KARACHI INSTITUTE OF ECONOMICS & TECHNOLOGY

PAF KIET

Assignment 3 - 102539

Type: Group Assignment (3 or 4 members per team)

Course: Operating Systems

Instructor: Ayaz ul Hassan Khan

Objective

In this assignment; you will learn:

- How to create and use shared memory
- How to create and use semaphores for managing concurrent access of shared resources
- Linux system calls for shared memory and semaphores

Problem Statement

You are required to produce two programs, ***chatserver*** and ***chatclient***. There can be more than one instances of chatclient running at a time and the purpose of ***chatserver*** is to provide a link between all the available clients so that they can talk to each other.

Requirements

chatserver

On startup, server will create a shared memory area which will be used as ***messageboard***. All messaging is done through this ***messageboard***. It means that whenever a client has to send a message to another client; it will place it in the ***messageboard***. There can be one message in ***messageboard*** at a time. All available clients will regularly check this ***messageboard*** and if a client finds some message for it, it picks it and deletes the message from ***messageboard***. Clients also use this ***messageboard*** to send messages to the server. The server if it has a message for a client, who is not online anymore, frees the ***messageboard***. The server will keep a list of online clients with it and clients can ask about this list by sending a message to the server. The server is also responsible for sending the latest client list to all the clients whenever there is a change in it.

chatclient

On startup, each client will send a connect message to the server in which it will send its id. The server will register this client. Each client is required to send an alive message to the server after regular intervals so that the server is updated about its presence. The server will tell to the client about the length of this interval in response to the connect message. If the server doesn't receive an alive message from some client then it de-lists it and sends the latest list to all online clients. Whenever a client wants to send a message, it checks the ***messageboard***; if it is free then the client places its message there. Similarly all clients check the ***messageboard*** after regular intervals to know that if there is a message for them. After picking a message; a client is required to free the ***messageboard***. Clients can also request the latest client list by sending a message to server.

messageboard

It is a shared memory area created by server and used for communication between clients and between a client and a server. The size of messageboard is 256 Bytes.

message
A message sent by a process (client/server) to another. It is a null terminated string with the maximum length of 255 bytes (256^{th} byte is null). First 8 bytes are the name of destination, next 8 bytes are the name of the source and the remaining bytes up-to the null byte are the actual message. It means that the actual message length can't be more than 239 bytes. It also implies that a client name can't be more than 8 bytes, and if some client has name less than 8 bytes then you have to perform padding to keep it of 8 Bytes. Moreover, as the server sends the list of online clients through a message and the message length is limited, therefore, either you have to limit the number of concurrent clients or use some other way to send the list in more than one iterations.

Standard/Control Message Formats

The following standard/control messages should be implemented:

Connect
Syntax: Connect clientid
Purpose: automatically sent by a client to the server when the client comes online
@Quit
Syntax: @Quit
Purpose: user types it at the client prompt to end his session
Quit
Syntax: Quit clientid
Purpose: automatically sent by a client to the server when a user requests for session end
@List
Syntax: @List
Purpose: user types it at the client prompt to view the current list of online clients
List
Syntax: List
Purpose: automatically sent by a client to the server when a user requests the list of online clients
Alive
Syntax: Alive clientid
Purpose: automatically sent by client to server after regular intervals that it is still alive
General Message to some other client
Syntax: (otherclientid) message-statement
Purpose: typed by the user at the client prompt when he want to send a message to an online client

Note: In all messages to the server, the destination address is "-SERVER-". You can implement server in a way that it can broadcast a single message to all clients by typing destination as "ALL" or some thing else, but it is not required. All concurrent accesses to ***messageboard*** are managed by semaphores.

Submission:

Deadline: December 05, 2019 till midnight.

Required Deliverables:

Two source code files and a report should include the usage of each source code and test cases (covering all the defined requirements) with brief explanation and screen shots of sample executions.
Note: You should provide proper comments in source code to get full marks.
Submission should be done on GitHub Assignment 3 through the following link:
https://classroom.github.com/g/wbOA4xuM

Contribution of each member should be highlighted properly on GitHub.

Grading Scheme:

Server (100)

Creation of Shared Memory (Message Board): 5
Creation of Semaphore for Message Board: 5
Deletion of Shared Memory (Message Board): 5
Deletion of Semaphore for Message Board: 5
Reading Message from Message Board (includes tokenization of message): 10
Writing Message to Message Board (includes creation of message in specified format): 10
Creating Client List: 5
Sending Client List: 5
Adding new Client to List: 5
Deleting Client from List: 5
Retiring Client: 10
Informing about change in List: 5
Freeing Message Board if message for offline Client: 10
Freeing Message Board if message from offline Client: 10
Resetting Client life on receiving alive message: 5

Client (100)

Creation of Shared Memory (Message Board): 5
Creation of Semaphore for Message Board: 5
Deletion of Shared Memory (Message Board): 5
Deletion of Semaphore for Message Board: 5
Taking Message from user: 20
Reading Message from Message Board (includes tokenization of message): 10
Writing Message from Message Board (includes creation of message in specified format): 10
Displaying Client List: 10
Sending alive message: 10
Sending connect message: 10
Sending quit message: 10