Tema 4. Sistemas de tiempo real

Índice

[1. Definición de sistema de tiempo real 2](#_Toc93407028)

[2. Propiedades de un sistema real 2](#_Toc93407029)

[3. Modelo de tareas 2](#_Toc93407030)

[4. Recursos en un STR. Atributos. 3](#_Toc93407031)

[5. Planificación de tareas. 3](#_Toc93407032)

[6. Atributos temporales de una tarea 4](#_Toc93407033)

[7. Diseño de planificación de tareas 5](#_Toc93407034)

[8. Esquema de planificación de tareas 5](#_Toc93407035)

[9. Cálculo de WCET 5](#_Toc93407036)

[10. Factor de utilización 5](#_Toc93407037)

[11. Esquemas de planificación. Tipos. 6](#_Toc93407038)

[11.1. Planificación cíclica 6](#_Toc93407039)

[11.2. Planificación con prioridades 6](#_Toc93407040)

[11.2.1. Planificación RMS 7](#_Toc93407041)

[11.2.2. Test de planificabilidad 7](#_Toc93407042)

[11.2.3. Test de Liu & Layland 7](#_Toc93407043)

[11.2.2. Planificación EDF 8](#_Toc93407044)

[Preguntas examen 8](#_Toc93407045)

# Definición de sistema de tiempo real

Se tratan de unos sistemas en los que la ejecución del sistema se debe dar en unos plazos de tiempo. Si es concurrente, cada proceso del procesador debe ejecutarse en un plazo de tiempo. Un sistema de tiempo real, por lo tanto, debe ser **correcto funcionalmente y correcto temporalmente.**

Si falla en alguno, ocurre un fallo del sistema.

Se suele atribuir la definición a **sistemas en línea** (disponibles pero no se garantiza una respuesta en un intervalo de tiempo acotado), **sistemas interactivos** (respuesta en tiempo acotado pero no pasa nada si a veces no lo cumplen) y **sistemas de respuesta rápida** (es vital que lo cumplan).

# Propiedades de un sistema real

Las propiedades de un sistema de tiempo real son las siguientes:

1. *Reactivo*: interacciona con el entorno y da respuesta en un intervalo de tiempo previamente definido.
2. *Predecible*: la respuesta se debe dar en un tiempo acotado. Es necesario conocer muy bien los componentes hardware para conseguirlo.
3. *Confiabilidad*: mide el grado de confianza del sistema. Depende de varios factores: **disponibilidad** (capacidad de proporcionar servicios cuando se solicita), **robustez** (si falla o no falla), **fiabilidad** (ofrecer siempre los mismos resultados) y **seguridad** (capacidad para protegerse ante ataques o fallos (safety) e invulnerabilidad a los datos (security)).

Cuando el sistema depende muy fuertemente de la propiedad de confiabilidad, (pueden conllevar pérdidas humanas) se les llaman sistemas de tiempo real críticos.

# Modelo de tareas

Una **tarea** es un conjunto de acciones que describen el comportamiento del sistema o parte de él en base a la ejecución de un trozo de código de manera secuencial (equivalente a proceso o hebra).

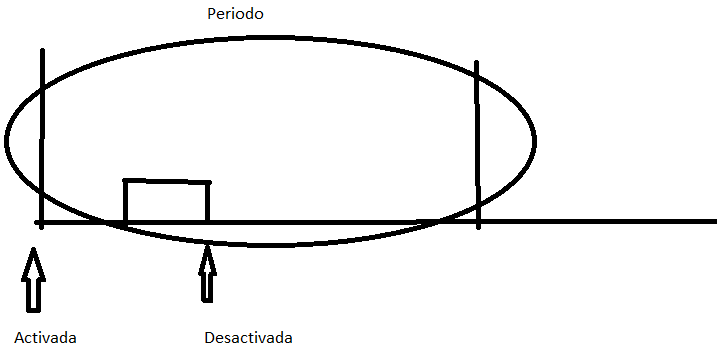
Las tareas tienen definidas restricciones temporales a partir de los **atributos temporales.**

Un STR se estructura en tareas, que acceden a recursos del sistema. Una tarea satisface una necesidad concreta.

Una tarea está **activada** si está ejecutándose o está preparada para ejecutarse. En un periodo, desde que finaliza su ejecución hasta el final del periodo se dice que está **desactivada**.

Diagrama

Descripción generada automáticamente



# Recursos en un STR. Atributos.

Son los elementos disponibles para la ejecución de las tareas (activos, como el procesador, red... y pasivos como los datos, memoria, etc.)

Asumimos que cada CPU disponible ejecuta una o varias tareas. Si se ejecutan varias, el tiempo se debe repartir entre ellas.

# Planificación de tareas.

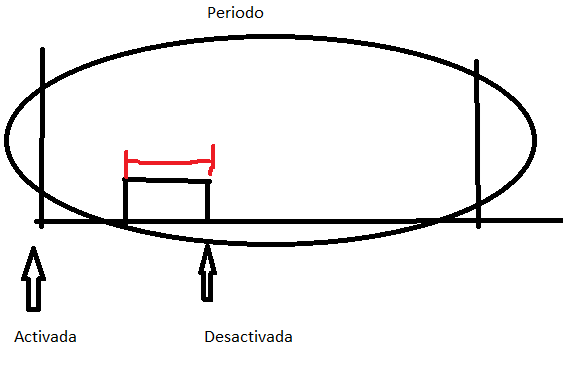
Se trata de determinar cómo se le asignan a cada tarea los recursos (principalmente procesador) de forma que se garanticen las restricciones temporales.

Diagrama

Descripción generada automáticamente

# Atributos temporales de una tarea

* Tiempo de cómputo o de ejecución ©: lo que tarda en ejecutarse la tarea.



* Tiempo de respuesta ®: Tiempo que se ha necesitado para completarse desde que se activa.
* Plazo de respuesta máxima (D): Máximo tiempo de respuesta posible. Algo como desde que se inicia la activación hasta que finaliza no puede pasar más tiempo que D.
* Periodo (T): tiempo entre dos activaciones en el caso de una tarea periódica.

Imagen que contiene Gráfico

Descripción generada automáticamente

# Diseño de planificación de tareas

El diseño supone lo siguiente:

* Determinar los procesadores disponibles.
* Determinar un orden de ejecución de las tareas para garantizar las restricciones específicas.

# Esquema de planificación de tareas

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

* Un **algoritmo de planificación (o política de planificación)** que determina el orden de acceso de las tareas a los procesadores.
* Un **Método de análisis (test de planificabilidad)** para que podamos predecir el comportamiento temporal del sistema y determinar si es planificable o no. En general, se estudia el peor comportamiento posible (WCET).

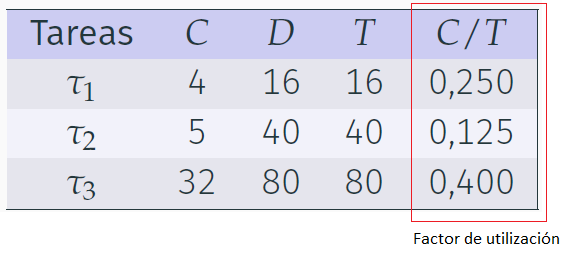
# Cálculo de WCET

El valor de WCET es el máximo valor de C para cualquier ejecución de dicha tarea. Hay dos formas de obtenerlo: **medida directa** del tiempo de ejecución (ver en cual caso tarda más en ejecutarse la tarea) y el **análisis del código.**

# Factor de utilización

Las restricciones temporales son T, C y D (las que se dan en las tablas).

El **factor de utilización** es la suma de C/T de todas las tareas. Si la sumatoria de todas es mayor que p, entonces no es planificable. Cuanto más cerca de 1 más se utilizará la CPU.



# Esquemas de planificación. Tipos.

Los tipos de planificación para monoprocesador son:

* Estática off-line sin prioridades (se conoce antes de la ejecución). **Ejecutivo cíclico.**
* Basada en prioridades de tareas:
* Estáticas: la prioridad de la tarea no cambia en tiempo de ejecución. **RMS** (se da la prioridad a la tarea con menor T) y **DMS** (se da la prioridad a la tarea con menor D)
* Dinámicas: la prioridad no está fijada. Va cambiando. **EDF** (se asigna la prioridad a la tarea con el D más próximo. En un instante dado, el que va a acabar antes). **LLF (**prioridad a tarea de menor holgura(tiempo hasta deadline – tiempo de ejecución restante)).

## Planificación cíclica

Se implementa un **plan principal** (o ciclo principal) que garantice los requerimientos temporales. Es un bucle infinito. Su duración se llama **hiperperiodo** y se calcula como el MCM de los periodos de todas las tareas.

Dentro del plan principal, se ejecuta otro bucle acotado con K iteraciones. Dentro de cada iteración (que dura siempre lo mismo) se ejecutan completamente una o varias tareas. Este se llama **bucle secundario.**

En este caso, **no hay concurrencia.** Se ejecutan todos secuencialmente repartiendo los tiempos del procesador. Además, las tareas pueden compartir datos ya que se ejecutan independientemente unas de otras (no se necesitan mecanismos de EM).

## Planificación con prioridades

Cada tarea tiene asociado un valor entero no negativo, llamado **prioridad.** Cuanto mayor es el número, mayor es la prioridad. El orden de ejecución de las tareas se determina por la prioridad cuando el procesador está libre. Además, las prioridades pueden ser **estáticas** o **dinámicas.**

En este tipo de planificaciones debe existir un componente (**planificador**) que se encargue de:

* Asignar el procesador a una tarea activa o ejecutable (dispatch)
* Desalojar la tarea de la CPU

Las tareas tienen tres estados (suspendida, ejecutable, ejecutándose).

El planificador se activa en dos eventos:

1. Una o más tareas se activan (pasan a estado ejecutable)
2. La tarea en ejecución termina (pasa a suspendido)

Al inicio, **todas las tareas pasan a activadas.**

### Planificación RMS

Es un método de planificación **estático online** en el que se asigna la mayor prioridad a las tareas más frecuentes (a las que tienen menor periodo). Es método es óptimo en el caso de que todas las tareas sean periódicas y que D=T.

### Test de planificabilidad

Permiten determinar si el conjunto de tareas es planificable según un algoritmo. Existen varios tipos de test:

* Test suficientes: en caso de éxito, el sistema es planificable. En caso contrario, no podemos afirmar nada.
* Test exactos (suficientes y necesarios): en caso de éxito, el sistema es planificable. En caso contrario, no es planificable (no se cumplirá algún deadline).

### Test de Liu & Layland

Es un test suficiente. Se puede aplicar a RMS y EDF. Se usan el **factor de utilización** y el **factor de utilización máximo.** Un sistema pasa el test si el factor de utilización es menor o igual que el factor de utilización máximo. En caso contrario, no se puede afirmar nada.

Logotipo, nombre de la empresa

Descripción generada automáticamente

#### Ejemplo

Texto

Descripción generada automáticamente con confianza media

El cronograma es el siguiente:

Gráfico, Gráfico de cajas y bigotes

Descripción generada automáticamente

### 11.2.2. Planificación EDF

Aquí las prioridades son dinámicas. Primero el más urgente. Se asigna una prioridad más alta a la tarea que se encuentre más próxima a su plazo de respuesta máxima. Es más óptimo ya que no se necesita que las tareas sean periódicas y menos pesimista que el RMS.

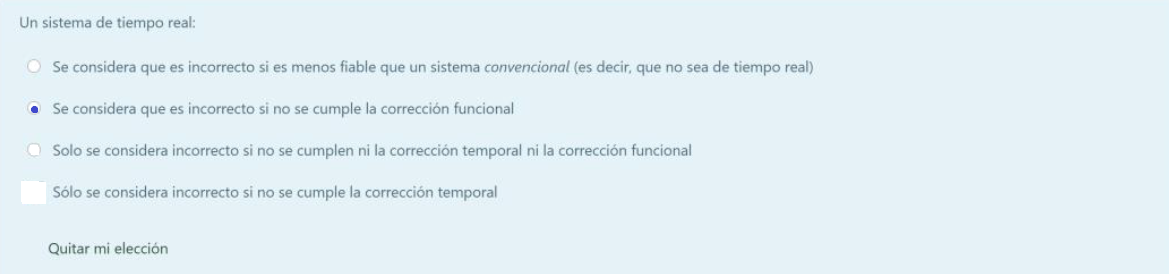
Para este algoritmo, el test de Liu & Layland se pasa si el factor de utilización es menor o igual a 1.

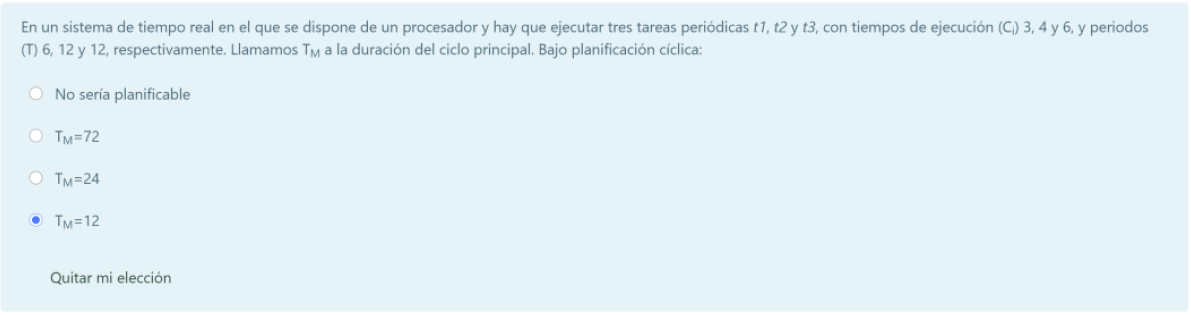
Un reloj en el medio

Descripción generada automáticamente con confianza media

# 

# Preguntas examen





Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Una captura de pantalla de un celular con texto

Descripción generada automáticamente con confianza media

Captura de pantalla de un celular con texto

Descripción generada automáticamente