SISTEMAS OPERATIVOS I

Segundo curso Ingeniería Informática. Curso 2005-2006

Práctica 2: Sincronización. Implementar semáforos generales a partir de semáforos binarios según la implementación de Hemmendinger y utilizarlos para solucionar el problema de los filósofos comensales. Comprobar el funcionamiento ejecutando varios filósofos desde distintas ventanas.

Comentarios:

Hay que hacer DOS programas

- *inicializa.c:* Recibe como parámetro el número de filósofos. Crea el array de estados y lo inicializa al valor adecuado. Crea los semáforos y los inicializa a los valores correspondientes.
- *filosofo.c:* Recibe el número de filósofo como parámetro y simula el comportamiento de un filósofo indicando en pantalla cuando comienza y termina de pensar y de comer.

Cada filósofo, en lugar de un bucle infinito, se ejecuta n veces (n predeterminado a 50). Es decir, sería algo así:

```
filosofo (int i)
{
 int veces=N;
 while (--veces){
   toma_cubiertos (i);
   come(i);
   deja_cubiertos(i);
   piensa(i);
 }
}
```

Dado que no se ha visto la compartición de memoria, las variables compartidas por los procesos se simularán en fichero utilizando para ello las llamadas al sistema *open, read, write, lseek y close* según sea necesario.

Las funciones piensa() y come() dejan al filósofo en espera un tiempo aleatorio e imprimen la hora a la que empieza y termina de comer o pensar el filósofo en cuestión. LA UNICA salida que realiza por pantalla cada filósofo es la de las funciones piensa y comE, y además, dicha salida la hará con la llamada al sistema *write*, como en el código que se muestra a continuación:

```
void piensa(int i)
{
 char mensaje [MAXCADENA];
 time_t t;
 pid_t p;

 p=getpid();
 t=time(NULL);
 sprintf (mensaje,"filosofo %d (%u): Comienzo a pensar a las %s",i,p,ctime(&t))
 write (STDOUT_FILENO,mensaje,strlen(mensaje));
 /*esperar tiempo aleatorio entre 0 y 5 segundos*/
 sprintf (mensaje,"filosofo %d (%u): termino de pensar a las %s",i,p,ctime(&t))
 write (STDOUT_FILENO,mensaje,strlen(mensaje));
}
```

Deben implementarse semáforos generales a partir de semáforos binarios según la implementacion de Hemmendinger. Los semáforos binarios se implementarán haciendo uso de que en UNIX la llamada open cuando intenta crear un fichero en modo exclusivo lo hace de manera atómica. Así, las operaciones binarias Pb y Vb quedarían:

```
Pb (char * s)
{
 int df
  while ((df=open(s,O_CREAT | O_EXCL))==-1);
  close (df);
}

Vb(char * s)
{
 unlink (s);
}
```

La implementación de los semáforos generales debe incluir:

- La definición del tipo semáforo general.
- Las operaciones
  - InicializarSemaforo: Da al semáforo el valor inicial, e inicializa los semáforos binarios auxiliares a su valor correspondiente.
  - Operacion P

– Operacion V

IMPORTANTE

Debe además tenerse en cuenta que:

- La llamada open no funciona de manera atómica en sistemas de ficheros NFS, por tanto, los ficheros que implementan los semáforos binarios deben crearse en un directorio LOCAL de la máquina donde se trabaja (p.e. /tmp)
- El valor del semáforo debe ser compartido por los procesos que usan el semáforo. Al igual que otras variables compartidas se almacenará en un fichero. Toda la entrada/salida se hará con las llamadas al sistema *open, read, write, lseek y close*; además, el fichero donde se guarda el valor del semáforo no debe quedar abierto entre operaciones P y/o V sobre el semáforo correspondiente.

FORMA DE ENTREGA Como en la práctica anterior.

Fecha de entrega: semana 19 al 23 Diciembre 2005