README – ÜBUNG 2 Viktor Werle 3603083

Systemvoraussetzungen:

- mindestens Python 3.5
- ZEROMQ Library

Umsetzung

Als Prozesse dienen die bisherigen Knoten. Jeder Knoten verwaltet seine eigene Queue. In der Queue werden Lamport-Zeit und Prozess_ID (in unserem Fall die Knoten-ID) gespeichert.
Beispiel:
Queue: [[1,0],[1,1],[2,0], ….]
Als Queue wurde eine PriorityQueue gewählt. Damit wird gewährleistet, dass alle Einträge immer korrekt sortiert sind.

Beim Start verschickt jeder Knoten jeweils einen Write-Request mit der Lamport-Zeit und der Prozess_ID an alle übrigen Knoten. Die Knoten bestätigen den Empfang. Alle Wirte-Requests landen in der Queue. Durch die Sortierung der Queue (nach Lamport-Zeit und PID) ist es gewährleistet, dass am Kopf der Queue der Prozess mit kleinster Lamport-Zeit und PID steht.

Sobald der Knoten alle Bestätigungen erhalten hat, prüft er die Queue ob er an der ersten Stelle in der Queue steht. Ist es der Fall, dann schreibt er in die Datei, löscht den Eintrag aus der Queue und schickt RELEASE sowie erneuten Write-Request an alle anderen Knoten. Beim RELEASE entfernen die Knoten das erste Element aus der Queue und prüfen ob sie an der Reihe sind.

Terminierung
Hat ein Knoten dreimal den Wert 0 gelesen, schickt er an seinen Nachbarn und sich selbst die Anforderung zu terminieren. Bevor der Knoten terminiert, informiert er alle übrigen Knoten, dass er nicht mehr verfügbar ist.
Ein Knoten, welcher solche Nachricht bekommt, entfernt aus der Queue alle Prozesse von dem terminierenden Knoten.

Programmstart

Zum starten des Programm muss lediglich der Skript node_starter.py in dem Verzeichnis src/ ausgeführt werden. Die Datei ts_id.txt in welche alle Prozesse schreiben liegt im Verzeichnis conf/. Die Anzahl der Knoten kann in der Datei conf/settings.txt eingestellt werden. Zum Übernehmen der Einstellungen muss der Skript src/create_config_files.py ausgeführt werden.