Sistemi Operativi (MZ)

COMPLEX INTELLIGENT SYSTEMS RESEARCH GROUP

ComplexIntelligentSystems@gmail.com

http://cis-lab.dmi.unict.it/

Mario Pavone

mario.pavone@unict.it

https://www.dmi.unict.it/mpavone/

https://www.dmi.unict.it/mpavone/so.html

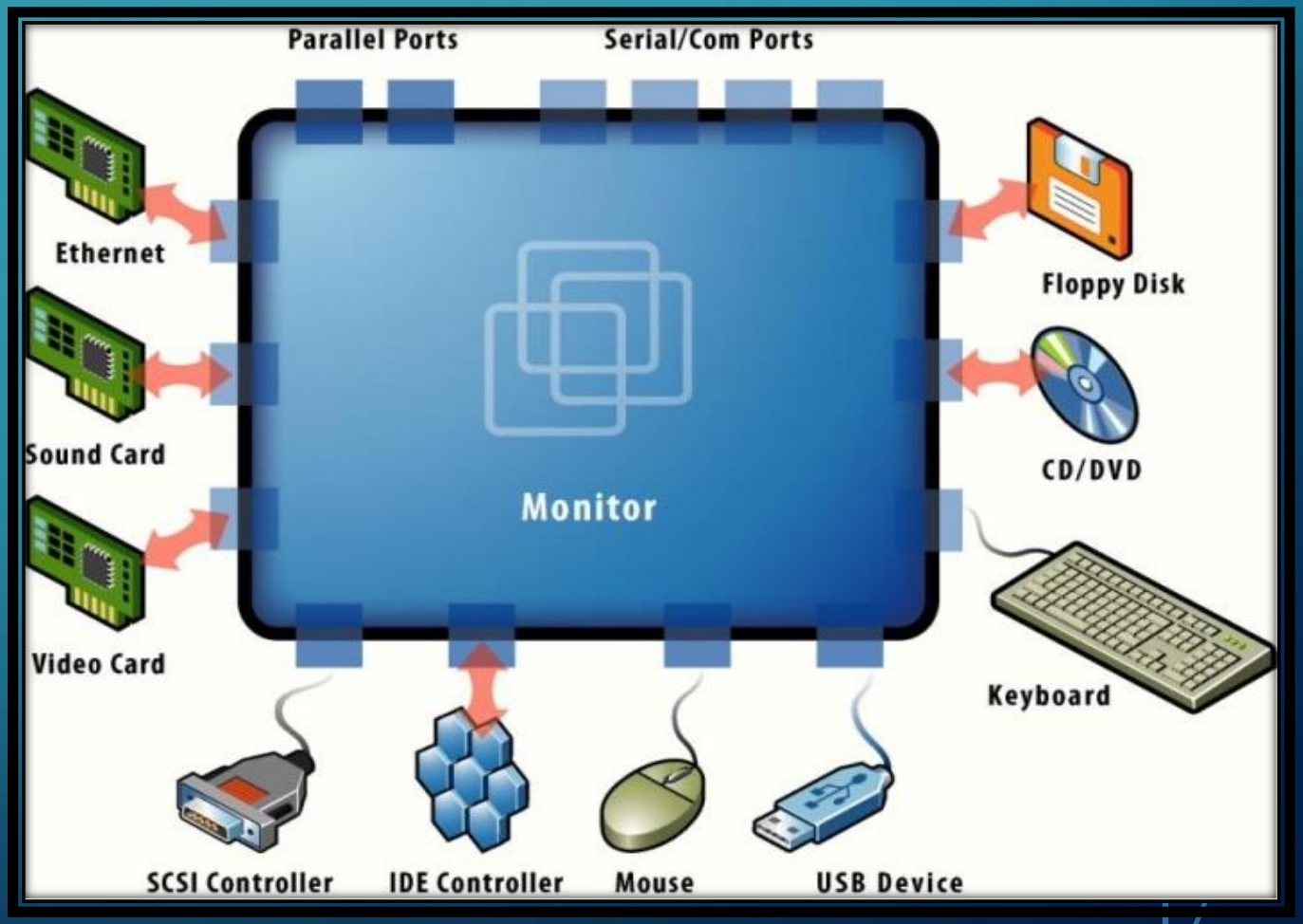
C.d.L. in Informatica

(laurea triennale)

AA 2025-2026

DISPOSITIVI DI I/O

SO interagisce spesso con Dispositivi I/O



DISPOSITIVI DI I/O

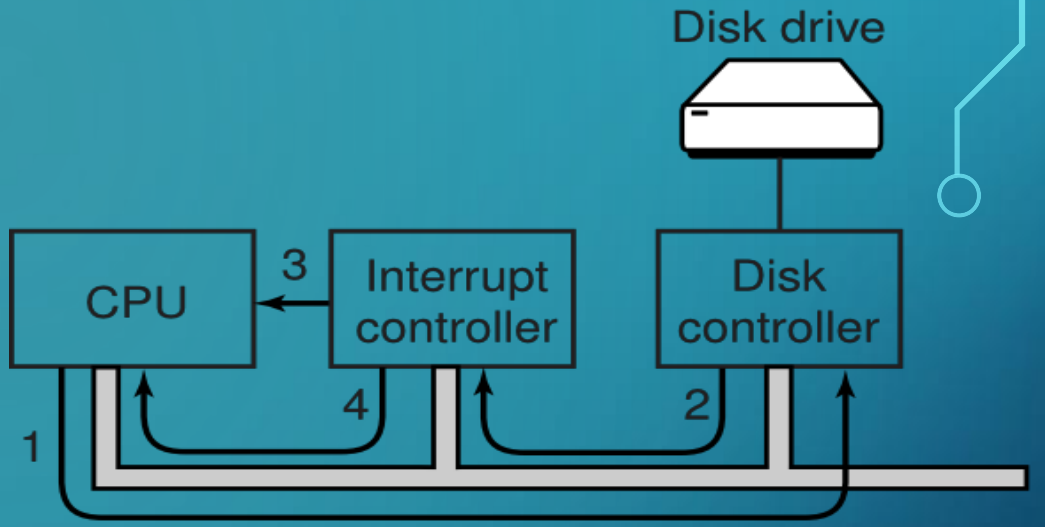
SO interagisce spesso con Dispositivi I/O

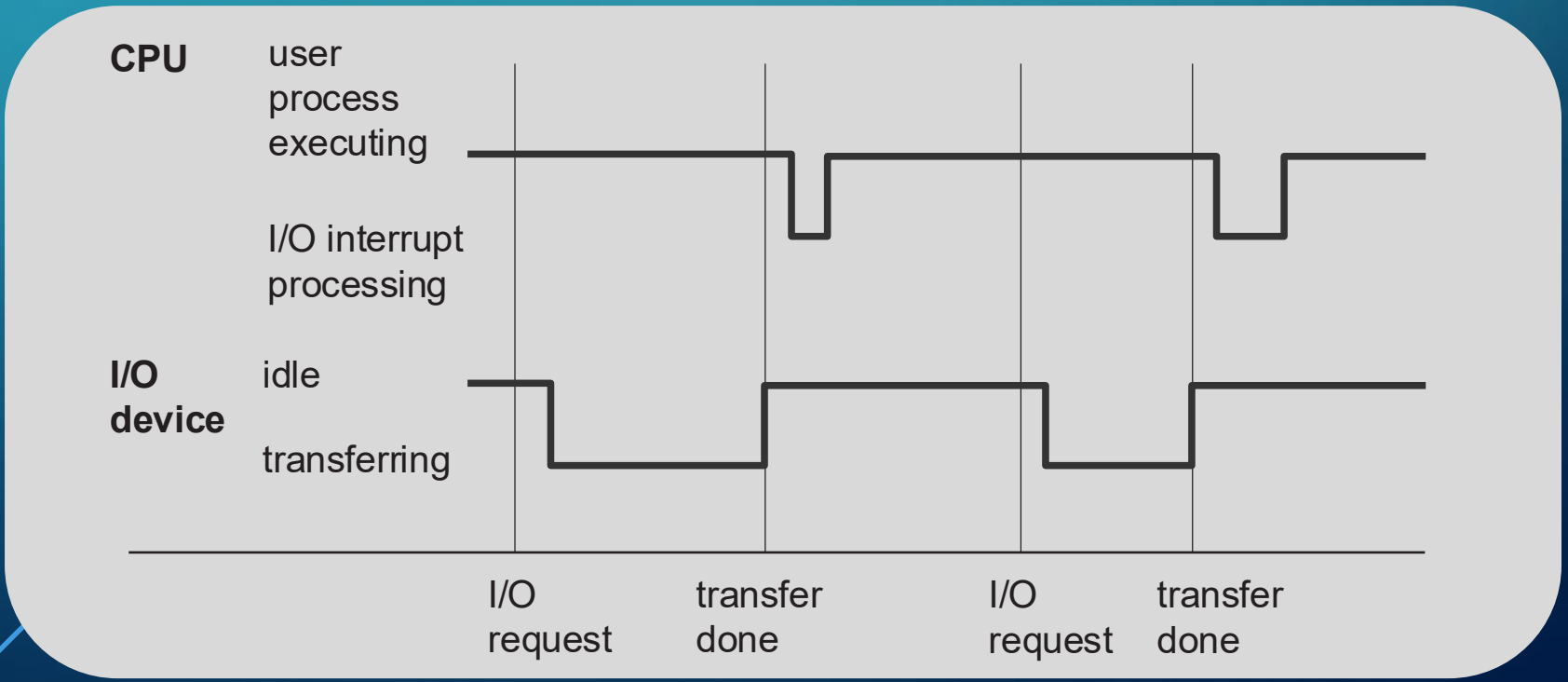
Costituito da due componenti:

- **Controller**: chip o insieme di chip che controlla il dispositivo
  - accetta comandi dal SO
- più semplice da usare per il SO;
- **Dispositivo** stesso: interfaccia elementare ma complicata da pilotare.
- esempio: dischi SATA.
- **DRIVER**: software che comunica con il controller

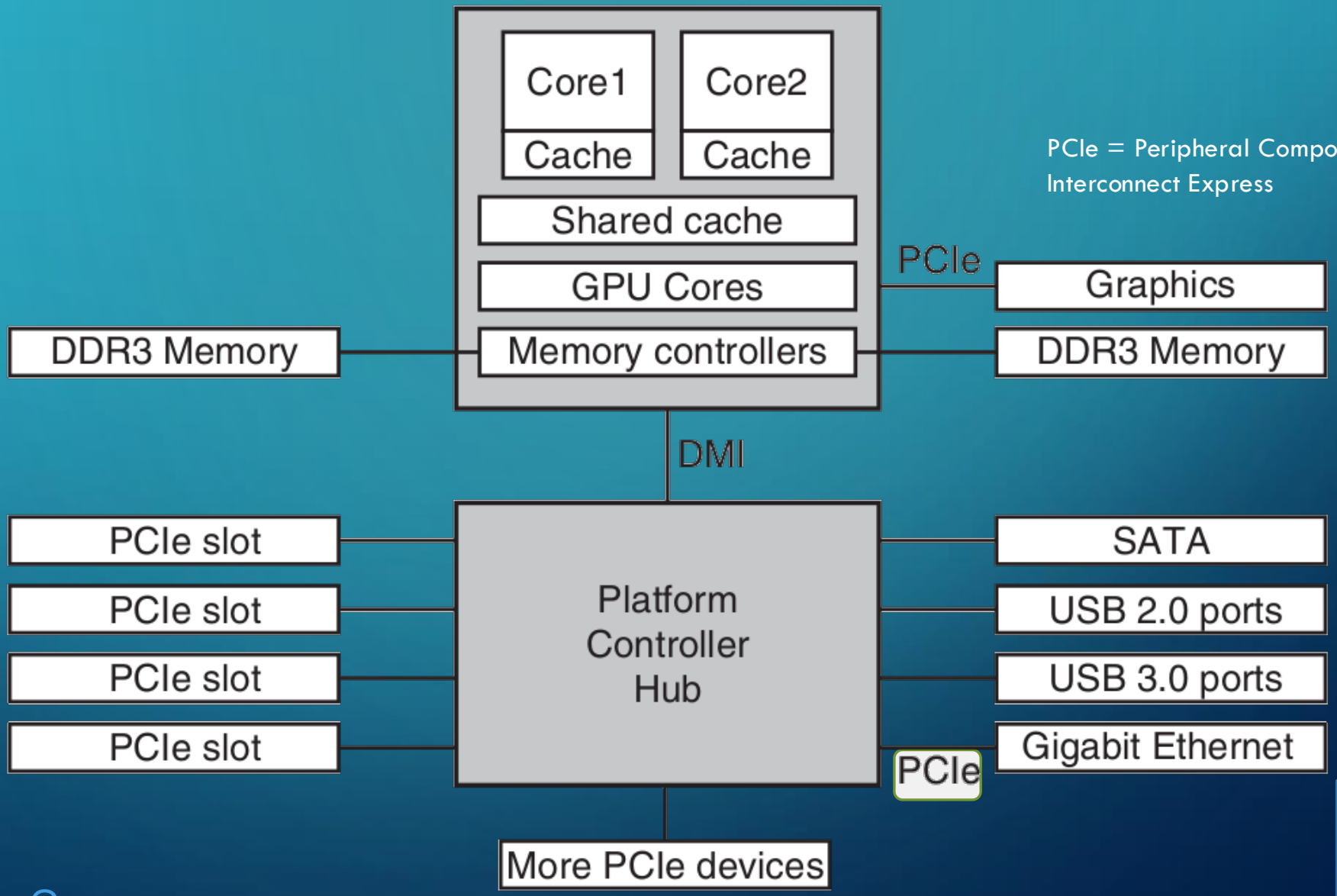
Dispositivi di I/O

- Modalità di I/O:
  - **busy waiting**;
  - con programmazione di **interrupt**;
  - con uso del **DMA**.





Bus



PCIe = Peripheral Component Interconnect Express

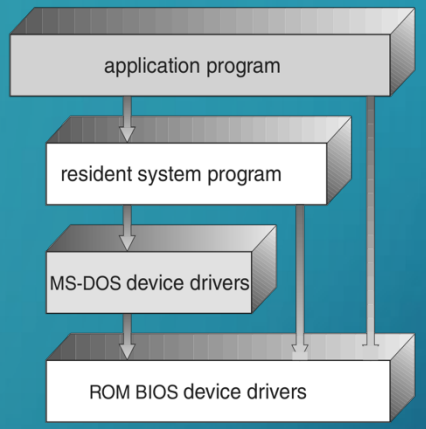
STRUTTURA DI UN SISTEMA OPERATIVO

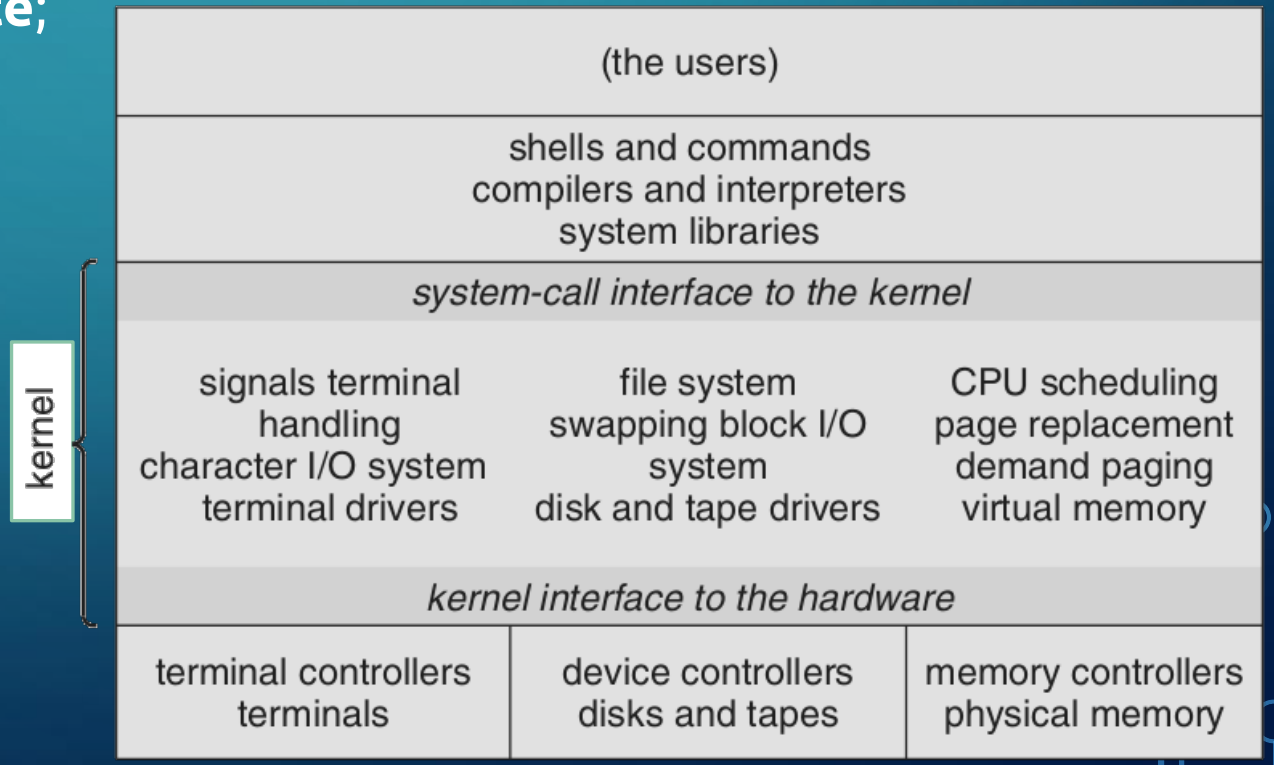
Alcune **possibili strutture** per un SO:

- Monolitici
- A livelli (o a strati)
- Microkernel
- A Moduli
- Macchine virtuali

Struttura monolitica

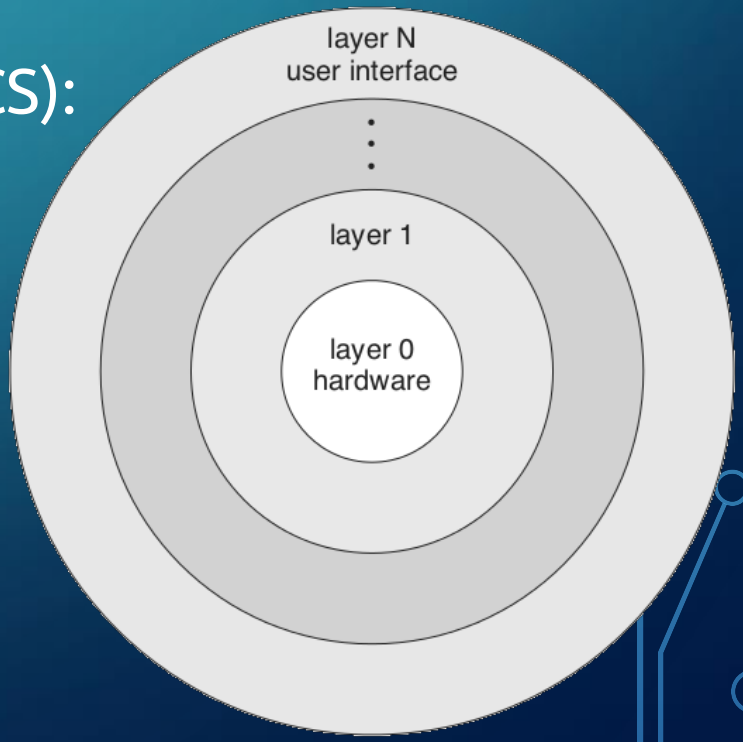
- **Monolitici:**
  - **nessun supporto** hardware;
    - problemi...;
    - esempi: MS-DOS, UNIX;
  - arrivò il supporto hardware alla **modalità kernel/utente**;
    - **unico kernel** con tutto dentro;
    - ogni componente può richiamare tutti gli altri
    - poco gestibile nel tempo.



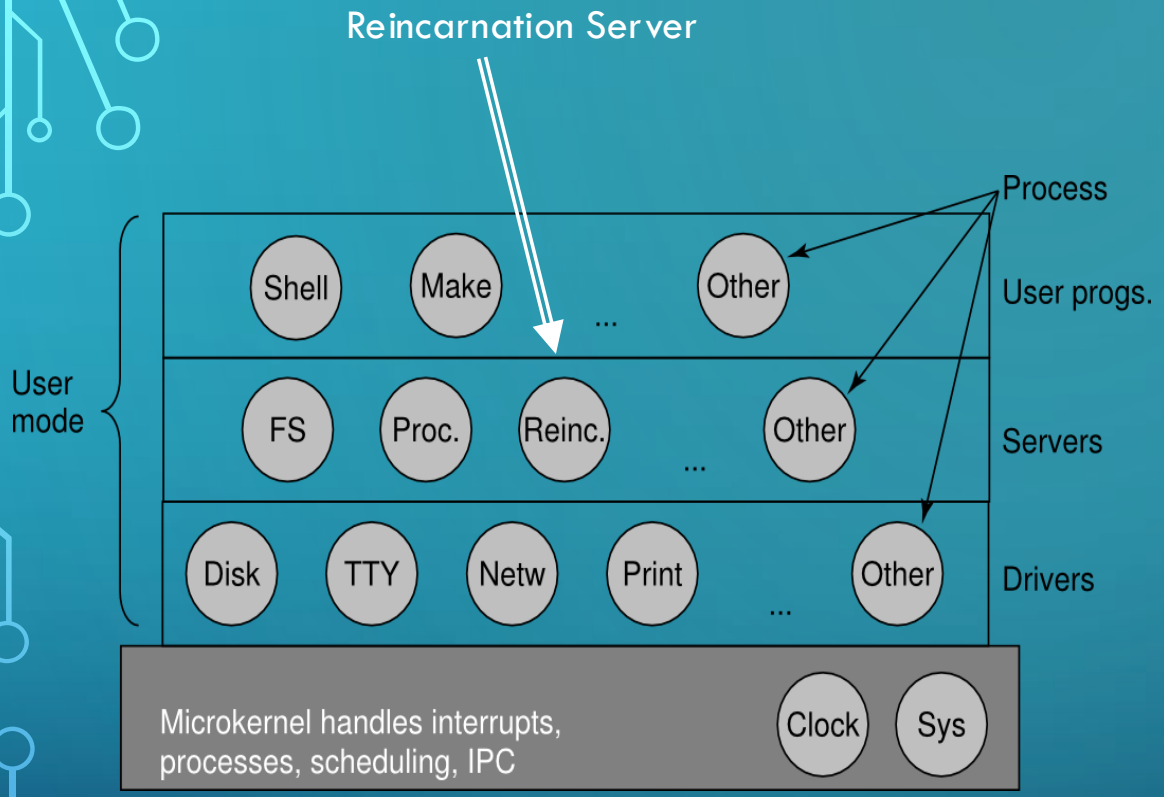


Struttura a livelli (o a strati)

- Si utilizza una **gerarchia di livelli**;
- ogni livello implementa delle **funzionalità** impiegando quelle fornite da quello inferiore;
- **migliore progettazione**, più semplice da sviluppare e controllare (incapsulamento tipo OOP); suddivisione a livello progettuale;
- variante ad **anelli concentrici** (MULTICS):
  - **separazione forzata** dall'hardware;
- **problemi di prestazioni** dovuti alle chiamate nidificate e al relativo overhead.



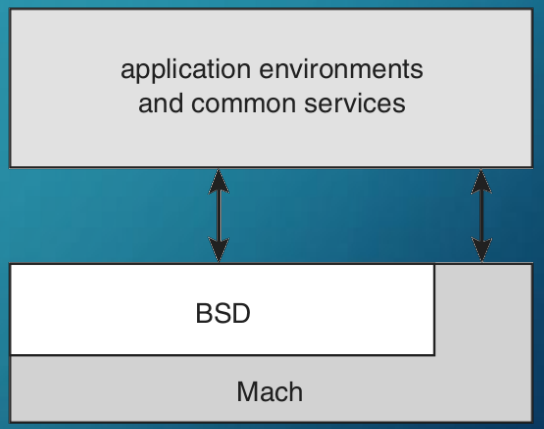
MICROKERNEL

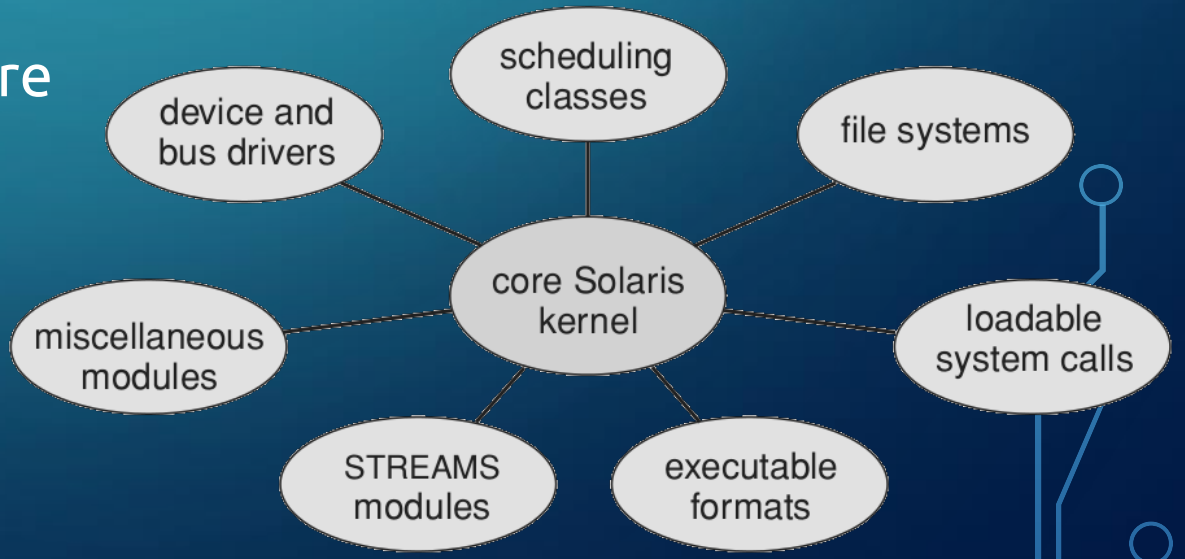


- Uso di un microkernel minimale che si occupa dello scheduling, memoria e IPC;
- tutto il resto gestito da **moduli** (livello utente): filesystem, driver di dispositivi;
- comunicazione attraverso **messaggi**;
- miglior design (componenti piccoli) e migliore stabilità;
- esempi: MINIX 3, Mach, QNX, Mac OS X (Darwin), Windows NT

Struttura a Moduli

- Idea della **programmazione OO** applicata al kernel;
- **moduli** che implementano un qualche aspetto specifico;
  - filesystem, driver,...
- **kernel principale** a funzionalità ridotte;
- moduli **caricabili dinamicamente**;
- design pulito (ad oggetti);
- **efficiente**:
  - ogni modulo può invocare qualunque altro modulo direttamente;
  - niente messaggi;
- esempi: Solaris, Linux, Mac OS X (ibrido).





Macchine virtuali

- L'estremizzazione del concetto di astrazione porta alla **virtualizzazione**;
- **Perchè?** Uso di più SO, VPS, isolamento dei servizi,...
- **Simulazione**, **paravirtualizzazione**.
- Viene tutto gestito dallo **Hypervisor**:
  - **Hypervisor di tipo 1**:
    - gira direttamente sull'hardware;
    - esempi: VMware ESX/ESXi, Microsoft Hyper-V hypervisor;
  - **Hypervisor di tipo 2**:
    - è un processo in un SO Host
    - esempi: VMware Workstation, VirtualBox.
- **Supporto hardware** per efficienza.
- **Java Virtual Machine**.