Strathmore
UNIVERSITY

SCHOOL OF COMPUTING AND ENGINEERING SCIENCES

Bachelor of Science in Informatics and Computer Science

Bachelor of Science in Telecommunications

BTC4201 / ICS4104 – Distributed Systems

Assignment – Inter-process Communications in Distributed Environment (Worth 16%)

Assignment Deadline: 6th August 2021 @ 5.15 pm

A. Assignment Questions

Sockets and RMI are two techniques in Java which could be used to establish two-way communication between two running programs (typically a client and a server) running on the network. But in principal, they work in quite different ways. A socket is just a way to send data (only data, not methods) on a port to a different host; it's up to you to define your own protocol.

RMI is a technology in which the methods of remote Java objects can be invoked from other Java virtual machines running on the different hosts. In this assignment, you will implement client-server communication programs based on both Java Socket and RMI techniques. In this Assignment we'll use plain sockets.

Learning about Sockets

Read the following resource carefully to learn about sockets:

http://java.sun.com/docs/books/tutorial/networking/sockets/index.html

If you never did any network programming in Java, it may be a good idea to read some more: http://java.sun.com/docs/books/tutorial/networking/index.html

You may want to have a look at other information:

http://www.javaworld.com/javaworld/jw-12-1996/jw-12-sockets.html

B. Assignment Tasks:

You will need to write client and a server programs to facilitate interactions using RPC techniques for the following tasks for Client and Server respectively:

Your client should be able to

1. Send your student number to the server program
2. Send your student name (first name, surname etc.) to the server program
3. Send your student faculty, course and degree to the server program
4. Send a thank you message with personal code (Innovate☺) to the server program.
5. Send all the above in one instruction to the server program

Your server should be able to

1. Ask the client program to send the student number
2. Ask the client program to send the student name (first name, surname etc.)
3. Ask the client program to send the student faculty, course and degree to the server program
4. Ask the client program to send the personal code (Innovate ☺)
5. Ask the client program to send all the above in one instruction to the server program.
6. Send the client program a message to indicate the communication succeeded or aborted.

Note: How to design the *server-client communication protocol* is up to you but you must ensure that you have a GUI for interactions between the Clients and Servers.

Implementation:

You have to implement in at least four (4) java classes namely:

1. *SocketClient.java*: This class connects to the server and communicates with it according to the Client Protocol
2. *ClientProtocol.java*: This class describes how your client reacts to your servers' messages (see the "Knock-Knock-Protocol in the Java Tutorial)
3. *SocketServer.java*: This class sets up a socket listening for your client to connect - and communicates with your client
4. *ServerProtocol.java*: This class describes how your server reacts to the messages of your client

Additional Information:

You are only required to test your server and client communication locally. When you create a socket, use **localhost** instead of IP address and port number. We encourage each group (**max. of 4 persons**) to run your server and client programs on different machines and do ensure port number should be dynamic noting that some ports might not be available on certain clients or servers' machines. You can use any RPC implementation or Remote Java implementation of ONC/RPC.

C. Group Composition and Report

- Assignment Group can only have a maximum of four (4) students (but it can less ☺ ☺)
- Submission Report and respective code should be submitted through GitHub by sending me a link.
- Make sure your programs work correctly by running clients and servers' locations. The codes should include well documented comments or short write-ups. The client and server programs should have separate code which the group member(s) has/have to demonstrate. Please supply the files in the following order:

  - Implementation files for the classes or functions (you will have to demonstrate your implementations);
  - A main program for the client and the server
  - Log information about what steps you exactly you took to produce the working code. (Keep a diary of events which you will hand in with the program code!!)

D. Reference Texts

a. Coulouris, G., Dollimore, J. & Kindburg, T.: *Distributed Systems, Concepts and Design*. Addison Wesley. ISBN 0-201-61918-0 (4th or 5th Editions).

b. Tanenbaum, A. S. & van Steen, M: *Distributed Systems: Principles and Paradigms*. Pearson Education International/Prentice-Hall International. ISBN 0-13-121786-0. (2nd Edition).

E. References Listings

For all the consulted any references for the assignment, cite/list them using the APA style. Check these URLs for guidance:

a. http://www.library.cornell.edu/resrch/citmanage/apa

b. https://apastyle.apa.org/6th-edition-resources/basics-tutorial

c. https://guides.library.ubc.ca/howtocite