Big Data Hadoop and Spark Developer

YARN: Introduction

Learning Objectives

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

- Describe YARN and its features along with advantages
- Explain the elements of YARN architecture
- Identify how YARN runs on applications
- Classify tools for YARN developers
- List useful YARN commands

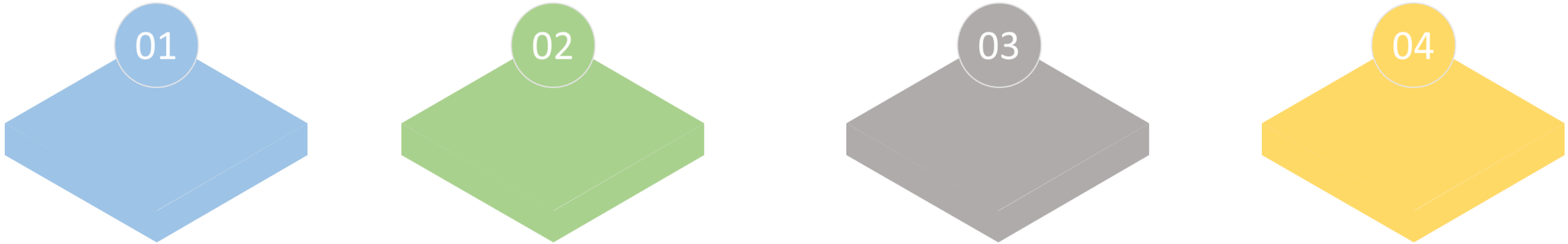
YARN: Yet Another Resource Negotiator

What Is YARN?

- YARN is the acronym for *Yet Another Resource Negotiator.*
- YARN is a Hadoop ResourceManager that was established by dividing the MapReduce processing engine and management function.
- The basic idea behind YARN is to separate the resource management and task scheduling, also monitor functions in independent daemons.

Why Was YARN Implemented?

It is essential to know the limitations of the Hadoop 1.0 framework to understand the advantages of YARN.



Availability

Resource utilization

Failure in cascading

Running non-MapReduce applications

Why Was YARN Implemented?

Availability: In Hadoop 1.0, the job tracker is the only point of availability. This means if the job tracker fails, all tasks are automatically restarted.

Resource utilization: There are two predefined slots. If one of the slots is full and the others are empty, the empty slots will be reserved and remain idle. This could result in a resource utilization problem.

Why Was YARN Implemented?

Failure in cascading: When the number of nodes in this framework exceeds 4000, one of the primary challenges arises. That is cascading failure occurs in this situation.

Running non-MapReduce applications: Job tracker is a MapReduce framework-based application. The problem arises when a non-MapReduce application tries to execute in this framework. Some of the most typical challenges that develop will include ad-hoc inquiry and real-time analysis.

Advantages of YARN

Following are the few advantages of YARN:

Advantages of YARN

01

Scalability: Hadoop can extend and manage thousands of nodes and clusters due to the Scheduler in the YARN architecture’s Resource Manager.

02

Compatibility: YARN maintains the functionality of the existing MapReduce applications while also making them compatible with Hadoop 1.0.

Advantages of YARN

03

Cluster utilization: As YARN supports dynamic cluster utilization in Hadoop, it will also enable optimized cluster utilization.

04

Multi-tenancy: It enables multiple engine access, providing organizations with the benefit of multi-tenancy.

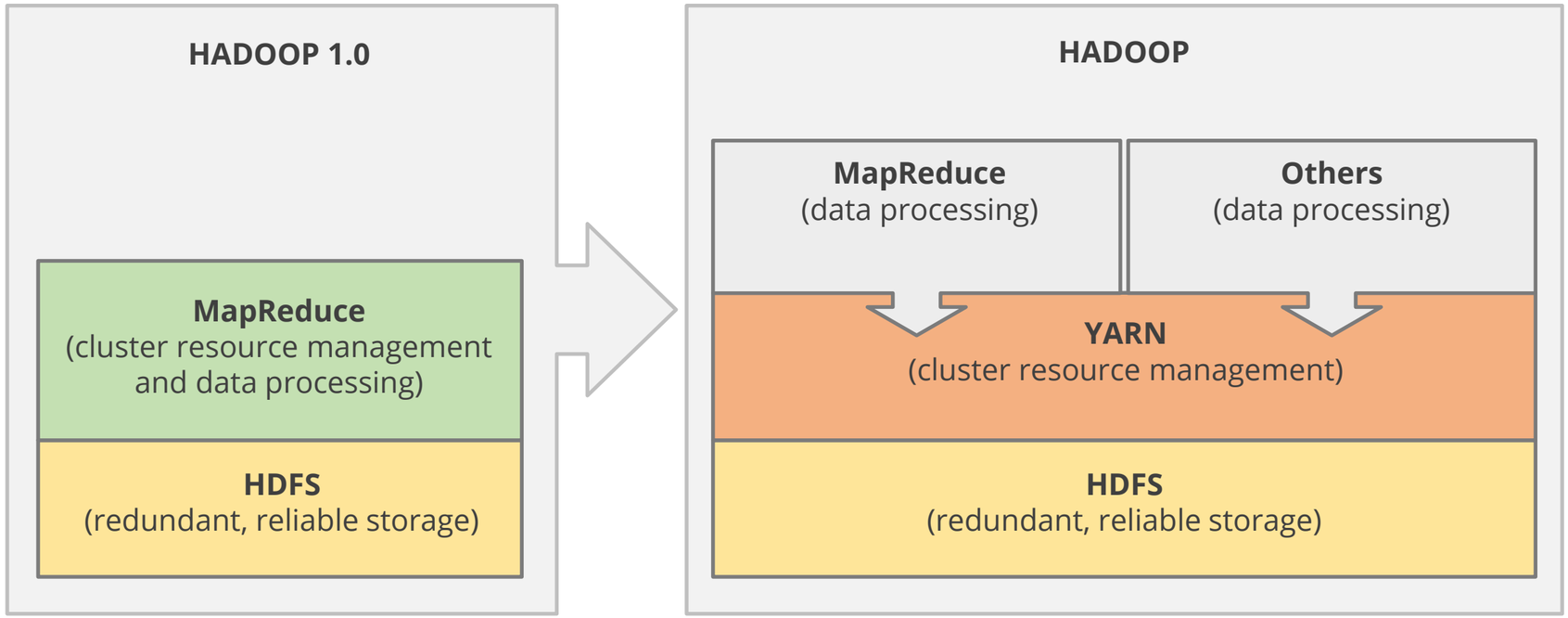
Need for YARN

- YARN tends to make Hadoop more available by allowing users to process.
- YARN runs data that are stored in HDFS like a batch, stream, interactive, and graph processing.
- YARN helps the execution of many distributed applications other than MapReduce.

Need for YARN

Users could develop MapReduce applications using scripting languages before 2012.

Users were able to work with different processing models in addition to MapReduce since 2012.



Use Case: YARN

YARN: Use Case

YAHOO!

Yahoo was the first company to implement Hadoop, and it continues to set the standard high in the Hadoop ecosystem. Due to MapReduce limitations, Yahoo struggled to handle iterative and stream processing of data on the Hadoop infrastructure in late 2012.

YARN: Use Case

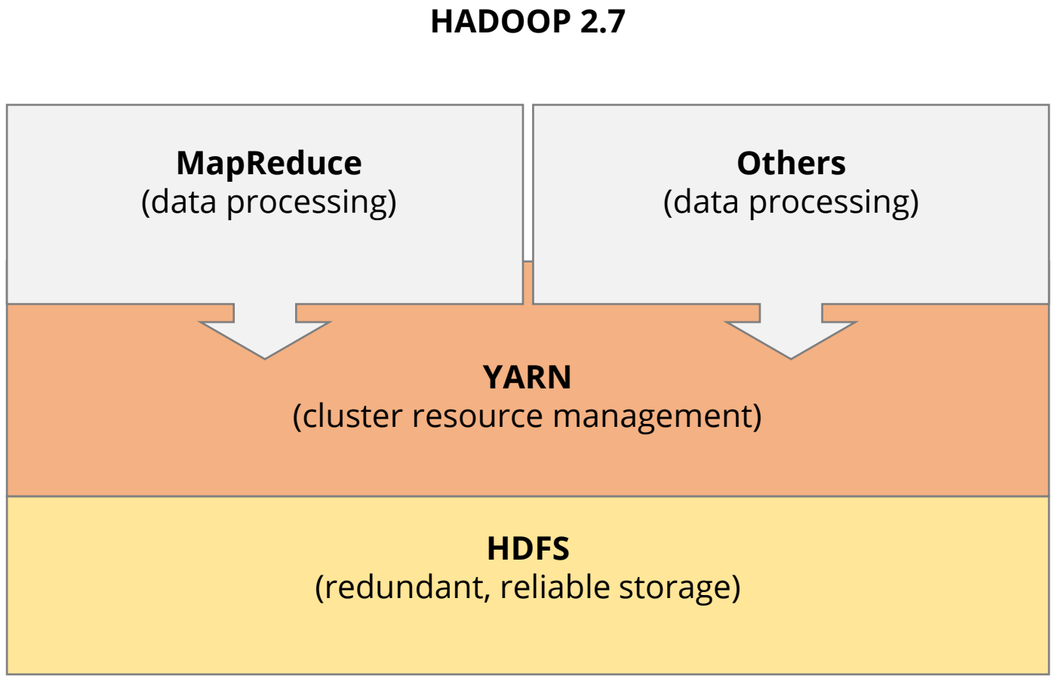
YAHOO!

- 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 Infrastructure

YARN Infrastructure

The YARN Infrastructure is responsible for delivering computational resources for application execution, such as CPUs and memory.



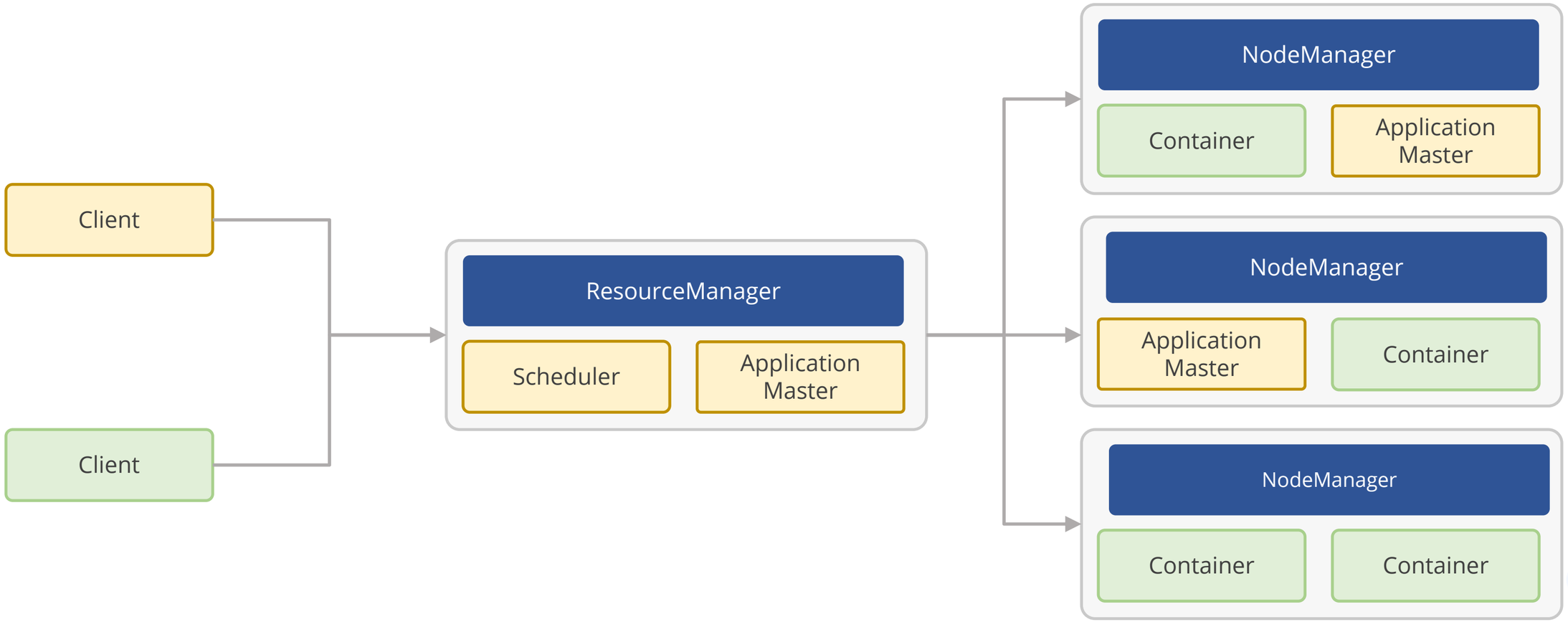
YARN Infrastructure

- HDFS and the YARN infrastructure are fully independent. The former gives resources for an application to run, while the latter provides storage.
- The MapReduce framework is one of many that can run on YARN.
- The core idea behind MapReduce version 2 is to separate the two key functions of resource management and job scheduling.

YARN Architecture

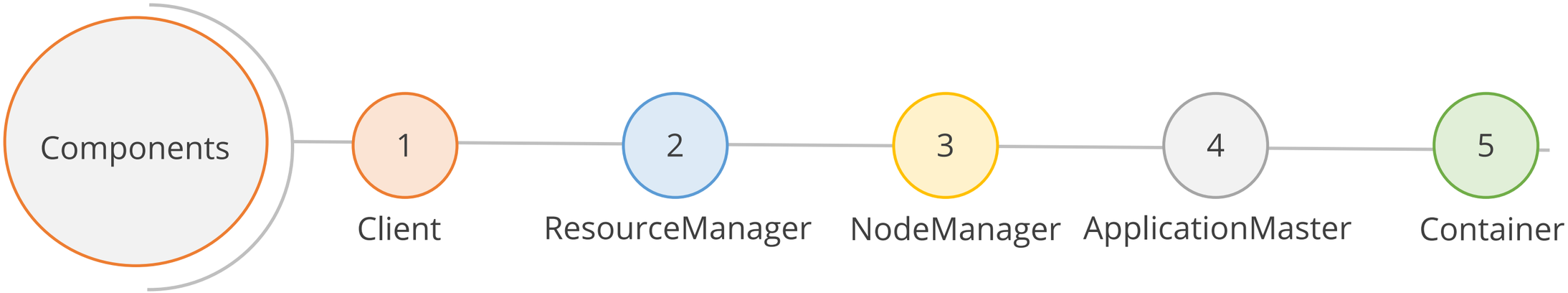
YARN Architecture

The following diagram shows the YARN architecture:



YARN Architecture

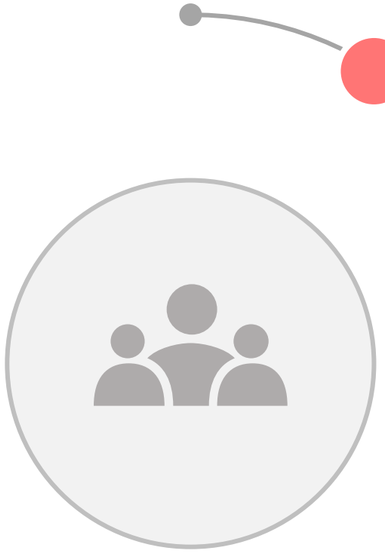
The following are the main components of the YARN architecture:



Client

- The client contacts the ResourceManager in the process.
- The client is used for submitting the MapReduce jobs and reading the result.
- The client also monitors the status of the application.

ResourceManager



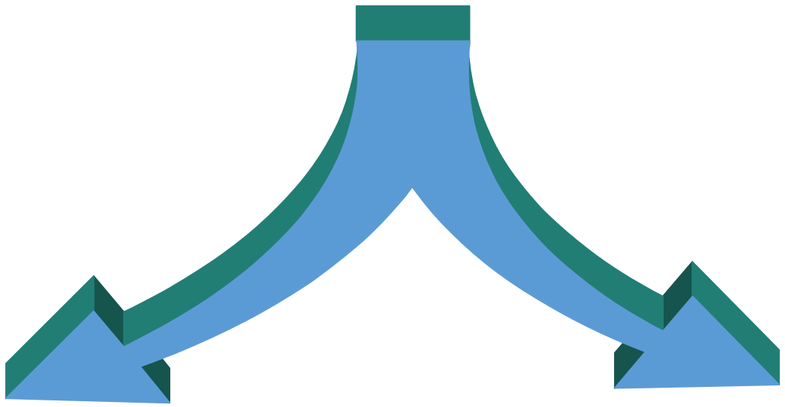
The ResourceManager is the master daemon of YARN.

The ResourceManager is responsible for assigning and managing resources across all applications.

Whenever ResourceManager receives a processing request, it sends it to the appropriate NodeManager, who then assigns resources to complete the request.

ResourceManager

The ResourceManager is consists of the following two parts:



Scheduler

Application manager

Scheduler

The Scheduler performs scheduling based on the application and the available resources.

It is a pure scheduler, which means it does not do anything else like monitoring or tracking, and it does not guarantee a task restart if it fails.

Scheduler

The scheduler assigns resources for running applications based on capacity and queue.

To partition cluster resources, the YARN Scheduler includes plugins like a Capacity Scheduler and a Fair Scheduler.

Application Manager

- The responsibilities of an Application Manager include accepting job submissions and negotiating the first container with resource management.
- If a job fails, the Application Manager additionally restarts the ApplicationMaster container.
- The Application Manager oversees the operation of the ApplicationMasters in a cluster and offers support for restarting the ApplicationMaster container in the event of a failure.

NodeManager

NodeManager is responsible for each node in the Hadoop cluster and administers the application and workflow for that node.

It is the primary responsibility is to keep track of the ResourceManager.

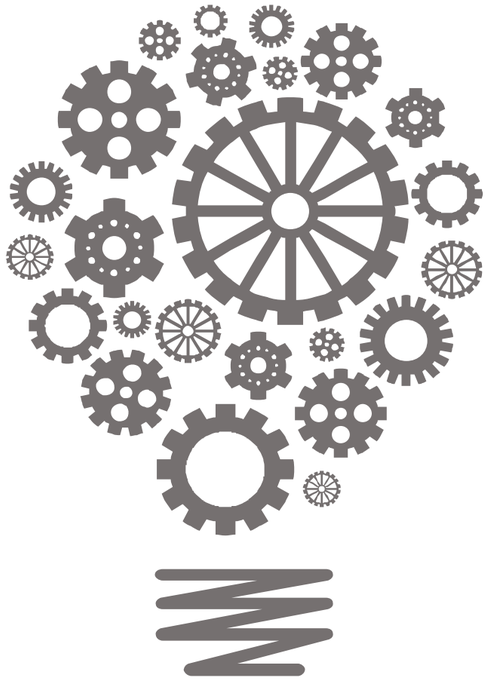
NodeManager

It connects to the ResourceManager and sends heartbeats with the node's health status.

It tracks resource utilization, manages logs, and destroys Containers at the request of ResourceManager.

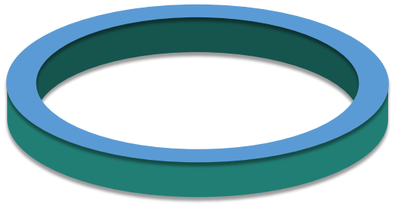
It is also in charge of starting the Container process when the ApplicationMaster requests it.

ApplicationMaster

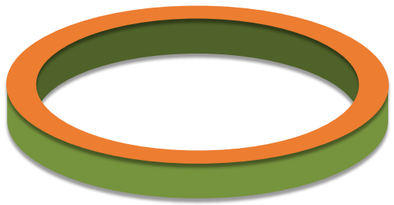


- A single job is submitted to a framework as an application.
- Each of these applications has its own ApplicationMaster, which is a framework-specific entity.
- It is the procedure that controls faults and coordinates an application's execution in a cluster.
- The ApplicationMaster is responsible for negotiating resources with the ResourceManager, tracking the status of a specific application, and monitoring its progress.

ApplicationMaster



The ApplicationMaster sends a Container Launch Context (CLC) to the node management to request the Container.



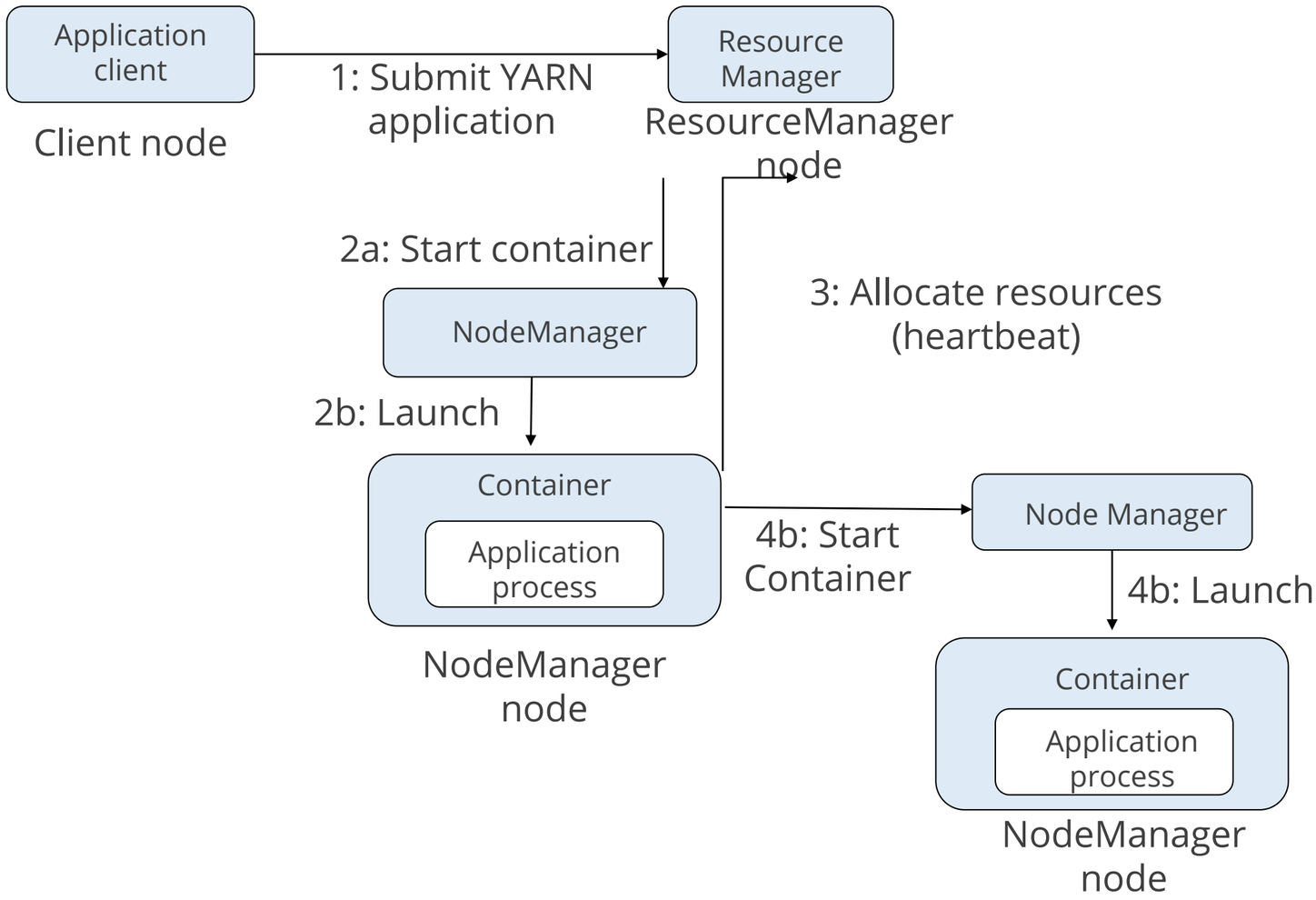
The CLC contains everything required for an application to run. When the program is launched, it sends a health report to the ResourceManager regularly.

Container

- On a single node, a Container is a collection of physical resources such as RAM, CPU cores, and disc.
- A Container lifecycle record contains information such as a map of environment variables, dependencies stored in remotely accessible storage, security tokens, the payload for NodeManager services, and the command required to create the process for YARN Containers.
- It enables an application to access a set number of resources (memory, CPU, and others) on a given host.

Application Workflow

The following diagram represents the application workflow:



Application Workflow

Running a YARN application involves five steps:

1. The ResourceManager receives an application from the client.
2. The ResourceManager assigns a Container.
3. The ApplicationMaster contacts the NodeManager.

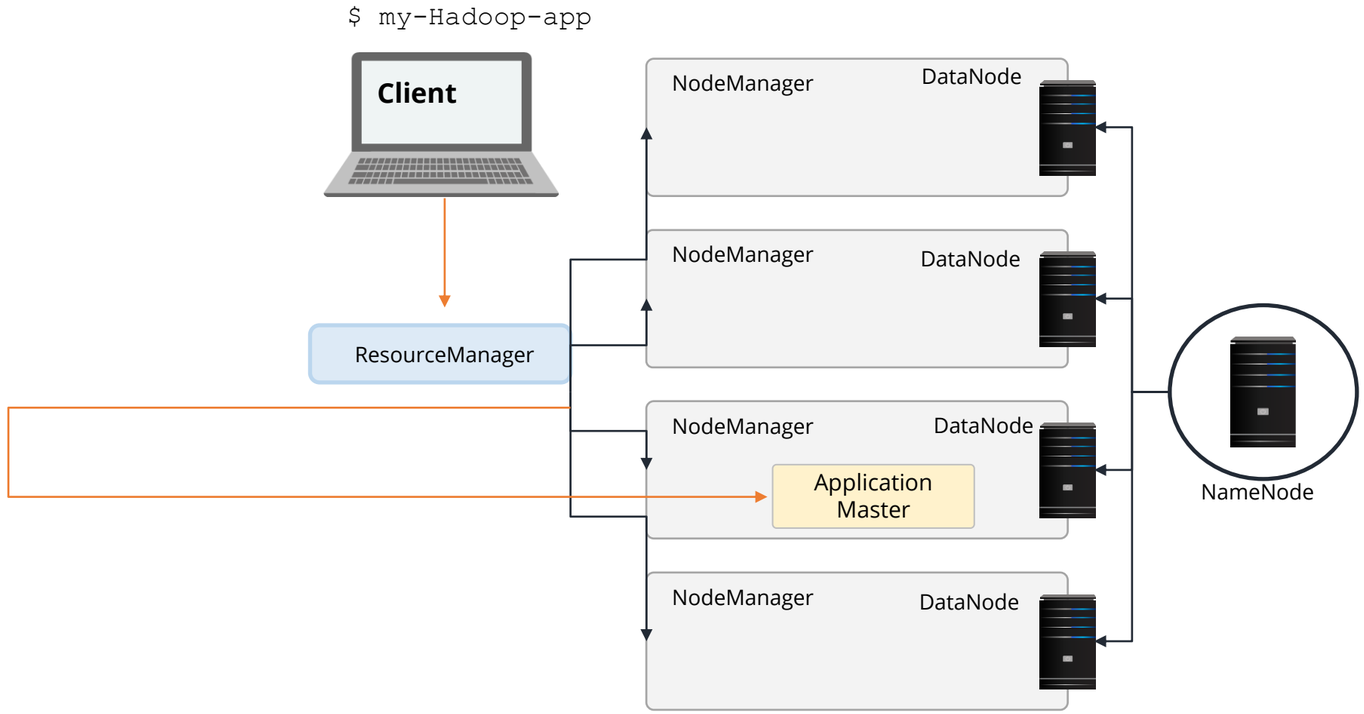
Application Workflow

4	The NodeManager launches the Container.
5	The Container executes the ApplicationMaster.

Application Workflow

Step 1: The ResourceManager receives an application from the client.

By entering the Hadoop jar command, users can submit applications to the ResourceManager.



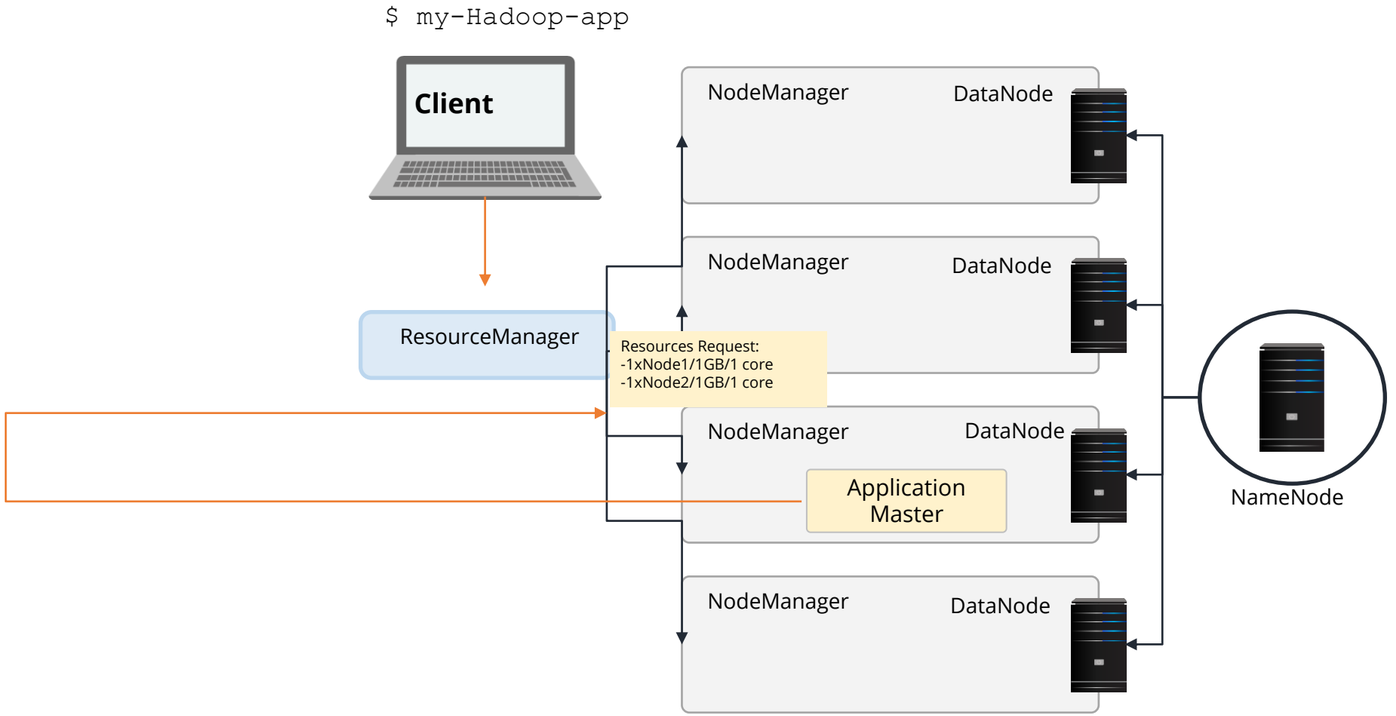
Application Workflow

- On the NodeManager, the ResourceManager keeps track of the cluster's applications and available resources.
- The ResourceManager selects the next application in line for a share of the cluster's resources.
- Many factors influence the decision, including queue capacity, access control lists, and fairness.

Application Workflow

Step 2: The ResourceManager assigns a Container.

One of the first decisions the Scheduler takes when the ResourceManager accepts a new request is to choose a Container. Then, the ApplicationMaster takes over and is responsible for the application's lifecycle.



Application Workflow

- The Container makes resource requests to the ResourceManager, requesting Containers to run the app's tasks.
- A resource request is essentially a request for a set of Containers that meet certain resource requirements, such as the amount of memory and CPU shares stated in MB.

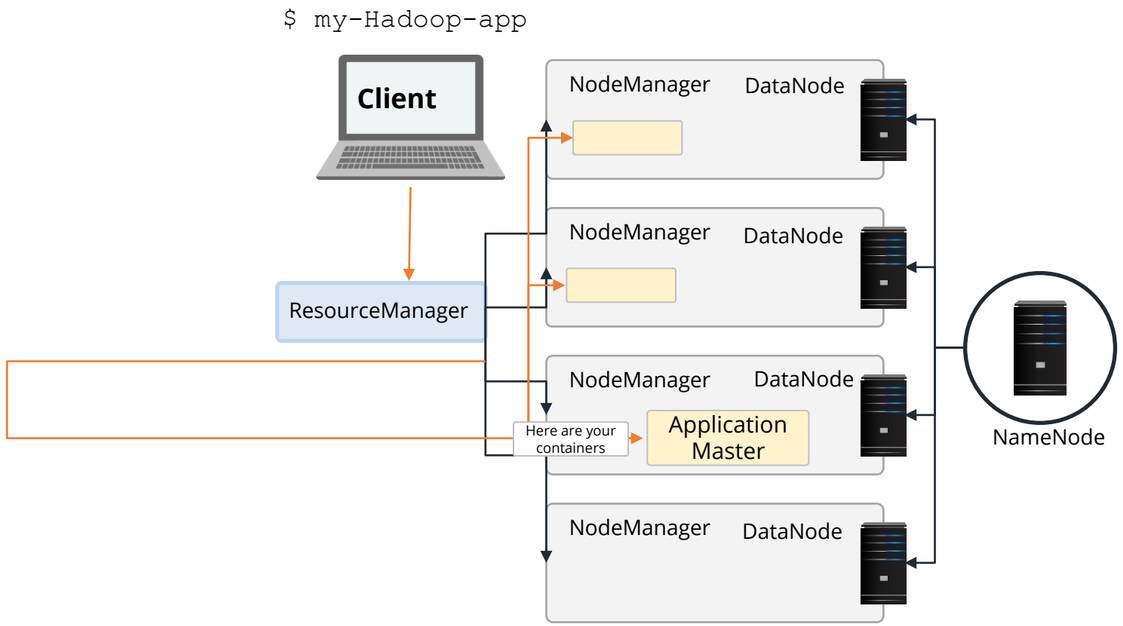
Application Workflow

- Hostname specifies the preferred location. Priority is only given in this application and not to other applicants.
- The ResourceManager assigns a Container to the ApplicationMaster by giving it a Container ID and a hostname that meets the ApplicationMaster’s requirements.

Application Workflow

Step 3: The ApplicationMaster contacts the NodeManager.

Application Manager informs NodeManager to start the Containers.



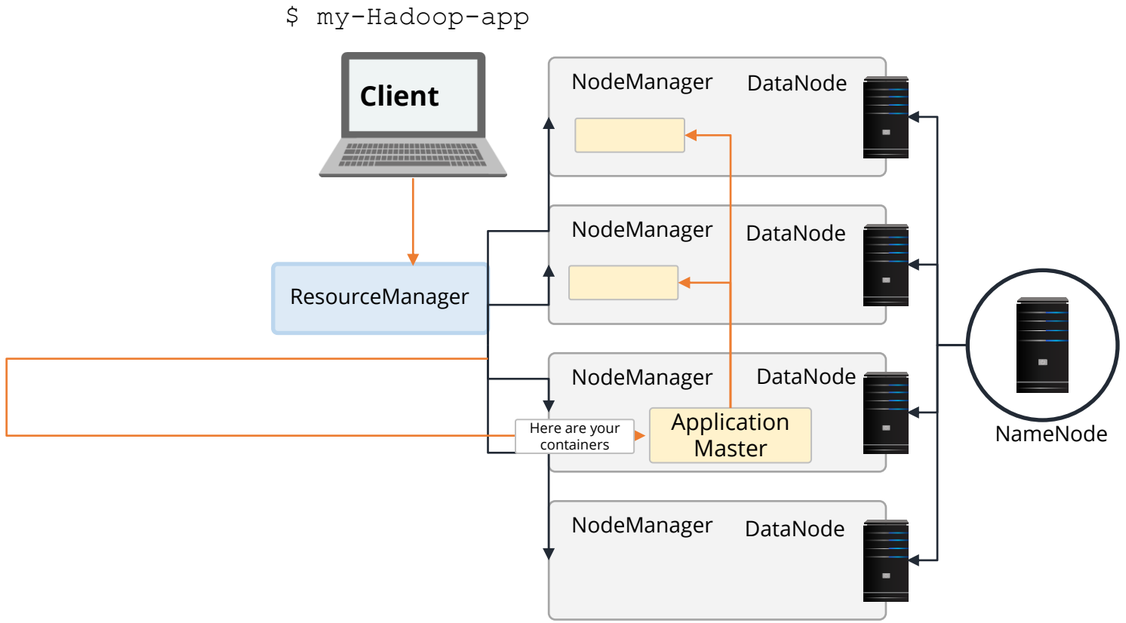
Application Workflow

- Following the allocation of a Container, the ApplicationMaster requests that the NodeManager is in charge of the host where the Container was assigned and uses these resources to run an application-specific task.
- Any process developed in any framework, such as a MapReduce task, can be used for this purpose.

Application Workflow

Step 4: The NodeManager launches the container.

The NodeManager does not keep track of tasks; instead, it keeps track of resource utilization in the Containers.



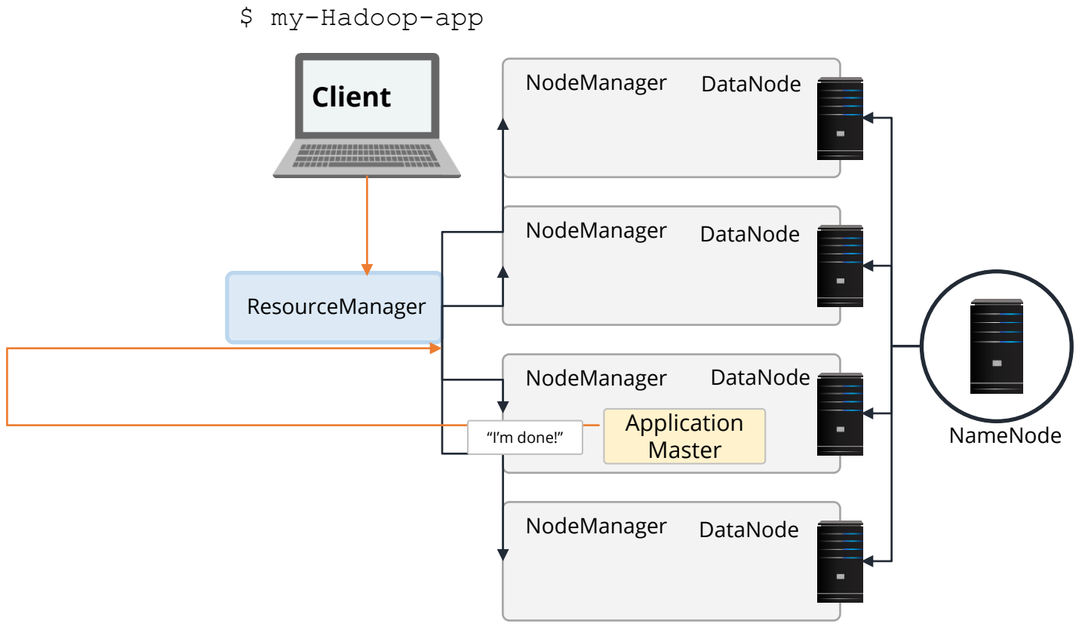
Application Workflow

- If a Container consumes more memory than it was assigned, it will be removed.
- The ApplicationMaster negotiates Containers during its life to launch all the tasks required to accomplish its application.
- It also keeps track of applications and the task's progress, restarts unsuccessful tasks in freshly requested Containers, and updates the client who applied its progress.

Application Workflow

Step 5: The Container executes the ApplicationMaster.

The ApplicationMaster closes itself and releases its Container after the application is completed.



Application Workflow

- The ResourceManager does not monitor the tasks within an application, though it verifies the ApplicationMaster's health.
- If the ApplicationMaster fails, the ResourceManager can restart it in a new Container.
- As a result, the ResourceManager is responsible for the ApplicationMaster, while the ApplicationMaster is responsible for the tasks.

Tools for YARN Developers

Tools for YARN Developers

Hadoop has three tools for YARN developers:

YARN Web UI

Hue Job Browser

YARN Command Line

YARN Web UI

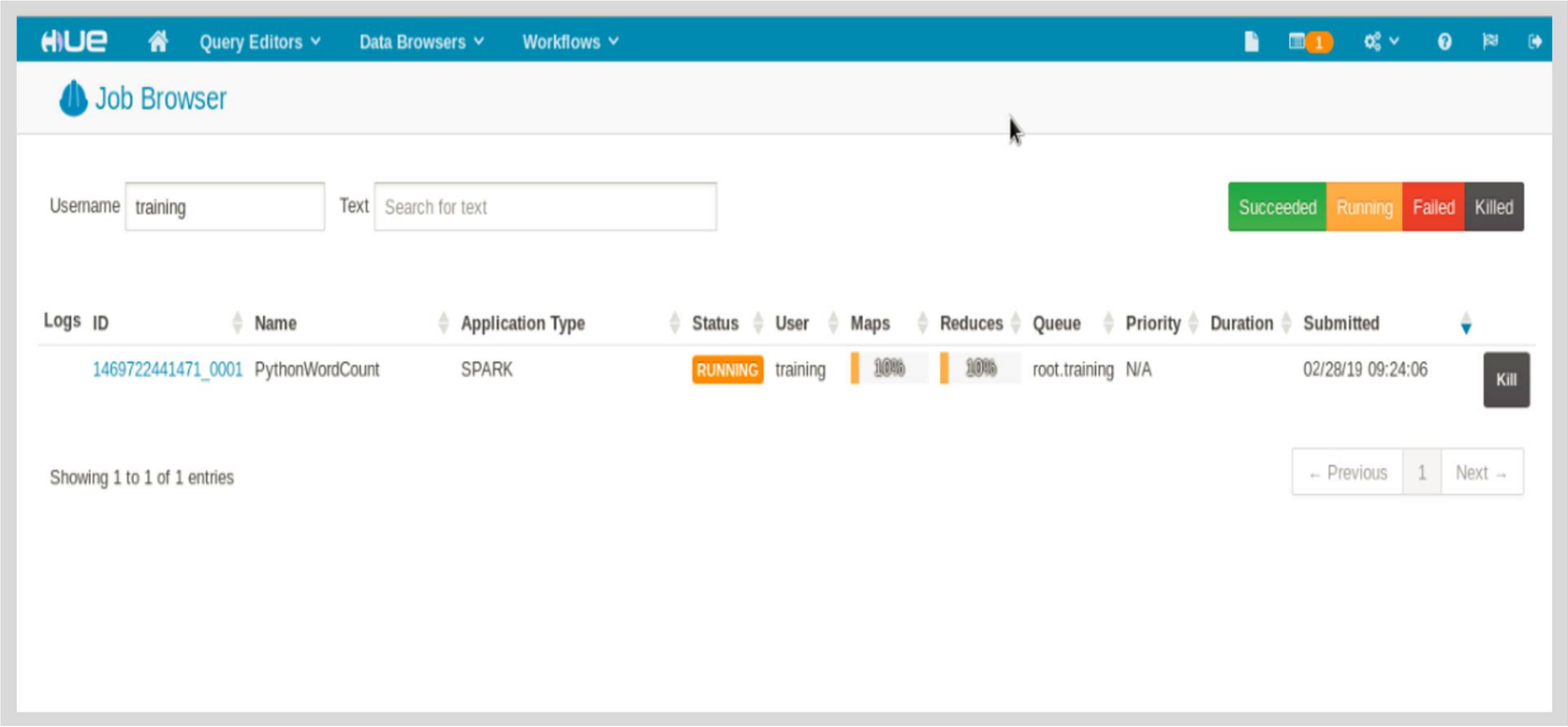
By default, the YARN Web UI runs on port 8088.

It offers a better perspective than Hue. The online UI does not allow to control or configure YARN.

The YARN web user interface maintains a record of clusters, queues, applications, services, and flow operations.

HUE Job Browser

Use the Hue task browser to check on the status of a job and see the logs



HUE Job Browser

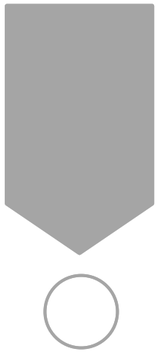
- One can use the job browser application to look at the Hadoop MapReduce jobs that are currently executing on the Hadoop cluster.
- The job and tasks are presented in tiers in the job browser.
- The job and tasks can be linked to a list of the job duties from the top layer.

HUE Job Browser

- The properties of each try, such as status, start, end times, and output size, can then be viewed for a task's attempts.
- Also look at the logs of each attempt to troubleshoot failed jobs.

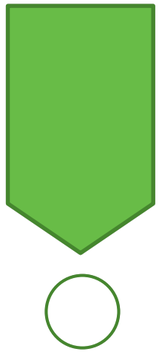
YARN Command Line

- Yarn includes several command-line tools to assist with various parts of the Yarn package, such as installation, administration, and publishing.
- Most YARN commands are intended for administrators rather than developers.
- The developer will find the following few commands useful:



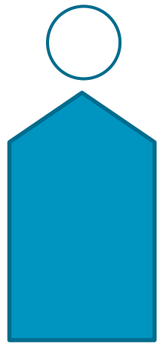
- yarn –help

lists all commands of yarn



- yarn logs -applicationId <app-id>

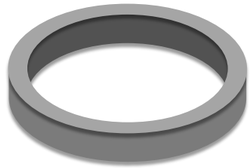
views logs of specified application ID



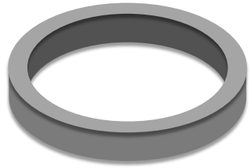
- yarn –version

prints the version

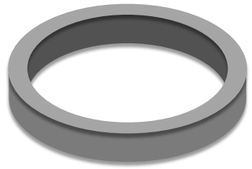
YARN Command Line



yarn add: This command adds a package to the current package.

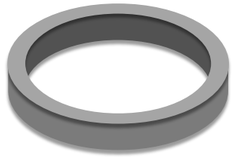


yarn init: This command starts the package development process.

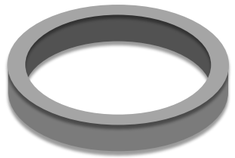


yarn install: This command installs all of the package's dependencies in a file in JSON.

YARN Command Line



yarn publish: This command sends a package to the package manager.



yarn remove: This command removes an unused package from the current package.

Key Takeaways

- YARN is a Hadoop ResourceManager that was established by dividing the MapReduce processing engine and management function.
- YARN was implemented due to the limitations of Hadoop 1.0.
- YARN architecture consists of client, ResourceManager, NodeManager, ApplicationMaster, and Container.
- Hadoop has 3 tools for YARN developers, which are YARN Web UI, Hue job browser, and YARN command line.

Knowledge Check

Knowledge Check 1

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

A. FIFO scheduler

B. Capacity scheduler

C. Fair scheduler

D. All the above

Knowledge Check 1

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

A. FIFO scheduler

B. Capacity scheduler

C. Fair scheduler

D. All the above

The correct answer is **D**

YARN provides three scheduling options: FIFO Scheduler, Capacity Scheduler, and a Fair Scheduler for scheduling resources to user applications.

Knowledge Check 2

_____ negotiates resources from the ResourceManager.

A. NodeManager

B. ResourceManager

C. ApplicationMaster

D. None of the above

Knowledge Check 2

_____ negotiates resources from the ResourceManager.

A. NodeManager

B. ResourceManager

C. ApplicationMaster

D. None of the above

The correct answer is **C**

Each ApplicationMaster is responsible for negotiating resources with the ResourceManager.

Lesson-End Project

Problem Statement:

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

They began using YARN as a ResourceManager for Spark and Hadoop MR applications. You must include the following information for any job submitted to YARN using the YARN console or the YARN Cluster UI.

Objective:

The objective is to identify jobs submitted using the yarn application.

LESSON-END PROJECT

Lesson-End Project

Tasks to Perform:

1. Open the Readme file on the Web desktop
2. Start the JobHistory daemon by using the command from the Readme file
3. Check the list of applications in YARN
4. Open the JobHistory server and check the list of applications running on it

LESSON-END PROJECT

Thank You