PRIMER PROYECTO

Jhonny Ismael García Hernández

Carnet: 201830454

CUNOC

Quetzaltenango

Laboratorio de Sistemas operativo 1

Tabla de Contenidos

OBJETIVOS

Objetivo general:

Diseñar un software que permita la creación e implementación de árbol de procesos que pueden ser alterados según la siguiente instrucción.

Objetivos específicos:

- Implementar fork para el multiproceso.
- Diseñar una metodología para el control de procesos.
- Realizar operaciones sobre procesos.
- Determinar el árbol de procesos para cada planta.

INTRODUCCIÓN

En el presente trabajo se presenta la implementación de diferentes arboles de procesos mediante un archivo de texto o de manera interactiva, dichos arboles pueden ser operados desde agregar o eliminar nodos del mismo, así mismo pueden ser mostrados de manera colorida.

El hecho de realizar esto, es de entender de como los procesos son ejecutados en simultaneo y como es que loarán compartir recurso para evitar congelar el software y como es que acceden a archivos sin quebrantar la integridad de los datos.

PLANTEAMIENTO DEL PROBLEMA

Construir una aplicación de procesos que pueda generar N<=10 cantidad de arboles con X<=5 cantidad de ramas y Z<=10 cantidad de hojas por rama. El árbol debe estar completamente construido en base a procesos que se generan de manera dinámica.

Ademas de la creación del árbol, se debe permitir operar sobre ello, es decir eliminar ramas, hojas o agregas de la misma manera. Cada proceso debe ser independiente de los otros. Los árboles tienen dos maneras de ser mostradas, escritas en un archivo de texto “pstree” o ser mostrada en pantalla, pero este ultimo debe ser interactivo, es decir que irán cambiando de color independiente, ademas de esto, se

dispone de que el software funcione por un archivo de entrada con N cantidad de instrucciones o teclado.

A continuación la descripción de los colores que se deben mostrar en pantalla por árbol y las instrucciones validas del software.

Tallo:

- Gris
- Negro

Rama:

- Todos los colores

Hojas:

- Verde
- Café

Instrucciones:

- (P, N) Intrusión de crear una planta.
- (P, N, 0) La planta se seca, únicamente quedará el tallo.
- (P, N, X) La planta se seca, únicamente quedará el tallo.
- (P, N, X, Z) Instrucción de crear o reestructurar ramas.

Donde N,X,Z son valores numericos.

MARCO TEÓRICO

MULTITAREA:

Es la característica de los sistemas operativos que permite que los procesos se ejecuten al mismo tiempo compartiendo recursos, esta característica permite al usuario ejecutar varios programas.

FORK:

En los sistemas linux como se sabe es multitarea, permite ejecutar varias tareas en simultaneo donde cada operación o acción es un proceso que contempla uno o más hilos, esto hace que la memoria sea compartida por todos. Al crear un proceso el sistema operativo se encarga de la creación y distribución pero con peticiones a otros procesos.

Una bifurcación o fork, permite la creación de una copia de sí mismo por parte de un programa, que entonces actúa como un "proceso hijo" del proceso originario, ahora llamado "padre". Los procesos resultantes son idénticos, salvo que tienen distinto número de proceso (PID), aunque de cierta manera son dependientes del padre para su finalización.

SEMÁFOROS:

Es una metodología clásica que da acceso o evita a recursos compartidos en donde n cantidad de procesos están intentando acceder. Al emplear semáforos el objetivo es que la sección critica de acceso controle la manipulación de variables o recursos para que dos procesos o más no alteren algo al mismo tiempo ya que puede generar errores grandes.

PIPE:

Una tubería consiste en una secuencia de procesos conectados de manera que la salida de uno es la entrada de otra, esto permite la comunicación basado en productor/consumidor donde los productores son quienes envían datos a los procesos consumidores siguiendo un orden FIFO.

ANÁLISIS DE REQUERIMIENTOS

Requerimiento	Función	Categoría
Lectura de un archivo de texto	Trata sobre que un usuario indica la ruta de un archivo con N cantidad de instrucciones para luego ser procesado por el software.	Evidente
Lectura de datos de manera interactiva mediante el teclado	Para este caso las instrucciones se van pidiendo por teclado uno por uno al usuario.	Evidente
Creación de árbol de procesos	En base a las instrucciones leídas el software crea un árbol de procesos para cada planta.	Oculta
Mostrar árbol de procesos	Despliega un árbol de procesos indicado por el usuario de manera colorida.	Evidente
Imprimir árbol de procesos	Imprime el árbol de procesos en un archivo de texto, tipo pstree	Evidente

Alterar árbol de procesos	Realiza operaciones sobre el árbol, tales como eliminar nodos o agregar unos nuevos.	Oculta

REQUERIMIENTOS DEL SISTEMA

Como tal el proyecto no elije un sistema operativo con especificaciones a seguir, lo único necesario es de escoger un sistema operativo basado en GNU/LINUX ya que algunas instrucciones no son las mismas a otros, como Windows por ejemplo, Nota: para este proyecto se usara ubuntu 18.

HERRAMIENTAS

CODE BLOCKS :

Blocks es un IDE gratuito de C / C ++ y Fortran creado para satisfacer las necesidades más exigentes de sus usuarios. Está diseñado para ser muy extensible y completamente configurable. Nota: Este es un Ide de bajos recursos y facil de usar, pero si desea puede usar otro pero debe mantener el orden de los archivos que se implementaran para el proyecto.

Instalación:

Para este caso se tomara la instalación en ubuntu y derivados:

- sudo add-apt-repository ppa:damien-moore/codeblocks-stable
- sudo apt update
- sudo apt install codeblocks codeblocks-contrib

  De la misma manera si desea mayor información de instalación puede verificar el siguiente

enlace:https://computoevolutivo.eventos.cimat.mx/sites/computoevolutivo/files/Tutorial_para_instalar_CodeBlocks_en_Linux.pdf

GCC COMPILER:

Gcc compiler es una colección de compiladores GNU incluyendo interfaces para C,C++, Objective-C,Fortran,Ada,Go y D. Es compilador viene instalado en las distribuciones GNU/LINUX ya que al final se escribió para el sistema GNU.

GCC requiere el conjunto de aplicaciones conocido como binutils para realizar tareas como identificar archivos objeto u obtener su tamaño para copiarlos, traducirlos o crear listas, enlazarlos, o quitarles símbolos innecesarios.

Instalación:

El compilador GCC no es necesario su instalación ya que el sistema operativo lo implementa normalmente, pero si no lo trae puede intentar con lo siguiente (ubuntu)

- sudo apt-get install gcc
- sudo apt-get install g++

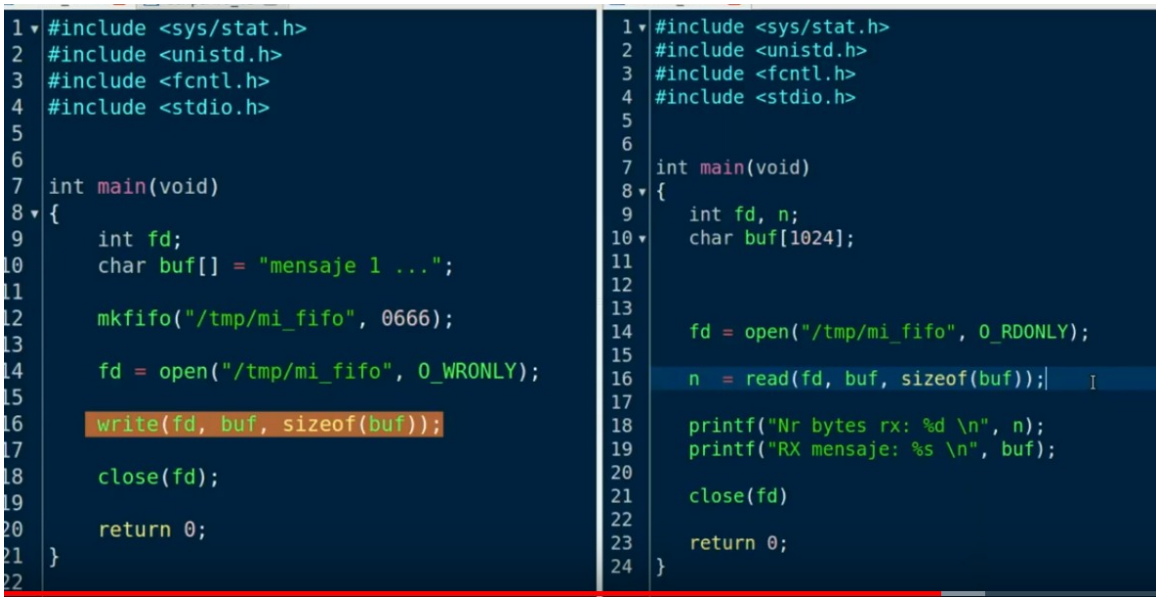
CONCLUSIONES

- Este tipo de redes son bien interesantes y útiles al momento de que se quiera trabajar algo en especifico, además resulta económicamente barato si se dispone de una computadora.

- Al tener un red creada por uno mismo se llega a tener más control sobre esta, a diferencia si se deja que un router lo haga por si solo, además la conectividad es mejor.

BIBLIOGRAFIAS

- https://1984.lsi.us.es/wiki-ssoo/index.php/Sem%C3%A1foros
- https://culturacion.com/sistema-operativo-multitarea-cual-es-su-funcion/
- http://sopa.dis.ulpgc.es/ii-dso/leclinux/procesos/fork/LEC7_FORK.pdf
- https://es.wikipedia.org/wiki/Tuber%C3%ADa_(inform%C3%A1tica)
- https://www.muspells.net/blog/2011/02/dos-formas-de-mandar-senales-a-procesos-en-linux/
- https://es.slideshare.net/jcfarit/unidad-3-gestion-de-procesos-en-linux



```c
#include <sys/stat.h>
#include <unistd.h>
#include <fcntl.h>
#include <stdio.h>


int main(void)
{
    int fd;
    char buf[] = "mensaje 1 ...";

    mkfifo("/tmp/mi_fifo", 0666);

    fd = open("/tmp/mi_fifo", O_WRONLY);

    write(fd, buf, sizeof(buf));

    close(fd);

    return 0;
}
```

```c
#include <sys/stat.h>
#include <unistd.h>
#include <fcntl.h>
#include <stdio.h>


int main(void)
{
   int fd, n;
   char buf[1024];



   fd = open("/tmp/mi_fifo", O_RDONLY);

   n  = read(fd, buf, sizeof(buf));

   printf("Nr bytes rx: %d \n", n);
   printf("RX mensaje: %s \n", buf);

   close(fd)

   return 0;
}
```