Sistemas Operativos - Sinronización (1)

---------------------------------------

Objetivo: Problema del productor/consumidor

Se le pide que resuelva el problema clasico del productor/consumidor,

utilizando como base el código de la solución utilizando semáforos

presentado en el libro.

Dicho código, que no compila y muy incompleto, es el siguiente

/\* BEGIN código del libro \*/

#define N 100 /\* number of slots in the buffer \*/

typedef int semaphore; /\* semaphores are a special kind of int \*/

semaphore mutex = 1; /\* controls access to critical region \*/

semaphore empty = N; /\* counts empty buffer slots \*/

semaphore full = 0; /\* counts full buffer slots \*/

void producer(void)

{

int item;

while (TRUE) { /\* TRUE is the constant 1 \*/

item = produce item( ); /\* generate something to put in buffer \*/

down(&empty); /\* decrement empty count \*/

down(&mutex); /\* enter critical region \*/

inser t item(item); /\* put new item in buffer \*/

up(&mutex); /\* leave critical region \*/

up(&full); /\* increment count of full slots \*/

}

}

void consumer(void)

{

int item;

while (TRUE) { /\* infinite loop \*/

down(&full); /\* decrement full count \*/

down(&mutex); /\* enter critical region \*/

item = remove item( ); /\* take item from buffer \*/

up(&mutex); /\* leave critical region \*/

up(&empty); /\* increment count of empty slots \*/

consume item(item); /\* do something with the item \*/

}

}

/\* END código del libro \*/

Claramente esto no funciona de forma directa, los semáforos no

son enteros, pero nos muestra un poco la idea.

Recordar que tan to productor como consumidor deben ser procesos

o hilos independientes independientes.

Hay que investigar:

1) Semáforos: se utilizarán semáforos POSIX

2) Tipos de semáforos: con/sin nombre, uso similar a tuberías

3) Como se declara y accede a un semáforo

4) Como se hacen las operaciones equivalentes a up() y down()

Par esto pueden leer las páginas de manual de sem\_overview, sem\_wait

y sem\_post.

También https://users.cs.cf.ac.uk/Dave.Marshall/C/node26.html#SECTION002640000000000000000

Un esquema de la solución podría ser:

1) Utilizamos semáforos sin nombre (procesos relacionados)

2) Un proceso padre que crea e incializa los semáforos necesarios

3) Se crean dos procesos hijos para productor y consumidor

4) los hijos utilizan una espera aleatoria cuando producen/consumen

5) los hijos muestran su actividad

6) luego de un nro finito de items se termina

La variante que utiliza semáforos con nombre es parecida pero

para procesos independientes, uno de ellos crea e inicializa los

semáforos, luego 'espera' a que el otro esté en línea para comenzar

ambos el ciclo.