Every server, whether a replica or failure detector, pings every failure detection service once a second. When a machine wants to ask whether or not something has failed, it queries each failure detector, which votes that it has failed if the last seen ping occurred more than five seconds ago. The querying machine then takes a majority of the decisions.

Since our solution relies on pings, we have chosen times of sending pings every second, and deciding failed jvms if the time from last ping exceeds five seconds. Since all transactions begin from a GUI, these won’t be happening a particular high rate; we believe one second is more than sufficient. Further, we felt that lost pings are a considerable concern; thus, we vote that a jvm has failed only if multiple concurrent pings are never received.

A case that will lead to an halt in the network is if n/2 nodes have failed, where n is the number of jvms initially launched. Since a client has no reason to ask the failure detection service if anything but its branch’s primary have failed, then if the primary fails after n/2 nodes have failed, the client cannot reach a majority. Similarly, if a branch does not receive any transactions from its client while n/2 nodes fail, then it also cannot obtain a majority on the next transaction. However, neither of these cases lead to an inconsistent state, and require a at least half the network to fail, so we find them acceptable.

In our architecture, there is one failure detection service that all servers query. Each service contains n different failure detection servers, where n is the number of jvms started. Each failure detection server contains n sensors. If a sensor does not receive a ping from a server, the server it is located on is notified the next time it receives a query. If any failure detection server thinks a node has failed, it assumes that everything running on that node’s processor has also failed.

We define “work correctly” to mean that we will never reach an inconsistent state with regard to transactions. If a transaction cannot be replicated to all backups, then it is not replicated, and the transaction is considered failed. Only if all replicas in a branch allow a transaction will it occur.

Our system will work correctly since a majority is needed every time the failure detection service is queried. If a single sensor thinks something has failed, it can continue to vote that it has without affecting the entire service’s decision.

We assume that colocated on every server is a replica of a branch, be it the primary or a backup, and a failure detection server.