Report 4: Groupy

Nikita Gureev

September 29, 2016

1 Introduction

In this homeworka group membership service that provides atomic multicast is implemented. The aim is to have several application layer processes with a coordinated state i.e. they should all perform the same sequence of state changes.

2 Main problems and solutions

The main problems that were encountered during this task were the implementation of leader election procedure, handling the failure of leader during broadcast, and failure in message delivery. The first problem was solved using an assumption of a list of peers, first peer after the leader becomes the new leader in case of leader failure. The second problem was solved using the same assumption. If the leader dies during the broadcast the new leader is the peer that will be responsible for resending of messages to others. The last problem was the most complicated and has the most impact on the performance. It will be discussed in Evaluation section in detail.

3 Evaluation

The first problem of leader election is the most simple to resolve. In this implementation a simple approach was chosen, where from a list of peers the first one is chosen as a new leader during election. The simplicity of the procedure allows us not to spend much resources on leader election and enables a simple a robust way to maintain a leader.
The second problem was more difficult as it deals with a leader failure during message broadcast, which means that some of slave nodes were not notififed of the state change. The solution to this problem can be naive, but it dramatically increases the number of messages being sent. A more sensible approach was chosen, when only the new leader boradcasts the new message to all of the slaves after it in the list of peers. This solution is better, as every node does not send the last message to every other node. Also,

duplicate message handling was added in order not to change the state unnecessarily.
The last problem is the most complex and bears the most impact on the performance of the system. With the possibility of message delivery failure we need to introduce a way to acknowledege message receiving. It decreases the system performace, a the leader now waits for confirmation of the previous state change and resends the state to nodes that did not reply. In an event of node failure it will make the system much less efficient. It also introduces the risk of leader and node failure, in which the system may fall out of sync. The solution to that problem is to store more messages not only on the leader node, but also replicate them onto slave nodes, and in such an event the new leader can determine the point of out of sync and remedy it. It also further decreases performance and introduces costs to memory, as more information about system state needs to be stored. Moreover, the state of replicas should also be managed and it introduces further complexity and performance decrease. Overall, the trade off between reliability and performance is clearly visible, as the more reliable the system gets, the more is performance decreased.

4 Conclusions

In conclusion, the group communication is essential in today's distributed systems is widely used. Not only the states of nodes need to be kept in sync, but the whole concept of data replication, which is used in most of the systems, is built upon it. This homework builds upon the simple example and then introduces different real life challanges that complicate the system and lessen its performance, but increase its reliability. As in many real life situation, the trade off between reliability and performance should be resolved individually for each and every system.