UNIVERSIDAD DE MÁLAGA

Software para Sistemas empotrados y dispositivos móviles

Febrero 2020

Grado Ing. Software

(5 puntos) EJERCICIO 1

Diseñar e implementar un programa para controlar la salud estructural de un viaducto. El sistema activa alarmas cuando los datos generados por un acelerómetro (X,Y,Z) detectan valores fuera de un rango. Supondremos que monitorizaremos tres zonas críticas de la infraestructura. Un monitor recibe las notificaciones y muestra por pantalla los avisos de vibraciones fuera de rango.

El sistema será simulado por cuatro tareas (hebras), de las cuales las tres primeras están encargadas de recibir los datos del acelerómetro y generar los avisos correspondientes, y la cuarta tarea (hebra) será la encargada de mostrar la información relativa a dichos avisos.

Las tres tareas de gestión de datos de los sensores son periódicas, con periodo de 1 sg, 2sg y 3 sg y generarán de forma aleatoria los valores X,Y,Z para simular la toma de datos del sensor, y en el caso de que no estén en los límites (INIX,FINX),(INIY,FINY),(INIZ,FINZ), enviará una **señal** adecuada y diferenciada a la tarea que muestra la información. Cada tarea deberá distinguir de qué zona se realiza el aviso.

La tarea de monitorización se encuentra, de forma permanente, a la espera de recibir notificaciones (señales) de las tareas de toma de datos, y mostrará el aviso, diferenciado por zona, según las señales recibidas.

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

Se puede generar un número aleatorio entre 0 y N (*rand() % (N+1)*) y comprobar si el número aleatorio se encuentra en un determinado rango. Es necesario inicializar el generador de números aleatorios al principio del programa mediante *srand(time(NULL)).*

(5 puntos) EJERCICIO 2

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*.
Nota 3.- Es necesario inicializar el generador de números aleatorios al principio del programa mediante *srand(time(NULL)).*

Supongamos que tenemos el mismo sistema del ejercicio 1, formado por 4 tareas (con requisitos temporales en la tabla adjunta), tres de ellas reciben datos de los sensores y la cuarta muestra ahora los datos de las aceleraciones de cada sensor y los avisos del sistema siguiendo el siguiente procedimiento:

- Si algunos de los valores X,Y,Z de la aceleración supera el 50% de del rango máximo (FINX,FINY,FINZ) o es menor en un 50% sobre el mínimo (INIX,INIY,INIZ) mostrará un mensaje de "Posible fallo estructural"
- En otro caso, el mensaje será, "Estructura correcta"

Los requisitos temporales de las tareas son:

Tareas	**Periodo**	**Plazo**
Zona 1	2000 ms	1500 ms
Zona 2	3000 ms	2500 ms
Zona 3	4000 ms	4000 ms
Monitor	5000 ms	5000 ms

Problema (10 puntos)

Consideremos el siguiente conjunto de tareas:

Tarea	**Periodo**	**Plazo**	**Tiempo de ejecución en el caso peor**	**Recursos utilizados**
A	90	70	2	R1,R3
B	100	100	9	R1,R2,R5
C	30	30	4	R4,R5
D	1000	600	11	R1,R3
E	150	95	12	R1,R2,R3,R4

Los tiempos de ejecución en el caso peor de los recursos indicados en la tabla son los siguientes:

R1	0.5
R2	0.6
R3	0.4
R4	0.25
R5	0.1

Calcular los factores de bloqueo suponiendo que el planificador usa **techo de prioridad** y calcular el tiempo de respuesta de las tareas del sistema.