Computación 2

Trabajo Final - 2018

"IM"

Enunciado

Desarrollar un sistema cliente-servidor para administración de mensajería instantánea (IM).

Funcionamiento general:

- El servidor hará las veces de gateway de comunicación entre los diferentes clientes, de modo que los clientes puedan "**chatear**" entre si por medio del servidor.
- Los clientes van a mantener un **status** de operación: **Idle** (desocupado) y **busy** (ocupado)
- Cuando un cliente se conecta al servidor, el server le deberá pedir un **nick** de usuario que luego le servirá a los demás clientes para ubicarlo.
- Al conectarse, el cliente automáticamente pasa a tener status "**idle**".
- Cuando el cliente se conecta dispondrá de tres comandos para ejecutar desde su entrada estándar:
  - **list**: le permitirá listar a los clientes conectados y con estado "idle"
  - **chat <nick_destino>**: le permitirá iniciar una sesión de chat contra el cliente destino identificado por su nick.
  - **bye**: le permitirá a cualquiera de los dos clientes cerrar su sesión, con lo que el otro cliente pasará a status idle nuevamente, y podrá iniciar o recibir otra conexión de chat.
- Una vez que el cliente se conecta y ejecuta "chat" contra otro cliente, ambos pasarán a status "**busy**", y todo lo que uno escriba en su terminal deberá recibirlo el otro en la suya. *Aquí se acepta comunicación por turnos (primero uno, luego el otro) o instantánea (cualquiera que escriba en cualquier momento, el otro recibirá el mensaje), eso depende de cómo se implemente, y la complejidad que quiera darse al algoritmo.*
- El servidor deberá almacenar en un archivo LOG general la información de conexiones y chat de los clientes, especificando la fecha y hora del mensaje. Dependiendo de cómo se implemente (multiprocesos, multihilos, quién escriba en el archivo, etc) deberá garantizarse acceso único al mismo, por ejemplo, mediante el uso de semáforos POSIX. El contenido de este archivo será algo similar a esto:

```
20171114 – 18.56 | Client connect: PID/TID client: 123123 | Nick: d1cor
20171114 – 19.15 | Client chat: PID/TID client: 123123 | Nick: d1cor – Nick
Destination: juancho
20171114 – 19.25 | Client bye: PID/TID client: 123123 | Nick: d1cor
```

- El servidor deberá almacenar en archivos de texto identificados e individuales, los mensajes de cada conexión, fecha y hora. Algo así:

```
20171114 – 19.15 - d1cor: Hola juancho!
20171114 – 19.16 - juancho: Hola d1cor! Todo bien?
```

Especificaciones:

- La comunicación entre clientes y servidor será mediante **sockets INET STREAM** en un puerto servidor elegido por el alumno (documentado debidamente).
- Para llevar a cabo las conexiones, el servidor de alguna manera deberá mantener las asociaciones entre los nick's de los clientes, y sus sockets de conexión.
- El software deberá estar escrito en módulos, con sus correspondientes archivos de cabecera, y **makefile** para poder compilar cliente y servidor.
- Deberá hacer un uso apropiado para los mecanismos de comunicación entre procesos solicitados.
- Podrá **ampliar la consigna** y adaptarla a las necesidades de la línea de desarrollo, siempre y cuando se cumplan los objetivos antes planteados, y se utilicen las herramientas vistas.
- El código, debidamente separado en **directorios** (include, obj, src, lib, etc), deberá contar con las correspondientes instrucciones de instalación en un archivo de texto INSTALL y la ayuda básica en un archivo README.
- *Opcionalmente, podrá documentarse el código mediante **Doxygen** o herramientas similares.*
- El código deberá subirse a un repositorio **GitHub** y entregar la **URL** del mismo **una semana antes** de la mesa de examen, para poder evaluar el trabajo.