## Frameworks de procesamiento de datos.

Estos frameworks procesan los datos del sistema, ya sea leyendo desde almacenamiento no volátil o mientras son consumidos por el sistema. Procesar los datos es el proceso de extraer información y patrones de largas cantidades de “data points” individuales.

Mientras el sistema que maneja esta etapa del ciclo de vida de los datos, es complejo, las metas a grandes rasgos son muy similares: actuar sobre los datos para incrementar la comprensión, resaltar patrones y obtener inputs de interacciones complejas.

Agruparemos estos frameworks por el estado de los datos que están designados a procesar. Algunos sistemas procesan en pedazos (batches), mientras otros procesan los datos continuamente mediante van fluyendo al sistema (streams), algunos pueden procesar de ambas maneras (hybrid).

### Procesamiento por Batch

El procesamiento por batch involucra operar sobre un gran data set estático y regresar el resultado después cuando el procesamiento ha sido completado.

Los data sets en este tipo de procesamiento son generalmente:

- Cerrados: Un batch data set representa una colección finita de datos
- Persistentes: Los datos generalmente están respaldados en algún tipo de almacenamiento permanente
- Largos: Las operaciones en batch son generalmente la única opción para procesar data sets extremadamente grandes.

Es recomendado para cálculos en los que se necesitan sets completos de registros. Por ejemplo, calculando totales o medias, el data set se debe procesar entero en lugar de cada registro individual.

La desventaja de trabajar con grandes cantidades de datos es que generan un tiempo más largo de procesamiento de los datos. Por este detalle, este tipo no es apropiado en situaciones donde el tiempo de procesamiento es especialmente importante.

*Hadoop*

Apache Hadoop y su motor de procesamiento “MapReduce” ofrecen un modelo de procesamiento en batch estable y probado, que es indicado para manejar data sets muy largos donde el tiempo no es un factor importante. El bajo costo de mantenimiento de los componentes necesarios para un cluster de Hadoop funcional, hacen este procesamiento efectivo para muchos casos de uso. La compatibilidad e integración con otros frameworks y motores significan que Hadoop puede servir como la base en diferentes trabajos de procesamiento que usen diferentes tecnologías.

## Sistemas de procesamiento por Stream

Estos frameworks procesan los datos, así como van llegando al sistema. Esto requiere un modelo de procesamiento diferente, en lugar de definir operaciones que apliquen al data set completo, definen operaciones que aplican a cada data point individual, en cuanto este entra al sistema.

Los datasets en este tipo de procesamiento son considerados "abiertos". Esto tiene las siguientes implicaciones:

- El dataset *total* es definido como el total de datos que ha ingresado al sistema hasta el momento.
- El dataset *de trabajo* es un poco más relevante y es limitado a un data point a la vez.
- El procesamiento está basado en eventos y no "*termina*" hasta que es detenido explícitamente. Los resultados están disponibles inmediatamente y se actualizan con cada nuevo flujo de información.

Este tipo de procesamiento se presta más a ciertos tipos de trabajos. Cuando se está procesando con requerimientos casi en "tiempo real" se puede ver el beneficio del modelo de streaming. Datos de análisis, logs de error de servidor o aplicación y otras métricas basadas en tiempo real son un fit natural dado que puede ser crítico para el negocio reaccionar a cambios en estas áreas.

Este procesamiento es indicado para cuando se debe responder a cambios o picos y cuando se está interesado en detectar modas o comportamientos en periodos de tiempo.

*Apache Storm*
Para trabajos de procesamiento puro en stream con requerimientos estrictos de latencia, Storm es probablemente la mejor opción. Puede garantizar el procesamiento de mensajes y puede usarse con bastantes lenguajes de programación.

*Apache Samza*
Es una Buena opciones para trabajos de streaming donde Hadoop y Kafka están ya sea disponibles o sensibles a la implementación. Samza es una buena opción para organizaciones con múltiples equipos usando streams en varias etapas del procesamiento. Samza simplifica muchas partes del procesamiento y ofrece baja latencia, sin embargo, no tan baja como Storm.

## Sistemas de procesamiento Híbrido

Algunos frameworks de procesamiento pueden manejar ambos casos, stream y batch. Estos frameworks simplifican diversos requerimientos de procesamiento al permitir que los mismos componentes o los que se relacionan, se usen para los 2 tipos de datos.
Mientras que los frameworks enfocados en un solo tipo de procesamiento pueden resolver mejor ciertos casos de uso, los frameworks híbridos intentan ofrecer una solución general para

el procesamiento de los datos. Pueden llegar a ofrecer sus propias integraciones, librerías y herramientas para realizar gráficos, machine learning y consultas interactivas.

*Apache Spark*

Spark es una gran opción para trabajos con procesamiento diverso. El procesamiento batch de Spark ofrece ventajas de velocidad y uso de memoria. El procesamiento stream de Spark es una gran solución para trabajos en los que es más importante el rendimiento que la latencia.

*Apache Flink*

Flink provee ambos, procesamiento stream con baja latencia con soporte para tareas tradicionales de batch. Flink puede ser mejor aprovechado en organizaciones que tienen requerimientos fuertes de procesamiento y algunas tareas orientadas a batch. Su compatibilidad con Storm nativo y Hadoop, así como su habilidad de correr en un cluster con YARN (Yet Another Resource Negotiator) lo hacen fácil de evaluar.

## Conclusión

Hay varias opciones de procesamiento en sistemas de big data.

Para trabajos exclusivos de batch que no son sensibles al tiempo de procesamiento, Hadoop es una buena opción que también es más fácil de implementar que otras.

Para trabajos exclusivos de stream, Storm tiene soporte a muchos lenguajes de programación y puede entregar procesamiento de muy baja latencia, pero puede ocasionar duplicados y no puede garantizar un orden en su configuración por default. Samza se integra bien con YARN y Kafka para poder entregar flexibilidad y uso entre varios equipos.

Para trabajos mixtos, Spark entrega procesamiento en batch de alta velocidad así como procesamiento micro-batch para necesidades de streaming. Tiene mucho soporte, librerías integradas y herramientas extras.

Flink tiene un procesamiento stream puro con soporte para procesamiento batch. Está muy optimizado, puede correr tareas escritas para otras plataformas y tiene procesamiento de baja latencia, pero sigo en sus primeras etapas de adopción.

La mejor opción para tu situación dependerá fuertemente del estado de los datos que tengas que procesar, de tus requerimientos y de los problemas que quieras resolver.