淘宝网
Taobao.com

HDFS元数据的独立服务和独立持久化存储

罗李
Email: luoli523@gmail.com
Twitter: luoli523

主要内容

- 起因
- 现状
- 我们的想法
- 我们的实现
- 后续的发展

起因

- 数据的急剧膨胀
- 文件数的不断增多
- Block随之成倍的增长
- 内存的急剧上涨
- 内存数据结构
- 一致性保证造成的性能瓶颈
- Meta服务依靠namenode的启停
- 部分meta数据没有持久化(block->dn)

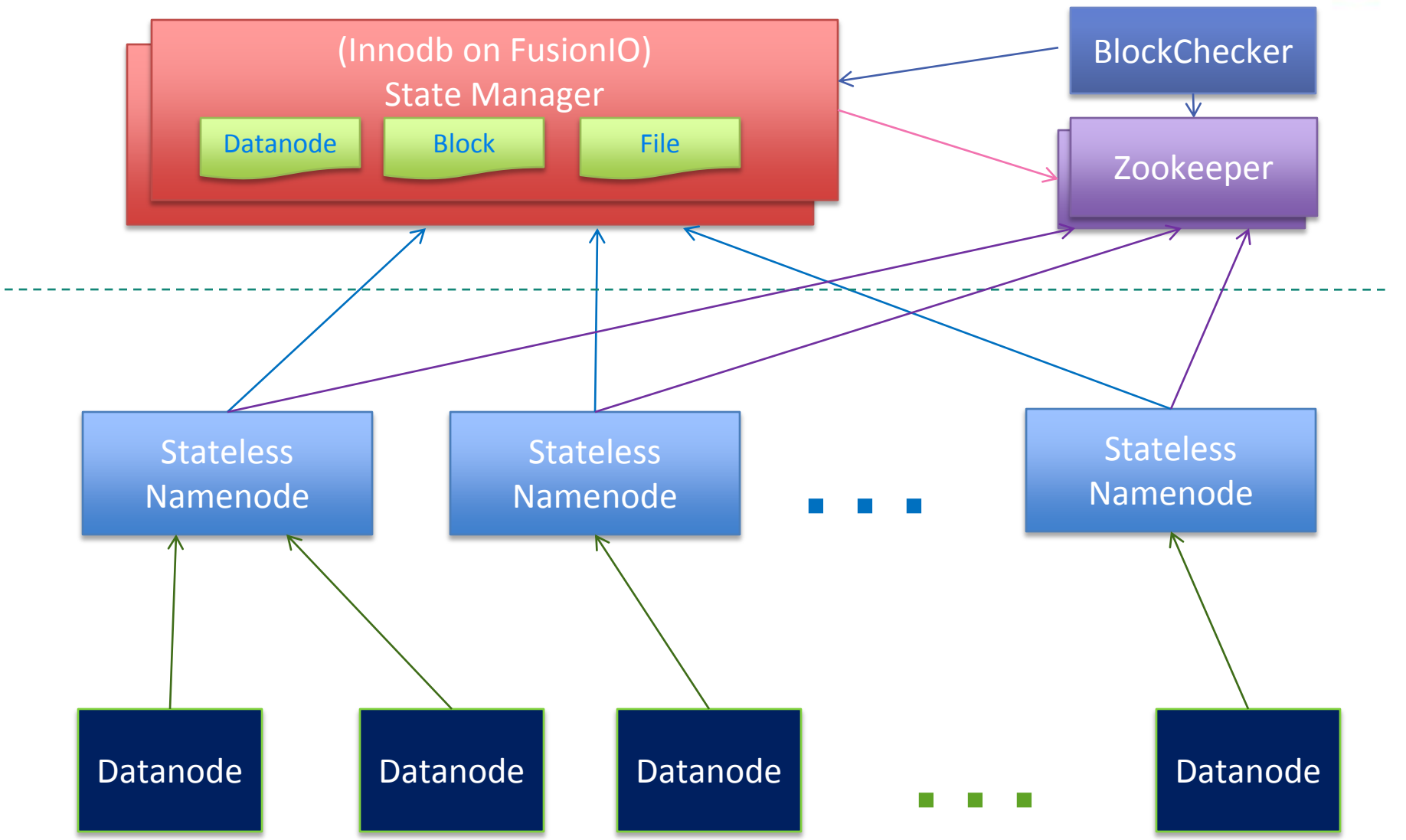
现状

- 集群
  - 单个集群1900台机器 1T×12（2T×6）
- 数据量
  - 22.28 PB/36.98 PB 60%
- 文件数
  - 1亿左右
- Block数
  - 1.3亿左右
- Meta存储
  - 只持久化了namespace的信息到fsimage

现状

- 内存
  - 60G / 80G ～75%
- 数据结构
  - BlockMap靠内存中ref来维护block->dn的信息
- 响应
  - 删除文件个数1100万，每天的删除操作为240万
  - 创建文件操作900万～1200万
  - 重命名文件数量为1050万
  - 通过文件名获取block及其位置的操作getBlockLocations有近3亿
  - 类似“ls”的操作有700万

新的架构

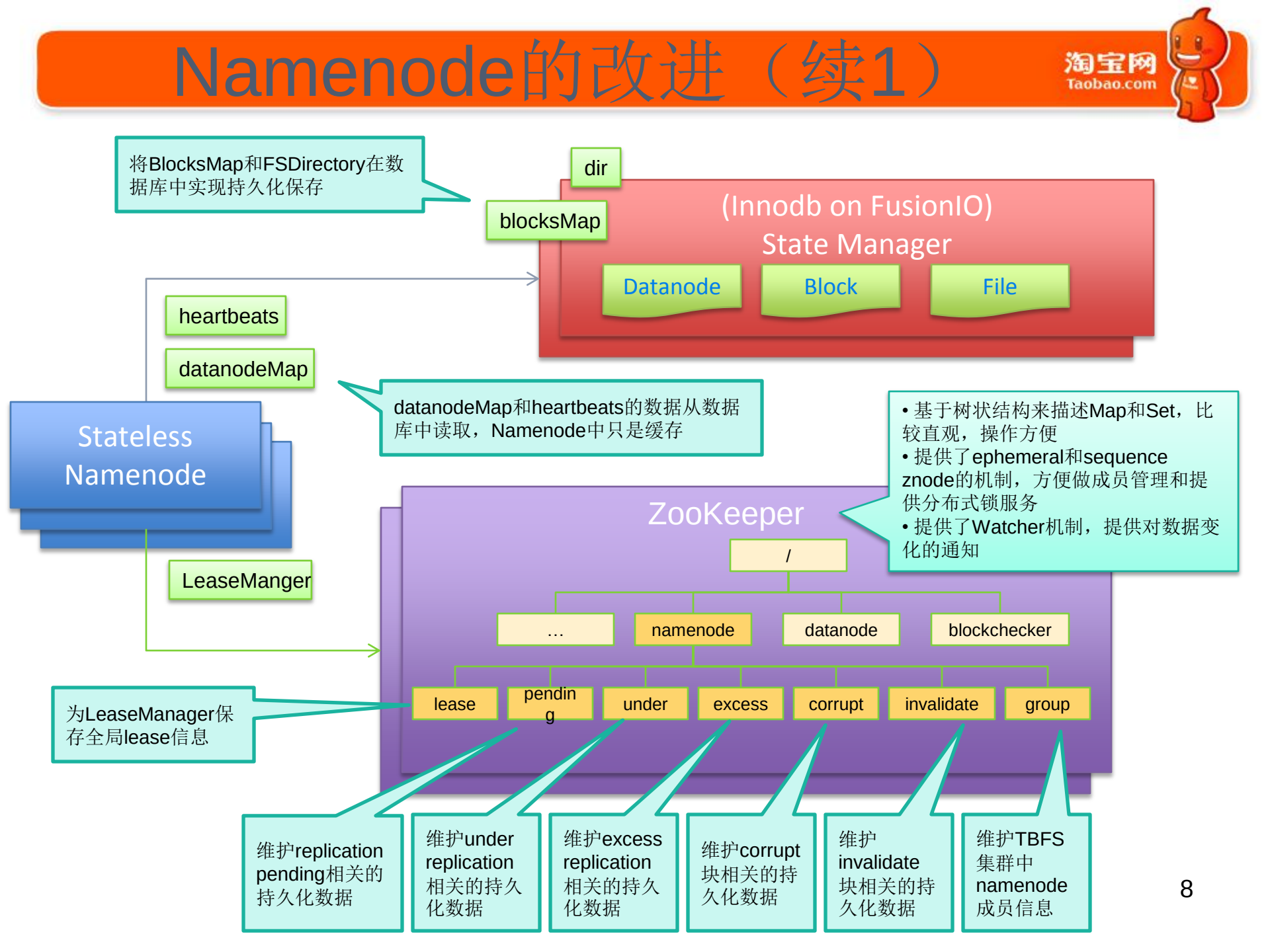

(Innodb on FusionIO)
State Manager
Datanode
Block
File
BlockChecker
Zookeeper
Stateless Namenode
Stateless Namenode
Stateless Namenode
Datanode
Datanode
Datanode
Datanode

Namenode的改进

- 无状态NN: 针对HDFS中Namenode单点瓶颈的问题，TBFS通过无状态方式实现Namenode的水平扩展。为了实现无状态Namenode，需要将以前保留在Namenode内存中的关键数据结构部分或全部挪到第三方，并持久化保存。

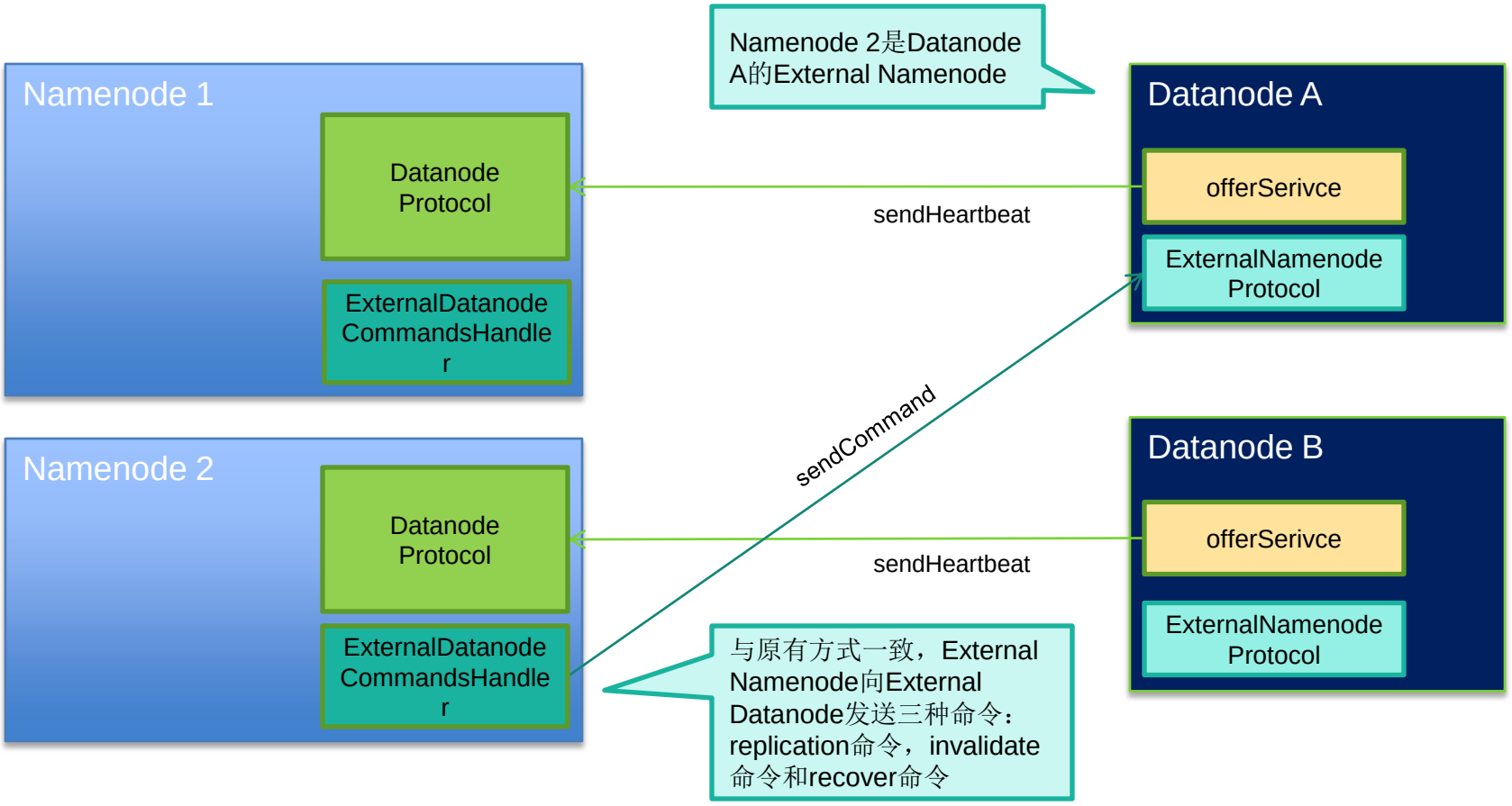
数据结构名称	描述
dir	保存HDFS目录结构的数据结构FSDirectory（文件->块的对应关系）
blocksMap	保存块与文件、块与datanode和datanode与块的对应关系
datanodemap	保存datanode的storageID和对应DatanodeDescriptor的Map容器
heartbeats	保存拥有心跳的Datanode的DatanodeDescriptor的容器
corruptReplicas	保存损坏块的Map容器，key为Block，value为对应Datanode的DatanodeDescriptor集合
recentInvalidateSets	保存即将删除的块的Map容器，key为Datanode的StorageID，value是块的Block集合
excessReplicateMap	保存多余块的Map容器，key为Datanode的storageID，value是块的Block集合
neededReplications	保存少于replication数的块的数据结构，其内部维护了一个List<TreeSet<Block>>类型的优先级队列
pendingReplications	保存处于replication pending状态的block，如果超时则放入TimeoutItems列表中
leaseManager	维护写操作和追加操作租约的数据结构

Namenode的改进（续1）



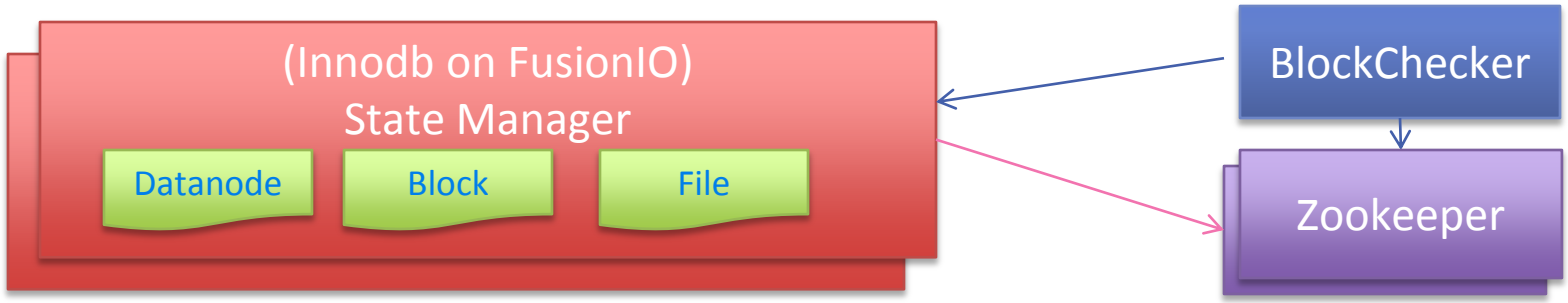
Namenode的改进（续2）

- Namenode与非心跳Datanode进行通信。Datanode实现了ExternalNamenodeProtocol协议，Namenode可以通过该协议与非心跳Datanode进行通信，即Namenode主动调用该协议提供的方法。



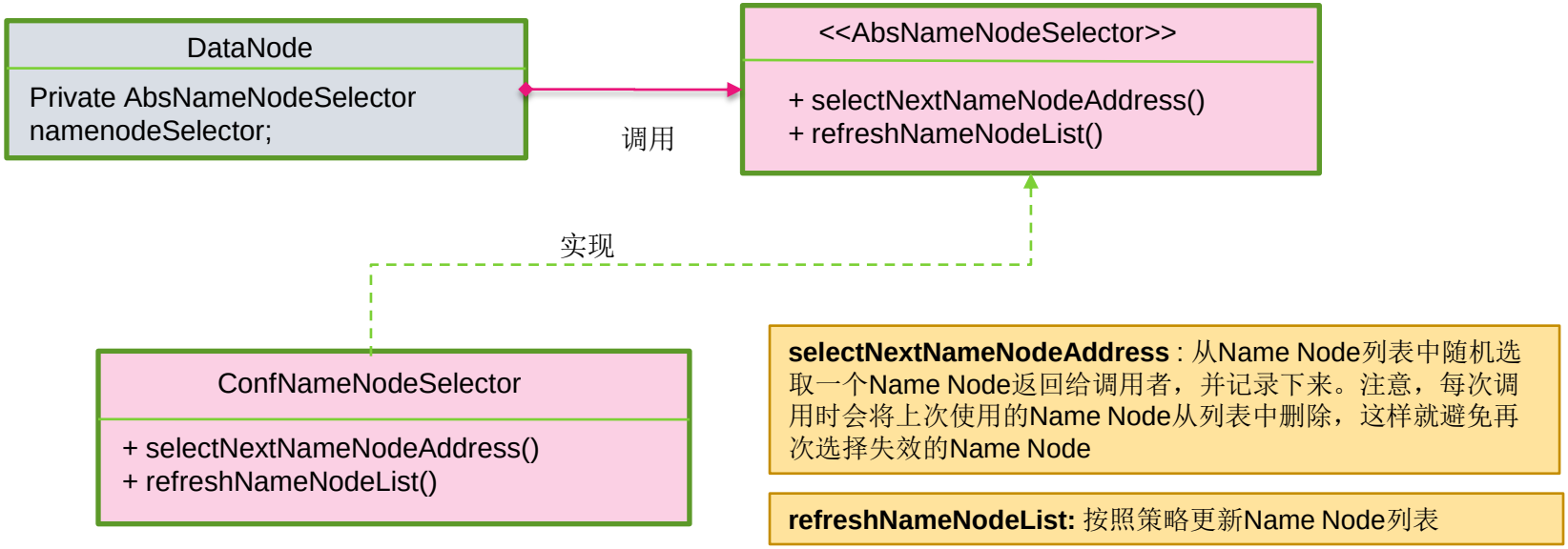
BlockChecker的引入

- BlockChecker解决Namenode无法判断出的数据不一致的情况，主要是检测Block副本数是否满足期望，类似于社区版中离开安全模式(SafeMode.leave)时processMisReplicatedBlocks机制。为了不影响Namenode的核心逻辑，它只和数据库和Zookeeper交互。
- 运行方式：1. 每隔一段时间运行一次；2. 手动执行；3. Namenode下线时执行
- 典型场景：
  - 某个block的副本数小于期望值，在数据库中增加一条伪记录，触发Namenode进行检查
  - 某个block的副本数大于期望值，综合zookeeper中的记录，决定是否删除一条记录，触发Namenode进行检查



Datanode的改进

- 提供Namenode的连接/重连机制，从而提高整个系统的可用性。在以下几种场景下，Datanode会连接/切换目标Namenode： 1. Datanode启动时；2. 当前Namenode失效（异常）并超过一定时限和重试次数；3. 管理员调用切换命令。同一时刻一个Datanode只汇报给一个Namenode。
- Namenode选择策略实现： AbsNameNodeSelector作为选择Namenode策略的接口，ConfNameNodeSelector实现了该接口。



Datanode的改进（续1）

- 目前已实现的Namenode选择策略ConfNameNodeSelector需要在配置文件中做如下配置

```
<property>
  <name>dfs.namenode.selector</name>
  <value>org.apache.hadoop.hdfs.server.common.ConfNameNodeSelector</value>
  <description>The policy of looking for and selecting name node</description>
</property>

<property>
  <name>dfs.namenode.selector.timeout</name>
  <value>180000</value>
  <description>The timeout value for retrying connection to a namenode</description>
</property>

<property>
  <name>dfs.namenode.rpcaddr.list</name>
  <value>hdfs://dw30.kgb.sqa.cm4:51199,hdfs://dw39.kgb.sqa.cm4:51199</value>
  <description>The list of name nodes' RPC addr list, separated with comma</description>
</property>
```

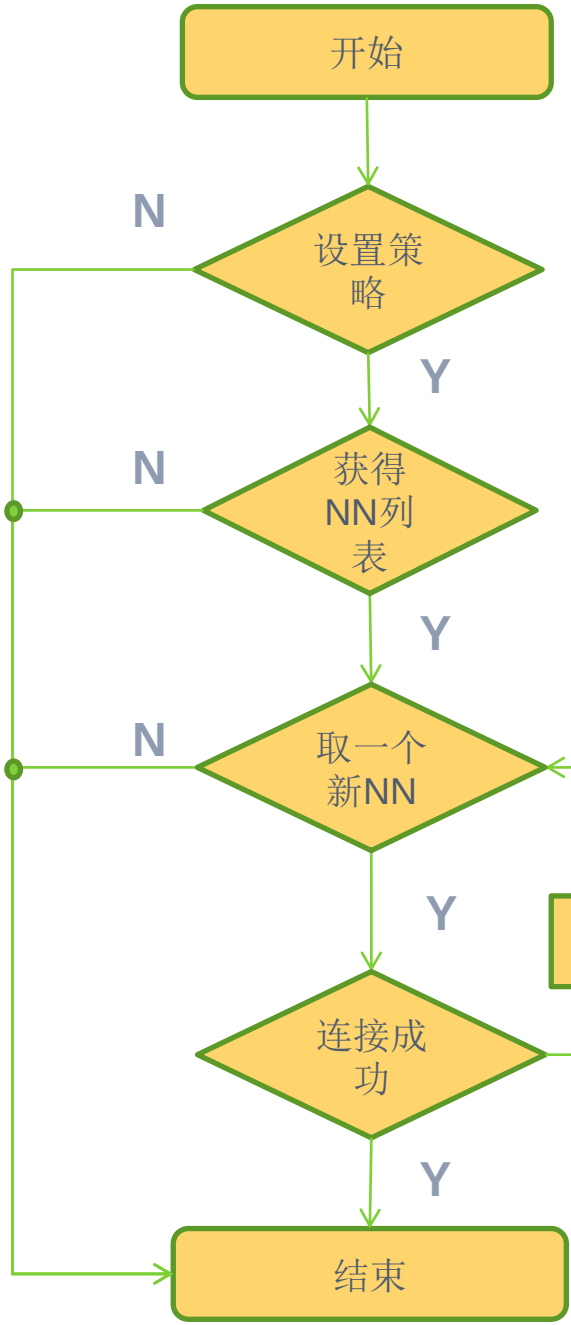
ConfNameNodeSelector的类路径

一个Namenode失效后重连的超时时间

Namenode的列表

- Datanode在线辅助判断机制。Datanode上线后，在zookeeper中创建一个Ephemeral Node，用以给Namenode判断该Datanode是否在线。该类型的Node会在Datanode下线后(会话失效)自动删除。如果Namenode通过datanode表中的lastupdate判断已经下线，但是zookeeper中还有对应的node，会将其列入怀疑对象。造成这种现象一般在TBFS重启初期，Namenode信息更新不及时。怀疑对象一般会在下一次更新时自动排除，否则就认为它已经下线。

流程一：DN启动时连接NN。DN需要根据选取策略，从NN列表中选取一个可用的NN地址建立连接，否则流程失败



在Data Node初始化时执行下面的逻辑，会重构DataNode类的startDataNode方法

DN启动时设定，目前包括两方面：1. 从何处获得NN列表（包括配置文件或者zookeeper）；2. 如何选择NN（随机或者根据某种权值）

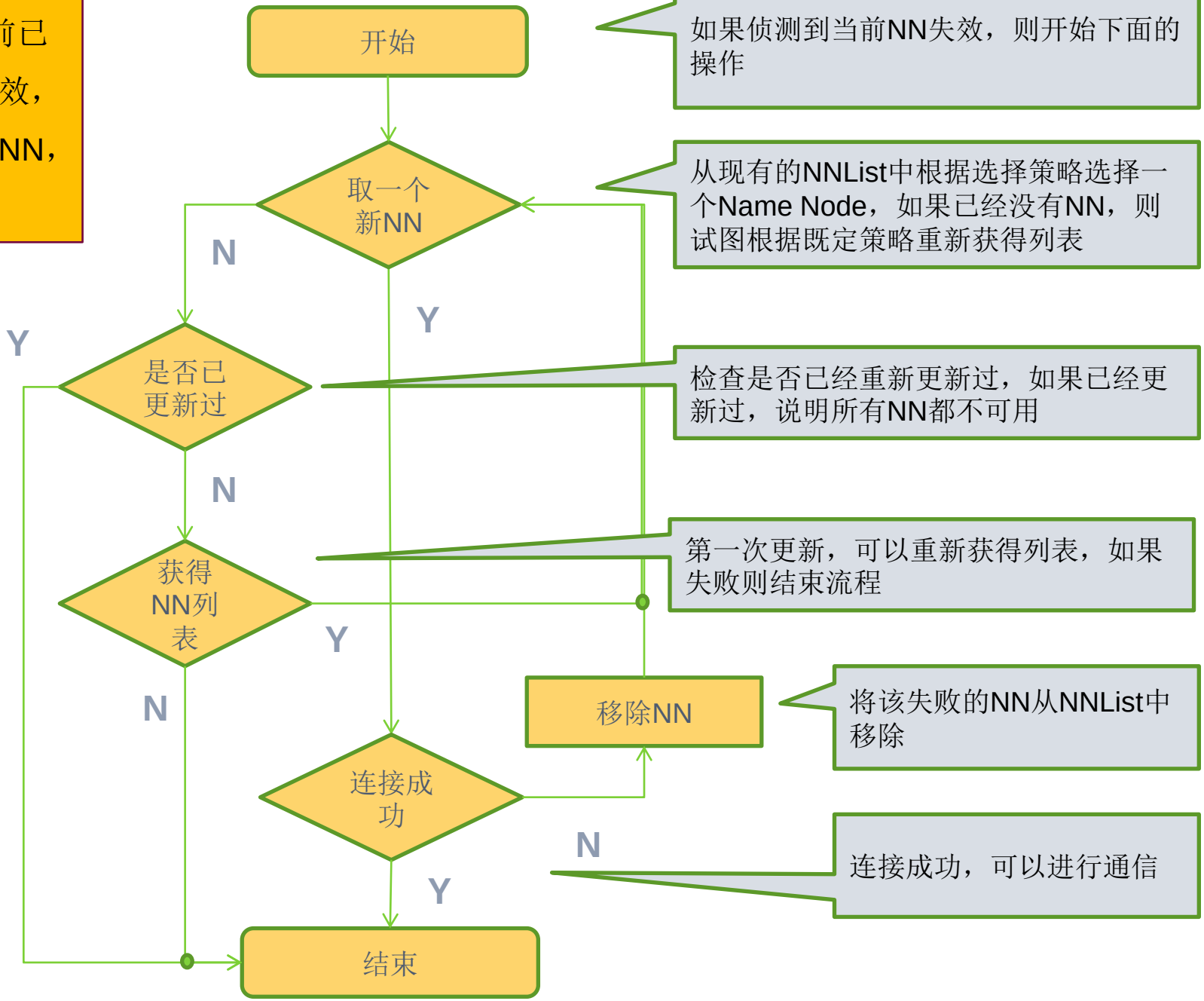
在配置文件中，NN之间以逗号隔开；如果从zookeeper中读取，需要讨论接口。该列表用List<String> NNList表示

从NNList中根据选择策略选择一个Name Node，如果NNList中已经没有可用的，则返回失败

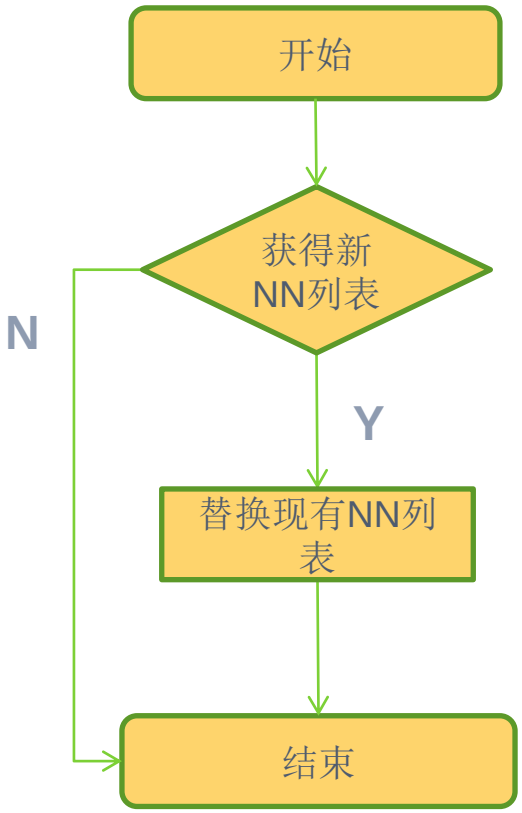
将该失败的NN从NNList中移除

连接成功，可以进行通信

流程二：当前已连接的NN失效，DN重新选择NN，并进行连接



流程三：手动或自动更新NN列表



由管理员或者后台监控进程发起更新NN列表操作

根据策略获得新的NN列表，如果无法获取，则返回失败

更新NNList操作

Client的改进

- 重连机制
- 和datanode同样的机制选择NN节点

淘宝网
Taobao.com

谢谢

罗李
Email: luoli523@gmail.com
Twitter: luoli523