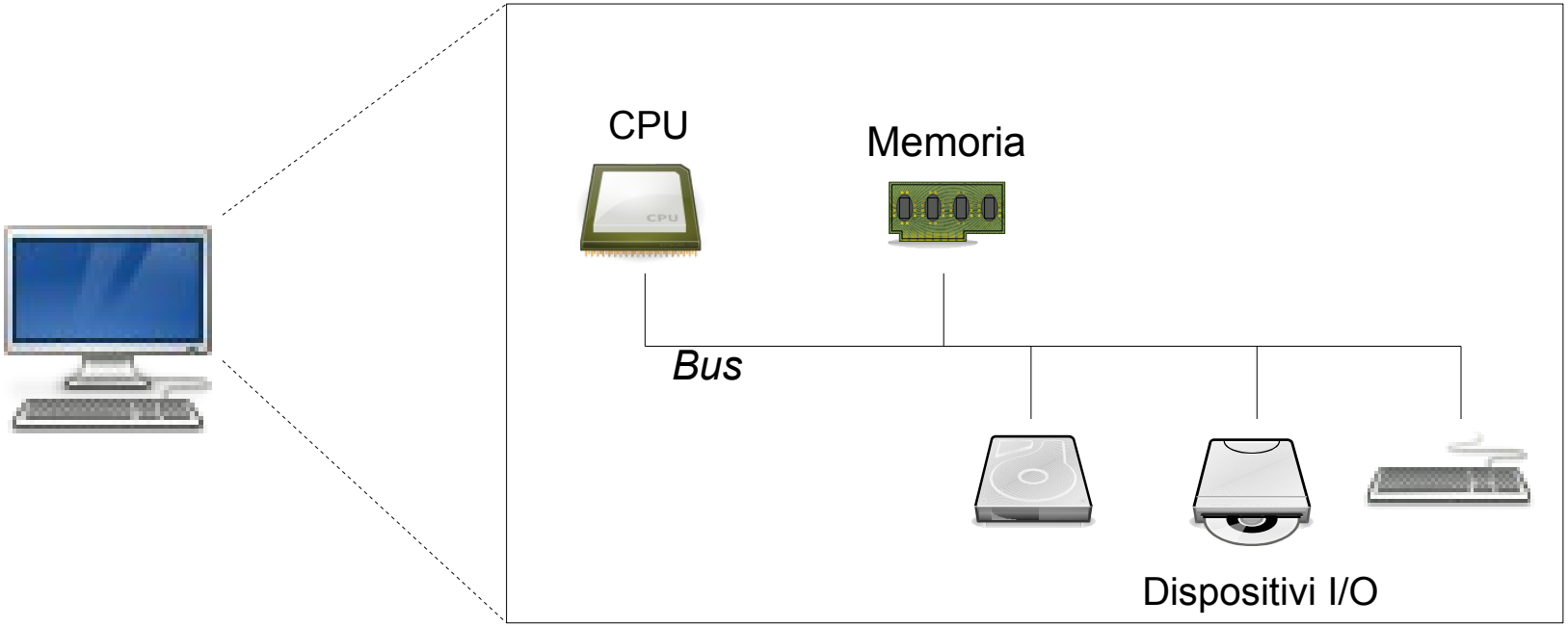
Scuola universitaria professionale della Svizzera italiana
Dipartimento tecnologie innovative
Istituto sistemi informativi e networking

SUPSI

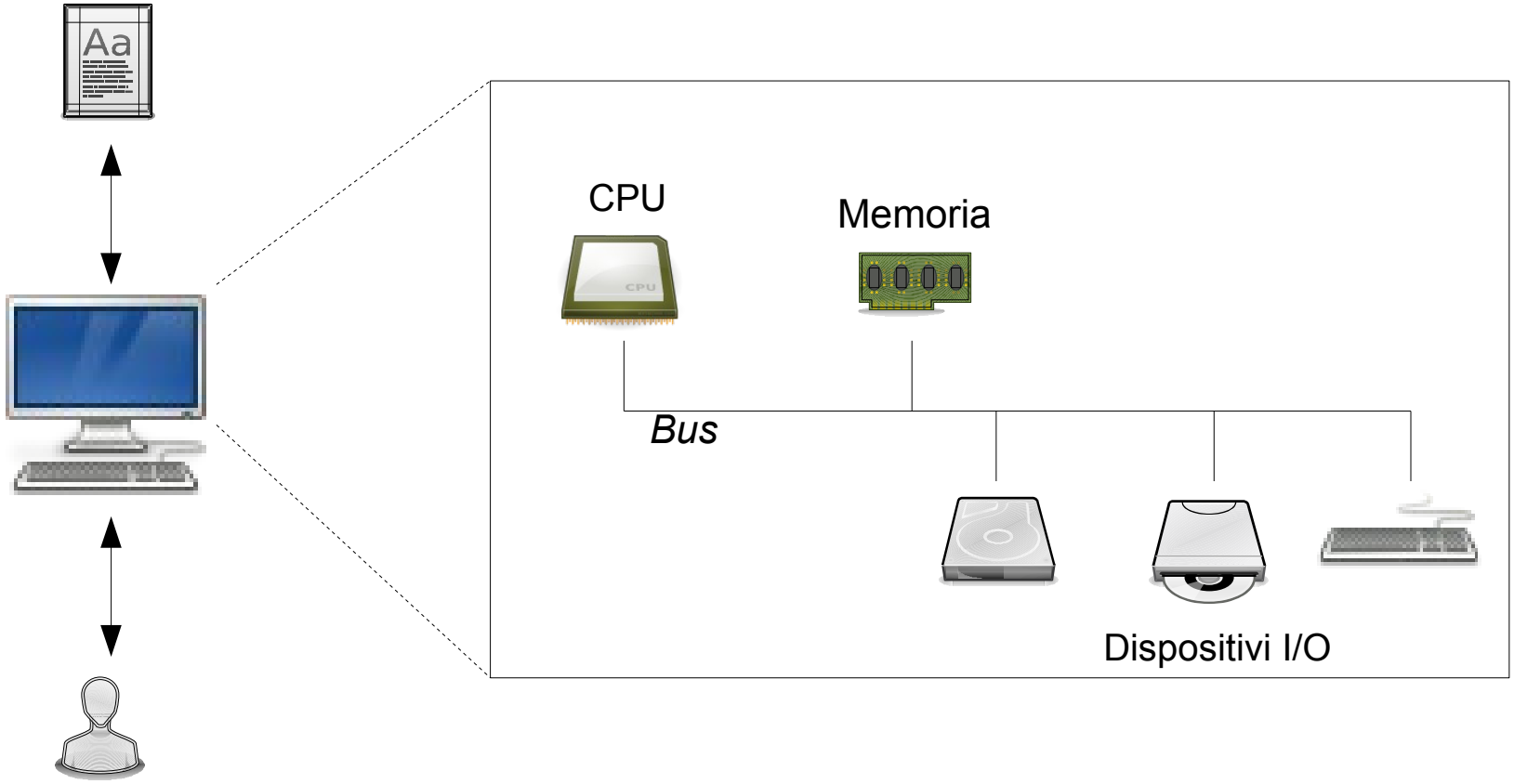
Ambienti Operativi: Introduzione

Amos Brocco, Ricercatore, DTI / ISIN

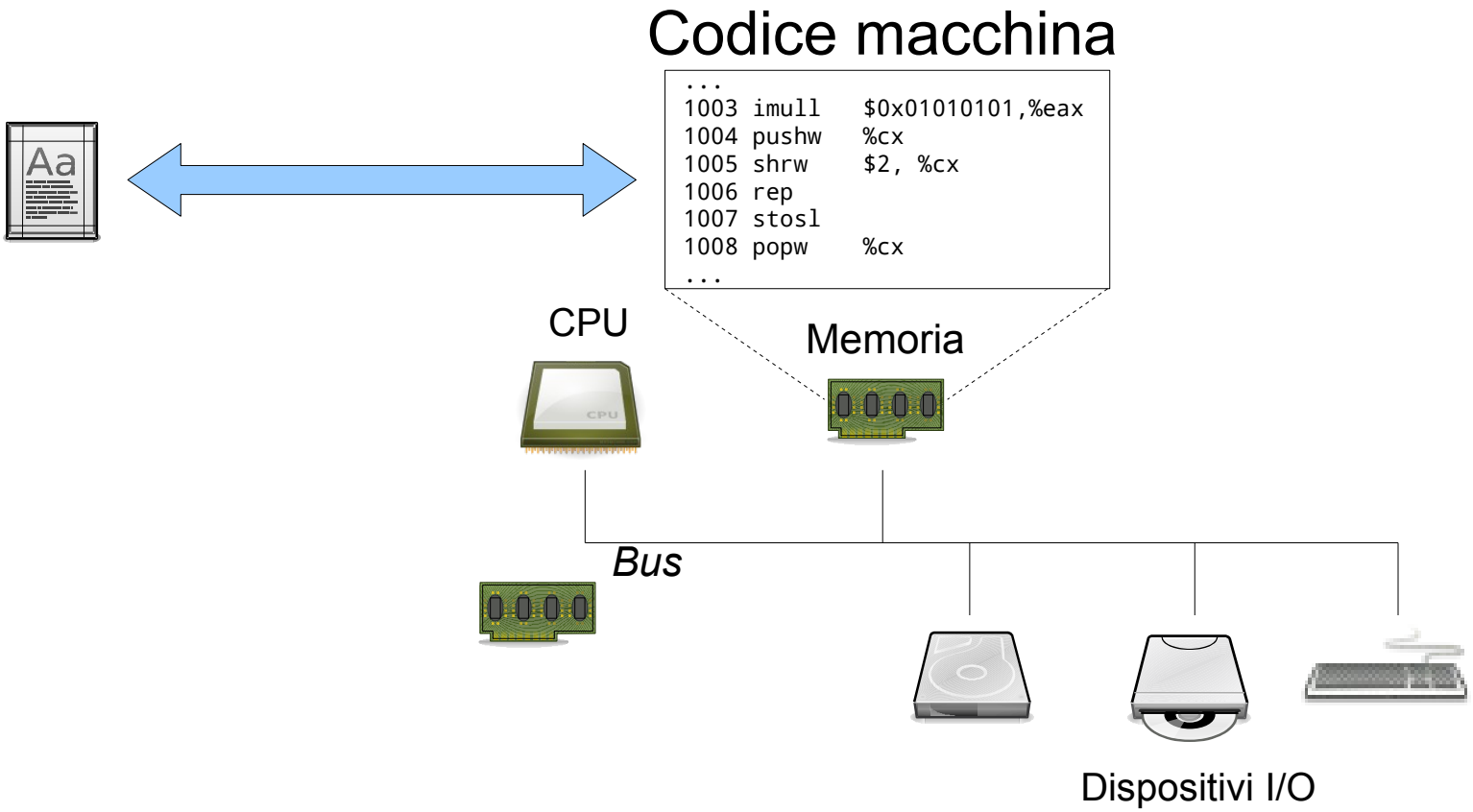
Architettura di un computer



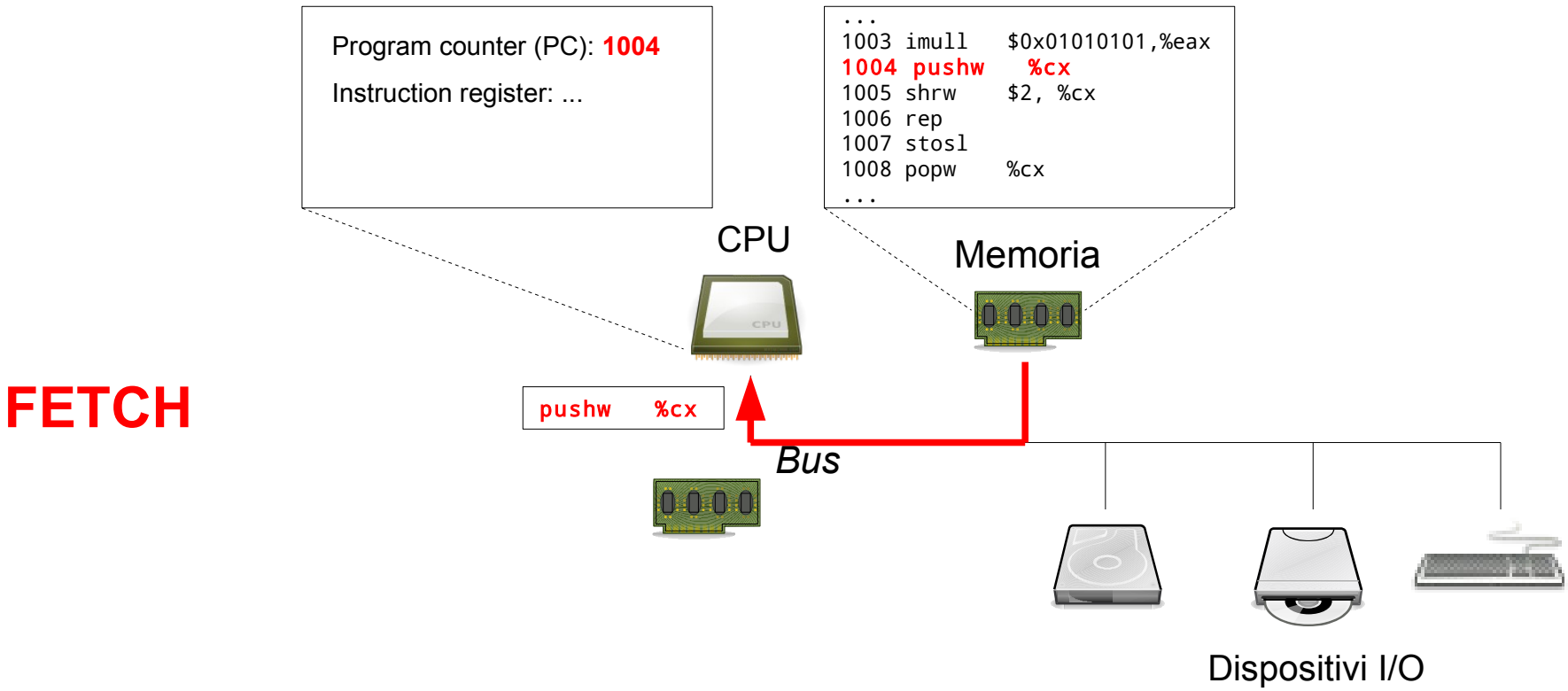
Esecuzione del software



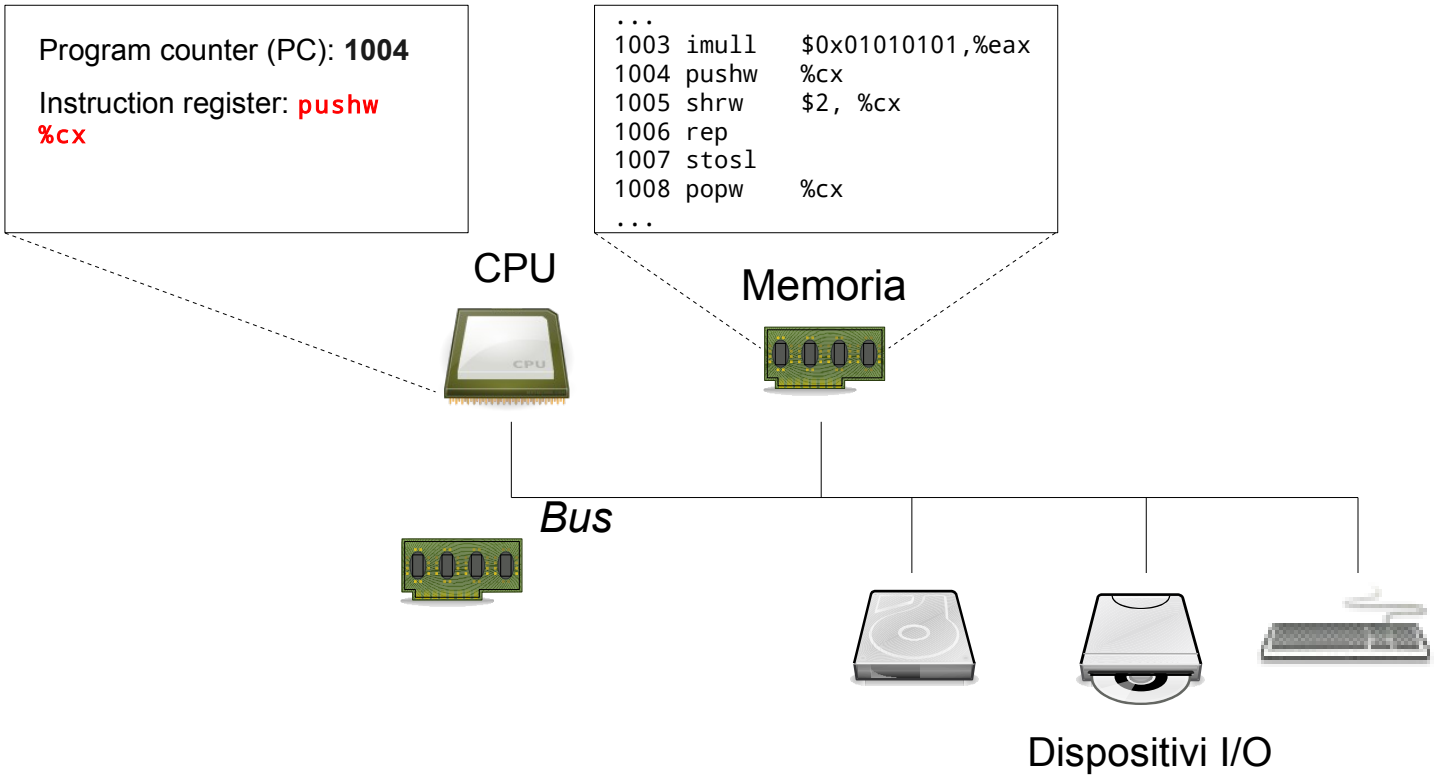
Esecuzione del software



Esecuzione del software



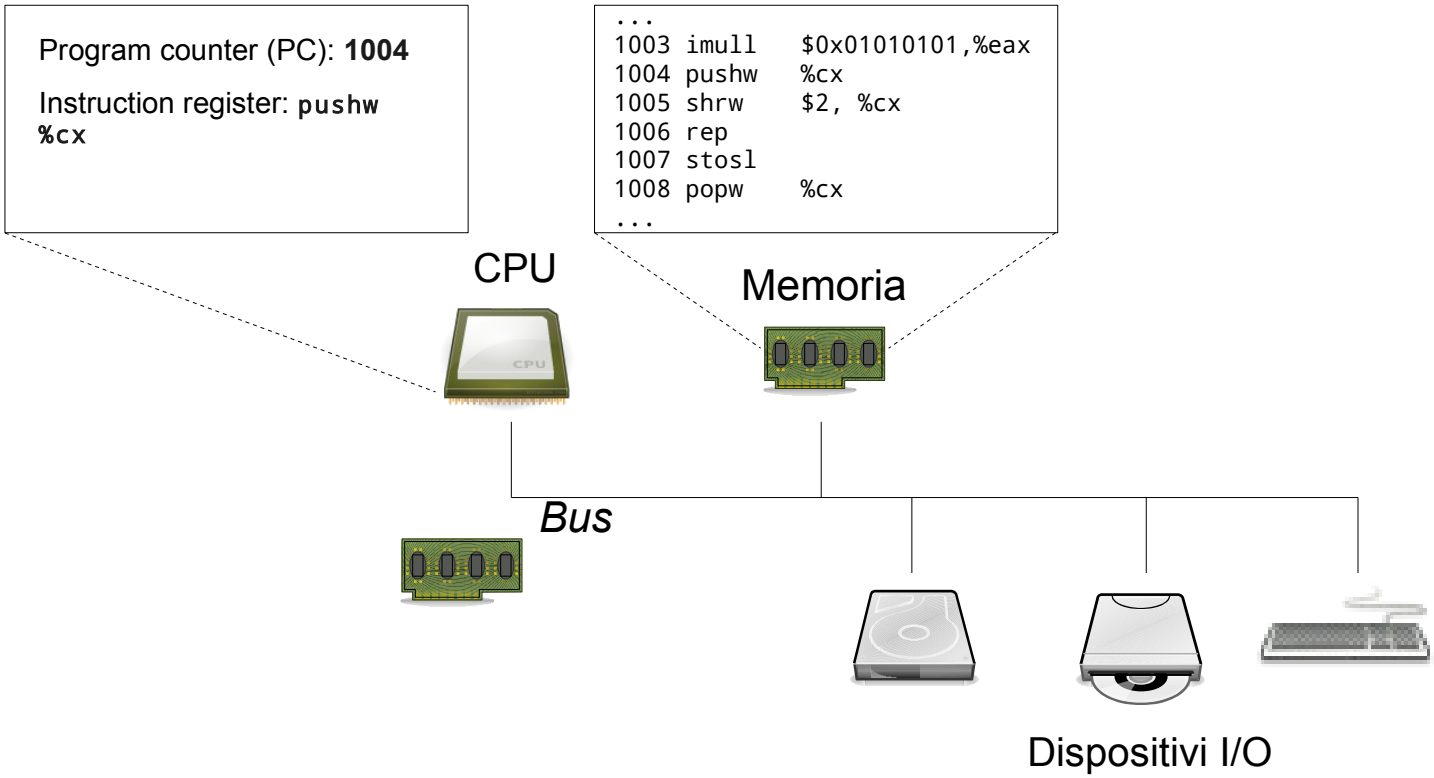
Esecuzione del software



FETCH

DECODE

Esecuzione del software

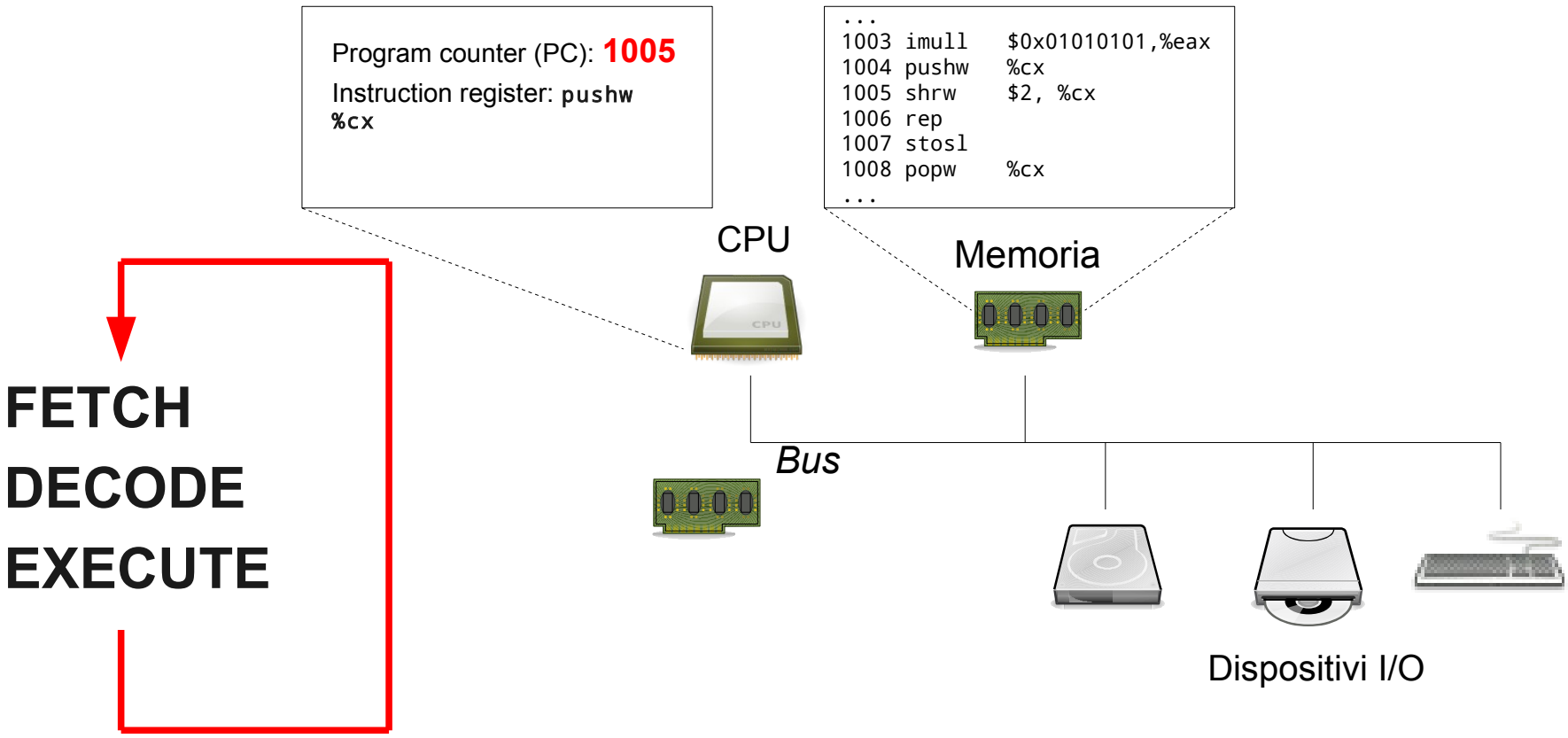


FETCH

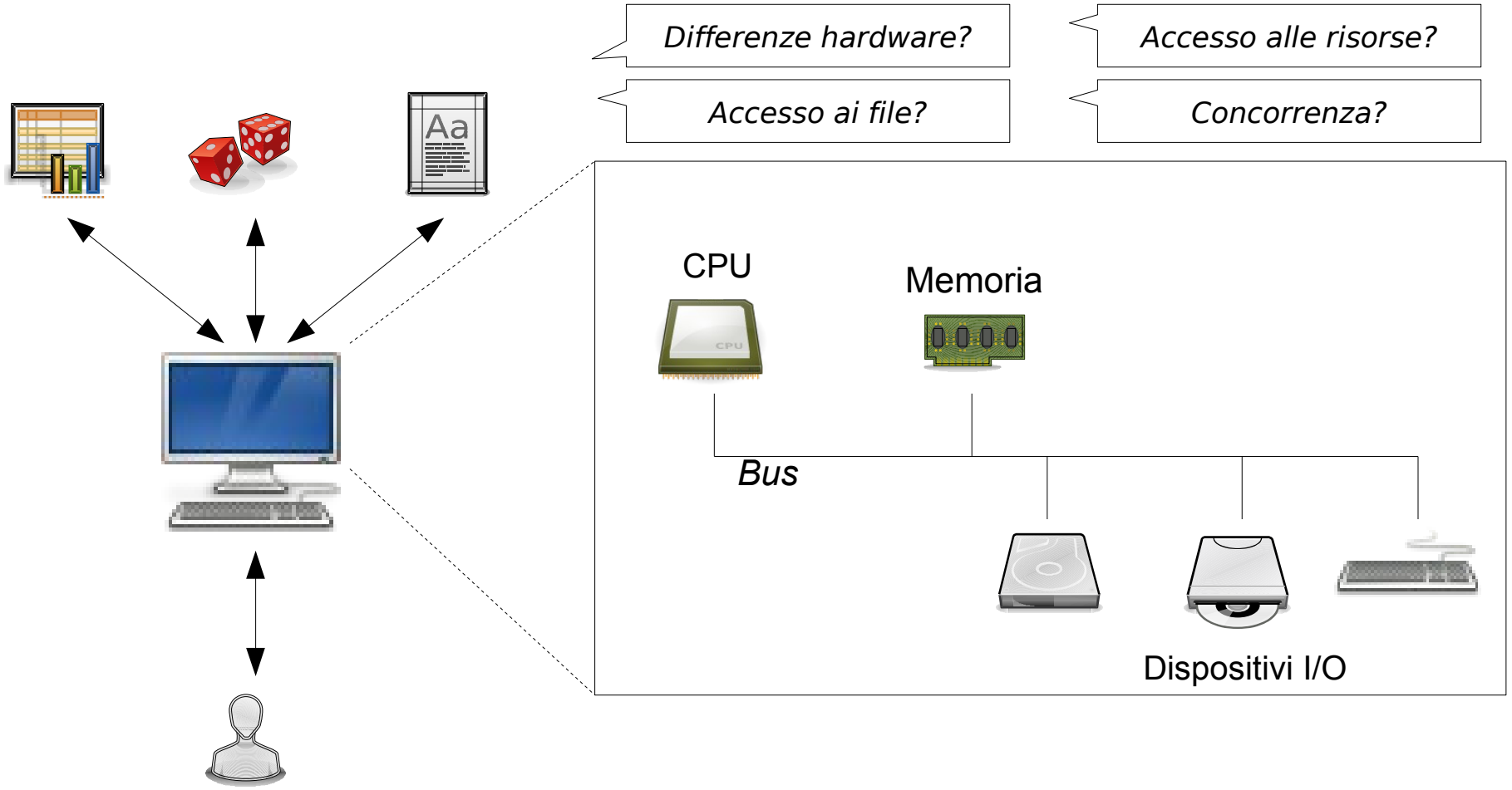
DECODE

EXECUTE

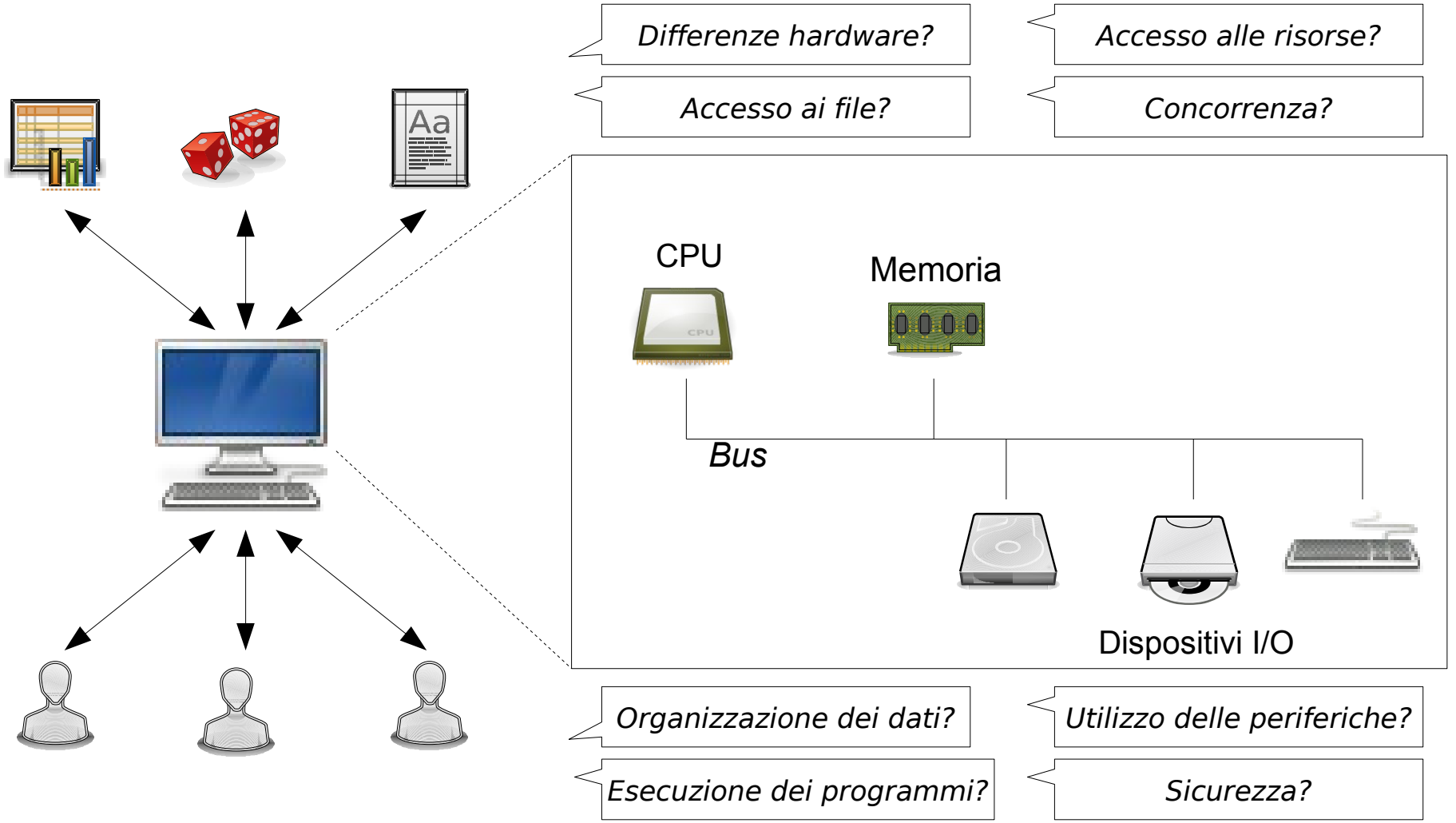
Esecuzione del software



Sistema multi-processo

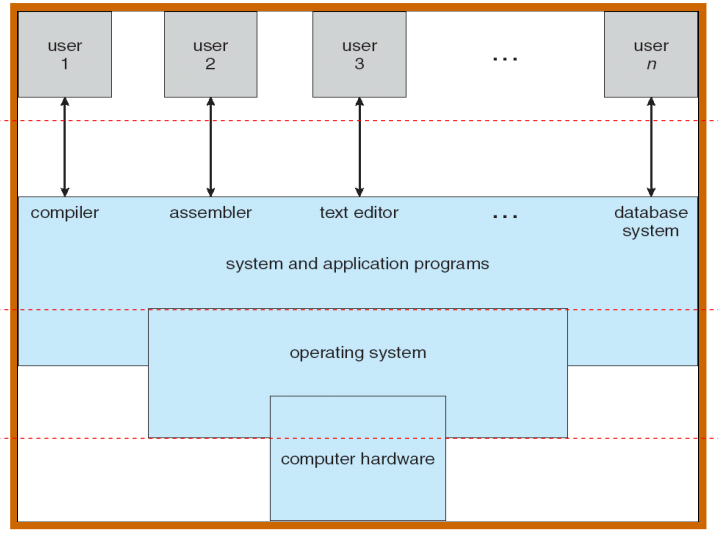


Sistema multi-processo, multi-utente



Cos'è un sistema operativo

- **Astrazione** per nascondere i meccanismi di gestione della macchina
  - Semplifica la programmazione e l'utilizzo: L'utente è in grado di utilizzare la macchina fisica senza conoscere i dettagli della sua struttura interna e del suo funzionamento
- **Gestore di risorse**
  - Permette di far funzionare efficientemente insieme tutte le risorse e fornisce una visione coerente e globale del computer



Programmi applicativi e di sistema: definiscono il modo in cui le risorse di calcolo sono utilizzate per risolvere i problemi degli utenti

Sistema operativo (kernel): controlla e coordina l'uso dell'hardware tra applicazioni e utenti attivi sul sistema

Hardware: fornisce le risorse di calcolo e di memorizzazione (CPU, memoria, periferiche I/O)

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)

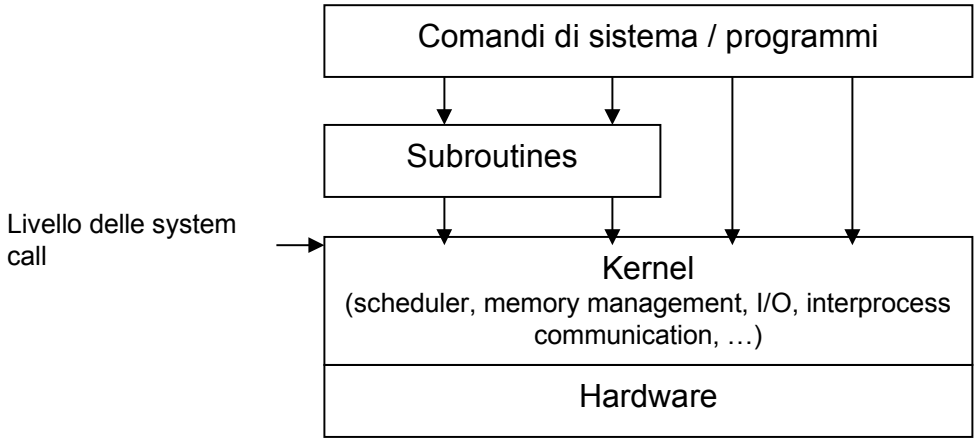
- Definizione
  - Un sistema operativo è un programma che controlla l'esecuzione di programmi applicativi ed agisce come interfaccia fra le applicazioni e l'hardware del calcolatore

Il kernel

- I SO sono costituiti da un insieme di moduli, ciascuno dedicato a svolgere una determinata funzione
- I vari moduli del SO interagiscono tra loro secondo regole precise, al fine di realizzare le funzionalità di base della macchina
- L'insieme dei moduli per la gestione della CPU, della memoria centrale, del filesystem, e per l'accesso controllato alle periferiche è il **kernel**

Comandi, subroutines, system call

- Per chi scrive programmi c'è la possibilità di usufruire dei servizi del kernel attraverso:
  - le system call:
    - Es. apertura/chiusura/lettura/scrittura file
  - le subroutines, cioè funzioni di più alto livello
- I comandi non sono altro che programmi nel cui codice sorgente ci sono chiamate a system call e/o subroutines



Gestione dei processi

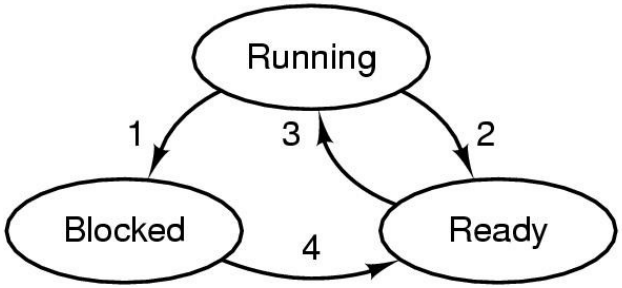
- Un processo è un programma (software) in esecuzione
  - Ad ogni processo sono associati:
    - Spazio di indirizzamento nella memoria (dove è possibile leggere e scrivere)
      - Codice macchina del programma
      - Dati del programma
      - Stack
    - Registri
    - Stato

- Sistemi a singolo processo
  - Un solo processo in esecuzione
    - es. sistema batch
- Sistemi multi-programmati:
  - Più processi vengono eseguiti contemporaneamente:
    - Pseudo-parallelismo (CPU/core singola)
    - Vero parallelismo (multiprocessore)

Necessità di condividere tra i processi le risorse e il tempo CPU per l'esecuzione

Gestione dei processi

- Il kernel gestisce i processi e la loro esecuzione
  - Tabella dei processi
  - Algoritmo di scheduling (“in quale ordine eseguire i processi?”)



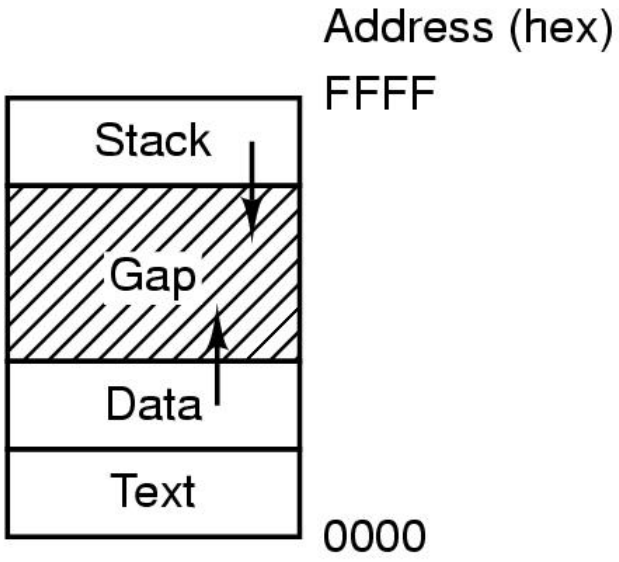
1. Process blocks for input
2. Scheduler picks another process
3. Scheduler picks this process
4. Input becomes available

- I processi possono accedere ai servizi/astrazioni del sistema operativo attraverso delle funzioni specifiche dette chiamate di sistema (system call)
  - Creazione di nuovi processi, arresto di processi esistenti
  - Allocazione memoria
  - Lettura / Scrittura dati (file, directory)
  - Accesso alle periferiche (dischi, rete, stampanti,...)

Gestione della memoria

- I dati e i programmi risiedono su memoria di massa, tipicamente hard disks.
- Per la loro esecuzione i programmi sono trasferiti nella memoria di sistema (RAM).
  - Ogni programma può teoricamente accedere a tutto lo spazio di indirizzamento (es. 2^{32} bytes in un architettura a 32-bit)

Struttura dello spazio di indirizzamento di un processo

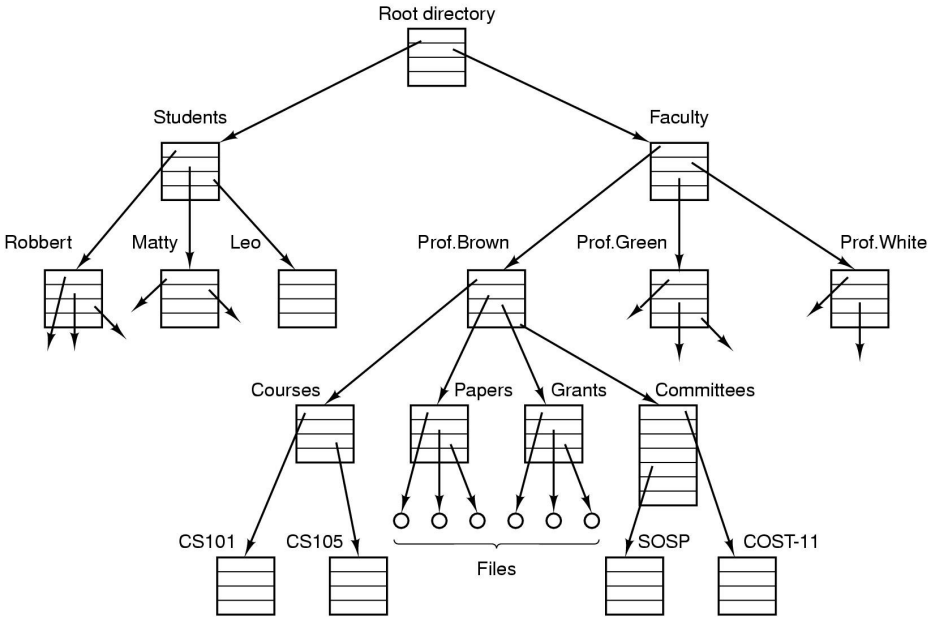


Gestione della memoria

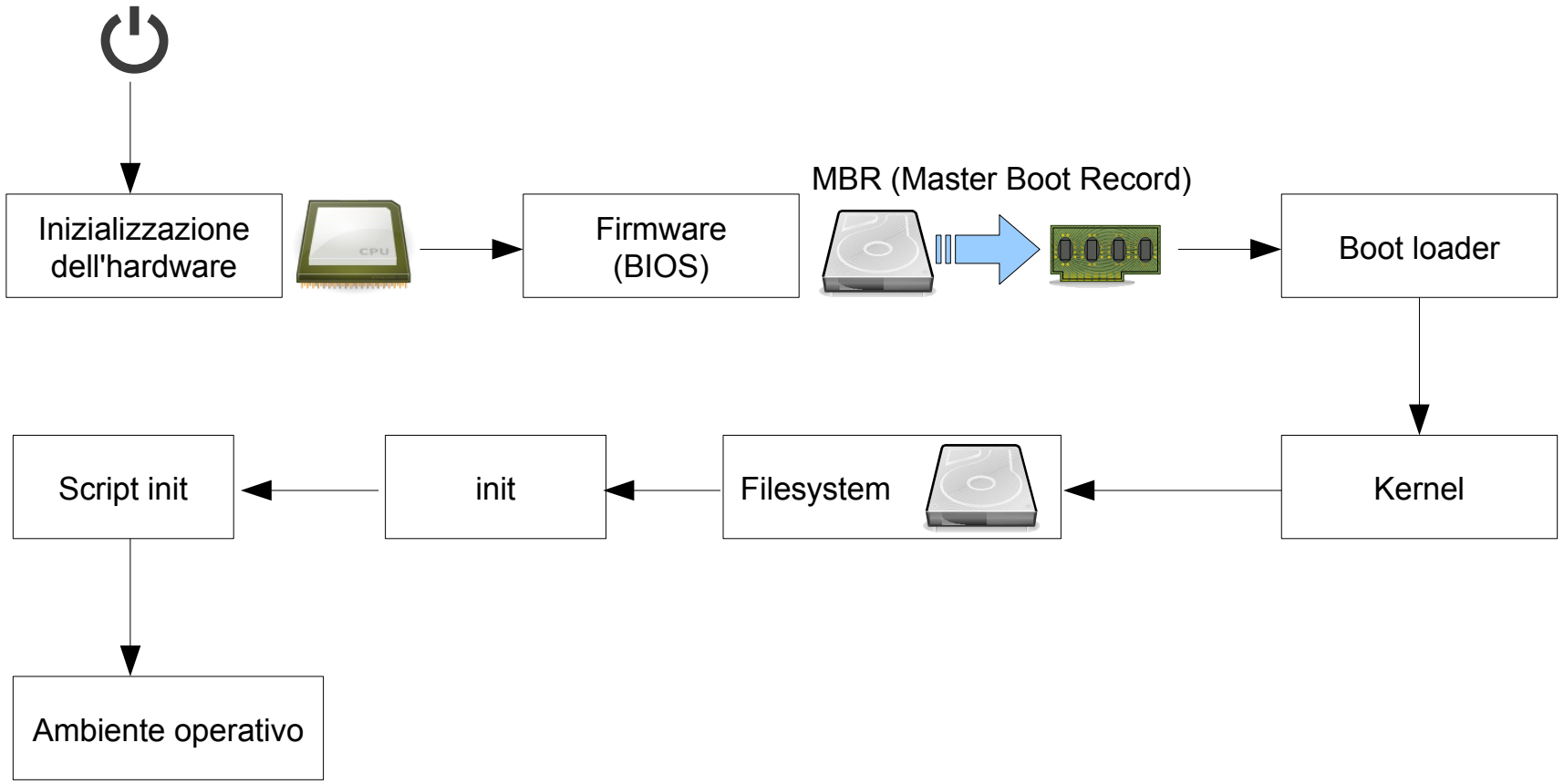
- La memoria è una risorsa limitata che deve essere condivisa tra i programmi in esecuzione
  - Se la memoria fisica non è sufficiente, il sistema operativo può utilizzare una memoria secondaria più lenta ma più capiente (il disco rigido) per spostare temporaneamente dei dati: si parla di memoria virtuale o swap

Gestione del file system

- Memorizzazione dei dati su disco
- Implementazione di una struttura gerarchica
  - Directory
  - Files senza struttura ("byte stream")
- Protezione da accessi non autorizzati

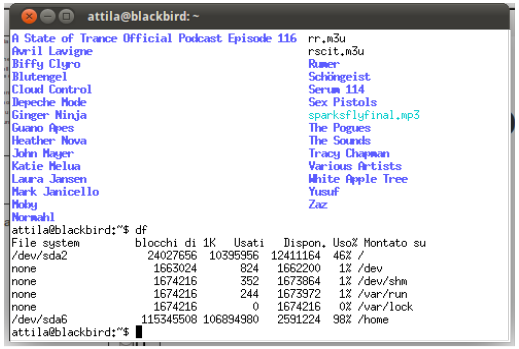


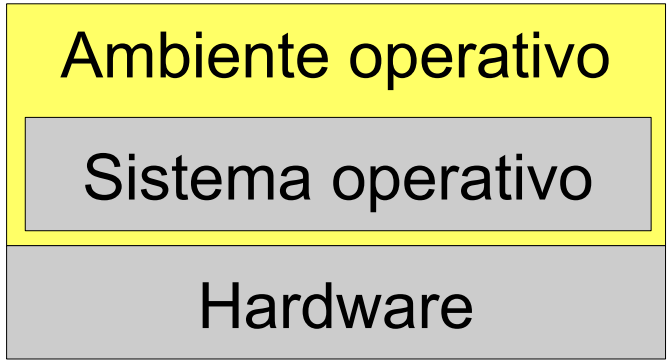
Boot (esempio Unix - semplificato)



Cos'è un ambiente operativo

- Interfaccia tra il sistema operativo e l'utente
  - Richiede un sistema operativo per funzionare
  - Esempi:
    - Shell Unix (su un sistema GNU/Linux, Solaris, o BSD)
    - Ambienti desktop GNOME e KDE (su un sistema GNU/Linux)
    - Windows 3.x (con MS-DOS come sistema operativo)





Cos'è una shell?

- Interfaccia tra il computer e l'utente
  - Testuale
  - Grafica
- Permette di utilizzare le funzioni del sistema operativo (kernel)
  - Gestire i processi
  - Gestire i file
- È un programma che interpreta i comandi

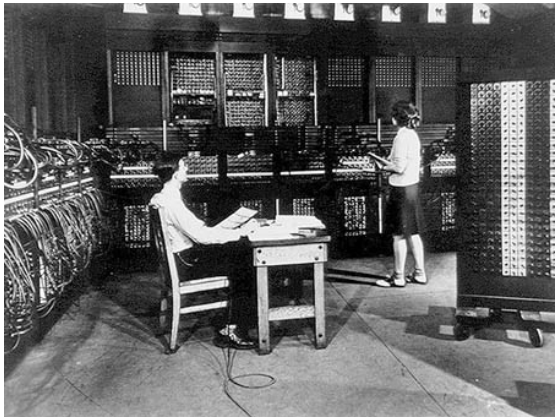
Un po' di storia

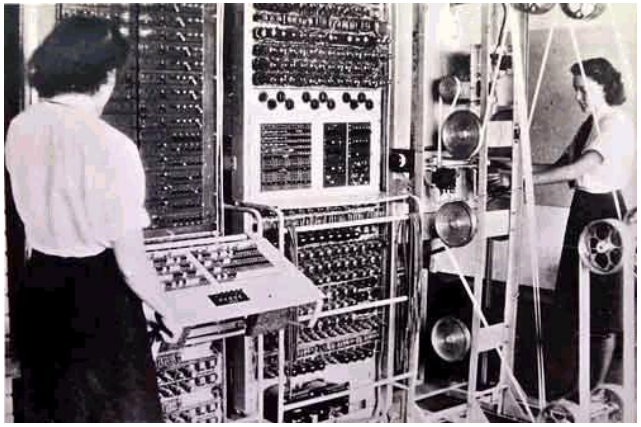
- L’evoluzione dei sistemi operativi
  - è stata spinta dal progresso tecnologico dell’hardware
  - ha guidato il progresso tecnologico dell’hardware
- Perché analizzare la storia dei sistemi operativi?
  - Perché permette di capire l’origine di certe soluzioni presenti nei SO attuali
  - Perché è l’approccio migliore per capire come certe idee si sono sviluppate
  - Perché alcune delle soluzioni più antiche sono ancora utilizzate

Prima generazione: 1945-1955

- Elaboratori elettronici erano progettati con valvole termoioniche
  - Occupavano intere stanze
  - Costosi: soltanto grossi centri di calcolo o università potevano permetterseli
  - Usati solo per calcoli numerici (calcolatori)
  - Inaffidabili (guasti frequenti)

- In questo periodo non esisteva ancora il concetto di sistema operativo
- Venivano programmati in linguaggio macchina (stringhe di 0 e 1)
  - La programmazione avveniva su tavole di commutazione

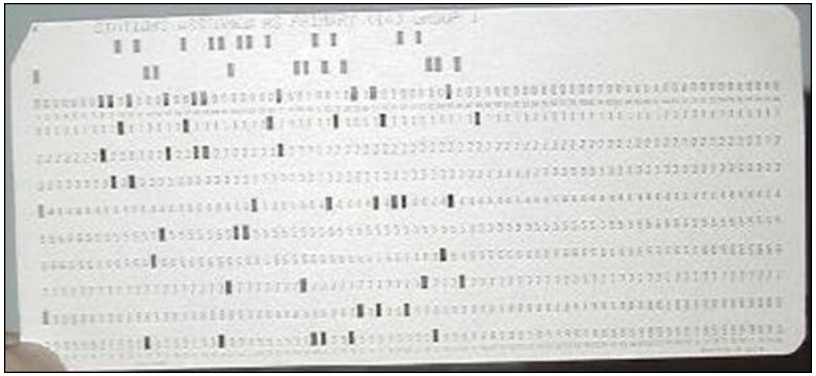


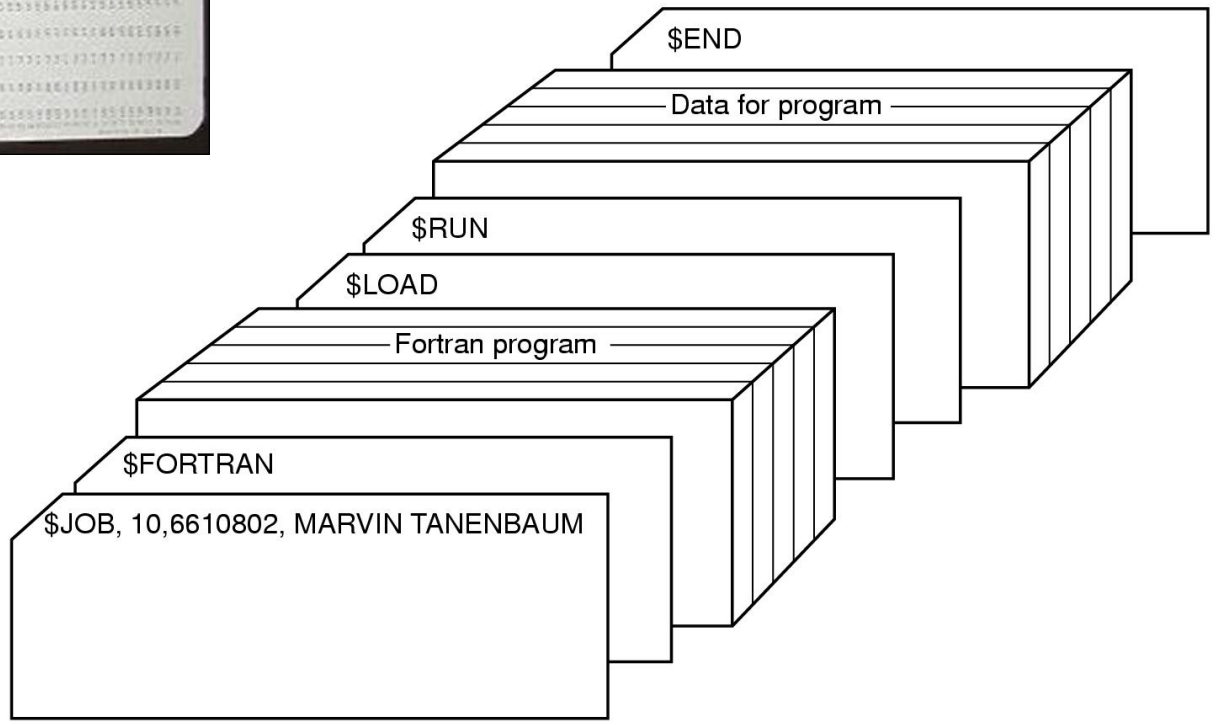


Seconda generazione: 1955-1965

- Transistor
  - Più affidabilità
  - Minor costo
- Programmazione tramite schede perforate
  - Esecuzione di un programma (job):
    - Scrittura del programma su carta
    - Trasferimento su schede perforate
    - Caricamento nel computer
    - Esecuzione e stampa del risultato

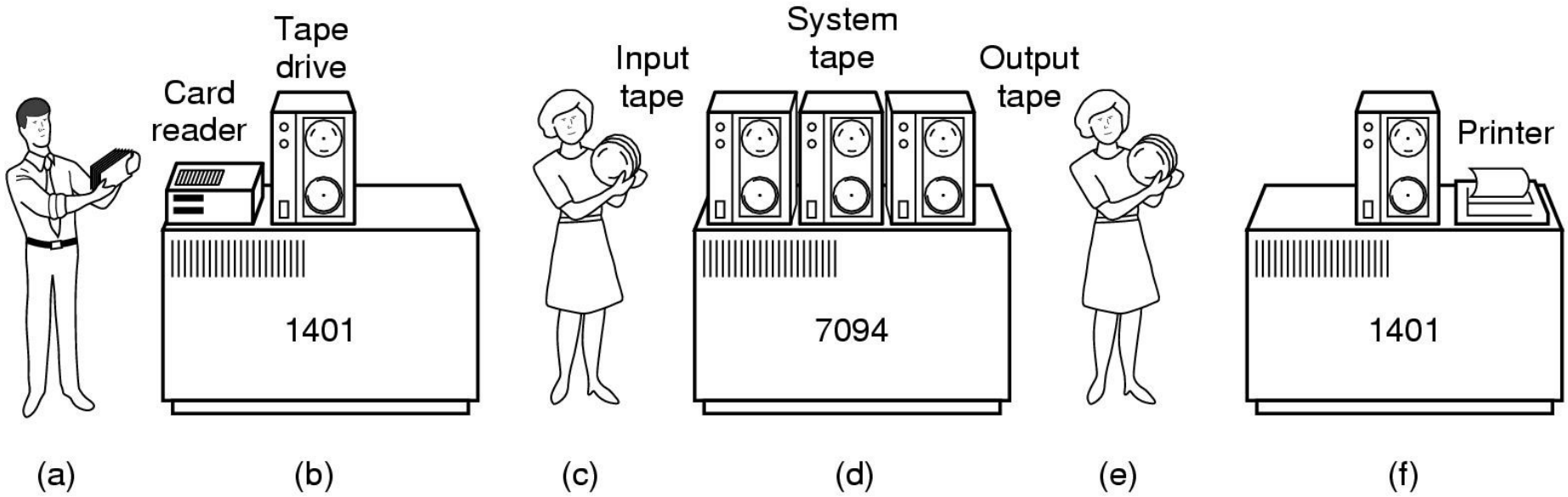
Seconda generazione: 1955-1965





Seconda generazione: 1955-1965

- Sistema batch (a lotti)
  - dividere i tre lavori, ovvero il caricamento dei dati, il calcolo e la stampa su macchine distinte
  - permette di aumentare la capacità di processing del sistema



Seconda generazione: 1955-1965

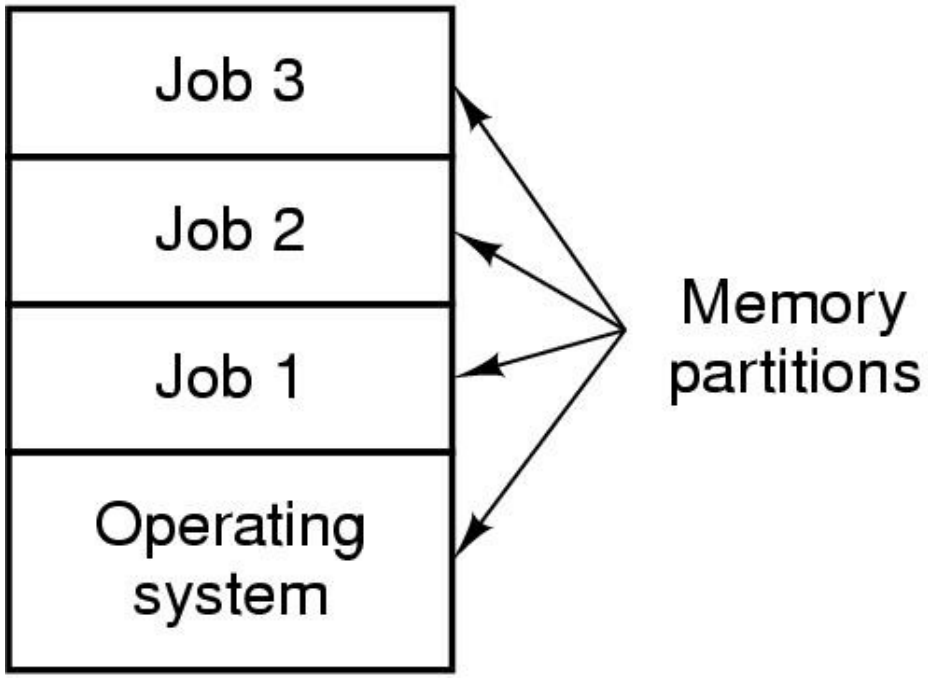
- I sistemi operativi tipici per questi elaboratori, per lo più programmati in FORTRAN e in Assembler erano il FMS (Fortran Monitor System) e IBSYS
- Il sistema operativo residente è in grado di eseguire una sequenza di job, trasferendo il controllo dall'uno all'altro in successione
  - Controllo iniziale del computer al sistema operativo
  - Il controllo viene ceduto al job corrente
  - Una volta terminato il job, il controllo ritorna al sistema operativo

Terza generazione: 1965-1980

- Nel 1964 IBM presenta una famiglia di computer chiamata IBM System/360:
  - Distinzione tra architettura hardware e implementazione
  - Serie di sistemi compatibili
  - Le differenze erano solo nelle performance (memoria, processore, numero di periferiche) e nel prezzo
  - Possibilità di scrivere programmi capaci di funzionare su macchine diverse della serie 360
- Sistemi basati su circuiti integrati
- Per la prima volta è introdotta la multiprogrammazione
  - più programmi in memoria contemporaneamente
  - necessità di hardware specializzato per proteggere i programmi dalle reciproche interferenze

Multiprogrammazione

- Utilizzo del processore durante i periodi di I/O di un job per eseguire altri job
  - Vantaggi:
    - Il processore non viene lasciato inattivo (idle) durante le operazioni di I/O (molto lunghe)
    - La memoria viene utilizzata al meglio, caricando il maggior numero di job possibili
- Per gestire la multi–programmazione, il SO deve gestire un pool di job da eseguire, fra cui alternare il processore



Multiprogrammazione

- Caratteristiche della multiprogrammazione:
  - Più job contemporaneamente in memoria
  - Una componente del SO, detta scheduler, si preoccupa di alternarli nell'uso della CPU
  - Quando un job richiede un'operazione di I/O, la CPU viene assegnata ad un altro job
- Necessita di un sistema che permette alle periferiche di accedere direttamente alla memoria, senza passare per la CPU (Direct Memory Access, DMA)
- Problematiche dei SO multi–programmati:
  - Routine di I/O sono fornite dal SO
  - Il SO deve essere in grado di allocare le risorse di I/O fra diversi processi
  - Il sistema deve allocare la memoria per i job presenti contemporaneamente
  - Il sistema deve scegliere tra i diversi job pronti all'esecuzione (scheduling)

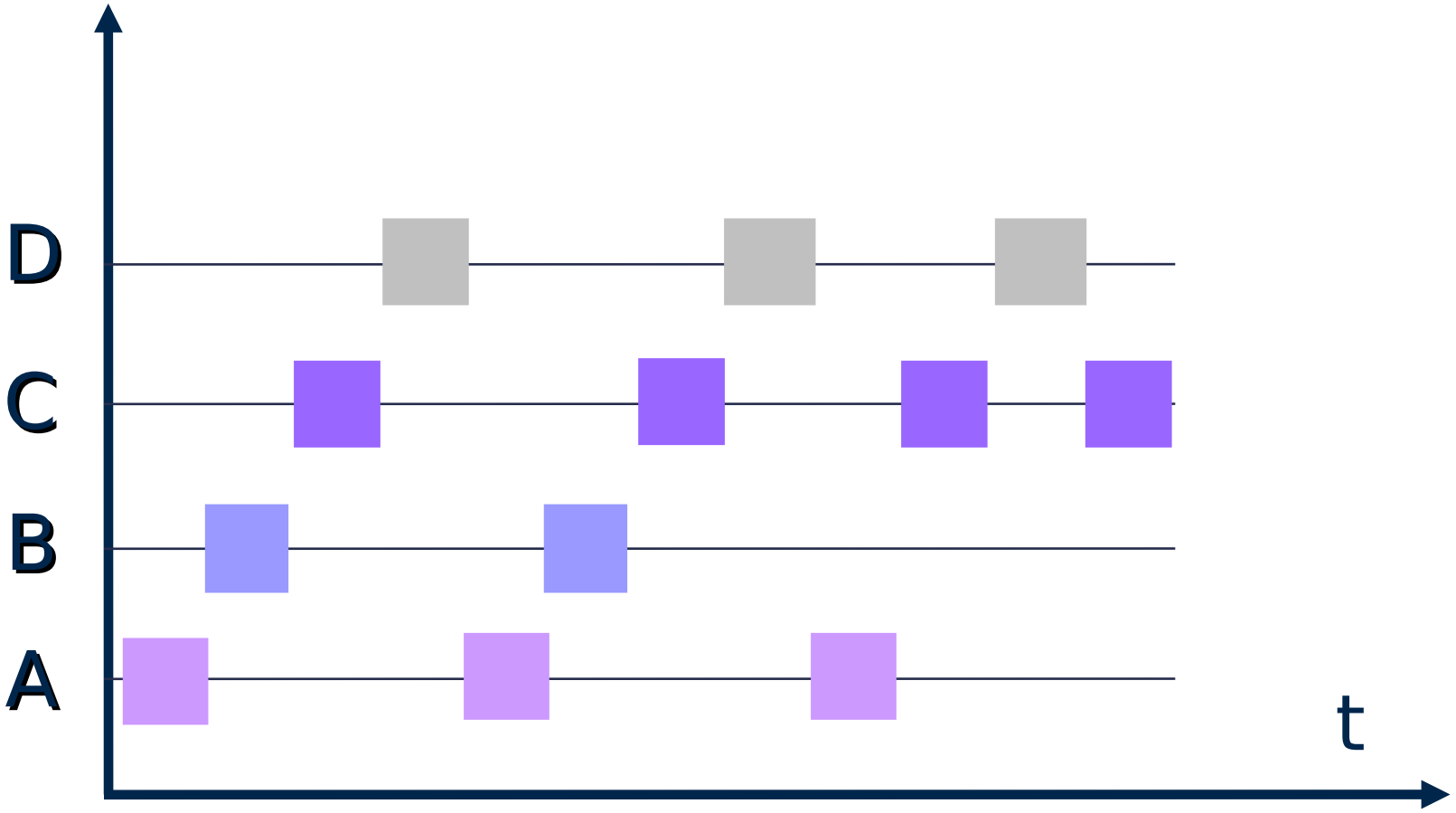
Timesharing

- Il timesharing nasce dalla necessità di controllare / fare il debugging di un'applicazione senza attendere il termine dell'esecuzione (che poteva durare diverse ore)
- Ogni utente dispone di un dispositivo di ingresso (la tastiera) e un dispositivo di uscita (un monitor o stampante)
- Nel 1962 venne realizzato al MIT il primo sistema di timesharing su un IBM 7094 ma la vera rivoluzione si ebbe con il MULTICS Sviluppato congiuntamente dal MIT, dalla General Electric e dai Bell Labs
- Negli stessi anni fu sviluppato il minielaboratore PDP-1 (costava solo120.000$) che ebbe un gran successo
- Per questi sistemi vennero progettati appositi Sistemi Operativi, il più famoso dei quali fu UNIX

Timesharing

- Caratteristiche del timesharing:
  - La risorsa CPU viene suddivisa in quanti temporali; allo scadere di un quanto, il job corrente viene interrotto e l'esecuzione passa ad un altro job, anche in assenza di richieste di I/O
  - I context switch avvengono così frequentemente che più utenti possono interagire con i programmi in esecuzione
- Problematiche dei SO timesharing:
  - Il numero di processi utente può essere molto elevato; si rende necessario l'uso della memoria virtuale
  - Lo scheduling della CPU deve essere di tipo time–sliced, deve cioè sospendere periodicamente l'esecuzione di un programma a favore di un altro
  - La presenza di più utenti rende necessari meccanismi di protezione (e.g. protezione del file system, della memoria, etc.)

Timesharing



Esempio: Unix

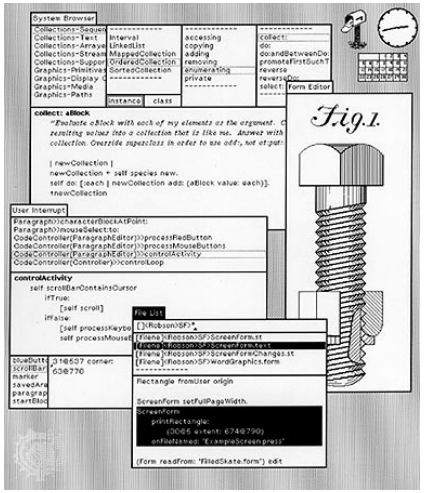
- UNIX fu progettato a partire dal 1969 da un gruppo di ricercatori della AT&T presso i Bell Labs, tra cui erano presenti Ken Thompson, Dennis Ritchie e Douglas McIlroy
- Divenne un sistema molto interattivo, affidabile e ricco di funzionalità, tanto che tuttora è molto presente nel mercato dei server
- Furono sviluppate diverse varianti di UNIX, come il System V, BSD (Berkley Software Distribution), e varie versioni sviluppate da singoli vendors (Solaris, AIX, HP-UX Irix, etc.) e Linux sviluppato dallo studente finlandese Linus Torvalds

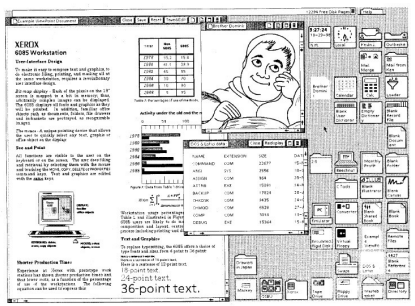
Quarta generazione: 1980-oggi

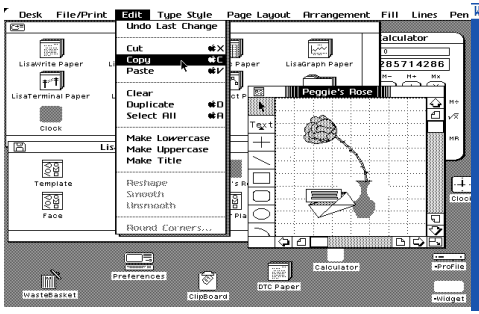
- Tecnologia LSI (large scale integration)
  - chip integrati
  - diminuzione prezzi dell'hardware
  - personal computer
- I personal computer sono dedicati ad utenti singoli:
  - L'obiettivo primario per i SO diventa la facilità d'uso; diminuisce l'interesse per la gestione ottima delle risorse
  - I SO per PC sono in generale più semplici; non implementano la protezione (almeno fino all'avvento di Internet)
  - Creazione di interfacce grafiche user–friendly
  - Tuttavia, tecnologie sviluppate per SO più complessi possono comunque essere adottate
- CP/M-80 della Digital Research per le CPU 8080 / 8085 / Z-80
- MS-DOS (o PC-DOS da IBM)

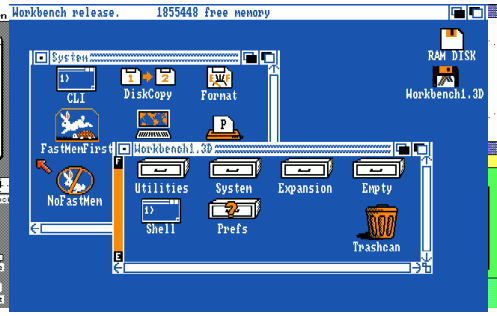
Quarta generazione: 1980-oggi

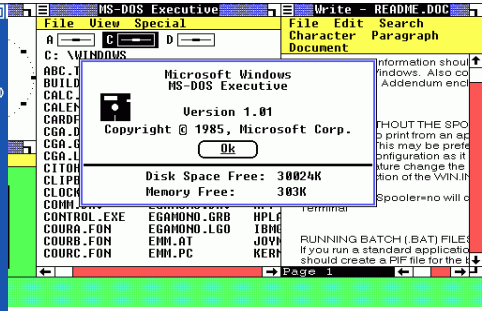
- Interfacce grafiche
  - Xerox Alto (1973) Star (1981)
  - Apple Lisa (1983)
  - AmigaDOS / Workbench (1985)
  - Microsoft Windows (1985)











Quarta generazione: 1980-oggi

- Nuovi scenari
  - Internet
    - sistemi operativi di rete
    - sistemi operativi distribuiti
  - Mobile computing
    - dispositivi handheld (Android, Symbian OS, iOS,...)
  - Microcontrollori (FreeRTOS, Contiki,...)
  - Sistemi multiprocessore/multicore
- Nuove problematiche
  - Sicurezza
  - Efficienza
  - Facilità di utilizzo