Big Data Hadoop and Spark Developer

Hadoop Architecture, Distributed Storage (HDFS), and YARN

Learning Objectives

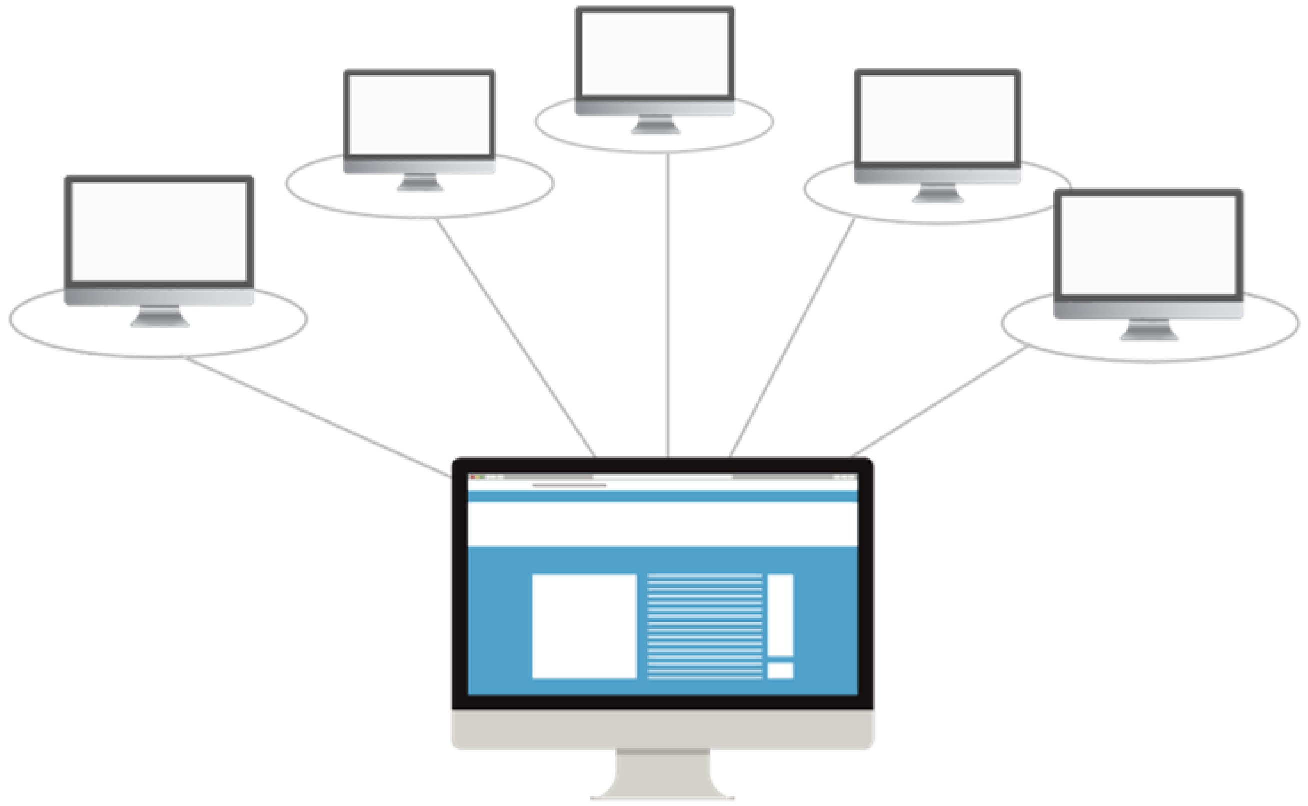
By the end of this lesson, you will be able to:

- Understand how Hadoop Distributed File System (HDFS) stores data across a cluster
- Illustrate the HDFS architecture and its components
- Demonstrate the use of HDFS Command Line Interface (CLI)
- Illustrate the YARN architecture and its components
- Demonstrate how to use Hue, YARN Web UI, and the YARN command to monitor the cluster

Hadoop Distributed File System (HDFS)

What Is HDFS?

HDFS is a distributed file system that provides access to data across Hadoop clusters.



Manages and supports analysis of very large volumes of Big Data

Challenges of Traditional Systems

In the traditional system, storing and retrieving volumes of data had three major issues:

Cost	Speed	Reliability
$10,000 to $14,000 per terabyte	Search and analysis is time-consuming	Fetching data is difficult

Need for HDFS

HDFS resolves all the three major issues of the traditional file system.

Cost

Zero licensing and support costs

Reliability

HDFS copies the data multiple times

Speed

Hadoop clusters read/write more than one terabyte of data in a second

Regular File System vs. HDFS

A patron gifts his popular books collection to a college library.

The librarian decides to arrange the books on a small rack.

Also, he distributes multiple copies of each book on other racks.

Regular File System vs. HDFS

Regular File System

HDFS

Size of Data

51 bytes—small block of data

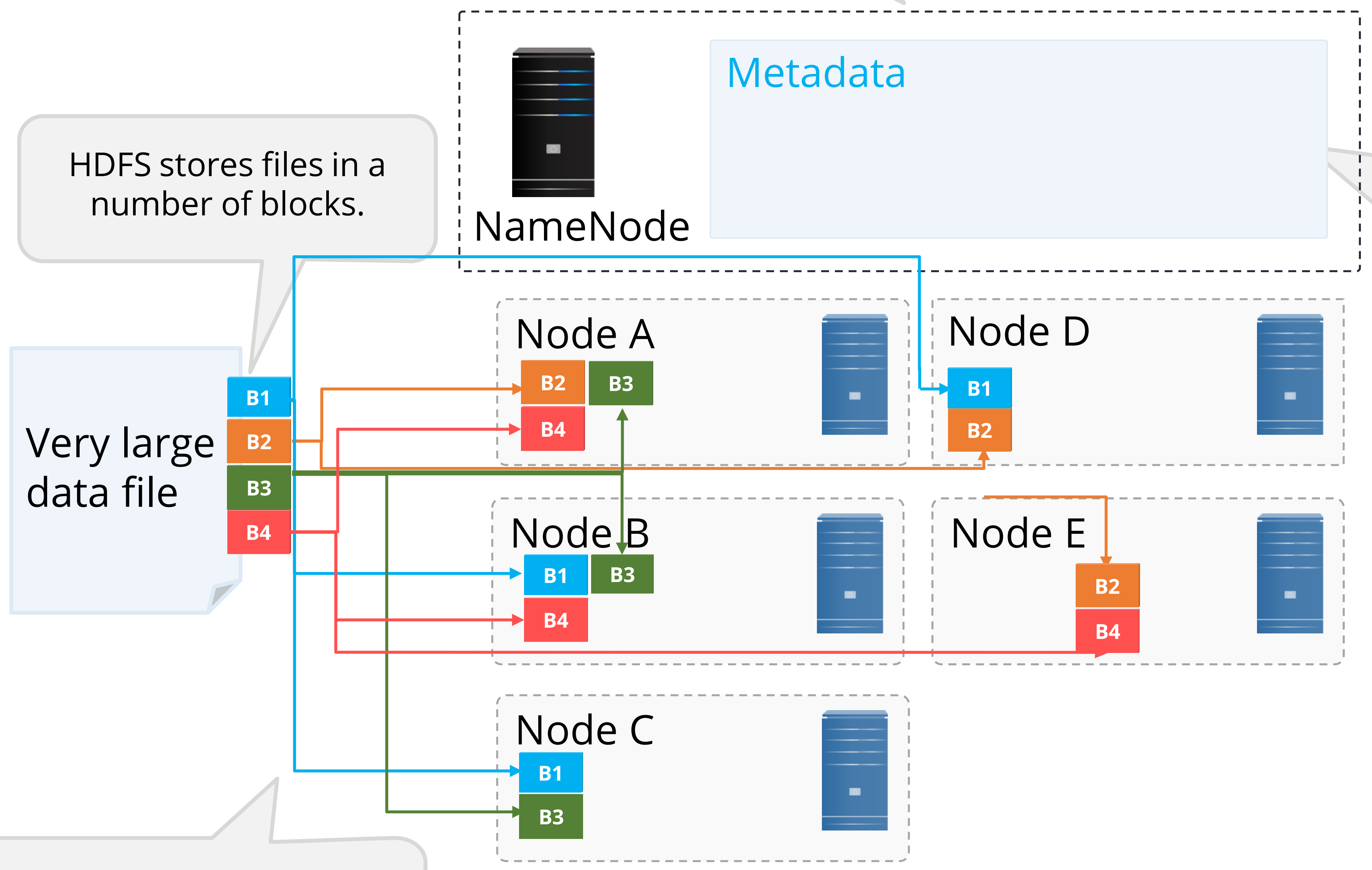
128 MB—large block of data

Access to Large Data

Suffers from disk I/O problems primarily because of multiple seek operations

Reads huge data sequentially in a single seek operation

HDFS Storage



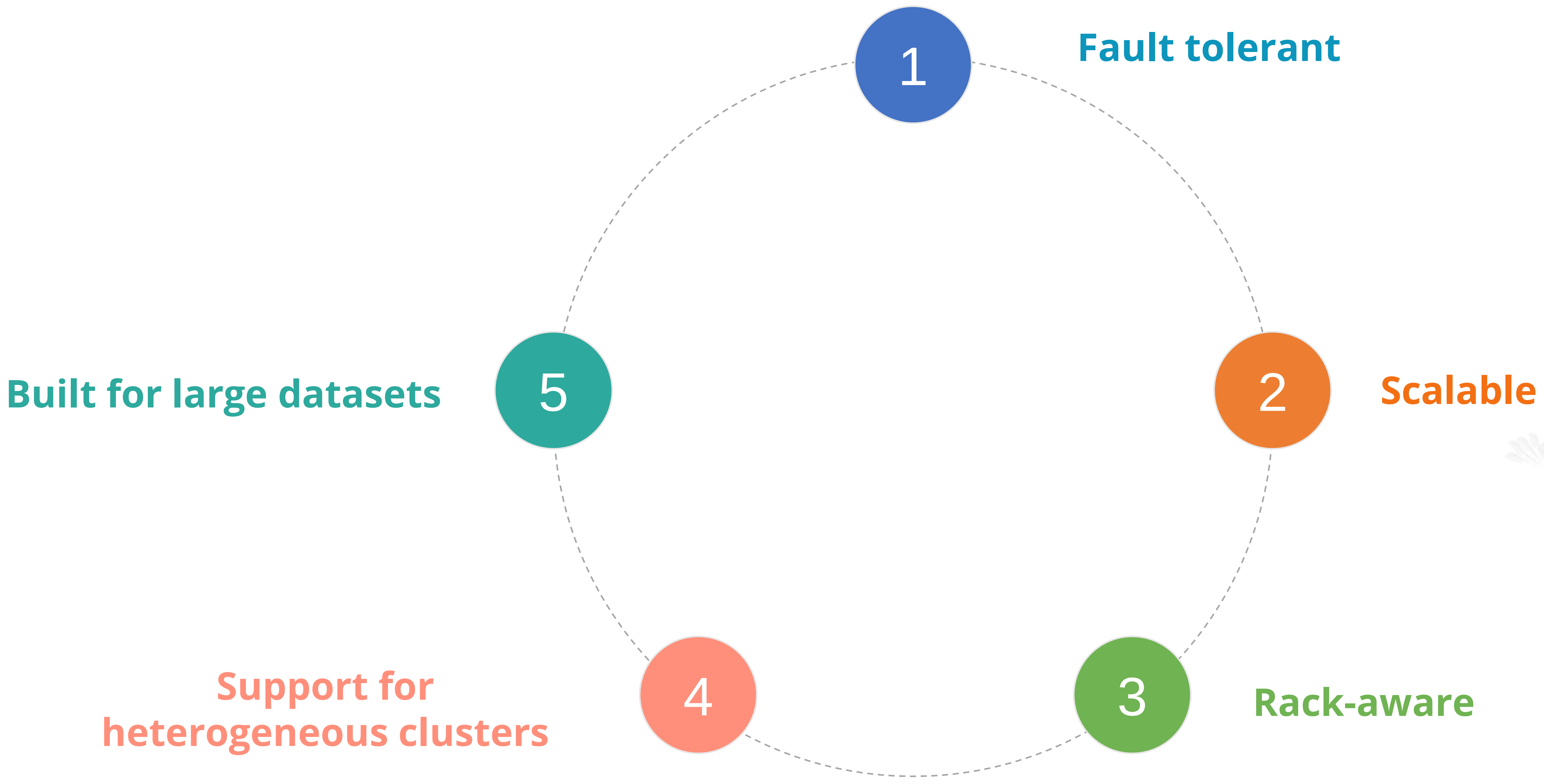
Data is divided into 128 MB per block.

HDFS stores files in a number of blocks.

Metadata keeps information about the block and its replication.
It is stored in NameNode.

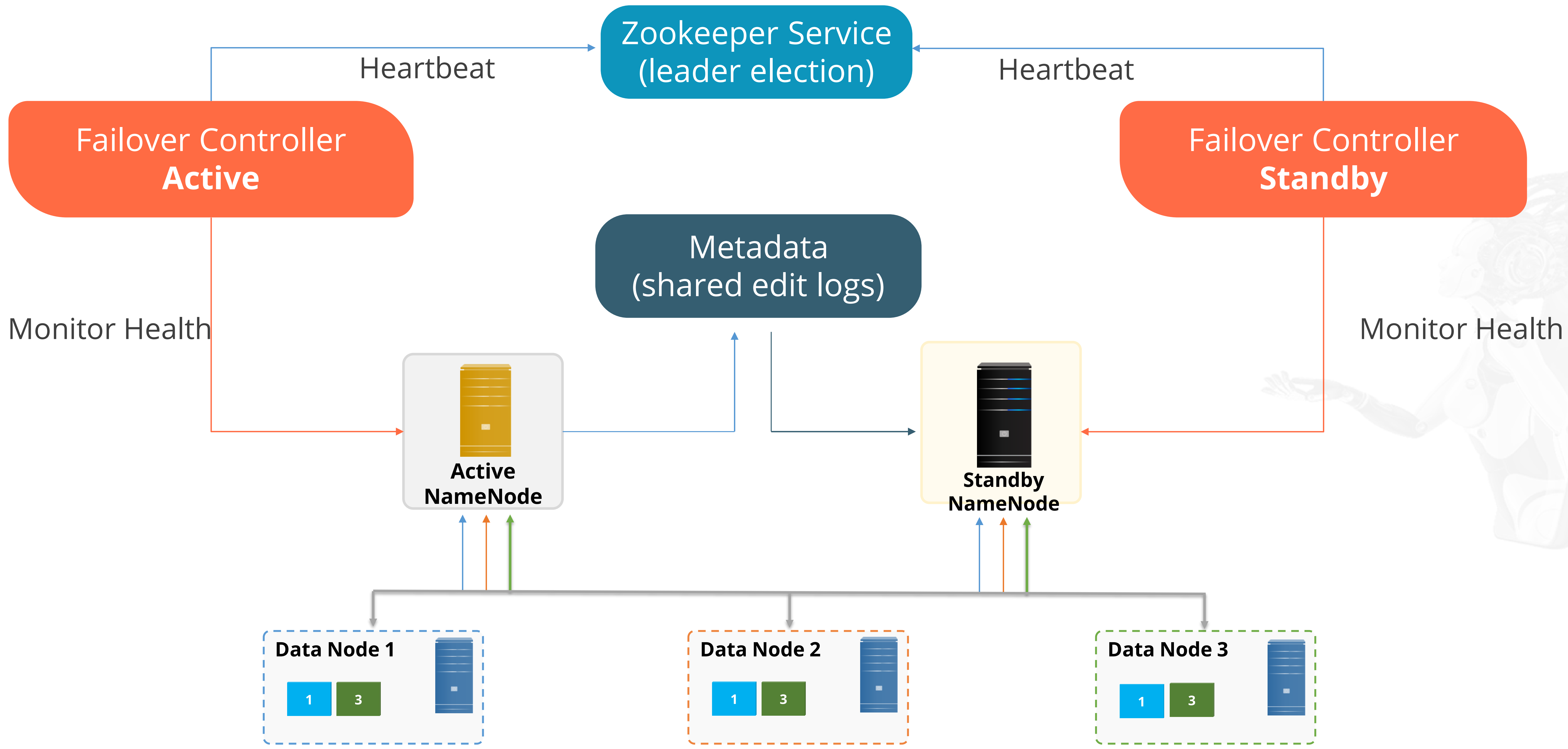
Each block is replicated to a few separate computers.

Characteristics of HDFS



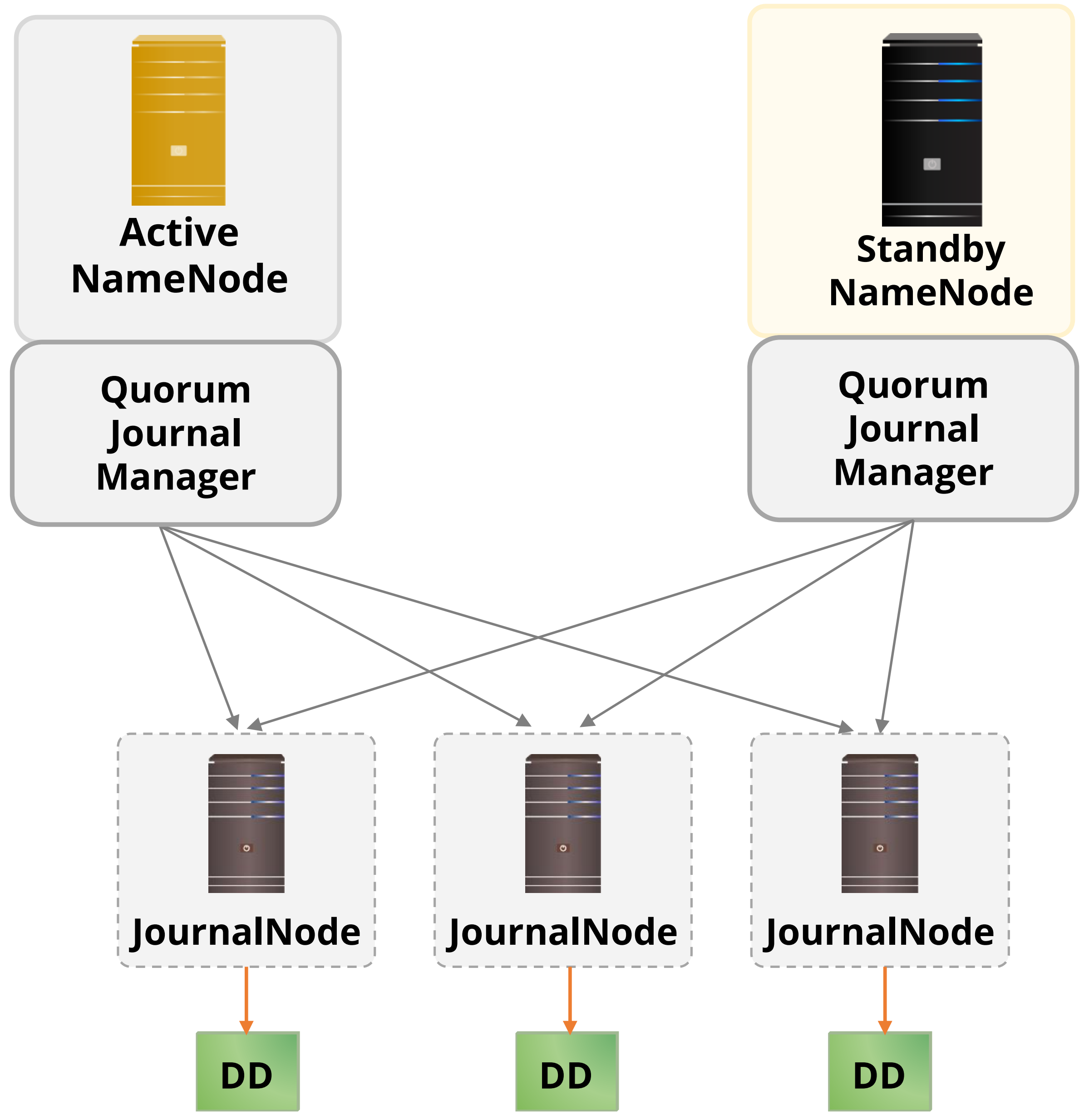
HDFS Architecture and Components

HDFS High Availability Architecture

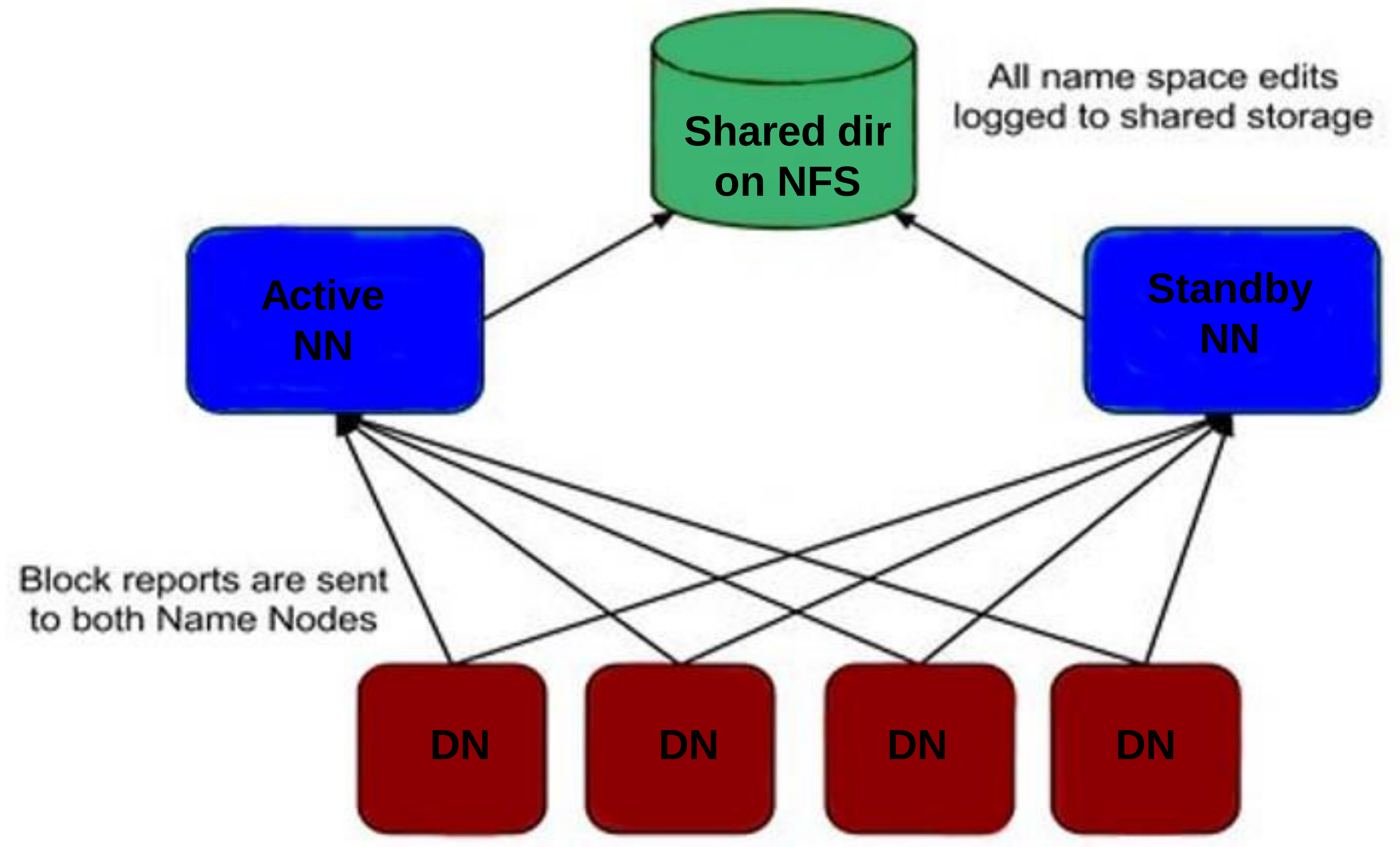


Types of HDFS HA Architecture

Quorum-based Storage

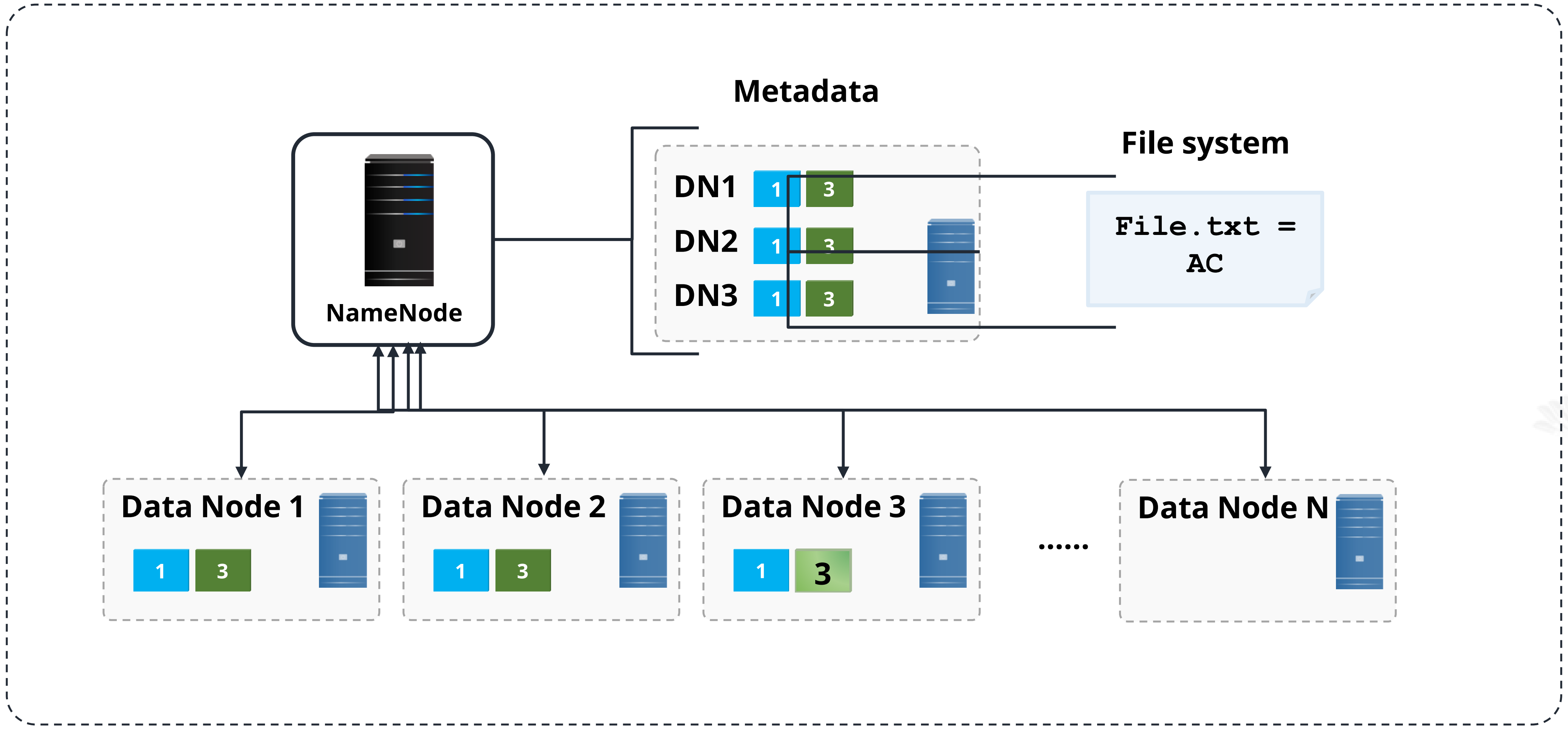


Shared storage using NFS


Shared dir on NFS
All name space edits logged to shared storage
Active NN
Standby NN
Block reports are sent to both Name Nodes
DN
DN
DN
DN

HDFS Component: NameNode

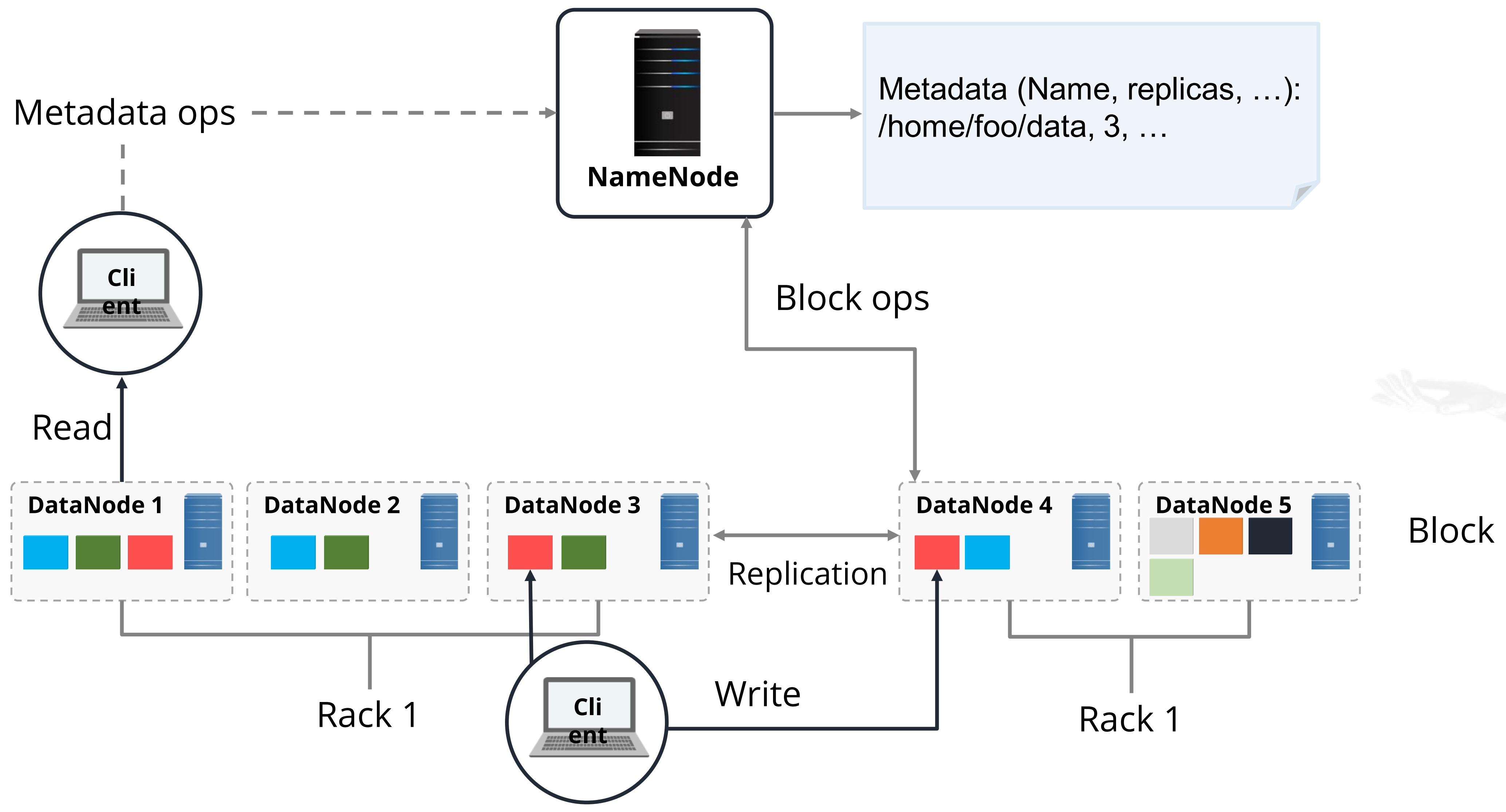
The HDFS components comprise different servers like NameNode, DataNode, and Secondary NameNode.



The NameNode server—core component of an HDFS cluster

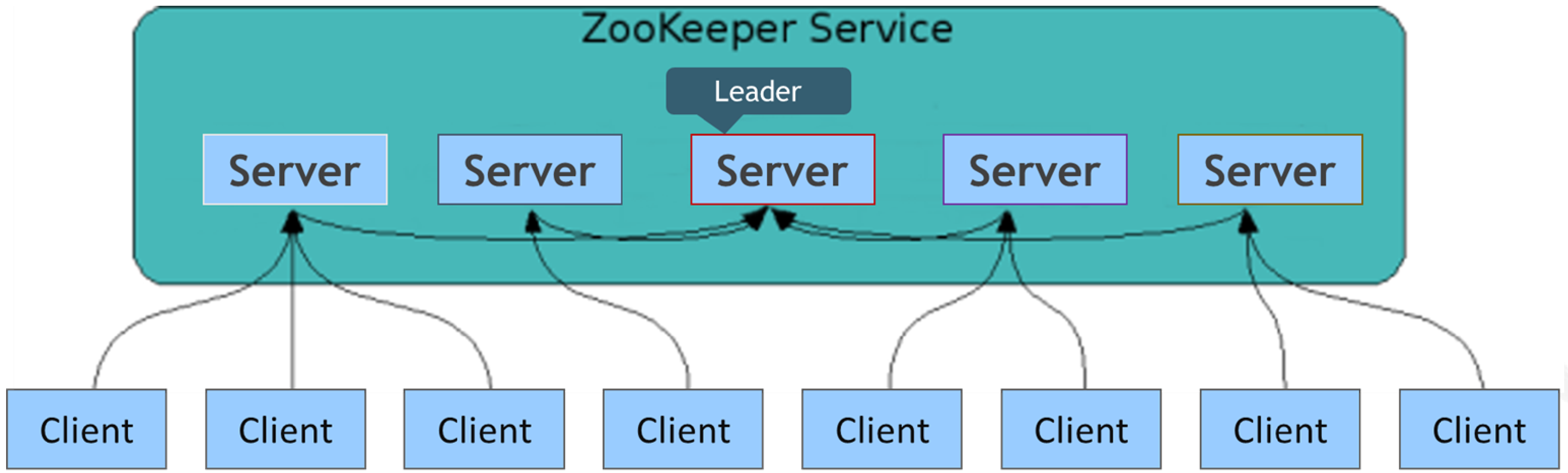
HDFS Component: DataNode

The DataNode is a multiple instance server.
The DataNode servers are responsible for storing and maintaining the data blocks.



HDFS Component: Zookeeper

ZooKeeper allows distributed processes to coordinate with each other through a shared hierarchical namespace.



The ZooKeeper implementation is simple and replicated which puts a premium on high performance, high availability, and a strictly ordered access.

HDFS Component: Zookeeper

Automatic HDFS failover relies on ZooKeeper for the following:

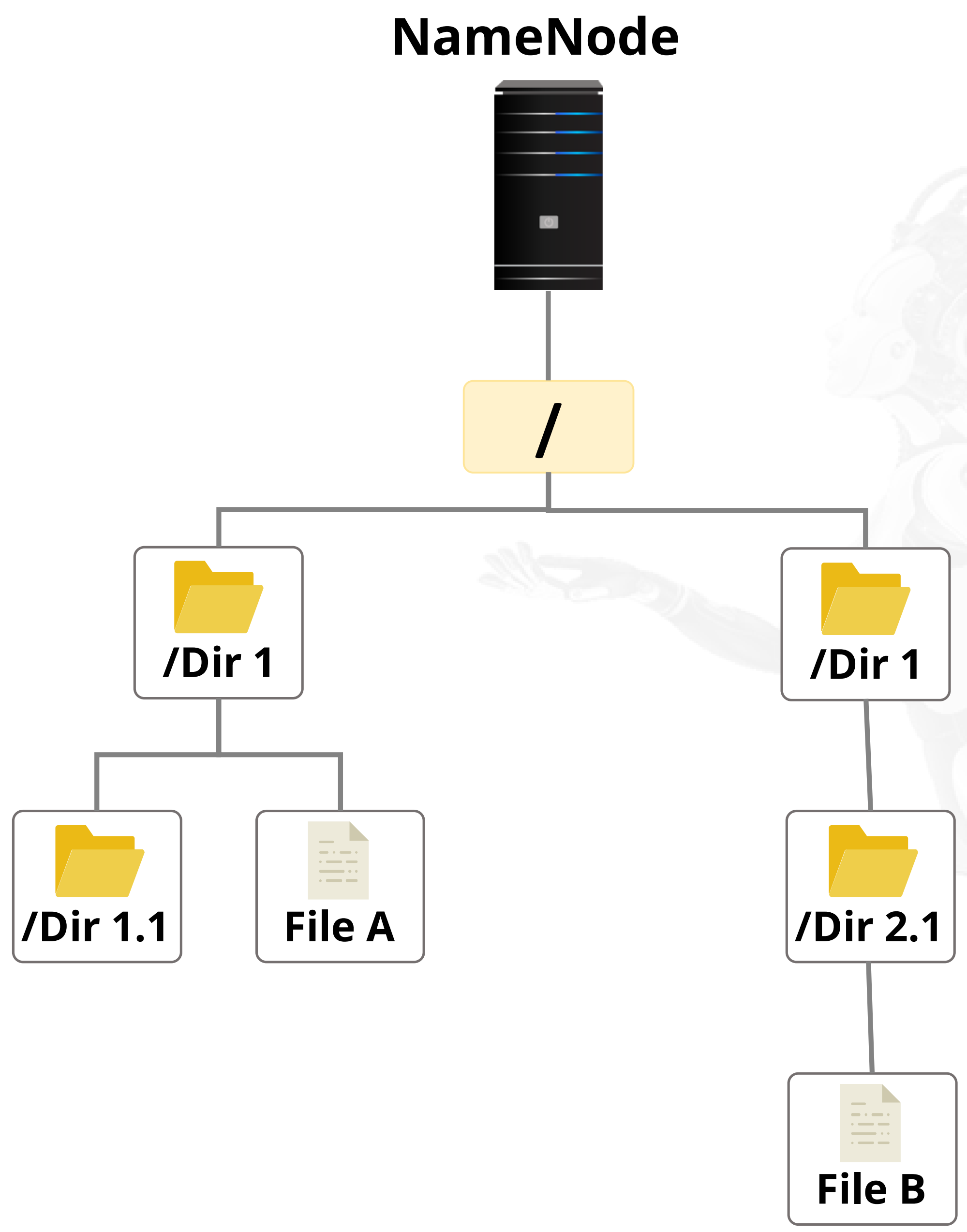
Failure detection

Active NameNode election

HDFS Component: File System Namespace

The characteristics of HDFS file system are as follows:

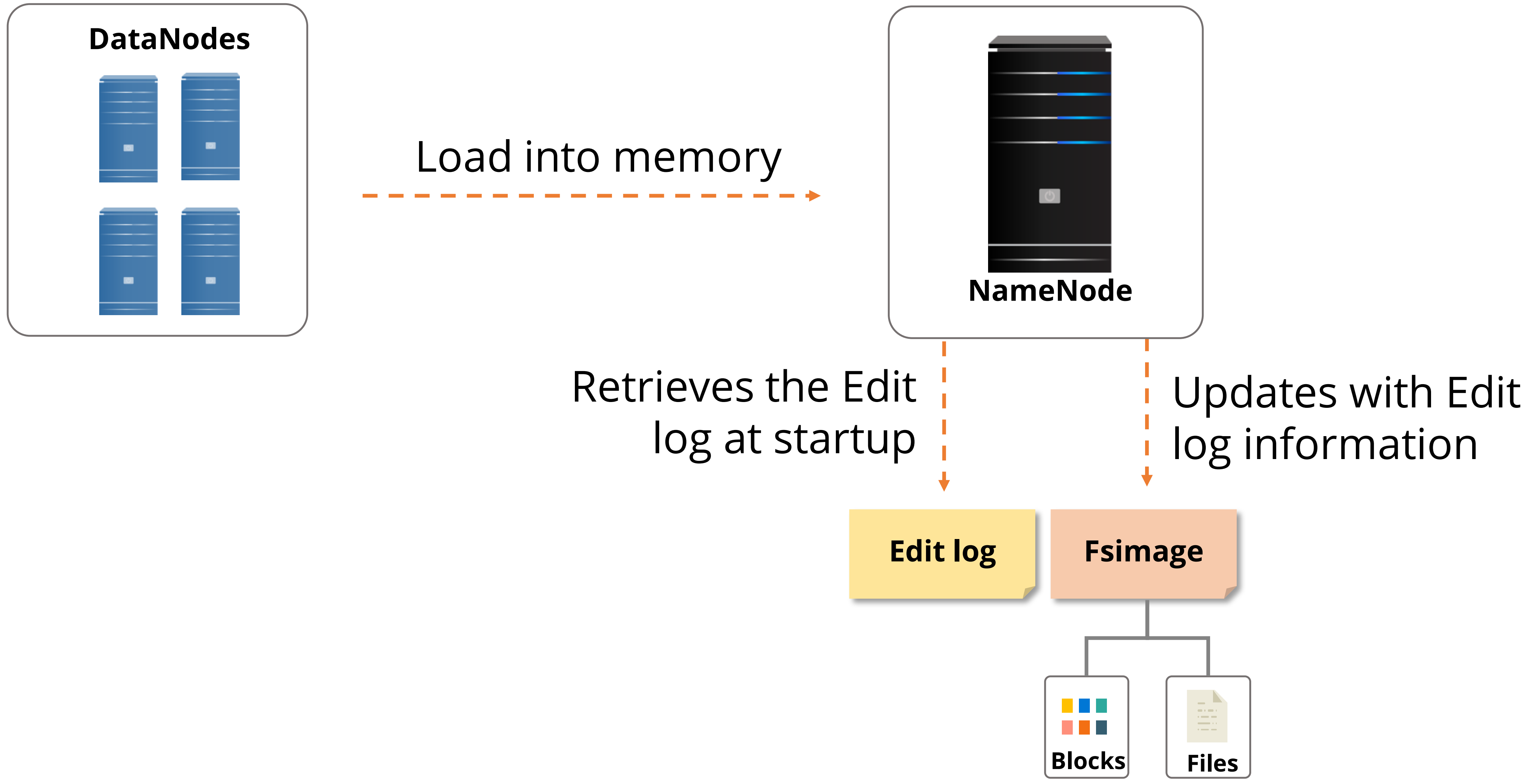
- Allows user data to be stored in files
- Follows a hierarchical file system with directories and files
- Supports operations such as create, remove, move, and rename



NameNode Operation

The NameNode maintains two persistent files:

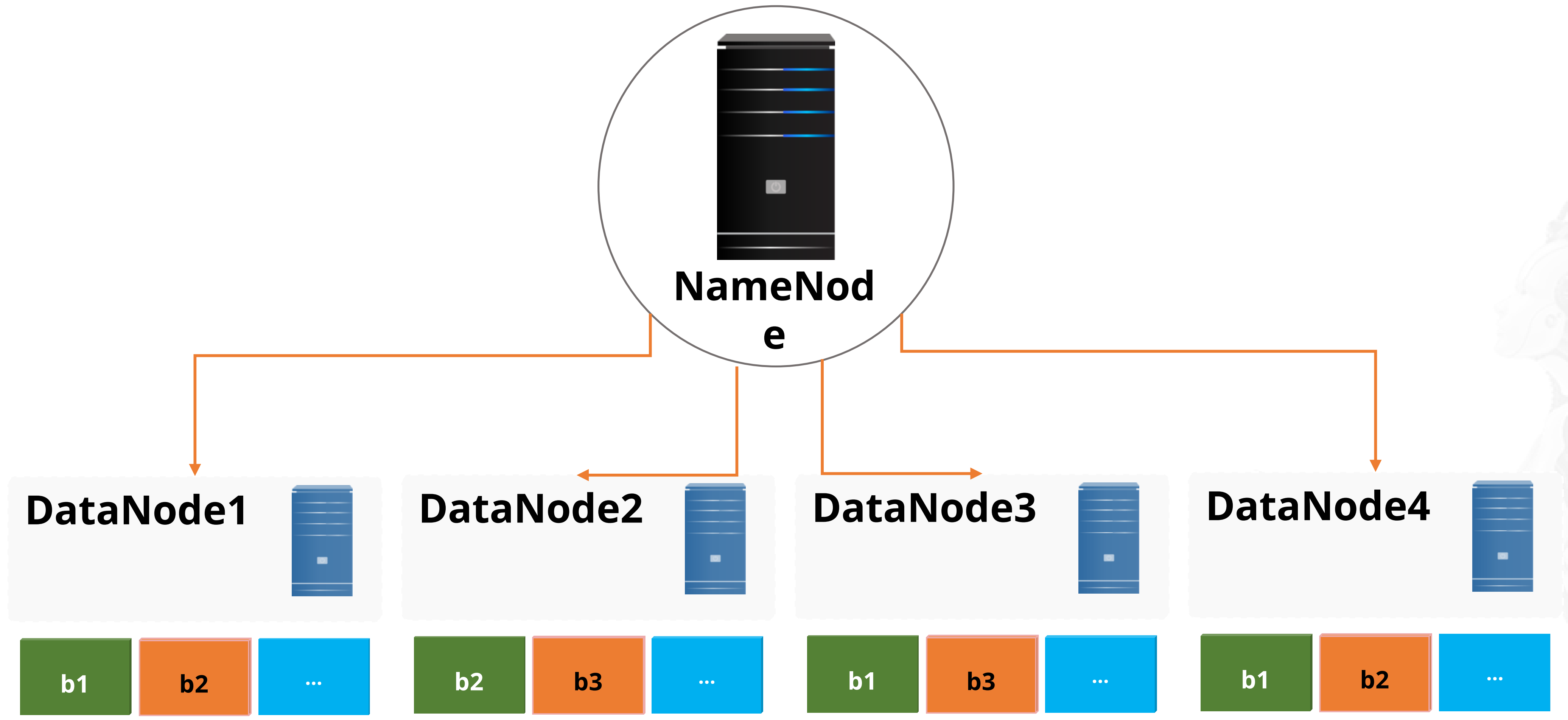
- A transaction log called an Edit Log and
- A namespace image called an FsImage


DataNodes
Load into memory
NameNode
Retrieves the Edit log at startup
Updates with Edit log information
Edit log
Fsimage
Blocks
Files

Data Block Split

Each file is split into one or more blocks which are stored and replicated in DataNodes.

A file split into blocks

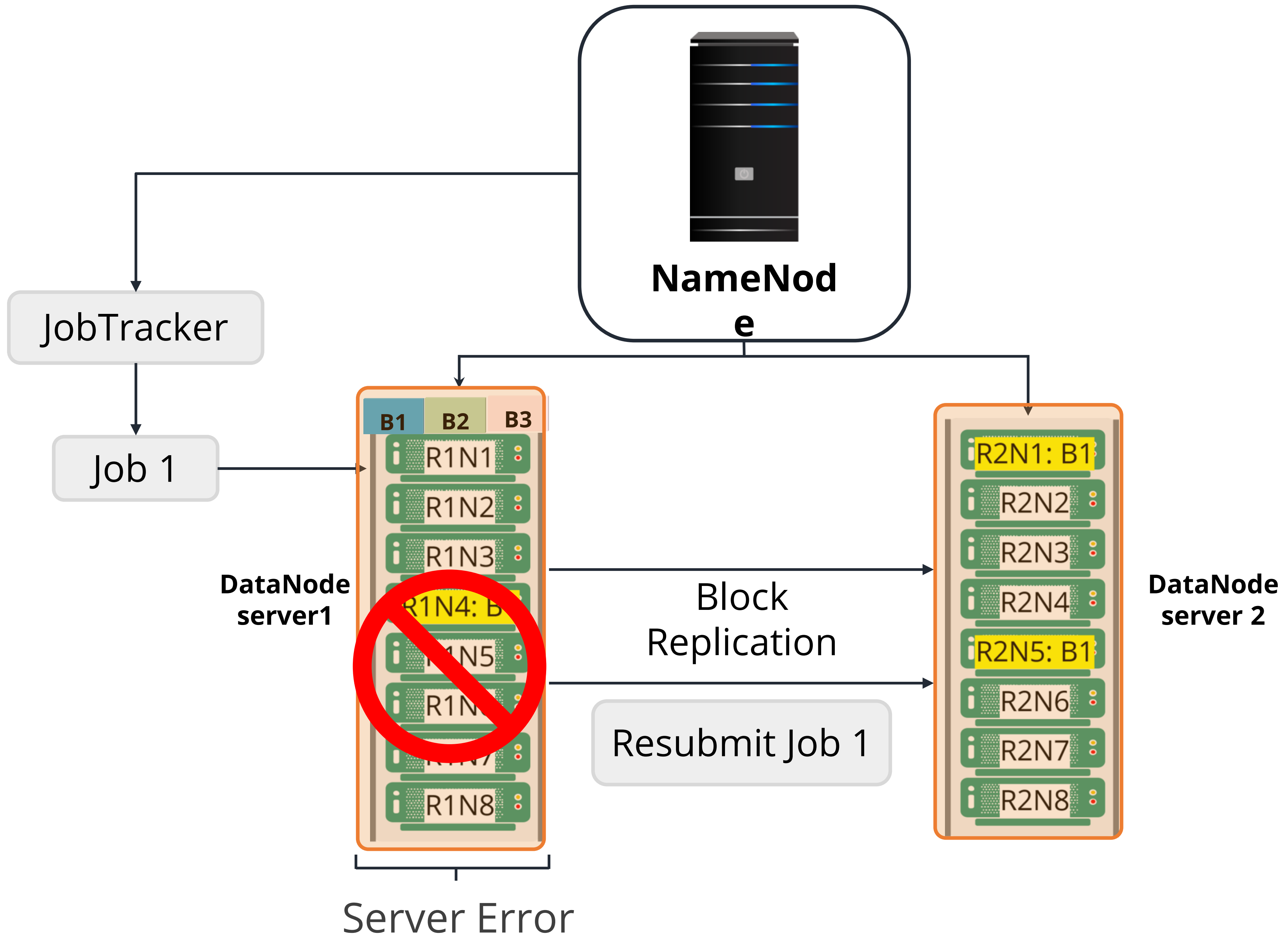


DataNodes managing blocks

The data block approach provides simplified replication, fault-tolerance, and reliability.

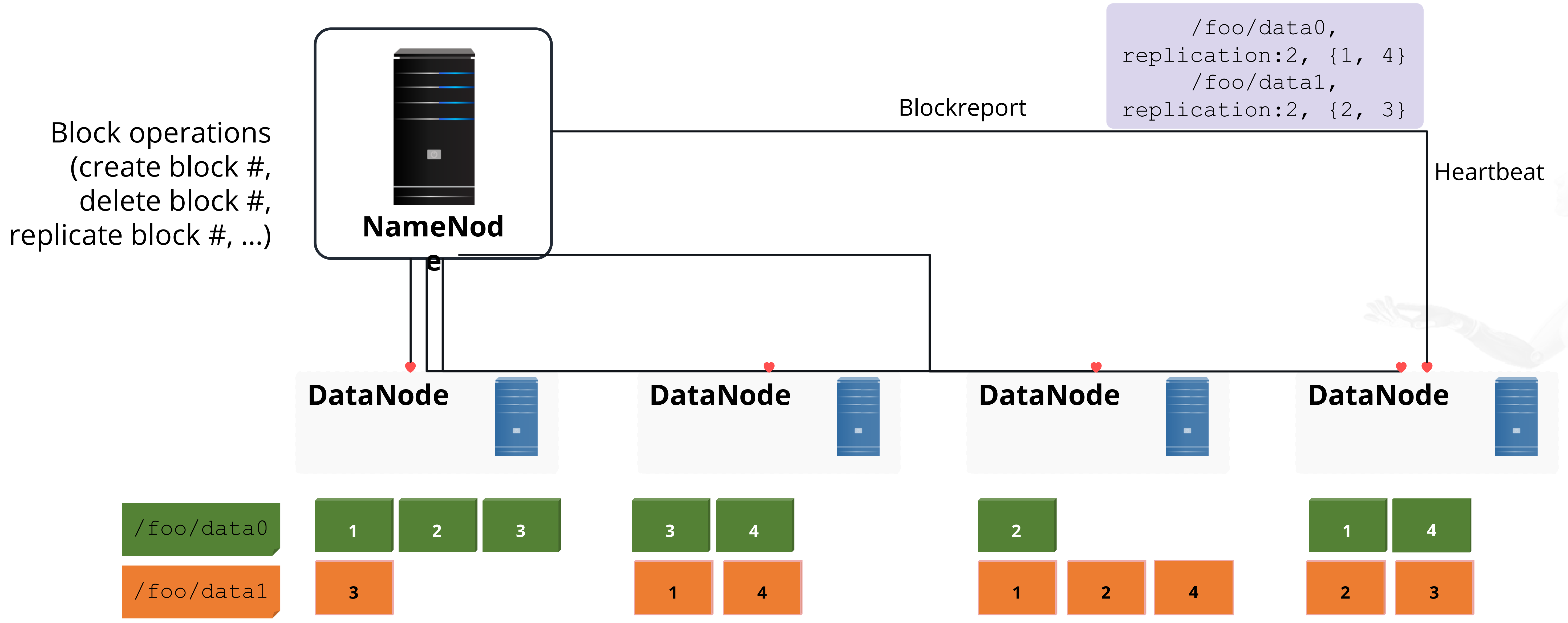
Block Replication Architecture

HDFS represents the unstructured data in the form of data blocks.
It performs block replication on multiple DataNodes.



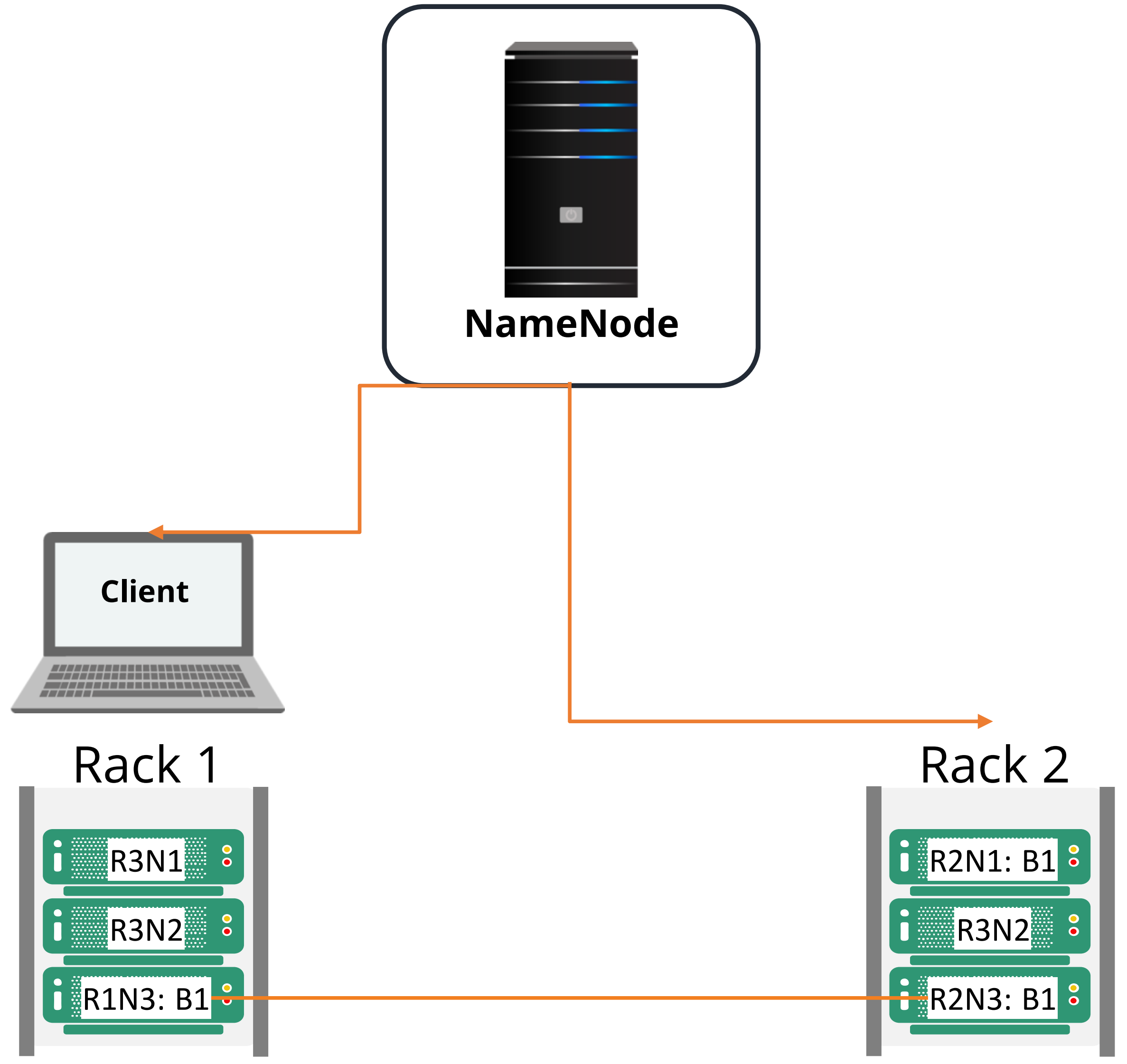
Replication Method

Each file is split into a sequence of blocks.
Except for the last one, all blocks in the file are of the same size.


Block operations (create block #, delete block #, replicate block #, ...)
NameNode
Blockreport
/foo/data0, replication:2, {1, 4}
/foo/data1, replication:2, {2, 3}
Heartbeat
DataNode
DataNode
DataNode
DataNode
/foo/data0
/foo/data1
1 2 3
3
3 4
1 4
2
1 2 4
1 4
2 3

Data Replication Topology

One of the suggested replication topology is as follows:



HDFS Access

HDFS provides the following access mechanisms:

Java API for applications

Python access and C language wrapper for non-Java applications

Web GUI utilized through an HTTP browser

FS shell for executing commands on HDFS

HDFS Command Line

Following are a few basic command lines of HDFS:

Copy file simplilearn.txt from local disk to the user's directory in HDFS –This will copy the file to /user/username/simplilearn.txt	`$ hdfs dfs -put simplilearn.txt simplilearn.txt`
Display a list of the contents of the directory path provided by the user, showing the names, permissions, owner, size, and modification date for each entry	`$hdfs dfs –ls /user/simpl/test`
Create a directory called test under the user's home directory	`$hdfs dfs –mkdir /user/simpl/test`
Delete the directory testing and all its contents	`hdfs dfs -rm -r testing`

Assisted Practice

HDFS Command Line

Duration: 15 mins

Problem Statement: In this demonstration, you will explore few basic command lines of HDFS .

Access: Click on the **Practice Labs** tab on the left side panel of the LMS. Copy or note the username and password that is generated. Click on the **Launch Lab** button. On the page that appears, enter the username and password in the respective fields, and click **Login**.

Unassisted Practice

HDFS Command Line

Duration: 15 mins

Problem Statement: Using command lines of HDFS, perform the below tasks:

- Create a directory named “Simplilearn“ in HDFS
- Transfer a sample text file from your local filesystem to HDFS directory
- List the files in HDFS directory
- Change the permissions on the file to read, write, and execute
- Remove the text file from HDFS directory

Access: Click on the **Practice Labs** tab on the left side panel of the LMS. Copy or note the username and password that is generated. Click on the **Launch Lab** button. On the page that appears, enter the username and password in the respective fields, and click **Login**.

Unassisted Practice

Steps to Perform

- Create a directory named “Simplilearn"
  hdfs dfs -mkdir Simplilearn

- Transfer a sample text file from your local filesystem to HDFS directory
  hdfs dfs -put /home/simplilearn_learner/test.txt Simplilearn

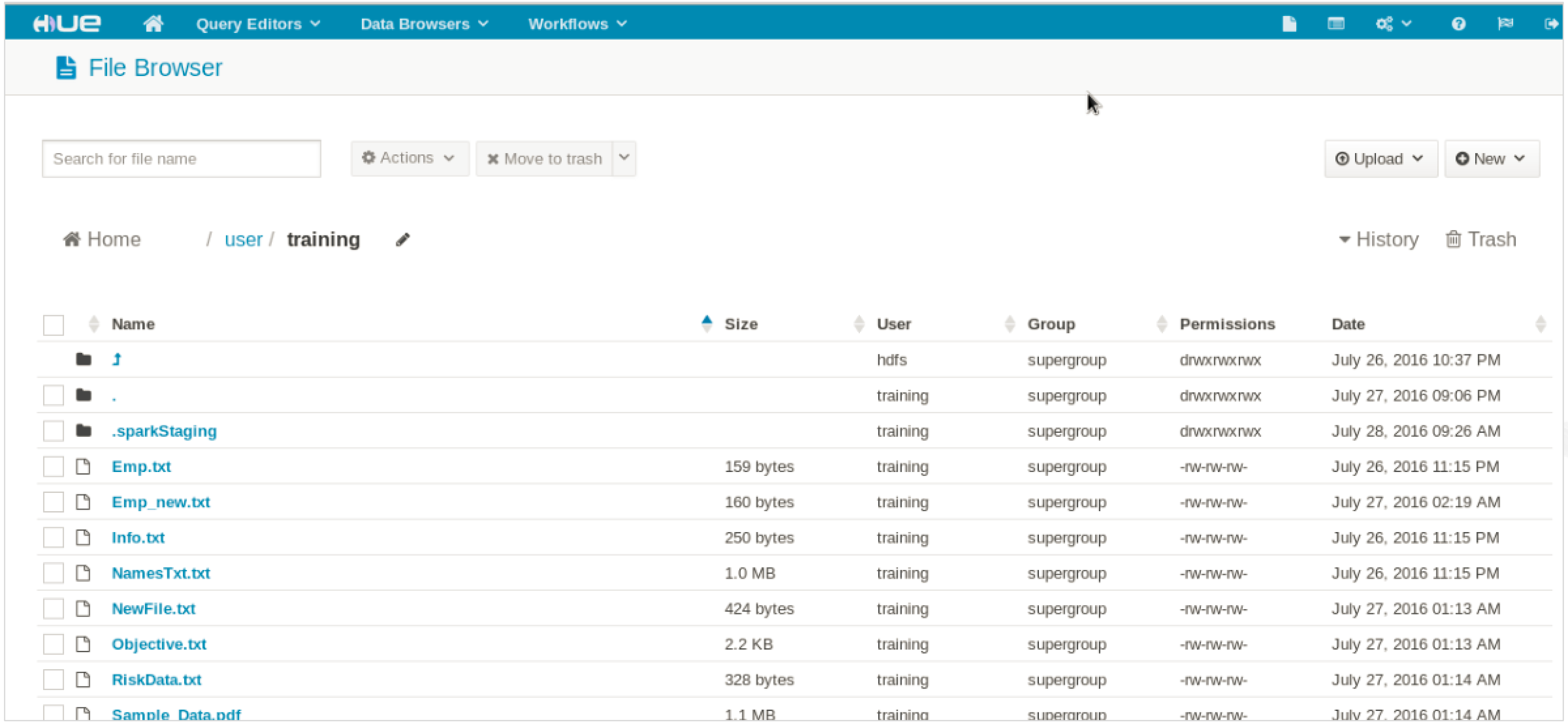
- List the files in HDFS directory
  hdfs dfs -ls Simplilearn

- Change the permissions on the file to read, write and execute
  hdfs dfs -chmod 700 Simplilearn/test.txt

- Remove the text file from HDFS directory
  hdfs dfs -rm -r Simplilearn/test.txt

Hue File Browser

The file browser in Hue lets you view and manage your HDFS directories and files.



Name	Size	User	Group	Permissions	Date
↥		hdfs	supergroup	drwxrwxrwx	July 26, 2016 10:37 PM
.		training	supergroup	drwxrwxrwx	July 27, 2016 09:06 PM
.sparkStaging		training	supergroup	drwxrwxrwx	July 28, 2016 09:26 AM
Emp.txt	159 bytes	training	supergroup	-rw-rw-rw-	July 26, 2016 11:15 PM
Emp_new.txt	160 bytes	training	supergroup	-rw-rw-rw-	July 27, 2016 02:19 AM
Info.txt	250 bytes	training	supergroup	-rw-rw-rw-	July 26, 2016 11:15 PM
NamesTxt.txt	1.0 MB	training	supergroup	-rw-rw-rw-	July 26, 2016 11:15 PM
NewFile.txt	424 bytes	training	supergroup	-rw-rw-rw-	July 27, 2016 01:13 AM
Objective.txt	2.2 KB	training	supergroup	-rw-rw-rw-	July 27, 2016 01:13 AM
RiskData.txt	328 bytes	training	supergroup	-rw-rw-rw-	July 27, 2016 01:14 AM
Sample_Data.pdf	1.1 MB	training	supergroup	-rw-rw-rw-	July 27, 2016 01:14 AM

Additionally, you can create, move, rename, modify, upload, download, and delete directories and files.

Assisted Practice

Hue File Browser

Duration: 15 mins

Problem Statement: In this demonstration, you will learn, how to access HDFS using Hue. You will also learn how to view and manage your HDFS directories and files using Hue.

Access: Click on the **Practice Labs** tab on the left side panel of the LMS. Copy or note the username and password that is generated. Click on the **Launch Lab** button. On the page that appears, enter the username and password in the respective fields, and click **Login**.

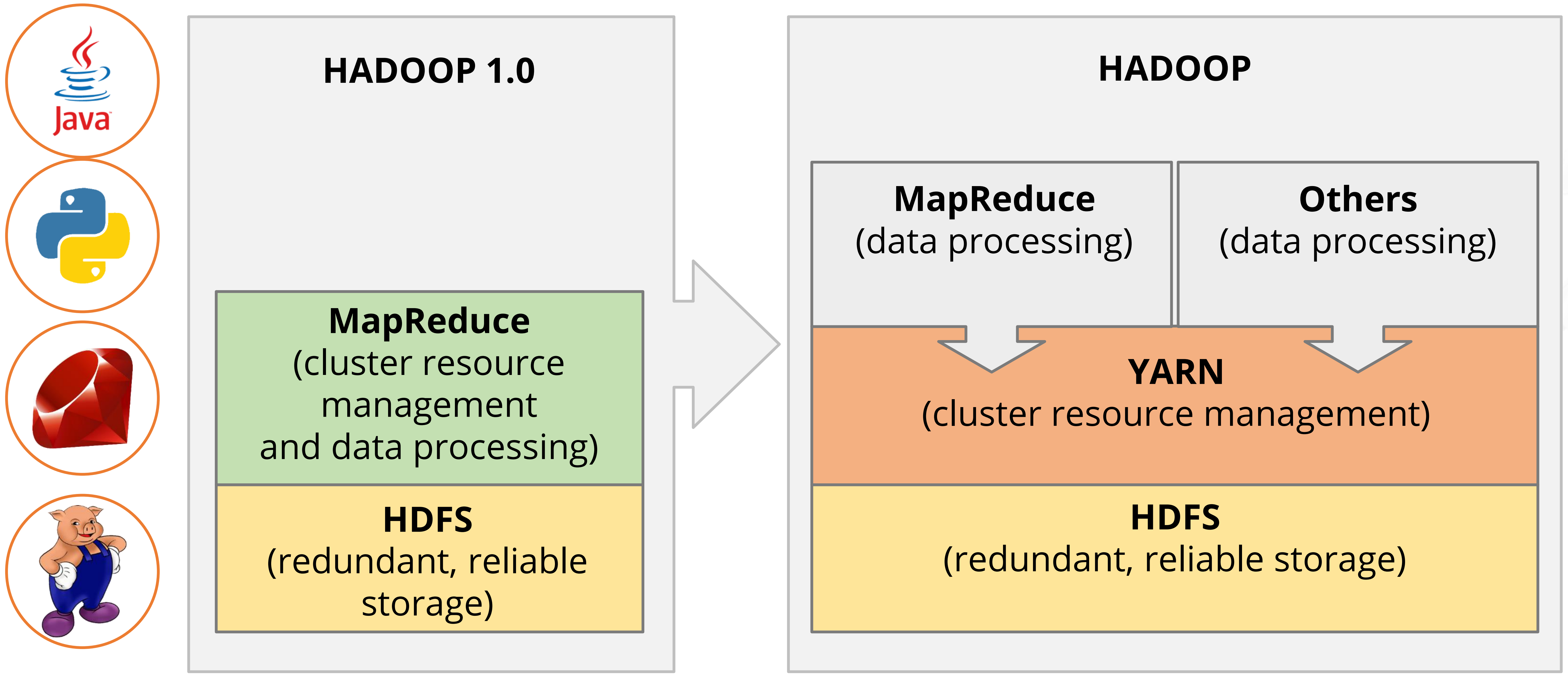
YARN: Introduction

What Is YARN?

YARN= Yet Another Resource Negotiator

- YARN is a resource manager
- Created by separating the processing engine and the management function of MapReduce
- Monitors and manages workloads, maintains a multi-tenant environment, manages the high availability features of Hadoop, and implements security controls

Need for YARN


Before 2012
Users could write MapReduce programs using scripting languages
Java
HADOOP 1.0
MapReduce (cluster resource management and data processing)
HDFS (redundant, reliable storage)
Since 2012
Users could work on multiple processing models in addition to MapReduce
HADOOP
MapReduce (data processing)
Others (data processing)
YARN (cluster resource management)
HDFS (redundant, reliable storage)

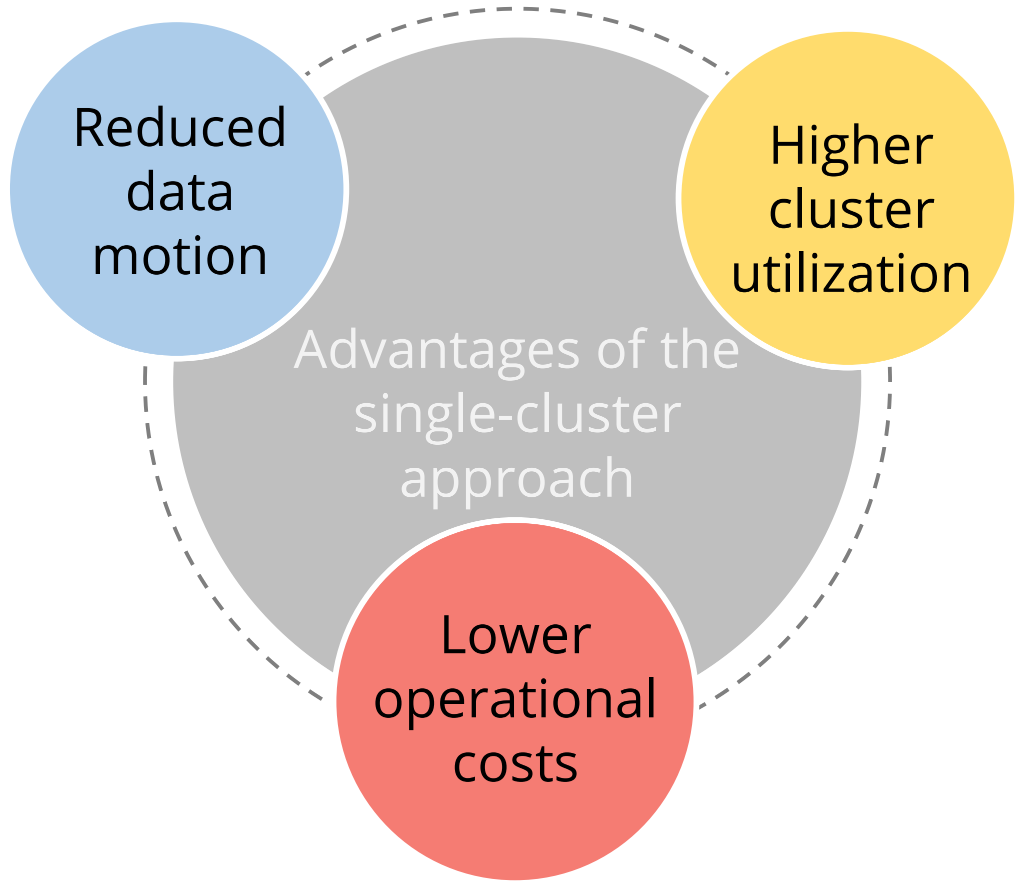
YARN: Use Case

YAHOO!

- Yahoo was the first company to embrace Hadoop, and it is a trendsetter within the Hadoop ecosystem. In late 2012, it struggled to handle iterative and stream processing of data on Hadoop infrastructure due to MapReduce limitations.
- After implementing YARN in the first quarter of 2013, Yahoo has installed more than 30,000 production nodes on
    - Spark for iterative processing
    - Storm for stream processing
    - Hadoop for batch processing
- Such a solution was possible only after YARN was introduced and multiple processing frameworks were implemented.

YARN: Advantages

The single-cluster approach provides a number of advantages including:


Reduced data motion
Higher cluster utilization
Advantages of the single-cluster approach
Lower operational costs

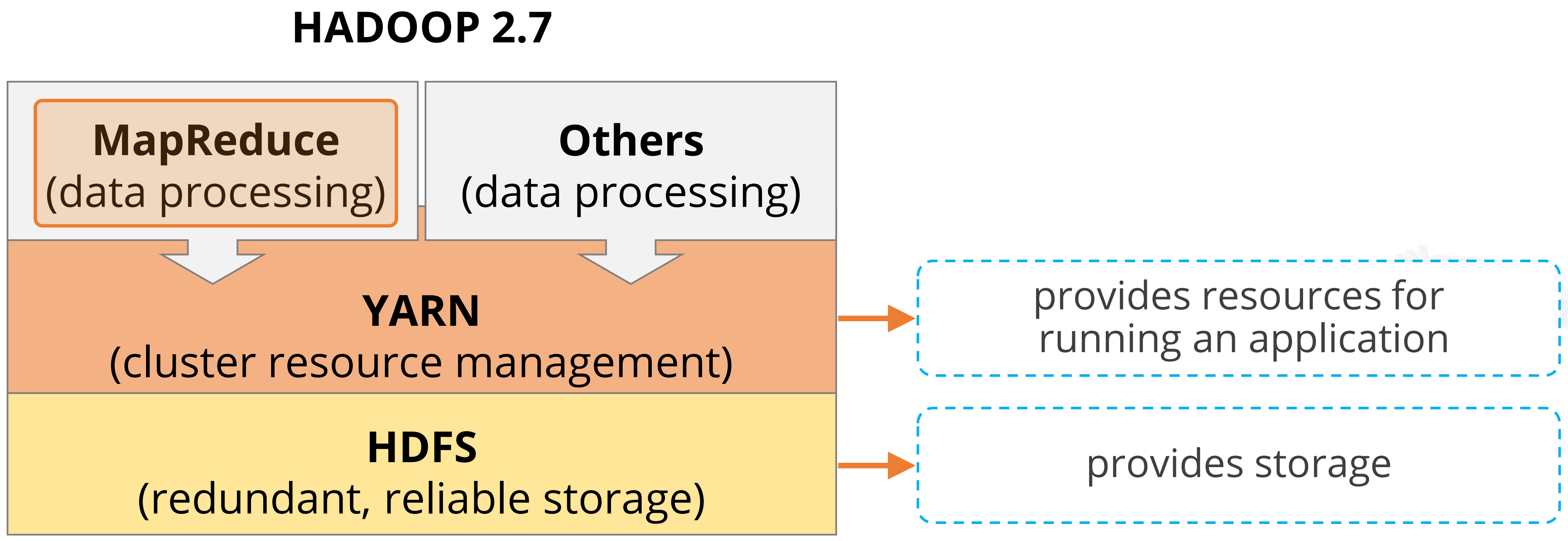
There's no need to move data between Hadoop YARN and systems running on different clusters of computers

Resources unutilized by a framework can be consumed by another

Only one "do-it-all" cluster needs to be managed

YARN Infrastructure

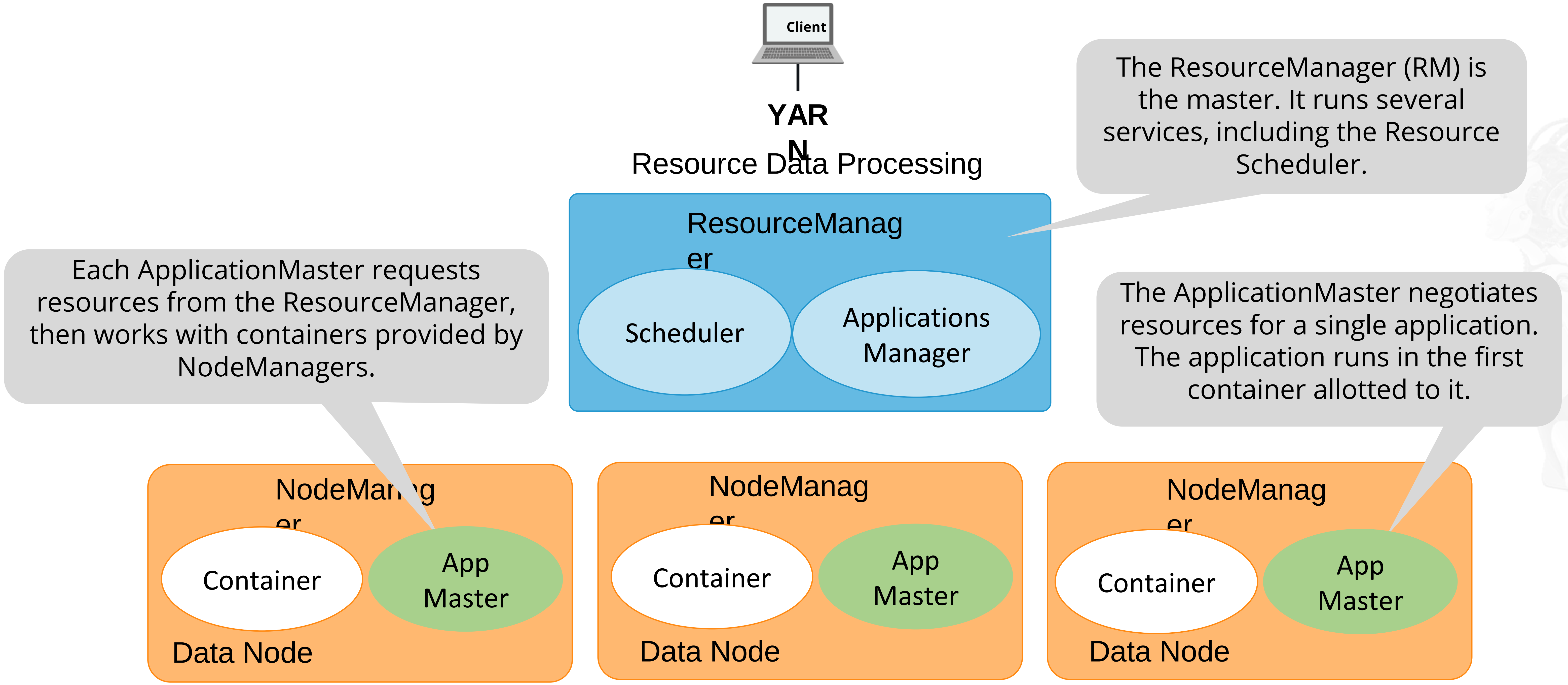
The YARN Infrastructure is responsible for providing computational resources for application executions.


HADOOP 2.7
MapReduce
(data processing)
Others
(data processing)
YARN
(cluster resource management)
HDFS
(redundant, reliable storage)
provides resources for running an application
provides storage

YARN and Its Architecture

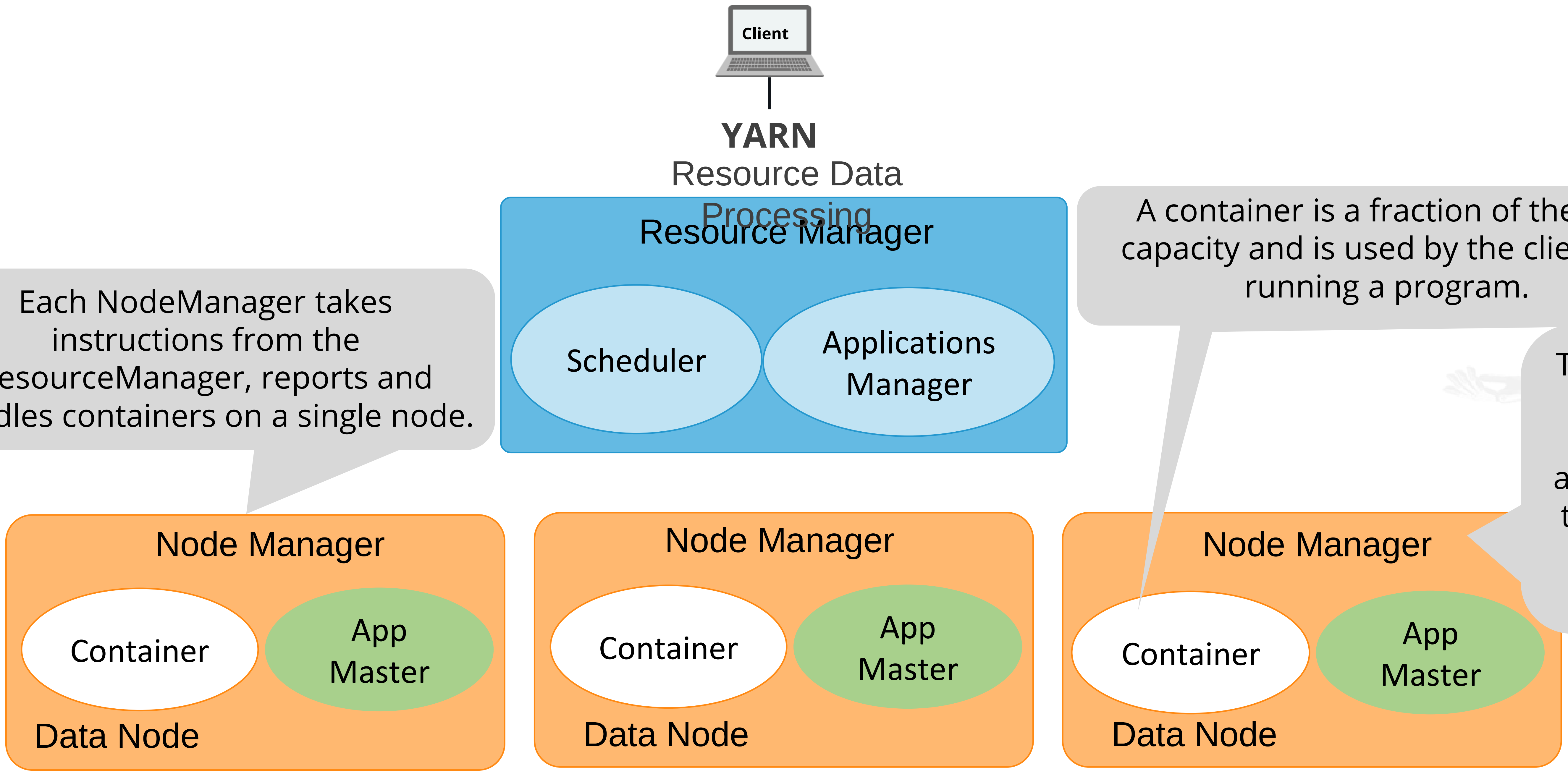
Elements of YARN Architecture

The three important elements of the YARN architecture are the ResourceManager, ApplicationMaster, and NodeManager.



Client
YARN
Resource Data Processing
The ResourceManager (RM) is the master. It runs several services, including the Resource Scheduler.
ResourceManager
Scheduler
Applications Manager
Each ApplicationMaster requests resources from the ResourceManager, then works with containers provided by NodeManagers.
The ApplicationMaster negotiates resources for a single application. The application runs in the first container allotted to it.
NodeManager
Container
App Master
Data Node
NodeManager
Container
App Master
Data Node
NodeManager
Container
App Master
Data Node

Elements of YARN Architecture



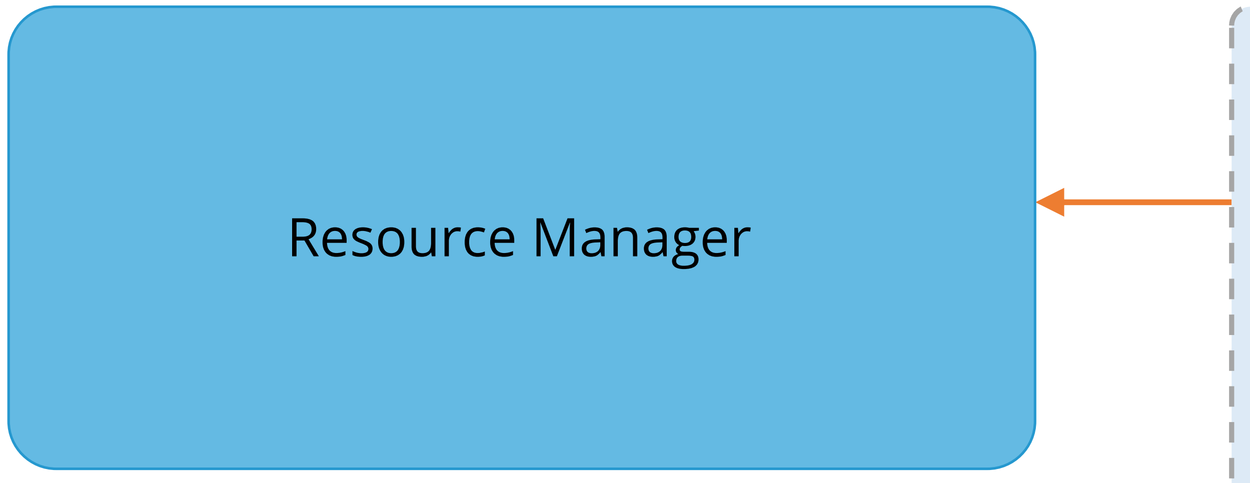
Each NodeManager takes instructions from the ResourceManager, reports and handles containers on a single node.

A container is a fraction of the NM capacity and is used by the client for running a program.

The NodeManager (NM) is the slave. When it starts, it announces itself to the RM and offers resources to the cluster.

YARN Architecture Element: ResourceManager

The RM mediates the available resources in the cluster among competing applications for maximum cluster utilization.



It includes a pluggable scheduler called the YarnScheduler which allows different policies for managing constraints such as capacity, fairness, and Service Level Agreements..

ResourceManager Component: Scheduler

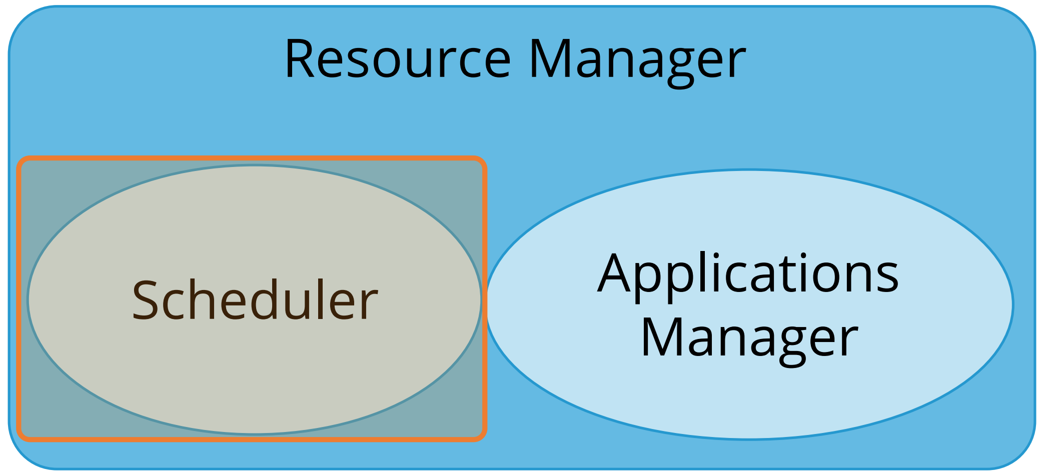
The Scheduler is responsible for allocating resources to various running applications.

The Scheduler does not monitor or track the status of the application.

The Scheduler does not restart failed tasks.

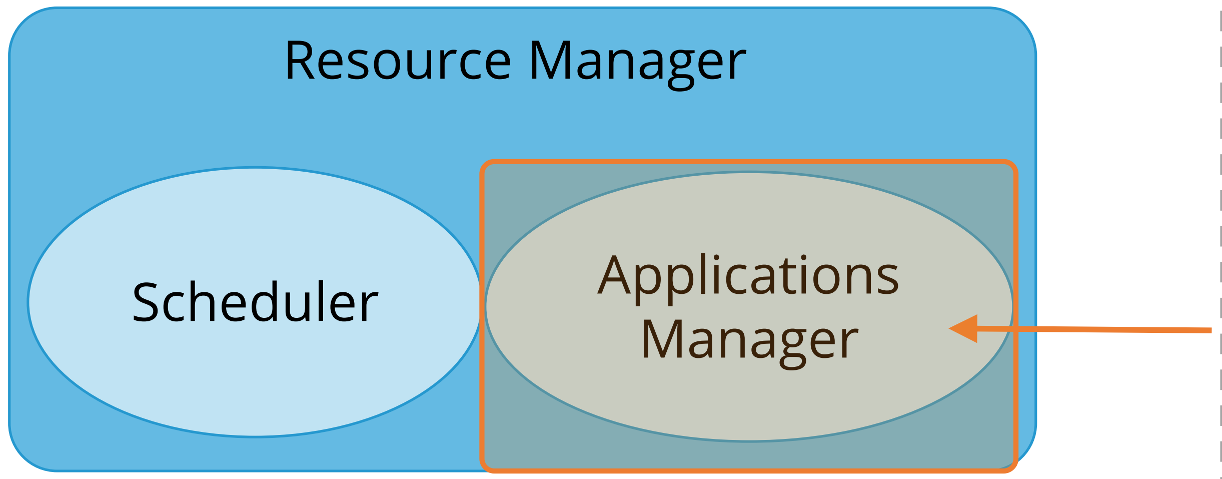
The Scheduler has a policy plug-in to partition cluster resources among various applications.

Most commonly used schedulers are FIFO Scheduler, Capacity Scheduler, and Fair Scheduler.



ResourceManager Component: ApplicationsManager

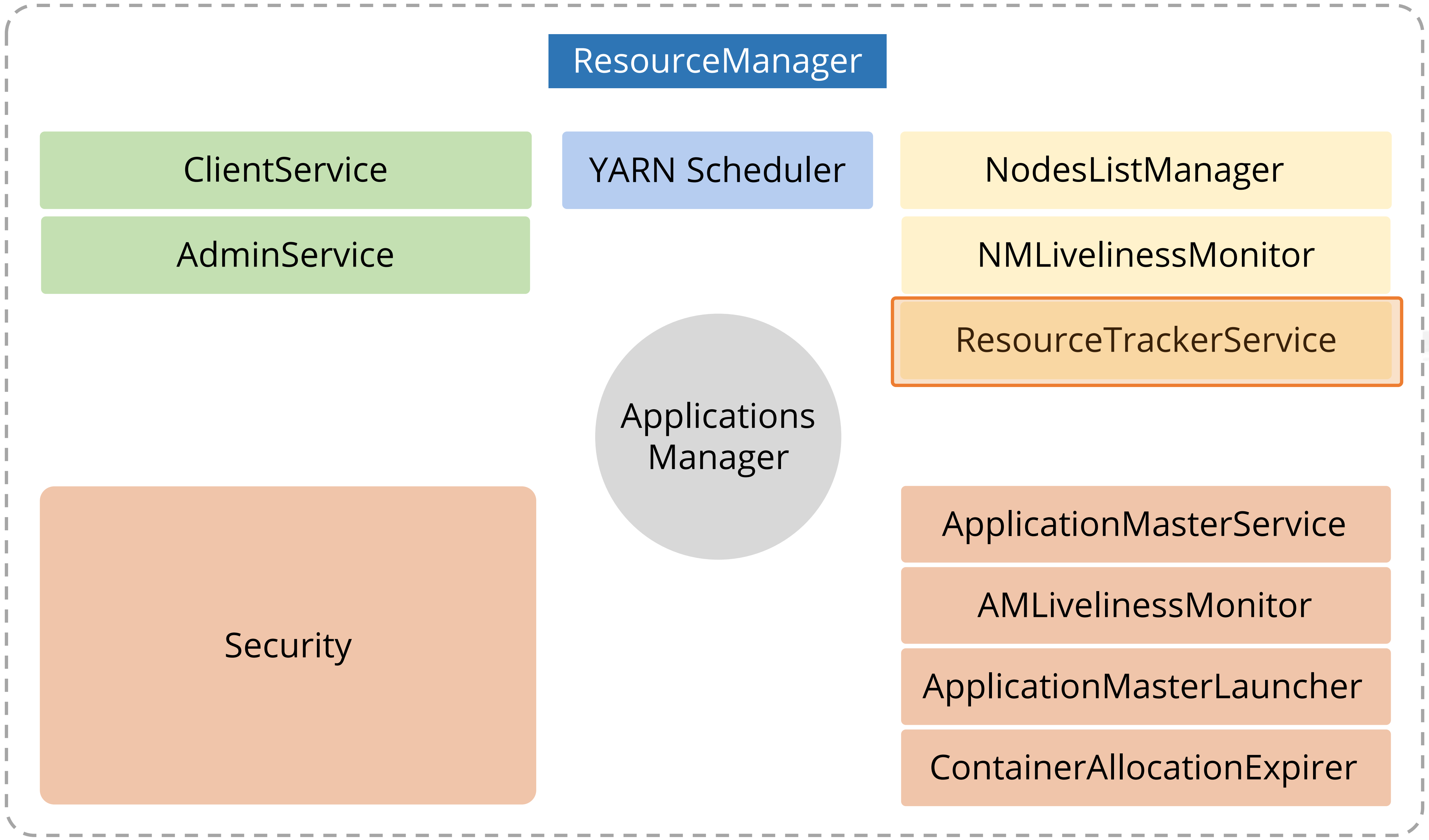
The ApplicationsManager is an interface which maintains a list of applications that have been submitted, currently running, or completed.



The ApplicationsManager accepts job submissions, negotiates the first container for executing the application, and restarts the ApplicationMaster container on failure.

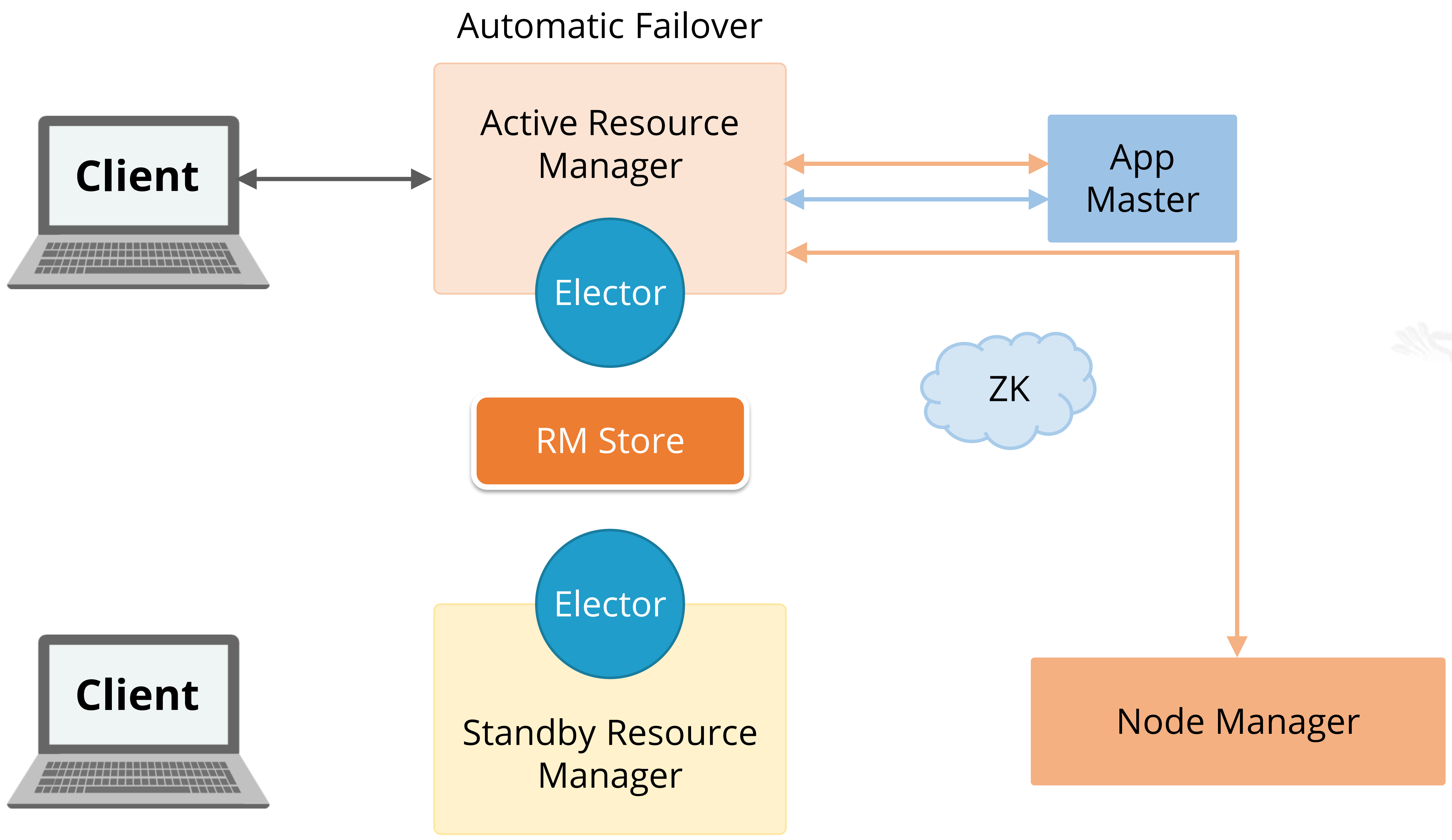
How a ResourceManager Operates

The figure shown here displays all the internal components of the ResourceManager.



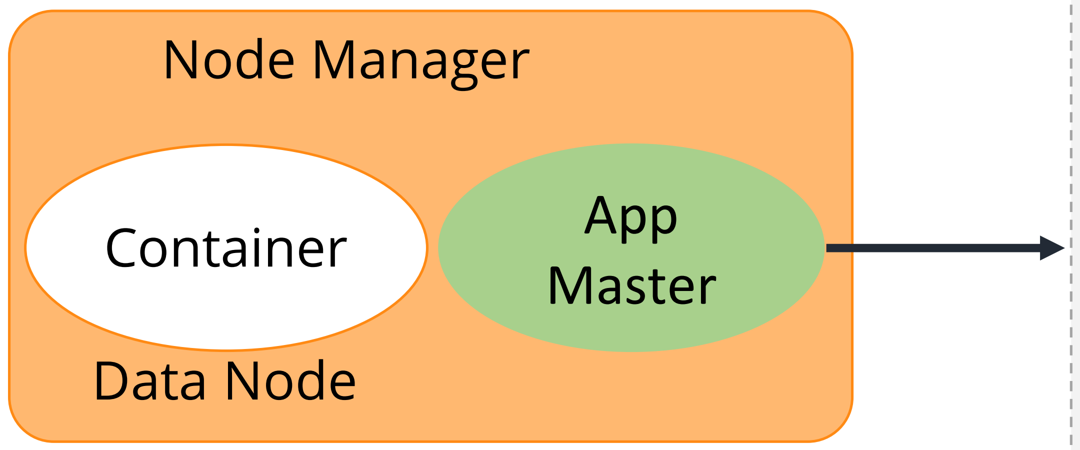
ResourceManager in High Availability Mode

Before Hadoop 2.4, the ResourceManager was the single point of failure in a YARN cluster.
The High Availability, or HA feature is an Active/Standby ResourceManager pair to remove this single point of failure.



YARN Architecture Element: ApplicationMaster

The ApplicationMaster in YARN is a framework-specific library which negotiates resources from the RM and works with the NodeManager or Managers to execute and monitor containers and their resource consumption.



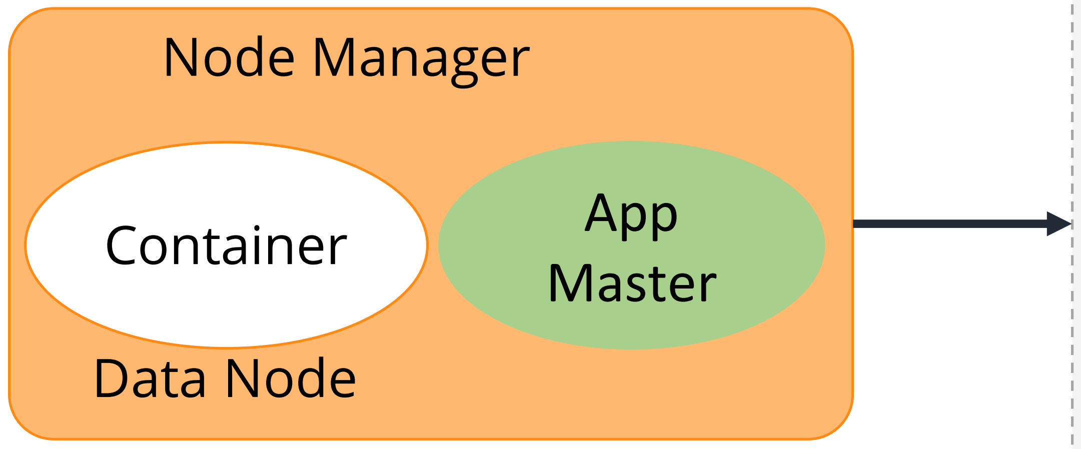
The ApplicationMaster:

- manages the application lifecycle
- makes dynamic adjustments to resource consumption
- manages execution flow
- manages faults
- provides status and metrics to the RM
- interacts with NodeManager and RM using extensible communication protocols
- is not run as a trusted service

While every application has its own instance of an AppMaster, it is possible to implement an AppMaster for a set of applications as well.

YARN Architecture Element: NodeManager

When a container is leased to an application, the NodeManager sets up the container's environment including the resource constraints specified in the lease and any dependencies.

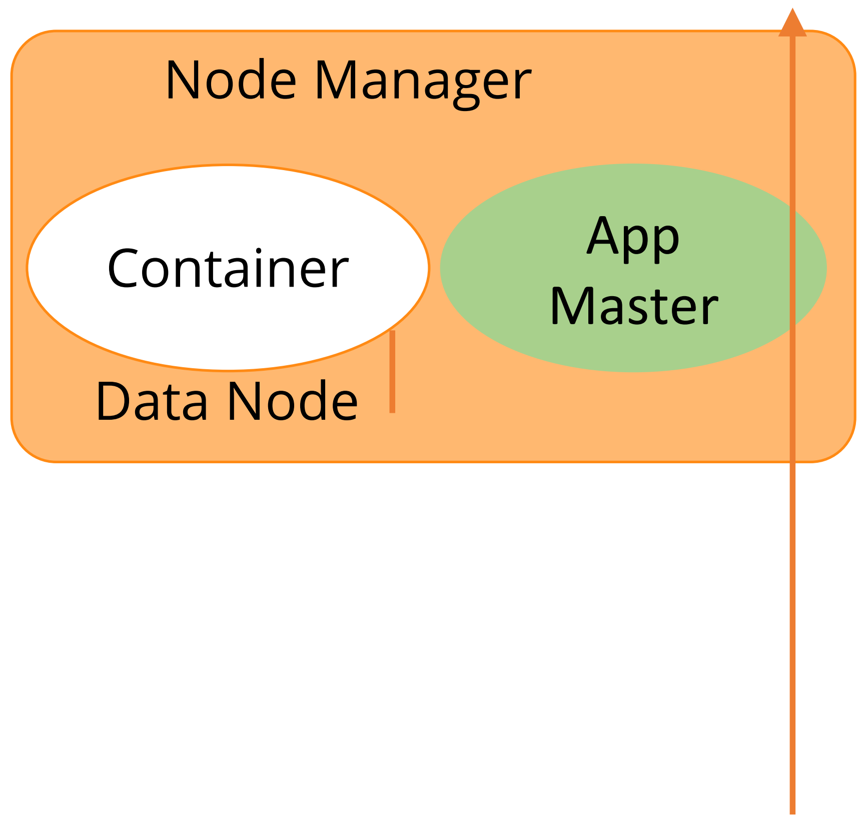


The NodeManager runs on each node and manages the following:

- Container lifecycle management
- Container dependencies
- Container leases
- Node and container resource usage
- Node health
- Log management
- Reporting node and container status to the RM

YARN Container

A YARN container is a collection of a specific set of resources to use in certain numbers on a specific node.

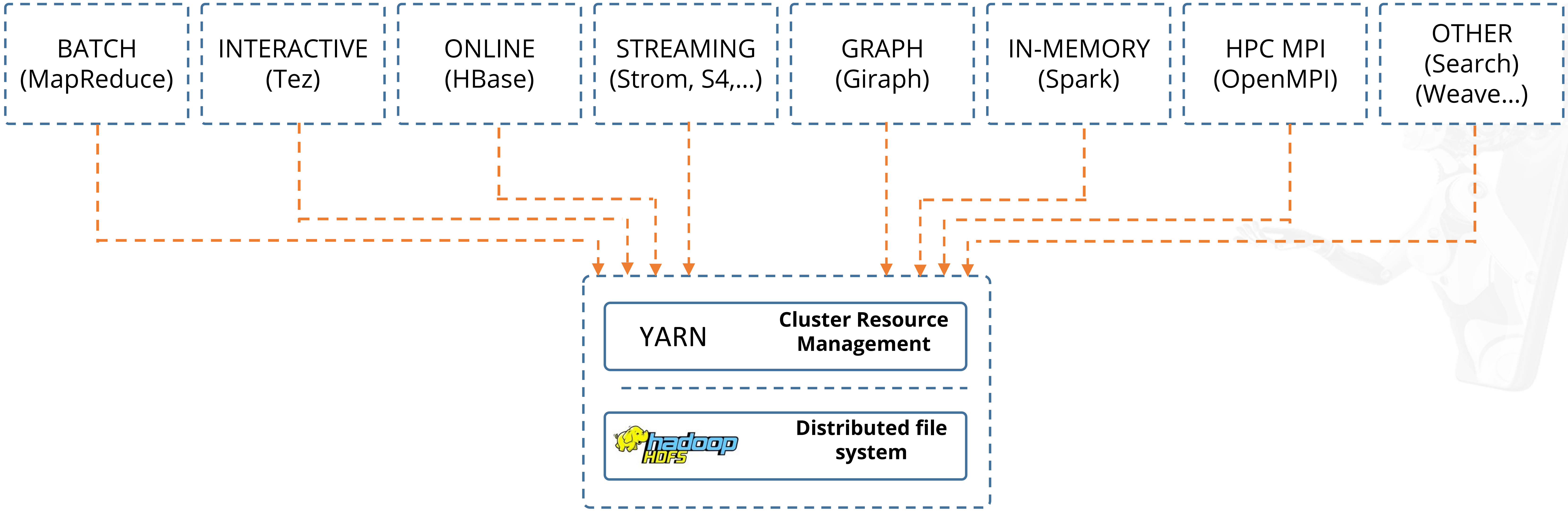


To launch the container, the ApplicationMaster must provide a Container Launch Context (CLC) that includes the following information:

- Environment variables
- Dependencies, that is, local resources such as data files or shared objects needed prior to launch
- Security tokens
- The command necessary to create the process the application wants to launch

Applications on YARN

There can be many different workloads running on a Hadoop YARN cluster.



How YARN Runs an Application

There are five steps involved in running an application by YARN:

01 The client submits an application to the ResourceManager

02 The ResourceManager allocates a container

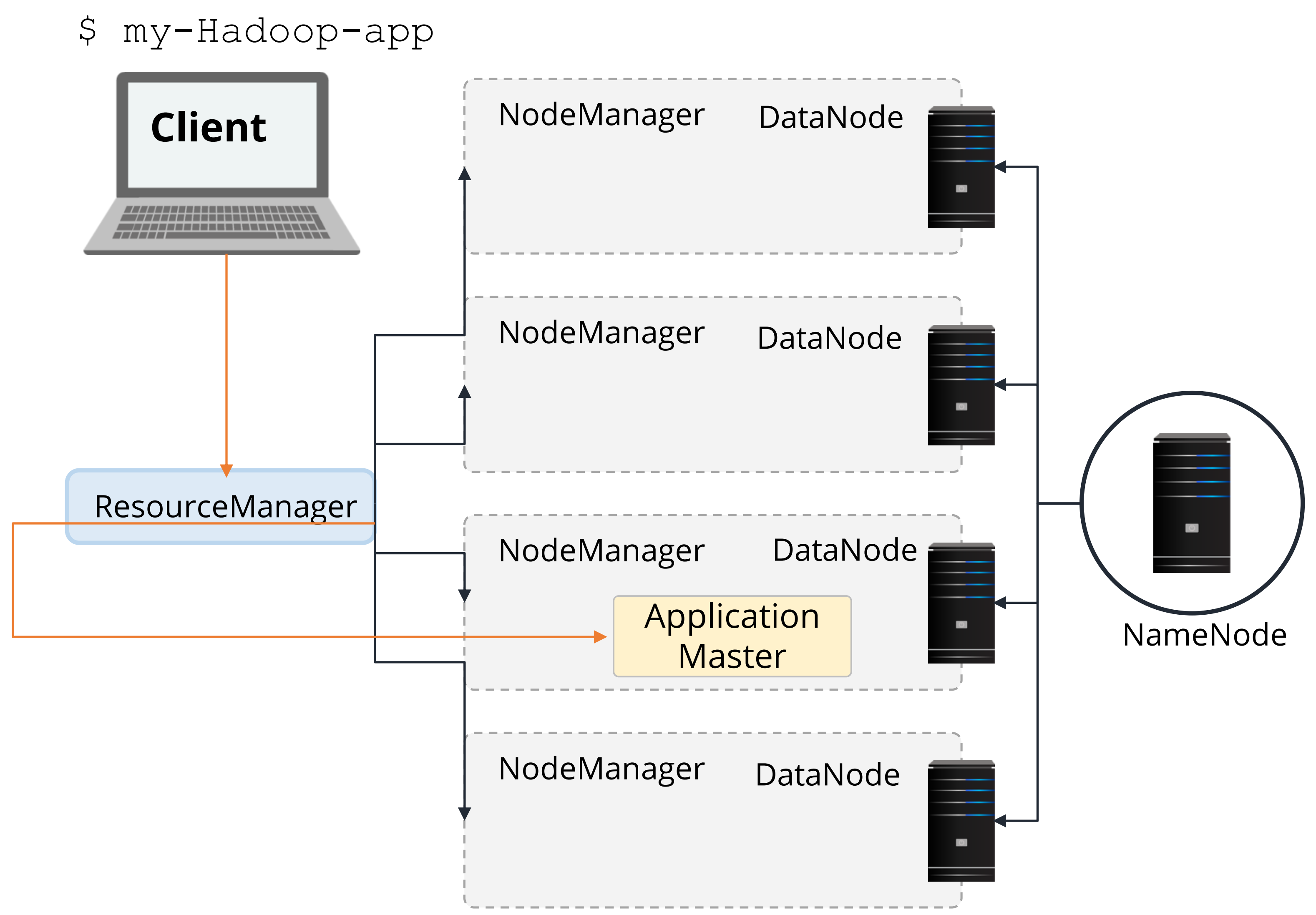
03 The ApplicationMaster contacts the related NodeManager

04 The NodeManager launches the container

05 The container executes the ApplicationMaster

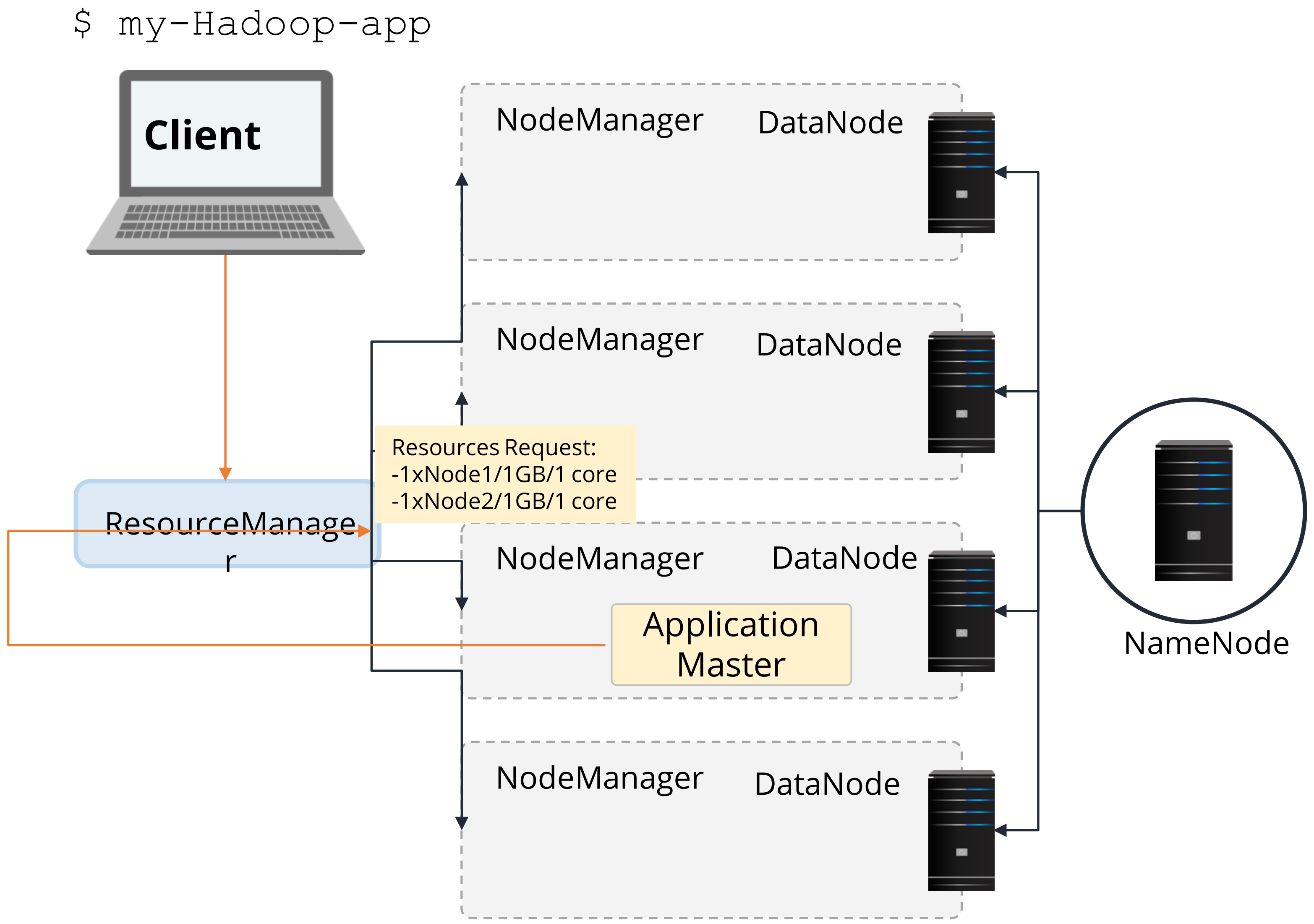
Step1: Application Submitted to ResourceManager

Users submit applications to the ResourceManager by typing the Hadoop jar command.



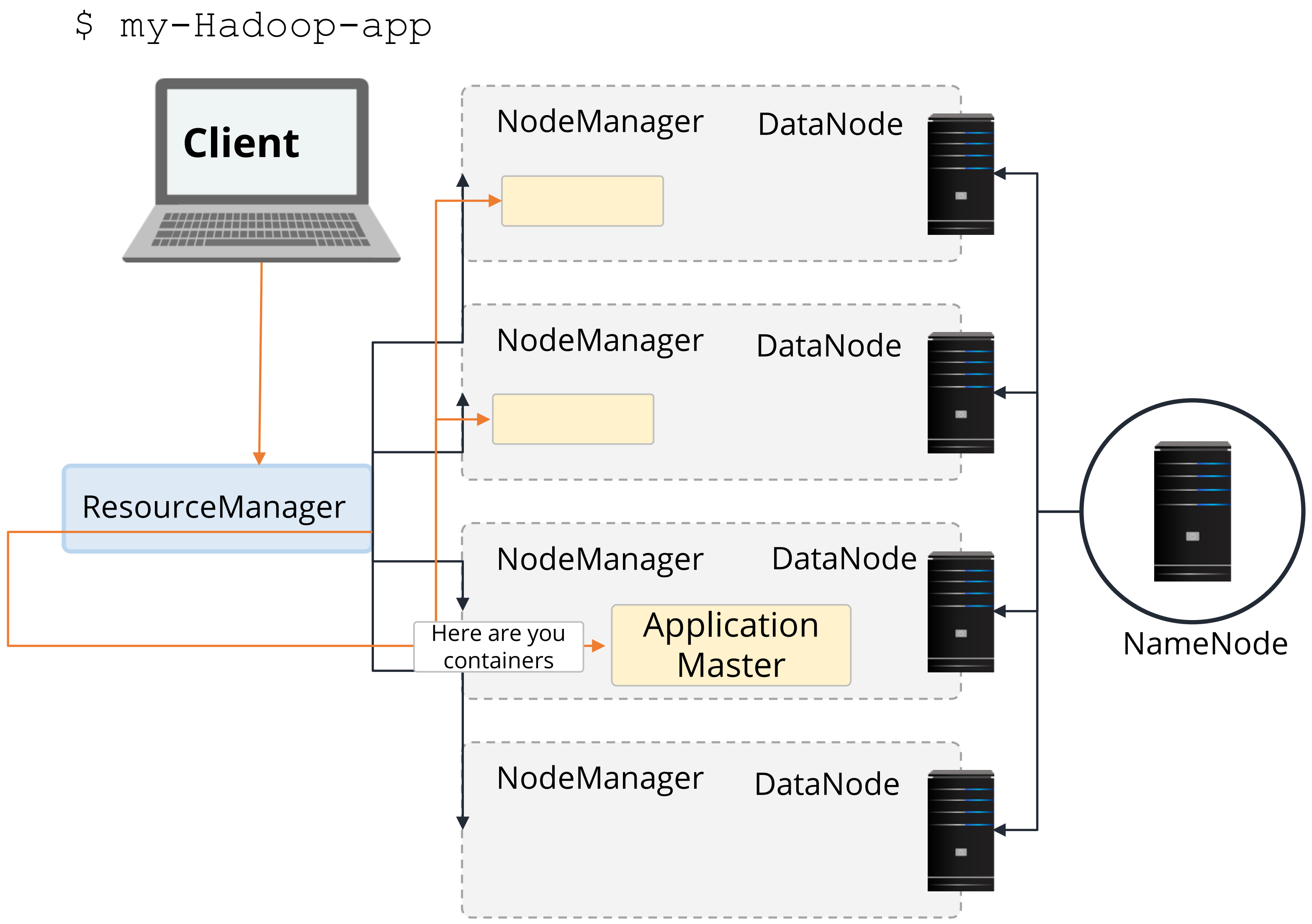
Step2: ResourceManager Allocates a Container

When the ResourceManager accepts a new application submission, one of the first decisions the Scheduler makes is selecting a container.



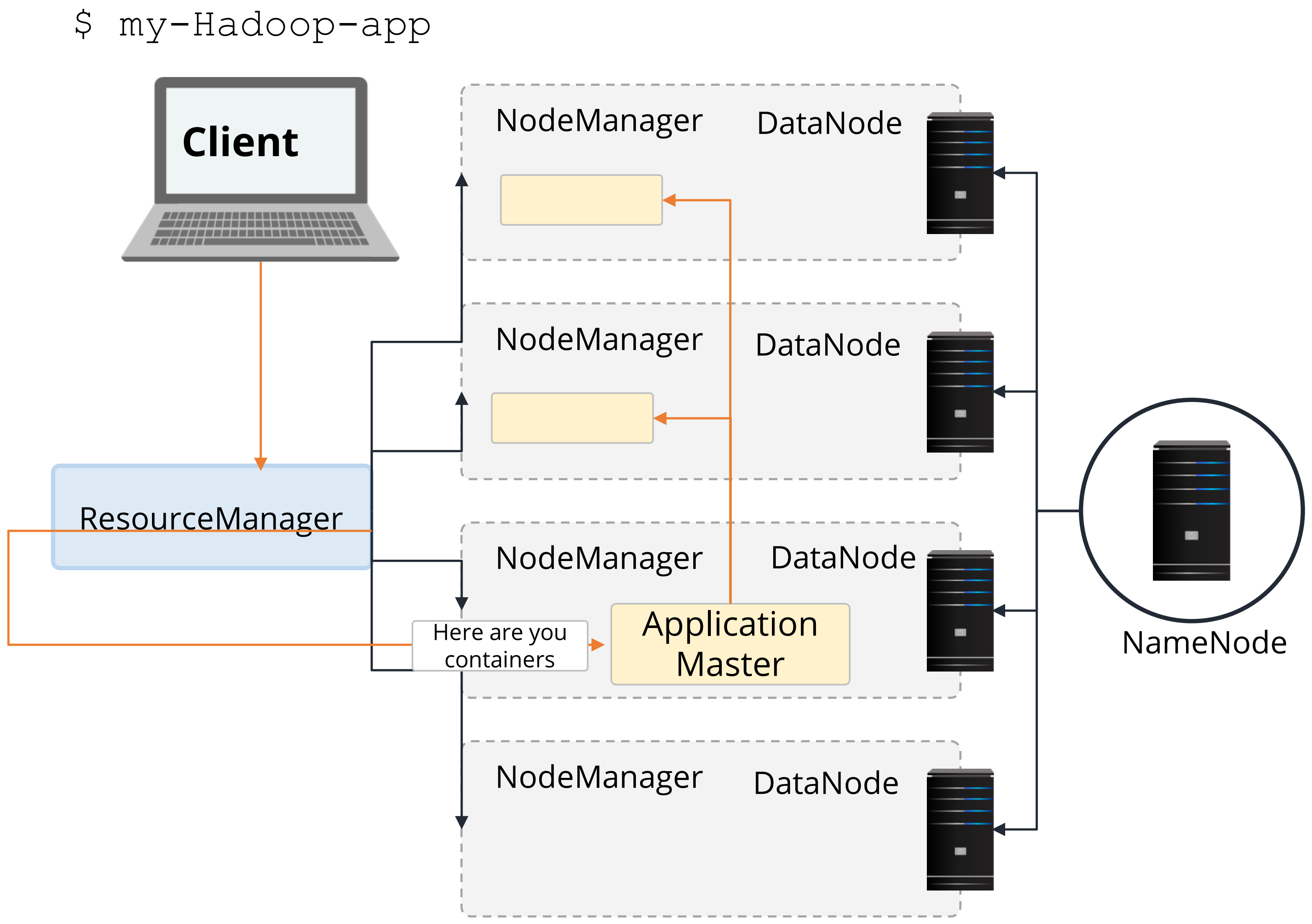
Step3: ApplicationMaster Contacts NodeManager

After a container is allocated, the ApplicationMaster asks the NodeManager managing the host on which the container was allocated to use these resources to launch an application-specific task.



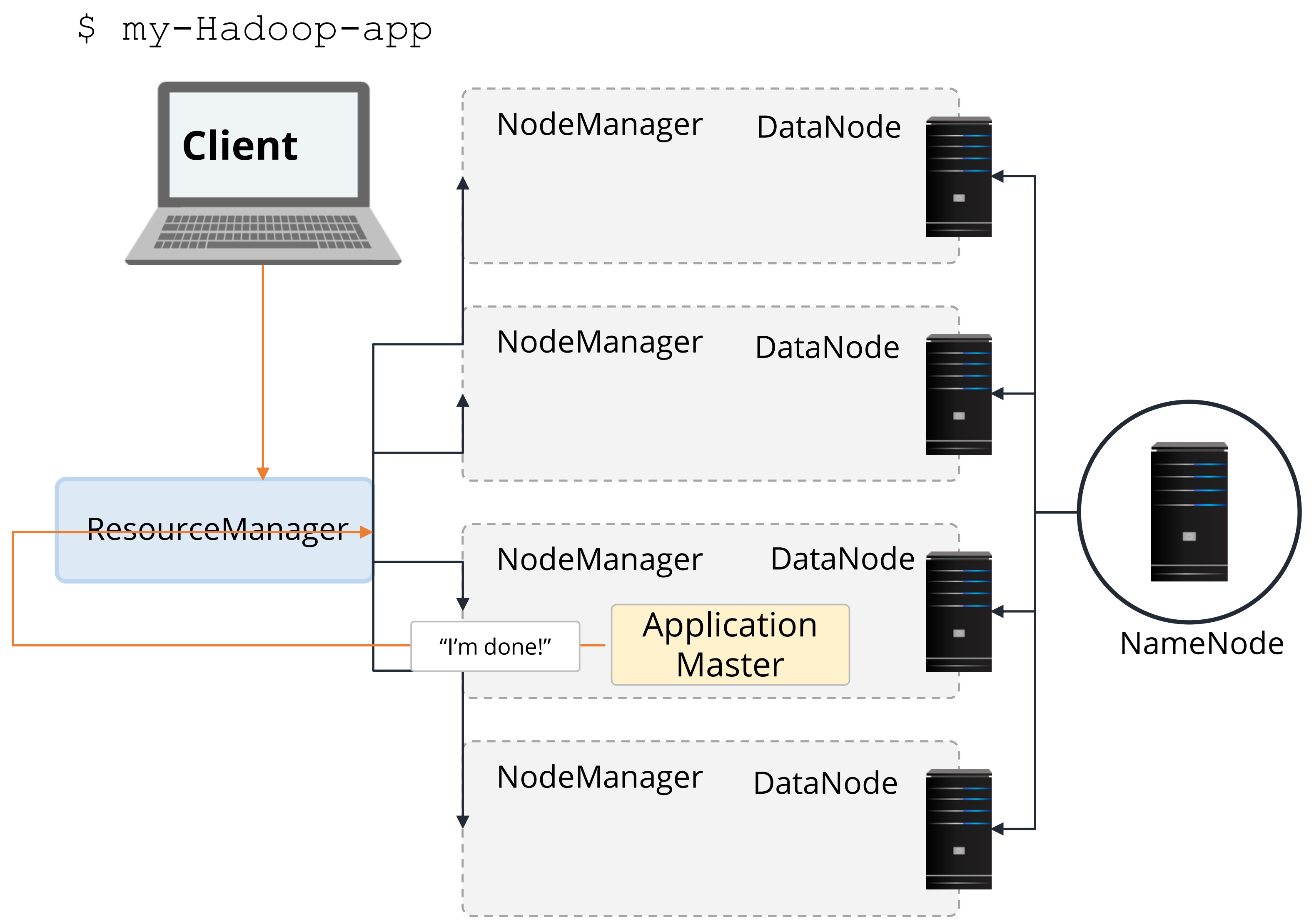
Step4: ResourceManager Launches a Container

The NodeManager does not monitor tasks; it only monitors the resource usage in the containers.



Step5: Container Executes the ApplicationMaster

After the application is complete, the ApplicationMaster shuts itself and releases its own container.



Tools for YARN Developers

Hadoop includes three tools for YARN developers:

- YARN Web UI
- Hue Job Browser
- YARN Command Line

YARN Web UI

YARN
Web UI

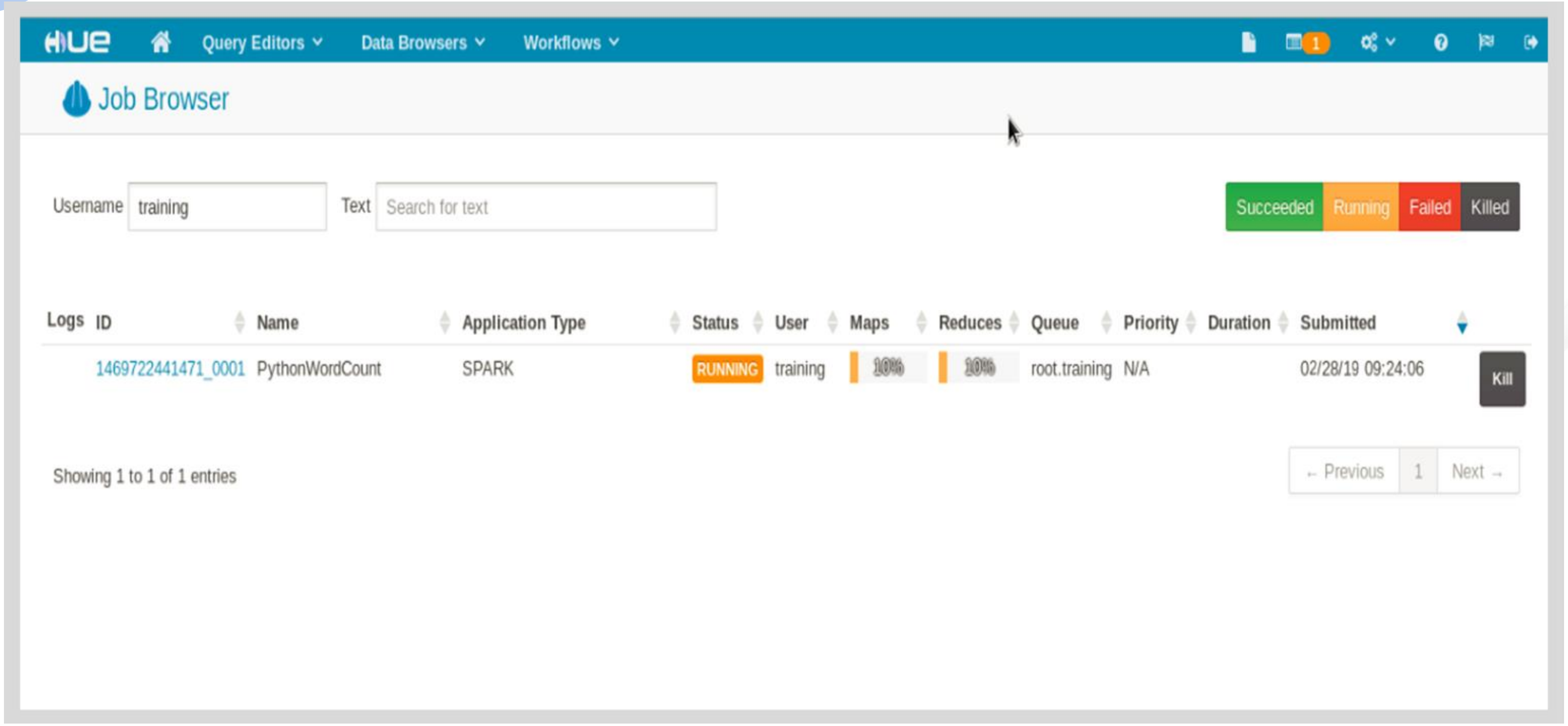
YARN web UI runs on 8088 port, by default.

It also provides a better view than Hue. However, you can’t control or configure YARN from web UI.

Hue Job Browser

Hue Job Browser

The Hue Job Browser allows you to monitor the status of a job, kill a running job, and view logs.



Username: training
Text: Search for text

Logs	ID	Name	Application Type	Status	User	Maps	Reduces	Queue	Priority	Duration	Submitted
	1469722441471_0001	PythonWordCount	SPARK	RUNNING	training	10%	10%	root.training	N/A		02/28/19 09:24:06

Showing 1 to 1 of 1 entries

YARN Command Line

YARN Command Line

Most of the YARN commands are for the administrator rather than the developer.

Few useful commands for developer:

- yarn –help
  list all command of yarn

- yarn –version
  print the version

- yarn logs -applicationId <app-id>
  views logs of specified application ID

Assisted Practice

YARN

Duration: 15 mins

Problem Statement: In this demonstration, you will learn how to use YARN.

Access: Click on the **Practice Labs** tab on the left side panel of the LMS. Copy or note the username and password that is generated. Click on the **Launch Lab** button. On the page that appears, enter the username and password in the respective fields, and click **Login**.

Key Takeaways

You are now able to:

- Understand how Hadoop Distributed File System (HDFS) stores data across a cluster
- Illustrate the HDFS architecture and its components
- Demonstrate the use of HDFS Command Line Interface (CLI)
- Illustrate the YARN architecture and its components
- Demonstrate how to use Hue, YARN Web UI, and the YARN command to monitor the cluster

Knowledge Check

Knowledge Check

1

Which of the following statements best describes how a large (100 GB) file is stored in HDFS?

a. The file is replicated three times by default. Each copy of the file is stored on a separate DataNode.

b. The master copy of the file is stored on a single DataNode. The replica copies are divided into fixed-size blocks which are stored on multiple DataNodes.

c. The file is divided into fixed-size blocks which are stored on multiple DataNodes. Each block is replicated three times by default. Multiple blocks from the same file might reside on the same DataNode.

d. The file is divided into fixed-size blocks which are stored on multiple DataNodes. Each block is replicated three times by default. HDFS guarantees that different blocks from the same file are never on the same DataNode.

Knowledge Check 1

Which of the following statements best describes how a large (100 GB) file is stored in HDFS?

a. The file is replicated three times by default. Each copy of the file is stored on a separate DataNode.

b. The master copy of the file is stored on a single DataNode. The replica copies are divided into fixed-size blocks which are stored on multiple DataNodes.

c. The file is divided into fixed-size blocks which are stored on multiple DataNodes. Each block is replicated three times by default. Multiple blocks from the same file might reside on the same DataNode.

d. The file is divided into fixed-size blocks which are stored on multiple DataNodes. Each block is replicated three times by default. HDFS guarantees that different blocks from the same file are never on the same DataNode.

The correct answer is **d.**

The file is divided into fixed-size blocks which are stored on multiple DataNodes. Each block is replicated three times by default. HDFS guarantees that different blocks from the same file are never on the same DataNode.

Knowledge Check

2

How many blocks are required to store a file of size 514 MB in HDFS using default block size configuration?

a. 4

b. 5

c. 6

d. 7

Knowledge Check

2

How many blocks are required to store a file of size 514 MB in HDFS using default block size configuration?

a. 4

b. 5

c. 6

d. 7

The correct answer is **b.**

The default block size in Hadoop 3.x is 128 MB. So, a file of size 514 MB will be divided into 5 blocks where the first four blocks will be of 128 MB and the last block will be of 2 MB.

Knowledge Check

3

Which of the following statements is not true about HDFS?

a. We cannot modify files already present in HDFS

b. Multiple clients can write into an HDFS file concurrently

c. Both A and B

d. None of the above

Knowledge Check
3

Which of the following statements is not true about HDFS?

a. We cannot modify files already present in HDFS

b. Multiple clients can write into an HDFS file concurrently

c. Both A and B

d. None of the above

The correct answer is **b.**

HDFS follows Write Once Read Many model.
So, multiple clients can't write into an HDFS file concurrently.

Knowledge Check

4

What is the model of a ZooKeeper cluster?

a. Master/Slave

b. Leader and Follower

c. Peer to Peer

d. Primary and Secondary

Knowledge Check

4

What is the model of a ZooKeeper cluster?

a. Master/Slave

b. Leader and Follower

c. Peer to Peer

d. Primary and Secondary

The correct answer is **b.**

Apache ZooKeeper deploys a Leader and Follower model to provide coordination service for distributed applications.

Knowledge Check

5

Which of the following scheduling policies can be implemented in Yarn?

a. FIFO scheduler

b. Capacity scheduler

c. Fair scheduler

d. All of the above

Knowledge Check

5

Which of the following scheduling policies can be implemented in Yarn?

a. FIFO scheduler

b. Capacity scheduler

c. Fair scheduler

d. All of the above

The correct answer is **d.**

YARN provides three scheduling options - FIFO scheduler, Capacity scheduler, and Fair scheduler for scheduling resources to user applications.

Knowledge Check

6

______ negotiates resources from the Resource Manager.

a. Node Manager

b. Resource Manager

c. Application Master

d. None of the above

Knowledge Check

6

______ negotiates resources from the Resource Manager.

a. Node Manager

b. Resource Manager

c. Application Master

d. None of the above

The correct answer is **c.**

Each Application Master is responsible for negotiating resources from the Resource Manager.

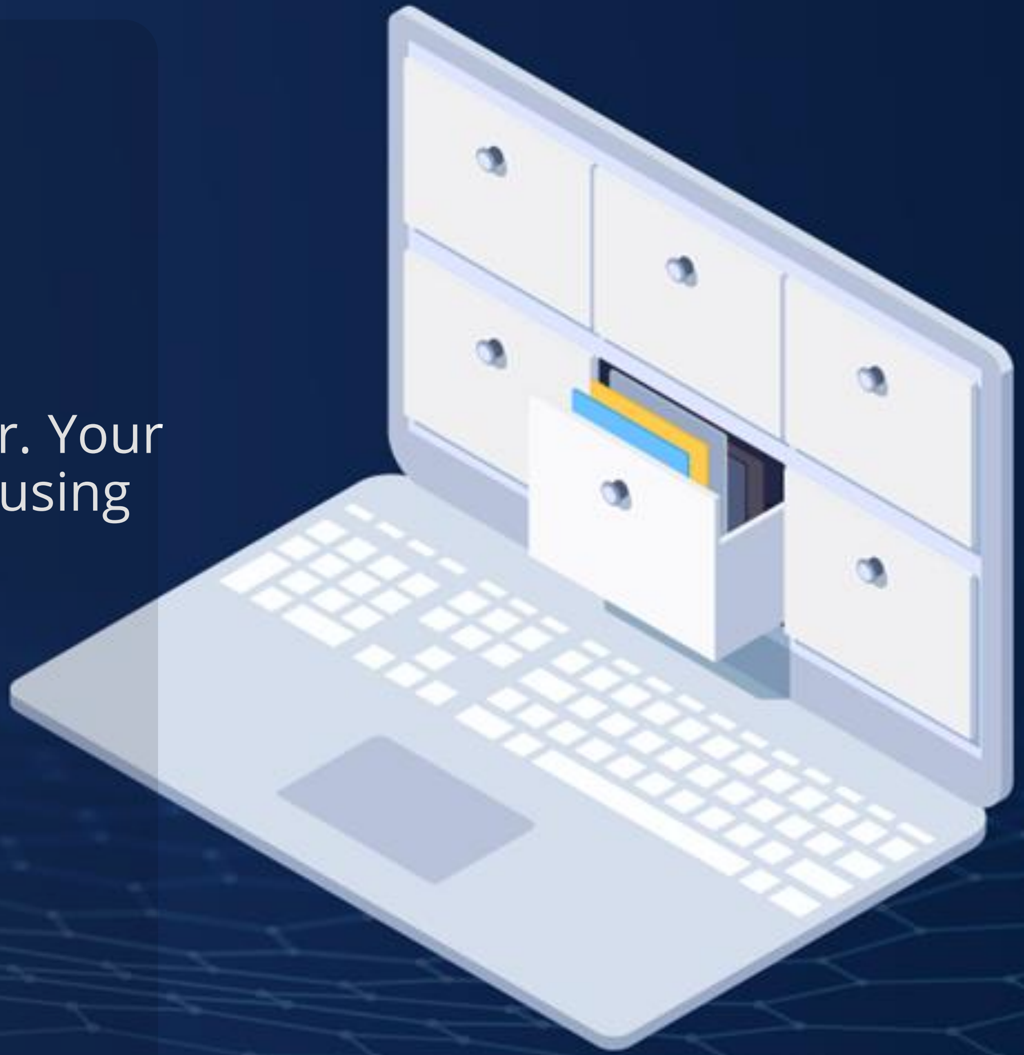
Lesson-End Project

Problem Statement:

PV Consulting is one of the top consulting firms for big data projects.
They mostly help big and small companies to analyze their data.

For Spark & Hadoop MR application they started using YARN as a resource manager. Your task is to provide the following information for any job which is submitted to YARN using YARN console and YARN Cluster UI:

1. Who has submitted the job?
2. To which YARN queue is a job submitted?
3. How much time did it take to finish the job?
4. List of all running jobs on YARN
5. How to kill a job using YARN?
6. Check logs using YARN



DATA AND
ARTIFICIAL INTELLIGENCE

Thank You

simplilearn