EXAMEN PRÁCTICAS MÓDULO 2 - ALEON - 2018/19

Debes entregar esta hoja cumplimentada para que el examen sea evaluado.

Ejercicio [10 puntos]

El objetivo del ejercicio es construir dos programas: `lanzador <num_cli> <dir_pathname>` y `servidor <dir_pathname>`. El programa `lanzador` tiene dos argumentos: un número, `<num_cli>`, que indica el número de procesos cliente que se van a lanzar, los cuales ejecutarán su parte correspondiente del programa `lanzador`; y el *pathname* del directorio, `<dir_pathname>`. El programa `servidor` tiene un argumento que representa el *pathname* de un directorio, `<dir_pathname>`, que será donde se realizará el procesamiento indicado por las peticiones de los clientes. A continuación, se describe la funcionalidad requerida para cada uno de los programas.

`lanzador <num_cli> <dir_pathname>`

El programa lanzador debe crear un cauce con nombre, `FIFOpet`, que permitirá al servidor aceptar solicitudes de servicio. La solicitud del cliente viene identificada por el `PID` del proceso cliente y otra información adicional. Además, `lanzador` debe establecer un manejador de señal para la señal `SIGCHLD` que muestre por pantalla un mensaje indicando la finalización de cada uno de sus hijos con su `PID` correspondiente.

A continuación, `lanzador` debe crear tantos procesos cliente como indique el argumento `<num_cli>`. Cada proceso clientetiene que crear un cauce con nombre, `FIFO.pid`, cuyo *filename* se construye con la cadena "FIFO." y su propio `PID` que constituirá el canal privado de recepción de información. En relación a las solicitudes al servidor, el cliente genera un número aleatorio entre 1 y 20, el cual representa la entrada n-ésima de un directorio, y un número aleatorio entre 1 y 2, el cual permite permite elegir entre dos metadatos de archivo: "i" número de inodo o "s" tamaño de archivo. Después, el proceso cliente escribe en el cauce `FIFOpet` el número de entrada, el identificador de metadato y su propio `PID` como una petición atómica. Posteriormente, lee del cauce privado `FIFO.pid` el resultado que enviará al servidor, el cual puede ser **-1** o un número mayor o igual que **0**. Si es **-1**, muestra en la salida estándar "FALLO"; en otro caso, muestra el número de inodo o tamaño de archivo (en función de su petición) junto con el valor devuelto por el servidor, finalizando su ejecución. El formato de salida es:

```
Cliente <Pid>: FALLO | Número de Inodo = <resultadoServidor> | Tam Archivo =
<resultadoServidor>
```

Tras esto, `lanzador` debe crear un proceso que ejecute el programa `servidor`, `servidor <dir_pathname>`. Tenga en cuenta que el programa `servidor` lee peticiones de su entrada estándar y escribe resultados en el cauce privado asociado al cliente que realiza la petición.

Cuando finalice el último de sus procesos hijos, el programa lanzador debe eliminar el archivo `FIFOpet` y todos los archivos privados de los clientes, `FIFO.pid`, del sistema de archivos y finalizar.

`servidor <dir_pathname>`

Debe leer de la entrada estándar peticiones que contengan: la entrada n-ésima de un directorio, un carácter 'i' o 's' que identifica el metadato requerido para el archivo de la entrada correspondiente y el `PID` del proceso cliente que realiza la solicitud. El programa finaliza cuando no hay más peticiones desde la entrada estándar.

Para cada petición, el programa debe comprobar si la entrada solicitada es válida en el directorio indicado por `<dir_pathname>`, y además se corresponde con un archivo regular. Si no es así debe escribir en el fifo privado del cliente correspondiente el valor **-1**. Si la entrad es válida y se corresponde con un archivo regular, entonces el servidor escribe en el fifo privado del cliente correspondiente el valor del metadato de archivo (atributo) que indica el carácter de la petición: 'i' número de inodo o 's' tamaño de archivo.