Manual de usuario

**Moisés Bernardo Suárez Gámez**

**Andrés Esteban Romero**

**Cesar Pedraza Bonilla**

Universidad Nacional de Colombia

Facultad de Ingeniería, Departamento de Sistemas y Computación

Bogotá D.C., Colombia

20 de Noviembre de 2019

**Contenido**

Pág.

1. **Introducción………………………………………………………………………………….. x**
   1. **Descripción del programa……………………………………………………………. x**
      1. **Funcionalidad…………………………………………………………………... x**
      2. **Objetivos………………………………………………………………………… x**
   2. **Novedades……………………………………………………………………………… x**
2. **Requisitos previos…………………………………………………………………………... x**
   1. **Conocimiento mínimo del usuario………………………………………………….. x**
   2. **Requisitos técnicos……………………………………………………………………. x**
      1. **Capacidades técnicas mínimas del equipo………………………………… x**
      2. **Software necesario…………………………………………………………….. x**
   3. **Mecanismo para acceder al sistema………………………………………………… x**
3. **Instalación y configuración………………………………………………………………... x**
4. **Utilización del programa…………………………………………………………………… x**
5. **Problemas frecuentes………………………………………………………………………. x**

**Introducción**

**Descripción del programa**

Con este programa se simula la gestión de la información de mascotas (concretamente perros) que residen en una veterinaria, en donde varios usuarios (clientes) podrán interactuar con dicha veterinaria(servidor), la cual se encargará de gestionar todas las operaciones que los clientes le soliciten. La gestión de esta información la controlará la veterinaria por medio de un archivo, donde tendrá todos los registros de los perros que residen actualmente en la veterinaria. Cada cliente puede hacer tantas operaciones o solicitudes como desee y mientras sea posible, ya sea, ingresar una mascota, ver el registro de una mascota, buscar el registro de una mascota o borrar el registro de una mascota.

Con este programa se busca manejar, implementar y profundizar los conceptos aprendidos en clase, principalmente el manejo de memoria, tanto dinámica(memoria) como estática(disco), y la sincronización entre diferentes procesos, por medio del uso de hilos y semáforos.

**Novedades**

Respecto a la primera práctica, en esta segunda práctica, al programa se le añade la capacidad de soportar más de un cliente de forma concurrente, es importante aclarar que en la primera práctica el programa funcionaba como un único administrador que gestionaba toda la información de la veterinaria, mientras que en esta segunda práctica se le atribuye el acceso a la veterinaria a múltiples clientes, los cuales interactúan con un único agente que se encarga de gestionar toda la información de las mascotas, de acuerdo a las solicitudes realizadas por los clientes.

**Requisitos previos**

**Conocimientos mínimos del usuario**

El usuario al que va destinado este manual es afín al aprendizaje o la enseñanza de tópicos relacionados con los sistemas operativos, por lo que el usuario debe tener ciertos conocimientos mínimos de programación en el lenguaje de C y en el uso de alguna distribución del sistema operativo Linux, en este caso, Ubuntu.

**Requisitos técnicos**

Al ser un programa bastante sencillo, los requisitos de hardware son muy pocos e incluso pueden llegar a ser un tanto insignificantes, por lo que es una tarea complicada establecer realmente que se necesita como mínimo para ejecutar nuestro programa y más hoy en día, cuando las tecnologías de algunos componentes del computador tienen mejoras significativas, sin embargo, podríamos decir que se puede usar el programa de forma estable con un procesador de 2 núcleos, 4GB de memoria RAM y un disco duro con más de 256GB.

El programa se ejecutará por medio de la terminal de Ubuntu, sin embargo, para ver el cómo se desarrolló será necesario cualquier entorno de programación que lo permita, así como vim, o cualquier otro editor de texto para código, en este caso se recomienda el uso de Visual Studio Code.

Como lo nombramos anteriormente es necesario contar con una distribución del sistema operativo Linux, pues nuestro programa hace uso de algunos recursos de nuestro computador, por lo que es necesario que el sistema operativo le permita a nuestro programa dicha manipulación, es por esto que es necesaria una distribución de Linux(Ubuntu) como sistema operativo.

**Mecanismos para acceder al sistema**

Nuestro programa es accesible, lo único que se debe tener, como lo nombramos en el apartado anterior, es tener un entorno de desarrollo para poder visualizarlo, y la hora de ejecutarlo proporcionarle permisos de administrador, pues este frecuenta el uso de llamadas a funciones del sistema, las cuales requieren de dichos permisos.

**Instalación y configuración**

Para hacer el uso del programa, solo basta con descargar los archivos necesarios para la correcta ejecución del programa.

Los archivos necesarios se listan a continuación:

* generador.c
* dataNames.dat
* dataBreeds.dat
* dataTypes.dat
* dataDogs.dat
* p2-dogServer.c
* p2-dogClient.c
* makefile
* dataPointers.dat

Todos estos se encuentran en el repositorio: <https://github.com/aeromeror/practica-2>

Como configuración adicional, todos los archivos descargados deben residir en un mismo directorio para que el programa funcione correctamente.

**Utilización del programa**

Una vez descargados todos los archivos mencionados anteriormente, el programa se podrá ejecutar de dos formas distintas, ambas necesitaran de 2 o más terminales en ejecución y un dataDogs.dat, una terminal se encargará de ejecutar el servidor y las restantes terminales el o los clientes, de no existir un dataDogs.dat, usted puede ejecutar el programa generador.c de forma independiente para la creación de este. La primera forma es ejecutar directamente los programas p2-dogServer.c y p2-dogClient.c por medio del compilador gcc, creando primero el ejecutable de cada uno, y posteriormente corriendo dichos ejecutables; antes de correr estos ejecutables, debe tener en cuenta que debe existir un archivo dataDogs.dat y que primero se ejecuta el ejecutable del servidor y posteriormente el de los clientes. La segunda forma es haciendo uso del archivo makefile, en el mismo orden de ideas que se nombró anteriormente.

Una vez se ejecutan tanto cliente como servidor, en la(s) consola(s) donde se está corriendo el usuario se mostrará un menú con las diferentes opciones que este podrá realizar, una vez terminadas todas las solicitudes de todos los clientes, es decir, cuando todos los clientes dejan el servicio, se debe parar la ejecución del servidor, pues este siempre estará a la espera de nuevos clientes, para la

terminación de este usted puede simplemente cerrar la terminal donde se está ejecutando el servidor, sin embargo, es recomendable finalizar su ejecución con la combinación de teclas Ctrl + c.

**Problemas frecuentes**

Con el desarrollo completo del programa se podrían presentar alguno de los siguientes problemas:

* La cantidad de nombres distintos supera el número de la tabla hash, esto generará colisiones, y probablemente datos erróneos a la hora de hacer algún tipo de operación/solicitud.
* Corrupción en la información del archivo principal(dataDogs.dat) o de la tabla hash(dataPointers.dat), debido al uso indebido del programa. Esto se puede solucionar, por medio de una nueva ejecución del programa generador.c.
* Error al reusar el socket del servidor, esto sucede al intentar ejecutar el servidor repetidas veces en cortos intervalos de tiempo. Para solucionarlo basta con dejar los programas sin ejecutar alrededor de 2 minutos para que se desocupe la dirección del socket.