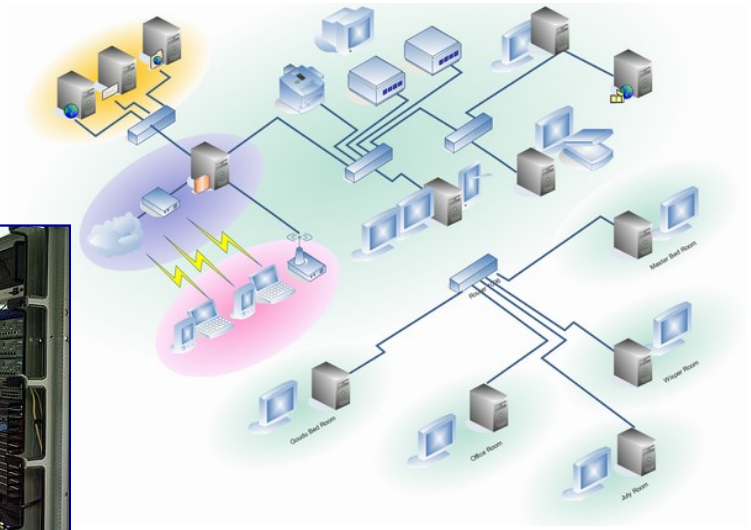
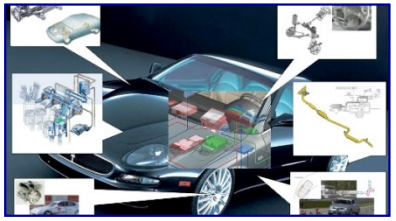
INFORMATICA

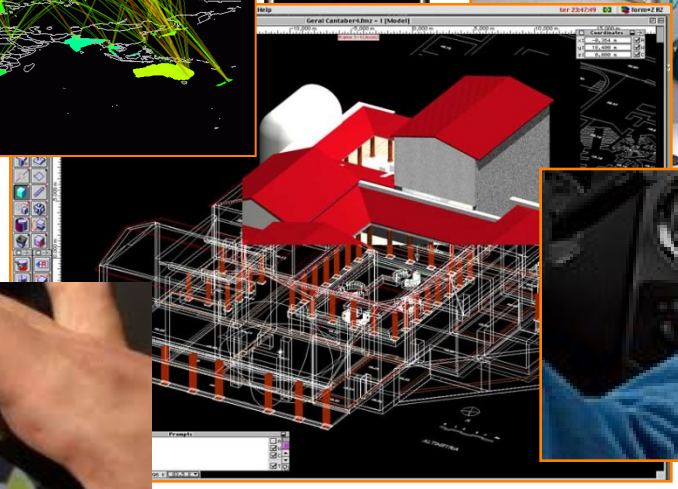
- Varie definizioni:
  - "Scienza degli elaboratori elettronici" (*Computer Science*)
  - "Scienza dell'informazione"
- Definizione proposta:
  - ***Scienza della rappresentazione e dell'elaborazione dell'informazione***

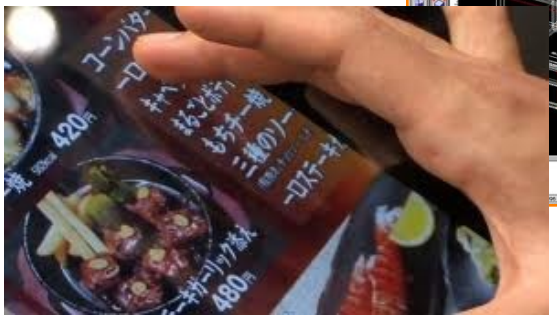
SISTEMI INFORMATICI





L'INFORMATICA È DAPPERTUTTO!





QUALI SISTEMI, QUALI PROBLEMI?

Sistemi distribuiti: gestione e trattamento di risorse e informazioni distribuite all'interno di una rete.

Esempi:

- *cloud*, sistemi che consentono l'uso trasparente di risorse (computer, storage, applicazioni ..) distribuite all'interno di una rete.

iCloud

Expedia.com

- sistemi web per la prenotazione di viaggi

QUALI SISTEMI, QUALI PROBLEMI?

Sistemi mobili: adattamento al “contesto” nella ricerca/elaborazione di informazioni

Esempio:

app su smartphone (es. previsioni meteo, guide turistiche, etc.)

QUALI SISTEMI, QUALI PROBLEMI?

Sistemi multimediali: gestione ed elaborazione di dati multimediali.

Esempi:

- sistemi di classificazione e riconoscimento di immagini o tracce audio.

- sistemi di videosorveglianza: riconoscimento automatico di situazioni sospette/pericolose.

QUALI SISTEMI, QUALI PROBLEMI?

Sistemi intelligenti: ragionamento automatico basato su conoscenza, per estrarre nuova conoscenza.

Esempi:

- Semantic web (Web 3.0): gestione, integrazione, interpretazione e ragionamento automatico relativi a conoscenza disponibile tramite il web.

- Sistemi di *market mining*, analisi dei dati relativi alle vendite per estrarre regole sui processi decisionali che guidano in clienti negli acquisti.

amazon.com

QUALI SISTEMI, QUALI PROBLEMI?

Sistemi embedded: sistemi di elaborazione progettati appositamente per una determinata applicazione realizzati spesso con una piattaforma hardware ad hoc.

- Hardware-Software "co-design": conoscendo a priori lo scopo del dispositivo, è possibile ottimizzare il progetto hardware in funzione della particolare applicazione, e viceversa.

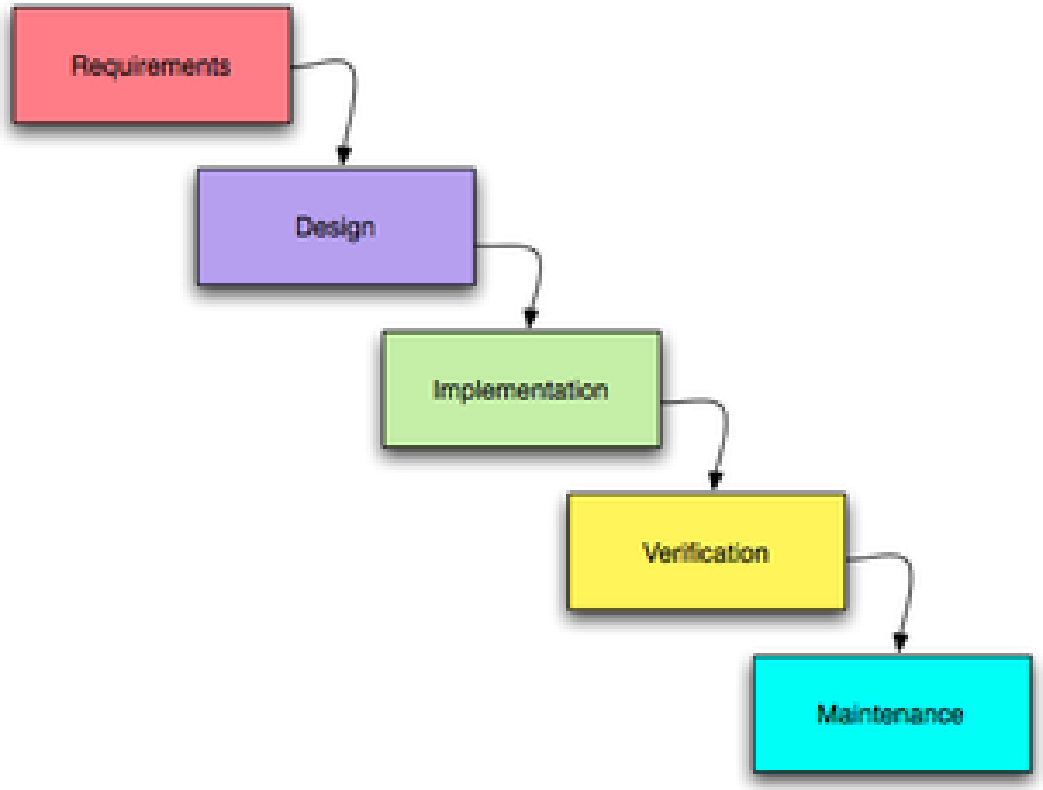
Esempi:

- Computer di bordo di automobili.

- Dispositivi portatili per l'ascolto di musica.

COME VENGONO RISOLTI I PROBLEMI?

- Modelli
- Tecnologie
- Strumenti
- Creatività



L'INFORMATICA COMPRENDE:

- Metodi per la rappresentazione delle informazioni
- Metodi per la rappresentazione delle soluzioni
- Linguaggi di programmazione
- Architettura dei calcolatori
- Sistemi operativi
- Tecnologie Web, reti, middleware e servizi
- Calcolo numerico
- Complessità
- Sistemi informativi
- Intelligenza Artificiale

L'ELABORATORE

Componenti principali

- Unità centrale
- Video ("monitor")
- Tastiera e Mouse
- Lettore CD/DVD
- Dischi fissi ("hard disk")

Componenti accessori

- Stampante
- Modem
- Scanner
- Tavolette grafiche
- Penne USB

…

Strumento per la rappresentazione e l'elaborazione delle informazioni

HARDWARE

TECNOLOGIA DIGITALE

CPU, memoria centrale e dispositivi sono realizzati con **tecnologia elettronica digitale**.

Dati ed operazioni vengono codificati a partire da due valori distinti di grandezze elettriche:

- tensione alta (V_H, 5V)
- tensione bassa (V_L, 0V)

A tali valori vengono convenzionalmente **associate le due cifre binarie 0 e 1:**

- **logica positiva: $1 \leftrightarrow V_H$, $0 \leftrightarrow V_L$**
- logica negativa: $0 \leftrightarrow V_H$, $1 \leftrightarrow V_L$

TECNOLOGIA DIGITALE (segue)

Dati ed operazioni vengono codificati tramite **sequenze di bit**

01000110101 ....

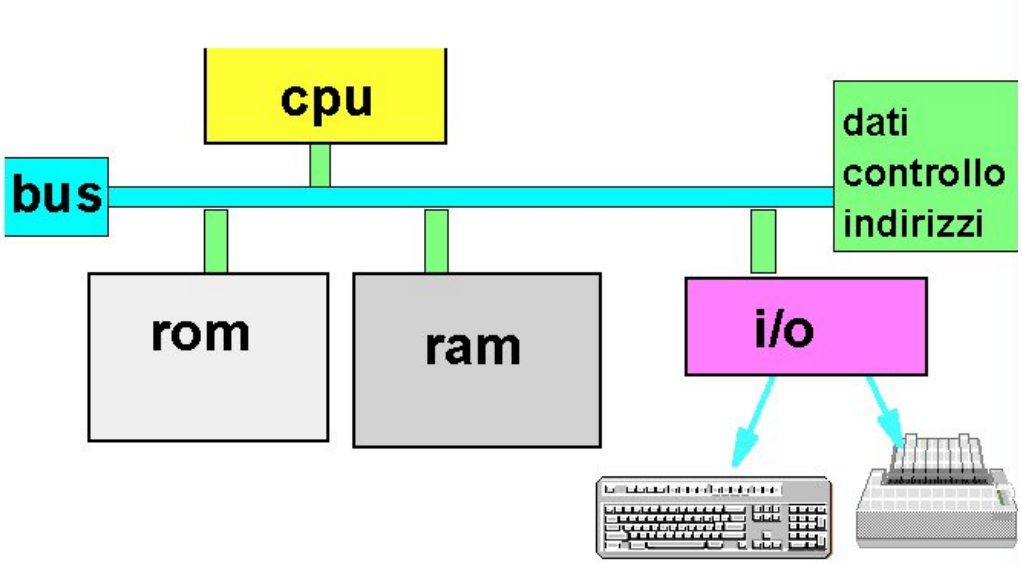
CPU è in grado di operare soltanto in aritmetica binaria, effettuando operazioni *elementari* :

- somma e differenza
- scorrimento (shift)
- ...

Lavorando direttamente sull'hardware, **l'utente è forzato a esprimere i propri comandi *al livello della macchina*, tramite *sequenze di bit.***

HARDWARE

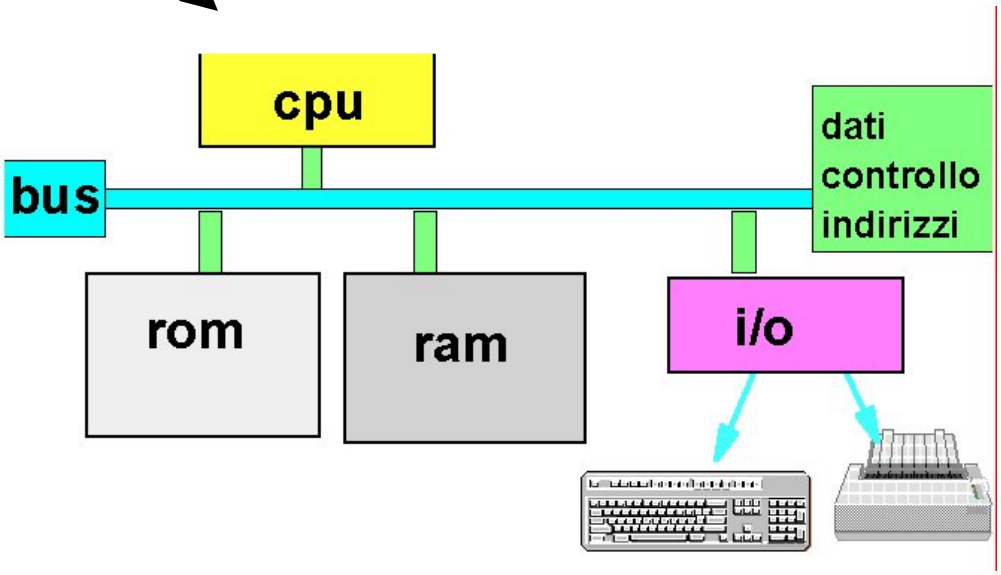
E’ composto da un insieme di ***unità funzionali***



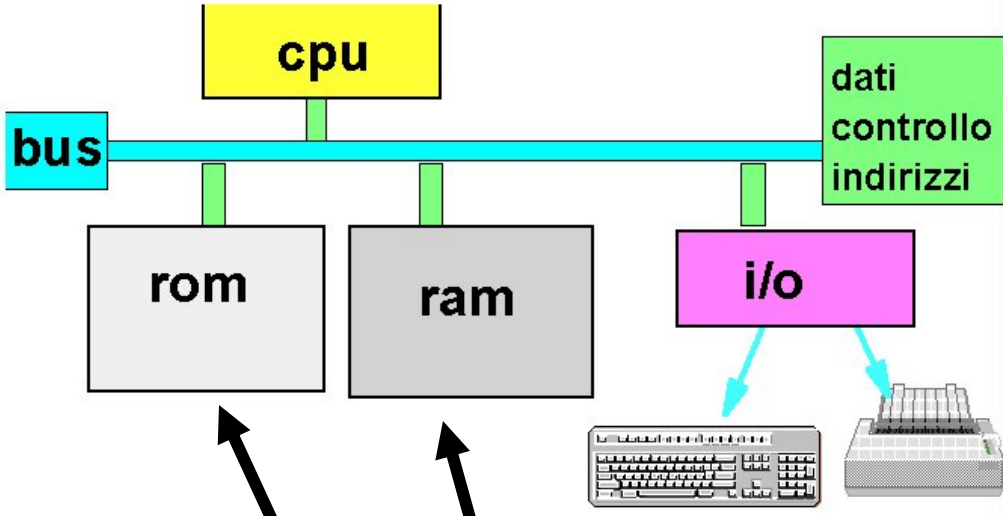
HARDWARE

CPU (Central Processing Unit), o **Processore**

CPU: Svolge le elaborazioni e il trasferimento dei dati, cioè *esegue i programmi*



HARDWARE



RAM & ROM

- Dimensioni relativamente limitate
- Accesso molto rapido

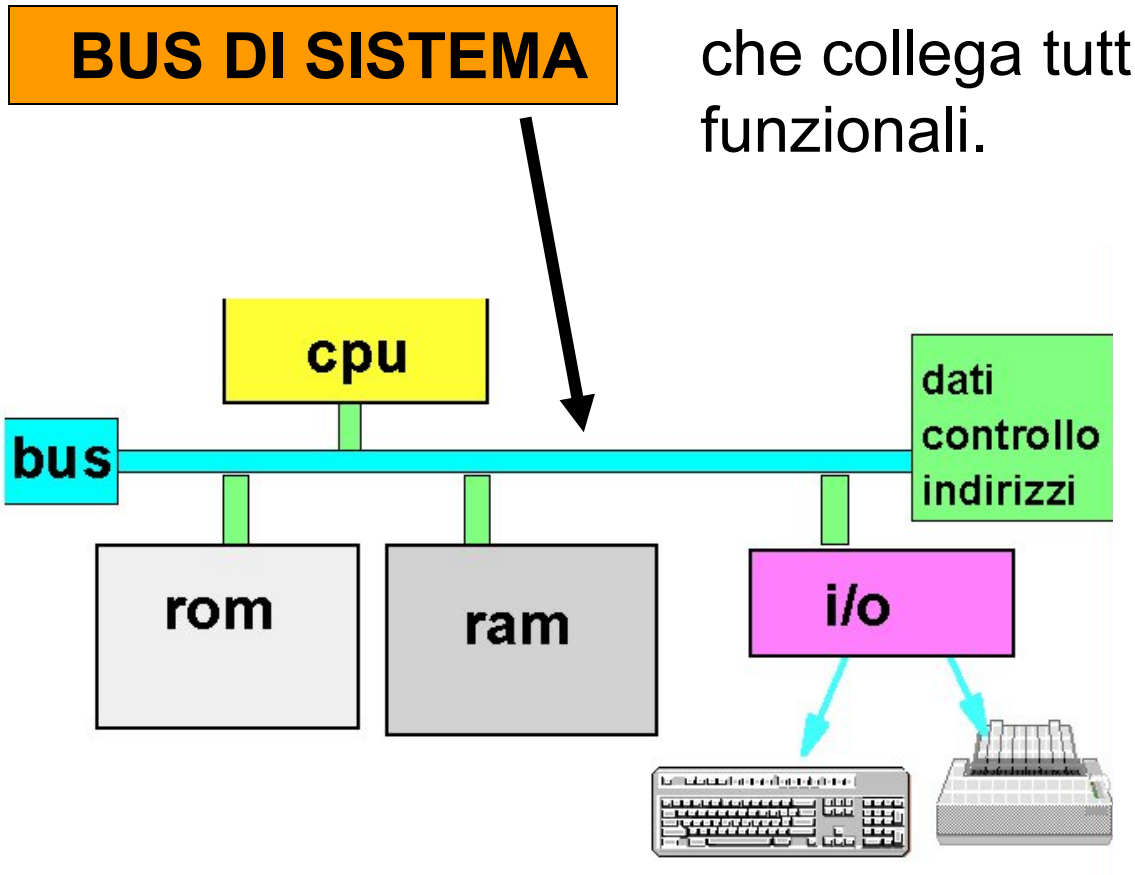
RAM (***Random Access Memory***), e
ROM (***Read Only Memory***)
Insieme formano la *Memoria centrale*

HARDWARE

ATTENZIONE

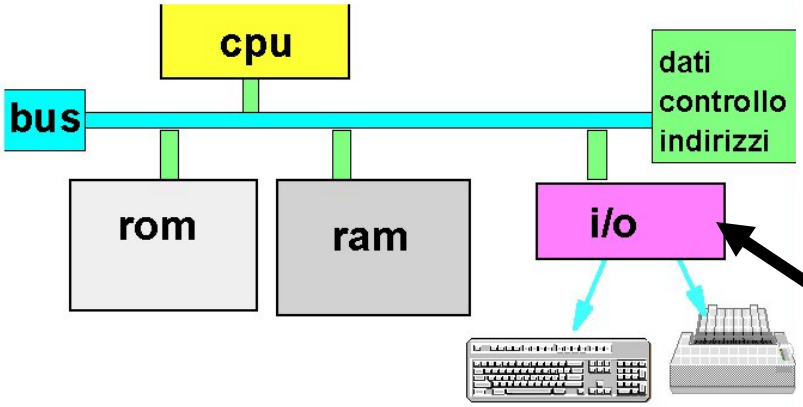
- **RAM è volatile** (perde il suo contenuto quando si spegne il calcolatore)
  - usata per memorizzare dati e programmi
- **ROM è persistente** (mantiene il suo contenuto quando si spegne il calcolatore) ma il suo ***contenuto è fisso e immutabile***
  - usata per memorizzare programmi di sistema

HARDWARE



È una “linea di comunicazione” che collega tutti gli elementi funzionali.

HARDWARE

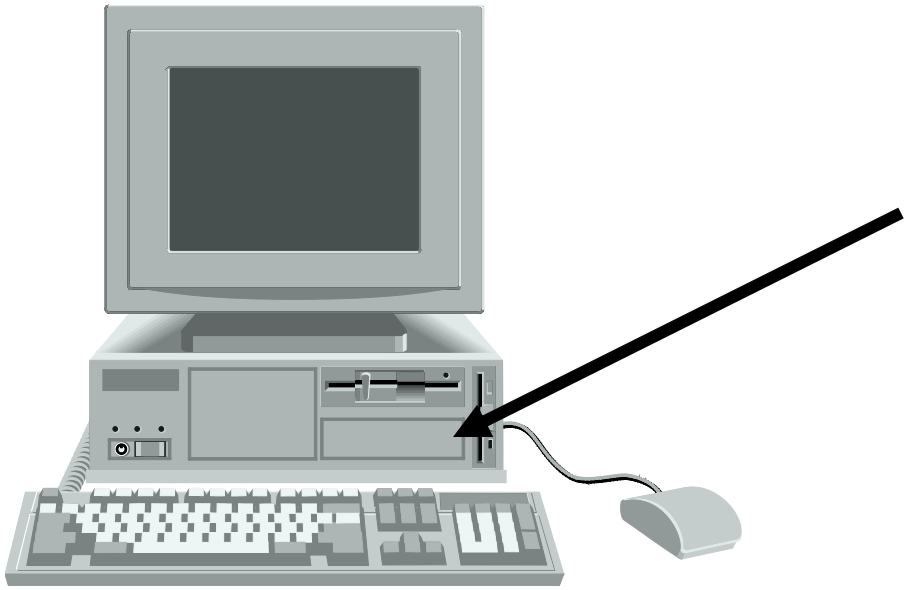


Sono usate per far comunicare il calcolatore con l'esterno (in particolare con l'utente)

UNITÀ DI INGRESSO / USCITA (I/O)

- Tastiera e Mouse
- Video e Stampante
- Scanner
- Tavoletta grafica
- **Dispositivi di memoria di massa**
- …

HARDWARE



MEMORIA DI MASSA

- Dischi
- CD
- DVD
- Penne USB
- SSD
- Nastri (old-style)

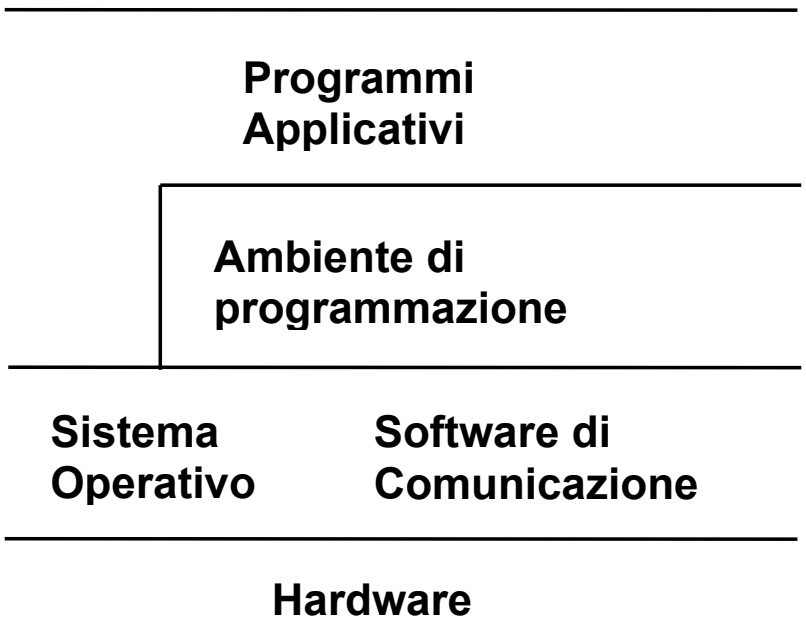
- memorizza **grandi quantità** di informazioni
- **persistente** (le informazioni non si perdono spegnendo la macchina)
- accesso molto meno rapido della memoria centrale (**milli**secondi contro **nano**secondi / differenza 10^{6})

IL SOFTWARE

Software:
insieme (complesso) di programmi.

Organizzazione a strati, ciascuno con funzionalità di livello più alto rispetto a quelli sottostanti

Concetto di
MACCHINA VIRTUALE

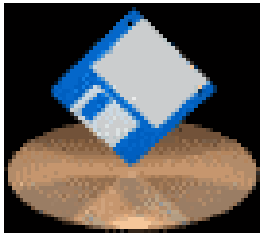


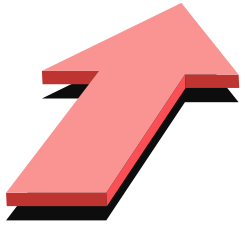
IL SISTEMA OPERATIVO

Strato di programmi che opera *al di sopra di hardware e firmware* e **gestisce l'elaboratore**.

Solitamente, è venduto insieme all'elaboratore.

Spesso si può scegliere tra *diversi sistemi operativi* per lo stesso elaboratore, con diverse caratteristiche.





Esempi:

- Windows
- Linux
- macOs
- Android
- iOS

FUNZIONI DEL SISTEMA OPERATIVO

Le funzioni messe a disposizione dal S.O. dipendono dalla complessità del sistema di elaborazione:

- gestione delle risorse disponibili
- gestione della memoria centrale
- organizzazione e gestione della memoria di massa
- interpretazione ed esecuzione di comandi elementari
- gestione di un sistema multi-utente

Un utente "vede" l'elaboratore solo tramite il Sistema Operativo
→ il S.O. realizza una "macchina virtuale"

FUNZIONI DEL SISTEMA OPERATIVO

Qualsiasi operazione di accesso a risorse implicitamente richiesta da comando utente **viene esplicitata dal SO**

Conseguenza:
diversi SO possono realizzare *diverse macchine virtuali* **sullo stesso elaboratore fisico**

Attraverso il S.O. il livello di interazione fra utente ed elaboratore viene elevato:

- senza S.O.: sequenze di bit
- con S.O.: comandi, programmi, dati

I sistemi operativi si sono evoluti nel corso degli ultimi anni (interfacce grafiche, Macintosh, Windows, ...)

RUOLO DEL SISTEMA OPERATIVO

Il S.O. **traduce le richieste dell'utente** in opportune **sequenze di istruzioni,** a loro volta trasformate in **valori e impulsi elettrici** per la macchina fisica.

e viceversa:

ESEMPIO

e viceversa:

Utente:	Sistema Operativo:
“esegui progr1”	- input da tastiera - ricerca codice di “progr1” su disco - carica in memoria centrale codice e dati <elaborazione>
Utente:	**Sistema Operativo:**
“stampa 10”	- output su video

CLASSIFICAZIONE dei S.O.

In base al numero di utenti:

- **Mono-utente (*mono-user*):** un solo utente alla volta può utilizzare il sistema
- **Multi-utente (*multi-user*):** più utenti possono interagire contemporaneamente con la macchina.

Nel caso di più utenti contemporanei, **il Sistema Operativo deve fornire a ciascuno l'astrazione di *un sistema "dedicato".***

CLASSIFICAZIONE dei S.O.

In base al numero di programmi in esecuzione:

- **Mono-programmato (*mono-task*):** si può eseguire *un solo programma* per volta
- **Multi-programmato (*multi-task*):** il S.O. è in grado di portare avanti contemporaneamente l'esecuzione di più programmi (pur usando una sola CPU).

Nel caso di multi-programmazione **il S.O. deve gestire la suddivisione del tempo** della CPU fra i vari programmi.

PROGRAMMI APPLICATIVI

Risolvono problemi specifici degli utenti:

- *word processor*: elaborazione di testi (*Es. MSWord*)
- *fogli elettronici*: gestione di tabelle, calcoli e grafici (*Es.Excel*)
- *database*: gestione di archivi (*Es. Access*)
- *suite* (integrati): collezione di applicativi capaci di funzionare in modo integrato come un'applicazione unica. (*Es. Office*)

- Sono scritti in **linguaggi di programmazione** di alto livello
- Risentono in misura ridotta delle caratteristiche della architettura dell'ambiente sottostante (*portabilità*)

AMBIENTI DI PROGRAMMAZIONE

È l'insieme dei programmi che consentono la scrittura, la verifica e l'esecuzione di nuovi programmi (***fasi di sviluppo***).

Sviluppo di un programma

- Affinché un programma scritto in un qualsiasi linguaggio di programmazione sia comprensibile (e quindi eseguibile) da un calcolatore, *occorre* ***tradurlo*** dal linguaggio originario al linguaggio della macchina.

- Questa operazione viene normalmente svolta da speciali programmi, detti ***traduttori***.

TRADUZIONE DI UN PROGRAMMA

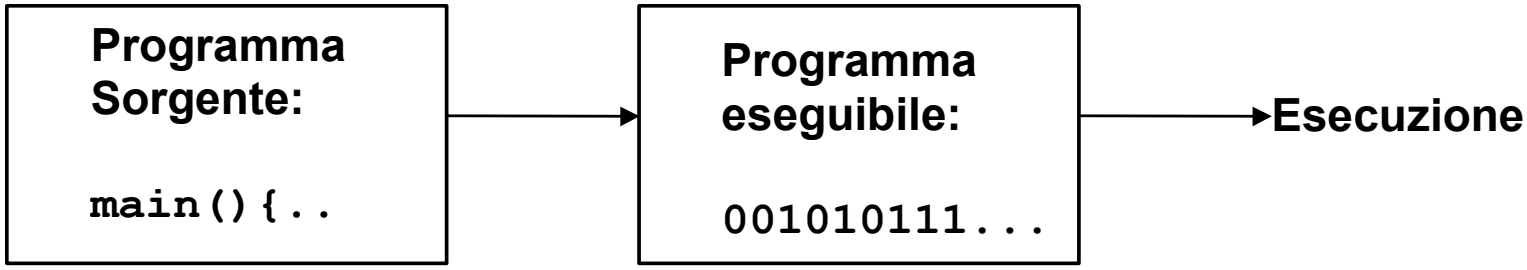
PROGRAMMA	TRADUZIONE
`main()`	
`{ int A;`	`00100101`
`...`	
`A=A+1;`	`11001..`
`if....`	`1011100..`

Il traduttore converte

- ***il testo*** di un programma scritto in un particolare linguaggio di programmazione (***sorgenti***)

- nella corrispondente ***rappresentazione in linguaggio macchina*** (programma ***eseguibile***).

SVILUPPO DI PROGRAMMI

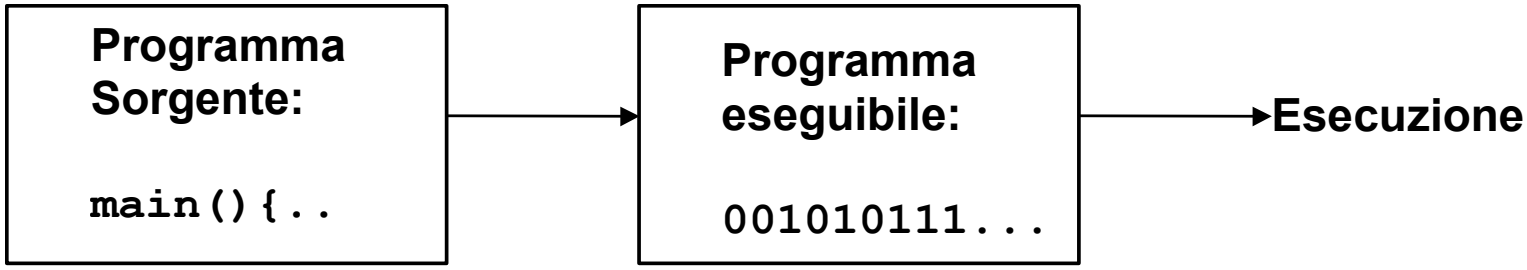


Due categorie di traduttori:

- i ***Compilatori*** traducono l'intero programma (senza eseguirlo!) e producono in uscita il programma convertito in linguaggio macchina

- gli ***Interpreti*** traducono ed eseguono immediatamente **ogni singola istruzione** del *programma sorgente.*

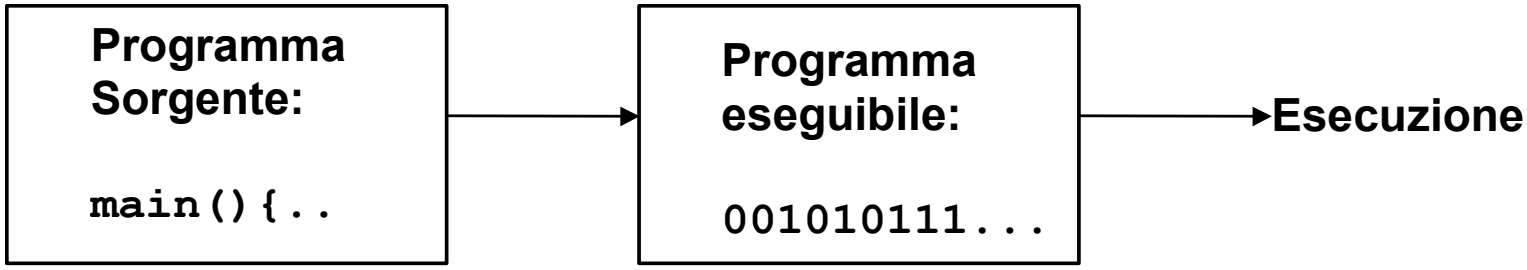
SVILUPPO DI PROGRAMMI (segue)



Quindi:

- **nel caso del compilatore**, lo schema precedente viene percorso ***una volta sola*** prima dell'esecuzione
- **nel caso dell'interprete**, lo schema viene invece attraversato ***tante volte quante sono le istruzioni*** che compongono il programma.

SVILUPPO DI PROGRAMMI (segue)



L'esecuzione di un programma ***compilato*** è **più veloce** dell'esecuzione di un programma ***interpretato***

AMBIENTI DI PROGRAMMAZIONE

COMPONENTI

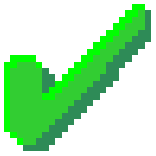
- **Editor:** serve per creare file che contengono **testi** (cioè sequenze di caratteri).
  In particolare, l'editor **consente di scrivere il *programma sorgente.***

E poi….

AMBIENTI DI PROGRAMMAZIONE

I° CASO: COMPILAZIONE

- **Compilatore:** opera la **traduzione di un programma *sorgente*** (scritto in un linguaggio ad alto livello) **in un *programma oggetto*** direttamente eseguibile dal calcolatore.



PRIMA si traduce ***tutto il programma***
POI si esegue **la versione tradotta**.

AMBIENTI DI PROGRAMMAZIONE (2)

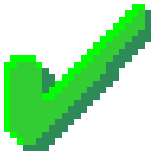
I° CASO: COMPILAZIONE (segue)

- **Linker:** (*collegatore*) nel caso in cui la costruzione del programma oggetto richieda l'unione di ***più moduli*** (compilati separatamente), il linker provvede a **collegarli** formando un unico *programma eseguibile.*
- **Debugger:** (*"spulciatore"*) consente di **eseguire passo-passo** un programma, **controllando via via quel che succede**, al fine di ***scoprire ed eliminare errori*** non rilevati in fase di compilazione.

AMBIENTI DI PROGRAMMAZIONE (3)

II° CASO: INTERPRETAZIONE

- **Interprete**: ***traduce ed esegue*** direttamente ***ciascuna istruzione*** del *programma sorgente*, **istruzione per istruzione**.
  È alternativo al compilatore (raramente sono presenti entrambi).



Traduzione ed esecuzione sono *inter-calate*, e avvengono *istruzione per istruzione.*

ATTENZIONE: PROBLEMI

- Progetti oltre il budget
- Progetti oltre i limiti di tempo
- Software di scarsa qualità
- Software che spesso non rispettava i requisiti
- Progetti ingestibili e codice difficile da manutenere

“Se il settore dell’automobile si fosse sviluppato come l’industria informatica, oggi avremo veicoli che costano 25 dollari e fanno 500 Km con un litro”.

(Bill Gates).

“Se le auto funzionassero come i software, si bloccherebbero due volte al giorno senza motivo e l’unica soluzione sarebbe reinstallare il motore”

(Dirigente General Motors)

METODOLOGIE E STRUMENTI

- Programmazione strutturata (Böhm-Jacopini-1966)
- Tecniche di decomposizione (Dijkstra- 1968)
- Verifica formale delle proprietà dei programmi (Floyd, Hoare, fine anni 60)
- Modularizzazione e progettazione per il cambiamento (Parnas, anni 70)
- Programmazione orientata agli oggetti (anni 70)
- Linguaggi di programmazione: ADA, JAVA (90)
- Service oriented architecture (componenti software)
- ……