Rochester Institute of Technology

Real Time and Embedded Systems

Project 4 – Bank Simulator Design Document

March 26, 2017

Zachary Weeden | zdw7287@rit.edu

**Overview**

This project is to demonstrate a simulation of a banking environment. Furthermore, the main goal is to emulate the workflow of said environment via a queue queuing to a multi-threaded server (bank tellers).

Customers are to enter the bank at a rate of somewhere between 1-4 minutes based on a uniform distribution. This is to say that at completion, when looking at all intermittent customer arrival times, there is the roughly the same amount of customers that arrived 1 minute after and arrival at 2, 3, and 4 minutes. Each of these customers are to enter a main customer queue.

Customers are then dequeued and seen by 1 of 3 tellers to complete their transaction/business.

This transaction time is randomly generated and can range anywhere from 30 seconds to 8 minutes for a single customer. This is also a uniform distribution amongst all customer transaction times. At this time the teller is unavailable to serve other customers.

Hours of operation are existent and must be abided by. Customers can be queued up until closing and customers arriving before close and queued prior to close are guaranteed to be seen.

**System Block Diagram/Workflow**



**Hours of Operation Timer**

This is the main control of the program and used in determining if more customers should be generated and queued for servicing. This individual component is emulating the hours of operation; open and close of the bank.

**Customer Queue**

This is a structure to organize the arrival of customers. This can be of the array data type and be in it’s own thread. The queue will continually be dequeuing by the 3 teller threads throughout program execution and customer completion as well as be appended to with the new arrival of customers until the close time.

**Bank Teller X**

These are 3 separate threads that will be used to handle and process customers from the customer queue. Each teller thread will lock once a customer is taken into that thread via a semaphore/mutex to indicate that this particular teller is unavailable.

After the randomly generated transaction time of the specific customer has been exceeded, the customer is deemed to be handled and is completed with their business. Metadata of the customer’s transaction will be retained for processing of overall/average metrics at the program’s completion and the thread will then unlock indicating that a customer can be seen/taken in by this teller thread.

**Software Design**

In regards to answering listed questions in the report document:

1. The customer queue can simply just to be thought of as an array of elements – I plan on making these elements the randomly generated transaction times of that particular customer.
2. On the topic of generation – I plan to do something like specify a range for valid values and generate a uniform distribution with something like:

*return lower\_bound+(rand()/(RAND\_MAX/(upper\_bound+1-lower\_bound));*

1. Customer-Teller assignment for the first of each teller is to simply take in the first 3 customers to each one of the tellers. From there, the unlocking of a thread will indicate precedence and more importantly, availability to accept another customer.
2. Metrics will be handled at the end of the program but individual customer metrics will be stored as soon as each customer has been completed.