


YARN Architectural Components

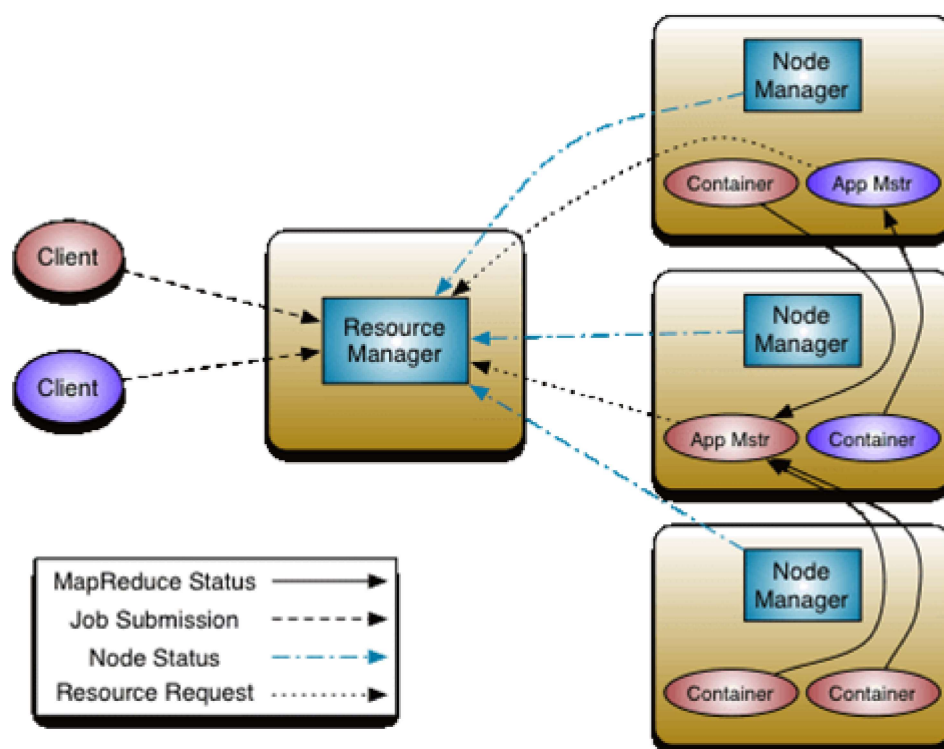
YARN is a distributed resource management and scheduling framework written in Java, running on the same nodes in the cluster that HDFS runs. It is composed of the following daemons:

- **Resource Manager (one or two per cluster) that provides**
 - Global resource scheduler
 - Hierarchical queues
- **Node Manager (running next to the DataNode)**
 - Encapsulates RAM and CPU resources available on a worker node into units called YARN containers
 - Manages the lifecycle of YARN containers
 - Container resource monitoring
- **Application Master (created on-demand)**
 - Manages application scheduling and task execution
 - Typically, specific to a higher-level framework (e.g. MapReduce Application Master).

Just like with HDFS, what determines whether you are running just one or two Resource Managers per cluster is your tolerance for YARN outages. With just one Resource Manager - if it goes down, your cluster can no longer accept new jobs and provide some of the YARN capabilities for managing jobs that are running (even though the existing jobs running via Node Managers will be fine). With two Resource Managers running in an Active/Standby configuration, the Standby can take over in cases where the Active one fails or needs 

be brought down for maintenance. Even though a single Resource Manager going down presents less of a problem than a single Name Node going down, the High Availability (HA) configuration for YARN is pretty popular, and we will cover it in more details later in this chapter.

The interaction of all these parts of YARN will look something like this, and we will spend the rest of this chapter explaining the internals of its architecture:



YARN Architecture (by The Apache Software Foundation, retrieved from hadoop.apache.org)