Branden Siegle
Gabriela Hernandez

In this design, the nodes associate themselves to two consecutive keys in the KV Service, where they keep their nodeID and their availability status stored. By limiting all keys to be a number from 0 to n, we were able to assign the leader role to be the first two available keys in the KV Service. Apart from this, each node keeps a copy of the KV Service locally, in a map, which it uses to spot changes in the KV Service so that it can act accordingly. To further clarify the leader election process, we will refer to two consecutive keys (ei k1 and k2, k3 and k4, etc) as a "node key".

As a node joins the service, it associates itself to a node key in the KV service. After doing so, it iterates through the KV service, referencing its aforementioned map in order to spot any changes and to update its map values. Once a change is spotted (ie. a node is added, a node fails, etc) it prints out an active-node list that contains the change. After it does this, the node will check to see if it is the current leader as per the leader selection criteria outlines below. If it is, it will change its status in the KV service from "active" to "leader" in order for non-leader nodes to know it is indeed still connected. It will keep doing this until the KV service replies with an "unavailable" response, in which case it will find a new node key to associate itself with, and the next active key will become leader. Our method for leader selection was to have the lowest most active node in the KV service be the leader node and the second lowest be responsible for monitoring the status of the leader. If a node is responsible for monitoring the status of the leader node it will change its status from "Leader" to "Test x" where x increases with each check. While this is occurring, each time the leader rechecks if it is still the leader it will change its status back to "Leader." We assume that if after five checks the leader node hasn't changed its status it must have gone offline so we indicate it as such. To handle the case where the node hasn't failed but may have just been really slow to respond, the old leader will just reconnect to the KV service as a normal node.

To handle keys becoming unavailable, we made it the responsibility of the leader to move the node associated with that key over to another key that is available. If the key associated with the leader node fails, the next node will reassign the node to another key.

Our clients never explicitly communicate with each other except through the statuses and positions in the KV service.