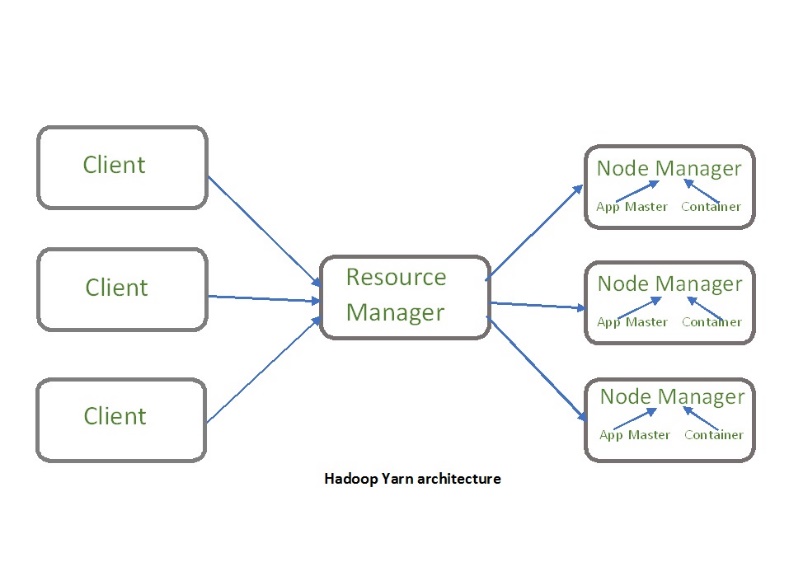
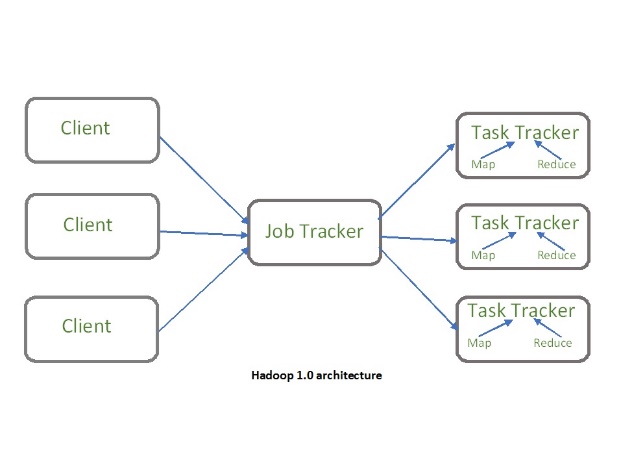
**Hadoop YARN Architecture :**

* YARN (Yet Another Resource Negotiator) architecture is a core component of Apache Hadoop's ecosystem, introduced in Hadoop 2.x.
* It enhances the MapReduce framework by decoupling resource management and job scheduling. This allows for more efficient cluster utilization and supports multiple data processing frameworks.
* Initially described as a "Redesigned Resource Manager," it has since evolved into a large-scale distributed operating system for Big Data processing.

**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:

1. **Scheduler**: Allocates resources to applications based on availability but does not monitor or restart tasks. Supports plugins like Capacity Scheduler and Fair Scheduler for resource partitioning.
2. **Application Manager**: Handles application submission, negotiates the first container from the Resource Manager, and restarts the Application Master container if a failure occurs.

* **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.

Monitoring resource usage, handling log management, and terminating containers as directed by the Resource Manager.

Creating and starting container processes upon requests from the Application Master.

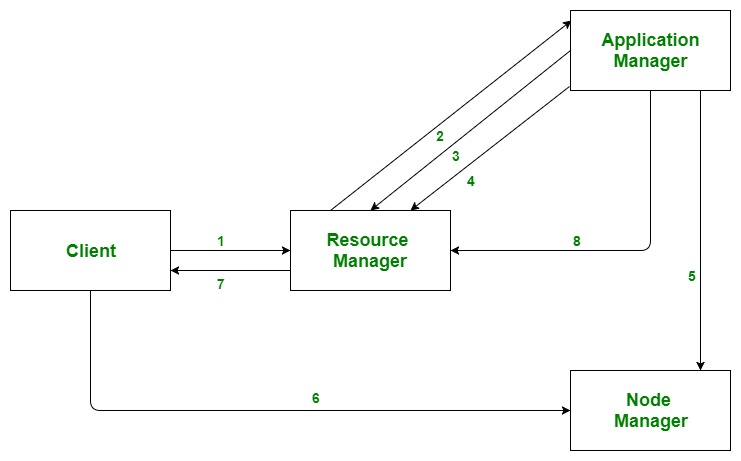
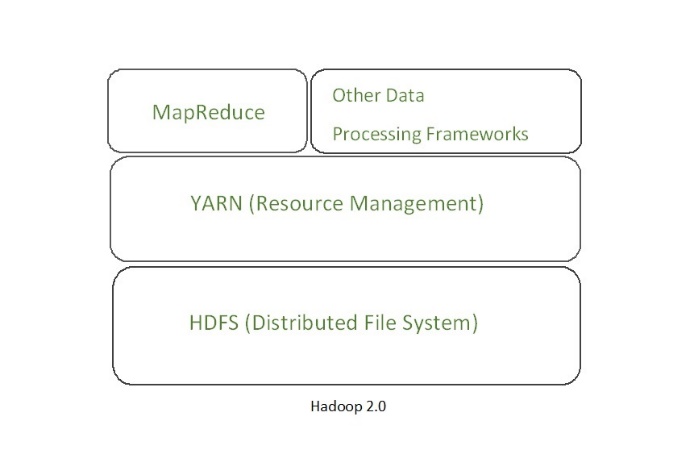
* **Application Master:** It is responsible for negotiating resources with the resource manager, tracking the status and monitoring progress of a single application.

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 unit of allocated physical resources, including RAM, CPU cores, and disk space, on a single node. Containers are launched using a **Container Launch Context (CLC)**, which contains essential information such as environment variables, security tokens, and application dependencies.

**Application workflow in Hadoop YARN:**

**YARN Workflow Summary**

1. **Application Submission**: The client submits an application to the Resource Manager.
2. **Application Manager Initialization**: The Resource Manager allocates a container to start the Application Manager.
3. **Registration**: The Application Manager registers itself with the Resource Manager.
4. **Resource Negotiation**: The Application Manager negotiates containers from the Resource Manager.
5. **Container Launch**: The Application Manager instructs the Node Manager to launch containers.
6. **Task Execution**: The application code executes within the containers.
7. **Status Monitoring**: The client contacts the Resource Manager or Application Manager to monitor the application’s status.
8. **Completion**: Upon task completion, the Application Manager unregisters itself from the Resource Manager.

**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.

**Advantages of YARN :**

* **Scalability**: Efficiently manages resources in large clusters.
* **Flexibility**: Supports various processing frameworks (e.g., MapReduce, Spark, Tez).
* **Improved Resource Utilization**: Better resource allocation compared to the original Hadoop MapReduce model.
* **Security:** It ensures that the data stored and processed on the Hadoop cluster is secure.

**Disadvantages of YARN :**

* **Complexity**: Requires careful configuration and management.
* **Overhead**: Introduces additional layers of management, which can impact performance
* **Latency:**YARN introduces additional latency in the Hadoop ecosystem.
* **Limited Support:** YARN has limited support for non-Java programming languages.