Programación de Sistemas Concurrentes y Distribuidos 2ª Convocatoria curso 12/13

13 de septiembre de 2013

Dpto. de Informática e Ingeniería de Sistemas
Universidad de Zaragoza

Ejercicio 1 (2.0 ptos.)

Considérese el siguiente programa concurrente:

```
-----------------------------------------------------------
semaphore mt_2 := 2

process P(i:1..3)::
   loop forever
      wait(mt_2)
      --SC
      signal(mt_2)
   end loop
end process
-----------------------------------------------------------
```

El ejercicio pide:

- Dar un modelo red de Petri correspondiente al programa
- A partir del modelo anterior, construir su espacio de estados
- Basándonos en el espacio de estados del modelo, responder razonadamente si se cumplen las siguiente propiedades:

- El programa nunca de bloquea, ni total ni parcialmente
- Nunca los tres procesos pueden estar simultáneamente en la sección crítica, pero sí que puede llegar a haber dos
- El programa es equitativo, en el sentido de que en toda historia, todo proceso acaba ejecutándose

Ejercicio 2 (2.0 ptos.)

Un hotel dispone de una sala de ordenadores que consta de 10 máquinas dedicadas exclusivamente a la consulta del correo electrónico y 5 máquinas más para acceder únicamente a servicios turísticos de la zona. En el hotel hay actualmente alojados 100 *huéspedes*. Todos se comportan de forma similar durante su estancia. La mañana la pasan en la piscina, aunque en algún momento concreto se acercan a la sala de ordenadores a consultar el correo. Una vez leído, vuelven a la piscina. Al mediodía, vuelven de nuevo a la sala de ordenadores para planificar su excursión vespertina, consultando los servicios turísticos accesibles a través de la red. Finalmente, dedican la tarde a hacer la excursión planificada.

Dado el uso continuo que se hace de la sala de ordenadores, el hotel tiene contratado a un *administrador de sistemas*. Éste lleva control de los ordenadores que ha mantenido. Primero revisa una a una y de forma ordenada las máquinas dedicadas al correo electrónico. Posteriormente, realiza la misma tarea con las máquinas de consulta de servicios turísticos. Una vez finalizada esta revisión, repite el comportamiento previo.

Se pide escribir un programa concurrente que simule el funcionamiento del sistema descrito. En primer lugar se deberá programar un monitor que gestione el uso de las máquinas disponible y la sincronización necesaria entre los huéspedes y el administrador. Desde el punto de vista del monitor, todos los procesos que forman en sistema (huéspedes y administrador) tienen la misma prioridad a la hora de asignarles una máquina libre que se ajuste a sus necesidades. Posteriormente, se deberán programar un proceso huésped y un proceso administrador, conforme a las condiciones previamente descritas. Dado que el administrador tiene interés en hacer uso de una máquina concreta en cada ocasión, estás deberán estar identificadas de forma única desde el punto de vista del monitor (que es quien las gestiona) y del propio proceso administrador.

Finalmente, se ofrecen dos pistas útiles de cara al diseño y programación de la solución:

- Cuando el proceso huésped solicita usar una máquina, el monitor debe identificarle qué máquina concreta le ha sido asignada. El proceso guardará este identificador localmente para especificar posteriormente qué máquina libera.
- El monitor necesita identificar las máquinas y saber cuáles están siendo usadas. Esto puedo ser programado por medio de una estructura tipo array de booleanos,

donde el índice representa el identificador de la máquina.

Ejercicio 3 (2.0 ptos.)

Debido a las múltiples averías, en la facultad hay disponible un único baño. Ante esta circunstancia, se ha decidido que temporalmente sea utilizado tanto por hombres como por mujeres. No obstante, se ha restringido que personas de ambos sexos puedan utilizar simultáneamente el baño. De todas formas, el baño es suficientemente grande como para ser utilizado por todos los hombres o todas mujeres que están en la facultad.

Dentro del edificio, actualmente hay 50 hombres y 50 mujeres. Independientemente del sexo, todos ellos se comportan de una manera similar. Durante un número indefinido de veces, trabajan un tiempo aleatorio, entran al baño (si es posible, de manera inmediata; y, caso contrario, esperan hasta que sea posible), están dentro un tiempo cualquiera, y salen del baño para seguir con su trabajo. Todas las personas conocen las nuevas normas de uso del baño.

Se pide:

- Programar un sistema concurrente que simule el sistema anteriormente descrito. El sistema constará de dos tipos de procesos: los procesos *hombre* y los procesos *mujer*. La comunicación y coordinación de los procesos para garantizar las condiciones del problema debe ser programada utilizando Linda.
- Después de unos días se ha detectado que no todas las personas están respetando las condiciones de uso del baño. Por este motivo la dirección de la facultad ha decidido que una persona de administración controle el acceso y vele por el cumplimiento de las normas. Programar un sistema cliente-servidor que simule el comportamiento de nuevo sistema. La persona de administración actuará como servidor y los hombre y mujeres que trabajan en el edificio como clientes. La comunicación entre todos ellos debe ser programada en base a Linda.

Ejercicio 4 (2.0 ptos.)

La figura 1 muestra la organización de un sistema en el que mediante siete procesos $P_x, P_y, P_w, P_z, P_{xy}, P_{wz}, P_{xywz}$, más un proceso coordinador P_{coord}, se evalúa la expresión $x^2 + y^3 - 2w + 3z^7$, donde los parámetros x, y, w y z son números reales. Para ello, en un bucle infinito, el proceso coordinador lee de la entrada estándar los valores de los parámetros, los suministra a los procesos P_x, P_y, P_w, P_z y espera el resultado que le suministrará el proceso P_{xywz} para mostrarlo por la salida estándar. Cada proceso realiza un cálculo parcial, que suministra a otro proceso, según se esquematiza en la figura.

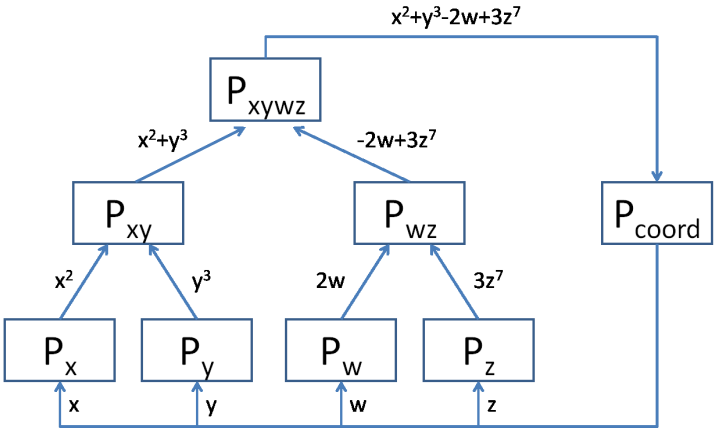

Figura 1: Esquema del flujo de datos entre los procesos

El ejercicio pide:

- Definir la red de canales síncronos para resolver el problema
- Escribir del código de los ocho procesos, de acuerdo a la especificación dada

Ejercicio 5 (2.0 ptos.)

En este ejercicio se pide implementar un programa concurrente en Java que se comporte como la siguiente solución al problema de los fílósofos.

```
-------------------------------------------------------------
process filosofo(i:1..n)
    loop
        --pensar
        tenedores.coger(i)
        --comer
        tenedores.dejar(i)
    end loop

monitor tenedores
    boolean array[1..n] ocupado := (1..n,FALSE)
    condition liberados

    operation coger(natural i)

        while (ocupado(i) OR ocupado(i+1))  --suma módulo n
            wait(liberados)
```

```
        ocupado(i) := TRUE
        ocupado(i+1) := TRUE   --suma modulo n


   operation dejar(natural i)

        ocupado(i) := FALSE
        ocupado(i+1) := FALSE   --suma modulo n
        signal_allC(liberados)
-------------------------------------------------------------
```

Notas

Es importante tener presente los siguientes aspectos:

- Las soluciones planteadas deben ser explicadas y justificadas: describir las ideas que se van a aplicar, comentar código cuando lo haya, usar nombres de variables significativos, etc.
- La presentación del examen debe ser limpia y ordenada, la letra legible, etc.