Cortosi Vittorio

Storia dell’informatica

Prima Parte del corso

Indice

Architetture e sistemi di calcolo

- Chi era **Charles Babbage**?
  - Figura eminente del suo tempo a cui fu affidata anche la cattedra di Matematica di Cambridge.
  - A metà del diciannovesimo secolo concepisce (dopo il primo progetto mai dato alla luce, chiamato difference engine) **l'Analytical Engine**, da molti considerato il primo computer avente architettura simile ai computer moderni.
  - Nel suo percorso per riuscire sviluppare l'Analytical Engine, Babbage viaggia per l'europa (Torino in particolare), incontrando **Giovanni Plana**, a cui chiede di pubblicare un resoconto delle conferenze di Charles:
    - Plana affida l'incarico ad un suo allievo: Luigi Federico Menabrea.
  - L'articolo pubblicato venne poi tradotto in inglese da **Ada Lovelace**
  - Con un po' di esagerazione è tradizione considerare Charles Babbage l'inventore del primo computer moderno e Ada la prima programmatrice della storia.

- Cos'è l'algebra di Boole e perché è importante?
  - Nel 1854 l'inglese George Boole pubblica un articolo in cui vengono illustrati i principi dell'algebra booleana: strumento su cui si basano i calcolatori digitali moderni
  - Articolo considerato puramente teorico fino al 1930 circa

- Cosa sono le **schede perforate** e che influenza hanno avuto?
  - La prima ricorrenza storica delle schede perforate si ha nel 1801 con **Joseph Jacquard** che ebbe l'idea di usare schede perforate (in origine rotoli di carta perforata da Basile Bouchon nel 1725), per il controllo automatico di un telaio in cui una scheda perforata
    permetteva di stabilire quale combinazione di fili di diverso colore
    dovessero essere usati in ogni riga di un tessuto.
  - L'idea di usare schede perforate permase nel tempo fino al 1980 in quanto:
    - Le schede rappresentavano un metodo semplice ed economico per codificare informazioni che dovevano essere elaborate in modo automatico
  - Le schede venivano perforate attraverso un perforatore di schede
  - Ogni scheda conteneva una linea del programma da eseguire, altre contenevano i dati del programma. (Io ancora non ho capito come cazzo facevano).

- Chi era **Herman Hollerith**?
  - Nel 1890 venne bandita in USA una gara per chi riuscisse a trovare un sistema in grado di velocizzare i dati raccolti dal censimento. Hollerith vince il concorso con la sua Macchina Tabulatrice Elettro-Meccanica:
    - La macchina gestiva schede perforate, ognuna delle quali conteneva informazioni anagrafiche di un cittadino (erano l'input della macchina)
    - la macchina era collegata ad un circuito elettrico, che veniva acceso o spento a seconda della presenza o meno dei buchi nelle schede (Wiki).
  - La macchina funzionò perfettamente riuscendo ad esaminare fino ad 800 schede al minuto.

- Successivamente estese il campo di applicazione della sua macchina (ad esempio per le compagnie ferroviarie, assicurazioni, etc..), fondando la **Tabulating Machine Corporation** per la produzione e la commercializzazione delle varie invenzioni/macchine. **Dalle ceneri di questa compagnia nascerà la International Business Machine Corporation (IBM).**

- **Relè e Valvole termoioniche**, perché sono importanti?
  - Sostanzialmente entrambi posso implementare il concetto interruttore (aperto/chiuso) a comando elettrico (che è alla base di un qualsiasi circuito digitale booleano)
  - La differenza sostanziale tra i due:
    - Il relè è elettro meccanico, la valvola termoionica è completamente elettrica.
    - La valvola termoionica è un componente elettrico attivo, quindi in grado di amplificare il segnale elettrico. I vantaggi rispetto ad un relè sono evidenti, perché non ci sono parti meccaniche in gioco (quindi maggior velocità di commutazione e minor usura) e il consumo di corrente per azionare l'interruttore è inferiore a quello richiesto ad una elettrocalamita.

- L'ENIAC
  - L'ENIAC fu progettato con l'entrata in guerra dell'esercito americano, in quanto avesse bisogno di calcolare traiettorie e tavole balistiche per le armi sul campo (cannoni, tank, …).
  - La realizzazione del progetto è merita soprattutto di Herman Goldstine e John Mauchly che ebbero l'idea di sostituire le persone che effettuavano il calcolo manualmente e con strumenti analogici con un calcolatore elettronico.
  - Goldstine e Mauchly, incontrandosi alla Moore School, insieme a John Eckert si misero d'accordo e successivamente stipularono un contratto con l'esercito per lo sviluppo dell'Eletronic Numerical Integrator and Computer.
  - Sfortunatamente i lavori terminarono troppo tardi (autunno 1945) troppo tardi per la guerra e l'ENIAC fu presentato solo nel febbraio '46.
  - La macchina era comporta da 20 macchine addizionatrici e unità aritmetico logiche controllate da un enorme rete di cavi elettrici e switch riconfigurabili manualmente (una merda).
  - Una particolarità dell'ENIAC è che veniva programmato riconfigurando opportunamente gli switch ed i collegamenti elettrici a mano:
    - "Programmare" l'ENIAC significava decidere per ogni accumulatore quale operazione aritmetica doveva eseguire, da quale altro accumulatore doveva ricevere l'input, e a quale accumulatore trasmettere l'output.
    - La programmazione di ogni accumulatore avveniva riconfigurando manualmente i collegamenti elettrici e gli switch di cui ogni accumulatore era dotato
    - Il programma da eseguire vero e proprio consisteva nell'esecuzione in sequenza delle operazioni programmate su ogni accumulatore.
    - 7. L'ordine di esecuzione veniva a sua volta "programmato" operando, sempre manualmente, su una serie di switch e collegamenti elettrici del master programmer (inclusi eventuali loop): una merda abissale.

- **L'EDVAC** e l'architettura Von Neumann
  - Ideato da Ecker e Mauchly nel 1944, l'EDVAC fu la prima **idea** di computer a programma memorizzato.
  - Insieme all'EDVAC descrivono un nuovo tipo di memoria (la MDL), con lo scopo di memorizzare i programmi da elaborare, in memoria.
  - Successivamente Von Neumann si aggiunse come consulente.
  - **Von Neumann scrisse un articolo sull'EDVAC** in cui espone la prima descrizione della struttura logica di un computer che fa uso di programma memorizzato:
    - L'articolo suscitò subito polemiche in quanto era firmato a sola firma Von Neumann ma che conteneva idee di un computer a programma memorizzato che erano state sviluppate anche da altri.
    - Il motivo per il quale l'articolo fosse a sola firma Von Neumann è che ai tempi era considerato normale firmare l'articolo con il componente più "importante" del gruppo. Inoltre, era molto più famoso degli altri ricercatori e considerato una delle più grandi mente matematiche del secolo.
    - Dunque, il report avrebbe sicuramente avuto più risonanza se fosse stato a nome di von Neumann, anziché a nome di uno (quasi) sconosciuto come Goldstine (Cit. Gunetti).
  - **È dal nome dell'autore di questo articolo che deriva l'espressione Architettura Von Neumann**.

- **L'SSEM**
  - **Primo e vero computer a programma memorizzato**.
  - Progettato da **Williams e Kilburn** inizialmente per testare il funzionamento di un nuovo tipo di memoria: la WTK, che elimina alcuni difetti della MDL:
    - Accesso sequenziale
    - Funzionamento a temperature controllate
    - Ingombranti
  - Le WKT sono da considerare la prima vera e propria memoria ad accesso diretto (RAM):
    - Consisteva in un tubo catodico la cui superficie era suddivisa in una matrice di n slot, ognuno dei quali poteva essere polarizzato +/-.
  - Le WKT, quindi, occupavano meno spazio ed erano ad accesso diretto, quindi più veloci, ma meno affidabili con il tempo rispetto alle MDL.

- Whirlwind, perché è ricordato?
  - Per aver introdotto alcune innovazioni che ritroviamo ancora oggi nei computer moderni:
    - L'introduzione del bit-parallel che a differenza del bit- serial permetteva di elaborare più bit in parallelo (nel caso del Whirlwind 16). Questo richiedeva più hardware ma rendeva l'elaborazione più veloce di qualsiasi altro computer.
    - L'introduzione di un nuovo tipo di memoria: Core Memory (memorie a nucleo magnetico).
    - L'introduzione di microprogramma: lo strato intermedio tra le porte logiche di un processore e le sue istruzioni macchina:

        - Per la prima volta era presente un control store che stabiliva, per ogni istruzione, come pilotare le porte logiche del datapath

- Le memorie che precedono la RAM a semiconduttore
    - Delay Line Memory
    - William Kilburn Tube
    - Memorie a taburo rotante
    - Macnetic Core Memory

- **Transistor**, cosa lo rende migliore?
    - Inventato ai laboratori Bell, è una delle invenzioni più importanti della storia.
    - In maniera simile alla valvola termoionica può essere usato come interruttore o amplificatore di segnale. Quando usato come interruttore, rispetto alle valvole termoioniche, commuta molto più velocemente il suo stato (Dunque può funzionare a frequenze più elevate (del clock di un computer, ad esempio).
    - Rispetto alle valvole termoioniche:
        - Costano meno
        - Consumano meno
        - Scaldano meno
        - Possono essere miniaturizzati fino quasi a raggiungere la dimensione atomica.
        - Funziona a frequenze più elevate
        - Più affidabili
    - La tecnologia verrà migliorata con gli anni fino ad arrivare a costruire transistor a semiconduttore.

- Nastri magnetici come memorie di massa
    - Il primo computer ad utilizzare questa tecnologia fu l'UNIVAC I (1951).
    - Chiamato anche UNISERVO, il nastro magnetico permetteva di memorizzare su un nastro di carta dei dati permanenti utilizzando una telescrivente oppure mediante un lettore di schede perforate, collegati al dispositivo che gestiva il nastro.
    - Ricoprirono un'importanza fondamentale nel campo delle memorie fino agli anni 70 con l'invenzione degli Hard Disk.

- Memorie a nucleo magnetico
    - Forma predominante di memoria centrale dal 1955 al 1975
    - Le ricerche nascevano con l'esigenza di sostituire le WKT (poco affidabili)
    - Sono memorie non volatili, quindi in cui l'informazione permane nel tempo.
    - Il funzionamento è il seguente: Fili elettrici adeguatamente intrecciati ai nuclei di ferrite (che di solito hanno una forma ad anello) permettono di generare il campo magnetico col quale polarizzare opportunamente ogni singolo nucleo, memorizzando così su di esso un valore 0 oppure 1. Un terzo filo passante tra gli anelli (non presente in figura) permetteva la lettura del bit memorizzato, che poi però andava riscritto perché l'anello di ferrite si smagnetizzava
    - Erano chiamate "core memory" o "core", ed è per questo che la memoria centrale oggi viene spesso chiamata così. Stessa cosa per i "core dump".

- Terza generazione di Computer

- Per terza generazione si intende quei Computer costruiti con tecnologia a semiconduttore e circuiti integrati (La seconda gen è quella a transistor e la prima quella a valvole).

- **Perottina**
  - Sviluppata da Pier Giorgio Perotto tra il 1962 e il 1964, è considerata da alcuni il primo personal computer della storia
  - Presentata come calcolatrice di fatto era un vero e proprio computer, dotato di 10 registri a 22 bit e programmabile in linguaggio macchina.

- **Seymour Cray** ed il CDC 6600
  - Figura quasi leggendaria nel campo dei supercomputer.
  - Era un ingegnere statunitense che progettò il primo supercomputer di successo della storia (nel 1965): il CDC6600.
  - Raggiunse i 40MFlops la versione 7600
  - Cray partorì l'idea di cedere a processori periferici alcuni dei compiti che fino a quel momento erano assegnati alla CPU (come Comunicazione con le periferiche, memorie):
    - Affiancò così al processore principale processori periferici che operando in parallelo al processore principale, lo liberava dalla gestione diretta di vari moduli a cui era collegato.
  - È sulla base dell'idea di Cray che oggi abbiamo quello che chiamiamo Chipset: Northbridge e southbridge che comunicano con dispositivi più lenti.
  - Introdusse anche la tecnologia superscalare

- Ruolo della **intel** nella storia dei microprocessori:
  - Fondata da Noyce e Moore nel 1968, inizialmente concepita per produrre memorie RAM a semiconduttori.
  - Gli eventi fondamentali della nascita della Intel hanno a che fare con eventi che accadono fuori dalla Intel stessa
  - Primo evento:
    - Nel 1969 la Busicom contatta la Intel (appena nata) per costruire una versione elettronica della sua Calcolatrice (progetto che ricalcava la Perottina)
    - Ted Hoff e Stan Mazor (della intel) mettono a punto un progetto per la Busicon che raccoglie 12 TTL IC in un unico Chip.
    - Il progetto era guidato da Federico Faggin a cui si era aggiunto successivamente Masatoshi Shima della Busicom.
    - Il chip è pronto per la fine del 1971 ma la Busicom non era rimasta impressionata dal progetto e non potendo affrontare i costi di produzione, lo abbandonarono **cedendo** alla Intel i diritti di produzione a patto che la Intel restituisse alla Busicom i costi di produzione affrontati fino a quel momento.
    - Faggin decise di chiamare il Chip 4004: primo processore sviluppato completamente su un'unica fetta di silicio.

  - Secondo evento:
    - La **Datapoint/CTC** (fondata da Ray e Roche), nell'autunno del 1969, affida a Poor e Pyle un progetto: un semplice computer in grado di emulare un qualsiasi terminale telescrivente. La leggenda narra che

sviluppparono lo schema dell'architettura di base (x86) dei microprocessori attuali (Non ARM), instruction set incluso, nel pavimento del soggiorno della casa di Poor.
        - Poor aveva l'intenzione di costruire il chip con un centinaio di IC TTL già prodotti da Texar e Intel, ma Ray e Roche inviano Poor a parlare con la intel per capire se si potesse realizzare un unico IC TTL
        - Inizialmente intel rifiuta, ma dopo le minacce della Datapoint (che acquistava le RAM da Intel), decide di accettare.
        - Intel se la prende comoda, e la Datapoint arriva ad un punto in cui deve per forza presentare un prototipo del nuovo prodotto. Presentarono un prototipo usando circuiti integrati e nel 1970 commercializzarlo (Datapoint 2200) senza il chip concluso da Intel.
        - Intel terminò il progetto alla fine del 1971, ma alla Datapoint questo servì ben poco (Datapoint stava già lavorando ad una macchina più potente).
        - Datapoint ed intel si misero d'accordo (Deja vu): l'intel avrebbe trattenuto i diritti del nuovo chip e la Datapoint non avrebbe pagato i costi di progettazione (Affare peggiore della storia).
        - Successivamente Intel presentò, nel 1972, quello che è il padre dell'architettura x86: Intel 8008

- Come si sono evoluti i microprocessori?
    - La tecnologia di costruzione ha giocato un ruolo importante nell'evoluzione dei microprocessori: la capacità di inserire componenti sempre più miniaturizzati, permette di implementare meglio le tecniche ideate negli anni precedenti:
        - Velocità: La frequenza del clock passa dai 0,8Mhz del 8008 ai quasi 4GHz del Pentium 4
        - Memoria indirizzabile: si passa dai 4 bit del 4004 ai 64 bit dei processori attuali
        - Livelli di cache sempre più grandi e sempre più livelli di cache introdotti (fino ai 3 attuali).
        - Introduzione del RISC
        - Pipeline
        - Multithreading
    - La capacità di costruire chip sempre più piccoli viene indicata con "Tecnologia a x Micro/nano metri (attuali 5nm Apple A14).

- **Il primo Personal Computer**
    - Nel 1973 alla Xerox, viene sviluppato il primo personal computer dotato di mouse e GUI: **Alto**.
    - Il progetto influenzò altri progetti tra i quali: il Machintosh e le workstation Sun. (Alto fu sviluppato quasi 10 anni prima del primo Machintosh).

- Il ruolo della **Apple** nella storia dell'Informatica
    - Steve Jobs e Wozniak, uno genio dell'elettronica (Woz) e l'altro visionario, lasciarono il college senza laurearsi, conoscendosi già da quando Jobs aveva 16 anni e Woz 21. Jobs spinse Woz ad interessarsi di più ai computer, tanto che spinse Woz a partecipare ad incontri di appassionati di computer con l'intento di costruire un computer utilizzando un terminale video che aveva costruito negli anni precedenti.

- Non avendo disponibile denaro per acquistare i nuovi intel/Motorola, il suo progetto rimane su carta fino alla comparsa del 6502, un processore economico che permise a Woz di costruire il suo primo Computer e metterlo in mostra al club di hoppysti di computer. Qui incontra di nuovo Jobs, il quale frequentava il club perché aveva intuito il potenziale commerciale dei compuiter.
- Jobs riesce a trovare un acquirente al quale vendere il computer appena costruito, chiamato e venduto con il nome di **Apple 1**.
- Nel 1976 insieme a Mike Mrkkula fondano la Apple Computers Inc. (successivamente rinominata Apple Inc. nel 2007)
- Seguono poi Apple 2 ed **Apple 3**, che fu una disgrazia per Apple, in quanto presentava un problema di surriscaldamento dovuto alla mancanza del sistema di raffreddamento (voluto da Jobs) che portò la Apple a ritirare 14k esemplari. **Inoltre, la IBM stava per lanciare il suo personal computer** (RIP Apple).

- Il personal Computer di IBM
  - Nel 1981, la IBM presenta il suo personal computer: **IBM 5150**.
  - Data la diffusione del 5150, i produttori di periferiche, schede di espansione e software compatibili con i PC IBM cominciano a sbucare come funghi: **"IBM COMPATIBILE"** diventa una qualità irrinunciabile per garantire il successo sul mercato.
  - Nel giro di 3 anni al IBM detiene il 56% del mercato dei PC contro il 16% di Apple

- **Il Machintosh 128K**
  - Il successo dell'IBM diede in qualche modo una scossa ad Apple che mise sul mercato qualcosa di rivoluzionario:
    - A fine 1979, ad alcuni dirigenti Apple viene concesso di visitare per tre giorni Xerox Parc. Da qui nascerà l'idea di produrre un nuovo PC dotato di Interfaccia grafica a icone: LISA.

  - Apple produsse il LISA ma troppo costoso e quindi di scarso successo.
  - Subito dopo seguì il Machintosh 128K **lanciato sul mercato con una imponente campagna pubblicitaria** a partire dal gennaio 1984. Il machintosh 128k ebbe subito un enorme successo e segna l'inizio della divisione tra PC e MAC

- **RISC Revolution**
  - Tra la fine degli anni 70 e 80, Dadiv Patterson e John Hennessy conducono esperimenti su un nuovo modello architetturale e un nuovo instruction set: i risultati daranno luogo a nuovi tipi di processori caratterizzati da istruzioni macchina semplici, a struttura regolare e dimensione fissa.
  - Sostituiranno quelli che sono stati i processori progettati fino a quel momento (denominati a loro volta: CISC).

- Caratteristiche del CISC?
  - Le architetture CISC esprimevano molto lavoro in modo conciso (istruzioni più articolate espresse in una sola riga) che permettava:
    - La generazione di eseguibili più corti e quindi poco spazio in memoria

        - Dato che i tempi di accesso alle memorie erano alti, aveva senso esprimere grandi quantità di lavoro in poco spazio così da accedere meno alla memoria.
        - L'uso della microprogrammazione contribuiva a generare instruction set sempre più complessi e articolati:
            - Il microprogramma era memorizzato su ROM e se fosse convenuto sarebbe bastato aggiungere un nuovo comando alla ROM arricchendo l'instruction set.
            - Si poteva addirittura cambiare l'intera ROM e quindi l'intero instruction set.
    - Tutta questa versatilità era alla fine una lama a doppio taglio:
        - Avendo istruzioni di lunghezza variabile, alcune istruzioni potevano necessitare di 1 come 10 byte o come 20! Questo per non sprecare inutilmente spazio.
        - Operandi messi a caso nell'istruzione: potevano trovarsi in posizioni diverse a seconda dell'istruzione
    - Inoltre, la parte di memoria in cui risiedevano le istruzioni da processare, veniva letta a blocchi, ed essendo istruzioni di lunghezza variabile, quando veniva letto un blocco non si sapeva ancora se si fosse letta una istruzione completa o meno, oppure se ne conteneva addirittura due.
    - Una volta individuata una istruzione, non sapendo dove si trovano gli operandi, bisognava individuare i bit che corrispondevano quest'ultimo.
    - Istruzioni più lunghe e complesse = datapath più complesso ed articolato

- In cosa è meglio di RISC?
    - Nel paradigma architetturale RISC le istruzioni:
        - Sono in numero ristretto
        - Hanno tutte la stessa lunghezza (quindi anche l'accesso alla memoria delle istruzioni prelevava esattamente una istruzione)
        - Struttura regolare (operandi nella stessa posizione)
        - Solo due metodi di indirizzamento della memoria: LOAD e STORE
    - Favoriva inoltre, grazie alle istruzioni strutturare questo modo, la pipeline del datapath.
    - Per contro hanno che:
        - Utilizzano più registri
        - Istruzioni che esprimono meno lavoro, necessitano di più linee per essere descritte, quindi eseguibili più lunghi e quindi necessità di avere spazio in memoria più ampio (più RAM):
            - Questo non era un problema in quanto in quel periodo la costruzione di RAM a semiconduttore aveva iniziato la sua scalata, avendo dimensione sempre più capiente.

Linguaggi di programmazione

alla fine degli anni ‘40, erano in funzione sicuramente meno di 20 computer, il concetto di programmazione come lo intendiamo noi era totalmente nuovo, e si potevano contare poche decine di programmatori. Ovviamente non esistevano corsi o tutorial di programmazione, e si imparava a programmare semplicemente seguendo il lavoro di qualcuno che aveva imparato prima oppure, ancora più semplicemente, provando per conto proprio. Pochissimi programmi erano stati scritti, e naturalmente erano tutti in linguaggio macchina, dato che assembler e compilatori non erano ancora stati inventati (e neppure era chiaro che ci fosse bisogno di qualcosa come un linguaggio ad alto livello).

- **Initial Order**
  - Con l’avvento di un nuovo computer chiamato EDSAC, nel 1949, venne introdotta una **nuova forma mnemonica** per le istruzioni macchina chiamate Initial Orders, facilmente comprensibili dall’essere umano:
    - In sostanza il codice operativo delle istruzioni macchina era associato ad una lettera dell’alfabeto
  - Rappresentava una forma primitiva di assembler

- **Lo Short Code**
  - Concepito da Mauchly per il BINAC (computer), in contemporanea agli Initial Orders, è considerato il primo linguaggio di programmazione ad alto livello (Non girò mai sul BINAC ma poi verrà usato sull’UNIVAC 1).
  - L’idea dello Short Code era quello di **scrivere equazioni matematiche attraverso simboli e non più solo numeri**:

        - Ogni numero di un operando era riscritto attraverso un sumbolo
            - Ad esempio, l'equazione: X = Y + (X * Z)

              diventava: 40 03 41 07 09 40 42 02
    - Con lo short code emergeva l'idea di codificare/immagazzinare nella memoria, non solo numeri ma anche simboli e caratteri alfanumerici attraverso codici numerici, poi successivamente tradotti ed interpretati.

- **Il compilatore di Corrado Bohm**
    - Nel 1950 Corrado Bohm, italiano, definisce su carta un linguaggio ad alto livello, un linguaggio macchina ed un metodo di traduzione tra il primo ed il secondo.
    - Gli aspetti innovativi di questo lavoro furono i seguenti:
        - L'uso dei concetti GOTO e IF-THEN-ELSE
        - L'uso dello statement di assegnamento in tutte le istruzioni
        - La possibilità di usare subroutine/funzioni
        - Possibilità di scrivere un compilatore scritto nello stesso linguaggio dei programmi che deve compilare
    - L'intero compilatore era composto di sole 130 istruzioni per l'analisi sintattica e la generazione del codice macchina.
    - Inoltre, il compilatore genera codice in numero di passi proporzionale alla lunghezza del programma, rispetto ad altri linguaggi che ne richiedevano il doppio.

- **Il Fortran**
    - Fa la sua comparsa nel 1957, il linguaggio più famoso di quegli anni ed anche il primo che sia sopravvissuto fino ai giorni nostri (ovviamente con diversi update).
    - Ha origine nel 1954 da un gruppo dell'IBM composto da Backus, Herrick e Ziller. I tre si riuniscono per creare un sistema "per la programmazione automatica" e si recano all'MIT per studiare un sistema progettato da Laning e Zierler
    - Backus osservò come, in quegli anni, si pensava fosse impossibile che un compilatore avrebbe potuto generare codice efficiente e versatile. **Tutti penavano che un compilatore potesse produrre solo codice macchina molto meno efficiente di quello scritto da un programmatore umano**.
    - Anche Backus e il suo gruppo temevano che potesse succedere tutto ciò al loro sistema e che nessuno avrebbe voluto utilizzare un sistema così inefficiente
    - In ogni caso, alla fine del 1954 il gruppo di Backus completò le specifiche del FORTRAN in cui osservavano che
        - Il fortran sarebbe stato capace di scrivere programmi facilmente comprensibili e, una volta compilato, efficienti da eseguire rispetto agli altri sistemi in cui era presente facilità di sviluppo del codice ma inefficienza (o viceversa).
        - Non ritenevano fondamentale che il FORTRAN fosse svincolato dall'hardware ma invece **che avesse una forma NON assembler-like**, **e quindi ad alto livello**.
        - Doveva essere facile da apprendere, a detta loro era necessario un ora di tutorial per apprenderlo.
    - Venne reso disponibile nell'aprile del 1957 dopo più di due anni di annunci e ritardi vari. Molti programmatori erano conviti del fatto che non avrebbe mai

funzionato, ma circa un anno dopo il rilascio, metà dei computer IBM avevano al loro interno programmi scritti in FORTRAN
    - Ovviamente era presente di Bug e imperfezioni che verranno fixate con il tempo (**Ad esempio la presenza del GOTO**, rimosso poi con il FORTRAN 77).
    - **Quindi il FORTRAN è ricordato per essere il primo linguaggio ad alto livello in grado di generare codice altamente efficiente e avente una struttura del compilatore efficacie tanto da influenzare lo sviluppo dei linguaggi successivi**

- **Il LISP**
    - Dopo il FORTRAN, è il linguaggio più vecchio ancora in uso
    - Concepito tra il 1956 e il 1858 da John McCarthy per scrivere programmi in formalismo fortemente matematico (quello che chiameremo linguaggio di programmazione funzionale) per rappresentare e manipolare espressioni simboliche.
    - Molte furono le innovazioni:
        - **Primo linguaggio funzionale**
        - Aggiunta della ricorsione
        - Uso di strutture dati come Alberi e Liste
        - Memoria dinamica e garbage collector
        - Composizione annidata di funzioni
    - Fu sviluppato inizialmente per un computer dell'IBM (IBM 704) ed era interpretato (il compilatore fu sviluppato nel 1962)
    - Alcune caratteristiche del LISP lo rendevano estremamente efficiente:
        - Notazione prefissa per le espressioni
        - la possibilità di implementare direttamente nel codice macchina dell'IBM 704 alcuni degli operatori principali del LISP (CAR e CDR in particolare)
    - **Da ricordare che nel LISP tutto è rappresentato attraverso liste concatenate**
    - Il LISP fu un successo soprattutto negli ambienti accademici e nel campo dell'intelligenza artificiale fino all'avvento del PROLOG.

- **IL COBOL**
    - Fa il suo esordio nel 1959 e **concepito esplicitamente per applicazioni in campo amministrativo e commerciale**. Sviluppato dal consorzio: CODASYL (formato da rappresentanti dell'industria e governo USA).
    - Il dipartimento della difesa americano obbligò le compagnie produttrici di computer ad installarlo nelle nuove macchine così da favorirne la diffusione
    - Sintassi fortemente influenzata dal FLOW-MATIC e dal COMTRAN della IBM (altri due linguaggi).
    - Uno degli obbiettivi del COBOL era quello di essere portabile, efficiente e facilmente comprensibile da programmatori esperti.
    - Ebbe una forte diffusione negli anni 60 e negli anni 70 divenne il linguaggio più usato al mondo
    - Tuttavia, il mondo degli informatici ha sempre guardato il COBOL (ed i loro programmatori) con aria di dipresso e scetticismo

- **L'ALGOL 60**
  - Inizialmente chiamato ALGOL 58, ma quest'ultimo ebbe poche implementazioni e quindi dimenticato.
  - Divenne negli anni 60 il lignaggio di programmazione di riferimento nel monto scientifico per la descrizione di algoritmi (**Ma meno per la loro effettiva implementazione, per cui si preferiva ancora il FORTRAN**).
  - Non era particolarmente adatto per applicazioni di tipo commerciale
  - Le innovazioni che portò l'ALGOL furono le seguenti:
    - Call by value
    - Call by name (Non sopravvissuta nel tempo, sostituito dal reference)
    - Call by reference
    - Primo linguaggio a richiedere l'espressiva dichiarazione dei tipi delle variabili (INTEGER, BOOLEAN, DOUBLE, …)
    - Compare per la prima volta la distinzione tra assegnazione e confronto rispettivamente -> a = b, a := b
    - Compaiono per la prima volta primitive per la programmazione strutturata (Anche se il GOTO rimane):
      - If-then-else
      - While
      - For
    - Introdotti blocchi di istruzioni delimitati da "Begin" e "End", con cui costruire funzioni anche annidate e variabili locali.

- **Il BASIC**
  - Concepito nel 1964 da Kemeny e Kurtz che affidarono lo sviluppo ad un gruppo di studenti della Università di Dartmouth e reso disponibile nel primo maggio 1964. Inizialmente pensato per scrivere algoritmi matematici in modo da rendere facile l'implementazione da parte degli studenti di matematica e scienze affini.
  - Dato che in quegli anni la scrittura di programmi era considerata una pratica solo per pochi esperti del settore, l'obbiettivo del BASIC era quello di permettere a studenti di altre discipline o inesperti del settore di scrivere programmi relativamente complessi senza aver bisogno di avere troppe conoscenze specifiche nel campo dell'informatica
  - Kemeny e Kurtz furono molto bravi nel promuovere il BASIC e lo resero disponibile gratuitamente e ne incoraggiarono l'uso nelle scuole della loro provincia. La loro campagna ebbe successo tanto che dalla seconda metà degli anni 70, qualsiasi PC forniva un compilatore BASIC.
  - La fortuna del BASIC fu che era così leggero da poter girare su computer di potenza limitata ed allo stesso tempo sufficientemente ad alto livello e semplice tanto da poter essere usato da chiunque: contribuì in modo determinante nella diffusione tra il grande pubblico di una cultura informatica
  - Sebbene contribuì nella diffusione della cultura informatica, **c'è da chiedersi se fosse una vera cultura**:
    - **Il BASIC favoriva una programmazione non strutturata**
    - A questo proposito, Dijkstra scrive un articolo in cui critica il BASIC ed il COBOL, in quanto asserisce che linguaggi come il BASIC mutilasse la mente delle ed oltre ogni speranza di cura, mentre l'uso del COBOL paralizza la mente ed il suo insegnamento dovrebbe essere considerato un reato.

- **Cos'è lo Spaghetti Code**
  - Era il nome con cui veniva denominato, in modo allegorico, il flusso computazionale di un programma scritto con codice non strutturato:
    - Linguaggi come l'ALGOL, che favorivano la strutturata ma comunque contenevano il GOTO, permettevano comunque una programmazione non strutturata

- **Teorema di Bohm-Jacopini**
  - Questo teorema, dimostrato e formulato nel 1966 dagli italiani Corrado Bohm e Giuseppe Jacopini, asserisce che qualsiasi funzione computabile può essere calcolata da un programma che utilizza solamente le seguenti istruzioni:
    - Selezione (if then else)
    - Iterazione (while, for)
  - Il teorema mostra che è possibile **scrivere programmi senza l'uso del GOTO** e quindi saltare dentro e fuori in modo disordinato da una parte all'altra del codice.
  - Da questo teorema nasceranno diversi dibattiti tra i sostenitori della programmazione strutturata e quelli invece contro.
  - Il nostro amico Dijkstra definisce il GOTO addirittura nocivo e dovrebbe essere abolito dai programmi in quanto ne complica l'analisi e il debugging.

- **Il Simula**
  - Tecnicamente derivante dall'ALGOL e sviluppato tra il 1962 e il 1965 da Dahl e Nygaard, è stato pensato specificamente per compiere simulazioni (da cui il nome). Linguaggio a cui **spetta il titolo di primo linguaggio OOP (Object Oriented Programming).**
  - Per concepire l'idea di Classe ed oggetto, si ispirarono al concetto di record class inglobato nell'ALGOL: nella sua proposta, Hoare usava l'espressione di "record class" per indicare un generico concetto che si vuole modellare (la Classe) mentre i record sono le istanze della classe (gli oggetti)
  - Nel 1966 Dahl e Nygaard riescono a definire un modello di classi e superclassi con tanto di ereditarietà.
  - Con il Simula quindi vengono introdotti:
    - Paradigma OOP
    - Concetti di classe e sottoclasse
    - Data abstraction
    - Ereditarietà

- **Pascal**
  - Dominò la scena dei linguaggi negli anni 70, presentato ufficialmente nel 1971 da Wirth con l'ambizione di: essere efficiente da compilare ed eseguire, permettere la scrittura di programmi ben strutturati ed organizzati. Inoltre, era obbiettivo del Pascal, diventare uno strumento di insegnamento per la programmazione (obbiettivo poi ampiamente raggiunto).
  - Il progetto del Pascal nasce dal tentativo di correggere alcune imperfezioni dell'ALGOL, in quanto assente del tipo CHAR e dei puntatori. Inoltre, era indipendente dalla piattaforma e quindi lo rendeva difficile da implementare alle diverse piattaforme hardware (scrivere il suo compilatore non era qualcosa di così banale e i suoi programmi erano spesso inefficienti)

- Data l'assenza di varie implementazioni, il FORTRAN rimaneva il linguaggio preferito dagli ingegneri. L'ALGOL rimaneva solo un linguaggio per descrivere algoritmi.
- Il Pascal incoraggiava la programmazione strutturata e modulare con tutte le caratteristiche ereditate dall'ALGOL e in più aveva:
  - Sistema di tipi molto ricco e rigido
  - Implementazione di nuovi tipi
  - Meccanismo di type checking statico
  - Sintassi elegante e facilmente leggibile
- **Il vero successo del Pascal fu l'introduzione di uno strato intermedio tra compilatore e codice macchina, il P-Code**:
  - Un programma Pascal non veniva tradotto immediatamente in codice macchina ma in istruzioni P-Code
  - Successivamente il P-Code veniva interpretato da un altro modulo che lo mandava in esecuzione (come il ByteCode di Java).
  - In questo modo il Pascal veniva fatto girare su piattaforme diverse
- Verso la fine del 70 era ormai insegnato in tutte le università.

- **Il PROLOG**
  - Come il LISP, non ha avuto molto successo. Il suo uso è stato confinato a poche università e centri di ricerca.
  - Esordisce nel 1972 ed ha un posto nella storia dei linguaggi come iniziatore del paradigma di programmazione logica e alla base del progetto FGCS
  - Nasce come dimostratore automatico di teoremi, usando come inferenza la regola di risoluzione
  - Una cosa particolare quindi del PROLOG e che funziona come dimostratore di teoremi per verificare il valore di verità di una proposizione, e come linguaggio per calcolare valori (?).
  - Nel PROLOG, tutti i programmi sono ricorsivi, il che porta ad un codice molto elegante e conciso.

- **Fifth Generation Computer System (FGCS)**
  - Progetto ideato dal governo giapponese nel 1981 che consisteva nel progettare un supercomputer (usando il PROLOG come linguaggio) che sfruttava il parallelismo, il linguaggio naturale e l'AI.
  - Il progetto doveva segnare un profondo salto rispetto alle tecnologie precedenti (anche ai microprocessori).
  - Le macchina FGCS avrebbero dovuto eseguire operazioni logiche e quindi sarebbero state valutate in LIPS anziché FLOPS:
    - **L'obbiettivo era di arrivare a raggiungere il GLIPS quando le macchine dell'epoca arrivavano al massimo al KLIPS**
  - Il progetto ebbe risonanza in tutto il mondo tanto da favorire progetti anche in USA ed in Europa, ma dopo dieci anni di investimenti il progetto si rivelò un fallimento per varie ragioni:
    - Negli anni 80 stava arrivando il passaggio a RISC che rendeva i microprocessori molto più veloci e superiori ai computer FGCS
    - **Lo sfruttamento del parallelismo come lo avevano ideato, per quei tempi era quasi impensabile da riprodurre**.

- **Lo Smalltalk ed il Dynabook**

- Lo smalltalk era un linguaggio progettato inizialmente solo per diventare interfaccia del sistema operativo di un dispositivo chiamato Dynabook:
  - Progettato alla Xerox Parc da Alan Kay nel 1972, **il Dynabook era un dispositivo pensavo per favorire l'apprendimento nei bambini**.
  - Era un vero e proprio tablet dei nostri tempi, in grado di stare in una tasca e di usarlo per tutta una serie di scopi personali
  - Un dispositivo del genere, a quei tempi, era rivoluzionario e difficile da immaginare viste anche le dimensioni dei PC dell'epoca
- Lo sviluppo dello Smalltalk proseguì in maniera indipendente dal Dynabook, passando attraverso varie versioni successive.
- Lo Smalltalk fu il primo ambiente di programmazione ad interfaccia grafica (**il progetto a cui Apple si ispirò per progettare l'Apple LISA e poi il Machintosh 128k**).
- Il linguaggio fu sviluppato da zero in modo completamente indipendente con sintassi e terminologie originali
- Le caratteristiche dello Smalltalk sono:
  - Programmazione rigorosamente ad oggetti, tutti i programmi sono costituiti solo da classi ed oggetti
  - Gli oggetti hanno dati privati e comunicano con l'esterno attraverso metodi
  - Sistema di tipi dinamico e flessibile
- Tutta la terminologia che abbiamo oggi sulla OOP si deve soprattutto allo Smalltalk piuttosto che al Simula, in cui i concetti della OOP erano ancora imprecisi e parzialmente nascosti.

- **ADA**
  - Prende il suo posto nella storia come il più faraonico dei progetti per dare alla luce un nuovo linguaggio di programmazione
  - L'idea parte dal dipartimento della difesa americano nel 1974, il quale si rende conto che i progetti in corso usano più di 450 linguaggi diversi:
    - I vertici dell'esercito decisero di avviare lo sviluppo di un linguaggio specifico per i sistemi embedded
  - Dopo circa 3 anni, il gruppo di lavoro aveva terminato una specifica completa del linguaggio che, resa pubblica, ricevette più di 500 report da tutto il mondo con commenti e suggerimenti per alcuni miglioramenti.
  - Il nome ADA venne dato in onore di Ada Lovelace. Il manuale venne approvato il 10 dicembre del 1980 (data di nascita di Ada) e il nome in codice è MIL-STD-1815 (Anno di nascita di Ada).
  - In molti pensavano che ADA fosse destinato a divenire il linguaggio di riferimento (si prevedeva che nel giro di dieci anni sarebbero sopravvissuti solo l'ADA ed il LISP)
  - **Invece si rivelò tutt'altro**:
    - Tony Hoare nel discorso al Turing Award del 1980 critica l'ADA per essere troppo complesso e inaffidabile
    - In effetti i primi compilatori ADA erano lenti e producevano codice inefficiente
  - **Il fatto che l'ADA prendesse le sue caratteristiche da altri linguaggi come l'ALGOL, il PASCAL ed il SIMULA e che introdusse altri costrutti articolati, non lo ha mai portato alla diffusione** ed alla popolarità che ci si sarebbe aspettati.

- **Il linguaggio C**
  - Il C è da considerare il linguaggio di programmazione più di successo della storia dell'informatica, considerandone la longevità e la diffusione.
  - Molti sono i linguaggi che si sono ispirati al C o diretti discendenti, come il C++, C#, l'Objective C…
  - La sua storia incomincia ai laboratori Bell quando si stava sviluppando il MULTICS OS. Tra i ricercatori del progetto c'erano Ritchie, Thompson e Kernighan.
  - Svilupparono il MULTICS rinominandolo per scherzo Unix.
  - Unix era scritto nell'assembler del PDP-7, ma successivamente utilizzarono un altro linguaggio ad alto livello che avevano chiamato "B" ed il relativo interprete (Il B derivava a sua volta dal BCPL che venne rinominato "Before C Programming Language)
  - Dal BCPL si deve la comparsa delle parentesi per la delimitazione di blocchi di codice:
    - Thompson trasformo il BCPL in B rendendolo sintatticamente più conciso ed introdusse alcune notazioni familiari come: x++, x += y, ==
  - Il B inizialmente era non tipato, ma dopo la modifica di Ritchie del 1972 (che introdusse il sistema di tipi scrivendo un nuovo compilatore) il linguaggio venne rinominato NB e dopo semplicemente C.
  - Il successo del C era dovuto soprattutto alle sue qualità:
    - Portabilità: un programma C gira facilmente su qualsiasi piattaforma
    - Modularità: grandi progetti posso essere spezzati in più moduli
    - Efficienza: i programmi C girano più veloce dei programmi scritti in qualsiasi altro linguaggio ad alto livello. Ciò rende il C perfetto per la programmazione di sistemi operativi, sistemi embedded e per scrivere compilatori stessi.
    - Compattezza: Possibilità di scrivere programmi molto sintetici (ma altrettanto oscuro e incomprensibili, sebbene funzionante).
    - Tipi flessibili: al contrario del Pascal il C permette di combinare fra loro variabili di tipo diverso
    - Gestione efficiente della memoria: con uso estensivo dei pointers
    - Bitwise programming
  - Il C diede vita ad altri linguaggi, come le sue estensioni Object Oriented:
    - Il C++
    - Objective C
    - C-Sharp

- **Java e l'era di Internet**
  - Java fu concepito alla Sun sotto la guida di G osling, nel 1990, con lo scopo di creare applicazioni complesse per piccoli dispositivi elettronici (l'idea centrale era quella di sviluppare software interattivo per le TV).
  - C'era dunque bisogno di un linguaggio con il quale era possibile sviluppare applicazioni che girassero su processori poco potenti:
    - Il linguaggio più in voga in quel periodo era il C++ che però non era adatto per CPU di basse prestazioni.
  - Gosling pensò di modificare il C++ per le esigenze del progetto (inizialmente chiamato C++ ++ --, nome di merda per indicare che erano state tolte alcune cose ed aggiunte altre. Successivamente ridenominato Oak).

- Un evento epocale (nel 1993) rischiò di mandare all'aria tutto il progetto: **l'avvento di Internet e del WWW**:
  - Con l'avvento di Internet, l'idea di un TV interattiva diventava quasi obsoleta.
- Alla Sun furono bravi nel convertire il progetto in un'altra idea di successo:
  - Con l'avvento del WWW, iniziarono a comparire i primi Browser Web (Il primo al CERN di Ginevra)
  - La Sun ebbe quindi questa idea: Applicazioni scaricate da Internet insieme all'HTML ed eseguite localmente sul browser in modo da aggiungere capacità interattive alle pagine Web: **Applet Java**.
- Il linguaggio delle Applet doveva essere cross-browser e indipendente dall'hardware
- La sintassi del C++ venne conservata ed il nome venne cambiato in JAVA (dall'omonima varietà di caffè indonesiano)
- Java fu rilasciato finalmente nel 1995, dopo grandi sforzi di marketing e con grandi aspettative da tutto il mondo informatico (non c'era tanto Hype dai tempi del FORTRAN).
- Visto il successo, tutti i principali Browser vennero aggiornati incorporando la JVM per l'esecuzione delle Applet Java.
- Java si estese poi oltre le semplici Applet, grazie alla sintassi simile al C++, diventando sempre più semplice ed affidabile.
- Uno dei successi del Java fu la sua portabilità dovuta al bytecode (Idea già usata dal Pascal con il P-Code).

- **Linguaggi di Scripting**
  - Un linguaggio di scripting è un linguaggio per scrivere script: uno script è un programma relativamente corto eseguito da un interprete.
  - A differenza di un linguaggio di programmazione tradizionale (che viene compilato ed eseguito) un linguaggio di scripting è ospitato da un ambiente software all'interno del quale lo script può essere scritto e fatto girare quasi istantaneamente:
    - Una qualsiasi Shell Unix è un ambiente software in cui scrivere script ed eseguirlo.
    - Anche un Browser Web è un ambiente in cui è possibile scrivere ed eseguire script

- **Linguaggi di scripting General Purpose**
  - È un linguaggio che mette insieme le caratteristiche dei classici linguaggi di programmazione e le caratteristiche dei veri e propri linguaggi di scripting, come:
    - Python
    - Perl
    - PHP
    - Javascript
    - Ruby

Sistemi Operativi

La storia dei sistemi operativi è suddivisa in otto fasi.

- **Prima fase: Job by Job**
  - L'idea di farsi aiutare da un programma per farne girare altri, prima degli anni 50 era qualcosa di completamente impensabile.
  - Per far girare un programma sui computer primitivi nei primi anni 50, i passaggi erano più o meno i seguenti:
    - Scrivere su carta in codice binario (con indirizzi assoluti)
    - Codificare quanto scritto sopra su schede perforate
    - Il programmatore si recava al PC prenotato (durata di 15/30 minuti), sperando che non fosse guasto
    - Buona parte dello slot prenotato era perso a preparare il computer e sistemare le schede nel lettore che trasferiva tutto alla memoria
    - Il programmatore poi poteva sedersi e comandare l'esecuzione del programma
    - L'output veniva stampato su una telescrivente oppure riversato su nastro magnetico e convertito in schede perforate
  - Molto tempo veniva passato per la gestione manuale delle schede e nastri.
  - Inoltre, nella meta del 1950, con l'avvento dei primi assembler e dei primi linguaggi ad alto livello tutto venne reso più complesso: bisognava caricare in memoria anche l'assembler e il compilatore
  - Per gestire tutto quanto in maniera efficiente **venne introdotta una figura professionale addetto alla sala macchine**, a cui il programmatore affidata il pacchetto di schede e l'addetto si occupava di mandarlo in esecuzione e restituire al programmatore il risultato.
  - In questa fase mancano quindi ancora:
    - Interfaccia tra programma e macchina
    - File system e concetto di file
    - Singolo programma in memoria e singolo programma mandato in esecuzione

- Seconda Fase: **Sistemi Batch**
  - L'introduzione dell'addetto alla sala macchine suggerì l'idea di creare un programma che sostanzialmente facesse il suo lavoro: prendere pacchetti di schede e organizzare la loro esecuzione. Questo programma, chiamato **Resident Monitor**, lavorava attraverso **schede perforate di controllo**:

        - I pacchetti di schede in questa fase venivano inseriti tutti insieme, separati da schede di controllo che indicavano l'inizio o la fine di un pacchetto di schede e cosa fare con quel pacchetto di schede.
        - Le istruzioni di controllo scritte su quelle schede speciali sarebbero state eseguite proprio dal Resident Monitor
    - Naturalmente il Resident Monitor doveva riuscire ad interpretare queste istruzioni di controllo, gestire l'I/O, far partire un programma e al termine dell'esecuzione riprendere il controllo della macchina.
    - **Rispetto alla fase 1** quindi:
        - Alla macchina veniva fornito un insieme di pacchetti di schede (non uno alla volta), separati da schede di controllo.
        - **Venivano caricati quindi più programmi in memoria** e compariva la distinzione tra codice e dati
        - Il Resident Monitor permetteva un'automazione dell'esecuzione di un gruppo di programmi.
        - i dispositivi di input/output sono gli stessi rispetto alla fase 1 (lettori di schede perforate e lettori di nastri magnetici), **ma ora possiamo inserire nella memoria computer più programmi da eseguire** (ci penserà poi il resident monitor a lanciarli uno dopo l'altro), e questo fa risparmiare tempo. Infatti, mentre questi programmi girano, possiamo caricare su un altro nastro un successivo gruppo (batch, appunto) di programmi, che verranno gestiti allo stesso modo una volta finita l'esecuzione del gruppo precedente. Questo è più efficiente che caricare in memoria un programma alla volta, eseguirlo, e poi ripetere il tutto col programma successivo (Cit. Gunetti).
    - Tuttavia, i processori di quei tempi non fornivano una protezione hardware adeguata e il sistema era esente da protezione della memoria, quindi se un programma girava male bisognava spesso fare un reboot.
    - Il lavoro che eseguiva il Resident Monitor venne chiamato **Batch Processing**.
    - L'insieme di programmi che venivano fatti girare prendevano il nome di **Batch Jobs**
    - Le macchine su cui girava un Resident Monitor erano chiamate **Batch System**, ad esempio il Philco 2000:
        - messo in commercio nel 1957 e anche uno die primi completamente a transistor su cui girava il sistema BKS (sistema batch)

- Fase 3: **Multiprogrammazione**
    - In questa fase furono di enorme importanza i dispositivi di memorizzazione di massa ad accesso diretto, in quanto **l'efficienza dei sistemi Batch era limitata dall'uso di dispositivi di massa** ad accesso sequenziale e molto lenti. Vennero introdotte le memorie a tamburo rotante e successivamente gli Hard Disk.
    - Il termine Multiprogrammazione compare per la prima volta in un articolo di Strachey del 1959 il quale osserva che:
        - Con l'utilizzo di un Hard Disk era possibile leggere contemporaneamente due programmi in simultanea dalla memoria rispetto ai nastri magnetici in quanto la testina di lettura/scrittura poteva spostarsi avanti e indietro in aree diverse del disco portando avanti in maniera quasi parallela diverse operazioni di lettura/scrittura.
    - **Rispetto alla fase 2**, quindi, l'Hard Disk veniva usato come Buffer al posto dei nastri:

        - Dalle schede perforate veniva letto il programma ed i dati, che venivano riversati sul disco.
        - Una volta letto tutto, veniva fatto girare il programma
        - Una volta finito il programma, l'output memorizzato sul disco veniva stampato
    - Inoltre, con l'introduzione degli interrupt, l'I/O poteva avvenire in maniera contemporanea: non era necessario che venisse prima letto tutto e poi stampato tutto.
    - Questo modo di operare degli Hard Disk (gestione dei programmi e dell'I/O), venne rinominato **Spooling** (Simultaneous Peripheral Operation On Line).
    - Il termine Multiprogrammazione fu coniato per intendere qualcosa di diverso da quello che intendiamo oggi: si intendeva la possibilità di eseguire in maniera asincrona diverse operazioni di I/O. Successivamente il concetto fu generalizzato a programmi che si fermavano temporaneamente per eseguire I/O e che lasciavano la CPU libera permettendo l'esecuzione di un altro programma (modello di esecuzione che è noto come Multitasking o più propriamente Multitastking preemptive).
    - Primo Computer Multitasking: Atlas Computer
        - Presentato nel 1961 con sistema operativo "Atlas".
        - Sviluppato da Kilburn, Payne e Howard è riconosciuto come il primo Computer multitasking e per molti il più importante passo avanti nella storia dei sistemi operativi
        - Compaiono per la prima volta le System Call (inizialmente extracode) e memoria virtuale con paginazione su richiesta

- Fase 4: **Timesharing**
    - Idea che nasce più o meno in contemporanea alla Multiprogrammazione in un report del 1959 di McCarthy in cui propone un'idea per ridurre il tempo richiesto per risolvere un problema: il time-sharing. Il computer dovrà occuparsi di altri utenti mentre uno degli altri utenti sta interagendo con qualche output del computer.
    - Poco tempo dopo McCarthy chiarisce la sua idea, ed è proprio qui che nasce l'idea di Time-Sharing: lui intendeva un sistema che interagisce simultaneamente con più utenti attraverso computer remoti. **Tale sistema sembrerà agli utenti come se stesse utilizzando un computer privato** (il sistema operativo è reso trasparente agli utenti). Inoltre, i programmi che si fermavano per operazioni di I/O non venissero spostati completamente in memoria di massa ma che rimanessero in memoria centrale: questo richiedeva di avere sufficientemente memoria da memorizzare un numero abbondante di programmi. Un ultimo requisito era che la memoria secondaria fosse sufficientemente grande da contenere i files degli utenti così che gli utenti non avessero bisogno di usare schede perforate o sistemi esterni di I/O.
    - CTSS
        - Primo sistema operativo time-sharing sviluppato nel 1961 all'MIT
        - L 'obbiettivo del CTSS era quello di permettere a più persone di utilizzare il computer, attraverso un supervisor che alternava l'esecuzione dei vari programmi, assegnando un breve burst di CPU a ciascuno di essi, garantendo ad ogni utente di ricevere risposte in tempi ragionevoli.

        - Una delle novità fu la possibilità di interagire con il sistema specialmente durante la fase di debugging
        - I progettisti del CTSS si trovarono davanti a problemi che non erano mai stati affrontati prima:
            - Per permettere l'interattività, i programmi dovevano essere tutti tenuti in memoria, tenendo traccia delle diverse aree dei programmi per garantirne la protezione
            - Codice rilocabile dinamicamente
            - Al termine del quanto di tempo e alla ripresa dell'esecuzione di un programma, doveva ripartire nel punto in cui era rimasto
            - Operazioni di I/O sotto il controllo del sistema
            - ….
        - Inizialmente il CTSS era in grado di gestire 3 utenti, successivamente 32
        - Oltre al Time-sharing, il CTSS introdusse altre idee che sono scontate ai giorni nostri:
            - Accesso controllato dagli utenti, con login e password
            - Gli utenti potevano creare nuovi comandi
            - Mail: Gli utenti del CTSS comunicavano tra di loro attraverso file lasciati sul sistema
    - **Il MULTICS**
        - Il CTSS diede vita al progetto MULTICS, il sistema più controverso ed ambizioso degli anni 60
        - Partito nel 1964 all'MIT (progetto diretto da Robert Fano) con l'obbiettivo di progettare un sistema capace di soddisfare tutte le necessità presenti degli utenti.
        - Al progetto parteciparono anche la General Electric e Bell che si ritirarono nel 1969.
        - Il Multics non ebbe un grande successo commerciale, probabilmente perché era un sistema decisamente esagerato (progettato per gestire contemporaneamente centinaia di utenti su processori scuffed).
        - Inoltre, sviluppato in PL/1: linguaggio progettato nel 1964 da IBM che era pesante ed inefficiente e non contribuì certamente al successo del MULTICS.
        - Il Multics non ebbe successo ma giocò comunque un ruolo importante nella storia dei sistemi operativi: Ispirò lo sviluppo dello UNIX
    - La nascita del FileSystem
        - Nel CTSS lo spazio per i file era organizzato in modo che ogni utente avesse i file su un nastro diverso, inoltre, era disponibile una directory comune per condividere file: soluzione poco efficiente dato che era a directory singola
        - Utilizzando i sistemi Time-Sharing nasce la necessità di organizzare meglio le informazioni:
            - Organizzare i file in maniera più strutturata anziché in una singola directory, mescolando tutto
            - Definire politiche di accesso ai file (chi li poteva usare e chi no).
            - File facilmente accessibili e garantire l'integrità di quest'ultimi.
        - Nel 1965, Robert Daley e Peter Neumann (partecipanti del progetto MULTICS) pubblicano un articolo che conteneva l'essenza del File

System (l'articolo era solo una proposta, l'implementazione fu portata a termine poco dopo):
    - L'articolo specifica che il file system dovrà essere indipendente da come viene implementato su una specifica macchina e che gli utenti dovranno interagire con i file solo attraverso nomi simbolici (il lavoro di mappatura deve essere invisibile)
    - I file devono essere organizzati in una struttura ad albero in cui i nodi sono le directory che possono contenere file o altre directory (nasce il concetto di pathname)
    - Il proprietario di una directory/file potrà decidere chi potra farci accesso e condividerlo con opportuni link
    - Gli utenti avranno a disposizione comandi per manipolare i file
  - Il primo Computer con file system fu il Titan (successore dell'Atlas), con tecnica di memorizzazione FAT con blocchi non contigui da 4Kbyte
- **L'OS/360**
  - Sistema operativo degli anni 60 noto non per gli aspetti innovativi ma perché dominò la scena a livello commerciale.
  - Progetto che nasce perché la IBM notò che suoi mainframe di fascia alta e bassa non erano compatibili tra di loro. Quindi nel 1964 lanciò questa serie di computer di potenza e costi diversi ma con periferiche e software compatibili.
  - Sebbene ormai noto, il time-sharing non fu adottato dall'IBM per questa gamma di Computer (era solo Multiprogrammato).
- UNIX
  - Uno degli eventi cruciali della storia dell'informatica, il sistema più influente della storia. Addirittura, nel 2000, Per Brinch Hansen osservò che: "un sistema superiore come lo Unix appare spesso così naturale da non invogliare i programmatori a cercarne uno migliore. Dopo trent'anni, si può ritenere che l'enorme diffusione dello Unix sia divenuto un ostacolo per ulteriori progressi."
  - Si osservò che Unix ebbe la fortuna di apparire nel momento giusto:
    - Con l'avvento dei primi minicomputer, gli utenti erano insoddisfatti dei sistemi operativi disponibili e quindi erano pronti per adottare Unix.
  - La maggior parte delle idee dello Unix non erano nuove ma, come osserva uno dei padri di Unix (Dennis Ritchie) "costituivano una buona ingegnerizzazione di idee in giro già da un po' in forme diverse, ed erano ora pronte per essere usate in modo conveniente."
  - Mentre lavoravano al MULTICS, Ken Thompson trovò un PDP-7 poco usato nei laboratori Bell e decise di svilupparci sopra un sistema più facile da usare del MULTICS. Raggiunto da Ritchie, in pochi mesi svilupparono un file system gerarchico, un interprete dei comandi, un editor ed un gestore dei processi.
  - La prima versione era Single User ma pensato fin da subito come sistema interattivo multiutente
  - Il successo della prima versione fece sì che Thompson e Ritchie potessero acquistate il PDP-11/20 su cui proseguire i loro sviluppi.
  - Nel 1970 Peter Neumann suggerisce di chiamare UNICS il nuovo sistema come gioco di parole con MULTICS. Infine, Kernighan propose Unix come nome definitivo.

        - **Nella versione del 1973 Unix venne scritto in C, linguaggio progettato dallo stesso Ritchie proprio per questo scopo**. La nuova versione è un terzo più lingua di quelle scritte in assembler, più facile da capire, modificare e installare su hardware diversi.
        - Ritchie e Thompson ricevettero le prime richieste per poter usare il sistema, ma alla AT&T era vietato entrare nel mercato dei computer e quindi Unix non poteva essere venduto come prodotto, ma poteva essere distribuito dietro licenza d'uso a chi ne facesse richiesta:
            - **Il sorgente veniva ceduto gratuitamente al solo costo del nastro e delle spese di spedizione**
        - L'AT&T, capito il potenziale economico del progetto, provò a vendere una sua versione (Unix V6) con licenza d'uso per attività non educational a circa 20k dollari, ma ormai molte versioni erano state sviluppate in giro per il mondo e altre in via di sviluppo.
        - **Per sistema Unix-Like si intende un sistema fortemente compatibile con le specifiche Unix ufficiali di Ritchie e Thompson**.
        - Nel 1974 Ritchie e Thompson pubblicano un articolo chiamato: **The Unix Time-Sharing System**. In questo articolo è contenuto tutto ciò che normalmente oggi intendiamo per sistema operativo (più di 40 anni dopo questo articolo è mantenuto aggiornato).

- Fase 5: **Sistemi Concorrenti**
    - Negli anni 60, i sistemi operativi erano diventati molto complessi ma presenti ancora di problemi assai rilevanti come il Thrashing, Deadlock e Starvation. Questo portò negli anni successivi a sviluppare le basi concettuali per rendere i sistemi comprensibili e affidabili, ponendo così le basi per una programmazione concorrente.
    - **Il primo a contribuire fu Dijkstra con il sistema THE**:
        - Sistema non Time-Sharing ma Multiprogrammato
        - **Introdusse il concetto di semaforo** per la sincronizzazione tra processi
    - Un altro che contribuì fu Per Brinch Hansen con il suo sistema multiprogrammato RC 400 (noto anche come Monitor). Primo esperimento di sviluppo di un Kernel attorno al quale poi sviluppare il sistema operativo completo in base alle esigenze.
    - Per Brinch Hansen è anche il padre del primo linguaggio di programmazione concorrente: il Concurrent Pascal del 1973:
        - Estensione del Pascal che conteneva le primitive per la sincronizzazione tra processi
        - La sua potenzialità fu dimostrata con lo sviluppo del sistema operativi SOLO.

- Fase 6: **Sistemi per personal computer**
    - Oggi si intende associare Windows e Mac OS come i sistemi operativi per eccellenza e molti li associano ai primi sistemi mai sviluppati, ma alla fine sono solo prodotti commerciali che hanno migliorato idee precedenti e aggiungendone di nuove (o migliorando quelle precedenti).
    - Se bisogna trovare un capostipite per i sistemi operativi per PC, questo titolo forse **spetta al sistema OS 6**, sviluppato tra il 1969 e il 1972 alla Oxford University:
        - Scritto in BCPL

        - Girava solo un programma alla volta
        - La sua importanza storica di sa soprattutto all'implementazione del File System ed ai meccanismi di I/O del programma in esecuzione
    - Un altro sistema sviluppato subito dopo fu Alto. Sviluppato alla Xerox tra il 1973 e il 1976, anche lui scritto in BCPL, rimasto famoso per la sua interfaccia grafica.
    - All'Alto seguirno il Pilot, il Cedar.
    - **Nel 1981, alla Xerox, sfruttando le idee dell'Alto, progettarolo lo Xerox Star**: il primo sistema dotato di mouse e interfaccia grafica ad icone e finestre. **L'idea era quella di lavorare all'idea di Ufficio Elettronico**: fornire un computer che sembrasse a chi lo usava un ufficio elettronico a tutti gli effetti e quindi creare un sistema virtuale di un ufficio fisico:
        - Infatti, i documenti ad esempio non erano solo rappresentati da un nome ma da un'immagine di un figlio sullo schermo.
        - Un cestino
        - etc..
    - Ai tempi innovazioni simili risultavano rivoluzionarie (si può considerare l'interfaccia grafica una delle innovazioni più importanti nella tecnologia dei Sistemi Operativi)
- Fase 6: **La storia di Microsoft**
    - Questo sarà l'esempio più eclatante del fatto che a volte il successo di una ricerca scientifica dipendono molto dal caso e dalle scelte personali.
    - Nei primi anni 70, **Gary Kildall**, giovane talento informatico, acquistò un intel 4004 per cui scrisse alcuni programmi e iniziò a collaborare come consulente per Intel.
    - Nel 1973 mette a punto un linguaggio ad alto livello (il PL/M) **e soprattutto il CP/M**: un semplice sistema operativo per controllare il drive di un floppy disk. Per commercializzare questo suo prodotto, insieme alla moglie, fonda la Digital Research (per intel non aveva riscosso molto successo).
    - Perché il CP/M ebbe una diffusione così rapida e di grande successo?
        - Il motivo è che Kildall fu l'ideatore, nel 1975, **del BIOS** (Basic Input Output System): è la logica che permette di interfacciare un sistema operativo con lo specifico hardware su cui sta girando (**prima del BIOS, un sistema doveva essere scritto pensando ad una ben precisa piattaforma hardware**)
    - Il CP/M alla fine era suddiviso in tre parti:
        - CPP
        - BDOS
        - BIOS
    - Portare il CP/M da una piattaforma hardware ad un'altra (ma stessa CPU) richiedeva solo la riscrittura del BIOS. Questo lo rese molto famoso e come conseguenza molti software furono scritti per girare su CP/M
    - **La sfortuna di Kildall arrivò con la fondazione di Micro-Soft** nel 1975 **e con l'arrivo dell'IBM nel mercato dei PC**:
        - Allen e Gates (fondatori) scrivono per il CP/M un compilatore BASIC, inoltre la Micro-Soft produsse una scheda di espansione che porto il CP/M a girare anche su Apple II
        - Viste le invenzioni, la IBM pensò che Gates fosse in possesso della licenza d'uso di CP/M chiedendogli i diritti. La Microsoft, non avendo i riditti, indirizzò la IBM verso Kildall. Tra la IBM e Kildall **non fu possibile**

stipulare un accordo (la storia è un po' confusa riguardo questo accordo)
        - I dirigenti IBM tornarono da Gates chiedendogli di trovare una soluzione:
            - Gates non si lasciò sfuggire l'occasione e comprò dalla Seattle Computer Products, la licenza d'uso dell'86-DOS, una variante del CP/M (per 8086).
            - Dopo aver avuto la licenza, ingaggia Tim Paterson (sviluppatore stesso di 86-DOS) chiedendogli di portare il sistema su processori 8088 (che saranno usati da IBM)
            - La nuova creazione venne chiamata MS-DOS. La licenza, finalmente, venne venduta a IBM che lo commercializza con i suoi PC con sigla PC-DOS.
        - Kildall entra in possesso di una copia del PC-DOS e nota che è praticamente una copia del suo sistema, minacciando l'IBM di fargli causa:
            - La IBM accetta di offrire i suoi PC con all'interno il CP/M invece che il PC-DOS ma ad un costo 6 volte superiore (ovviamente tutti finiranno per continuare ad usare il PC-DOS).
    - Nel 1985 Microsoft rende disponibile la prima versione di Windows (un'applicazione che poteva essere lanciata all'interno del DOS che trasformava lo schermo in un desktop "moderno"). Attraverso poi diverse versioni venne migliorato (nel 1993 la versione NT che lo porta ad essere multiprocessore e multiutente).
    - Nel 1995 viene lanciato Windows95, che segna la fine dell'era DOS.
    - Il DOS (disk operating system) era in sostanza un sistema operativo ridotto all'essenziale che permetteva di lanciare i programmi utente e la manipolazione di base del File System. Tutte le versioni DOS furono ispirate al CP/M (tra le più famose, la versione Altari-DOS e Commodore-DOS)
- Fase 6: **Mac OS**
    - Si tende ad associare Apple a Mac OS ed OS X, ma **il primo computer venduto da Apple (Apple II) montava un umile DOS**: l'Apple DOS, sviluppato inizialmente da Wozniak e alcuni suoi collaboratori
    - All'inizio degli anni 80 però la Apple era ormai lanciata verso lo sviluppo di interfacce grafiche, soprattutto dopo aver visitato la Xerox.
    - Il primo tentativo fatto con l'Apple Lisa fu un fallimento che presto verrà rimpiazzato con il celeberrimo Macintosh:
        - Dotato del sistema operativo System (che dalla versione 7.5 verrà chiamato Mac OS)
        - Costituiva una rivoluzione in quanto andava a sostituire completamente l'interfaccia a linea di comando con una a icone e finestre.
    - Con il passaggio alla versione 10, Mac OS venne rinominato in Mac OS X. Questo non fu un semplice cambio di nome ma un cambiamento radicale, portando i sistemi Apple ad essere Unix-Like. In particolare, derivanti da NEXTSTEP: sistema sviluppato da Jobs durante il suo esilio dalla Apple.
    - A partire dalla versione 10.8 venne abbandonato il prefisso "Mac".

- Fase 6: **GNU, Minix e Linux**
    - Secondo la Storia, Linux sembra nascere quasi per caso e non sarebbe esistito se prima del suo sviluppo non fossero successe alcune altre cose importanti.

- **Nel 1983 Richard Stallman, al MIT, da vita al progetto GNU**: progetto collaborativo di massa per lo sviluppo di software libero.
- Per far in modo che tutto il software di un sistema sia libero, deve esserlo anche il suo sistema operativo:
  - Stallman allora inizia a lavorare su un sistema operativo free chiamandolo appunto GNU (GNU is Not Unix)
- Lo sviluppo dello GNU però andò molto a rilento tanto che agli inizi degli anni 90 ancora un Kernel GNU non era disponibile:
  - Intanto nel 1985 intel presenta la prima CPU a 32 bit (l '80386)
  - **Nel 1986 Maurice Bach pubblica il fondamentale: The Design of the Unix Operating System,** che racchiude la descrizione definitiva di come è implementato lo Unix.
  - **Nel 1987 Andrew Tanenbaum pubblica i sorgenti del MINIX** (sistema Unix-like sviluppato per scopi didattici).
- Dato che il MINIX era a 16 bit ed il porting sull'80386 non funzionava molto bene ed una licenza UNIX per l'80386 costava troppo per un privato, portarono **Linus Torvalds a sviluppare un nuovo sistema operativo** (infatti lui dichiara che se solo ci fosse stata già una versione di GNU o il 386BS il progetto Linux non sarebbe mai partito):
  - Torvalds sviluppa il Linux nei primi mesi del 1991 sul suo PC dotato di 80386 usando lo GNU C Compiler come linguaggio, il MINIX come ambiente di sviluppo e il The Design of the Unix Operating System come riferimento.
  - **Il 25 agosto 1991, all'età di 21 anni, pubblica un messaggio sul NG comp.os.minix che fa parte della storia dell'informatica** in cui afferma di stare lavorando ad un progetto per il 386 da qualche mese che vorrebbe presentare alla community per avere feedback da parte degli utilizzatori (e non) del MINIX, affermando di aver preso lo stesso layout fisico del file system, di aver già fatto il porting della bash e il gcc. Il sistema funziona solamente con un AT-Hard Disk (dato che era l'unico a sua disposizione), ed è esente da qualsiasi codice MINIX.
- Inizialmente Torvalds lo chiamo Freax, ma quando venne l'ora di caricarlo sul server FTP dell'università di Helsinki, l'amico di Torvalds lo cambiò il Linux.
- Nel 1992 il Kernel venne rilasciato sotto licenza GNU GPL e da quel momento inizia lo sviluppo dei vari componenti che lo trasformeranno poi in un sistema operativo completo.
- Il logo (il pinguino) venne scelto perché Torvalds venne morso da un Pinguino da piccolo ad uno Zoo.

- Fase 7 e 8: breve lettura…