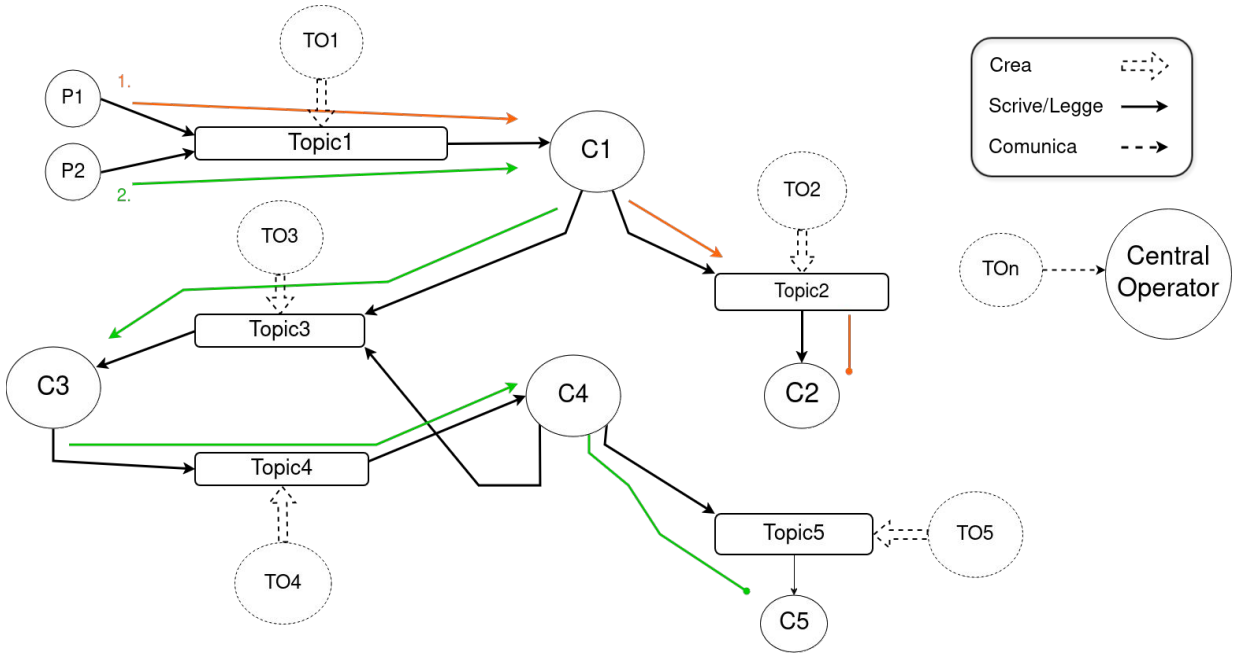
Sviluppo di una piattaforma per l'osservabilità di Kafka

Implementando OpenTelemetry in Kafka possiamo **testare diverse configurazioni** in modo da verificare i cambiamenti in **prestazioni all'aumentare del carico**

Le varie **configurazioni** sono caratterizzate da **due parametri**:

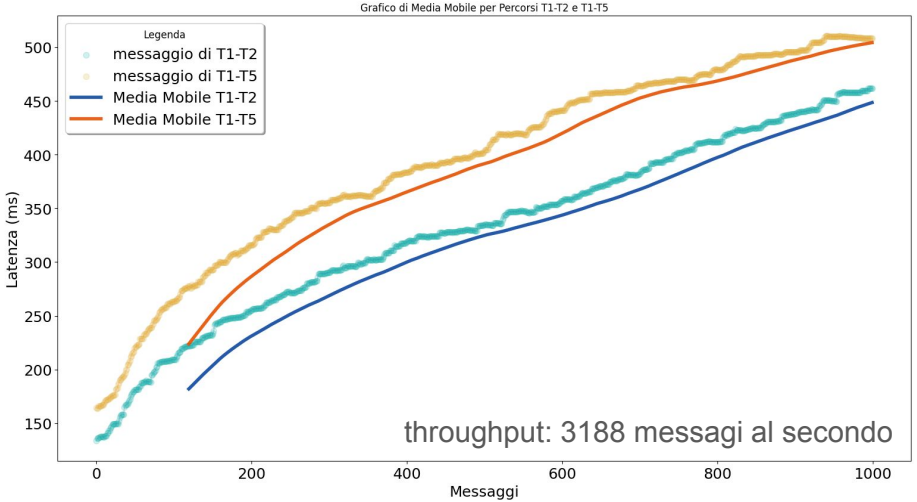
- **numero di broker** attivi
- **numero di partizioni** di cui sono composte le topic

I test permettono di **definire una logica** con la quale **modificare i parametri per intervenire sulle prestazioni,** continuamente **monitorate** da OpenTelemetry

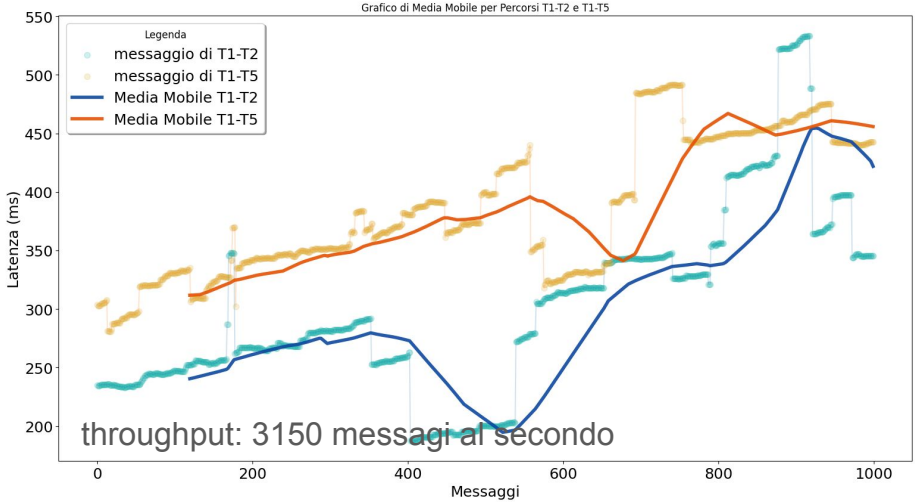


Risultati dei test

Infrastruttura a **un broker e una partizione:**



Infrastruttura a **un broker e dieci partizioni:**

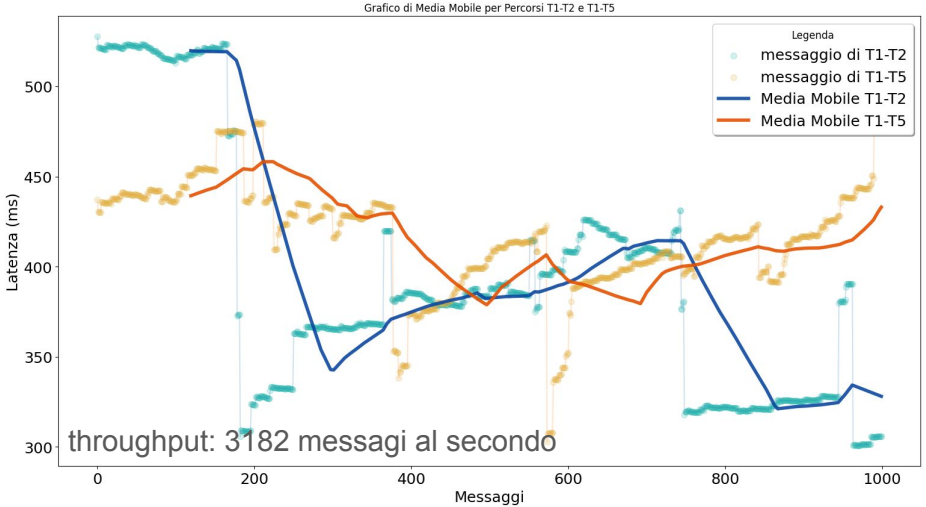


Partiamo con notare il **cambiamento di prestazioni** introducendo **parallelismo tra le topic**, quindi **aumentando le partizioni**.

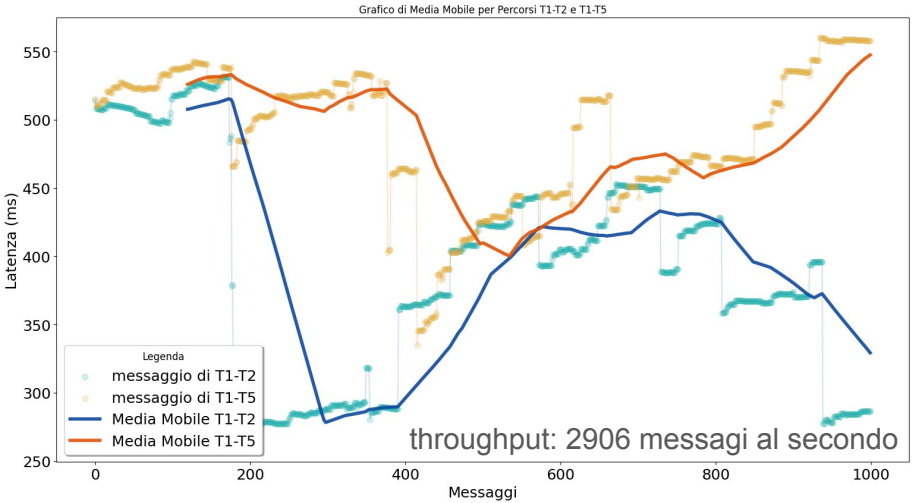
Il sistema partizionato parte in modo **più lento** rispetto a quello singolo, tuttavia riesce a **contenere meglio** la crescita della **latenza** durante l'**aumento del carico.**

Risultati dei test

Infrastruttura a **tre broker e dieci partizioni:**



Infrastruttura a **cinque broker e dieci partizioni:**



Introduciamo ora il **parallelismo anche tra i broker**. Notiamo come il sistema con **cinque broker** abbia **difficoltà** a mantenere **stabili le latenze messaggi**. La **complessità di gestione** di tanti broker e tante partizioni introduce un **overhead** che causa nei messaggi un **aumento della latenza**.

Ora che c**onosciamo l'impatto dei parametri** sul sistema, possiamo **ottimizzarli** per mantenere **alte prestazioni.**

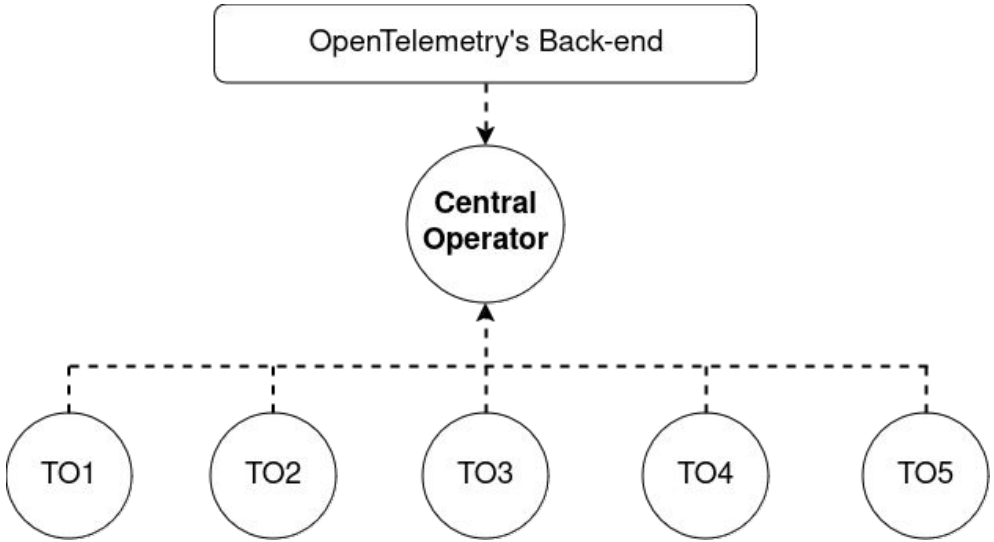
Ruolo degli Operator

I **topic operator**, una volta creata la topic, **inviano** all'operatore centrale informazioni quali:

- numero di **partizioni**
- fattore di **replicazione**
- numero di **client** attivi

L'**operatore centrale** riceve sia questi parametri sia le **telemetrie** da OpenTelemetry e attraverso un **algoritmo di decisione parametrica** calcola il numero ideale di:

- **partizioni**
- **broker** attivi nel sistema



Conclusioni e sviluppi futuri

Opportuno **avere chiare le caratteristiche che si ritengono più significative** per **scegliere l'algoritmo** di gestione del proprio sistema:

- **Sistema robusto**: adatto a **carichi maggiori** ma **messaggi con latenza più alta**.
- **Sistema veloce**: **latenza bassa** a discapito di **sicurezza** e **distribuzione** del carico

Sviluppi futuri: sviluppare **algoritmi di controllo più efficienti** per rendere Kafka completamente **autogestito dall'operatore centrale**, che utilizzerà i dati continuamente forniti da OpenTelemetry e dai topic operator per ottimizzare l'architettura.

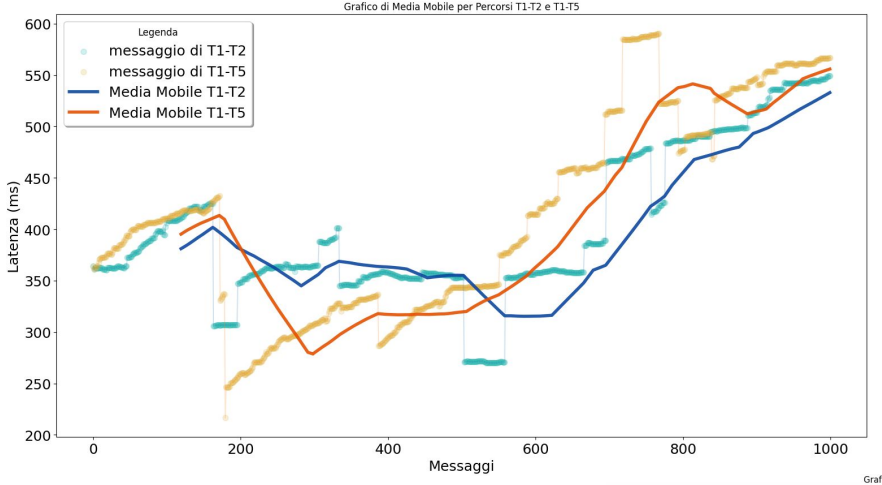
Grazie dell’attenzione

Alessio Giorgetti

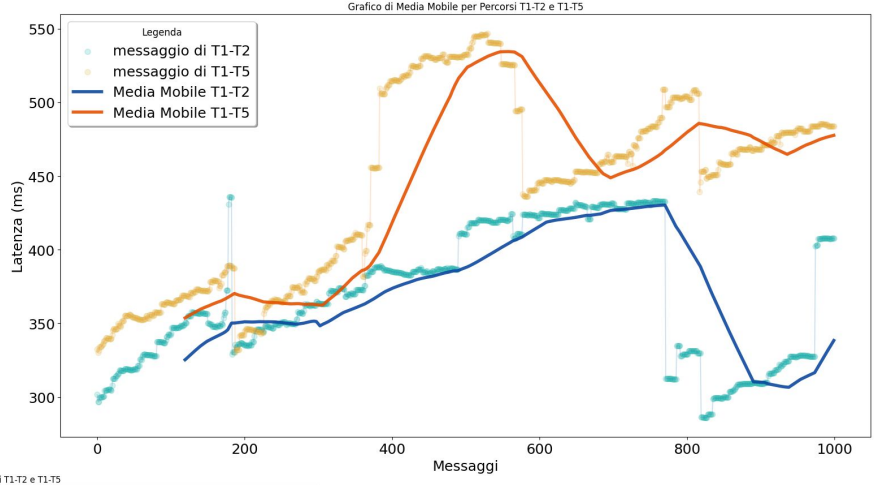
Anno accademico 2023/2024

Risultati dei test

un broker cinque partizioni:



tre broker cinque partizioni:



cinque broker cinque partizioni: