Batalla naval en lenguaje de programación C

Kener, Sebastian – Universidad Nacional de Moreno – Informatica 1 – comision A3

Docentes: Ing. Pascual, Gustavo – Ing. Bassi, Matias

***Resumen*—En base a los contenidos vistos en clase se realizó una: "BATALLA NAVAL". En ella se representó los dos tableros clásicos del juego, radares enemigo y aliado, donde se coloca nuestra flota de barcos (tres) y pasando coordenadas a través de una computadora se logra descubrir la localización de la flota enemiga y hundir todos los acorazados de la otra PC. Para lograr este programa se usaron funciones, tipos de datos, manejo de strings y estructuras; complementados con información adicional como la librería Socket Connection, proporcionados por los docentes para hacer una conexión TCP.**

***Abstract*--Based on the contents seen in class, we made an "BATALLA NAVAL". In it the two classic game boards are represented; those are enemy's and ally's radar, where our fleet of ships (three) is placed. Passing coordinates through a computer is able to discover the location of the enemy fleet and sink all battleships of the other PC. To achieve this program functions, data types, string handling and structures were used; supplemented with additional information such as Socket Connection library, provided by teachers to make a TCP connection.**

I. Objetivo

El objetivo del trabajo es aprobar la cursada cumpliendo con las exigencias de la materia. Para esto se puso en practica y se experimento con los contenidos aprendidos a lo largo del cuatrimestre.

II. Introduccion

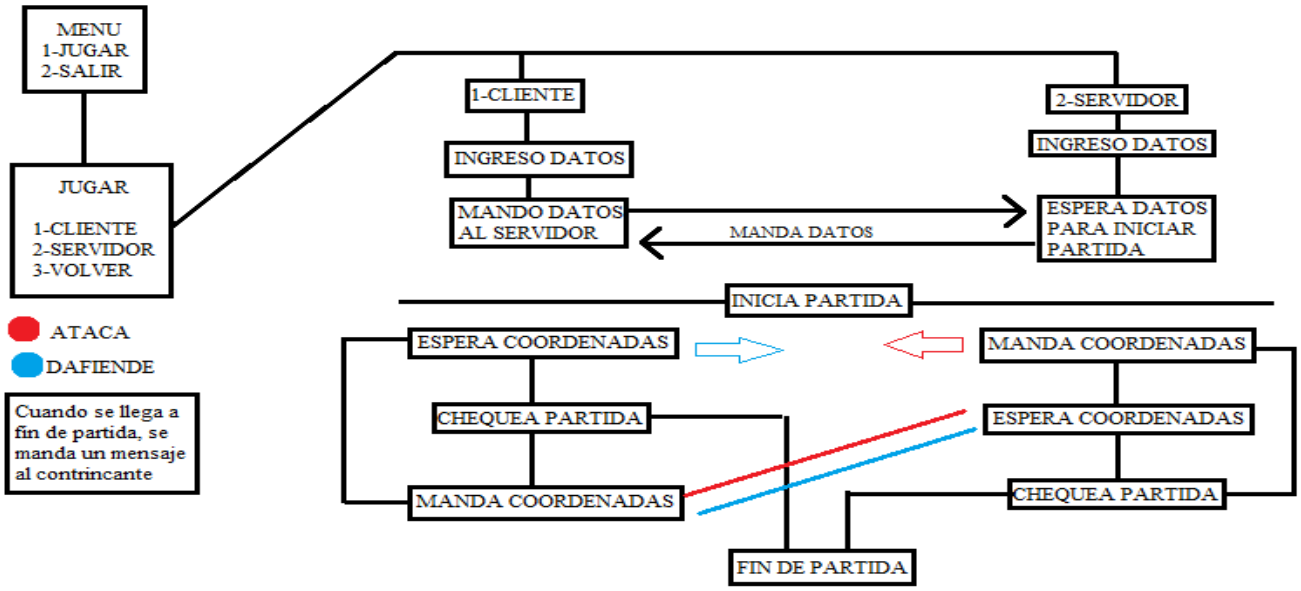
En esta Batalla naval dentro de la función principal, el "main", se hizo un lazo do-while donde se presenta el titulo con el primer menú, y por medio de una switch-case nos permite salir del juego o iniciar. Dentro del menú JUGAR se presentan opciones, seguido nuevamente de otro switch, en esta ocasión elegiremos entre ser servidor, cliente o volver al menú anterior. Tanto como servidor o como cliente se deben cargar datos que permitan la conexión. Una vez presentado el tablero y los barcos en su posición, el programa entra en un lazo for donde se ataca y se reciben disparos, este terminara cuando la flota de alguno sea destruida.

III. Contexto

Uno de los requerimientos para aprobar la cursada es desarrollar un programa que tenga un cierto grado de complejidad, de manera tal que se puedan implementar y ver con mas profundidad los temas dictados.
Para esto los docentes propusieron una varios proyectos. A pesar de haber algunos realmente interesantes, como un programa para reconocer las resistencias por imagen u otro sobre administración de cuentas bancarias, los juegos siempre son un punto de inflexión cuando se toma una decisión. Si bien había opciones en este aspecto, la idea de una batalla naval generaba un desafío distinto.

IV. Medios

El lenguaje de programación C es la principal herramienta para este trabajo. Se caracteriza por ser el mas global de todos, ya que sirve como base para cualquier otro tipo de lenguaje de programación. Permite el uso y manejo de los tipos de datos a través de funciones, y da la posibilidad de crear funciones propias en base a las situaciones que se enfrentan. Así como



también la definición de tipos de datos como estructuras y uniones, que contienen datos de tipo estándar (char, short, long, etc).

Para manejar este tipo de lenguaje es necesario una IDE (Integrated Development Environment o entorno de desarrollo integrado) o un, comúnmente llamado, editor. Esto es básicamente una aplicación que proporciona un manejo mas ágil en cuanto a desarrollo de un software. Para este trabajo se eligió CodeBlocks, un IDE libre, gratuito y muy fácil de usar. Junto con el Qt creator es uno de los mas recomendados por el cuerpo docente.

Para la conexión se utilizaron las librerías "SocketConnection.h" y "SocketTrama.h" proporcionadas por el Ing. Matías Bassi. La primera nos permite la conexión entre computadoras y la segunda nos permite darle el contexto a la información que queremos mandar.

V. Descripción de las funciones utilizadas

Se hará una descripción mas detallada de las funciones que componen el programa agrupándolas por tipo, de manera que sea mas sencilla su lectura. Del mismo modo se describirá los tipos de datos definidos por el usuario.

A- Datos definidos por el usuario

```
struct datos
{
  char nombre[30];
  long id_jugador;
  char port[20];
  char ip[25];
};
struct datos cliente;
struct datos servidor;
```

Datos: Esta estructura se creo para facilitar el guardado y traspaso de información al momento de iniciar la partida. Contiene nombre del jugador, id, puerto y la ip. También se creo las variables a continuación, haciéndolas generales fuera del int *main()* y facilitando que cualquier función las pueda usar.

```
enum contenido
{
      vacio = 32,
     barco1 = 63,
     barco2 = 168,
     barco3 = 64,
     tocado = 88,
     agua   = 79,
yaingresado = 81,
    hundido = 45,
};
```

Contenido: En esta enumeración se le asigna el código ASCII a las variables, permitiendo la ubicación de los caracteres en el tablero y el traspaso de información a la hora de responder cuando se envían coordenadas.

```
struct coord
{
    char x;
    char y;
};
```

Struct coord: Guarda los caracteres ingresados como coordenadas para luego usarlo en el tablero. La 'x' guardaría la posición numérica y la 'y' la letra(Ej: F5).

Enum trama: Esta enumeración se usa con el objetivo de asignarle un codigo a la trama al momento de mandar información. Así cuando se recibe, las funciones entienden lo que llego.

B- Funciones

Función: `void posicion (int x, int y);`
Básicamente es un reemplazo a la función gotoxy. Permite posicionar el cursor en cualquier punto de la pantalla ingresando dos valores uno para 'x' e 'y'. Esta necesita la librería <windows.h> y toma el control de la pantalla.

Función: `void funcion_titulo();`
Dibuja el titulo en pantalla BATALLA NAVAL. Es usada dentro del main.

Funciones:
```
void CD_Cliente(struct datos,
  struct datos);
void CD_servidor(struct datos,
struct datos);
```
Desplega un menu donde pide los datos nesesarios de ambos jugadores para iniciar la conexión y empezar el juego.

Funcion: `void FX_mandarinfoCliente();`
Dentro de esta es donde se produce la primer conexion. El cliente le manda los datos al servidor indicandole que se inicia una partida. Para esto acomoda/transforma la informacion para que la funcion socket_sendReceive, que es la que permite conectarse, pueda utilizarla

Funciones:
```
int serializarDatosCli(char
*dataSerializada, struct datos
cliente);
int serializarDatosServ(char
*dataSerializada, struct datos
servidor);
```
Permiten pasar los datos(estructuras) a strings con el objetivo de enviarlo.

Funcion:
```
struct datos desserializarDatos
(char *dataSerializada );
```
Reconvierte la respuesta, en este caso los datos del servidor,

a estructura para poder utilizarlo en el juego.

Funcion: `void campodebatalla(unsigned short matriz1[][maxcol], unsigned short matriz2[][maxcol]);`

Esta agrupa otras tres con el objetivo de hacer el llamado mas facil.

```
void tablero (void);
void tableromatriz1(unsigned short matriz1[][maxcol]);
void tableromatriz2(unsigned short matriz2[][maxcol]);
```

La primera dibuja en pantalla las letras del tablero, los numeros, un titulo y titulo de tablero.

Las otras dos el tablero propiamente dicho, matriz1 donde se ponen los barcos y se reciben los disparos y matriz2 donde se marcan las respuestas luego de mandar las coordenadas.

Funcion: `struct coord coordenadas();`

Pide las coordenadas. Primero la letra, comprueva si es correcta, y mediante los codigos ASCII lo transforma en una posicion valida para ser un indice de la matriz. Luego con el número cumple el mismo procedimiento. Ambos valores son guardados en un struct coord, que es lo que devuelve la funcion.

Funcion: `void dibujarbarcos(unsigned short matriz1[][maxcol], unsigned short matriz2[][maxcol]);`

Pide las coordenadas donde ubicar los barcos y redibuja la pantalla despues de cada posicion ingresada, con el barco posicionado.

Funcion: `int FXRespuesta(unsigned short matriz1[][maxcol], struct coord Bnrec);`

Recibela coordenada ya pasada a struct y revisa la matriz en busca de respuesta, devuelve 'Tocado', 'Agua' o 'yaingresado', para pasarlo a procesar y su posterior envio.

Funcion: `void funcion_hundido(unsigned short matriz1[][maxcol]);`

Despues de cada coordenada recibida se encarga de buscar los 'Tocado'(una 'X' en el tablero) y compara sus posiciones, si el indice que guarda coincide con otros dos, tanto el primero de la matriz como el segundo, manda un mensaje por pantalla y lo marca con un '-'. De esta manera en el proximo chequeo no lo tomara como tocado. Esto se hace con dos lazos for para recorrer la matriz, al estilo de la busqueda secuencial.

Funcion: `int funcionCheck(unsigned short matriz1[][maxcol]);`

Mediante una busqueda secuencial por la matriz se busca la cantidad de tocado y se devuelve un short indicandola.

Funcion: `void FX_ganaste();`

Manda al oponente un mensaje indicando que gano y una trama de fin de partida.

Las restantes funciones hacen lo necesario para acomodar lo que se envia o lo que se recibe, de modo que las otras funciones puedan entender lo transmitido.

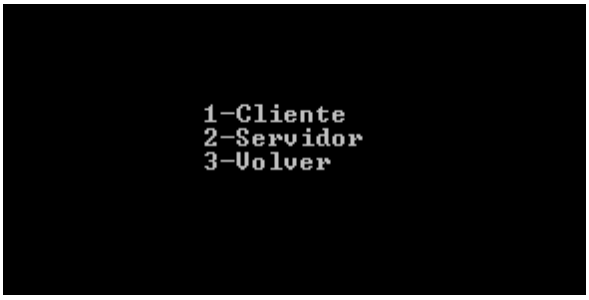
VI. Manual de usuario

A- Iniciando el juego

En el menu principal encontraras el titulo y las opciones jugar y salir.

B-Jugar

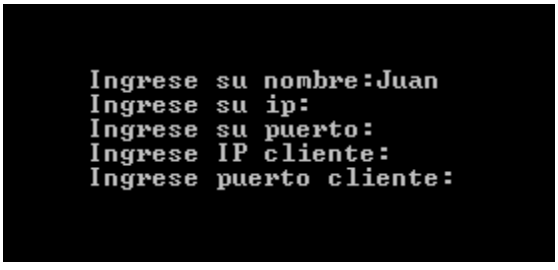
En la opcion jugar tenes las siguientes opciones:



Como cliente te conectaras a la partida del servidor y empezaras el juego recibiendo un ataque.

Como servidor seras el primero en atacar y el que espere los datos del cliente para comenzar el juego.

Con ambas opciones, tanto servidor como cliente, deberan cargar datos para la conexion.



C-Comenzando el juego

Antes de comenzar los ataques se deberas ubicar los barcos en tu radar. Los barcos figuraran en el tablero con los siguientes simbolos: '**?**' - '**¿**' - '**@**'.

Recorda poner las letras en mayuscula

Una vez ingresados los barcos iniciara la batalla. Si eres cliente quedaras esperando un ataque y si sos servidor atacaras.

Ya habiendo derrotado la flota enemiga aparecera en la pantalla un mensaje: “GANASTE” o “PERDISTE ...JAJAJA”.

Luego, presionando enter volvera al menu donde elegiste ser *servidor* o *cliente.*

VII. Conclusion

Se logro poner en practica de manera satisfactoria los temas explicados por los docente a lo largo de la catedra.

Se pudo desarrollar un software que nos permitio demostrar la implementacion de los conocimientos adquiridos.

Tambien se experimento, de alguna manera y salvando las diferencias, una metodologia de trabajo profesional.

VIII. Referencias

*Ing. Pascual, Gustavo y Bassi, Matias. “Comunicacion de programas en C mediante sockets“ - V1.0

*Aparicio, Hugo - ”Informatica I – Guia para estudiantes – Ingenieria electronica” - UNM Editora – 2015.

*https://www.youtube.com/watch?v=evmIH4nyTgE

*https://es.wikibooks.org/wiki/Programación_en_C