Programación de Sistemas Concurrentes y Distribuidos 1ª Convocatoria curso 15/16

20 de enero de 2016

Dpto. de Informática e Ingeniería de Sistemas
Universidad de Zaragoza

Ejercicio 1 (2.5 ptos.)

Consíderese el caso de un sistema concurrente en el que intervienen procesos de dos tipos distintintos, habiendo 10 instancias de cada tipo de proceso. Se trata de resolver el problema de la 6-exclusión mutua no saturante: los procesos iteran 66 veces un bucle en el que pasan por una zona crítica, debiéndose respetar las siguientes propiedades invariantes: 1) no puede haber más de 6 procesos en la sección crítica simultáneamente; 2) no puede haber seis procesos del mismo tipo en la sección crítica simultáneamente. Una vez todos los procesos han terminado, el proceso de tipo 1 de identificador 1 informa por la salida estándar de la terminación de todos los procesos.

Se pide desarrollar el esqueleto que se muestra a continuación para que, usando semáforos, se respeten los requisitos propuestos. Es importante que las sincronizaciones restrinjan la concurrencia lo menos posible.

```
                                ....
Process T1(i: 1..10)                        Process T2(i: 1..10)

    ....                                      ....
       //sección no crítica                       //sección no crítica
       //protocolo de entrada                     //protocolo de entrada
       //sección crítica                          //sección crítica
       //protocolo de salida                      //protocolo de salida
    ....                                      ....
End Process                                 End Process
```

Ejercicio 2 (2.5 ptos.)

Se pide resolver el mismo problema que en el ejercicio 1, para el caso de un sistema distribuido en el que los procesos interaccionan mediante comunicación síncrona, y utilizan un proceso servidor para el control de acceso a la sección crítica. Para ello hay que:

- Definir la estuctura de comunicación necesaria, declarando los canales
- Escribir el código de los procesos involucrados

En este caso, será el proceso servidor el que informará de la terminación de todos los procesos.

Ejercicio 3 (2.5 ptos.)

Un sistema distribuido de gestión de pujas consta de tres tipos de procesos: un gestor de pujas, varios procesos subastadores y varios procesos cliente. El gestor de pujas almacena el listado de los cincuenta productos que deben ser subastados, junto con el precio de partida para su subasta (en la estructura de datos que te parezca más adecuada). Su papel es hacer llegar la información de cada producto a los subastadores para que gestionen su subasta. Después deberá recoger el precio final alcanzado por cada producto, pudiendo en este instante terminar.

Los procesos subastadores son los encargados de gestionar la puja de un producto concreto. Para comenzar la puja, un subastador toma cualquiera de las demandas que haya comunicado el gestor. Posteriormente, pondrá disponible la información del inicio de la subasta, de manera que los clientes puedan acceder a ella. Para el producto seleccionado realiza 10 iteraciones, de manera que en cada iteración espera un tiempo (*tiempoEsperaSubastador*), para dar tiempo a los clientes interesados a hacer pujas por ese producto, pasado el cuál recoge las pujas que haya (tantas como clientes hayan mostrado interés) y actualiza el valor con la mayor puja. Al cabo de las 10 iteraciones da por terminada la puja, informando al gestor del precio final, y haciendo posible que los que han pujado conozcan el valor final del producto y el identificador del cliente que la ha ganado.

El comportamiento de los clientes es siempre el mismo. Inicialmente un cliente entra en una de las pujas abiertas y comienza a realizar ofertas. Para realizar una oferta deberá primero conocer el precio actual, publicado por el subastador y, después, aleatoriamente, decidirá si mejora o no esa oferta en una unidad de la moneda. Si la supera, realiza una nueva puja por el producto. Repite este proceso, esperando un tiempo *tiempoEsperaCliente* entre pujas, hasta que el subastador dé la puja por terminada, momento en que se informará del resultado y volverá a buscar una nueva puja en la que participar conforme el comportamiento anterior.

Se pide programar el sistema anterior para un configuración que conste de 100 clientes y 5 procesos subastadores. El gestor publicará los 50 productos diferentes, que serán identificados de forma única por un identificador de producto. La comunicación y sincronización entre los procesos estará basada en Linda. Los protocolos definidos para guiar la interacción entre los procesos deberán ser representados y explicados de forma concisa y clara.

Ejercicio 4 (2.5 ptos.)

Una empresa de alquiler de bicicletas dispone para una ciudad dada de un punto de alquiler con diez bicicletas individuales y cinco bicicletas tándem (para dos personas).

Una particularidad del funcionamiento del proceso de alquiler es el criterio que utilizan para asignar las bicicletas a los clientes. Cuando un cliente solicita una bicicleta, se le proporciona una bicicleta individual, si la hay disponible. En caso contrario se le proporcionará un tándem cuando un segundo cliente solicite una bicicleta. De esta forma, los dos clientes podrán conocerse y disfrutar de un agradable paseo. No obstante, si antes de llegar un segundo cliente una bicicleta individual es devuelta, ésta será asignada al cliente que estaba esperando.

El sistema de alquiler consta de 100 clientes registrados. Su comportamiento se repite de manera indefinida: en primer lugar, solicitan una bicicleta; una vez han logrado su alquiler, dan un agradable paseo y, posteriormente, devuelven la bicicleta.

Se pide implementar, utilizando la notación presentada en las sesiones de aula, un programa concurrente que simule el funcionamiento del sistema descrito. Los requisitos de sincronización del sistema deberán ser resueltos por medio de monitores.