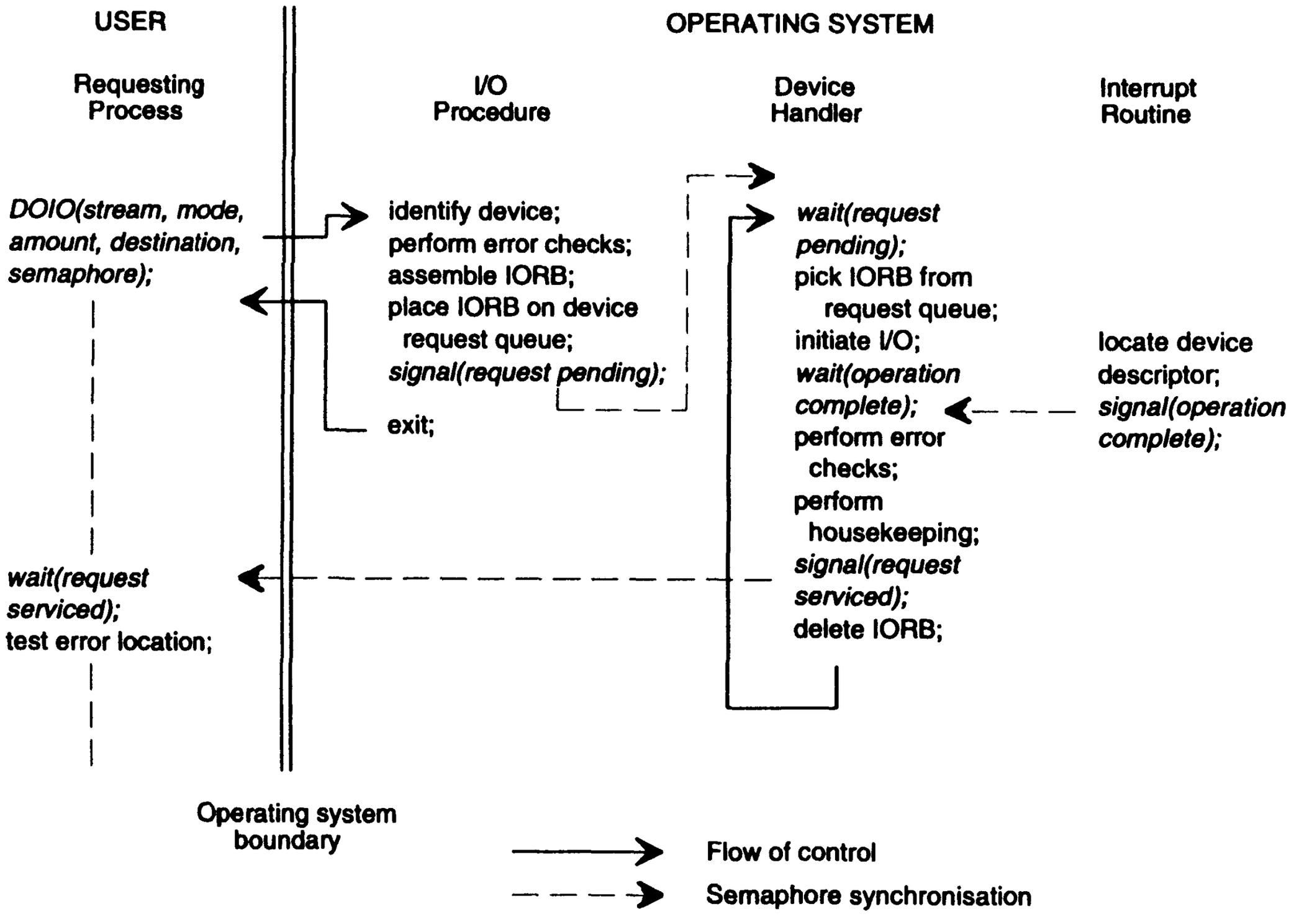
Arquitectura de manejo de E/S

El panorama completo

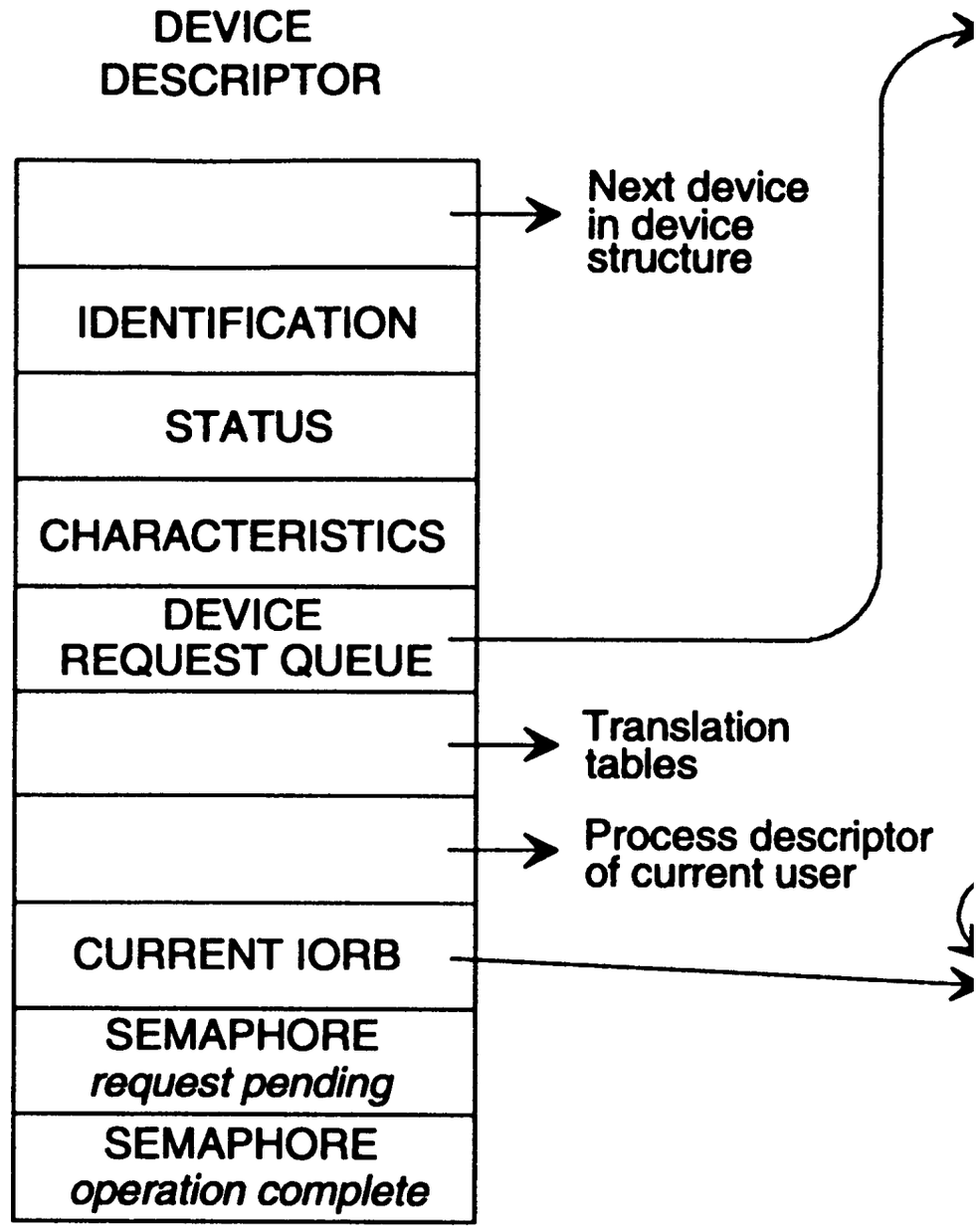


Repaso...

- Dispositivo
- Controlador
- Driver
- Interface (API): *DOIO*() / *ioctl*()
- Manejador de interrupciones

Descriptor de dispositivo

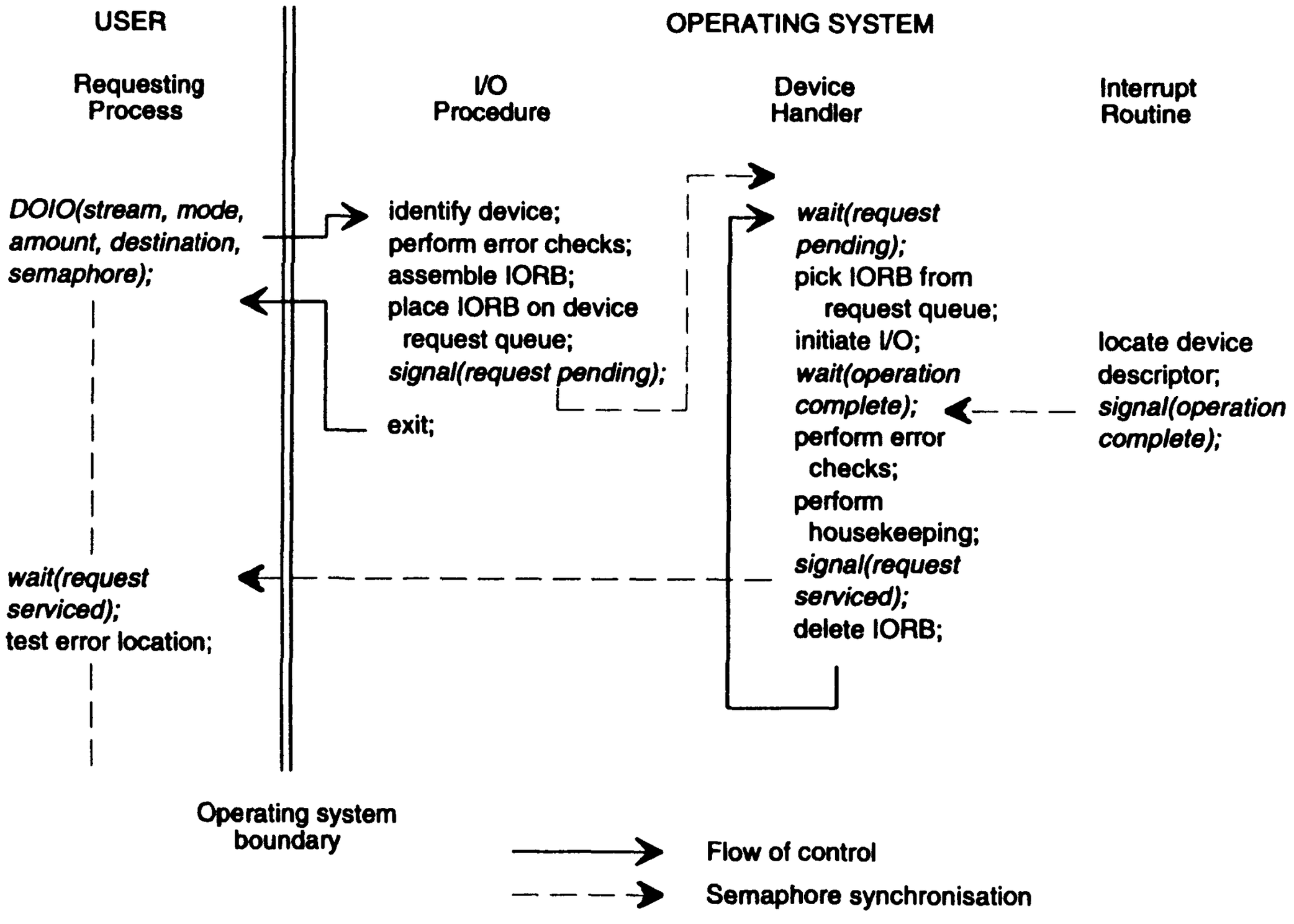
- Es una estructura de datos. Cada dispositivo tiene su propio descriptor. Algunos campos son:
  - Identificador del dispositivo
  - Apuntadores a funciones para manipular el dispositivo (Abrir/Cerrar/Leer/ Avanzar/...)
  - Estado: Si el disp. Está ocupado, libre, o roto
  - El proceso que lo está usando
  - Dos monitores para sincronización
- El núcleo accede a los dispositivos mediante la lista “estructura de dispositivos”.



Repaso...

- Dispositivo
- Controlador
- Driver
- Interface (API): *DOIO*() / *ioctl*()
- Manejador de interrupciones

El panorama completo

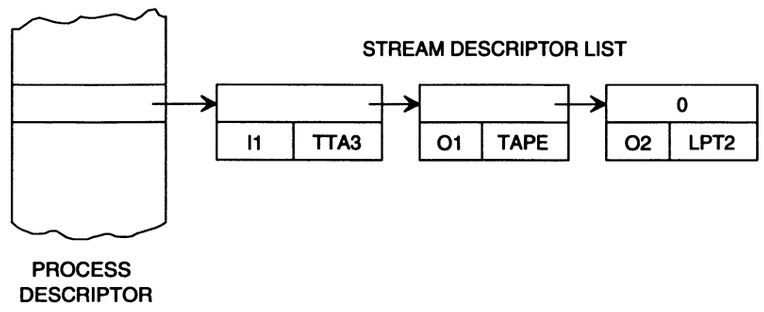


DOIO() / ioctl()

- DOIO() [Lister] e ioctl() [Unix], es la llamada al sistema que se ofrece al programador para solicitar control de un dispositivo.
    - Activar/Desactivar dispositivo; Avanzar/Rebobinar dispositivo, etc...

- Ofrece un acceso *uniforme* a todos los dispositivos: virtualmente todos son archivos
    - DOIO(stream, mode, amount, destination, rs_semaphore)
    - int ioctl(int fildes, unsigned long request, ...);

- Asociado a cada proceso hay una “lista de descriptores”:



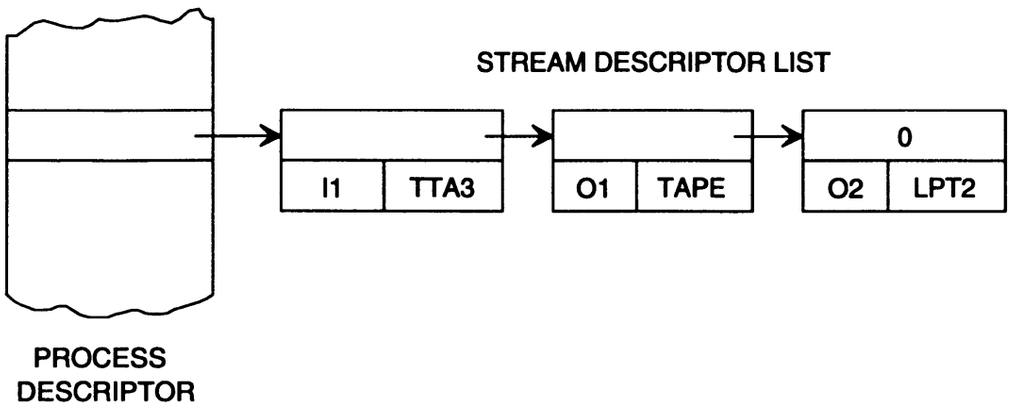
DOIO(stream, mode, amount, destination, rs_semaphore)

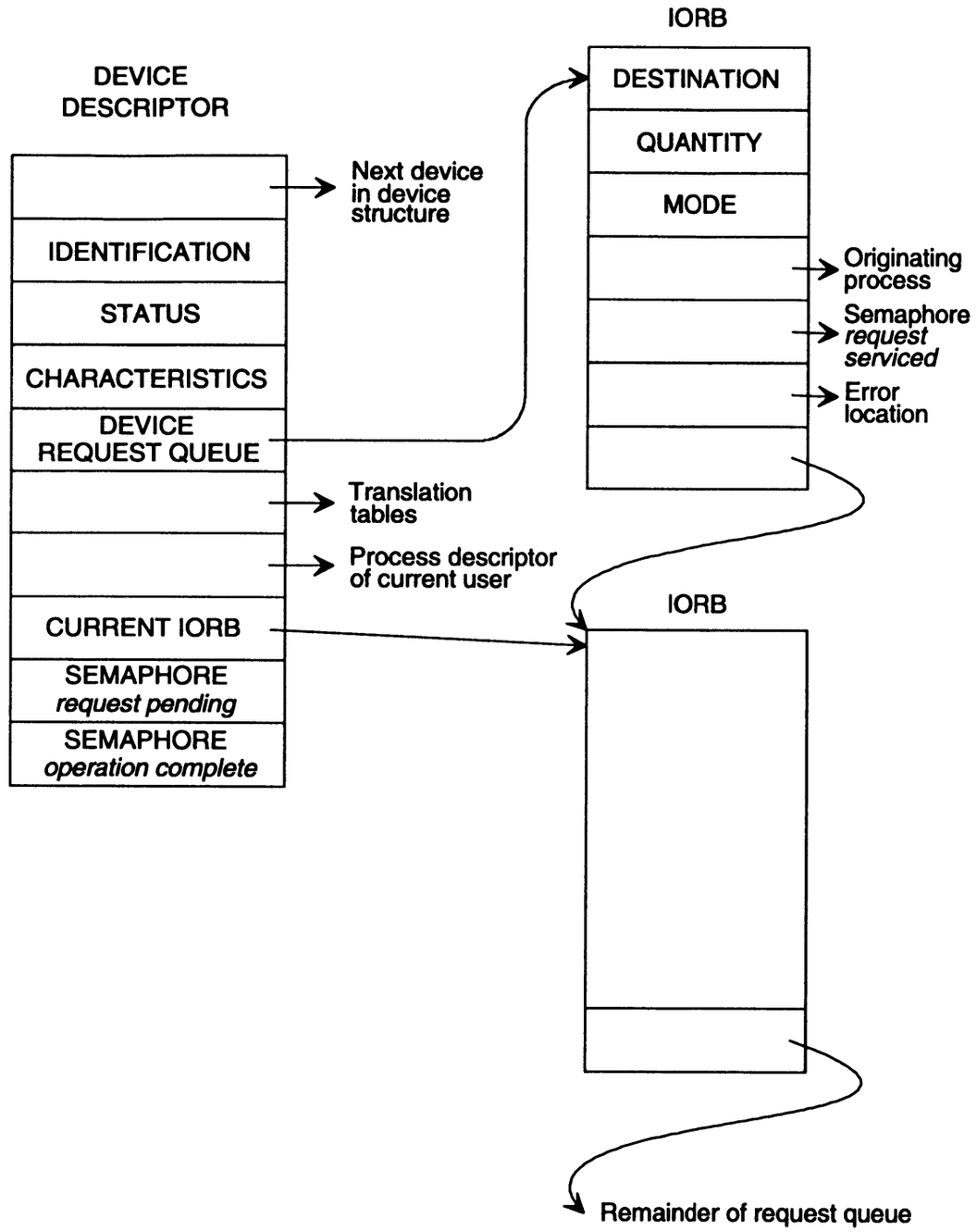
- stream: dispositivo a manipular
- mode: operación que se quiere realizar; p. ej. Reiniciar/Rebobinar/Avanzar
- amount: cantidad de datos a transmitirse
- destination: apuntador a bloque de memoria desde/hacia donde se copiarán los datos entre RAM y dispositivo
- rs_semaphore: semáforo para indicar que la transferencia ha terminado
    - Su trabajo es revisar que los parámetros sean consistentes con las características del dispositivo, y de ensamblar un Bloque de Petición de Entrada/Salida (IORB). El IORB será después procesado por el driver de dispositivo adecuado.

DOIO(stream, mode, amount, destination, rs_semaphore)

procedure DOIO(stream, mode, amount, destination, semaphore)

begin
  busca el descriptor de dispositivo en PCB->s_descriptors;
  checa los parámetros de acuerdo a las especificaciones del disp.
  if error entonces indica_error; exit
  contruye IORB;
  añade IORB a descriptor->IORB_queue;
  signal(descriptor->request_pending);
end;

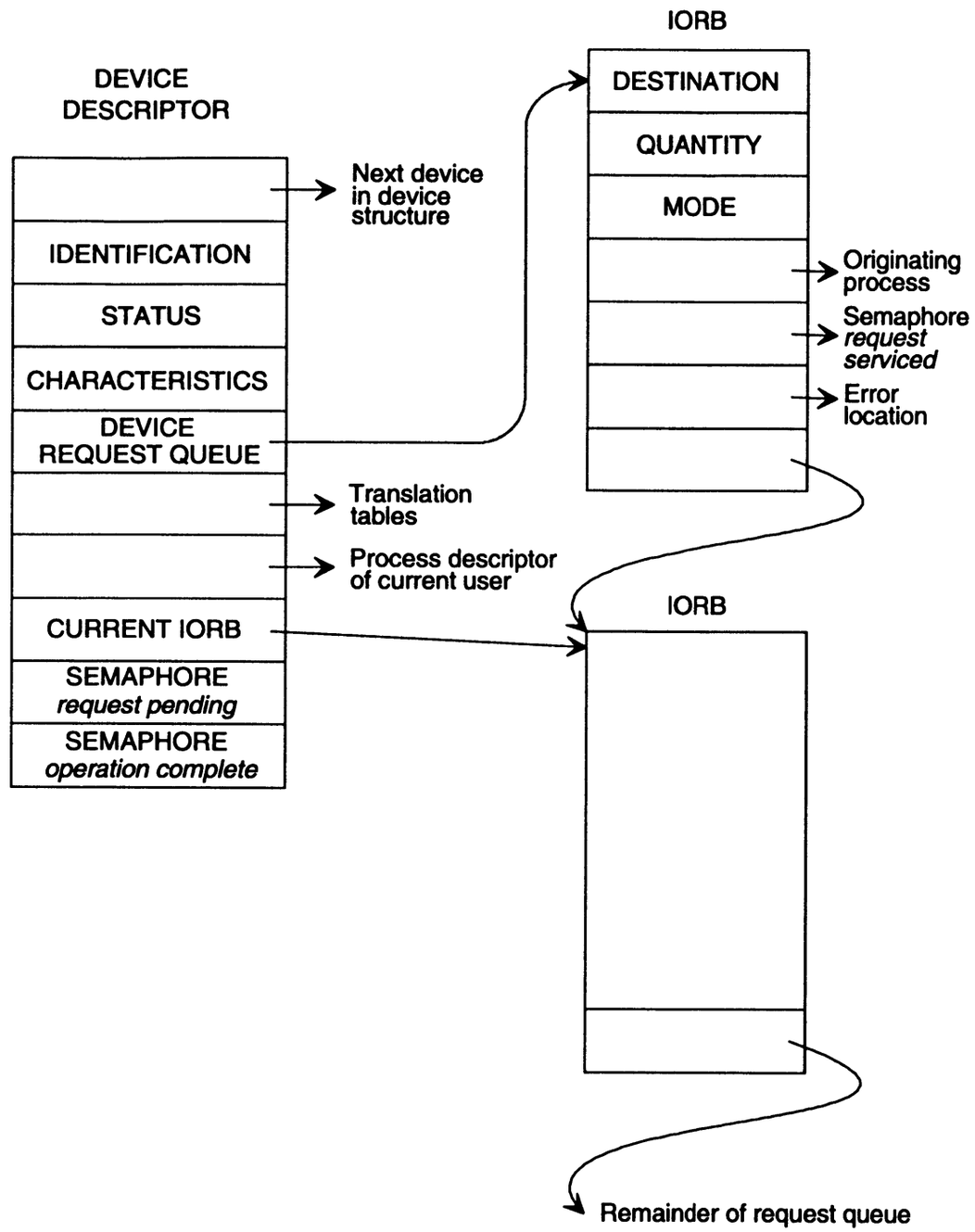




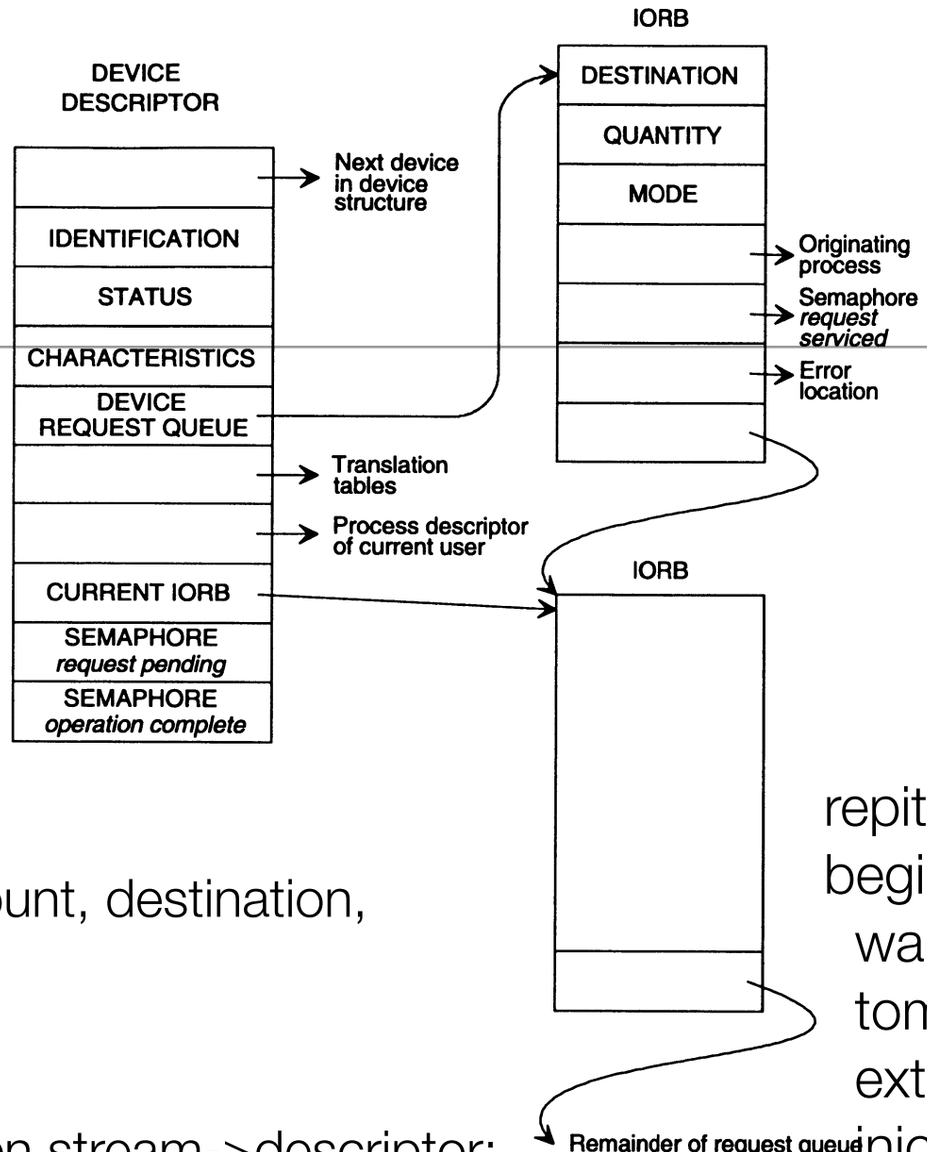
Driver de dispositivo

- Hay un proceso “driver” por cada dispositivo
  - Su trabajo es esperar a que haya peticiones (IORB’s) y procesarlas
  - Es un “consumidor” de IORB’s.

```
repite indefinidamente
begin
  wait(request_pending);
  toma un IORB de la cola de IORB’s;
  extrae los detalles de la solicitud;
  inicia operación de E/S;
  wait(operation_complete);
  if error then plant_error_info; exit
  traduce caracteres si es necesario;
  transfiere datos al destino;
  signal(request_serviced);
  delete IORB;
end;
```



Proceso
solicitando IO
+
driver



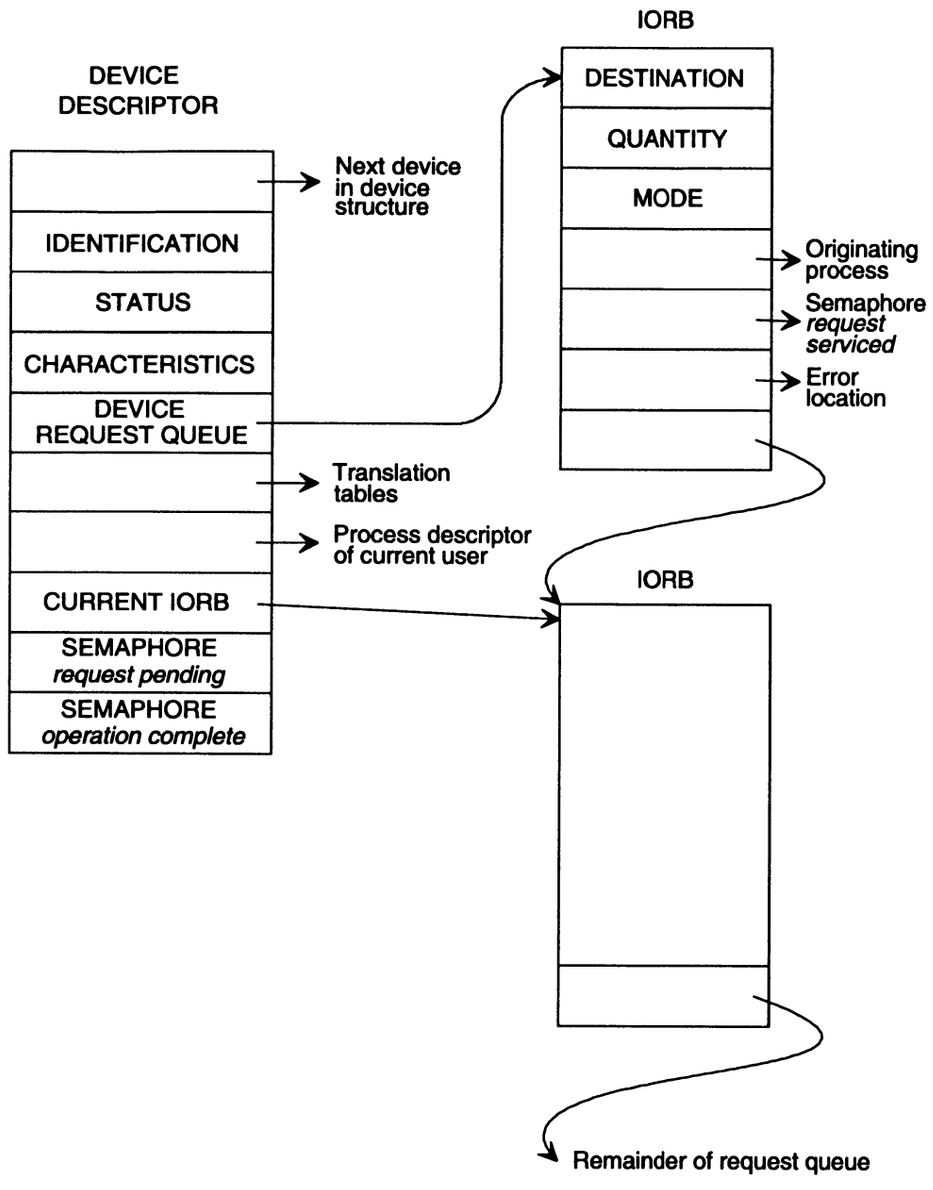

```
procedure DOIO(stream, mode, amount, destination,
semaphore)

begin
  busca el descriptor de dispositivo en stream->descriptor;
  checa los parámetros de acuerdo a las especificaciones del
disp.
  if error entonces indica_error; exit
  contruye IORB;
  añade IORB a descriptor->IORB_queue;
  signal(descriptor->request_pending);
end;
```

```
repite indefinidamente
begin
  wait(request_pending);
  toma un IORB de la cola de IORB
  extrae los detalles de la solicitud;
  inicia operación de E/S;
  wait(operation_complete);
  if error then plant_error_info; exit
  traduce caracteres si es necesario
  transfiere datos al destino;
  signal(request_serviced);
  delete IORB;
end;
```

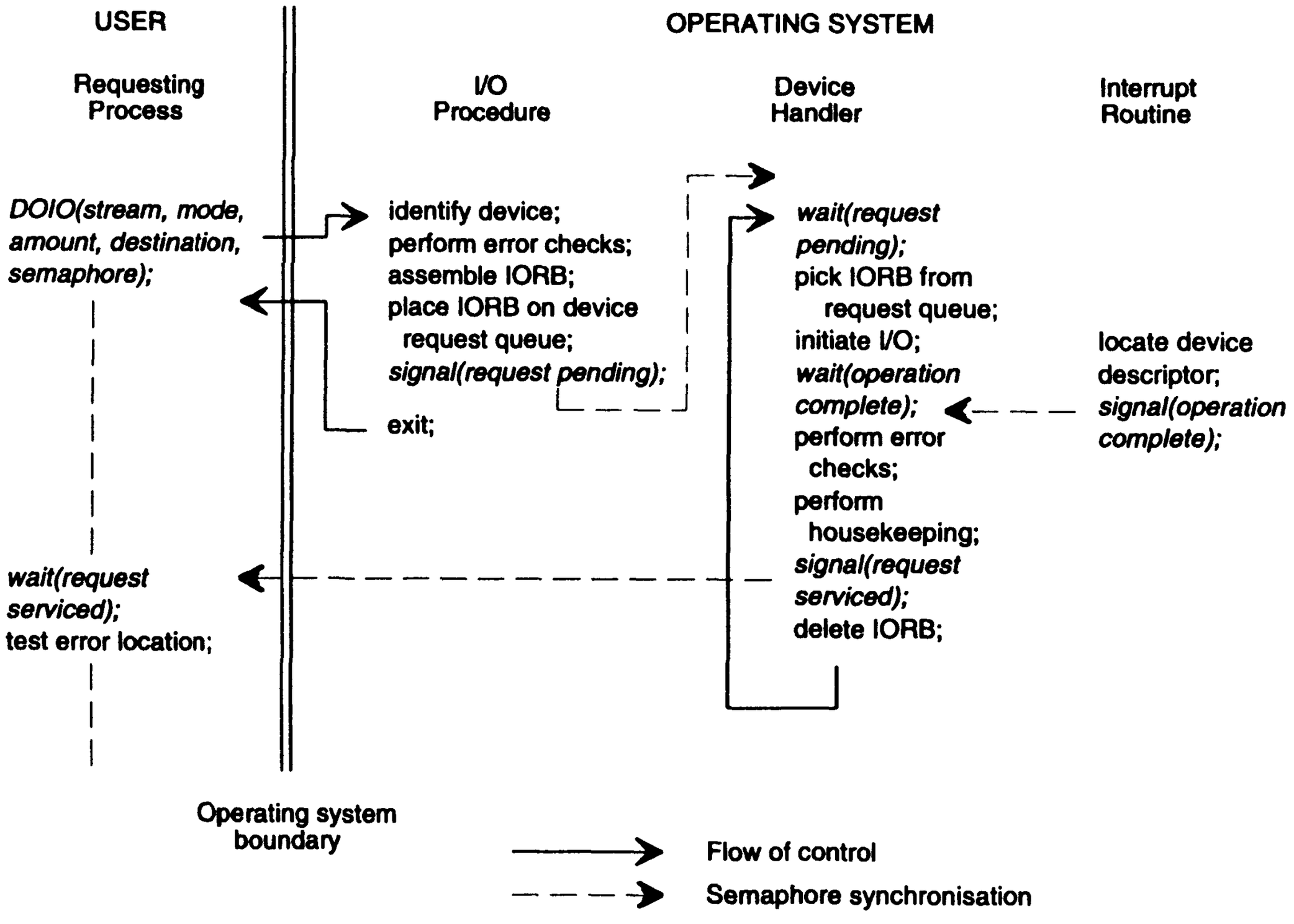
Manejador de interrupciones

- hace un signal(operation_complete) cuando el dispositivo lo interrumpe para indicar que finalizó la operación.




```
repite indefinidamente
begin
  wait(request_pending);
  toma un IORB de la cola de IORB
  extrae los detalles de la solicitud;
  inicia operación de E/S;
  wait(operation_complete);
  if error then plant_error_info; exit
  traduce caracteres si es necesario
  transfiere datos al destino;
  signal(service_requested);
  delete IORB;
end;
```

El panorama completo



Cómo vamos hasta ahora