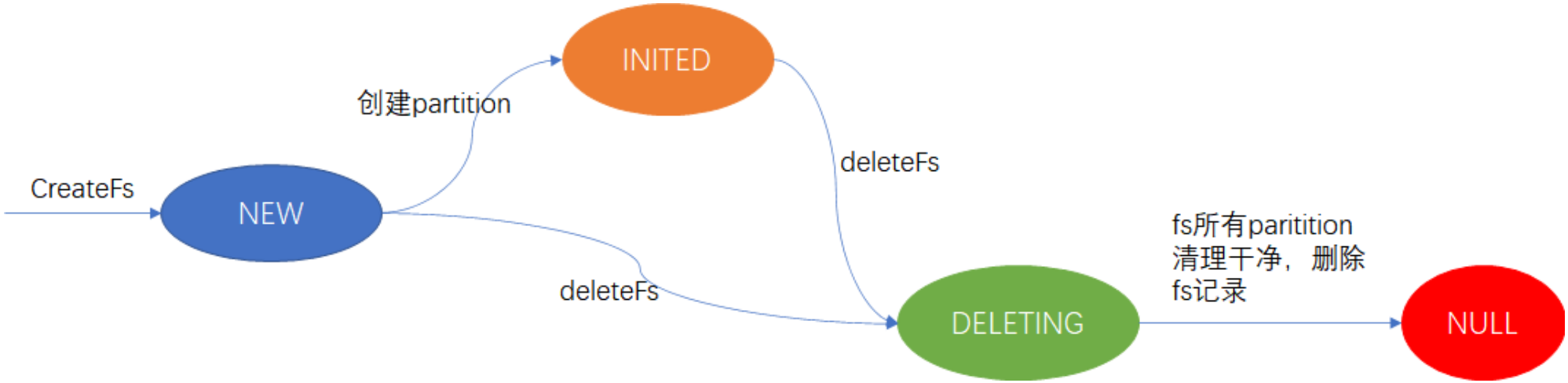
curvefs文件系统删除

删除文件系统，涉及到删除文件系统所有的元数据，删除文件系统所有的数据。文件系统的元数据和数据可能十分庞大，所有删除文件系统时，这里考虑放到后台去进行删除。

当用户需要删除文件系统的时候，curvefs标记文件系统为deleting状态，并生成后台删除任务，交给后台线程去进行处理，此时curvefs可直接向用户返回删除成功。删除中的fs会修改名字，在用户看来，fs已经删除成功，不影响用户继续创建同名fs。

目前文件系统的状态变化为NEW → INITED → DELETING → 查询不到fs。

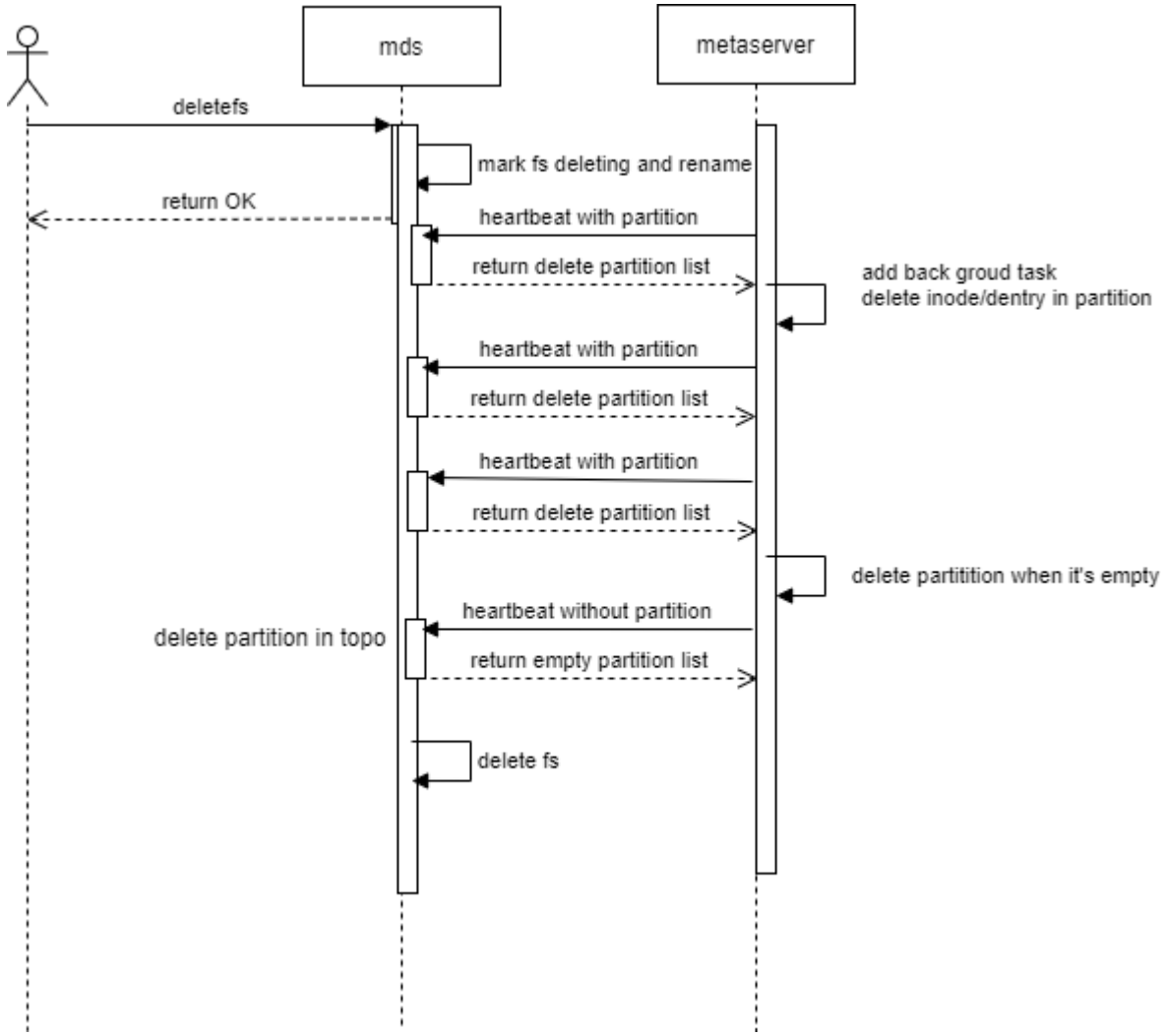


注意：

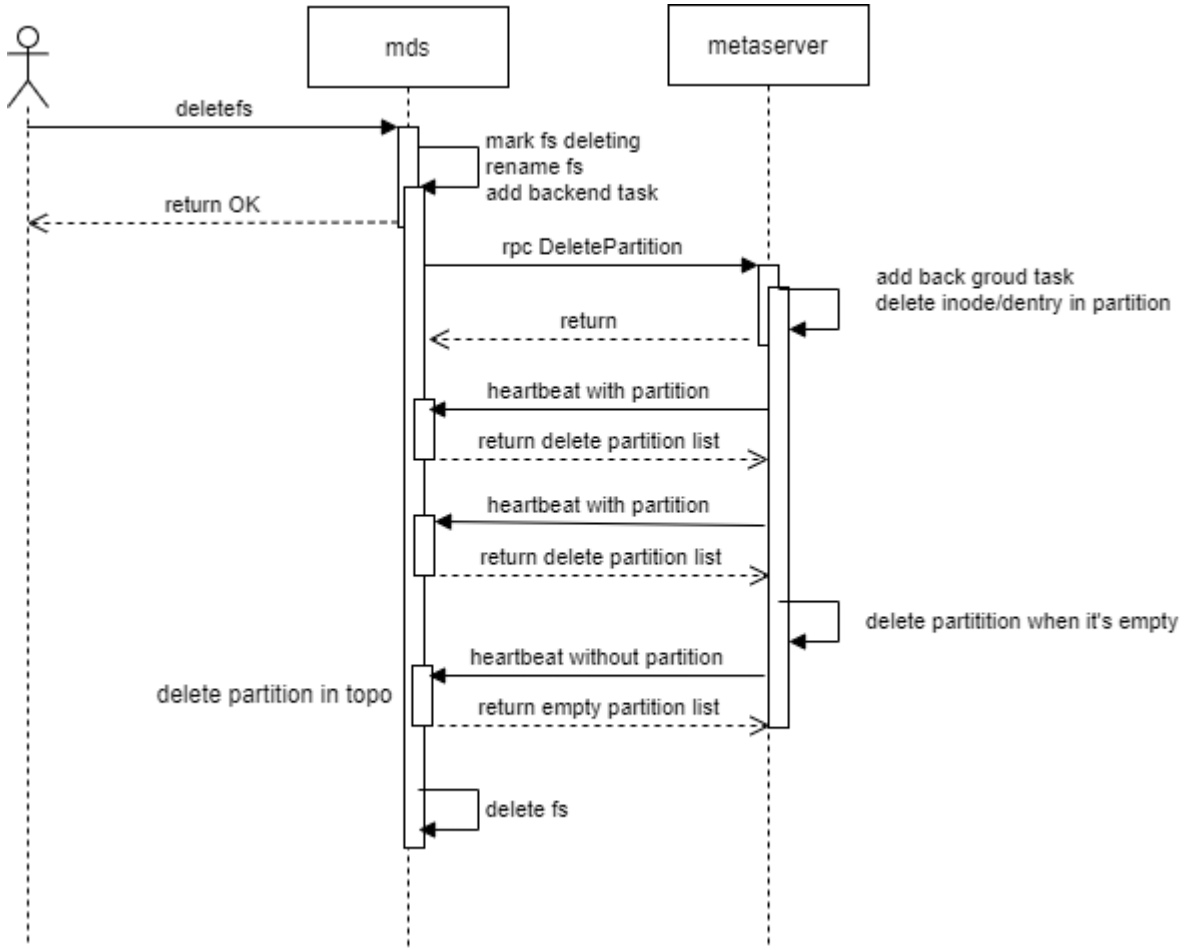
delete fs功能由运维工具提供命令，需要打印提示是否真的需要删除fs，且fs删除之后不可恢复，并需要用户手动输入类似“Yes, delete fs!”的校验信息。

1、整体方案：

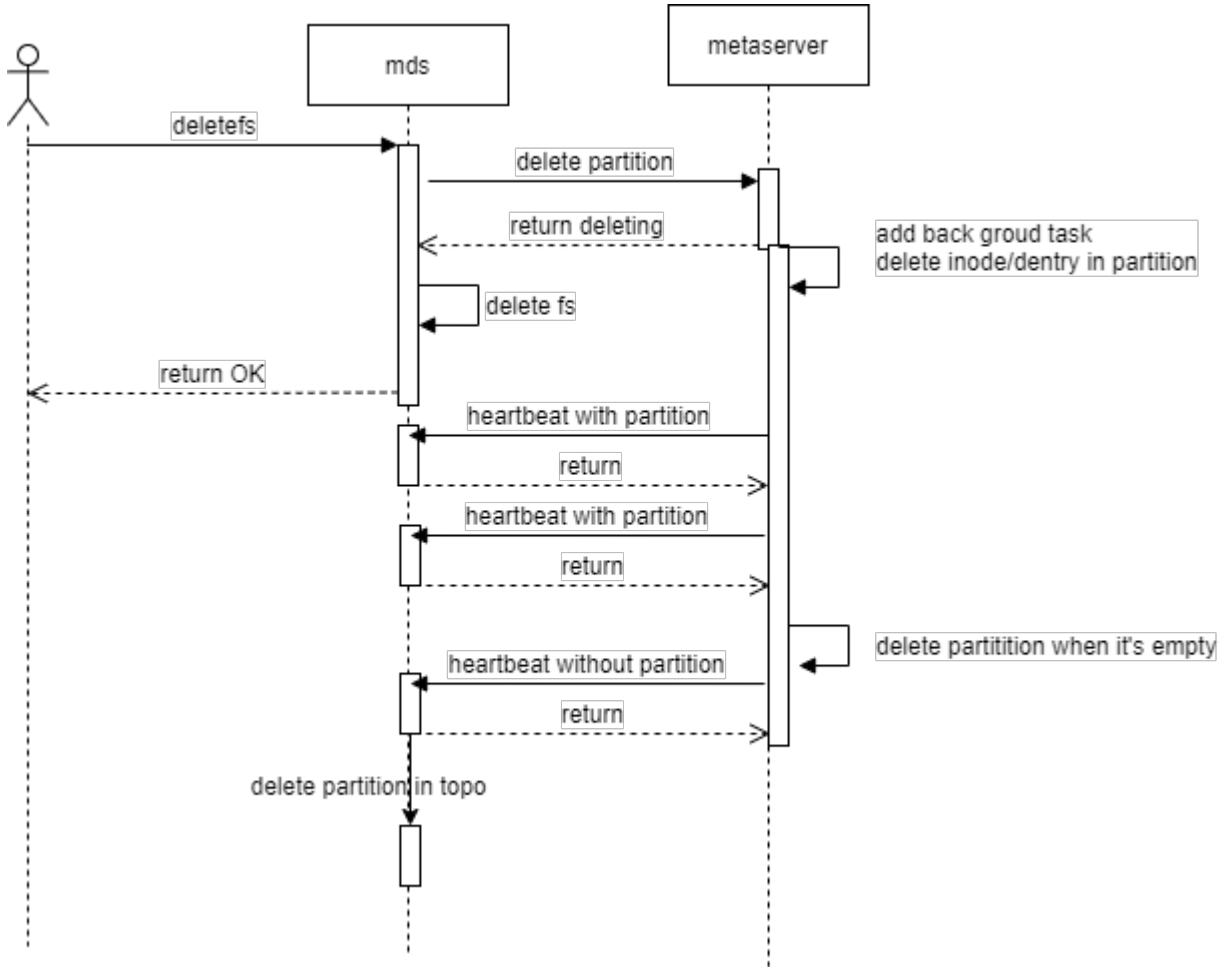
方案一：curvefs tool工具向mds发送删除fs的请求。mds收到请求之后，把fs标记为deleting并重命名为的 "旧名字" + "_deleting_" + "删除时间"，然后返回删除成功。mds心跳模块发送需要删除的partition列表到metaserver，metaserver后台增减删除partition任务。当partition删除完成之后，删除fs记录。



方案二：curvefs tool工具向mds发送删除fs的请求。mds收到请求之后，把fs标记为deleting并重命名为的 "旧名字" + "_deleting_" + "fsid"+"删除时间"，然后返回删除成功。mds后台任务发送删除partition任务到metaserver，metaserver后台增加删除partition任务。当partition删除完成之后，删除fs记录。（以下方案设计基于此方案）



方案三：curvefs
tool工具向mds发送删除fs的请求。mds收到请求之后，想metaserver发送删除partition命令，然后删除fs记录，返回删除成功。partition由metaserver后台删除，删除完成后，通过心跳通知topo删除mds的partition记录。



2、mds端修改

2.1 RPC DeleteFs 请求，删除fs的判断条件：

1、fs是否存在

2、fs是否还有挂载点

3、检查fs状态

如果状态为NEW / INITED，标记为DELETING，继续后面处理；如果状态为DELETING，直接返回UNDER_DELETING。

2.2 删除fs的处理过程：

1、fs状态标记为DELTEDING，把fs标记为deleting并重命名为的 "旧名字" + "_deleting_" + "fsid" + "删除时间"。

2、增加一个删除文件系统的后台任务，后台任务不需要进行持久化。后台任务也可以省略，起个后台线程，定期扫描所有的fs。

3、返回删除成功

2.3 deletefs后台任务处理

定期扫描所有后台任务，对每一个后台任务做以下处理

1、查询fs的所有partition信息，查询metaserver上partiton是否存在以及状态

2、如果partition状态不是deleting，向metaserver下发删除partition的命令，标记topology的partition状态为deleting

3、向topology查询partition的清除情况

4、如果所有partition都删除成功了，删除mds的fs记录

2.4 mds服务初始化相关处理

mds服务起来之后，加载fs信息的时候，如果fs是DELETING状态，生成删除fs后台任务

2.5 心跳相关修改

根据heartbeat心跳上报信息，如果topology的parition为deleting状态，且metaserver已经没有该partition，则删除topology的parition记录。

3、metaserver端修改

3.1 接口变化

DeletePartition已有proto定义。

curvefs/proto/metaserver.proto接口变化

```
message DeletePartitionRequest {
    required common.PartitionInfo partition = 1;

}

message DeletePartitionResponse {
    required MetaStatusCode statusCode = 1;
    optional uint64 appliedIndex = 2;
}
```

删除partition的时候，不用传入partition所有的参数，可以把接口修改成这样：

```
message DeletePartitionRequest {
    required uint32 poolId = 1;
    required uint32 copysetId = 2;
    required uint32 partitionId = 3;
}

message DeletePartitionResponse {
    required MetaStatusCode statusCode = 1;
    optional uint64 appliedIndex = 2;
}
```

delete partition接口目前没有调用，可以直接修改。

partition状态增加DELETING状态。

```
enum PartitionStatus {
    READWRITE = 1;
    READONLY = 2;
    DELETING = 3;
}
```

3.2 metaserver DeletePartition处理逻辑

首先，判断partition的状态。

case1 : partition状态为非DELETING

1、判断partition是否可以直接删除

2、如果可以直接删除，删除partition，返回OK

3、如果不能直接删除，partition状态标记为deleting

2、增加后台删除partition任务

3、返回partition删除中

case2: partition状态为DELETING

1、返回partition删除中

3.3 partition后台删除任务

具体实现：

为每个metaserver增加一个单例PartitionCleanManager。PartitionCleanManager扫描所有的partition，如果partition为deleting，在删除所有的Dentry，Inode之后，删除这个parititon。

1、删除partition的dentry信息，删除dentry不用走一致性协议，直接清理dentry即可。

2、依次删除所有partition下的inode（删除inode的时候，需要删除inode分配的数据存储空间）。对于每个inode，如下处理：

2.1 判断是否为leader，如果不为leader，不用处理。

2.2 先发送请求到s3，删除inode中保存的chunk信息。然后生成一个DeleteInode的请求，交给copyset经过一致性协议处理。

2.3 当前使用s3删除记录的接口一次只能删除一条记录，需要自己封装批量删除接口。参考aws的文档 https://sdk.amazonaws.com/cpp/api/LATEST/class_aws_1_1_s3_1_1_s3_client.html#ab4836eb38ad26b7402868acd8931a962。

3、当所有inode和dentry都删除成功，删除partition。metaserver的有一个trash map，保存着延迟删除的inode，如果inode被删除，trash map会认为的删除成功。新增加的partition清理inode不会影响trash的后台删除。

3.4 心跳的修改

metaserver端心跳不用修改，当前的心跳的会返回metaserver上的partition的信息，以及partition的inode和dentry的个数。curvefs tool可以根据心跳上报的信息查看partition的删除进度。

3.5 trash的修改

如果trash的partition为deleting状态，停止trash。

3.6 S3 compact的修改

如果partition为deleting状态，停止对partition的inode进行compact。

4、讨论点

4.1 fs要不要等7天再删除？

现在方案不支持等7天删除。建议后续如果有用户明确需要这个功能，再支持。

如果要支持等7天再删除，有几个问题需要解决。首先，在等待的时候，是否允许同名fs创建；其次，fs状态也要新增一个等待删除的状态；再次，fs等待删除的过程中是否需要重命名，如果有重命名，恢复的时候如何找到要恢复的fs……等问题都需要考虑。

讨论结果：不需要。fs的删除由管理工具发起，用户需要明确删除fs的后果，管理工具也会提醒删除fs的后果。如果确定要删除fs，fs就直接进行删除，不支持fs级别的找回功能。