ESCUELA SUPERIOR POLITECNICA DEL LITORAL

Facultad de Ingeniería Eléctrica y Computación

PROYECTO DE GRADUACION

"Juego de Damas"

Previa a la obtención del Título de

INGENIERO EN COMPUTACION

PRESENTADA POR:

Guillermo Araujo

José Luis Loaiza

César Páez

Grace Pastorelly

Guayaquil - Ecuador

-:- 1 9 9 5 -:-

INDICE

1. ESPECIFICACIONES

1.1 DESCRIPCIÓN GENERAL DEL PROYECTO

Este proyecto consiste en un juego de damas que será implementado como un sistema CLIENTE-SERVIDOR. La parte servidora estará implementada en lenguaje C bajo una plataforma UNIX y la parte cliente en Visual Basic bajo la plataforma windows.

Se hará uso de Sockets para la comunicación entre nuestros programas(cliente y servidor) aprovechando su característica de usar protocolo TCP/IP. Para el desarrollo de la parte de comunicación del CLIENTE se usará DISTINCT.

1.2 REQUERIMIENTOS FUNCIONALES

Este sistema realizará las siguientes funciones:

- Mantendrá una lista de jugadores inscritos al club de juego de damas.
- Podrán jugar al mismo tiempo varios jugadores.
- Se llevará registro de los partidos .
- Cada jugador podrá invitar a otro siempre que este esté inscrito.
- Cada jugador podrá retirarse de cualquier partido cuando guste.
- Cada jugador podrá retirarse del juego cuando guste(solicitar ser eliminado de la lista de jugadores).
- Se mantendrá una cola de clientes.
- Se mantendrá un registro de los jugadores que han ganado.
- Implementará medidas de seguridad para que cada cliente accese al juego por medio de una contraseña.

1.3 REQUERIMIENTOS DE RENDIMIENTO Y CONFIABILIDAD

El rendimiento y la confiabilidad de nuestro sistema dependerá de dos factores : Primero del porcentaje de carga donde el programa servidor este corriendo. Segundo del tipo de enlace existente entre el CLIENTE y el SERVIDOR, por ejemplo:

Si el cliente está corriendo en una PC y el enlace es vía telefónica trabajando a 9600 bps, entonces el tiempo de respuesta para cada jugada no será más allá de 30 segundos. En cuanto a la confiabilidad, podemos decir que nuestro sistema es muy confiable (los protocolos TCP/IP son bastante confiables en la transmisión de datos) y solo se verá restringido por la calidad de la señal telefónica.

El performance del sistema se mantendrá en un nivel óptimo siempre y cuando el número de jugadas simultáneas sea hasta un máximo de 10, luego de lo cual este empezará a disminuir.

2. DISEÑO DEL PROTOCOLO

2.1 ARQUITECTURA CLIENTE - SERVIDOR DE LA APLICACIÓN

El cliente será realizado en Visual Basic 3.0, ya que este presenta gran facilidad de programación. La programación esta orientada a eventos, permite mantener funcionalidad en la implementación, así como facilidad en el manejo de archivos y uso de base de datos (ACCESS 1.1).

Se determinó que el CLIENTE mantendrá un archivo de configuración, que contiene información tanto del directorio en el cual se encuentra la base de datos, así como de la dirección ip del servidor con el que se desea comunicar y respectivo well-known port.

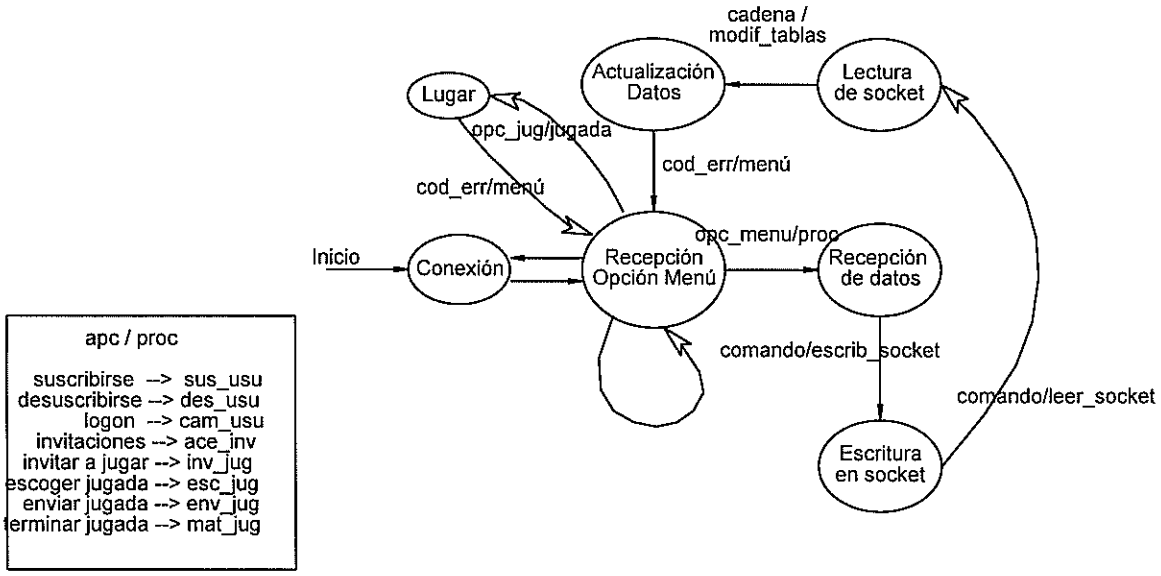
Otra de las razones por la cual el CLIENTE será desarrollado en V.B., es que la interfaz con el usuario es en ambiente gráfico.

Por otro lado, el SERVIDOR será implementado en lenguaje C con la característica que será concurrente para ofrecer servicio a diferentes usuarios al mismo tiempo. Del tipo stateless, y será orientado a conexión es decir utilizando los protocolos TCP/IP, para ofrecer seguridad en la transmisión de datos.

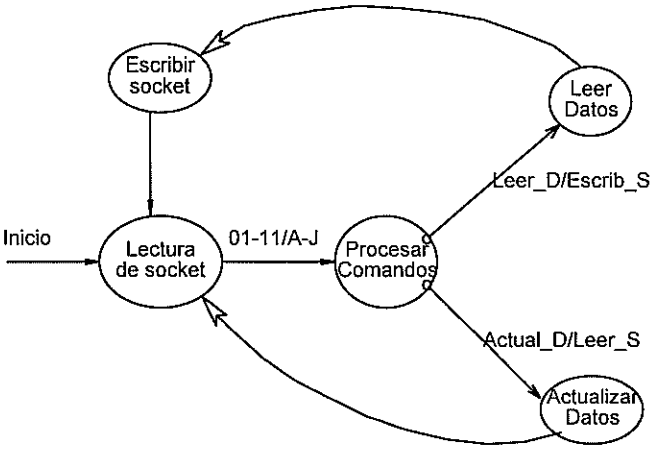
Para su implementación se usará lenguaje "C", que trabaje sobre el sistema operativo UNIX. , y para la comunicación con el cliente se lo hará a través de los sockets.

Así mismo, utilizará diferentes tablas para el almacenamiento de los datos de los jugadores y el estado de cada uno de sus partidos.

2.2 *MAQUINA DE ESTADOS DEL CLIENTE Y DEL SERVIDOR*



MAQUINA DE ESTADOS DEL SERVIDOR



Leer_D/Escrib_S	Actual_D/Leer_S
01/B ----> env_list 03/D ----> env_inv 06/G ----> env_jug	00/A ----> registrar 02/C ----> reg_inv 04/E ----> reg_resp_inv 05/F ----> recv_jugada 07/H ----> matar_partido 10/I ----> desuscribirse 11/J ----> env_no_opc

2.3 *SINTAXIS Y SEMANTICA DEL PROTOCOLO EN EL QUE SE BASAN EL CLIENTE Y EL SERVIDOR*

Las opciones con que dispone el cliente se encuentran representadas por un código único. Este código es de tipo string, y puede tomar los siguientes valores:

CODIGO	*DESCRIPCION*
01	Recibir la lista de jugadores ya suscritos, sin incluir la propia.
02	Invitar a alguien a jugar.
03	Recibir invitaciones de otros suscritores.
04	Enviar respuesta de invitaciones.
05	Enviar jugadas (el cliente envía).
06	Recibir jugadas (el cliente recibe).
07	Matar partidos en los que el suscritor esta involucrado.
10	Desuscribirse.
11	Obtener lista de los partidos que el subscriptor esta jugando.
12	Logon, Verificación de password
13	Suscribirse al juego

Tabla 2.3.1 Protocolo de comunicación de datos entre el servidor y el cliente

Si ocurriese un error en el servidor, este informa al cliente el tipo de error ocurrido, para lo cual se han definido un conjunto de posibles mensaje y errores. Los mensajes constan de 4 dígitos, que se encuentran representados por la serie 99xx. Los que se encuentran codificados mediante la siguiente tabla:

9900	El cliente fue registrado con éxito.
9901	El alias a usar esta siendo usado. Alias duplicado.
9902	Ud. ya se encuentra registrado. No puede ser registrado nuevamente.
9904	Opción no implementada.
9905	El contrincante se ha retirado del partido.
9906	Ud. ha ganado el juego.
9907	El contrincante ha ganado el juego.
9909	Clave Incorrecta.
9910	Todo fue realizado exitosamente.
9911	No se podido realizar la operación.
9912	No tiene Invitaciones.
9913	Ya está jugando con todos los suscriptores.

Tabla 2.3.2 Protocolo de comunicación de errores entre el servidor y el cliente

Se ha determinado que para facilidad de envío y recepción entre el cliente y servidor, se usara un delimitador de campo, así como un delimitador de fin de buffer.

;	delimitador de campo
#	delimitador de fin de buffer

es decir un típico buffer ya sea de envío o recepción tendrá un formato como el que se muestra:

cod_oper;campo_uno;campo_dos;campo_tres;campo_cuatro;.....;campo_n;#

Las fichas del tablero son representadas en el buffer mediante un identificador de 2 caracteres; donde el caracter mas significativo representa el color de la ficha, y el caracter menos significativo el tipo de ficha (dama o reina).

El color rojo será representado por un 1, mientras que el de color azul mediante un 1. Una ficha dama tiene un código 0 y una reina será representada por el código 1.

Código	Significado
00	azul dama
10	roja dama
01	azul reina
11	roja reina

Tabla 2.3.3 Color y tipo de ficha

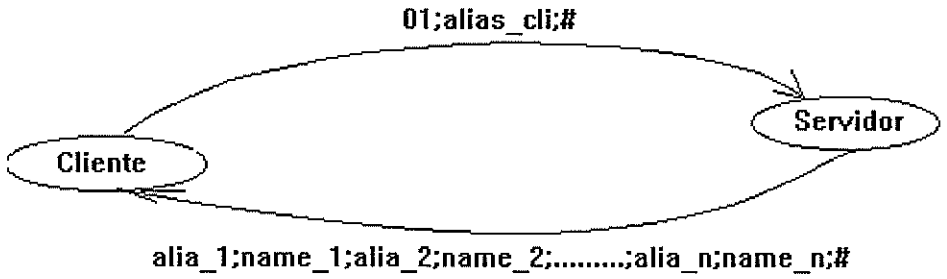
Las posiciones de las fichas en el tablero serán representadas desde la 0,0 la parte superior izquierda del tablero; y la parte inferior derecha como 8,8.

Un subscriptor puede encontrarse registrado solo una vez. Esto implica que no puede existir dos subscriptores con un mismo nombre o identificador (alias).

El alias de un subscriptor puede ser alfanumérico (a-z|A-Z|0-9) de longitud máxima de 8 caracteres. El nombre solo puede ser alfabético (A-Z|.) y de una longitud máxima de 35 caracteres.

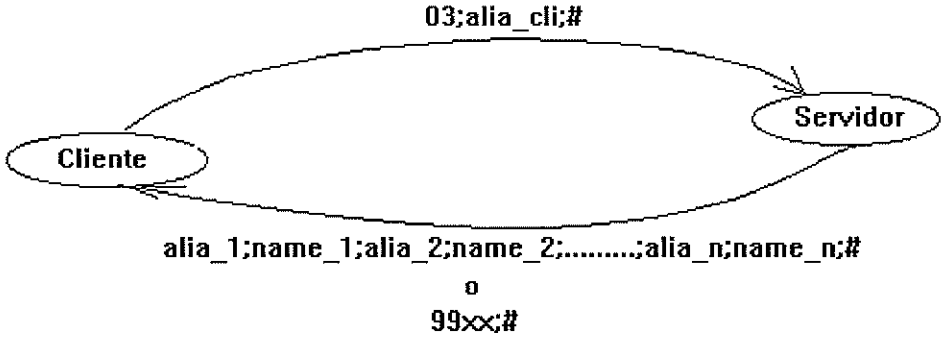
2.3.1 Recibir lista de jugadores (01):

Una vez recibido el buffer del servidor, el cliente envía el correspondiente mensaje 99xx al servidor.

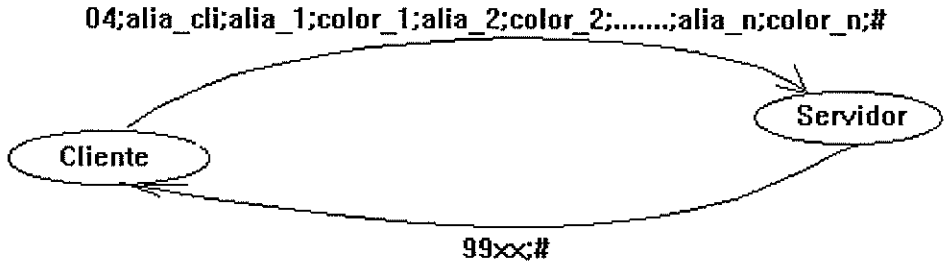


2.3.2 Invitar a alguien a jugar (02):

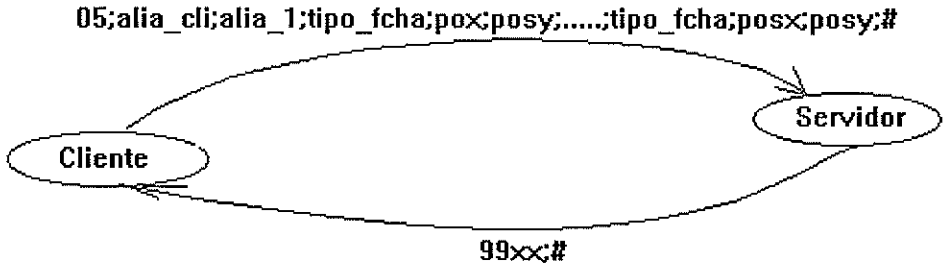
2.3.3 Recibir Invitaciones (03):



2.3.4 Enviar Respuesta de Invitación (aceptar invitación) (04):

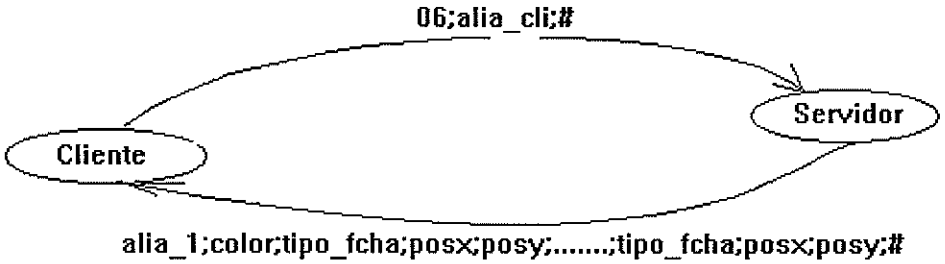


2.3.5 Enviar jugada (05):



Las jugadas por el cliente serán almacenadas en la tabla fichas_out hasta el momento de su envío, lo que se hará jugada por jugada, hasta que hayan sido enviadas todas.

2.3.6 Recibir jugada (06):



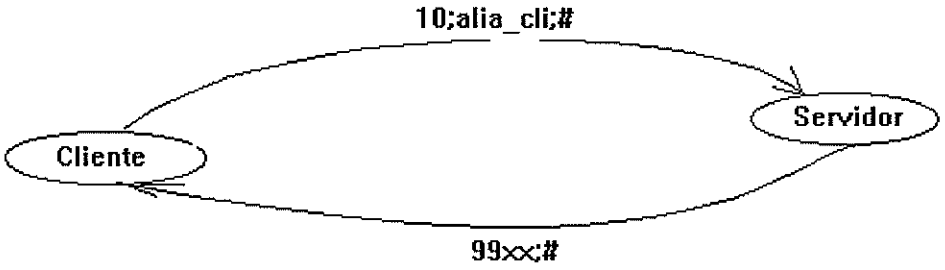
ALIAS1 + DELIMITADOR + COLOR + TIPO FICHA + DELIMITADOR + POSX + DELIMITADOR + POSY + DELIMITADOR + TIPO FICHA + DELIMITADOR + POSX + DELIMIATDOR + POSY + DELIMITADOR + ...+ CHR(13) + ALIAS2 + DELIMITADOR + TIPO FICHA + DELIMITADOR + POSX + DELIMITADOR + POSY + DELIMITADOR + TIPO FICHA + DELIMITADOR + POSX + DELIMIATDOR + POSY + DELIMITADOR + FIN BUFFER

Una vez enviado el primer tramo al servidor, el cliente espera por el buffer indicado arriba, y este es grabado en la tabla fichas y contrin de la base para mayor facilidad de uso.; y será mostrado al cliente para que este realice la selección del partido que desea realizar la jugada. El color especificado en la trama de recepción del cliente, es el color con que el cliente realiza la partida.

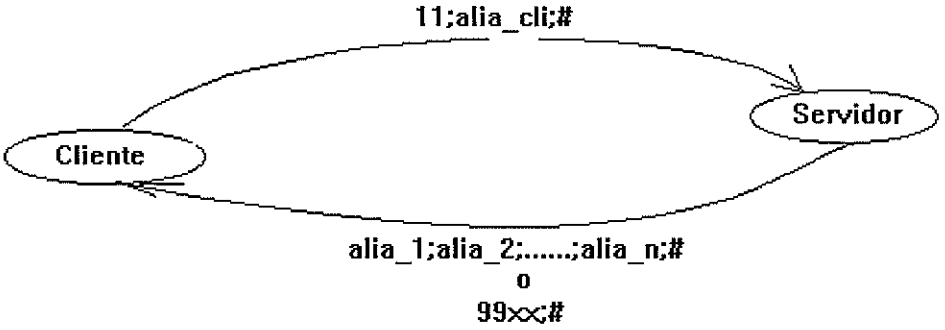
El cliente leerá se mantendra en lectura hasta detectar que el servidor ya no posee más datos para enviar.

2.3.7 Matar partidos (07):

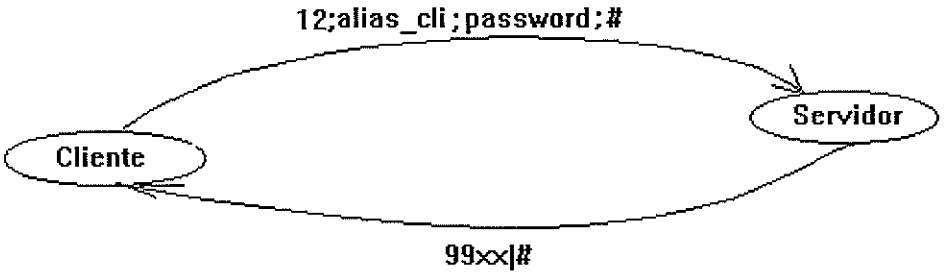
2.3.8 Desuscribirse (10):



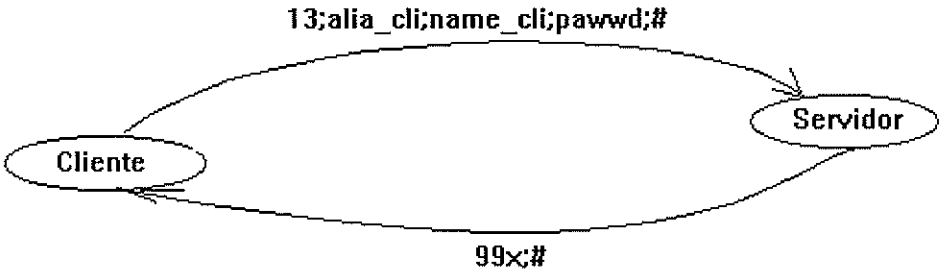
2.3.9 Traer Lista de Partidos del Suscriptor (11):



2.3.10 Logon(12)



2.3.11 Subscribirse (13):



3. DISEÑO DEL SERVIDOR

3.1 FUNCIONALIDADES DEL SERVIDOR

El servidor se encargará de mantener tablas con datos relacionados al juego en sí, tales como jugadores, invitaciones ,etc. Las tablas son :

1. Tabla de jugadores, con su nombre, alias y password.

2. Tabla de invitaciones, conteniendo a los jugadores que invitan, los que son invitados y su estado (esto es: si no recibe respuesta, si ya aceptó la invitación) y en el caso de haber terminado el juego registrará quien lo ganó.

3. Tabla de estado, que registre el estado actual de cada uno de los partidos que se han generado.

De esta forma podrá dar servicio al cliente respondiendo cada uno de sus requerimientos. Estas respuestas que dará el servidor se traducen en las siguientes funciones:

- Comprobar que el cliente está autorizado para jugar y verificar su contraseña.
- Inscribir a un jugador cuando el cliente lo solicite, asi pasa a formar parte del club de juego de damas.
- Enviar una lista de todos los jugadores inscritos cuando el cliente lo solicite
- Registrar una invitación cada vez que el jugador_a invita al jugador_b e informar sobre el particular al jugador_b.
- Registrar las jugadas que cada jugador realiza, esto es, mantener el estado actual de cada partido.
- Matar los partidos cuando cualquier jugador asi lo solicite, esto no implica que el jugador deje de formar parte del club de juego de damas.
- Eliminar a un jugador del club de juego de damas cuando lo solicite.

3.1.1 *Tipo de Servidor y su Justificación*

Puesto que queremos desarrollar un sistema "JUEGO DE DAMAS" en el que cualquier persona pueda jugarlo sin importar el lugar en el que se encuentre, necesitamos usar ciertos protocolos de comunicación que nos provea de mecanismos básicos para la transferencia de datos, y sobre todo, que dicha transferencia se realice de manera confiable y con un alto grado de performance, además que pueda trabajar tanto en redes LAN como en WAN, estos protocolos son los protocolos TCP/IP.

Además queremos que los jugadores se conecten en el momento que deseen, teniendo presente que usaremos los protocolos TCP/IP y considerando que tales protocolos no dictaminan cuándo o porqué las aplicaciones interactúan , debemos organizar nuestra aplicación en un ambiente distribuído. Quien nos ofrece este ambiente es el modelo CLIENTE-SERVIDOR, de tal manera que para implementar nuestro sistema "Juego de Damas" usaremos el modelo cliente-servidor.

Así tendremos un servidor que esté escuchando continuamente para determinar cuándo arriva un requerimiento de conexión y establecer el enlace.

3.1.2 Servidor no orientado u orientado a conexión?

Dentro de la serie de protocolos TCP/IP tenemos dos tipos de protocolos de transporte, estos son : TCP y UDP .

El protocolo de transporte TCP provee toda la confiabilidad necesaria para comunicarse a través de una red internet, sus características son las siguientes:

1) Este verifica que los datos arriven y automáticamente retransmite los datos que no lo hacen.
2) Contabiliza un checksum sobre los datos para garantizar que no fueron corrompidos durante la transmisión.
3) Utiliza un número de secuencia para asegurar que los datos arrivan en orden y automáticamente elimina paquetes duplicados.
4) Provee control de flujo para asegurar que el emisor no transmita datos más rápido de lo que el receptor pueda consumir.
5) Informa a ambos el cliente y al servidor si la red llega a ser inoperable por alguna razón.

El protocolo de transporte UDP depende del hardware de la red y de los gateways intermedios. Si trabaja en una red de área local entonces lo harán correctamente. Pero si es en una WAN puede tener problemas puesto que los datos pueden ser perdidos, duplicados, retardados o distribuídos fuera de orden.

Con este antecedente y tomando en cuenta que a nuestro juego pueden conectarse de diferentes lugares en forma confiable entonces necesitamos usar un protocolo de transporte orientado a conexión, de esta manera utilizaremos TCP.

3.1.3 Servidores Stateless o Statefull?

Un servidor statefull es aquel que guarda información de estado acerca de las interacciones que está haciendo con el cliente, de esta forma se gana eficiencia pero no es muy confiable porque esa información puede llegar a ser incorrecta si los mensajes son perdidos, duplicados o distribuídos fuera de orden o si el cliente corta bruscamente la conexión y reinicia (reboot) la máquina. Un servidor stateless es aquel que no mantiene información de estados.

En nuestro caso haremos al servidor stateless debido a que este consume menos recursos que el statefull y puesto que será instalado en la SUN Sparc Station 2 de CESERCOMP debemos cuidar de sus recursos, ya que ésta provee muchos servicios más, que necesitan de esos recursos (no recargar ni alterar el performance de los sistemas que en él están corriendo).

3.1.4 Servidor Concurrente o Iterativo?

El término CONCURRENCIA se refiere a un procesamiento simultáneo real o aparente.

El procesamiento concurrente es fundamental en un ambiente distribuído y puede ocurrir en muchas formas. Por ejemplo este ocurre entre máquinas sobre una red, en la cual muchas parejas de aplicaciones pueden comunicarse concurrentemente, "compartiendo la red que las interconecta". La concurrencia también ocurre tanto en CLIENTES como en SERVIDORES.

Para efectos de programación del cliente, este no necesita realizar ninguna consideración especial para desarrollar su CLIENTE CONCURRENTE puesto que es el sistema operativo quien se encarga de permitir que múltiples usuarios invoquen al cliente concurrentemente.Los servidores para conseguir

concurrencia requieren de un esfuerzo considerable. El SERVIDOR CONCURRENTE se encarga de permitir que múltiples usuarios se conecten a él para solicitar un servicio. De esta forma atiende a varios usuarios simultáneamente, y lo puede hacer gracias a la ayuda que le provee el Sistema Operativo.

Los SERVIDORES ITERATIVOS manejan un solo cliente a la vez, lo que produce que existan varios clientes que tendrán que esperar un tiempo considerable para poder ser atendidos. En otras palabras, una vez que un cliente se conecta con el servidor, se apropia de él y no es sino, hasta que termina todas sus transacciones y lo libera, que otro cliente puede accederlo.

Como nuestro juego de damas debe permitir que varios clientes jueguen simultáneamente, debemos implementar un SERVIDOR CONCURRENTE.

3.2 DISEÑO DE LOS DATOS MANEJADOS EN EL SERVIDOR

Usaremos los siguiente tipos de datos:

3.2.1 Estructuras

sockaddr_in	Estructura que contiene una dirección IP y un número de puerto de protocolo
u_short sin_family	tipo de dirección
u_short sin_port	número del puerto del protocolo
u_long sin_addr	dirección IP
char sin_zero[8]	no usado (seteado a 0)

el cliente usará esta estructura como cli_addr, y el servidor como svr_addr.

3.2.2 Variables de Tipo Entera

argc	es el número de parámetros pasados a la función main
port	es el número del puerto
msock	es un descriptor entero para el socket maestro
ssock	es un descriptor entero para el socket esclavo
fatherpid	es el identificador del proceso padre
length	longitud de la estructura cli_addr

3.2.3 Variables de Tipo Char

argv[]	contiene los parámetros pasados a la función main
*hos_ip	contiene la dirección ip del host

ARCHIVOS

players.txt	cada línea contiene la lista de los jugadores inscritos en el club de damas

alias ; nombre_jugador ; partidos_ganados ; partidos_perdidos
alias ; nombre_jugador ; partidos_ganados ; partidos_perdidos
.
.
.
alias ; nombre_jugador ; partidos_ganados ; partidos_perdidos"

invita.txt	cada línea contiene las invitaciones hechas entre jugadores, si ha sido aceptada la invitación, el color elegido por el invitado

"alias_invitador ; alias_invitado ; V
alias_invitador ; alias_invitado ; F
.
.
alias_invitador ; alias_invitado ; V

damas.txt	cada línea contiene el alias de quien le toca jugar primero,el alias del otro jugador,el color de quien le toca el turno, el tipo de ficha (si es azul o roja y simple o reina), la posición en X , la posición en Y

alias1 ; alias2 ; color1; tipo_ficha ; pos_x ; pos_y ; tipo_ficha ; pos_x ; pos_y ;.....
alias1 ; alias2 ;color1 ; tipo_ficha ; pos_x ; pos_y ; tipo_ficha ; pos_x ; pos_y ;.....
.
.
.
alias1 ; alias2 ;color1 ;tipo_ficha ; pos_x ; pos_y ; tipo_ficha ; pos_x ; pos_y ;.....

3.3 DISEÑO DE LA APLICACION SERVIDORA.-

Nuestro diseño se basa en un sistema del tipo Top-Down, con una función principal que se encarga de llamar a las otras, a medida que las necesite.

3.3.1 Algoritmo PRINCIPAL

1. Recibir los parámetros del host y del puerto
2. Crear un socket y enlazarlo a una dirección conocida para el servicio que será ofrecido.
3. Poner el socket en modo pasivo, habilitándolo para que sea usado por el servidor.
4. Hacer una llamada a *accept* para recibir el próximo requerimiento de un cliente.
5. Crear un nuevo proceso esclavo para manejar su respuesta. Use *fork()*
6. Si está en el proceso padre, cerrar el socket del proceso hijo
7. Si No, cerrar el socket del proceso padre y llamar al algoritmo *cliente* para procesar el requerimiento solicitado.
8. Regresar al paso 4.

3.3.2 Algoritmo RECV_CLI

1. Crear un puntero a char *cadena* en donde almacenaremos el string enviado por el cliente.
2. Leer un caracter del socket, haciendo un read (socket, buff, buflen)
3. Si el caracter es diferente de '#', concatenarlo a *cadena*.
4. Si No, retornar *cadena* y terminar
5. Regresar al paso 2.

3.3.3 Algoritmo CORTAR_CAMPO

1. Recibe un string
2. Crear un puntero a char *cadena* en donde almacenaremos el comando enviado en el string.
3. Leer por caracteres hasta encontrar el delimitador de campo (;), y almacenarlo en *cadena*
4. Sacar del string dado lo que contiene cadena
5. Retornar *cadena* y terminar

3.3.4 Algoritmo CLIENTE

1. Recibir el socket del esclavo que procesará este requerimiento.
2. Obtener el string enviado por el cliente llamando al algoritmo *recv_cli.*
3. Obtener el comando del requerimiento llamando al algoritmo cortar_campo.
4. Si el comando es 13, registrar al jugador en el club de juego de damas. *registrar*
5. Si el comando es 1, Enviar la lista de los jugadores ya suscritos. *env_list.*
6. Si el comando es 2, Registrar invitación para un partido. *reg_inv.*
7. Si el comando es 3, Enviar invitaciones para iniciar un partido. *env_inv.*
8. Si el comando es 4, Registrar la respuesta a una invitación.*reg_resp_inv.*
9. Si el comando es 5, Recibir una jugada, *recv_jugada*
10. Si el comando es 6, Enviar jugadas, *env_jugada.*
11. Si el comando es 7, matar las partidas que el usuario desee. *matar_partido.*
12. Si el comando es 8, Saltar al paso 16.
13. Si el comando es 10, Desuscribirse del club de juego de damas. *desuscribir.*
14. Si el comando es 12, Verificar la contraseña (el password) llamando al algoritmo Logon
15. Si el comando no es ninguno de los anteriores, Enviar un mensaje al cliente. *env_no_opc.*
16. Regresar al paso 2.
17. Concluir la sesión con ese cliente haciendo un *shutdown.*

3.3.5 Algoritmo REGISTRAR

1. Recibe un string
2. Crear un puntero a char *cadena* para almacenar cada uno de los campos
3. Leer un caracter del string y eliminarlo del string
4. Si el caracter es el delimitador de buffer (#), saltar al paso 8
5. De lo contrario, Si el caracter es diferente del delimitador de campo (;), almacenarlo en *cadena*
6. De lo contrario, añadir *cadena* y un delimitador de campo (;) al archivo players.txt
7. Regresar al paso 3.
8. Añadir un retorno de línea al archivo players.txt, luego cerrarlo .
9. Enviar un OK al cliente y retornar.

3.3.6 *Algoritmo ENV_LIST*

1. Recibe un string
2. Crear un puntero a char *lista_f* para almacenar los alias de los jugadores que no serán enviados ya que ellos ya fueron invitados por el solicitante o ellos ya lo invitaron.
3. Crear un puntero a char *cadena* para almacenar la lista de los alias y sus respectivos nombres que serán enviados al solicitante.
4. Obtener el *alias* del jugador que solicita la lista
5. Abrir el archivo invita.txt y ubicarse en la primera línea.
6. Si es fin de archivo, Saltar al paso 11
7. Leer una línea del archivo invita.txt y sacar *alias1* y *alias2* de esa línea
8. Si alias1 es igual a alias, añadir alias2 en *lista_f*
9. Si alias2 es igual a alias, añadir alias1 en *lista_f*
10. Regresar al paso 6
11. Cerrar el archivo invita.txt
12. Abrir el archivo players.txt y posicionarse en la primera línea.
13. Si es fin de archivo, Saltar al paso 18
14. Leer una línea del archivo players.txt y sacar alias1 y nombre1
15. Si alias1 es diferente de alias y no consta en *lista_f,* añadir alias1 y nombre1 a cadena
16. Añadir un delimitador de campo (;) a cadena
17. Regresar al paso 13
18. Cerrar el archivo players.txt
19. Añadir un delimitador de fin de buffer (#) a cadena
20. Enviar cadena al cliente y retornar.

3.3.7 Algoritmo REG_INV

1. Recibe un *string*
2. Obtener el *alias* del jugador que realiza las invitaciones y eliminarlo de *string.*
3. Obtener el *alias1* del jugador a quien invita *alias* y eliminarlo de *string* .
4. Añadir una línea al archivo invita.txt que contenga *alias, alias1, F*
5. Enviar un OK al cliente y Retornar.

3.3.8 Algoritmo ENV_INV

1. Recibe un *string*
2. Crear un puntero a char *lista* que guardará la lista de los *alias_invitador* que han invitado a *alias* y que aún no han recibido respuesta.
3. Crear un puntero a char *cadena* que guardará los *alias_invitador,* el delimitador, y su respectivo *nombre*
4. Obtener el *alias* del jugador que quiere revisar la lista de los jugadores que lo han invitado
5. Abrir el archivo invita.txt y obtener todos los *alias_invitador* que han invitado a *alias* y aún no han recibido respuesta, luego guardarlo en *lista.*
6. Abrir el archivo players.txt y para cada *alias_invitador* que se encuentre en *lista* obtener su *nombre,* luego añadir: *alias_invitador,* delimitador de campo (;) y *nombre,* a *cadena.*
7. Añadir el delimitador de fin de buffer (#) a *cadena.*
8. Cerrar los archivos, invita.txt, players.txt
9. Enviar *cadena* al cliente y retornar

3.3.9 Algoritmo REG_RESP_INV

1. Recibe un string
2. Abrir los archivos invita.txt y damas.txt
3. Obtener el *alias* del jugador que envía respuestas a las invitaciones y eliminarlo del *string*
4. Si encontró el delimitador de fin de buffer (#), Saltar al paso 9
5. Obtener *alias1* y *color1* del *string* y luego eliminarlo del *string.*
6. Buscar en el archivo invita.txt la línea que contiene *alias1 ; alias* , y en el tercer campo, en lugar de F ponga V .
7. Añadir una línea al archivo damas.txt que contenga el "estado inicial del nuevo partido", es decir, el alias de quien le toca jugar de acuerdo al color escogido por el invitado, el alias del otro jugador, el color de quien le toca jugar primero, el tipo de ficha, pos_x, pos_y..., con sus respectivos delimitadores de campo.
8. Regresar al paso 4
9. Cerrar los archivos invita.txt y damas.txt .
10. Enviar un OK al cliente.

3.3.10 Algoritmo RECV_JUGADA

1. Recibe un *string*
2. Abrir los archivos invita.txt y damas.txt
3. Obtener el *alias1* y el *alias2* del *string*
4. Verificar en el archivo invita.txt si existe un partido entre *alias1* y *alias2 .Si es asi continuar, sino saltar al paso xxx*
5. Añadir un línea al archivo damas.txt eliminando los delimitadores de fin de campo y de fin de buffer finales.
6. Cerrar los archivos invita.txt y damas.txt .
7. Enviar un OK al cliente.

3.3.11 Algoritmo ENV_JUGADA

1. Recibe un *string.*
2. Declarar un puntero a char *cadena* que guardará las jugadas de los diferentes partidos en los que está jugando *alias*
3. Obtener el *alias* del jugador que solicita el envío de las jugadas
4. Abrir el archivo damas.txt y posicionarse al principio del archivo.
5. Buscar en el archivo damas.txt la siguiente línea que contengan como primer alias el *alias* obtenido en el paso anterior y guardarla en cadena.
6. Si encuentra la línea anterior eliminarla del archivo , enviarla al cliente y saltar al paso 5 ,sino continuar.
7. Retornar

3.3.12 Algoritmo DESUSCRIBIR

1. Recibe un *string*
2. Obtener el *alias* de quien quiere desuscribirse del club de juego de damas
3. Buscar en el archivo damas.txt todas las líneas que contienen a *alias* y eliminarlas, luego por cada línea eliminada añadir una línea que contenga, *alias1, alias,color_cualquiera, codigo (9905).*
4. Eliminar del archivo invita.txt la línea que contenga *alias*
5. En el archivo players.txt eliminar el jugador *alias*
6. Retornar

3.3.13 Algoritmo MATAR_PARTIDOS

1. Recibe un *string*
2. Obtener el *alias* de quien quiere matar los partidos, y eliminarlo de *string*
3. Si obtiene el delimitador de fin de buffer (#), saltar al paso 9
4. Obtener *alias1* y eliminarlo de *string.*
5. Buscar en el archivo damas.txt la línea que contiene *alias* y *alias1* y eliminarla, luego añadir la línea que contenga,*alias1,alias,color cualquiera,codigo (9905).*
6. Eliminar del archivo invita.txt la línea que contenga *alias* y *alias1*
7. En el archivo players.txt, aumentar en uno el tercer campo (partidos_ganados) del jugador *alias1,* y aumentar en uno el cuarto campo (partidos_perdidos) del jugador *alias*
8. Regresar al paso 3
9. Retornar

3.3.14 Algoritmo ENV_NO_OPC

1. Enviar el siguente comando al cliente: "9999 | #"
2. Retornar

3.3.15 Algoritmo LOGON

1. Recibe un string
2. Obtener el *alias* y el *password* del jugador que quiere jugar.
3. Abrir el archivo players.txt y obtener la linea que empieza con *alias*
4. De la línea anterior comparar el tercer campo con *password. Si son iguales enviar al cliente un OK, de lo contrario enviar un codigo de error.*
5. *Retornar*

3.4 COMPROMISOS DEL DISEÑO DEL SERVIDOR

En el servidor se utilizarán archivos para el almacenamiento de todos los datos concernientes a los jugadores y el estado de sus partidos. Con esto se pretende facilitar la administración del servidor y dar mayor facilidad al desarrollo de la aplicación.

Todas las transacciones hechas por el servidor serán realizadas de manera coherente para evitar conflictos con los datos de los jugadores, se mantendrá un estricto control de acceso para dar mayor seguridad a los jugadores y sus partidos evitando asi filtraciones de información.

Entre los archivos que usaremos tenemos a players.txt en el cual se guardarán los usuarios, sus nombres, sus contraseñas y el número de partidos ganados y perdidos.

El archivo invita.txt que contiene las distintas invitaciones realizadas entre los jugadores, asi como también la respuesta dada, sea esta afirmativa o negativa, y el color escogido por el jugador invitado.

Finalmente tendremos el archivo damas.txt quien se encargará de mantener el estado en el que se encuentran todos los partidos existentes en el juego, y para cada partido se indicará el jugador a quien le corresponde el turno de juego junto con su color. En este mismo archivo se encontrarán los partidos que hayan sido eliminados por cualquiera de los jugadores propietarios de aquel partido.

Para la comunicación con el cliente se definirá un protocolo de comunicación orientado a cadena de caracteres, el cual será armado por el cliente para realizar todos sus requerimientos. De igual forma el servidor retornará al cliente los datos solicitados por el mismo, un código para indicarle que su solicitud ha sido realizada, o un código de error. Todo esto se encuentra detallado en el protocolo de comunicaciones mencionado anteriormente.

3.5 ADMINISTRACIÓN DEL SERVIDOR

Realmente la tarea que tendrá el administrador del sistema para controlar y mantener el servidor de damas será mínima, ya que todas las transacciones se las realiza de manera automática con el cliente. Por ejemplo , cuando el cliente hace un requerimiento de suscripción, el servidor obtiene el string de datos enviado por el cliente, realiza la suscripción y le devuelve un mensaje al cliente para indicarle que su pedido fue ejecutado.

Entre los puntos que tendrá el administrador que realizar se encuentran los siguientes.

1. Verificar la correcta configuración del puerto de acuerdo a los pasos indicados en el MANUAL DE USUARIO DEL SERVIDOR.

2. Levantar el servidor indicando la dirección IP y el puerto que va a ser utilizado por el mismo.

3. Informar a los usuarios clientes del puerto que deberán utilizar para conectarse con el servidor de damas, y sobre todo , tener presente que ese puerto debe ser utilizado única y exclusivamente para brindar ese servicio, evitando asi conflictos con otros servicios.

4. Tomar en cuenta que el servidor utiliza un número de archivos temporales para ejecutar ciertas transacciones con los clientes , los mismos que son borrados automáticamente por el servidor una vez que haya terminado su transacción con aquel cliente. Estos archivos temporales toman diferntes nombres de acuerdo al cliente con el que estén trabajando.

5. Si por alguna razón, los archivos temporales no se están borrando, entonces el administrador deberá hacer limpieza con determinada frecuencia, para esto, primero debe bajar el servidor , luego verificar que solamente deben existir los siguientes archivos:

   damas.exe
   players.txt
   invita.txt
   damas.txt

   El resto tendrá que ser borrado.

6. Todos los archivos .txt están conformados por líneas, las mismas que contienen un determinado número de campos, los cuales a su vez, están separados por el delimitador de punto y coma **(;)**.

7. Las líneas del archivo players.txt contienen 5 campos, estos son:
   alias;nombre;password;partidos ganados;partidos perdidos
   En este se encuentran todos los jugadores que se han suscrito al clud de juego de damas.

8. Las líneas del archivo invita.txt contienen los siguientes campos:
   alias del que invita ;alias del invitado;respuesta
   el campo *respuesta* será V o F dependiendo de si ha sido aceptada o no la invitación.

9. Las líneas del archivo damas.txt contienen el alias del jugador al que le tocaría jugar , luego el alias del jugador que estaría en espera, el color de quien le toca el turno y finalmente las fichas y sus posiciones de acuerdo a un cierto formato.

   Solamente en el caso de que uno de los jugadores decide matar un partido, aparecerá una línea conteniendo los dos alias y un código (9905), el mismo que sirve para informarle al otro jugador que su partido ha sido muerto por su contrincante. Una vez informado sobre este particular, la línea desaparece del archivo.

10. CAMBIO DE PASSWORD.- Si por algún motivo uno de los jugadores desea cambiar el password, entonces el administrador deberá ir al archivo players.txt y cambiar el tercer campo por el nuevo password en la línea que corresponde al usuario que lo solicita.

4. DISEÑO DEL CLIENTE

4.1 FUNCIONALIDAD DEL CLIENTE

El CLIENTE será realizado en Visual Basic versión 3.0, por presentar gran comodidad de programación, así como el hecho de que ya sabíamos manejarlo, esto se debe a que la programación esta orientada a eventos, permite definir funciones, así como tipos de datos, facilidad en el manejo de archivos y de la base de datos en ACCESS 1.1.

El CLIENTE será implementado de tal manera que este use eficientemente el espacio en disco, es decir se eliminara código redundante, para lo cual se hará uso de funciones, manteniendo así funcionalidad.

Para la implementaron del cliente, se usará un archivo de inicialización, damas.ini, donde se especifíca el número de puerto y dirección ip del servidor, lo cual hace que este sea portable, y fácil de mantener en caso de que el numero de puerto o dirección del servidor sea alterada, así como el path del directorio donde se almacenan las tablas de la base.

Las opciones del menú estarán disponibles al cliente conforme este se introduzca en el juego.

El manejo de datos de las jugadas serán almacenado en tablas, lo que permitirá que el acceso a las jugadas sea rápida y eficiente.

Para manejar con mayor comodidad los errores que se presentasen durante el uso de las funciones de WINDOWS SOCKET, así como los obtenidos durante la transmisión o implementación, se usara una tabla, error_code. Esta tabla permitirá obtener el significado del error obtenido, lo que hace que futuros errores que surgiesen sean ingresados en esta, evitando así que la codificación sea alterada.

Las posiciones de las fichas en el tablero de los juegos serán almacenadas en un tabla de entradas, fichas; luego de realizar el movimiento, estas serán trasladadas a una tabla de salida, fichas_out.

Los movimientos de las fichas dentro del tablero serán validadas en lo permisible, así como que el usuario no realice mas de un movimiento consecutivo.

Será capaz de mantener información de varios partidos que el usuario pueda mantener simultáneamente.

Deberá establecer la conexión con el servidor, así como la consistencia de los datos que serán enviados según el protocolo propuesto. Mantendrá de igual manera, la autenticación y autorización del usuario.

Proveerá al usuario de ayuda en línea en cada una de las opciones posibles del juego, así como de un archivo de ayuda.

4.2 DISEÑO DE DATOS MANEJADOS EN EL CLIENTE

4.2.1 Archivo DAMAS.INI

Contiene variables de inicialización del sistema, como son:

well_port

host_ip

definen las características de la comunicación como son: numero de puerto donde escucha el servidor y dirección del servidor(en notación decimal).

PATH

que contiene el directorio donde se almacena la tabla de datos, así como otros archivos propios de la aplicacion cliente.

4.2.2 Base de Datos DAMAS.MDB

Usada para almacenar datos sobre los diferentes juegos pendientes que podría tener el jugador, códigos de error, invitaciones recibidas.

Tabla FICHAS, FICHAS_OUT

Perteneciente a la base de datos damas.mdb, sus campos son:

usuario	String(8)	Alias del contrincante
equipo	Integer	Identifica tipo de ficha: 1=roja,2=azul
posx	Integer	posición en x de la ficha en el tablero
posy	Integer	posición en y de la ficha en el tablero
reina	Integer	bandera: 1=reina, 0=no reina

TABLA ERROR_CODE

Contiene el código de error y su respectiva descripción.

err_code	string (5)	Código de error
descrpcion	string(100)	Descripción del error

esta tabla contiene tanto posibles errores de las funciones de WINSOCK como de los definidos en la aplicación.

TABLA INVITACIONES

Contendrá las invitaciones que el usuario hubiera recibido de otros suscritores y que no han sido contestadas.

alias	string(8)	Contiene el alias del suscriptor del cual se obtuvo la invitación.
name	string(35)	Nombre del suscriptor.

TABLA CONTRIN

En esta tabla se almacena el nombre al contrincante con quien el usuario tiene partidos en los que puede realizar jugada con otros jugadores, color de ficha con la que el usuario mantiene la partida, asi como el estado de si ya se ha efectuado el movimiento o no.

alias	string(8)	Alias del jugador con quien el usuario mantiene partido.
color	integer	0 si el usuario juega con las rojas y 1 si es con azul.
est	boolean	True si el movimiento de la jugada ya fue efectuado, False si aun no se efectua

Las datos de las tablas de un usuario específico son devueltas al servidor si este termina la ejecución del programa o si se decide cambiar de usuario.

4.2.3 Estructuras

Estructura Tipoficha

Esta estructura sirve para llevar el estado de las fichas en el momento mismo de la jugada con un contrincante específico, con estos datos se realizan las validaciones necesarias para hacer de una jugada válida. Esta estructura es llenada de datos(traídos de una base de dato) antes de la jugada, y luego de la misma los valores almacenados en la estructura son copiados en la base de datos.

posx	Integer	*posición en x de la ficha en el tablero
posy	Integer	*posición en y de la ficha en el tablero
reina	Integer	bandera: 1=reina, 0=no reina
estado	Integer	bandera: 1=jugando,0= perdida

Tipo list_alias

Define el alias del cliente, con su respectivo nombre.

alias	String(8)	Contiene el alias del cliente
name	String(35)	Define el nombre del cliente

4.2.4 Variables Globales

NOMBRE	TIPO	DESCRIPCION
des_fich	Single	distancia entre los contornos d un cuadro del tablero y la ficha (objeto).
tam_cua	Single	dimensión de un cuadro del tablero
		es la ordenada de la esquina superior izquierda del tablero con respecto a la forma ficha.frm
usuario	String	alias del contrincante
fic_act	String	valor índice de los arreglos (ficha_roja, ficha_azul) escogido para la jugada
tipf	String	tipo de ficha escogida a la que le toca jugar
turno	String	guarda el tipo de ficha a la que le toca jugar
f_azul(12)	Tipo_ficha	almacena estado actual de las fichas azules
f_roja(12)	Tipo_ficha	almacena estado actual de fichas rojas
delimeter	String	contiene el delimitador de campo
endbuff	String	contiene el delimitador de fin de buffer
CNTD_K	Integer	contiene la cantidad de partidos en los que el usuario se encuentra
CNTD_I	Integer	contiene cantidad de invitaciones recibidas por el usuario
CNTD	Integer	cantidad de jugadores a los que se puede invitar
FILEIN	String	contiene el nombre del archivo de inicialización
ERR_DB	String	contiene el nombre de la tabla de errores
PATH	String	contiene el path del directorio del cliente
kill[]	String * 8	contiene alias de partidos posibles a matar

4.3 DISEÑO DE LA APLICACION CLIENTE

4.3.1 SUSCRIBIRSE

1. Ingreso el nombre del usuario, alias a usar y password.
2. Si alguno de los campos a ingresar se encuentra en blanco o el alias comienza con caracter numérico, presenta mensaje de error respectivo, regrese al 1
3. Armar buffer de envío.
4. Enviar al servidor el buffer
5. Recibir el buffer del servidor
6. Obtener el código del buffer recibido, su significado y presentarlo.
7. Actualizar archivo de inicialización "damas.ini"
8. Actualizar opciones del menú.

4.3.2 VERIFICACION DE USUARIO

1. Ingrese el password con el que se ha suscrito.
2. Armar buffer de envío al servidor.
3. Enviar el buffer a servidor.
4. Recibir buffer de verificación del servidor.
5. Obtener el código de respuesta.
6. Si el código es 9910, presentar menú principal.
7. Si el código es 9911, regrese al paso 1.

4.3.3 RECIBIR LISTA DE JUGADORES

1. Armar buffer de envío.
2. Enviar al servidor el buffer
3. Recibir el buffer del servidor.
4. Si el buffer recibido comienza con 99, obtener significado y presentarlo, continuar al paso 6.
5. Desencadenar el buffer recibido y almacenarlo en el arreglo.
6. Actualizar archivo de inicialización y las opciones del menú.

4.3.4 INVITAR A ALGUIEN A JUGAR

1. Seleccionar el alias con quien se desea mantener partido.
2. Si desea seleccionar otro jugador, regresar al paso 1.
3. Si se desea enviar la lista y no se ha seleccionado algún alias ir al paso 8 .
4. Armar el buffer a enviar.
5. Enviar el buffer al servidor.
6. Recibir buffer del servidor.
7. Obtener el código contenido en el buffer, seleccionar el significado y presentarlo.
8. Actualizar opciones del menú.

4.3.5 RECIBIR INVITACIONES

1. Armar el buffer con código y alias del usuario.
2. Enviar el buffer al servidor.
3. Recibir el buffer del servidor.
4. Si se trata el buffer de un código de error, obtener el significado del código y presentarlo, ir al paso 6.
5. obtener el alias y nombre de cada invitación en el buffer y almacenarlos la tabla invitaciones.
6. Actualizar archivo "damas.ini" y opciones del menú.

4.3.6 ENVIAR RESPUESTA DE INVITACION

1. Mostrar invitaciones recibidas que son cargadas de la tabla de invitaciones.
2. Seleccionar la invitación a la que se desea aceptar y el color con el que se desea jugar y eliminarlo de la lista.
3. Añadir al buffer el alias seleccionado, si se desea seleccionar otro regresar al paso 2.
4. Enviar el buffer al servidor.
5. Recibir el buffer del servidor.
6. Obtener código de retorno del buffer, buscar su significado en la tabla de error_code y mostrarlo.
7. Si el código obtenido es 9910, actualizar tabla de invitaciones, eliminar las invitaciones aceptadas.

4.3.7 ENVIAR JUGADA

1. Obtener la jugada de la tabla ficha_out y de la de contrin y armar el buffer.
2. Enviar al servidor el buffer.
3. Recibir del servidor el buffer.
4. Obtener código del buffer.
5. Si se trata del código 9908 continuar al paso 8.
6. Si se trata del código 9910 eliminar la jugada de la tabla de fichas_out y de contrin.
7. Si se trata de error en comunicación ir al paso 9.
8. Regresar al paso 1 hasta que la tabla este vacía.
9. Actualizar opciones del menú.

4.3.8 RECIBIR JUGADA

1. Armar buffer con código de opción y alias.
2. Enviar el buffer al servidor.
3. Recibir del servidor el buffer. si el buffer esta vacío ir al paso 6.
4. Levantar información a las tablas de fichas_in y contrin.
5. Actualizar archivo de inicialización, damas.ini, y las opciones del menú.
6. Salir.

4.3.9 MATAR PARTIDOS

1. Armar buffer de recibir lista de partidos en los que el usuario se encuentra involucrado (11;alias;#).
2. Enviar al servidor el buffer.
3. Recibir el buffer del servidor.
4. Si el buffer recibido es del tipo error, ir al paso .
5. Armar lista (arreglo) con los alias recibidos del buffer.
6. Presentarlos en pantalla.
7. Seleccionar el partido a matar.
8. Añadirlo al buffer.
9. Si se desea añadir otro partido, ir al paso 7.
10. Enviar el buffer al servidor.
11. Recibir buffer del servidor.

12. Obtener código del buffer, buscar significado de la tabla y mostrarlo.

4.3.10 DESUSCRIBIRSE

1. Armar buffer conteniendo el código y el alias.
2. Enviar el buffer hacia el servidor.
3. Recibir buffer del servidor.
4. Obtener código del buffer y buscar su significado en la tabla de error_code y presentarlo.
5. Si el código es 9910, actualizar archivo.ini y las opciones del menú.

4.3.11 ALGORITMO GRABAR TABLERO

1. Abrir base de datos *damas.mdb*
2. Abrir las tablas: *fichas_in, fichas_out*
3. Borrar todos los datos con el alias del contrincante de la tabla *ficha_in*
4. Tomar primer registro de estructura *f_roja*
5. Si estado de la ficha *f_roja(indice).estado=1* entonces crear un registro con los datos de la ficha en la tabla *ficha_out*
6. Tomar siguiente registro de estructura *f_roja* e ir al paso 5 hasta terminar de leer la estructura
7. Tomar primer registro de estructura *f_azul*
8. Si estado de la ficha *f_azul(indice).estado=1* entonces crear un registro con los datos de la ficha en la tabla *ficha_out*
9. Tomar siguiente registro de estructura *f_azul* e ir al paso 5 hasta terminar de leer la estructura.

4.3.12 ALGORITMO CARGAR_TABLERO

1. Abrir base de datos *damas.mdb*
2. Abrir la tabla *fichas_in*
3. Inicializar las estructuras *f_roja y f_azul*
4. Tomar primer registro de tabla *fichas_in*
5. Si el registro tomado corresponde al contrincante deseado tomar estos datos y llenarlos en la estructura *f_roja o f_azul*
6. Posicionar en pantalla la ficha
7. Tomar siguiente registro de la tabla y repetir paso 5 hasta llegar al final de la misma

4.3.13 ALGORITMO VALIDAR_JUGADA

1. Verificar si corresponde turno al jugador, sino ir a paso 10
2. Validar la ficha que hizo la jugada, si es roja ir a paso 3 sino al paso 6
3. Obtener las posiciones inicial(xi,yi) y final(xf,yf) del movimiento de la ficha
4. Si el ABS(xf-xi)=1 y (yf-yi)=1 entonces mover la ficha a la posición final, actualizar la estructura *f_roja(indice actual)* y retornar.
5. Si el ABS(xf-xi)=2 y (yf-yi)=2 entonces verificar si ficha entre posición final e inicial es del contrincante, si lo es entonces ocultar ficha contrincante, actualizar la estructura *f_azul(ficha comida)*, actualizar estructura *f_roja* con posición final y retornar
6. Obtener las posiciones inicial(xi,yi) y final(xf,yf) del movimiento de la ficha
7. Si el ABS(xf-xi)=1 y (yf-yi)=1 entonces mover la ficha a la posición final, actualizar la estructura *f_azul(indice actual)* y retornar.
8. Si el ABS(xf-xi)=2 y (yf-yi)=2 entonces verificar si ficha entre posición final e inicial es del contrincante, si lo es entonces ocultar ficha contrincante, actualizar la estructura *f_roja(ficha comida)*, actualizar estructura *f_azul* con posición final y retornar

4.4 COMPROMISOS DEL DISEÑO DEL CLIENTE

Por la facilidad de programación, se ha tenido que utilizar tablas, lo que hace que estas a pesar de estar vacías ocupen espacio en disco, lo cual se hubiese evitado si se usaban archivos, los que solo existirían si estos conteniesen datos, pero por otra parte el uso de archivos introduciría complejidad en la programación en el momento de eliminar columnas de estos.

El hecho de utilizar un archivo de inicialización, damas.ini, hace que la información que este contiene pierda confiabilidad, ya que pueden ser editados y alterados fácilmente.

Permite que una persona se pueda cambiar en un momento dado de usuario sin necesidad de salir y volver a ejecutar el programa cliente.

Si un usuario desea conectarse a un servidor para el cual deberá realizar una llamada telefónica o conectarse mediante una tarjeta de red, se deberá configurar el software DISTINCT de acuerdo al servidor o enlace a usar.

4.5 DISEÑO DE INTERFACES DEL USUARIO

En el diseño de las interfaces con el usuario, se ha tratado de mantener el estandard establecido en WINDOWS.

Las opciones del Menú seran visualizadas conforme el usuario haya navegado por estas. Es decir si para realizar la ejecución de una opción se debe previamente haber ejecutado otra, la primera no podrá ser accesada por el usuario. Además se dispone de hot-keys para las opciones que más frecuentemente son usadas.

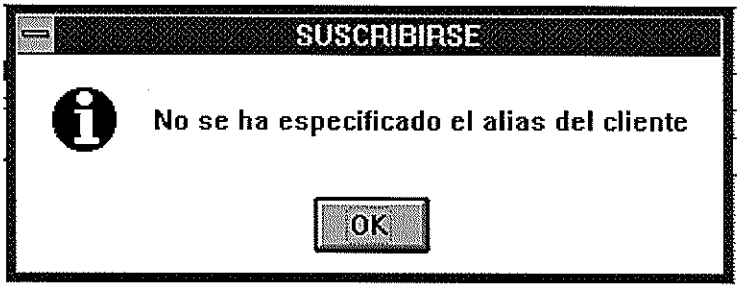
Se contará tambien con una Ayuda, tanto acerca de la realización de la aplicación así como del contenido de cada una de las opciones y campos de la aplicación.

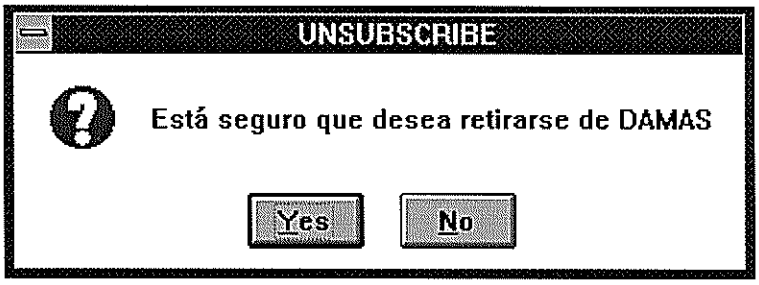
Se dispone de una barra de estado, que se visualizará en la parte inferior de la pantalla, en la que se mostrará una ayuda en línea del campo sobre el cual el cursor del mouse se encuentre.

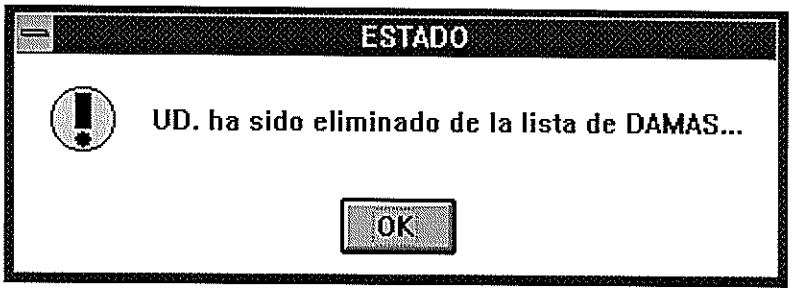
Las pantallas con que se disponen durante la aplicación tendrán un título descriptivo de esta. Tendrán deshabilitados los botones de maximización y minimización.

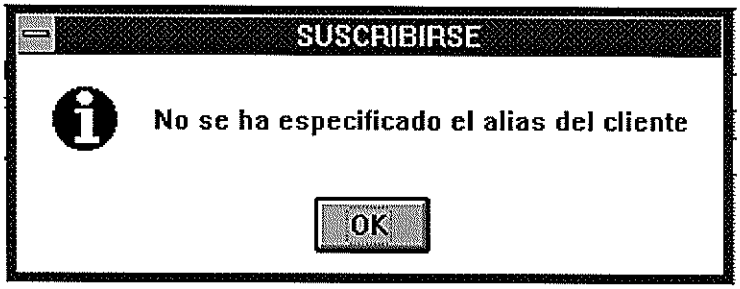
Las pantallas de indicación de estado, errores, poseerán: título de pantalla a la que corresponde el error obtenido, icono representativo de la gravedad del error causado y una breve descripción de este.

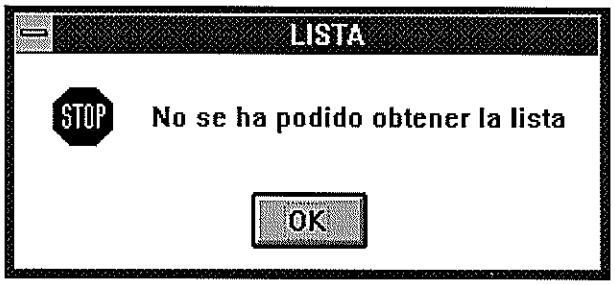
Las pantallas de cada opción, asi como de los mensajes al ser mostrados al usuario seran visualizados en el centro del monitor.

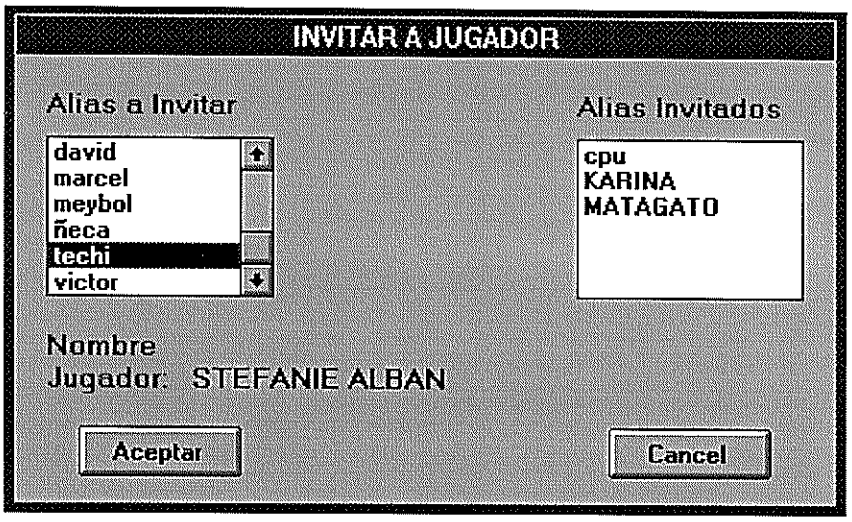





ESTADO
UD. ha sido eliminado de la lista de DAMAS...
OK


SUSCRIBIRSE
No se ha especificado el alias del cliente
OK


LISTA
STOP
No se ha podido obtener la lista
OK


INVITAR A JUGADOR
Alias a Invitar
david
marcel
meybol
ñeca
techi
victor
Alias Invitados
cpu
KARINA
MATAGATO
Nombre
Jugador: STEFANIE ALBAN
Aceptar
Cancel

5. MANUALES

5.1 MANUAL DEL USUARIO DE LA APLICACION CLIENTE

5.1.1 Instalación

Previo a la instalación, se deberá tener en cuenta que para obtener un buen rendimiento en la ejecución de la aplicación, se deberá contar con los siguientes requerimientos básicos:

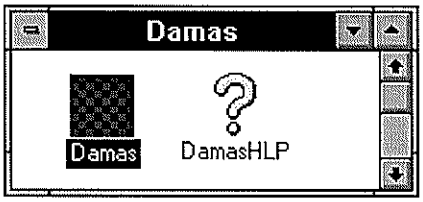
- Windows versión 3.0 o superior.
- Mouse
- Espacio disponible en disco mínimo 2M.

Se deberá ejecutar desde el File Manager el comando ***a:\instalar.exe***, o realizar doble click en el nombre del archivo.

Como resultado de esto, se le solicitara el nombre del directorio destino. Una vez ingresado este, presione continuar.

5.1.2 Como iniciarse en el Juego DAMAS?

Presione consecutivamente dos veces el botón izquierdo del Mouse sobre el icono DAMAS del grupo DAMAS.



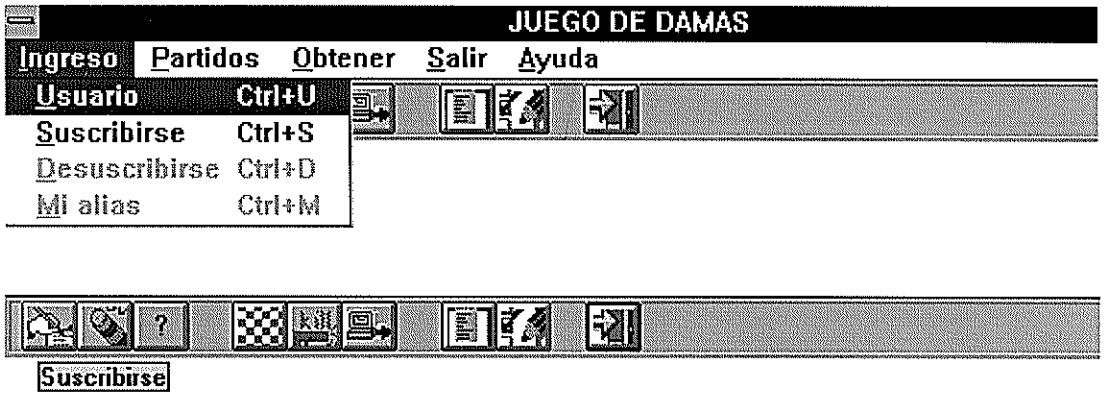
La pantalla que aparecerá es la que contiene el menu principal, como se muestra en la figura inferior.

La mayoría de las opiones del menú en este momento no se encuentran aún activas. Sólo las opciones de Suscribirse, Usuario, Salir y el Help se encuentran diponibles. Las emás opciones del menú estarán disponibles cuando Ud. haya ejecutado la opción Suscibirse o Usuario satisfactoriamente.

5.1.3 Suscribirse

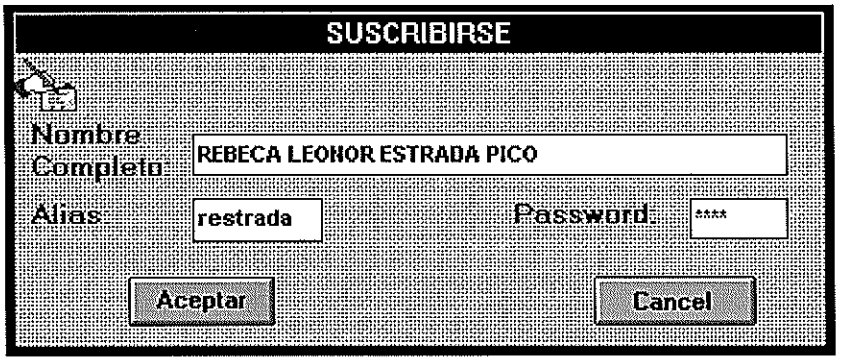
Si desea suscribirse al juego de DAMAS, Ud. cuenta con 3 métodos.

- Seleccionar del menú principal la opción Ingreso y de este Suscribirse.
- Presionando las teclas ***Ctrl+S***.
- Seleccionarlo de la barra de Herramientas (el primer botón a la izquierda).



Una vez selecionada la opcion de culquiera de los métodos antes descritos, ud. podrá ingresar sus datos como son: nombre, alias con el cual desea ser identificado en el juego, y la clave de acceso.

Cuando Ud. digite su clave, en lugar de esta, aparerá en ella una mascará (*) de manera que ninguna otra persona pueda verla. Asegurese de que la clave que ingrese sea fácil de recordar pero al mismo tiempo difícil para las demas personas de obtener.

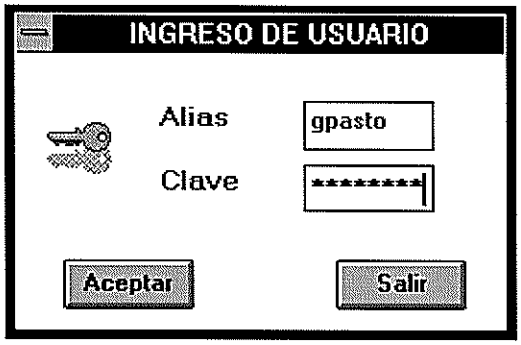


Cuando Ud. crea que los datos ingresados son corectos presione el botón de Aceptar, si ya no desea ingresar o suscribir un usuario presione el botón de Cancelar.

5.1.4 Conectarse con un usuario

Si Ud. ya se ncuentra suscrito al juego, podra usar la opción de Usuario del submenú de Ingreso del menú principal, o presionndo ls tecls ***Ctrl+U***. Para así poder realizar todas las tareas que considere necesarias. Al realizar esta opción correctamente, le permitirá visualizar las demás opciones del menú y poder hacer uso de ellas.

Una vez selecionada la opción, una ventana como la que se muestra a continuación le será presentada. Se deberán ingresar tanto el alias como la clave con la cual se suscribio, y presionar el botón de Aceptar.



Si los datos que ingreso son correctos, podrá Ud. hacer uso de las demás opciones del menú.

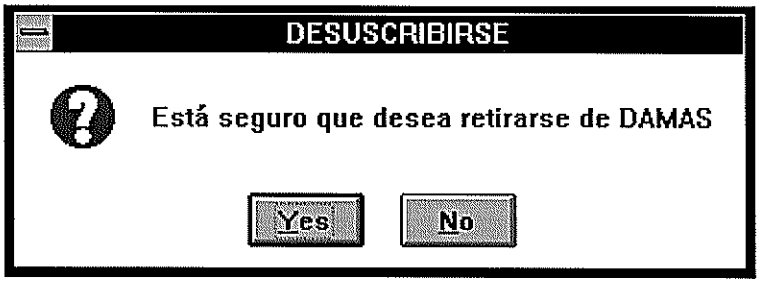
5.1.5 Desuscribirse

Esta opción se encuentra disponible una vez se haya ejecutdo la opión de Usuario o de Suscribirse satisfactoriamente.

Para ejecutar esta opcón se disponen de 3 métodos para ello.

- Seleccione del menú principal la opción Ingresar y de este Desuscribirse.
- Presionando las teclas ***Ctrl-D***.
- Realizando doble Clik sobre el 2do botón de la izquierda de la barra de Herramientas (al posicionar el mouse sobre esta aparecerá una leyenda indicando su función).

Si está seguro de que desea desuscribirse presione Yes o de lo contrario No en la pantalla que se muentra (que le será visualizada).



El resultado de la operación será mostrada.

5.1.6 Determinar el usuario en uso

Si por algún motivo Ud. olvidó con que usuario está trabajando, puede obtener información de el alias en uso de cualquiera de una de las siguientes mneras.

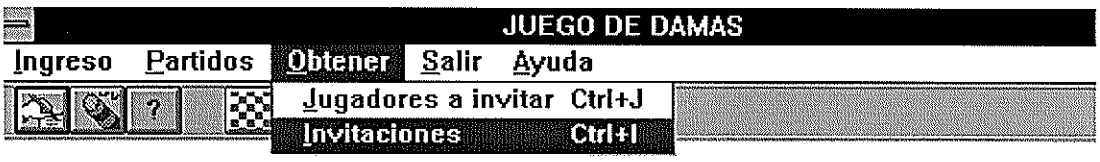
- Presionendo el 3er botón de la izquierda de la barra de Herramienta (donde al pocisionar al mouse sobre esta aparece la leyenda "Mi alias").
- Selecione del menú principal la opcion Ingreso y de este "Mi alias".
- Presione las teclas Ctrl+M.

Una vez selecionada cualquiera de las alternativas antes expuestas, aparecerá la siguiente ventana proporcionandole la información deseada.

5.1.7 Obtener y Contestar Invitaciones Recibidas

Para este fin, se disponen de tres métodos para ello.

- Seleccionando del menú principal la opción Obtener y de este Invitaciones.



- Realizar doble Clik sobre el 2do botón a la derecha de la barra de Herramientas.

- Presionando las teclas ***Ctrl+I***.

Si Ud. dispone de invitaciones que otros suscritores le hayan solicitado y que aún no hayan sido aceptadas, estos serán visualizados en una ventana como la que se muestra a continuación.

Luego seleccione de la lista de Contrincantes el oponente al cual se desea aceptar, el nombre del mismo podra ser visualizado en la parte inferior de dicha pantalla, realice doble Clik, lo cual hara que inmediatamente este este sea trasladado a la lista de Seleciconados. Seleccione también el color con que Ud. desea jugar antes de realizar el doble Clik.

Realice la operación anterior con cada uno de los oponentes a los cuales se les desea responder.

Si por cualquier motivo Ud. decide no responder a uno de los alias ya seleccionados antes de presionar Aceptar, estos pueden ser trasladados a la lista de Contrincante de la misma manera en la que fueron colocados en la lista de Seleccionados.

Si desea aceptar los oponentes ya seleccionados presione Aceptar.

Si desea regresar al menu principal presione Cancelar.

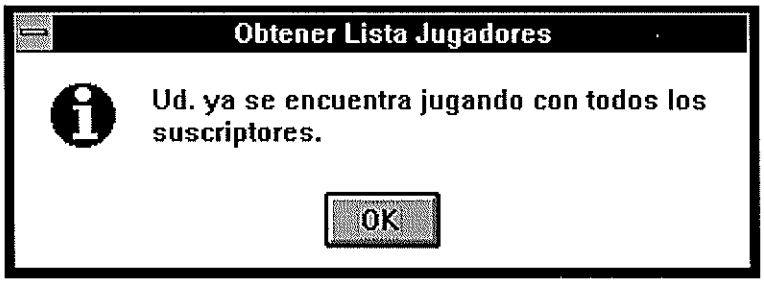
5.1.8 Invitar a otro Suscritor a mantener partido

Se disponen de tres métodos para ello.

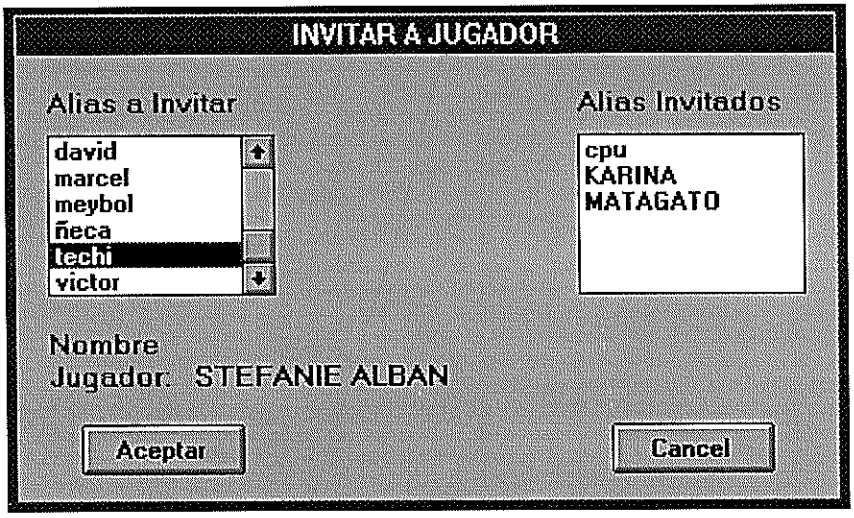
- Seleccionando del menú principal la opción Obtener y de este Jugadores a Invitar.
- Realizar doble Clik sobre el 2do botón a la derecha de la barra de Herramientas.

- Presionando las teclas ***Ctrl+J***.

Si Ud. ya se encuentra jugando con todos los suscritores, le será preentada una pantalla como se muestra en l figura inferior.



Si aún no se encuentra jugndo con todos os suscritores, Ud. podrá seleccionar alguno de ellos de lapantalla que le será mostrada.



Para seleccionar un contrincante al que se desea invitar, realice doble click en e alias que desea invitar de la lista de Alias a Invitar, este será trasladado a la lista de Alias Invitados.

Si desea elliminar alguno de los usuarios ya selecionados, realice la operación inversa a la de selecionar, lo que hará que dicho usuario sea trasladado nuevamente a la lista de Alias a Invitar.

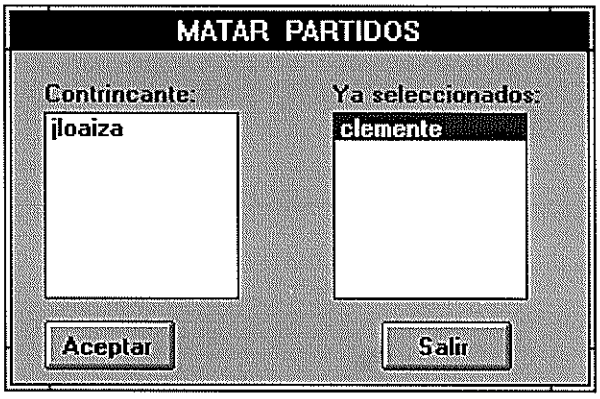
Cuando Ud. decida que ya han sido seleccionados todos los alias a quien desea invitar, presione Aceptar. Si desea anular o abandonar la operación presione Cancel.

5.1.9 Matar Partidos

Seleccione del menú principal la opción Juego y de este Matar Partidos o presione las teclas ***Ctrl-K,*** o realice doble Clik sobre el botón de la barra de herramientas que dice "kill".

Si Ud. posee algun partido con otro suscritor, le será visulizada la pantalla siguiente. Seleccione de la lista contrincante aquel con el cual se desea matar el partido realizando doble Clik sobre el alias de la lista de Contrincante y luego presione el botón Matar.

Si por cualquier motivo desea ya no matar algún partido, realice la operación anterior en orden inverso, es decir doble Clik sobre el alias de la list de Ya seleccionados, lo quehará que este regrese nuevamente a la lista de Contrincante.



Una vez que ya haya seleccionado todos los partidos que desea eliminar, presione el botón Aceptar. Si desea anular la operación y abandonar la ejecución de la opción presione Salir.

5.1.10 Enviar Partidos

Seleccione del menú principal la opción Juego y de este Enviar Partidos o presione ***Ctrl-E,*** o de la barra de Herramientas el 6to botón del lado izquierdo (cuya leyenda es "Enviar Partidos").

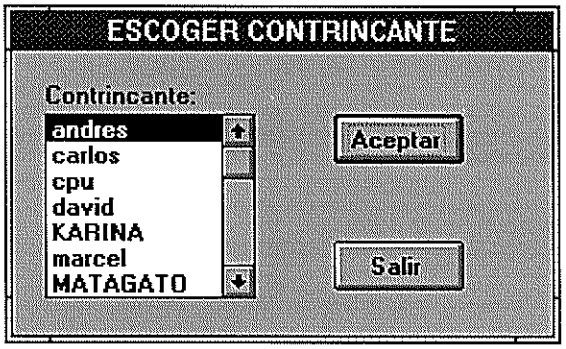
5.1.11 Mover Ficha de Jugada

Si desea realizar un movimiento de alguna jugada o partido que Ud. tenga, seleccione cualquiera de las siguientes alternativas

- Seleccione del menú principal la opción Partidos y de este Obtener.

- Presione las teclas ***Ctrl+O***.
- Selecione de la barra de Herramientas el botón cuyo dibujo es un tablero (4to a la izquierda)

Seleccione el partido del que desee mover ficha de la lista de contrincantes y presione Aceptar.



Si desea regresar al menú principal, presione Salir.

Para realizar una jugada tiene que posicionar el mouse sobre la ficha(de su color) que desea mover y hacer un click. Entonces un recuadro aparecerá en la ficha escogida. Ahora debe mover el mouse a la posición donde desea poner la ficha y hacer nuevamente un click del mouse.

Si la jugada es lícita la ficha se moverá a la posición deseada y el recueadro de la ficha escogida desaparecerá , si la jugada no es válida entonces aparecerá un mensaje indicándole lo sucedido.

Si su jugada conlleva a comer varias fichas contrincante entonces se tiene que hacer esto ficha por ficha, esto es, primero seleccionará la ficha a mover haciendo un click del mouse luego seleccionará la casilla del tablero (con otro click de mouse) donde se moverá la ficha para comer a una de las contrincante luego se moverá a la casilla correspondiente para comer la siguiente ficha y así sucesivamente.

5.1.12 Salir

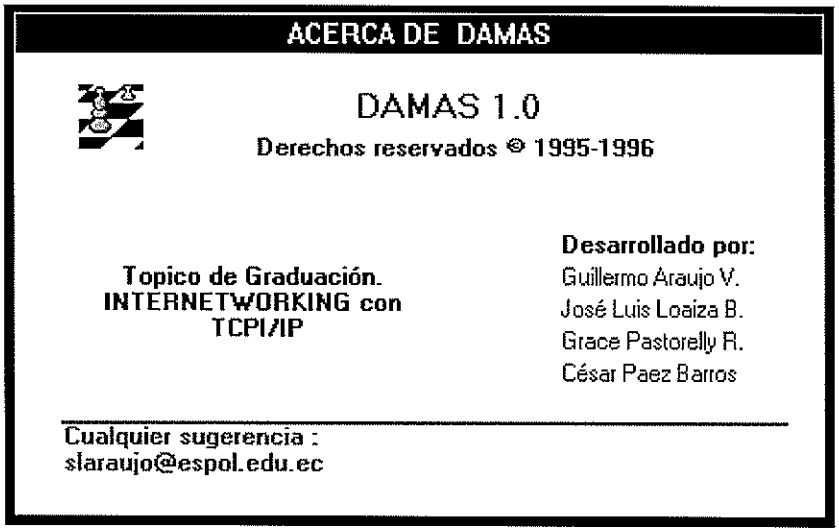
Seleccione del menú principal la opción Salir o presionando ***Alt-S,*** o de la barar de Herramientas el 1er botón del lado derecho.

5.1.13 Contenido de la Ayuda

Presione la tecla ***F1*** o seleccione del menú principal la opción Ayuda y de este Contenido.

5.1.14 Acerca de

Seleccione del menú principal la opción Ayuda y de este Acerca de.

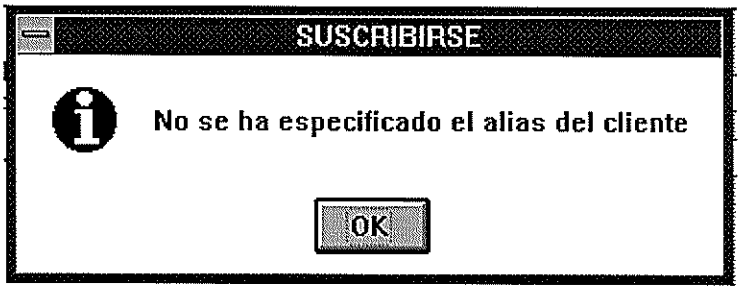


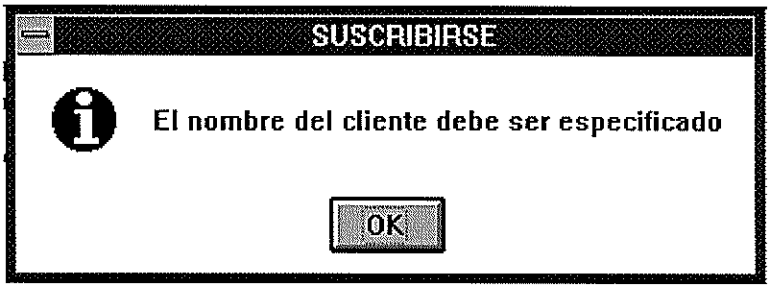
5.1.15 Tipos de Errores

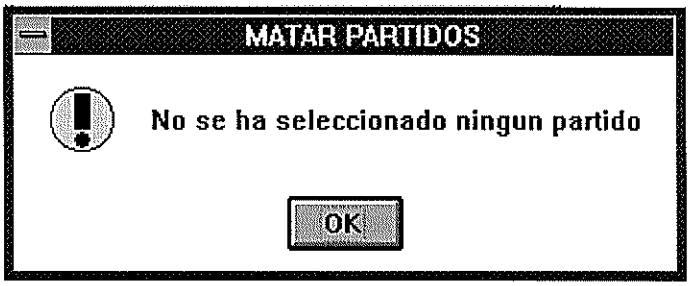
Los errores posibles durante la ejecución de la aplicación son:

de comunicación:

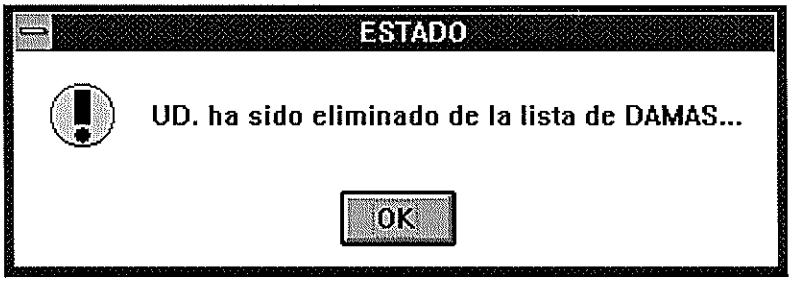
de validación de datos:

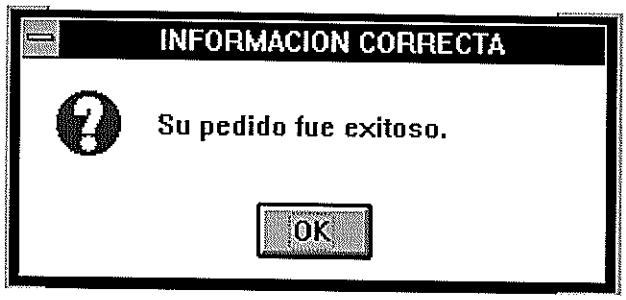






de estado o respuesta:




INFORMACION CORRECTA
Su pedido fue exitoso.
OK

5.2 MANUAL DEL ADMINISTRADOR DE LA APLICACION SERVIDORA

5.2.1 Instalación

Para la instalación del servidor, el administrador del centro de cómputo debe seguir los siguientes pasos distribuídos en dos etapas:

PRIMERA ETAPA (Preparación del puerto)

Realizarlo en el modo mantenimiento

1. Configurar uno de los puertos para establecer el enlace con el software cliente. Por ejemplo configurar el puerto COM2 a una velocidad de 9600 bps mediante el administrador del shell **sysadmsh--systems--hardware--serial/parallel port.**

2. Reconstruir el nuevo kernel, las variables de ambiente y que se levante por default.

3. En el modo de mantenimiento configurar el puerto anterior para que trabaje en red con el protocolo tcp a través del comando **netconfig**

4. Seleccione lo siguiente:

   Add a Chain

   sco_tcp

   interface SLIP **sl0** o **sl1.**

   tty2a o **tty1a** (para COM2 o COM1 según sea el caso)

   9600 bps

   Dirección de la interface local donde estará el servidor, ejm., 192.192.192.2

   Dirección de la interface destino, es decir de donde proviene el cliente, ejm:

   192.192.192.3

5. Reconstruir el nuevo kernel, sus variables de ambiente y que se levante por default.

SEGUNDA ETAPA (Instalación del software servidor)

1. Crear el directorio ***/home/fie/estud/jloaiza*** para almacenar todos los archivos del servidor .

   Si desea almacenar el servidor del juego de damas en otro directorio como por ejemplo en el directorio /damas, entonces debe crear ese directorio .Además editar el archivo *damas.c* en el que se encuentran las variables globales *players, inivita, damas,*las cuales contienen los path de acceso, las mismas que deben ser cambiadas por el nuevo path de acceso. Ejemplo:

   Antes:
   players = /home/fie/estud/jloaiza/players.txt
   invita = /home/fie/estud/jloaiza/invita.txt
   damas = /home/fie/estud/jloaiza/damas.txt

   Ahora:
   players = /damas/players.txt
   invita = /damas/invita.txt
   damas = /damas/damas.txt

   Finalmente, compilar el archivo damas.c para que el nuevo cambio tome efecto, para esto en el shell del unix escriba el siguiente comando.

   cc damas.c .o damas.exe -lsocket

2. Grabar los siguientes archivos en el directorio que creó en el paso anterior
   damas.exe
   players.txt
   invita.txt
   damas.txt

3. Levantar el software damas.exe indicando la dirección IP y el puerto que usará para comunicarse con el software cliente (Lo puede hacer en modo backgound si lo desea). Por ejemplo con el siguiente comando lo haremos con la dirección 192.192.192.2 y el puerto 1995 (Recuerde que el software cliente enviará sus datos a la dirección y al puerto que aqui se indican).

   ./damas.exe 192.192.192.2 1995

5.2.2 Administración del software servidor de damas

Realmente la tarea que tendrá el administrador del sistema para controlar y mantener el servidor de damas será mínima, ya que todas las transacciones se las realiza de manera automática con el cliente. Por ejemplo, cuando el cliente hace un requerimiento de suscripción, el servidor obtiene el string de datos enviado por el cliente, realiza la suscripción y le devuelve un mensaje al cliente para indicarle que su pedido fue ejecutado.

Entre los puntos que tendrá el administrador que realizar se encuentran los siguientes.

1. Verificar la correcta configuración del puerto de acuerdo a los pasos indicados anteriormente.

2. Levantar el servidor indicando la dirección IP y el puerto que va a ser utilizado por el mismo.

3. Informar a los usuarios clientes del puerto que deberán utilizar para conectarse con el servidor de damas, y sobre todo , tener presente que ese puerto debe ser utilizado única y exclusivamente para brindar ese servicio, evitando asi conflictos con otros servicios.

4. Tomar en cuenta que el servidor utiliza un número de archivos temporales para ejecutar ciertas transacciones con los clientes , los mismos que son borrados automáticamente por el servidor una vez que haya terminado su transacción con aquel cliente. Estos archivos temporales toman diferntes nombres de acuerdo al cliente con el que estén trabajando.

5. Si por alguna razón, los archivos temporales no se están borrando, entonces el administrador deberá hacer limpieza con determinada frecuencia, para esto, primero debe bajar el servidor , luego verificar que solamente deben existir los siguientes archivos:

   damas.exe
   players.txt
   invita.txt
   damas.txt

   El resto tendrá que ser borrado.

6. Todos los archivos .txt están conformados por líneas, las mismas que contienen un determinado número de campos, los cuales a su vez, están separados por el delimitador de punto y coma **(;)**.

7. Las líneas del archivo players.txt contienen 5 campos, estos son:
alias;nombre;password;partidos ganados;partidos perdidos
En este se encuentran todos los jugadores que se han suscrito al clud de juego de damas.

8. Las líneas del archivo invita.txt contienen los siguientes campos:
alias del que invita ;alias del invitado;respuesta
el campo *respuesta* será V o F dependiendo de si ha sido aceptada o no la
invitación.

9. Las líneas del archivo damas.txt contienen el alias del jugador al que le tocaría jugar , luego el alias del jugador que estaría en espera, el color de quien le toca el turno y finalmente las fichas y sus posiciones de acuerdo a un cierto formato.

Solamente en el caso de que uno de los jugadores decide matar un partido, aparecerá una línea conteniendo los dos alias y un código (9905), el mismo que sirve para informarle al otro jugador que su partido ha sido muerto por su contrincante. Una vez informado sobre este particular, la línea desaparece del archivo.

10. CAMBIO DE PASSWORD.- Si por algún motivo uno de los jugadores desea cambiar el password, entonces el administrador deberá ir al archivo players.txt y cambiar el tercer campo por el nuevo password en la línea que corresponde al usuario que lo solicita.

6. LISTADO

6.1 LISTADOS PROGRAMA SERVIDOR

ARCHIVO damas.c

```
#include <sys/types.h>
#include <sys/socket.h>
#include <sys/errno.h>
#include <netinet/in.h>
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <malloc.h>
#include <memory.h>
#include <sys/stat.h>
#include <unistd.h>

/*************************************************************/
# define LONG 300
# define FIN_BUFFER '#'
# define DELIMIT ';'
char *STR_FIN_BUFFER = "#";
char *STR_DELIMIT = ";";
char *STR_COLOR_INI = "00";
char *STR_COLOR_FIN = "10";

/****************** VARIABLES GLOBALES ********************************/
char * players = "/home/fie/estud/jloaiza/play2.txt";
char * damas = "/home/fie/estud/jloaiza/damas.txt";
char * invitados = "/home/fie/estud/jloaiza/invita.txt";
char * tmp_p = "/home/fie/estud/jloaiza/tmp/tmp_p";
char * tmp_d = "/home/fie/estud/jloaiza/tmp/tmp_d";
char * tmp_i = "/home/fie/estud/jloaiza/tmp/tmp_i";

/*****************************************************************************/
extern int errno;
extern char *sys_errlist[];

/*************************************************************/
int NULO= 1000;
int BIEN= 1;

/*************************************************************/
void limpiar(char *, int);
void bloquear(char *);
int desbloquear(char *);
int cortar_campo(char *, char *, char );
int leer_sock (int, char *);
//int registrar (int, char *);
int cliente ( int );
void registrar (int , char * );
int write_cli (int, char * );
int no_consta(char* , char *);
void env_list(int, char *);
```

```
int elimina_alias_play( char *);
void elimina_alias_inv(char *);
int elimina_alias_damas(char *);
void desuscribir(int ,char *);
void logon (int, char * );
void reg_inv(int, char *);
void env_inv(int, char*);
void insertar_linea_damas(char *,char *,char *);
void reg_resp_inv(int , char *);
void recv_jugada(int,char *);
void cambia_puntos(char *,int);
void elimina_alias_invitados(char *,char *);
void elimina_partido_damas(char *,char *);
void matar_partido(int,char *);
void partidos_actuales (int, char *);
void env_jugada (int, char *);


/*************************************************************/
```

6.1.1 void limpiar(char * cadena,int N)

```
{
  int i=0;

  for(i=0;i<N;i++)
  {
    cadena[i]='\0';
  }
}

/*****************************************************************************/
```

6.1.2 void bloquear(char * arch)

```
{
  FILE *fp;
  while(1)
  {
    if( (fp =fopen(arch,"r")) == NULL )
    {
      fp = fopen(arch,"w");
      fclose(fp);
      printf("El cliente acaba de bloquear el archivo: %s\n",arch);
      break;
    }
    else
    {
      fclose(fp);
      printf("El archivo : %s, esta bloqueado\n",arch);
    }
  }
}
```

```
/*************************************************************************/
```

6.1.3 int desbloquear(char * arch)

```
{
 char  *cadena;
 int N=80;

 cadena = (char *) malloc(N);
 if( cadena== NULL)
 {
   return(NULO);
 }
 limpiar(cadena,N);
 sprintf(cadena,"rm %s",arch);
 system(cadena);
 free(cadena);
 return(BIEN);
}

/*************************************************************************/
```

6.1.4 int cortar_campo(char * campo,char * cadena,char c)

```
{
  char *cadena1;
  int i=0, len, ban;

  len = strlen(cadena) + 1;
  cadena1 = (char *) malloc(len);
  if(cadena1==NULL)
  {
    return (NULO);
  }
  limpiar(cadena1,len);
  cadena1 = cadena;
  ban=1;
  while(ban)
  {
   if(cadena1[i] == c)
   {
     ban=0;
     break;
   }
   campo[i]=cadena1[i];
   campo[i+1]= '\0';
   i++;
  }
  i = len = strlen (campo) + 1;
  while(cadena[i] != '\0')
  {
    cadena[i- len ] = cadena1[i];
    i=i+1;
  }
  cadena[i-len] = '\0';
  free(cadena1);
  return(BIEN);
 }
```

```
/*************************************************************/
```

6.1.5 int leer_sock (int ssock,char * cadena)

```
{
  int rc,N=3;
  char *ch;

  ch =(char *) malloc(N);
  memset(ch,'\0',N);
  while ( strcmp (ch, "#") != 0 )
  {
    limpiar(ch,N);
    rc = read (ssock, ch, 2);
    printf("EL CH ES: %s\n",ch);
    if ( rc == -1 )
    {
      free(ch);
      return (NULO);
    }
    strcat (cadena, ch);
    if(ch[0]=='#' || ch[1] == '#')
       break;
  }
  free(ch);
  return(BIEN);
}

/*************************************************************************/
```

6.1.6 int cliente(int ssock)

```
{
  char * buffer, * codigo;
  int LIMITE=300,N=3,code;

  buffer = (char *) malloc(LIMITE);
  codigo = (char *) malloc(N);
  if(buffer==NULL || codigo==NULL)
  {
    free(buffer);
    free(codigo);
    return (NULO);
  }
  limpiar(buffer,LIMITE);
  if( leer_sock(ssock, buffer) >= NULO)
  {
    free(buffer);
    free(codigo);
    return(NULO);
  }
  if (buffer != NULL)
  {
    limpiar(codigo,N);
    if( cortar_campo(codigo,buffer,DELIMIT) >=NULO )
    {
      free(buffer);
      free(codigo);
```

```
    return(NULO);
  }
  code = atoi(codigo);
  switch (code)
   {
    case 13:
        registrar (ssock, buffer);
        break;
    case 1:
        env_list (ssock, buffer);
        break;
    case 2:
        reg_inv (ssock, buffer);
        break;
    case 3:
        env_inv (ssock, buffer);
        break;
    case 4:
        reg_resp_inv (ssock, buffer);
        break;
    case 5:
        recv_jugada (ssock, buffer);
        break;
    case 6:
        env_jugada (ssock, buffer);
        break;
    case 7:
        matar_partido (ssock, buffer);
        break;
    case 10:
        desuscribir (ssock, buffer);
        break;
    case 12:
        logon (ssock, buffer);
        break;
    case 11:
        partidos_actuales (ssock, buffer);
        break;
   }/*fin switch*/
}/*fin if . Entro si se pudo leer el socket */
 free(buffer);
 free(codigo);
return(BIEN);
}/*fin de cliente*/

/*********************************************************/
```

6.1.7 main (int argc,char * argv[])

```
{
 char * host_ip;
 int port, msock, ssock, pid, length, N=20;
 struct sockaddr_in  cli_addr, svr_addr;

 host_ip=(char *)malloc(N);
 if(host_ip==NULL)
 {
   free(host_ip);
```

```
  exit(0);
}
limpiar(host_ip,N);
if ( argc > 1 )
{
  host_ip = argv[1];
  port = atoi (argv[2]);
  if ( (msock = socket(AF_INET, SOCK_STREAM, 0) ) > 0 )
  {
    bzero ((char *) & svr_addr, sizeof (svr_addr));
    svr_addr.sin_family = AF_INET;
    svr_addr.sin_addr.s_addr = htonl (INADDR_ANY);
    svr_addr.sin_port = htons (port);
    if ( bind(msock, (struct sockaddr_in *) &svr_addr, sizeof (svr_addr)) <0 )
    {
      printf ("Error al hacer al BIND\n");
      printf("El error es %s\n", sys_errlist[errno] );
    }
    else
    {
      listen (msock, 5);
      printf("Luego del listen\n\n");

      for ( ; ; )
      {
        length = sizeof (cli_addr);
        ssock = accept (msock, (struct sockaddr_in *) &cli_addr, &length );
        if ( ssock > 0 )
        {
          if (  (pid = fork() ) > 0 )
          {
            printf("Estoy en el proceso padre y el PID es: %d\n\n",pid);
          }
          else
          {

            printf("Estoy en el proceso hijo y el PID es: %d\n\n",pid);
            if( cliente (ssock) >=NULO)
            {
              printf("Hubo problemas con el proceso cliente\n\n");
            }
            close (ssock);
            exit(0);
          }
        } /*fin del if ssock > 0 luego de hacer el accept*/
        else
          printf ("Error al realizar ACEPT\n");
      }/*fin del for*/
    }/*fin del else*/
  }
  else
    printf ("El servidor no pudo asignar el socket\n");
}
else    /* viene el if argc > 1 */
{
  printf ("No se ha especificado el puerto y la direccion ip\n");
}
}
```

```
/*************************************************************************/
```

6.1.8 void registrar (int sock, char * buff)

```
{
 int N = 50, N1 = 95, N2 = 200, N3 = 50;
 char * alias, * name, * passwd, * cadena, * tmp, * tmp1;
 struct stat est_arch;
 FILE * fp;

 alias = (char *)malloc(N);
 passwd = (char *)malloc(N);
 name = (char *)malloc(N1);
 cadena = (char *)malloc(N2);
 tmp = (char *)malloc(N3);
 tmp1 = (char *)malloc(N3);

 tmp = "/home/fie/estud/jloaiza/play2.txt";
 alias = strtok(buff, STR_DELIMIT);
 name = strtok(NULL, STR_DELIMIT);
 passwd = strtok(NULL,STR_DELIMIT);
 sprintf (tmp1, "/home/fie/estud/jloaiza/%s", alias);
 sprintf (cadena, "grep \"^%s;\" %s > %s", alias, tmp, tmp1);
 if (system (cadena)!= -1 )
 {
  printf ("1er system\n");
  if( ! stat (alias, &est_arch))
  {
   printf ("en 1er estat\n");
   if (est_arch.st_size > 0)
   {
    write_cli (sock, "9901;#");
   }//finif
   else
   {
    sprintf (cadena, "grep \";%s;\" %s > %s", name, tmp, tmp1);
    if (system(cadena) != -1)
    {
     if (!stat(alias, &est_arch))
     {
      if (est_arch.st_size > 0)
      {
       write_cli (sock, "9902;#");
      }//fin 2do est_arch.st_size
      else
      {
       if ( (fp = fopen (tmp, "a+")) )
       {
        fseek (fp, 0L, SEEK_END);
        sprintf (cadena, "%s;%s;%s;00;00\n", alias, name, passwd);
        fwrite (cadena, strlen (cadena), 1, fp);
        fclose (fp);
        write_cli (sock, "9900;#");
       }
       else
        write_cli (sock, "9911;#");
      }//ya se escribe en arch.
     }//ofin 2do stat
```

```
     }//fin
    }//fin 1er est_arch.st_size
   }//fin de 1er stat
   sprintf (cadena, "rm %s", tmp1);
   system (cadena);
 }//fin system
 else
  write_cli (sock, "9911;#");
 free (cadena);
 free (alias);
 free (passwd);
 free (tmp);
 free (name);
 free (tmp1);
}//fin fun

/*************************************************************************/
```

6.1.9 int write_cli (int ssock,char * cadena)

```
{
  int rc;
  rc = write (ssock, cadena, strlen(cadena));
  if (rc == -1)
  {
    printf("No se puede escribir en el socket");
    return rc;
  }
  return rc;
}

/*************************************************************************/
```

6.1.10 int no_consta(char * alias1,char * lista_no)

```
{
  if(strstr(lista_no,alias1) == NULL)
  {
    return(1);
  }
  return(0);
}
```

```
/*************************************************************************/
```

6.1.11 void env_list(int ssock,char * buffer)

```
{
 int N =80,N2=30,N3=40,N4=4000;
 char * lista_no, * lista_si, * cadena;
 char * alias1, * alias2, *alias_sol, * nombre1;
 FILE * fp;

 alias_sol=(char *)malloc(N2);
 alias1=(char *)malloc(N2);
 alias2=(char *)malloc(N2);
 nombre1=(char *)malloc(N3);
 lista_no=(char *)malloc(N4);
 lista_si=(char *)malloc(N4);
 cadena = (char *) malloc(N);

 alias_sol=strtok(buffer,STR_DELIMIT);
 if( (fp = fopen (invitados,"r")) != NULL)
 {
   fseek(fp,0,0);
   limpiar(lista_no,N4);
   strcat(lista_no,alias_sol);
   strcat(lista_no,STR_DELIMIT);
   limpiar(cadena,N);
   while( fgets(cadena,N-1,fp) != NULL)
   {
     alias1=strtok(cadena,STR_DELIMIT);
     alias2=strtok(NULL,STR_DELIMIT);
     if(strcmp(alias1 , alias_sol) == 0)
     {
       strcat(lista_no,alias2);
       strcat(lista_no,STR_DELIMIT);
     }
     else if(strcmp(alias2 , alias_sol) == 0)
     {
       strcat(lista_no,alias1);
       strcat(lista_no,STR_DELIMIT);
     }
     limpiar(cadena,N);
   }
   fclose(fp);
   if( (fp = fopen(players,"r")) != NULL)
   {
     fseek(fp,0,0);
     limpiar(lista_si,N4);
     limpiar(cadena,N);
     while( fgets(cadena,N-1,fp) != NULL)
     {
       alias1=strtok(cadena,STR_DELIMIT);
       nombre1=strtok(NULL,STR_DELIMIT);
       if( (alias1 != alias_sol) && no_consta(alias1,lista_no) )
       {
         strcat(lista_si,alias1);
         strcat(lista_si,STR_DELIMIT);
         strcat(lista_si,nombre1);
         strcat(lista_si,STR_DELIMIT);
```

```
        }
        limpiar(cadena,N);
      }
      fclose(fp);
      strcat(lista_si,STR_FIN_BUFFER);
      if( lista_si[0] == FIN_BUFFER)
        strcpy(lista_si,"9913;#");
      write_cli(ssock,lista_si);
    }
    else
    {
      strcpy(lista_si,"9911;#");
      write_cli(ssock,lista_si);
    }
  }
  else
  {
      strcpy(lista_si,"9911;#");
      write_cli(ssock,lista_si);
  }
  free(alias_sol);
  free(alias1);
  free(alias2);
  free(nombre1);
  free(lista_no);
  free(lista_si);
  free(cadena);
}

/*************************************************************************/
```

6.1.12 int elimina_alias_play(char * alias)

```
{
 FILE *ptr;
 int N1=15,N2=150;
 char comando[150],alias1[15];

 limpiar(comando,N2);
 sprintf(comando,"grep '%s;' %s > tem1",alias,players);
 system(comando);
 ptr = fopen("tem1","r");
 if(ptr != NULL)
 {
   limpiar(comando,N2);
   if(fgets(comando,150,ptr) != NULL)
   {
     strcpy(alias1,strtok(comando,STR_DELIMIT));
     if(strcmp(alias1,alias)==0)
     {
       fclose(ptr);
       sprintf(comando,"grep -v '%s;' %s > tem1",alias,players);
       system(comando);
       sprintf(comando,"cp tem1 %s",players);
       system(comando);
       return(1);
     }
   }
```

```
 }
 fclose(ptr);
 return(0);
}

/*************************************************************/
```

6.1.13 void elimina_alias_inv(char * alias)

```
{
 char comando[100];

 sprintf(comando,"grep -v '%s;' %s > tem1",alias,invitados);
 system(comando);
 sprintf(comando,"cp tem1 %s",invitados);
 system(comando);
}

/*************************************************************/
```

6.1.14 int elimina_alias_damas(char * alias)

```
{
 FILE *ptr,*qtr;
 char comando[100], alias1[15], alias2[15],color[10];
 char cadena[250];
 int len,b = 1;

 /* copia en el archivo tem1 todas las ocurrencias del alias en damas */
 sprintf(comando,"grep '%s;' %s > tem1",alias,damas);
 system(comando);

 /* copia en tem2 todas las lineas que no tengan alias en damas */
 sprintf(comando,"grep -v '%s;' %s > tem2",alias,damas);
 system(comando);
 sprintf(comando,"cp tem2 %s",damas);
 system(comando);
 ptr = fopen(damas,"a");
 if(ptr == NULL)
 {
   b = 0;
 }
 qtr = fopen("tem1","r");
 if(qtr == NULL)
 {
   b = 0;
 }
 rewind(qtr);
 /* recorre el archivo temporal y agrega la linea a damas */
 while(b)
 {
  if (fgets(cadena,200,qtr) == NULL) break;
  strcpy(color,strtok(cadena,STR_DELIMIT));
  strcpy(alias1,strtok(NULL,STR_DELIMIT));
  strcpy(alias2,strtok(NULL,STR_DELIMIT));
  if(strcmp(alias,alias1) == 0)
  {
    sprintf(comando,"00;%s;%s;9905\n",alias2,alias1);
    len = strlen(comando);
```

```
   fseek(ptr,0,SEEK_END);
   fwrite(comando,len,1,ptr);
 }
 else if(strcmp(alias,alias2) == 0)
 {
   sprintf(comando,"00;%s;%s;9905\n",alias1,alias2);
   len = strlen(comando);
   fseek(ptr,0,SEEK_END);
   fwrite(comando,len,1,ptr);
 }
}
fclose(ptr);
fclose(qtr);
return 0;
}


/*************************************************************************/
```

6.1.15 void desuscribir(int ssock,char * buffer)

```
{
 char alias[10];
 int l ;

 limpiar(alias,10);
 strcpy(alias , strtok(buffer,STR_DELIMIT));
 l = elimina_alias_play(alias);
 if(l)
 {
    elimina_alias_inv(alias);
    printf("Todo salio bien\n");
    write_cli(ssock,"9910;#");
 }
 else
 {
   write_cli(ssock,"9911;#");
 }
}

/*****************REALIZAR VERIFICACION DE ALIAS Y PASSWD*******************/
```

6.1.16 void logon (int ssock, char * buffer)

```
{
  char * alias, * passwd, * tmp, * tmp1, * cadena;
  struct stat est_arch;
  int N = 30, N1 = 200, N2 = 200;

  alias = (char *)malloc(N);
  passwd = (char *)malloc(N);
  tmp = (char *)malloc(N1);
  tmp1 = (char *)malloc(N1);
  cadena = (char *)malloc(N2);

  tmp = "/home/fie/estud/jloaiza/play2.txt";
  alias = strtok (buffer, STR_DELIMIT);
  passwd = strtok (NULL, STR_DELIMIT);
  sprintf (tmp1, "/home/fie/estud/jloaiza/%s", alias);
```

```
  sprintf (cadena, "grep \"^%s;\" %s|cut -f3 -d\';\'|grep \"^%s$\" > %s", alias, tmp, passwd, tmp1);
  if (system(cadena) != -1)
  {
   if ( !stat(tmp1, &est_arch))
   {
    if (est_arch.st_size > 0)
    {
     cadena = "9910;#";
     write_cli (ssock, cadena);
    }
    else
    {
     cadena = "9909;#";
     write_cli (ssock, cadena);
    }
   }//fin de if stat
   else
   {
    cadena = "9909;#";
    write_cli (ssock, cadena);
   }
  }//fin if system
  else
  {
   cadena = "9909;#";
   write_cli (ssock, cadena);
  }
  sprintf (cadena, "rm %s", tmp1);
  system (cadena);
  free (cadena);
  free (tmp);
  free (tmp1);
  free (alias);
  free (passwd);
}//fin logon

/*****************************************************************************/
```

6.1.17 void reg_inv(int ssock, char * buffer)

```
{
  int j, ban=0;
  char alias_sol[10], alias_inv[10], cadena[120];
  struct stat est_arch;

  strcpy(alias_sol , strtok (buffer, STR_DELIMIT));
  j = strlen (alias_sol) + 1;
  buffer = buffer + j;
  while ( strcmp(buffer,STR_FIN_BUFFER) != 0)
  {
   strcpy (alias_inv, strtok(buffer,STR_DELIMIT));
   j = strlen (alias_inv) + 1;
   buffer = buffer + j;
   sprintf(cadena,"grep \"^%s%s%s%s\" %s >
tmp_%s",alias_sol,STR_DELIMIT,alias_inv,STR_DELIMIT,invitados,alias_sol);
   if (system (cadena)!=-1)
   {
    sprintf (cadena, "tmp_%s", alias_sol);
```

```
    if (stat(cadena,&est_arch)==0)
    {
     if (est_arch.st_size == 0 )
     {
      sprintf(cadena,"grep \"^%s%s%s%s\" %s >
tmp_%s",alias_inv,STR_DELIMIT,alias_sol,STR_DELIMIT,invitados,alias_sol);
       if( system (cadena)!=-1)
       {
        sprintf (cadena, "tmp_%s", alias_sol);
        if (stat(cadena,&est_arch)==0)
        {
         if (est_arch.st_size == 0)
         {
          sprintf (cadena, "/bin/echo -n \"%s%s%s%sF\n\" >> %s ", alias_sol, STR_DELIMIT,
alias_inv,STR_DELIMIT, invitados);
           system(cadena);
           if(ban==0)
           {
            write_cli(ssock,"9910;#");
            ban=1;
           }
         }//fin 2do est_arch.st_size
         else
          printf ("el alias ya ha sido invitado por el otro..\n");
        }//fin 2do estat
        else
         printf ("no se pudo hacer 2do stat..\n");
       }//fin 2do system
       else
        printf ("no se pudo hacer 2do system..\n");
      }//fin 1er est_arch.size
      else
       printf ("ya existe partido con dicho contrncante..\n");
     }//fin de 1er stat
     else
      printf ("no se pudo realizar 1er estat\n");
     sprintf (cadena, "rm tmp_%s", alias_sol);
     system (cadena);
    }
  }//fin while
  if(ban==0)
    write_cli(ssock,"9909;#");
 }

 /*************************************************************************/
```

6.1.18 void env_inv(int ssock,char * buffer)

```
{
  int N =80,N2=30,N3=40,N4=4000,N5=10;
  char * lista_no, * lista_si, * cadena;
  char * alias1, * alias2, *alias_sol, * nombre1;
  char * ban_inv;
  FILE * fp;

  alias_sol=(char *)malloc(N2);
  alias1=(char *)malloc(N2);
  alias2=(char *)malloc(N2);
```

```
nombre1=(char *)malloc(N3);
lista_no=(char *)malloc(N4);
lista_si=(char *)malloc(N4);
cadena = (char *) malloc(N);
ban_inv = (char *) malloc(N5);

alias_sol=strtok(buffer,STR_DELIMIT);
buffer = buffer + strlen(alias_sol) + 1;
if( (fp = fopen (invitados,"r")) != NULL)
{
  fseek(fp,0,0);
  while( fgets(cadena,N,fp) != NULL)
  {
    alias1=strtok(cadena,STR_DELIMIT);
    alias2=strtok(NULL,STR_DELIMIT);
    ban_inv=strtok(NULL,"\n");
    if( strcmp(alias2 , alias_sol)==0  &&  strcmp(ban_inv,"F")==0 )
    {
      strcat(lista_no,alias1);
      strcat(lista_no,STR_DELIMIT);
    }
  }
  fclose(fp);
  if( (fp = fopen(players,"r")) != NULL)
  {
    fseek(fp,0,0);
    while( fgets(cadena,N,fp) != NULL)
    {
      alias1=strtok(cadena,STR_DELIMIT);
      nombre1=strtok(NULL,STR_DELIMIT);
      if( no_consta(alias1,lista_no)==0 )
      {
        strcat(lista_si,alias1);
        strcat(lista_si,STR_DELIMIT);
        strcat(lista_si,nombre1);
        strcat(lista_si,STR_DELIMIT);
      }
      else
        printf("No consta en lista_no\n");
    }
    fclose(fp);
    strcat(lista_si,STR_FIN_BUFFER);
    if(strcmp(lista_si,"#")==0)
      strcpy(lista_si,"9912;#");
    write_cli(ssock,lista_si);
  }
  else
  {
    write_cli(ssock,"9911;#");
  }
}
else
{
  write_cli(ssock,"9911;#");
}
free(alias_sol);
free(alias1);
free(alias2);
```

```
  free(nombre1);
  free(lista_no);
  free(lista_si);
  free(cadena);
  free(ban_inv);
}
/*****************************************************************************************/
```

6.1.19 void insertar_linea_damas(char * color_sol, char * alias_sol, char * alias_inv)

```
{
  char cadena[210];
  char cadena2[30];
  int i,j,len;
  FILE *fp;
  limpiar(cadena2,30);
  limpiar(cadena,210);
  if(strcmp(color_sol,STR_COLOR_INI)==0 )
  {
    strcat(cadena2,alias_sol);
    strcat(cadena2,STR_DELIMIT);
    strcat(cadena2,alias_inv);
    strcat(cadena2,STR_DELIMIT);
    strcat(cadena2,STR_COLOR_INI);
sprintf(cadena,"%s;00;0;0;00;2;0;00;4;0;00;6;0;00;1;1;00;3;1;00;5;1;00;7;1;00;0;2;00;2;2;00;4;2;
00;6;2;10;1;5;10;3;5;10;5;5;10;7;5;10;0;6;10;2;6;10;4;6;10;6;6;10;1;7;10;3;7;10;5;7;10;7;7",caden
a2);
    len=strlen(cadena);
    cadena[len]='\n';
    cadena[len+1]='\0';
  }
  else
  {
    strcat(cadena2,alias_inv);
    strcat(cadena2,STR_DELIMIT);
    strcat(cadena2,alias_sol);
    strcat(cadena2,STR_DELIMIT);
    strcat(cadena2,STR_COLOR_INI);
sprintf(cadena,"%s;00;0;0;00;2;0;00;4;0;00;6;0;00;1;1;00;3;1;00;5;1;00;7;1;00;0;2;00;2;2;00;4;2;
00;6;2;10;1;5;10;3;5;10;5;5;10;7;5;10;0;6;10;2;6;10;4;6;10;6;6;10;1;7;10;3;7;10;5;7;10;7;7",caden
a2);
    len=strlen(cadena);
    cadena[len]='\n';
    cadena[len+1]='\0';
  }
 if( (fp = fopen (damas,"a+")) != NULL)
 {
   fseek(fp,0,2);
   fwrite(cadena,strlen(cadena),1,fp);
   fclose(fp);
 }
 else
 {
   printf("No se puede abrir el archivo damas.txt \n");
 }
}
```

```
/*************************************************************************/
```

6.1.20 void reg_resp_inv(int ssock, char * buffer)

```
{
 int N=80, ban =1, len=0 ,bandera=0;
 long len2=0,len3=0;
 char alias_sol[10], alias_inv[10], color_sol[80],  ban_inv[80];
 char alias1[10], alias2[10], cadena[80] , cadena2[80];
 FILE * fp;

 strcpy(alias_sol,strtok(buffer,STR_DELIMIT));
 buffer=buffer + strlen(alias_sol) +1;
 if( (fp = fopen (invitados,"r+")) != NULL)
 {
   while(strcmp(buffer,STR_FIN_BUFFER) != 0)
   {
     strcpy(alias_inv , strtok(buffer,STR_DELIMIT));
     buffer = buffer + strlen(alias_inv) + 1;
     strcpy(color_sol , strtok(buffer,STR_DELIMIT));
     buffer = buffer + strlen(color_sol) + 1;
     fseek(fp,0,0);
     ban = 1;
     limpiar(cadena,N);
     len2 =0;
     while( fgets(cadena,N-1,fp) != NULL)
     {
       len3 = strlen(cadena);
       strcpy(alias1 , strtok(cadena,STR_DELIMIT));
       strcpy(alias2 , strtok(NULL,STR_DELIMIT));
       strcpy(ban_inv , strtok(NULL,"\n"));
       if(strcmp(alias1 , alias_inv) == 0 && strcmp(alias2 , alias_sol)==0)
       {
         if( strcmp(ban_inv,"F")==0 )
         {
           limpiar(cadena2,N);
           sprintf(cadena2,"%s;%s;V",alias_inv,alias_sol);
           len = strlen(cadena2);
           cadena2[len] = '\n';
           cadena2[len+1] = '\0';
           fseek(fp,len2,0);
           fwrite(cadena2,strlen(cadena2),1,fp);
           ban = 0;
           insertar_linea_damas(color_sol,alias_sol,alias_inv);
           if(bandera==0)
           {
              write_cli(ssock,"9910;#");
              bandera=1;
           }
           break;
         }
         else
         {
           ban = 0;
         }
       }
       len2 = len2 + len3;
       limpiar(cadena,N);
```

```
      } /* Sale del while luego de recorrer todo el archivo invitados.txt*/
      if(ban==1)
      {
        printf("%s :No ha sido invitado por: %s \n",alias_sol,alias_inv);
      }
    }
    fclose(fp);
  }
  else
  {
    printf("No se puede abrir el archivo invitados.txt \n");
  }
  if(bandera==0)
    write_cli(ssock,"9911;#");
}

/*************************************************************************/
```

6.1.21 void recv_jugada(int ssock, char * buffer)

```
{
  int N=50, N2=300,len, i;
  char alias_sol[10], alias_inv[10], cadena[500], temporal[300];
  struct stat est_arch;

  strcpy (temporal, buffer);
  i = strlen(temporal);
  temporal[i-2]='\n';
  temporal[i-1]='\0';
  strcpy(alias_inv , strtok (buffer, STR_DELIMIT));
  strcpy(alias_sol , strtok (NULL, STR_DELIMIT));

  sprintf(cadena,"grep \"^%s%s%s%s\" %s >
05_%s",alias_sol,STR_DELIMIT,alias_inv,STR_DELIMIT,invitados,alias_sol);

  if (system (cadena)!=-1)
  {
    sprintf (cadena, "05_%s", alias_sol);
    if (stat(cadena,&est_arch)==0)
    {
      if (est_arch.st_size == 0 )
      {
        sprintf(cadena,"grep \"^%s%s%s%s\" %s >
05_%s",alias_inv,STR_DELIMIT,alias_sol,STR_DELIMIT,invitados,alias_sol);
        if (system (cadena)!=-1)
        {
          sprintf (cadena, "05_%s", alias_sol);
          if (stat(cadena,&est_arch)==0)
          {
            if (est_arch.st_size == 0 )
            {
              printf("No existe partido con ese par de alias:%s,%s\n",alias_sol,alias_inv);
            }
            else
            {
              sprintf (cadena, "/bin/echo -n \"%s\" >> %s ",temporal,damas);
              system(cadena);
            }
```

```
          }
          else
            printf ("no se pudo hacer 2do stat..\n");
        }//fin 2do system
        else
          printf ("no se pudo hacer 2do system..\n");
      }//fin 1er est_arch.size
      else
      {
        sprintf (cadena, "/bin/echo -n \"%s\" >> %s ",temporal,damas);
        system(cadena);
      }
    }//fin de 1er stat
    else
      printf ("no se pudo realizar 1er estat\n");
    sprintf(cadena,"rm 05_%s",alias_sol);
    system(cadena);
  }
  else
    printf ("no se pudo realizar el system\n");
}

/*****************************************************************************************/
```

6.1.22 void cambia_puntos(char * alias, int tipo)

```
{
 FILE *ptr,*qtr;
 int b = 1,ipg,ipp;
 char alias1[10],nombre[40],pg[4],pp[4];
 char *cadena, *file1;
 char comando[100];
 char pass[10];

 cadena = (char *)malloc(4000);
 sprintf(comando,"grep %s %s > 074_%s",alias,players,alias);
 system(comando);
  sprintf(comando,"grep -v %s %s > 075_%s",alias,players,alias);
 system(comando);
 sprintf(comando,"cp 075_%s %s",alias,players);
 system(comando);
 ptr = fopen(players,"a+");
 if(ptr == NULL)
 {
   b = 0;
 }
 sprintf(file1,"074_%s",alias);
 qtr = fopen(file1,"r");
 if(qtr == NULL)
 {
   b = 0;
 }
 if (b)
 {
   if(fgets(cadena,4000,qtr) != NULL)
   {
     strcpy(alias1,strtok(cadena,STR_DELIMIT));
     strcpy(nombre,strtok(NULL,STR_DELIMIT));
```

```
    strcpy(pass,strtok(NULL,STR_DELIMIT));
    strcpy(pg,strtok(NULL,STR_DELIMIT));
    strcpy(pp,strtok(NULL,"\n"));
    ipg = atoi(pg);
    ipp = atoi(pp);
    if (tipo == 0)
       ipg++;
    else
       ipp++;
    sprintf(cadena,"%s;%s;%s;%d;%d\n",alias1,nombre,pass,ipg,ipp);
    fwrite(cadena,strlen(cadena),1,ptr);
  }
 }
 fclose(ptr);
 fclose(qtr);
 sprintf(comando,"rm 074_%s",alias);
 system(comando);
 sprintf(comando,"rm 075_%s",alias);
 system(comando);
}

/*************************************************************************/
```

6.1.23 void elimina_alias_invitados(char * alias1, char * alias2)

```
{
 char comando[100],alias[30],file1[100],file2[100];
 struct stat est_arch;

 sprintf(file1,"071_%s",alias1);
 strcpy(alias,alias1);
 strcat(alias,STR_DELIMIT);
 strcat(alias,alias2);
 strcat(alias,STR_DELIMIT);
 strcat(alias,"V");
 sprintf(comando,"grep \'%s\' %s > 071_%s",alias,invitados,alias1);

 if (system (comando)!=-1)
 {
 if (stat(file1,&est_arch)==0)
 {
   if (est_arch.st_size != 0 )
   {
     sprintf(comando,"grep -v \'%s\' %s > 072_%s",alias,invitados,alias1);
     system(comando);
     sprintf(comando,"cp 072_%s %s",alias1,invitados);
     system(comando);
     elimina_partido_damas(alias1,alias2);
     cambia_puntos(alias1,1);
     cambia_puntos(alias2,0);
   }
   else
   {
     strcpy(alias,alias2);
     strcat(alias,STR_DELIMIT);
     strcat(alias,alias1);
     strcat(alias,STR_DELIMIT);
     strcat(alias,"V");
```

```
    sprintf(comando,"grep \'%s\' %s > 071_%s",alias,invitados,alias1);
    if (system (comando)!=-1)
    {
     if (stat(file1,&est_arch)==0)
     {
      if (est_arch.st_size != 0 )
      {
         sprintf(comando,"grep -v \'%s\' %s > 072_%s",alias,invitados,alias1);
         system(comando);
         sprintf(comando,"cp 072_%s %s",alias1,invitados);
         system(comando);
         elimina_partido_damas(alias1,alias2);
         cambia_puntos(alias1,0);
         cambia_puntos(alias2,1);
      }
      else
        printf("los alias no existen dentro de invita.txt\n");
     }//cierra el stat
     else
        printf("Error en el 2do stat\n");
    }// cierra el system
    else
       printf("No pudo hacer el 2do system\n");
  }//cierra el else
 }// cierra el 1er stat
 else
   printf("Error en el 1er stat\n");
}//cierra el 1er system
else
   printf("No pudo hacer el 1er system\n");
sprintf(comando,"rm 071_%s",alias1);
system(comando);
sprintf(comando,"rm 072_%s",alias1);
system(comando);
}//cierra la funcion

/*****************************************************/
```

6.1.24 void elimina_partido_damas(char * alias1, char * alias2)

```
{
 FILE *ptr;
 int b = 1;
 char comando[100];

 sprintf(comando,"grep -v \'%s;%s\' %s > 073_%s",alias1,alias2,damas,alias1);
 system(comando);
 sprintf(comando,"cp 073_%s %s",alias1,damas);
 system(comando);
 ptr = fopen(damas,"a+");
 if(ptr == NULL)
 {
   b = 0;
 }
 if(b)
 {
  fseek(ptr,0,2);
  sprintf(comando,"%s;%s;9905\n",alias2,alias1);
```

```
 fwrite(comando,strlen(comando),1,ptr);
 }
 fclose(ptr);
 sprintf(comando,"rm 073_%s",alias1);
 system(comando);
}

/*********************************************************/
```

6.1.25 void matar_partido(int ssock,char * buffer)

```
{
 char alias1[10],alias2[10],alias[21];

 strcpy(alias1,strtok(buffer,STR_DELIMIT));
 buffer = buffer + strlen(alias1) + 1;
 while(buffer[0]!= '#')
 {
   strcpy(alias2,strtok(buffer,STR_DELIMIT));
   buffer = buffer + strlen(alias2) + 1;
   elimina_alias_invitados(alias1,alias2);
 }
}

/*********************************************************/
```

6.1.26 void partidos_actuales (int ssock, char * buffer)

```
{
  int ban=0, i;
  char alias_sol[10], lista[1000], cadena[500], file1[20];
  struct stat est_arch;
  FILE *fp;

  strcpy(alias_sol,strtok(buffer,STR_DELIMIT));
  sprintf (file1, "11_%s", alias_sol);
  sprintf(cadena,"grep \"^%s%s\" %s |grep \";V$\" |cut -f2 -d\';\' >
%s",alias_sol,STR_DELIMIT,invitados,file1);

  if (system (cadena)!=-1)
  {
    sprintf(cadena,"grep \"%s%s%s\" %s |grep \";V$\" |cut -f1 -d\';\' >>
%s",STR_DELIMIT,alias_sol,STR_DELIMIT,invitados,file1);
    system(cadena);
     if (stat(file1,&est_arch)==0)
     {
       if (est_arch.st_size != 0 )
       {
         if ((fp = fopen (file1,"r"))!=NULL)
         {
           limpiar(cadena,15);
           limpiar(lista,500);
           while((fgets(cadena,15,fp)) != NULL)
           {
             i=strlen(cadena);
             cadena[i-1]='\0';
             strcat(lista,cadena);
             strcat(lista,STR_DELIMIT);
             limpiar(cadena,15);
```

```
            }
            strcat(lista,STR_FIN_BUFFER);
            fclose(fp);
            write_cli(ssock,lista);
            ban=1;
          }//fin fopen
          else
            printf("No pudo abrir el archivo 11_alias_inv\n");
        }//fin de est_arch
        else
          printf("La dimension del archivo es cero\n");
      }//fin stat
      else
        printf("No hizo el stat \n");
      sprintf(cadena,"rm %s",file1);
      system(cadena);
    }//fin system
    else
      printf("NO pudo hacer el 1er system\n");
    if(ban=0)
      write_cli(ssock,"9911;#");
}//fin fun

/********************** ENVIAR JUGADA ***********************************/
```

6.1.27 *void env_jugada(int ssock,char * buffer)*

```
{
 FILE *qtr;
 char * temporal;
 char comando[200], cadena[400], alias[10];
 char file1[20];
 int len;
 struct stat est_arch;

 strcpy(alias,strtok(buffer,STR_DELIMIT));
 sprintf(file1,"061_%s",alias);
 temporal = (char *)malloc(700);
 sprintf(comando,"grep %s %s > 061_%s",alias,damas,alias);
 system(comando);

 /* copia en tem2 todas las lineas que no tengan alias en damas */
 sprintf(comando,"grep -v %s %s > 062_%s",alias,damas,alias);
 system(comando);
 sprintf(comando,"cp 062_%s %s",alias,damas);
 system(comando);
 stat(file1,&est_arch);
 qtr = fopen(file1,"r");
 printf("Antes del if qtr \n");
 if((qtr != NULL) && (est_arch.st_size>0))
 {
   rewind(qtr);
   printf("Antes del while \n");
   while((fgets(temporal,500,qtr)) != NULL)
   {
     len = strlen(temporal);
     temporal[len-1]=';';
     temporal[len]='#';
```

```
    temporal = temporal + strlen(alias) + 1;
    write_cli(ssock,temporal);
  }
  fclose(qtr);
}
else
{
  write_cli(ssock,"9911;#");
}
free (temporal);
sprintf(file1,"rm 061_%s",alias);
system(file1);
sprintf(file1,"rm 062_%s",alias);
system(file1);
}
```

6.2 LISTADO DEL CLIENTE

6.2.1 Archivo about.txt

```
Option Explicit

Sub Form_Click ()
   main!comentario.Caption = ""
   main!Panel3D1.Caption = ""
   Unload Me
End Sub

Sub Form_Load ()
   main!Panel3D1.Caption = "Haga CLICK en la forma para abandonarla"
End Sub

Sub Form_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
   main!comentario.Caption = "Acerca de DAMAS...."
End Sub

Sub Image1_Click ()
   main!comentario.Caption = ""
   Unload Me
End Sub

Sub Image1_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
   main!comentario.Caption = "Acerca de DAMAS...."
End Sub

Sub Label1_Click ()
   main!comentario.Caption = ""
   Unload Me
End Sub

Sub Label1_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
   main!comentario.Caption = "Acerca de DAMAS...."
End Sub

Sub Label2_Click ()
   main!comentario.Caption = ""
   Unload Me
End Sub

Sub Label2_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
   main!comentario.Caption = "Acerca de DAMAS...."
End Sub

Sub Label3_Click ()
   main!comentario.Caption = ""
   Unload Me
End Sub

Sub Label3_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
   main!comentario.Caption = "Acerca de DAMAS...."
End Sub
```

```
Sub Label4_Click ()
   main!comentario.Caption = ""
   Unload Me
End Sub

Sub Label4_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
   main!comentario.Caption = "Acerca de DAMAS...."
End Sub

Sub Label5_Click ()
   main!comentario.Caption = ""
   Unload Me
End Sub

Sub Label5_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
   main!comentario.Caption = "Acerca de DAMAS...."
End Sub

Sub Label6_Click ()
   main!comentario.Caption = ""
   Unload Me
End Sub

Sub Label6_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
   main!comentario.Caption = "Acerca de DAMAS...."
End Sub

Sub Label7_Click ()
   main!comentario.Caption = ""
   Unload Me
End Sub

Sub Label7_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
   main!comentario.Caption = "Acerca de DAMAS...."
End Sub

Sub Label8_Click ()
   main!comentario.Caption = ""
   Unload Me
End Sub

Sub Label8_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
   main!comentario.Caption = "Acerca de DAMAS...."
End Sub

Sub Label9_Click ()
   main!comentario.Caption = ""
   Unload Me
End Sub

Sub Label9_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
   main!comentario.Caption = "Acerca de DAMAS...."
End Sub

Sub Picture1_Click ()
   main!comentario.Caption = ""
   Unload Me
```

```
End Sub

Sub Picture1_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
   main!comentario.Caption = "Acerca de Damas..."
End Sub
```

6.2.2 *Archivo ficha.txt*

```
Dim dragx As Single, dragy As Single
Dim des_fich As Single, tam_cua As Single
Dim fic_act As Single
Dim equipo As String, usuario As String
Dim desf As Single
Dim tipf As String
Dim ban As Integer
Dim turno As String
Dim f_roja(12) As tipoficha
Dim f_azul(12) As tipoficha


Sub cargar ()
Dim db As database
Dim fichas As table
Dim vista As Dynaset
Dim query1 As String

  ruta$ = PATH + ERR_MDB$
  Set db = OpenDatabase(ruta$)
  Set vista = db.CreateDynaset("fichas")
  selec = "usuario= '" & usuario & "'"
  indr = 0
  inda = 0
  vista.FindFirst selec
  If Not vista.NoMatch Then
    For i = 0 To 11
       Form1!ficha_roja(i).Visible = False
       Form1!ficha_azul(i).Visible = False
       f_roja(i).posx = 0
       f_roja(i).posy = 0
       f_roja(i).reina = 0
       f_roja(i).estado = 0
       f_azul(i).posx = 0
       f_azul(i).posy = 0
       f_azul(i).reina = 0
       f_azul(i).estado = 0
    Next i
    Do Until vista.NoMatch
       If vista("equipo") = 1 Then
          f_roja(indr).posx = vista("posx")
          f_roja(indr).posy = vista("posy")
          f_roja(indr).reina = vista("reina")
          If f_roja(indr).reina = 1 Then Form1!ficha_roja(indr).Picture =
LoadPicture("/damas2/ficha2r.bmp")
          f_roja(indr).estado = 1
          Form1!ficha_roja(indr).Visible = True
```

```
        Form1!ficha_roja(indr).Move coor(vista("posx")) + des_fich, coor(vista("posy")) +
des_fich
        indr = indr + 1
      Else
        f_azul(inda).posx = vista("posx")
        f_azul(inda).posy = vista("posy")
        f_azul(inda).reina = vista("reina")
        If f_azul(inda).reina = 1 Then Form1!ficha_azul(inda).Picture =
LoadPicture("/damas2/ficha3r.bmp")
        f_azul(inda).estado = 1
        Form1!ficha_azul(inda).Visible = True
        Form1!ficha_azul(inda).Move coor(vista("posx")) + des_fich, coor(vista("posy")) +
des_fich
        inda = inda + 1
      End If
      vista.FindNext selec
   Loop
  End If

End Sub

Sub Command2_Click ()
   Unload Me
End Sub

Sub Command3_Click ()
Dim db As database
Dim fichas As table, fichas2 As table, fichas_out As table
Dim vista As Dynaset, vista2 As Dynaset, vista_out As Dynaset

   ruta$ = PATH + ERR_MDB$
   Set db = OpenDatabase(ruta$)
   'Borra datos anteriores del usuario
   Set fichas2 = db.OpenTable("fichas")
   Set vista2 = db.CreateDynaset("fichas2")
   selec = "usuario = '" & usuario & "'"
   vista2.FindFirst selec
   If Not vista2.NoMatch Then
     Do Until vista2.NoMatch
       vista2.Delete
       vista2.FindNext selec
     Loop
   End If
   vista2.Close
   'graba datos en tabla fichas_out
   Set fichas = db.OpenTable("fichas_out")
   Set vista = db.CreateDynaset("fichas")

   For ind = 0 To 11
      If f_roja(ind).estado = 1 Then
         fichas.AddNew
         fichas("usuario") = usuario
         fichas("equipo") = 1
         fichas("posx") = f_roja(ind).posx
         fichas("posy") = f_roja(ind).posy
         fichas("reina") = f_roja(ind).reina
         fichas.Update
      End If
```

```
    Next ind

    'Fichas Azules

    For ind = 0 To 11
       If f_azul(ind).estado = 1 Then
          fichas.AddNew
          fichas("usuario") = usuario
          fichas("equipo") = 2
          fichas("posx") = f_azul(ind).posx
          fichas("posy") = f_azul(ind).posy
          fichas("reina") = f_azul(ind).reina
          fichas.Update
       End If
    Next ind
    vista.Close
    borra_contrinin user$
    Unload Me
End Sub

Function coor (xx)
      coor = (xx * tam_cua) + desf
End Function

Function existe (eq, xin, xfi, yin, yfi)
    pbx = 0
    pby = 0
    pbx = (xin + xfi) / 2
    pby = (yin + yfi) / 2
    existe = -1

    Select Case eq

    Case "roja"
       For i = 0 To 11
          If f_azul(i).posx = pbx And f_azul(i).posy = pby And f_azul(i).estado = 1 Then
             existe = i
             Exit For
          End If
       Next i

    Case "azul"
       For i = 0 To 11
          If f_roja(i).posx = pbx And f_roja(i).posy = pby And f_roja(i).estado = 1 Then
             existe = i
             Exit For
          End If
       Next i
    End Select
End Function

Sub ficha_azul_Click (Index As Integer)
    If tipf = "x" And turno = "azul" And ban = 0 Then
       tipf = "azul"
       fic_act = Index
       ficha_azul(Index).BorderStyle = 1
    End If
End Sub
```

```
Sub ficha_roja_Click (Index As Integer)
   If tipf = "x" And turno = "roja" And ban = 0 Then
      tipf = "roja"
      fic_act = Index
      ficha_roja(Index).BorderStyle = 1
   End If
End Sub

Sub Form_Load ()
des_fich = 30
tam_cua = 618
desf = 600
ban = 0
usuario = user$
turno = clr$
If clr$ = "roja" Then
   elrojo.Caption = "Mi turno: " + alias_gm
   elazul.Caption = "Contrincante: " + user$
Else
   elazul.Caption = "Mi turno: " + alias_gm
   elrojo.Caption = "Contrincante: " + user$
End If
Form1.Caption = "Juego de Damas" + " Turno: " + clr$
tipf = "x"
cargar
End Sub

Sub Form_MouseDown (Button As Integer, Shift As Integer, X As Single, Y As Single)
Dim xi, xf, yi, yf, com, mov
equipo = tipf
cam_tur = 1
com = 0
mov = 0

If turno = equipo Then
  xf = pos(X)  'se guarda la posicion final
  yf = pos(Y)
  If xf < 0 Then xf = 0
  If yf < 0 Then yf = 0
  If xf > 7 Then xf = 7
  If yf > 7 Then yf = 7
  Select Case equipo
    Case "roja"

      xi = f_roja(fic_act).posx
      yi = f_roja(fic_act).posy

      If f_roja(fic_act).reina = 0 Then
        cond1 = yf - yi = 1
        cond3 = yf - yi = 2
      Else
        cond1 = Abs(yf - yi) = 1
        cond3 = Abs(yf - yi) = 2
      End If
      cond2 = Abs(xf - xi) = 1
      cond4 = Abs(xf - xi) = 2
```

```
    If (cond1 And cond2) Then mov = 1
    ind_enc = existe(equipo, xi, xf, yi, yf)
    If (cond3 And cond4 And ind_enc <> -1) Then
      mov = 1
      com = 1
    End If

    If com = 1 Then
      ficha_azul(ind_enc).Visible = False
      f_azul(ind_enc).estado = 0
    End If

    If mov = 1 Then
      ficha_roja(fic_act).Move coor(xf) + des_fich, coor(yf) + des_fich
      f_roja(fic_act).posx = xf
      f_roja(fic_act).posy = yf
      If yf = 7 Then
        ficha_roja(fic_act).Picture = LoadPicture("/damas2/ficha2r.bmp")
        f_roja(fic_act).reina = 1
      End If
    End If

    If (mov = 0 And com = 0) Then
      MsgBox ("Movimiento invàlido")
      cam_tur = 0
    End If

    ficha_roja(fic_act).BorderStyle = 0

  Case "azul"
    xi = f_azul(fic_act).posx
    yi = f_azul(fic_act).posy

    If f_azul(fic_act).reina = 0 Then
      cond1 = yf - yi = -1
      cond3 = yf - yi = -2
    Else
      cond1 = Abs(yf - yi) = 1
      cond3 = Abs(yf - yi) = 2
    End If
    cond2 = Abs(xf - xi) = 1
    cond4 = Abs(xf - xi) = 2

    If (cond1 And cond2) Then mov = 1
    ind_enc = existe(equipo, xi, xf, yi, yf)
    If (cond3 And cond4 And ind_enc <> -1) Then
      mov = 1
      com = 1
    End If

    If com = 1 Then
      ficha_roja(ind_enc).Visible = False
      f_roja(ind_enc).estado = 0
    End If

    If mov = 1 Then
      ficha_azul(fic_act).Move coor(xf) + des_fich, coor(yf) + des_fich
      f_azul(fic_act).posx = xf
```

```
      f_azul(fic_act).posy = yf
      If yf = 0 Then
        ficha_azul(fic_act).Picture = LoadPicture("/damas2/ficha3r.bmp")
        f_azul(fic_act).reina = 1
      End If
    End If

    If (mov = 0 And com = 0) Then
      MsgBox ("Movimiento invàlido")
      cam_tur = 0
    End If

  ficha_azul(fic_act).BorderStyle = 0
 End Select

 tipf = "x"
If cam_tur <> 0 Then
  ban = 1
End If
End If

End Sub

Function pos (xx)
   pos = Int((xx - desf) / tam_cua)
End Function

Function prox (eq, rei, xin, yin)
xf1 = -1
xf2 = -1
yf1 = -1
yf2 = -1
ind1 = 0
ind2 = 0
ind3 = 0
ind4 = 0


If xin + 2 <= 7 Then xf1 = xin + 2
If xin - 2 >= 0 Then xf2 = xin - 2
If yin + 2 <= 7 Then yf1 = yin + 2
If yin - 2 >= 0 Then yf2 = yin - 2


If rei = 1 Then
   ind1 = existe(eq, xin, xf2, yin, yf2)
   ind2 = existe(eq, xin, xf1, yin, yf2)
   ind3 = existe(eq, xin, xf1, yin, yf1)
   ind4 = existe(eq, xin, xf2, yin, yf1)
Else
   If eq = "roja" Then
      ind3 = existe(eq, xin, xf1, yin, yf1)
      ind4 = existe(eq, xin, xf2, yin, yf1)
    Else
      ind1 = existe(eq, xin, xf2, yin, yf2)
      ind2 = existe(eq, xin, xf1, yin, yf2)
    End If
```

```
End If

MsgBox (ind1 & ind2 & ind3 & ind4)
If ind1 <> 0 Or ind2 <> 0 Or ind3 <> 0 Or ind4 <> 0 Then prox = 1 Else prox = 0

End Function
```

6.2.3 *Archivo inv_game.txt*

```
Sub aceptar_Click ()
Dim cantd As Integer
Dim i As Integer
Dim buff As String
Dim value$

   main!comentario.Caption = ""
   buff = INVITEPLAYER$ + delimeter$ + alias_gm + delimeter$
   cantd = List2.ListCount
   If cantd > 0 Then
         For i = 0 To cantd - 1
            buff = buff + Trim(List2.List(0)) + delimeter$
            List2.Selected(0) = True
            List2.RemoveItem List2.ListIndex
            CNTD% = CNTD% - 1
         Next i
         buff = buff + endbuff$
         main!sock.Action = 1
         If conectado = True Then
            value$ = send_server(buff)
            If CInt(value$) = Len(buff) Then
                  buff = ""
                  buff = recv_server(buff)
                  main!sock.Action = 4
                  If Left(buff, 4) = "9910" Then
                  Else
                     message Left(buff, 4), "ERROR", 32  'msgbox
                  End If

             Else
                  value$ = errmsg(value$)
                  message value$ + Chr(10) + "No se han podido enviar " + Chr(10) + "al servidor
las solicitudes", "DESCRIPCION", 48
             End If
             main!sock.Action = 4
          Else
          End If
    Else
          message "No se ha seleccionado contrincante", "ERROR SELECCION", 32
    End If
    main!Panel3D1.Caption = ""
    If CNTD% = 0 Then Unload Me

 End Sub

 Sub aceptar_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
    main!comentario.Caption = "Aceptar las invitaciones selecionadas"
```

```
End Sub

Sub Form_Load ()
Dim query As String
Dim i As Integer

   main!Panel3D1.Caption = "Haga doble Click en el alias para seleccionarlo"
   For i = 0 To CNTD% - 1
         List1.AddItem lista(i).alias
   Next i
   List1.Selected(0) = True
   i = indice(CStr(List1.Text), 1)
   label3.Caption = UCase$(lista(i).name)

End Sub

Sub Form_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
   main!comentario.Caption = "Invitar a jugar a alguien"
End Sub

Sub Label1_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
   main!comentario.Caption = ""
End Sub

Sub Label2_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
   main!comentario.Caption = ""
End Sub

Sub Label3_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
   main!comentario.Caption = ""
End Sub

Sub List1_Click ()
Dim i As Integer
Dim alia_a As String

   alia_a = List1.Text
   i = indice(alia_a, 1)
   label3.Caption = UCase$(lista(i).name)

End Sub

Sub List1_DblClick ()
Dim i As Integer
Dim alia_a As String

   alia_a = List1.Text
   i = indice(alia_a, 1)
   List1.RemoveItem List1.ListIndex
   List2.AddItem lista(i).alias

End Sub

Sub List1_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
   main!comentario.Caption = "Subscriptores a invitar"
End Sub

Sub List2_Click ()
```

```
Dim i As Integer
Dim alia_a As String

   alia_a = List2.Text
   i = indice(alia_a, 1)
   label3.Caption = UCase$(lista(i).name)

End Sub

Sub List2_DblClick ()
Dim i As Integer
Dim alia_a As String

   alia_a = List2.Text
   i = indice(alia_a, 1)
   List2.RemoveItem List2.ListIndex
   List1.AddItem lista(i).alias

End Sub

Sub List2_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
   main!comentario.Caption = "Subscriptores invitados ya seleccionados"
End Sub

Sub name_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
   main!comentario.Caption = "Nombre del subscriptor de la lista sombreado"
End Sub

Sub salir_Click ()
Dim lista_tmp() As list_alias
   main!Panel3D1.Caption = ""
   main!comentario.Caption = ""

   cantd% = List1.ListCount
   CNTD% = cantd%
   For i = 0 To CNTD% - 1
         ReDim Preserve lista_tmp(i + 1)
         List1.Selected(i) = True
         j = indice(CStr(List1.Text), 1)
         lista_tmp(i).name = Trim$(UCase$(lista(j).name))
         lista_tmp(i).alias = List1.Text
   Next i
   For i = 0 To CNTD% - 1
         lista(i).name = lista_tmp(i).name
         lista(i).alias = lista_tmp(i).alias
   Next i

   Unload invite_game

End Sub

Sub salir_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
   main!comentario.Caption = ""
End Sub
```

6.2.4 Archivo kill.txt

```
Option Explicit
Dim buf$

Sub aceptar_Click ()
Dim value$, j%, i

   j% = List2.ListCount
   For i = 0 To j% - 1
      List2.Selected(0) = True
      buf$ = buf$ + Trim$(List2.Text) + delimeter$
      List2.RemoveItem List2.ListIndex
   Next i

   buf$ = buf$ + endbuff$
    If Len(Trim$(buf$)) > (Len(KILLGAME$ + delimeter$ + alias_gm + delimeter$ + endbuff$))
Then
      main!sock.Action = 1
      If conectado = True Then
         value$ = send_server(buf$)
         If Len(value$) = Len(buf$) Then
            buf$ = ""
            value$ = recv_server(buf$)
            value$ = errmsg(value$)
            message value$, "RESPUESTA DE MATAR", 64
         End If
      Else
         message "No se ha podido hacer conexion con servidor", "CONEXION", 16
      End If
   Else
      message "No se ha seleccionado ningun partido", "KILL GAMES", 48
   End If

   If List1.ListCount = 0 Then
      Unload Me
   End If
   buf$ = KILLGAME$ + delimeter$ + alias_gm
   main!Panel3D1.Caption = ""
End Sub

Sub aceptar_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
   main!comentario.Caption = "Matar los partidos ya seleccionados"
End Sub

Sub Form_Load ()
Dim i As Integer

   buf$ = KILLGAME$ + delimeter$ + alias_gm + delimeter$
   For i = 0 To CNTD_K% - 1
      List1.AddItem kil(i)
   Next i
   List1.Selected(0) = True

End Sub

Sub Form_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
```

```
   main!comentario.Caption = "Matar partidos en los que se encuentra involucrado"
End Sub

Sub List1_DblClick ()
Dim alia_a$

   alia_a$ = Trim$(List1.Text)
   List2.AddItem alia_a$
   List1.RemoveItem List1.ListIndex

End Sub

Sub List1_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
   main!comentario.Caption = "Lista de contrincantes con los que se tiene partidos"
End Sub

Sub List2_DblClick ()
Dim alia_a$

   alia_a$ = Trim$(List2.Text)
   List1.AddItem alia_a$
   List2.RemoveItem List2.ListIndex

End Sub

Sub matar_Click ()

   buf$ = buf$ + delimeter$ + Trim$(List1.Text)
   If List1.ListCount > 0 Then
      List1.RemoveItem List1.ListIndex
      If List1.ListCount > 0 Then
         List1.Selected(0) = True
      End If
   Else
      message "No existen mas contrincantes", "ERROR TONTO", 16
   End If

End Sub

Sub matar_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
   main!comentario.Caption = "Permite añadir un partido que se desea terminar"
End Sub

Sub salir_Click ()
   main!comentario.Caption = ""
   Unload Me
End Sub

Sub salir_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
   main!comentario.Caption = ""
End Sub
```

6.2.5 Archivo main.txt

```
Sub about_Click ()
   about_h.Show
End Sub

Sub ami_Click ()
   If alias_gm <> "" Then
      message alias_gm, "Mi Alias", 64
   End If
End Sub

Sub answer_invite_Click ()
   If alias_gm <> "" Then
      recv_inv
      If CNTD_I% > 0 Then see_inv.Show
      Else message "Ud. no posee invitaciones", "INVITACIOES", 16
   End If
   ver
End Sub

Sub Command3D1_Click ()
   borra_gameout
   borra_gamein
   borra_listplay
   End
End Sub

Sub Command3D1_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)

   main!Label1.Visible = True
   main!Label2.Visible = False
   main!Label3.Visible = False
   main!Label4.Visible = False
   main!Label5.Visible = False
   main!Label6.Visible = False
   main!Label7.Visible = False
   main!Label8.Visible = False
   main!label9.Visible = False

End Sub

Sub Command3D2_Click ()
   If alias_gm <> "" Then
      desuscribirse
   End If
End Sub

Sub Command3D2_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)

   main!Label3.Visible = True
   main!Label1.Visible = False
   main!Label2.Visible = False
   main!Label4.Visible = False
   main!Label5.Visible = False
   main!Label6.Visible = False
   main!Label7.Visible = False
   main!Label8.Visible = False
   main!label9.Visible = False
```

```
End Sub

Sub Command3D3_Click ()
   subscribe.Show
End Sub

Sub Command3D3_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)

   main!Label2.Visible = True
   main!Label1.Visible = False
   main!Label3.Visible = False
   main!Label4.Visible = False
   main!Label5.Visible = False
   main!Label6.Visible = False
   main!Label7.Visible = False
   main!Label8.Visible = False
   main!label9.Visible = False

End Sub

Sub Command3D4_Click ()
   mover
End Sub

Sub Command3D4_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)

   main!Label4.Visible = True
   main!Label1.Visible = False
   main!Label2.Visible = False
   main!Label3.Visible = False
   main!Label5.Visible = False
   main!Label6.Visible = False
   main!Label7.Visible = False
   main!Label8.Visible = False
   main!label9.Visible = False

End Sub

Sub Command3D5_Click ()
   matar_gm
   matar.Show
End Sub

Sub Command3D5_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)

   main!Label5.Visible = True
   main!Label1.Visible = False
   main!Label2.Visible = False
   main!Label3.Visible = False
   main!Label4.Visible = False
   main!Label4.Visible = False
   main!Label6.Visible = False
   main!Label7.Visible = False
   main!Label8.Visible = False
   main!label9.Visible = False

End Sub
```

```
Sub Command3D6_Click ()
   If alias_gm <> "" Then
      obt_lista
      If CNTD% <> 0 Then invite_game.Show
      Else message "Ud. ya ha invitado a todos los jugadores", "INVITAR JUGADOR", 16
   End If
   ver
End Sub

Sub Command3D6_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)

   main!Label1.Visible = False
   main!Label2.Visible = False
   main!Label3.Visible = False
   main!Label4.Visible = False
   main!Label5.Visible = False
   main!Label6.Visible = False
   main!Label7.Visible = True
   main!Label8.Visible = False
   main!label9.Visible = False

End Sub

Sub Command3D7_Click ()
   If alias_gm <> "" Then
      recv_inv
      If CNTD_I% > 0 Then see_inv.Show
      Else message "Ud. no posee invitaciones", "INVITACIOES", 16
   End If
   ver
End Sub

Sub Command3D7_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)

   main!Label1.Visible = False
   main!Label2.Visible = False
   main!Label3.Visible = False
   main!Label4.Visible = False
   main!Label5.Visible = False
   main!Label6.Visible = True
   main!Label7.Visible = False
   main!Label8.Visible = False
   main!label9.Visible = False

End Sub

Sub Command3D8_Click ()
   If alias_gm <> "" Then
      borra_gameout
   Else
      message "No se ha identificado Ud. como jugador", "USUARIO", 16
   End If
   ver
End Sub

Sub Command3D8_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
```

```
    main!Label1.Visible = False
    main!Label2.Visible = False
    main!Label3.Visible = False
    main!Label4.Visible = False
    main!Label5.Visible = False
    main!Label6.Visible = False
    main!Label7.Visible = False
    main!Label8.Visible = False
    main!label9.Visible = True

End Sub

Sub Command3D9_Click ()
    If alias_gm <> "" Then
        message alias_gm, "Mi Alias", 64
    End If
End Sub

Sub Command3D9_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)

    main!Label1.Visible = False
    main!Label2.Visible = False
    main!Label3.Visible = False
    main!Label4.Visible = False
    main!Label5.Visible = False
    main!Label6.Visible = False
    main!Label7.Visible = False
    main!Label8.Visible = True
    main!label9.Visible = False

End Sub

Sub exit_Click ()
    borra_gameout
    borra_gamein
    borra_listplay
    Unload Me
    End
End Sub

Sub Form_Load ()
    cuenta_lstplayers
    cuenta_partidosin
    cuenta_partidosout
    ver
End Sub

Sub Form_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)

    main!comentario.Caption = ""
    main!Label1.Visible = False
    main!Label2.Visible = False
    main!Label3.Visible = False
    main!Label4.Visible = False
    main!Label5.Visible = False
    main!Label6.Visible = False
    main!Label7.Visible = False
    main!Label8.Visible = False
```

```
    main!label9.Visible = False

End Sub

Sub get_game_Click ()
    mover
End Sub


Sub Image1_Click ()
    Unload main
End Sub

Sub Image2_Click ()

End Sub

Sub invitar_jugador_Click ()
    If alias_gm <> "" Then
        obt_lista
        If CNTD% <> 0 Then invite_game.Show
        Else message "Ud. ya ha invitado a todos los jugadores", "INVITAR JUGADOR", 16
    End If
    ver
End Sub

Sub killer_Click ()
    If alias_gm <> "" Then
        matar_gm
    End If
    ver
End Sub

Sub matado_Click ()
    matar_gm
    matar.Show
End Sub

Sub mover ()
Dim buff$, value$, file$
Dim ch As String * 2, ban%
Dim longfile As Long

    ban% = False
    If alias_gm <> "" Then
    main!sock.Action = 1
    If conectado = True Then
        file$ = PATH + GAMEIN$
        buff$ = RECIBEGAME$ + delimeter$ + alias_gm + delimeter$ + endbuff$
        value$ = send_server(buff$)
        main!sock.ReceiveLen = 300
        j% = 0
        If CInt(value$) = Len(buff$) Then
          Do
            buff$ = ""
            buff$ = Trim(buff$) + recv_server(buff)'Trim(main!sock.Receive)
            If buff$ <> "" And Left$(buff$, 2) <> "99" Then
                If Len(buff$) < 15 Then
```

```
            lng = desconcadena(buff$)
            ch$ = Left$(buff, lng)  'obtengo primer campo
            usr$ = Trim$(ch$) 'almaceno el alias
            code$ = Left$(Right$(buff$, Len(buff$) - lng), 4)
            code$ = errmsg(code$)
            message code$ + usr$, "JUGADAS", 48
          Else
            graba_fichasin buff$
            ban% = True
          End If
          send_gm = True
          get_gm = True
          actualiza
          j% = 0
        Else
          If Left$(buff$, 4) <> "9909" Then
            buff$ = errmsg(Left$(buff$, 4))
            message buff$, "ERROR DE SOCKETS si buff es vacio", 16
          Else
          End If
        End If
        j% = j% + 1
      Loop Until j% = 2 Or Left$(buff$, 4) = "9911" Or Left$(buff$, 4) = "9909"
    Else
      value$ = errmsg(Left$(value$, 4))
      message value$, "ERROR del es le de lo q envie", 64
    End If
  Else
  End If
  main!sock.Action = 4
  main!Panel3D1.Caption = ""
  If ban% = True Then mov_fcha.Show
  Else
    message "Ud. no posee jugadas donde realizar movimientos", "OBTENER JUGADAS", 16
  End If
  ver


End Sub

Sub Panel3D2_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)

  main!Label1.Visible = False
  main!Label2.Visible = False
  main!Label3.Visible = False
  main!Label4.Visible = False
  main!Label5.Visible = False
  main!Label6.Visible = False
  main!Label7.Visible = False
  main!Label8.Visible = False
  main!label9.Visible = False

End Sub

Sub ranking_Click ()
Dim buff$, value$

  main!sock.Action = 1
```

```
    If conectado = True Then
       buff = RANKIN$ + delimeter$ + alias_gm + delimeter$ + endbuff$
       value$ = send_server(buff$)

       If (CInt(value$) = Len(buff$)) Then
          buff = ""
          value$ = recv_server(buff)
          main!sock.Action = 4
          If CInt(StrComp(Left$(value$, 2), "99")) <> 0 Then
             rank.Show
          Else
             value$ = errmsg(Left$(value$, 4))
             message value$, "ERROR", 64
          End If
       Else
          value$ = errmsg(Left$(value$, 4))
          message value$, "ERROR EN COMUNICACION", 64
       End If
    Else
    End If

End Sub

Sub registar_jugador_Click ()
    subscribe.Show
End Sub


Sub send_game_Click ()
    If alias_gm <> "" Then
       borra_gameout
    Else
       message "No se ha identificado Ud. como jugador", "USUARIO", 16
    End If
    ver
End Sub

Sub sock_OnClose ()
    conectado = False
End Sub

Sub sock_OnConnect ()
    conectado = True
End Sub

Sub sock_OnError (ErrorCode As Integer)
    If ErrorCode < 25 Then
        message "No se ha podido mantener conexión " + CStr(ErrorCode), "SOCKET ERROR",
32
    Else
       codigo$ = errmsg(CStr(ErrorCode))
       message codigo$, "SOCKET ERROR", 64
    End If
End Sub

Sub subscribe__Click ()
    subscribe.Show
    ver
```

```
End Sub

Sub unsubscribe_Click ()
   If alias_gm <> "" Then
      desuscribirse
      alias_gm = ""
   End If
   ver
End Sub

Sub user_Click ()
   borra_gameout
   borra_gamein
   borra_listplay
   passwd.Show
   ver
End Sub
```

6.2.6 Archivo module1.txt

```
Type tipoficha
    posx As Integer  ' coordenada en x actual
    posy As Integer  ' coordenada en y actual
    reina As Integer  ' 1=reina, 0=no reina
    estado As Integer ' 1=viva ,0=muerta
End Type

Global user$
Global clr$
```

6.2.7 Archivo mov_fcha.txt

```
Sub Command1_Click ()

   usr$ = List1.Text
   color% = color_contrin(usr$)
   List1.RemoveItem List1.ListIndex
   user$ = usr$
   If color% = 0 Then
      clr$ = "roja"
   Else
      clr$ = "azul"
   End If
   Form1.Show
   If List1.ListCount = 0 Then
      Unload Me
   End If

End Sub

Sub Command2_Click ()
   Unload Me
End Sub
```

```
Sub Form_Load ()
Dim ERR_DB As Database, myset As Dynaset
Dim SQLStmt As String
Const DB_READONLY = 4

   base_dat$ = PATH$ + ERR_MDB$
   Set ERR_DB = OpenDatabase(base_dat$)
   SQLStmt = "SELECT * FROM " + CONTRIN$
   Set myset = ERR_DB.CreateDynaset(SQLStmt, DB_READONLY)
   If (myset.RecordCount > 0) Then
      myset.MoveFirst
      myset.MoveLast
      j% = myset.RecordCount
      myset.MoveFirst
      For i = 1 To j% Step 1
         If myset.Fields("est") = False Then
            List1.AddItem myset.Fields("usr")
         End If
         myset.MoveNext
      Next i
      List1.Selected(0) = True
   End If

End Sub
```

6.2.8 *Archivo passwd.txt*

```
Dim value$
Dim bien As Integer

Sub Command3D1_Click ()
Dim buff$, cod$

   main!sock.Action = 1
   If conectado = True Then
      If Text1 <> "" And Text2 <> "" Then
         alias_gm = Text2
              buff$ = PASSWORD$ + delimeter$ + alias_gm + delimeter$ + Trim(Text1.Text) +
delimeter$ + endbuff$
         value$ = send_server(buff$)
         If value$ = Len(buff$) Then
            buff$ = ""
            buff$ = recv_server(buff$)
            main!sock.Action = 4
            If Left(buff$, 4) = "9910" Then
               value$ = errmsg(Left$(buff$, 4))
               message value$, "INFORMACION CORRECTA", 32
            Else
               value$ = errmsg(Left$(buff$, 4))
               message value$, "INFORMACION INCORRECTA", 16
               alias_gm = ""
            End If
         Else
            main!sock.Action = 4
            alias_gm = ""
```

```
            message "No se pudo enviar los datos al servidor", "ERROR", 64
          End If
        Else
          main!sock.Action = 4
          message "No se ha especificado la clave de acceso", "CLAVE DE ACCESO", 48
        End If
    Else
      message "No me pude conectar", "ERROR", 32
    End If
    ver
    Unload Me
End Sub

Sub Command3D2_Click ()
    Unload Me
End Sub

Sub sock_OnClose ()
    bien = False
End Sub

Sub sock_OnConnect ()
    bien = True
End Sub

Sub sock_OnError (ErrorCode As Integer)
    MsgBox "ERROR del SOCKET" + CStr(ErrorCode)
    If ErrorCode = 10 Then
      End
    End If
End Sub

Sub Text1_Change ()
    If ((keyascii = 8) Or (keyascii > 64 And keyascii < 91) Or (keyascii > 96 And keyascii < 123)
Or (keyascii > 47 And keyascii < 58)) Then
      keyascii = keyascii
    Else
      main!Panel3D1.Caption = "La tecla presionada no es valida"
      keyascii = 0
    End If

End Sub

Sub Text2_Change ()
    If ((keyascii = 8) Or (keyascii > 64 And keyascii < 91) Or (keyascii > 96 And keyascii < 123)
Or (keyascii > 47 And keyascii < 58)) Then
      keyascii = keyascii
    Else
      main!Panel3D1.Caption = "Dicha tecla no esta permitida"
      keyascii = 0
    End If

End Sub
```

6.2.9 Archivo see_inv.txt

```
Option Explicit
```

```
Dim color$
Dim buff$, C%
Dim tmp() As list_alias, IT%
Dim tmpi() As list_tmp

Sub aceptar_Click ()
Dim Value$, i%, j%
Dim tempo() As list_alias

   If C% > 0 Then

      For i = 0 To C%
         buff$ = buff$ + Trim$(tmpi(i).alias) + delimeter$ + Trim$(tmpi(i).color) + delimeter$

      Next i
      buff$ = buff$ + endbuff$

   main!sock.Action = 1
   If conectado = True Then
      buff$ = buff$ + delimeter$ + endbuff$
      If Len(buff$) > Len(ACEPINVITE$ + delimeter$ + alias_gm + delimeter$ + endbuff$) Then
         Value$ = send_server(buff$)
         IT% = 0
         If CInt(Value$) = Len(buff$) Then
            Value$ = ""
            Value$ = recv_server(Value$)
            main!sock.Action = 4
            If Left$(Value$, 4) = "9910" Then
               Value$ = errmsg(Left$(Value$, 4))
               message Value$, "RESPUESTA DEL SERVIDOR", 64
               For i% = 0 To List1.ListCount - 1
                  ReDim Preserve tempo(i + 1)
                  List1.Selected(i%) = True
                  j% = indice(CStr(List1.Text), 3)
                  tempo(i%).name = lista3(j%).name
                  tempo(i%).alias = List1.Text
               Next i%
               CNTD_I = i
               If CNTD_I > 0 Then
                  For i% = 0 To CNTD_I - 1
                     lista3(i%).name = tempo(i%).name
                     lista3(i%).alias = tempo(i%).alias
                     ReDim Preserve lista3(i% + 1)
                  Next i%
                  If List1.ListCount > 0 Then
                     List1.Selected(0) = True
                  End If
               End If'fin CTD_I
               i = List2.ListCount
               For j = 0 To i - 1
                  List2.Selected(0) = True
                  List2.RemoveItem List2.ListIndex
               Next j
               C% = 0
            Else
               Value$ = errmsg(Value$)
               message Value$, "ERROR EN COMUNICACIONES", 16
```

```
            End If 'fin left$(value$,4)
          Else
          End If'de cint(value$)
        Else
          Value$ = errmsg(Value$)
          message "No se ha seleccionado contrincante alguno", "ERROR EN DEFINICION", 64
        End If 'fin no seleccionado contrincante
    Else
    End If 'fin conectado
    If CNTD_I = 0 Then
      main!Panel3D1.Caption = ""
      Unload Me
    End If
    main!Panel3D1.Caption = ""
    Else
      message "No se ha selecionado ningun contrincante", "ERROR SELECCION", 64
    End If
End Sub

Sub aceptar_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
    main!comentario.Caption = "Aceptar las invitaciones selecionadas"
End Sub

Sub anadir_Click ()

    If List1.ListCount > 0 Then
      If color$ = "00" Then
        buff$ = buff$ + delimeter + Trim$(List1.Text) + delimeter + color$
      Else
        buff$ = buff$ + delimeter + Trim$(List1.Text) + delimeter + color$
      End If
      borra_lstplayer Trim$(List1.Text)
      IT% = IT% + 1
      ReDim Preserve tmp(IT%)
      tmp(IT%).alias = Trim$(List1.Text)
      tmp(IT%).name = Trim$(name_clt.Caption)
      List1.RemoveItem List1.ListIndex
      If List1.ListCount > 0 Then
        List1.Selected(0) = True
      End If
    Else
      message "No existen más jugadores", "ERROR ELECCION", 16
    End If

End Sub


Sub anadir_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
    main!comentario.Caption = "Añadir una invitación a la lista de aceptaciones"
End Sub

Sub cancel_Click ()
Dim i%

    ver
    For i% = 1 To IT%
      graba_lstplayer tmp(i%).alias, tmp(i%).name
    Next i%
```

```
    Unload Me

End Sub

Sub cancel_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
    main!comentario.Caption = "Abandonar la opción"
End Sub

Sub Command1_Click ()

End Sub

Sub Command3_Click ()
    Unload Me
End Sub

Sub Form_Load ()
Dim i, j As Integer
C% = 0

    act_lista3
    For i = 0 To CNTD_I% - 1
        List1.AddItem lista3(i).alias
    Next i
    List1.Selected(0) = True
    i = indice(CStr(List1.Text), 3)
    name_clt.Caption = UCase$(lista3(i).name)
    CNTD = i
    color$ = "10"
    buff$ = ""
    buff$ = ACEPINVITE$ + delimeter + alias_gm + delimeter$
    Option3D1.Value = True

End Sub

Sub Form_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
    main!comentario.Caption = "Aceptar Invitaciones de otros Subscriptores"
End Sub

Sub Frame3D1_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
    main!comentario = "Elija el color con el que desea jugar"
End Sub

Sub Label1_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
    main!comentario.Caption = ""
End Sub

Sub Label2_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
    main!comentario.Caption = ""
End Sub

Sub List1_Click ()
Dim i As Integer
Dim alia_a As String

    alia_a = List1.Text
    i = indice(alia_a, 3)
    name_clt.Caption = UCase$(lista3(i).name)
```

```
End Sub

Sub List1_DblClick ()
Dim alia_a$
Dim i%
   alia_a$ = List1.Text
   i% = indice(alia_a$, 3)
   List1.RemoveItem List1.ListIndex
   List2.AddItem lista3(i%).alias
   ReDim Preserve tmpi(C% + 1)
   tmpi(C%).alias = lista3(i%).alias
   tmpi(C%).name = lista3(i%).name
   tmpi(C%).color = color$
   C% = C% + 1
End Sub

Sub List1_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
   main!comentario.Caption = "Lista invitaciones recibidas"
End Sub

Sub List2_Click ()
Dim i As Integer
Dim alia_a As String

   alia_a = List2.Text
   i = indice(alia_a, 3)
   name_clt.Caption = UCase$(lista3(i).name)

End Sub

Sub List2_DblClick ()
Dim i As Integer
Dim alia_a As String

   alia_a = List2.Text
   i = indice(alia_a, 3)
   List2.RemoveItem List2.ListIndex
   List1.AddItem lista3(i).alias
   C% = C% - 1

End Sub

Sub List2_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
   main!comentario.Caption = "Lista invitaciones ya seleccionadas"
End Sub

Sub name_clt_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
   main!comentario.Caption = "Nombre del alias de quien se recibe la invitación"
End Sub

Sub Option1_Click ()
   color$ = "10"
End Sub

Sub Option1_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
   main!comentario.Caption = "Color con que se desa jugar"
End Sub
```

```
Sub Option2_Click ()
   color$ = "00"
End Sub

Sub Option2_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
   main!comentario.Caption = "Color con el que se desea jugar"
End Sub

Sub Option3D1_Click (Value As Integer)
   color$ = "10"
End Sub

Sub Option3D2_Click (Value As Integer)
   color$ = "00"
End Sub

Sub Picture1_Click ()
   Unload Me
End Sub
```

6.2.10 Archivo suscrib.txt

```
Sub acept_Click ()
Dim file$
Dim f_large As String * 1024


   If (name_clt <> "" And alias_clt <> "") Then
   If Left$(alias_clt, 1) > "9" Or Left$(alias_clt, 1) < "0" Then
      alias_gm = alias_clt
        buff$ = SUSBCRIVE$ + delimeter$ + alias_clt + delimeter$ + name_clt + delimeter$ +
passwd + delimeter$ + endbuff$
      main!sock.Action = 1
      If conectado = True Then
         value$ = send_server(buff$)
         If (CInt(value$) = Len(buff$) And Left$(value$, 2) <> "99") Then
            buff$ = ""
            answer = recv_server(buff$)
            main!sock.Action = 4
            buff$ = answer
            buff$ = Left$(buff$, 4)
            If (buff$ = "9900") Then
               subscribe_gm = True
               list_gm = True
               get_gm = True
               ranking_gm = True
               alias_gm = alias_clt 'take tha alias from the form

            Else
               value$ = errmsg(value$)
               message value$, "ERROR", 16

            End If

         Else
         End If
      Else 'if error occurs during send_server
         value$ = errmsg(value$)
         message value$, "SOCKET", 16
      End If
      Unload Me
   Else
            message "El ALIAS del cliente no puede empezar con un simbolo numérico",
"SUSCRIBIRSE", 64
      alias_clt.Text = ""
      alias_clt.SetFocus
   End If
   Else
      If (name_clt = "" And alias_clt = "") Then
                  message "El NOMBRE del cliente y su ALIAS deben ser especificados",
"SUSCRIBIRSE", 64
         name_clt.SetFocus
      Else
         If (name_clt = "") Then
            message "El nombre del cliente debe ser especificado", "SUSCRIBIRSE", 64
            name_clt.SetFocus
```

```
        ElseIf (alias_clt = "") Then
          message "No se ha especificado el alias del cliente", "SUSCRIBIRSE", 64
          alias_clt.SetFocus
        End If
      End If
    End If

    actualiza
    ver
    main!Panel3D1.Caption = ""

End Sub

Sub acept_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
    main!comentario.Caption = "Subscribirse en el juego de DAMAS"
End Sub

Sub alias_clt_KeyPress (keyascii As Integer)

    main!Panel3D1.Caption = ""
     If ((keyascii = 8) Or (keyascii > 64 And keyascii < 91) Or (keyascii > 96 And keyascii < 123)
Or (keyascii > 47 And keyascii < 58)) Then
      keyascii = keyascii
    Else
      main!Panel3D1.Caption = "Dicha tecla no esta permitida"
      keyascii = 0
    End If

End Sub

Sub alias_clt_LostFocus ()
    main!Panel3D1.Caption = ""
End Sub

Sub alias_clt_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)

    main!comentario.Caption = "El alias del cliente es de 8 caracteres alfanuméricos"
    main!Panel3D1.Caption = ""

End Sub

Sub Alias_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
    main!comentario.Caption = "Subscribirse en el juego de DAMAS"
End Sub

Sub Command1_Click ()
    Unload Me
End Sub

Sub Command3D1_Click ()
Dim file$
Dim f_large As String * 1024


    If (name_clt <> "" And alias_clt <> "") Then
    If Left$(alias_clt, 1) > "9" Or Left$(alias_clt, 1) < "0" Then
      alias_gm = alias_clt
```

```
        buff$ = SUSBCRIVE$ + delimeter$ + alias_clt + delimeter$ + name_clt + delimeter$ +
passwd + delimeter$ + endbuff$
      main!sock.Action = 1
      If conectado = True Then
         value$ = send_server(buff$)
         If (CInt(value$) = Len(buff$) And Left$(value$, 2) <> "99") Then
            buff$ = ""
            answer = recv_server(buff$)
            main!sock.Action = 4
            buff$ = answer
            buff$ = Left$(buff$, 4)
            If (buff$ = "9900") Then
               subscribe_gm = True
               list_gm = True
               get_gm = True
               ranking_gm = True
               alias_gm = alias_clt 'take tha alias from the form

            Else
               value$ = errmsg(value$)
               message value$, "ERROR", 16

            End If

         Else
         End If
      Else 'if error occurs during send_server
         value$ = errmsg(value$)
         message value$, "SOCKET", 16
      End If
      Unload Me
   Else
            message "El ALIAS del cliente no puede empezar con un simbolo numérico",
"SUSCRIBIRSE", 64
      alias_clt.Text = ""
      alias_clt.SetFocus
   End If
   Else
      If (name_clt = "" And alias_clt = "") Then
                  message "El NOMBRE del cliente y su ALIAS deben ser especificados",
"SUSCRIBIRSE", 64
         name_clt.SetFocus
      Else
         If (name_clt = "") Then
            message "El nombre del cliente debe ser especificado", "SUSCRIBIRSE", 64
            name_clt.SetFocus
         ElseIf (alias_clt = "") Then
            message "No se ha especificado el alias del cliente", "SUSCRIBIRSE", 64
            alias_clt.SetFocus
         End If
      End If
   End If

   actualiza
   ver
   main!Panel3D1.Caption = ""

End Sub
```

```
Sub Command3D1_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
   main!comentario.Caption = "Aceptar los datos para Subscribirse en el Juego de DAMAS"
End Sub

Sub Form_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)

   main!comentario.Caption = "Subscribirse en el juego de DAMAS"

End Sub

Sub name_clt_KeyPress (keyascii As Integer)

   main!Panel3D1.Caption = ""
   If ((keyascii = 32) Or (keyascii = 8) Or (keyascii > 64 And keyascii < 91) Or (keyascii > 96 And
keyascii < 123)) Then
      If (keyascii > 96 And keyascii < 123) Then
         keyascii = Asc(UCase(Chr(keyascii)))
      End If
   Else
      main!Panel3D1.Caption = "El caracter no es valido"
      keyascii = 0
   End If

End Sub

Sub name_clt_LostFocus ()
   main!Panel3D1.Caption = ""
End Sub

Sub name_clt_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
   main!comentario.Caption = "El nombre del cliente es de 35 caracteres alfabéticos"
End Sub

Sub name_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)

   main!comentario.Caption = "Subscribirse en el juego de DAMAS"

End Sub

Sub passwd_KeyPress (keyascii As Integer)

   main!Panel3D1.Caption = ""
   If ((keyascii = 8) Or (keyascii > 64 And keyascii < 91) Or (keyascii > 96 And keyascii < 123)
Or (keyascii > 47 And keyascii < 58)) Then
      keyascii = keyascii
   Else
      main!Panel3D1.Caption = "La tecla presionada no es valida"
      keyascii = 0
   End If

End Sub

Sub passwd_LostFocus ()
   main!Panel3D1.Caption = ""
End Sub

Sub passwd_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
```

```
    main!Panel3D1.Caption = ""
    main!comentario.Caption = "El password es de máximo 8 caracteres alfanumérico "

End Sub

Sub salir_Click ()

    main!comentario.Caption = ""
    Unload Me

End Sub

Sub salir_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)

    main!comentario.Caption = "Salir de Subscribirse en el juego de DAMAS"

End Sub
```

6.2.11 Archivo tonto.txt

```
Sub Command1_Click ()
    Unload Me
End Sub

Sub Form_Load ()

    inicialization
    conectado = False
    'value$ = InputBox$(port, "puerto")
    'If value$ = "" Then
    '    End
    'End If
    'well_port = CInt(value$)
    ini
    main.Show
    Unload Me

End Sub

Sub sock_OnAccept ()

End Sub

Sub sock_OnClose ()
    MsgBox "la conexion ha sido cerrada"
End Sub

Sub sock_OnConnect ()
    MsgBox "Conectado"
End Sub

Sub sock_OnError (ErrorCode As Integer)
```

```
  If ErrorCode < 25 Then
    message "No se ha podido mantener conexión " + CStr(ErrorCode), "SOCKET ERROR", 32
    End
  Else
    codigo$ = errmsg(CStr(ErrorCode))
    message codigo$, "SOCKET ERROR", 64
  End If

End Sub
```

6.2.12 Archivo varios.txt

```
'
'Para definicion de las opciones disponibles del menu
'
Global conectado    As Integer

Global subscribe_gm As Integer  'if the person is subscribed
Global get_gm       As Integer  'if he's subscribed
Global send_gm      As Integer  'if he has a one game to send
Global kill_gm      As Integer  'if he has a game to another person
Global list_gm      As Integer  'to obtain the subscribed list
Global invite_gm    As Integer  'if he's subcribed and someone are subscribed
Global acept_gm     As Integer  'if he's subcribed
Global see_gm       As Integer  'if he has any game in his machine
Global ranking_gm   As Integer  'if thes opcion is maded
Global alias_gm     As String   'contain the alias of the player
Global mov_ficha    As Integer  'if have some games to play
Global PATH         As String   'path de directorio de damas
Global answer_gm    As Integer
'Delimitadores
Global Const delimeter$ = ";"   'delimeter betwen fields
Global Const endbuff$ = "#"     'say that the buffer is over

Global well_port    As Integer  'server's well known port
Global version_sock As Integer  'winsoket version
Global host_ip      As String   'server ip adress
Global sockfd%                  'socket descriptor

'Definicion de codigos de operaciones
Global Const SUSBCRIVE$ = "13"
Global Const LISTPLAYER$ = "01"
Global Const INVITEPLAYER$ = "02"
Global Const RECIBEINVITE$ = "03"
Global Const ACEPINVITE$ = "04"
Global Const SENDGAME$ = "05"
Global Const RECIBEGAME$ = "06"
Global Const KILLGAME$ = "07"
Global Const FINISH$ = "08"
Global Const RANKIN$ = "09"
Global Const UNSUBSCRIVE$ = "10"
Global Const GAMEINI$ = "11"
Global Const PASSWORD$ = "12"

'Definicion de tipo de fichas
```

```
Global Const NEGRODOBLE$ = "01"
Global Const NEGROSIMPLE$ = "00"

'Definicion de cantidad de datos en arreglos
Global CNTD%    'players to invite
Global CNTD_k%  'partidos a matar
Global CNTD_l%  'players who invite me


Global Const FILEINI = "damas.ini" 'archivo de inicializacion
Global Const ERR_MDB$ = "damas.mdb"
Global Const FICHASIN$ = "fichas"
Global Const CONTRIN$ = "contrin"
Global Const FICHASOUT$ = "fichas.out"
'************************************************************
'LISTA DE ALIAS y sus respectivos NOMBRES
Type list_alias
   alias   As String * 8   'playe's alias
   name    As String * 35  'player's name
End Type
Type list_tmp
   alias   As String * 8   'playe's alias
   name    As String * 35  'player's name
   color   As String * 2   'color a jugar
End Type

Global lista() As list_alias    'lista de alias subscriptos
Global lista3() As list_alias   'lista de allias who invite me
Global kil() As String * 8      'lista de alias de partidos a matar

Sub act_lista3 ()
Dim ERR_DB As Database, myset As Dynaset
Dim SQLStmt As String
Const DB_READONLY = 4

   main!Panel3D1.Caption = "La lista se esta actualizando"
   base_dat$ = PATH + ERR_MDB$
   Set ERR_DB = OpenDatabase(base_dat$)
   SQLStmt = "SELECT * FROM list_player"
   Set myset = ERR_DB.CreateDynaset(SQLStmt, DB_READONLY)
   myset.MoveLast
   j = myset.RecordCount
   If j > 0 Then
         myset.MoveFirst
         For i = 0 To j - 1
            ReDim Preserve lista3(i + 1)
            lista3(i).alias = myset.Fields("alias")
            lista3(i).name = myset.Fields("name")
            myset.MoveNext
         Next i
         CNTD_l% = j
   End If
   main!Panel3D1.Caption = ""

End Sub

'Actualiza el archivo DAMAS.INI con las opciones del menu
'asi como el numero de puerto, direccion ip del servidor,
```

```
'alias del cliente, version del winsock del cliente
'
Sub actualiza ()

End Sub

'Permite armar la lista, arreglo, dado el buffer de recepcion del servidor
'Parametros: buffer cadena de recepcion enviado por servidor,
'           j identificador de lista
'Retorna:    cantidad de elementos almacenados en lista
Function armar_lista% (buff As String, j As Integer)
Dim lng, i
Dim ch$

   i = 0
   lng = desconcadena(buff)
   ch$ = Left$(buff, lng)  'obtengo primer campo
   If j = 1 Then   'si se trata de lista de suscriptores
         While buff <> endbuff$   'mientras no sea fin del buffer
            ReDim Preserve lista(i + 1)
            lista(i).alias = Trim$(ch$) 'almaceno el alias
            buff = Mid$(buff, lng + 2, Len(buff) - lng + 2)
            lng = desconcadena(buff)
            ch$ = Trim$(Left$(buff, lng))
            lista(i).name = ch$ 'asigno el nombre
            buff = Mid$(buff, lng + 2, Len(buff) - lng + 2)
            lng = desconcadena(buff)
            ch$ = Trim$(Left$(buff, lng))
            i = i + 1
            ReDim Preserve lista(i + 1)
         Wend
   ElseIf j = 3 Then   'si se trata de lista de invitaciones recibidas
         While buff <> endbuff$
            ReDim Preserve lista3(i + 1)
            lista3(i).alias = Trim$(ch$)
            buff = Mid$(buff, lng + 2, Len(buff) - lng + 2)
            lng = desconcadena(buff)
            ch$ = Trim$(Left$(buff, lng))
            lista3(i).name = Trim$(ch$)
            buff = Mid$(buff, lng + 2, Len(buff) - lng + 2)
            lng = desconcadena(buff)
            ch$ = Trim$(Left$(buff, lng))
            graba_lstplayer lista3(i).alias, lista3(i).name
            i = i + 1
            ReDim Preserve lista3(i + 1)
         Wend
   ElseIf j = 11 Then  'si se trata de partidos a matar
         While buff <> endbuff$
            ReDim Preserve kil(i + 1)
            kil(i) = Trim$(ch$)
            buff = Mid$(buff, lng + 2, Len(buff) - lng + 2)
            lng = desconcadena(buff)
            ch$ = Trim$(Left$(buff, lng))
            i = i + 1
            ReDim Preserve kil(i + 1)
         Wend
   End If
   armar_lista% = i
```

```
End Function

Sub borra_contrin (usr$)
Dim ERR_DB As Database, myset As Dynaset
Dim SQLStmt As String
Const DB_CONSISTENT = 32

   base_dat$ = PATH + ERR_MDB$
   Set ERR_DB = OpenDatabase(base_dat$)
   SQLStmt = "SELECT * FROM " + CONTRIN$ + " WHERE usr='" + usr$ + "'"
   Set myset = ERR_DB.CreateDynaset(SQLStmt, DB_CONSISTENT)
   If (myset.RecordCount > 0) Then
         j% = myset.RecordCount
         myset.MoveFirst
         For i = 1 To j% Step 1
            myset.Delete
         Next i
   End If

End Sub

Sub borra_contrinin (usr$)
Dim ERR_DB As Database, myset As Dynaset
Dim SQLStmt As String
Const DB_CONSISTENT = 32

   base_dat$ = PATH + ERR_MDB$
   Set ERR_DB = OpenDatabase(base_dat$)
   SQLStmt = "SELECT * FROM " + CONTRIN$ + " WHERE usr='" + usr$ + "'"
   Set myset = ERR_DB.CreateDynaset(SQLStmt, DB_CONSISTENT)
   If (myset.RecordCount > 0) Then
         j% = myset.RecordCount
         myset.MoveFirst
         For i = 1 To j% Step 1
            myset.Fields("est") = 1
            myset.Update
         Next i
   End If

End Sub

Sub borra_gamein ()
Dim ERR_DB As Database, ERR_DBB As Database, myset As Dynaset, mset As Dynaset
Dim SQLStmt As String, buff$
Const DB_CONSISTENT = 32

   main!Panel3D1.Caption = "Los datos de las jugadas se estan enviando"
   base_dat$ = PATH + ERR_MDB
   Set ERR_DB = OpenDatabase(base_dat$)
   SQLStmt = "SELECT * FROM fichas"
   Set myset = ERR_DB.CreateDynaset(SQLStmt, DB_CONSISTENT)
   j = myset.RecordCount
   buff = ""
   If j > 0 Then
         myset.MoveLast
         j = myset.RecordCount
         myset.Edit
```

```
            usr$ = ""
            For i = 1 To j Step 1
               myset.MoveFirst
               If usr$ <> myset.Fields("usuario") And buff$ <> "" Then
                     buff$ = buff$ + endbuff$
                     main!sock.Action = 1
                     If conectado = True Then
                        value$ = send_server(buff$)
                        main!sock.Action = 4
                     Else
                     End If
                     usr$ = myset.Fields("usuario")
                     buff$ = SENDGAME$ + delimeter$ + delimeter$ + alias_gm + usr$ + delimeter$
                     Set ERR_DBB = OpenDatabase(base_dat$)
                     SQLStmt = "SELECT color FROM contrin WHERE usr='" + usr$ + "'"
                     Set mset = ERR_DBB.CreateDynaset(SQLStmt, DB_CONSISTENT)
                     If mset.Fields("color") = 0 Then
                        buff$ = buff$ + "1" + delimeter$
                     Else
                        buff$ = buff$ + "0" + delimeter$
                     End If
                     borra_contrin (usr$)
               End If
               eq% = myset.Fields("equipo")
               If eq% = 1 Then
                     buff$ = buff$ + "0" + CStr(myset.Fields("reina")) + delimeter$
               Else
                     buff$ = buff$ + "1" + CStr(myset.Fields("reina")) + delimeter$
               End If
               buff$ = buff$ + CStr(myset.Fields("posx")) + delimeter$
               buff$ = buff$ + CStr(myset.Fields("posy")) + delimeter$
               myset.Delete
            Next i
            buff$ = buff$ + endbuff$
            main!sock.Action = 1
            If conectado = True Then
               value$ = send_server(buff$)
               main!sock.Action = 4
            Else
            End If
      End If
      main!Panel3D1.Caption = ""

End Sub

Sub borra_gameout ()
Dim ERR_DB As Database, ERR_DBB As Database, myset As Dynaset, mset As Dynaset
Dim SQLStmt As String, buff$
Const DB_CONSISTENT = 32

      main!Panel3D1.Caption = "Los datos de las jugadas se estan enviando"
      base_dat$ = PATH + ERR_MDB
      Set ERR_DB = OpenDatabase(base_dat$)
      SQLStmt = "SELECT * FROM fichas_out"
      Set myset = ERR_DB.CreateDynaset(SQLStmt, DB_CONSISTENT)
      j = myset.RecordCount
      buff = ""
      If j > 0 Then
```

```
        myset.MoveLast
        j = myset.RecordCount
        myset.Edit
        usr$ = ""
        For i = 1 To j Step 1
           myset.MoveFirst
           If usr$ <> myset.Fields("usuario") And buff$ <> "" Then
                 buff$ = buff$ + endbuff$
                 main!sock.Action = 1
                 If conectado = True Then
                    value$ = send_server(buff$)
                    main!sock.Action = 4
                 Else
                 End If
                 usr$ = myset.Fields("usuario")
                 buff$ = SENDGAME$ + delimeter$ + usr$ + delimeter$ + alias_gm + delimeter$
                 Set ERR_DBB = OpenDatabase("c:\damas\damas.mdb")
                 SQLStmt = "SELECT color FROM contrin WHERE usr='" + usr$ + "'"
                 Set mset = ERR_DBB.CreateDynaset(SQLStmt, DB_CONSISTENT)
                 If mset.Fields("color") = 0 Then
                    buff$ = buff$ + "1" + delimeter$
                 Else
                    buff$ = buff$ + "0" + delimeter$
                 End If
                 borra_contrin (usr$)
           End If
           eq% = myset.Fields("equipo")
           If eq% = 1 Then
                 buff$ = buff$ + "0" + CStr(myset.Fields("reina")) + delimeter$
           Else
                 buff$ = buff$ + "1" + CStr(myset.Fields("reina")) + delimeter$
           End If
           buff$ = buff$ + CStr(myset.Fields("posx")) + delimeter$
           buff$ = buff$ + CStr(myset.Fields("posy")) + delimeter$
           myset.Delete
        Next i
        buff$ = buff$ + endbuff$
        main!sock.Action = 1
        If conectado = True Then
           value$ = send_server(buff$)
           main!sock.Action = 4
        Else
        End If
     End If
     main!Panel3D1.Caption = ""
End Sub

Sub borra_listplay ()
Dim ERR_DB As Database, myset As Dynaset
Dim SQLStmt As String
Const DB_CONSISTENT = 32

     main!Panel3D1.Caption = "Los jugadores estan siendo borrados"
     base_dat$ = PATH + ERR_MDB$
     Set ERR_DB = OpenDatabase(base_dat$)
     SQLStmt = "SELECT * FROM list_player"
     Set myset = ERR_DB.CreateDynaset(SQLStmt, DB_CONSISTENT)
     j = myset.RecordCount
```

```
  If j > 0 Then
        myset.MoveFirst
        myset.MoveLast
        j = myset.RecordCount
        For i = 1 To j
           myset.MoveFirst
           myset.Delete
        Next i
  End If
  main!Panel3D1.Caption = ""


End Sub

Sub borra_lstplayer (alia$)
Dim ERR_DB As Database, myset As Dynaset
Dim SQLStmt As String
Const DB_CONSISTENT = 32

   main!Panel3D1.Caption = "Los jugadores estan siendo borrados"
   base_dat$ = PATH + ERR_MDB$
   Set ERR_DB = OpenDatabase(base_dat$)
   SQLStmt = "SELECT * FROM list_player WHERE alias='" + alia$ + "'"
   Set myset = ERR_DB.CreateDynaset(SQLStmt, DB_CONSISTENT)
   j = myset.RecordCount
   If j > 0 Then
        myset.Delete
   End If
   main!Panel3D1.Caption = ""

End Sub

Function color_contrin% (usr$)
Dim ERR_DB As Database, myset As Dynaset
Dim SQLStmt As String
Const DB_READONLY = 4

   base_dat$ = PATH + ERR_MDB$
   Set ERR_DB = OpenDatabase(base_dat$)
   SQLStmt = "SELECT color FROM " + CONTRIN$ + " WHERE usr='" + usr$ + "'"
   Set myset = ERR_DB.CreateDynaset(SQLStmt, DB_READONLY)
   color_contrin% = myset.Fields("color")

End Function

Sub cuenta_lstplayers ()
Dim ERR_DB As Database, myset As Dynaset
Dim SQLStmt As String
Const DB_READONLY = 32

   base_dat$ = PATH + ERR_MDB$
   Set ERR_DB = OpenDatabase(base_dat$)
   SQLStmt = "SELECT * FROM list_player"
   Set myset = ERR_DB.CreateDynaset(SQLStmt, DB_READONLY)
   If myset.RecordCount > 0 Then
        answer_gm = True
        act_lista3
   Else
```

```
        answer_gm = False
    End If

End Sub

Sub cuenta_partidosin ()
Dim ERR_DB As Database, myset As Dynaset
Dim SQLStmt As String
Const DB_READONLY = 4

    base_dat$ = PATH + ERR_MDB$
    Set ERR_DB = OpenDatabase(base_dat$)
    SQLStmt = "SELECT * FROM " + CONTRIN$ + " WHERE est=0"
    Set myset = ERR_DB.CreateDynaset(SQLStmt, DB_READONLY)
    If myset.RecordCount > 0 Then
        mov_ficha = True
    Else
        mov_ficha = False
    End If

End Sub

Sub cuenta_partidosout ()
Dim ERR_DB As Database, myset As Dynaset
Dim SQLStmt As String
Const DB_READONLY = 4
    i = Len(PATH) - 1
    base_dat$ = Left$(PATH, i) + "2\" + ERR_MDB$
    Set ERR_DB = OpenDatabase(base_dat$)
    SQLStmt = "SELECT * FROM fichas"
    Set myset = ERR_DB.CreateDynaset(SQLStmt, DB_READONLY)
    If myset.RecordCount > 0 Then
        send_gm = True
    Else
        send_gm = False
    End If

End Sub

'Permite obtener la longitud del primer campo antes del delimitador de campo
'Parametro: buff cadena enviada por servidor
'Retorna:   longitud del campo antes del delimitador
'
Function desconcadena% (buff As String)
Dim ch As String * 1
Dim i As Integer

    ch$ = "a"
    i = 0
    'mientras no sea delimitador de campo o fin de buffer
    While ch$ <> delimeter$ And ch$ <> endbuff$
        ch$ = Right$(Left$(buff, i), 1)
        'incrementa longitud
        i = i + 1
    Wend
    If i <> 0 Then
        'si existe algun campo
        desconcadena% = i - 2
```

```
  Else
        'si no existe campo en buff
        desconcadena% = 0
  End If

End Function

Sub desuscribirse ()
Dim buff As String * 20, alia  As String

  answer = MsgBox("Está seguro que desea retirarse de DAMAS", 36, "DESUSCRIBIRSE")

  If (answer = 6) Then 'if the answer is "YES"
        buff = UNSUBSCRIVE$ + delimeter$ + alias_gm + delimeter$ + endbuff$
        main!sock.Action = 1
        If conectado = True Then
           value$ = send_server(buff)
           If (CInt(value$) = Len(buff)) Then
                 buff = ""
                 value$ = recv_server(buff)
                 main!sock.Action = 4
                   buff = Left$(value$, 4)
                   If (buff <> "9910") Then
                         message "UD. ha sido eliminado de la lista de DAMAS...", "ESTADO
EXITO", 48
                         subscribe_gm = False
                         get_gm = False
                         send_gm = False
                         kill_gm = False
                         list_gm = False
                         invite_gm = False
                         acept_gm = False
                         see_gm = False
                         ranking_gm = False
                         alias_gm = ""
                         actualiza
                         main!invitar_jugador.Visible = False
                   Else
                         message "No ha sido posible eliminarlo de la lista de DAMAS...",
"ESTADO ERROR", 48
                   End If
                 Else
                   value$ = errmsg(value$)
                   message value$, "SOCKET ERROR", 48
                 End If
           Else
           End If
  Else
        message "No se ha podido conectar", "CONECTARSE", 16
  End If
  ver

End Sub

'Permite obtener de la base DAMAS.MDB y tabla error_code
'la descripcion del error dado el codigo
'Parametro: cod, del tipo string, codigo de error
'Retorna:   errmsg, string, contiene descripcion del codigo
```

```
'
Static Function errmsg$ (cod$)
Dim ERR_DB As Database
Dim cod_err As Table

   base_dat$ = PATH + ERR_MDB$
   Set ERR_DB = OpenDatabase(base_dat$)
   Set cod_err = ERR_DB.OpenTable("error_code")
   cod_err.Index = "error_idx"
   cod_err.Seek "=", Trim$(cod$)
   If cod_err.NoMatch Then
         errmsg$ = cod$
   Else
         cod_err.Edit
         cod$ = cod_err("descrpcion")
         errmsg$ = cod$
   End If
   cod_err.Close
   ERR_DB.Close

End Function

Sub graba_contrin (usr$, color%)
Dim ERR_DB As Database, myset As Dynaset
Dim SQLStmt As String
Const DB_CONSISTENT = 32

   base_dat$ = PATH + ERR_MDB$
   Set ERR_DB = OpenDatabase(base_dat$)
   SQLStmt = "SELECT * FROM contrin WHERE usr='" + usr$ + "'"
   Set myset = ERR_DB.CreateDynaset(SQLStmt, DB_CONSISTENT)
   If myset.RecordCount = 0 Then
         myset.AddNew
         myset.Fields("usr") = usr$
         myset.Fields("color") = color%
         myset.Fields("est") = 0
         myset.Update
   End If

End Sub

Sub graba_fichasin (buff$)
Dim ERR_DB As Database, myset As Dynaset
Dim SQLStmt As String
Const DB_CONSISTENT = 32
Dim lng As Integer, ch$

   lng = desconcadena(buff)
   ch$ = Left$(buff, lng)  'obtengo primer campo
   usr$ = Trim$(ch$) 'almaceno el alias

   base_dat$ = PATH + ERR_MDB$
   Set ERR_DB = OpenDatabase(base_dat$)
   SQLStmt = "SELECT * FROM " + FICHASIN$ + " WHERE usuario='" + usr$ + "'"
   Set myset = ERR_DB.CreateDynaset(SQLStmt, DB_CONSISTENT)
   If (myset.RecordCount > 0) Then
         j% = myset.RecordCount
         myset.MoveFirst
```

```
        For i = 1 To j% Step 1
           myset.Delete
        Next i
    End If

    buff$ = Mid$(buff$, lng + 2, Len(buff$) - lng + 2)
    lng = desconcadena(buff$)
    ch$ = Trim$(Left$(buff$, lng))
    color% = CInt(ch$)
    buff$ = Mid$(buff$, lng + 2, Len(buff$) - lng + 2)
    While buff$ <> "#"
        lng = desconcadena(buff$)
        ch$ = Trim$(Left$(buff, lng))
        fcha$ = ch$
        If CInt(Left$(fcha$, 1)) = 0 Then 'azul
           eq% = 1
        Else
           eq% = 2
        End If
        r% = CInt(Right$(fcha$, 1))
        buff$ = Mid$(buff$, lng + 2, Len(buff$) - lng + 2)
        lng = desconcadena(buff$)
        ch$ = Trim$(Left$(buff, lng))
        x% = CInt(ch$)
        buff$ = Mid$(buff$, lng + 2, Len(buff$) - lng + 2)
        lng = desconcadena(buff$)
        ch$ = Trim$(Left$(buff, lng))
        y% = CInt(ch$)
        buff$ = Mid$(buff$, lng + 2, Len(buff$) - lng + 2)
        myset.AddNew
        myset.Fields("usuario") = usr$
        myset.Fields("equipo") = eq%
        myset.Fields("reina") = r%
        myset.Fields("posx") = x%
        myset.Fields("posy") = y%
        myset.Update

    Wend
    graba_contrin usr$, color%


End Sub

Sub graba_lstplayer (alia$, nombre$)
Dim ERR_DB As Database, myset As Dynaset
Dim SQLStmt As String
Const DB_CONSISTENT = 32

    base_dat$ = PATH + ERR_MDB$
    Set ERR_DB = OpenDatabase(base_dat$)
    SQLStmt = "SELECT * FROM list_player WHERE alias='" + alia$ + "'"
    Set myset = ERR_DB.CreateDynaset(SQLStmt, DB_CONSISTENT)
    If myset.RecordCount = 0 Then
        myset.AddNew
        myset.Fields("alias") = alia$
        myset.Fields("name") = nombre$
        myset.Update
    End If
```

```
End Sub

'Permite obtener el indice de un arreglo o lista dado el alias
'Parametros:    alia, del tipo string, contiene al alis a buscar
'              j, tipo de lista en la que se buscara
'Retorna:   indice, numero de indice en lista o arreglo
'
Function indice (alia As String, j As Integer) As Integer

   If j = 1 Then
         For i = 0 To CNTD
            If lista(i).alias = alia Then   'si se trata del alia a buscar
                  'asigno el numero del indice
                  indice = i
                  Exit Function
            End If
         Next i
    ElseIf j = 3 Then
         For i = 0 To CNTD_I
            If lista3(i).alias = alia Then  'si se trate de alia a buscar
                  'asigno indice
                  indice = i
                  Exit Function
            End If
         Next i
    End If

End Function

Sub ini ()

    main!sock.HostName = host_ip
    main!sock.RemotePort = well_port
    main!sock.HostConvert = 2   'convert_name to address pag.1-21
    main!sock.Protocol = 0 'protocol_tcp

End Sub


Sub inicialization ()
Dim file$, buff  As String

On Error GoTo error_ini

    fnun = 1
    file$ = "c:\windows\" + FILEINI
    Open file$ For Input Access Read As #fnun
    Do While Not EOF(fnun)
          Line Input #fnun, buff
          Select Case Left$(Trim$(buff), 6)
             Case "path_ "
                   i = Len(buff)
                   PATH = Right$(buff, i - 7)
             Case "w_port"
                   i = Len(buff)
                   well_port = CInt(Right$(buff, i - 7))
             Case "ip_add"
```

```
                i = Len(buff)
                host_ip = Right$(buff, i - 7)

        End Select
    Loop
    Close '#fnum
    Exit Sub

error_ini:
    message Error$, "INICIALIZACION", 16
    Exit Sub
End Sub


Sub matar_gm ()
Dim value$

    main!sock.Action = 1
    If conectado = True Then
        buff$ = GAMEINI$ + delimeter$ + alias_gm + delimeter$ + endbuff$
        value$ = send_server(buff$)
        If CInt(value$) = Len(buff$) Then
            buff$ = ""
            For j = 0 To 10
            value$ = recv_server(buff$)
                If value$ <> "" Then
                    j = 10
                End If
            Next j
            main!sock.Action = 4
            If value$ <> "" Then
                If Left$(value$, 2) <> "99" Then
                    CNTD_k% = armar_lista(value$, 11)
                    matar.Show
                Else
                    value$ = errmsg(value$)
                    message value$, "ERROR KILL RECV", 64
                End If
            Else
                message "NO SE RECIBIO NADA", "ERROR", 64
            End If
        Else
            value$ = errmsg(value$)
            message value$, "ERROR KILL SEND", 64
        End If
    Else
        message "No se pudo mantener conexion", "CONEXION", 64
    End If

End Sub

'Permite mostrar en pantalla un mensaje
'Parametros: buff, descripcion del mensaje
'           title, titulo de la ventana
'           class, tipo de mensaje o icono a presentar
'
Sub message (buff As String, title As String, class As Integer)
```

```
    MsgBox buff, class, title

End Sub



Sub obt_lista ()
Dim buff$, value$
   buff$ = ""
   main!sock.Action = 1
   If conectado = True Then
         buff$ = LISTPLAYER$ + delimeter$ + alias_gm + delimeter$ + endbuff$
         value$ = send_server(buff$)
         If (CInt(value$) <> Len(buff$)) Then
            message "No se ha podido obtener la lista", "LISTA2", 16
         Else
            buff = ""
            value$ = recv_server(buff$)
            main!sock.Action = 4
            buff$ = Left$(value$, 2)
            If buff$ <> "99" Then
                  acept_gm = True
                  CNTD% = armar_lista(value$, 1)
                  invite_gm = True
            Else
                  value$ = Left$(value$, 4)
                  value$ = errmsg(value$)
                  message value$, "Obtener Lista Jugadores1", 64
            End If
         End If
   Else
         message "TONTO", "ERROR", 16
   End If
   ver
   main!Panel3D1.Caption = ""

End Sub


Sub recv_inv ()
Dim buff$, value$, j As Integer

   main!sock.Action = 1
   If conectado = True Then
         buff$ = RECIBEINVITE$ + delimeter$ + alias_gm + delimeter$ + endbuff$
         value$ = send_server(buff$)
         If CInt(value$) = Len(buff$) Then
            buff$ = ""
            value$ = recv_server(buff$)
            main!sock.Action = 4
            If Left$(value$, 2) <> "99" Then
                  acept_gm = True
                  CNTD_I% = armar_lista(value$, 3)
            Else
                  value$ = errmsg(Left$(value$, 4))
                  message value$, "ERROR RECIBE INVITACIONES STATUS", 16
```

```
                acept_gm = False
            End If
        Else
        value$ = errmsg(Left$(value$, 4))
        message value$, "ERROR RECIBE INVITE SEND", 32
        End If
    End If
    actualiza
    ver
    main!Panel3D1.Caption = ""

End Sub

Function recv_server$ (ByVal buff As String)
Dim temp As String, j%

On Error GoTo error_recv_server

    main!Panel3D1.Caption = "Se estan recibiendo los datos del servidor"
    main!sock.ReceiveLen = 100
    While InStr(temp, endbuff$) = 0 And j <> 2000
        rval = main!sock.ReceiveCount
        If rval > 0 Then
            tem = main!sock.Receive
            buff$ = Trim(buff$) + Trim(tem)
            temp = Right(tem, 1)
            recv_server = "OK"
            j = 0
        End If
        j = j + 1
    Wend
    If j <> 2000 Then
    recv_server = buff$
    Else
    recv_server = "9909" + delimeter$ + endbuff$
    End If
    main!Panel3D1.Caption = ""
    Exit Function

error_recv_server:
    recv_server = "9911"
End Function


Sub send_gameout ()
Dim ERR_DB As Database, myset As Dynaset
Dim SQLStmt$, buffer$, alias1$, alias2$, alia_tmp$, posx$, posy$, fcha$
Const DB_READONLY = 4
Const DB_CONSISTENT = 32

    base_dat$ = PATH + ERR_MDB$
    Set ERR_DB = OpenDatabase(base_dat$)
    SQLStmt$ = "SELECT * FROM error_code"
    Set myset = ERR_DB.CreateDynaset(SQLStmt$, DB_READONLY)
    myset.Edit
    myset.MoveFirst

    myset.Fields("err_code") = "error"
```

```
    myset.Fields("descrpcion") = "temporalmente un error"

    myset.Update

End Sub

Function send_server$ (ByVal buff As String) 'As String
Dim value As Integer

    main!Panel3D1.Caption = "Espere un momento mientras se transmiten los datos al servidor"
    If Len(buff) > 1024 Then
        main!sock.SendSize = Len(buff)
    End If
    main!sock.Send = buff
    value = main!sock.Result
    send_server$ = CStr(value)

End Function


Sub ver ()

    main!subscribe_.Enabled = False
    If alias_gm <> "" Then
            main!Command3D2.Enabled = True
            main!Command3D4.Enabled = True
            main!Command3D5.Enabled = True
            main!ami.Enabled = True
            main!get_game.Enabled = True
            main!send_game.Enabled = True
            main!unsubscribe.Enabled = True
            main!matado.Enabled = True
            main!invitar_jugador.Enabled = True
            main!answer_invite.Enabled = True
    Else
            main!Command3D2.Enabled = False
            main!Command3D4.Enabled = False
            main!Command3D5.Enabled = False
            main!ami.Enabled = False
            main!get_game.Enabled = False
            main!send_game.Enabled = False
            main!unsubscribe.Enabled = False
            main!invitar_jugador.Enabled = False
            main!answer_invite.Enabled = False
            main!matado.Enabled = False
            'main!subscribe_.Enabled = False
    End If
    main!subscribe_.Enabled = True

End Sub
```