COMP 3331/9331 Assignment T3 2020

Online Discussion Forum

All details are in the specification

- **READ THE SPECIFICATION**
- **READ THE SPECIFICATION (AGAIN)**
- Information about deadlines, file names, submission instructions, marking guidelines, example interactions and various other specifics are in the specification
- Choice of programming languages: C, Java, Python
- This talk provides a high-level overview

Client Server Architecture

- Server
  - Always on
  - Responsible for managing the discussion forum
    - Maintains state about the forum
  - Implements a request/response API for interacting with client(s)
  - Prints informative updates at the terminal

- Client
  - Allows a user to interact with the discussion forum
  - Interaction is through the command line terminal

- Transport Protocol
  - Must use TCP

Execution

- Server
  - Command line arguments:
    - Server port (use a value greater than 1023 and less than 65536)
    - Admin password (used for shutting down the forum)
  - Executed first – waits for client(s) to connect

- Client
  - Command line arguments:
    - Server IP address (use “localhost” as client(s) and server will be executing on same machine)
    - Server port number (should match the first argument for the server)
  - Let the OS pick an ephemeral port
  - Client should initiate TCP connection with server (“127.0.0.1”, server port)

Two Configurations

- **One client at a time** (14 marks – 70% of the assignment marks)
  - At any given time only one client will interact with the server
  - Simple sequential interaction between client and server
  - Should not require multi-threading
  - Multiple clients can connect sequentially – i.e., one client connects, executes some commands, quits, second client connects, executes commands, quits, etc.
  - You will need to define various data structures to maintain state of the message board (# of active threads, thread creators, active username, etc.)
    - You must manage several files
      - one for each thread and any specific files uploaded by users
  - You must design your own **application layer protocol**
    - This includes the syntax/semantics of the messages exchanged between the client and server and the actions to be taken upon receiving each message
  - Non-CSE Students may chose to only implement functionality for this configuration
    - **MUST request approval** – check spec for details

Two Configurations

- **Concurrent multiple clients** (6 marks – 30% of the assignment marks)
  - Client program should only require minor changes
  - Server program must interact with multiple clients
    - We recommend a **multi—threaded approach**
    - Main thread should listen for new connections from clients
    - A new thread should be forked to manage interaction with each client
    - The interactions with an individual client should still be sequential as in the first configuration
    - However, the message board state is now updated by multiple clients
      - Client A adds a new message to a thread. Following this, when Client B reads the thread, client A's message should be viewable
    - You may assume that each interaction with a client is "**atomic**"
      - Client A initiates an interaction to add a new message to a thread. While the server is processing this interaction, it cannot be interrupted by a command from another client (B). Client B's command will be acted upon after the command from client A is processed
  - You must be particularly careful about managing multiple threads and how they access the shared state (data structures, files, etc.) for the message board

User Authentication

- A credentials file will be available in the server working directory (example is on the assignment page)
- The client will first authenticate the user
  - Ask user to enter username
  - If username matches with one of the entries in the credentials file, the user is requested to enter the password
    - If password matches, authentication successful
    - If password doesn’t match, the user is asked to enter username again
  - If username does not match it is assumed this is a new user and user is asked to enter a new password
    - User is authenticated and a new entry is created in the credentials file

Discussion Forum Operations

- 11 specific commands
- User is prompted to enter one of the commands
- Basic error checking for syntax, appropriate arguments, etc. should be implemented
  - NOTE: We are more interested in observing if you have implemented the “normal” functionality, so our tests for error checking will be limited and won’t include bizarre edge cases

Commands

- CRT – Create thread
  - `CRT threadtitle`
  - Create a new thread – represented as a text file (threadtitle), update state information about active threads and thread creators
- MSG – Post message
  - `MSG thread message`
  - Add message to the thread – update the thread file (message number, username, message)
- DLT – Delete message
  - `DLT thread messagenumber`
  - A message can only be deleted by the original creator (error if condition not met)
  - Delete message – update the thread file (remove the message and update message numbers if required)

Commands

- EDT – Edit message
  - `EDT threadtitle messagenumber message`
  - A message can only be edited by the original creator (error if condition not met)
  - Edit the thread – update the thread file (only update message)

- LST – List threads
  - `LST`
  - List all active threads
  - Read data structure that stores this information and send it to client

- RDT – Read thread
  - `RDT threadtitle`
  - Display all messages contained in thread
  - Read the corresponding thread file and send relevant information to client

Commands

- UPD – Upload file
  - `UPD threadtitle filename`
  - Create filename in server working directory and store all file data received from client
  - Update thread file with information about file (username uploaded filename)
- DWN – Download file
  - `DWN threadtitle filename`
  - Create filename in client working directory and store all file data received from server
- RMV – Remove thread
  - `RMV threadtitle`
  - A thread can only be removed by the original creator (error if condition not met)
  - Remove all state associated with the thread (thread file, uploaded files)

Commands

- XIT – Exit
  - `XIT`
  - User quits
  - Update state about active users

- SHT - Shutdown
  - `SHT adminpassword`
  - Server shutdowns – all state maintained is removed, all active clients are informed

How to start and progress

- Focus on **first configuration**
- Get the client and server to setup a TCP connection
- Implement authentication
- Implement one command, test it thoroughly, move to the next
- Suggest the following order: XIT, CRT, LST, MSG, RDT, EDT, DLT, UPD, DWN, RMV, SHT
- Once all individual commands are done and tested, then make sure you test extensive interaction with a single client (meaningful combination of commands)
- Next progress to **second configuration – multi-threading**
- **STRONGLY SUGGEST TO DEVELOP YOUR IMPLEMENTATION IN VLAB ENVIRONMENT (NOT ON YOUR NATIVE MACHINE)**
- Added benefit – CSE accounts are backed up

Resources

- Many program snippets are on the web page
  - Including multi-threading code snippets
- Your socket programming experience in Labs 2 and 3 will be useful
- Repository of resources is here

Testing

- Test, Test, Test
- Server and client(s) executing on **same machine but different working directories**
- Emphasis on correct behaviour
- Basic error checking
- **MUST Test In VLAB environment**
- If we cannot run your code, then we cannot award you any marks
- Your assignment will be MANUALLY marked by your tutors
- Client and server must display meaningful messages at the terminal
- Extensive examples are available at the end of the spec
- You are NOT required to use the exact same text

Plagiarism

- **DO NOT DO IT**
- If caught
  - You will receive zero marks (and there may be further repercussions if this is not your first offence)
  - Your name will be added to the school plagiarism register

Seeking Help

- Assignment specific consults (for all 3 programming languages) from Weeks 7-10
  - Schedule to be announced in a few days
- Course message forum
  - Read posts from other students before posting your question
- Read the spec – very often your answer will be in there