cloudera

Ozone HA Overview

Bharat Viswanadham

Agenda

OM HA Architecture

OM HA components

Steps required to implement write requests in OM HA.

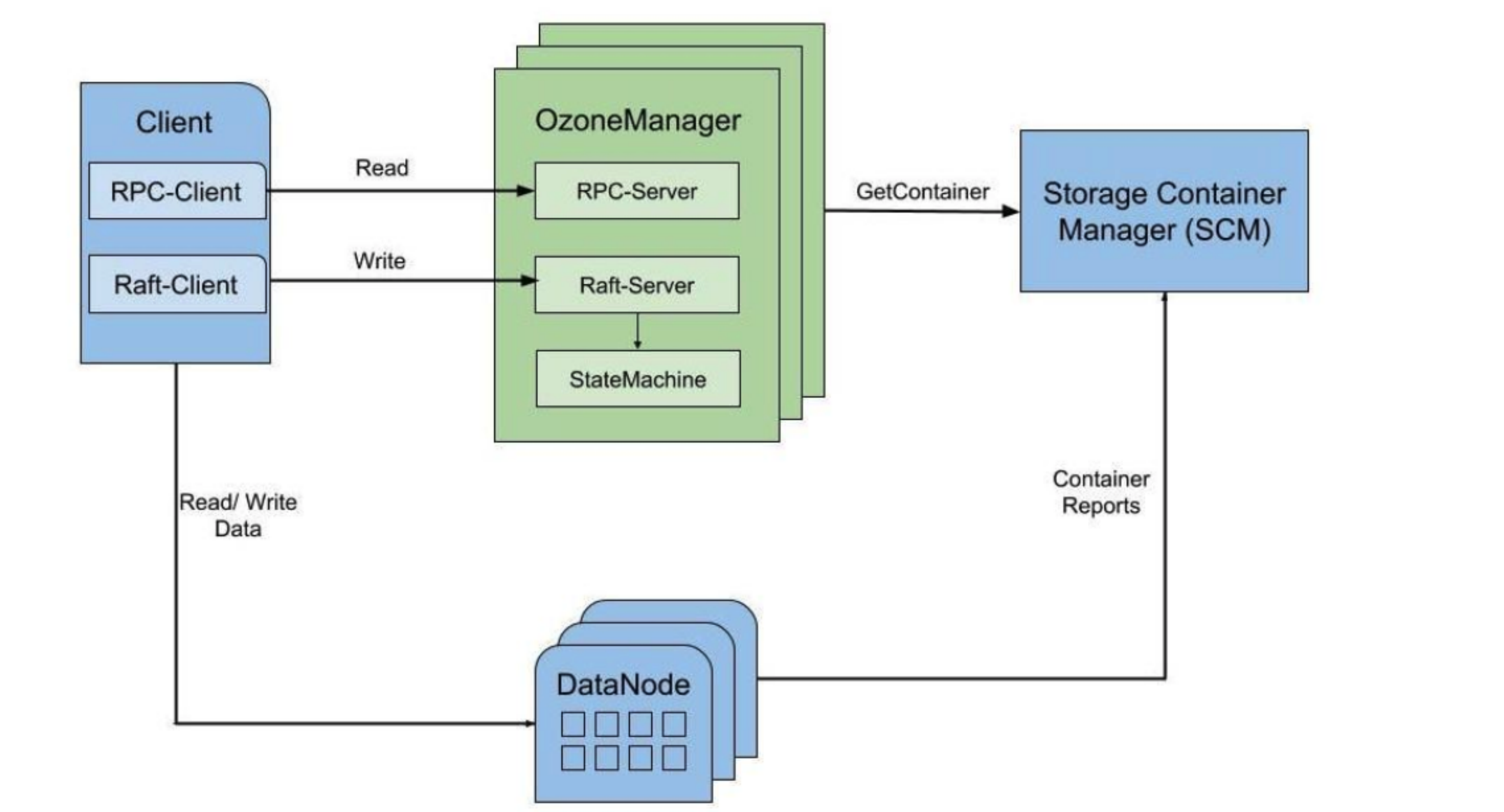
Handle write requests in OM HA

Sequence Diagram of write request handling in OM HA

Code walk through

Q & A

Ozone HA


Client
RPC-Client
Raft-Client
Read
Write
OzoneManager
RPC-Server
Raft-Server
StateMachine
GetContainer
Storage Container Manager (SCM)
Read/ Write Data
Container Reports
DataNode

Ozone HA Components

- OM Ratis Server
  - Each OM node starts Ratis Server. Ratis Server API is used to submit request to Ratis.
- OM State Machine
  - Ratis after appending transaction to log, will call State Machine to apply the transaction by calling applyTransaction.
  - In apply Transaction actual logic for writeRequest will be executed.
- Table Cache
  - Each Table has cache, which is an in-memory map to store the table info.
  - Cache has Eviction policies. Current eviction policies supported are NEVER, CUSTOM.

Ozone HA Components

- Double Buffer
  - Each OM node will have double buffer, when write request completes its execution it adds it’s entries to table cache.
  - Double Buffer maintains two buffers.
    - ReadyBuffer
    - CurrentBuffer
  - Double Buffer has a daemon thread, checks if there are any entries in its currentBuffer, swap buffers and flush the entries to DB.

Steps required to handle write requests in OM.

- Developer need to implement two abstract classes.
    - OMClientRequest
    - OMClientResponse

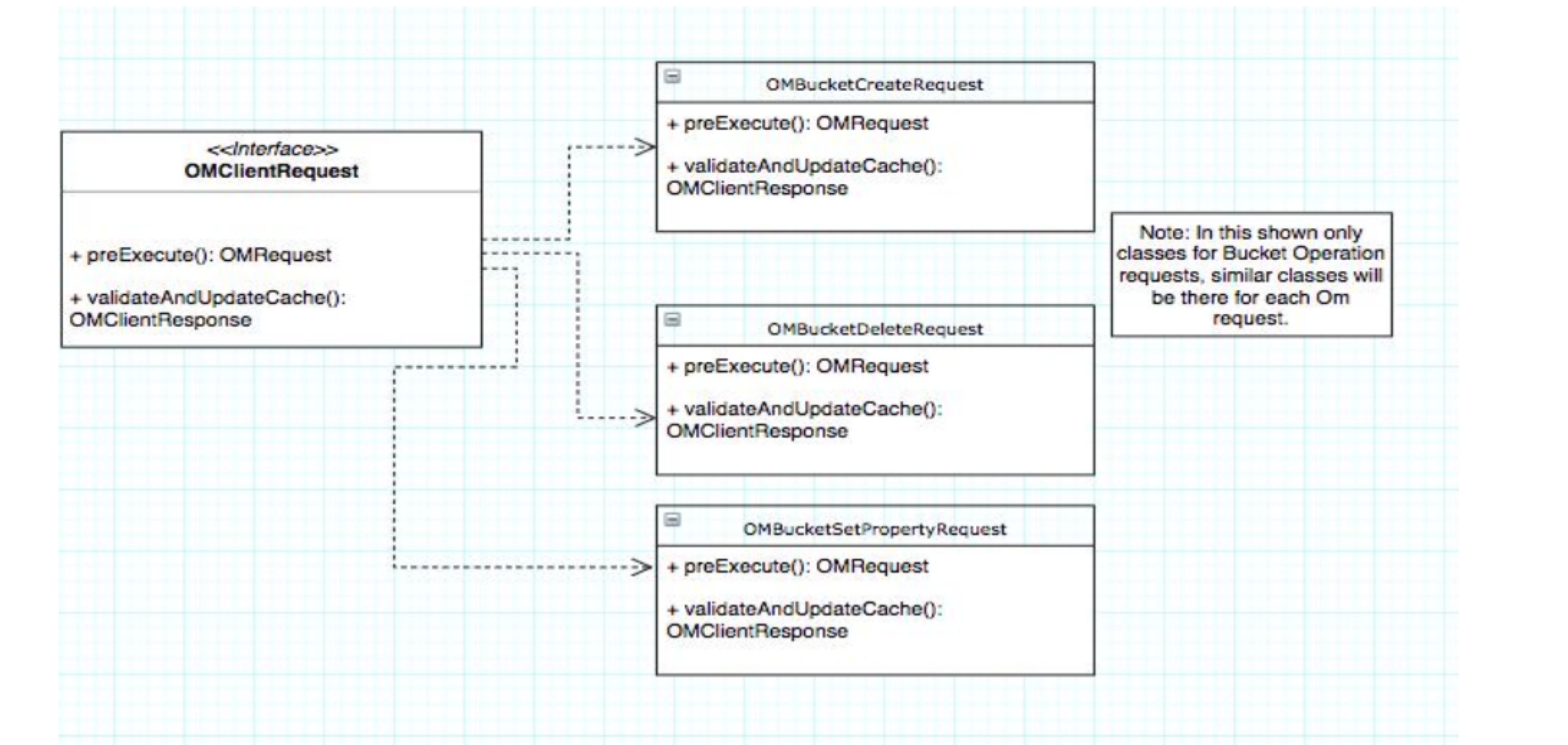
OMClientRequest :

- PreExecute.
    - Perform steps need to be happen on only leader OM.
        - Like allocating blocks,
        - Setting creation/modification time.
        - Set userInfo.
- ValidateAndUpdateCache.
    - Request validation, and update the cache.

OMClientResponse:

- addToDBBatch.
  - Implement the logic to add the response to DB batch.

Class Diagram showing OMClientRequest for Bucket requests



<<Interface>>
OMClientRequest

+ preExecute(): OMRequest

+ validateAndUpdateCache(): OMClientResponse

OMBucketCreateRequest

+ preExecute(): OMRequest

+ validateAndUpdateCache(): OMClientResponse

OMBucketDeleteRequest

+ preExecute(): OMRequest

+ validateAndUpdateCache(): OMClientResponse

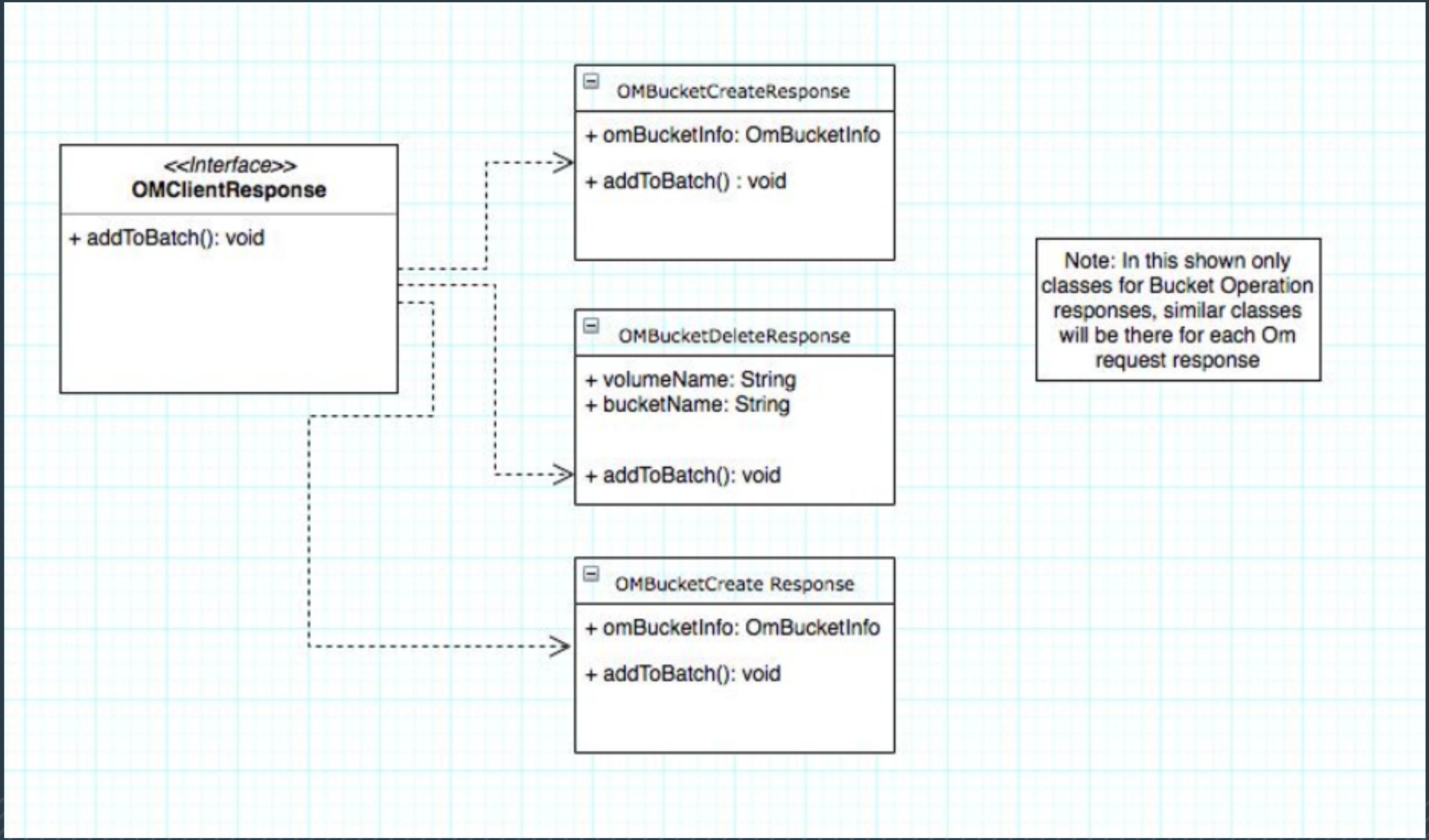
OMBucketSetPropertyRequest

+ preExecute(): OMRequest

+ validateAndUpdateCache(): OMClientResponse

Note: In this shown only classes for Bucket Operation requests, similar classes will be there for each Om request.

Class Diagram showing OMClientResponse for Bucket requests



<<Interface>>
OMClientResponse

+ addToBatch(): void

OMBucketCreateResponse

+ omBucketInfo: OmBucketInfo

+ addToBatch() : void

OMBucketDeleteResponse

+ volumeName: String
+ bucketName: String

+ addToBatch(): void

OMBucketCreate Response

+ omBucketInfo: OmBucketInfo

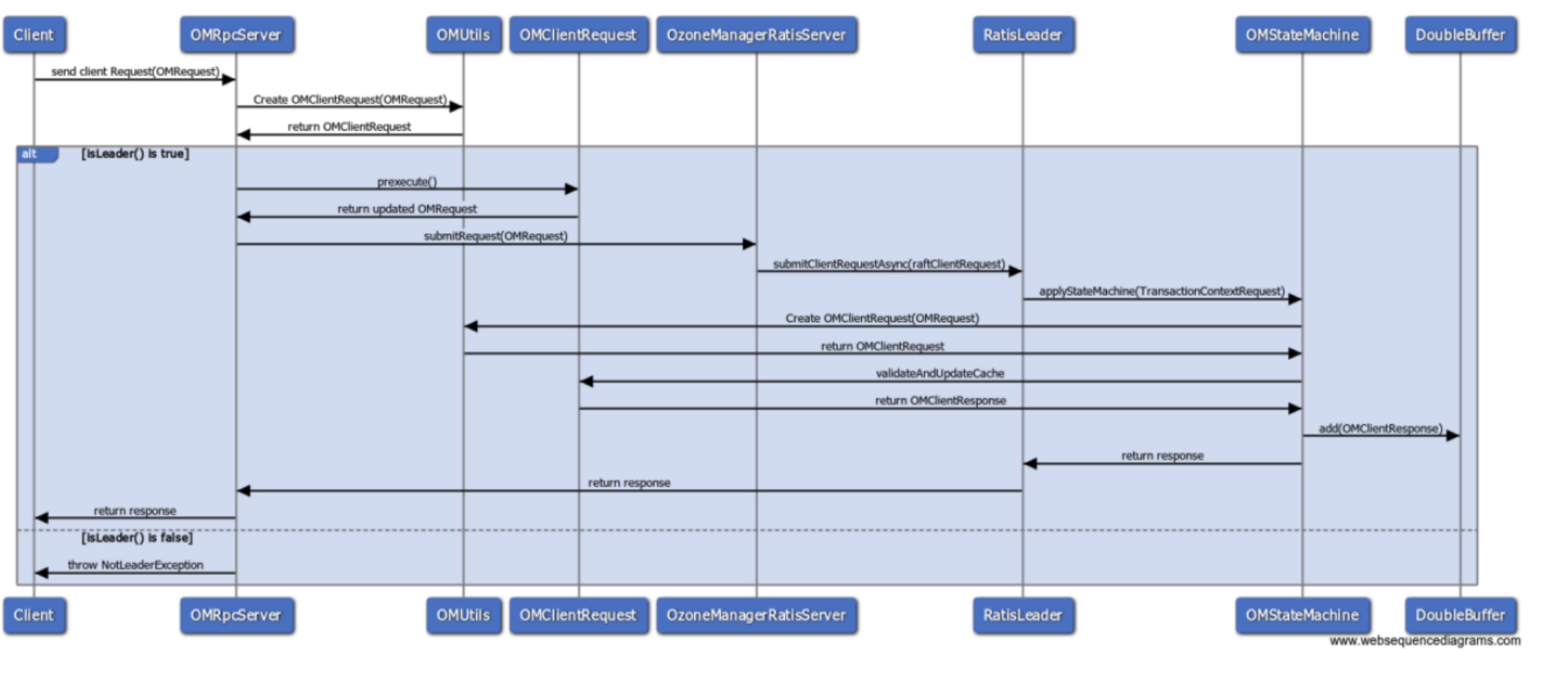
+ addToBatch(): void

Note: In this shown only classes for Bucket Operation responses, similar classes will be there for each Om request response

How write requests are handled in OM HA

- Request is received from Client through Rpc.
- Once request is received, check if isLeader(), and perform preExecute steps like allocating block, setting creation/modification time and set userInfo.
- After that submit request to ratis.
- After ratis got quorum, and write entries to log, it calls stateMachine to apply transaction.
- In apply transaction we perform validation, execute actual request, add the response data to cache.
- Once we get the client response, we add the response to double buffer to flush to DB.

Flow of write request in OM



cloudera