Concordia University

Department of Software Engineer and Computer Science

COMP 6231 Distributed System

Fall 2017

Assignment 3

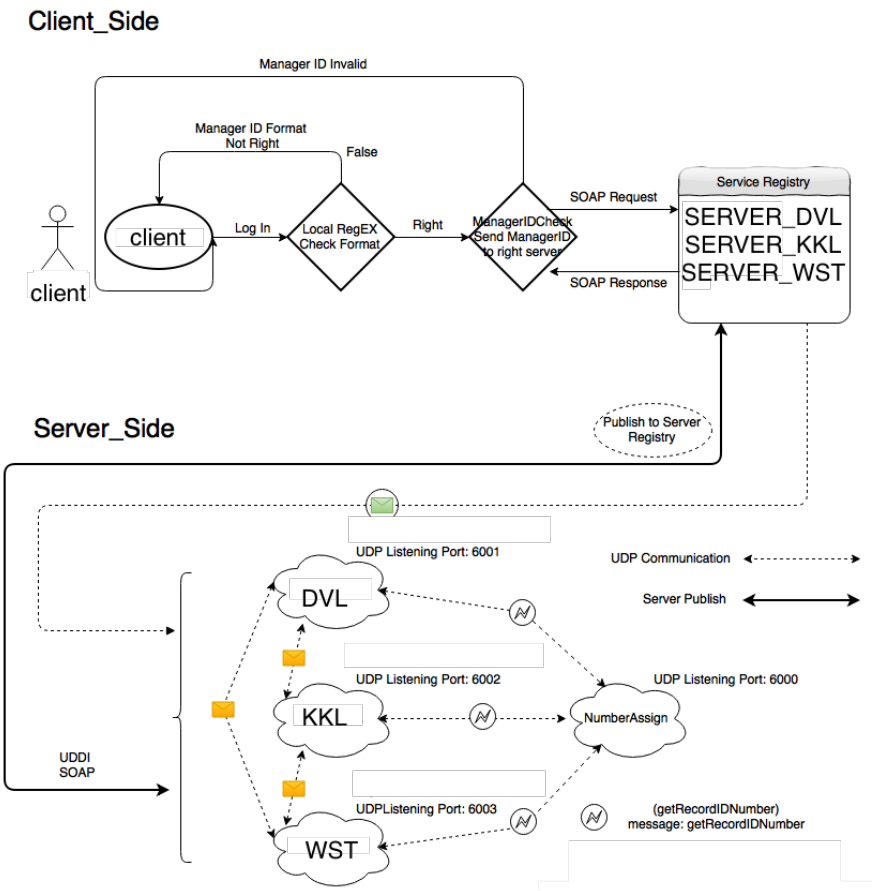
Mengran Li: 40022244

- ## About this project:

In this project, we are asked to enhance implementation of the Distributed Room Reservation System (DRRS) developed by web service.

To complete this project, I use IntelliJ IDEA with JRE1.8.0_152. The core techniques of this DRRS system is using web service and UDP programing.

- ## Design & Architecture



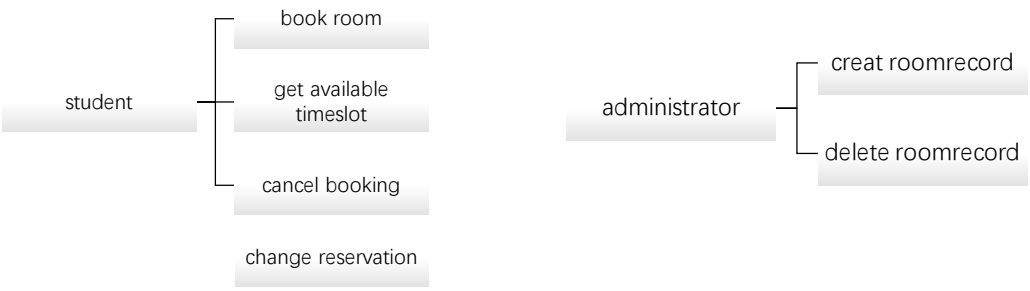
- ## Design of DRRS

Step 1: Start server and udp listener;

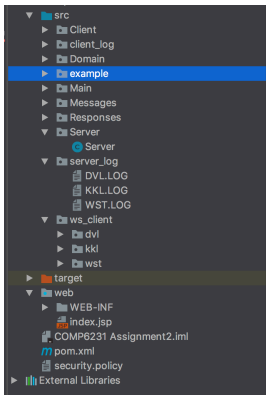
Step 2: open user terminal and administrator terminal. Require student or administrator to input the ID as a log in function.

Step 3: Using regular expression to check the format of ID. If format is right, then go to Step 3, else go back to Step 1. Step 3: Get the prefix of userID then send userID to specific server to check valid of account. If return value is valid, then go to Step 4, else go back to Step 1.

Step 4: Client get the right stub of server, can do some functions such as creating room record and delete room record for administrator; book room, get available time slot, cancel booking and change reservation for student.



About the design of this DRRS, we have multiple packages like below:



▪ Implementation of web service

Server_Side.Server_DVL, Server_Side.Server_KKL, Server_Side.Server_WST:

This three package are different server of DVL,KKL and WST. Each package includes four file. (*.impl.java) is an implementation file implements the interface which is contained in Server_Side. (*.Server_Publish.java) is a file for publishing the server to service registry. (*.UDP_Listener.java) is a file for opening the UDP listener for receive the request from client or other server. The last one with prefix of Config_XXX.java is a config file of certain server.

Step 1: Create the interface for the implementation class

Mark interface to @WebService (Remember to import javax.jws.WebService)

Mark five method to @WebMethod (Remember to import javax.jws.WebMethod)

Step 2: Code the implementation class. Mark the class to @WebService and overwrite the method of interface.

Step 3: Use wsgen to generate wsdl and xds file.(Please go to the project bin directory before use wsgen)

wsgen -verbose -cp . Server_Side.Server_WST.Clinic_WST_Impl –wsdl

wsgen -verbose -cp . Server_Side.Server_KKLClinic_KKL_Impl –wsdl

wsgen -verbose -cp . Server_Side.Server_DVL.Clinic_DVL_Impl -wsdl

Step 4: publish the web service and test it

Endpoint endpoint = Endpoint.publish("http://localhost:9001/server_WST", new Clinic_WST_Impl());

Endpoint endpoint = Endpoint.publish("http://localhost:9002/server_KKL", new Clinic_KKL_Impl());

Endpoint endpoint = Endpoint.publish("http://localhost:9003/server_DVL", new Clinic_DVL_Impl());

Step 5: Before code the client use wsimport download the wsdl to local.

wsimport -keep -d . -p Client_Side.ManagerClients http://localhost:9001/server_WST?wsdl

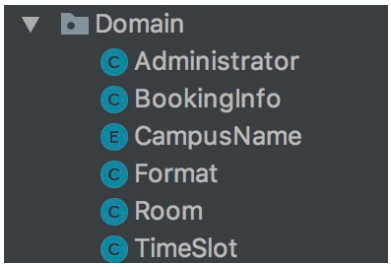
wsimport -keep -d . -p Client_Side.ManagerClients http://localhost:9002/server_KKL?wsdl

wsimport -keep -d . -p Client_Side.ManagerClients http://localhost:9003/server_DVL?wsdl

Step 6: Code the client class and run this.

- **object-oriented programming**

In Domain section, there are 6 classes to define objects.



(1) Administrator

Data: list of administrators and Campus name.

Function: Check administratorID();

(2) BookingInfo

Data: Campusname, studentCampusname, studentID, bookingDate, roomNumber, bookingStarttime, bookingEndtime

Function: BookingInfo decode ();

encode BookingInfo();

(3) CampusName

Data: campusname, campusabrev, port, servername, inport

Function: getCampusname();

(4) Format

Function: format time ();

format date ();

(5) Room

Data: roomnumber, timeslots

Function: Addtimeslots();

Removetimeslotes();

Getroomnumber();

(6) Timeslot

Data: starttime, endtime, StudentID, studentCampus, bookingID

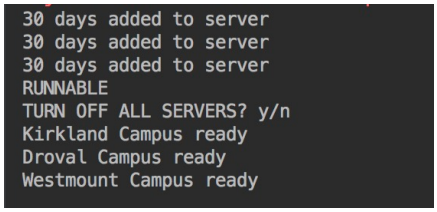
Function: set start/end time ();

setStudentID ();

- **Test part**

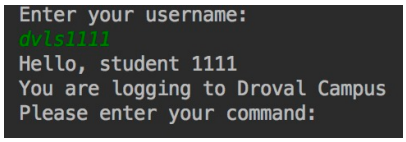
Start test:

Step 1: Start server

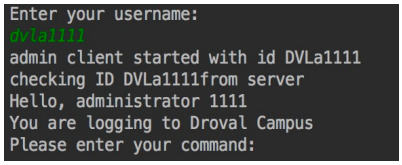


Step 2: open administrator or student terminal and log in with right ID

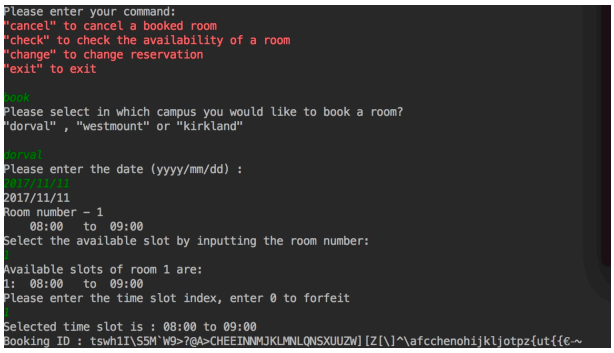
Student:



Administrator:



Step 3: Test functions (for example: bookroom)

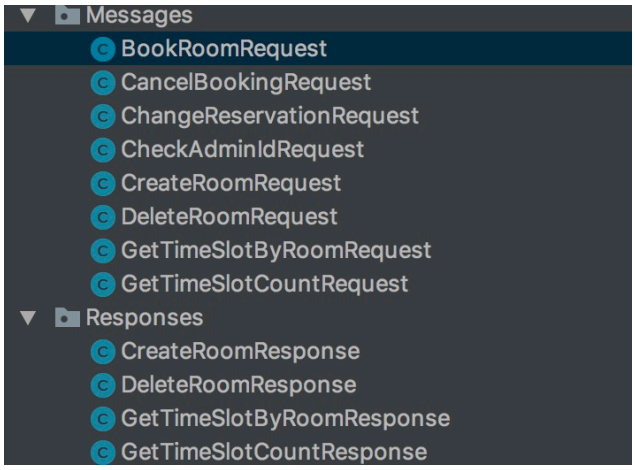


- **Implementation**

Part1. Implementation of webservice

Part2. Use json to modify parameter type.

In CORBA architecture, all functions need to modify parameter types from object to String, Boolean or int. There are two packages designed to change objects to string json and get return value from string json to objects.



Part 3: Use multi thread to handle the request.

If we use single thread to handle the request of server or client, the performance of that server will be very slow, because each time the server can only deal with one request, after reply succeed then will handle another request.