HDFS

Senthil Kumar A

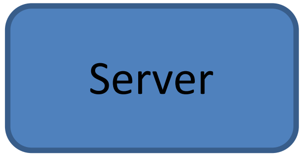
DataDotZ

Basics – *optional*

- Program
  - sequence of instructions written to perform a specified task with a computer
  - or a piece of code
- Process
  - an instance of a computer program that is being executed.
  - or a execution of a program
- Daemon Process
  - process which runs in background and has no controlling terminal.
- JVM – Java Virtual Machine
  - program which executes certain programs, namely those containing Java bytecode instructions

Basics – *optional*

- Client-server Concept
  - Client sends requests to one or more servers which in turn accepts, processes them and return the requested information to the client.
  - A server might run a software/program which listens on particular ip and port number for requests
    - Servers can also be said as daemons
  - Examples:
    - Server - web server
    - Client – web browser



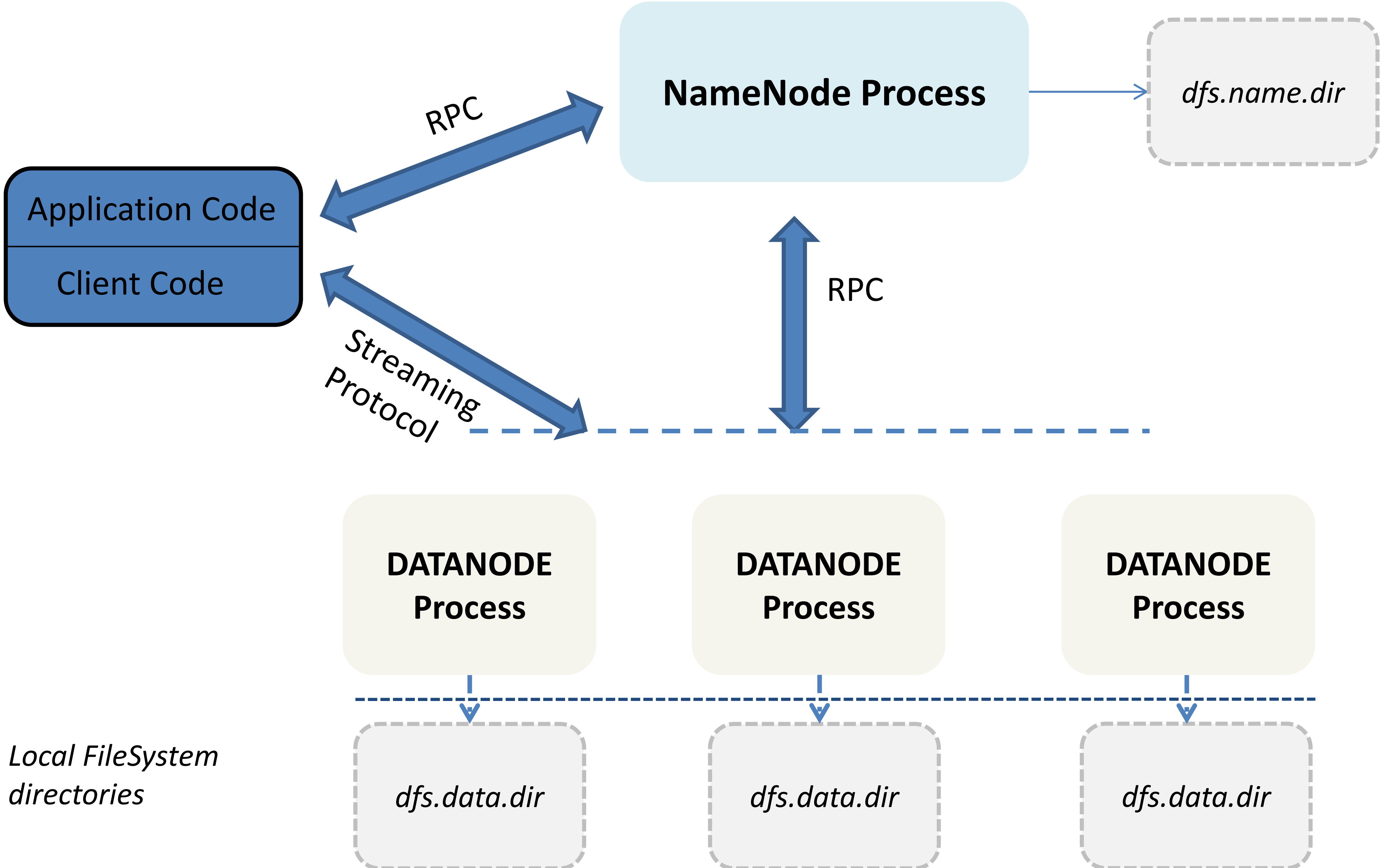
Server

Introduction

- A distributed File System - **STORAGE**
  - A File System on multiple machines which sits on native filesystem
    - ext4,ext3
- Hardware Failure
  - Due to usage of Commodity machines, failure is a common phenomenon
  - Designed for failure
- Large Data Sets
  - Small Files Problem Due to NameNode
- Simple Coherency Model
  - Write Once , Read Many Times
- Streaming Data Access
  - High Throughput instead of low latency access

- ext3? ext4?

Continued…


NameNode Process
dfs.name.dir
RPC
Application Code
Client Code
RPC
Streaming Protocol
DATANODE Process
DATANODE Process
DATANODE Process
Local FileSystem directories
dfs.data.dir
dfs.data.dir
dfs.data.dir

Daemons in Hadoop Core

- NameNode
- DataNode
- Secondary NameNode

HDFS

- Resource Manager*
- Node Manager*

YARN

* - will be seen later during MR

Block Concept

TestFile1.txt -> 1GB
Block Size -> 64 MB

No of Blocks = 1GB / 64MB = 16 blocks
Blocks are B1,B2,…..B16

Files are splitted into number of chunks(Blocks) of pre-defined size

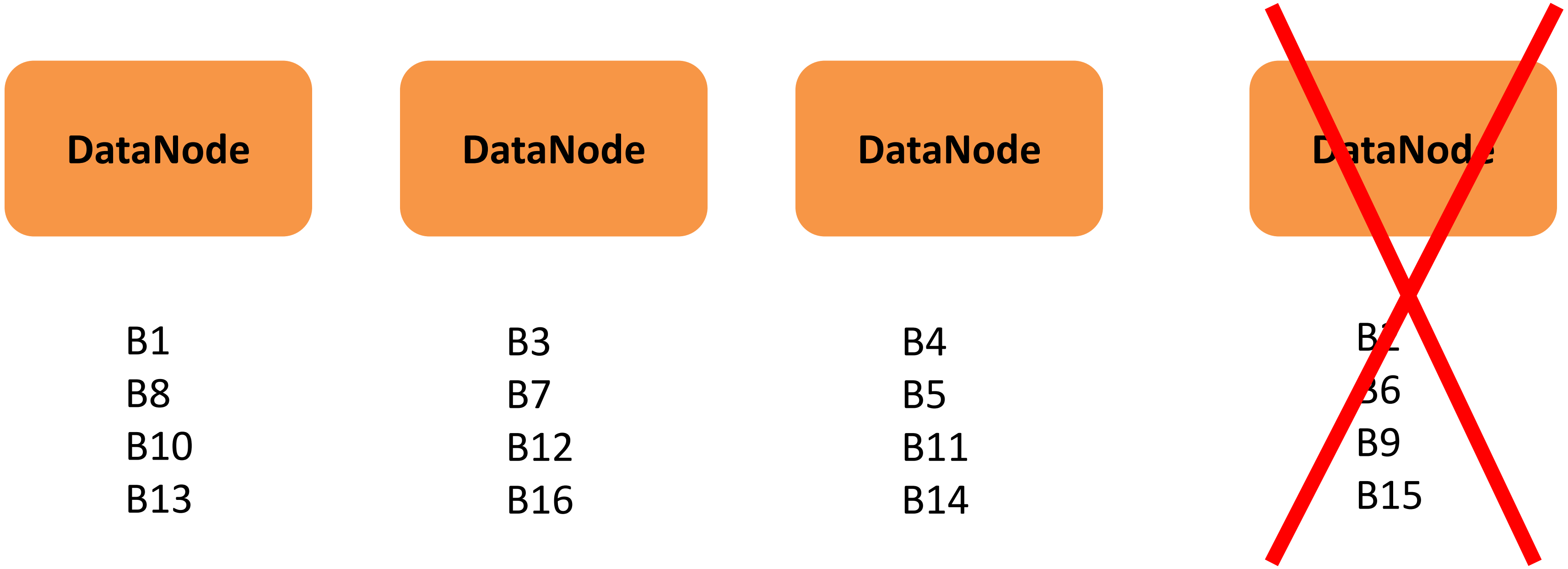
DataNode	DataNode	DataNode	DataNode
B1	B3	B4	B2
B8	B7	B5	B6
B10	B12	B11	B9
B13	B16	B14	B15

Block Concept

What happens to my data if node 4 goes down??

TestFile1.txt -> 1GB
Block Size -> 64 MB

No of Blocks = 1GB / 64MB = 16 blocks
Blocks are B1,B2,…..B16



DataNode	DataNode	DataNode	DataNode
B1	B3	B4	B2
B8	B7	B5	B6
B10	B12	B11	B9
B13	B16	B14	B15

Fault Tolerant in HDFS

HDFS provides fault tolerant by replication of each block by 3

TestFile1.txt -> 1GB
Block Size -> 64 MB

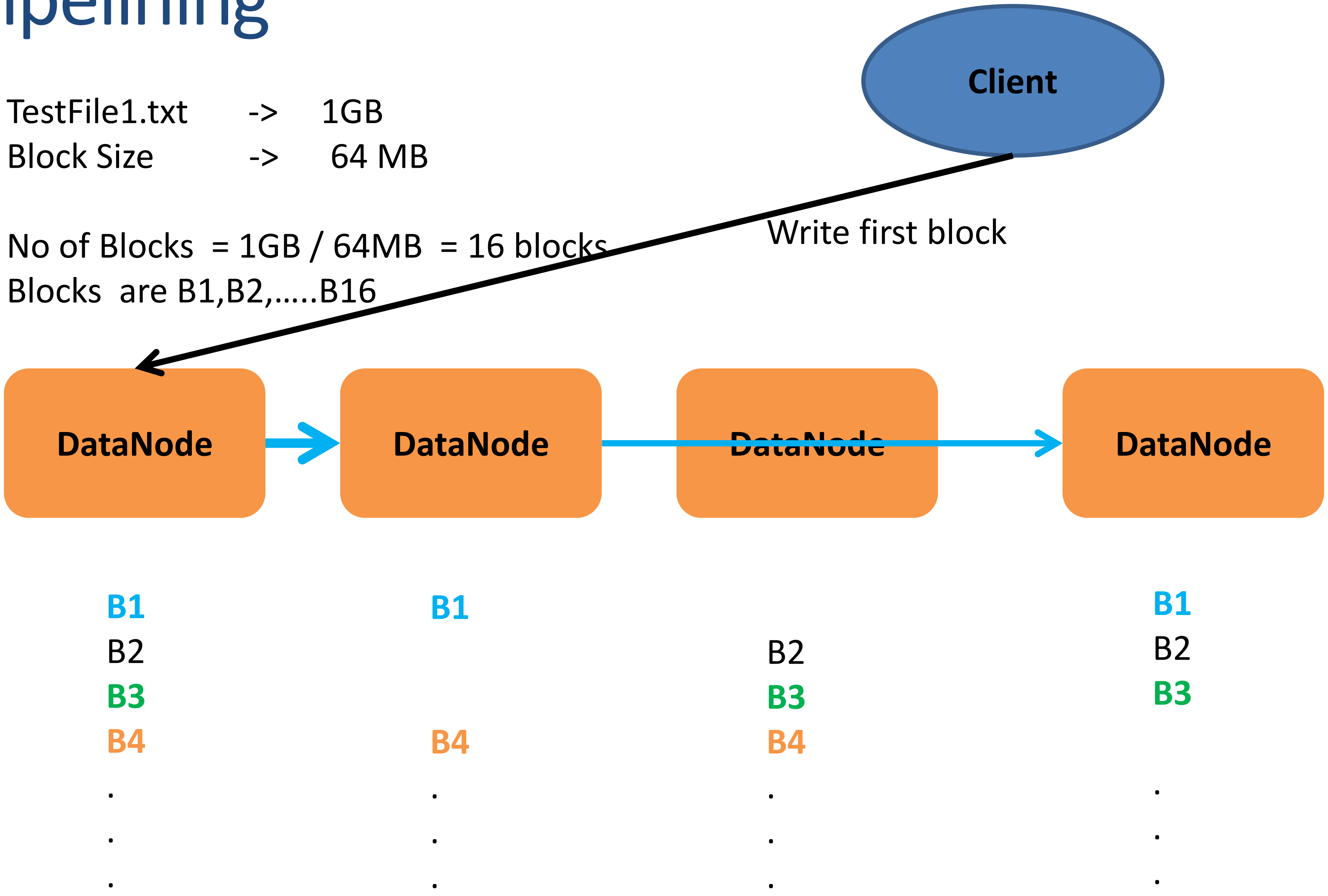
No of Blocks = 1GB / 64MB = 16 blocks
Blocks are B1,B2,…..B16

DataNode	DataNode	DataNode	DataNode
B1	B1		B1
B2		B2	B2
B3		B3	B3
B4	B4	B4	
.	.	.	.
.	.	.	.
.	.	.	.

Data Pipelining

TestFile1.txt -> 1GB
Block Size -> 64 MB

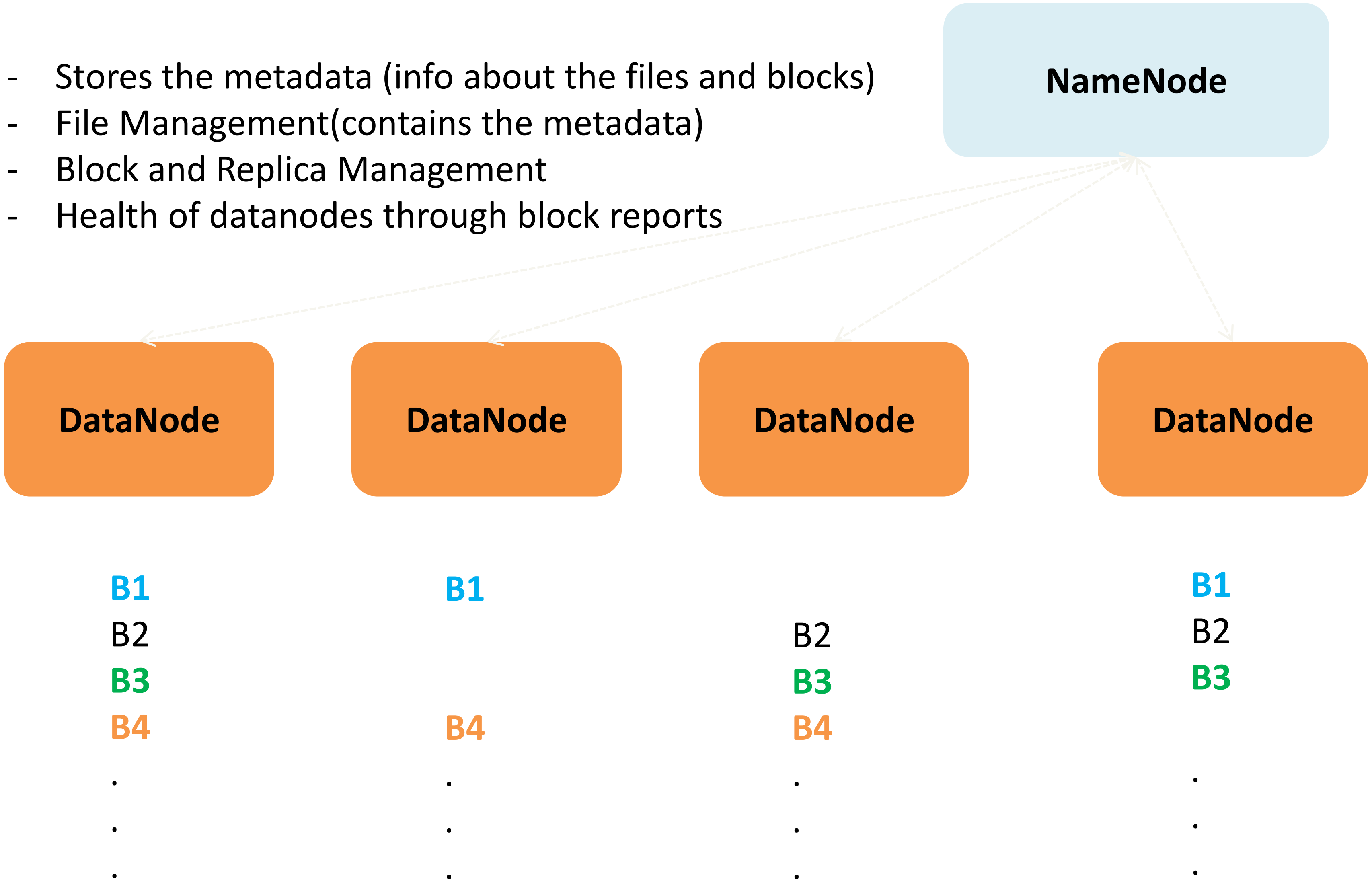
No of Blocks = 1GB / 64MB = 16 blocks
Blocks are B1,B2,…..B16



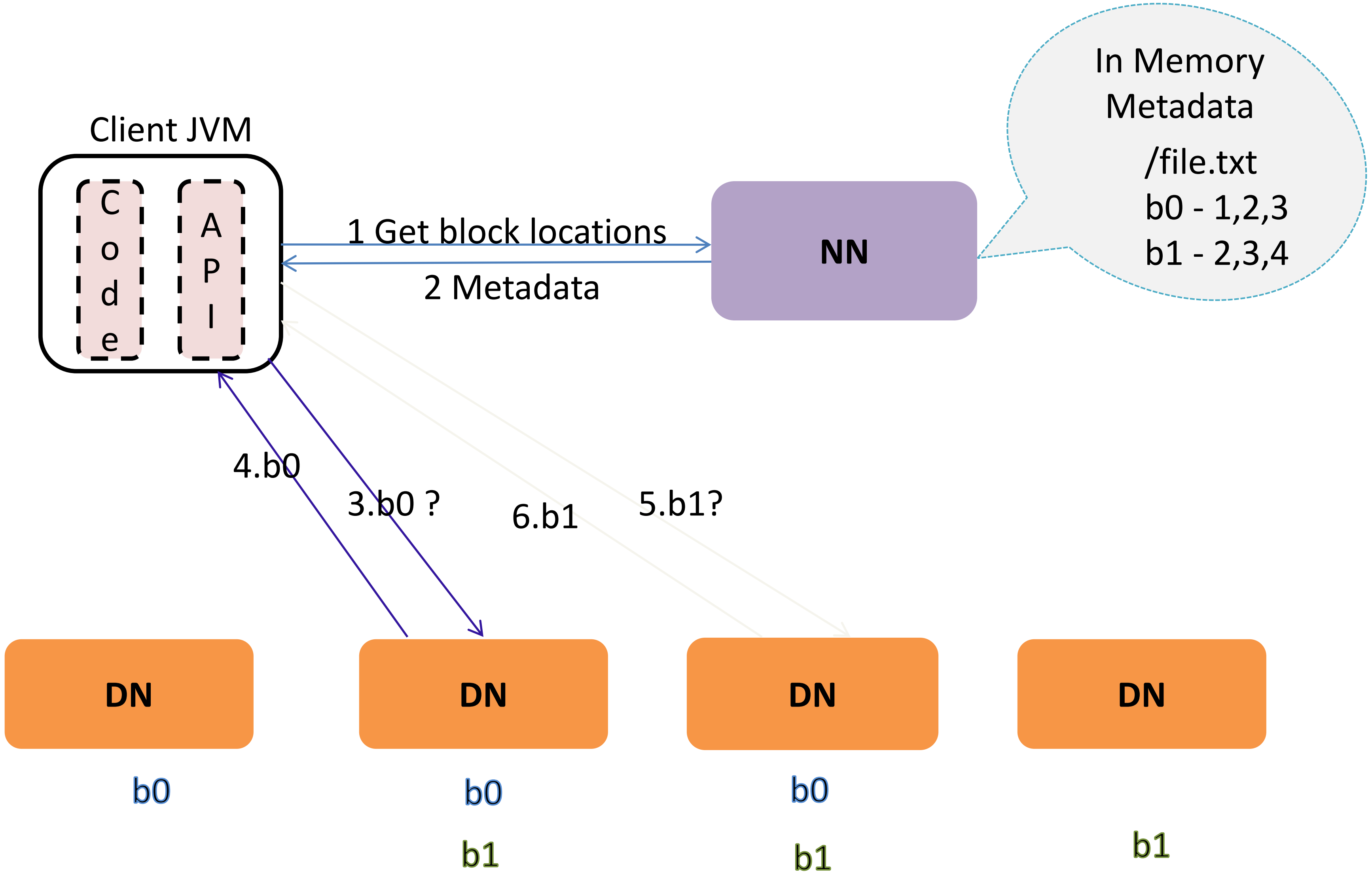
DataNode	DataNode	DataNode	DataNode
B1	B1		B1
B2		B2	B2
B3		B3	B3
B4	B4	B4	
.	.	.	.
.	.	.	.
.	.	.	.

Role of NameNode

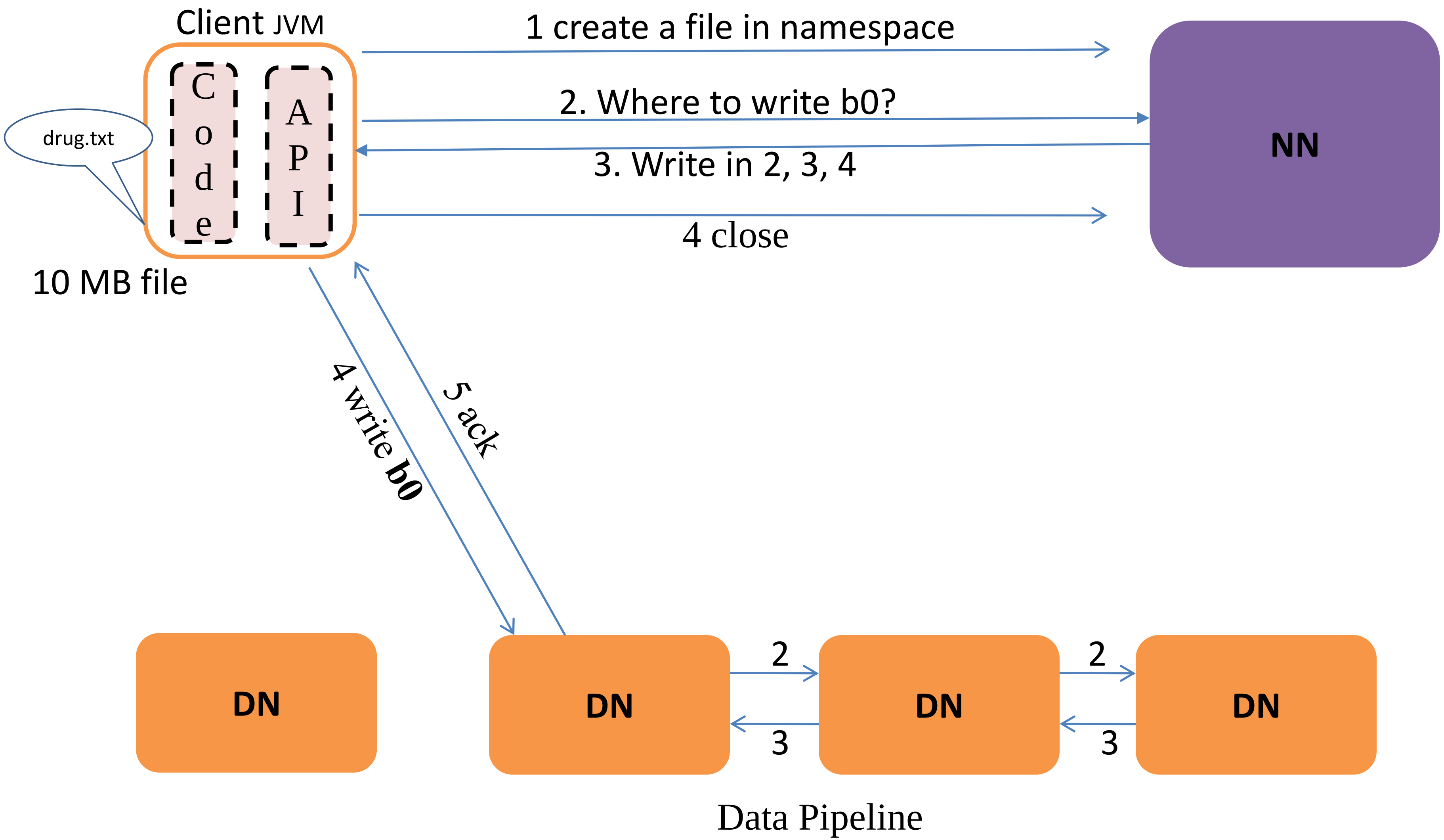
- Stores the metadata (info about the files and blocks)
- File Management(contains the metadata)
- Block and Replica Management
- Health of datanodes through block reports



File Read - Flow



File Write - Flow



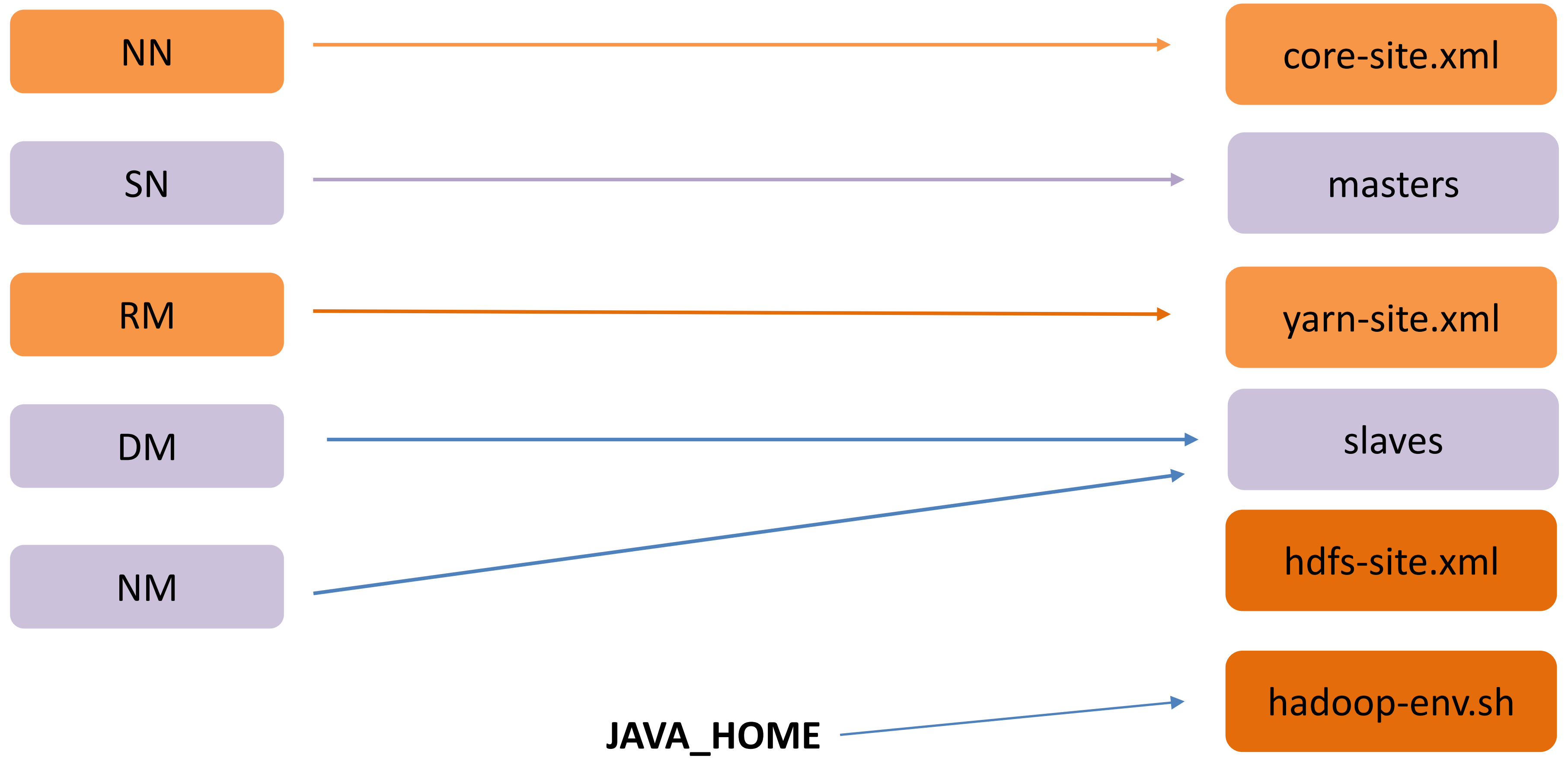
Coherency in HDFS

- Write once, Read many times
- Completed blocks will be visible to users.
- Current block being in written will not be visible to users

Execution Modes &Installation

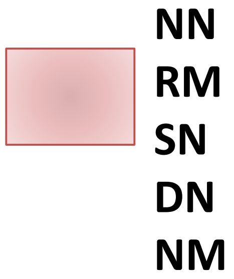
- Modes
  - Single Stand Alone
    - All Process runs in a single jvm
    - Does not use HDFS
  - Pseudo Distributed Mode **- for our training**
    - All daemon process runs in separate jvm in a single local machine
    - Used for development and testing
    - Uses HDFS to store data
  - Distributed Mode
    - A cluster of nodes more than 1
    - Each Process may run in different nodes
- Please follow instructor and doc provided

Minimal Configuration (pseudo distributed cluster)



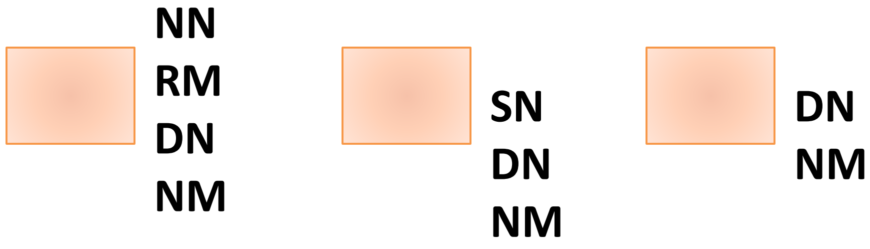
SINGE NODE CLUSTER

Pseudo Distributed For Learning purpose



NN
RM
SN
DN
NM

3 NODES CLUSTER

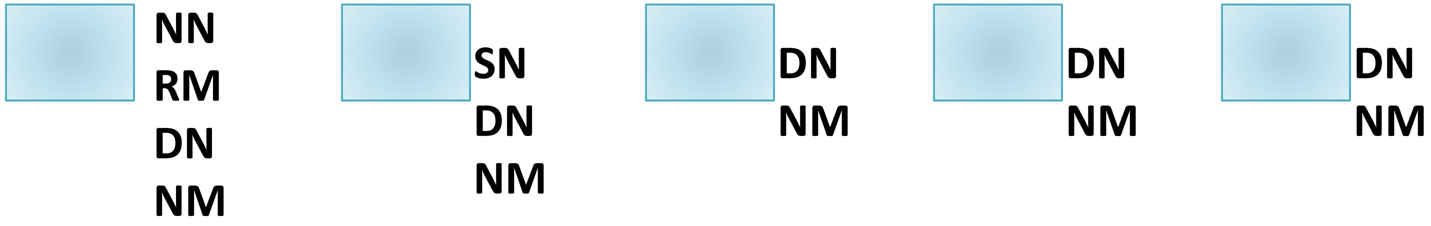


NN
RM
DN
NM

SN
DN
NM

DN
NM

5 NODES CLUSTER



NN
RM
DN
NM

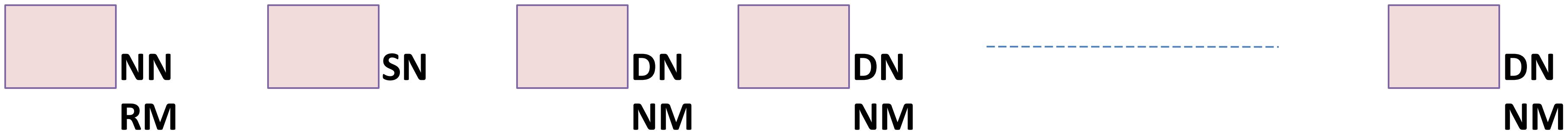
SN
DN
NM

DN
NM

DN
NM

DN
NM

20 NODES + CLUSTERS



NN
RM

SN

DN
NM

DN
NM

DN
NM

Ports used by Hadoop Daemons

Daemon	RPC	WEB
NameNode	8020 (50000*)	50070
SecondaryNameNode		50090
Resourcemanager	8032	8088
Node Manager	8040 , 8041	8042
DataNode	50010	50075
Mapreduce Job History Server		19888

After installation

- jps
  - JPS - JVM profiling status tool
- Web UI
  - NameNode - http://localhost:50070
  - Resource Manager – http://localhost:8088

Accessing HDFS

- Command line
  - Usage: *hadoop dfs <command>*
- JAVA API
- webHDFS

HDFS commands

- *hadoop dfs -copyFromLocal <srcLOCALfile> <destHDFSfile>*
- *hadoop dfs -ls /*
- *hadoop dfs -cat /<destHDFSfile>*
- *hadoop dfs -copyToLocal <srcHDFSfile> <destLOCALfile>*
- *hadoop dfs -mkdir /test*
- *hadoop dfs –rmr /test*

- *Please follow the document given*

JAVA API

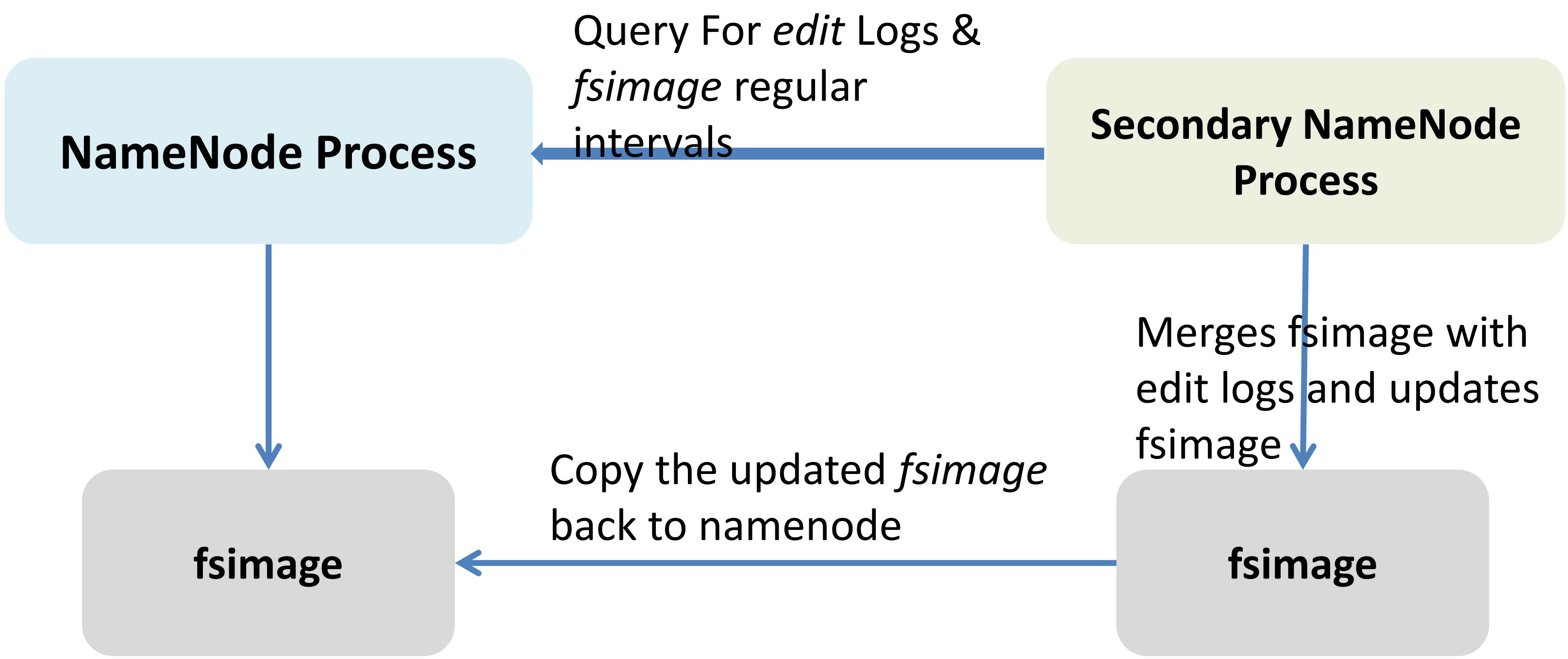
- Most Packages Used
  - *org.apache.hadoop.conf.Configuration*
  - *org.apache.hadoop.fs.BlockLocation*
  - *org.apache.hadoop.fs.FSDataInputStream*
  - *org.apache.hadoop.fs.FSDataOutputStream*
  - *org.apache.hadoop.fs.FileStatus*
  - *org.apache.hadoop.fs.FileSystem*
  - *org.apache.hadoop.fs.Path*
  - *org.apache.hadoop.hdfs.DistributedFileSystem*
  - *org.apache.hadoop.hdfs.protocol.DatanodeInfo*

Please see and execute the example code provided

FileSystem API methods

- append()
- copyFromLocalFile()
- create()
- delete()
- mkdirs()
- open()

Role of Secondary NameNode



NameNode Process

Query For *edit* Logs & *fsimage* regular intervals

Secondary NameNode Process

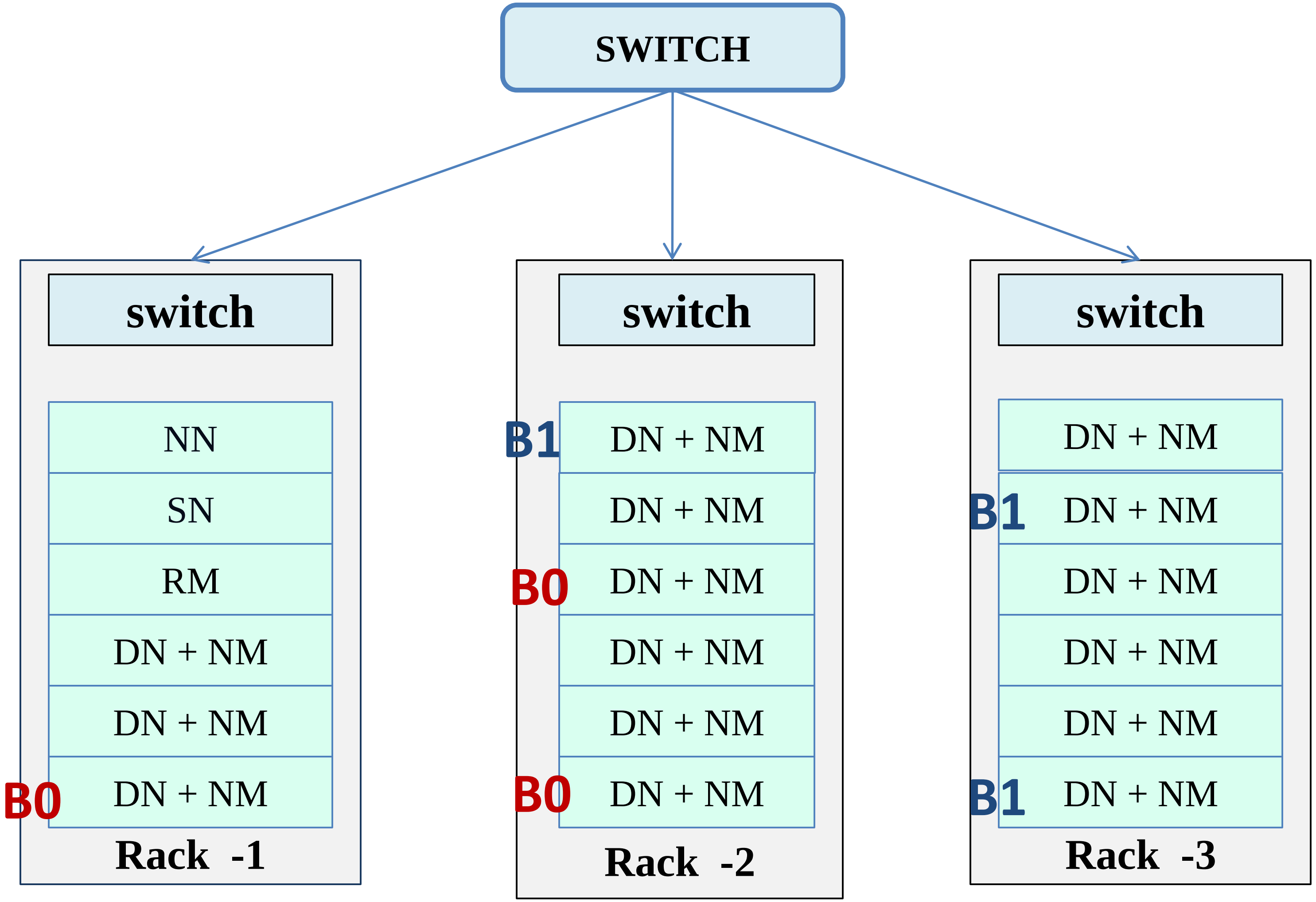
Merges fsimage with edit logs and updates fsimage

fsimage

Copy the updated *fsimage* back to namenode

fsimage

Rack AwareNess Policy



SWITCH

switch

NN

SN

RM

DN + NM

DN + NM

B0 DN + NM

Rack -1

switch

B1 DN + NM

DN + NM

B0 DN + NM

DN + NM

DN + NM

B0 DN + NM

Rack -2

switch

DN + NM

B1 DN + NM

DN + NM

DN + NM

DN + NM

B1 DN + NM

Rack -3