

CS447-Network and Data Communication

Project I Specification, Summer 2026

Due: June 26th (the end of June 26th, 11:00 a.m.)

Project I - Narrative:

The goal of Project I is to develop a server process that communicates with multiple clients using sockets for Windows platform. The programs should be console-mode application (text-base, old MS-DOS type application) that should run on Win32 environment. The server process is supposed to be started at one of the PCs (Windows-based PCs) before any client makes a connection request to it. The source code for the client process will be provided by the instructor (Fujinoki). The client process is designed to take three parameters at the command prompt when one is started. The format of the three parameters at the command prompt is:

client <CLIENT ID#> <SERVER IP ADDRESSES> <SERVER PORT #>

Client ID: The client ID is the identification of clients that uniquely identify each of the multiple clients at the server. The client ID is simply an integer (e.g., 1, 2, ...) assigned to each client that talks to the same server (therefore the same client ID should not be assigned to more than one client).

Server IP address: The IP address of a server this client is going to talk to.

Server Port Number: The server process is listening to a particular TCP port at a network host with an IP address. In order for a client to talk to the server, both IP address and TCP port number must be specified at each client.

What this client and server application will perform:

After a client process is started (by entering the command-line prompt shown above), the client sends a connection request (using “connect” at the client) to the server process at a specified network host computer. Then, a TCP/IP connection will be established between the server and the client. At the server, each time the server receives a connection request from a client, the server process should display a message, “a connection request arrived ...” on its local display. Then, the server process extracts the client’s IP address and the TCP port number. After a socket connection is successfully established, the client process should send its client ID (this message from a client is called “the second message (from a client)”). The server process has a specific client called “prohibited client”. The client should send its client ID to the server and the server should receive the message (i.e., the client ID) through the socket connection and display the following message, “Request from client #”, where # is the ID of the client. After the server process displays the client ID, the server returns the timestamp (the timestamp at the arrival of the second message from each client) to the requesting client if the client is not the prohibited client. If it is the prohibited client, the server returns a message for rejection (“Your access is denied by

this server!”) to the client, instead of the timestamp message. The server process, then, should wait for one second after it returns the timestamp to the client. The client should receive the message from the server (either timestamp or rejection message) and display it on the client’s local display. After the client displays the message, the client should disconnect the socket connection and start the whole process from the beginning. Multiple clients (but up to three in this project) should be able to connect to the server at the same time (and the server should be able to handle up to three clients simultaneously). Each client should be designed to repeat this procedure 15 times and after 15 repeats, it should terminate itself. For each repeat, each client process will wait about one second. In the server side, execution should go back to “accept” function call. This procedure is visualized in the figure on the next page of this document.

Technical Requirements:

- a. Your source code file should be developed using MS Visual C++ (in Visual Studio) for Windows PCs. Program source code files written in any other programming language, for any other compilers, or for any other hardware platforms, will not be accepted for grading.
- b. Declaration of the prohibited client: the IP address of the prohibited client must be declared using `#define` declarative at the beginning of your source code file.
- c. Your server should communicate with the client process (client.cpp) posted to the CS447 home and the client process can not be modified.
- d. The messages displayed by the client process should not contain any extra contents.
- e. Your server process should be ready for serving multiple clients at any time (even when a client completes, your server must be up and running, ready for other clients).

Submissions:

The source code file for your server should be submitted ([email](mailto:hfujino@siue.edu) to: hfujino@siue.edu) for full credit.

Program soft copy – due at 11:00 a.m. on June 26th

- **server_XXX.cpp** source code file (while “XXX” is your last three digits).

Note: In your program source code, include your name, SIUE student ID in your program header.

Late Submission:

- Penalty of -10% will be given for every 12 hours after the due (i.e., -10% for a submission within the first 12 hours after the due).
- Submission more than 48 hours after the due will not be accepted.

Grading:

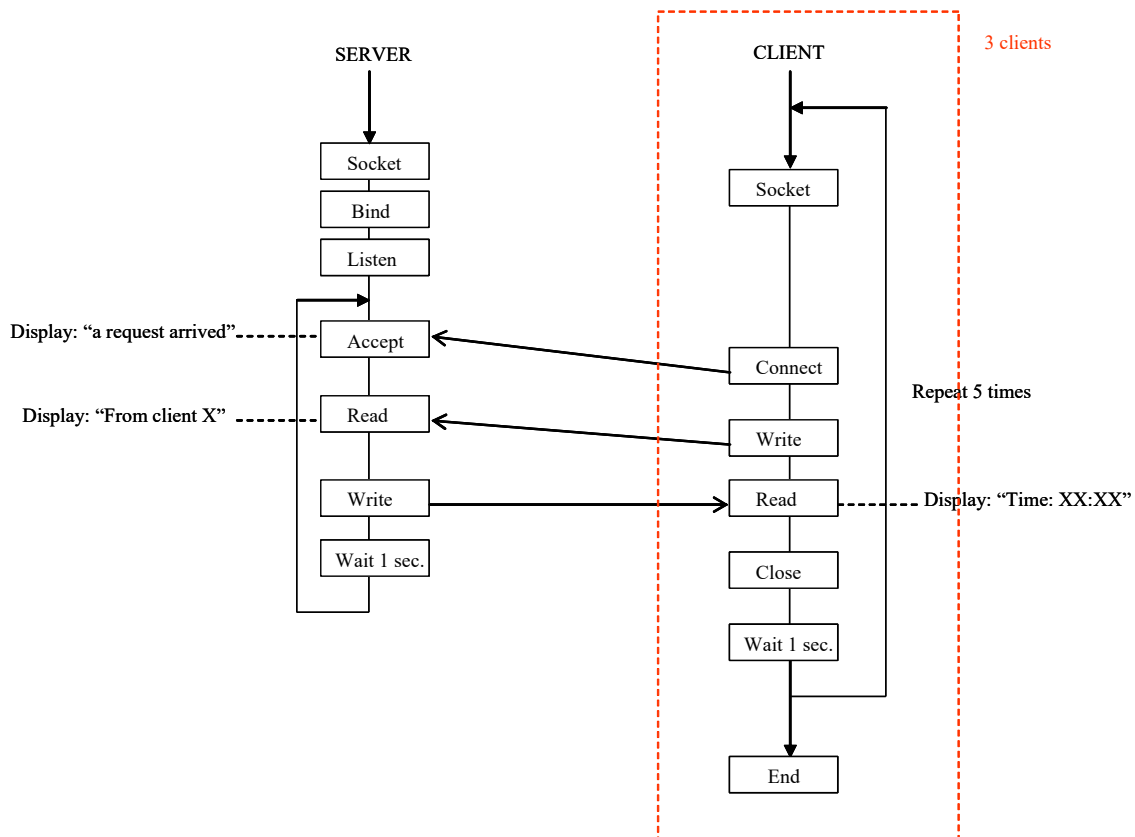
Your work will be graded by observing how it works using up to three PCs. Your server process is going to run at a PC, two clients will be run at another PC and the third PC will be used for the third client process.

- **Compile-time error:** If Dr. Fujinoki is unable to compile your c/C++ source code file due to any compile-time error: 0 ~ 20% of the credit for this assignment phase will be given.
- **Run-time error:** After your source code file is successfully compiled to an executable, if your program does not satisfy any requirement: 5% penalty for each minor problem. For any major failure, penalty depends on each such major problem.

Technical Issues: For socket connection to be established, the following specifications should be satisfied.

- (1) Network protocol: Internet Protocol (IP)
- (2) Connection type: connection-oriented socket (stream socket = TCP)

Required Process Structure:



Prohibited activities:

- This programming project is an individual project. Sharing an entire or any portion of a source code file (created by other people inside or outside of this course) is prohibited and it is considered an act of academic dishonesty.

Suggested references:

- For “ipconfig” and “netstat”:
- For W32 Winsock APIs: