skv客户端和服务端之间增加features negotiate机制(客户端设计)

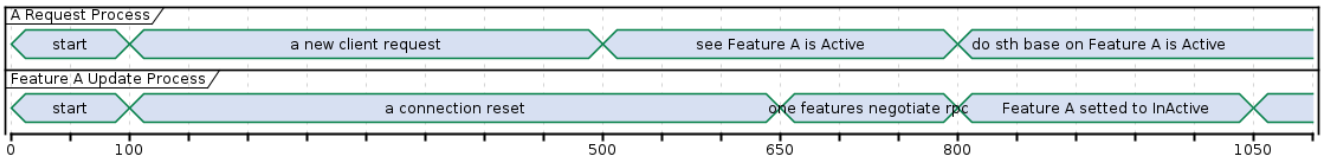
客户端侧实现的难点

分布式环境的易变性 和复杂性

- feature negotiate rpc会fail，需要重试，重试频率 可能也需要控制
- 多个版本的服务是有可能在一段时间共存的
    - 目前 pegasus是在一段很短时间内 会有两个版本(上线)
    - 未来(目前纯属个人yy)，如果 pegasus集群能做到 100~1000 个节点(主要是replica)那么
        - 上线时，不同版本共存的时间可能会很长
        - 实验需求， 只更新部分节点为最新版本

feature negotiate 功能本身 的界定问题

- 不同feature参与逻辑走向决策 的时候的"粒度" 可能会不同(也有点个人yy的意思)
    - "分片"粒度 (replica粒度)的feature ： 一个'batch' 的请求，在实际 逻辑中 会 拆成 对许多分片的 读写请求，每个分片的读写 请求 最终会 打向对应的replicaserver, 每个分片的读写流程可以根据 "分片 "粒度的feature 各自做相应的抉择
    - "global" 粒度的feature： 必须所有节点都支持这个feature才会走新逻辑， 否则就走旧逻辑
- 对强一致性的要求
    - 要求强一致性的feature
        - 在实际发送rpc给某个服务端之前发现服务端链接已经重置，并且这个链接另一头的服务端已然不支持这个feature，得报失败，或者整个请求进行重试
    - 不要求一致性的feature
        - 无所谓，随心随性， 比如batchGet3
            - 只有很短时间内一个集群会有两个版本
            - 这段很很短的时间内 batchGet3相关的rpc发送了低版本的server返回错误也还好，不会有严重后果



设计原则

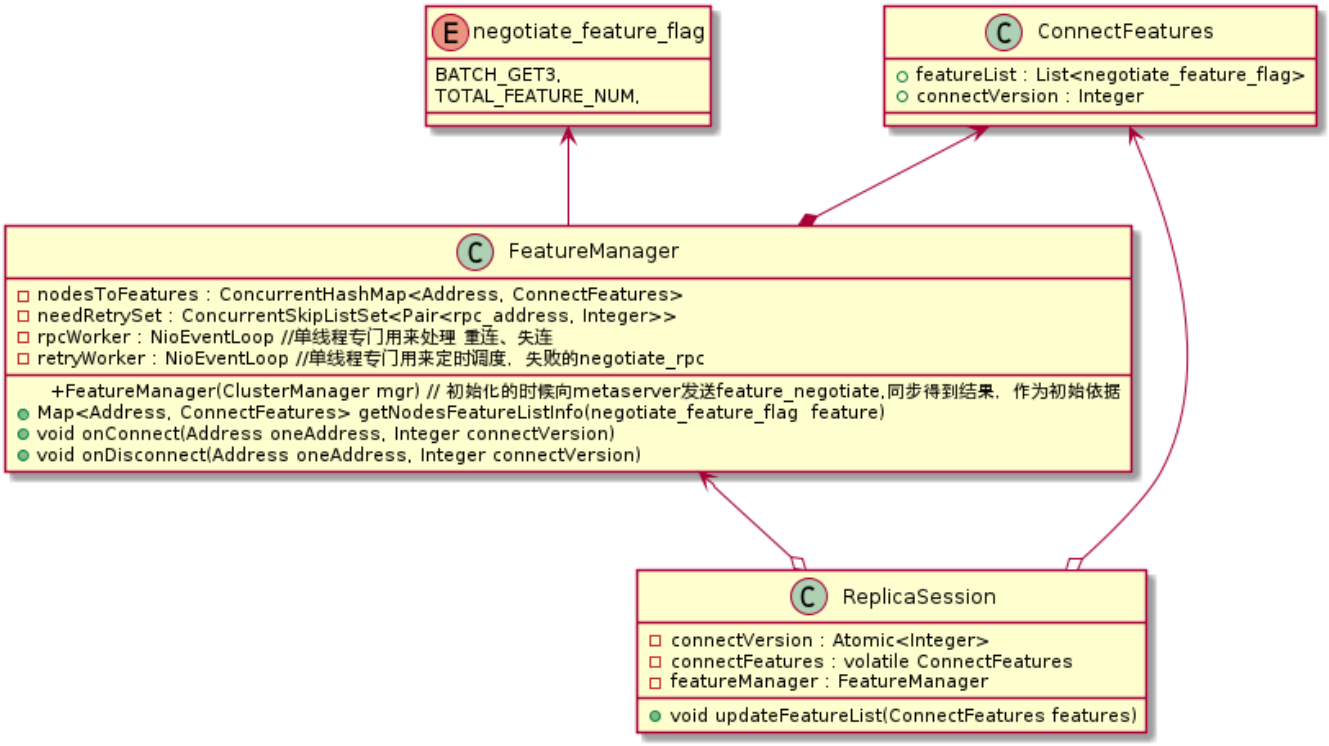
- 未来可能出现的东西未来再考虑， 不过度设计和实现
- 尽量在不过度增加代码复杂性 & 考虑自己目前码力和时间的情况下，适当考虑未来功能的延展性
- 支持并不复杂batchGet3 feature

功能设计

- 这一版不考虑feature的强一致性
    - 改动较大，短时间内快速实现可能 写出bug，死锁之类
    - 目前并没有功能 需求
- 客户端实现需要保存并能动态更新下列状态数据
    1. 每个节点 支持的feature 列表
    2. 尽量保证 上述数据的最终一致性
- 尽量simple和克制，只针对batchGet3场景

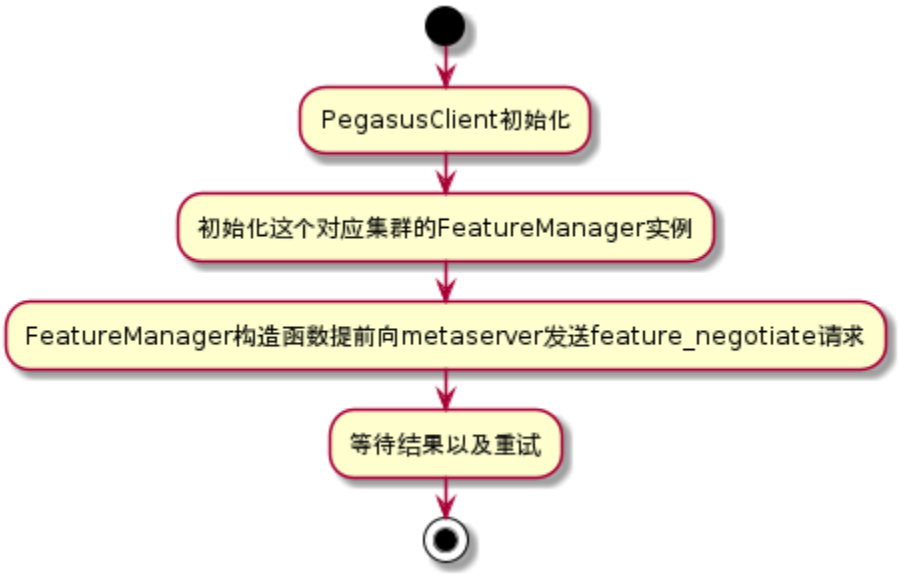
具体设计

类静态设计



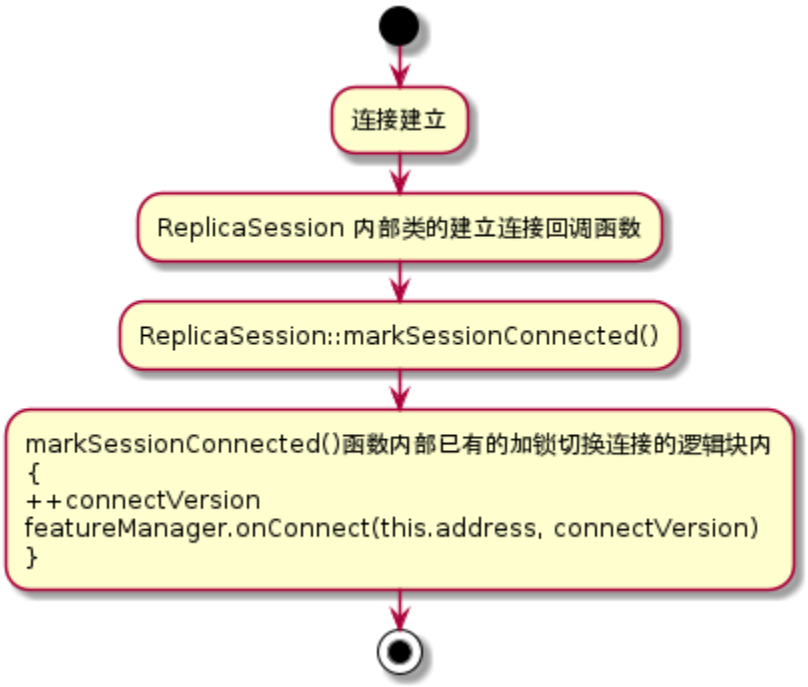
主要流程图

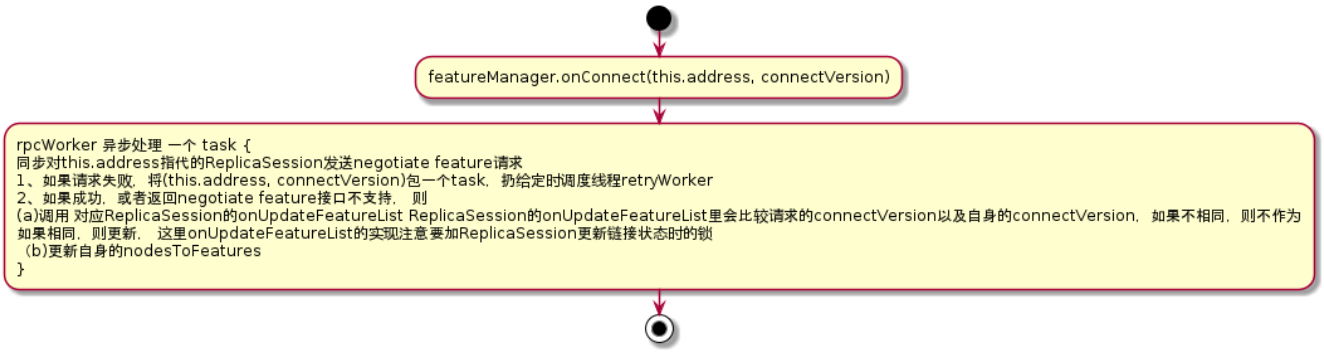
初始化



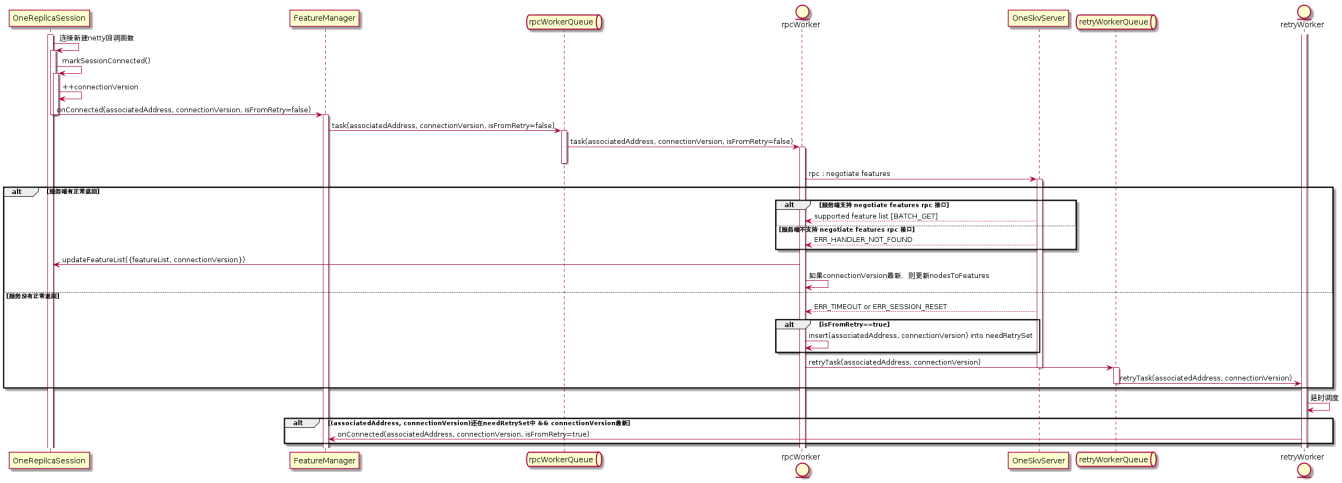
- 客户端初始化的时候只是被传入了metaserver-list, replicaserver节点客户端是不知道的，只有客户端调用openTable，打开了具体的表，客户端才会同步向metaserver调用rpc，拉取这个表对应的各个分片所在的replicaserver列表
- 我们这里初始化的时候可以同步调用negotiate feature rpc 也同步等待结果，至少知道了metaserver对feature支持的列表，作为初始依据
- 如果对于metaserver的simple请求经过一定重试都不ok(这里指超时或者拒绝链接； 服务端接收请求， 但是返回不支持negotiate接口等原因除外)， 那么 就直接挂掉算了， 说明集群不正常

连接建立

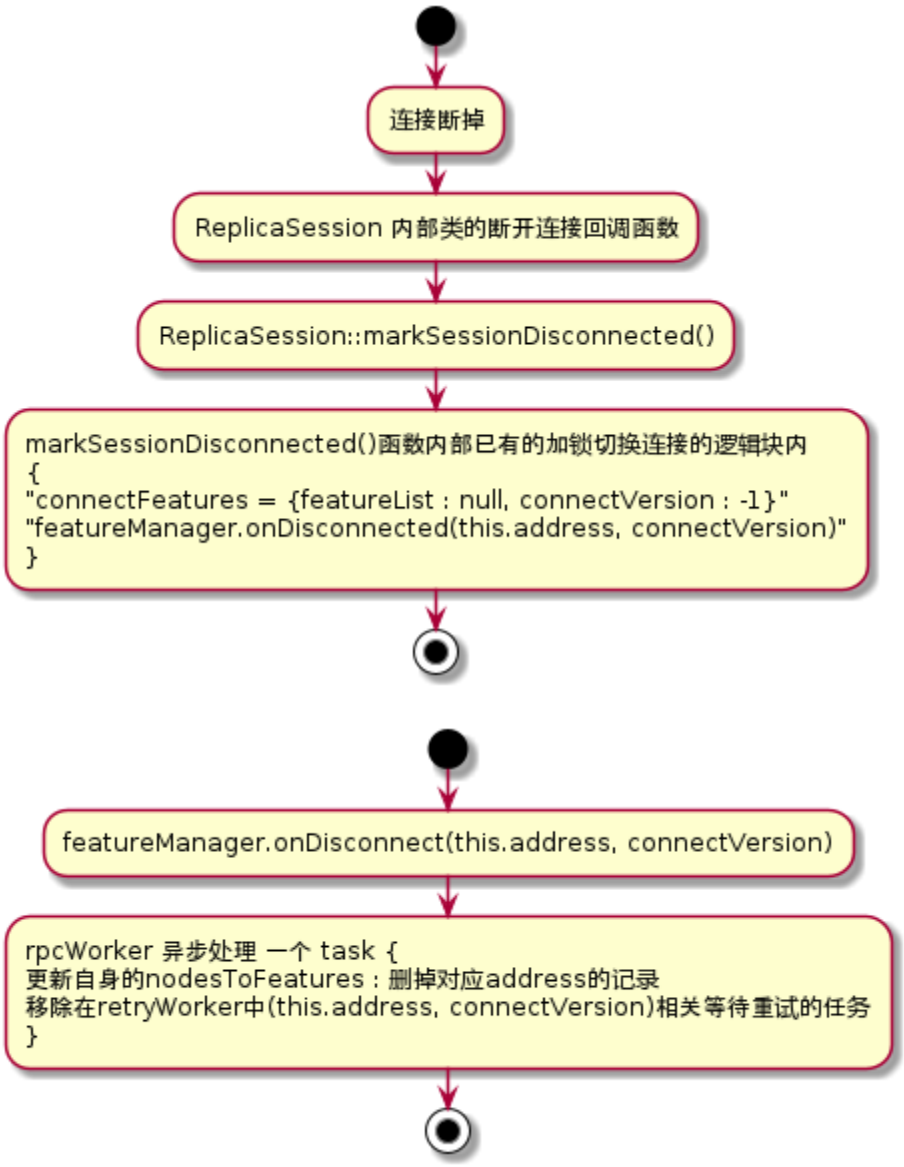

连接建立
ReplicaSession 内部类的建立连接回调函数
ReplicaSession::markSessionConnected()
markSessionConnected()函数内部已有的加锁切换连接的逻辑块内
{
++connectVersion
featureManager.onConnect(this.address, connectVersion)
}


featureManager.onConnect(this.address, connectVersion)
rpcWorker 异步处理 一个 task {
同步对this.address指代的ReplicaSession发送negotiate feature请求
1、如果请求失败，将(this.address, connectVersion)包一个task，扔给定时调度线程retryWorker
2、如果成功，或者返回negotiate feature接口不支持，则
(a)调用 对应ReplicaSession的onUpdateFeatureList ReplicaSession的onUpdateFeatureList里会比较请求的connectVersion以及自身的connectVersion，如果不相同，则不作为
如果相同，则更新， 这里onUpdateFeatureList的实现注意要加ReplicaSession更新链接状态时的锁
（b)更新自身的nodesToFeatures
}

时序图如下


OneReplicaSession
FeatureManager
rpcWorkerQueue
rpcWorker
OneSkvServer
retryWorkerQueue
retryWorker
连接前进netty层调用数
markSessionConnected()
++connectVersion
onConnected(associatedAddress, connectionVersion, isFromRetry=false)
task(associatedAddress, connectionVersion, isFromRetry=false)
task(associatedAddress, connectionVersion, isFromRetry=false)
rpc：negotiate features
alt
supported feature list {BATCH_GET}
ERR_HANDLER_NOT_FOUND
updateFeatureList({featureList, connectionVersion})
如果connectionVersion最新，则更新nodesToFeatures
ERR_TIMEOUT or ERR_SESSION_RESET
alt
insert(associatedAddress, connectionVersion) into needRetrySet
retryTask(associatedAddress, connectionVersion)
retryTask(associatedAddress, connectionVersion)
延时调度
alt
onConnected(associatedAddress, connectionVersion, isFromRetry=true)

连接断开



定时调度重试retryWorker

feature 是否 被 支持的判断

- 直接通过FeatureManager的接口 getNodesFeatureListInfo 获取 节点到feature列表的映射
    - getNodesFeatureListInfo的实现注意是深拷贝整个内部的map，否则会造成对象逃逸
- 不同请求和流程根据获得的节点-feature列表 映射自行判断取舍
    - 对于batchGet3只有4种可能
        - 存在节点明确不支持batchGet3
            - 降级到batchGet2
        - 所有节点不支持negotiate接口
            - 降级到batchGet2
        - 所有节点都支持batchGet3
            - 支持batchGet3
        - 部分节点支持batchGet3，部分节点不支持negotiate接口
            - 降级到batchGet2