CC41B - CC4302 Sistemas Operativos

Examen – Semestre Primavera 2010

Prof.: Luis Mateu

Pregunta 1

El cuadro de más abajo muestra la implementación actual del procedimiento `safePrint` que sirve para que una tarea de nSystem imprima un documento.

```
nSem mutex; /* = nMakeSem(1); en nMain */
void safePrint(Doc *doc) {
  nWaitSem(mutex);
  seqPrint(doc, 0); /* dado! */
  nSignalSem(mutex);
}
```

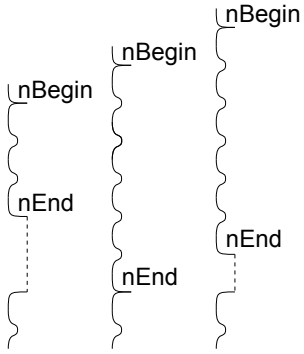
El procedimiento `seqPrint(Doc *doc, int id)` es dado y realiza la impresión del documento doc en la impresora con identificación `id`. Sólo retorna una vez que la impresión concluyó. El semáforo se usa para que cada tarea tenga acceso exclusivo a la impresora.

Reprograme `safePrint` de tal modo que se impriman hasta 4 documentos en paralelo. Para ello Ud. dispone de 4 impresoras con identificadores de 0 a 3. Ud. debe garantizar que cuando una tarea solicita imprimir y hay una impresora disponible, el documento debe comenzarse a imprimir inmediatamente en esa impresora. Para resolver este problema Ud. debe usar las tareas y monitores de nSystem (*no use semáforos*).

Pregunta 2

Implemente en nSystem la herramienta de sincronización denominada sala de reuniones (nRoom) con los siguientes procedimientos:

- `nRoom nMakeRoom( )`: construye una nueva sala de reuniones.
- `void nBegin(nRoom r)`: ingresa a una sala de reuniones.
- `void nEnd(nRoom r)`: vota por finalizar la reunión. Se bloquea en espera hasta que todas las tareas que hayan ingresado hayan votado por finalizar la reunión. Esto incluye aquellas tareas que ingresen después de esta invocación de nEnd.



La figura muestra con un diagrama de threads un ejemplo de uso de una sala de reuniones. Observe que cuando se invoca el último nEnd todas la tareas pasan a estado READY.

Implemente esta API usando los procedimientos de bajo nivel de nSystem (`START_CRITICAL`, `Resume`, `PutTask`, etc.). Ud. *no puede usar* otros mecanismos de sincronización ya disponibles en nSystem, como semáforos, monitores, mensajes, etc.

Pregunta 3

i. ¿Cuál de las 2 estrategias de reemplazo de páginas vistas en clases usaría Ud. en un sistema operativo mono-proceso para implementar paginamiento en demanda? Explique por qué descarta la otra estrategia.

ii. ¿Tiene sentido usar alguna estrategia de *scheduling de disco* en un disco SSD? Explique.

iii. Considere un núcleo moderno de sistema operativo para un multiprocesador. ¿Que herramienta de sincronización usaría para garantizar la exclusión mutua al acceder a la cola de procesos "ready" y por qué?

iv. Un programador usuario de un computador Linux descubre que la implementación de `read(fd, n, buf)` en el núcleo no valida que `buf` apunte a un área de memoria del proceso. Explique cómo este programador podría lograr correr su propio código en modo sistema (¡sin ser el usuario root!).

v. ¿Cuales serían las consecuencias de implementar un sistema de paginamiento sin una TLB (*Translation Lookaside Buffer*)?