Victor Wu

CS 350

Design Document

RWMON – Project 5

04/17/2015

**Purpose:**

To create reading and writing threads that reads a “data” file (which is really just the time). By default, we’re targeting the Reading Writer Monitor Scenario 2, in which writers are given priority. In this scenario, when writers are in queue, no new readers will execute until the writers are out of the C.S.

**Design:**

Though there is probably a more efficient design but we could implement this using five semaphores locks, them being mutex. Let s1,s2,s3,s4,s5 be the semaphores used in our program, then we would implemented it roughly like this:

s1 keeps the order

s2 tracks the read counter

s3 tracks the writing counter

s4 basic mutex ensuring mutual exclusion

s5 Ensures readers from entering iff there *∃* a thread waiting

The monitor will be a virtual abstraction of the threads being queued up to enter the Critical Section – being the “data” file (the time variables).

After each thread completion, the threads will wait for the time delayed (in milliseconds) before restarting (up to a total of 10 times).

**Usage:**

Please refer to the –-help command for additional details.

./wu\_p5

To run in user-input mode with raw data input into stdin

./wu\_p5 [Number of Writes] [Number of Reads] [Write Delay] [Read Delay] (opt.)

In our project, we’re only going for option 2 so by default (without the 6th argument), it will execute the program in option 2. However option 1 will call functions in which readers are given priority, and option 3 in which neither readers nor writers are given priority.