DEPARTMENT OF
COMPUTER SCIENCE AND SOFTWARE ENGINEERING

COMP 6231 - DISTRIBUTED SYSTEM DESIGN

WINTER 2019

DISTRIBUTED LIBRARY MANAGEMENT SYSTEM USING JAVA RMI

Submitted By:

Alekya Karicherla (40059347)

Contents

Overview

The assignment is to develop a Distributed System for a group of libraries: used by library managers to manage the information about the items available in the libraries and library users to borrow or return items across the libraries.

This document presents the designs used in implementing the project.

Terminology & Definitions

Abbreviations

DLMS: Distributed Library Management System

RMI: Remote Method Invocation

JVM: Java Virtual Machine

UDP: User Datagram Protocol

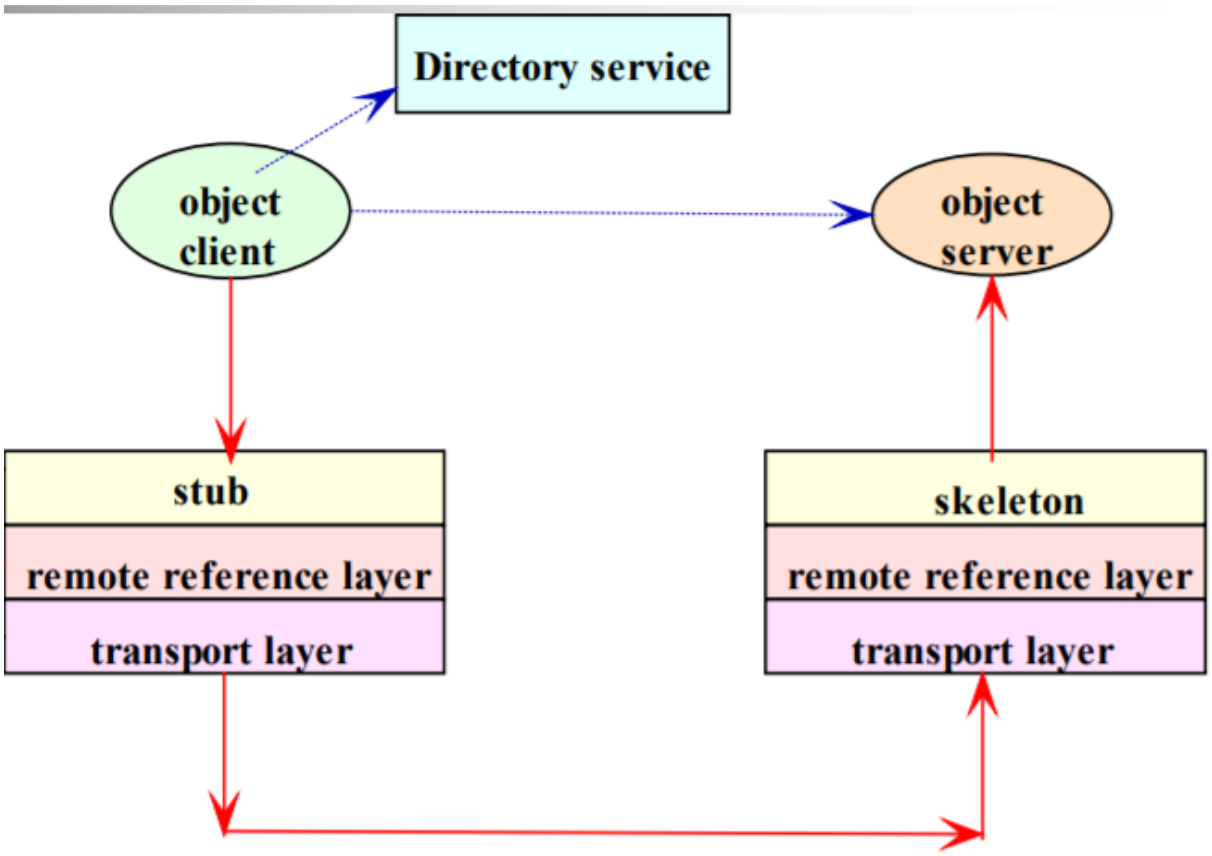
Functional Descriptions

Requirements

Feature	Description
Add Item	This allows Manager to add/update details about the books into the respective library.
Remove Item	This allows the Manager either to reduce the count or to remove the book from the respective library.
List Item Availability	This feature is used by manager to look all the books and their quantity available in the respective library.
Borrow Item	This allows the user to borrow a book from the library, else add the user to a waiting list.

Find Item	This allows the user to know in which library the book exists along with their available quantity.
Return Item	This allows the user to return the book to the library where the book belongs to.

System Architecture



The system is built on JAVA RMI Architecture.

It is a client-server model, in which the server provides a set of procedures that are available for use by clients.

Client & Server: A client Java program communicates with the other Java program on the server side. RMI is nothing but a communication between two JVMs placed on different systems.

Stub/Skeleton: Stub is client-side proxy and Skeleton is server-side proxy**.** The client server communication goes through these proxies. Client sends its request of method invocation (to be executed on remote server) to stub. Stub in turn sends the request to skeleton. Skeleton passes the request to the server program. Server executes the method and sends the return value to the skeleton (to route to client). Skeleton sends to stub and stub to client program.

Remote Reference Layer (RRL): Proxies are implicitly connected to RMI mechanism through Remote reference layer, the layer responsible for object communication and transfer of objects between client and server.

Transport Layer: sets up, maintains, and shuts down connections; and carries out the transport protocol.

Detailed Architecture

Our DLMS consists of three servers namely, CON, MCG and MON. All the servers perform same type of operations.

The functionalities mentioned above will be defined in an interface. The implementations of the functionalities will be added in ServerImplementation Class.

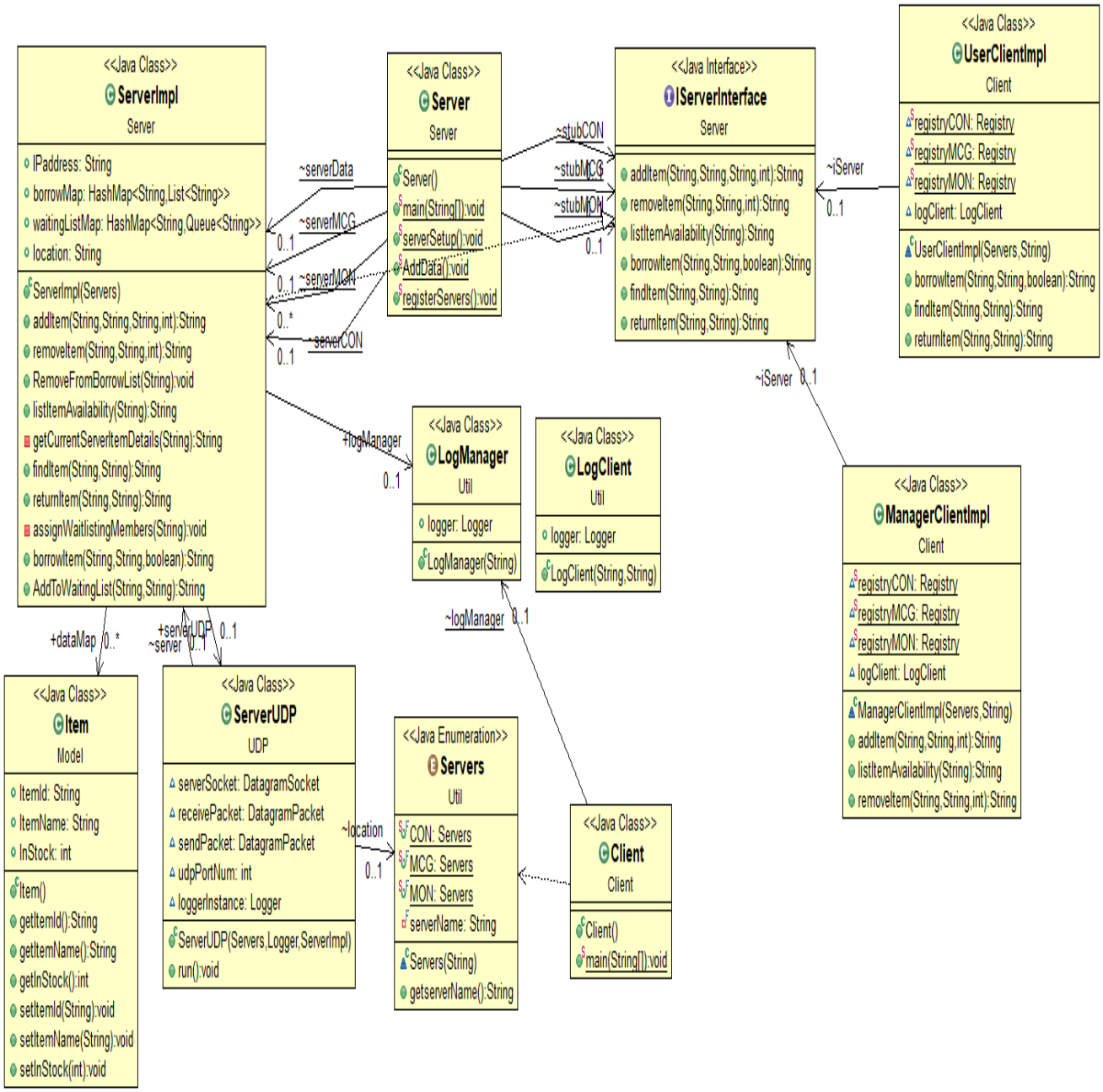
We also have ManagerClient and UserClient to perform their respective functionalities.

Data is stored in the form of HashMap & Queues in each server.

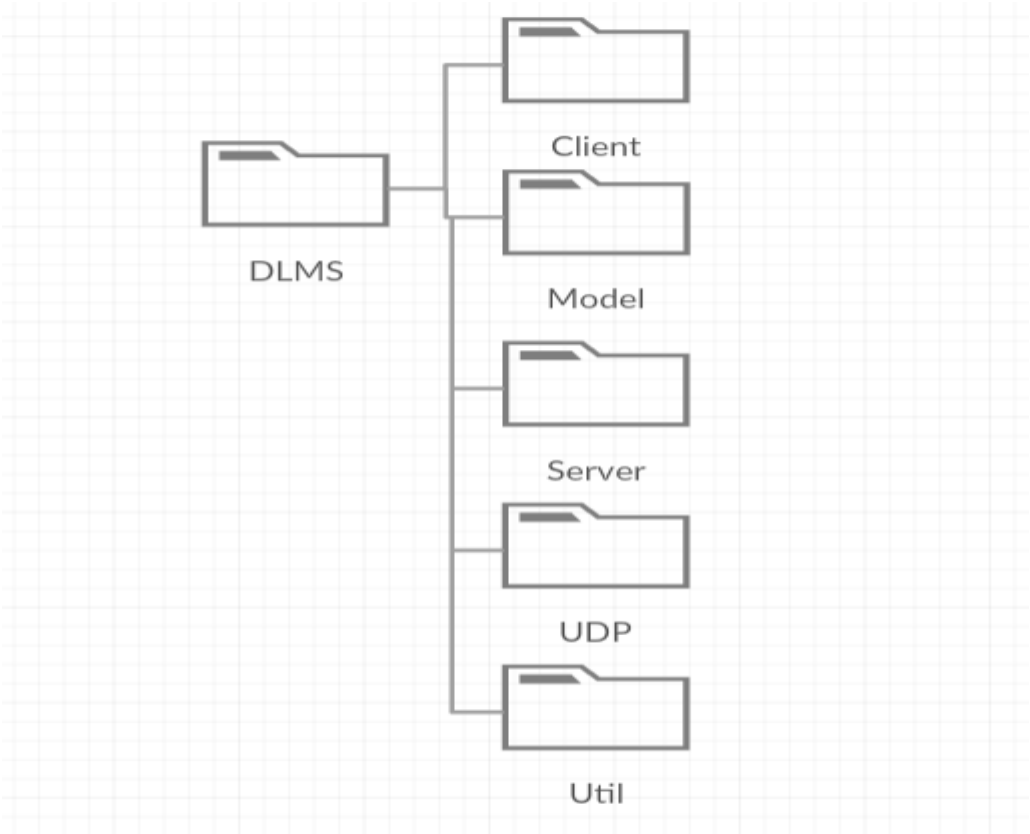
Server to Server communication is maintained by UDP.

Detailed System Design

Class Diagram



Folder Structure



Test Scenarios

Test Id	Test Description	Expected Result	Actual Result
T001	Manager/User Id is entered.	Menu should be displayed according to the user.	As Expected
T002	Add Item: Manager adds an item to the library.	Item should be added to the HashMap of the respective server	As Expected
T003	Add Item: Manager adds existing item to the library	Item quantity should get updated.	As Expected.
T004	Remove Item: Manager gives the item Id and quantity to the removed for that item	Item quantity will be reduced/the item will be removed from the HashMap.	As Expected

T005	Remove Item: Manager gives wrong Item details.	Should display a message stating, "Item does not exist".	As Expected
T006	List Item Availability: Manager requests to see all the items available in their library	All the items and their available Quantity will be displayed.	As Expected
T007	Borrow Item: User requests for an item from his/her own library and the book is available.	The book will be assigned to the user.	As Expected
T008	Borrow Item: User requests for an item from his/her own library and the book is not-available.	User will be prompted whether to be in waiting list for the book or not.	As Expected
T009	Borrow Item: User wants to be in waiting list	User will be added to a waiting list queue of that item.	As Expected
T010	Borrow Item: User doesn't want to be in waiting list.	Main menu for the user will be displayed	As Expected
T011	Find Item: User searches for a book with its name.	Displays the details about the Item Id and quantity available across all the libraries.	As Expected
T012	Find Item: User searches for a book with wrong name.	Displays "No records found" message.	As Expected
T013	Return Item: User returns the borrowed book	The user and book will be removed from the borrowed list, the count of book in the library server will be incremented.	As Expected
T014	Return Item: User returns a wrong item Id	Error message "No such Item borrowed. Please try again." will be displayed.	As Expected
T015	Return Item: Checks the waiting list queue of the book	If there is any user waiting for it, assigns the book to the user and removes him/her from the waiting list.	As Expected
T016	Borrow Item: User requests to borrow item from another library.	If the item is available, it will be assigned to the user, else adds to the waiting list, as per the user selection.	As Expected
T017	Borrow Item: User already borrowed one item from other library and requests for another book from the same library	An error message will be displayed as "User can borrow only one book from other libraries".	As Expected

T018	User enters id in wrong format.	An error message "Invalid choice! Please try again." will be displayed	As Expected
T019	List Item Availability: Manager requests to see all the items available in their library, when no items were added.	A message "No Records found" will be displayed.	As Expected
T020	Manager/User Id is entered in the form of LIBRXXX	A message "Too many/less characters in the ID. Please enter in (LIBRXXXX) format, where LIB={CON,MCG,MON} and R={M,U}" will be displayed.	As Expected
T021	User enters an invalid ID (E.g.: CONM7Y*6)	A message "Invalid character in ID. Please enter in (LIBRXXXX) format, where XXXX can only be numbers" will be displayed.	As Expected
T022	Remove Item: Manager tries to remove an item with quantity greater than available quantity.	A message "Quantity entered is incorrect" will be displayed.	As Expected
T023	Remove Item: Manager enters the quantity as "-1".	If the book is borrowed by any user, it will be removed, and the item will be completely removed.	As Expected
T024	User/ Manager enters invalid item Id. (E.g., XYZ7845)	An error message "Invalid ItemId. Please try again." will be displayed.	As Expected
T025	Return Item: User tries to return a book which is not borrowed.	Error message "No such Item borrowed. Please try again." will be displayed.	As Expected
T026	Remove Item: Manager tries to remove/reduce another library's item.	An error message "Item doesn't exist will be displayed.	As Expected

Challenges

Understanding the UDP connection and establishment among the servers is challenging.

Borrow Item: Implementation of Adding User to the waiting list, if the book is not available.

Assigning user to the book, when the book becomes available.

References

Asg1.6231w19.pdf

Lecture Note: Remote Invocation and Java RMI

https://way2java.com/rmi/java-rmi-architecture/

https://www.tutorialspoint.com/java_rmi/java_rmi_introduction.htm