CS39002: Operating Systems Lab.
Assignment 6
Floating date: 14/3/2016
Due date: 21/3/2016

Problem statement: In this assignment, you have to manage an ATM system, where clients can perform various operations like withdraw money, deposit money, or check balance. The master process creates a set of ATM processes, an ATM locator file and different communication mediums. Each client interacts with the ATM with the help of the ATM locator. The ATM locator is used to lock and access the individual ATM. Once gained the access, client performs different ATM transaction operations. Each ATM has its own shared memory to store information about the transactions conducted on it. The ATMs executes two different types of consistency check protocols, based on the client transaction request.

Major processes: In the following, we illustrate the major processes connected with the system.

Master process: The master process spawns the ATM processes. Moreover, it creates the message queue via which different ATM processes can communicate with the master process. The master process also creates and populates the ATM locator file. The detail content of the ATM locator is described later. The Master process maintains a global shared memory space which keeps the information of the each client, updation timestamp and the current balance.

ATM process: The different activities of the ATM get simulated inside the ATM process. ATM process supports different operations on behalf of the clients such as "WITHDRAW", ''DIPOSIT", "VIEW" etc. Each ATM process maintains a local shared memory which stores the transactions performed by each client. Moreover, each ATM process creates a message queue to enable communication with the client. The message queue information becomes available in the ATM locator.

Client process: The client connects with the ATM by executing a client program. The communication between the client process and the ATM process is performed via a message queue; one queue for each ATM. The client executes the joining protocol to connect to the ATM; the joining protocol is described later. Once you execute the client on a terminal, the prompt waits for your command. As client triggers "ENTER ATM n", the client process contacts to the ATM locator to take the lock and connect to ATM n. The process ID of the client acts as the a/c number of the client. If the a/c of the joining client exists, the process continues. Otherwise the ATM process creates a new account for the client.

ATM locator: ATM locator is a shared file, created by the master process, which contains information of each ATM. Each entry x of the file stores (a) ATM ID x(b) the message queue key of the corresponding queue, which enables the client communication with ATM process x (c) semaphore which ensures lock on ATM x (d) Key of the shared memory segment of the ATM x. Once the client attempts to connect to ATM x (via ENETR ATM x), client contacts the ATM locator and locks the corresponding semaphore. Once the client locks the ATM, it starts the communication with ATM x via the message queue; otherwise client is free to choose any other ATM. The concurrent access of the ATM locator is allowed.

Operations: Next we describe the client activities and the different operations supported by the ATM

Clients: Each client has a unique account number (process PID), which can be communicated during the interaction with the ATM. Each client will have a terminal open for her, where the client can specify an ATM she wants to access.

Joining protocol: The ENTER command enables client to access the ATM, after achieving a lock of the ATM, such that there will be only one client inside an ATM at any time. The client sends the process ID (account number) to the ATM process. ATM process in turn communicates with the master process to check if this is an existing client. If it's true, the client continues. Otherwise, master process creates a new entry in the global address space (i.e creates a new client). Appropriate message should be printed by the ATM process.

After choosing the ATM, the client can perform the following operation. Each client is shown a list of options to choose from.

WITHDRAW: The client can withdraw money from the account; she enters the amount that she wants to withdraw. But this might not always be a valid operation; this operation may lead to negative balance for the client. So before performing any withdraw operation, the ATM does a consistency check for this client's account balance with all other ATMs to ensure that it is a valid operation. The protocol followed is called "Local consistency check", which is described later. If the withdraw operation is valid, then the ATM stores this transaction in its shared memory. Otherwise, it notifies the client that the operation is invalid. Each transaction triggered by the client process, will be timestamped. Note that, during this (withdraw & local consistency check) process, previous transactions remain intact.

Deposit: The client enters the amount of money he wants to deposit. Once the client deposits the amount in her account, ATM process adds the corresponding transaction in the local shared memory. Note that, during this (deposit) process, previous transactions remain intact.

VIEW: The client wants to check her current account balance. The corresponding ATM requests the master process (via message queue) to perform a global consistency check. The master process contacts all the ATM processes (transactions) to find out the correct account balance for this client. After computing the balance, the master process updates the figure in the global address space, for that client and the ATM process removes the corresponding transactions from the local shared space. This communication is performed via shared memory. Additionally, ATM process keeps an image of the updated balance information of that client in the local shared space. Note that, this local image is necessary for the local consistency check.

LEAVE: Client leaves the ATM, and releases the lock.

In the following, we describe the two different shared memories, maintained in this system.

Local shared memory: This is local to each ATM process. During local concurrency check, this local memory segment gets attached to the peer ATM processes and performs the required update. The peer ATM shared memory information can be accessed from the ATM locator. The local shared memory maintains two tables (a) stores timestamped transactions after each withdraw and deposit (b) timestamped balance image of each client; this table gets updated only after the global consistency check.

Global shared memory: This space is maintained inside the master process. This table contains timestamped balance of each client. This table gets updated only after the global consistency check. If a client joins, master checks this table to decide if she is an existing client; otherwise creates a new entry in this table. Note that, you may not need to share this space with any other process.

Consistency Checks protocols:

Since the ATM processes are distributed, we need a consistency data in all the ATMs. These will be ensured by consistency checks in the system. There will be two kind of consistency check in the system.

- **Local Consistency** : This protocol gets executed when a **WITHDRAW** command is received by an ATM. Then the requesting ATM runs a consistency check across all the ATMs for that specific client. During this consistency check, the initiating ATM process attaches the local shared memory of other ATM processes and accesses the transactions and decides whether this WITHDRAW is a valid operation. Note that, this check does not alter the shared memory; the transactions remain intact.
- **Global Consistency :** This consistency check will be carried out by the Master Process (whenever one client requests for a **View** operation). If any client wants to view his account balance through a particular ATM, then that ATM process requests the master process through message queue for global consistency check. Master process then periodically attaches with the local shared space of the ATM processes (i.e ATM1,ATM2,....,ATMn according to the order of creation) and access the transactions of that particular client. In this protocol, the master process reads the transactions from the shared memory of each ATM and then calculates the latest balance. This updated balance should be written in the global shared memory by the master process and also an image should be kept in the local shared space of the requesting ATM. During this procedure, all the accessed transactions from the local shared space will be removed. This protocol ensures that the requesting client account balance should get synchronized across the ATMs

Every client process has a terminal open for them, and the ATM processes write all their transactions in a central terminal.

Submission:

You need to implement three programs, **master.c, atm.c , client.c** . Master process creates global shared memory and a set of ATM processes; one process per ATM. Each ATM process creates its own local shared memory in atm.c and maintains its clients' list and transactions. All query (**Enter, Add**, **Withdraw**, **View**, **Leave**) operations are implemented in client.c.

Sample Output:

Client Screen:

Client: ENTER ATM1
ATM1 occupied
Client: ENTER ATM2
Welcome Client 1100011
ATM Options will appear
Client: ADD 500
Rs 500 to account.
Client: WITHDRAW 600
Balance Insufficient
Client: WITHDRAW 200
Withdrawal Successful
Client: LEAVE
Good Bye Client 1100011.

Client: ENTER ATM1
Welcome Client 1100011
……

The Master Terminal

ATM2: Client 1100011 entered
ATM2 : Running a consistency check for a/c 1100011.
Master: Done
ATM2: Client 1100011 left.
ATM1: Client 1100011 entered