23MCT305 - Data Analytics in Automation System

Managing Hadoop File

HDFS Architecture

Faculty: N. KARTHI, AP/MCT

January 22, 2026

Distributed Storage | Fault Tolerance | Scalability

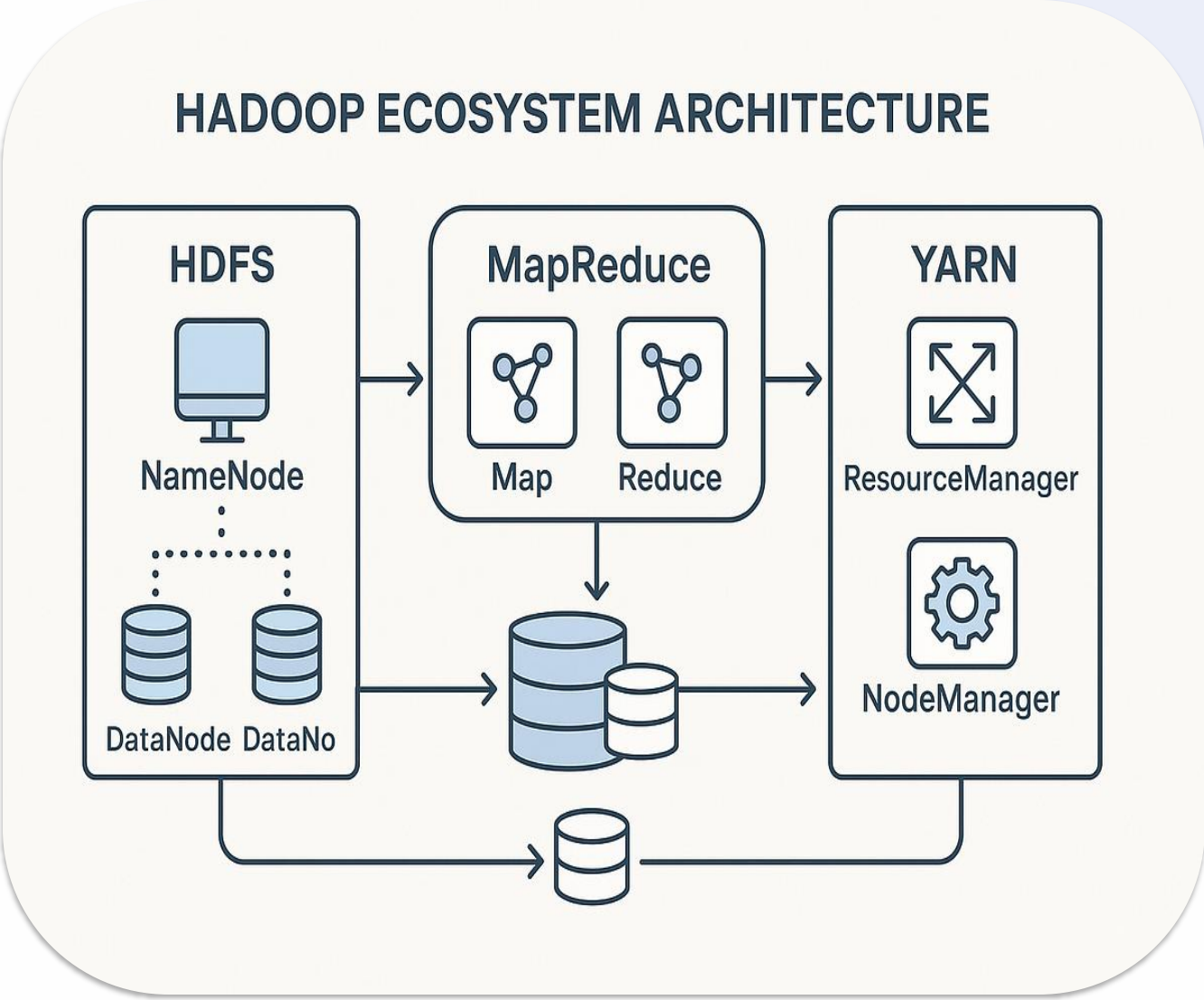
Introduction to HDFS

What is HDFS?

- Distributed file system for Hadoop
- Stores large datasets across nodes
- High fault tolerance & reliability
- Scalable to petabytes of data

Key Characteristics

- **Master-Slave Architecture**
- **Data Replication** (3x by default)
- **Rack Awareness** for efficiency
- **Commodity Hardware** support



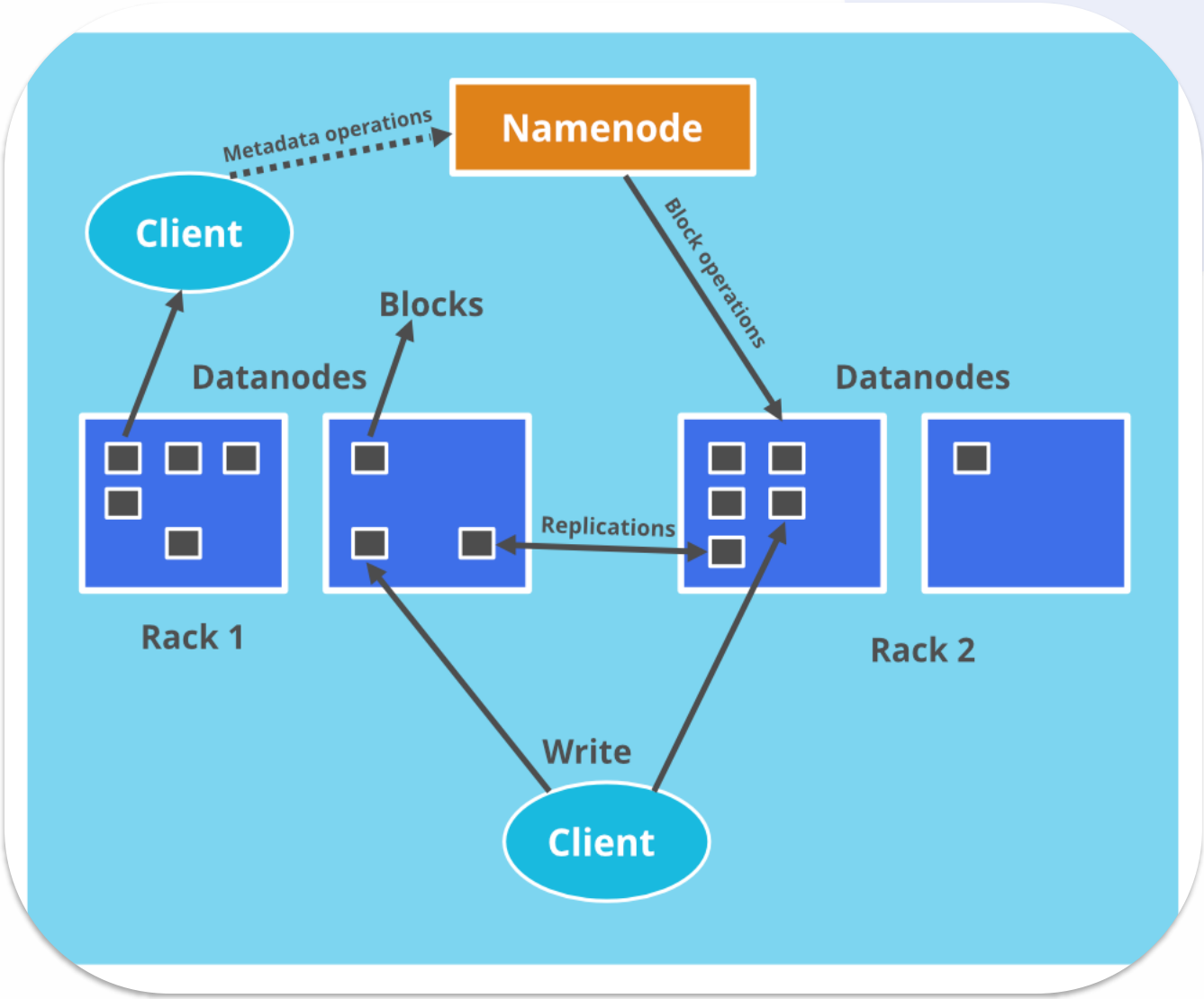
HDFS Architecture Overview

Architecture Type

- Master-Slave design
- Single Master Node
- Multiple Slave Nodes

Key Components

- **NameNode:** Master server
- **DataNode:** Slave servers
- **Client:** Application interface



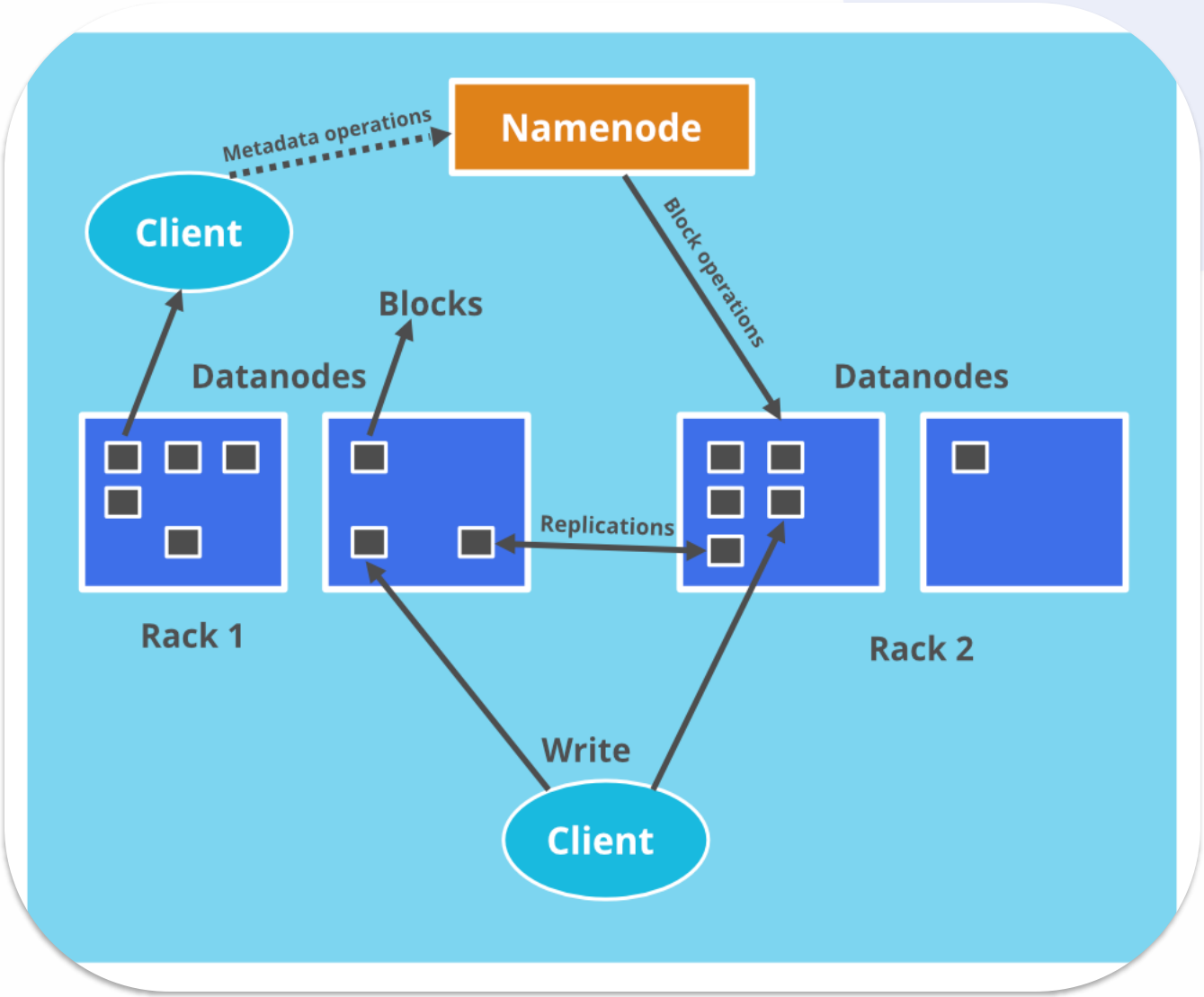
HDFS NameNode

Role & Responsibilities

- Manages file system namespace
- Stores metadata in RAM
- Handles client requests
- Coordinates DataNodes

Key Functions

- **Metadata Management**
- **File Operations** (create, delete, rename)
- **Block Mapping**
- **Heartbeat Monitoring**



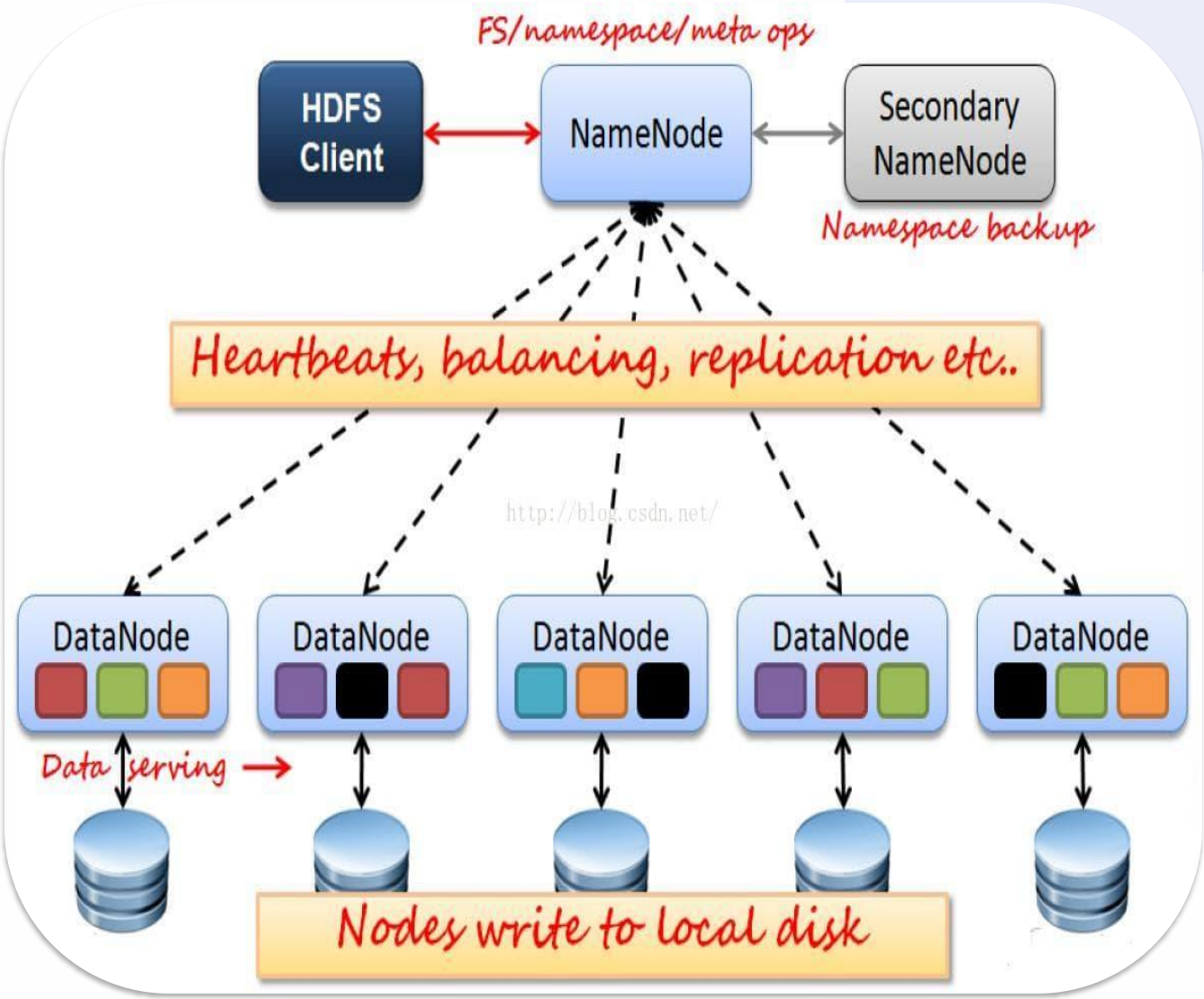
HDFS DataNode

Role & Responsibilities

- Stores actual data blocks
- Manages block replicas
- Serves read/write requests
- Sends heartbeats to NameNode

Key Functions

- **Block Storage** on local disks
- **Data Block Management**
- **Health Reporting** via heartbeats
- **Block Reports** to NameNode



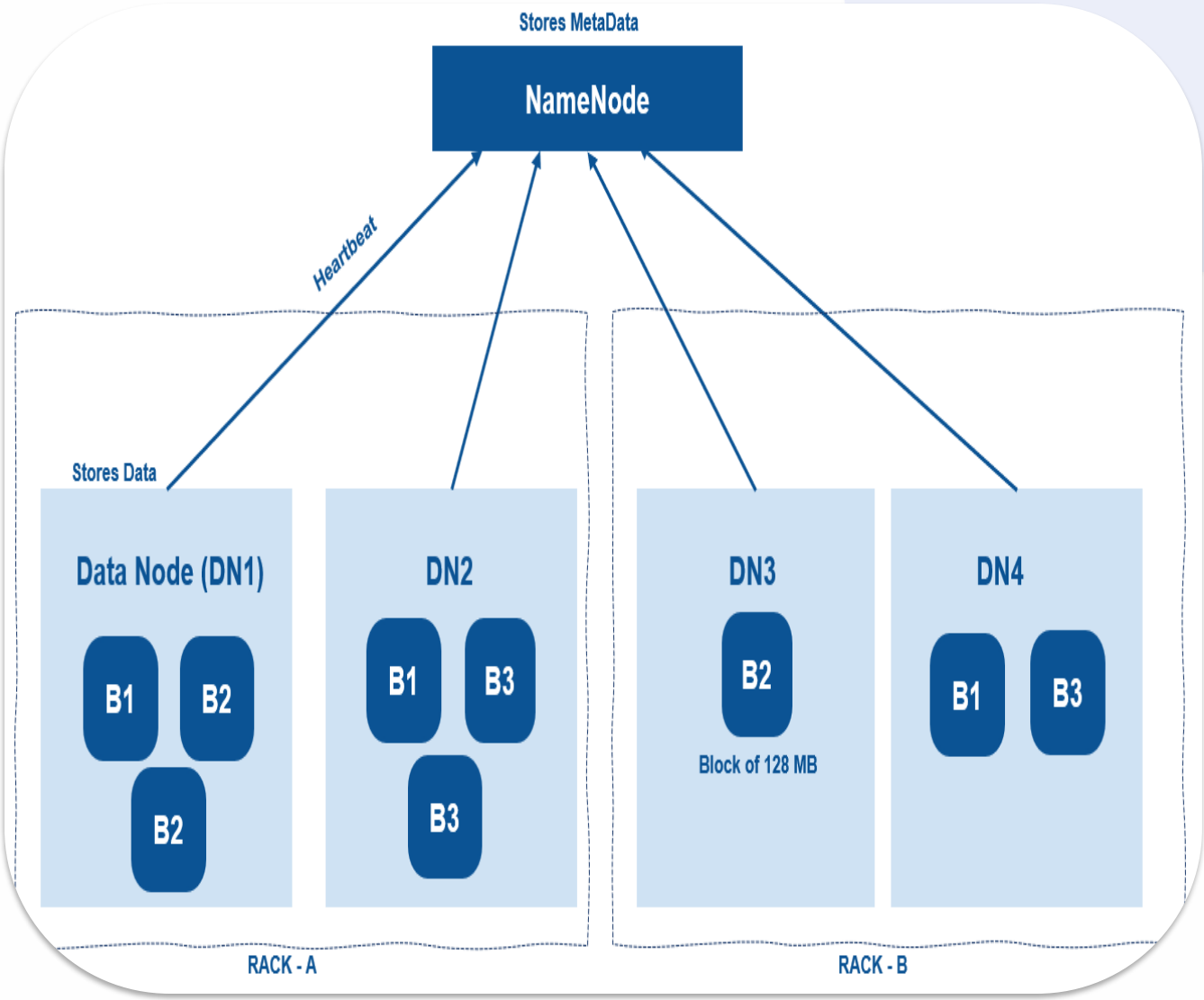
HDFS Data Replication

Replication Factor

- Default: 3 replicas
- Configurable per file
- First replica: local rack
- Second: different rack

Replication Strategy

- **Rack Awareness** for fault tolerance
- **Pipeline** replication process
- **Automatic** replica recovery
- **Load Balancing** across nodes



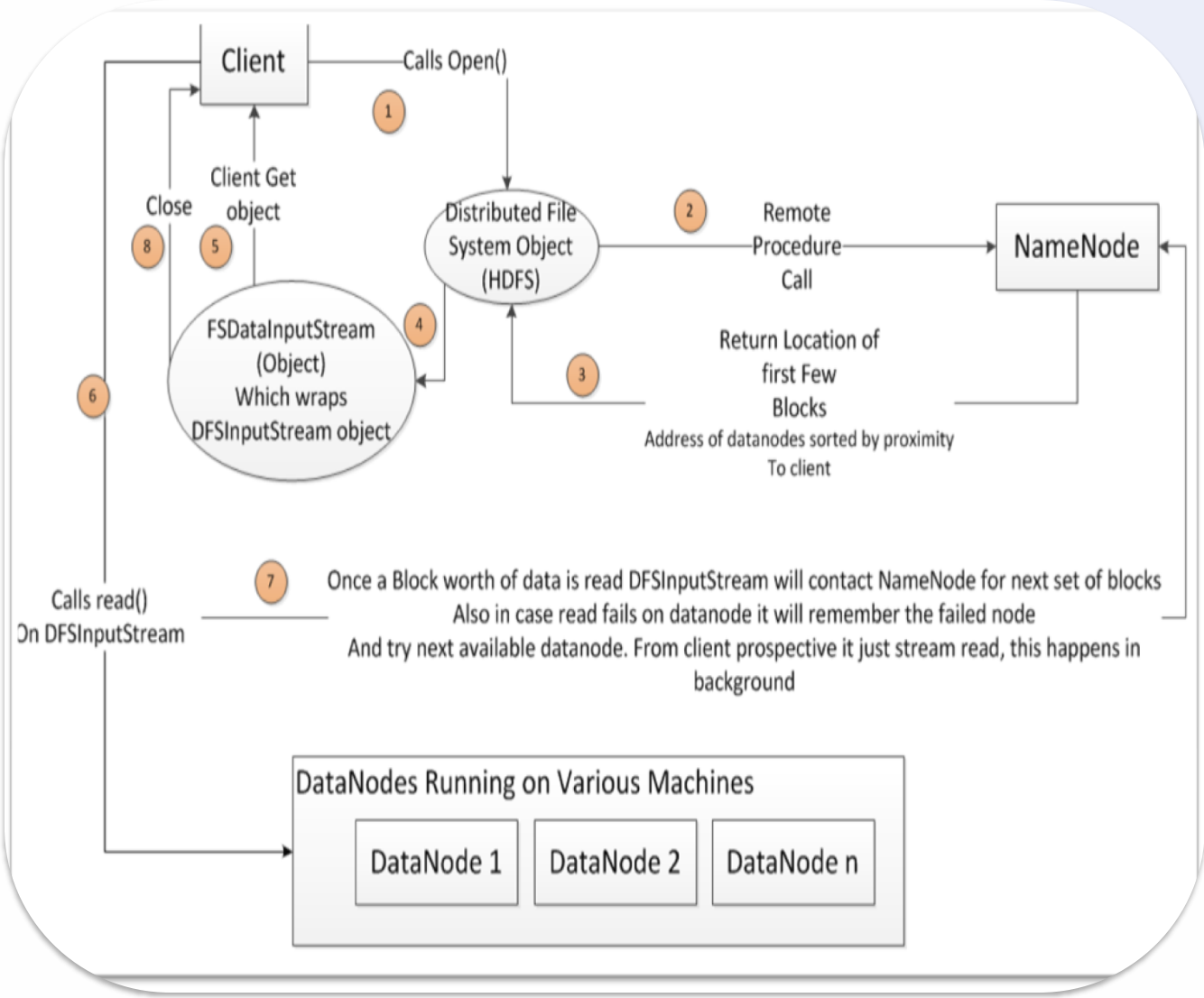
HDFS Read/Write Operations

Read Operation

- Client requests file from NameNode
- NameNode returns block locations
- Client reads from nearest DataNode
- Data verification via checksums

Write Operation

- **Create File** via NameNode
- **Pipeline** replication to DataNodes
- **Block Placement** strategy applied
- **Acknowledgments** sent back



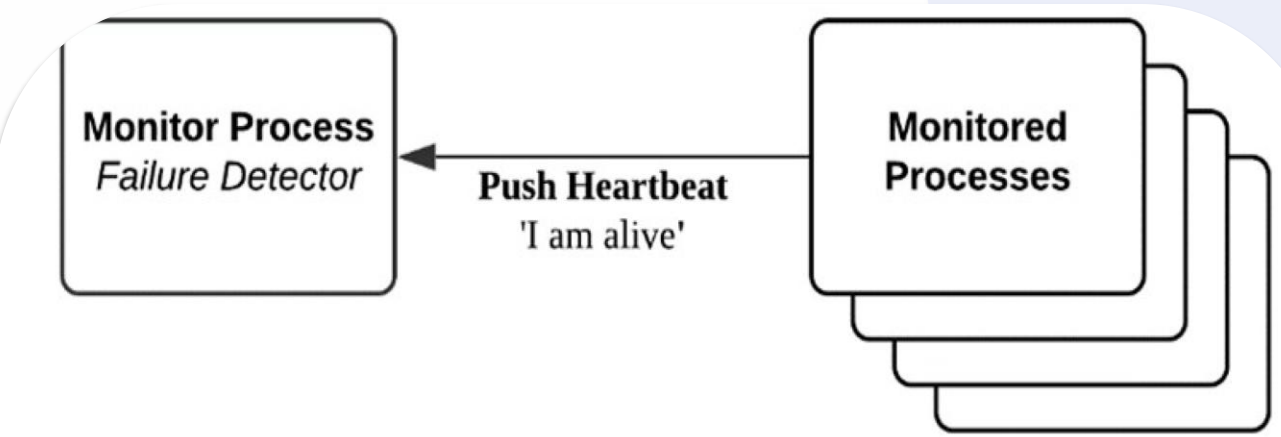
HDFS Fault Tolerance and Recovery

Heartbeat Mechanism

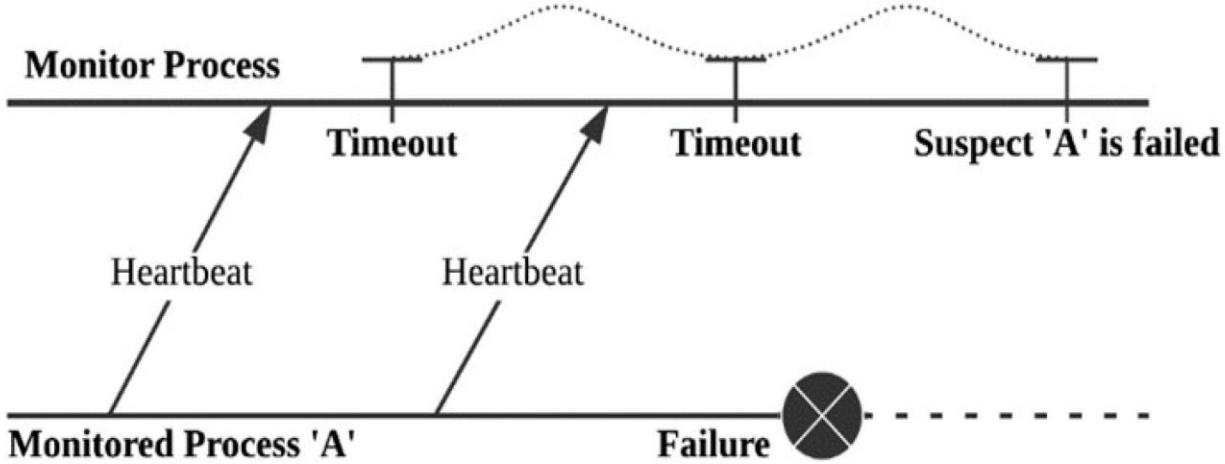
- DataNodes send heartbeats every 3 seconds
- NameNode monitors node health
- Timeout after 10 minutes of silence

Failure Recovery

- **Automatic** replica creation
- **Re-balancing** across cluster
- **Data integrity** checks via checksums
- **High availability** configurations



(a) Fault detection based on push model.



Heartbeat messages exchange between the monitor and monitored proc

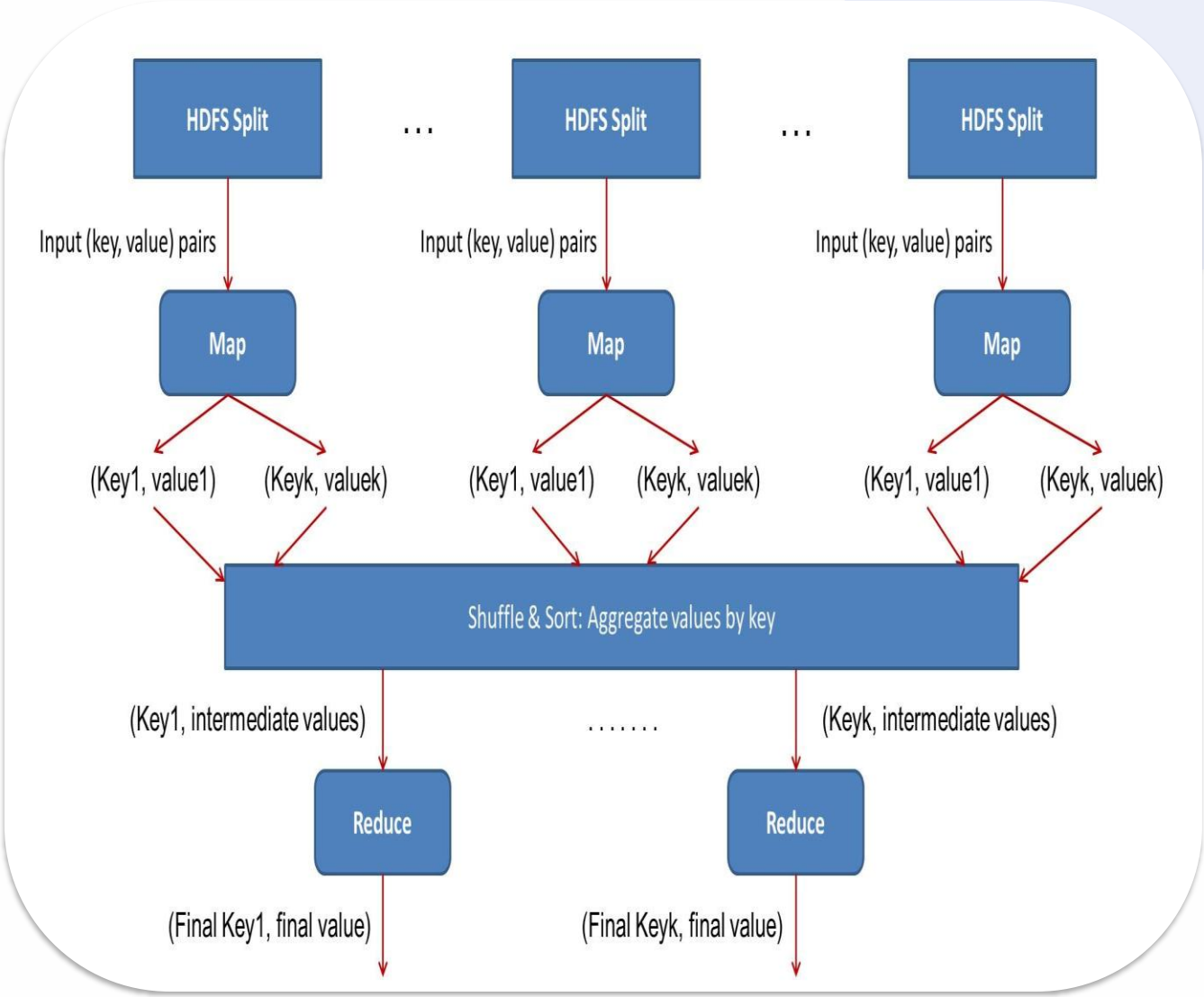
Mind Map of HDFS Architecture

Core Components

- **NameNode** - Master server
- **DataNode** - Slave servers
- **Secondary NameNode** - Checkpoint
- **Client** - User interface

Key Features

- **Fault Tolerance** via replication
- **Rack Awareness** optimization
- **High Throughput** data access
- **Scalability** to petabytes



Recap and Summary

Key Takeaways

- Master-Slave architecture design
- NameNode manages metadata
- DataNodes store actual data
- Replication ensures reliability

Learning Outcomes

- **Understand** HDFS components
- **Explain** replication mechanisms
- **Analyze** fault tolerance
- **Apply** in real-world scenarios

- NameNode: Metadata Management
- DataNode: Block Storage
- Client: User Interface
- Replication: 3x Default

Fault Tolerance

High Throughput

Scalability

Thank You!

Questions & Discussion