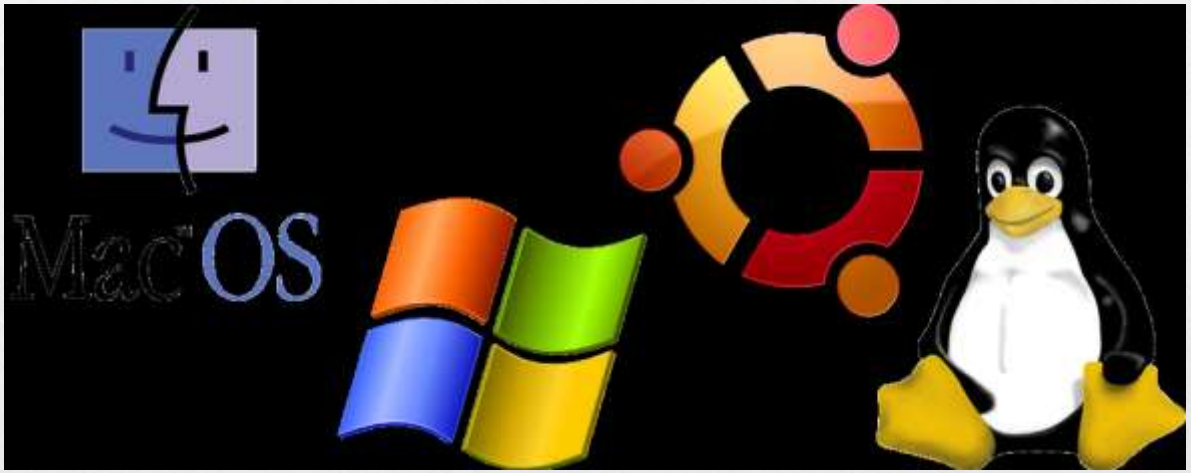
ELEMENTI BASE DEI SISTEMI OPERATIVI.

- Caratteristiche Generali.



Si definiscono programmi ***user friendly*** quei software facili da usare ovvero poter utilizzare il PC senza ricorrere a consulenze tecniche.
Si definisce ***software house*** l'insieme delle *aziende* produttrici di software, il cui investimento in ricerche sempre migliorativo è un obiettivo finalizzato alla garanzia del vantaggio competitivo di mercato.

Il software che permette un utilizzo più semplice, attraverso strumenti **visivi viene esplicitamente indicato con il termine di sistema operativo.**


Sistema
operativo =
Macchina
virtuale

Il ***Sistema Operativo (O.S. Operating System) è un insieme di programmi che governa e controlla l'uso delle componenti del sistema di elaborazione ed ha lo scopo di renderli disponibili all'utente per un utilizzo trasparente ed efficace.***

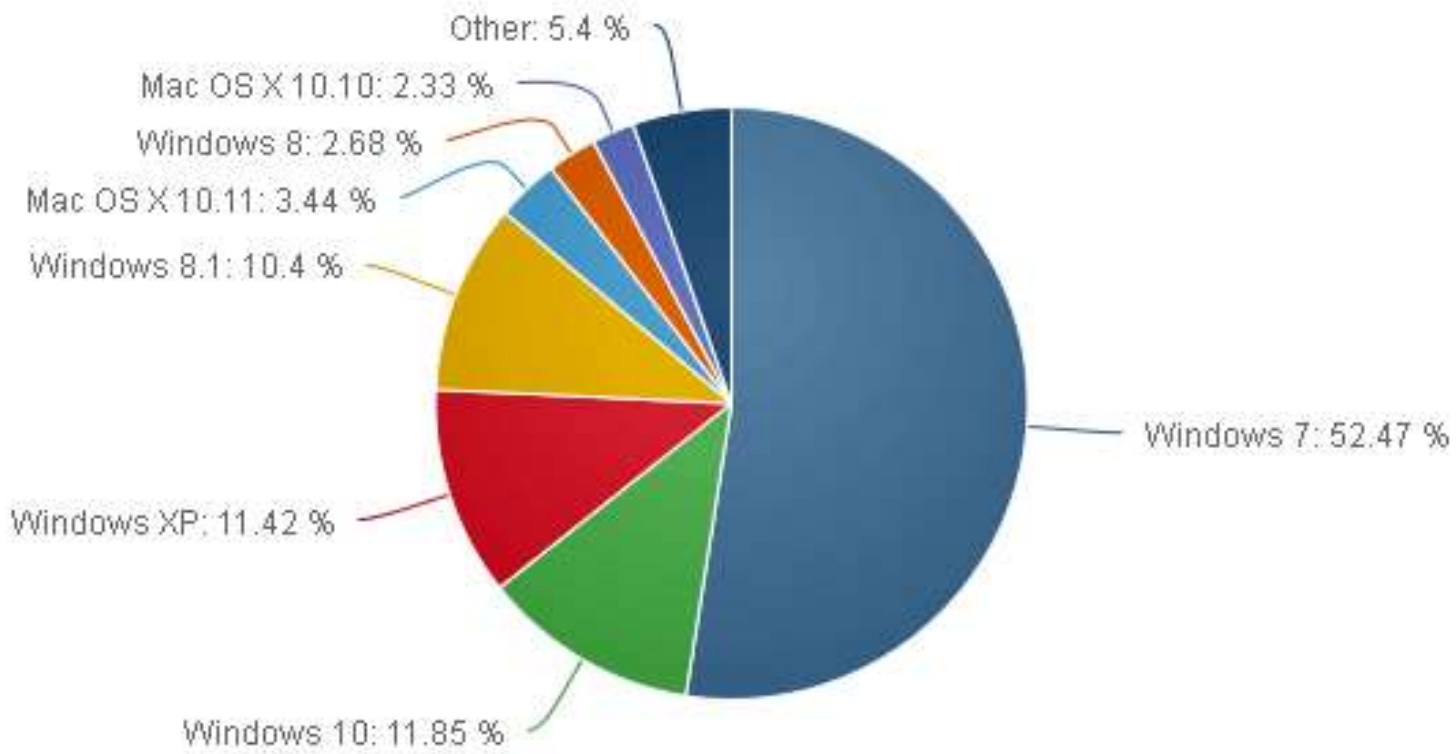
Con il termine ***risorsa*** *si intende un elemento hardware (CPU, Memoria Centrale, Unità di memorie, Unità Periferiche) o software (Dati, programmi)*

Il Sistema Operativo può essere inteso ***come aiuto verso l'utente*** *attraverso i programmi di utilità: editor, compilatori, ecc. ecc. GUI, device, plug and play e come* ***gestore delle risorse****: gestione della memoria, gestione della cpu, gestione di i/o e gestione del file system)*

Un Sistema Operativo si può pensare come gli impiegati di un'azienda, ciascuno dei quali ha dei compiti specifici da svolgere e tutti insieme lavorano per raggiungere gli obiettivi prefissati dalla direzione e propagati attraverso i vari livelli dell'organizzazione.
Quando un utente interagisce con il computer vengono emanate richieste di utilizzo di risorse che riceveranno dopo essere state interpretate dai vari programmi organizzati in modo gerarchico.

Di cosa stiamo parlando?

Sistemi operativi per PC più diffusi?



Sistemi operativi per dispositivi mobili

Smartphone OS Sales Share (%)

Germany	3 m/e Mar 2014	3 m/e Mar 2015	% pt. Change
Android	77.0	71.3	-5.7
iOS	15.3	18.3	3.0
Windows	6.6	8.7	2.1
Other	1.1	1.7	0.4

USA	3 m/e Mar 2014	3 m/e Mar 2015	% pt. Change
Android	57.9	58.1	0.2
iOS	36.7	36.5	-0.2
Windows	4.4	4.3	-0.1
Other	1.0	1.1	-0.1

GB	3 m/e Mar 2014	3 m/e Mar 2015	% pt. Change
Android	57.7	52.9	-4.8
iOS	31.2	38.1	6.9
Windows	9.2	8.0	-1.2
Other	1.9	1.0	-0.9

China	3 m/e Mar 2014	3 m/e Mar 2015	% pt. Change
Android	80.0	72.0	-8.0
iOS	17.9	26.1	9.2
Windows	1.2	1.2	0
Other	0.9	0.7	-0.2

France	3 m/e Mar 2014	3 m/e Mar 2015	% pt. Change
Android	65.2	64.6	-0.6
iOS	23.4	19.4	-4.0
Windows	8.3	14.1	5.8
Other	3.1	1.9	-1.2

Australia	3 m/e Mar 2014	3 m/e Mar 2015	% pt. Change
Android	57.3	52.3	-5.0
iOS	33.1	38.4	5.3
Windows	6.9	7.3	0.4
Other	2.7	2.0	-0.7

Italy	3 m/e Mar 2014	3 m/e Mar 2015	% pt. Change
Android	70.7	66.2	-4.6
iOS	12.9	17.5	4.6
Windows	13.9	14.4	0.5
Other	2.5	1.9	-0.6

Japan	3 m/e Mar 2014	3 m/e Mar 2015	% pt. Change
Android	42.1	52.3	10.2
iOS	57.6	45.1	-12.5
Windows	0.3	0.4	0.1
Other	0.0	2.2	2.1

Il Sistema Operativo racchiude alcuni di questi livelli di organizzazione e, rende disponibile il computer per le esecuzioni di azioni complesse.

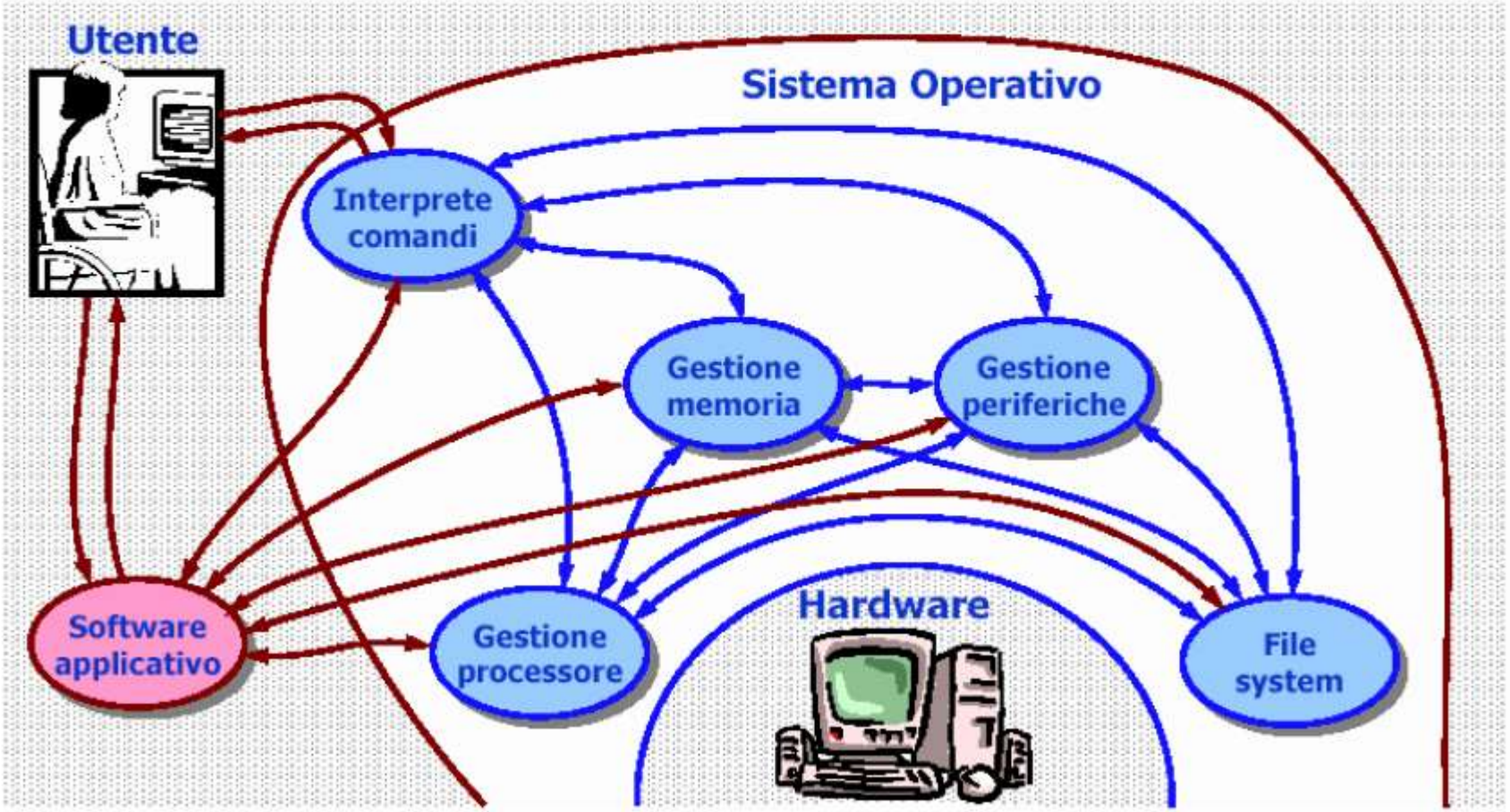
I programmi del Sistema Operativo che interagiscono con le componenti hardware devono conoscere le caratteristiche dei dispositivi in modo da poterli controllare ed adoperare.

è opportuno quindi che i programmi possano essere ampliati, aggiornati e personalizzati a seconda della macchina su cui verrà installato il Sistema Operativo che comunque deve essere in grado di accedere anche a dispositivi non previsti inizialmente purché questi vengano forniti con il corredo del software necessario, comunemente chiamato *device driver* (pilota del dispositivo).

Sugli ultimi Sistemi Operativi in commercio (Windows, Linux) queste operazioni vengono svolte in modo automatico con percorsi di autoinstallazione o comunque senza intervento di personale specializzato.

I dispositivi realizzati in tecnologia *plug and play* (inserisci e usa) interagiscono con il sistema facendosi riconoscere e installando i driver corretti. Il Sistema Operativo sarà in grado di configurare tali dispositivi evitando di creare conflitti con i dispositivi preesistenti e che tutto il sistema funzioni correttamente.

ELEMENTI DI UN S.O.



SISTEMA APERTO

Un **sistema aperto** è un sistema operativo costruito in modo da poter operare con una molteplicità di dispositivi, che possono essere reperiti sul mercato senza dover fare riferimento ad una ristretta cerchia di produttori.

Caratteristiche più importanti sono: possibilità di controllare macchine tra loro molto diverse, costruite addirittura su unità centrali non compatibili.

Inoltre un sistema operativo aperto permette l'uso di dispositivi hardware standard senza imporre particolari condizioni oltre alla scrittura di una quantità piccola di software. Un esempio di sistema operativo aperto è il sistema operativo LINUX.

SISTEMA PROPRIETARIO

Un ***sistema proprietario*** è un sistema operativo costruito per essere eseguito su un insieme ristretto e ben specifico di macchine, corredate o meno di specifici dispositivi.

Caratteristiche più importanti sono: soddisfano le esigenze specifiche dell'utente in modo molto efficace, il servizio offerto dalla casa proprietaria è sufficiente a coprire tutte le esigenze dell'utenza a costi ragionevoli. Un esempio di sistema operativo proprietario è il sistema operativo Windows.

Quale di questi è un sistema aperto?

POLITICHE DI GESTIONE

Il Sistema Operativo essendo un insieme di programmi che sovrintende al corretto funzionamento delle apparecchiature che costituiscono il sistema di elaborazione, soddisfa le richieste dell'utente;
esse vengono acquisite dal Sistema Operativo e gestite in modo coordinato, rispettando le cosiddette politiche di gestione.

Le politiche di gestione sono delle ***regole*** scelte per una gestione razionale delle risorse ovvero un codice di comportamento del sistema di elaborazione:

esse indicano al calcolatore come reagire agli stimoli e come rispondere alle richieste dell'utente, stabilendo gerarchie e priorità con le quali ogni richiesta deve essere servita.

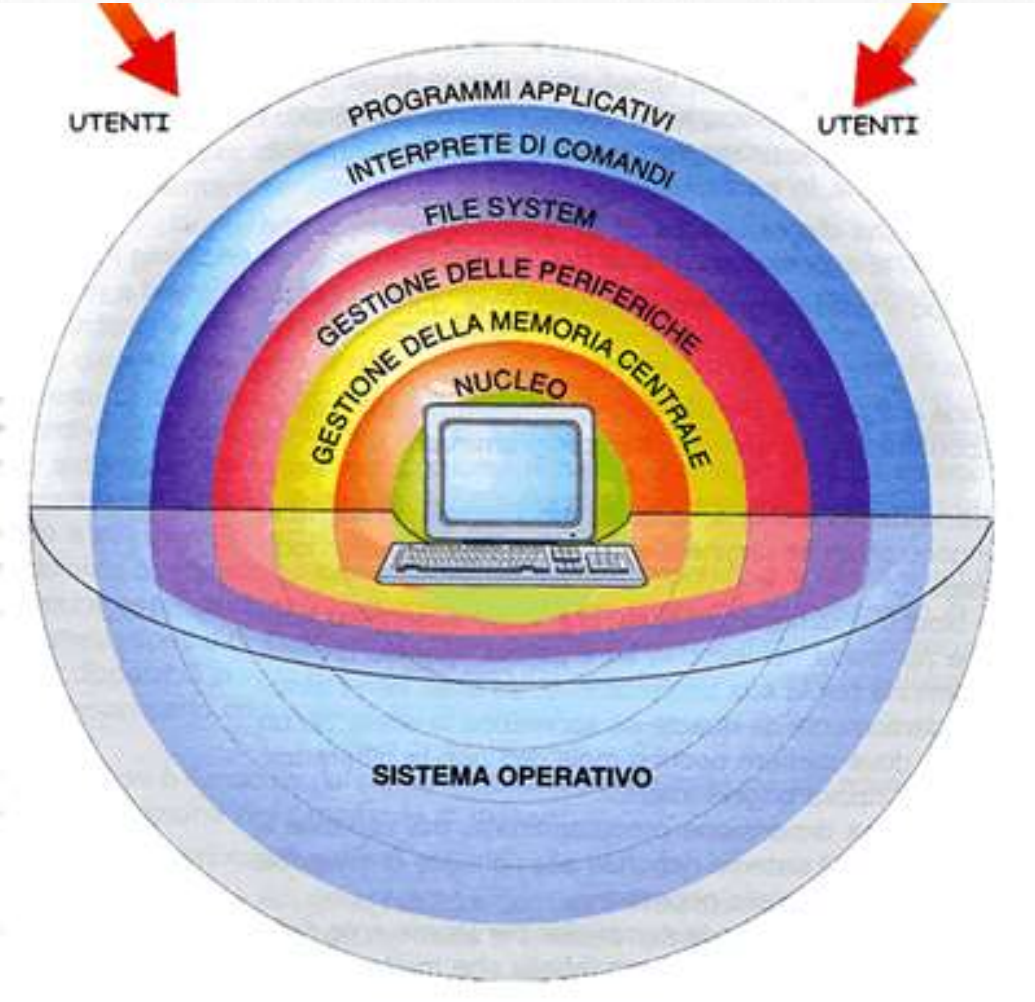
La struttura formale di un Sistema Operativo è gerarchica: ciò significa che i programmi che lo compongono si collocano a livelli diversi, si servono di programmi che stanno a livelli sottostanti e servono i programmi dei livelli superiori.

Si può pensare come una struttura a cipolla, cioè l'insieme dei dispositivi fisici che costituiscono il computer come ad un livello più basso, e all'utente come al livello più alto: il sistema operativo contiene i programmi tra il livello più basso (vicino alla macchina) e il livello più alto (molto vicino all'utente).

All'interno di ogni livello, si usa raggruppare tutti i programmi scritti per risolvere un problema, o una classe di problemi simili, in *moduli*.

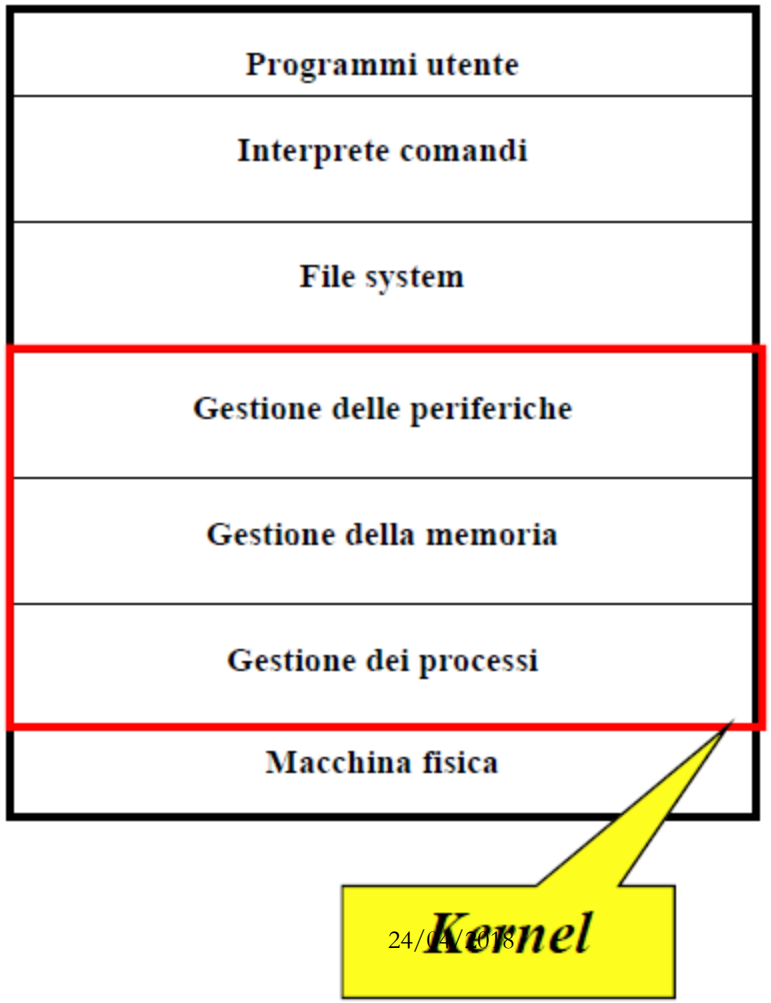
Schematizzazioni di sistemi operativi

Modello a strati di un S.O.



Architettura del sistema operativo

- Il SO è tipicamente organizzato a strati
- Ciascun strato costituisce una *macchina virtuale* che gestisce una risorsa del calcolatore
- Le principali funzionalità offerte sono:
  - La gestione dei processi
  - La gestione della memoria
  - La gestione delle periferiche
  - La gestione del file system
  - La gestione della rete
  - La gestione dell'interfaccia utente
- Le prime tre funzionalità sono indispensabili per il funzionamento del sistema e pertanto costituiscono il nucleo del SO (Kernel)



Ogni modulo si può pensare come una scatola nera (black box) che al suo interno contiene le competenze per risolvere un problema e presentare i risultati in conseguenza dell'elaborazione di richieste. Quel che avviene all'interno della scatola non ha alcuna importanza per chi effettua la richiesta.

Per i programmi che fanno uso del modulo, quel che conta è che continui a rispondere coerentemente alle richieste effettuate attraverso procedure standard indipendentemente se il modulo sarà modificato o riscritto.

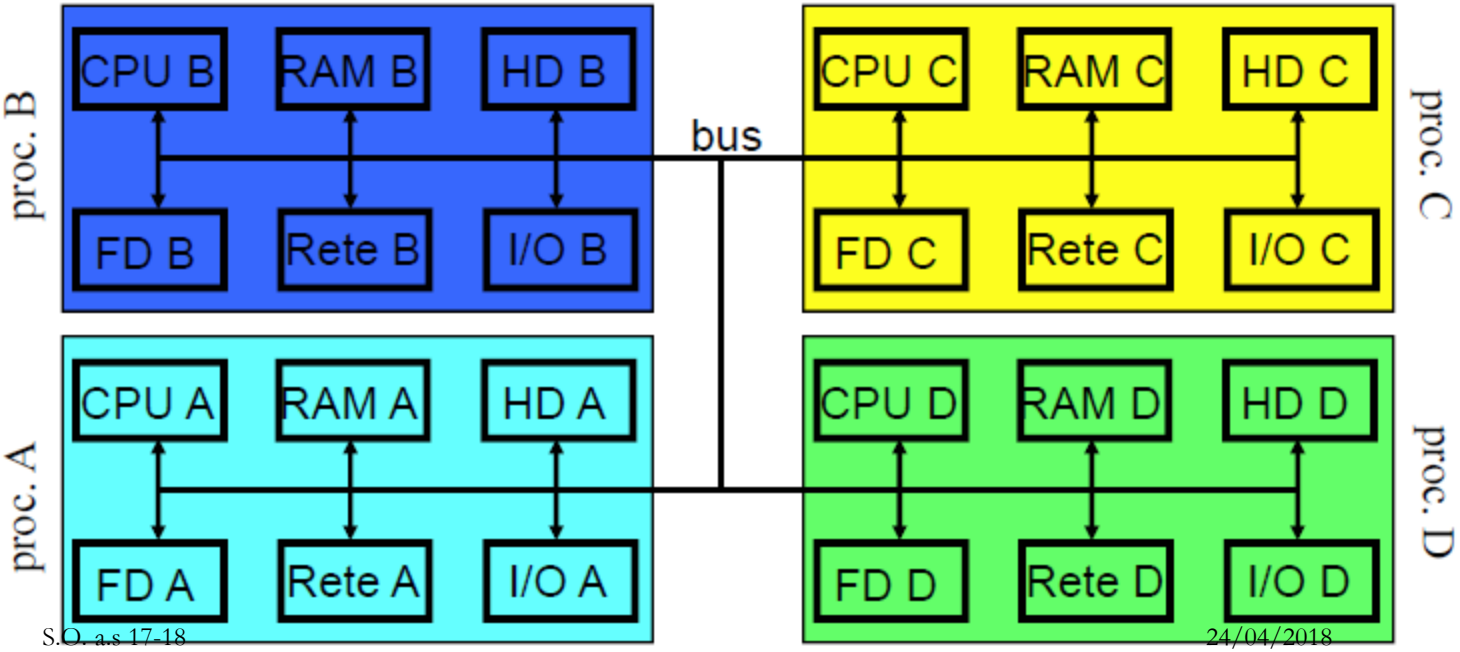
I moduli, definiti in modo logico, trasformano i dispositivi in macchine astratte o macchine virtuali in grado di eseguire operazioni complesse come se fossero elementari.

Un'operazione logica corrisponde all'esigenza dell'utente di ottenere un risultato.

Esempio scrivere un testo per stamparlo; la corrispondente operazione fisica consiste nell'eseguire molte operazioni elementari dipendenti dal particolare elaboratore e stampante sui quali è stata richiesta l'operazione logica. La stampante ad aghi utilizza una tecnologia diversa da quella a getto d'inchiostro o laser ma l'operazione eseguita è concettualmente uguale.

Il sistema operativo e le macchine virtuali

- Il sistema operativo esegue più processi contemporaneamente
- Rende quindi visibile ad ogni processo una macchina virtuale ad esso interamente dedicata e quindi con risorse proprie



L’aspettativa dell’utente di stampare il documento viene trasformata in richieste elementari al dispositivo e il sistema si preoccupa che ognuno delle richieste vada a buon fine gestendo le varie situazioni di errori che si possono presentare, senza che l’utente debba intervenire.

I moduli, alle quali le primitive appartengono inoltrano le richieste ai moduli di livello superiore utilizzando le loro primitive fino ad raggiungere l’hardware che esegue materialmente l’operazione:

i circuiti della stampante trasformano i comandi specifici ricevuti in sequenze di impulsi elettrici che attivano i dispositivi elettrici o meccanici coinvolti nell’operazione: nel caso di stampante ad aghi vengono spinti contro il nastro i sottili aghi mentre per la stampante laser, il laser traccia sul tamburo fotosensibile l’immagine della pagina.

I comandi da impartire all'hardware, per ottenere il risultato voluto, nei due casi sono scritti in modo diverso, l'utente non è tenuto a conoscere i dettagli grazie al fatto che il Sistema Operativo trasforma il suo comando *Stampa* nei comandi appropriato in modo invisibile.

Due dispositivi fisici differenti corredati dai rispettivi moduli di Sistema Operativo, possono diventare due dispositivi logici identici, l'unica condizione è che i moduli relativi mettano a disposizione del sistema le medesime primitive per eseguire le stesse operazioni logiche.

A questo punto occorre dare il concetto di *grado di* ***portabilità*** ***che è l'attitudine di un Sistema Operativo ad essere eseguito su macchine diverse: tale grado è tanto maggiore quanto è minore il costo delle modifiche necessarie.***

Se il Sistema Operativo è modulare il costo si riduce notevolmente in quanto le modifiche sono concentrate nei soli moduli interni riguardanti l'hardware e non sugli altri.

CLASSIFICAZIONE DEI S.O.

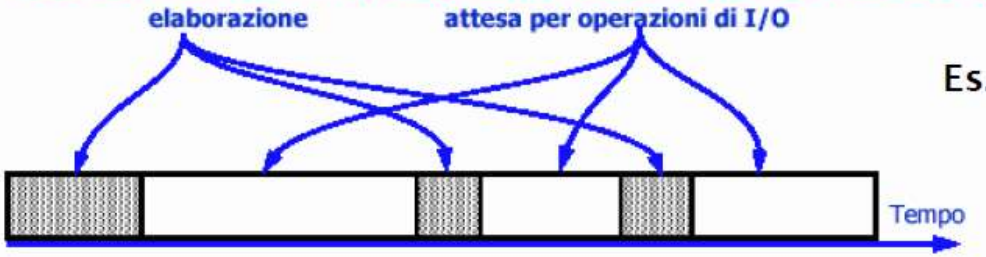
- In base al numero di utenti:
  - mono-utente (mono-user)
    - un solo utente alla volta può utilizzare il sistema
  - multi-utente (multi-user)
    - più utenti in contemporanea interagiscono con la macchina
    - il S.O. fornisce a ciascuno l'astrazione di un sistema "dedicato"
- In base al numero di processi:
  - Mono-programmato (mono-task)
    - si può eseguire un solo programma per volta
  - Multi-programmato (multi-task)
    - il SO permette di eseguire più programmi in contemporanea
    - il SO gestisce la suddivisione del tempo della CPU fra i vari processi (*time-sharing*)

LIMITI DEI S.O. MONOPROGRAMMATI

- Qualunque programma alterna fasi di esecuzione a fasi in cui è bloccato in attesa di qualche evento esterno
  - attesa che sia terminata un'operazione di input
  - attesa per usare una risorsa al momento occupata
- Sotto-utilizzo del processore
  - mentre il programma è bloccato in attesa di eventi esterni, il processore rimane inattivo (*idle*)
  - i tempi di lavoro delle periferiche di input/output, o addirittura i tempi di reazione umani sono maggiori di molti ordini di grandezza della velocità del processore

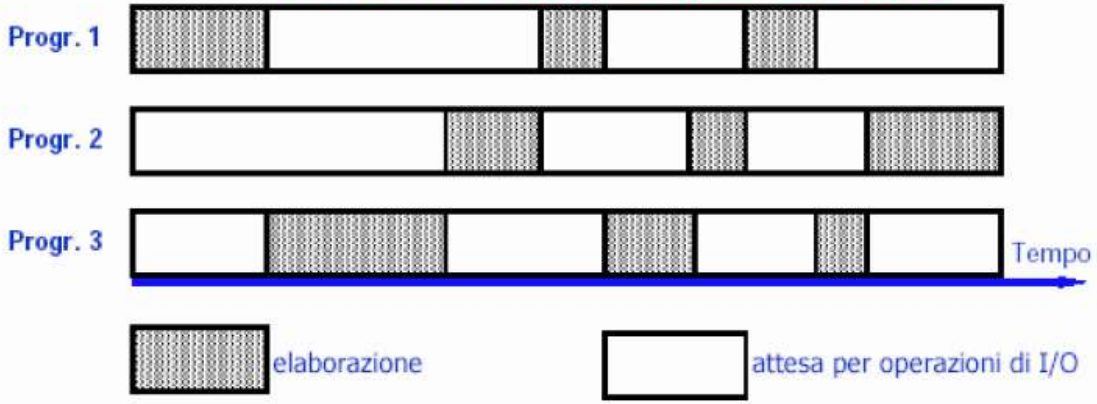
MULTIPROGRAMMAZIONE

Sistema monoprogrammato: mono-tasking



Es. Programma videoscrittura
Utilizzo CPU 2-5%

Sistema multiprogrammato: multi-tasking e time-sharing



What's time sharing?

Time Sharing

Il termine "Time Sharing" proviene dall'inglese e significa letteralmente "partizione di tempo". Questa è una tecnica sviluppatasi negli anni

Sessanta, che permette di avere in esecuzione più di un programma sullo stesso computer e nello stesso momento (da l'impressione di farlo nello stesso momento).

In questo modo, il computer può anche servire più utenti su differenti terminali simultaneamente. Anche i classici Windows e Linux sono sistemi operativi time sharing.

Le idee che stanno alla base

Il time sharing è basato sull'idea che un computer consumi la maggior parte del suo tempo in attesa che succeda qualcosa.

In un sistema time sharing, perciò, si cerca di ovviare a questo problema; si riesce, infatti, a caricare più programmi alla volta in memoria in modo che, quando il computer non è in grado di eseguirne uno, passa ad un altro.

I programmi che in quel dato momento non vengono eseguiti, non rimangono nella memoria occupando spazio inutilmente.

What's time slice?

Come viene suddiviso il tempo?

*Grazie al time sharing, il calcolatore ammette la presenza simultanea di più processi attivi, ma ne esegue uno per volta a rotazione e dedicando ad esso un intervallo di tempo massimo brevissimo, detto "**Time Slice**" ("quanto di tempo").*

Se il processo che ha l'attenzione non ha in quell'istante operazioni di CPU da svolgere, il sistema operativo passa subito al processo successivo; altrimenti esegue le istruzioni di quel processo per un time slice, quindi lo sospende e passa al processo successivo.

Chi ha la precedenza?

*Cosa decide quale processo ha la precedenza sull'altro? Qui entra in gioco lo "**Scheduler**" ("pianificatore"). Esso è un componente fondamentale dei sistemi operativi multitasking (ossia che permettono di eseguire più programmi contemporaneamente).*

*La sua funzione è quella di gestire in modo ottimale i turni dei processi (task) sulla CPU attraverso l'omonima operazione di **scheduling** e definendo delle politiche di ordinamento, di gestione delle priorità.*

Di cosa tiene conto lo Scheduler?

*Esistono vari **algoritmi di scheduling** che tengono conto di varie esigenze e che possono essere più indicati in alcuni contesti piuttosto che in altri. La scelta dell'algoritmo da usare dipende da cinque principali criteri:*

1. *Utilizzo del processore: la CPU dev'essere attiva il più possibile, ovvero devo essere ridotti al minimo i tempi morti*
2. *Produttività: il numero di processi completati in un determinato lasso di tempo*
3. *Tempo di completamento: il tempo che intercorre tra la sottomissione di un processo e il completamento della sua esecuzione*
4. *Tempo d'attesa: il tempo in cui un processo pronto rimane in attesa della CPU*
5. *Tempo di risposta: il tempo che trascorre tra la sottomissione del processo e l'ottenimento della prima risposta*

Obiettivi dello Scheduling

Gli obiettivi principali degli algoritmi di scheduling sono:

- *Fairness (equità): processi dello stesso tipo devono avere trattamenti simili*
- *Balance (bilanciamento): tutte le parti del sistema devono essere utilizzate al massimo*
- *Throughput: massimizzare il numero di lavori completati in un intervallo di tempo*
- *Turnaround Time: minimizzare il tempo di permanenza di un lavoro nel sistema*
- *Tempo di Risposta: minimizzare il tempo di risposta agli eventi*
- *Proporzionalità: assicurare un tempo di risposta proporzionale alla complessità dell'azione*

Esempio

- Si considerino 7 job che arrivano in sequenza, con tempi di esecuzione rispettivamente di 15, 24, 13, 6, 21, 32, 15 (in minuti). Si valuti il tempo di turnaround e il throughput.
- **Soluzione:** Il tempo di turnaround e' il tempo medio di permanenza dei job nel sistema, il throughput del sistema e' il numero medio di job eseguiti in un'ora, quindi :
- **Turnaround**: (15*7+24*6+13*5+6*4+21*3+32*2+15*1)/7=68,571 minuti
- **Throughput**: Tempo totale di esecuzione: 126 minuti per 7 job , numero medio di job completati in un'ora è dato dalla proporzione: 126: 7 = 60 : x da cui il throughput è : 3,333 minuti

Politiche di Scheduling

Con questo termine si intende la strategia usata per assegnare la risorsa CPU ai vari processi.

Un sistema di tipo time sharing deve gestire le risorse in modo tale da dare l'impressione che i vari processi procedano simultaneamente.

Il sistema di gestione dell'attribuzione della risorsa CPU si basa sulla ***priorità*** *di un processo; tale priorità varia in continuazione nel corso della vita di un processo.*

Può essere opportuno, infatti, rendere prioritari alcuni processi di particolare importanza o che accedono a strutture molto utilizzate per evitare code di processi in attesa. Bisogna fare in modo, però, che tutti i processi possano ottenere la CPU senza essere scavalcati

continuamente da processi più importanti.

Le priorità, ovvero cosa faccio

Calcolo delle priorità

*Un elaboratore è dotato del **clock**, un meccanismo hardware che genera di tanto in tanto delle interruzioni: il processo attualmente in esecuzione viene sospeso per effettuare alcune procedure di manutenzione del sistema, fra cui il **calcolo della priorità dei processi**.*

Questo calcolo tiene conto di una stima dell'utilizzo della CPU da parte dei processi con un fattore correttivo che può essere utilizzato dall'utente per influenzare le procedure per il calcolo della priorità. Chiaramente l'obiettivo di tale fattore non è quello di attribuire in modo esclusivo le risorse ad un solo processo, ma fare avanzare più rapidamente un processo rispetto ad un altro.

Politiche di gestione

Esistono due politiche di gestione dello scheduling che vengono eseguite periodicamente, a intervalli regolari:

*"**Schedcpu**": ricalcola la stima dell'utilizzo della CPU per tutti i processi. Se la priorità di qualche processo è **diventata maggiore** di quella del processo corrente, segnala la necessità di interrompere il processo corrente che avrà priorità più bassa e sarà rimpiazzato da quello con priorità più alta.*

*"**Roundrobin**": eseguita con frequenza maggiore rispetto alla prima, cerca di vedere se c'è la necessità di una **sostituzione** del processo corrente. Tale sostituzione avviene se c'è in coda un processo con priorità almeno uguale a quella del processo corrente.*

Round Robin

Processo	Durata
P_1	24msec
P_2	3msec
P_3	3msec

Quanto=4msec

0 4 7 10 14 18 22 26 30

Esempio

- Si considerino 3 processi (A, B e C) che vengono attivati contemporaneamente, con tempi di esecuzione rispettivamente di 30, 20, 10 (millisecondi). Valutare il turnaround time dei tre processi nell'ipotesi di schedulazione round-robin, con un quanto di tempo di 10 ms, trascurando il tempo necessario per la commutazione di contesto e supponendo che passi in esecuzione prima il processo A, poi B e infine C.
- **Soluzione:** l'ordine con cui vengono eseguiti i processi è il seguente:
- A (per 10 ms), B (per 10 ms), C (per 10 ms), A (per 10 ms), B (per 10 ms), A (per 10 ms) per cui C termina in 30 ms, B termina in 50 ms e A termina in 60 ms.
- Quindi il turnaround time è: (30+50+60)/3=46,666 ms

Due tipi di Scheduling

- ***Senza pre-rilascio*** *(non pre-emptive): un processo attivato viene eseguito fino a quando ha completato la sua esecuzione oppure è entrato in uno stato di attesa. Può capitare che, una volta che un processo a bassa priorità sia cominciato, uno con priorità più alta diventato pronto possa aspettare anche una notevole quantità di tempo prima di essere eseguito.*

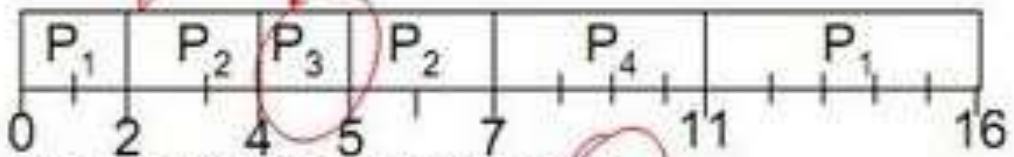
- ***Con pre-rilascio*** *(pre-emptive): ad intervalli di tempo regolari viene controllato e deciso quale processo eseguire. Rispetto alla versione non-preemptive, ha il vantaggio che quando un processo viene eseguito non ci possono essere processi pronti con una priorità maggiore.*

Processo	P_1	P_2	P_3	P_4
Tempo di arrivo	0	2	4	5
Tempo di elaborazione	7	4	1	4

Non pre-emptive

Tempo di attesa medio: 4

Pre-emptive



Tempo di attesa medio: 3

Esempio pratico: Linux

In Linux lo scheduling si basa sul concetto di time sharing, per cui ad ogni processo è assegnato un quanto di tempo massimo per l'esecuzione.

Di norma Linux prevede uno scheduling pre-emptive, per cui ad un processo viene tolta la CPU se:

1. *Esaurisce il quanto di tempo a sua disposizione*
2. *Un processo a priorità più alta è pronto per l'esecuzione.*
3. *Il tempo è suddiviso in periodi, detti epoche, che si ripetono ciclicamente. Quando un processo viene creato, il sistema operativo assegna un quanto di tempo al processo; quando tutti i processi eseguibili hanno esaurito il loro quanto, l'epoca corrente termina e ne inizia un'altra.*

MULTIPROGRAMMAZIONE

- Nel sistema sono presenti diversi programmi, ognuno con un proprio tempo di elaborazione e propri tempi di attesa per le operazioni di ingresso/uscita.
- Per evitare che la CPU venga utilizzata in modo esclusivo (o per troppo tempo) da parte di un solo programma, il tempo viene idealmente suddiviso in unità elementari, dette **quanti**, da assegnare secondo opportune politiche a tutti i programmi.
- **Round-robin**: assegnare a rotazione la disponibilità di un quanto di tempo della CPU ai vari programmi presenti contemporaneamente in memoria.
- La durata del quanto di tempo incide significativamente sia sulle prestazioni del sistema che sull'efficacia del quasi parallelismo, che tende a scomparire se la durata diviene eccessiva e degrada nella sequenzializzazione dei programmi. D'altra parte, pur migliorando in generale le proprietà di parallelismo la scelta di un valore molto piccolo può comportare un degrado delle prestazioni complessive del sistema, qualora il tempo di commutazione fra programmi sia dello stesso ordine della durata del quanto di tempo (un valore tipico per il sistema operativo Unix è 100 ms).

MULTIPROGRAMMAZIONE

- **Programma**:
  entità statica composta dal codice eseguibile dal processore.
- **Processo**:
  entità dinamica che corrisponde al programma in esecuzione, composto da:
  - codice (il programma);
  - dati (quelli che servono per l'esecuzione del programma);
  - stato (a che punto dell'esecuzione ci si trova, cosa c'è nei registri, …).

Un po' di storia

Gli S.O. dalle valvole termoioniche

al touch screen

Anni '40, assenza di SO, "programmatore-operatore"

Non esisteva il concetto di Sistema Operativo.

Il programmatore inseriva il programma, scritto in **linguaggio macchina**,attraverso primitivi lettori di schede perforate e i risultati venivano inviati alla stampante.

Un primo abbozzo di SO si è avuto con la messa a disposizione di tutti i programmatori dei sottoprogrammi di I/O e con l'avvento del linguaggio simbolico **Assembler**.

Anni '50, i primi Sistemi Operativi, Sistemi Batch [I generazione]

- Non essendo stata ancora introdotta la tecnologia di accesso diretto alla memoria (DMA) tutte le operazioni di input/output erano a carico della CPU, con conseguente rallentamento dell'esecuzione vera e propria. Per questo che si adottò la soluzione Batch (a *lotti)*.
- Il Sistema Operativo di questi calcolatori (***Batch Monitor***) svolgeva pochi servizi quali la gestione dell'input/output e l'interpretazione ed esecuzione dei comandi contenuti nelle schede di controllo. Tali comandi costituiscono il **JCL** (*Job Control Language*).

Anni '60, la multiprogrammazione [II generazione]

- Si hanno i primi SO in multiprogrammazione *time-sharing* e sistemi *real-time* per il controllo di processo.
- Nel 1962 al MIT si realizza il **CTSS**, il primo sistema timesharing, su un IBM 7094; il MIT, la General Electric e i Bell Labs realizzano congiuntamente il **MULTICS**, un sistema operativo in grado di supportare centinaia di utenti in timesharing.
- Questo rivoluzionario sistema introdusse molte idee che influenzarono notevolmente i successivi Sistemi Operativi.

Anni '65, i sistemi *general purpose*, l'IBM S/360 [III generazione]

- Nell'aprile 1964 la IBM presenta la famiglia di computer System 360 (una serie di computer di potenza diversa, ma con la stessa architettura) che utilizza il sistema operativo **OS/360**.
- Tale sistema operativo era onnicomprensivo, gestiva il *batch*, il *time-sharing* e il *real-time*, supportava sia il calcolo scientifico (in Fortran) che quello commerciale.

Anni '75-oggi, sistemi *user-friendly*, data base, reti [IV generazione]

- I sistemi operativi tendono a fornire funzioni prima caratteristiche dei programmi applicativi (data base, protocolli di comunicazione, …), compaiono elaboratori multiprocessor con relativi sistemi operativi multiprocessing.

Sistemi Operativi per Personal Computer

- I più importanti sistemi operativi per PC erano il **CP/M-80** (della Digital Research per le CPU Intel 8080 e Z-80) e l'**MS-DOS**, della Microsoft, simile al CP/M, adottato dalla IBM per il proprio Personal Computer lanciato nel 1981 (fatto, questo, che propiziò lo sviluppo irresistibile di Microsoft, oggi la più grande azienda informatica a livello mondiale)

Sistemi Operativi GUI (*Graphic User Interface*) 1

- Steve Jobs, il fondatore di ***Apple*** Computer, era uno dei pochi che credeva nell'idea del Personal Computer (a quell'epoca era difficile immaginare l'utilità di un computer personale).
- Nel 1984, ispirandosi all'interfaccia grafica sviluppata da ***Xerox*** qualche anno prima, Apple lancia, sul Macintosh, **Mac OS** il primo sistema operativo per PC con interfaccia grafica. **OS** il primo sistema operativo per PC con interfaccia grafica.
- Questa fu una vera rivoluzione. Poco dopo in ambiente Unix nasce **X Window System** e l'anno successivo ***Microsoft*** commercializza **Windows**.

Sistemi Operativi GUI (*Graphic User Interface*) 2

- All'inizio Windows non era un vero e proprio sistema operativo, ma un'estensione di MS-DOS. Nel 1987 la IBM lancia **OS/2**, un sistema operativo grafico per il suo PC PS/2, ma con scarso successo.
- Nel 1990, con Windows 3.0 che supporta il multitasking e la memoria virtuale, Microsoft si impone sul mercato. Con Windows 3.10 e 3.11, nel 1992, viene introdotto il supporto alla multimedialità e le funzioni di rete per lan peer to peer.
- È con l'introduzione, nel 1995, di **Windows 95** che si può parlare, per Windows, di sistema operativo vero e proprio e si passa dal calcolo a 16 bit a quello a 32 bit.
- Seguiranno Windows **98** (1998), **ME** (2000), **2000** (2000), **XP** (2001), **Vista** (2007), **Windows 7** (2009), **Windows 8** (2012) e **Windows 10** (2015).
- Anche **Linux**, nelle sue varie distribuzioni - *Red Hat* (1994), *Debian GNU/Linux* (1996), *Ubuntu* (2005), etc. - fornisce l'interfaccia grafica.

Unix, Linux, Android

- **UNIX** fu progettato a partire dal 1969 da un gruppo di ricercatori della AT&T presso i Bell Labs, tra i quali erano presenti Ken Thompson (che lavorò anche al progetto Multics) e Dennis Ritchie. Da esso furono realizzate varianti come BSD(*Berkley Software Distribution*) e……..
- successivamente l'ormai famosissimo **LINUX** sviluppato, nel 1991, dallo studente finlandese Linus Torvalds. La fortuna di Linux è dovuta al suo abbinamento con il Progetto GNU di Richard Stallman, portavoce della filosofia del *Software Libero*.
- **Android** è un sistema operativo per dispositivi mobili(mobile OS) sviluppato da Google Inc (Andy Rubin) e basato su kernel Linux; non contiene codice GNU, pertanto non è da considerarsi una distribuzione GNU-LINUX.

Caratteristiche di un S.O

Quali parametri devo considerare per distinguere un buon sistema oprativo?

Un sistema operativo, affinché possa avere effetto positivo sul libero mercato deve presentare alcune caratteristiche quali:

1. **CONCORRENZA**: esistenza di più attività contemporanee o parallele sulla stessa macchina;
2. **CONDIVISIONE**: utilizzazione contemporanea di una stessa risorsa da parte di due o più attività;
3. **AFFIDABILITA'**: un S.O. deve essere completamente immune da commettere errori ed essere quindi in grado di controllare tutte le situazioni e gestire tutti gli eventi;
4. **FACILITA' DI MANUTENZIONE**: è necessario che il sistema presenti una struttura modulare e che sia dotato di un'ampia e chiara documentazione nel caso si dovesse intervenire per eventuali aggiornamenti;
5. **FACILITA' D'USO**: rappresenta uno dei requisiti più importanti per la sua diffusione.

Un Sistema Operativo facile da usarsi, richiede un linguaggio comprendente un numero limitato e basso di comandi ed una sintassi molto semplice e di tipo MNEMONICO.

EFFICIENZA: come ottimizzazione dello sfruttamento delle potenzialità delle risorse di calcolo valutate in base alla conoscenza dei parametri:

- **THROUGHPUT**: misura dell'ammontare di lavoro che viene svolto dal sistema in un tempo determinato (produttività)
- **TURNAROUND TIME**: misura del tempo che intercorre tra l'esecuzione di un lavoro e l'uscita dei risultati finali (attività della CPU)
- **RESPONSE TIME** (tempo di risposta): caratteristica basilare nei sistemi Time Sharing: misura del tempo che intercorre dal momento della richiesta ed effettiva risposta dal sistema ;

- **AVAILABILITY** (DISPONIBILITA'): misura della facilità di accesso alle risorse del sistema;
- **SICUREZZA**: requisito principale e importante specie in multiutenza cioè protezione di dati e programmi di ciascun utente;
- **AFFIDABILITA'**: cioè regolarità di funzionamento;
- **COSTO:** mantenersi nei limiti cioè essere il più possibile contenuto;
- **SEMPLICITA' D'USO**: indica quanto l'interfaccia del Sistema Operativo verso
- l'utente sia semplice da utilizzarsi;

Il Sistema Operativo ideale dovrebbe quindi essere in grado di massimizzare il **THROUGHPUT** (produttività),

e minimizzare allo stesso tempo il TURNAROUND TIME (misura del tempo che intercorre tra l'esecuzione di un lavoro e l'uscita dei risultati finali) ossia il **TEMPO di RISPOSTA**,

dovrebbe presentare una disponibilità in qualunque utente, in un tempo estremamente breve, ed inoltre dovrebbe avere caratteristiche di sicurezza e protezione del 100%, il tutto ad un costo molto basso.

Quali sono i tipi di software?

IL SOFTWARE DI BASE

Il software di base è considerato parte di quella informatica che viene definita *informatica metodologica*. Fa riferimento alle ricerche sui metodi di programmazione e di sfruttamento degli elaboratori e dei sistemi informatici.

Un calcolatore è praticamente inutilizzabile se non è accompagnato da una serie di programmi che ne rendono possibile il funzionamento per gli utilizzatori, che non conoscono l'architettura e la realizzazione della macchina.

Questi programmi costituiscono il software di base.

Questo software raggruppa i sistemi operativi e *programmonitor* ovvero dei programmi di traduzione in linguaggio macchina delle istruzioni forniti in linguaggi simbolici a livello elevato,
programmi chiamati assemblatori e compilatori,
e infine una serie di programmi di interesse generale che forniscono un gran numero di funzioni universali.
Un settore di pertinenza di questa branca riguarda le modalità di utilizzazione dell'elaboratore. Vi sono diverse modalità possibili.

Questi modi diversi di operare implicano l'esistenza di un programma di gestione interno alla macchina, detto Sistema Operativo. La progettazione e la realizzazione dei Sistema Operativi costituisce un problema di grandissima complessità e dunque richiede metodi particolarmente evoluti.

Fin dai tempi dei primi calcolatori, si è sentita l'esigenza di dotarli di programmi, prima semplici, poi sempre più complessi e sofisticati, accessibili a qualunque utente, che fossero in grado di facilitare l'interazione uomo-macchina, rendendo agevole l'utilizzo del computer anche ai meno specialisti, e nello stesso tempo favorissero lo sfruttamento efficiente delle potenzialità del sistema.

Tale Software prende il nome di Software di Base che comprende quindi, tutti i programmi che gestiscono il sistema.

Con lo sviluppo tecnologico dell' Hardware, anche il Software di Base ha subito un'evoluzione, assumendo proporzioni sempre più vaste e svolgendo mansioni di gestione ed elaborazione sempre più complesse e sofisticate, con la finalità di rendere più agevole e semplice l'utilizzo del computer da parte dell'utente, e di aumentare l'efficienza della gestione della macchina, intesa come sfruttamento delle sue potenzialità.

Dallo sviluppo del Software di Base si è formato il concetto di Sistema Operativo come quell'insieme di programmi che costituisce il livello del Software di Base più vicino all'Hardware

Questa figura rappresenta la schematizzazione dei livelli gerarchici tra il Software e Hardware

Nella realtà possiamo definire un elaboratore elettronico un macchina stupida (non in grado da sola a prendere decisioni), le cui potenzialità di elaborazione possono essere sfruttate soltanto se adeguatamente istruita (programmata) nel linguaggio di sua conoscenza.

E' necessario quindi avere a disposizione un Software che agisca in modo da:

- permettere lo sfruttamento delle potenzialità del sistema di elaborazione senza conoscere le caratteristiche Hardware di tutti i suoi componenti;
- ottimizzare lo sfruttamento del sistema e facilitare il colloquio uomo - macchina.

IL SOFTWARE APPLICATIVO

Sono compresi quei programmi creati mediante il Software di Base (EDITORI, TRADUTTORI ecc) **e indirizzati a risolvere problemi di diverso tipo** (gestionale, scientifico ecc).

Il Software Applicativo viene definito non flessibile, data la sua non utilizzazione in ambiti diversi da quello del particolare problema a cui è stato indirizzato.

IL SOFTWARE ORIZZONTALE

Per rendere più semplice l'uso del calcolatore da parte di utenti non specialisti di informatica, si è diffuso un tipo di Software con caratteristiche di versatilità e con possibilità di impiego diverse da quelle dei linguaggi di programmazione classici (Pascal, Basic, Cobol ecc.).

Questo Software classificato Orizzontale, salvo che per usi molto sofisticati, **non richiede la conoscenza della programmazione e può essere utilizzato in modo quasi immediato come l'elaborazione di testi** (WORD), **la gestione degli archivi** (DATA BASE), **il foglio elettronico** (EXCEL), **creazione di ipertesti** (HTML), **la costruzione di modelli**, **il calcolo finanziario, la grafica ecc.**

Tutti programmi flessibili. Sono Software di questo tipo anche: FOGLIO ELETTRONICO o SPREAD-SHEET (LOTUS 123...) e i cosiddetti INTEGRATI (SINPHONY, WINDOWS versione 3.1, EXCEL) che inglobano, al loro interno, applicazioni varie. Questi programmi sono detti Tools.

PROGRAMMI DI UTILITA'

Oltre al Sistema Operativo che è la componente principale, il Software di Base è composto dai PROGRAMMI di UTILITA' che facilitano l'utente per lavori di uso comune in sistema di elaborazione.

I principali Programmi di Utilità sono:

- EDITOR(editore di testi o linee): che consente di memorizzare nuovi testi e modificare di già esistenti (esempio PROGRAMMI SORGENTI, TESTI, LETTERE ecc);
- SORT (ordinamento): che consente di ottenere un archivio ordinato da uno non ordinato;
- MERGE (fusione): che permette di fondere più archivi in uno solo; COPY(riproduzione): che consente di generare nuove copie di archivi, programmi, documenti, ecc. su supporti di memoria anche diversi da quelli in cui risiede;
- FORMAT (tracciato): serve per preparare un supporto nuovo di memoria da essere in sintonia con la Memoria centrale, cioè rende compatibile il supporto esterno (disco) e Memoria Centrale per eventuali scambi di dati.

TIPI DI ELABORAZIONE

Il modo di operare di un sistema definisce i seguenti tipi di elaborazione:

A LOTTI (BATCH PROCESSING): più lavori vengono raccolti e presentati al sistema che li elabora in sequenza e ne fornisce i risultati relativi,(esempio: il calcolo degli stipendi mensili ecc.).

IN TEMPO REALE (REAL TIME PROCESSING): i dati su cui operano i programmi per ottenere risultati arrivano ininterrottamente e vengono elaborati istantaneamente influenzando eventualmente elaborazioni successive(esempio: le prenotazioni di posti su linee aeree).

Bibliografia, sitografia

- TIC Tecnologie dell'informazione e della comunicazione. Camagni-Nikolassy. Hoeply
- Tecnologie e Progettazione di sistemi informatici e di telecomunicazioni. Lorenzi-Cavalli. ATLAS
- SISTEMI OPERATIVI silberschatz abraham - galvin p. baer. Hoeply.
- http://www.orangeworldsite.com/2012/10/appunti-di-sistemi-operativi-20122013.html
- http://www.riccardogalletti.com/appunti_gratis/doc/dispense_sistemi_operativi_tor_vergata.pdf
- http://www-db.deis.unibo.it/~slodi/I/2006-2007/presentazioni/so.pdf
- http://www.informatica.uniroma2.it/upload/2013/SOR/SISTEMI%20OPERTATIVI%20LEZ02.pdf
- http://mp7.altervista.org/Multiprogrammazione.html
- http://didattica.agentgroup.unimore.it/wiki/images/b/b9/Sistemi_Operativi.pdf
- http://datasked.com/pub/doc/it/thesis/mtrentini/tesi.pdf