CSE 430 Summer 2014 - Project 2

Ryan Dougherty

Abstract

This document will cover Project 2, which is about solutions to the Reader-Writer problem. We will observe the output produced by the code given, then modify the critical section in Database.java to implement the starvation-free pseudo-code algorithm (but is unfair), and finally modify RWQueue so that our implementation is fair.

1 Part 1 Output

The Part 1 output is the following:

```
reader 0 is sleeping.
reader 2 is sleeping.
reader 1 is sleeping.
writer 0 is sleeping.
writer 1 is sleeping.
reader 2 wants to read.
reader 2 is reading. Count = 1
reader 2 is done reading. Count = 0
reader 2 is sleeping.
reader 0 wants to read.
reader 0 is reading. Count = 1
writer 0 wants to write.
reader 0 is done reading. Count = 0
reader 0 is sleeping.
writer 0 is writing.
reader 1 wants to read.
writer 0 is done writing.
writer 0 is sleeping.
reader 1 is reading. Count = 1
reader 2 wants to read.
reader 2 is reading. Count = 2
```

As we can see, there are some sections that allow reader and writer 0 to take much more time than 1 or 2. Therefore, this algorithm is not starvation-free.

2 shm-posix-{consumer, producer}.c

The purpose of these programs: to create a consumer-producer relationship in C where the producer produces items, and the consumer consumes them once they are available. They accomplish this by creating a shared-memory space from which the producer can insert items and the consumer can read from them. The program output is the following:
Studying Operating Systems Is Fun!

3 Date{Server, Client}.java

The purpose of these programs: to create a server-client relationship in Java to provide a service to a client. The functionality is to give the current date and time. The program output is the following:
Thu May 29 10:11:16 MST 2014

4 unix_pipe.c

The purpose of the program: to create a pipe and fork a child process in C. The program constructs a pipe to be shared between the child and parent process. If we are the parent process, we write the message to the array buffer fd. If we are the child process, we read from the READ_END of the buffer fd, and print out that the child read the message from the buffer. The program output is the following:
child read Greetings

5 thrd-posix.c

The purpose of the program: to work with the Pthread library in C to create a pthread that will execute some work. The work in this program will sum all of the integers from 1 to the supplied argument. The program initiates, creates, and joins the pthread created, and then prints the sum. In the pthread_create method, we assign the "work" to be done to the pthread, which is the runner method, and forwards the argument supplied to the program in pthread_create.

The program output is the following (with integer parameter 2000):
sum = 2001000