SESIÓN TP6

Sistemas Operativos 1
Abril 2025

1 INTRODUCCIÓN

En esta sesión de problemas se presentan ejercicios relacionados con la tercera práctica (Comunicación y sincronización entre procesos). Esta sesión de problemas está centrada en utilizar tuberías para comunicar datos entre procesos, así como las señales que sirven para que procesos se envíen "interrupciones" entre sí. Sin embargo, estas interrupciones son un mecanismo gestionado a nivel de software y no de hardware.

Esta sesión tiene un ejercicio que debe entregarse mediante el campus virtual, así como la elaboración de rúbricas con la evaluación de códigos de otros alumnos.

2 LAS TUBERÍAS

Las tuberías son un mecanismo de comunicación entre procesos que permite enviar información de un proceso a otro. Suele usarse en la línea de comandos para enviar, por ejemplo, la salida estándar de un proceso a la entrada estándar de otro proceso.

Las tuberías se utilizarán aquí (y, en particular, en la práctica 3) para que un proceso pueda enviar datos a otro proceso. Se realizarán varios experimentos para que se puedan entender bien los conceptos asociados.

1. Empezaremos analizando el tamaño de la tubería. Para ello utilizaremos el archivo `pipe_size.c`, que hace que el proceso hijo escriba en la tubería. Pero el padre no lee ninguna cada de la tubería. Esto hará que la tubería se llene. Observa que llega un momento en que el programa es "cuelga". ¿Por qué se cuelga? ¿Qué información nos está dando el programa sobre el tamaño de la tubería?
2. En Linux existe una forma de aumentar el tamaño de la tubería. Tiene un ejemplo en `pipe_large.c`. Las llamadas a sistema que se utilizan para aumentar no forman parte del estándar POSIX y, por tanto, no tienen por qué funcionar en otros sistemas Unix (como el Mac).

¡Prueba el código! Puede parecer útil tener una tubería grande, pero no es eficiente, dado que una tubería grande hace que se utilicen más recursos del sistema operativo. Además, existen otros sistemas más eficientes de comunicación entre proceso si se quiere transferir mucha información (p.ej. comunicación vía red, archivos, archivos mapeados a memoria). Ten en cuenta que la tubería fue una de las primeras estrategias de comunicación interproceso.

3. A continuación, haremos que el buffer de la tubería se llene por el hijo. Luego el padre leerá los datos escritos por el hijo. Tiene un ejemplo en el código `pipe_write_read.c`. Analiza el código, ejecútalo y observe qué hace. Observa que el hijo acaba antes de que el padre empieza a leer. Además, verá que el código se "cuelga" al final. ¿Por qué?
4. Por último, haremos una prueba con un código que genera un "flujo continuo" de datos: el hijo escribe datos y el padre los lee (`pipe_write_read_continuous.c`). Este último ejemplo es la forma en la que se comunican dos procesos cuando generamos una tubería.

A continuación, se proponen dos ejercicios relacionados con la práctica, que usan las tuberías.

5. En el archivo `pipe_size.c` hemos visto el tamaño, en bytes, del buffer asociado a la tubería. Modifica el código para escribir valores enteros. A continuación id incrementando el valor que se escribe en el buffer. ¿Cuántos valores enteros puede escribir?
6. Toma el archivo `pipe_write_read.c` para que el hijo escriba enteros en formato binario (p. ej. utilizando las funciones `read` y `write`). Incrementa el valor que escriba en el buffer. Una vez el buffer esté lleno, el padre empezará a leer los valores. Imprime los valores leídos por pantalla para asegurar que los valores que lea son los correctos.

3 LAS SEÑALES

Las señales son otro mecanismo de comunicación muy sencillo entre procesos. Son interrupciones de software que dos procesos pueden enviar entre sí para señalizar que ha ocurrido un evento. Veamos un par de ejemplos

1. El archivo `sigterm_sigint.c` contiene un ejemplo en el que la aplicación espera a recibir una señal. Observa el código e indica cuál es el comportamiento de este: ¿cuándo imprimirá el mensaje "*Exiting main*"?
2. Por otor lado, el archivo `sigprocesses.c` contiene otro ejemplo en el que dos procesos, padre e hijo, se envían señales entre sí. Este es un mecanismo de sincronización sencillo entre dos procesos (ahora existen mecanismos de sincronización más avanzados que se verán en Sistemas Operativos 2 como los semáforos). ¿Cómo sabe el padre quien es su hijo? ¿Cómo sabe el hijo quien es su padre?

4 EJERCICIO A ENTREGAR

Se propone a continuación un ejercicio que forma parte de los *problemes avaluables* (PA) de la asignatura. Los detalles de la evaluación se indican al final.

En concreto, el código a entregar debe hacer las siguientes tareas:

1. El proceso padre crea un proceso hijo. Ambos procesos se comunicarán mediante una tubería.
2. El proceso padre lee una cadena introducida por el usuario desde la línea de comandos.
3. Mientras tanto, el proceso hijo espera a recibir una señal del padre.
4. Una vez que el proceso padre ha recibido la cadena, la envía a través de la tubería al hijo y envía una señal SIGUSR1 al proceso hijo.
5. Al recibir la señal, el proceso hijo lee la cadena desde la tubería y la transforma a mayúsculas.
6. Mientras el proceso hijo transforma la cadena, el proceso padre espera.
7. Una vez que el proceso hijo ha terminado la transformación, escribe el resultado en la tubería y envía la señal SIGUSR2 al padre.
8. Al recibir la señal, el padre lee la cadena transformada a mayúsculas desde la tubería y la imprime por pantalla.
9. Mientras tanto, el proceso hijo espera a recibir una señal del padre.
10. Una vez que el padre ha escrito la cadena en mayúsculas, vuelve a enviar la cadena por la tubería para una segunda transformación y vuelve a enviar la señal SIGUSR1 al proceso hijo.
11. El proceso hijo lee la cadena desde la tubería y la transforma ahora invirtiendo su orden.
12. Mientras el hijo invierte la cadena, el padre espera.
13. Una vez que el hijo ha terminado, escribe el resultado en la tubería y envía la señal SIGUSR2 al padre.
14. Al recibir la señal, el padre lee la cadena invertida desde la tubería y la imprime por pantalla.

Ambos procesos finalizan satisfechos y felices :-D.

Para realizar el ejercicio se proporciona una plantilla que será de ayuda para hacerlo.

Este es un ejemplo de la salida esperada del ejercicio:

```
Introdueix una cadena: sistemes operatius 1 2024
[signal] El fill ha rebut el SIGUSR1
[signal] El pare ha rebut el SIGUSR2
[pare] Cadena uppercase rebuda: SISTEMES OPERATIUS 1 2024
[signal] El fill ha rebut el SIGUSR1
[signal] El pare ha rebut el SIGUSR2
[pare] Cadena invertida rebuda:
4202 1 SUITAREPO SEMETSIS
```

5 PROCESO DE EVALUACIÓN

Se pide entregar mediante la tarea del campus virtual el ejercicio propuesto en la sección 4. Se realizará una única entrega por cada grupo de laboratorio. Cada pareja tiene un número asignado, que puede consultar en el campus virtual. El nombre de archivo deberá indicar sólo la pareja que entrega: así, por ejemplo, **el grupo 31** entregará un único archivo llamado **grupo31.sh**.

La fecha límite de entrega es mañana jueves hasta las 15h. A las 18h se harán disponibles todas las entregas en el campus virtual. Es muy importante que haga las entregas con el nombre de archivo correcto para evitar retrasos para hacer disponibles las entregas.

A continuación, **cada grupo evaluará dos ejercicios entregados por l@s compañer@s** que tengan un número de grupo inmediatamente inferior al que tengan asignados ell@s. Es decir, el grupo 31 tendrá que evaluar pues los grupos 30 y 29. Si el 30 no ha entregado, el grupo 31 evaluará el 29 y 28. El grupo 02 evaluará el grupo 01 y el que tenga el número más grande (es decir, la numeración es circular).

La evaluación se utilizará utilizando la **rúbrica** disponible en el campus. Utilice los listados de archivos proporcionados para asegurar el buen funcionamiento del ejercicio entregado. Evaluar cada ejercicio no debería llevar más de 15 minutos. **Cada grupo entregará sus dos rúbricas** vía campus virtual. Para facilitar la revisión de las rúbricas se pide que cada rubrica tenga el nombre **rubrica_grupoXX_grupoYY.xlsx**, donde XX es el grupo que ha evaluado y YY corresponde a su grupo.

La fecha plazo para entregar las dos rúbricas asignadas es el viernes (23:59). La evaluación asignada por usted nos ayudará a decidir qué soluciones se muestran en la próxima sesión de problemas.

La evaluación de este ejercicio tendrá valores entre 0 y 10. El ejercicio debe intentar resolver el problema propuesto utilizando las instrucciones presentadas en el primer y segundo tutorial. A la hora de evaluar no se penalizará entregar un ejercicio que no funcione. En cambio, en caso de que el código de un grupo no funcione y la rúbrica correspondiente de quien lo ha evaluado no lo refleje, este hecho quedará reflejado en la nota. Por último, **en caso de que no se entregue el ejercicio o la rúbrica, o la solución entregada utilice instrucciones no permitidas o la solución no tenga nada que ver con el ejercicio propuesto la evaluación será de cero**.