CS 425/ECE 428
Distributed System

MP1 Report

Group Member:

Name: Rui Xia
NetID: ruixia2
UIN:660935771

Name: Youjie Li
NetID: li238
UIN: 668129373

Spring 2017
February 25, 2017

Design

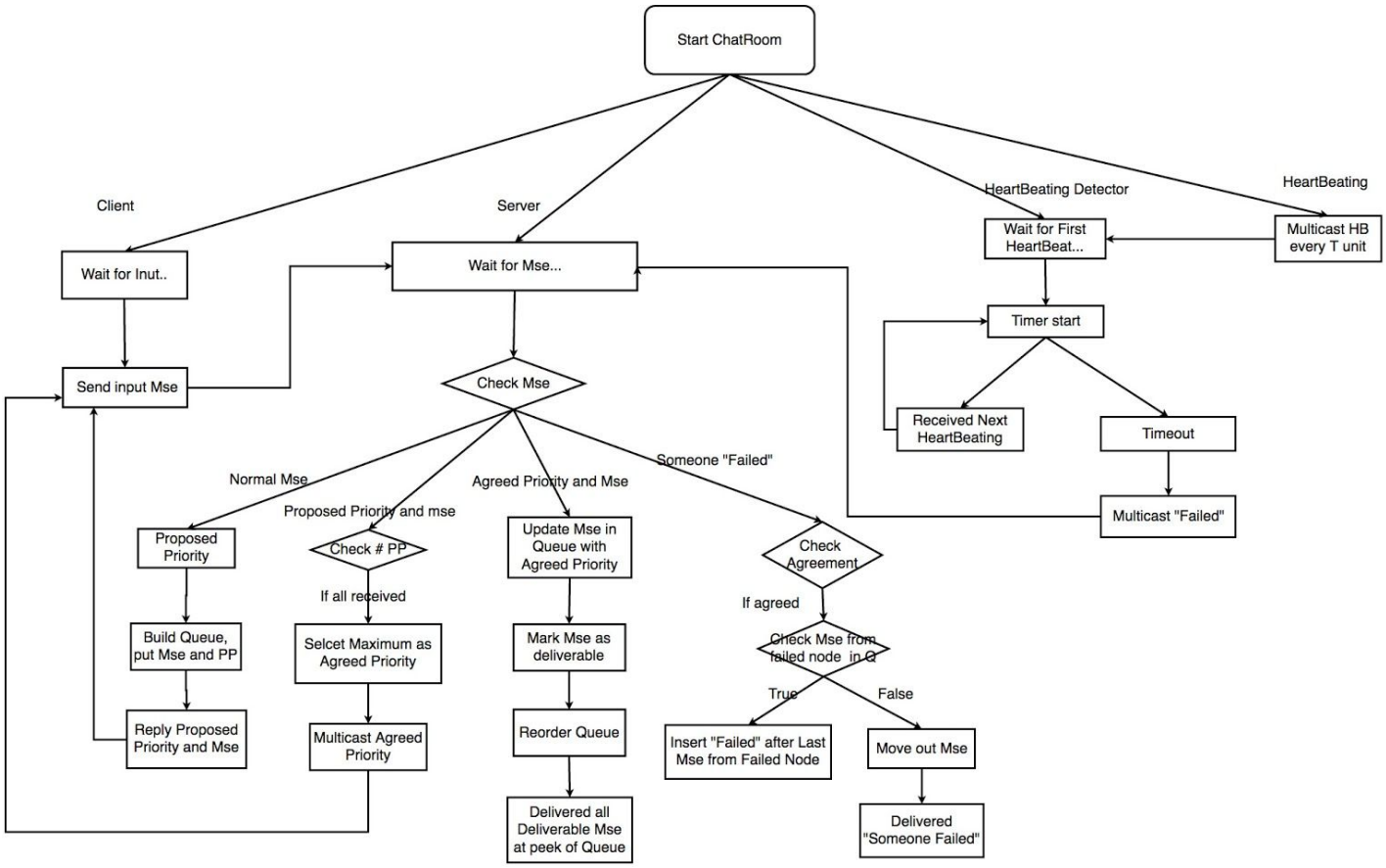

Figure 1 Flow Chart for Code Design

As shown in Figure 1, 4 threads will run synchronously to implement all functions for the Chat Room, two of which are designed to implement a total ordering reliable multicast while others designed for an all to all Heart Beating Failure Detector.

- Client: Multicast input message to all nodes in the chatroom
- Server: Receive message and deliver in order.
- HeartBeating: send heartbeating for failure detection
- Detector: Wait for heartbeating, detect and broadcast failure

Algorithms:

- ISIS for Total Ordering:
  The reason why we choose ISIS is that it is more distributed compared to the centralized sequencer algorithm, which is unreliable because of the possible failure of leader and the waste of time for leader election.
- All to All Heart Beating for Failure Detection
  The reason why we choose All to all HeartBeating is that heartbeating will occupy less bandwidth compared to ping act while all to all have a better completeness.

Details:

- For ISIS, A list of information including priority for any message will be stored in a Queue. The element of Queue in our algorithm is designed as [proposed_priority, deliverable_flag, message]
- The proposed_priority is a float number with priority as digit and process ID as fraction.
- We inserted "failed message" after the last deliverable message from failed node in Queue to verify that any messages sent by the failed node always are delivered **before** the failure notification
- For All to all HeartBeating, we design a branch thread for detector as a timer. If timeout without receiving next heartbeating, the branch thread will multicast failed message.

Parameters:

- Period of Heart Beating: T= 4s
- Timeout: T + delta = 8s
- user_port=9999
- fail_detect_port=8888

Metrics

Supposed there is N nodes in a cluster:

- Bandwidth of failure detector when no failure happens:
  (N-1) msg every T units (per node)
  N*(N-1) msg every T units (whole systems)
- Failure Detection Time:
  In theory, it is (T+2*max-one-way-delay).
  In practice, it is (Timeout + max-one-way-delay).

Testing Approach

We have setup the experimental environment to test our chatroom, which includes NTP synchronization among all virtual machines and simulated network delay of random value from 50 to 150ms. Results are as follows:

- We have tested multiple independent failures in the system and here are the results.

Experiment #	Failure Detection Time
1	8968.06001663 ms
2	9489.95614052 ms
3	9747.03502655 ms

4	8885.94698906 ms
5	9397.81904221 ms
6	9106.24504089 ms
7	9070.58596611 ms
8	9094.5789814 ms
9	8585.95585823 ms
10	9019.06394958 ms

Average Failure Detection Time: 9136.524701 ms
95% Confidence Interval: [8946.524701, 9326.524701] ms

As can be seen, the actual failure detection time is larger than the theoretical value. It is because the timeout checkings in the process are delayed to speedup process execution. If the timeout-checkings were invoked too frequently, the failure detection time might would be shortened but the execution speed of the entire process would drop significantly.

- Bandwidth of failure detector (when no failure). We have evaluated the heartbeat bandwidth in the system. After 10 min, each node posts its own heartbeat bandwidth in average value and the summation of all nodes shows the system bandwidth.

Average Bandwidth per-node	1.509475 msg/sec
95% Confidence Interval	[1.507575, 1.511375] msg/sec

Average Bandwidth in system	12.0758 msg/sec
95% Confidence Interval	[12.0606, 12.091] msg/sec

As can be seen, the actual bandwidths are smaller than theoretical values as well. The reason is the same as mentioned above. i.e. The timing-check for heartbeats is delayed to speed-up process execution so the effective heartbeat period is enlarged.

Future Work

We came up with an hybrid failure detector to increase the accuracy. The new detector is based on the all to all heartbeating but also involves a Ping-Ack detection after heartbeat detects the potential failure. This additional Ping-Ack is to verify the result of the heartbeat. As a result, the accuracy of the failure detection is enhanced.