

4 – File Systems

Sommario

Gerarchia dei dati

File tipi, struttura, attributi, accesso

File System:

- Directory
- Metadati
- Montaggio

Organizzazione dei file

Implementazione e allocazione dei file

- Contigua
- Non-contigua con liste collegate / tabellare / indicizzata

Organizzazione delle directories

Gestione e ottimizzazione dello spazio libero

Controllo di accesso ai file

- Matrice di controllo degli accessi
- Controllo degli accessi con classi di utente

Tecniche di accesso ai file systems

- Protezione dell'integrità dei dati
- Backup e Recovery
- Integrità dei dati e file di log strutturati

Casi di studio

S. Balsamo – Università Ca' Foscari Venezia – SO.4..0

Obbietivi

- Necessità di un file system
- file, directories e relative operazioni
- Organizzare e gestire i dati e lo spazio libero in un dispositivo dei memoria secondario
- Controllo degli accessi ai dati in un file system
- Trattamento dei guasti: meccanismi di backup, recovery and integrità di un file system

S. Balsamo – Università Ca' Foscari Venezia – SO.4..1

Introduzione

- **Files**
 - Raccolte di dati denominati e trattati come una unità
 - Risiedono in unità di memoria secondaria (e non solo)
- I Sistemi Operativi possono creare **interfacce** che facilitano la navigazione dei file degli utenti
 - I file system possono **proteggere** tali dati da corruzione o perdita totale causata da disastri
 - I sistemi che gestiscono grandi quantità di dati condivisi possono beneficiare di **database** in alternativa ai file

S. Balsamo – Università Ca' Foscari Venezia – SO.4..2

Gerarchia dei dati

- Le informazioni sono memorizzate nei computer secondo una gerarchia di dati
- Il **livello più basso** della gerarchia dei dati è composta da **bit**
- schemi di bit (*bit patterns*) rappresentano tutti i dati di interesse nei sistemi

S. Balsamo – Università Ca' Foscari Venezia – SO.4..3

Gerarchia dei dati

- Il secondo livello successivo nella gerarchia dei dati è formato da modelli a **lunghezza fissa di bit**, come **byte**, **caratteri** e **parole**
 - Byte**: solitamente 8 bit
 - Word**: il numero di bit di un processore può elaborare in una sola volta
 - I **caratteri** mappano i byte (o gruppi di byte) in **simboli** come lettere, numeri, segni di punteggiatura e nuove linee
 - Tre set di caratteri più utilizzati: ASCII, EBCDIC e Unicode
 - Field (campo)**: un gruppo di caratteri
 - Record**: un gruppo di campi
 - File**: un **gruppo di record correlati**

S. Balsamo – Università Ca' Foscari Venezia – SO.4.4

Gerarchia dei dati

- Il più alto livello della gerarchia dei dati è un **file system** o un **database**
- File system**: insieme di file
- Database**: insieme di dati
- Un **volume** è un'unità di memorizzazione dei dati che può contenere anche più file
 - Volume fisico**: singolo dispositivo di memoria
 - Volume logico**: anche più dispositivi di memoria

S. Balsamo – Università Ca' Foscari Venezia – SO.4.5

Files

- File: un **insieme di dati denominati** (associata ad un **nome**) che possono essere manipolati come una **unità** per operazioni quali:
 - Open prepara
 - Close
 - Create
 - Destroy
 - Copy
 - Rename
 - List

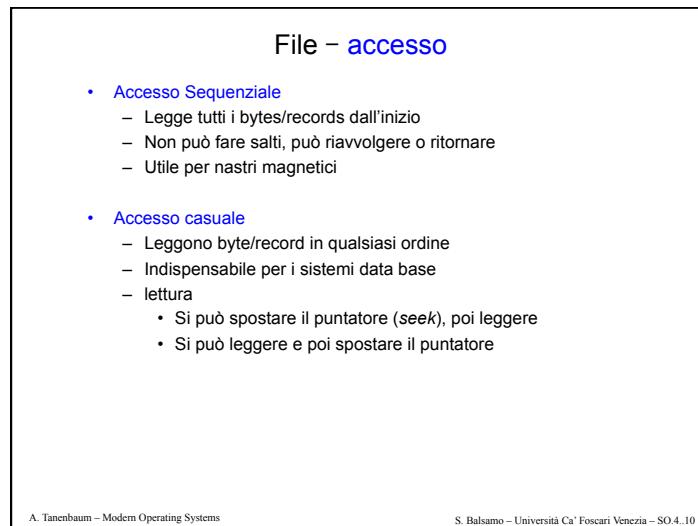
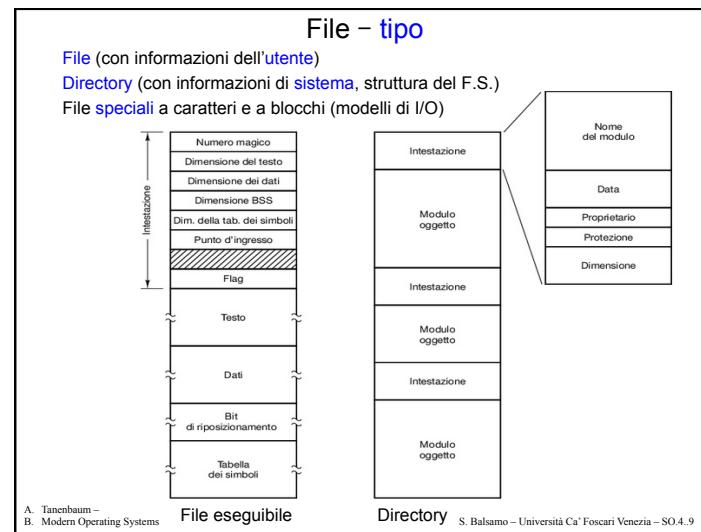
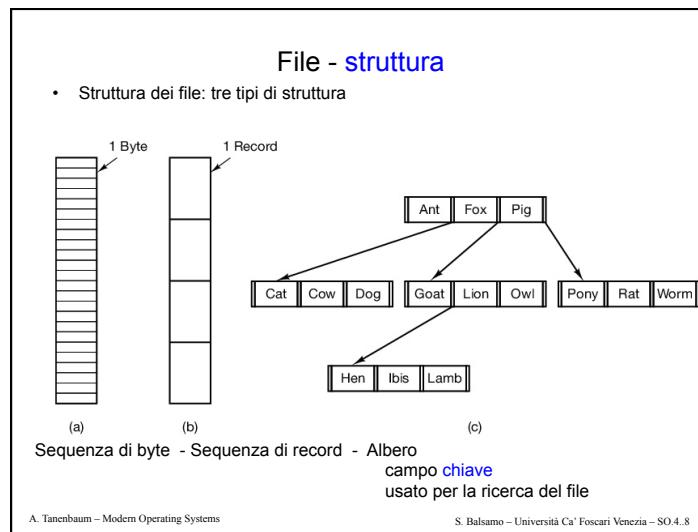
S. Balsamo – Università Ca' Foscari Venezia – SO.4.6

File - nomi

- Nomi dei file
- Spesso nomi composti da due parti separate da . *nome_file.estensione* anche più estensioni in successione gerarchica es. *nome_file.html.zip*

Estensione	Significato
file.bak	File di backup
file.c	Programma sorgente in linguaggio C
file.gif	Immagine in Compuserve Graphical Interchange Format
file.hlp	File di aiuto
file.html	Documento HTML (world wide web hypertext markup language)
file.jpg	Immagine codificata con lo standard JPEG
file.mp3	Musica codificata in formato audio MPEG layer 3
file.mpg	Filmato codificato in formato audio MPEG standard
file.o	File oggetto (output da compilatore, non ancora linkato)
file.pdf	Documento in formato Adobe PDF (portable document format)
file.ps	File PostScript
file.tex	Input per il programma di formattazione TEX
file.txt	File di testo generico
file.zip	Archivio compresso

A. Tanenbaum – Modern Operating Systems S. Balsamo – Università Ca' Foscari Venezia – SO.4.7



File - attributi

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 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 has last changed
Current size	Number of bytes in the file
Maximum size	Number of bytes the file may grow to

Esempio di attributi di file

A. Tanenbaum – Modern Operating Systems S. Balsamo – Università Ca' Foscari Venezia – SO.4.11

Files

- I singoli elementi di dati all'interno di un file possono essere manipolati da **operazioni** come

– Create – Delete – Open – Close – Read – Write	– Append – Seek – Get attributes – Set attributes – Rename
--	--
- Caratteristiche dei file includono
 - Locazione
 - Accessibilità
 - Tipo
 - Volatilità
 - Attività

S. Balsamo – Università Ca' Foscari Venezia – SO.4..12

Files

- Files possono includere uno o più record
- Record fisico** (blocco fisico)
 - Unità di informazione sul dispositivo di memoria
- Record logico** (blocco logico)
 - Insieme di dati trattati dal software come una unità logica
- Un record fisico → un record logico
 - File con record non bloccati
- Un record fisico → più record logici
 - File con record bloccati
- Dimensione del record
 - Fisse
 - Variabili

S. Balsamo – Università Ca' Foscari Venezia – SO.4..13

File Systems

- File systems
 - **Organizza** i file e **gestisce l'accesso** ai dati
 - Responsabile per
 - la **gestione dei file** (memoria, accesso, condivisione, sicurezza)
 - la gestione della **memoria ausiliaria** (allocazione dello spazio)
 - i meccanismi di **integrità** dei file (regolazione dell'accesso)
 - metodi di accesso
 - Principalmente si occupano di **gestione** dello spazio di **memoria secondaria**, e in particolare la **memoria su disco**

S. Balsamo – Università Ca' Foscari Venezia – SO.4..14

File Systems

- Caratteristiche dei File System
 - Dovrebbe essere **indipendente dal dispositivo**:
 - Gli utenti dovrebbero essere in grado di fare riferimento ai propri file di **nomi simbolici** piuttosto che dover usare i nomi dei dispositivi fisici
 - **Gestione dei guasti e sicurezza**:
 - dovrebbe anche fornire funzionalità di **backup** e **ripristino** per prevenire o la perdita accidentale o la distruzione malevola di informazioni
 - Può anche fornire funzionalità di **crittografia** e de-crittografia per rendere le informazioni utili solo per suoi destinatari

S. Balsamo – Università Ca' Foscari Venezia – SO.4..15

Directories

- Directories
 - File contenenti i **nomi** e le **posizioni** dei altri file nel file system, con lo scopo di **organizzare** e **individuare** rapidamente i file
- Gli elementi della directory memorizzano informazioni quali:

– File name	
– Location	posizione logica (<i>pathname</i>) o blocco fisico
– Dimensione	numero di byte
– Tipo	
– Accesso	tempo di ultimo accesso
– Tempo di ultima modifica e di creazione	

S. Balsamo – Università Ca' Foscari Venezia – SO.4.16

Directories

Esempio di directory di file

<i>Directory Field</i>	<i>Description</i>
Name	Character string representing the file's name.
Location	Physical block or logical location of the file in the file system (i.e., a pathname).
Size	Number of bytes consumed by the file.
Type	Description of the file's purpose (e.g., data file or directory file).
Access time	Time the file was last accessed.
Modified time	Time the file was last modified.
Creation time	Time the file was created.

© Deitel & Ass. Inc. S. Balsamo – Università Ca' Foscari Venezia – SO.4..17

Directories

- Organizzazione del file system
- File system a livello singolo o **piatto** (*flat*)
 - Organizzazione più **semplice** del file system
 - Memorizza tutti i suoi file utilizzando **una** directory
 - Non ci sono due file possono avere lo stesso nome
 - Il file system deve eseguire una **ricerca lineare** dei contenuti della directory per individuare ogni file, che può portare a scarse prestazioni

S. Balsamo – Università Ca' Foscari Venezia – SO.4.18

Directories livello singolo

Root directory

4 file
3 proprietari A, B, C

Directory principale (root)

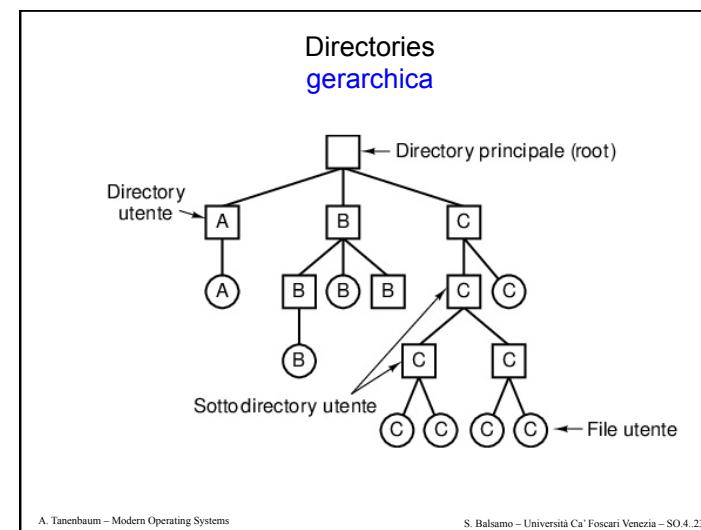
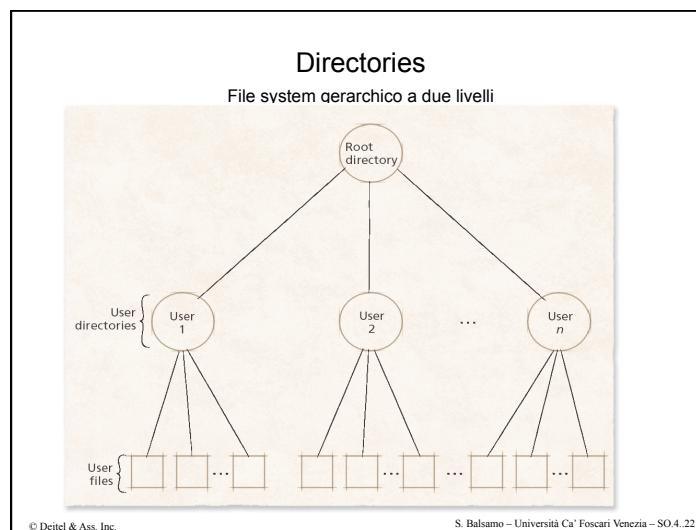
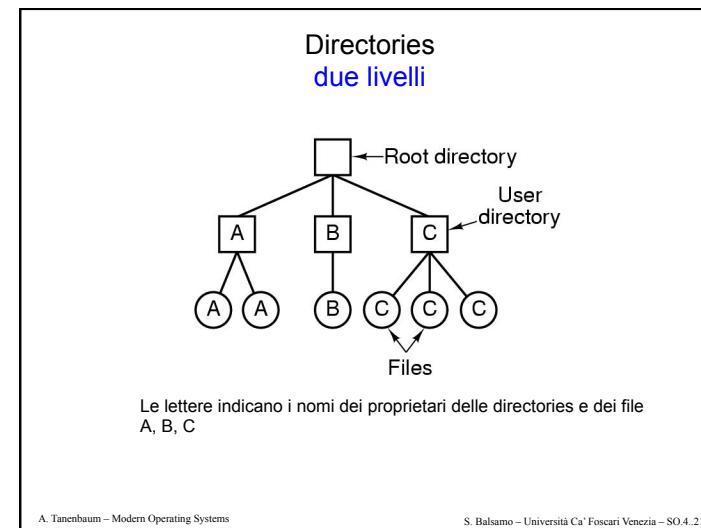
4 file
4 proprietari A, B, C, D

A. Tanenbaum – Modern Operating Systems S. Balsamo – Università Ca' Foscari Venezia – SO.4..19

Directories

- File system **gerarchico**
 - Una **radice** indica dove sul dispositivo di archiviazione inizia la directory principale (directory radice)
 - La **directory radice** punta alle varie directory, ognuna delle quali contiene una riga per ciascuno dei suoi file
 - I **nomi** dei file devono necessariamente essere **unici** solo all'interno di una determinata **directory utente**
 - Il nome di un file è di solito formato da **pathname** dalla directory radice al file
 - Esempi: directory radice delimitatore pathname
 - Windows C: \ C:\nome_dir1\nome_file
 - Unix / / /nome_dir1/nome_file

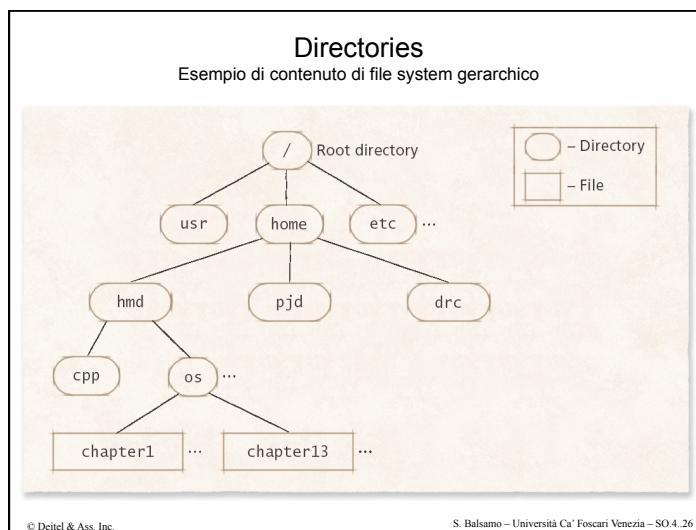
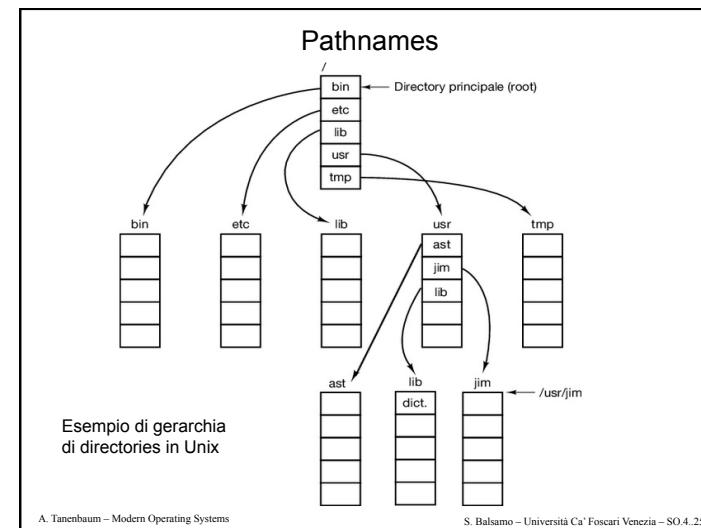
S. Balsamo – Università Ca' Foscari Venezia – SO.4.20



Directories

- Working directory
 - Semplifica la navigazione usando [pathnames](#)
 - Consente agli utenti di specificare un pathname che non inizia nella directory principale (i.e., un cammino [relativo](#))
 - Cammino [assoluto](#) (i.e. il cammino iniziando dalla radice)
 - = working directory + cammino relativo

S. Balsamo – Università Ca' Foscari Venezia – SO.4.24



Directories

- **Link:**
 - Una entry di directory che fa [riferimento](#) a un file di dati o una directory situata in una [directory diversa](#)
 - Facilita la [condivisione dei dati](#) e può rendere più facile agli utenti accedere ai file situati in tutta la struttura delle directory di un file system
 - **soft link:** entry di directory che contiene il [pathname](#) per un altro file
 - **hard link:** entry di directory che specifica la [posizione del file](#) (in genere un numero di blocco) sul [dispositivo fisico](#) di memoria

S. Balsamo – Università Ca' Foscari Venezia – SO.4.27

Directories - operazioni

- Operazioni su directories e su file, possono essere diverse nei f.s.
- Esempio in Unix*
- Create crea una nuova directory
 - Delete cancella una directory se vuota
 - Opendir apre una directory già creata
 - Closedir chiude e libera le tabelle
 - Readdir legge l'elemento successivo in una directory aperta
Indipendente dalla struttura interna
 - Rename modifica il nome
 - Link permette di far apparire un file in diverse directories
 - Unlink se non ci sono altri link il file viene cancellato

S. Balsamo – Università Ca' Foscari Venezia – SO.4.28

Directories

• Link

- Poiché un **hard link** specifica una posizione **fisica** di un file, si fa riferimento a dati **non validi** quando **cambia la posizione fisica** dei suoi file corrispondenti
- Perché i **soft link** memorizza la posizione **logica** del file nel file system, non occorre l'aggiornamento quando i file dati sono spostati
- Tuttavia, se un utente **sposta un file** in una diversa directory o rinomina il file, eventuali soft link a quel file non più validi

S. Balsamo – Università Ca' Foscari Venezia – SO.10.29

Directories

Links in un file system.

Directory	
Name	Location
foo	467
bar	843
:	:
foo_hard	467
foo_soft	./foo

Soft link

© Deitel & Ass. Inc.

S. Balsamo – Università Ca' Foscari Venezia – SO.4.30

Metadata

- Metadata
 - Informazione che protegge l'**integrità** del file system
 - Non possono essere modificate direttamente dagli utenti
- Molti file system creano un **super-blocco** per memorizzare le informazioni critiche che proteggono l'integrità del file system
Un super-blocco può contenere:
 - L'**identificatore** di sistema di file
 - La **posizione di blocchi liberi** del dispositivo di memorizzazione
 - Posizione delle directory **radice**
 - Tempo** dell'ultima modifica
- Per ridurre il rischio della perdita di dati, la maggior parte dei file system distribuiscono le **copie ridondanti** del super-blocco in tutto il dispositivo di memorizzazione

S. Balsamo – Università Ca' Foscari Venezia – SO.4.31

Metadata

- L'operazione `open` di un File restituisce un **descrittore di file**
 - Un indice intero non negativo alla tabella dei file aperti
- Da questo punto in poi, l'accesso al file viene diretto attraverso il descrittore di file
- Per consentire un rapido accesso alle informazioni specifiche di file come ad esempio i permessi, la tabella di dei file aperti spesso contiene blocchi di controllo e i file, ovvero alcuni **attributi** di file:
 - Strutture altamente dipendenti dal sistema che possono includere
 - nome** simbolico del file
 - posizione** nella memoria secondaria
 - dati di **controllo** di accesso
 - altro

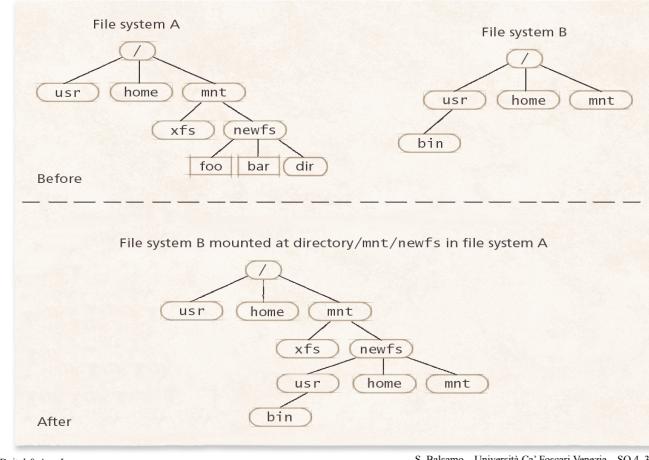
S. Balsamo – Università Ca' Foscari Venezia – SO.4.32

Mounting

- Operazione **Mount**
 - Combinare più file system in un **unico spazio dei nomi** in modo che possano essere riferiti da una singola directory radice
 - Assegna una directory, chiamato il **punto di mount**, nel file system nativo alla radice del file system montato
- Il file system gestisce le directory con **tabelle di mount**:
 - Contengono informazioni sulla **posizione** dei **punti di mount** e dei **dispositivi** ai quali puntano
- Quando il file system nativo incontra un punto di montaggio, usa la tabella di mount per determinare il dispositivo e il tipo di file system montato
- Gli utenti possono creare **soft link** ai file nel file system montati,
ma **non** possono creare **hard links** tra i file systems

S. Balsamo – Università Ca' Foscari Venezia – SO.4.33

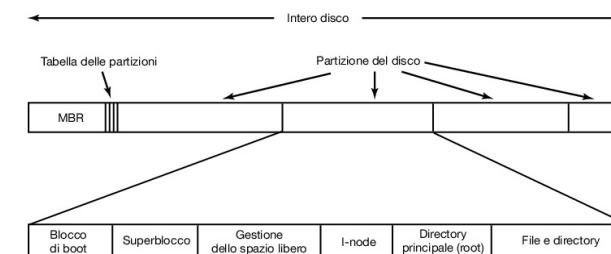
Mounting di un file system



© Deloitte & Ass. Inc.

Organizzazione dei File

- Organizzazione dei File:
come sono disposti i record di un file in memoria secondaria?
- File memorizzati su **disco**, partitionato, con file system in ogni parte
 - Settore 0 del disco: **MBR (master boot record)** per l'avvio del computer
 - Contiene anche la **tavella delle partizioni** (indirizzi), delle quali una sola è attiva
 - Il **primo** blocco di una partizione è detto **blocco di boot**
 - Se eseguito carica il S.O. contenuto nella partizione
 - Esempio* di layout di un file system su disco



A. Tanenbaum – Modern Operating Systems

S. Balsamo – Università Ca' Foscari Venezia – SO.4.35

Organizzazione dei File

- Come sono organizzati i File, memorizzati come **record** di un file in memoria secondaria
Quali blocchi del disco sono occupati dal file system?
- Schemi di organizzazione** dei File che includono:
 - Sequenziale**
 - Diretto** per dispositivi ad accesso diretto
 - Indirizzato non sequenziale** sequenze logiche tramite chiave
 - Partizionato** sottofile, memorizzati nella directory del file

S. Balsamo – Università Ca' Foscari Venezia – SO.4.36

Allocazione dei File

- Allocazione dei File**
 - Problema di **allocazione e liberazione dello spazio** nella memoria secondaria simile a quello sperimentato nell'allocazione della memoria principale con multiprogrammazione a partizioni variabili
 - Località** spaziale
 - I sistemi di allocazione **contigui** sono stati generalmente sostituiti da sistemi di allocazione **non contigui più dinamici**
 - I file tendono a crescere o ridursi nel tempo
 - Gli utenti raramente conoscono in anticipo le dimensioni dei loro file

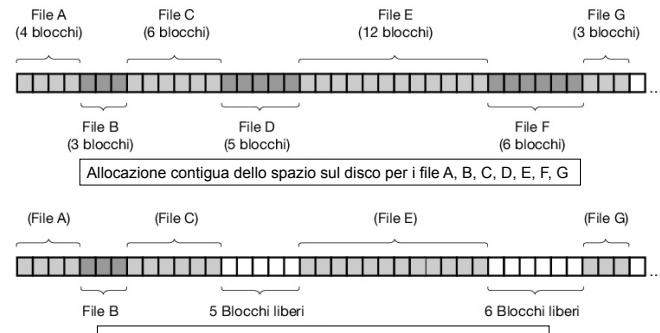
S. Balsamo – Università Ca' Foscari Venezia – SO.4.37

Allocazione dei file **contigua**

- Allocazione contigua**
 - Allocà i dati dei file ad **indirizzi contigui** sul dispositivo di memoria
 - Dimensione nota e specificata dall'utente
 - Vantaggi**
 - record logici successivi tipicamente sono **fisicamente adiacenti** l'uno all'altro
 - semplice** memorizzazione
 - Conveniente in caso di memorizzazione **statica** (e.g. CD R, DVD R)
 - Svantaggi**
 - Frammentazione esterna**
 - Possibili scarse **prestazioni** se i file crescono e si riducono nel tempo
 - Se un file cresce oltre la dimensione originariamente specificata e non sono altri blocchi liberi contigui disponibili, il file deve essere spostato in una nuova area di dimensioni adeguate, portando ad ulteriori operazioni di I / O

S. Balsamo – Università Ca' Foscari Venezia – SO.4.38

Allocazione dei file **contigua**



A. Tanenbaum – Modern Operating Systems

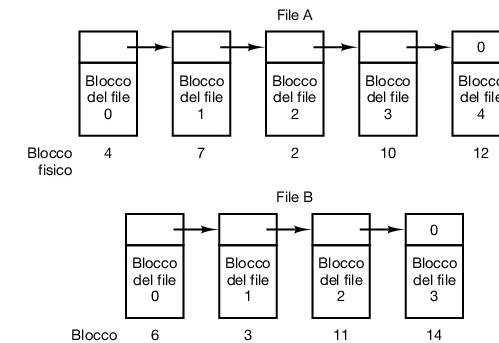
S. Balsamo – Università Ca' Foscari Venezia – SO.4.39

Allocazione non contigua con liste collegate

- Schema di allocazione di file **non contigua** basata su settore e **liste collegate**:
 - Una riga della directory punta al **primo settore** di un file
 - La porzione di dati di un settore memorizza il contenuto del file
 - La porzione puntatore punta al settore successivo del file
 - I **settori** che appartengono a un file comune formano una **lista collegata**
 - Si elimina la frammentazione esterna
 - Si può avere **frammentazione interna** all'ultimo blocco

S. Balsamo – Università Ca' Foscari Venezia – SO.4.40

Allocazione dei file con liste collegate



A. Tanenbaum – Modern Operating Systems

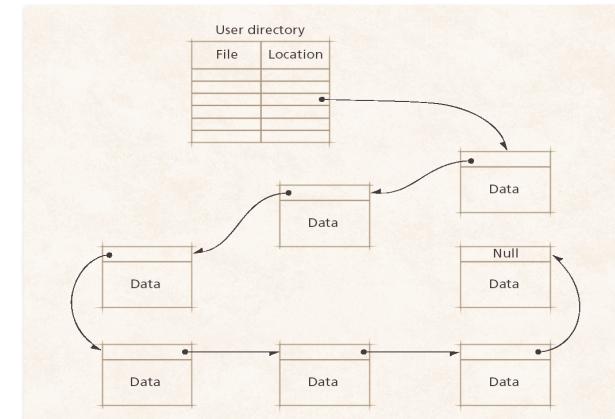
S. Balsamo – Università Ca' Foscari Venezia – SO.4.41

Allocazione non contigua con liste collegate

- Quando si esegue l'assegnazione del blocco, il sistema **alloca blocchi di settori contigui** (a volte chiamati **extent**)
- Maggior efficienza
- block chaining**
 - Le righe della directory utente punta al **primo blocco** di ogni file
 - I blocchi di file contengono:
 - Un blocco di dati
 - Un puntatore al blocco successivo

S. Balsamo – Università Ca' Foscari Venezia – SO.4.42

Allocazione non contigua con liste collegate



S. Balsamo – Università Ca' Foscari Venezia – SO.4.43

Allocazione non contigua con liste collegate

- Quando si individua un record
 - Con la catena o concatenamento dei blocchi la **ricerca** deve essere effettuata **dall'inizio**
 - Se i blocchi sono **dispersi** in tutto il dispositivo di memorizzazione (che è normale), il processo di **ricerca** può essere **lento** come la ricerca da blocco a blocco
- Inserimento e cancellazione sono fatte modificando il puntatore nel blocco precedente
- Spazio per i puntatori
- Possibili liste doppie

S. Balsamo – Università Ca' Foscari Venezia – SO.4.44

Allocazione non contigua con liste collegate

- **Dimensione dei blocchi grande**
 - Può portare ad una significativa **frammentazione interna**
 - Minor operazioni di I/O
- Dimensione dei blocchi **piccola**
 - Può causare dati la **dispersione dei dati** dei file distribuiti su più blocchi sparsi in tutto il dispositivo di memorizzazione
 - **Scarse prestazioni** perché il dispositivo di memoria esegue molte ricerche per accedere a tutti i record di un file
- Esempi: 1K, 8K

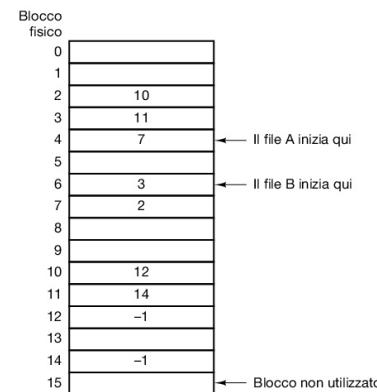
S. Balsamo – Università Ca' Foscari Venezia – SO.4.45

Allocazione dei file non contigua e tabellare

- Utilizza **tabelle** che memorizzano **puntatori ai blocchi** dei file
 - Riduce il numero di ricerche lunghe per accedere ad un record particolare
- Le righe della directory indicano il **primo blocco di un file**
- Il **numero di blocco** attuale viene utilizzato come un **indice** nella **tabella** di allocazione blocco per determinare la posizione del blocco successivo.
 - Se il blocco attuale è ultimo blocco del file, allora il valore nella tabella di allocazione dei blocchi è nullo

S. Balsamo – Università Ca' Foscari Venezia – SO.4.46

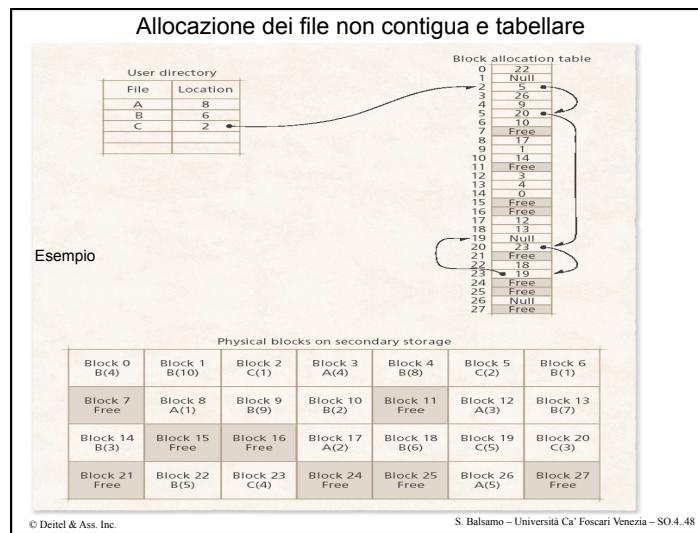
Allocazione dei file non contigua e tabellare



Esempio:
il file A è formato dai blocchi
(4, 7, 2, 10, 12)
il file B è formato dai blocchi
(6,3,11, 14)

A. Tanenbaum – Modern Operating Systems

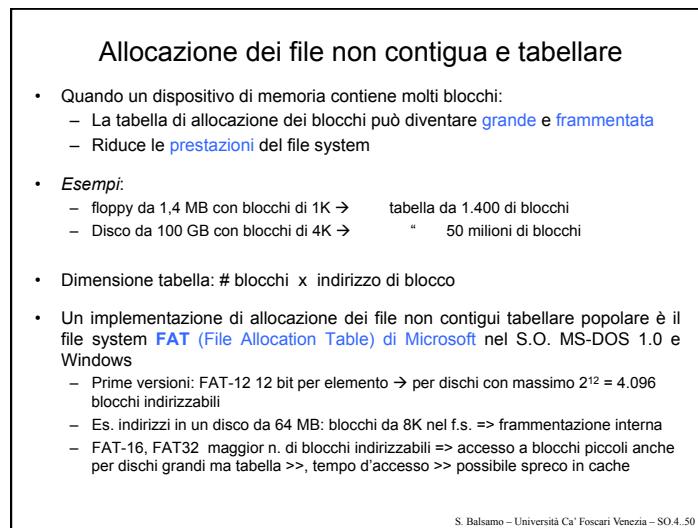
S. Balsamo – Università Ca' Foscari Venezia – SO.4.47



Allocazione dei file non contigua e tabellare

- I **puntatori** che individuano dati del file sono memorizzati in una **posizione centrale**
 - La tabella può essere memorizzata nella cache in modo che la catena di blocchi che compongono un file possa essere attraversata rapidamente
- Migliora i tempi di accesso**
- Per individuare l'ultimo record di un file, tuttavia:
 - Il file system potrebbe aver bisogno di seguire molti puntatori nella tabella di allocazione dei blocchi
 - Potrebbe richiedere un tempo significativo

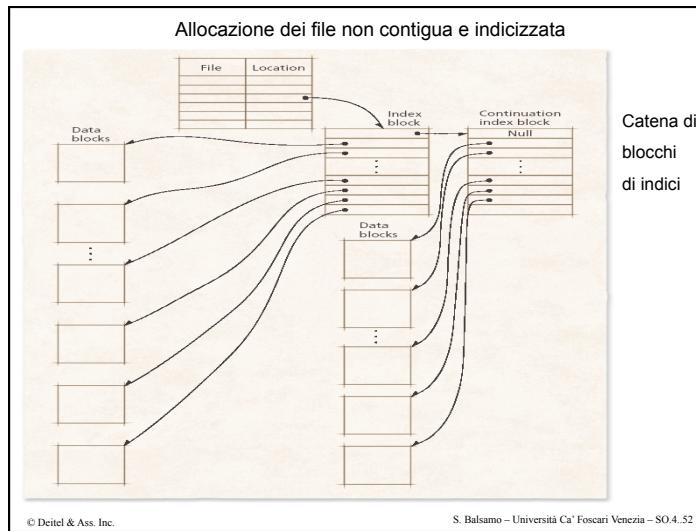
S. Balsamo – Università Ca' Foscari Venezia – SO.4.49



Allocazione dei file **non contigua e indicizzata**

- Ogni file ha un **blocco indice** o più blocchi indice
- I blocchi indice contengono un **elenco di puntatori** che puntano ai **blocchi di dati** del file
- Le righe della directory puntano ai blocchi indice del file
- Un blocco indice può riservare gli ultimi elementi per puntatori ad altri **blocchi** di indice, con una tecnica chiamata **concatenamento (chaining)**

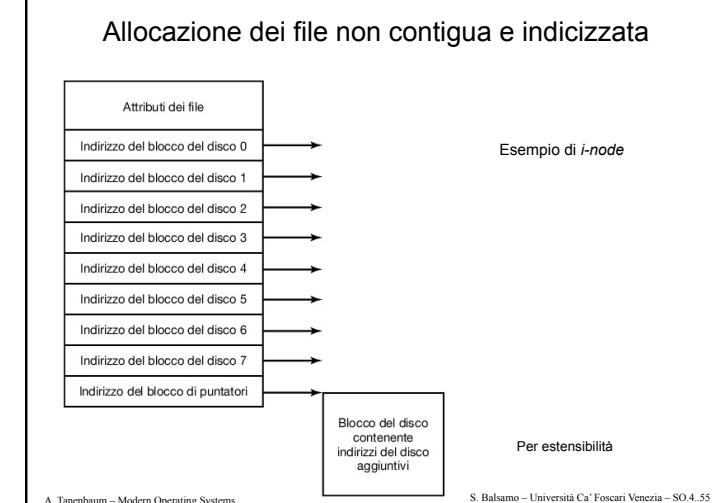
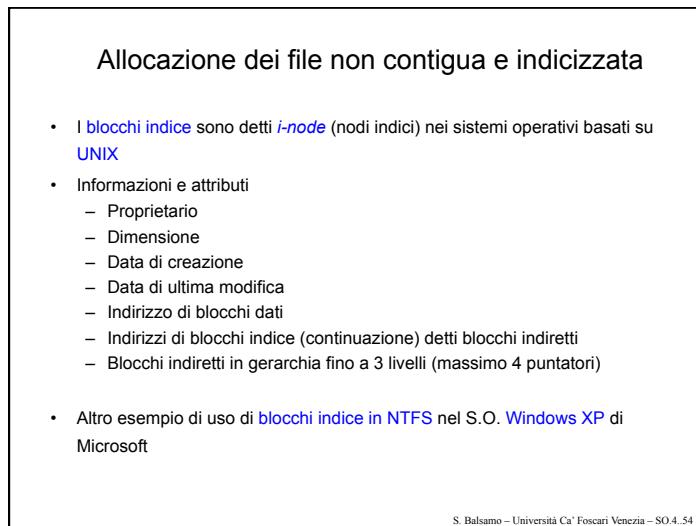
S. Balsamo – Università Ca' Foscari Venezia – SO.4.51

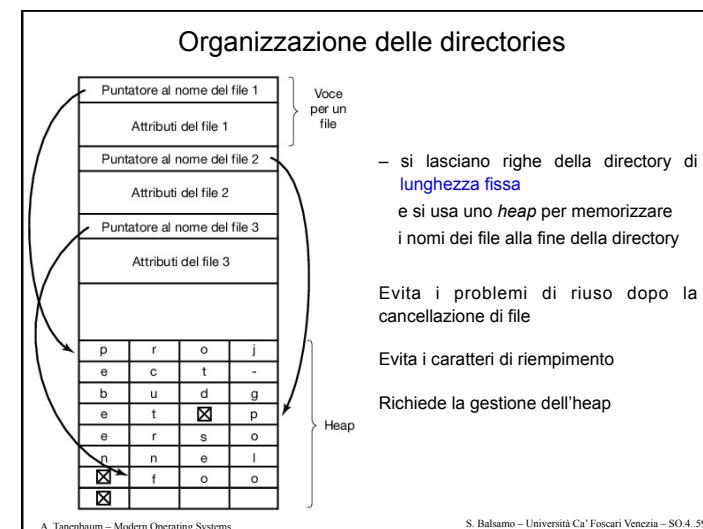
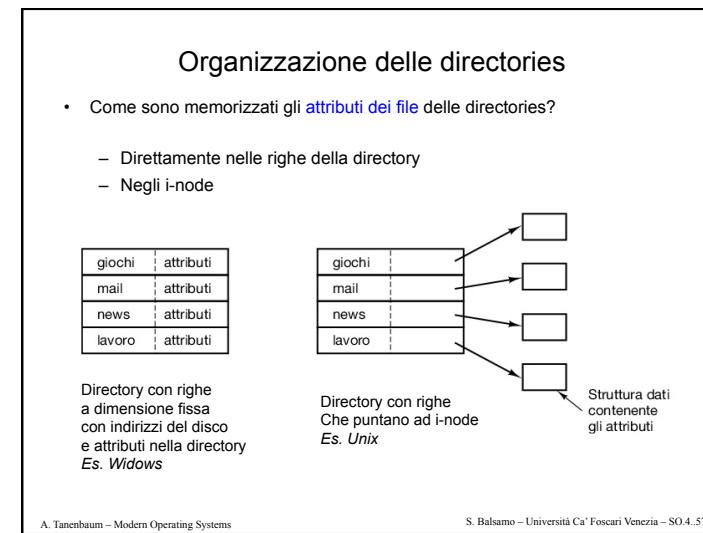
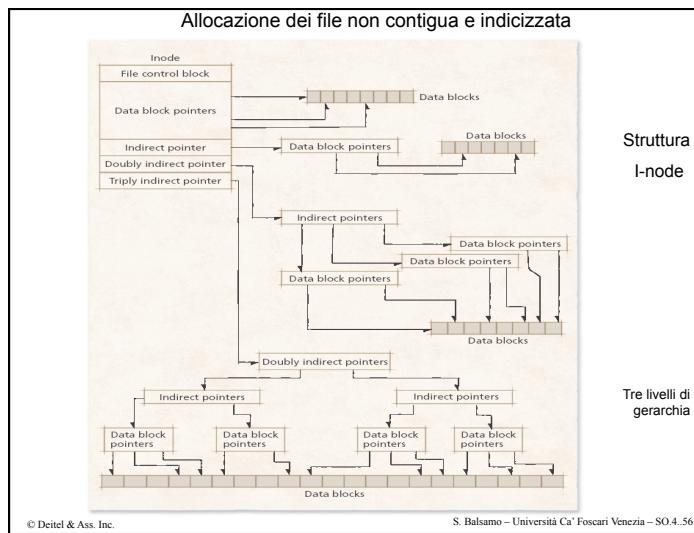


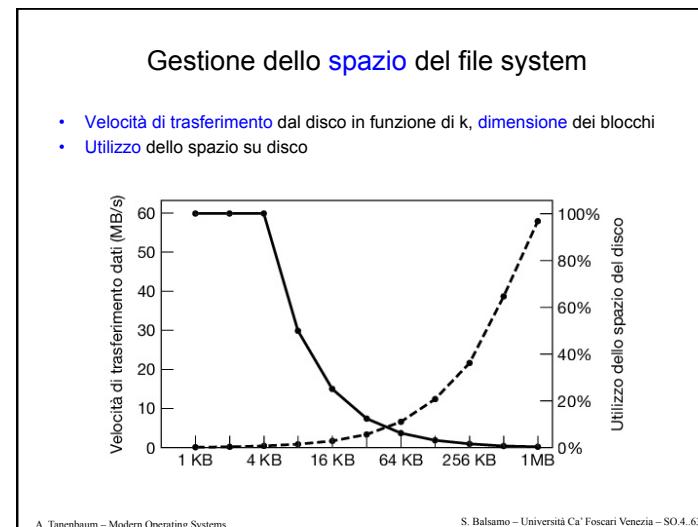
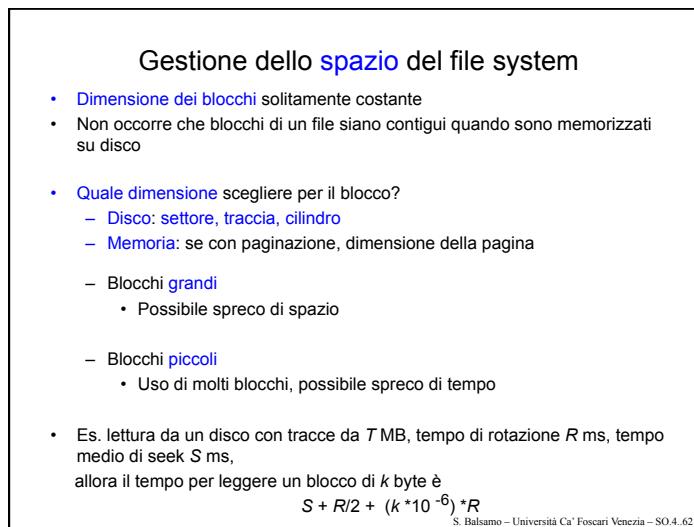
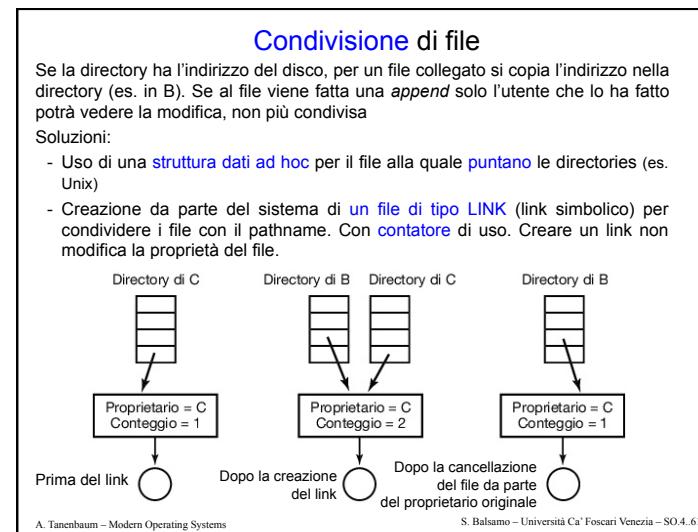
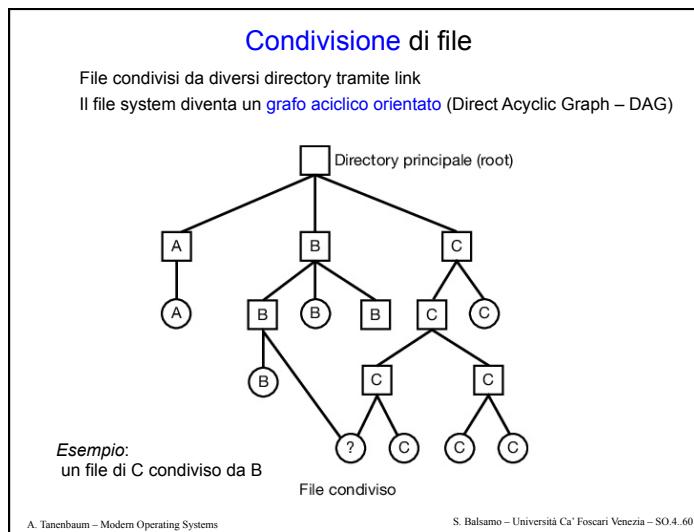
Allocazione dei file non contigua e indicizzata

- Principale vantaggio del **chaining** di blocco di indice rispetto alle semplici implementazioni con liste collegate:
 - La ricerca può avvenire nei **blocchi di indice** stessi
 - Uso di cache
 - I file system tipicamente collocano i **blocchi indice vicino ai blocchi di dati** a cui fanno riferimento, in modo che sia possibile accedere rapidamente ai blocchi di dati dopo che è stato caricato il loro blocco indice

S. Balsamo – Università Ca' Foscari Venezia – SO.4..53



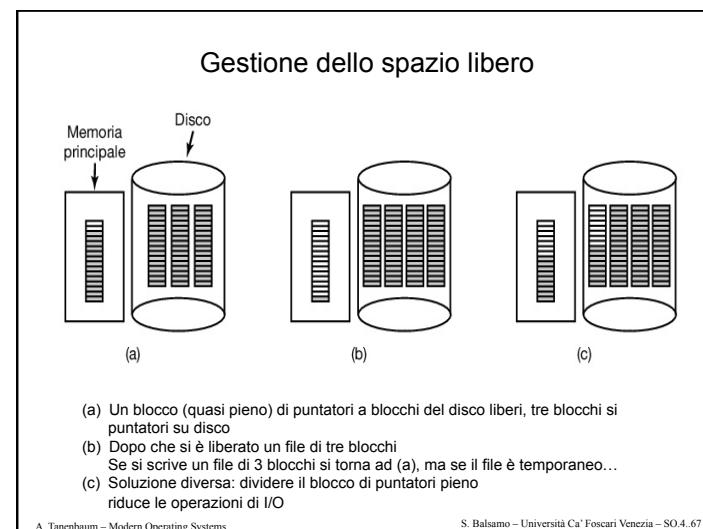
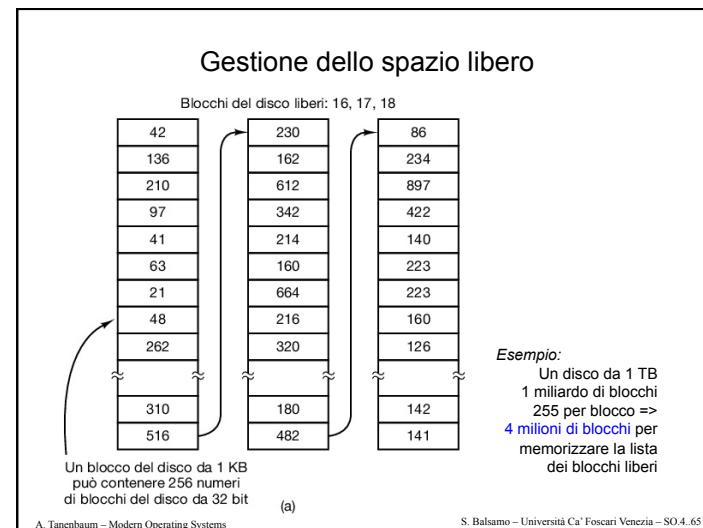




Gestione dello spazio libero

- Alcuni sistemi utilizzano una **lista libera** di gestire lo spazio libero del dispositivo di memoria
 - Lista libera: *Linked list* di blocchi contenenti le **posizioni dei blocchi liberi**
 - I blocchi vengono **assegnati a partire dall'inizio** della lista libera
 - I blocchi appena **liberati** sono **aggiunti alla fine** della lista
 - Puntatori alla testa e alla coda della lista
- Basso overhead** per eseguire le operazioni di manutenzione lista libera
- Frammentazione
- Per evitare l'allocazione sparsa si può forzare il f.s. ad allocare file possibilmente in blocchi contigui
 - Aumenta il tempo di accesso ai file

S. Balsamo – Università Ca' Foscari Venezia – SO.4.64



Gestione dello spazio libero

- Una **bitmap** (mappa di bit) contiene un **bit** per ogni **blocco** in memoria
 - i*-esimo bit corrisponde al blocco *i*-esimo sul dispositivo di memoria
 - Bit 1 se il blocco è in uso 0 se libero
- Vantaggio** della bitmap sulle liste libere
 - Velocità: il f.s. può determinare **rapidamente** se blocchi contigui sono disponibili in determinate posizioni in memoria secondaria
- Svantaggio** della bitmap
 - Overhead**: per trovare un blocco libero il f.s. potrebbe dover cercare l'intera bitmap, con sostanziale aumento dell'overhead di esecuzione

S. Balsamo – Università Ca' Foscari Venezia – SO.4.68

Gestione dello spazio libero

Per un disco di n blocchi occorrono n bit

Esempio:
Un disco da 1 TB
1 miliardo di blocchi
da 1 KB=>
130.000 blocchi per memorizzare la mappa di blocchi liberi

1001101101101100
0110110111110111
1010110110110110
0110110110111011
1110111011101111
1101101010001111
0000111011010111
1011101101101111
1100100011101111
~
0111011101110111
1101111011101111

Una bitmap

A. Tanenbaum – Modern Operating Systems

S. Balsamo – Università Ca' Foscari Venezia – SO.4.69

Gestione dello spazio libero

con una bitmap

Bitmap

Secondary storage

Blocks/bits

— Occupied

— Free

Bitmap located in first block of storage and describes first three rows of storage

S. Balsamo – Università Ca' Foscari Venezia – SO.4.70

© Deitel & Ass. Inc.

Gestione dello spazio del file system

- Utenti** diversi e **quote del disco** per memorizzare i **file** nel file system
- L'amministratore assegna e controlla le quote
- Tabella dei file aperti**
- Tabella delle quote degli utenti** con file aperti

Attributi
Indirizzi del disco
Utente = 8
Puntatore alla quota
~

Limite "soft" sui blocchi
Limite "hard" sui blocchi
Numero di blocchi attuale
Numero di avvisi sui blocchi rimasti
Limite "soft" sui file
Limite "hard" sui file
Numero di file attuale
Numero di avvisi sui file rimasti
~

Record di quota per l'utente 8

S. Balsamo – Università Ca' Foscari Venezia – SO.4.71

Backup e Recovery

- Tecniche di **Backup**
 - Conservare le **copie ridondanti** di informazioni
- Tecniche di **Recovery**
 - Consentire il ripristino dei dati del sistema dopo un errore di sistema
- Protezioni **fisiche**, quali serrature e allarmi antincendio, gruppi di continuità, copie distribuite geograficamente sono il più basso livello di protezione dei dati
- Problema dei guasti del sistema sw e ai dispositivi di memoria secondaria
- L'esecuzione dei **backup periodici** è la tecnica più comune utilizzata per garantire la continua disponibilità dei dati (**availability**)

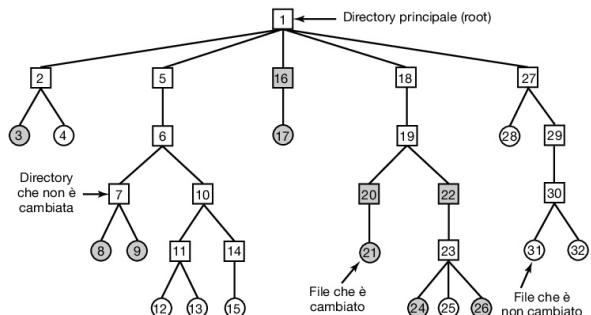
S. Balsamo – Università Ca' Foscari Venezia – SO.4..72

Backup e Recovery

- **Backup fisico**
 - Duplicare i **dati** di un dispositivo di memoria a livello di bit
 - Senza informazioni logiche – compatibilità limitata
 - Semplice tecnica di backup
- **Backup logico**
 - Memorizzare i **dati** del file system e la sua **struttura logica**
 - Controlla la struttura delle directory per determinare **quali file** devono essere sottoposti a **backup**, quindi scrive questi file su un dispositivo di backup in un formato di archiviazione, spesso compresso
 - Esempio: formato **tar** in Unix
 - I **backup incrementali** sono backup logici che memorizzano solo i dati del file system che sono stati modificati rispetto al backup precedente
 - Possibile maggior frequenza

S. Balsamo – Università Ca' Foscari Venezia – SO.4..73

Backup e Recovery



- Backup di un file system
 - quadrati=directories, circoli=files
 - grigi: file **modificati** dall'ultimo backup
 - Directory e file etichettati dal numero di i-node

A. Tanenbaum – Modern Operating Systems

S. Balsamo – Università Ca' Foscari Venezia – SO.4..74

Backup e Recovery

- (a)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----
- (b)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----
- (c)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----
- (d)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----

Bitmap per un backup logico

- (a) parte dalla directory principale e esamina tutti i nodi, directory e file indicando quelle modificate
- (b) deselectiona le directory che non hanno internamente file modificati
- (c) backup delle directory selezionate in ordine di i-node
- (d) backup dei file selezionati in ordine di i-node

Mantiene la struttura

A. Tanenbaum – Modern Operating Systems

S. Balsamo – Università Ca' Foscari Venezia – SO.4..75

Backup e Recovery

- Bitmap per un backup **logico**

Nota

- La lista dei blocchi liberi va ricostruita
- File condivisi (link)
- File con 'buchi'

A. Tanenbaum – Modern Operating Systems

S. Balsamo – Università Ca' Foscari Venezia – SO.4.76

Integrità dei dati e file system strutturato a log

- Nei sistemi che non possono tollerare la perdita di dati o tempi di inattività, sono scelte adeguate:
 - RAID
 - registrazione (*logging*) delle transazioni
- Se si verifica un errore di sistema durante un'operazione di scrittura, i dati dei file **possono** essere lasciati in uno **stato incoerente**

S. Balsamo – Università Ca' Foscari Venezia – SO.4.77

Integrità dei dati e file system strutturato a log

- File system **basati su transazioni**
 - Riducono la perdita di dati utilizzando le transazioni **atomiche**
 - Eseguono un gruppo di operazioni in **tutto o niente**
 - Se si verifica un errore che impedisce il completamento di una transazione, il sistema viene ripristinato riportandolo all'ultimo stato precedente dell'inizio dell'operazione
 - **rollback**

S. Balsamo – Università Ca' Foscari Venezia – SO.4.78

Integrità dei dati e file system strutturato a log

- **Transazioni atomiche**
 - Implementabile registrando il risultato di ogni operazione in un file di log invece di modificare dati esistenti
 - Una volta che la transazione è stata completata, è **committed** (confermata) registrando un valore sentinella nel log
- **Checkpoints**
 - Per ridurre il tempo per le operazioni di ri-esecuzioni nel log, la maggior parte dei sistemi basati su transazioni mantengono checkpoints che puntano l'ultima transazione che è stata trasferita nella memoria permanente
 - Se il sistema si blocca, è sufficiente esaminare solo le transazioni dopo il checkpoint
- La **paginazione shadow** (ombra) implementa transazioni atomiche scrivendo dati modificati ad un blocco libero al posto del blocco originale
 - Terminata la transazione i metadati sono aggiornati al nuovo blocco liberando il vecchio
 - Se la transazione viene annullata, si annulla l'aggiornamento liberando il nuovo blocco.

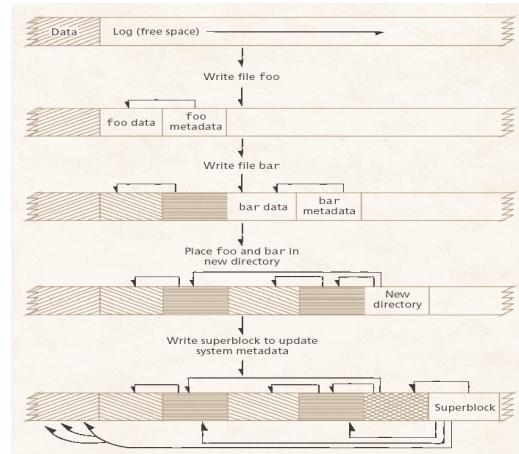
S. Balsamo – Università Ca' Foscari Venezia – SO.4.79

Integrità dei dati e file system strutturato a log

- **Log-structured file systems** (LFS)
 - Chiamato anche **journaling file system**
 - Esegue tutte le operazioni di file system come transazioni registrate per garantire che non lascino il sistema in uno stato incoerente
 - L'intero **disco** serve come un **file di log**
 - Registra le transazioni
 - I **nuovi dati** sono **scritti in sequenza** nello spazio libero del file di **log**
 - Esempi: NTFS Journaling File System di Microsoft, ext3 di Red Hat per Linux

S. Balsamo – Università Ca' Foscari Venezia – SO.4.80

Integrità dei dati e file system strutturato a log



S. Balsamo – Università Ca' Foscari Venezia – SO.4.81

Integrità dei dati e file system strutturato a log

- Le **directory** e i **metadati** modificati sono sempre scritte **alla fine del log**
- quindi un LFS potrebbe aver bisogno di leggere l'intero registro per individuare un file particolare, con conseguente **scarse prestazioni in lettura**
- Per ridurre questo problema, un LFS
 - Memorizza nella **cache** le **posizioni** dei **metadati** del file system
 - Ogni tanto scrive le **mappe degli i-node** o i super-blocchi nel log per indicare la posizione di altri metadati
 - Consente al S.O. di individuare e memorizzare rapidamente in cache i metadati quando il sistema si avvia
 - Migliori prestazioni temporali, maggiore memoria

S. Balsamo – Università Ca' Foscari Venezia – SO.4.82

Integrità dei dati e file system strutturato a log

- Alcuni file system cercano di **ridurre il costo** dei file system strutturati a log utilizzando **un log solo** per memorizzare i **metadati**
 - Assicura l'integrità del file system con un overhead relativamente basso
 - Non garantisce l'integrità del file in caso di guasto del sistema

S. Balsamo – Università Ca' Foscari Venezia – SO.4.83

Integrità dei dati e file system strutturato a log

- I dati sono scritti nel registro **sequenzialmente**
 - Ogni **scrittura LFS** richiede **una sola ricerca** mentre c'è ancora spazio su disco
- Quando il registro si riempie, è probabile che lo spazio libero del file system sia **frammentato**
- Questo può portare a **scarse prestazioni** in lettura e scrittura
- Per risolvere questo problema, un LFS può **creare spazio libero contiguo** nel log copiando i dati validi in una **regione contigua** alla fine del log

S. Balsamo – Università Ca' Foscari Venezia – SO.4.84

File Servers e Sistemi distribuiti

- Un modo per gestire riferimenti ai file non locali in una rete di calcolatori è di instradare le richieste a un **file server**
 - Un sistema dedicato alla **risoluzione di riferimenti** ai file fra sistemi
 - Controllo **centralizzato** di questi riferimenti
 - Il File server potrebbe facilmente diventare un **collo di bottiglia**, perché tutti i clienti inviano le richieste a tale server
- Un approccio migliore è quello **distribuito**, lasciando che i sistemi siano separati e comunichino direttamente tra loro
- File system distribuiti**
 - Eterogeneità, Trasparenza
 - Esempi: NFS (Network File System) di Sun Microsystems, AFS, Coda,...

S. Balsamo – Università Ca' Foscari Venezia – SO.4.85

Controllo degli accessi ai File

- I file sono spesso usati per memorizzare i **dati sensibili** come ad esempio:
 - numeri di carta di credito
 - password
 - numeri di previdenza sociale
- Pertanto, essi dovrebbero includere **meccanismi** per **controllare l'accesso** degli utenti **ai dati**
 - Matrice di controllo di accesso**
 - Controllo** di accesso per **classi** di utenti

S. Balsamo – Università Ca' Foscari Venezia – SO.4.86

Matrice dei controlli di accesso

- Matrice di accesso bidimensionale accesso
 - $a_{ij} = 1$ se all'utente i è consentito l'accesso ai file j
 - $a_{ij} = 0$ altrimenti
- In un sistema con un numero di utenti elevato e molti file, questa matrice generalmente sarebbe **grande e sparsa**
- Inappropriati per la prevalenza dei sistemi

S. Balsamo – Università Ca' Foscari Venezia – SO.4.87

		Matrice dei controlli di accesso									
User \ File		1	2	3	4	5	6	7	8	9	10
1	1	1	0	0	0	0	0	0	0	0	0
2	0	0	1	0	1	0	0	0	0	0	0
3	0	1	0	1	0	1	0	0	0	0	0
4	1	0	0	0	0	0	0	0	0	0	0
5	1	1	1	1	1	1	1	1	1	1	1
6	0	0	0	0	0	1	1	0	0	0	0
7	1	0	0	0	0	0	0	0	0	0	1
8	1	0	0	0	0	0	0	0	0	0	0
9	1	1	1	1	0	0	0	0	1	1	1
10	1	1	0	0	1	1	0	0	0	0	1

© Deitel & Ass. Inc.

S. Balsamo – Università Ca' Foscari Venezia – SO.4.88

Controllo degli accessi per classi di utente

- Una tecnica che richiede molto meno spazio è controllare l'accesso alle varie classi di utenti
- Le classi di utenti possono includere:
 - Proprietario del file *può cambiare i permessi*
 - Un particolare utente
 - Gruppo
 - Progetto
 - Pubblico *soltamente sola lettura*
- Dati per il controllo di accesso
 - Possono essere memorizzati come parte del blocco di controllo di file ([File Control Block](#))
 - Spesso richiedono poco spazio

S. Balsamo – Università Ca' Foscari Venezia – SO.4.89

Tecniche di accesso ai dati

- I moderni S.O. forniscono in genere [molti metodi di accesso](#)
- Obiettivo: migliorare le [prestazioni](#) (tempo di accesso)
- [Pipeline](#) fra elaborazione e I/O
- Metodi di [accesso a coda](#)
 - Migliori prestazioni
 - Utilizzato quando la [sequenza di accesso](#) ai record può essere [prevista](#), ad esempio accesso sequenziale e sequenziale indicizzato
 - Esegue una [memorizzazione a previsione](#) e lo [scheduling](#) delle operazioni di I/O
- Metodi di [accesso di base](#)
 - Normalmente utilizzati quando non può essere [prevedibile](#) la sequenza di accesso ai record, in particolare con l'accesso diretto

S. Balsamo – Università Ca' Foscari Venezia – SO.4.90

Tecniche di accesso ai dati

- File mappati in memoria**
 - i dati dei file mappati nello spazio di [indirizzamento virtuale](#) di un [processo](#) invece di usare la cache del file system
 - Poiché i riferimenti ai file mappati in memoria si verificano nello spazio di indirizzamento virtuale di un processo, il [gestore della memoria virtuale](#) può prendere decisioni di sostituzione di pagina basate sullo specifico modello di riferimento di ogni processo (successione di riferimenti)

S. Balsamo – Università Ca' Foscari Venezia – SO.4.91

Protezione dell'integrità dei dati

- I sistemi spesso memorizzano alcune [informazioni critiche](#), come ad esempio:
 - Inventari
 - Record finanziari
 - Informazioni personali
- Un crash di sistema, disastri naturali e programmi malevoli possono distruggere queste informazioni
- I risultati di tali eventi possono essere catastrofici
- I sistemi operativi e i sistemi di memorizzazione dei dati di funzionamento devono essere **fault tolerant**
 - Considerare la possibilità di disastri e fornire le [tecniche di recupero e ripristino \(recovery\)](#) dopo un guasto

S. Balsamo – Università Ca' Foscari Venezia – SO.4.92

Controllo degli accessi

- I moderni S.O. devono
 - [Controllare](#) attentamente contro un uso improprio e dannoso delle risorse del computer
 - [Proteggere](#) i servizi del S.O. e le [informazioni sensibili](#) da parte degli utenti e/o software che hanno accesso alle risorse
- Diritti di accesso**
 - [Proteggere](#) le risorse ed i servizi di sistema da parte degli utenti potenzialmente pericolosi
 - [Limitare](#) le azioni che possono essere eseguite sulle risorse
- Solitamente gestito da
 - Liste di controllo di accesso ([access control lists](#))
 - Liste di 'capacità' ([capability lists](#))

S. Balsamo – Università Ca' Foscari Venezia – SO.4.93

Diritti di Accesso e domini di protezione

- La chiave per la sicurezza del S.O. è [controllare l'accesso alle risorse](#) del sistema
- I diritti di accesso più comuni ([privilegio](#))
 - Lettura
 - Scrittura
 - Esecuzione
- Dominio di protezione**
 - insieme di diritti di accesso, ciascuno indicato come [`<nome oggetto, privilegi>`](#)
- Le tecniche impiegate per la gestione dei diritti di accesso:
 - [Matrici di controllo di accesso](#)
 - [Lista di controllo di accesso](#)
 - [Liste di capability](#)

S. Balsamo – Università Ca' Foscari Venezia – SO.4.94

Modelli di controllo degli accessi e politiche

- Modello di sicurezza**
 - Definisce soggetti, oggetti e privilegi di un sistema
- Esempi:
 - [classi](#) utente
 - il controllo di accesso basato sui ruoli
 - [Ruolo](#) (insieme di operazioni) e [privilegi associati](#)
- [Discretionary Access Control \(DAC\)](#) (a discrezione)
 - Il proprietario del file controlla i permessi
- [Mandatory Access Control \(MAC\)](#) (obbligatorio)
 - Predefinire un sistema di autorizzazione centrale

S. Balsamo – Università Ca' Foscari Venezia – SO.4.95

Meccanismi di controllo degli accessi

- **Politica di sicurezza**

- Solitamente specificato dall'**utente** o dall'**amministratore** di sistema
- Definisce **quali privilegi** agli oggetti vengono assegnati ai soggetti
- La maggior parte incorporano il principio del **privilegio minimo**
- Al soggetto viene concesso l'accesso solo agli oggetti di cui ha bisogno per svolgere i suoi compiti

- **Meccanismo di sicurezza**

- Il metodo con cui il sistema **implementa** la politica di sicurezza

S. Balsamo – Università Ca' Foscari Venezia – SO.4.96

Meccanismi di controllo degli accessi

- **Matrici di controllo degli accessi**

- Associa i soggetti (righe) e gli oggetti (colonne) ai **diritti di accesso appropriati** (contenuto, azioni ammesse)
- Semplice concetto sotteso al modello
- Solo le autorizzazioni **esplicite**
- Molti sistemi contengono molti soggetti e oggetti, di conseguenza la matrice può essere di grandi dimensioni e spesso sparsa
- Può diventare un mezzo **inefficiente** per il controllo dell'accesso
- Elemento critico centrale da proteggere

S. Balsamo – Università Ca' Foscari Venezia – SO.4.97

Meccanismi di controllo degli accessi

Matrice di controllo degli accessi per un piccolo gruppo di soggetti e oggetti

	File A	File B	Printer
Alice	Read*	Read*	Print*
Bob	Read*	Read*	Print
Chris	Read		Print
David		Read	
Guest			

* diritti duplicabili fra utenti

© Deitel & Ass. Inc.

S. Balsamo – Università Ca' Foscari Venezia – SO.4.98

Meccanismi di controllo degli accessi

- **Lista dei controlli degli accessi**

- Stesse informazioni della matrice, ma solo per diritti esplicitamente attribuiti
- O sulle righe (soggetti) o sulle colonne (oggetti)
- Possibili elenchi lunghi, ricerca non sempre efficiente
- Minor spazio di memoria per matrici sparse

S. Balsamo – Università Ca' Foscari Venezia – SO.4.99

Meccanismi di controllo degli accessi

Lista dei controlli di accesso derivata dalla matrice dei controlli di accesso

```
1 File A:  
2   <Alice, {read*, write*}>  
3   <Bob, {read*, write}>  
4   <Chris, {read}>  
5 File B:  
6   <Alice, {read*, write*}>  
7   <Bob, {read*, write}>  
8   <David, {read}>  
9 Printer:  
10  <Alice, {print*}>  
11  <Bob, {print}>  
12  <Chris, {print}>
```

© Deitel & Ass. Inc.

S. Balsamo – Università Ca' Foscari Venezia – SO.4..100

Meccanismi di controllo degli accessi

- **Lista di capability**
 - Deriva dal principio del **minimo privilegio**
 - Capability
 - Puntatore o token che garantisce l'accesso ad un soggetto che lo possiede
 - Con un identificatore
 - Creato per un oggetto
 - Riproducibile (copia)
 - Controllo della propagazione
 - Dominio di protezione: insieme di capabilities di un soggetto
 - Spesso sono metodi più **efficienti** e **flessibili** di gestione dei diritti di accesso

S. Balsamo – Università Ca' Foscari Venezia – SO.4..101