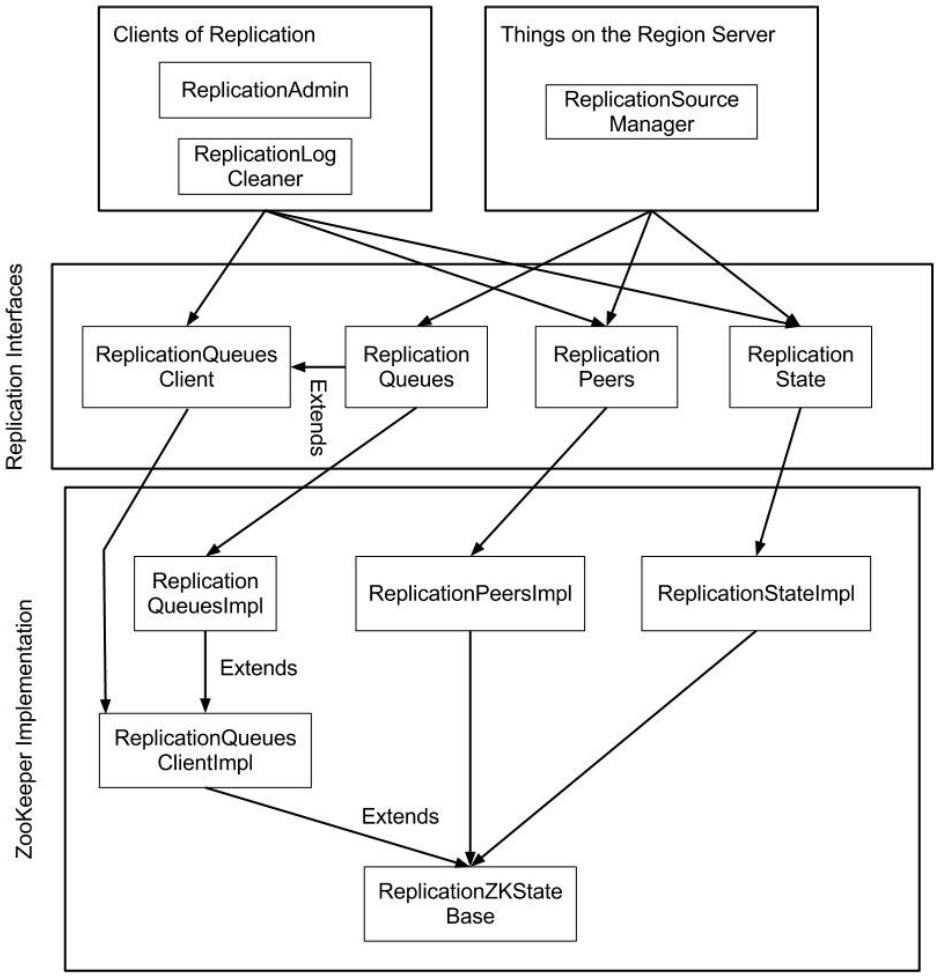
Replication Refactor

Motivation and goal

Currently, the ReplicationZookeeper class contains most of the logic for maintaining the state of replication (status, peers, queues). The class has grown organically and is now complex and slightly confusing. There are also multiple types of users that leverage the class, and it implicitly provides multiple interfaces. Finally, the implementation details of how replication state is maintained (i.e. zookeeper) have leaked up into higher-level parts of the code such as the ReplicationSourceManager. The goal of this refactor is to break up the complex logic of ReplicationZookeeper into smaller, more digestible parts and to provide an interface/abstraction for the implementation details of maintaining replication state. This will lend itself to a more approachable code base, and more flexibility for future implementations of replication state management.

New class structure



Interfaces

There are four new interfaces:

- ReplicationState - Responsible for maintaining the status of replication (i.e. whether it is enabled or disabled).
- ReplicationPeers - Responsible for maintaining the peers (i.e. remote slave clusters) that this cluster needs to replicate data to.
- ReplicationQueues - Responsible for maintaining a region server's replication queues.
- ReplicaitonQueuesClient - A read only interface that clients can use to view the content of the replication queues. This is a cluster wide view that is used by things like ReplicaitonLogCleaner and ReplicationAdmin.

Implementation

The ZooKeeper implementation of managing replication state has been divided into several classes that somewhat mirror the interfaces. Previously, almost all of this logic was inside of ReplicationZooKeeper, with some of it being leaked up to ReplicationSourceManager. The new structure is as follows:

- ReplicationZKStateBase - A base abstract class that has common things needed by all state implementation classes (i.e. setting up replication znode configuration).
- ReplicationStateImpl - Contains the status tracker and manipulates the atomic boolean that acts as the on/off switch for replicaiton.
- ReplicationPeers - Add/removes peer clusters and maintains a list of them.
- ReplicationQueuesClient - Contains read only methods at the cluster level. This interface is used by processes that only need to view the contents of the replication queues. This interface was split out to prevent clients from inadvertently calling methods that impact the set of replication queues.
- ReplicationQueues - This interface extends the client interface and is used at the region server level. This class contains all the logic to add/remove queues, add HLogs, and failover queues from a dead region server.