

Sistemas Concurrentes y Distribuidos.

Tema 4. Introducción a los sistemas de tiempo real.

Dpt. Lenguajes y Sistemas Informáticos
ETSI Informática y de Telecomunicación
Universidad de Granada

Curso 13-14

Índice

Sistemas Concurrentes y Distribuidos.

Tema 4. Introducción a los sistemas de tiempo real.

- 1 Concepto de sistema de tiempo real. Medidas de tiempo y modelo de tareas.
- 2 Esquemas de planificación

Indice de la sección

Sección 1

Concepto de sistema de tiempo real. Medidas de tiempo y modelo de tareas.

1.1. Definición, tipos y ejemplos

1.2. Propiedades de los Sistemas de Tiempo Real

1.3. Modelo de Tareas

Sistemas de Tiempo Real

- Constituyen un tipo de sistema en el que la ejecución del sistema se debe producir dentro de unos plazos de tiempo predefinidos para que funcione con la suficiente garantía.
- En un sistema además concurrente será necesario que todos los procesos sobre un procesador o sobre varios se ejecuten en los plazos de tiempo predefinidos.

Definición de un Sistema de Tiempo Real

Stankovic (1997) da la siguiente definición:

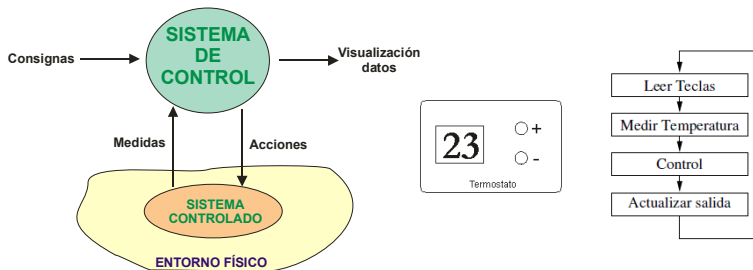
Un sistema de tiempo real es aquel sistema cuyo funcionamiento correcto depende no sólo de resultados lógicos producidos por el mismo, sino también del instante de tiempo en el que se producen esos resultados

Corrección Funcional + Corrección Temporal

El no cumplimiento de una restricción temporal lleva a un **fallo** del sistema

Ejemplo: Sistema de Control

- Objetivo: Ejecutar el lazo de control del sistema en instantes de tiempo prefijados.
- Condición de tiempo real: no puede haber retrasos en la ejecución del lazo de control, ya que afecta al rendimiento y provoca pérdida de estabilidad.

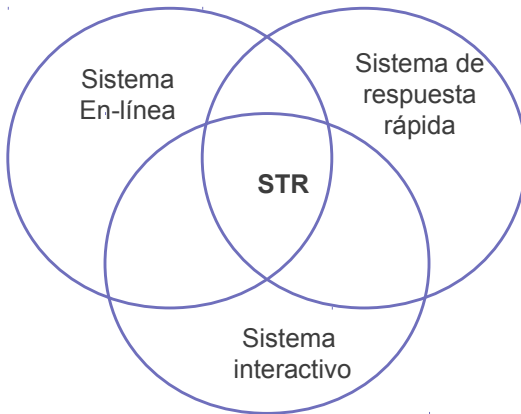


Tipos de Sistemas de Tiempo Real

Habitualmente suele asociarse la denominación de sistemas de tiempo real a los siguientes tipos de sistemas:

- **Sistema “en-línea”:** Siempre está disponible, pero no se garantiza una respuesta en intervalo de tiempo acotado.
 - Ejemplos: Cajeros automáticos, sistemas de reservas de vuelo.
- **Sistema interactivo:** El sistema ofrece una respuesta en un tiempo, aunque no importa el tiempo que necesita para su ejecución.
 - Ejemplos: Reproductor DVD, sistema aire acondicionado, juegos, ..
- **Sistema de respuesta rápida:** El sistema ofrece una respuesta en el menor tiempo posible y de forma inmediata.
 - Ejemplos: Sistema anti incendios, alarmas, etc.

Tipos de Sistemas de Tiempo Real



Ejemplos de Sistemas de Tiempo Real

En la actualidad hay muchos ejemplos de uso de Sistemas de Tiempo Real, podemos citar algunos de ellos:

- ▶ **Automoción:** sistema de ignición, transmisión, dirección asistida, frenos antibloqueo (ABS), control de la tracción, ...
- ▶ **Electrodomésticos:** televisores, lavadoras, hornos de microondas, reproductores de videos o DVDs, sistemas de seguridad y vigilancia, ...
- ▶ **Aplicaciones aeroespaciales:** control de vuelos, controladores de motores, pilotos automáticos, ...
- ▶ **Equipamiento médico:** como sistemas de monitorización de anestesia, monitores ECG,
- ▶ **Sistemas de defensa:** como sistemas radar de aviones, sistemas de radio, control de misiles, ...

Propiedades de un STR

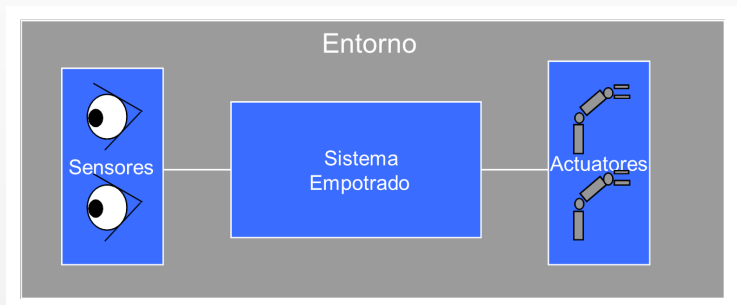
Al depender la corrección del sistema de las restricciones temporales, los sistemas de tiempo real tienen que cumplir una serie de propiedades:

- **Reactividad.**
- **Predecibilidad.**
- **Confiabilidad.**

a continuación veremos cada una de estas propiedades con más detalle.

Propiedad: Reactividad

El sistema tiene que interaccionar con el entorno, y responder de la manera esperada a los estímulos externos dentro de un intervalo de tiempo previamente definido.



Propiedad: Predecibilidad

Tener un comportamiento predecible implica que la ejecución del sistema tiene que ser determinista, y por lo tanto, se debe garantizar su ejecución dentro del plazo de tiempo definido.

- ▶ Las respuestas han de producirse dentro de las restricciones temporales impuestas por el entorno (sistema controlado), y que suele ser diferente para cada proceso del sistema.
- ▶ Es necesario conocer el comportamiento temporal de los componentes software (SO, middleware, librería, etc) y hardware utilizados, así como del lenguaje de programación.
- ▶ Si no se puede tener un conocimiento temporal exacto, hay que definir marcos de tiempo acotados; p.e. conocer el tiempo de peor ejecución de un algoritmo, el tiempo máximo de acceso a un dato de E/S)

Propiedad: Confiabilidad

- ▶ La **Confiabilidad** (*Dependability*) mide el grado de confianza que se tiene del sistema. Depende de:
 - ▶ Disponibilidad (*availability*). Capacidad de proporcionar servicios siempre que se solicita.
 - ▶ Robustez o tolerancia a fallos (*fault tolerant*). Capacidad de operar en situaciones excepcionales sin poseer un comportamiento catastrófico.
 - ▶ Fiabilidad (*reliability*). Capacidad de ofrecer siempre los mismos resultados bajo las mismas condiciones.
 - ▶ Seguridad: Capacidad de protegerse ante ataques o fallos accidentales o deliberados (*safety*), y a la no vulnerabilidad de los datos (*security*).
- ▶ Cuando esta propiedad es crítica (su no cumplimiento lleva a pérdida humana o económica), el sistema se denomina sistema de tiempo real **crítico** (*safety-critical system*)
 - ▶ Ejemplos: Sistemas de Aviónica, Sistemas de automoción, Centrales nucleares, etc.

Tipos de STRs (otra clasif. distinta)

Tipo	Características	Ejemplos
No permisivos o estrictos (<i>hard</i>)	Plazo de respuesta dentro del límite preestablecido.	Control de vuelo de una aeronave. Sistema automático de frenado ABS.
Permisivos, flexibles o no estrictos (<i>soft</i>)	Aunque el plazo de respuesta es importante, el sistema funciona correctamente aunque no respondan en el plazo de tiempo fijado	Sistema de adquisición de datos meteorológicos.
Firmes	Se permiten algunos fallos en el plazo de respuesta, pero un número excesivo provoca una fallo completo del sistema	Sistema de control de reserva de vuelos. Sistema de video bajo demanda.

Los sistemas de tiempo real pueden tener componentes de las tres clases.

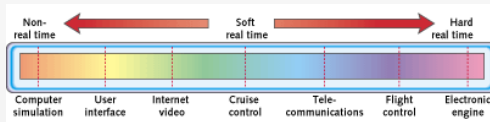
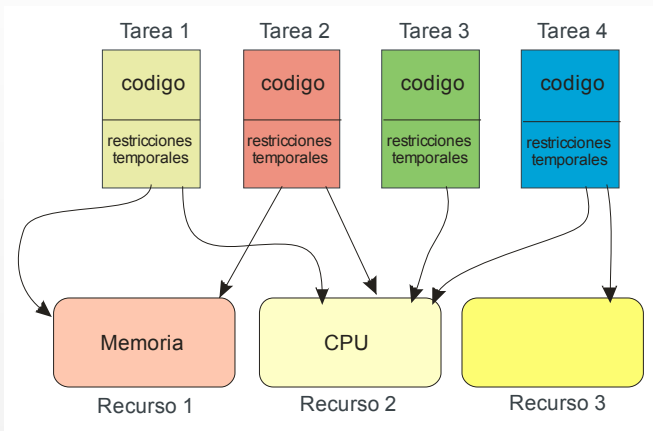


Imagen obtenida de:

<http://www.embedded.com/electronics-blogs/beginner-s-corner/4023859/Introduction-to-Real-Time>

Modelo de Tareas

- Un sistema de tiempo real se estructura en **tareas** que acceden a los recursos del sistema.



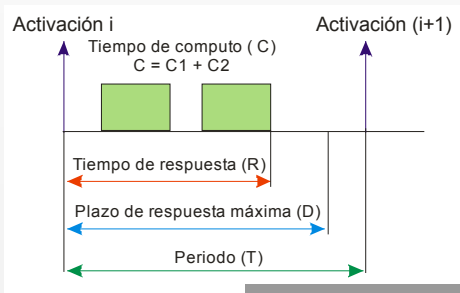
Tareas y recursos. Definición.

En los STR cabe distinguir estos dos tipos de elementos:

- **Tarea:** El conjunto de acciones que describe el comportamiento del sistema o parte de él en base a la ejecución secuencial de una sección de código (o programa). Equivalente al *proceso o hebra*.
 - La tarea satisface una necesidad funcional concreta.
 - La tarea tiene definida unas restricciones temporales a partir de los **Atributos Temporales**.
- **Recursos:** Elementos disponibles para la ejecución de las tareas.
 - Recursos Activos: Procesador, Red, ...
 - Recursos Pasivos: Datos, Memoria, Dispositivos de E/S, ...

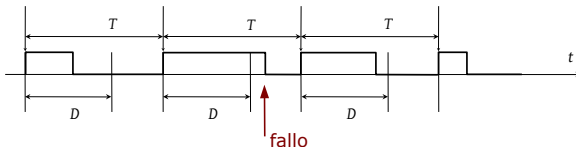
Atributos temporales

- ▶ Tiempo de computo o de ejecución (C): Tiempo necesario para la ejecución de la tarea.
- ▶ Tiempo de respuesta (R): Tiempo que ha necesitado el proceso para completarse totalmente.
- ▶ Plazo de respuesta máxima (*deadline*) (D): Define el máximo tiempo de respuesta posible.
- ▶ Periodo (T): Intervalo de tiempo entre dos activaciones sucesivas en el caso de una tarea periódica.

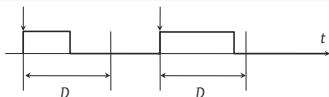


Tipos de Tareas

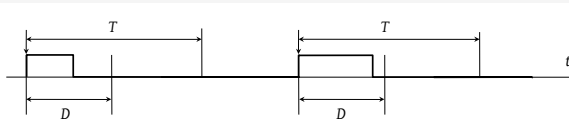
- **Periódica:** T es el período de activación de la tarea.



- **Aperiódica:**

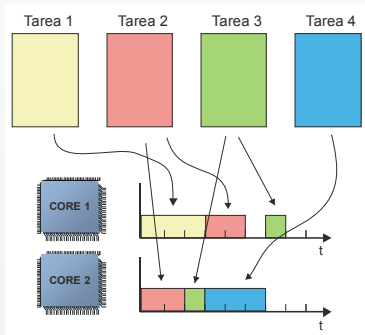


- **Esporádica:** T es la separación mínima entre eventos.



Planificación de Tareas

- La **planificación** de tareas consiste en asignar las tareas a los recursos activos de un sistema (principalmente el procesador o procesadores) para garantizar la ejecución de todas las tareas de acuerdo a un conjunto de restricciones específicas.
 - En tareas de tiempo real las restricciones suelen estar asociadas a restricciones temporales como los plazos límites.



Diseño de la planificación de tareas.

El problema de la planificación supone:

- ▶ Determinar los procesadores disponibles a los que se puede asociar las tareas.
- ▶ Determinar las relaciones de dependencias de las tareas:
 - ▶ Relaciones de precedencia que hay entre las distintas tareas.
 - ▶ Determinar los recursos comunes a los que acceden las distintas tareas.
- ▶ Determinar el orden de ejecución de la tareas para garantizar las restricciones especificadas.

Esquema de planificación de tareas

Para determinar la planificabilidad de un conjunto de tareas se requiere un esquema de planificación que cubra los dos siguientes aspectos:

- Un **algoritmo de planificación**, que define un criterio (política de planificación) que determina el orden de acceso de las tareas a los distintos procesadores.
- Un **método de análisis** (test de planificabilidad) que permite predecir el comportamiento temporal del sistema, y determina si la planificabilidad es factible bajo las condiciones o restricciones especificadas
 - Se pueden comprobar si los requisitos temporales están garantizados en **todos los casos posibles**.
 - En general se estudia el **peor comportamiento posible**, es decir, con el WCET (*Worst Case Execution Time*).

Cálculo del WCET.

- ▶ Todos los métodos de planificación parte de que se conoce el WCET (C), el tiempo de peor ejecución de cada tarea.
- ▶ ¿ Cómo podemos calcular el valor de C para cada tarea ?
- ▶ Hay dos formas de obtenerlo:
 - ▶ Medida directa del tiempo de ejecución (en el peor caso) en la plataforma de ejecución.
 - ▶ Se realizan múltiples experimentos, y se hace una estadística.
 - ▶ Análisis del código ejecutable
 - ▶ Se descompone el código en un grafo de bloques secuenciales.
 - ▶ Se calcula el tiempo de ejecución de cada bloque.
 - ▶ Se busca el camino más largo.

Restricciones temporales

Para determinar la planificación del sistema necesitamos conocer las **restricciones temporales** de cada tarea del sistema.

Aquí vemos dos ejemplos de restricciones temporales:

- ▶ $C = 2\text{ ms}$, $T = D = 10\text{ ms}$. Es una tarea periódica que se activa cada 10 ms, y se ejecuta en un tiempo máximo de 2 ms en el peor de los casos.
- ▶ $C = 10\text{ ms}$, $T = 100\text{ ms}$; $D = 90\text{ ms}$. Es una tarea periódica que se activa cada 100 ms, se ejecuta en cada activación un máximo de 10 ms, y desde que se inicia la activación hasta que concluye no puede pasar más de 90 ms.

Indice de la sección

Sección 2

Esquemas de planificación

2.1. Planificación Cíclica.

2.2. Planificación con prioridades

Tipos de Esquemas de Planificación

Los esquemas de planificación para un sistema monoprocesador son:

- **Planificación estática *off-line* sin prioridades**
 - Planificación cíclica (ejecutivo cíclico)
- **Planificación basada en prioridades**
 - Prioridades estáticas
 - Prioridad al más frecuente (RMS, *Rate Monotonic Scheduling*)
 - Prioridad al más urgente (DMS, *Deadline Monotonic Scheduling*)
 - Prioridades dinámicas
 - Proximidad del plazo de respuesta (EDF, *Earliest Deadline First*)
 - Prioridad al de menor holgura (LLF, *Least Laxity First*)

Planificación Cíclica

- ▶ La planificación se basa en construir un plan de ejecución (**plan principal**) de forma explícita y fuera de línea (*off-line*) en el que se especifica el entrelazado de las tareas periódicas de tal forma que su ejecución cíclica garantiza el cumplimiento de los plazos de las tareas.
 - ▶ La estructura de control o programa cíclico se denomina **ejecutivo cíclico**.
- ▶ El entrelazado es fijo y se define en el **plan principal** que se construye antes de poner en marcha el sistema (*off-line*).
- ▶ La duración del ciclo principal es igual al hiperperiodo
 - ▶ $T_M = \text{mcm}(T_1, T_2, \dots, T_n)$
 - ▶ se supone tiempo entero (ticks de reloj)
 - ▶ el comportamiento temporal del sistema se repite cada ciclo principal

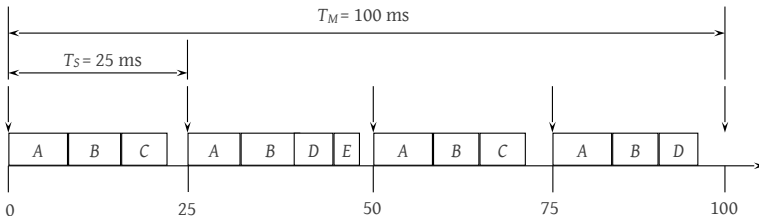
Ejemplo de planificación cíclica.

Tarea	T	C
A	25	10
B	25	8
C	50	5
D	50	4
E	100	2

- El ciclo principal dura 100 ms

$$T_M = \text{mcm}(25, 25, 50, 50, 100) = 100$$

- Se compone de 4 ciclos secundarios de 25 ms cada uno.



Implementación de la Planificación Cíclica

```
void ejecutivo_ciclico()
{
    int nciclos = 4 ,
        frame   = 0 ;
    while (true)
    {
        switch (frame)
        {
            case 0 : A; B; C;      break ;
            case 1 : A; B; D; E;   break ;
            case 2 : A; B; C;      break ;
            case 3 : A; B; D;      break ;
        }
        frame = (frame+1) % nciclos;
        wait_until_next_clock_tick();
    } // final del ejecutivo ciclico
}
```

Propiedades de la Planificación Cíclica

- ▶ No hay concurrencia en la ejecución.
 - ▶ Cada ciclo secundario es una secuencia de llamadas a **procedimientos**
 - ▶ No se necesita un núcleo de ejecución multitarea
- ▶ Los procedimientos pueden **compartir datos**.
 - ▶ No se necesitan mecanismos de exclusión mutua como los semáforos o monitores
- ▶ No hace falta analizar el comportamiento temporal.
 - ▶ El sistema es correcto por construcción.

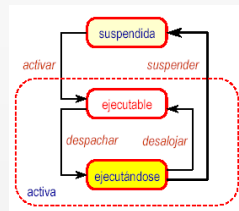
Problemas de la Planificación Cíclica

- Dificultad para incorporar tareas con periodos largos.
- Las tareas esporádicas son difíciles de tratar.
 - Se puede utilizar un servidor de consulta.
- El plan ciclo del proyecto es difícil de construir.
 - Si los periodos son de diferentes órdenes de magnitud el número de ciclos secundarios se hace muy grande.
 - Puede ser necesario partir una tarea en varios procedimientos.
 - Es el caso más general de sistemas en tiempo real críticos.
- Es poco flexible y difícil de mantener.
 - Cada vez que se cambia una tarea hay que rehacer toda la planificación.

Planificación con prioridades

- ▶ La prioridad es un mecanismo elemental para planificar la ejecución de un conjunto de tareas.
 - ▶ Es un atributo de las tareas normalmente ligado a su importancia relativa en el conjunto de tareas.
 - ▶ Por convención se asigna números enteros mayores a procesos más urgentes.
 - ▶ La prioridad de una tarea la determina sus necesidades temporales; no es importante el rendimiento o comportamiento del sistema.

- ▶ Una tarea puede estar en varios estados:
- ▶ Las tareas ejecutables se despachan para su ejecución en orden de prioridad.



Planificación RMS

- ▶ RMS (*Rate Monotonic Scheduling*): Es un método de planificación estático on-line con asignación de prioridades a las tareas más frecuentes.
- ▶ A cada tarea se le asigna una (única) prioridad basada en su periodo; cuanto menor sea el periodo (mayor frecuencia) → mayor prioridad.

$$\forall i, j : T_i < T_j \implies P_i < P_j$$

- ▶ Esta asignación de prioridades es optima en el caso de que todas las tareas sean periódicas, y el plazo de respuesta máxima D coincida con el periodo.

Nota: aquí se emplean siempre numeros enteros menores para procesos más urgentes o más prioritarios.

Ejemplo de planificación RMS (1)

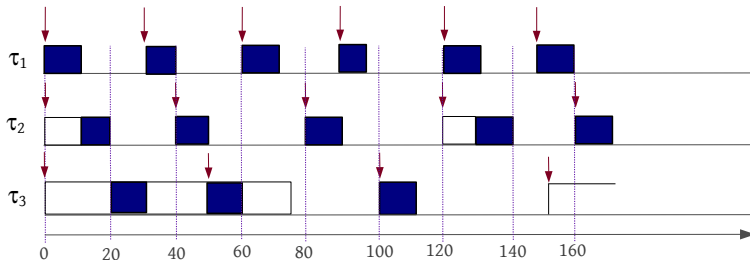
- Dada un conjunto de tres tareas con las siguientes restricciones temporales:

Tareas	C	D	T
1	10	30	40
2	5	40	40
3	9	50	50

- La aplicación del método de planificación RMS indica que la tarea 1 es la que tiene mayor prioridad, luego la 2, y luego la 3.

Ejemplo de planificación RMS (2)

- Para analizar la planificabilidad del sistema con dichas restricciones, hay que estudiar en el cronograma que:
 - Para cada tarea: $\rightarrow R_i < T_i$
- Solo hay que probar la ejecución correcta de todas las tareas en el hiperperiodo, es decir, $H = \text{mcm}(T_i)$.
 - Si se cumple en el hiperperiodo, se repetirá indefinidamente.



Planificación EDF

- EDF (*Earliest Deadline First*, primero al más urgente): La asignación de prioridad se establece una prioridad más alta a la que se encuentre más próxima a su plazo de respuesta máxima (deadline). En caso de igualdad se hace una elección no determinista de la siguiente tarea a ejecutar.
- Es un algoritmo de planificación dinámica, dado que la prioridad de cada tarea cambia durante la evolución del sistema.
- Es más óptimo porque no es necesario que las tareas sean periódicas.
- En menos pesimista que RMS.

Ejemplo de planificación EDF

Tareas	C	T	D
J1	2	5	5
J2	4	7	7

