Lab 10

[valid 2020-2021]

Networking
Create an application where clients connect to a server in order to form a *social network*. The application will contain two parts (create a project for each one):

- The **server** is responsible with the management of the clients and the implementation of the services.
- The **client** will communicate with the server, sending it *commands* containing the name of the service and the required parameters. The commands are:
    - register *name*: adds a new person to the social network;
    - login *name*: establishes a connection between the server and the client;
    - friend $name_1$ $name_2$ ... $name_k$: adds friendship relations between the person that sends the command and other persons;
    - send *message*: sends a message to all friends.
    - read: reads the messages from the server.

The main specifications of the application are:

Compulsory (1p)

- Create the project for the server application.
- Implement the class responsible with the creation of a *ServerSocket* running at a specified port. The server will receive requests (commands) from clients and it will execute them.
- Create a class that will be responsible with communicating with a client *Socket*. The communication will be on a separate thread. If the server receives the command *stop* it will stop and will return to the client the respons "Server stopped", otherwise it return: "Server received the request ... ".
- Create the project for the client application.
- A client will read commands from the keyboard and it will send them to the server. The client stops when it reads from the keyboard the string "exit".

Optional (2p)

- Create an object-oriented model for your application and implement the commands.
  The command *stop* should "gracefully" stop the server - it will not accept new games but it will finish those in progress. When there are no more games, it will shutdown.
- Implement a timeout for a connection (a number of minutes). If the server does not receive any command from a logged in client in the specified period of time, it will terminate the connection.
- Create an HTML embeddable representation of the social network using JFreeChart, JGraphT and Apache Batik, or other technology.
- Upload the HTML representation directly from the application to a Web server. You may use JCraft for connecting to a server using SFTP and transferring a file (or a similar solution).

Bonus (2p)

- Implement an algorithm to determine the structural cohesion of the network.
- Rewrite the application so it can act both as a server or a client, depending on circumstances.
  Suppose that the clients (agents) are inside an intranet and their IP addresses are known (or within a known limited range).
  Implement a leader election algorithm such that, if the server (the coordinator agent) fails, one of the other agents "elects" itself as the coordinator of the network and the communication can still continue.

Resources

- Custom Networking
- Remote Method Invocation (RMI)
- Java Networking

Objectives

- Understand terms and concepts related to networking: protocol, IP, port, URL, socket, and datagram.
- Get familiar with the client-server programming model.

- Write programs that communicate with other programs on the network, using TCP or UDP.
- Get acquainted with RMI technology.