Jasper Hale V00888481

Design Document

CSC 360 Assignment 2

1. How many threads are you going to use? Specify the task that you intend each thread to perform.

I intend to use a thread for each clerk and for each customer. There will also be a main thread for when the process starts, which will create the customer and clerk threads. The clerk threads will be used to signal the customer queues and allow a customer thread to execute their service time. During this customer execution time, the clerk thread will be waiting for the customer to return. Once a Customer is finished, the clerk will signal to the next customer.

2. Do the threads work independently? Or, is there an overall "controller" thread?

All the threads work independently. The main thread will be waiting for the customer threads to return and then print out statistical information about the service times. The clerk threads should be able to signal to the queues as soon as they are available.

3. How many mutexes are you going to use? Specify the operation that each mutex will guard

I will use 13 mutexes. There will be waiting time mutexes for the economy and business customers, these waiting time mutexes are needed for threads to add their waiting time to the overall waiting time. There will be mutexes for locking the economy que and business que to enqueue and dequeue customers. A que status lock is used to guard which clerk served a certain que. There is a que length mutex that guards the length of each queue, which is manipulated when enqueue or dequeue is called. A winner selected mutex so that only one customer is picked from a que by only one clerk. A serve lock for both the economy and business que to indicate to the customer that a clerk wants to serve someone. Mutexes for both economy and business for when the customer has finished, and the clerk can go now wait for another customer. Receive mutex which ensures that a customer got a clerk request to serve them. A mutex to see if the right clerk got the finish signal from the customer

4. Will the main thread be idle? If not, what will it be doing?

Once the main thread creates the customer threads and clerk threads, it will wait for the clerks to serve the customers by calling pthread_join on the customer threads. After which the main thread will destroys the mutexes and print statistical information about waiting times.

5. How are you going to represent customers? what type of data structure will you use?

I will use a typedef structure that will be used to store information of each customer. So once all the threads are created, they will have a struct that contains all the information for a particular customer.

6. How are you going to ensure that data structures in your program will not be modified concurrently?

Each thread will have their own data structure so no other thread can modify that data structure. The structure will also be saved locally to each thread. Also, there will be no operations to modify another threads data structure.

7. How many convars are you going to use? For each convar:
    a. Describe the condition that the convar will represent

There will be 4 convars. For each business and economy que there will be a convar for clerks serving customers. The other two convar is for when the customer is done being served and signals to the clerk that it is finished.

    b. Which mutex is associated with the convar? Why?

The mutexes associated with the serve convars will be the serve mutexes because the purpose of those mutex is for clerk about to serve a customer. The finish mutex will be used for the finish convars.

    c. What operation should be performed once pthread cond wait() has been unblocked and re-acquired the mutex

After wait for the serve convar, the customer will check to see if it at the front of the queue, if so it can be served. After wait for the finished convar, the clerks will return back to a waiting state and the customer will return.

8. Briefly sketch the overall algorithm you will use. You may use sentences such as:

Main:

For cust in customer.txt
    Store information in CustomerInfo
For each clerk:
    Createthread(clerk)
For each customer in customer.txt
    Createthread(customer, customerInfo)
For each customerThread:
    Pthread_join()
Destroy mutex

Do calculations

Customer:

Record customer info
Sleep for the arrival time
Lock either business or economy que depending on class type
Lock que length

Enqueue
Unlock both que length and the que that is got into
Record entry waiting time

```
        While true
                Check which que the customer belongs to
                Lock the mutex of the que the customer is in
                Get the head of the que
                Unlock the queue
                Lock the served mutex
                Pthread_cond_wait to be served
                Unlock the serve
                Lock clerk recieved
                See what clerk will serve me
                Update clerk received signal
                Unlock receive
                Lock winner slected
                If (thread_is_at_front && !selected[pos])
                        Dequque
                        Queue length—
                        Selected[pos] = true
                        Break
                Unlock winner selected
        Lock, update waiting time, unlock
        Update que service time
        Starts recording service time
        Start serive time
        Record end time
        Check which que
        Lock finished mutex
        Lock clerk server
        Set clerk server to true
        Unlock clerk serve
        Send broadcast to fin convar
        Unlock the finish lock
        Pthread_exit
        Return
Clerk:
While true
        Record id
        While true
                Check if either que is empty
                        Break if there is someone in a que
        Check if its business or economy
        Lock qstatus
        If there isn't anyone in the que
                Lock winner selected
                Set winner selected to true
                Unlock winner selected
                Unlock qstatus
                Continue
        Change queue stat of the que to clerk ID
```

```
Lock the winner selected
Change winner selected to false
Unlock winner selected
While true
        Lock the serve mutex depening on if its econ or business
        Broadcast serve
        If clerkRecved[clerID-1]
                Unlock serve mutex
                Break
        Unlock serve mutex
Unlock q status
While true
        Lock fin lock depending on econ or business
        Wait on finish convar
        If clerkServer[clerkID-1]
                Lock clerkserver
                Change clerkserver of clerk to false
                Unlocl clerk server loc
                Unlock fin lock
                 Break
        Unlock fin lock
Lock reve serve lock
Change clerk reiece to false of specific clerk
Unlock receive lock
```