Distributed Library Management System (DLMS) using Java RMI

Name : Hao Lei
StudentID : 40056875

1.Design

This project is about distributed library management system that include three departments and two kinds of clients: student and manager. In my implementation, I use Java RMI to communicate between client and server UDP/IP sockets to ensure inter server communications. I build my system in the follow steps:

First, build Server Object interface. It is a remote interface that define the method for the server object. The functions which are offered in servers should be introduced at this interface. For example,there are six client operations, so at least the functions realize those operations should be included in this server object interface.

Second, build Server Object. It is the part of an instance of the server object interface, and the function of which is to register the remote object and make its reference available to clients. Besides the setting part, the server object has a more important duty, which is implementing the functions in the interface, and all function should have systematic result since this design deploy at least once service. Moreover, even there are only six functions in server object interface, it may contains tens of functions to achieve the function. This section is the core for DLMS.

Third, construct server stub, it is the part of an object resides on the client host and servers as a representative of the remote server object. Server stub in DLMS has two mission, one is decided the client belongs to which server. Second is listen to every thread, start registry and list all those registries.

Forth, UDPRequest. because here are three libraries and the servers need to communicate with each others so UDP socket is needed. UDP Request is a tool to get command and message and pass the requests to other servers for clients. If a function in the server need to communicate with other servers, that function should be listed in this session.

In additional, UDPListening. There are so many threads in server socket, a UDP listening is necessary to ensure those thread can execute in an order.

Next, ExecutionCommand. Threads are need in UDP because we use thread to achieve concurrent server access. Functions that are listed in UDPRequest should have detail implementation here. Functions of one university, say CON can communicate to functions of another server, for example: MCG.

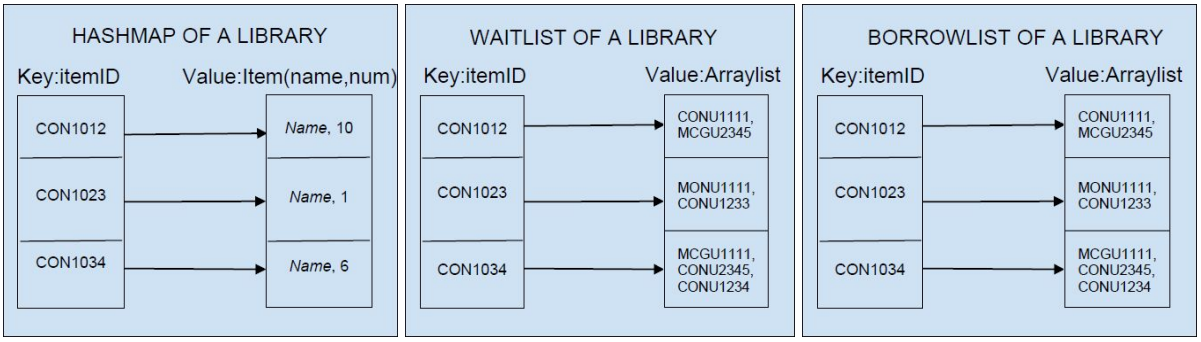
Finally, two kinds of user client are needed. They are student role and manager role.When a client is logging in, a function is need to make sure the client is valid. And after logged in successfully, client can do further operations according to their roles:

1. If the fourth letter is “M”, this client will be logged in as manager and has three options: addItem, removeItem, listItemAvailability.
2. If the fourth letters is “U”, then this client will be logged in as user and can also do three operations: borrowItem, findItem, returnItem.

Besides, both manager and user will be connected to different servers according to their prefix(“CON” to Concordia, “MCG” to McGill and “MON” to Montreal).

2.Data Structure

1.One item hashmap is needed, as the assignment description said, named “items”. This hashmap maintains all the items available in a library. The key of this hashmap is string type of itemID, the value of the key is an “Item” object with two fields name and num which stands for item name and item current quantity.



1.items 2.borrowedItem 3.waitList

2.BorrowedItem hashmap maintains all the items that borrowed by users. The map key is itemID and the value is a string arraylist with userID. Whenever one item is borrowed by someone successfully, that item will be added to this map and the user will be added to arraylist. When a user return item to library, we check this map to ensure that user is the one who borrowed this item.

3.WaitList hashmap keep the record of users who wait to borrow that item. The key of map is itemID and the value is a arraylist with userID in it just as borrowedItem hashmap. When a user is failed to borrow a item, the server will ask that user whether they want to be added to waiting queue, is the answer is yes then the user will be added to waitlist. Whenever the item is available the first user in the list will be auto lended by server.

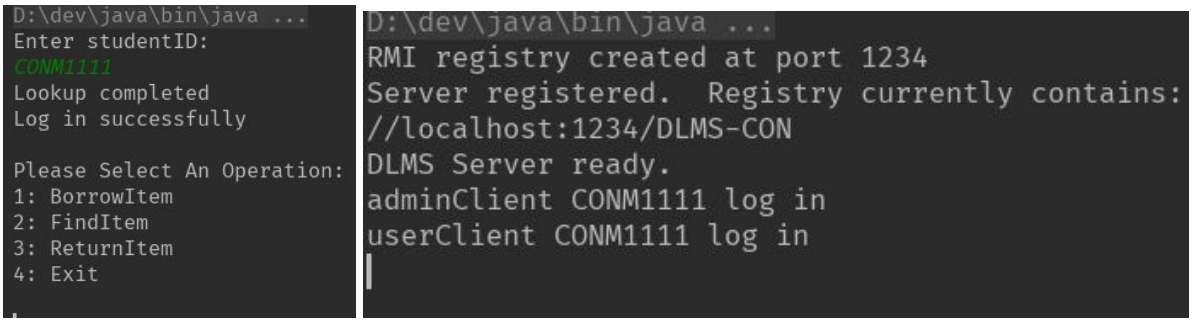
3.Test Case

1.Manager log in

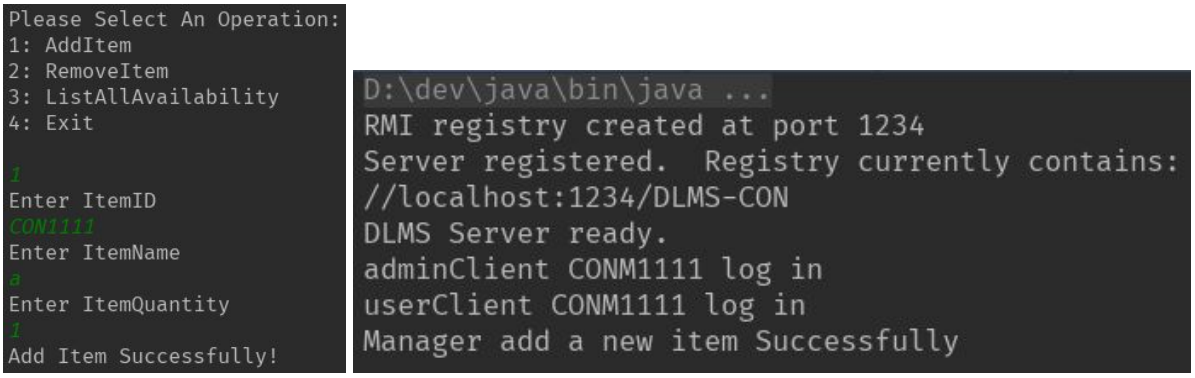
```
D:\dev\java\bin\java ...
Enter adminID:
CONM1111
Lookup completed
Log in successfully

Please Select An Operation:
1: AddItem
2: RemoveItem
3: ListAllAvailability
4: Exit
```

```
D:\dev\java\bin\java ...
RMI registry created at port 1234
Server registered.  Registry currently contains:
//localhost:1234/DLMS-CON
DLMS Server ready.
adminClient CONM1111 log in
```

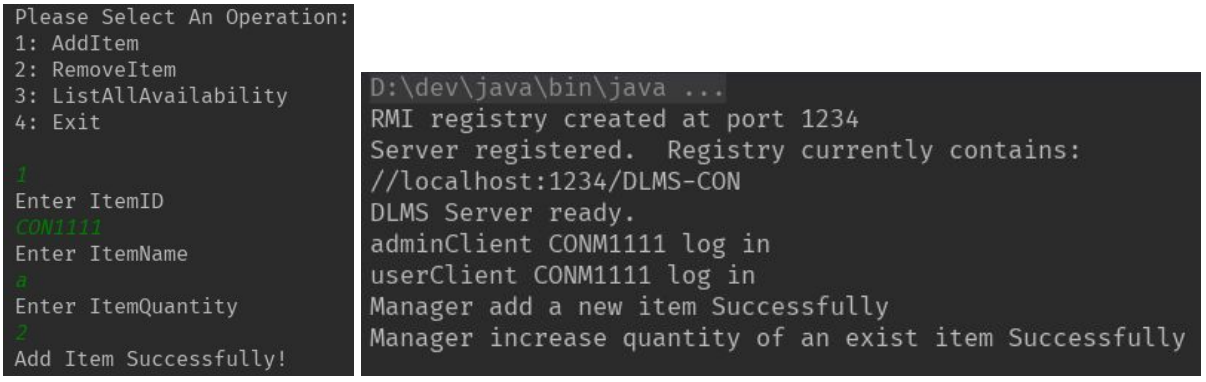
2.User log in



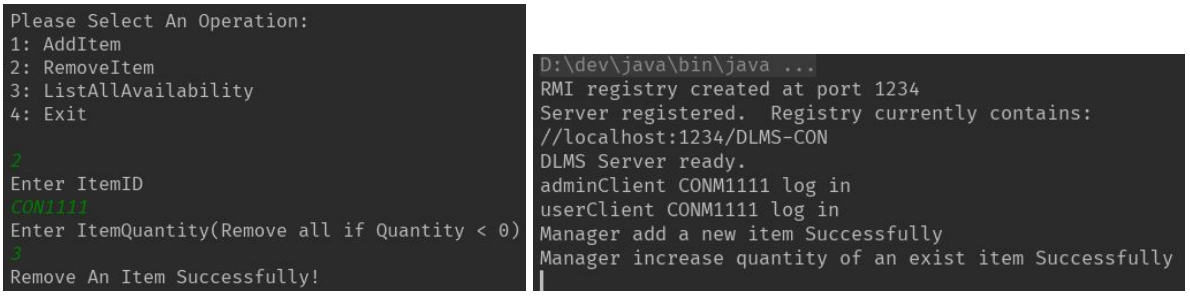
3.Manager add item



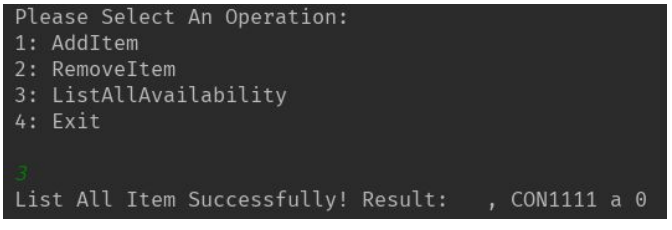
4.Manager add duplicate item



5.Manager remove item



6.Manager list item availability



7.User borrow item

```
Please Select An Operation:
1: BorrowItem
2: FindItem
3: ReturnItem
4: Exit

1
Enter The ItemID
CON1111
Enter The NumberOfDays
1
Borrow Item Successfully!
```

```
D:\dev\java\bin\java ...
RMI registry created at port 1234
Server registered.  Registry currently contains:
//localhost:1234/DLMS-CON
DLMS Server ready.
adminClient CONM1111 log in
userClient CONM1111 log in
Manager add a new item Successfully
Manager increase quantity of an exist item Successfully
Manager increase quantity of an exist item Successfully
User CONM1111 borrow an item Successfully
```

8.User wait in list

```
1
Enter The ItemID
CON2222
Enter The NumberOfDays
1
Borrow Item Failed!
Do You Want To Wait In The Queue?(y/n)
y
Wait In Queue Successfully!  User CONU1111 waiting in queue of item CON2222 at position 1
```

9.User find item “a”

```
Please Select An Operation:
1: BorrowItem
2: FindItem
3: ReturnItem
4: Exit

2
Enter the ItemName
a
Find An Item Successfully! Result: CON1111 0 MCG1111 1 MON1111 1
```

10.User return item

```
1
Enter The ItemID
MCG1111
Enter The NumberOfDays
1
Borrow Item Successfully!

Please Select An Operation:
1: BorrowItem
2: FindItem
3: ReturnItem
4: Exit

3
Enter ItemID To Be Return:
MCG1111
Return Item Successfully
```

4.Summary

From the above sessions, grasp design, data structure and test cases, it shows how this DLMS system constructed and how functions implemented, and the result of all client operations.

The most important part of this project is client and server communication using Java RMI, server inter communication using UDP. During the project, the most challenging part, though, is the implementation of server object , a lot of things must be concerned since each function will be called by different servers many times. To make the function results systematic, the data structure also must be choose carefully.

Another difficult part would be the design of each server since each of them could be both server and client at the same time, listener thread must be employ to make sure the inter communication between servers works as expected.