

Introducción a los sistemas de tiempo real

Juan Antonio de la Puente
DIT/UPM

Objetivos

- ◆ Veremos los conceptos más importantes relacionados con los sistemas de tiempo real
- ◆ Analizaremos sus requisitos y características más importantes
- ◆ Examinaremos los tipos de sistemas de tiempo real más comunes
- ◆ Veremos algunos de los métodos y herramientas que se utilizan para diseñar, analizar y realizar sistemas de tiempo real

Sistema de tiempo real

Un **sistema de tiempo real** es un sistema informático que

- Interacciona repetidamente con su entorno físico
- Responde a los estímulos que recibe del mismo dentro de un plazo de tiempo determinado

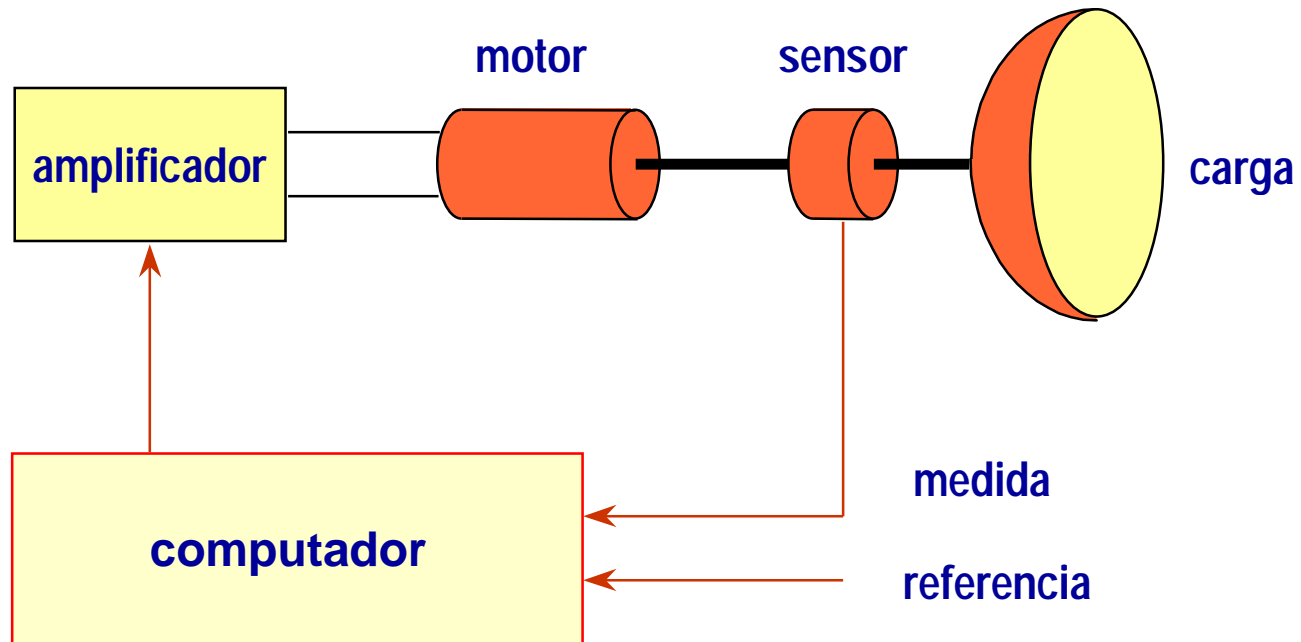
Para que el funcionamiento del sistema sea correcto no basta con que las acciones sean correctas, sino que tienen que ejecutarse dentro del intervalo de tiempo especificado

El tiempo en que se ejecutan las acciones del sistema es significativo

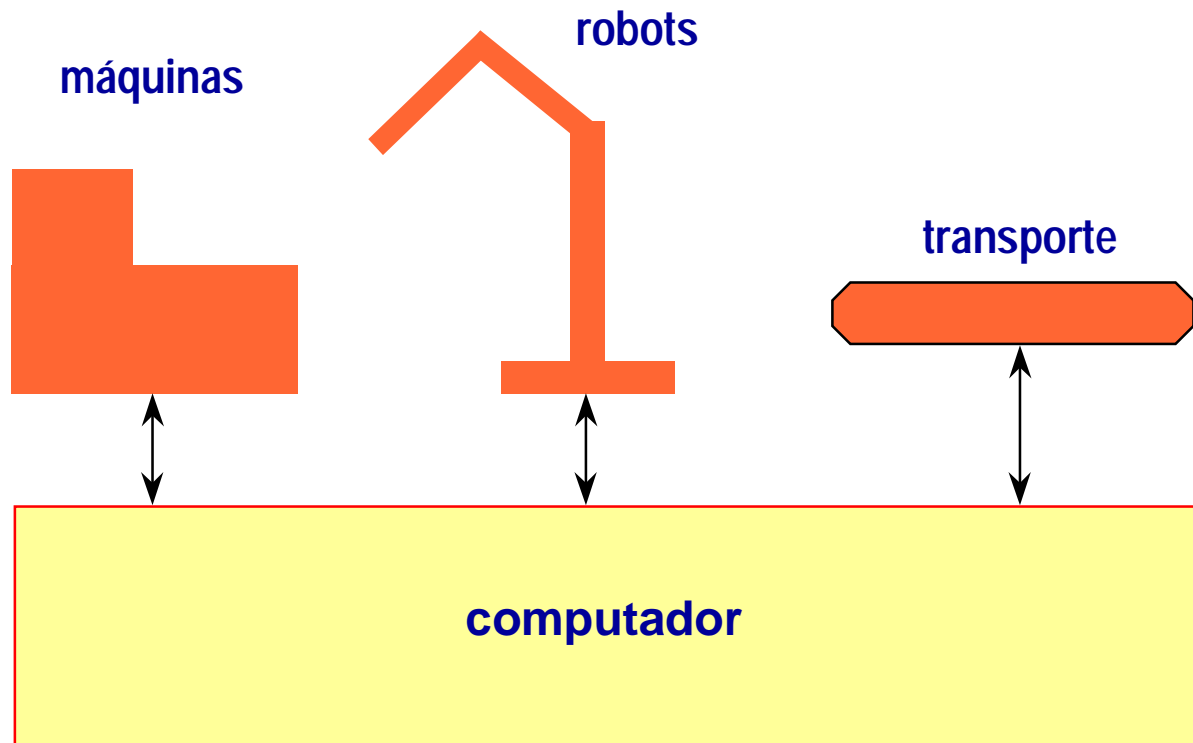
Ejemplos de aplicación

- ◆ Aviónica
- ◆ Control de tráfico aéreo
- ◆ Control de trenes
- ◆ Control de automóviles
- ◆ Telecomunicaciones
- ◆ Producción y distribución de energía eléctrica
- ◆ Electrónica de consumo
- ◆ Sistemas multimedia

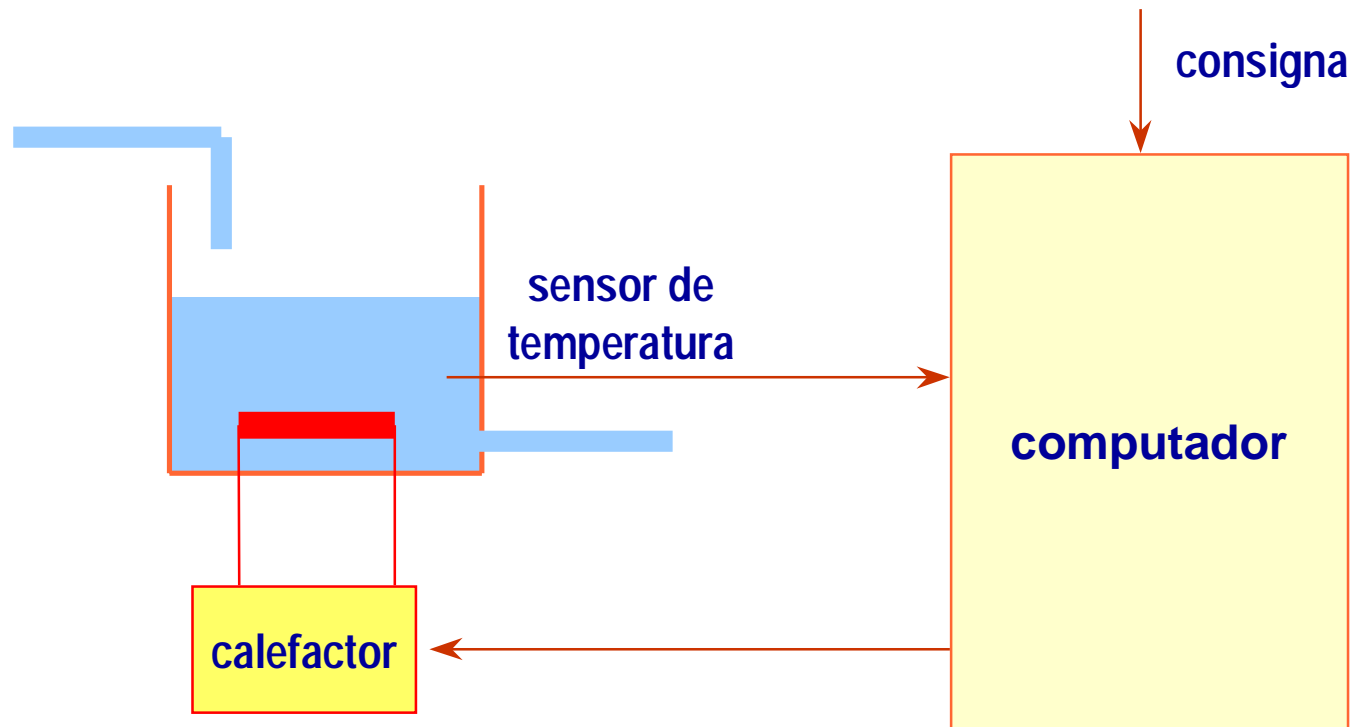
Ejemplo: control de posición



Ejemplo: control de fabricación



Ejemplo: control de procesos



Entorno de un sistema de tiempo real



Sistemas empotrados

- ◆ Muchos sistemas de tiempo real son componentes de otros sistemas, en los que realizan funciones de control
- ◆ En este caso se dice que se trata de **sistemas empotrados** (*embedded systems*)
- ◆ Ejemplos
 - automóviles
 - electrónica de consumo: teléfonos, radios, televisores
 - electrodomésticos
 - periféricos de computador
- ◆ Características
 - A menudo, el computador no es visible desde fuera
 - Los recursos son limitados

Requisitos funcionales (1)

◆ Adquisición de datos

– Medida de variables

- » cada variable pertenece la **esfera de control** de un subsistema
- » fuera de su esfera de control se puede observar, pero no cambiar
- » la **imagen** de una variable en el STR tiene un **intervalo de validez**
- » las medidas pueden estar dirigidas por **tiempo** o por **sucesos**

– Acondicionamiento de señales

- » los **datos brutos** se filtran y convierten a unidades de ingeniería
- » luego se analizan para obtener **datos validados**

– Supervisión

- » cuando algún dato tiene valores incorrectos se generan **alarmas**
- » es importante identificar el **suceso original** de una serie de alarmas encadenadas

Requisitos funcionales (2)

◆ Control digital directo

- El sistema de tiempo real puede **actuar** sobre el sistema controlado para conseguir que tenga un comportamiento determinado
- Normalmente se usa un esquema con **realimentación**
 - » la **accion de control** es función de la desviación entre los valores de **referencia** y los valores medidos de las variables
 - » el diseño de los **algoritmos de control** es un problema importante
- El comportamiento es muy regular (**muestreo periódico**)

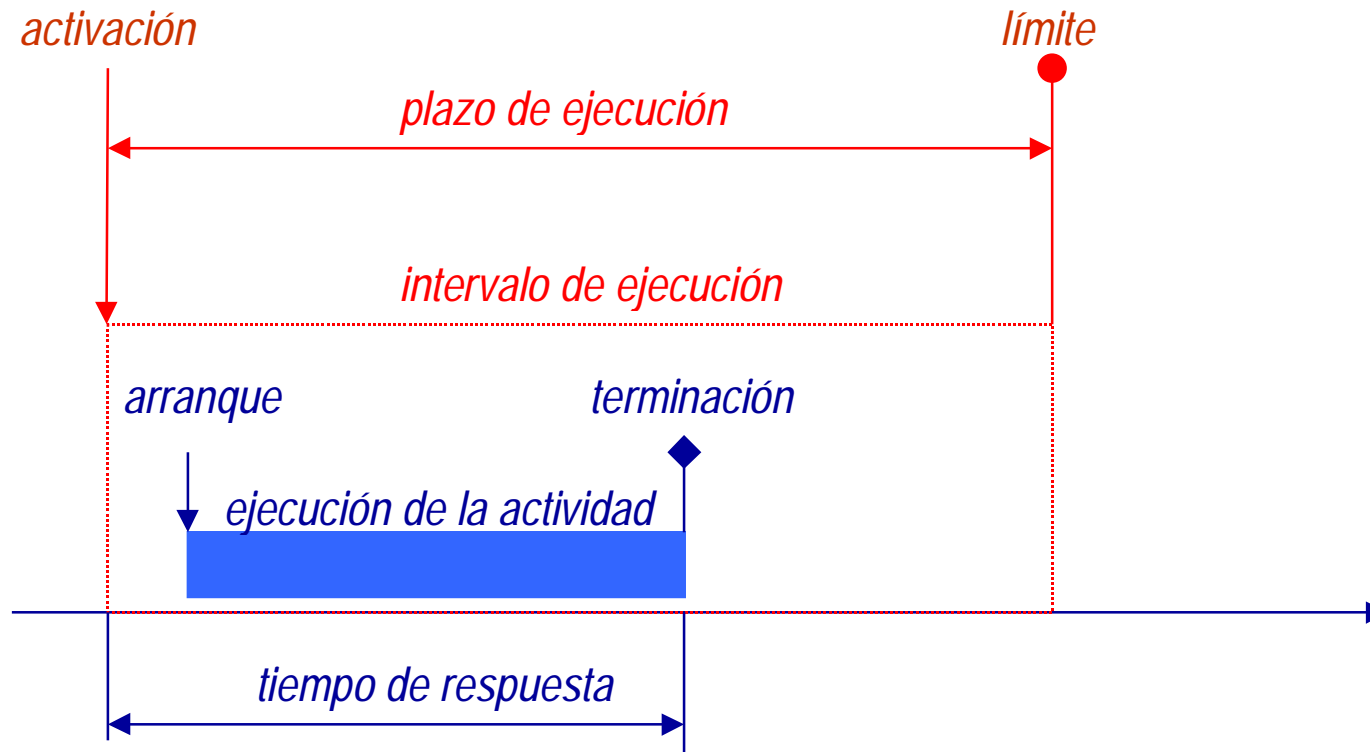
Requisitos funcionales (3)

◆ Interacción con personas

- La interfaz de operador es muy importante
 - » una mala interfaz puede causar accidentes
- Aspectos típicos:
 - » presentación de datos
 - » presentación de alarmas
 - » presentación de tendencias
 - » registro de datos
 - » generación de informes

Requisitos temporales

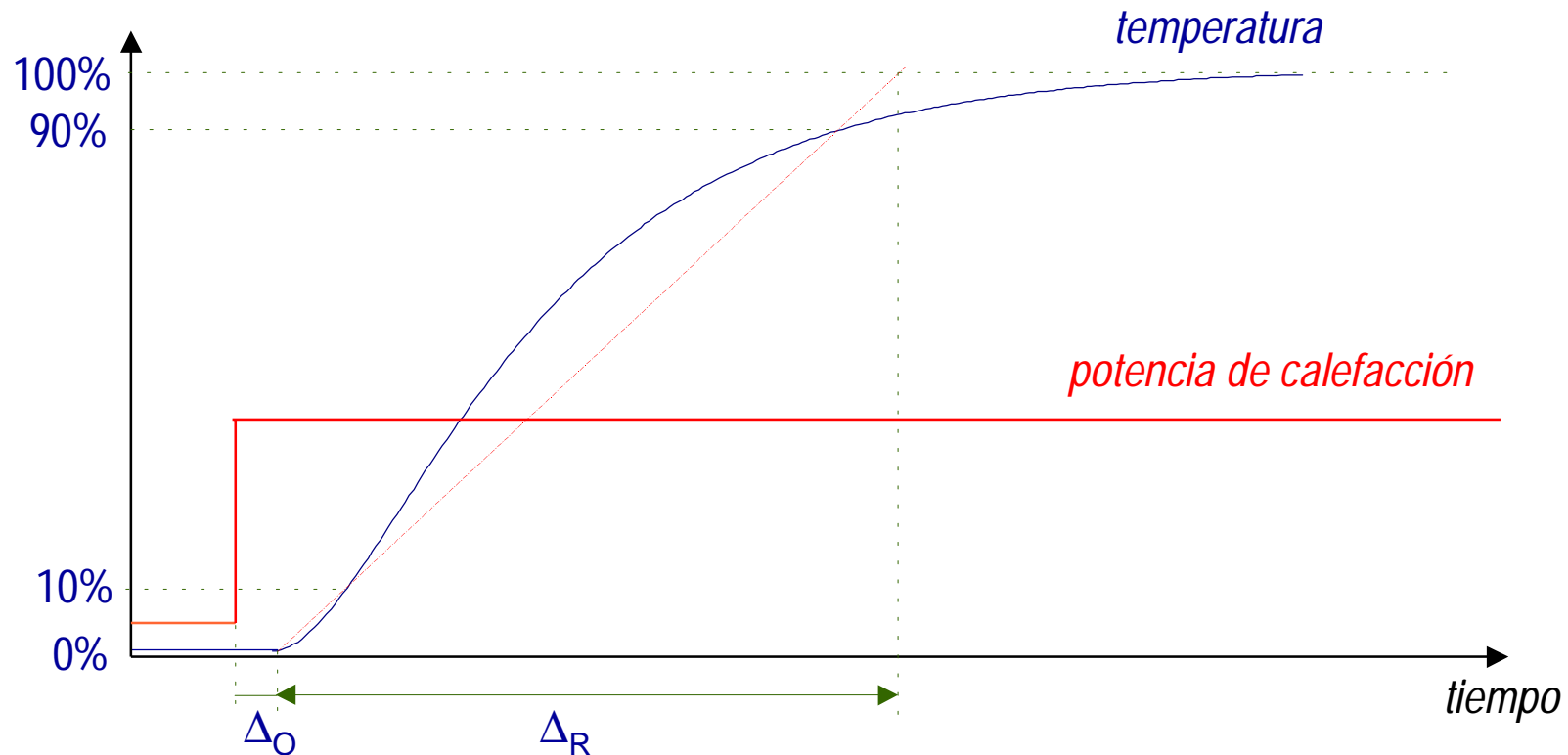
Las acciones de los sistemas de tiempo real se ejecutan repetidamente, dentro de intervalos de tiempo determinados



Origen de los requisitos temporales

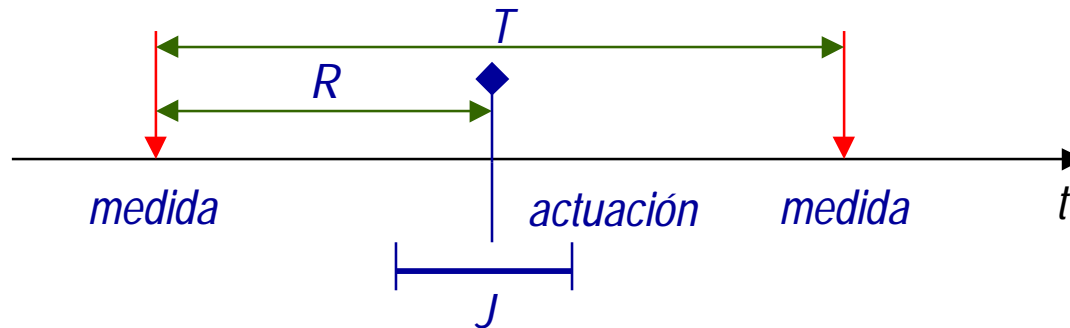
- ◆ Los más críticos aparecen en los bucles de control

Ejemplo:



Control por computador

- ◆ El computador muestrea periódicamente la *variable controlada*, la compara con el *valor de referencia*, y calcula el valor de la *variable de control* mediante un *algoritmo de control*
- ◆ El computador tarda un cierto *tiempo de respuesta* R en calcular la acción de control:



- ◆ J es la *variación (jitter)* en el tiempo de respuesta, $J = R_{MAX} - R_{MIN}$

Relación entre los parámetros temporales

Período	$T < D_R / 10$
Tiempo de respuesta	$R < D \leq T$
Variación (<i>jitter</i>)	$J \ll R$

- Un período de muestreo excesivo puede falsear la imagen de las variables del sistema controlado
- D es un *plazo de respuesta* menor o igual que el período
- Una variación significativa en R puede comprometer el correcto funcionamiento del algoritmo de control

Requisitos de fiabilidad y seguridad (1)

◆ Fiabilidad

- Probabilidad de proporcionar el servicio especificado
 - » Para una tasa de fallos constante de λ averías/hora se tiene
- Tiempo medio entre averías:
 - » Los sistemas con $MTTF > 10^9$ se denominan *sistemas ultrafiables*

◆ Seguridad

- Tipos de averías: *malignas* y *benignas*
 - » Una avería maligna tiene un coste muy superior a la utilidad del sistema
- Sistemas críticos
 - » Deben ser ultrafiables respecto a las averías malignas
 - » En muchos casos se exige una *certificación* de seguridad efectuada por un organismo independiente

Requisitos de fiabilidad y seguridad (2)

◆ Mantenibilidad

– Medida del tiempo necesario para reparar una avería benigna

» Para una tasa de reparación de μ reparaciones / hora :

– Tiempo medio de reparación: $MTTR = 1/\mu$

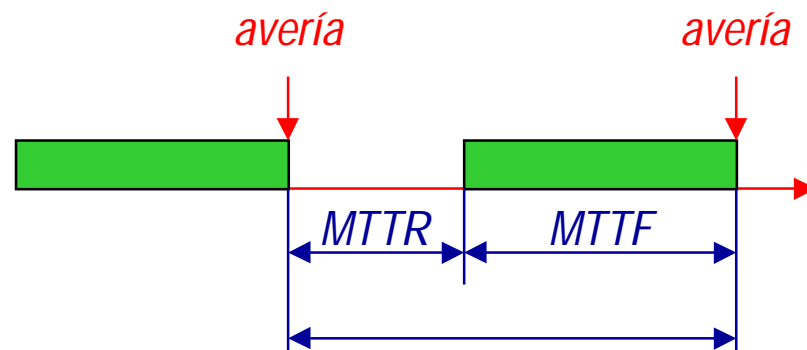
» Conflicto: los elementos fáciles de mantener son menos fiables

$$M(t) = e^{-\lambda(t-t_f)}$$

◆ Disponibilidad

– Fracción de tiempo durante el cual el sistema está disponible

$$A = \frac{MTTF}{MMTF + MTTR}$$
$$= \frac{MTTF}{MTBF}$$



Clases de sistemas de tiempo real

- ◆ Según las propiedades del sistema controlado
 - sistemas críticos y sistemas acrícos
 - sistemas con parada segura y sistemas con degradación aceptable
- ◆ Según las propiedades del sistema de tiempo real
 - sistemas con tiempo de respuesta garantizado y sistemas que hacen lo que pueden
 - sistemas con recursos adecuados y sistemas con recursos inadecuados
 - sistemas dirigidos por tiempo y sistemas dirigidos por eventos

Sistemas de tiempo real críticos y acrícos

Se distinguen por sus requisitos temporales y de fiabilidad

Sistemas críticos *(hard real-time systems)*

- ◆ Plazo de respuesta estricto
- ◆ Comportamiento temporal determinado por el entorno
- ◆ Comportamiento en sobrecargas predecible
- ◆ Requisitos de seguridad críticos
- ◆ Redundancia activa
- ◆ Volumen de datos reducido

Sistemas acrícos *(soft real-time systems)*

- ◆ Plazo de respuesta flexible
- ◆ Comportamiento temporal determinado por el computador
- ◆ Comportamiento en sobrecargas degradado
- ◆ Requisitos de seguridad acrícos
- ◆ Recuperación de fallos
- ◆ Gran volumen de datos

Sistemas con parada segura y sistemas con degradación aceptable

Se distinguen por su comportamiento en caso de avería

Sistemas con parada segura (*fail-safe*)

- ◆ Detención en estado seguro
- ◆ Probabilidad de detección de fallos elevada

Sistemas con degradación aceptable (*fail-soft*)

- ◆ Funcionamiento con pérdida parcial de funcionalidad o prestaciones
- ◆ También hay sistemas con *tolerancia de fallos completa (fail operational)*

Sistemas con respuesta garantizada y sistemas que hacen lo que pueden

Se distinguen por su grado de determinismo temporal

Sistemas con respuesta garantizada (*guaranteed response systems*)

- ◆ Comportamiento temporal garantizado analíticamente
- ◆ Hace falta caracterizar con precisión la carga máxima y los posibles fallos

Sistemas que hacen lo que pueden (*best-effort systems*)

- ◆ Comportamiento temporal de tipo “lo mejor que se pueda”
- ◆ No se hace una caracterización precisa de carga y fallos
- ◆ Sólo sirve para sistemas acríticos

Sistemas con recursos adecuados e inadecuados

Se distinguen por la cantidad de recursos disponibles

Sistemas con recursos adecuados (*resource-adequate systems*)

- ◆ Diseño con suficientes recursos para garantizar el comportamiento temporal con máxima carga y en caso de fallos

Sistemas con recursos inadecuados (*resource-inadequate systems*)

- ◆ Diseño con recursos “razonables” desde un punto de vista económico
- ◆ Sólo sirve para sistemas acríticos

Sistemas dirigidos por tiempo y por sucesos

Se distinguen por la forma de arrancar la ejecución de sus actividades

Sistemas dirigidos por sucesos *(event-triggered systems)*

- ◆ Arranque cuando se produce un suceso de cambio de estado
- ◆ Mecanismo básico: interrupciones

Sistemas dirigidos por tiempo *(time-triggered systems)*

- ◆ Arranque en instantes de tiempo predeterminados
- ◆ Mecanismo básico: reloj

Resumen de características de los sistemas de tiempo real críticos

- ◆ Gran tamaño y complejidad
- ◆ Simultaneidad de acciones (conurrencia)
- ◆ Seguridad y fiabilidad
- ◆ Determinismo temporal
- ◆ Dispositivos de entrada y salida especiales

Desarrollo de software para sistemas de tiempo real críticos

- ◆ Las características específicas de los sistemas de tiempo real críticos condicionan los métodos y herramientas que se utilizan para desarrollar el software
- ◆ No todas las técnicas que se usan para construir otros tipos de sistemas sirven para el software de tiempo real crítico
 - suele haber problemas de fiabilidad

Lenguajes de programación

- ◆ Un lenguaje de programación de sistemas de tiempo real debe facilitar la realización de sistemas
 - concurrentes,
 - fiables,
 - con un comportamiento temporal analizable.
- ◆ Hay tres clases de lenguajes de interés para STR:
 - **Lenguajes ensambladores**
 - » Flexibles y eficientes, pero costosos y poco fiables
 - **Lenguajes secuenciales** (Fortran, Pascal, C, ...)
 - » Necesitan un SO para concurrencia y tiempo real
 - **Lenguajes concurrentes** (Modula, Ada, ...)
 - » Concurrencia y tiempo real incluidos en el lenguaje

C

- ◆ Es un lenguaje muy utilizado para programación de sistemas
- ◆ Es un lenguaje
 - **estructurado**, con **bloques**
 - **sin tipado fuerte**
 - muy **flexible** (pero a veces **poco seguro**)
- ◆ No tiene integrada la concurrencia ni el tiempo real
 - se consigue invocando **servicios del sistema operativo** de forma explícita
- ◆ No facilita la descomposición en módulos ni la programación con objetos
 - se puede hacer con **C++**
(una extensión de C para programar con objetos)

Ada

- ◆ Es un lenguaje diseñado específicamente para sistemas de tiempo real empotrados
 - concurrencia
 - tiempo real
 - acceso al hardware e interrupciones
- ◆ Es un lenguaje imperativo, descendiente de Pascal
 - estructura en bloques
 - fuertemente tipado
- ◆ Está pensado para construir sistemas grandes y cambiantes
 - paquetes (módulos) y esquemas genéricos
 - extensión de tipos con herencia
 - biblioteca jerárquica
 - interfaces normalizadas con otros lenguajes (C, Fortran)

Ada 95

- ◆ Es la versión actual normalizada de Ada
 - ◆ La norma define
 - un **núcleo** común para todas las implementaciones
 - unos **anexos** especializados para
 - » programación de sistemas
 - » sistemas de tiempo real
 - » sistemas distribuidos
 - » sistemas de información
 - » cálculo numérico
 - » **fiabilidad y seguridad**
 - Los anexos definen
 - » **paquetes** de biblioteca
 - » **mecanismos** de implementación
- no añaden sintaxis ni vocabulario al lenguaje

Sistemas operativos

- ◆ Los sistemas operativos convencionales no son adecuados para realizar sistemas de tiempo real
 - no tienen un comportamiento determinista
 - no permiten garantizar los tiempos de respuesta
- ◆ Un sistema operativo de tiempo real debe soportar
 - **conurrencia**: procesos ligeros (*threads*) con memoria común
 - **temporización**: medida de tiempos y ejecución periódica
 - **planificación**: prioridades fijas con desalojo, acceso a recursos con protocolos de herencia de prioridad
 - **dispositivos de E/S**: acceso a recursos de hardware e interrupciones

POSIX

- ◆ Es un conjunto de normas IEEE/ISO que definen **interfaces de sistemas operativos**
- ◆ Permiten desarrollar software **portátil** y **reutilizable** (**P**ortable **O**perating **S**ystem **I**nterface) + **X**
- ◆ Las normas definen **servicios** que se pueden incluir o no en un sistema operativo particular
- ◆ Además se definen **perfiles de aplicación** con conjuntos de servicios estándar
- ◆ Hay interfaces para C, Ada, y otros lenguajes

Normas POSIX

POSIX 1, 1a	Interfaz básica similar a UNIX™
POSIX 1b,1d,1i,1j	Extensiones de tiempo real
POSIX 1c	Procesos ligeros (<i>threads</i>)
POSIX 1e	Seguridad
POSIX 1f	NFS
POSIX 1g	Servicios de red (<i>sockets</i> , XTI)
POSIX 1h	Tolerancia de fallos
POSIX 21	Comunicaciones de tiempo real
POSIX 5,5a,5b	Interfaces para Ada

Perfiles de aplicación

- ◆ Definen subconjuntos de servicios para distintos tipos de aplicaciones

POSIX 13 : Perfiles para sistemas de tiempo real

- PSE50 : sistema de tiempo real mínimo
 - » sin gestión de memoria, ficheros ni terminal
 - » sólo *threads* (no procesos pesados)
- PSE51 : controlador de tiempo real
 - » tiene sistema de ficheros y terminal
- PSE52 : sistema de tiempo real dedicado
 - » tiene gestión de memoria y procesos pesados
- PSE53 : sistema de tiempo real generalizado
 - » sistema completo con todo tipo de servicios

Resumen (1)

- ◆ Los sistemas de tiempo real interactúan con su entorno y ejecutan sus acciones dentro de intervalos de tiempo determinados
- ◆ Tienen requisitos funcionales, temporales y de fiabilidad muy exigentes
- ◆ Hay varias clases de sistemas de tiempo real:
 - críticos y acrícos
 - con parada segura, degradación aceptable y tolerancia de fallos completa
 - con respuesta garantizada y que hacen lo que pueden
 - con recursos adecuados e inadecuados
 - dirigidos por eventos y por tiempo

Resumen (2)

- ◆ Las características principales de los sistemas de tiempo real son:
 - gran tamaño y complejidad
 - concurrencia
 - dispositivos de entrada y salida especiales
 - seguridad y fiabilidad
 - determinismo temporal
- ◆ Para desarrollar software para sistemas de tiempo real críticos es necesario utilizar métodos y herramientas especiales
 - lenguajes de programación: C, Ada
 - sistemas operativos: normas POSIX para tiempo real