Inspire…Educate…Transform.

Big Data & Spark

Foundations, HDFS Storage & Spark

Bala Subrahmanya Sharma Vedam

This presentation may contain references to findings of various reports available in the public domain. INSOFE makes no representation as to their accuracy or that the organization subscribes to those findings.

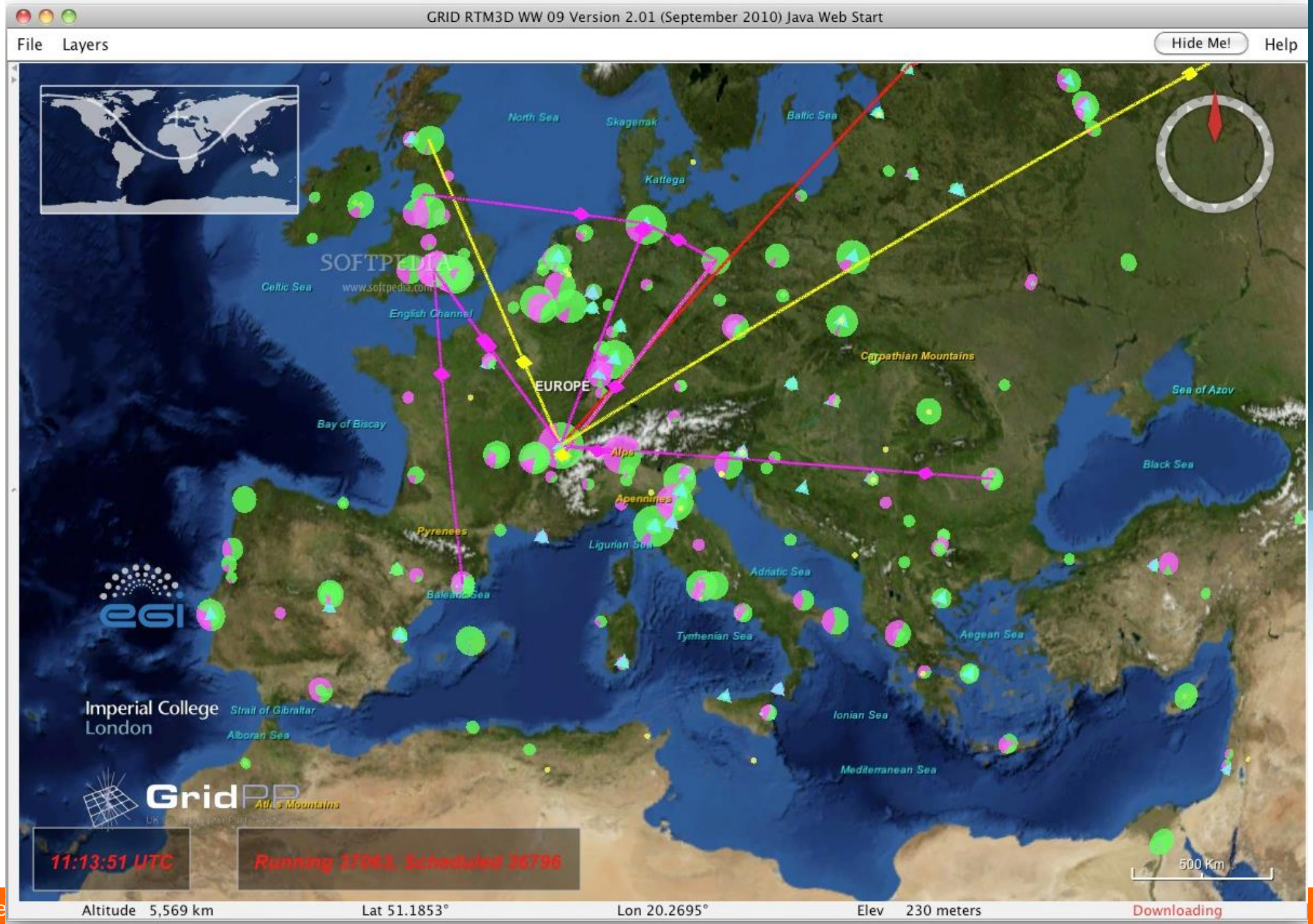
Topics:-

- Different Architectures
- Cloud: Transition from Databases to data warehouses and data lakes
- What is BigData
- What is Hadoop
- Major Components of Hadoop
- HDFS, MR, YARN, SPARK

Different Architectures

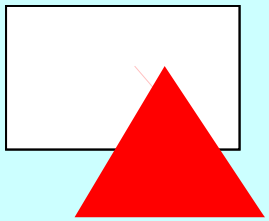
- Cellular Computing
  - IBM Cell/BE and Cyclops64 (https://en.wikipedia.org/wiki/Cell_(microprocessor))
- Grid Computing
  - SETI(https://www.seti.org/, https://www.youtube.com/watch?v=_aIJV5aQR68)
  - Worldwide LHC Grid (https://en.wikipedia.org/wiki/Worldwide_LHC_Computing_Grid)
  - Primary funding is from grant agencies (https://en.wikipedia.org/wiki/TeraGrid,https://www.nsf.gov/pubs/2018/nsf18513/nsf18513.htm)
- Cluster Computing
- Cloud Computing

A Sample Grid

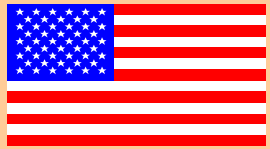


Some Grid Projects & Initiatives

- Australia
  - Nimrod-G
  - Gridbus
  - GridSim
  - Virtual Lab
  - DISCWorld
  - GrangeNet.
  - ..etc
- Europe
  - UK eScience
  - EU Data Grid
  - Cactus
  - XtremeWeb
  - ..etc.
- India
  - I-Grid
- Japan
  - Ninf
  - DataFarm
- Korea...
  - N*Grid
- Singapore
  - NGP

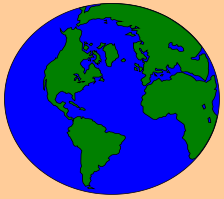


USA
- AppLeS
- Globus
- Legion
- Sun Grid Engine
- NASA IPG
- Condor-G
- Jxta
- NetSolve
- AccessGrid
- and many more...



Cycle Stealing & .com Initiatives
- Distributed.net
- SETI@Home, ….
- Entropia, UD, SCS,….

Public Forums
- Global Grid Forum
- Australian Grid Forum
- IEEE TFCC
- CCGrid conference
- P2P conference

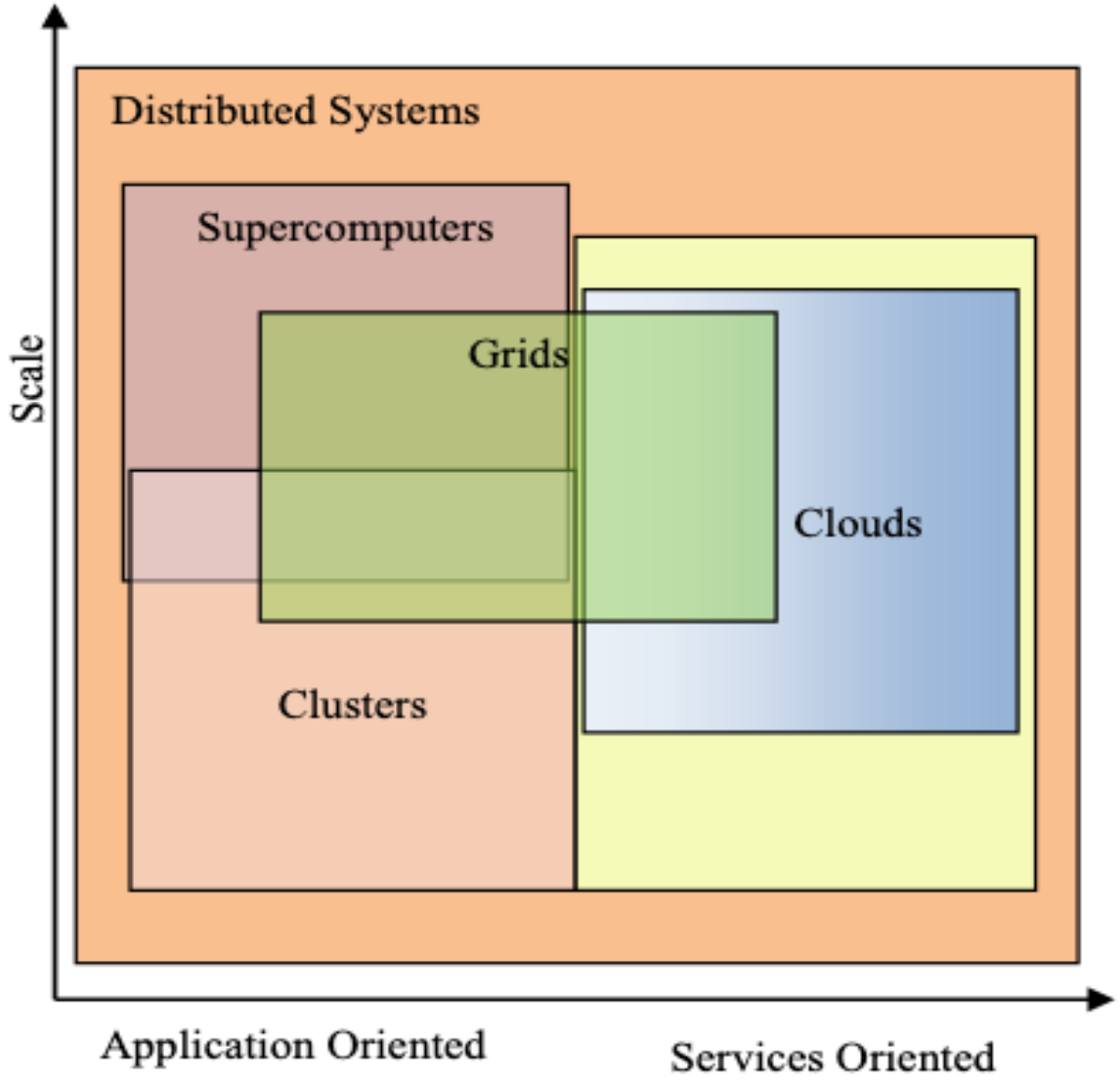


Figures due to Rajkumar Buyya, University of Melbourne, Australia, www.gridbus.org

Cluster Computing

- A cluster is a type of parallel or distributed processing system, which consists of a collection of interconnected stand-alone computers cooperatively working together as a single, integrated computing resource.

- A typical cluster:
  - Network: Faster, closer connection than a typical network (LAN)
  - Low latency communication protocols
  - Loose connections

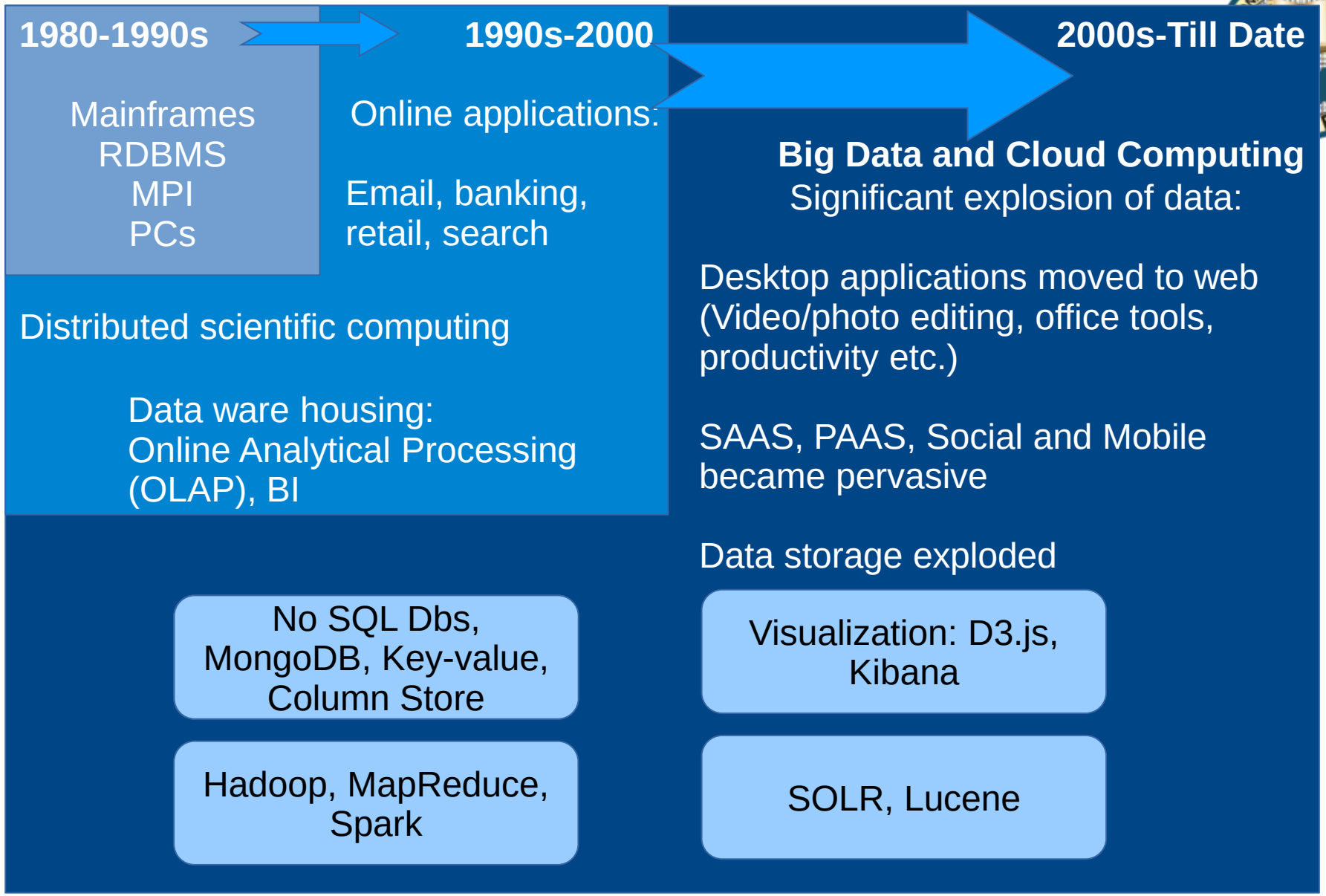

Distributed Systems
Supercomputers
Grids
Clouds
Clusters
Scale
Application Oriented
Services Oriented

Agenda

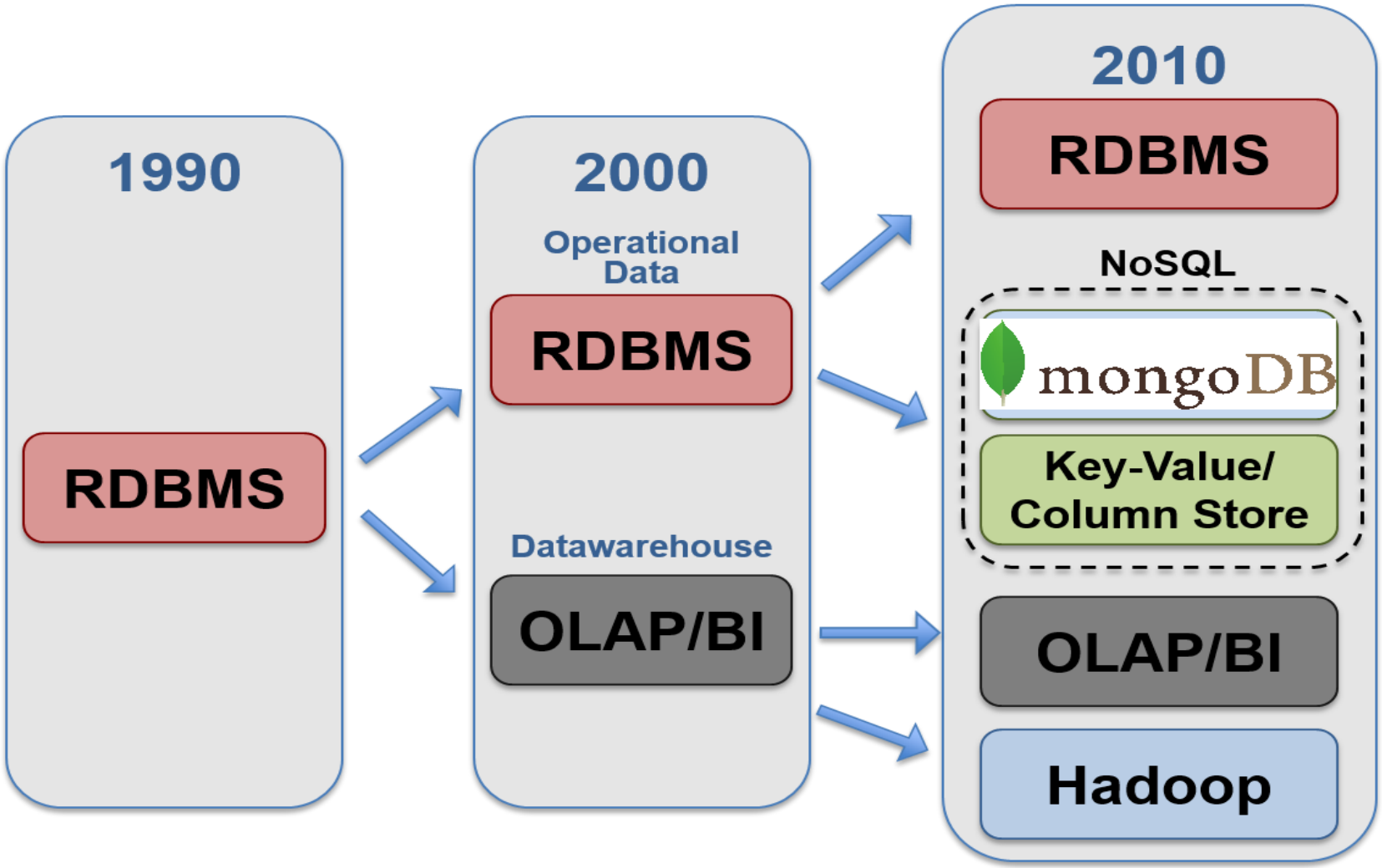
- Different architectures
- Cloud: Transition from Databases to data warehouses and data lakes
- Major Components of HADOOP
- HDFS
- YARN
- SPARK

Cloud: Transition from Databases to data warehouses and data lakes

What Changed?

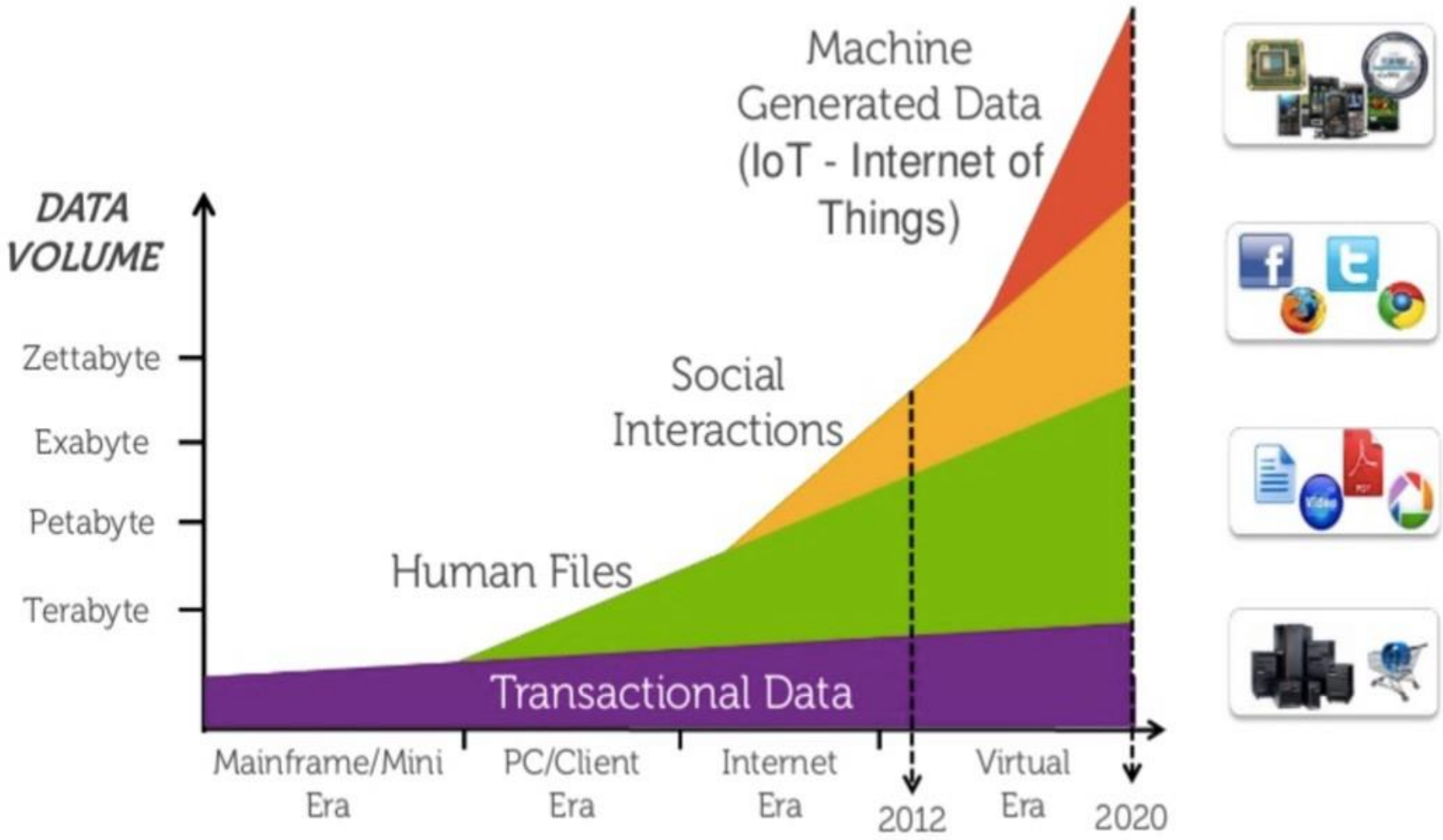


Transition from databases to data warehouses to data lakes



Transition from databases to data warehouses to data lakes

The Explosion of Data



Explosion of Data (Copyrights of Image are Unknown)

Volume: How huge is data?

Multiples of bytes		
SI decimal prefixes		Binary usage
Name (Symbol)	Value	
kilobyte (kB)	10^3	2^{10}
megabyte (MB)	10^6	2^{20}
gigabyte (GB)	10^9	2^{30}
terabyte (TB)	10^{12}	2^{40}
petabyte (PB)	10^{15}	2^{50}
exabyte (EB)	10^{18}	2^{60}
zettabyte (ZB)	10^{21}	2^{70}
yottabyte (YB)	10^{24}	2^{80}

YouTube
>Billion users
72 hours of video per minute
~200 – 350K servers

Microsoft Office 365
~10-20 Million users
Online documents/ppts/excel
~100K servers

CERN
Worlds biggest machine
LHC has 27 km circumference
30 PetaBytes of data in 2012
~100K servers

http://home.cern/about/computing
https://www.youtube.com/yt/press/statistics.html
http://windowsitpro.com/blog/office-365-numbers-ever-increasing-trajectory

“Scale” of data in today’s world

Google processes 3.5 billion requests per day, storing 10 Exabytes of data.

Amazon hosts as many as 1,400,000 servers.

Amazon Web Services (AWS) are used by 60,000 companies and field more than 650,000 requests every second.

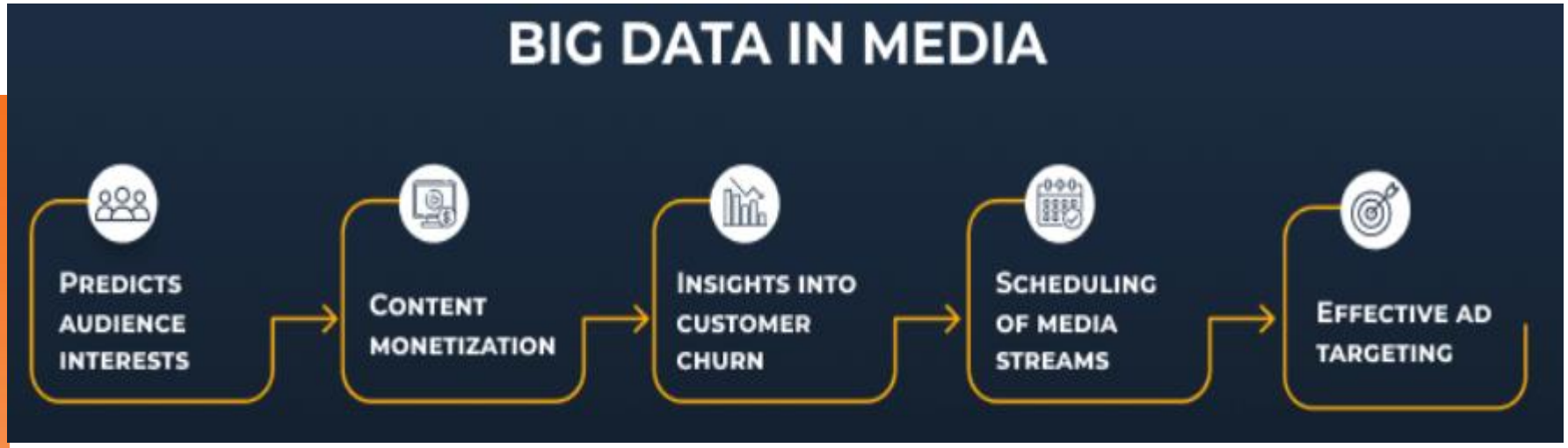
Facebook collects 500 terabytes of data daily, including 2.5 billion pieces of content, 2.7 billion likes and 300 million photos.

90% of all the data in the world was produced in the last 2 years.

It is estimated that 40 zettabytes (40,000 Exabytes) of data will be created by 2020. 3.2 zettabytes [in 2014]

The total amount of data being captured and stored by industry doubles every 1.2 years

Big data usage in different industries



Big data usage in different industries

Recent Announcements

Home / Cities / Hyderabad / Microsoft's largest data centre in India to come up in Hyderabad

Microsoft's largest data centre in India to come up in Hyderabad

Over the next 15 years, the multinational company will invest Rs 15,000 crore into the new data centre region in Hyderabad, spread across three sites – Chandanvelly, Ellikatta, and Kottur.

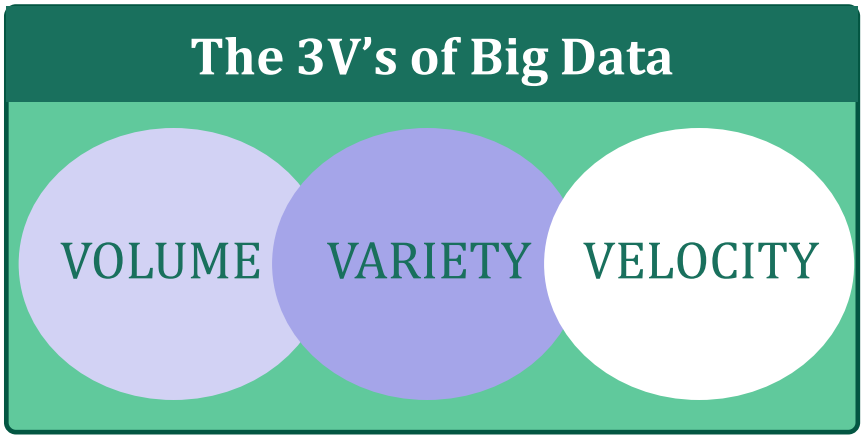
By: **Express News Service** | Hyderabad |
Updated: March 8, 2022 10:31:23 am

ADVERTISEMENT

What is Big Data ?

Big does not refer to size or volume alone !

So what else comes into



The V's of Big Data

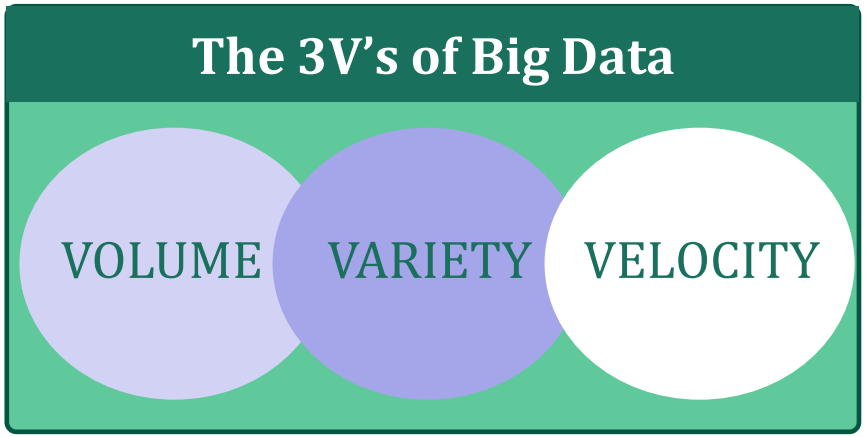
Volume, Velocity

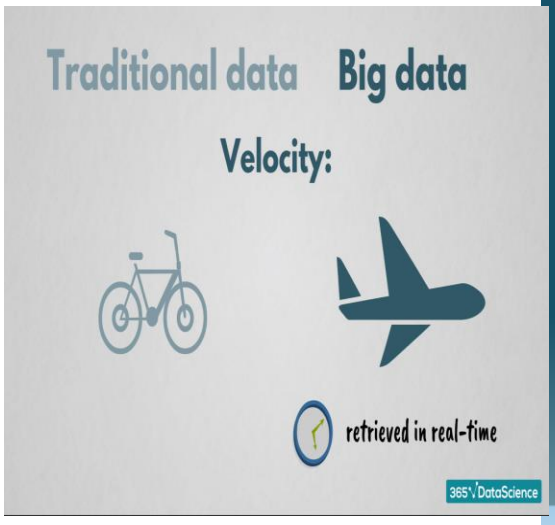
Speed at which data is generated

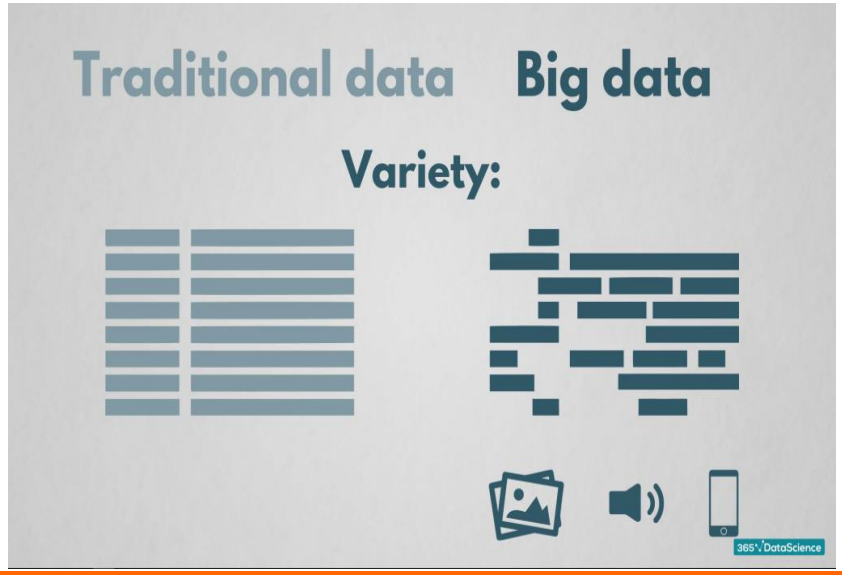
Speed at which it has to be analysed for actionable insights

chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://www.irjet.net/archives/V4/i9/IRJET-V4I957.pdf

The V's of Big Data







New Types of Data

Sentiment
Understand how your customers feel about your brand and products – right now

Clickstream
Capture and analyze website visitors' data trails and optimize your website

Sensors
Discover patterns in data streaming automatically from remote sensors and machines

Geographic
Analyze location-based data to manage operations where they occur

Server Logs
Research logs to diagnose process failures and prevent security breaches

Unstructured
Understand patterns in files across millions of web pages, emails, and documents






Gartner, Big Data Definition

"Big data is high-volume, high-velocity and/or high-variety information assets that demand cost-effective, innovative forms of information processing that enable enhanced insight, decision making, and process automation."

Volume, Velocity, Variety, Veracity

Big Data Use-case at CERN

https://www.youtube.com/watch?v=j-0cUmUyb-Y

Physicists at CERN have been pondering how to store and share their ever more massive data for decades - stimulating globalization of the internet along the way, whilst 'solving' their big data problem.

Five Big Data Use Case Categories & Many Use Cases

Big Data Exploration
Find, visualize, understand all big data to improve decision making

Enhanced 360° View of the Customer
Extend existing customer views (MDM, CRM, etc) by incorporating additional internal and external information sources

Security/Intelligence Extension
Lower risk, detect fraud and monitor cyber security in real-time

Operations Analysis
Analyze a variety of machine data for improved business results

Data Warehouse Augmentation
Integrate big data and data warehouse capabilities to increase operational efficiency

1. Expertise Location
2. Recommendation
3. Commerce
4. Financial Analysis
5. Social Media Monitoring
6. Telco Customer Analysis
7. Healthcare Analysis
8. Data Exploration and Visualization
9. Personalized Search
10. Anomaly Detection
11. Fraud Detection
12. Cybersecurity
13. Sensor Monitoring (Smarter another Planet)
14. Cellular Network Monitoring
15. Cloud Monitoring
16. Code Life Cycle Management
17. Traffic Navigation
18. Image and Video Semantic Understanding
19. Genomic Medicine
20. Brain Network Analysis
21. Data Curation
22. Near Earth Object Analysis

CY Lin, Columbia University, 2017

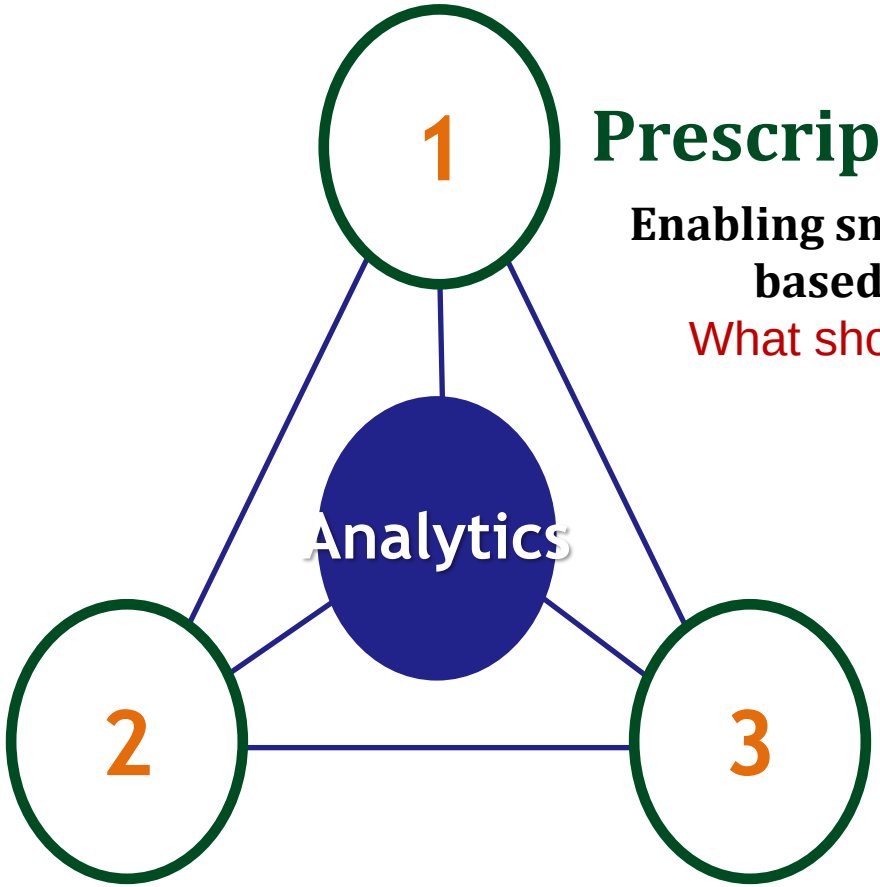
What is Analytics?

Data on its own is useless unless you can make sense of it!

WHAT IS ANALYTICS?

The scientific process of transforming data into insight for making better decisions, offering new opportunities for a competitive advantage

Types of Analytics



Prescriptive Analytics

Enabling smart decisions based on data

What should we do?

Predictive analytics

Predicting the future based on historical patterns

What could happen?

Descriptive analytics

Mining data to provide business insights

What has happened?

Types of Analytics

Why do airline prices change every hour?

Prescriptive Analytics
advice on possible outcomes

How do grocery cashiers know to hand you coupons you might actually use?

Predictive Analytics
understanding the future

How does Netflix frequently recommend just the right movie?

Descriptive Analytics
insight into the past

Problems with Traditional Approach

The world of Hadoop

Doug Cutting Basics of Hadoop Video

https://www.youtube.com/watch?v=0GOxDBR6VAU

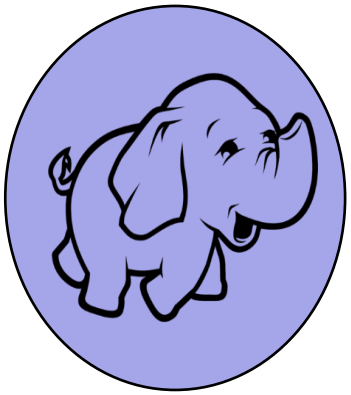
What is Hadoop?

Hadoop is an open source framework, from the Apache foundation, capable of processing large amounts of heterogeneous data sets in a distributed fashion across clusters of commodity computers and hardware using a simplified programming model.

The Hadoop framework is based closely on the following principle:

In pioneer days they used oxen for heavy pulling, and when one ox couldn't budge a log, they didn't try to grow a larger ox. We shouldn't be trying for bigger computers, but for more systems of computers.

What is Hadoop? (contd.)



Hadoop is Transforming Businesses Across Industries

Organizations use Hadoop to manage their data today (up from 1 out of 10 in 2012)

BIG DATA STORING AND FASTER PROCESSING

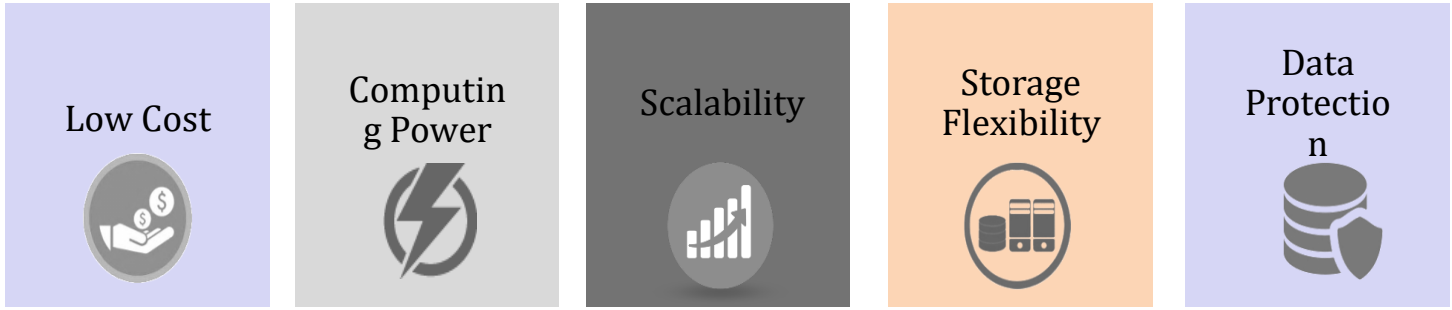
Hadoop is an open source software framework created in 2005 that keeps and processes big data in a distributed manner on large collection of hardware.

BUSINESS SOLUTIONS ACROSS DOMAINS AND INDUSTRIES:

Low cost solution with a high fault tolerance to access and create value from data.

Why Hadoop?

Top 5 Reasons Organizations are using Hadoop

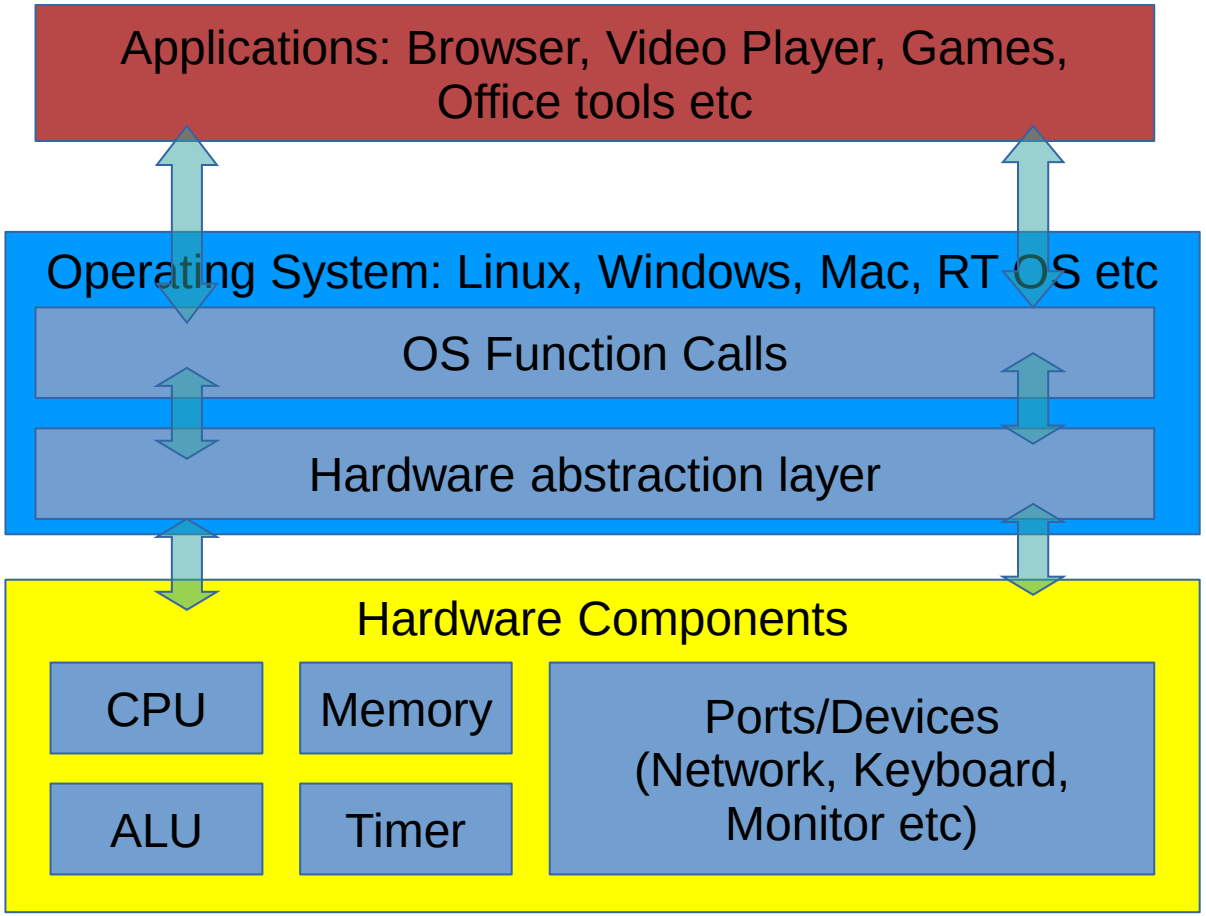


Enterprises using Hadoop

Top 5 Industries using Hadoop:

- Computer Manufacturing
- Business Services
- Finance
- Retail & Wholesale
- Education & Government

Big Data Tools: Why, really, Why???



Typical Operating System View

Role of an OS:

1) Application does not have to worry about hardware details. Ex: Different types or processors, Keyboards, Mice, or Graphics cards
2) Failure of hardware is handled gracefully, Ex: If the Graphics card fails or keyboard is not connected, give some beeps
3) Handle multiple applications

Role of a **Big Data** OS; All of the above plus:

1) Handle massively large amounts of data and applications
2) Keep backups
3) Provide system health
4) Support OS function calls that are err... ***complicated***

Distributed File Systems: GFS was a new breed

NFS, etc.

Independence
Small Scale
Variety of workloads

GFS

Cooperation
Large scale
Very specific, well-understood workloads

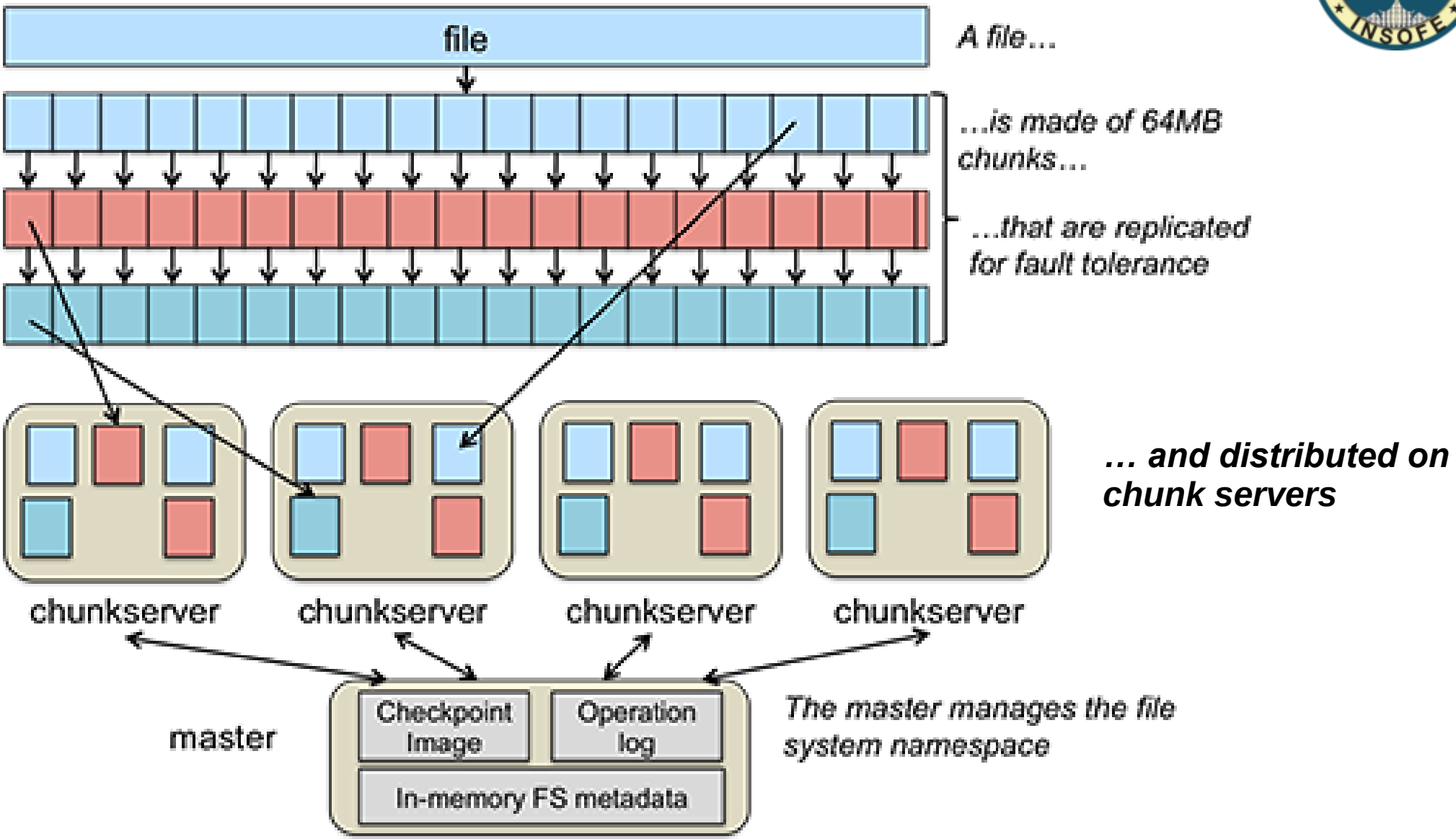
The Google File System

Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung

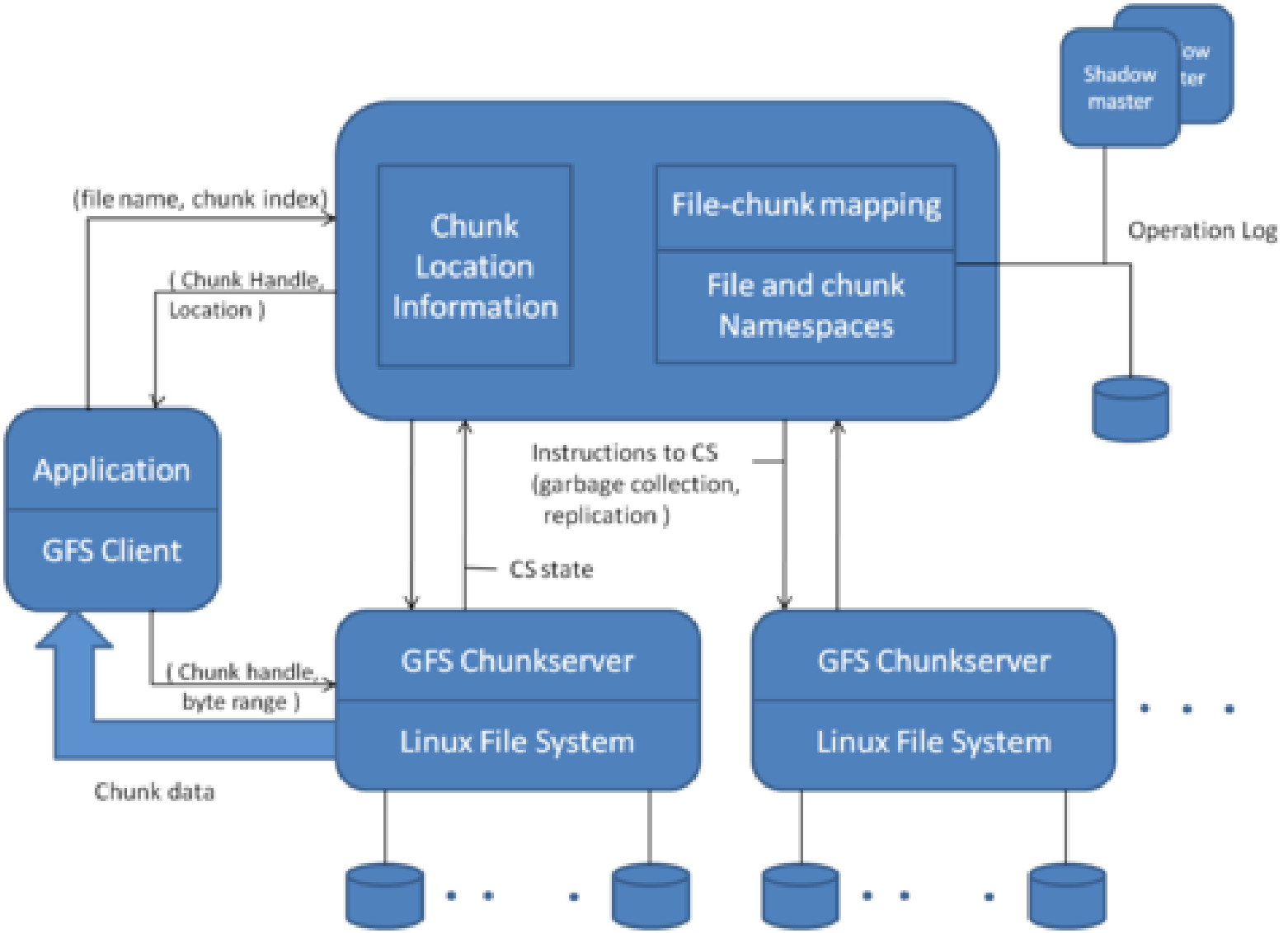
Google*

Sanjay Ghemawat 2003

GFS


file
A file…
…is made of 64MB chunks…
…that are replicated for fault tolerance
chunkserver
chunkserver
chunkserver
chunkserver
… and distributed on chunk servers
master
Checkpoint Image
Operation log
In-memory FS metadata
The master manages the file system namespace

GFS: Architecture



HDFS and MapReduce

Distributed Storage

HDFS - Reliable Shared Storage

+

MapReduce - Distributed Computation

Parallel processing

=

Hadoop: 30,000 feet view (contd.)

Distribute data initially

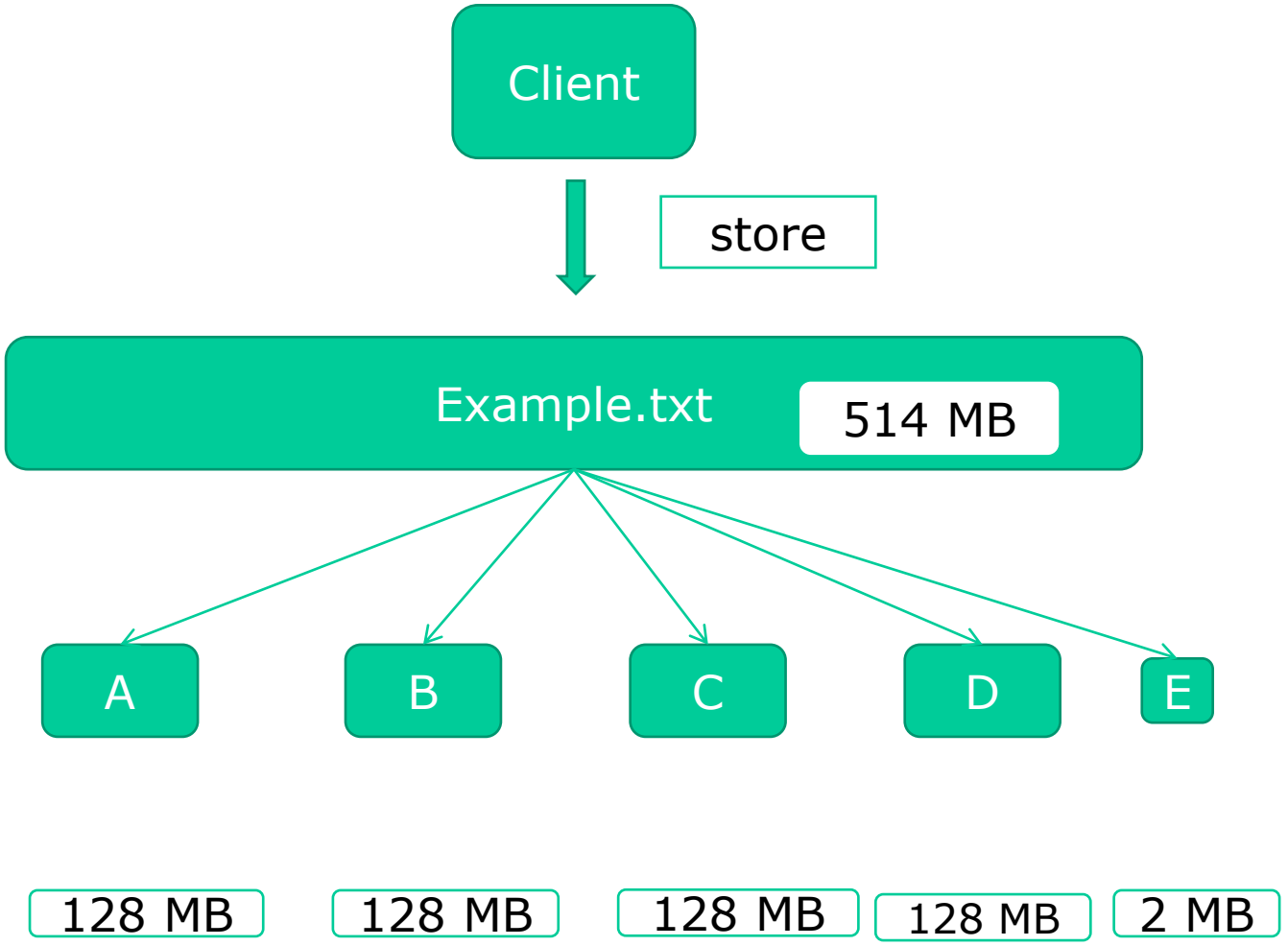
- Let processors / nodes work on local data
- Minimize data transfer over network
- Replicate data multiple times for increased availability

Write applications at a high level

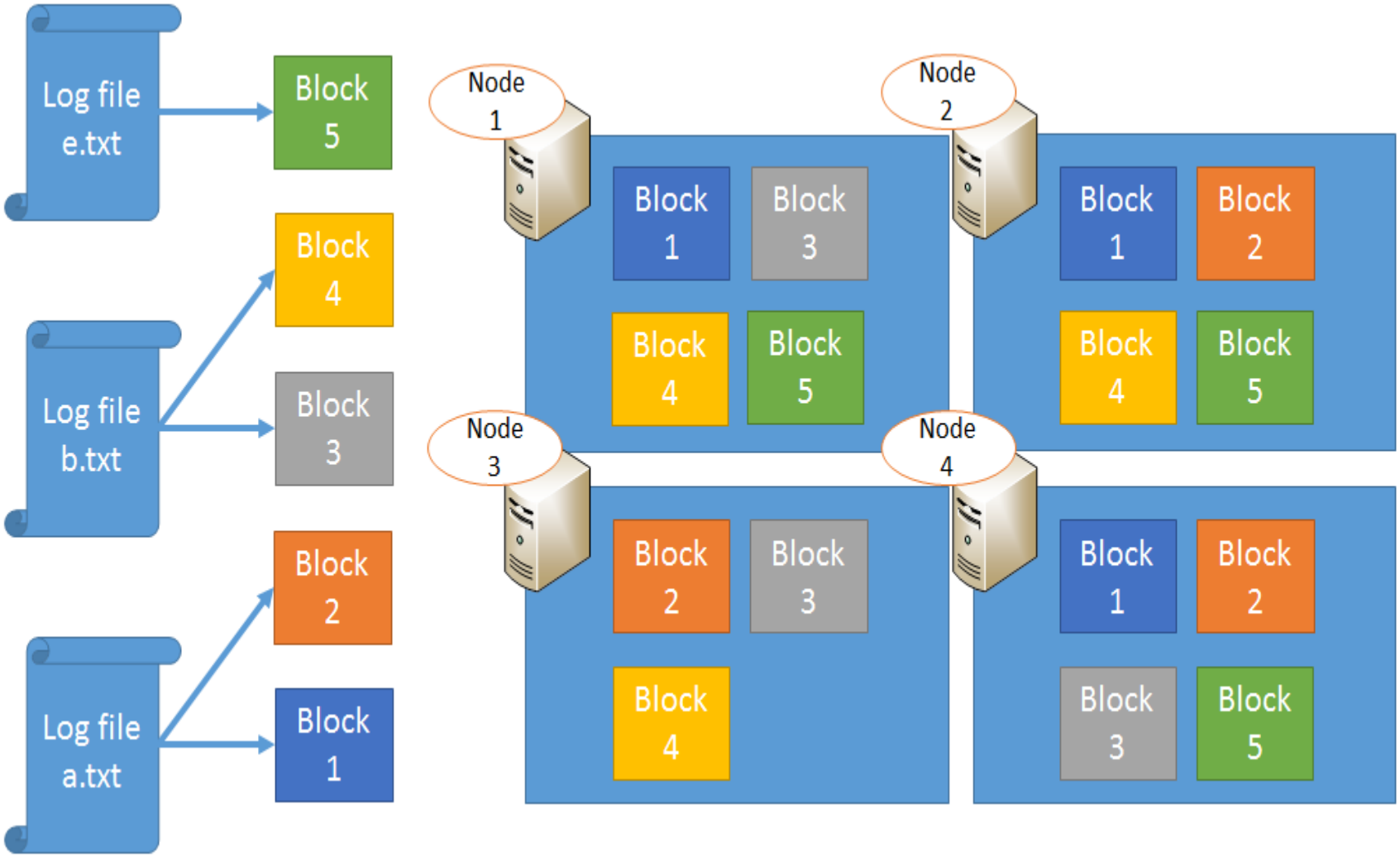
- Programmers should not have to worry about network programming, temporal dependencies, low level infrastructure, etc

Minimize talking between nodes (*share-nothing)*

Blocks?

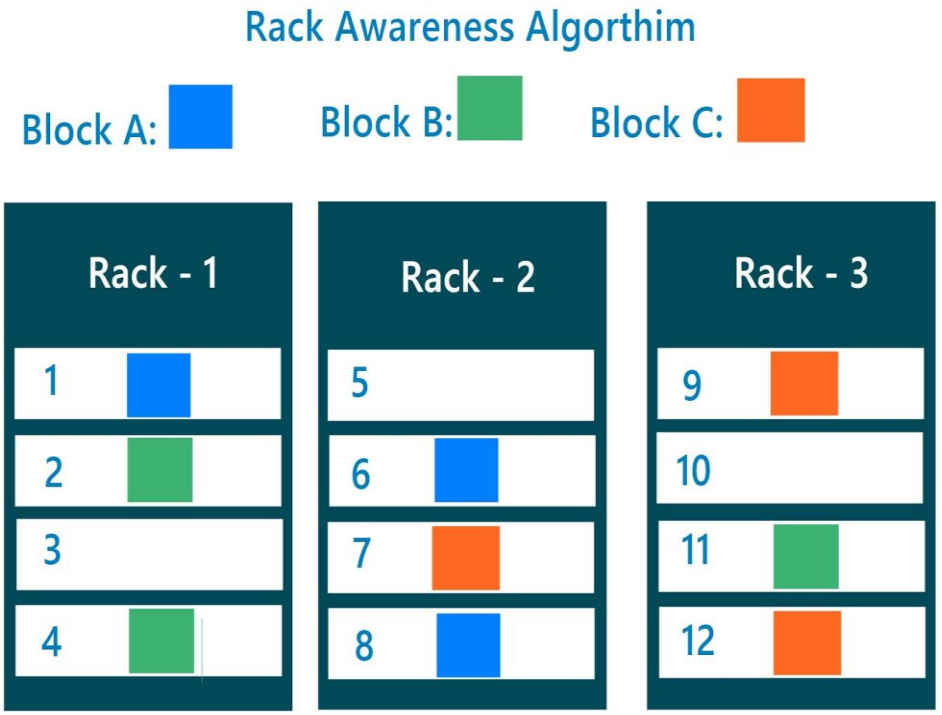

Client
store
Example.txt
514 MB
A
B
C
D
E
128 MB
128 MB
128 MB
128 MB
2 MB

Replication Management:

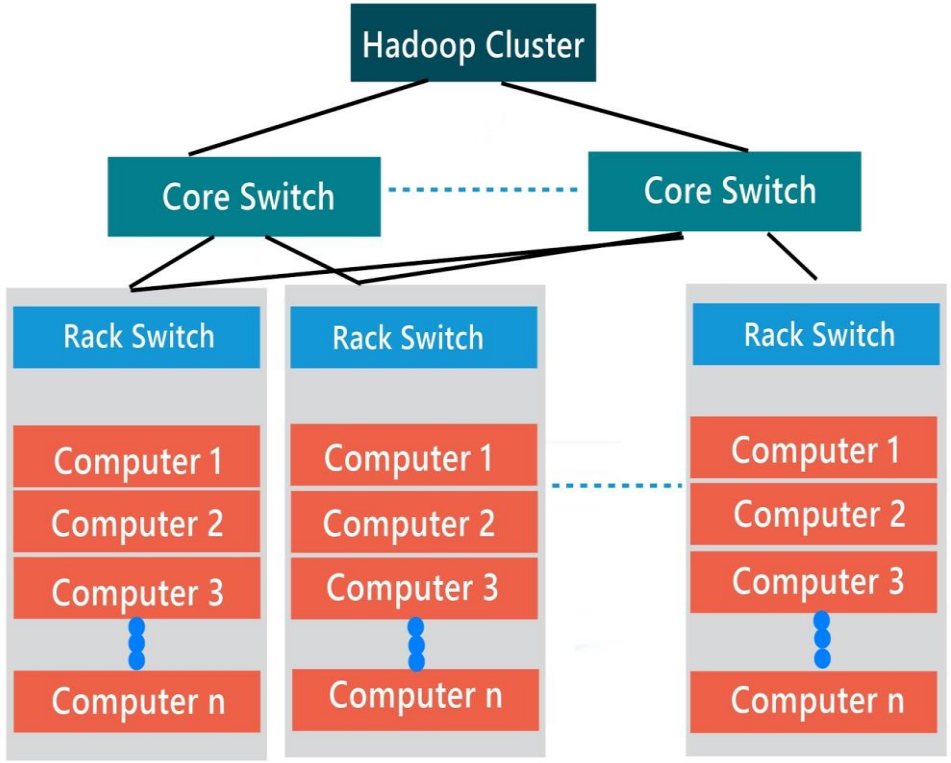


The default replication factor is 3 which is again configurable.

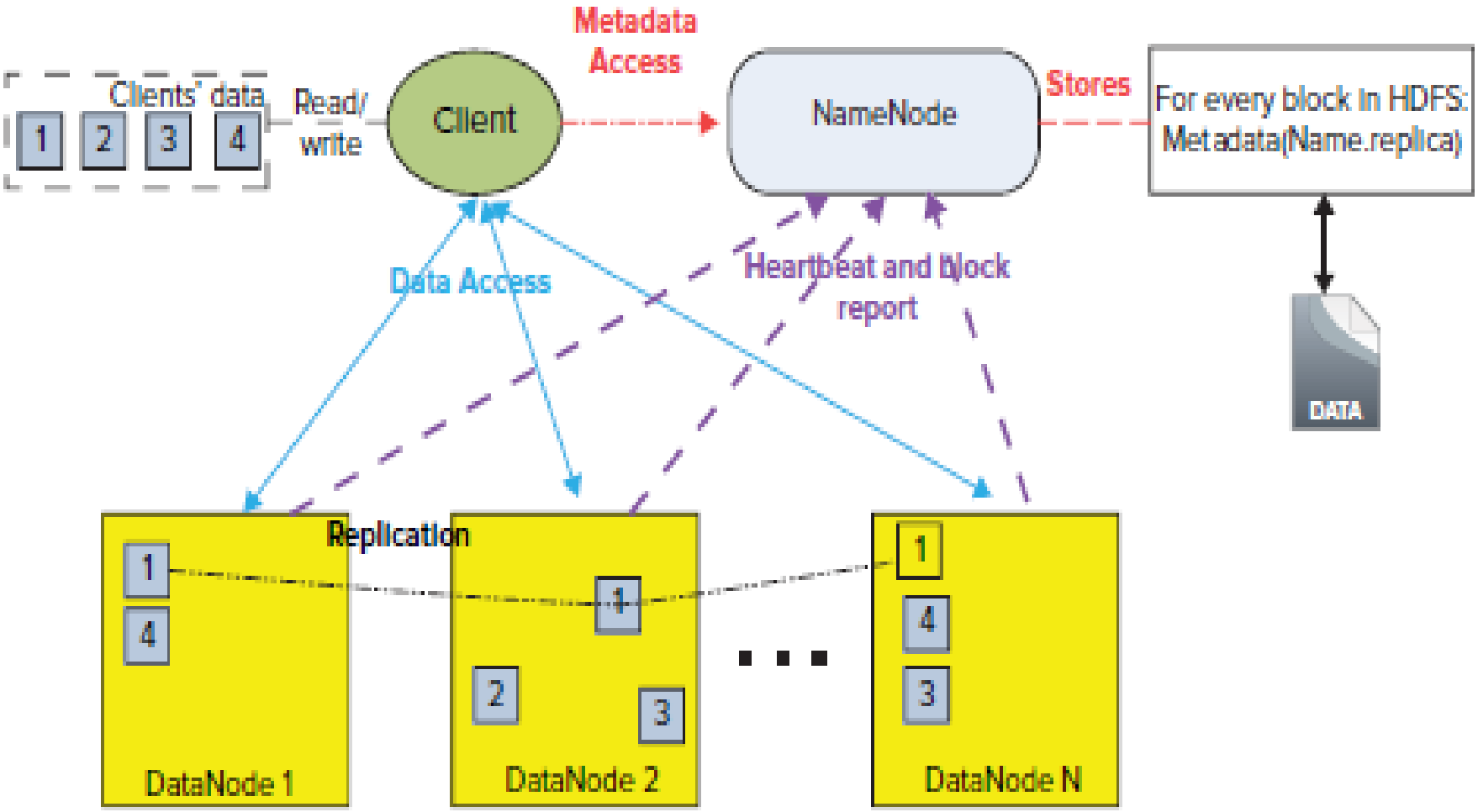
Rack Awareness: in-built Rack Awareness Algorithm



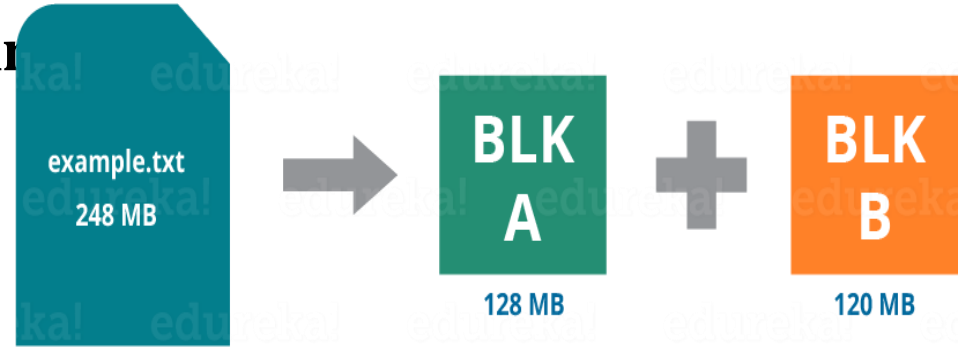
- To improve the network performance
- To prevent loss of data



Hadoop Architecture:-



HDFS Write Architectur



At first, the HDFS client will reach out to the NameNode for a Write Request against the two blocks, say, Block A & Block B.

The NameNode will then grant the client the write permission and will provide the IP addresses of the DataNodes where the file blocks will be copied eventually.

For Block A, list A = **{IP of DataNode 1, IP of DataNode 4, IP of DataNode 6}**

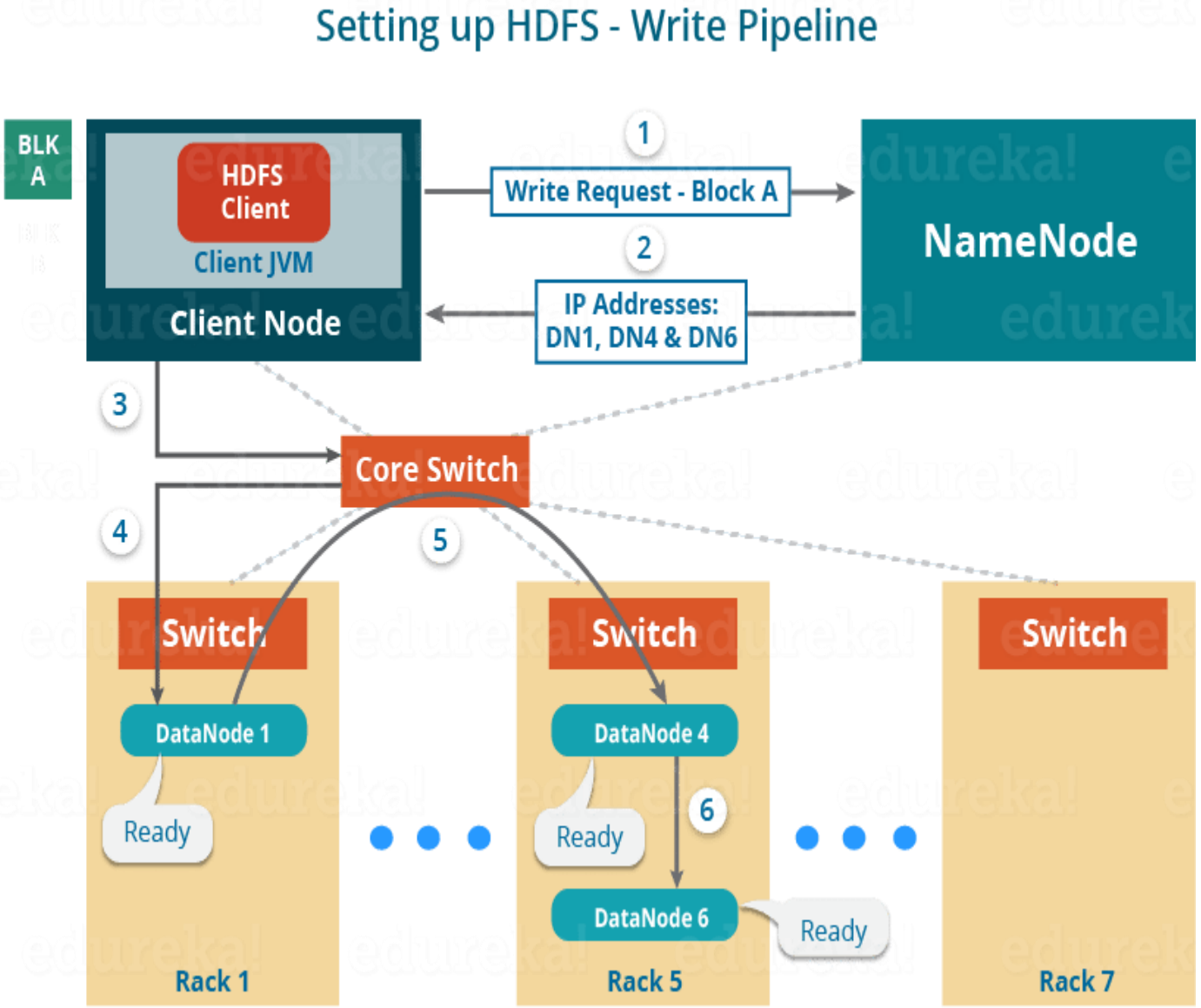
For Block B, set B = **{IP of DataNode 3, IP of DataNode 7, IP of DataNode 9}**

Set up of Pipeline

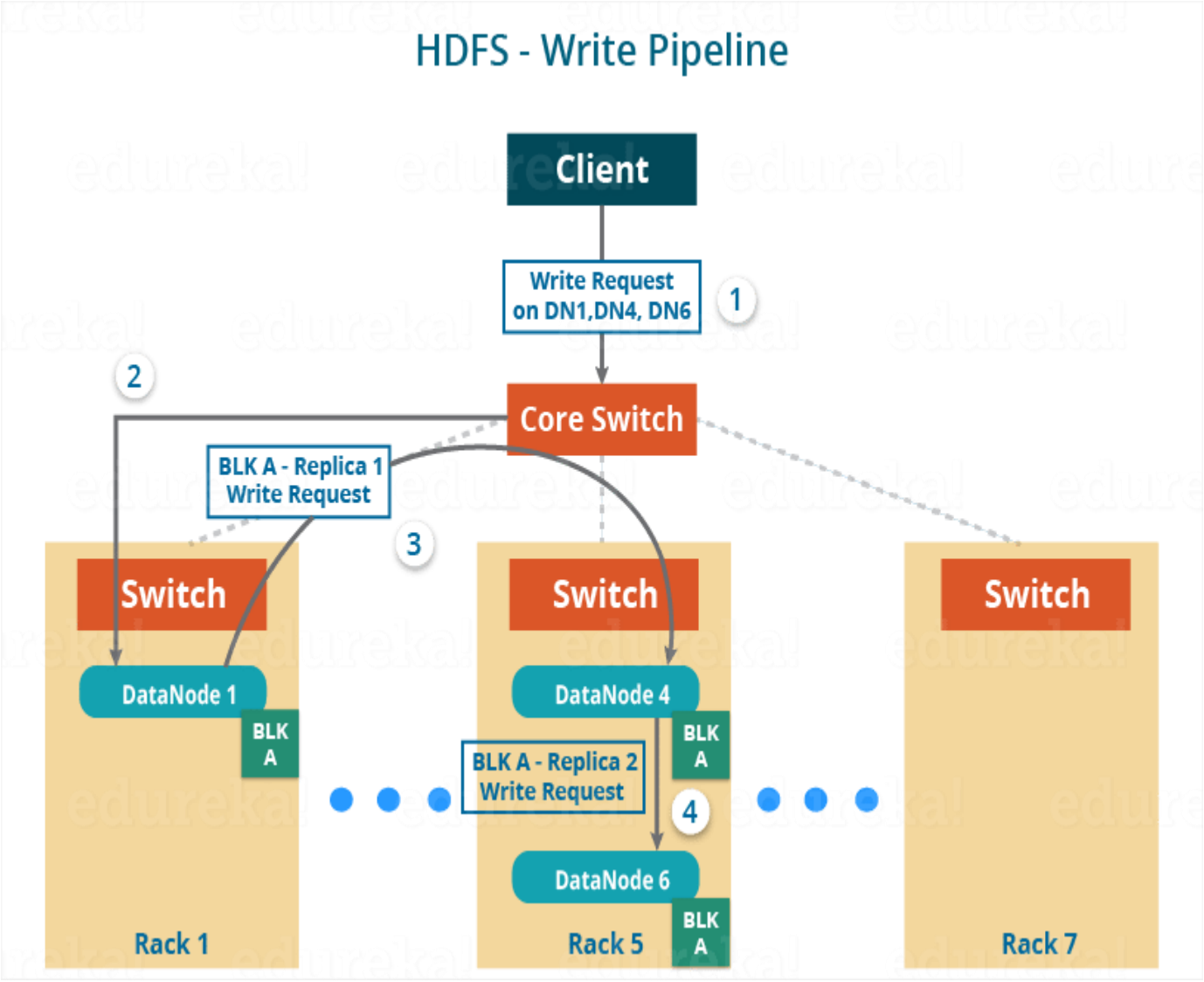
Data streaming and replication

Shutdown of Pipeline (Acknowledgement stage)

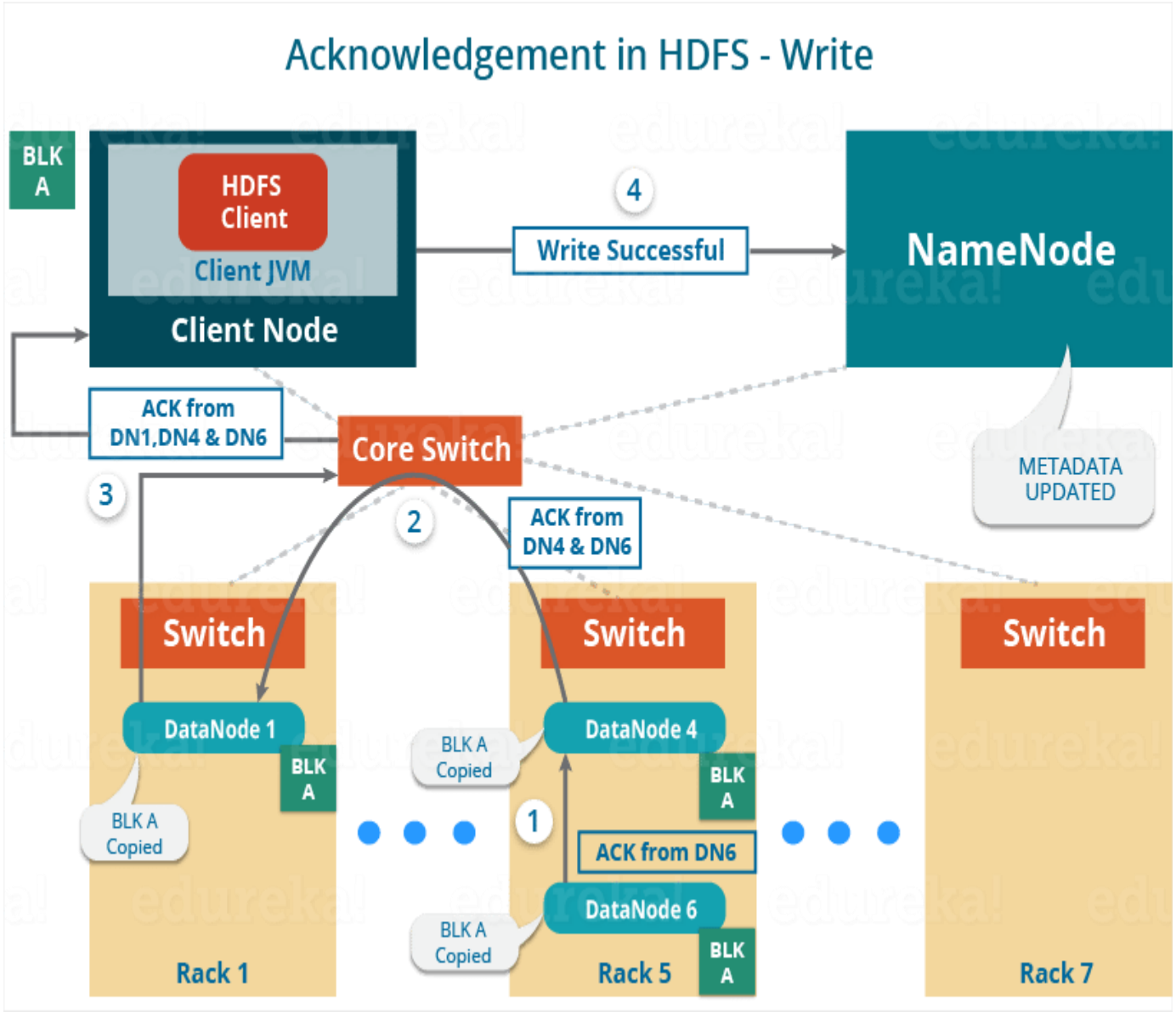
Set up of Pipeline



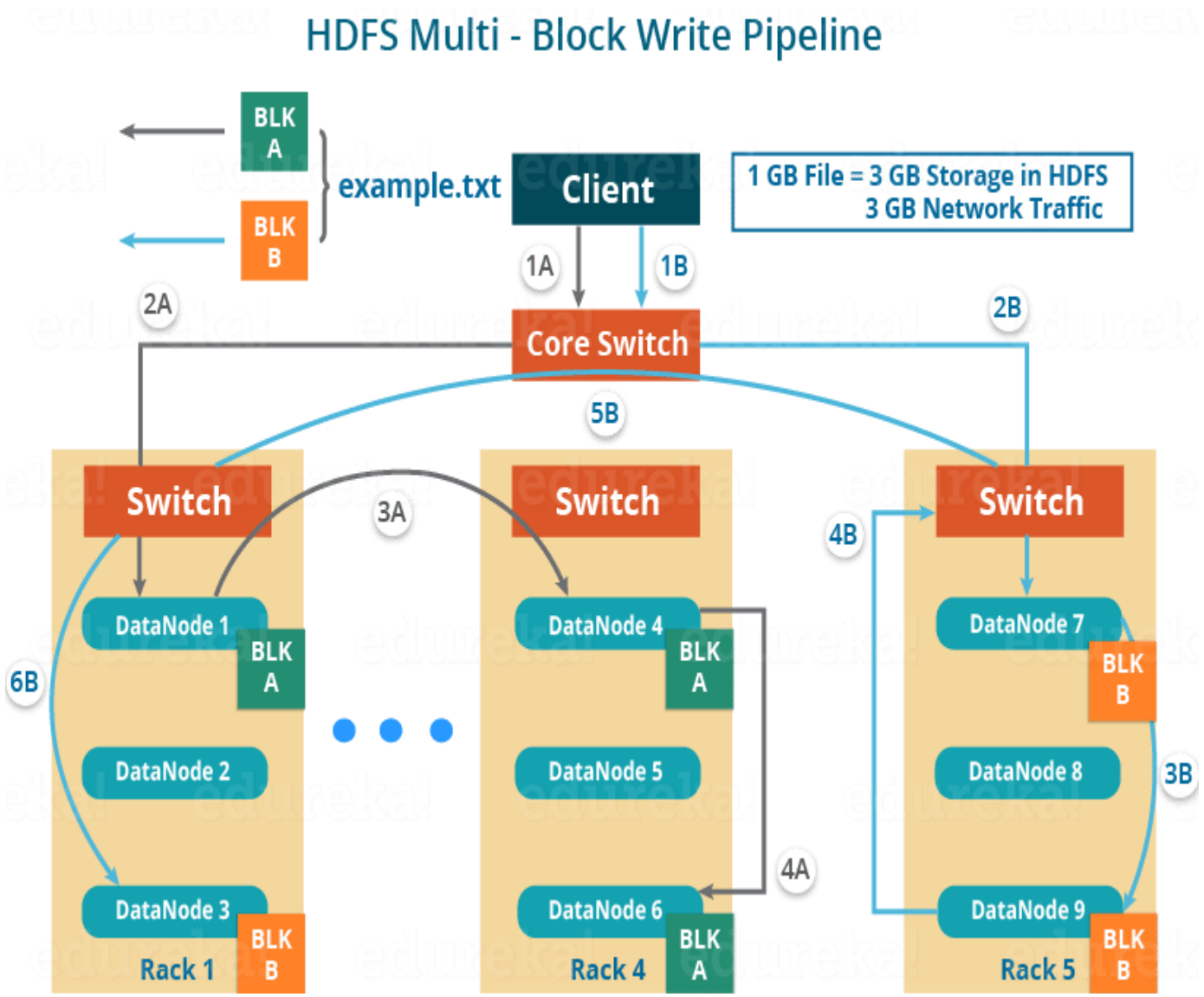
Data streaming and replication



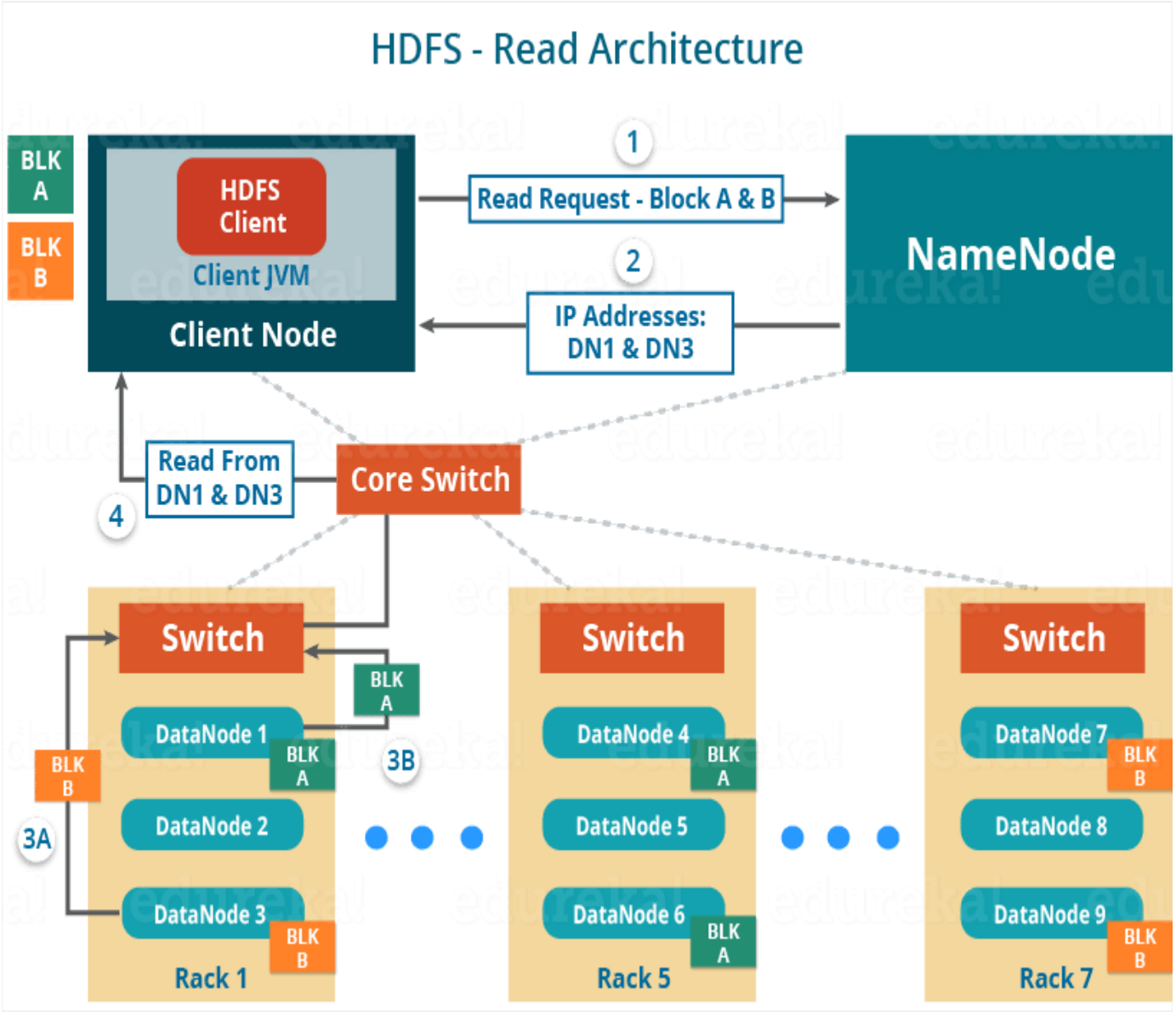
Shutdown of Pipeline (Acknowledgement stage)



HDFS – Multi Write Request



HDFS Read Architecture:



Why Hadoop?

Scalability: Need to process Multi Petabyte Datasets, quickly.

Data may not have strict schema

Expensive to build reliability in each application

Fault tolerant: Nodes fails everyday

Need common infrastructure

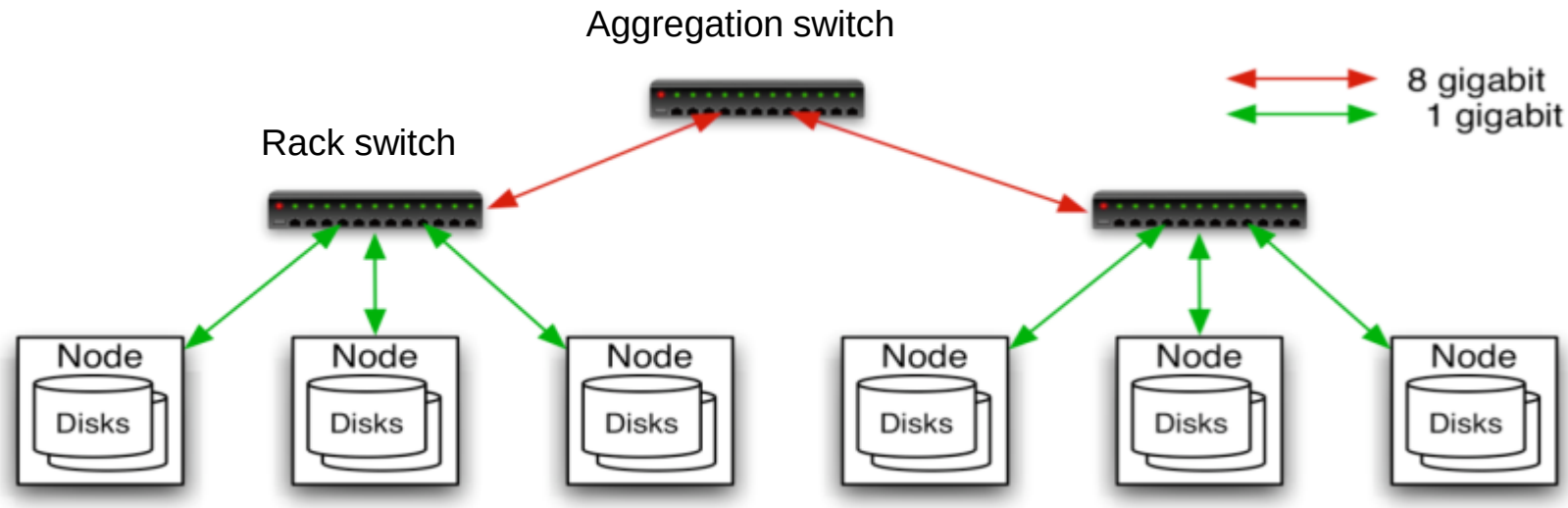
Very Large Distributed File System

Low cost: Assumes Commodity Hardware

Optimized for Batch Processing

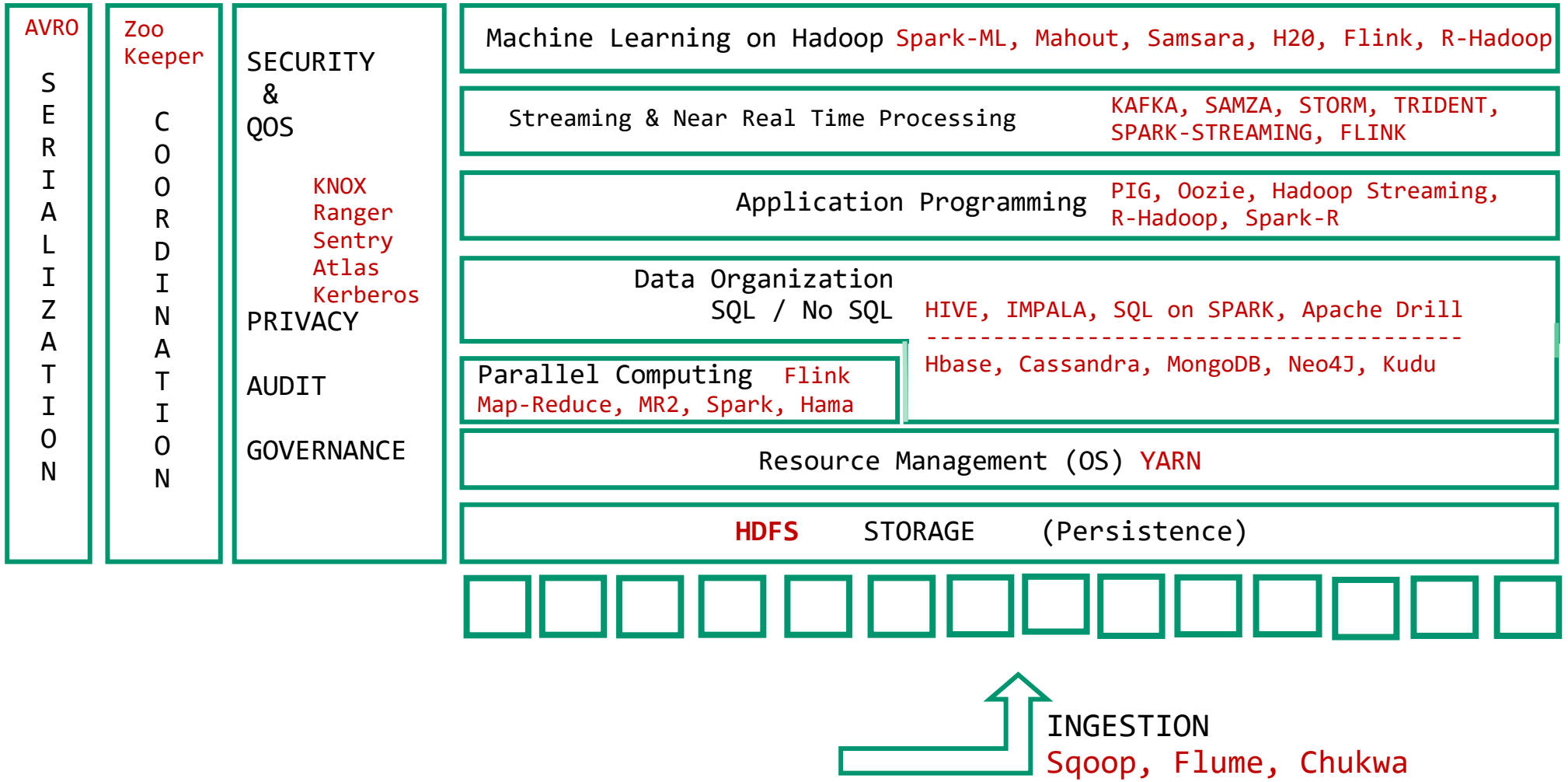
Runs on heterogeneous OS

Hub & Spoke Hardware

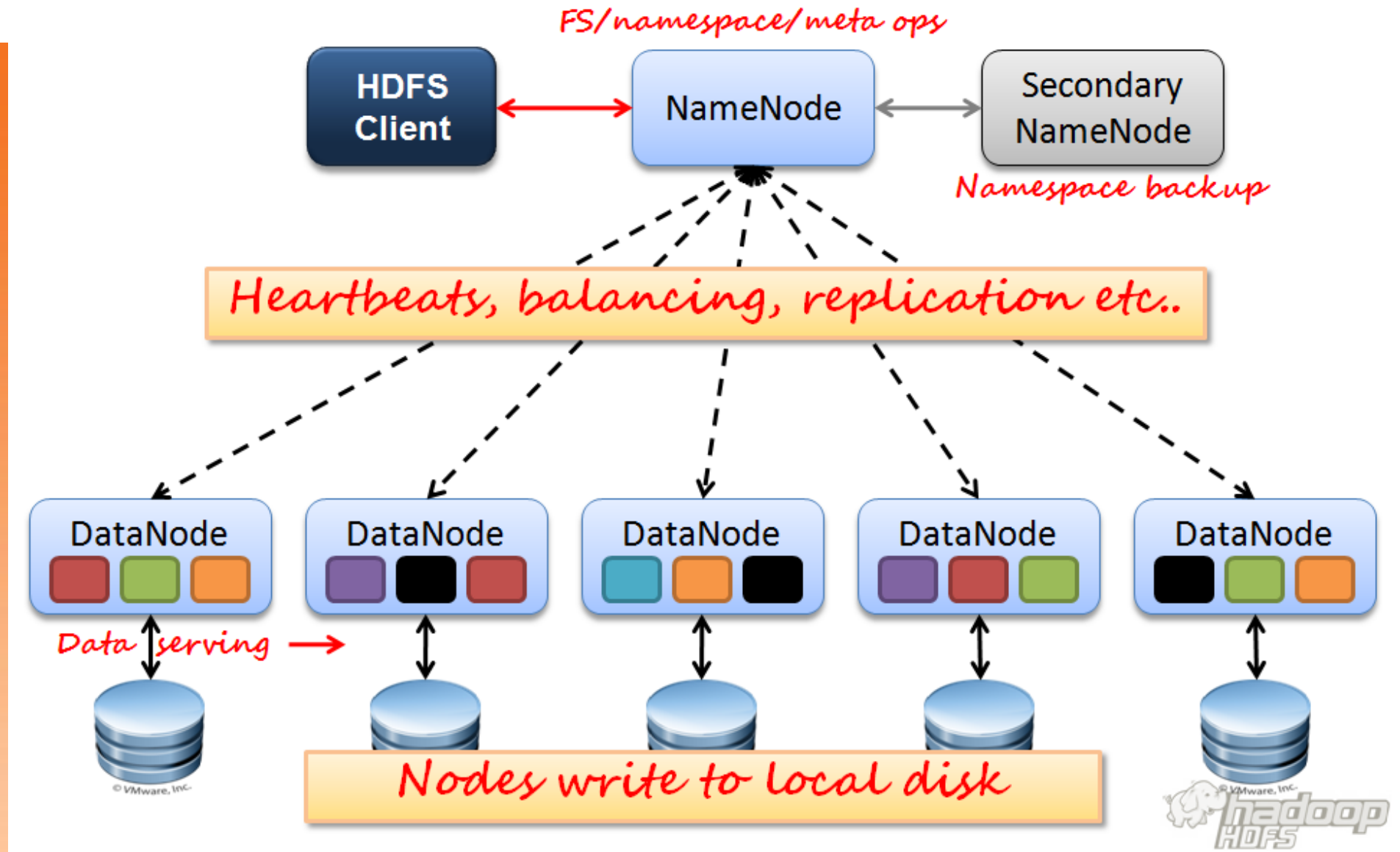


Typically in 2 level architecture

- Nodes are commodity PCs
- 30-40 nodes/rack
- Uplink from rack is 3-4 gigabit
- Rack-internal is 1 gigabit


AVRO
SERIALIZATION
Zoo Keeper
COORDINATION
SECURITY
&
QOS
KNOX
Ranger
Sentry
Atlas
Kerberos
PRIVACY
AUDIT
GOVERNANCE
Machine Learning on Hadoop Spark-ML, Mahout, Samsara, H20, Flink, R-Hadoop
Streaming & Near Real Time Processing
KAFKA, SAMZA, STORM, TRIDENT, SPARK-STREAMING, FLINK
Application Programming
PIG, Oozie, Hadoop Streaming, R-Hadoop, Spark-R
Data Organization
SQL / No SQL
HIVE, IMPALA, SQL on SPARK, Apache Drill
Hbase, Cassandra, MongoDB, Neo4J, Kudu
Parallel Computing Flink
Map-Reduce, MR2, Spark, Hama
Resource Management (OS) YARN
HDFS STORAGE (Persistence)
INGESTION
Sqoop, Flume, Chukwa

HDFS CDH3: Open-source reimplementation of GFS



HDFS Components

NameNode manages overall file system metadata

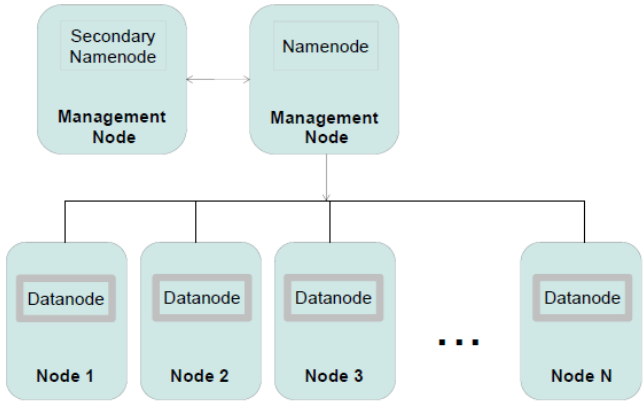
- Manages the file systems namespace
- Map from file name to where data is stored, like other file systems
- Can be a single point of failure in the system

DataNodes (one per machine) store and retrieves actual data

- DataNodes are easy to add, expanding the file system
- Each datanode reports to namenode.
- Acts as a block server: Stores data as blocks and metadata as block checksum. Serves data and metadata to clients.

Both DataNode and NameNodes include a webserver, so node status can be easily checked

Secondary namenode: does housekeeping (checkpointing, logging)



Replication and Reliability

Namenode is “rack aware”: knows how machines are arranged

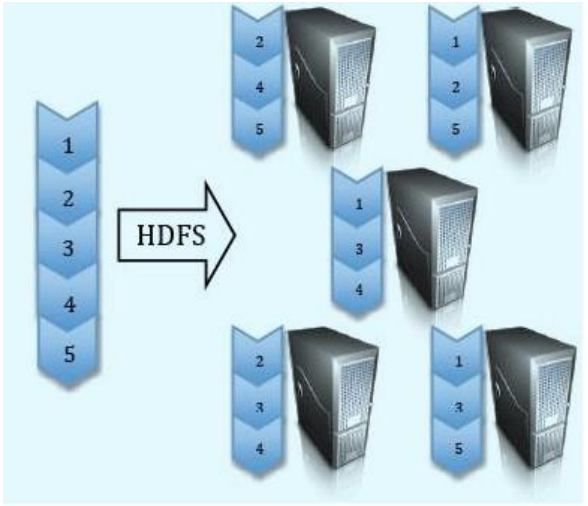
- Second replica is on same rack as the first, but different machine
- Third replica is on a different rack
- Additional replicas are randomly placed
- Balances performance (failover time) vs. reliability (independence)

Namenode does **not** directly read/write data

- Client gets data location from namenode
- Namenode ensures that clients read from nearest replicas
- Client interacts directly with datanode to read/write data

Namenode keeps all block metadata in (fast) memory

- Puts constraint on number of files stored: millions of large files



Metadata storage

- HDFS stores file system metadata and application data separately.
- **Metadata** refers to file metadata (attributes such as list of files, permissions, modification, access times, namespace and disk space quotas), list of blocks that belong to the file, list of datanodes for each block.
- Metadata is always stored in main memory.
- HDFS stores metadata on a dedicated server, called the NameNode. (Master) Application data are stored on other servers called DataNodes. (Slaves)
- All servers are fully connected and communicate with each other using TCP-based protocols. (RPC=Remote Procedure Calls)

Responsibilities of a Namenode

- Maintain the namespace tree (a hierarchy of files and directories) operations like opening, closing, and renaming files and directories.
- Determine the mapping of file blocks to DataNodes (the physical location of file data).
- File metadata (i.e. “inode”) .
- Authorization and authentication.
- Collect block reports from Datanodes on block locations.
- Replicate missing blocks.
- HDFS keeps the entire namespace in RAM, allowing fast access to the metadata.
- % disk full on Data Nodes should be similar

Responsibilities of a DataNode

- The DataNodes are responsible for serving read and write requests from the file system's clients.
- The DataNodes also perform block creation, deletion, and replication upon instruction from the NameNode.
- Data nodes periodically send block reports to Namenode.

GFS vs HDFS

GFS	HDFS
Master	Name Node
Chunk Server	Data Node
Shadow master	Secondary name node
Chunk	Block
Allows append	Does not allow append
C++	Java

What is the rationale behind this?
Think about the primary workload on Yahoo or Google:

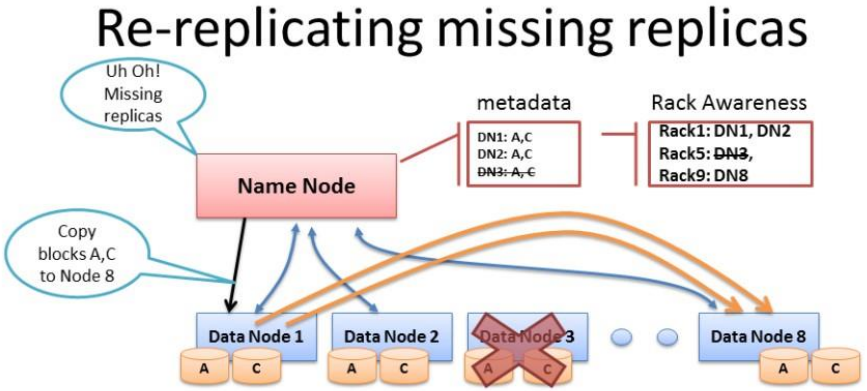
- Search (Crawling/Indexing/Perform search on index)
- Video and photo uploads (Upload/Add comments/Display some videos and photos)

Most of these need:

- Write once, rarely append or modify, random reads that need to be very fast

Failure Recovery

- The NameNode does not directly call DataNodes. It uses replies to heartbeats to send instructions to the DataNodes. The instructions include commands to:
  - Replicate blocks to other nodes:
    - ➢ DataNode died.
    - ➢ copy data to local.
  - Remove local block replicas
  - Re-register or to shut down the node
- So when dataNode died, NameNode will notice and instruct other dataNode to replicate data to new dataNode. What if NameNode died?



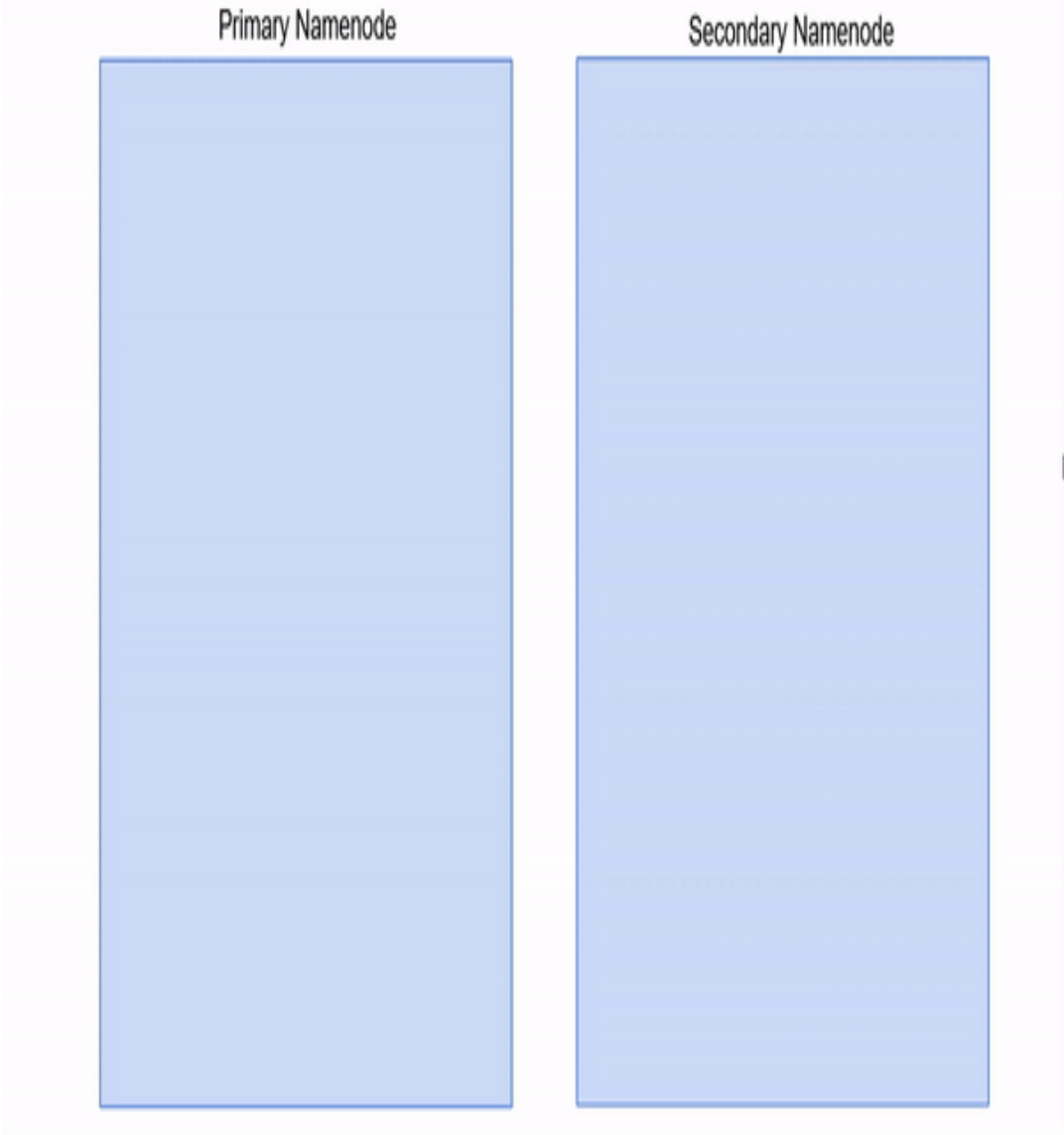
- Missing Heartbeats signify lost Nodes
- Name Node consults metadata, finds affected data
- Name Node consults Rack Awareness script
- Name Node tells a Data Node to re-replicate

Secondary Namenode in HDFS

Secondary Namenode is another node present in the cluster whose main task is to regularly merge the Edit log with the Fsimage and produce check-points of the primary's in-memory file system metadata. This is also referred to as Checkpointing.

But the checkpointing procedure is computationally very expensive and requires a lot of memory, which is why the Secondary namenode runs on a separate node on the cluster.

However, despite its name, the Secondary Namenode does not act as a Namenode. It is merely there for Checkpointing and keeping a copy of the latest Fsimage.



Issues with one namenode

- **Tight coupling of Block Storage and Namespace:** This makes alternate implementations of namenodes challenging and limits other services from using the block storage ***directly***.
- **Namespace scalability:** While HDFS cluster storage scales horizontally with the addition of datanodes, the namespace does not. Currently the namespace can only be vertically scaled on a single namenode. The namenode stores the entire file system metadata in memory. This limits the number of blocks, files, and directories supported on the file system to what can be accommodated in the memory of a single namenode.
- **Performance:** File system operations are limited to the throughput of a single namenode, which currently supports 60K tasks.
- **Isolation:** For many deployments, the cluster is used in a multi-tenant environment where many organizations share the cluster. A single namenode offers no isolation in this setup. A separate namespace for a tenant is not possible. An experimental application that overloads the namenode can slow down the other production applications. A single namenode also does not allow segregating different categories of applications (such as HBase) to separate namenodes.

MapReduce

MapReduce is a programming model for processing large data sets with a parallel and distributed algorithm on a cluster (source: Wikipedia).

Map Reduce when coupled with HDFS can be used to handle big data. The fundamentals of this HDFS-MapReduce system, which is commonly referred to as Hadoop.

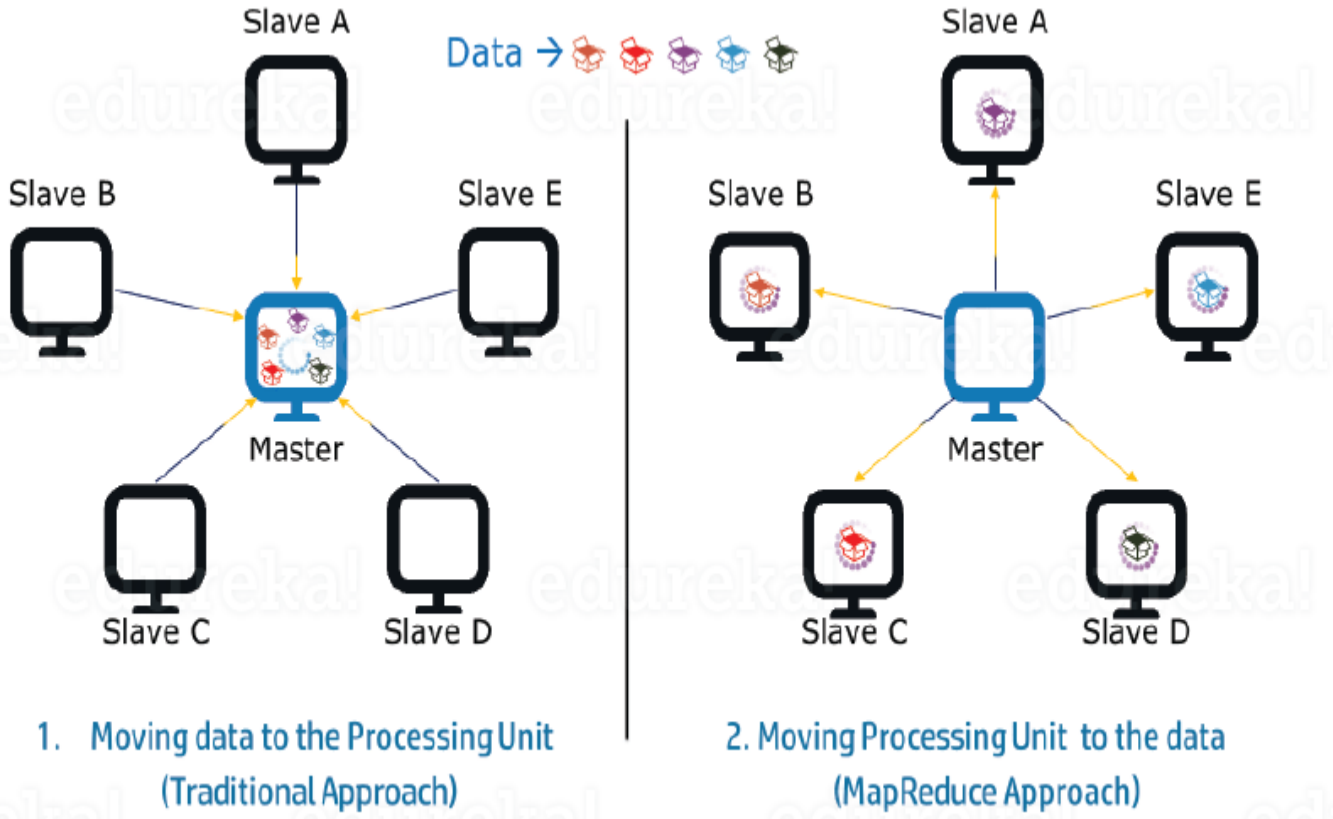
The basic unit of information, used in MapReduce is a (Key,value) pair. All types of structured and unstructured data need to be translated to this basic unit, before feeding the data to MapReduce model.

As the name suggests, MapReduce model consist of two separate routines, namely

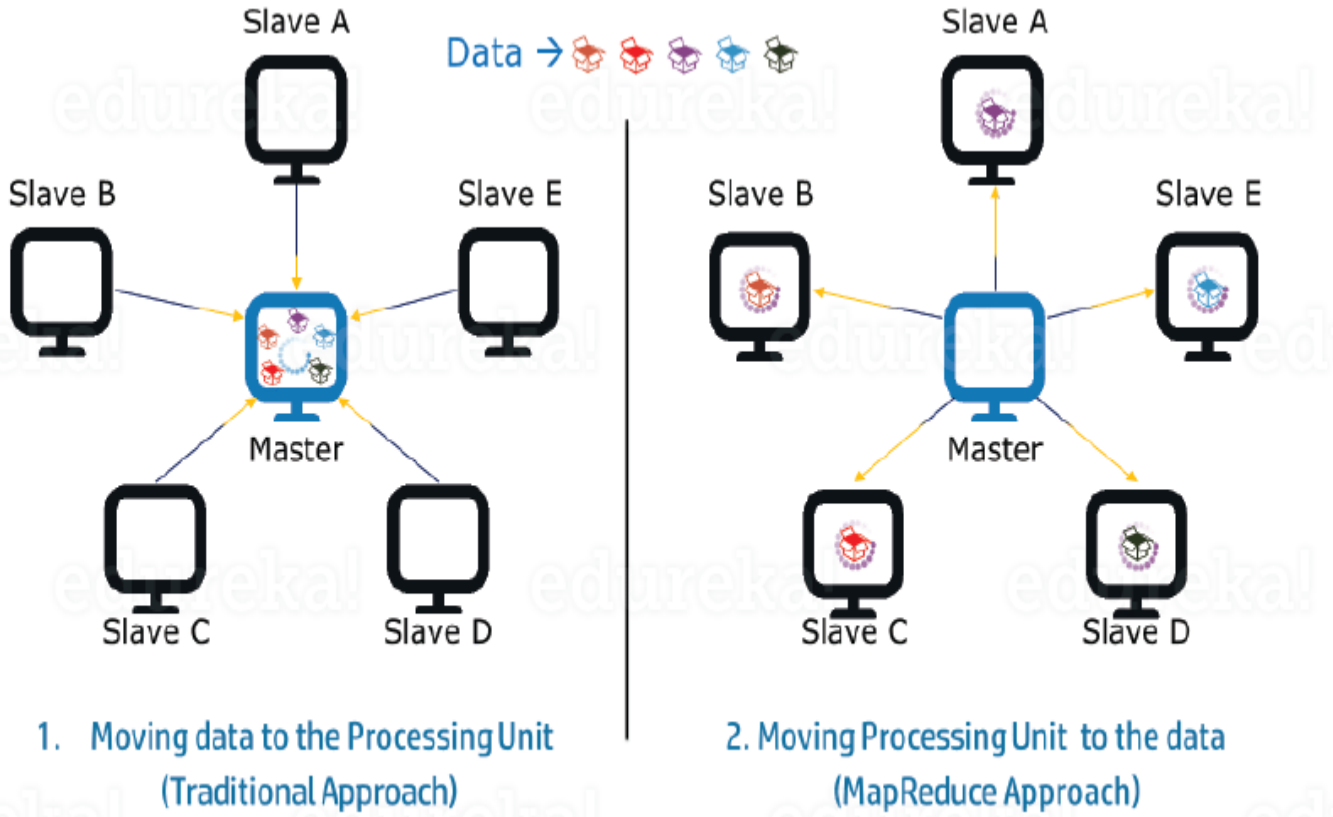
- Map-function
- Reduce-function

Advantages of MapReduce

1. Parallel Processing:



2. Data Locality:



Map – Reduce operation in Hadoop:-

- Hadoop divides the job into tasks. There are two types of tasks:
  - ✓ Map tasks (Spilts & Mapping)
  - ✓ Reduce tasks (Shuffling, Reducing)
- The complete execution process (execution of Map and Reduce tasks, both) is controlled by two types of entities called
  - ✓ **Jobtracker** : Acts like a master (responsible for complete execution of submitted job)
  - ✓ Multiple **Task Trackers** : Acts like slaves, each of them performing the job

MapReduce

For every job submitted for execution in the system,

-**Jobtracker** that resides on Namenode

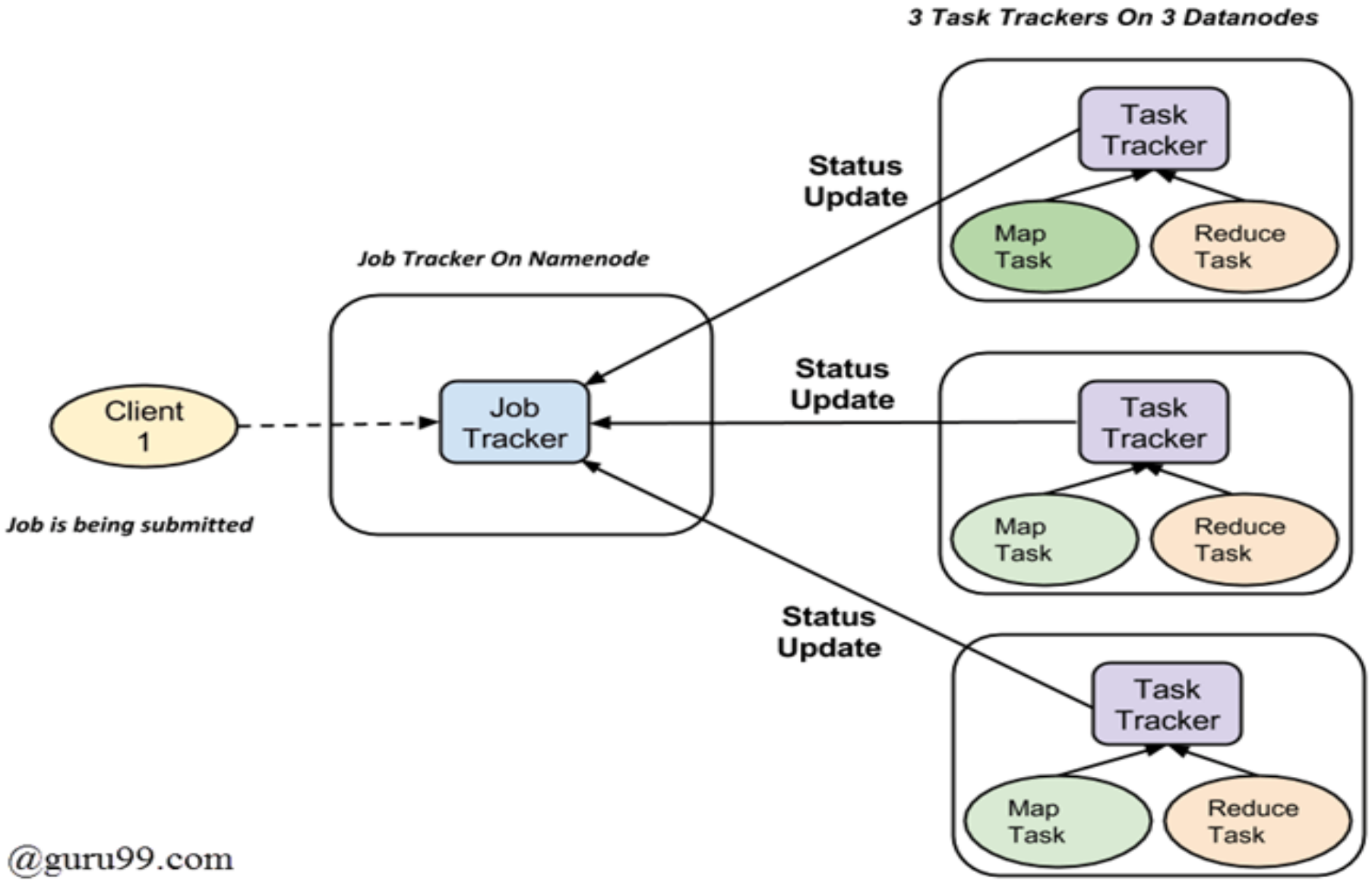
-**tasktrackers** which reside on Datanode

- ✓ A job is divided into multiple tasks which are then run onto multiple data nodes in a cluster.
- ✓ It is the responsibility of jobtracker to coordinate the activity by scheduling tasks to run on different data nodes.
- ✓ Execution of individual task is then look after by tasktracker, which resides on every data node executing part of the job.

  Tasktracker's → progress report → jobtracker → keeps track each job.

- ✓ In the event of task failure, the jobtracker can reschedule it on a different tasktracker.

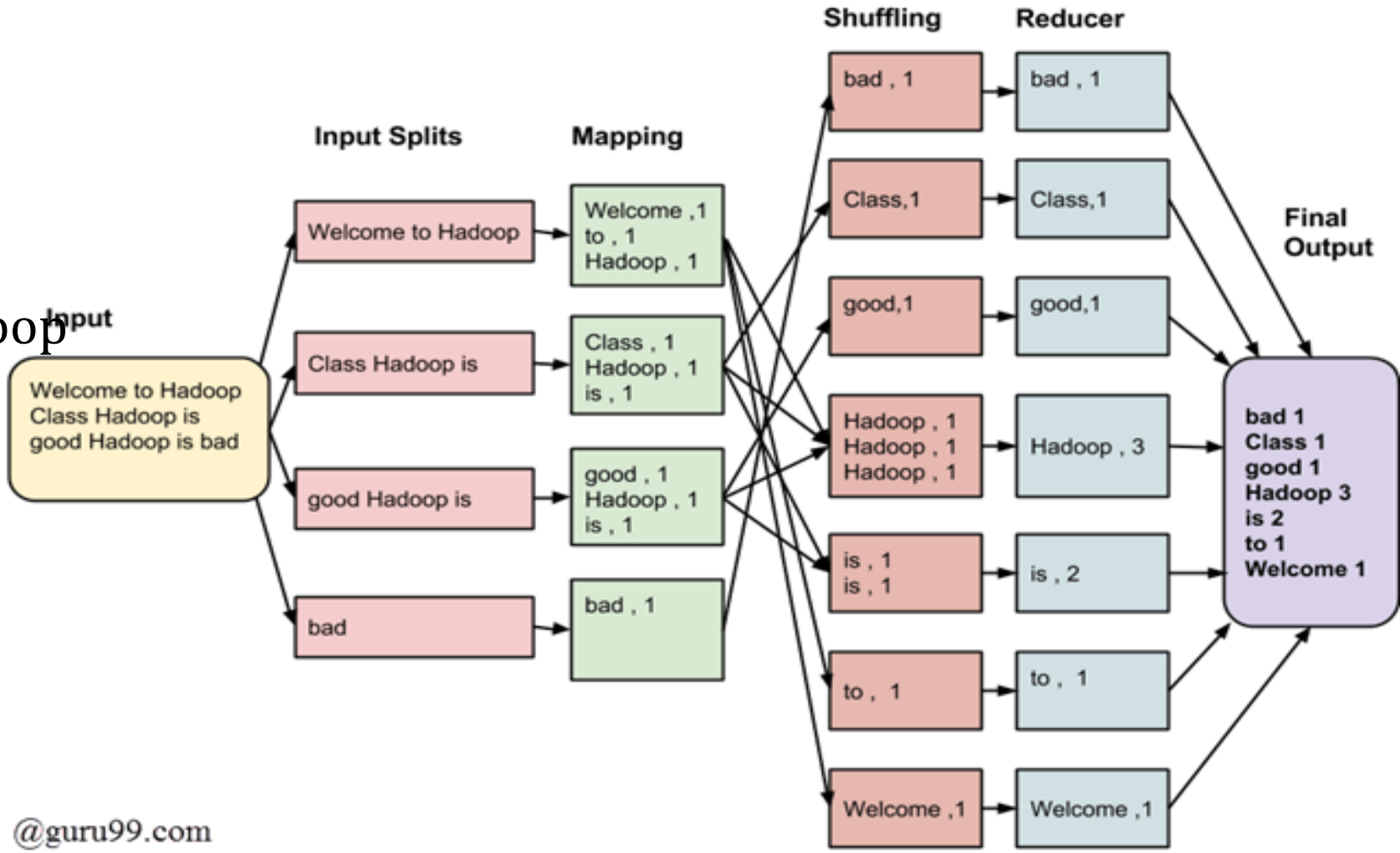
Hadoop core components or daemons:-



Example 1:-

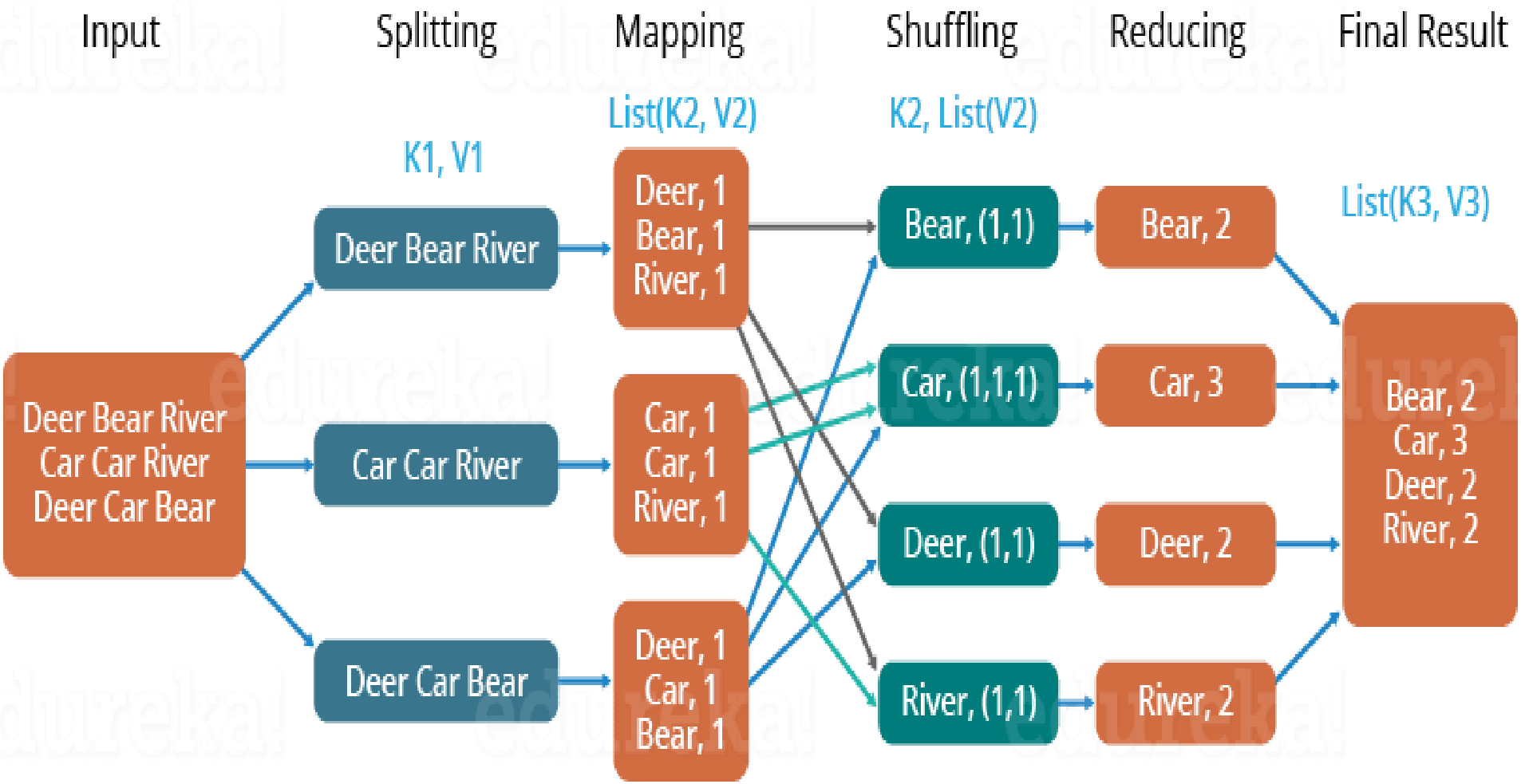
Consider that you have following input data for your MapReduce Program

Welcome to Hadoop
Class
Hadoop is good
Hadoop is bad



Example 2:-

A Word Count Example of MapReduce



Clustering types in Hadoop:-

Standalone Mode:

In this mode, there are no Hadoop Daemons (NameNode, DataNode, Secondary NameNode, JobTracker & TaskTracker) that are running in the background. As the name suggests, everything in standalone mode runs in a single machine.

Pseudo-Distributed Mode:

Pseudo distributed cluster is a cluster where all daemons (NameNode, DataNode, Secondary NameNode, JobTracker & TaskTracker) are running on one node itself.

Fully Distributed Mode:

This mode involves the code running on an actual Hadoop cluster.

Data are used and distributed across many nodes.

Different Nodes will be used as Master Node / Data Node / Job Tracker / Task Tracker

Hadoop 1.x Limitations

Main drawback of Hadoop 1.x is that MapReduce Component in it's Architecture

- ✓ It is only suitable for Batch Processing of Huge amount of Data, which is already in Hadoop System.
- ✓ It is not suitable for Real-time Data Processing / Data Streaming.
- ✓ It supports upto **4000 Nodes** per Cluster.
- ✓ It has a single component : JobTracker to perform many activities like Resource Management, Job Scheduling, Job Monitoring, Re-scheduling Jobs etc.

- ✓ JobTracker is the single point of failure.
- ✓ It runs only Map/Reduce jobs.
- ✓ It follows Slots concept in HDFS to allocate Resources (Memory, RAM, CPU). It has static Map and Reduce Slots. That means once it assigns resources to Map/Reduce jobs, it cannot re-use them even though some slots are idle.

For Example:- Suppose, 10 Map and 10 Reduce Jobs are running with 10 + 10 Slots to perform a computation. All Map Jobs are doing their tasks but all Reduce jobs are idle. We cannot use these Idle jobs for other purpose.

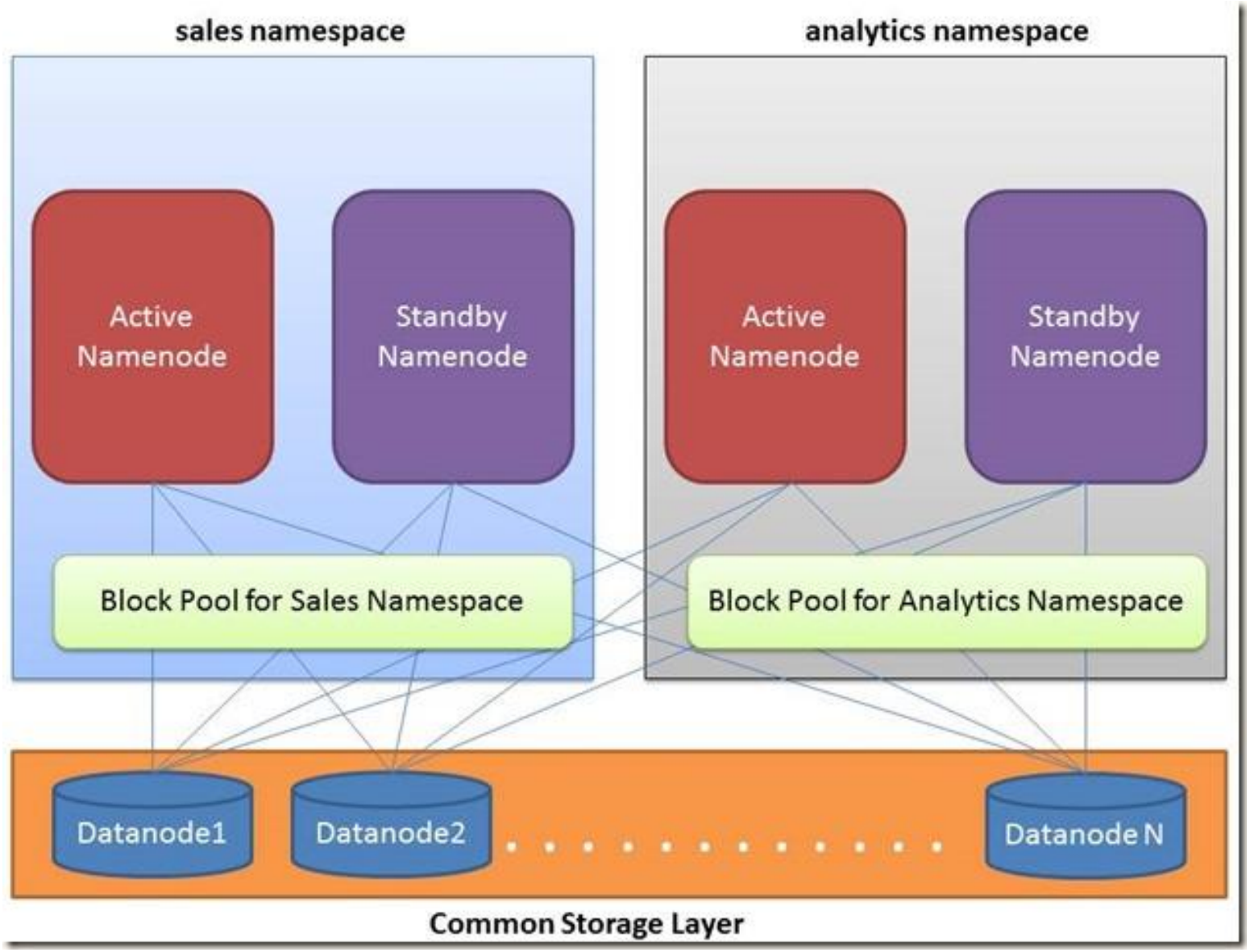
Hadoop 2.x

YARN (Yet Another Resource Negotiator)

- ✓ Hadoop 2.x Allows to work in MR as well as other distributed computing models like Spark, Hama, Giraph, Message Passing Interface) MPI & HBase coprocessors.
- ✓ 2.x Has better scalability. Scalable up to 10000 nodes per cluster.
- ✓ Works on concepts of containers. Using containers can run generic tasks.

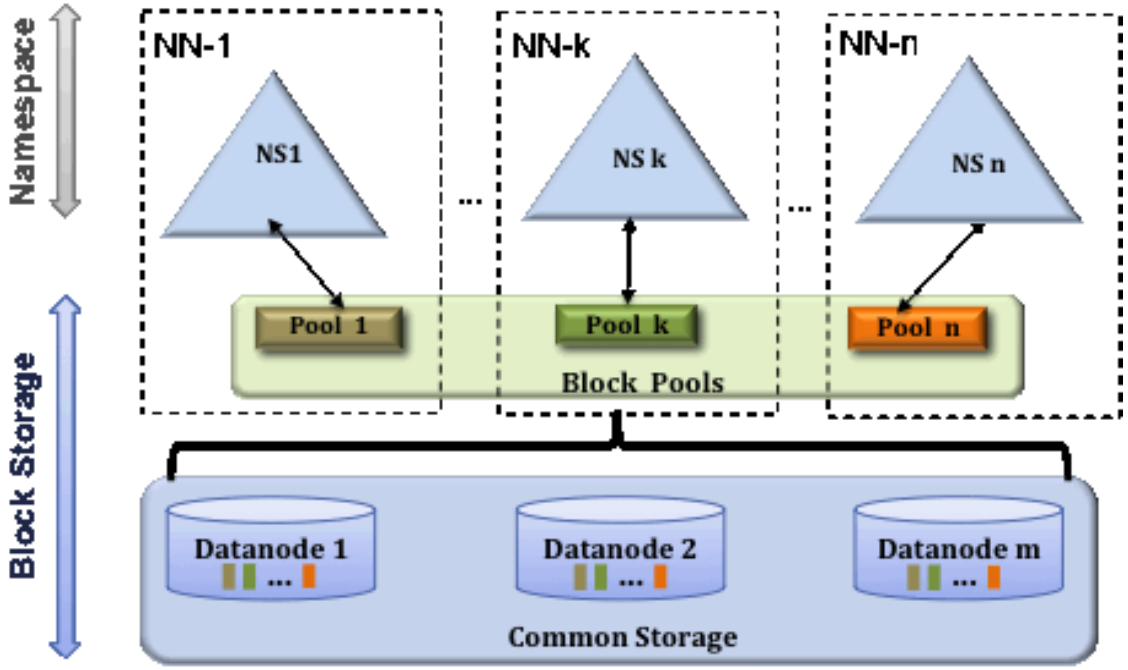
- ✓ Multiple Namenode servers manage multiple namespace.
- ✓ 2.x Has feature to overcome SPOF with a standby Namenode and in case of Namenode failure, it is configured for automatic recovery.
- ✓ Can serve as a platform for a wide variety of data analytics-possible to run event processing, streaming and real time operations.

HDFS 2.0: High Availability, Federated



http://pe-kay.blogspot.in/2016/02/configuring-federated-hdfs-cluster-with.html

HDFS 2.0: Name Node Federation Elaborated



- Multiple **independent** Namenodes and Namespace Volumes in a cluster
  - Namespace Volume = Namespace + Block Pool
- Block Storage as generic storage service
  - Set of blocks for a Namespace Volume is called a ***Block Pool***
  - DNs store blocks for all the Namespace Volumes – no partitioning
- Now DataNode will need to send heartbeats to multiple namenodes
  - With advent of zookeeper, datanode needs to send one heartbeat, zookeeper controls the broadcast