Fundamentos de los Sistemas Operativos (FSO)

Departamento de Informática de Sistemas y Computadoras (DISCA)
Universitat Politècnica de València

Bloque Temático 3: Sistema de Archivos y E/S
Seminario Unidad Temática 8

SUT8:
Sistema de archivos Minix

fSO

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

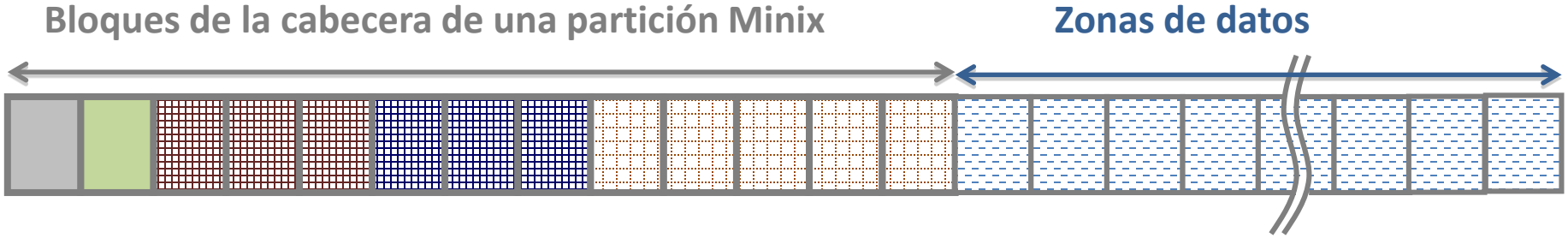
DISCA

- **Objetivos**
  - Describir la **estructura de una partición en Minix**
  - Estudiar el **nodo-i** como elemento del sistema operativo para mantener la información del archivo
  - Comprender el concepto de **mapa de bits** para la gestión de estructuras libre/ocupado
  - Ser capaz de **localizar un archivo** concreto en una estructura de directorios a partir su ruta absoluta
- **Bibliografía**
  - "Sistemas operativos: Diseño e implementación", Andrew S. Tanenbaum, Prentice Hall

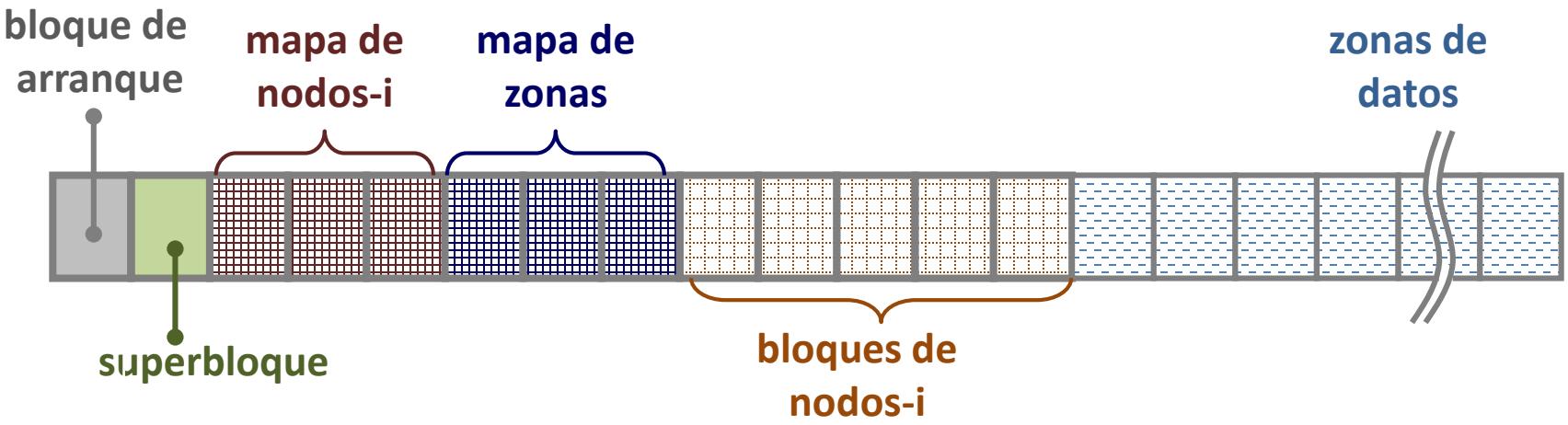
- **Estructura de una partición**
- Estructura de un nodo-i
- Entradas de directorio
- Tamaños estándar
- Ejercicios

Estructura de una partición

- **Partición Minix** se construye sobre un entramado de bloques de tamaño fijo (p.ej. 1KByte)
  - La estructura de la partición consta de:
    - La **cabecera** formada por grupos de bloques destinados a almacenar las estructuras de datos del sistema de ficheros
    - **Zonas de datos**, formada por bloques destinados a almacenar la información propia del archivo

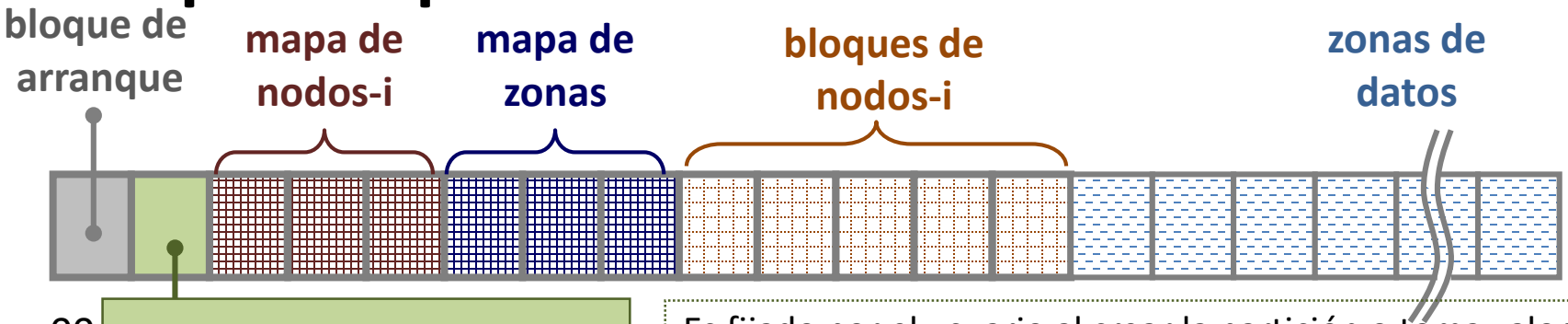


- **Partición Minix bloques de la cabecera**



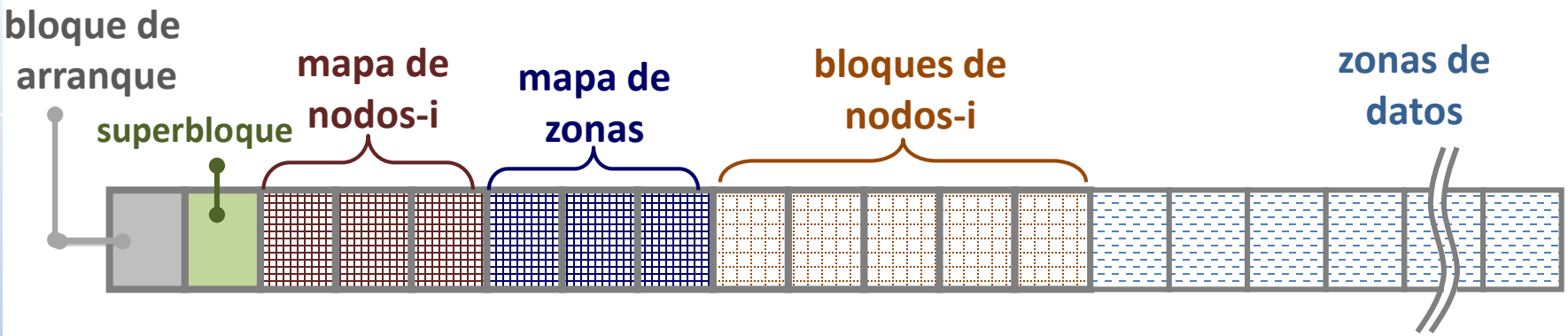
- **Bloque de arranque:** contiene el programa de arranque que carga el sistema operativo y le transfiere control
- **Superbloque:** es una estructura de datos con la descripción del sistema de archivos, indica tamaño y ubicación de cada elemento
- **Mapa bits nodos-i**: vector de bits para gestionar nodos-i libres y ocupados. Contiene un bit por cada nodo-i
- **Mapa bits zonas**: vector de bits para gestionar de zonas libres y ocupadas. Contiene un bit por cada zona
- **Bloques de nodos-i:** contienen las estructuras de datos nodos-i. El número de nodos-i depende del tamaño de la partición. El nodo-i 0 no se utiliza

- **Superbloque de Minix**



	Superbloque	
00	Número de nodos-i	Es fijado por el usuario al crear la partición o toma valores por defecto
02	Número de zonas de datos	Es fijado al crear la partición, puede ser menor que el tamaño de la partición
04	Número de bloques para el mapa de nodos-i	[Número de nodos-i / Número de bits por bloque]
06	Número de bloques para el mapa de zonas	[Número de zonas de datos / Número de bits por bloque]
08	Nº de bloque de la primera zona de datos	2 (Bloque de arranque y superbloque) + Número de bloques para el mapa de nodos-i + Número de bloques para el mapa de bloques + Número de bloques de nodos-i
0A	N (1 zona= 2^N * 1024)	1 zona = 2^N bloques El valor N se almacena en el superbloque
0C	Tamaño máximo de archivo	Tamaño máximo de archivo en bytes
10	Número mágico	Valor numérico que garantiza que esta partición contiene un sistema de archivos MINIX

- **Mapa de bits/vector de bits**



mapa de nodos-i

0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
≈							≈
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1

mapa de zonas

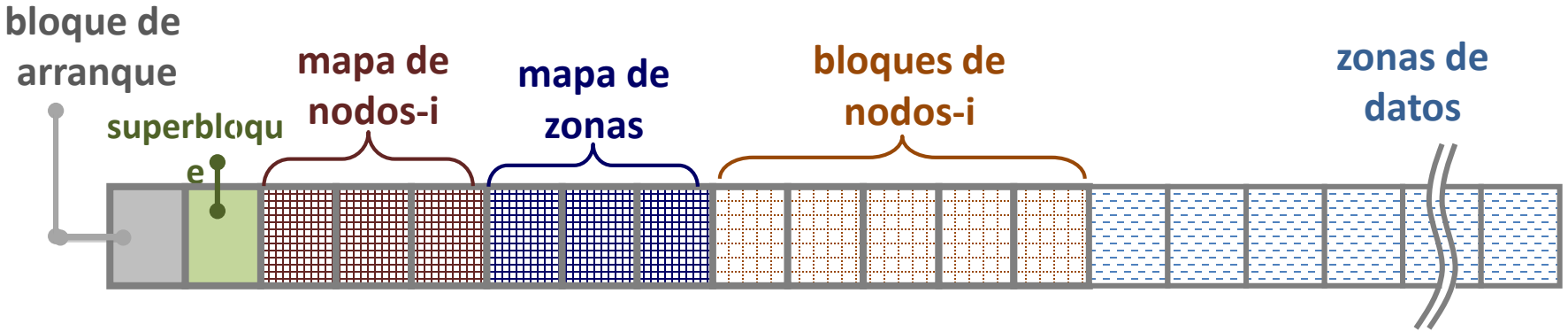
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	1	0	0	1	1	0	0
1	0	1	1	1	0	0	0
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
≈							≈
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1

bit a 0 asignado
bit a 1 libre

Cada **nodo-i** esta **representado** por **un bit** que contendrá 0 ó 1 si ya ha sido asignado o esta libre para ser asignado a un archivo

Cada **zona de datos** esta **representado** por **un bit** que contendrá 0 ó 1 si ya ha sido asignado o esta libre para ser asignado a un archivo

- **Zona de datos**

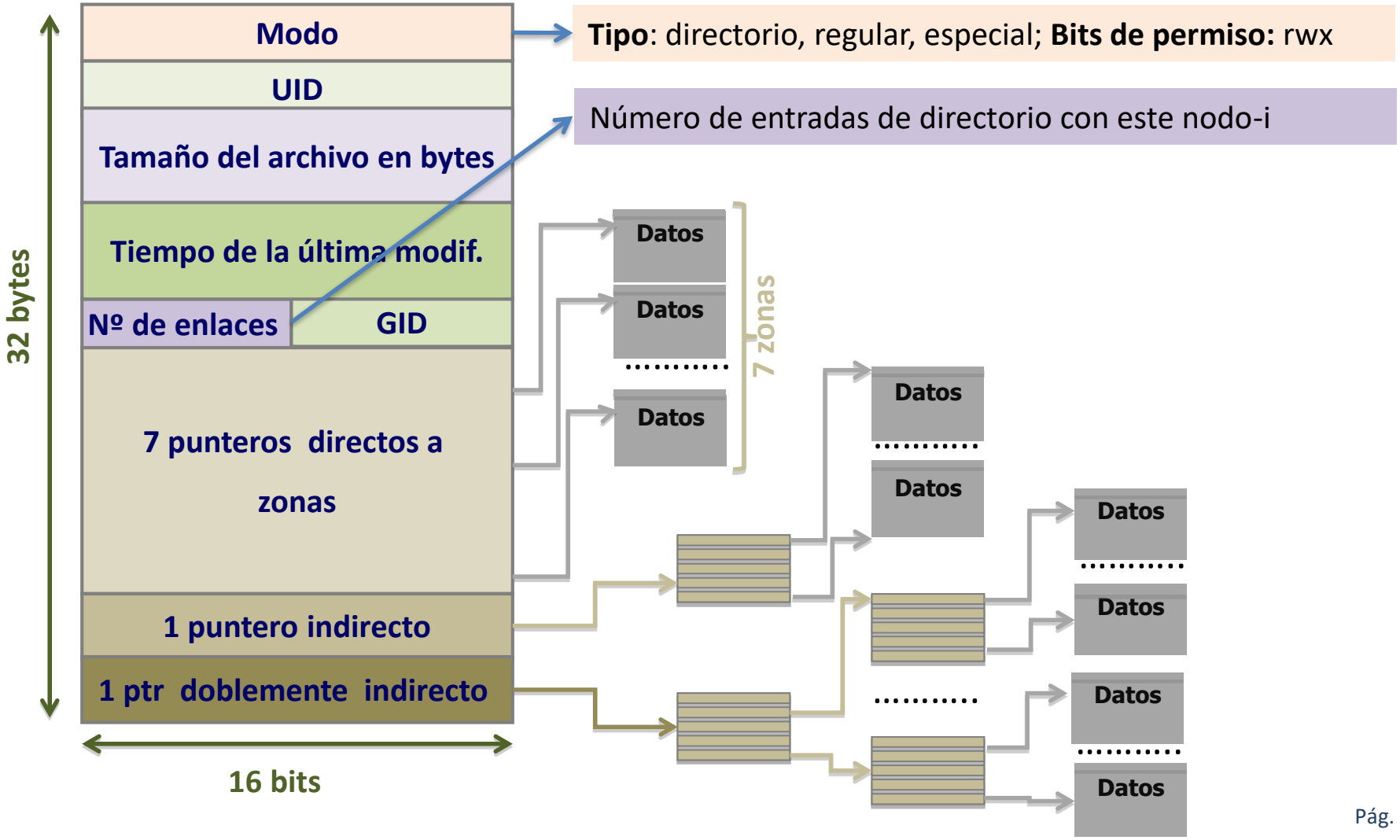


- **Zona de datos:** bloques para almacenar la información de archivos regulares, entradas de directorios y referencias a otros bloques
  - Para direccionar particiones de gran tamaño, Minix permite agrupar los bloques de datos en zonas
  - La zona es la unidad de asignación a archivo
    - **1 zona = 2^N bloques** → por defecto 1 zona = 2^0 bloques = 1bloque
    - El primer bloque de datos (valor almacenado en el superbloque) se ajusta para que coincida con el comienzo de una zona

- Estructura de una partición
- **Estructura de un nodo-i**
- Entradas de directorio
- Tamaños estándar
- Ejercicios

Estructura de un nodo-i

- **Nodo-i de Minix**
  - Estructura de datos con todos los atributos del archivo excepto el nombre
    - Cada archivo en Minix se le asigna un nodo-i
    - Asignación indexada con punteros a bloques directos, indirectos y doblemente indirectos



- **Nodo-i Minix**
  - Análisis de eficiencia
    - **Acceso aleatorio eficiente:** El número máximo de accesos a disco está limitado a 4
      - Los punteros indirectos sólo se utilizan para ficheros grandes y muy grandes (que son escasos)
      - Para los ficheros pequeños el acceso es muy eficiente
    - **Diseño elegante y fiable**: cada fichero tiene su estructura de datos separada

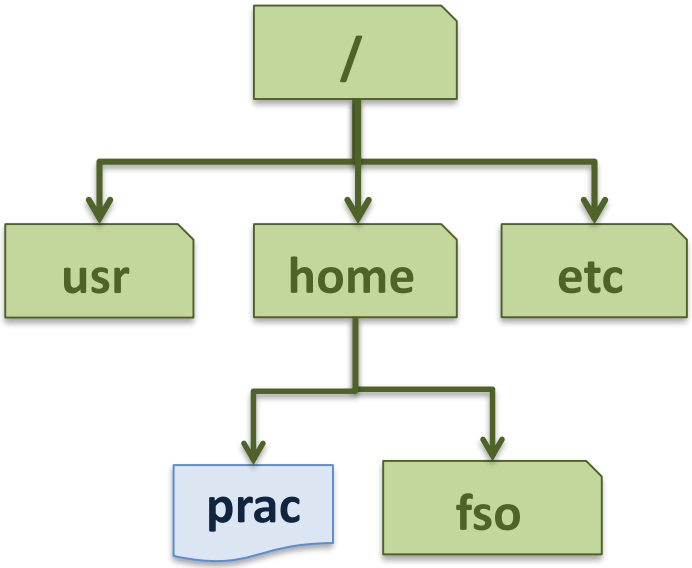
Contenido

- Estructura de una partición
- Estructura de un nodo-i
- **Entradas de directorio**
- Tamaños estándar
- Ejercicios

Entradas de directorios

- **Directorios en Minix**
  - Estructura de directorios como grafo acíclico dirigido (DAG)
  - Los directorios son archivos cuyos bytes se interpreta como registros → entradas de directorios o enlaces
- **Entrada de directorio Minix**
  - La entrada de directorio también se denominada enlace
  - Con un tamaño de 16 bytes
    - 2 bytes para el número de nodo-i
    - 14 bytes para el nombre del archivo

- **Entradas de directorio**
  - Cuando se crea un directorio, se crean **las entradas '.' y '..'** automáticamente.
  - El **nodo-i 1 describe al directorio raíz**
  - Cuando se borra una entrada se marca con el nodo-i 0



1	.
1	..
3	usr
4	home
5	etc

4	.
1	..
35	prac
84	fso

84	.
4	..

3	.
1	..

5	.
1	..

- Estructura de una partición
- Estructura de un nodo-i
- Entradas de directorio
- **Tamaños estándar**
- Ejercicios

Tamaños estándar

- Tamaños por defecto para los diferentes elementos de Minix
  - 1 Zona =2^0 bloques = 1024 bytes
  - 1 Puntero a zona o bloque = 2 bytes = 16 bits
  - 1 Entrada de directorio = 16 bytes
  - 1 Nodo-i = 32 bytes

- Estructura de una partición
- Estructura de un nodo-i
- Entradas de directorio
- Tamaños estándar
- **Ejercicios**

- **Ejercicio 1: Tamaño máximo de un archivo**

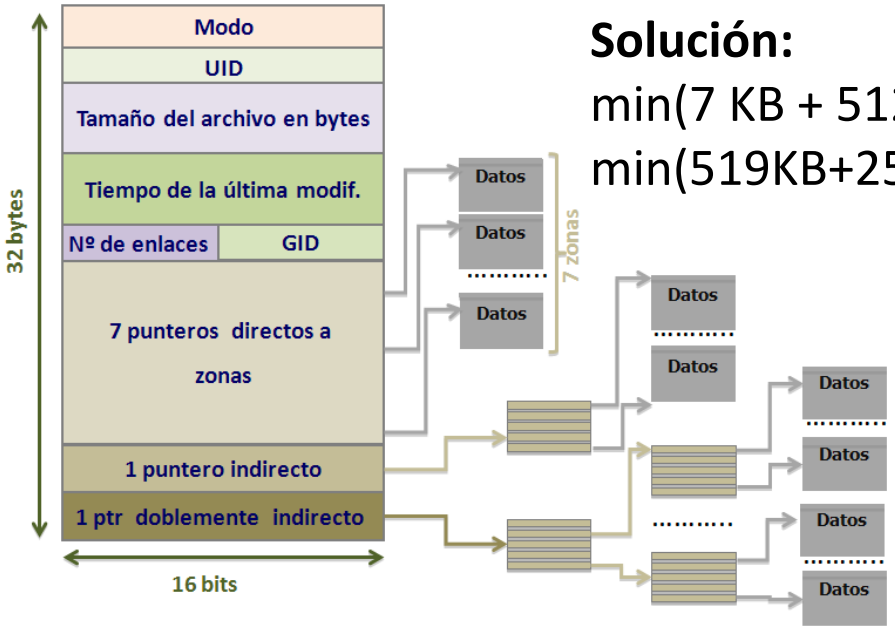
  Determine el máximo tamaño teórico (no tenga en cuenta el tamaño real del dispositivo sobre el que deba residir el archivo) de un archivo Minix. Especifique los bloques direccionados con cada tipo de puntero. Los parámetros de Minix ha considerar son:

  Punteros a zonas de datos de 16 bits

  Tamaño de bloque 1K

  1 zona = 1 bloque

  Estructura de nodo-i: 7 punteros directos,1 indirecto y 1 doble indirecto



Solución:

min(7 KB + 512 KB + 512*512*1KB, 64K*1KB) =

min(519KB+256K*1KB, 64K*1KB) = 64MB

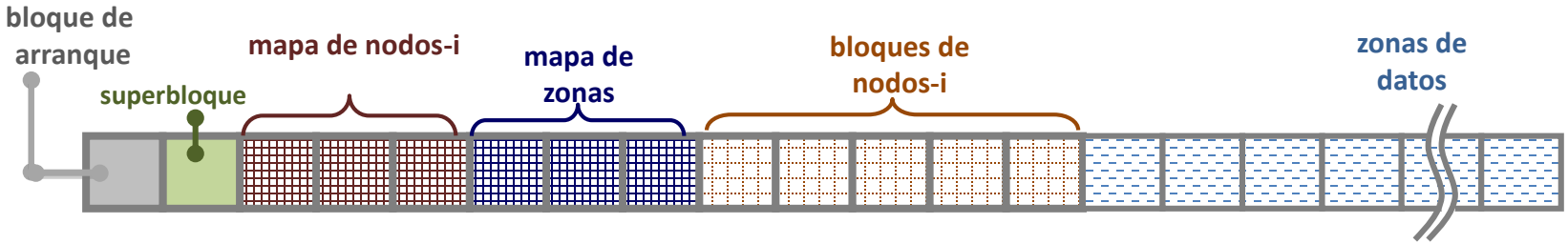
- **Ejercicio 2: Cálculo de la estructura de una partición**

  Dado un disco de 20 Mb en MINIX con los siguientes parámetros:

  - Punteros a zonas de datos de 16 bits
  - Tamaño de bloque 1K (1 zona = 1 bloque)
  - Tamaño de nodo-i de 32 bytes
  - Número máximo de nodos-i: 512

  a) Especifique claramente todas las estructuras de datos que forman el sistema de archivos y los bloques que ocupa cada una

  b) En caso de resultar dañada la estructura del mapa de bits de zonas, piense la forma de reconstruirla con la información de la que se dispone en el resto de estructuras del sistema de archivos (suponga que el resto de las estructuras están inalteradas).



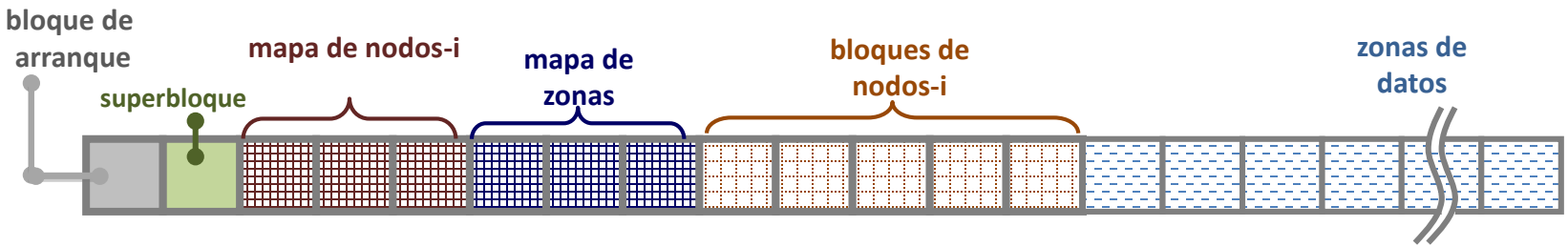
- **Ejercicio 2 (Solución): Cálculo de la estructura de una partición**

  Dado un disco **de 20 Mb e**n MINIX con los siguientes parámetros:

  - Punteros a zonas de datos de 16 bits
  - Tamaño de bloque 1K (1 zona = 1 bloque)
  - Tamaño de nodo-i de 32 bytes
  - Número máximo de nodos-i: 512

  a)Especifique claramente todas las estructuras de datos que forman el sistema de archivos y los bloques que ocupa cada una

- **Bloque de arranque = 1 Bloque**
- **Super Bloque = 1 Bloque**
- **Mapa de nodos-i → Es necesario 1 bit por cada nodo-i**
  - Numero máximo de Nodos-i =512 → 512 bits como mínimo en el Mapa de Nodos-i
  - 512 bits/1bloque = 512 Bits/ 1Kbyte = 512 bits /(8 *1024 Bits) → **1 Bloque para los Nodos-i**
- **Mapa de Zonas → Es necesario 1 bit por cada zona**
  - Tamaño del disco = 20MBytes =20MBytes/1KBytes = 20 K zonas = 20*1024 Zonas
  - Son necesario 20* 1024 bits en el mapa de zonas → (20 *1024 bits)/ 1KByte = (20 *1024 bits) / (8*1024 bits) = 20/8 =2,5 → son necesarios **3 Bloques para el mapa de Zona**s
- **Nodos-i → Son necesarios 32Bytes por cada Nodo-i**
  - 32Bytes *512Nodos-i = 16 K Bytes → **16 Bloques para los Nodos-i**



- **Ejercicio 3: Búsqueda de un archivo en un directorio**

Sea un sistema de archivos Minix, creado con los tamaños estándar cuyas entradas de directorio actualmente son la siguientes

1	.
1	..
3	usr
4	home
5	etc

4	.
1	..
35	prac
84	fso

3	.
1	..
60	ut12
61	sut12

84	.
4	..
90	map
91	listado

5	.
1	..

90	.
84	..

a) Dibuje el árbol de directorios y archivos que corresponde a dicho sistema

b) Indique de forma justificada **que números de nodos-i y cuantos bloques** es necesario acceder, si se requiere la lectura de los 128 primeros Bytes del archivo **/home/fso/listado**

c) Que información deberíamos conseguir al leer los 32 primeros Bytes del archivo **/home/fso/map**

- **Ejercicio 3 (solución): Búsqueda de un archivo en un directorio**

Sea un sistema de archivos Minix, creado con los tamaños estándar cuyas entradas de directorio actualmente son la siguientes

1	.
1	..
3	usr
4	home
5	etc

4	.
1	..
35	prac
84	fso

3	.
1	..
60	ut12
61	sut12

84	.
4	..
90	map
91	listado

5	.
1	..

90	.
84	..

b) Indique de forma justificada **que números de nodos-i y cuantos bloques** es necesario acceder, si se requiere la lectura de los 128 primeros Bytes del archivo **/home/fso/listado**

/home/fso/listado
Nodo-i 1 del raíz **/** + Nodo-i 4 **/home** + Nodo-i 84 de **/home/fso**
+ Nodo-i 91 de **/home/fso/listado**
Total 4 nodos i (1,4,84 y 91)

- **Ejercicio 3 (solución): Búsqueda de un archivo en un directorio**

1	.
1	..
3	usr
4	home
5	etc

4	.
1	..
35	prac
84	fso

3	.
1	..
60	ut12
61	sut12

84	.
4	..
90	map
91	listado

5	.
1	..

90	.
84	..

b) Indique de forma justificada **que números de nodos-i y cuantos bloques** es necesario acceder, si se requiere la lectura de los 128 primeros Bytes del archivo **/home/fso/listado**

Bloques

- En un bloque de 1024 Byte pueden ubicarse 32 nodos-i de 32 bytes
  - Por tanto, los nodos del 0 al 31 se encuentran en el mismo bloque (1er bloque de nodos-i)
  - Los nodos –i del 32 al 63 se encuentran en el segundo bloque de nodos-i
  - Los nodos-i del 64 al 95 se encuentran en el tercer bloque de nodos-i
- **Bloques necesarios a consultar**
  - El bloque de la cabecera donde esta el nodo-i 1
  - El bloque del directorio raíz que contiene las entradas de directorio
  - El bloque de la cabecera que contiene el nodo-i 4
  - El bloque del directorio home que contiene la entrada fso
  - El bloque de la cabecera que contiene el nodo-i 84
  - El bloque del directorio fso que contiene la entrada listado
  - El bloque de la cabecera que contiene el nodo-i 91
  - El bloque que contiene los primeros 128 bytes del archivo listado

- **Ejercicio 3: Búsqueda de un archivo en un directorio**

Sea un sistema de archivos Minix, creado con los tamaños estándar cuyas entradas de directorio actualmente son la siguientes

1	.
1	..
3	usr
4	home
5	etc

4	.
1	..
35	prac
84	fso

3	.
1	..
60	ut12
61	sut12

84	.
4	..
90	map
91	listado

5	.
1	..

90	.
84	..

c) Que información deberíamos conseguir al leer los 32 primeros Bytes del archivo **/home/fso/map**

Dos entradas de directorio.
Dos registros con número de nodo-1 y nombre

- 90 .
- 84 ..

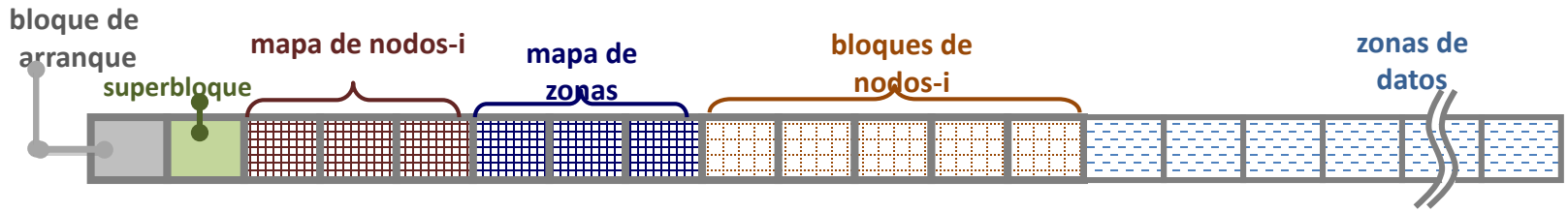
Ejercicio 4:

- Un disco con una capacidad de 8GB, se formatea con una versión de Minix. Los tamaños usados en el formateo son:
  - Tamaño de bloque = 2KBytes . Tamaño de Zona = 2^0 bloques = 1 Bloque
  - Los punteros a Zona son de 32bits=4Bytes
  - El tamaño del nodo-i es de 64 Bytes (7 punteros directos, 1 indirecto, 1 doble indirecto) .
  - Cada entrada de directorio ocupa 32 Bytes.
  - Al formatear se ha reservado espacio en la cabecera para 4.096 nodos-i
- Se pide:

  a) Calcule el número de bloques que ocupa cada uno de los elementos de la cabecera: Mapa de bits nodos-i, Mapa de bits Zonas y Nodos-i.

  b) Calcule el bloque que corresponde a la primera Zona de datos y el número de Zonas de datos.

  c) Suponga que este disco almacena un único directorio, el directorio raíz que contiene 10 archivos regulares,
  - c1) Indique el número de zonas de datos que ocupa el directorio raíz
  - c2) Suponga además que cada uno de los archivos regulares contiene una información que ocupa 50KBytes e indique de forma justificada el número de zonas de datos ocupadas para este caso, tenga en cuenta tanto los datos como los metadatos del archivo.



Ejercicio 4 (solución):

- Un disco con una capacidad de 8GB, se formatea con una versión de Minix. Los tamaños usados en el formateo son:
  - Tamaño de bloque = 2KBytes . Tamaño de Zona = 2^0 bloques = 1 Bloque
  - Los punteros a Zona son de 32bits=4Bytes
  - El tamaño del nodo-i es de 64 Bytes (7 punteros directos, 1 indirecto, 1 doble indirecto) .
  - Cada entrada de directorio ocupa 32 Bytes.
  - Al formatear se ha reservado espacio en la cabecera para 4.096 nodos-i
- Se pide:

  a) Calcule el número de bloques que ocupa cada uno de los elementos de la cabecera: Mapa de bits nodos-i, Mapa de bits Zonas y Nodos-i.

a)

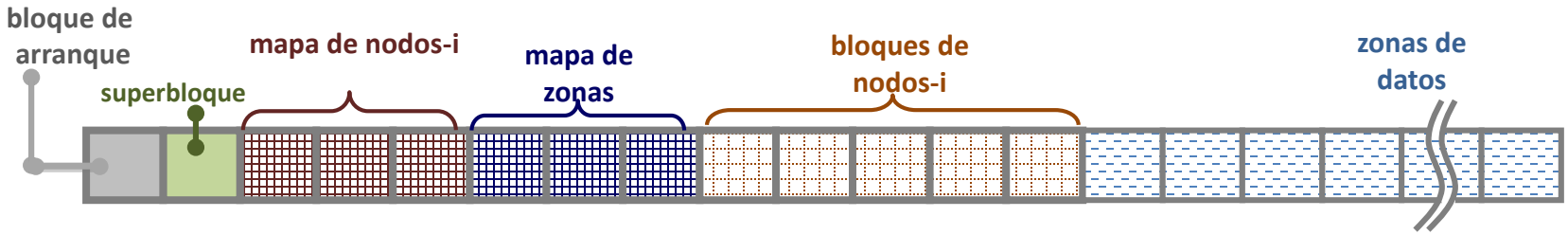
Tamaño de Bloque=2KBytes → 2*8* 1024 bits

4096 nodos-i →Mapa de nodos-i = 4096 bits → 1 bloque

Nodos-i → 4096 nodos-i * 64Bytes = 2^{12} * 2^6 =2^{18} Bytes → 2^{18} /2KBytes =2^7=128 bloques

Disco 8GBytes y Zonas = 2KBytes → 8GBytes/2KBytes= 2^{22} zonas

Mapa de zonas con 2^{22} bits → 2^{22}/2*8*1024 =2^8 bloques= 256 bloques

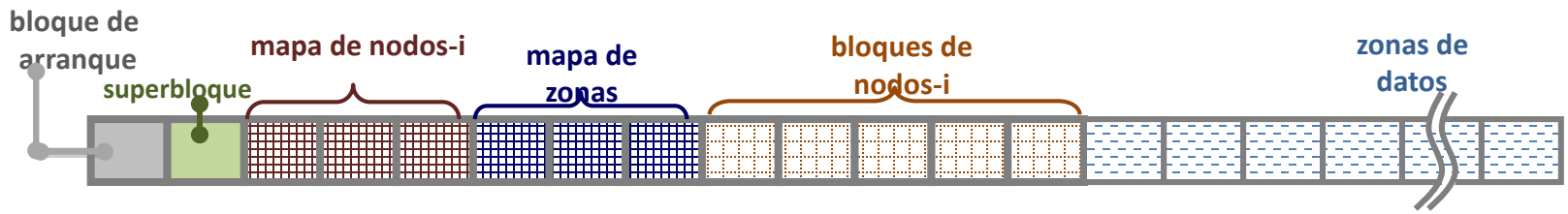


Ejercicio 4:

- Un disco con una capacidad de 8GB, se formatea con una versión de Minix. Los tamaños usados en el formateo son:
  - Tamaño de bloque = 2KBytes . Tamaño de Zona = 2^0 bloques = 1 Bloque
  - Los punteros a Zona son de 32bits=4Bytes
  - El tamaño del nodo-i es de 64 Bytes (7 punteros directos, 1 indirecto, 1 doble indirecto) .
  - Cada entrada de directorio ocupa 32 Bytes.
  - Al formatear se ha reservado espacio en la cabecera para 4.096 nodos-i
- Se pide:

  b) Calcule el bloque que corresponde a la primera Zona de datos y el número de Zonas de datos.

b)
1 Arranque + 1 SuperBloque + 1 Mapa nodos-i + 256 Mapa de Zonas +128 Nodos-i = 387
Bloque 387 será el primer bloque de datos = Primera zona de datos

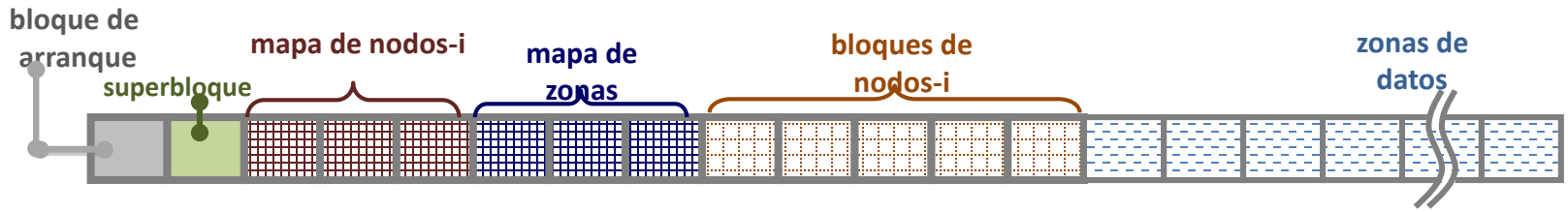


Ejercicio 4:

- Un disco con una capacidad de 8GB, se formatea con una versión de Minix. Los tamaños usados en el formateo son:
  - Tamaño de bloque = 2KBytes . Tamaño de Zona = 2^0 bloques = 1 Bloque
  - Los punteros a Zona son de 32bits = 4Bytes
  - El tamaño del nodo-i es de 64 Bytes (7 punteros directos, 1 indirecto, 1 doble indirecto) .
  - Cada entrada de directorio ocupa 32 Bytes.
  - Al formatear se ha reservado espacio en la cabecera para 4.096 nodos-i
- Se pide:

  c) Suponga que este disco almacena un único directorio, el directorio raíz que contiene 10 archivos regulares,
  - c1) Indique el número de zonas de datos que ocupa el directorio raíz

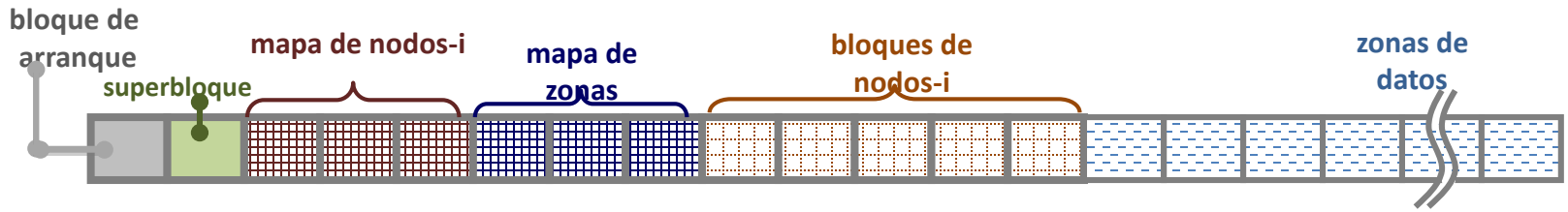
c1)
Directorio raíz contiene 10 entradas de archivos regulares + . y .. = 12 entradas → 12 entradas *64 Bytes =1280 Bytes → 1 zona



Ejercicio 4:

- Un disco con una capacidad de 8GB, se formatea con una versión de Minix. Los tamaños usados en el formateo son:
  - Tamaño de bloque = 2KBytes . Tamaño de Zona = 2^0 bloques = 1 Bloque
  - Los punteros a Zona son de 32bits=4Bytes
  - El tamaño del nodo-i es de 64 Bytes (7 punteros directos, 1 indirecto, 1 doble indirecto) .
  - Cada entrada de directorio ocupa 32 Bytes.
  - Al formatear se ha reservado espacio en la cabecera para 4.096 nodos-i
- c) Suponga que este disco almacena un único directorio, el directorio raíz que contiene 10 archivos regulares,
  - c1) Indique el número de zonas de datos que ocupa el directorio raíz
  - c2) Suponga además que cada uno de los archivos regulares contiene una información que ocupa 50KBytes e indique de forma justificada el número de zonas de datos ocupadas para este caso, tenga en cuenta tanto los datos como los metadatos del archivo.

50KBytes → 25 zonas de datos → Necesita 25 punteros a zona → 7 punteros directos + 1 zona que contiene 18 punteros → Total 26 zonas para cada archivos
26 zonas *10 archivos =260 zonas para archivos regular
260 zonas + 1 zona del raíz = 261 zonas



Ejercicio-5

Dado el siguiente código en el cual se generan al menos tres procesos P1, P2 y P3:

```
...
pipe(fd);/*pipe1*/
pipe(fd2);/*pipe2*/
if(fork() != 0){
  /***Proceso P1 ***/
  dup2(fd[1],STDOUT_FILENO);
  close(fd[0]); close(fd[1]);
  dup2(fd2[0],STDIN_FILENO);
  close(fd2[0]);
close(fd2[1]);
  /*tabla proceso P1*/
}else{
  /***Proceso P2 ***/
  dup2(fd[0],STDIN_FILENO);
  close(fd[0]); close(fd[1]);
```

```
  pipe(fd); /*pipe3*/
  if(fork() != 0){
    close(fd2[0]);
    close(fd2[1]);
    dup2(fd[1],STDOUT_FILENO);
    close(fd[0]);
    close(fd[1]);
    /*tabla proceso P2*/
  }else{
     /***Proceso P3 ***/
     dup2(fd[0],STDIN_FILENO);
     close(fd[0]);
     close(fd[1]);
     dup2(fd2[1],STDOUT_FILENO);
     close(fd2[0]);
     close(fd2[1]);
    /*tabla proceso P3*/
  }
 }
...
```

a) Indique el contenido de las tablas de descriptores de archivo para los procesos P1, P2 y P3, en los puntos del código marcados como /*tabla proceso Pi*/.

b) Determine el parentesco que existe entre P1, P2 y P3 así como el esquema de redirecciones resultante de ejecutar el código.

Ejercicio-5 (solución)

Dado el siguiente código en el cual se generan al menos tres procesos P1, P2 y P3:

```
...
pipe(fd);/*pipe1*/
pipe(fd2);/*pipe2*/
if(fork() != 0){
  /***Proceso P1 ***/
  dup2(fd[1],STDOUT_FILENO);
  close(fd[0]); close(fd[1]);
  dup2(fd2[0],STDIN_FILENO);
  close(fd2[0]);
close(fd2[1]);
  /*tabla proceso P1*/
}else{
  /***Proceso P2 ***/
  dup2(fd[0],STDIN_FILENO);
  close(fd[0]); close(fd[1]);
```

```
  pipe(fd); /*pipe3*/
  if(fork() != 0){
    close(fd2[0]);
    close(fd2[1]);
    dup2(fd[1],STDOUT_FILENO);
    close(fd[0]);
    close(fd[1]);
    /*tabla proceso P2*/
  }else{
     /***Proceso P3 ***/
     dup2(fd[0],STDIN_FILENO);
     close(fd[0]);
     close(fd[1]);
     dup2(fd2[1],STDOUT_FILENO);
     close(fd2[0]);
     close(fd2[1]);
    /*tabla proceso P3*/
  }
}
```

Proceso P1	
0	fd2[0]
1	fd[1]
2	stderr

/*pipe2*/
/*pipe1*/

Proceso P2	
0	fd[0]
1	fd[1]
2	stderr

/*pipe1*/
/*pipe3*/

Proceso P3	
0	fd[0]
1	fd2[1]
2	stderr

/*pipe3*/
/*pipe2*/

Ejercicio-5 (solución)

Dado el siguiente código en el cual se generan al menos tres procesos P1, P2 y P3:

```
...
pipe(fd);/*pipe1*/
pipe(fd2);/*pipe2*/
if(fork() != 0){
  /***Proceso P1 ***/
  dup2(fd[1],STDOUT_FILENO);
  close(fd[0]); close(fd[1]);
  dup2(fd2[0],STDIN_FILENO);
  close(fd2[0]);
close(fd2[1]);
  /*tabla proceso P1*/
}else{
  /***Proceso P2 ***/
  dup2(fd[0],STDIN_FILENO);
  close(fd[0]); close(fd[1]);
  pipe(fd); /*pipe3*/
  if(fork() != 0){
    close(fd2[0]);
    close(fd2[1]);
    dup2(fd[1],STDOUT_FILENO);
    close(fd[0]);
    close(fd[1]);
    /*tabla proceso P2*/
  }else{
     /***Proceso P3 ***/
     dup2(fd[0],STDIN_FILENO);
     close(fd[0]);
     close(fd[1]);
     dup2(fd2[1],STDOUT_FILENO);
     close(fd2[0]);
     close(fd2[1]);
    /*tabla proceso P3*/
  }
 }
...
```

El proceso P1 es el padre de P2, P2 es el padre de P3

Se establece un anillo de comunicación entre los tres procesos utilizando tres tubos