# DISTRIBUTED FILE SYSTEM

**Venkatesh Vinayakarao**
venkateshv@cmi.ac.in
http://vvtesh.co.in

Chennai Mathematical Institute

**“Hadoop” is a philosophy — a movement towards a modern architecture for managing and analyzing data.** – Arun Murthy, Hortonworks, Cloudera, 2019.

**The notion of time is an important concept in every day life of our decentralized “real world“** - Friedemann Mattern.

# What Comes Next?

byte

kilobyte

megabyte

gigabyte

??

???

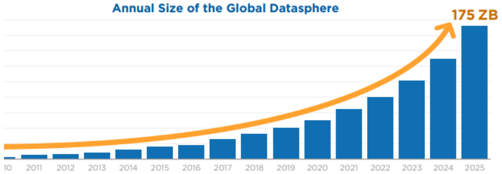
????

?????

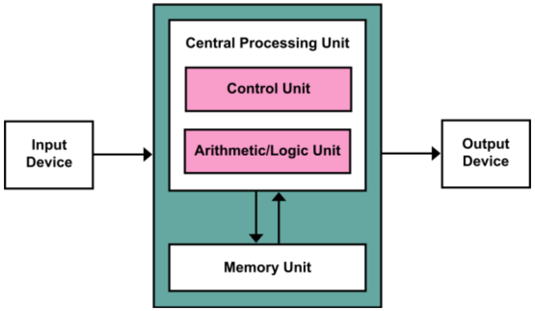
# Sizes

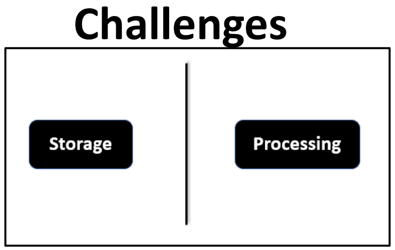
| Name | Size |
|---|---|
| Byte | 8 bits |
| Kilobyte | 1024 bytes |
| Megabyte | 1024 kilobytes |
| Gigabyte | 1024 megabytes |
| Terabyte | 1024 gigabytes |
| Petabyte | 1024 terabytes |
| Exabyte | 1024 petabytes |
| Zettabyte | 1024 exabytes |
| Yottabyte | 1024 zettabytes |

# Recap



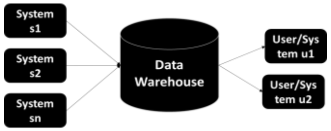
Source: Data Age 2025, sponsored by Seagate with data from IDC Global DataSphere, Nov 2018

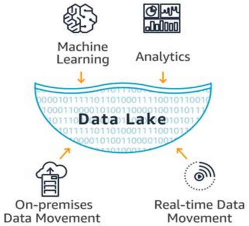




# Recap

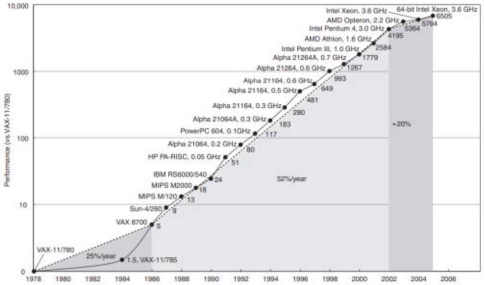
## Data Storage



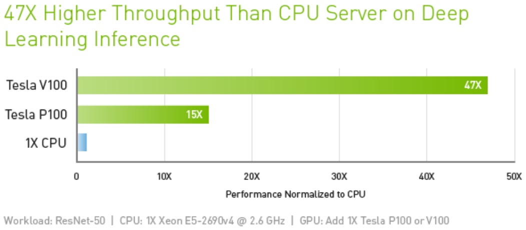


STaaS

## Data Processing



CPU Performance



GPU Performance

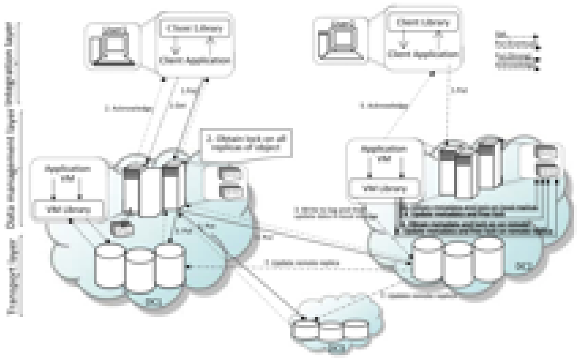
SuperComputers

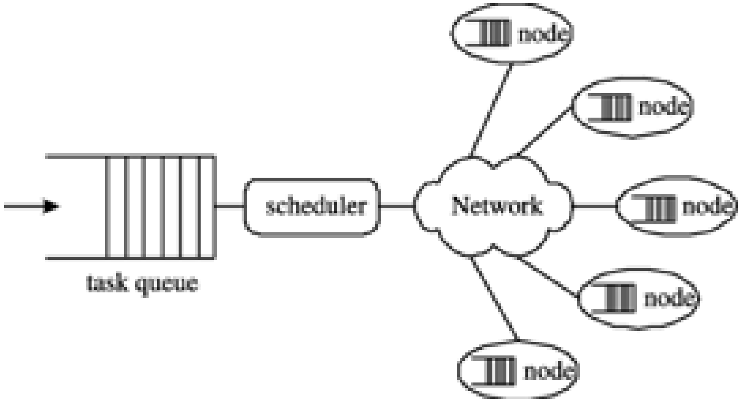
# Cloud Computing

Two kinds of Big Data Opportunities

**Storage**

**Processing**





**So, we have the cloud. But, how to store and retrieve data? How to process jobs?**

# Role of File Systems

Partitioning

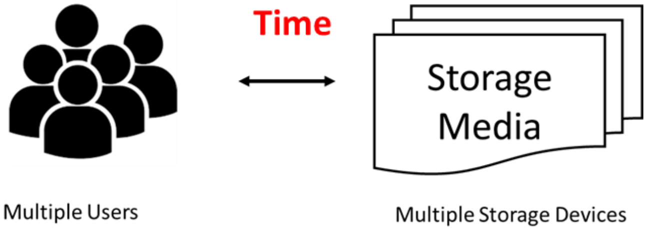
Multiple OS, Multiple File Systems

Compression

High Data Transfer Time

Defragmentation

High Seek Time



File Allocation, Free Space Management

Space Utilization

Multi-Tenancy & data privacy

Permissions and Sharing

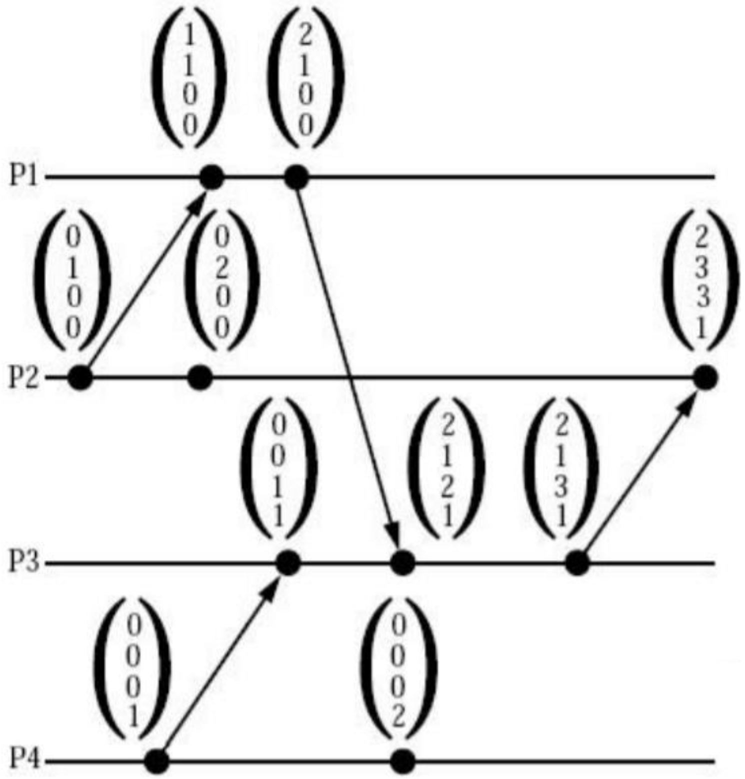
Data Variety

Naming Convention - Standards

**File systems are key to handling data.**

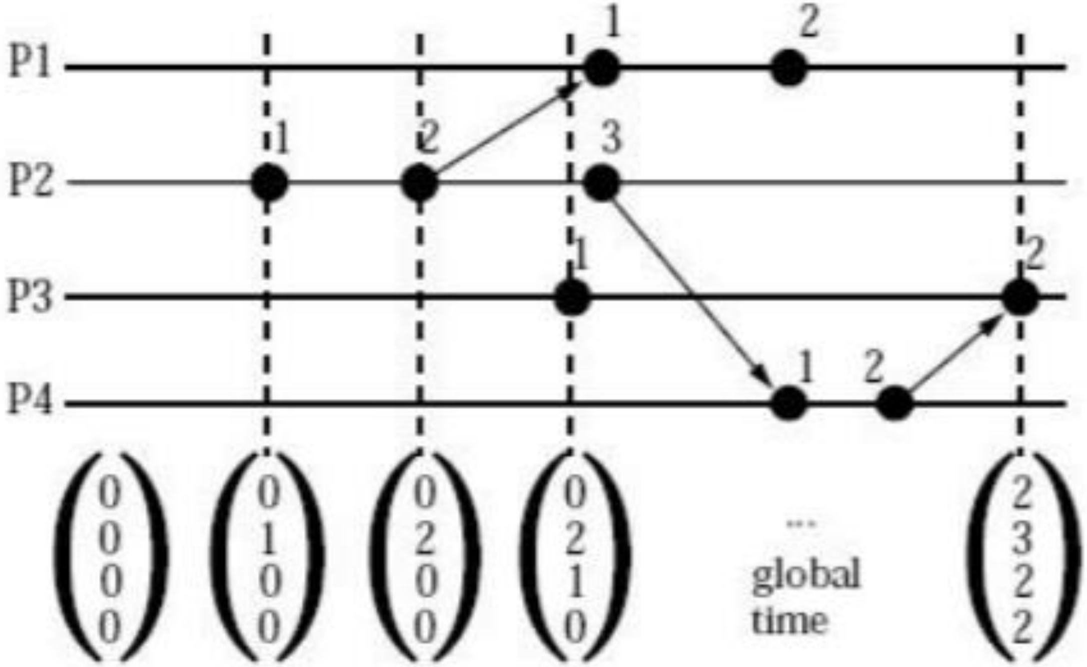
Variety of FS exist
NTFS, FAT, DOS, CDFS, NFS, …

# Vector Time Stamps



- Local clock is incremented every time an event occurs.
- An external observer may know about all events.
- Global time knowledge can be saved as a vector, with one element per process.

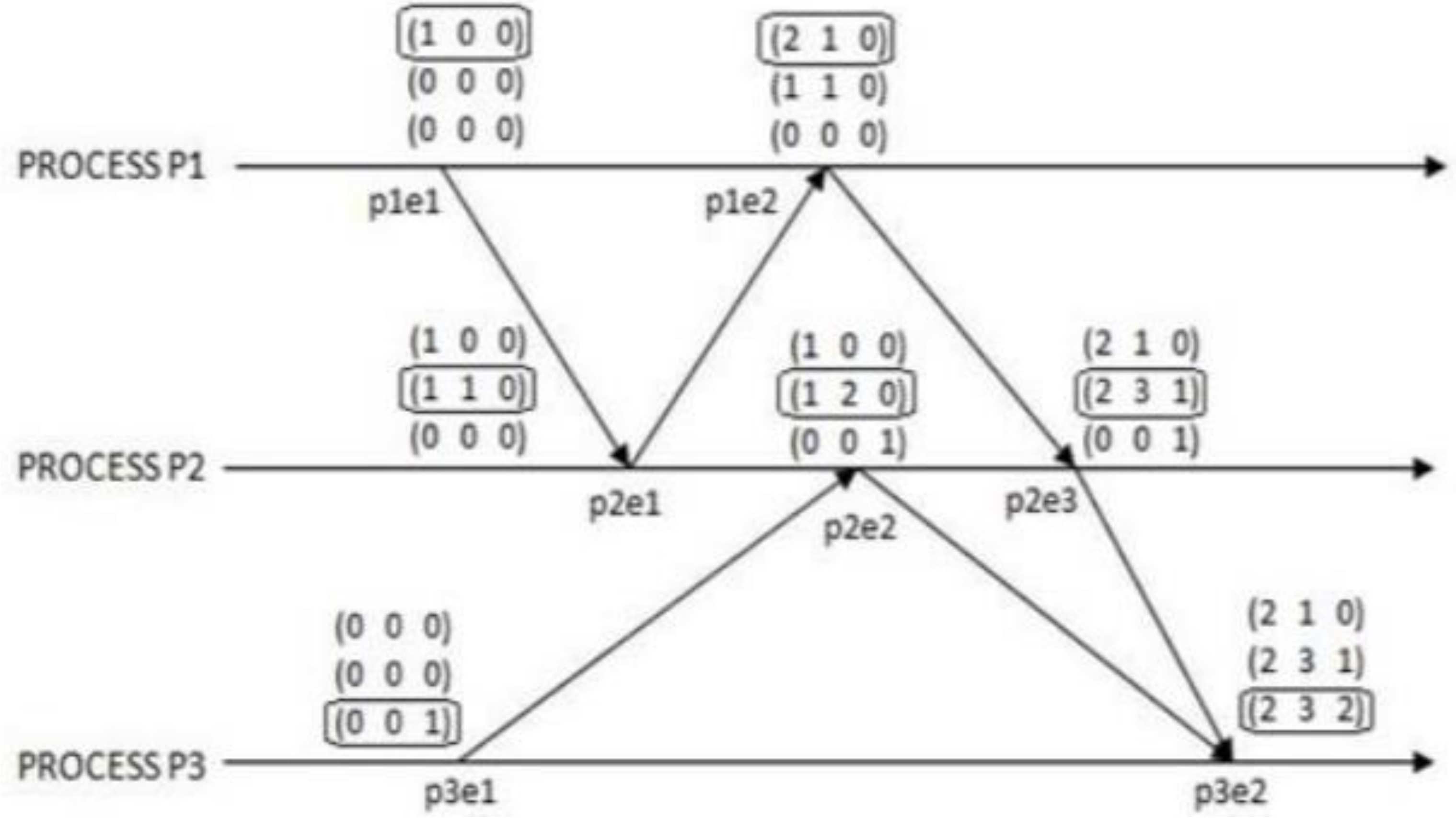
# Global Vector Time



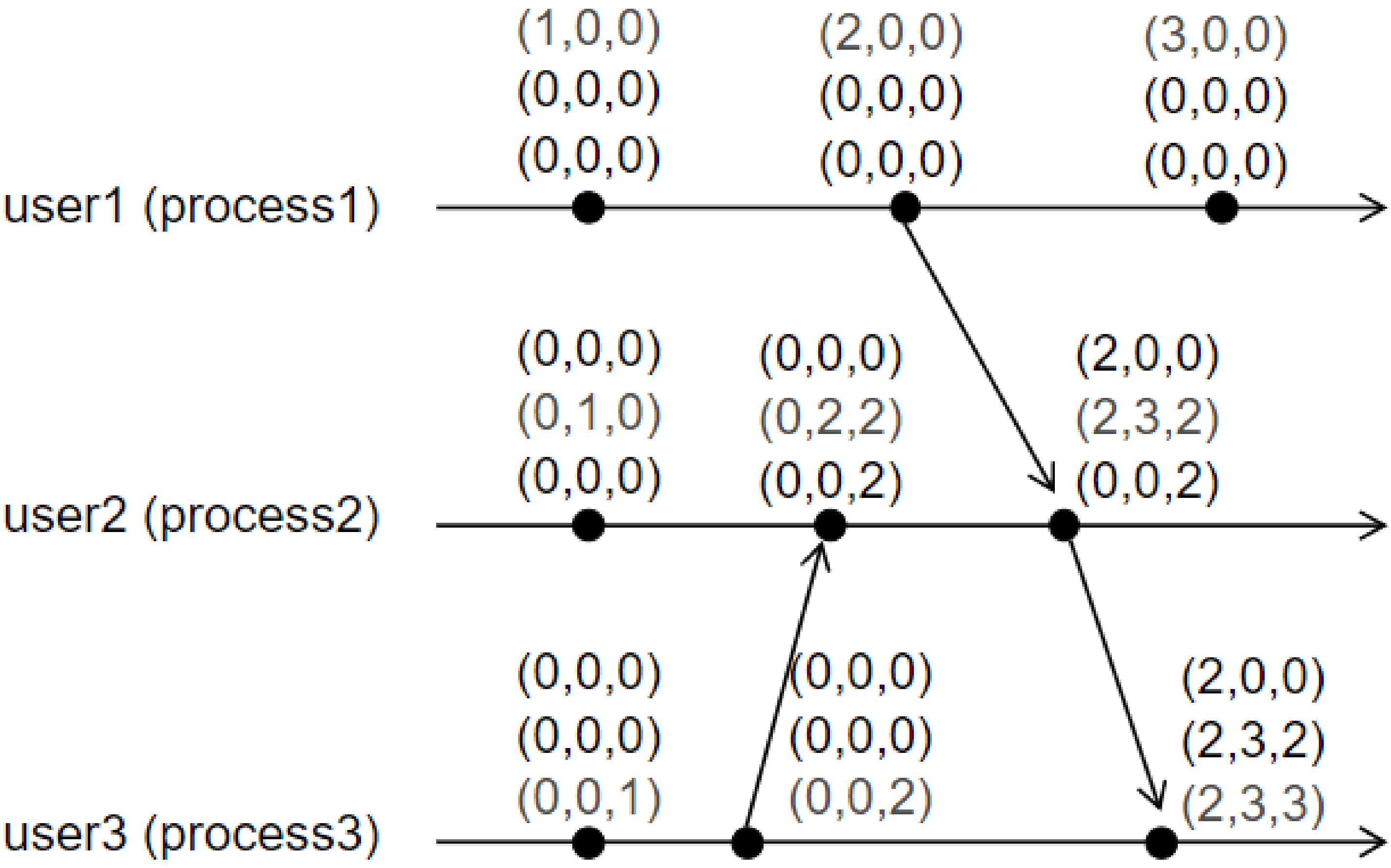
# Quiz

- Which of the below specify a strictly "**happens-before**" relationship between two event time stamps?
    - (2,0,0) → (3,0,0)
    - (2,2,1,3) → (3,3,2,4)
    - (1,2,1,2) → (1,1,2,2)
    - (3,3,2,4) ← (2,2,1,3)

# Matrix Time

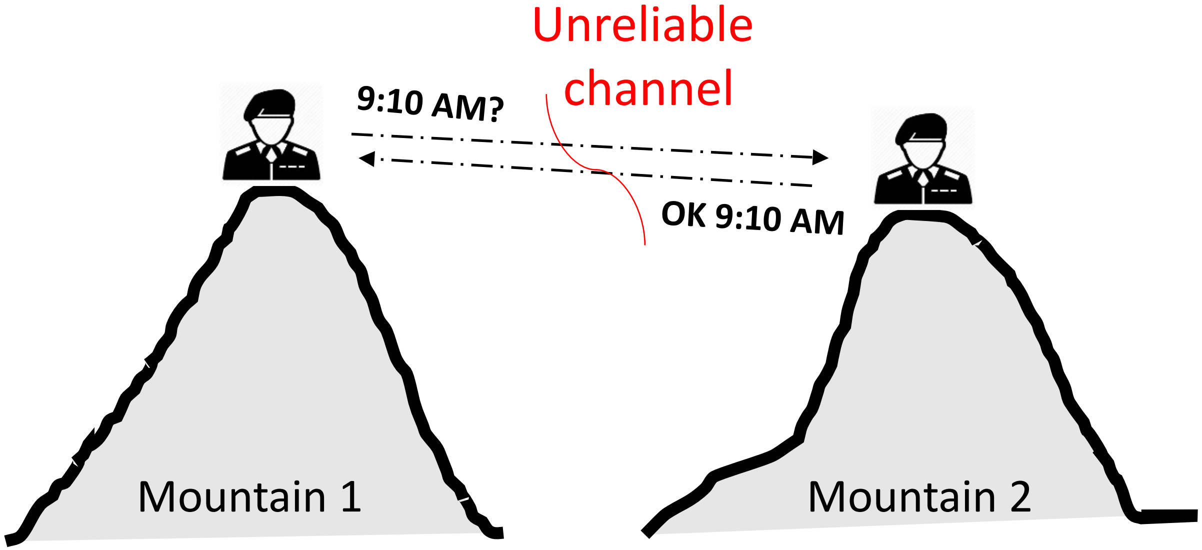


Read http://ijcsit.com/docs/Volume%206/vol6issue04/ijcsit2015060455.pdf



Source: https://slideplayer.com/slide/8289837/

# Limitations



**General's Paradox**

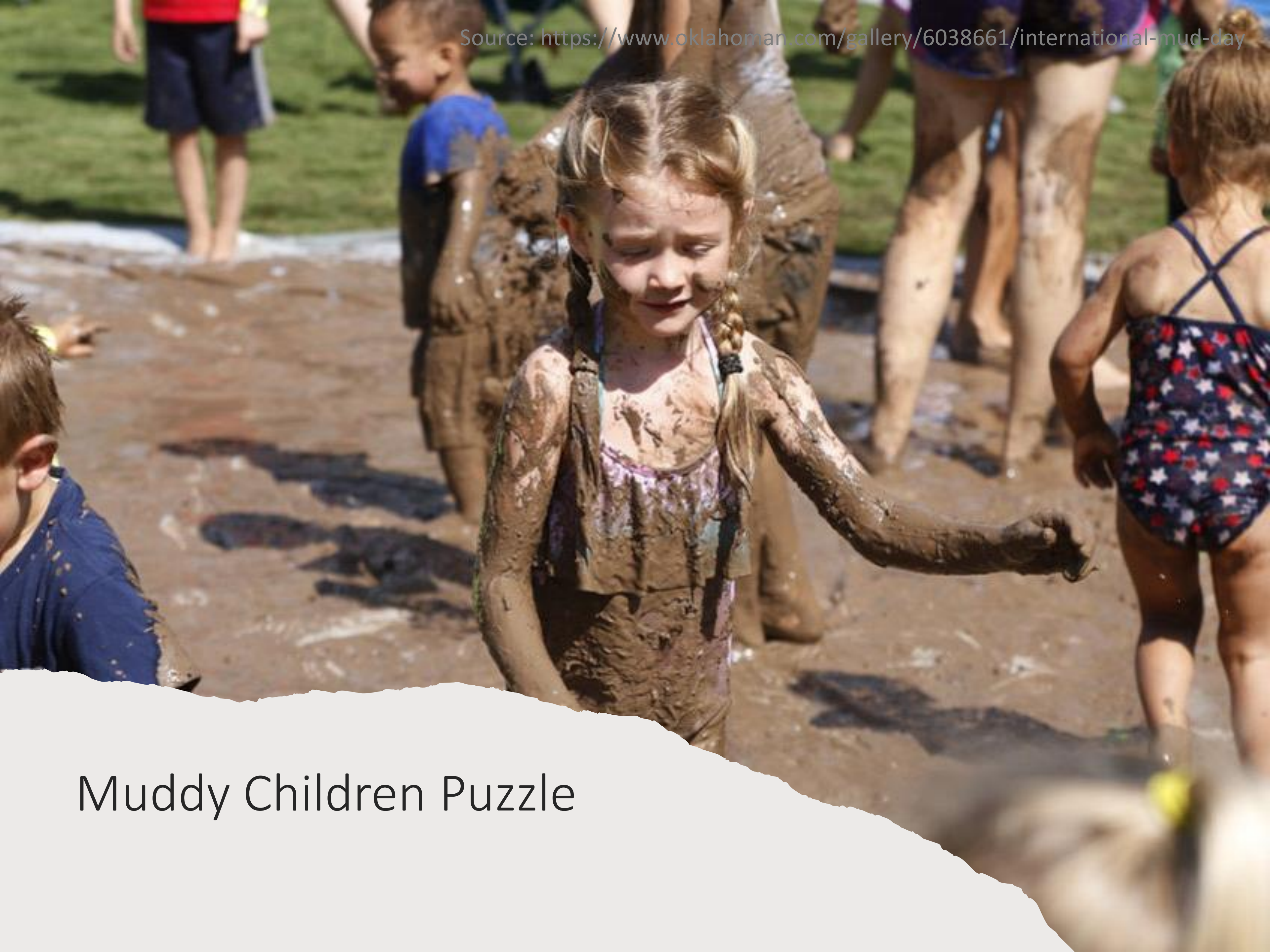
Two generals must attack at the same time, or they die.

**Is there a way to coordinate?**

What if, two machines need to coordinate, but not necessarily at the same time?

**Is it possible?**

Yes! Message Passing.



Source: https://www.oklahoman.com/gallery/6038661/international-mud-day

# Muddy Children Puzzle

# Muddy Children Puzzle

- At least one child has mud on forehead and every child knows that.
- There are k (where k <= n) out of n children with mud on head.
- Children are intelligent and truthful.
- A child can see all other children and say if others have mud on their forehead.
- A child does not know if his/her forehead is muddy.
- C[illegible]

Assuming each child as a distributed process, how will they co-ordinate so that only the k children who have muddy forehead say ‘yes’.

# Muddy Children Puzzle

- How would it work?
    - n=1
        - k=1
    - n=2
        - k=1
        - K=2
    - n=3
        - k=1
        - k=2
        - k=3

## What is an operating system?

Yarn is now the Apache Hadoop Operating System

**Apache Hadoop**

Open source platform for reliable, scalable,  distributed processing of large data sets, built on clusters of commodity computers.

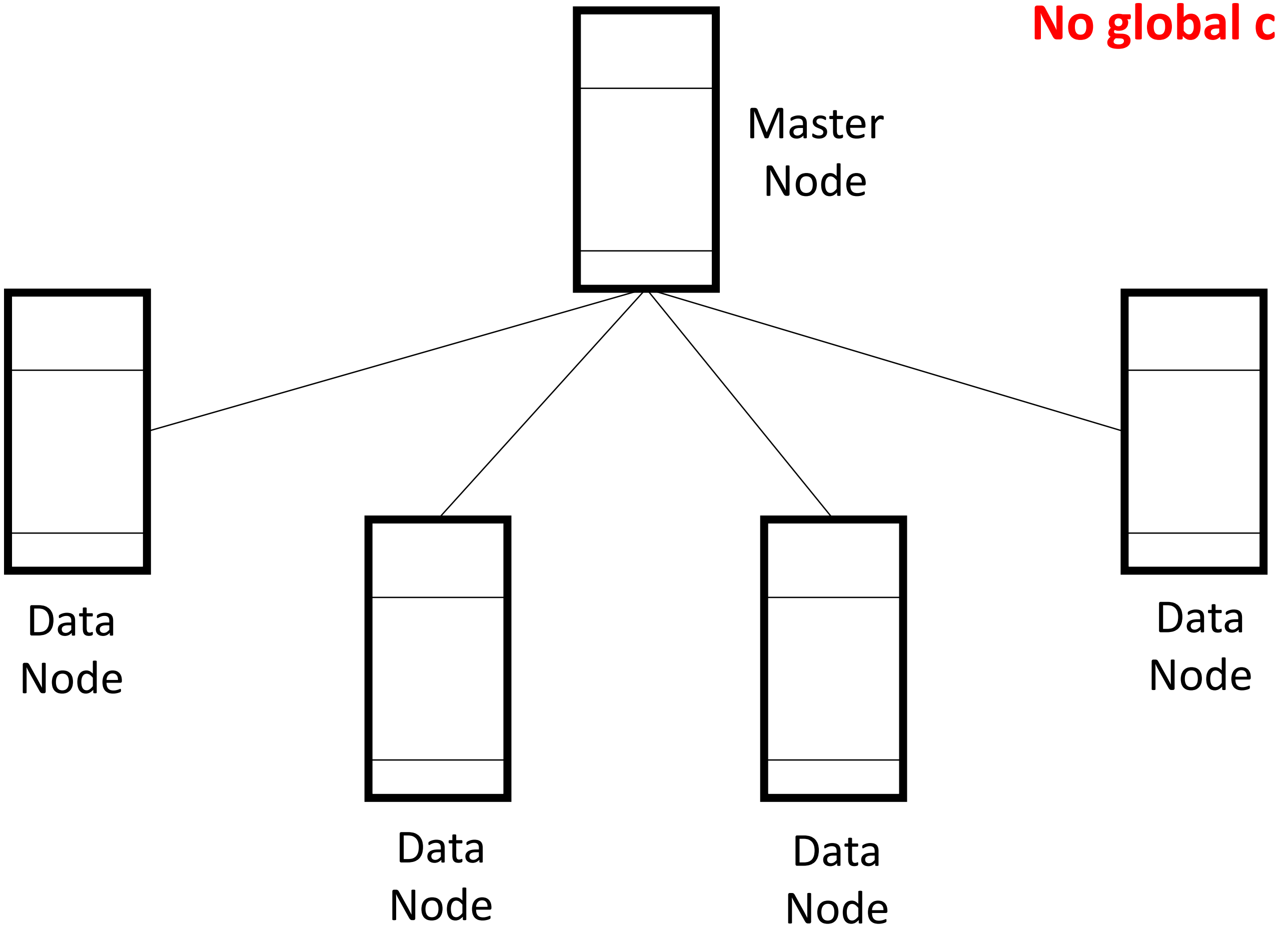
# Distributed File Systems - Key Goals

- Distribution Transparency
- Location Transparency
- Scalability
- Fault Tolerance
- Efficient Data Access
  - Specifically designed for batch jobs
- “Write Once Read Many” (WORM) model

**Several examples: Andrew FS, Network FS, HDFS**

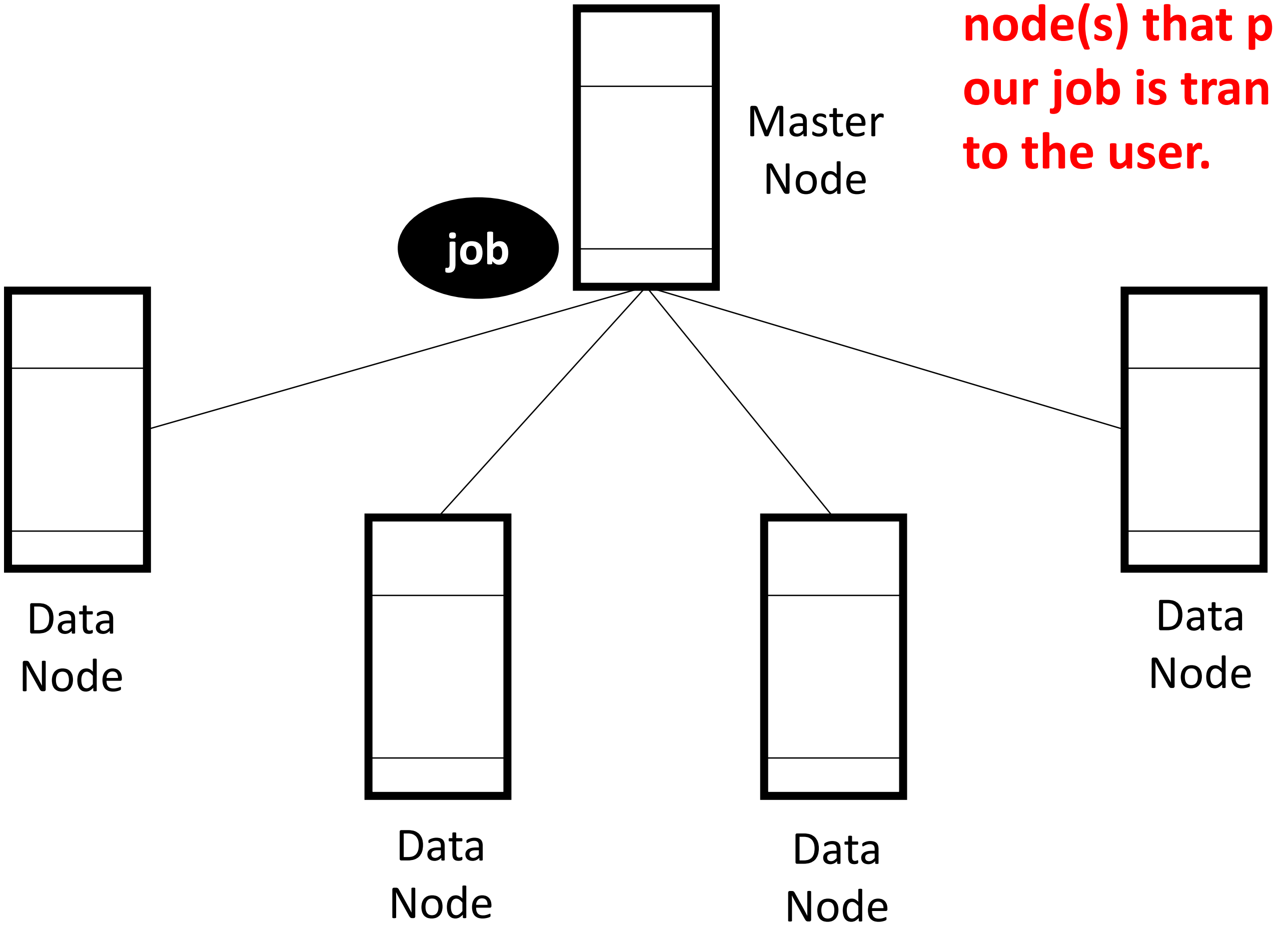
# Distributed System

No global clock!



**Master-Slave Architecture**

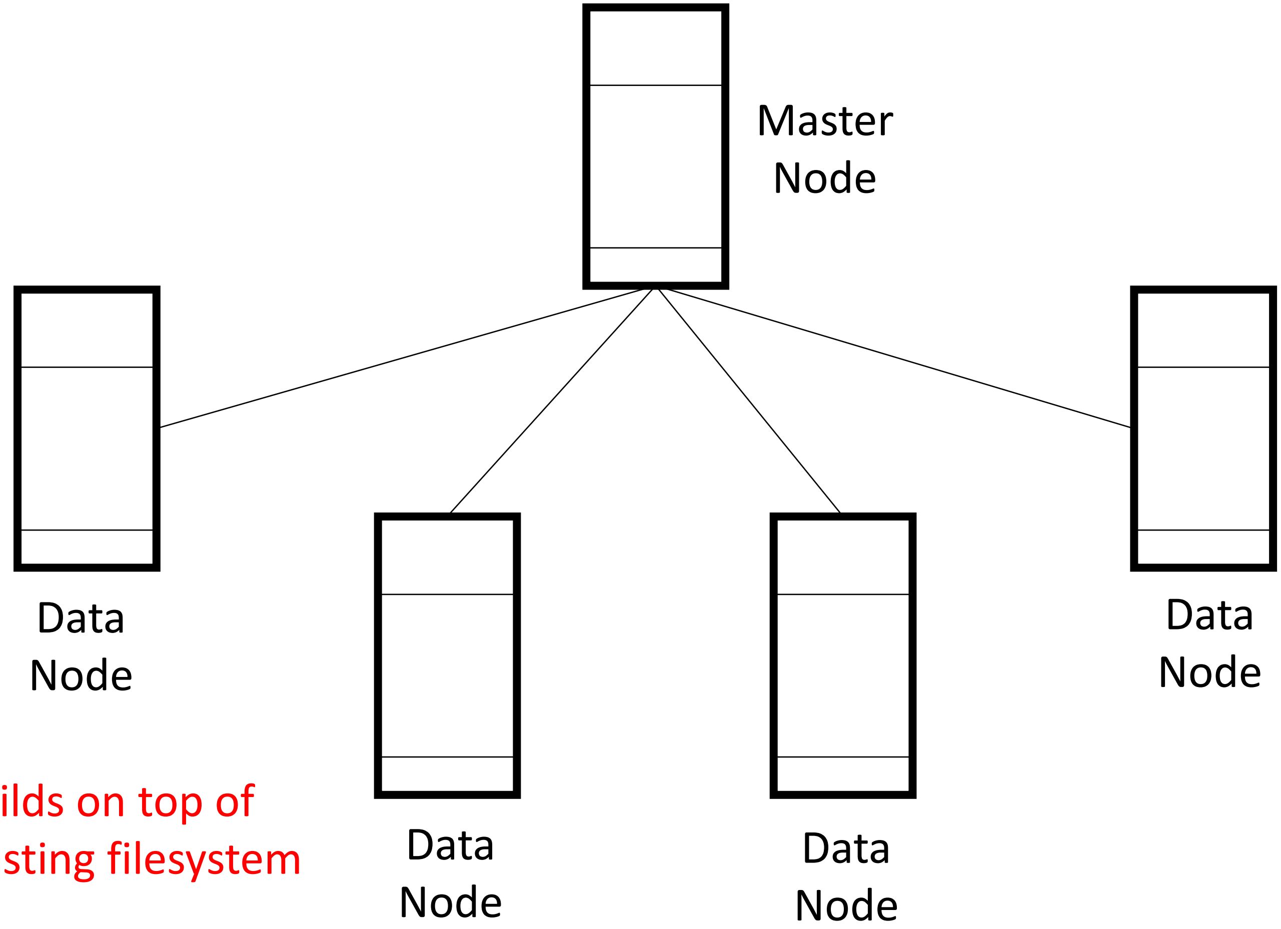
# Distribution Transparency



**node(s) that process our job is transparent to the user.**

**Master-Slave Architecture**

# Hadoop Distributed File System Architecture



Builds on top of existing filesystem

**Master-Slave Architecture**

# Location Transparency

- Refers to uniform file namespace.

Example

```
hdfs  dfs  -cat  hdfs://nn1.cmi.ac.in/file1  hdfs://nn1.cmi.ac.in/file2
```

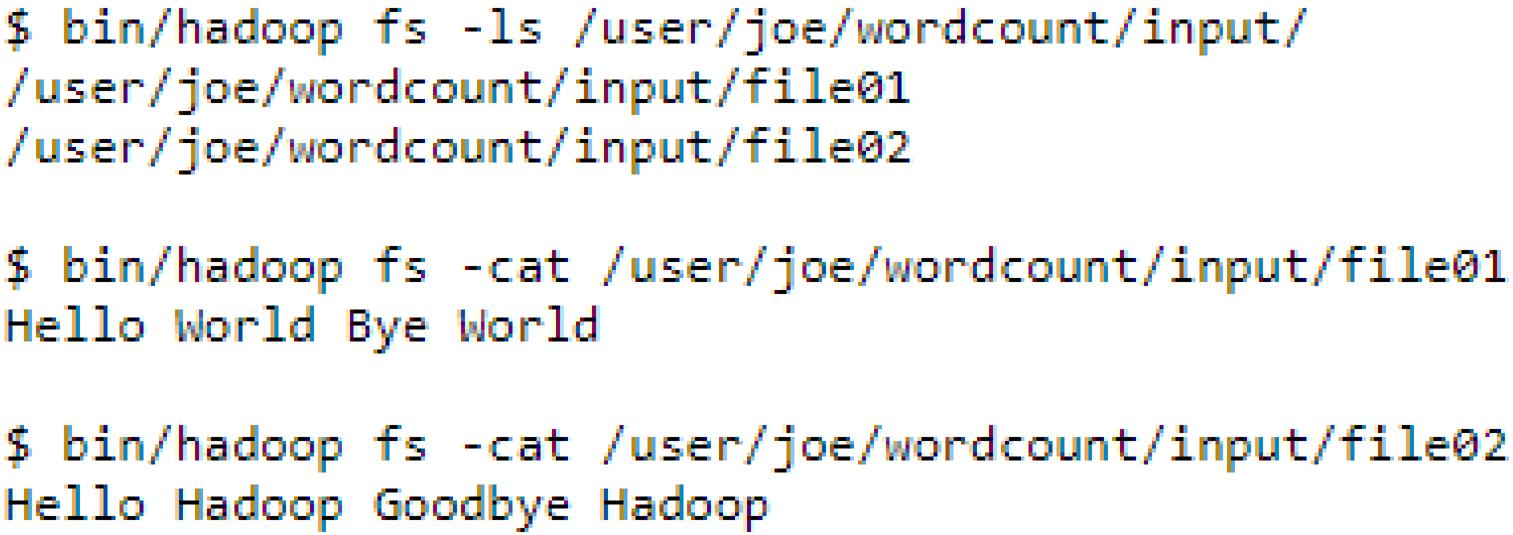
https://hadoop.apache.org/docs/r2.4.1/hadoop-project-dist/hadoop-common/FileSystemShell.html

# HDFS

- HDFS commands are very similar to UNIX shell commands
  - ls
  - du
  - mkdir
- Some additional commands
  - copyToLocal
  - copyFromLocal

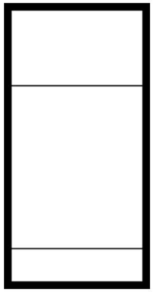
```
cd usr/data/
hdfs dfs -copyToLocal test/cmi.csv cmi.csv
```

# HDFS Commands

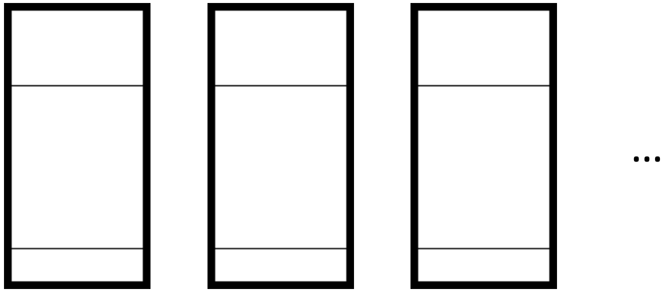


```
$ bin/hadoop fs -ls /user/joe/wordcount/input/
/user/joe/wordcount/input/file01
/user/joe/wordcount/input/file02

$ bin/hadoop fs -cat /user/joe/wordcount/input/file01
Hello World Bye World

$ bin/hadoop fs -cat /user/joe/wordcount/input/file02
Hello Hadoop Goodbye Hadoop
```

# Scalability



Scale **up**
(add resources
to a single node)
Vertical Scaling



Scale **out**
(add more nodes)
Horizontal Scaling

With Hadoop, we can scale both vertically and horizontally.

# Scale-up or Scale-out?

- What would you prefer and why?

https://www.microsoft.com/en-us/research/publication/scale-up-vs-scale-out-for-hadoop-time-to-rethink/

# Scale-up or Scale-out?

- What would you prefer and why?
  - Depends on data size.
    - Majority of real-world analytic jobs process < 100 GB data.
    - Hadoop is designed for petascale processing.
    - An evaluation (done at Microsoft) across 11 representative Hadoop jobs shows that scale-up is competitive in all cases.

https://www.microsoft.com/en-us/research/publication/scale-up-vs-scale-out-for-hadoop-time-to-rethink/

# Adding a New Data Node is Easy

- Prepare the datanode
  - JDK, Hadoop, Environment Variables, Configuration (point to master)
- Start the datanode
  - hadoop-daemon.sh start datanode
- Run disk balancer to if you wish to redistribute existing data.

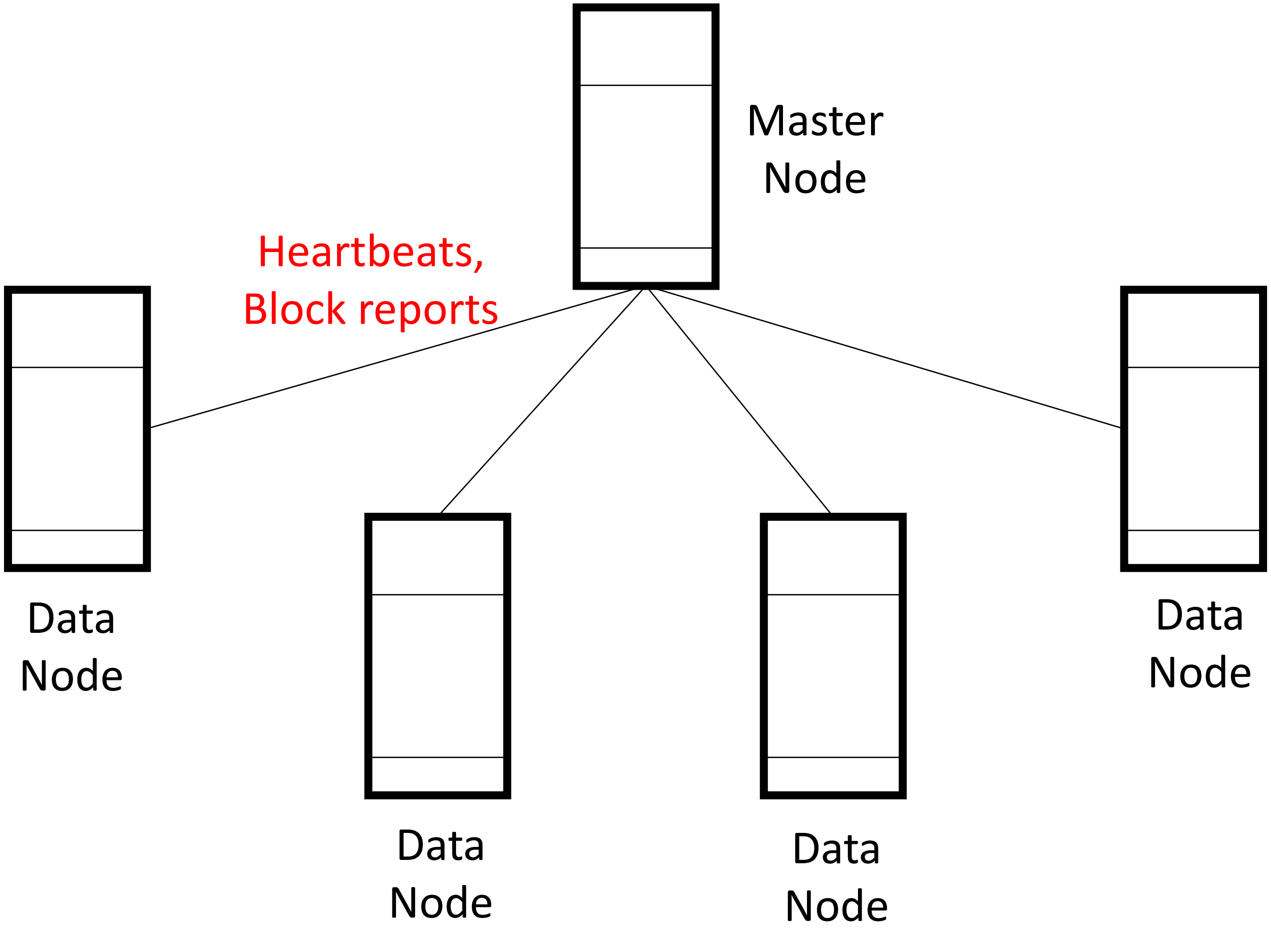
# Fault Tolerance

- Typical clusters have 1000+ datanodes.
- Unfavorable Situations
  - Blocks of data may get corrupted.
  - Datanodes may go down.
  - Network links may go down.

Try This!

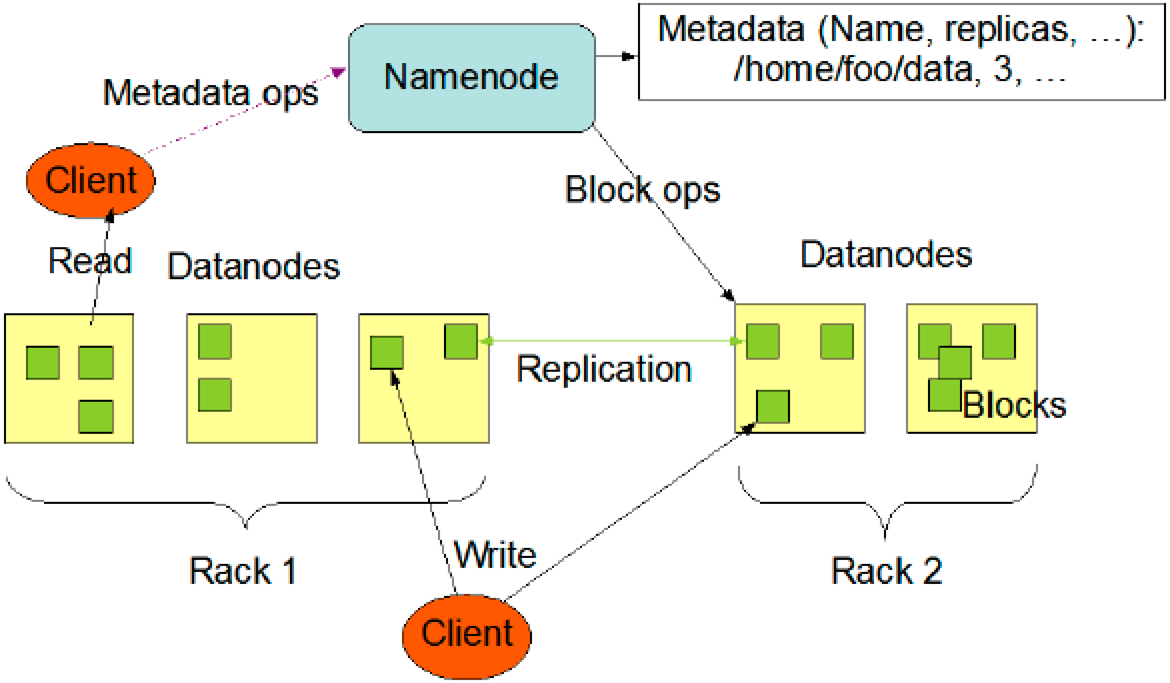
**Assume we have a 1000 node cluster with each node having a single disk. Also, assume that the disk life is such that every disk fails in three years. How many nodes will be down on an arbitrary day due to disk failure?**

# Fault Tolerance



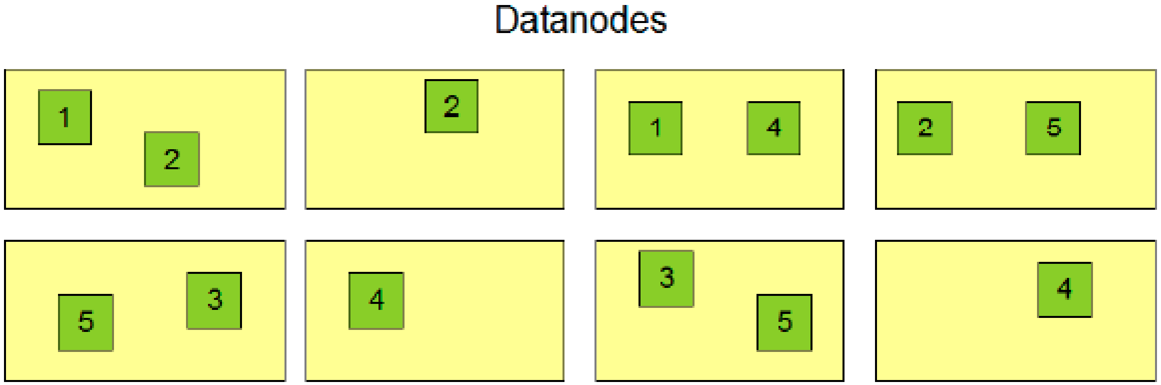
**Master-Slave Architecture**

# HDFS Data Replication



Source: HDFS Architecture Guide, Dhruba Borthakur.

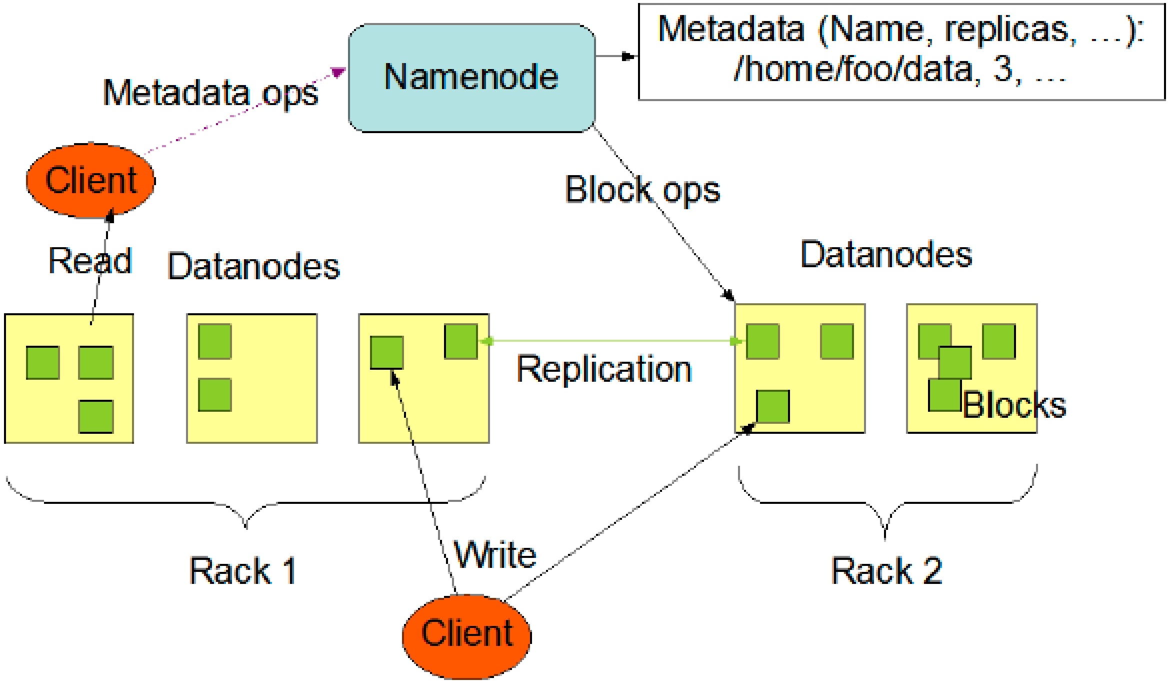
# Replication



# Single Point of Failure

**Is namenode a single point of failure?**

**Hadoop 2.0 supports primary and secondary namenodes**



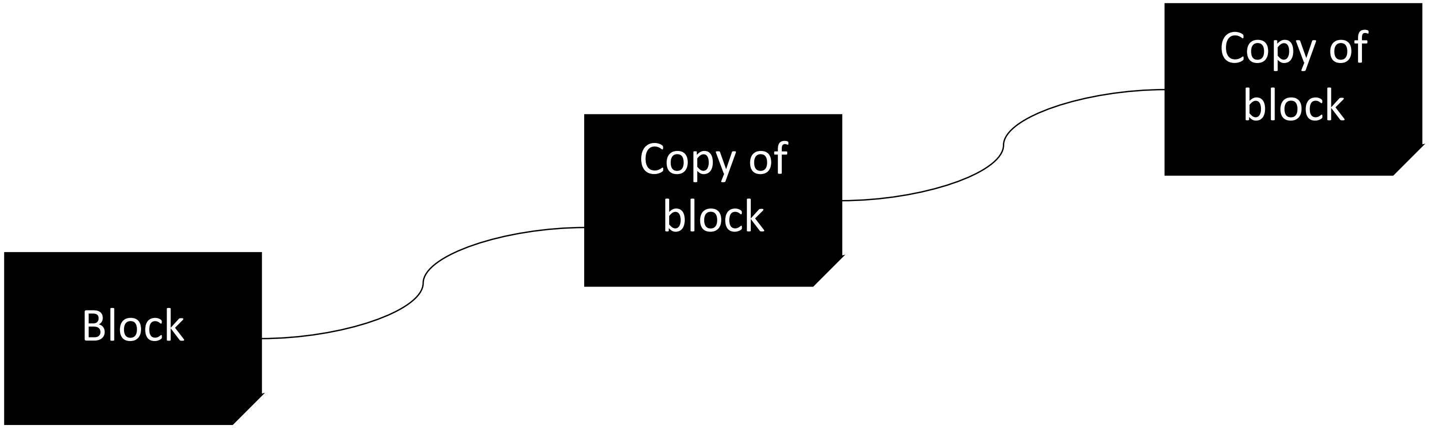
Source: HDFS Architecture Guide, Dhruba Borthakur.

# Efficient Data Access

- Write Once Read Many (WORM) model

# Write Once Read Many

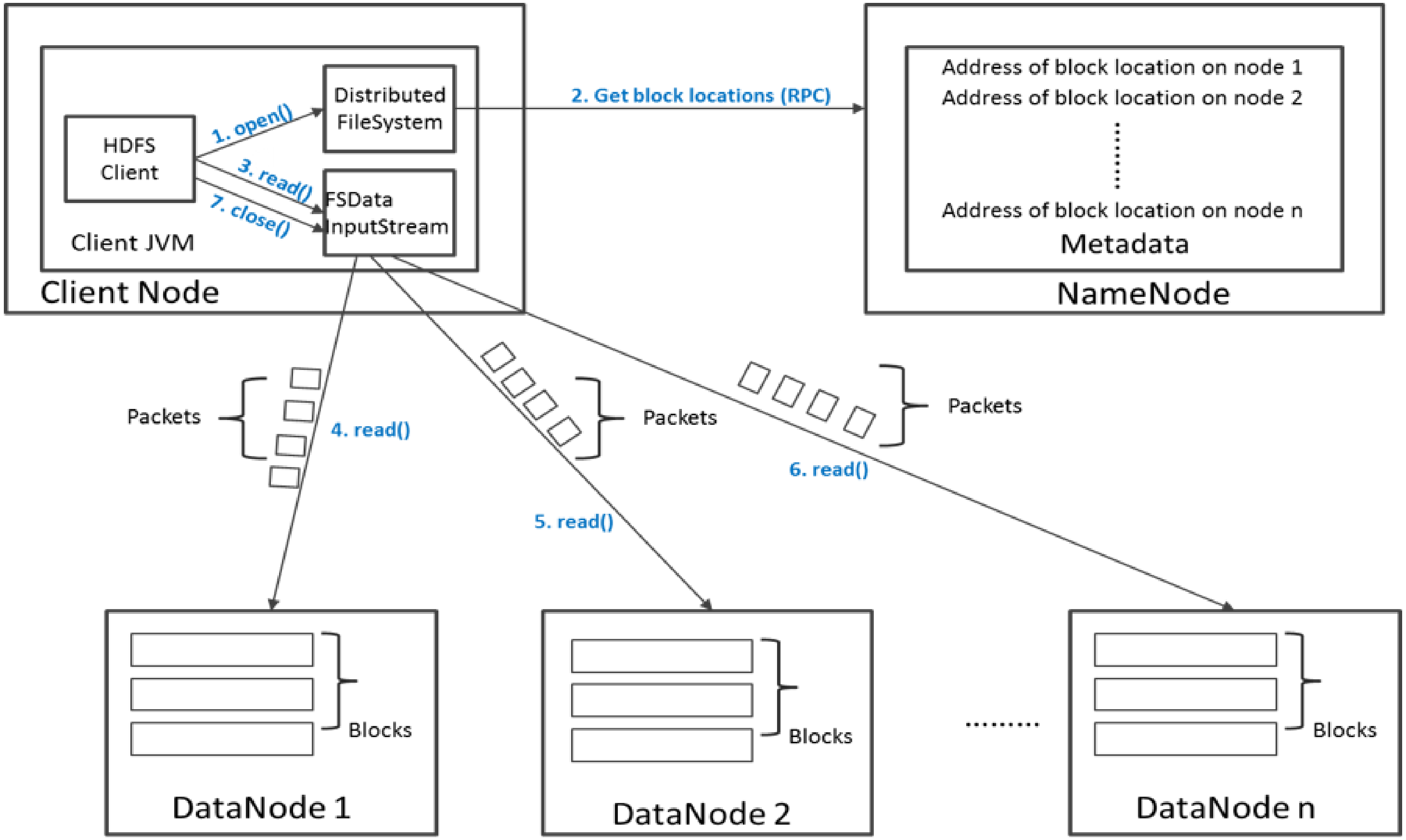
- Simplifies Data Coherency



**Need to keep all the copies in sync.**

- Designed for batch jobs

# HDFS – Data Read Operation



# Data Read Operation

- Client asks Namenode for block addresses
- Client accesses each block by accessing the datanodes **directly**.
- Since data is **accessed in parallel**, the reads are highly optimized.

# Data Write Operation

- Namenode **provides the address** of the datanodes
- Client **directly writes** data on the datanodes
- Datanode will **create data write pipeline**
  - First datanode copies the block to another datanode, which intern copy it to the third datanode
- Datanodes send **acknowledgment**

# Design Choices

- Default block size is 64 MB. Often used as is, or as 128 MB.
- How do you decide the right value for block size?

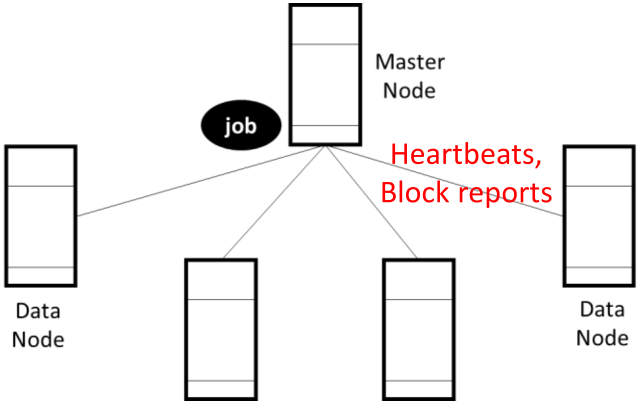
# Design Choices

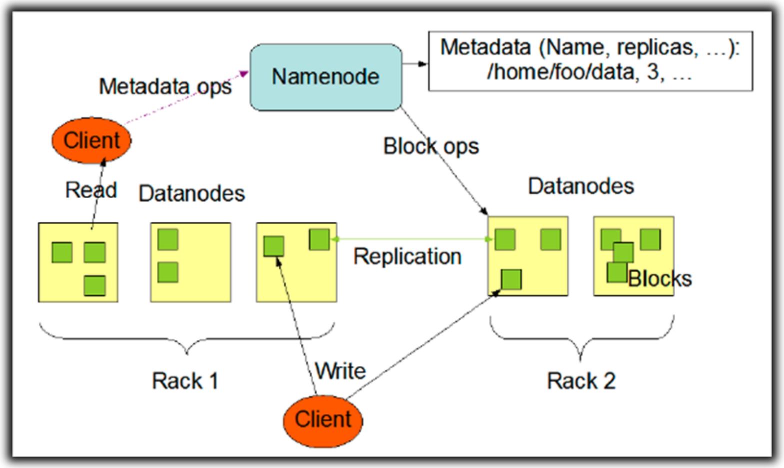
- Default block size is 64 MB. Often used as is, or as 128 MB.
- Block Size – Concerns:
  - Designed to handle large files (not small files, not even large number of small files).
  - For large number of small files, namenode needs to store too much metadata.
    - Solution: Sequence files.

# Sequence Files

- Hadoop specific file format.
- Files consisting of binary key/value pairs.
- Three types:
  - Uncompressed
  - Record Compressed
  - Block Compressed

# Summary





Distribution Transparency
Location Transparency
Scalability
Fault Tolerance

Efficient Data Access
“Write Once Read Many” (WORM) model