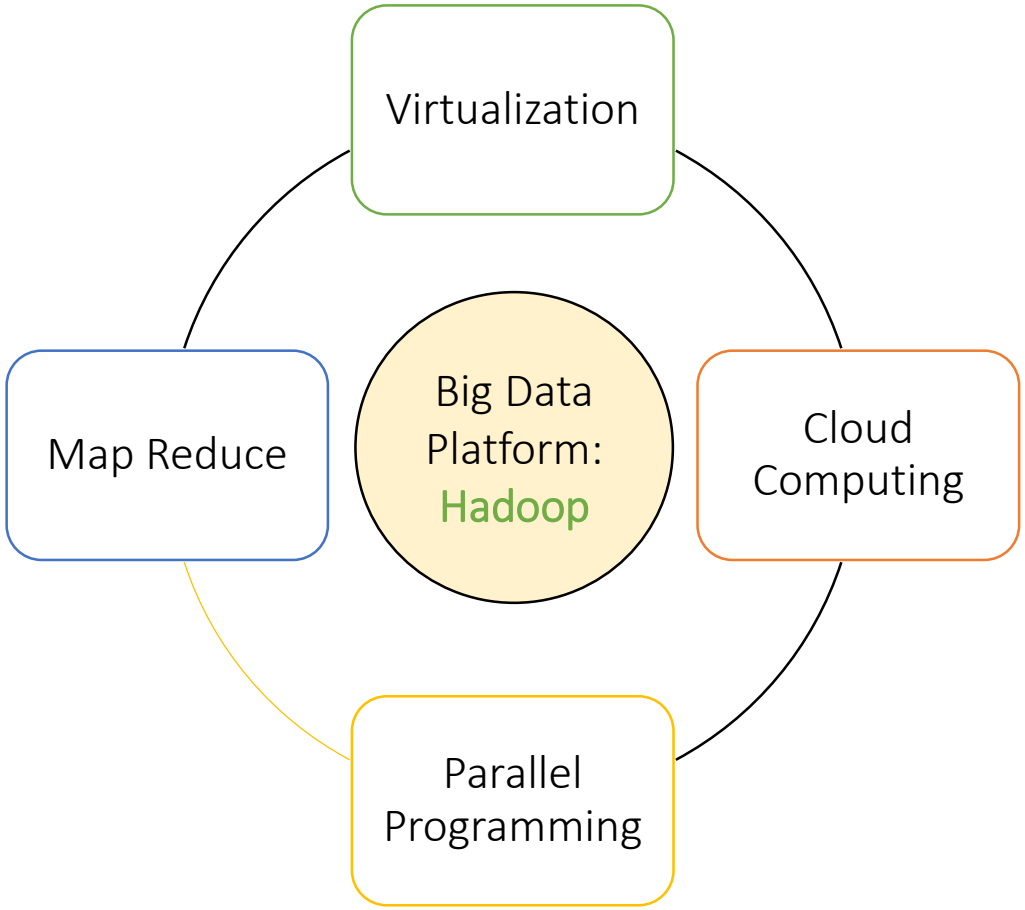
Hadoop Big Data Platform

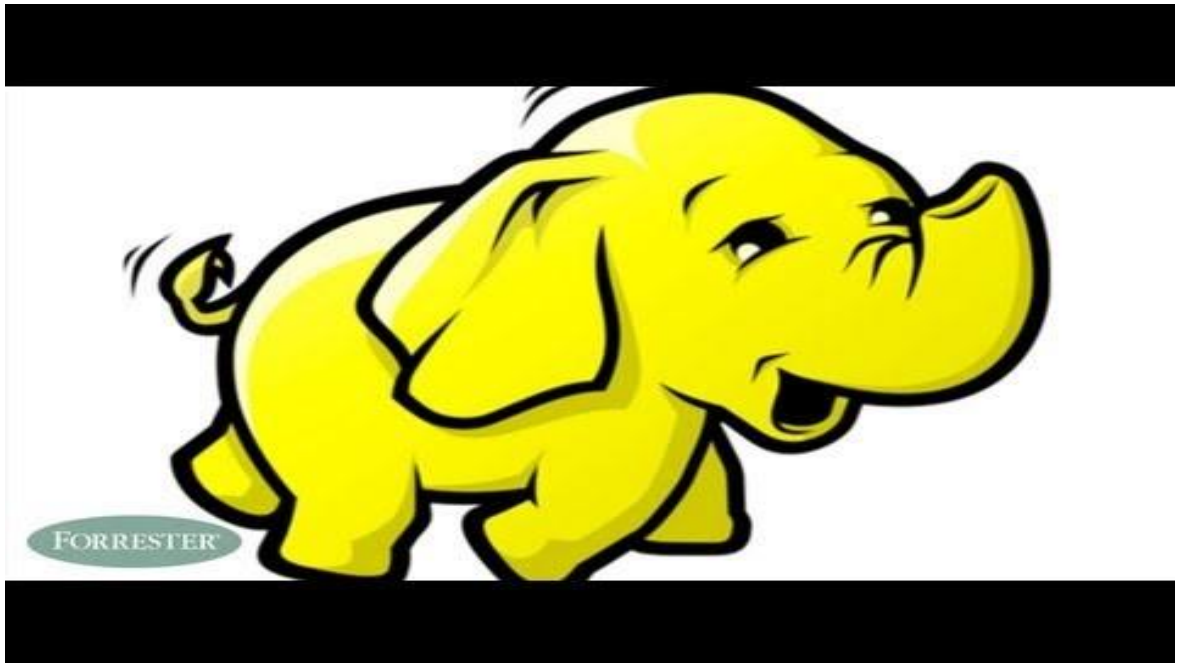
Edgard Luque

Big Data Platform: Hadoop



Hadoop 1.0

What is Hadoop?



Source: https://www.youtube.com/watch?v=4DgTLaFNQq0

So what is Hadoop?

- “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.”

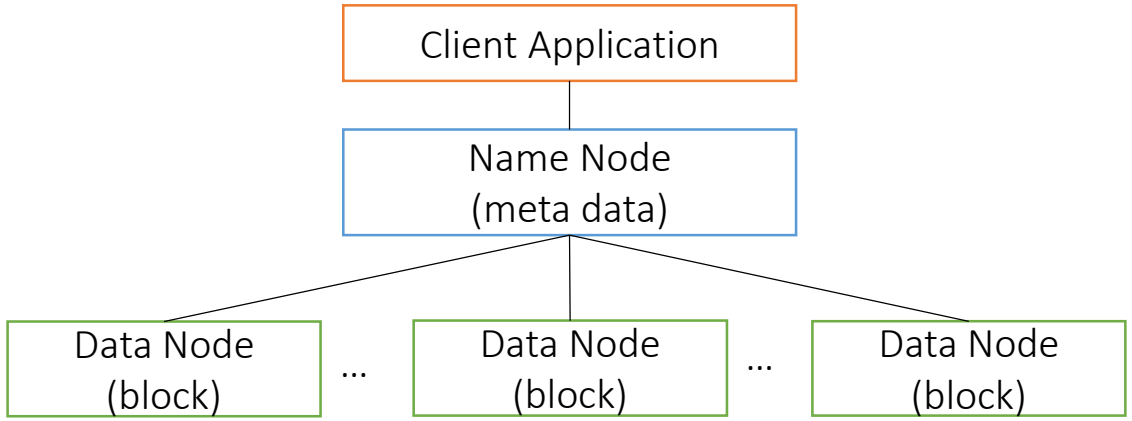
Source: https://hadoop.apache.org/

We already learned about MapReduce…
But what is HDFS?

First Recall what a File System does…

- Controls how data is managed, stored and retrieved.
- Without a file system, we would just have a large blob of data with no way to identify different connected pieces of information.
- File systems are organized around groups of data called files, and groups of files called directories or folders.
- Distributed files systems are files systems that are spread across multiple servers.

HDFS 1 (Hadoop Distributed File System)

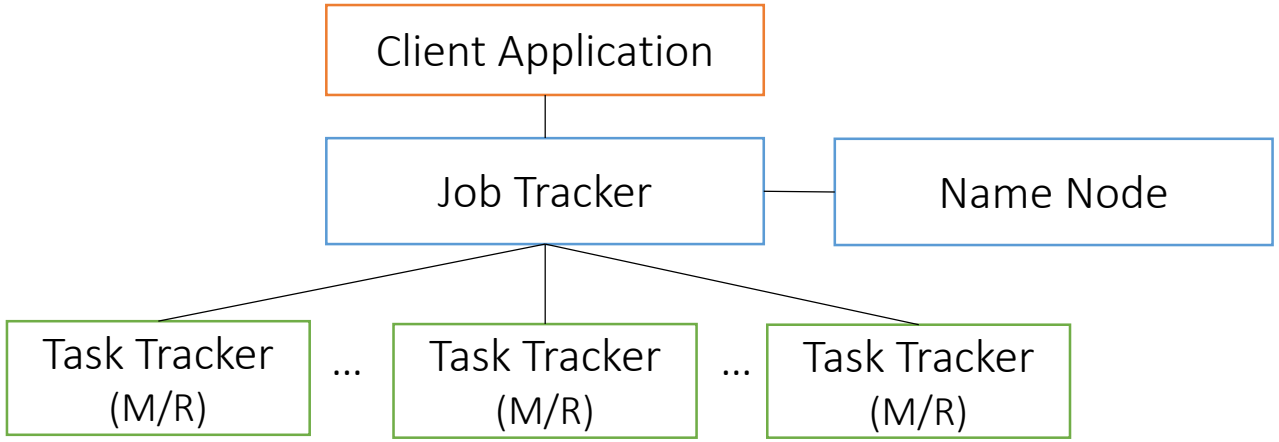


- Java-based distributed file system
- Provides scalable and reliable data storage, designed to span large clusters of commodity servers
- An HDFS cluster is comprised of two kinds of nodes:
  - DataNodes store the data
    - The file content is split into large blocks (typically 128 megabytes)
    - Each block of the file is independently replicated at multiple DataNodes
  - NameNode manages the cluster metadata
    - Maintains namespace tree and mapping of blocks to DataNodes
    - Maintains attributes like permissions, modification, access times, disk space quotas
    - Monitors number of replicas of a block, and if a replica is lost due to node failure, creates another
    - Client applications talk to the NameNode whenever they wish to locate or manipulate a file
- The NameNode is a Single Point of Failure. When the NameNode goes down, the file system goes offline.

Reference: http://hortonworks.com/hadoop/hdfs/

We already learned about the MapReduce Programming Model…
How does it work in the Hadoop Context?

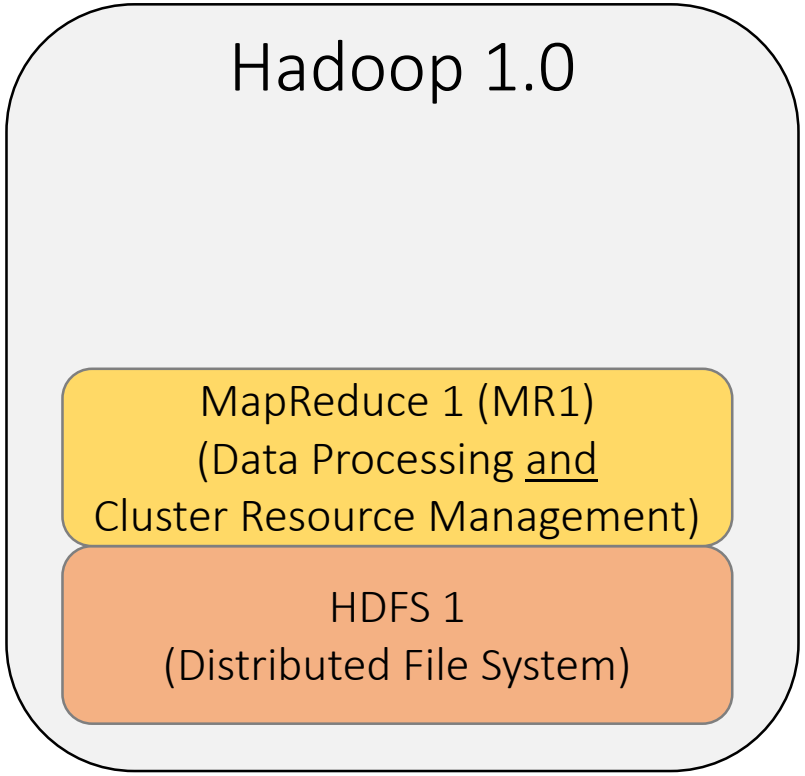
MapReduce 1 (MR1)



- The JobTracker is the service within Hadoop that farms out MapReduce tasks to specific nodes in the cluster, ideally the nodes that have the data, or at least are in the same rack.
    - Client applications submit jobs to the JobTracker.
    - The JobTracker talks to the NameNode to determine the location of the data. The JobTracker locates TaskTracker nodes with available slots at or near the data. The JobTracker submits the tasks (Map, Reduce or Shuffle) to the chosen TaskTracker nodes.
    - If the JobTracker node fails to register a TaskTracker node heartbeat, or if the TaskTracker node notifies the JobTracker of a Task failure, the JobTracker resubmits the job.
    - When the work is completed, the JobTracker updates its status. Client applications can poll the JobTracker for information.
- The JobTracker is a point of failure for the Hadoop MapReduce service. If it goes down, all running jobs are halted.

Source: http://wiki.apache.org/hadoop/JobTracker

Hadoop 1.0 <HDFS1 + MR1>

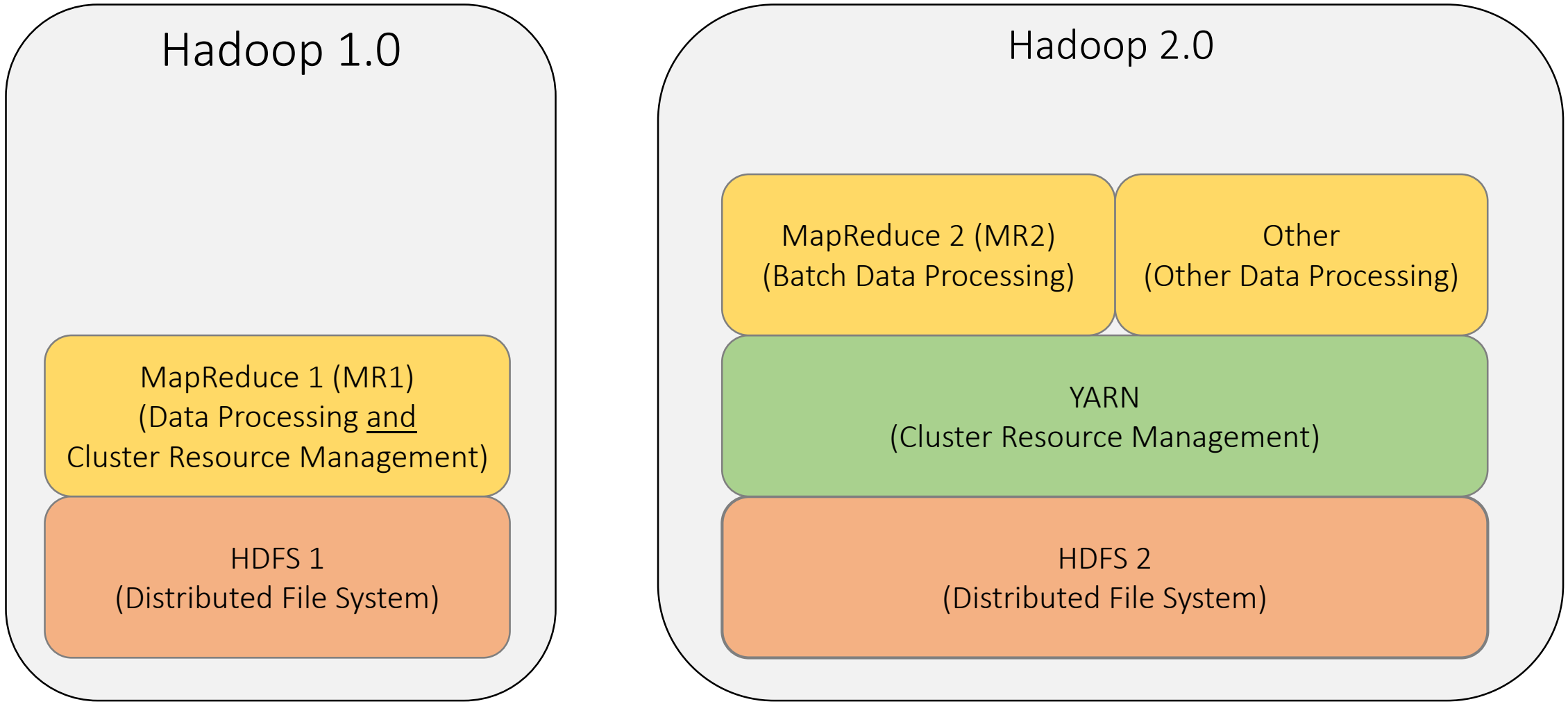


- MR1 is Non-Modular:
  - MR1 couples Data Processing and Resource Management
- MR1 has Processing Model Limitation:
  - Batch Data Processing (through MR1) is all that is supported
- HDFS1 has Single Point of Failure:
  - Single NameNode in HDFS is a single point of failure
- MR1 has a Scaling Issue:
  - Single JobTracker limits cluster sizes to 4000 nodes
- MR1 has an Underutilization Issue:
  - Nodes can only be Map or Reduce not both

Hmmm that’s pretty limiting. So do we bid Hadoop farewell?
Well no, not really, Hadoop has been evolving…

Hadoop 2.0

Hadoop 1.0 vs Hadoop 2.0



HDFS 2 (Hadoop Distributed File System)

- Removes single point of failure by providing the High Availability option of running 2 redundant NameNodes in the same cluster in an Active/Passive configuration with a hot standby.

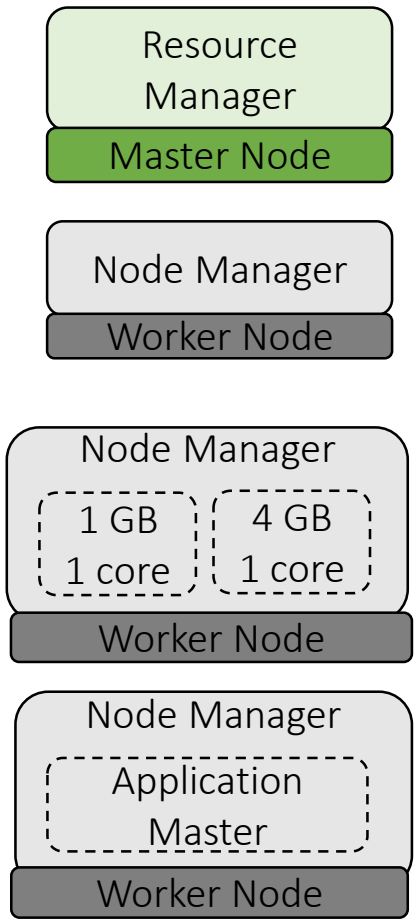
Reference: HDFS 2

YARN (Yet Another Resource Negotiator)

- Allows multiple access engines to use Hadoop as the common standard for batch, interactive and real-time processing that can simultaneously access the same data set. Not restricted to only MR batch jobs anymore.

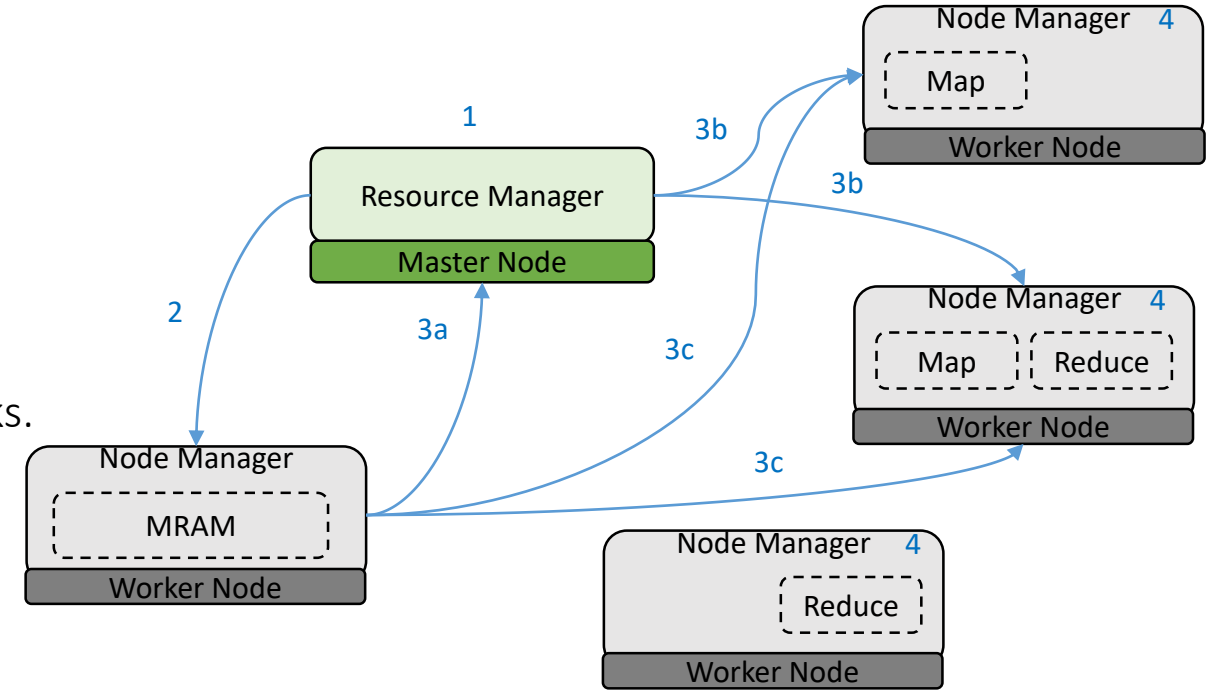
- Four separate entities:
  - The **Resource Manager** (RM) runs on the Master Node. It is the global resource scheduler. It arbitrates system resources between competing applications. It manages worker nodes, manages container creation and de-allocation, manages AM container creation and AM resource requests
  - The **Node Manager** (NM) runs on the Worker Nodes. It communicates node status and container status to RM, launches AM on request from RM, launches application processes on request from AM.
  - **Containers** are created by RM upon request. They allocate a certain amount of CPU/memory resources on the Worker Nodes. Applications run in one or more Containers.
  - There is one **Application Master** (AM) per Application. It runs in a Container. An AM can request more Containers to run Application Tasks.

- Fault Tolerance:
  - If Task fails AM will reattempt task
  - If Application fails or AM not responding, RM will reattempt whole application
  - If Worker Node not responding, removed from active node list and tasks on it considered failed by AM. If the AM is on the failed Worker Node, it is treated as a failed Application
  - If RM unavailable, no application can be launched. Have option to set up RM with High Availability configuration (run 2 redundant RMs in an Active/Passive configuration with a hot standby).



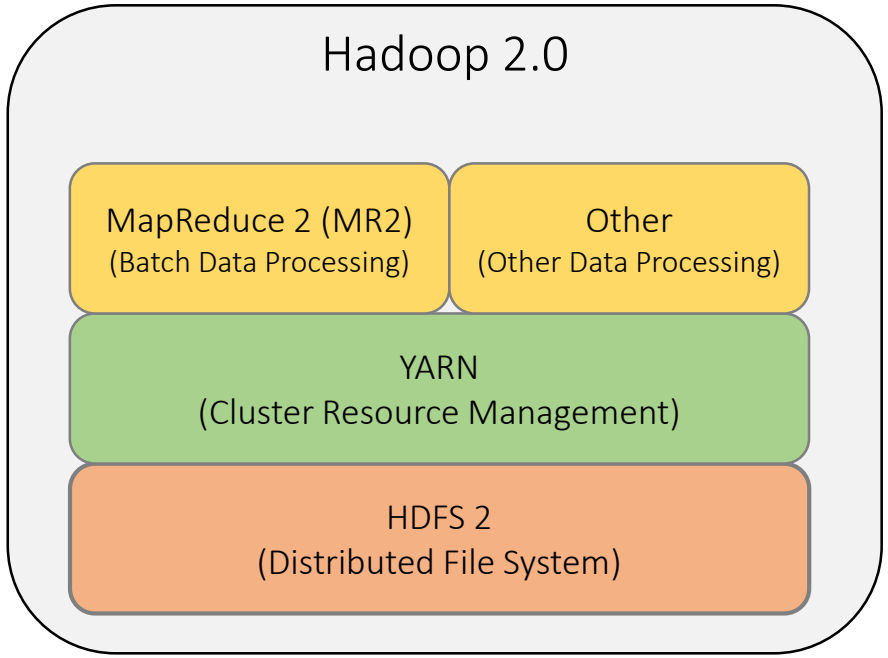
Reference: Hadoop, Hortonworks, Cloudera

MapReduce 2 (MR2)



- MapReduce is now just an application with Map and Reduce Tasks.
- It uses HDFS for data storage
- It uses YARN for resource management
    1) The Resource Manager runs on the Master Node
    2) RM creates a Container for the MapReduce Application Master (MRAM) on a Worker Node
    3) (a) The MRAM requests RM for Containers for Map Tasks
       (b) The RM create Containers for Map Tasks on Worker Nodes
       (c) The MRAM runs Map Tasks in Containers on Worker Nodes
    4) Shuffle runs as an auxiliary service on Worker Nodes
    5) (a) The MRAM requests RM for Containers for Reduce Tasks
       (b) The RM create Containers for Reduce Tasks on Worker Nodes
       (c) The MRAM runs Reduce Tasks in Containers on Worker Nodes
    6) RM deallocates Containers once Application completes

Hadoop 2.0 <HDFS2 + YARN + MR2/Other>



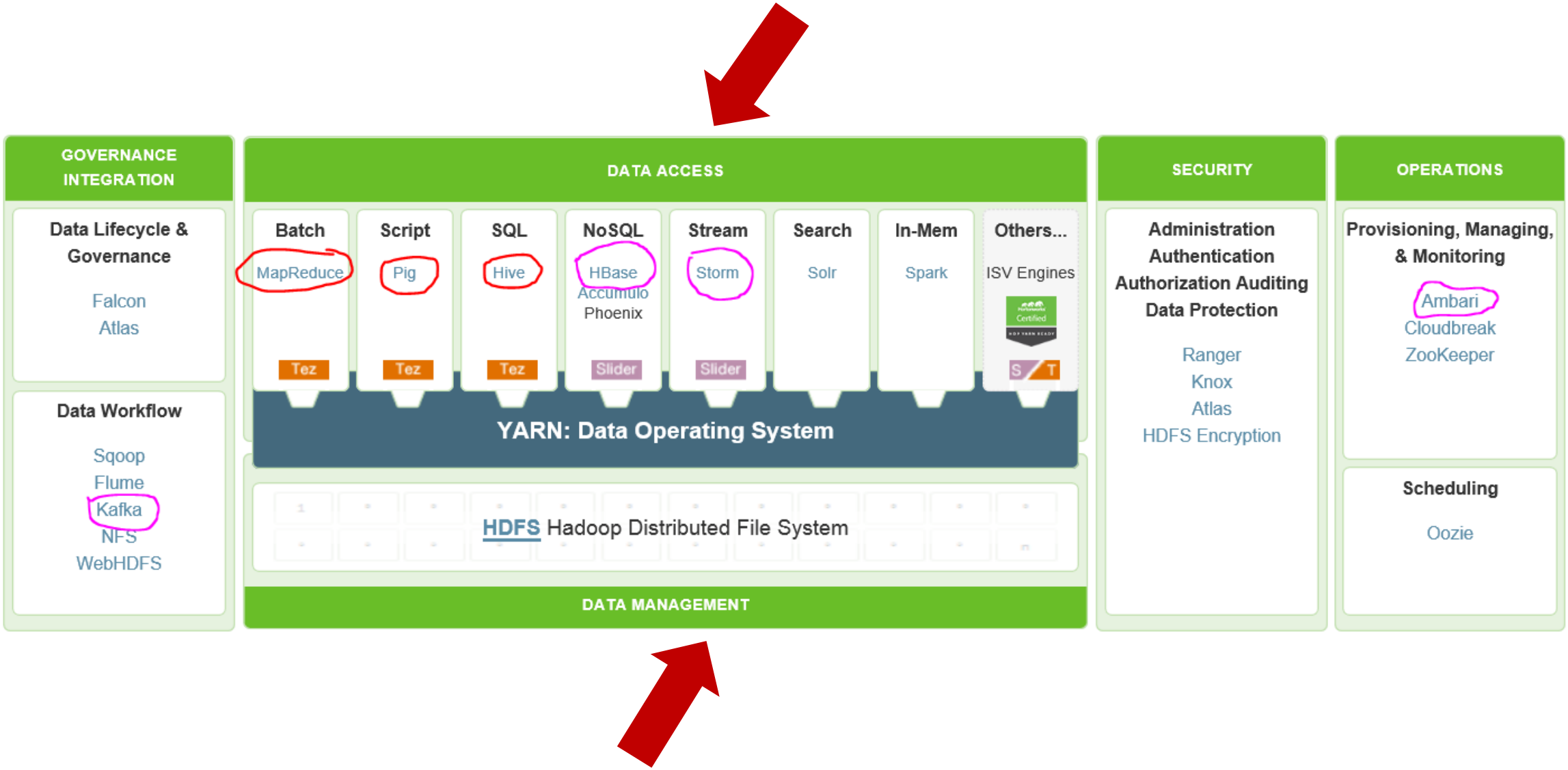
- YARN:
  - Allows multiple access engines to use Hadoop as the common standard for batch, interactive and real-time engines that can simultaneously access the same data set. So not restricted to only MR batch processing anymore.
- MR2 vs YARN:
  - Decouples (Batch) Data Processing from Cluster Resource Management
- HDFS2:
  - Removes single point of failure by providing the High Availability option of running 2 redundant NameNodes in the same cluster in an Active/Passive configuration with a hot standby.
- Better Scaling:
  - JobTracker is gone, and replaced by better Resource Management through YARN
- Better Utilization:
  - Nodes have “resources” (CPU and Memory) which are allocated as they are requested

Hortonworks Hadoop

Hadoop Distributions

- There are many Hadoop distributions – that essentially package Apache Hadoop in more stable, user-friendly, or enterprise-ready ways
- Chief among these are Hortonworks and Cloudera
- Lots of Cloud Service providers (such as Amazon AWS and Microsoft Azure) are also now providing “Hadoop as a Service”
- We are going to look at the Hortonworks distribution in more detail in the next few lectures

Hortonworks Data Platform (HDP)



Source: https://hortonworks.com/ecosystems/