Lab Manual: Socket Programming in Java

Lab Manual: Socket Programming in Java

Course: Operating Systems

Language: Java

Module: Inter-Process Communication (IPC) via Sockets

Lab Objectives

- Understand the concept of socket communication.
- Learn the difference between TCP and UDP protocols.
- Implement a client-server model using Java sockets.
- Explore synchronous and asynchronous communication.
- Analyze how Operating Systems manage socket-based communication.

Theory Overview

What is Socket Programming?

A socket is an endpoint for communication between two machines. Socket programming allows processes to communicate either on the same machine or across different machines on a network.

Types of Sockets

1. TCP (Transmission Control Protocol):
   - Connection-oriented
   - Reliable
   - Used for applications like HTTP, FTP

2. UDP (User Datagram Protocol):
   - Connectionless
   - Fast but not reliable
   - Used for streaming, DNS, VoIP

Experiment 1: TCP Socket Programming

Aim

To implement a simple TCP client-server program in Java.

See accompanying files: TCPServer.java and TCPClient.java

Experiment 2: UDP Socket Programming

Aim

To implement a UDP-based client-server communication.

See accompanying files: UDPServer.java and UDPClient.java

Lab Exercises

Exercise 1 (5 Marks):

Modify the TCP server to handle multiple client connections using threads.

- Each client should be able to send a message to the server.
- The server should respond with a confirmation message including the client's message.
- The server must not terminate after serving one client.

Exercise 2 (5 Marks):

Implement a simple UDP-based time server.

- The client sends a message "TIME" to the server.
- The server responds with the current system time.
- If the message is anything else, the server replies with "INVALID REQUEST".