King Abdulaziz University
Faculty of Computing and Information Technology
Jeddah, Saudi Arabia

Winter 2022
CPCS 371-Computer Networks 1

Course Project Phase II

Simulating a Remote Health Monitoring System for elderly patients using client-server TCP Sockets

Assigned Date: Monday 23 Jummadah (2) [16-1-2023]

Project Objective:

The objective of this project is to help students understand the basics of client-server internet applications and to be able to develop their own client-server applications based on TCP socket programming.

Covered CLO:

- [CLO#5] Implement a simple client-server socket-based application.

Possible points: 8 marks

Due Date:

- The deadline for submitting the project application and report is on **Week#10 (Thursday, 18 Rajab 1444 [9-Feb-2023] @8:00 AM).**
- The project demo and discussion will be on the same day **Week #10 (TBA).**

Instructors:

Dr. Etimad Fadel (Coordinator + Instructor), and Dr. Ohoud Alzamzami (Instructor).

Project Instructions:

- This is a group project with *EXACTLY* **6** members in each project group.
- The project includes three main tasks, which are: **implementing** a **client-server application**, **writing** a **professional project report**, and **project discussion** which includes a **demo** of the application and a discussion of the **report**.
- The client-server application should be written in **Java** programming language.
- The implementation of the Remote Health Monitoring System (RHMS) application is based on the client-server architecture, and it should use **TCP sockets**.
- *A bonus of 1.5 point will be given on using **additional GUI** elements for implementing the application. In addition to the application windows and planes, the elements that can be used are images, radio buttons, test Fields and buttons.*

Project Description:

Many elderly patients suffer from chronic diseases, which require costly long time care services. Wireless Body Area Networks (WBAN) enable remote monitoring of patients with chronic diseases, thereby, improving the elderly patients quality of life and reducing the medical care cost. In this project, you are going to implement a simulation of a typical Remote Health Monitoring System (RHMS) which uses WBAN. RHMS is based on the client-server architecture. The system you are required to implement considers using three types of sensors, which are heart rate, oxygen saturation, and temperature, for monitoring the health of elderly patients. The RHMS consists of (3) main components as shown in Figure 1:

1. **The Sensor Client Application:** which collects the sensors' data and sends them to the server side of the **Personal Server** via TCP sockets.
1. **The Personal Server:** which has two roles. It acts as a server for the **Sensor Client Application** that processes the received sensor information and decides whether or not this information needs to be sent to the caregiver. If the processed information needs to be sent to the caregiver, the client side of the **Personal Server** sends the appropriate messages to the **Medical Server**.
2. **The Medical Server:** which receives messages about the patient status from the personal server. The medical server displays the received messages from the **Personal Server** to the caregiver. In addition, it processes the received information and displays the appropriate action to be taken by the caregiver.

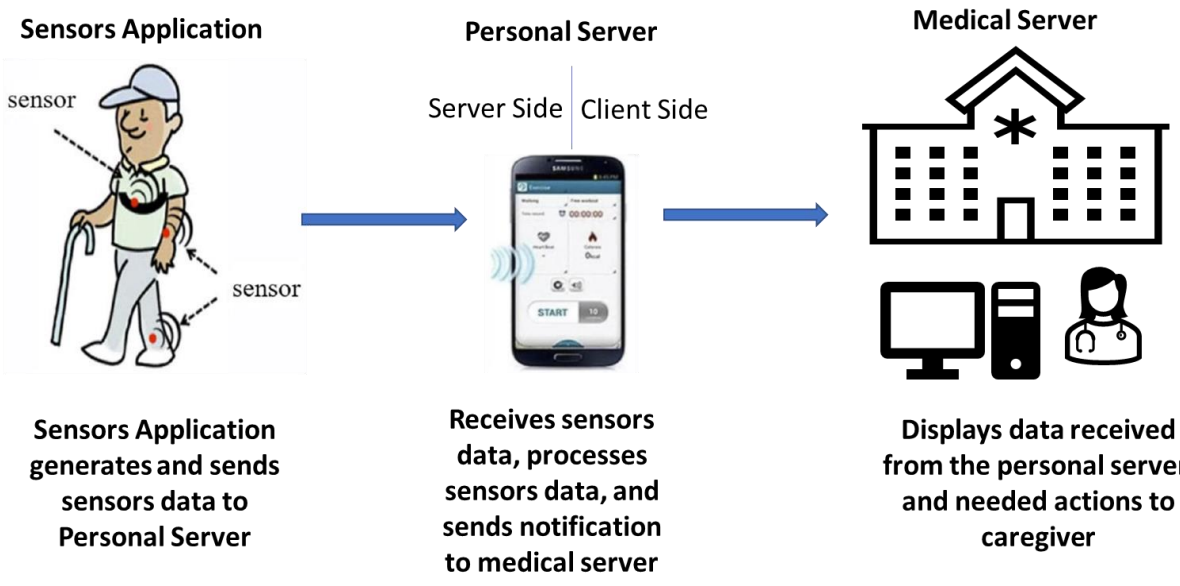

Figure 1: Remote Health Monitoring System (RHMS) Application Architecture

Description of the RHMS Operation Steps:

1. Three types of sensors are installed on the elderly patient body which are heart rate, temperature, and oxygen saturation sensors. For simulating the generated data by these sensors the **Sensor Client Application** must generate them randomly**.** The random numbers are generated depending on the sensor type according to the following ranges:
    a. Temperature sensor data is between 36C and 41C.
    b. Heart rate sensor is between 50 and 120 heart beats each minute (bpm).
    c. Oxygen level in the blood data is sent by the sensor with values between 60% to 100%.
2. The sensors send their sensed data every 5 seconds to a **Personal Server** application installed on the patient's mobile phone.
3. The **Personal Server** displays the received sensor's data on its application interface.
4. The **Personal Server** application processes the received sensor data and decides based on the processed data if the medical caregiver should be notified. The personal server contacts the caregiver server in any of the following cases and send the appropriate messages based on the following cases:
    a. If the temperature exceeds 38, the message to be sent is "At date: <current date>, time: <current time>,*Temperature is high <current temperature>.*"

b. If the heart rate exceeds 100, the message to be sent is “At date: <current date>, time: <current time>, *Heart rate is above normal <current heart rate>*”
c. If the heart rate is below 60, the message to be sent is “At date: <current date>, time: <current time>, *Heart rate is below normal <current heart rate>*”
d. If the oxygen saturation is below 75, the message to be sent is “At date: <current date>, time: <current time>, *Oxygen saturation is low <current measurement of oxygen>*”

5. At the **Medical Server,** the received messages from the **Personal Server** are displayed on the caregiver application interface.
6. The **Medical Server** processes the received measurement notifications and based on processing this data decides on the appropriate action. The following output is displayed on the caregiver interface:
    a. If the temperature exceeds 39 and heart rate is above 100 and oxygen is below 95, the message to be displayed is “*Send an ambulance to the patient!*”.
    b. If the temperature is between 38 and 38.9, and the heart rate is between 95 and 98, and oxygen is below 80, the message to be displayed is “*Call the patient's family!*”.
    c. Otherwise, the message to be displayed is “Warning, advise *patient to make a checkup appointment!*”
7. The RHMS Simulation Software execution must start and run as following:
    a. The **Personal Server** Application should have an infinite loop to accept connection requests from the **Sensor Client Application.**
    b. The **Medical Server** Application should have an infinite loop to accept connection requests from the **Personal Server.**
    c. The **Sensor Client Application** needs to run over a period of time. The minimum time of execution is (60) seconds which is equivalent to 12 readings. You can allow this value to be entered at runtime (e.g. user input or read from file).
    d. Finally, the **client** Personal Server runs when needed.

RHMS Example Output Examples:

Sensors Application Output
At date: 15 Jan 23, time 09:05:14, sensed temperature is 37

At date: 15 Jan 23, time 09:05:15, sensed heart rate is 96
At date: 15 Jan 23,time 09:05:16, sensed oxygen saturation is 96

At date: 15 Jan 23, time 09:05:17, sensed temperature is 39
At date: 15 Jan 23, time 09:05:18, sensed heart rate is 101
At date: 15 Jan 23, time 09:05:19, sensed oxygen saturation is 93

Personal Server Output

At date: 15 Jan 23, time 09:05:14, Temperature is normal
At date: 15 Jan 23, time 09:05:15, Heart rate is normal
At date: 15 Jan 23,time 09:05:16, Oxygen Saturation is normal

At date: 15 Jan 23, time 09:05:17, Temperature is high 39. An alert message is sent to the Medical Server.
At date: 15 Jan 23, time 09:05:18, Heat rate is above normal 101. An alert message is sent to the Medical Server.
At date: 15 Jan 23, time 09:05:19, Oxygen Saturations is low 93. An alert message is sent to the Medical Server.

Medical Server

At date: 15 Jan 23, time 09:05:17, Temperature is high 39.
At date: 15 Jan 23, time 09:05:18, Heat rate is above normal 101.
At date: 15 Jan 23, time 09:05:19, Oxygen Saturations is low 93.
ACTION: Send an ambulance to the patient!

NOTE: for formatting Date and Time any format could be used.

Project Report:

You should write a **professional, and well-organized report** which addresses the followings:

- The report should reflect your understanding of the project problem and the provided solution, code, including sockets, and GUIs elements.
- You will be graded on the writing style, clarity, and proper use of English language and networking terms.
- **The report should have the following structure:**
    1. **Introduction**

- Discusses client-server applications, Java TCP sockets, and the selected GUIs elements used in your application.
- Briefly describe the RHMS application and the role of the clients and servers in this application
- Give an overview of the of the remaining report sections (e.g. in section 2 the RHMS application interaction diagram will be presented, etc...)

2. **Patient Monitoring Application interaction diagram [Similar to Figure 2.30 in the book]**
   - Draw **one** interaction diagram which demonstrates the **interaction** between the **Sensor Client** with the **Personal Server** and the **Personal Server** with the **Medical Server** .
   - You should include a reference within the text to that diagram.
   - Use English-like sentences and pseudocode to demonstrate socket related activity of the client and server that communicate using TCP.
3. **RHMS implementation**
   - You should include in this section a listing of your code
   - Your code should include **comments** that demonstrate:
     - Where are TCP sockets used?
     - What are the functions used for sending and receiving information using sockets?
     - What are the GUIs elements used in your application?
4. **RHMS Application run snapshots**
   - Include snapshots of the **different cases** described in operation steps above. Also, show the run of the program on:
     - one machine
     - different machines
   - Make sure that you reference each snapshot and explain it within the report text
5. **Teamwork and Lessons learned**
   - How is the work divided among the group members?
   - How did you coordinate the work between the group members?
   - What are the difficulties encountered? And how did you overcome them?
   - What did you learn from this project?
6. **Conclusion**
   - Summarize your project

Project demo and discussion:

Your work will be demonstrated by presenting a demo of the application run and discussing the project report at the end of the semester. The project demo and discussion should be no more than 10 minutes and it should include:

- A demo of your application.
- Explanation of the main parts of your project report.
- Explanation of your code.

Supporting Material:

- You could find supporting material that would help you understand how to develop the client-server application using java in the following books, which include some examples on java client-server applications:
    - Liang, Y. Daniel. **Introduction to Java programming: comprehensive version.** Pearson Education, 2015. (This is CPCS 203 reference book, refer to chapter 31 on networking).
    - Deitel, P., & Deitel, H. (2011). Java How to program. Prentice Hall Press.
- A code for the hangman game with only one client can be found in the following link: https://github.com/codermango/Hangman
- The TicTacToe game is a good starting point for implementing your own applications. It includes all the different concepts needed for developing your applications, which are sockets, threads, and GUIs elements.
- You can find some samples the **interaction diagram** in the following sources:
    - Lecture slides: Chapter2_lesson6, slide 110.
    - CPCS 371 reference book page, figure 2.30, page 165.
    - Introduction to Java programming: comprehensive version, figure 31.13, page 1157.

Strategy to work on the project:

To understand and master the concepts of sockets programming and threads in java, you could do the following:

- Make sure you understand the implementation of the simple client-server application done in part (1) of the project.
- Refer to the example TicTacToe game given in the Blackboard and the references.
- Make sure that you test all the different run cases mentioned in the operation steps by allowing **manual input** of sensor readings in addition to the randomized numbers.

Deliverables:

You are required to submit the project application and report files on Blackboard before the deadline as follows:

- The Source Code files in a directory with the name containing “Group#_RHMS”, e.g. “Group1_RHMS”.

- Project Report should also be named "Group#_RHMSReport.(Doc/Pdf)", e.g. "Group1_RHMSReport.(Doc/Pdf)".

Bonus:

A bonus of 1.5 point will be given for designing the patient monitoring application using GUI elements such as images, buttons and radio buttons, etc..

Grading Policy:

- The project will be graded on the correct functionality, application run, solution design, teamwork, quality of project report, and code documentation.
- Code documentation includes appropriate use of comments and indentation for readability, good naming conventions for variables, constants, and methods. Your code should also be well structured (i.e. not all code should be in the main method).
- Any form of plagiarism will result in receiving ZERO in the project.
- Note that it is fine to search and study public code and algorithms, which are available on the internet. However, it is important to write the application code by yourself. DO NOT COPY AND PASTE CODE. If you use a code from any reference, please cite that reference in your code using comments!

Rubric:

Find the below rubric used in grading each member of the group project:

Criteria	Unacceptable	Poor	Good	Excellent
Program Solution [SO-C]	**Unacceptable (1):** An incomplete solution is implemented on the required platform. It does not compile and/or run.	**Poor (2):** A completed solution is implemented on the required platform and uses the compiler specified. It runs but has logical errors.	**Good (3):** A completed solution is tested and runs but does not meet all the specifications and/or work for all test data.	**Excellent (4):** A completed solution runs without errors. It meets all the specifications and works for all test data.
Program Design [SO-C]	**Unacceptable (1):** Few of the selected structures are appropriate. Program elements are not well designed.	**Poor (2):** Not all of the selected structures are appropriate. Some of the program elements are appropriately designed.	**Good (3):** The program design generally uses appropriate structures. Program elements exhibit good design.	**Excellent (4):** The program design uses appropriate structures. The overall program design is appropriate.

Teamwork / Participation	**Unacceptable (0):** Student did not participate in group work.	**Poor (0.25):** Student did not participate fully in group work. Student needed reminders from team member.	**Good (0.5):** Student mostly participated in group work. Student did what was required.	**Excellent (1):** Student fully participate in group work. Went above and beyond to help group members succeed.
Code Readability, Format and Style of report	**Unacceptable (0):** Insufficient program documentation, incorrect indentation, and/or poor identifier selection.	**Poor (0.25):** Program is minimally documented; some identifiers are inappropriate or inconsistent indentation.	**Good (0.5):** Some required documentation is missing, or identifiers are inappropriate, or statements are not indented correctly.	**Excellent (1):** All required documentation is present, the program is correctly indented, and appropriate identifiers are selected

"Tell me and I forget, teach me and I may remember, involve me and I learn."

-Benjamin Franklin