Could Computing, ITNT404

Lecturer: Dr. Omar Abusada

E-mail: omar.abusaeeda@uot.edu.ly

Hadoop As Development Tools

Motivations

- How **Google** and **Facebook** for instance are able to quickly deal with such large quantities of information?
- The majority of the data in the world was mostly generated in the last few years, and this accelerated trend is going to continue.
- All this new data is coming from smartphones, social networks, trading platforms, machines, and other sources.

Motivations

- All the world turned digital
- Storing and processing data is a big concern.
- Recently, **1000's** of data are generated in ***μ.Sec*** in different format (**audio, image video and documents**).
- This type of data is known as **<u>BIG DATA</u>.**
- Typical storage of data and typical process of data is not sufficient for **BIG DATA**.
- **Solution?** Multiple storage unit and multiple processors run in parallel. The concept is termed as **Hadoop.**

Big Data (Mobile Device Data)

- The amount of mobile data at the start of **2014**, uploaded and downloaded was around **2 Exabytes** (**1 Exabyte = 1 billion gigabytes**) of data.

- At the start of **2017**, data created on mobile devices quadrupled to over **8 Exabytes**.

- At the start of **2017**, there were **3.394 billion mobile internet users**. This means that in **2017** there are more mobile internet users than desktop internet users, with mobile being used to access **51.4% of web pages** and **desktop to access 43.4%** (tablet is used for **4.9%** and other devices for the remaining **0.3%**)

Big Data (Mobile Device Data)

Big Data (Mobile Device Data)

- Approximately **21.9 billion** text messages are sent each day in **2017,** compared to **18.7 billion** in **2016** – a **17% increment in one year**.

- Due to a **social media**, and **IoT**, and other mine of data we arrive to generate **2.5 Quintillion** Bytes (18 zeros) each day in **2015**

History Of Hadoop

- Hadoop was started with **Doug Cutting** and **Mike Cafarella** in the **year 2002** when they both started to work on **Apache Nutch** project of building a search engine system that **can** index 1 billion page. Estimate cost; around **$500,000** in hardware with a monthly **running cost of $30,000.**
- **Apache Nutch** project has been inspired by Google's File System (**GFS**) which was detailed in a paper released by **Google** in **2003.**
- In 2004, **Nutch's** developers set about writing an open-source implementation, the **Nutch Distributed File System (NDFS).** Same year, Google introduced **MapReduce** to the world by releasing a paper on **MapReduce**

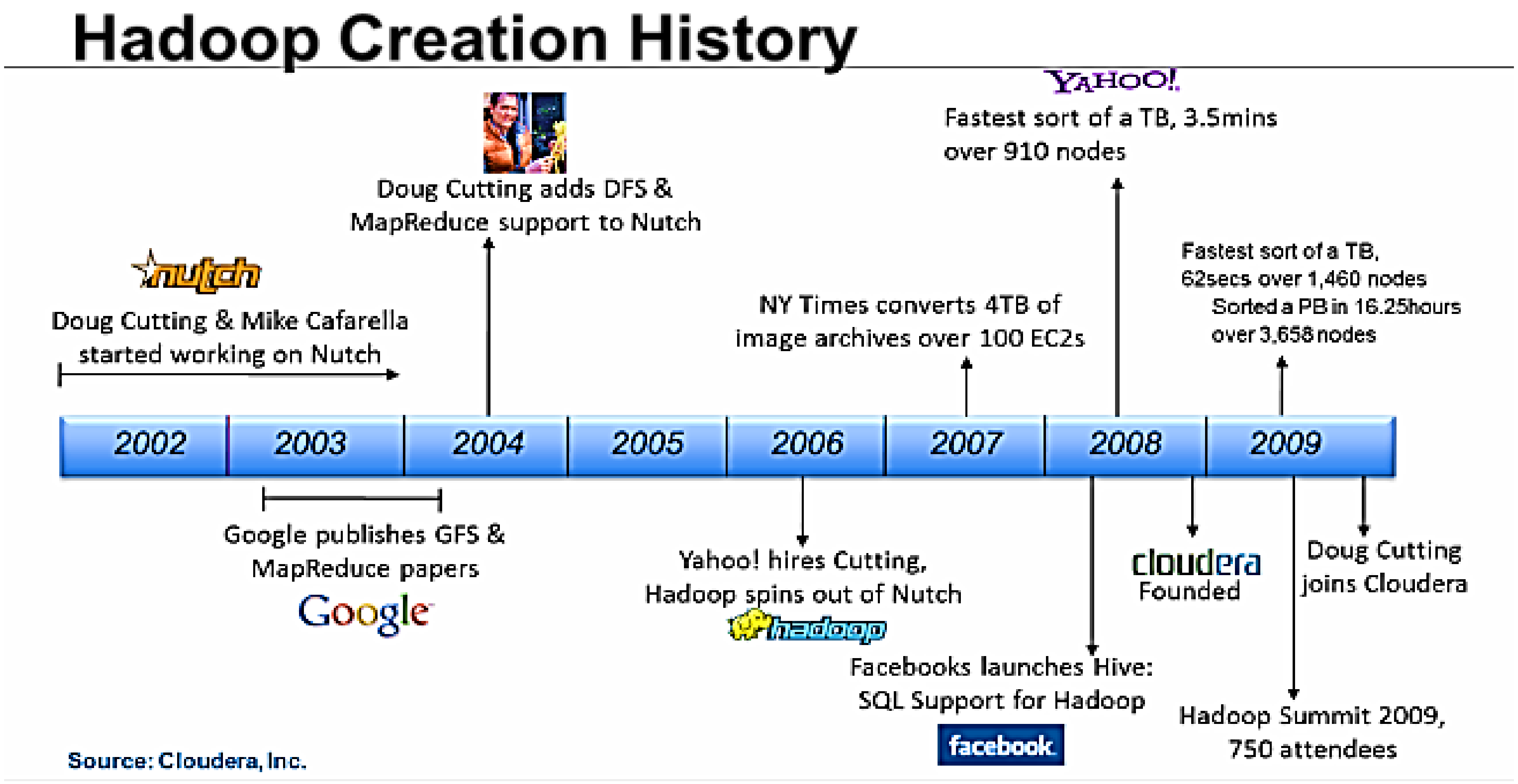
History Of Hadoop

- **Hadoop**, originally called **Nutch Distributed File System (NDFS)** split from **Nutch in 2006** to become a sub-project of **Lucene**. At this point it was renamed to **Hadoop**.
- In **2007**, Yahoo started using **Hadoop** on **1000 nodes cluster**.
- In **January 2008**, Hadoop **confirmed its success by becoming the top-level project at Apache.**
- In **November 2008**, Google reported that its **MapReduce** implementation sorted **1 terabyte in 68 seconds.**
- In **April 2009**, a team at Yahoo used **Hadoop** to **sort 1 terabyte in 62 seconds**, beaten Google **MapReduce** implementation.

History Of Hadoop

- On **27 December 2011**, **Apache** released **Hadoop version 1.0** that includes support for Security.
- On **10 March 2012, release 1.0.1** was available. This is a bug fix release for **version 1.0**.
- On **23 May 2012,** the Hadoop **2.0.0-alpha** version was released. This release contains **YARN**.
- The **second (alpha) version** in the **Hadoop-2.x** series with a more stable version of **YARN** was released on **9 October 2012**.

History Of Hadoop



Hadoop Creation History

Doug Cutting & Mike Cafarella started working on Nutch

Google publishes GFS & MapReduce papers

Doug Cutting adds DFS & MapReduce support to Nutch

Yahoo! hires Cutting, Hadoop spins out of Nutch

NY Times converts 4TB of image archives over 100 EC2s

Facebooks launches Hive: SQL Support for Hadoop

Fastest sort of a TB, 3.5mins over 910 nodes

cloudera Founded

Fastest sort of a TB, 62secs over 1,460 nodes

Sorted a PB in 16.25hours over 3,658 nodes

Hadoop Summit 2009, 750 attendees

Doug Cutting joins Cloudera

Source: Cloudera, Inc.

Motivations

- Since most of this data is already available, the question is whether you are going to use it or not.
- Terms become very popular such as:
  - **BigData**
  - **Hadoop**
  - **MapReduce**

Hadoop

- **Hadoop**,
  - Is an open-source framework, has reformed the way we handle and process Big Data with its suite of tools.
- **Hadoop**
  - Tools are a suite of software applications and frameworks designed to facilitate the storage, processing, management, and analysis of vast volumes of data, enabling efficient Big Data operations.

Distributed File System (DFS)

- A Distributed File System (DFS) is simply a classical model of a file system distributed across multiple machines. The purpose is to promote sharing of dispersed files.
- This is an area of active research interest today.
- Clients should view a DFS the same way they would a centralized FS; the distribution is hidden at a lower level.
- A DFS provides high throughput data access and fault tolerance.

What is Hadoop?

- Open source software platform for scalable, distributed computing.
- Hadoop provides fast and reliable analysis of both structured data and unstructured data.
- Apache Hadoop software library is essentially a framework that allows for the distributed processing of large datasets across clusters of computers using a simple programming model.
- Hadoop can scale up from single servers to thousands of machines, each offering local computation and storage.

Architecture of Hadoop

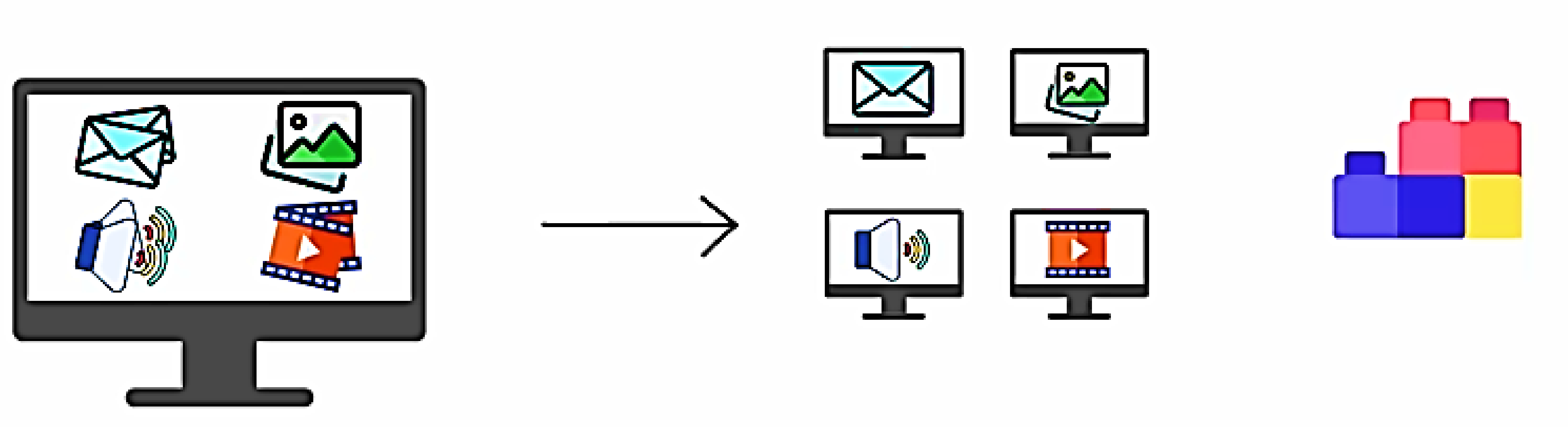
- Hadoop consists of three components that were designed to work on big data.

  - Storage unit (storage data).
  - MapReduce (processing data).
  - YARN (Resource manager).

Hadoop Component

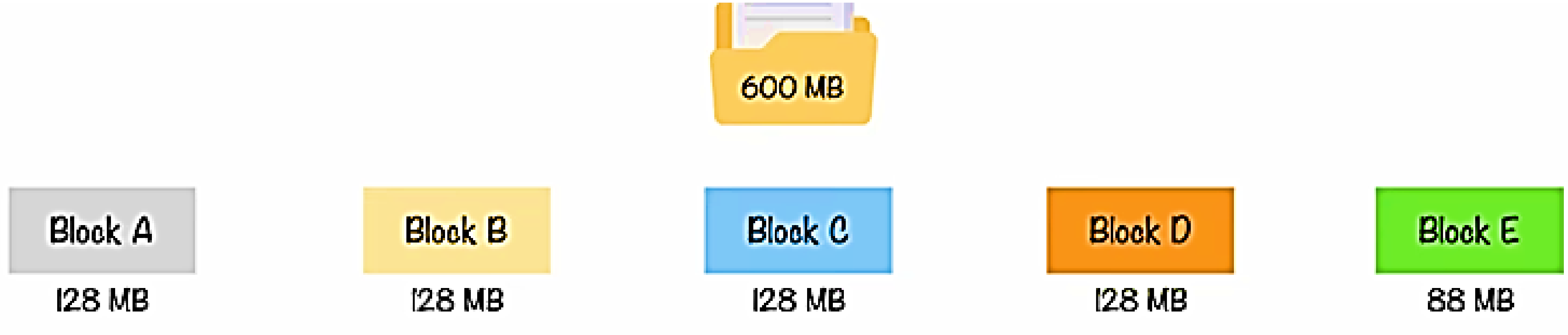
1- Storage unit (storage data).

- **Hadoop Distributed File System (HDFS)**
- Storing mass of data in one computer is unusable. Hence, data is distributed among many computers and stored in blocks.



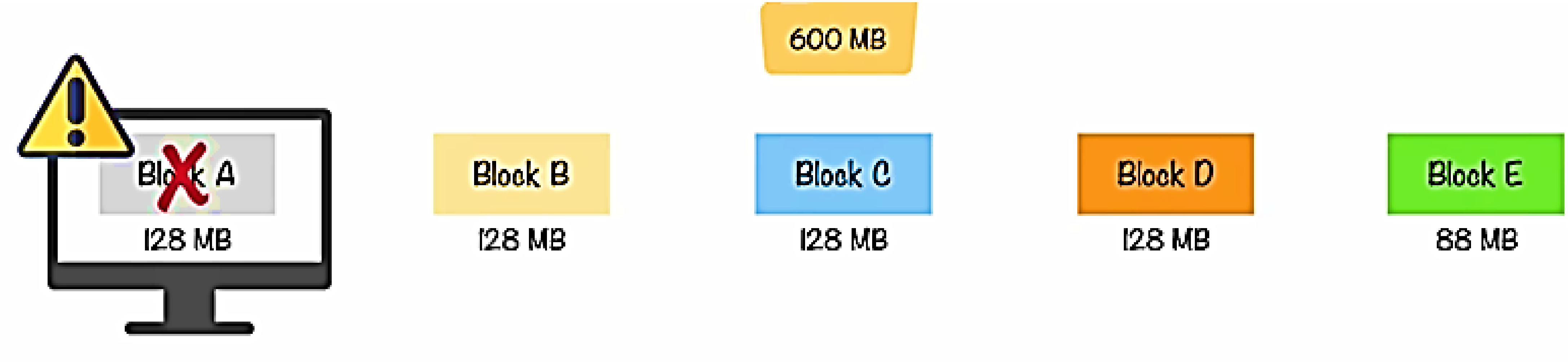
Hadoop Component

- **Hadoop Distributed File System (HDFS)**
- For example, if we have **600MB** of data, **HDFS** blocks the data in blocks of **128MB**. Thus, the data can be represented as:



Hadoop Component

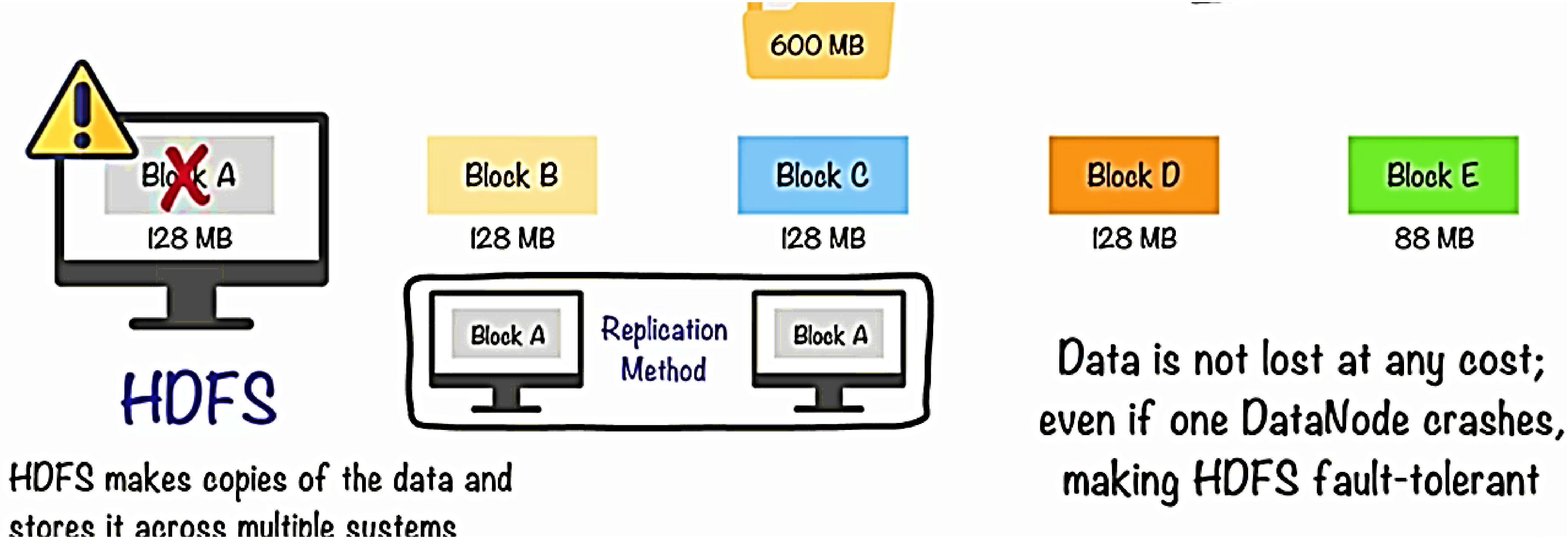
- **What about if one of the blocks is crashed?**
- **Will we loose that specific piece of data? Answer is NO. That is the beauty of HDFS.**



- HDFS makes copies of the data and stores it a cross multiple systems (Replication Method)

Hadoop Component

- **What about if one of the blocks is crashed?**
- **Will we loose that specific piece of data? Answer is NO. That is the beauty of HDFS.**

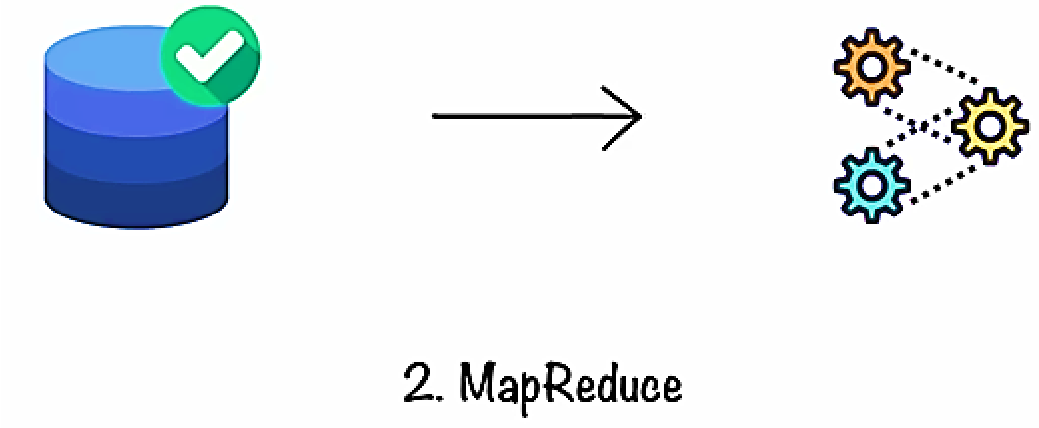

600 MB
Block A
128 MB
Block B
128 MB
Block C
128 MB
Block D
128 MB
Block E
88 MB
Block A
Replication Method
Block A
HDFS
HDFS makes copies of the data and stores it across multiple systems
Data is not lost at any cost; even if one DataNode crashes, making HDFS fault-tolerant

Hadoop Component

- Traditional data processing methods process data on single machine with single processor.
- Consume time and insufficient, specially with large volume of data .

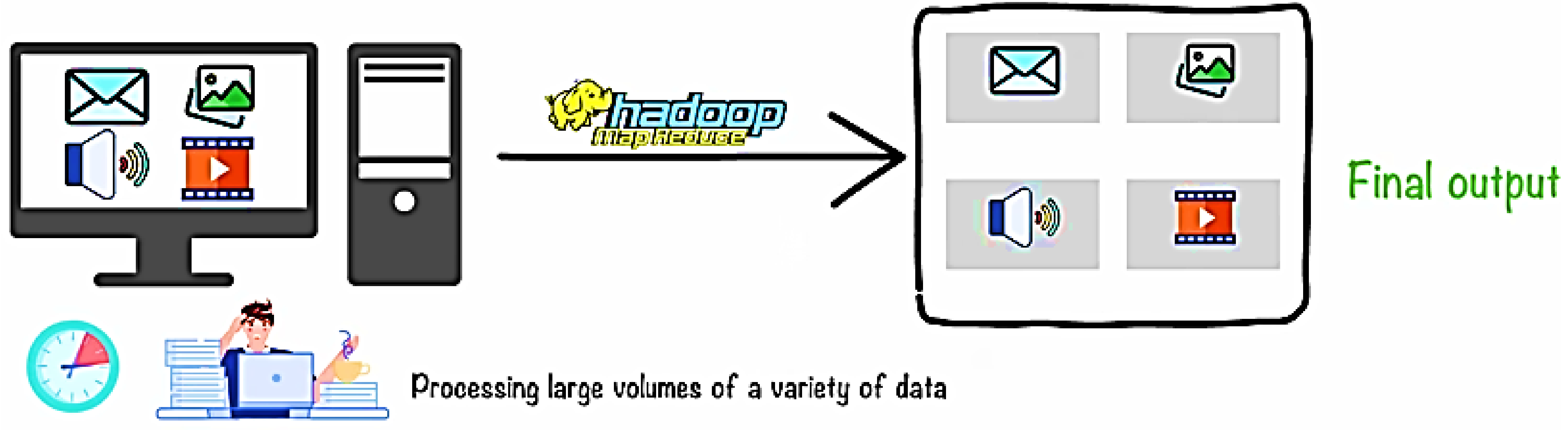
2- MapReduce (processing data).

- After the data is stored successfully, it needs to be processed.



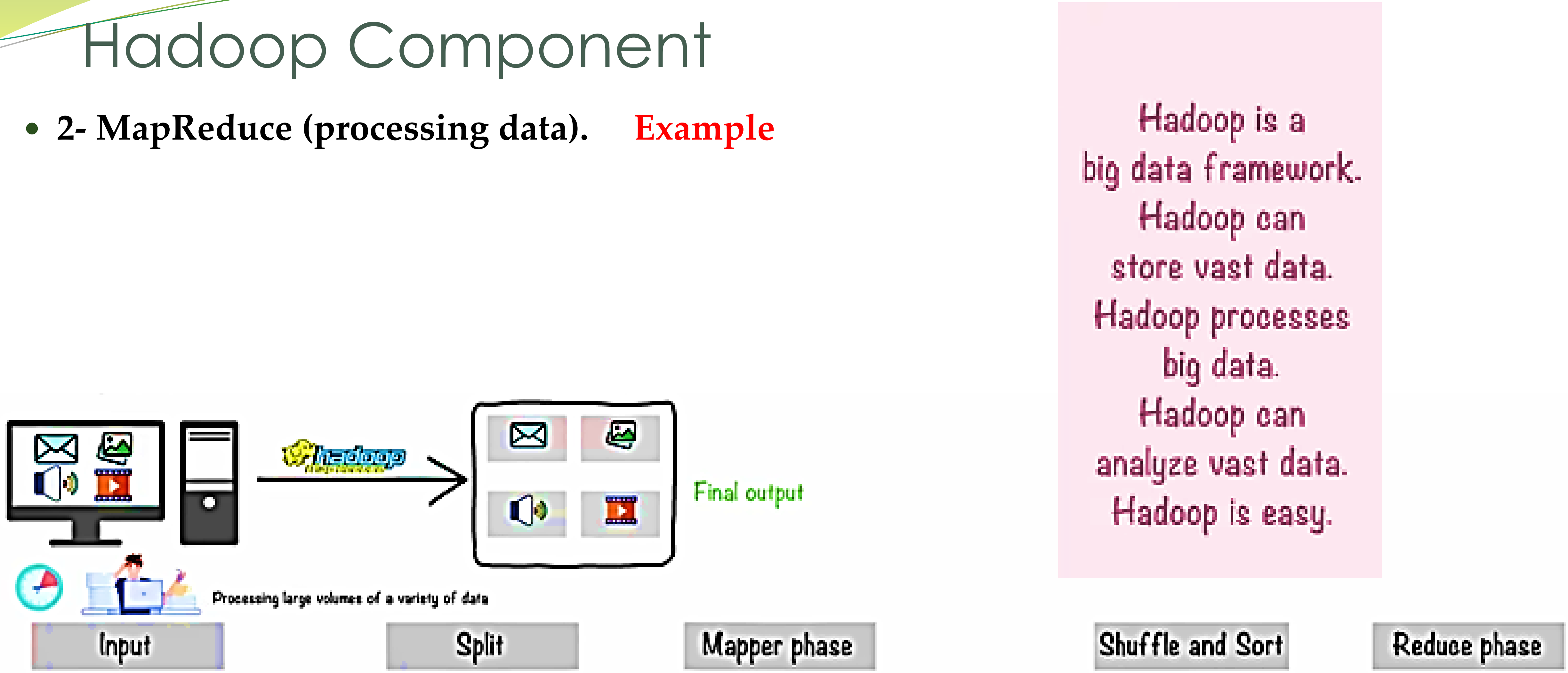
Hadoop Component

- **2- MapReduce (processing data).**
- To overcome the problem associated with single processor, **MApReduce** divide data in parts and process them separately. The individual resource are then give the final output.



Hadoop Component

- **2- MapReduce (processing data).** **Example**

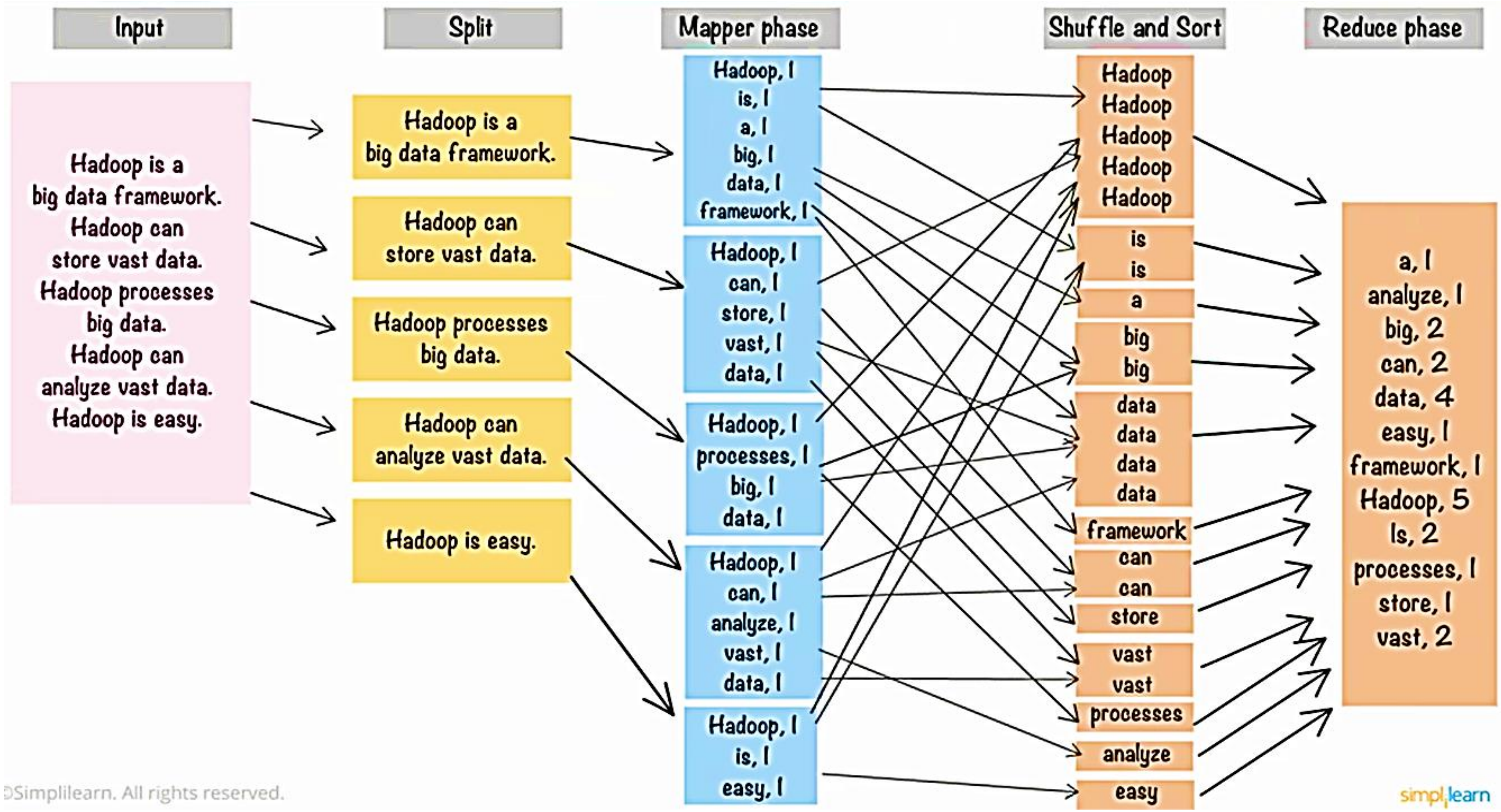


Hadoop is a
big data framework.
Hadoop can
store vast data.
Hadoop processes
big data.
Hadoop can
analyze vast data.
Hadoop is easy.

Hadoop Component

- **2- MapReduce (processing data).** Example

- Simply done by writing a simple program



Input

Hadoop is a big data framework. Hadoop can store vast data. Hadoop processes big data. Hadoop can analyze vast data. Hadoop is easy.

Split

Hadoop is a big data framework.

Hadoop can store vast data.

Hadoop processes big data.

Hadoop can analyze vast data.

Hadoop is easy.

Mapper phase

Hadoop, 1
is, 1
a, 1
big, 1
data, 1
framework, 1

Hadoop, 1
can, 1
store, 1
vast, 1
data, 1

Hadoop, 1
processes, 1
big, 1
data, 1

Hadoop, 1
can, 1
analyze, 1
vast, 1
data, 1

Hadoop, 1
is, 1
easy, 1

Shuffle and Sort

Hadoop
Hadoop
Hadoop
Hadoop
Hadoop

is
is

a

big
big

data
data
data
data

framework

can
can

store

vast
vast

processes

analyze

easy

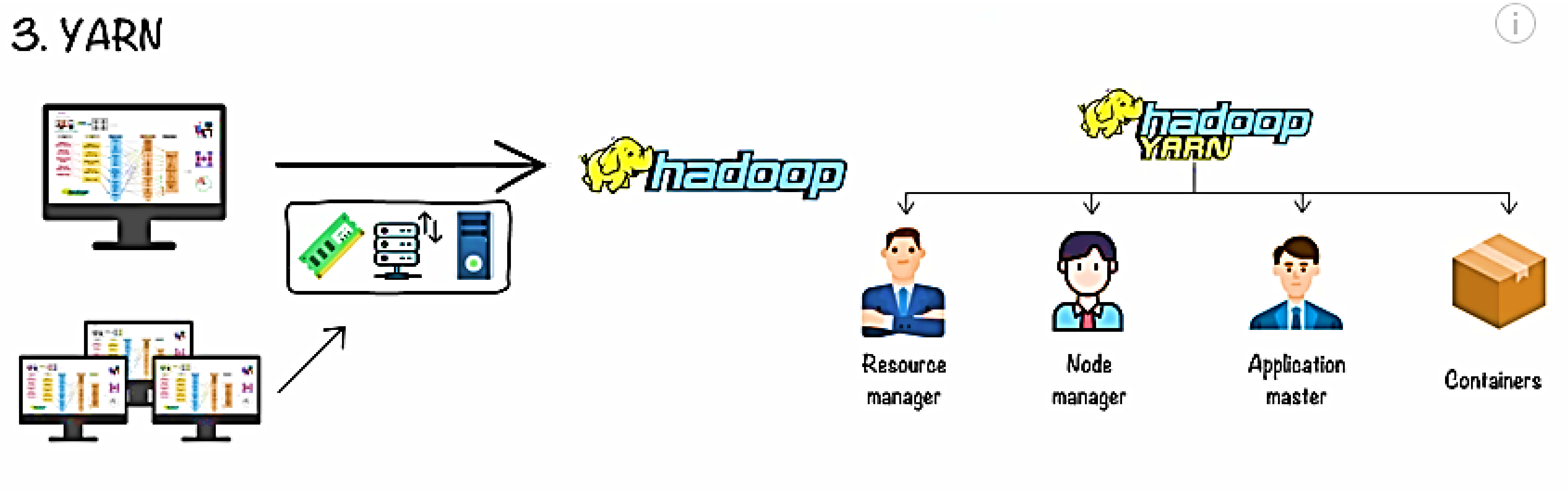
Reduce phase

a, 1
analyze, 1
big, 2
can, 2
data, 4
easy, 1
framework, 1
Hadoop, 5
Is, 2
processes, 1
store, 1
vast, 2

Hadoop Component

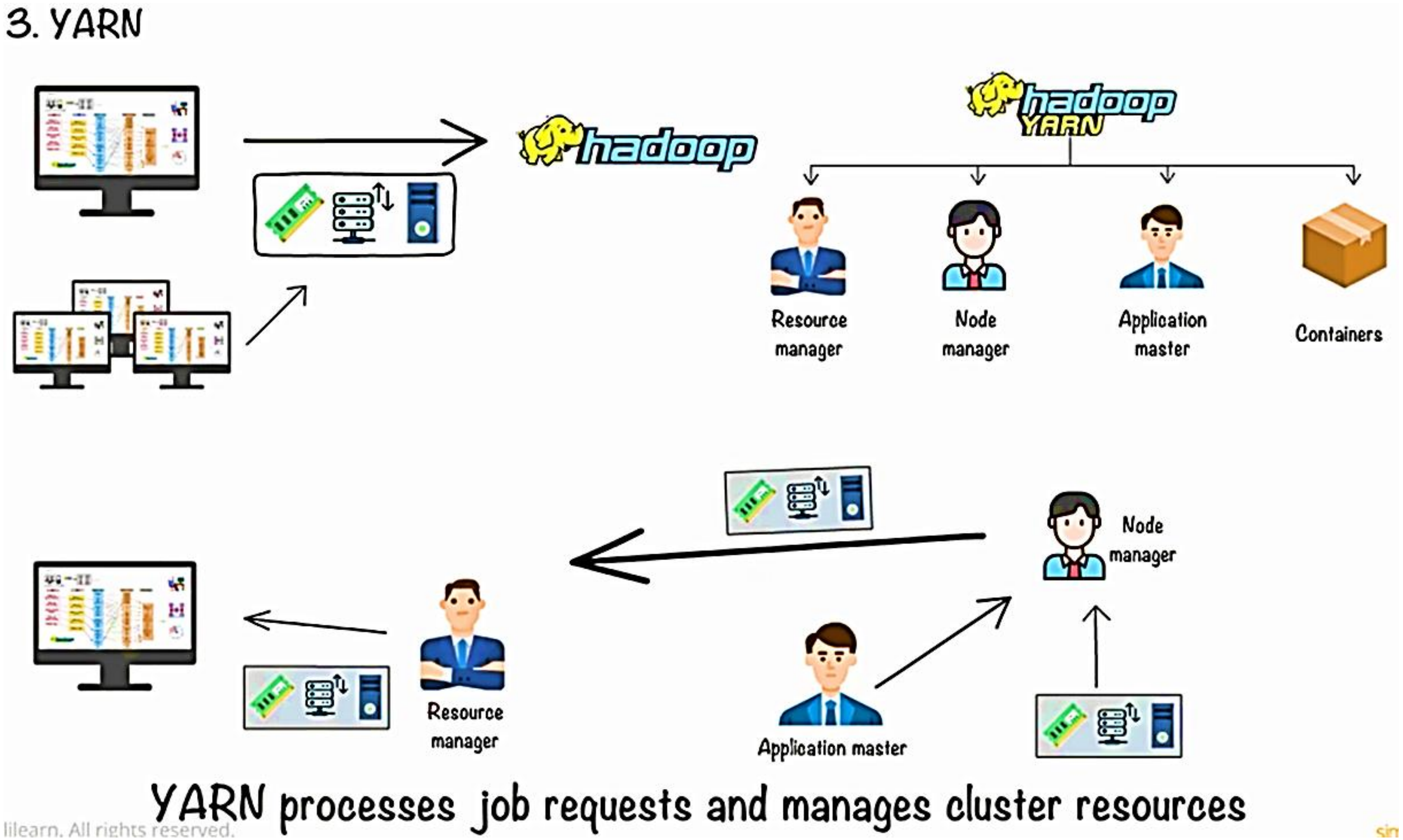
3- YARN (Resource manager).

- Once data is ready from the previous stage it is a time to run YARN.
- This is done by the help of set of resources such as **RAM**, **CPU**, and **Network bandwidth**.
- Multiple jobs are run semantically.
- Each of them need some resources to complete the task successful.

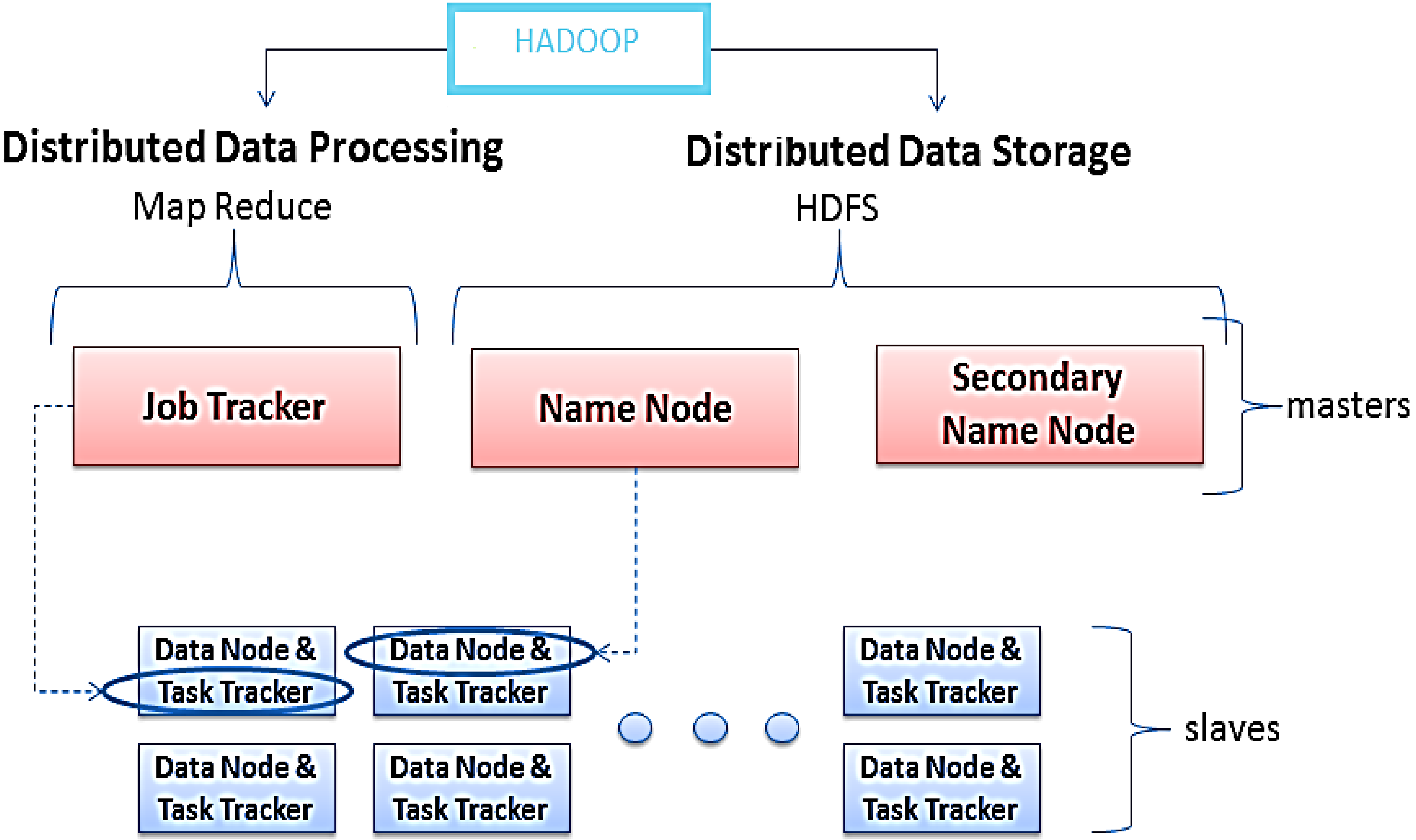


Hadoop Component

3- YARN (Resource manager).



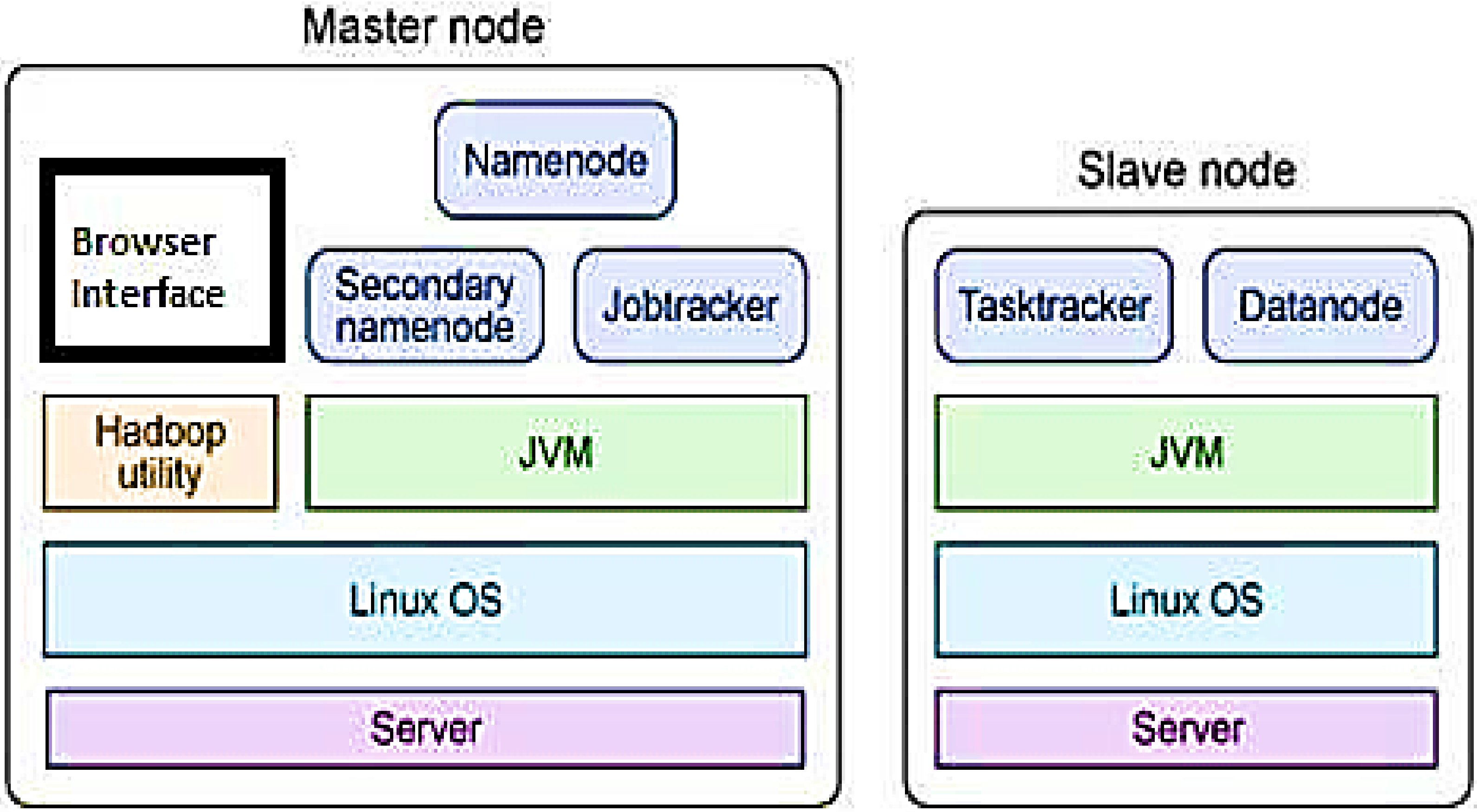
Hadoop Server Roles



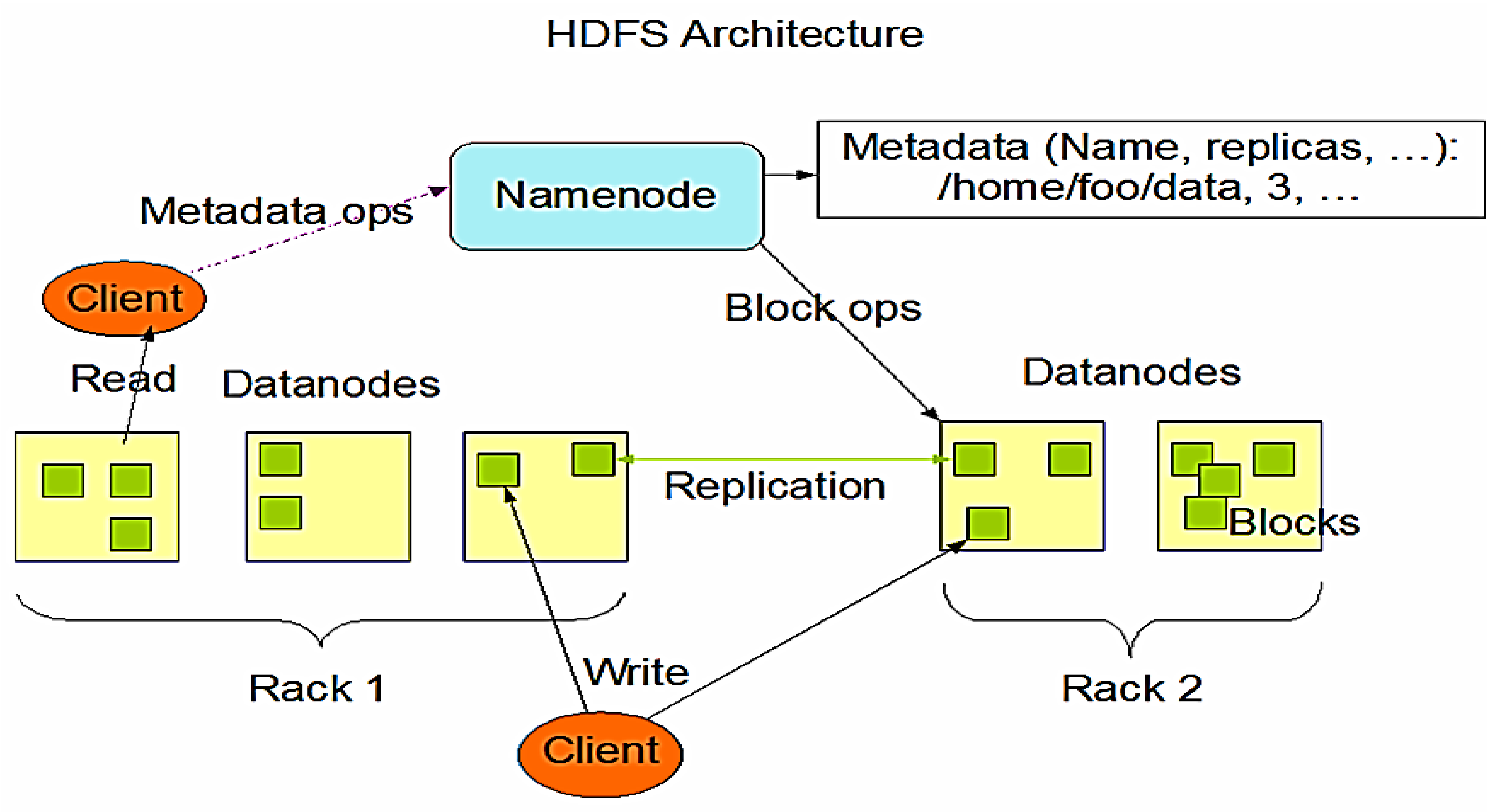
HDFS (Hadoop Distributed File System)

- A distributed file system that provides high-throughput access to application data
- HDFS uses a **master/slave** architecture in which one device (master) termed as **NameNode** controls one or more other devices (slaves) termed as **DataNode**.
- It breaks Data/Files into small blocks (128 MB each block) and stores on **DataNode**.
- **Each block replicates on other nodes to accomplish fault tolerance.**
- **NameNode** keeps the track of blocks written to the **DataNode**.

HDFS Cluster Architecture



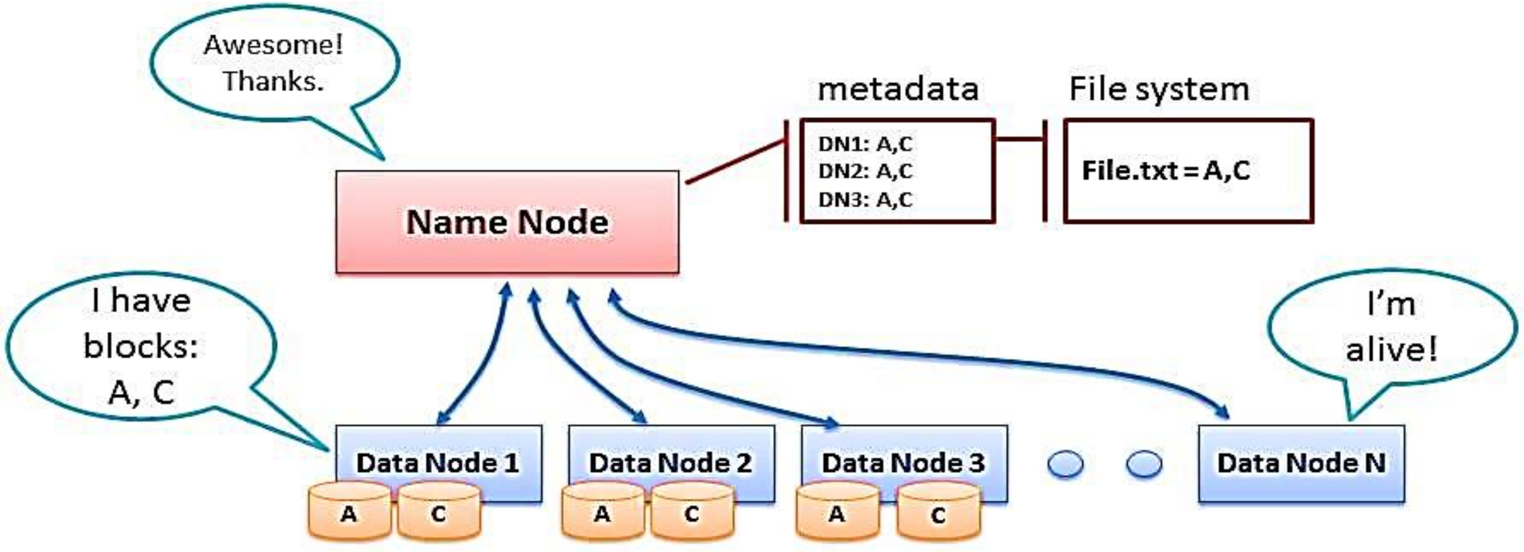
HDFS Architecture



Name Node

- Keeps the metadata of all files/blocks in the file system, and tracks where across the cluster the file data is kept.
- It does not store the data of these files itself. Kind of block lookup dictionary(index or address book of blocks).
- **Client applications talk to the NameNode** whenever they wish to locate a file, or when they want to add/copy/move/delete a file.
- **The NameNode responds the successful** requests by returning a list of relevant **DataNode servers** where the data lives.

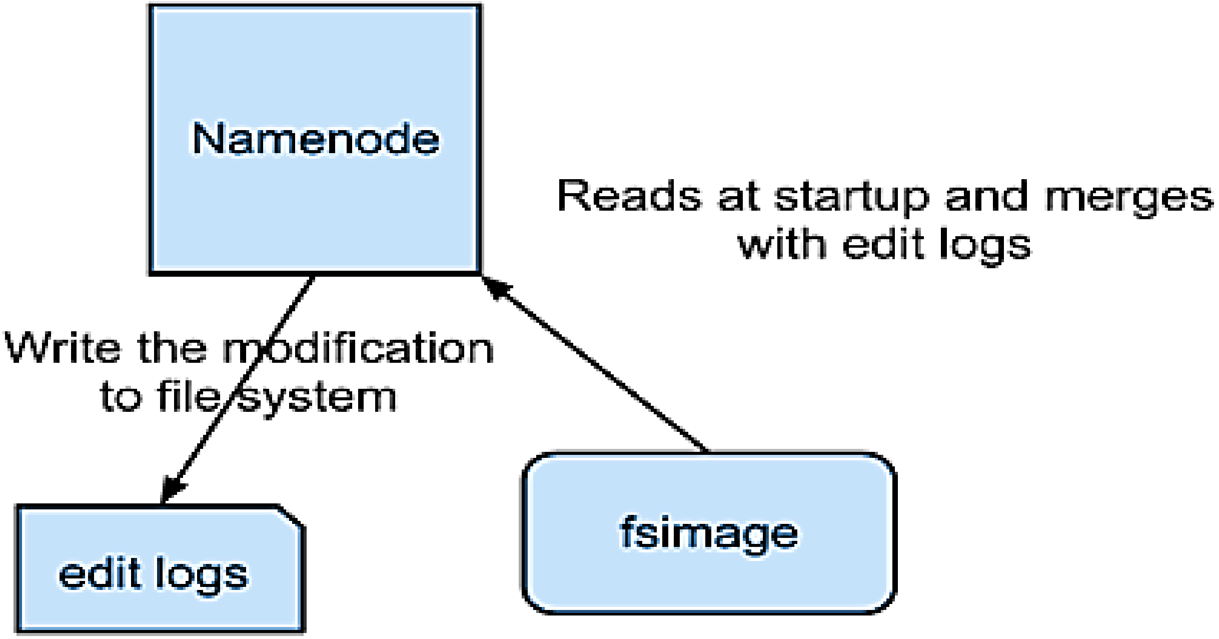
Name Node



- Data Node sends Heartbeats
- Every 10th heartbeat is a Block report
- Name Node builds metadata from Block reports
- TCP – every 3 seconds
- If Name Node is down, HDFS is down

Name Node

- **FSimage** - Its the snapshot of the **filesystem** when **NameNode** started
- **Edit logs** - Its the sequence of changes made to the filesystem after **NameNode** started


Namenode
Reads at startup and merges with edit logs
Write the modification to file system
edit logs
fsimage

Data Node

- **DataNode** stores data in the Hadoop Filesystem

- A functional filesystem has more than one **DataNode**, with data replicated across them

- On startup, a **DataNode** connects to the **NameNode**; spinning until that service comes up. It then responds to requests from the **NameNode** for filesystem operations.

- Client applications can talk directly to a **DataNode**, once the **NameNode** has provided the location of the data

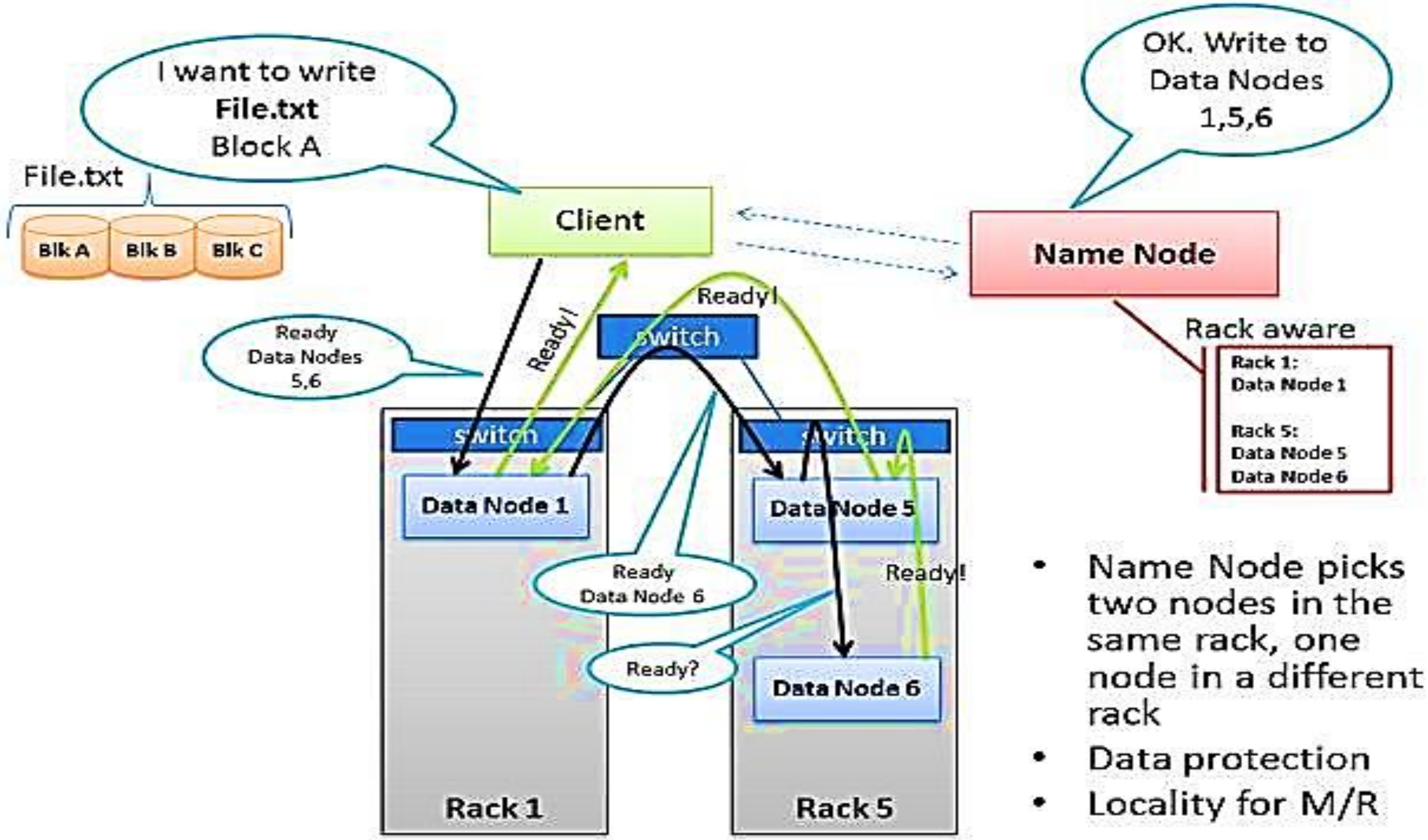
Data Replication

- **Why need data replications ?**
  - HDFS is designed to handle large scale data in distributed environment.
  - Hardware or software failure, or network partition exist
  - Therefore need replications for those fault tolerance
- **Replications factor**
  - Decided by users, and can be dynamically tuned.

Data Replication

- **How to Create replications efficiently ?**
  - HDFS is designed to handle large scale data in distributed environment.
  - Replication pipeline: Instead of single machine create replications, a pipe line is applied
  - Machine 1 make replication to machine 2, at the same time machine 2 make the replication to machine 3, etc.
- **Replications placement**
  - High initialization time to create replication to all machines
  - An approximate solution: Only 3 replications One replication resides in current node One replication resides in current rack One replication resides in another rack

Data Replication



- Name Node picks two nodes in the same rack, one node in a different rack
- Data protection
- Locality for M/R

Data Node Failure

- Data Node Failure Condition

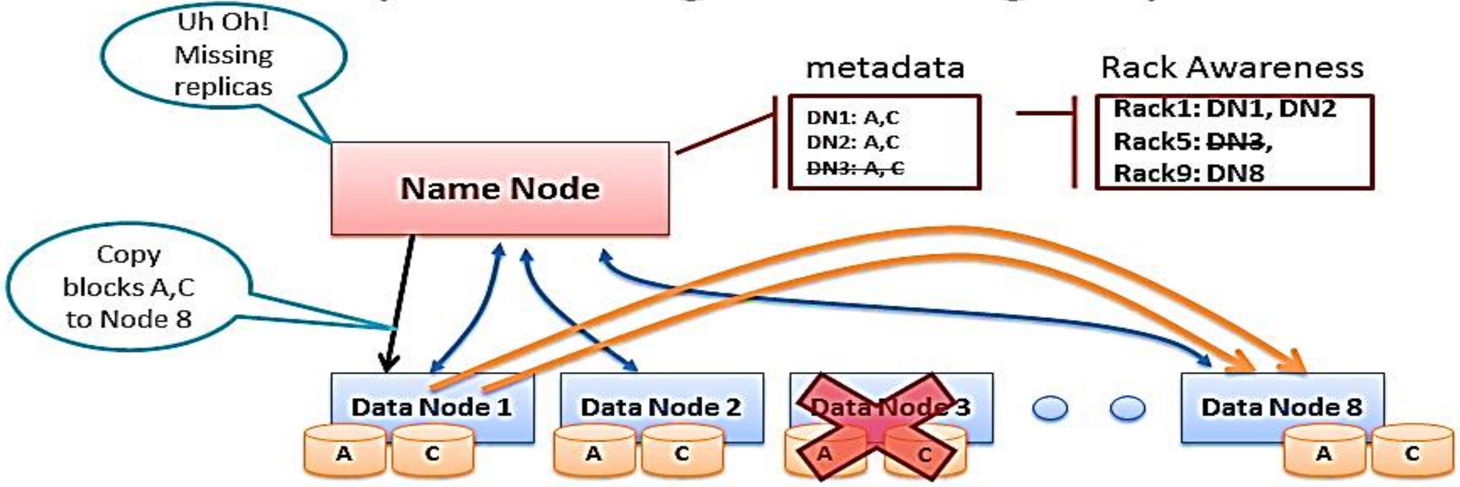
- If a **DataNode** failed, **NameNode** could know the blocks it contains, create same replications to other alive nodes, and unregister this dead node

- Data Integrity: Corruption may occur in network transfer, Hardware failure etc.
- Apply **checksum checking** on the contents of files on HDFS, and store the **checksum** in **HDFS namespace**

- If checksum is not correct after fetching, drop it and fetch another replication from other machines.

Heartbeats and Re-Replication

Re-replicating missing replicas

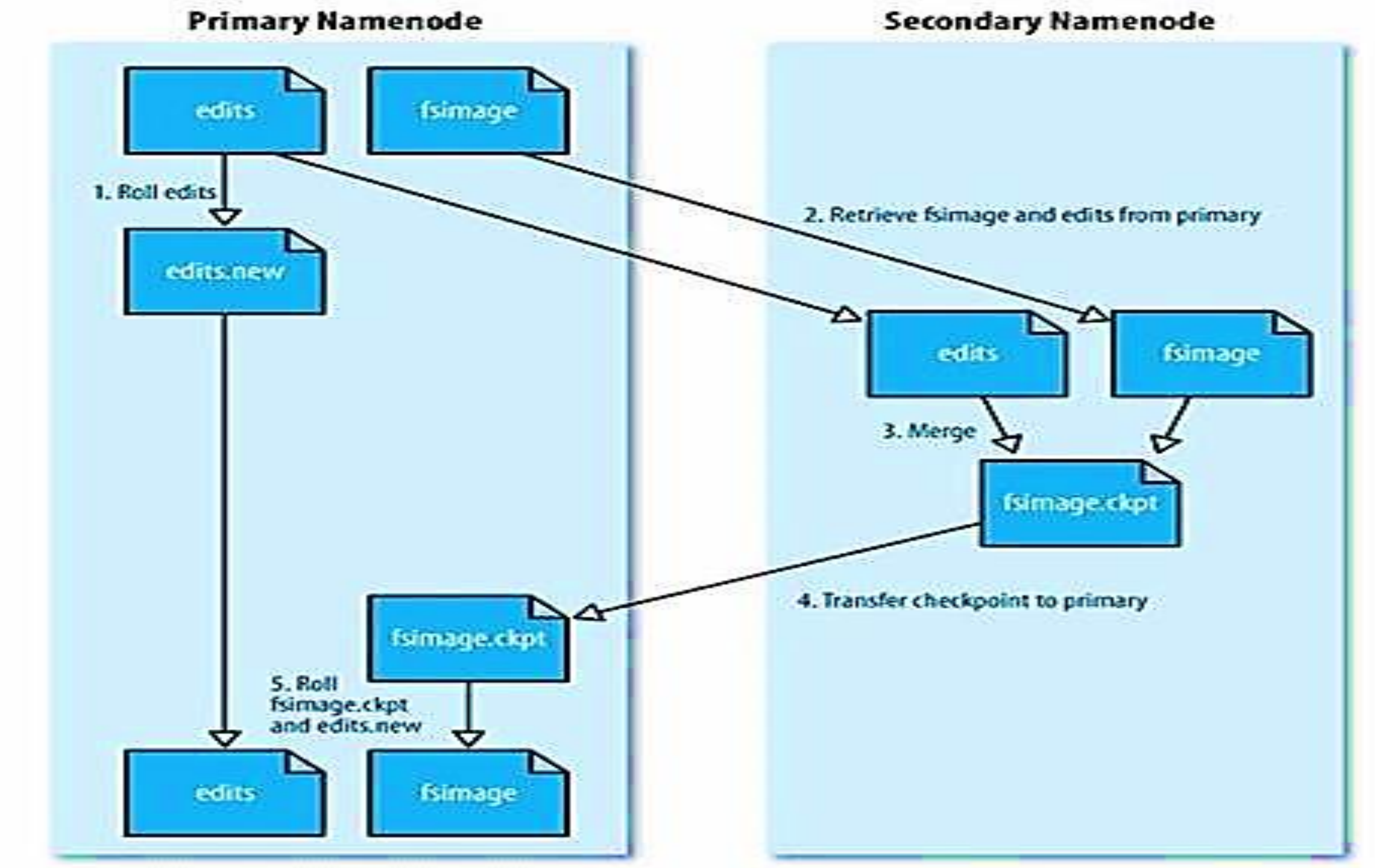


- Missing Heartbeats signify lost Nodes
- Name Node consults metadata, finds affected data
- Name Node consults Rack Awareness script
- Name Node tells a Data Node to re-replicate

Secondary Name Node

- Not a failover **NameNode**
- The only purpose of the **secondary name-node** is to perform periodic checkpoints. The **secondary name-node** periodically **downloads current name-node image and edits log files,** joins them into new image and uploads the new image back to the (primary and the only) name-node
- **Default checkpoint time is one hour**. It can be set to one minute on highly busy clusters where lots of write operations are being performed.

Secondary Name Node



Name Node Failure

- **NameNode** is the single point of failure in the cluster.
- If NameNode is down due to software glitch, restart the machine.
- If original **NameNode** can be restored, secondary can re-establish the most current metadata snapshot.
- If machine don’t come up, metadata for the cluster is irretrievable. In this situation create a new **NameNode**, use secondary to copy metadata to new primary, restart whole cluster.

… Thank you …