Gabriela Suarez Carvajal Código: 28791162

Lizzy Tengana Hurtado Código: 28791177 18 de abril de 2017

**Práctica 2**

**Documentación**

En la práctica 2 se solicitó hacer un programa para una veterinaria implementando el paradigma Cliente-Servidor. Ya que en la práctica 1 se elaboró el programa que contenía las funciones principales de la veterinaria, la práctica 2 consistió en transformar dichos archivos para hacer posible la manipulación de las funciones desde máquinas remotas.

A continuación se detalla la elaboración y el funcionamiento de los programas de la práctica 2 de Sistemas Operativos: p2-dogServer.c, p2-dogClient.c y Makefile.

1. **Manual de uso y especificaciones**
   1. **Servidor:** el programa p2-dogServer.c es programa que corre sobre la máquina que hace las veces de servidor, se encarga de recibir las comunicaciones de clientes remotos a través de sockets y procesar las solicitudes de ingresar perro, ver perro, borrar perro y buscar perro.

Toda la información de los perros se guarda en el archivo dataDogs.dat y cada vez que un cliente hace una de las cuatro solicitudes disponibles, el servidor guarda información sobre la operación en el archivo serverDogs.log.

Lo primero que hace el servidor es configurar un socket con la función listen para que esté listo para atender conexiones entrantes; una vez entre una conexión, el programa crea un nuevo socket para atender al cliente por medio de un nuevo hilo. En el hilo se encuentran las cuatro funciones principales del programa que se disparan a petición del cliente:

* Ingresar perro: cuando el cliente selecciona la opción 1 desde su interfaz, el servidor recibe una estructura dogType y la envía a la función ingresar(struct dogType \*perro), la cual calcula el hash con el nombre del perro y lo guarda en dataDogs.dat con las referencias correctas.
* Ver perro: cuando el cliente envía la opción 2, el servidor le envía el número de registros existentes, el cliente devuelve un número con su elección, el servidor llama a la función ver(int v, struct dogType \*perro) la cual abre un archivo con la historia clínica del perro o la crea si no existe. Luego el servidor le envía el archivo al cliente, el cliente lo modifica y lo envía de vuelta para que sea guardado en la máquina del servidor.
* Borrar perro: cuando el cliente envía la opción 3, el servidor le envía el número de registros existentes, el cliente devuelve un número con su elección, el servidor llama a la función borrar(int v) y borra el registro seleccionado modificando las referencias hacia él en la tabla hash y el archivo dataDogs.dat. Finalmente le envía al cliente una bandera indicando si el borrado fue exitoso o no.
* Buscar perro: cuando el cliente envía la opción 4, el servidor recibe una cadena de caracteres con el nombre del perro que se desea buscar y llama a la función buscar(char nombre[32], int positions[1000] ), la cual calcula el número de registros existentes que coinciden con la cadena enviada, entonces el servidor los envía uno por uno al cliente.
  1. **Cliente:** el programa p2-dogClient.c es el programa que corren los clientes para comunicarse con el servidor a través de sockets. La función principal del programa es mostrar un menú para que el cliente seleccione la solicitud que desea hacerle al servidor y luego mostrar la respuesta correspondiente. Las solicitudes pueden ser:
* Ingresar perro: cuando el usuario selecciona la opción 1, el cliente crea una estructura dogType y le pide al usuario que ingrese los campos necesarios y finalmente le envía dicha estructura al servidor para que éste la ingrese al registro de perros de la veterinaria.
* Ver perro: cuando el usuario selecciona la opción 2, el cliente se la envía al servidor y recibe a cambio el número total de estructuras existentes. El usuario selecciona el número de la estructura que quiere ver, el cliente le envía ese número al servidor y recibe una estructura dogType con el perro solicitado que finalmente imprime en pantalla.
* Borrar perro: cuando el usuario selecciona la opción 3, el cliente se la envía al servidor y recibe a cambio el número total de estructuras existentes. El usuario selecciona el número de la estructura que quiere borrar, el cliente le envía ese número al servidor y recibe una bandera indicando si el borrado fue exitoso o no y se lo notifica al usuario en pantalla.
* Buscar perro: cuando el usuario selecciona la opción 4, el cliente le pide ingresar una cadena de caracteres con el nombre del perro que desea buscar, el cliente se la envía al servidor y éste le envía el número de estructuras que coincidieron con la búsqueda. Luego el cliente se prepara para recibir dicho número de estructuras y las va imprimiendo en pantalla a medida que llegan.

1. **Informe de elaboración**

* **Funciones p2-dogServer.c**
  + void imprimir(struct dogType \*perro): imprime todos los atributos de una estructura dogType.
  + unsigned int funcion\_hash(char \*s): calcula el hash del nombre de los perros para saber en qué lista de la tabla hash ubicarlos.
  + void initNumRegistro(): calcula el número de registros existentes en dataDogs.dat.
  + void sobreescribir(): reduce el tamaño del archivo dataDogs.dat en exáctamente el tamaño de una estructura dogtype.
  + void ingresar(struct dogType \*perro): llama a la función hash, busca la ubicación del último perro de la lista de ese hash en dataDogs e ingresa el perro al final de dataDogs actualizando los índices de la lista de ese hash.
  + int ver(int v, struct dogType \*perro): mira que v esté entre el número de perros existentes y retorna un flag = true si el perro existe o flag = false si el perro no existe.
  + int borrar(int v): borra el perro en la posición v del archivo dataDogs y actualiza los índices de la lista de su hash.
  + int buscar(char nombre[32], int positions[1000] ): llama a la función hash y retorna todos los perros que pertenecen a esa lista hash.
  + int registro\_serverDogs(char IP[INET\_ADDRSTRLEN], int opcion, int reg, char cadena[32]): guarda en serverLog.log los datos del cliente que se conectó y de su consulta.
  + int \*nuevaConexion(void \*thrData): contiene las opciones que puede escoger un cliente: ingresar perro, ver perro, borrar perro y buscar perro.
  + int main(): configura el socket del servidor, recibe las conexiones de los clientes y crea un hilo para cada una.
* **Funciones p2-dogClient.c**
  + void imprimir(struct dogType \*perro): imprime todos los atributos de una estructura dogType.
  + int press\_any\_key( void ): para el programa hasta que el usuario presione una tecla.
  + int main(): contiene las opciones que puede escoger un cliente: ingresar perro, ver perro, borrar perro y buscar perro.