This project is about replica management in distributed systems. The purpose of the project is to implement a simplified version of the asynchronous distributed commit protocol presented in the paper “Support for Speculative Update Propagation in Mobility in Deno,” by Cetintemel, Keleher, and Franklin, which you have to read in preparation for this project. As part of your project you will also evaluate the performance of the protocol.

As part of the project, you will implement a simulated distributed system composed of N servers and one workload generator (WG). Each server is implemented as a UNIX process and the WG is implemented as a separate UNIX process for a total of N+1 processes. The servers and the WG communicate through UNIX message passing. See Fig. 1 for a depiction of this arrangement.

For the purpose of this project you will assume that there are M objects which are fully replicated at all servers. Each object is an integer initialized to zero. The WG selects one of the M objects with equal probability and generates a read or a write transaction to the selected object. Read transactions only read the value of one object and write transactions read and then update the value of one object only. The request is sent to one of the N servers, selected by the WG with equal probability.
The pseudo-code for the WG is

Loop
    Select one of the M objects with equal probability.
    Select one of the N servers with equal probability.
    Select a random number x between 0 and 1.
    If x <= probability the read probability Pr
       Then send a read request for the selected object to the selected server.
    Else send a write request for the selected object to the selected server.
    Record the result of the request including its response time.
End_Loop

Your first steps to carry our the project should be:

- Study the Deno paper.
- Specify the needed data structures to implement the Deno approach.
- Write a pseudo-code for the server in the case of the Deno approach.
- Implement the main program that forks the N servers, the WG, and creates all message queues (one message queue per server).

Some simplifications are made here with respect to the Deno paper:

- Transactions only deal with one object.
- Objects are integers stored in a main data structure and are replicated at all servers.
- The weak consistency protocol should be implemented.
- The currency of each server is 1/N.
- Each server starts a synchronization session with a randomly selected server at intervals of 0.2 sec.

The parameter values for your experiments are: M = 10, N = 4, Pr = 0.7.

Metrics to be collected during your experiments:

- Average response time of read transactions.
- Average response time of write transactions.
- Average time to commit a write transaction.

Deliverables:

Report with the following structure:

1. Introduction (describes the problem and organization of the report).
2. Description of the Deno protocol (using the simplifying assumptions given above).
3. Description of the Process Structure
4. Description of the Relevant Data Structures
5. Description of the Messages Exchanged Between Processes
6. Client Pseudo-code
7. Server Pseudo-code
8. Results of Experiments
   Appendix A: Source Code. Please provide a very well documented code. Points will be taken for lack or poor documentation.
   Appendix B: Sample Output.

Your code should be available for execution in case it is required. The accuracy, as well as the presentation of the report will be considered for grading purposes. Please make sure you use a spell and a grammar checker.