Names: Aneesha Dasari and Veena Nalluri

VUnet IDs: dasaria and nallurv

Emails: [aneesha.dasari@vanderbilt.edu](mailto:aneesha.dasari@vanderbilt.edu) and [veena.nalluri@vanderbilt.edu](mailto:veena.nalluri@vanderbilt.edu)

Class: CS 3281 – Vanderbilt University

**Overview**

This project implements a bank ledger. Using the distributed consensus problem solution of the two phase commit protocol, this program simulates five different ATMS or instances that hold information about one bank account. An instance of the program runs by calling main and being given the port number for itself and the port number for the other instances. Each program, or ATM, supports three actions. The first is “credit” which will add money to the bank account. The second is “debit” which will remove money from the bank account if there is enough money. The third is “query” which will return the amount of money in the bank account. There is also a “help” command which will describe how to use the other three. This project also handles the fault case of if an instance of the program terminates. When the instance, or ATM, is revived it will get the current balance of the bank account from other processes.

**Implementation Details**

We implemented this by having a main function and a function called receiving\_ports which implements the two phase commit protocol. Within the program we have a few important global variables: the balance of the bank account and a value called acknowledge that tracks the number of acknowledgements from other instances of the program.

The main function takes in the arguments that are given to the program, which are the first port number (the coordinator port) and the rest are the connecting ports. It then creates a socket and binds it to an arbitrary address. It then creates a thread to send and receive commands which enters a loop to do so. This loop takes in a command and figures out what command was given to it to act accordingly, which only occurs if all other instances have acknowledged the command. This is implemented through an if statement. The receiving\_ports function works with getting the acknowledgments. It received the type of transaction and uses the two phase commit protocol using sockets to send “yes” or “no” for when being prompted by a different coordinator and to count acknowledgments and “commit” for when it is the coordinator.

**Main Challenges**

There were a few challenges to this project. The first was to figure out the specifics of the desired solution to the problem. It took a lot of research and asking questions to figure out exactly what was needed from this program, to figure out what should happen in different cases, and to figure out how this should work. We struggled a bit in trying to outline the frame of the program itself. It was difficult to take our working understanding of the program and figure out what files and methods we needed. We outlined something that we later changed. Another difficulty was implementing the sockets and the communication. Although we had an example in class, using sockets in the program using C proved difficult. We finally figured this out by using online forums. Finally, the division of work was hard. We found working remotely and evenly was difficult, as this was a smaller scale project. As such, we had to divide certain things up, and work together in person on one computer for other parts of the project.

**Division of Work**

As stated, this was difficult to divide every aspect of the project evenly. Although both people worked on all parts of this project, Veena worked more on the working base of the project with assistance with implementation, debugging, and cleaning up code from Aneesha. Aneesha worked more on the report and presentation preparation, with assistance with writing, proofreading, and preparation from Veena. Overall, division of total time and effort ended up pretty even.

**The Result**

The result is a working prototype. Opening up five different terminals and beginning five instances of the program, a user can issue commands from any of these instances to credit or debit to the bank account. The program could definitely be cleaned up even more to account for even more fault cases. However, the resulting program is a basic beginning for a bank ledger using the two phase commit protocol to sole the distributed consensus problem.