

# Sistemas en Tiempo Real

---

## Introducción a los Sistemas en Tiempo Real

Francisco Andrés Candelas Herías

fcandela@dfists.ua.es



Universitat d'Alacant  
Universidad de Alicante

### Objetivos de la clase

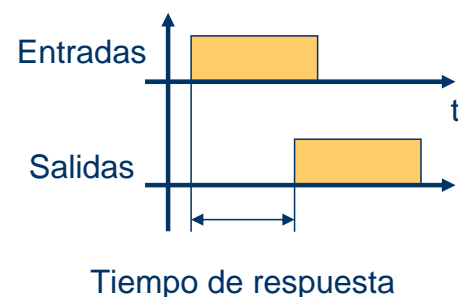
---

2

- Presentar los conceptos más importantes sobre los Sistemas de Tiempo Real (STR).
- Enseñar los distintos tipos de STR.
- Ver los pasos en el diseño y puesta en marcha de un STR.
- Introducir los conceptos de asignación y planificación de tareas en STR.

- Definiciones
- Clasificación de los STR
- Los STR y el control por computador
- Estructura de un STR
- Diseño de un STR
- Caracterización de las tareas
- Implementación de un STR

- **Sistema:** Interconexión de dispositivos en el que unas señales de entrada son transformadas para generar otras señales de respuesta.
- Un sistema presenta un **tiempo de respuesta** desde que se aplican las señales de entrada hasta que se generan las de salida.



- **Sistema de Tiempo Real:**
  - Sistema en el que el resultado obtenido depende, no solo de la ejecución de un proceso, si no también del tiempo en que se produce (J. Stankovic).
  - Sistema que debe satisfacer unos requisitos temporales, o sino se producirán consecuencias graves (Phillip A. Laplante).
- En un STR, el tiempo de respuesta es critico o, al menos, muy importante.

- **Ejemplos de aplicación de un STR:**
  - Sistemas industriales de inspección de productos.
  - Control de procesos en plantas industriales.
  - Aplicaciones de robótica. Robots cooperativos.
  - Transmisiones de información multimedia.
  - Tele-operación de equipos remotos.
  - Aviónica. Sistemas de navegación.
  - Edificios inteligentes.

- Definiciones
  - Clasificación de los STR
- 
- Los STR y el control por computador
  - Estructura de un STR
  - Diseño de un STR
  - Caracterización de las tareas
  - Implementación de un STR

- Se pueden establecer distintas clasificaciones:
  - Dependencia del hardware.
  - Según las características del sistema.
  - Según el modo de funcionamiento.

- Según la dependencia del hardware:
  - Sistemas **empotrados**.

- El STR es un componente interno de otro sistema que realiza alguna función de control. El software es dependiente del hardware.
- Ejemplo: microcontrolador que controla la mezcla de aire-gasolina en un motor de inyección electrónica.



- Según la dependencia del hardware:
  - Sistemas **orgánicos**.

- Los no empotrados. El software no depende directamente del hardware.
- Ejemplo: una aplicación de videoconferencia en tiempo real, ejecutándose en un PC.



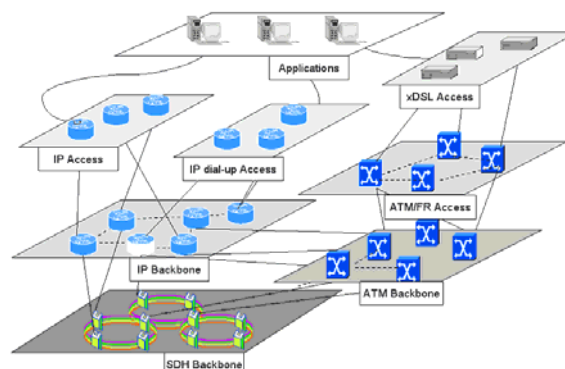
- Según las características del sistema:
  - Sistemas **críticos** (hard real-time systems).
    - El tiempo de respuesta del sistema es crítico, y tiene que estar forzosamente dentro de unos límites.
    - Es preferible un resultado de poca calidad en su tiempo, que un resultado muy bueno u óptimo fuera de tiempo.
    - Ejemplo: piloto automático de un avión, al chequear y validar la altura actual de la nave.



- Según las características del sistema:
  - Sistemas **no críticos** (soft real-time systems).
    - El tiempo de respuesta es importante, pero no crítico.
    - Pueden ofrecer un resultado muy bueno.
    - Ejemplo: aplicación de videoconferencia.

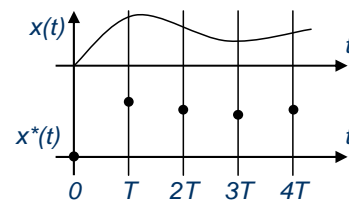
- Según el modo de funcionamiento (I):
  - Sistemas con **respuesta garantizada**.
    - El comportamiento temporal del sistema está bien definido.
    - Es necesario caracterizar bien las tareas a procesar, la carga máxima, los posibles fallos...
    - Ejemplo: sistema de control de la mezcla de aire-gasolina.

- Según el modo de funcionamiento (I):
  - Sistemas que **hacen lo que pueden** (best-effort systems).
    - No se requiere una caracterización precisa.
    - Solo sirve para sistemas no críticos.
    - Ejemplo: equipos de encaminamiento de paquetes y gestión de tráfico en redes de datos.





- Según el modo de funcionamiento (II):
  - Sistemas **dirigidos por tiempo** (time-triggered systems).
    - El procesamiento se realiza según unos instantes predeterminados de tiempo, habitualmente de forma periódica.
    - Ejemplo: procesador digital de señales (DSP) que opera una muestra de la señal de entrada cada periodo de muestreo.



- Según el modo de funcionamiento (II):
  - Sistemas **dirigidos por sucesos** (event-triggered systems).
    - El sistema realiza un procesamiento cada vez que ocurre un cambio de estado.
    - El sistema está dirigido por sucesos esporádicos.
    - Ejemplo: controlador que ejecuta determinadas rutinas en función de entradas de interrupción.
  - Hay STR que atienden sucesos esporádicos y tareas periódicas.

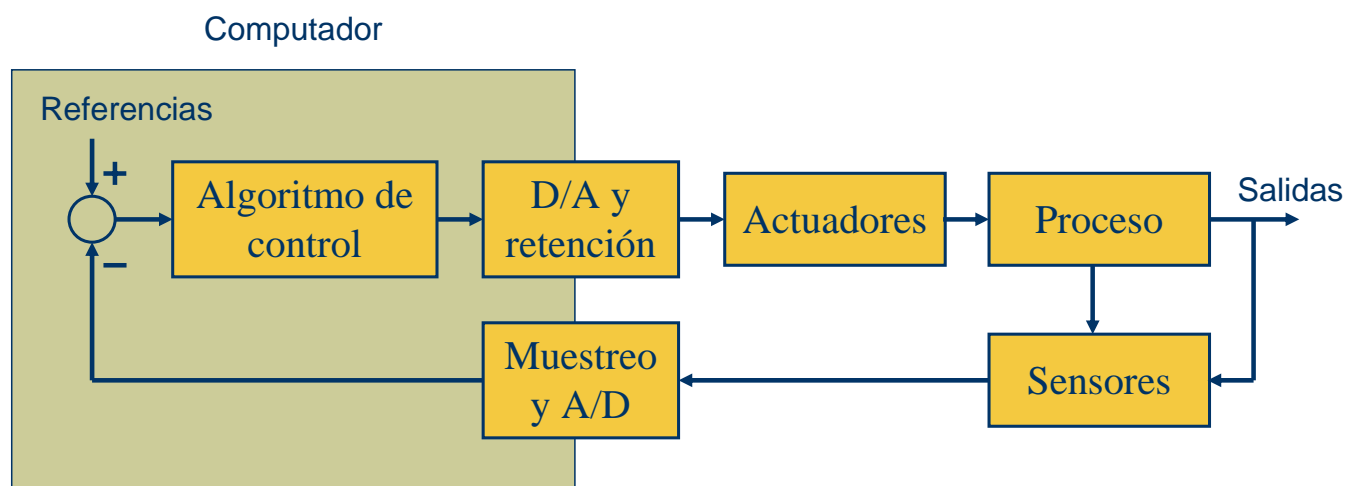


- Definiciones
  - Clasificación de los STR
  - Los STR y el control por computador
- 
- Estructura de un STR
  - Diseño de un STR
  - Caracterización de las tareas
  - Implementación de un STR

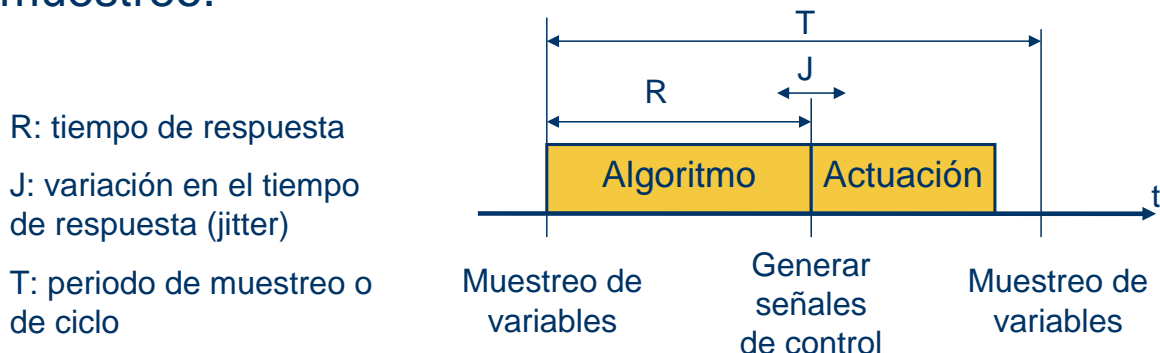
## Los STR y el control por computador

18

- Control digital directo: el control de un sistema real continuo controlado lo realiza un programa en un computador. Se requiere conversiones A/D y D/A.



- El sistema de control puede tener requisitos de tiempo real:
  - En un ciclo se muestrea las variables de proceso, y se aplica el algoritmo de control para generar los nuevos valores de control.
  - Todo ello se debe hacer dentro del periodo de muestreo.



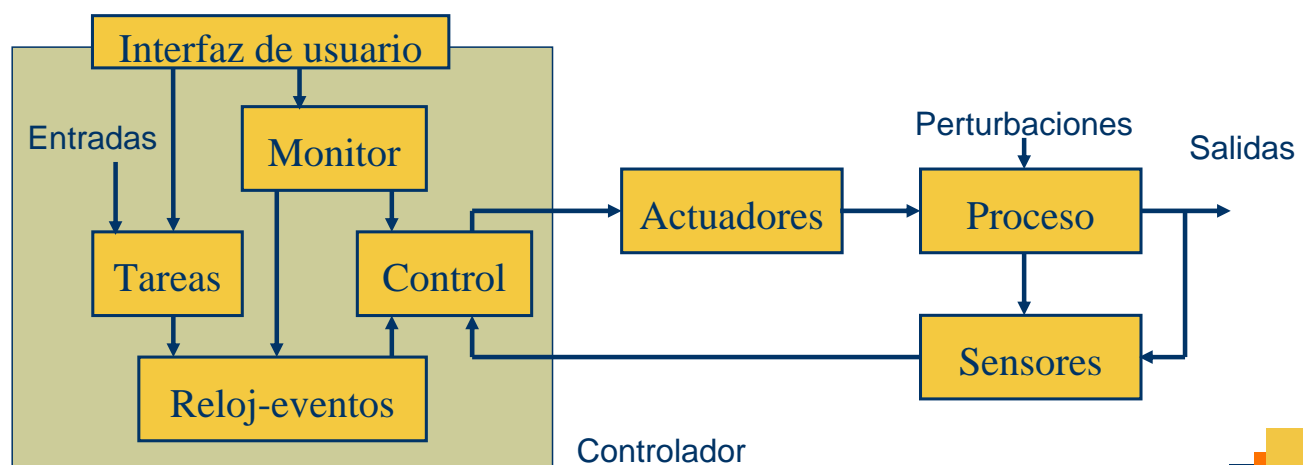
- Un sistema de control secuencial con un PLC también ofrece características de STR:
  - Se dispone de un tiempo de ciclo en el que se leen las entradas, se aplica el programa y se generan las salidas.

- Definiciones
  - Clasificación de los STR
  - Los STR y el control por computador
  - Estructura de un STR
- 
- Diseño de un STR
  - Caracterización de las tareas
  - Implementación de un STR

## Estructura de un STR

22

- Elementos de un STR:
  - Controlador. Habitualmente es un computador.
  - Proceso controlado (planta).
  - Además hace falta: sensores, actuadores e interfaz de usuario.

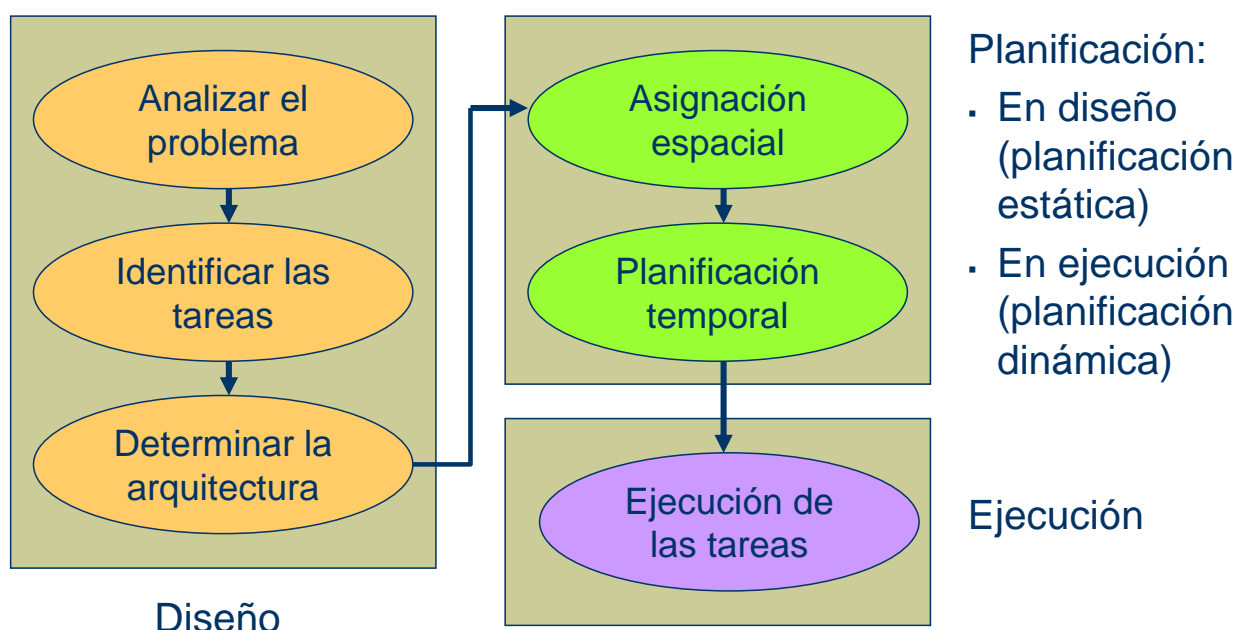


- Definiciones
  - Clasificación de los STR
  - Los STR y el control por computador
  - Estructura de un STR
  - Diseño de un STR
- 
- Caracterización de las tareas
  - Implementación de un STR

## Diseño de un STR

24

- Etapas básicas del diseño y puesta en marcha de un STR:



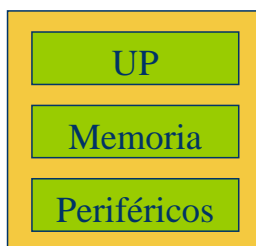
- Analizar el problema. Se debe considerar aspectos como...
  - Tipo de sistema:
    - Empotrado u orgánico.
    - Crítico o no crítico.
    - De respuesta garantizada o tipo best-effort.
    - Dirigido por tiempo o por eventos.
  - Posibles fluctuaciones significativas en la carga de actividades a realizar, que implican una demanda de procesamiento imprevisible.
  - Escasez de recursos en sistemas empotrados.
  - Necesidad de un sistema tolerante a fallos.

- Identificar las tareas: particionado software.
  - Las actividades a realizar se deben dividir o agrupar en tareas que optimicen la ejecución.
  - Hay que determinar las posibles relaciones entre tareas.
  - Para multiprocesador: se debe buscar el máximo paralelismo entre las tareas.
  - Según como se agrupen las tareas se tendrá una necesidad de comunicación diferente entre ellas.
  - En general...
    - Pocas tareas grandes: poca comunicación, y poco paralelismo.
    - Muchas tareas pequeñas: mucho paralelismos y mucha necesidad de comunicación.

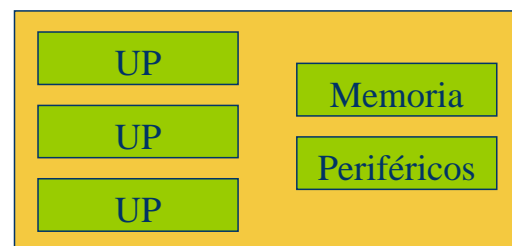
- Determinar arquitectura:
  - El tipo de tareas a ejecutar:
    - Conocidas a priori, o no predecibles.
    - Periódicas o esporádicas, interrumpibles o no, críticas o no.
    - Tareas que requieran procesadores de diferente tipo.
  - Arquitectura monoprocesador, multiprocesador o multicomputador.
  - Asignación y planificación estáticas o dinámicas.
  - Planificación dinámica centralizada o distribuida.
  - Acceso a recursos compartidos y necesidad de exclusión.

- Determinar arquitectura: procesadores.

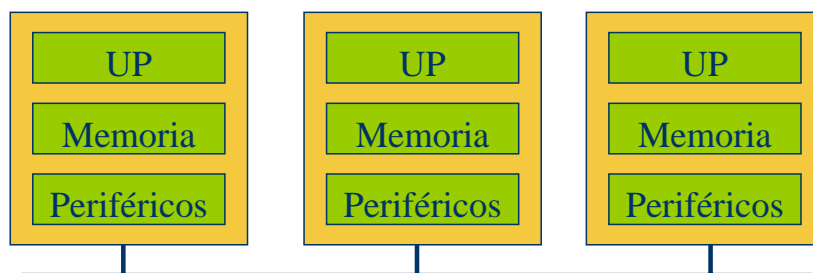
Monoprocesador



Multiprocesador centralizado



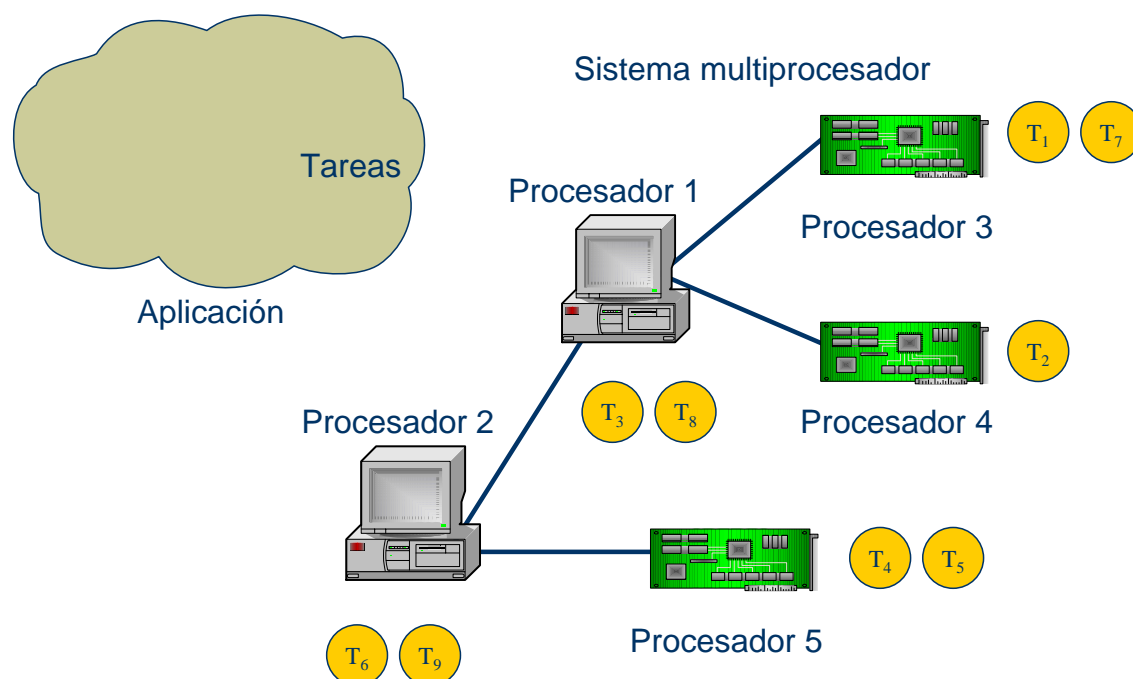
Multiprocesador distribuido (multicomputador)



- Procesadores iguales
- Procesadores diferentes

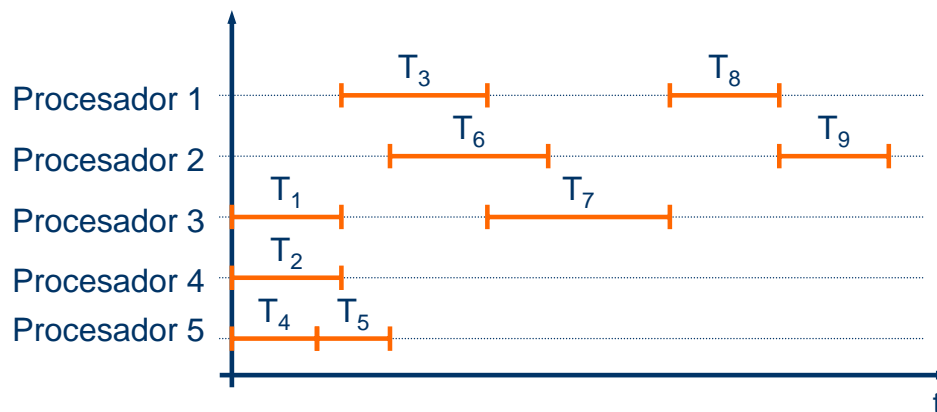
- Asignación espacial de las tareas:
  - Determinar en que procesador se ejecuta cada tarea.
  - Se hace según el resultado del particionado (tareas y relaciones entre ellas) y el hardware disponible.
  - Se pretende obtener el mejor rendimiento y disminuir en lo posible el tiempo de respuesta en la ejecución.

- Asignación espacial de las tareas:





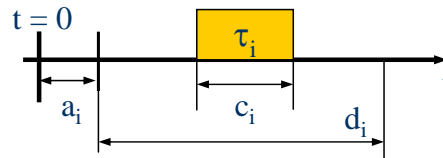
- Planificación temporal de las tareas:
  - Ordenar en el tiempo la ejecución de las tareas asignadas a cada procesador.
  - También pretende lograr el mejor rendimiento y disminuir en lo posible el tiempo de respuesta.



- Definiciones
- Clasificación de los STR
- Los STR y el control por computador
- Estructura de un STR
- Diseño de un STR
- Caracterización de las tareas
- Implementación de un STR

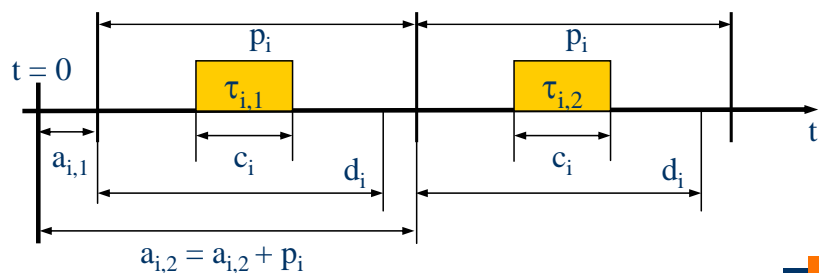
- Tipos de tareas según su generación:
  - Tareas **esporádicas** (aperiódicas). Asociadas a sistemas dirigidos por eventos.

$$\tau_i = (a_i, c_i, d_i)$$



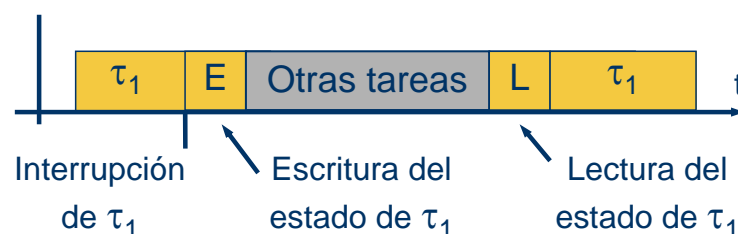
- Tareas **periódicas**. Se ejecutan de forma repetida y están asociadas a sistemas dirigidos por tiempo.

$$\tau_i = (a_i, c_i, d_i, p_i)$$



## Caracterización de las tareas

- Tipos de tareas según opción de interrupción:
  - Tareas **interrumpibles** (preemptive tasks). Su ejecución puede ser interrumpida y reanudada, lo que supone un coste adicional.



- Tareas **no interrumpibles** (non-preemptive tasks).

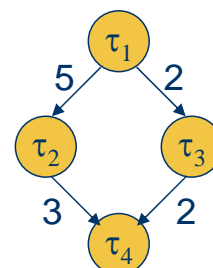
- Tipos de tareas según su importancia:
  - Tareas **críticas**. Su ejecución no puede superar su plazo de respuesta (deadline); en caso contrario se producen resultados catastróficos.
  - Tareas **no críticas**. Aunque presentan un plazo de respuesta, su superación no produce resultados catastróficos. Simplemente, se debe intentar ejecutarlas en su plazo de respuesta.

- Relaciones entre tareas.
  - Relaciones de precedencia. Habitualmente definidas con un grafo dirigido acíclico (DAG).
  - Relaciones de prioridad (orden de interrupción).
  - Relaciones de exclusión en el acceso a recursos.

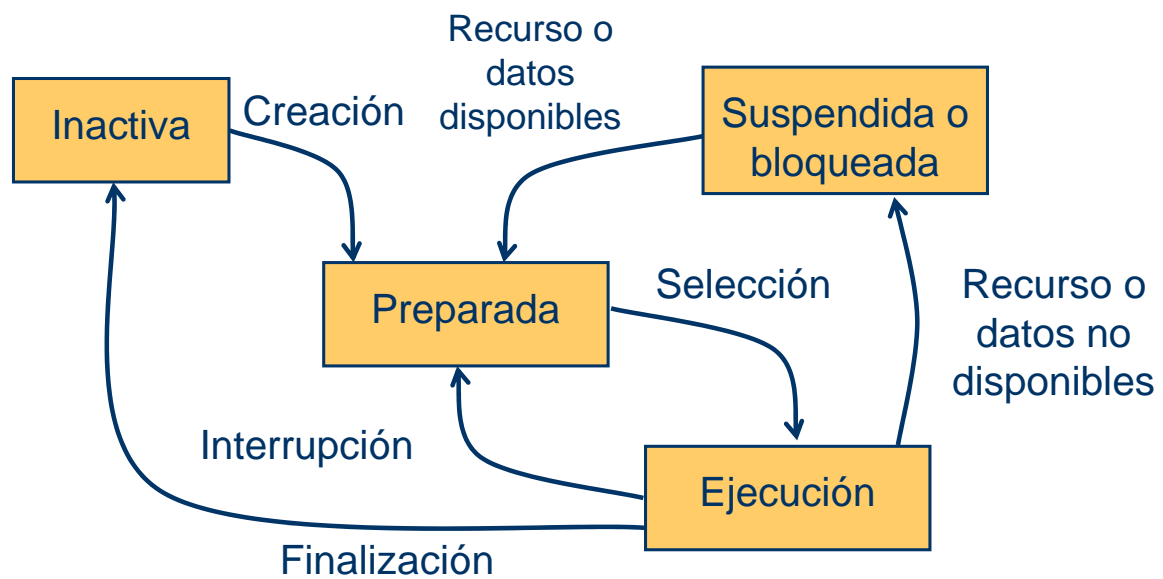
$$G = (T, E)$$

$$T = \{\tau_1, \tau_2, \dots, \tau_N\}, \quad \tau_i = (a_i, c_i, d_i)$$

$$E = \{(\tau_i, \tau_j, c_{i,j}) / \tau_i R_{\text{prec}} \tau_j \text{ con coste } c_{i,j}\}$$



- Estados de ejecución de una tarea



- Definiciones
- Clasificación de los STR
- Los STR y el control por computador
- Estructura de un STR
- Diseño de un STR
- Caracterización de las tareas
- Implementación de un STR

- Dos aspectos a elegir:
  - Sistema operativo para los procesadores.
  - Lenguajes de programación.

- El sistema operativo debe garantizar:
  - Concurrencia o ejecución paralela de tareas.
  - Medida de tiempos fiable y con buena resolución.
  - Control de la planificación de las tareas.
  - Acceso a las E/S y recursos hardware necesarios.
- Los S.O. más comunes (MS. Windows, Linux) no suelen garantizar todo esto, pero...
  - Existen versiones para STR (Windows CE, RTLinux).
  - Existen módulos que se añaden al sistema y dan capacidades de tiempo real.
- Hay S.O. específicos para STR empotrados.

- El lenguaje de programación debe ofrecer:
  - Concurrencia (procesos, hilos...) y comunicación entre procesos.
  - Control de interrupciones o eventos.
  - Comportamiento temporal analizable y control de tiempos.
  - Fiabilidad del código fuente.
  - Facilidades de depuración para múltiples procesos.

- Posibles lenguajes:
  - De bajo nivel. Eficientes, pero poco fiables y de difícil depuración.
    - Ensamblador, C.
  - De uso general. Por si solos no suelen proporcionar capacidades de tiempo real, y necesitan un S.O. que ofrezca funciones para esas capacidades.
    - C, C++, Java...
  - Concurrentes. Incluyen capacidades de concurrencia, control de tiempos, eventos... Son adecuados para sistemas complejos y que se deben actualizar.
    - Modula, Ada, Ada95.