Práctica 2: Casos prácticos de monitores en C++11

Sistemas Concurrentes y Distribuidos

Francisco Lara Marín

77447257-R

franciscolara@correo.ugr.es

Grado en Ingeniería Informática

ETS de Ingenierías Informática y de Telecomunicación

Universidad de Granada

Curso 2022-2023

Contenido

1

Introducción

El objetivo es ilustrar el proceso de diseño e implementación de los monitores SU. Se resolverán un par de problemas ya resueltos:

- Productores y consumidores múltiples
- El problema de los fumadores.

Además un nuevo problema:

- El problema de los lectores-escritores.

2

Productor-consumidor SU

A partir del problema ya resuelto mediante semáforos en la práctica anterior, se realizarán una serie de cambios para resolver con monitores SU.

2.1 | Implementación

La función producir_dato tiene como argumento el número de hebra productora. La hebra productora número i produce los p números enteros quqe hay entre i*p y i*p + p-1.

```
int producir_dato(int x)
{
   this_thread::sleep_for( chrono::milliseconds( aleatorio<min_ms,max_ms>() ));

   int valor_producido;

   mtx.lock();
   valor_producido = (x*(num_items/n_productoras)) + producidos_hebra[x];
   producidos_hebra[x]++;

   cout << "hebra productora, produce " << valor_producido << endl << flush ;
   mtx.unlock();
   cont_prod[valor_producido]++ ;
   return valor_producido ;
}
```

Figure 2.1: Función producir_dato()

Debemos tener un array compartido con n entradas que indique cuantos ítems ha producido cada hebra.

```
int
   producidos_hebra[n_productoras];
```

Figure 2.2: Array compartido

2.2 | Ejemplo de ejecución

```
------------------------------------------------------------------
Problema del productor-consumidor múltiple (Monitor SU, buffer LIFO).
------------------------------------------------------------------
hebra productora, produce 0
hebra productora, produce 5
hebra productora, produce 6
                  hebra consumidora, consume: 0
hebra productora, produce 1
hebra productora, produce 10
                  hebra consumidora, consume: 6
                  hebra consumidora, consume: 5
hebra productora, produce 2
                  hebra consumidora, consume: 1
hebra productora, produce 7
hebra productora, produce 3
                  hebra consumidora, consume: 10
                  hebra consumidora, consume: 2
hebra productora, produce 11
hebra productora, produce 8
                  hebra consumidora, consume: 11
hebra productora, produce 4
                  hebra consumidora, consume: 7
                  hebra consumidora, consume: 3
hebra productora, produce 9
hebra productora, produce 12
                  hebra consumidora, consume: 8
                  hebra consumidora, consume: 4
hebra productora, produce 13
                  hebra consumidora, consume: 12
                  hebra consumidora, consume: 9
hebra productora, produce 14
                  hebra consumidora, consume: 13
                  hebra consumidora, consume: 14
comprobando contadores ....

solución (aparentemente) correcta.
```

Figure 2.3: Salida

3

El problema de los fumadores

Se implementa un monitor SU con distintos requerimientos. Se mantienen 3 hebras de fumadores y una de estanquero, además de mantener todas las condiciones de sincronización de la práctica anterior. Código completo proporcionado en la entrega.

3.1 | Ejemplo de ejecución

```
----------------------------------------------------------------------
Problema de los fumadores (Monitor SU).
----------------------------------------------------------------------
Estanquero : empieza a producir ingrediente (65 milisegundos)
Estanquero : termina de producir ingrediente 2
Estanquero : coloca el ingrediente 2
Fumador 2  : recoge el ingrediente
Fumador 2  : empieza a fumar (60 milisegundos)
Estanquero : empieza a producir ingrediente (95 milisegundos)
Fumador 2  : termina de fumar, comienza espera de ingrediente.
Estanquero : termina de producir ingrediente 1
Estanquero : coloca el ingrediente 1
Fumador 1  : recoge el ingrediente
Fumador 1  : empieza a fumar (89 milisegundos)
Estanquero : empieza a producir ingrediente (56 milisegundos)
Estanquero : termina de producir ingrediente 1
Estanquero : coloca el ingrediente 1
Fumador 1  : termina de fumar, comienza espera de ingrediente.
Estanquero : empieza a producir ingrediente (91 milisegundos)
Fumador 1  : recoge el ingrediente
Fumador 1  : empieza a fumar (122 milisegundos)
```

Figure 3.1: Salida

4

El problema de los lectores y escritores

Se crea un monitor llamado Lec_Esc con varias variables permanentes que nos permiten controlar cuando un escritor está escribiendo y el número de lectores leyendo en un momento dado.

4.1 | Ejemplo de ejecución

```
------------------------------------------------------------------------
Problema de lectores-escritores (SU).
------------------------------------------------------------------------
Escritor 1 comienza a escribir:84 milisegundos)
Escritor 1 : termina de escribir
Lector 0 comienza a leer:100 milisegundos)
Lector 0 : termina de leer
Escritor 0 comienza a escribir:78 milisegundos)
Escritor 0 : termina de escribir
Lector 0 comienza a leer:81 milisegundos)
Lector 0 : termina de leer
Escritor 1 comienza a escribir:87 milisegundos)
Escritor 1 : termina de escribir
Lector 0 comienza a leer:55 milisegundos)
Lector 0 : termina de leer
```

Figure 4.1: Salida