DISEÑO Y ADMINISTRACIÓN DE SISTEMAS OPERATIVOS

PED2

COMUNICANDO PROCESOS

Misrraim Suárez Pérez - misrraimsp@gmail.com

Introducción

En este trabajo se ha desarrollado un conjunto de tres códigos fuente en C, que compilados y ejecutados conducen a la interacción de tres procesos (P1, P2 y P3) mediante diferentes mecanismos estudiados en la asignatura.

Implementación

Se ha escrito un conjunto de tres ficheros fuente en C. A continuación se pasa a comentar los detalles más relevantes de cada uno de ellos.

Conviene destacar que cada uno de los ficheros fuente posee código de sobrecarga, bajo comentario para que en principio no tenga efecto, para testar el cálculo de las estadísticas. Esto ha sido necesario ya que el código pedido en la práctica no tiene el suficiente tamaño como para que las estadísticas de Linux lo contabilicen, saliendo siempre tiempos nulos.

1. fuente1.c

   Este fichero es el correspondiente al proceso P1. Realiza todas las tareas especificadas en el enunciado de la práctica. Esto es, capta un mensaje desde el usuario, crea otro proceso P2, le pasa el mensaje por una tubería sin nombre, crea una cola de mensajes y se queda esperando a recibir por dicha cola. Una vez recibe el mensaje envía señales de terminación a los procesos P2 y P3, para concluir calculando las estadísticas de uso y mostrándolas por pantalla.

   Especial mención debe ser hecha para resaltar que primero se envía la señal de terminación al proceso P3. Esto causará que P3 ingrese en el estado zombie, permitiendo a P2 recolectar sus estadísticas. Para dar tiempo a que P2 sea planificado antes de que P1 le envíe la señal de terminación, P1 se duerme durante un segundo. Cuando vuelve del sueño, P1 envía a P2 la señal de terminación, lo que coloca a P2 en estado zombie. A continuación invoca a *wait* para recoger las estadísticas de P2 (y de P3).

   Dentro de este fichero también se encuentra el código de la primera parte de la vida de P2. Este proceso lee el mensaje de la tubería sin nombre que le pasa su padre (P1) y lo coloca en un fichero FIFO que él mismo ha creado. Destacar que el fichero FIFO se abre en modo *O_RDWR,* lo que permite escribir en él sin que simultáneamente otro proceso distinto tenga que estar al otro lado para leer el contenido. Hecho lo anterior, P2 efectúa una llamada al sistema *execl* para cambiar su región de código y datos por el fichero ejecutable Ej2, resultado de compilar el fichero fuente2.c

2. fuente2.c

   Este fichero es el correspondiente al resto del proceso P2. Realiza todas las tareas especificadas en el enunciado de la práctica. Esto es, lee el mensaje del fichero FIFO, crea una región de memoria compartida y un semáforo para protegerla. Crea un proceso hijo (P3) y duerme un segundo. Al despertar coloca el mensaje en la

región de memoria compartida y desbloquea el semáforo. Finaliza llamando a *wait* para recoger las estadísticas de su hijo (P3) y se pausa su ejecución.

Destacar que el semáforo se inicializa a cero, y que la acción de desbloqueo es incrementarlo una unidad.

La parte del proceso hijo P3 en este fichero fuente tan solo corresponde a la llamada al sistema *execl* para comenzar a ejecutar el contenido del fichero fuente3.c.

3. fuente3.c

   Este fichero es el correspondiente al proceso P3. Realiza todas las tareas especificadas en el enunciado de la práctica. Esto es, cargar la región de memoria compartida y el semáforo creados por P2 e intentar decrementar el semáforo, pasando a quedar bloqueado. Una vez incrementado por P2, P3 es desbloqueado y accede a la memoria compartida para lee el mensaje del usuario, mostrándolo en pantalla. A continuación carga la cola de mensajes creada por P1 y deposita en ella un mensaje, cuyo contenido es su propio PID. Tras ello suspende la ejecución.

Referencias

[1] http://man7.org

[2] The Linux Programming Interface, Michael Kerrisk

[3] Fundamentos del Sistema Operativo Unix, 2nd Ed, Díaz, Muñoz, Chaos