HDFS

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

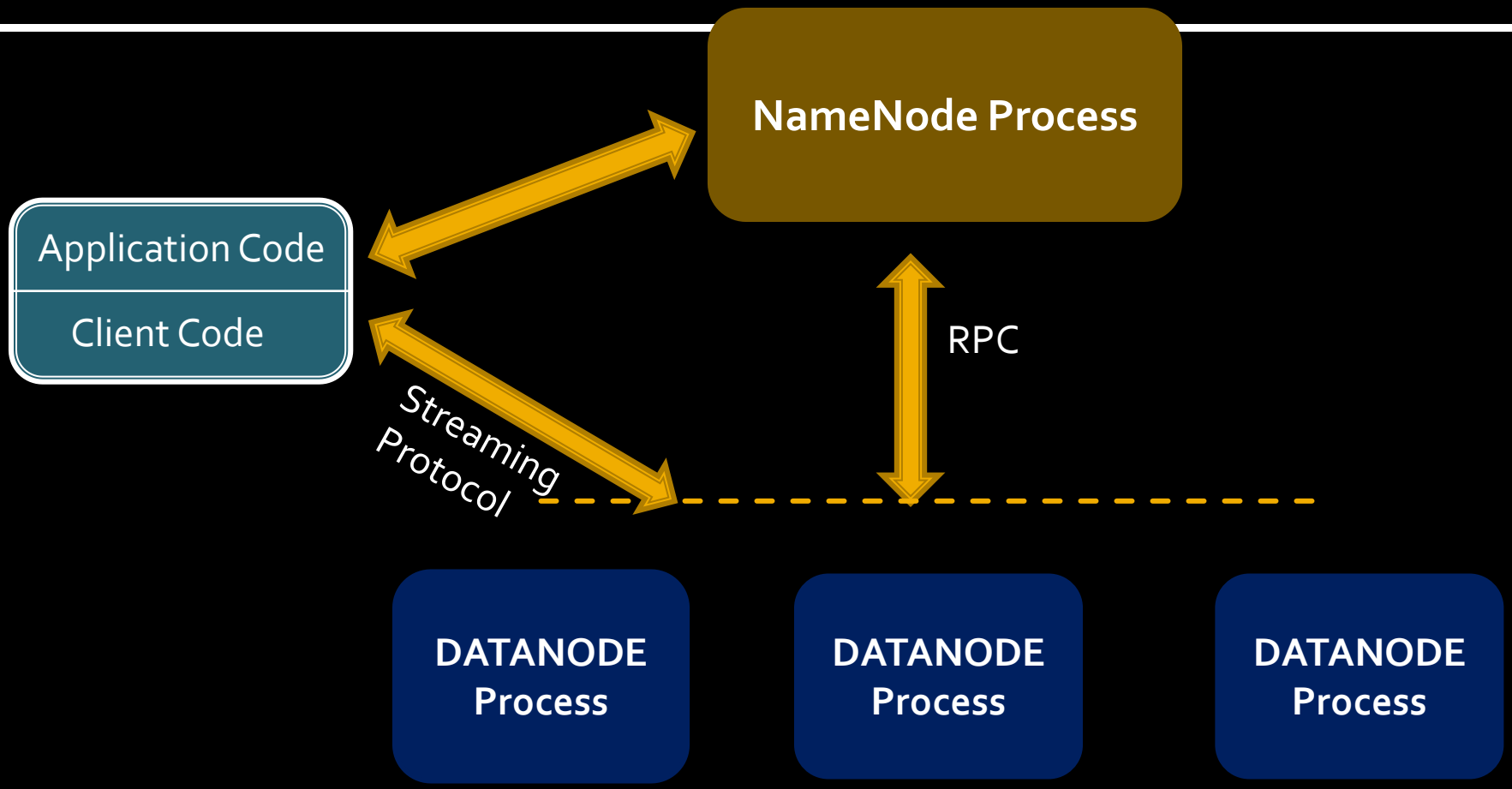
- Client-server Concept
  - Client sends requests to one or more servers which in turn accepts, processes them and return the requested information to the client.
  - A server might run a software which listens on particular ip and port number for requests
  - Examples:
    - Server - web server
   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

Introduction

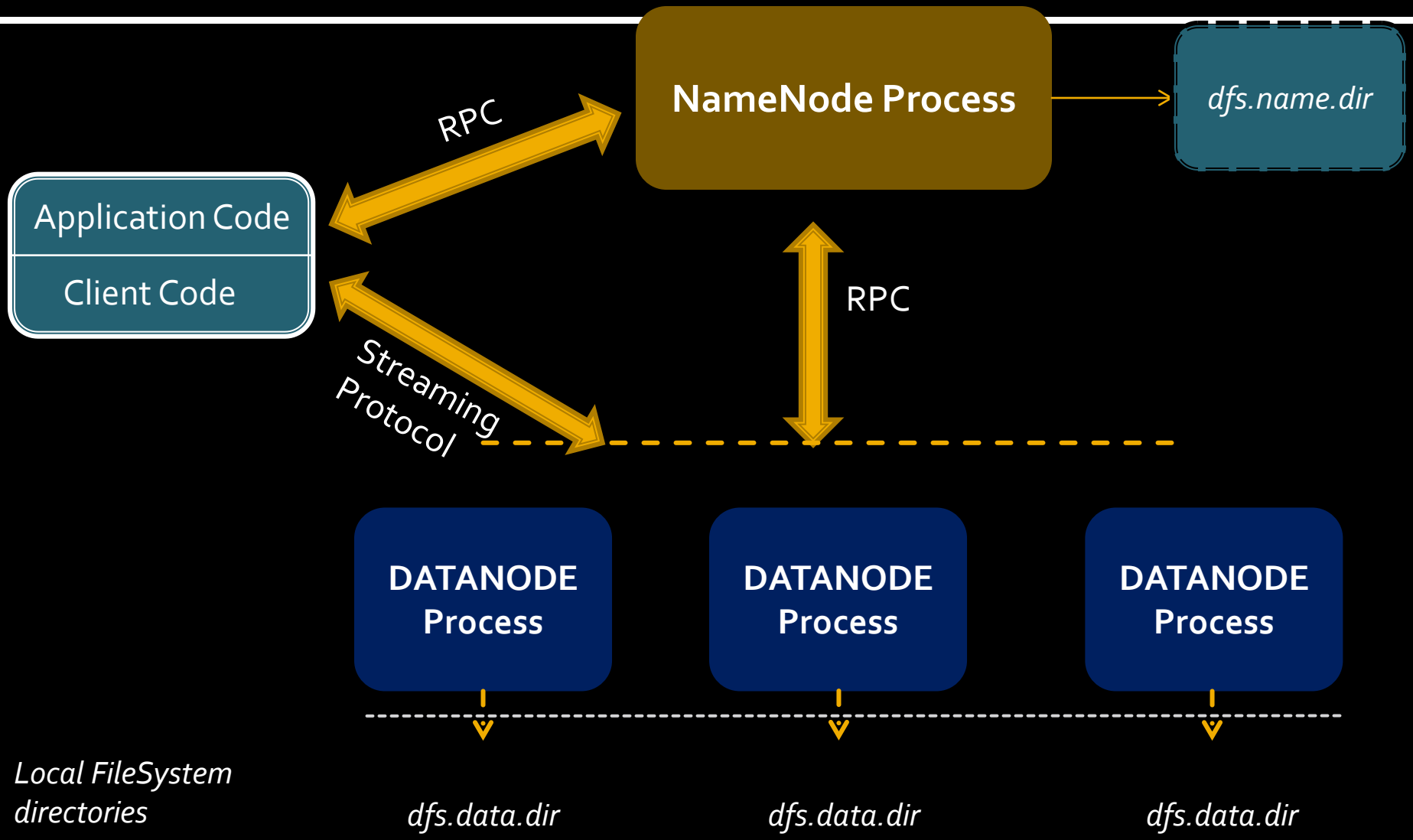
- A distributed File System - **STORAGE**
  - A File System on multiple machines which sits on native filesystem
    - ext4,ext3
- Hardware Failure
  - Due to usage of Commodity machines, failure is a common phenomenon
  - Designed for failure
- Large Data Sets
  - Small Files Problem Due to NameNode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

- ext3? ext4?

Continued…


NameNode Process
Application Code
Client Code
RPC
Streaming Protocol
DATANODE Process
DATANODE Process
DATANODE Process

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*
- JobTracker*
- TaskTracker*

HDFS

MR

* - will be seen later in class

Block Concept

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

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

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

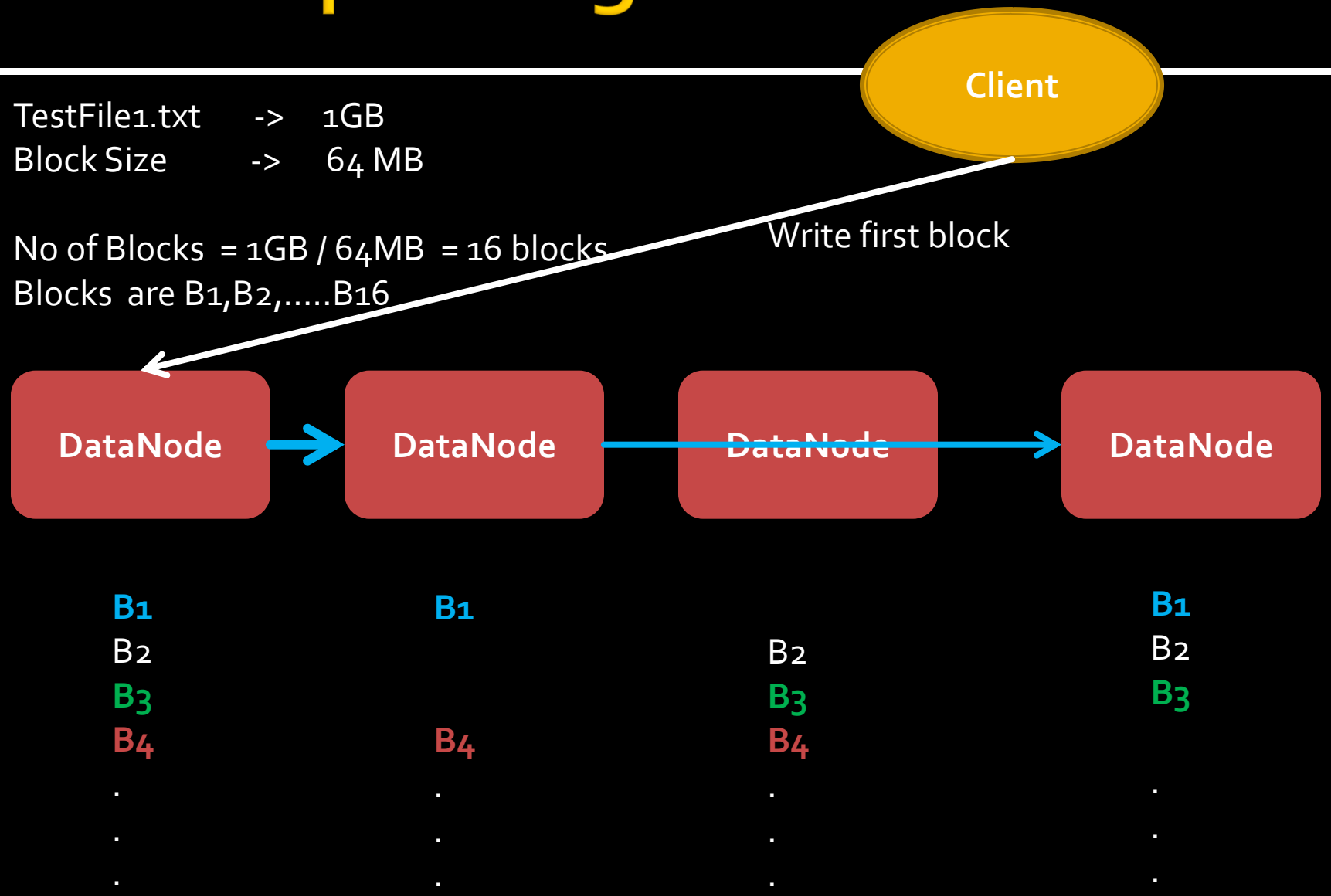
No of Blocks = 1GB / 64MB = 16 blocks
Blocks are B1,B2,.....B16

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

Data Pipelining

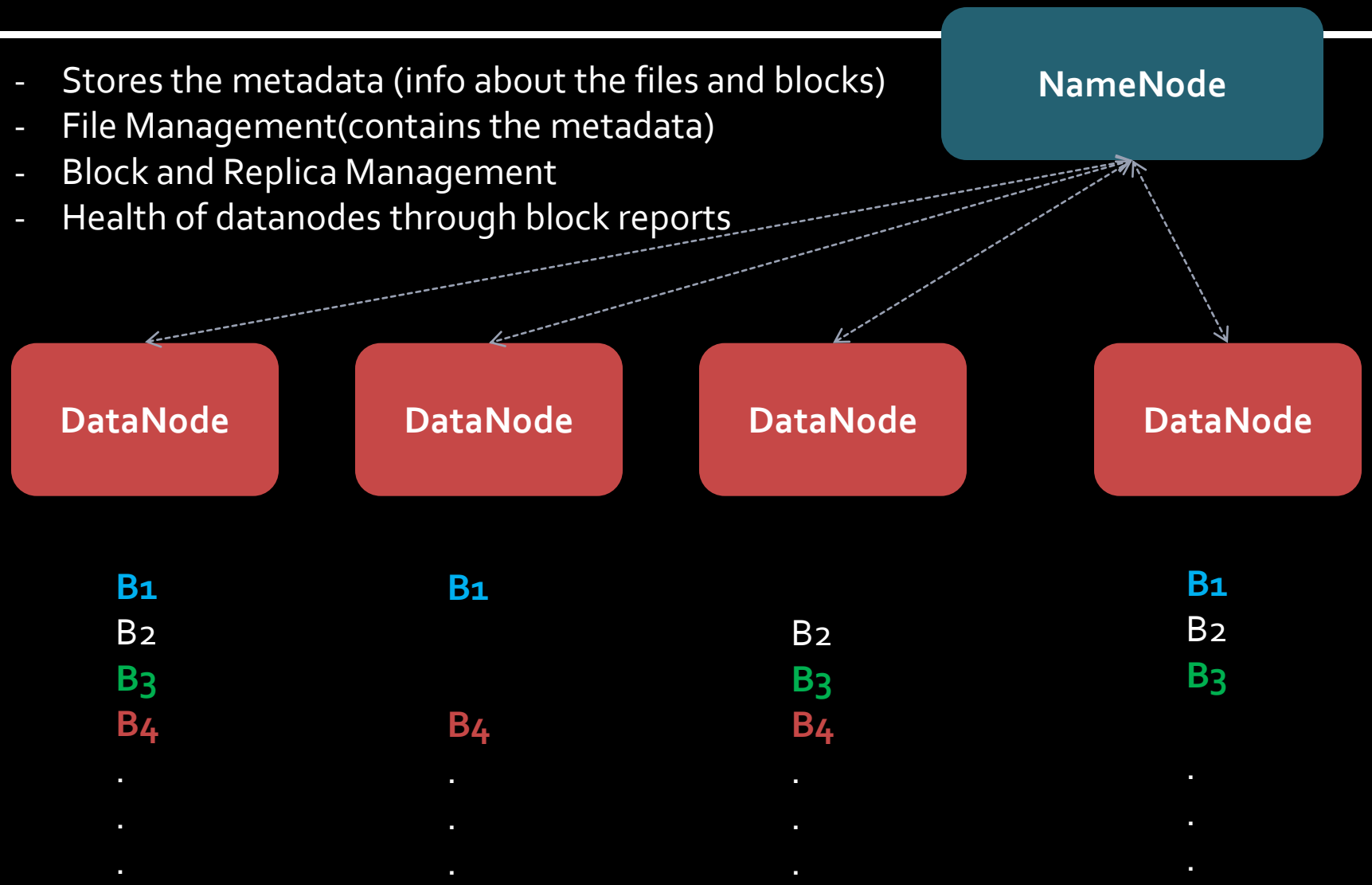
TestFile1.txt -> 1GB
Block Size -> 64 MB

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

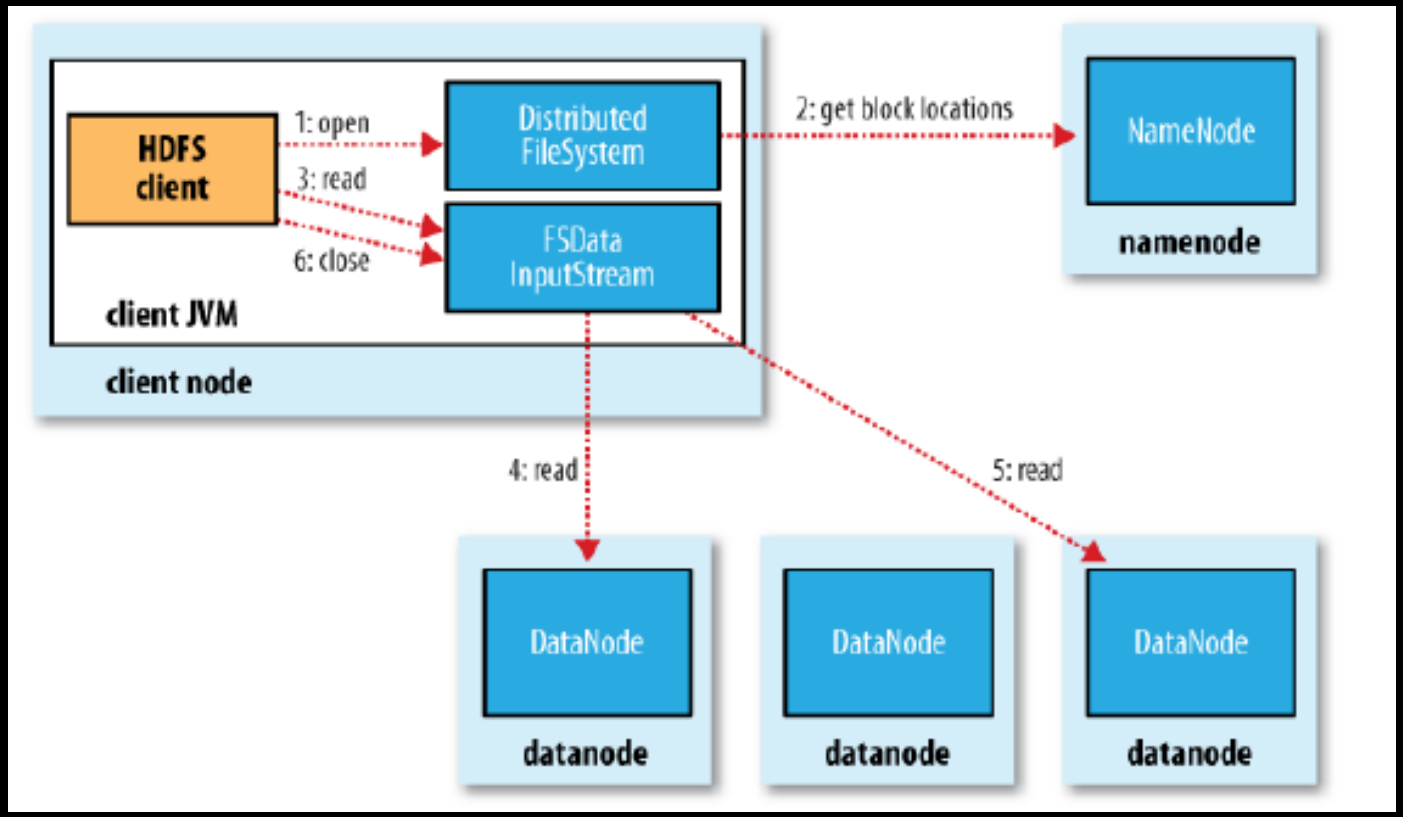


Role of NameNode

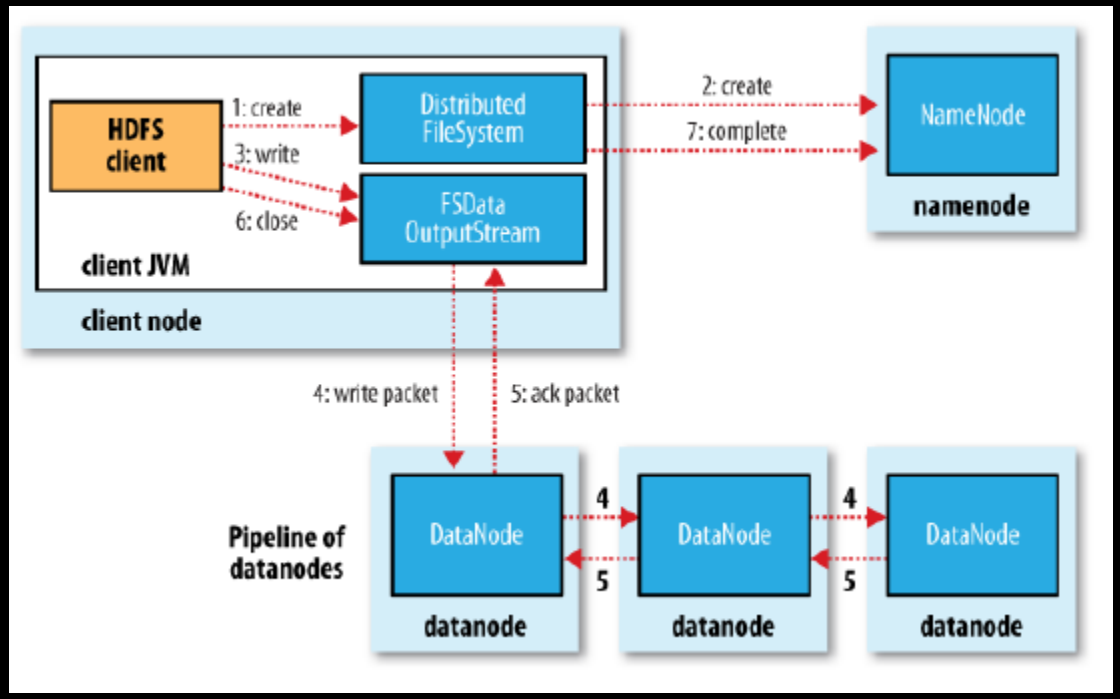
- Stores the metadata (info about the files and blocks)
- File Management(contains the metadata)
- Block and Replica Management
- Health of datanodes through block reports



File Read - Flow



File Write - Flow



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

Installation

- Please follow the steps in the document given

After installation

- jps
  - Jps - jvm profiling status tool
- Web UI
  - NameNode - http://localhost:50070
  - JobTracker – http://localhost:50030

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*

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

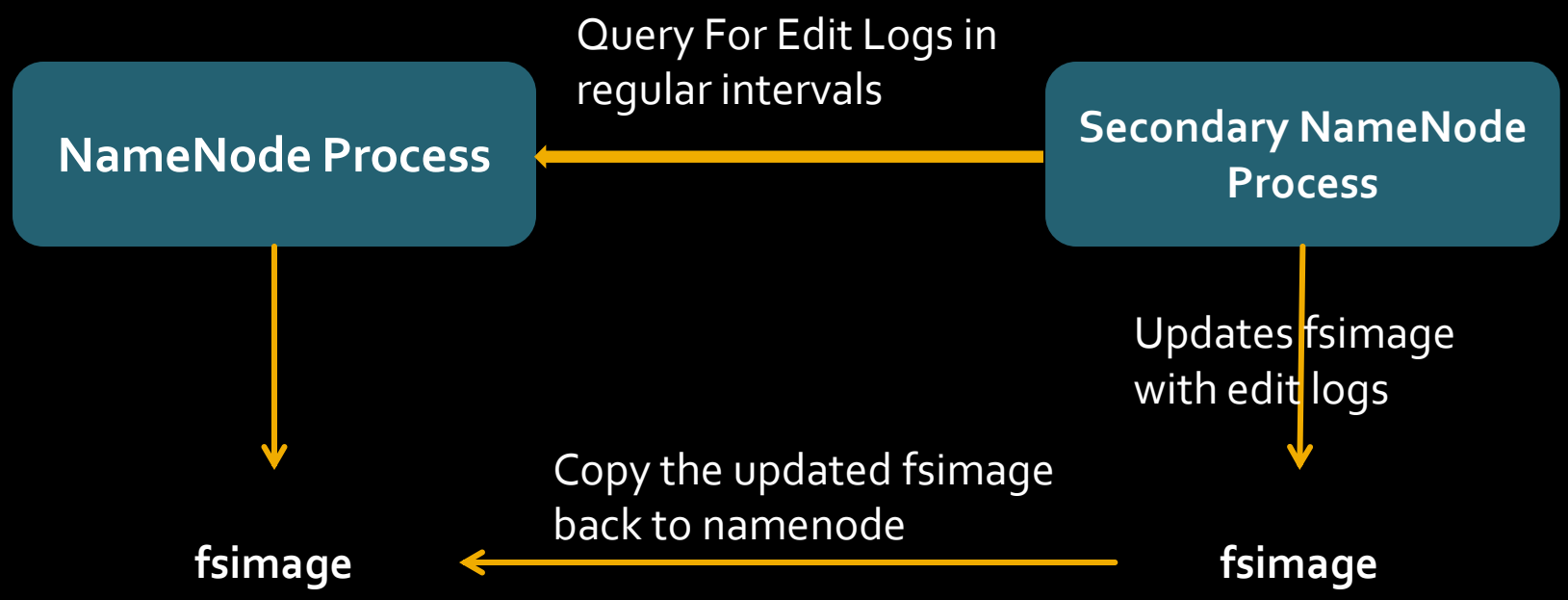
FileSystem API methods

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

Secondary NameNode

- A helper node for NameNode
- Performs memory-intensive administrative functions for the NameNode
- Have a checkpoint for the file system (HDFS)
- Not a Backup Node

Role of Secondary NameNode



THANK YOU!!