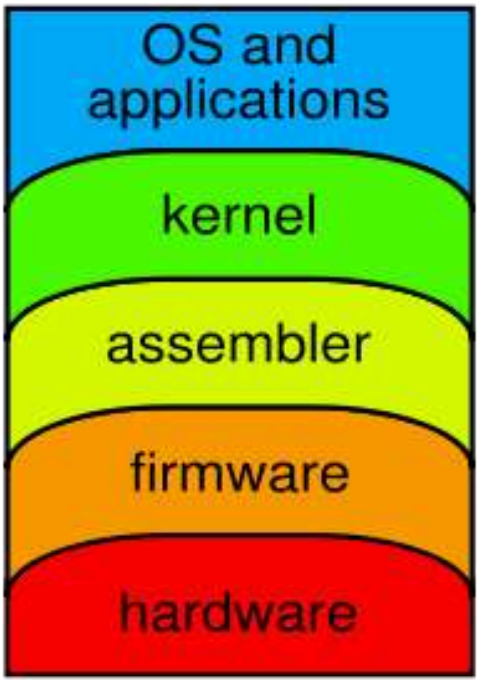
1. Sistemi Operativi

Paolo Giangrandi
paolo.giangrandi@dimi.uniud.it

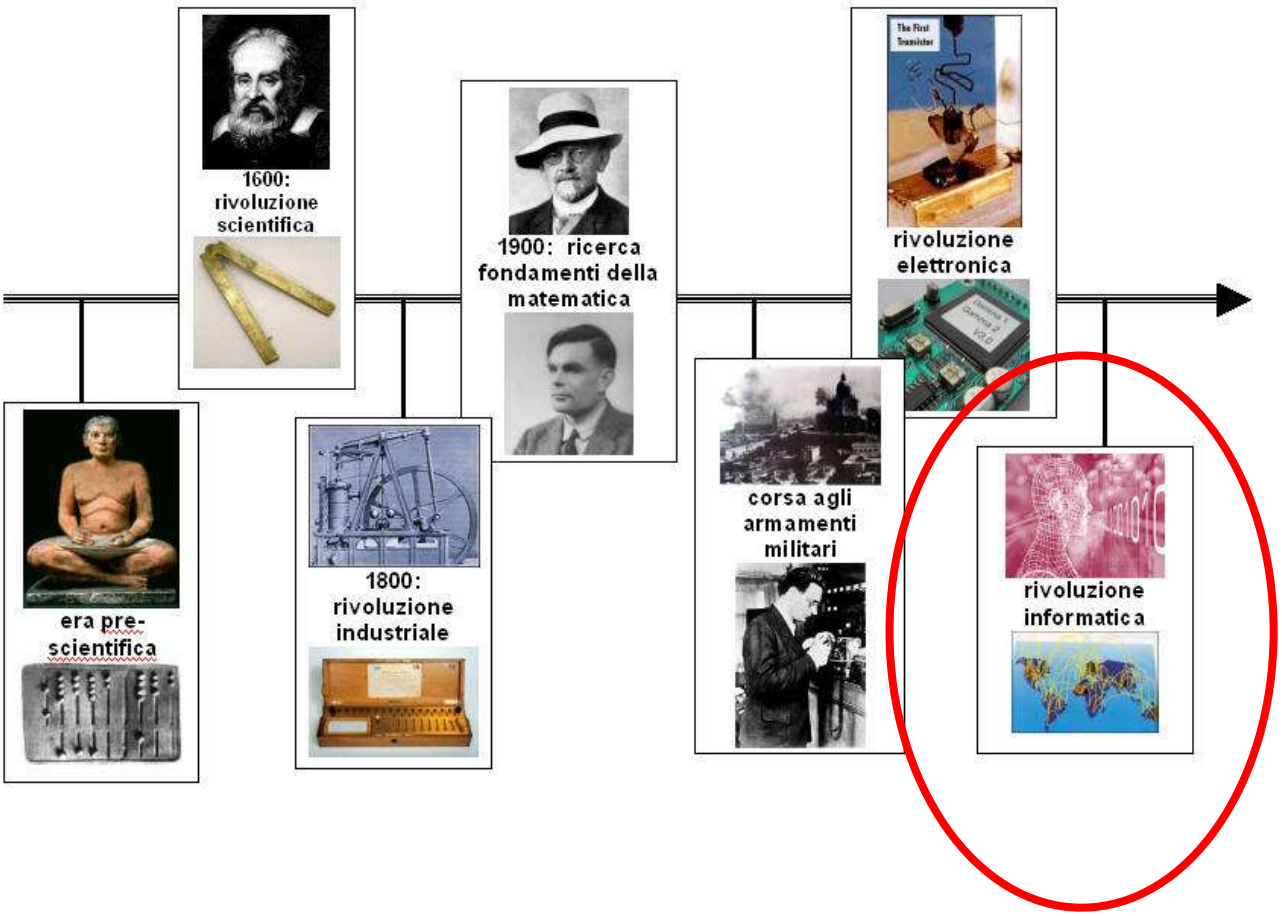


Università degli Studi di Udine

18/05/2010

Sommario

1. L'informatica attraverso la storia della scienza e della tecnica



1.1. Introduzione

Fig. Microsoft Windows, il sistema operativo attualmente più diffuso.

La costruzione di un computer non comporta semplicemente la realizzazione delle parti fisiche ma, indipendentemente dall'applicazione specifica che richiediamo alla macchina, include lo sviluppo di una **serie di programmi essenziali per il funzionamento della macchina**:

> l'insieme di questi programmi va sotto il nome di **sistema operativo** (SO).

Il sistema operativo trasforma una macchina primitiva di difficile utilizzazione in una macchina molto più ricca di quella base, più facile da usare e con un insieme di funzionalità molto più ampio di quello iniziale.

Ruolo del sistema operativo

Senza il sistema operativo il computer non sarebbe in grado di funzionare e l'unità centrale non potrebbe neppure comunicare con le altre unità che compongono l'intero sistema di elaborazione.

I sistemi operativi rappresentano un componente fondamentale dei moderni computer:

- Questi programmi di base permettono di **rendere fruibili all'utente le molteplici risorse** del computer (gestione della memoria, della stampante, della tastiera, ecc.).

- I sistemi operativi rendono queste macchine **strumenti amichevoli** utili ad affrontare molteplici attività quotidiane.

- I sistemi operativi **permettono di virtualizzare** i computer introducendo delle funzionalità che difficilmente potrebbero essere realizzate dall'hardware.

- I sistemi operativi **rappresentano una sorta di tecnologia per realizzare nuove macchine** non più basate sull'hardware ma sul software.

Diversi sono i compiti affidati al sistema operativo:

- agire come intermediario tra l'utente e l'hardware del computer stesso;
- controllare e coordinare l'utilizzo dell'hardware tra i programmi applicativi;
- offrire gli strumenti per utilizzare in modo corretto le risorse hardware e software del sistema;
- mascherare i dettagli legati alla gestione delle risorse di sistema.

I primi sistemi operativi cominciarono ad apparire verso la metà degli anni '50 quando si cominciò ad individuare una serie di routine (programmi) standard di comune utilizzo indipendenti dall'applicazione specifica richiesta al computer.

Oggi il tema dei sistemi operativi ha un'**importanza notevole nella progettazione di un computer** e le problematiche considerate risultano assai complesse.

L'**evoluzione dei sistemi operativi ha influenzato anche quella dell'hardware**, dal momento che per supportare certe funzioni del sistema operativo sono necessari meccanismi hardware ad hoc (si pensi ad esempio, alla gestione delle interruzioni, o alla gestione della memoria, …)

I primi computer senza sistema operativo

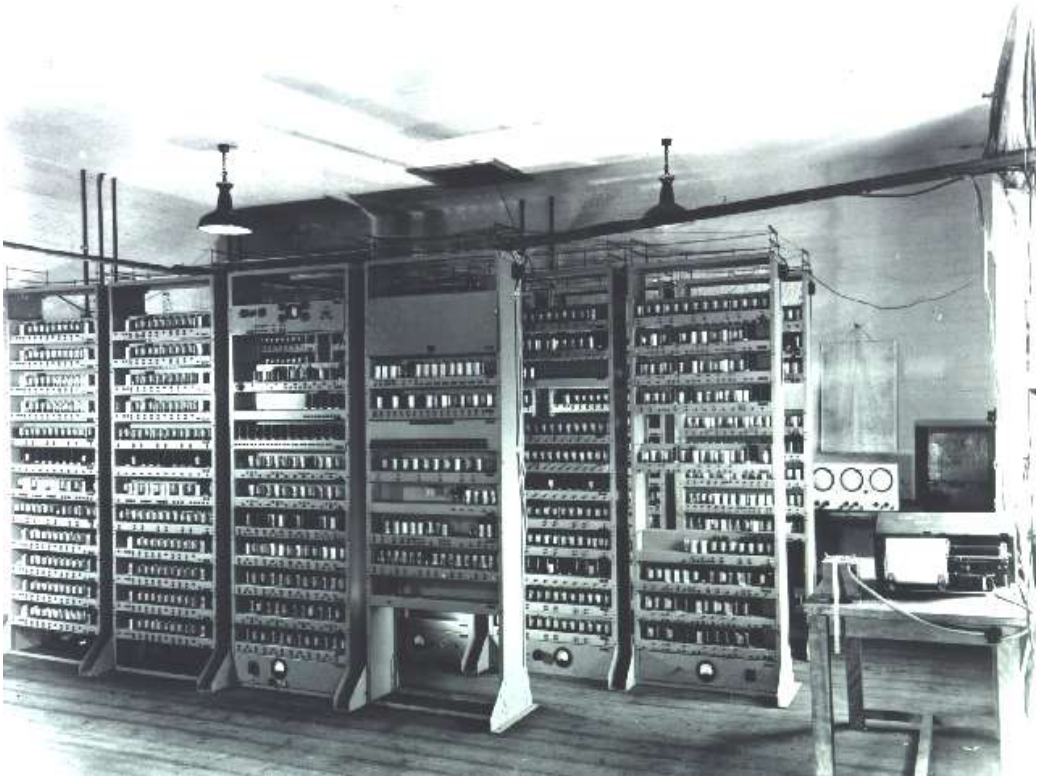
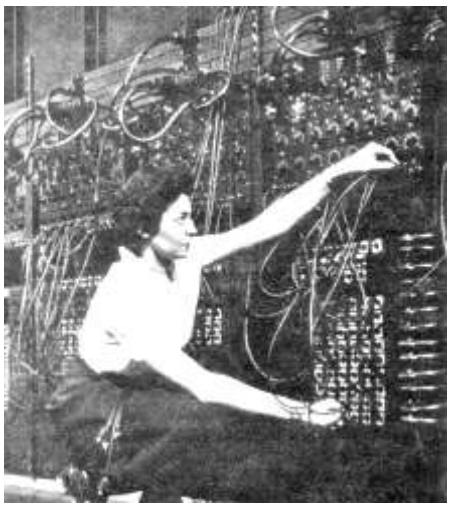


Fig. Il computer Edsac, sviluppato all'Università di Cambridge, nella seconda metà degli anni '40.

I primi computer come la macchina Z3 di Zuse, l'Eniac, l'Edsac erano del tutto sprovvisti di sistema operativo.



L'**inserimento di un programma** (scritto in linguaggio macchina) era piuttosto scomodo: in certi casi l'**inserimento di dati e istruzioni avveniva azionando un gruppo di interruttori** o modificando collegamenti tramite opportuni spinotti.

Tutto questo rendeva molto difficile l'attività di programmazione, poiché la risoluzione di un problema mediante l'uso del computer richiedeva non solo la competenza specifica per risolvere il problema affrontato, ma anche una grande conoscenza tecnica della macchina su cui operare.

Il programma in esecuzione doveva includere sia le istruzioni specifiche per la risoluzione del problema, sia le istruzioni per gestire le unità di input e output e le altre periferiche collegate al computer.

Naturalmente, durante il funzionamento del computer un solo programma alla volta poteva essere eseguito sulla macchina, per cui **in un dato momento la macchina poteva essere a disposizione di un singolo utente** (solitamente l'operatore macchina o sistemista).

Tenendo conto degli altissimi costi per la realizzazione e la gestione dei primi computer, il calcolo automatico costituiva una risorsa preziosa a disposizione di pochi utenti.

Tutti questi fattori, assieme al miglioramento delle capacità di calcolo dei computer, **portarono ad un ripensamento del modo di utilizzare i computer** con l'introduzioni delle prime idee di sistema operativo.

VIDEO

Video 0: programmare l’Eniac

[Video 1: EDSAC programmazione]

[Video 3: EDSAC scrittura del nastro]

[Video 4: EDSAC lettura del nastro]

[Video 2: EDSAC output]

1.2. I sistemi operativi nei vecchi mainframe

Le prime librerie di sistema

Il **primo passo** per semplificare le modalità d'uso dei computer e rendere più facile la programmazione fu quello di **realizzare delle librerie con le istruzioni necessarie per eseguire le operazioni più comuni** per la gestione delle periferiche del computer (operazioni di ingresso e uscita dei dati, operazioni di accesso alla memoria dei nastri magnetici, ecc.)

Poiché solitamente queste istruzioni non dipendono dal tipo di problema affrontato, ma sono dipendenti dalle caratteristiche fisiche del computer impiegato, una delle prime idee fu quella di **raggrupparle in librerie riguardanti funzionalità comuni in modo che potessero essere utilizzare da chiunque** aveva la necessità di operare sul computer.

Nella prima generazione di computer (**mainframe**), basata sulle valvole termoioniche, di più non si poteva pensare. Le cose migliorarono con il passaggio alla seconda generazione di computer, a transistor, con una maggiore affidabilità delle macchine.

Lavorare con il computer negli anni ‘50

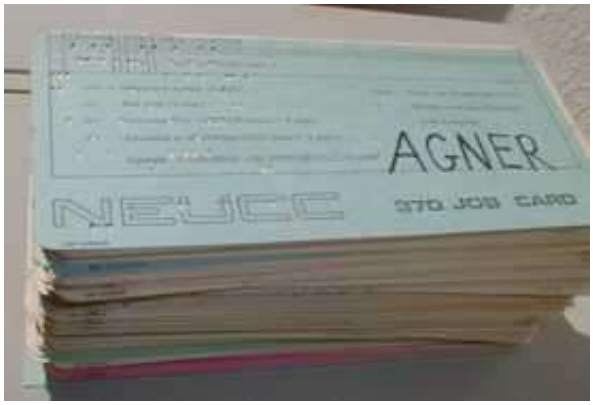


Fig. Un job era costituito da un pacco di schede perforate contenente il programma da eseguire, i dati da elaborare e le schede di controllo per la gestione dell’esecuzione.

Nei primi anni ’50 l’**utente solitamente non interagiva direttamente con il computer**, ma le informazioni erano inserite solitamente nel computer mediante **schede perforate**.

Il programmatore tipicamente

- si limitava a **preparare le schede perforate** su cui era codificato il programma da eseguire;
- **le schede** del programma assieme alle schede con i dati **venivano poi passate all’operatore** di computer;
- l’operatore aveva il compito di caricare le schede, di **eseguire il programma** sul computer e
- di restituire i risultati (su stampa) all’utente.

Più precisamente il programmatore preparava un cosiddetto ***job***, costituito da un pacco di schede, con le istruzioni del programma e i dati da elaborare (input). Per un corretto funzionamento del sistema **il job doveva includere anche schede di controllo necessarie per la gestione delle operazioni di input/output**.

Queste schede di controllo specificano ad esempio che ciò che seguiva era un programma in COBOL, o una sequenza di schede di dati, ecc.

Per istruire il computer sulle operazioni da compiere sui vari job venne introdotto un linguaggio apposito, il **Job Control Language (JCL)**, una sorta di interprete shell in grado di eseguire le operazioni richieste.

Un esempio di job intercalato da schede di controllo è il seguente:

```
$JOB user_spec      ; identify the user for
                      accounting purposes
$FORTRAN            ; load the FORTRAN compiler

   <<source program cards>>

$LOAD               ; load the compiled program
$RUN                ; run the program

   <<data cards>>

$EOJ                ; end of job

$JOB user_spec      ; identify a new user
$LOAD application
$RUN

   <<data>>

$EOJ
```

Il job (in forma di pacco di schede) veniva poi affidato all'**operatore del computer**, che si incaricava di farlo eseguire sul computer non appena la macchina era libera.

I **risultati dell'esecuzione del programma** ovviamente non era immediatamente disponibile all'utente, ma potevano essere riconsegnati all'operatore (in genere, in forma stampata) dopo ore o addirittura dopo giorni.

Se poi il programma conteneva qualche errore, il programmatore riceveva, oltre al programma, solo la stampa di **qualche messaggio di errori e dei dati parzialmente elaborati**. Tutto questo rendeva molto difficile la messa punto del programma.

Fig. Lettore di schede perforate.

Il monitor

Il sistema operativo di questi computer era ancora molto semplice:

- oltre ad includere le librerie con le routine di input/output,
- comprendeva **un semplice programma per la gestione dei job** (detto **monitor**) in modo da aiutare l'operatore del computer nella loro gestione.

Il monitor aveva semplicemente il compito di **caricare nella memoria centrale del computer un job** alla volta e **di passare il controllo dell'unità di elaborazione al job** da eseguire.

In seguito, questo nucleo di sistema operativo fu raffinato in modo che potesse caricare automaticamente in sequenza i diversi job da eseguire.

I monitor possono essere considerati la prima forma primitiva di sistema operativo.

Una piccola porzione della memoria centrale veniva dedicata al monitor che veniva attivato ogni volta che era necessario.

Sistemi batch semplici

I **job venivano raggruppati dall'operatore del computer in gruppi** (denominati **lotti**) a seconda delle caratteristiche. In questo modo **i job simili, che richiedevano le medesime risorse, venivano eseguiti tutti insieme** evitando di dover cambiare continuamente le predisposizioni del computer.

Questo tipo di gestione era denominata **sistema a lotti (batch)**. Il monitor riusciva a gestire i caricamento di un intero lotto di job e l'operatore interveniva solo alla fine dell'intero lotto.

Ad esempio, tutti i programmi scritti in Fortran venivano messi insiemi e distinti da quelli scritti in Cobol, in modo da non dover cambiare continuamente compilatore. **Sulle unità a nastro venivano mantenuti pronti i diversi compilatori di programmi e altri programmi di utilità per l'esecuzione dei job**.

Ogni lotto veniva introdotto nel computer solitamente mediante un lettore di schede perforate e al termine dell'esecuzione dei singoli programmi l'output (in forma cartacea) veniva consegnato all'utente.

Uno dei primi sistemi capace di lavorare con questa gestione fu il GM-NAA I/O System, realizzato su un **IBM 704** nel 1956 nei laboratori della General Motors, evoluzione di un precedente monitor sperimentato su un IBM 701 nel 1955.

Sistemi batch con spooling

L'inconveniente maggiore della gestione a lotti era legato alla grande diversità di velocità dell'unita di elaborazione (CPU) rispetto ai dispositivi di ingresso e uscita.

Mentre la CPU viaggiava al ritmo di milioni di operazioni al secondo, le unità di ingresso e uscita lavorano ad una velocità di 1.000 o anche 100.000 volte più bassa, lasciando la CPU inattiva per gran parte del tempo.

Fino a quel momento **tutte le operazioni di I/O avvenivano con la tecnica a controllo di programma**, impegnando di fatto la CPU nel trasferimento dei dati con il conseguente rallentamento dell'esecuzione dei job.

Un altro inconveniente era legato al fatto che la sequenza di esecuzione dei job era legata semplicemente al modo con cui l'operatore aveva caricato sul lettore i diversi pacchi di schede. **Il sistema operativo non poteva alterare in alcun modo quest'ordine per rendere più efficiente l'esecuzione dell'intero lotto di programmi**.

Un primo passo in avanti, che contribuì a migliorare i primi sistemi operativi, fu quello di **sfruttare il disco rigido**, introdotto a metà degli anni ’50 (dall’IBM), **per gestire in modo più rapido i vari job**.

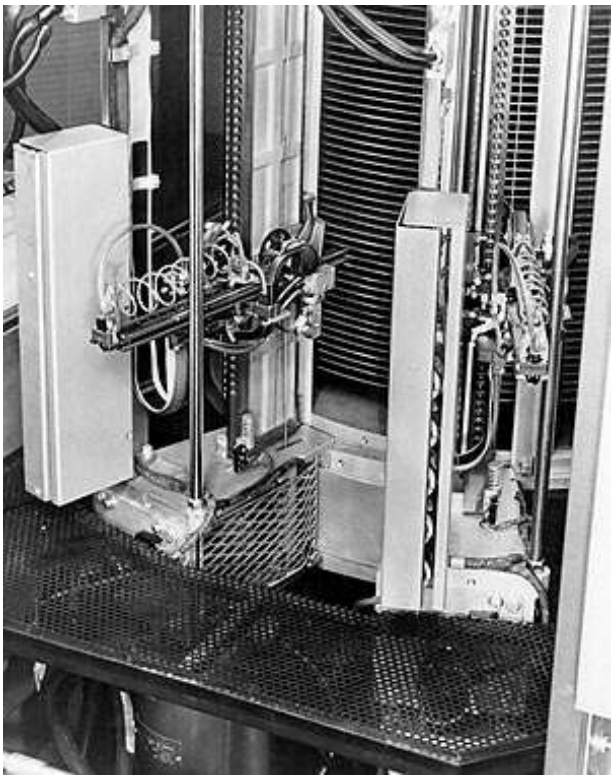


Fig. Una delle prime unità a dischi, l’IBM 350 (,).

Questa memoria

- non solo **consentiva tempi di accesso molto più brevi** rispetto a un lettore di schede o ad un nastro magnetico, ma
- consentiva di realizzare un **accesso diretto alla memoria** consentendo di superare le limitazioni dell’accesso sequenziale delle unità a nastro magnetico.

Così **mentre la CPU eseguiva un dato programma,**

- **i job non caricati potevano essere trasferiti dal lettore di schede al disco rigido e**
- **quando la CPU si rendeva disponibile,**
  - **ciascun job poteva essere trasferito in memoria centrale**.

Una cosa analoga poteva essere fatta per i dati in output: invece di stampare i risultati durante l'esecuzione del programma impegnando la CPU, questi potevano essere temporaneamente memorizzati sul disco rigido (con tempi molto più brevi di quelli impiegati da una stampante) e dal disco rigido possono essere trasferiti alla stampante senza richiedere il controllo diretto della CPU, che nel frattempo poteva fare altro.

Questo tipo di gestione era chiamato ***spooling*** (acronimo di ***simultaneous peripheral operation on-line***) ed ha rappresentato la **prima forma di parallelismo nell'impiego delle diverse unità del computer**: lo spooling consentiva di sovrapporre le operazioni di I/O di un job con la computazione di altri job.

Fig. Sebastian de Ferranti and Tom Kilburn at the Atlas Console.

Lo spooling venne sperimentato (almeno secondo alcuni storici) per la prima volta sul **sistema Atlas** sviluppato presso l'Università di Manchester nei primi anni '60.

Sempre agli inizi degli anni '60 l'IBM implementò la tecnica dello spooling nel sistema operativo IBSYS con i sistemi **IBM 7090 e IBM 7094** utilizzando unità a nastri e usando come sistema ausiliario un piccolo IBM 1401.

La multiprogrammazione nei sistemi batch

Ben presto l'idea di caricare i job dalle schede al disco rigido prima di essere eseguiti dalla CPU portò a raffinare i sistemi operativi per la gestione dei job.

Infatti, **una volta che i job venivano registrati su disco rigido, era possibile elaborarli non secondo una rigida sequenza prestabilita eseguendoli uno dopo l'altro, ma era possibile eseguirli secondo politiche di gestione più raffinate in modo da aumentare l'utilizzo della CPU**.

Queste idee portarono negli anni '60 ad introdurre il concetto di **multiprogrammazione**.

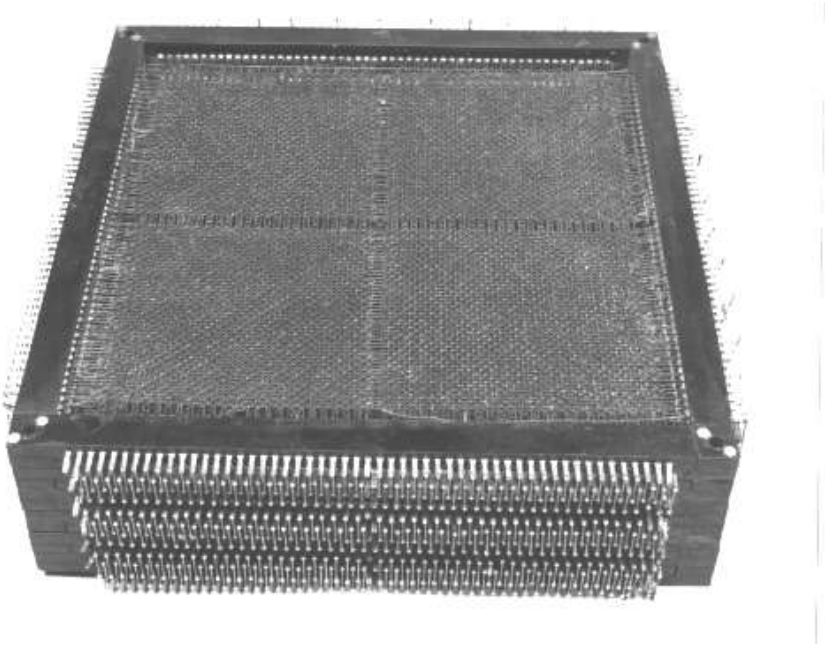


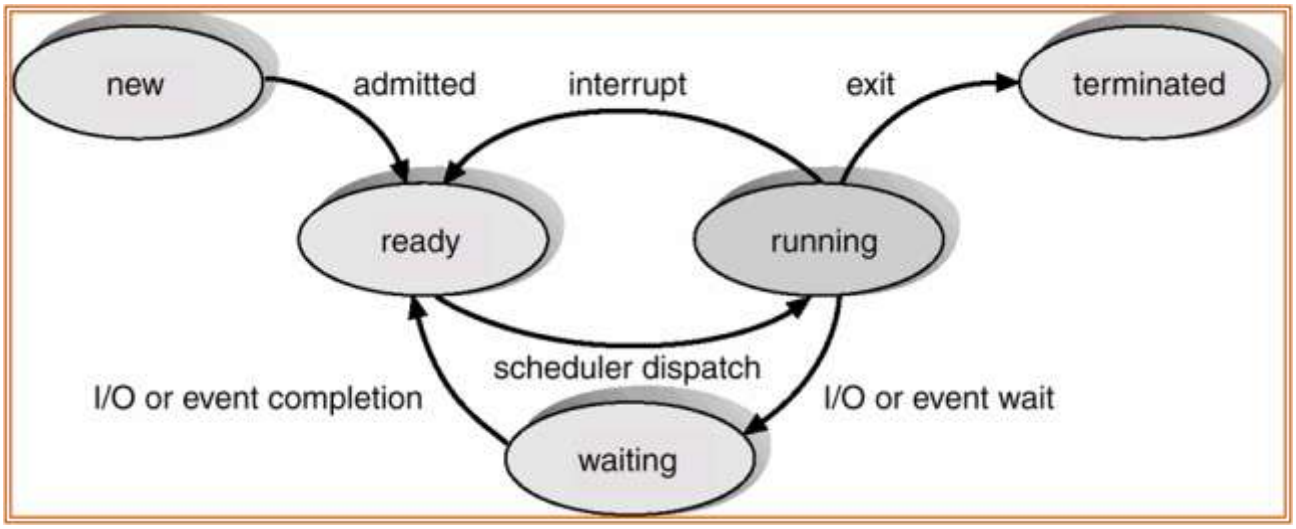
Fig. Blocco a sei piani di memoria a nuclei.

L’**idea della multiprogrammazione** divenne possibile negli anni ’60 quando la dimensione della memoria centrale aumentò sensibilmente grazie all’**introduzione delle memorie a nucleo**.

Secondo questa tecnica,

- in **memoria centrale vengono tenuti contemporaneamente alcuni programmi e i loro dati** pronti per l’esecuzione.
- In un dato momento, **uno solo** di questi programmi (più precisamente **processi**) **viene eseguito**,
- ma non appena il programma in esecuzione richiede un’istruzione (molto lenta) di ingresso o uscita, il **programma in esecuzione viene sospeso** attivando le unità periferiche necessarie per eseguire l’istruzione di ingresso e uscita.
- Contemporaneamente, **il sistema operativo cede la CPU ad uno degli altri programmi** già residenti in memoria centrale. In questo modo, la CPU è sempre impegnata con uno dei job in memoria centrale.

Il collo di bottiglia più significativo per questa tecnica era ovviamente rappresentato dalla capacità della memoria centrale: solo i sistemi con una sufficiente memoria centrale potevano funzionare in modo adeguato.



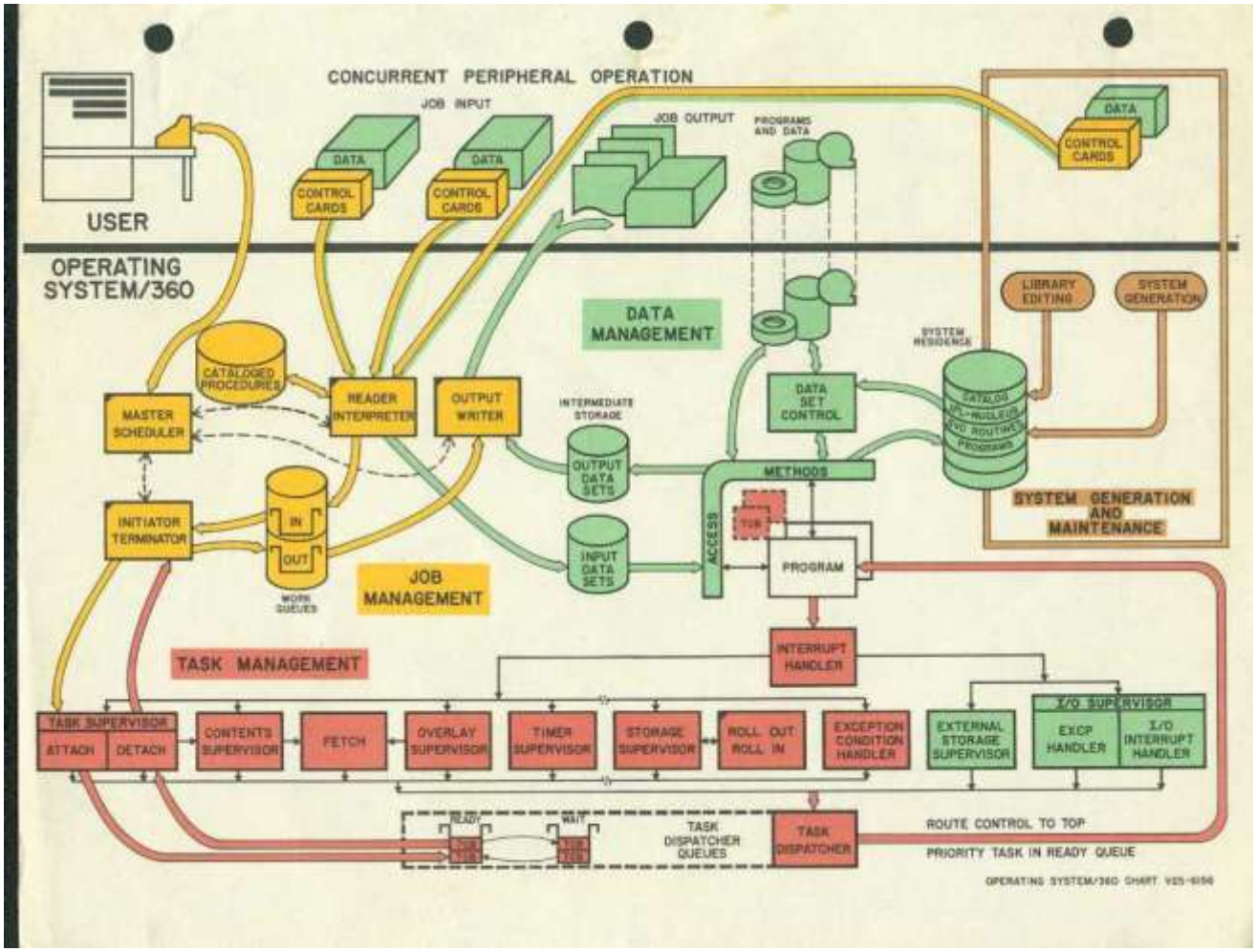
L'introduzione della multiprogrammazione portò anche a **studiare le prime strategie utili per la scelta più efficace dei programmi da eseguire** in modo da sfruttare al meglio la poca memoria centrale disponibile e da rendere più rapida l'esecuzione complessiva dei programmi.

Pertanto **il sistema operativo doveva prendere decisioni su quali job caricare in memoria centrale e quale job eseguire in ogni dato istante**.

Da questo momento in poi la struttura e il funzionamento dei sistemi operativi cominciarono a diventare sempre più complessi e sofisticati.

Uno dei primi sistemi ad utilizzare la multiprogrammazione fu il **sistema OS/360** realizzato per la serie di computer **IBM 360**. Fra l'altro questo sistema operativo va ricordato per due motivi importanti:

- fu il primo tentativo di **realizzare un sistema operativo uniforme e compatibile per tutta una serie di macchine IBM** molto diverse tra loro per l'hardware sottostante; fino a quel momento ogni macchina aveva un proprio sistema operativo diverso da quello presente in altri computer;
- **lo sviluppò di questo sistema operativo si rivelò inoltre un'impresa estremamente complessa**, che spronò gli studiosi ad occuparsi per la prima volta delle problematiche relative **all'ingegneria del software**.


CONCURRENT PERIPHERAL OPERATION
JOB INPUT
JOB OUTPUT
PROGRAMS AND DATA
DATA
CONTROL CARDS
USER
OPERATING SYSTEM/360
DATA MANAGEMENT
LIBRARY EDITING
SYSTEM GENERATION
SYSTEM RESIDENCE
CATALOGED PROCEDURES
READER INTERPRETER
OUTPUT WRITER
INTERMEDIATE STORAGE
DATA SET CONTROL
MASTER SCHEDULER
OUTPUT DATA SETS
METHODS
INITIATOR TERMINATOR
IN
OUT
WORK QUEUES
INPUT DATA SETS
ACCESS
TCB
PROGRAM
SYSTEM GENERATION AND MAINTENANCE
JOB MANAGEMENT
TASK MANAGEMENT
INTERRUPT HANDLER
TASK SUPERVISOR
ATTACH
DETACH
CONTENTS SUPERVISOR
FETCH
OVERLAY SUPERVISOR
TIMER SUPERVISOR
STORAGE SUPERVISOR
ROLL OUT ROLL IN
EXCEPTION CONDITION HANDLER
EXTERNAL STORAGE SUPERVISOR
I/O SUPERVISOR
EXCP HANDLER
I/O INTERRUPT HANDLER
READY
WAIT
TASK DISPATCHER QUEUES
TASK DISPATCHER
ROUTE CONTROL TO TOP
PRIORITY TASK IN READY QUEUE
OPERATING SYSTEM/360 CHART

Il time-sharing e i primi sistemi interattivi

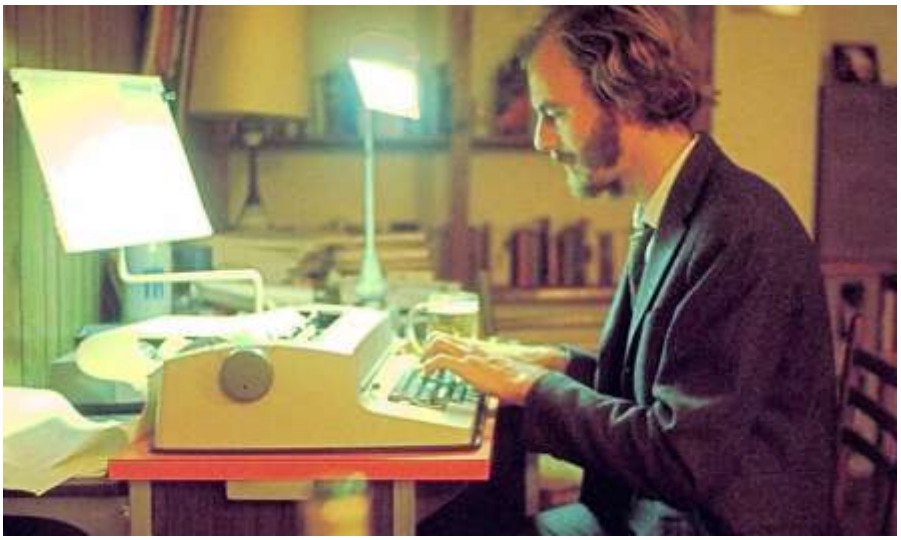


Fig. Un terminale cartaceo per interagire con il computer (al posto del monitor è presente una stampante, 1973).

Se la tecnica della multiprogrammazione aveva migliorato di molto lo sfruttamento della CPU, poco aveva fatto sul versante dell'interattività dei computer.

Il **sistemi di batch e multiprogrammati avevano l'inconveniente di offrire una scarsa interazione con l'utente**:

di fatto **l'utente consegnava il pacco di schede perforate** (con i dati e il programma) all'operatore del computer e **poteva rivedere solo alcune ore più tardi (o il giorno successivo) i risultati** in forma cartacea ad esecuzione conclusa.

Questa modalità sebbene consentisse di affrontare con il computer problemi anche molto complessi, risultava assai scomoda per tutte quelle attività che richiedevano una costante interazione con il computer.

Ad esempio, l'impiegato di uno **sportello di una banca**, pur facendo operazioni molto semplici, aveva bisogno di accedere costantemente ai dati e alle risorse di calcolo del computer senza dover passare attraverso tutta la trafila caratteristica dell'elaborazione batch a schede.

Fig. Terminale video, DEC VT 100.

Questo tipo di esigenze portò a **concepire nuove forme di interazione con il computer più rapide** di quelle batch, basate sul tradizionale uso delle schede perforate.

- Da un lato fu necessario **sostituire l'uso delle schede con l'uso di appositi terminali costantemente collegati al computer**;
- dall'altro lato fu necessario **cambiare le modalità di gestione stessa dell'unità centrale** modificando i sistemi operativi esistenti.

L'interazione ora poteva avvenire attraverso **terminali costituiti da una coppia tastiera-stampante** (nei modelli più primitivi), oppure da una **coppia tastiera-monitor** (nei modelli più avanzati).

Dal momento, che un singolo computer non poteva essere messo a disposizione di un singolo utente visti gli alti costi di queste macchine, **fu necessario escogitare una modalità di funzionamento in grado di servire contemporaneamente molti utenti**.

I sistemi con time-sharing furono sviluppati per garantire l'uso interattivo di un sistema di calcolo contemporaneamente a molti utenti a un costo ragionevole.

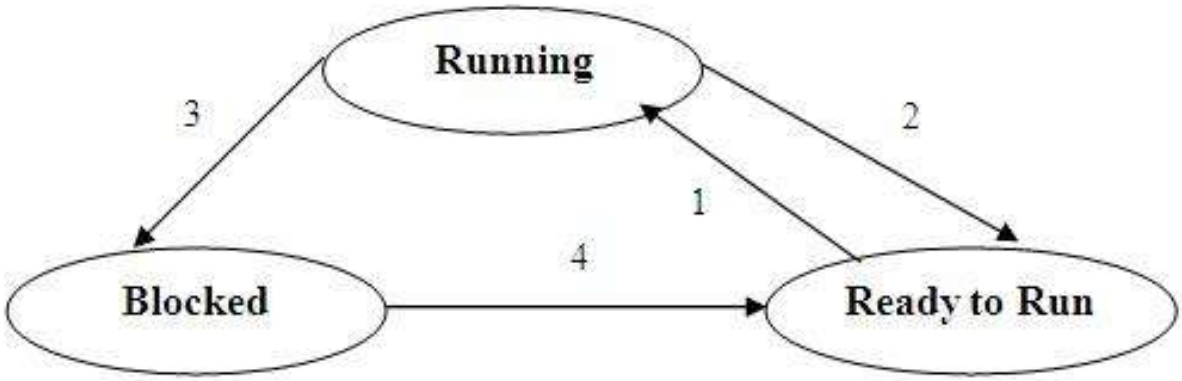
L'obiettivo

- non era più quello di garantire la massima efficienza nell'esecuzione dei programmi,
- ma dare all'utente un minimo di interazione continua con il computer.

L'idea per giungere a tale tipo di modalità derivò da un **perfezionamento della tecnica della multiprogrammazione**.

Nella multiprogrammazione il cambio tra un processo e l'altro avviene solo quando il processo in esecuzione richiede un'operazione di input/output.

Il time-sharing prevede invece di assegnare un piccolo intervallo di tempo (time slice) a ciascuno dei processi residenti in memoria in modo tale che ogni processo o si interrompe per un'operazione di input/output, oppure si ferma al termine del proprio intervallo di tempo. Il processo sospeso viene poi rimesso in esecuzione quando anche gli altri processi hanno esaurito il loro intervallo di tempo e così via in modo ciclico.



Transitions

1→ The process is scheduled
2→ Time-slice gets over
3→ Process has to wait for an event
4→ Event has occurred

In questa maniera, **l'esecuzione di ciascun processo può procedere un po' alla volta dando l'impressione a ciascun utente di avere il computer tutto per sé**. Infatti, se l'intervallo di esecuzione per ciascun processo è sufficientemente piccolo, si ha l'impressione che tutti i programmi in memoria centrale stiano avanzando in parallelo. Ad esempio, se a ciascuno dei dieci processi viene assegnato 1/50 di secondo della CPU, ogni utente può interagire con il computer ad intervalli di 1/5 di secondo.

Tenendo conto che l'utente è abbastanza lento nella lettura sullo schermo e nella digitazione da tastiera, egli può lavorare quasi come se fosse il solo utente del computer.

Fig. Bob Bemer.

Il **concetto di time-sharing compare per la prima volta nel 1957**, quando viene descritto in un articolo di Bob Bemer sulla rivista *Automatic Control Magazine*.

Il primo progetto per l'implementazione di questo meccanismo fu intrapreso **presso il MIT** sotto la guida di John McCarthy, Robert Fano e Fernando Corbato e portò a realizzare **nel 1961** il **sistema sperimentale CTSS in grado di funzionare in time-sharing** su computer IBM 7090 e 7094.

Video: intervista Robert Fano sul time-sharing.

I presupposti principali per poter realizzare questo tipo di modalità di funzionamento furono due:

- la **memoria centrale doveva essere abbastanza capiente** per poter accogliere i programmi degli utenti e
- la **CPU doveva essere sufficientemente veloce** per poter eseguire comunque un gran numero di operazioni nel quanto di tempo previsto per ciascun programma.

Pertanto, questa tecnica, sebbene sperimentata nel corso degli anni ’60, **divenne una modalità consolidata solo negli anni ’70** quando le performance dei computer migliorarono.

L’esigenza di avere una memoria ampia per gestire i diversi programmi **rese ancora più complicato il problema della gestione di memoria** spingendo i tecnici ad elaborare tecniche per estendere in qualche modo la memoria centrale.

Il time sharing con l’utilizzo contemporaneo delle risorse del computer da parte di più programmi pose inoltre anche le **prime problematiche relative alla sicurezza e alla protezione delle informazioni** in modo che il lavoro di ciascun utente non potesse creare problemi agli altri utenti.

GE-400 Time-sharing Information Systems

Bring a powerful computer to the fingertips of all your people

Il file system

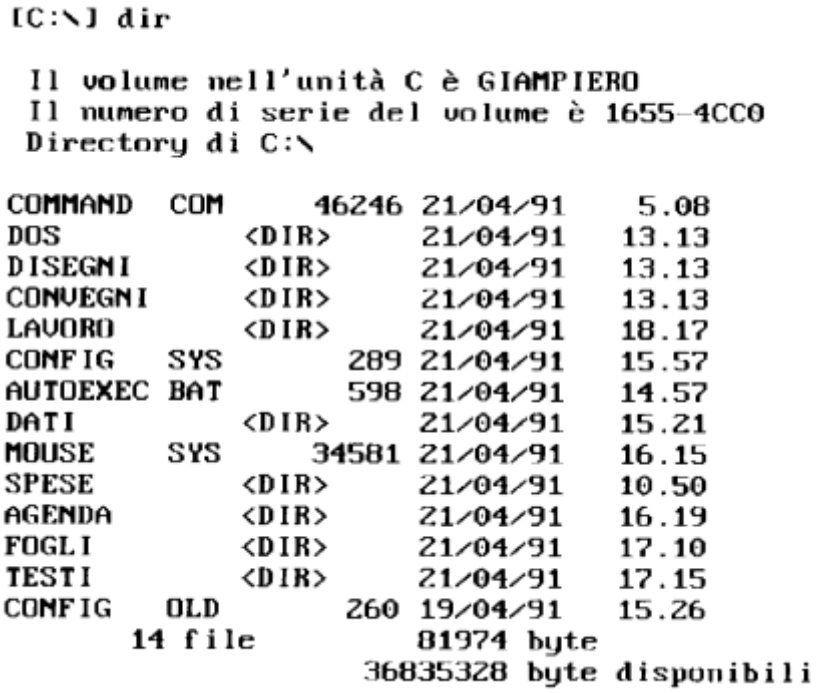


Fig. Elenco di file e cartelle in un sistema gestito mediante il DOS.

L'introduzione di memorie di massa capienti come quelle basate su nastri magnetici e sul disco rigido ben presto portò alla necessità di gestire le informazioni mediante quello che oggi è noto come concetto di file.

Un **file** costituisce un insieme di informazioni solitamente della stessa natura e logicamente correlate.

Nella maggior parte dei casi, un file contiene un programma (programma sorgente o programma eseguibile), oppure una sequenza di dati più o meno omogenea, come ad esempio, dati numerici, alfabetici o alfanumerici.

Il termine **file** (che in inglese significa ***archivio***) compare nei primi anni ’50 e inizialmente si riferiva a un pacco di schede contenente informazioni omogenee.

E’ il sistema operativo a realizzare il concetto astratto di file gestendo i dispositivi di memoria di massa, come nastri e i dischi e facendo apparire le informazioni del file all’utente in forma aggregata.

Normalmente i file sono organizzati in raccolte specifiche (note come **directory o cartelle**), che ne facilitano l’individuazione e l’accesso.

L’**intera gestione dei file è a carico di un componente del sistema operativo denominato *file system***. Questa fornisce all’utente anche i comandi specifici per manipolare i file consentendo operazioni come la copiatura, la cancellazione, l’accesso sequenziale o diretto, ecc.

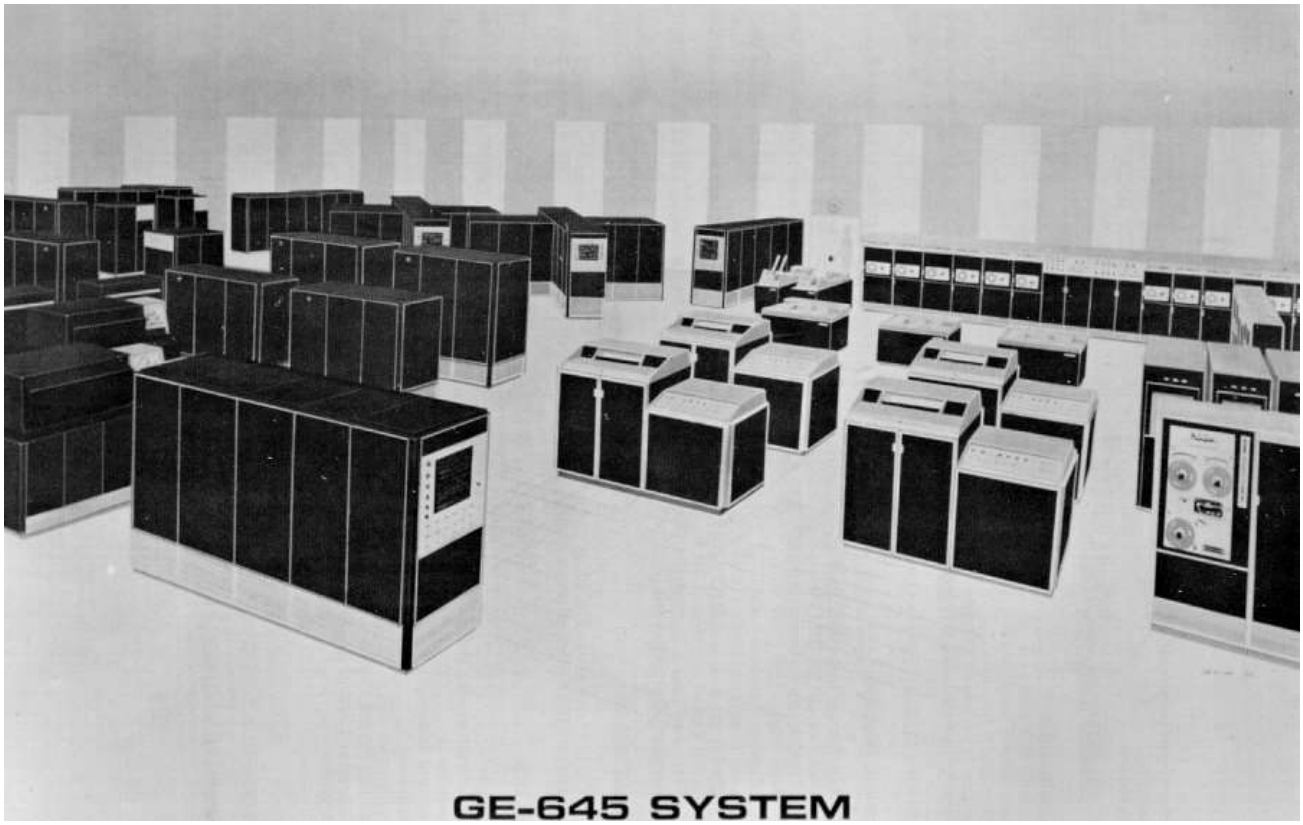


Fig. Il computer della General Electric GE-645, istallato presso il MIT (1968), su cui funzionava il sistema Multics.

La **struttura multilivello dei file** venne implementata per la prima volta sul sistema operativo **MULTICS**, realizzato attorno alla metà degli anni ’60 da una collaborazione del MIT con i laboratori della Bell e con il Computer Department della General Electric.

Questo sistema operativo fra l'altro era estremamente innovativo anche perché:

- rappresentava uno dei primi sistemi ad offrire il **time-sharing**;
- sperimentava le prime **tecniche di allocazione dinamica della memoria** per la realizzazione della memoria virtuale;
- adottava il **concetto moderno di processo** (in stato pronto, in esecuzione, o bloccato);
- offriva i primi **meccanismi di comunicazione tra processi**.
- offriva **meccanismi innovativi per protezione** delle informazione tra processi;

Successivamente **questa struttura a multilivello**, che consentiva l'organizzazione dei file in cartelle e sottocartelle, **venne adottata nel sistema operativo Unix**, su MS-Dos e su numerosi altri sistemi operativi ed oggi fa parte delle normali caratteristiche di un sistema operativo.

Memoria virtuale

In un computer, l'esecuzione di un programma richiede che il programma e i dati da esso elaborati siano collocati nella memoria centrale.

Poiché la capienza della memoria centrale, specialmente nel passato, era abbastanza limitata a causa degli alti costi, **spesso sorgevano grosse difficoltà per gestire programmi che necessitavano grandi quantità di memoria** o per gestire in memoria più programmi contemporaneamente.

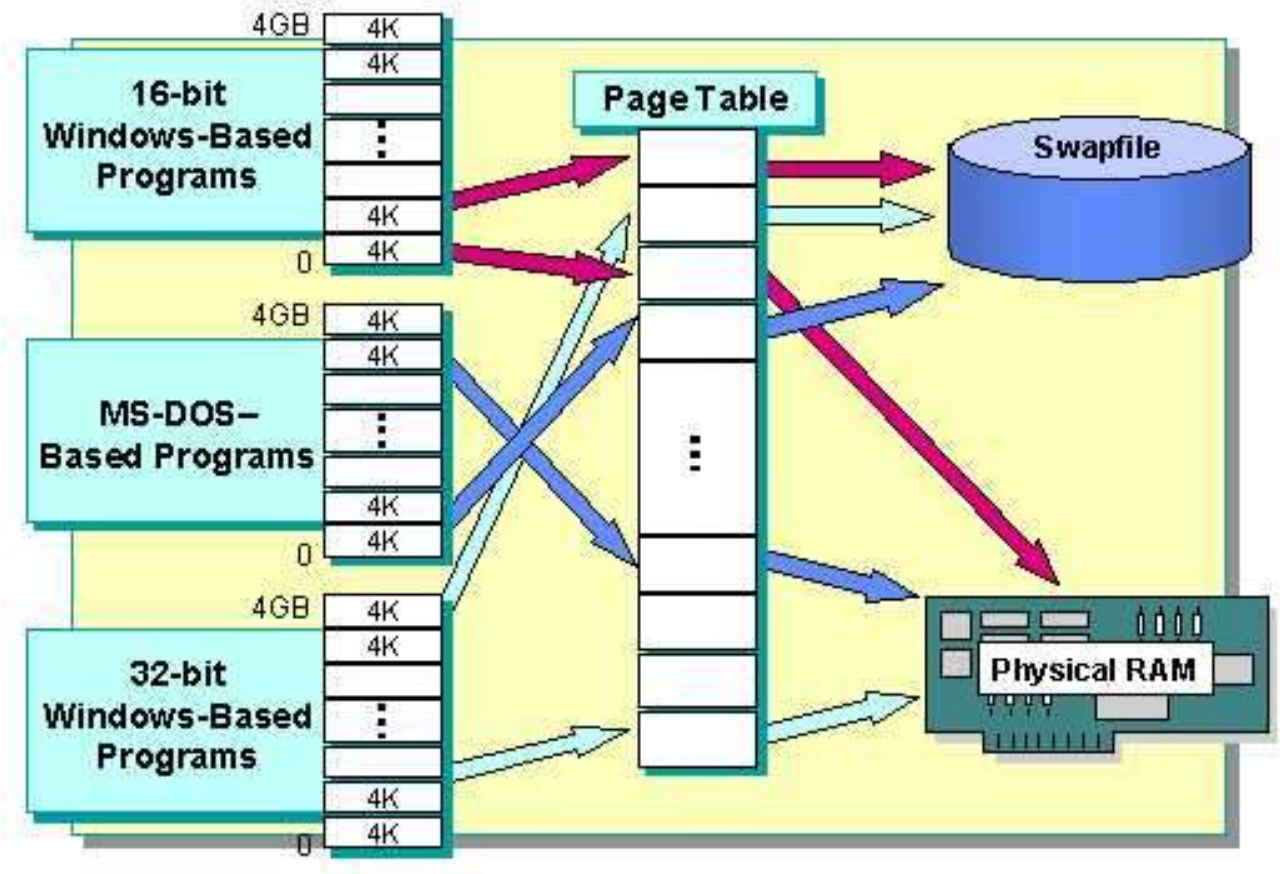
Per superare questa limitazione, un altro dei miglioramenti introdotti nei sistemi operativi fu quello della **memoria virtuale**.

Con questa tecnica è possibile simulare la presenza di una memoria centrale molto più grande di quella effettivamente disponibile.

L'utente non si accorge di questo meccanismo poiché tutta la gestione della memoria viene svolta "di nascosto" dal sistema operativo.

In particolare, **la memoria centrale viene "simulata" utilizzando la memoria di massa** e più precisamente dal disco rigido.

In questo modo, è possibile lanciare programmi che richiedono grandi quantità di memoria senza che l'utente si debba preoccupare delle reali limitazioni della memoria centrale.



Naturalmente, poiché la memoria del disco rigido è molto più lenta di quella centrale, il **prezzo da pagare per questa soluzione è quello di rallentare l'esecuzione dei programmi**.

Questa scelta infatti rappresenta un compromesso tra quantità di memoria e velocità:

- da un lato è possibile **dotare il computer di grande quantità di memoria** di massa poiché questa ha costi molto più bassi di quella centrale,
- ma dall'altro lato **la velocità di esecuzione del computer è più bassa** rispetto ad una reale disponibilità di memoria centrale poiché il disco rigido ha tempi di accesso molto più elevati della memoria centrale.

Adottando comunque delle **"buone politiche" di gestione per l'implementazione del meccanismo della memoria virtuale**, la performance complessiva del sistema rimane comunque abbastanza buona e sufficiente per molte applicazioni.

Per virtualizzare la memoria si diffusero due tecniche alternative:

- la **paginazione** con la gestione di blocchi di memoria di dimensione fissa e
- la **segmentazione** con blocchi di dimensione variabile.

Fig. Sebastian de Ferranti and Tom Kilburn at the Atlas Console.

I primi esperimenti dell'uso di memoria virtuale risalgono circa al 1959 con la realizzazione del sistema ATLAS, completato nel 1962 presso l'Università di Manchester in Gran Bretagna. Su questo computer fra l'altro furono sperimentate anche interessanti tecniche di uso del microcodice.

Nel 1961, il **computer B5000 della Burroughs** fu uno dei primi (o forse il primo) computer commerciali dotato di memoria virtuale (utilizzando la segmentazione invece che la paginazione).

Fig. Borroughs B5000 (video).

Inizialmente le difficoltà implementative di questa tecnica ne limitarono la diffusione, ma negli anni ’70 cominciò a diffondersi sempre di più nei grandi computer e infine nel 1992 venne introdotta nel sistema operativo Microsoft Windows 3.1 per i personal computer.

1.3. Sistemi operativi per piccoli computer

Sistema operativo Unix

Fig. Ken Thompson e Dennis Ritchie.

Tra i numerosi sistemi operativi sviluppati nel corso degli ultimi cinquant'anni, il sistema Unix è stato uno di quelli che ha maggiormente influenzato questo settore dell'informatica.

Il sistema Unix venne sviluppato alla fine degli anni '60 presso i laboratori della Bell Telephones e i principali artefici di questo sistema furono Ken Thompson e Dennis Ritchie.

La filosofia di base che caratterizzò il progetto fu quella di **realizzare un sistema semplice** (rispetto ad altri sistemi operativi in uso sui grandi mainframe del tempo) **adatto per la ricerca e lo sviluppo**. Unix venne infatti realizzato in pochi mesi **nel 1969** e i primi esperimenti furono fatti su un minicomputer, il **PDP-7**.

Mentre la prima versione fu scritta in linguaggio Assembly (come veniva allora fatto per tutti i sistemi operativi) ed era quindi dipendente dal tipo di macchina impiegata, per quelle successive **Ritchie pensò di utilizzare un linguaggio ad alto livello, il linguaggio C**, che egli stesso progettò appositamente per il sistema Unix.

In questo modo, **solo una minima parte del sistema venne implementata in Assembly**, mentre la gran parte del sistema venne realizzata in **linguaggio C**, rendendo molto più facile la stesura dei programmi e rendendo il sistema operativo facilmente portabile su macchine di tipo diverso **senza dipendere eccessivamente dalle caratteristiche dell'hardware su cui veniva fatto funzionare**. Di fatto, era la prima volta che ciò succedeva per un sistema operativo.

Fig. Ken Thompson e Dennis Ritchie.

Molte idee di Unix erano derivate dal sistema Multics:

- il concetto di processo
- la multiprogrammazione
- la gestione degli utenti con il time-sharing
- l'uso del concetto di file per la rappresentazione di ogni risorsa del computer
- l'organizzazione gerarchica del file system
- il sistema di protezione dei dati

Diversamente dalle consuetudini del tempo, l'**azienda AT&T distribuì Unix nelle università e rese disponibili i codici sorgenti** utilizzati per realizzare il sistema.

Questa mossa **spinse molti ricercatori delle università a sperimentare questo sistema**, che peraltro inglobava anche numerose idee innovative nell'ambito dei sistemi operativi e in un decennio divenne uno standard di riferimento per lo studio e lo sviluppo per molti programmatori del tempo.

Versioni più raffinate e stabili vennero introdotte a poco a poco con il contributo stesso delle università, fino ad arrivare ad una versione standard approvata da un comitato apposito.

VIDEO

Video 1: April 1999 -- Dennis Ritchie and Ken Thompson of Bell Labs received the U.S. National Medal of Technology from President Bill Clinton at ceremonies televised at the White House.

I primi sistemi operativi per microcomputer

Fig. Altair 8800.

Nel 1975 fu messo in commercio il primo microcomputer, l'Altair 8800, basato sui microprocessori da poco introdotti sul mercato dalla Intel.

A questo modello di microcomputer seguirono in breve numerosi altri modelli sviluppati da aziende diverse. Dal momento che **le risorse di calcolo di tali macchine erano molto limitate**, non fu possibile impiegare un sistema operativo già esistente (come, ad esempio, il sistema Unix), ma **fu necessario realizzare sistemi operativi appositamente progettati per sfruttare i pochi Kbyte di memoria** e le poche risorse disponibili.

Poiché questi microcomputer era pensati come strumenti per gli appassionati e non per il personale tecnico esperto, **era necessario che il sistema operativo fornisse una modalità d'uso relativamente semplice**.

In questo campo, si distinsero **Bill Gates e Paul Allen**, i quali incominciarono la loro attività scrivendo il **linguaggio di programmazione Basic** per il personal computer Altair. La loro cooperazione li portò a fondare nel 1975 una ditta, che denominarono **Microsoft**.

Fig. La squadra della Microsoft, circa 1978.

Fig. Steve Wozniak e Steve Jobs

Un altro microcomputer molto popolare tra la fine degli anni ’70 e gli inizi degli ’80 fu l’**Apple (I e II)** sviluppato da **Steve Wozniak e Steve Jobs**.

Alla fine del 1977 per l’Apple II fu sviluppato un sistema operativo più semplice ed efficiente di quello usato dall'Altair. Il sistema operativo venne realizzato da Wozniak e Jobs e si ispirava vagamente al sistema Unix.

Fig. Gary Kildall

Un altro contributo importante nei sistemi operativi per microcomputer venne da **Gary Kildall**, il quale implementò nel 1976 il **sistema CP/M** (Control Program for Micros). In particolare, il sistema consentiva di gestire e registrare facilmente file su floppy disk. Questo sistema si dimostrò migliore di altri sistemi sviluppati negli stessi anni e fu adottato in molti microcomputer.

I sistemi operativi dei microcomputer **dovevano essere molto più semplici di quelli impiegati nei grandi computer**, poiché la macchina veniva utilizzata da un singolo utente e le periferiche collegate erano poche e molto semplici.

Il problema maggiore era quello di gestire i file su floppy disk o su nastri magnetici e mettere a disposizione dell'utente un linguaggio di programmazione semplice, come il Basic.

Fig. Una videata del computer Commodore 64, uno dei microcomputer più diffusi in Italia negli anni ’80. L’interfaccia del sistema poteva comprendere sia comandi Basic che comandi tipici di un sistema operativo.

Molto **spesso il confine tra linguaggio di programmazione e sistema operativo non era ben delimitato** e quando il sistema veniva avviato, era pronto per ricevere sia comandi del sistema operativo, sia istruzioni in Basic.

Il Dos per i personal computer

Fig. Il primo personal computer dell'IBM.

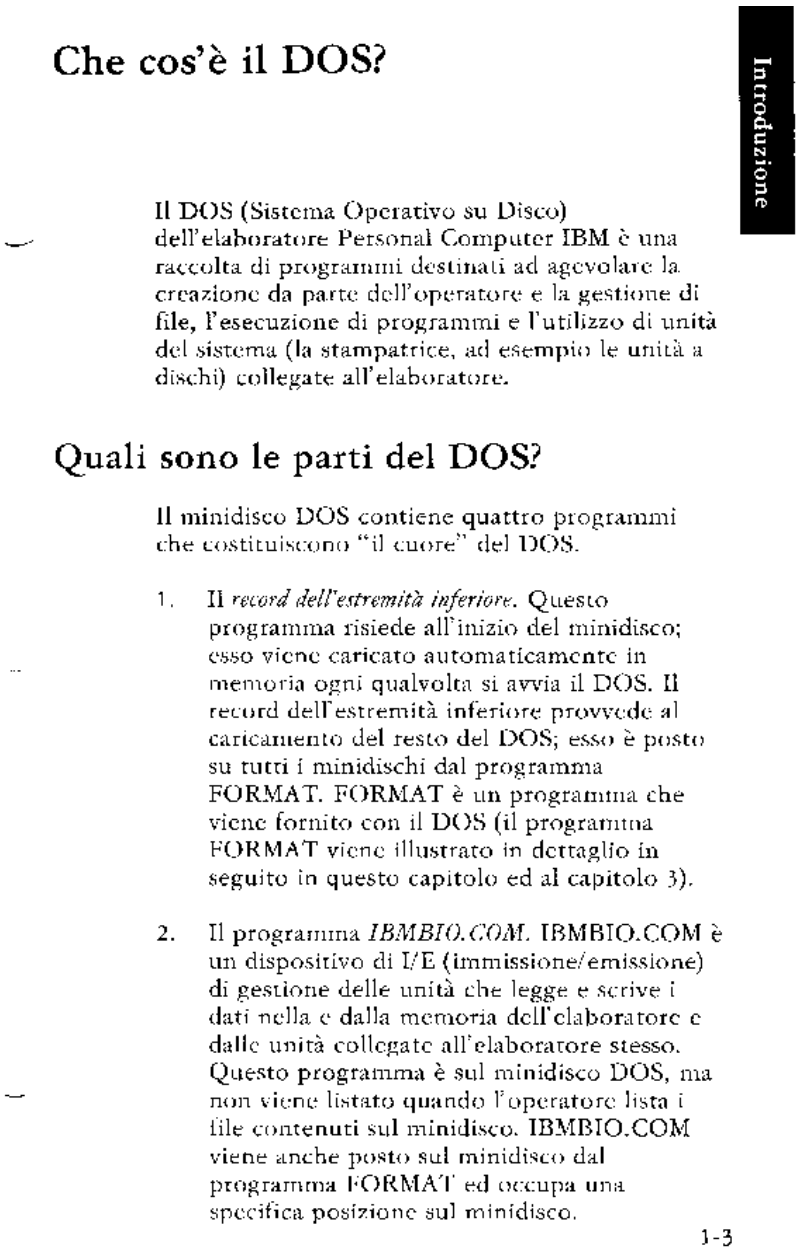
Il successo dei microcomputer spinse l'IBM, che fino ad allora si era occupata di grandi computer e software, ad entrare in campo nel 1980-81 con l'**introduzione del personal computer IBM PC**.

Il computer introdotto dall'IBM adottava un microprocessore (l'**Intel 8086**) un po' più potente di quelli presenti nei primi microcomputer, poiché l'ambizione di questa azienda era quella di realizzare un sistema da utilizzare non solo per gli appassionati e per i giochi (principale fattore che aveva determinato il successo dei microcomputer), ma di **realizzare uno strumento effettivamente utile per lo studio, per la casa e per la gestione dei dati di una piccola azienda o di un professionista**.

Fig. Bill Gates.

In quel momento, il sistema operativo migliore per i microcomputer era il sistema CP/M, ma alcuni disguidi impedirono all'IBM di fare un contratto con Gary Kildall, autore del CP/M. L'**IBM incaricò Bill Gates di realizzare un sistema operativo per il nuovo personal computer**.

Il successo dell'IBM PC decretò la fortuna della Microsoft, i cui profitti cominciarono a crescere in modo esponenziale.



Introduzione

Che cos'è il DOS?

Il DOS (Sistema Operativo su Disco) dell'elaboratore Personal Computer IBM è una raccolta di programmi destinati ad agevolare la creazione da parte dell'operatore e la gestione di file, l'esecuzione di programmi e l'utilizzo di unità del sistema (la stampatrice, ad esempio le unità a dischi) collegate all'elaboratore.

Quali sono le parti del DOS?

Il minidisco DOS contiene quattro programmi che costituiscono "il cuore" del DOS.

1. Il *record dell'estremità inferiore*. Questo programma risiede all'inizio del minidisco; esso viene caricato automaticamente in memoria ogni qualvolta si avvia il DOS. Il record dell'estremità inferiore provvede al caricamento del resto del DOS; esso è posto su tutti i minidischi dal programma FORMAT. FORMAT è un programma che viene fornito con il DOS (il programma FORMAT viene illustrato in dettaglio in seguito in questo capitolo ed al capitolo 3).

2. Il programma *IBMBIO.COM*. IBMBIO.COM è un dispositivo di I/E (immissione/emissione) di gestione delle unità che legge e scrive i dati nella e dalla memoria dell'elaboratore e dalle unità collegate all'elaboratore stesso. Questo programma è sul minidisco DOS, ma non viene listato quando l'operatore lista i file contenuti sul minidisco. IBMBIO.COM viene anche posto sul minidisco dal programma FORMAT ed occupa una specifica posizione sul minidisco.

1-3

Fig. Una pagina del manuale DOS versione 1.10.

Il sistema realizzato dalla Microsoft venne denominato **MS-Dos** e grazie alla standardizzazione dei personal computer lanciata dall'IBM, **divenne il sistema operativo più diffuso al mondo**, poiché ne furono venduti milioni di copie.

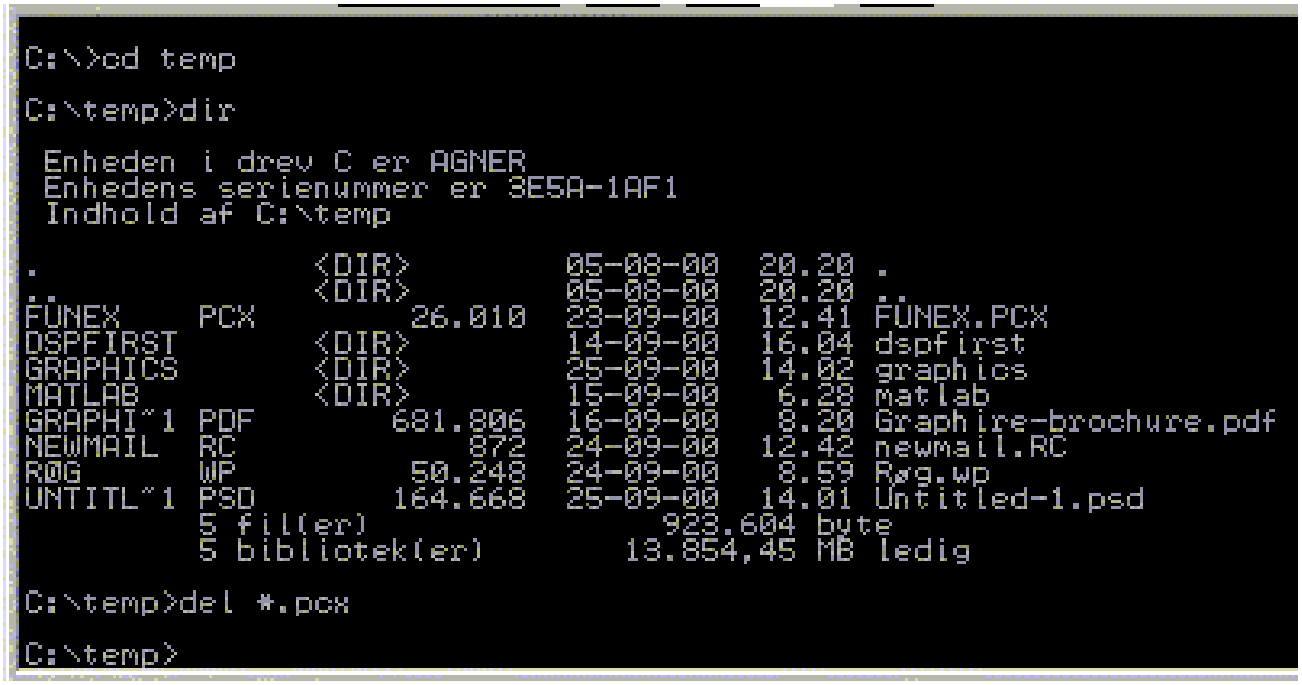


Fig. Una tipica videata di un PC con il sistema operativo MS-DOS.

Il sistema MS-Dos non era molto facile da usare rispetto agli attuali standard; infatti, l'**utente interagiva con il computer attraverso comandi testuali**, la cui sintassi non era semplice da ricordare.

Per questa ragione, qualche anno più tardi incominciò ad apparire sul mercato una nuova proposta per rendere più amichevole il dialogo con il personal computer.

VIDEO

Video 1: documentario sulla nascita del DOS

Video 2: operazioni con il Dos

L'interfaccia grafica del Macintosh

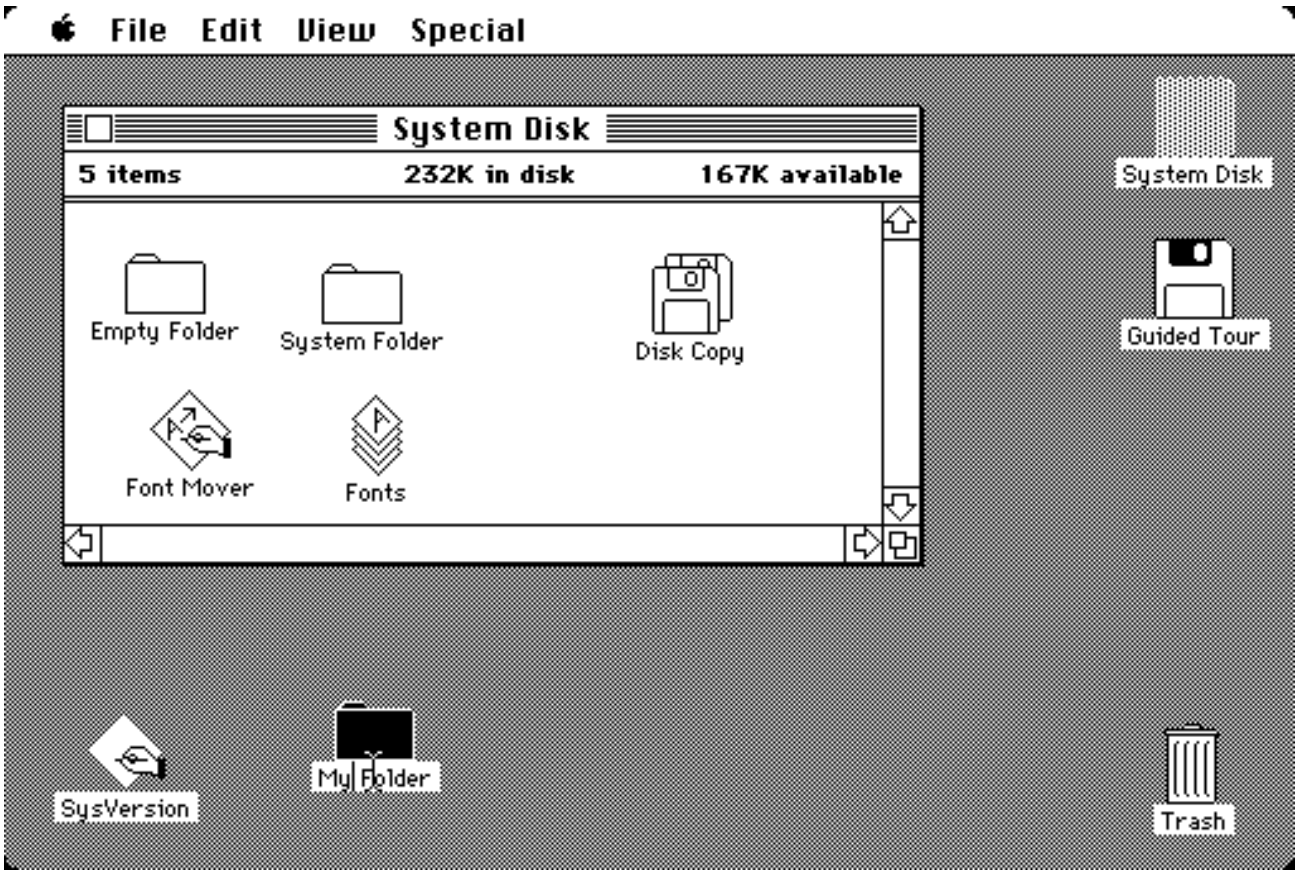
Come abbiamo detto, il sistema operativo MS-Dos era scomodo e poco intuivo da utilizzare.

Un passo fondamentale per rendere più facile e amichevole il dialogo con il computer venne compiuto a metà degli anni ’80 dalla Apple di Steve Jobs.

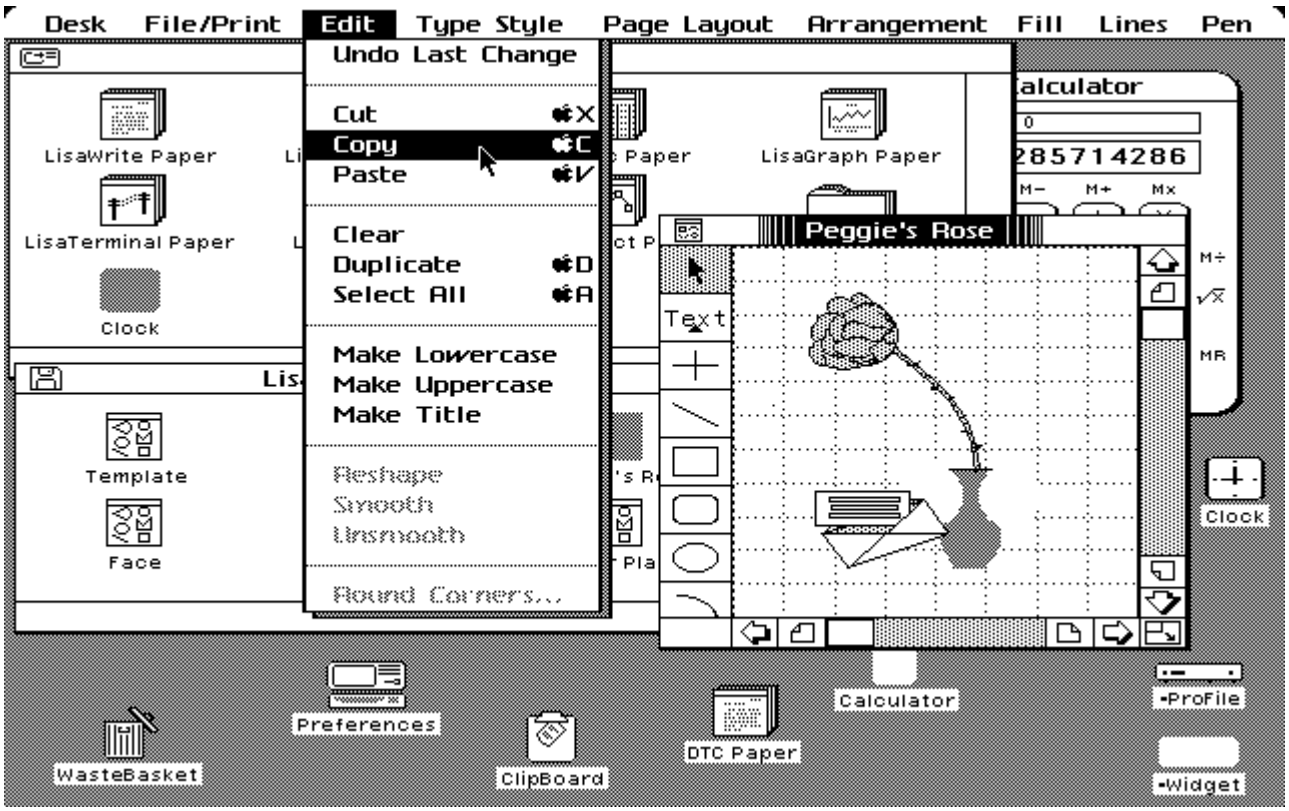
Il **1984 vide la nascita del personal computer Macintosh della Apple** che adottava un **nuovo tipo di interfaccia grafica** appositamente progettata per interagire in modo facile ed intuitivo con l’utente.

Fig. Il personal computer Macintosh della Apple.

Il Macintosh usava un interfaccia per il dialogo con l'utente di tipo grafico chiamata GUI (Graphic User Interface) fatta di icone, finestre, menù, ecc. L'utente poteva attivare le operazioni desiderate mediante il mouse con cui selezionava le finestre, le icone e i menù.



Gli oggetti dell'ambiente operativo invece di essere riferiti mediante nomi, erano in gran parte rappresentati in forma grafica rendendo molto più semplice la comprensione delle operazioni da svolgere sul computer.


Desk
File/Print
Edit
Type Style
Page Layout
Arrangement
Fill
Lines
Pen
Undo Last Change
Cut
Copy
Paste
Clear
Duplicate
Select All
Make Lowercase
Make Uppercase
Make Title
Reshape
Smooth
Unsmooth
Round Corners...
LisaWrite Paper
LisaTerminal Paper
Clock
LisaGraph Paper
Template
Face
Peggie's Rose
Text
Calculator
Clock
Preferences
WasteBasket
ClipBoard
DTC Paper
Calculator
ProFile
Widget

Questa modalità **non era un'invenzione della Apple, ma era stata sperimentata nel corso degli anni '70 nei laboratori della Xerox**. Sfortunatamente la Xerox non comprese le straordinarie potenzialità del lavoro svolto dai propri ricercatori e di fatto consegnò a Steve Jobs della Apple il primato di introdurre sul grande mercato questa nuova forma di dialogo più amichevole tra utente e macchina.

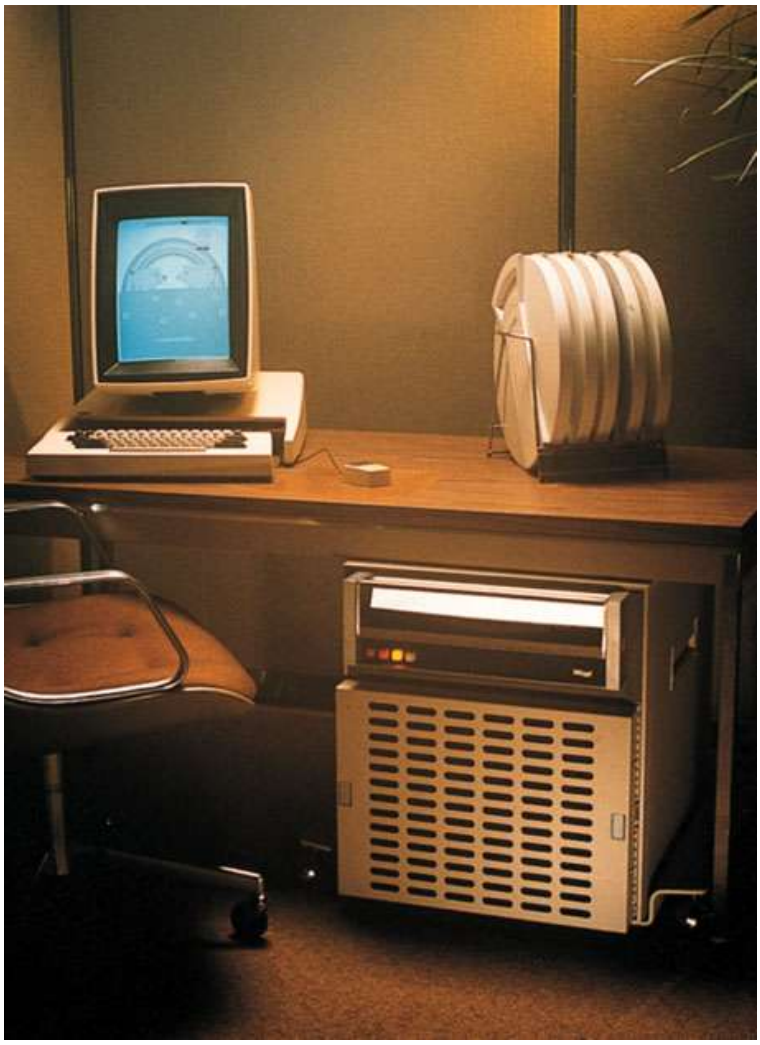
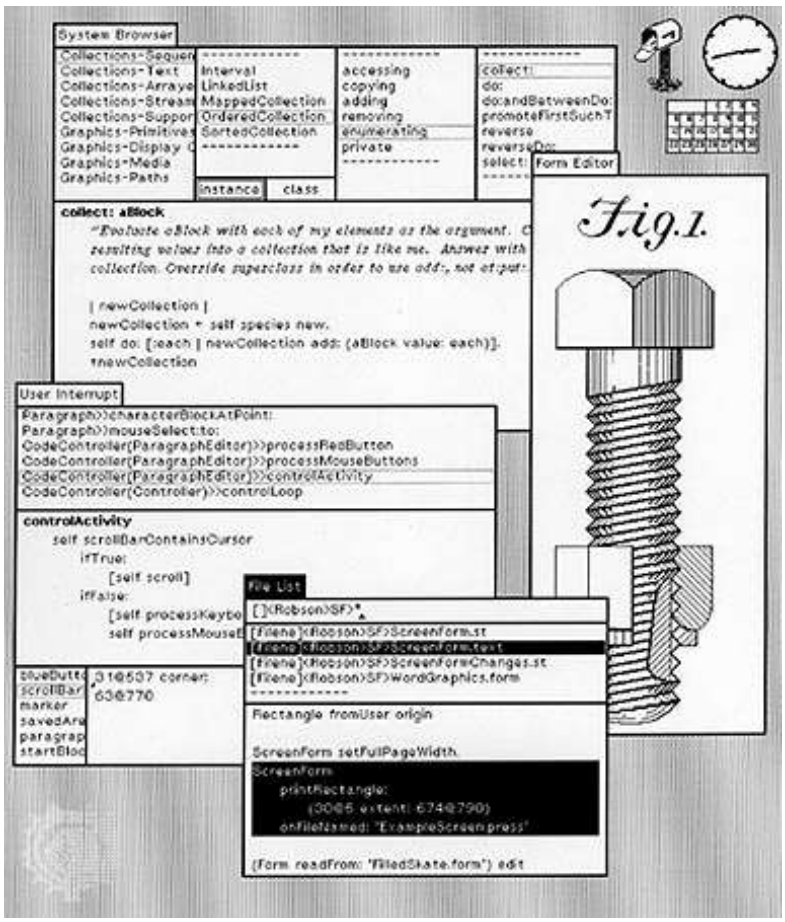


Fig. The Alto was the first system to pull together all of the elements of the modern Graphical User Interface.



The Xerox Alto system was the first computer to use a graphical user interface (GUI).

Nel 1979 Steve Jobs aveva potuto vedere l'uso delle nuove interfacce grafiche proprio presso i laboratori della Xerox e ne comprese a fondo la grande importanza per il mercato dei personal computer.

L'efficacia di questa modalità di interazione lo convinse che fosse la chiave giusta per fare apprendere l'uso del personal computer a chiunque senza dover comprendere i tecnicismi dell'informatica.

Sebbene i computer Macintosh non riuscirono a scalzare il predominio dei PC (stile IBM), si diffusero comunque notevolmente e si fecero apprezzare molto da numerosi utenti. Dopo qualche anno la Microsoft dovette colmare il gap, realizzando un sistema operativo basato su interfaccia grafica.

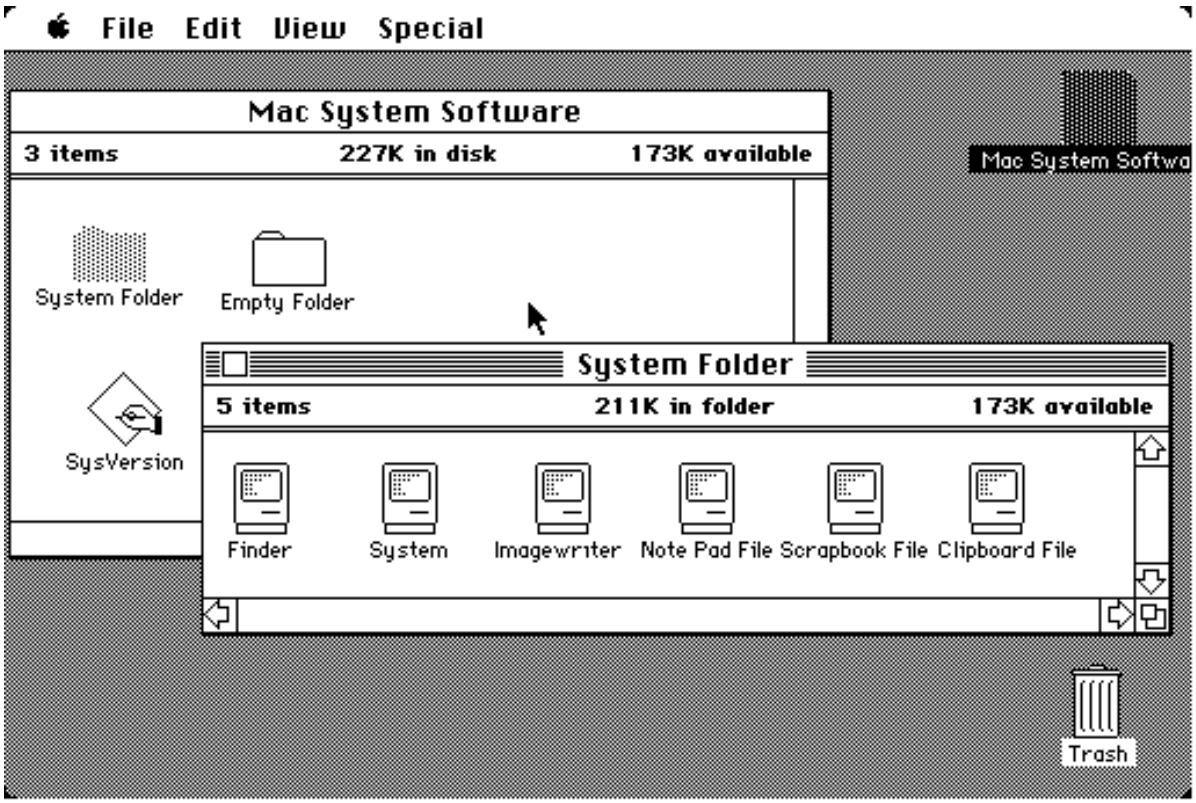


Fig. L'interfaccia grafica del Macintosh.

L'introduzione del Macintosh ebbe anche il merito di diffondere l'uso di un dispositivo di ingresso tanto semplice quanto utile come il ***mouse***.

Video: pubblicità del Macintosh

Video 2: demo Macintosh con Steve Jobs

Windows e i personal computer

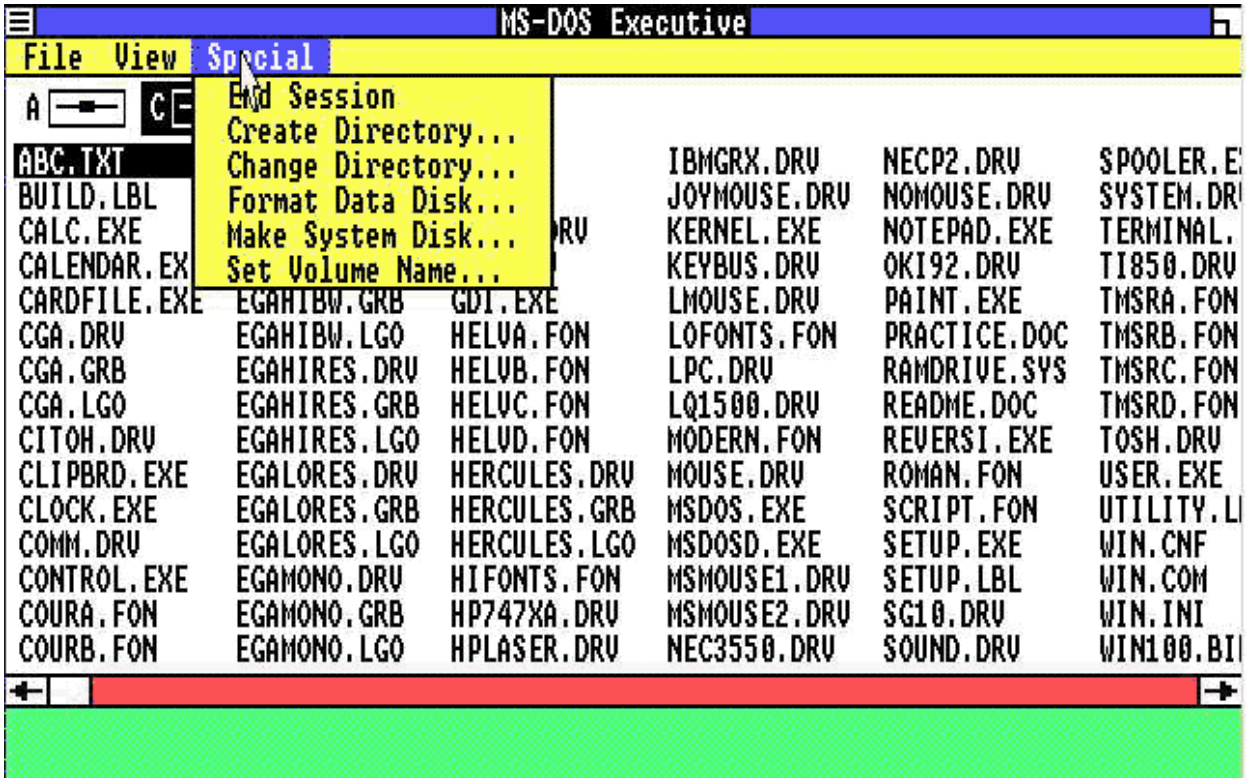


Fig. L'interfaccia grafica di Windows 1.0.

Il successo dei Macintosh con il notevole apprezzamento dell'interfaccia grafica **spinsero la Microsoft a realizzare un'interfaccia grafica GUI**. Il risultato di questo lavoro portò nel 1985 allo sviluppo del sistema operativo **Windows 1.0**, che però trovò pochi consensi nel pubblico, perché era troppo lento e instabile. Solo con la comparsa dei microprocessori Intel 80286 e 80386 il sistema operativo della Microsoft cominciò funzionare in modo adeguato.

Le versioni che però portarono al totale predominio della Microsoft nel mercato dei sistemi operativi furono prima **Windows 3.1 e poi Windows ’95**, che determinarono a poco a poco anche il declino dei sistemi Macintosh.

L’enorme successo determinato da questi sistemi operativi e dalle versioni successive fecero crescere in modo abnorme lo strapotere della Microsoft rendendo Bill Gates l’uomo più ricco del pianeta.

Nonostante la Microsoft sia incorsa più volte in problemi di monopolio del mercato, Windows continua a tutt’oggi ad essere il sistema più usato in assoluto.

VIDEO

Video 1: Bill Gates presenta Windows ’95

Linux

Fig. Linus Benedict Torvalds (video).

In questi ultimi quindici anni, i tentativi di altre case produttrici di software di frenare il dominio incontrastato della Microsoft, realizzando un sistema operativo concorrente sono di fatto falliti tutti.

La strada per ridurre lo strapotere della Microsoft sembra quella invece intrapresa da **Linus Benedict Torvalds**, un informatico nato a Helsinki nel 1969, con l'introduzione del sistema **Linux**.

Già durante gli studi universitari Torvalds si era interessato di sistemi operativi approfondendo il sistema ***Minix***, che rappresentava una **versione ridotta e semplificata di Unix**, sviluppata sotto la guida di Andrew S. Tanenbaum per scopi puramente didattici.

Il sistema Minix poteva

- funzionare su personal computer e
- veniva distribuito con i sorgenti disponibili.

L’idea di Linus fu quella di realizzare **una versione migliorata del sistema Minix** per poterla usare non solo a scopo didattico, ma **come alternativa a Windows**.

L’idea fondamentale di Torvalds fu quella di sviluppare un progetto **non seguendo la logica del profitto, ma di realizzare un sistema realizzato con il contributo di tutti** e utilizzabile gratuitamente da tutti.

A tale scopo egli realizzò un sistema rendendo disponibile i codici sorgenti, cioè adottando la cosiddetta filosofia dell’**open source**. L’obiettivo era di quello di spingere altri utilizzatori a migliorare la versione iniziale in modo che a poco a poco il sistema potesse crescere.

Nel 1991, Torvalds completò la prima versione del sistema, che denominò **Linux** e che **mise i sorgenti a disposizione di tutti**. Per sé riservò solo il compito di coordinare i diversi miglioramenti via via introdotti da altri sviluppatori.

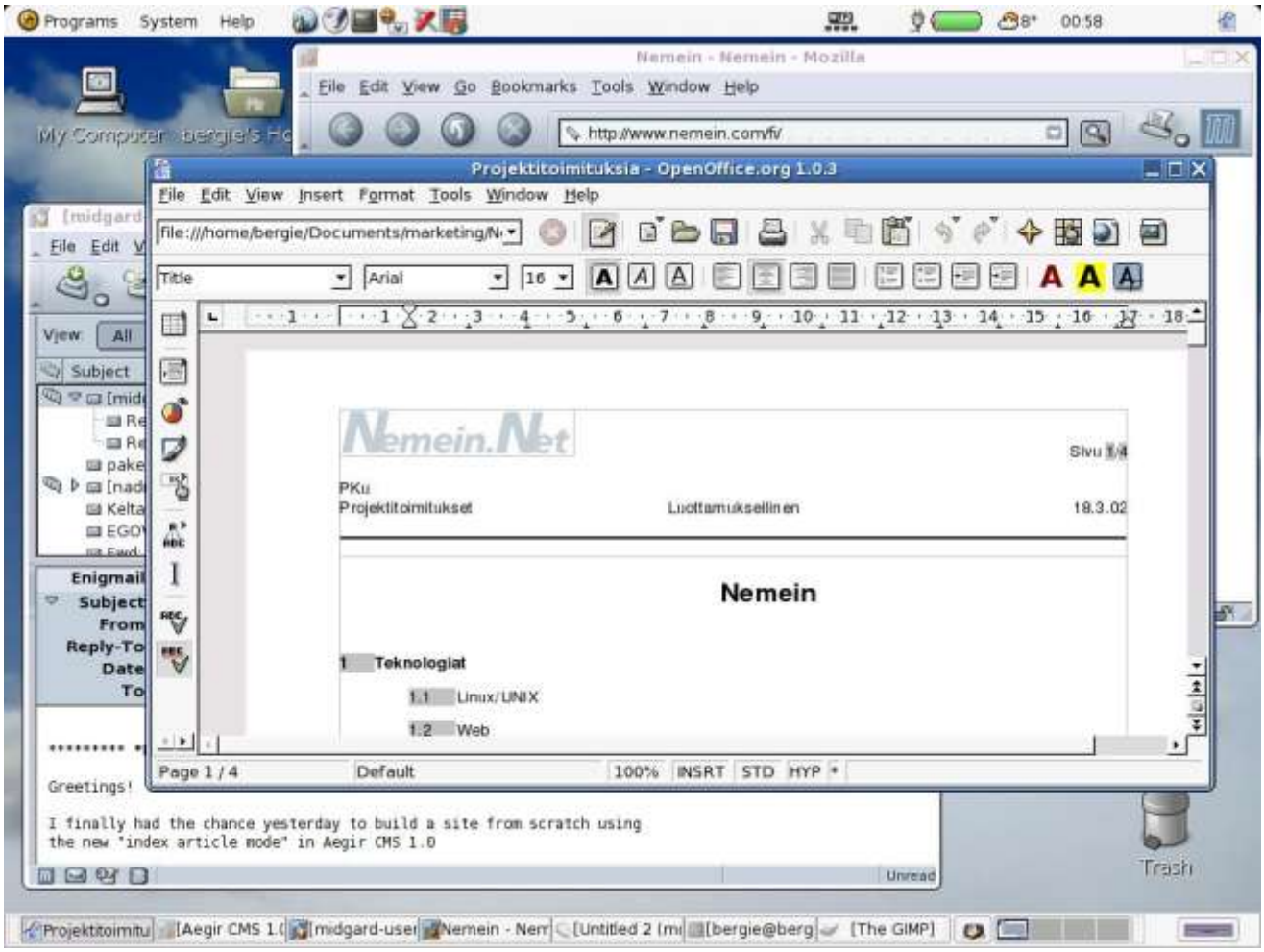


Fig. L’interfaccia grafica dell’ambiente Linux.

Fig. Richard Stallman.

In definitiva, il **sistema Linux è un software open source**, cioè è possibile vedere, studiare e usare i codici sorgenti dei singoli programmi senza che ci sia un copyright su di essi.

Si tratta di una filosofia completamente diversa da quella praticata dalla stragrande maggioranza delle aziende, che invece basano i propri guadagni sui diritti derivanti dall'uso del software.

Uno dei più importanti propugnatori di questa filosofia era **Richard Stallman**, che già a partire dal 1984 aveva iniziato a realizzare un progetto di software "libero", denominato **GNU**, in cui i sorgenti dei programmi erano disponibile per tutti gli sviluppatori.

Fig. Il logo classico di Linux.

Linux è cresciuto molto in questi anni con il contributo di tanti sviluppatori, e oggi appare l'unico sistema operativo nell'ambito dei PC in grado di fare concorrenza allo strapotere di Windows.

Linux si è dimostrato un valido sistema operativo, affidabile, sicuro e di buone prestazioni ed è in grado di gestire sia situazioni multiutente che multitasking.

Nel corso degli anni è aumentato di molto il numero non solo di utenti, ma anche di sviluppatori interessati a questo progetto.

Molte componenti sono state migliorate negli anni e inoltre l'ambiente Linux si è arricchito di molti pacchetti software di tipo applicativo costituendo una valida alternativa a quelli funzionanti in ambiente Windows.

VIDEO

Video 1: Breve storia del software libero e gnu linux