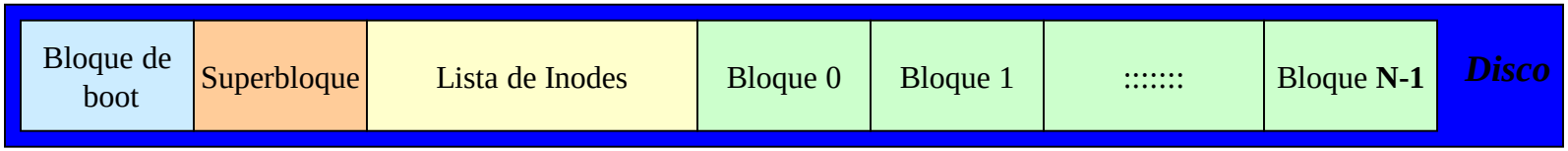
Resumen E/S Linux

Visión del Sistema de Ficheros (UNIX)



Bloques del Disco

Código encargado de buscar el SO y cargarlo en memoria para inicializarlo

Estado del sistema de ficheros:
Tamaño, Lista de bloque libres, Tamaño de la lista de Inodes, Lista de Inodes libres,...

Una entrada por cada fichero, con información de control del mismo.

Bloques de datos del Disco

Inode

Tipo
Permisos
Propietario
Grupo
Fechas
Nºenlaces
Tamaño

Índices de Bloques Datos

Estructura del i-node (UNIX)

i-node

Tipo
Permisos
Propietario
Grupo
Tamaño
Fechas
Nºenlaces
...

Índices de Bloques Datos

Tipo de fichero: Ordinario, Directorio, caracteres, de bloques, Pipe, Enlace, Shocket

Permisos de RWX para el propietario, Grupo o resto de usuarios

Identificación de Propietario y Grupo,

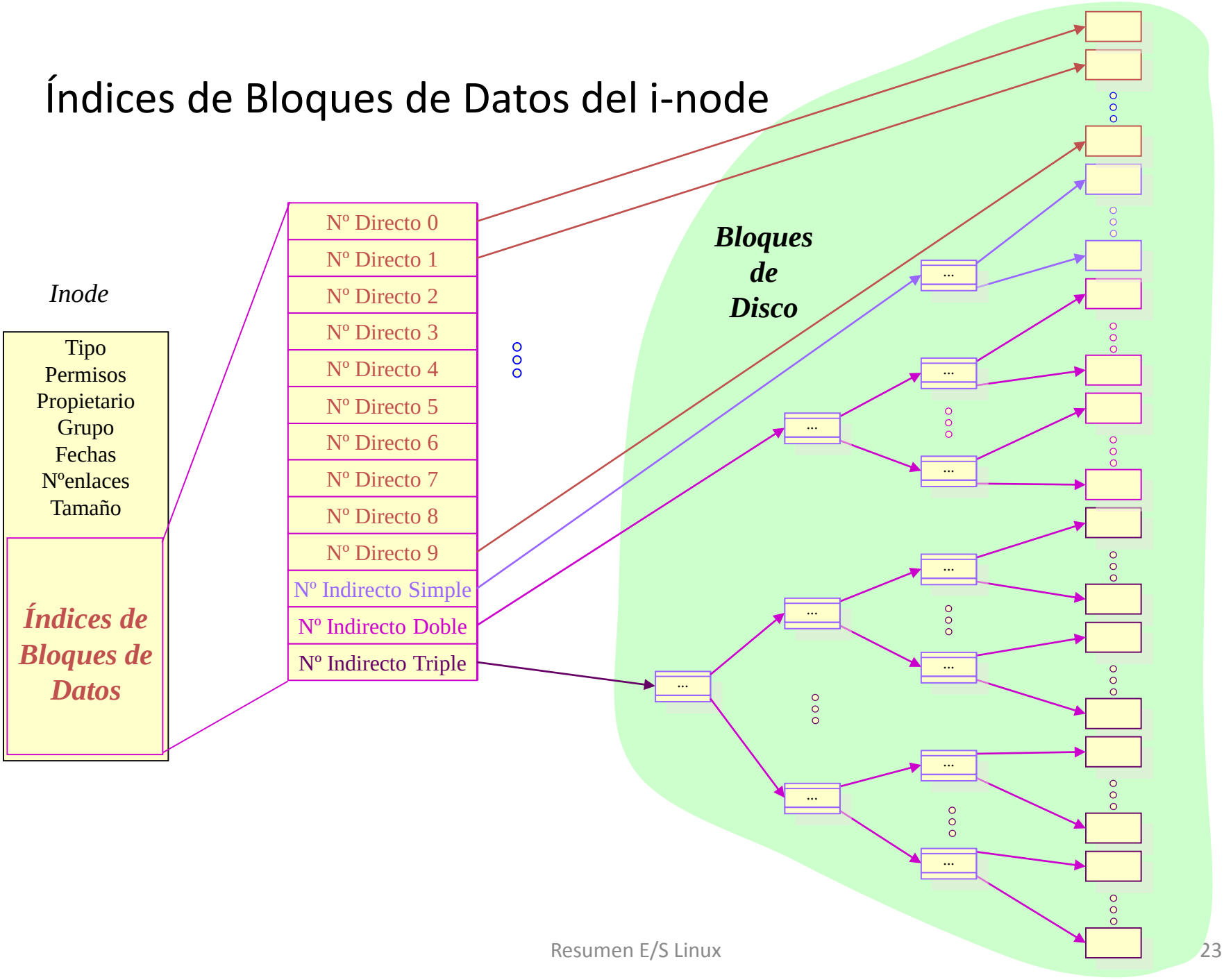
Tamaño en bytes del fichero

Varias fechas como la del último acceso o la de la última modificación de datos.

Nº de enlaces al fichero (nº de nombres en el sistema de ficheros)

Otros datos como número de dispositivo, stiky-Bit (cod. Reentrante), Cambiar UID o GIG.

Índices de Bloques de Datos del i-node



Sistema de Ficheros Virtual de Linux

- Diseñado para soportar diferentes sistemas de gestión de ficheros

- Sistema de Ficheros Virtual (VFS)
  - Representa una única interfaz a los procesos de usuario
  - Define un modelo de ficheros común
  - Supone que los ficheros son objetos con propiedades básicas, independientemente del sistema de ficheros real sobre el que están implementados

Sistema de Ficheros Virtual de Linux

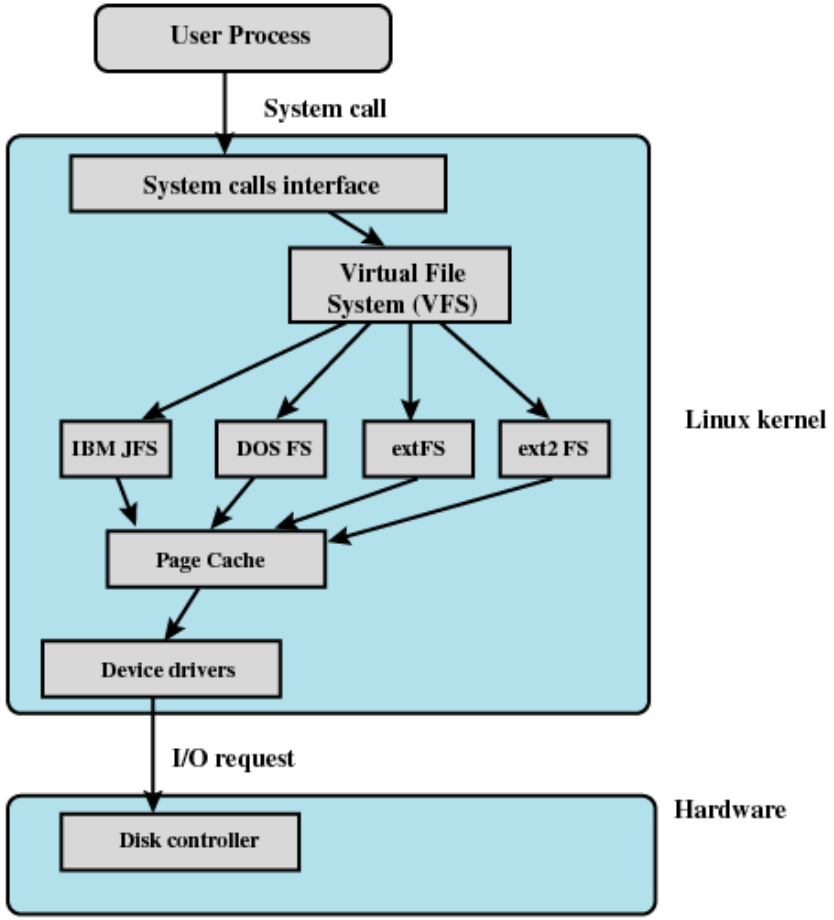


Figure 12.15 Linux Virtual File System Context

Fuente: W. Stallings: Sistemas Operativos. 5º Ed. Pearson Prentice-Hall, 2005

Sistema de Ficheros Virtual de Linux

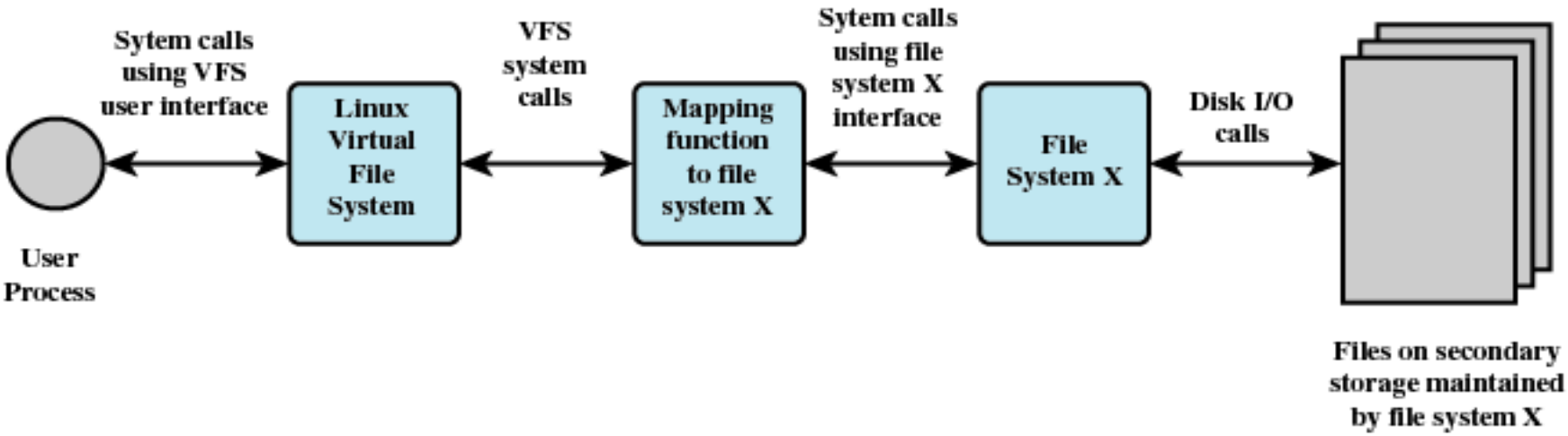


Figure 12.16 Linux Virtual File System Concept

Fuente: W. Stallings: Sistemas Operativos. 5º Ed. Pearson Prentice-Hall, 2005

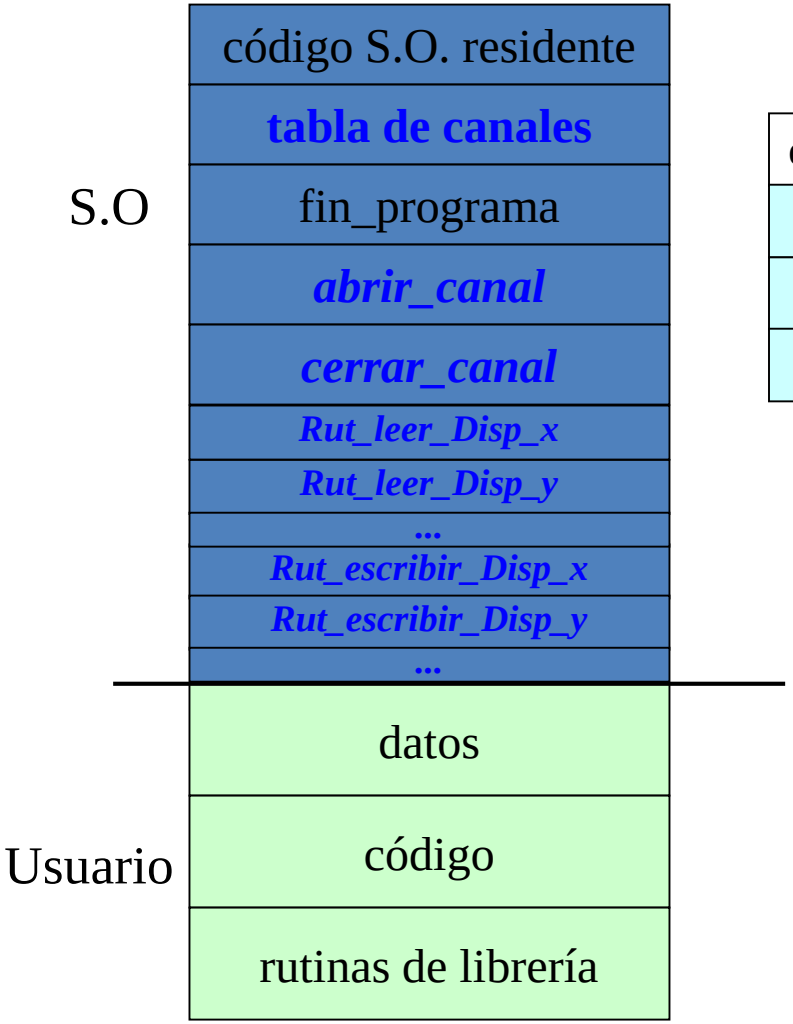
4.4 Independencia del Dispositivo

- **Problema:** a la hora de utilizar un dispositivo
  ⇒ referenciar explícitamente las rutinas del dispositivo
  ⇒ al cambiar de dispositivo es necesario cambiar el programa ⇒ necesidad de compilar de nuevo los programas.

- uso de dispositivos lógicos, **canales**

  - Concepto de ***independencia de los dispositivos***

Independencia del Dispositivo



	direcciones de rutinas de E/S			
disp. lógico	leer	escribir	posicionar	• • •
	@x	@y	@z	
• • •	@w	• • •		

Tabla de canales: utilizada para guardar la información del dispositivo físico a que alude cada uno de los dispositivos lógicos

Independencia del Dispositivo (cont.I)

rutinas de biblioteca

```
rut_lib_leer (int dis_lógico,char*vector,
              int num,int asin_sin);
{  par0=dis_lógico;
   par1=ENTRADA;
   par2=vector;
   par3=num;
   par4=asin_sin;
   llamada_sistema(ENTRADA_SALIDA)
}
```

```
rut_lib_escribir (int dis_lógico, char *vector,
                  int num, int asin_sin);

{  par0=dis_lógico;
   par1=SALIDA;
   par2=vector;
   par3=num;
   par4=asin_sin;
   llamada_sistema(ENTRADA_SALIDA)
}
```

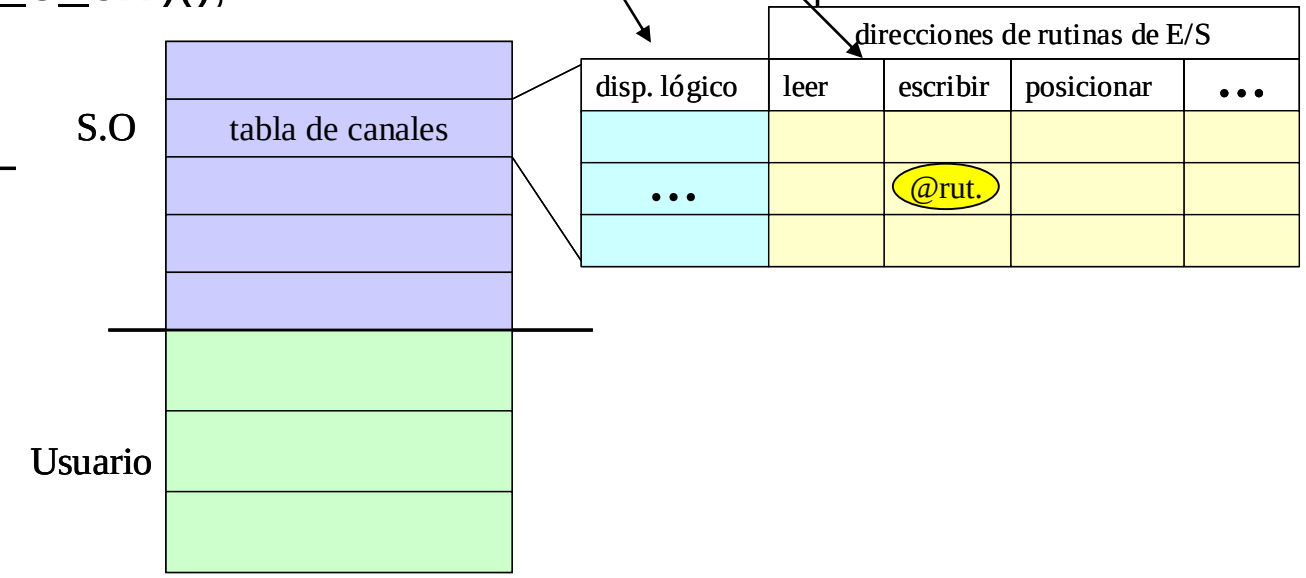
programa usuario (síncrono)

```
main ()
{  int f1, f2;
   char vec1[80], vec2[80];
   abrir (f1,"dispositivo_entrada" );
   abrir (f2,"dispositivo_salida");
   while (TRUE)
   {
      rut_lib_leer (f1,vec1,80,SINCRO);
      rut_lib_escribir (f2,vec1,80,SINCRO);
    }
    cerrar (f1);
    cerrar(f2);
}
```

Independencia del Dispositivo (cont.II)

rutina dispersora

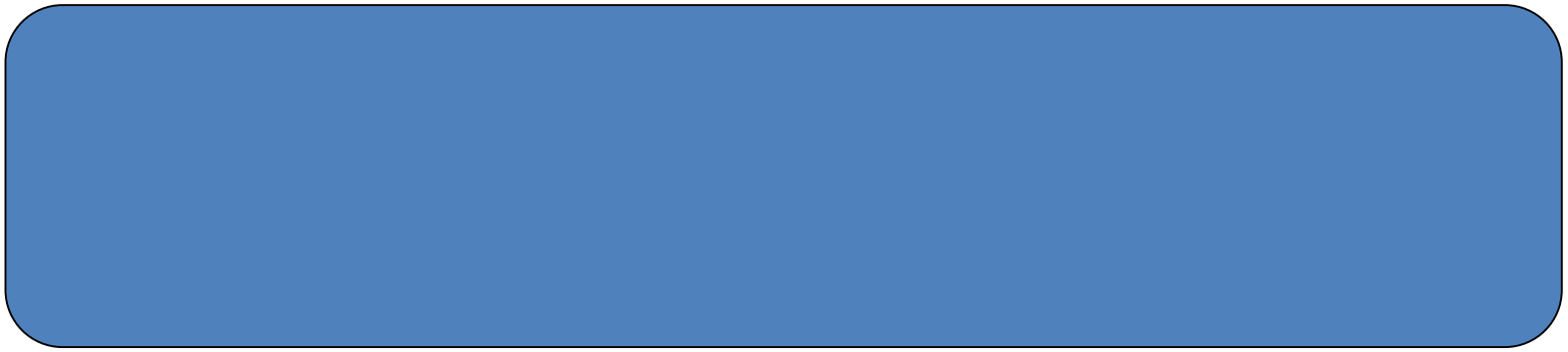
```
rut_dispersora ()
{  int (*rut_o_err)();
   rut_o_err = obten_rutina(par0, par1);
   if (!errores(DISPO, &rut_o_err))
       (*rut_o_err)();
}
```



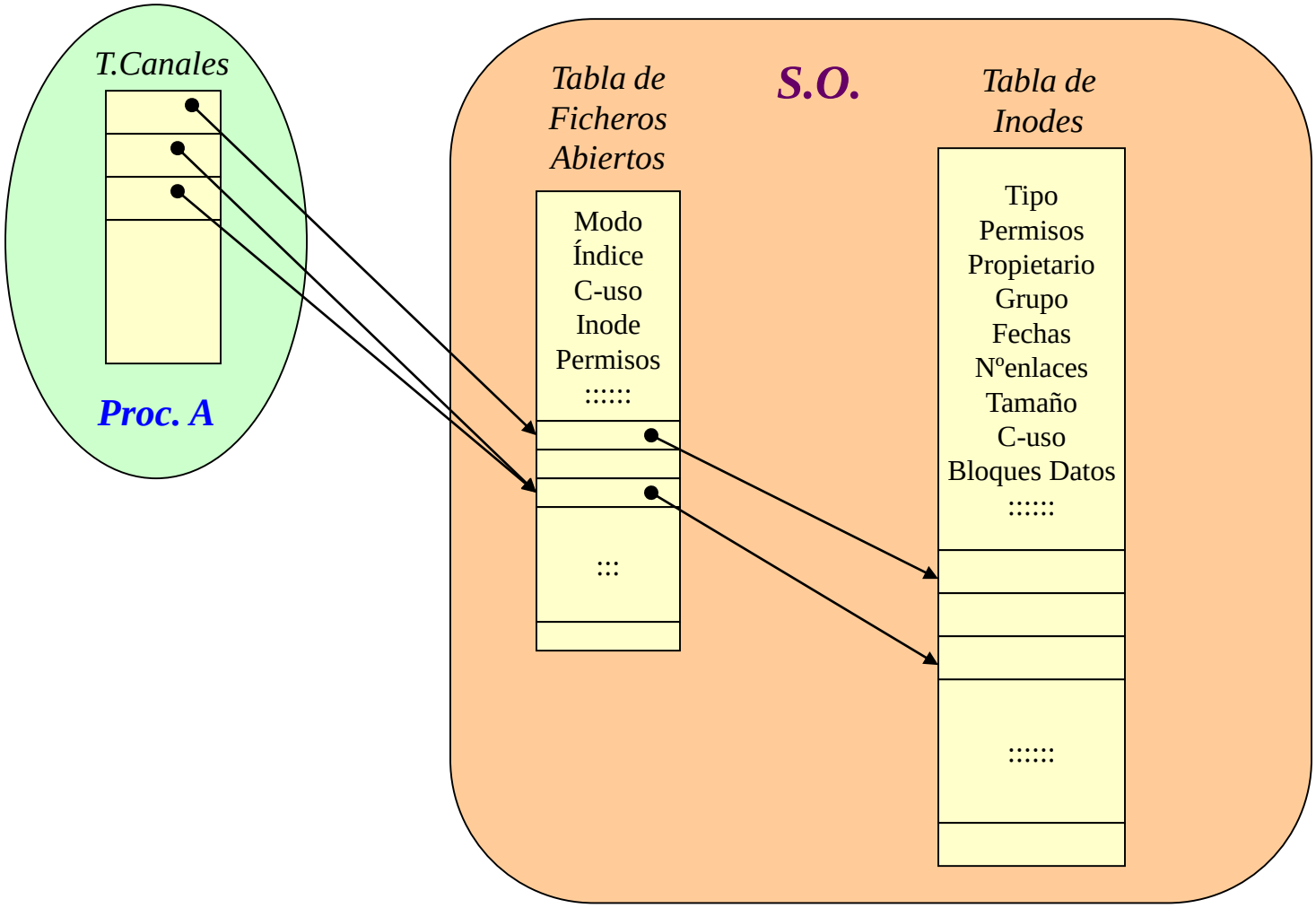
Canales preestablecidos (estándar)

- Canales estándares = dispositivos preestablecidos (entrada, salida, error). En Unix: 0, 1, 2 (***stdin***, ***stdout***, *stderr*)

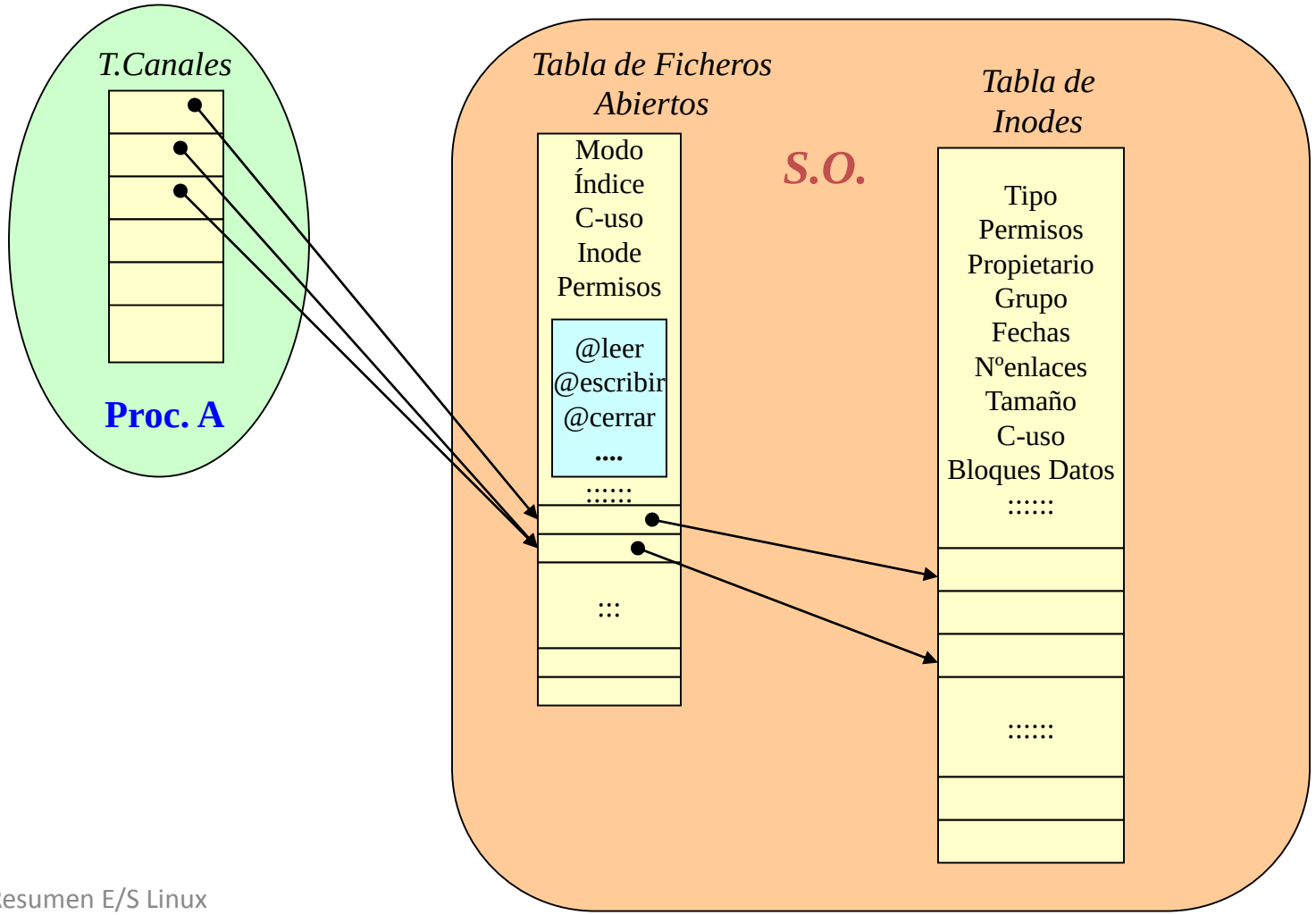
- El IC dispone de mecanismos para trabajar con canales:
  - **Redirección de canales** (**<** **>** **>&**)
    (**<<** **>>** **>>&**)



Independencia del Dispositivo en UNIX

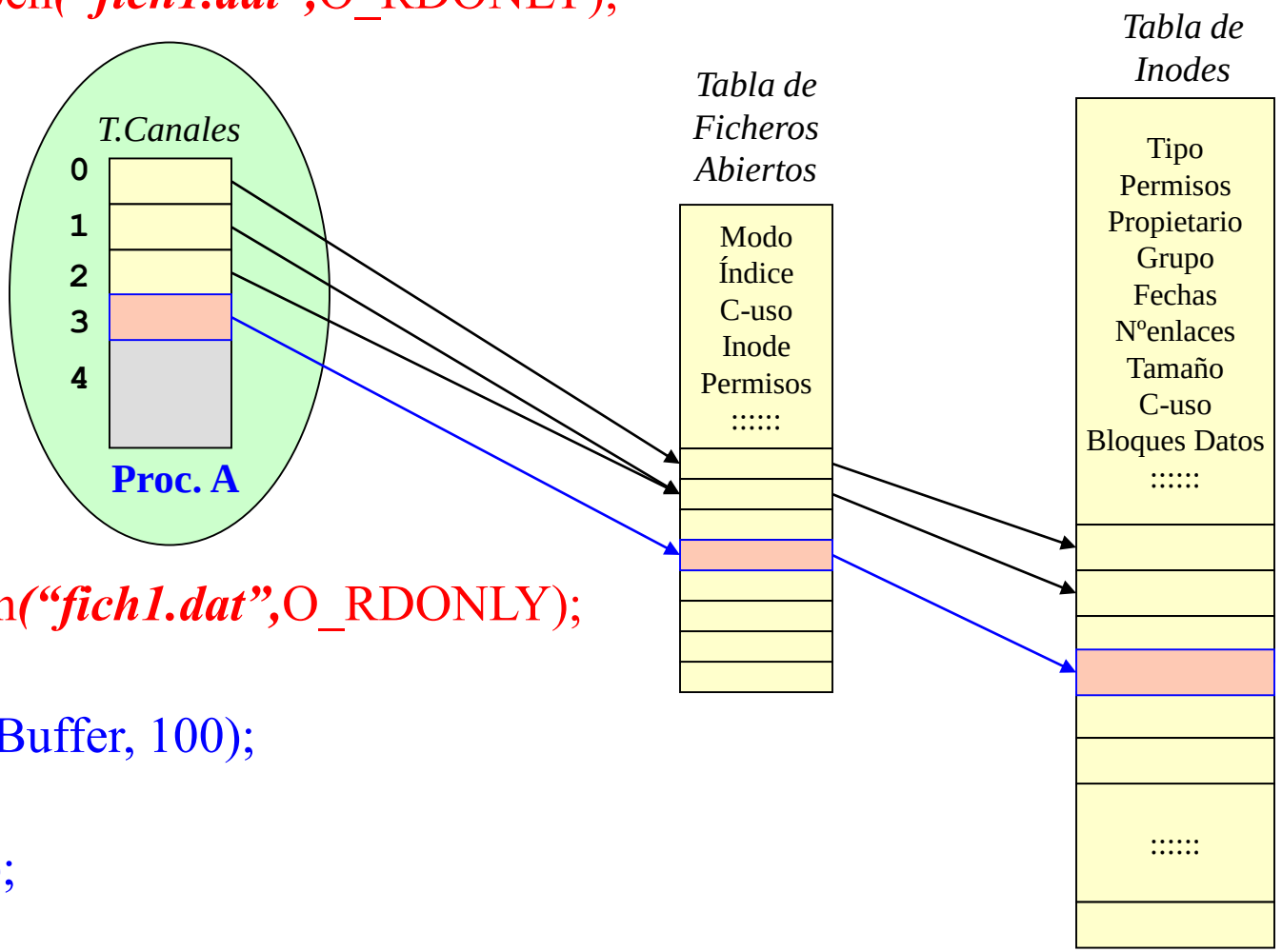


Independencia del Dispositivo en UNIX (cont.I)



Independencia del Dispositivo en UNIX (cont.II)

fd = open(***"fich1.dat"***,O_RDONLY);



fd = open(***"fich1.dat"***,O_RDONLY);

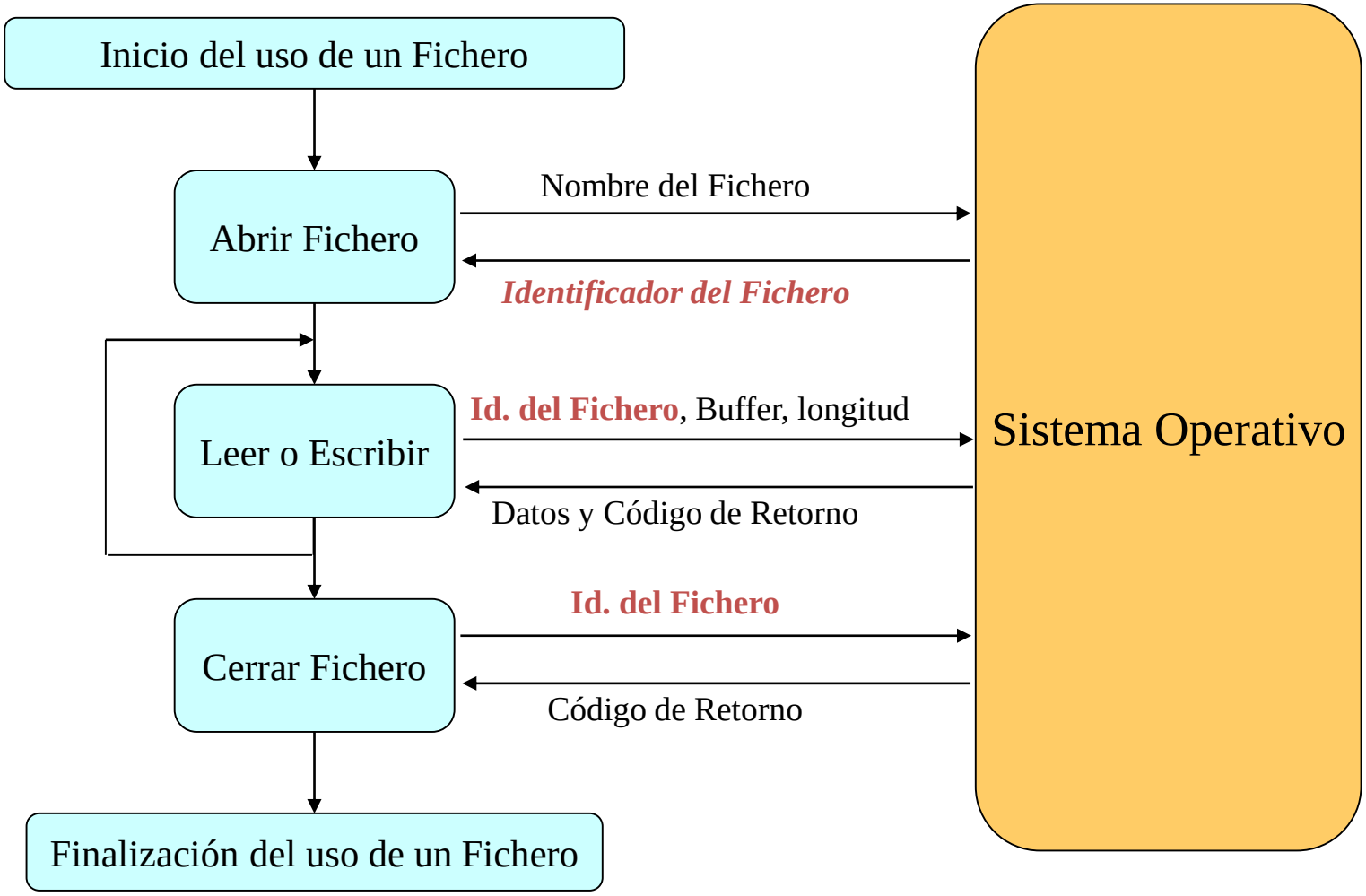
…

read(**fd**, Buffer, 100);

…

close(**fd**);

Pasos para usar un Fichero



Niveles de traducción de direcciones

Leer/Escribir
Registro, canal

Rutinas de Librería
Relativo a Reg. -> Relativo a byte

Leer/Escribir (Llamada al sistema)
byte, canal

USUARIO

Rutina Dispersora + Tabla de canales
Relativo a byte -> Bloque Relativo

Leer/Escribir
Bloque, fichero o dispositivo

Información de ubicación (I-node)
Bloque relativo -> Bloque absoluto

SISTEMA

Leer/Escribir
Bloque Absoluto, dispositivo

Características físicas del dispositivo
Bloque absoluto -> Cilindro, pista, sector

Leer/Escribir
Datos registros controlador