CS/CE/TE 6378: Project III

Instructor: Ravi Prakash

Assigned on: April 14, 2016
Due date and time: April 29, 2016, 11:59 pm

This project is to be completed in groups of two students, and both members of the group are expected to be present during its demonstration to the TA.

1 Requirements

1. Source code must be in the C /C++ /Java programming language.
2. The program must run on UTD lab machines (`dc01, dc02, ....`).

2 Total Ordering of Appends

This project is an extension of Project 2. Here are the departures from Project 2:

- Instead of using Maekawa's algorithm for serializing updates/appends, a variation of two-phase commit is be employed as follows:
  1. One of the three servers is designated as the leader, in advance.
  2. A client sends its append request (along with relevant data) to any of the three servers, selected randomly.
  3. The selected server buffers the data to be appended, and sends a REQUEST message with the data to the two other servers, and waits for their response.
  4. The other servers, on receiving the data, buffer it and send an AGREED message if they are willing to perform the append (this is the default response for the purpose of this project).
  5. When the selected server receives AGREED messages from both the servers it sends a COMMIT REQUEST to the leader.
  6. The leader serializes the COMMIT REQUESTs based on the order in which it receives them. For each COMMIT REQUEST, the leader appends the corresponding data to the file, and sends a COMMIT message to each of the other servers.
  7. The other servers, on receiving the COMMIT message from the leader, also append the corresponding data to their respective copies of the file and send an ACK to the leader.
  8. When the leader receives ACKs from all the other servers it sends an ACK to the client.
- A client can have one outstanding append request at a time.
- Make sure that all the servers perform the appends in exactly the same order at all three servers.
- Each client goes through the same sequence of requesting appends as it requested writes in Project 2.

3 Data Collection

Report the following:

1. The total number of messages exchanged.
2. For each client, log the following (in a file) for each of its attempts to append data:
   (a) The number of messages of each type exchanged between the client and servers, and among the servers.
   (b) The elapsed time between making a request to append data to a selected server and receiving an ACK from the leader.

Do you notice something interesting about the frequency of some of the message types? If so, would these messages be more or less likely to be generated if the time between generating successive WRITEs were to be reduced?

The submission should be through eLearning in the form of an archive consisting of:

1. File(s) containing the source code.
2. The README file, which describes how to run your program.

DO NOT submit unnecessary files.