==========

Proyecto I Redes de Computadoras: Sala de Chat en linea

Jose Montenegro 10-10469

Gamar Azuaje 10-10051

==========

NOTA: Para comoilar se usaron los flags -pthread y -std=gnu99

+SERVIDOR (SCHAT)

-Corrida: El programa servidor schat es invocado por el terminal con la linea ./schat -p <puerto> -s <sala>, donde <puerto> es el puerto por el que la aplicacion escuchara comandos de los clientes y sala es el nombre de la sala por defecto. El puerto es obligatorio, mientras que si falta la sala, la sala por defecto se llamara actual

-Inicializacion: Al iniciar el programa con los parametros correctos, la aplicacion primero inicializara las variables que necesitara a lo largo de todo el programa, a decir:

-sockfd: El file descriptor del socket por el cual se aceptaran conexiones

-newsockfd: El file descriptor generado cada vez que se acepta una conección. Su valor cambia cada vez que se conecta un nuevo cliente, ya que dicho valor se asocia a uno de los campos que representa la estructura cliente

-portno: Numero de puerto en el cual la aplicacion escucha

-clilen: Largo de la informacion suministrada por el cliente en conexion

-serv_addr, cli_addr: Direcciones tanto del cliente como del servidor. La del cliente cambia en cada conexión

-conns: Arreglo de hilos que manejan los requests de los clientes. Hay un hilo por cliente

-salaDefecto: Nombre de la sala por defecto. Inmediatamente se inicializa a "Actual". Mas tarde en corrida se decide si este nombre cambia o no

-mg: Estructura manager que se encarga de manejar los clientes y las salas que el servidor posee en todo momento

-Chequeo de Argumentos: Primero se chequea si el numero de argumentos dados es correcto. En caso de que no, se da un mensaje de error y se termina la corrida. Luego se verifica cada flag para ver si se dio la informacion correcta para cada uno o si se usaron los flags correctos.
Es aqui cuando el programa cambia el valor de salaDefecto si es necesario y le da un valor a portno

-Conexion: Se inicia la fase de conexion. Primero se crea un nuevo socket y se le asigna el file descriptor a sockfd. Para el protocolo de red se decidio usar TCP, pues garantiza que el mensaje llegue completo y en el orden correcto, caracteristica esencial en un programa de chateo. Acto seguido se hace bind al puerto especificado y a todas las interfaces de red de la maquina que lo corre. Luego se crea la sala por defecto antes de escuchar conexiones. Se hace la prevencion de agregar un salto de linea al nombre de la sala, pues asi es como estaran colocados todos los comandos de los clientes. Por ultimo se comienzan a escuchar conexiones en el socket.

-Ciclo de Creacion de Hilos: Cuando un cliente se conecta al servidor, se genera un nuevo

file descriptor asociado a ese cliente. En este momento el servidor lee en ese fd para obtener

el nombre del cliente y le manda un mensaje de confirmacion para iniciar el servicio. Es entonces

suscrito a la sala por defecto. Acto seguido se crea una nueva variable de tipo clientData que contendra

el nombre y fd del cliente. Esta informacion se le suministrara a un nuevo hilo para que este se

encarge de procesar todas las ordenes provenientes del cliente. Entonces se repite el ciclo hasta que

el usuario le mande una señal de SIGINT al server.

-Proceso de comandos de clientes: Cada hilo se mantiene leyendo en el fd de su cliente. Cuando recibe

un comando, este esta garantizado de ser correcto en sintaxis (el cliente hace esta verificacion). El

hilo ejecuta una accion diferente dependiendo del comando que se pidio

-fue: Elimina al cliente del manager (Desuscribiendolo en el proceso). El cliente se desconecta por

su cuenta. El server manda un mensaje de alerta al resto de los clientes y termina la ejecucion del

hilo

-sal: Genera el texto que menciona las salas abiertas y lo manda a su fd asignado. Esto lo hace la

funcion enviarSalas

-men: Genera el mensaje que se quiere enviar y le coloca el header con el nombre del usuario.

Entonces se llama a writeToAllCLientRooms, funcion que busca todos los clientes suscritos a todas

las salas a las que esta suscrito el cliente de este hilo y les manda el mensaje a su fd. La funcion se

encarga de agregar un header con la sala de la cual proviene el mensaje.

-sus: Suscribe al cliente a la sala usando una funcion de commandManager.h y le manda al cliente un

mensaje de confirmacion

-eli: Elimina la sala y desuscribe a todos los clientes de la misma. Esto lo hace una funcion de

commandManager.h. Se manda un mensaje de confirmacion a todos los clientes para alertalos de que

la sala ya no existe

-usu: Genera el texto con los nombres de los usuarios conectados y se la manda al cliente

-des: Desuscribe al usuario de todas las salas por medio del uso de una funcion de

commandManager.h

y manda un mensaje de alerta a todos los clientes

-cre: Crea una nueva sala con commandManager.h y alerta a todos los clientes de la creacion

-Terminacion del programa: Por los momentos, solo se puede detener al server matando su proceso. Se desea implementar un cambio a la señal SIGINT para que el server cierre el socket antes de salir.

+CLIENTE (CCHAT)

-Corrida: El programa cliente cchat es invocado por el terminal con la linea

./cchat -h <host> -p <puerto> -n <nombre> -a <filename>, donde <host> es el hostname o direccion ip del

server, <puerto> es el puerto a usar en la conexion, nombre es el nombre de usuario y filename es un

archivo que contiene un comando por linea. Solo -a es opcional.

-Inicializacion: Estas son las variables que se crean al inicio:

-sockfd, portno: File descriptor del socket de comunicacion con el servidor y el numero de puerto

-serv_addr: Direccion del server

-server_address: Direccion del server en formato de cadena de caracteres. Se usa para generar la verdadera direccion a partir del input del usuario

-client_name: Nombre del cliente

-file_name: Nombre del archivo de comandos. Se inicializa en NULL para que, si no es cambiado, no ocurra nada al tratar de correr comandos por archivo

-server: Estructura hostent que contiene toda la informacion necesaria para establecer conexion con el server

-buffer: Buffer usado para mandar y recibir mensajes hacia y desde el server

-lectura: hilo que se encargara de leer todos los mensajes del server

-datosLectura: Estructura por la cual se enviaran los datos necesarios al hilo de lectura

-Chequeo de Argumentos: Igual que en schat, se chequea primero si el numero de argumentos es el correcto para luego hacer la verificacion de los flags. En cualquier caso de error de imprime un mensaje de uso y se suspende la corrida

-Conexion: Se abre un nuevo socket con TCP y se obtiene la direccion del server usando la funcion auxiliar obtenerDireccion, que distingue enter ip y hostname. Finalmente se hace la llamada a connect. En caso de exito, se escribe el nombre del cliente al servidor y se lee en busqueda de un mensaje de confirmacion. Justo despues, se cambian las señales SIGINT y SIGTSTP, para evitar que el usuario mate un proceso cliente.

-Ciclo de Comandos: Primero se llama a executeFromFile, funcion que ejecuta los comandos dados

en el archivo. No hace nada si el archivo es NULL. Luego se crea el hilo de lectura con la funcion leerRespuestas. Este hilo simplemente leera todo lo proveniente del servidor y lo imprimira en pantalla. Asi se logra imprimir toda respuesta del server en tiempo real, a costa de a veces fragmentar algun comando que el cliente este escribiendo al momento de recibir la respuesta. Fuera del hilo, el cliente entra en un ciclo de escritura constante, hasta que el cliente escriba fue, lo que cierra la conexion y termina la corrida

+DIAGRAMA DE SECUENCIA