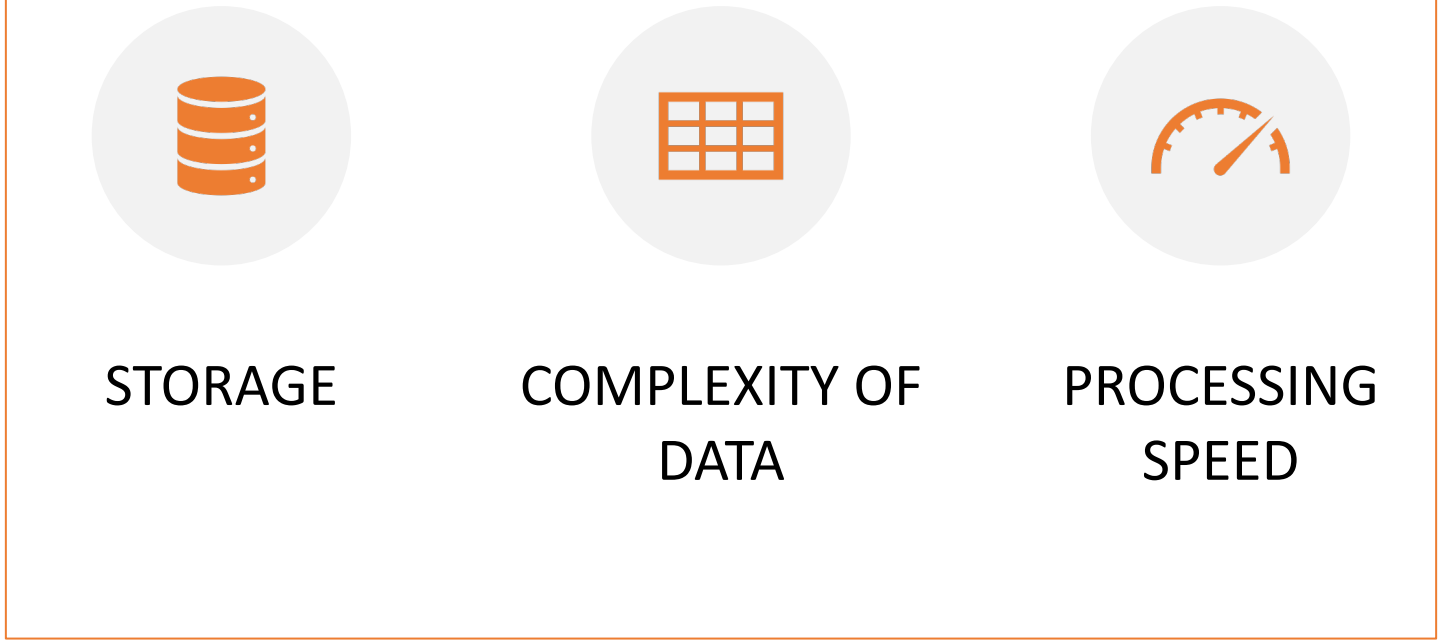
Big Data Analytics

- Ravi Kumar Gupta
- https://kravigupta.in

Big Data challenges [Recap]

- Data Quality
- Cost
- Data integration
- Storage and processing
- Data Security and Privacy
- Data Analysis and Interpretation
- Real time analysis
- Data Governance

Big Data challenges [Recap]



Problem Statement

Process big data with reasonable cost and time?

Reading and Processing Time

Reading the Data – Sequential, Single Node

- General Hard Disks support up to 100MBPS
- SSD supports up to ~700 MBPS

- Data to Read – 10 TB
- SSD => 10*1024*1024 / (60*700) => Approx 250 Minutes
- HDD => 10*1024*1024 / (60*100) => Approx 1750 Minutes

Reading the Data – Parallel, 5 Nodes

- General Hard Disks support up to 100MBPS
- SSD supports up to ~700 MBPS

- Data to Read – 10 TB
- Single Node to read 2 TB
- SSD => 2*1024*1024 / (60*700) => Approx 50 Minutes
- HDD => 2*1024*1024 / (60*100) => Approx 350 Minutes

Network transfer rates limit the reading speed. For SSD 700MBPS may not achieve.

Cost

Reading 100 TB Data

- Data to Read – 100 TB
- Single Node with SSD (700MBPS) => Approx 42 Hours
- 1000 Nodes with SSD (700MBPS) => Approx 2.5 Minutes

- SSD – Costly Affair

- HDD
- Single Node with HDD (100MBPS) => Approx 12 Days
- 1000 Nodes with HDD (100MBPS) => Approx 17 Minutes

Reading 100 TB Data

- 1000 Nodes
  - SSD => 2.5 minutes approx.
  - HDD=> 17 minutes approx.

- Although HDD is taking almost 7 times more time, still
  - Not too much difference
  - Comparing the cost, waiting for 17 min is preferable

- **Reasonable?**

Network transfer rates limit the reading speed. For SSD 700MBPS may not achieve.

Bottlenecks

- Network Data Transfer rates
- For 100 TB, 1000 nodes
  - Remote storage with 10MBPS => 165 Minutes
  - Local storage with 50MBPS => 33 Minutes
  - Better to move computation rather than data
- Machine Failure
  - Need Fault Tolerance

THE

APACHE®

SOFTWARE FOUNDATION

ESTABLISHED 1999

APACHE PROJECT LIST

Overview
All Projects

BY CATEGORY

Attic
Big Data
Build Management
Cloud
Content
Databases
FTP
Graphics
HTTP
HTTP-module
Incubating
JavaEE
Libraries
Mail
Mobile
Network-client
Network-server
OSGi
RegExp
Retired
Search
Security
SQL
Testing
Virtual-machine
Web-framework
XML

BY NAME

HTTP Server

A
AGE
APISIX
Accumulo
ActiveMQ
Airavata
Airflow
Allura
Ambari
Ant
Archiva
Aries
Arrow
AsterixDB
Atlas
Attic
Avro
Axis

B
BVal
Bahir
Beam
Bigtop
Bloodhound
BookKeeper
Brooklyn
BuildStream
bRPC

C
CXF
Calcite
Camel
CarbonData
Cassandra
Causeway
Cayenne
Celix
CloudStack
Cocoon
Commons
Community
Development
Cordova
CouchDB
Creadur
Curator
cTAKES

D
DB
Daffodil
DataFu
DataSketches
DeltaSpike
Directory
DolphinScheduler
Doris
Drill
Druid
Dubbo

E
ECharts
Empire-db
EventMesh

F
Felix
Fineract
Flagon
Flex
Flink
Flume
Fluo
FreeMarker

G
Geode
Geronimo
Giraph
Gobblin
Gora
Griffin
Groovy
Guacamole
Gump

H
HAWQ
HBase
Hadoop
Helix
Hive
Hop
HttpComponents
Hudi

I
Iceberg
Ignite
Impala
InLong
Incubator
IoTDB

J
JMeter
JSPWiki
Jackrabbit
James
Jena
Johnzon
Juneau
jclouds

K
Kafka
Karaf
Kibble
Knox
Kudu
Kvrocks
Kylin
Kyuubi

L
Libcloud
Linkis
Logging Services
Lucene
Lucene.Net

M
MADlib
MINA
MXNet
Mahout
ManifoldCF
Maven
Mesos
Mnemonic
MyFaces
Mynewt

N
NetBeans
NiFi
Nutch
NuttX

O
OFBiz
ORC
Olingo
Oozie
OpenJPA
OpenMeetings
OpenNLP
OpenOffice
OpenWebBeans
OpenWhisk
Ozone

P
PDFBox
PLC4X
POI
Parquet
Perl
Petri
Phoenix
Pig
Pinot
Pivot
Portable Runtime (APR)
Portals
Pulsar

Q
Qpid

R
Ranger
Ratis
RocketMQ
Roller
Royale
Rya

S
SINGA
SIS
Samza
Santuario
SeaTunnel
Sedona
Serf
ServiceComb
ServiceMix
ShardingSphere
ShenYu
Shiro
SkyWalking
Sling
Solr
SpamAssassin
Spark
Steve
Storm
StreamPipes
Streams
Struts
Submarine
Subversion
Superset
Synapse
Syncope
SystemDS

T
TVM
Tapestry
Tcl
Tez
Thrift
Tika
TinkerPop
TomEE
Tomcat
Traffic Control
Traffic Server
Turbine

U
UIMA
Unomi

V
VCL
Velocity

W
Web Services
Whimsy
Wicket

X
XML Graphics
Xalan
Xerces

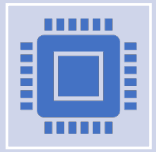
Y
Yetus
YuniKorn

Z
Zeppelin
ZooKeeper


hadoop

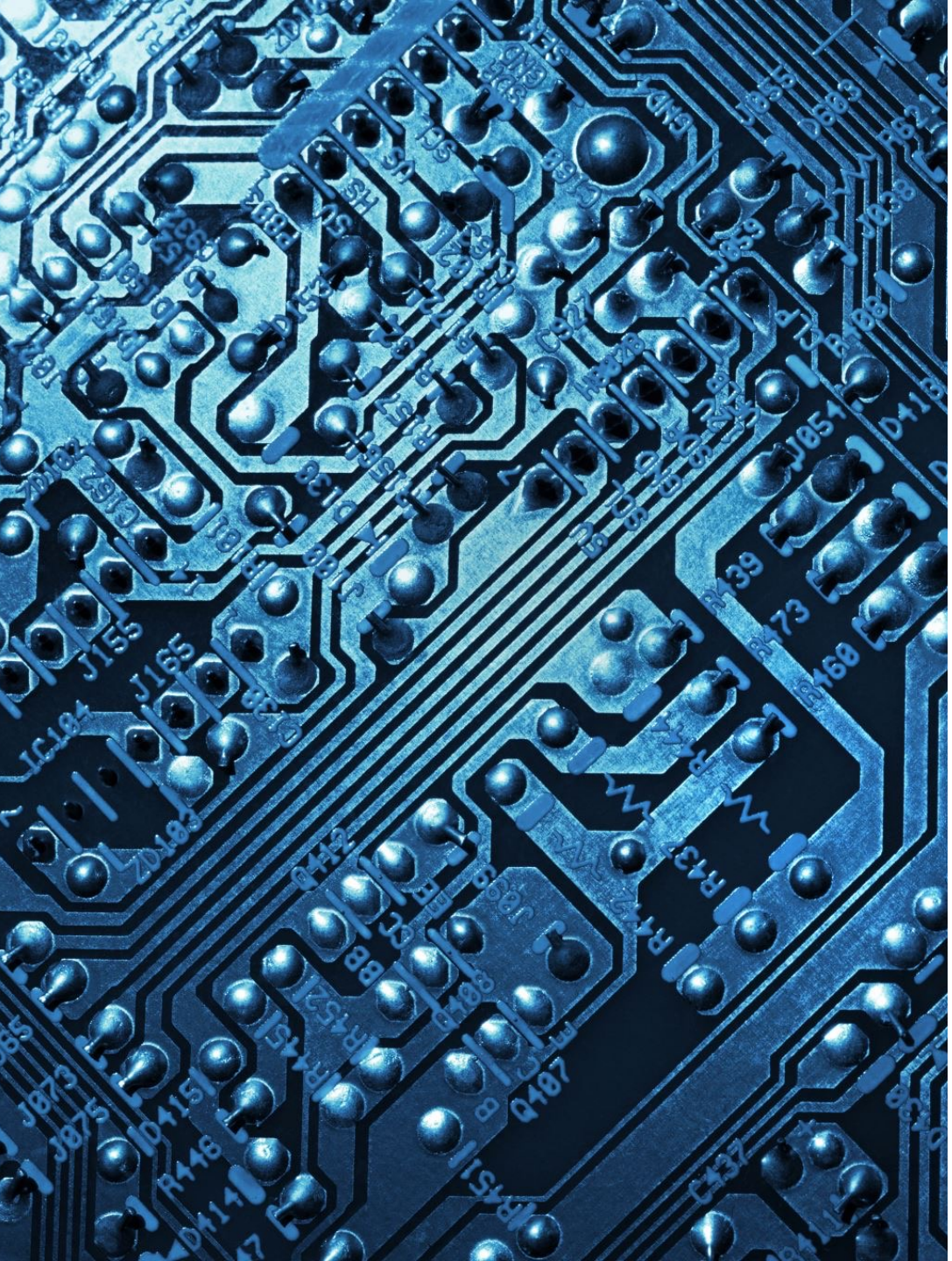
From Apache..

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.

Rather than rely on hardware to deliver high-availability, the library itself is designed to detect and handle failures at the application layer, so delivering a highly-available service on top of a cluster of computers, each of which may be prone to failures.



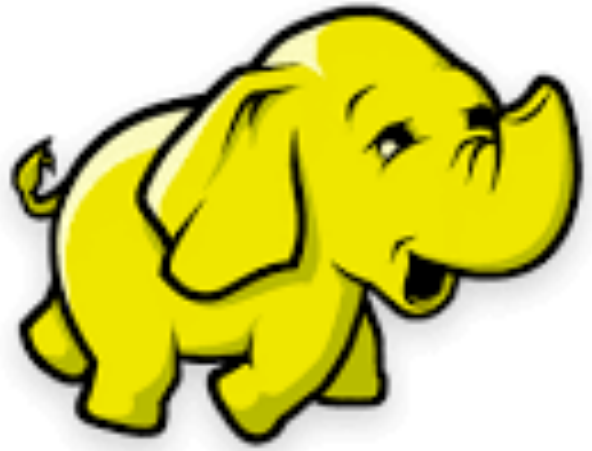
What is Hadoop?

- Open-Source software platform
- Distributed storage and processing
- On computer clusters built using
- Commodity hardware

- Commodity Hardware => Cheaper hardware
- Computer Clusters => Group of computers in a network
  - Usually for single or similar purpose

About Hadoop

- Google
  - 2003 – Paper on GFS, Google File System
  - 2004 – Paper on Map Reduce
  - Launched GFS and Map Reduce
- 2006, Doug Cutting at Yahoo along with Mike Cafarella Made Hadoop
  - Originally designed to support distribution for Nutch
  - Open Source
  - Named after Doug's son's Toy Elephant
- Apache Projects



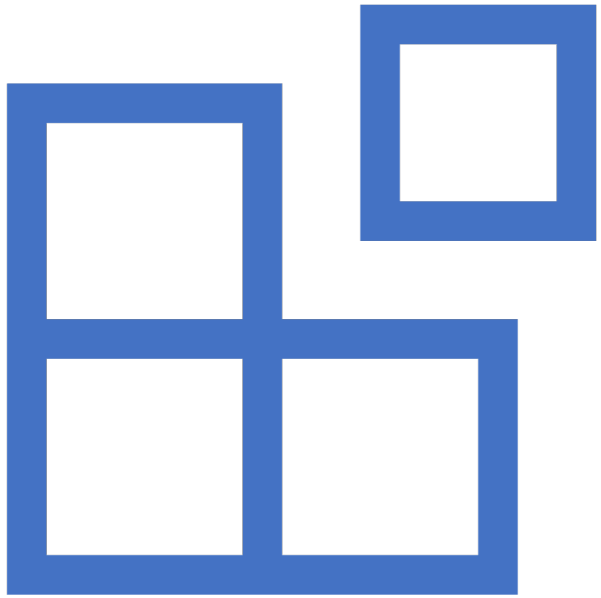
Creators

Mike cafarella and Doug Cutting

Hadoop Goals

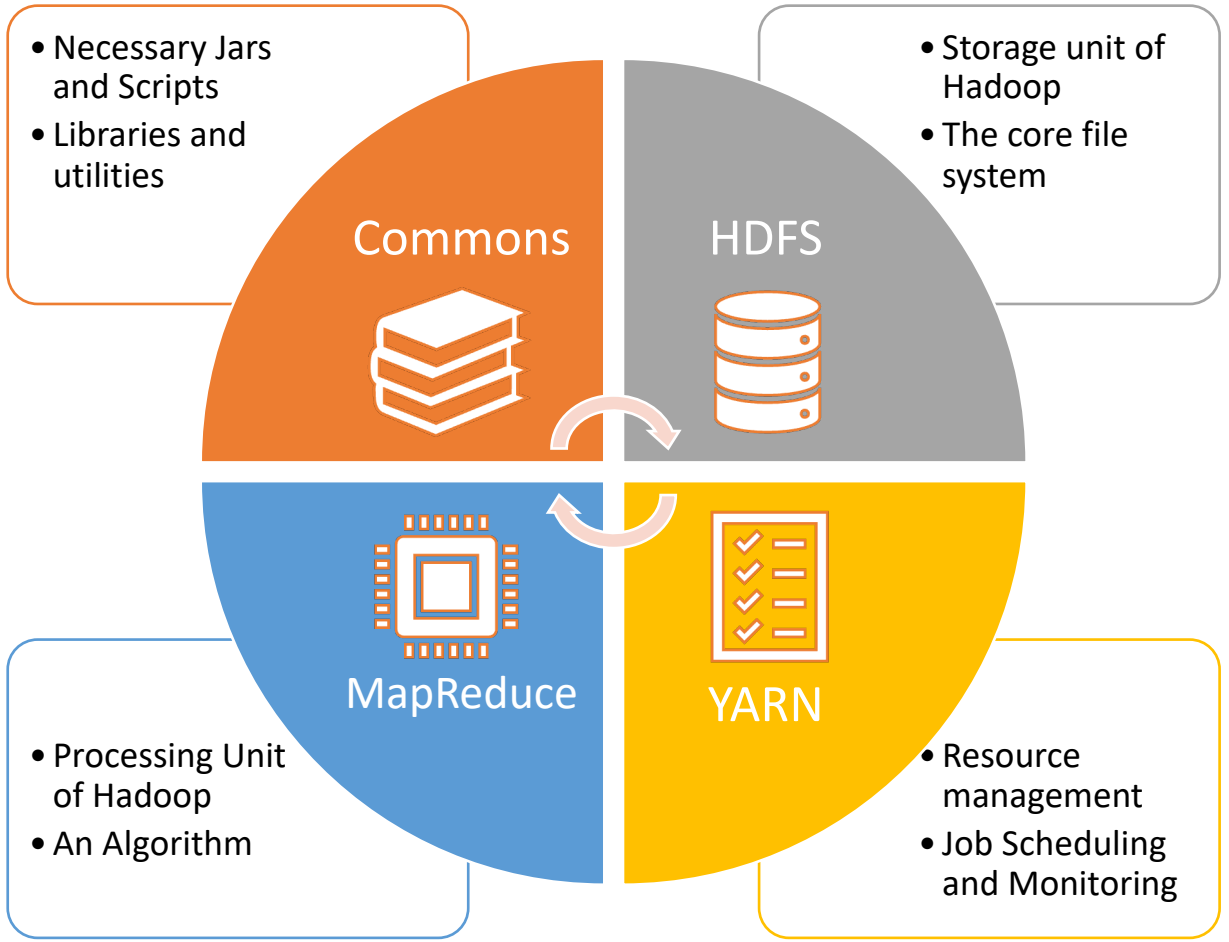
- Faster Data processing
- Scalability
- Cost
- Fault Tolerance
- Ability to handle Hardware Failures

Hadoop Components



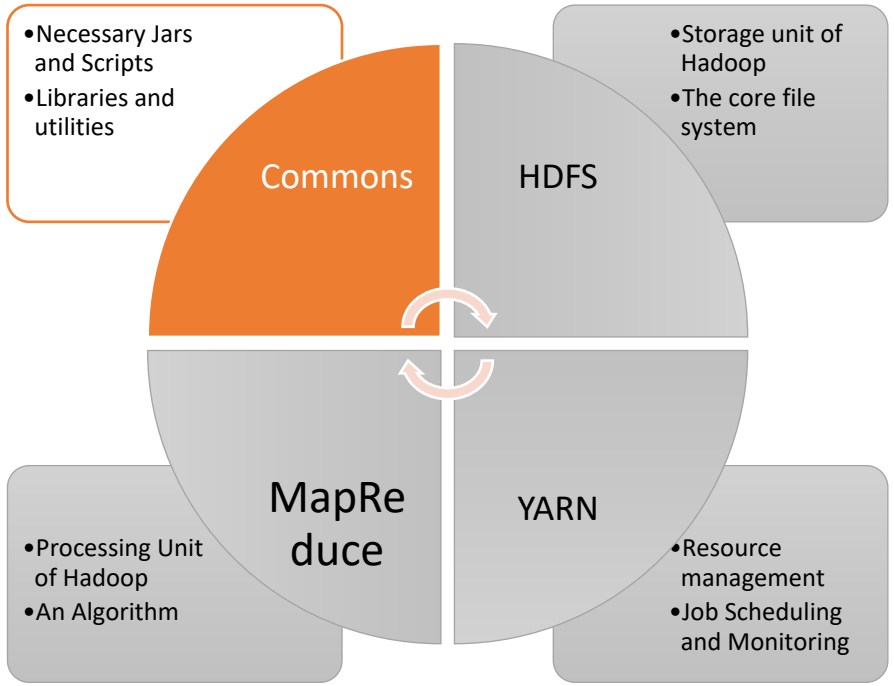
Hadoop Components

- Hadoop Common
- Hadoop Distributed File System (HDFS)
- Hadoop MapReduce
- Hadoop Yet Another Resource Navigator (YARN)



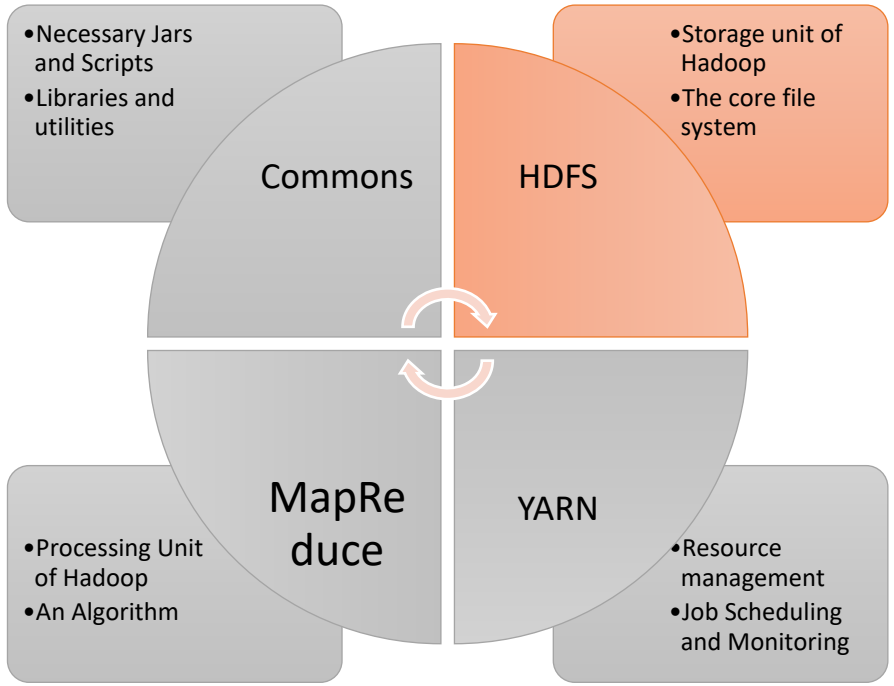
Hadoop Commons

- Provides libraries and utilities
- Necessary java jar files and scripts for start/stop Hadoop
- Acts as a bridge between
  - Hadoop modules
  - And Operating system layer



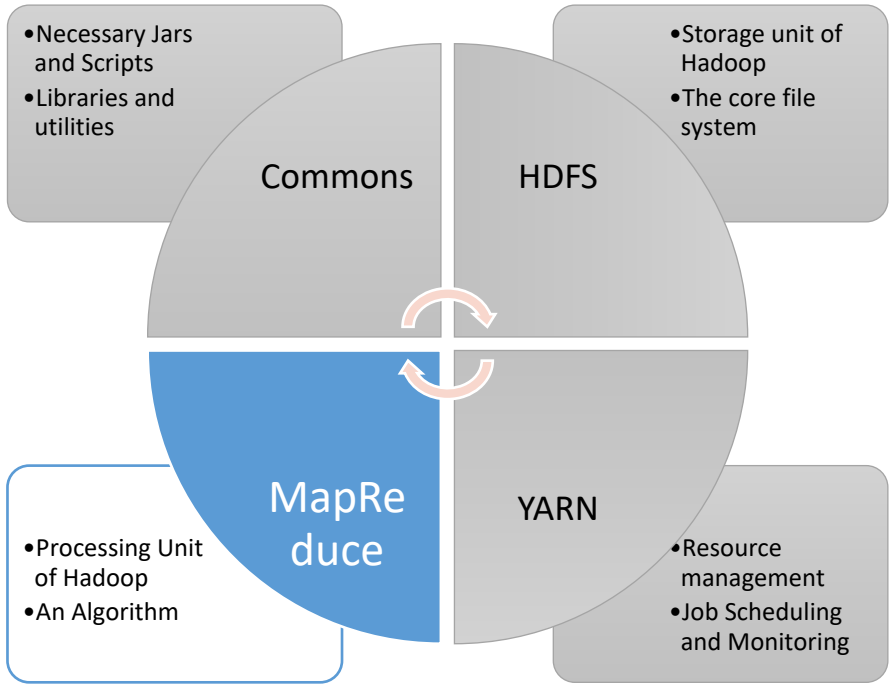
HDFS

- Based on GFS – Google file system
- Stores large data sets
- High-throughput access to application data
- Distributed file system
  - Can span multiple nodes in a cluster
- Reliable
- Deals with faults and failures
  - at application layer



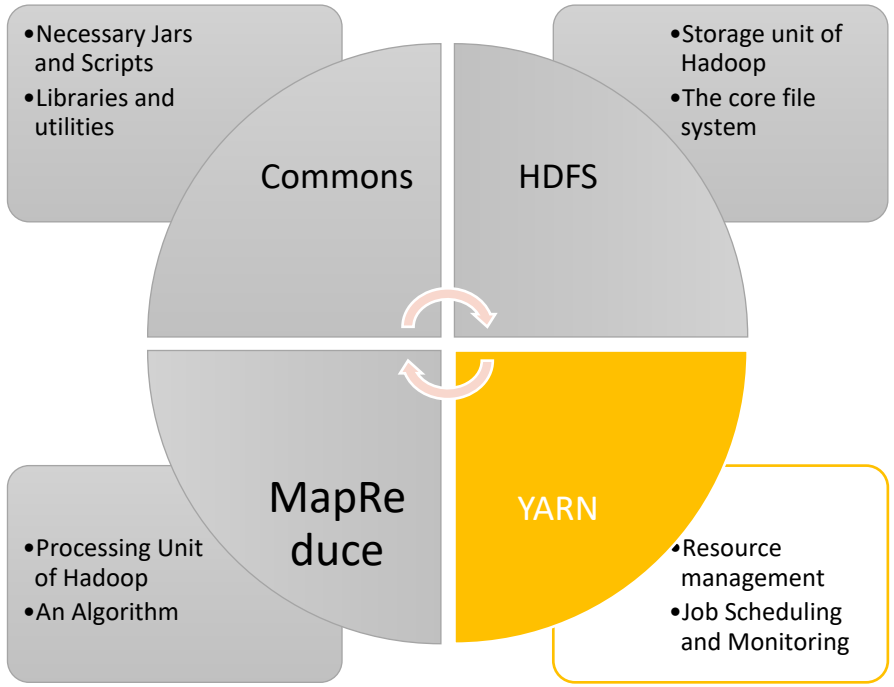
MapReduce

- Simple yet powerful model
- Breaks down bigger tasks into smaller tasks
- Two phases –
  - Map : Filtering and Sorting
  - Reduce: Combine / Summary function
- Distributed Processing
  - Each node processes small data



YARN

- Yet Another Resource Navigator
- Cluster Management Layer
- Does resource allocation
  - to various applications in Hadoop Cluster
- Task Scheduler and Management
- Decouples functions of MapReduce
  - Resource Management
  - Scheduling Capabilities



Hadoop


Hadoop v1.0
MapReduce
Data Processing
& Resource Management
HDFS
Distributed File Storage


Hadoop v2.0
MapReduce
Other Data
Processing
Frameworks
YARN
Resource Management
HDFS
Distributed File Storage

File System

File System

- A process that manages how and where data on a storage disk is –
  - Stored
  - Accessed
  - And Managed
- General Storage –
  - HDD – Hard disk, Low speed, Low cost
  - SSD – Solid state drives, higher speed, Higher cost
- General File systems
  - FAT – FAT12, FAT16, FAT32, exFAT
  - NTFS
  - HFS, HFS+, HPFS, APFS, UFS
  - Ext2, ext3, ext4

GFS vs HDFS

GFS

- Google File System
- Proprietary by Google
- Not publicly available, Internally used by Google
- 2003

HDFS

- Hadoop Distributed File System
- Open Source
- Publicly available and being used by numerous industries
- Part of Apache Friends
- 2006

Hadoop In Real Life

Hadoop Wins Terabyte Sort Benchmark (July 2008)

- One of Yahoo's Hadoop clusters sorted 1 terabyte of data in **209 seconds**
- The cluster had 910 nodes;
- Each Node:
  - 2 quad core Xeons @ 2.0ghz per node;
  - 4 SATA disks per node;
  - 8GB RAM per a node;
  - 1 gigabit ethernet on each node;
  - Red Hat Enterprise Linux Server Release 5.1 (kernel 2.6.18);
  - Sun Java JDK 1.6.0_05-b13
- 40 nodes per a rack;
- 8 gigabit ethernet uplinks from each rack to the core;

Who Uses Hadoop?

Facebook

- Stores copies of their internal data
  - User profiles
  - User network of friends
  - Photos etc.
- Analysis of user behaviour
- Goals:
  - Improve services
  - Target advertising
  - Detect fraudulent activities

Amazon

- Product Recommendation Engine
- Analysis of
  - Past purchases
  - Browsing History
  - User behaviour
- Goals
  - Improve product suggestions
  - Better sales
  - Customer experience

LinkedIn

- Recommendations
  - People You May Know
  - Jobs You might be interested in
  - Skill endorsement suggestions

Twitter

- Stores and processes
  - Tweets
  - Log files
  - Other data
- Goals
  - Various Analytics Tasks
  - Understanding user behaviour
  - Trending Topics
  - Sentiment Analysis

Uber

- Data Analysis
  - Drivers
  - Rides
  - User behaviour
- Machine Learning
  - Predict ride demand
  - Estimate Fares
  - Detect fraudulent activities

Session 2

Questions from Previous Session

- “Moving Processing Rather than Data”
  - Traditionally,
    - Data is extracted from stored location and moved to a location where processing will happen
    - Works for small amount of data
    - Highly inefficient for massive volume
  - Hadoop, follows – “Moving Processing to the Data”
    - The code which will do the processing is moved to the stored data
    - Efficient for large datasets, Minimal data movement over the network
    - Implemented through MapReduce
      - Map tasks are distributed across the cluster and run on the nodes where the data blocks are
      - The results (which are much smaller than raw data) need to be transferred

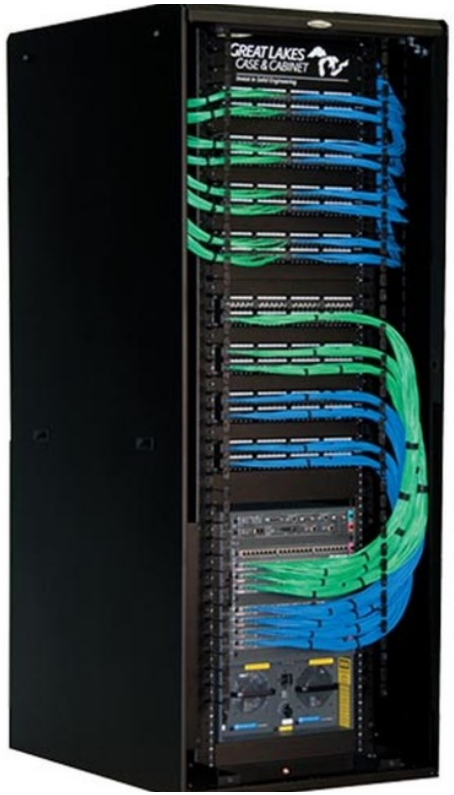
Data Centers

- Facebook Data center on the right.
- Huge Campus of many buildings
- Each building has tons of racks.
- Three main components of infrastructure
  - Compute
  - Storage
  - Network
- Needs
  - Cooling
  - Avoiding high temperature
  - Controlled humidity levels

Facebook Data Center

Hadoop Cluster: Rack

- Rack is a collection of nodes.
- Racks are interconnected over network



Hadoop Nodes

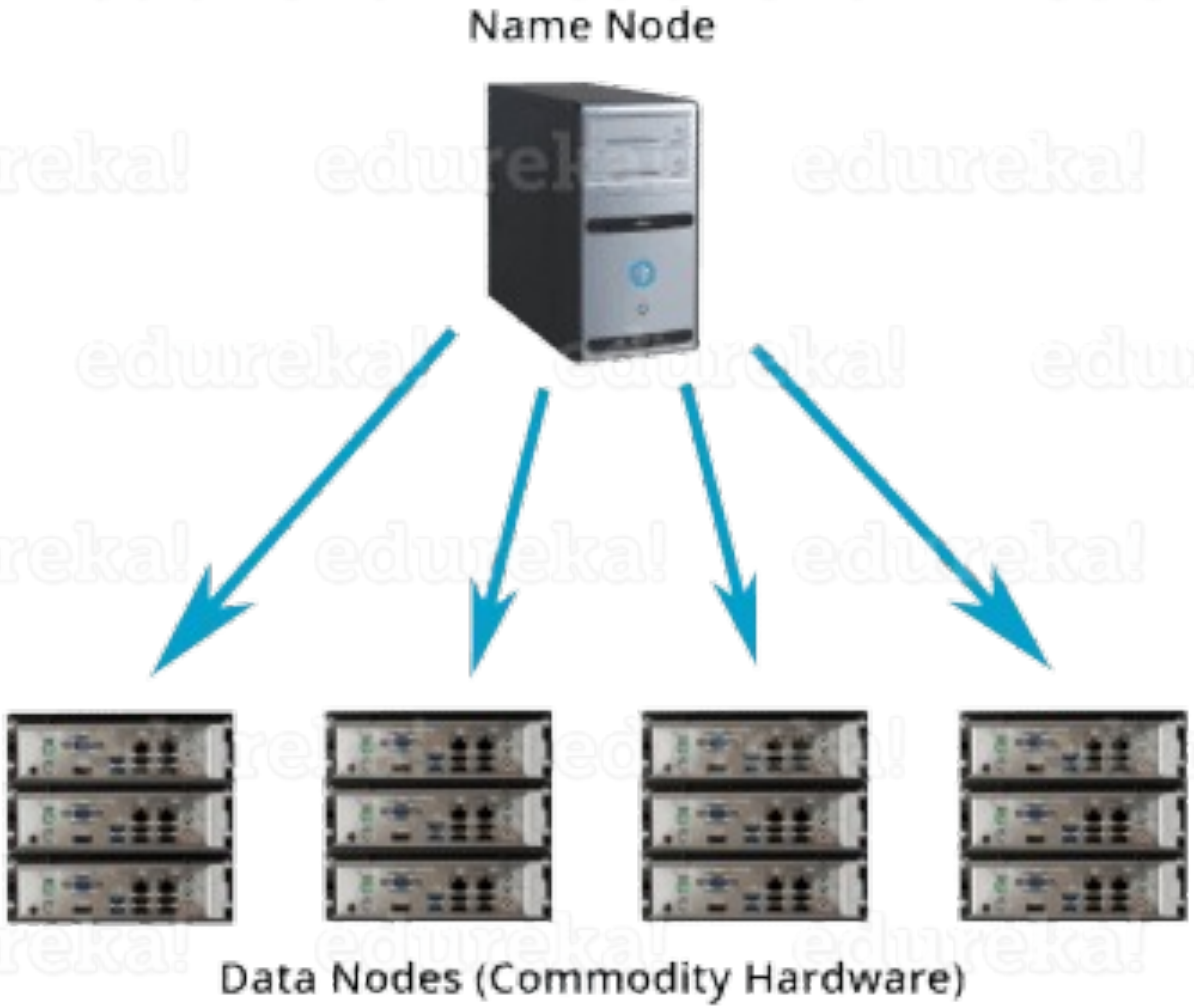
- NameNode
- DataNode

- A node is a machine / a computer
  - CPU
  - RAM
  - Disk
- A node is connected to a network
- All nodes are interconnected

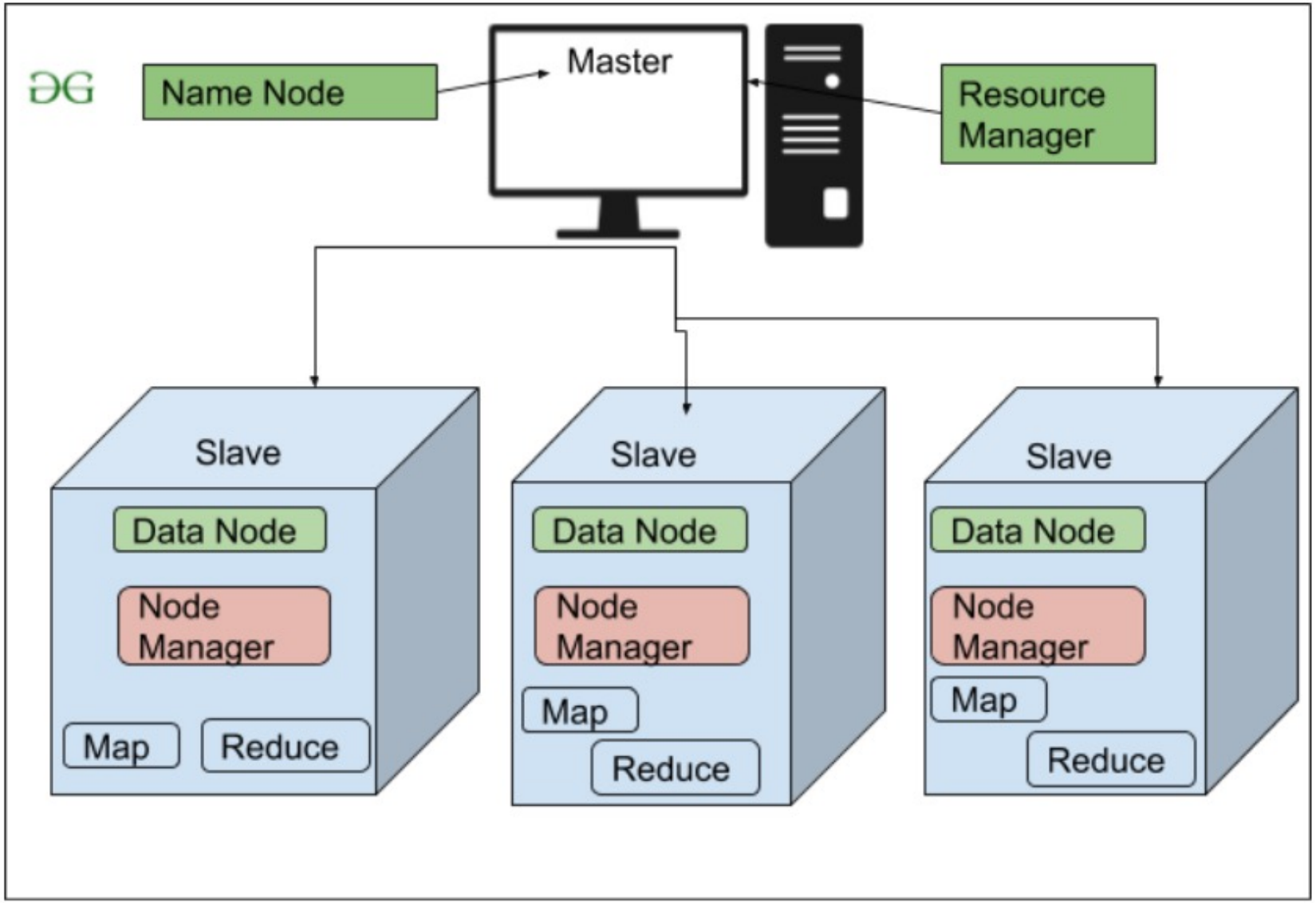
Hadoop Nodes

X	Name Node	Data Node
Processor	2 Quad Core CPUs running @ 2 GHz	2 Quad Core CPUs running @ 2 GHz
RAM	128 GB	64 GB
Disk	6x 1 TB SATA	12-24x 1 TB SATA
Network	10 Gigabit Ethernet	10 Gigabit Ethernet

HDFS architecture

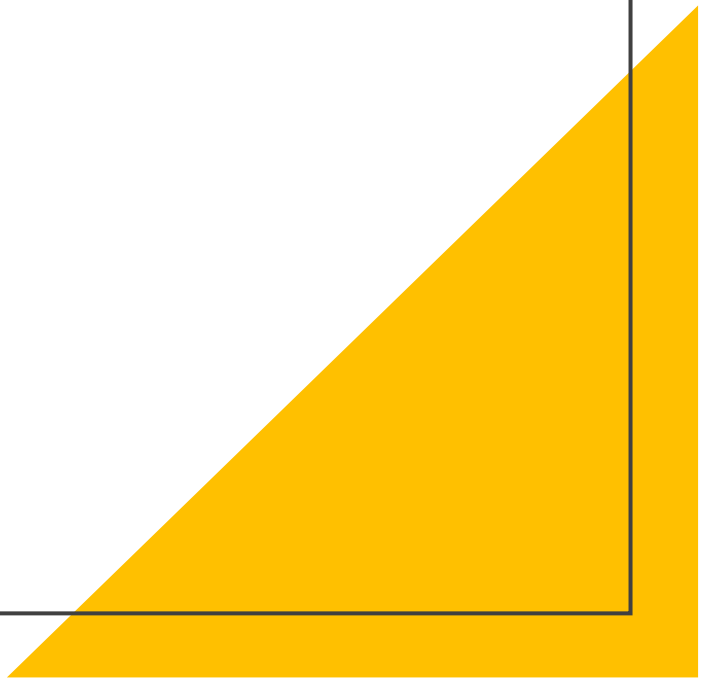


HDFS architecture



HDFS

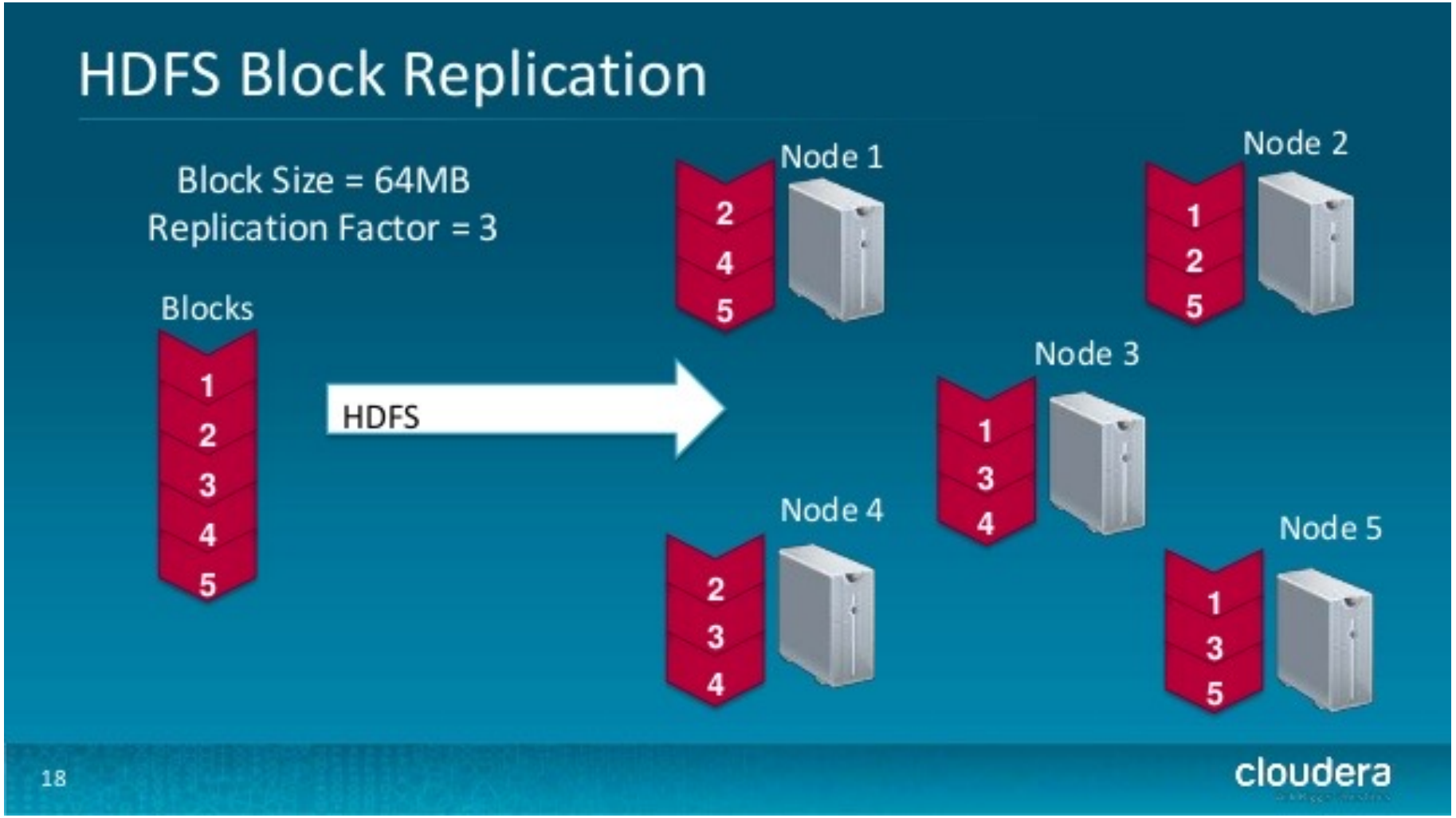
- Distributed file system
- In General, File Systems are embedded within OS Kernel
  - Runs as OS Process
- HDFS is a user-space file system
  - Works as a user process
  - Within the process space allocated to user process
- Creates multiple replicas of data blocks
  - Better Reliability
  - Rapid Access



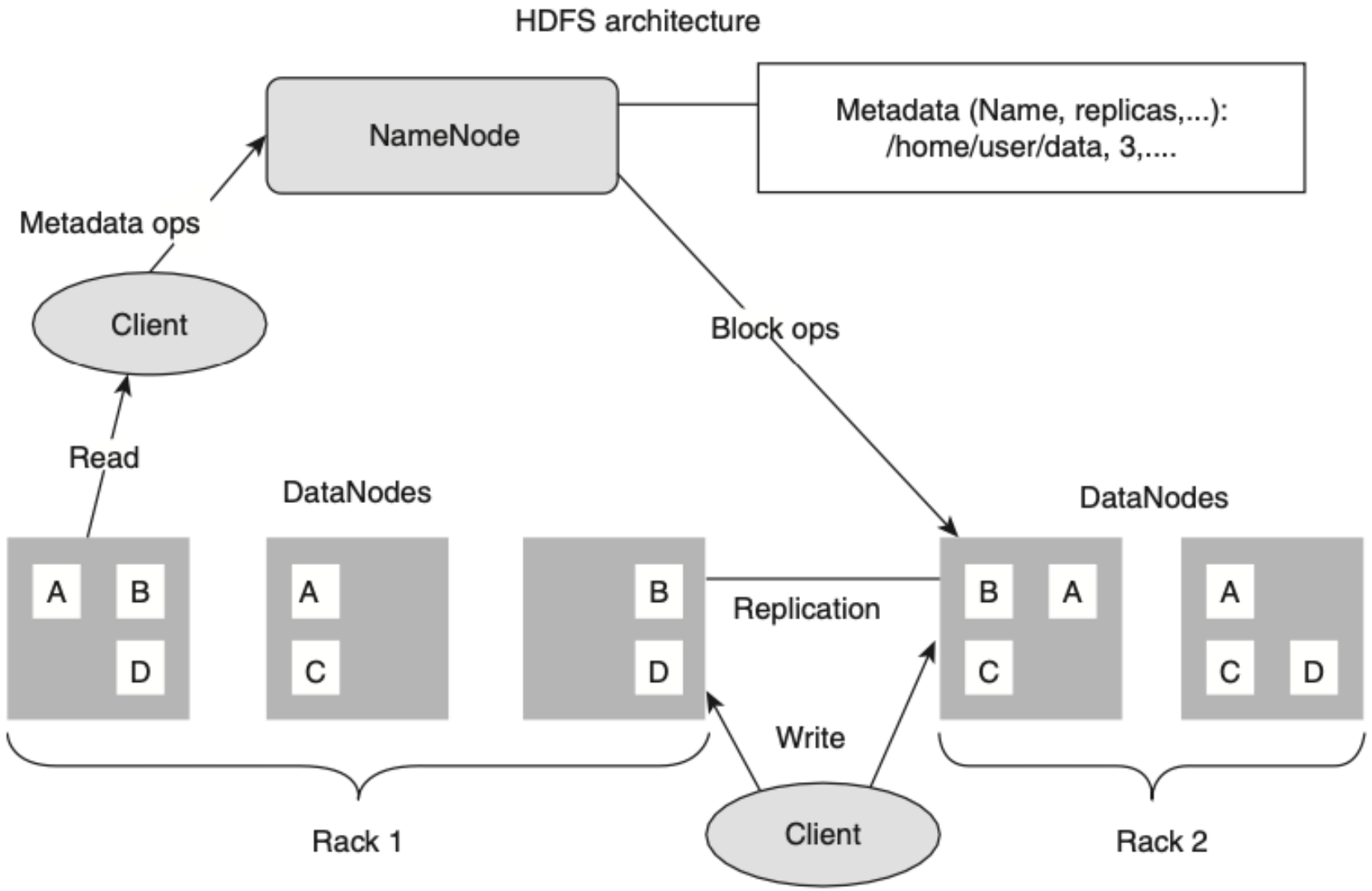
HDFS contd..

- Block size – 64MB, 128 MB
- Traditional file system block size is – 4-8 KB
- These data blocks are placed on the computer nodes
- Nodes which are in cluster => Cluster Nodes – aka – Data nodes
- NameNode
  - Responsible for storage and management of metadata
  - Informs MapReduce where the data is

HDFS Replication



NameNode and DataNode



NameNode

- Master node, Contains metadata
- Maintains the directories and files and blocks on Data Nodes
- Functions
  - Manages namespace of the file system in memory
  - Maintains inode information
  - Maps inode to the list of blocks and locations
  - Authorization and authentication
  - Creates checkpoints and logs the namespace changes
  - Monitors the health of DataNodes
  - In case there is a missing block, replicates it.

Meta Data at NameNode

- Metadata has many information stored

- Namespace Information
  - Details about directories and files, names, permissions, ownership, last modified time
- File to Block Mapping
- Block to DataNode Mapping
- File System Properties
  - Block size, replication factor
- Transaction Logs
- Checkpoints

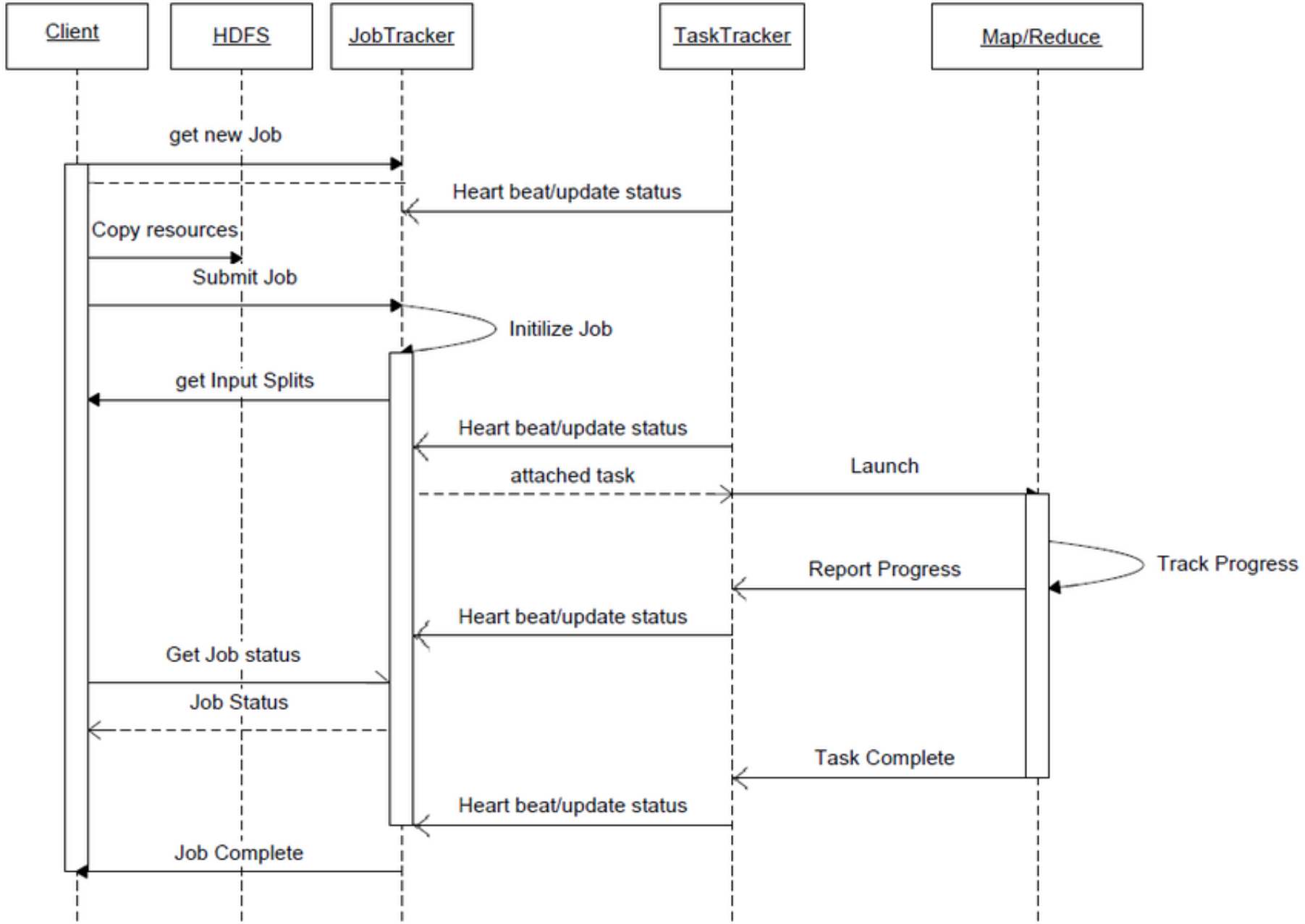
DataNode

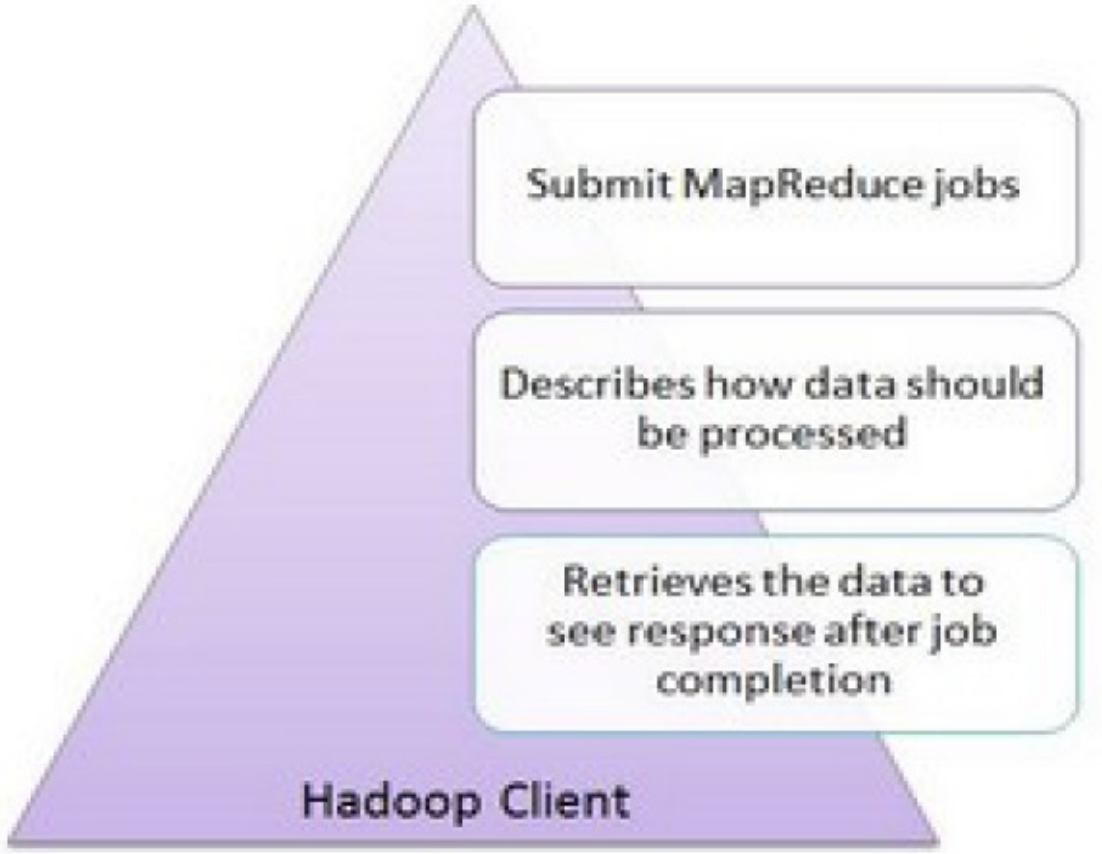
- Slave Node
- Provide the actual storage
- Deployed on each machine
- Functions
  - Responsible to process read and write requests for the clients
  - Handles block storage on multiple volumes while maintaining integrity
  - Periodically sends info to NameNode
    - Heartbeats
    - Block reports

What is Heartbeat?

Heartbeat is a signal sent from DN to NN to indicate that it is operational and functioning correctly

Sequence Diagram

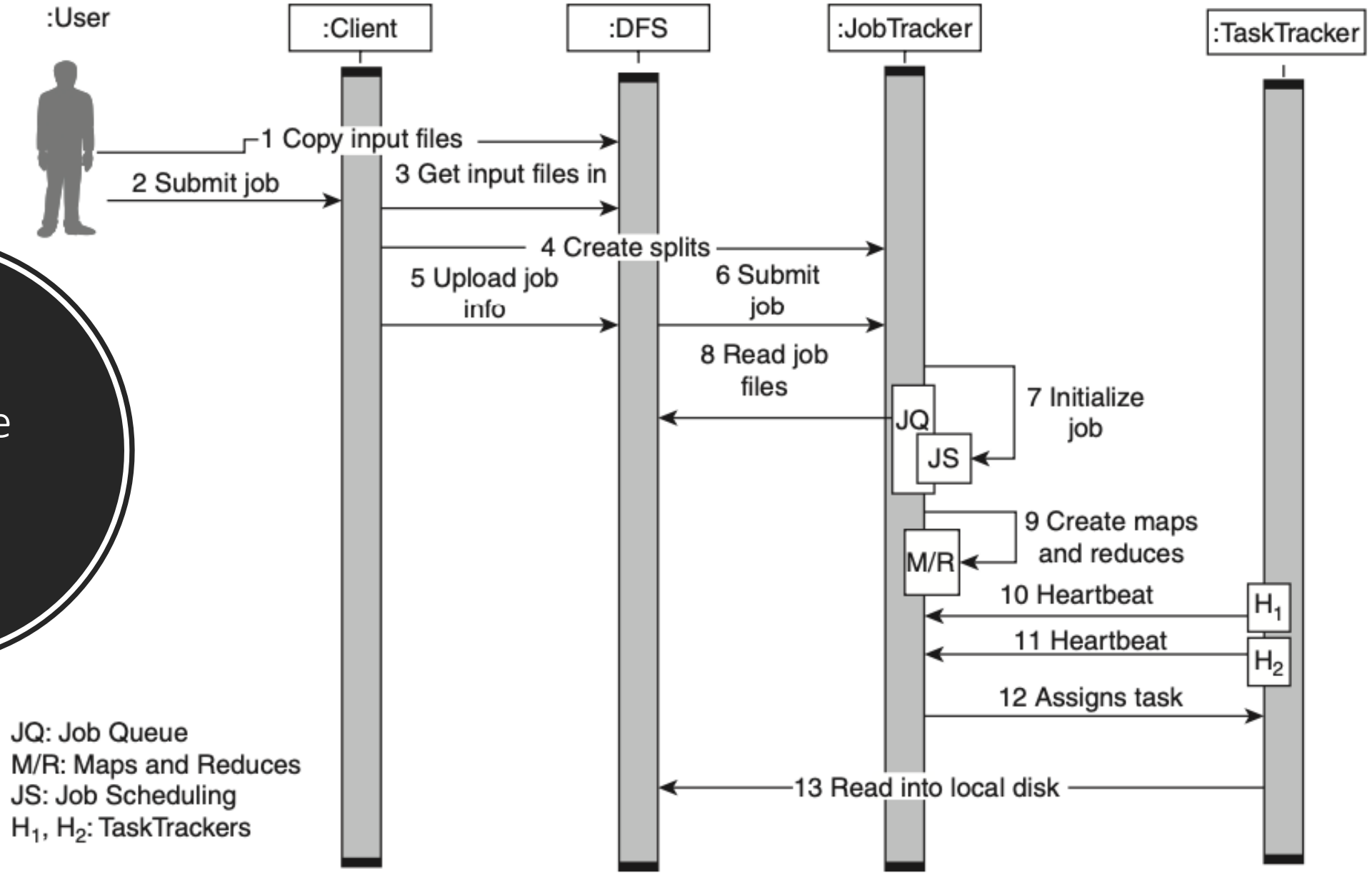




Hadoop Client

- It is neither master nor slave.
- Used for:
  - loading the data into cluster,
  - submit MapReduce jobs describing how the data should be processed
  - and then retrieves the data to see the response after job completion.

Sequence Diagram



Secondary NameNode

- This is not a backup NameNode
- Performs housekeeping functions for primary NameNode
- Checkpointing
  - Process of -
  - Periodically merging the namespace image with the edit log
  - So that the edit log does not become very large
- Relieving the NameNode
  - Offloads some work of the primary NameNode
- Recovery Aid
  - Does not become primary but the checkpointing function aids in reduced recovery time

HDFS Contd..

- Implemented on commodity hardware
- Used for Large files – GB, TB and beyond
- Automatic recovery of nodes
  - If a node fails, File system auto recovers
- Suitable for
  - Data written once, accessed many times
  - E.g log files – audit logs, server logs, stream of data
- Not suitable for
  - Small data files
  - Reading content from any random position
    - Best for reading from beginning or end of the file

Hadoop Assumptions

- Hardware will fail
- Processing will be done in batches
- Apps that run HDFS have large datasets, GBs to TBs or more..
- Portability is important
- Availability of high bandwidth scaled to hundred of node in a cluster
- Support tens of millions of file in a single instance
- Application that needs write once- read many access model.

Map Reduce

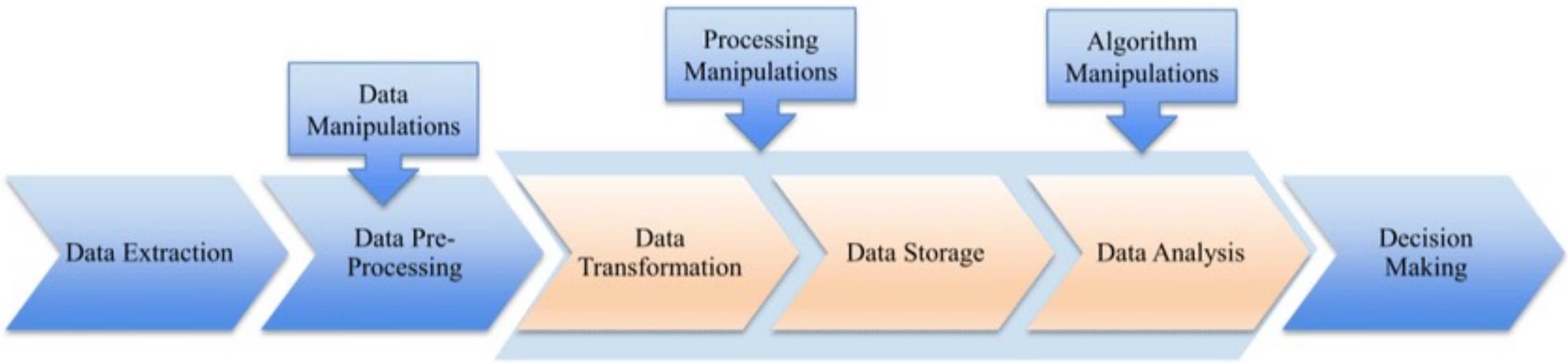
- Job Trackers
  - Master which manages the jobs and resources in the cluster
  - Schedules the Map task on TaskTracker of the same machine
- Task Trackers
  - Slaves deployed on each machine
  - Responsible for running the Map and Reduce tasks
- Job History Server
  - A Daemon
  - Saves historical info about the tasks completed

YARN

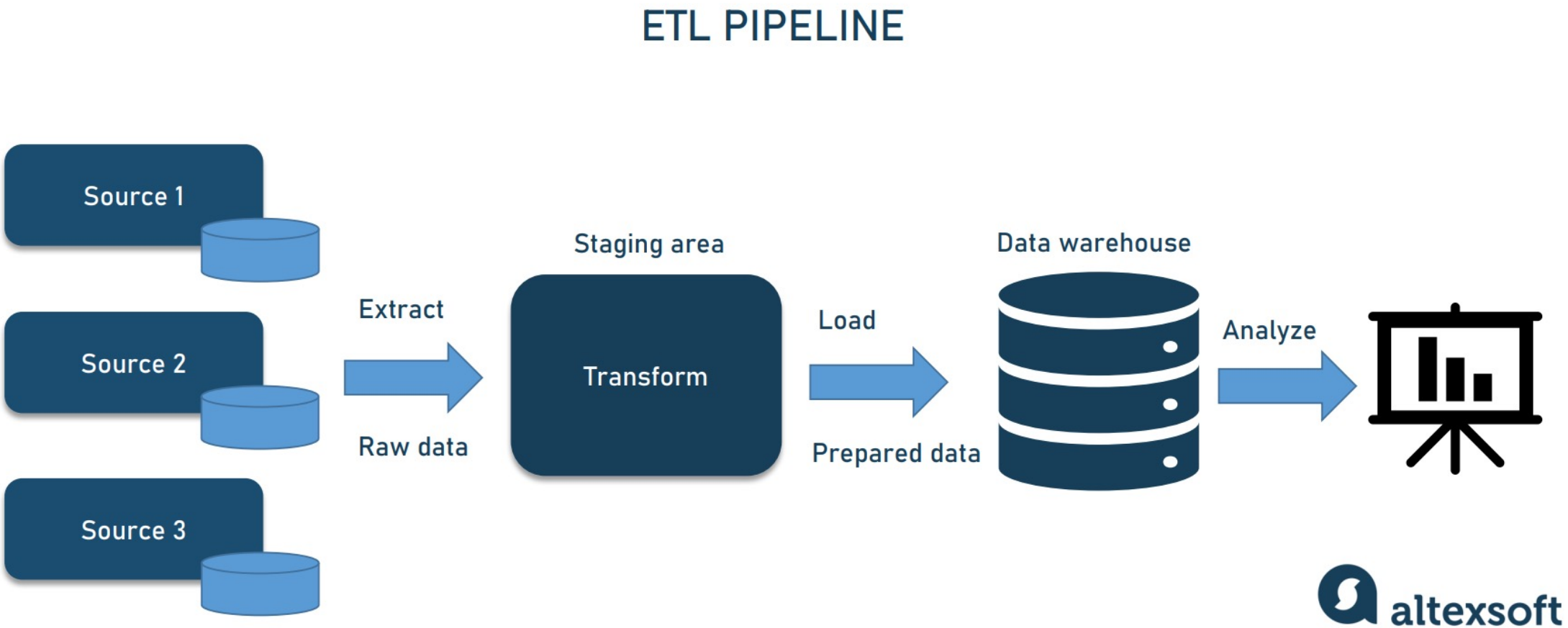
- Splits up
  - Resource Management
  - Job Scheduling and monitoring
- Into separate daemons
  - Global resource manager (RM)
  - ApplicationMaster (AM)
- Single node does not have to handle both -
  - scheduling and resource management
- Responsibilities are distributed across cluster

Hadoop Ecosystem

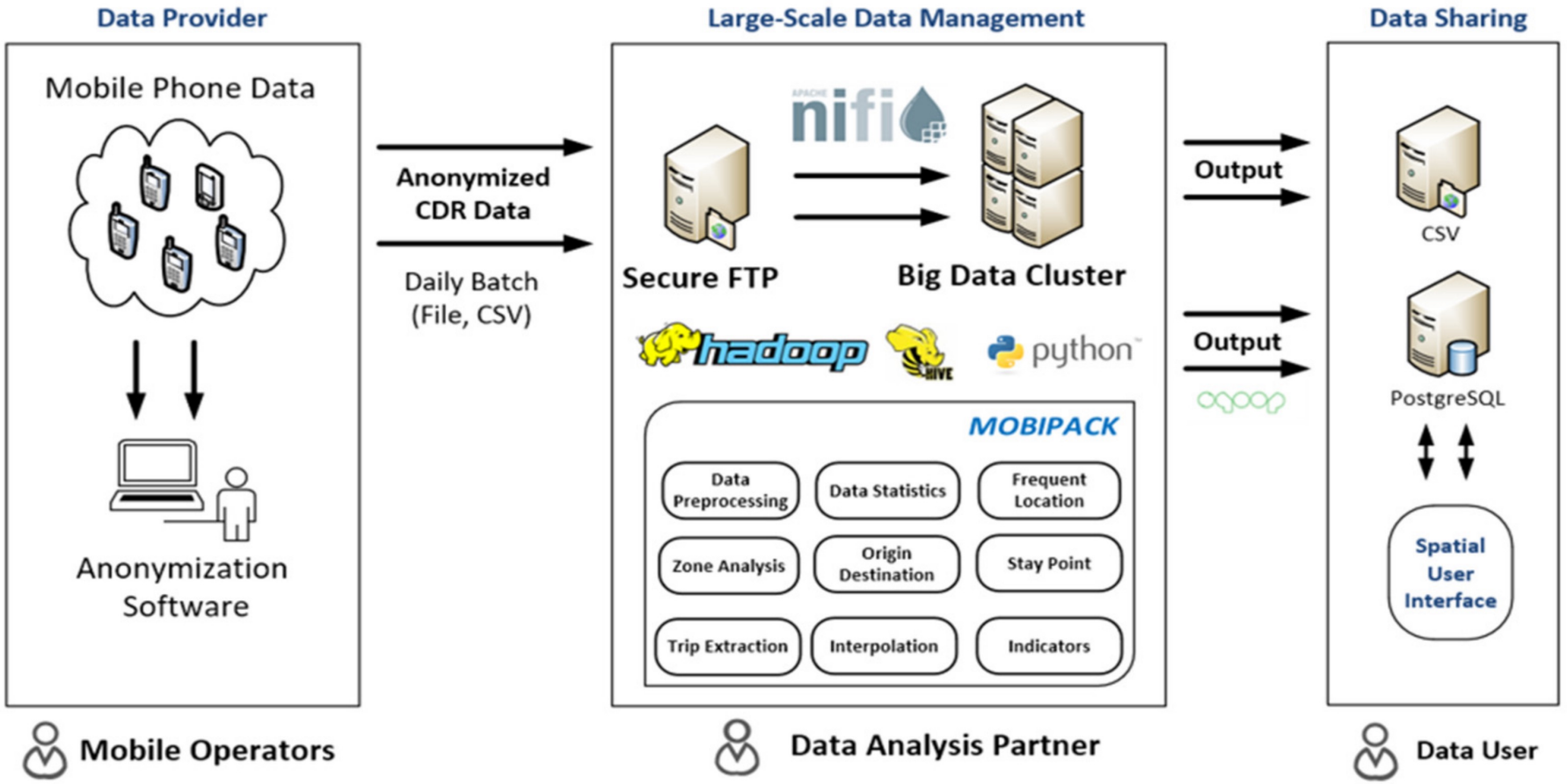
Let’s discuss Pipeline



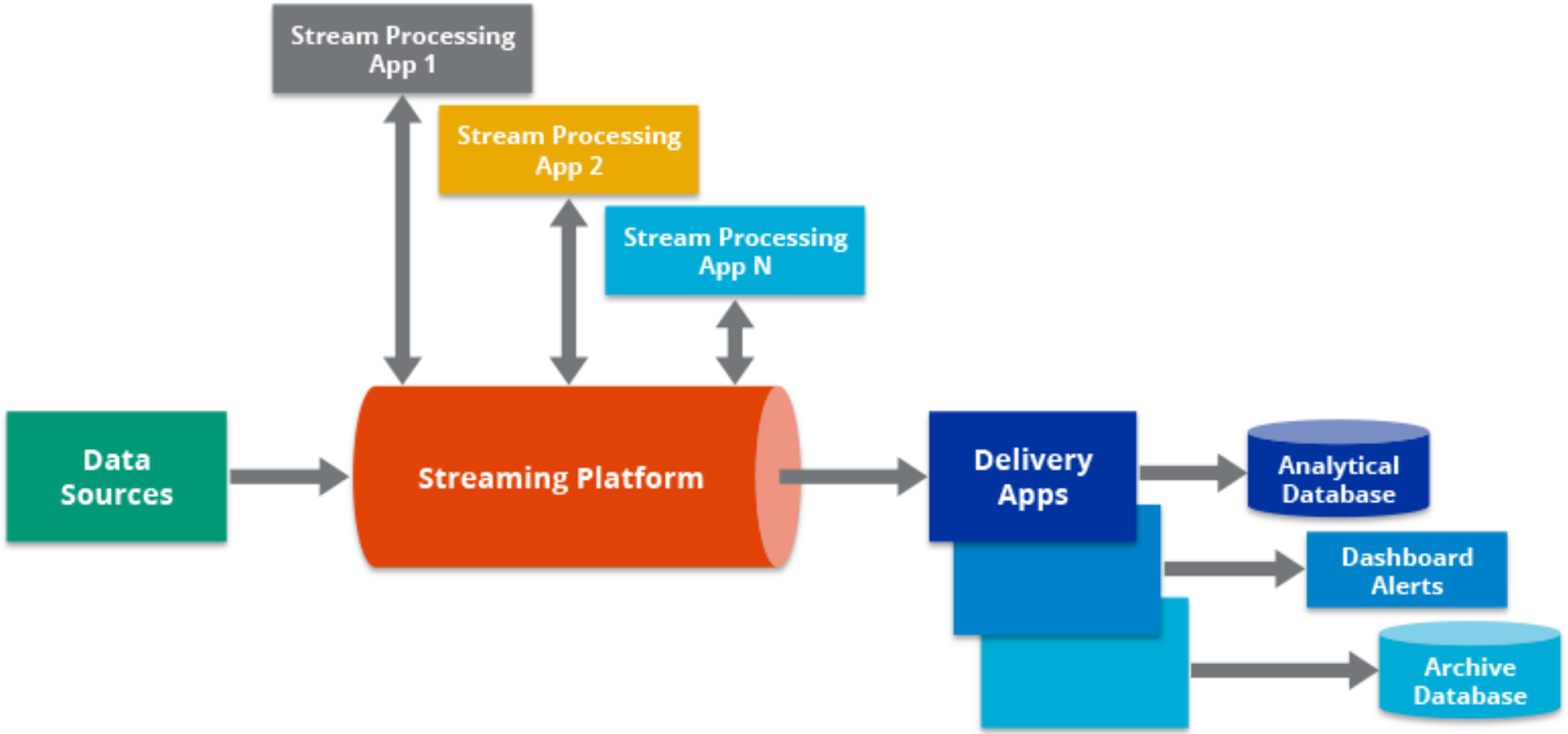
Let’s discuss Pipeline



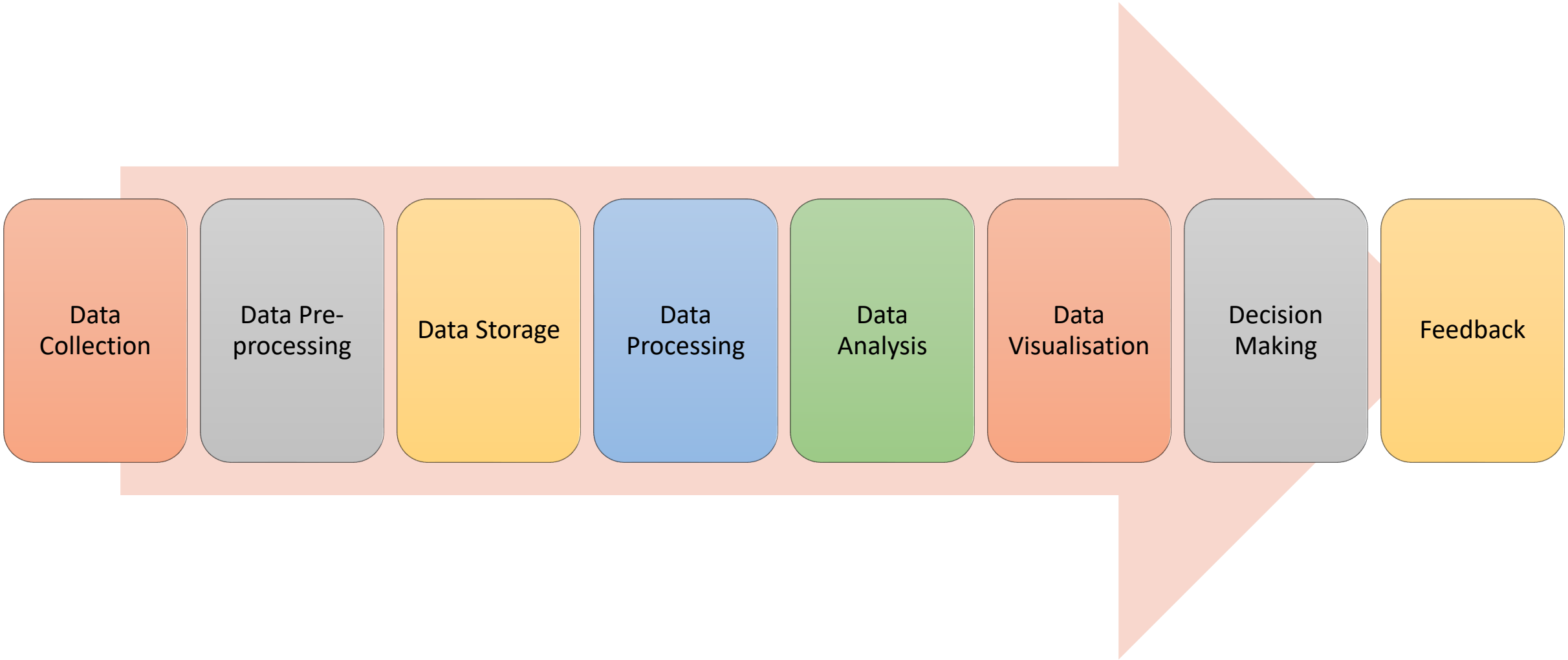
Let’s discuss Pipeline



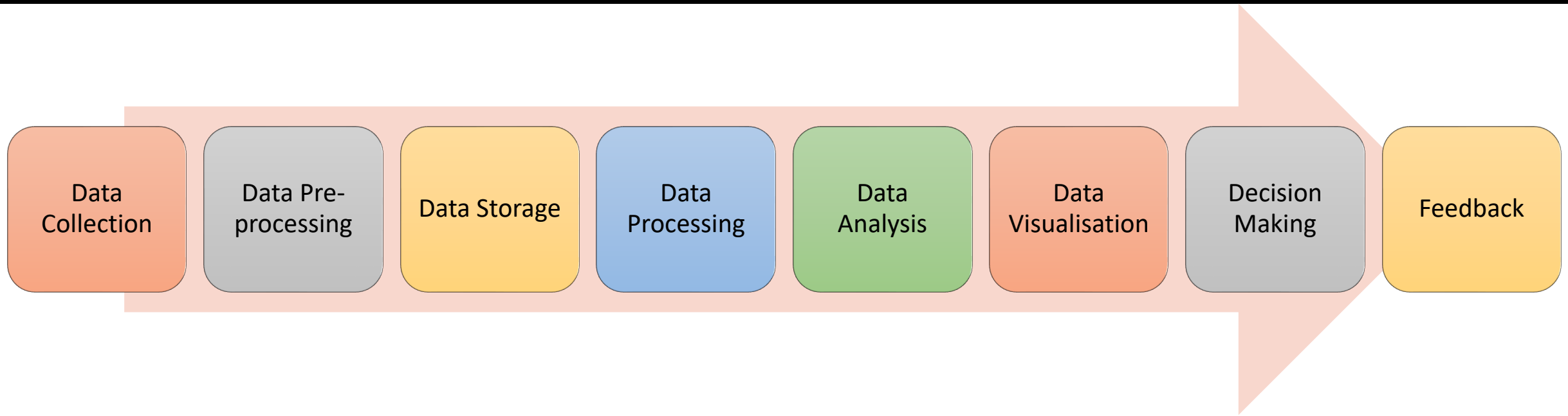
Let’s discuss Pipeline



Let’s discuss Pipeline

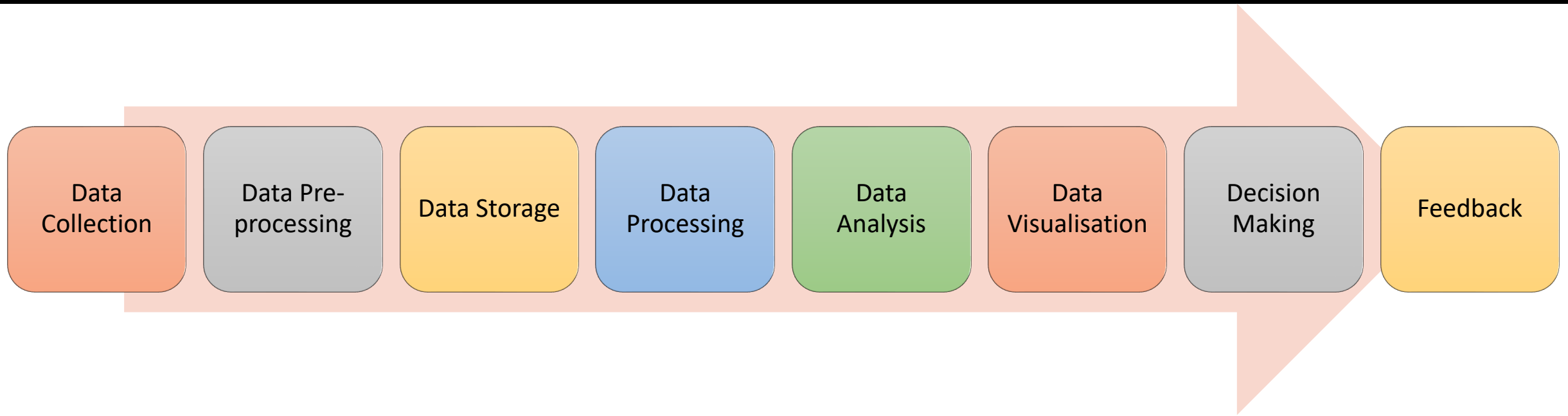


Hadoop Ecosystem – Data Ingestion



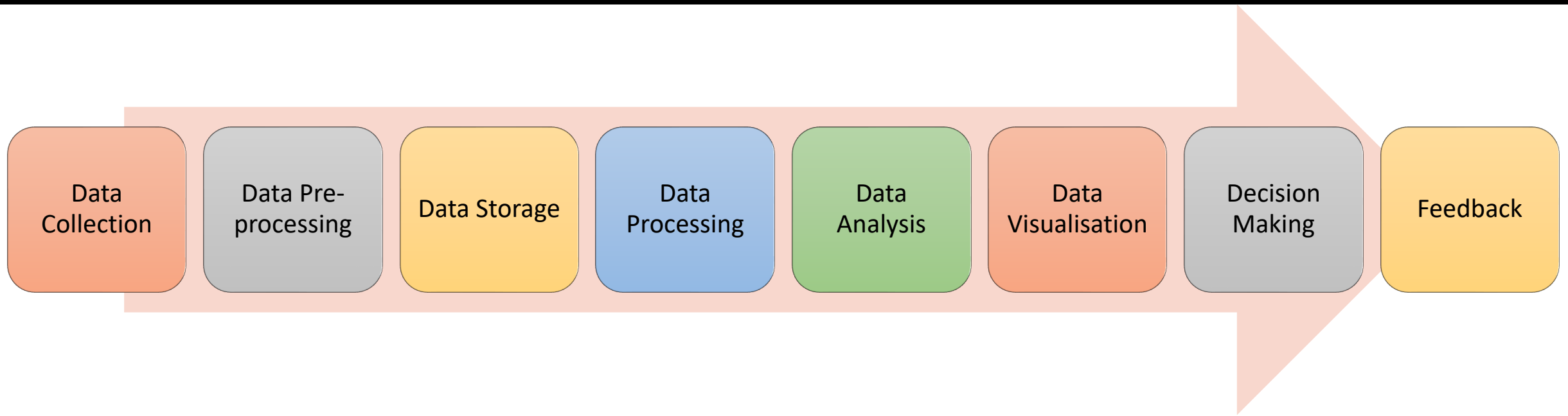
Data Ingestion	Data Pre-processing	Data Storage	Data Processing	Data Analysis	Data Visualisation	Decision Making	Feedback
• Apache Flume • SQOOP							

Hadoop Ecosystem – Data Pre-processing



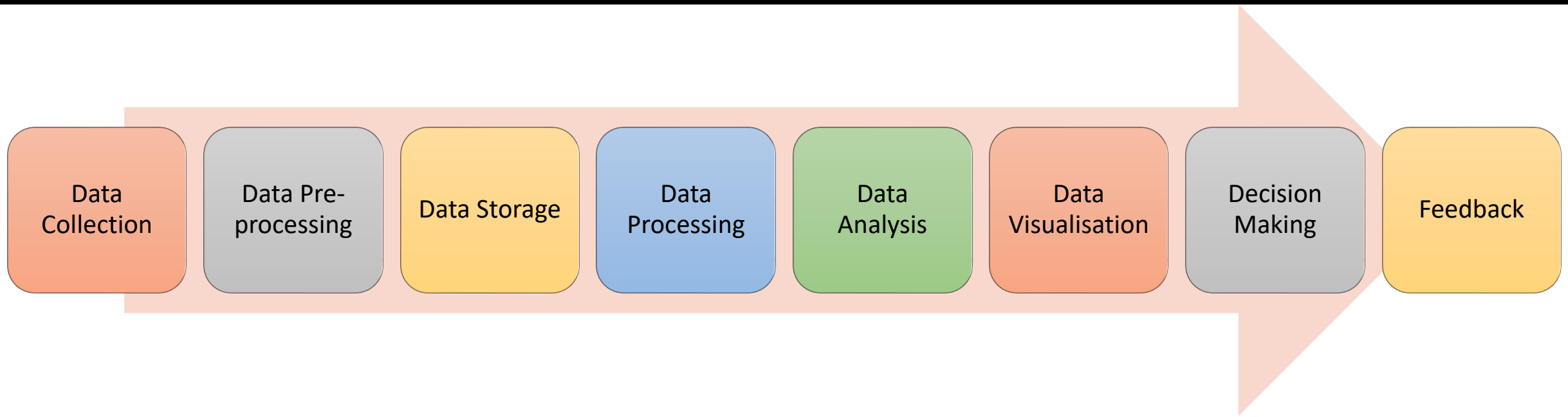
Data Ingestion	Data Pre-processing	Data Storage	Data Processing	Data Analysis	Data Visualisation	Decision Making	Feedback
• Apache Flume • SQOOP	• Hive • Pig						

Hadoop Ecosystem – Data Storage



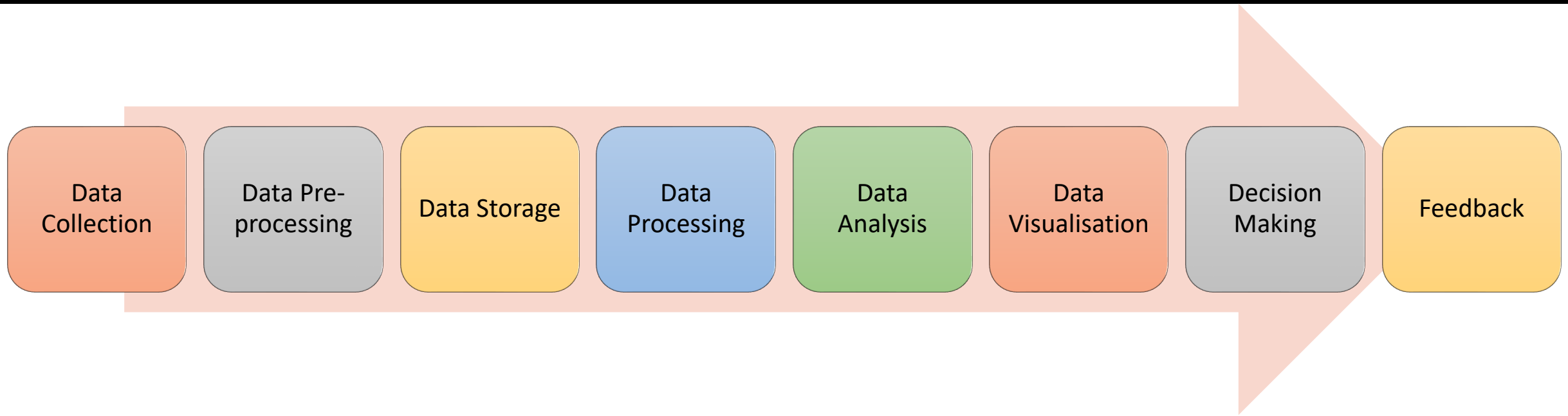
Data Ingestion	Data Pre-processing	Data Storage	Data Processing	Data Analysis	Data Visualisation	Decision Making	Feedback
• Apache Flume • SQOOP	• Hive • Pig	• HDFS					

Hadoop Ecosystem – Data Processing



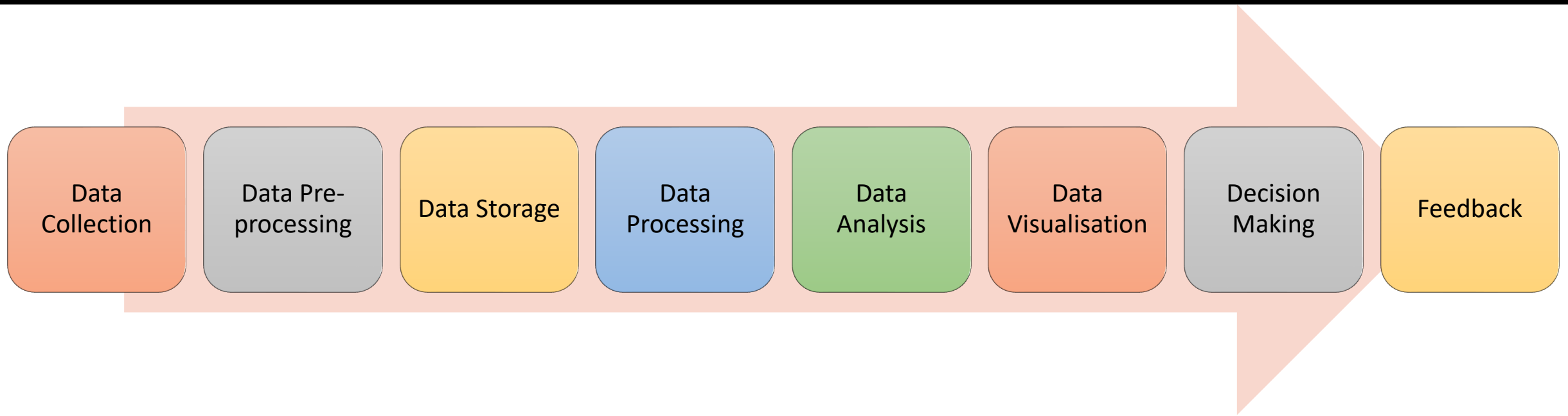
Data Ingestion	Data Pre-processing	Data Storage	Data Processing	Data Analysis	Data Visualisation	Decision Making	Feedback
• Apache Flume • SQOOP	• Hive • Pig	• HDFS	• MapReduce • YARN				

Hadoop Ecosystem – Data Analysis



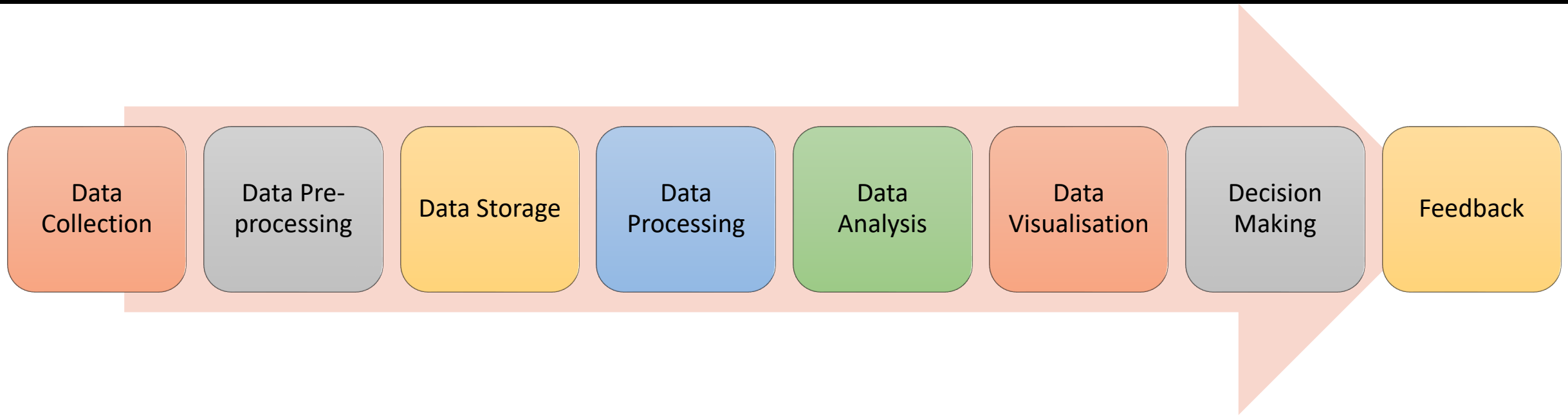
Data Ingestion	Data Pre-processing	Data Storage	Data Processing	Data Analysis	Data Visualisation	Decision Making	Feedback
• Apache Flume • SQOOP	• Hive • Pig	• HDFS	• MapReduce • YARN	• Hive, Pig • Mahout			

Hadoop Ecosystem – Data Visualization



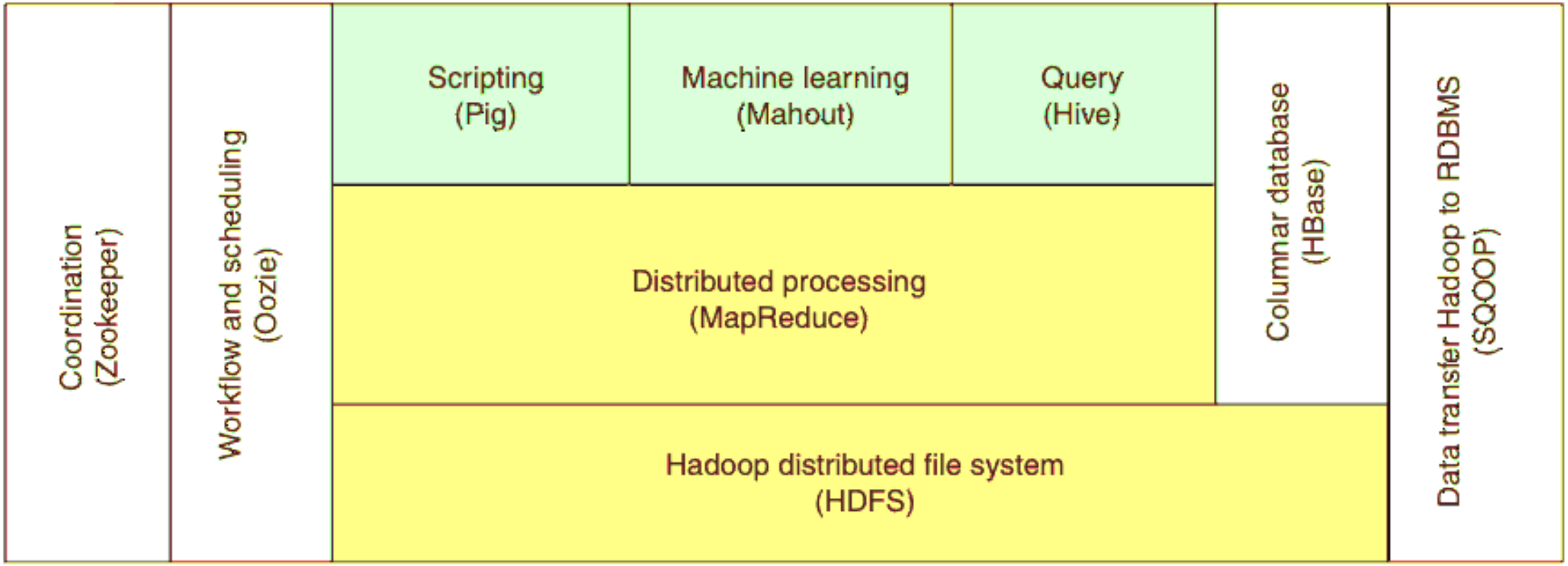
Data Ingestion	Data Pre-processing	Data Storage	Data Processing	Data Analysis	Data Visualisation	Decision Making	Feedback
• Apache Flume • SQOOP	• Hive • Pig	• HDFS	• MapReduce • YARN	• Hive, Pig • Mahout	• External tools • Tableau • Power BI		

Hadoop Ecosystem



Data Ingestion	Data Pre-processing	Data Storage	Data Processing	Data Analysis	Data Visualisation	Decision Making	Feedback
• Apache Flume • SQOOP	• Hive • Pig	• HDFS	• MapReduce • YARN	• Hive, Pig • Mahout	• External tools • Tableau • Power BI	• "You"	

Hadoop Ecosystem



Key Components of Hadoop Ecosystem

- Apache Hbase - Non-Relational Database
- Apache Hive - Data Access and Query
- Apache Hcatalog - Metadata Service
- Apache Pig - Scripting platform
- Apache Mahout - Machine learning libraries for Data Mining
- Apache Oozie - Workflow and scheduling services
- Apache ZooKeeper - Cluster coordination
- Apache Sqoop - Data Integration Services

HBase

- **Column-Oriented Database:** HBase is a type of NoSQL database that stores data in a column-oriented format, which is efficient for read and write operations on big datasets.
- **Scalability:** It provides linear and modular scalability, allowing you to add more nodes as your data grows.
- **Real-Time Access:** HBase supports real-time read/write access to your Big Data. It's designed to host very large tables with billions of rows and millions of columns.
- **Automatic Sharding:** HBase automatically partitions data and distributes it across the cluster, eliminating the need for manual sharding.
- **Fault-Tolerant:** Provides automatic recovery support and replicates data across different nodes to ensure high availability and fault tolerance.
- **Consistent Read/Write:** Provides strong consistency guarantees for reads and writes, which is a unique feature among NoSQL databases.
- Does not provide own query or scripting language – can be accessible through Java Thrift and REST APIs.

Hive

- Data warehouse software
- Responsible for reading-writing and managing large datasets
- SQL Like interface
- Supports various data formats such as
  - Text Files, CSV, Parquet, JSON, and JDBC storage handlers.
  - And more..

HCatelog

- **Table Management Layer:** HCatalog is a table and storage management layer for Hadoop that allows users to share and access data in a tabular format across Hadoop tools.
- **Data Abstraction:** It abstracts the user from the complexities of where and in what format data is stored.
- **Interoperability:** Allows MapReduce, Hive, and Pig jobs to read and write data in the same format, improving interoperability.
- **Schema and Location Transparency:** Provides a centralized location for storing data schema information, allowing for schema and location transparency.

Pig

- **Scripting Language:** Pig uses a scripting language called Pig Latin, which is specifically designed for expressing data transformations in parallel across large data sets.
- **High Level Abstraction:** Provides a high level of abstraction for analyzing large datasets, making it easier to write and read data processing programs.
- **Extensible:** Allows developers to create custom functions to meet their specific data processing requirements.

Sqoop

- **Data Transfer:** Sqoop is a command-line interface application for transferring data between relational databases and Hadoop.
- **Bi-Directional Transfer:** It supports bi-directional data transfer, allowing data to be moved from a relational database to Hadoop and vice versa.
- **Integration with RDBMS:** Provides connectivity with relational databases such as MySQL, Oracle, PostgreSQL, etc.
- **Efficiency:** Uses MapReduce to efficiently transfer bulk data.

Oozie

- **Workflow Scheduler:** Oozie is a workflow scheduler system that manages Hadoop jobs.
- **DAG:** It allows users to create Directed Acyclic Graphs (DAGs) of workflows, which can be run in parallel and sequentially in Hadoop.
- **Integration:** Supports various Hadoop jobs such as MapReduce, Pig, Hive, and Sqoop, as well as system-specific jobs like Java programs and shell scripts.

Mahout

- **Machine Learning:** Mahout is a machine learning library for scalable data analytics.
- **Wide Variety of Algorithms:** It includes various implementations of machine learning algorithms, such as clustering, classification, and collaborative filtering.
- **Scalability:** It's built on top of Hadoop, allowing it to leverage Hadoop's MapReduce and HDFS for distributed computation.

ZooKeeper

- **Distributed Coordination:** Zookeeper provides a reliable distributed coordination service which is used for maintaining configuration information, naming, providing distributed synchronization, and providing group services.
- **Reliability:** It is fast with high reliability. Data stored in Zookeeper is replicated over all the nodes in the ecosystem, making it highly reliable.
- **Simple API:** Zookeeper provides a simple API that allows developers to implement common coordination tasks, such as electing a master server, managing group membership, and managing metadata.

Hadoop Limitations

- Security Concerns
  - Built in java
- Vulnerable by Nature
- Not fit for small data
- Potentially Stability Issues
- General Limitations