Capitolo 7
Le Infrastrutture Software

Cos'è un Sistema Operativo?

- Un programma che agisce come tramite tra l'utente e gli elementi fisici del calcolatore.
- E' un insieme di programmi (software) che:
  - gestisce gli elementi fisici di un calcolatore (hardware),
  - fornisce una piattaforma ai programmi di applicazione
  - agisce da intermediario tra l'utente e la struttura fisica del calcolatore

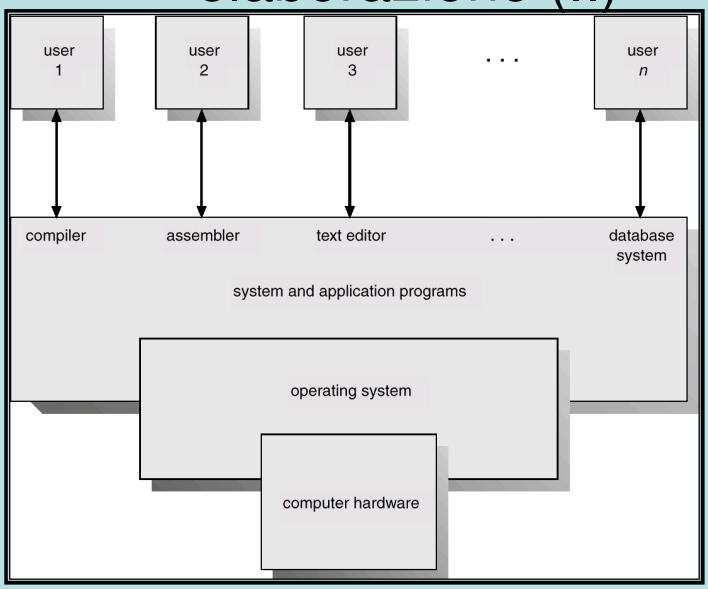
Cos'è un Sistema Operativo? (II)

- Scopi di un sistema operativo:
  - Eseguire programmi utente e rendere più semplice la soluzione dei problemi dell'utente.
  - Rendere conveniente ed efficiente l'utilizzo del sistema di calcolo.
- Un sistema operativo deve assicurare il corretto funzionamento di un calcolatore.
- Funzioni del Sistema Operativo
  - Estendere e astrarre l'hardware (per semplificare la programmazione, per rendere i programmi portabili, etc..).
    - (ad es. un "file" è un astrazione)
  - Gestire le risorse
    - (ad es. suddividere stampanti, dischi, tempo di CPU fra più programmi)

Componenti di un sistema di elaborazione

1. Hardware – (CPU, memoria, dispositivi di I/O, etc.).
2. Sistema Operativo – controlla e coordina l'uso delle risorse hardware su richiesta dei (vari) programmi applicativi dei (vari) utenti
3. Programmi applicativi – definiscono il modo in cui le risorse del sistema sono utilizzate per risolvere i problemi computazionali degli utenti (compilatori, database, video games, programmi finanziari, etc.).
4. Utenti (persone, macchinari, altri computer, etc.).

Componenti di un sistema di elaborazione (II)



Ruolo del Sistema Operativo

- I sistemi operativi (S.O. in breve) esistono perché forniscono agli utenti uno strumento conveniente per l'uso di un sistema di calcolo
- Convenienza:
  - facilità d'uso,
  - efficienza uso risorse.
- Gran parte della teoria dei S.O. si è concentrata sull'efficienza.
- Inoltre, hardware e S.O. si sono influenzati vicendevolmente.

Ruolo del Sistema Operativo: punto di vista dell'utente

- **PC**
  - Il sistema operativo è progettato principalmente per facilitare l'uso del computer.
- **Mainframe e Minicomputer**
  - Occorre massimizzare l'uso delle risorse.
- **Workstation**
  - Compromesso ottimale tra l'uso delle risorse individuali e risorse condivise.
- **Palmari e simili**
  - Progettati per l'uso individuale prestando attenzione alle prestazioni della batteria
- **Sistemi Embedded**
  - Concepiti per funzionare senza l'intervento dell'utente

Ruolo del Sistema Operativo: punto di vista del sistema

- Il sistema operativo è il programma più strettamente connesso con l'hardware.
- Quindi, è:
  - **allocatore di risorse:** di fronte a richieste conflittuali decide come assegnare equamente ed efficientemente le risorse ai programmi,
  - **programma di controllo:** garantisce l'esecuzione dei programmi senza errori e usi impropri del computer,
  - **esecutore di funzioni comuni:** esegue funzioni di utilità generale comuni ai diversi programmi (ad es. routine di I/O),
  - **nucleo** (Kernel): l'unico programma sempre in esecuzione (tutti gli altri sono "programmi applicativi").

Servizi di un sistema operativo

- Un S.O. offre un ambiente in cui eseguire i programmi e fornire servizi ai programmi e ai loro utenti.
- Ecco una lista di alcune classi di servizi comuni offerti dal S.O. per rendere più agevole la programmazione:
  - Interfaccia con l'utente:
    - interfaccia a riga di comando (CLI) - basata su stringhe che codificano i comandi, insieme ad un metodo per inserirli e modificarli,
    - interfaccia a lotti - comandi e relative direttive sono codificati nei file ed eseguiti successivamente a lotti,
    - interfaccia grafica con l'utente (GUI) - sistema grafico a finestre dotato di un dispositivo puntatore (ad es. il mouse).
  - Esecuzione di un programma – il sistema deve poter caricare un programma in memoria ed eseguirlo.

Servizi di un sistema operativo (II)

- Operazioni di I/O - i programmi utenti non possono eseguire direttamente operazioni di I/O:
  - S.O. deve fornire strumenti per permettere l'esecuzione di operazioni di I/O.
- Gestione del file system – esecuzione di operazioni di lettura, scrittura, creazione e cancellazione file.
- Comunicazioni– scambi di informazioni tra processi in esecuzione sullo stesso calcolatore o collegati tra loro per mezzo di una rete.
  - Realizzate tramite *memoria condivisa* o *scambio di messaggi*.
- Rilevamento di errori – assicurare la correttezza della computazione rilevando eventuali errori di CPU, di memoria, di I/O o in programmi utenti.

Servizi di un sistema operativo (III)

- Esiste un'altra serie di funzioni del S.O. che non riguarda direttamente l'utente ma assicura il funzionamento efficiente del sistema stesso:
  - Assegnazione delle risorse – allocare risorse a più utenti o processi che sono concorrentemente in esecuzione.
  - Contabilizzazione dell'uso delle risorse – registrare quali utenti usino il calcolatore, segnalando quali e quante risorse impieghino.
  - Protezione e sicurezza – assicurare il controllo dell'accesso a tutte le risorse condivise di sistema identificando l'utente ad ogni suo accesso.

Vantaggi di un SO

- Sono legati alla possibilità di definire modalità standard di interfaccia con i dispositivi fisici, cosicché sia possibile:
  - sviluppare programmi in modo semplice, modulare ed indipendente dallo specifico calcolatore su cui viene fatto funzionare il sistema operativo;
  - aggiornare il software di base e l'hardware in modo trasparente ai programmi applicativi e all'utente, ossia senza che vengano influenzati dall'operazione.

Struttura del sistema operativo

- Concetto chiave è quello della **multiprogrammazione:**
  - necessaria per aumentare l'efficienza.
- Un solo utente non può tenere CPU e dispositivi I/O occupati per tutto il tempo.
- La multiprogrammazione consente di aumentare la percentuale di utilizzo della CPU organizzando i lavori in modo tale da mantenerla in continua attività.
- Un sottoinsieme dei job si trova in memoria centrale (**job pool**).
- Un job viene selezionato (**job scheduling**) ed eseguito.
- Quando il job è in attesa (ad es. di un' operazione di I/O), il S.O. esegue un altro job.

Struttura del sistema operativo (II)

- Altro concetto chiave: **timesharing** (multitasking):
  - estensione logica della multiprogrammazione,
  - la CPU commuta tra i job così frequentemente che gli utenti possono interagire con ciascun job mentre è in esecuzione, realizzando una computazione interattiva.
- Tempo di Risposta < 1 secondo.
- Ciascun utente ha almeno un processo in esecuzione in memoria.
- Se diversi processi sono pronti per essere eseguiti sarà necessaria la **schedulazione** della CPU.

Struttura del sistema operativo (III)

- Se lo spazio di memoria non è sufficiente per contenere tutti i processi,
  - tramite lo swapping alcuni processi verranno spostati temporaneamente su memoria di massa e poi riportati in memoria centrale per essere eseguiti.
- La **memoria virtuale** permette l'esecuzione di processi che non sono completamente in memoria e separa la memoria fisica da quella logica.

Attività del S.O.: gestione delle interruzioni

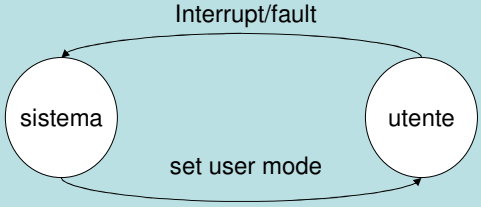
- I sistemi operativi moderni sono caratterizzati dal fatto di essere *guidati dalle interruzioni (interrupt driven*).
  - se non ci sono processi da eseguire, dispositivi di I/O da servire o utenti con cui interagire, il S.O. resta inattivo nell'attesa che accada qualcosa.
- In presenza di una interruzione:
  - il sistema operativo preserva lo stato della CPU salvando lo stato dei registri e del contatore di programma prima di servire l'interruzione.
  - Determina di che tipo sia l'interruzione.
  - Segmenti diversi di codice determinano quale azione debba essere presa per ciascun tipo di interrupt.
  - Dopo aver servito l'interruzione il S.O. ripristina lo stato della CPU (ad es. i registri) e del contatore di programma originali.

Duplice modo di funzionamento (dual mode)

- La protezione deve essere garantita per qualsiasi risorsa condivisibile del sistema.
- L'architettura del sistema deve supportare almeno due distinti *modi* di funzionamento:
  1. *Modo d'utente* (*User mode)* in cui avviene l'esecuzione dei programmi utente
  2. *Modo di sistema (Monitor mode* o *kernel mode* o *system mode*) in cui avviene l'esecuzione delle chiamate e dei programmi di sistema.
- Un *bit di modo (mode bit)* di cui deve essere dotata l'architettura (hardware) della CPU indica il modo corrente:
  - sistema (0),
  - utente (1).

Duplice modo di funzionamento (dual mode) (II)

- In presenza di un'interruzione o eccezione l'hardware commuta il modo di sistema.



- Le *istruzioni privilegiate* possono essere date solo in modo di sistema
- L'utente, per richiedere un servizio al Sistema Operativo, utilizza una *chiamata di funzione del sistema operativo*, detta anche *chiamata del sistema* (system call),
  - gestita dal sistema tramite interrupt.

Gestione dei processi

- Un *processo di elaborazione* si può considerare come un "programma in esecuzione".
- Un programma di per se non è un processo:
  - un programma è un'entità passiva, come il contenuto di un file memorizzato su disco,
  - mentre un processo è un'entità attiva, con un contatore di programma.
- Un processo necessita di alcune risorse, tra cui tempo di CPU, memoria, accesso ai files e ai dispositivi di I/O.

Gestione dei processi (II)

- Il sistema operativo è responsabile delle seguenti attività connesse alla gestione dei processi:
  - Creazione e cancellazione dei processi utenti e di sistema.
  - Sospensione e ripristino dei processi.
  - Fornitura di meccanismi per:
    - Sincronizzazione dei processi
    - Comunicazione tra processi
    - Gestione delle situazioni di stallo (*deadlock*)

Gestione della memoria centrale

- La memoria è un vasto vettore di dimensioni che variano tra le centinaia di migliaia ed i miliardi di parole
- E' un "magazzino" di dati velocemente accessibili condivisi dalla CPU e da alcuni dispositivi di I/O.
- La memoria centrale contiene memorie "volatili", che perdono il loro contenuto in caso di mancanza di alimentazione.
- Il S.O. è responsabile delle seguenti attività connesse alla gestione della memoria centrale:
  - Tenere traccia di quali parti della memoria sono attualmente usate e da che cosa.
  - Decidere quali processi si debbano caricare nella memoria quando vi sia spazio disponibile.
  - Assegnare e revocare lo spazio di memoria secondo le necessità.

Gestione dei file

- Un file è una raccolta di informazioni correlate definite dal loro creatore.
- Comunemente i file rappresentano programmi (codice sorgente o oggetto) e dati.
- S.O. fornisce una visione logica uniforme del processo di registrazione delle informazioni:
  - astrae le caratteristiche fisiche dei dispositivi per definire una unità di memorizzazione, cioé il *file*.
- S.O. associa i file ai mezzi fisici e vi accede attraverso i dispositivi che li controllano.

Gestione dei file

- S.O. è responsabile delle seguenti attività connesse alla gestione dei file:
  - Creazione e cancellazione di file.
  - Creazione e cancellazione di directory.
  - Fornitura delle funzioni fondamentali per la gestione di file e directory.
  - Associazione dei file ai dispositivi di memoria secondaria.
  - Creazione di copie di riserva (backup) dei file su dispositivi di memorizzazione non volativi.

Gestione del sistema di I/O

- Uno tra gli scopi di un S.O. è nascondere all'utente le caratteristiche degli specifici dispositivi.
- Un *sottosistema di I/O* consiste delle parti seguenti:
  - Un componente di gestione della memoria comprendente
    - la gestione delle regioni della memoria riservate ai trasferimenti di I/O (*buffer*),
    - la gestione della cache
    - la gestione asincrona delle operazioni di I/O e dell'esecuzione di più processi *(spooling*).
  - Un'interfaccia generale per i driver dei dispositivi.
  - I driver per gli specifici dispositivi.

Gestione della memoria di massa

- Giacché la memoria centrale è volatile ed è troppo piccola per contenere tutti i dati e tutti i programmi permanentemente,
  - il calcolatore deve disporre di una memoria secondaria, non volatile, in ausilio alla memoria centrale.
- I dischi sono uno dei mezzi più usati per la memorizzazione secondaria, su di essi vengono memorizzati sia dati che programmi.
- S.O. è responsabile delle seguenti attività connesse alla gestione dei dischi:
  - Gestione dello spazio libero
  - Assegnazione dello spazio
  - Scheduling del disco

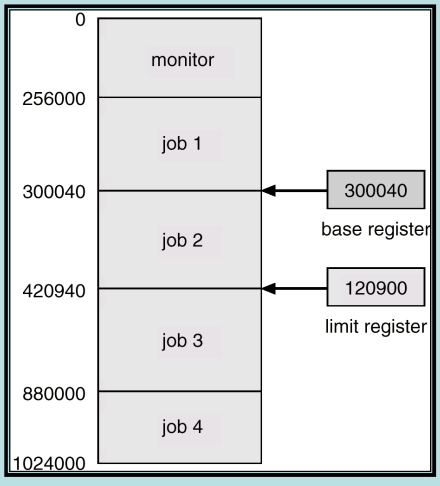
Protezione dell'I/O

- Tutte le istruzioni di I/O sono istruzioni privilegiate.
- E' necessario evitare che l'utente possa in qualche modo ottenere il controllo del calcolatore quando questo è in modo di sistema,
  - ad esempio un utente non deve poter modificare il vettore delle interruzioni.

Protezione della memoria (I)

- Bisogna proteggere il vettore delle interruzioni e anche le procedure di servizio dei segnali di interruzione.
- Per separare lo spazio di memoria dei programmi serve la capacità di determinare l'intervallo di indirizzi cui il programma può accedere.
- Due registri determinano il range di indirizzi legali a cui un programma può accedere:
  - **Registro di base (base register)** – contiene il più basso indirizzo della memoria fisica al quale il programma può accedere
  - **Registro di limite (limit register)** – contiene la dimensione dell'intervallo.
- La memoria al di fuori dell'intervallo individuato dai due registri non deve essere accessibile al programma.

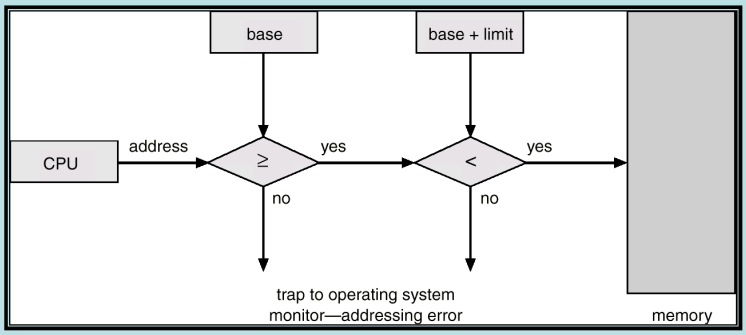
Uso di un registro di base e di un registro di limite



Protezione della memoria (II)

- Funzionando in modo di sistema S.O. può accedere sia alla memoria ad esso riservata sia a quella riservata agli utenti.
- Le istruzioni di caricamento dei registri di base e di limite devono essere istruzioni privilegiate.

Architettura di protezione degli indirizzi con registri di base e di limite



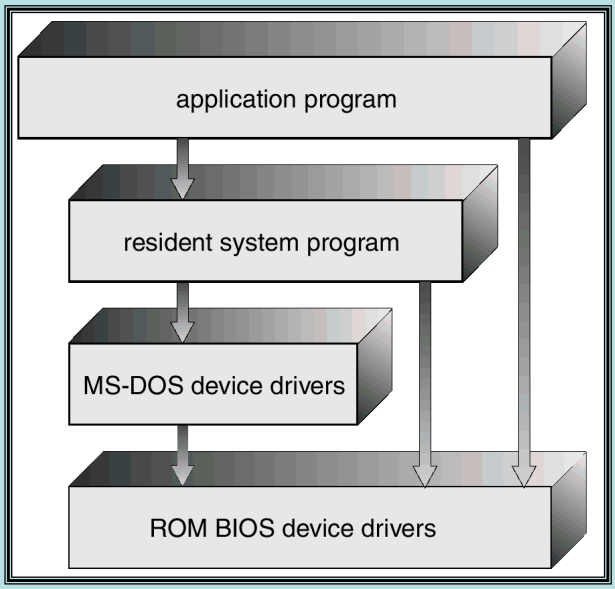
Protezione della CPU

- Occorre assicurare che S.O. mantenga il controllo dell'elaborazione,
  - cioé impedire che un programma utente entri in un ciclo infinito senza più restituire il controllo.
- *Temporizzatore* – interrompe l'esecuzione di un processo dopo un periodo predeterminato per assicurare che il S.O. mantenga il controllo della CPU.
  - Il temporizzatore è decrementato ad ogni ciclo di clock.
  - Quando il temporizzatore arriva a zero viene generato un interrupt.
- Il temporizzatore viene generalmente utilizzato soprattutto nei sistemi a partizione di tempo (time sharing).
- Può essere utilizzato anche per determinare l'ora corrente
- Il caricamento del temporizzatore è un'istruzione privilegiata.

Struttura del sistema

- Affinché possa funzionare correttamente ed essere facilmente modificabile un S.O. non viene in genere progettato come un sistema monolitico ma suddiviso in piccoli componenti.
- *Struttura semplice:*
  - Molti sistemi sono nati come sistemi piccoli e solo in un secondo tempo si sono accresciuti superando il loro scopo originale.
  - Ad es. MS-DOS, aveva come scopo il fornire la massima funzionalità nel minimo spazio.
    - Non è modulare
    - Nonostante la presenza di una struttura elementare le sue interfacce ed i livelli di funzionalità non sono ben separati

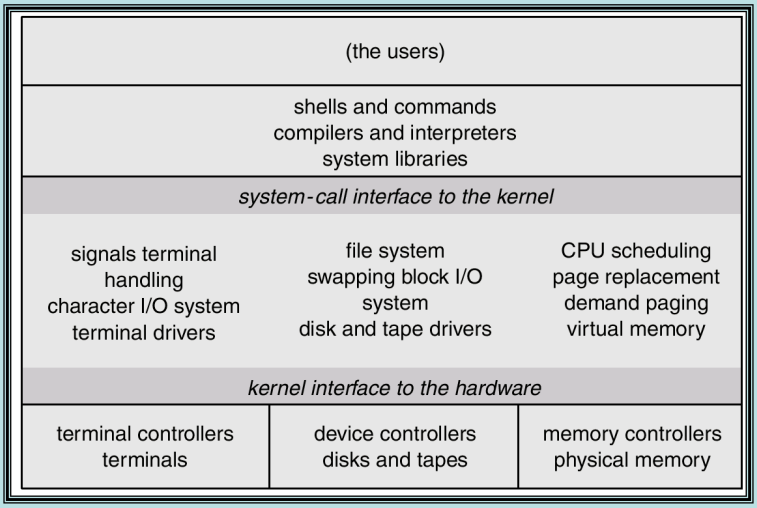
Struttura degli strati di MS-DOS



Struttura di sistema di UNIX

- A causa dei limiti delle architetture per cui era stato progettato, anche la strutturazione di UNIX non risultò completa.
- Il S.O. UNIX consiste di due parti separate:
  - Programmi di sistema
  - Kernel
    - Consiste di tutto ciò che nel diagramma a stati di un sistema è compreso tra l'hardware e l'interfaccia delle chiamate del sistema.
    - Fornisce il file system, lo scheduling della CPU, la gestione della memoria e altre (forse troppe) funzioni.
    - Difficile da migliorare: le modifiche in una parte possono avere effetto negativo in un'altra.

Struttura di sistema di UNIX (II)



Organizzazione a “strati”

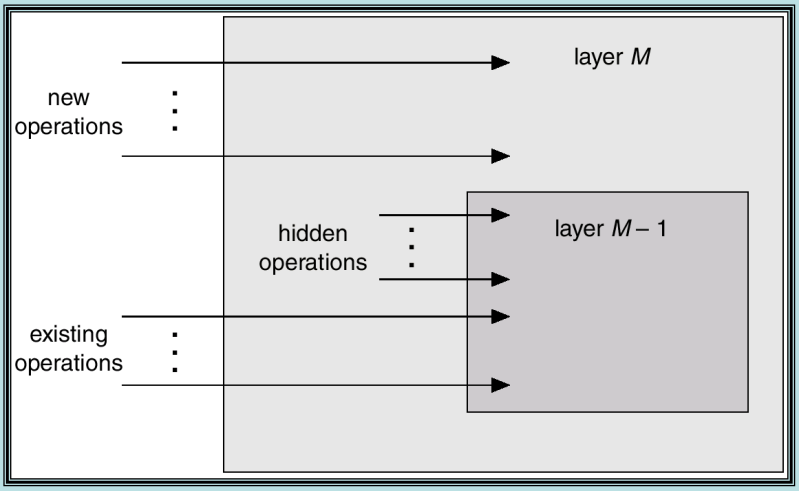
- Ogni macchina virtuale è un insieme di programmi che realizza delle funzionalità che utilizzano i servizi forniti a livello inferiore.
- Ogni macchina virtuale ha il compito di gestire risorse specifiche di sistema regolandone l’uso e mascherandone i limiti.
- I meccanismi che garantiscono la correttezza logica sono separati dalle politiche di gestione (maggiore flessibilità).

ogni “strato” risolve un problema specifico

Metodo stratificato

- In presenza di hardware appropriato si suddivide il S.O. in un certo numero di strati (livelli), ciascuno costruito sopra gli strati inferiori.
- Lo strato più basso (0) è lo strato fisico, quello più alto (*n*) è l'interfaccia utente.
- Gli strati sono composti in modo che ciascuno di essi usi solo funzioni o operazioni e servizi che appartengono a strati di livello inferiore.
- Ogni strato si realizza impiegando unicamente le operazioni messe a disposizione dagli strati inferiori, considerando soltanto le azioni che compiono senza entrare nel merito di come sono realizzate.
- Ogni strato nasconde a quelli superiori l'esistenza di determinate strutture dati, operazioni ed elementi fisici.

Uno strato di sistema operativo



Organizzazione di un SO

- Gerarchia di "macchine virtuali"
- La visione della macchina virtuale a livello n è quella fornita dall'HW e dagli strati del SO fino all'ennesimo (incluso)

Nucleo

- Interagisce direttamente con l'hardware
- Si occupa dell'esecuzione dei programmi e della risposta agli eventi esterni generati dalle unità periferiche.
- Scopo principale: gestire i processi corrispondenti ai programmi che sono contemporaneamente attivi.

- Fornisce alle macchine virtuali di livello superiore la visione di un insieme di unità di elaborazione virtuali ciascuna dedicata a un processo presente in memoria
- Gestisce il contesto di esecuzione dei vari processi
- Attua una politica di alternanza (*scheduling*) nell'accesso alla CPU da parte dei processi in esecuzione.

Gestore della memoria

- Controlla la memoria centrale, al fine di risolvere le relative esigenze dei vari processi in modo trasparente ed efficiente.
- Consente ai programmi di lavorare in un proprio *spazio di indirizzamento virtuale* e di ignorare quindi le effettive zone di memoria fisica occupata.
- Si occupa di:
  - proteggere programmi e relativi dati caricati nella memoria di lavoro;
  - mascherare la collocazione fisica dei dati;
  - permettere, in modo controllato, la parziale sovrapposizione degli spazi di memoria associati ai vari programmi.
- Fornisce alle macchine di livello superiore la possibilità di lavorare come se esse avessero a disposizione una memoria dedicata, di capacità anche maggiore di quella fisicamente disponibile.

Gestore delle periferiche

- Fornisce una visione del sistema in cui i processi possono operare mediante *periferiche astratte*.
- Maschera le caratteristiche fisiche delle periferiche e le specifiche operazioni di ingresso/uscita
- Ogni processo ha a disposizione delle periferiche virtuali

File System (gestore dei file)

- Gestisce la memoria di massa
- Gestisce i file

Interprete dei comandi

- Modulo del SO direttamente accessibile dall'utente
- Ha la funzione di interpretare i comandi che gli giungono (da tastiera e/o point&click) e di attivare i programmi corrispondenti.
- Le operazioni che svolge sono:
  - *lettura* dalla memoria di massa del programma da eseguire;
  - *allocazione* della memoria centrale;
  - *caricamento* del programma e dei relativi dati nella memoria allocata;
  - *creazione* e *attivazione* del processo corrispondente.

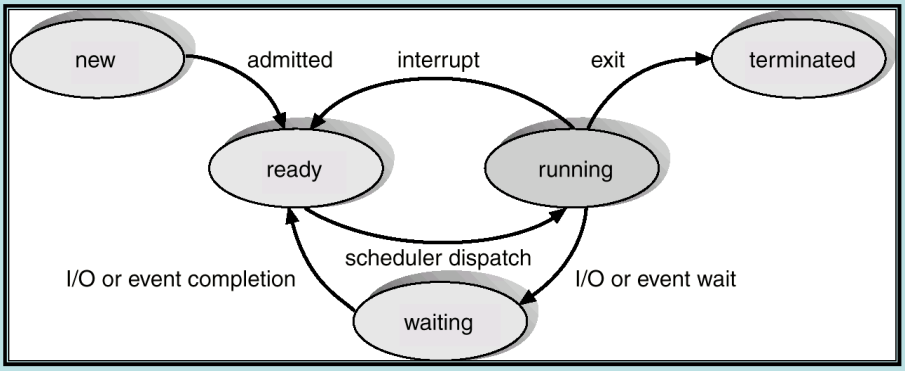
Concetto di processo

- Un S.O. esegue una varietà di “programmi”:
  - i sistemi a lotti (*batch*) eseguivano *lavori* (*job*)
  - un sistema a partizione di tempo esegue *programmi utente* (*task)*
- Spesso i termini *lavoro* e *processo* sono utilizzati in modo intercambiabile.
- Informalmente un processo può essere considerato come un programma in esecuzione.
- Un programma di per sé non è un processo.
- Il programma, anche detto *sezione testo*, è un’entità *passiva*, mentre il processo è un’entità *attiva*.
- Un processo include:
  - Contatore di programma (program counter)
  - Pila (stack)
  - Sezione di dati (data section)

Stato del processo

- L’esecuzione di un processo progredisce in maniera sequenziale.
- Mentre un processo è in esecuzione è soggetto a cambiamenti di *stato.*
- Ogni processo può trovarsi in uno tra i seguenti stati:
  - **Nuovo (new)**: Il processo viene creato.
  - **In esecuzione (running)**: quando è in memoria ed ha il controllo della CPU (esecuzione in modalità utente o supervisore);
  - **In attesa (waiting)**: quando è temporaneamente sospeso in attesa di un evento, quale la terminazione di I/O, lo scadere di un timer, la ricezione di un messaggio etc.;
  - **Pronto (ready)**: quando è in memoria e pronto per l' esecuzione, ma non ha il controllo della CPU e attende di essere assegnato ad un’unità di elaborazione.
  - **Terminato (terminated)**: quando termina e abbandona il sistema.

Diagramma di transizione degli stati di un processo



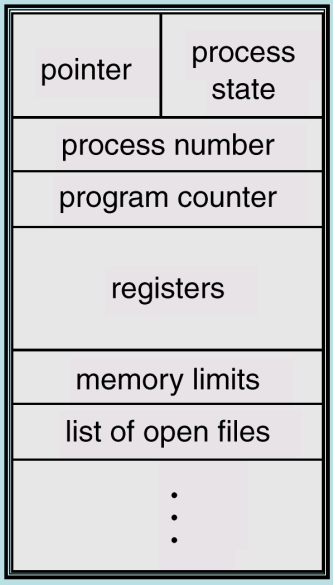
Blocco di controllo dei processi

- In un sistema operativo ogni processo è rappresentato da un descrittore di processo:
  - *process descriptor - PD* - detto anche blocco di controllo di un processo (*process control block - PCB*).
- Nel PCB sono contenute informazioni connesse ad uno specifico processo:
  - **Stato del processo:** tra new, ready, running, waiting, terminated.
  - **Contatore di programma:** indica l'indirizzo della successiva istruzione da eseguire.
  - **Registri di CPU:** accumulatori, registri indice, puntatori alla cima delle strutture a pila (stack pointer), registri d'uso generale e registri contenenti informazioni relative ai codici di condizione.
  - **Informazioni sullo scheduling di CPU:** priorità del processo, puntatori alle code di scheduling e altri parametri di schedulazione.

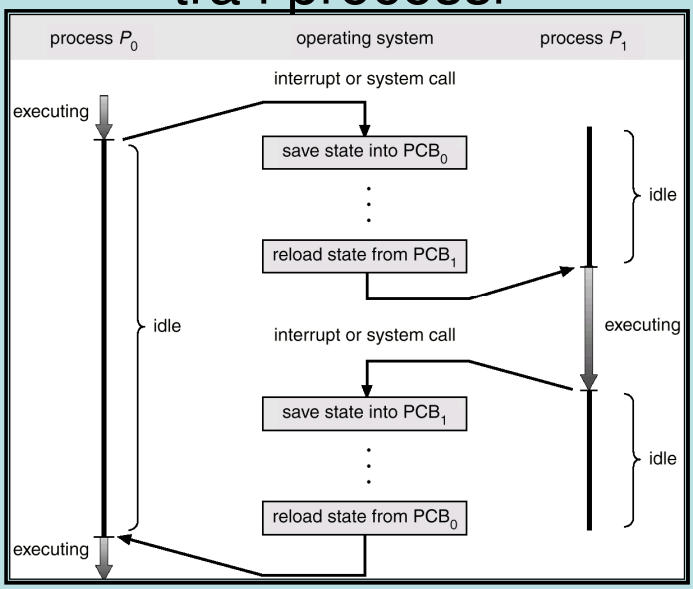
Descrittore di processo (II)

- **Informazioni sulla gestione della memoria:** registri di base e di limite, tabelle delle pagine in memoria o dei segmenti (a seconda della tecnica usata dal S.O.).
- **Informazioni di contabilizzazione delle risorse:** tempo di CPU, tempo reale di CPU, numero del processo, etc..
- **Informazioni di I/O:** lista dei dispositivi di I/O assegnati al processo, elenco file aperti, etc..

Descrittore di processo (PCB)



La CPU può essere commutata tra i processi



"Concorrenza" fra processi

- Vantaggi dell'esecuzione concorrente di più processi:
  - impiegare in maniera trasparente una o più CPU (sia inserite in un solo calcolatore che in più calcolatori, collegati in rete);
  - aumentare l'utilizzo della CPU nei sistemi a partizione di tempo, ove si eseguono più lavori in quasi parallelismo;
  - condividere la stessa risorsa fisica fra diversi utenti in modo del tutto trasparente ma controllato;
  - accedere contemporaneamente, da parte di diversi utenti, a una base di dati comune e centralizzata;
  - …

“Concorrenza” fra processi (II)

- Problemi
  - **starvation**: un processo non riesce ad accedere ad una risorsa perché la trova sempre occupata da altri processi (che per esempio possono avere un livello di priorità maggiore);
  - **blocco critico**: un insieme di processi rimane bloccato perché ciascuno di essi aspetta delle risorse che sono occupate da un altro processo compreso in questo stesso insieme (**vincolo circolare**).
  - Evitare (prevenzione) o risolvere (eliminazione) situazioni di blocco critico o di starvation riduce le prestazioni complessive del sistema.

Interazioni tra processi

- Le interazioni fra processi sono classificabili in:
  - **indesiderate e (spesso) impreviste**
  - **desiderate e previste**.
- La gestione delle interazioni fra i processi implica
  - la **sincronizzazione** fra le varie attività che ogni singolo processo deve svolgere in modo parallelo rispetto agli altri
  - la **comunicazione**, ovvero una modalità per lo scambio dei dati fra i processi
- Modalità di funzionamento dei processi:
  - **in foreground**, quando il processo è abilitato all’interazione con l’utente;
  - **in background**, quando il processo non è in grado, almeno temporaneamente, di interagire direttamente con l’utente; questo è lo stato in cui si trovano parecchi dei processi relativi alle funzioni interne del sistema operativo

Processi cooperanti

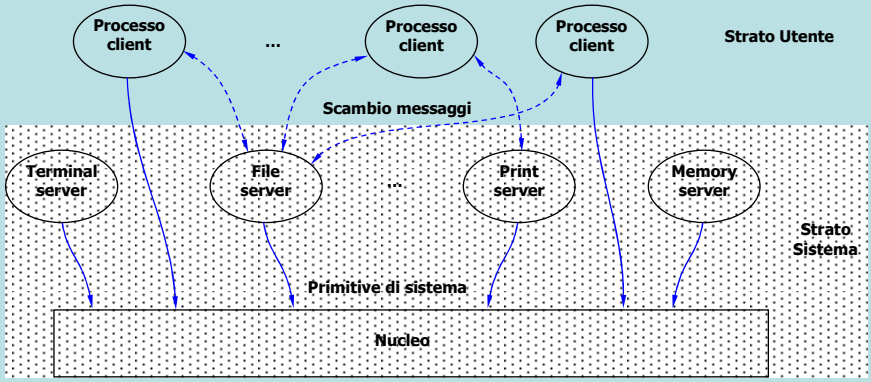
- I processi concorrenti nel sistema operativo possono essere *indipendenti* o *cooperanti*.
- Un processo è *indipendente* se non può influire su altri processi, né può subirne l'influsso, durante la sua esecuzione.
- Un processo *cooperante* può influire su altri processi, e può subirne l'influsso, durante la sua esecuzione.

Processi cooperanti (II)

- Vantaggi della cooperazione tra processi:
  - Condivisione di informazioni:
    - più utenti possono essere interessati alle stesse informazioni.
  - Accelerazione del calcolo:
    - se vi sono più CPU o canali di I/O.
  - Modularità:
    - può essere più pratica ed efficiente la costruzione di un sistema modulare.
  - Convenienza:
    - anche un solo utente può avere la necessità di compiere più operazioni contemporaneamente.
- I modelli fondamentali della comunicazione tra processi sono due:
  - a memoria condivisa;
  - a scambio di messaggi.

Organizzazione client-server

- Obiettivo: minimizzare le dimensioni del nucleo
  - si spostano alcune componenti del sistema verso gli **strati applicativi**
  - le funzionalità estranee al nucleo sono "servizi" forniti da **processi server**.



Micronucleo

- Il kernel è il nucleo di un sistema operativo:
  - il software che ha il compito di fornire ai processi in esecuzione un accesso sicuro e controllato all'hardware.
- Il kernel ha anche la responsabilità
  - di assegnare una porzione di tempo-macchina,
  - di permettere l'accesso alle risorse hardware a ciascun programma.
- Nell'approccio di progettazione del S.O. basato su microkernel, si cerca di spostare i servizi dal kernel al livello dei programmi utente.
- In generale un micronucleo offre i servizi minimi di gestione dei processi, della memoria e di comunicazione.

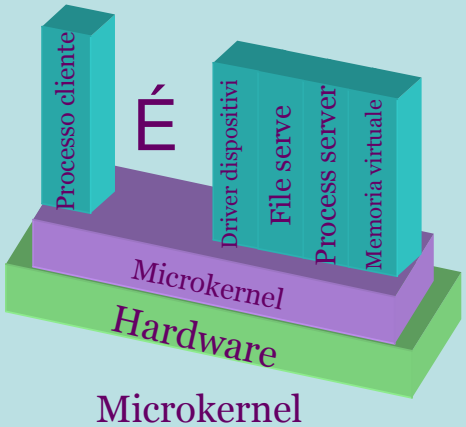
Micronucleo (II)

- La “filosofia” del microkernel:
  - Solo le funzioni assolutamente essenziali del nucleo del sistema operativo dovrebbero essere nel kernel;
  - I sevizi meno essenziali e le applicazioni sono costruiti sopra il microkernel e vengono eseguiti in modalità utente.
- La linea di separazione fra cosa è dentro e cosa è fuori dal microkernel varia da un progetto all’altro.
- La caratteristica comune è che molti servizi che facevano parte del S.O. diventano sottosistemi esterni che interagiscono con il kernel e tra di loro,
  - Ad es. driver dei dispositivi, file system, gestore della memoria virtuale, sistema a finestre, servizi di sicurezza, etc..
- L’architettura a microkernel sostituisce la tradizionale stratificazione verticale dei SO con una orizzontale.

Micronucleo (III)

- I componenti del SO esterni al microkernel sono implementati come processi server:
  - interagiscono fra di loro su una base di parità, tramite passaggio di messaggi attraverso il microkernel.
- Il microkernel gestisce lo scambio dei messaggi.
- Scopo principale del micronucleo è fornire funzioni di comunicazione tra i programmi client ed i vari servizi,
  - anch’essi eseguiti nello spazio utente, tramite scambio di messaggi.
- Vantaggi:
  - facilità di estensione del sistema (che va fatta nello spazio utente)
  - S.O. può essere semplicemente adattato a nuove architetture
  - Più affidabile (meno codice) e sicuro

Microkernel



Microkernel

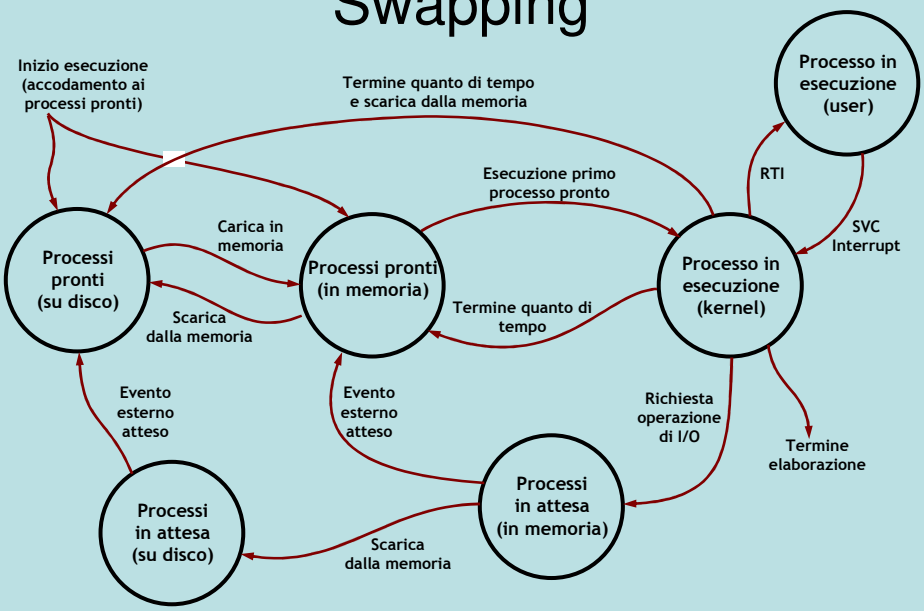
Gestore della memoria

- Applica tecniche per gestire il conflitto fra dimensione della memoria fisica e spazio complessivo richiesto dai programmi che devono essere eseguiti in modo concorrente e dai relativi dati.
- Combina le seguenti strategie:
  - consentire il caricamento di un programma a partire da un indirizzo qualunque della memoria;
  - ridurre la necessità di spazio tenendo in memoria solo una porzione dei programmi e dei dati;
  - condividere parte delle istruzioni (codice eseguibile) fra diversi processi corrispondenti a uno stesso programma.
- Il gestore della memoria
  - garantisce ai vari processi uno **spazio di indirizzamento virtuale** in cui lavorare, che può essere superiore alla memoria fisica presente nel calcolatore
  - mette in atto dei meccanismi di protezione che tutelano la privatezza dello spazio di lavoro assegnato a ogni processo.

La rilocabilità del codice

- Durante la compilazione i nomi simbolici e i riferimenti a celle di memoria sono stati risolti:
- Due spazi di memoria
  - **spazio logico**;
  - **spazio fisico**.
- Per far funzionare il programma caricandolo a partire da una posizione arbitraria della memoria bisogna effettuare una **rilocazione**: sommare a tutti gli indirizzi presenti nel programma un valore (**spiazzamento**) corrispondente alla differenza fra l'indirizzo a partire dal quale verrà effettivamente caricato il programma e il valore a partire dal quale sono stati calcolati gli indirizzi.

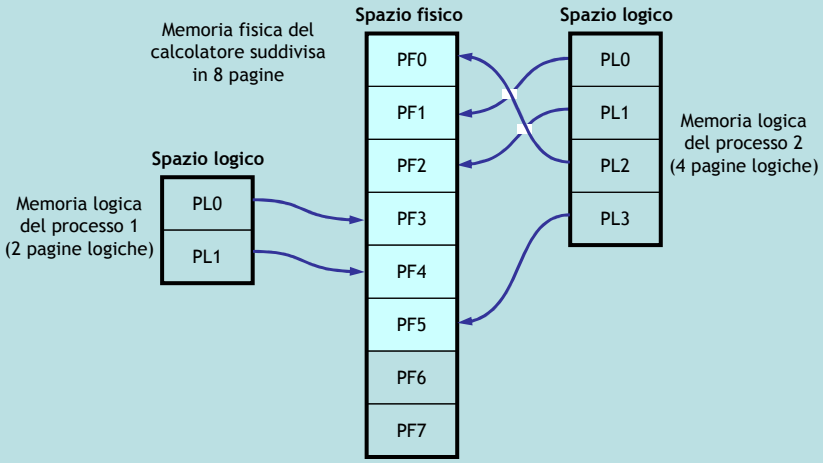
Swapping



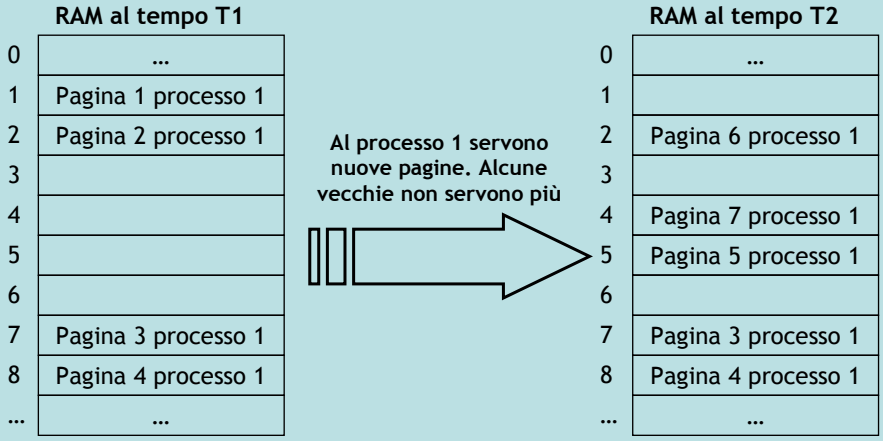
Paginazione

- Frammentazione della memoria (logica e fisica) in blocchi di dimensioni prefissate: le pagine**.**
- Lo spazio logico di indirizzamento del processo è suddiviso in sezioni, di dimensioni fisse e uguali fra loro, dette **pagine logiche**
- Lo spazio fisico di indirizzamento disponibile nel calcolatore è anch'esso suddiviso in **pagine fisiche**, della stessa dimensione delle pagine logiche.
- Si basa sul principio di località spazio-temporale
- Meccanismo: Vengono caricate, in alcune pagine fisiche su RAM, solo alcune pagine logiche del codice in esecuzione. Le pagine logiche necessarie vengono caricate di volta in volta, in base all'esigenza.

Corrispondenza tra pagine logiche contigue e pagine fisiche non contigue

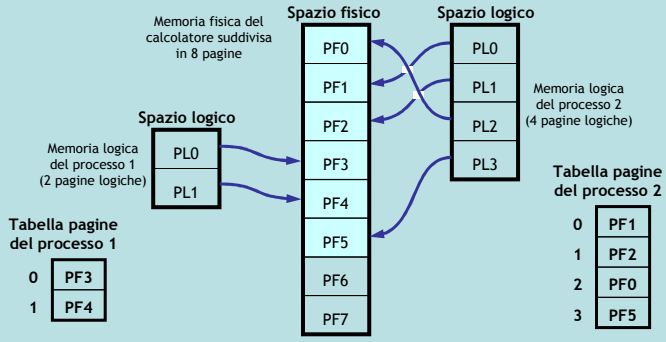


Paginazione



Memory Management Unit - MMU

- Serve un dispositivo dispositivo hardware aggiuntivo in grado di convertire gli indirizzi logici cui fa riferimento il programma nei corrispondenti indirizzi fisici: **Memory Management Unit.**
- La MMU utilizza una **tabella delle pagine**:
  - mantiene la relazione tra ogni pagina logica e l'indirizzo della pagina fisica corrispondente.



Paginazione

- La paginazione risolve contemporaneamente tre problemi:

1. Dove mettere il processo in memoria
2. Superare il numero di processi che posso gestire contemporaneamente
3. Superare la dimensione fisica della memoria di lavoro

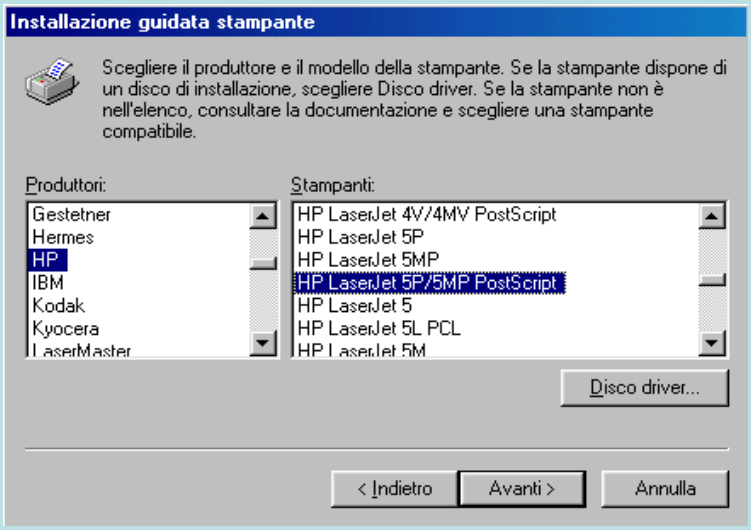
Gestore delle periferiche

- Comunicazione tra l'ambiente CPU-RAM ed i dispositivi esterni.
- Mascherare ai processi l'esistenza di un numero limitato di risorse.
- Mascherare ai processi la differenza tra risorse dello stesso tipo (o di tipo simile)

Gestione periferiche I/O

- Comandi ad alto livello per accedere alle periferiche che usano meccanismi quali:
  - **i controller**,
  - **i driver**.
- I sistemi operativi comprendono i driver per la gestione delle periferiche più comuni.
- Ogni aggiunta o modifica alla configurazione standard comporta l'installazione di software addizionale (driver aggiuntivi).

Installazione driver



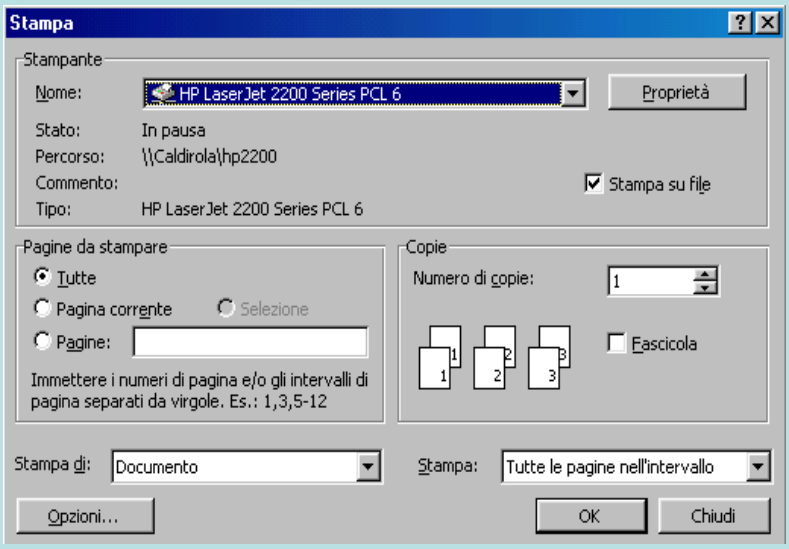
Plug&Play

- I sistemi operativi più recenti sono dotati di funzioni di Plug&Play (PnP) che permettono la configurazione automatica dei driver.
- Un sistema PnP consente di aggiungere (plug) nuove periferiche al sistema che possono essere utilizzate (play), senza necessità di intervento da parte dell'utente per la selezione e l'installazione dei driver.

Spooling

- I driver servono anche a virtualizzare la presenza di più periferiche intrinsecamente non condivisibili, tramite la tecnica di **spooling**.
- Esempio: gestione di una stampante
  - quando un processo desidera stampare un file, lo invia al driver,
  - il driver lo accoda in un'opportuna directory di spooling,
  - i file contenuti nella directory di spooling vengono stampati in ordine di arrivo (a meno che siano stabilite delle politiche di gestione delle priorità);
  - quando la directory di spooling si svuota il driver rimane in memoria in attesa che un processo cerchi di stampare.
- Questa soluzione
  - consente di disaccoppiare il programma che deve stampare e la periferica
  - rende possibile l'uso della stampante da parte di molti processi senza attese inutili.

Gestione stampe



Gestione memoria di massa (file system)

- Obiettivo:
  presentare all'utente l'organizzazione logica dei dati e le operazioni che è possibile compiere su di essi.
- Operazioni di base di un file system:
  - **recupero** di dati precedentemente memorizzati;
  - **eliminazione** (**cancellazione**) di dati obsoleti;
  - **modifica/aggiornamento** di dati preesistenti;
  - **copia** di dati (e.g. da HD a FD) per backup o per il trasferimento;
  - …
- I servizi vengono forniti sia ai **programmi applicativi** che direttamente agli **utenti**.

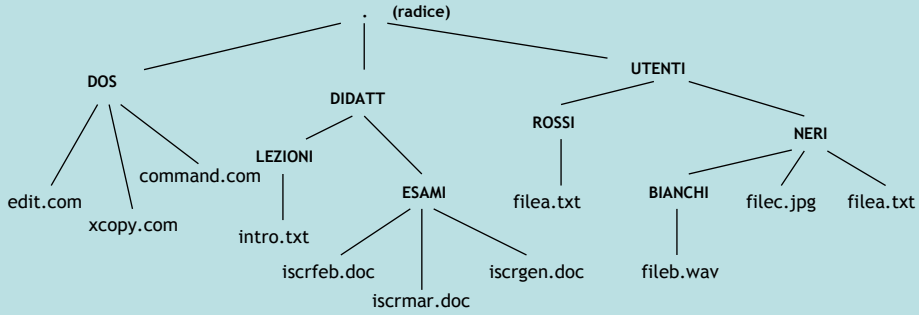
File system

- FILE:
  - contenitore logico di informazioni (dati o istruzioni);
  - oggetto a “lunga vita”, per conservare le informazioni anche dopo la terminazione del processo che lo ha generato.
- Per ogni file:
  - Identificatore (**nome.estensione**)
  - Periferica (**drive**) e percorso sulla periferica
  - Data creazione
  - Dimensione
  - Posizione effettiva dei dati nella memoria di massa
  - Altre informazioni
    - applicazione che consente all’utente di “usare” il file
    - data di ultima modifica
    - diritti di accesso al contenuto del file
    - …

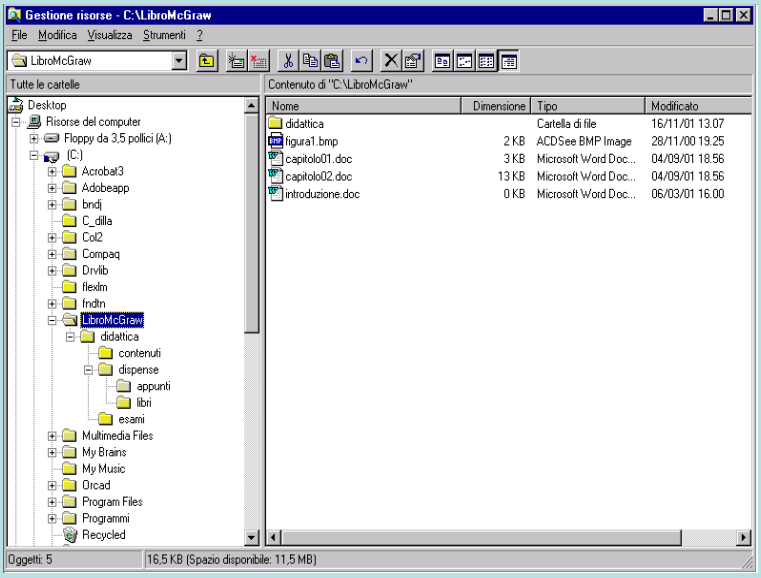
File

- I nomi dei file sono in genere composti da due parti:
  - nome (vero e proprio), che viene assegnato dall’utente
  - estensione, associata al programma che ha generato il file e consente quindi di identificare la tipologia dei dati contenuti nel file
- Ogni sistema operativo pone dei vincoli sulla lunghezza dei filename e sui caratteri di cui possono essere costituiti
  - MS-DOS imponeva una lunghezza massima di 8+3 caratteri per nomi ed estensioni
  - Windows ha un limite di 254 caratteri (compreso il path)
- I file sono generalmente organizzati in cartelle (directory) e sottocartelle in una gerarchia ad albero (o, al limite, a grafo aciclico).

Un esempio di struttura



Interfaccia grafica



Organizzazione fisica dei dati

- Come mantenere la corrispondenza tra il nome del file e i blocchi su disco che ne contengono i dati:
  - **lista concatenata** (e.g. Windows 95/98)
  - **i-node** (e.g. UNIX)

Il controllo degli accessi

- Identificazione degli accessi al sistema.
  - Associare a ogni utente un **account** (**login**) e una parola d'ordine (**password**).
  - All'interno del sistema operativo, in un apposito file, è contenuta la lista di tutti gli account e delle relative password: solo se viene specificato un account fra quelli previsti (utente abilitato) e la password corrisponde a quella memorizzata (certificazione di identità) viene consentito l'accesso al sistema.
- Questo consente di **personalizzare** il sistema, per esempio definendo:
  - la distribuzione dei costi di gestione fra i vari utenti;
  - la visibilità del sistema in termini di porzione del file system complessivo, periferiche e programmi applicativi disponibili;
  - la personalizzazione dell'ambiente operativo.
- Consente di controllare gli accessi ai file:
  - livello di **protezione** a livello di file o di directory;
  - altro metodo: **Access Control List**

Virtualizzazione delle risorse di rete

- Estendere anche a processi in esecuzione su calcolatori diversi il principio di virtualizzazione delle risorse.
- Condividere in modo trasparente dati, periferiche e unità di elaborazione.
- In particolare si tratta di poter gestire le *periferiche* e il *file system*.

File system di rete

- Un sistema operativo che consente una gestione distribuita del file system deve:
  - integrare in modo organico i singoli file system dei calcolatori della rete;
  - risolvere i problemi dell'univocità dei nomi di file e directory per i calcolatori della rete;
  - consentire un accesso efficiente anche a file presenti su calcolatori remoti.
- Questi requisiti vengono soddisfatti con un file system di tipo client-server.

Organizzazione client-server

- I client possono usufruire dei servizi di sistema inviando una richiesta al server
- Ottime proprietà di modularità e portabilità: