Using Kafka in Big Data Applications

What is Big Data?

Big data is data that contains greater variety, arriving in increasing volumes and with more velocity. This is also known as the three Vs.

Put simply, big data is larger, more complex data sets, especially from new data sources. These data sets are so voluminous that traditional data processing software just can’t manage them. But these massive volumes of data can be used to address business problems you wouldn’t have been able to tackle before.

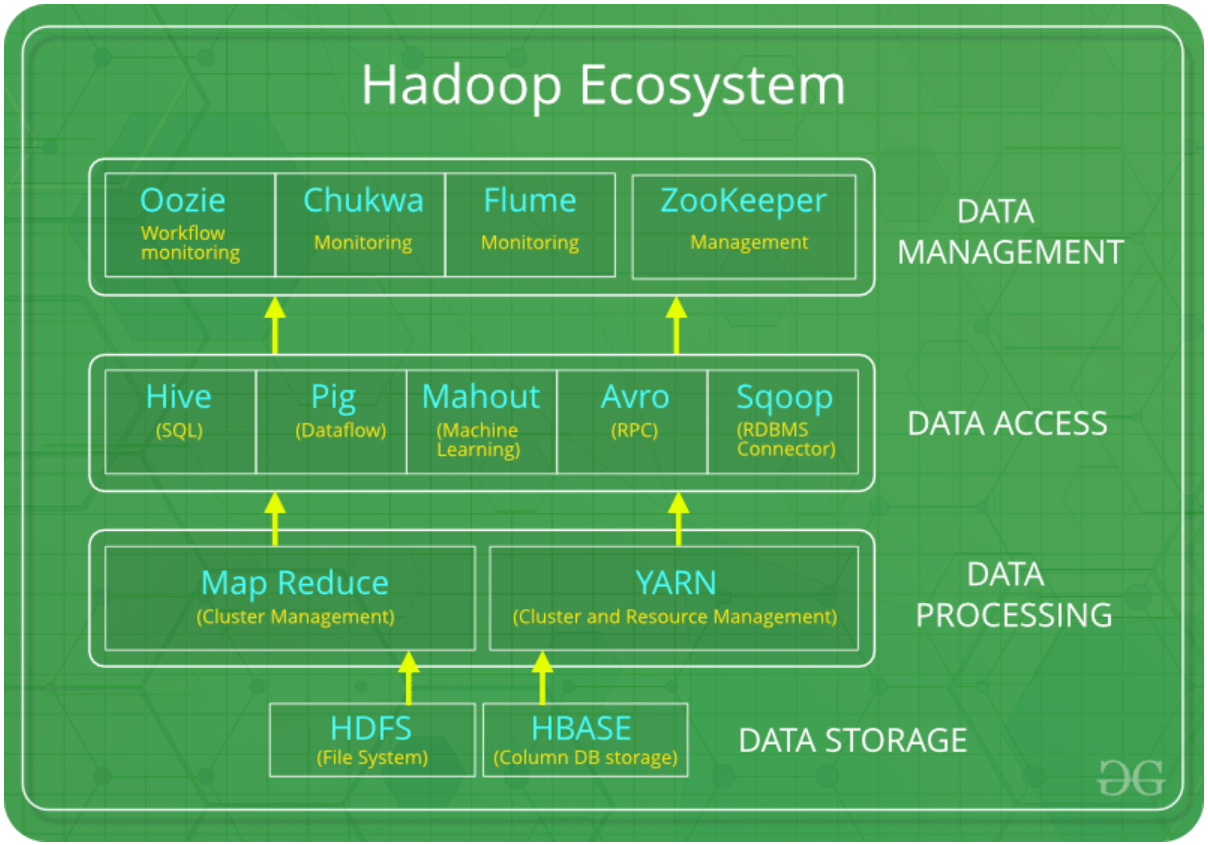
The three Vs of big data

Volume	The amount of data matters. With big data, you'll have to process high volumes of low-density, unstructured data. This can be data of unknown value, such as Twitter data feeds, clickstreams on a web page or a mobile app, or sensor-enabled equipment. For some organizations, this might be tens of terabytes of data. For others, it may be hundreds of petabytes.
Velocity	Velocity is the fast rate at which data is received and (perhaps) acted on. Normally, the highest velocity of data streams directly into memory versus being written to disk. Some internet-enabled smart products operate in real time or near real time and will require real-time evaluation and action.
Variety	Variety refers to the many types of data that are available. Traditional data types were structured and fit neatly in a relational database. With the rise of big data, data comes in new unstructured data types. Unstructured and semistructured data types, such as text, audio, and video, require additional preprocessing to derive meaning and support metadata.

Sector	Sample Use-cases
Retail	• What are customers shopping/buying patterns? • What is the effectiveness of various promotions?
Banking	• What transactions look fraudulent? • What is the risk profile of new accounts?
Manufacturing	• Where are the bottlenecks in the supply chain? • How good is our Quality Control?
Healthcare	• Do genetics affect outcomes of medical trials? • How effective are drugs in different patients?
Energy	• Monitor safety of rigs, wells, distribution facilities • What is the change in energy consumption over time?

Apache Hadoop is an open source framework intended to make interaction with big data easier.

Hadoop allows to store and process big data in a distributed environment across clusters of computers using simple programming models. It is designed to scale up from single servers to thousands of machines, each offering local computation and storage.


Hadoop Ecosystem
Oozie
Workflow monitoring
Chukwa
Monitoring
Flume
Monitoring
ZooKeeper
Management
DATA MANAGEMENT
Hive
(SQL)
Pig
(Dataflow)
Mahout
(Machine Learning)
Avro
(RPC)
Sqoop
(RDBMS Connector)
DATA ACCESS
Map Reduce
(Cluster Management)
YARN
(Cluster and Resource Management)
DATA PROCESSING
HDFS
(File System)
HBASE
(Column DB storage)
DATA STORAGE

Following are the components that collectively form a Hadoop ecosystem:

- **HDFS:** Hadoop Distributed File System
- **YARN:** Yet Another Resource Negotiator
- **MapReduce:** Programming based Data Processing
- **Spark:** In-Memory data processing
- **PIG, HIVE:** Query based processing of data services
- **HBase:** NoSQL Database
- **Mahout, Spark MLLib:** Machine Learning algorithm libraries
- **Solar, Lucene:** Searching and Indexing
- **Zookeeper:** Managing cluster
- **Oozie:** Job Scheduling

Introduction to Apache Kafka

- Kafka is a distributed event streaming platform.
- Core Capabilities: publish, subscribe, store, process streams.
- Acts as central nervous system for data architectures.

Kafka in Big Data Ecosystem

- Data Sources: logs, IoT sensors, databases, social media feeds.
- Integration with Hadoop, Spark, Flink, Cassandra, MongoDB.
- Pipeline Example: Sensors → Kafka → Spark Streaming → HDFS → ML Model.

Managing High Volumes in Kafka

- Kafka topics are partitioned for horizontal scaling.
- Handles millions of messages per second.
- Configurable log retention allows replay of messages.
- Best Practices: multiple partitions, balance replication, compression.

Kafka Message Delivery Semantics

- At-most-once: Message delivered zero or one time.
- At-least-once: Message delivered one or more times (possible duplicates).
- Exactly-once: Message delivered only once with transactions.

Hadoop vs Kafka

Developers describe Hadoop as "Open-source software for reliable, scalable, distributed computing".

The Apache Hadoop software library is a framework that allows for the distributed processing of large data sets across clusters of computers using simple programming models. It is designed to scale up from single servers to thousands of machines, each offering local computation and storage.

On the other hand, Kafka is detailed as "Distributed, fault tolerant, high throughput pub-sub messaging system". Kafka is a distributed, partitioned, replicated commit log service. It provides the functionality of a messaging system, but with a unique design.

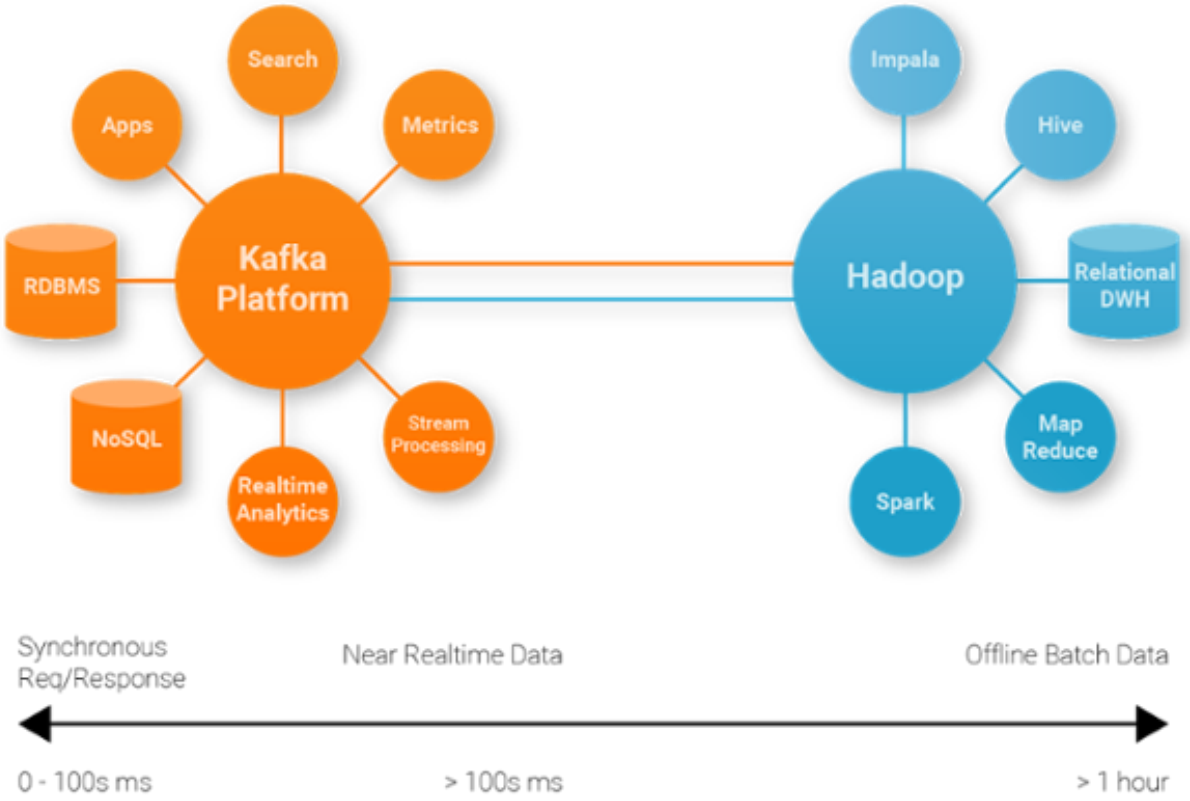
Hadoop and Kafka are primarily classified as "Databases" and "Message Queue" tools respectively.

Big Data & Kafka – Common Usage Patterns

- Log Aggregation: Collect logs → Kafka → Analysis.
- Real-time Analytics: Kafka + Spark/Flink dashboards.
- Data Lake Ingestion: Kafka Connect → HDFS, S3, cloud storage.
- Event Sourcing: Store all state changes as Kafka events.
- Machine Learning Pipelines: Stream training data → update models.

Sr.No	Spark streaming	Kafka Streams
1	Data received form live input data streams is Divided into Micro-batched for processing.	processes per data stream(real real-time)
2	Separated processing Cluster is requried	No separated processing cluster is requried.
3	Needs re-configuration for Scaling	Scales easily by just adding java processes, No reconfiguration requried.
4	At least one semantics	Exactly one semantics
5	Spark streaming is better at processing group of rows(groups,by,ml,window functions etc.)	Kafka streams provides true a-record-at-a-time processing capabilities. it's better for functions like rows parsing, data cleansing etc.
6	Spark streaming is standalone framework.	Kafka stream can be used as part of microservice,as it's just a library.

Kafka is suitable for microservices integration use cases and have wider flexibility.
Spark streaming is suitable for requirements with batch processing for massive datasets, for bulk processing and have use-cases more than just data streaming.


Search
Apps
Metrics
RDBMS
Kafka Platform
NoSQL
Stream Processing
Realtime Analytics
Impala
Hive
Hadoop
Relational DWH
Map Reduce
Spark
Synchronous Req/Response
Near Realtime Data
Offline Batch Data
0 - 100s ms
> 100s ms
> 1 hour

Kafka and Data Governance

- Challenges: ownership, quality, privacy, compliance.
- Schema Registry: enforce schemas, prevent breaking changes.
- Access Control: ACLs, RBAC for topic security.
- Audit & Monitoring: track data access/production.
- Data Lineage for transformation tracking.

Benefits of Kafka in Big Data

- High throughput & low latency.
- Scalable & fault tolerant.
- Real-time + batch integration.
- Flexible delivery semantics.
- Strong ecosystem: Streams, Connect, Schema Registry.

Challenges & Considerations

- Operational complexity at scale.
- Data governance overhead.
- Need for schema evolution management.
- Monitoring lag & partition imbalance.

Real-World Use Cases

- LinkedIn: Activity feed, metrics, log processing.
- Netflix: Real-time recommendations, monitoring.
- Uber: Event streaming for ride tracking.
- Banking/Finance: Fraud detection, real-time monitoring.

Conclusion

- Kafka is backbone of real-time Big Data architectures.
- Reliable, scalable, and flexible event streaming.
- Critical for data-driven decision-making in enterprises.