Taller Big-Data

Docente: Jesus Ariel

Estudiante: Paula Andrea Terrios

Corporacion Universitaria del Huila (CORHUILA)

Ciencia de Datos

Neiva – Huila

2025

Documento Comparativo de Arquitecturas Big Data

1. Introducción
Propósito

El crecimiento exponencial de los datos en sectores como el comercio electrónico, la salud, la banca y las ciudades inteligentes ha impulsado la necesidad de arquitecturas Big Data capaces de almacenar, procesar y analizar información en grandes volúmenes, velocidades y variedades. Comparar arquitecturas permite identificar la solución más adecuada según las necesidades de escalabilidad, costo, facilidad de uso y casos de aplicación.

Alcance

Este documento compara tres arquitecturas Big Data representativas:

- Hadoop: ecosistema basado en almacenamiento distribuido y procesamiento batch.
- Apache Spark: motor de procesamiento en memoria con soporte para batch y streaming.
- Arquitecturas en la nube (AWS como ejemplo): servicios gestionados que facilitan la implementación de entornos Big Data sin necesidad de administrar infraestructura propia.

Glosario

- HDFS (Hadoop Distributed File System): sistema de archivos distribuido para almacenar grandes volúmenes de datos.
- YARN (Yet Another Resource Negotiator): gestor de recursos y planificador de trabajos en Hadoop.
- MapReduce: modelo de programación para procesar datos en paralelo.
- RDD (Resilient Distributed Dataset): estructura de datos inmutable de Spark para procesamiento distribuido en memoria.
- ETL (Extract, Transform, Load): proceso de extracción, transformación y carga de datos.

Descripción de Arquitecturas
Hadoop

Hadoop es un framework de código abierto diseñado para almacenar y procesar grandes volúmenes de datos en clústeres de hardware commodity.

- Componentes principales:
  - HDFS: almacenamiento distribuido tolerante a fallos.
  - YARN: gestión de recursos y planificación de tareas.
  - MapReduce: paradigma de procesamiento batch.
- Ventajas: Alta capacidad de almacenamiento y procesamiento batch confiable.
- Limitaciones: Latencia alta, curva de aprendizaje pronunciada, complejidad en la gestión de clústeres.

Apache Spark

Spark es un motor de análisis en memoria que mejora el rendimiento de Hadoop al reducir la latencia.

- Componentes principales:
  - RDD y DataFrames para procesamiento distribuido.
  - Librerías integradas: SparkSQL, MLlib, GraphX, Structured Streaming.
  - APIs en Scala, Java, Python y R.
- Ventajas: Procesamiento en tiempo real, APIs más amigables, integración con machine learning.
- Limitaciones: Requiere más memoria RAM, costos mayores en hardware si se despliega on-premise.

Arquitecturas en la Nube (ejemplo: AWS)

La nube ofrece entornos Big Data gestionados que eliminan la necesidad de administrar infraestructura propia.

- Componentes principales:
  - AWS EMR (Elastic MapReduce): clústeres gestionados para Hadoop y Spark.
  - AWS S3: almacenamiento masivo y económico.
  - AWS Kinesis: ingesta y análisis de datos en streaming.
  - AWS Redshift / BigQuery / Synapse: data warehouse escalables para analítica avanzada.
- Ventajas: Escalabilidad automática, pago por uso, facilidad de integración con otros servicios.
- Limitaciones: Dependencia del proveedor, costos variables a gran escala.

Conclucion

La arquitectura propuesta presenta un flujo organizado de datos, que va desde la ingesta en tiempo real y por lotes, hasta el almacenamiento, procesamiento y consumo de información mediante dashboards y reportes. Esto permite responder a las necesidades de un e-commerce moderno, donde la velocidad y la precisión en el análisis de datos son claves para mejorar la experiencia del cliente, con unas pautas:

Escalabilidad: El uso de un Data Lake junto con bases NoSQL facilita la integración de grandes volúmenes de datos heterogéneos (transacciones, IoT, redes sociales).

Flexibilidad: Se combinan procesamientos batch y streaming, lo que permite tanto análisis históricos como respuestas en tiempo real.

Orientación al negocio: La capa de consumo traduce la complejidad técnica en información útil para la toma de decisiones estratégicas (ej. gestión de inventarios, personalización de ofertas).

Complejidad técnica: Requiere un equipo especializado en Big Data, lo que implica altos costos iniciales de implementación y mantenimiento.

Seguridad y cumplimiento: El manejo de datos sensibles de clientes obliga a cumplir normativas (ej. GDPR, PCI-DSS), lo que añade carga administrativa y tecnológica.

Dependencia tecnológica: La arquitectura depende de múltiples servicios (Kafka, Spark, BigQuery, etc.), lo que puede generar riesgos de integración y dependencia de proveedores cloud.

Si se optimiza la gobernanza de datos y se implementan políticas de seguridad desde el diseño, esta arquitectura puede evolucionar hacia un ecosistema de inteligencia artificial, incorporando modelos predictivos y sistemas de recomendación en tiempo real para anticipar la demanda y personalizar aún más la experiencia de compra.

En conclusión, la arquitectura propuesta no solo satisface las necesidades actuales del e-commerce, sino que también abre la puerta a una evolución futura hacia analítica avanzada e inteligencia de negocio en tiempo real, aunque exige una estrategia sólida en talento, seguridad y gestión de costos.

Bibliografia

https://www.oracle.com/latam/big-data/what-is-big-data/
https://www.obsbusiness.school/blog/big-data-y-sus-principales-aplicaciones-beneficios-yejemplos

https://www.teamcore.net/es/blog/5-empresas-que-usan-big-data-y-han-conseguido-losmejores-resultados/ https://www.planttext.com/?utm_source=chatgpt.com