Jonathan Chu
CSCI-402
February 22, 2013

Part C: RW

Design and Implementation

The main problem to solve for this part was to design a way to allow for separation of reading and writing. To solve this problem, I consulted the following link for a design outline: http://cs.gmu.edu/cne/modules/ipc/orange/readsem.html. I modified it to use mutexes instead of semaphores. The following is my design for readers and writers:

```
mutex_reader = 1;                  // Controls access to the reader count
mutex_writer = 1;                  // Controls access of the writer
int reader_count = 0;              // The number of reading processes accessing the data

Read
{
     lock(mutex_reader);                      // gain access to reader_count
     reader_count = reader_count + 1;         // increment the reader_count
     if (reader_count == 1)
         lock(mutex_writer);                  // if this is the first thread to read,
                                              // a lock the writer
     unlock(mutex_reader);                    // allow other reader threads to access
reader_count
     read();                                  // read
     lock(mutex_reader);                      // gain access to reader_count
     reader_count = reader_count - 1;         // decrement reader_count
     if (reader_count == 0)
         unlock(mutex_writer);                // if there are no more threads reading from
the
                                              // DB, allow writing
     unlock(mutex_reader);                    // allow other threads to access reader_count
}

Write
{
     lock(mutex_reader);                      // gain access to the database
     write();                                 // write information to the database
     unlock(mutex_writer);                    // release exclusive access to the database
}
```

This design allows for mutual exclusion of readers from writers, while allowing for multiple readers to access the database at a time. After doing this design, the next step was to determine where to do the locking, and what kind of locking (reader lock or writer lock). I decided to follow the same plan as db_coarse, and lock only in query, add, and xremove. Further more, I do write locking for add and xremove, and read locking for query.

Known Defects

I have not found any known defects with my code.