UNIVERSIDAD CATÓLICA ANDRÉS BELLO
EXTENSIÓN GUAYANA, NÚCLEO PUERTO ORDAZ
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA INFORMÁTICA
CÁTEDRA DE SISTEMAS DISTRIBUIDOS

Proyecto I: Battleship

Docente:

Ing. Saab Antonio

Integrantes:

Beckles Xavier. C.I. 24964060

Godoy Anthony. C.I. 23638966

Lara Luis. C.I. 25083570

Puerto Ordaz, Estado Bolívar

Resumen técnico

Problema Asignado: Elaboración de un Battleship, multijugador, el cual sea capaz de comunicarse entre dos PC’s para así jugar mutuamente.

Lenguaje de programación utilizado: C.

Factores claves: Socket, reglas del Battleship.

Sistema en que correrá: Linux.

Entorno gráfico: Parcial, funciona en consola con manipulación por teclado.

Pruebas realizadas: Si.

Cantidad de barcos editables: Si.

Tamaño del tablero editable: Si.

Objetivos

Objetivo general

Recrear el famoso juego Battleship utilizando las redes y la tecnología de Sockets para la comunicación entre dos PCs. La arquitectura a implementar consta de un servidor y dos clientes, en la que el servidor será el encargado de gestionar el juego en todo momento mientras que los clientes serán los encargados de interactuar con los jugadores.

Objetivos específicos

- Recrear el juego de Blattleship bajo el lenguaje de programación C o C++.
- Comunicar dos computadores mediante Sockets para jueguen entre sí.

El juego

El Battleship es un juego popular, de dos personas, que consiste en posicionar una serie de diferentes barcos en un tablero estratégico, para así, tras dicha fase de colocación, proceder a disparar por coordenadas para intentar hundir la flota enemiga. Es un juego con reglas claras y simples, el cual limita las posibilidades de variación, permitiendo así un desarrollo estructural muy procedimental. Las reglas son las siguientes:

- Dos jugadores por partidas.
- Dos tableros por persona (propio y de disparo).
- Diez fichas por juego (Dos portaaviones, tres buques, y cinco lanchas).
- Un movimiento por turno.
- Un solo ganador.

Pensando en un problema futuro, el proyecto fue diseñado para originalmente dos tipos de usuarios, el primero, usuario-Cliente, el cual es un usuario que se conecta para jugar, y un segundo usuario apodado usuario-Cliente-Servidor, el cual no solo se dedica a jugar, sino que también sirve el papel de servidor, el cual, funciona y trabaja de manera invisible para ambos jugadores.

Para el tablero se diseñó una matriz con valores máximos definidos, la cual, será la zona de configuración. Como el usuario necesita tener dos tableros, uno donde visualiza sus fichas, y otro en donde disparará, se ideó un tablero interactivo que permite la simulación de un movimiento gracias al teclado. Para dicho movimiento, el usuario se somete a una constante lectura de letra y limpieza de pantalla, para luego, reimprimir dicha matriz de movimiento a tiempo real.

En el caso de los barcos, se idearon “espacios lógicos”, los cuales son casillas rellenas por un indicador de tipo de barco para así, poder ser dibujados en la matriz. Como se tienen tres tipos diferentes de barcos, han de existir estructuras que delimiten tanto el tamaño, como la cantidad barcos de ese mismo tipo que pueden estar en el tablero. Básicamente se rellenarán las casillas de la matriz con “banderas” o indicadores que delimitarán el tipo de barco, y donde comienza y termina el mismo.

Para el caso de los movimientos, al ser un juego por turnos, el servidor se encargará de mediar los turnos de los usuarios, iniciando por él mismo (el usuario que ejecuta el servidor) como el primer jugador en jugar. El juego se divide en dos etapas, la etapa de colocación, y la etapa de disparos. En la etapa de colocación, se pedirá al usuario-Servidor que coloque su flota, para así, luego, pasar el turno al usuario-Cliente y que éste haga lo mismo; en el caso de la fase de disparos, la cual se ejecutará con la inmediatez en que el cliente haya colocado sus barcos, bien que pasarán a una serie de turnos intercalados, en donde tras el disparo de un usuario, vendría el otro a atacar.

Para el caso del ganador, se comprueba constantemente si un usuario posee o no, toda su flota hundida. Si ese es el caso, se avisa para así indicar quien ha ganado. No se permiten

dos ganadores, por ende, la comprobación se realiza cada vez que un usuario haya disparado.

La conexión

La conexión para este proyecto se basa, inmediatamente (y por asignación), en Sockets. Se utilizan Sockets de modo que existen, como antes fue mencionado, dos tipos diferentes de usuarios. El primer usuario: Servidor, el cual es el usuario que levanta la conexión gracias a una apertura de puertos, dedicándose también a ejecutar el juego en una constante series de preguntas y espera que permitirá, desde un inicio, la interacción entre los dos jugadores; el segundo usuario: Cliente, el cual se conecta mediante una dirección de servidor, y un puerto, para así, formar parte de dicha conexión e interactuar con su contraparte.

Se usa el protocolo TCP porque es necesario tener un acuse de recibo que me garantice que la conexión llegó, y por ende, se efectuó. Esta medida es elaborada para así comprobar y garantizar que es la información, pues si digamos, en el caso del intercambio de mapas (en este proyecto), dicho intercambio falla, el juego se corrompe, y por ende, falla en su ejecución.

Como los jugadores están conectados en la misma red, la conexión se basará en la ip de la máquina Servidor, y con ello, la misma trabajará como cliente. Pasando a ejemplificar ésta teoría, se tiene dos máquinas, una cliente y una servidor, en donde se espera el intercambio de cualquier tipo de flujo de datos. El usuario-Servidor es a su vez un usuario-Cliente, por ende, los procedimientos del servidor le han de ser invisibles para así evitar problemas de conocimientos de datos. El usuario-Servidor inicia un proceso a tiempo real, el cual, ejecuta a su propio cliente interno, para así, comenzar a jugar, sirviendo luego como mediador de datos, accionándose y deteniéndose para permitir el pase de información en los turnos.

En el caso del inicio del juego, el usuario-Servidor realiza la apertura de un puerto en común, esperando entonces una conexión entrante de parte de un cliente, el cual "notifica" que desea acceder y se le concede entonces el permiso. Basado en la cantidad máxima de conexiones que se le permita al servidor (en este caso una sola), el servidor pasaría a bloquear el acceso a dicho puerto, limitando así conexiones nuevas que pudieran interrumpir el proceso del juego. El servidor ejecuta un comando "listen" para estar a espera de la conexión, una vez recibe respuesta de una conexión, lanza la ejecución del programa.

En el desarrollo el juego, el cliente lanza un "read" que lo bloquea a la espera de un "write" del servidor que le indicará que el turno del enemigo ha finalizado. Desde la fase de colocación de barcos, hasta la fase de disparos, ambas trabajan de la misma manera, un cliente se bloquea a espera del turno, mientras que el otro juega. Al realizar una acción, ya sea la colocación de los barcos, o el disparo, el Servidor recibe y envía información de texto que es comparada para visualizar las jugadas. En el caso del ordenamiento de la flota, se pasa al servidor ambos tableros, el propio y el del cliente, de modo que sea éste quien comprueba los disparos que se vayan a realizar. En el caso de los disparos, se le pasa la posición del disparo (i, j), y el mapa adonde se disparará. Se pasa un buffer que se comprueba la acción, y tras el turno, se valida con el mapa original. El cliente es el único que pasa las coordenadas, el servidor no las pasa porque él ya las conoce, al ser quien se ejecuta.