Software para Sistemas empotrados y dispositivos móviles

Febrero 2014

Grado Ing. Software

1. Diseñar e implementar un programa para controlar el aforo de una discoteca. Existen una entrada y una salida independientes en el recinto y cada una está controlada por un torno. Cada vez que una persona entra o sale del recinto, se incrementará o decrementará el contador de personas del recinto.

El sistema será simulado por tres tareas (hebras), de las cuales las dos primeras están encargadas del control de cada torno de entrada y salida, y la tercera tarea (hebra) será la encargada de gestionar y monitorizar la cuenta de personas en el recinto.

La tarea de control de torno de entrada simulará, mediante una tarea periódica con periodo de 1 sg, la llegada de personas al recinto, y enviará, con una probabilidad del 60%, una **señal** adecuada a la tarea de gestión de cuenta.

La tarea de control de torno de salida simulará, mediante una tarea periódica con periodo de 2 sg, la salida de personas del recinto, y enviará, con una probabilidad del 40%, una **señal** adecuada a la tarea de gestión de cuenta.

La tarea de gestión y monitorización de la cuenta de personas es una tarea que se encuentra, de forma permanente, a la espera de recibir notificaciones (señales) de las tareas de control de tornos, y actualiza el contador de personas según las señales recibidas. Además, también mostrará el valor del número de personas en el recinto

Diseñar la solución con los mecanismos de temporizadores y hebras de POSIX. La gestión de las tareas periódicas de control de tornos se realizará mediante temporizadores (create_timer, etc).

Para gestionar la probabilidad en la generación de señales de entrada y salida, se puede generar un número aleatorio entre 0 y 100 (*rand()%101*) y comprobar si el número aleatorio

supera un determinado umbral. Es necesario inicializar el generador de números aleatorios al principio del programa mediante *srand(time(NULL)).*

2. Se desea implementar el control de la temperatura para 2 depósitos de petróleo de una refinería (depósito 1 y depósito 2). **Cada** depósito lleva asociado una **tarea de control,** para vigilar la temperatura. Cada tarea realiza de forma periódica (según el periodo y plazo indicado en la tabla) el siguiente procedimiento:

- Calcula el valor de la temperatura del depósito. Para ello, los valores de temperatura se pueden generar aleatoriamente, inicializando el generador de números aleatorios al principio del programa mediante *srand(time(NULL)), y generando los números aleatorios dentro del rango [MIN..MAX] como MIN+(rand()%(MAX-MIN+1))*. Nota: esta generación de números aleatorios debe protegerse mediante exclusión mutua.

- Si la temperatura es menos del 80% de la temperatura máxima MAX, mostrará un mensaje de "Temperatura óptima"

- Si la temperatura del depósito es mayor de un 80 % y menor de un 90% de la temperatura máxima MAX, mostrará un mensaje de "Temperatura Alta".

- En el caso de que se superara el 90% de la temperatura máxima MAX, mostrará un mensaje de aviso "Temperatura extrema".

Por la disposición de los depósitos, el depósito 1 es más crítico que el depósito 2, por lo que la tarea de control del primer depósito es prioritaria sobre el segundo. Además, existe una tarea de monitorización, con mínima prioridad *MonitorTemp,* que informa periódicamente de la temperatura de los depósitos con un periodo de 4 s.

Los requisitos temporales de las tareas son:

Tareas	**Periodo**	**Plazo**
Depósito 1	2000 ms	1500 ms
Depósito 2	3000 ms	2500 ms
Monitor	4000 ms	4000 ms

Nota 1.- Asignar prioridades en función de los requisitos temporales y recordar que se debe limitar la inversión de prioridades.

Nota 2.- El comportamiento periódico de las hebras deberá diseñarse con *clock_gettime* y *clock_nanosleep*.