dit
UPM

Entorno de ejecución

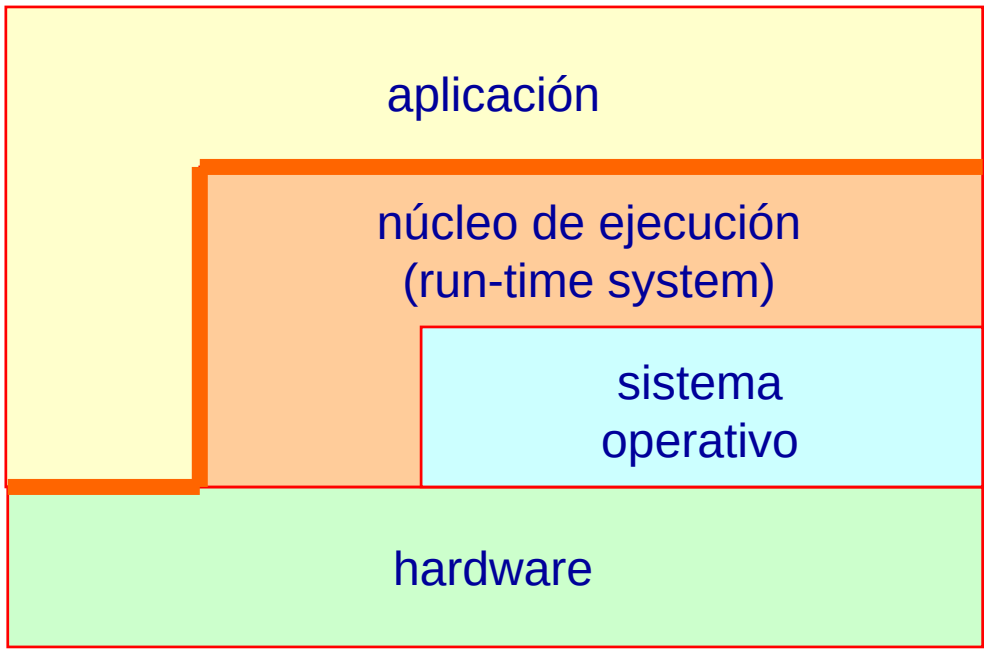
Juan Antonio de la Puente
DIT/UPM

Índice

- **Introducción**
- Perfiles de ejecución
- Modelos de planificación
  - cambio de contexto
  - operaciones no desalojables
  - tareas esporádicas
  - manejador de reloj

©1997-2000 Juan Antonio de la Puente

Entorno de ejecución



- Comprende el hardware y el software sobre el que se desarrolla la aplicación
  - Incluye software gráfico, comunicaciones, etc..

©1997-2000 Juan Antonio de la Puente

Proceso de diseño

- Podemos ver el diseño como un proceso de refinamiento de modelos con niveles de abstracción decrecientes
- En el proceso de refinamiento hay que considerar
  - **Compromisos,** propiedades que ya no se cambiarán en los niveles inferiores
  - **Obligaciones,** que cubren aspectos del diseño que se dejan para los niveles inferiores
- En el diseño se van transformando las obligaciones en compromisos
  - Este proceso está sujeto a **restricciones** impuestas por el entorno de ejecución
  - Los requisitos del sistema o los compromisos de diseño pueden imponer **restricciones** sobre el entorno de ejecución

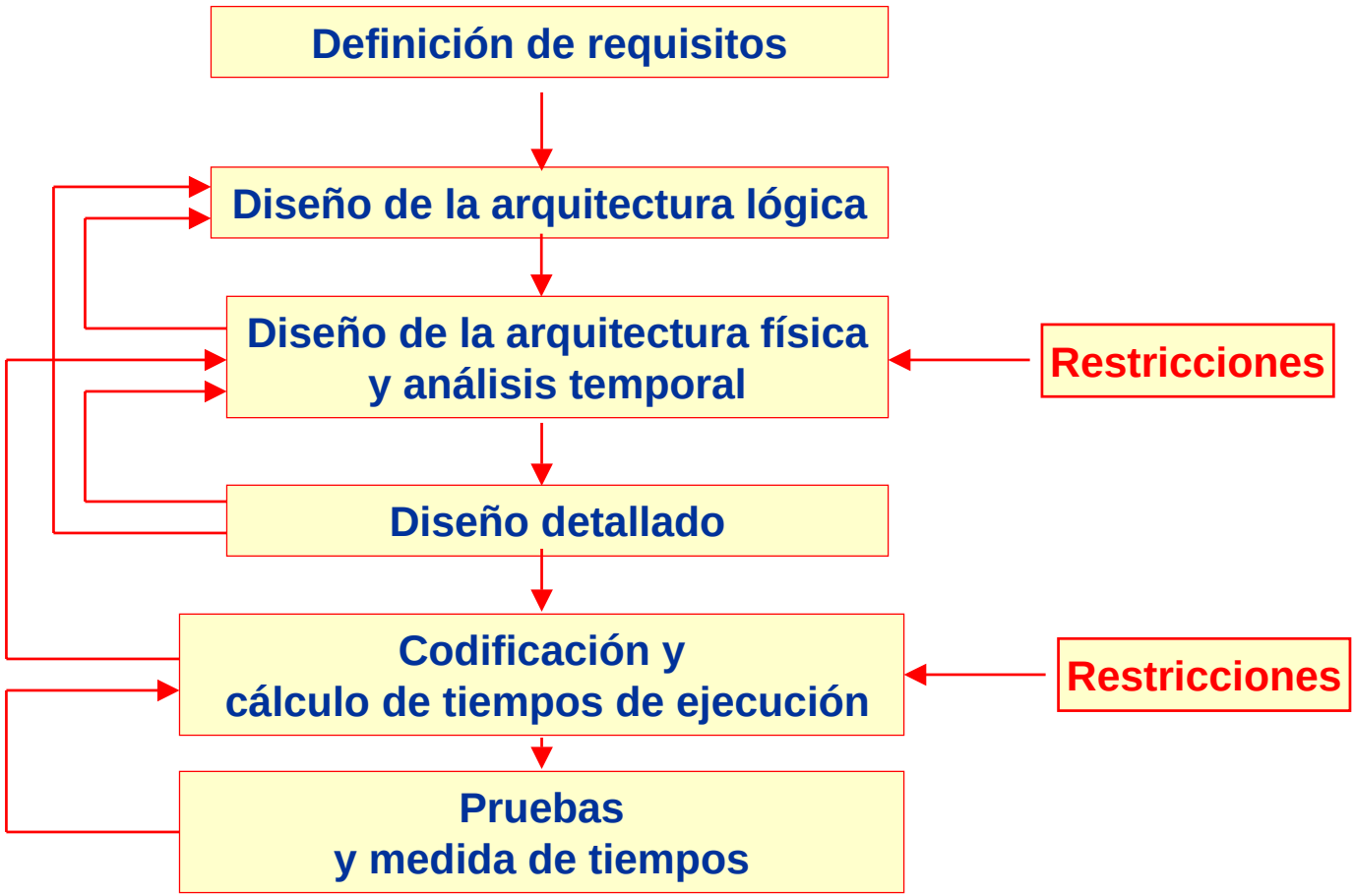
©1997-2000 Juan Antonio de la Puente

Niveles de descripción

- Dos niveles de diseño
  - **Modelo lógico** - compromisos independientes del entorno de ejecución
  - **Modelo físico** - compromisos dependientes del entorno de ejecución
- La arquitectura física debe tener en cuenta las propiedades del entorno de ejecución
  - Puede ser necesario modificar el diseño para adaptarlo al entorno de ejecución
  - Debe permitir el análisis de las propiedades no funcionales
    - » comportamiento temporal
    - » fiabilidad

©1997-2000 Juan Antonio de la Puente

Proceso de desarrollo



©1997-2000 Juan Antonio de la Puente

Requisitos del entorno de ejecución

- Determinismo
  - Compatible con las condiciones necesarias para poder analizar el comportamiento temporal
- Eficiencia
  - Se debe poder garantizar el cumplimiento de los requisitos temporales
  - Puede haber restricciones de memoria u otras que afecten a la velocidad
- Simplicidad
  - No se deben incluir funciones innecesarias

©1997-2000 Juan Antonio de la Puente

Índice

- Introducción
- **Perfiles de ejecución**
  - **Ada 95 (perfil de Ravenscar)**
  - POSIX
- Modelos de planificación
  - cambio de contexto
  - operaciones no desalojables
  - tareas esporádicas
  - manejador de reloj

©1997-2000 Juan Antonio de la Puente

Ajuste del entorno de ejecución

- Se trata de eliminar las funciones que no sean necesarias para
  - evitar un uso excesivo de recursos
  - simplificar el software y facilitar la certificación de seguridad
  - eliminar el software que no cumple ninguna función útil
- Algunos lenguajes y sistemas operativos permiten definir configuraciones restringidas
  - pragma *Restrictions* en Ada
  - perfiles en POSIX

©1997-2000 Juan Antonio de la Puente

Restricciones en Ada

- Se pueden especificar restricciones para hacer que el núcleo de ejecución sea simple y eficiente

```
pragma Restrictions (restriction{,restriction});

restriction ::= restriction_identifier
  | restriction_parameter_identifier => expression
```

- Las restricciones se comprueban al compilar el sistema

  Ejemplo:

```
pragma Restrictions (Max_Task_Entries => 0);
```

  (no se permiten citas entre tareas)

©1997-2000 Juan Antonio de la Puente

Restricciones para sistemas de tiempo real

- `No_Task_Hierarchy`
- `No_Abort_Statements`
- `No_Terminate_Alternatives`
- `No_Task_Allocators`
- `No_Dynamic_Priorities`
- `No_Asynchronous_Control`
- `Max_Select_Alternatives => ...`
- `Max_Task_Entries => ...`
- `Max_Protected_Entries => ...`
- `Max_Tasks => ...`

etc.

©1997-2000 Juan Antonio de la Puente

Otros requisitos para sistemas de tiempo real

- Requisitos de implementación
  - intervalos de prioridad
  - valores de tiempo
- Documentación
  - propiedades del reloj de tiempo real
  - valor mínimo de un retardo que causa una suspensión de la tarea
- Métricas
  - duración máxima de un tic de reloj y amplitud máxima de un salto de reloj
  - cota superior de la deriva del reloj
  - tiempo de ejecución de *Clock*, *delay* y *delay until*, y cota superior del retraso de *delay* y *delay until*
  - tiempo de ejecución de Set__Priority sin desalojo
  - tiempo de ejecución de *abort*
  - tiempo de ejecución de una selección asíncrona

  etc.

©1997-2000 Juan Antonio de la Puente

Restricciones para seguridad

El anexo H del estándar de Ada incluye todas las restricciones anteriores (con `Max_Tasks => 0`), y además otras como

- `No_Protected_Types`
- `No_Allocators`
- `No_Exceptions`
- `No_Floating_Point`
- `No_Fixed_Point`
- `No_IO`
- `No_Delay`
- `No_Recursion`
- `No_Reeentrancy`

etc.

©1997-2000 Juan Antonio de la Puente

Sistemas críticos
(HIS, High Integrity Systems)

- Son sistemas con requisitos de seguridad muy exigentes
  - deben someterse a procesos de análisis estático y dinámico especificados por organismos de certificación

- En Ada se sigue el documento ISO/IEC TR 15942:2000.

  Guide for the use of the Ada programming language in High Integrity Systems

- Se suele usar un *subconjunto seguro* del lenguaje
  - tradicionalmente sin tareas (p.ej. SPARK)
  - subconjunto seguro con tareas: perfil de Ravenscar

©1997-2000 Juan Antonio de la Puente

El perfil de Ravenscar

- Subconjunto de la parte concurrente de Ada para aplicaciones críticas

- Estrategia:
  - elminar elementos con tiempo de ejecución excesivo o imprevisible
  - permitir el análisis temporal del sistema
  - facilitar la implementación de la concurrencia mediante un núcleo de tiempo real pequeño, eficiente y fiable

©1997-2000 Juan Antonio de la Puente

Modelo de tareas de Ravenscar

- Tareas y objetos protegidos estáticos
  - no hay creación dinámica ni declaraciones anidadas
  - las tareas no terminan
- Objetos protegidos con una entrada, como máximo, con
  - barrera simple (varable booleana declarada en el mismo objeto
  - una tarea como máximo esperando que se abra la barrera
- Control de tareas síncrono (objetos de suspensión)
- Paquete Ada.Real_Time y retardo absoluto (delay until)
- Protocolos FIFO within priorities y ceiling locking
- Manejadores de interrupciones con procedimientos protegidos

La adecuación al modelo se puede comprobar al compilar mediante restricciones (excepto terminación y colas)

©1997-2000 Juan Antonio de la Puente

Elementos prohibidos en el perfil de Ravenscar

- Task hierarchies
- Protected object hierarchies
- Dynamic POs and tasks
- Task entries
- Protected types with more than one entry
- Complex barriers
- More than one task in one entry queue
- Requeue
- ATC
- Select statement
- Abort
- Dynamic priorities
- Calendar package
- Relative delays
- Asynchronous task control
- User-defined task attributes

©1997-2000 Juan Antonio de la Puente

Elementos permitidos en el perfil de Ravenscar

- Library level tasks and POs
- Task discriminants
- Task identifiers
- FIFO within priority and Ceiling Locking policies
- Real-Time package
- Delay until statements
- Protected procedures as interrupt handlers
- Synchronous task control
- Atomic and Volatile pragmas
- Count Attribute
  (but not within entry barriers)

©1997-2000 Juan Antonio de la Puente

Restricciones del perfil de Ravenscar

Estándar en Ada 95

No_Task_Hierarchy
No_Abort_Statements
No_Task_Allocators
No_Dynamic_Priorities
No_Asynchronous_Control
Max_Task_Entries => 0
Max_Protected_Entries => 1
Max_Asynchronous_Select_Nesting => 0
Max_Tasks => N

Nuevas

Simple_Barrier_Variables
Max_Entry_Queue_Depth => 1
No_Calendar
No_Relative_Delay
No_Protected_Type_Allocators
No_Local_Protected_Objects
No_Requeue
No_Select_Statements
No_Task_Attributes
No_Task_Termination

©1997-2000 Juan Antonio de la Puente

Pragma Ravenscar

- Resumen de todas las restricciones
- Ejemplo: GNAT

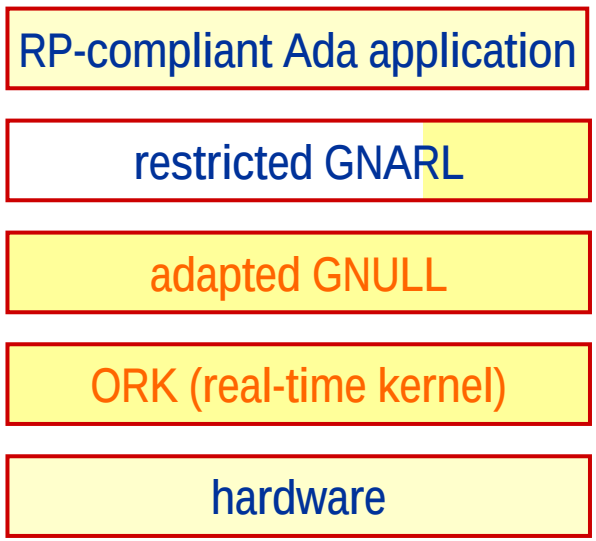
```
-- file gnat.adc

pragma Ravenscar;
pragma Restrictions (Max Tasks => N);
pragma Task Dispatching Policy (FIFO Within Priorities);
pragma Locking Policy (Ceiling Locking);

-- otras restricciones de seguridad
pragma Restrictions (No_Allocators,
                     No_IO,
                     -- etc.
                     );
```

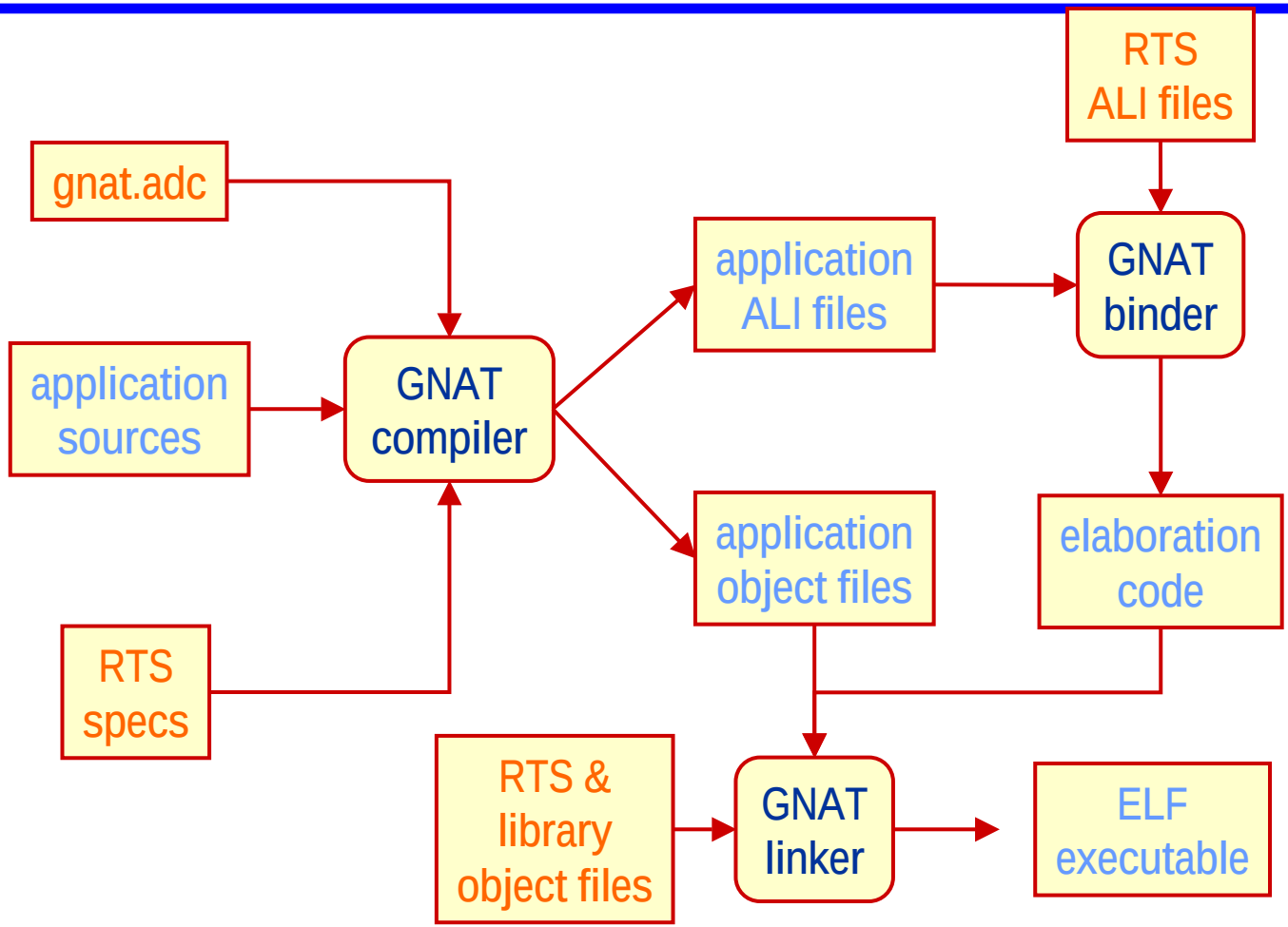
©1997-2000 Juan Antonio de la Puente

Ejemplo: GNAT/ORK (Open Ravenscar Kernel)



©1997-2000 Juan Antonio de la Puente

Proceso de compilacion con GNAT/ORK



©1997-2000 Juan Antonio de la Puente

Índice

- Introducción
- **Perfiles de ejecución**
  - Ada 95 (perfil de Ravenscar)
  - **POSIX**
- Modelos de planificación
  - cambio de contexto
  - operaciones no desalojables
  - tareas esporádicas
  - manejador de reloj

©1997-2000 Juan Antonio de la Puente

Perfiles de aplicación en POSIX

- 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

©1997-2000 Juan Antonio de la Puente

Índice

- Introducción
- Perfiles de ejecución
- **Modelos de planificación**
  - cambio de contexto
  - operaciones no desalojables
  - tareas esporádicas
  - manejador de reloj

©1997-2000 Juan Antonio de la Puente

Entorno de ejecución y comportamiento temporal

- Las características del núcleo de multiprogramación influyen en el tiempo de respuesta
- Los factores que hay que tener en cuenta son
  - cambio de contexto
    - » su duración puede variar
  - operaciones del núcleo que no se pueden desalojar
  - manejo de interrupciones
  - interrupciones del reloj
    - » mantenimiento del tiempo
    - » activación de procesos periódicos

©1997-2000 Juan Antonio de la Puente

Operaciones del núcleo no desalojables

- Equivalen a un recurso compartido por todas las tareas
- Se modelan como un bloqueo:

$$B_i = \max_{j \in lp(i)} (S_{ij}, S_K)$$

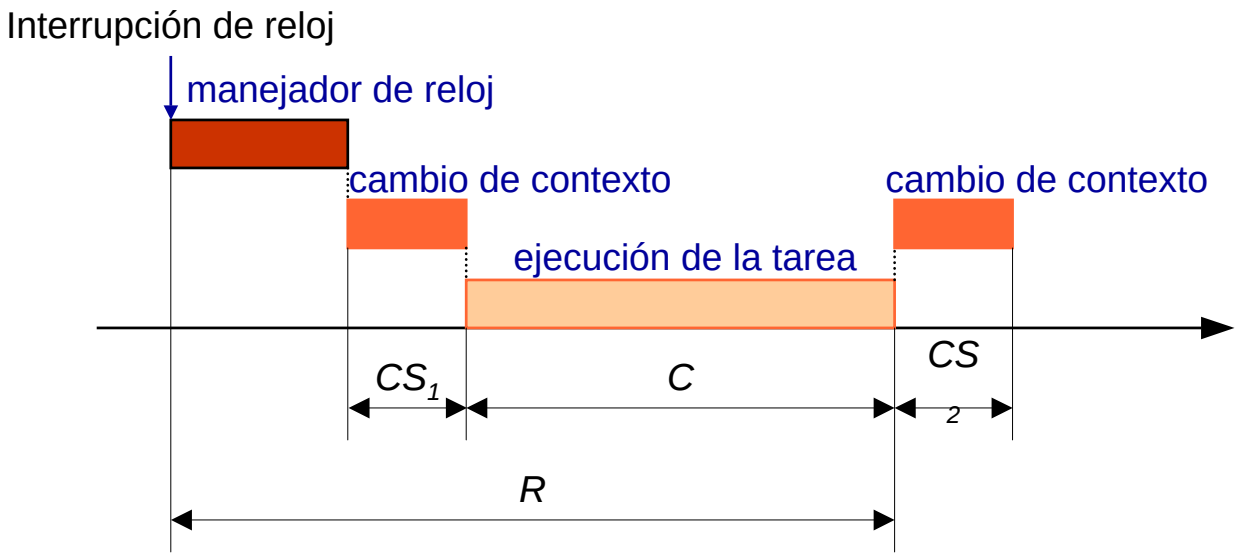
donde

S_{ij} es la duración máxima de una sección crítica invocada por τ_j sobre un objeto con techo de prioridad mayor o igual que P_i

S_K es la duración máxima de una sección no desalojable del núcleo

©1997-2000 Juan Antonio de la Puente

Cambio de contexto en tareas periódicas

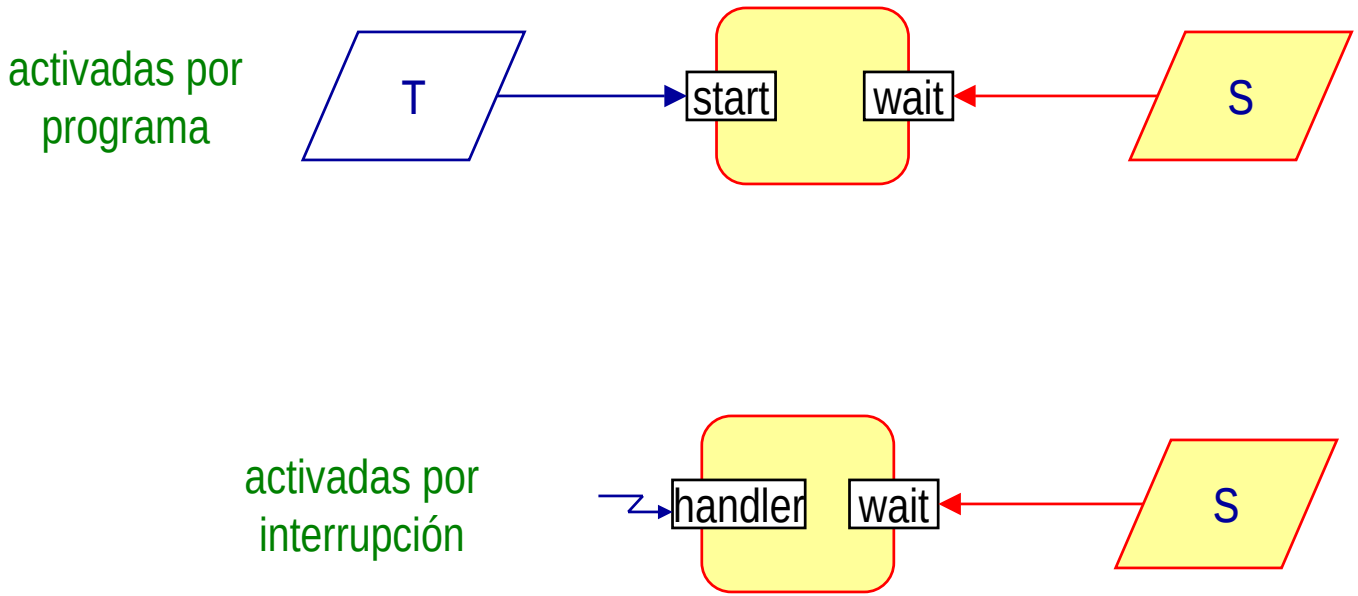


$$R_i = CS_1 + C_i + B_i + \sum_{j \in hp(i)} \left\lceil \frac{R_i}{T_j} \right\rceil \cdot \left(CS_1 + C_j + CS_2 \right)$$

- El manejador del reloj se modela aparte (interferencia)
- El coste del *delay* se incluye en C o en CS_2

©1997-2000 Juan Antonio de la Puente

Tareas esporádicas



©1997-2000 Juan Antonio de la Puente

Cálculo del tiempo de respuesta

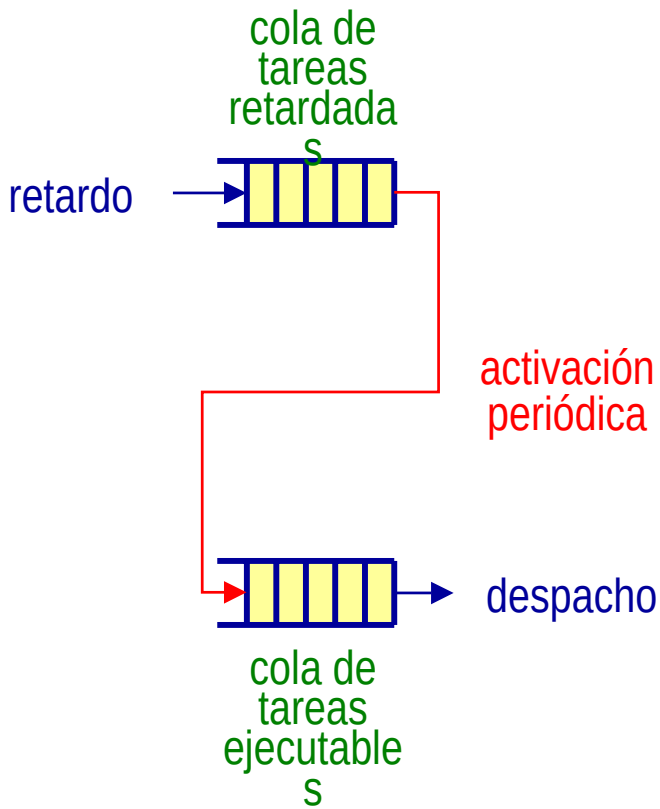
- **Tareas esporádicas activadas por software**
  - El modelo anterior es válido
  - El parámetro *C* debe incluir el coste de la operación de espera en el objeto protegido que controla la activación
- **Tareas esporádicas activadas por interrupción**
  - El manejador se ejecuta con prioridad alta
    - » puede producir interferencia y bloqueo en otras tareas

$$R_i = CS_1 + C_i + B_i + \sum_{j \in hp(i)} \left\lceil \frac{R_i}{T_j} \right\rceil \cdot \left(CS_1 + C_j + CS_2 \right) + \sum_{k \in INT} \left\lceil \frac{R_i}{T_k} \right\rceil \cdot C_{IH}$$

©1997-2000 Juan Antonio de la Puente

Manejador del reloj

- El manejador se comporta como una tarea periódica con período T_{clk}
- En cada ciclo saca de la cola de retardo todas las tareas cuyo retardo haya vencido y las pone en la cola de tareas ejecutables
  - la cola de tareas retardadas está ordenada por tiempo de activación
  - la cola de tareas ejecutables está ordenada por prioridades
  - el tiempo de ejecución depende de cuántas tareas se activen
  - el intervalo de variación es muy amplio



©1997-2000 Juan Antonio de la Puente

Tiempo de ejecución del manejador

- Consideramos tres componentes:
  - CT_c es el tiempo de cómputo constante correspondiente a las funciones que se ejecutan en cada ciclo de reloj
  - CT_s es el tiempo que se tarda en sacar la primera tarea que se activa de la cola de retardos y ponerla en la cola de tareas listas
  - CT_m es el tiempo que se tarda en sacar cada una de las demás tareas que se activan de la cola de retardos y ponerla en la cola de tareas listas
- El tiempo necesario para activar *n* tareas es:

$$C_{clk} = CT_C + CT_S + (n-1)CT_m$$

©1997-2000 Juan Antonio de la Puente

Ejemplo

Cola de tareas retardadas	C_{CLK}
Cola vacía	16 μs
0 tareas activadas	24 μs
1 tarea activada	88 μs
2 tareas activadas	128 μs
25 tareas activadas	1048 μs

Para este núcleo,

CT_C = 24 μs

CT_S = 64 μs

CT_M = 40 μs

©1997-2000 Juan Antonio de la Puente

Modelo elemental

- Podemos modelar el reloj como una sola tarea periódica con

$$C = \max C_{clk} = CT_c + CT_s + (N_P - 1)CT_m$$

$$T = T_{clk}$$

donde N_P es el número total de tareas periódicas

- Es muy pesimista
  - este valor solo se alcanza en los instantes críticos (una vez cada hiperperíodo)
- Podemos estimar mejor el efecto del manejador considerando que la activación de cada tarea se realiza al comienzo del período de la misma

©1997-2000 Juan Antonio de la Puente

Modelo detallado

- Modelamos el efecto del reloj mediante una *tarea ficticia* τ_i' por cada tarea periódica τ_i, con

  $T_i' = T_i$

  $C_i' = CT_s$

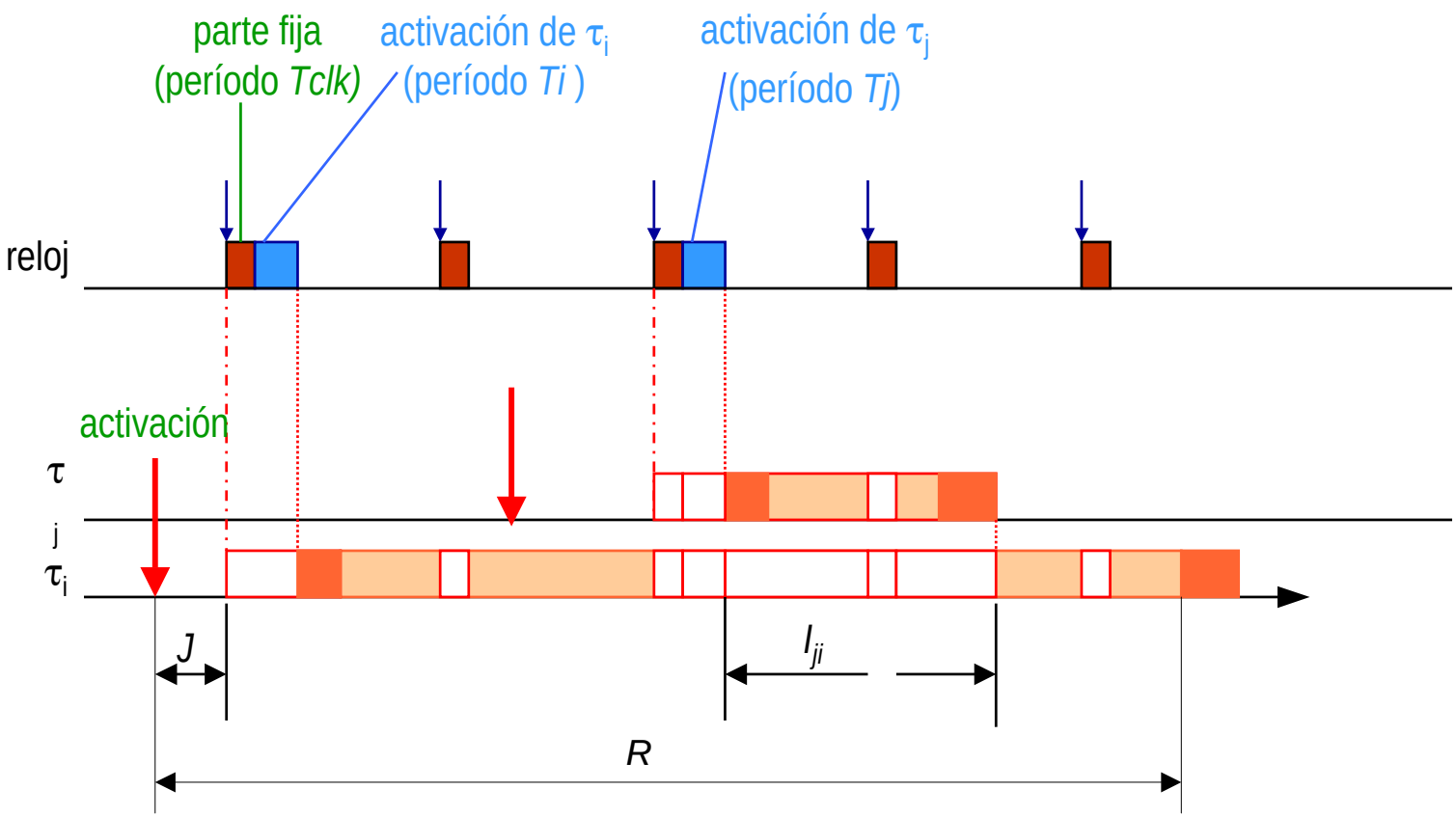
- Además hay que considerar la interferencia debida a la actividad básica del manejador (CT_C cada T_{clk})

$$R_i = CS_1 + C_i + B_i + \sum_{j \in hp(i)} \left\lceil \frac{R_i}{T_j} \right\rceil (CS_1 + C_j + CS_2)$$

$$+ \sum_{j \in INT} \left\lceil \frac{R_i}{T_j} \right\rceil C_{IH} + \left\lceil \frac{R_i}{T_{clk}} \right\rceil CT_C + \sum_{j \in PER} \left\lceil \frac{R_i}{T_j} \right\rceil CT_s$$

©1997-2000 Juan Antonio de la Puente

Interpretación del modelo



©1997-2000 Juan Antonio de la Puente

Granularidad en la activación

- Si los instantes de activación de las tareas periódicas no coinciden con los *tics* del reloj hay un efecto equivalente a un *jitter*

$$J_i = T_{clk} - \text{mcd}\left(T_{clk}, T_i\right)$$

- A veces las tareas esporádicas sólo se activan cuando se ejecuta el manejador de reloj. En este caso,

$$J_i = T_{clk}$$

©1997-2000 Juan Antonio de la Puente

Resumen

- El entorno de ejecución es un elemento clave de un sistema de tiempo real
- Conviene adaptar el tamaño y complejidad del entorno a las necesidades del sistema
- Hay que tener en cuenta el efecto del entorno de ejecución en los tiempos de respuesta

©1997-2000 Juan Antonio de la Puente