[[Sistemi operativi - info]]

> Un sistema operativo è un insieme di programmi che operano sull'hardware di un sistema di calcolo. Per l'utente è un'interfaccia verso l'hardware che renda facile risolvere problemi. Per il sistema di calcolo, è il "programma" più strettamente correlato ai dispositivi fisici. Funge da assegnatole di risorse e controllore del loro utilizzo.

Gli **obiettivi principali** di un Sistema Operativo sono: - Semplificare all'utente l'uso e lo sviluppo di programmi - Rendere efficiente l'utilizzo delle risorse hardware - Realizzare un sistema di calcolo che supporti l'esecuzione di programmi utente in modo utile, efficace, efficiente… - Realizzare le funzionalità necessarie a gestire e assegnare le risorse agli utenti, supervisionando le attività

Un sistema di calcolo è composto da: [[Pasted image 20240307145602.png|500]] - dispositivi fisici - CPU, RAM, periferiche - sistema operativo - Gestisce e controlla l'utilizzo di hardware da parte dei programmi - Programmi di sistema - Forniscono interfaccia verso le risorse dal S.O. semplificando lo sviluppo e esecuzione di programmi applicativi (gestione file, config sistema, editor, compilatori) - Programmi applicativi - Applicazioni che risolvono problemi specifici (DBSM, Word processor, Browser…) - Utenti - Qualsiasi agente che usi il sistema di calcolo per risolvere un problema (può essere umano o altro sistema operativo)

Kernel

Programma che ha il completo controllo su tutti i componenti del sistema, responsabile per prevenire e mitigare conflitti tra processi. [[Pasted image 20240307150019.png|200]] Solitamente si definisce il sistema operativo come quel programma sempre in esecuzione nel sistema di calcolo: si usa il termine **Kernel**. Tutto il "resto" è costituito da programmi di sistema o applicativi.

Gestione delle interrupt

- CPU e device operano concorrentemente.
- Un **controller** gestisce uno specifico device e usa un **buffer** locale per interagire con la CPU
- Il controller legge e scrive i dati nel suo buffer locale
- La CPU legge e scrive nel buffer locale del controller
- La sincronizzazione tra controller e CPU avviene tramite le **interrupt**

1. Il segnale di interruzione viene inviato dal controller

2. il processore intercetta il segnale e determina di quale interruzione si tratti
3. Il processore invoca l'esecuzione di una appropriata procedura di servizio per gestire l'interruzione
4. Lo stato della CPU viene salvato e ripristinato al termine della gestione
5. La tecnica più usata sfrutta l'interrupt vector

[[Pasted image 20240307150915.png|400]] ## Gerarchia di memoria 1. Registers 2. Cache 3. Main Memory (RAM) 4. SSD 5. HDD 6. Optical disk 7. Magnetic tapes ### Direct Memory Access Tecnica che permette ai dispositivi I/O di accedere direttamente alla RAM senza coinvolgere la CPU. Quindi il trasferimento di dati da dispositivi I/O a RAM può avvenire senza interrupt dei processi in esecuzione. Causa possibili conflitti di accesso alla memoria Si usa quando è necessario trasferire grandi quantità di dati tra RAM e dispositivi I/O e quando è importante ridurre il trasferimento di dati o quando la CPU è già impegnata con altri processi. [[Pasted image 20240307151845.png#invert|400]]

Sistemi mono e multi programmati

Mono-programmato: - In memoria risiede al più un applicativo, oltre all'OS. - Il sistema esegue un lavoro alla volta Multi-programmato: - In memoria sono caricati molti processi allo stesso tempo, eventualmente anche solo parti di essi = **multiprogrammazione** ### Time Sharing in Multiprogramming - Ad ogni processo viene assegnato un breve quantum di tempo CPU - I processi vengono eseguiti in modo alternato, sfruttando al meglio le risorse di sistema - Richiede un algoritmo di Scheduling per gestire la concorrenza tra processi L'esecuzione sequenziale, invece: - I processi vengono eseguiti uno alla volta - Un processo termina prima che il successivo possa iniziare - Risorse non sfruttate a pieno

[[Pasted image 20240307153240.png|500]]

Processi / Job / Task

> Attività unitaria di elaborazione, caratterizzata da un singolo flusso sequenziale di esecuzione, uno stato corrente ed una collezione di risorse assegnate dal sistema.

> [!example]- Altre definizioni > Un programma caricato in memoria e predisposto per l'esecuzione
>
> > Un'istanza di un programma in esecuzione su un computer
>
> > Un'attività controllata da un programma che può essere assegnata ad un processore e da questo eseguita

- Per ogni utente possono esistere uno o più processi allo stesso tempo in memoria centrale(RAM)

- La memoria centrale può non essere abbastanza per contenerli tutti: si richiede memoria di massa in una zona detta *job pool*
- L'OS decide quali processi caricare in RAM (**job scheduling**) e quali eseguire tra quelli in RAM (**CPU scheduling**)
- L'OS può anche decidere di spostare dei processi (parzialmente eseguiti) dalla RAM alla ROM (**Swap-out**) per liberare RAM. Ovviamente viene congelato e la sua esecuzione viene congelata.

Tutti i processi non possono essere caricati nella RAM allo stesso tempo: vengono spostati nella ROM > ==Job Pool== tramite lo **swap-out** Tramite il **job scheduling** l'OS determina come smistare i processi Tramite la **cpu scheduling** l'OS determina quali processi eseguire tra quelli in RAM

Tecniche a supporto della multiprogrammazione

Swapping

migrazione di processi tra memoria principale e secondaria ### Memoria Virtuale Introduce uno spazio di indirizzamento logico svincolando i processi dalla memoria fisica e dai suoi limiti. (si può eseguire un processo NON completamente caricato su RAM)

Principali funzionalità di un Sistema Operativo

> [!attention] **In presenza di più processi che condividono risorse è necessario garantire che ogni processo non danneggi gli altri** Due modalità di funzionamento per l'OS: - Utente - Kernel (abilità l'esecuzione di istruzioni privilegiate) Nel caso più semplice consiste in un bit di modalità della CPU

> [!question] **Gestione di risorse = istruzioni privilegiate. Come fa un processo utente a ottenere risorse? System Calls.** - Il processo richiede un servizio - La chiamata genera una ==trap== (interruzione). Il controllo passa alla routine di gestione interna all'OS. - La richiesta viene analizzata e il servizio richiesto viene fornito - Si torna in modalità utente - [[Pasted image 20240306172404.png|500]]

Tipi di funzionalità offerte dall'OS:

- Gestione processi
  - Creazione e cancellazione

  - Sospensione e ripristino
  - Comunicazione e sincronizzazione tra processi
  - Gestione stallo
- Gestione RAM
  - Tenere traccia delle porzioni utilizzate
  - Decidere quali processi caricare in memoria
  - Assegnare / revocare spazio ai processi in base alle necessità
- Gestione ROM
  - Informazioni organizzate in file, organizzati in file system.
  - Creazione / cancellazione, accesso
  - Gestione affidabilità (backup)
  - Gestione / assegnazione di spazio libero
  - Scheduling del disco
- Gestione I/O
  - Gestire i ==buffer== = zone di memoria dove scambiare i dati con i dispositivi
  - Gestire il ==caching== = spostamento temporaneo di dati nella cache
  - Gestire lo ==spooling== = esecuzione asincrona di processi I/O (lenti)
- Protezione e sicurezza
  - Protezione = controllo dell'accesso alle risorse di sistema da parte di processi o utenti
  - Sicurezza = difesa da accessi / operazioni dannose
  - Si impiegano **user IDs**, **Grop IDs** assegnati a utenti e processi.