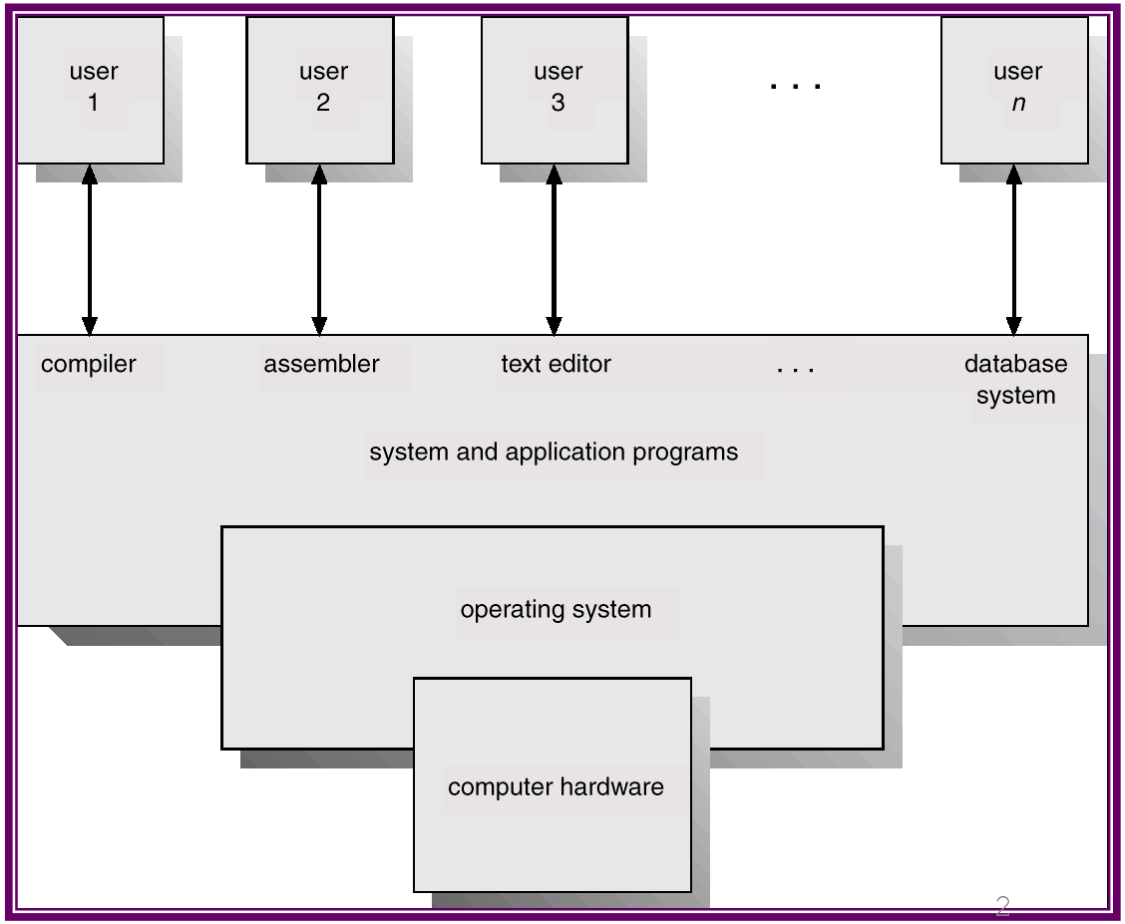
UNIVERSITÀ DI PARMA
il mondo che ti aspetta
DIPARTIMENTO DI INGEGNERIA E ARCHITETTURA

Introduzione ai Sistemi Operativi

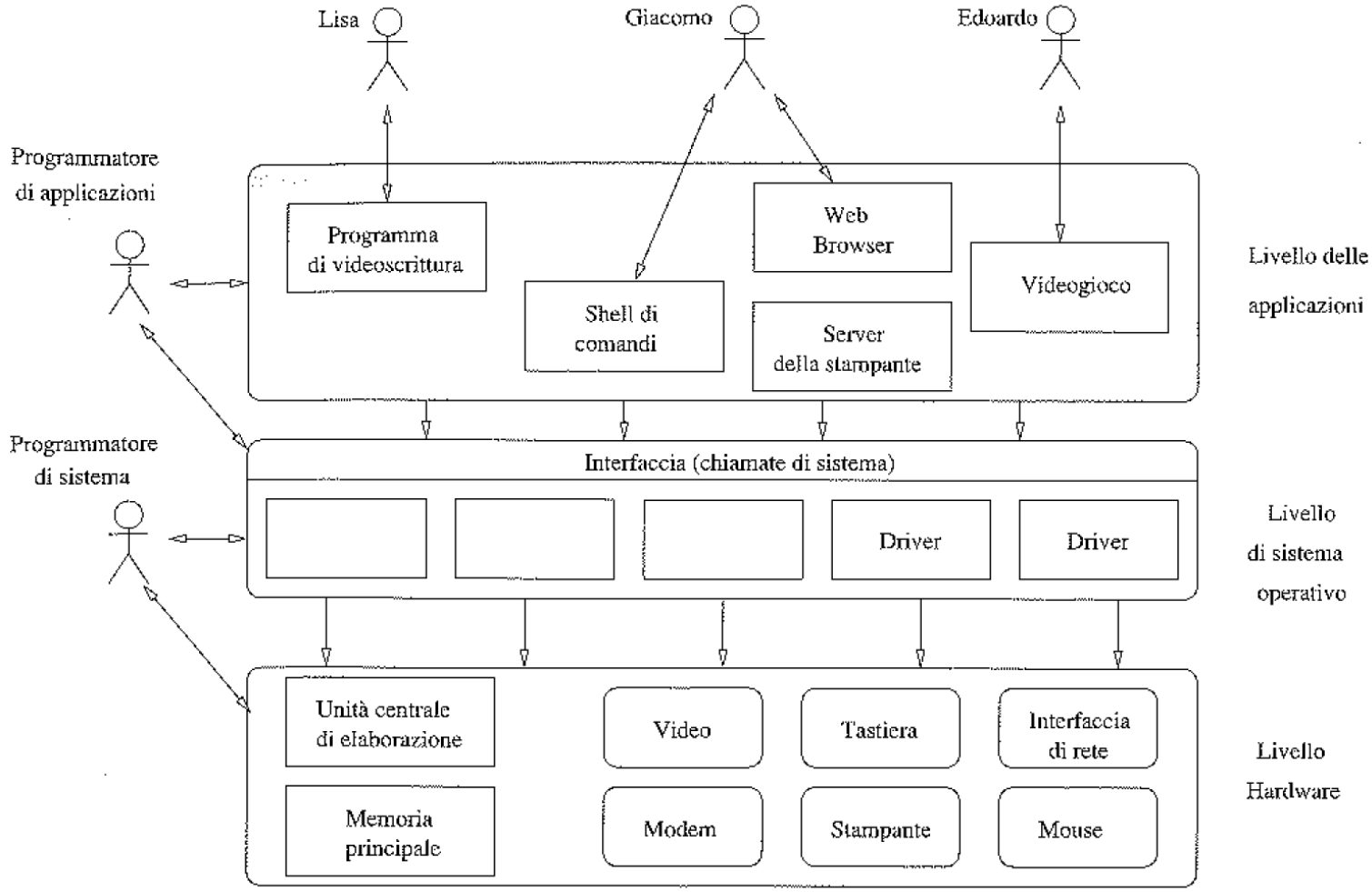
prof. Francesco Zanichelli

Introduzione

- Un sistema di elaborazione può essere visto come l'insieme di:
- hardware
- sistema operativo
- programmi applicativi
- utenti



Introduzione



(da Sistemi Operativi, Ancillotti et. al., MCGraw-Hill 2007)

Che cos'è un sistema operativo?

- Un sistema di calcolo può essere visto come un insieme di risorse Hw e Sw (CPU, memorie, periferiche, file, …) utilizzate per lo sviluppo e la esecuzione dei programmi di utente.

- Tali risorse devono essere:
  - utilizzate secondo un determinato ordine;
  - rese disponibili a più utenti;
  - protette contro accessi non autorizzati;
  - organizzate in modo da garantire la sopravvivenza del sistema in caso di guasti;
  - gestite in modo che risulti semplificato ed efficiente il loro uso, etc.

Definizioni di Sistema Operativo

- Non c'è una definizione universalmente accettata
  - Con il termine sistema operativo si intende quell'insieme di programmi che provvedono alla gestione delle risorse Hw e Sw di un sistema di calcolo
  - "Tutto ciò che viene fornito quando si ordina un Sistema Operativo"
- Una definizione alternativa (Tanenbaum):

  un sistema operativo è un programma che controlla le risorse di un calcolatore e fornisce ai suoi utenti un'interfaccia o macchina virtuale più agevole da utilizzare della macchina "nuda".
- L'unico programma che è sempre in esecuzione sul computer è il *kernel* (nucleo) del SO.
  Il resto è:
  - un programma di sistema (parte del SO)
  - un programma applicativo

Punti di vista sul Sistema Operativo

No, non può andare in cucina
ad aggiungere la panna al suo piatto.
Deve dire a me...

Nella tua cella puoi fare
(quasi) quello che vuoi,
Per tutto il resto devi chiedere a me...

Sistema Operativo

- Può essere visto come:
  1. allocatore di risorse Hw e Sw
     - tempo di CPU, spazio di memoria, dispositivi di I/O, compilatori, etc.
     - Le risorse devono essere assegnate a programmi specifici secondo determinate politiche.

  2. programma di controllo

     controlla l'esecuzione dei programmi per prevenire errori ed usi impropri del calcolatore (in particolare per il controllo dei dispositivi di I/O).

- Obiettivi principali del S.O.:
  - rendere più **semplice** l'uso di un sistema di elaborazione
  - rendere più **efficiente** l'uso delle risorse del sistema di elaborazione.

Sistema Operativo

Il sistema operativo è costituito dall'insieme dei programmi (software o firmware) che **rendono praticamente utilizzabile** l'elaboratore agli utenti cercando contemporaneamente di **ottimizzarne le prestazioni**.

- **Visione top-down**: il sistema operativo come una macchina estesa (fornisce astrazione, nasconde dettagli)
- **Visione bottom-up**: il sistema operativo come un gestore di risorse (fornisce protezione, risoluzione di conflitti o interferenze)

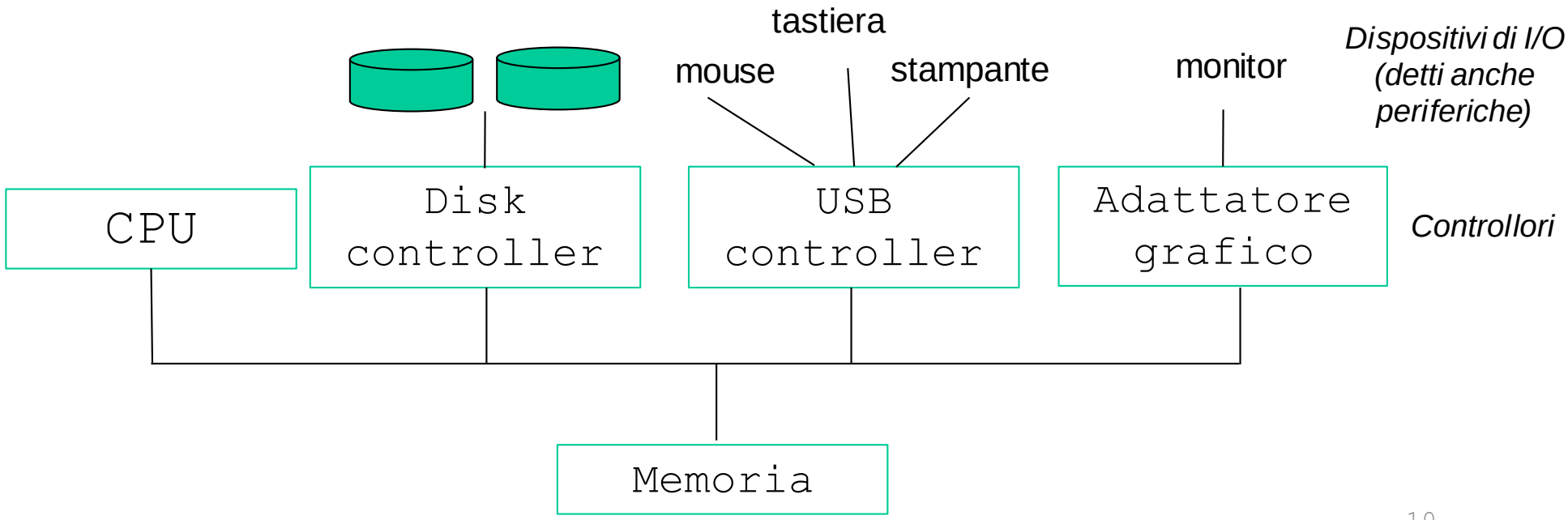
Avvio dell'elaboratore

- Un programma di *bootstrap* è caricato all'accensione o al reboot:
  - è normalmente memorizzato in una memoria non volatile (ROM/EPROM/EEPROM/Flash)
  - inizializza e verifica il corretto funzionamento i componenti HW del sistema
  - carica il kernel del SO e inizia l'esecuzione

Nei PC il tradizionale programma di bootstrap BIOS (Basic Input/Ouput System) è stato sostituito dal più complesso UEFI (Unified Extensible Firmware Interface)

Semplice organizzazione di un elaboratore

- Uno o più processori (CPU, controllori di dispositivi) si connettono mediante un bus condiviso che fornisce anche accesso a memoria condivisa
- Esecuzione concorrente delle CPU e dei dispositivi che competono per i cicli di memoria



Risorse hardware

- processori (registri, unità aritmetiche, parallelismo interno)
- memorie
- canali di comunicazione (ad es. USB)
- dispositivi di I/O (ad es. monitor, stampanti, etc.) e controllori dei dispositivi

- Grande evoluzione sia sui singoli componenti che sulle tecniche di collegamento

- Spostamento di intelligenza verso i dispositivi

- Gerarchia di memoria
  - memoria cache
  - memoria centrale (principale, RAM, core)
  - memoria secondaria (dischi magnetici, elettronici)
  - memoria terziaria (supporti a nastro o ottici per backup)

Gestione delle risorse

Significa:

- tenere traccia delle risorse
- adottare strategie di assegnazione
- allocare le risorse
- recuperare le risorse inutilizzate
- rilevare eventuali usi impropri

Funzioni specifiche:

- gestione della memoria principale
- gestione dei processori
- gestione dei dispositivi periferici
- gestione della memoria secondaria

Funzioni specifiche di gestione

- gestione dei dispositivi periferici
  - mascherare al programmatore la complessità delle operazioni di I/O
  - effettuare controlli sul corretto funzionamento delle operazioni
  - risolvere conflitti nell'utilizzo di una stessa periferica da parte di più programmi
  - consentire il massimo sfruttamento delle periferiche.

- gestione dei processori
  - decidere quale programma userà il processore (**scheduling**) in base a criteri di corretto funzionamento e di efficienza
    - NB **un programma in esecuzione è detto processo** (vedi più avanti)
  - verificare che i programmi rilascino il processore entro il tempo stabilito.

Funzioni specifiche di gestione

- gestione della memoria centrale
  - caricare in memoria programmi e dati
  - evitare interferenze fra programmi diversi
  - assegnare la memoria in base a criteri di efficienza
  - minimizzare i trasferimenti tra memoria centrale e memoria di massa.

- gestione della memoria secondaria
  - consentire l'accesso all'informazione in base alla sua organizzazione logica (File System) anziché fisica (ad es. piatti, tracce, settori)
  - controllare i diritti di accesso ai file da parte degli utenti
  - consentire creazione, modifica e cancellazione dei file, ...

Proprietà fondamentali di un S.O.

- affidabilità
- efficienza
- sicurezza

Proprietà fondamentali di un S.O.

- **Architettura**: come è organizzato il SO? Quali componenti? Quali relazioni tra componenti?
- **Condivisione**: quali risorse vengono condivise tra utenti e/o programmi? In che modo?
- **Efficienza**: come massimizzare l'utilizzo delle risorse disponibili?
- **Affidabilità/tolleranza ai guasti**: quale probabilità di malfunzionamenti? come reagisce il SO ad eventuali malfunzionamenti (HW/SW)?
- **Estendibilità**: è possibile aggiungere funzionalità al sistema?
- **Protezione** e **Sicurezza**: il SO deve impedire **interferenze** tra programmi/utenti e attacchi dalla rete. In che modo?
- **Conformità a standard**: portabilità, estendibilità, apertura

Aree di applicazione di un S.O.

- sistemi di tipo generale

- sistemi in tempo reale

  - applicazioni per il controllo di processo e di apparati fisici
    - sistemi in tempo reale in senso stretto, → spesso «hard real-time»

  - applicazioni interattive, interrogazione di basi di dati, query web, applicazioni multimediali
    - better sooner than later … → sistemi «soft real-time»

Funzioni di un S.O.

- definizione e gestione dell'interfaccia utente
- gestione dei «lavori» (job, programmi) degli utenti
- gestione delle risorse del sistema
- ausili per la messa a punto dei programmi
- ausili per la gestione dei dati - file system
- funzioni ausiliarie di sistema per
  - affidabilità
  - sicurezza
  - accounting (registrazione degli utilizzi delle risorse)

Utenti del S.O.

- **utenti finali del sistema**
  per essi il sistema operativo è trasparente

- **programmatori applicativi**
  utilizzano i servizi del S.O. per la realizzazione e l'esecuzione dei loro programmi

- **programmatori di sistema**
  aggiornano e modificano i programmi del S.O. per adeguarli a nuove necessità del sistema o degli utenti applicativi

- **operatori**
  controllano il funzionamento e rispondono alle richieste di intervento da parte del sistema

- **amministratore del sistema**
  stabilisce le politiche di gestione del sistema e ne cura l'osservanza

Tipi di S.O.

sistemi proprietari

- progettati dai costruttori al fine di sfruttare in modo ottimale le risorse di ciascun tipo di macchina
- programmi utente e applicazioni si interfacciano al SO in modo diverso tra le diverse famiglie di sistemi
- esempi storici:
  - IBM: OS/360 370, VM, MVS
  - DEC: RT-11, VMS

sistemi standard

- progettati da aziende software o da grandi utenti per consentire lo sviluppo di applicazioni portabili su sistemi diversi
- l'interfaccia di programmazione con cui le applicazioni interagiscono con il SO rimane costante nelle diverse versioni
- esempi:
  - UNIX, MS-DOS, Windows

Tipi di S.O.

sistemi aperti

- Realizzati e mantenuti da comunità di sviluppatori volontari (spesso con il supporto di aziende) per consentire applicazioni portabili e indipendenti da piattaforme proprietarie
- Movimento per il software «open source»
- Le applicazioni beneficiano della piena conoscenza del SO, il SO evolve nel tempo e migliora grazie alla sua trasparenza
- Problemi: stabilità e controllo delle versioni e dei rilasci del SO
- Esempio canonico: Linux
- Interfaccia utente e *look and feel* sono modificabili

Evoluzione nell'uso dei calcolatori

- Scrivere programmi che realizzano algoritmi:
  - strutture dati => transienti
  - libreria di sottoprogrammi => capitale

- Evoluzione verso applicazioni in cui i dati rappresentano lo stato del sistema che evolve
  - strutture dati => capitale
  - le strutture dati sopravvivono al programma

- Applicazioni dedicate (embedded, sistemi bancari, banche dati, controllo di processo, automobili, computer and internet everywhere)

- Evoluzione verso applicazioni di A.I.:
  - riconoscimento del linguaggio naturale
  - basi di conoscenza, sistemi esperti, agenti intelligenti
  - robotica
  - visione

Evoluzione nei Sistemi Operativi

- Il SO deve evolvere in conseguenza dell'evoluzione dei sistemi di elaborazione e delle applicazioni
- Alcune tendenze consolidate dell'ultimo decennio:
  - l'interconnessione, le reti
  - la mobilità
  - la multimedialità
  - il parallelismo nell'hardware (multi-core CPU, GPU, heterogeneous computing)
  - la complessità dei sistemi

- L'architettura interna del SO deve fornire supporto adeguato anche a queste nuove esigenze

  → principi architetturali robusti e stabili