Contact

Prof. Sara Bouchenak

Sara.Bouchenak@insa-lyon.fr http://liris.cnrs.fr/sara.bouchenak/

Building Socket-based distributed systems

1. Objectives

- Building distributed systems
- Synchronous network communication systems
- Introduction to Java Sockets

2. Prerequisites

- Java programming

3. Description of work

a) Run the provided examples of mono-threaded programs: first, the server program, then, the client program.

b) Extend the server program and the client program with additional operations, e.g., output a message for the user on the client, output a message on the server side.

c) Stop all runs. Run again the server program. Then, run the client program. And run again another instance of the client program. What do you observe?

d) Run the provided examples of multi-threaded programs: first, the multi-threaded server program, then, the client program.

We consider a distributed system that represents a chat system where users can dynamically join, leave, and exchange messages.

e) Specify the distributed architecture, design principles, and communication protocol of a chat system based on TCP sockets:

- Describe the distributed architecture of the chat system and its different components: server, clients.

- Describe all the sockets used at each component of the distributed system: at the server side, at each client side.
- Describe all the network directed communication channels connecting two communicating components.
- Describe all the threads involved at the server side, and the threads involved at the client side.
- Describe the communication flow between all these components when a participant sends a message to the chat.
- Remember that whenever two tasks may be executed in parallel, they should be run by two threads.
- Remember that two threads (of the same process/Java virtual machine) may share global variables.

f) Describe the implementation details of the chat system using TCP sockets.

g) Build the TCP socket-based chat system.

h) Specify the distributed architecture, design principles, and communication protocol of a chat system based on multicast/UDP:

- Describe the distributed architecture of the chat system and its different components.
- Describe all the sockets used at each component of the distributed system.
- Describe all the network communication channels connecting communicating components.
- Describe all the threads involved on all the components.
- Describe the communication flow between all these components when a participant sends a message to the chat.
- Remember that whenever two tasks may be executed in parallel, they should be run by two threads.
- Remember that two threads (of the same process/Java virtual machine) may share global variables.

i) Describe the implementation details of the chat system using multicast/UDP.

j) Implement a distributed chat system with group communication multicast/UDP.

k) Add a history to the chat system: a new user who joins the system first receives the messages exchanged before by other users. This first version of the history is ephemeral, i.e., when the chat system is stopped, the history of messages is lost.

l) Implement a persistent history of exchanged messages, i.e., when the chat server is stopped and restarted, the history of exchanged messages is restored.

m) Implement a graphical user interface for the chat system.

4. Software environment

Used software

- Java 2 SDK
- IDE (e.g. Eclipse)

Directory organization

Organize your directory as follows:

- src/
    - .java files
- classes/
    - .class files
- doc/
    - Javadoc API and HTML files
- lib/
    - .jar files