Anatomy of Map Reduce Job Run

The anatomy of a MapReduce job run can be divided into the following steps:

Client Submits The Job: The client submits the MapReduce job to the YARN **Resource Manager**.

- ✓ The resource manager allocates the necessary resources (containers) to the job and launches the MapReduce **Application Master**.
- ✓ All the Containers are managed by **Node Manager**

Application Master Coordinates The Job: The application master splits the input data into splits and assigns them to map tasks.

- ✓ It also monitors the progress of the map and reduce tasks and restarts any failed tasks.

Map Tasks Process The Input Data: The map tasks process their assigned input splits and **Generate Intermediate Key-value Pairs**.

Shuffle Phase: The shuffle phase **Sorts The Intermediate Key-value Pairs** from the map tasks and groups them by key.

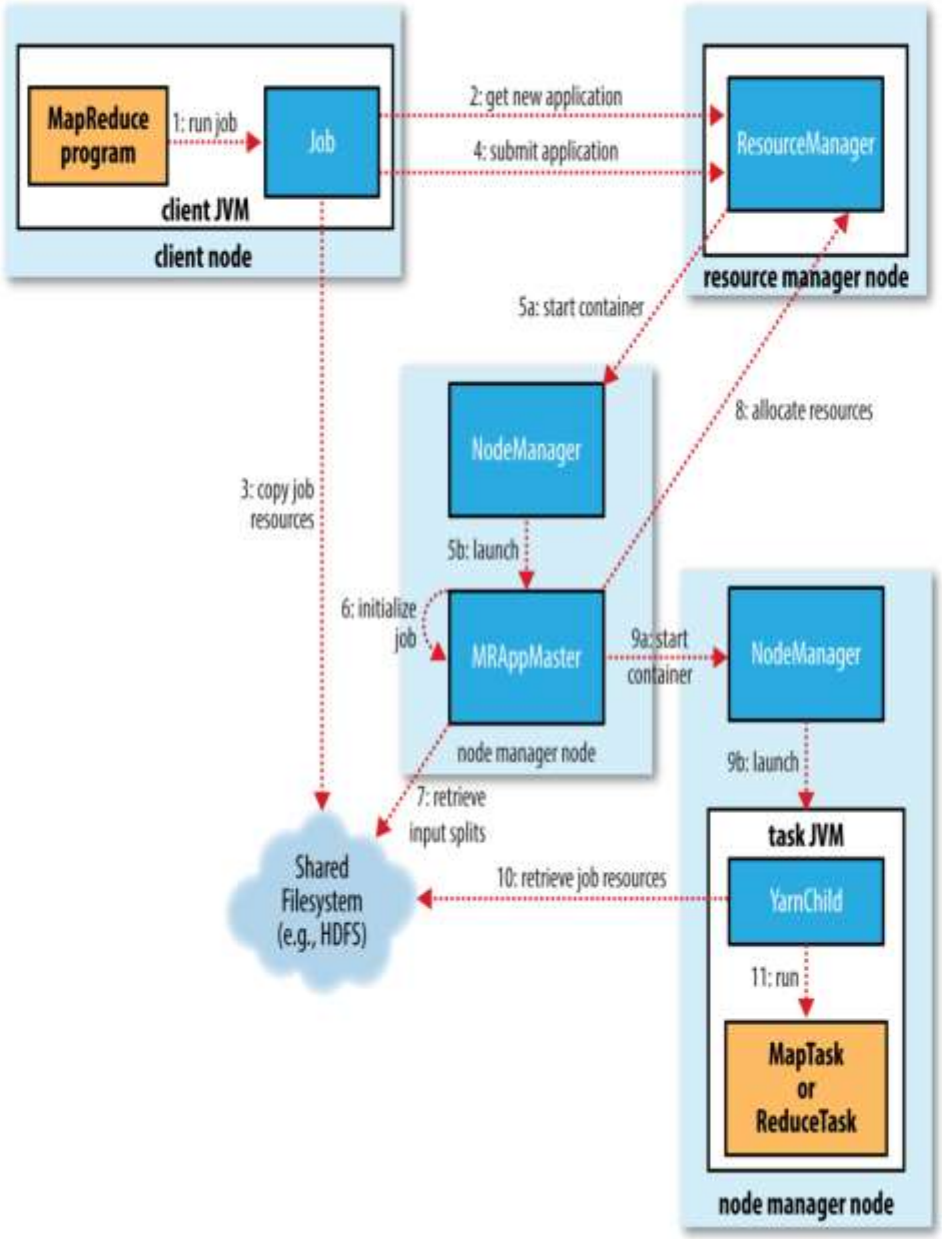
Reduce Tasks Process The Intermediate Data: The reduce tasks process the intermediate key-value pairs from the shuffle phase and **Generate The Final Output**.

Application Master Completes The Job: Once all of the reduce tasks have completed, the application master merges the output from the reduce tasks and writes it to the distributed file-system.

- ✓ **Client JVM**: The client Java Virtual Machine is the **Java process that submits the MapReduce job**.
- ✓ **Client node**: The client node is the machine where the client **JVM Is Running**.

- ✓ **ResourceManager**: The ResourceManager is responsible for **Allocating Resources** (containers) to MapReduce jobs.
- ✓ **Resource Manager Node**: The resource manager node is the machine where the **Resource Manager Is Running**.
- ✓ **NodeManager**: The NodeManager is responsible for **Managing Containers On A Machine.**
- ✓ **Node manager node**: The node manager node is the machine where the **Node Manager Is Running**.
- ✓ **MRAppMaster**: The MRAppMaster is a Java process that is responsible for coordinating the **Execution Of The Map Reduce Job**.

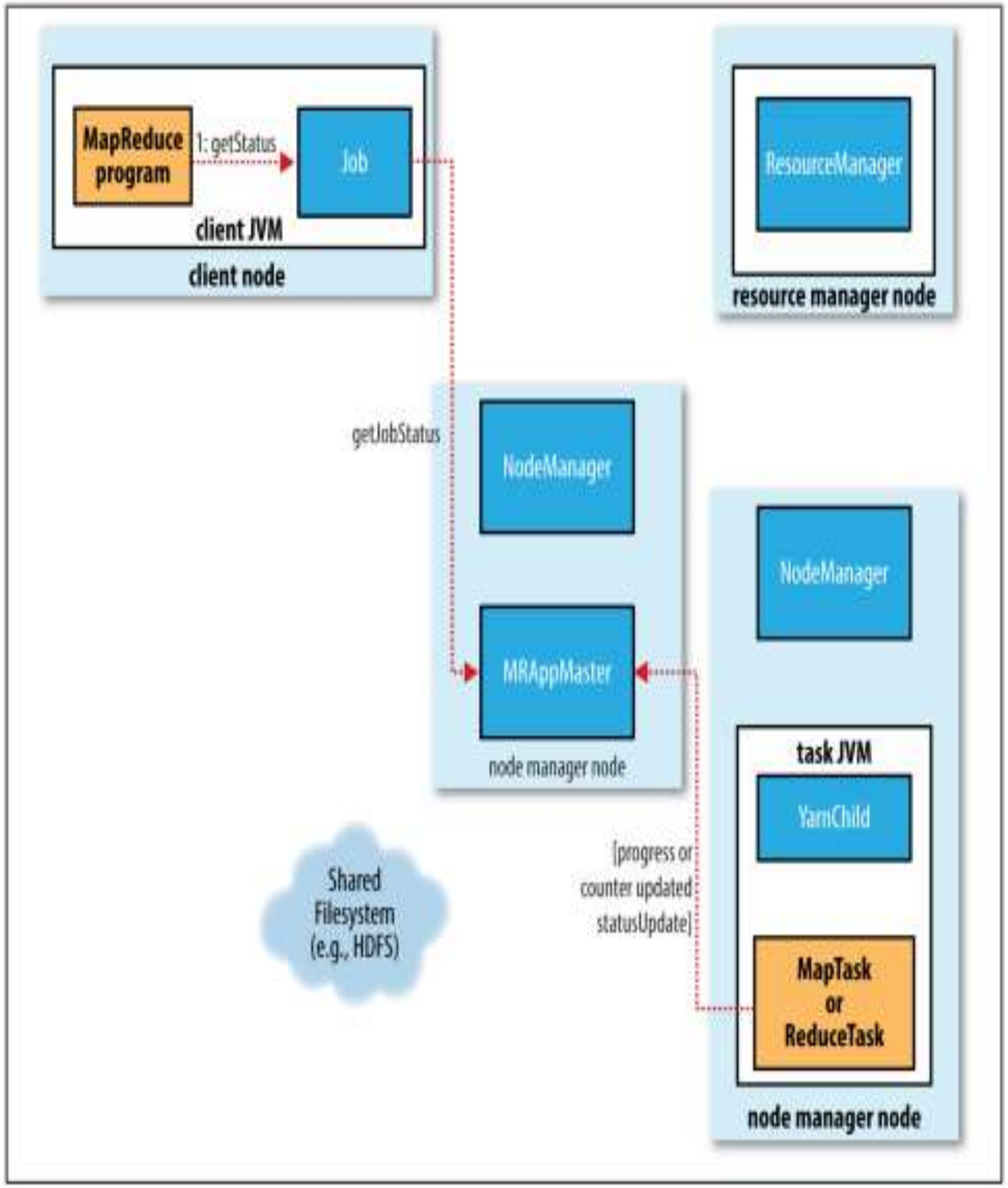
- ✓ **Task JVM**: The task JVM is a Java process that is responsible for **Executing A Mapper Or Reducer Task**.
- ✓ **Yarn Child**: The Yarn Child is a Java process that is responsible for **Running The Task JVM**.
- ✓ **Shared Filesystem**: The shared filesystem is a distributed filesystem, such as **HDFS**, that is used to store the input and output data for the MapReduce job



How Hadoop Runs a Map Reduce Job

1. Job submission
2. Job Initialization
3. Task Assignment
4. Task execution

Progress & Status Update



The status updates are propagated through the MapReduce system as follows:

- ✓ The task JVM sends status updates to the Yarn Child.
- ✓ The Yarn Child sends status updates to the MRAppMaster.
- ✓ The MRAppMaster sends status updates to the ResourceManager.
- ✓ The ResourceManager sends status updates to the client JVM.

Job Completion

- Application Master receives a notification for a job is completed.
- It changes job status to successful.
- Application master and the task container also clean up their working state.