Máster en II, curso MHDTR: Práctica 1

Objetivos:

- Practicar el desarrollo de aplicaciones con tareas periódicas y observar su comportamiento bajo diferentes esquemas de planificación en monoprocesadores

Descripción:

- Utilizar los threads del sistema operativo y los mecanismos de planificación para ejecutar diferentes conjuntos de tareas periódicas con requisitos temporales estrictos
- Verificar el comportamiento temporal bajo dos esquemas de planificación (prioridades fijas y EDF) y diferentes condiciones de ejecución

Actividad 1

Desarrollar una aplicación con 4 threads periódicos con las siguientes características (tiempos en segundos):

Thread	C	T	D
τ_1	1.0	3.0	3.0
τ_2	1.0	4.0	4.0
τ_3	0.9	5.0	5.0
τ_4	0.2	6.0	6.0

Los plazos son iguales a los periodos

Se deben realizar dos versiones de la aplicación:

- una con planificación por prioridades fijas y asignación de prioridades RMA
- la otra con planificación por prioridades dinámicas EDF

Actividad 1 (cont.)

Ejecutar las dos versiones de la aplicación bajo dos circunstancias diferentes:

- arranque de los threads con lectura inicial del reloj en el propio thread
- creación del instante crítico (arranque de los threads en el mismo instante de tiempo)

Ir observando los tiempos de respuesta que se van obteniendo, y después del tiempo suficiente anotar los tiempos de respuesta de peor caso de los threads (para todas las ejecuciones). Mostrar los resultados al profesor.

Comentar los resultados en relación con la planificación de los threads (observar el orden en el que ejecutan los threads), verificar si se cumplen los plazos y comparar los tiempos de respuesta observados en las diferentes ejecuciones.

Actividad 2

Modificar las características temporales de la aplicación de la Actividad 1 de acuerdo con la siguiente tabla (tiempos en segundos):

Thread	C	T	D
τ_1	1.0	3.0	3.0
τ_2	1.0	4.0	4.0
τ_3	0.9	5.0	5.0
τ_4	0.9	6.0	6.0

En este caso se ha aumentado la carga del sistema:

- calcular la nueva utilización del sistema y compararla con la de la actividad anterior

Probar de nuevo la ejecución de las dos versiones de la aplicación (a partir del instante crítico, por ejemplo) y anotar y comentar los resultados como en la actividad anterior.

Actividad 3

Modificar de nuevo las características temporales de la aplicación de la Actividad 1 de acuerdo con la siguiente tabla (tiempos en segundos):

Thread	C	T	D
τ_1	1.0	3.0	3.0
τ_2	1.0	4.0	4.0
τ_3	0.9	5.0	2.5
τ_4	0.9	6.0	2.0

En este caso hay plazos inferiores a los periodos:

- en la versión EDF añadir un nuevo campo a la `struct periodic_data` para que pueda manejar el *deadline*

Probar de nuevo la ejecución de las dos versiones de la aplicación (a partir del instante crítico, por ejemplo) y anotar y comentar los resultados como en las actividades anteriores.

Actividad 4

Modificar de nuevo las características temporales de la aplicación de la Actividad 1 de acuerdo con la siguiente tabla (tiempos en segundos):

Thread	C	T	D
τ_1	1.0	3.0	3.0
τ_2	1.0	4.0	4.0
τ_3	1.5	5.0	5.0
τ_4	1.5	6.0	6.0

En este caso se ha sobrecargado el sistema:

- calcular la nueva utilización del sistema y compararla con la de las actividades anteriores

Probar de nuevo la ejecución de las dos versiones de la aplicación (a partir del instante crítico, por ejemplo) y anotar y comentar los resultados como en las actividades anteriores.

Características de los programas

Se proporciona el software básico en el que se programan 2 threads periódicos tanto para prioridades fijas como para EDF.

Cada thread:

- indica el comienzo de ejecución con un mensaje
- ejecuta durante su tiempo ejecución de peor caso (C)
- indica el final de su ejecución con un mensaje
- calcula su tiempo de respuesta y lo anota

Además, el programa principal:

- arranca los threads y
- entra en un lazo que cada 5 segundos escribe los tiempos de respuesta de peor caso de los threads

Carga simulada de tiempo de ejecución

Se utiliza la función `eat` de carga simulada de tiempo de ejecución del fichero `eat.h` en MaRTE OS:

```
void eat(const struct timespec *cpu_time);
```

POSIX struct timespec

```
struct timespec {
   time_t tv_sec; // segundos
   long tv_nsec; // nanosegundos
}
```

- mide tiempo absoluto desde la Época,
- o un intervalo relativo

Compilación y ejecución en MaRTE OS

Para compilar y ejecutar el programa desarrollado con MaRTE OS sobre máquina desnuda:

- Ajustar la variable PATH ejecutando
  - `module load marte/x86`
- Compilar el programa con `mgcc`
- Copiar el programa en el servidor
  - `cp el_programa /net/lctrserver/puesto_XX`
    `(XX es el número del puesto del laboratorio)`
- Encender el computador (PC industrial) si es la primera vez o reiniciarlo en las sucesivas pruebas

NOTA: se puede compilar también para ejecutar como un proceso de Linux ajustando el PATH: `module load marte/linux`