

Stadi evolutivi dei sistemi di elaborazione

1. Sistemi isolati

Stand alone. Elaborazione di tipo *batch*. Nessuna comunicazione diretta utente-macchina.

2. Sistemi centralizzati

Elaboratori di grosse dimensioni. Accesso remoto tramite *terminali passivi* (non intelligenti) collegati via linea telefonica. Tecniche di *time-sharing*.

Pagina 1

3. Sistemi decentrati - Minielaboratori

Di nuovo decentralizzazione. Notevole potenza di calcolo ad un prezzo relativamente basso. "Rivolta dei mini" per superare la lentezza e la rigidità del servizio centralizzato.

4. Sistemi distribuiti

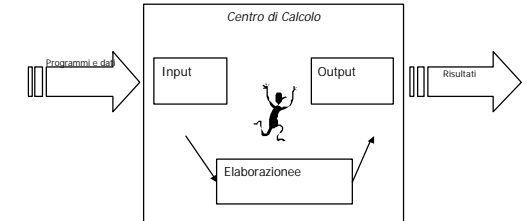
Concepiti come un insieme di unità dotate di capacità operativa autonoma ed al tempo stesso in grado di scambiare mutuamente dati e risorse attraverso una rete di comunicazione.

Intelligenza distribuita in un'ottica di cooperazione ed integrazione delle risorse.

Pagina 2

Stadi evolutivi e modalità d'uso dei sistemi

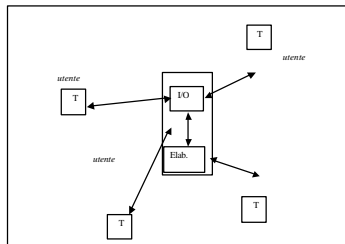
- il sistema *isolato* ('50-'60):
- *batch*



Pagina 3

Stadi evolutivi e modalità d'uso dei sistemi

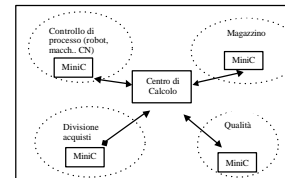
- il sistema *centralizzato* ('60-'70):
- *remote job entry*
- *teleprocessing*
- *time-sharing*
- *multiprocessing*



Pagina 4

Stadi evolutivi e modalità d'uso dei sistemi

- i sistemi *decentrati* ('70 -):
- *minicalcolatore*
- *controllo real-time*
- *data logging*

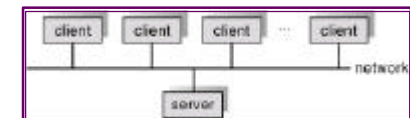


- autonomia locale
- disponibilità sw
- gestione dati non integrata

Pagina 5

Stadi evolutivi e modalità d'uso dei sistemi

- i sistemi *distribuiti* ('80 -):
- *multiprocessori*
- *reti locali*
- *reti geografiche*
- *basi di dati distribuite*
- *protocolli di comunicazione*



Pagina 6

Sistemi distribuiti

sistema	rete	distanza (m)	velocità (bit/sec)
distribuzione funzionale	integrata	1-10	107-109
distribuzione locale	privata	10-100	104-107
distribuzione geografica	privata/ /pubblica	>1000	103-105

Pagina 7

Sistema di elaborazione distribuito

E' costituito da nodi tra loro collegati in ciascuno dei quali sono presenti capacità di:

- elaborazione
- memorizzazione
- comunicazione

Pagina 8

Sistema di elaborazione distribuito

Vantaggi (1):

- *Tolleranza al guasto*

Il verificarsi di un guasto non provoca l'arresto del sistema, ma solo una riduzione delle sue prestazioni. Si ha una minore vulnerabilità rispetto ad evenienze catastrofiche (naturali o dolose).

Pagina 9

Sistema di elaborazione distribuito

Vantaggi (2):

- *Prestazioni*

Essendo l'elaborazione *di norma* effettuata nel posto stesso di utilizzazione, si ha un *miglioramento delle prestazioni* (tempo medio di risposta, throughput) rispetto al caso di elaborazione centralizzata.

Pagina 10

Sistema di elaborazione distribuito

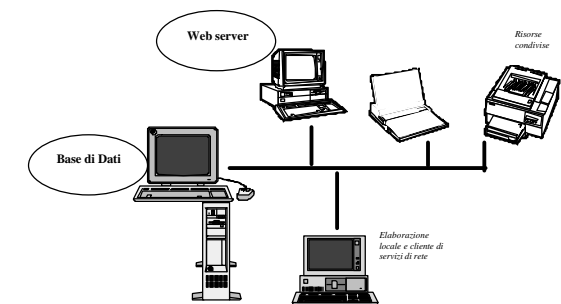
Vantaggi (3):

- *Condivisione*

La capacità di elaborazione, i programmi ed i dati esistenti nell'intero sistema sono, in linea di principio, *patrimonio comune* di tutti gli utenti.

Pagina 11

Sistema di elaborazione distribuito



Pagina 12

Evoluzione dei S.O.

I primi calcolatori:

- Sono privi di S. O.
- Il programmatore è anche operatore interattivo ed ha visione diretta della macchina e disponibilità di tutte le sue risorse
- L'accesso da parte di più utenti è ottenuto mediante meccanismi di prenotazione
- problemi: complessità operazioni, inefficienza e rigidità della prenotazione

Pagina 13

Evoluzione dei S.O.

Prima generazione ('50-'60):

- *virtualizzazione dell'I/O*, librerie di controllo dei device
- separazione del programmatore dalla macchina tramite l'*operatore*
- problemi: *debug* (dump), *set-up* dei *job*

Pagina 14

- riduzione dei tempi di set-up tramite:

- gestione dei job di tipo *batch*
- gestione periferiche con tecniche di *spooling*
- *automatic job sequencing*

=> nasce il S.O. come stratificazione successiva di funzioni volte ad aumentare l'efficienza e la semplicità d'uso della macchina

Pagina 15

Evoluzione dei S.O.

Seconda generazione ('60-'65):

- indipendenza tra programmi e dispositivi usati (*logical I/O*)
- parallelizzazione degli utenti tramite *multiprogrammazione* e *time-sharing*

Pagina 16

Evoluzione dei S.O.

Terza generazione ('65-'75):

- S.O. unico per una famiglia di elaboratori
- risorse virtuali (memoria)
- sistemi multifunzione (scientifico, gestionale)
- linguaggi di comando complessi

Pagina 17

Evoluzione dei S.O.

Quarta generazione ('75-):

- sistemi a macchine virtuali
- sistemi multiprocessore e distribuiti
- interfacce amichevoli per l'utente

Pagina 18

Tecniche di gestione di un sistema di calcolo

- monoprogrammazione
- multiprogrammazione

Pagina 19

Sistema monoprogrammato

- Gestisce *in modo sequenziale* nel tempo i diversi programmi:
 - L'inizio dell'esecuzione di un programma avviene solamente *dopo il completamento del programma precedente*
- Tutte le risorse hw e sw del sistema sono dedicate ad *un solo programma*.

Pagina 20

- Bassa utilizzazione delle risorse:

$$\text{utilizzo CPU} = \frac{T_p}{T_t}$$

- T_p = tempo dedicato dalla CPU all'esecuzione del programma
- T_t = tempo totale di permanenza nel sistema del programma

Throughput = numero di programmi eseguiti per unità di tempo

Pagina 21

Sistema multiprogrammato

- Gestisce *simultaneamente più programmi indipendenti* ciascuno di essi può iniziare o proseguire l'elaborazione prima che un altro sia terminato.
- Le risorse risultano *meglio utilizzate* in quanto si riducono i tempi morti.
- Cresce la *complessità* del Sistema Operativo: occorrono algoritmi per la *gestione delle risorse* (CPU, memoria, I/O), nascono *problemi di protezione*, etc.

Pagina 22

Gestione Batch

- Significa raggruppare i lavori o i programmi *in lotti* per conseguire una *maggiore utilizzazione delle risorse*, cioè un throughput più elevato
- Un concetto che si è evoluto nel tempo:
 - l'operatore raggruppa i programmi in lotti e li immette in tale forma nel sistema per un più razionale utilizzo delle risorse
 - i programmi sono inseriti *nella memoria di massa* e poi elaborati in multiprogrammazione

Pagina 23

Gestione Batch (2)

- Nel caso di multiprogrammazione il S.O. deve provvedere ad algoritmi per la scelta di quell'insieme di programmi che, in esecuzione contemporanea, massimizza il throughput
- La gestione batch può essere *locale* (unità centrale direttamente collegata ai dispositivi di I/O) o *remota* (è presente una trasmissione dei job e dei risultati ed eventualmente una memorizzazione intermedia)

Pagina 24

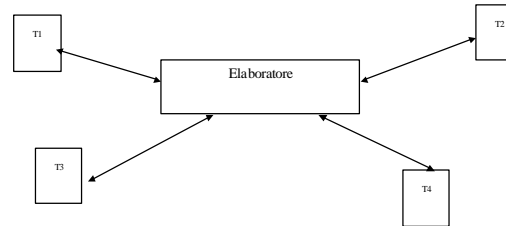
Time sharing (1)

- L'elaboratore serve "simultaneamente" una pluralità di utenti, dotati di terminali, dedicando a ciascuno di essi tutte le risorse del sistema *per quanti fissati di tempo*
- Migliora i *tempi di risposta* (turn-around time) ma peggiora l'utilizzazione delle risorse
- Può essere presente sia in sistemi monoprogrammati che multiprogrammati

Pagina 25

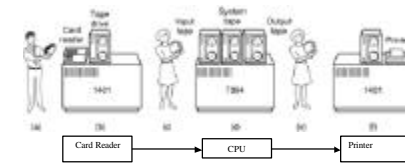
Time sharing (2)

- Normalmente una modalità di gestione time-sharing è adottata nei sistemi *conversazionali*, in cui più utenti contemporaneamente "colloquiano" con il sistema



Pagina 26

Evoluzione dei sistemi batch (1)



- operazioni di I/O *fuori-linea*:
 - il calcolatore principale non è più rallentato da periferiche lente
 - trasparente ai programmi applicativi

Pagina 27

Evoluzione dei sistemi batch (2)

- calcolatori *satellite*:
 - con il compito di scrivere e leggere nastri
 - di potenza ridotta rispetto a quello centrale
 - primo esempio di sistema *multi-computer*

Pagina 28

Evoluzione dei sistemi batch (3)

- Vantaggio ulteriore delle operazioni *fuori-linea* è la possibilità di utilizzare più *lettori di scheda* collegati con una stessa unità nastro in ingresso e più *stampanti* collegate con una stessa unità nastro in uscita
- Non ci può essere accesso contemporaneo da parte della CPU e del lettore di schede o della stampante allo stesso nastro

Pagina 29

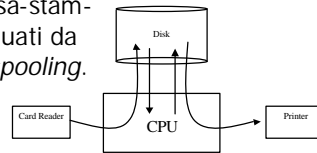
Spooling (1)

- SPOOL: acronimo da Simultaneous Peripheral Operation On Line (NASA Houston Computation Center)
- Per accrescere la velocità di esecuzione dei programmi conviene utilizzare la memoria a disco per simulare i dispositivi di I/O: il disco viene usato come un buffer di grosse dimensioni a cui accedono sia il lettore di schede (e la stampante) che la CPU.

Pagina 30

Spooling (2)

- I trasferimenti lettore di schede-memoria di massa (spool-in) e memoria di massa-stampante (spool-out) sono effettuati da appositi programmi detti di *spooling*.



- Compare il concetto di insieme di programmi pronti per l'esecuzione su disco. Il S.O. può scegliere quale programma mettere in esecuzione (a differenza del caso dei nastri magnetici e delle schede).