

# Estudio de plataformas de streaming, Spark, Flink

En el análisis de datos procedentes de  
vehículos

**MOVILDATA**

A Verizon company



Facultad  
Informática  
Universidad  
Murcia

UNIVERSIDAD DE  
**MURCIA**





# Introducción

Una de las partes más importantes de una empresa es el procesamiento de los datos que se obtienen en en la misma. En nuestro caso hacemos un servicio de procesamiento de datos a empresas de transporte que quieran monitorizar sus vehículos.

Obteniendo los datos que producen los vehículos podemos orientar a diferentes empresas para aumentar su productividad , reduciendo costes de entre sus trayectos.

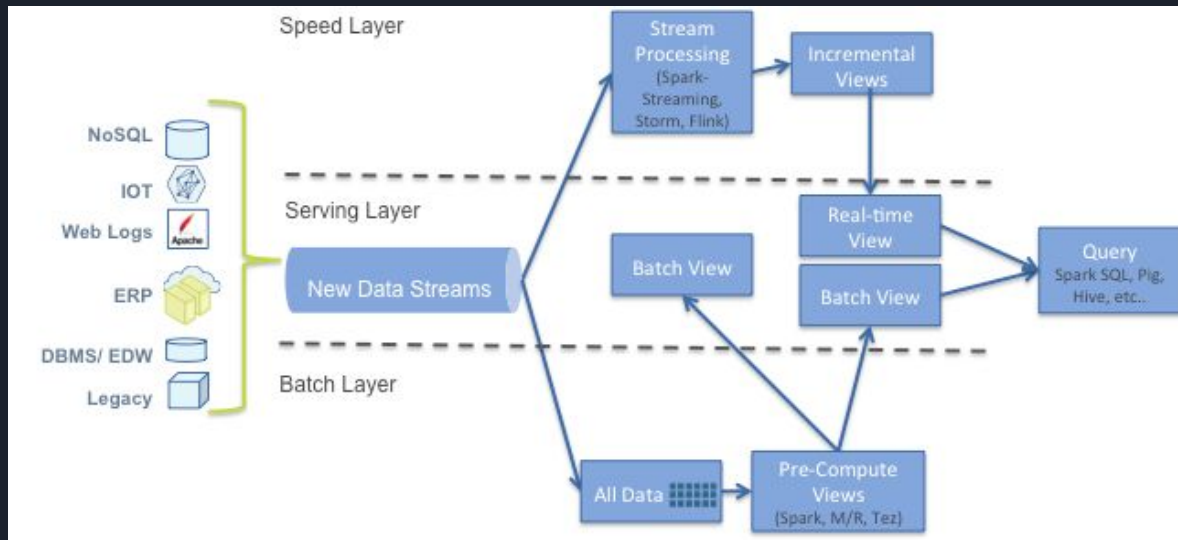


# Problema

- Se pretende buscar una plataforma de streaming para procesar una gran cantidad de datos en tiempo real obteniendo diferentes métricas que finalmente serán mostradas en un dashboard.
- Necesitamos una escalabilidad horizontal debido a la gran cantidad de datos que se van a almacenar. Además necesitamos alta disponibilidad (si cae un servidor, otro cualquiera debe hacer su trabajo) (ej: una granja de servidores pierde la conexión a internet)
- Debemos encontrar una arquitectura que reciban, procesen y almacenen datos a gran velocidad emitiendo notificaciones al mismo tiempo. Una vez hecho eso, habrá otros procesos que resumen la información para realizar informes.
- Los datos proceden de una gran cantidad de vehículos y se necesita una respuesta rápida
- Posibles problemas:
  - Se necesitan predecir posibles accidentes, tiempo estimado de llegada al destino, posibles saturaciones. Además también se puede intentar obtener los excesos de velocidad en la vía.
  - Se podrían obtener alarmas de los diferentes sensores de los vehículos, como alarmas de temperatura en los camiones frigoríficos.

# ¿En qué consiste este proyecto?

Este proyecto se define como la comparación de diferentes tecnologías en arquitecturas de streaming. Para ello se va a hacer uso de una arquitectura lambda que tan de moda está ahora:





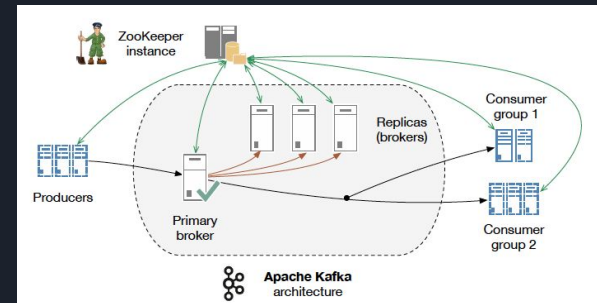
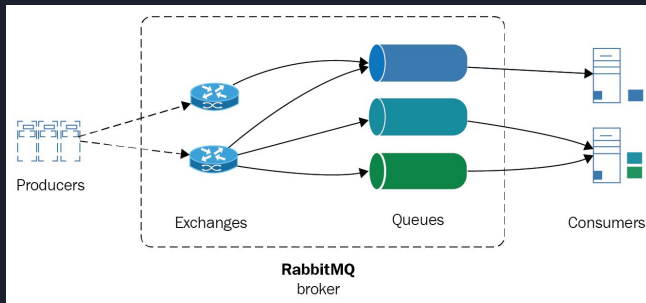
# Definir una arquitectura teórica

- Para repartir la carga de los mensajes enviados/recibidos necesitaremos un distribuidor de los diferentes datos que recibamos (**broker**).
- Para el procesamiento en tiempo real necesitaremos una tecnología para **Streaming** que nos permita procesar el dato bruto y convertirlo en los diferentes datos útiles que se precisan.
- Para poder consultar cualquier dato en cualquier momento necesitaremos almacenar los datos en una o varias bases de datos.
- Para procesar los datos posteriormente necesitamos un sistema de batching que vaya haciendo resúmenes y obteniendo estadísticas.
- Para ofrecer los diferentes servicios necesitamos una plataforma de dashboard que nos ayude a mostrar los datos en tiempo real y los post-procesados. También podríamos añadir informes.
- Cuando el rendimiento empiece a decaer por la carga de trabajo (existan más cantidad de datos por segundo) siempre se podrá añadir un nuevo host que alivie dicha carga.

# Tecnologías para brokers

- Kafka
  - En principio usaremos esta dado que tiene una comunidad más grande en estos entornos. Además sabemos que es fácil de escalar.
- RabbitMQ
  - Si da tiempo lo probaremos para ver si conseguimos mejor rendimiento. En principio parece tener un sistema de colas más complejo. También se suele usar más para la parte servidor->web.

Podría decir que la diferencia fundamental es el tratamiento de las colas y la escalabilidad de las soluciones (rabbitMQ es más vertical mientras que kafka es más horizontal).



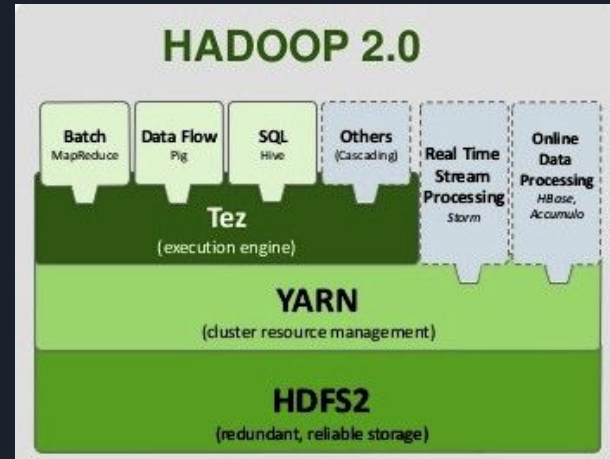


# Tecnologías para streaming

- Spark
  - El más usado ahora mismo. No es streaming puro si no microbatching (simulación de tiempo real)
- Flink
  - Usa streaming puro, no tiene microbatching. (muy usado junto al modelo ELK de ElasticSearch)
- Apex (si da tiempo)
  - El nuevo proyecto de streaming
- OTROS:
  - Storm (microbatching)
  - Samza (streaming puro)

# Tecnologías para batching en Big data

- Logstash
- Spark
- Específicos para hadoop
  - Está demostrado que el motor Tez optimiza las búsquedas en el caso que queramos usar hadoop
- Específicos para otras bases de datos







# Bases de datos

Una parte importante a tener en cuenta es el procesamiento de series temporales en la base de datos. Además también es importante el procesamiento de datos geográficos y la horizontalidad de la solución.

- SQL
  - Postgre
  - Sql Server
- NoSql
  - MongoDB (Documental)
  - Hadoop y ElasticSearch (Motor de búsquedas)
  - InfluxDB (series temporales)
  - Redis (clave valor)

Es posible que dependiendo de lo que queramos hacer en cada momento usemos diferentes BBDD



# Dashboard en tiempo real

- Kibana
- Web hecha a mano
- <https://www.astronomer.io/blog/six-open-source-dashboards/>
- Otros dashboard de pago
- Grafana (se integra muy bien con influxDB)




# Si necesitamos un DataCenter manager

- **Mesos**
  - <http://mesos.apache.org/>
  - Muy sencillo de usar aparentemente
- **Ambari (para gestionar la arquitectura de Big Data)**
  - (<https://siftery.com/product-comparison/apache-mesos-vs-apache-ambari>)



# Propuestas de análisis de los datos

- Predicción de llegadas a puntos negros.
- Predicción de congestiones.
- Posibles excesos de velocidad en la vía.
- Predecir tiempo de llegada (si llega tarde o a la hora prevista).
- Predicción de alarmas (temperatura).
- Emisión de alarmas (temperatura, conducción agresiva, problemas del EBS).
- Estimar en tiempo real las horas de conducción (te podría avisar si te antes de que te vayas a pasar de horas).
- Cualquier cosa que de tiempo a añadir tras montar la arquitectura.



# Necesitaremos un tiempo de respuesta rápido

Para obtener un mejor tiempo de respuesta en el procesamiento pretendo evitar cualquier petición externa que me pueda poner limitaciones. Por ejemplo, vamos a evitar peticiones a google. Para ello intentaremos almacenar la base de datos de OSM para consultarlos localmente:

- <https://planet.openstreetmap.org>
  - Mongo: <https://github.com/iandees/mongosm>



# Algún proyecto de GPS en streaming

- <https://demo.thingsboard.io/dashboards/3d0bf910-ee09-11e6-b619-bb0136cc33d0?publicId=963ab470-34c9-11e7-a7ce-bb0136cc33d0>
-

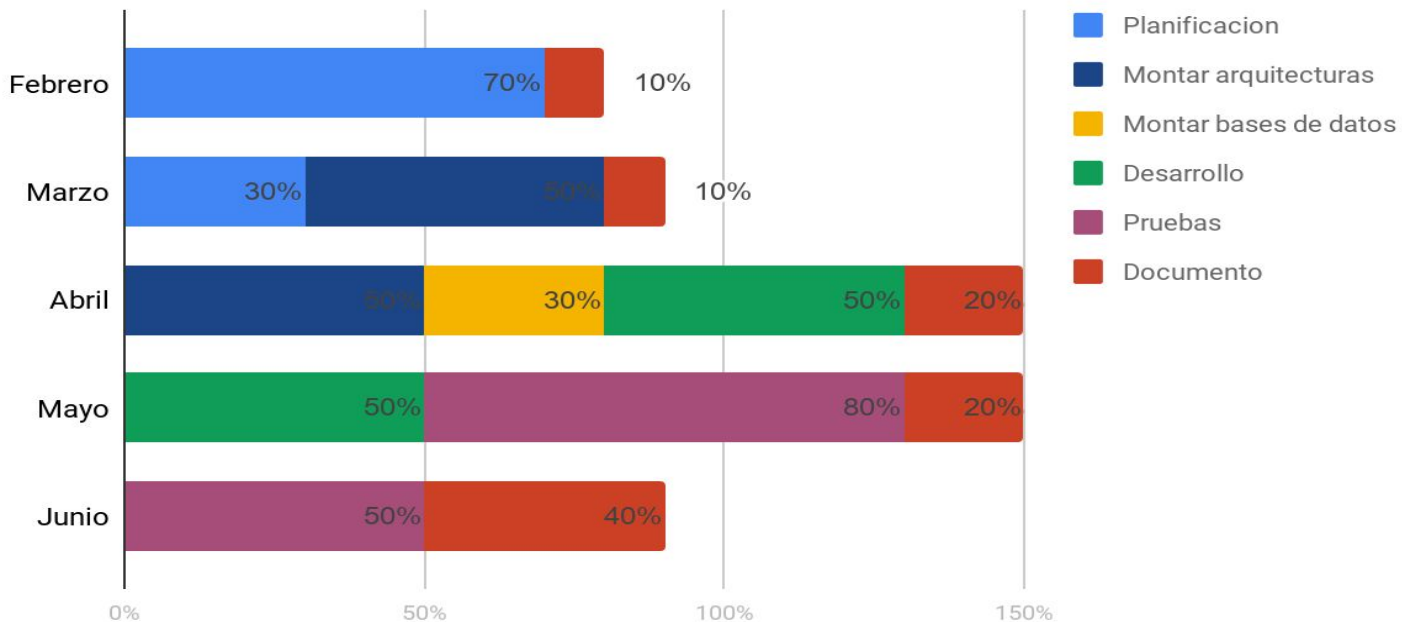


# Visiones de los participantes

- Tutores:
  - Realizar comparaciones entre diferentes tecnologías, probar sus rendimientos y sus usos en las nuevas arquitecturas de software que están surgiendo.
- Empresa
  - Obtener más información útil de sus datos y explorar otras arquitecturas de procesamiento de datos.
- Alumno:
  - Explorar nuevas tecnologías y arquitecturas y obtener un mejor rendimiento de todas ellas. Mejorar su currículum y demostrar lo que ha aprendido tras realizar el máster.
  - Diferenciarse como ingeniero.

# Planificación

## Planificación



Entrega: 25 de junio