Práctica 1 – Aplicación Cliente-Servidor usando Sockets

Objetivos

El alumno:

- Analiza los servicios definidos en la capa de transporte
- Emplea el modelo Cliente-Servidor para construir aplicaciones en red
- Programa aplicaciones Cliente-Servidor utilizando sockets de flujo o de mensaje

Desarrollo

Implementar uno de los siguientes juegos, usando el modelo Cliente-Servidor y Sockets:

1. Buscaminas
2. Gato Dummy
3. Ahorcado

El alumno debe diseñar un algoritmo para calcular el número de días que ha vivido hasta del día 15 de agosto del 2019. Con número de días módulo 3 se puede saber si le toca el juego de buscaminas (0), gato (1) o ahorcado (2).

El alumno debe elegir si va a usar sockets de flujo (TCP) o sockets de mensajes (UDP) para su implementación. Justificar su elección.

No es necesario que el juego tenga una interfaz gráfica. Sin embargo, el cliente y el servidor tendrán un tablero usando la consola.

Fecha de entrega 26 de agosto. Hora de entrega: 8:30 Buscaminas. 9:00 Gato. 9:30 Ahorcado. Lugar: Laboratorio de redes.

Buscaminas.

Deberá implementarse un cliente con las siguientes características:

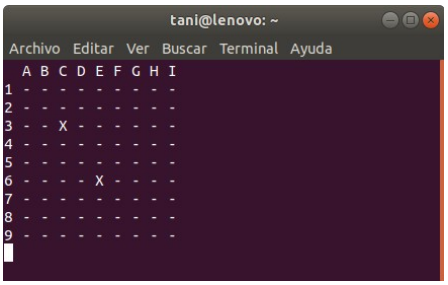
- La aplicación cliente deberá conectarse con la aplicación servidor a través de un socket (la dirección y puerto destino deberán ser proporcionados por el usuario).
- El cliente deberá recibir desde el servidor el tablero que contendrá una matriz cuyo tamaño será dependiente de la dificultad de juego escogida por el usuario. El número de minas será también dependiente de la dificultad elegida y éstas serán acomodadas de manera aleatoria en las casillas del tablero.
    - Nivel principiante: tablero de 9 × 9 casillas y 10 minas.
    - Nivel avanzado: tablero de 16 × 16 casillas y 40 minas.
- Al iniciar la partida deberá tomarse una marca de tiempo del lado del servidor tanto al inicio como al finalizar la partida de modo que pueda registrarse la duración de la misma.
- El jugador tendrá la capacidad de descubrir o marcar una casilla teniendo en cuenta que solo podrá marcar un máximo de N casillas; donde N = número de minas en el tablero de acuerdo a la dificultad elegida.
- En el caso que se descubra una casilla que contiene una mina, la aplicación deberá descubrir todas las minas del tablero y notificar al usuario que ha perdido.

- En el caso que se descubran todas las casillas del tablero, menos las que contienen minas, la aplicación deberá notificar al usuario que ha ganado la partida, asimismo deberá tomar una marca de tiempo y mostrar el tiempo realizado por el jugador.

Deberá implementarse un servidor con las siguientes características:

- El servidor solicitará al usuario especificar el puerto de servicio en el que aceptará al jugador mediante sockets.
- Una vez iniciado el servidor, este recibirá la conexión del jugador.
- En cuanto se conecte un cliente, el servidor recibirá por parte de éste el nivel de dificultad que se desea jugar y en base a esta dificultad se generará un tablero que contendrá el número de minas correspondientes a la dificultad y colocadas de forma aleatoria dentro del tablero. Una vez enviado el tablero, el servidor deberá prepararse para recibir y validar cada una de las acciones del jugador (destapar, marcar), así como enviar la actualización correspondiente del tablero al otro jugador.
- Al finalizar la partida, (es decir, cuando todas las minas han sido marcadas, cuando todas las casillas han sido destapadas, excepto las que contienen minas, o cuando se ha destapado una casilla que contiene mina), el servidor deberá informar al jugador si ganó o perdió la partida, así como mostrar un registro del tiempo que duró la partida.

Ejempo tablero



Gato Dummy

Deberá implementarse un cliente con las siguientes características:

- La aplicación cliente deberá conectarse con la aplicación servidor a través de un socket (la dirección y puerto destino deberán ser proporcionados por el usuario).
- El cliente deberá recibir desde el servidor el tablero que contendrá una matriz cuyo tamaño será dependiente de la dificultad de juego escogida por el usuario.
    - Nivel principiante, tres en línea: m=3, n=3, k=3.
    - Nivel avanzado, cinco en línea: m=10, n=10, k=5.

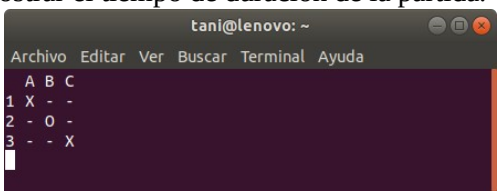
  En ambos casos, se construye un tablero de m x n y el objetivo es conseguir k en línea (horizontal o vertical).
- Al iniciar la partida deberá tomarse una marca de tiempo del lado del servidor tanto al inicio como al finalizar la partida de modo que pueda registrarse la duración de la misma.
- El jugador tendrá la capacidad de elegir una casilla donde colocará su marca. Una vez elegida la casilla, esperará que el servidor haga su elección. Cada aplicación debe esperar su turno para poder elegir casilla.
- El juego termina cuando todas las casillas están ocupadas (empate), o cuando algún jugador logre conseguir K casillas en línea.
- El servidor debe mostrar el resultado de la partida, es decir, quién fue el ganador o si fue empate. También debe mostar el tiempo que duró la partida.

Deberá implementarse un servidor con las siguientes características:

- El servidor solicitará al usuario especificar el puerto de servicio en el que aceptará al jugador mediante sockets.
- Una vez iniciado el servidor, este recibirá la conexión del jugador.

- En cuanto se conecte un cliente, el servidor recibirá por parte de éste el nivel de dificultad que se desea jugar y en base a esta dificultad se generará un tablero que contendrá el número de casillas correspondente. El servidor debe enviar el tablero al cliente. Una vez enviado el tablero, el servidor deberá prepararse para recibir y validar cada una de las acciones del jugador (elegir casilla), así como enviar la actualización correspondiente del tablero al otro jugador.
- El servidor deberá elegir una casilla de manera aleatoria. Deberá marcarla en el tablero y mandar la actualización del tablero al cliente.
- El servidor deberá validar si algún jugador logra marcar k casillas en línea. Si es el caso, el servidor finaliza la partida.
- Al finalizar la partida, (empate o k en línea), el servidor deberá informar al jugador si ganó, perdió o empató la partida, así como mostrar el tiempo de duración de la partida.

Ejemplo tablero gato



AHORCADO

Deberá implementarse un cliente con las siguientes características:

- La aplicación cliente deberá conectarse con la aplicación servidor a través de un socket (la dirección y puerto destino deberán ser proporcionados por el usuario).
- El cliente deberá recibir desde el servidor el tablero que contendrá un arreglo cuyo tamaño será dependiente de la dificultad de juego escogida por el usuario.
    - Nivel principiante: arreglo de máximo 10 letras.
    - Nivel avanzado: arreglo de máximo 20 letras.
- Al iniciar la partida deberá tomarse una marca de tiempo del lado del servidor tanto al inicio como al finalizar la partida de modo que pueda registrarse la duración de la misma.
- El jugador tendrá la capacidad de mandar una letra para saber si forma parte de la palabra. Si es así, el servidor debe mostrar todas las letras que existen en la palabra, en caso contrario, el servidor deberá contabilizar el error y mostrarlo en el tablero.
- El jugador tendrá hasta cinco oportunidades para equivocarse, en caso de que la letra que eligió no exista en la palabra.
- En el caso que se descubran todas las letras que componen la palabra, la aplicación deberá notificar al usuario que ha ganado la partida, asimismo deberá tomar una marca de tiempo y mostrar el tiempo realizado por el jugador.

Deberá implementarse un servidor con las siguientes características:

- El servidor solicitará al usuario especificar el puerto de servicio en el que aceptará al jugador mediante sockets.
- Una vez iniciado el servidor, este recibirá la conexión del jugador.
- En cuanto se conecte un cliente, el servidor recibirá por parte de éste el nivel de dificultad que se desea jugar y en base a esta dificultad elegirá de manera aleatoria una palabra de su repositorio. El servidor debe tener un repositorio de 20 palabras fáciles y 20 palabras difíciles.
- Una vez enviado el tablero, el servidor deberá prepararse para recibir y validar cada una de las acciones del jugador (adivinar letra), así como enviar la actualización correspondiente del tablero al otro jugador.
- Cuando el servidor recibe una letra, debe verificar si existe dentro de la palabra correspondiente. Si existe, deberá mostrar todas las letras dentro de la palabra. Si no existe deberá incrementar la variable de número de errores. En ambos casos debe actualizar el tablero y mandar la actualización al cliente.
- La partida termina cuando el cliente adivina la palabra o termina con el número de errores (6).

- Al finalizar la partida, el servidor deberá informar al jugador si ganó o perdió la partida, así como mostrar un registro del tiempo que duró la partida.

Ejemplo tablero ahorcado