# Storia dell'informatica

Un riassunto di
Paolo Alfano


## Note Legali



### Liberatoria, aggiornamenti, segnalazione errori:

Quest'opera viene pubblicata in formato elettronico senza alcuna garanzia di correttezza del suo contenuto. Il documento, nella sua interezza, è opera di Paolo Didier Alfano
e viene mantenuto e aggiornato dallo stesso, a cui possono essere inviate eventuali segnalazioni all'indirizzo *paul15193@hotmail.it*
Ultimo aggiornamento: 15 dicembre 2016

# Indice

# 1 Storia dei sistemi di calcolo

## 1.1 Storia antica

> Possono essere state necessarie [all'uomo] ere intere per realizzare che una coppia di contadini e un paio di giorni erano entrambi istanze del numero 2
> Bertrand Russel

Un sistema di calcolo, manipola dei simboli. I primi simboli per cui e' nata un'esigenza di manipolazione sono i numeri.
I numeri sono stati una grande invenzione da cui sono stati messi a punto dei sistemi per rappresentarli, prima ancora che esistesse la scrittura[1].
Intorno al 3000 A.C. gli egizi sviluppano uno dei primi sistemi di numerazione addizionali non posizionali(la posizione del simbolo non conta), mentre i romani sviluppano un sistema additivo e sottrattivo posizionale.
Intorno al 500 D.C. in India viene messo a punto il sistema decimale usato ancora oggi, basato su dieci simboli e uno zero esplicito, rendendo piu' semplice il sistema di numerazione. Si diffonde velocemente nel mondo arabo.
Intorno al 820 D.C. Mohammed ibn-Musa al-Khuwarizmi scrive alcuni trattati di algebra e aritmetica che diffonderanno in Europa il sistema arabo-indiano. Tale diffusione si deve anche ai lavori di Leonardo Pisano, più noto come il Fibonacci, in particolare grazie al *Liber Abaci*.

## 1.2 Supporti meccanici al calcolo

Il primo strumento meccanico di supporto al calcolo e' l'abaco, anche se e' piu' uno strumento per memorizzare risultati intermedi.
All'inizio del '600 John Napier inventa i logaritmi, comodi perche' trasformano un prodotto in una somma

$$log(ab) = log(a) + log(b)$$

quindi per moltiplicare due numeri molto grandi $a$ e $b$ e' possibile prenderne i logaritmi, sommarli ottenendo $c$ e cercare nella tabella il numero il cui logaritmo e' $c$. Quel numero e' il prodotto di $a$ e $b$.
Edmund Gunter sfrutta questa proprieta' per creare il regolo calcolatore nel 1620 formato da due righelli scorrevoli che permettono di eseguire meccanicamente le moltiplicazioni. Rimane uno strumento diffuso fino agli anni '70 del ventesimo secolo.
Nel 1623 Wilhelm Schikard progetta l'orologio calcolatore, primo calcolatore meccanico capace di eseguire somme e sottrazioni.
Nel 1645 Blaise Pascal crea la Pascalina capace di eseguire solo addizioni e sottrazioni tramite complemento.
Nel 1672 Wilhelm Leibniz progetta una macchina capace di eseguire moltiplicazioni e divisioni, seppur coi limiti tecnologici del tempo. Credeva che l'uomo non

[1] Primi ritrovamenti risalenti a 10.000 anni fa

dovesse perdere tempo a svolgere i calcoli. Inoltre Leibniz concepi' il sistema binario e fu un precursore della moderna logica matematica.
Nel 1801 Joseph Jacquard usa schede perforate metalliche per il controllo di un telaio. La sua invenzione rivoluziona la produzione tessile. L'idea delle schede perforate resta in uso fino al 1980.
Nel 1822 il francese Thomas de Colmar costruisce il primo prototipo dell'aritmometro, una macchina calcolatrice portatile in grado di eseguire le 4 operazioni con risultati fino a 12 cifre.
Tra il 1820 e il 1830 Charles Babbage progetta il difference engine che non vede mai completamente la luce. Non riuscendo a trovare finanziamenti per il suo nuovo progetto, l'analytical engine basato su schede perforate, Babbage si sposta in Italia e presenta il suo progetto ad un congresso nel 1840. Riguardo al lavoro di Babbage scrivera' un articolo Luigi Federico Menabrea, che verra' tradotto da Ada Byron contessa di Lovelace e presentato nel 1843. E' considerato il primo articolo di informatica.
La macchina di Babbage era progettata per eseguire operazioni su variabili usando un mill che dirige le componenti meccaniche. Nella macchina di Babbage emergono gia' alcuni concetti dell'informatica moderna:

- L'unita' di controllo, ovvero il Mill
- L'utilizzo di variabili per memorizzare risultati finali e intermedi
- Il concetto di memoria
- Il concetto di computazione parallela visto che diverse operazioni potevano essere eseguite parallelamente
- Il concetto di input e output basato sulle variabili

Negli stessi anni, George Boole pubblica un lavoro in cui presenta i principi dell'algebra booleana. Considerato inizialmente un lavoro teorico, negli anni '30 del ventesimo secolo molti ricercatori mostrano come sia possibile usarla per costruire circuiti.

## 1.3 Supporti elettro-meccanici e schede perforate

Nel 1890, negli USA, viene bandita una gara per un sistema di conteggio per elaborare velocemente i dati del censimento. Vince Herman Hollerith con il suo sistema elettrico di tabulazione basato sull'uso di schede perforate. La macchina progettata da Hollerith raccoglie le sue informazioni in base allo schema dei buchi sulla scheda[2].
La macchina possedeva un'insieme di aghi retrattili che, in presenza di un buco sulla scheda, facevano chiudere un circuito che attivava un contatore dedicato ad uno specifico campo dell'utente.
La macchina era capace di analizzare ottocento schede al minuto e in seguito

[2]Ad esempio, in una certa posizione stabilita a priori, la presenza di un foro indicava "sposato", la sua assenza indicava "nubile/celibe"

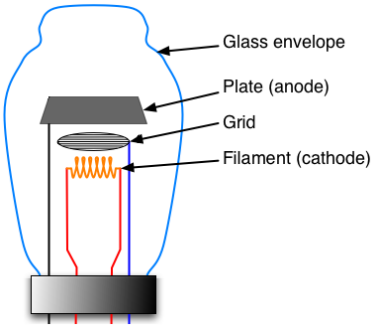


Figura 1: Struttura di un triodo

venne venduta ad altri enti come ferrovie, assicurazioni..
Nel 1924 nasce la International Business Machine Corporation, meglio nota come IBM e destinata a dominare il mercato dei computer sino agli anni '70.
Negli anni a seguire le schede perforate ricoprono un ruolo importante perche' comode ed economiche per codificare informazioni. In seguito su di esse verranno scritti anche i programmi da eseguire. Ogni linea conteneva una linea del programma da eseguire, e veniva letta da una macchina stampante che riportava il programma su di un foglio stampato per controllare che non vi fossero errori. Il programma veniva poi consegnato ad un operatore del centro di calcolo che restituiva i risultati ottenuti, anche qualche giorno dopo.

Nel frattempo vengono creati alcuni calcolatori analogici capaci di rappresentare numeri reali poiche' lavorano su grandezze fisiche continue. Alcune grandezze sono: elettriche(tensione, corrente), idrauliche(livello di un liquido) e meccaniche(posizione di un ingranaggio). Nel 1931 venne creato il primo calcolatore analogico capace di risolvere equazioni differenziali del terzo ordine. Purtroppo tali calcolatori erano difficili da programmare e poco flessibili nel rappresentare grandezze non reali.

## 1.4 Nascita e componenti dei computer digitali

Gli anni tra il 1930 e il 1945 si rivelano fondamentali nello sviluppo dei moderni calcolatori, complici le tensioni internazionali che spingono gli investimenti nel settore.
Ogni circuito digitale[3] si basa sull'uso di interruttori a comando elettrico. I primi di questo tipo sono i rele' in cui un'elettrocalamita fa scattare l'interruttore. Piu' importante e' l'invenzione del triodo nel 1906, la cui struttura viene mostrata in Figura 1.

Assomiglia ad una lampadina e al suo interno viene fatto il vuoto. Il filamento-catodo viene riscaldato producendo un'emissione di elettroni che at-

[3] Il computer e' sostanzialmente un grande circuito digitale

traversano la griglia e raggiungono l'anodo in proporzione alla tensione applicata alla griglia, creando un segnale variabile e amplificato. Le macchine basate su triodi hanno dominato la scena per molti anni ma ad oggi sono usati solo negli impianti audio ad alta fedelta'. I vantaggi rispetto al rele' sono evidenti visto che non ci sono parti meccaniche. Questo permette di avere maggiore velocita' e minore usura. Inoltre il consumo di corrente e' minore.

Nonostante questo i primi computer vennero realizzati usando rele'. Nel 1937 l'americano George Sitbit progetta nella sua cucina il model K(da kitchen), a cui poi segue il Complex Number Calculator. Formato da 450 rele', e' in grado di operare sui numeri complessi. Nel frattempo l'americano Claude Shannon (il padre della teoria dell'informazione) studia la corrispondenza tra la logica booleana e i circuiti digitali a relè.
Intanto l'ingegnere civile tedesco Konrad Zuse tra il 1936 e il 1938 progetta una macchina simile a quella di Sitbit, chiamata V1, parzialmente programmabile e capace di eseguire calcoli. Dopo la seconda guerra mondiale il nome viene cambiato in Z1.
La Z1 era completamente meccanica e capace di proseguire nella computazione di un passo al secondo, avendo quindi una frequenza di clock di 1Hz. Era composta da 20.000 parti e pesava circa una tonnellata. Era capace di eseguire una moltiplicazione in cinque secondi. La Z1 e' importante perche e' la prima macchina ad avere una struttura simile a quella dei moderni computer. Questo perche' possedeva

- una memoria
- un'unita aritmetica
- un lettore di nastro che leggeva le istruzioni
- un'unita di controllo che gestiva l'esecuzione
- un memory selector per indirizzare la memoria
- un dispositivo di input che convertiva i decimali in binari e uno di output che stampava i valori in binario convertiti in decimale

inoltre possedeva un set di istruzioni per leggere e scrivere la memoria, e per eseguire le quattro operazioni.
Intorno al 1939 Zuse, insieme al suo amico Helmut Schreyer, progetta un calcolatore a valvole che pero' non viene finanziato dalle autorita' statali.

Nel 1937 il fisico americano Howard Aiken di Harvard progetta una macchina simile all'analytical engine di Babbage usando soltanto i rele'. Riceve un finanziamento dall'IBM e lo completa nel 1943, chiamandolo Mark I. Di fatto era costituito da 78 calcolatrici sincronizzate fra loro e sebbene non fosse particolarmente veloce[4], viene ricordato perche' introduce il concetto di architettura

[4]Eseguiva una moltiplicazione in sei secondi e un logaritmo in oltre un minuto

Harvard nella quale memoria per i dati e memoria per le istruzioni sono nettamente divise. Questa architettura viene usata ancora oggi nei microcontrollori. Ad ogni modo al Mark I seguirono altri modelli, come il Mark II capace di eseguire moltiplicazioni in 0.75 secondi, ed e' in questo momento che viene coniato il termine "bug" quando Grace Hopper scopre una tarma insinuatasi nel computer causa il malfunzionamento di un rele'.

Ad ogni modo il primo computer completamente elettronico fu l'ABC, acronimo di Atanasoff-Berry Computer, realizzato tra il 1937 e il 1941. Era alimentato da una tensione alternata e non possedeva componenti meccaniche o elettromeccaniche. Era capace di eseguire trenta somme al secondo e la sua memoria era costituita da condensatori, come nelle moderne DRAM.

Lo scoppio della seconda guerra mondiale ostacola il lavoro di ricerca nel settore in molti paesi, ma non in Gran Bretagna.
A partire dal 1925 i tedeschi usano la macchina enigma per criptare i messaggi, ritenuta sicura perche' ogni messaggio poteva essere codificato in 100.000 messaggi diversi. L'intelligence polacca nel 1932 mette a punto una macchina elettromeccanica chiamata Bomba in grado di decifrarli. Dunque Enigma viene modificata portando ad un massimo di $18 \cdot 10^{19}$ codifiche possibili. Nel 1939 due matematici inglesi, Gordon Welchman e Alan Turing sviluppano una versiona piu' sofisticata di Bomba in grado di decifrare i messaggi di Enigma. A partire dal 1941 i tedesci applicano una cifratura diversa, detta cifratura di Lorenz che impedisce ai britannici di comprendere i messaggi scambiati dai tedeschi.
Nel 1944 Tommy Flower, grazie ai lavori di Turing progetta una nuova macchina completamente elettrica detta Colossus Mark 1 che riesce a decodificare i messaggi tedeschi. Sebbene il Colossus non fosse Turing completo e' stato dimostrato nel 2008 che una macchina Turing universale puo' essere simulata usando dieci esemplari di colossus in parallelo.

Sempre in quegli anni, con l'ingresso in guerra dell'esercito americano, divenne importante aggiornare le tabelle balistiche dei colpi dei cannoni a lunga gittata per i territori europei e africani. Inizialmente venivano impiegate 176 persone che eseguivano i calcoli con calcolatrici meccaniche a manovella. Avevano a disposizione anche due computer analogici capaci di risolvere equazioni differenziali ma ogni tabella conteneva 3000 traiettorie ognuna delle quali richiedeva 750 calcoli per essere computata. Il direttore di tali lavori era Herman Goldstine. Un giorno gli viene segnalato che un altro professore dello stesso istituto, John Mauchly, lavorava ad un calcolatore elettronico. Viene cosi' stipulato un contratto tra l'esercito e l'universita' della Pensylvania per produrre l'Electronic Numerical Integrator and Computer (ENIAC). Presentato al pubblico nel febbraio del 1946, troppo tardi per la guerra, verra' usato per i calcoli per l'uso della bomba a idrogeno.
L'ENIAC pesava 27 tonnellate e occupava una stanza di 10x15 metri. Formato da oltre 17.000 tubi a vuoto, 1500 rele', 10.000 condensatori e $5 \cdot 10^6$ saldature, si guastava spesso data la natura delicata dei tubi a vuoto.

L'ENIAC non usava un programma memorizzato, aveva schede perforate come input e come output. Inoltre andava configurato manualmente per ogni problema, lavoro che richiedeva alcune settimane.
Usava venti macchine addizionatrici che usate in parallelo eseguivano fino a 5.000 addizioni al secondo, pur non rappresentando in binario i numeri![5]
Programmare l'ENIAC significava schematizzare il problema in un'insieme di equazioni da scomporre in operazioni, e stabilire quale macchina avrebbe eseguito ogni calcolo e in che ordine. Poiche' le macchine non potevano memorizzare troppi dati in input, si utilizzavano anche delle "function tables" che contenevano fino a 104 numeri a 10 cifre con segno.
Nel 1948 furono apportate alcune modifiche suggerite da John von Neumann: usare una delle venti macchine per il program counter, un'altra per contenere gli indirizzi da leggere dalla function table e un'altra macchina che leggeva il valore contenuto presso tali indirizzi. La computazione rallentava di circa 6 volte ma la configurazione richiedeva ore e non giorni.

## 1.5 L'EDVAC e John von Neumann

Durante gli anni dell'ENIAC Mauchly e John Eckert lavorano gia' ad un nuovo progetto: l'Electronic Discrete Variable Automatic Computer, o EDVAC.
Nel 1944 Eckert scrive la descrizione di un nuovo tipo di memoria che dovrebbe memorizzare istruzioni e dati.
Nei vari meeting tenuti nell'istituto spesso partecipa anche John von Neumann che presenta un report dal titolo *First Draft of a Report on the EDVAC*, distribuito il 25 giugno 1945.
Contiene la prima descrizione logica di un computer che fa uso del concetto di programma memorizzato. Da questo articolo deriva il termine "architettura di von Neumann".
Comunque il documento contiene anche le idee di altri ricercatori del tempo(Eckert, Mauchly e Goldstine in particolare). A riprova di questo, il report era stato in realta' scritto da Goldstine. Alla luce di cio' sarebbe piu' corretto riferirsi all'architettura di von Neumann con il nome di "Architettura Eckert-Goldstine-Mauchly-von-Neumann".
L'architettura di von Neumann, e' mostrata in Figura 2

Di chiunque sia stata l'intuizione, e' un ottima idea che permette di rendere piu' veloce l'esecuzione dei programmi e delle istruzioni di controllo, inoltre rende piu' facile cambiare il programma da eseguire a seguito di modifiche. Il problema del tempo stava nel costruire una memoria sufficientemente spaziosa ed efficiente per contenere i dati e i programmi.
L'EDVAC divenne funzionante dal 1951, ed e' il primo a contenere l'idea di programma memorizzato. Possedeva 128 "linee di memoria" a mercurio capaci di memorizzare 8 parole ognuna. Una unita' computazionale eseguiva le operazioni aritmetiche su coppie di numeri e salvava il risultato in memoria. Una

[5] Ogni numero da rappresentare possedeva dieci flip flop, se ad esempio si doveva rappresentare la cifra 4, veniva messo a 1 il flip flop numero 4, e tutti gli altri rimanevano a 0

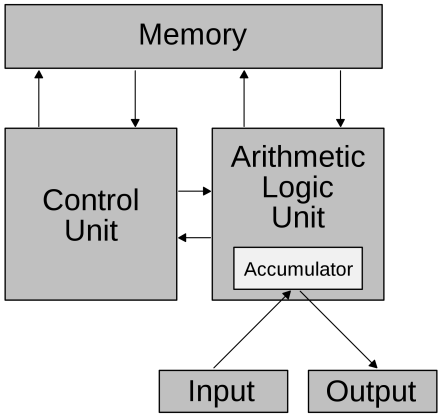


Figura 2: Architettura di von Neumann

differenza fondamentale tra ENIAC e EDVAC e' che il secondo rappresentava i numeri in binario.
Ogni istruzione dell'EDVAC era formata da 44 bit di cui: 4 per il codice dell'istruzione, 30 per l'indirizzo dove scrivere il risultato e 10 per l'indirizzo della successiva istruzione. Erano possibili 12 istruzioni macchina: 4 operazioni aritmetiche, salto condizionato, shift, lettura/scrittura su nastro, lettura da console di un dato, stop.
ENIAC ed EDVAC diedero il via a molti progetti di computazione del tempo e sin da subito divenne chiaro che l'efficienza dei computer non dipendeva solo dalla velocita' con cui eseguivano i calcoli ma anche dalla velocita' con cui erano in grado di reperire i dati e le istruzioni. Questa idea e' gia' presente nell'articolo *Preliminary Discussion of the Logical Design of an Electronic Computing Instrument* scritto da Burks, Goldstine e von Neumann, di cui riportiamo alcune righe

> Ideally, one would desire an indefinitely large memory capacity such that any particular...word would be immediately available...We are forced to recognize the possibility of constructing a hierarchy of memories, each which has greater capacity than the preceding but which is less quickly accessible

## 1.6 Eredita' dell'EDVAC

Vediamo ora due calcolatori creati partendo dalle idee dell'ENIAC.
Il primo e' il Manchester Small Scale Experiment Machine, noto come SSEM. Lo ricordiamo perche' e' il primo computer a programma memorizzato. Venne progettato da F. Williams e T. Kilburn per testare un nuovo tipo di memoria, il Williams-Kilburn Tube(WKT), alternativa alle linee di memoria a mercurio. L'innovazione del WKT stava nell'accesso non piu' sequenziale come nelle linee

a mercurio, bensi' diretto come nelle RAM.
Il WKT consisteva di un tubo catodico contenente un certo numero di slot che potevano essere scritti. Un ulteriore vantaggio era che il WKT occupava molto meno spazio rispetto alle linee di mercurio. Il SSEM non aveva un vero e proprio sistema di input visto che le istruzioni erano scritte in binario direttamente nella RAM, ma potevano essere solo programmi a non piu' di 32 word, quindi molto corti ma efficienti[6]. Per il SSEM vennero scritti solo tre programmi.
A partire dalla memoria innovativa del SSEM venne poi prodotto il Manchester Mark 1 su cui si baso' il Ferranti Mark 1, primo computer della storia ad essere messo sul mercato.
Alcuni invece considerano il Whirlwind 1 sviluppato al MIT come il primo computer a programma memorizzato visto che pur essendo stato costruito dopo, era stato progettato prima.
Il Whirlwind porto' due fondamentali innovazioni

1. Fu il primo computer ad operare in parallelo sui 16 bit che componevano le sue word e non in serie

2. Introdusse il concetto di microcodice o microprogramma che rappresentava il livello intermedio tra le porte logiche e le istruzioni macchina del computer. A coniare il termine fu Maurice Wilkes, nel 1951

## 1.7 Sviluppi tra fine anni '40 e inizio anni '50

Come abbiamo visto, tra la fine degli anni '40 e l'inizio degli anni '50, il mondo dei computer vide un florido sviluppo che getto' le basi per i grandi sviluppi degli anni '50.
Un'invenzione veramente importante per lo sviluppo dei computer si ebbe nel 1947 quando ai laboratori Bell, i fisici John Bardeen, Walter Brattain e William Shockley annunciano lo sviluppo dei transistor.
Un transistor e' un componente elettronico formato da tre strati di materiale semiconduttore[7] in cui ogni strato e' drogato[8] in modo alterno(solitamente NPN).
Come le valvole termoioniche, applicando una tensione variabile si ottiene una variazione del segnale in uscita, che puo' essere amplificato. Il transistor puo' anche essere usato come interruttore elettrico.
Quindi con il transistor possono essere costruiti circuiti molto piu' veloci, affidabili, compatti e con consumi minori.
Nel 1953 viene quindi costruito il primo computer a transistor e un anno dopo la Texas instrument commercializza il primo transistor al silicio.

[6] Circa 700 operazioni al secondo

[7] Un semiconduttore e' un materiale con conduttivita' intermedia tra un materiale conduttore e uno non conduttore

[8] Un materiale e' drogato positivamente se vengono aggiunte cariche positive, negativamente se le cariche sono negative

Nel 1952 viene commercializzato il primo mainframe dell'IBM, serie 700. Quattro anni dopo vengono costruiti a transistor indicati come serie 7000, domineranno il mercato dei computer per una decina d'anni.

Nel 1953 viene annunciato l'IBM 650 che viene consegnato nel dicembre del 1954. Nel 1959 ne viene commercializzata una versione a transistor.
Fu il primo computer ad essere commercializzato in grandi quantita'(duemila esemplari). Il suo successo e' dovuto al fatto che costava "poco"(200.000-400.000$), aveva dimensioni contenute(occupava una stanza) ed era piu' facile da usare perche' le istruzioni andavano date in decimale anziche' in binario.
Viene considerato l'antenato dei PC perche' permetteva al programmatore di usarlo direttamente e non tramite un operatore. Veniva assegnato uno slot di tempo ad ogni programmatore(magari anche in tarda notte!) che poteva usarlo liberamente.
La configurazione di base prevedeva unita' principale, console di controllo, modulo alimentazione e lettore/perforatore di schede.
Le istruzioni erano formate da dieci numeri decimali della forma "xx yyyy zzzz" dove

- xx indicavano il codice dell'operazione
- yyyy indicava l'indirizzo di memoria del dato da usare
- zzzz indicava l'indirizzo di memoria della prossima istruzione

Anche se era scomodo scrivere i programmi in decimale, sempre meglio che in binario!
Ad ogni modo nel 1957 circola il primo compilatore FORTRAN che non era ancora molto comodo da usare visto che per usarlo si doveva passare al calcolatore il codice del compilatore fortran seguito dal programma. A questo punto il calcolatore produceva un testo compilato che andava rieseguito sul calcolatore stesso.

Un'eminente figura che ha utilizzato l'IBM 650 e' Donald Knuth. Ha scritto un'opera monumentale chiamata *The art of computer programming* che consta di oltre duemila pagine in quattro distinti volumi dove tratta moltissimi fondamenti teorici e algoritmi. A proposito di tale opera, Bill Gates scrive

> If you think you're a really good programmer..read(Knuth's) Art of Computer Programming..You should definitely send me a résumé if you can read the whole thing

Inoltre, nel 1976 dovendo preparare la seconda edizione sente la necessita' di un sistema di composizione di testi per il computer. Nel 1984 aveva sviluppato Tex.

Intorno al 1955 cominciano ad essere impiegate le memorie a nucleo magnetico(magnetic core memory) destinate a restare la forma predominante di memoria primaria per circa venti anni. Sviluppati da una ricerca che mirava a

sostituire i Williams-Kilburn Tube perche' ritenuti inaffidabili, si basano sulla modifica del campo magnetico di certi materiali in modo che la direzione del campo indichi valore 0 o valore 1. Erano costruiti con dei fili elettrici intrecciati a nuclei di ferrite che generavano il campo magnetico col quale polarizzare ogni singolo nucleo. Poter conservare la direzione del campo magnetico rendeva le MCM memorie non volatili, sebbene venissero usate come memoria primaria[9].
Sempre in quegli anni viene sviluppato un altro supporto di memoria chiamato memoria a tamburo rotante o Magnetic Drum Memory(MDM). Questa tecnologia gia' esistente dagli anni '30 viene migliorata negli anni '50. Come le memorie a nucleo magnetico erano non volatili ma troppo piccole per essere usate come memoria secondaria.
Erano costituite da bande di materiale ferromagnetico che venivano lette e scritte da testine. Da questo punto di vista assomigliavano agli hard disk moderni ma con l'inizio degli anni '60 verranno rimpiazzati proprio da questi.

## 1.8 Hard disk, circuiti integrati e il computer moderno

Nel 1956 viene presentato l'IBM RAMAC contenente il primo hard disk magnetico molto simile a quelli moderni. Memorizzava 5 Mbit su 50 piatti di 61 cm di diametro. Possedeva molti bracci meccanici e due testine che leggevano il disco che ruotava a 1200 RPM.

Gli anni '50 si concludono con l'invenzione dei circuiti integrati presso la Texas Instruments nel 1958. Jack Kilby, dimostro' che i circuiti integrati potevano essere miniaturizzati. Il problema dei transistor al tempo era che occupavano poco spazio in se' ma la capsula di plastica che li conteneva li rendeva piu' ingombranti. Nei circuiti integrati si mettevano insieme piu' transistor che venivano poi inseriti in un unico contenitore da collegare ad altri componenti.
Col tempo si e' riuscito ad inserire sempre piu' transistor per $mm^2$ secondo la legge di Moore la quale teorizza che il numero di transistor in un circuito integrato raddoppi ogni 18/24 mesi.

Su queste basi poggiano gli sviluppi degli anni '60. Nel 1960 viene commercializzato il Programmed Data Processor-1(PDP-1), primo computer commerciale ad usare un monitor, primo ad ospitare un videogame un text editor un word processor e un debugger interattivo. Usava un nastro perforato come memoria di massa e 8Kbyte di MCM. Eseguiva circa 100.000 istruzioni al secondo.
Il PDP-1 era costruito interamente con transistor e diodi. Il fatto che possedesse una console presso cui scrivere rendeva il lavoro sul PDP-1 simile a quello dei giorni nostri.

Sempre negli anni '60 vengono inoltre sviluppate diverse idee e tecnologie

[9] Da qui il nome che avevano, *core memory* da cui discende il nome dell'errore di sistema core dump

- Nel 1961 viene implementata l'idea della paginazione virtuale e della segmentazione della memoria
- Nel 1962 compare l'idea di pipeline basata sul ciclo fetch, decode, execute
- Sempre nel 1962 viene implementata la protezione della memoria e la memoria interlacciata

Un punto di svolta viene rappresentato dall'immissione sul mercato dell'IBM/360 nel 1964 che viene realizzato in parte a circuiti integrati[10].
L'IBM/360 aveva una velocità di esecuzione variabile da 0,0018 a 0,034 MIPS(Millions of Instructions Per Second), possedeva una memoria primaria da 1 Mbyte a MCM e una memoria secondaria da 7.2 Mbyte su hard disk removibile. Inoltre porta diverse innovazioni progettuali: la memoria viene indirizzata a byte, utilizzava registri general purpose, usava una TLB, sfruttava una gerarchia di interrupt e introduceva il Direct Memory Access.

Nel frattempo in Italia veniva sviluppata la Olivetti Programma 101 che, presentata nel 1964 a New York, vendette circa 40.000 esemplari. Sebbene fosse presentata come calcolatrice, di fatto era programmabile, quindi assimilabile alla branca dei computer.

Nel 1965 viene commercializzato il CDC 6600, progettato anche da Seymour Cray, ed e' considerato il primo supercomputer della storia. Raggiunse la potenza di calcolo di 1 Mflops, ovvero eseguiva 1 milione di operazioni in virgola mobile al secondo.
Il problema dei computer dell'epoca era che i processori dovevano gestire troppe cose: calcoli, periferiche, accessi in memoria.. Questo dava luogo a processori complessi e di grandi dimensioni, in cui il tempo del ciclo di clock si dilatava.
L'intuizione di Cray fu quella di separare nettamente le operazioni di esecuzione di istruzioni da quelle necessarie per far funzionare il computer.
Creando una CPU che eseguisse solo operazioni aritemetiche e logiche, la CPU del CDC 6600 era semplice e piccola. Questo principio e' quello che stara' alla base delle future architetture RISC. Per gestire periferiche e memoria Cray penso' di affiancare un processore periferico che si occupava delle comunicazioni tra CPU e le periferiche.
Inoltre il CDC 6600 fu il primo computer superscalare in quanto era dotato di dieci unita' funzionali in grado eseguire le istruzioni in parallelo.

Nel 1964 viene creato da Douglas Engelbart il primo mouse, pensato come strumento per espandere l'espressivita' degli esseri umani. Negli anni che seguono Engelbart sviluppera' altre idee fondamentali per l'interazione uomo macchina come gli schermi bitmapped, le interfacce grafiche e gli ipertesti.
Sempre in quegli anni vengono presentati i primi minicomputer. Chiamati in questo modo perche' di dimensione piu' limitata(grandi quanto un armadio) e

[10] I primi computer ad usare circuiti integrati in modo massiccio emergono intorno al 1968

relativamente piu' economici. Declineranno intorno agli anni '80 con la diffusione dei primi microprocessori economici serie x86.

Un'altra invenzione fondamentale degli anni '60 e' la cache. A quei tempi si cominciava ad intendere che con i nuovi processori ad alte prestazioni l'accesso alla memoria avrebbe rappresentato un collo di bottiglia. Nel 1965 Maurice Wilkes pubblica un articolo in cui presenta la necessita' di possedere una memoria intermedia tra i registri del processore e la RAM, chiamata appunto cache.
Nel 1968 la IBM presenta un computer con cache che viene confrontato con un altro dal processore piu' potente ma privo di cache. Il confronto viene vinto dal computer con cache e mette in evidenza l'aumento di prestazioni ottenibile utilizzando una memoria intermedia.
Sempre per ridurre l'impatto dell'accesso in memoria, nel 1967 Robert Tomasulo inventa una schema di esecuzione delle istruzioni detto out of order. In questo schema le istruzioni non vengono necessariamente eseguite in base all'ordine in cui sono prelevate ma in base alla disponibilita' degli operandi. Sebbene questo schema renda piu' complicata l'architettura del processore, ne aumenta di molto le prestazioni. Infatti e' una soluzione applicata in tutti i moderni processori.

Un ultimo contributo degli anni '60 e' la prima demo. Nel 1968 Douglas Engelbart a San Francisco effettua una dimostrazione di funzionamento delle caratteristiche principali di un moderno computer. Engelbart illustra molti nuovi elementi: le finestre, gli ipertesti, la grafica, il word processor, e il mouse. Questa demo avra' una notevole influenza sullo sviluppo degli anni a venire.

## 1.9 Gli anni '70

Gli anni '70 si aprono con la commercializzazione del PDP-11 il primo computer per la produzione di massa che poteva anche essere assemblato da addetti non specializzati. Nel 1971 viene commercializzato il floppy disk che inizialmente aveva un diametro di 20cm. Il floppy disk si diffondera' massivamente negli anni '80 e '90.

Qualche anno prima, nel 1968, Robert Noyce e Gordon Moore fondano una societa' chiamata NM Electronics, che cambia nome in Integrated Electronic Corporation, abbreviato poi in Intel.
L'Intel nasce con l'intento di costruire RAM a semiconduttori. L'azienda mette insieme i vari sviluppi del tempo relativi a memorie a transistor e circuiti integrati per costruire una RAM che fosse inserita in un unico chip. Dopo aver prodotto alcuni innovativi chip, rilascia tra il 1970 e il 1971 l'Intel 1103 noto anche come Magnetic Core Memory Killer. Per la prima volta una quantita' significativa di RAM e' contenuta in un unico chip e nel 1972 e' il circuito integrato piu' venduto sul mercato.
In contemporanea, nel 1969 la Busicom(un'azienda giapponese) contatta la appena nata Intel per farle costruire una versione elettronica della propria calcolatrice. L'Intel decide di mettere a punto un unico grande chip(invece che 12

come progettato dall'azienda giapponese) che non soddisfa la Busicom. La Intel che recupera i diritti sul chip in cambio della restituzione del denaro speso dalla Busicom e decide, mettendo a capo del progetto l'italiano Federico Faggin che ha gia' lavorato per Olivetti, di produrre un nuovo chip, chiamato 4004.
Conteneva circa 2300 transistor ed eseguiva circa 60.000 istruzioni al secondo. Insieme al 4004 vennero sviluppati altri tre chip, chiamati 4001, 4002, 4003. L'assemblaggio di questi quattro chip produceva un computer funzionante.
Nel frattempo, in un progetto Intel indipendente dal 4004 andava sviluppandosi un nuovo chip chiamato 8008. L'8008 e' il capostipite della famiglia x86, anche se l'origine di tale famiglia risale a prima della Intel, nel 1968.
Nel 1968 viene fondata la Computer Terminal Corporation(CTC) che cambiera' nome in Datapoint. A quel tempo i computer si usavano collegando terminali telescriventi che stampavano su carta l'input per il computer. Il modello piu' diffuso era la Teletype model 33 della IBM. Alla CTC viene in mente di produrre un terminale piu' veloce e silenzioso il cui output venisse mostrato su un monitor e non stampato su carta. Viene sviluppato il Datapoint 3300 che ha un immediato successo tanto che in quel momento la Datapoint fa costruire ad aziende terze(come Texas Instruments e Intel appunto) le componenti. La Datapoint decide di andare oltre progettando un terminale che emulasse il comportamento di ogni telescrivente e assume Victor Poor e Harry Pyle che sviluppano lo schema dell'architettura che diventera' la base per la famiglia di microprocessori piu' famosa al mondo, la x86.
La Datapoint contatta la Intel per capire se l'intera implementazione possa essere realizzata su un unico circuito integrato. In quel momento la Intel non e' particolarmente interessata perche', come sappiamo, e' impegnata a produrre memorie e il 4004. Ma visto che la Datapoint si riforniva di memorie presso la Intel, la Datapoint minaccia di non servirsi piu' presso la Intel e quest'ultima avvia la produzione del chip. La produzione procede con calma e nel frattempo vengono commercializzate dalla Datapoint due versioni(successive) a normali circuiti integrati dell'emulatore di telescriventi. Questi due modelli prendono il nome di Datapoint 2200 e Datapoint 2200 II ed entrambi riscuotono grandissimo successo.
Nel 1971 la Intel termina la produzione della CPU su circuito unico ma per la Datapoint e' troppo tardi, visto che aveva gia' messo in vendita i modelli 2200 e 2200 II.
Allora la Datapoint cede i diritti sul nuovo processore alla Intel e in cambio non paga le spese sostenute dalla Intel per svilupparlo nei due anni precedenti.
Dal chip ceduto dalla Datapoint nascera' poi l'8008.

Dagli sviluppi dei primi anni '70 nascono dunque le basi del computer moderno a microprocessori. Infatti la capacita' di comprimere su un circuito sempre piu' piccolo le varie componenti elettroniche permette di sviluppare processori piu' veloci e sofisticati(cioe' in grado di usare parallelismo, scheduling dinamico della pipeline, predizione dinamica dei salti..). Per avere un'idea del livello di miniaturizzazione si consideri che negli anni '60 un chip conteneva centinaia di transistor e nel 2005 viene presentato il primo chip con piu' di un miliardo di

transistor. La capacita' di costruire componenti sempre piu' piccole viene indicata con "tecnologia a $n$ nanometri" dove $n$ indica l'ampiezza di una linea di connessione tra due microcomponenti elettronici. Si prevede che per effetti fisici non si possa scendere al di sotto dei 10 nanometri con le tecnologie attuali(nel 2015 e' stata presentata una architettura a 14 nanometri).

Un altro computer da ricordare e' stato sviluppato presso Palo Alto dalla Xerox, chiamato per l'appunto Alto. Possedeva un'interfaccia grafica ed era collegato ad una rete locale ethernet. Viene anche ricordato perche' influenzo' grandemente i progetti Macintosh e le workstation Sun.

Qualche anno dopo, nel 1976, Steve Jobs e Steve Wozniak dopo aver lasciato i rispettivi college si incontrano presso un club di appassionati di computer. Wozniak, genio autodidatta dell'elettronica sviluppa un suo piccolo computer, che verra' poi chiamato Apple I. Jobs riesce a venderne 50 esemplari presso un negozio locale di computer, costruiti utilizzando componenti vendute a credito. Jobs, Wozniak e Mike Markkula(che finanzio' l'azienda con 250.000$) fondano la Apple Computers Inc.
Intanto Wozniak progetta l'Apple II presentato nel 1977. Nel 1980 seguira' l'Apple III, che pero' non contenendo un sistema di ventilazione interno costringera' la Apple a ritirarne 14.000 esemplari, subendo un notevole danno di immagine. Nel frattempo sul mercato dei personal computer fa il suo ingresso la IBM con il suo IBM 5150. Da subito molto popolare, negli anni a seguire l'etichetta "IBM compatibile" nel mondo delle periferiche sara' irrinunciabile per molti.
La Apple non si rende bene conto dell'ingresso di IBM e tratta l'azienda rivale con sufficienza. Ma nel giro di tre anni la IBM detiene il 56% del mercato dei personal computer, contro il 16% della Apple. Nonostante questo la Apple riuscira' a riprendersi con la commercializzazione del Macintosh 128k, che riscuotera' un enorme successo.

## 1.10 Dagli anni '80 ad oggi

Per osservare un vero cambiamento dovremo aspettare gli anni '80 in cui si sviluppano le nuove architetture RISC(Restricted Instruction Set Computer).
Durante gli anni '60 e '70 la definizione dell'instruction set dei processori aveva seguito linee guida precise infatti quasi tutti i processori possedevano set di istruzioni:

- Con un gran numero di istruzioni
- Ogni istruzione era complessa
- Molte istruzioni indirizzavano e gestivano la memoria

A quei tempi un'insieme complesso di operazioni poteva essere rappresentato con una singola istruzione. Questo rappresentava un vantaggio poiche' gli eseguibili erano corti e occupavano poco spazio in memoria che a quei tempi era

ben poca. I tempi di accesso in memoria erano elevati quindi aveva senso che ogni accesso ad essa prelevasse un'istruzione che eseguisse una gran quantita' di lavoro. Inoltre istruzioni complesse riducevano il gap semantico tra un linguaggio scritto ad alto livello e la traduzione in linguaggio macchina, trasferendo parte della complessita' all'interno del processore.
Questa versatilita' aveva pero' un costo: istruzioni macchina cosi' diverse fra loro avevano lunghezza variabile(in tal modo non si sprecava spazio). Visto che venivano prelevate in memoria quantita' fisse' di bit non si sapeva da subito se il blocco prelevato contenesse una nuova istruzione, una parte di una precedente o di una successiva. Inoltre poiche' le istruzioni potevano indirizzare liberamente la RAM, questa diviene velocemente un collo di bottiglia.
Viene quindi sviluppato a Berkeley da D. Patterson e C. Sequin un processore che evolvera' nelle famiglie di processori che conieranno l'acronimo RISC contrapposto a quello CISC(Complex Instruction Set Computer) con cui si indicavano le architetture precedenti.
L'architettura RISC impiega un ristretto numero di semplici istruzioni macchina. Questo permette di avere istruzioni della stessa lunghezza e una struttura regolare. Inoltre solo due istruzioni (LOAD e STORE) possono indirizzare la RAM, eliminando il collo di bottiglia verso di essa. Questa struttura rendeva le architetture RISC particolarmente adatte per le pipeline di istruzioni.
Quindi rispetto alle macchine CISC, un'architettura RISC eseguiva piu' istruzioni ma molto piu' semplici, occupando una maggiore quantita' di memoria. Anche se il numero di istruzioni era 4 o 5 volte maggiore, chi sosteneva le architetture RISC asseriva che il tempo di esecuzione di una singola istruzione fosse di circa 1/10, portando quindi un guadagno in efficienza.
Il primo vero test in tal senso viene fatto nel 1989. La macchina RISC utilizzava in media il doppio delle istruzioni della macchina CISC ma le eseguiva in 1/6 del tempo, portando ad una performance tre volte superiore in media.
Dagli anni '90 le grandi compagnie cominceranno a produrre computer RISC. Anche se l'Intel utilizza ancora le istruzioni x86 tipicamente CISC, vengono tradotte dopo la fase di fetch in piu' microistruzioni semplici di lunghezza fissa a 72 bit.

Negli anni '90 i processori subiscono ulteriori migliorie basate sullo scheduling dinamico della pipeline, predizione dei salti condizionati basandosi su esecuzioni precedenti..

L'ultima miglioria notabile viene portata alla fine degli anni '90 quando diviene chiaro che il parallelismo tra istruzioni e' stato portato al massimo. Questo fatto spinge i produttori ad utilizzare nuovi processori multi core in grado di eseguire piu' thread. Vengono anche sviluppati chip dedicati all'elaborazione grafica, le GPU. Ad oggi i piu' grandi supercomputer sono costituiti da milioni di core e GPU che raggiungono un potere computazionale di migliaia di TeraFLOPS.

# 2 Storia dei linguaggi di programmazione

## 2.1 Albori della programmazione

Nonostante negli anni siano comparsi e si siano estinti centinaia di linguaggi di programmazione, solo una decina di questi hanno dato un contributo importante all'evoluzione del concetto di linguaggi di programmazione.
Esagerando un po' la specifica per calcolare numeri di Bernoulli elaborata da Ada Lovelace sull'Analytical Engine di Babbage viene considerato il primo programma per computer della storia.
Successivamente i computer analogici degli anni '30 calcolavano varie funzioni e il loro "linguaggio" parlava tramite la configurazione dei circuiti elettrici.
Nel 1943 i programmi vengono codificati su nastro perforato mentre un paio di anni dopo vengono sviluppati dei diagrammi di flusso che illustravano graficamente l'esecuzione del programma. Nel 1948 nel Manchester SSEM(che ricordiamo essere il primo computer a programma memorizzato) i dati scritti in binario venivano inseriti uno per uno a mano in memoria.
Possiamo riassumere che alla fine degli anni '40 erano veramente poche le macchine calcolatrici in funzione e dunque erano pochi anche i programmatori.

Nel 1949 entra in funzione l'EDSAC, un computer a programma memorizzato che possiede un instruction set da 18 istruzioni macchina, dette Initial Orders. Per la prima volta viene assegnata ad ogni istruzione una diversa lettera dell'alfabeto che rendeva piu' facile l'utilizzo agli esseri umani.
Il programma veniva dunque scritto su carta e codificato successivamente su nastro perforato usando un perforatore manuale con una corrispondenza tra le varie lettere che rappresentavano le istruzioni e la sequenza di buchi da applicare al nastro. Di fatto gli Initial Orders erano una forma primitiva di Assembler.
Piu' o meno in contemporanea John Mauchly concepisce il primo linguaggio di programmazione ad alto livello: lo Short(Order)Code che verra' poi utilizzato sull'UNIVAC I[11]. Lo Short Code rappresentava ogni elemento del codice(operatori matematici, parentesi..)con un numero, opportunamente interpretato dal computer.

Nel frattempo in quegli stessi anni Haskell Curry mette a punto un sistema di regole logiche che sebbene rimaste su carta rappresentano il primo esempio di descrizione della fase di generazione del codice di un compilatore.

Un evento importante che si verifica in quegli anni e' la nascita del costrutto FOR che viene ideato da Heinz Rutishauer tra il 1949 e il 1951.

Sempre in questo periodo cominciano a svilupparsi le prime forme di compilatore. Nel 1950 l'italiano Corrado Böhm definisce un linguaggio ad alto livello, un linguaggio macchina e un metodo di traduzione dal primo al secondo. Nel suo lavoro sono contenuti concetti equivalenti a goto e if-then-else, l'uso dello

[11] Uno dei calcolatori figli dell'esperienza dell'EDVAC

statement di assegnamento e delle subroutine. Rispetto al compilatore pensato da Rutishauer, quello di Bömh e' piu' chiaro, semplice e ha una complessita' di traduzione pari a $n$ e non $n^2$.
Per il primo vero compilatore dobbiamo aspettare il 1952 quando Alick Glennie in Inghilterra produce AUTOCODE. Sebbene l'output dell'AUTOCODE fosse comunque complicato, Glennie si rese conto che potevamo usare lo stesso computer per tradurre il codice ad alto livello in linguaggio macchina per poi far girare sulla stessa macchina il codice compilato. Ad ogni modo il lavoro di Glennie non ebbe un impatto cosi' evidente poiche' troppo in anticipo sui tempi. In quel periodo i programmatori avevano altri problemi di cui occuparsi(mancanza di memoria, malfunzionamenti hardware, fallimenti di esecuzioni..)

## 2.2 1954-1956: FORTRAN e altri

Il periodo dal 1954 al 1956 risulta particolarmente fertile di nuove idee ed implementazioni per la storia dei linguaggi di programmazione. Le elenchiamo di seguito:

- Nel 1954 viene presentato il sistema George per il whirlwind. Usava un linguaggio svincolato dalla macchina su cui girava, e non richiedeva conoscenze sul computer usato. Anche se il codice generato era 10 volte piu' lento si dimostro' efficiente nella stesura veloce di programmi complessi.
- Nel 1954 viene sviluppato un nuovo AUTOCODE capace di operare utilizzando i numeri in virgola mobile.
- In Russia viene avviato un programma di sviluppo di sistemi composti da linguaggio e compilatore.
- Viene sviluppato un sistema dalla Boeing Airplane Company capace di tradurre espressioni algebriche in linguaggio macchina
- Nel 1955 Kenton Elsworth sviluppa un sistema di traduzione di equazioni in linguaggio macchina chiamato Kompiler.
- Nel 1955/56 viene presentato il primo linguaggio di programmazione dichiarativo
- Nel 1954 viene sviluppato il primo assembler moderno che viene seguito nel 1955 dal SOAP(Symbolic Optimal Asssembly Program)
- Nel 1956 al Carnegie Institute of Technology viene sviluppato un linguaggio ed un compilatore chiamati IT(Internal Translator). Il compilatore generava prima un codice intermedio da cui generava il codice macchina. Il codice intermedio era scritto in SOAP. IT fu il primo sistema realmente utile. Per primo possedeva un linguaggio potente ed espressivo, una implementazione efficiente, una adeguata documentazione.

Sempre in questi anni viene sviluppato anche il FORTRAN. Nel 1954 alla IBM un progetto intende sviluppare un linguaggio capace di essere semplice e la cui traduzione sia efficiente, chiamato IBM Mathematical FORmula TRANslating System, universalmente noto come FORTRAN. Il FORTRAN voleva essere efficiente, non del tutto svincolato dalla macchina su cui girava ma che avesse una notazione concisa. FORTRAN verra' rilasciato nel 1957 e nel giro di un anno molti prodotti IBM 704 lo utilizzavano. Nel frattempo vennero introdotti nuovi costrutti e la possibilita' di aggiungere i commenti. L'enorme vantaggio del FORTRAN e' che davvero riusci' a creare un compilatore che generasse codice efficiente e per molto tempo e' stato utilizzato abbondantemente nelle applicazioni scientifiche.

Nel 1958 fa il suo debutto un altro linguaggio usato ancora oggi: LISP. Sviluppato nei due anni precedenti da John McCarthy era il primo esempio di linguaggio funzionale basato sul formalismo del $\lambda$-calcolo. Le migliorie che introduce sono diverse e notevoli: permetteva la ricorsione, usava strutture dati avanzate come alberi e liste per rappresentare i dati, allocava dinamicamente la memoria, implementava un garbage collector e un programma LISP poteva essere usato come input per un altro programma LISP. Inoltre la scelta di utilizzare una notazione infissa permetteva un aumento notevole dell'efficienza. Nel complesso LISP ebbe un enorme successo e rimase il linguaggio di riferimento per lo sviluppo dell'intelligenza artificiale almeno fino all'uscita del PROLOG

Un altro linguaggio di rilievo sviluppato in quegli anni e' il COBOL, il cui primo pregio era di essere facilmente leggibile visto che usava operazioni scritte in inglese molto esplicative(multiply, subtract..) che rendevano superflui ulteriori commenti. Il dipartimento di difesa americano costrinse le compagnie a supportarlo nelle loro installazioni, dunque si diffuse velocemente. Per tutti questi motivi COBOL era anche stato pensato per poter essere utilizzato anche da programmatori non esperti. Per questo motivo in quegli anni molti informatici guardavano con sufficienza il COBOL pensando che i veri programmi dovessero essere scritti in FORTRAN.

## 2.3 Anni '60: ALGOL, BASIC e Simula

Gli anni '60 cominciano con l'introduzione di ALGOL. Al pari di FORTRAN, questo linguaggio influenza pesantemente lo sviluppo dei linguaggi di programmazione futuri. Aveva il difetto di non essere pensato per gli sviluppi commerciali(dipendeva troppo dall'hardware perdendo di portabilita'), quindi veniva usato per lo piu' per sviluppare algoritmi in ambito scientifico durante gli anni '60.
Introduce molte novita': per la prima volta il passaggio dei parametri viene definito in modo accurato(per valore, per nome, per riferimento) anche se il passaggio per nome ad oggi non esiste piu'. Compaiono inoltre per la prima volta in modo ufficiale le strutture if-then-else, while e for. Inoltre per la prima volta viene concepito il blocco di codice inteso come entita' avente le sue fun-

zioni nidificate e le sue variabili locali, dunque introduce lo scope statico.
Un'ulteriore innovazione portata da ALGOL e' l'utilizzo di una notazione formale per descrivere la sintassi del linguaggio stesso, chiamata Backus Naur Form.

Nel 1964 John Kemeny e Thomas Kurtz concepiscono il BASIC il cui scopo era quello di permettere a persone non eccessivamente specializzate di scrivere programmi relativamente complessi. Kemeny e Kurtz furono bravi a promuovere il proprio linguaggio rendendo disponibile gratuitamente il proprio compilatore incoraggiandone l'uso nelle scuole della loro provincia. A partire dagli anni '70 il BASIC si diffondera' sempre piu' e quasi ogni personal computer forniva un compilatore o un'interprete BASIC e in alcuni casi addirittura il BASIC costituiva l'ambiente di lavoro.
La fortuna del BASIC e' che era sufficientemente leggero da girare su pc di media potenza ma di fornire allo stesso tempo funzionalita' sufficientemente avanzate. Ad ogni modo non tutti lo apprezzavano e molti lo guardavano con aria di sufficienza.
Tra i molti miglioramenti sviluppati negli anni successivi figura l'introduzione nel 1991 di Visual BASIC anche se in quel momento dovette competere con linguaggi molto piu' avanzati come C e C++. Inoltre e' tramite il BASIC che diversi grandi personalita' dell'informatica faranno fortuna. Stiamo parlando di Bill Gates e Steve Wozniak.

Negli anni '60 si accende anche un dibattito relativamente alla programmazione strutturata/non strutturata. Nella programmazione strutturata i programmi fanno uso di strutture come procedure e costrutti che permettono di avere un codice fondamentalmente ordinato. Nella programmazione non strutturata invece si fa massiccio uso di GOTO che rende il codice meno leggibile e disordinato(per questo motivo anche detto spaghetti-code).
Nel 1966 viene dimostrato il teorema di Böhm e Jacopini che dimostra che ogni programma puo' essere scritto utilizzando i due costrutti if-then-else e while.
Il teorema mostra che si possano scrivere programmi senza GOTO. Questo fatto accende il dibattito e in molti esprimono la propria opinione. Dijkstra critica aspramente il GOTO mentre Knuth, Ritchie e Kernighan ad esempio sostengono che in alcuni casi il GOTO sia il costrutto ideale.

L'ultimo linguaggio degli anni '60 da ricordare e' il Simula, precursore dei linguaggi object oriented.
Sviluppato presso il Norvegian Computing Center di Oslo da Ole-Johan Dahl e Kristeen Nygaard era pensato per le simulazioni e negli anni che seguirono si diffuse un po' ovunque.
I due sviluppatori notarono che i processi condividevano un certo numero di proprieta' e si chiesero se ci fosse un modo per modellarle. Dahl e Nygaard si ispirarono al concetto di record class proposto da Tony Hoare nel 1965. Hoare proponeva di avere delle record class(le classi) e dei record(le istanze). Nel dicembre del 1966 Dahl e Nygaard propongono un modello di classi e sottoclassi sufficientemente chiaro in cui come oggi con la new si creavano gli oggetti e in

cui gli oggetti di una sottoclasse erano definiti per ereditarieta'. In Simula un oggetto e' un record di attivazione prodotto dalla chiamata di una classe(ovvero un particolare tipo di procedura). Il riferimento a un record era contenuto in una sorta di puntatore chiamato ref.
Altri due meriti importanti di Simula sono che implemento' un meccanismo di garbage collection per i record di attivazione non piu' attivi e permise di modellare la concorrenza dei programmi in cui ogni processo veniva eseguito alternatamente ad altri per mostrare il procedere della simulazione.

## 2.4 Anni '70: PASCAL e C

Il primo sviluppo di rilievo negli anni '70 e' la commercializzazione del PASCAL nel 1971, progettato da Niklaus Wirth. Gli obiettivi che si poneva erano di: proporre un linguaggio efficiente da compilare e da eseguire, permettere la stesura di programmi ben strutturati e organizzati, e che servisse come strumento didattico per insegnare la programmazione.
Lo sviluppo del PASCAL venne anche guidato dagli errori di progettazione di ALGOL di cui era difficile scrivere un compilatore e che non implementava diversi tipi fondamentali(char e puntatori). Per questo motivo era relegato alla descrizione dei programmi.
Il PASCAL incoraggiava la programmazione strutturata fornendo un sistema di tipi molto ricco ma anche molto rigido(ad esempio la conversione tra tipi era ammessa solo usando specifiche funzioni). Il PASCAL e' anche il primo linguaggio a contenere il concetto di insieme.
I programmi scritti in PASCAL erano leggibili ed efficienti perche' il compilatore produce un P-code specifico della macchina su cui girava.
Per tutti questi motivi verso la fine degli anni '70 il PASCAL era insegnato in molte universita' e tre ulteriori motivi ne aumentarono la diffusione: il primo e' che l'Educational Testing Service (ETS), la compagnia che negli USA amministra e scrive i test di ingresso ai college americani decise di inserire un esame di Informatica e di usare il Pascal per il test. In secondo luogo nel 1983 Anders Hejlsberg progetta il Turbo Pascal per i pc IBM. Modificando marginalmente il linguaggio si otteneva un aumento notevole i prestazioni sia in compilazione che in esecuzione. Infien quando la Apple lancio' la linea Macintosh scelse il PASCAL come linguaggio di programmazione standard.

Un altro linguaggio da ricordare sviluppato in quegli anni e' il PROLOG. Fece il suo esordio nel 1972, portato avanti da Alan Colmerauer e Philippe Roussel dell'Universita' di Marsiglia.
Il PROLOG e' un dimostratore automatico di teoremi basato sulla regola di risoluzione. Era ricorsivo, e cio' rendeva i programmi molto eleganti.
Il PROLOG ebbe pero' altre importanti applicazioni nell'intelligenza artificiale in Europa e nei database deduttivi capaci di inferire nuove informazioni a partire da quelle possedute.
Inoltre il PROLOG fu inserito come base per un importante progetto giapponese(anche se fallito): la quinta generazione di computer.

La quinta generazione doveva utilizzare processori specificamente orientati all'esecuzione efficiente e parallelizzata di clausole dei programmi logici. Questo progetto ebbe enorme risonanza perche' l'industria giapponese aveva acquisito una certa fama e desto' non poche preoccupazioni nell'industria e nei governi del tempo.
Il progetto falli' per diversi motivi: le nuove architetture RISC portarono ad aumenti netti di prestazioni superando le prestazioni ottenute in giappone. Inoltre lo sfruttamento del parallelismo era un obiettivo troppo ambizioso per il tempo. Infine il paradigma di programmazione logica si rivelo' una scelta errata perche' difficile da implementare in modo efficiente

Intanto nel 1972 Alan Kay descrive l'idea del Dynabook, un piccolo pc che assomiglia moltissimo a un tablet moderno. Per il Dynabook Kay pensa a un'interfaccia chiamata Smalltalk, primo ambiente di programmazione ad interfaccia grafica. Sebbene si ispirasse al Simula venne sviluppato da zero e in modo indipendente. Gli aspetti innovativi sono diversi: in Smalltalk esistono solo oggetti, anche le classi sono oggetti. Gli oggetti hanno dati privati e comunicano con l'esterno usando i metodi. Le classi possono anche essere definite tramite una tavola grafica riempiendone i campi predefiniti. La GUI di Smalltalk conteneva gia' agli albori un editor per classi, per illustrazioni, un word processor e un editor di font.

Sempre in quegli anni si sviluppa un nuovo linguaggio ad opera del Department of Defense(DoD) degli USA. Intorno al 1974 i vertici dell'esercito della marina e dell'aviazione avviano lo sviluppo di un linguaggio di programmazione adatto per i sistemi embedded. Dopo tre anni viene pubblicata la specifica e aperto un bando alle aziende che vogliano sviluppare tale specifica. Il progetto di Jean Ichbiah viene considerato il migliore e dopo ulteriori modifiche nel dicembre del 1980 viene rilasciato ADA in onore di Ada Lovelace. Il progetto ebbe forte risonanza e in molti si aspettavano che spazzasse via la concorrenza anche se non tutti erano d'accordo visto che il codice prodotto non era particolarmente efficiente. Nel 1991 il DoD obbligo' all'uso di ADA per lo sviluppo di tutti i progetti interni al DoD stesso. Questo obbligo perduro' sino al 1997.
I pregi di ADA erano che facilitava lo sviluppo di grandi progetti software e la sua attenzione era incentrata sullo scrivere programmi esenti da errori, fatto molto importante in campo militare.
Probabilmente a causa di questi motivi non raggiunse la diffusione che ci si aspettava ma sicuramente ADA e' davvero adatto allo sviluppo di grandi progetti in cui sicurezza e affidabilita' sono fondamentali.

L'ultimo linguaggio di rilievo che viene rilasciato negli anni '70 e' il C. La sua storia comincia negli anni '60 quando Dennis Ritchie, Ken Thompson e Brian Kernighan erano tra i ricercatori dei prestigiosi Bell Labs coinvolti nel progetto MULTICS, da cui però la AT&T(proprietaria dei Labs) decise di ritirarsi nel 1969.
Thompson procede nello sviluppo del codice fino a sviluppare un sistema ope-

rativo proprio, chiamato Unix. Il suo linguaggio era il B, derivato da un altro linguaggio, il BCPL. Il B aveva pero' una struttura piu' coincisa. Con le modifiche apportate da Ritchie emerge il C, rilasciato nel 1972.
All'inizio degli anni '70 la AT&T trova inutile investire nei sistemi operativi e decide di rilasciare copie di Unix in varie universita' insieme al sorgente scritto in C. Per questo motivo molti studenti si trovano a seguire corsi e a scrivere programmi in C all'interno delle loro universita'.
Il C ebbe un enorme successo perche': era portabile, era modulare, era efficiente, era compatto, possedeva tipi flessibili e gestiva in modo efficiente la memoria(anche se queste ultime tre caratteristiche potevano portare facilmente ad errori).

## 2.5 Anni '80: varianti del C

Negli anni '80 cominciano a circolare alcune varianti del C orientate agli oggetti. La prima e' il C++, nato sempre presso i Bell Labs grazie a Bjarne Stroustrup che decide di estendere il C semplicemente aggiungendo gli oggetti. In effetti la prima implementazione del C++ era un preprocessore che convertiva i programmi C++ in C. L'idea di implementare gli oggetti unita al fatto che non vadano perse le qualita' del C favorisce la diffusione del C++, anche se mantenere la compatibilita' tra C e il suo successore ha pero' portato ad un linguaggio non sempre chiaro e talvolta difficile da comprendere.

Sempre all'inizio degli anni '80 si sviluppa l'Objective-C che, ispirato a Smalltalk e' divenuto uno dei linguaggi piu' usati negli ambienti Apple.

Un'altra variante(anche se di molto successiva) e' C# sviluppato dalla Microsoft tra il 1999 e il 2000 che combina caratteristiche del C, del C++ e del Java.

## 2.6 Anni '90: internet e Java

All'inizio degli anni '90 James Gosling della Sun Microsystems[12] sviluppa un nuovo linguaggio pensato per le interfacce di un ricevitore di TV via cavo che rendesse interattiva l'esperienza tra utente e TV.
In quel momento il C++ era il linguaggio piu' in voga e sulle sue basi si sviluppo' il primo prototipo chiamato "C++ ++ --" a simboleggiare che alcune cose erano state tolte e altre aggiunte a partire dal C++. Il nome fu poi cambiato in Oak.
L'arrivo del Web e dei browser rese obsoleta l'idea della Sun rischiando di sacrificare anni di lavoro. Alla Sun furono pero' molto intelligenti nell'adattare Oak e a pensare di far scaricare insieme alle pagine HTML delle applicazioni eseguibili scritte nel loro linguaggio. In questo momento il linguaggio cambio' nome in Green e poi in Java. Insieme a Java venne rilasciato un browser dedicato ma il successo dell'idea costrinse tutti i principali browser a incorporare una Java

[12]Nota in altri ambiti visto che sviluppo' un suo sistema operativo, un Network File System e una famosa linea di workstation

Virtual Machine per poter garantire l'esecuzione delle applet Java.
I motivi del successo di Java furono che era facilmente apprendibile da chi conoscesse gia' il C o il C++. Anche se le applicazioni prodotte erano meno efficienti erano piu' affidabili. Ma soprattutto il codice scritto in Java era altamente portabile visto che generava un bytecode intermedio che veniva poi eseguito da una Java Virtual Machine che esisteva per ogni tipo di hardware. Per tutti questi motivi nel giro di dieci anni Java e' diventato il linguaggio di programmazione piu' diffuso

## 2.7 Verso il XXI secolo: scripting

Durante gli anni '90 si sviluppa un nuovo tipo di linguaggio: quello di scripting. Un linguaggio di scripting serve per scrivere script, ovvero programmi molto corti eseguiti da un interprete. Normalmente gli script vengono scritti in un ambiente capace di eseguirli istantaneamente. Nel tempo pero' i linguaggi di scripting hanno acquisito sempre piu' caratteristiche dei linguaggi di programmazione, ovvero essere strutturati e object-oriented. Vediamo alcuni dei linguaggi di scripting piu' usati:

- Perl: nasce alla fine degli anni '80 per manipolare testi e file. Si evolve negli anni che seguono e diventa un linguaggio general purpose il cui scopo principale e' quello di eseguire script GCI, programmi lato server il cui output viene inviato al client. Il suo successo e' dovuto al fatto che e' facilmente integrabile con altri linguaggi.

- Python: nato nel 1991 possiede una sintassi estremamente semplice. Ad esempio, i blocchi di codice sono delimitati non da parentesi o keywords, ma dall'indentazione mediante spazi bianchi. Ad oggi viene usato estensivamente in campo scientifico

- PHP(Personal Home Page): creato nel 1994 e concepito per la scrittura di pagine web dinamiche viene poi impiegato per lo sviluppo di applicazioni lato server. Nel seguito diventa un linguaggio general purpose e ad oggi puo' essere usato anche per applicazioni grafiche stand alone

- Javascript: sviluppato in 10 giorni nel 1995 era pensato per scrivere applet in modo piu' leggero di Java. La microsoft ne sviluppo' una propria versione alternativa senza successo. Ad oggi Javascript viene usato per scrivere applicazioni lato client

- Ruby: sviluppato nel 1995 e' particolarmente adatto per lo sviluppo di applicazioni web e molta della sua popolarita' la deve a Rails la piattaforma software open source scritta in Ruby per lo sviluppo di applicazioni web

# 3 Storia dei sistemi operativi

Fino agli anni '40 il concetto di programma era assai lontano da quello moderno e il concetto di sistema operativo era del tutto sconosciuto. A partire dagli anni '50 le cose cominciano a cambiare e possiamo individuare otto fasi fondamentali nella storia dei sistemi operativi

1. Job by Job/Open Shop, $\approx$ fino al 1955
2. Sistemi Batch, seconda meta' anni '50
3. Multiprogrammazione, inizio anni '60
4. Timesharing, metà anni '60
5. Sistemi concorrenti, fine anni '60
6. Sistemi per Personal Computer, seconda metà anni '70
7. Sistemi Distribuiti, anni '80
8. Sistemi per dispositivi mobili, XXI secolo

## 3.1 Job by Job

In questo momento storico l'idea di un sistema operativo non esisteva e i computer venivano usati solo da chi li aveva costruiti. Per far girare un programma era necessario: scrivere il programma su carta in codice macchina(binario, decimale..) e codificarlo su schede con opportuni perforatori, recarsi nella sala computer e preparare adeguatamente il computer predisponendo le schede perforate nel lettore di schede, eseguire il programma e valutare l'output che veniva stampato su telescrivente.
Veniva dunque speso molto tempo per gestire manualmente le schede. Lo sviluppo dei primi assembler e dei linguaggi ad alto livello rese le cose ancora piu' laboriose.
Per questo venne introdotta una figura professionale che eseguisse tutte queste operazioni, un addetto alla sala macchine che eseguiva i programmi e restituiva i risultati con eventuali anomalie (in caso di esecuzioni non terminate correttamente) al programmatore

## 3.2 Sistemi Batch

Il modo di operare dell'addetto suggeri' il successivo sviluppo: permettere al computer di pianificare almeno in parte il lavoro da fare. L'addetto poteva raccogliere pacchetti di schede detti job dei programmi da eseguire e inserirli tutti assieme(batch jobs) nel lettore di schede separati da opportune schede di controllo che indicassero l'inizio e la fine di ogni job.
Le istruzioni contenute nelle schede di controllo venivano eseguite da un programma speciale sempre residente in memoria, chiamato Resident Monitor.

Rispetto alla fase Job by Job ora in memoria risiedevano due programmi e cominciava ad emergere una distinzione tra programma e dati. I sistemi di questo tipo vennero chiamati batch system.
Un'ulteriore miglioria era quella di utilizzare un "computer satellite" che gestisse l'input/output mentre il computer principale eseguiva un altro programma.
Grazie a queste migliorie l'efficienza del sistema in generale aumentava ma a volte il singolo utente doveva attendere ore o giorni prima di ricevere i risultati.

Un tipico esempio di sistema Batch e' il BKS, sviluppato per il Philco Transac 2000. Il Philco possedeva circa 32.000 word di memoria di cui 2600 circa per il BKS. Il Philco possedeva diversi lettori di nastro per gestire meglio il lavoro ma come spesso accadeva in quegli anni non esisteva una protezione per la macchina da usi impropri e ai programmatori si raccomandava di non usare istruzioni di riavvolgimento nastri, non indicassero locazioni di memoria del sistema..

## 3.3 Multiprogrammazione

L'efficienza dei sistemi batch era fortemente limitata dall'uso di supporti di memorizzazione ad accesso lento. Il termine multiprogrammazione compare per la prima volta in un articolo di Christopher Strachey del 1959.
L'arrivo degli hard-disk permise di apportare notevoli miglioramenti perche' a differenza della lettura dai nastri, con gli hard disk si poteva portare avanti in parallelo in modo efficiente programmi diversi.
Quindi quando un programma veniva trasferito completamente su hard disk poteva essere eseguito e quando terminava l'esecuzione se ne potevano stampare i risultati.
Questa modalita' asincrona di esecuzione veniva chiamata spooling(dalle iniziali di simultaneous peripheral operation on line) e il suo principale vantaggio era che se un programma doveva eseguire le operazioni di I/O, poteva fermarsi e lasciare l'esecuzione ad un altro programma. Questo modello viene detto multitasking cooperativo.

Un noto esempio di sistema a multiprogrammazione e' quello dell'Atlas progettato nel 1961 per L'Atlas computer dell'Universita' di Manchester.
Nell'Atlas compaiono due concetti fondamentali dei sistemi operativi moderni: le system call per le operazioni delicate e l'implementazione della memoria virtuale con paginazione su richiesta. Per la prima volta si fa strada l'idea che il sistema operativo venga invocato di frequente. In particolare l'Atlas possedeva una core store per il processo in esecuzione e una drum store per i processi non in esecuzione.
Quando il programma in esecuzione si fermava per una operazione di I/O le sue pagine venivano trasferite dalla core store alla drum store, e al suo posto veniva fatto partire un altro programma. In attesa di poter tornare in esecuzione nella core store, più programmi si potevano trovare nella drum store. Se un programma della core store richiedeva una pagina della drum store gli veniva fornita dal sistema operativo ma se la pagina non era neanche nella drum store allora

il processo veniva tolto dalla core store in attesa che recuperasse i dati dalla system input tape(un'altra memoria piu' ampia e piu' lenta), nel frattempo un altro programma veniva eseguito.
Dunque Atlas introdusse un meccanismo sofisticato di gestione della memoria in cui veniva anche utilizzata la core store come una memoria cache mentre la drum store svolgeva il ruolo di memoria principale.

Sempre nel 1961 l'americana Burroughs Corp. presenta il B5000, primo computer ad implementare un'area di memoria a stack dove salvare i valori temporanei. Il sistema operativo del B5000(chiamato Master Control Program o MCP) fu anche il primo sistema ad implementare una memoria virtuale segmentata e dunque a soffrire di trashing che veniva risolto avviando nuovamente la macchina.
Tre ulteriori dati importanti furono che l'MCP fu il primo sistema operativo scritto completamente ad alto livello(con una variante dall'Algol), che possedeva un timer hardware per terminare processi in esecuzione che non rilasciassero il processore e infine che era configurato con due processori, rendendolo il primo sistema operativo multi-processore.

## 3.4 Timesharing

Nel gennaio del 1959 viene pubblicato un report di John McCarthy in cui viene proposto un nuovo modello di sistema operativo a suddivisione di tempo. L'idea di McCarthy era di avere un'insieme di utenti che tramite delle console remote potessero eseguire in contemporanea un programma per uno.

Il primo sistema operativo di questo tipo fu il CTSS sviluppato nel 1961 al MIT. L'obiettivo era quello di permettere a piu' persone in contemporanea di utilizzare il computer. CTSS doveva garantire tempi di risposta ragionevoli ad ogni utente. Una delle migliorie che porto' fu' quella di poter interagire col sistema in particolare durante il debugging. I progettisti del CTSS dovettero risolvere problemi che non erano mai stati affrontati prima e che oggi costituiscono l'essenza di un sistema operativo.
I vincoli che CTSS dovette rispettare affinche' tutto funzionasse furono molteplici: i programmi dovevano essere tenuti contemporaneamente in memoria primaria e dovevano poter essere spostati da memoria primaria a secondaria e viceversa. Il programma in esecuzione doveva poter essere interrotto e poter riprendere successivamente oltre che garantire operazioni di I/O controllate per evitare usi impropri. Inoltre se le operazioni di I/O erano lente CTSS doveva spostare il programma in memoria secondaria. Infine gli utenti dovevano poter interrompere un loro programma malfunzionante e poter interrogare il sistema per sapere chi fosse in esecuzione e altre informazioni.
CTSS ci riusci' e inizialmente si potevano collegare fino a 3 utenti ma nel seguito si arrivo' a poterne gestire fino a 32. Ci sono ancora un paio di cose interessanti da dire su CTSS. La prima e' che il sistema era dotato di un login(remoto) e la seconda e' che su di esso gli utilizzatori cominciarono a sviluppare nuovi

comandi e programmi per il CTSS stesso, come il programma MAIL utilizzato per scambiarsi messaggi fra utenti.

Da CTSS prese vita MULTICS il sistema operativo piu' ambizioso degli anni '60. Non ebbe un gran successo perche' era decisamente esagerato. Pensato per gestire fino a centinaia di utenti con una memoria virtuale paginata con un'architettura multiprocessore con memoria condivisa. Venne sviluppato in un linguaggio dedicato e permetteva di eseguire programmi FORTRAN, COBOL e LISP e nel tempo se ne aggiunsero altri. Il Multics era dotato di una grande memoria primaria da 2 Mbyte, possedeva 135 Kbyte di Kernel(fonte di scherno per il mondo informatico a quei tempi) ed era composto in totale da 300.000 righe di codice.

Anche se il MULTICS non fu un successo ispiro' Unix e la struttura del File System moderno. Ogni utente in MULTICS poteva avere una sola "directory", e nel 1965 Robert Daley e Peter Neumann, che facevano parte del gruppo che lavorava al MULTICS pubblicano un articolo che contiene l'essenza di ciò che ancora oggi chiamiamo File System Gerarchico.
Secondo l'articolo gli utenti devono usare i file solo attraverso i nomi simbolici e la mappatura e' lasciata alla macchina. Inoltre i file devono essere organizzati gerarchicamente tramite directory e sotto-directory dando i natali al concetto di pathname di un file. Sempre secondo l'articolo gli utenti devono poter manipolare i file e rimodellare la propria porzione di file system.
Il sistema operativo Titan implementera' per primo questo tipo di file system consentendo a circa 700 utenti di memorizzare il propri file. Il tutto veniva gestito da una File Allocation Table.

Un altro sistema operativo da ricordare (non per innovazione ma per diffusione) e' l'OS/360. All'inizio degli anni '60 i vertici IBM si accorsero che le incompatibilita' tra computer di fascia alta e fascia bassa diminuivano le possibilita' di condivisione e cooperazione. Per questo motivo la linea IBM sembrava meno appetibile. Nel 1964 viene rilasciata una linea di computer contenenti sistemi operativi specifici per la macchina ma "compatibili" fra loro. Questo permise un'ampia diffusione del marchio IBM, anche se OS/360 era ancora un sistema multiprogrammato e non timesharing poiche' al momento del rilascio non c'era stato il tempo di implementare un sistema timesharing. Inoltre il grande sforzo effettuato da IBM per sviluppare del software su macchine diverse getto' le basi del software engineering moderno.

L'ultimo sistema operativo fondamentale per questa fase e' il sistema operativo UNIX. Non porto' con se' idee particolarmente innovative, ebbe solo il pregio di metterne insieme molte di quelle gia' esistenti in un momento in cui il sistema operativo piu' diffuso(OS/360) era ancora multiprogrammato.
Ken Thompson nel 1969 insieme a Dennis Ritchie sviluppano in pochi mesi la versione basilare di UNIX che riscontra un certo successo tra i colleghi. Successivamente UNIX viene aggiornato e nel 1970 assume il suon nome. Nel 1973

viene riscritto in C diventando molto piu' leggibile e facile da installare su hardware diversi.
La AT&T non puo' venderlo per motivi legali e quindi viene distribuito gratuitamente. Grazie al gruppo USENIX si diffonde e nel 1978 si contavano piu' di 600 installazioni. Negli anni si svilupperanno le due famiglie piu' importanti di UNIX, la BSD e la System V.
Nel 1974 RItchie e Thompson rilasciano l'articolo *The Unix Time-Sharing System* che contiene la quintessenza del sistema operativo moderno. Ad oggi il documento e' ancora aggiornato.

## 3.5 Sistemi concorrenti

A meta' degli anni '60 i sistemi operativi soffrono di trashing, deadlock e starvation. Nei successivi dieci anni vi fu un grosso sforzo per rendere i sistemi operativi piu' comprensibili prevedibili e affidabili andando a sviluppare le principali tecniche di programmazione concorrente(semafori, monitor, regioni critiche). Nel 1968 Edger Dijkstra sviluppa il sistema THE, un semplice sistema operativo multiprogrammato a strati sovrapposti che trasformano la macchina fisica in una macchina virtuale piu' semplice da usare che usa semafori per gestire le comunicazioni tra processi.

Un altro sistema fondamentale emerso nel periodo fu l'RC4000 sviluppato da Per Brinch Hansen, che seppur non completo fu il primo vero esperimento di sviluppo di un kernel attorno al quale sviluppare il sistema operativo in base alle proprie esigenze. Ricordiamo inoltre che Hansen e' il padre del primo linguaggio concorrente, il Concurrent Pascal che conteneva le primitive per sincronizzare e far comunicare i processi. Le sue potenzialita' vennero evidenziate con il sistema operativo Solo il cui intero codice era estremamente sintetico.

## 3.6 Sistemi per personal computer

Il fatto che all'inizio degli anni '70 cominciassero a circolare computer di dimensione e capacita' limitate, spinse gli sviluppatori a percorrere nuove strade.
Il capostipite fu l'OS 6 sviluppato alla Oxford University. Era capace di funzionare su un computer dotato di soli 32 Kbyte di memoria principale. Inoltre diede la base per il sistema operativo Alto progettato alla Xerox. Alto era molto semplice e poteva eseguire un solo processo alla volta in modalita' sequenziale. E' rimasto famoso per la sua interfaccia grafica(che pero' era legata all'applicazione e non al sistema operativo). Dopo diversi modelli nel 1981 venne rilasciato lo Xerox Star che era il primo sistema dotato di mouse e interfaccia a finestre sul mercato. L'idea era quella di creare un vero e proprio ufficio elettronico con carta, classificatori, caselle di posta e calcolatrici. Non ebbe pero' un gran successo commerciale perche' era costoso.

La storia della Microsoft invece comincia grazie ad un colpo di fortuna. Vediamo il perche'. Nei primi anni '70 Gary Kildall era un giovane ricercatore

informatico che mette a punto un suo linguaggio di programmazione, il PL/M e un sistema operativo chiamato CP/M(Control Program/Monitor) per controllare i floppy disk.
Il CP/M non interesso' alla Intel per la quale Kildall lavorava. Dunque fonda una propria compagnia e lo commercializza in proprio. Inaspettatamente diventa popolarissimo e l'azienda di Kildall fattura milioni di dollari. Fu quindi il primo sistema operativo ad ampissima diffusione. Questo avvenne perche' il CP/M era composto da tre parti:

- CCP(Console Command Processor): interprete di comandi
- BDOS(Basic Disk Operating System) insieme di routine necessarie che implementavano il CCP
- BIOS(Basic Input/Output System): conteneva le istruzioni dipendenti dalla macchina

Questa progettazione a strati consentiva di far girare CP/M su macchine diverse andando a riscrivere solo il BIOS.
Nel 1975 Paul Allen e Bill Gates fondano la Micro-soft e scrivono un compilatore BASIC usando il CP/M. Quando la IBM entra nel mercato dei personal computer si rivolge a Kildall per poter usare CP/M sui suoi computer. Inspiegabilmente l'accordo non viene siglato e la IBM si rivolge alla Microsoft che ingaggia Tim Paterson. Insieme a quest'ultimo viene creato MS-DOS.
Quando Kildall se ne accorge minaccia di fare causa e la IBM offre la possibilita' di utilizzare CP/M al posto di PC-DOS ma ad un costo sei volte superiore. Di fatto il mondo continuo' ad usare PC-DOS.
Nel 1985 verra' introdotta un'interfaccia grafica per MS-DOS chiamata Windows. Dopo varie versioni nel 1995 viene rilasciato Windows 95 che segna la fine dell'era DOS.

Alla Apple come al solito guardano con aria di sufficienza a tutti questi eventi dimenticandosi che il loro primo computer venduto su larga scala, cioe' l'Apple II montava DOS. Il DOS era in sostanza il sistema operativo di un pc ridotto all'essenziale su un floppy disk che veniva caricato in memoria principale permettendo di lanciare i programmi degli utenti. Naturalmente su Apple II girava Apple-DOS e l'idea vincente fu di pubblicare sempre le caratteristiche software della macchina per permettere a sviluppatori terzi di sfruttarla al massimo. VisiCalc ad esempio era il primo foglio elettronico disponibile su mercato che per certe aziende divenne cosi' fondamentale da giustificare l'acquisto di un Apple II.
Ad ogni modo il primo sistema ad intefaccia grafica Apple non fu un grande successo: si chiamava Lisa ed essendo troppo lento e inaffidabile venne rimpiazzato un anno dopo dal piu' celebre Macintosh.
Macintosh usava il sistema operativo System che successivamente cambio' nome in Mac OS e fu' rivoluzionario perche' per la prima volta rimosse completamente l'interfaccia da linea di comando con una a finestre. Mac OS arriva fino alla

versione 9 e poi cambia nome in Mac OS X. Il passaggio da Mac OS a Mac OS X non e' solo di nome visto che Mac OS X e' un sistema Unix-like.

Sempre in questo periodo, nel 1983, Richard Stallman al MIT di Boston da via al progetto GNU. Un progetto collaborativo di massa per lo sviluppo di software libero. L'idea e' di avere un software che tutti siano in grado di usare, condividere studiare e modificare. Naturalmente Stallman pensa anche ad un sistema operativo il cui sviluppo procede a rilento. Nel 1986 viene pubblicata un'opera fondamentale per i sistemi Unix chiamata *The Design of the Unix Operating System* di Maurice Bach, e ne 1987 Andrew Tanenbaum rilascia i sorgenti di MINIX un sistema operativo Unix-like per scopi didattici.
Ma MINIX e' un sistema a 16 bit e comprare una licenza Unix costava troppo. Per questo Linus Torvalds sviluppa Freax, che verra' poi rilasciato col nome di Linux. Torvalds lo sviluppa usando un GNU C compiler e basandosi sull'opera di Bach.
Nel 1992 Torvalds, rilascia il kernel Linux sotto licenza GNU GPL(General Public Licence).
Linux gode subito di grande successo e gran parte del lavoro viene portato avanti dai programmatori della sua comunita'. Gli sviluppatori sono raggruppati sotto varie compagnie e progetti che forniscono varianti del sistema(Fedora, RedHat, Ubuntu..)

## 3.7 Sistemi distribuiti

L'idea del sistema operativo distribuito e' quella di integrare il funzionamento di piu' computer per dare l'illusione di avere una unica unita' computazionale che condivida il file system che e' distribuito tra i vari nodi del sistema. Gli utenti mandano in esecuzione un programma ma e' il sistema operativo a decidere su quale nodo vada eseguito. Gli eventuali guasti di un nodo sono mascherati dal sistema operativo.
Negli anni '80 si diffondono alcuni sistemi distribuiti basati su diversi computer collegati da una rete Ethernet che permetteva di condividere alcune risorse come stampanti e il file system.
Con Unix cominciano a svilupparsi i primi sistemi distribuiti che sono essenzialmente basati sul meccanismo di Remote Procedure Call(RPC), un'estensione dell'Inter Process Communication.
Fu la compagnia Sun a implementare le prime RPC in ambiente Unix nel 1984 che costituirono la base per il Network File System(NFS), il primo file system distribuito su una rete di computer.
In effetti i sistemi distribuiti non sono mai andati oltre una fase sperimentale e non si e' mai formato uno standard. Menzioniamo ora due esempi di sistemi distribuiti.

Unix United System, sviluppato nel 1982 presso l'Universita' di Newcastle trasformava cinque macchine in un ambiente Unix omogeneo. Il file system era condiviso e gli utenti potevano usare ogni risorsa remota come se fosse locale.

Un altro sistema era il sistema Amoeba sviluppato in Olanda da Tanenbaum che era due volte piu' veloce di NFS

## 3.8 Sistemi per dispositivi mobili

Nella meta' degli anni '90 l'avanzamento tecnologico permette di costruire dispositivi di piccola dimensione il cui sistema operativo deve integrare molte funzioni: telefonia, navigazione in rete, riproduzione di informazione multimediale..
Il sistema operativo mobile comunque non differisce molto da quello classico. Deve pero' fornire una certa funzionalita' real time, gestire al meglio la memoria e la batteria. I primi dispositivi mobili che entrano nel mondo dei computer sono il Nokia 9000 del 1996 che poteva inviare fax e connettersi a Internet, e il MessagePad della Apple, un dispositivo palmare dotato di calendario rubrica..
Dall'unione di questi due dispositivi nasce lo smartphone e nei primi anni 2000 anche il tablet.

I sistemi operativi piu' diffusi per questi dispositivi sono due. Il primo e' Android, la cui compagnia omonima viene fondata nel 2003 e nel 2005 acquisita da Google. Android e' basato sul kernel Linux ma viene sviluppato in Java. Il suo pregio e' di poter girare su qualsiasi dispositivo mobile, che gli permette di avere un ampio parco applicazioni. Ad oggi Android e' usato sul 65%-80% dei dispositivi mobile.
L'altro sistema e' Apple iOS che usa un kernel XNU ibrido tra BSD Unix e il microkernel Mach. Ad Apple iOS ad oggi spetta circa il 15%-25% del mercato.