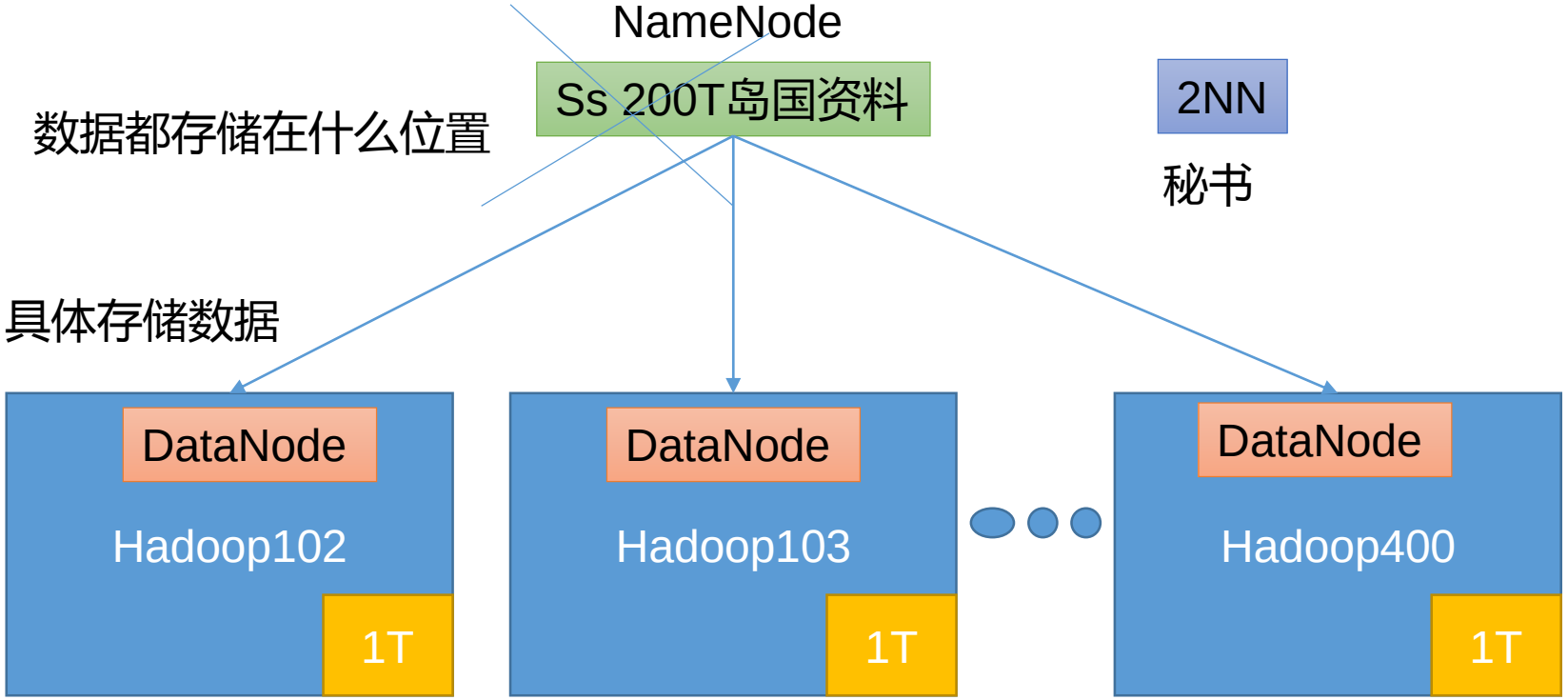
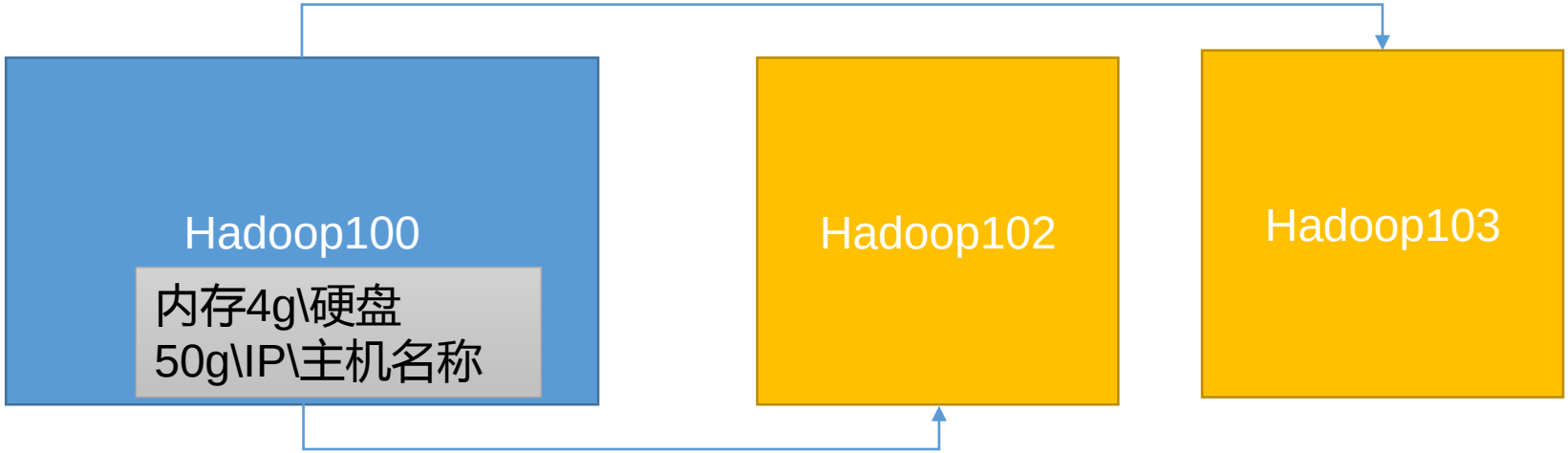
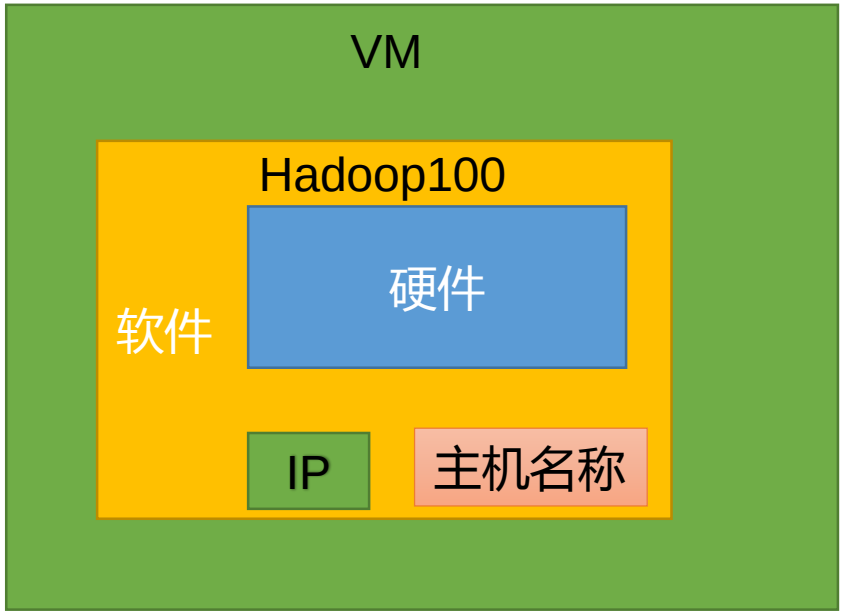
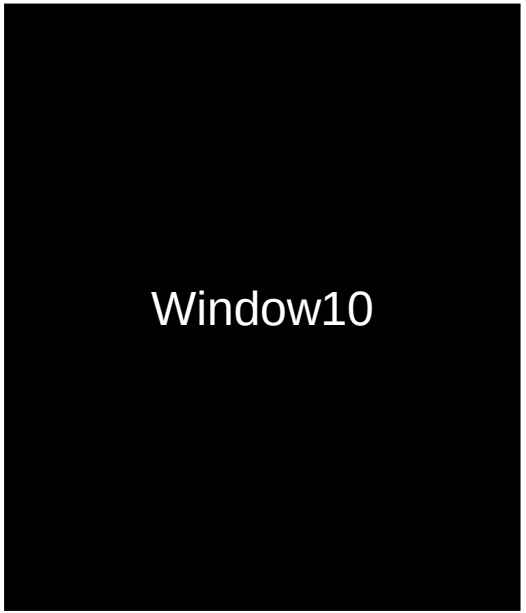

NameNode
Ss 200T岛国资料
2NN
秘书
数据都存储在什么位置
具体存储数据
DataNode
Hadoop102
1T
DataNode
Hadoop103
1T
DataNode
Hadoop400
1T


Hadoop100
内存4g\硬盘50g\IP\主机名称
Hadoop102
Hadoop103
操作系统/其他配置





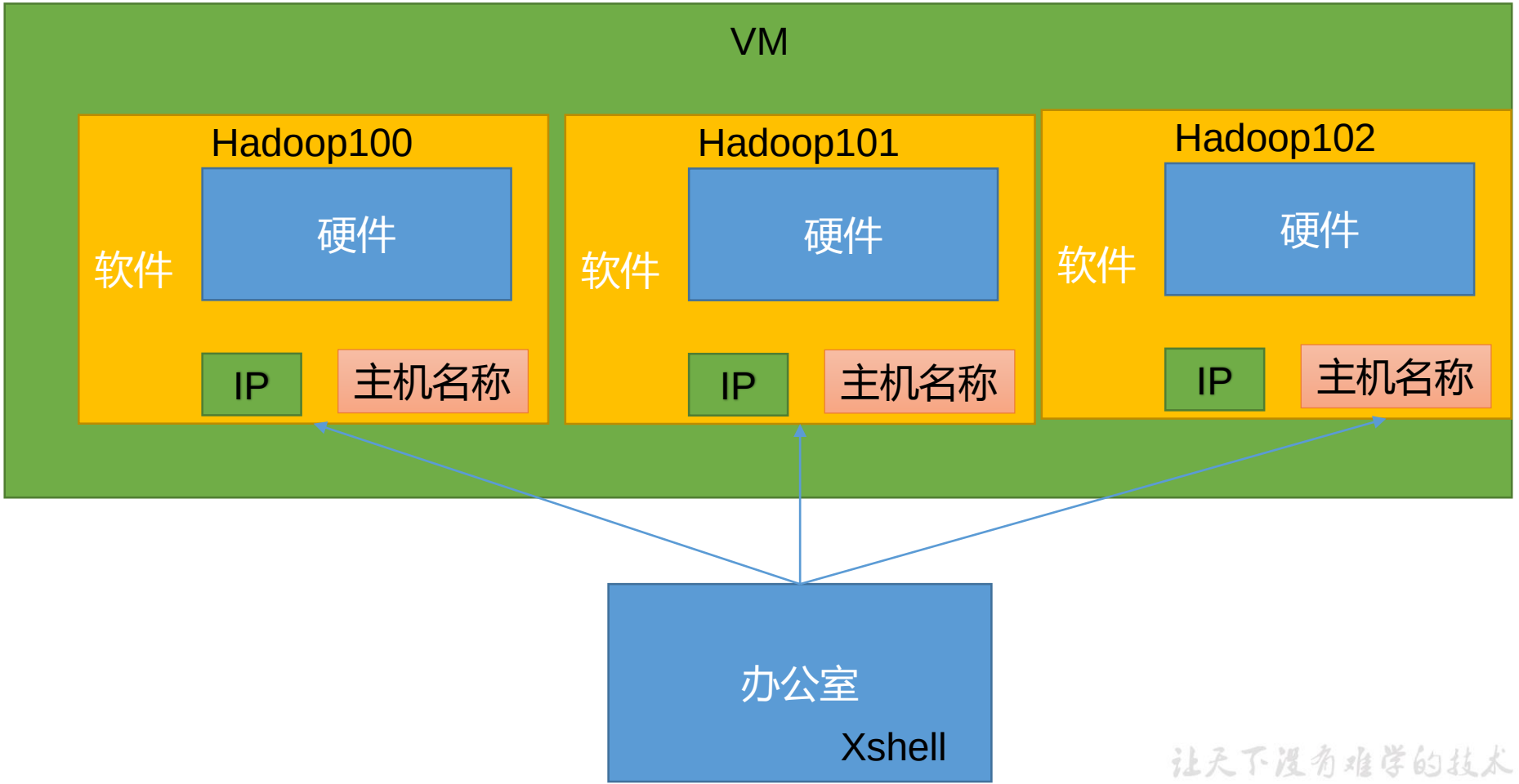
192.168.10.105 hadoop100

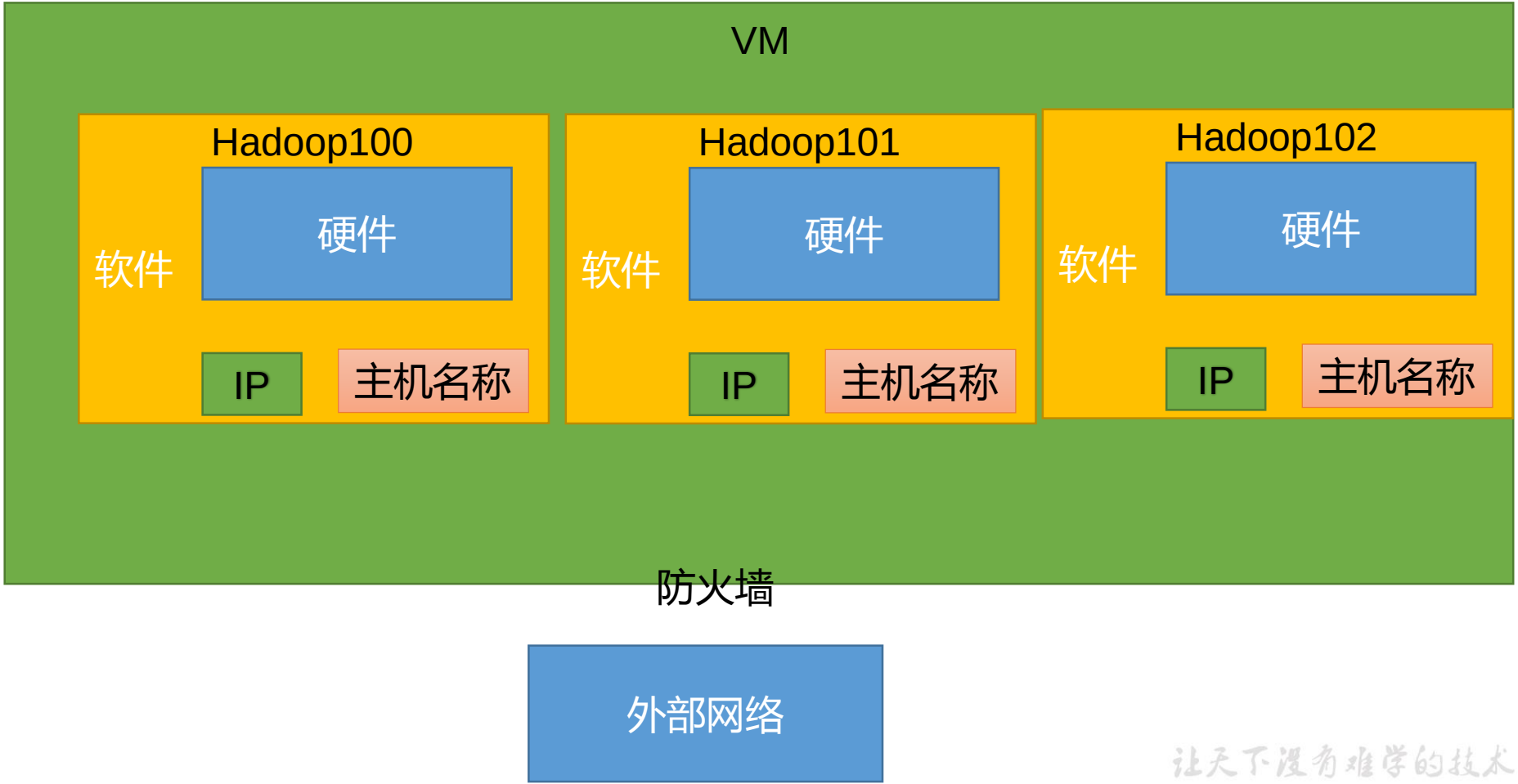
hadoop100

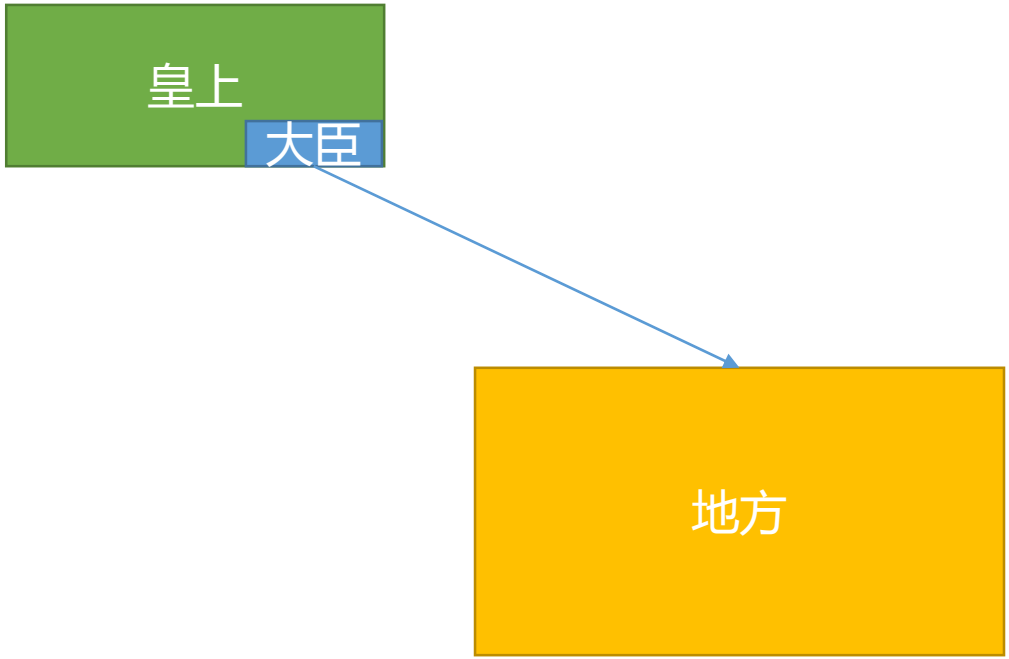
hadoop100

hadoop100

hadoop100


VM
Hadoop100
软件
硬件
IP
主机名称
Hadoop101
软件
硬件
IP
主机名称
Hadoop102
软件
硬件
IP
主机名称
办公室
Xshell

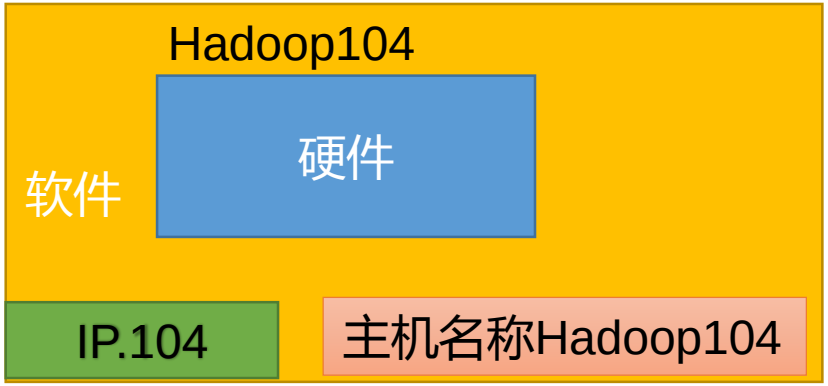

VM
Hadoop100
软件
硬件
IP
主机名称
Hadoop101
软件
硬件
IP
主机名称
Hadoop102
软件
硬件
IP
主机名称
防火墙
外部网络

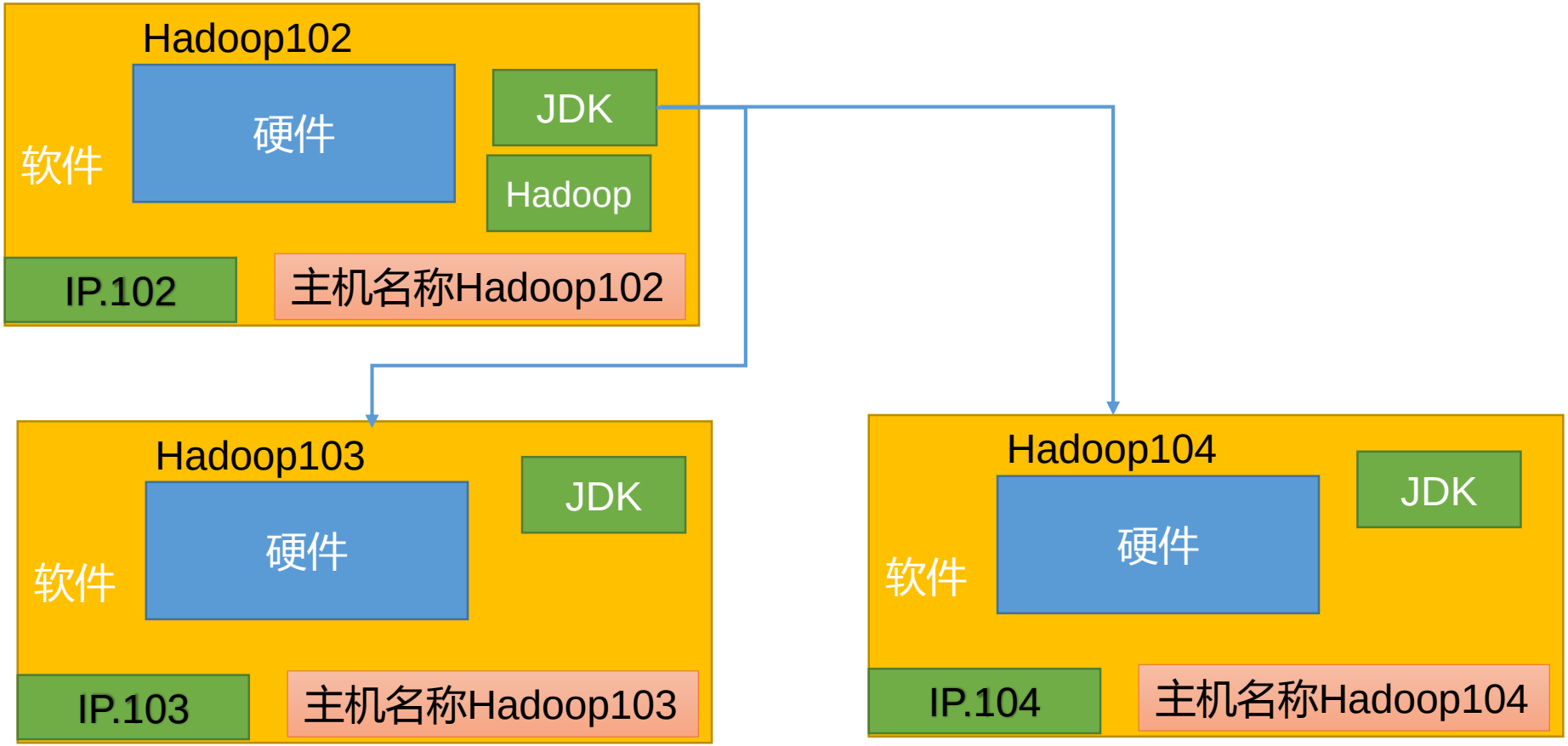


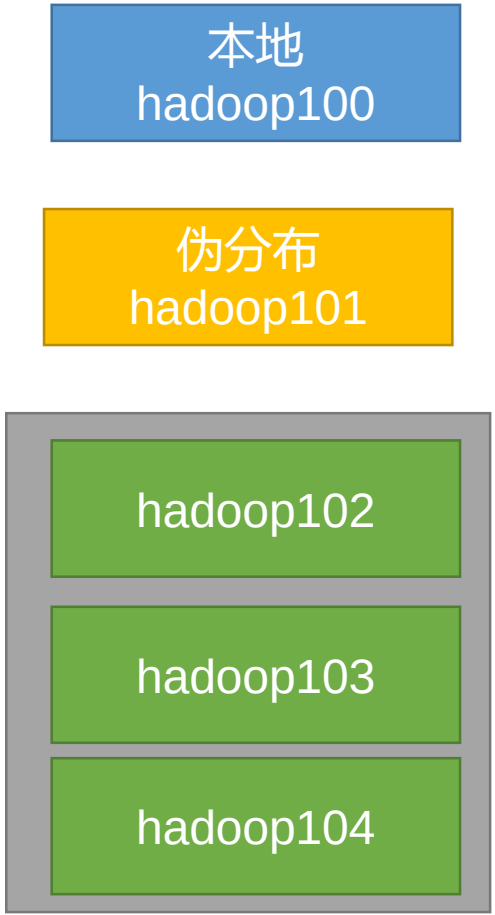

Hadoop100
硬件
软件
IP.100
主机名称Hadoop100


Hadoop102
硬件
软件
IP.102
主机名称Hadoop102


Hadoop103
硬件
软件
IP.103
主机名称Hadoop103


Hadoop104
硬件
软件
IP.104
主机名称Hadoop104


Hadoop102
软件
硬件
JDK
Hadoop
IP.102
主机名称Hadoop102
Hadoop103
软件
硬件
JDK
IP.103
主机名称Hadoop103
Hadoop104
软件
硬件
JDK
IP.104
主机名称Hadoop104

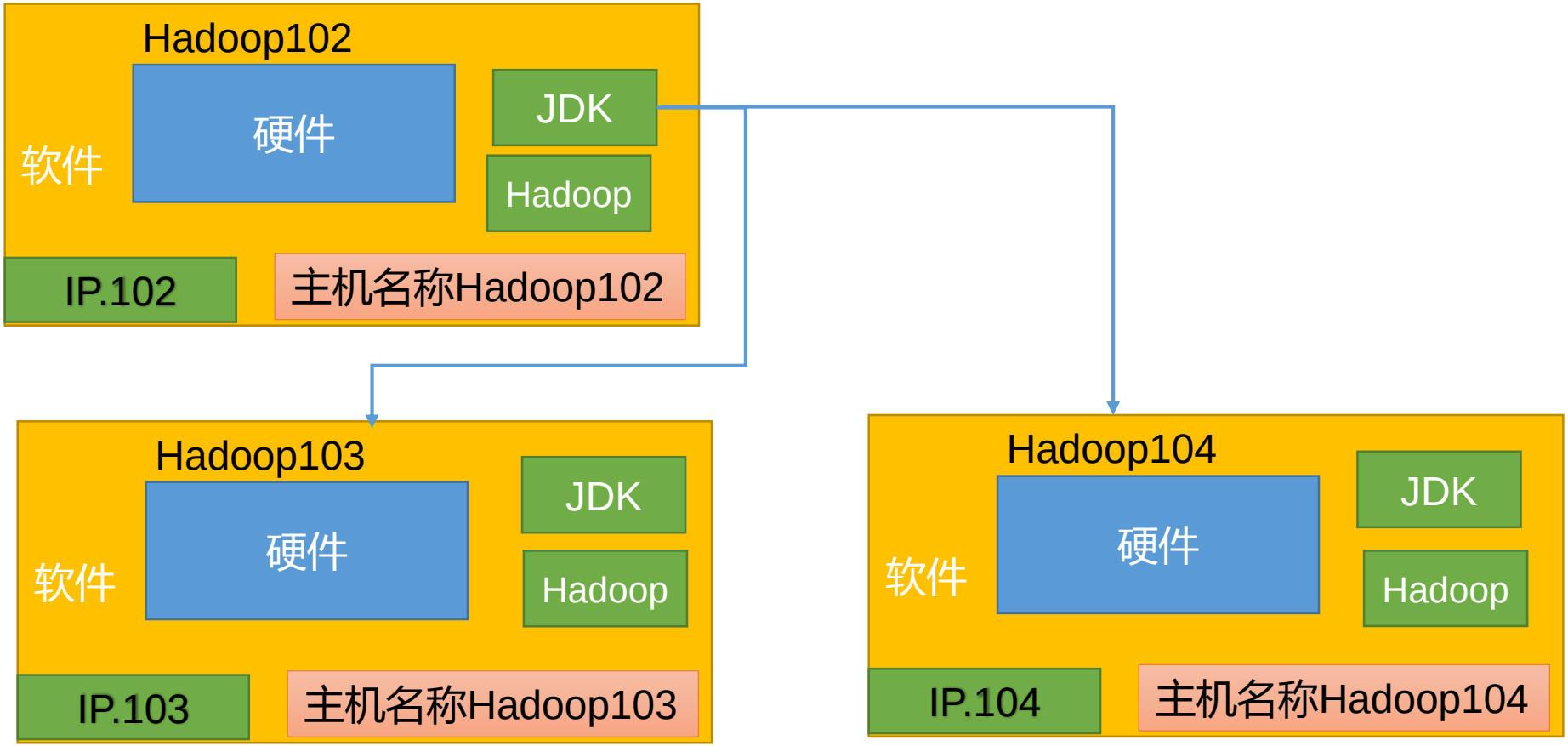


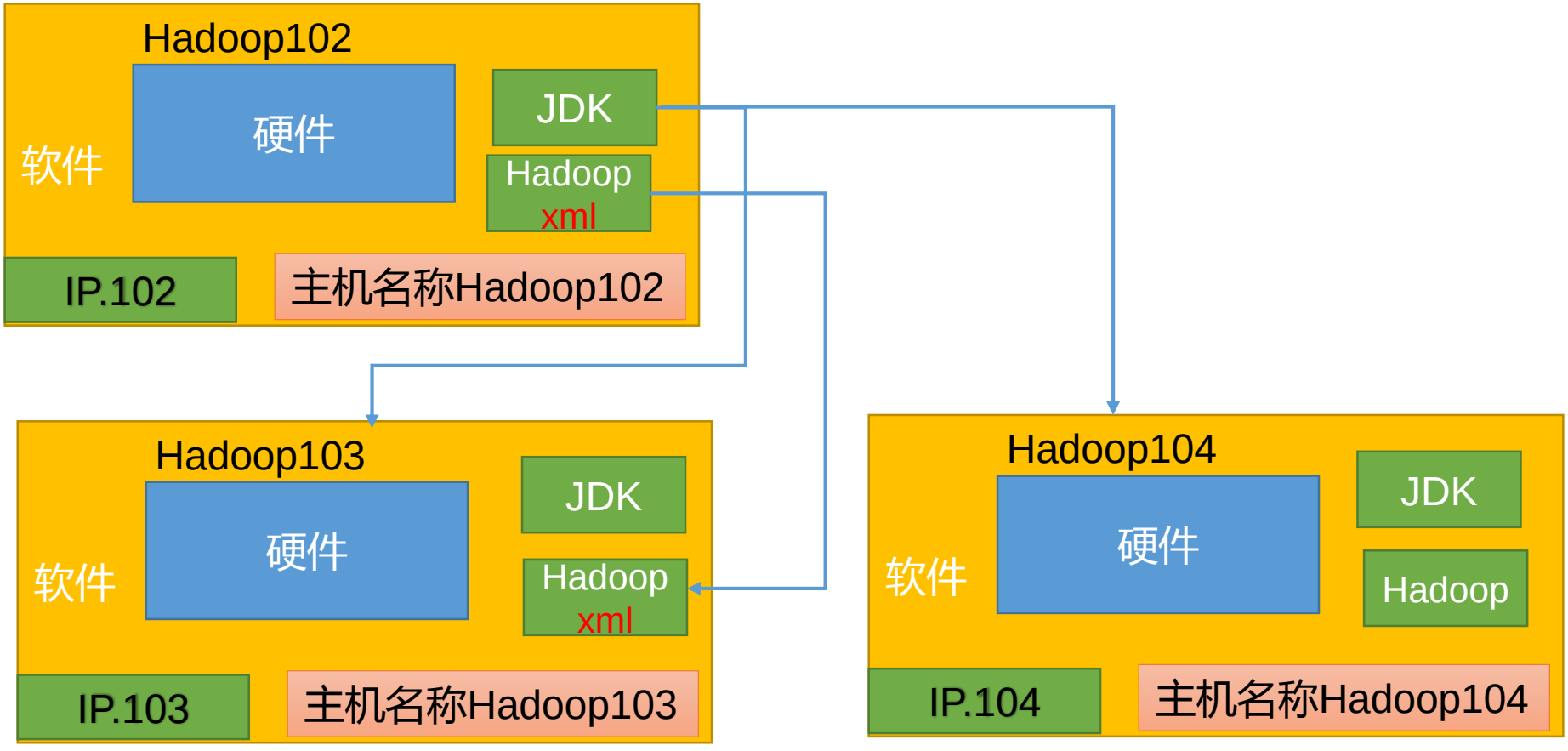
数据存储在linux本地
测试偶尔用一下

数据存储在HDFS
公司中比较差钱　2m　16g 1台 =》2m -1T

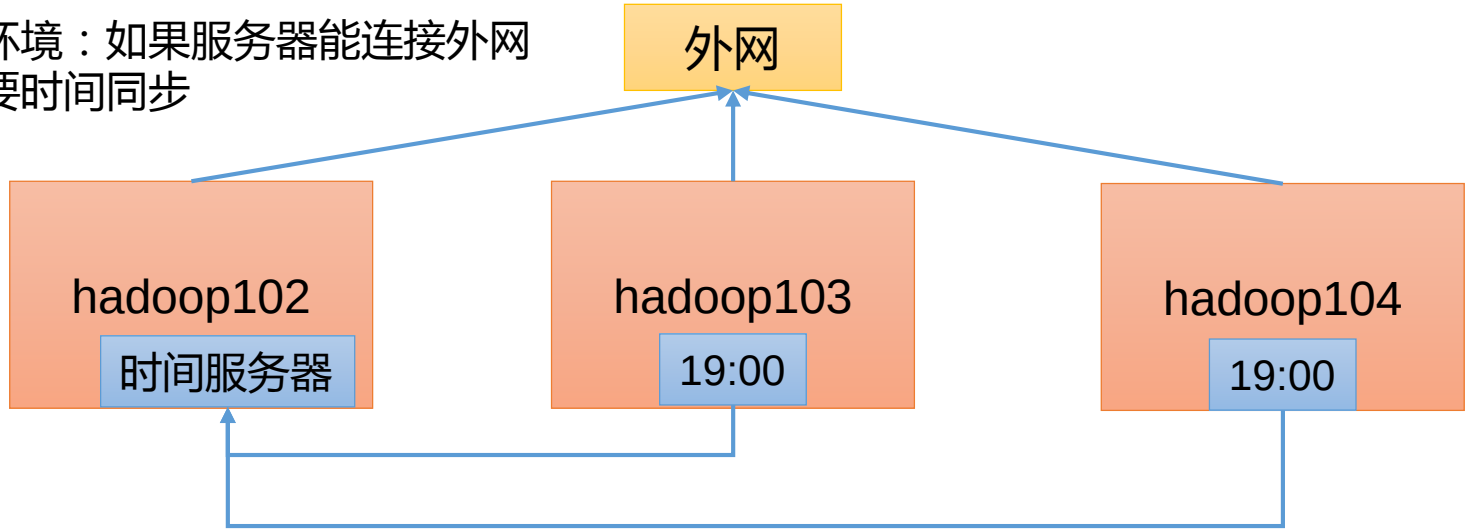
完全分布式　　数据存储在HDFS/多台服务器工作

企业里面大量使用

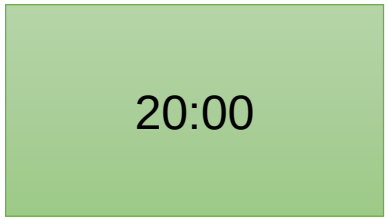

Hadoop102
硬件
软件
JDK
Hadoop
IP.102
主机名称Hadoop102
Hadoop103
硬件
软件
JDK
Hadoop
IP.103
主机名称Hadoop103
Hadoop104
硬件
软件
JDK
Hadoop
IP.104
主机名称Hadoop104


Hadoop102
软件
硬件
JDK
Hadoop
xml
IP.102
主机名称Hadoop102
Hadoop103
软件
硬件
JDK
Hadoop
xml
IP.103
主机名称Hadoop103
Hadoop104
软件
硬件
JDK
Hadoop
IP.104
主机名称Hadoop104

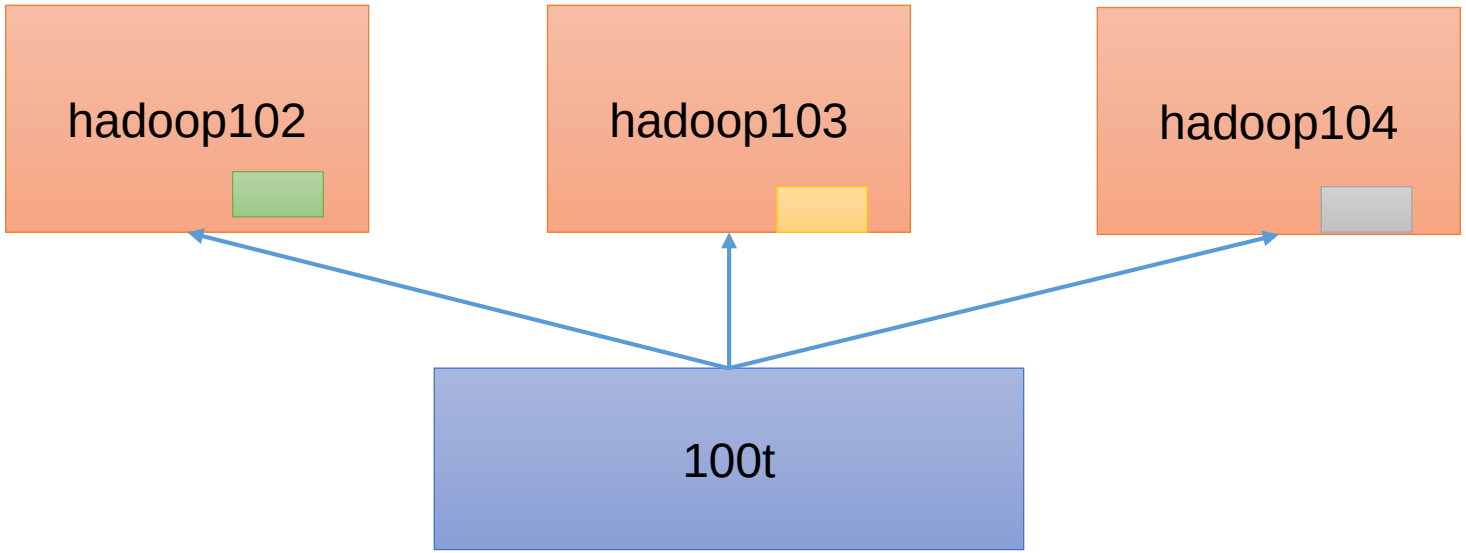
生产环境：如果服务器能连接外网
不需要时间同步

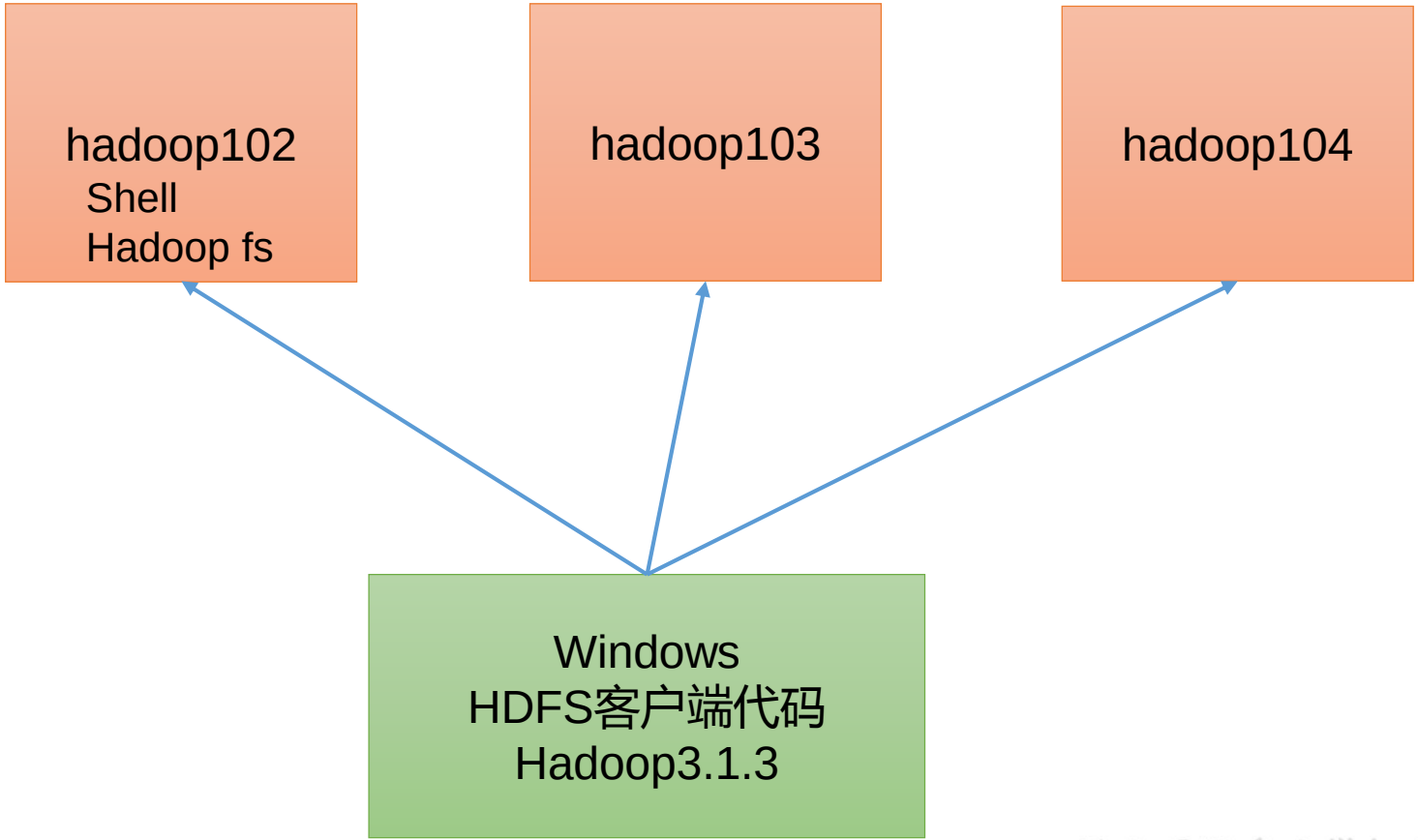


生产环境：如果服务器
能连接不了外网
需要时间同步



HDFS




hadoop102
Shell
Hadoop fs
hadoop103
hadoop104
Windows
HDFS客户端代码
Hadoop3.1.3

内存

fsImage 存储数据

Edits 追加

内存：

好处：计算快

坏处：可靠性差

磁盘：

好处：可靠性高

坏处：计算慢

内存 + 磁盘 =》效率低

fsImage 存储数据（如果是随机读写效率

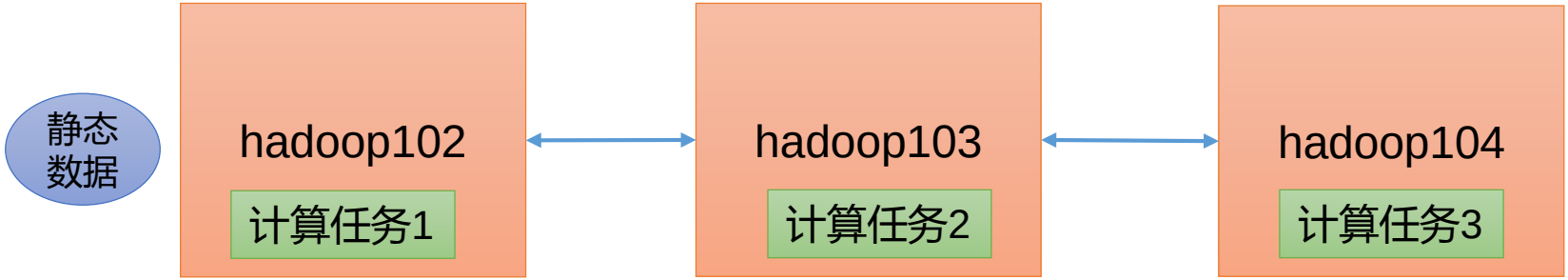
a =10 a + 10 => a =20 ）

Edits 追加 =>

a +10

a -30

a *20



MapReduce： 自己处理业务相关代码+ 自身的默认代码

优点：

1、易于编程。 用户只关心，业务逻辑。 实现框架的接口。

2、良好扩展性：可以动态增加服务器，解决计算资源不够问题

3、高容错性。任何一台机器挂掉，可以将任务转移到其他节点。

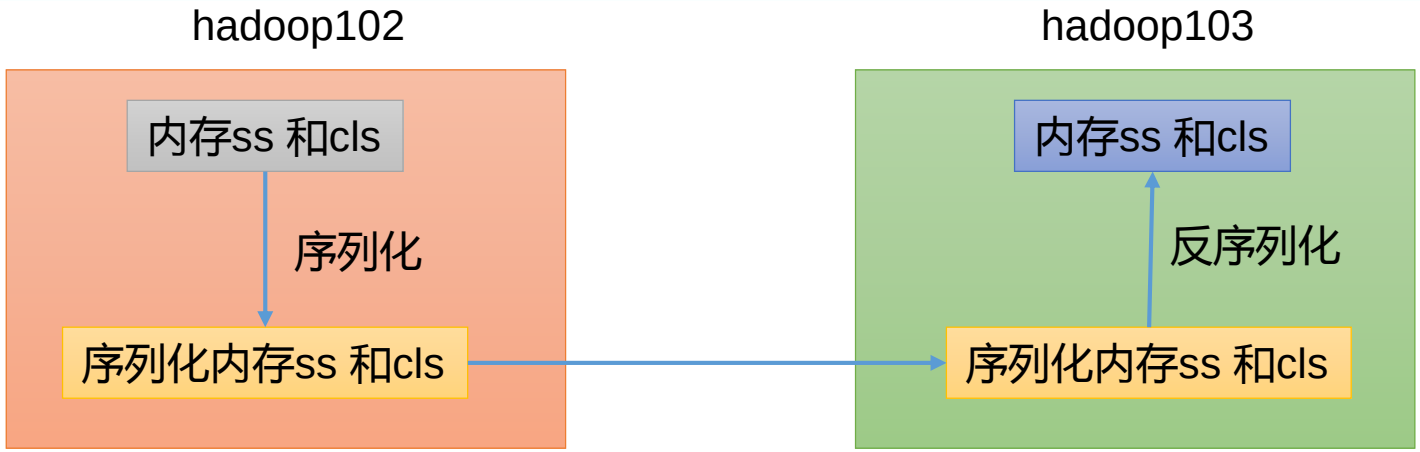
4、适合海量数据计算（TB/PB） 几千台服务器共同计算。

缺点：

1、不擅长实时计算。 Mysql

2、不擅长流式计算。 Sparkstreaming flink

3、不擅长DAG有向无环图计算。spark



Java Serializable 很重

紧凑：存储空间少
快速：传输速度块
互操作性：

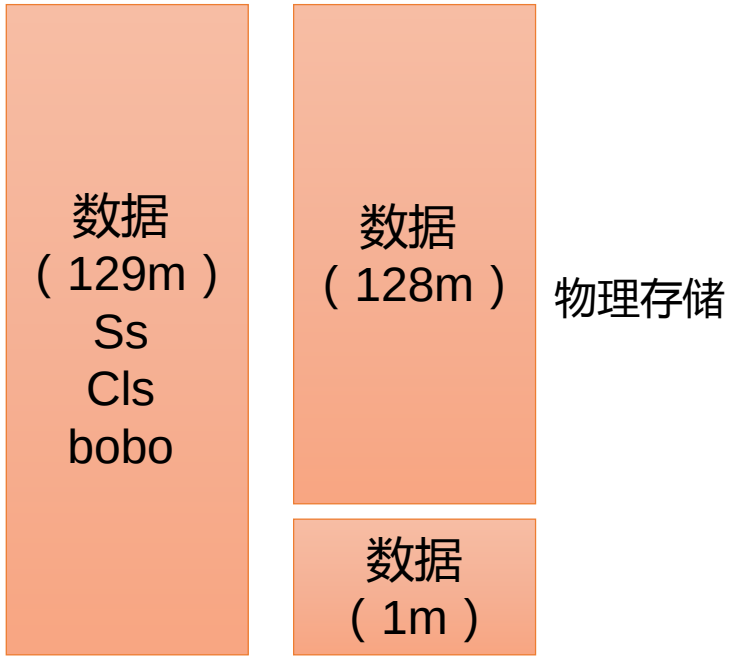
序列化

sumFlow downFlow upFlow

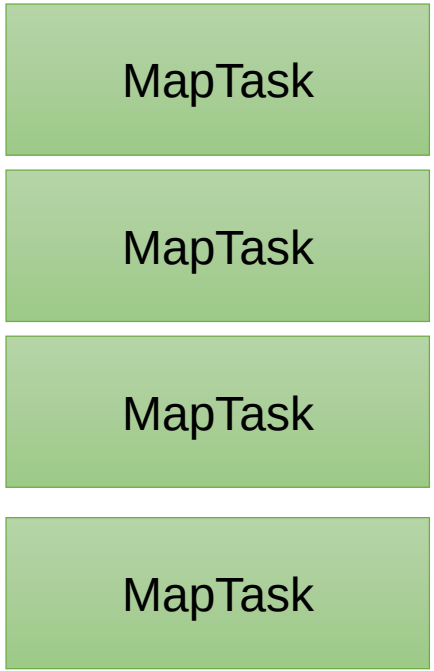
反序列化

upFlow downFlow sumFlow

Map
<key,val, key, val>

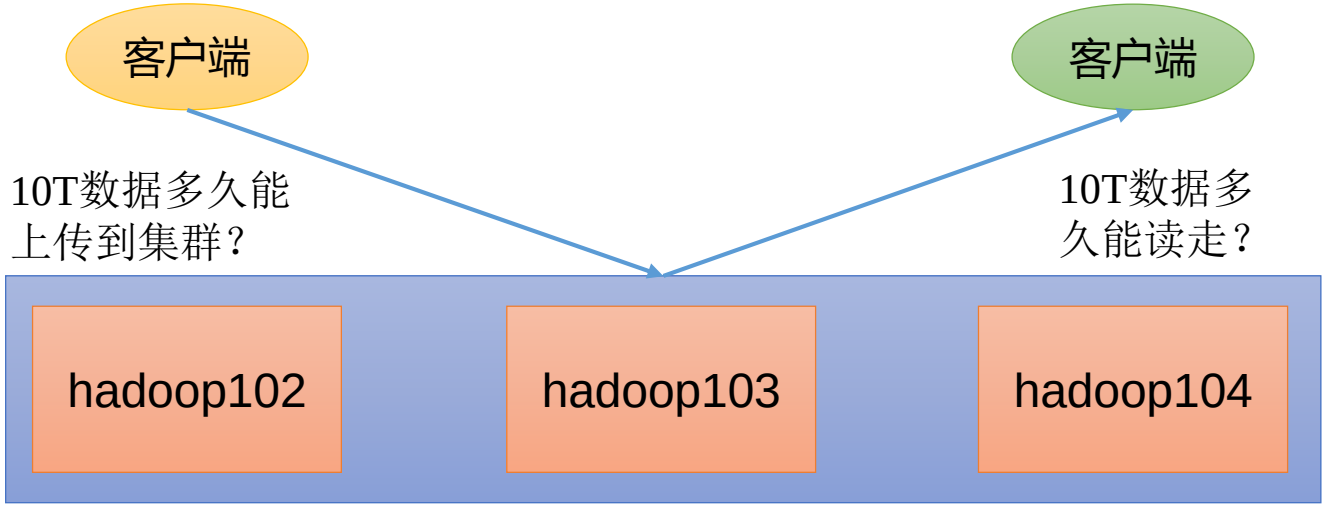


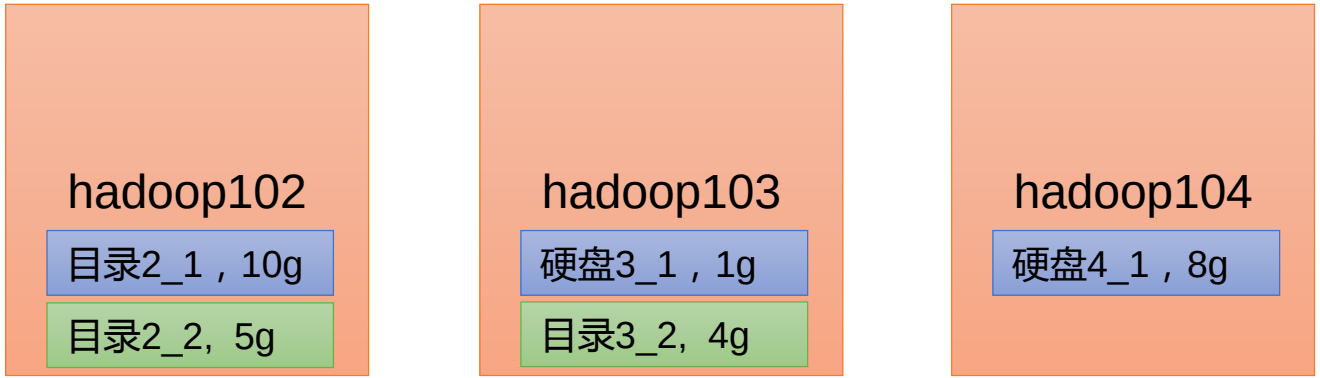
MapTask个数，决定了我的并行度



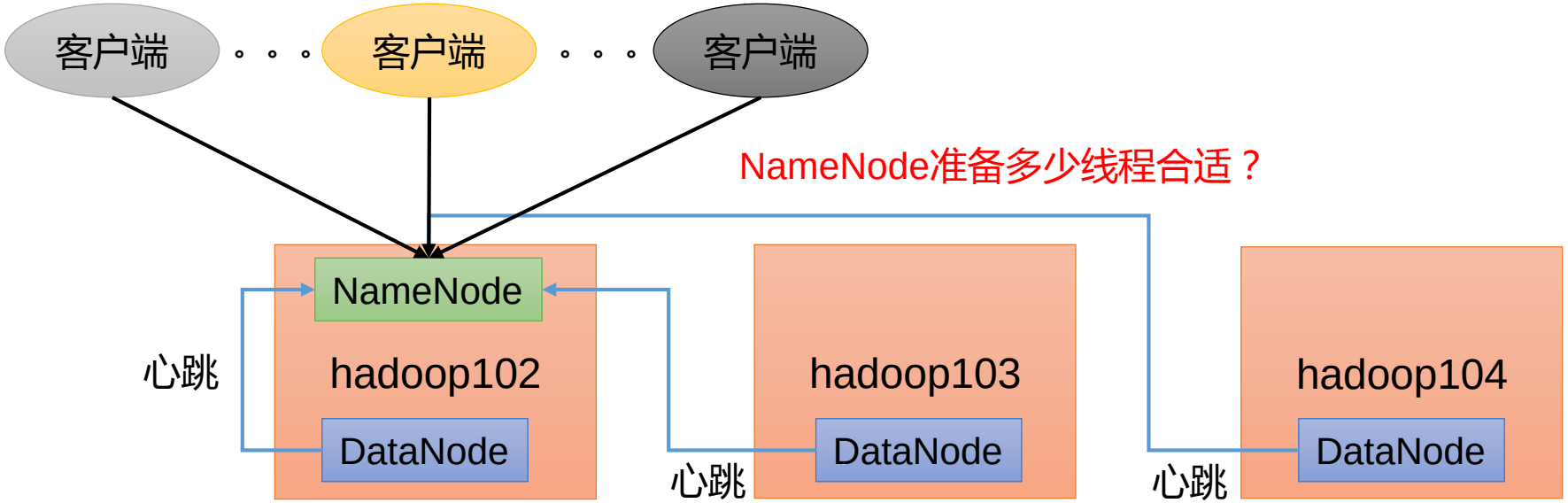
0-128索引 是一片
128-129索引 是一片

逻辑存储

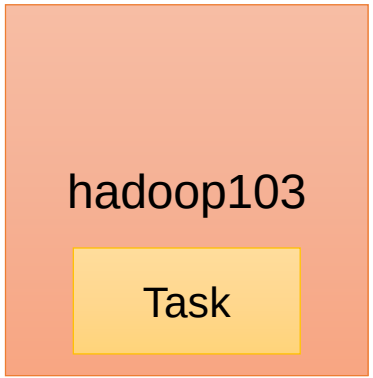

客户端
客户端
10T数据多久能
上传到集群？
10T数据多
久能读走？
hadoop102
hadoop103
hadoop104



每个目录存储的数据不一样


客户端
客户端
客户端
NameNode准备多少线程合适？
NameNode
心跳
hadoop102
DataNode
hadoop103
DataNode
心跳
hadoop104
DataNode
心跳


hadoop102
硬盘2_1，10g
硬盘2_2，0g
hadoop103
硬盘3_1，1g
hadoop104
硬盘4_1，8g



拍摄快照
Ss 18岁

hadoop102
mrAPPMaster

hadoop103
Task

hadoop104
Task

岁月杀猪刀

Ss 50岁

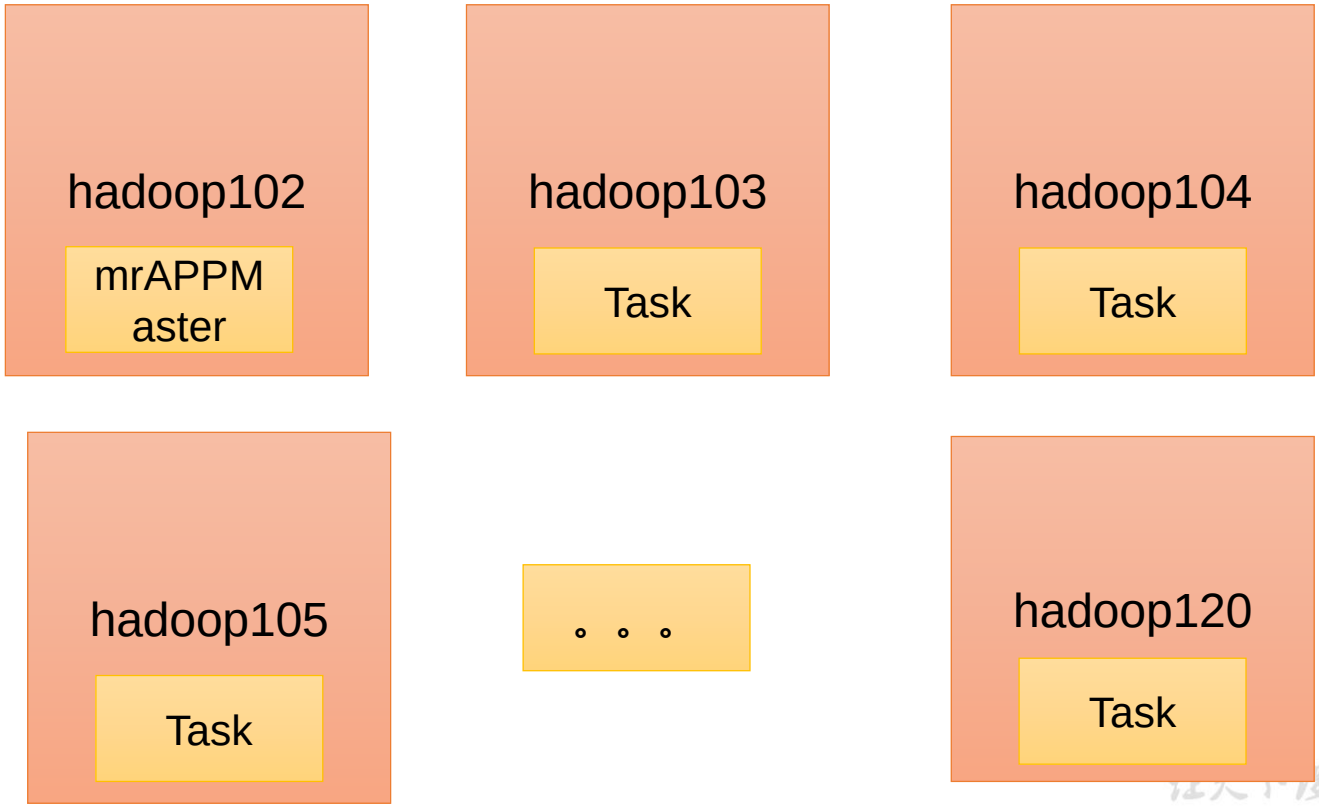
hadoop102
mrAPPMaster

hadoop103
Task

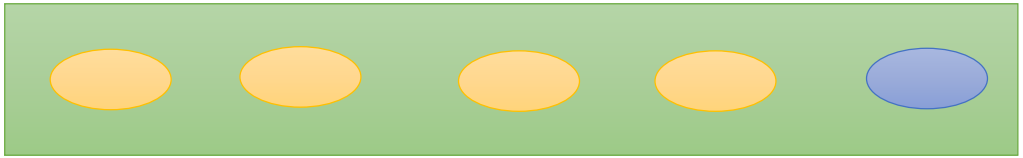
hadoop104
Task

多个队列
11.11　6.18
10个队列

1 /2 3
降级

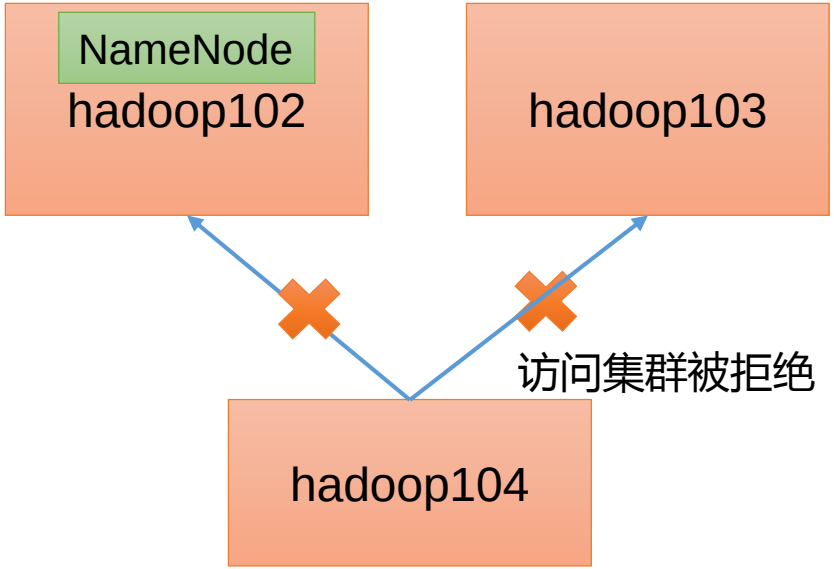


有5个任务

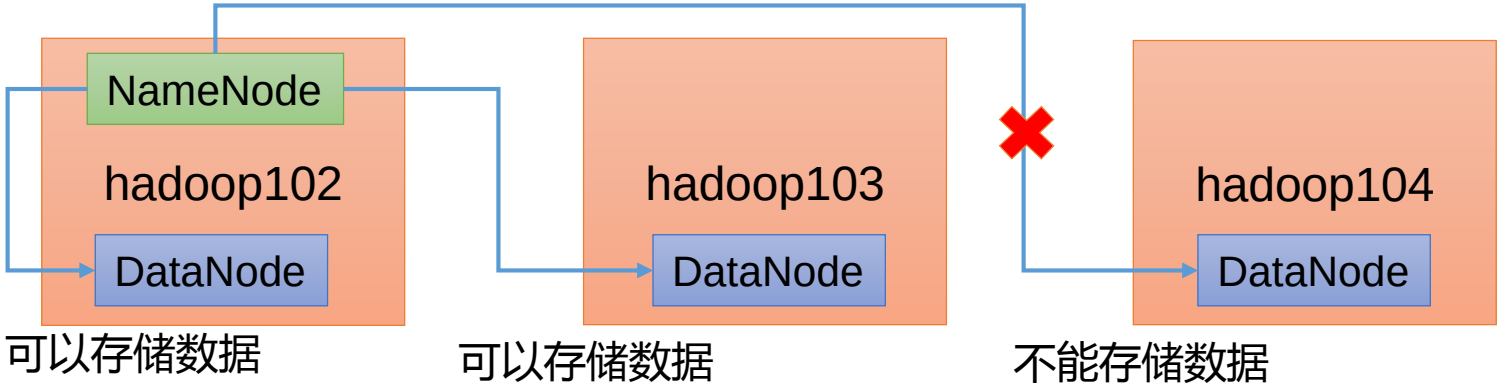


任务1

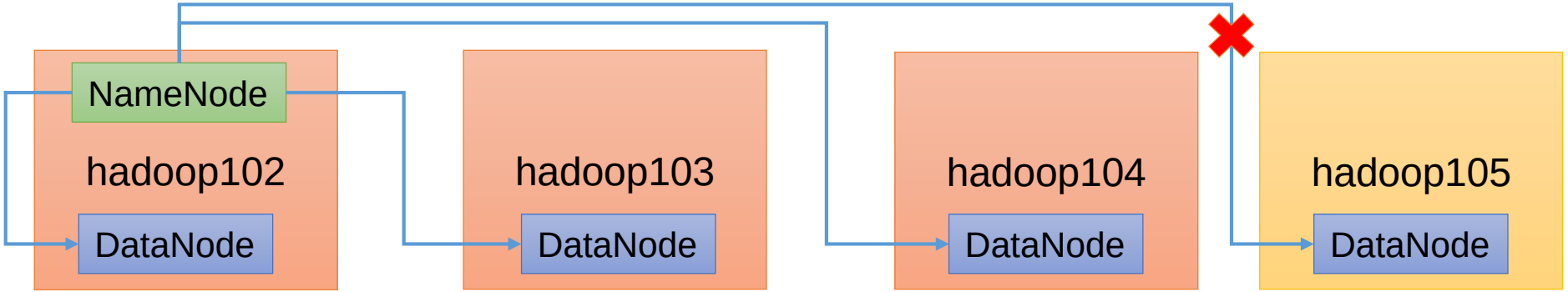
白名单：hadoop102、hadoop103



白名单：hadoop102、hadoop103



黑名单：hadoop105



hadoop102

硬盘2_1，10g

hadoop103

硬盘3_1，1g

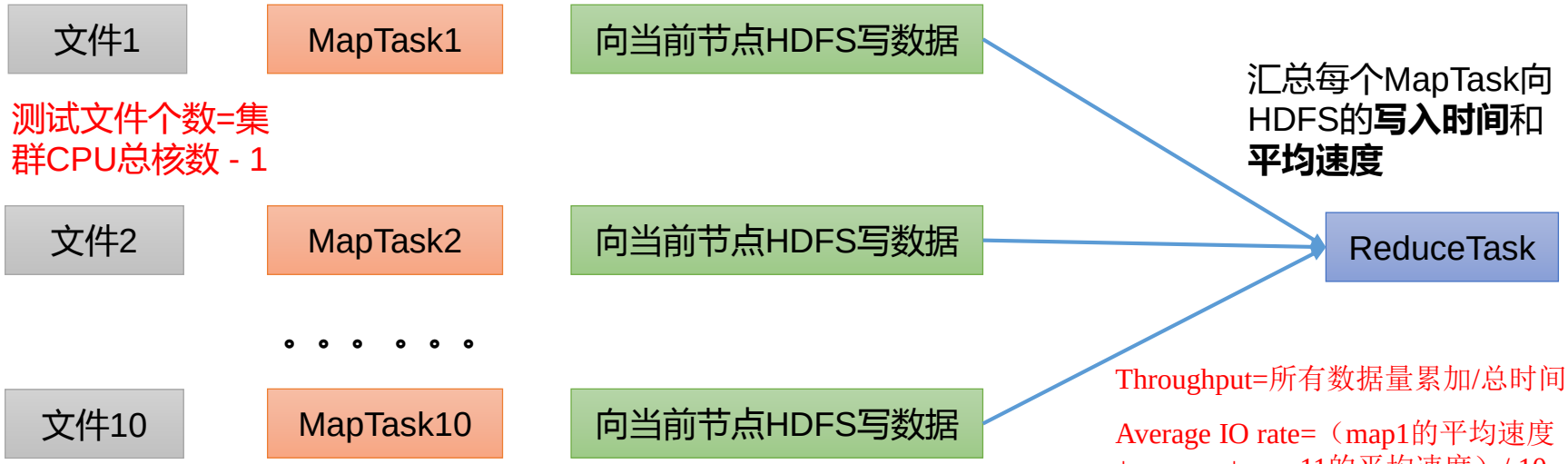
hadoop104

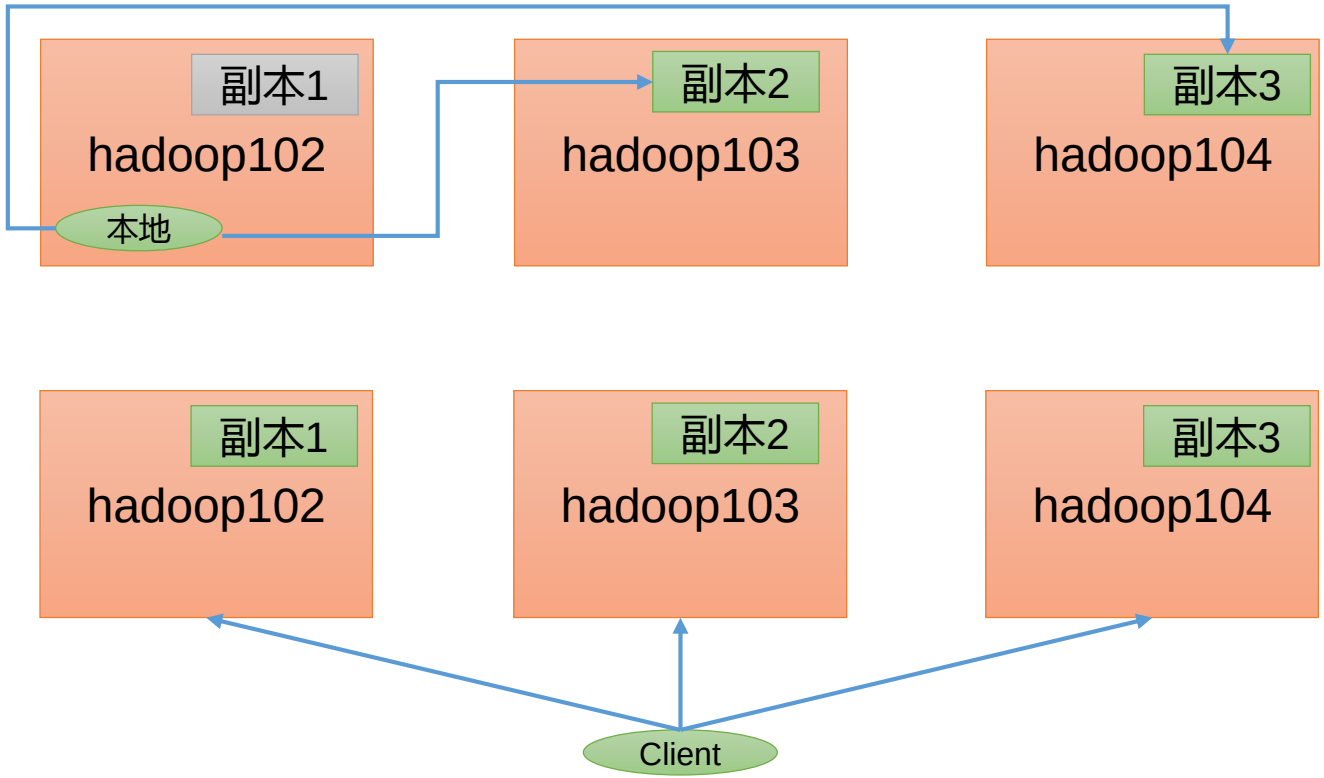
硬盘4_1，8g

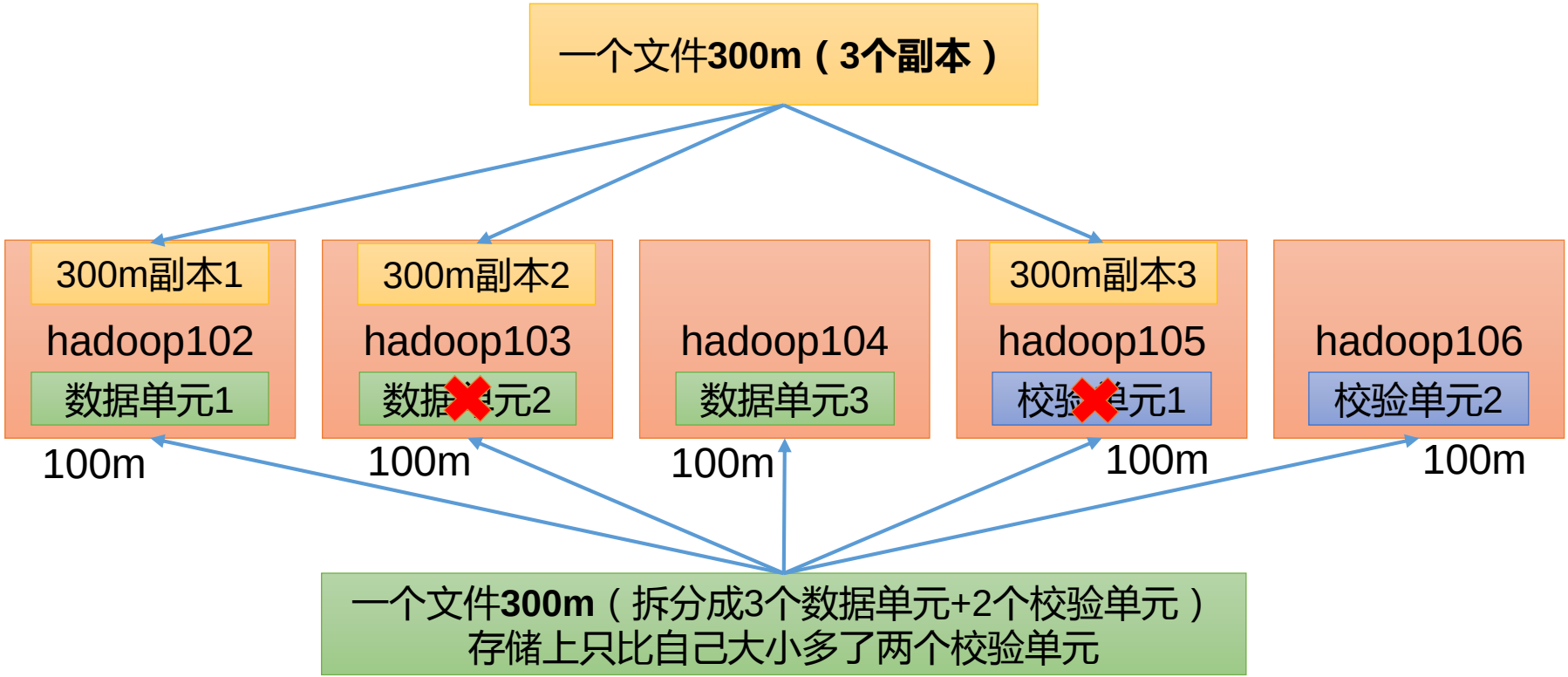
hadoop105

硬盘5_1，0g

记录每个Map的**写时间**和**平均速度**

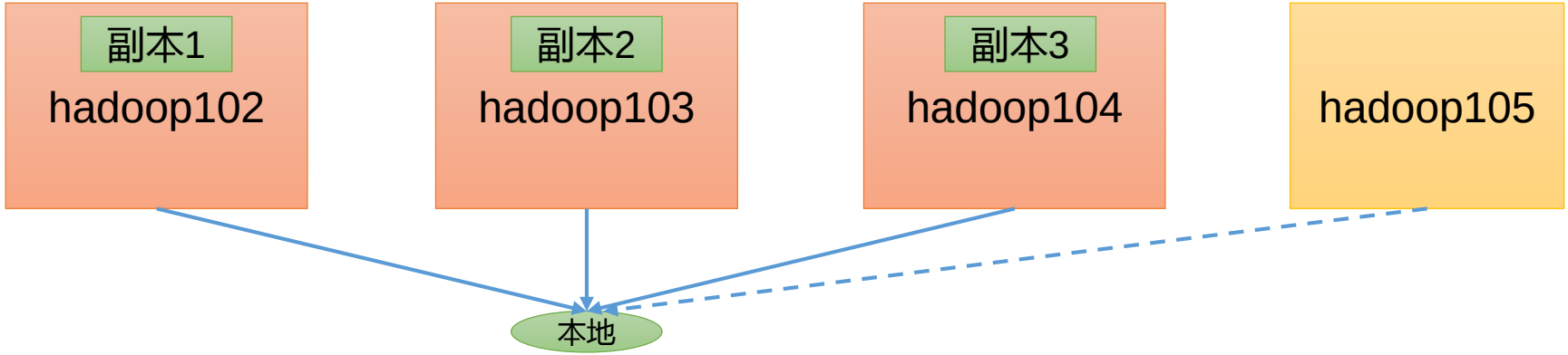



副本1
hadoop102
本地
副本2
hadoop103
副本3
hadoop104
副本1
hadoop102
副本2
hadoop103
副本3
hadoop104
Client


一个文件300m（3个副本）
300m副本1
hadoop102
数据单元1
300m副本2
hadoop103
数据单元2
hadoop104
数据单元3
300m副本3
hadoop105
校验单元1
hadoop106
校验单元2
100m
100m
100m
100m
100m
一个文件300m（拆分成3个数据单元+2个校验单元）
存储上只比自己大小多了两个校验单元

纠删码策略是给具体一个路径设置。

默认只开启对RS-6-3-1024k策略的支持，如要使用别的策略需要提前启用。



SS

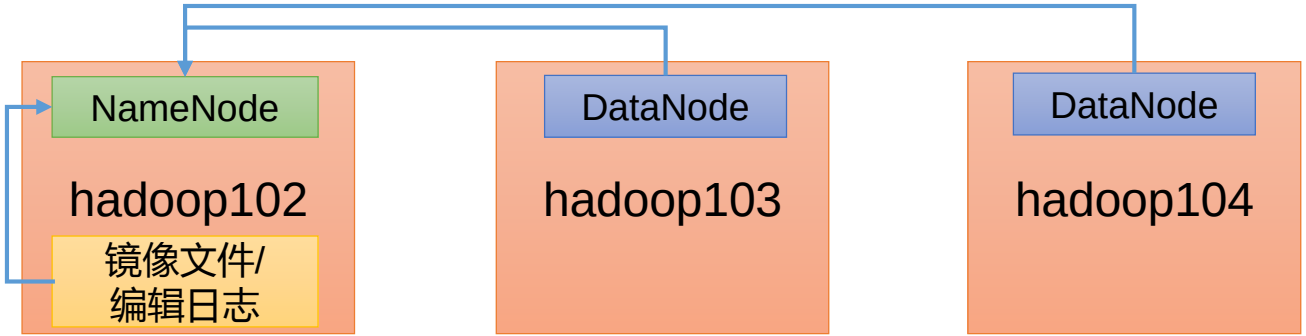
正在看的、经常看的、不经常看的、永久保存的视频，怎么存？

hadoop102	hadoop103	hadoop104	hadoop105
硬盘5_1，内存镜像	硬盘2_1，固态	硬盘3_1，机械	硬盘4_1，破旧

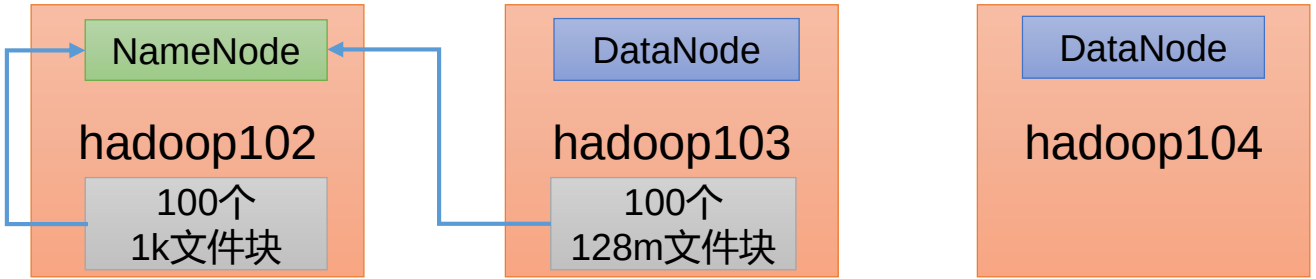
1）安全模式：文件系统只接受读数据请求，而不接受删除、修改等变更请求

2）进入安全模式场景

- NameNode在加载镜像文件和编辑日志期间处于安全模式；
- NameNode再接收DataNode注册时，处于安全模式



100个1k文件块和100个128m的文件块，占用NN内存大小一样



接口协议{
    版本id;
    创建文件夹;
}

服务端实现接口协议

重写创建文件夹方法

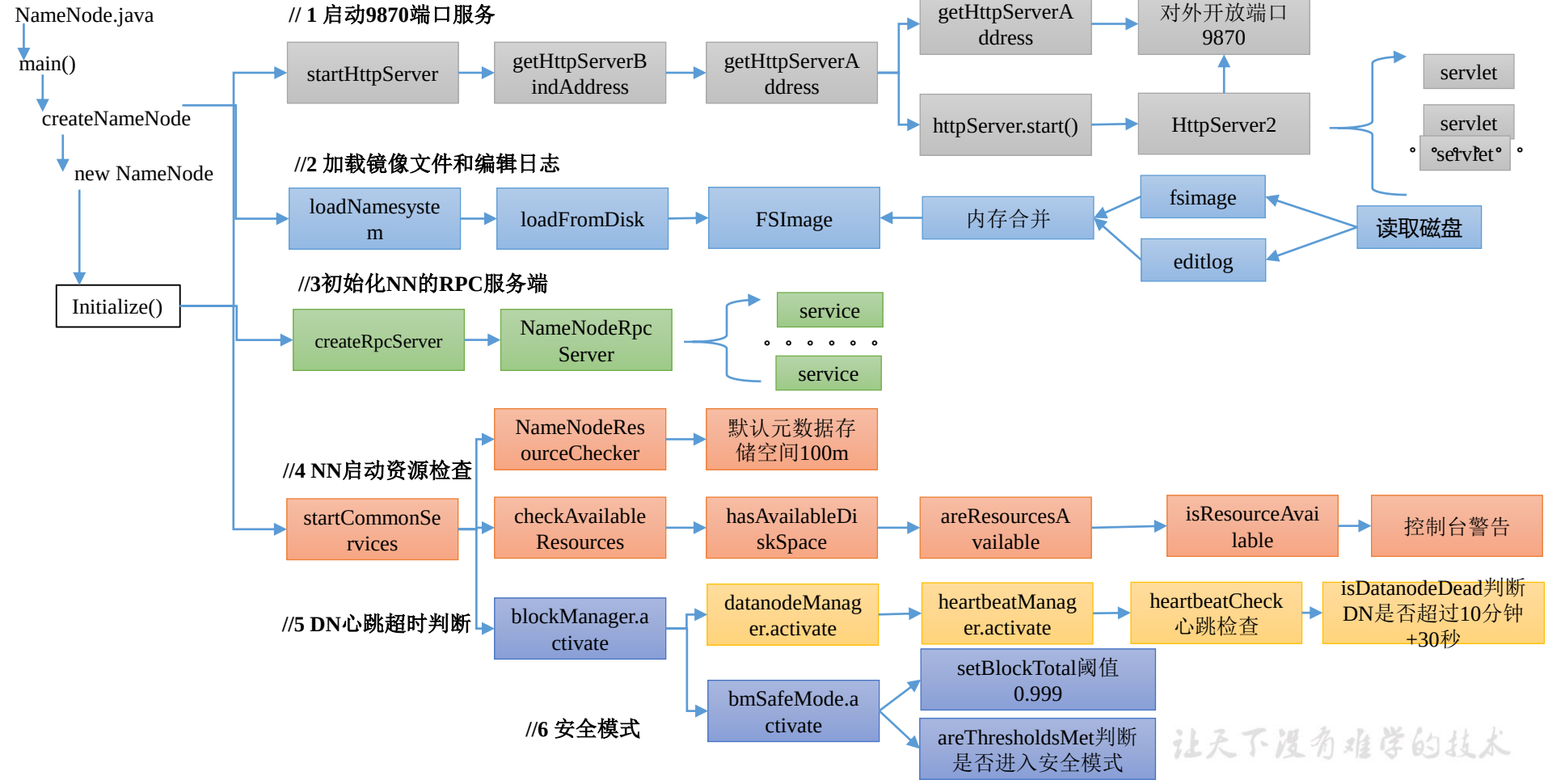
创建RPC服务（服务器地址、端口号、通信协议）

客户端

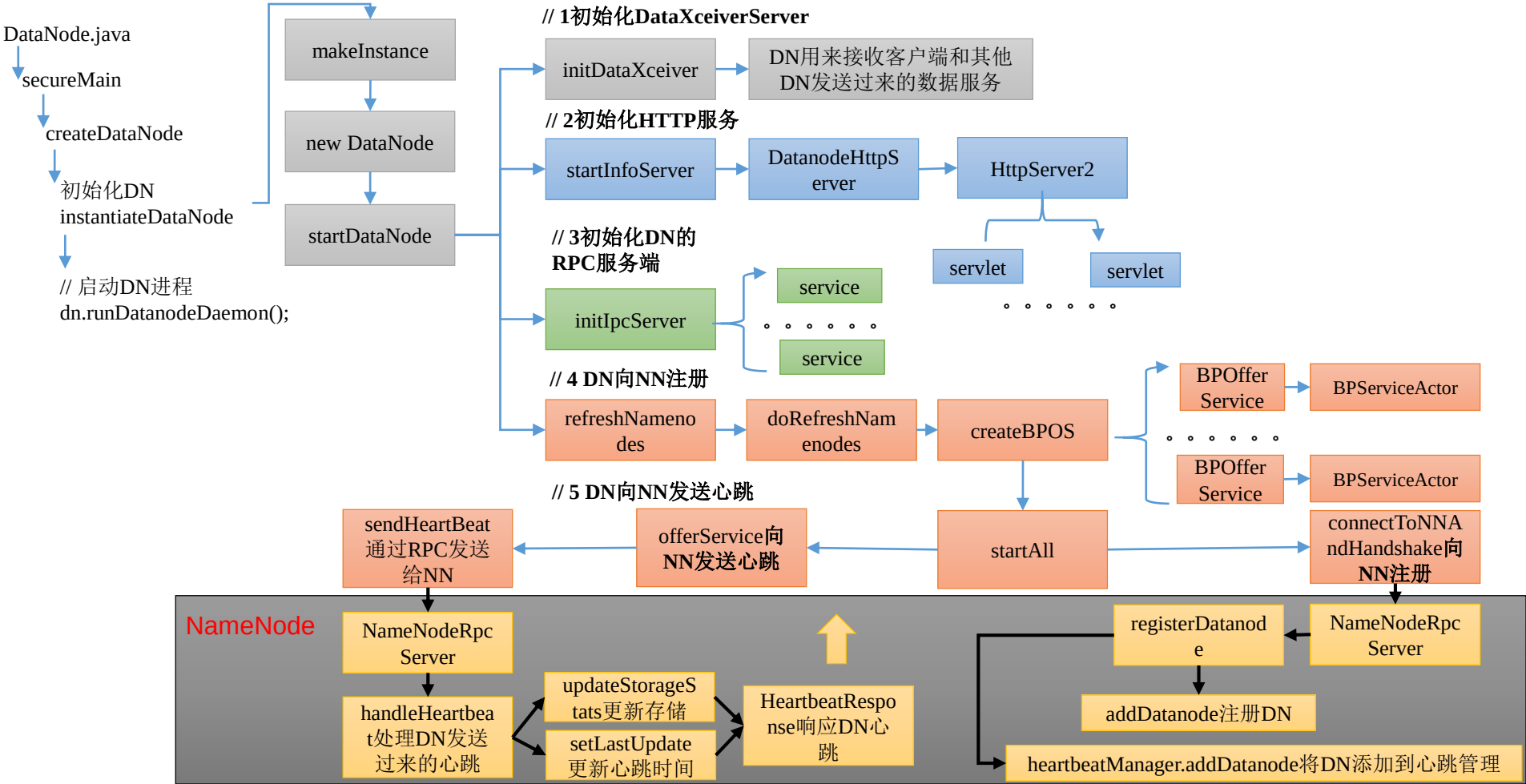
获取服务端代理（服务器地址、端口号、通信协议）

服务端代理调用：创建文件夹方法

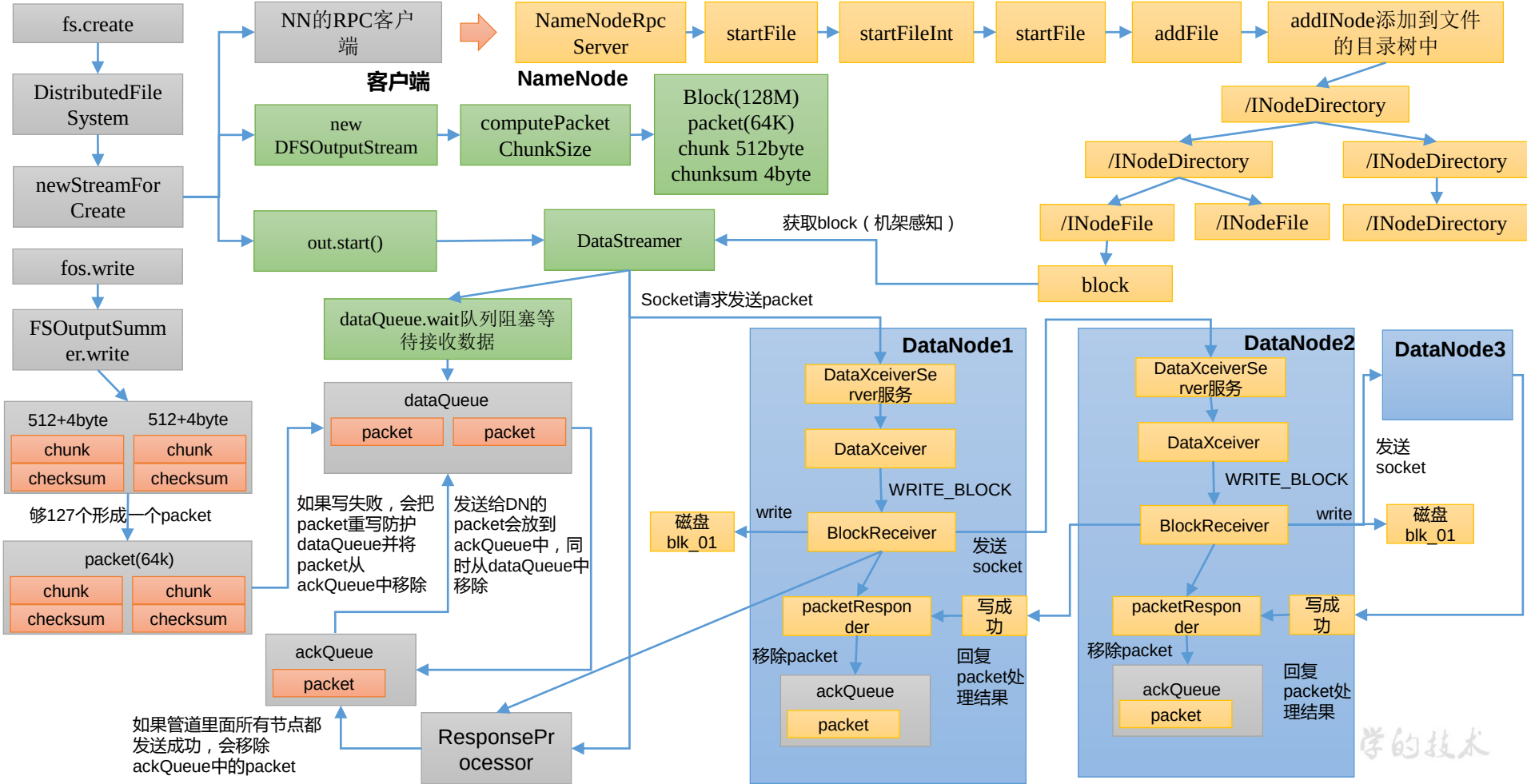
NameNode启动流程源码



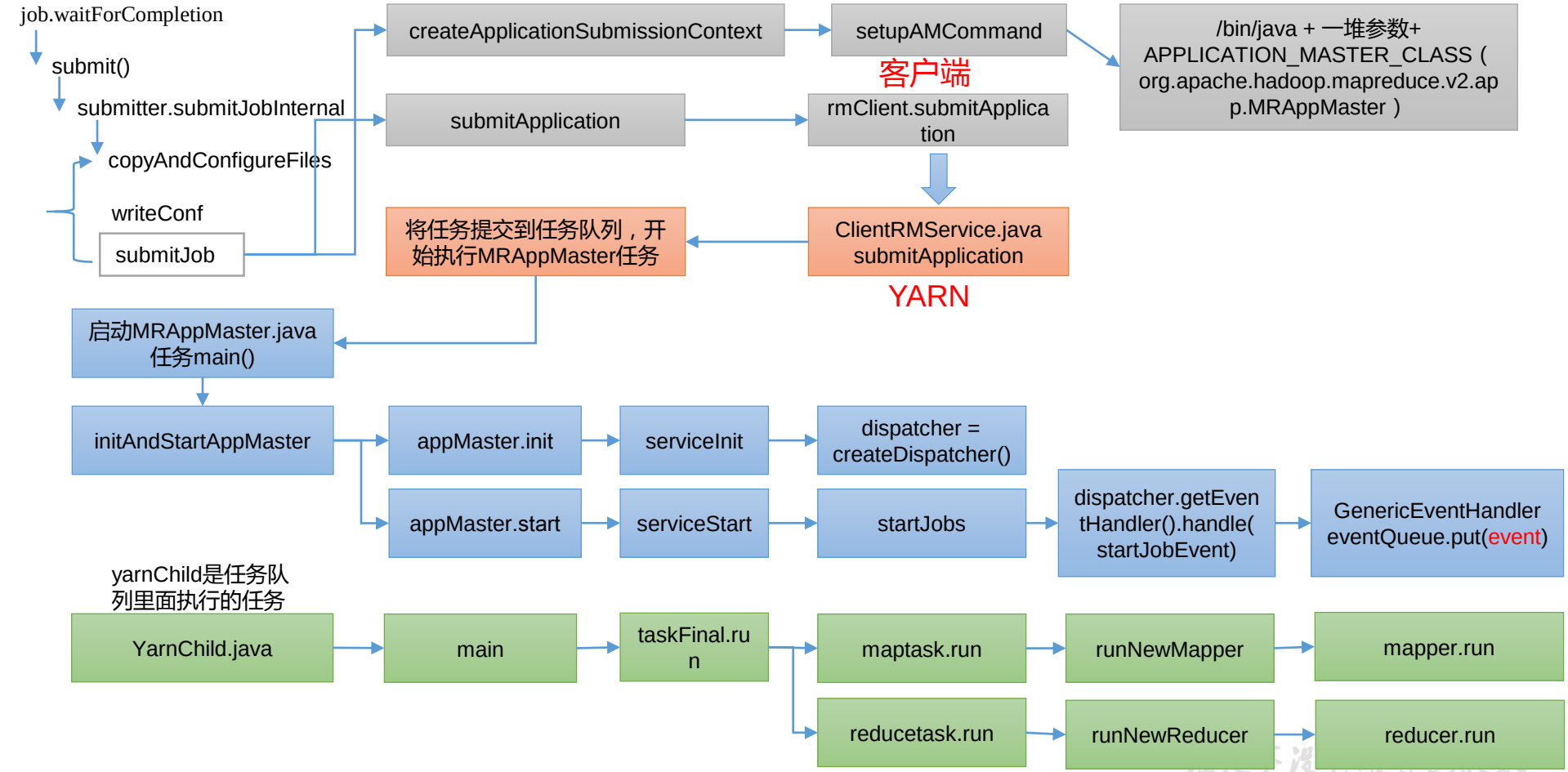
DataNode启动流程



HDFS上传源码解析


fs.create
DistributedFileSystem
newStreamForCreate
fos.write
FSOutputSummer.write
NN的RPC客户端
客户端
NameNodeRpc Server
startFile
startFileInt
startFile
addFile
addINode添加到文件的目录树中
NameNode
new DFSOutputStream
computePacket ChunkSize
Block(128M) packet(64K) chunk 512byte chunksum 4byte
/INodeDirectory
/INodeDirectory
/INodeDirectory
/INodeFile
/INodeFile
/INodeDirectory
block
获取block（机架感知）
out.start()
DataStreamer
Socket请求发送packet
dataQueue.wait队列阻塞等待接收数据
dataQueue
packet
packet
512+4byte
512+4byte
chunk
checksum
chunk
checksum
够127个形成一个packet
packet(64k)
chunk
checksum
chunk
checksum
如果写失败，会把packet重写防护dataQueue并将packet从ackQueue中移除
发送给DN的packet会放到ackQueue中，同时从dataQueue中移除
ackQueue
packet
如果管道里面所有节点都发送成功，会移除ackQueue中的packet
ResponseProcessor
DataNode1
DataXceiverServer服务
DataXceiver
WRITE_BLOCK
write
BlockReceiver
磁盘 blk_01
发送 socket
packetResponder
写成功
移除packet
回复packet处理结果
ackQueue
packet
DataNode2
DataXceiverServer服务
DataXceiver
WRITE_BLOCK
write
BlockReceiver
磁盘 blk_01
发送 socket
packetResponder
写成功
移除packet
回复packet处理结果
ackQueue
packet
DataNode3

YARN源码解析


job.waitForCompletion
submit()
submitter.submitJobInternal
copyAndConfigureFiles
writeConf
submitJob
createApplicationSubmissionContext
setupAMCommand
/bin/java + 一堆参数+ APPLICATION_MASTER_CLASS（ org.apache.hadoop.mapreduce.v2.app.MRAppMaster ）
客户端
submitApplication
rmClient.submitApplication
将任务提交到任务队列，开始执行MRAppMaster任务
ClientRMService.java submitApplication
YARN
启动MRAppMaster.java 任务main()
initAndStartAppMaster
appMaster.init
serviceInit
dispatcher = createDispatcher()
appMaster.start
serviceStart
startJobs
dispatcher.getEventHandler().handle(startJobEvent)
GenericEventHandler eventQueue.put(event)
yarnChild是任务队列里面执行的任务
YarnChild.java
main
taskFinal.run
maptask.run
runNewMapper
mapper.run
reducetask.run
runNewReducer
reducer.run