Sistemas Distribuidos I
(75.74)

Intro a Sistemas de Tiempo Real

Sistemas RT. Sistemas de Control.

Docentes

- Pablo D. Roca
- Ezequiel Torres Feyuk
- Guido Albarello
- Ana Czarnitzki
- Cristian Raña

Agenda

- **Sistemas de Tiempo Real (RT)**
- Sistemas de Control

Sistemas de Tiempo Real (RT) | Definición

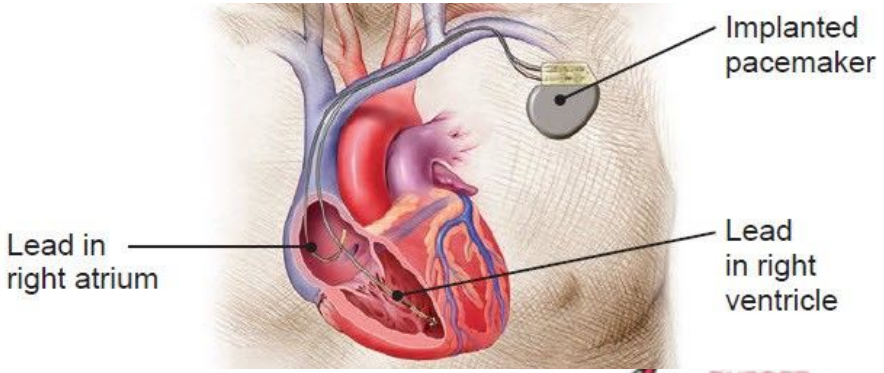
- Son aquellos sistemas cuya evolución se especifica en términos de requerimientos temporales requeridos por el entorno
- La correctitud del sistema depende de que entregue respuestas correctas y en tiempo correcto.
- Un sistema es RT si tiene al menos un servicio RT
- Ejemplos de sistemas RT:
  - Electrodomésticos digitales, medidores de señales (presión, pulsaciones, ritmo cardíaco, etc.) mediciones por sensores, control de automóviles, control en aeronaves, marcapasos, etc.

Hard RT:

- se debe evitar todo fallo relacionado con el tiempo de *delivery*
- perder un *deadline* o plazo de respuesta es un fallo total

Soft RT:

- fallos relacionados con el tiempo de *delivery* pueden ser admitidos ocasionalmente.
- la utilidad de un resultado disminuye tras el *deadline*



RT | Previsibilidad

- RT implica previsibilidad, no performance.
- Puede tratarse de sistemas con tiempos característicos lentos pero aún así ser RT
- Se trata de hacer un correcto *scheduling* para que se cumplan los deadlines previstos por diseño

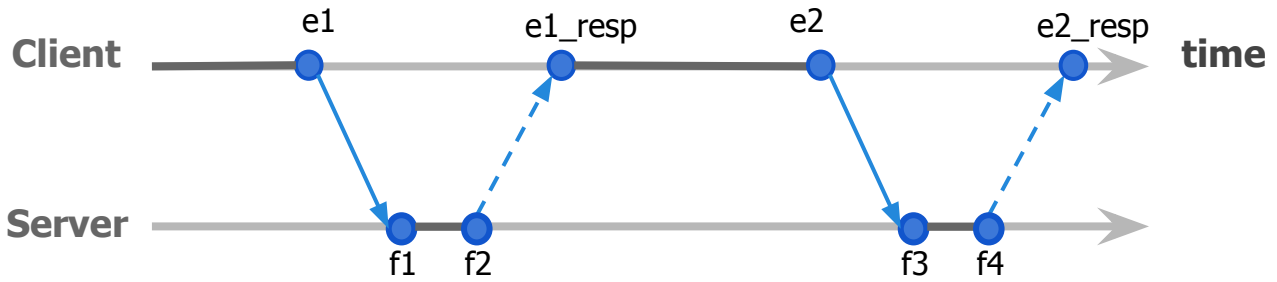
RT | Comunicación en sistemas RT

- RT require comunicación fiable y sincrónica (con *deadlines* bien definidos)
- TCP/IP no permite asegurar estos atributos.
- Comunicación Serial permite control sobre estos aspectos: Profibus
- Ethernet por otro lado, puede ser utilizado con el protocolo adecuado RT para capas de transporte y superiores:
  - Se requiere evitar no-determinismo en protocolo
  - En Ethernet esto sólo puede ocurrir en caso de colisiones en transmisión (asumiendo switching determinístico)
  - Existen protocolos basados en Ethernet que lo admiten: Profinet

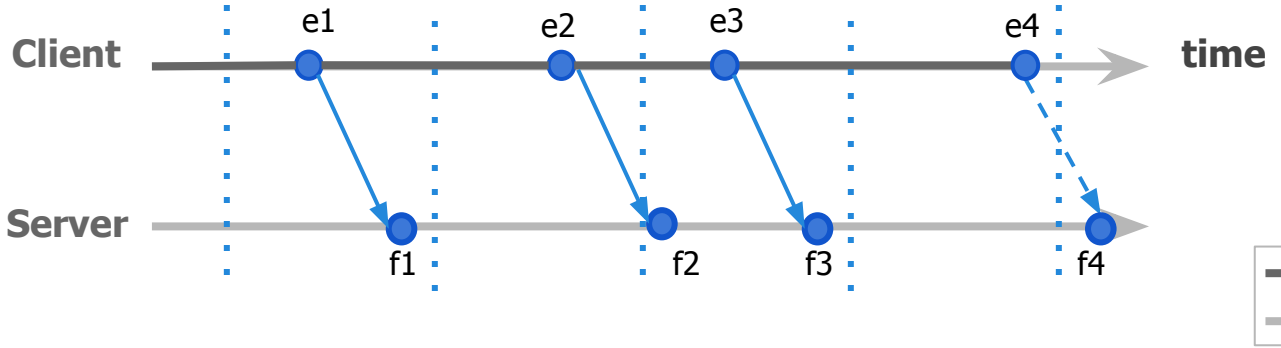
RT | *Fault Tolerance* RT

- Los sistemas ahora deben ser tolerantes a fallos de tiempo
- Distintos tipos de estrategias:
  - **Soft RT:**
    - Ej.: Sistemas web de gran escala
    - El 90% de los requests deben reponderse en 2 seg. El 10% restante se deben responder en 10 seg.
  - **Hard RT:**
    - Ej.: Misión crítica
    - El 100% de los requests debe resolverse en 1 seg. Frente a errores, se asume un fallo catastrófico y se recomienda *hard reset*.
    - Muy importante revisar el factor de *Maintainability*.

Event - Triggered



Time - Triggered



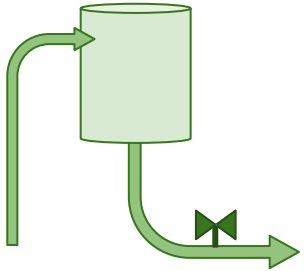
Agenda

- Sistemas de Tiempo Real (RT)
- **Sistemas de Control**

- Distintos escenarios cotidianos plantean un sistema a controlar de forma manual o automática

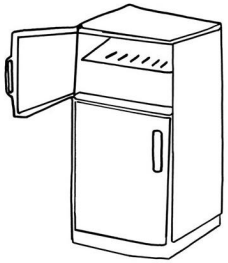
En la Industria

- Esquemas de irrigación
- Procesos de transferencia térmica
- Procesos químicos
- Líneas de producción



En la Vida Cotidiana

- Termostatos, ascensores, expendedores de líquidos, control de luminosidad, electrodomésticos en general, etc



Sistemas de Control | Nociones

Control
capacidad de actuar para garantizar el comportamiento de un suceso.

Proceso
toda sucesión de operaciones que se desea controlar.

Variable controlada
cantidad o condición que se mide o controla. Salida del sistema.

Variable manipulada
cantidad o condición que se modifica para afectar el valor de la variable controlada.

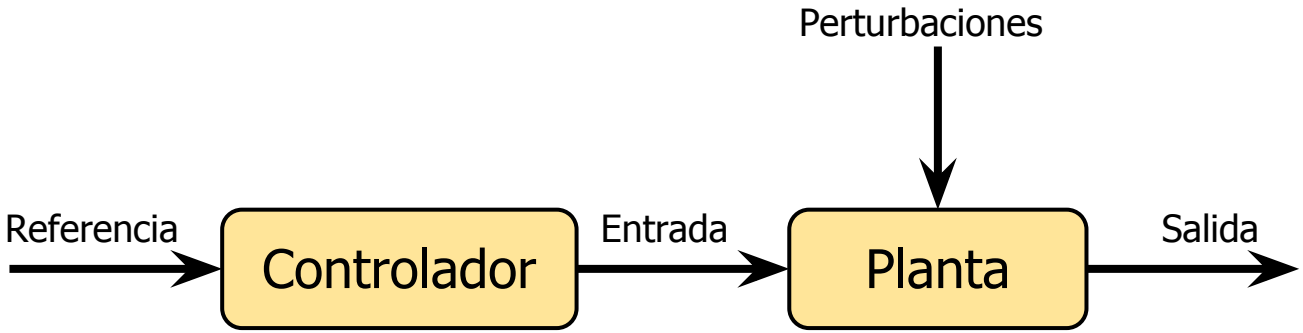
Perturbación
señal que tiende a afectar negativamente el valor de la salida del sistema.

Planta
cualquier sistema físico que se desea controlar.

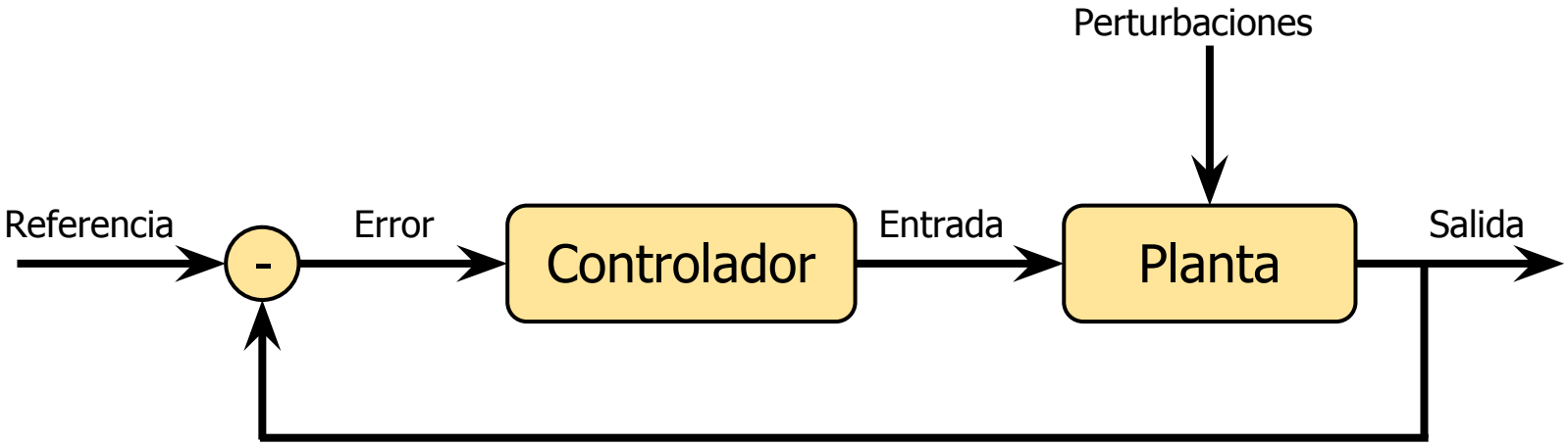
Controlador (referencia)
sistema encargado de determinar la actuación para conseguir cierto objetivo del proceso

Actuador
elemento físico de la planta que frente a señales del controlador opera sobre el proceso

- **Control a lazo abierto:**
  - la salida del sistema no afecta la acción de control.



- **Control a lazo cerrado** o realimentado (feedback):
  - se utiliza información sobre el estado del sistema para actuar sobre el sistema y llevar la salida del mismo a los valores deseados.

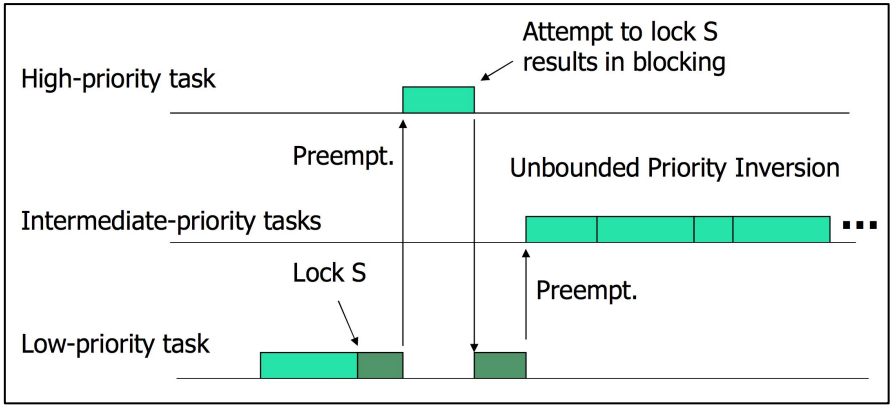


Programación y Tiempo Real

- Arquitecturas dirigidas por eventos (event-triggered) o por el tiempo (time-triggered)
- *Scheduling* es importante:
  - Apropiativo (non-*preemptive*) y de acuerdo a un esquema de prioridades para poder cumplir deadlines
- Protocolos de comunicación específicos:
  - Algoritmos de backoff => imprevisibilidad

Caso de Estudio: Mars Pathfinder

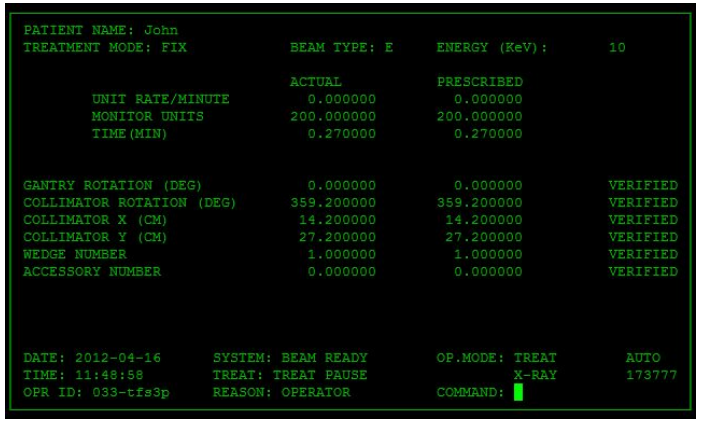
- Mars Pathfinder fue la primer misión a marte usando rovers. A los pocos días de aterrizar, se detectan continuos reinicios al intentar enviar datos.
- Motivos detectados:
  - **Watchdog** que reiniciaba el sistema al perderse el deadline de una tarea crítica.
  - **Inversión de prioridades**: una tarea de alta prioridad es interrumpida por una tarea de prioridad media.



Fuente: http://blog.shiftehfar.org/?p=207

Caso de Estudio: Therac-25

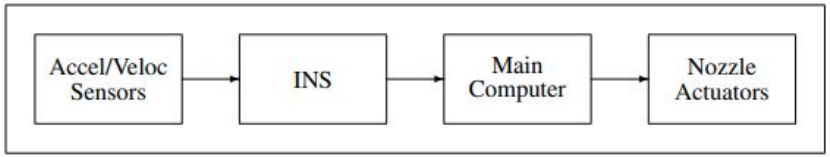
- Therac-25 consiste de un acelerador de electrones y un sistema de control real-time para tratamientos de pacientes por radiación.
- Durante 2 años de empleo del equipo se detectaron al menos 6 fallos con pacientes recibiendo hasta 100x de la radiación recetada.
- Fallas presentes desde versiones anteriores del equipo (Therac-20) pero se manifestaron al eliminar controles por hardware y confiar en el software.
  - **Race Conditions:** UI mostrando modo erróneo al operador durante el bootstrapping del magnetrón
  - **Overflow:** 1 byte para contador de errores detectados. Overflow a 0 al detectar 256 chequeos fallidos...



References:
https://www.buqsnag.com/blog/bug-day-race-condition-therac-25
Leveson, Turner, An investigation of the Therac-25 accidents, IEEE Computer, 1997

Caso de Estudio: Ariane 5

- Ariane 5 fue un desarrollo de la Agencia Espacial Europea para colocar satélites en órbita.
- En su despegue de Junio/1996 el cohete modifica su trayectoria de forma brusca y estalla en el aire.
- Se hereda el sistema de navegación del Ariane 4, casi sin modificaciones (y casi sin testing)
- En un análisis post-mortem se detectan distintos fallos:
  - **Cast Overflow:** El algoritmo de navegación asume posibilidad de casteo de valores 64-bits a variables 16-bits sin chequear overflow.
  - **Error in Redundant Systems:** Los sistemas redundantes reprodujeron exactamente las mismas condiciones.



References: http://www2.ing.unipi.it/~a009435/issw/extra/ariane5-benari.pdf

Bibliografía

- P. Verissimo, L. Rodriguez: Distributed Systems for Systems Architects, Kluwer Academic Publishers, 2001.
  - Cap. 11 - Real Time System Foundations
  - Cap. 12 - Paradigms of Real Time