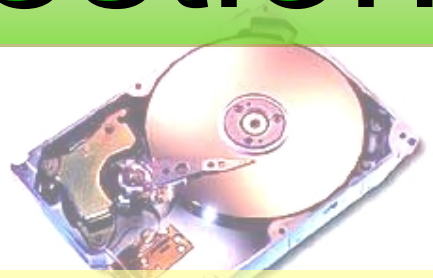
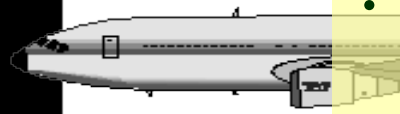
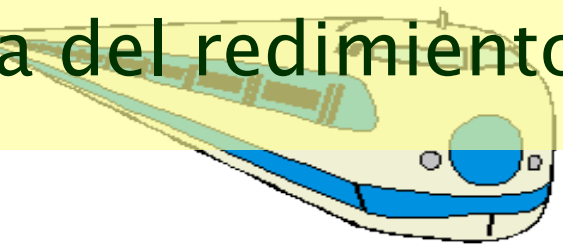
5

Gestión de E/S

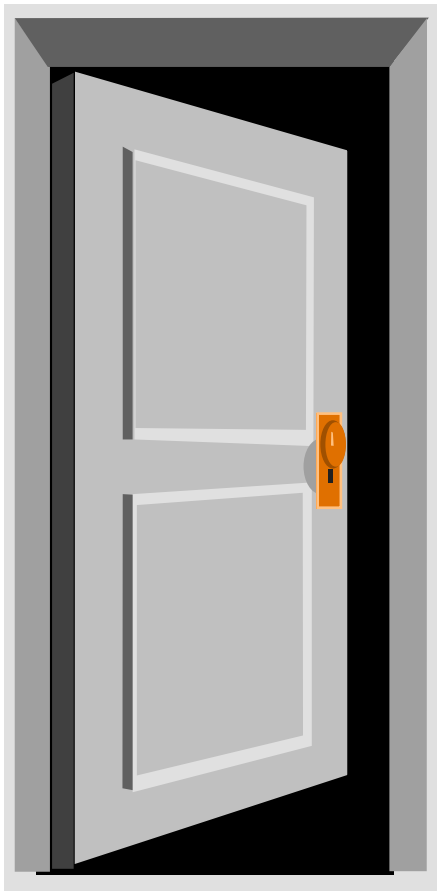


- Servicios de E/S que suministra el SO
- Hardware de Entradas/salidas
- Implementación de los servicios
- Mejora del redimiento de E/S





Hardware de E/S



- Elementos hardware del sistema de E/S y arquitectura:
  - Bus
  - Puerto
  - Controlador
  - Dispositivo
- Comunicación con el hardware de E/S:
  - Sondeo
  - Interrupciones
  - DMA

Arquitectura hardware del sistema de E/S

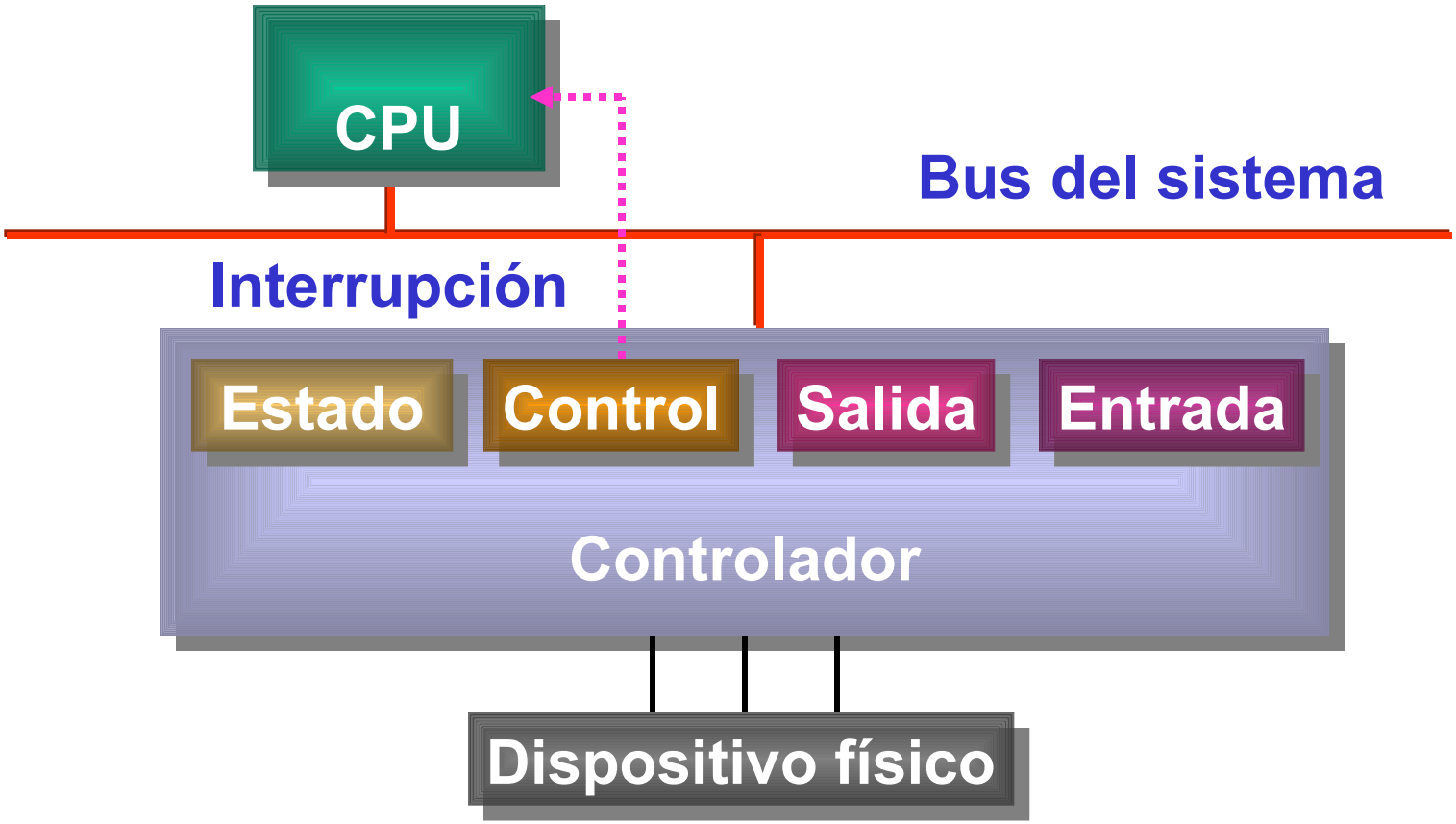
- El hardware asociado con un dispositivo de E/S consta de cuatro elementos básicos:
  - Un **bus** para comunicarse con la CPU y es compartido entre varios dispositivos.
  - Un **puerto** que consta de varios registros:
    - **Estado**–indica si esta ocupado, los datos están listos, o ha ocurrido un error.
    - **Control**–operación que ha de realizar.
    - **Datos_entrada**–datos a enviar a CPU.
    - **Datos_salida**–datos recibidos de la CPU.

Arquitectura (ii)

- Un controlador que recibe ordenes del bus del sistema, traduce ordenes en acciones del dispositivo, y lee/escribe datos desde/en el bus del sistema.
- El propio dispositivo.

- Existen una gran variedad de dispositivos:
  - Tradicionales: discos, impresoras, teclado, modem, ratón, pantalla, etc.
  - No tradicionales: joystick, actuador de robot, superficie de vuelo de un avión, sistema de inyección de un coche, etc.

Arquitectura (y iii)



Comunicación entre SO y dispositivo de E/S: Sondeo

- Los pasos a seguir al usar sondeo son:
  - CPU espera hasta que el estado sea libre.
  - CPU ajusta el registro de ordenes y datos-salida, si la operación es de salida.
  - CPU ajusta el estado a orden-preparada.
  - El controlador reacciona a orden-preparada y pone estado a ocupado. Lee registro de ordenes y ejecuta orden, pone un valor en datos-salida, si es una orden de salida.

Sondeo (cont.)

- Suponiendo que la orden tiene éxito, el controlador cambia el estado a ocioso.
- La CPU observa el cambio a ocioso y lee los datos si es una operación de salida.

- Es buena elección si los datos van a ser manejados al instante (un modem o teclado), ua que los datos se perderían si no se retiran del dispositivo lo suficiente-mente rápido; pero ¿y si el dispositivo es lento comparado con la CPU?

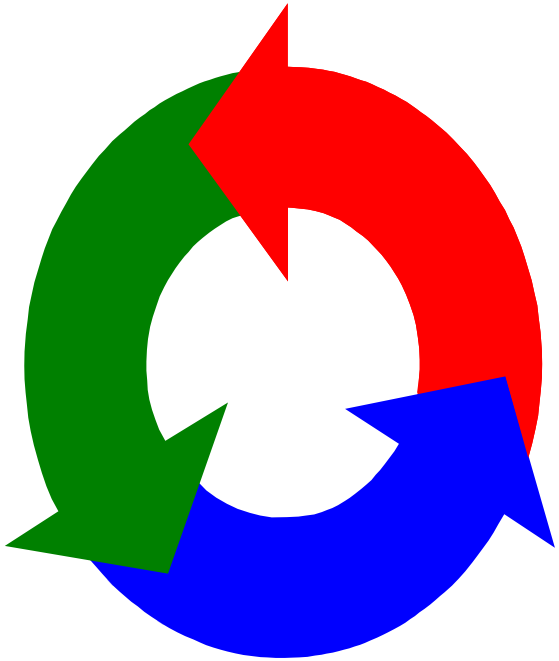
Comunicación entre SO y dispositivo: interrupción

- En lugar de tener la CPU ocupada esperando la disponibilidad del dispositivo, el dispositivo interrumpe a la CPU cuando ha terminado una operación de E/S.
- Cuando se produce la interrupción de E/S:
  - Determinar que dispositivo la provocó.
  - Si la última orden fue una operación de entrada, recupera los datos del registro del dispositivo.
  - Inicia la siguiente operación para el dispositivo.

Acceso directo a memoria (DMA)

- La CPU recupera la información byte a byte ⇒ no adecuado para grandes volúmenes..
- DMA (*Direct Memory Access*) – Controlador de dispositivo que puede escribir directa-mente en memoria. En lugar de registros de e/s, tiene un registro de dirección.
  - La CPU indica al DMA la ubicación de la fuente/destino de la transferencia.
  - DMA opera el bus e interrumpe a la CPU cuando se completa la transferencia.
- DMA y CPU compiten por el bus de memoria.

Implementación



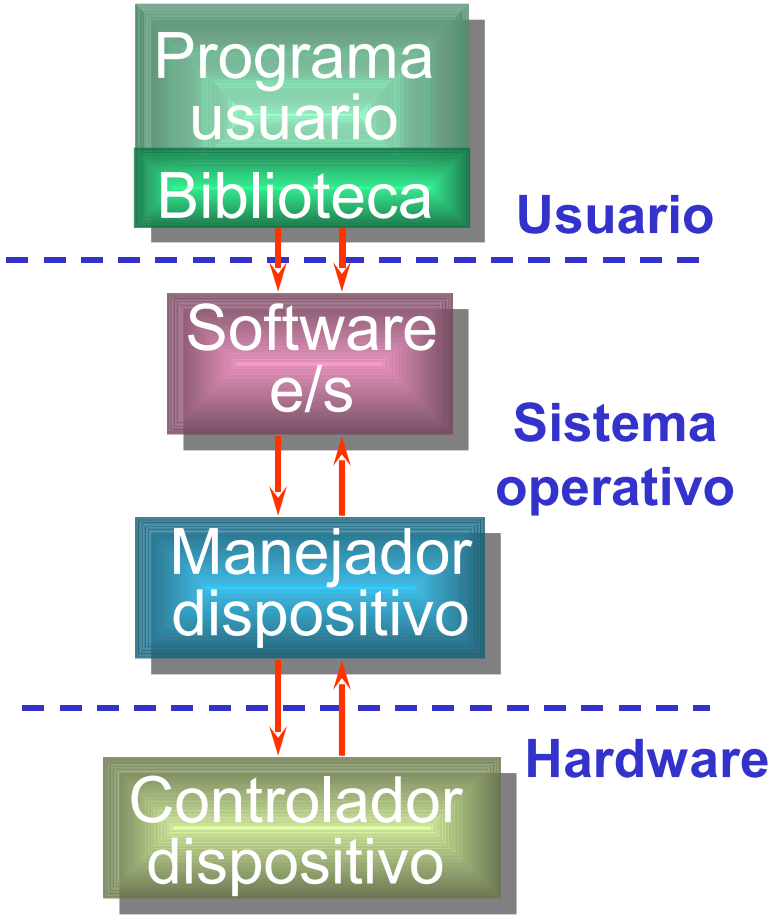
- Servicios suministrados
- Estructura del software de E/S:
  - Manejadores
  - Soft. independiente del dispositivo
  - Software e/s en espacio de usuario

Servicios de E/S

- Denominación de archivos y dispositivos.
- Control de acceso.
- Operaciones adecuadas para archivos y dispositivos.
- Asignación de dispositivos.
- Búfering. caché, y spooling, para suministrar una comunicación eficiente con el dispositivo.
- Planificación de E/S.
- Gestión de errores y recuperación de fallos asociados con el dispositivo.
- Aislar en un módulo las características y conducta específica del dispositivo.

Arquitectura software del sistema de E/S

- Podemos estructurar el soft. de e/s en capas:
  - Manejadores de dispositivos
  - Software de e/s independiente del dispositivo
  - Software a nivel de usuario.



Manejador de dispositivos

- Contienen todo el código dependiente del dispositivo. Cada manejador gestiona un tipo o clase de dispositivo.
- Acepta peticiones “abstractas” de la capa de software independiente del dispositivo, y controla que la petición se realiza:
  1. Traduce petición abstracta en ordenes para el controlador del dispositivo.
  2. Se bloquea o no,según tipo de operación
  3. Si no hay errores, da respuesta si es necesario, y retorna al llamador.

Software de e/s independiente del dispositivo

- Realiza las tareas comunes a todos los dispositivos y suministra una interfaz común al usuario.
- Las principales funciones son:
  - Suministra interfaz uniforme a los manejadores
  - Realiza la designación de dispositivos
  - Implementa la protección de dispositivos
  - ... (continua)

Software de e/s independiente del dispositivo (y ii)

- Establece el tamaño de bloque independiente del dispositivo (para dispositivos de bloques) .
- Implementa el búfering
- Realiza la asignación de almacenamiento en dispositivos de bloques (básicamente discos)
- Asignar y liberar dispositivos dedicados
- Informar de errores producidos en la realización de la operación.

Designación de dispositivos

- Espacio de nombre de dispositivos –define cómo identificar y nombrar los dispositivos
- Existen diferentes espacios de nombre:
  - **Espacio de nombres hardware** –especifica el dispositivo por el controlador al que esta ligado y el número de dispositivo lógico dentro del controlador.
  - **Espacio de nombres kernel** –utilizado por el núcleo, suele basarse en el anterior.
  - **Espacio de nombres de usuario** –debe ser un esquema sencillo y familiar.

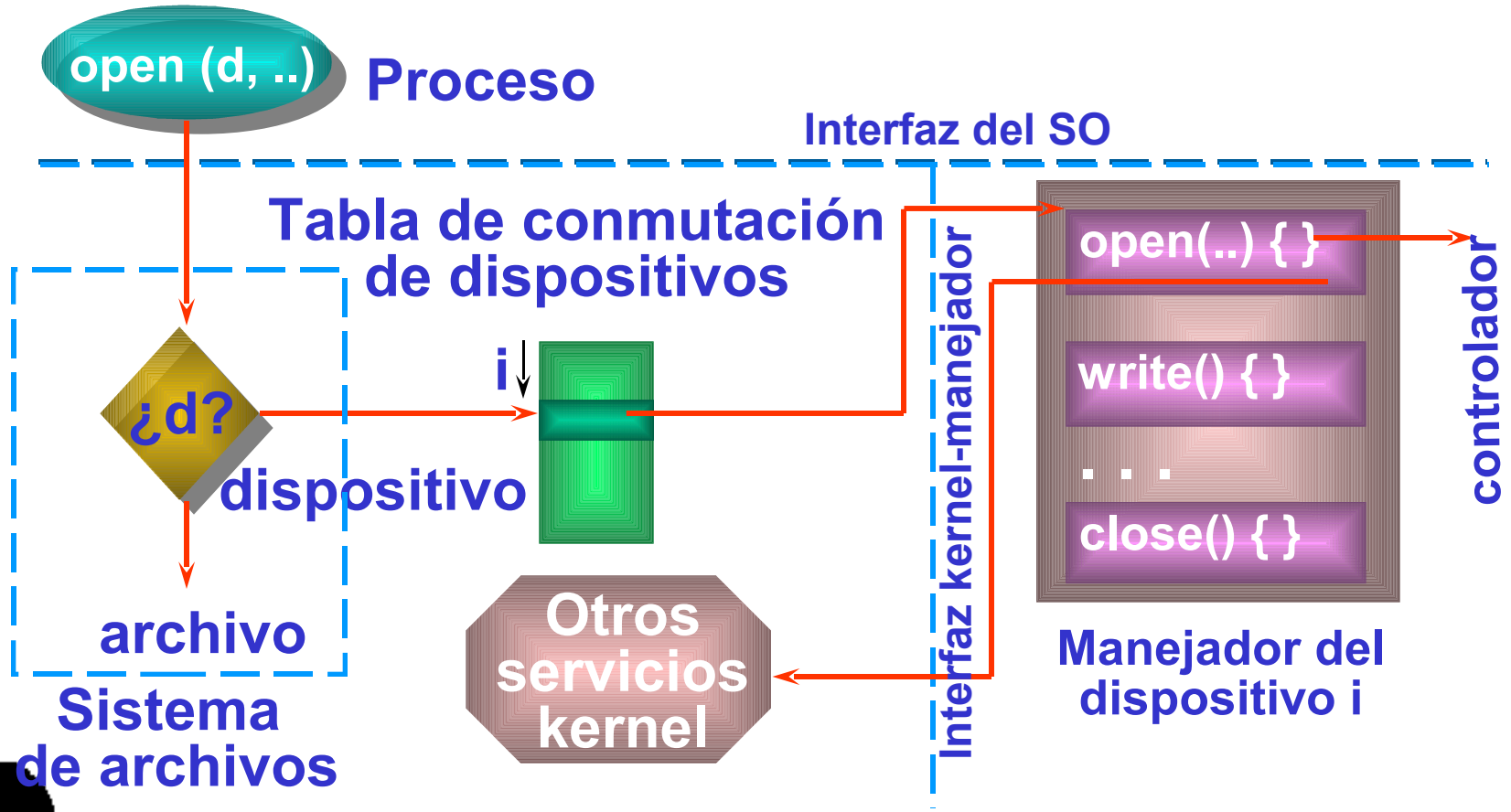
Designación (cont.)

- El sistema de e/s independiente del dispositivo define las semánticas de los espacios de nombres kernel y usuario, y establece las correspondencias entre ellas.
- En Unix, el espacio de nombres kernel identifica un dispositivos por:
  - número principal–identifica el controlador
  - número secundario–instancia del dispositivo.
  - tipo de dispositivo –carácter o bloque.

Espacio de nombres de usuario

- La forma de designación más extendida es la integración del espacio de nombres de dispositivos en el de archivos.
- El concepto central es el archivo de dispositivo: permite manejar de la misma forma archivos y dispositivos (mismas llamadas al sistemas), y aplicarles los mismos mecanismo de protección.
- En UNIX un archivo dispositivo es un archivo especial que no contiene datos, si no el número principal y secundario.

Designación e independencia del dispositivo



Búfering de E/S

- Los dispositivos suelen tener una pequeña memoria en la tarjeta para almacenar datos temporalmente antes de transferirlos a/desde la CPU.
- ¿Por qué tener búferes en el SO? Para:
  - Acoplar la diferencia de velocidades entre la CPU y el dispositivo.
  - Hacer frente a la diferencia de tamaños de transferencia de datos entre dispositivos.
  - Minimizar el tiempo que el proceso de usuario esta bloqueado en una escritura.

Caché

- Mejorar el rendimiento del disco reduciendo el número de acceso al disco.
- Mantener bloques de datos recientemente usados en memoria después de que se complete la llamada de E/S que lo trajo.
- Ej.:

```
read(DireccionDisco)
      if (bloque en memoria)then
         Retorna su valor
      else
         LeeSector(DireccionDisco).
```

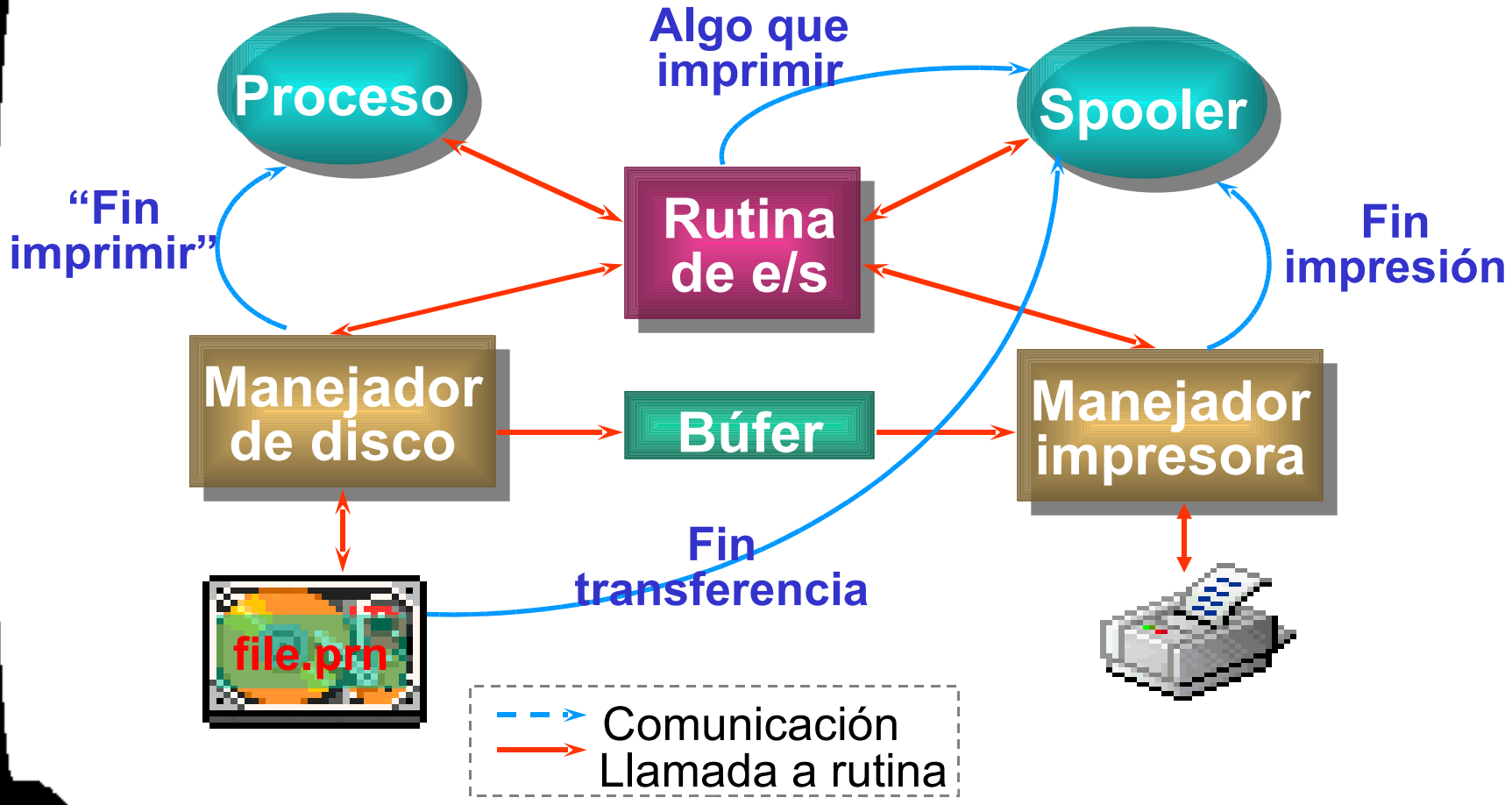
Caché (cont.)

- Ej.:
```
write(direcciónDisco)
If (bloque en memoria) then
    actualiza su valor
else
    asigna espacio en memoria; leelo de disco;
    actualiza su valor en memoria.
```
- Políticas de escritura:
  - Write-trouhg –escribir todos los niveles de memoria que contienen el bloque, incluido el disco. Muy fiable.
  - Write-back – escribir sólo en memoria más rápida que contiene el bloque; escribir en memorias más lentas y disco más tarde. Más rápido.

Software de e/s en espacio de usuario

- **Bibliotecas estándar de e/s** – permiten realizar las llamadas al sistema de e/s para:
  - Gestión el formatos (ej. *printf*)
  - Control de los dispositivos (p. ej. *iocntl*)
- Spooling – técnica para manejar dispositivos dedicados en sistemas multiprogramados. P. ej. No asignamos impresora sino que generamos la impresión en un archivo; un proceso especial manda los archivos a impresión. Igual que el correo electrónico.

Spooling


Proceso
Algo que imprimir
Spooler
"Fin imprimir"
Rutina de e/s
Fin impresión
Manejador de disco
Búfer
Manejador impresora
Fin transferencia
file.prn
Comunicación
Llamada a rutina

Visión del programador

- Visión del usuario de los dispositivos de E/S ofrecida por el SO.
- Una operación completa de E/S.
- Cómo afecta la visión del programador a la estructura del manejador.

Visión del programador de los dispositivos de E/S

- El SO suministra una interfaz de los dispositivos que simplifica el trabajo del programador.
  - Interfaz estándar para dispositivos relacionados.
  - El manejador encapsula las dependencias del dispositivo.
  - El SO puede soportar nuevos dispositivos simplemente con suministrar el manejador del dispositivo.

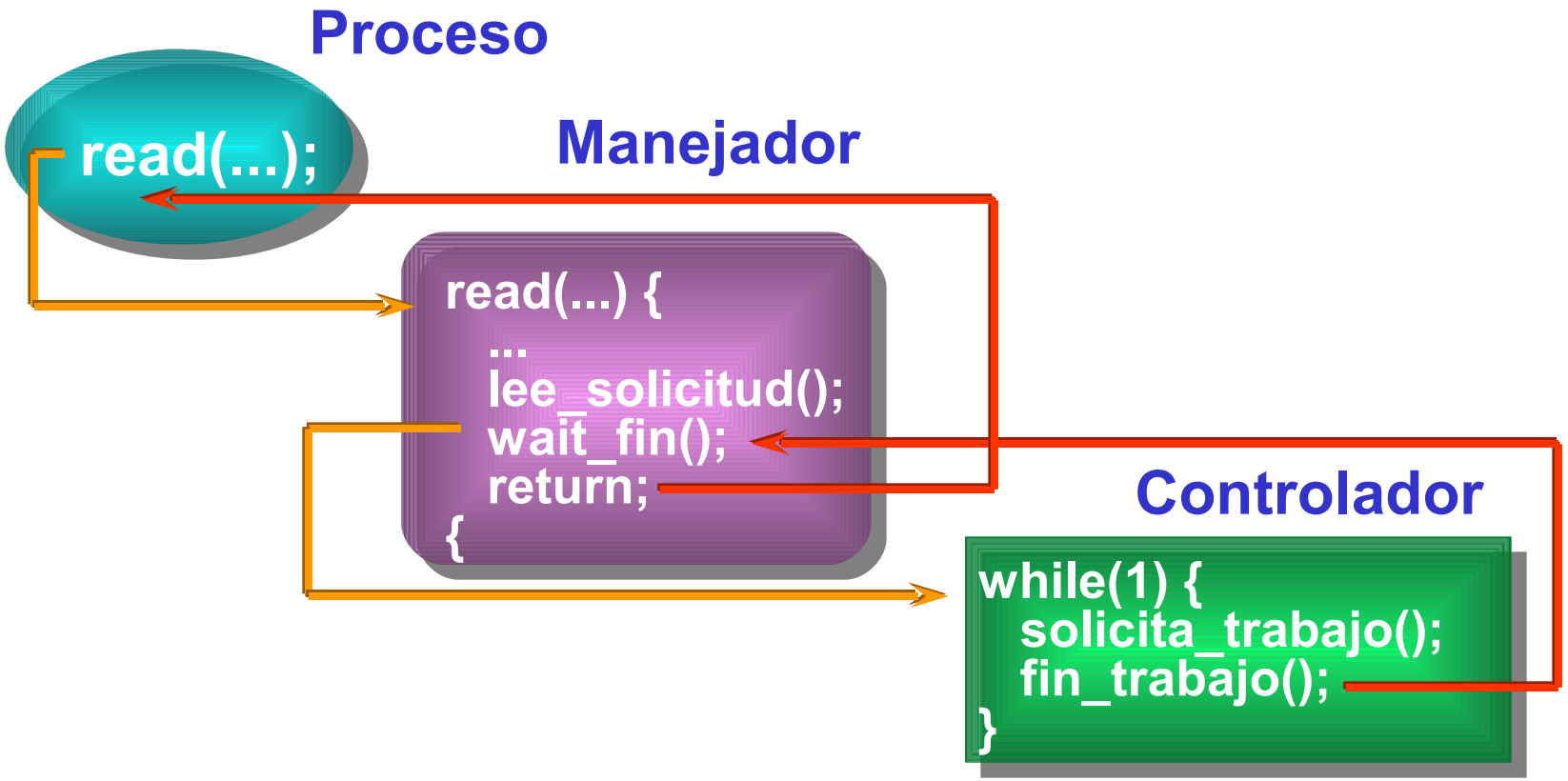
Visión del programador (cont.)

- Características del dispositivo:
  - Unidad de transferencia: carácter/bloque
  - Método de acceso: secuencial/aleatorio
  - Temporización: síncrona/asíncrona. Observar que la mayoría de los dispositivos son asíncronos, mientras que la llamadas al sistema de E/S son síncronas. El SO implementa *E/S bloqueantes*.
  - Compartido o dedicado.
  - Velocidad.
  - Operación: entrada, salida, o ambas.

Todo junto: una lectura

① El usuario solicita una lectura de dispositivo
② Si los datos están en un búfer, salta a paso 3. Si no están:
   a. El SO instruye al dispositivo y "espera".
   b. Cuando el dispositivo tiene los datos interrumpe a la CPU, que los transfiere a un búfer del SO. Con DMA ...
③ El SO transfiere los datos al proceso de usuario y lo desbloquea.
④ Cuando el proceso alcanza la CPU, sigue la ejecución después de la llamada.

Operación de lectura



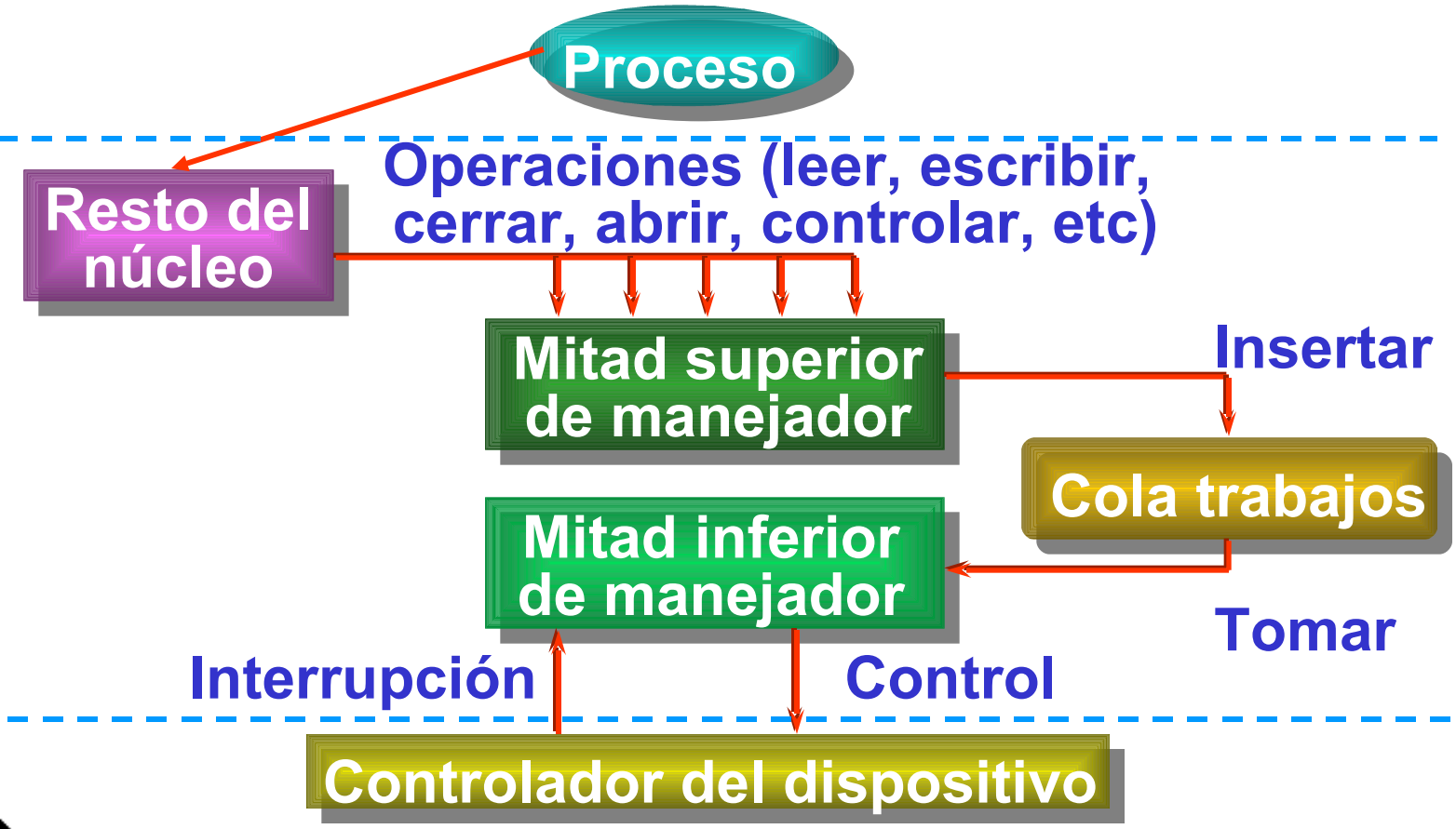
Un esclavo con dos amos

- El dibujo muestra como el manejador obedece ordenes del resto de módulos del SO y del controlador del dispositivo. Siendo esto así ¿cómo espera, `wait_fin()`, el manejador ? si este, que se ejecuta en el contexto del proceso, debe bloquearse para implementar E/S síncronas desde el punto de vista del programador, y la invocación de las rutinas de servicio de interrupción es asíncrona y no debe utilizar el contexto del proceso.

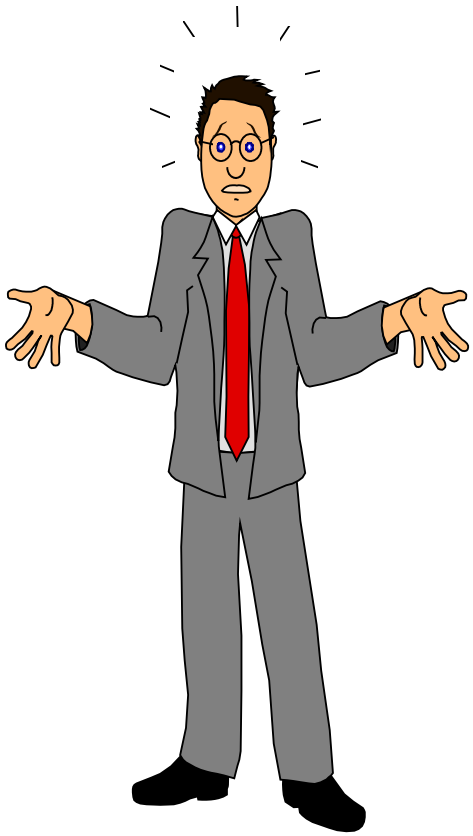
Estructura de dos niveles para el manejador

- El manejador se divide en dos mitades:
  - La **mitad superior** contiene las rutinas síncronas, se ejecuta en el contexto del proceso y puede acceder a su espacio de direcciones. Puede bloquear al proceso.
  - La **mitad inferior** contiene las rutinas asíncronas, no accede al contexto del proceso y normalmente no tiene relación con el proceso. No puede bloquearse, pues bloquearía a un proceso no relacionado con la interrupción.

Manejador de dos mitades



Rendimiento y sus soluciones



A modo de conclusión:

- Problemas de rendimiento.
- Posibles soluciones a los problemas de rendimiento.

Problemas de rendimiento

- Las E/S son costosas por varias razones:
  - Involucran movimientos físicos lentos (cabezal disco) o líneas de comunica-ciones (teléfono-red) que también lo son.
  - Los dispositivos de E/S son a menudo disputados por múltiples procesos.
  - Las operaciones de E/S se suministran por medio de llamadas al sistema y gestión de interrupciones, que son lentas.

Rendimiento: soluciones (i)

- Reducir el número de veces que los datos son copiados manteniendolos en caché.
- Reducir la frecuencia de interrupciones utilizando, si es posible, grandes transferencias de datos.
- Descargar computación de la CPU principal utilizando controladores DMA.

Rendimiento: soluciones (y ii)

- Aumentar el número de dispositivos para reducir la contención de uno único, y así, mejorar el uso de CPU.
- Incrementar memoria física para reducir la cantidad de tiempo en paginación y por ello mejorar el uso de CPU.