Sistemi Operativi e Reti di Calcolatori (SOReCa)

Corso di Laurea in *Ingegneria Informatica e Automatica (BIAR)*
Terzo Anno | Primo Semestre
A.A. 2024/2025

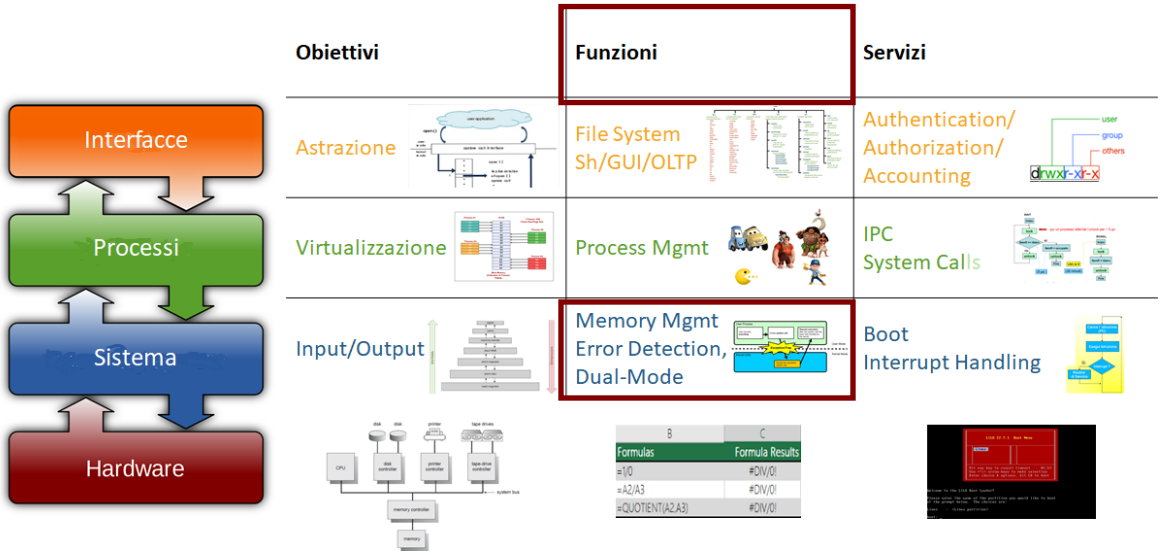
Memoria di massa e File system

DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI

SAPIENZA
UNIVERSITÀ DI ROMA

Sistemi operativi (3 CFU)

- Il sistema operativo
- Concorrenza e sincronizzazione
- Deadlock
- Inter-process communication (IPC)
- Scheduling
- Memoria centrale e virtuale
- Memoria di massa e File system
- Sicurezza informatica



Lezioni: Settembre - Ottobre

Dipartimento di Ingegneria informatica
automatica e gestionale Antonio Ruberti

Sapienza
Università di Roma

Input/Output

Comunicazione con i dispositivi periferici (!= CPU | RAM)

Operating Systems: Input/Output

Finalità

Per essere elaborate, le informazioni in Input devono essere portate in memoria ← **Dispositivi**

Interfaccia:

Dispositivi ↔ (**memoria**, **CPU**):

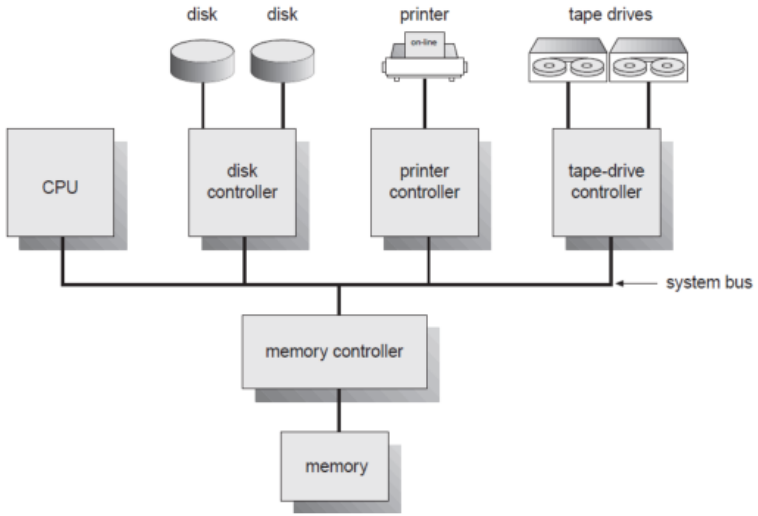
- ➔ **Inviare comandi**
- ➔ **Gestire gli Interrupts**
- ➔ **Amministrare gli errori**

Buffer: dispositivo-supporto che separa due dispositivi con velocità molto differenti (>> 10):

- **Carica** dal dispositivo **lento**
- **Fornisce** i (dopo aver preso **tutto**) i dati **a velocità** ben maggiore al dispositivo più veloce

Operating Systems: Obiettivi Funzioni Servizi

Input/Output



Tecniche di I/O:

buffer locale dei controller ↔ **memoria**

1. I/O programmato
2. I/O gestito con interrupt
3. I/O con DMA
4. I/O con canali (processore separato)

Rendere il sistema quanto meno impattato possibile dalla lentezza dei dispositivi periferici.

Dipartimento di Ingegneria Informatica Automatica e Gestionale Antonio Ruberti

Sapienza Università di Roma

Operating Systems: Input/Output

Modalità

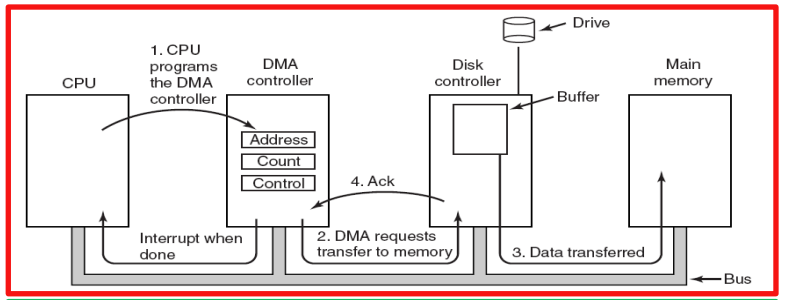
I/O: implementato in modi svariati, a seconda dell'HW sottostante e del reale utilizzo

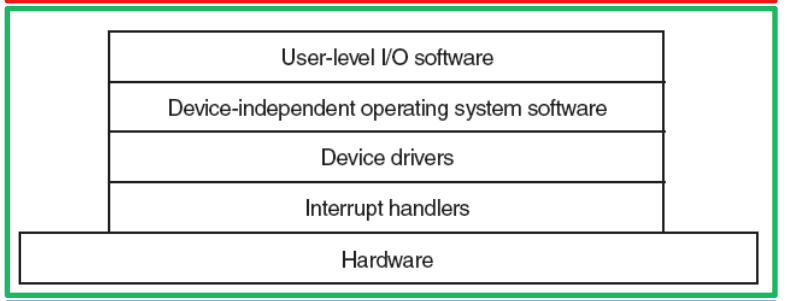
1. **I/O Hardware**: **Dispositivi di I/O**, **Memory Mapped**, **DMA**, **Interrupt**
   Principi e criteri per l'interfaccia con l'HW.

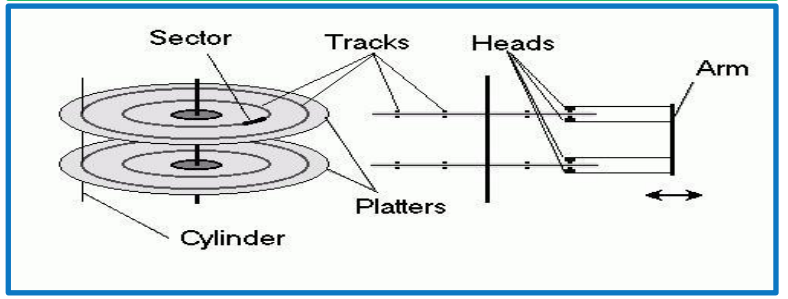
2. **I/O Software**: **Programmed I/O**, **Interrupt**, **DMA**, **Gerarchia (Layers)**
   Implementazione delle richieste di I/O nei programmi.

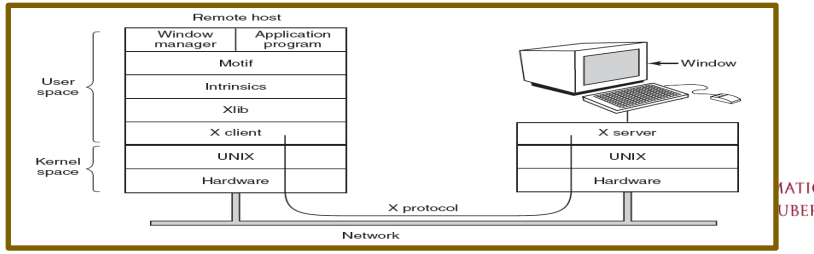
3. **Dischi**: **HDD**, **RAID**, **Formatting**, **Arm Scheduling**
   gestire i dischi.

4. **Others**: **Terminals**
   Interazione con altre periferiche.









Operating Systems: Input/Output

I/O Hardware: Dispositivi

Device	Data rate
Keyboard	10 bytes/sec
Mouse	100 bytes/sec
56K modem	7 KB/sec
Scanner	400 KB/sec
Digital camcorder	3.5 MB/sec
802.11g Wireless	6.75 MB/sec
52x CD-ROM	7.8 MB/sec
Fast Ethernet	12.5 MB/sec
Compact flash card	40 MB/sec
FireWire (IEEE 1394)	50 MB/sec
USB 2.0	60 MB/sec
SONET OC-12 network	78 MB/sec
SCSI Ultra 2 disk	80 MB/sec
Gigabit Ethernet	125 MB/sec
SATA disk drive	300 MB/sec
Ultrium tape	320 MB/sec
PCI bus	528 MB/sec

Character Device: inviano ed accettano flussi (stream) di caratteri (byte):

- ➔ **Caratteri non Indirizzabili**
- ➔ **No Operazione di Ricerca (seek)**
- ➔ **Solitamente in /dev/input**

Block Device: inviano ed accettano blocchi (block) di caratteri (byte):

- ➔ **Blocchi Indirizzabili singolarmente**
- ➔ **Operazione di Posizionament e Ricerca (seek)**
- ➔ **Dispositivi di Massa (es. dischi: /dev/sda, ROM: /dev/sr0, etc)**

Tutti i dispositivi (sia Char, sia Block) dispongono di
Device Controller: interfaccia tra sistema e dispositivo

- Acceduto come insieme di registri
- Traduce il formato delle informazioni da e verso il dispositivo
- Istruisce il buffer

Operating Systems: Input/Output

I/O Hardware: esempio lista (lsdev)

- **lsdev**: List all devices.

```
root@unixmantra:/ #
root@unixmantra:/ #lsdev
L2cache0    Available            L2 Cache
cd0         Available            Virtual SCSI Optical Served by VIO Server
cd1         Available            Virtual SCSI Optical Served by VIO Server
cd2         Defined   01-08-00   IDE DVD-RAM Drive
cluster0    Available            Cluster Node
datavg      Defined              Volume group
en0         Available            Standard Ethernet Network Interface
ent0        Available            Virtual I/O Ethernet Adapter (l-lan)
et0         Defined              IEEE 802.3 Ethernet Network Interface
fslv00      Defined              Logical volume
hd1         Defined              Logical volume
hd2         Defined              Logical volume
hd3         Defined              Logical volume
hd4         Defined              Logical volume
hd5         Defined              Logical volume
hd6         Defined              Logical volume
hd8         Defined              Logical volume
hd10opt     Defined              Logical volume
hd11admin   Defined              Logical volume
hd9var      Defined              Logical volume
hdisk0      Available            Virtual SCSI Disk Drive
hdisk1      Available            Virtual SCSI Disk Drive
hdisk2      Available            Virtual SCSI Disk Drive
hdisk3      Available            Virtual SCSI Disk Drive
hdisk4      Available            Virtual SCSI Disk Drive
ide0        Defined   01-08      ATA/IDE Controller Device
inet0       Available            Internet Network Extension
iocp0       Defined              I/O Completion Ports
iscsi0      Available            iSCSI Protocol Device
livedump    Defined              Logical volume
lo0         Available            Loopback Network Interface
```

Operating Systems: Input/Output

I/O Hardware: esempio lspci

- **lspci**: List PCI devices Peripheral Component Interconnect (personal computer bus).

ls: List files in the file system.

```
jason@mint-vm:~$ lspci -vvv
00:00.0 Host bridge: Intel Corporation 440BX/ZX/DX - 82443BX/ZX/DX Host bridge (AGP disabled) (rev 03)
        Control: I/O- Mem+ BusMaster+ SpecCycle- MemWINV- VGASnoop- ParErr- Stepping- SERR- FastB2B- DisINTx-
        Status: Cap- 66MHz- UDF- FastB2B- ParErr- DEVSEL=medium >TAbort- <TAbort- <MAbort- >SERR- <PERR- INTx-
        Latency: 0

00:07.0 ISA bridge: Intel Corporation 82371AB/EB/MB PIIX4 ISA (rev 01)
        Subsystem: Microsoft Corporation 82371AB/EB/MB PIIX4 ISA
        Control: I/O+ Mem+ BusMaster+ SpecCycle- MemWINV- VGASnoop- ParErr- Stepping- SERR- FastB2B- DisINTx-
        Status: Cap- 66MHz- UDF- FastB2B- ParErr- DEVSEL=medium >TAbort- <TAbort- <MAbort- >SERR- <PERR- INTx-
        Latency: 0

00:07.1 IDE interface: Intel Corporation 82371AB/EB/MB PIIX4 IDE (rev 01) (prog-if 80 [ISA Compatibility mode-only
 controller, supports bus mastering])
        Control: I/O+ Mem- BusMaster+ SpecCycle- MemWINV- VGASnoop- ParErr- Stepping- SERR- FastB2B- DisINTx-
        Status: Cap- 66MHz- UDF- FastB2B+ ParErr- DEVSEL=medium >TAbort- <TAbort- <MAbort- >SERR- <PERR- INTx-
        Latency: 0
        Region 0: [virtual] Memory at 000001f0 (32-bit, non-prefetchable) [size=8]
        Region 1: [virtual] Memory at 000003f0 (type 3, non-prefetchable)
        Region 2: [virtual] Memory at 00000170 (32-bit, non-prefetchable) [size=8]
        Region 3: [virtual] Memory at 00000370 (type 3, non-prefetchable)
        Region 4: I/O ports at ffa0 [size=16]
        Kernel driver in use: ata_piix
        Kernel modules: pata_acpi

00:07.3 Bridge: Intel Corporation 82371AB/EB/MB PIIX4 ACPI (rev 02)
        Control: I/O+ Mem- BusMaster- SpecCycle- MemWINV- VGASnoop- ParErr- Stepping- SERR- FastB2B- DisINTx-
        Status: Cap- 66MHz- UDF- FastB2B+ ParErr- DEVSEL=medium >TAbort- <TAbort- <MAbort- >SERR- <PERR- INTx-
        Interrupt: pin A routed to IRQ 9
        Kernel modules: i2c_piix4

00:08.0 VGA compatible controller: Microsoft Corporation Hyper-V virtual VGA (prog-if 00 [VGA controller])
        Control: I/O+ Mem+ BusMaster+ SpecCycle+ MemWINV+ VGASnoop- ParErr- Stepping- SERR+ FastB2B- DisINTx-
        Status: Cap- 66MHz- UDF- FastB2B- ParErr- DEVSEL=fast >TAbort- <TAbort- <MAbort- >SERR- <PERR- INTx-
        Latency: 0
        Interrupt: pin A routed to IRQ 11
        Region 0: Memory at f8000000 (32-bit, non-prefetchable) [size=64M]
        [virtual] Expansion ROM at 000c0000 [disabled] [size=128K]
```

Operating Systems: Input/Output

I/O Hardware: lista di Block Device, esempio

- **lsblk**: List block devices (for example, the drives).

Visualizza le informazioni:

- Name
- Major:Minor device number
- Is it removable
- Size
- Is it read-only
- Is it a disk or a partition
- Where is the partition mounted

```
root@ubuntu16:~# ip addr |grep 'inet 10'
    inet 10.101.26.88/22 brd 10.101.27.255 scope global enp1s0f0
root@ubuntu16:~# lsblk
NAME    MAJ:MIN RM   SIZE RO TYPE  MOUNTPOINT
sda       8:0    0 139.8G  0 disk
├─sda1    8:1    0   476M  0 part
│ └─md0   9:0    0 475.7M  0 raid1 /boot
├─sda2    8:2    0   1.9G  0 part  [SWAP]
└─sda3    8:3    0 137.4G  0 part
  └─md1   9:1    0 137.3G  0 raid1 /
sdb       8:16   0 139.8G  0 disk
├─sdb1    8:17   0   476M  0 part
│ └─md0   9:0    0 475.7M  0 raid1 /boot
├─sdb2    8:18   0   1.9G  0 part  [SWAP]
└─sdb3    8:19   0 137.4G  0 part
  └─md1   9:1    0 137.3G  0 raid1 /
sdc       8:32   0 186.3G  0 disk
sdd       8:48   0 186.3G  0 disk
sde       8:64   0 186.3G  0 disk
sdf       8:80   0 186.3G  0 disk
root@ubuntu16:~#
```


DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI

SAPIENZA
UNIVERSITÀ DI ROMA

Operating Systems: Input/Output

I/O Hardware: lista dei dispositivi USB, esempio

•**lsusb**: List USB devices.
Universal Serial Bus.

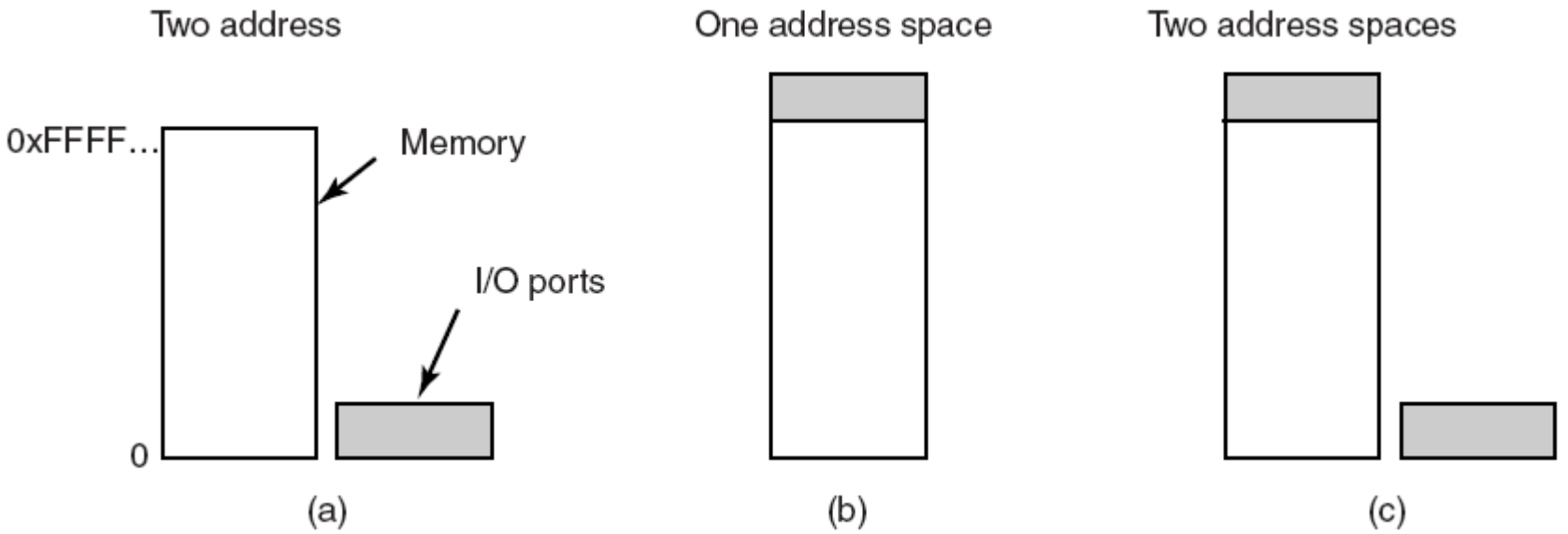
```
Bus 002 Device 002: ID 8087:0024 Intel Corp. Integrated Rate Matching Hub
Bus 002 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 001 Device 005: ID 0c45:64ad Microdia
Bus 001 Device 004: ID 0bda:0129 Realtek Semiconductor Corp. RTS5129 Card Reader Controller
Bus 001 Device 007: ID 0cf3:e004 Atheros Communications, Inc.
Bus 001 Device 002: ID 8087:0024 Intel Corp. Integrated Rate Matching Hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 004 Device 002: ID 0bc2:231a Seagate RSS LLC
Bus 004 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 003 Device 002: ID 054c:05a8 Sony Corp.
Bus 003 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI

SAPIENZA
UNIVERSITÀ DI ROMA

Operating Systems: Input/Output

I/O Hardware: Memory Mapped I/O 1/3



I/O Port: Spazi degli indirizzi di memoria ed I/O separati

Memory Mapped I/O: Spazi degli indirizzi di memoria ed I/O contigui

Hybrid: Spazi degli indirizzi di memoria ed I/O contigui

Dipartimento di Ingegneria informatica automatica e gestionale Antonio Ruberti

Sapienza Università di Roma

Operating Systems: Input/Output

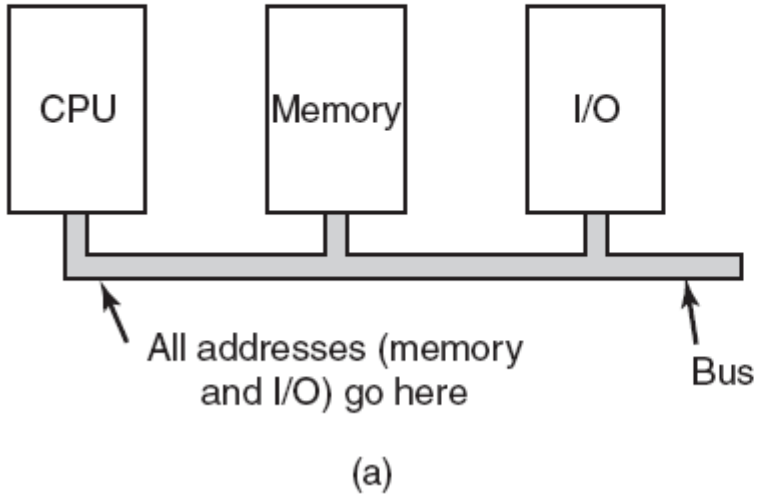
I/O Hardware: Memory Mapped I/O 2/3

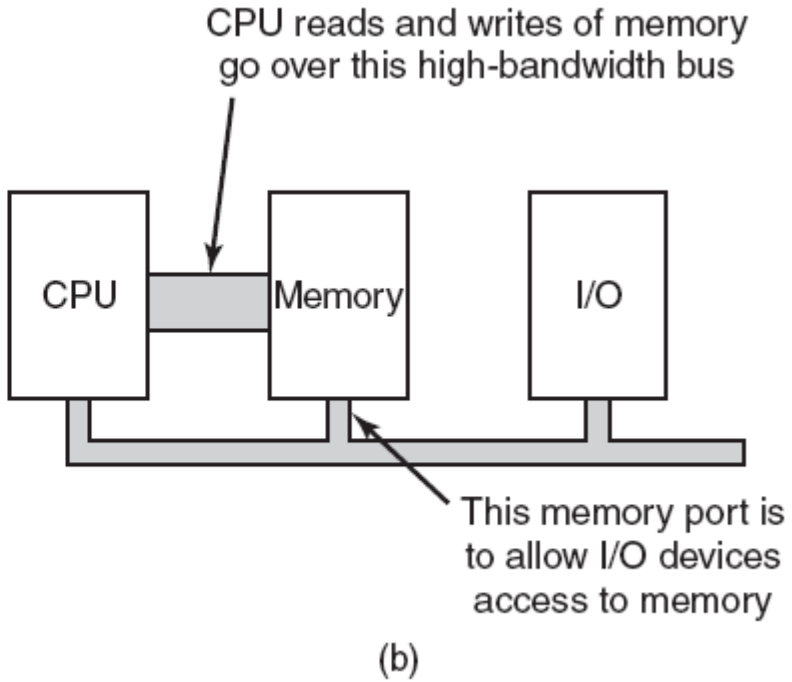
Vantaggi

1. **No Assembler**: non occorre scrivere routine specifiche
2. **Protezione**: semplice tramite inibizione di accesso a certe porzioni di indirizzo di memoria
3. **Meno Istruzioni**: Istruzioni analoghe all'accesso in RAM

Svantaggi

1. **Cache**: l'analogia con la memoria potrebbe far scattare meccanismi di caching che non hanno senso (comunque la informazione di I/O è copiata in RAM)
2. **All-Memory Scan**: esaminare tutti i riferimenti alla memoria per identificare a chi occorre rispondere ➔ **Bus Bridge**





Dipartimento di Ingegneria informatica automatica e gestionale Antonio Ruberti
Sapienza Università di Roma

Operating Systems: Input/Output

I/O Hardware: Memory Mapped I/O 3/3

Hybrid

- **Memory Map**
- **I/O Port**

```
$ lsdev                                  I/O Port     Memory Map
Device            DMA   IRQ  I/O Ports
------------------------------------------------
0000:00:07.1                   0170-0177   01f0-01f7   0376-0376   03f6-03f6   1060-106f
0000:00:07.3                   1000-103f   1040-104f
0000:00:07.7                   1080-10bf
0000:00:0f.0                   1070-107f
0000:00:10.0                   1400-14ff
0000:02:00.0                     2080-209f
0000:02:01.0                     2000-203f
0000:02:02.0                     2040-207f
acpi                      9
ACPI                               1000-1003       1004-1005       1008-100b       100c-100f       1010-1015
ata_piix             14 15         0170-0177     01f0-01f7     0376-0376     03f6-03f6     1060-106f
cascade              4
dma                            0080-008f
dma1                           0000-001f
dma2                           00c0-00df
e1000                              2000-203f
```

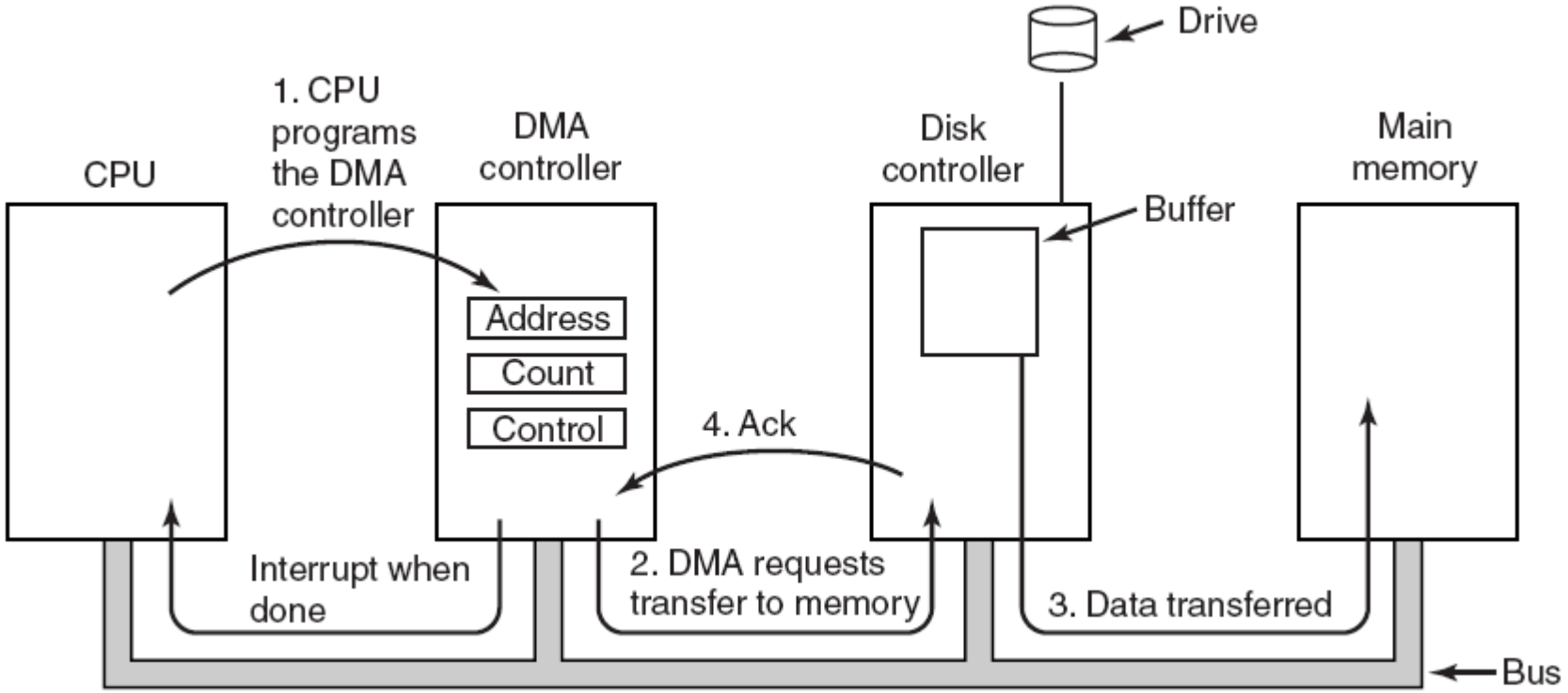
Operating Systems: Input/Output

I/O Hardware: Direct Memory Access (DMA)

DMA: dispositivo dotato di accesso al bus cui sono collegati Memoria e Device I/O.

È dotato di 3 insiemi di registri:

- **Address**: contiene l'indirizzo di memoria cui occorre accedere
- **Count**: conteggio dei byte coinvolti nel trasferimento
- **Control**: registri che specificano:
  - I/O port da usare
  - Direction (reading from/writing to) I/O device
  - Transfer Unit (byte/word)
  - Byte per burst

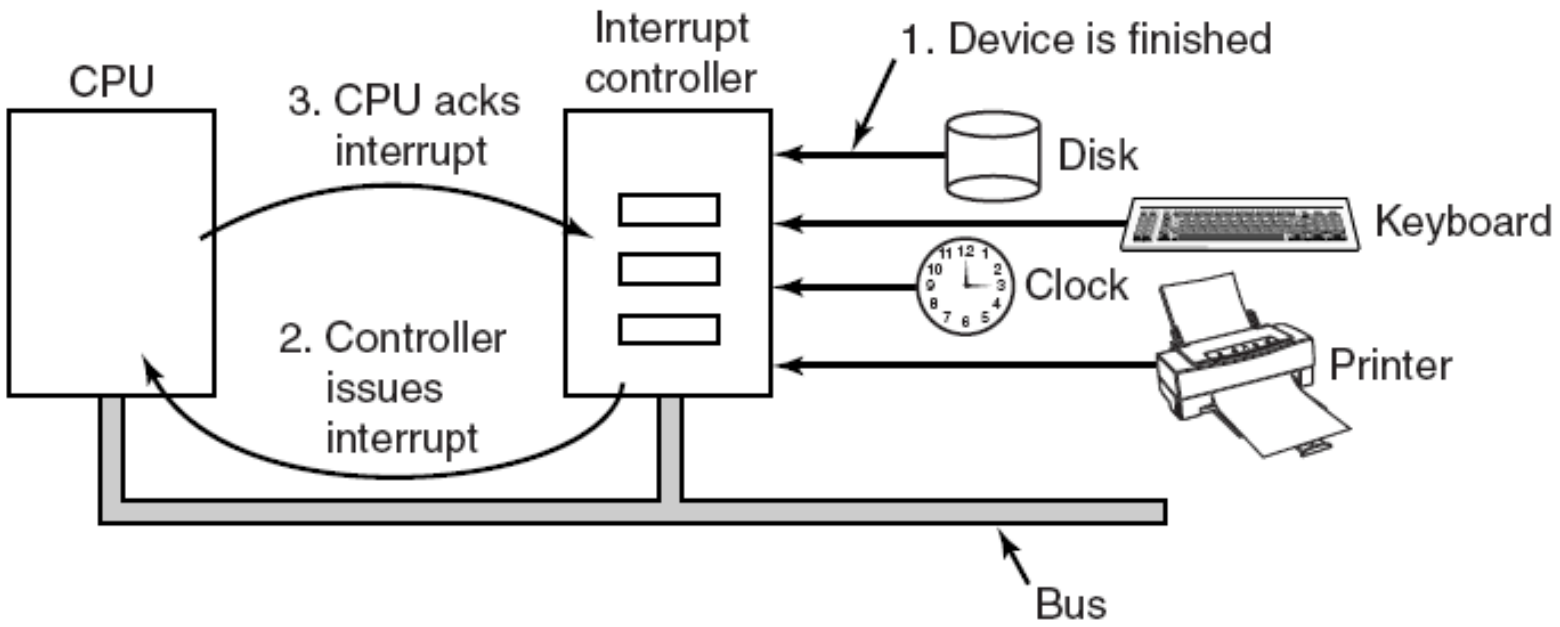


Nota: Il DMA, generalmente, non è presente nei sistemi Embedded.
Questi possono far uso di I/O Programmato (piccole moli di dati)

Operating Systems: Input/Output

I/O Hardware: Interrupt 1/2

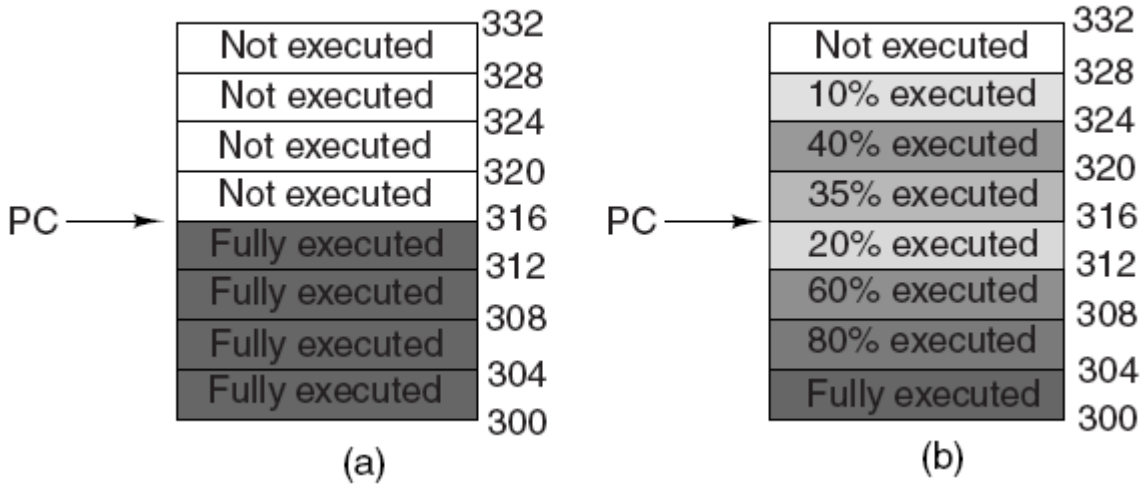
Interrupt: Le connessioni tra i dispositivi e il controllore d'interrupt utilizzano effettivamente le linee d'indirizzamento sul bus piuttosto che fili dedicati.



Operating Systems: Input/Output

I/O Hardware: Interrupt 2/2

Interrupt: L'interrupt si verifica, generalmente, in un momento non opportuno (es. pipeline della CPU piena)



(a) A precise interrupt. (b) An imprecise interrupt.

Precise Interrupt: un interrupt che presenta le seguenti proprietà

- **PC (Program Conter) salvato in un luogo noto.**
- **Tutte le istruzioni prima di quella indicata dal PC sono state completamente eseguite.**
- **Nessuna istruzione oltre a quella indicata dal PC è stata eseguita.**
- **Lo stato di esecuzione dell'istruzione indicata dal PC è noto.**

DIPARTIMENTO DI INGEGNERIA INFORMATICA AUTOMATICA E GESTIONALE ANTONIO RUBERTI

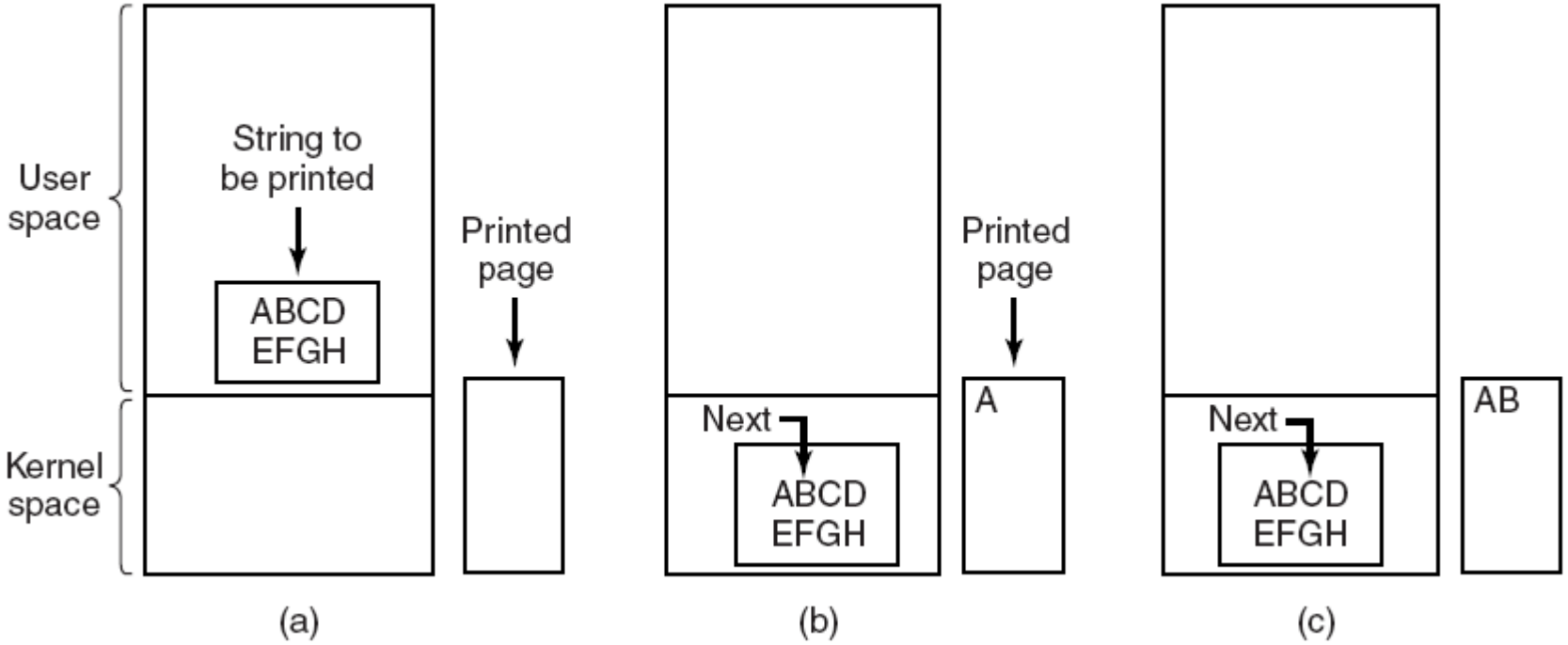
SAPIENZA UNIVERSITÀ DI ROMA

Operating Systems: Input/Output

I/O Software: Programmed I/O 1/2

I/O Programmato: tutto il lavoro è svolto dalla CPU

La CPU deve aspettare il completamento dell'operazione di I/O



Driver: parte del kernel che gestisce un particolare dispositivo di I/O

Dipartimento di Ingegneria informatica automatica e gestionale Antonio Ruberti

Sapienza Università di Roma

Operating Systems: Input/Output

I/O Software: Programmed I/O 2/2

I/O Programmato: tutto il lavoro è svolto dalla CPU

```
copy_from_user(buffer, p, count);                    /* p is the kernel buffer */
for (i = 0; i < count; i++) {                        /* loop on every character */
        while (*printer_status_reg != READY) ;       /* loop until ready */
        *printer_data_register = p[i];               /* output one character */
}
return_to_user( );
```

La CPU deve:

1. Aspettare che la stampante sia pronta
2. Provvedere a inviare un carattere alla volta
3. Aspettare che l'utente immetta il carattere

DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI

SAPIENZA
UNIVERSITÀ DI ROMA

Operating Systems: Input/Output

I/O Software: Interrupt

Interrupt Driven I/O: ogni I/O necessita l'interruzione della elaborazione corrente

```
copy_from_user(buffer, p, count);
enable_interrupts( );
while (*printer_status_reg != READY) ;
*printer_data_register = p[0];
scheduler( );
```

(a)

```
if (count == 0) {
    unblock_user( );
} else {
    *printer_data_register = p[i];
    count = count – 1;
    i = i + 1;
}
acknowledge_interrupt( );
return_from_interrupt( );
```

(b)

Scrivere una stringa alla stampante usando l'I/O guidato da interrupt.

(a) Codice eseguito al momento della chiamata al sistema di stampa.

(b) Procedura di servizio d'interruzione per la stampante

DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI

SAPIENZA
UNIVERSITÀ DI ROMA

Operating Systems: Input/Output

I/O Software: DMA

I/O con DMA: DMA agisce in nome e per conto della CPU, semplificando anche il SW

```
copy_from_user(buffer, p, count);
set_up_DMA_controller( );
scheduler( );
```

(a)

```
acknowledge_interrupt( );
unblock_user( );
return_from_interrupt( );
```

(b)

Scrivere una stringa alla stampante usando il DMA.

(a) Codice eseguito al momento della chiamata al sistema di stampa.

(b) Procedura di servizio per la gestione dell'avvenuta stampa.

DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI

SAPIENZA
UNIVERSITÀ DI ROMA

Operating Systems: Input/Output

I/O Software: Progettazione gerarchica

Gerarchia ➔ Layers

- **Servizio** (offerta): fornisce servizi al livello superiore
- **Interfaccia** (fruizione del servizio): il livello sottostante effettua operazioni più primitive
- **Protocollo** (funzionamento): nello stesso livello funzionalità simili per complessità, tempi di esecuzione, astrazione

➔ **Standard**: Il SO deve poter far sembrare uguali la lettura da qualsiasi dispositivo (tastiera, file, mouse).

(Nella figura, i caratteri generati da un mouse).

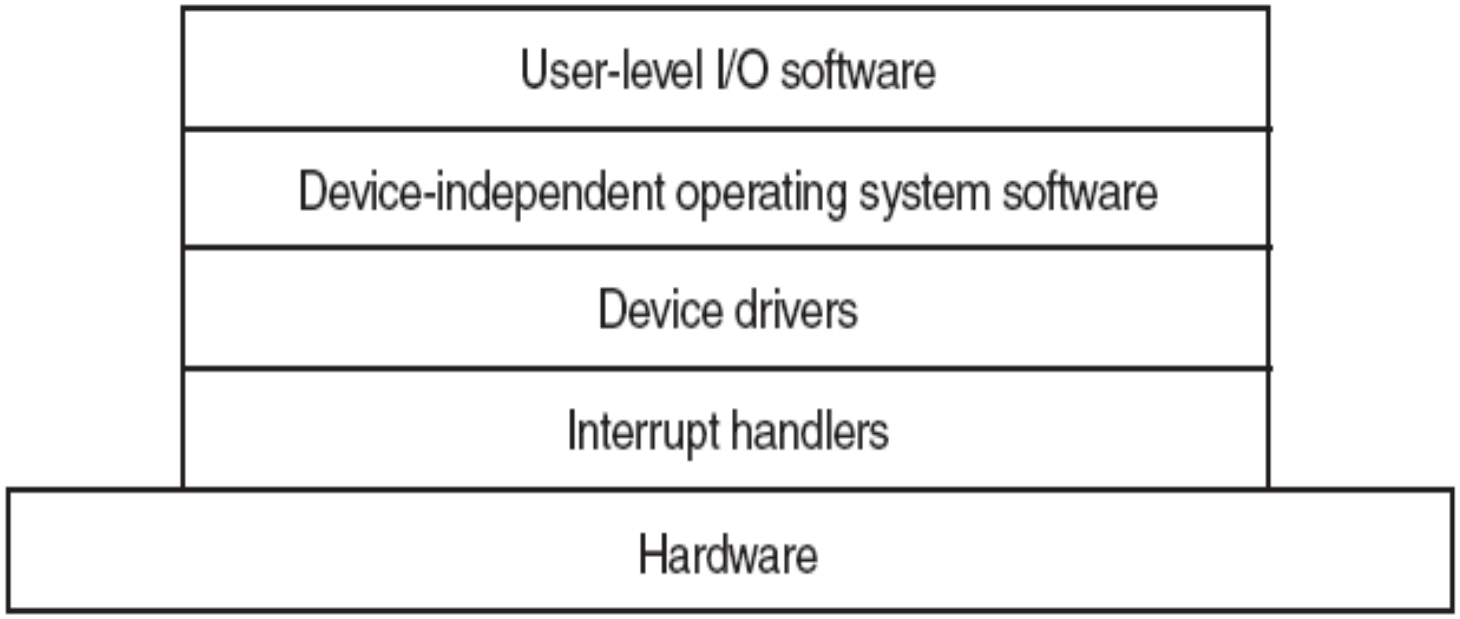
DIPARTIMENTO DI INGEGNERIA INFORMATICA AUTOMATICA E GESTIONALE ANTONIO RUBERTI

SAPIENZA UNIVERSITÀ DI ROMA

Operating Systems: Input/Output

I/O Software: Layers 1/2

I/O SW Layers: Strutturazione del software a livelli di servizio



- **Management**: operazioni utente che tengono conto della semantica
- **Logical I/O**: dispositivo viene visto come una risorsa logica (open, close, read, ...) ➔ File System
- **Device I/O**: trasforma richieste logiche in sequenze di comandi di I/O
- **Scheduling and Control**: esegue e controlla le sequenze di comandi, eventualmente gestendo l'accodamento

Dipartimento di Ingegneria informatica automatica e gestionale Antonio Ruberti

Sapienza Università di Roma

Operating Systems: Input/Output

I/O Software: Layers 2/2

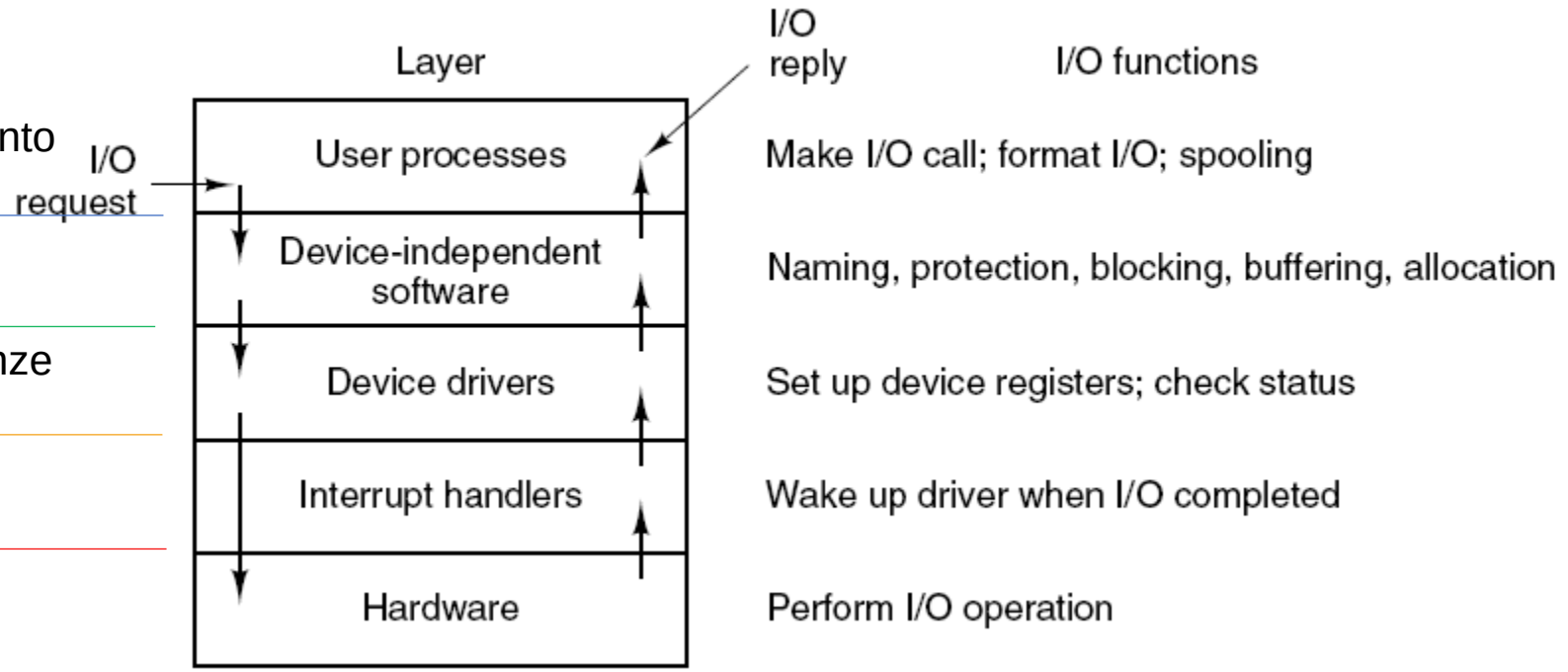
I/O SW Layers: Strutturazione del software a livelli di servizio

Management: operazioni utente che tengono conto della semantica ➔

Logical I/O: dispositivo viene visto come una risorsa logica (open, close, read, ...) ➔

Device I/O: trasforma richieste logiche in sequenze di comandi di I/O ➔

Scheduling and Control: esegue e controlla le sequenze di comandi, gestendo l'accodamento



Dipartimento di Ingegneria informatica automatica e gestionale Antonio Ruberti

Sapienza Università di Roma

Operating Systems: Input/Output

Dispositvi I/O: esempio lsusb

- **lsusb**: List USB devices. Universal Serial Bus.

```
Bus 002 Device 002: ID 8087:0024 Intel Corp. Integrated Rate Matching Hub
Bus 002 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 001 Device 005: ID 0c45:64ad Microdia
Bus 001 Device 004: ID 0bda:0129 Realtek Semiconductor Corp. RTS5129 Card Reader Controller
Bus 001 Device 007: ID 0cf3:e004 Atheros Communications, Inc.
Bus 001 Device 002: ID 8087:0024 Intel Corp. Integrated Rate Matching Hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 004 Device 002: ID 0bc2:231a Seagate RSS LLC
Bus 004 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 003 Device 002: ID 054c:05a8 Sony Corp.
Bus 003 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```


DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI

SAPIENZA
UNIVERSITÀ DI ROMA

Operating Systems: Input/Output

I/O Software: Interrupt Handlers

Scheduling and Control: esegue e controlla le sequenze di comandi, eventualmente gestendo l'accodamento

1. Salva i registri che non sono già stati salvati dall l'hardware dell'interrupt.
2. Imposta un context per la procedura di servizio di interruzione.
3. Imposta uno stack per la procedura di servizio d'interruzione.
4. Riconosce il controllore dell'interruzione. Se non c'è un controllore d'interruzione centralizzato, riabilita gli interrupt.
5. Copia i registri da dove sono stati salvati nella tabella dei processi.
6. Esegue la procedura di servizio di interruzione.
7. Sceglie quale processo eseguire dopo.
8. Imposta il contesto MMU per il processo da eseguire successivamente.
9. Carica i registri del nuovo processo, incluso il suo PSW.
10. Inizia l'esecuzione del nuovo processo.

DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI

SAPIENZA
UNIVERSITÀ DI ROMA

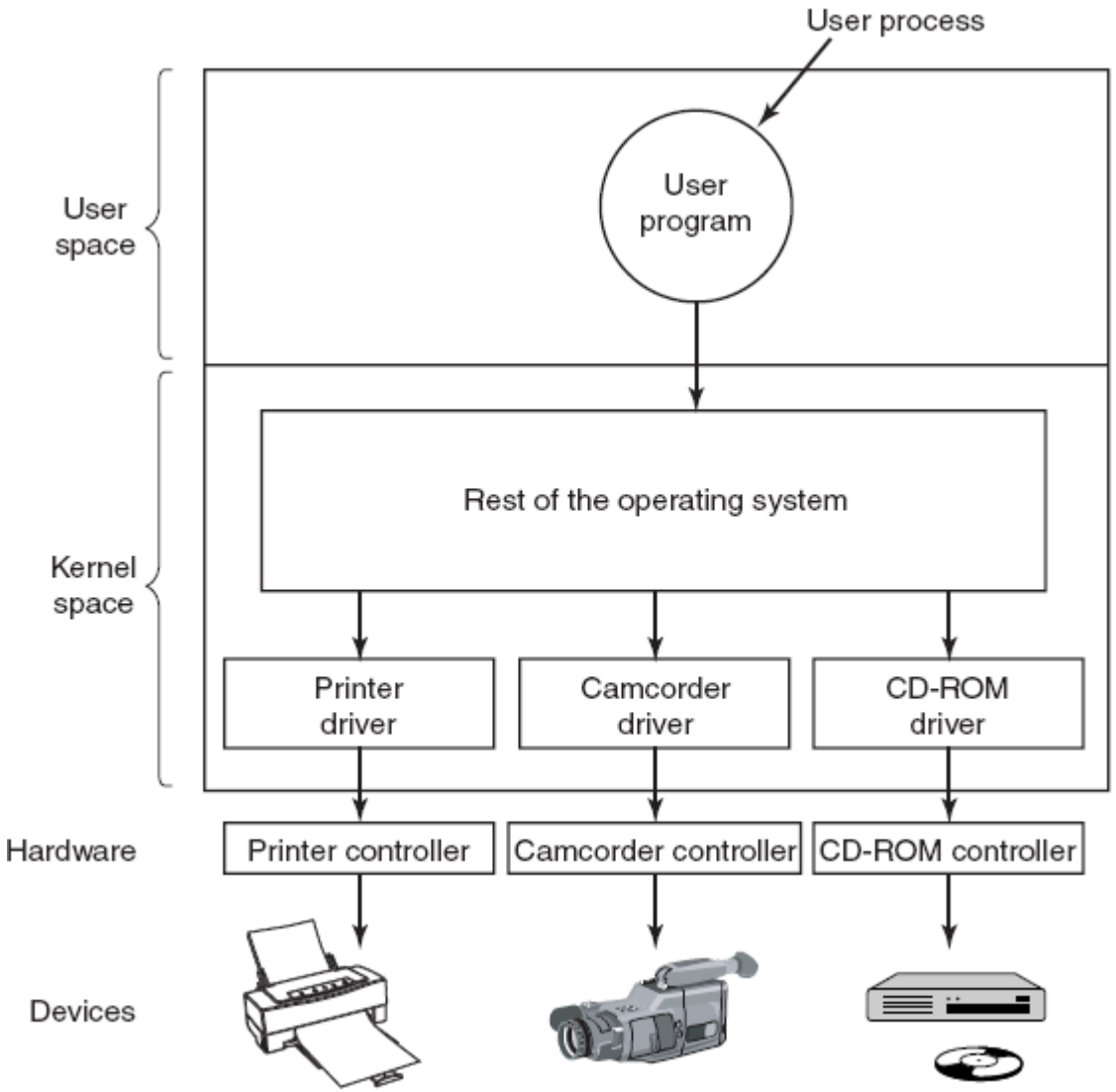
I/O Driver

Componente SW di interfaccia con i dispositivi periferici (!= CPU | RAM)

Operating Systems: Input/Output

I/O Software: Device Drivers

Device I/O: trasforma richieste logiche in sequenze di comandi di I/O



posizionamento di driver dei dispositivi. In realtà tutte le comunicazione tra i driver e i controllori di dispositivi passa attraverso il bus.

Dipartimento di Ingegneria informatica automatica e gestionale Antonio Ruberti

Sapienza Università di Roma

Operating Systems: Input/Output

I/O Software: Device-Independent OS Software 1/4

Logical I/O: dispositivo viene visto come una risorsa logica (open, close, read, ...)

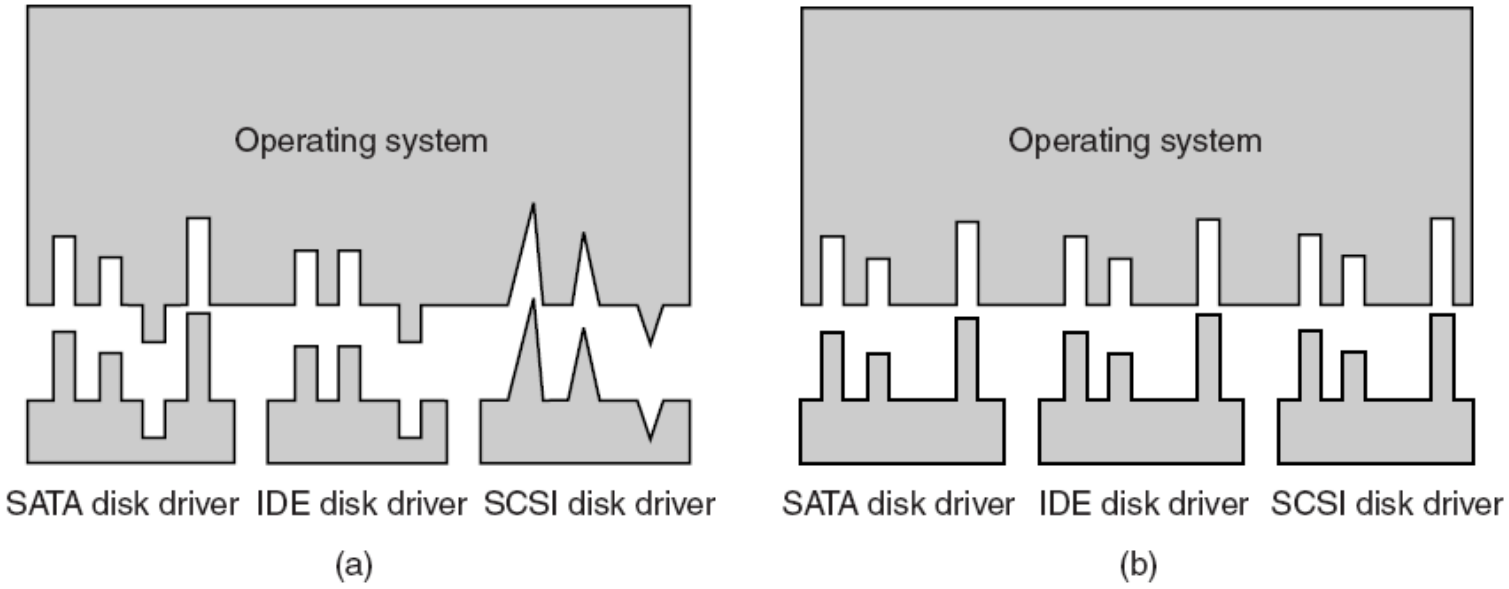
Funzioni

Uniform interfacing for device drivers
Buffering
Error reporting
Allocating and releasing dedicated devices
Providing a device-independent block size

Operating Systems: Input/Output

I/O Software: Device-Independent OS Software 2/4

Logical I/O: dispositivo viene visto come una risorsa logica (open, close, read, ...)



(a) Senza interfaccia standard del driver.

(b) Con interfaccia standard del driver.

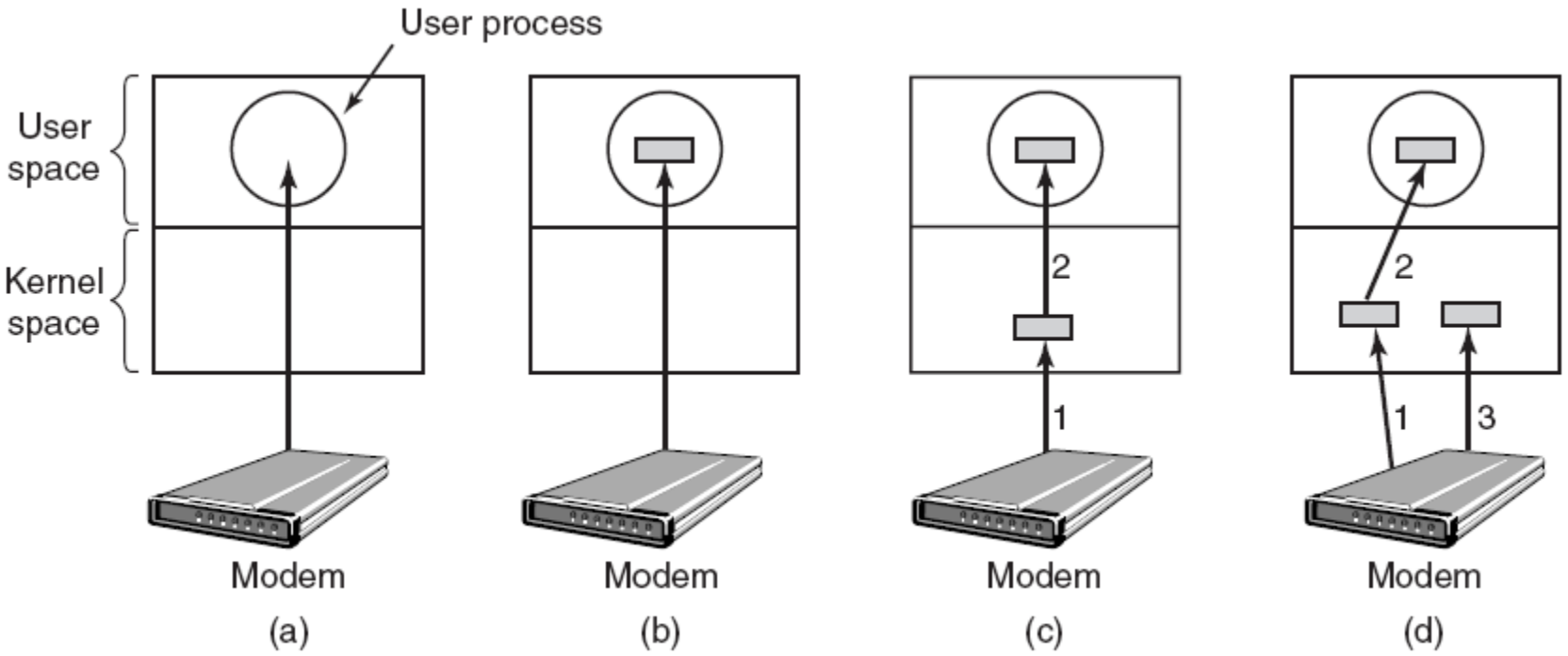
Dipartimento di Ingegneria informatica automatica e gestionale Antonio Ruberti

Sapienza Università di Roma

Operating Systems: Input/Output

I/O Software: Device-Independent OS Software 3/4

Logical I/O: dispositivo viene visto come una risorsa logica (open, close, read, ...)



(a) Ingresso non bufferizzato.
Il SO accede al dispositivo nel momento in cui ne ha necessità

(b) Buffering nello spazio utente.

(c) Buffering nel kernel seguito dalla copia nello spazio utente.
Il SO crea un buffer in memoria principale (system-space, non user-space!) per una data richiesta di I/O
Stream ➔ produttore/consumatore

(d) Doppio buffering nel kernel.
Prossimo blocco nel buffer:
- input anticipato (anticipated input)
- lettura in anticipo (read ahead)

(e) Buffering Multiplo nel kernel
➔ Produttore/Consumatore

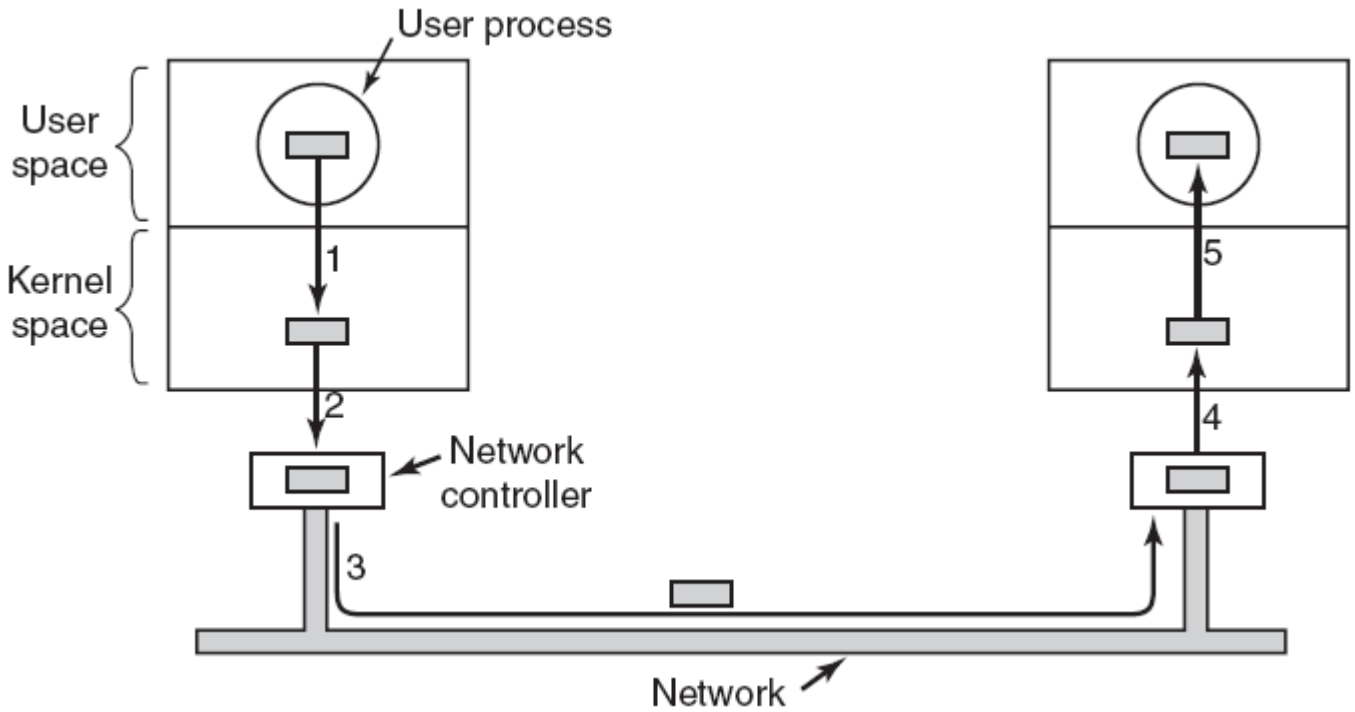
Operating Systems: Input/Output

I/O Software: Device-Independent OS Software 4/4

Logical I/O: dispositivo viene visto come una risorsa logica (open, close, read, ...)

Il networking può comportare molte copie di un pacchetto

1. User (sender)
2. Kernel (sender)
3. Scheda di rete (sender)
4. Scheda di rete (receiver)
5. Kernel (receiver)

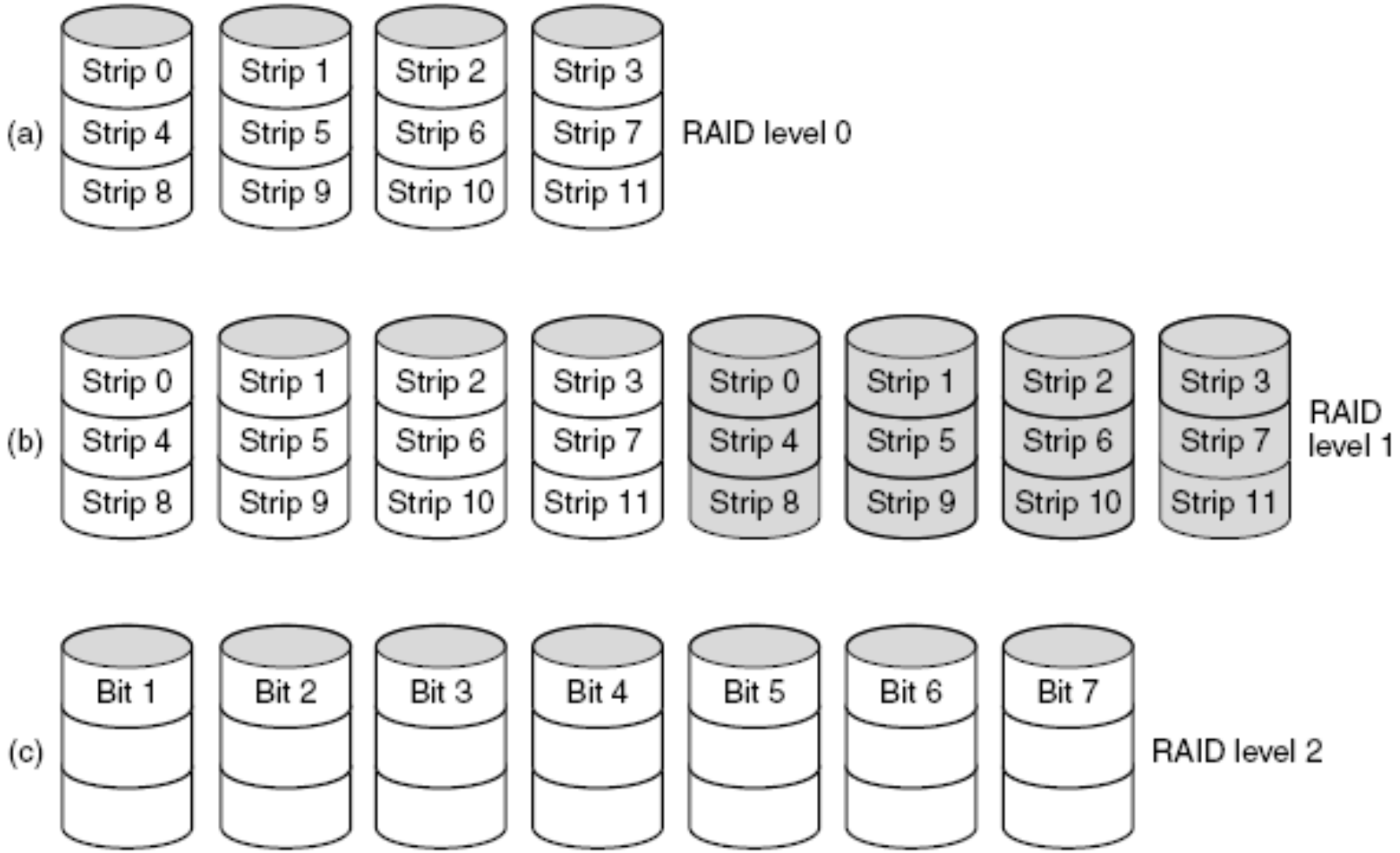


Dipartimento di Ingegneria informatica automatica e gestionale Antonio Ruberti

Sapienza Università di Roma

Operating Systems: Input/Output

Dischi: RAID (Redundant Array of Independent/Inexpensive Disk) 1/2

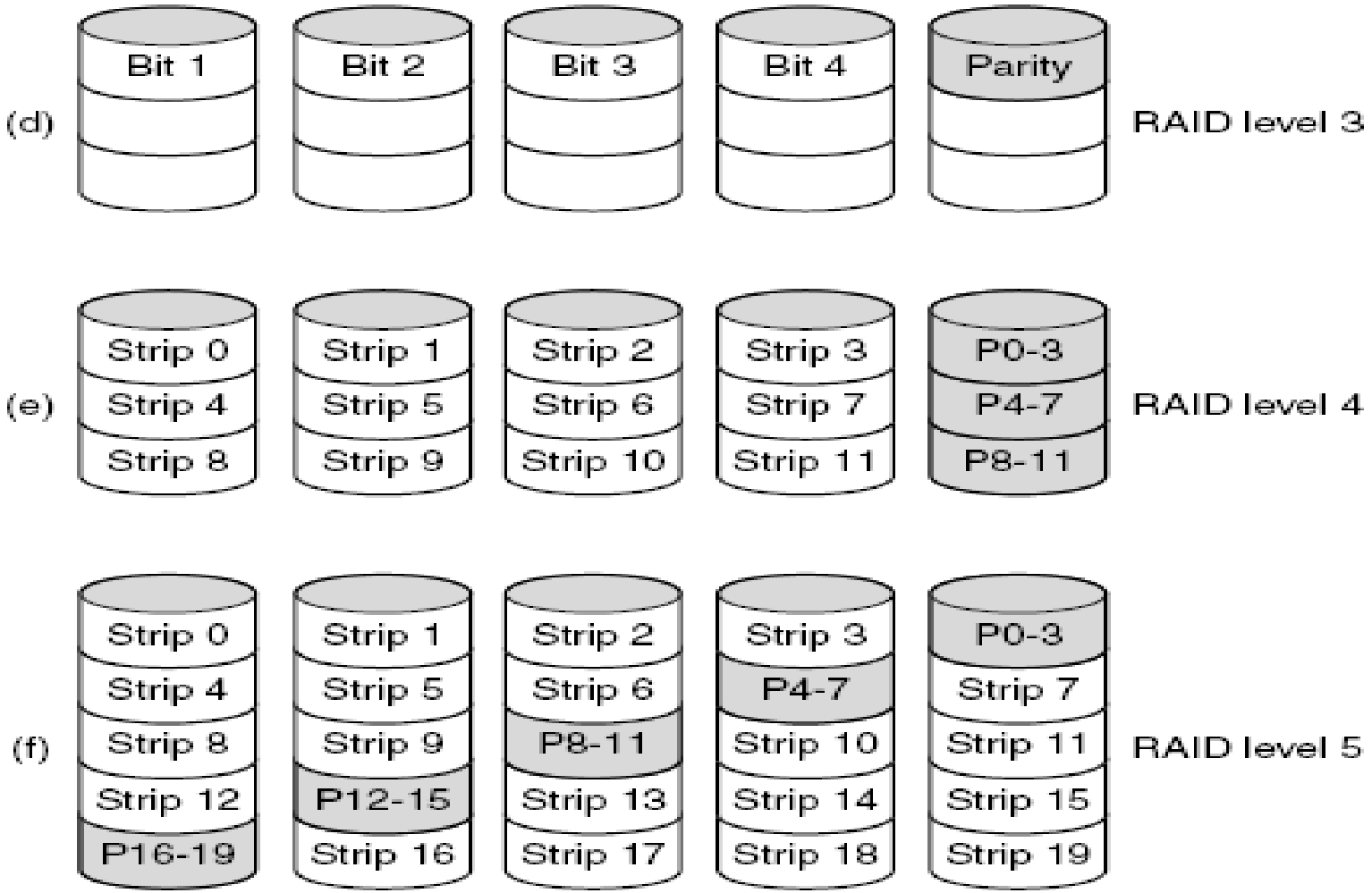


Dipartimento di Ingegneria informatica automatica e gestionale Antonio Ruberti

Sapienza Università di Roma

Operating Systems: Input/Output

Dischi: RAID (Redundant Array of Independent/Inexpensive Disk) 2/2



Dipartimento di Ingegneria informatica automatica e gestionale Antonio Ruberti

Sapienza Università di Roma

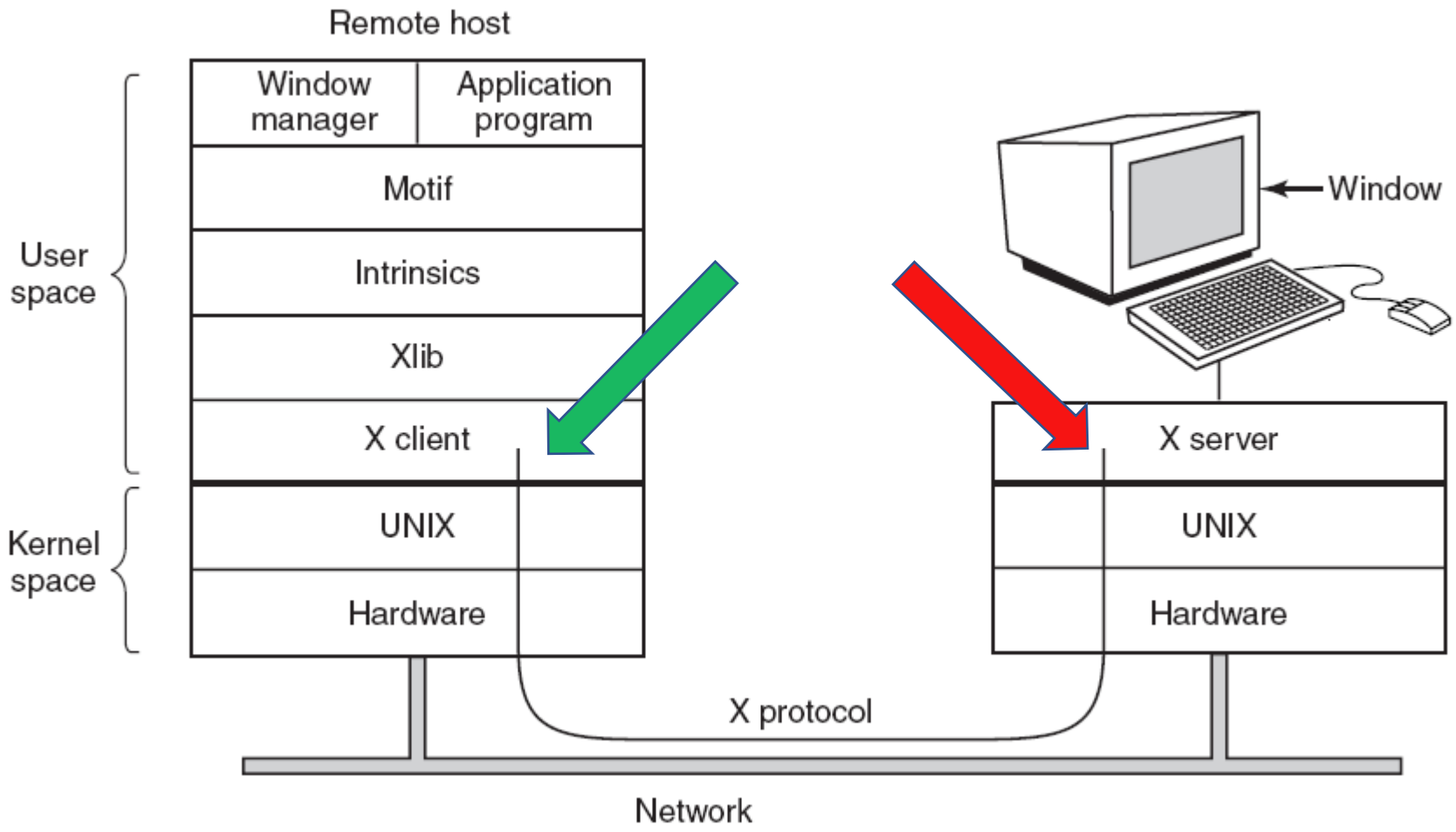
Client-Server

Esempio X-window

Operating Systems: Input/Output

Altri Dispositivi: Graphical User Interface

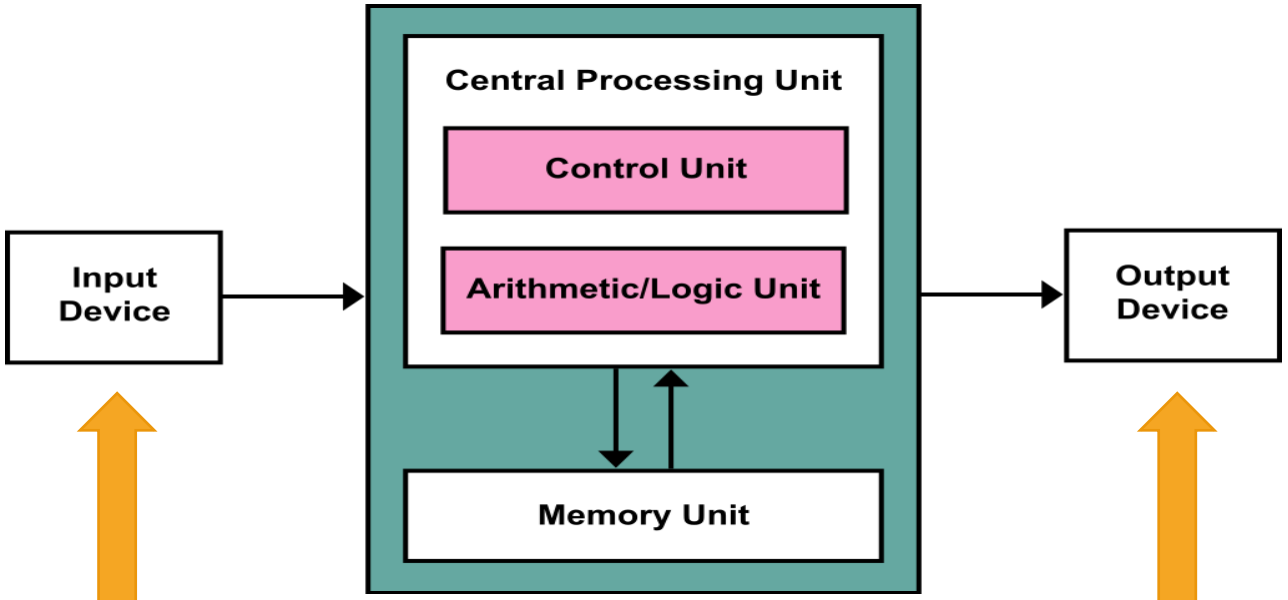
X-Window System: interfaccia grafica per Unix



File System

Operating Systems: File Systems

Concetto di File 1/3



Architettura di von Neumann

contenitore di informazioni

File = **unità di memorizzazione elementare**. Proprietà:

1. **Permanenza**: esistenza a lungo termine (oltre la terminazione dei processi) ➔ archiviazione (When)
2. **Ergonomicità**: contenere tutte e sole (Why) le informazioni di un certo argomento (es. Documento, Video, Musica) utili al'utente ➔ dimensione e struttura (How)
3. **Ricercabilità**: assegnazione di un nome (What) mnemonico, simbolico ed organizzazioni in strutture➔ directory gerarchiche (Where)
4. **Condivisibilità**: riuso anche da parte di altri processi (Who)

Dipartimento di Ingegneria Informatica
Automatica e Gestionale Antonio Ruberti

Sapienza
Università di Roma

Operating Systems: File Systems

Concetto di File 2/3

File:

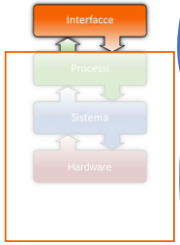
- **Interfaccia** (utente): Visione logica uniforme delle informazioni

- **Servizio**: Disaccoppiamento tra le caratteristiche fisiche dei vari dispositivi di memoria, le modalità di memorizzazione dell'informazione e l'interfaccia utente

- **Protocollo**: Indipendenza dal tipo di dispositivo fisico (es. Disco magnetico, disco a strato solido, nastro, cd, dvd). Ognuno con diversa velocità di acceso

Operating Systems: Obiettivi Funzioni Servizi

File System



File: contenitore di informazioni/dati in formato digitale

Nasconde all'utente l'organizzazione della memoria:

- File e directory
- Creazione e cancellazione
- Lettura e scrittura
- Copiatura
- Ricerca
- Salvataggio e ripristino
- Protezione e sicurezza (diritti di accesso)

In Unix "Everything is a File".

1. - : regular file
2. d : directory
3. c : character device file
4. b : block device file
5. s : local socket file
6. p : named pipe
7. l : symbolic link

Il file è una pregevole astrazione del S.O..

SAPIENZA UNIVERSITÀ DI ROMA

DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI

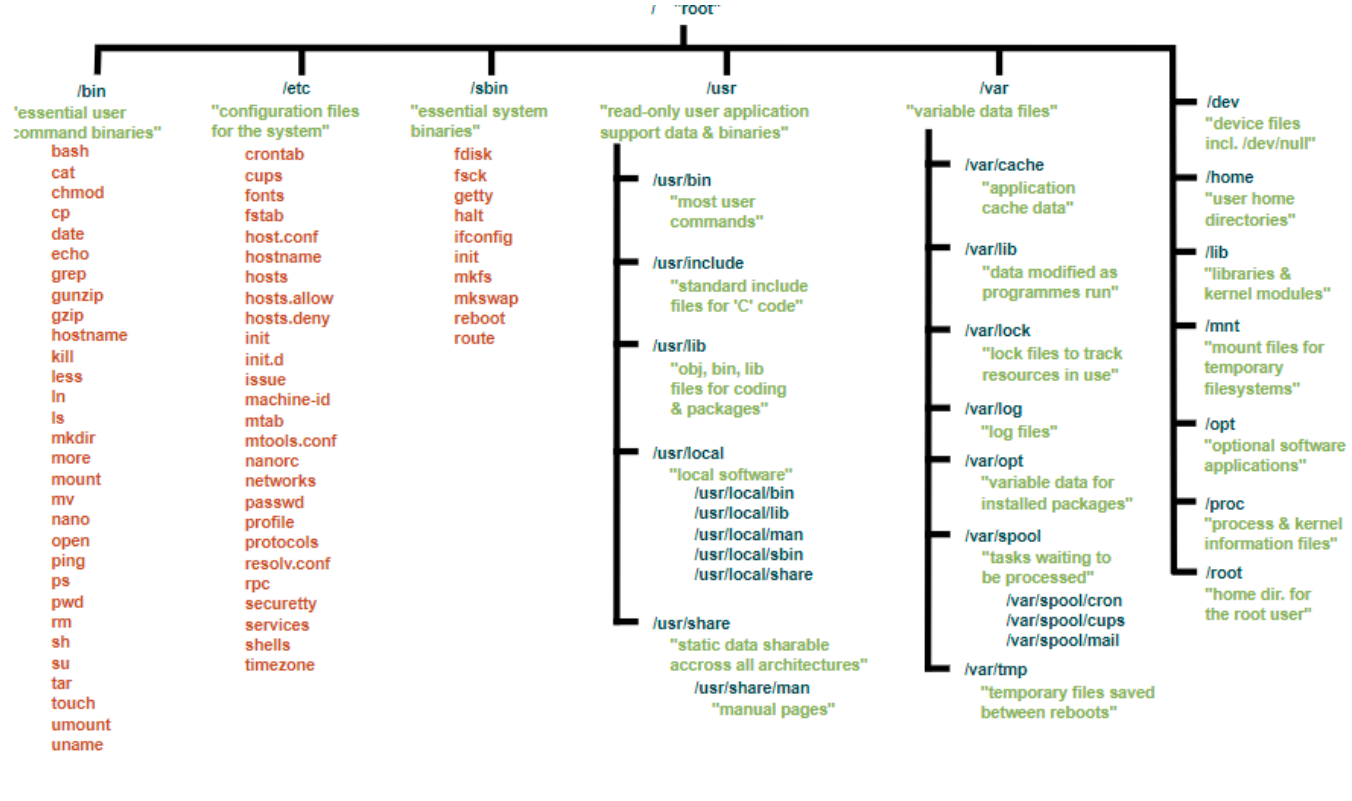
SAPIENZA
UNIVERSITÀ DI ROMA

Operating Systems: File Systems

Concetto di File 3/3

Operating Systems: Obiettivi Funzioni Servizi

File System



File System:

- **Astrazione**: gestire l'accesso alle informazioni conservate in blocchi su un disco.
- **Utilizzo**: operazioni (creazione, cancellazione, modifica), protezione ed organizzazione
- **Implementazione**: modalità con cui sono organizzati sui dischi. Strutturazione:
  - File regolare: -
  - Directory: d
  - Link: l

DIPARTIMENTO DI INGEGNERIA INFORMATICA AUTOMATICA E GESTIONALE ANTONIO RUBERTI

SAPIENZA UNIVERSITÀ DI ROMA

Operating Systems: File Systems

File: Utilizzo 1/13

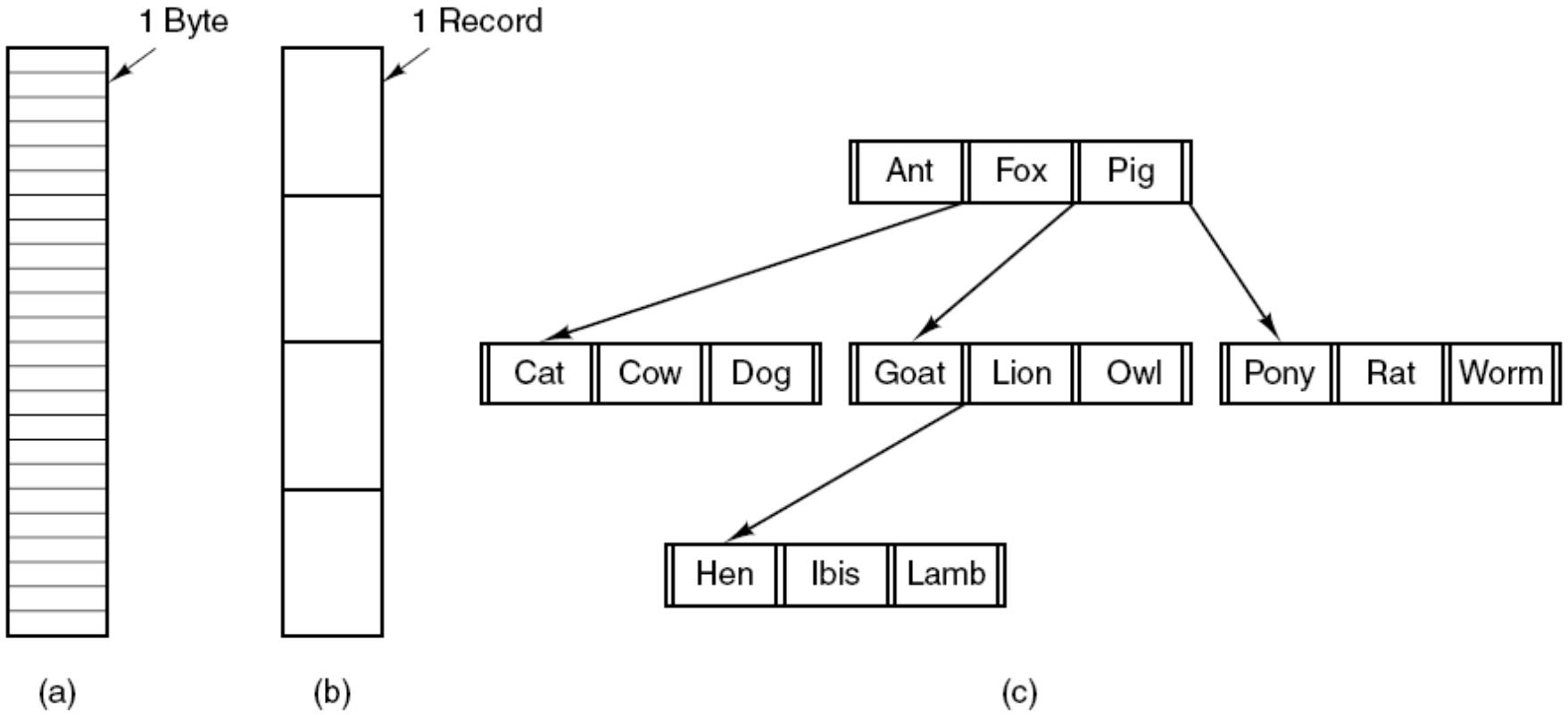
Denominazione

- Da 1 a 8 lettere in tutti i sistemi operativi attuali
- File system Unix, MS-DOS (Fat16)
- Fat (16 e 32) sono stati utilizzati nei primi sistemi Windows
- Gli ultimi sistemi Windows utilizzano il file system nativo
- Tutti i sistemi operativi utilizzano il suffisso come parte del nome
- Unix non impone sempre un significato per i suffissi
- DOS impone un significato

Extension	Meaning
file.bak	Backup file
file.c	C source program
file.gif	Compuserve Graphical Interchange Format image
file.hlp	Help file
file.html	World Wide Web HyperText Markup Language document
file.jpg	Still picture encoded with the JPEG standard
file.mp3	Music encoded in MPEG layer 3 audio format
file.mpg	Movie encoded with the MPEG standard
file.o	Object file (compiler output, not yet linked)
file.pdf	Portable Document Format file
file.ps	PostScript file
file.tex	Input for the TEX formatting program
file.txt	General text file
file.zip	Compressed archive

Operating Systems: File Systems

File: Utilizzo 2/13



(a) Byte sequence. (b) Record sequence. (c) Tree

Struttura

1. **Byte Sequence**: Sequenze di byte
   - Massima flessibilità: puoi inserire qualsiasi cosa
   - Unix e Windows usano questo approccio
2. **Record Sequence**: Record a lunghezza fissa (immagini delle carte ai vecchi tempi)
3. **Tree**: Albero dei record: utilizza il campo chiave per trovare i record nell'albero

Operating Systems: File Systems

File: Utilizzo 3/13

Tipi di File

1. - : regular file
2. d : directory
3. c : character device file
4. b : block device file
5. s : local socket file
6. p : named pipe
7. l : symbolic link

Informazioni Utente (formato ASCII o Binario)

Contenitori logici di FIle

Cfr. 9. Input/Output

Cfr. 5. Thread - IPC

Eventuale altra denominazione

File Regolare: accesso

1. **Sequenziale**: ordinato, dall'inizio del file (come i nastri)
2. **Random**: dal punto prescelto. Possibile

DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI

SAPIENZA
UNIVERSITÀ DI ROMA

Operating Systems: File Systems

File: Utilizzo 4/13

File Regolare: ASCII

American Standard Code for Information Interchange

ideato dall'ingegnere dell'IBM Bob Bemer nel 1961, proposto dall'ANSI (American National Standard Institute) nel 1963 e pubblicato nel 1968 (2 anni prima dell'Epoch Time).

ASCII Code Chart

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2		!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	DEL

1. 7 bit (l'8 era usato come parità) ➔ 128 caratteri
2. 0-31 (primi 32 caratteri) di **controllo** della trasmissione
   - 10: Line Feed (LF)
   - 13: Carriage Return (CR)
3. 32-127 (secondi 95 caratteri) **stampabili**
4. 128 (ultimo carattere) DEL: delete

Nota: codifica del "a capo" storicamente diversa tra SO:

- **Unix**: solo LF (Line Feed)
- **Mac/OS**: solo CR (Carriage Return)
- **MS-DOS**: CR + LF. Soluzione corretta cui si stanno conformando tutti.

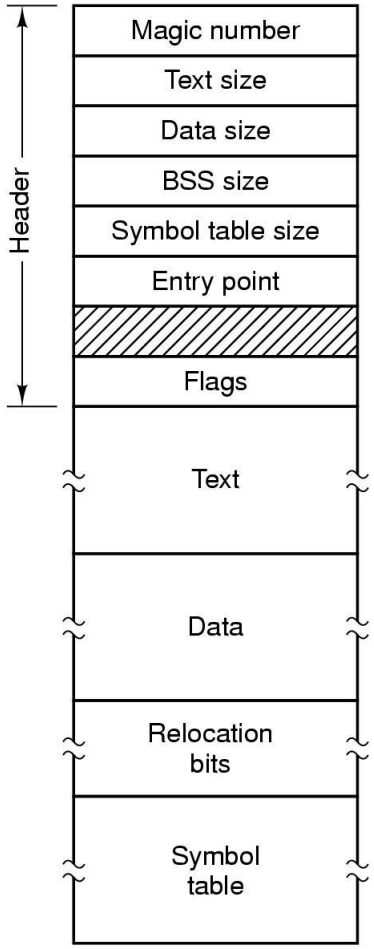
Operating Systems: File Systems

File: Utilizzo 5/13

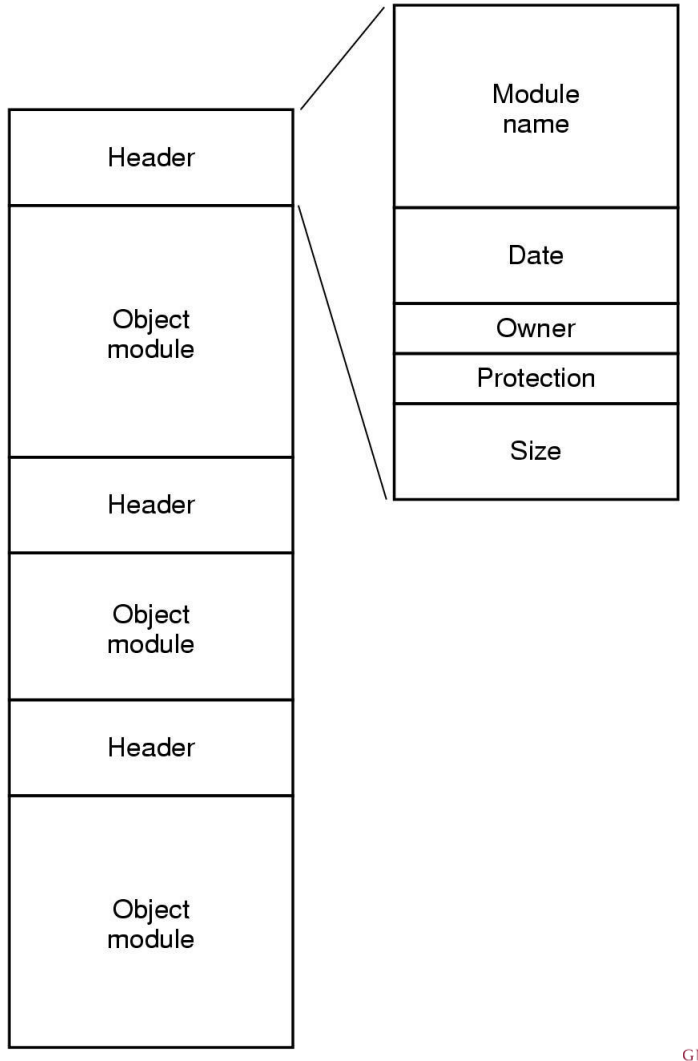
File Regolare: binario

All'inizio di ciascun file binario sono inserite informazioni sulla tipologia di file. Queste costituiscono i ***Magic Number***:

- da cui si può risalire alla tipologia di binario (es. eseguibile, audio, video, etc)
- che includono dettagli sul loro funzionamento (es. ELF, a.out; Wav, mp3, sample rate; MPEG, MP4; etc)
- (per i soli file eseguibili) I valori esadecimali da passare come argomento per la esecuzione



(a)



(b)

Operating Systems: File Systems

File: Utilizzo 5b/13

C:\WINDOWS\system32\cmd.exe

```
C:\Users\A705945>file c:\Windows\explorer.exe
c:\Windows\explorer.exe: PE32+ executable (GUI) x86-64, for MS Windows

C:\Users\A705945>file c:\Windows\win.ini
c:\Windows\win.ini: ASCII text, with CRLF line terminators

C:\Users\A705945>file c:\Windows\WindowsShell.Manifest
c:\Windows\WindowsShell.Manifest: XML 1.0 document, ASCII text, with CRLF line termina
tors

C:\Users\A705945>file c:\Windows\twain_32.dll
c:\Windows\twain_32.dll: PE32 executable (DLL) (console) Intel 80386, for MS Windows

C:\Users\A705945>file c:\Windows\WMSysPr9.prx
c:\Windows\WMSysPr9.prx: Little-endian UTF-16 Unicode text, with CRLF line terminators


C:\Users\A705945>file c:\Windows\OccaCollaborationLibPopulate.tlb
c:\Windows\OccaCollaborationLibPopulate.tlb: data

C:\Users\A705945>file c:\Windows\Fonts\arial.ttf
c:\Windows\Fonts\arial.ttf: TrueType font data

C:\Users\A705945>file c:\Paolo\Hom\Photos\Laser\20200604_154941.jpg
c:\Paolo\Hom\Photos\Laser\20200604_154941.jpg: JPEG image data, Exif standard: [TIFF i
mage data, little-endian, direntries=12, height=3096, manufacturer=samsung, model=SM-J
530F, orientation=upper-left, xresolution=158, yresolution=166, resolutionunit=2, soft
ware=J530FXXU6CSK9, datetime=2020:06:04 15:49:41, width=4128], baseline, precision 8,
4128x3096, frames 3
```

Magic Number: File command

Comando sviluppato nel 1973 (3 dopo la nascita ufficiale di Unix) : confronta l'inizio del file con un archivio di Magic Number (esterno nel System V della AT&T dal 1983).

Versione Open Source: riscrittura completa presente in tutti i sistemi POSIX. Raccolta di Darwin/Zoulas di informazioni formattate su magic file.

Porting per windows su github

Name	Size	Modified
COPYING.file	1 656	2017-01-08 12:40
COPYING.libgnurx	26 536	2017-01-08 12:40
file.exe	172 635	2017-01-08 12:40
libgnurx-0.dll	194 835	2017-01-08 12:40
libmagic-1.dll	775 304	2017-01-08 12:40
magic.mgc	4 873 104	2017-01-08 12:40

MATICA RUBERTI

Operating Systems: File Systems

File: Utilizzo 5c/13

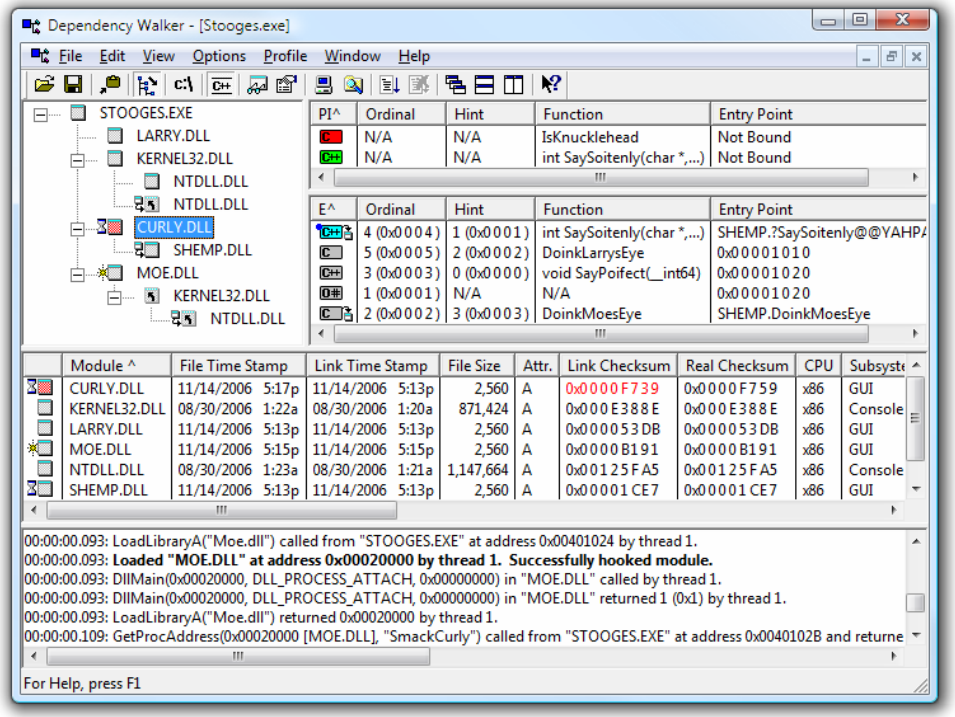
Dependence Links

I file eseguibili e le librerie possono contenere link a funzioni contenute nelle librerie condivise (DLL: Dynamic Link Library) contenute nel sistema. Il SO dovrebbe anche fornire intrinsecamente strumenti per effettuare eventuali troubleshooting.

Ldd: Load Dynamic Dependencies

Comando presente in tutti i sistemi Unix: stampa le dipendenze delle librerie condivise

```
$ ldd /bin/cp
 linux-vdso.so.1 =>  (0x00007fffaf3ff000)
    libselinux.so.1 => /lib64/libselinux.so.1 (0x0000003a06a00000)
    librt.so.1 => /lib64/librt.so.1 (0x0000003a06200000)
    libacl.so.1 => /lib64/libacl.so.1 (0x0000003a13000000)
    libattr.so.1 => /lib64/libattr.so.1 (0x0000003a0ea00000)
    libc.so.6 => /lib64/libc.so.6 (0x0000003a05200000)
    libdl.so.2 => /lib64/libdl.so.2 (0x0000003a05a00000)
    /lib64/ld-linux-x86-64.so.2 (0x0000003a04a00000)
    libpthread.so.0 => /lib64/libpthread.so.0 (0x0000003a05600000)
```



Depends: Dependency Walker

Utilità free per Windows, esterna

DIPARTIMENTO DI INGEGNERIA INFORMATICA AUTOMATICA E GESTIONALE ANTONIO RUBERTI

SAPIENZA UNIVERSITÀ DI ROMA

Operating Systems: File Systems

File: Utilizzo 6/13

File Regolare: attributi

Ogni SO utilizza almeno un File System. Deve implementare l'utilizzo completo, in modo da poter gestire le meta informazioni sui file.

Difatti, ogni file è dotato di determinate proprietà da considerare per il suo utilizzo, addizionali rispetto al contenuto stesso.

Attribute	Meaning
Protection	Who can access the file and in what way
Password	Password needed to access the file
Creator	ID of the person who created the file
Owner	Current owner
Read-only flag	0 for read/write; 1 for read only
Hidden flag	0 for normal; 1 for do not display in listings
System flag	0 for normal files; 1 for system file
Archive flag	0 for has been backed up; 1 for needs to be backed up
ASCII/binary flag	0 for ASCII file; 1 for binary file
Random access flag	0 for sequential access only; 1 for random access
Temporary flag	0 for normal; 1 for delete file on process exit
Lock flags	0 for unlocked; nonzero for locked
Record length	Number of bytes in a record
Key position	Offset of the key within each record
Key length	Number of bytes in the key field
Creation time	Date and time the file was created
Time of last access	Date and time the file was last accessed
Time of last change	Date and time the file was last changed
Current size	Number of bytes in the file
Maximum size	Number of bytes the file may grow to

Operating Systems: File Systems

File: Utilizzo 6b/13

Attributi per ext2

Oltre agli attributi `rwx` (Read, Write, eXecute), relativi ai permessi `ugo` (User, Group, Other), ne sono disponibili altri. Ad esempio, nei FS ext2.

```
sysadmin@ubuntu:~$ lsattr /etc/passwd
------------e-- /etc/passwd
sysadmin@ubuntu:~$
sysadmin@ubuntu:~$
sysadmin@ubuntu:~$ sudo chattr +i /etc/passwd
sysadmin@ubuntu:~$
sysadmin@ubuntu:~$
sysadmin@ubuntu:~$ lsattr /etc/passwd
----i-------e-- /etc/passwd
sysadmin@ubuntu:~$
sysadmin@ubuntu:~$
sysadmin@ubuntu:~$ sudo rm /etc/passwd
rm: cannot remove '/etc/passwd': Operation not permitted
sysadmin@ubuntu:~$
```

Add immutable attribute

Check immutable attribute

- **A** Non aggiornare l'atime(*)
- **S** Aggiornamento sincrono
- **D** Aggiornamento sincrono delle directory
- **a** solo append
- **c** compresso
- **d** no dump
- **i** immutabile
- **s** cancellazione sicura
- **T** top of directory hierarchy
- **j** data journalling
- **t** no tail-merging
- **u** Non cancellabile

(*)

- atime (access time): istante di ultimo accesso al file
- mtime (modify time): istante di ultimo modifica al file
- ctime (change time): istante di ultimo cambiamento ai metadata del file

Questi valori sono visualizzati con il commando `stat <nome-file>`

DIPARTIMENTO DI INGEGNERIA INFORMATICA AUTOMATICA E GESTIONALE ANTONIO RUBERTI

SAPIENZA UNIVERSITÀ DI ROMA

Operating Systems: File Systems

File: Utilizzo 7/13

File: System Call

Ogni SO utilizza almeno un File System. Deve implementare tutte le chiamate di sistema per gestire i file.

Crea: senza dati, imposta alcuni attribute ➔ touch
Elimina: per liberare spazio su disco ➔ delete
Apri: dopo la creazione, ottiene attributi e indirizzi del disco nella memoria principale
Close: libera lo spazio della tabella utilizzato da attributi e indirizzi
Lettura: di solito dalla posizione corrente del puntatore. È necessario specificare il buffer in cui vengono inseriti i dati
Scrivi: di solito nella posizione corrente
Append: alla fine del file
Seek: mette il puntatore del file in una posizione specifica nel file. Leggi o scrivi da quella posizione in poi
Ottieni attributi: ad es. solo append (a), compresso (c), secure deletion (s) etc ➔ lsattr
Imposta attributi: ad es. attributi di protezione ➔ chattr
Rinominare ➔ rename

DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI

SAPIENZA
UNIVERSITÀ DI ROMA

Operating Systems: File Systems

File: Utilizzo 8/13

System Call: esempio

Copiare abc in xyz.

Copia il file abc in xyz
Se xyz esiste è sovrascritto
Se non esiste, si crea
Utilizza le chiamate di sistema (lettura, scrittura)
Legge e scrive in blocchi 4K
Leggi (chiamata di sistema) in un buffer
Scrivi (chiamata di sistema) dal buffer al file di output

```
/* File copy program. Error checking and reporting is minimal. */

#include <sys/types.h>                  /* include necessary header files */
#include <fcntl.h>
#include <stdlib.h>
#include <unistd.h>

int main(int argc, char *argv[]);       /* ANSI prototype */

#define BUF_SIZE 4096                   /* use a buffer size of 4096 bytes */
#define OUTPUT_MODE 0700                /* protection bits for output file */

int main(int argc, char *argv[])
{
        int in_fd, out_fd, rd_count, wt_count;
        char buffer[BUF_SIZE];

        if (argc != 3) exit(1);                 /* syntax error if argc is not 3 */

        /* Open the input file and create the output file */
        in_fd = open(argv[1], O_RDONLY);        /* open the source file */
        if (in_fd < 0) exit(2);                 /* if it cannot be opened, exit */
        out_fd = creat(argv[2], OUTPUT_MODE);   /* create the destination file */
        if (out_fd < 0) exit(3);                /* if it cannot be created, exit */

    /* Copy loop */
    while (TRUE) {
            rd_count = read(in_fd, buffer, BUF_SIZE); /* read a block of data */
    if (rd_count <= 0) break;                       /* if end of file or error, exit loop */
            wt_count = write(out_fd, buffer, rd_count); /* write data */
            if (wt_count <= 0) exit(4);             /* wt_count <= 0 is an error */
    }

    /* Close the files */
    close(in_fd);
    close(out_fd);
    if (rd_count == 0)                              /* no error on last read */
            exit(0);
    else
            exit(5);                                /* error on last read */
}
```

Operating Systems: File Systems

File: Utilizzo 9/13

Directory

File utilizzati per organizzare una raccolta di file

Chiamate anche cartelle in sistemi operativi strani.

(Soft) Link: puntatore (software) ad altro file (anche directory).

QEMU

```
root@openindiana:/root# ls -l /
total 592
lrwxrwxrwx   1 root     root           7 Oct  3 11:06 bin -> usr/bin
drwxr-xr-x   5 root     sys          512 Oct  3 11:07 boot
drwxr-xr-x   2 root     root         512 Oct 23 13:53 cdrom
drwxr-xr-x 266 root     sys         4608 Oct 23 13:52 dev
drwxr-xr-x   2 root     sys          512 Oct 23 13:52 devices
drwxr-xr-x  76 root     sys         4096 Oct 23 13:53 etc
dr-xr-xr-x   1 root     root           1 Oct 23 13:53 home
drwxr-xr-x   5 jack     staff        512 Oct  3 11:06 jack
drwxr-xr-x  18 root     sys          512 Oct  3 11:07 kernel
drwxr-xr-x  12 root     bin         4608 Oct  3 11:07 lib
drwxr-xr-x   3 root     root         512 Oct 23 13:53 media
drwxr-xr-x   4 root     root         512 Oct  3 11:07 mnt
dr-xr-xr-x   1 root     root           1 Oct 23 13:53 net
lrwxrwxrwx   1 root     root          13 Oct  3 11:07 opt -> /mnt/misc/opt
drwxr-xr-x   5 root     sys          512 Oct  3 11:07 platform
dr-xr-xr-x  44 root     root      260032 Oct 23 13:54 proc
drwx------   2 root     root         512 Oct  3 11:06 root
drwxr-xr-x   2 root     sys         1536 Oct  3 11:07 sbin
drwxr-xr-x   4 root     root         512 Oct  3 11:07 system
drwxrwxrwt   3 root     root         329 Oct 23 13:53 tmp
drwxr-xr-x  34 root     sys         8192 Oct  3 10:40 usr
drwxr-xr-x  41 root     sys         1024 Oct  3 11:07 var
root@openindiana:/root#
```

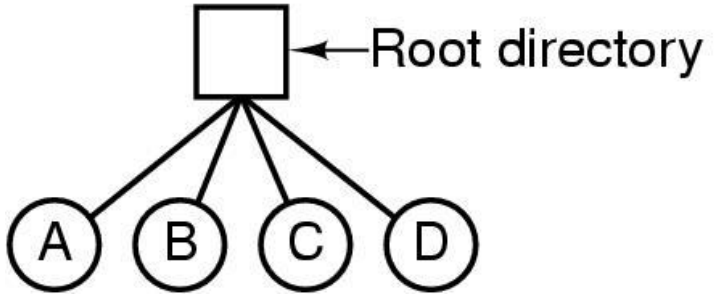
Operating Systems: File Systems

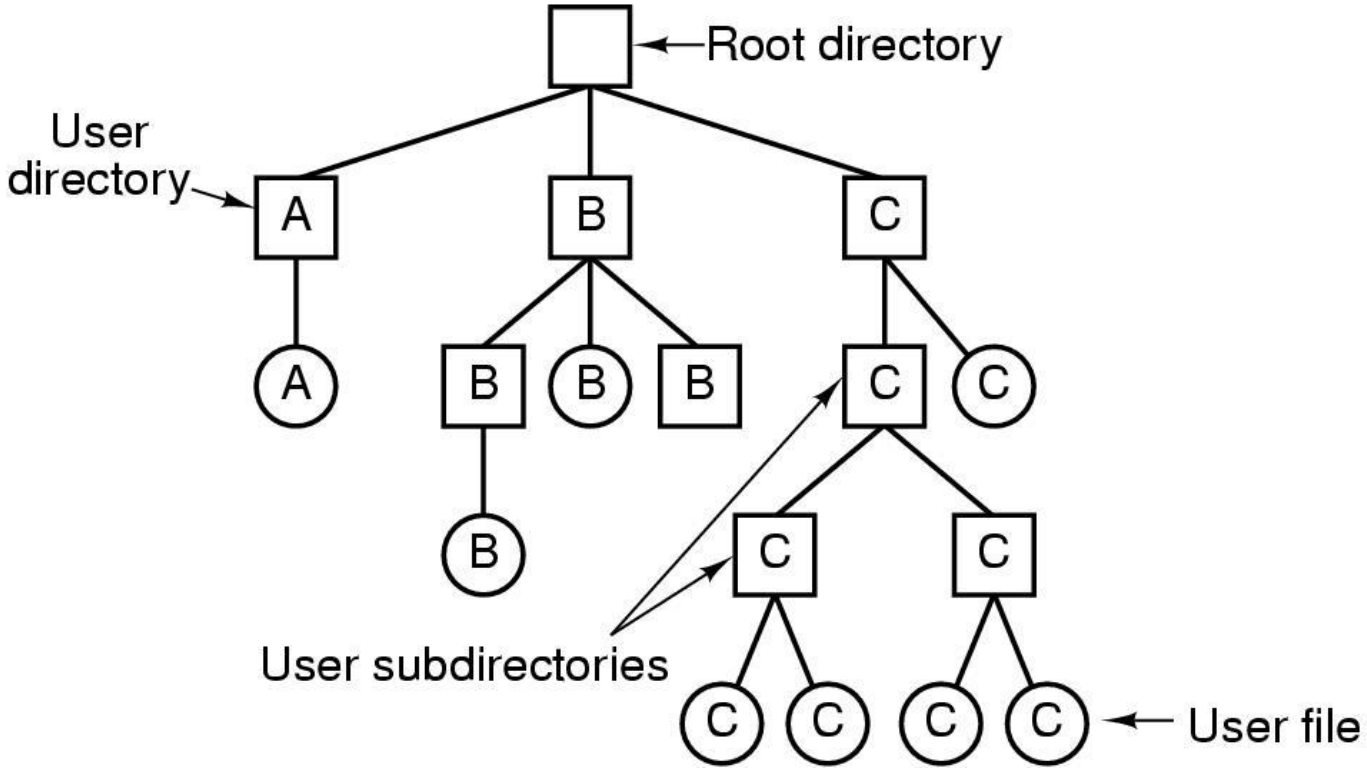
File: Utilizzo 10/13

Directory

Sistema A livello Singolo

Un sistema di directory a livello singolo contenente quattro file





Sistema Gerarchico

Un sistema di directory a livello multiplo

➔ **Path**: persorso sull'albero dei file per arrivare al file.

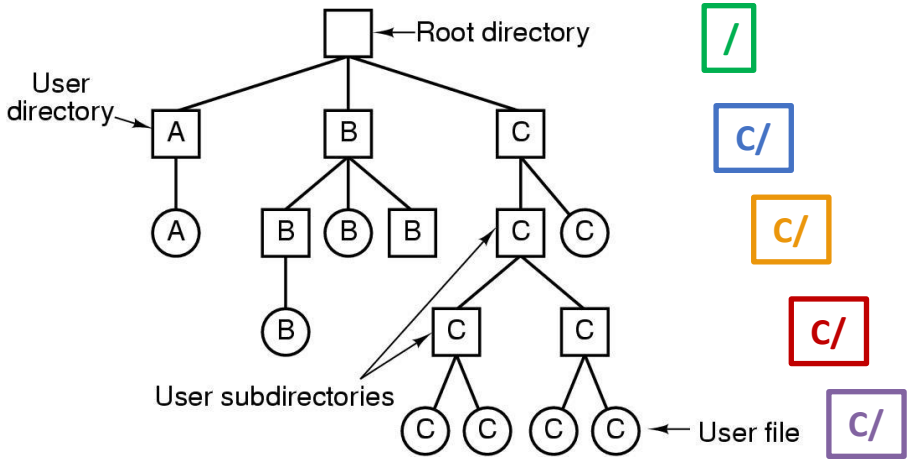
DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI

SAPIENZA
UNIVERSITÀ DI ROMA

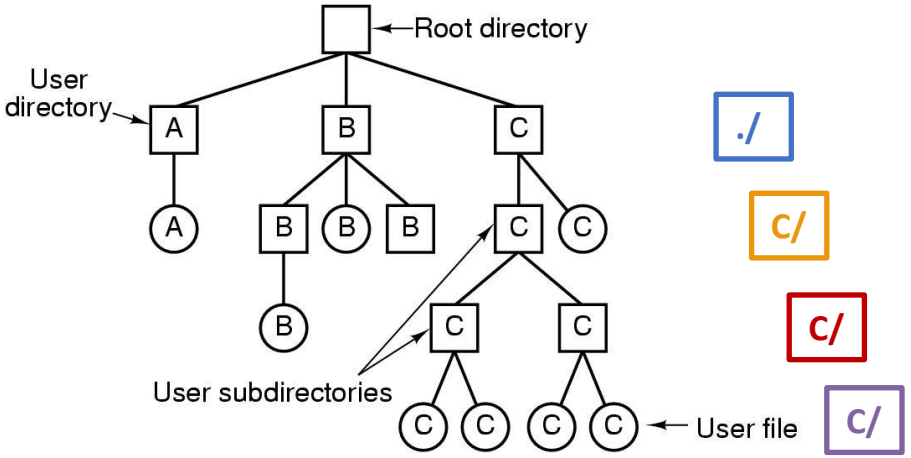
Operating Systems: File Systems

File: Utilizzo 11/13

Directory: Path



Assoluto: componendo tutto il percorso (path), dalla root
/C/C/C/C.txt
(es. /home/docente/sistemioperativi/esame.soluzioni.txt)



Relativo: componendo solo il percorso dalla directory in cui ci si trova (pwd: proper working directory).
Supponendo pwd = User Directory:
./C /C/C.txt
(es. ./docente/sistemioperativi/risposte.txt)

DIPARTIMENTO DI INGEGNERIA INFORMATICA AUTOMATICA E GESTIONALE ANTONIO RUBERTI

SAPIENZA UNIVERSITÀ DI ROMA

Operating Systems: File Systems

File: Utilizzo 11b/13

Path Names: punti

- `..` dice vai al genitore

```
/home/../home/docente/../docente/sistemioperativi/esame.soluzioni.txt
```

Produce le stesso effetto di

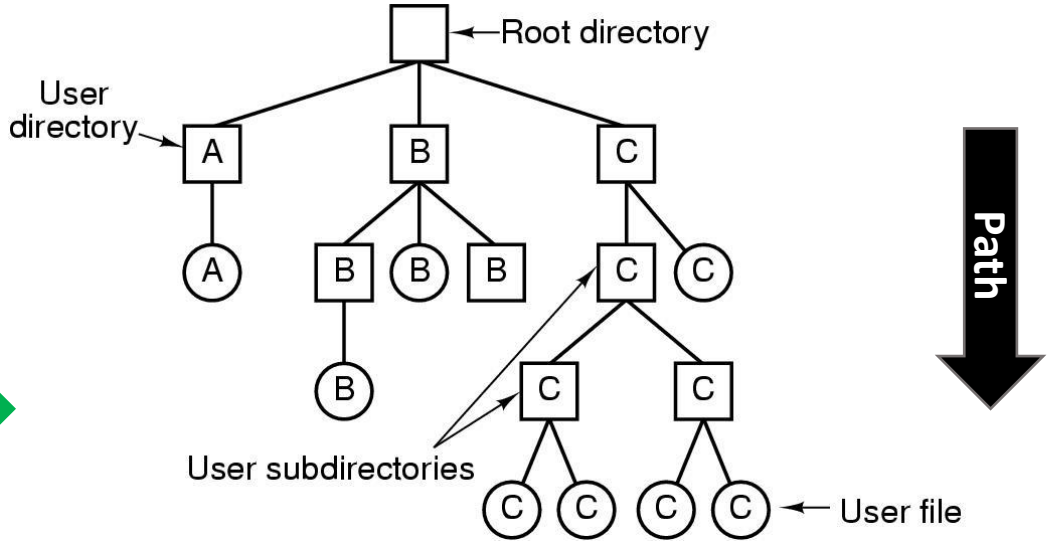
```
/home/docente/sistemioperativi/esame.soluzioni.txt
```

- `.` dice che l'obiettivo è la directory corrente

```
/home/docente/./././sistemioperativi/esame.soluzioni.txt
```

Produce le stesso effetto di

```
/home/docente/sistemioperativi/esame.soluzioni.txt
```



DIPARTIMENTO DI INGEGNERIA INFORMATICA AUTOMATICA E GESTIONALE ANTONIO RUBERTI

SAPIENZA UNIVERSITÀ DI ROMA

Operating Systems: File Systems

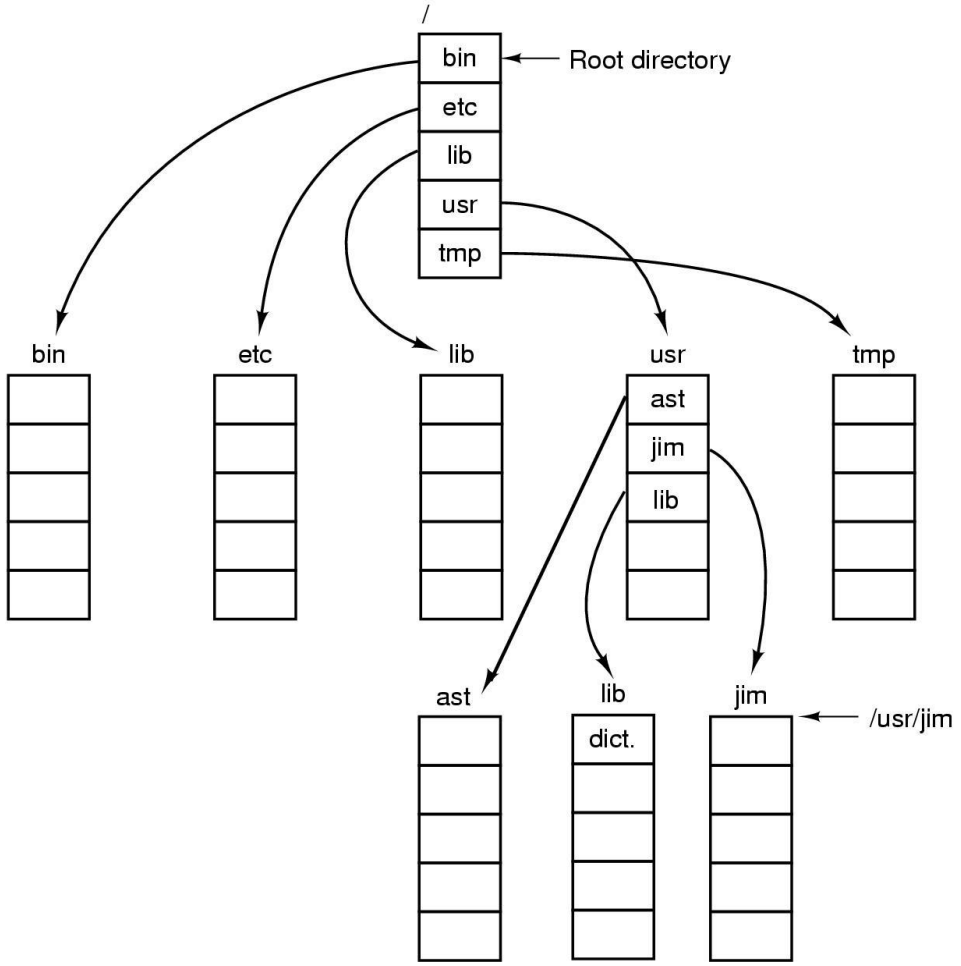
File: Utilizzo 12/13

Path Names: considerazioni

- Interfaccia più «vistosa» dei SO. Permette di tenere gli elementi ordinati. Utilizzata anche nelle piattaforme di collaboration

- La struttura dei file consente di determinare degli «strati» di funzionamento del SO (riprodotta nel setup dei sistemi Slackware):
  1. SO «interno»: funzionamento del sistema e dei servizi di base ➔ `/sbin`, `/slib`, `/bin`, `/lib`, `/etc`, `/var`
  2. SO «esterno»: funzionamento dei servizi a valore aggiunto ➔ `/usr`: `./sbin`, `./slib`, `./bin`, `./lib`
  3. SO «locale»: abilitazione di servizi locali, peculiari al sistema ➔ `/usr/local` , `/opt` , `etc`…

- Unico punto di intervento del chroot jail (indirizza la minaccia dei file contenenti malware). Fileless ➔ protezione non solo a livello di File System



Operating Systems: File Systems

File: Utilizzo 13/13

Directory: System Call

Le tipiche chiamate di sistema per gestire le directory.

Crea: crea directory ➔ mkdir
Elimina: controllare il contenuto della directory ➔ rm -r
Apri: essere eseguito prima di qualsiasi operazione
Close: libera lo spazio della lista fei file contenuti
Lettura: per leggere il contenuto di una directory si deve avere il permesso di potervi accedere, cioé di “eseguirla”
Scrivi: per scrivere il contenuto di una directory si deve avere il permesso di potervi accedere, cioé di “eseguirla”
Link: collega il file a un'altra directory ➔ ln
Unlink: Elimina la voce della directory ➔ rm
Rinominare ➔ rename

Dipartimento di Ingegneria informatica
automatica e gestionale Antonio Ruberti

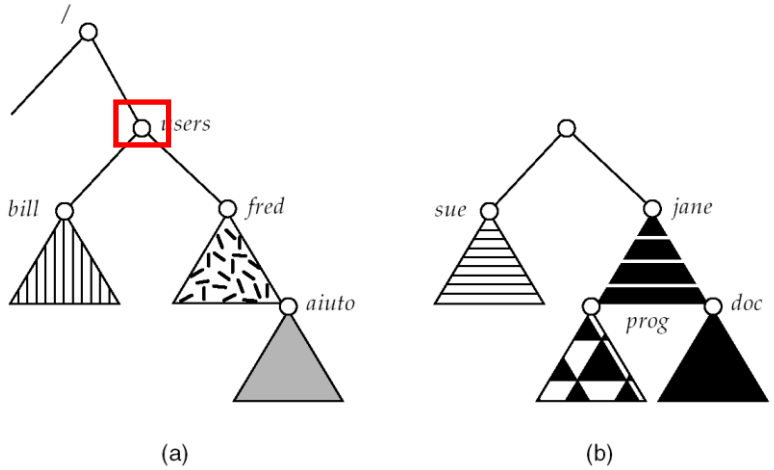
Sapienza
Università di Roma

Operating Systems: File Systems

File: Utilizzo 13b/13

Directory: Mount Point

"inserire" un nuovo dispositivo in una struttura di directory già esistente.



Mount point: una directory in cui un file system può essere montato

Mounted FS: invisibili i file precedentemente presenti fino allo smontaggio del dispositivo

Similmente alle operazioni di apertura dei file, esiste l'operazione di montaggio del file system

- Dato un nuovo dispositivo l'utente specifica una directory esistente (punto di montaggio) nella quale la directory del dispositivo dovrà essere inserita
- Successivamente il SO verifica la validità del file system da montare
- Si verifica che la directory da inserire abbia il formato desiderato
- Il SO annota nella struttura della directory l'inserimento del nuovo file system nel punto di montaggio desiderato

DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI

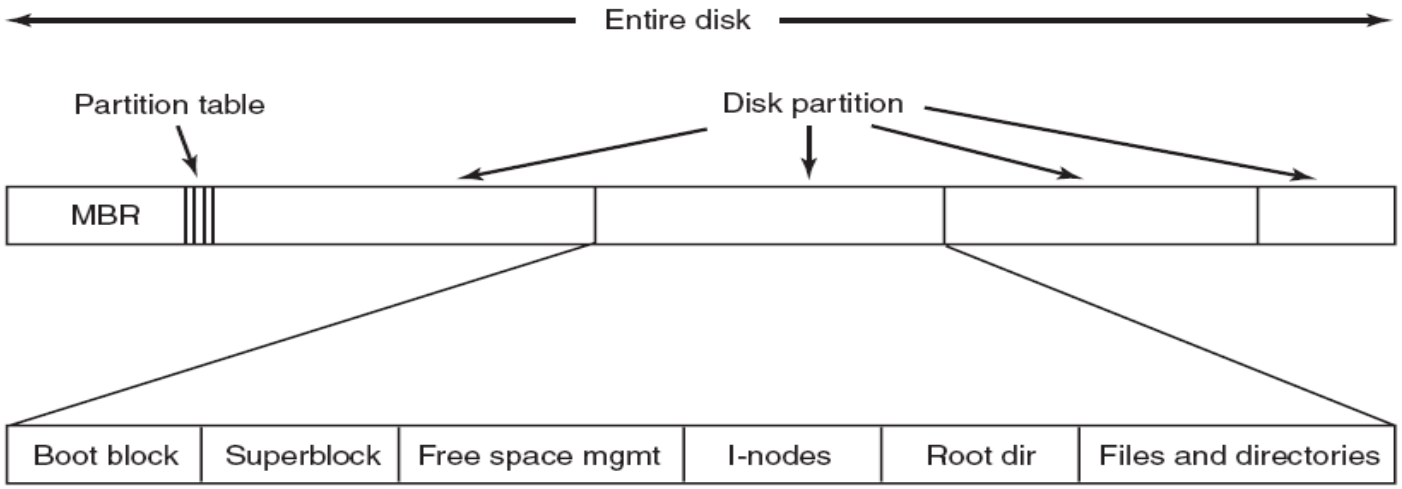
SAPIENZA
UNIVERSITÀ DI ROMA

Operating Systems: File Systems

File: Implementazione 1/30

MBR (Master Boot Record) e Partenza del sistema

File memorizzati su dischi.
Dischi suddivisi in una o più partizioni, con fs separati su ciascuna partizione
Il settore 0 del disco è il Master Boot Record
Usato per avviare il computer
End of MBR ha una tabella delle partizioni: indirizzi di inizio e fine di ogni partizione.
Una delle partizioni è contrassegnata come attiva nella tabella di avvio principale



1. Avvio del Computer ➔ Il BIOS legge/esegue MBR
2. MBR trova la partizione attiva e legge nel primo blocco (blocco di avvio)
3. Il programma nel blocco di avvio individua il sistema operativo per quella partizione e lo legge
4. Tutte le partizioni iniziano con un blocco di avvio

Dipartimento di Ingegneria informatica automatica e gestionale Antonio Ruberti

Sapienza Università di Roma

Operating Systems: File Systems

File: Implementazione 2/30

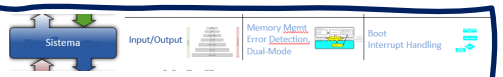
Allocazione dei Blocchi disco ai FIles

Blocco (Cluster) = gruppo di **Settori** (contigui). Quantità minima allocabile.

Allocazione:

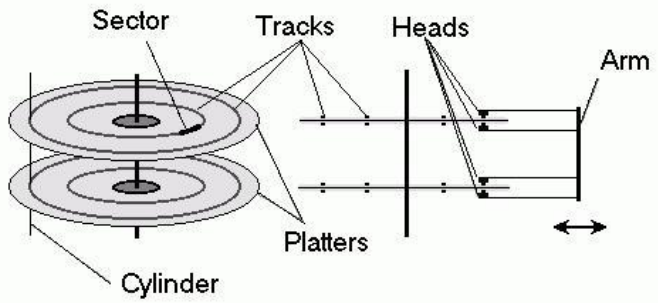
1. **Contigua**
2. **Lista Collegata** (Concatenata)
3. **Indicizzata**: Tavole di Allocazione

Operating Systems: Input/Output



Dischi: Magnetici (HDD) 1/3

- **Cilindro**: tracce presenti ad un certa circonferenza
- **Traccia**: striscia di disco in cui è possible registrare dati
- **Settore**: porzioni di circonferenza in cui è divisa una traccia ➔ #settori tot = #cilindri x #trace x #settori



Parameter	IBM 360-KB floppy disk	WD 18300 hard disk
Number of cylinders	40	10601
Tracks per cylinder	2	12
Sectors per track	9	281 (avg)
Sectors per disk	720	35742000
Bytes per sector	512	512
Disk capacity	360 KB	18.3 GB
Seek time (adjacent cylinders)	6 msec	0.8 msec
Seek time (average case)	77 msec	6.9 msec
Rotation time	200 msec	8.33 msec
Motor stop/start time	250 msec	20 sec
Time to transfer 1 sector	22 msec	17 µsec

Parametri del disco
- Floppy (5’, 12”)
- Hard Disk degli anni 2000

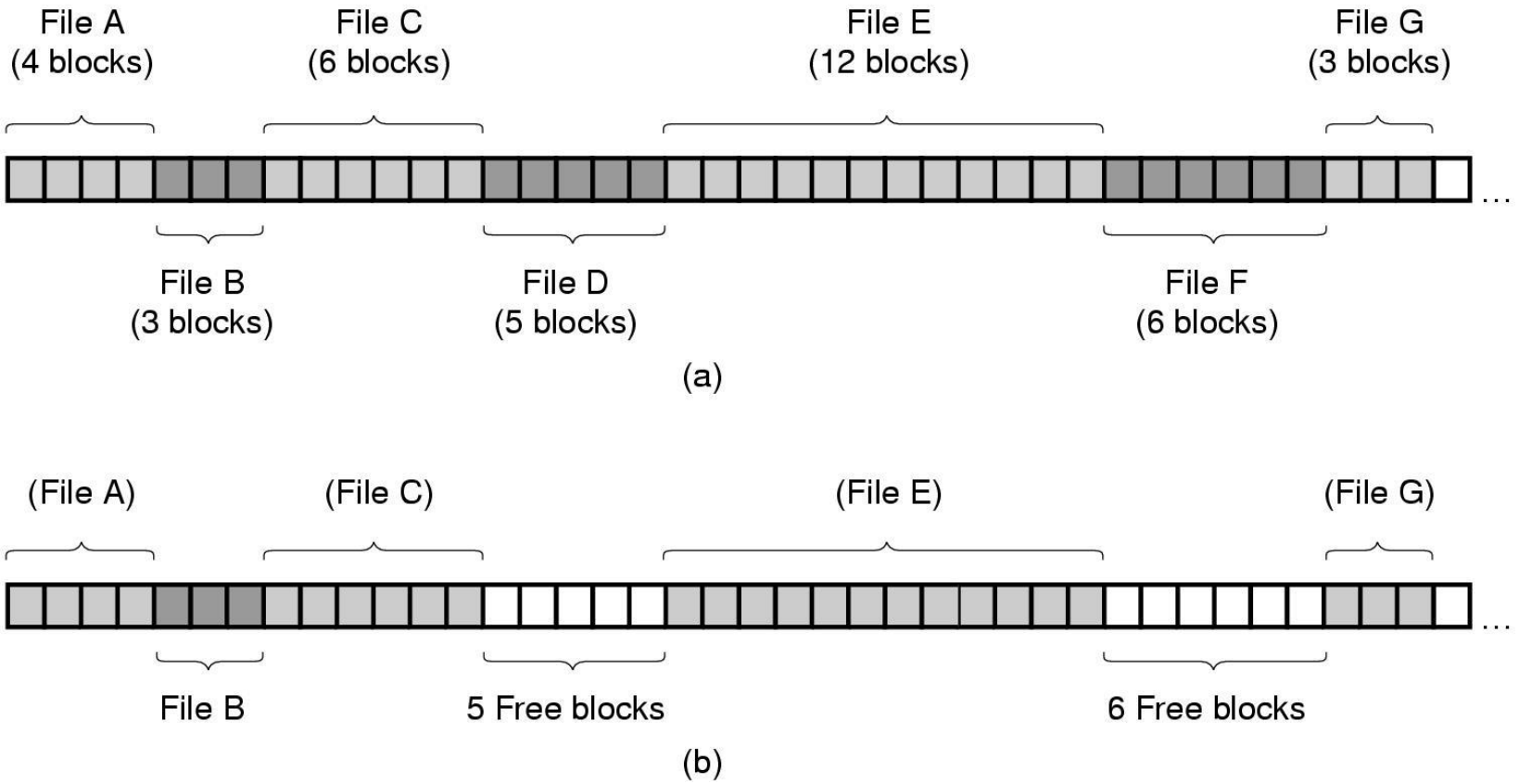
SAPIENZA UNIVERSITÀ DI ROMA

SAPIENZA UNIVERSITÀ DI ROMA

Operating Systems: File Systems

File: Implementazione 3/30

Allocazione Contigua



(a) Allocazione contigua di spazio su disco per 7 file.
(b) Lo stato del disco dopo che i file D e F sono stati rimossi.

Pro

- Facile da implementareLe prestazioni di lettura sono ottime.
- C'è solo bisogno di cercare di individuare il primo blocco nel file. Il resto è facile.

Contro

- Il disco diventa estremamente frammentato nel tempo

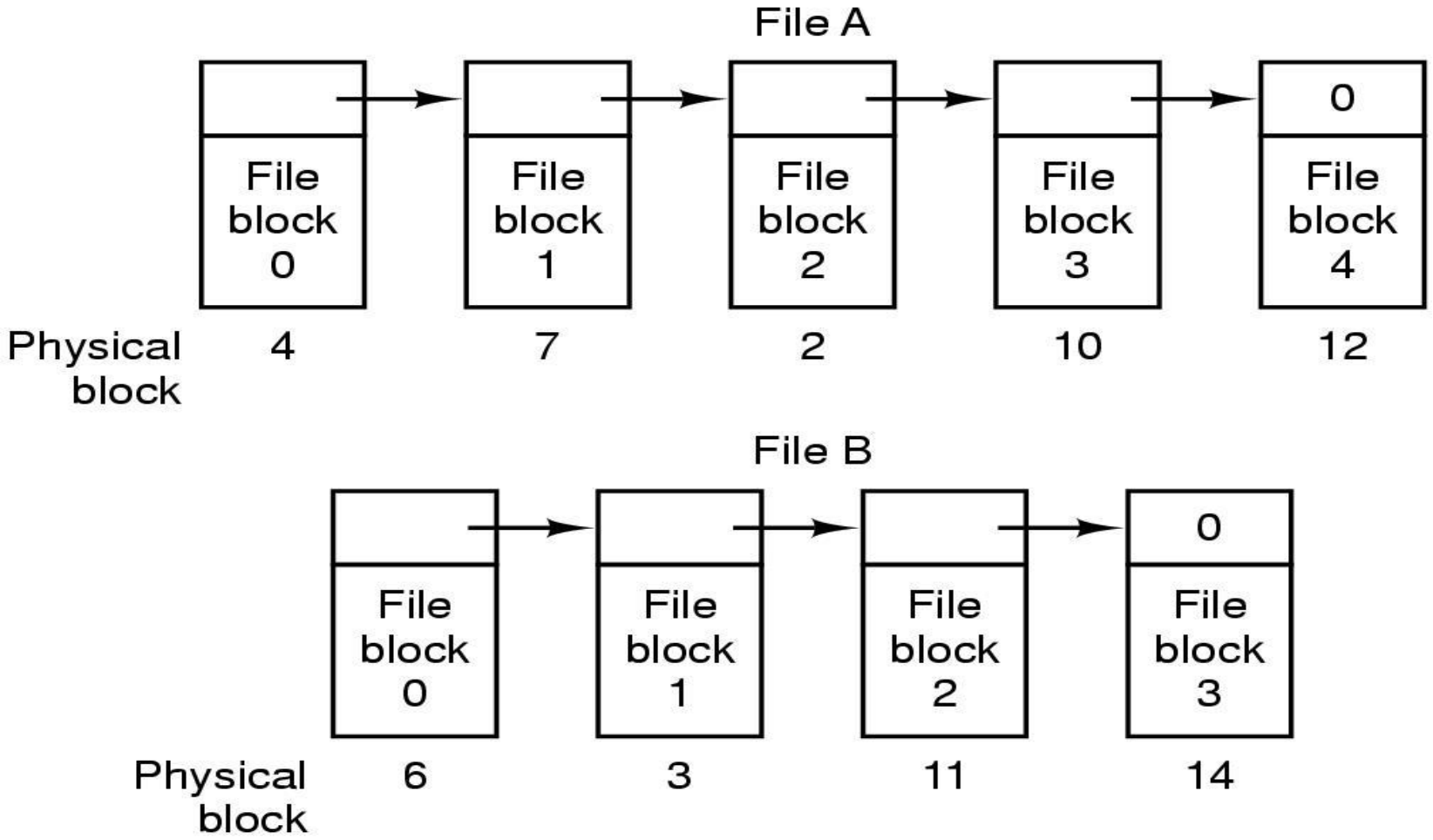
Dipartimento di Ingegneria informatica automatica e gestionale Antonio Ruberti

Sapienza Università di Roma

Operating Systems: File Systems

File: Implementazione 4/30

Allocazione a Lista Collegata



Un file è descritto similmente ad una lista di blocchi disco.

Pro

- Non genera frammentazione.

Contro

- L'accesso è lento. C'è bisogno di inseguire i puntatori per arrivare a un blocco

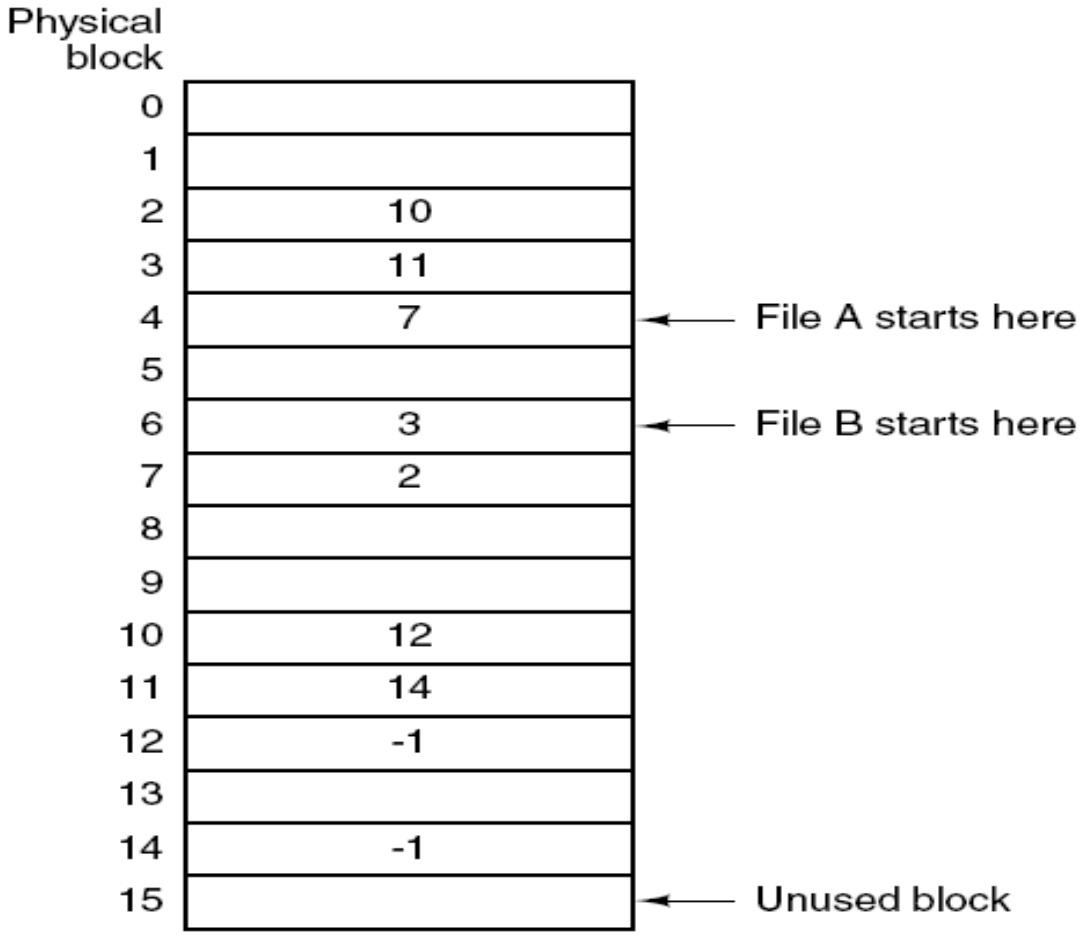
Dipartimento di Ingegneria informatica automatica e gestionale Antonio Ruberti

Sapienza Università di Roma

Operating Systems: File Systems

File: Implementazione 5/30

Allocazione a Lista Collegata con Tabella in Memoria



Puntatori nella tabella in memoria
Tabella di allocazione dei file (FAT)

Pro

- Facile implementazione.

Contro

- La tabella diventa davvero grande (es. un disco da 200 GB con blocchi da 1 KB richiede una tabella da 600 MB) ➔ blocchi di grandi dimensioni (spreco di spazio)
- La crescita della dimensione della tabella è lineare con la crescita della dimensione del disco

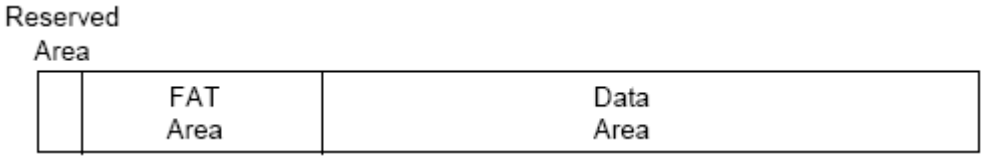
Dipartimento di Ingegneria informatica automatica e gestionale Antonio Ruberti

Sapienza Università di Roma

Operating Systems: File Systems

File: Implementazione 5b/30

FAT16 e FAT32



Puntatori nella tabella in memoria
Tabella di allocazione dei file (FAT).

FAT12: 12 bit per l'indirizzamento ➔ 4096 cluster
Filename: 8.3 caratteri

FAT16: 16 bit per l'indirizzamento ➔ 65535 cluster
Filename: 8.3 caratteri
VFAT -> Filename: 255 caratteri

FAT32: 28 bit (32-4 riservati) per l'indirizzamento
➔ 256 Milioni di cluster
Filename: 255 caratteri

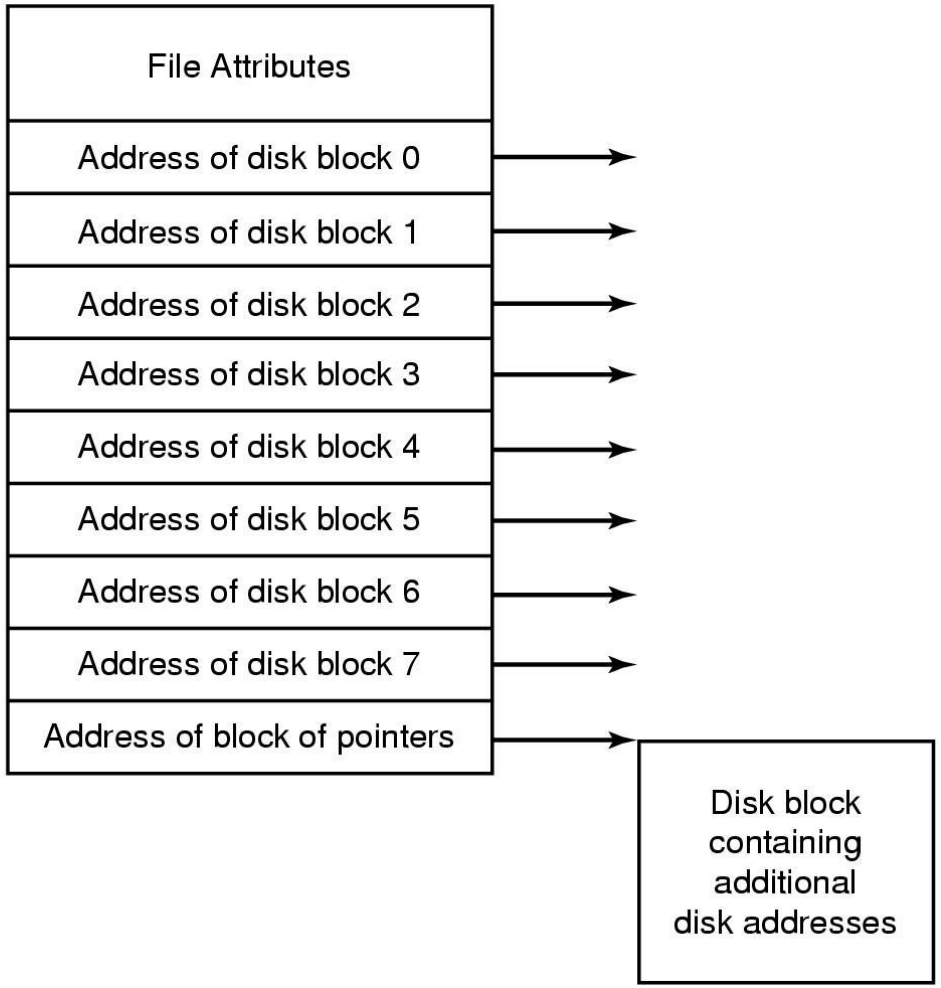
Block size	FAT-12	FAT-16	FAT-32
0.5 KB	2 MB		
1 KB	4 MB		
2 KB	8 MB	128 MB	
4 KB	16 MB	256 MB	1 TB
8 KB		512 MB	2 TB
16 KB		1024 MB	2 TB
32 KB		2048 MB	2 TB

Microsoft limita l'utilizzo a 2TB

Operating Systems: File Systems

File: Implementazione 6/30

Indicizzazione con Tabella di Allocazione (i-node)



Mantenere la struttura dei dati in memoria solo per i file attivi
La struttura dei dati elenca gli indirizzi dei dischi dei blocchi e gli attributi dei file
K file attivi, N blocchi per file => k*n blocchi max!!
Per rendere N più piccolo, l'ultima voce nella tabella punta al blocco del disco che contiene puntatori ad altri blocchi del disco

Pro

- Risolve il problema della crescita.

Contro

- Implementazione meno facile

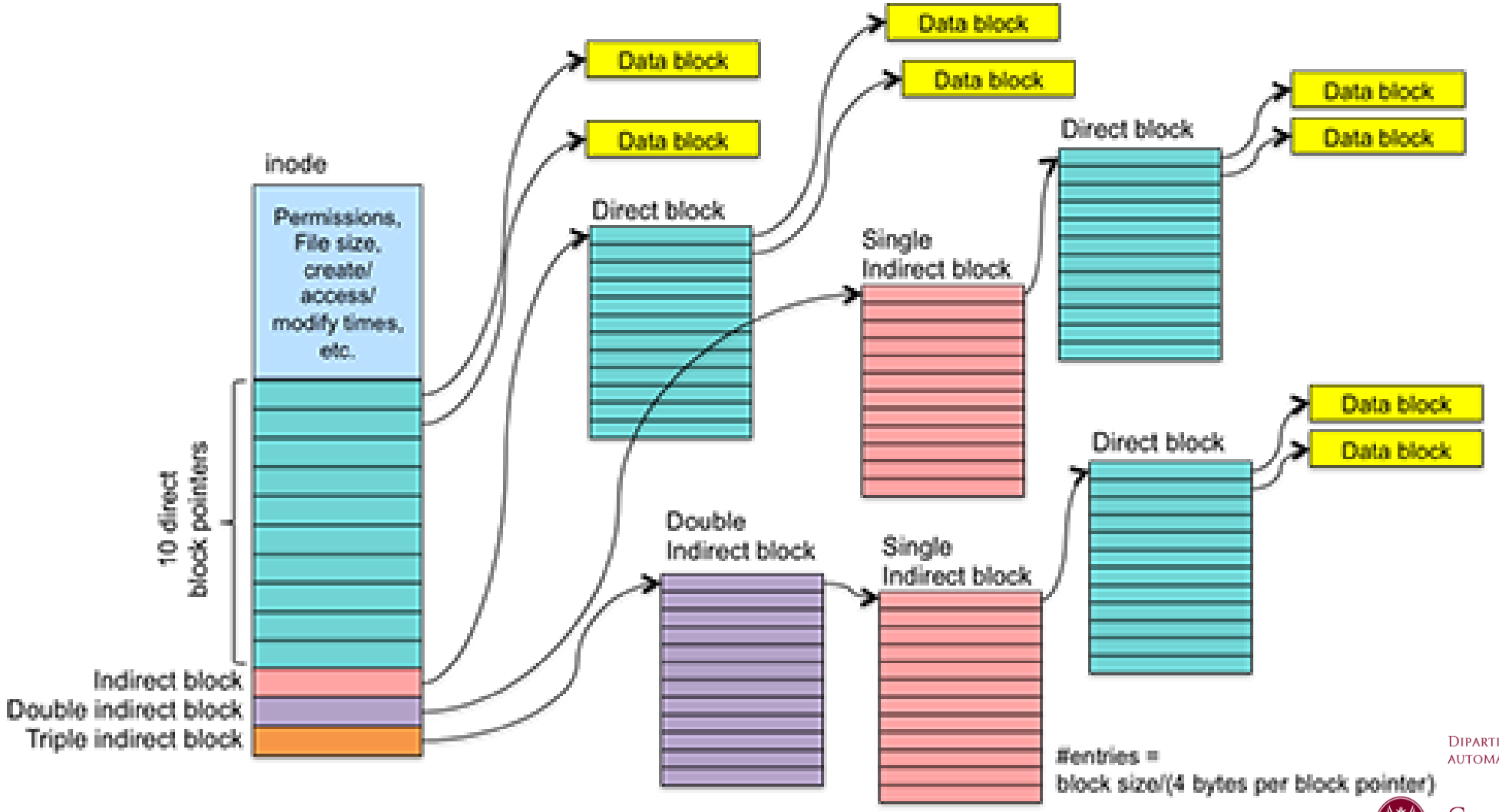
Dipartimento di Ingegneria informatica automatica e gestionale Antonio Ruberti

Sapienza Università di Roma

Operating Systems: File Systems

File: Implementazione 6b/30

Indicizzazione con Tabella di Allocazione (i-node)



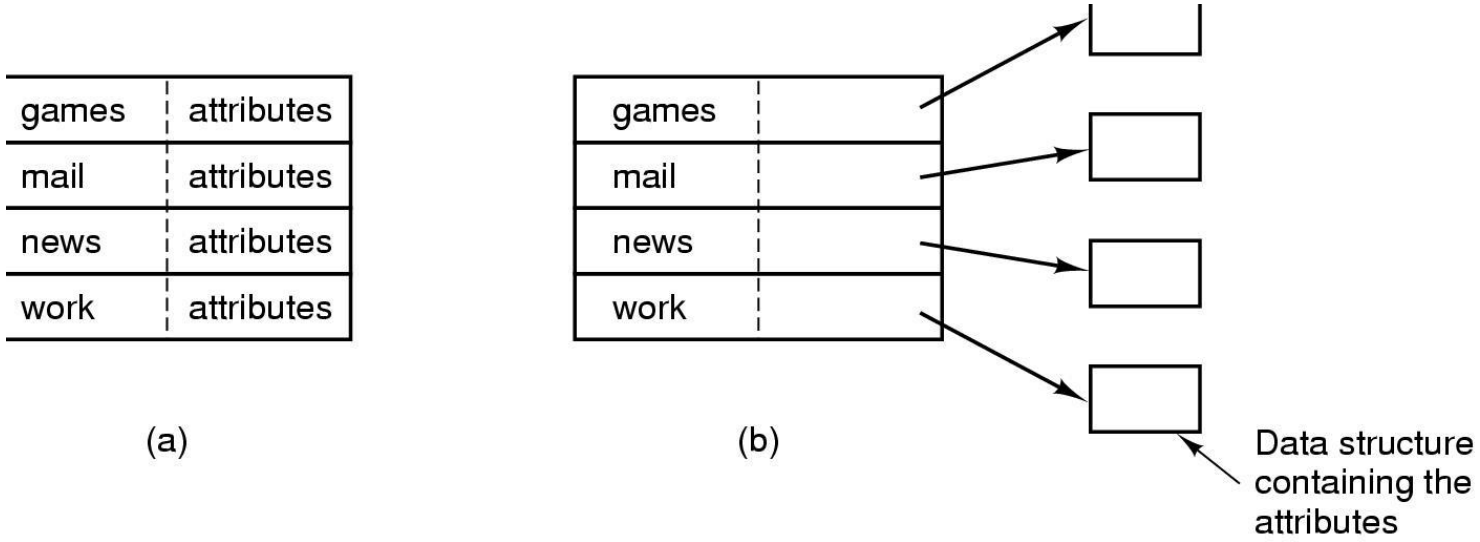
Dipartimento di Ingegneria informatica automatica e gestionale Antonio Ruberti

Sapienza Università di Roma

Operating Systems: File Systems

File: Implementazione 7/30

Directory



nome del percorso utilizzato per individuare la directory.
La directory specifica gli indirizzi dei blocchi fornendo:

- Indirizzo del primo blocco (contiguo)
- Numero del primo blocco (collegato)
- Numero di i-node

(a) entry di dimensione fissa con gli indirizzi e gli attributi del disco (DOS)
(b) ogni entry si riferisce a un i-node. La entry della directory contiene attributi. (Unix)

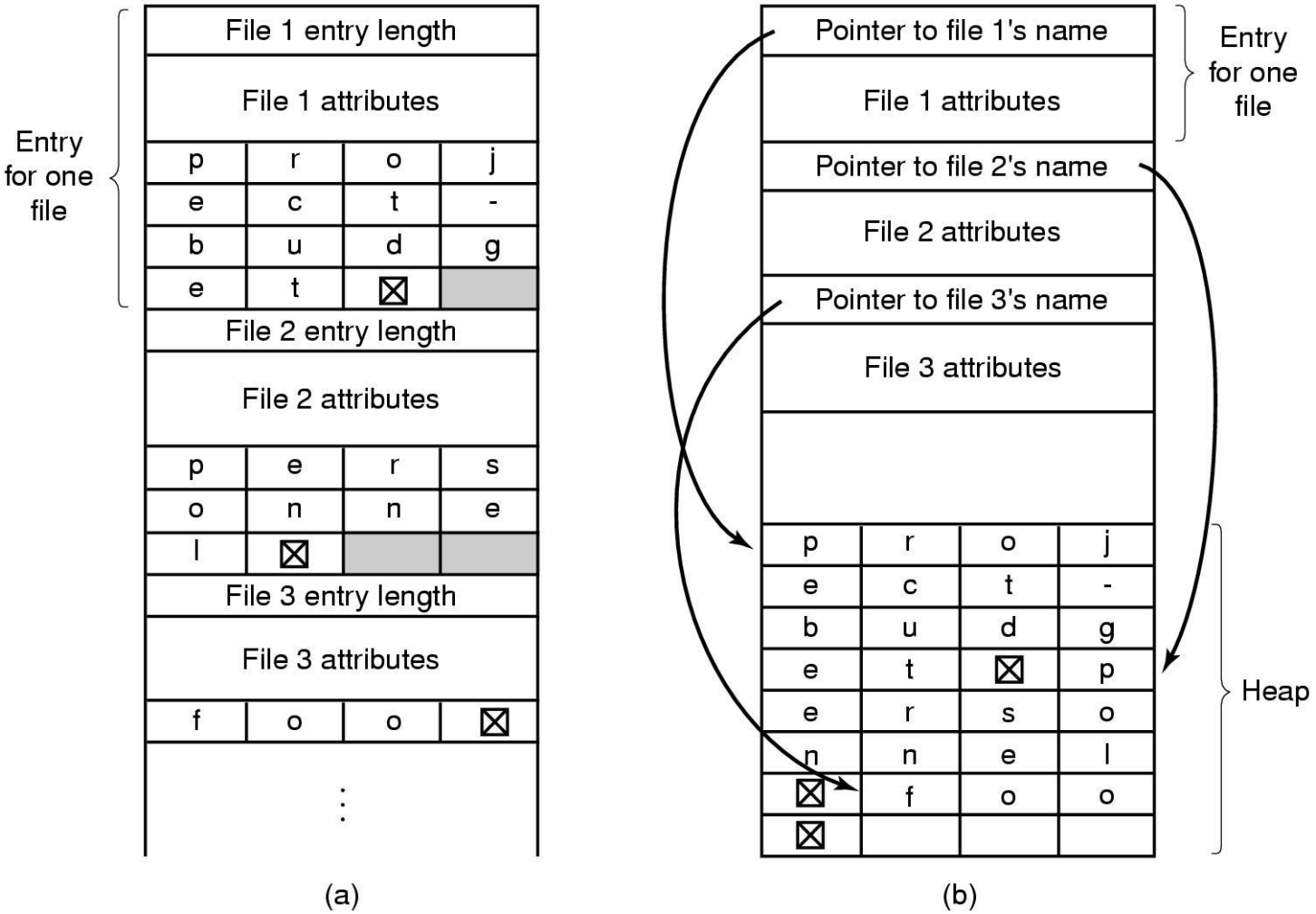
DIPARTIMENTO DI INGEGNERIA INFORMATICA AUTOMATICA E GESTIONALE ANTONIO RUBERTI

SAPIENZA UNIVERSITÀ DI ROMA

Operating Systems: File Systems

File: Implementazione 8/30

Directory con nomi dei file lunghi



Il problema è che i nomi sono diventati molto lunghi.
Due approcci

a. Intestazione fissa seguita da nomi di lunghezza variabile

b. Il puntatore heap punta ai nomi

Due modi per gestire nomi di file lunghi in una directory.
(a) In linea.
(b) (b) In un mucchio.

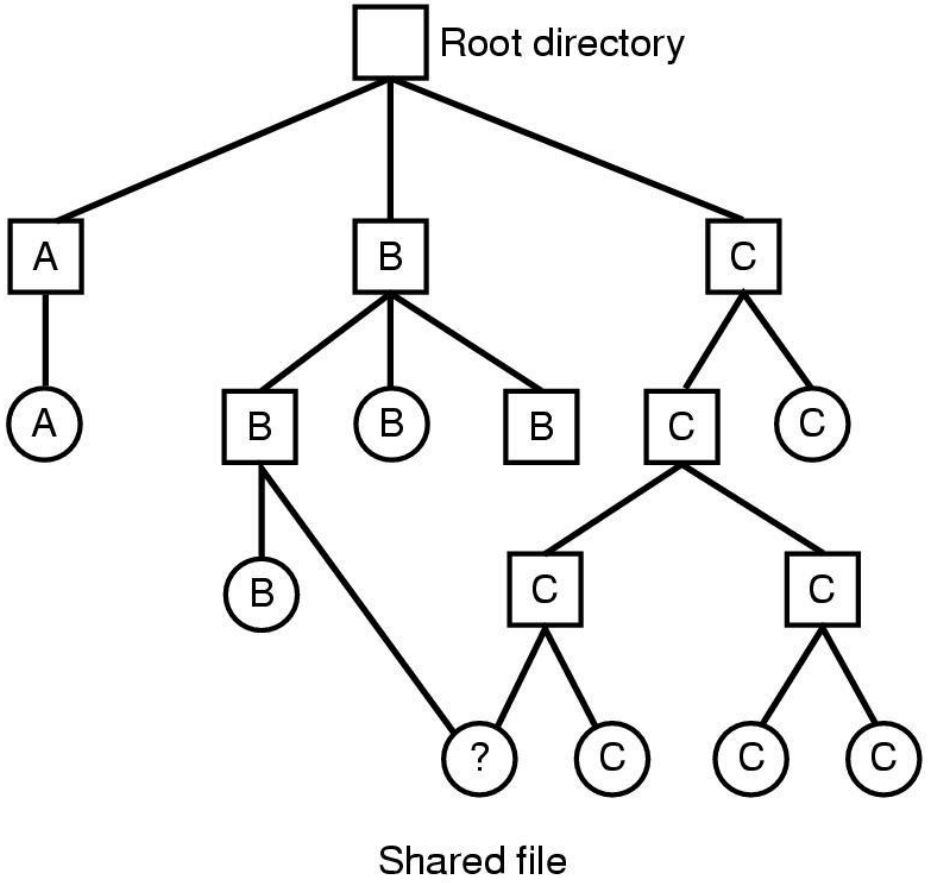
DIPARTIMENTO DI INGEGNERIA INFORMATICA AUTOMATICA E GESTIONALE ANTONIO RUBERTI

SAPIENZA UNIVERSITÀ DI ROMA

Operating Systems: File Systems

File: Implementazione 9/30

File con link simbolici: Directed Acyclic Tree (DAG)



collegamento simbolico - un file speciale inserito nella directory di B se C è il proprietario. Contiene il nome del percorso del file a cui è collegato ➔ file «l»

Pro

- Risolve il problema
- può puntare a file su un'altra macchina

Contro

- Troppi collegamenti simbolici richiedono tempo per essere eseguiti

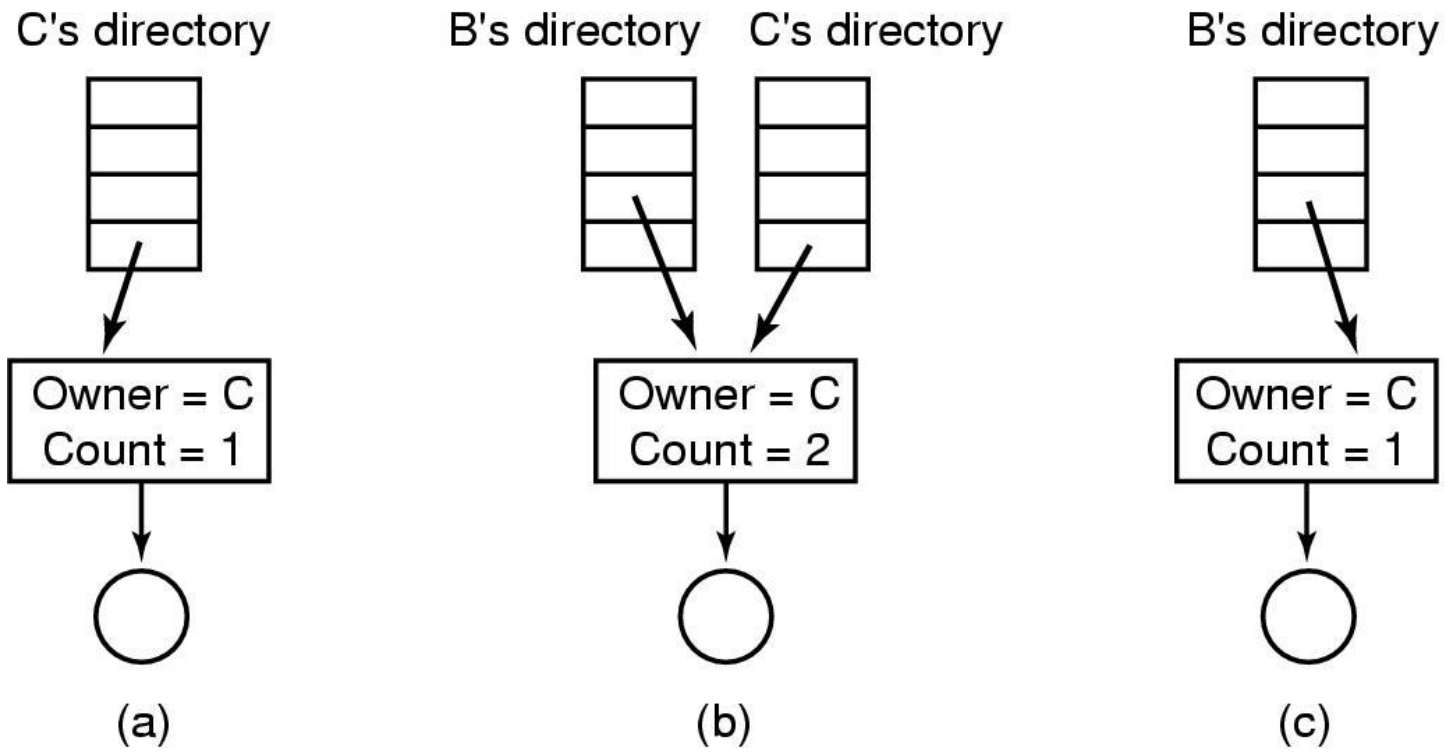
Dipartimento di Ingegneria informatica automatica e gestionale Antonio Ruberti

Sapienza Università di Roma

Operating Systems: File Systems

File: Implementazione 10/30

File con hard link: i-node



(a) Situazione prima del collegamento.
(b) Dopo la creazione del collegamento.
(c) Dopo che il proprietario originale ha rimosso il file.

Se C rimuove il file, la directory di B punta ancora a i-node per il file condiviso
Se l'i-node viene riutilizzato per un altro file, l'ingresso di B punta a un i-node errato
La soluzione è lasciare i-node e ridurre il numero di proprietari

Pro

- Risolve il problema

Contro

- Possibile sbagliarsi e rimuovere l'ultimo link ➔ cancellazione del file

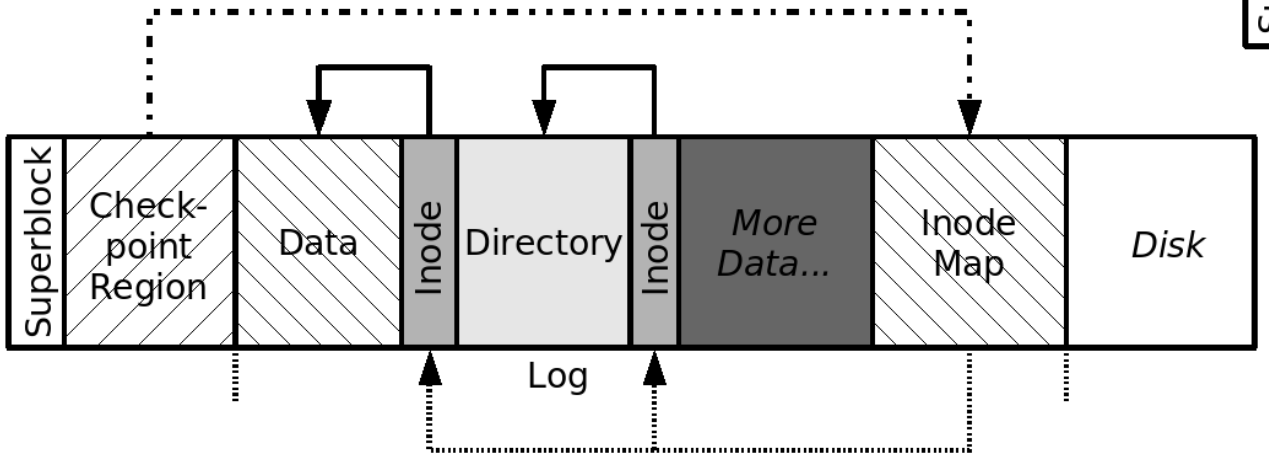
Dipartimento di Ingegneria informatica automatica e gestionale Antonio Ruberti
Sapienza Università di Roma

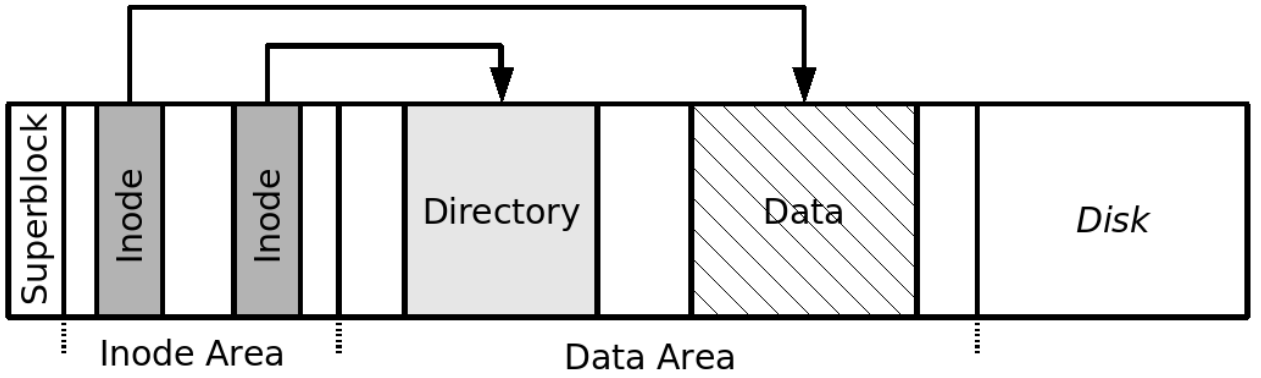
Operating Systems: File Systems

File: Implementazione 11/30

Log Structured File System

Struttura di un classico FS ad i-node





Struttura di un LFS (Log Structured File System)

memorizzare tutte le informazioni in sequenza su disco in una struttura di registro. Ciò include tutti i blocchi di dati, blocchi di inode, blocchi di directory, blocchi indiretti, ecc.
Le nuove informazioni vengono memorizzate in una cache di scrittura e scritte su disco in un'unica operazione di scrittura di grandi dimensioni.
In questo modo molte, piccole scritture sincrone di filesystem tradizionali vengono tradotte in poche, grandi operazioni di scrittura sequenziali asincrone.

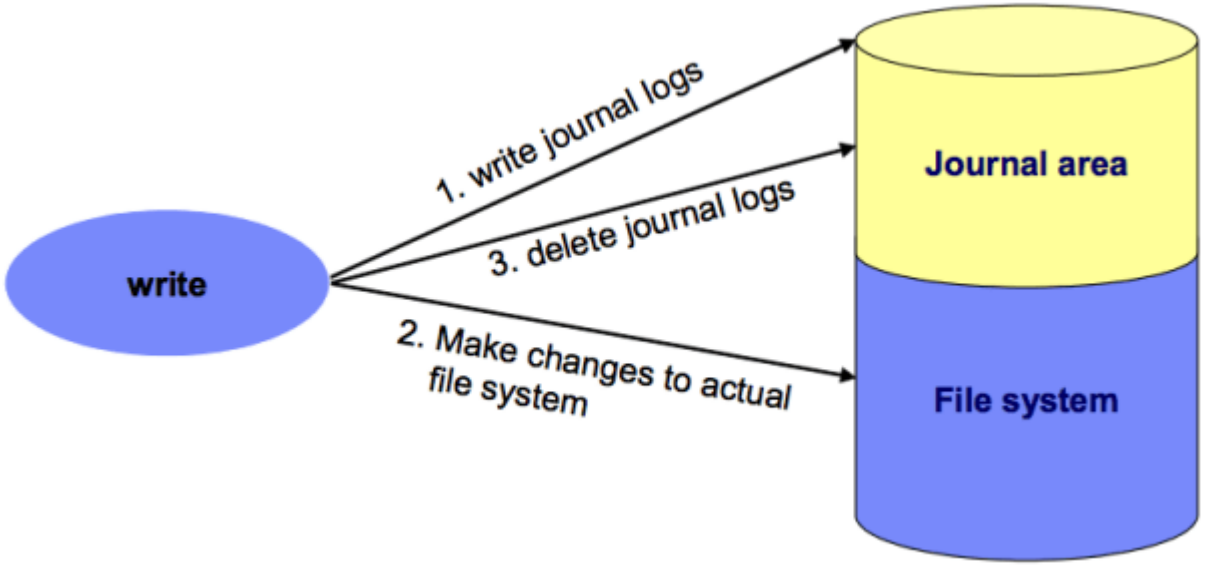
Operating Systems: File Systems

File: Implementazione 12/30

Journaling File System

Tenere un diario (cioè un elenco) di azioni
prima di eseguirle,
Scrivi il diario su disco,
Quindi, eseguire le azioni.
Cancellare il diario

Può recuperare da un incidente!
NTFS (Windows) e Linux utilizzano il
journaling.



In informatica il journaling è una tecnica utilizzata da molti file system moderni per preservare l'integrità dei dati da eventuali cadute di tensione.

Derivata dal mondo dei database, il journaling si basa infatti sul concetto di **transazione** dove ogni scrittura su disco è interpretata dal file system come tale.

DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI

SAPIENZA
UNIVERSITÀ DI ROMA

Operating Systems: File Systems

File: Implementazione 13/30

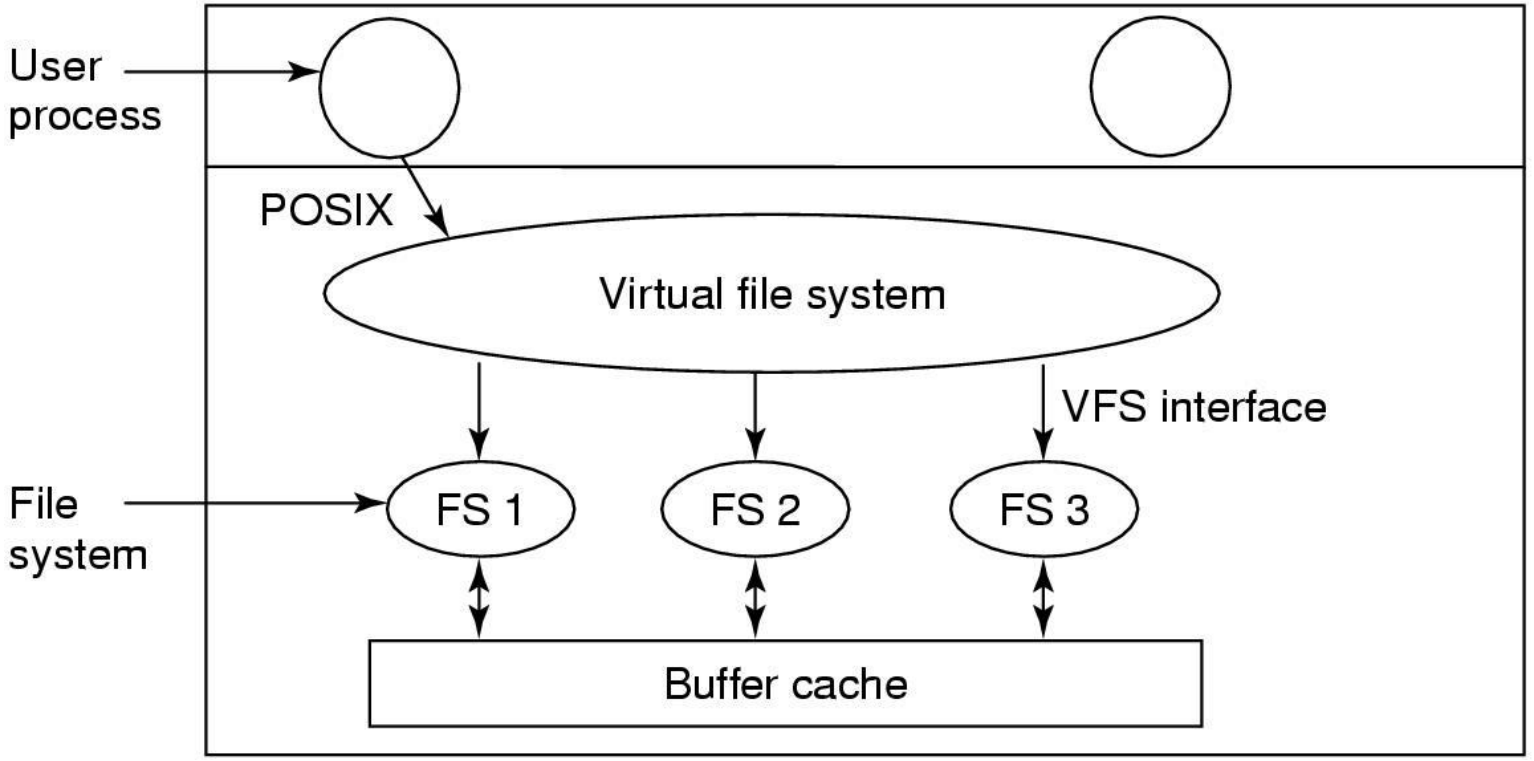
Virtual File System

Più fs sulla stessa macchina.

Unix si integra in VFS:

- Chiamate Vfs dall'utente
- Supporta il file system di rete: il file può essere su una macchina remota

Su Windows è più difficile: specifica l'unità



Dipartimento di Ingegneria informatica automatica e gestionale Antonio Ruberti

Sapienza Università di Roma

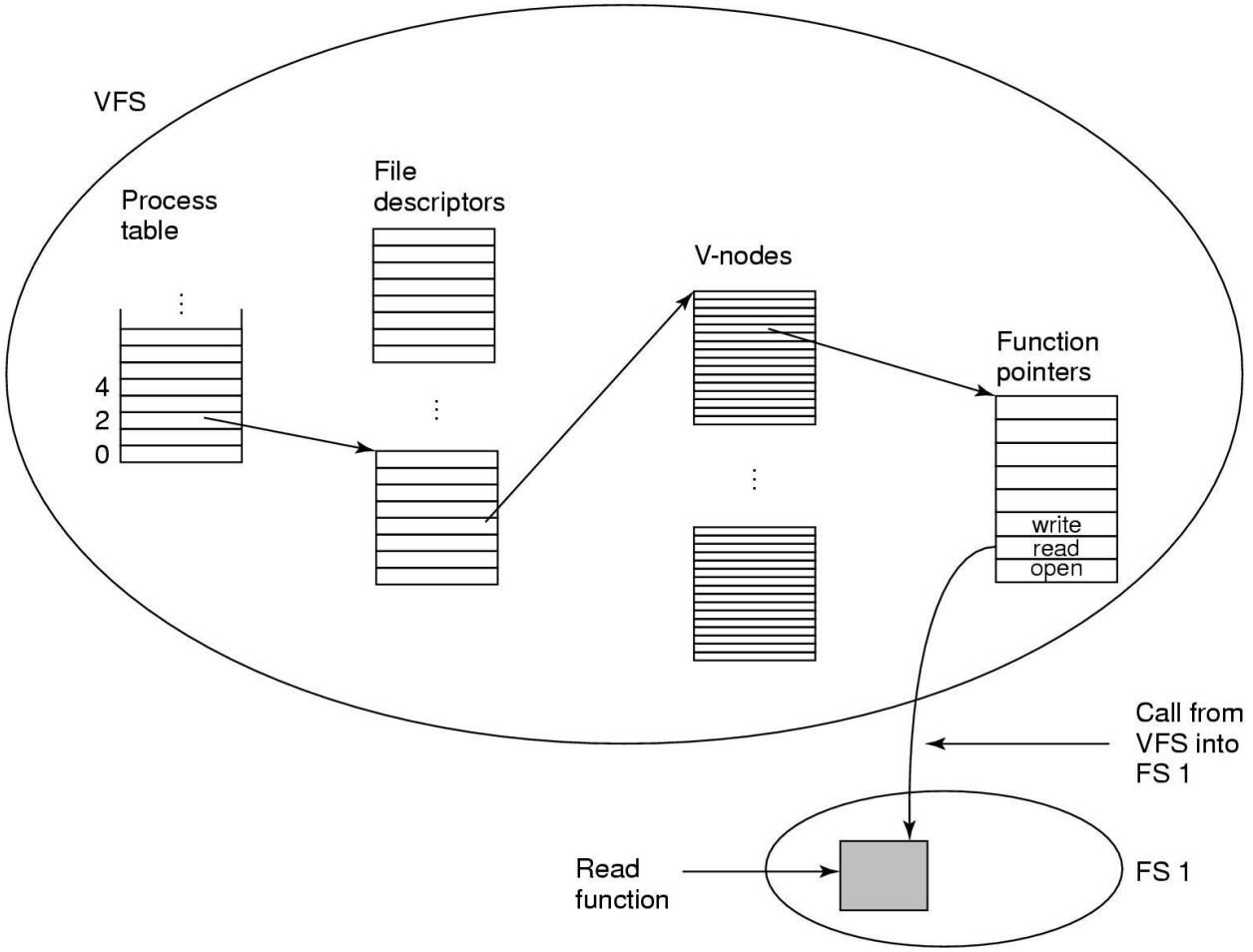
Operating Systems: File Systems

File: Implementazione 14/30

Virtual File System

Registri del file system con VFS (ad es. all'avvio)

- Al momento della registrazione, fs fornisce un elenco di indirizzi di chiamate di funzione che vfs vuole
- Vfs ottiene informazioni dal nuovo i-node fs e le inserisce in un v-node
- Entra nella tabella fd (File Descriptor) per il processo
- Quando il processo emette una chiamata (ad esempio read), i puntatori di funzione puntano a chiamate di funzione concrete



Una vista semplificata delle strutture dati e del codice utilizzati dal VFS e dal file system concreto per eseguire una lettura

DIPARTIMENTO DI INGEGNERIA INFORMATICA AUTOMATICA E GESTIONALE ANTONIO RUBERTI
SAPIENZA UNIVERSITÀ DI ROMA

Operating Systems: File Systems

File: Implementazione 14b/30

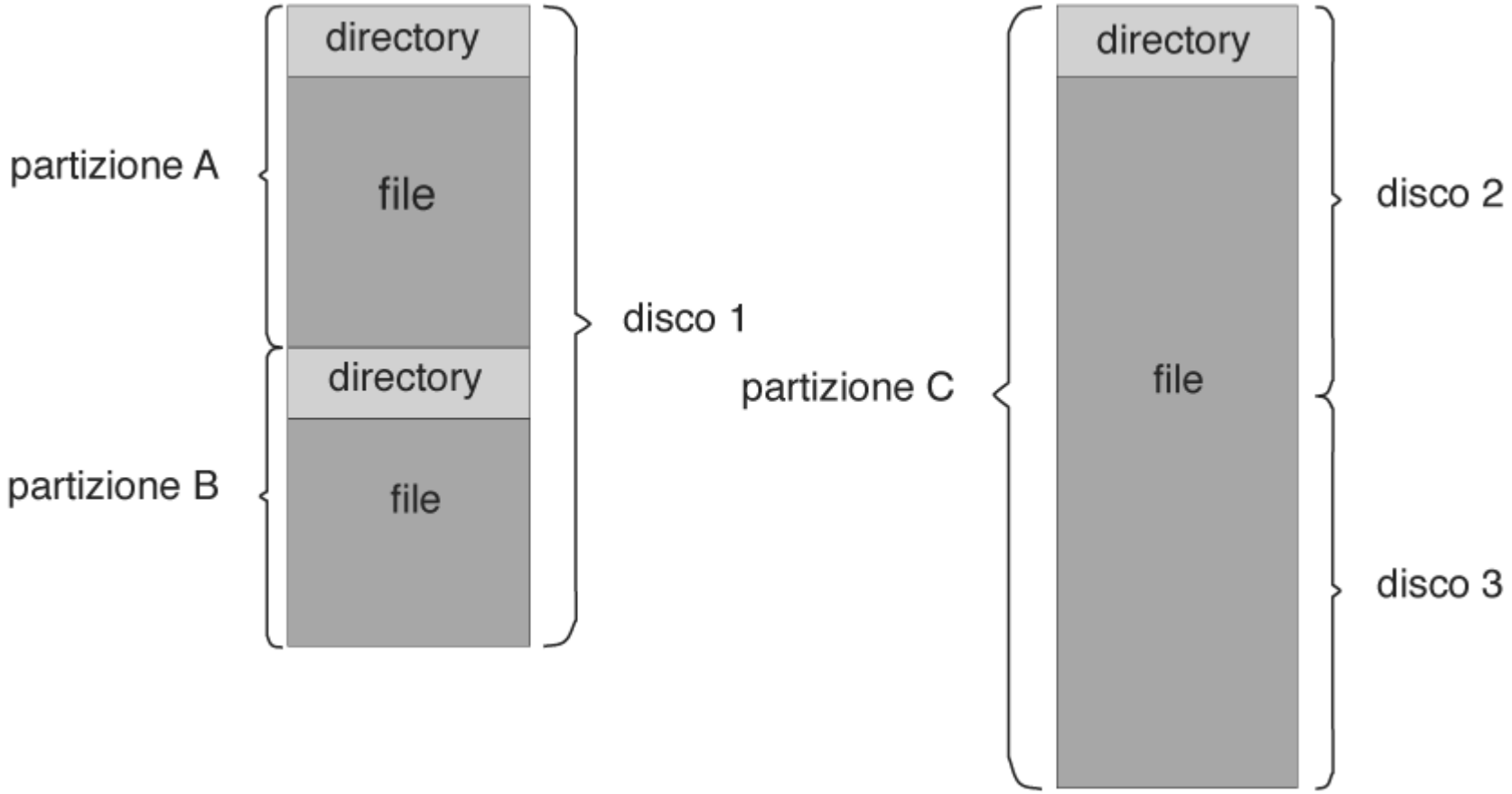
Virtual File System: partizioni

Un dispositivo di memorizzazione può essere diviso in più parti, chiamate partizioni. Utile per:

- Limitare le dimensioni del file-system
- Avere diversi tipi di file-system sullo stesso calcolatore
- Liberare per altri scopi parte del disco (e.g. Swap)

Ogni partizione contenente un file system è chiamata volume.

RAID: come visto in «9. Input/Output», Più dispositivi di memorizzazione possono essere raccolti in insiemi



Ogni volume contiene una tabella dei file (simboleggiata come «directory» nel disegno) che memorizza gli attributi di tutti i file memorizzati.

Operating Systems: File Systems

File: Implementazione 14c/30

Virtual File System: remoti

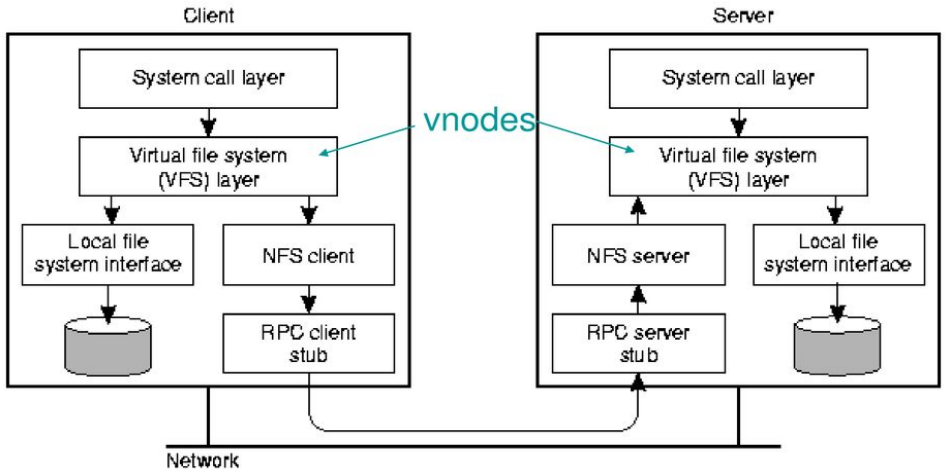
File System di rete:

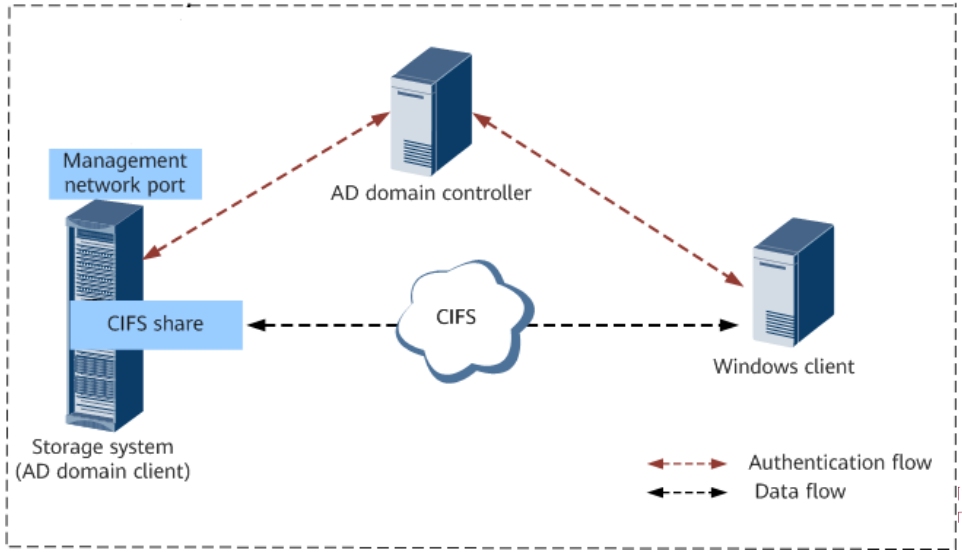
Unix: NFS

- Nome: Network File System
- Sviluppato: Sun Microsystems 1984
- NFSv4: RFC 7530
- Non usa più Portmap

Windows: CIFS
Dialetto del SMB (Server Message Block), sviluppato da IBM, nel 1983

- Nome: Common Internet File Systems
- Sviluppato: Microsoft 2000
- SMBv3: MS-CIFS
- Compatibilità Unix: Samba





Operating Systems: File Systems

File: Implementazione 14d/30

Virtual File System: file archivio

ECMA-376 : Office Open XML File Format

Europen Computer Manufactoring Association

User Interface: dimensione atta a contenere tutte le informazioni utili per l'utente

C:\Paolo\WRK\21.Sapienza-SO\SO-Corso\SO-8linux11.pptx\

File Edit View Favorites Tools Help

Add Extract Test Copy Move Delete Info

C:\Paolo\WRK\21.Sapienza-SO\SO-Corso\SO-8linux11.pptx\

Name	Size	Modified	Packed Size
docProps	13 351		7 692
ppt	18 758 897		5 290 576
_rels	885		273
[Content_Types]....	9 239	1980-01-01 00:00	656

Name	Size	Modified	Packed Size
drawings	1 736		784
embeddings	121 856		36 489
media	17 872 039		5 112 781
slideLayouts	56 887		15 973
slideMasters	15 867		2 339
slides	663 913		117 449
theme	8 399		1 732
_rels	6 770		535
presentation.xml	4 749	1980-01-01 00:00	880
presProps.xml	1 089	1980-01-01 00:00	482
tableStyles.xml	4 779	1980-01-01 00:00	740
viewProps.xml	813	1980-01-01 00:00	392

Media

Name	Size	Modified	Packed Size
image1.jpeg	65 646	1980-01-01 00:00	65 646
image2.png	36 429	1980-01-01 00:00	36 429
image3.png	34 241	1980-01-01 00:00	34 241
image4.png	12 738	1980-01-01 00:00	12 738
image5.png	5 518	1980-01-01 00:00	5 518
image6.png	69 708	1980-01-01 00:00	69 708
image7.emf	3 599 080	1980-01-01 00:00	1 116 582
image8.png	9 014	1980-01-01 00:00	9 014
image9.png	80 308	1980-01-01 00:00	80 308
image10.jpeg	29 116	1980-01-01 00:00	29 116
image11.png	85 067	1980-01-01 00:00	85 067
image12.png	26 542	1980-01-01 00:00	26 542
image13.png	24 968	1980-01-01 00:00	24 968
image14.png	6 381	1980-01-01 00:00	6 381
image15.png	27 901	1980-01-01 00:00	27 901
image16.png	32 429	1980-01-01 00:00	32 429
image17.emf	1 280 632	1980-01-01 00:00	346 058
image18.emf	1 881 896	1980-01-01 00:00	369 740
image19.jpeg	145 685	1980-01-01 00:00	145 685
image20.wmf	1 202 568	1980-01-01 00:00	11 711
image21.png	76 927	1980-01-01 00:00	76 927
image22.png	12 897	1980-01-01 00:00	12 897
image23.jpeg	92 105	1980-01-01 00:00	92 105
image24.png	7 285	1980-01-01 00:00	7 285
image25.png	51 106	1980-01-01 00:00	51 106
image26.png	109 439	1980-01-01 00:00	109 439
image27.jpeg	239 137	1980-01-01 00:00	239 137
image28.jpeg	112 979	1980-01-01 00:00	112 979

Slides

Name	Size	Modified	Packed Size
_rels	28 950		10 290
slide1.xml	3 729	1980-01-01 00:00	1 013
slide2.xml	10 915	1980-01-01 00:00	1 685
slide3.xml	37 332	1980-01-01 00:00	4 037
slide4.xml	14 678	1980-01-01 00:00	2 493
slide5.xml	27 172	1980-01-01 00:00	5 144
slide6.xml	26 003	1980-01-01 00:00	5 144
slide7.xml	13 685	1980-01-01 00:00	2 255
slide8.xml	13 846	1980-01-01 00:00	2 426
slide9.xml	35 984	1980-01-01 00:00	3 062
slide10.xml	23 138	1980-01-01 00:00	2 571
slide11.xml	9 936	1980-01-01 00:00	2 051
slide12.xml	10 088	1980-01-01 00:00	1 861
slide13.xml	21 625	1980-01-01 00:00	2 790
slide14.xml	6 913	1980-01-01 00:00	1 830
slide15.xml	28 209	1980-01-01 00:00	2 551
slide16.xml	12 439	1980-01-01 00:00	2 085
slide17.xml	11 514	1980-01-01 00:00	2 379
slide18.xml	7 660	1980-01-01 00:00	1 681
slide19.xml	9 354	1980-01-01 00:00	2 042
slide20.xml	24 647	1980-01-01 00:00	2 972
slide21.xml	19 928	1980-01-01 00:00	2 559
slide22.xml	24 544	1980-01-01 00:00	2 528
slide23.xml	11 278	1980-01-01 00:00	1 875
slide24.xml	13 022	1980-01-01 00:00	2 589
slide25.xml	8 225	1980-01-01 00:00	2 220
slide26.xml	27 107	1980-01-01 00:00	5 521
slide27.xml	8 820	1980-01-01 00:00	2 100

Layouts

Name	Size	Modified	Packed Size
_rels	3 557		2 095
slideLayout1.xml	5 414	1980-01-01 00:00	1 641
slideLayout2.xml	3 921	1980-01-01 00:00	1 101
slideLayout3.xml	5 442	1980-01-01 00:00	1 337
slideLayout4.xml	4 975	1980-01-01 00:00	1 199
slideLayout5.xml	7 937	1980-01-01 00:00	1 548
slideLayout6.xml	3 064	1980-01-01 00:00	978
slideLayout7.xml	2 550	1980-01-01 00:00	903
slideLayout8.xml	5 952	1980-01-01 00:00	1 452
slideLayout9.xml	5 899	1980-01-01 00:00	1 406
slideLayout10.xml	3 976	1980-01-01 00:00	1 130
slideLayout11.xml	4 200	1980-01-01 00:00	1 183

User eXperience: oggetti suddivisi per categorie

DIPARTIMENTO DI INGEGNERIA INFORMATICA AUTOMATICA E GESTIONALE ANTONIO RUBERTI

SAPIENZA UNIVERSITÀ DI ROMA

Operating Systems: File Systems

File: Implementazione 15/30

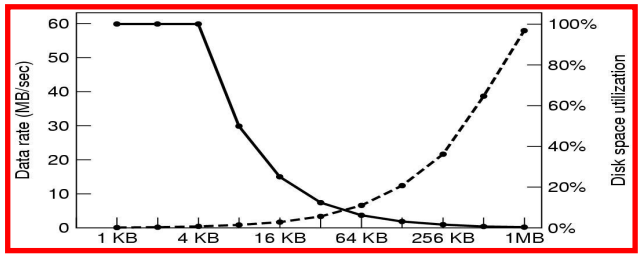
File System Management: Il «lavoro sporco»

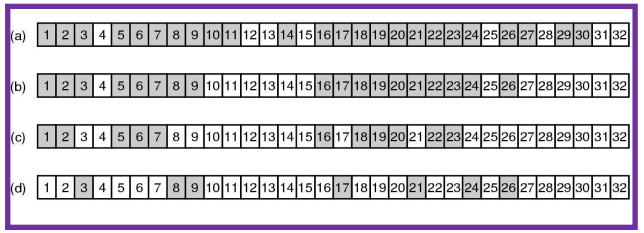
1. **Disk space management**: **Block Size**, **Free Blocks**, **Disk Quotas**
   Suddivisione e gestione della occupazione.

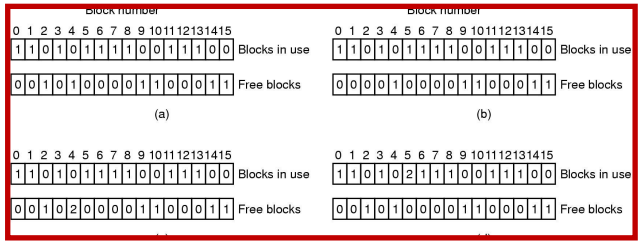
2. **File System Backups**: **Dump**, **Restore**
   Copie, generalmente, effettuate su nastro.

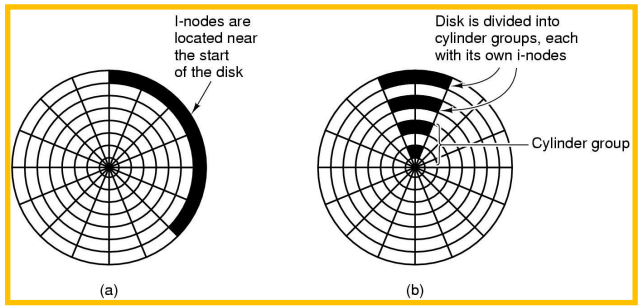
3. **File System Consistency**: **Block in Use**, **Free Blocks**
   Inconsistenza scaturita da crach di sistema occorsi prima della scrittura su disco.

4. **File System Performance**: **Caching**, **Replacement**, **Reduce Arm Motion**, **Defragging**
   Non peggiorare con l'utilizzo le già scadenti performance dei dischi.









DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI

SAPIENZA
UNIVERSITÀ DI ROMA

Operating Systems: File Systems

File: Implementazione 16/30

Disk space management

Block Size

Si utilizzano blocchi di dimensioni fisse che non devono essere adiacenti.
Se memorizzato come byte consecutivi e il file cresce dovrà essere spostato

dimensione ottimale (buona) del blocco: si necessita di informazioni sulla distribuzione delle dimensioni del file.

Length	VU 1984	VU 2005	Web
1	1.79	1.38	6.67
2	1.88	1.53	7.67
4	2.01	1.65	8.33
8	2.31	1.80	11.30
16	3.32	2.15	11.46
32	5.13	3.15	12.33
64	8.71	4.98	26.10
128	14.73	8.03	28.49
256	23.09	13.29	32.10
512	34.44	20.62	39.94
1 KB	48.05	30.91	47.82
2 KB	60.87	46.09	59.44
4 KB	75.31	59.13	70.64
8 KB	84.97	69.96	79.69

Length	VU 1984	VU 2005	Web
16 KB	92.53	78.92	86.79
32 KB	97.21	85.87	91.65
64 KB	99.18	90.84	94.80
128 KB	99.84	93.73	96.93
256 KB	99.96	96.12	98.48
512 KB	100.00	97.73	98.99
1 MB	100.00	98.87	99.62
2 MB	100.00	99.44	99.80
4 MB	100.00	99.71	99.87
8 MB	100.00	99.86	99.94
16 MB	100.00	99.94	99.97
32 MB	100.00	99.97	99.99
64 MB	100.00	99.99	99.99
128 MB	100.00	99.99	100.00

Percentuale di file più piccoli di una determinata dimensione (in byte).

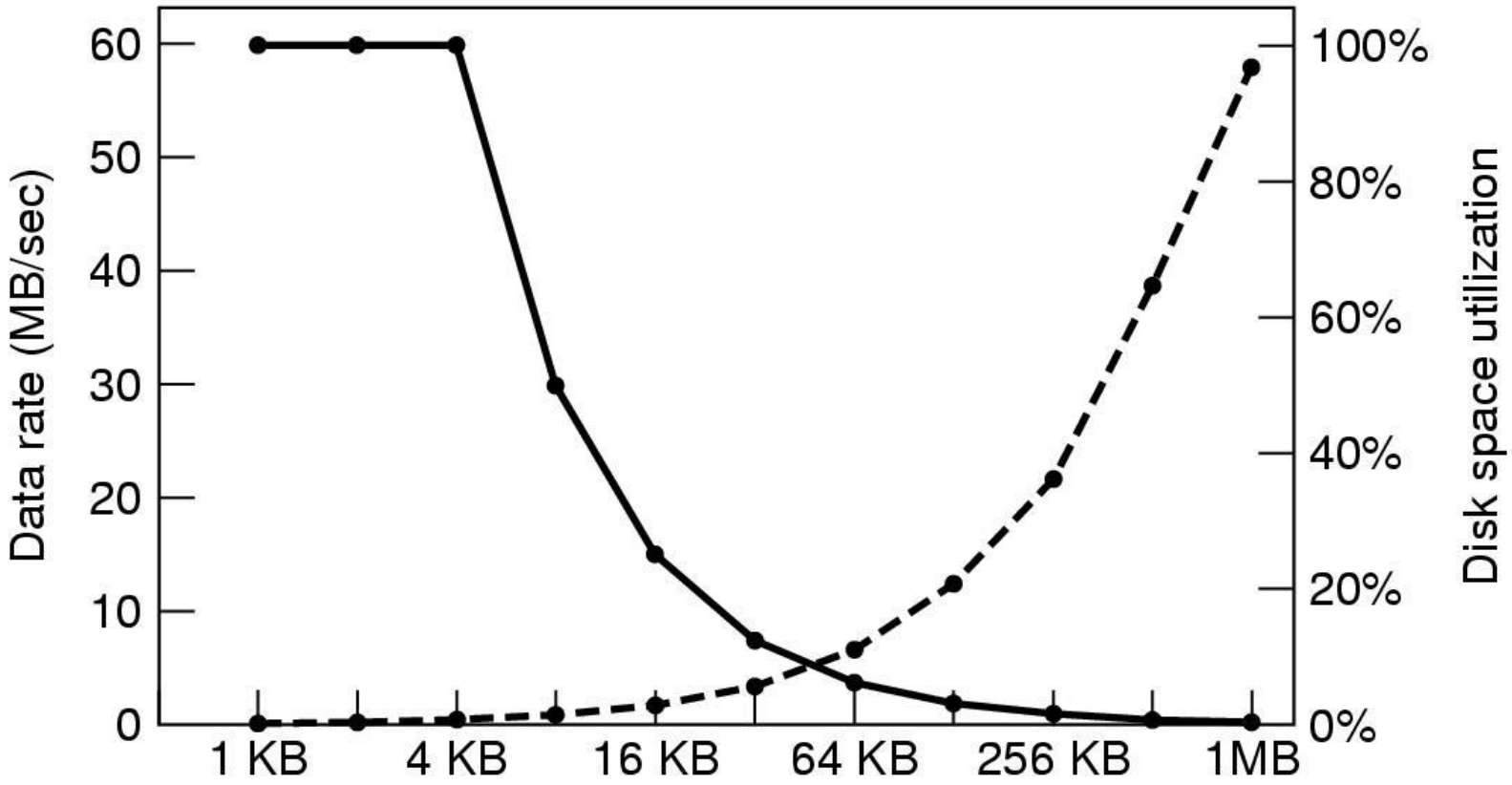
Dipartimento di Ingegneria Informatica
Automatica e Gestionale Antonio Ruberti

Sapienza
Università di Roma

Operating Systems: File Systems

File: Implementazione 17/30

Disk space management



La curva continua (scala di sinistra) fornisce la velocità dati di un disco.
La curva tratteggiata (scala di destra) fornisce l'efficienza dello spazio su disco.
Tutti i file sono 4 KB

Block Size

Maggiore è la dimensione del blocco:

- migliore è l'utilizzo dello spazio
- Peggiore è la velocità media di trasferimento

➔ compromesso:
spazio vs velocità dati
Non c'è una risposta sempre valida: es. per dischi molto grandi (come stanno diventando ora) si potrebbe anche usare dimensione del blocco grandi (64 KB).

Dipartimento di Ingegneria informatica automatica e gestionale Antonio Ruberti
Sapienza Università di Roma

Operating Systems: File Systems

File: Implementazione 18/30

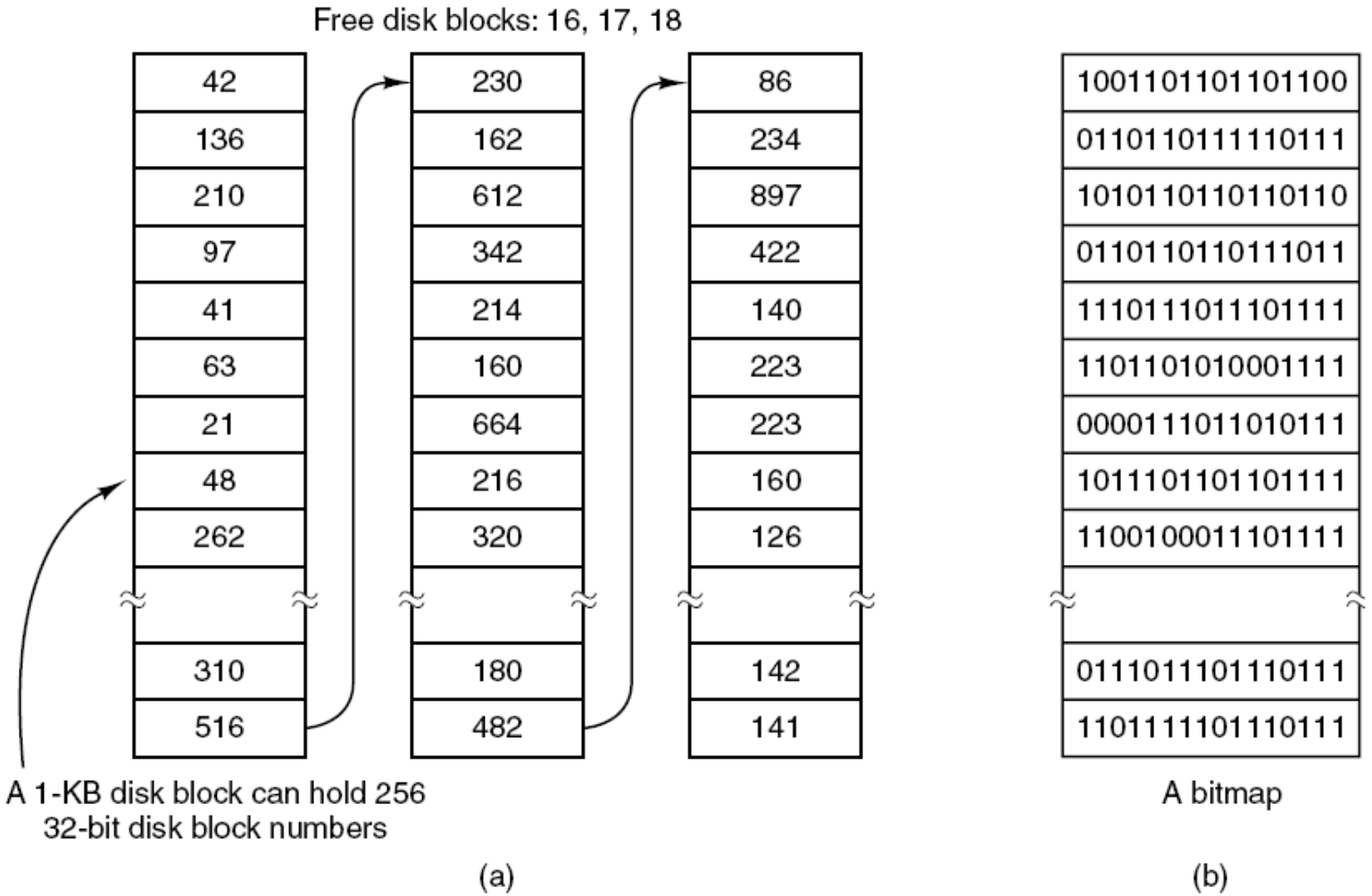
Disk space management

Free Blocks

Le liste hanno bisogno di ~1.9 milioni di blocchi
Bit-map richiede ~ 60.000 blocchi

Se i blocchi liberi sono contigui, è possibile tenere traccia dell'inizio, del blocco e della lunghezza della contiguità.

È necessario solo un blocco di puntatori alla volta nella memoria principale. Si riempie => vai a prenderne un altro



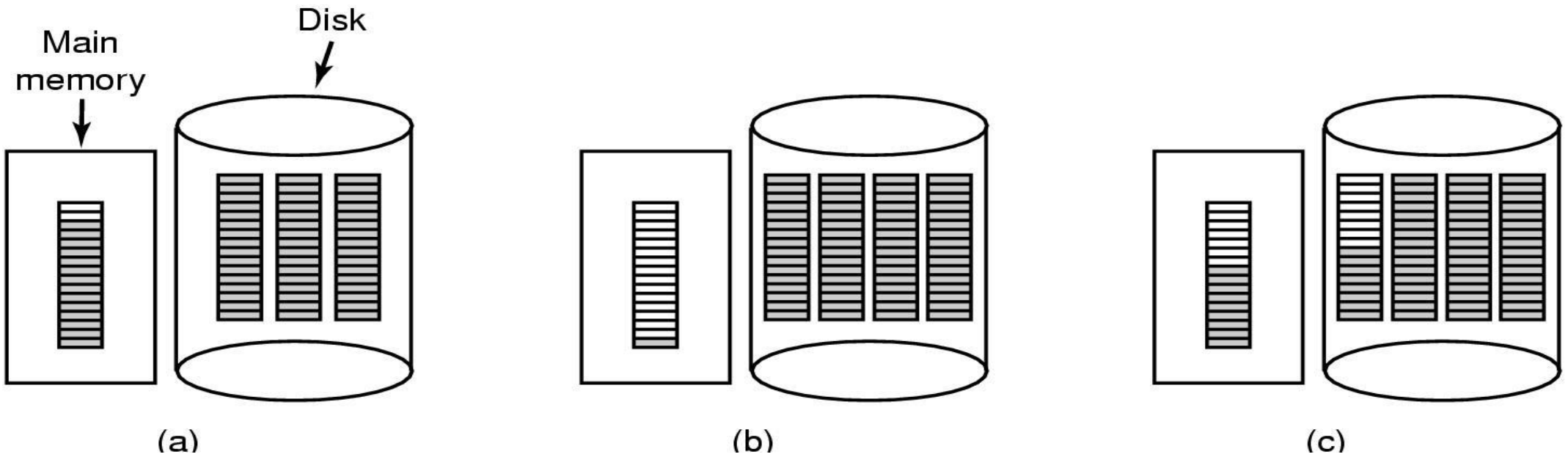
(a) Memorizzazione della lista libera su una lista collegata.
(b) Una bitmap.

Operating Systems: File Systems

File: Implementazione 19/30

Disk space management

Free Blocks



(a) Un blocco quasi pieno di puntatori a blocchi di disco liberi in memoria e tre blocchi di puntatori su disco.
(b) Risultato della liberazione di un file a tre blocchi.
(c) Una strategia alternativa per gestire i tre blocchi liberi.
Le voci ombreggiate rappresentano puntatori a blocchi di disco liberi.

DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI

SAPIENZA
UNIVERSITÀ DI ROMA

Operating Systems: File Systems

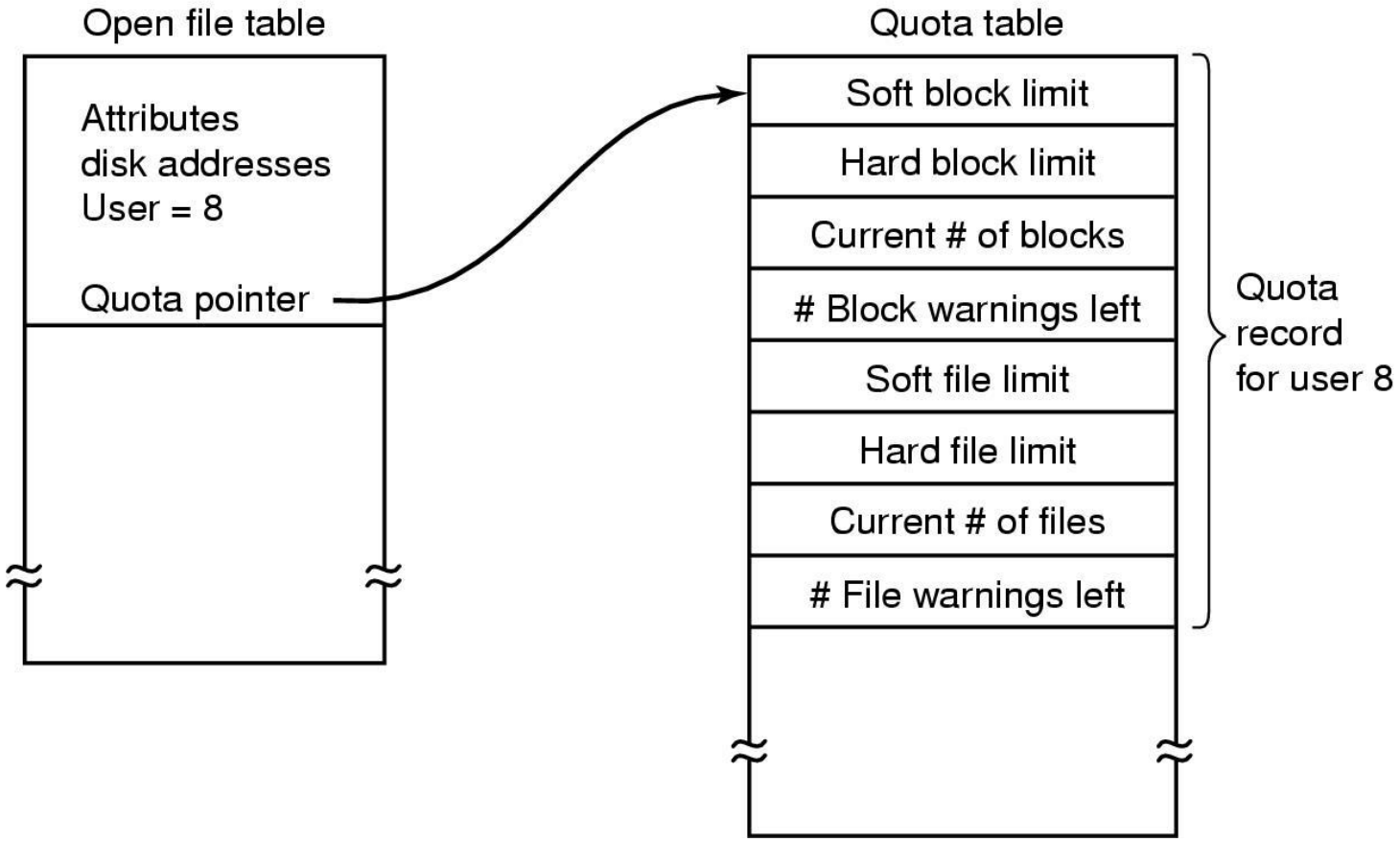
File: Implementazione 20/30

Disk space management

Disk Quotas

La voce nella tabella dei file aperti punta alla tabella delle quote.

Una voce per ogni file aperto impone i limiti (soft, hard) alla quota del disco degli utenti



Si tiene traccia delle quote per utente in una apposita tabella.

Operating Systems: File Systems

File: Implementazione 21/30

File System Backups

Dump

Copie, generalmente, effettuate su nastro:

- Recover from disaster
- Recover from stupidity

Tipologie di BackUp:

- **Totale**: completo
- **Incrementale**: giornaliero dei soli file modificati

Metodologie di BackUp:

- **Physical Dump**

➔ es. `dd /dev/sda1 /dev/tape`

- **Logical Dump** (anche incrementale)
  - Inizia dalla directory e scarica in modo ricorsivo tutti i file/directory sottostanti che sono cambiati da un dato momento
  - Possibile ripristinare il percorso su un computer diverso
  - Possibile ripristinare un singolo file

Tecniche di BackUp:

- Compressione
- A caldo/a freddo
- Snapshot

Forensic Backup

1. Raccoita del dump della memoria non elaborata e l'elenco dei processi in esecuzione.
2. Riavvio automatico della macchina nel supporto di avvio.
3. Creazione del backup che include sia lo spazio occupato che quello non allocato.
4. Autenticazione dei dischi di cui è stato eseguito il backup.

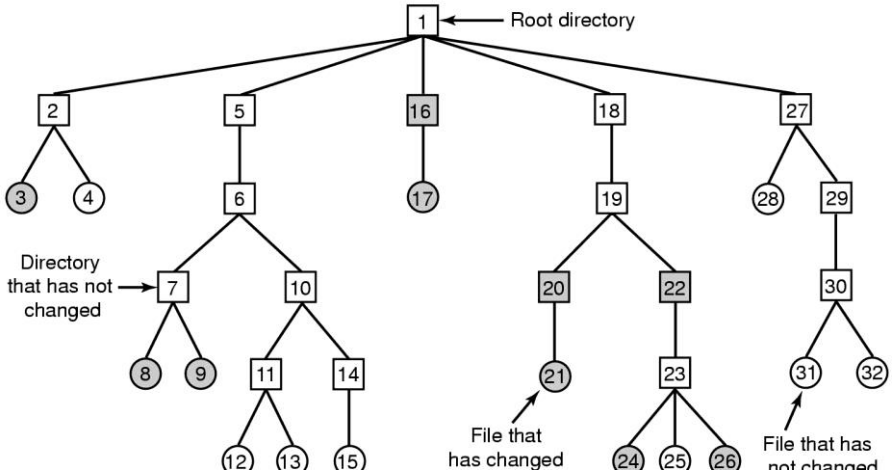
DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI

SAPIENZA
UNIVERSITÀ DI ROMA

Operating Systems: File Systems

File: Implementazione 22/30

File System Backups



Logical Dump

Utilizza bitmap indicizzata da i-node:

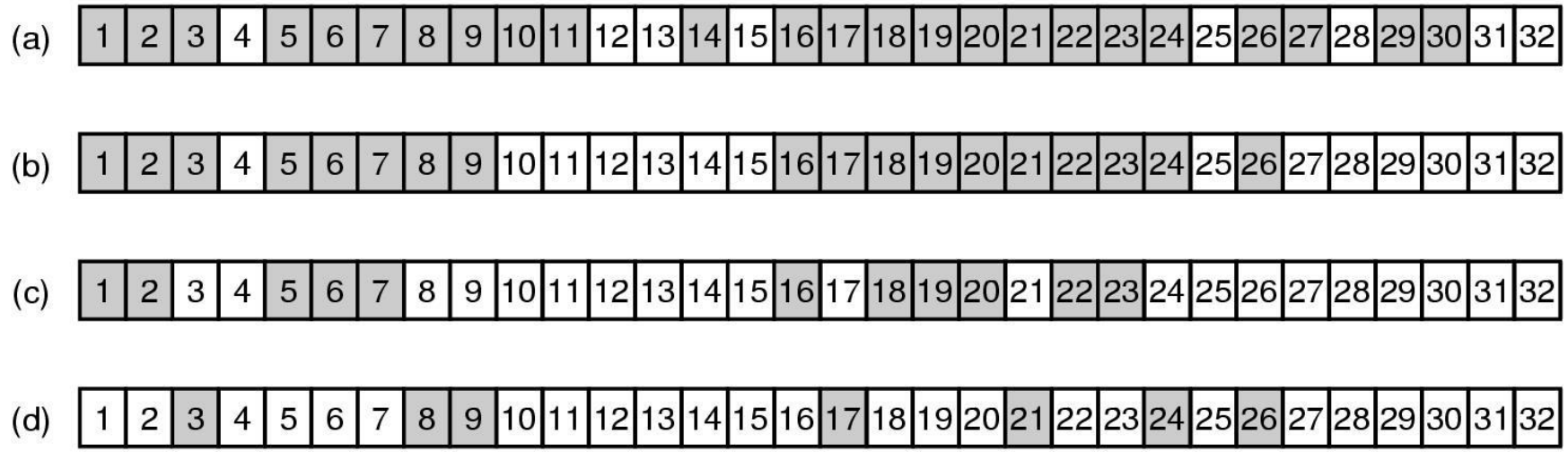
(a) La fase 1 inizia dalla radice e contrassegna i bit per i file modificati e tutte le directory

(b) La fase 2 percorre l'albero, deseleziona le directory senza file modificati al loro interno/sotto

(c) Fase 3-passa attraverso i-node e scarica le directory contrassegnate

(d) File di dump della fase 4

Un file system da scaricare. I quadrati sono directory, i cerchi sono file. Gli elementi ombreggiati sono stati modificati dall'ultimo dump. Ogni directory e file è etichettato dal suo numero i-node.



Bitmap utilizzate dall'algoritmo di dumping logico.

Operating Systems: File Systems

File: Implementazione 23/30

File System Backups

Restore

1. Inizia con fs vuoto su disco
2. Ripristina il dump completo più recente. Prima le directory, poi i file.
3. Quindi ripristina i dump incrementali (sono in ordine sul nastro).

Punti di attenzione

- L'elenco dei blocchi liberi deve essere ricostruito. ➔ è il complemento dei blocchi usati
- I collegamenti ai file devono essere ripristinati con attenzione
- I file possono avere dei buchi. Non voglio ripristinare i file con molti zeri al loro interno
- Non è possibile effettuare il dump delle pipe (sono in aree di memoria)
- La densità del nastro non è aumenta: spesso si usano i ischi per eseguire i backup

DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI

SAPIENZA
UNIVERSITÀ DI ROMA

Operating Systems: File Systems

File: Implementazione 24/30

File System Consistency

Block in Use/Free Blocks

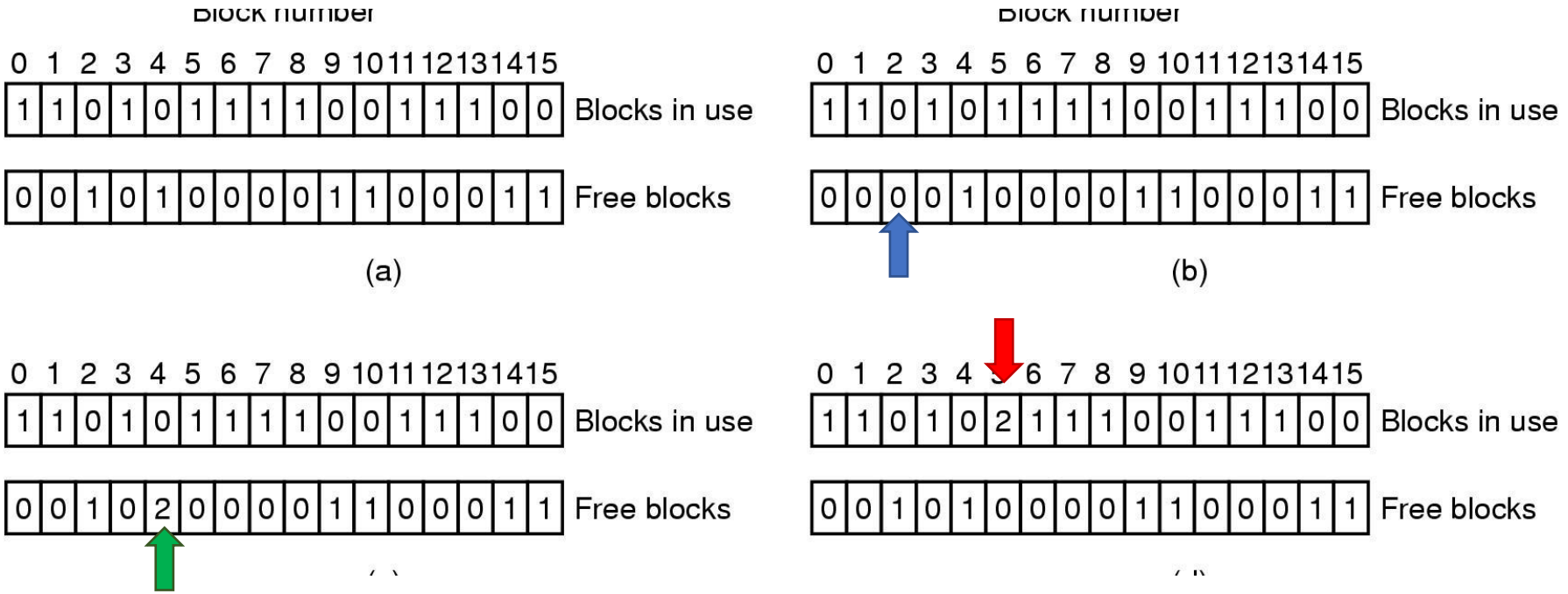
Inconsistenza scaturita da **crash di sistema** occorsi prima della scrittura su disco.

È necessario un programma di utilità per verificare la coerenza in blocchi e file (fsck in Unix, scandisk in Windows)

Utilizza due tabelle

1. Quante volte è presente un blocco in un file
2. Quante volte il blocco è presente nell'elenco dei blocchi liberi

Il dispositivo quindi legge tutti gli i-node, incrementa i contatori



Soluzione agli stati inconsistenti del file system.

(a) Coerente.

(b) Blocco mancante. ➔ blocco inserito nella lista dei blocchi liberi

(c) Blocco duplicato nella lista dei blocchi liberi ➔ ricostruzione della lista.

(d) Blocco dati duplicato ➔ notifica all'utente

Dipartimento di Ingegneria Informatica Automatica e Gestionale Antonio Ruberti

Sapienza Università di Roma

Operating Systems: File Systems

File: Implementazione 25/30

File System Consistency

File Consistency

- Analisi dei file anziché dei blocchi
- Tabella dei contatori, uno per file: incremento del contatore ogni volta che il file viene visualizzato in una directory
- Confronto dei i conteggi effettuati con i conteggi dei collegamenti presenti negli i-node (devono essere uguali per essere coerenti)

DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI

SAPIENZA
UNIVERSITÀ DI ROMA

Operating Systems: File Systems

File: Implementazione 26/30

File System Performance

Caching

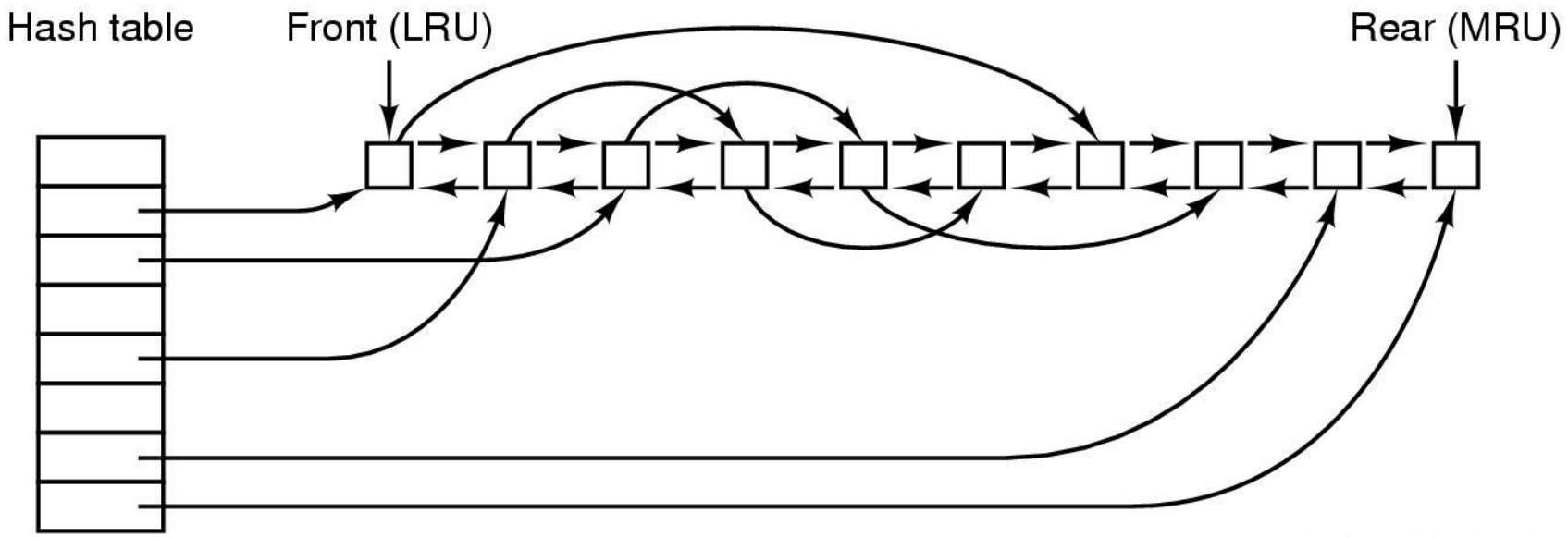
Tempo di lettura dalla memoria: 32 nsec

Tempo di lettura da disco: ricerca 5-10 msec + trasferimento 100 MB/sec

➔ Blocchi di cache in memoria

Hash table (dispositivo, indirizzo disco) da gestire

È necessario un algoritmo per sostituire i blocchi della cache: utilizzare algoritmi di paging, ad esempio LRU



Operating Systems: File Systems

File: Implementazione 27/30

File System Performance

Replacement

Problemi con LRU:

- alcuni blocchi sono usati raramente, ma devono essere in memoria

➔ blocchi indiretti
➔ blocchi di directory
➔ blocchi di dati completi
➔ blocchi di dati parziali

- i-node: deve essere riscritto su disco se modificato

➔ categorizzare gli i-node,

- L'arresto anomalo potrebbe lasciare il sistema in uno stato incoerente

- Modifica LRU: probabile che il blocco venga utilizzato di nuovo

➔ Inserire nella parte posteriore (MRU: Most Recently Used)

- Modifica LRU: essenziale per la coerenza del file system

➔ Se il blocco è necessario e viene modificato, va scritto su disco al più presto

Block Read-Ahead

(solo per file sequenziali)
come letto il blocco k nella cache, si legge k+1 se non è già presente.

Per mettere i blocchi modificati su disco al più presto:

UNIX: La sincronizzazione forza tutti i blocchi modificati su disco. Emesso ogni 30 secondi dal programma di aggiornamento

Windows: Blocco di modifica, scrittura immediata su disco (scrittura tramite cache)

Operating Systems: File Systems

File: Implementazione 28/30

File System Performance

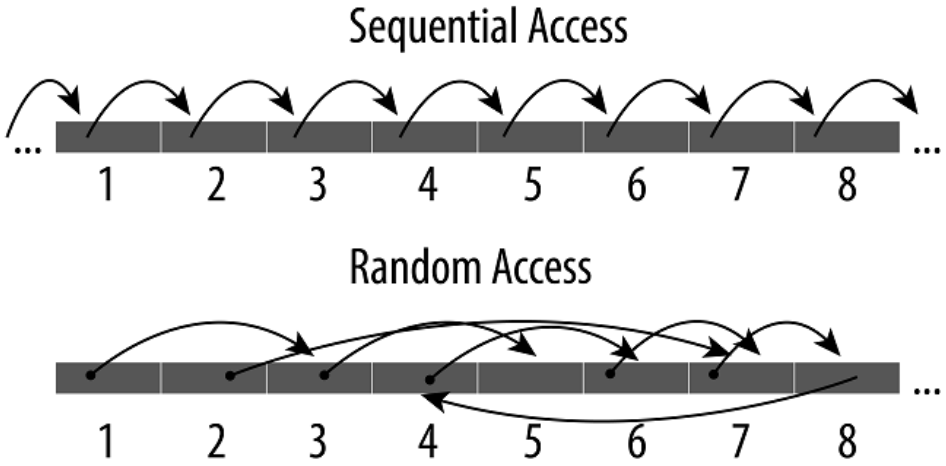
Reduce Arm Motion

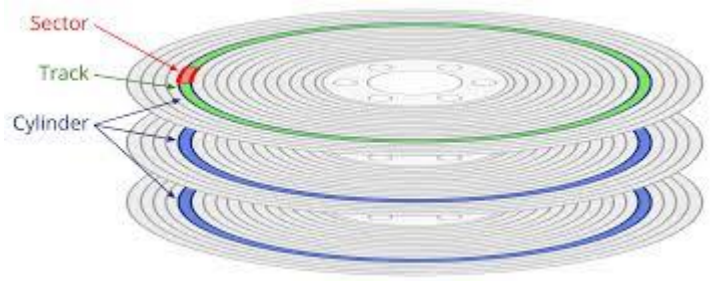
1. Provare a mettere i blocchi a cui è possibile accedere in **sequenza** uno vicino all'altro.

   Facile da fare con una bitmap in memoria: è necessario posizionare blocchi consecutivamente con un elenco libero.

2. Assegnare spazio di archiviazione dall'elenco libero in blocchi di 2 KB quando i blocchi della cache sono 1 KB

3. Provare a mettere blocchi consecutivi nello **stesso cilindro**





Dipartimento di Ingegneria informatica automatica e gestionale Antonio Ruberti

Sapienza Università di Roma

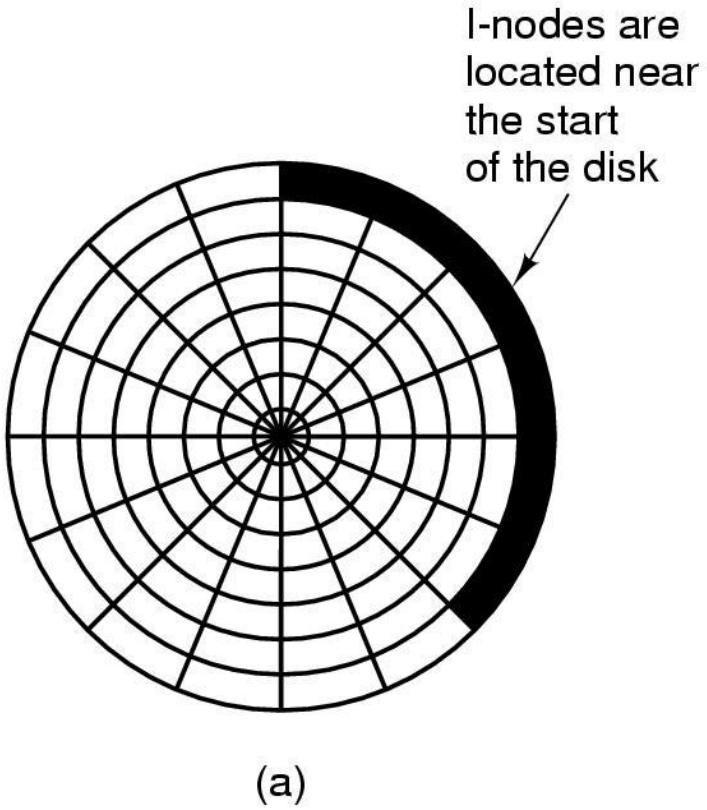
Operating Systems: File Systems

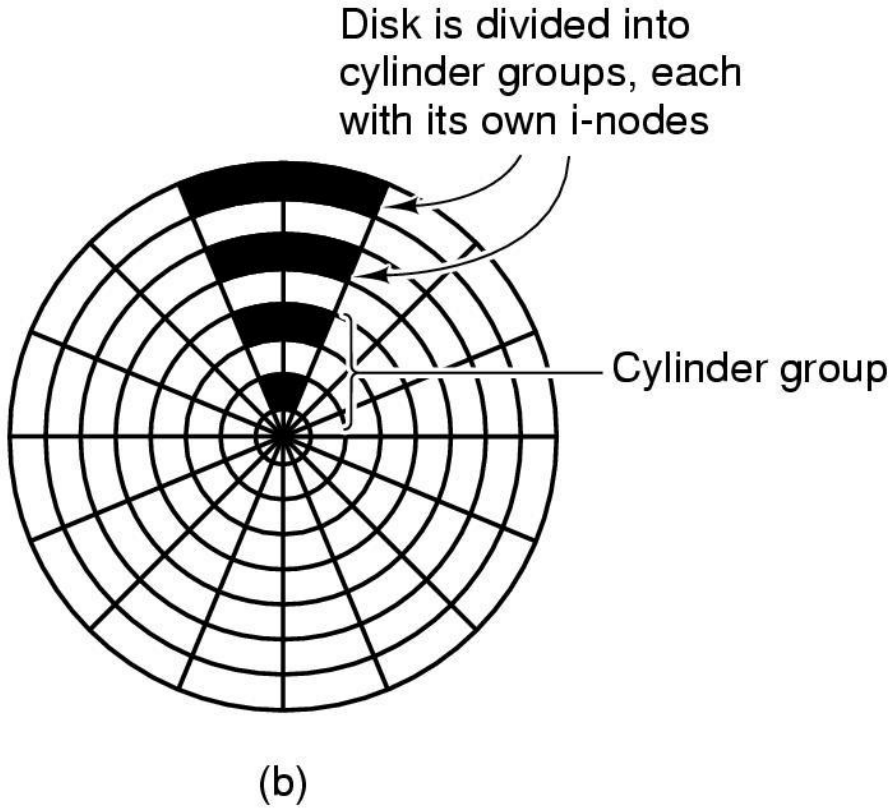
File: Implementazione 29/30

File System Performance

Reduce Arm Motion

i-nodes: posizionati per ridurre il tempo di ricerca





Dipartimento di Ingegneria informatica
automatica e gestionale Antonio Ruberti

Sapienza
Università di Roma

Operating Systems: File Systems

File: Implementazione 30/30

File System Performance

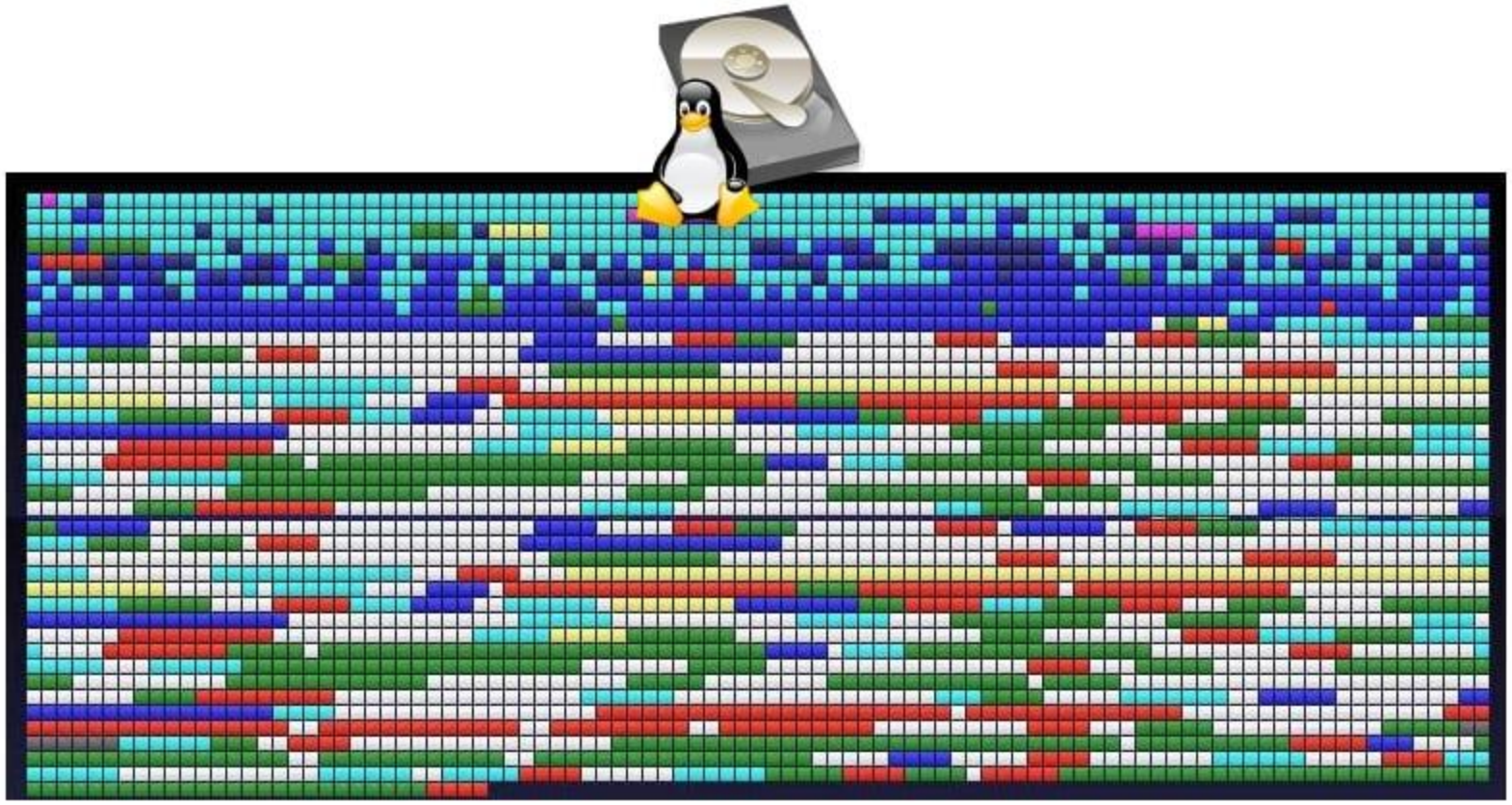
Defragging

All'inizio, i file vengono posizionati in modo contiguo sul disco.

Nel tempo compaiono dei buchi.

Programma di deframmentazione di Windows per mettere insieme blocchi disparati di un file.

Linux non ha tanti buchi



DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI

SAPIENZA
UNIVERSITÀ DI ROMA