upx
Move up in life

Introduction to

YARN

Module 3 : Hadoop, HDFS, & YARN

WHERE ARE WE
in this journey?

upx
Move up in life

INTRODUCTION TO BIG DATA

SCALING, STORAGE & COMPUTATION WITH APACHE HADOOP

PYTHON

HDFS

INTRODUCTION TO YARN

MAPREDUCE

PIG

HIVE

LEARNT SO FAR

HOW HADOOP OVERCOMES CHALLENGES FACED IN TRADITIONAL COMPUTING?

GETTING STARTED WITH HADOOP

INTRODUCTION TO HADOOP & HDFS

COMMONLY USED LINUX & HDFS COMMANDS

TAKEAWAYS
from today's session

- DRAWBACKS OF EARLIER HADOOP VERSIONS
- HOW YARN SOLVES DRAWBACKS OF EARLIER HADOOP VERSIONS
- WHERE DOES YARN FIT IN HADOOP ECOSYSTEM?
- ARCHITECTURE OF YARN

AGENDA

Introduction to YARN

- CHALLENGES OF EARLIER HADOOP VERSIONS
- WHAT IS APACHE YARN?
- FEATURES OF YARN
- ARCHITECTURE OF YARN
- COMPONENTS OF YARN

CHALLENGES
of earlier versions of Hadoop

DRAWBACKS IN EARLIER HADOOP VERSIONS

Underutilization of resources

Lack of recovery mechanism

Limited scalability

Processing using only MapReduce

Let us expand on these topics..

SOME DEFINITIONS
that are useful

- A process needs an **area** of computation where actual processing happens
- In Hadoop, this **computation area** is called **SLOT**
- **Types of slots in Hadoop** framework:
  - MAP SLOT
  - REDUCE SLOT

CHALLENGE 1

Under utilization of resources in earlier Hadoop Versions

Earlier Hadoop had ***predefined number of map & reduce slots*** which ***lead to inflexible slots*** configured on DataNodes.

Under utilization occurs because ***map slots*** might be ***'full'*** while ***reduce slots*** are ***empty*** (and vice-versa)

CHALLENGE 2:

Scalability => Limitations of adding resources in earlier Hadoop versions

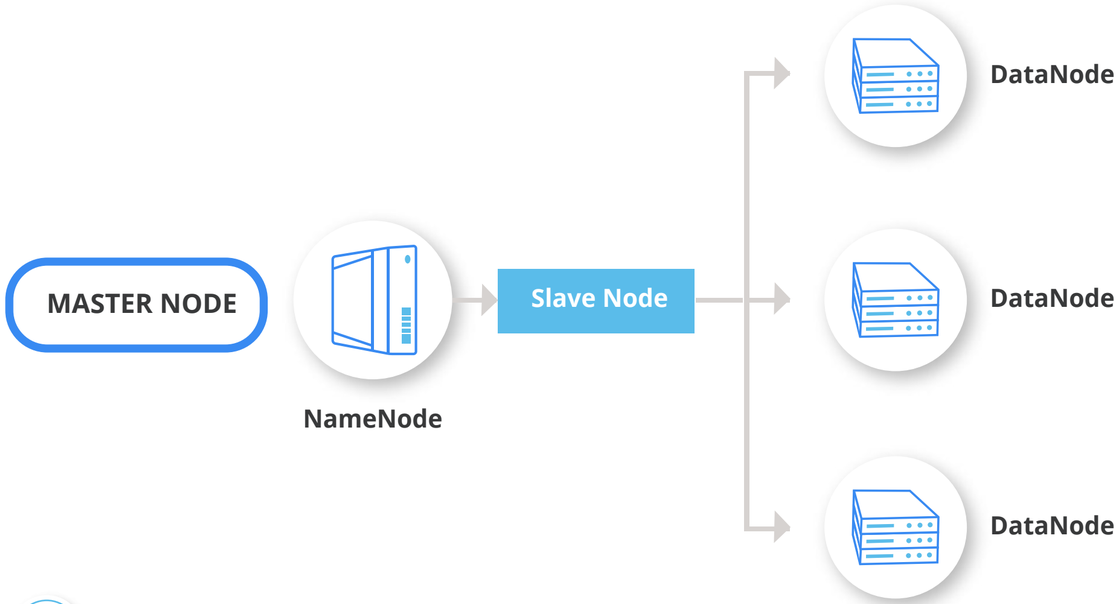
In earlier Hadoop versions, we had

- Maximum cluster size of ***4,000*** nodes
- At most ***40,000 concurrent tasks*** could be executed

CHALLENGE 3:

Lack of recovery mechanism: Single point of failure

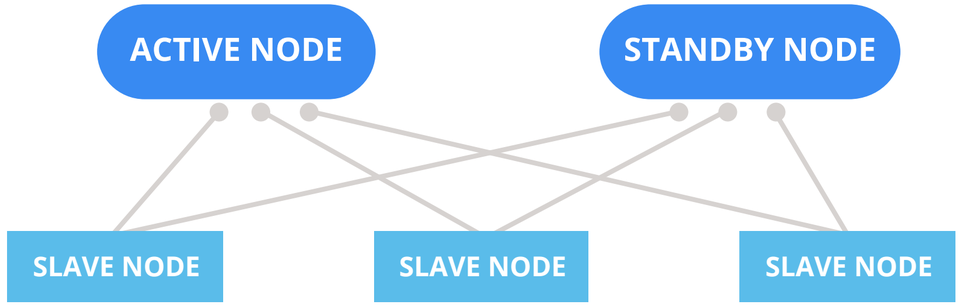
EARLIER HADOOP VERSIONS



- One NameNode
- In case of failure, entire data is lost

This is called ***SINGLE POINT OF FAILURE***

CURRENT HADOOP VERSIONS



- More than one NameNode
- No loss of data

CHALLENGE 4:

Processing using only MapReduce

EARLIER HADOOP VERSIONS

Supported (*MR-based*)

Not supported (*Non-MR based*)

YARN: SOLUTION TO PROBLEMS

in earlier Hadoop versions

upx
Move up in life

PROBLEMS

- Underutilization of resources
- Limited scalability
- Lack of recovery mechanism
- Processing using only MapReduce

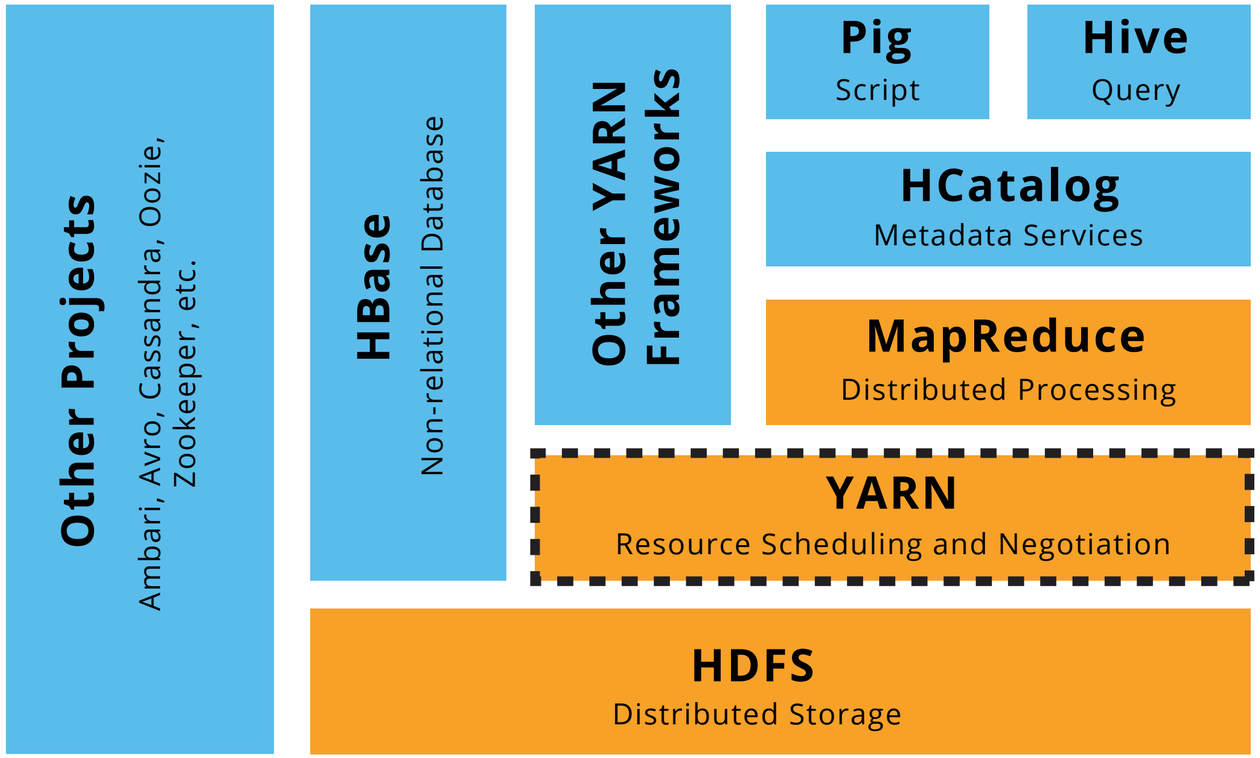
SOLUTION

AGENDA

Introduction to YARN

- CHALLENGES OF EARLIER HADOOP VERSIONS
- WHAT IS APACHE YARN?
- FEATURES OF YARN
- ARCHITECTURE OF YARN
- COMPONENTS OF YARN

WHAT IS YARN?



Apache YARN is a ***Resource Management Layer*** of Hadoop

WHAT IS YARN?

Apache ***YARN*** stands for **Y**et **A**nother **R**esource **N**egotiator

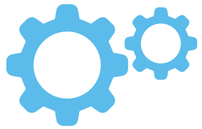
Separates ***resource management*** & ***processing components***

Is considered as a ***data operating system*** of Hadoop

WHAT IS YARN?

Continued

It is a ***central platform for***



CONSISTENT OPERATIONS

Multiple look ups onto the same non actively manipulated file should return the same content irrespective of its location.

DATA GOVERNANCE

YARN manages the lifecycle of data and its accessibility to users.

SECURITY

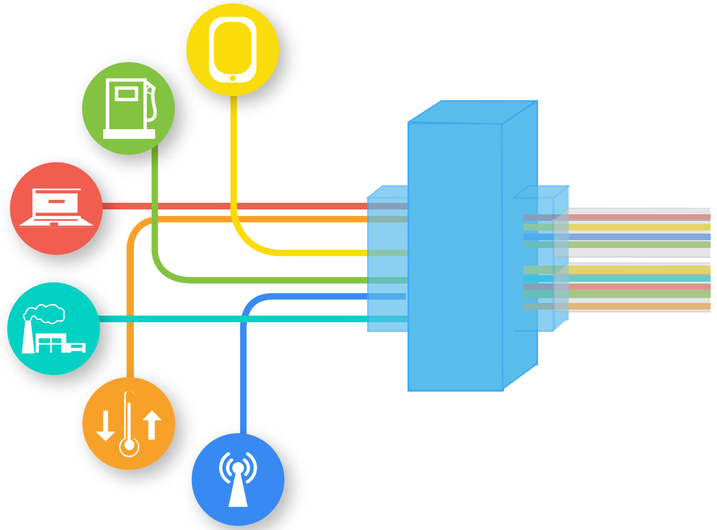
Hadoop is a distributed system and requires authentication between the different components while they communicate with each other.

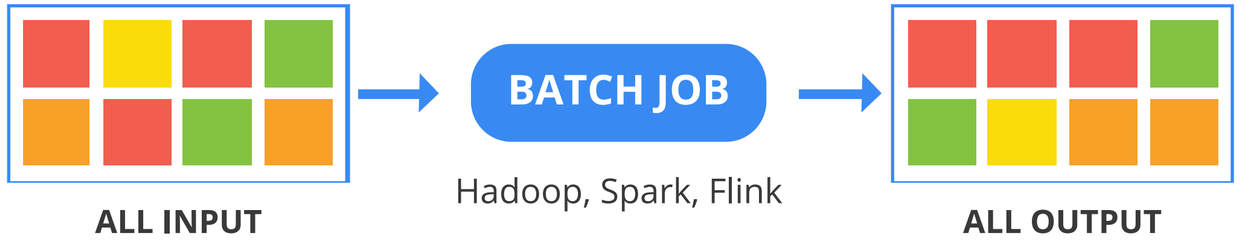
AGENDA

Introduction to YARN

- CHALLENGES OF EARLIER HADOOP VERSIONS
- WHAT IS APACHE YARN?
- FEATURES OF YARN
- ARCHITECTURE OF YARN
- COMPONENTS OF YARN

FEATURES OF YARN





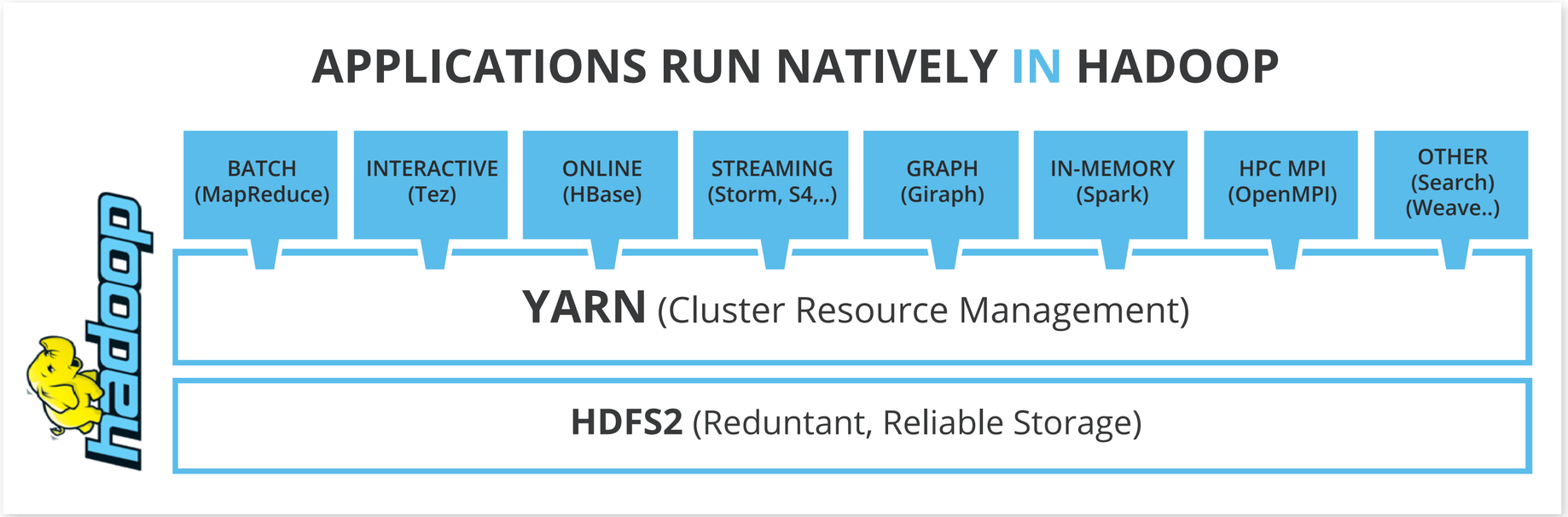
Allows multiple data processing engines such as

- Interactive SQL
- Real-time streaming
- Batch processing

to handle data stored in a single platform

FEATURES OF YARN

Takes ***Hadoop beyond Batch***

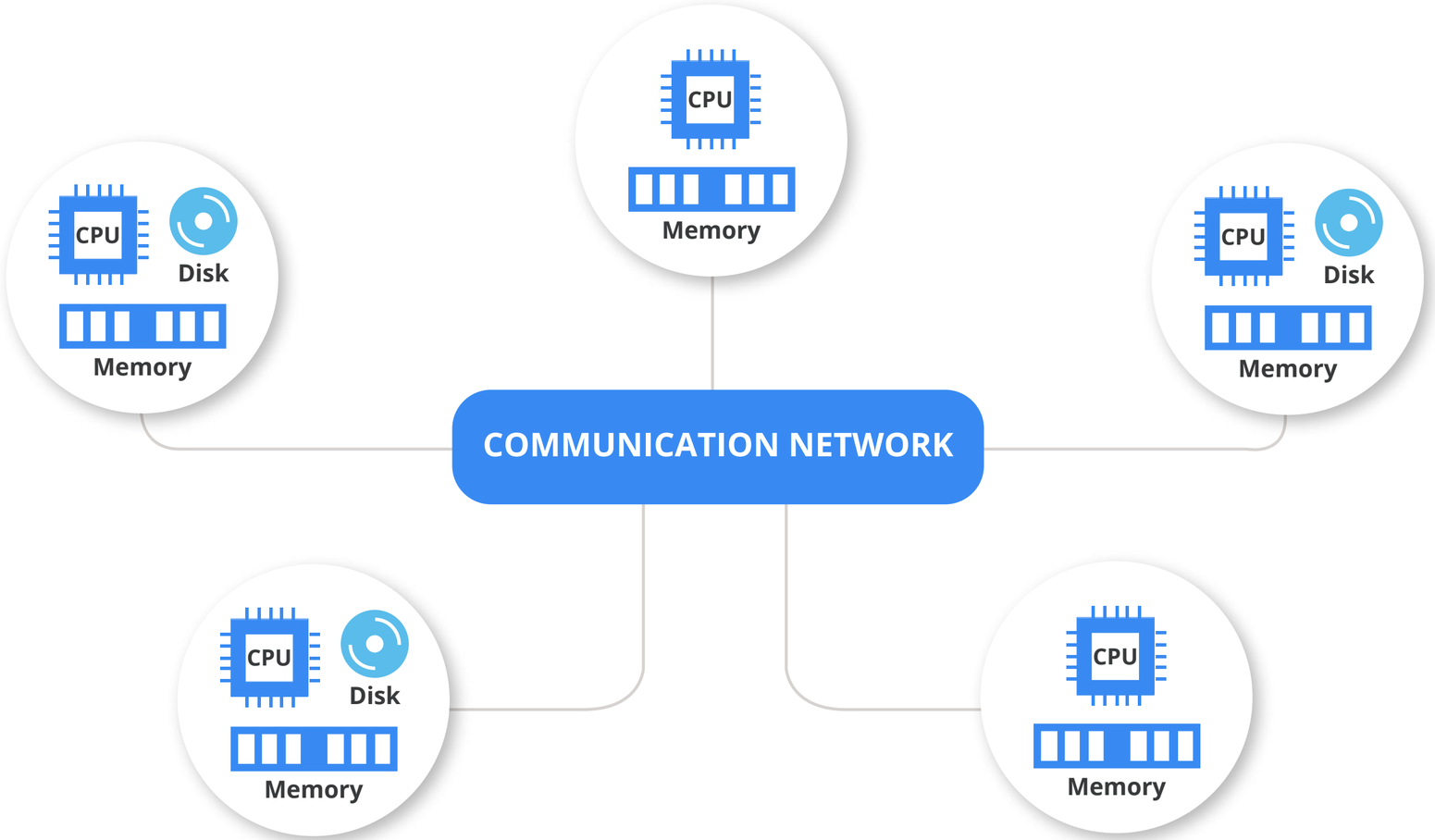


More applications than MapReduce

YARN for better resource utilization

FEATURES OF YARN

YARN is ***the distributed OS of Hadoop***



AGENDA

Introduction to YARN

- CHALLENGES OF EARLIER HADOOP VERSIONS
- WHAT IS APACHE YARN?
- FEATURES OF YARN
- ARCHITECTURE OF YARN
- COMPONENTS OF YARN

Diving deep into YARN!

OVERVIEW
of YARN architecture

YARN architecture has ***four components*** namely:

RESOURCE MANAGER (RM)

- Schedulers

CONTAINERS

NODE MANAGER (NM)

APPLICATION MASTER (AM)

OVERVIEW
of YARN architecture

upx
Move up in life

Continued

RESOURCE MANAGER (RM)

- Keep track of available resources (primarily CPU, memory and data location) on each node of the cluster and schedules jobs
- Arbitrates system resources between competing applications
- Runs on master node
- Has a core component named ***scheduler***

NODE MANAGER (NM)

- Communicate with RM
- Runs on slave nodes

OVERVIEW OF YARN ARCHITECTURE

upx Move up in life

Continued

CONTAINERS

- Allocate a certain amount of resources (memory, CPU) on a slave node
- Created by the RM upon request
- Applications run in one or more containers

APPLICATION MASTER (AM)

- Requests resources (containers) from RM, tracks their status and monitors progress
- One per application
- Requests more containers from RM when required

What are schedulers in YARN?

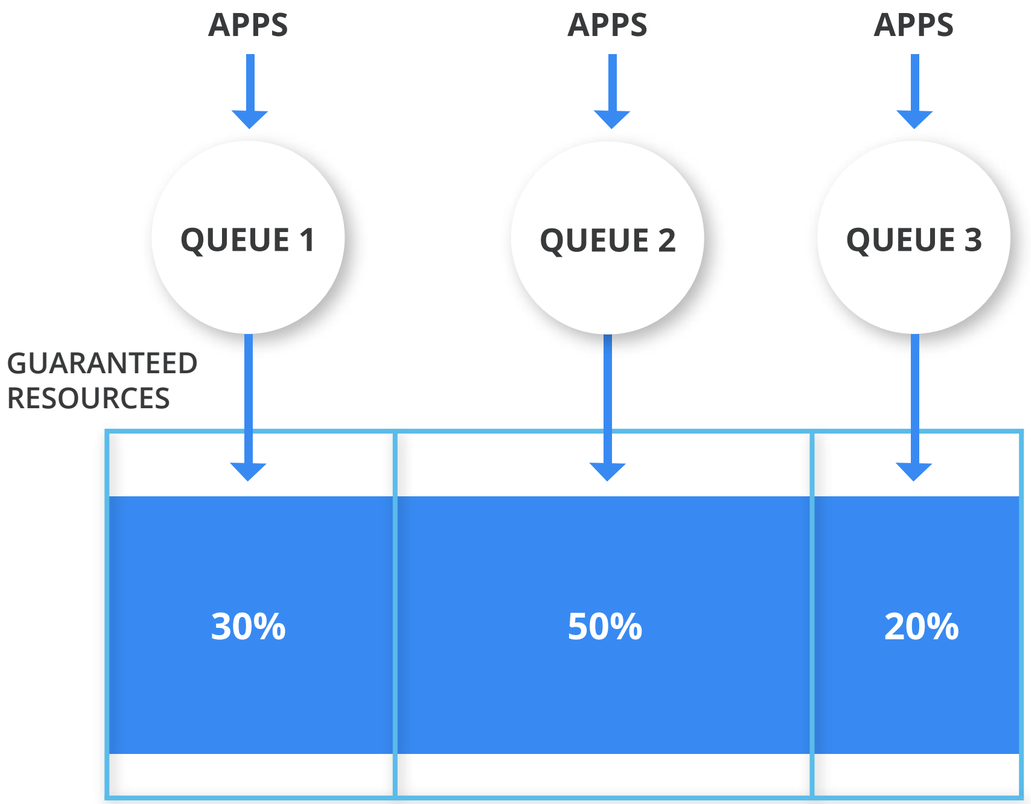
YARN SCHEDULER

Planner of Hadoop

- A Hadoop job consists of Map tasks and Reduce tasks
- Only one job in entire cluster => it occupies cluster
- Multiple customers with multiple jobs
  - *Users/jobs = “tenants”*
  - *Multi-tenant system*
- Need a way to schedule all these jobs (and their constituent tasks)
- Need to be fair across the different tenants
- Hadoop YARN has two popular schedulers
  - *Hadoop Capacity Scheduler*
  - *Hadoop Fair Scheduler*

HADOOP SCHEDULER TYPE 1

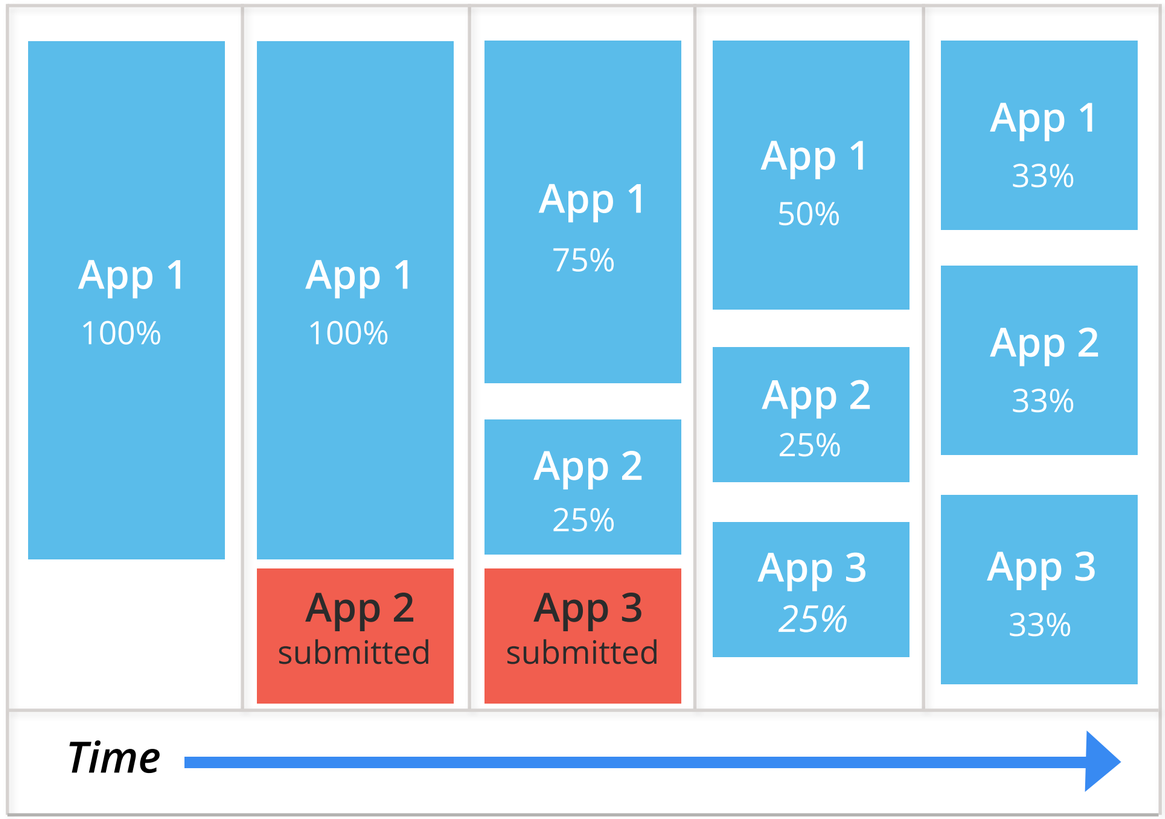
Capacity scheduler



- Contains multiple queues
- Each queue contains multiple jobs
- Each queue guaranteed some portion of the cluster capacity
  Eg:
  - *Queue 1 is given 80% of cluster*
  - *Queue 2 is given 20% of cluster*
  - *Higher-priority jobs go to Queue 1*
- For jobs within same queue, FIFO typically used
- Administrators can configure queues

HADOOP SCHEDULER TYPE 2

Fair scheduler



- All jobs get equal share of resources
- When only one job present, occupies entire cluster
- As other jobs arrive, each job given equal % of cluster
  Eg:
  - *Each job might be given equal number of cluster-wide YARN containers*
  - *Each container = 1 task of job*

HADOOP SCHEDULER TYPE 2

Fair scheduler

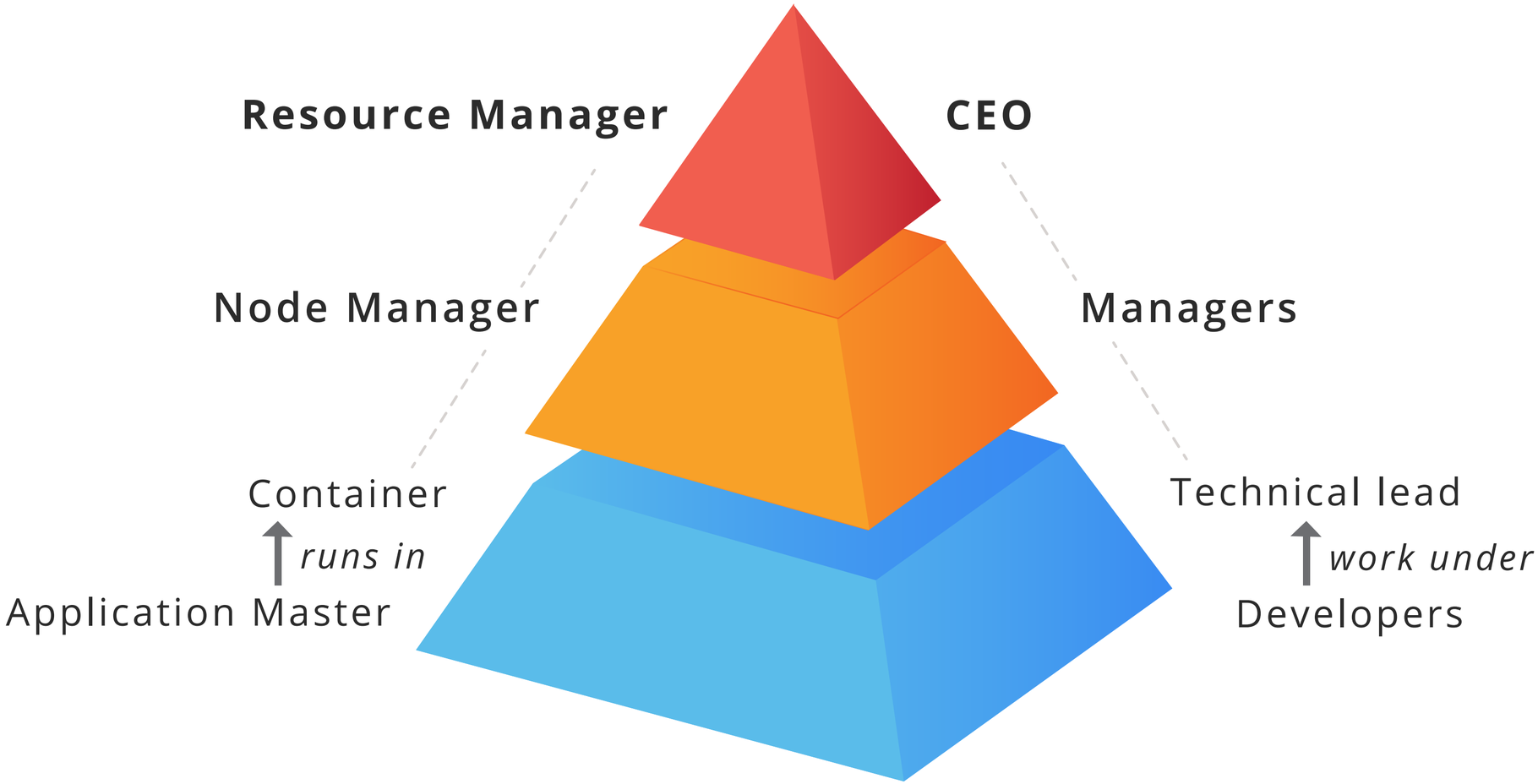
upx
Move up in life

Continued

- Divides cluster into pools (jobs)
  - *Typically one pool per user*
- Resources divided equally among pools
  - *Gives each user fair share of cluster*
- Within each pool, can use either
  - *Fair share scheduling, or*
  - *FIFO/FCS*
  - *(Configurable)*

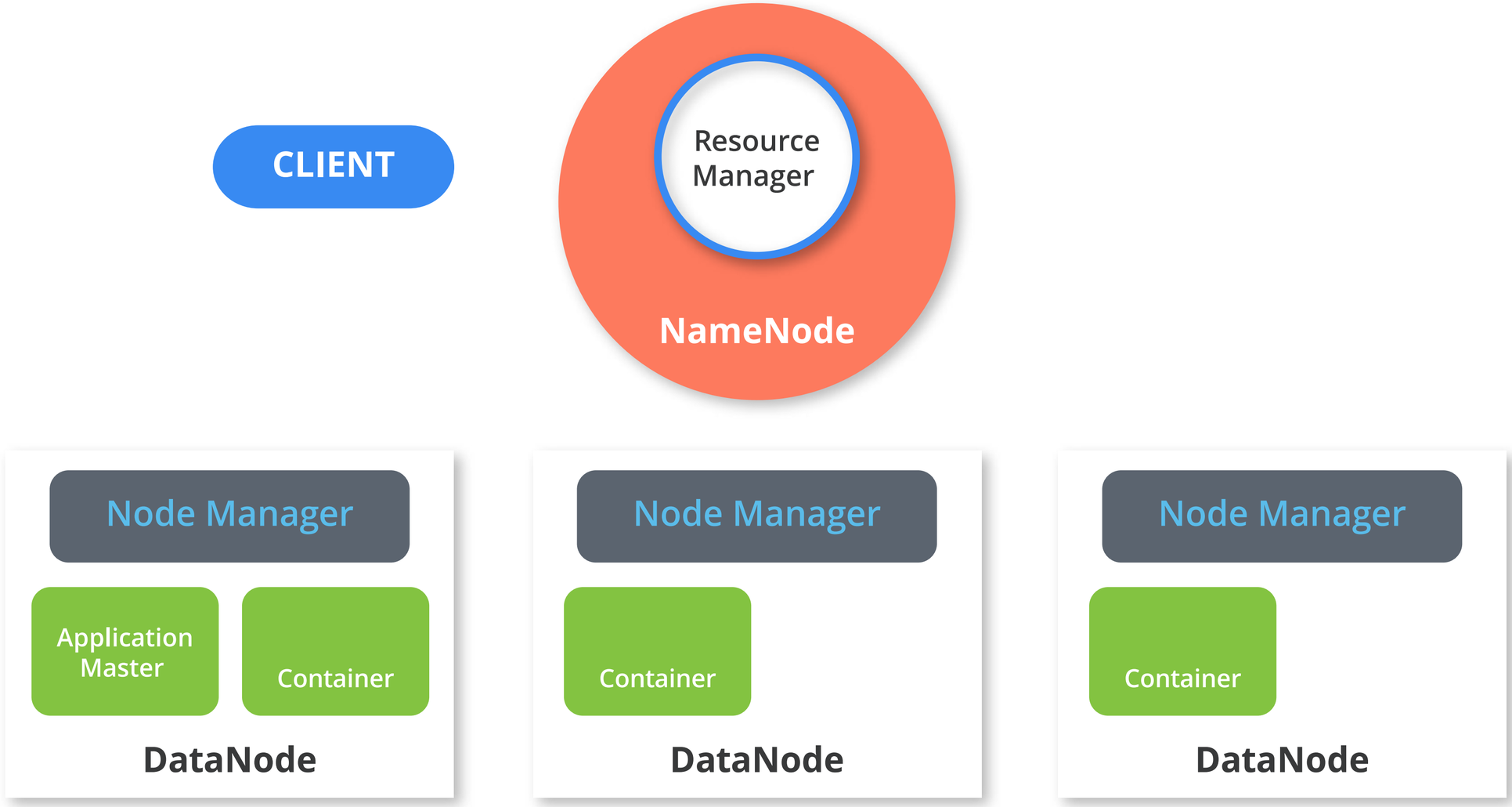
ANALOGY

YARN *architecture vs* ***Corporate Flow***



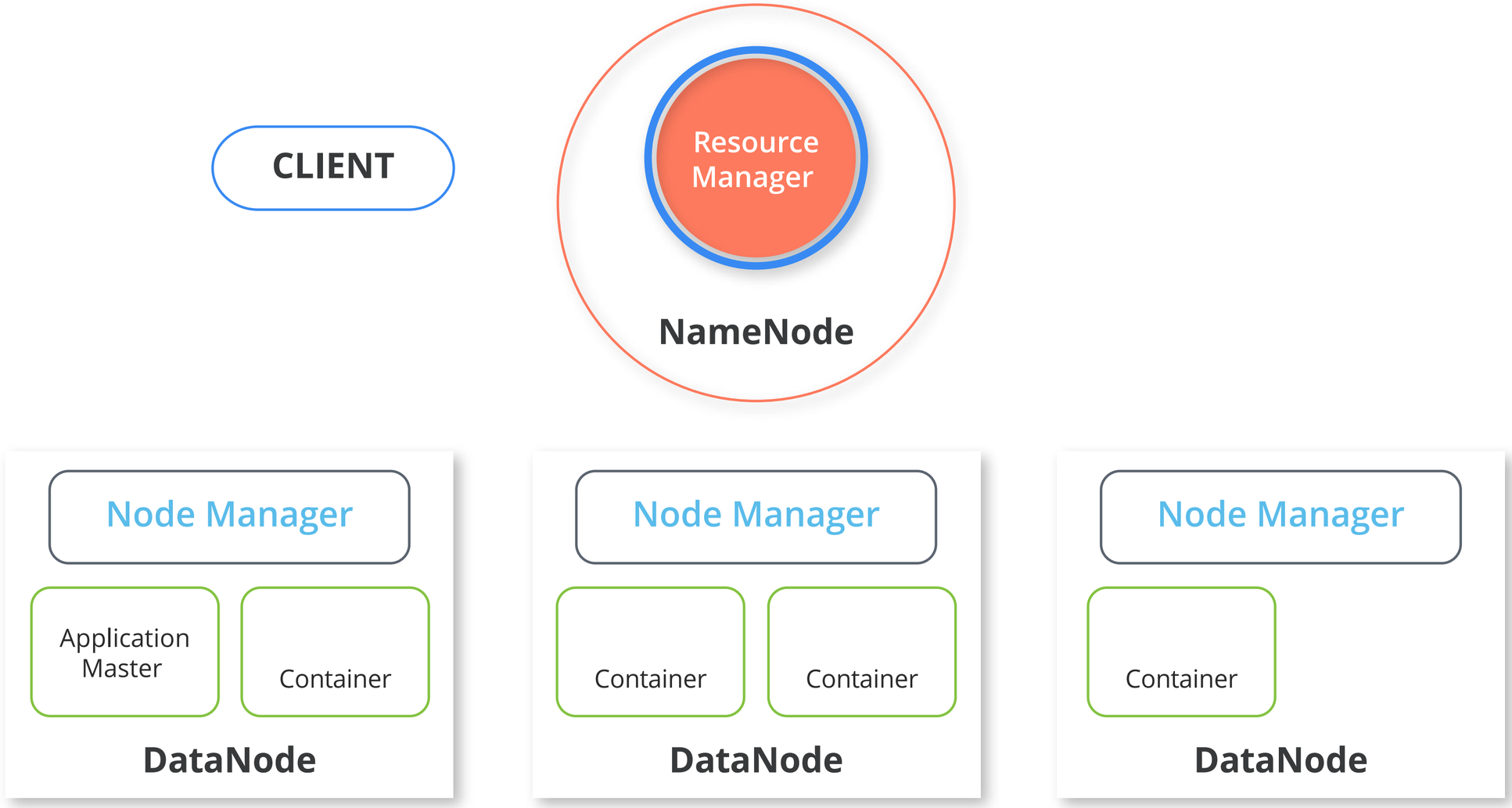
HOW THESE COMPONENTS FIT

into Hadoop cluster?



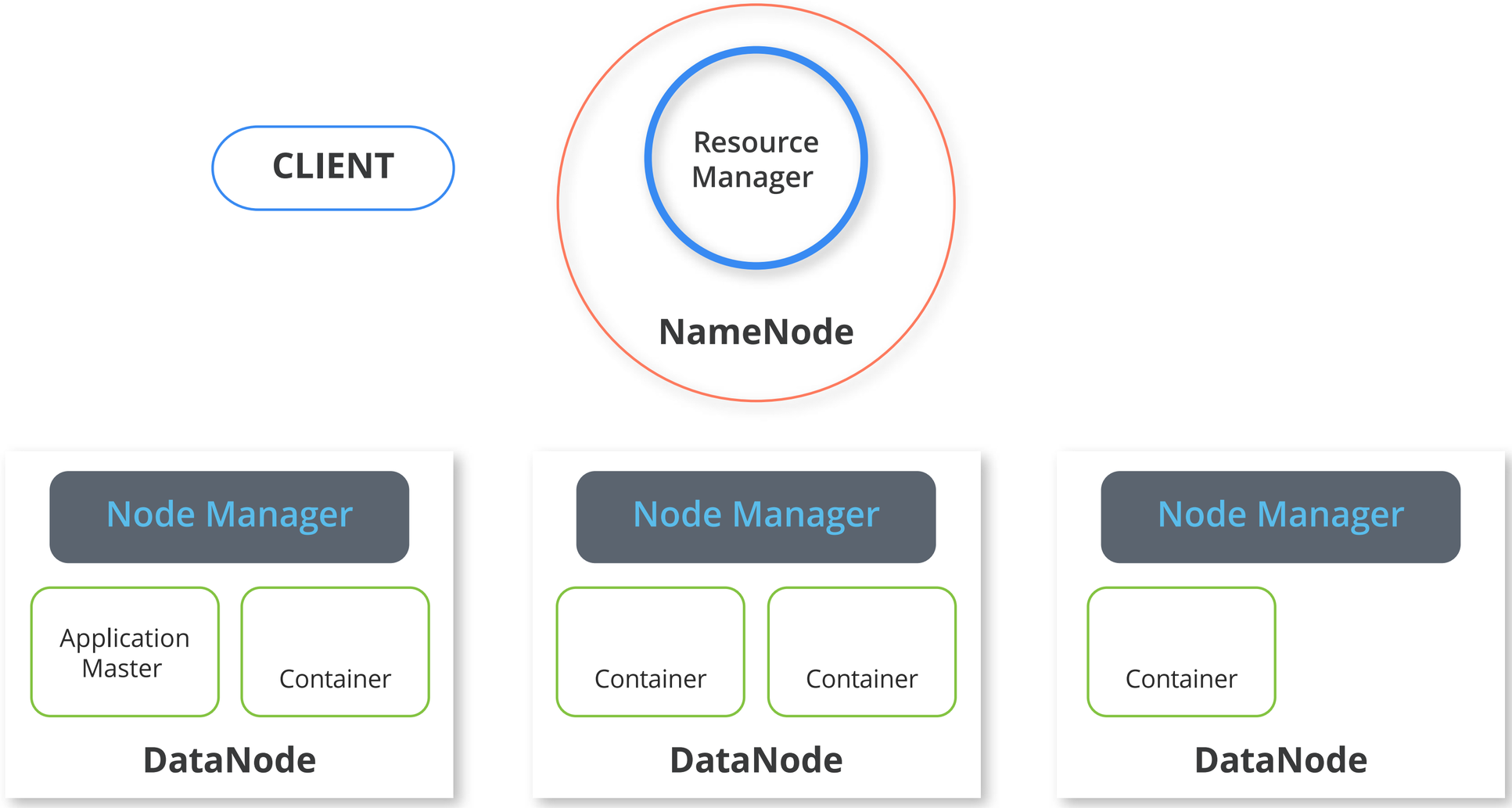
RESOURCE MANAGER

The brain of the entire process



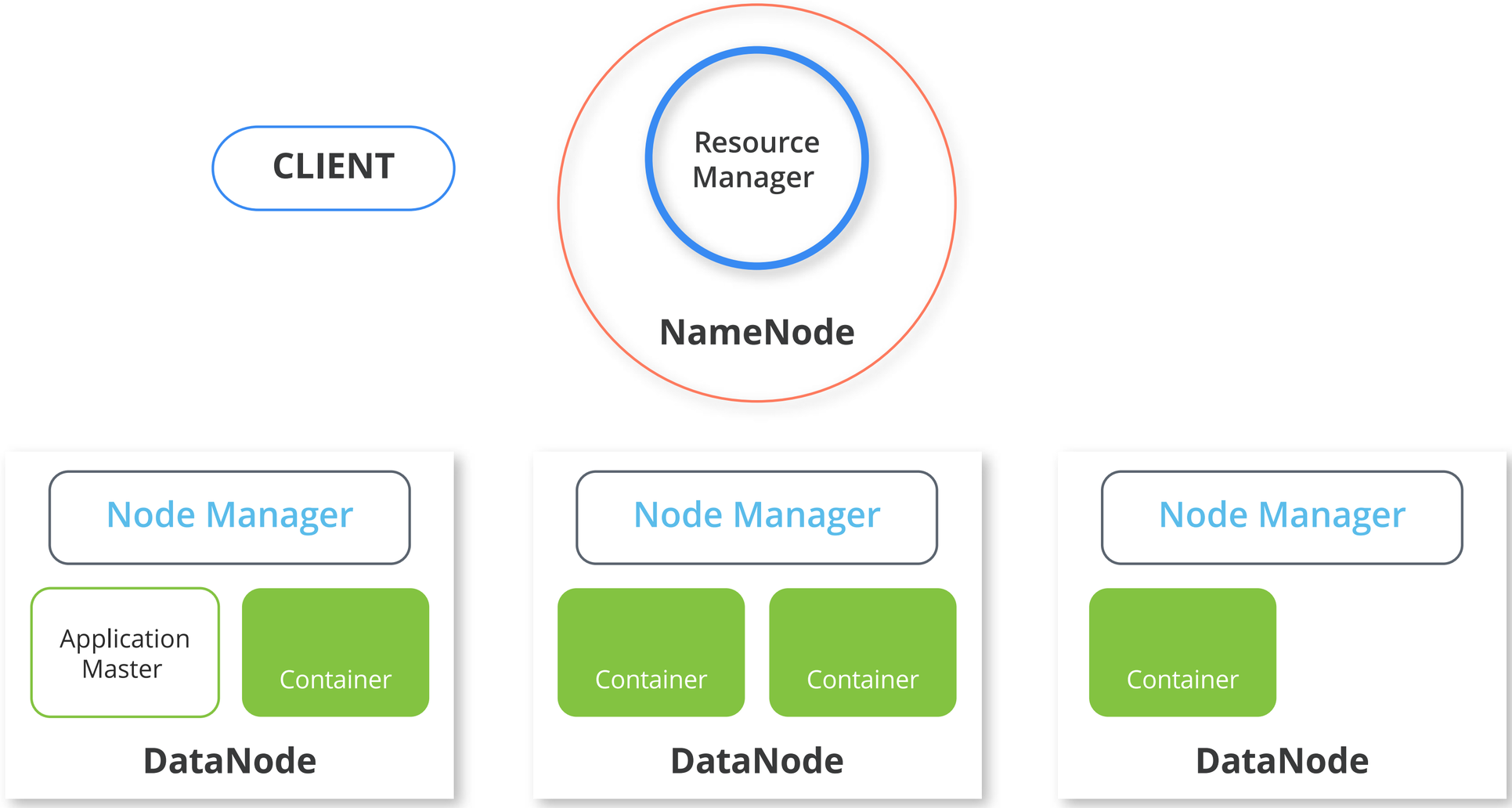
NODE MANAGER

Supervisor of the process



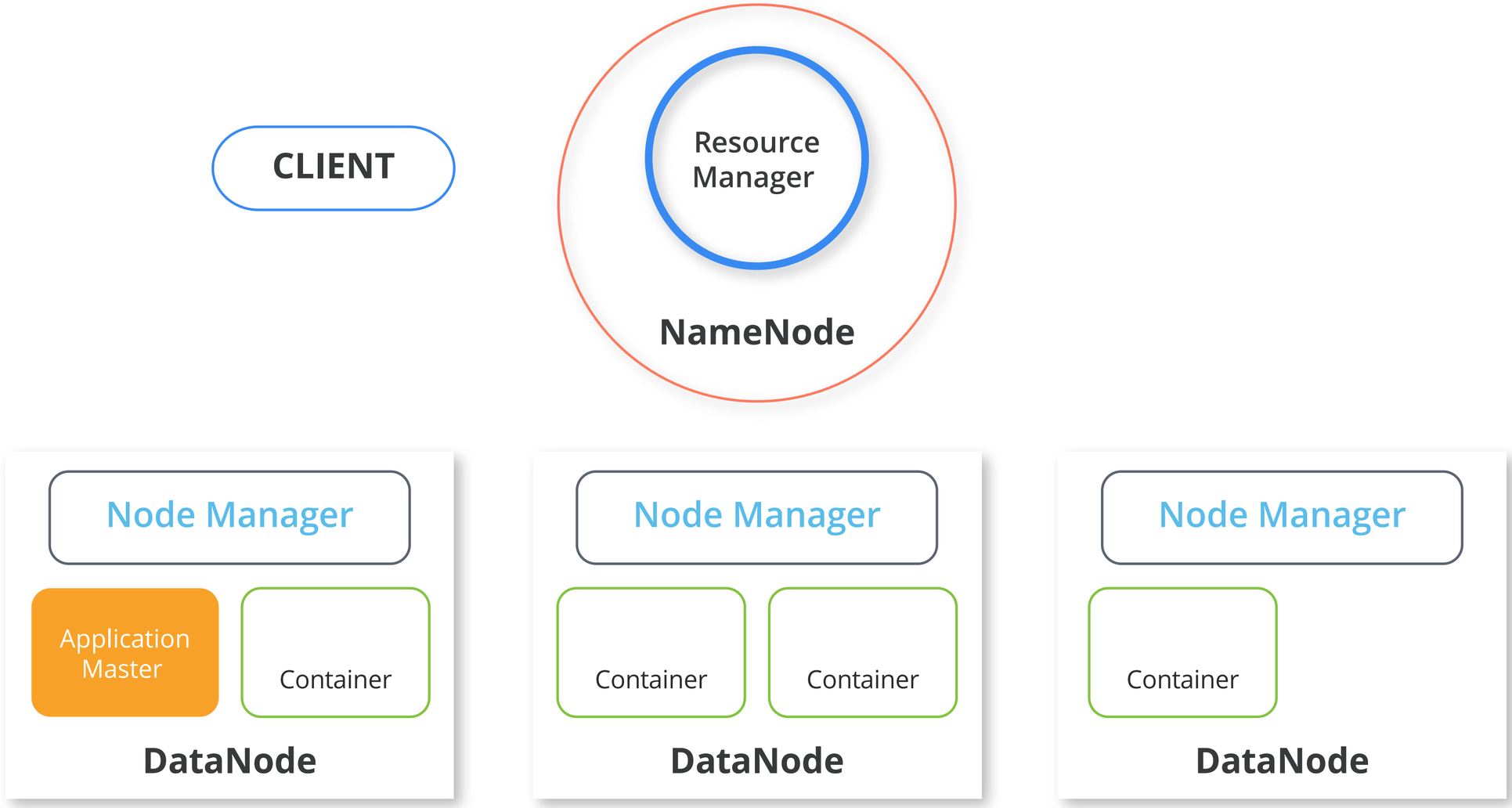
CONTAINERS

Place of execution



APPLICATION MASTER

Supplier of a process



How a job/application is submitted to YARN?

STARTING AN APPLICATION

in 6 easy steps

Step 1

Client submits a job/application

Step 2

RM asks Node Manager to launch Application Master

Step 3

Node Manager launches Application Master

Step 4

Application Master requests containers

Step 5

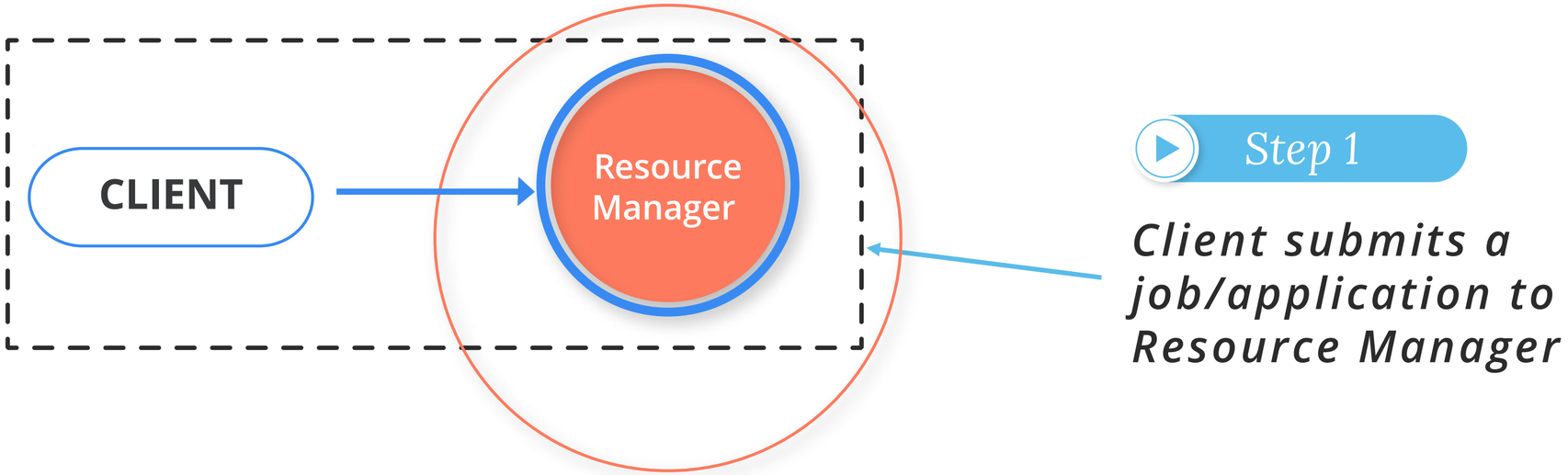
RM assigns containers

Step 6

Application Master launches application

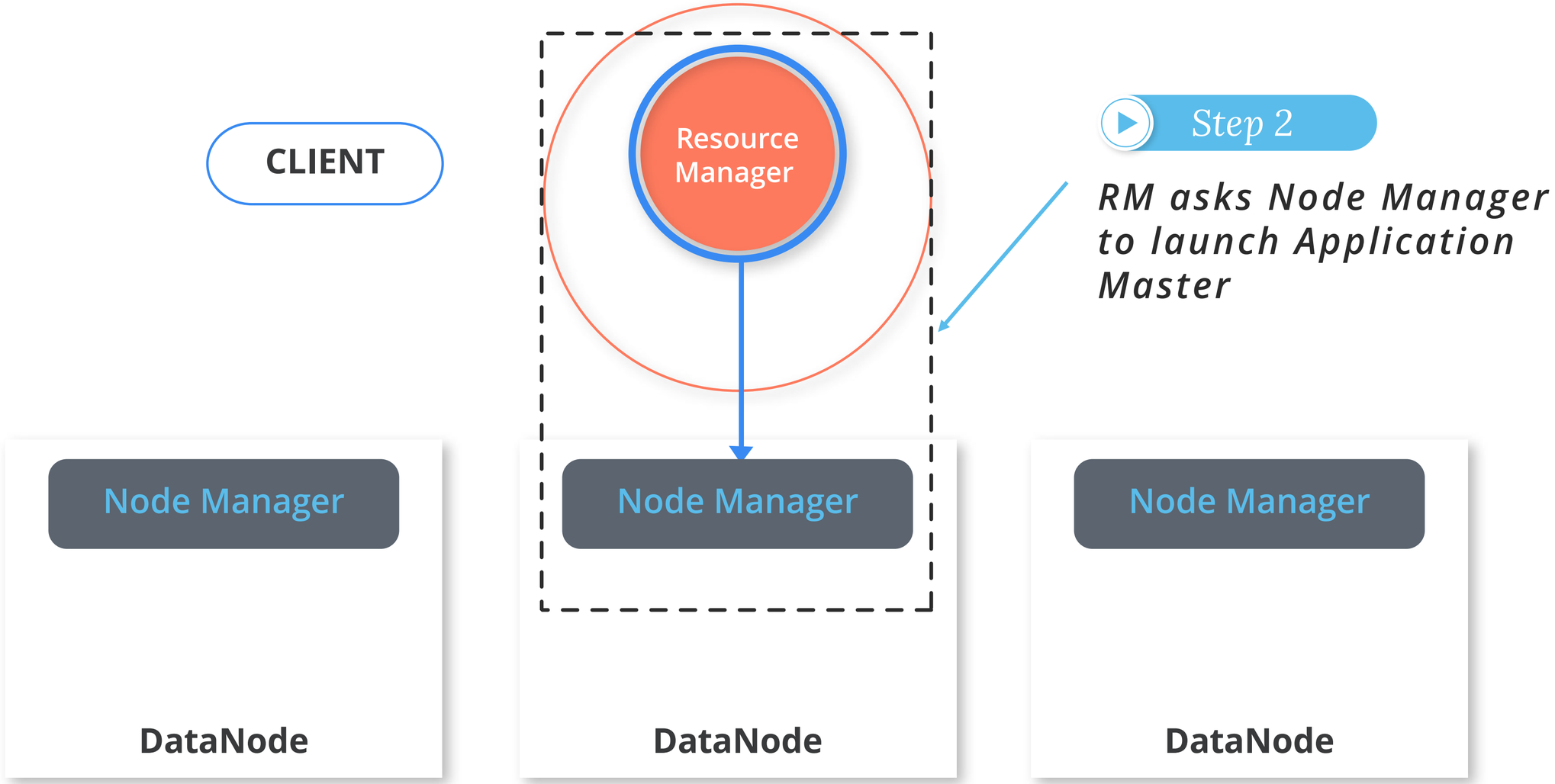
STEP 1

Client submits a job/application to RM



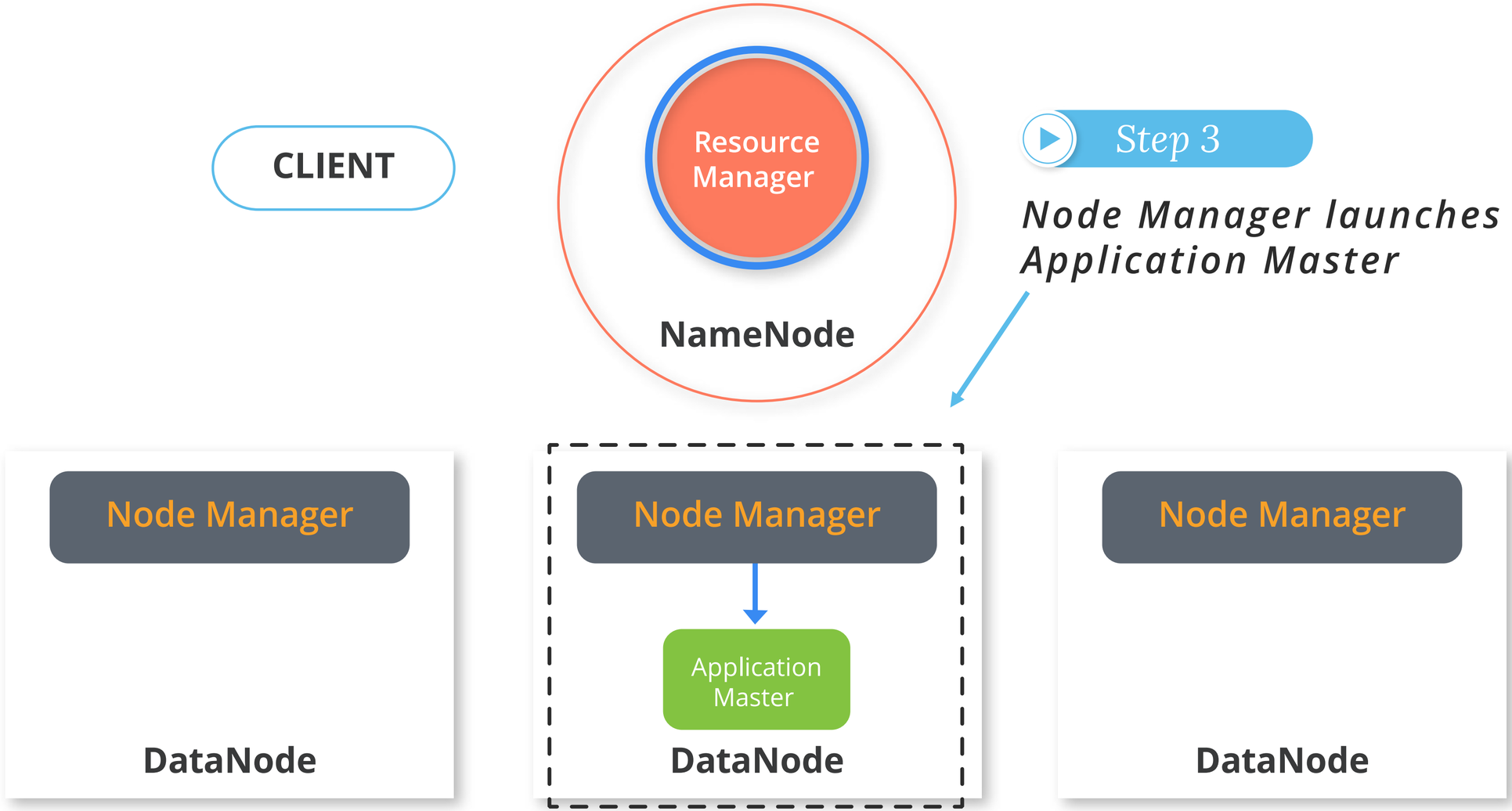
STEP 2

RM asks Node Manager to launch Application Master



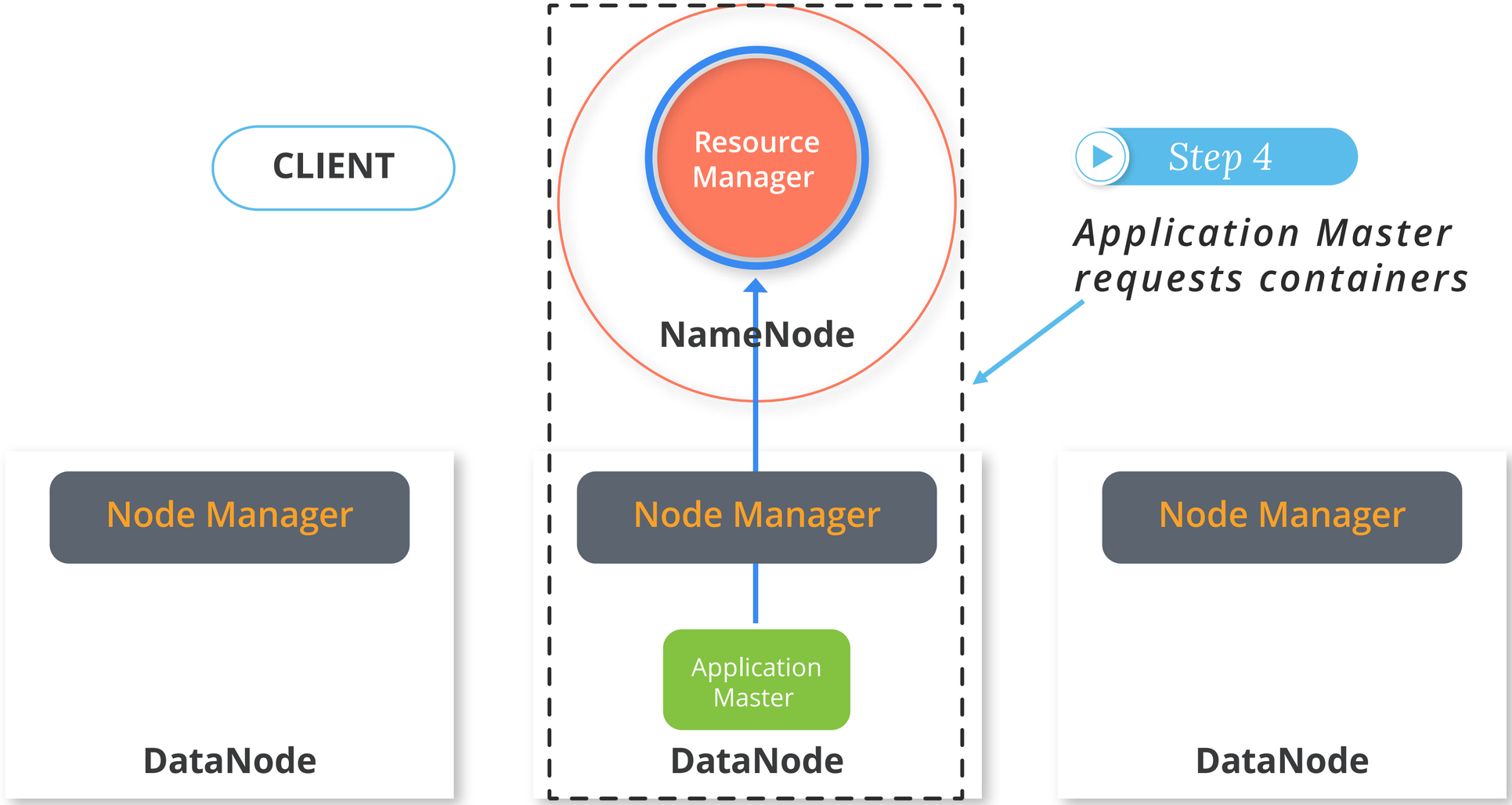
STEP 3

Node Manager launches Application Master



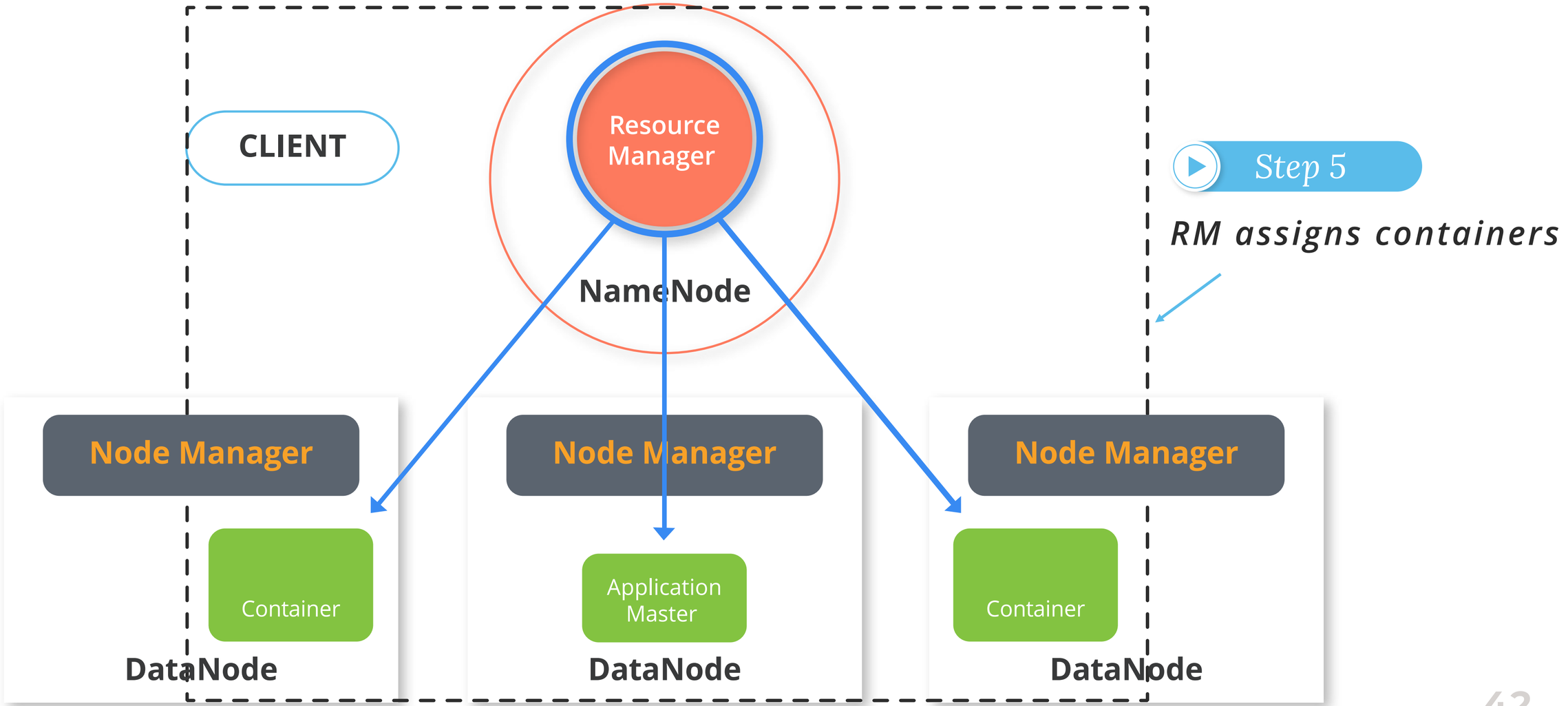
STEP 4

Apllication Master requests containers



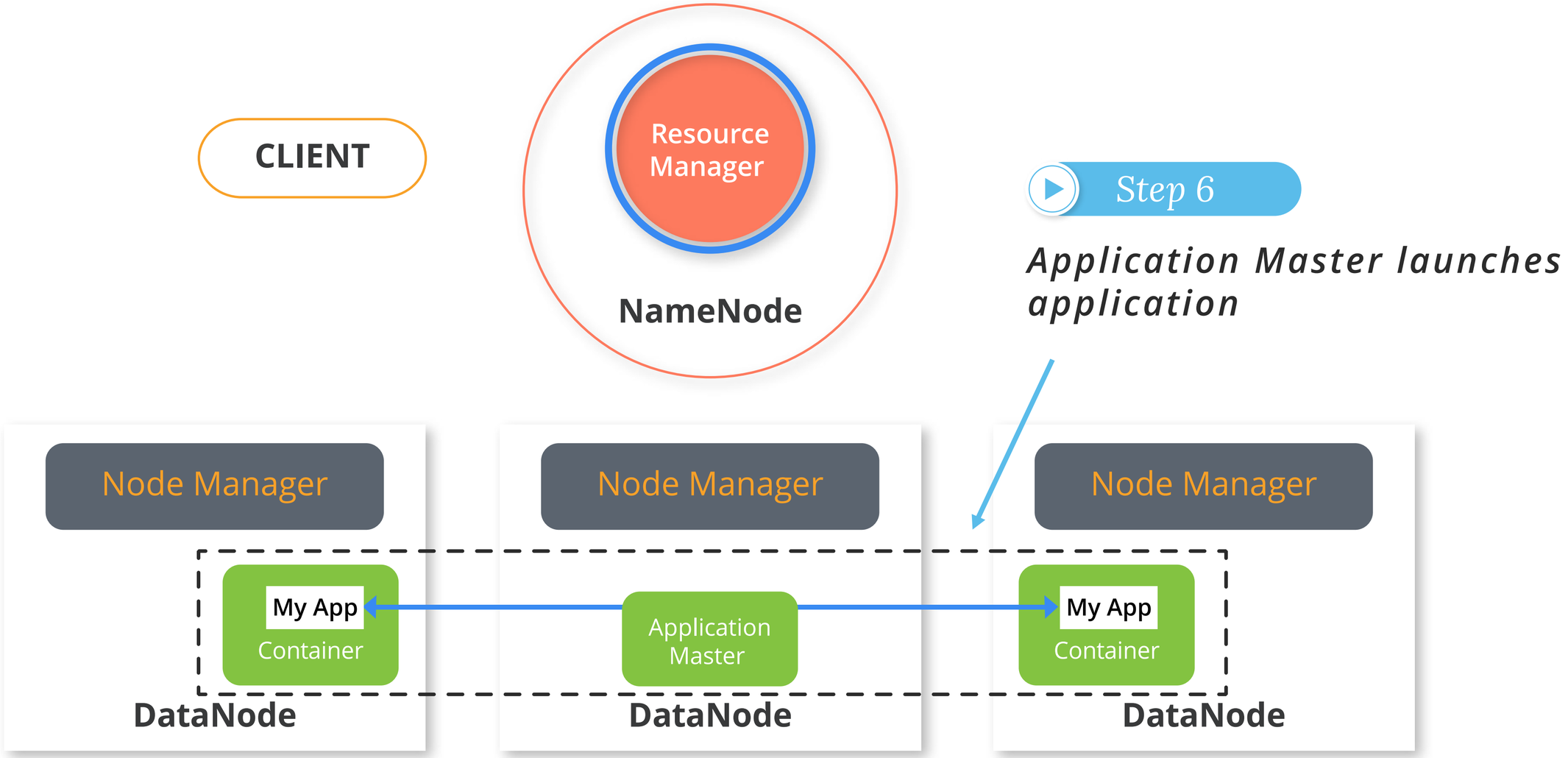
STEP 5

RM launches containers in DataNode and container IDs to AM



STEP 6

Application Master launches application in the containers

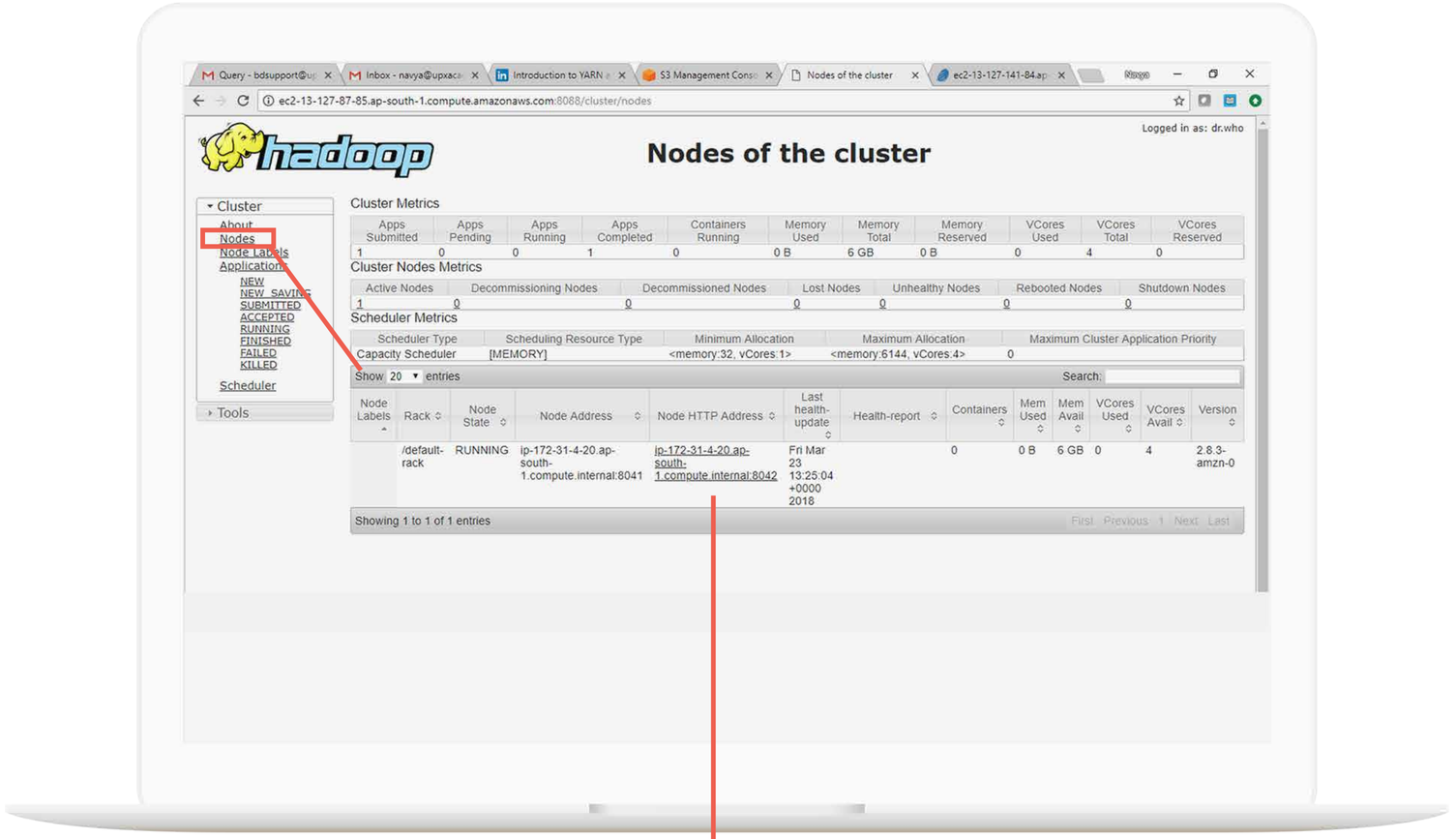


Congratulations!
Your Application is now running

Now, we will have a look at the Resource manager UI

RESOURCE MANAGER UI

https://ipaddress:8088

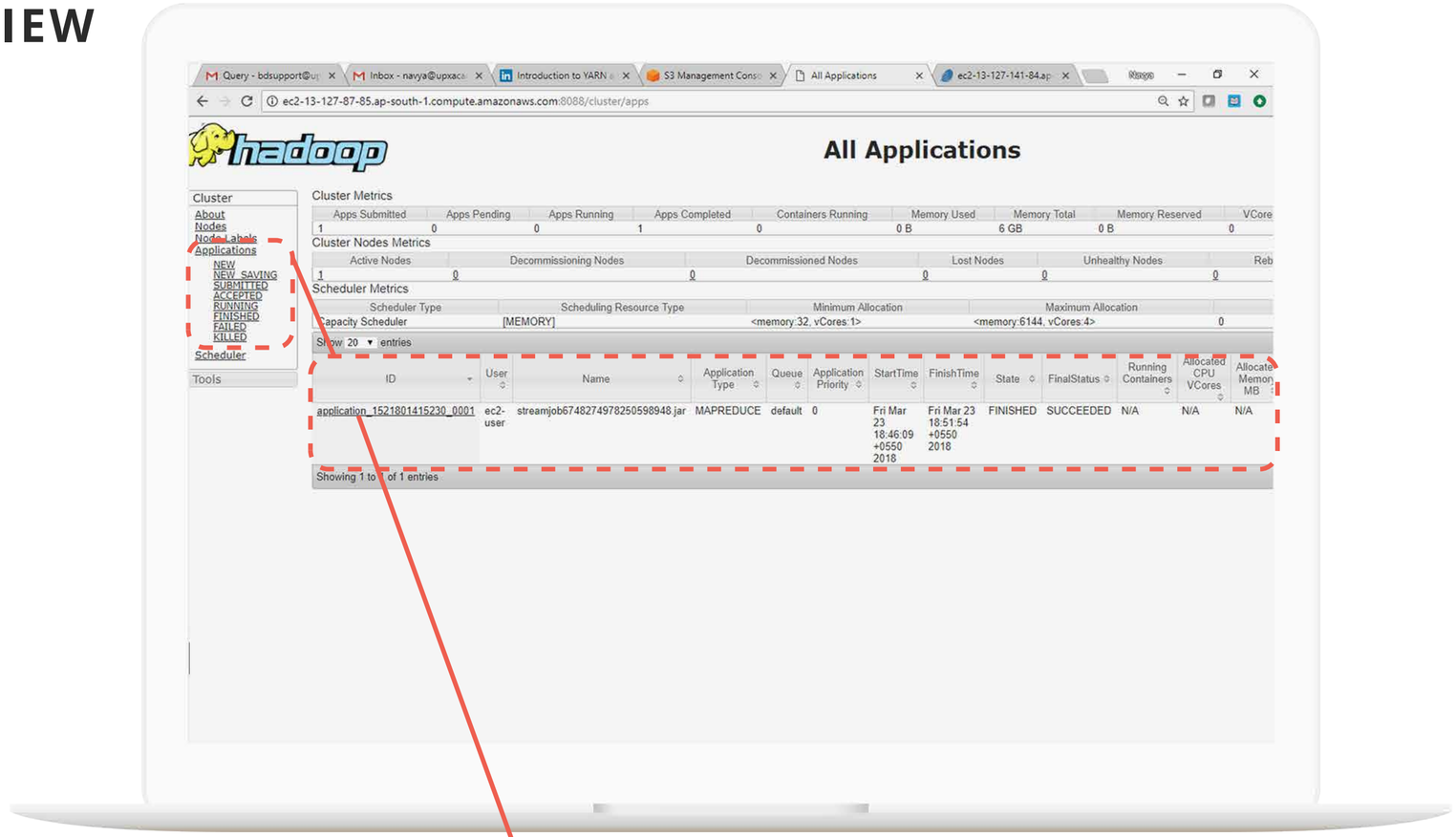


Link to Node Manager UI

RESOURCE MANAGER UI

Applications

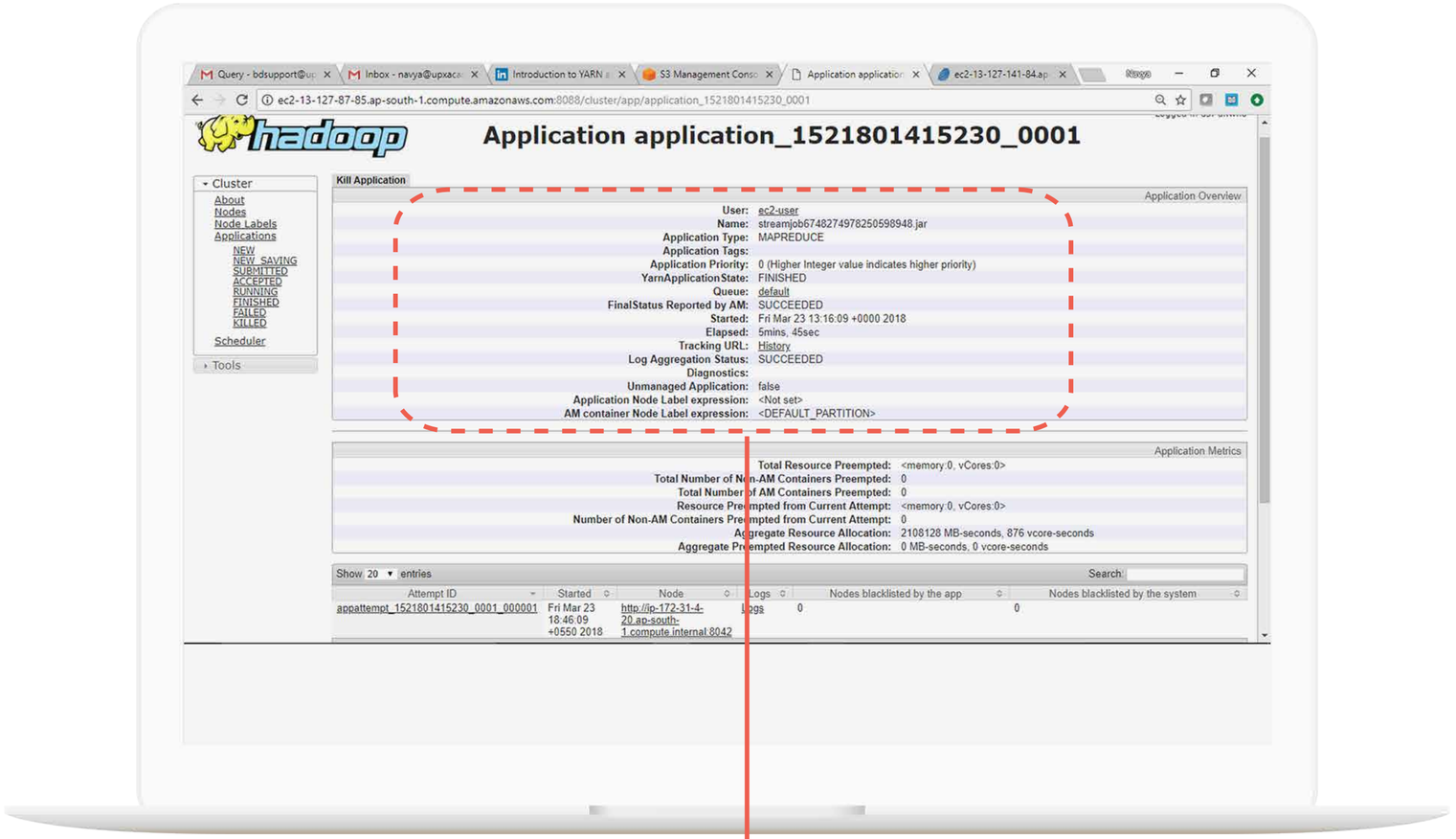
CLUSTER OVERVIEW



Link to Application details

RESOURCE MANAGER UI

Application Details



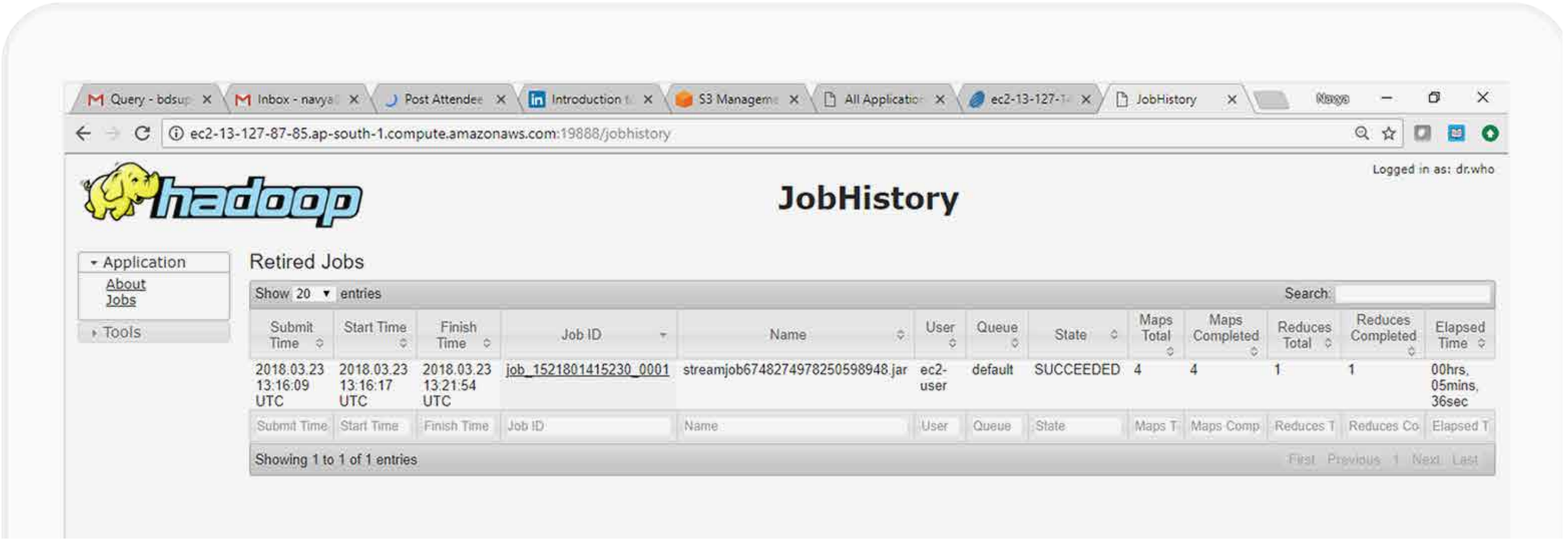
Link to Application details

MR APPMASTER UI

Tasks

MR JOB HISTORY SERVER

- **YARN does not keep track of job history**
- **MapReduce job history server**
  - *Archives jobs metrics & metadata*
  - *Can be accessed through Job history UI or HUE*



DISCLAIMER

All images used here in the deck are obtained from various sources online, through Google image search. We do not own any copyright claims.
The deck is prepared for educational purpose and no copyright violations were intended.

upx
Move up in life

www.upxacademy.com

bdsupport@upxacademy.com

1800-123-1260

Follow Us On