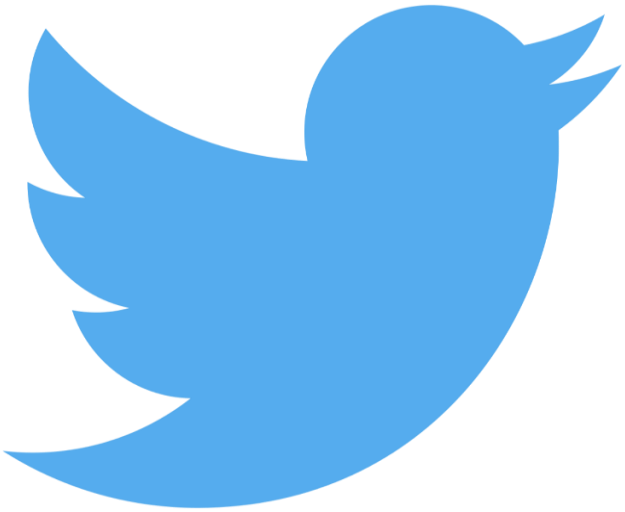


SEGUNDO PROYECTO

Mini-Twitter

DESCRIPCIÓN BREVE
Este es un informe que detalla el desarrollo del segundo proyecto de la materia sistemas operativos.

Camilo Andrés Oviedo Lizarazo
Oscar Alejandro Moreno Galeano
Sistemas Operativos

Contenido

Problema Planteado

El problema planteado era el de crear un mini-twitter en el lenguaje de programación preferido por el grupo de trabajo, esto con el fin de al final tener la capacidad de utilizar herramientas para la comunicación sincrónica y asincrónica (pipes y señales) entre procesos de sistemas basados en Unix.

Primero que todo se poseía dos procesos principales (Cliente y Gestor), el gestor es aquel programa que crea las estructuras de datos, permite almacenar cada uno de los usuarios y sus relaciones; además, se encarga de gestionar el envió de tweets entre usuarios. Los clientes son los procesos que representan cada uno de los usuarios (mínimo uno y máximo 10).El gestor estará en alguno de los dos modos requeridos por el enunciado (síncrono o asíncrono), así, cada uno de los clientes creados después de la inicialización del gestor estarán en el mismo modo que este.

Comunicación Sincrónica

La comunicación síncrona se utilizara principalmente para la gestión de los tweets. En este modo los usuarios reciben los tweets que les han enviado únicamente al conectarse al sistema y cuando explícitamente solicitan una actualización de sus tweets.

Comunicación Asincrónica

Para la comunicación asincrónica se reciben los tweets al momento de conectarse al sistema y al momento exacto cuando son enviados por otro usuario, no existe una forma de decidir si quiere recibirlos o no, simplemente los recibe una vez que se le envían.

Estructuras de datos

Struct datosCliente

Esta estructura se encuentra en el archivo “datosCliente.h”, contiene los siguientes datos:

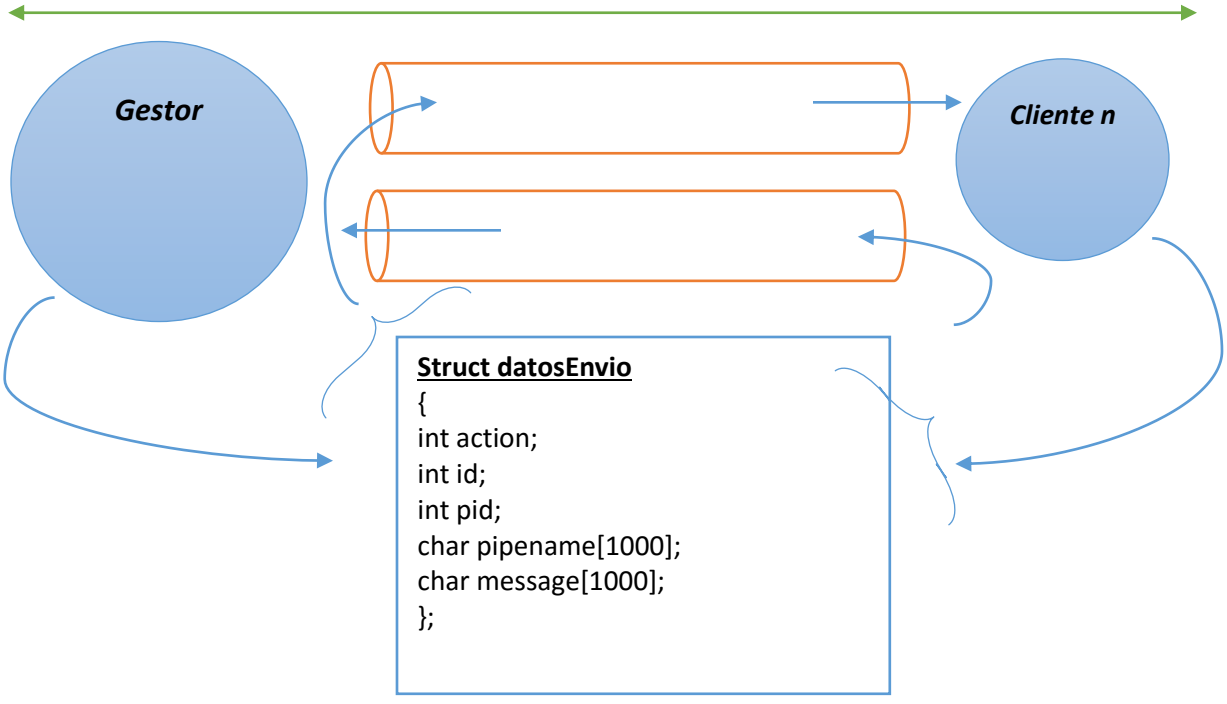
- Int pid. Este es el id del usuario o cliente.

- Map< int, int > follow. Este es un mapa creado para llevar el control de los tweets de los que ha leído el cliente. Por lo tanto, la llave del mapa indica al cliente o clientes que sigue el usuario guardado en pid, y el contenido de la llave es para saber hasta qué posición del vector de tweets ha leído de ese cliente que está en la llave.
- Vector < string > tweet. Este vector guarda los tweets del usuario o cliente.

Struct datosEnvio

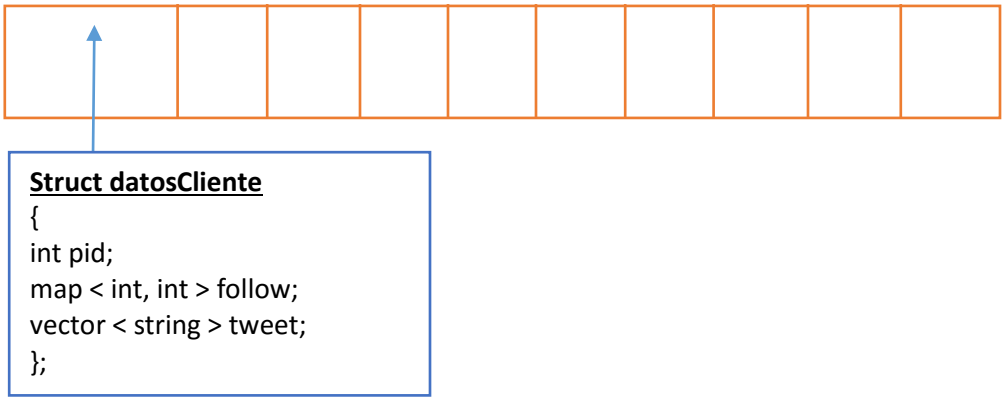
Esta estructura se encuentra en el archivo "datosCliente.h", es utilizada para leer datos entre pipes y contiene los siguientes datos:

- Int action. Este es un entero que permite saber cuál opción del menú ha escogido el cliente.
- Int id. Este es un entero que contiene el número del cliente (entre 1 y 10).
- Int pid. Este es un entero que contiene el pid del proceso cliente creado.
- Char pipename[1000]. Este contiene el nombre del pipe por donde se comunicará el gestor con cliente.
- Char message[1000]. Es utilizado para enviar mensajes entre el cliente y gestor o viceversa.



Vector< datosCliente >datos

Por parte del gestor tenemos un vector de tipo datosCliente que almacena en cada una de las posiciones del vector, información de cada uno de los clientes.



Vector<int> pids

De igual forma en el gestor se tiene un vector dedicado a guardar los pid de los diferentes clientes apenas se conectan, los cuales son muy útiles en el modo asíncrono ya que permiten hacer el kill de la señal correcta.

Algoritmos Importantes

1. Síncrono: Para generar la simulación del estado indicado se generó otra opción que solo es visible y ejecutable cuando el gestor recibe la instrucción “sincrono” para encenderse de esta manera. En esta opción el cliente que quiere ver los tweets solicita al gestor la información, comenzando a mirar en sus followers si hay publicaciones sin leer, eso sí sin importar si el seguido se encuentra conectado o no.

2. Asíncrono: Se inicializa recibiendo como parámetro “asincrono”, en el cliente inmediatamente se crea una señal que está vigente hasta que se cierra el programa, mientras que en el gestor solo se activa hasta entrar a la lectura de Tweets, el gestor determina en qué modo esta él, de estar en este estado entra a una opción especial, la cual revisa y realiza un recorrido sobre los usuarios conectados y de ser seguidos por quien envía el tweet, se les redirige el mensaje haciéndole kill a su señal respectiva, la cual en el cliente ejecuta el handler donde se lee la información enviada

3. El resto de las opciones se generaron a partir de lecturas e impresiones, y con la información capturada se enviaba a través de los diferentes pipes.

Comunicación entre Procesos

Para la comunicación entre los procesos se crearon dos pipes principalmente. El primer pipe es un pipe que todo el tiempo está escuchando, ubicado en el proceso Gestor, listo para obtener los datos que se le envían desde cada uno de los clientes. El segundo pipe es un pipe de comunicación que crea el gestor para cada uno de los clientes, por lo tanto si el cliente numero n espera una respuesta del gestor, este se lo enviara a través de un pipe exclusivo para este cliente con la siguiente estructura en el nombre del pipe "[pipenom]+[n]".

A través de los pipes se envía información contenida en la estructura datosEnvio en el que tanto el gestor como el cliente podrá colocar allí información importante.

Para la parte de comunicación asíncrona se eligió en el menú como la opción cuatro con la ayuda de señales, esta señal se crea al momento de iniciar el cliente y con un handler en el cliente se espera la lectura. Por lo tanto, se activa cada vez que se le indique.