CS/CE/TE 6378: Project II

Instructor: Ravi Prakash

Assigned on: March 10, 2016
Due date and time: March 31, 2016, 11:59 pm

This is an individual project and you are expected to demonstrate its operation to the instructor and/or 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 Client-Server Model

This project is an extension of Project 1. Here are the departures from Project 1:

- Instead of one client, now you have seven identical clients, numbered 1 through 7.
- The three servers maintain identical replicas of a single file.
- The only type of file operation the clients perform is write.
- Multiple clients could concurrently issue write requests.
- Your task is to ensure that the file replicas are consistent after every *WRITE*. This can be achieved by ensuring that at any time at most one client can access the replicas in an exclusive manner and perform the following operations:
  1. Open the file,
  2. Append $< ID, SEQ\_NUM, STRING >$ (where ID is the client number, SEQ_NUM is the sequence number of the client write operation (initialized to zero), and $STRING$ is the client's hostname), in a new line, to each of the replicas of the file,
  3. Close the file.
- Maekawa's algorithm is employed to ensure mutually exclusive access to the files by the clients, with the following quorums corresponding to the seven clients:

Client number	Clients in its quorum
1	1, 2, 4
2	2, 3, 4, 7
3	1, 3, 7
4	4, 5, 6, 7
5	2, 3, 5, 6
6	1, 3, 6
7	3, 4, 5, 7

- Each client goes through the following sequence of operations until it has successfully performed 40 *WRITEs*:
  1. Waits for a period of time that is uniformly distributed in the range [10, 50] milliseconds before requesting next *WRITE*.

2. Enters the critical section using the mutual exclusion protocol.
3. On entry into the critical section, it performs the write in the format mentioned above.
4. Exits the critical section.

- Each client informs the first server after successfully performing 40 WRITEs. When that server receives such a message from all clients, it sends messages to all clients and servers to terminate.

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 enter the critical section:
   (a) The number of messages of each type (Request, Reply, Release, Enquire, Fail, Yield) exchanged.
   (b) The elapsed time between making a request and being able to enter the critical section.

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.