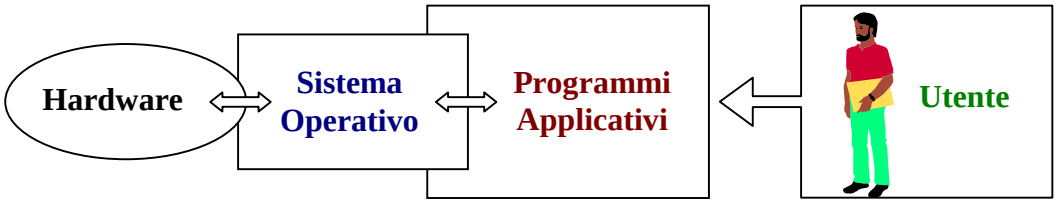
Il software

Il software viene usualmente suddiviso dal punto di vista della sua funzionalità. Si usano distinguere tre componenti: il sistema operativo, il software applicativo e i dati, anche se da un punto di vista teorico la terza componente può conglobarsi con la seconda.
Il sistema operativo (SO) o software di sistema è un insieme di programmi che gestisce il funzionamento di base del computer, il software applicativo è un insieme di programmi creati per utilizzare il calcolatore secondo le diverse finalità come ad esempio i programmi di scrittura, di ritocco fotografico, gli antivirus, ecc. (installati sulla macchina dopo che questa contiene già il sistema operativo), i dati sono una serie di informazioni ”statiche”, utilizzate dai programmi ma incapaci, da soli, di mettere in moto il calcolatore.
In definitiva si può dire che il Software di sistema serve alla macchina per funzionare, mentre il Software applicativo serve all’utente per lavorare.



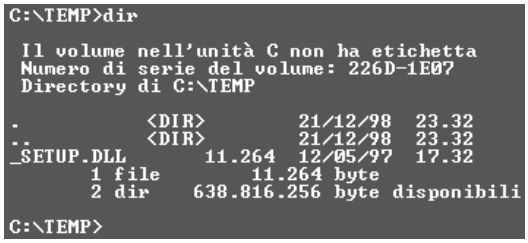
I sistemi operativi

Il sistema operativo si occupa dell’interazione tra utente e computer. E’ quell’insieme di programmi che rende all’utente il computer semplice ed efficiente.
Il sistema operativo agisce da intermediario tra l’utente (ed i suoi programmi applicativi) e l’hardware in modo da governare le risorse costituenti il computer.
Lo scopo è quello di fornire un ambiente in cui l’utente possa eseguire i suoi programmi in maniera semplice, conveniente (user friendly) ed efficiente.
Deve essere conveniente in quanto:
a) deve mostrare in maniera chiara all’utente i dati contenuti nella memoria di massa;
b) deve curarsi delle cose marginali che non interessano l’utente quali: I/O con le varie periferiche, tramite i device drivers. Non deve essere l’utente ad istruire il computer su come scrivere un dato di output; deve caricare i programmi dalla memoria di massa alla memoria volatile;
c) deve curarsi che il passaggio di dati tra un programma e l’altro sia semplice;
d) deve curarsi che passare da un programma in esecuzione all’altro sia semplice;
e) deve controllare l’esecuzione di un programma e se questo “abortisce” deve dire perché;
f) deve prevenire l’uso improprio di una qualsiasi componente del sistema
Deve essere efficiente in quanto:
a) deve allocare le varie risorse hardware tra i vari programmi applicativi. Deve fare in modo che le risorse siano sempre occupate a far lavoro utile e che tutti i programmi siano serviti delle risorse in maniera equa.
b) deve badare alla sua sopravvivenza.

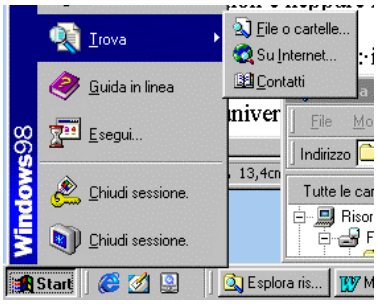
Come funzionano i sistemi operativi

I sistemi operativi sono degli Interrupt based systems, ovvero dei programmi sempre in esecuzione (sempre residenti nella RAM) che ascoltano e, quando interrotti dall’utente o dai programmi fanno quello per cui sono stati interrotti. Ciò fa sì che l’ambiente che creano sia interattivo.

Dal punto di vista dell'interfaccia con l'utente esistono due categoria di sistemi operativi: a linea di comando e ad interfaccia grafica.



Esempio tipico di sistema operativo a linea di comando è Ms-Dos: sullo schermo (di un colore scuro uniforme) non compare nessuna grafica e tutti i comandi devono essere digitati da tastiera.
Tali sistemi operativi sono estremamente scomodi e difficili da usare, per cui sono stati creati programmi che, appoggiandosi comunque alle funzioni del sistema a linea di comando, forniscono all'utente un'interfaccia grafica semplice da usare.



Esempi tipici di SO a interfaccia grafica sono Windows 98 (e successivi) e MacOS (piattaforma Macintosh). In questi sistemi GUI (Graphical User Interface) tutte le operazioni si svolgono tramite icone e finestre, usando intensivamente il mouse per lanciare comandi, scegliere opzioni, ecc.
Questi sistemi si dicono *user friendly* (amichevoli verso l'utente) perché anche un utente inesperto riesce, entro certi limiti, ad intuirne il funzionamento (il che non accade certamente con i sistemi a linea di comando).

Tramite il sistema operativo (SO) i programmi applicativi comunicano con l'hardware evitando di dover includere nei programmi stessi tutte le istruzioni necessarie a questa gestione. Nei primi calcolatori e in alcuni casi anche in moderni calcolatori destinati a funzioni particolari non esiste un sistema operativo.
La ragione principale che ha portato alla nascita dei sistemi operativi è la *concorrenza di accesso* ad una risorsa. Il sistema operativo permette infatti a più programmi, e quindi eventualmente a più utenti, di accedere contemporaneamente alla medesima risorsa hardware: ad esempio alla CPU, alla memoria o al disco.

Loader, ROM e bootstrap.

Nei primi calcolatori il sistema operativo svolgeva solamente il compito di caricare degli altri programmi. Questa parte particolare del sistema operativo è rimasta tuttora e viene detta *loader*.
Il loader viene caricato in memoria all'accensione della macchina usando la memoria ROM (alla partenza del computer, o all'esecuzione di un *RESET* della macchina, viene messo nel program counter l'indirizzo della ROM che contiene il loader).
Normalmente nella memoria ROM viene caricato quell'insieme minimo di funzioni che permette di far partire il computer caricando da altri dispositivi periferici il resto del sistema operativo. La fase di caricamento del sistema operativo viene detta *bootstrap* (letteralmente significa alzarsi tirandosi per i lacci delle scarpe, a sintetizzare il fatto che l'avvio avviene a partire dalla piccola parte che è stata precaricata in memoria ROM).
Nel caso dei personal computer IBM compatibili, le ROM contengono quello che viene chiamato BIOS (Basic Input/Output System). Il BIOS si incarica di eseguire una procedura di autodiagnostica (POST, Power On Self Test) che procede a tutta una serie di controlli e verifiche sulle varie componenti e permette di caricare il sistema operativo da hard disk (o da floppy disk).
Nei personal più vecchi, quando si voleva aggiungere un dispositivo hardware che non era stato previsto o sostituirne uno nuovo, era necessario sostituire la ROM. Con i sistemi attuali, invece, ciò non è più necessario in quanto ogni periferica nuova che si collega al computer (stampanti, unità di memoria di massa, schede...) richiede che venga installato il proprio *driver* nel sistema operativo.

Gestione delle risorse fisiche

Nel caso dei primi sistemi i programmi eseguivano direttamente le istruzioni di gestione delle risorse fisiche e quindi provvedevano a usare direttamente le istruzioni di Input/Output sulle periferiche. Per facilitare lo sviluppo di programmi sono state inserite nel sistema operativo funzioni di lettura e scrittura dalle periferiche che consentono di non conoscere i dettagli di questa gestione. Nei sistemi operativi più moderni la gestione delle periferiche viene fatta attraverso un insieme di funzioni molto limitato ed uguale su tutte le periferiche. Questa gestione è del tutto analoga a quella usata per utilizzare i file del disco.

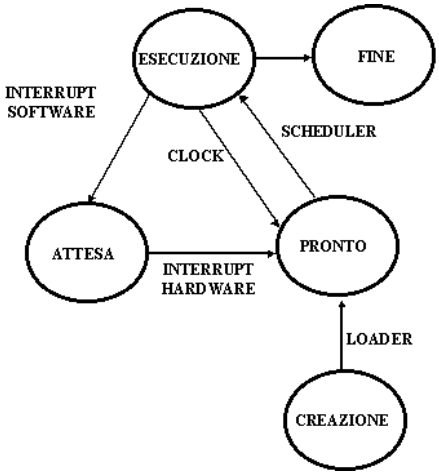
Interruzioni, device driver e virus

La CPU, dopo aver mandato in esecuzione delle istruzioni di I/O ad una periferica, rimane in attesa della risposta della periferica: ciò è necessario perché la CPU è molto più veloce delle periferiche. La soluzione che è stata inventata e che è la base dell'intera architettura di funzionamento dei sistemi operativi è la seguente: invece di mandare continuamente segnali alla periferica e disturbarla per chiedere se ha finito, è la periferica a mandare un segnale quando è pronta a spedire una risposta. Questo segnale è un particolare filo del bus di controllo che viene detto *Interrupt*.
L'idea che viene subito è di sfruttare questo periodo di attesa della CPU per eseguire le istruzioni di un'altro programma (paragrafo successivo).
Il sistema operativo, all'arrivo di una interruzione, stabilisce quale periferica ha provocato l'interruzione e quali istruzioni eseguire per la gestione di quella interruzione. L'insieme delle istruzioni che vengono eseguite in seguito ad una interruzione viene chiamato *device driver* di quella periferica.
Questo meccanismo è così comodo che viene usato anche per le chiamate del sistema operativo fatte dai programmi applicativi: quando si desidera una funzione particolare del sistema operativo viene chiamata una istruzione speciale di interrupt detta *system call* o *interrupt software*.
Lo stesso meccanismo è anche utilizzato anche per scopi illegali. I virus informatici sono dei piccoli programmi che, modificando gli indirizzi delle istruzioni di interrupt, si fanno mettere in esecuzione ogni qualvolta un programma richiede un particolare intervento del SO. In questo modo il virus in esecuzione scrive le proprie istruzioni nel disco (infettandolo o danneggiandolo) e se ha l'accortezza di terminare lanciando il vero device driver esso passa del tutto inosservato. I virus diventano in un certo senso una parte del sistema operativo che, essendo unico, consente di far funzionare il virus e diffondere l'infezione su tutte le macchine con quel SO.

Gestione della CPU: Kernel

Vediamo ora quale sia il modo con cui si possono sfruttare i periodi di attesa delle risposte dalle periferiche per poter eseguire più programmi continuamente.
Supponiamo di avere un programma in *esecuzione* e più programmi caricati in memoria in attesa di essere eseguiti. Il programma in esecuzione rimane l'unico utilizzatore della CPU fino a che non lancia un interrupt software: in quel momento una parte del SO detta *scheduler* si incarica di capire se è possibile mettere in esecuzione altri programmi (che siano "*pronti*" per essere eseguiti, cioè non in attesa di una qualche risposta da una periferica) e fra questi viene messo in esecuzione il programma con maggiore *priorità* rispetto agli altri (decisa, ad esempio, dal periodo che ogni programma ha atteso). Così anche quando una periferica invia un interrupt hardware, il programma che era in attesa di quella risposta viene rimesso nello stato di "pronto" e ritorna in esecuzione appena lo decide lo scheduler.
A turno tutti i programmi sono messi in esecuzione se essi effettuano operazioni di accesso alle periferiche. Tuttavia se un programma in esecuzione non effettua nessun interrupt software, è il clock di sistema a provvedere mandando periodicamente delle interruzioni.

Il grafico a fianco illustra sinteticamente il funzionamento dell'esecuzione concorrente di programmi in un moderno sistema operativo (i cerchi sono gli stati dei processi, le frecce sono gli eventi che determinano un cambiamento di stato) a partire dalla creazione del processo fino al suo termine.
La parte del sistema operativo che contiene lo scheduler e la sincronizzazione dei processi si occupa, di fatto, della gestione della risorsa più importante, cioè della CPU, e per questo viene detta *Kernel* (nocciolo).



Problemi di parallelismo e concorrenza

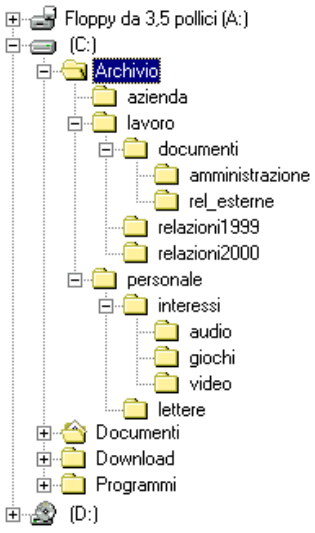
Se due processi vogliono accedere contemporaneamente alla stessa risorsa, cioè alla stessa periferica od alla stessa zona di memoria, in sola lettura non si pongono problemi di sorta.
La situazione cambia se qualcuno dei processi scrive sulla risorsa. In questo caso dobbiamo aspettare che il processo finisca la scrittura prima che un altro processo possa poter procedere ad una lettura od a una scrittura a sua volta. Il problema viene detto di *concorrenza* e può essere analogo a quello visto per la CPU.
La soluzione che forniscono quasi tutti i kernel dei sistemi operativi è quella che viene detta dei *semafori*. Un semaforo è un bit della memoria che viene gestito dal sistema operativo. Se vale 1 allora vuol dire che la risorsa è occupata e occorre che il processo si metta in attesa della sua liberazione. Se invece vale 0 allora la risorsa è libera e mettendo il semaforo a 1 si impedisce ad altri l'accesso.

Gestione della memoria

Il SO si occupa anche della gestione della memoria con attenzione ed efficienza, caricando nelle memorie "ad alto livello" solo i dati e le istruzioni che devono essere modificate ed eseguite. Le altre parti sono invece spostate sul disco per poter essere caricate quando necessario.
La tecnica oggi utilizzata è detta *memoria virtuale* (a paginazione).

Il File System

Il File System è il modo con cui il sistema operativo gestisce la memorizzazione dei dati sui dischi e sui vari supporti di memoria di massa.
Esistono diversi tipi di file system (a seconda dei vari sistemi operativi), ma comunque tutti organizzano il contenuto dei dischi in *Files* e *Cartelle* (o Directory o Folder), seguendo una metafora ripresa dall'archivistica.
Ogni singolo oggetto memorizzato su disco viene detto *file* (termine che anticamente corrispondeva all'italiano filza, fascicolo, incartamento): un file può essere un programma eseguibile, un insieme di dati numerici, un documento di testo, un'immagine, un documento sonoro, un filmato, una pagina web, un'animazione, o altra cosa.
Sui dischi di un PC possono trovarsi fino a molte decine di migliaia di file e se non ci fosse nessun criterio di ordinamento sarebbe molto complicato riuscire a rintracciare ogni volta il file che ci interessa, per questo motivo il file system permette di creare dei "contenitori", detti *cartelle*, che permettono di raccogliere i file in gruppi logicamente omogenei. A loro volta le cartelle possono contenere altre sottocartelle e così via, in un gioco di scatole cinesi. Per questo, per sapere dove si



trova un file è necessario conoscere tutto il *path* (percorso) per raggiungerlo. La radice (root) è la cartella di livello gerarchico più elevato, corrispondente a un intero volume (un *volume* è un contenitore logico di file, corrispondente solitamente a una unità a disco, ad esempio il volume *a:* indica il floppy disk e *c:* il disco fisso primario).
Nelle interfacce (GUI) messe a disposizione dal sistema operativo, i file e le cartelle vengono rappresentati con dei simboli grafici detti *icone*. Le cartelle sono quasi sempre rappresentate con l'immagine di una cartellina d'archivio  , mentre i file hanno le icone più varie, a seconda del tipo.
Ogni file e ogni cartella deve possedere un *nome* che lo distingua dagli altri. Molti sistemi operativi (fra cui Windows) includono nel nome una sigla aggiuntiva detta *estensione* che caratterizza il tipo di file. L'estensione viene separata con un punto dal resto del nome, così ad esempio in Windows i file che terminano con *.exe* sono programmi eseguibili, *.txt* sono file di testo semplice, *.doc .xls .mdb .ppt* sono documenti rispettivamente di Word, Excel, Access, PowerPoint, *.htm* e *.html* sono pagine Web, *.wav .mp3* sono file audio, *.jpg .gif .bmp .png* sono immagini, *.mov .avi .mpg* sono filmati, e così via. Di solito non si usano estensioni con i nomi delle cartelle (ma sarebbe comunque possibile).
Due oggetti con lo stesso nome (estensione compresa) non possono trovarsi all'interno di una stessa cartella, ma possono invece esistere in due cartelle diverse (anche se contenute una nell'altra).
Per quanto riguarda l'hard disk (su cui si trova il sistema operativo), la maggior parte dei file e delle cartelle che vi si trovano viene creata e gestita direttamente dalle applicazioni senza l'intervento diretto dell'utente (si tratta di file di sistema, di configurazione o di dati).
Ogni disco contiene l'indice di tutti i files, generato e aggiornato automaticamente dal sistema operativo. L'indice memorizza per ogni file il nome, la posizione fisica nel disco, le dimensioni in byte, la data di creazione o modifica, la cartella in cui è contenuto e altre informazioni (a seconda del file system). Il sistema operativo fornisce poi all'utente tutti i programmi per visualizzare l'indice dei dischi e per organizzarne il contenuto, spostando, copiando o cancellando i file, cambiandone il nome, creando nuove cartelle, ecc.
Alcuni sistemi operativi pongono infine restrizioni e protezioni sui file, impedendo (o perlomeno ostacolando) modifiche di files "delicati" (come i file di sistema) o anche impedendo la visualizzazione di file che contengono informazioni riservate o personali.
Dal punto di vista della memorizzazione su disco, un dato occupa un certo settore in una certa traccia, più dati collegati fra loro possono occupare diversi settori. Per salvare i dati indifferentemente nelle locazioni di memoria, anche in quelle isolate, il sistema operativo crea una tabella sul disco in cui inserire il nome e la lista dei settori usati: ogni riga è un nostro file.
Invece le cartelle sono di fatto tabelle le cui righe contengono nome del file, il numero del file nella tabella delle allocazioni in memoria, la dimensione, l'ultima data di variazione, se si tratta di un file o di una directory.
I file system più diffusi sono: FAT (File Allocation Table), FAT 32 e NTFS (New Tecnology File System) per le piattaforme Windows, HPFS (High Performance File System) per Os/2, EXT2 (Extended Filesystem 2) per Linux.

Classificazione dei SO

Un sistema operativo è *multi-programming* se è in grado di avere più programmi nel processo di essere eseguiti ad ogni istante. In un tale sistema la CPU salta da un lavoro ad un altro a convenienza, ad esempio se occorre aspettare che una qualche periferica compia un certo lavoro.
Un sistema operativo è *time sharing* (o *multi-tasking*) se la CPU è in grado saltare velocemente da un programma in esecuzione ad un altro e poi ad un altro ancora in modo che tutti i programmi in esecuzione siano effettivamente eseguiti e portati a termine. Il fatto di saltare da un programma all'altro è fatto così velocemente da creare l'illusione che vari lavori si stiano svolgendo simultaneamente, in parallelo. Il time sharing è la caratteristica fondamentale che rende interattivi gli odierni sistemi operativi.

Un sistema operativo è *multi-user* se è in grado di far sembrare un computer come tanti personal computer: uno per ogni utente. Gli utenti comunicano con il computer tramite più terminali.

I principali sistemi operativi

Unix	Il più vecchio, sviluppato nel mondo accademico americano per essere usato da supercomputers e mainframes. E' multiprogramming, time-sharing, multi-user ma non è un sistema GUI
Linux	Essenzialmente il sistema Unix per PC. E' multiprogramming, time-sharing, multi-user
MS-DOS	Il primo sistema operativo per PC. Simile allo Unix ma non è multiprogramming, non è time-sharing, non è multi-user
Macintosh	Il primo sistema operativo con GUI. E' multiprogramming e parzialmente time-sharing. La Macintosh ha inventato le finestre, l'uso del mouse ed il concetto di GUI
Windows	GUI della Microsoft per MS-DOS. Simile al Macintosh.
X11	GUI per Unix
KDE	GUI per Linux