Laboratorio Informatica

Laurea in Biologia & Biologia Molecolare, Università di Padova

Introduzione ai sistemi operativi

Introduzione a Linux

A/A 2010/2011

Docenti:

Dott. Ivilin Stoianov

Dott.ssa Alessia Ceccato

Lab P150 (Paolotti) e LabInf (Torre Archimede)

Per iniziare

Login registra

Password registra

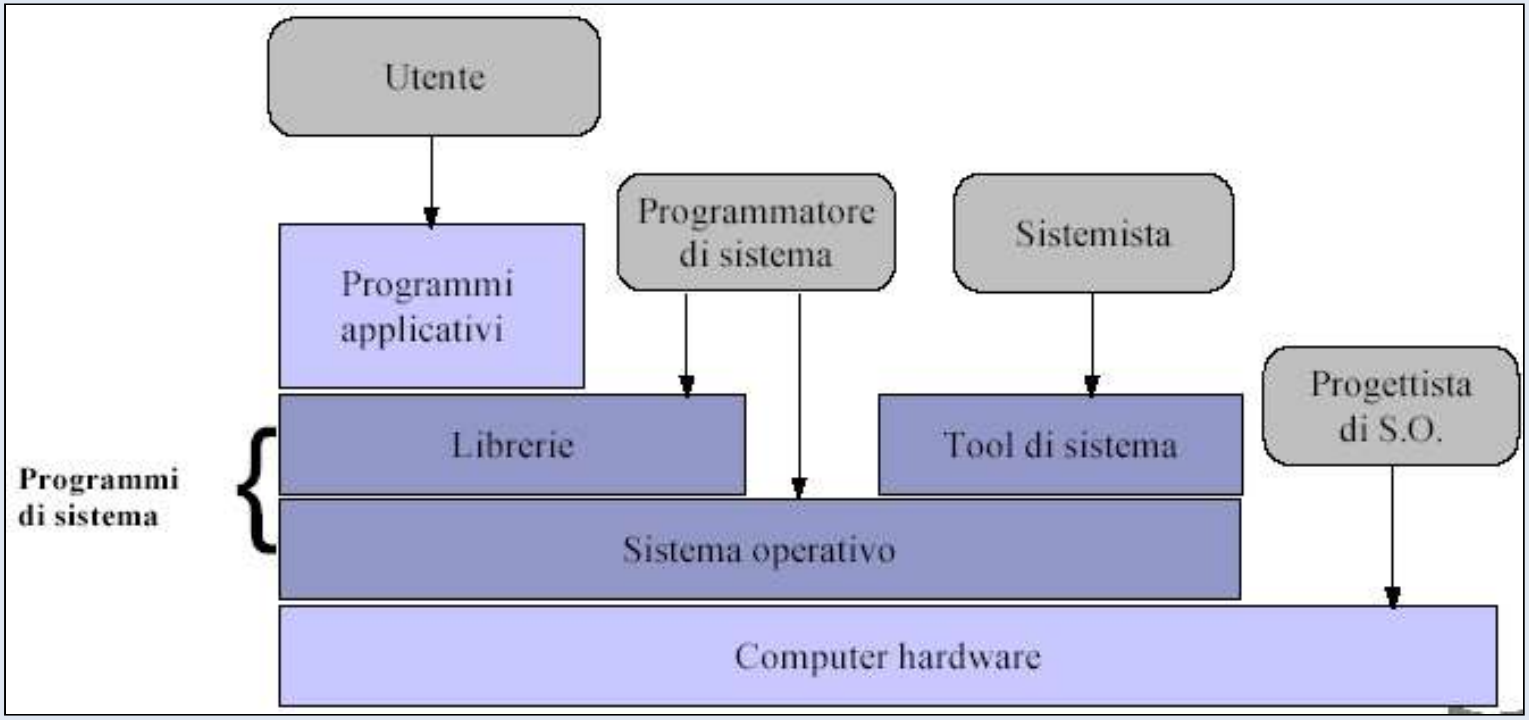
- Selezionare il proprio corso di laurea
- Immettere o la matricola o il proprio codice fiscale
- Scriversi a parte la password stando attenti a trascriverla correttamente (maiuscole/minuscole e punteggiatura varia)
- Al sito www.studenti.math.unipd.it è possibile cambiare la propria password

Cos'è un sistema operativo

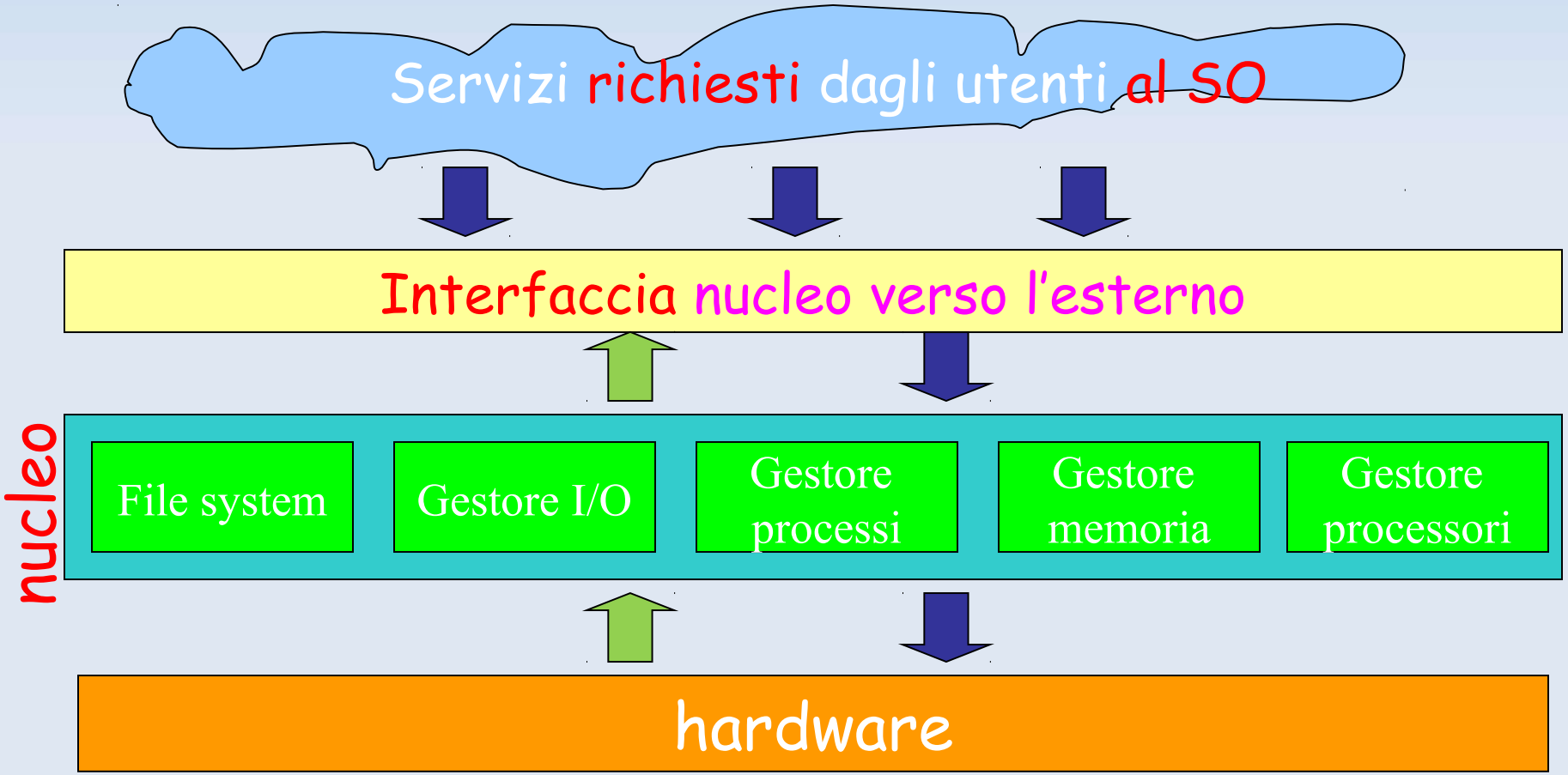
- Il software può essere diviso in due grandi classi:
  - i programmi di sistema, che gestiscono le funzionalità del sistema di calcolo
  - i programmi applicativi, che risolvono i problemi degli utenti
- L'insieme dei programmi di sistema viene comunemente identificato con il nome di Sistema Operativo (SO)
- Definizione: **Un sistema operativo è un programma che controlla l'esecuzione dei programmi applicativi ed agisce come interfaccia fra le applicazioni e l'hardware del calcolatore**

Il SO come macchina estesa

- Visione a strati delle componenti hardware/software che compongo un sistema di elaborazione



Struttura del SO

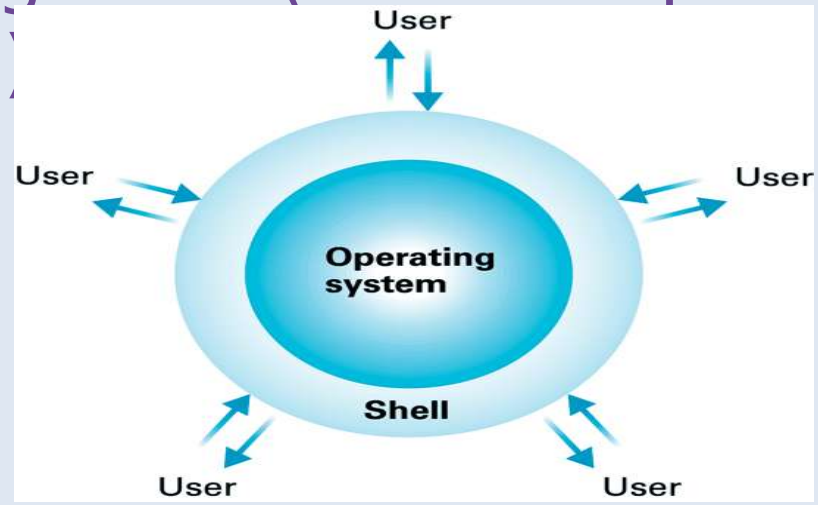


Struttura di un SO

- Elementi principali di un SO:
  - Shell (interfaccia SO e utente)
  - Kernel (insieme di programmi che realizzano le funzioni di base di un calcolatore)

Sistema operativo: shell

- Shell (guscio): **interfaccia tra SO e utenti**
- E’ il programma che permette agli utenti di comunicare con il sistema e di avviare I programmi
- Di solito grafica (GUI – Graphical User Interface)



Sistema operativo: kernel (1)

- Kernel (nucleo): **programmi per le funzioni base del calcolatore**
- Kernel **suddiviso in moduli**
- Ogni modulo ha **una funzione** diversa
- Funzioni piu' importanti:
  - gestione processi
  - gestione processori
  - gestione memoria (principale e secondaria)
  - gestione dispositivi di I/O

Sistema operativo: kernel(2)

- Funzioni piu' importanti:
  - **gestione processi:** controlla la sincronizzazione, l'interruzione e la riattivazione dei programmi in esecuzione
  - **gestione processori**: assegna il processore ai programmi che devono essere eseguiti (e gestisce la cooperazione tra varie CPU nel caso di piu' calcolatori)
  - **gestione memoria** (principale e secondaria): gestisce la collocazione dei dati e dei programmi nella RAM, la memorizzazione e il reperimento delle informazioni sulla memoria secondaria
  - **gestione dispositivi di I/O** (gestisce l'accesso e il controllo dei dispositivi periferici)

Programmi e processi

- **Programma:** insieme statico di istruzioni
- **Processo:** entita' che tiene traccia dello stato dell'esecuzione di un programma
  - Posizione nel programma
  - Valori dei registri della CPU
  - Valori delle celle di M assegnate al programma

- Anche **piu' processi** per lo **stesso programma**
  - Es.: due utenti che usano Word per scrivere due documenti diversi

La gestione dei processi – 1

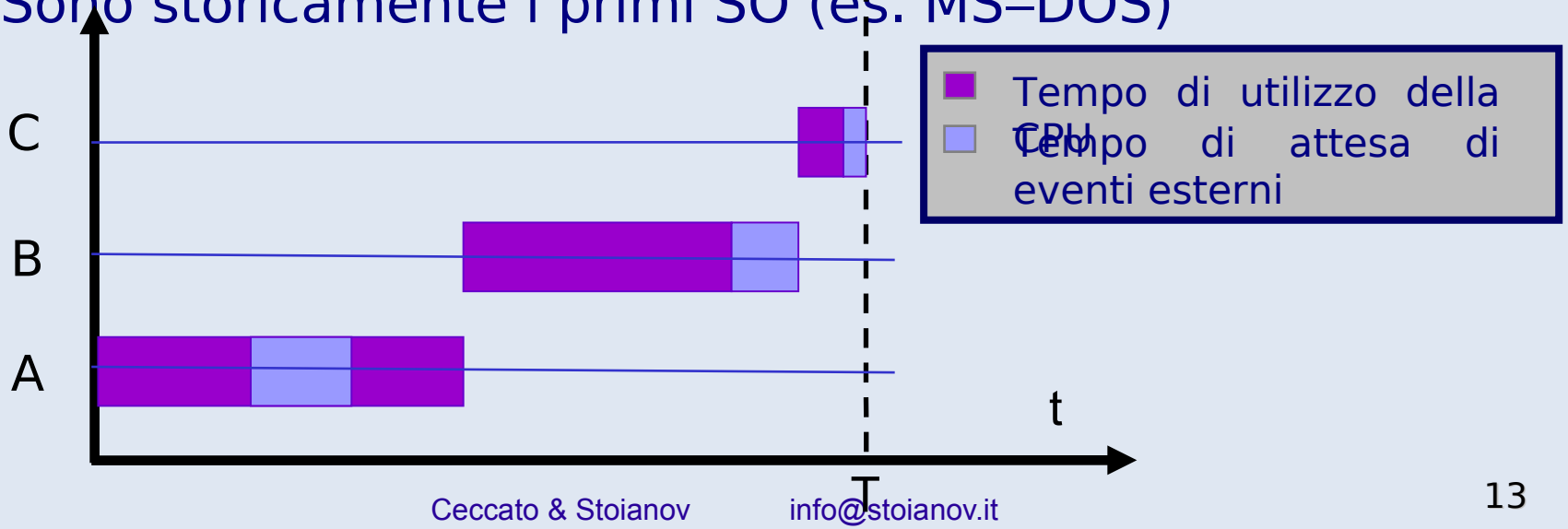
- Un SO consente il caricamento in memoria e l'esecuzione di più programmi che si alternano nell'uso della CPU.
- Per far ciò un programma può essere eseguito, sospeso e fatto ripartire più volte.

La gestione dei processi – 2

- Il sistema operativo è responsabile delle seguenti attività riguardanti la gestione dei processi:
  - creazione e terminazione dei processi
  - sospensione e riattivazione dei processi
  - gestione dei deadlock
  - comunicazione tra processi
  - sincronizzazione tra processi
- **Il gestore dei processi "realizza" una macchina virtuale in cui ciascun programma opera come se avesse a disposizione un'unità di elaborazione dedicata**

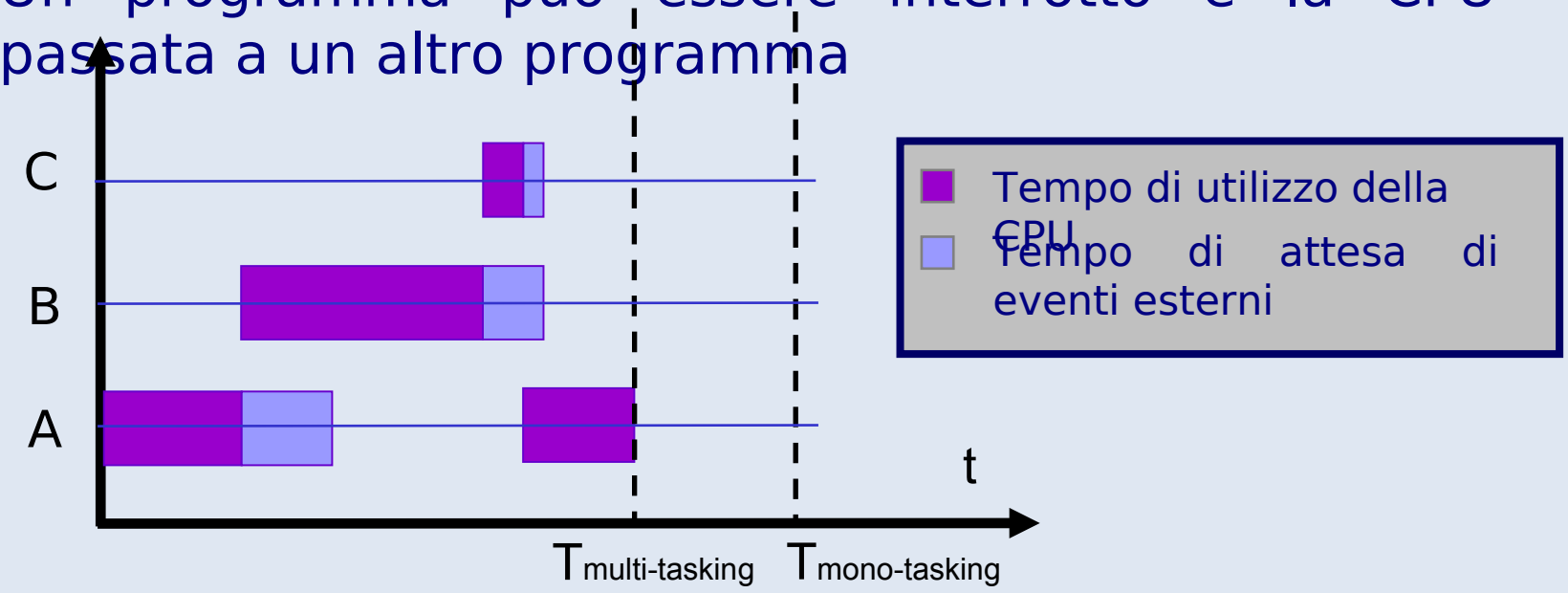
Sistemi mono–tasking

- I SO che gestiscono l'esecuzione di un solo programma per volta (un solo processo) sono detti **mono–tasking**
- Non è possibile sospendere un processo per assegnare la CPU ad un altro
- Sono storicamente i primi SO (es. MS–DOS)



Sistemi multi–tasking

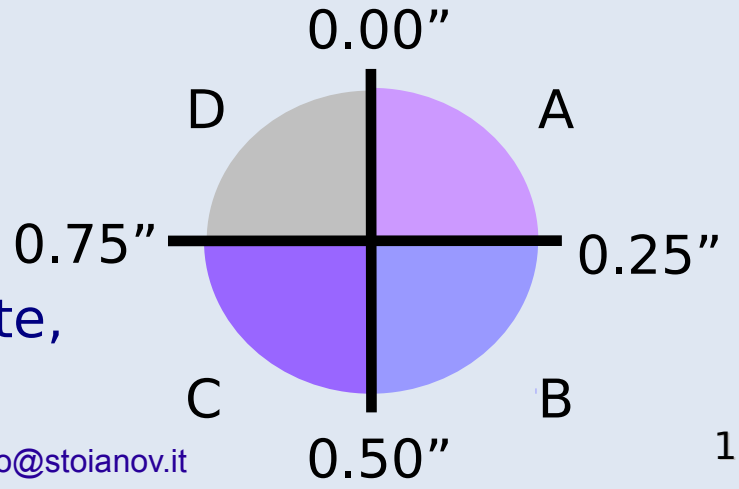
- I SO che permettono l'esecuzione contemporanea di più programmi sono detti multi–tasking o multi–programmati
- Un programma può essere interrotto e la CPU passata a un altro programma



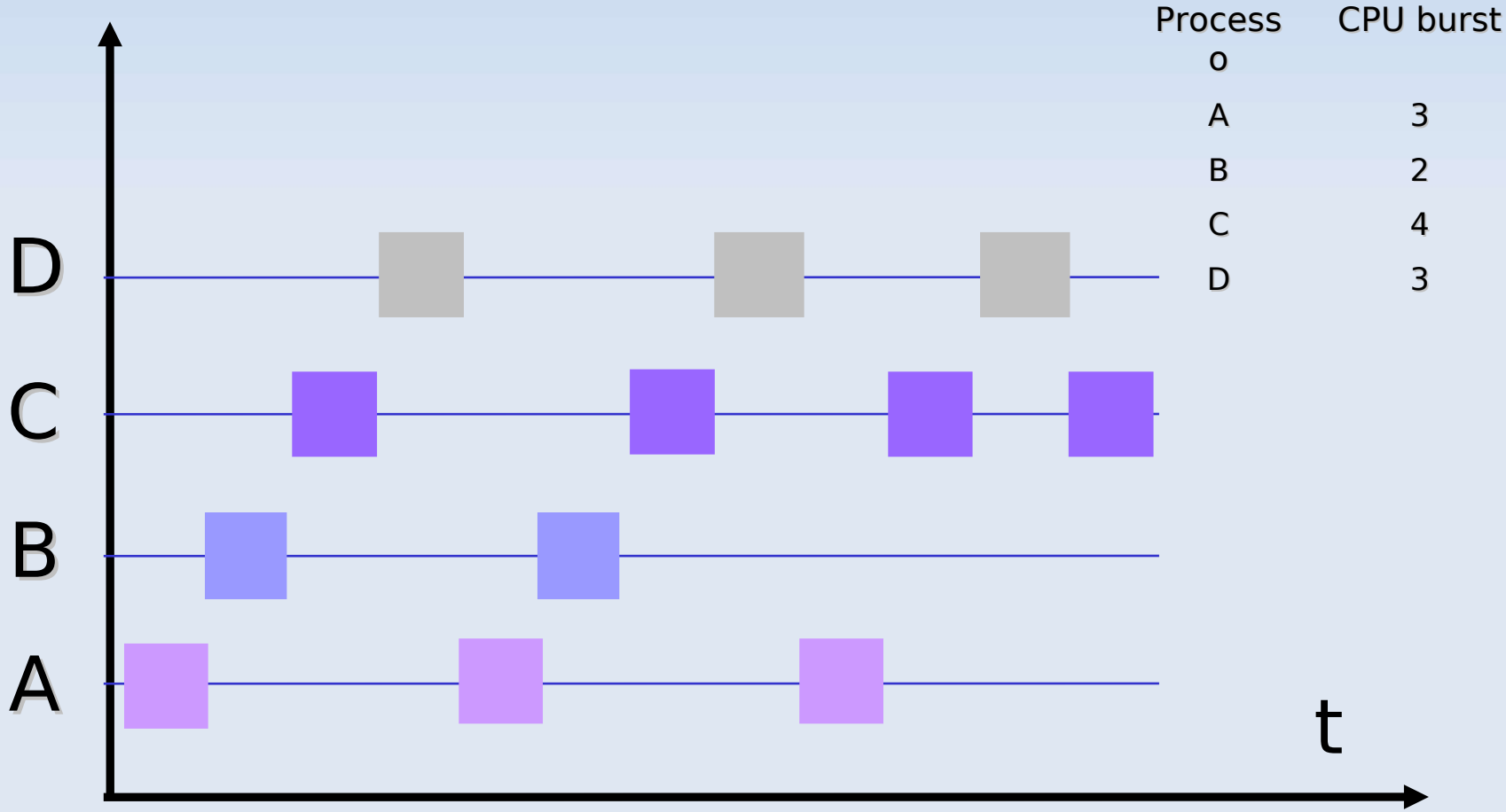
Sistemi time–sharing

- Un'evoluzione dei sistemi multi–tasking sono i sistemi time–sharing
- Ogni processo viene eseguito ciclicamente per piccoli quanti di tempo (TIME SLICE)
- Se la velocità del processore è sufficientemente elevata si ha l'impressione di un'evoluzione parallela dei processi
- Esempio
  - Ipotesi: 1 MIPS, 4 processi, 0.25 s/utente
  - Conseguenze: 0.25 MIPS/utente,

$$T_{ELA} = 4 \times T_{CPU}$$

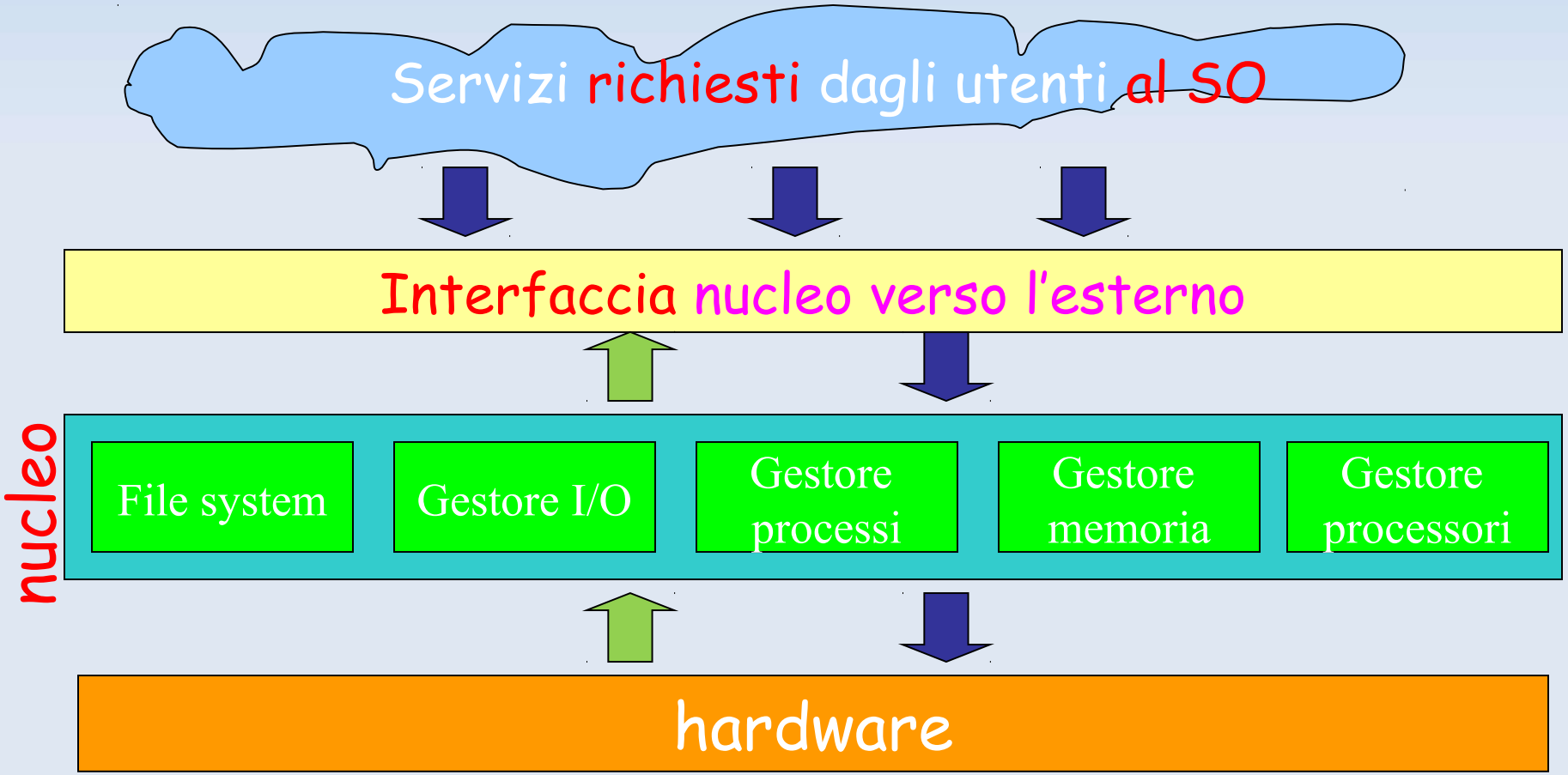


Time–sharing: diagramma temporale



Processo	CPU burst
A	3
B	2
C	4
D	3

Struttura del SO



La gestione della memoria principale – 1

- La memoria principale…
  - …è un “array” di byte indirizzabili singolarmente
  - …è un deposito di dati facilmente accessibile e condiviso tra la CPU ed i dispositivi di I/O
- Il SO è responsabile delle seguenti attività riguardanti la gestione della memoria principale:
  - Tenere traccia di quali parti della memoria sono usate e da chi
  - Decidere quali processi caricare quando diventa disponibile spazio in memoria
  - Allocare e deallocare lo spazio di memoria quando necessario
- **Il gestore di memoria “realizza” una macchina virtuale in cui ciascun programma opera come se avesse a disposizione una memoria dedicata**

La gestione della memoria principale – 2

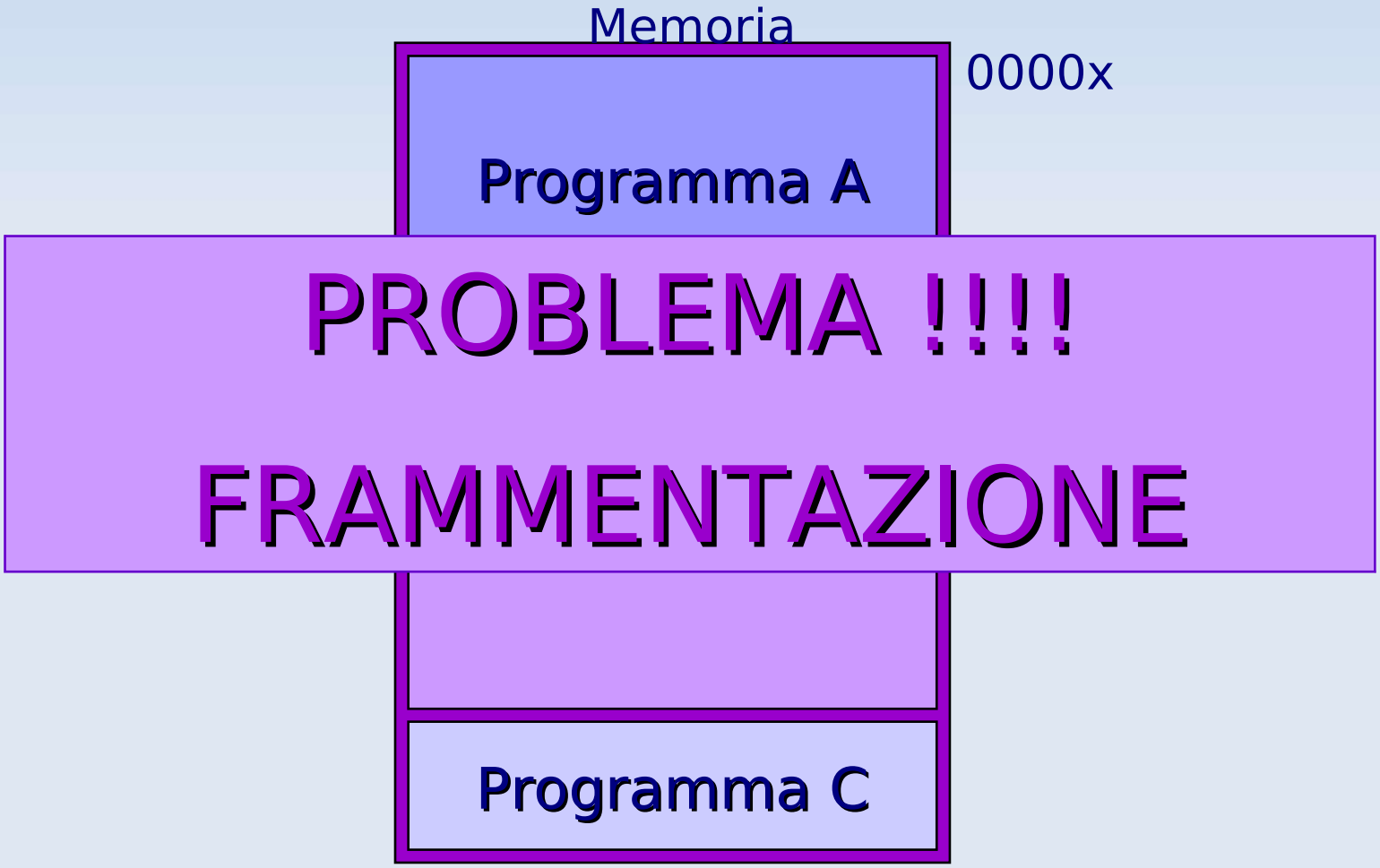
- L’organizzazione e la gestione della memoria centrale è uno degli aspetti più critici nel disegno di un SO
- Il gestore della memoria è quel modulo del SO incaricato di assegnare la memoria ai task (per eseguire un task è necessario che il suo codice sia caricato in memoria)
- La complessità del gestore della memoria dipende dal tipo di SO
- Nei SO multi–tasking, più programmi possono essere caricati contemporaneamente in memoria
- Problema: come allocare lo spazio in maniera ottimale?

Memoria reale

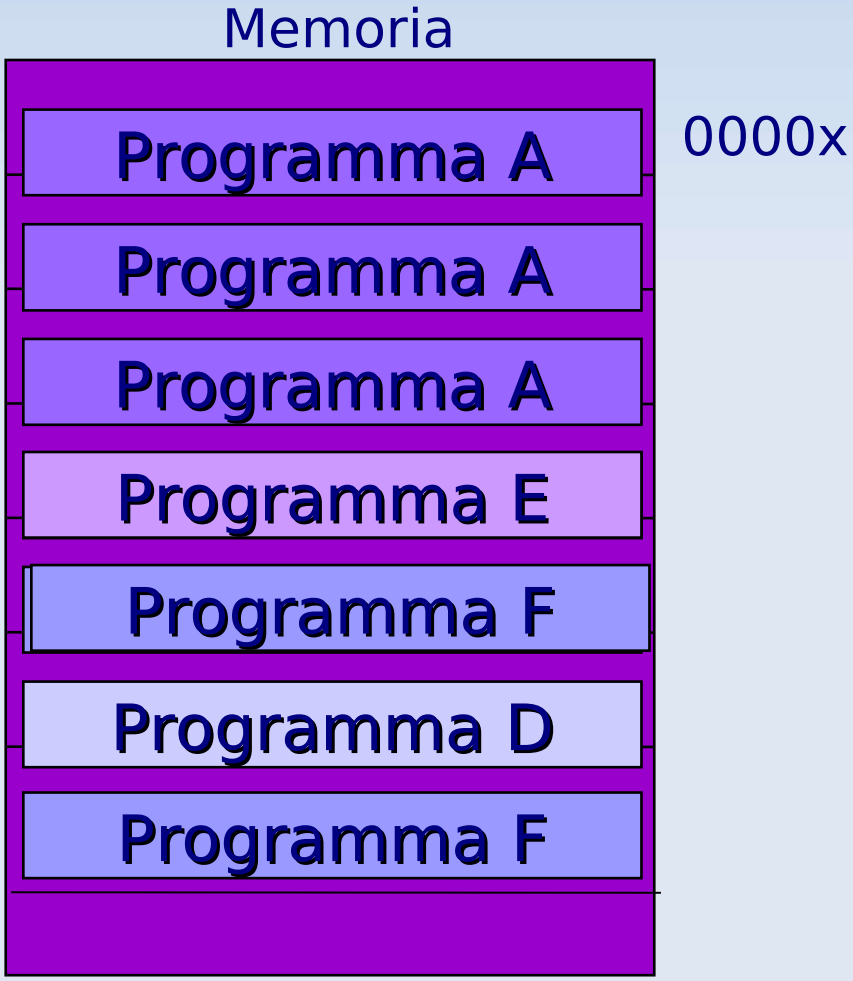
- Nessun problema se un programma per volta (tecnica di overlay)
- Se più task si partiziona la memoria:
  - Partizioni fisse
  - Paginazione

Attenzione: problema della frammentazione della memoria

Allocazione lineare



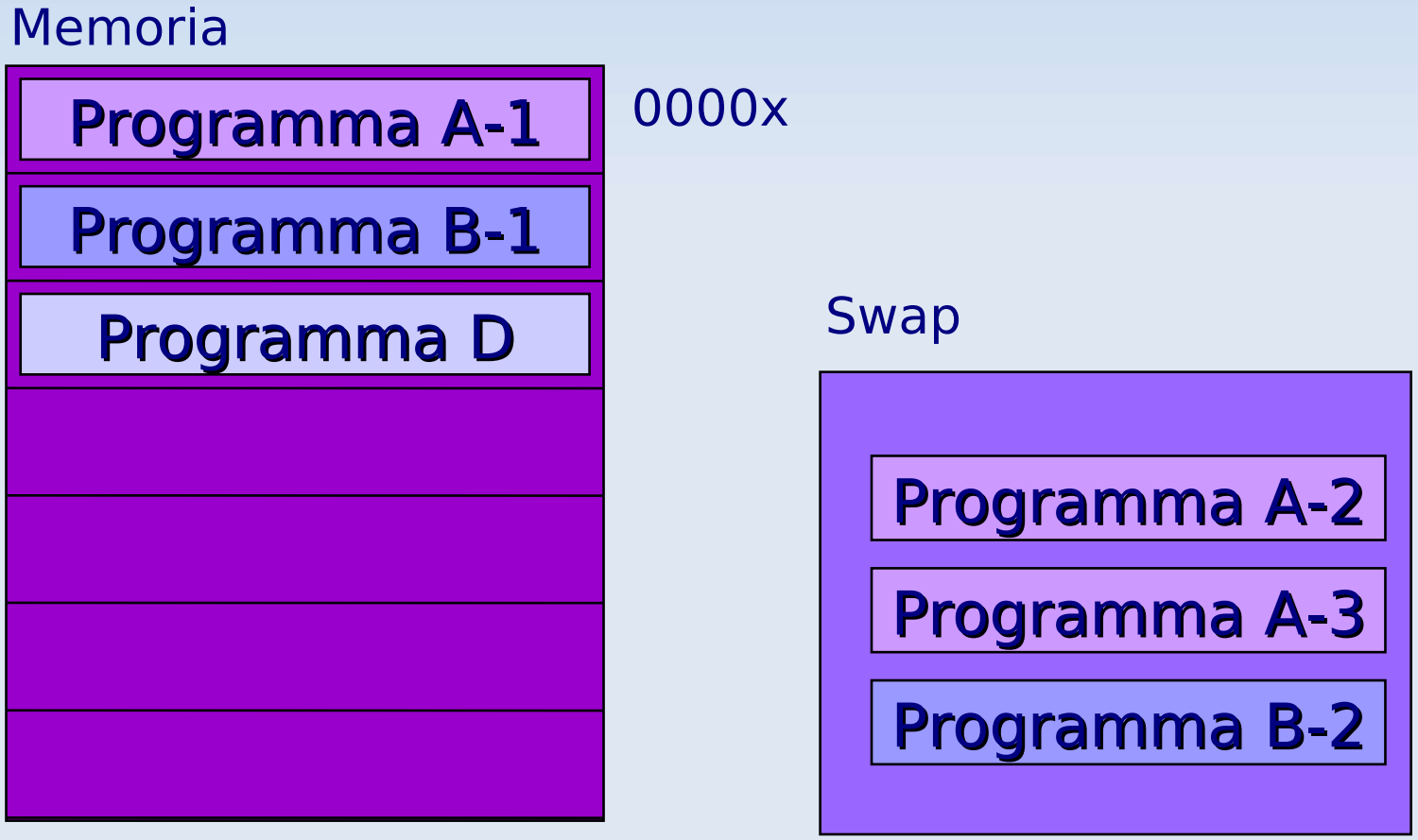
Paginazione



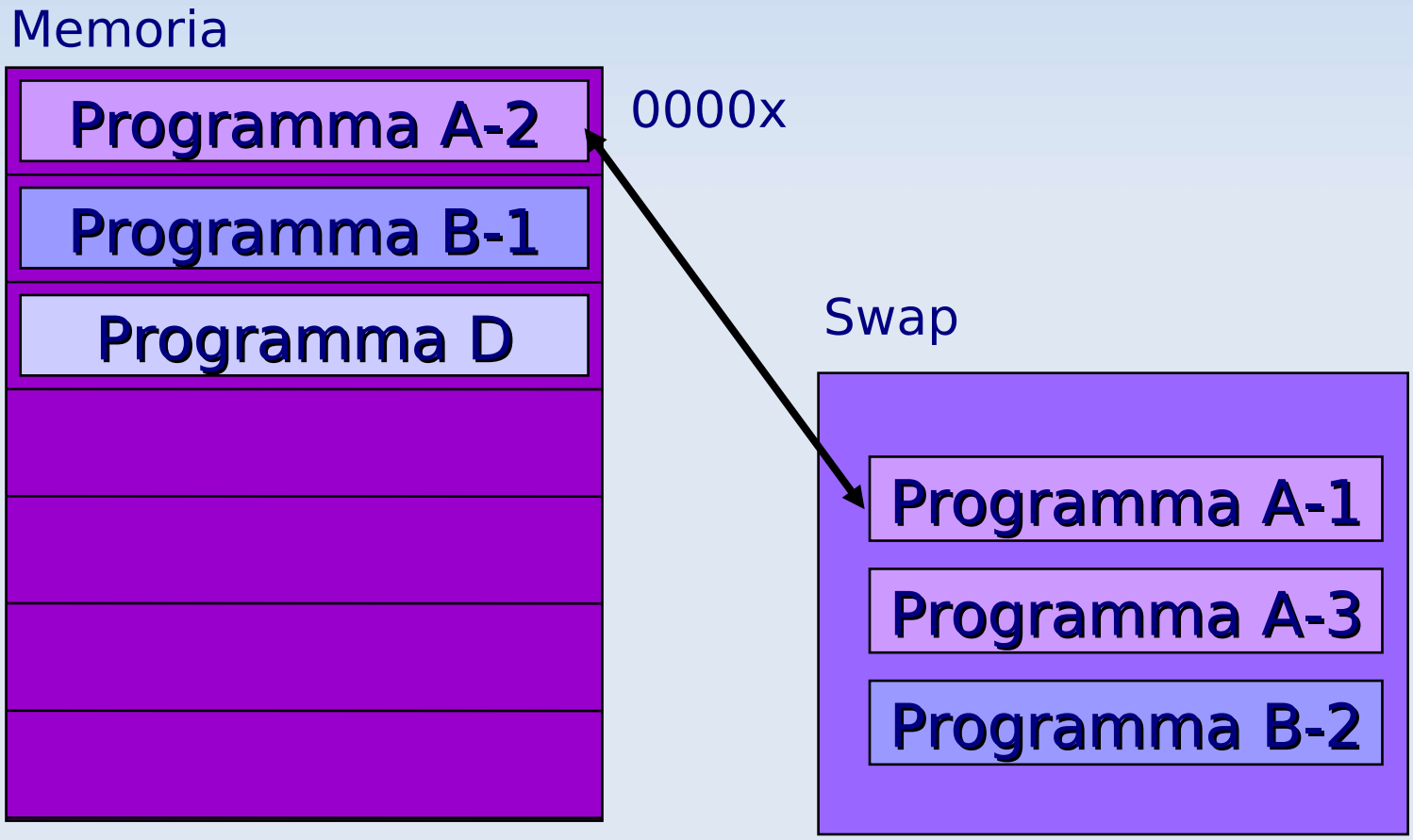
La memoria virtuale – 1

- Spesso la memoria non è sufficiente per contenere completamente tutto il codice dei processi
- Si può ***simulare*** una memoria più grande tenendo nella memoria di sistema (RAM) solo le parti di codice e dati che servono in quel momento
- Si usa il concetto di memoria virtuale
- I dati e le parti di codice relativi a programmi non in esecuzione possono essere tolti dalla memoria centrale e “parcheggiati” su disco nella cosiddetta area di swap
- I processori moderni sono dotati di meccanismi hardware per facilitare la gestione della memoria virtuale

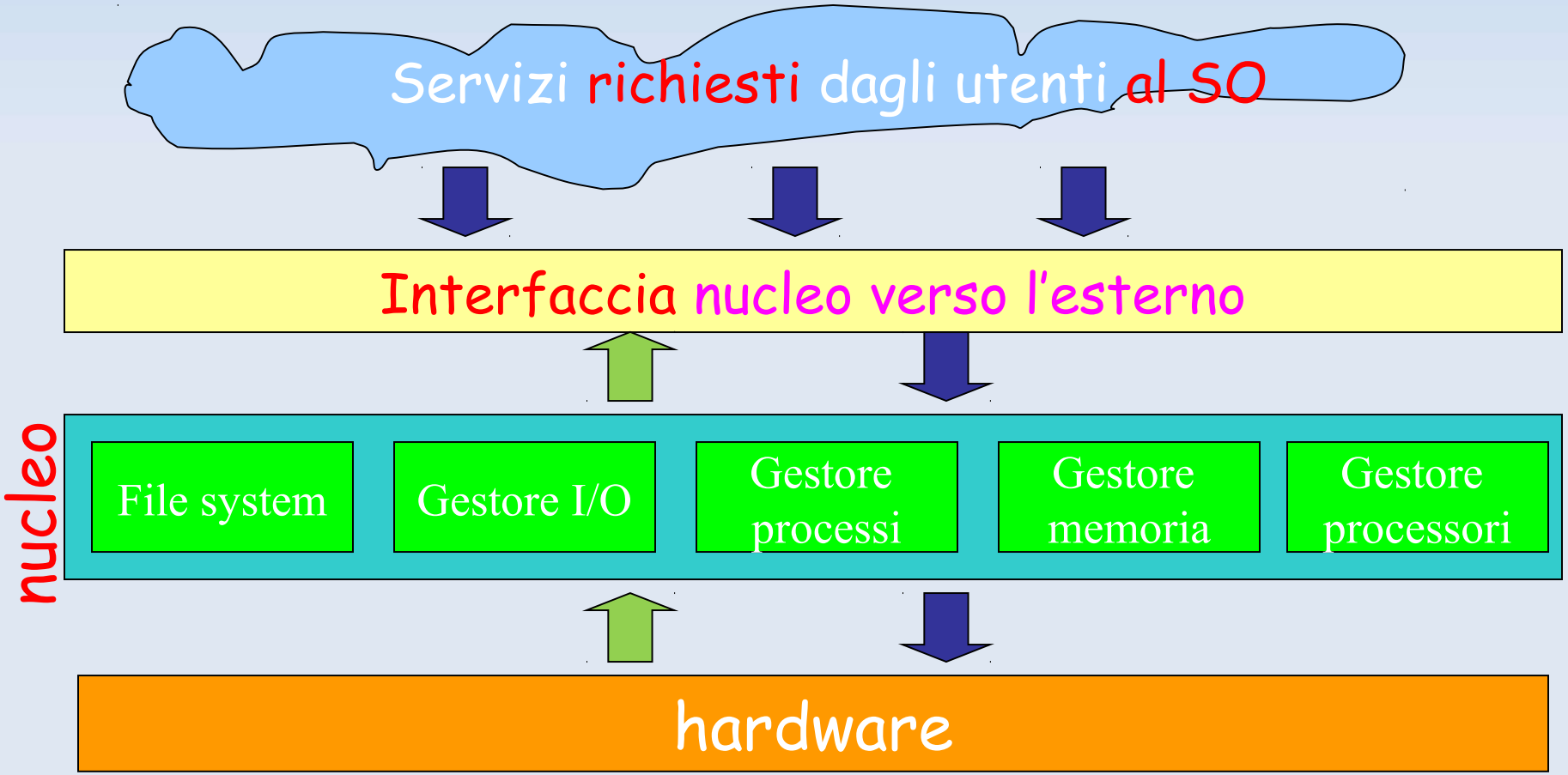
La memoria virtuale – 2



La memoria virtuale – 2



Struttura del SO



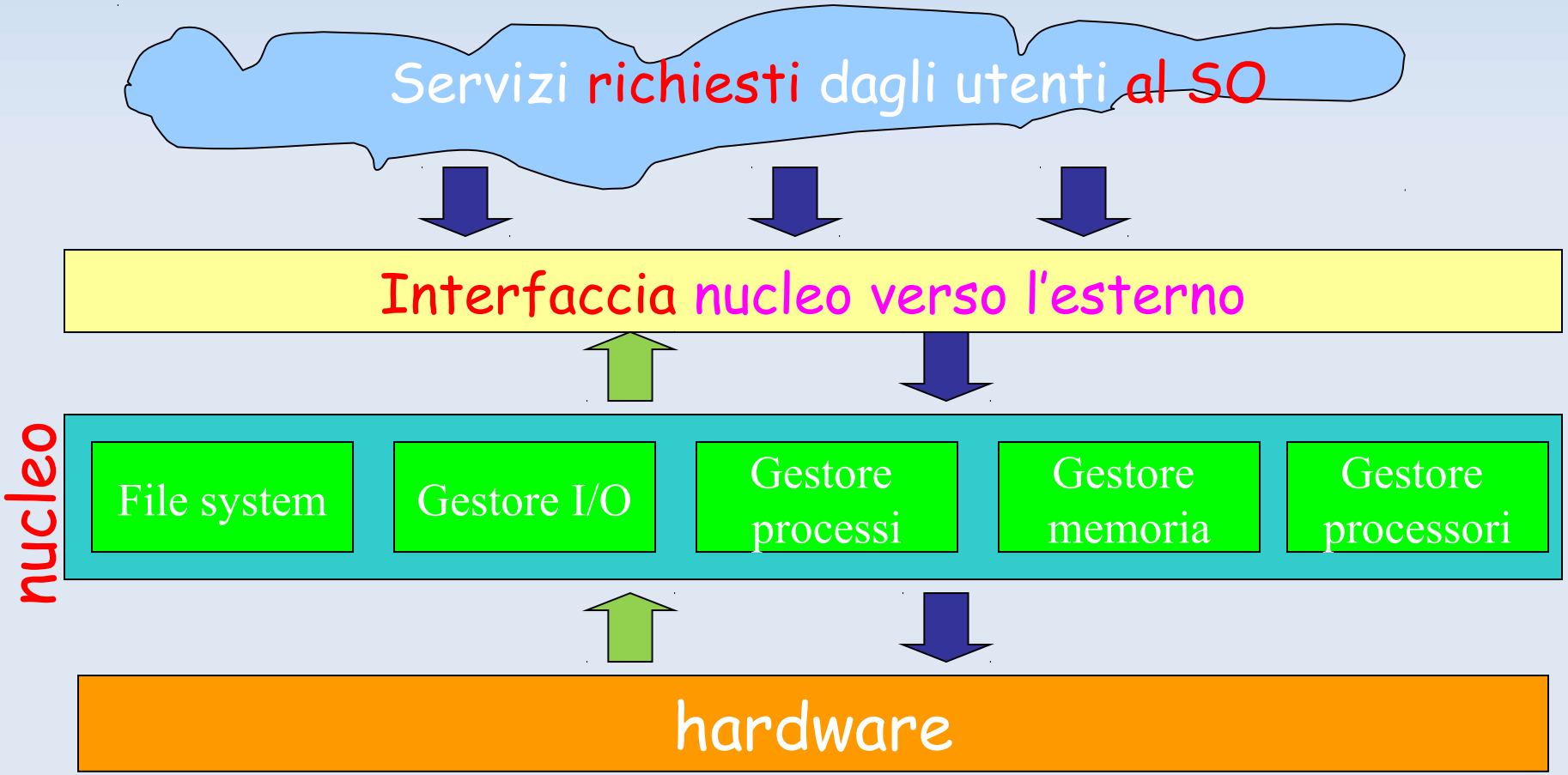
La gestione della memoria secondaria – 1

- Poiché la memoria principale è volatile e troppo piccola per contenere tutti i dati e tutti i programmi in modo permanente, il computer è dotato di memoria secondaria
  - In generale, la memoria secondaria è data da hard disk e dischi ottici
- Il SO garantisce una visione logica uniforme del processo di memorizzazione:
  - Astrae dalle caratteristiche fisiche dei dispositivi per definire un'unità di memorizzazione logica – il *file*
  - Ciascuna periferica viene controllata dal relativo device driver, che nasconde all'utente le caratteristiche fisiche variabili dell'hardware: modalità e velocità di accesso, capacità, velocità di trasferimento

La gestione della memoria secondaria – 2

- Il SO è responsabile delle seguenti attività riguardanti la gestione della memoria secondaria:
  - Allocazione dello spazio
  - Gestione dello spazio libero
  - Ordinamento efficiente delle richieste di accesso al disco

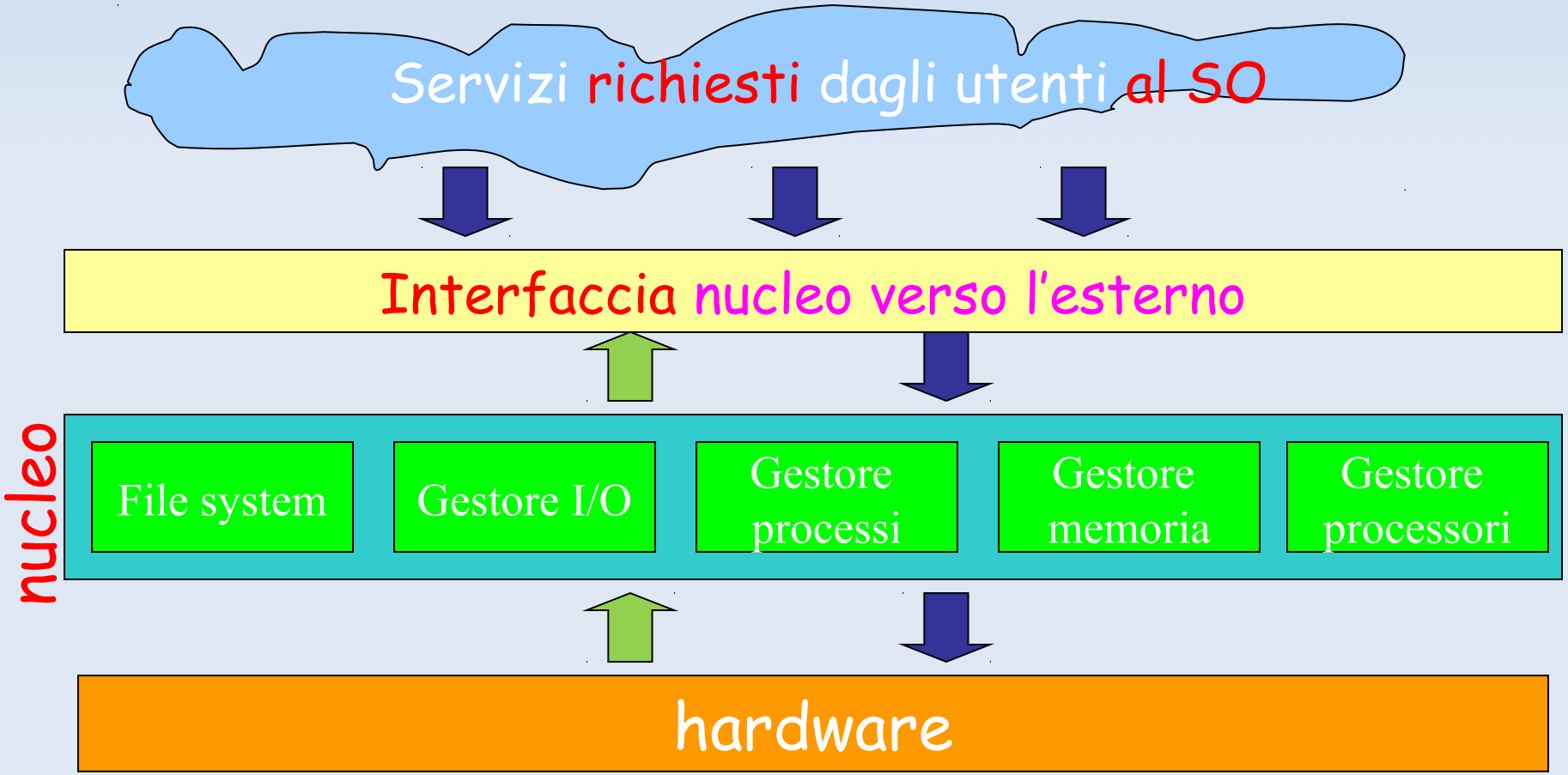
Struttura del SO



La gestione dei dispositivi di I/O

- La gestione dell'I/O richiede:
  - Un'interfaccia comune per la gestione dei ***device driver***
  - Un insieme di driver per dispositivi hardware specifici
  - Un sistema di gestione di buffer per il caching delle informazioni
- Il gestore dei dispositivi di I/O è il modulo del SO incaricato di assegnare i dispositivi ai task che ne fanno richiesta e di controllare i dispositivi stessi
- Da esso dipende la qualità e il tipo di periferiche riconosciute dal sistema
- **Il gestore delle periferiche offre all'utente una versione astratta delle periferiche hardware; l'utente ha a disposizione un insieme di procedure standard di alto livello per leggere/scrivere da/su una periferica che "percepisce" come dedicata**

Struttura del SO



La gestione del file system

- Il **file** è l'astrazione informatica di un archivio di dati
  - Il concetto di file è indipendente dal mezzo sul quale viene memorizzato (che ha caratteristiche proprie e propria organizzazione fisica)
- Un file system è composto da un insieme di file
- Il SO è responsabile delle seguenti attività riguardanti la gestione del file system:
  - Creazione e cancellazione di file
  - Creazione e cancellazione di directory
  - Manipolazione di file e directory
  - Codifica del file system sulla memoria secondaria

File

Per ogni file vengono memorizzate varie ulteriori informazioni

identificatore: nomefile.estensione

data di creazione e ultima modifica

dimensione

posizione effettiva dei dati nella memoria di massa

diritti di accesso

etc

Estensioni dei file

.exe : programma eseguibile

.txt : file di testo

.doc : file di Microsoft Word

.xls : file di Microsoft Excel

.jpg, .gif : file di immagini

.wav, .mp3 : file di suoni

.mpg, .avi : file di filmati

.c, .cpp, .java : file di programmi C, C++, Java

Operazioni su file

Creazione
Apertura
Chiusura
Cancellazione
Copia
Rinomina
Visualizzazione
Lettura
Scrittura
Modifica
...

Organizzazione dei file

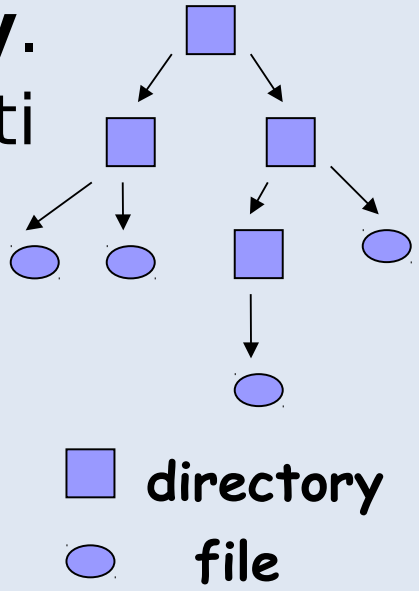
I file sono organizzati **logicamente** in modo **gerarchico**.

la organizzazione **logica** permette l'astrazione dalla loro organizzazione **fisica** nella memoria di massa.

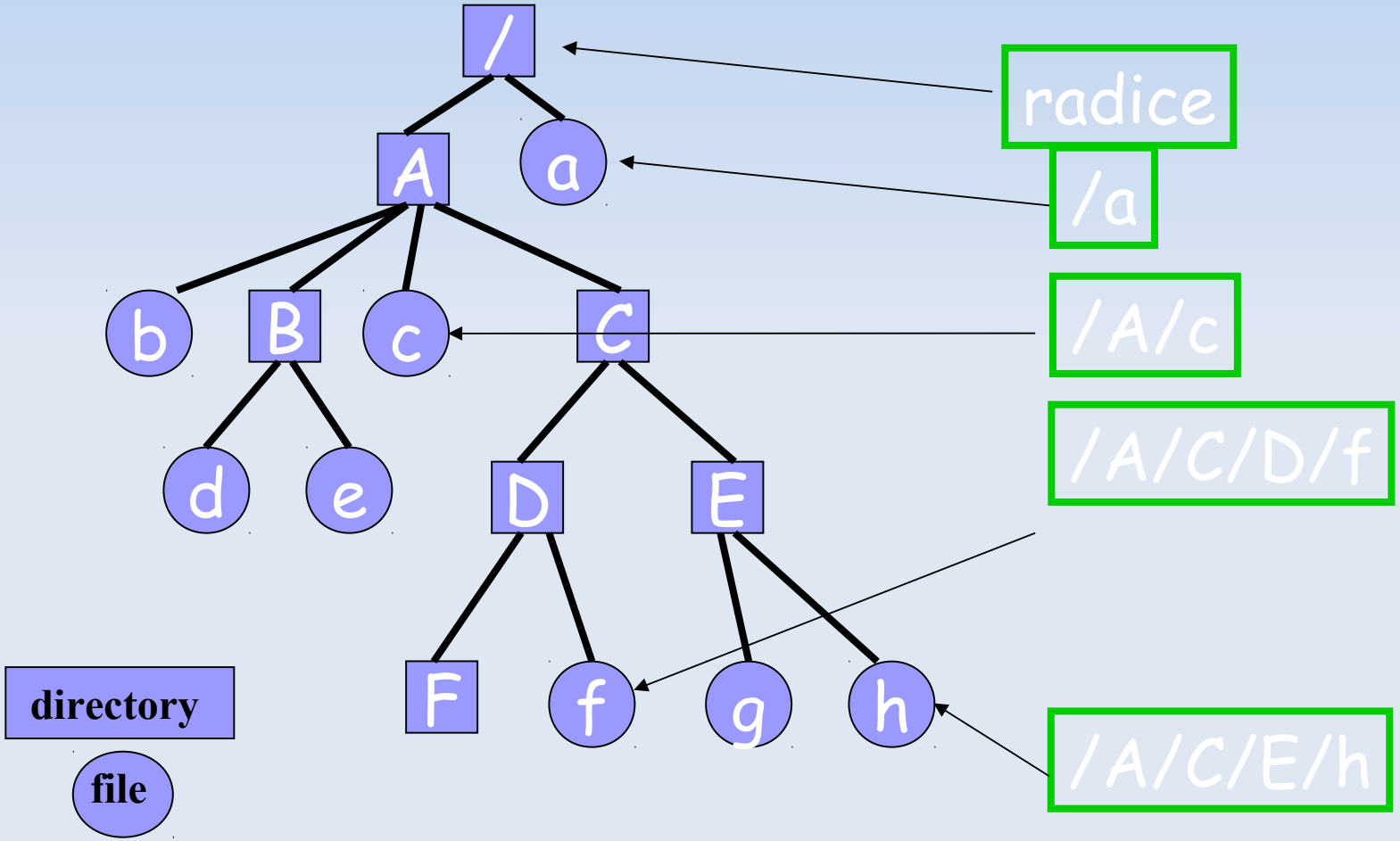
L'unità principale di raggruppamento si chiama **directory**: un insieme di **file** e altre **directory.**

Organizzazione ad albero

- I dischi fissi sono divisi in partizioni.
- Ogni partizione e' organizzata gerarchicamente come un **albero** rovesciato (come quello genealogico).
- Ogni nodo dell'albero e **file** o **directory**.
- I nodi sono organizzati per livelli e legati con relazioni del tipo **padre-figlo**.
- Un figlo ha un solo padre.
- Un padre potrebbe avere più figli.
- **Directory:** un nodo con figli.
- **File**: un nodo senza figli.
- Il padre più in alto = **radice**



Linux: Organizzazione dei file



path (percorso): la sequenza di nodi dalla radice "/" fino ad un certo nodo

Percorso relativo e assoluto

Percorso assoluto: un percorso che specifica la posizione di un file a partire dalla radice:

Esempio: percorso assoluto del file **d**:
/A/B/d (inizia per “/”)

Percorso relativo: la posizione di un file a partire da una posizione al interno dell'albero dei file (solitamente da una “directory corrente”)

Esempio: direttory corrente: **/A**

percorso relativo del file **d**:

B/d (inizia senza “**/**”)

Vari sistemi operativi

Unix

La sua storia inizia nel 1969
Sistema portabile scritto in linguaggio C
SO multi-utente, multi-tasking, time-sharing

Dos

Progettato da Microsoft nel 1980 per IBM
SO mono-utente,mono-tasking
Interfaccia a linea di comando

Windows

Progettato da Microsoft nel 1985
SO multi-utente,multi-tasking, time-sharing
Interfaccia grafica a finestre

Mac OS

Progettato dalla Apple a partire dal 1984
è il sistema operativo per i computer Macintosh

Occupiamoci in particolare di Linux

Nato negli anni 90

Basato sull'esperienza di Unix

SO multi-tasking, time-sharing,multi-utente a memoria protetta:

ogni processo ha una determinata zona di memoria assegnata

Varie distribuzioni disponibili

Raccolte di programmi componenti il sistema, configurati in modo da integrarsi meglio tra di loro e corredati da tool che semplificano l'installazione, la configurazione e la gestione del sistema (es. Ubuntu)

Linux: permessi sui file

Ogni file ha una serie di permessi associati che indicano chi puo' eseguire operazioni sui file e che tipo di operazioni

Garantisce la sicurezza delle attivita' degli utenti e del sistema stesso

Tipi di permesso

Lettura, scrittura, esecuzione

Destinatari

Proprietario, gruppo di utenti, altri

Linux: permessi sui file

I permessi

- rwxr-xr-x

- I permessi sono 9 caratteri, raggruppati per

3,

che indicano rispettivamente i permessi per l'owner, per il gruppo e per gli altri utenti.

Linux: permessi sui file

--- → nessuna autorizzazione

--x → esecuzione

-w- → scrittura

-wx → scrittura, esecuzione

r-- → lettura

r-x → lettura,esecuzione

rw- → lettura, scrittura

rwx → lettura, scrittura, esecuzione

Linux: permessi sui file

--- → nessuna autorizzazione

--x → esecuzione

-w- → scrittura

-wx → scrittura, esecuzione

r-- → lettura

r-x → lettura,esecuzione

rw- → lettura, scrittura

rwx → lettura, scrittura, esecuzione

Linux: interfaccia grafica e shell

Interfaccia grafica

permette di lavorare graficamente con l'utilizzo di finestre

Per essere piu' intuitivo e semplice da usare

Ricorda l'interfaccia di Windows

- Barra delle applicazioni
- Menu di avvio
- Visualizzazione del file system attraverso icone

Shell

l'interfaccia testuale tramite la quale l'utente puo' operare con il sistema

Linux: interfaccia grafica

Per aprire una shell:

Cliccare su ‘Applicazioni’ nel menu di avvio dove si trovano tutti gli applicativi del sistema

Cliccare su ‘Terminale’

Linux: comandi

Sintassi generale di un comando (dalla shell)

nome_comando [opzioni] [argomento1…argomentoN]

Tra parentesi quadre vengono poste le stringhe che non sono necessariamente usate

Le stringhe vanno separate da spazi

I nomi dei comandi vanno specificate con lettere minuscole (Linux e' *case sensitive*)

Linux: comando pwd

Indica la posizione attuale nel file system con il percorso completo

pwd

Esempio: pwd → /home/bio140

Esercizio: visualizzare la posizione all'avvio della shell

Linux: comando ls

Visualizza il contenuto di una cartella o i file

ls [opzione] [nome_directory] [nome_file]

Opzioni:

-a

- elenca tutti i file compresi quelli nascosti

- l

- per ogni elemento della lista fornisce informazioni aggiuntive

-R

- discende ricorsivamente anche tutte le sottodirectory

-t

- Elenca gli elementi in ordine temporale

Esercizio: visualizzare il contenuto della home con l'ausilio delle varie opzioni

Linux: comando cd

Utilizzato per spostarsi tra directory all'interno del file system

cd [nome-directory]

- Entra nella directory specificata
- Esempio: cd biologia

cd ..

- Si sale di un livello nell'albero, posizionandosi nella directory del livello immediatamente superiore

cd

- Si ritorna all propria home

Esercizio: spostarsi nella cartella Desktop visualizzarne il contenuto e poi ritornare alla home

Linux: comando mkdir

Utilizzato per creare directory

mkdir [opzioni] nome_directory

Nomi utilizzabili

Caratteri alfabetici e numerici, _, -, .

Esempio

mkdir biologia

Esercizio:

creare una directory ‘biologia’ e una directory ‘documenti’ nella propria home

crerare una directory ‘biologia2’ nella directory ‘biologia’

posizionarsi nella directory ‘documenti’

Linux: comando touch

Crea un file vuoto

touch nome_file

Esempio

Touch file.txt

Esercizio:

creare un file vuoto ‘prova.txt’ nella directory ‘documenti’

creare un file vuoto ‘bio.txt’ nella directory ‘biologia’

Linux: comando cp

Copia il file
nella directory indicata
cp [opzioni] file_origine nome_directory

nel file di destinazione
cp [opzioni] file_origine file_destinazione

Esempio: cp file1.txt file2.txt

Esercizio 1: copiare il file ‘prova.txt’ che si trova nella directory ‘documenti’ nel file ‘prova2.txt’

Esercizio 2: copiare il file prova2.txt nella directory ‘biologia’

Linux: comando rm

Cancella i file specificati (non le directory)

rm [opzioni] nome_file

Esempio: rm file1.txt

Esercizio:

creare un sottodirectory ‘corso’ nella directory ‘biologia’

copiare nella directory ‘corso’ il file ‘prova2.txt’ presente nella directory ‘biologia’

rimuovere il file ‘prova2.txt’ dalla directory ‘biologia’

Linux: comando **rmdir**

Cancella una directory se e' vuota

rmdir nome_directory

Esempio: rmdir biologia

Esercizio: creare il file vuoto 'nuovo.txt' nella directory 'corso' e poi rimuovere la directory 'corso'

Linux: comando pico

Semplice editor di testo dalla shell

Permette di creare, leggere o modificare file di testo

pico nome_file

Comandi principali

Ctrl O → salva file

Ctrl X → esce

Esercizio: aprire con pico il file ‘prova.txt’, che si trova nella directory ‘documenti’ , scrivere ‘Ciao’, salvare il file e uscire

Linux: comando man

Guida per i comandi di Linux

man nome_comando

Indica

Come si usa il comando

Tutte le opzioni permesse per i varia comandi

Si naviga attraverso le frecce

Tasto **q** per chiudere il manuale

Esempio: man mkdir, man ls

Linux: comando **more**

- Il comando:

>> more nome_file

stampa a schermo il contenuto del file “nome_file”, una videata alla volta. Per scorrere alla prossima pagina si usa la barra spaziatrice, per terminare la visualizzazione si usa il tasto q

Linux: comando **less**

- Il comando:

>> less nome_file

stampa a schermo il contenuto del file “nome_file” e permette di scorrere la visualizzazione per riga utilizzando i tasti “freccia in alto” e “freccia in basso”.

- Per terminare la visualizzazione si usa il tasto “q”

(quando finisce esce comunque da solo)

Il contenuto viene visualizzato una videata alla volta. Per scorrere alla prossima pagina si usa la barra spaziatrice, per terminare la visualizzazione si usa il tasto q

Linux: comando top

top [opzioni]

Top produce un output a monitor diviso in una parte alta, che contiene informazioni generali sul sistema, e in una parte sottostante, che mostra i processi e il loro utilizzo di CPU

Opzioni:

- k + numero processo, termina il processo in questione.
- r + numero processo, modifica il valore nice di un processo; cioè la sua priorità, da -20 (massima priorità) a 19 (minima priorità). Quindi un valore nice negativo migliora le prestazioni di esecuzione di un processo rispetto ad uno positivo.

Uscite da 'top' premendo il tasto q.

Open Office

Sviluppato dalla Sun Microsystem

Software free multipiattaforma

Sviluppato per essere concorrente e compatibile con Microsoft Office

Composto da diversi applicativi

Writer: elaboratore di testi (Word)

Calc: foglio di calcolo (Excel)

Impress: presentazioni (PowerPoint)

Base: gestione di DB (Access)

Per aprire gli applicativi

Applicazioni dal menu' di avvio