Week 4-1

Hadoop (HDFS & YARN)

Big Data

Prof. Hwanjo Yu
POSTECH

Preliminary (Environment Setting on Ubuntu)

- Install Java
- Download Hadoop 2.8.1 from Hadoop homepage
- Move hadoop-2.8.1.tar.gz to /usr/local/
  - mv hadoop-2.8.1.tar.gz /usr/local/
- unzip the Hadoop file
  - tar xvzf hadoop-2.8.1.tar.gz
- Change ownership of Hadoop directory
  - chown –R “hadoopUser” Hadoop-2.8.1

Preliminary (Environment Setting on Ubuntu)

- vim /etc/profile
  - export JAVA_HOME=/usr/lib/jvm/java-8-oracle
  - export CLASSPATH=$JAVA_HOME/lib:$JAVA_HOME/jre/lib/ext:$JAVA_HOME/lib/tools.jar
  - export HADOOP_HOME=/usr/local/hadoop-2.8.1
  - export HADOOP_CLASSPATH=$CLASSPATH/tools.jar
  - export HADOOP_PREFIX=$HADOOP_HOME
  - export PATH=$PATH:$JAVA_HOME/bin:$HADOOP_HOME/bin
- source /etc/profile

Preliminary (Environment Setting on Ubuntu)

- Setting up a Single Node Cluster with pseudo cluster mode.
  - https://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-common/SingleCluster.html#Pseudo-Distributed_Operation
- Do it yourself. (HW 1.5.)

Hadoop?

- Open-source software framework for reliable, scalable, distributed computing
- Motivated from Google's GFS, MapReduce
- Apache license (one of the top-level Apache project)

HDFS + MapReduce(Yarn)

Hadoop is…

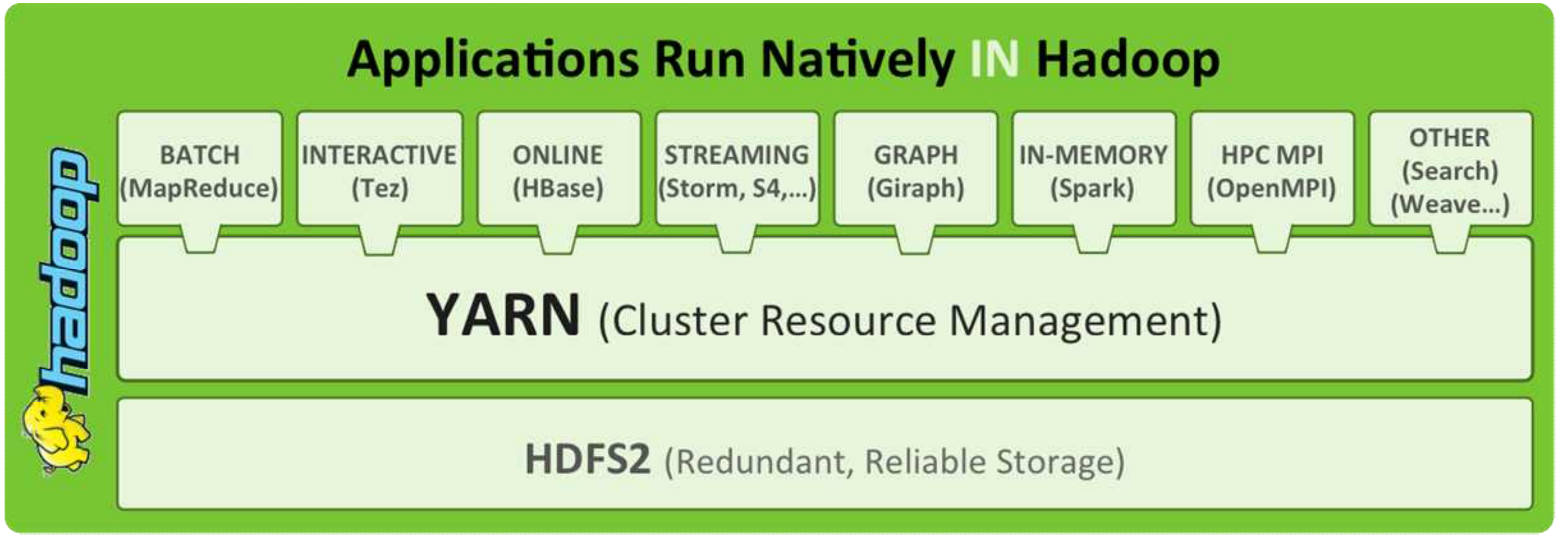
- Scale out (Many, cheap machines than few, expensive machines)
- Add, delete, append, ~~modify~~
- Transfer code, not data
- Appropriate in offline, batch processing
- Automatic Fault-tolerant scheme
- Linux environment, JAVA language

Hadoop

- Not substitutable for a database
    - Cannot modify data, doesn’t index data, integrity, constraint, …
- MapReduce is not always the best algorithm
- Not good at iteration process and interactive queries

Modules of Hadoop

- HDFS
- Hadoop YARN
- Hadoop MapReduce



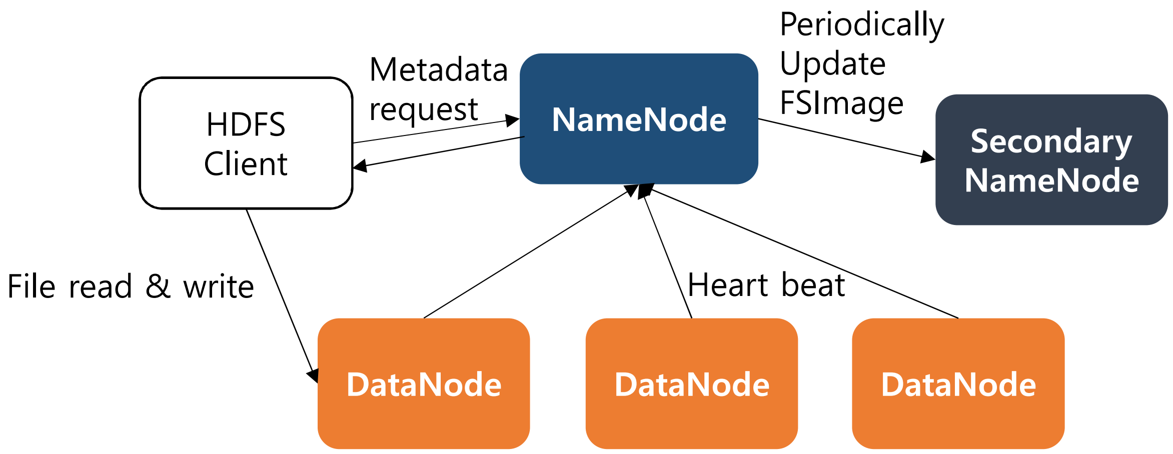
GFS? HDFS?

- GFS (The Google File System)
  - S. Ghemawat, H. Gobioff, S. Leung, The Google File System, SIGOPS 2003
  - Implemented for the rapidly growing demands of storing Google's data

- HDFS (The Hadoop Distributed File System)
  - K. Shvachko, H. Kuang, S. Radia, The Hadoop distributed file system, MSST 2010
  - Comparable to GFS

HDFS

- Distributed File System
- Can be used on the general purpose machine
- Unix-like file shell script



HDFS Client :
A program using HDFS
(e.g. Java Program, HDFS command)

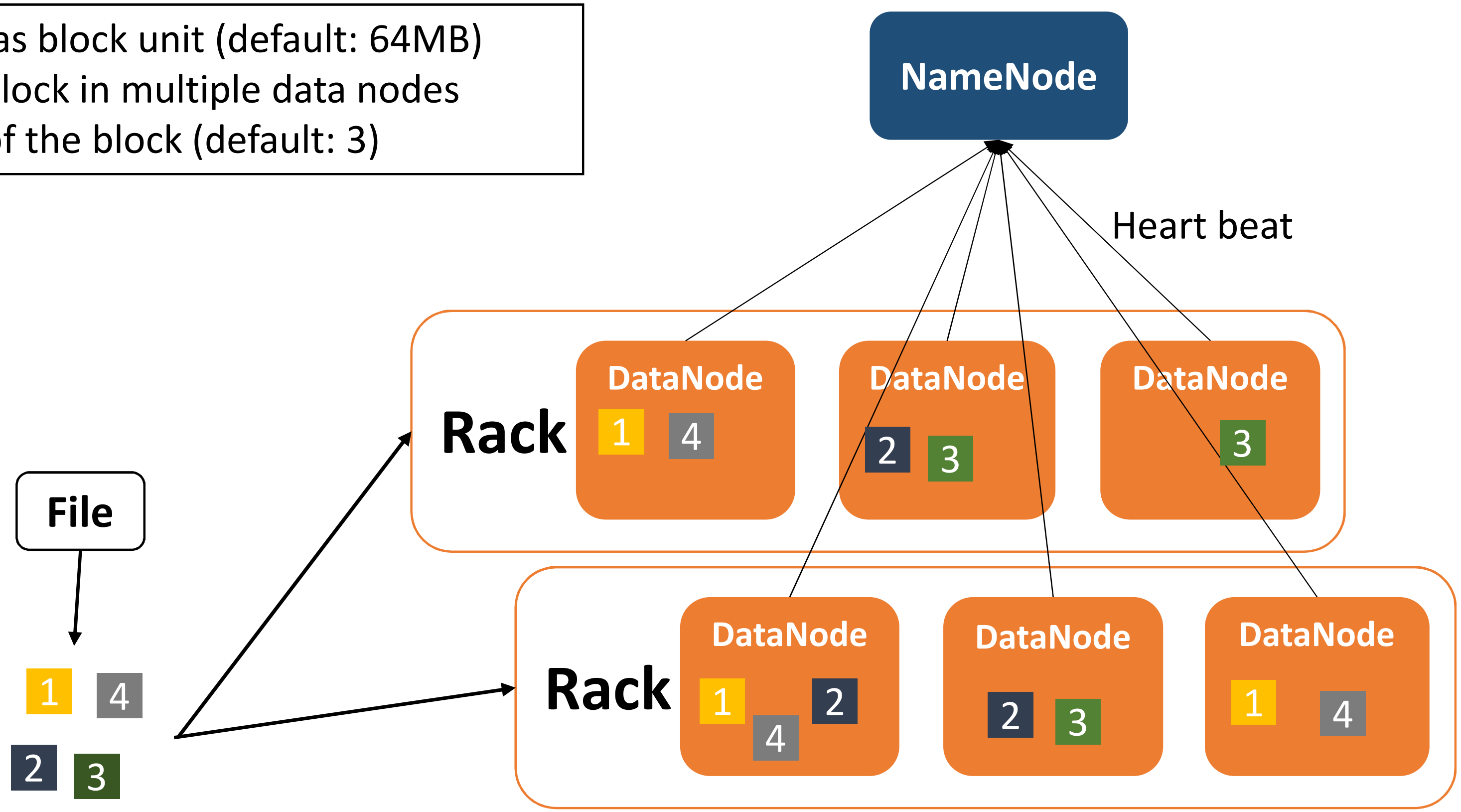
NameNode:
Manages the file system metadata

Secondary NameNode:
Support NameNode

DataNode:
Store the actual data

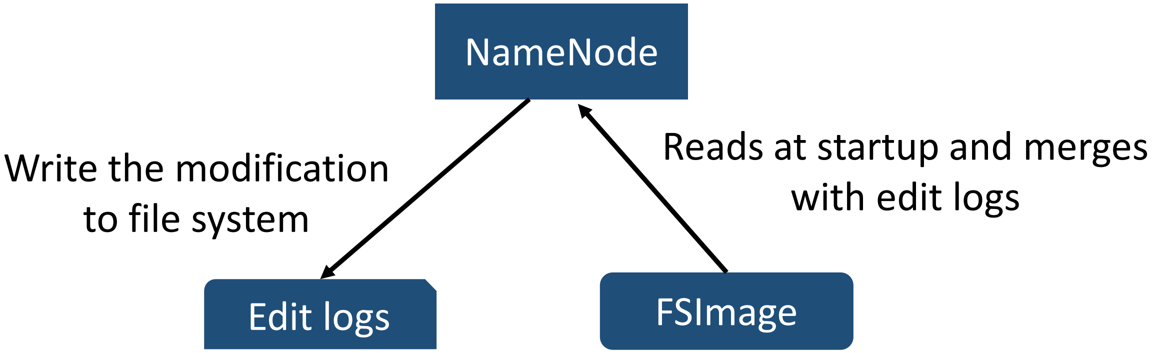
HDFS – Block based

- Storing files as block unit (default: 64MB)
- Storing the block in multiple data nodes
- Replication of the block (default: 3)



HDFS – NameNode

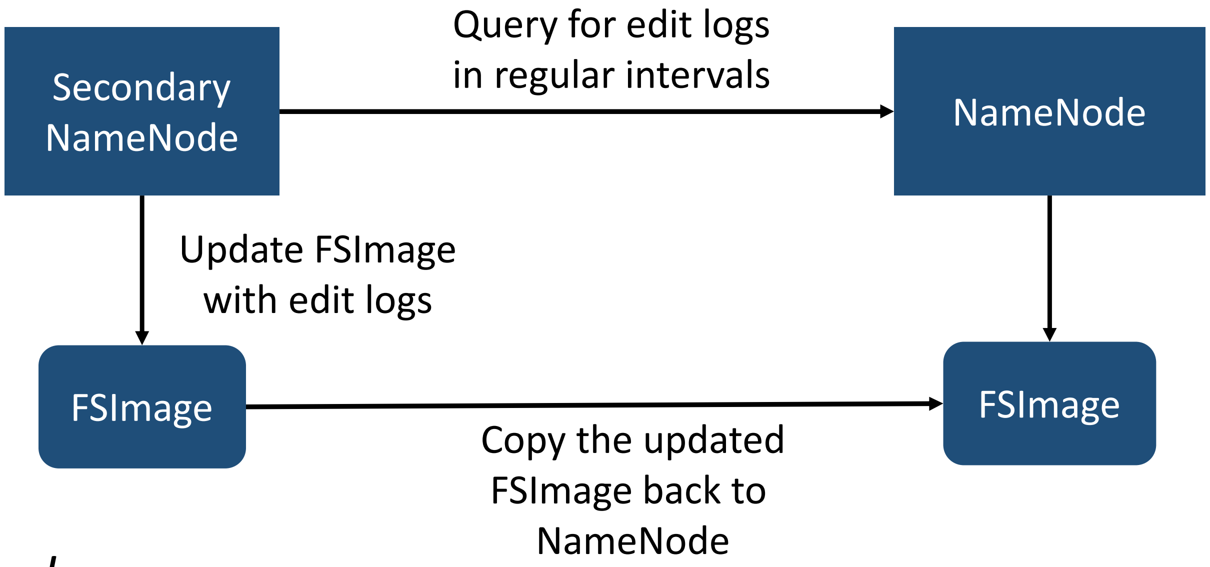
- FSImage : Snapshot of the file system when NameNode started
- Edit logs : Sequence of changes made to the file system after NameNode started


NameNode
Write the modification to file system
Reads at startup and merges with edit logs
Edit logs
FSImage

Problems

- Edit logs become very large
- NameNode restart take long time because lot of changes has to be merged
- In the case of crash, we will lost huge amount of metadata since FSImage is very old

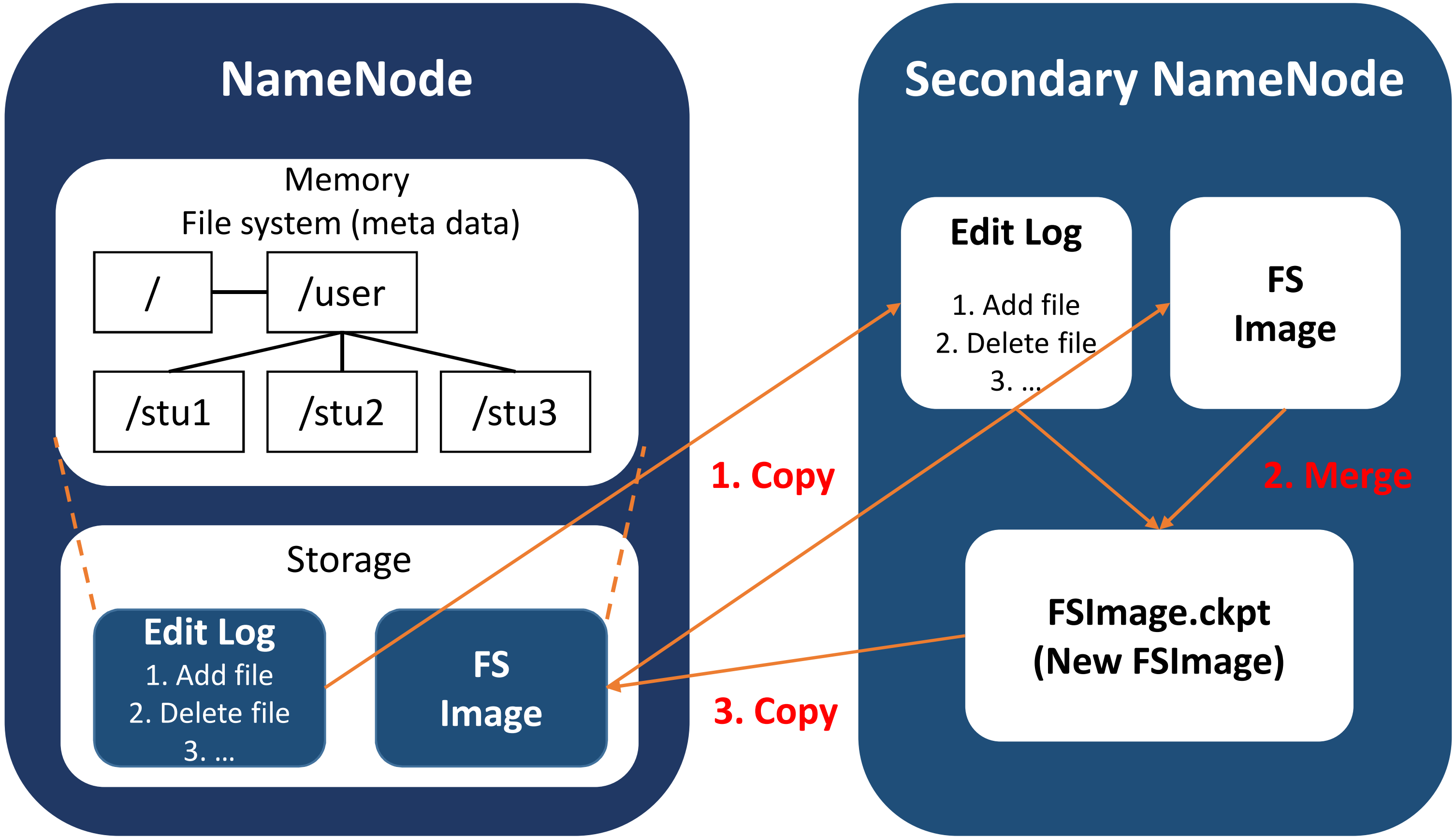
HDFS – Secondary NameNode


Secondary NameNode
Query for edit logs in regular intervals
NameNode
Update FSImage with edit logs
FSImage
FSImage
Copy the updated FSImage back to NameNode

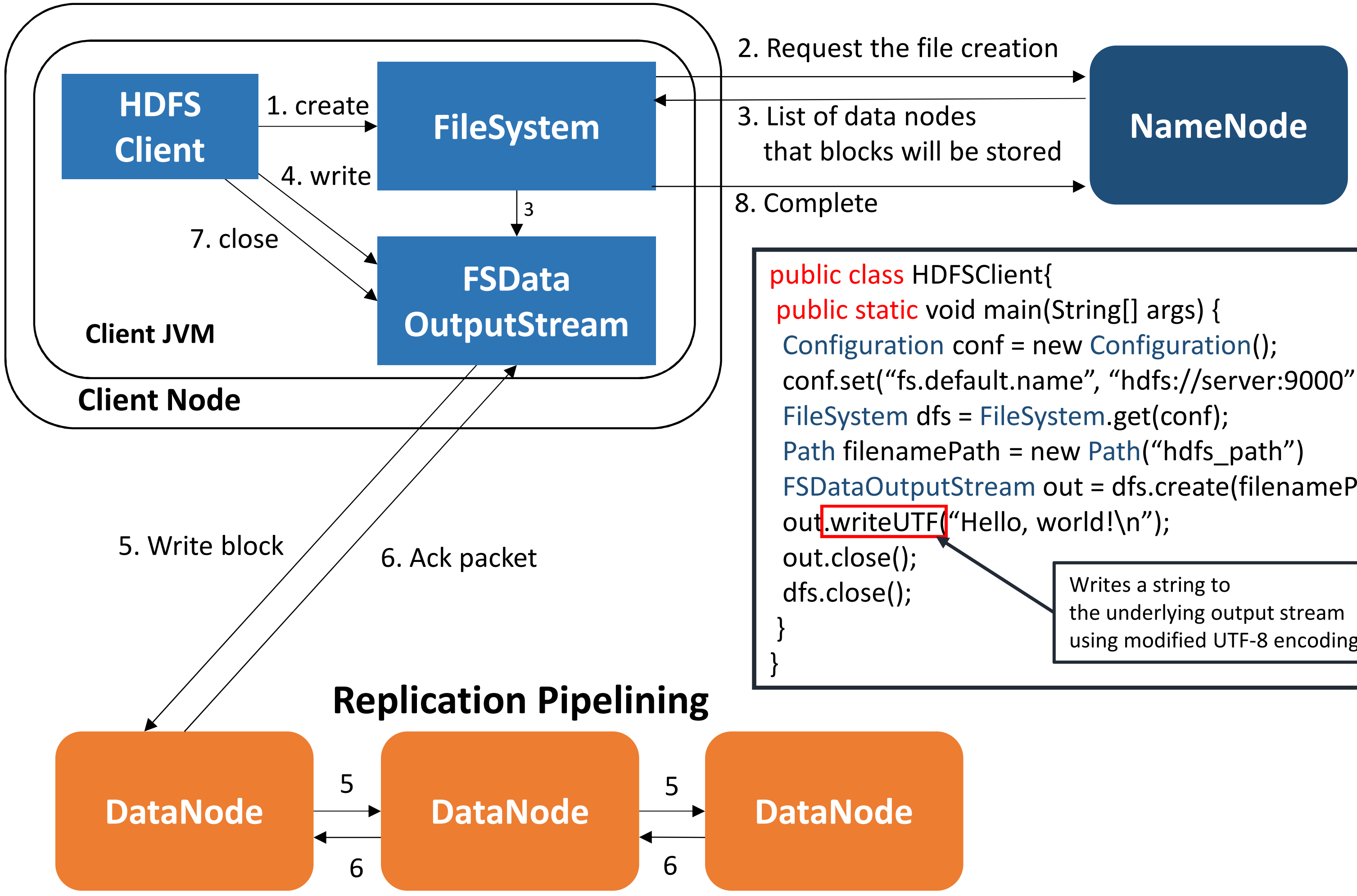
Problems are solved

- It gets the edit logs from the NameNode in regular intervals and applies to FSImage
- Once it has new FSImage, it copies back to NameNode
- NameNode will use this FSImage for the next restart, which will reduce the startup time.

HDFS – NameNode, Secondary NameNode



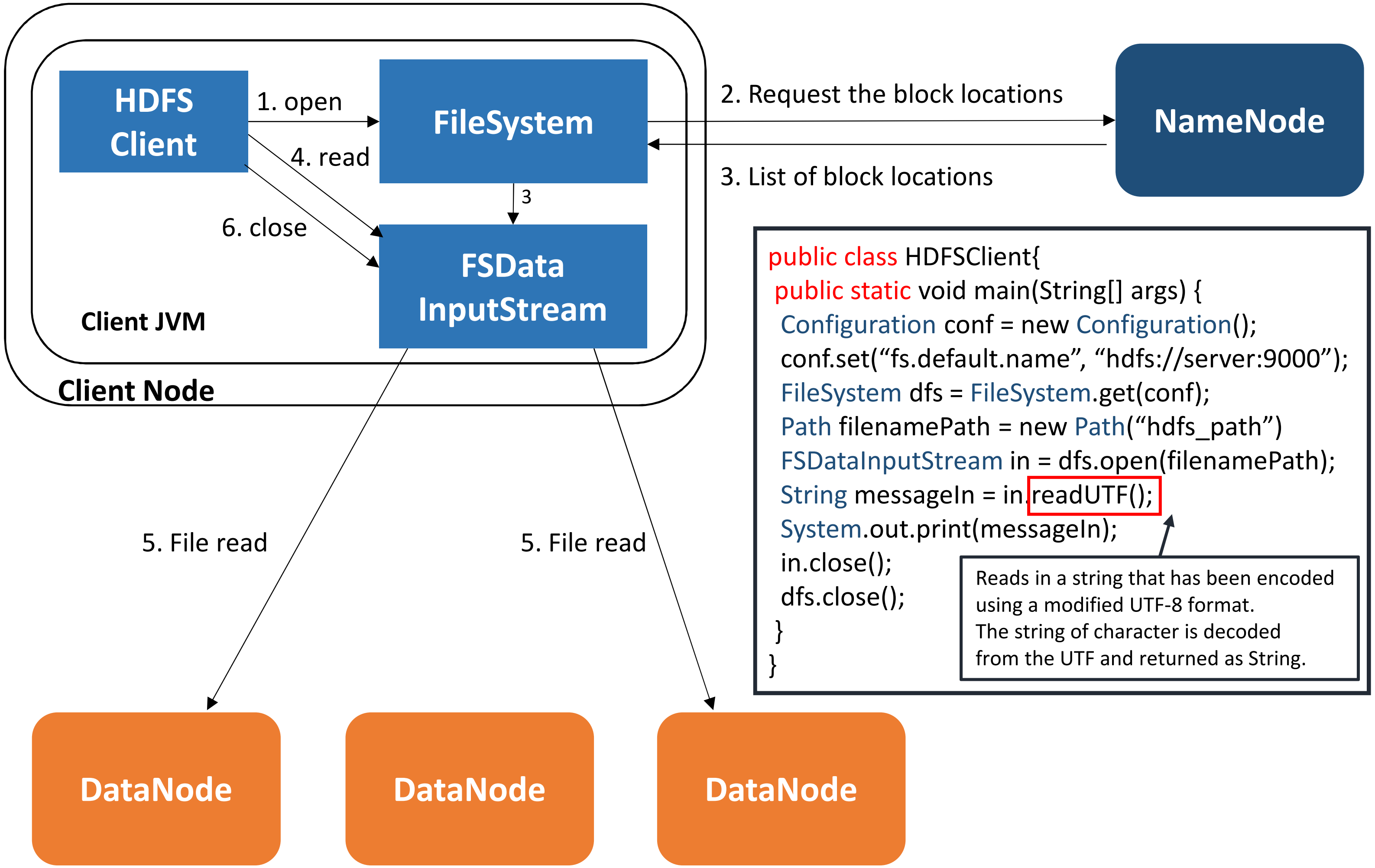
HDFS – Write



```
public class HDFSClient{
 public static void main(String[] args) {
  Configuration conf = new Configuration();
  conf.set(“fs.default.name”, “hdfs://server:9000”);
  FileSystem dfs = FileSystem.get(conf);
  Path filenamePath = new Path(“hdfs_path”)
  FSDataOutputStream out = dfs.create(filenamePath);
  out.writeUTF(“Hello, world!\n”);
  out.close();
  dfs.close();
 }
}
```

Writes a string to the underlying output stream using modified UTF-8 encoding.

HDFS – Read



```
public class HDFSClient{
 public static void main(String[] args) {
  Configuration conf = new Configuration();
  conf.set(“fs.default.name”, “hdfs://server:9000”);
  FileSystem dfs = FileSystem.get(conf);
  Path filenamePath = new Path(“hdfs_path”)
  FSDataInputStream in = dfs.open(filenamePath);
  String messageIn = in.readUTF();
  System.out.print(messageIn);
  in.close();
  dfs.close();
 }
}
```

Reads in a string that has been encoded using a modified UTF-8 format.
The string of character is decoded from the UTF and returned as String.

HDFS – Command

- Invoked by the $HADOOP_HOME/bin/hdfs script
- hdfs [SHELL_OPTIONS] COMMAND [GENERIC_OPTIONS] [COMMAND_OPTIONS]
- hdfs classpath : Prints the class path needed to get the Hadoop jar and the required libraries
- hdfs dfs : Runs a file system command on the file system supported in Hadoop
- hdfs getconf [option] : Gets configuration information from the configuration directory.
  - hdfs getconf –namenodes : Gets list of namenodes in the cluster
- hdfs groups [username] : Returns the group info
- hdfs version : Prints the version

http://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-hdfs/HDFSCommands.html#dfs

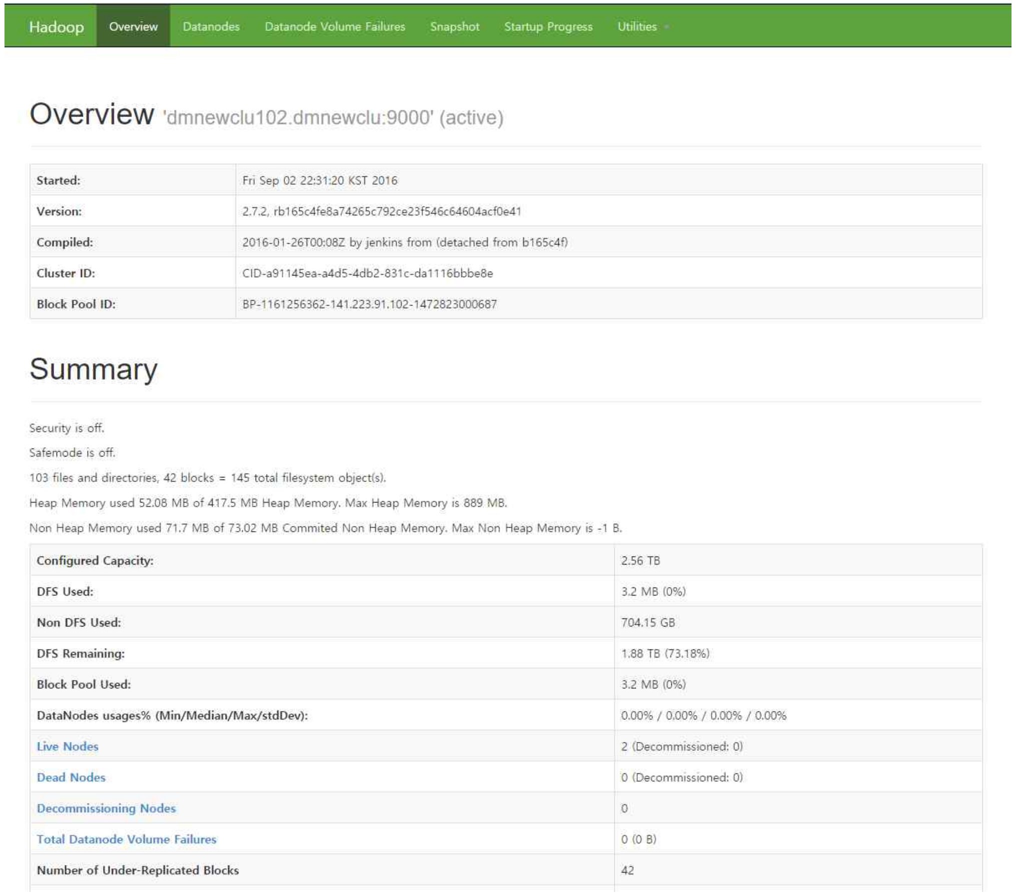
HDFS – File system shell command

- hdfs dfs <args>
  - hdfs dfs –ls [o] <paths> : see file list of the path
  - hdfs dfs –mkdir [o] <paths> : make a directory
  - hdfs dfs –mv URI <dst> : move files
  - hdfs dfs –cp [o] URI <dst> : copy files from source to dst
  - hdfs dfs –rm [o] URI : remove file or directory
  - hdfs dfs –put [o] <localsrc> <dst> : copy files to the HDFS
  - hdfs dfs –get [o] <src> <localdst> : copy files to the local file system
  - hdfs dfs –help

http://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-common/FileSystemShell.html

Practice (Hadoop Daemons)

- http://localhost:50070 : name node
- http://localhost:50075 : data node
- http://localhost:50090 : secondary name node
- http://localhost:8088 : ResourceManager
- http://localhost:8042 : NodeManager



Overview 'dmnewclu102.dmnewclu:9000' (active)

Started:	Fri Sep 02 22:31:20 KST 2016
Version:	2.7.2, rb165c4fe8a74265c792ce23f546c64604acf0e41
Compiled:	2016-01-26T00:08Z by jenkins from (detached from b165c4f)
Cluster ID:	CID-a91145ea-a4d5-4db2-831c-da1116bbbe8e
Block Pool ID:	BP-1161256362-141.223.91.102-1472823000687

Summary

Security is off.

Safemode is off.

103 files and directories, 42 blocks = 145 total filesystem object(s).

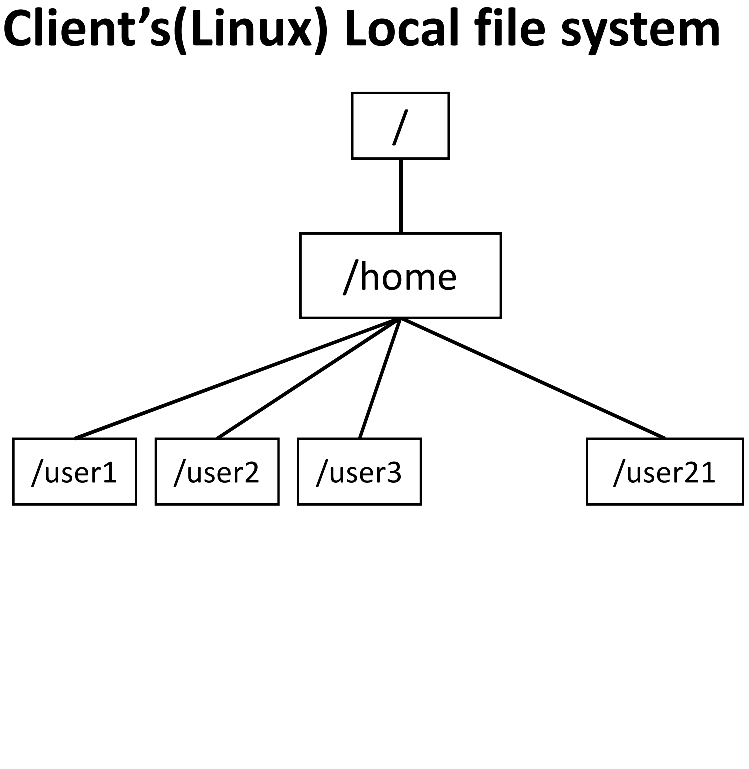
Heap Memory used 52.08 MB of 417.5 MB Heap Memory. Max Heap Memory is 889 MB.

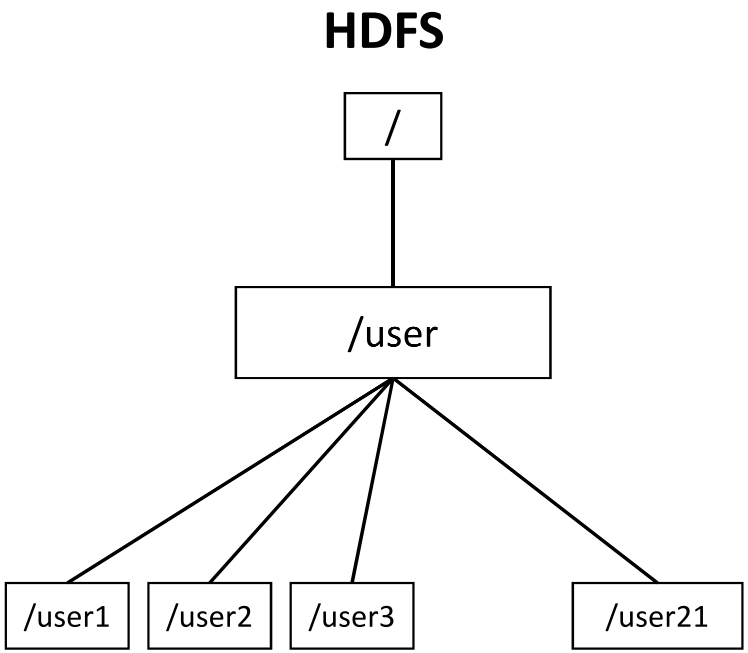
Non Heap Memory used 71.7 MB of 73.02 MB Commited Non Heap Memory. Max Non Heap Memory is -1 B.

Configured Capacity:	2.56 TB
DFS Used:	3.2 MB (0%)
Non DFS Used:	704.15 GB
DFS Remaining:	1.88 TB (73.18%)
Block Pool Used:	3.2 MB (0%)
DataNodes usages% (Min/Median/Max/stdDev):	0.00% / 0.00% / 0.00% / 0.00%
Live Nodes	2 (Decommissioned: 0)
Dead Nodes	0 (Decommissioned: 0)
Decommissioning Nodes	0
Total Datanode Volume Failures	0 (0 B)
Number of Under-Replicated Blocks	42

Practice (Local file system and HDFS)

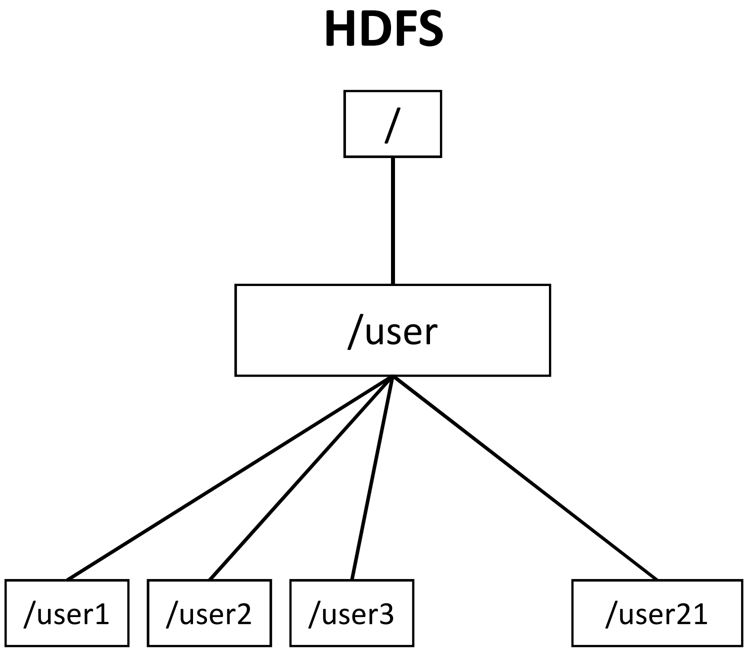
hdfs dfs –mkdir /user/user21/input = hdfs dfs –mkdir input





Practice (Making directories on HDFS)

- Build Default HDFS file system structure
  - hdfs dfs –mkdir /user
  - hdfs dfs –mkdir /user/”your_id”



Practice (Dealing with HDFS)

- Copy Hadoop *README.txt* from Hadoop directory to your home directory.
  - *cp /usr/local/hadoop-2.8.1/README.txt ~/*
- Make a directory on HDFS
  - *hdfs dfs -mkdir input (=hdfs dfs –mkdir /user/"your_id"/input)*
- Upload README.txt file to HDFS.
  - *hdfs dfs -put README.txt input/input.txt*

    README.txt: Input file from local

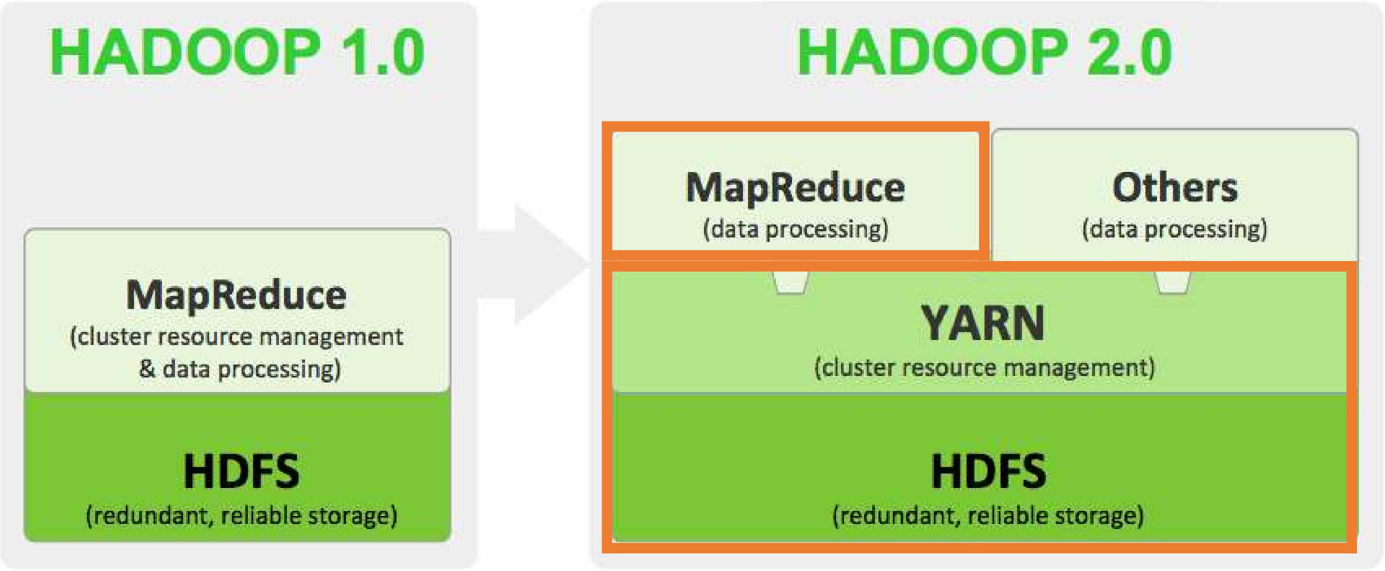
    input/input.txt: Copy the file to HDFS
  - *hdfs dfs -cat input/input.txt*

    input/input.txt: file location on HDFS

Yarn, MapReduce

- Why Yarn?

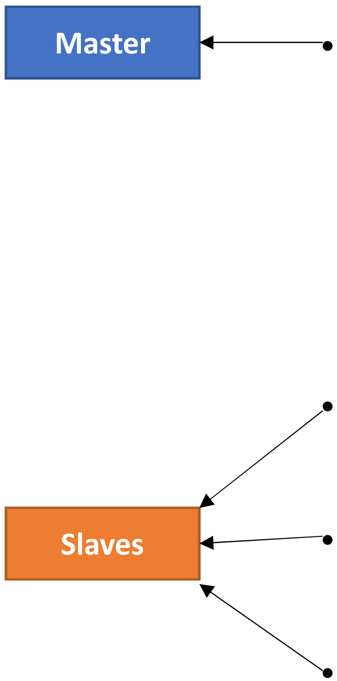
1. General, multi purpose cluster
2. Cluster utilization
3. Load balance


HADOOP 1.0
MapReduce
(cluster resource management & data processing)
HDFS
(redundant, reliable storage)
HADOOP 2.0
MapReduce
(data processing)
Others
(data processing)
YARN
(cluster resource management)
HDFS
(redundant, reliable storage)

MapReduce is just one application of various jobs which are executed on the YARN platform

Yarn (Yet Another Resource Negotiator)

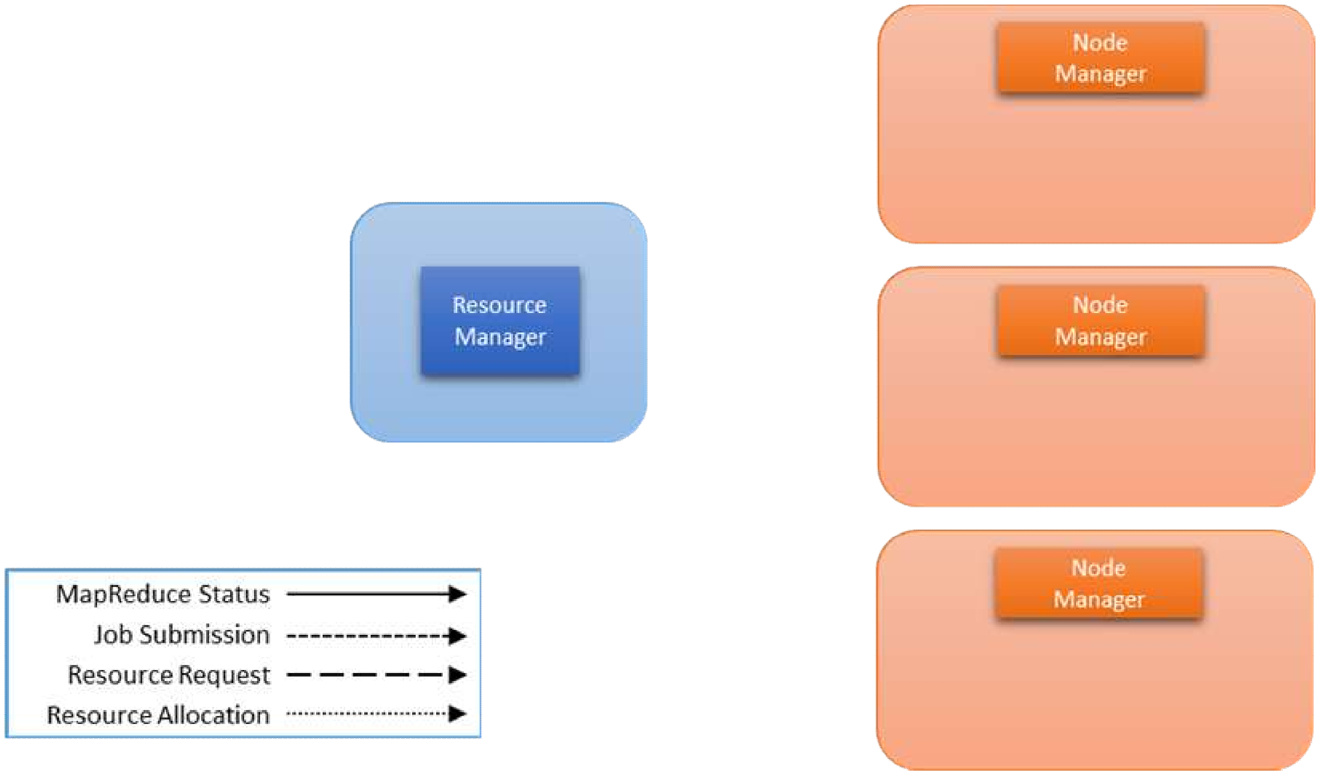
- Yarn – New architecture introduced in Hadoop-0.23
  - Cluster resource manager
  - Consists of 2 constant daemons and 2 temporary daemons



- Resource manager / entire cluster, Always
  - – Manages the global assignment of compute resources
    - Scheduler : allocate resources
    - Applications Manager : responsible for accepting job-submissions

- Node Manager / each slaves, Always
  - – Manages the Container on that machine
- Application master / each application, When the job is submitted
  - – Negotiate resources from the Scheduler and work with Node manager
- Container / depends on resource requirements, When resource request occur
  - – Abstract notion of a resource (cpu, memory, disk, network, etc)

Yarn process – MapReduce case



Resource Manager

Node Manager

Node Manager

Node Manager

MapReduce Status

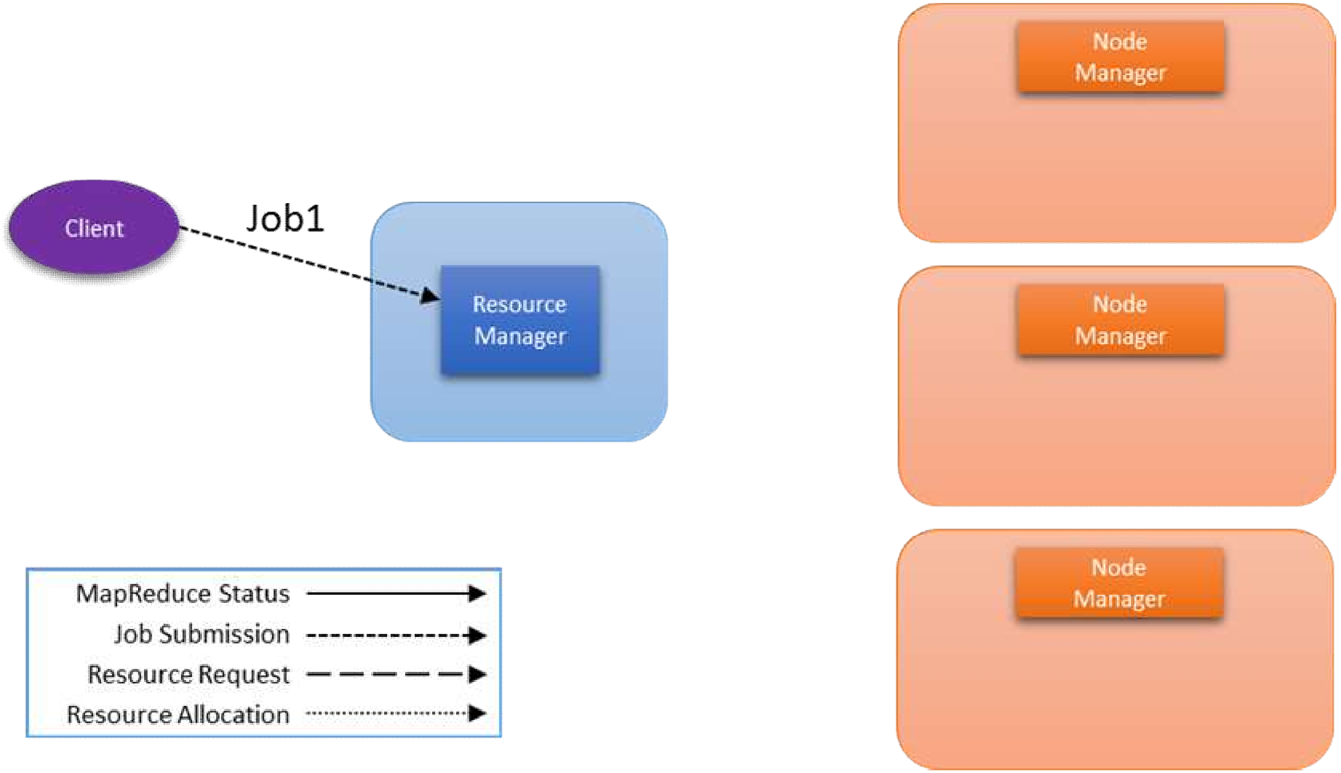
Job Submission

Resource Request

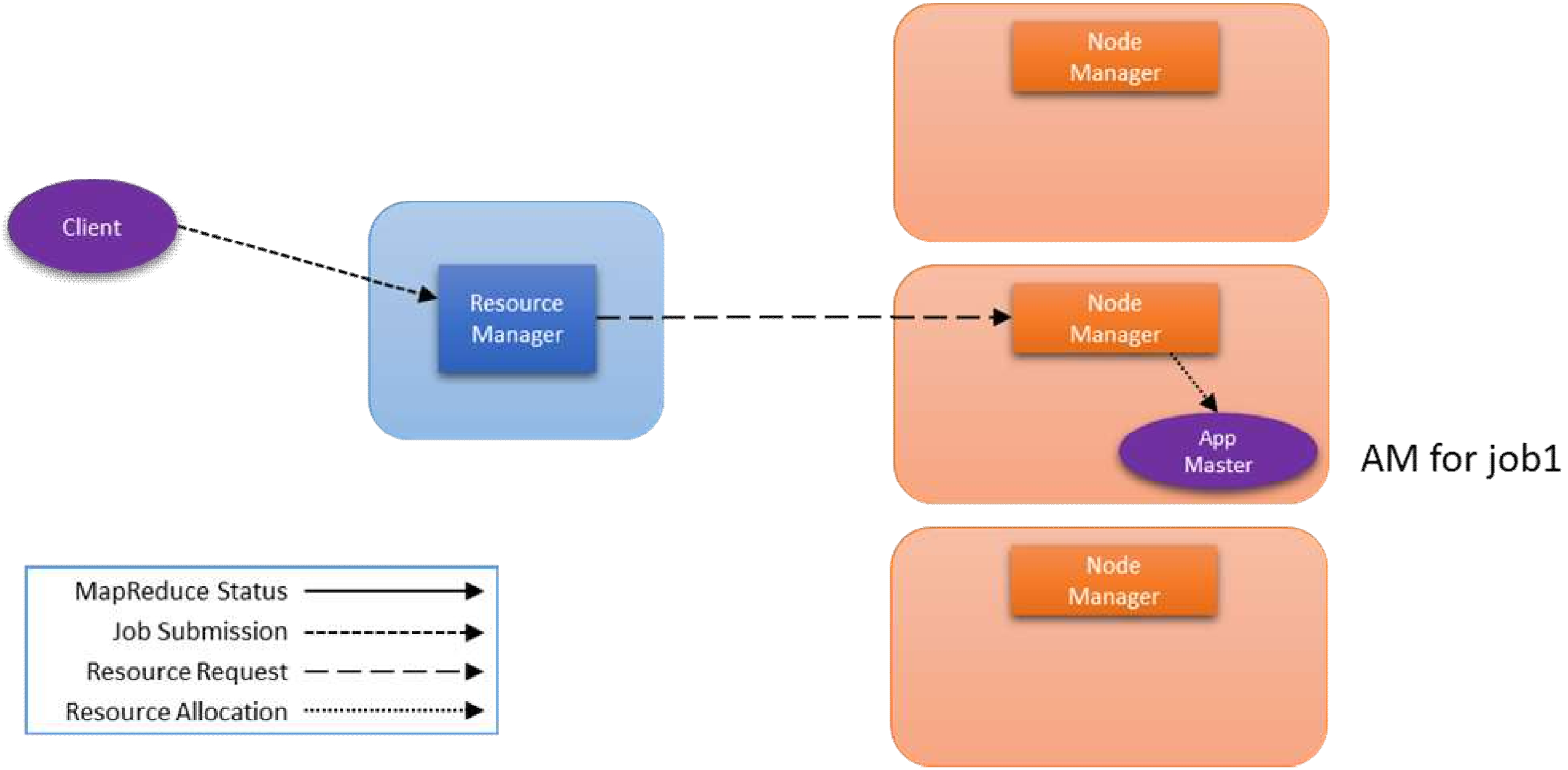
Resource Allocation

http://www.slideshare.net/gopass2002/yarn-overview?qid=f75ecc82-5366-4f42-9c2e-e3ce7d240b9f&v=default&b=&from_search=2 ,SlideShare, SeongHyun Jdeong

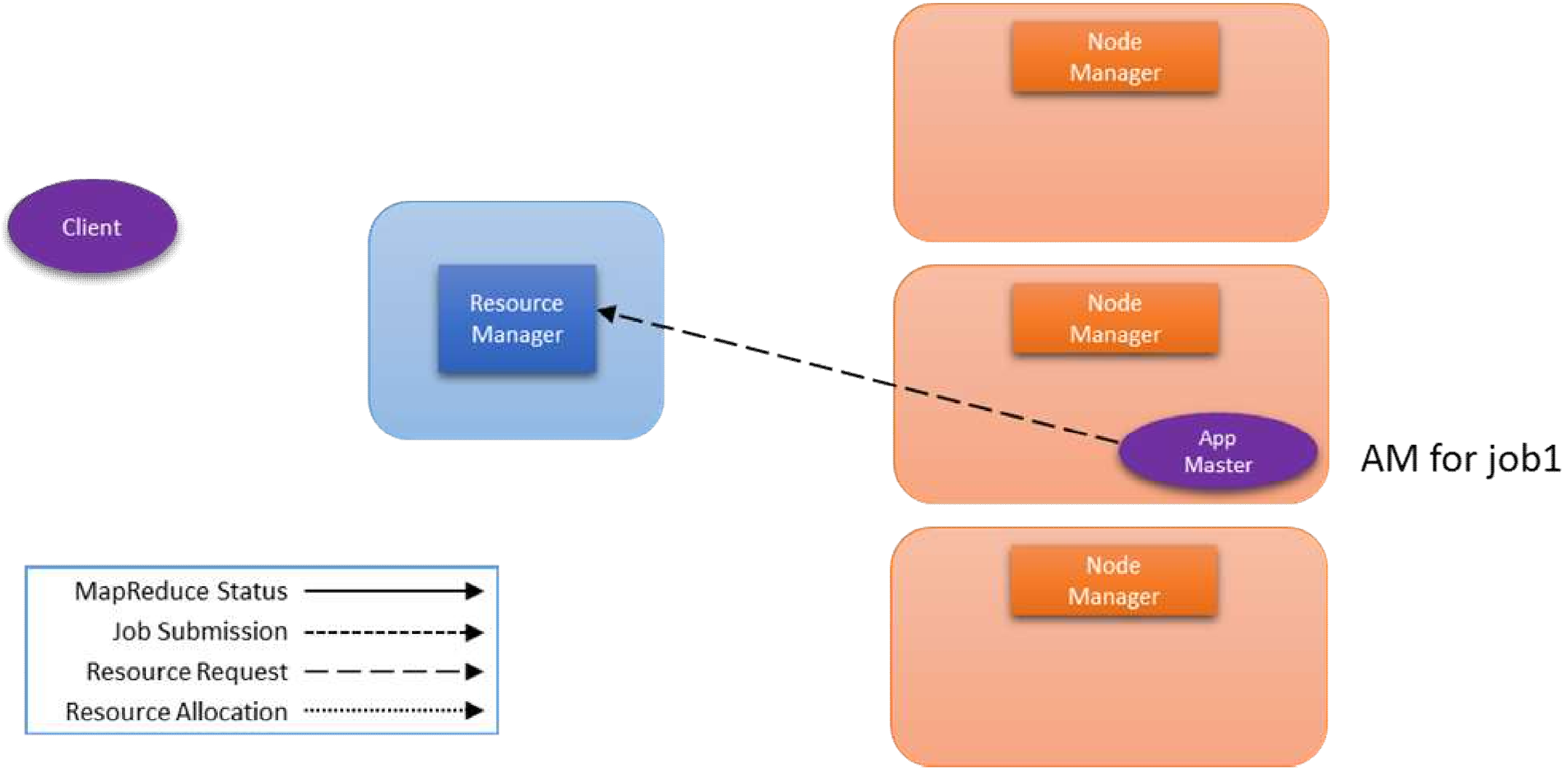
Yarn process – MapReduce case



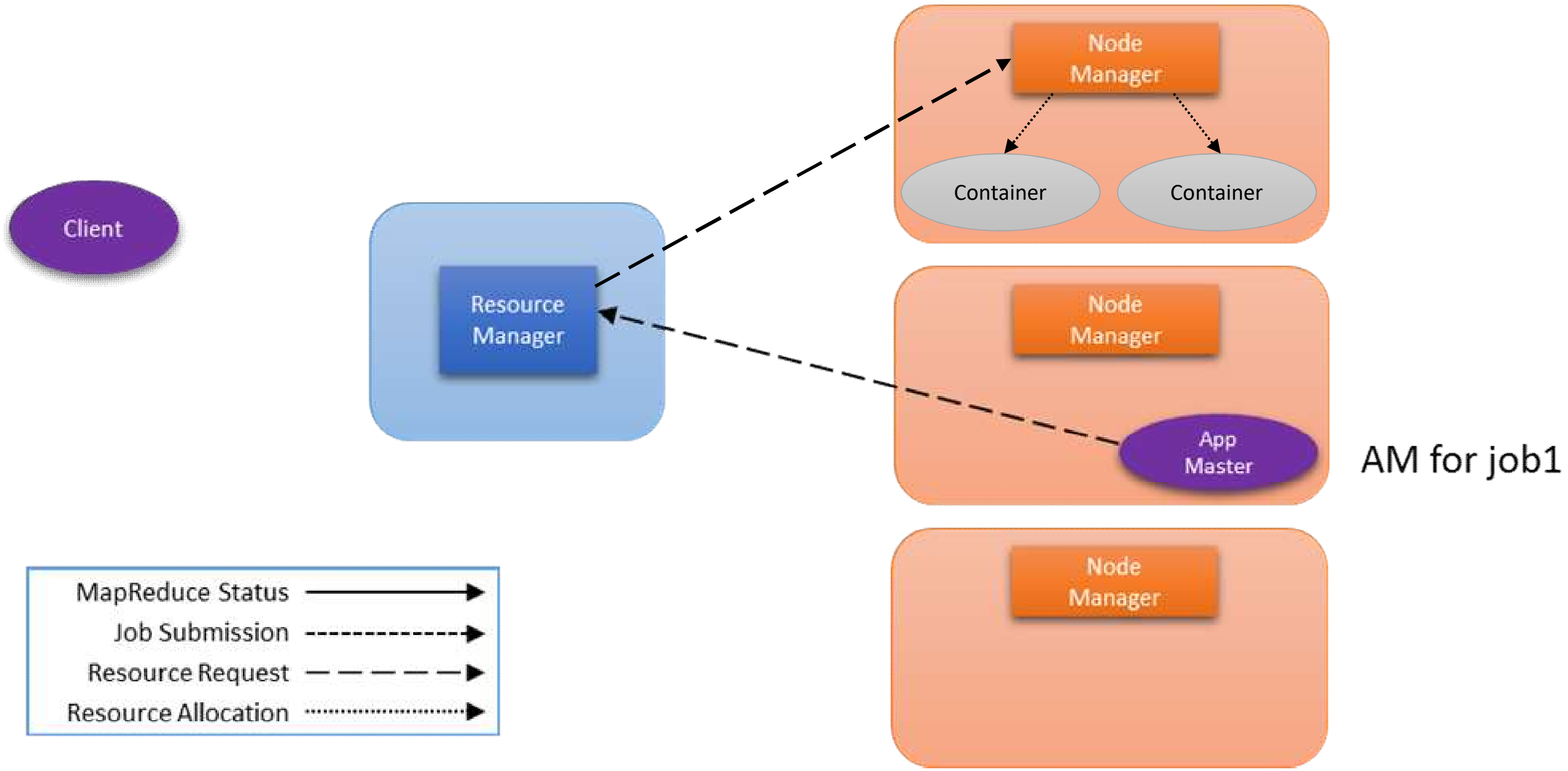
Yarn process – MapReduce case



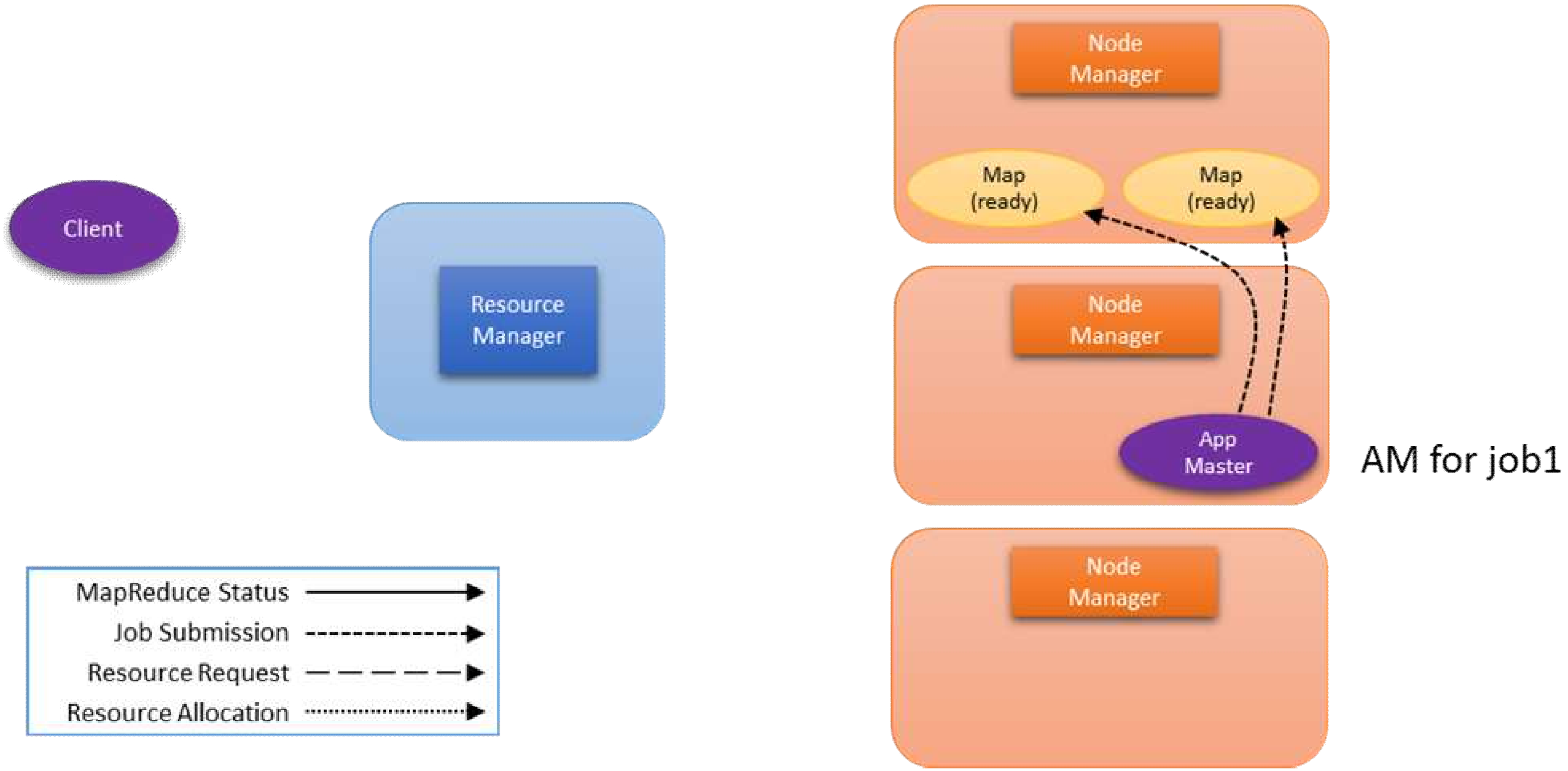
Yarn process – MapReduce case



Yarn process – MapReduce case

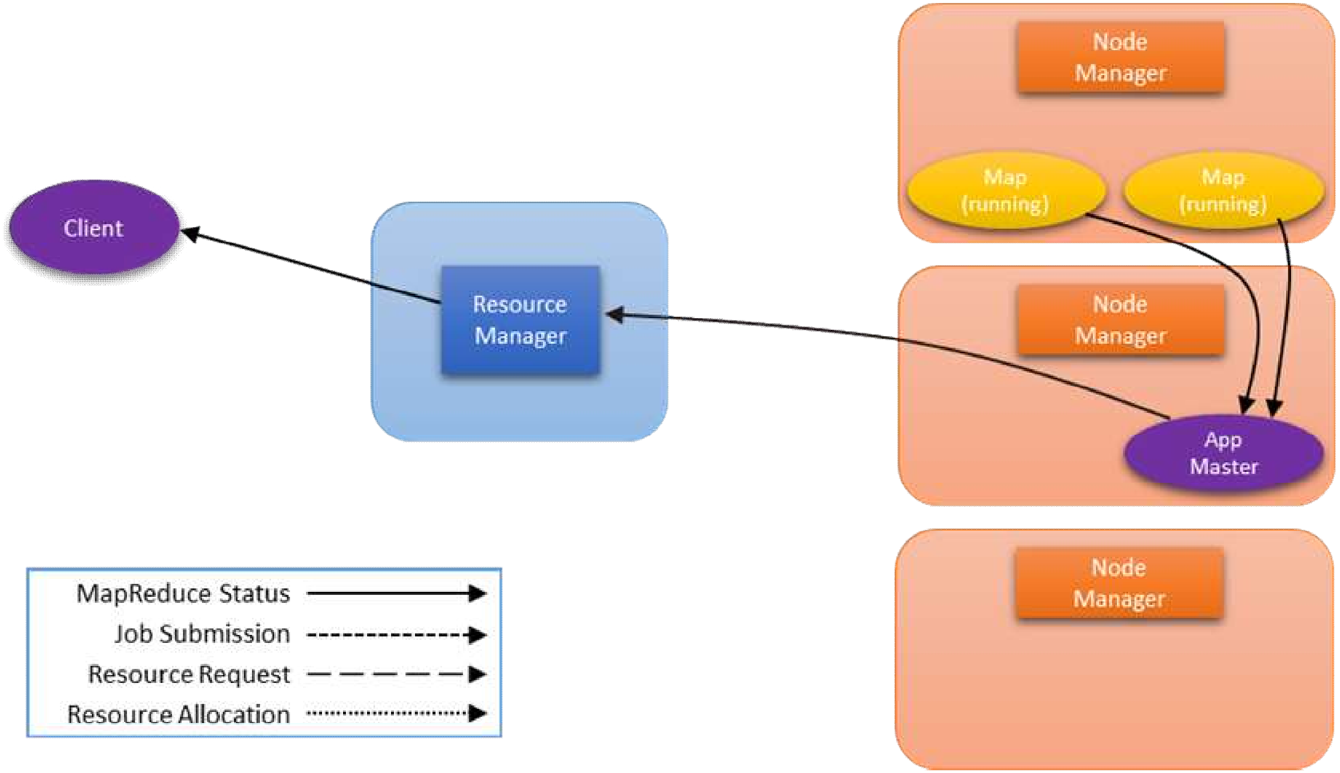


Yarn process – MapReduce case



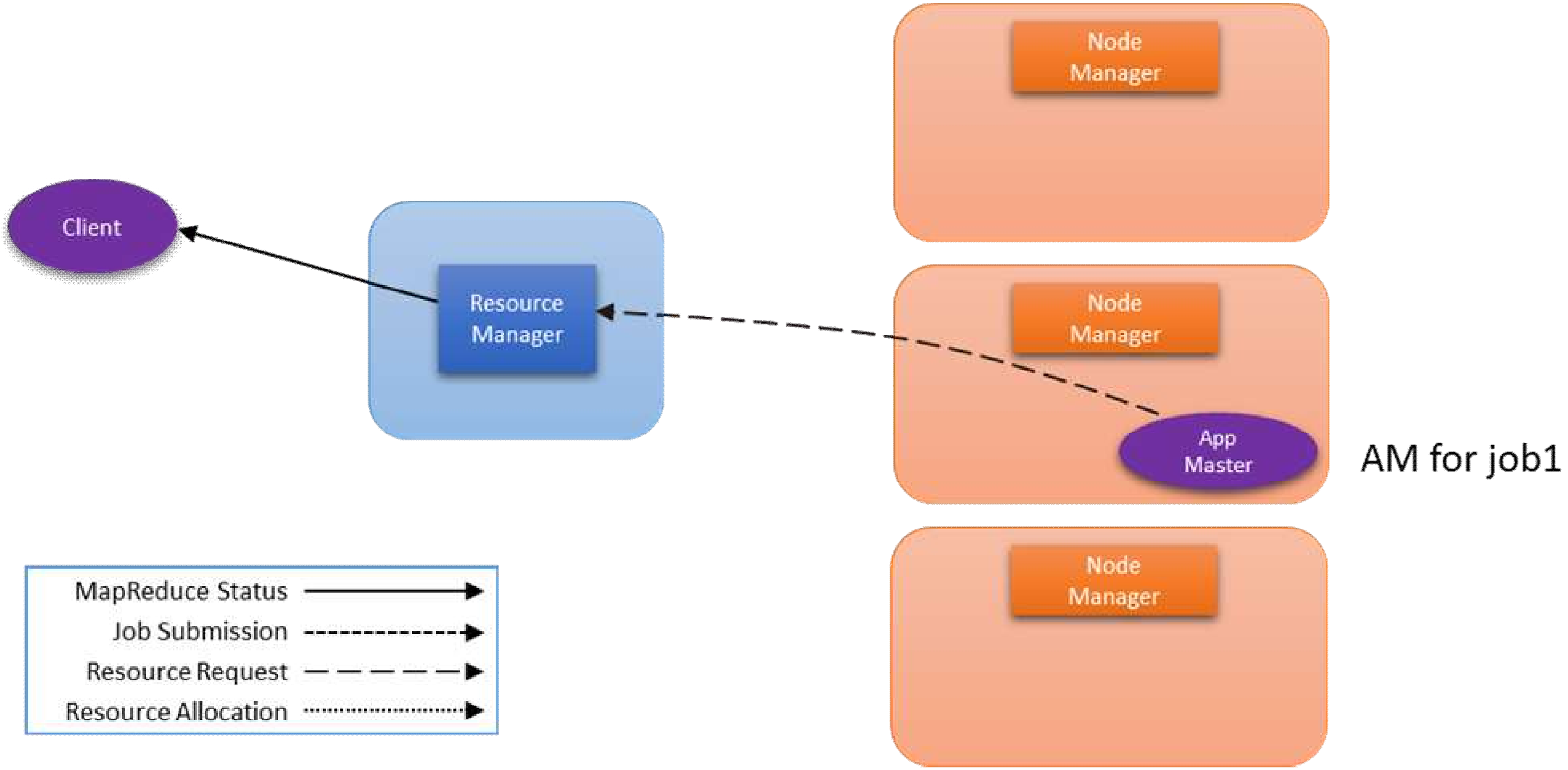
AM for job1

Yarn process – MapReduce case

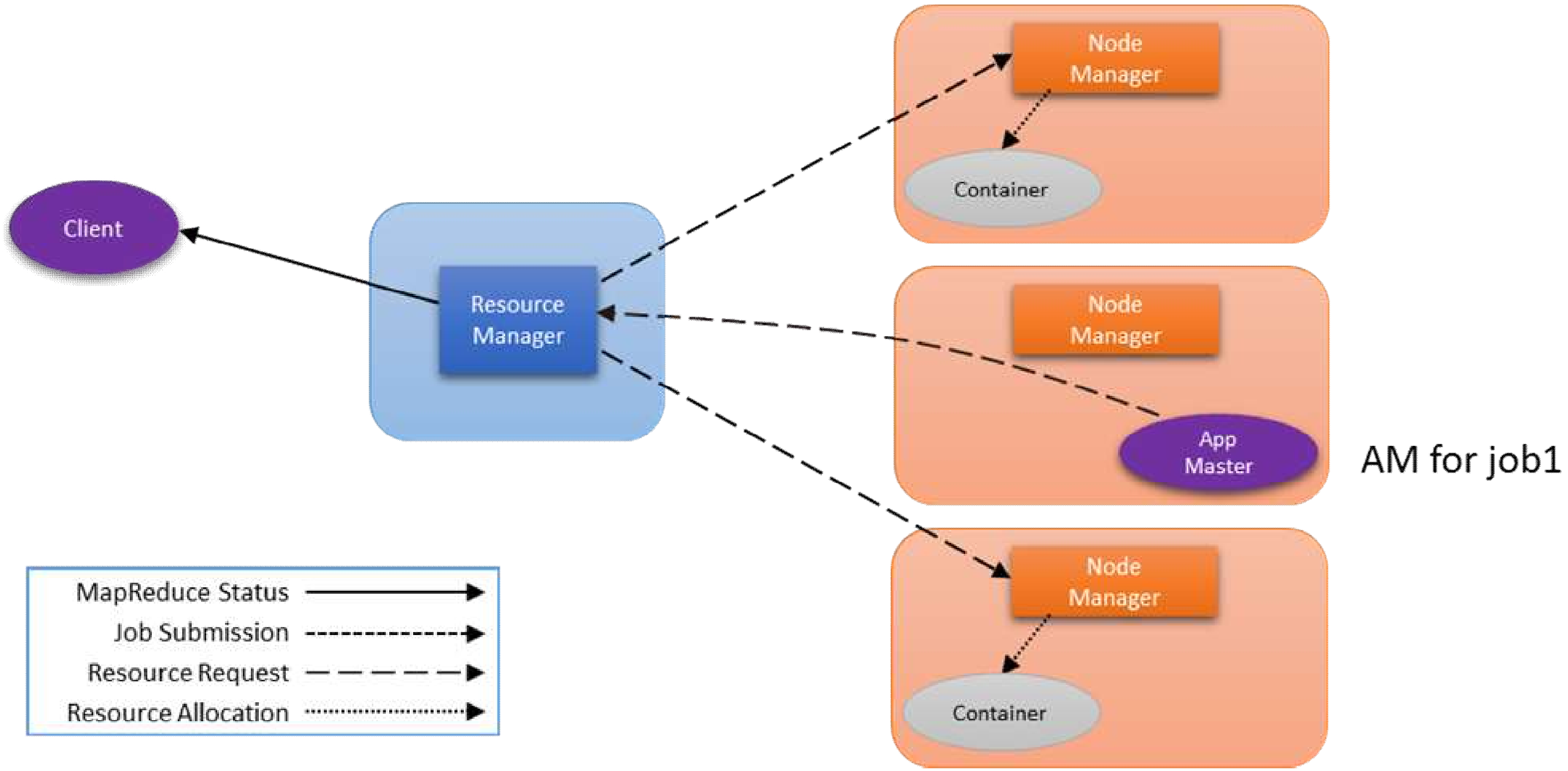


AM for job1

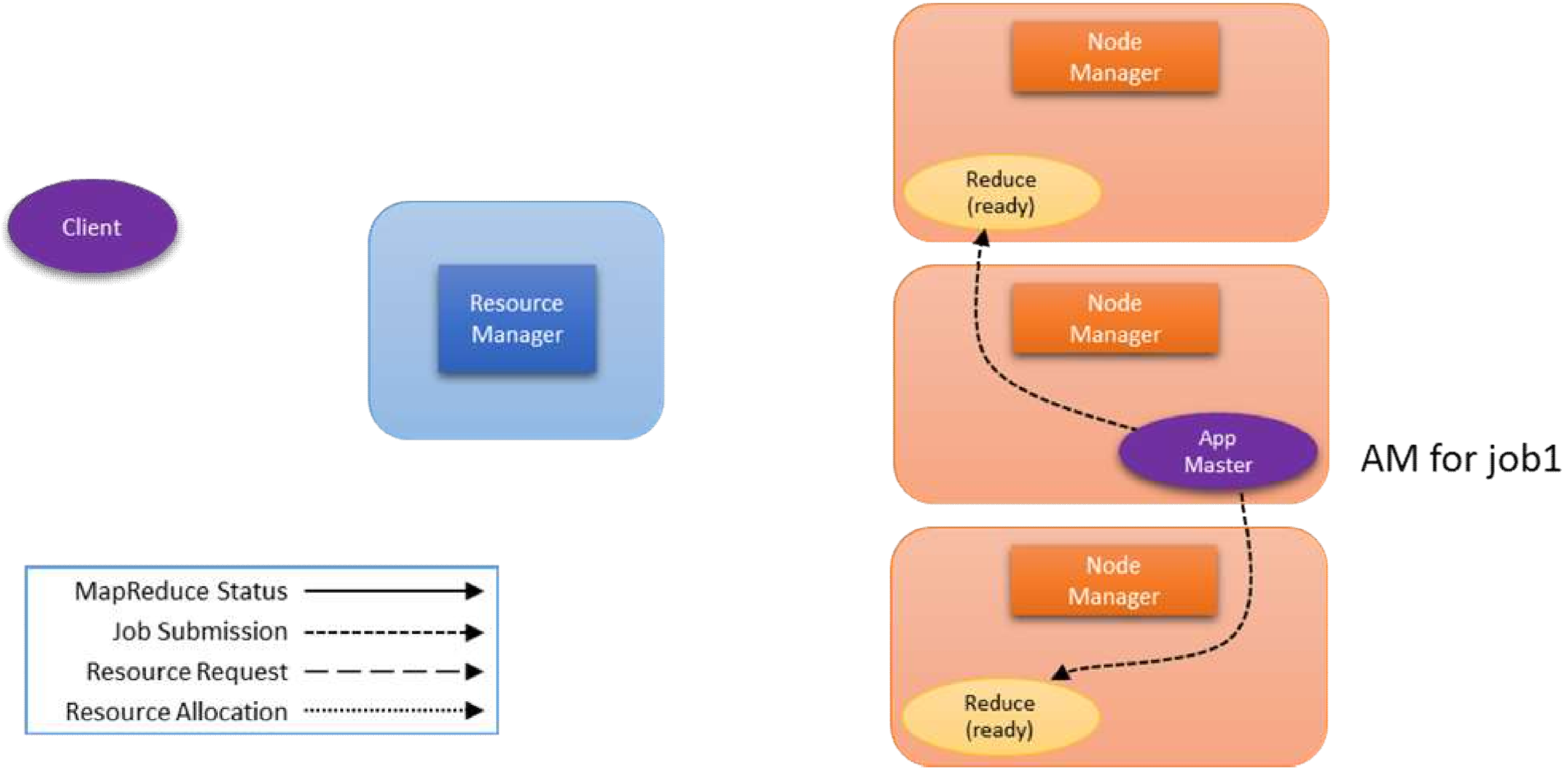
Yarn process – MapReduce case



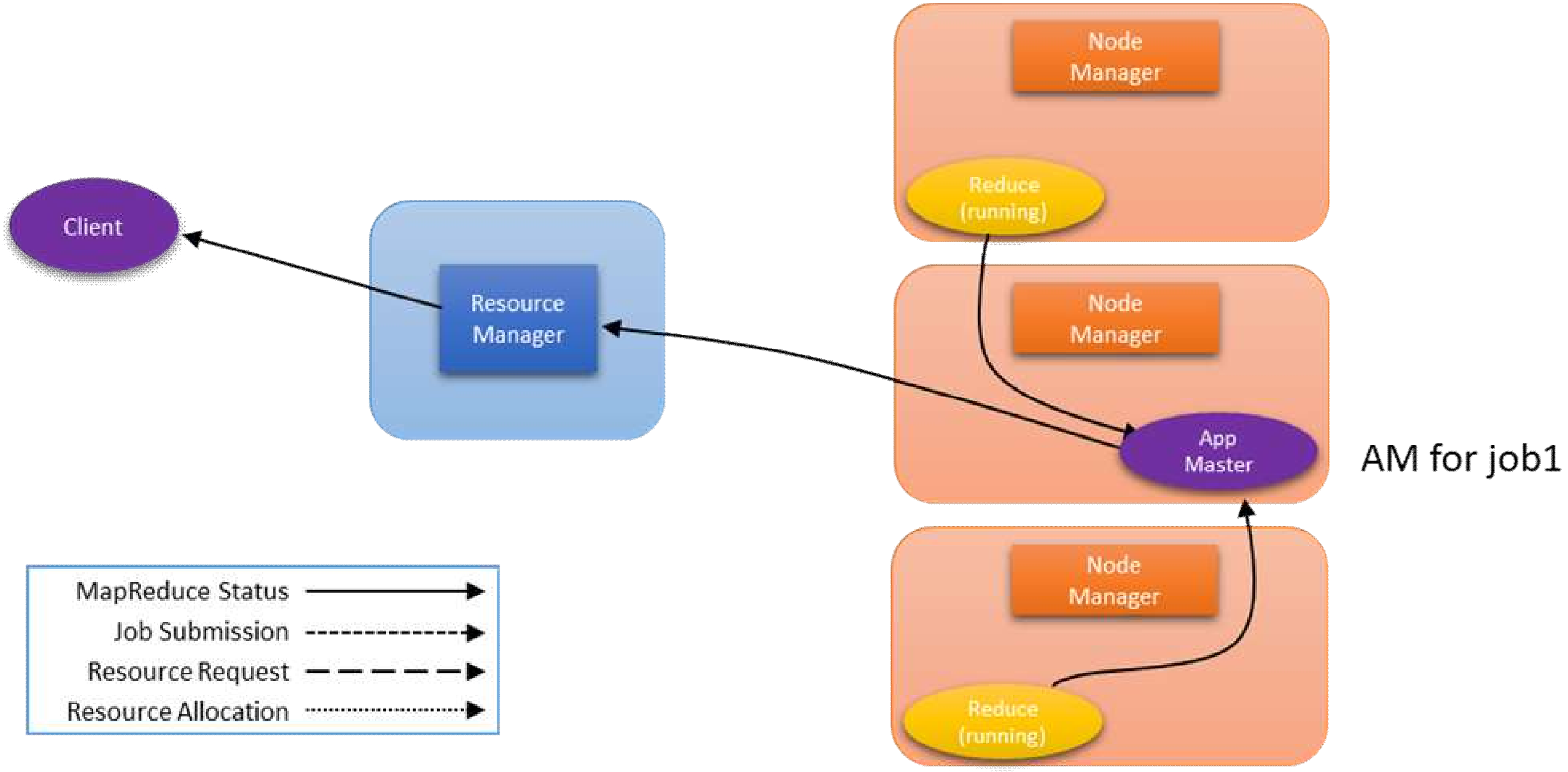
Yarn process – MapReduce case



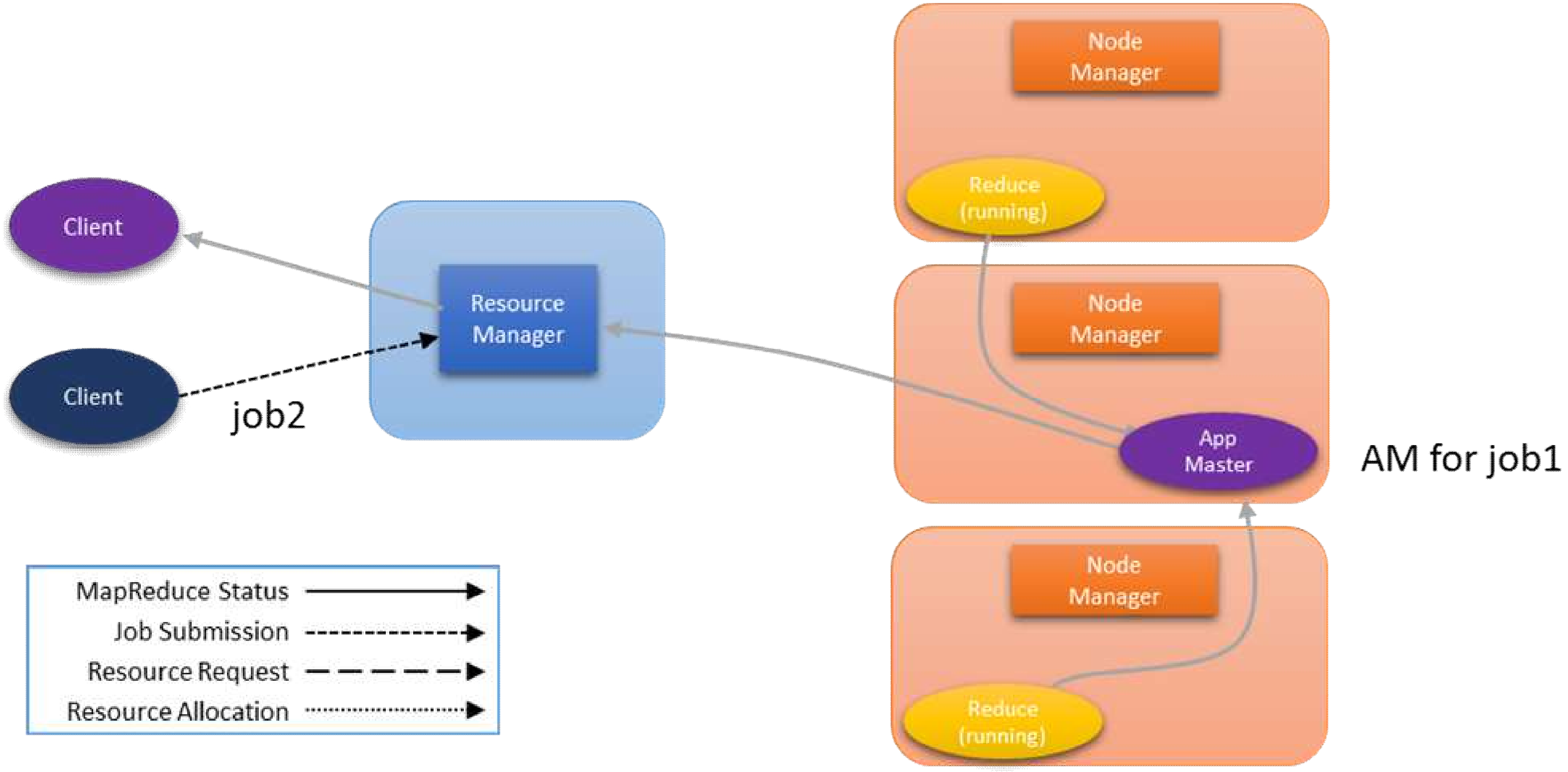
Yarn process – MapReduce case



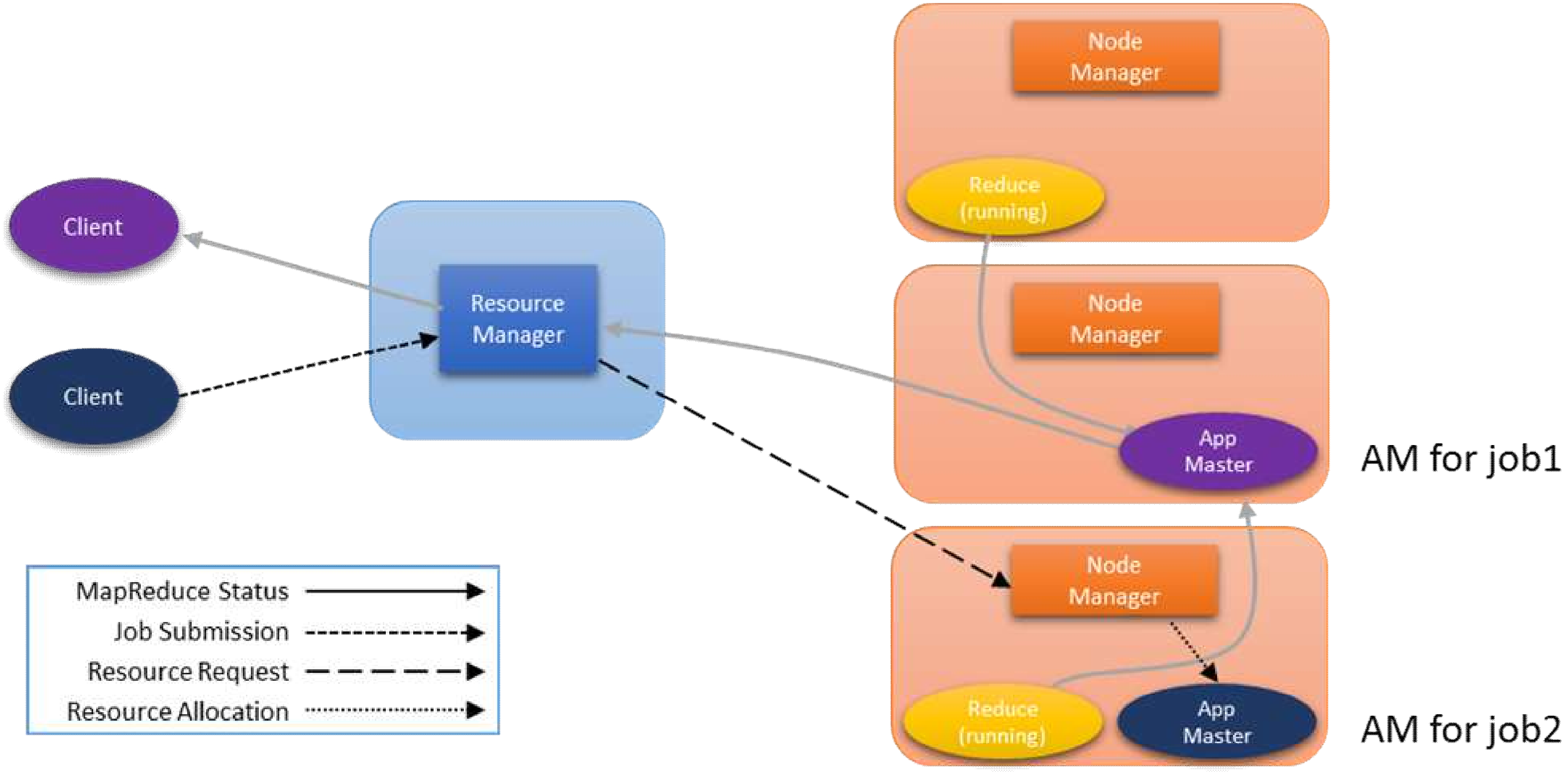
Yarn process – MapReduce case



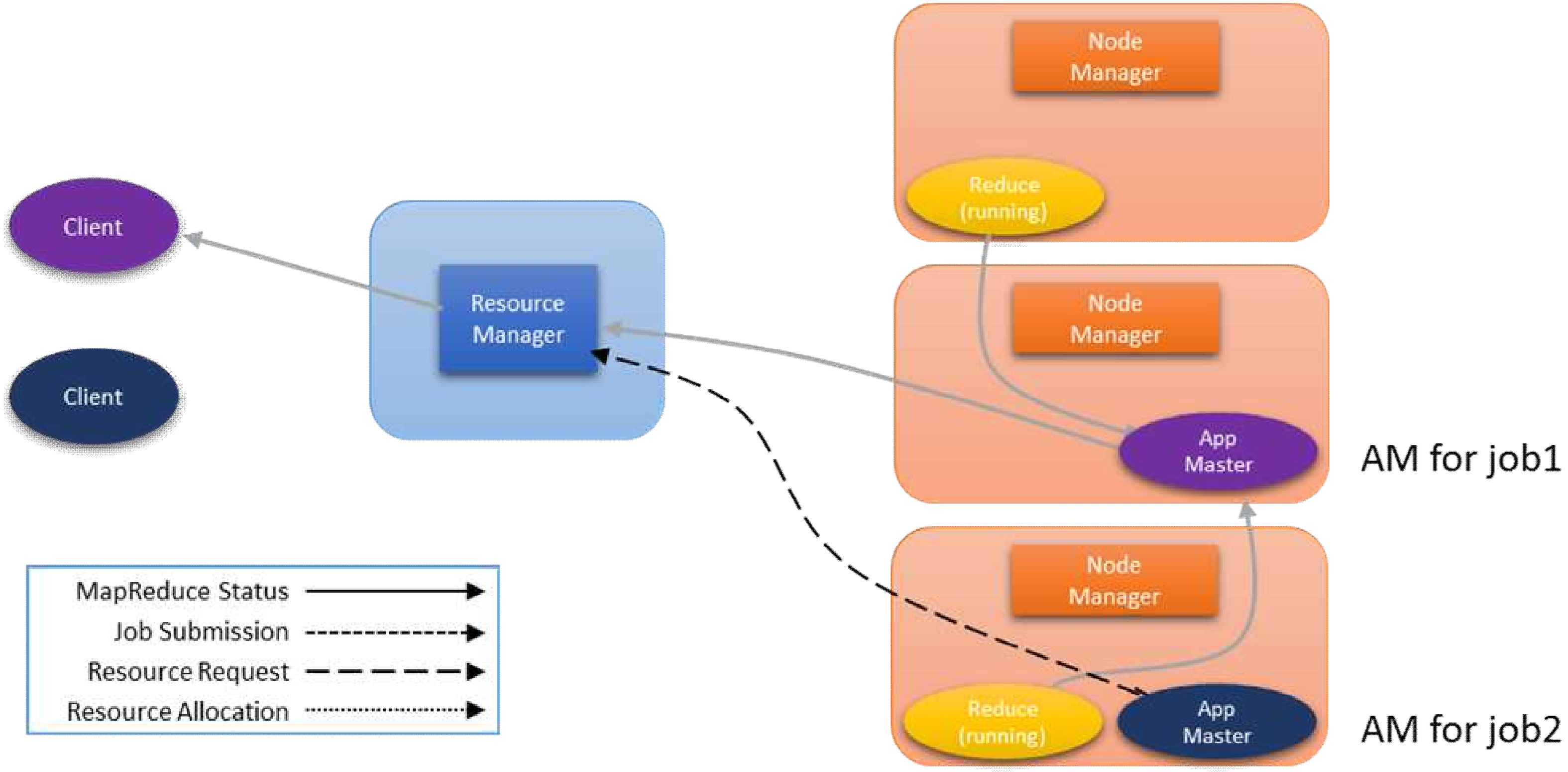
Yarn process – MapReduce case



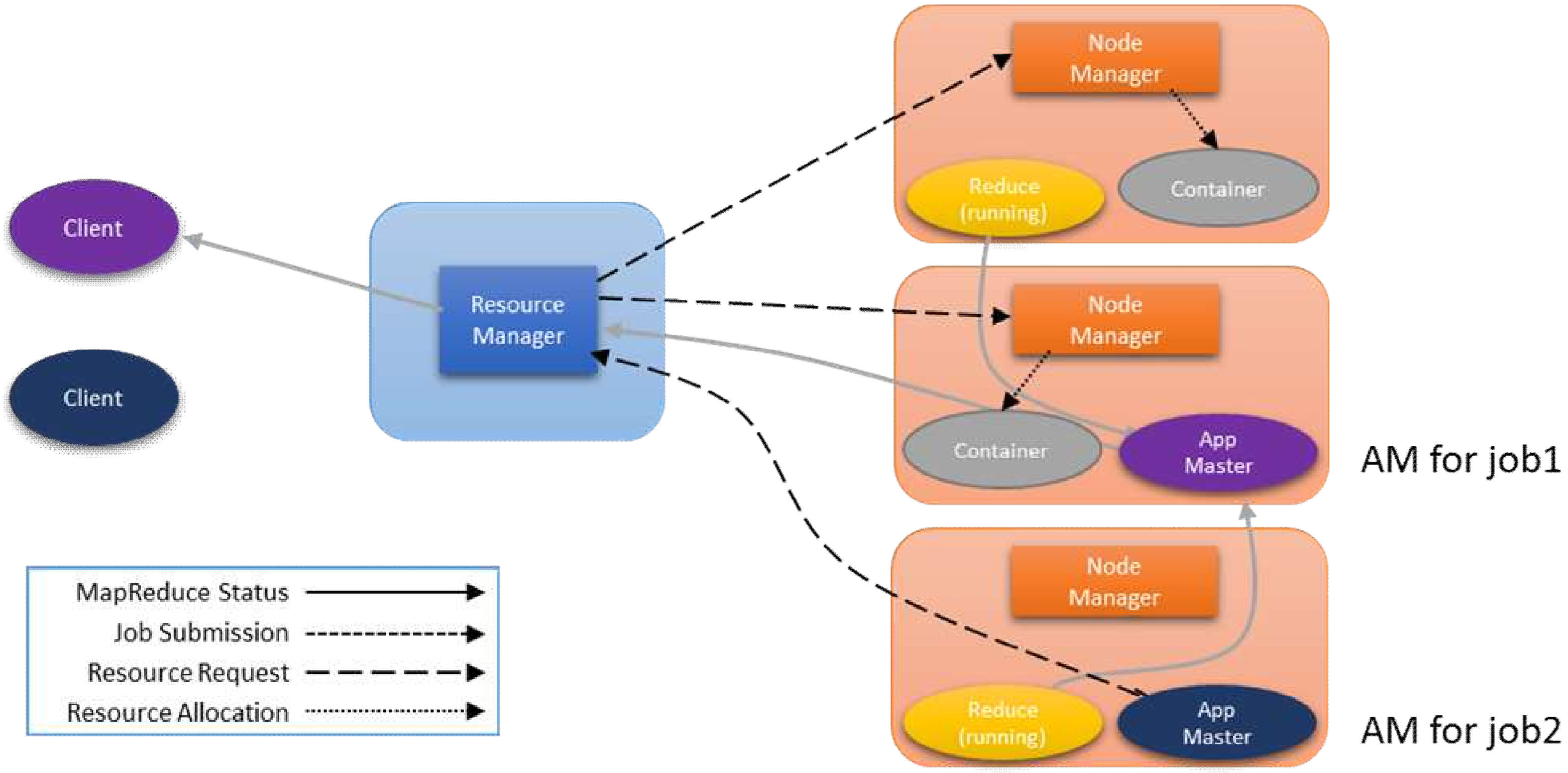
Yarn process – MapReduce case



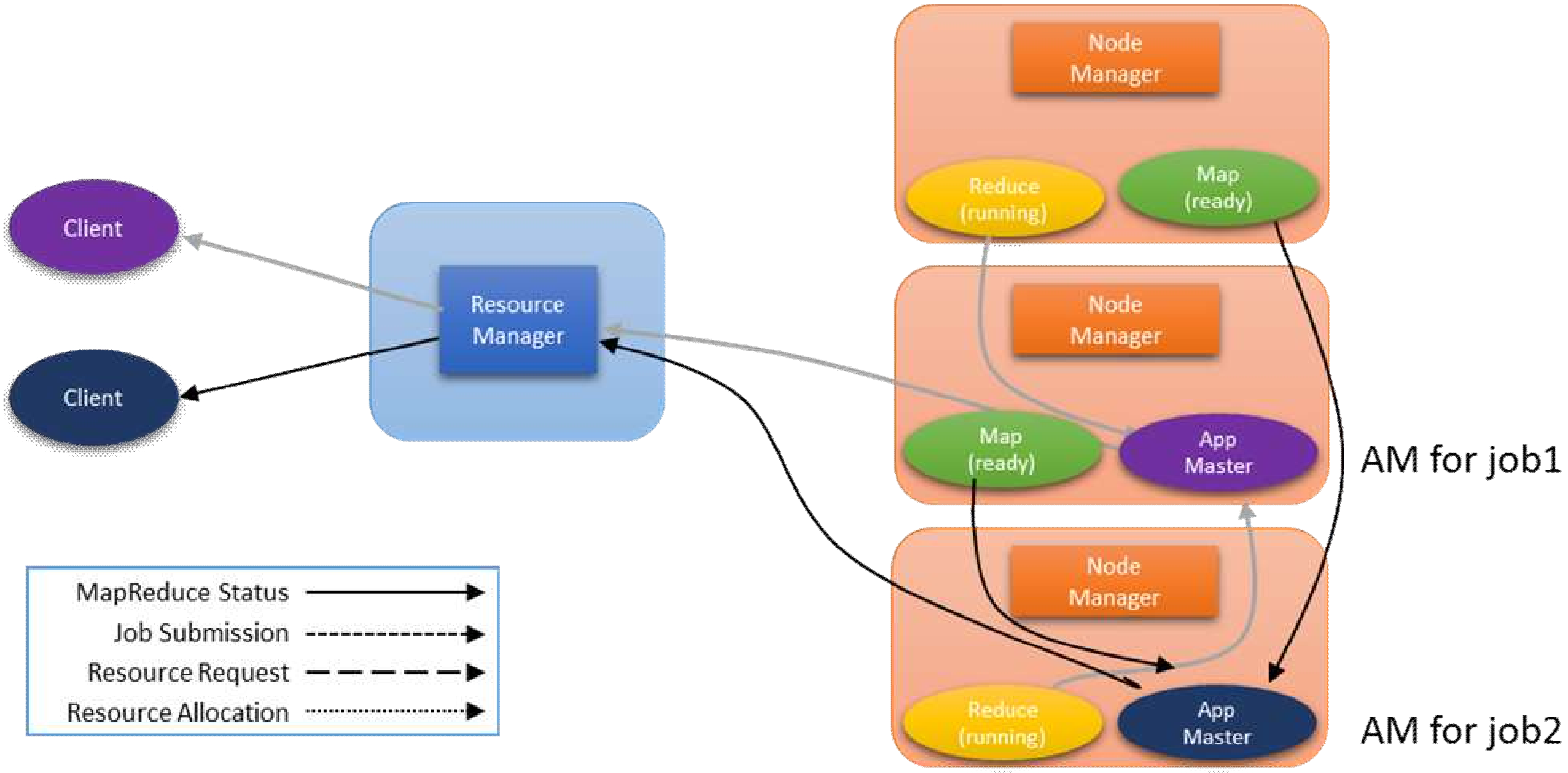
Yarn process – MapReduce case



Yarn process – MapReduce case



Yarn process – MapReduce case


Node Manager
Reduce (running)
Map (ready)
Client
Resource Manager
Client
Node Manager
Map (ready)
App Master
AM for job1
Node Manager
Reduce (running)
App Master
AM for job2
MapReduce Status
Job Submission
Resource Request
Resource Allocation