YARN

YARN

- YARN stands for "***Yet Another Resource Negotiator***".
- It was introduced in Hadoop 2.0 to remove the bottleneck on Job Tracker which was present in Hadoop 1.0.
- YARN was described as a "*Redesigned Resource Manager*" at the time of its launching, but it has now evolved to be known as large-scale distributed operating system used for Big Data processing.

YARN architecture

- YARN architecture basically separates resource management layer from the processing layer. In Hadoop 1.0 version, the responsibility of Job tracker is split between the resource manager and application manager.

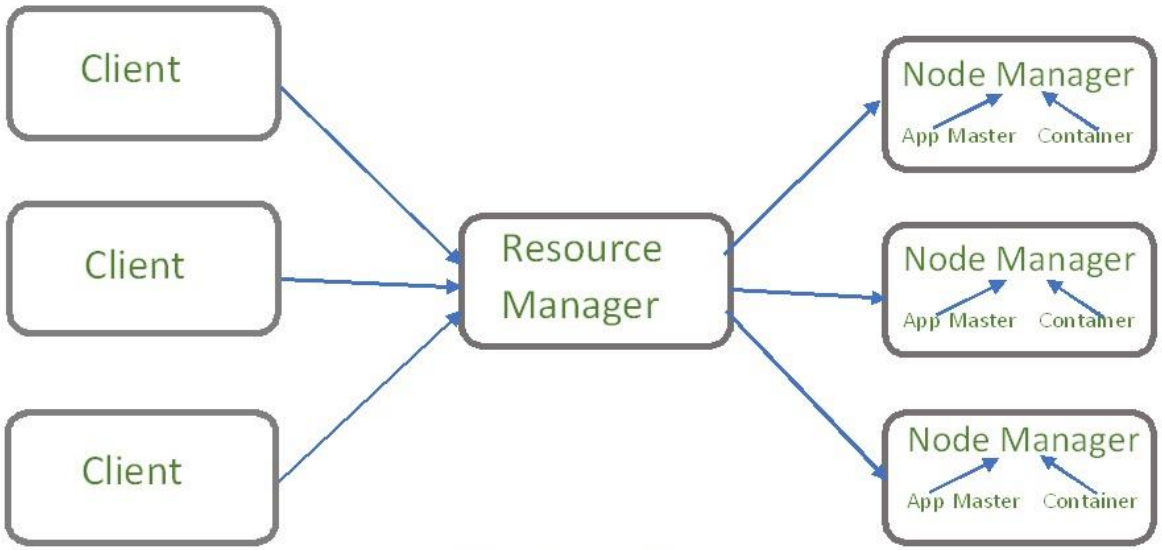
Hadoop 2.0

- YARN also allows different data processing engines like graph processing, interactive processing, stream processing as well as batch processing to run and process data stored in HDFS (Hadoop Distributed File System) thus making the system much more efficient.
- Through its various components, it can dynamically allocate various resources and schedule the application processing.
- For large volume data processing, it is quite necessary to manage the available resources properly so that every application can leverage them.

YARN Features: YARN gained popularity because of the following features-

- Scalability: The scheduler in Resource manager of YARN architecture allows Hadoop to extend and manage thousands of nodes and clusters.
- Compatibility: YARN supports the existing map-reduce applications without disruptions thus making it compatible with Hadoop 1.0 as well.
- Cluster Utilization:Since YARN supports Dynamic utilization of cluster in Hadoop, which enables optimized Cluster Utilization.
- Multi-tenancy: It allows multiple engine access thus giving organizations a benefit of multi-tenancy.

Hadoop YARN Architecture



Hadoop Yarn architecture

The main components of YARN architecture include:

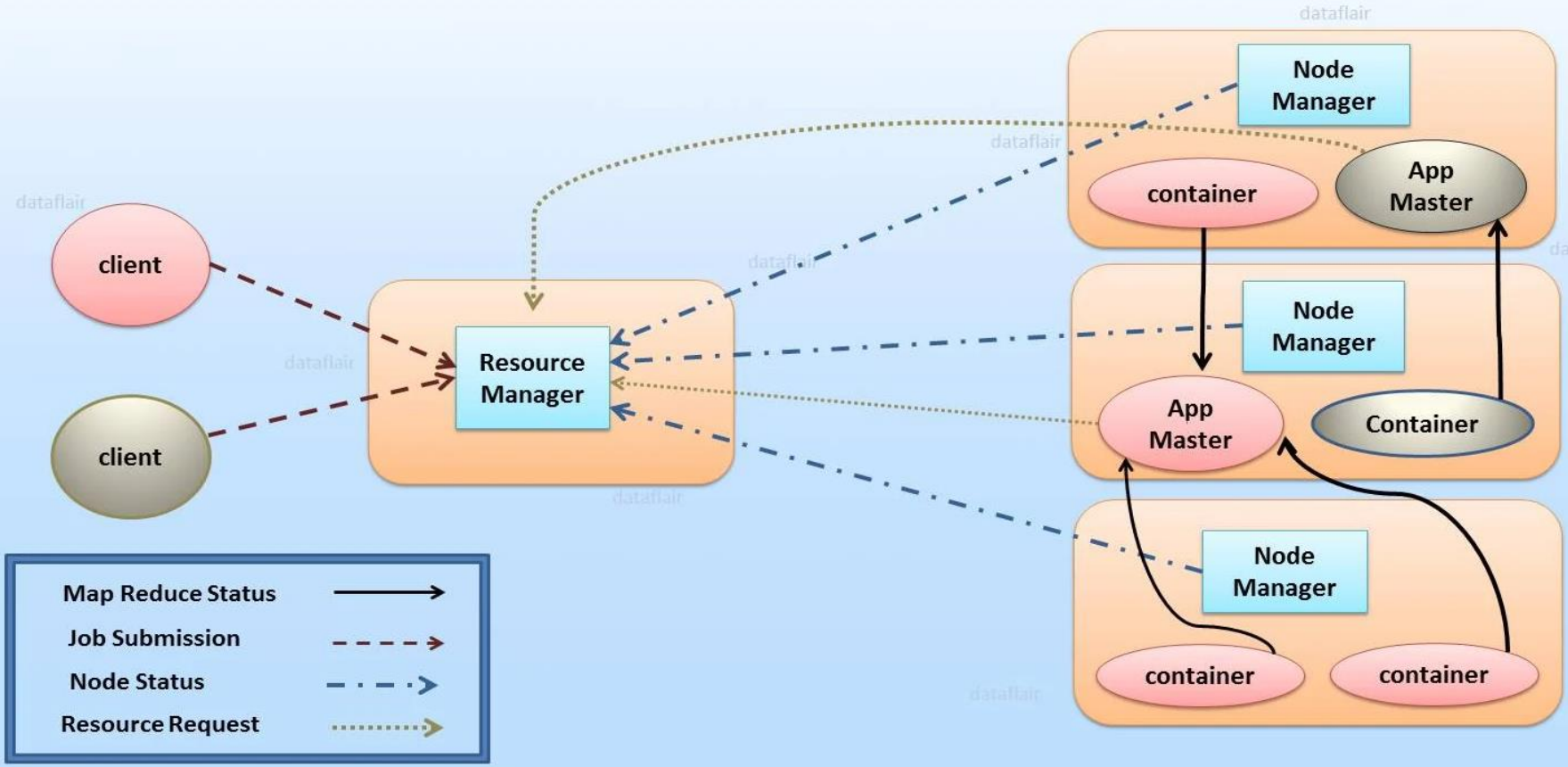
- **Client:** It submits map-reduce jobs.
- **Resource Manager:** It is the master daemon of YARN and is responsible for resource assignment and management among all the applications. Whenever it receives a processing request, it forwards it to the corresponding node manager and allocates resources for the completion of the request accordingly. It has two major components:
  - **Scheduler:** It performs scheduling based on the allocated application and available resources. It is a pure scheduler, means it does not perform other tasks such as monitoring or tracking and does not guarantee a restart if a task fails. The YARN scheduler supports plugins such as Capacity Scheduler and Fair Scheduler to partition the cluster resources.
  - **Application manager:** It is responsible for accepting the application and negotiating the first container from the resource manager. It also restarts the Application Master container if a task fails.

- Node Manager:
  - It take care of individual node on Hadoop cluster and manages application and workflow and that particular node.
  - Its primary job is to keep-up with the Resource Manager.
  - It registers with the Resource Manager and sends heartbeats with the health status of the node.
  - It monitors resource usage, performs log management and also kills a container based on directions from the resource manager.
  - It is also responsible for creating the container process and start it on the request of Application master.

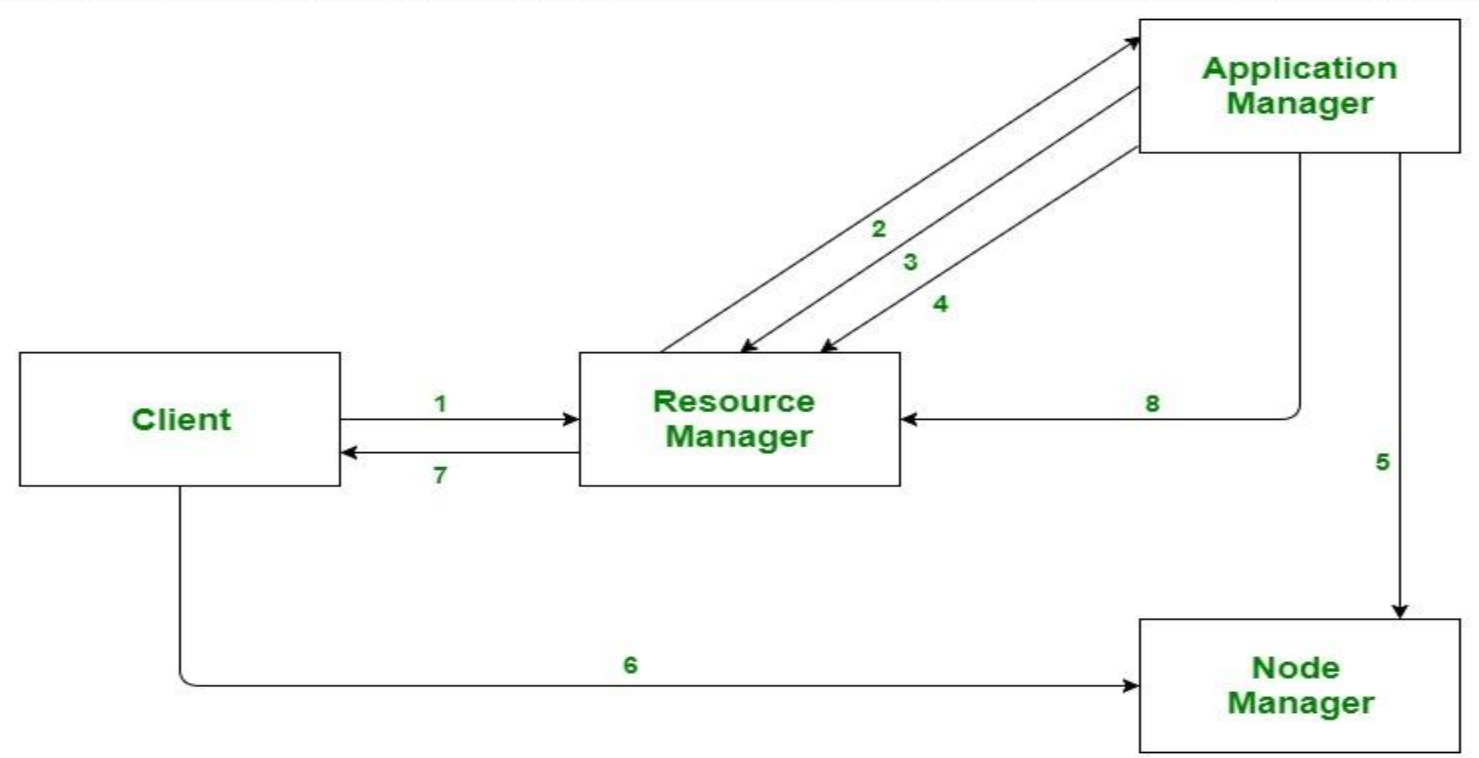
- Application Master:
  - An application is a single job submitted to a framework.
  - The application master is responsible for negotiating resources with the resource manager, tracking the status and monitoring progress of a single application.
  - The application master requests the container from the node manager by sending a Container Launch Context(CLC) which includes everything an application needs to run.
  - Once the application is started, it sends the health report to the resource manager from time-to-time.

- Container:
  - It is a collection of physical resources such as RAM, CPU cores and disk on a single node.
  - The containers are invoked by Container Launch Context(CLC) which is a record that contains information such as environment variables, security tokens, dependencies etc.

Apache Hadoop YARN-Architecture



Application workflow in Hadoop YARN:



Steps

1. Client submits an application
2. The Resource Manager allocates a container to start the Application Manager
3. The Application Manager registers itself with the Resource Manager
4. The Application Manager negotiates containers from the Resource Manager
5. The Application Manager notifies the Node Manager to launch containers
6. Application code is executed in the container
7. Client contacts Resource Manager/Application Manager to monitor application's status
8. Once the processing is complete, the Application Manager un-registers with the Resource Manager