CC4302 Sistemas Operativos

Tarea 3 – Semestre Primavera 2015 – Prof.: Luis Mateu

En un pub del barrio universitario los estudiantes se reunen en la noche a tomar cerveza. Cada cierto tiempo, los estudiantes deben visitar el baño. El baño es amplio y admite un número ilimitado de estudiantes. El problema es que es uno solo y como es lógico damas y varones no deben compartir el baño.

En esta tarea Ud. deberá implementar un driver para Linux que permita coordinar las entradas y salidas del baño. Cuando una dama necesita ingresar al baño abre el dispositivo */dev/damas* en modo escritura. Al salir lo cierra. De la misma forma cuando un varón necesita ingresar abre */dev/varones* y cuando sale lo cierra. Múltiples damas puedes escribir en */dev/damas* produciendo un solo flujo de bytes. Lo mismo para los varones en */dev/varones*. Pero damas y varones no puedes escribir al mismo tiempo. Por ejemplo si un varón abre */dev/varones* mientras que hay damas escribiendo en */dev/damas*, el open del varon debe esperar hasta que todas las damas cierren */dev/damas*.

Los dispositivos */dev/damas* y */dev/varones* tienen número *major* 62 y números *minor* 0 y 1 respectivamente. El siguiente ejemplo usa el comando *cat* de Unix para demostrar el comportamiento esperado para estos dispositivos. Su driver debe reproducir exactamente el mismo comportamiento. Si hay aspectos que el ejemplo no aclara, decida Ud. mismo tratando de simplificar su tarea. Las filas de la tabla están ordenadas cronológicamente. Lo que escribió el usuario aparece en **negritas**.

Shell 1	Shell 2	Shell 3	Shell 4
$ **cat > /dev/damas** (1) **hola**			
			$ **cat /dev/damas** (2) hola
que tal			que tal (3)
	$ **cat > /dev/damas** **estoy aqui**		estoy aqui (4)
		$ **cat /dev/damas** hola que tal estoy aqui **<control-C>** $ (5)	
		$ **cat > /dev/varones** **123** (6)	
yo tambien			yo tambien (7)
<control-D> $ **cat /dev/varones** $ (8)			
	<control-D> (9) $		$
$ **cat /dev/varones** 123 (10)			
456		**456** (11)	$ **cat /dev/damas** hola que tal estoy aqui yo tambien $
789	$ **cat > /dev/varones** **789** (12)		

			$ **cat > /dev/damas** **hola** (12)
<control-C> $			
$ **cat > /dev/damas** **de nuevo** ***<control-C>*** $ (13)			
$ **cat /dev/varones** 123 456 789			
$	***<control-D>*** $	***<control-D>*** $	
$ **cat /dev/damas** hola ***<control-C>*** $ (14)			
			<control-C> $

Notas:

(1) El comando cat escribe en /dev/damas todo lo leído desde el teclado hasta encontrar un <*control-D*>.

(2) El comando cat lee lo escrito hasta el momento en /dev/damas y lo muestra en el terminal. El comando cat no termina porque hay damas escribiendo en /dev/damas. Observe la diferencia entre *cat > /dev/damas*, que escribe en el dispositivo, y *cat /dev/damas*, que lee el dispositivo.

(3) La dama del shell 1 escribe en /dev/damas por lo que el cat lector en shell 4 muestra lo escrito en el terminal.

(4) Una nueva dama abre /dev/damas en el shell 2 y escribe un mensaje, que debe ser mostrado por el cat lector.

(5) El shell 3 lee el contenido actual de /dev/damas. Como cat no termina, se usa <*control-C*> para terminarlo.

(6) Un varón escribe /dev/varones, pero que se queda bloqueado en el open porque hay 2 damas escribiendo en /dev/damas.

(7) El shell 1 escribe en /dev/damas y el cat lector del shell 4 lo muestra.

(8) El <*control-D*> se interpreta como fin de archivo y por lo tanto el shell 1 cierra /dev/damas. Inmediatamente se examina /dev/varones pero está vacío porque el varón del shell 3 todavía no logra abrir /dev/varones, puesto que queda una dama escribiendo.

(9) La dama que quedaba en el shell 2 cierra /dev/dama. El cat lector del shell 4 debe terminar porque no hay más damas escribiendo. Para lograrlo haga que read retorne 0 bytes leídos. Por otra parte el varón pendiente en el shell 3 logra abrir /dev/varones y escribe 123.

(10) Cat lee /dev/varones. El comando no termina porque hay 1 varón escribiendo en /dev/varones.

(11) Desde el shell 3 se escribe 456 en /dev/varones. El cat lector del shell 1 debe mostrarlo. Mientras tanto el shell 4 lee /dev/damas, mostranto su contenido actual. Termina de inmediato porque no hay damas escribiendo.

(12) En el shell 4 una dama intenta abrir /dev/damas, pero el open debe esperar hasta que no hayan procesos escribiendo en /dev/varones.

(13) Otra dama hace lo mismo en el shell 3, pero se aburre y aborta el comando con <control-C>.

(14) Los 2 varones pendientes cerraron /dev/varones por lo que la dama del shell 4 logra abrir /dev/damas y escribe hola. El cat lector del shell 4 debe mostrar este mensaje y quedarse a la espera de más mensajes escritos en /dev/damas. El <*control-C*> termina su ejecución.

Recursos

Baje de U-cursos el archivo *modules2015-2.tgz*. Contiene enunciados y soluciones de tareas de semestres

anteriores con instrucciones para compilarlas y ejecutarlas (ver archivos README.txt en cada directorio). Además se adjunta un tutorial sobre programación de módulos y drivers en Linux. Para acceder al número *minor* en el driver use:

```
iminor(filp->f_dentry->d_inode)
```

Programe su solución en el archivo pub-impl.c del directorio *Pub*. Ahí encontrará un *Makefile* para compilar su tarea, las instrucciones para crear los dispositivos */dev/damas* y */dev/varones*, y la implementación de mutex y condiciones a partir de los semáforos del núcleo de Linux.

Antes de cargar y probar su tarea asegúrese de ejecutar el comando Unix *sync* para garantizar que sus archivos hayan sido grabados en disco y no están pendientes en un caché de Unix. Recuerde que los errores en su driver pueden hacer que Linux se bloquee indefinidamente y tenga que reiniciar el sistema operativo.

Plazo de entrega

La tarea se entrega *funcionando* en U-cursos. Para ello entregue solo el archivo *pub-impl.c* que implementa el driver pedido. El plazo de entrega es el Lunes 7 de diciembre a las 23:59. Se descontará medio punto por día de atraso, excepto días sábado, domingo o festivos. Resuelva la tarea antes del control 3, le servirá de estudio.