Grupo ARCOS
Departamento de Informática
Universidad Carlos III de Madrid

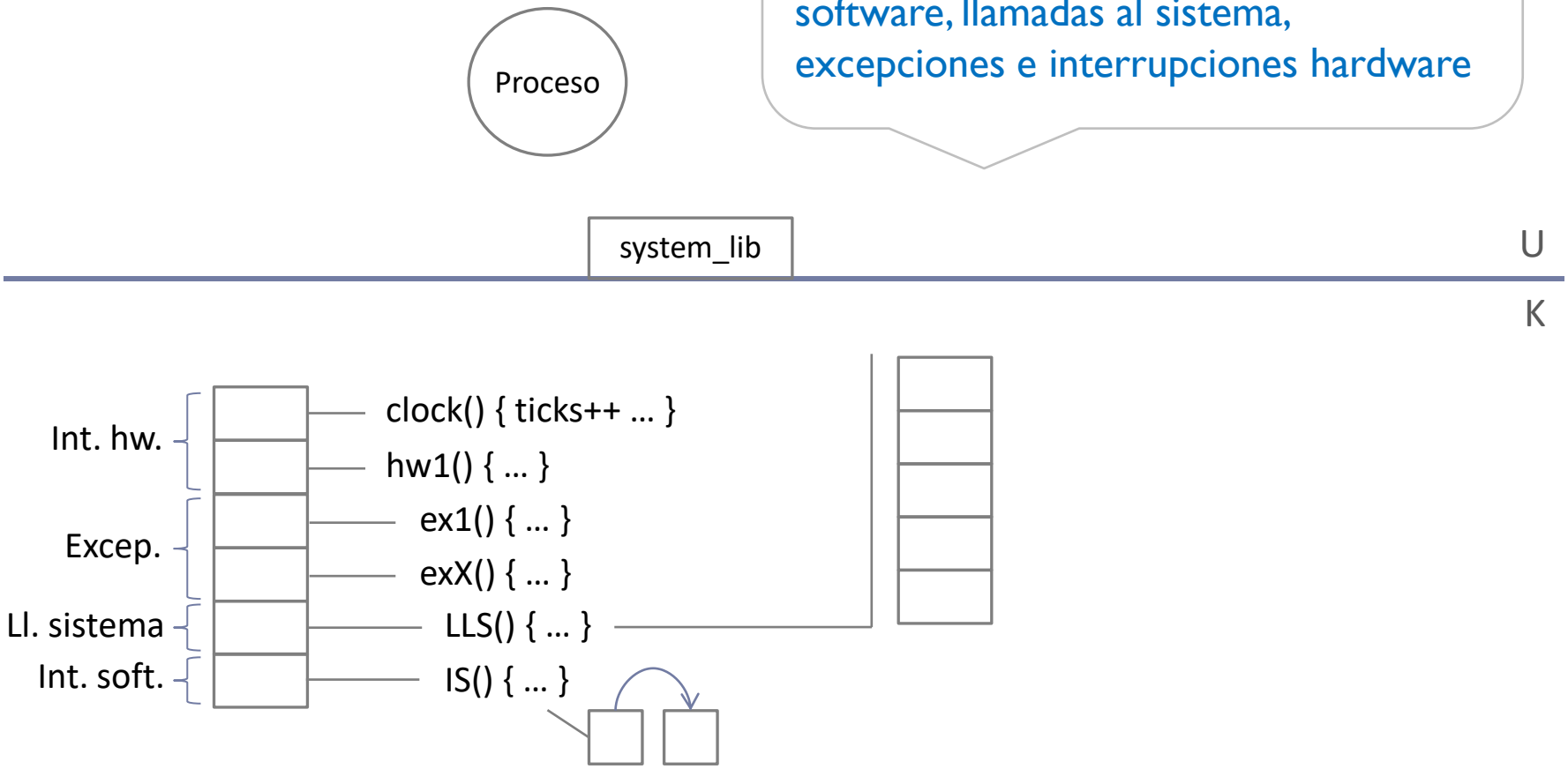
Lección 3a

procesos, periféricos, *drivers* y servicios ampliados

Diseño de Sistemas Operativos
Grado en Ingeniería Informática y Doble Grado I.I. y A.D.E.

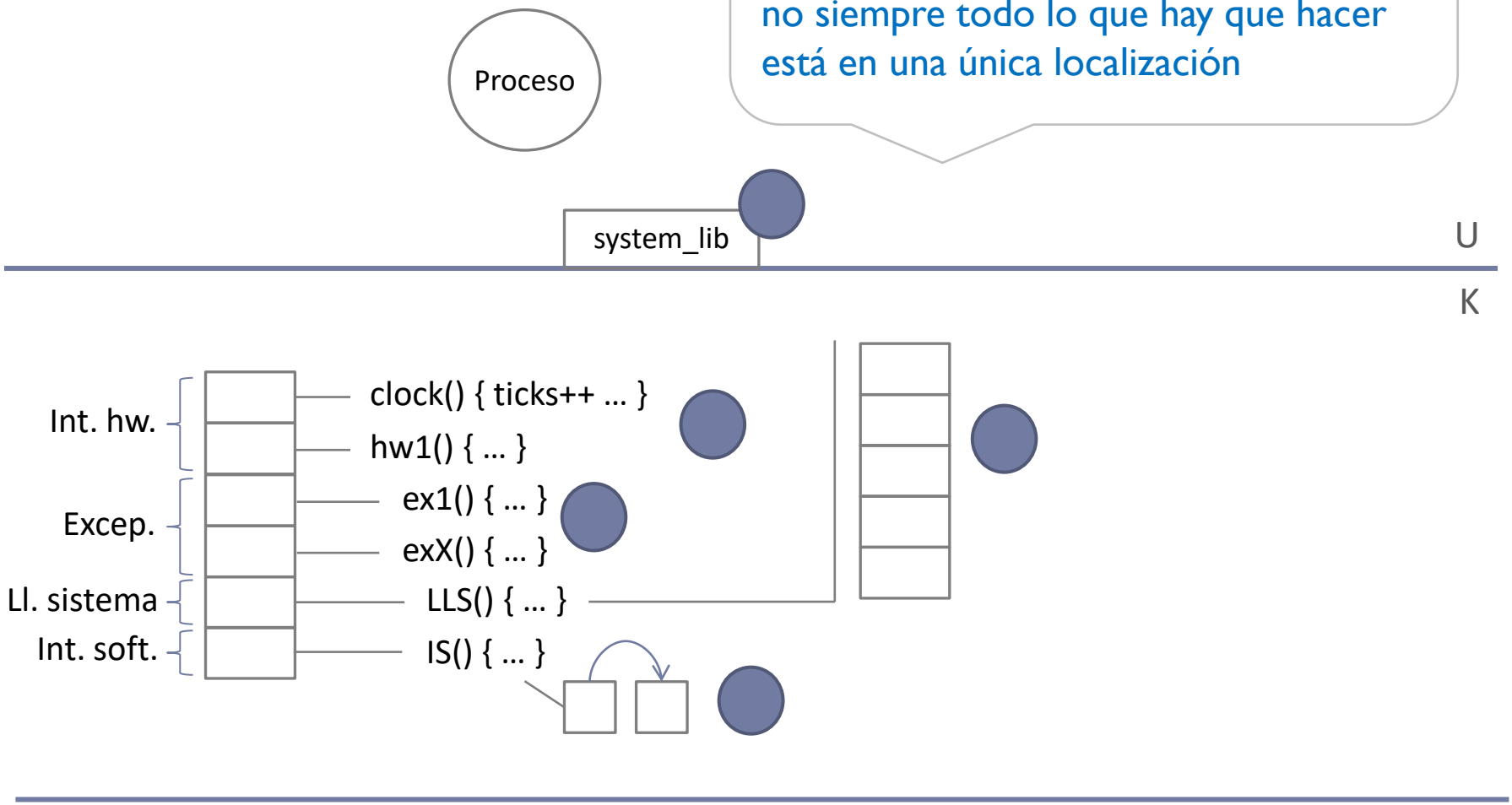
Contexto…

En el tema 2 se introducía el funcionamiento interno del núcleo del sistema operativo: interrupciones software, llamadas al sistema, excepciones e interrupciones hardware



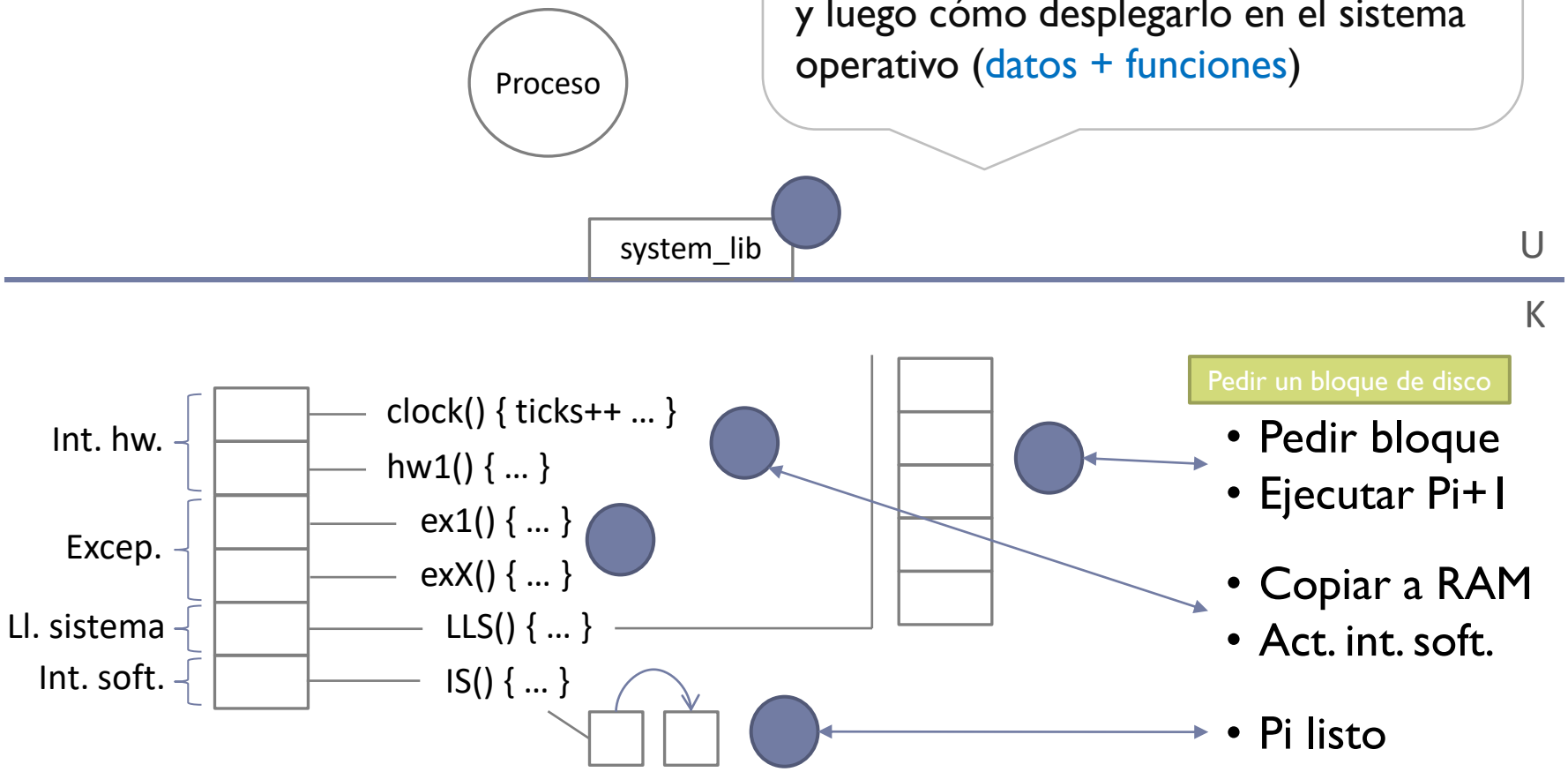
Contexto…

En el tema 2 se insistía del funcionamiento interno del núcleo del sistema operativo en que
no siempre todo lo que hay que hacer está en una única localización



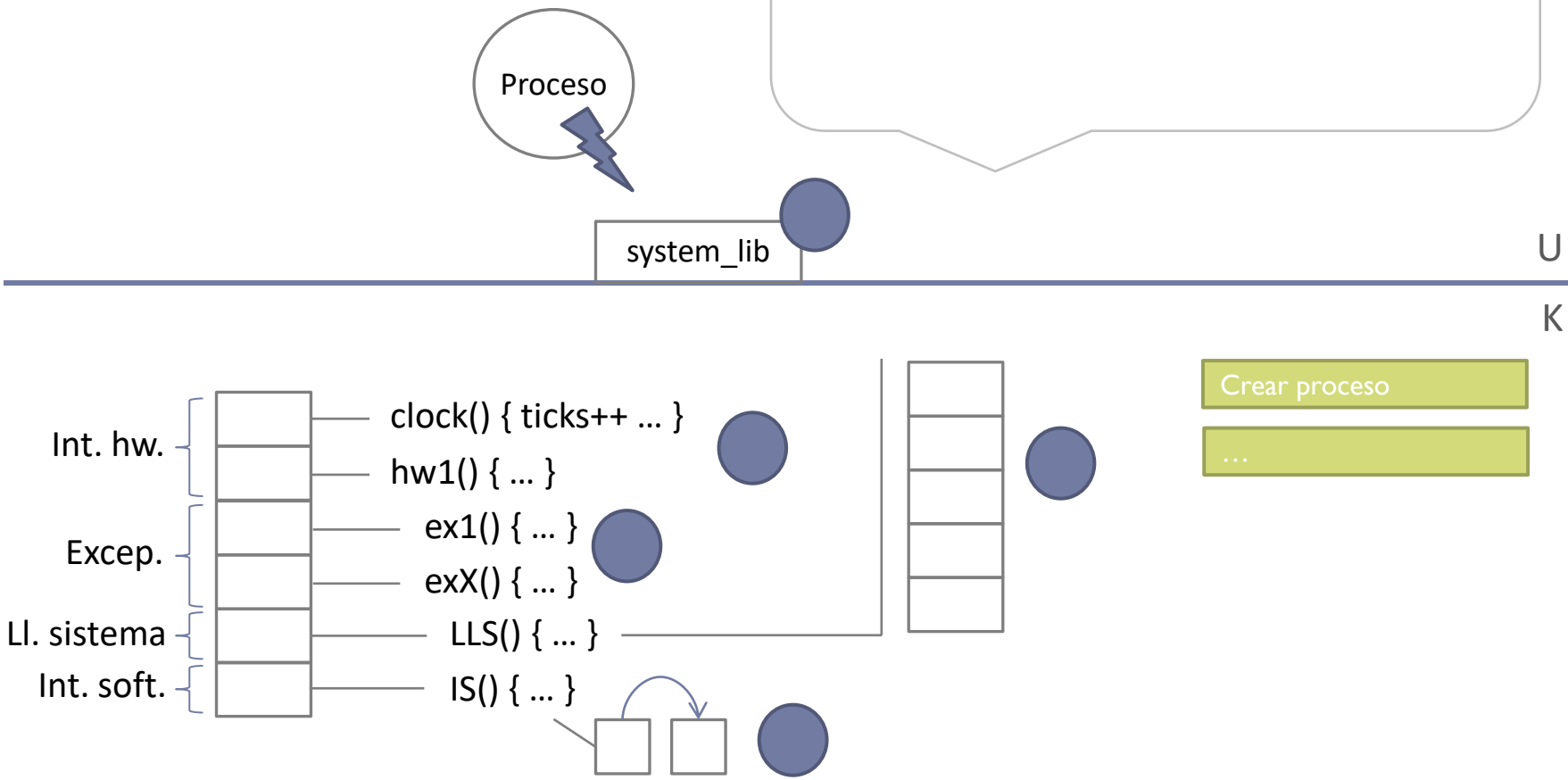
Contexto…

Primero conocer qué hay que hacer
(qué modelo de funcionamiento se desea tener)
y luego cómo desplegarlo en el sistema operativo (datos + funciones)



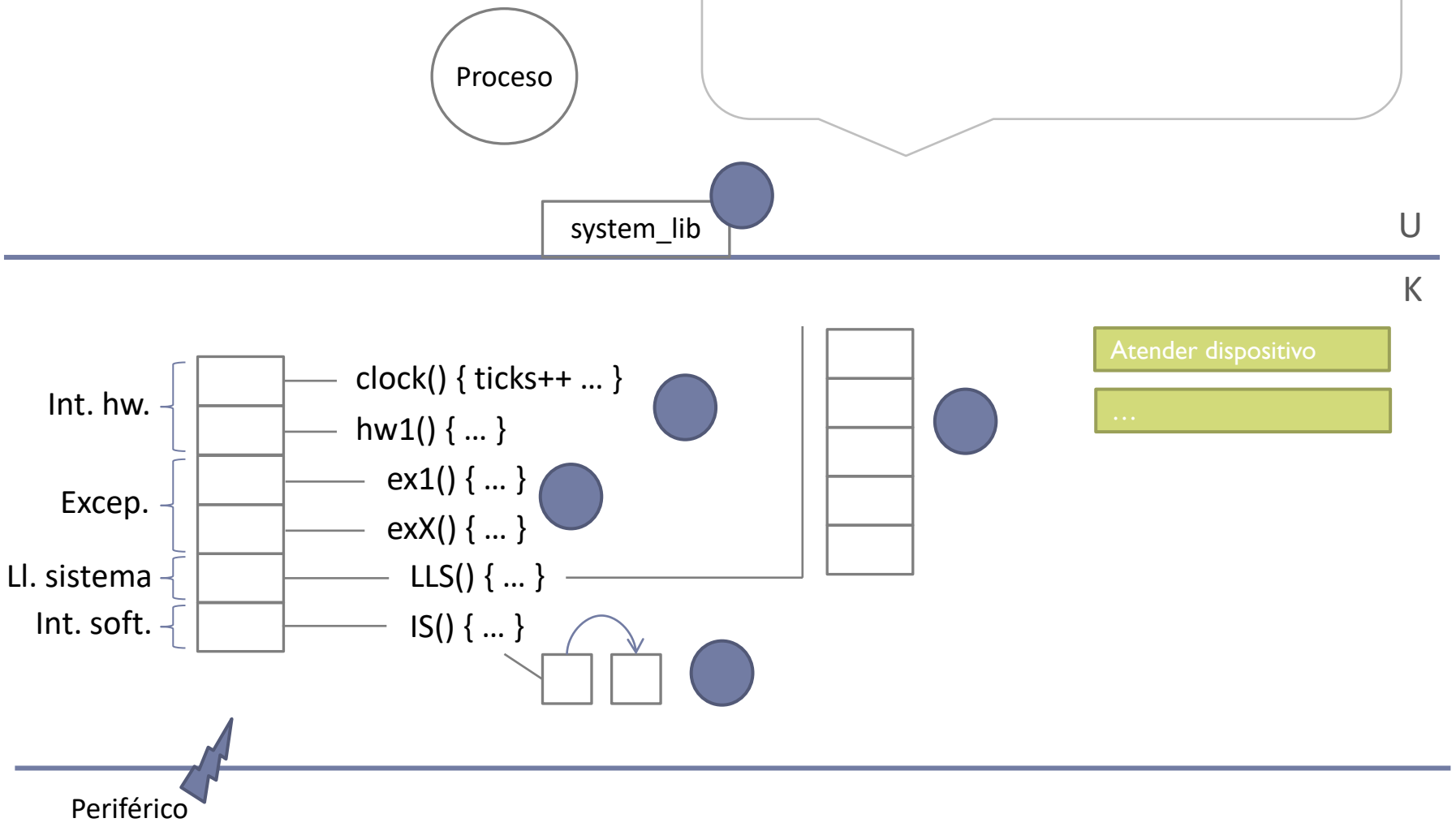
Objetivos…

En el tema 3 se introducirá los aspectos del funcionamiento relativos a la **gestión de procesos**…

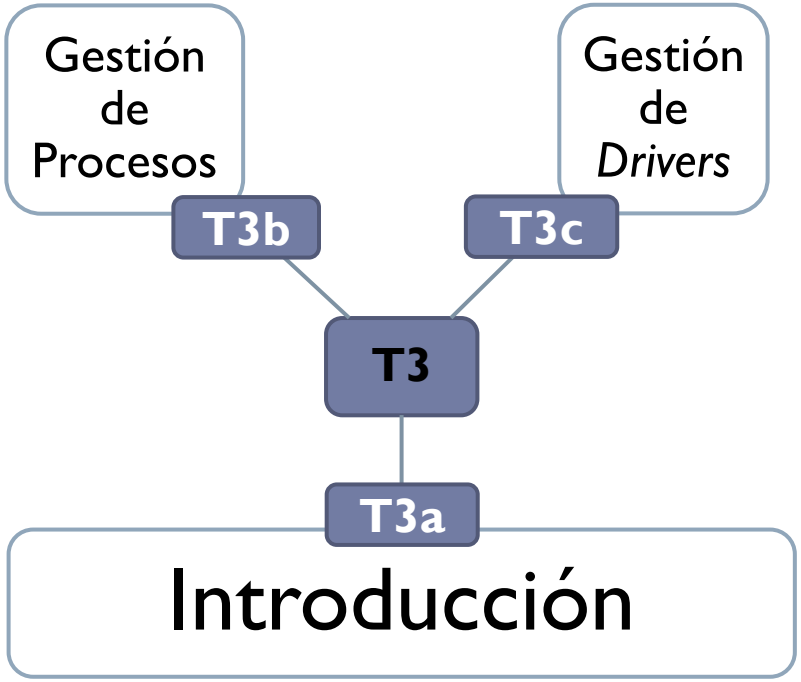


Objetivos…

En el tema 3 se introducirá los aspectos del funcionamiento relativos a la **gestión de dispositivos**…



Contexto…


Gestión de Procesos
Gestión de *Drivers*
T3b
T3c
T3
T3a
Introducción

A recordar…

Antes de clase

Clase

Después de clase

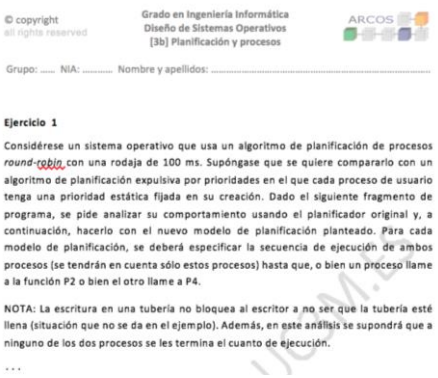
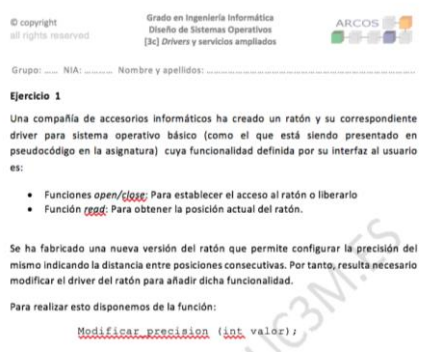
Preparar los pre-requisitos.

Estudiar el material asociado a la **bibliografía**:
las transparencias solo no son suficiente.
Preguntar dudas (especialmente tras estudio).

Ejercitar las competencias:

- Realizar todos los **ejercicios**.
- Realizar los **cuadernos de prácticas** y las **prácticas** de forma progresiva.

Ejercicios, cuadernos de prácticas y prácticas

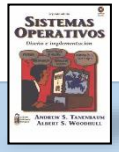
	Ejercicios ✔	Cuadernos de prácticas ✔	Prácticas ✔
b		DISEÑO DE SISTEMAS OPERATIVOS GRADO EN INGENIERÍA INFORMÁTICA DOBLE GRADO EN INGENIERÍA INFORMÁTICA Y ADMINISTRACIÓN DE EMPRESAS uc3m | Universidad Carlos III de Madrid	
c		**Introducción a un driver de teclado con Linux/Ubuntu**	Universidad Carlos III de Madrid **Planificación de procesos** David del Río Astorga

Lecturas recomendadas

Base

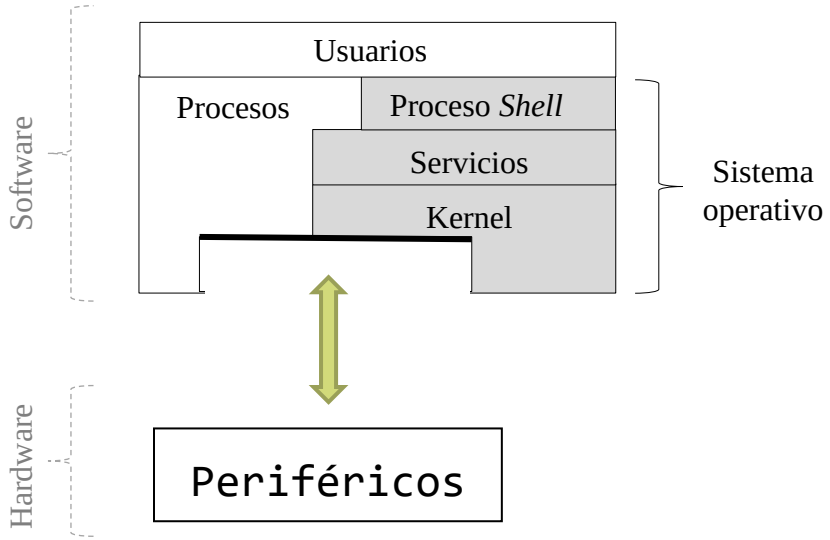
1. Carretero 2007:
   1. Cap.7

Recomendada



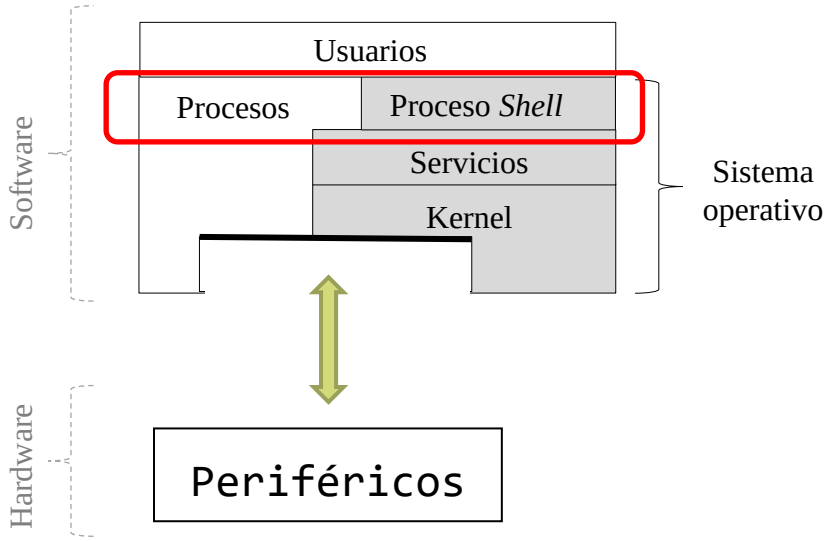
1. Tanenbaum 2006(en):
   1. Cap.3
2. Stallings 2005(en):
   1. Parte tres
3. Silberschatz 2006:
   1. Cap. Sistemas de E/S

Contenidos



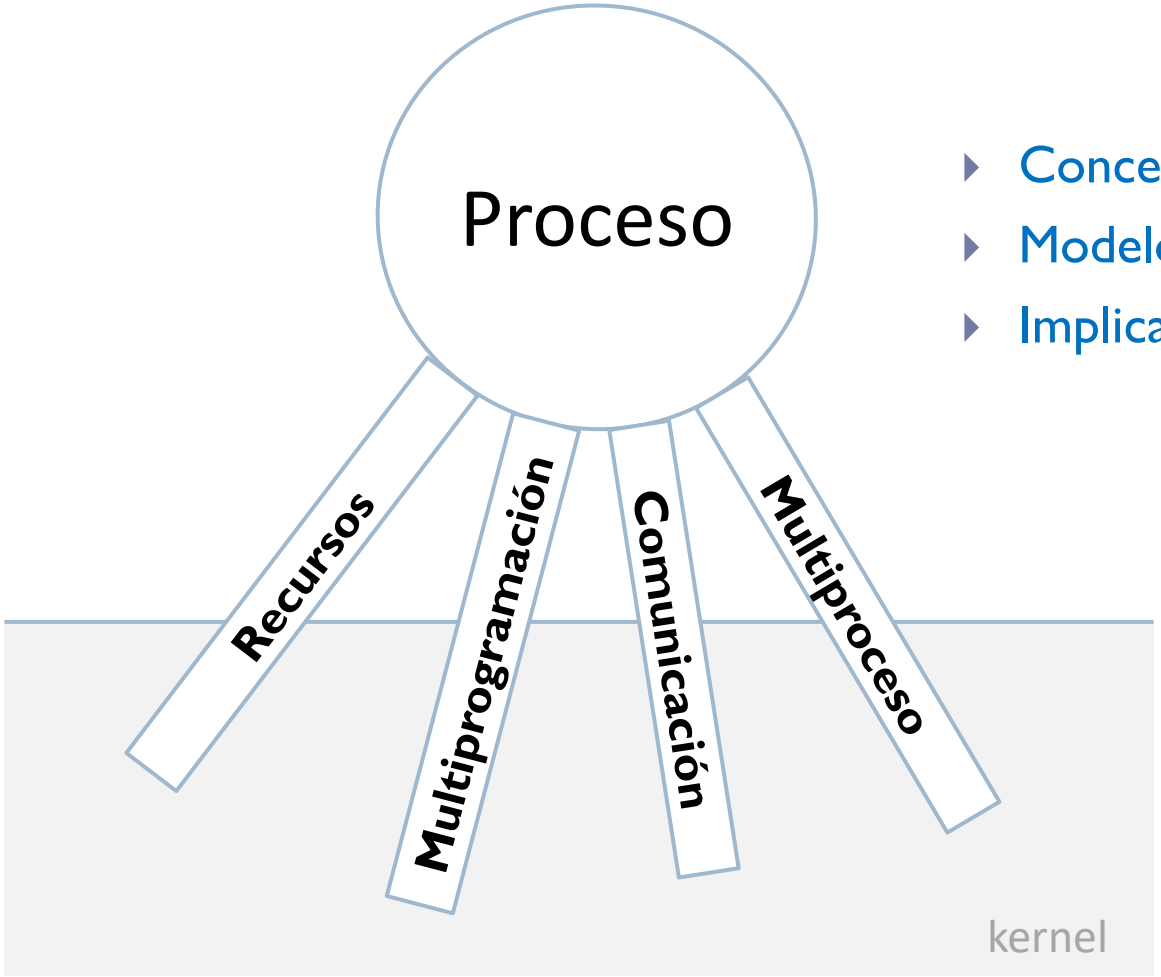
- Procesos
- Periféricos

Contenidos



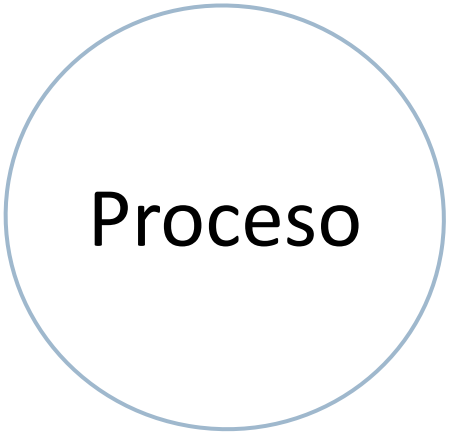
- **Procesos**
- Periféricos

Introducción



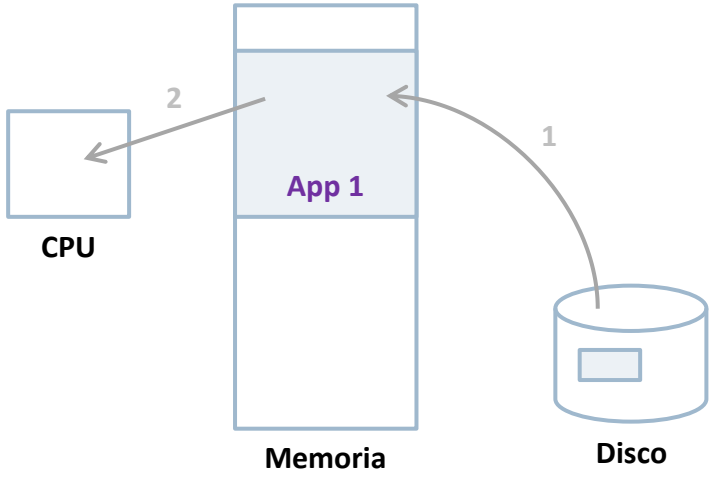
- Concepto de proceso
- Modelo ofrecido
- Implicaciones en S.O.

Introducción



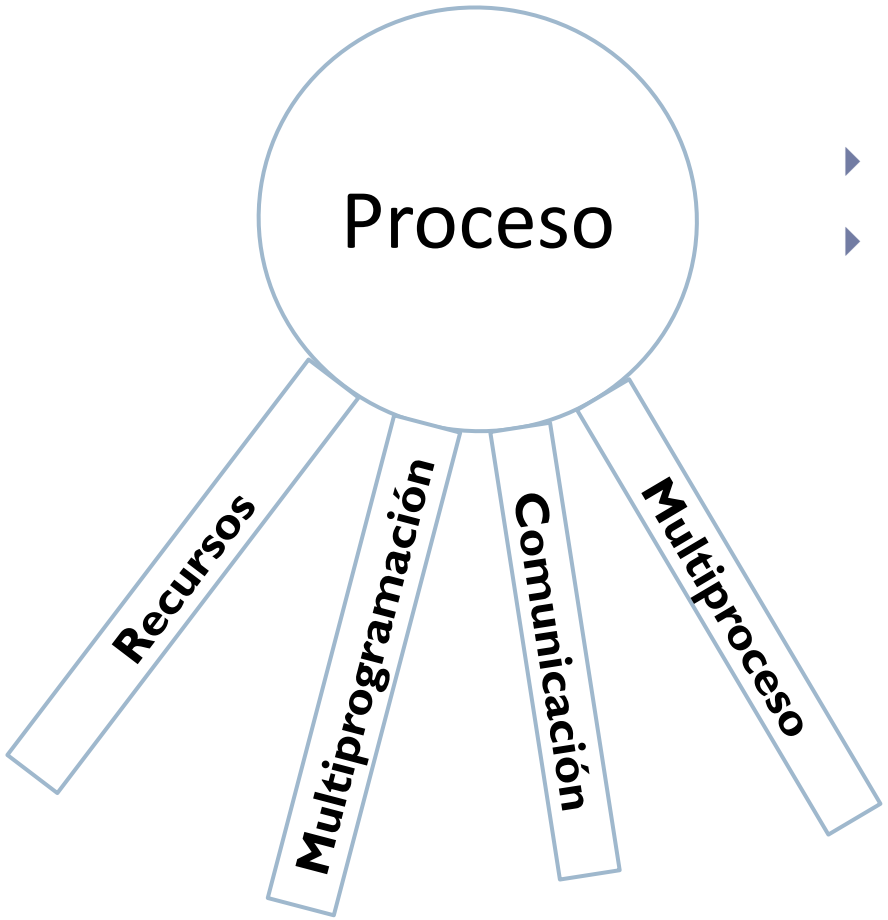
- **Concepto de proceso**

Concepto de proceso



- Proceso
  - Programa en ejecución
  - Unidad de procesamiento gestionada por el S.O.

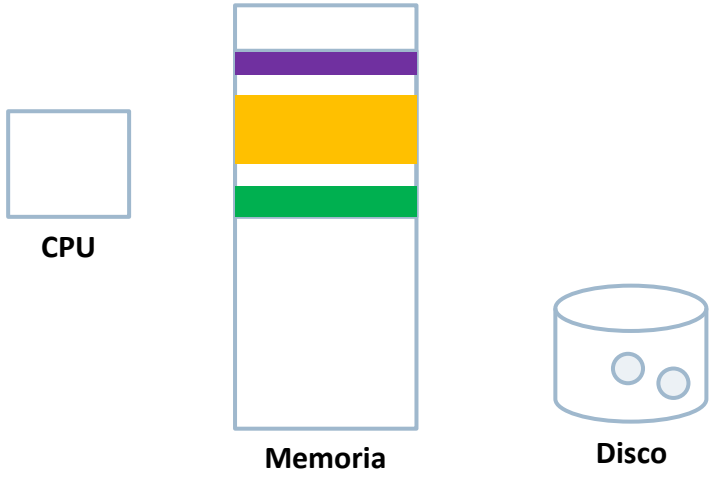
Introducción



- Concepto de proceso
- **Modelo ofrecido**

Modelo ofrecido

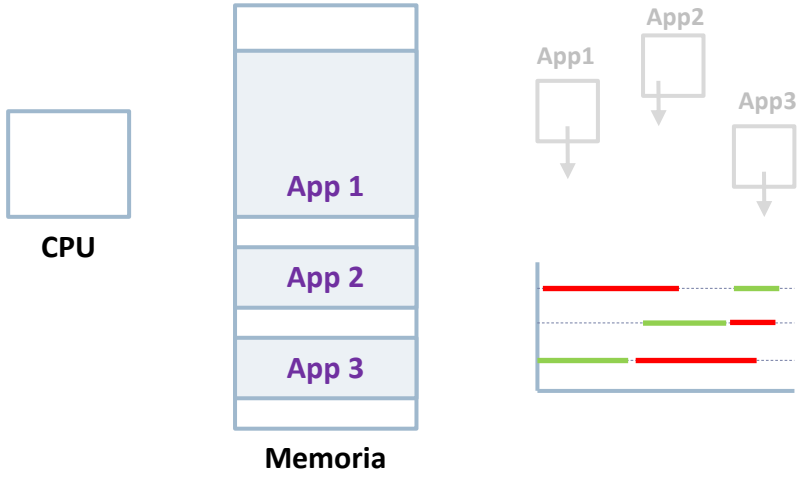
- recursos
- multiprogramación
  - protección/compartición
  - jerarquía de procesos
- multitarea
- multiproceso



- Recursos asociados
  - Zonas de memoria
    - Al menos: código, datos y pila
  - Archivos abiertos
  - Señales

Modelo ofrecido

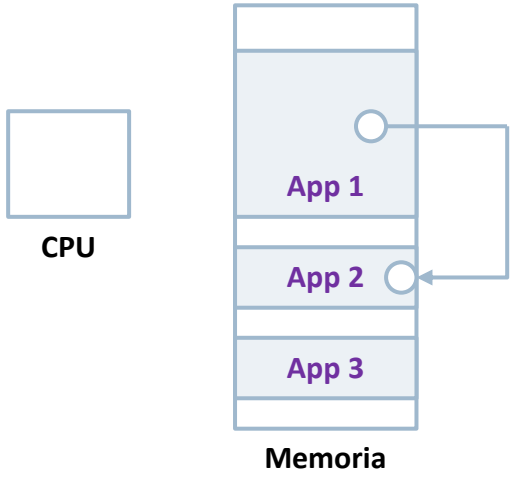
- recursos
- **multiprogramación**
  - protección/compartición
  - jerarquía de procesos
- multitarea
- multiproceso



- Multiprogramación
  - Tener varias aplicaciones en memoria
  - Si una aplicación se bloquea por E/S, entonces se ejecuta mientras otra hasta que quede bloqueada
    - Cambio de contexto voluntario (C.C.V.)
  - Eficiencia en el uso del procesador
  - Grado de multiprogramación = número de aplicaciones en RAM

Modelo ofrecido

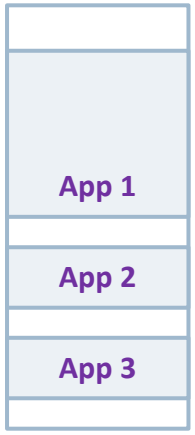
- recursos
- multiprogramación
  - **protección/compartición**
  - jerarquía de procesos
- multitarea
- multiproceso

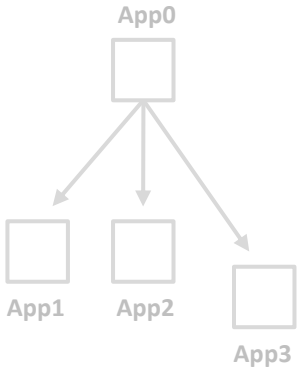


- Protección / Compartición
  - El espacio de direcciones privado por aplicación, pero
  - Posibilidad de comunicar datos entre dos aplicaciones
    - Paso de mensajes
    - Compartición de memoria

Modelo ofrecido

- recursos
- multiprogramación
  - protección/compartición
  - **jerarquía de procesos**
- multitarea
- multiproceso

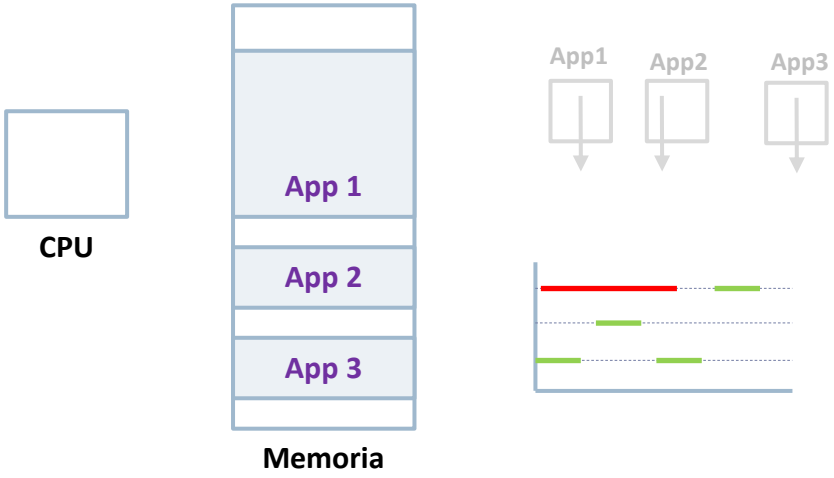




- Jerarquía de procesos
  - Creación de proceso
    - Como copia de otro proceso existente
    - A partir del programa en disco
    - Como proceso en el arranque
  - Grupo de procesos que comparten mismo tratamiento

Modelo ofrecido

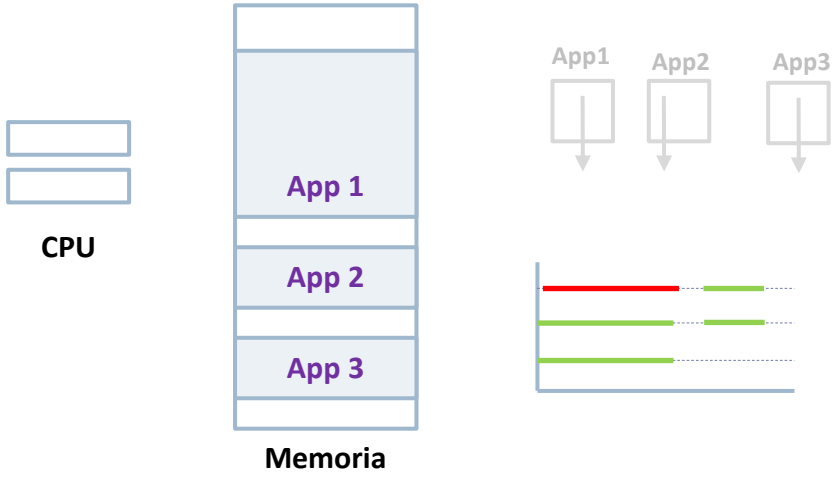
- recursos
- multiprogramación
  - protección/compartición
  - jerarquía de procesos
- **multitarea**
- multiproceso



- Multitarea
  - Cada proceso se ejecuta un quantum de tiempo (Ej.: 5 ms) y se rota el turno para ejecutar procesos no bloqueados
    - Cambio de contexto involuntario (C.C.I.)
  - Reparto del uso del procesador
    - Parece que todo se ejecuta a la vez

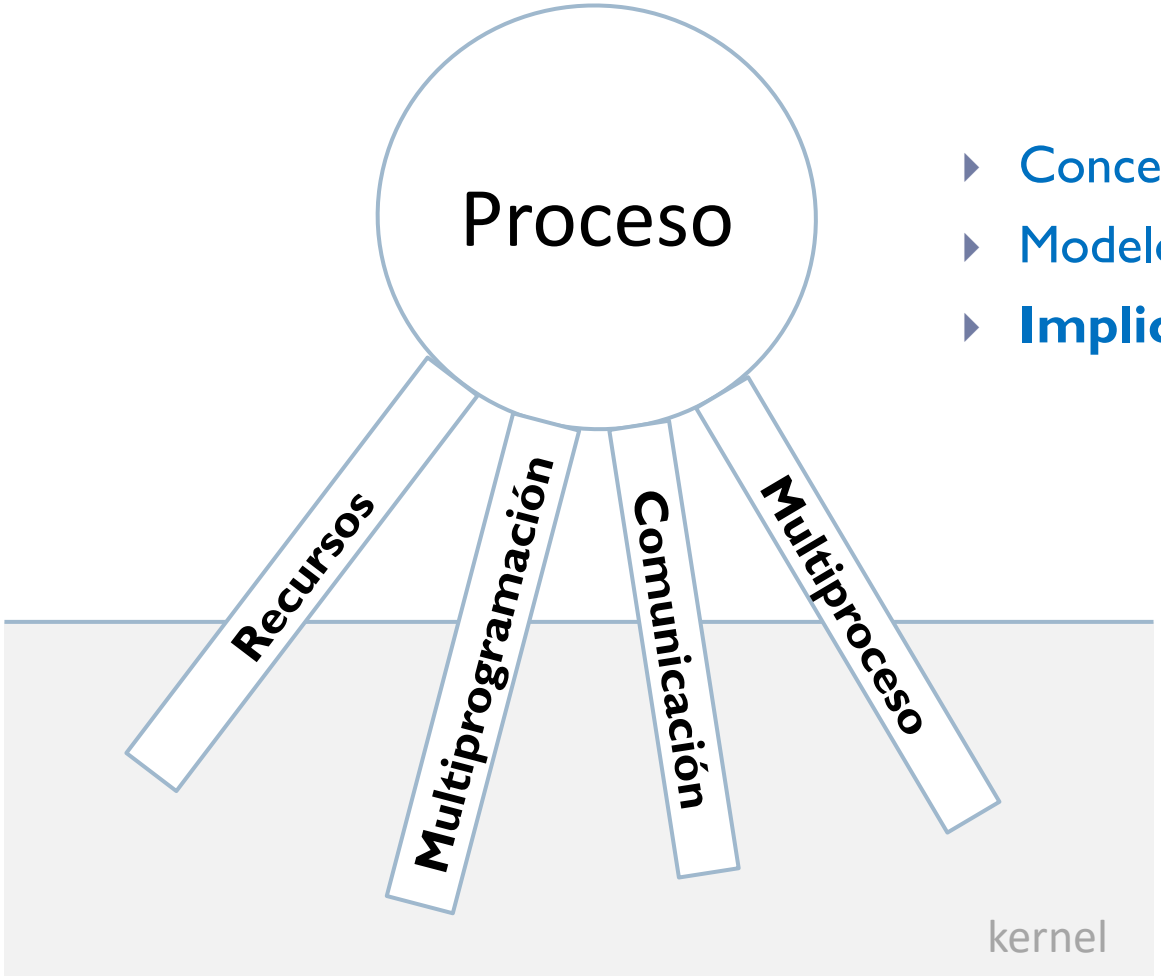
Modelo ofrecido

- recursos
- multiprogramación
  - protección/compartición
  - jerarquía de procesos
- multitarea
- **multiproceso**



- Multiproceso
  - Se dispone de varios procesadores (multicore/multiprocesador)
  - Además del reparto de cada CPU (multitarea) hay paralelismo real entre varias tareas (tantas como procesadores)
    - Se suele usar planificador y estructuras de datos separadas por procesador con algún mecanismo de equilibrio de carga

Introducción



- Concepto de proceso
- Modelo ofrecido
- **Implicaciones en S.O.**

Implicaciones en el sistema operativo

1. Estructuras de datos

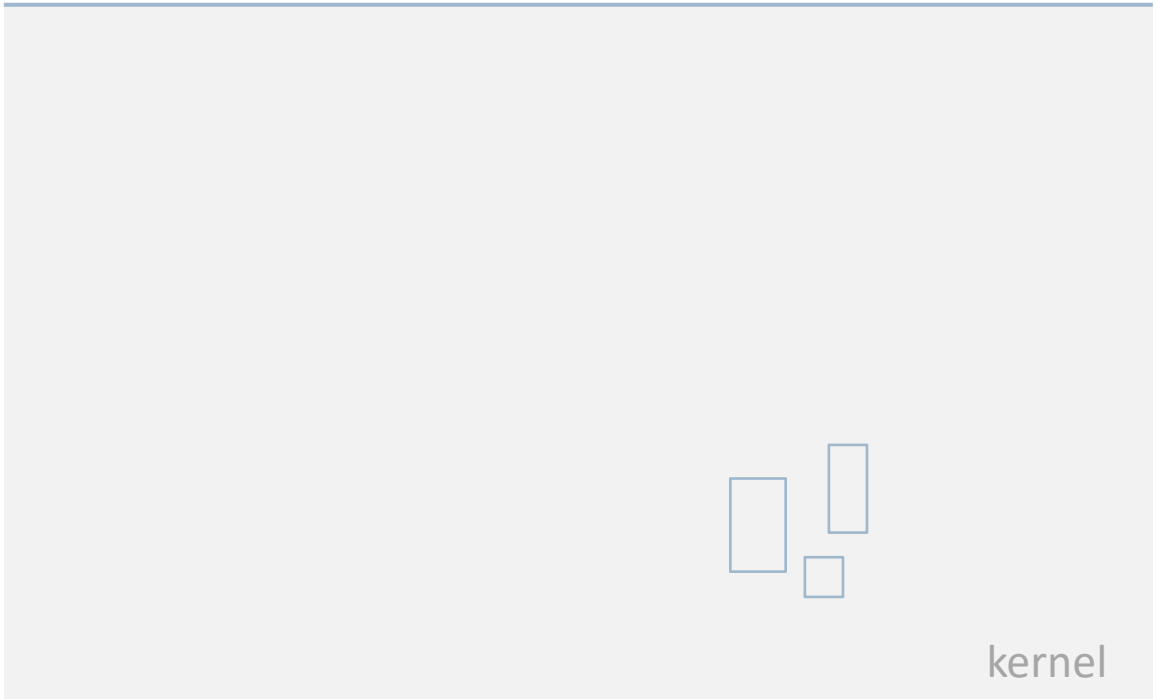
Requisitos	Información (en estructuras de datos)
Recursos	• Zonas de memoria (código, datos y pila) • Archivos abiertos • Señales activas
Multiprogramación	• Estado de ejecución • Contexto: registros de CPU… • Lista de procesos

○ Protección / Compartición	• Paso de mensajes
	  • Cola de mensajes de recepción
	• Memoria compartida
	  • Zonas, locks y conditions

○ Jerarquía de procesos	• Relación de parentesco • Conjuntos de procesos relacionados • Procesos de una misma sesión
Multitarea	• Quantum restante • Prioridad
Multiproceso	• Afinidad

Implicaciones en el sistema operativo

1. Estructuras de datos



Implicaciones en el sistema operativo

2. Funciones: de gestión internas

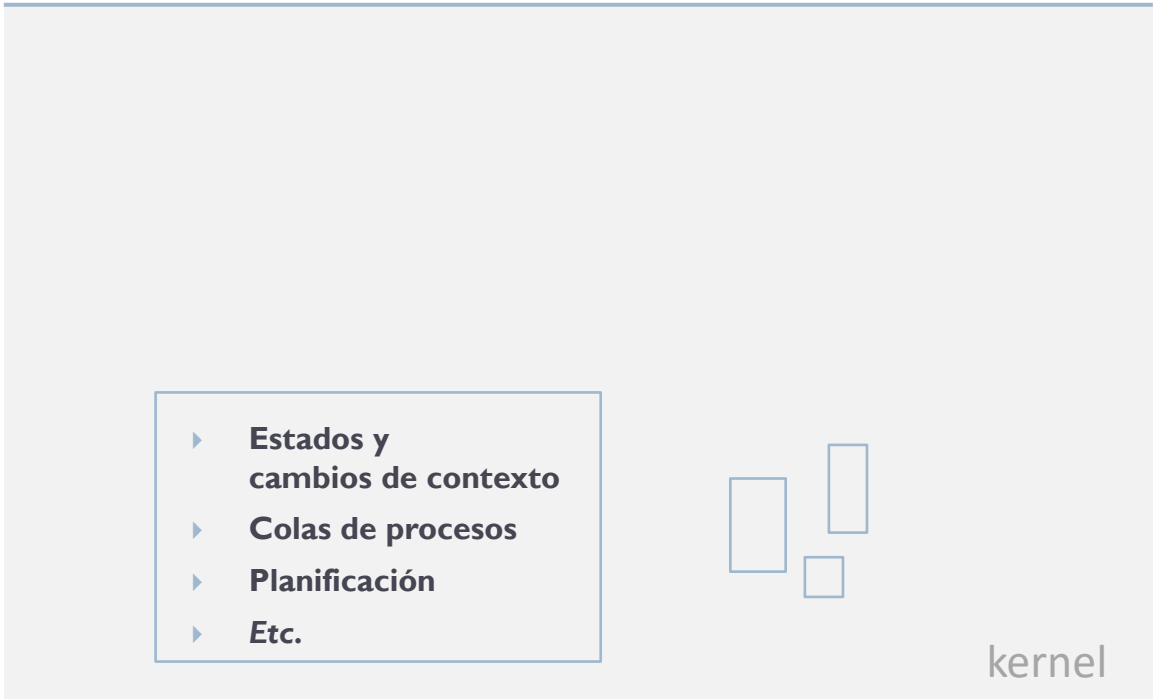
Requisitos	Información (en estructuras de datos)	Funciones (internas, servicio y API)
Recursos	• Zonas de memoria (código, datos y pila) • Archivos abiertos • Señales activas	• Diversas funciones internas • Diversas funciones de servicio para memoria, ficheros, etc.
Multiprogramación	• Estado de ejecución • Contexto: registros de CPU… • Lista de procesos	• Int. hw/sw de dispositivos • Planificador • Crear/Destruir/Planificar proceso

○ Protección / Compartición	• Paso de mensajes
	  • Cola de mensajes de recepción
	• Memoria compartida
	  • Zonas, locks y conditions
		• Envío/Recepción mensaje y gestión de la cola de mensaje
		• API concurrencia y gestión de estructuras de datos

○ Jerarquía de procesos	• Relación de parentesco • Conjuntos de procesos relacionados • Procesos de una misma sesión	• Clonar/Cambiar imagen de proceso • Asociar procesos e indicar proceso representante
Multitarea	• Quantum restante • Prioridad	• Int. hw/sw de reloj • Planificador • Crear/Destruir/Planificar proceso
Multiproceso	• Afinidad	• Int. hw/sw de reloj • Planificador • Crear/Destruir/Planificar proceso

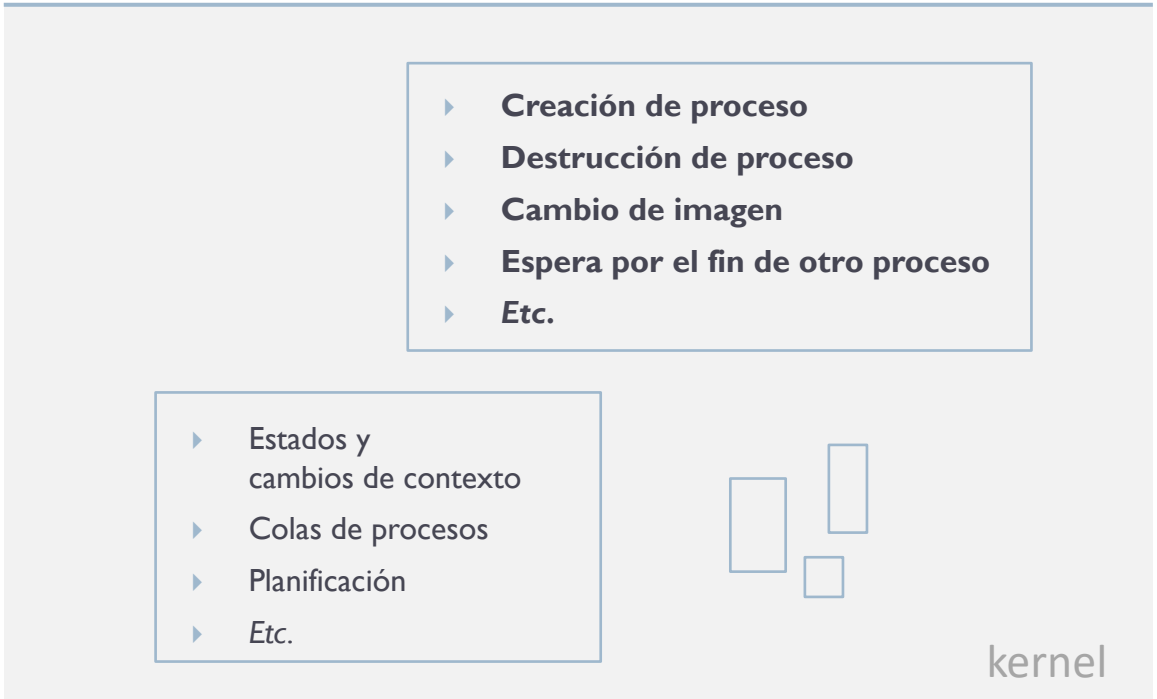
Implicaciones en el sistema operativo

2. Funciones: de gestión internas



Implicaciones en el sistema operativo

3. Funciones: de servicio



Implicaciones en el sistema operativo

3. Funciones: API de servicio

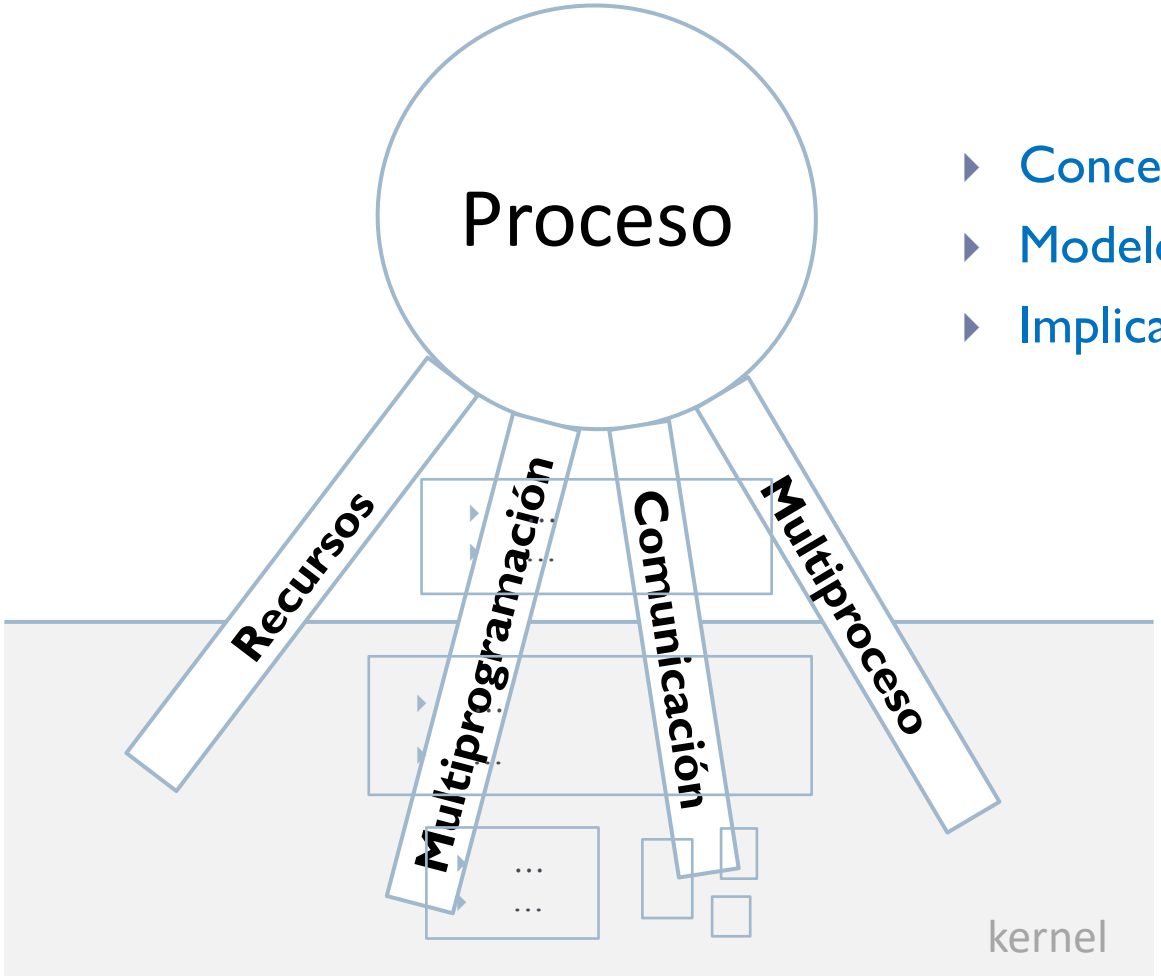
- fork, exit, exec, wait, …
- *pthread_create, pthread…*

- Creación de proceso
- Destrucción de proceso
- Cambio de imagen
- Espera por el fin de otro proceso
- *Etc.*

- Estados y cambios de contexto
- Colas de procesos
- Planificación
- *Etc.*

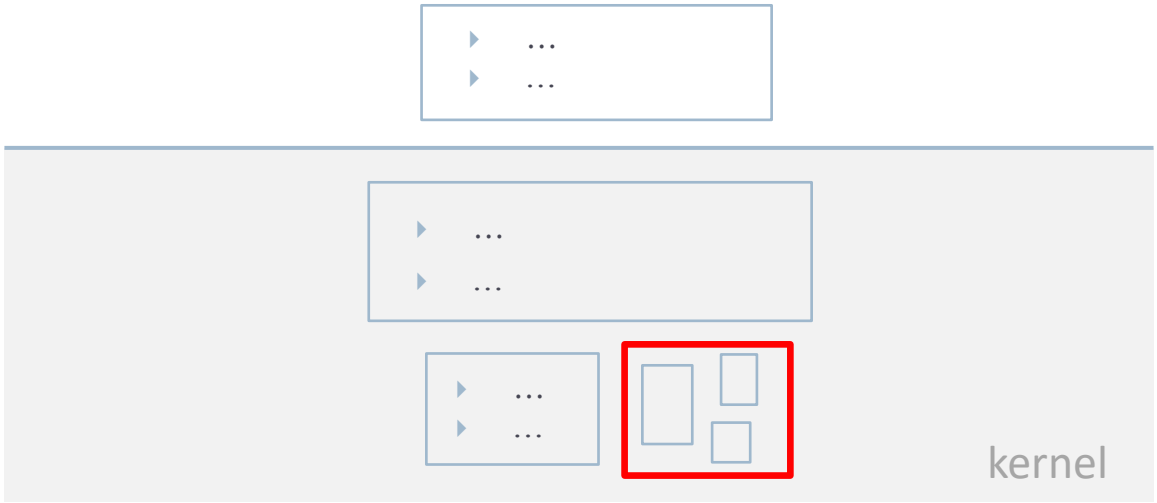
kernel

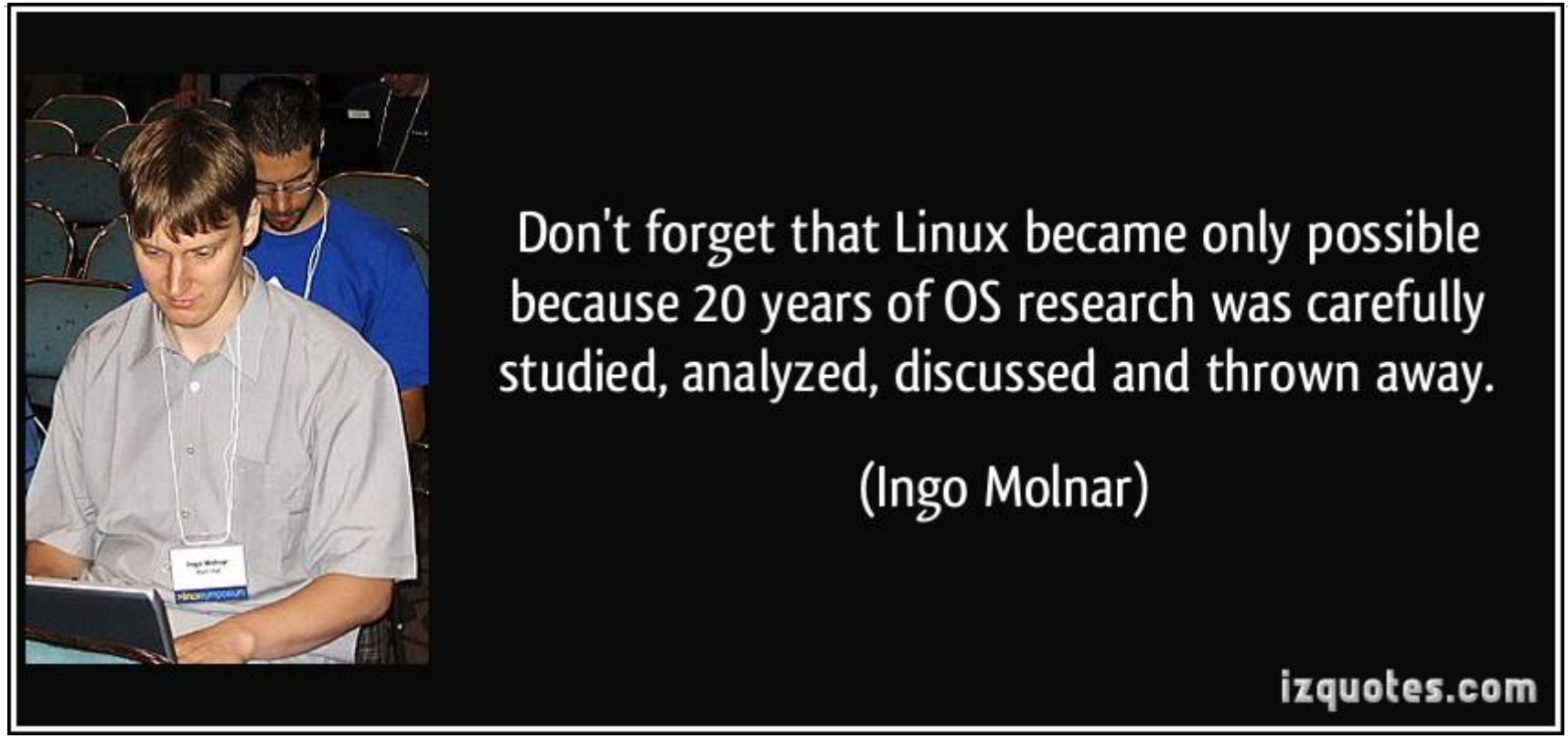
Introducción
resumen



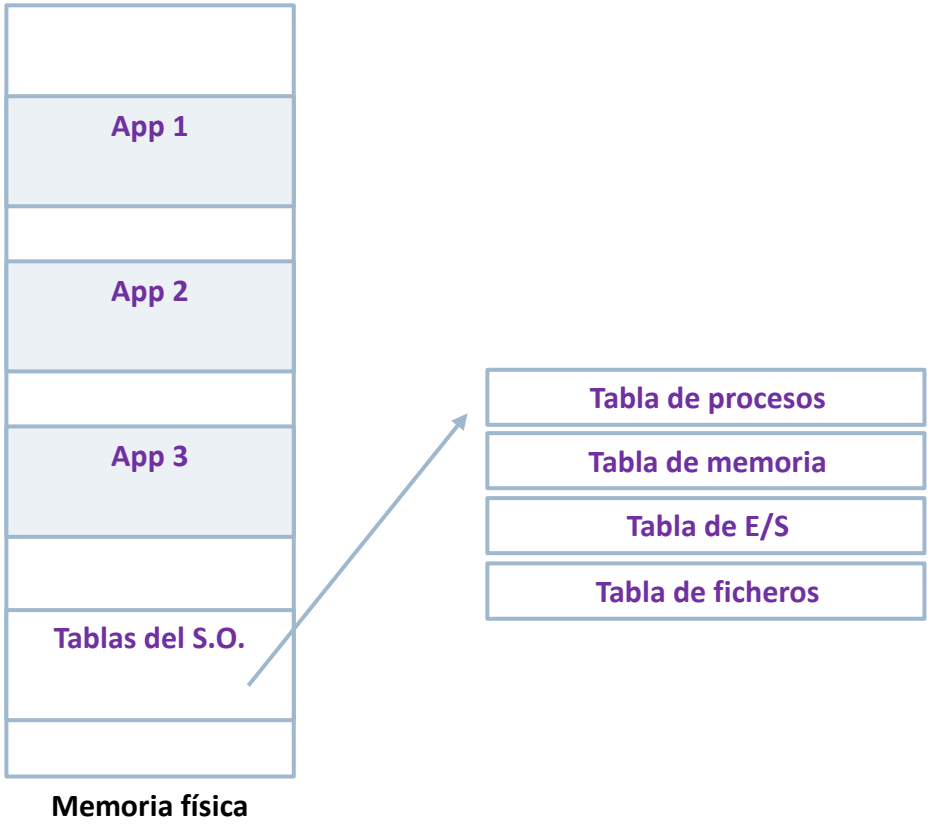
- Concepto de proceso
- Modelo ofrecido
- Implicaciones en S.O.

Principales estructuras de datos

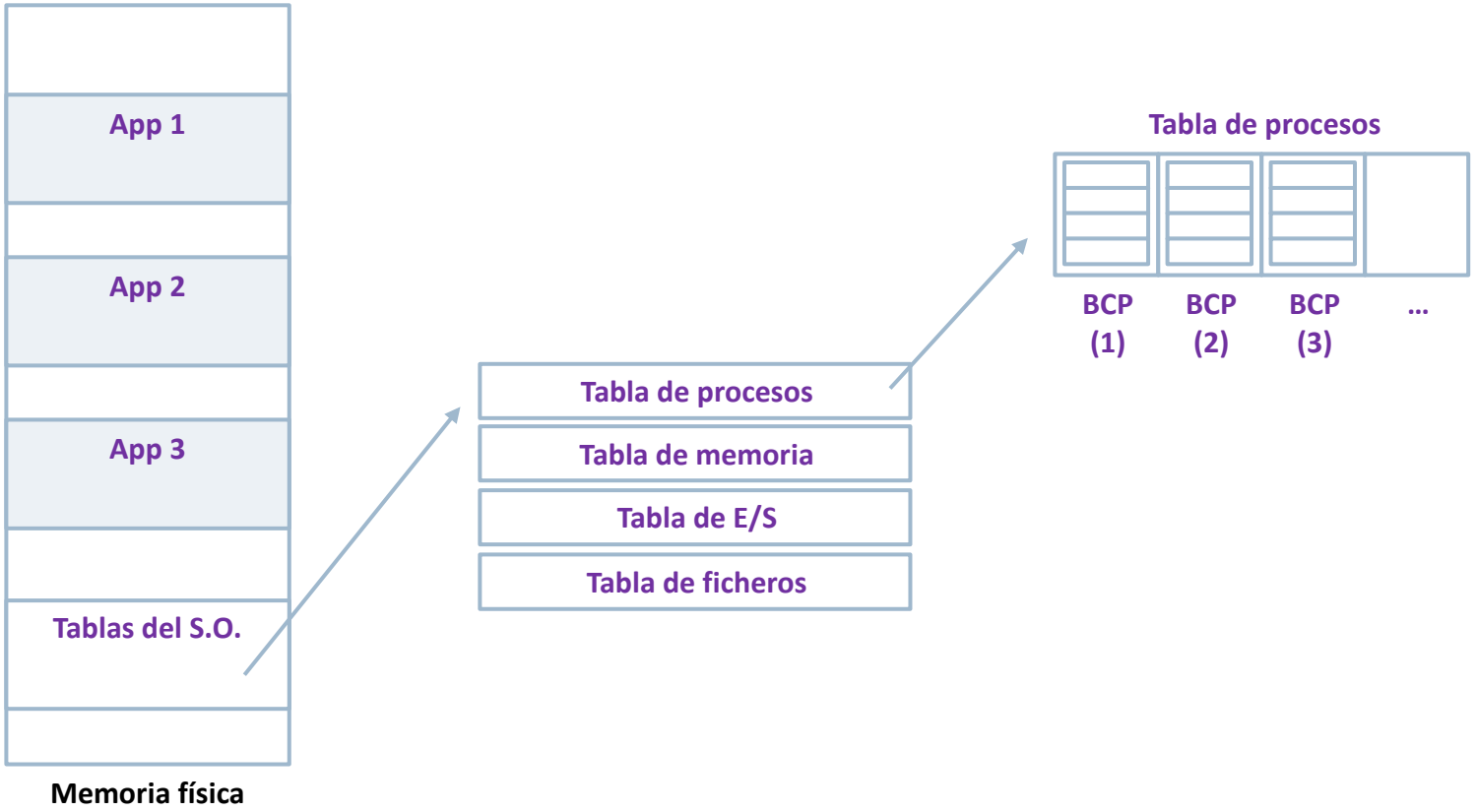



Don't forget that Linux became only possible because 20 years of OS research was carefully studied, analyzed, discussed and thrown away.
(Ingo Molnar)
izquotes.com

Información en el sistema operativo



Información para un proceso



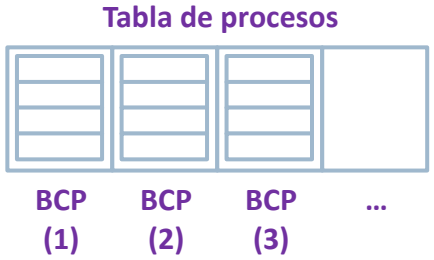
BCP: entrada de la tabla de procesos

- Gestión de proceso

estado
- Registros generales
- Contador de programa
- Registro de estado
- Puntero de pila

Id.
- Identificador del proceso
- Proceso padre
- Grupo de proceso

gestión
- Prioridad
- Parámetros del planificador
- Señales
- Instante inicio de ejecución
- Tiempo de uso de CPU
- Tiempo hasta siguiente alarma



- *Process Control Block* (PCB / BCP)
  - Estructura de datos con la información necesaria para gestionar un proceso en particular
  - Manifestación de un proceso en el kernel
- *Thread Control Block* (TCB / BCT)
  - Similar al BCP para cada hilo de un proceso

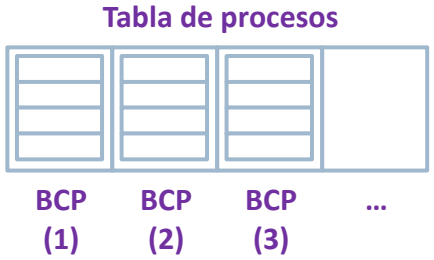
BCP: entrada de la tabla de procesos

- Gestión de proceso

estado
- Registros generales
- Contador de programa
- Registro de estado
- Puntero de pila

Id.
- Identificador del proceso
- Proceso padre
- Grupo de proceso

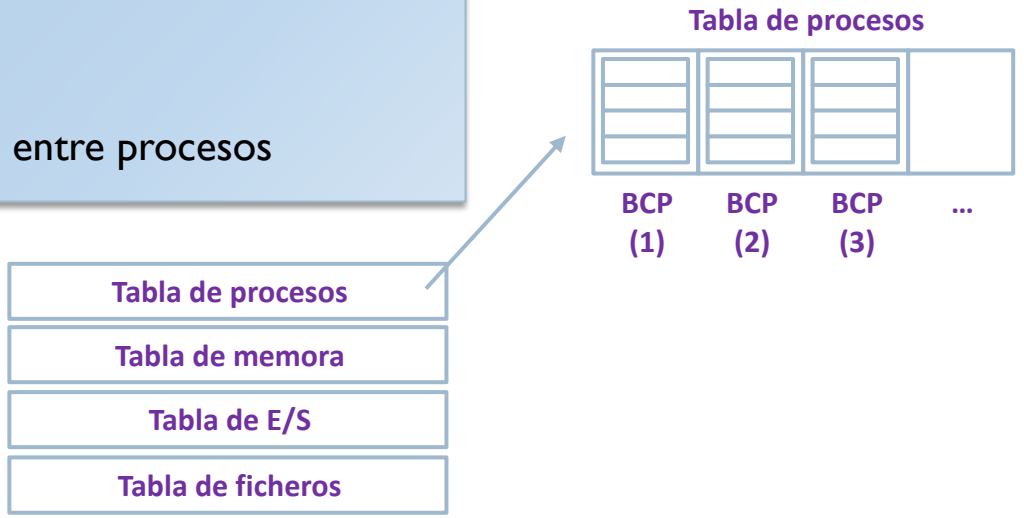
gestión
- Prioridad
- Parámetros del planificador
- Señales
- Instante inicio de ejecución
- Tiempo de uso de CPU
- Tiempo hasta siguiente alarma



- *Process Identification* (PID)
  - Identificador de cara a los usuarios
  - Número positivo de 16 bits (32767) dinámicamente asignado, reusado no de forma inmediata
- *Address of process descriptor* (APD)
  - Identificación dentro del kernel
  - Existe mecanismo PID -> APD (Ej.: hash)

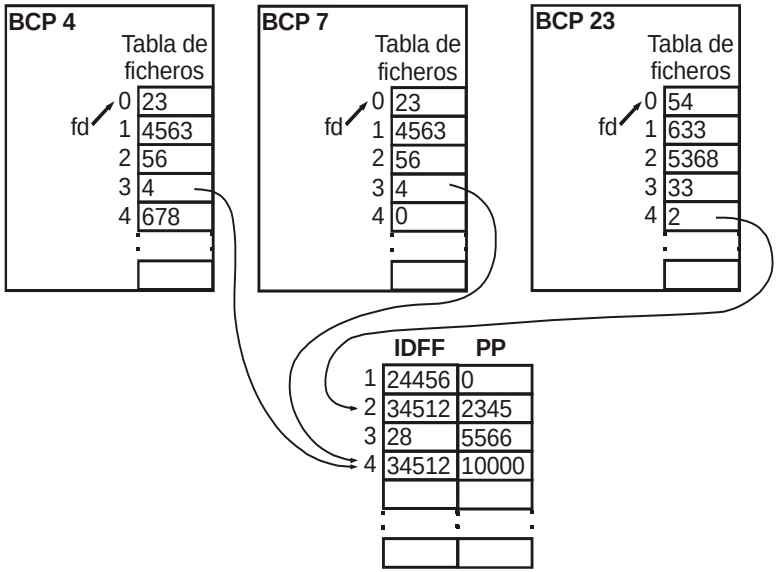
Dónde: información del proceso

- La información de cada proceso está en el BCP…
- Información fuera del BCP:
  - Por razones de eficiencia
  - Para compartir información entre procesos



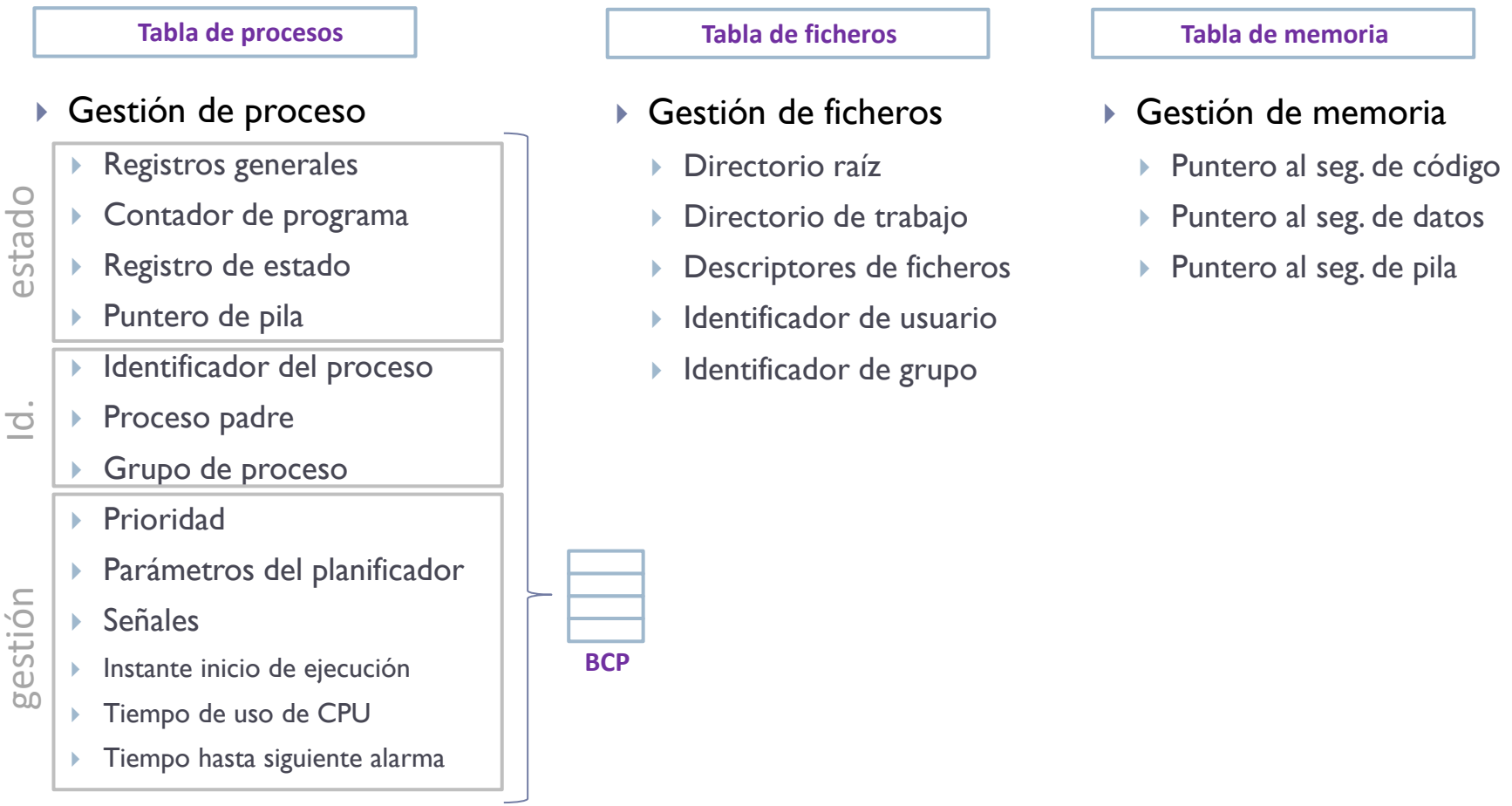
- Ejemplos:
  - Tabla de segmentos y páginas de memoria
  - Tabla de punteros de posición de ficheros
  - Lista de peticiones a dispositivos

Dónde: información del proceso

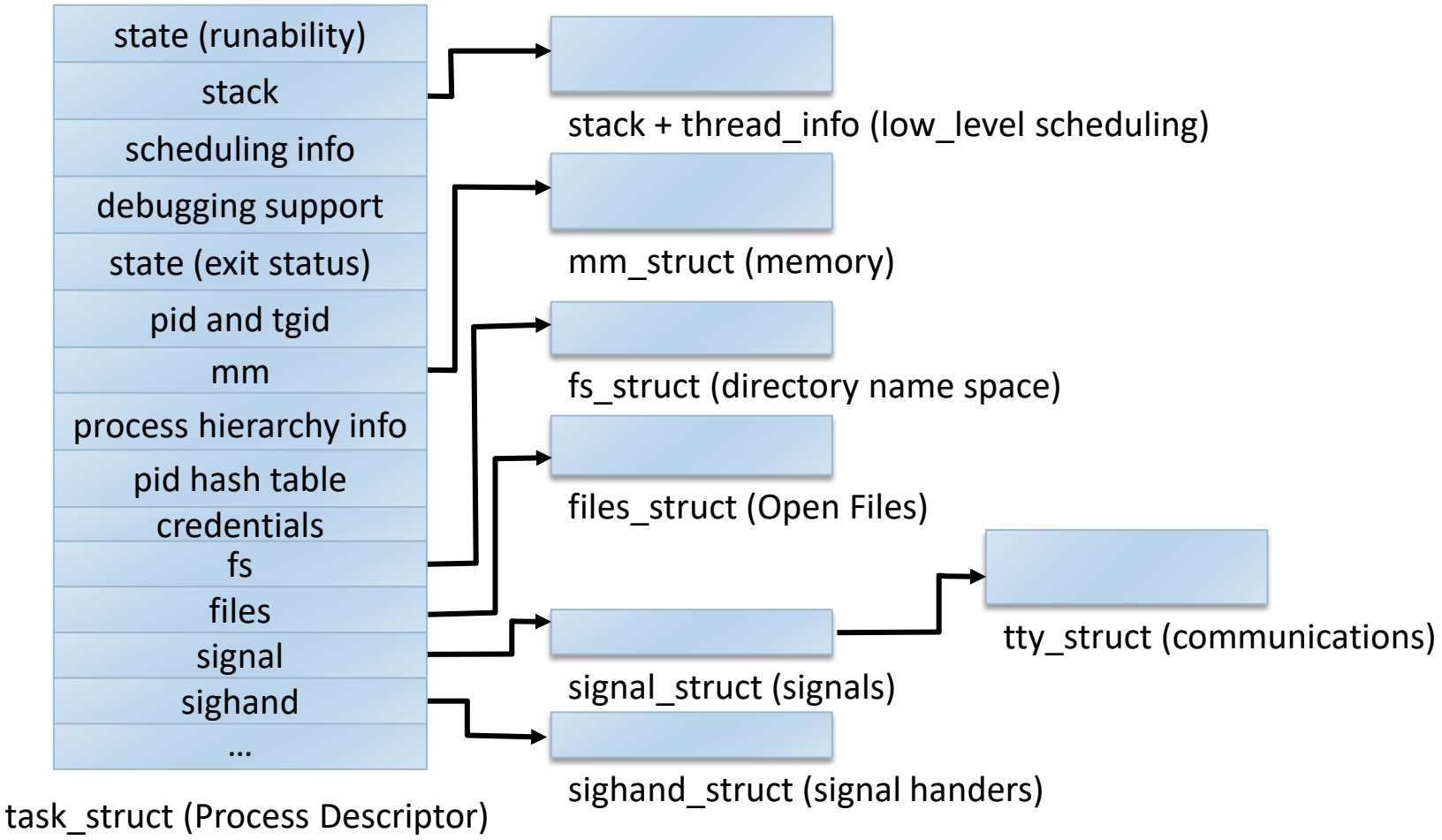


- Tabla de punteros de posición de ficheros:
  - Describe la posición de lectura/escritura de los ficheros abiertos
  - La compartición de estado del fichero entre procesos obliga a que sea externa al BCP
  - El BCP contiene el índice del elemento de la tabla que contiene la información del fichero abierto: el i-nodo y la posición de lectura/escritura.

Información del proceso
resumen



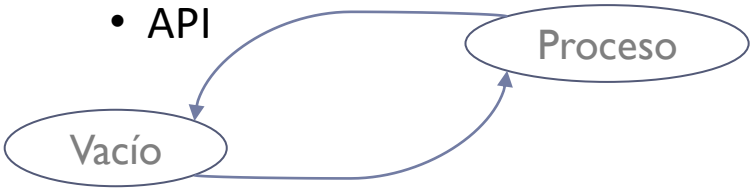
Información del proceso
Linux



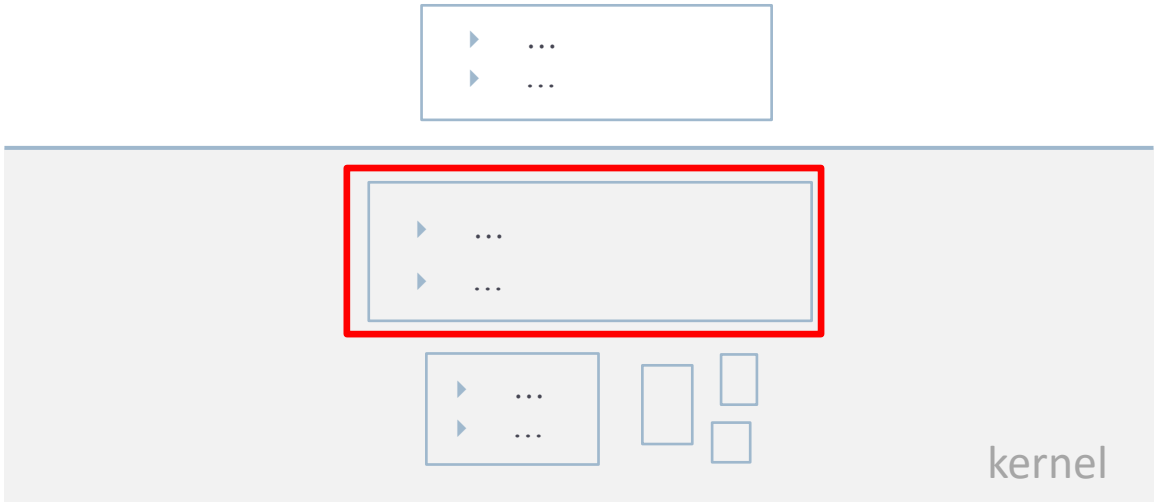
Understanding the Linux Kernel (O'Really)
http://www.eecs.harvard.edu/~cs161/assignments/sched.h.html

Servicios del sistema operativo
inicialización y finalización de procesos

Finalizar
- Razones
- API



Crear
- Razones
- API



Creación de procesos

- Un proceso se crea:
  - Durante el arranque del sistema
    - Hilos del kernel + primer proceso (Ej.: init, swapper, etc.)
  - Cuando un proceso existe hace una llamada al sistema para crear otro:
    - Cuando el sistema operativo comienza un nuevo trabajo
    - Cuando un usuario arranca un nuevo programa
    - Cuando durante la ejecución de un programa se necesite

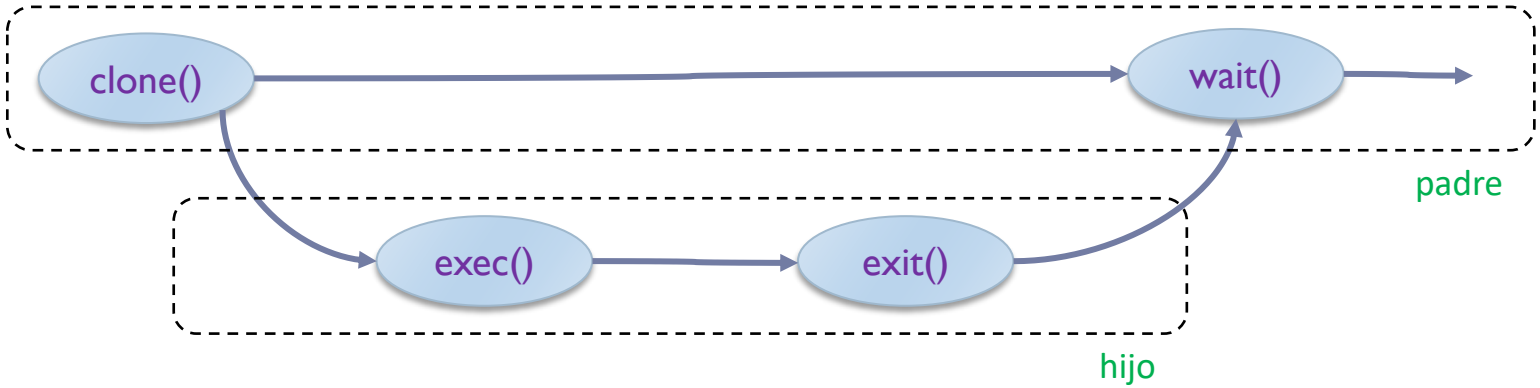
Finalización de procesos

- Un proceso termina:
  - De forma voluntaria:
    - Finalización normal
    - Finalización con error
  - De forma involuntaria:
    - Finalizado por el sistema (Ej.: excepción, sin recursos necesarios)
    - Finalizado por otro proceso (Ej.: a través de llamada al sistema)
    - Finalizado por el usuario (Ej.: control-c por teclado)

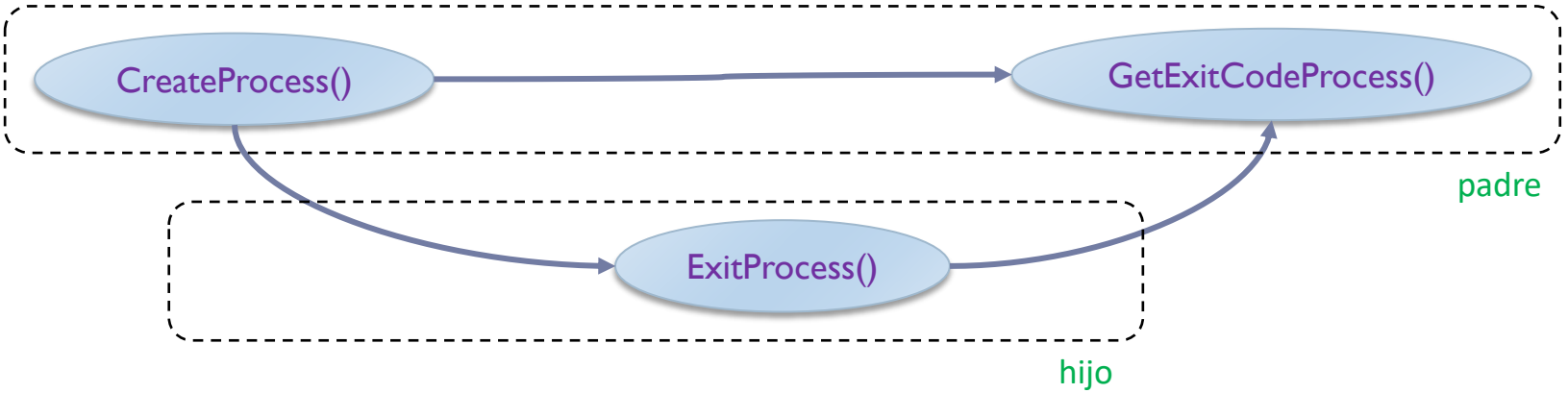
> - En Unix/Linux se usan señales como mecanismo
> - Se pueden capturar y tratar (salvo SIGKILL) para evitar finalizar involuntariamente

Creación y terminación de procesos
Llamadas al sistema

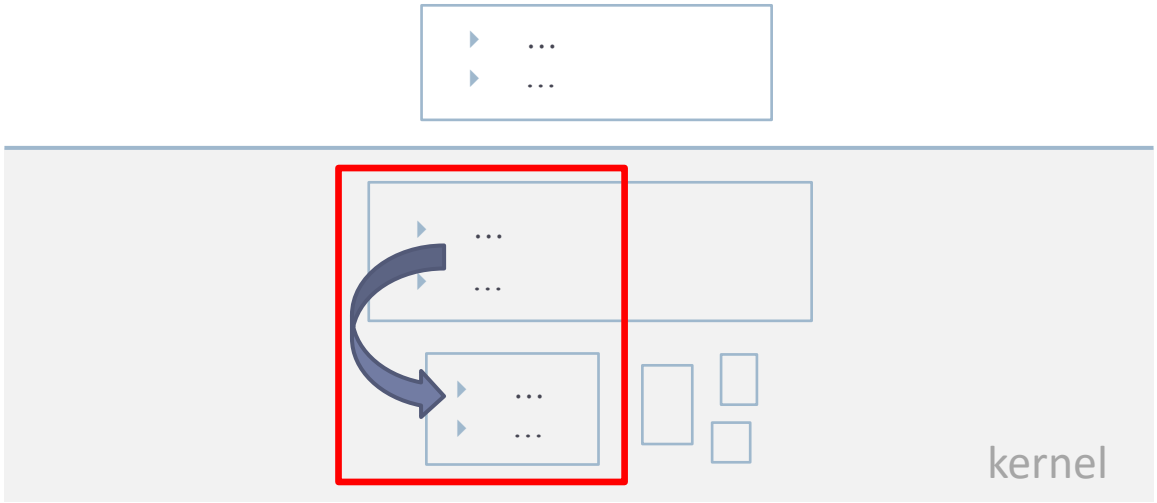
- Linux



- Windows



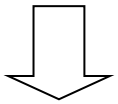
Servicios del sistema operativo
inicialización y finalización de procesos



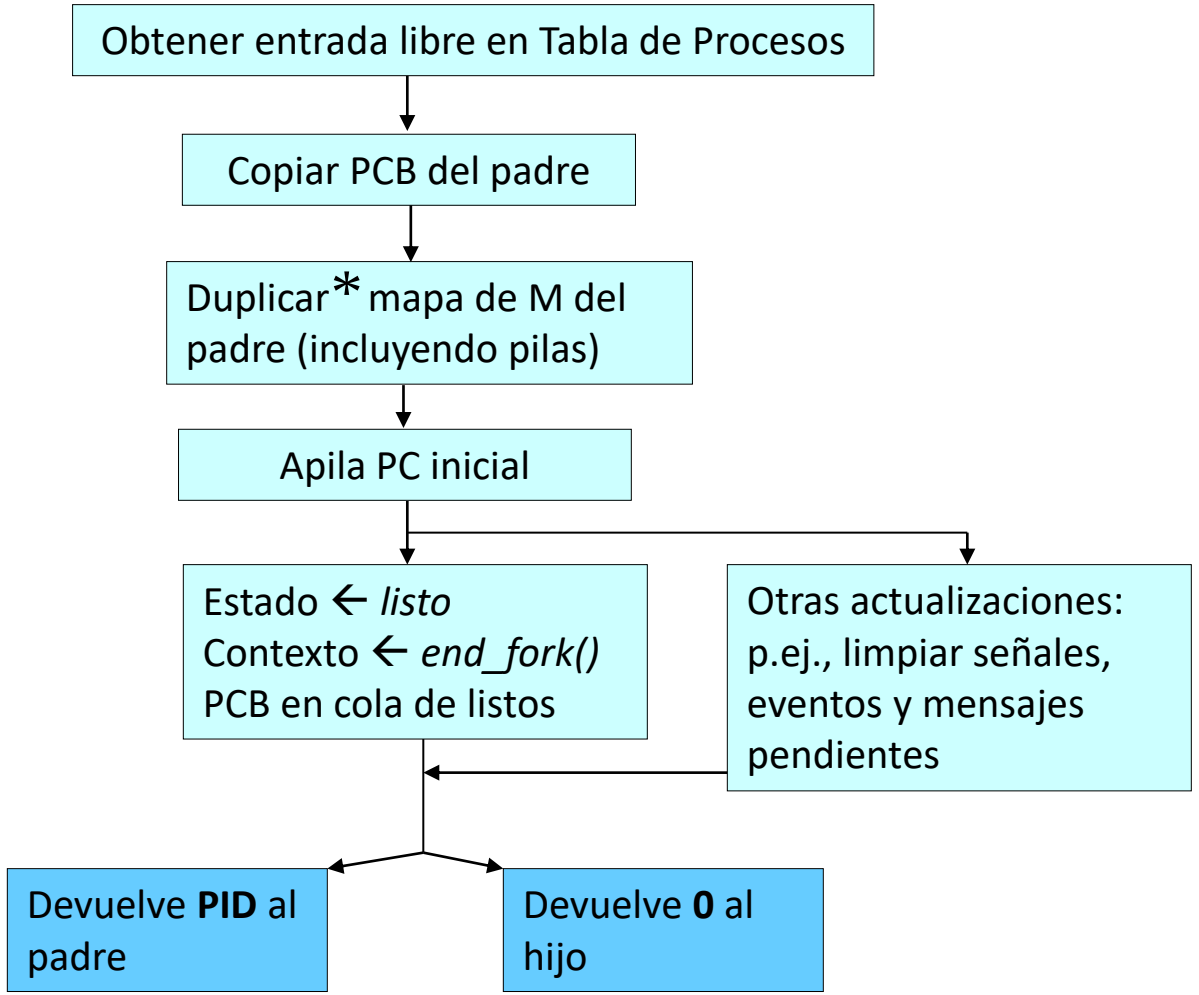
Creación de procesos

Linux: clone

clone:



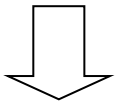
"Clona al proceso padre y da una nueva identidad al hijo"



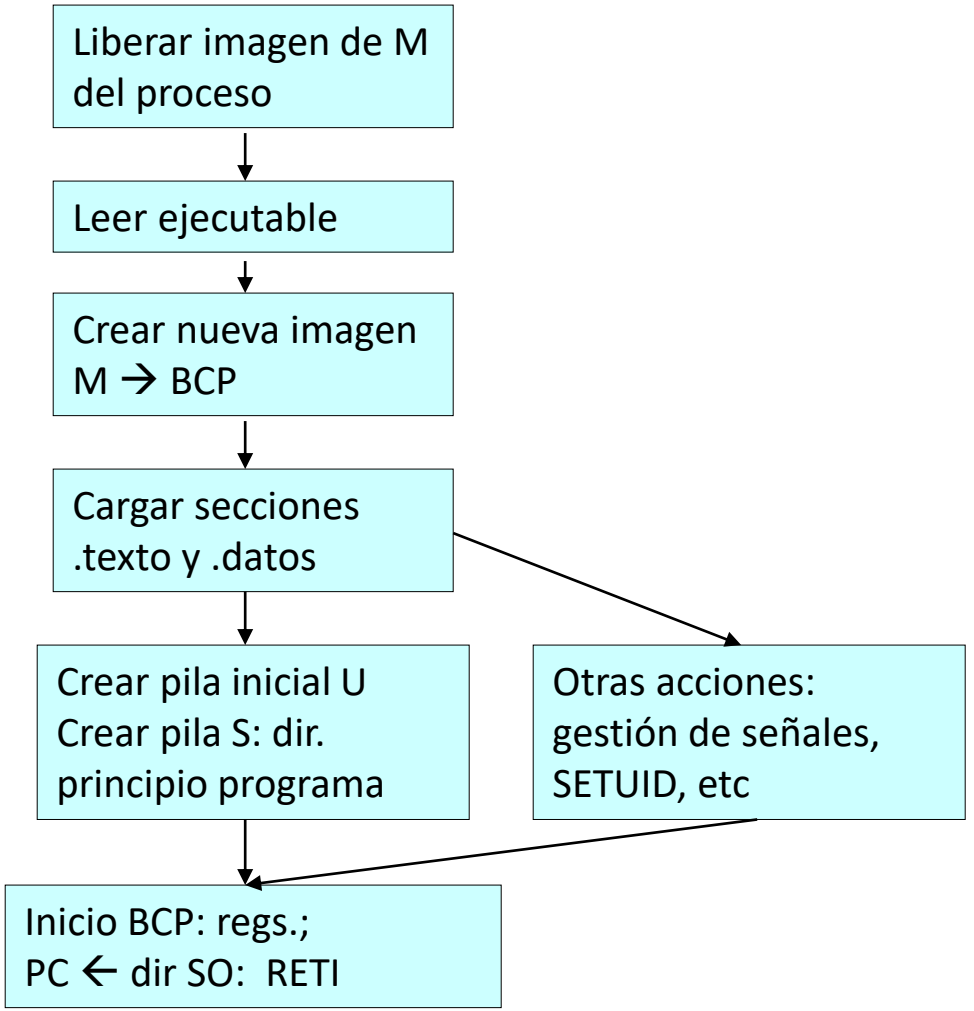
Cambio de imagen de un proceso

Linux: exec

exec:



"Cambia la imagen de memoria de un proceso usando como 'recipiente' uno previo"

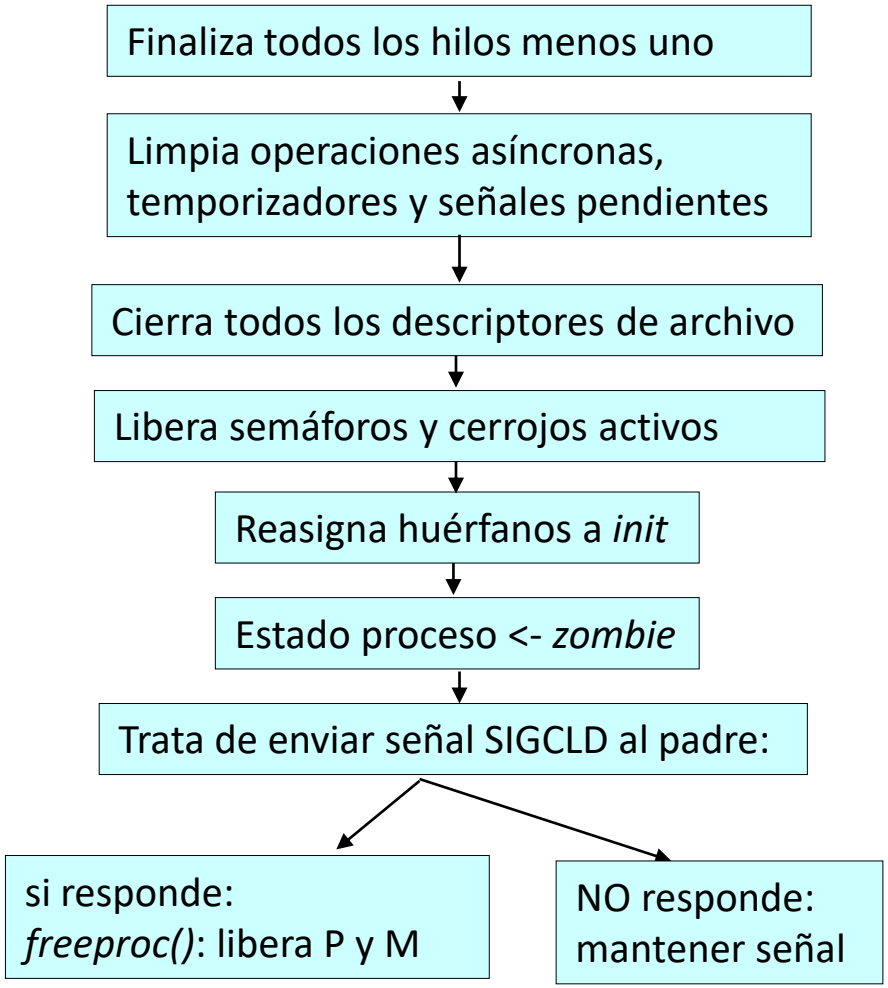


Terminación de procesos

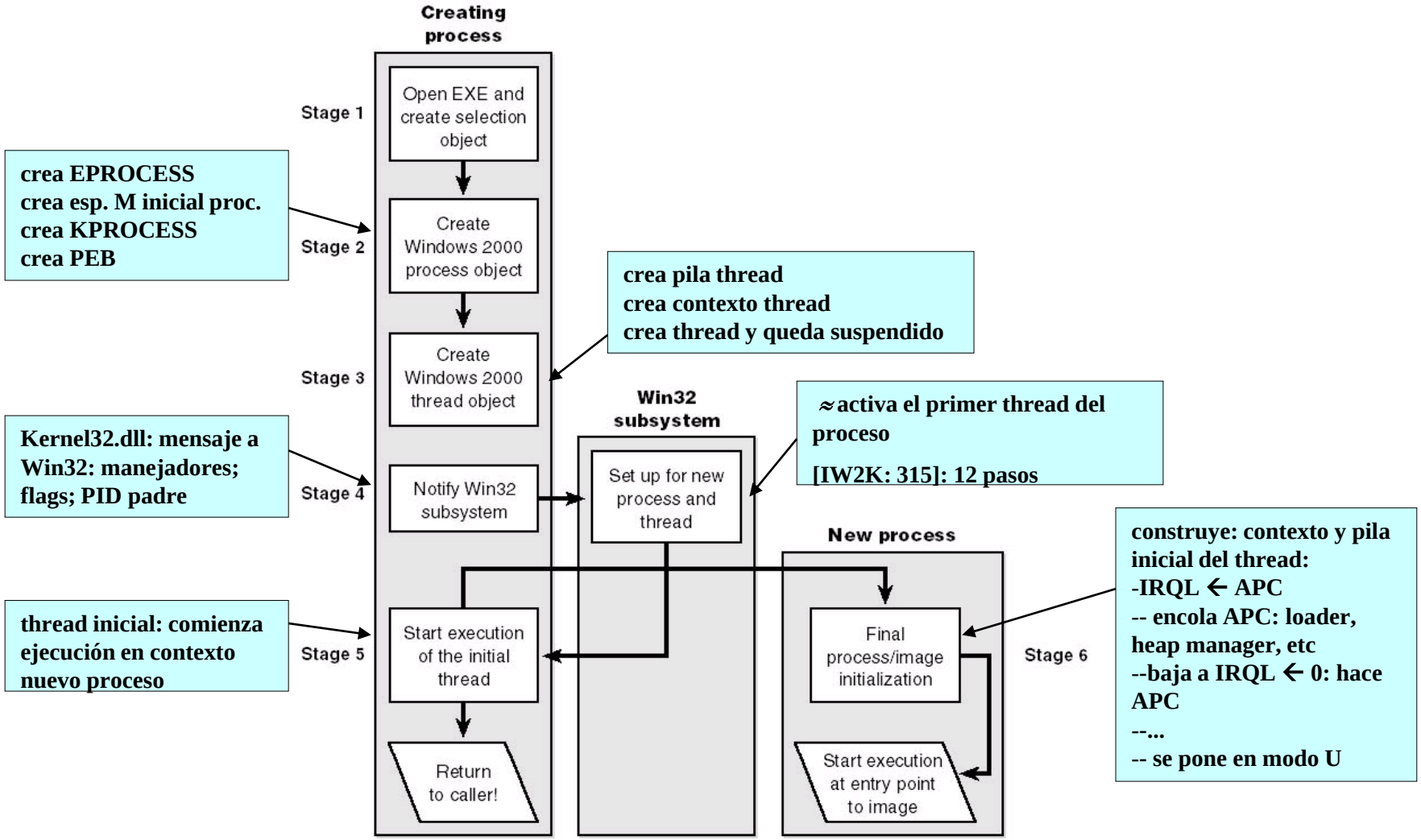
Linux: exit

exit:

"Termina la ejecución de un proceso y libera los recursos"

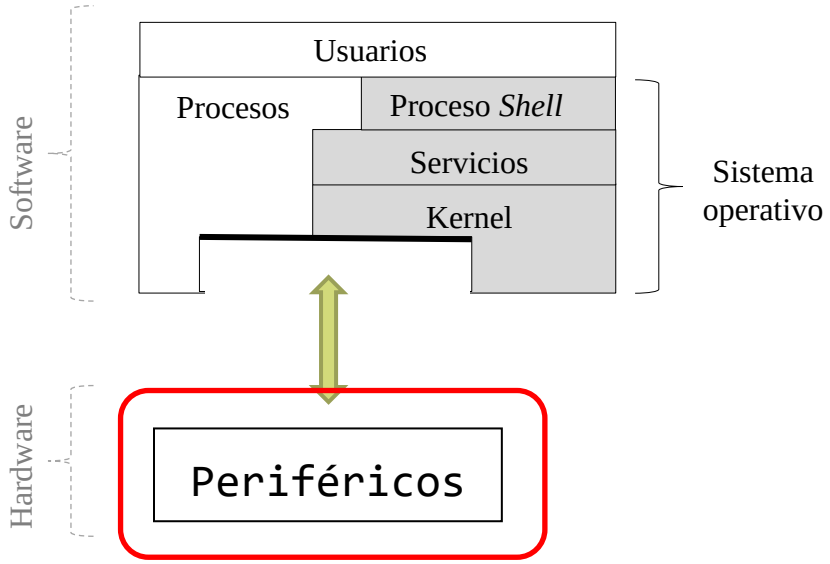


Creación de procesos
Windows: CreateProcess



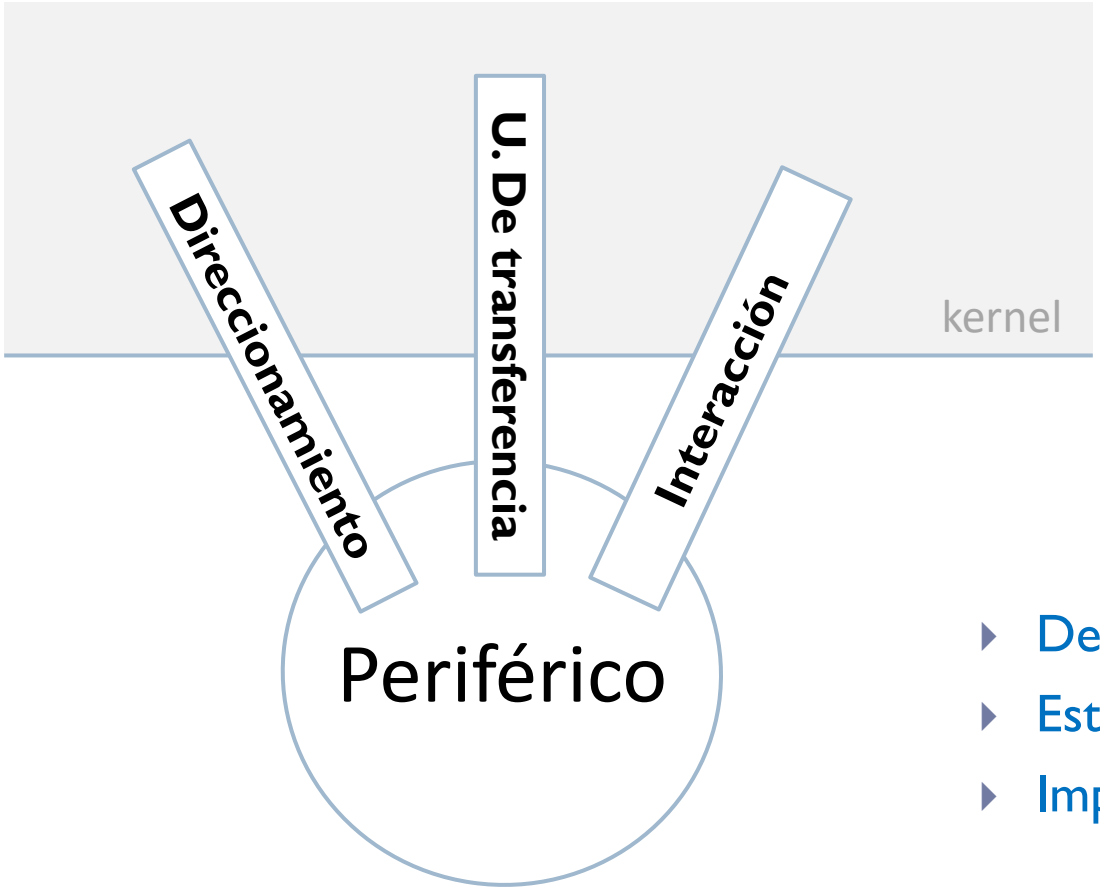
http://flylib.com/books/en/4.491.1.52/1/

Contenidos



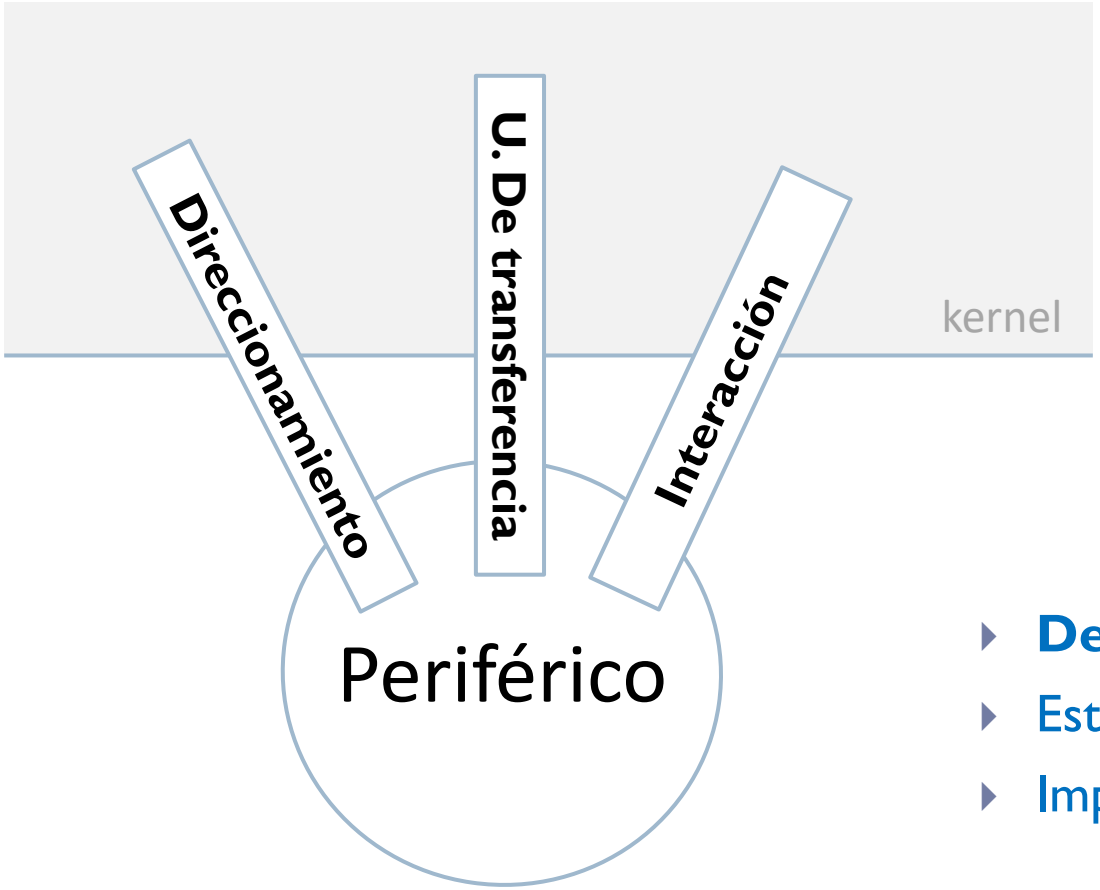
- Procesos
- **Periféricos**

Introducción



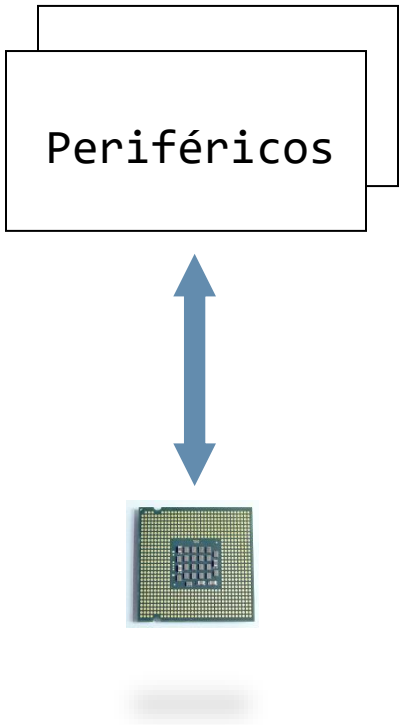
- Definición de periférico
- Estructura general
- Implicaciones en S.O.

Introducción



- **Definición de periférico**
- Estructura general
- Implicaciones en S.O.

Concepto de periférico

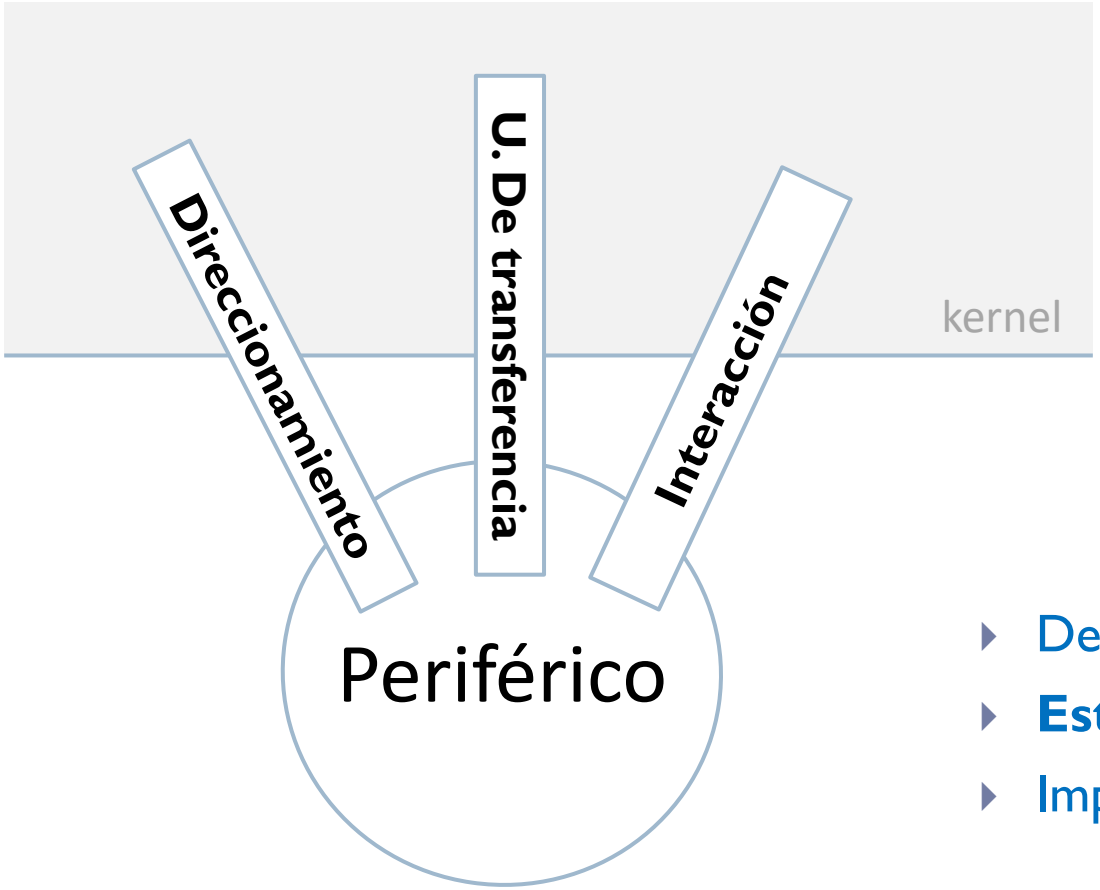


- **Periférico**:
  - Todo aquel dispositivo externo que se conecta a una CPU a través de la unidades o módulos de entrada/salida (E/S).
  - Permiten almacenar información o comunicar el computador con el mundo exterior.

Clasificación de periféricos (por uso)

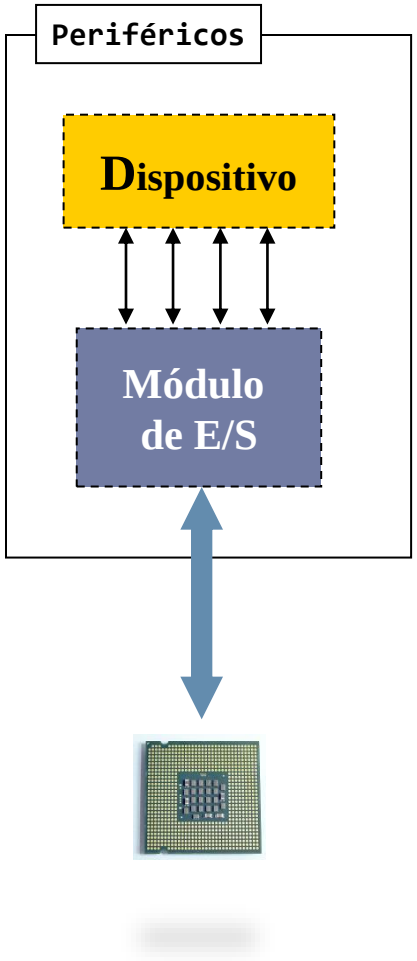
- **Comunicación**:
  - Hombre - máquina
    - (Terminal) teclado, ratón, …
    - (Impresa) plotter, escáner, …
  - Máquina - máquina (Módem, …)
  - Medio físico - máquina
    - (Lectura/accionamiento) x (analógico/digital)
- **Almacenamiento**:
  - Acceso "directo" (Discos, DVD, …)
  - Acceso secuencial (Cintas)

Introducción



- Definición de periférico
- **Estructura general**
- Implicaciones en S.O.

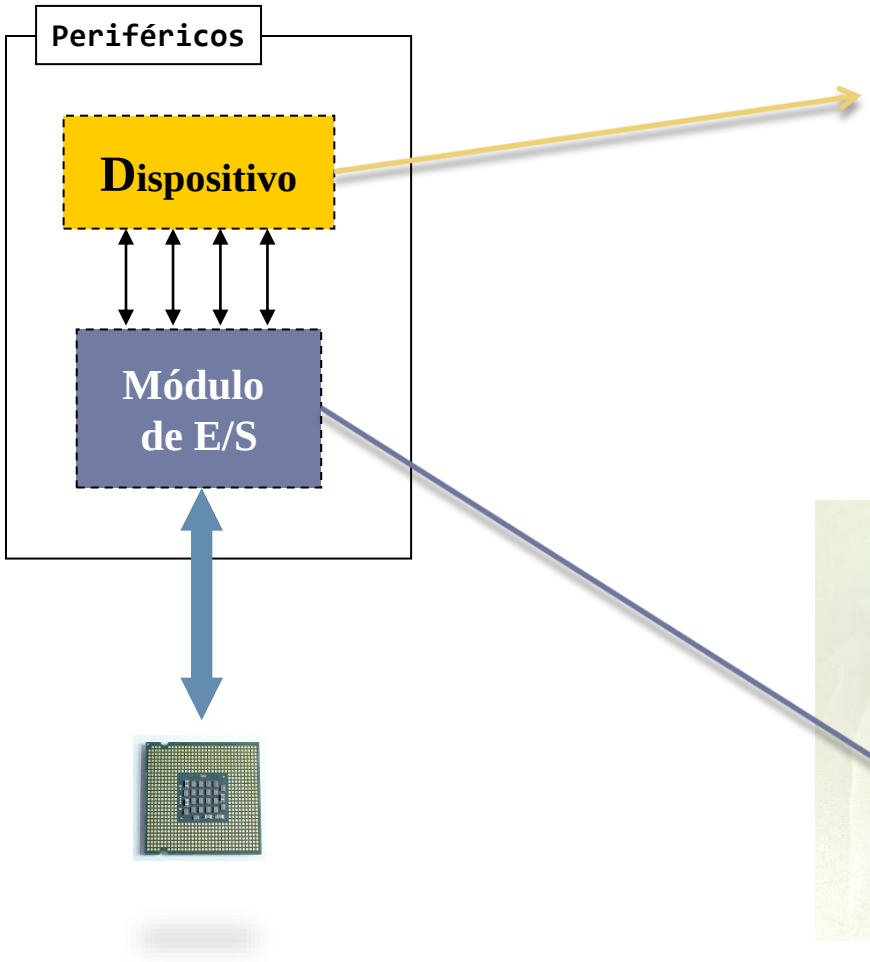
Estructura general de un periférico



- Compuesto de:
  - Dispositivo
    - Hardware que interactúa con el entorno
  - Módulo de Entrada/Salida
    - También denominado controlador
    - Interfaz entre dispositivo y la CPU, que le oculta las particularidades de este

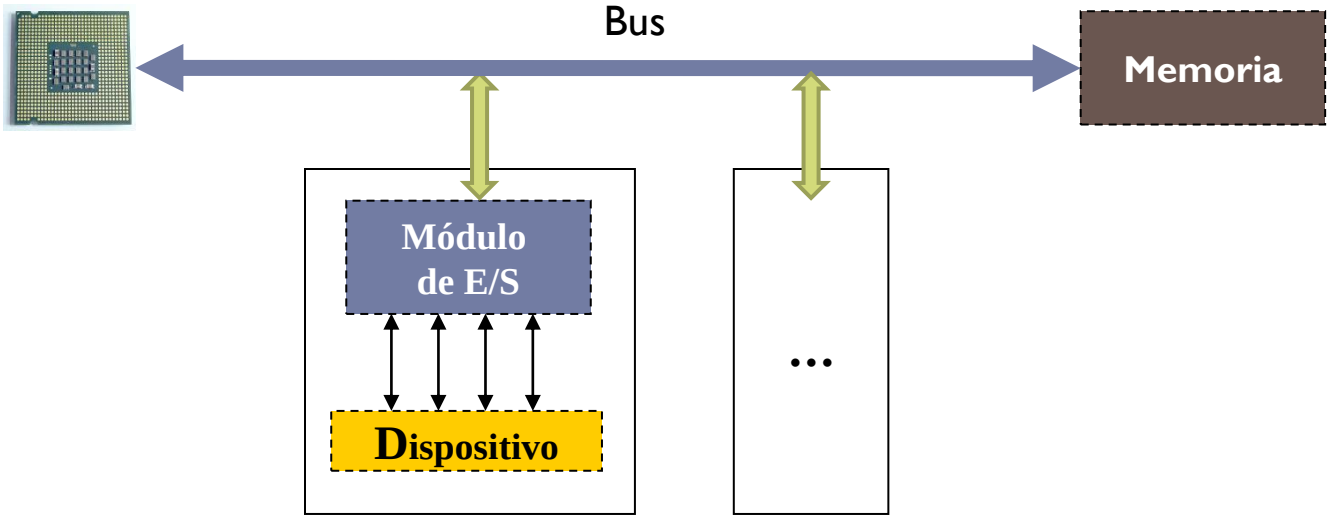
Periférico = Dispositivo + Controlador

Ejemplo
Disco duro

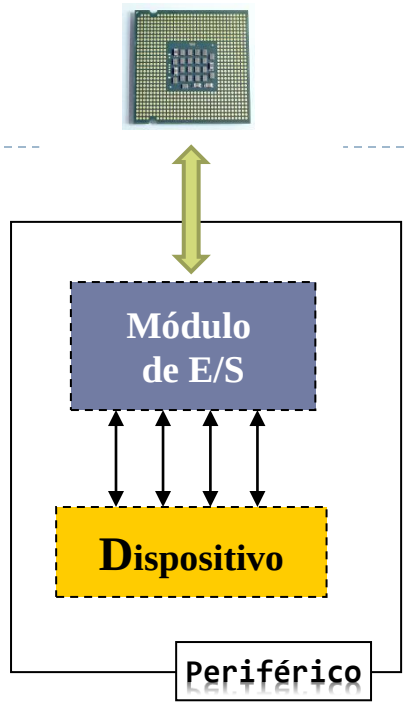


Módulo de E/S
qué son

- Las **unidades** o **módulos de E/S** realizan la conexión de la CPU con los dispositivos periféricos.



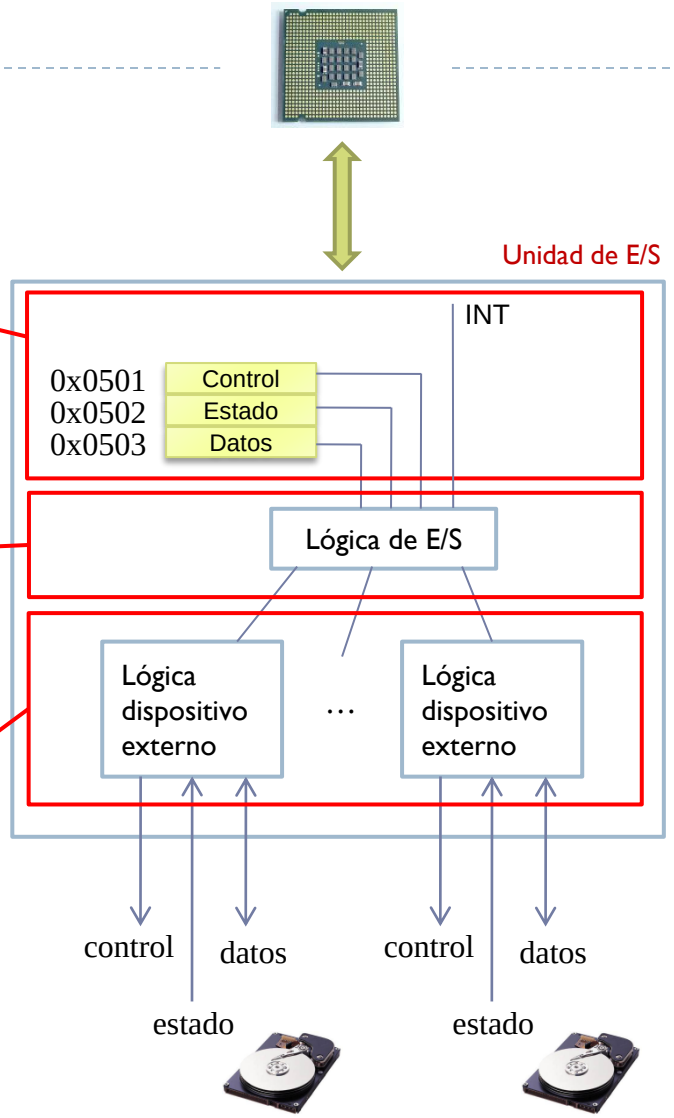
Módulo de E/S
necesidad



- Son necesarios debido a:
  - Gran variedad de periféricos.
    - Los periféricos son 'raros'
  - La velocidad de transferencia de datos de los periféricos es mucho menor que la de la memoria o el procesador.
    - Los periféricos son 'muy lentos'
  - Formatos y tamaños de palabra de los periféricos distintos a los del computador al que se conectan.

Módulo de E/S
estructura

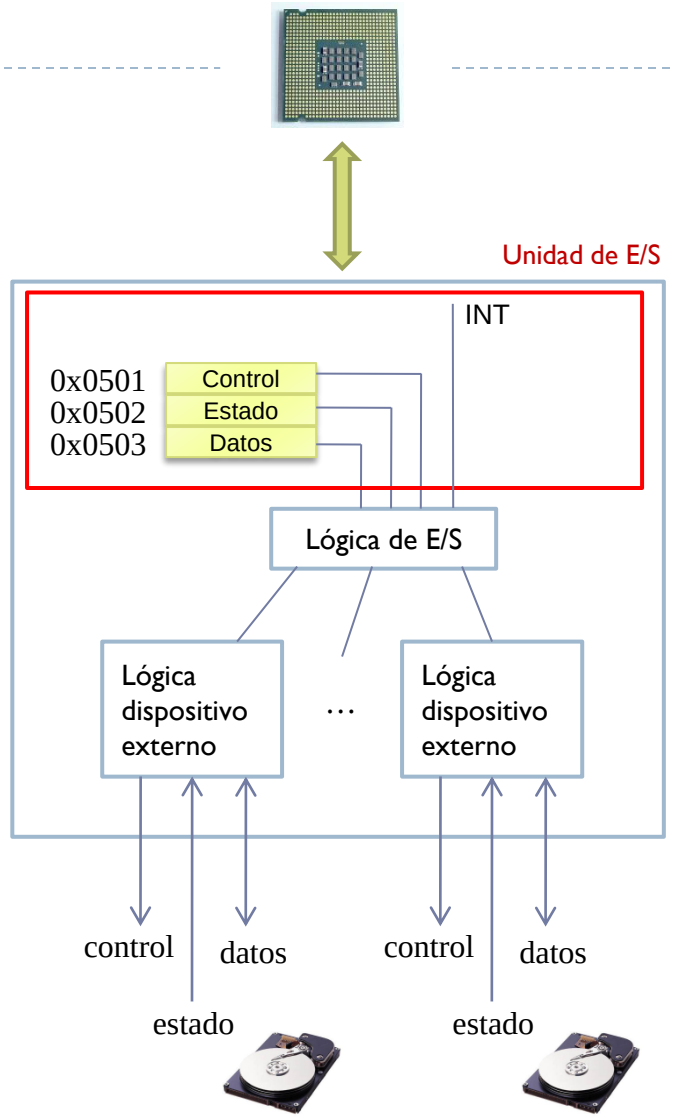
- **Interfaz**
  Ofrece la interacción entre CPU y unidad de E/S
- **Lógica general de interacción**
  Controla el protocolo de interacción entre CPU y unidad de E/S
- **Lógica específica de dispositivo**
  Gestión específica de dispositivo



Módulo de E/S
estructura: interfaz

Interacción entre CPU y Unidad de E/S a través de:

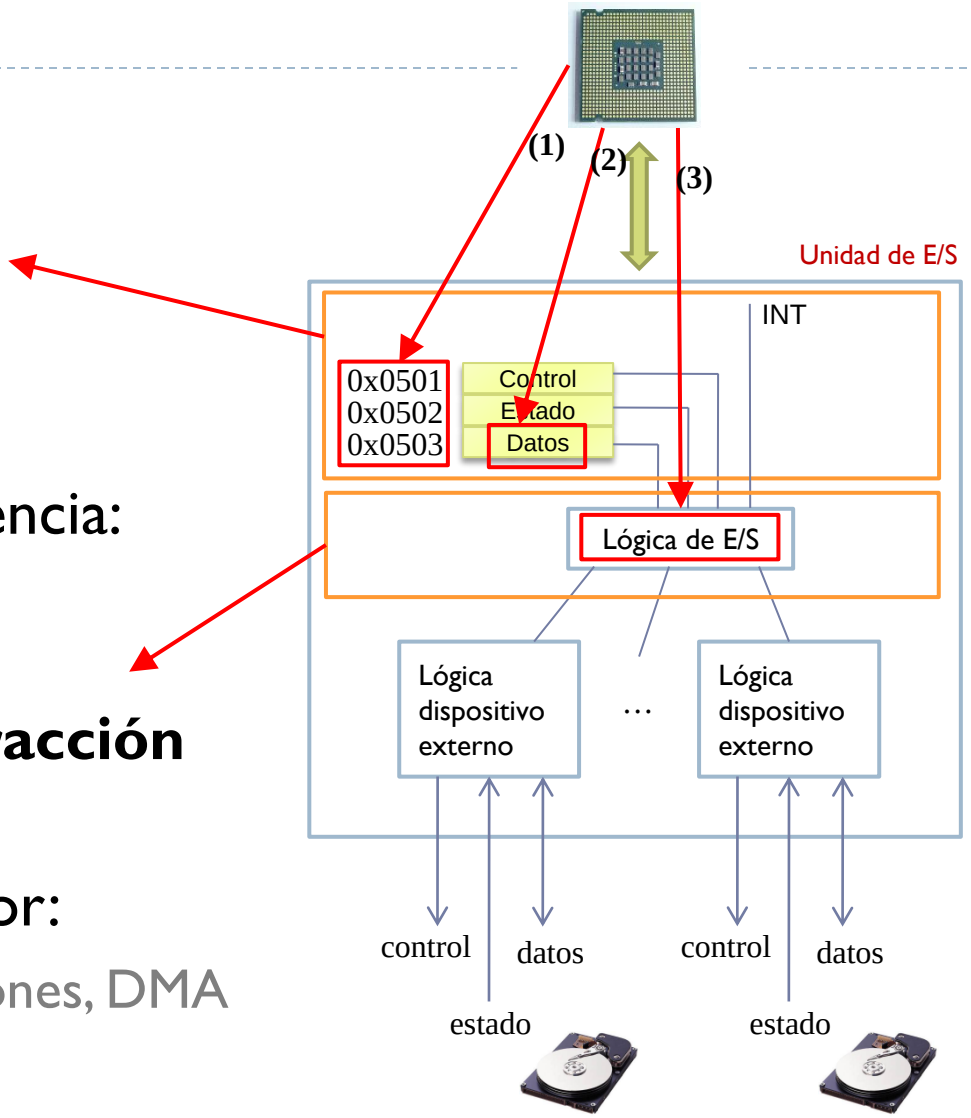
- 3 **tipos** de registros:
  - Registro de control
    - Ordenes para el periférico
  - Registro de estado
    - Estado desde de la última orden
  - Registro de datos
    - Datos intercambiados CPU/periférico
- 1 **tipo** de línea de interrupción:
  - Interrupción de aviso



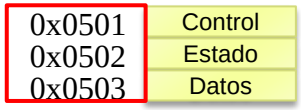
Módulo de E/S

aspectos a conocer

- **Interfaz**
  - (1) Direccionamiento:
    - Conjunto, separado
  - (2) Unidad de transferencia:
    - Carácter, bloque
- **Lógica general de interacción**
  - (3) Interacción computador-controlador:
    - Programada, Interrupciones, DMA

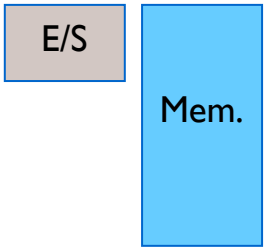


(1/3) Direccionamiento de E/S



- **Espacio de memoria conjunto**
  - Los registros del 'controlador' se proyectan en memoria y usando un conjunto de direcciones de memoria se acceden a dichos registros.
  - Ej: `int * rctrl = 0x105A ;`
    `(*rctl) = 1 ;`

- **Espacio de memoria separado (puertos)**
  - Con instrucciones ensamblador especiales (In/Out) se acceden a las direcciones de E/S (denominadas puertos) que representan los registros del 'controlador'.
  - Ej: `out(0x105A, 1) ;`



(2/3) Unidad de transferencia

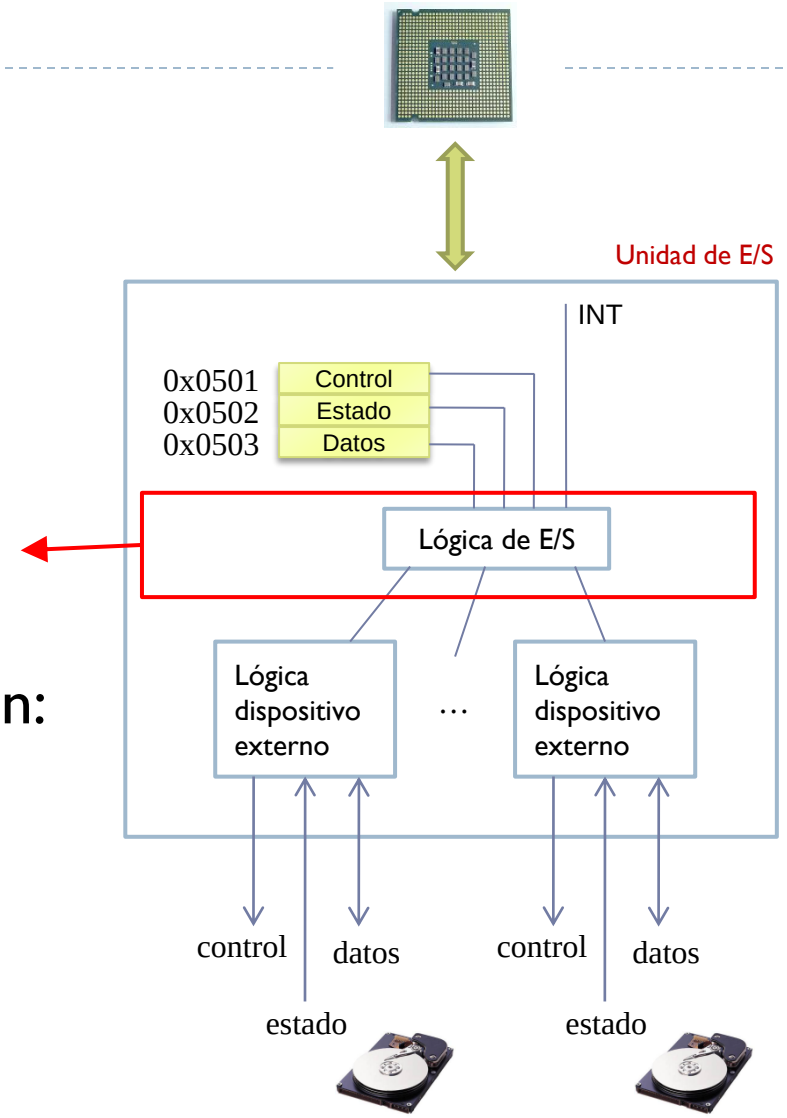
- **Dispositivos de bloque**:
  - Unidad: bloque de bytes
  - Acceso: secuencial o directo
  - Operaciones: leer, escribir, situarse, …
  - Ejemplos: "cintas" y discos

- **Dispositivos de carácter**:
  - Unidad: caracteres (ASCII, Unicode, etc.)
  - Acceso: secuencial
  - Operaciones: `get`, `put`, ….
  - Ejemplo: terminales, impresoras, etc.

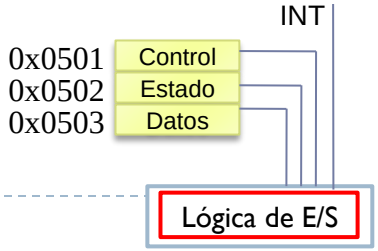
Módulo de E/S
funcionamiento

Interacción computador-controlador definida:

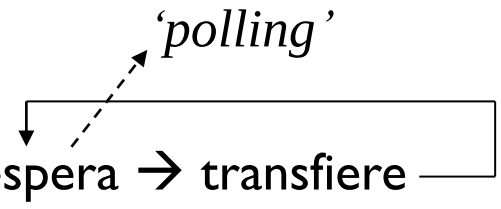
- Protocolo de interacción entre CPU y unidad de E/S
- 3 **tipos** principales de interacción:
  - **Programada**
  - **Interrupciones**
  - **DMA**



(3/3) Interacción con computador



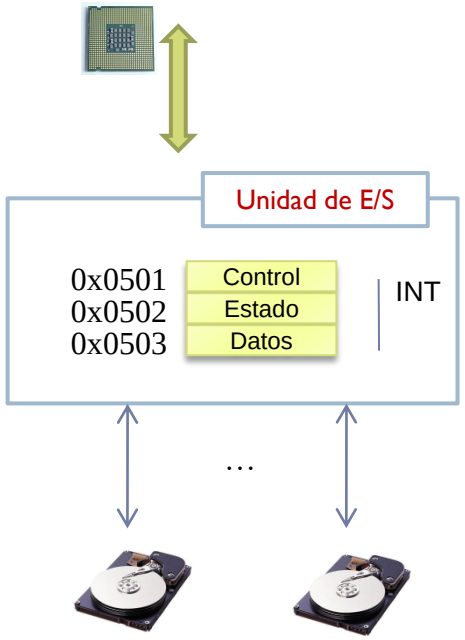
- **E/S programada o directa**
  - CPU no hace otra cosa que E/S: espera → transfiere



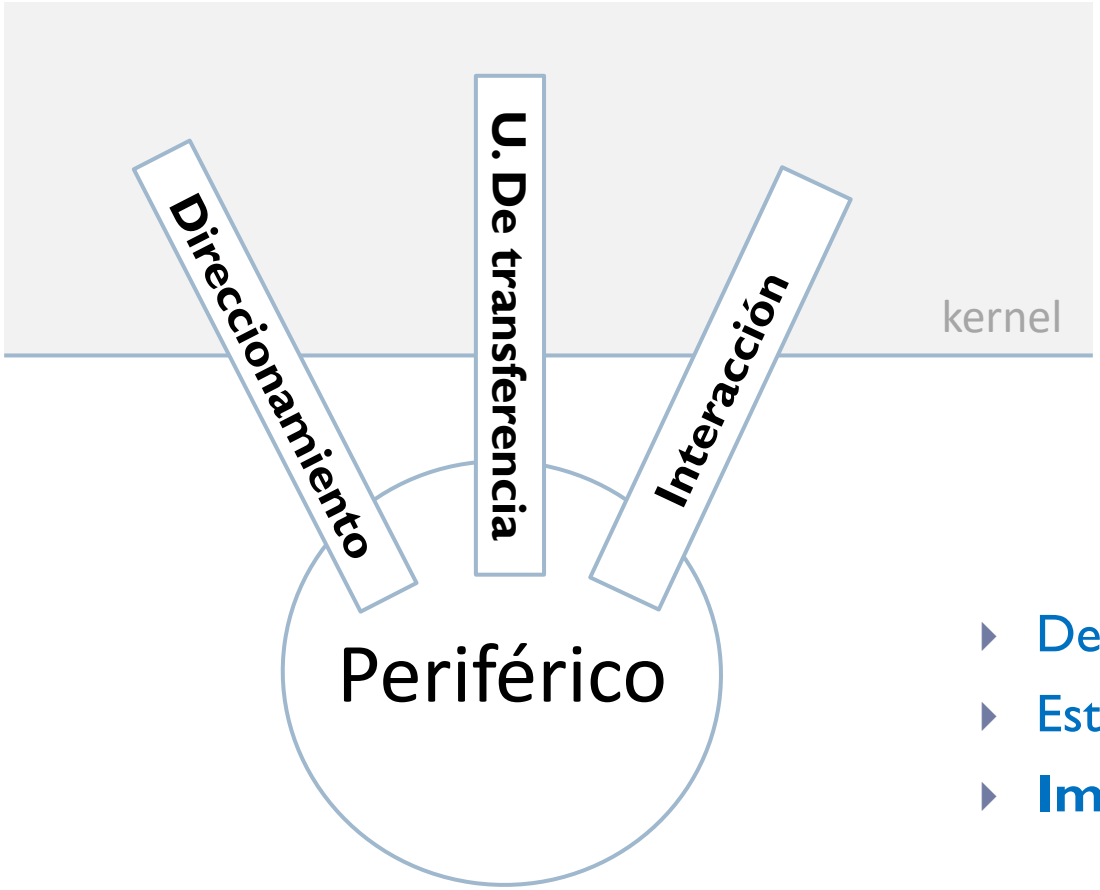
- **E/S por interrupciones**
  - CPU no espera, sólo transfiere
- **E/S por DMA (acceso directo a memoria)**
  - CPU no espera, no transfiere, le avisan del fin de los datos transferidos
    - Controlador de periférico más sofisticado (más costoso, mejor rendimiento)
    - Busca reducir la sobrecarga al transferir bloques de información

Módulo de E/S
resumen de las características fundamentales

- Unidad de transferencia
  - De bloques
  - De caracteres
- Direccionamiento
  - Proyectados en memoria
  - Mediante puertos
- Interacción computador-controlador
  - E/S programada o directa
  - E/S por interrupciones
  - E/S por DMA



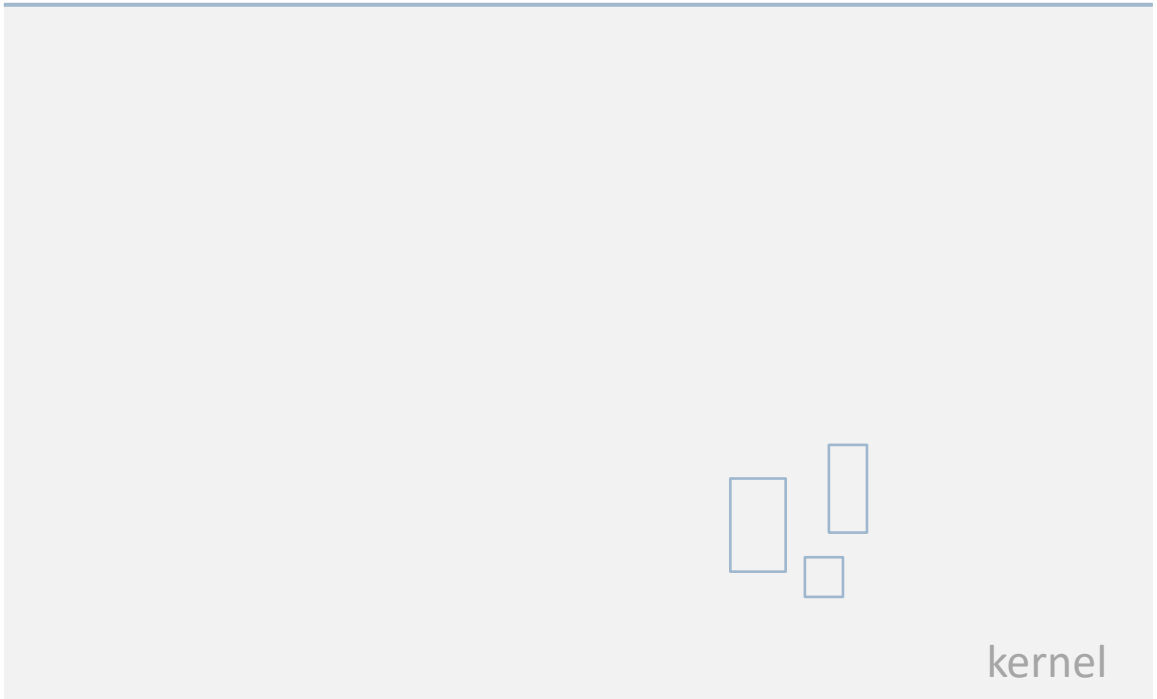
Introducción



- Definición de periférico
- Estructura general
- **Implicaciones en S.O.**

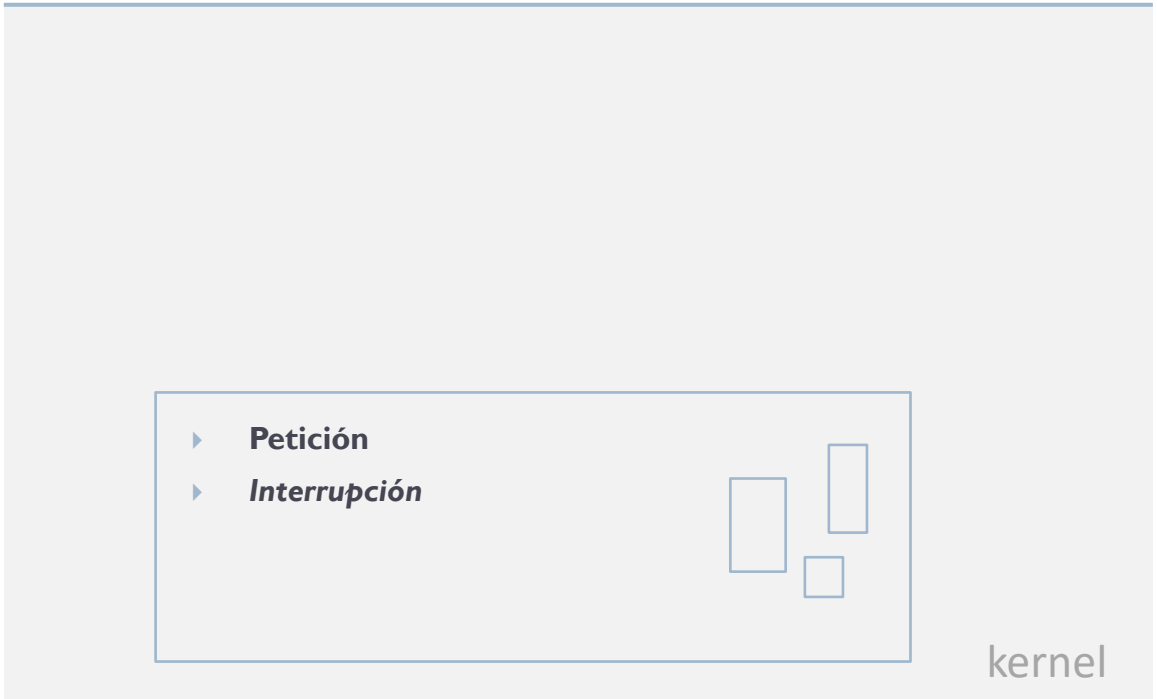
Implicaciones en el sistema operativo

1. Estructuras de datos



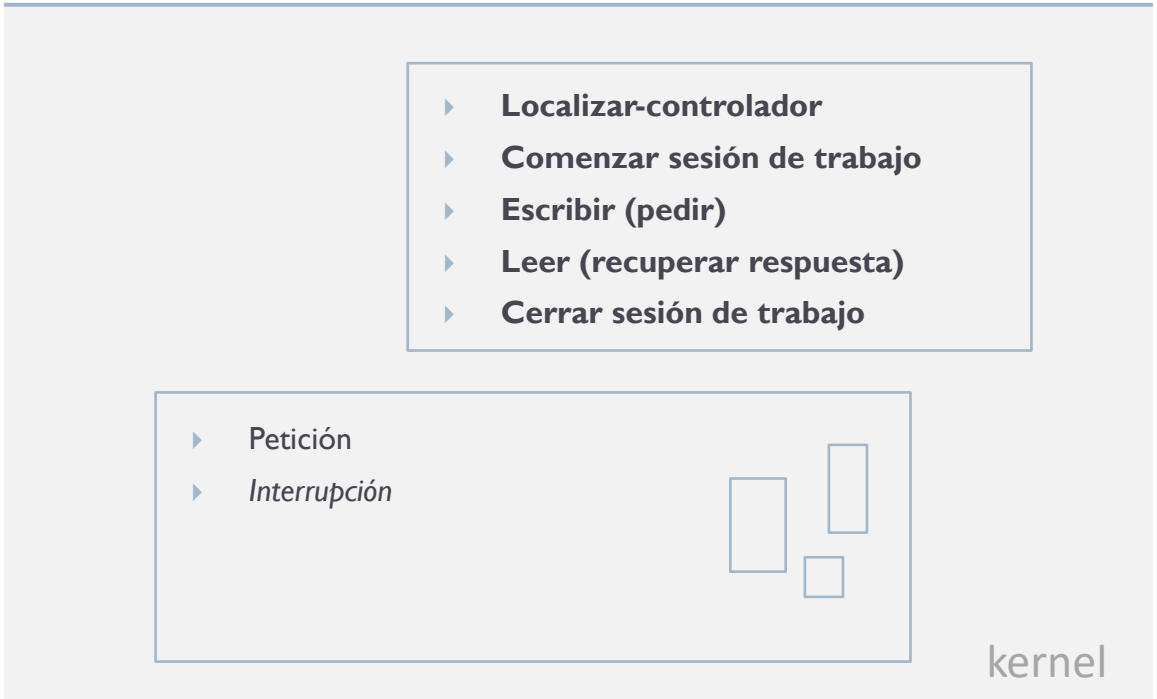
Implicaciones en el sistema operativo

2. Funciones: de gestión internas



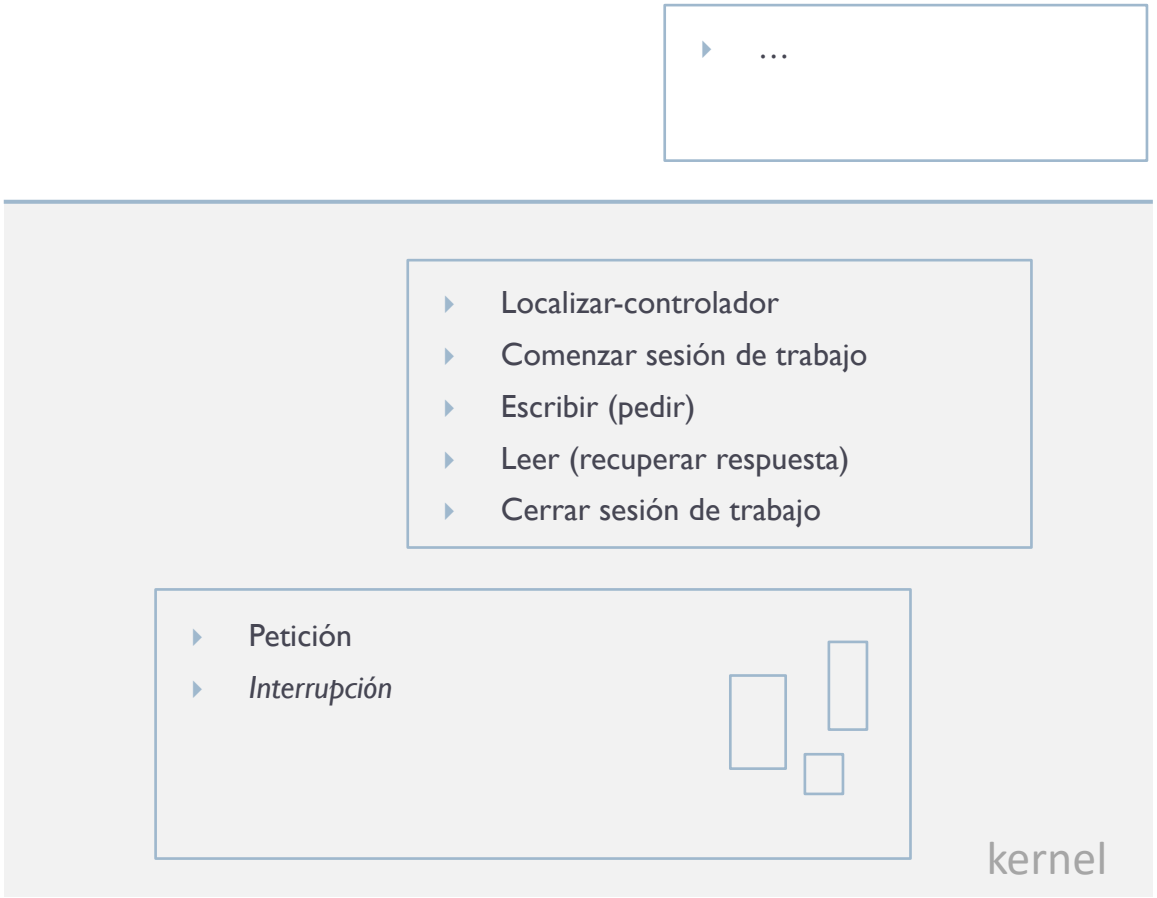
Implicaciones en el sistema operativo

3. Funciones: de servicio



Implicaciones en el sistema operativo

3. Funciones: API de servicio



Implicaciones en el sistema operativo

(1 y 2) Estructuras de datos + funciones de gestión internas
= controlador (driver)

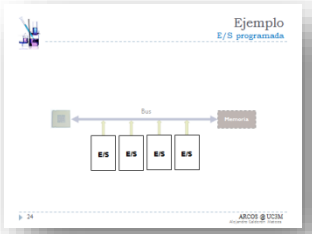
Implicaciones en el sistema operativo

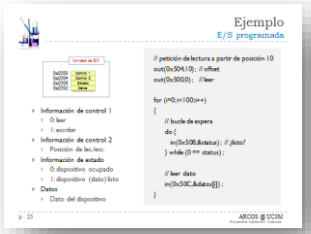
(1 y 2) Estructuras de datos + funciones de gestión internas
= controlador (driver)

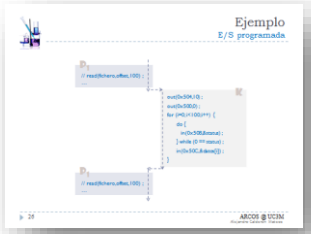
El controlador implementa la interacción computador-controlador

Impacto en el sistema operativo en el tratamiento de dispositivos

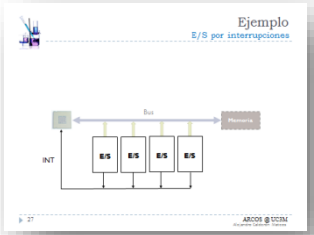
- **E/S programada o directa**

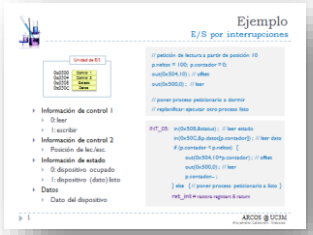


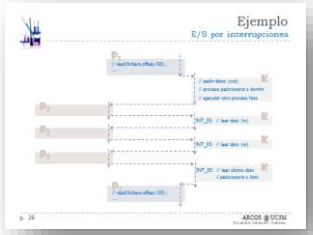




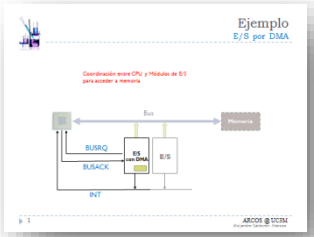
- **E/S por interrupciones**

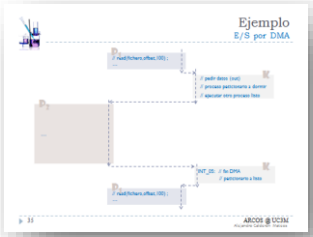




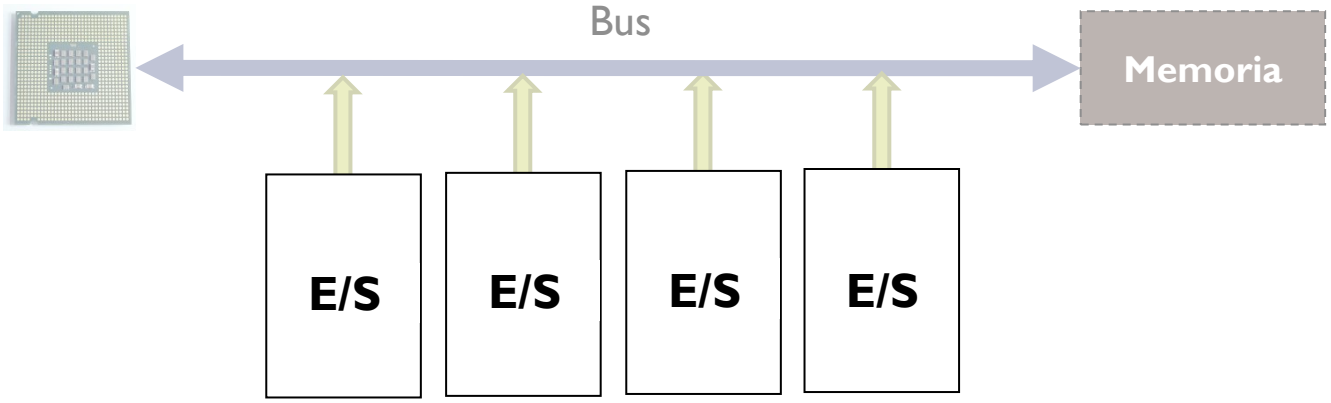


- **E/S por DMA**

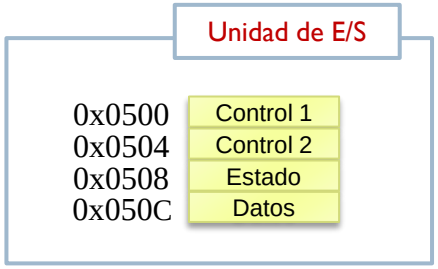




Ejemplo
E/S programada



Ejemplo
E/S programada

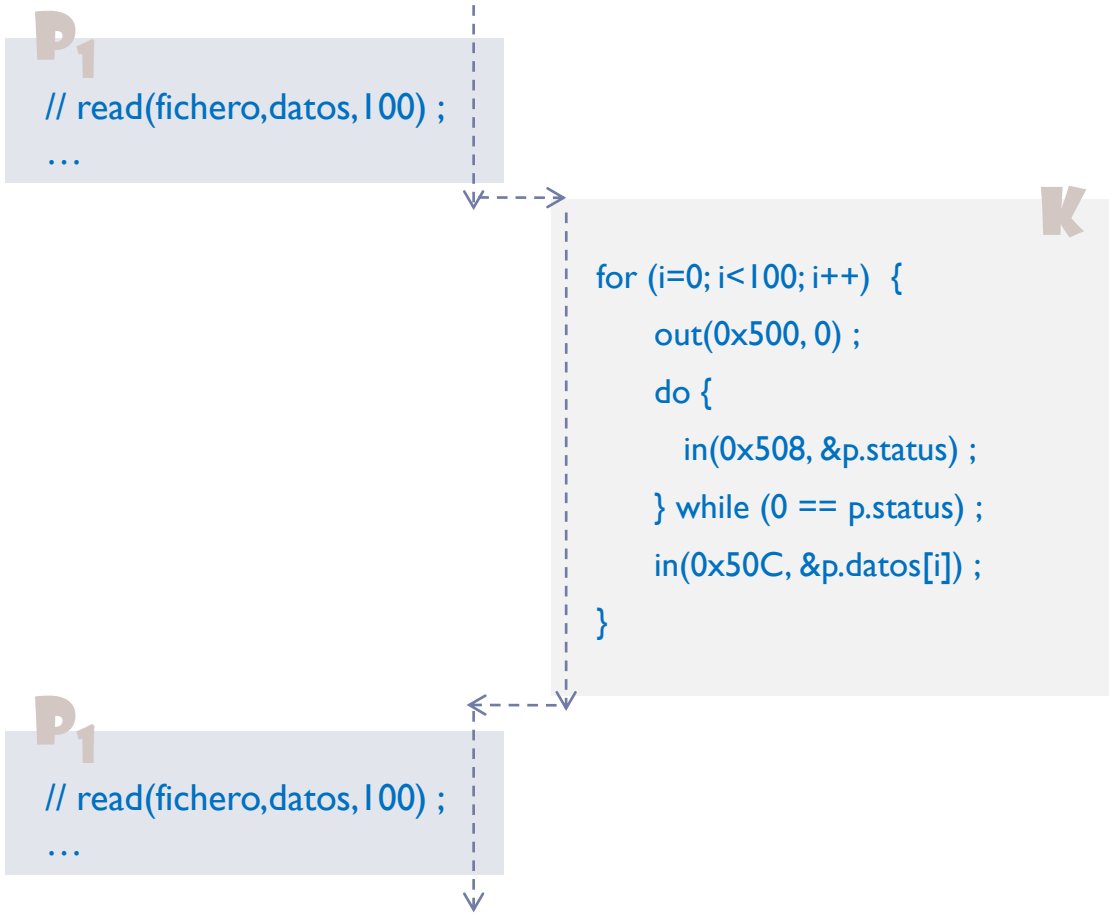


- Información de control I
  - 0: leer
  - : escribir
- Información de estado
  - 0: dispositivo ocupado
  - 1: dispositivo (dato) listo
- Datos
  - Dato del dispositivo

petición:

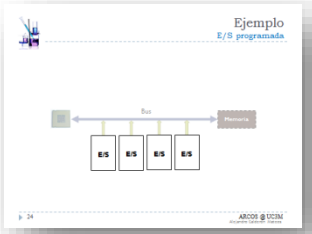
```
for (i=0; i<100 ;i++)
{
      // pedir siguiente lectura
      out(0x500, 0) ;

      // bucle de espera
      do {
         in(0x508, &(p.status)) ;  // ¿listo?
      } while (0 == p.status)) ;

      // leer dato
      in(0x50C, &(p.datos[i])) ;
}
```

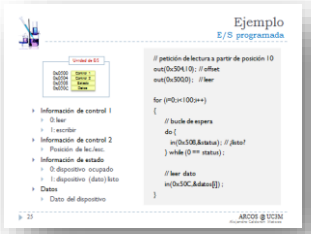
Ejemplo
E/S programada

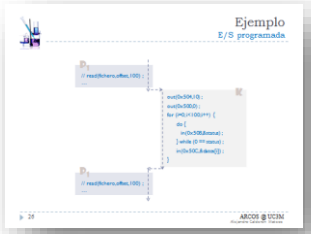


Impacto en el sistema operativo en el tratamiento de dispositivos

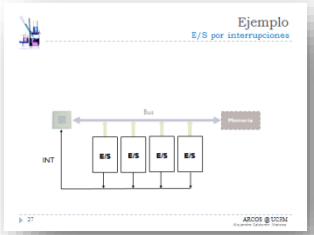
- **E/S programada o directa**

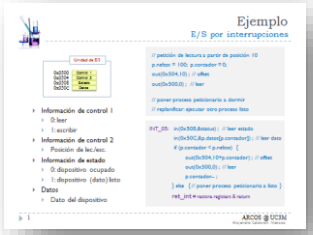


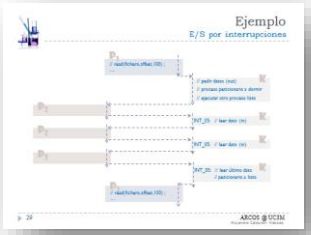




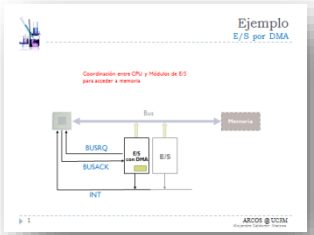
- **E/S por interrupciones**

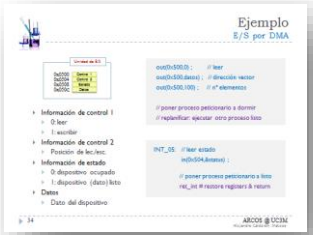


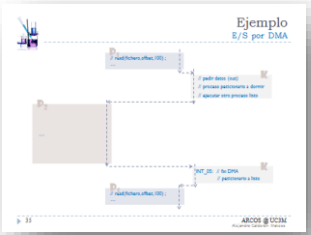




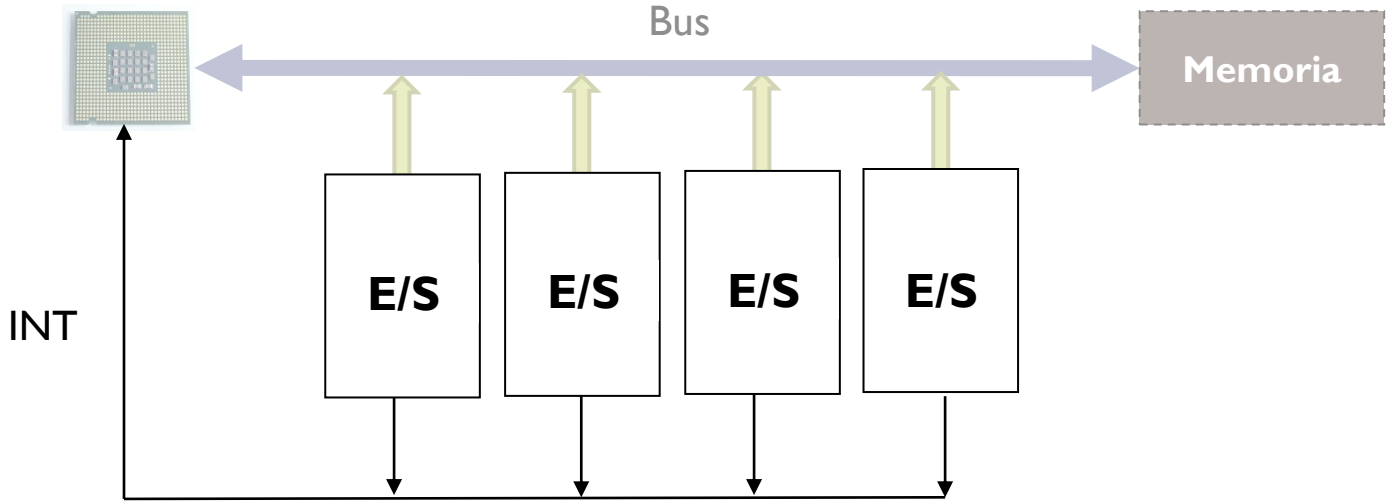
- **E/S por DMA**



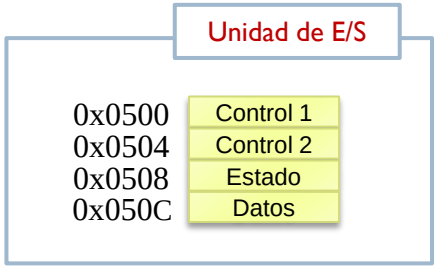




Ejemplo
E/S por interrupciones



Ejemplo
E/S por interrupciones

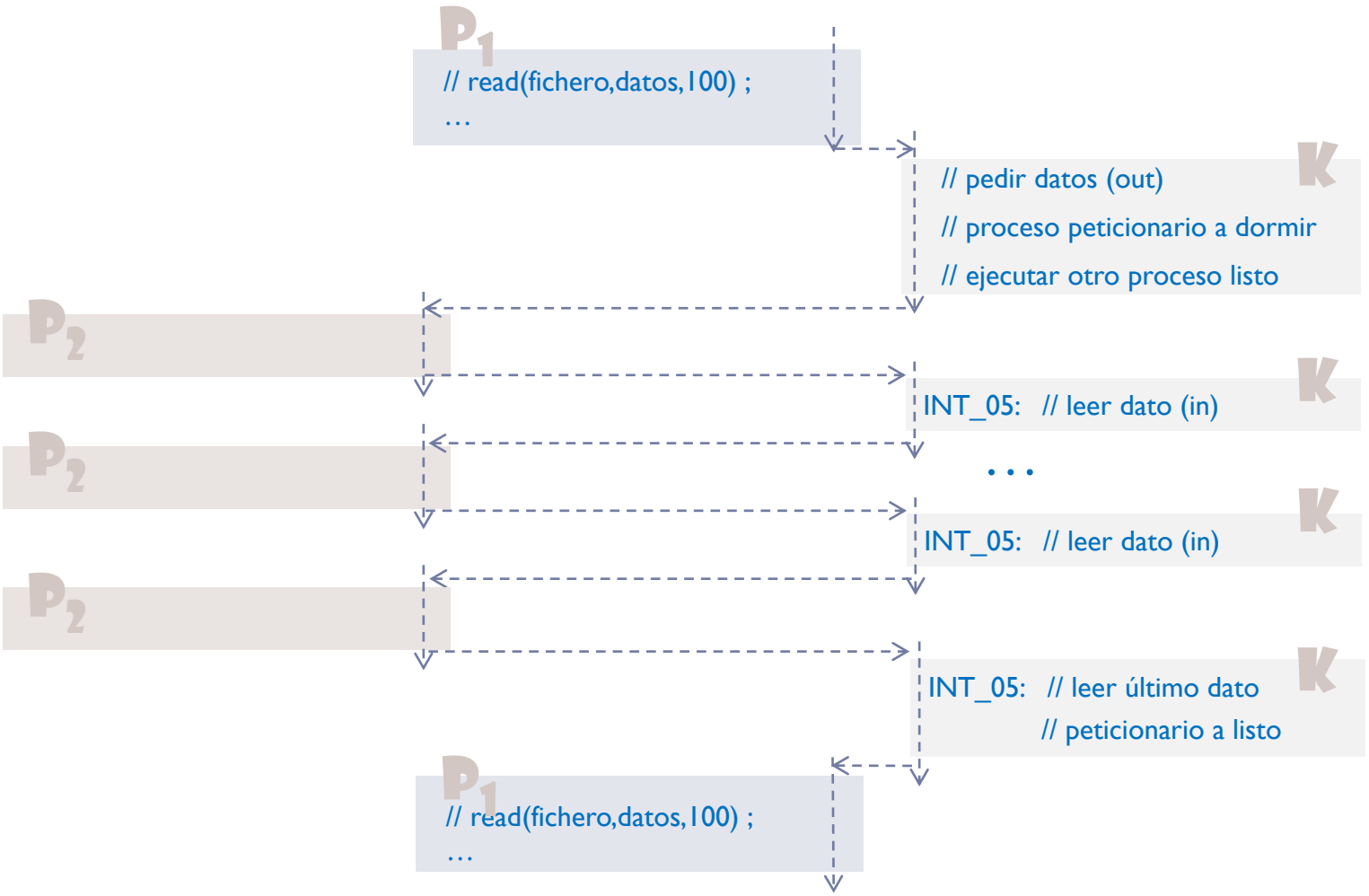


- Información de control 1
  - 0: leer
  - 1: escribir
- Información de estado
  - 0: dispositivo ocupado
  - 1: dispositivo (dato) listo
- Datos
  - Dato del dispositivo

```
petición:
        // pedir lectura
        p.contador = 0;
        p.neltos = 100;
        out(0x500,0) ;   // leer

      // cambio de contexto voluntario (C.C.V.)
        …
```

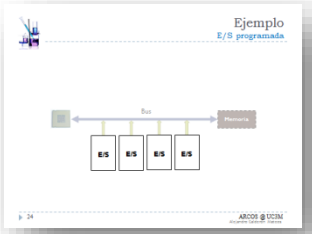
```
INT_05:  in(0x508,  &(p.status)) ;   // leer estado
         in(0x50C, &(p.datos[p.contador])); // leer dato
         if ((p.contador<p.neltos) && (p.status==OK))
         {
               out(0x500,0) ;  // leer
               p.contador++ ;
         } else  { // poner proceso peticionario a listo }
        ret_int # restore registers & return
```

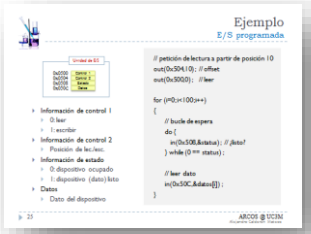
Ejemplo
E/S por interrupciones

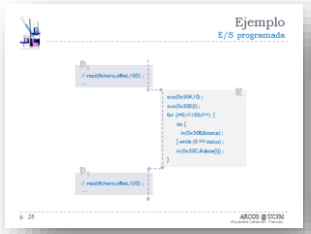


Impacto en el sistema operativo en el tratamiento de dispositivos

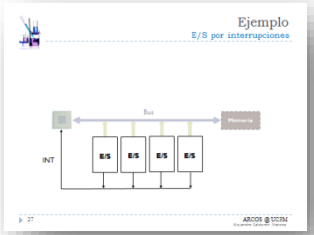
- **E/S programada o directa**

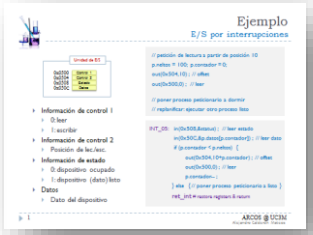


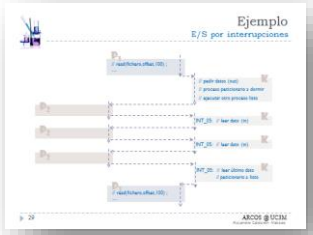




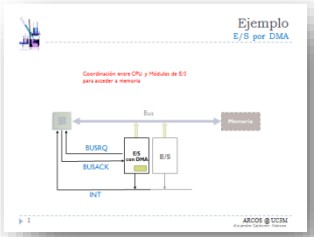
- **E/S por interrupciones**

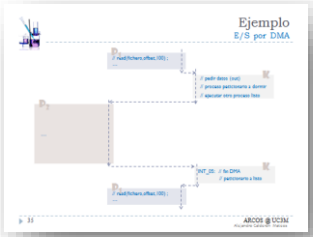






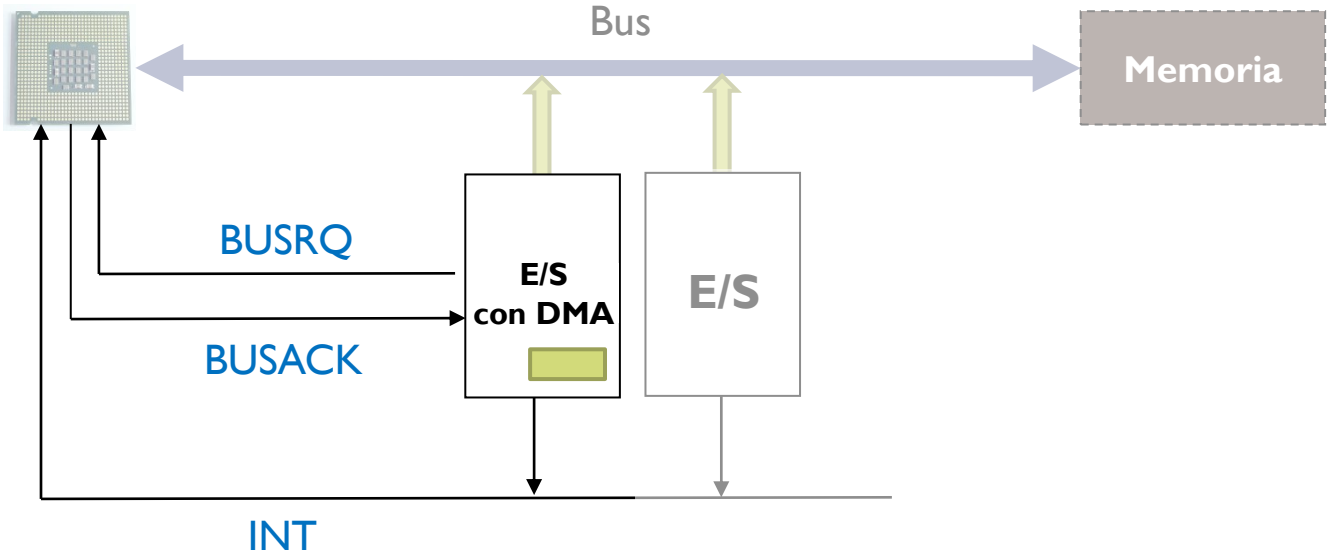
- **E/S por DMA**





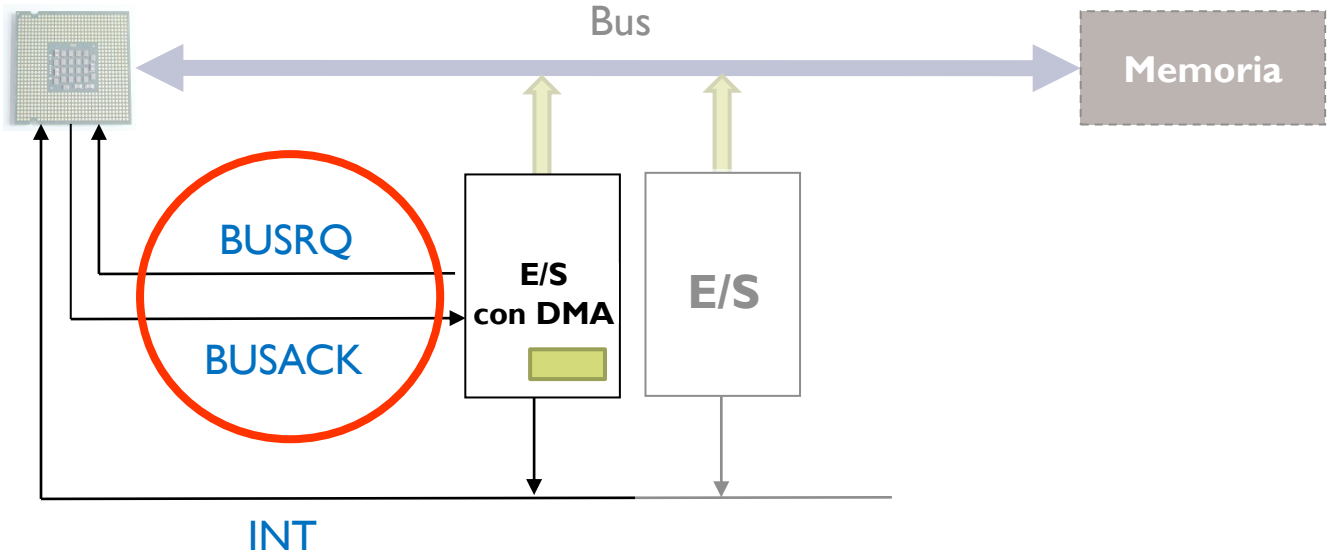
Ejemplo
E/S por DMA

Coordinación entre CPU y Módulos de E/S
para acceder a memoria



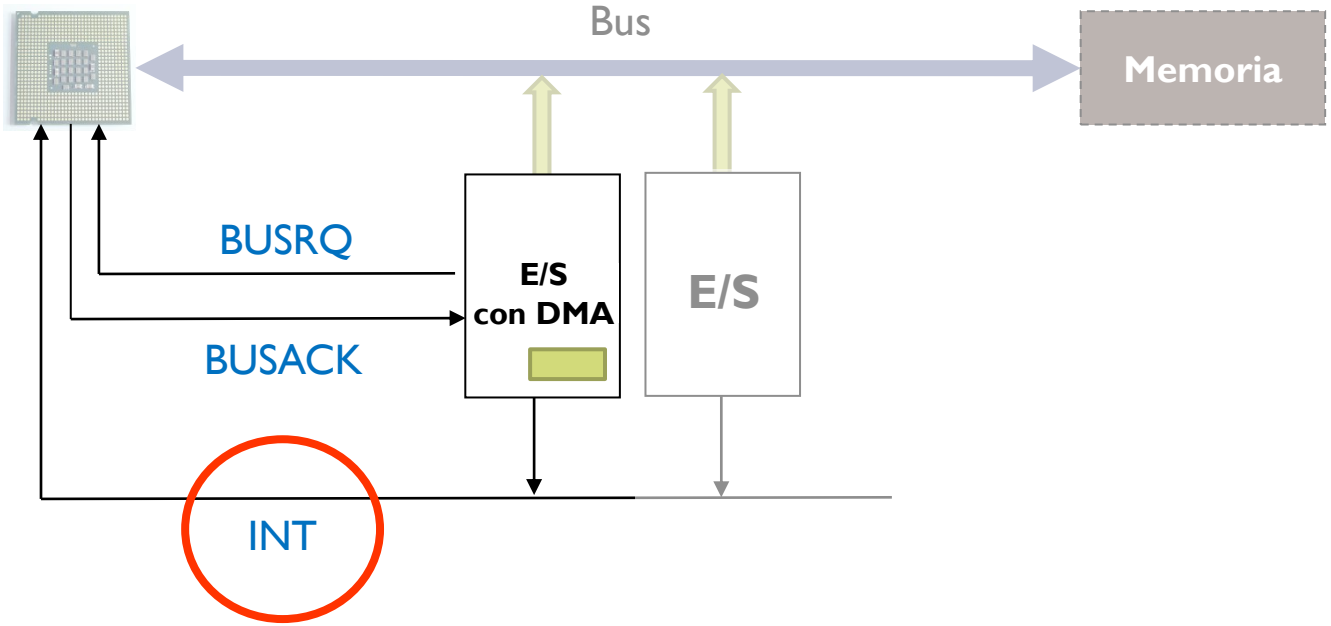
Ejemplo
E/S por DMA

Cada dato transferido a memoria supone:
- Pedir permiso para acceder a memoria (BUSRQ)
- Esperar el permiso (BUSACK)
- Transfiere a memoria
- Desactiva petición de permiso (BUSRQ)

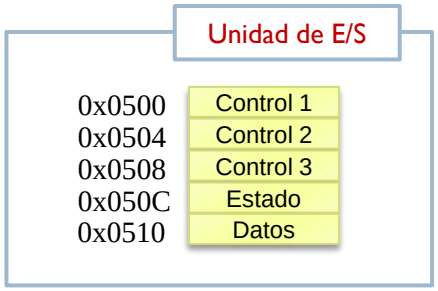


Ejemplo
E/S por DMA

Una vez transferido todos los datos:
- Generar una interrupción (INT) para avisar a la CPU



Ejemplo
E/S por DMA

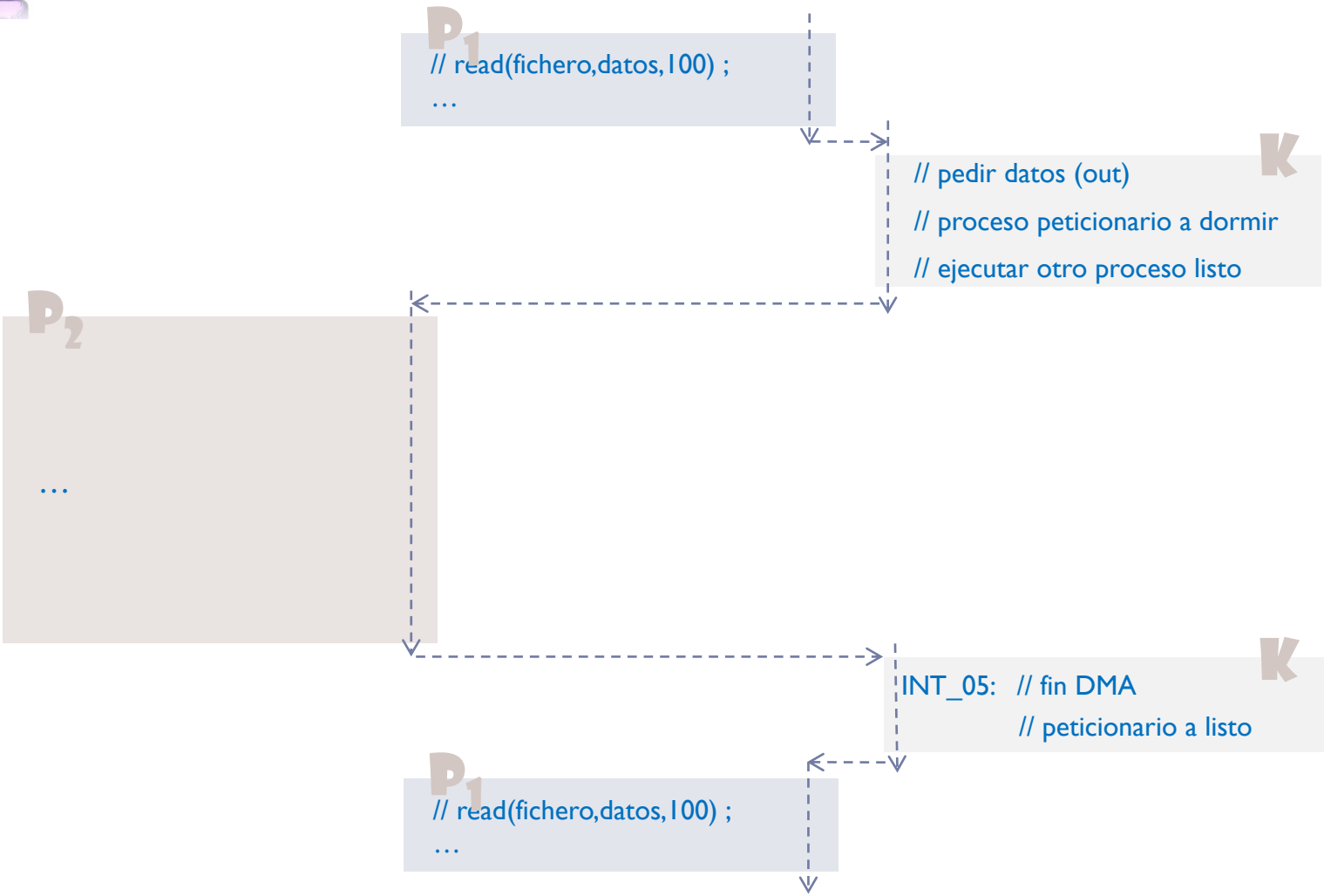


- Información de control 1
  - 0: leer, 1: escribir
- Información de control 2
  - Dirección memoria
- Información de control 3
  - Número elementos
- Información de estado
  - 0: dispositivo ocupado
  - 1: dispositivo (dato) listo
- Datos
  - Dato del dispositivo

```
petición:
        // Programar la petición del bloque
        out(0x500,0) ;          // leer
        out(0x504,p.datos) ;    // dirección vector
        out(0x508,100) ;        // nº elementos

        // Cambio de contexto voluntario (C.C.V.)
        …
```

```
INT_05:   // leer estado
          in(0x50C,  &p.status) ;

          if (p.status …

          // poner proceso peticionario a listo
          ret_int # restore registers & return
```

Ejemplo
E/S por DMA



Principales tipos de protocolos

- Petición -> respuesta individual
  - Mayoría de dispositivos
- Solo petición
  - Ej.: tarjeta gráfica
  - E/S programada (rápidos o tiempo real)
- Solo respuesta
  - Ej.: reloj
  - E/S por interrupciones (generan datos sin petición previa)
- Petición -> respuesta compartida
  - Ej.: disco duro

Grupo ARCOS
Departamento de Informática
Universidad Carlos III de Madrid

Lección 3a

procesos, periféricos, *drivers* y servicios ampliados

Diseño de Sistemas Operativos
Grado en Ingeniería Informática y Doble Grado I.I. y A.D.E.