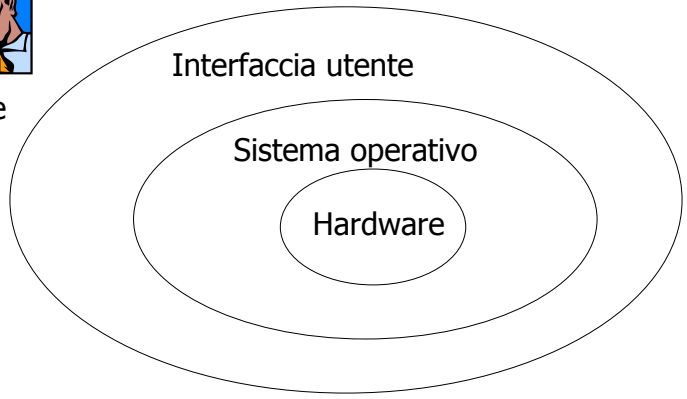
Il Sistema Operativo

Sistema Operativo (Software di base)

- Il ***sistema operativo*** è un insieme di programmi che opera sul *livello macchina* e offre funzionalità di *alto livello*
  - Es.organizzazione dei dati attraverso indici
- I sistemi operativi sono organizzati a strati
- Strato = macchina virtuale che maschera la macchina fisica (hardware)

Sistema operativo

utente



Funzionalità

- Possiamo identificare:
  - Gestione processi
  - Gestione memoria primaria
  - Gestione memoria secondaria + file system
  - Gestione dell'I/O (periferiche)
  - Interprete dei comandi
- La porzione del S.O. che contiene le funzionalità di base è detta *nucleo* (o *kernel*)

Gestore dei processi

- Gestisce l'esecuzione dei programmi
- Le unità di esecuzione vengono chiamate processi
- Per eseguire un programma occorre mantenere il corrispondente processo attivo nella CPU
- Multi-tasking: più attività (processi) procedono concorrentemente, anche usando una sola CPU
  - Il gestore deve decidere a quale processo assegnare la CPU
  - Inoltre deve reagire agli eventi esterni (le interruzioni provenienti dalle periferiche)

Cos'è un processo?

- *Programma* = lista di istruzioni = nozione statica
- *Processo* = programma in esecuzione
  = programma + stato corrente variabili
  = nozione dinamica
- Stato corrente=
  - valori in memoria centrale
  - valori nei registri della CPU
- In un PC **con una sola CPU**:
  un solo processo in esecuzione alla volta

Gestore della memoria

- Il gestore della memoria deve
  - allocare la memoria
  - partizionarla tra i processi che la richiedono
- Grazie al gestore della memoria gli strati superiori hanno l'illusione che ogni processo abbia una memoria dedicata

Gestore delle periferiche

- Maschera le caratteristiche hardware delle periferiche
- Gestisce le operazioni di input e output
- Fornisce procedure ad alto livello
  - ad esempio per la lettura, scrittura di dati su memorie secondarie
  - scrittura su stampanti, ecc

File System

- Gestisce i dati in memoria di massa
- Struttura i dati in modo gerarchico utilizzando *file* e *directory*
- Fornisce operazioni di alto livello per la gestione di file
  - ad esempio creazione di un nuovo documento, directory ecc
- *Protegge* i dati da accessi esterni
- Garantisce la *condivisione sicura* dei dati

Interprete dei comandi

- Consente all'utente di attivare i programmi
- Sfrutta le funzionalità degli strati inferiori per
  - cercare in memoria il programma invocato
  - allocare la memoria richiesta dal programma
  - attivare un processo per eseguire il programma
- Es: consolle DOS ("Command Prompt") sui sistemi MS Windows

Evoluzione dei Sistemi Operativi

- Sistemi batch con schede (50's-60's)
- System/360 IBM compatibili (65-70's)
- Sistemi operativi UNIX e DOS (80's)
- WINDOWS (90's)

Uno sguardo da vicino

- Gestione dei Processi
- Gestione della Memoria Centrale
- File System
- Gestione delle Periferiche

Ciclo di vita dei processi (I)

- Processo *utente:* deriva da un programma applicativo
- Processo di *sistema:* deriva da un programma del sistema operativo
  - Processi kernel (nucleo)
  - Gestori interruzioni
- L'esecuzione di un programma può comportare l'alternarsi di processi *utente* e di *sistema* all'interno della CPU
- L'esecuzione di un processo può essere interrotta!

Ricordate sempre che un processo corrisponde ad un programma + stato corrente della memoria centrale e dei registri

Ciclo di vita dei processi (II)

- Il ciclo di vita di un processo può essere modellato in modo semplice usando:
  - Stati, che rappresentano una condizione operativa in cui si trova il processo
  - Transizioni tra stati, ovvero schematizzazioni di eventi che determinano il passaggio di un processo da uno stato a un altro
- Stati significativi:
  - EXE: il processo si trova in esecuzione sulla CPU
  - READY: il processo è in condizione di essere *immediatamente* eseguito, ma non ha a disposizione la CPU
  - WAIT: il processo è bloccato in attesa che si verifichi un evento che gli permetta di procedere nella sua attività

Scheduling dei processi

- In un dato istante, un solo processo può essere in esecuzione sulla CPU (ovvero, in stato "exe")
- Un S.O. multitasking può interrompere i processi per assicurare una politica equa (fair) di esecuzione
- Scheduler = quella parte del sistema operativo che *seleziona* il processo da mandare in esecuzione tra quelli pronti (che si trovano in stato "ready")
- Due delle possibili politiche di *scheduling*:
  - Round robin
  - Con Priorità (utilizzata con molte varianti)
- Criteri di valutazione: efficienza, tempi di risposta, fairness, ecc.

Processi: stati e scheduling

- Avvicendarsi di due processi sulla CPU: "Context switch" (commutazione di contesto)
- Quando ad un processo viene revocata la CPU, occorre salvare in memoria le sue informazioni di stato
- Quando ad un processo viene nuovamente assegnata la CPU, occorre ripristinare il suo stato

Round Robin

- Gestione dei processi in attesa tramite *coda*
  - Cioè FIFO = *first-in first-out*
- Ad ogni processo viene assegnato un quanto di tempo di esecuzione ("timeslice") dopo il quale torna in attesa
  - Timeslice >> Tempo per salvare/ripristinare stato (*context switching*)
  - Timeslice << Tempo di esecuzione del programma (*per assicurare fairness*)

Priorità

- Si assegna una priorità ad ogni processo, e si manda in esecuzione uno dei processi con priorità *più alta*
  - In genere, tra i processi che hanno la stessa priorità, si adotta una politica di tipo round-robin
- La priorità può essere assegnata
  - *staticamente:*
    - se la priorità di ciascun processo non viene mai modificata, si ha il fenomeno della *starvation* (ovvero i processi con priorità più bassa aspettano indefinitamente di poter accedere alla CPU)
  - *dinamicamente:*
    - a seconda delle operazioni effettuate dal processo (ad es. seleziona subito processi con operazioni I/O)

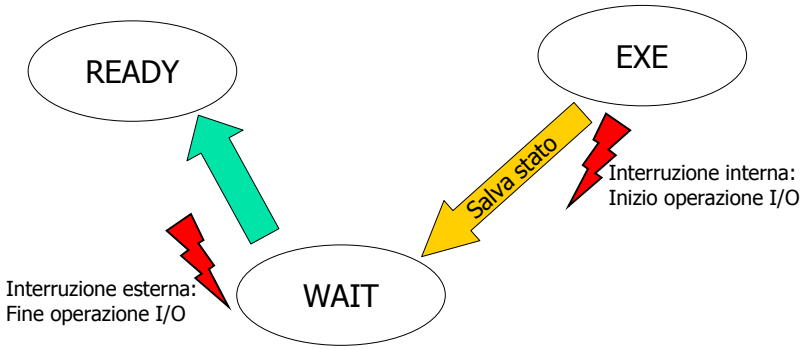
Gestire le attese: interruzione interna

- Per migliorare l'utilizzo della CPU, occorre trattare opportunamente i casi in cui un processo deve rimanere inattivo (p.es. in attesa di operazioni di input)
- L'esecuzione di un processo *attivo* (in stato "exe") si interrompe "volontariamente", ad es. per operazioni di *input/output*
- Lo stato corrente (contenuto registri ecc) del processo interrotto viene salvato in memoria
- Il processo passa allo stato "speciale" di *attesa* (o "*wait*"), e non in stato "ready", dato che non deve partecipare alla competizione per la CPU
- Il controllo passa ad un processo di sistema che assegna la CPU ad un altro processo

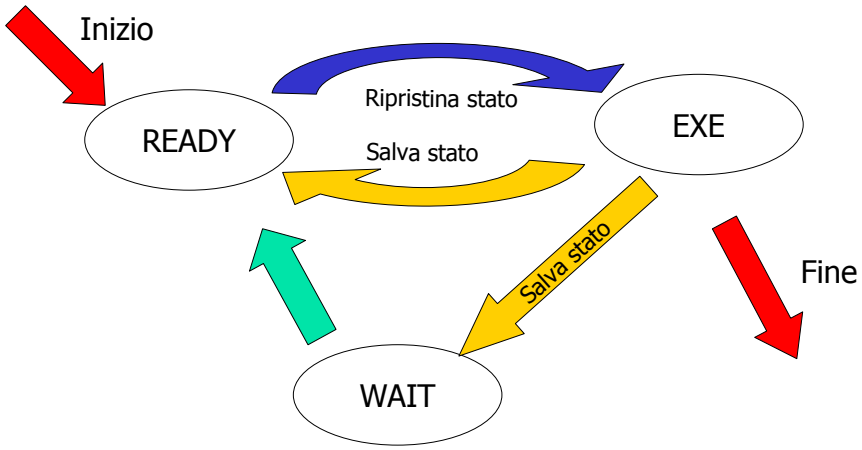
Gestire le attese: Interruzioni esterne

- Una periferica segnala la fine di un'operazione di input/output: questo significa che il processo in attesa su tale operazione adesso può potenzialmente procedere nella sua esecuzione.
- Il processo correntemente in esecuzione viene *interrotto*
- Il gestore delle interruzioni provvede a trasferire dati in memoria e risvegliare il processo in *attesa*, che passa dunque allo stato *"ready"*
  - Il gestore lavora con interruzioni disabilitate
- Il controllo passa poi al nucleo che manda in esecuzione uno dei processi in stato *ready*

Gestire le attese: Diagramma di stato



Ciclo di vita dei processi: Diagramma di stato



Gestione Memoria Primaria

- I processi si alternano nell'accesso alla CPU
- I S.O. mantengono più programmi in memoria primaria;
  - ad ogni processo viene *allocata* una porzione della memoria primaria
  - Ciò comporta il *partizionamento* della memoria primaria e del suo spazio di indirizzi.
- Il succedersi dei processi può determinare la *frammentazione* della memoria primaria:
  - la memoria disponibile risulta suddivisa in molti blocchetti, ognuno dei quali troppo piccolo per essere sfruttato
- Alcune tecniche di gestione risolvono anche la frammentazione:
  - P.es. la "paginazione"

Il fenomeno dello Swapping

- Per permettere l'esecuzione di un numero di processi che richiedono *una quantità di memoria superiore a quella contenuta nella memoria centrale*, il S.O. può far uso della memoria di massa per salvare porzioni di memoria non utilizzate dal processo attivo sulla CPU
- Quanto detto implica un continuo trasferimento di dati tra RAM e memoria di massa (e viceversa): questa azione è nota come "swapping".
- Uno swapping molto intenso (qualora si voglia utilizzare una quantità di memoria molto superiore a quella fisicamente presente) degrada significativamente le prestazioni di un calcolatore.

Gestione della Memoria Secondaria

Il File System

Gestione della Memoria Secondaria

- La memoria di massa serve per gestire grandi quantità di dati
  - Persistentenza
  - Sicurezza
  - Classificazione
- Soluzione
  - I dati vengono organizzati **logicamente** in file e gestiti dal sistema operativo

File

- Nome:
  - Identifica il file spesso con una estensione che indica il tipo di file
  - es. Tesi.doc oppure somma.exe
- Struttura:
  - Sequenza di byte
  - Sequenza di blocchi (record) di byte
- Tipo:
  - File di caratteri e binari (eseguibili)
  - Directory
- Attributi:
  - nome, diritti di accesso,proprietario

Operazioni su File

- Il *file system* consente di effettuare le seguenti operazioni:
  - creare, cancellare, spostare, recuperare, modificare documenti in memoria di massa (memoria persistente)
  - Modificare gli attributi di un file
  - Ridenominare i file

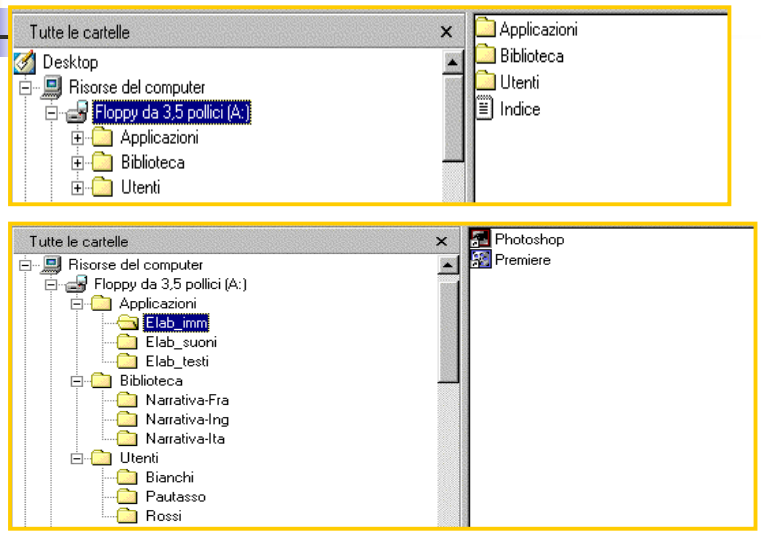
Organizzare i file: la Directory (o "folder", cartella)

- E' un file di tipo speciale che mantiene informazioni su altri file
  - permette di strutturare insiemi di file (dati) in maniera gerarchica
  - contiene la lista dei nomi e attributi dei file e delle directory al suo interno
- Quindi: il *file system* ha una struttura ad *albero*
  - Radice = radice dell'intero file system
  - Nodi interni = directory
  - Foglie = directory o file (documenti/programmi)

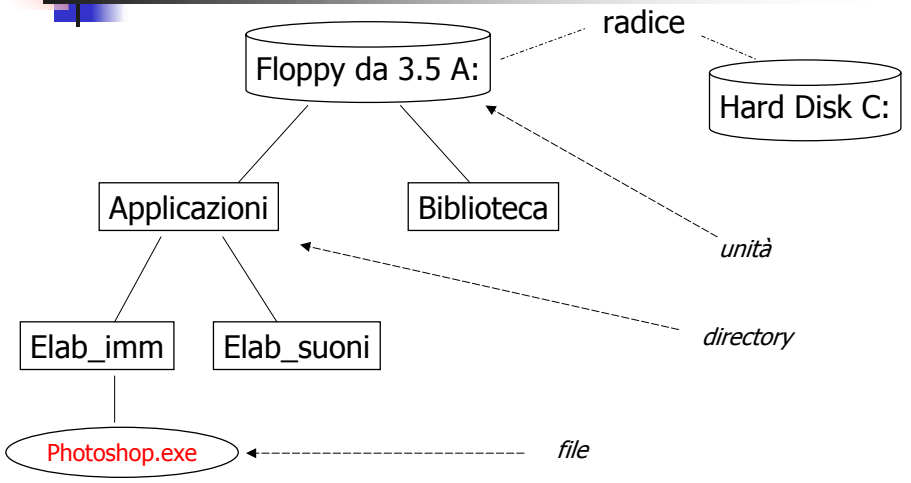
File System in DOS/Windows

- Esistono directory predefinite che corrispondono alle unità di memoria secondaria (dette anche "unità")
  - Disco fisso indicato con C:
  - Dischetto indicato con A:
- La memoria secondaria di un singolo dispositivo può essere divisa in più partizioni, ognuna delle quali è individuata come un'unità distinta

Struttura ad albero



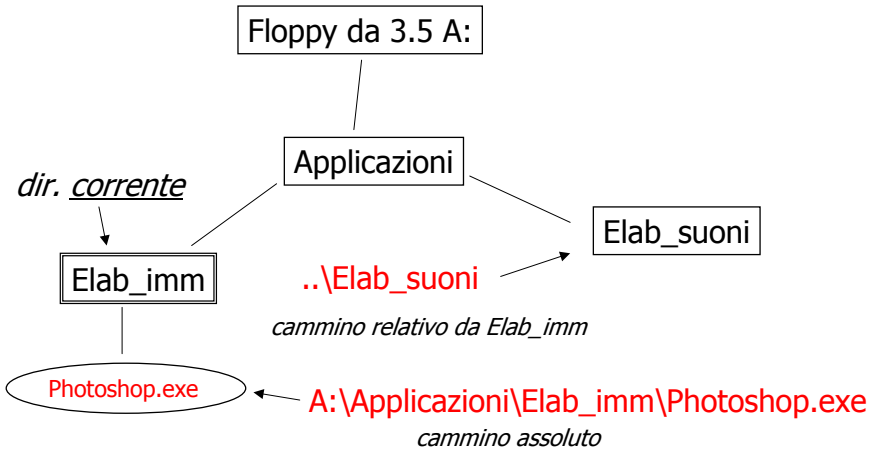
Struttura ad albero



Path names

- In un'organizzazione ad albero i nomi devono contenere informazioni sui cammini sui quali si trovano i corrispondenti file
  - Nomi relativi:
    - relativi ad una particolare directory
  - Nomi assoluti:
    - specificano il cammino a partire dalla radice (root) del file system (mai ambigui)
- Nei path names si possono utilizzare I due simboli speciali
  - Il punto (.) rappresenta la directory corrente
  - I due punti (..) rappresentano la directory immediatamente superiore nel'albero

Esempio cammini



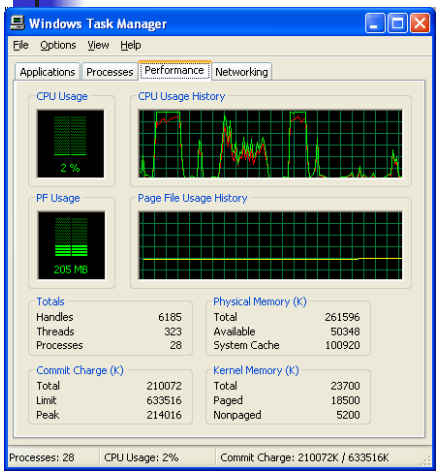
Gestione File System

- File possono venire memorizzati con
  - Allocazione continua di blocchi di byte
  - Allocazione collegata (un blocco contiene l'indirizzo del prossimo)
  - Allocazione con indice (si mantiene una tabella in memoria per recuperare i blocchi) – DOS -
  - I-node: tabella con puntatori ad altre tabelle (combina le ultime due) – UNIX –

Gestione delle periferiche

- Driver fisici (hardware)
  - per trasferire e manipolare dati (stampante, lettore floppy, ecc)
- Driver logici (software)
  - parte del sistema operativo che fornisce funzionalità ad alto livello che riguardano le periferiche

Il Task Manager di Windows



- Si può avere la percezione di cosa avviene a livello di S.O. utilizzando strumenti per il monitoraggio di sistema, come p.es. Task Manager dei sistemi Windows (famiglia NT)