UNIVERSITÉ Concordia UNIVERSITY

DEPARTMENT OF
COMPUTER SCIENCE AND SOFTWARE ENGINEERING

COMP 6231 - DISTRIBUTED SYSTEM DESIGN

WINTER 2019

WEB SERVICE IMPLEMENTATION OF THE DISTRIBUTED LIBRARY MANAGEMENT SYSTEM

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. All the above said is to be implemented as a web service. This document presents the designs, methods and architecture used in implementing the project.

Terminology & Definitions

Abbreviations

DLMS: Distributed Library Management System

SOAP: Simple Object Access Protocol

WSDL: Web Service Definition Language

XML: Extensible Markup Language

HTTP: Hyper Text Transfer Protocol

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.
Exchange Item	This allows the user to exchange the borrowed item with any other item from same or other libraries.

Web Service

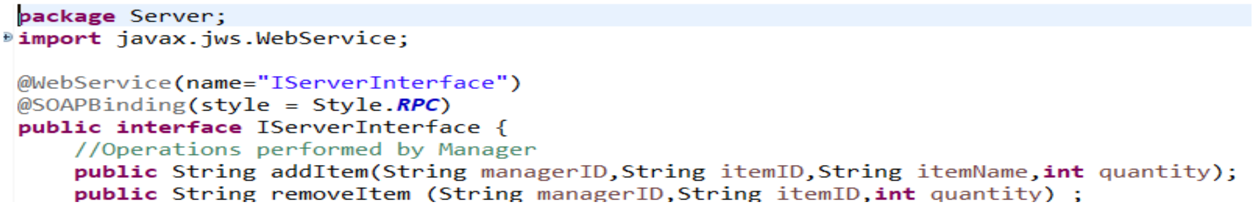
Web Services are software components described via WSDL which are capable of being accessed via standard network protocols such as SOAP over HTTP.

Web services provide interoperability between various software applications running on various platforms.

Detailed Architecture and Implementation

Create a dynamic web project that consists of client and three servers namely, CON, MCG and MON. All the servers perform same type of operations.

The functionalities mentioned above will be defined in interface file called as Service Endpoint Interface (SEI).

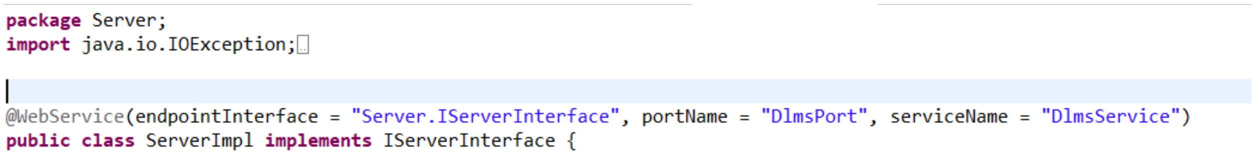


```
package Server;
import javax.jws.WebService;

@WebService(name="IServerInterface")
@SOAPBinding(style = Style.RPC)
public interface IServerInterface {
    //Operations performed by Manager
    public String addItem(String managerID,String itemID,String itemName,int quantity);
    public String removeItem (String managerID,String itemID,int quantity) ;
```

The implementation of the defined functions is added in ServerImpl Class.

This class is annotated as webservice endpoint using @WebService annotation, and the methods with @WebMethod annotation.



```
package Server;
import java.io.IOException;

@WebService(endpointInterface = "Server.IServerInterface", portName = "DlmsPort", serviceName = "DlmsService")
public class ServerImpl implements IServerInterface {
```

Later we navigate to the path which contains class files e.g., c://path/bin and the following command is executed.

wsgen -verbose -cp . Server.ServerImpl -wsdl

This generates .xsd and .wsdl files.

Later navigate to the path where the above files were generated and execute the following command.

wsimport -keep -d . -p ClientStub DlmsService.wsdl

This will generate the port,through the call can be made.It acts as a proxy for the remote service.

We can view the wsdl files for each webservice by using the urls,after the server and client are started.

For CON Server: http://localhost:2121/CON?wsdl

For MCG Server: http://localhost:2122/MCG?wsdl

For MON Server: http://localhost:2123/MON?wsdl

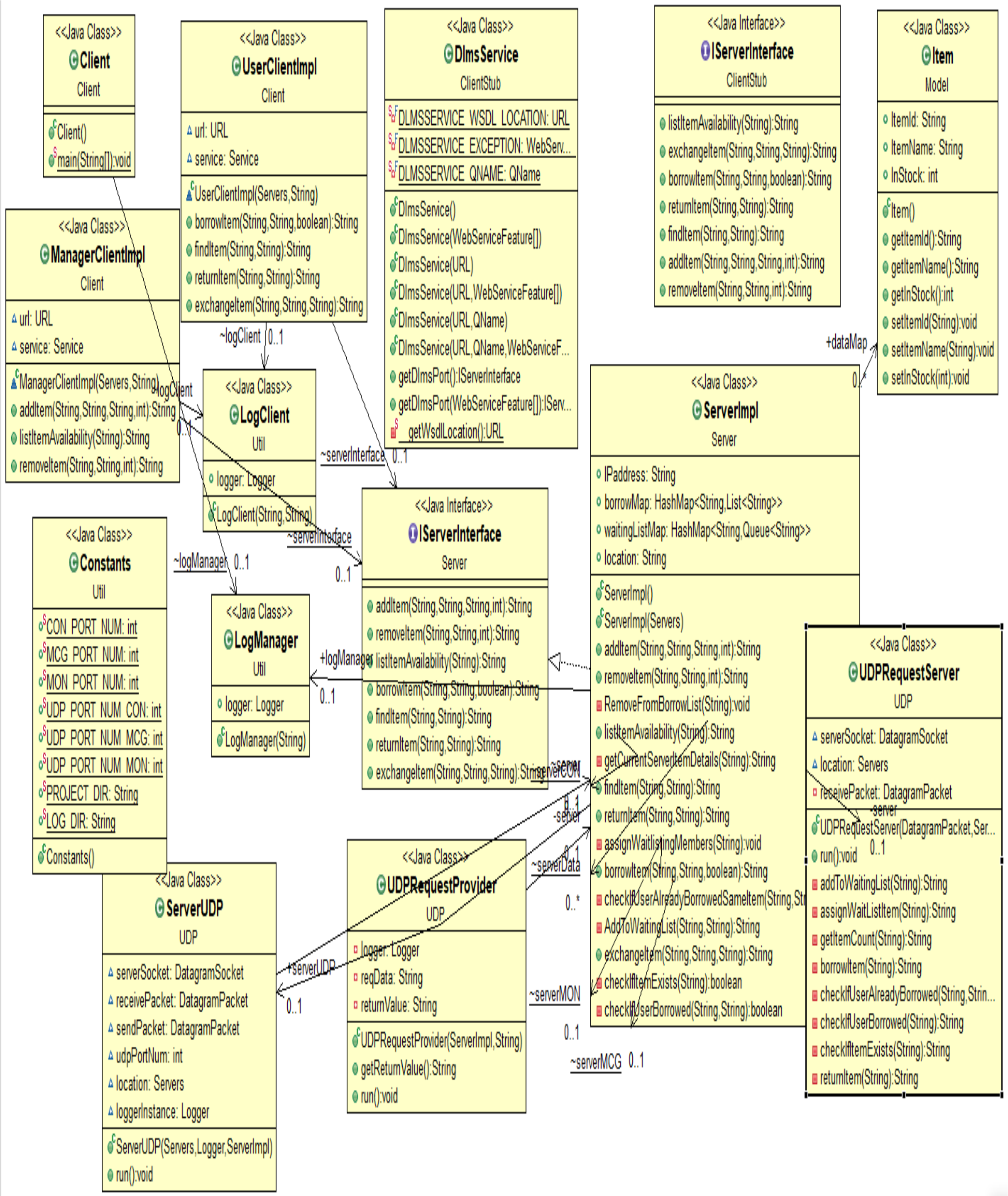
We also have ManagerClient and UserClient to perform their respective functionalities.

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

Server to Server communication is maintained by UDP.

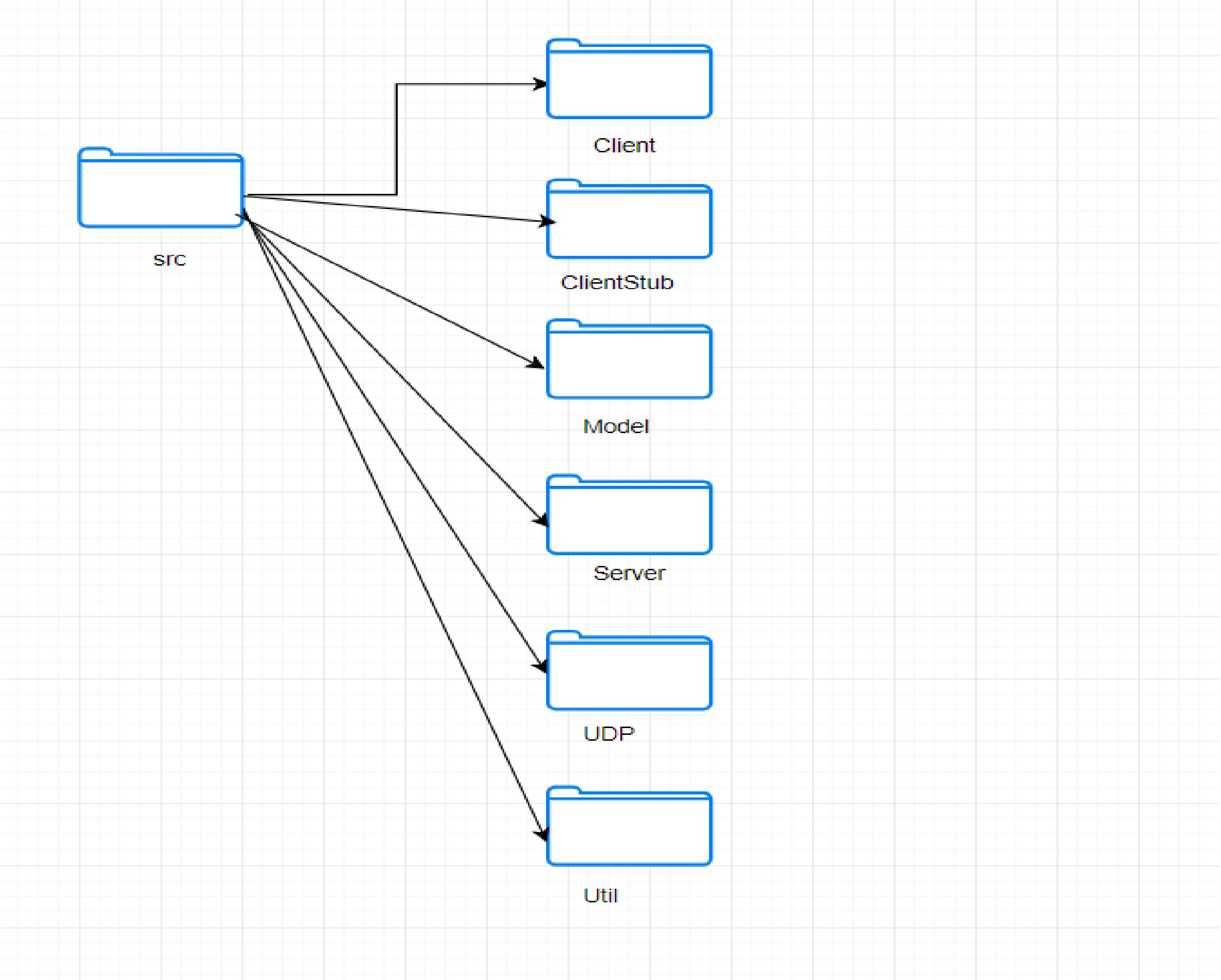
Detailed System Design

Class Diagram



<<Java Class>>
Client
Client
Client()
main(String[]):void
<<Java Class>>
UserClientImpl
Client
url: URL
service: Service
UserClientImpl(Servers,String)
borrowItem(String,String,boolean):String
findItem(String,String):String
returnItem(String,String):String
exchangeItem(String,String,String):String
<<Java Class>>
DlmsService
ClientStub
DLMSSERVICE_WSDL_LOCATION: URL
DLMSSERVICE_EXCEPTION: WebServ...
DLMSSERVICE_QNAME: QName
DlmsService()
DlmsService(WebServiceFeature[])
DlmsService(URL)
DlmsService(URL,WebServiceFeature[])
DlmsService(URL,QName)
DlmsService(URL,QName,WebServiceF...
getDlmsPort():IServerInterface
getDlmsPort(WebServiceFeature[]):IServ...
__getWsdlLocation():URL
<<Java Interface>>
IServerInterface
ClientStub
listItemAvailability(String):String
exchangeItem(String,String,String):String
borrowItem(String,String,boolean):String
returnItem(String,String):String
findItem(String,String):String
addItem(String,String,String,int):String
removeItem(String,String,int):String
<<Java Class>>
Item
Model
ItemId: String
ItemName: String
InStock: int
Item()
getItemId():String
getItemName():String
getInStock():int
setItemId(String):void
setItemName(String):void
setInStock(int):void
<<Java Class>>
ManagerClientImpl
Client
url: URL
service: Service
ManagerClientImpl(Servers,String)
addItem(String,String,String,int):String
listItemAvailability(String):String
removeItem(String,String,int):String
~logClient
0..1
~logClient
0..1
<<Java Class>>
LogClient
Util
logger: Logger
LogClient(String,String)
~serverInterface
0..1
~serverInterface
0..1
+dataMap
0..*
<<Java Class>>
ServerImpl
Server
IPaddress: String
borrowMap: HashMap<String,List<String>>
waitingListMap: HashMap<String,Queue<String>>
location: String
ServerImpl()
ServerImpl(Servers)
addItem(String,String,String,int):String
removeItem(String,String,int):String
RemoveFromBorrowList(String):void
listItemAvailability(String):String
getCurrentServerItemDetails(String):String
findItem(String,String):String
returnItem(String,String):String
assignWaitlistingMembers(String):void
borrowItem(String,String,boolean):String
checkIfUserAlreadyBorrowedSameItem(String,St...
AddToWaitingList(String,String):String
exchangeItem(String,String,String):String
checkIfItemExists(String):boolean
checkIfUserBorrowed(String,String):boolean
<<Java Class>>
Constants
Util
CON_PORT_NUM: int
MCG_PORT_NUM: int
MON_PORT_NUM: int
UDP_PORT_NUM_CON: int
UDP_PORT_NUM_MCG: int
UDP_PORT_NUM_MON: int
PROJECT_DIR: String
LOG_DIR: String
Constants()
~logManager
0..1
<<Java Class>>
LogManager
Util
logger: Logger
LogManager(String)
+logManager
0..1
<<Java Interface>>
IServerInterface
Server
addItem(String,String,String,int):String
removeItem(String,String,int):String
listItemAvailability(String):String
borrowItem(String,String,boolean):String
findItem(String,String):String
returnItem(String,String):String
exchangeItem(String,String,String):String
~serverCON
0..1
-server
0..1
~serverData
0..*
~serverMON
0..1
~serverMCG
0..1
<<Java Class>>
UDPRequestServer
UDP
serverSocket: DatagramSocket
location: Servers
receivePacket: DatagramPacket
-server
0..1
UDPRequestServer(DatagramPacket,Ser...
run():void
addToWaitingList(String):String
assignWaitListItem(String):String
getItemCount(String):String
borrowItem(String):String
checkIfUserAlreadyBorrowed(String,Strin...
checkIfUserBorrowed(String):String
checkIfItemExists(String):String
returnItem(String):String
<<Java Class>>
ServerUDP
UDP
serverSocket: DatagramSocket
receivePacket: DatagramPacket
sendPacket: DatagramPacket
udpPortNum: int
location: Servers
loggerInstance: Logger
ServerUDP(Servers,Logger,ServerImpl)
run():void
+serverUDP
0..1
<<Java Class>>
UDPRequestProvider
UDP
logger: Logger
reqData: String
returnValue: String
UDPRequestProvider(ServerImpl,String)
getReturnValue():String
run():void

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	As Expected

		removed from the HashMap.	
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	An error message will be displayed as “User can	As Expected

	library and requests for another book from the same library	borrow only one book from other libraries”.	
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
T027	Exchange Item: User wants to exchange an item with another item in the same/other library,	An error message “Sorry, cannot process the request currently. Please try again!!” will be displayed	As Expected

	and the new item to be borrowed doesn’t exist.		
T028	Exchange Item: User wants to exchange an item with another item in the same/other library, and the item already borrowed doesn’t exist.	An error message “Sorry, cannot process the request currently. Please try again!!” will be displayed	As Expected
T029	Exchange Item: User wants to exchange an item with another item in the same/other library	The new item will be borrowed, and old item will be returned respectively.	As Expected

Challenges

Understanding and Creation of WSDL files.

Usage of SOAP Binding failed to generate the WSDL files.

Usage of commands was difficult to understand.

References

- Asg3.6231w19.pdf
- WebServices lab tutorial
- https://docs.oracle.com/javase/8/docs/technotes/tools/unix/wsgen.html