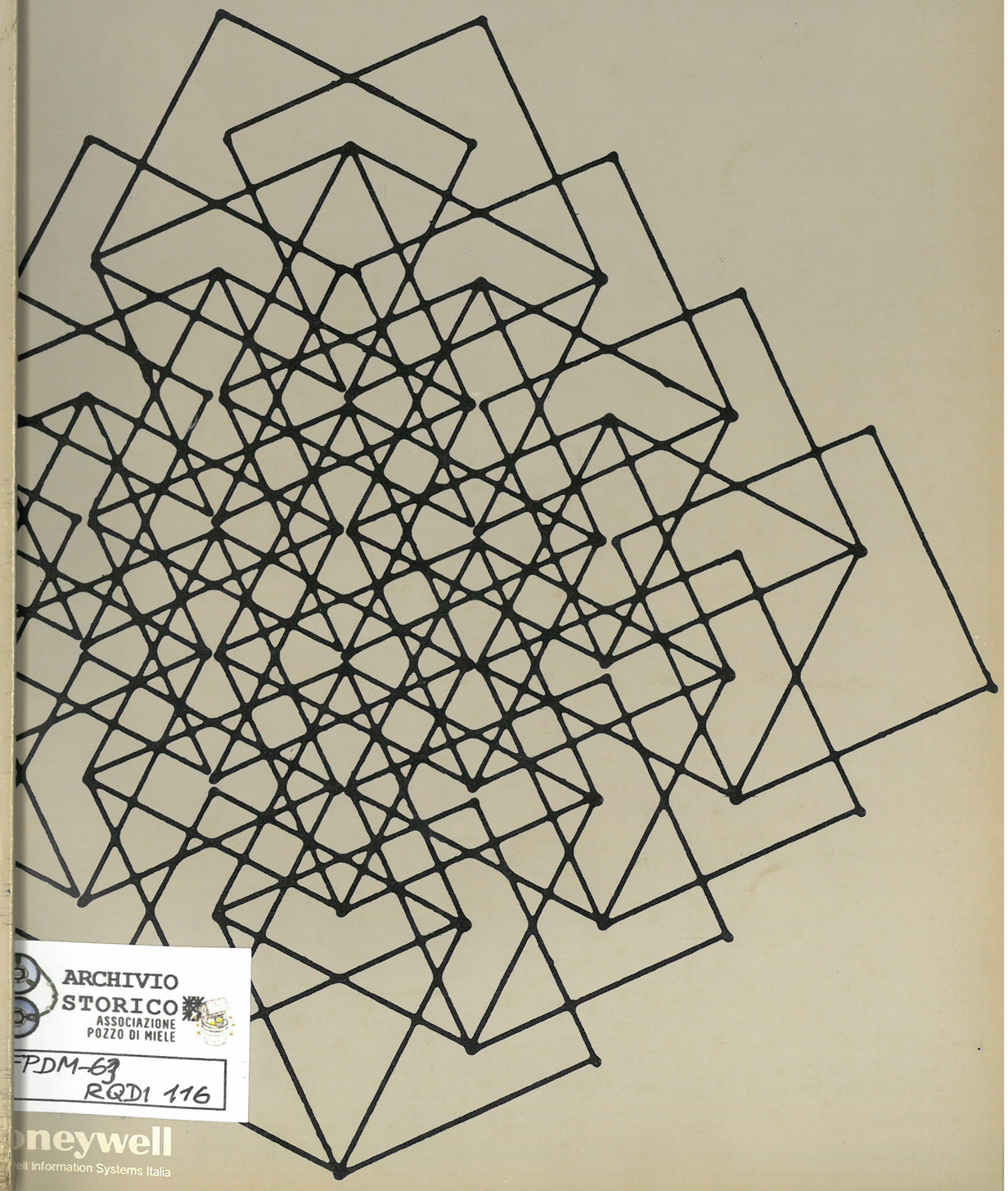
Quaderni di informatica 18

gna di tecnologia ed applicazioni degli elaboratori elettronici – Anno IX – Numero 2 – 1982



oneywell
ell Information Systems Italia

Honeywell

Honeywell Information Systems Italia

Quaderni di informatica

Rassegna di tecnologia ed applicazioni degli elaboratori elettronici
Anno IX – Numero 2 – 1982

Sommario

Introduzione alla affidabilità
"Affidabilità" è un termine ormai popolare ma di cui spesso si ha un'idea molto approssimativa. Questo articolo ha l'obiettivo di chiarire il concetto, sia nella sua accezione generale, sia con riferimento specifico ai componenti elettronici.
G. MATTANA

Strutture avanzate di elaborazione: le "data-flow machines"
La struttura dell'elaboratore è rimasta per lungo tempo ancorata alla concezione originaria di Von Neumann. Ora nuove strutture stanno emergendo. Tra queste, forse la più radicalmente diversa dal passato è costituita dalle "data-flow machines".
G.C. TESSERA

Un modello di automazione dell'ufficio
L'ufficio tradizionale è destinato a scomparire. Al suo posto nasce un ufficio nuovo basato sulle tecniche e gli strumenti dell'informatica. In questo articolo viene illustrato come può essere realizzato e come funziona questo nuovo ufficio.
A. CICU, M. GUALZETTI, R. POLILLO

La chimica computazionale: progettazione di molecole col calcolatore
Il calcolatore apre alla ricerca chimica una nuova era. Questo strumento rende infatti possibile progettare a tavolino molecole in accordo a desiderate proprietà.
G.F. PACCHIONI

Il microcalcolatore "single-chip"
Un microcalcolatore, ossia un microprocessore con memorie e dispositivi di ingresso-uscita, può ormai essere relizzato su un singolo chip. Questo articolo presenta lo stato dell'arte di tali dispositivi e le prospettive di evoluzione.
M. DE BLASI, A. GENTILE, P. PAPOFF

Direttore Responsabile
F. Filippazzi

Comitato di Redazione
A. Cicu, C. Falcetti, P. Lupo,
M. Nobile, G. Occhini
G. Rapelli, F. Sala

Redazione
Honeywell Information Systems Italia
Centro di Ricerca e Progettazione
20010 Pregnana Milanese (Milano)

Stampa
tipolito Maggioni s.a.s.

Autorizzazione del Tribunale di Milano
n. 93 del 20 Marzo 1974

«Quaderni di Informatica» ospita
articoli e contributi di varia provenienza.
La responsabilità delle opinioni
espresse rimane agli Autori.

Introduzione alla affidabilità

GIOVANNI MATTANA

TELETTRA
Divisione Qualità e Affidabilità
Vimercate (Milano)

Parte I - L'AFFIDABILITÀ COME DISCIPLINA E COME PROPRIETÀ DEGLI OGGETTI

1. Concetto generale e definizioni

L'Affidabilità (A. nel seguito) è insieme una *disciplina* ed una *proprietà* degli oggetti. Come disciplina è una teoria di validità generale - che ha per scopo quello di *descrivere, prevedere e dominare* il *comportamento* degli oggetti (semplici, complessi, sistemi) *nel tempo.* La variabile principale di cui si occupa è il guasto (o il suo complemento). Essa comprende un insieme di teorie matematiche e di modelli, di descrizioni di comportamenti fisici, di metodi tecnico/organizzativi.

Come proprietà è stata definita sia in termini *qualitativi* che quantitativi; per il primo caso la definizione è *"l'attitudine di un oggetto ad adempiere ad una richiesta funzione sotto determinate sollecitazioni, per uno stabilito periodo di tempo".*

Per esprimere l'A. in termini *quantitativi* si sono introdotte delle "grandezze" o "caratteristiche" costituite dalle operazioni numeriche o funzionali eseguite sulle variabili osservate. Con riferimento a fig. 1, le variabili osservate possono essere per es. il "tempo tra guasti" oppure il "numero di guasti nell'unità di tempo"; sull'andamento nel tempo di queste variabili, intese come grandezze aleatorie, si possono determinare delle funzioni quali, ad es., la densità di probabilità o la media aritmetica (per la definizione di queste grandezze vedi paragrafo seguente).

Per gli scopi pratici, a queste operazioni matematiche vengono associati degli aggettivi ("modifiers") quali: "nominale", "di previsione", "estrapolato", "vero" (in senso statistico), "osservato". Ciascuno di tale aggettivi implica delle ipotesi ed operazioni aggiuntive, di tipo statistico, oppure fisico (modello di estrapolazione) oppure organizzativo (es. criterio di raccolta dei dati del field). Scopo di tali aggettivi è quello di saldare la parte matematica - rigorosa - con le misure pratiche che contengono, in più, specifiche convenzioni ed ipotesi aggiuntive che devono essere esplicitate.

Conviene sottolineare il *carattere probabilistico dell'affidabilità:* l'affidabilità riguarda la probabilità di eventi futuri, sulla base di osservazioni passate. Pertanto ogni caratteristica di affidabilità può essere usata in funzione di *ciò che è stato osservato* oppure *di ciò che ci si può aspettare,* quest'ultimo essendo espresso in termini di *probabilità.*

Una analisi della definizione sopra riportata impone di soffermarci sulle tre parole chiave: la funzione richiesta, le sollecitazioni, il tempo.

La *funzione specifica* dovrà essere precisata in ogni contesto: più complesso è il sistema considerato e più essa comprenderà numerose sottofunzioni, per ciascuna delle quali deve essere assegnato un *criterio di guasto.* I guasti possono essere ordinati secondo varie *classificazioni* in relazione allo scopo, agli effetti, all'importanza, etc.

Le sollecitazioni, richiamate nella definizione, ci sottolineano che l'A. di un oggetto dipende dalle sollecitazioni; queste possono essere elettriche, ambientali, da trasporto, etc.; possono essere costituite da valori limite (da non superare) o da valori medi (se si assumono ipotesi di danno cumulativo) o da combinazioni - più o meno complesse e severe - di varie sollecitazioni (es. freddo con vibrazioni, fattori corrosivi con umidità, etc.). Per le sollecitazioni si presenta poi in A. il problema rilevante di poter simulare in laboratorio le condizioni presenti in esercizio, e saper instaurare le correlazioni più attendibili.

Definite la funzione e le sollecitazioni, scelta una "caratteristica" di A., interesserà conoscere il suo andamento nel *tempo.* Per es. per il tasso di guasto viene frequentemente ipotizzato un andamento del tipo "a vasca da bagno" (fig. 2); esso è costituito da un primo periodo ad andamento decrescente, chiamato - per analogia con la vita umana - di mortalità infantile; da un secondo periodo ad andamento sostanzialmente costante, chiamato dei guasti casuali, in quanto i cedimenti dipendono da moltissime possibili cause, ognuna con associata piccola probabilità; il terzo periodo infine, quello dei guasti per vecchiaia o per usura, mostra andamento crescente. Nei casi specifici si verificherà come ciascuno di questi andamenti collima con quello proprio di cia-

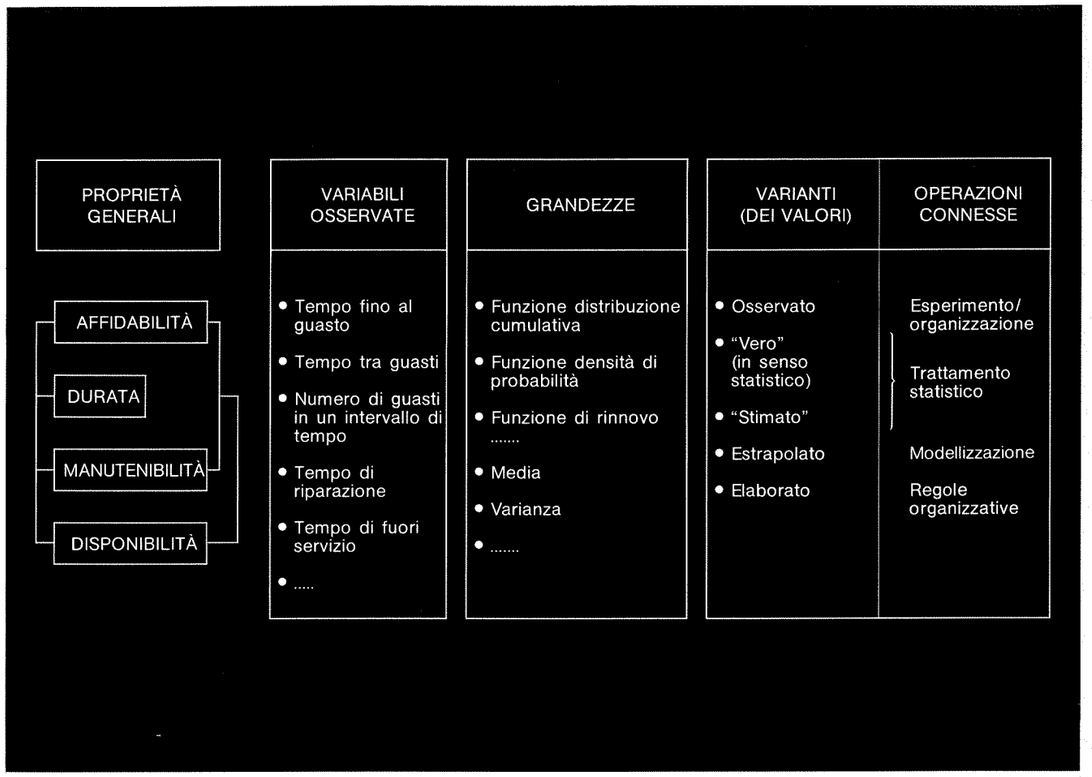


Fig. 1 - Rappresentazione grafica delle relazioni tra proprietà generali, variabili osservate, grandezze, varianti e connesse operazioni.

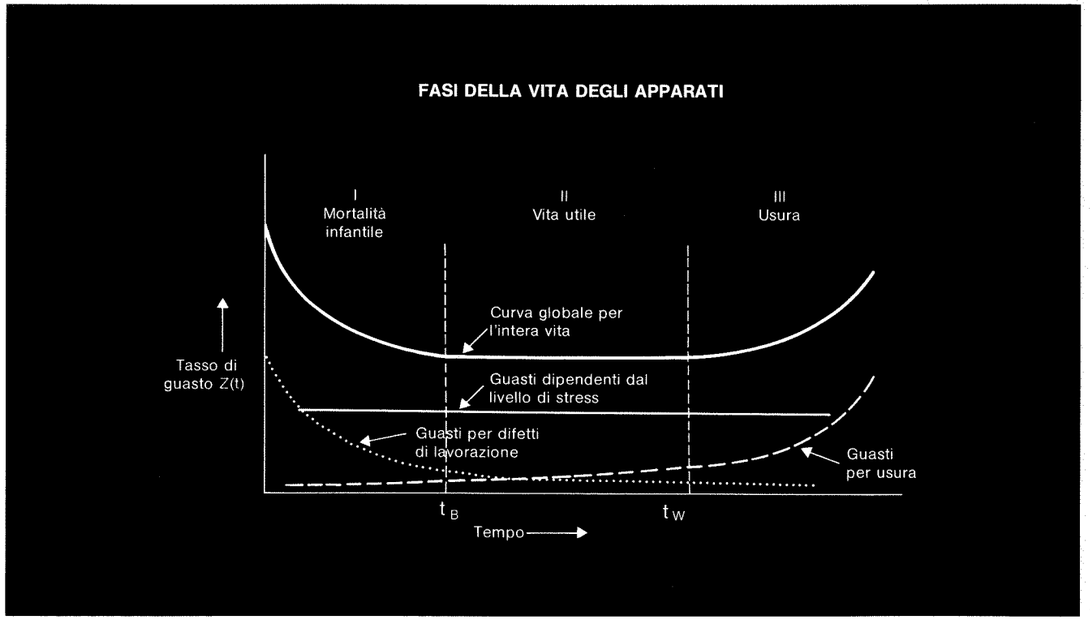


Fig. 2 - Profilo frequentemente ipotizzato per l'andamento nel tempo dei guasti (curva a "vasca da bagno") e le sue diverse componenti.

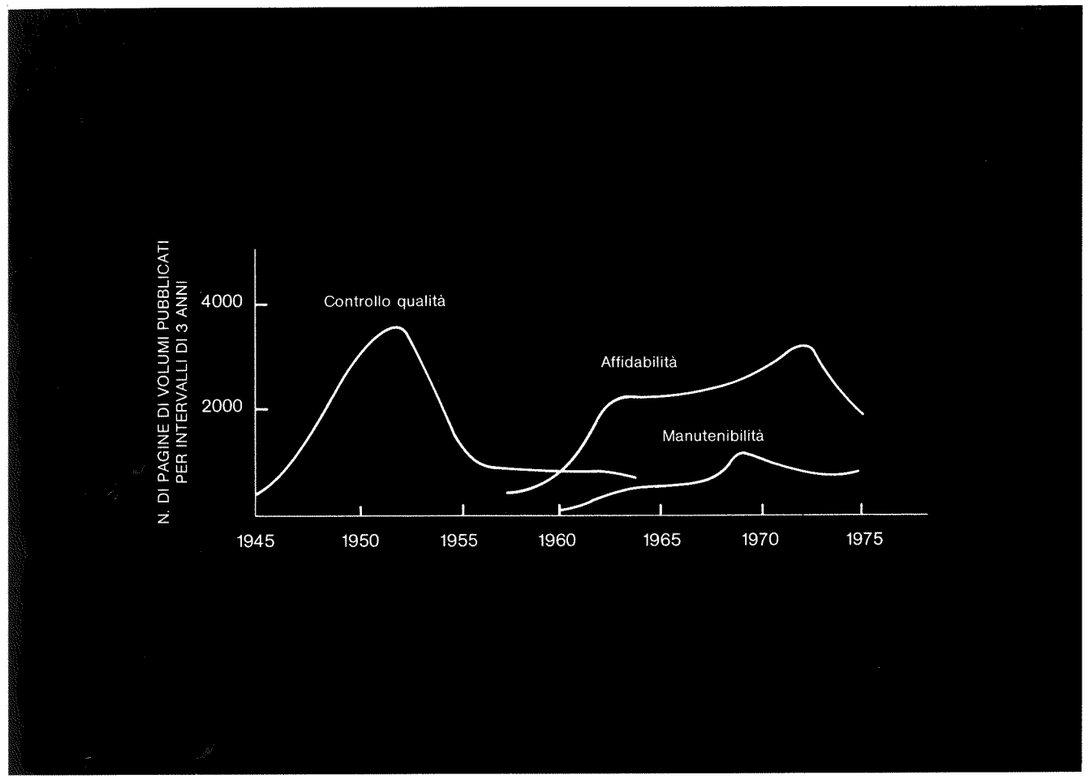


Fig. 3 - Sviluppo della affidabilità e manutenibilità misurato dalle pagine di libri pubblicati [1].

scuna legge di distribuzione statistica (l'esponenziale, la lognormale, la Weibul sono quelle più frequentemente riscontrate ed utilizzate).

2. Nascita e sviluppo dell'Affidabiltà

Per quanto fosse antichissimo il progettare opere aventi ottime durate ed affidabilità, è alla definizione di A. come probabilità (Lusser, 1952) che si fa risalire la nascita della A. come moderna disciplina. Essa si sviluppò molto rapidamente come risposta alla necessità di garantire il funzionamento di reti e sistemi, che divenivano sempre più complessi ed erano costituiti da un gran numero di parti; la bassa affidabilità di queste determinava l'insuccesso dell'intero sistema. In fig. 3 è rappresentata la crescita della disciplina in termini di pagine di libri pubblicati (in termini di articoli, nel 1970 erano già stati pubblicati oltre 20.000 articoli in lingua inglese).

Il miglioramento dell'affidabilità rese possibile lo sviluppo impressionante dell'elettronica; ma la crescita di complessità dei sistemi e l'importanza e l'estensione dei servizi svolti, spingevano a loro volta verso una maggiore A.: diventava infatti sempre più impellente ed importante conoscere in anticipo e dominare il comportamento futuro, cioè gli effetti differiti di determinate scelte. Basterà citare due esempi di segno opposto: lo sforzo della NASA ed in particolare il programma Apollo, ed il black-out del 1965 nella parte nord-orientale degli USA.

L'espansione dell'elettronica come mercato attirava l'attenzione sui costi di manutenzione che rappresentano, nel 1975, un terzo del comparto elettronico del bilancio USA della Difesa.

Di qui la messa a punto di tecniche quali il "Life Cycle Costing in Equipment Procurement" che prometteva ottimizzazioni ingenti e ritorni enormi (fig. 4).

In fig. 5 è riportata una suddivisione, in branche, della A. come disciplina.

Analoga suddivisione è stata proposta per la *manutenibilità,* la cui definizione è:

"Attitudine di un oggetto, in condizioni specificate di uso, ad essere conservato o ripristinato in uno stato nel quale può adempiere alle funzioni richieste, quando la manutenzione è espletata nelle condizioni specificate e usando le procedure ed i mezzi prescritti".

(La manutenibilità può, in dipendenza della particolare analisi della situazione, essere specificata mediante una o più caratteristiche di manutenibilità come una distribuzione discreta di probabilità, il tempo medio di manutenzione attiva, ecc.).

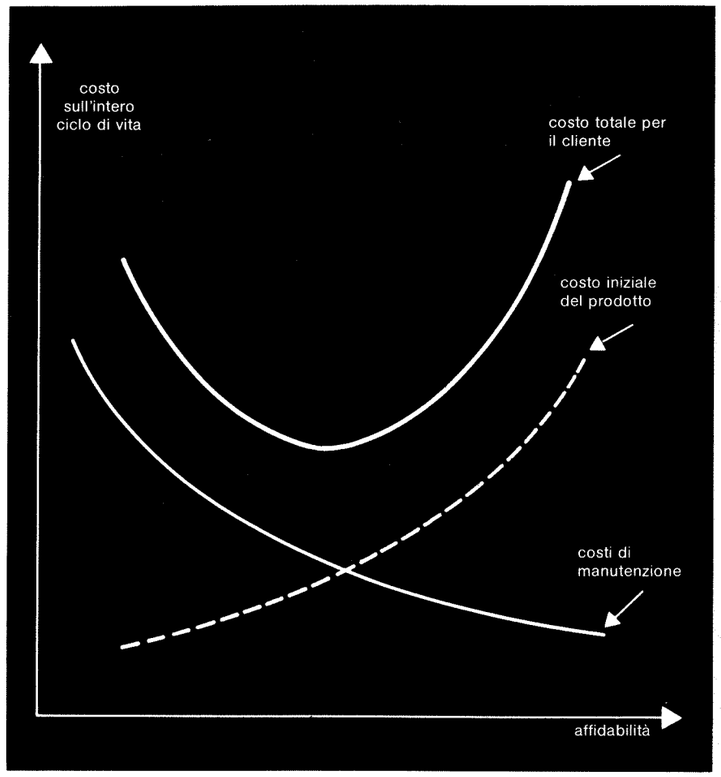


Fig. 4 - Andamento dei costi totali sull'intera vita utile, in funzione dell'affidabilità.

Definita in questo modo la manutenibilità, si può passare a considerare la *disponibilità* di un oggetto considerando non solo le sue rotture, ma anche la sua riparazione (risultanti della facilità a essere riparato, "manutenibilità", e dimensionamento del supporto logistico). Una definizione di "disponibilità" è: "l'attitudine di un oggetto a compiere la funzione richiesta ad un momento stabilito (o per un periodo di tempo stabilito) tenuto conto della combinazione degli aspetti d'affidabilità, manutenibilità e supporto logistico di manutenzione".

3. Il calcolo dell'affidabilità

3a) - Grandezze matematiche

La distribuzione dei "tempi fino al guasto" per oggetti che non vengono riparati e la distribuzione dei "tempi fra guasti" per oggetti che vengono riparati formano la base delle definizioni delle grandezze di A.. Questi tempi sono considerati *variabili casuali* e vengono trattati con i metodi del calcolo delle probabilità.

La probabilità di guasto come funzione del tempo, può essere definita da

$$P(\Theta \leq t) = F(t) \text{ per } (t \geq 0)$$

Essa è anche chiamata *funzione di distribuzione cumulativa,* mentre R(t)

$$R(t) = 1 - F(t)$$

è spesso chiamata *funzione affidabilità* o probabilità di successo.

Si ha inoltre una *funzione densità di probabilità,* f(t), legata alla F(t) dalla relazione:

$$f(t) = \frac{d\,F(t)}{dt}$$

Si ha anche

$$\int_0^\infty f(t)\,dt = 1$$

Il *tasso di guasto* è dato da

$$z(t) = -\frac{1}{R(t)} \cdot \frac{dR(t)}{dt} = \frac{f(t)}{R(t)}$$

Le funzioni f(t) più frequentemente usate sono l'esponenziale, la normale, la lognormale, la Weibull. Una distribuzione esponenziale (cioè $R = e^{-\lambda t}$) corrisponde ad un tasso di guasto costante nel tempo ($= \lambda$).

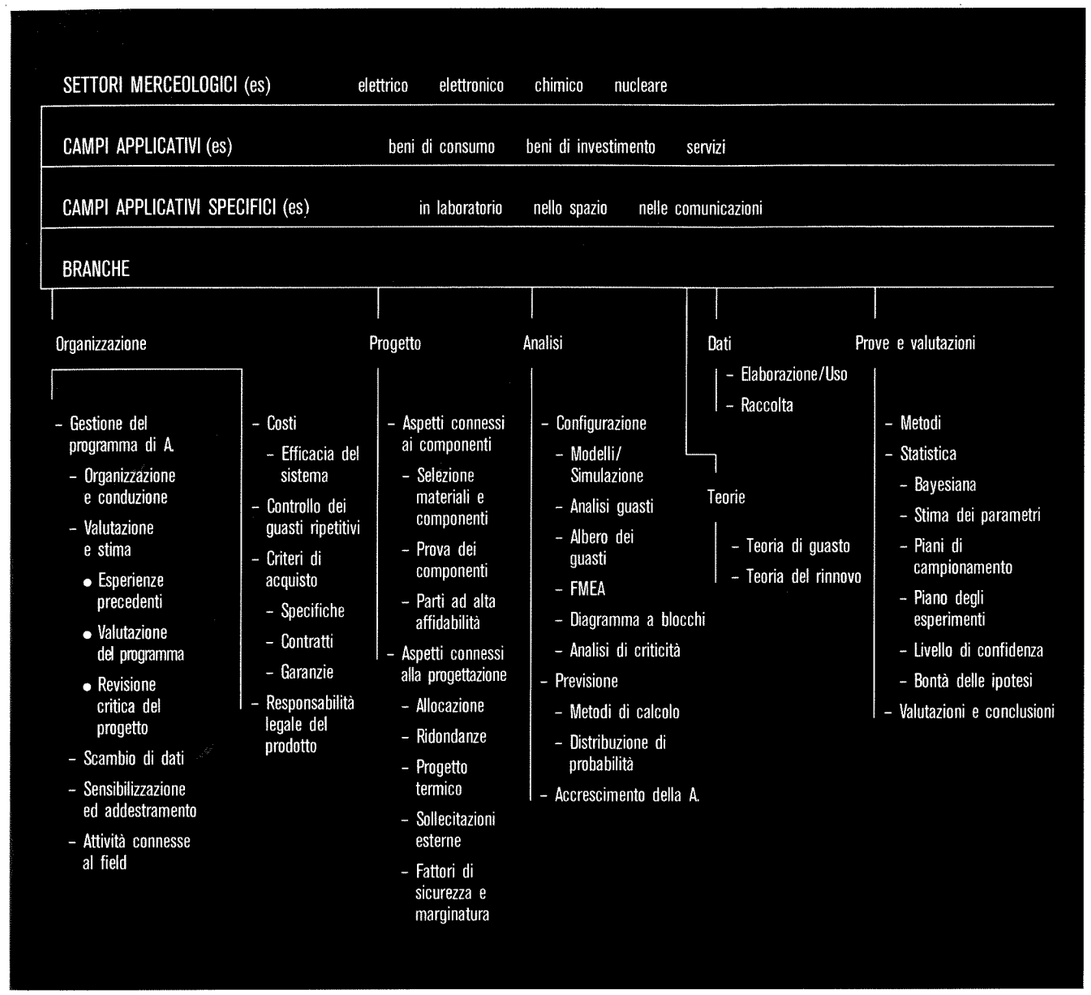


Fig. 5 - Tassonomia dell'affidabilità.

Oltre che con riferimento a tale modello, tali grandezze possono essere espresse in termini di valori della popolazione. Si può allora scrivere.

$$R(t) = \frac{N(t)}{N(o)}$$

dove N(o) è il numero di oggetti che costituisce la popolazione al tempo zero e N(t) il numero di sopravvissuti al tempo t; $z(t_1, t_2)$, tasso di guasto medio dell'intervallo t_1, t_2 è dato da

$$z(t_1, t_2) = \frac{N(t_1) - N(t_2)}{(t_2 - t_1) \cdot N(t_1)}$$

Il tasso di guasto medio è spesso misurato in fit (1 fit = 1 guasto all'ora x 10^{-9}).

3b) - Metodo combinatorio

È il metodo di calcolo più semplice e più usato in elettronica in quanto l'esperienza ha dimostrato la sua idoneità a risolvere la maggior parte dei problemi. Si basa sull'ipotesi che i guasti dei singoli componenti siano statisticamente indipendenti e che gli interventi di manutenzione correttiva siano effettuati solo quando si ha un guasto che pregiudica la funzionalità di tutta la struttura.

- *Struttura serie*

  Si parla di strutture serie quando la rottura di un qualsiasi elemento determina la cessazione della funzionalità dell'intera struttura. Dal calcolo delle probabilità si ha che la probabilità di sopravvivenza dell'intera struttura è data dal prodotto della sopravvivenza dei

suoi elementi; cioè

$$R = R_1 \cdot R_2 \ldots R_n = \prod_1^n R_i$$

Nel caso particolare, ma molto usato, che ogni componente segua la distribuzione esponenziale, cioè $R_i = e^{-\lambda t}$, ed abbia quindi tasso di guasto costante nel tempo, l'equazione precedente diviene

$$R = e^{-\Sigma\lambda_i t} = e^{-\lambda_{tot} t}$$

cioè il tasso di guasto della struttura (es. un circuito) si ottiene sommando i tassi di guasto dei suoi componenti.

– *Struttura parallelo (ridondanza attiva)*

Si parla di struttura parallelo quando la cessazione della funzionalità dell'intera struttura avviene se e solo se sono rotti tutti gli elementi che la compongono.

Nel caso più semplice di due blocchi in parallelo, statisticamente indipendenti, dal calcolo delle probabilità si ha

$$R = R_1 + R_2 - R_1 \cdot R_2 = 1 - (1-R_1)(1-R_2)$$

Più in generale, per n blocchi, si ha

$$R = 1 - \prod_1^n (1 - R_i)$$

Ad esempio per $R_i = 0.6$ con n = 3 si ha R = 0.936.

Tale espressione giustifica l'ampio impiego della ridondanza per aumentare l'affidabilità di sistemi quali gli aerei, etc.

Ovviamente un vantaggio ancora maggiore si ha nel caso che le azioni di manutenzione correttiva vengano eseguite non solo quando cessa la funzionalità dell'intera struttura ma al verificarsi del guasto di un qualsiasi suo elemento.

Si possono poi risolvere, con calcoli analoghi, opportune combinazioni di sistemi serie/parallelo etc.

– *Calcolo dell'affidabilità di apparati e sistemi*

Utilizzando le formule precedenti ed altre analoghe, la teoria matematica dell'A. ci insegna a calcolare l'A. di un sistema, anche complesso, a partire dai valori di A. dei suoi componenti. Occorre sottolineare subito un importante aspetto della A. come tecnica quantitativa. I rigorosi concetti matematici trovano applicazione pratica solo se si riesce ad attribuire dei valori quantitativi di A. ai *singoli blocchi* e quindi ai *componenti elementari.* Questa operazione di *assegnare un valore* diventa un momento essenziale per le applicazioni. (Storicamente il successo della applicazione dell'A. in elettronica è stato largamente determinato da questa possibilità). Ciò apre il problema dei *dati*: delle *banche di dati,* dell'uso dei dati, della *raccolta ed estrapolazione dei dati.*

Spesso si opera costruendo un modello, uno schema a blocchi dell'A., che può essere considerato come uno schema logico che, per mezzo di blocchi interconnessi, illustra l'effetto dei guasti sullo "stato" del sistema. Il diagramma a blocchi può comprendere strutture serie, strutture parallelo, "ridondanze" e combinazioni varie.

Vengono quindi stabilite le relazioni matematiche (funzioni di struttura) che legano l'A. del sistema (probabilità di essere in uno stato di funzionamento o di *non* funzionamento) a quella dei singoli blocchi e queste ultime alla A. dei componenti, utilizzando vari possibili modelli. Molto comode sono quelle che utilizzano la distribuzione esponenziale.

Metodi più complessi ricorrono alla descrizione del sistema attraverso la definizione di tutti i possibili stati del sistema (metodo dello spazio degli eventi), oppure procedono alla decomposizione del sistema in sottostrutture e l'affidabilità del sistema viene calcolata mediante successive applicazioni del teorema di Bayes (metodo di decomposizione) oppure ancora ricorrono a tecniche reticolari di rappresentazione a cui vengono applicati vari metodi di calcolo come il metodo del tracciamento dei percorsi, il metodo dei tagli ed altri. Usualmente per i sistemi elettronici viene ipotizzato l'equilibrio statistico del sistema in quanto praticamente verificato nella sua vita utile.

3c). Altri metodi di calcolo

Esistono altri metodi di calcolo, per descrivere non solo la A., Manutenibilità e Disponibilità di apparati, ma anche quelli di intere reti (di comunicazione, di trasmissione, di illuminazione, etc.). Può interessare sia "l'analisi" (conoscenze dello stato) sia la "sintesi" (problemi di ottimizzazione del progetto o della gestione tendenti a determinare le soluzioni che minimizzano o massimizzano opportune grandezze).

Conviene accennare ai seguenti:

a) *Failure Mode and Effect Criticality Analysis*
Spesso usati ai fini della conoscenza della *sicurezza* di sistemi, si basano, a differenza dei precedenti, sull'*effetto* che un *tipo di guasto* ha su altre parti del sistema e quindi sulle propagazioni dei guasti nella struttura considerata.
Particolarmente utile in questo caso la tecnica dell'albero dei guasti.

b) *Metodi di calcolo con impiego di modelli Markoviani*
Questi metodi possiedono una generalità più ampia di quelli combinatori e meglio si prestano a trattamenti con gli elaboratori numerici, quando si abbia a che fare con sistemi complessi a due stati ("buono"/"guasto"), sia non ripristinabili che ripristinabili (considerando in tal caso anche il processo di ripristino, – tempi di ripristino, etc.).

Per conoscere non solo lo "*stato*" del sistema ma anche "*l'evoluzione*" si definisce la legge che governa le *"probabilità di transizione"* tra gli stati. Interessano ovviamente sia le grandezze istantanee che quelle asintotiche, cioè in condizioni di equilibrio statistico.

Di particolare interesse risultano i processi "Markoviani omogenei" in cui non solo l'evoluzione futura del sistema dipende *solo* dallo stato presente (e non dalla sua storia), ma le singole probabilità di transizione sono non dipendenti dal tempo.

c) *Metodo di simulazione di Monte-Carlo* (applicato al calcolo dei parametri affidabilistici di sistemi complessi).

Consiste nel formulare un modello di tipo probabilistico del fenomeno da analizzare, generare gli eventi in modo casuale, valutare l'effetto di tali eventi sul modello ed elaborare statisticamente il campione ottenuto in modo da ottenere il dato richiesto come stima, con un fissato livello di confidenza.
Sono generalmente utilizzati quando la complessità di calcolo con gli altri metodi diventa proibitiva anche mediante l'uso del calcolatore.
Il lettore che volesse approfondire gli aspetti del calcolo dell'A. può consultare in particolare i volumi citati in bibliografia.

4. La "verifica" dell'affidabilità

Spesso sorge la necessità di dover decidere se il valore di un parametro (tempo tra i guasti, tasso di guasto) che caratterizza l'affidabilità di un prodotto è superiore o meno ad un valore prefissato. Ciò può avvenire sia per esigenze contrattuali (rispetto di valori dichiarati o richiesti) sia a scopo conoscitivo (raggiungimento di valori di obiettivo).

Lo strumento usato è un test di ipotesi statistica basato sulle tecniche di campionamento e la decisione di conformità o meno viene basata sui dati osservati da un campione di oggetti nelle effettive condizioni di esercizio (prove di accettazione in impianto) o in prove simulate (prove di accettazione in fabbrica).

Sussistono due cause di incertezza e quindi di possibile errore nella decisione; sono:

– l'incertezza dovuta al fatto che si decide in base ad un dato campionario. La curva operativa del piano (probabilità di accettazione in funzione del valore vero del parametro incognito) descrive completamente tale incertezza;
– l'incertezza connessa alla capacità di simulare le condizioni di stress più significative quando il test viene eseguito in una prova simulata.

Grandezze caratteristiche del piano sono:

– rischio del fornitore: probabilità di rifiutare un prodotto avente affidabilità pari a quella da verificare (α)
– rischio dell'acquirente: probabilità di accettare un prodotto avente affidabilità pari ad un limite inferiore prefissato (β)
– rapporto di discriminazione: rapporto (D_m) tra il valore da verificare (m_0) ed il limite inferiore prefissato. Tale rapporto caratterizza la selettività del piano.

I piani statistici di prova sono diversi a seconda dell'ipotesi di legge di distribuzione presunta (esponenziale, Weibull, Gaussiana) e possono essere di due tipi:

– a campionamento semplice, nei quali la decisione viene presa all'accadere di un numero fissato di guasti o dopo che è passato un tempo fissato
– a campionamento sequenziale, nei quali, al verificarsi di ogni guasto, è possibile decidere se accettare, se rifiutare oppure se continuare la prova.

La normativa più completa sulle prove di affidabilità per apparati è la IEC-605 che amplia la precedente MIL-STD-781 e l'adatta ad apparati non militari.

Nella fig. 6 è riportato un piano sequenziale tratto dalla norma IEC-605-7, relativa alla distribuzione esponenziale.

In esso sono inseriti due andamenti temporali di guasto, che portano rispettivamente al rifiuto ed alla accettazione del prodotto. Da notare il tempo cumulativo trascorso al momento della decisione, espresso in multipli di m_0, e quindi tanto minore quanti più oggetti è possibile mettere in prova.

5. L'affidabilità "pratica"

Può essere definita come l'uso appropriato e ponderato delle tecniche di A. per ottenere l'A. come proprietà degli oggetti ai livelli voluti.

È il risultato di:

– un uso appropriato delle tecniche (tutta l'*ingegneria* della A.)
– una organizzazione adeguata.

Di fatto *l'A. degli oggetti è una caratteristica cumulativa* di tutto il processo costruttivo e dipende quindi da numerosissimi fattori presenti in *tutte le fasi* di realizzazione del prodotto (progetto, industrializzazione, produzione, collaudo, etc.).

Ne segue che per costruire l'A. sono essenziali due operazioni: individuare i punti deboli di tale successione (analisi di sensitività) e massimizzare l'A. di ciascuna fase con metodi specifici orientati alla tecnologia. Il controllo dei fattori della lunga successione di fasi (materiali, progetto, cicli di lavoro, prelievi, analisi difetti, valutazioni ed azioni sul prodotto finito), assume una importanza determinante.

Ma ciò comporta:

a) *una organizzazione finalizzata all'Affidabilità* presente lungo tutto l'arco di realizzazione del prodotto, supportata da un efficiente *sistema informativo per l'affidabilità;*
b) *un processo di retroazione* (tecnico ed organizzativo insieme) per attuare tempestivamente le azioni migliorative;
c) *l'attuazione di azioni speciali* sul prodotto finito per filtrare quanto può essere sfuggito (e non si tratta in genere di azioni sempli-

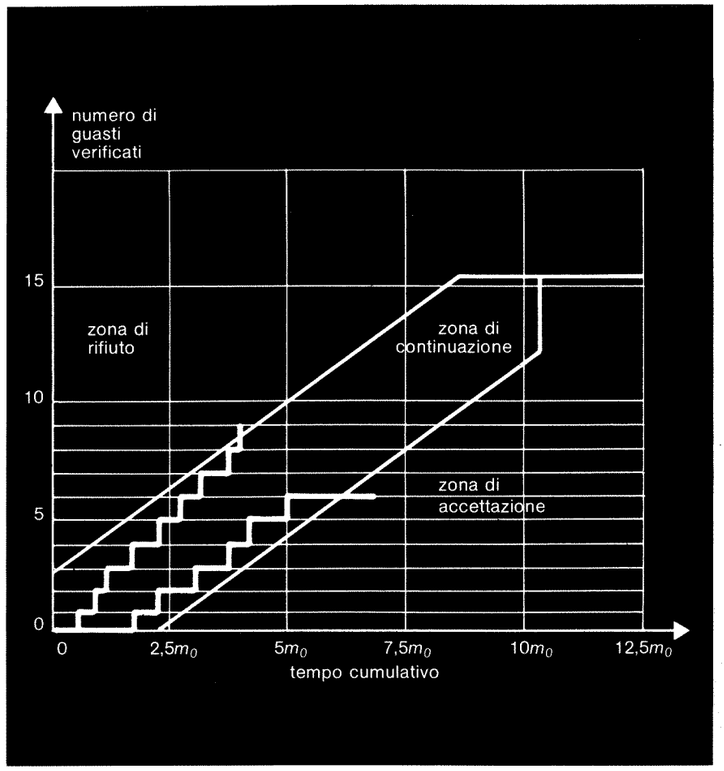


Fig. 6 - Esempio di piano di campionamento.
Valori: $\alpha = 0{,}10$ $\beta = 0{,}10$
$D_m = 2{,}0$
Tempo cumulativo = (tempo effettivo x numero apparati)/m_0

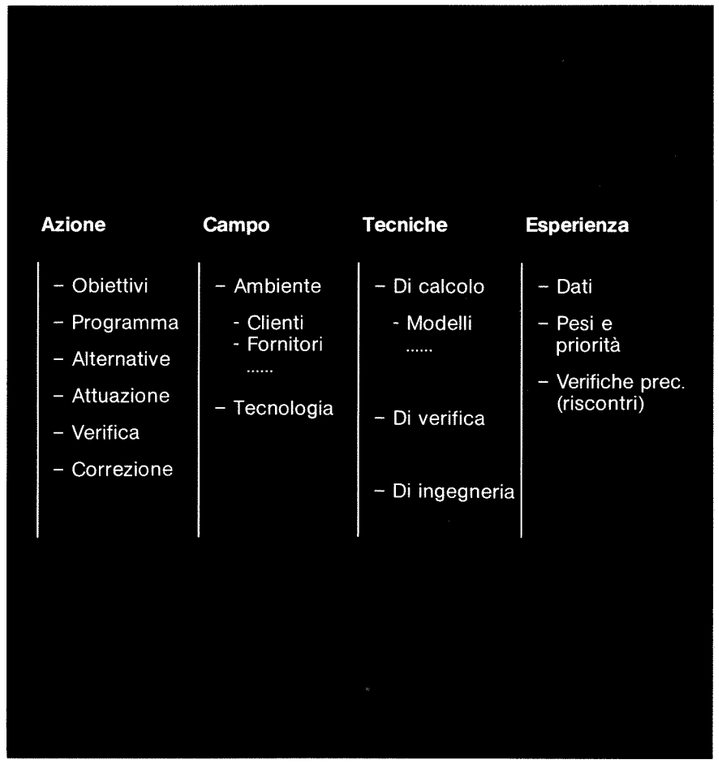


Fig. 7 - L'Affidabilità pratica e i suoi filoni principali.

ci: occorre misurare una idoneità futura o un grado di robustezza attuale).

Può essere descritta (v. fig. 7) come l'operare attivo lungo i filoni ivi considerati: l'azione, il campo, le tecniche, l'esperienza.

I *"Programmi"* di A. stabiliscono l'uso dei metodi a fronte degli obiettivi, lungo tutto il ciclo di un prodotto. Il documento più ufficiale che regolamenta tali piani è il MIL-STD-785.

A differenza dei metodi che si arricchiscono ma si conservano, la A. pratica è quindi qualcosa che si trasforma tutti i giorni, nel peso dei fattori e nella scelta dei metodi e che in definitiva si misura però concretamente attraverso il comportamento del prodotto (reale o simulato).

Per es., negli apparati elettronici, i tre fattori che più condizionano l'A. degli apparati sono: i componenti, il progetto, le lavorazioni.

È allora necessario utilizzare durante il progetto tutti i metodi offerti dalla ingegneria dell'A. (dalla ripartizione della A. nei vari blocchi alle ridondanze, dal sottosfruttamento delle parti al progetto termico, dalla semplificazione circuitale alle protezioni da sovratensioni, dalla design review alle verifiche).

In produzione per minimizzare l'effetto delle lavorazioni sulla A. si userà largamente il tradizionale Controllo Qualità ma finalizzato all'A. e quindi inclusivo della valutazione dei processi produttivi.

La disciplina dell'A. ha messo a punto vari *metodi,* tra cui:

- la previsione
- la dimostrazione
- la estrapolazione e le prove accelerate
- la raccolta dei dati
- l'analisi dei guasti
- "i programmi" di affidabilità ("costruzione" + verifica + crescita + organizzazione)

e le relative *normative* (sopratutto MIL e IEC).

La *precisione pratica* di ciascuno di questi metodi si affina in continuazione anche attraverso *confronti di dati* ottenuti con i diversi metodi.

PARTE II - AFFIDABILITÀ DEI COMPONENTI ELETTRONICI

1. Importanza dei componenti nella affidabilità degli apparati

I componenti concorrono in modo significativo alla A. degli apparati: si può stimare che i guasti casuali intrinseci pesino orientativamente per la metà dei guasti degli apparati (la letteratura e l'esperienza denotano però una sensibile dispersione) e anche quando il progettista può fare un progetto con A. maggiore di quella della parti, incidono comunque sensibilmente sulla manutenzione.

È sulla base dell'A. dei componenti (opportunamente introdotta in idonei modelli) che il progettista *calcola* l'A. degli apparati e sistemi, che il progettista *sceglie* tra soluzioni alternative, che il *cliente* verifica e/o confronta tali calcoli. E ciò vale non solo quando si è interessati al valore quantitativo delle stime, ma anche quando si debbono evitare certi tipi di guasti (tecniche di albero dei guasti, etc), cioè si è interessati al *modo* del guasto. Quindi l'operazione di assegnare un valore *quantitativo* alla A. dei componenti costituisce una operazione fondamentale per tutta l'Affidabilità in elettronica.

2. Miglioramenti nell'affidabilità dei componenti elettronici

A partire dagli anni 1950-55 l'A. dei componenti più importanti (gli attivi ed i condensatori), che era molto bassa, è migliorata di circa quattro decadi, a parità di complessità, passando grosso modo da 5 guasti su 10^4 comp. x ora a 5 su 10^9 comp. x ora.

In fig. 8 si riporta un esempio di questo trend per una tecnologia consolidata, quella dei circuiti integrati bipolari a piccola complessità. Per *conoscere* i tassi di guasti, il metodo seguito negli anni '60 fu quello di mettere in prova un numero di componenti sufficienti a generare i valori necessari in condizioni eventualmente accelerate.

Col progredire dell'A. intrinseca, questo metodo divenne impraticabile e da quel momento nell'A. dei componenti si affermarono i metodi fisico-tecnologici, tesi a *migliorare* la tecnologia, conoscerne i limiti, indagare sui meccanismi di guasto, più che a fissare dei valori; il "perchè" dei guasti divenne decisamente più importante del "quanto".

3. Trend storico e valori correnti di affidabilità dei componenti elettronici

Si è visto in fig. 8 il miglioramento della A. dei circuiti integrati bipolari a piccola complessità, col progredire delle tecnologie. Per i semiconduttori discreti si trovano andamenti analoghi.

A titolo orientativo si riportano in tabella alcuni valori attuali di famiglie di componenti, con l'avvertenza che si tratta di componenti "professionali", impiegati correttamente, in condizioni "favorevoli" ed in funzionamento continuativo.

I fattori di stress possono modificare drasticamente anche di alcuni ordini di grandezza, tali valori.

Sono quindi importanti i modelli che legano i valori di affidabilità ai fattori di stress considerati più rilevanti.

Tasso di guasto di componenti elettronici professionali (in "fit")	
Condensatori in mylar	1 ÷ 3
Elettrolitici al Ta solido	3 ÷ 8
Elettrolitici Alluminio	10 ÷ 40
Connettori (per "pin")	0,5 ÷ 1,5
Resistori	0,5 ÷ 2
Transistori, Si NPN, 1 W	8 ÷ 20
Saldature	0,1 ÷ 0,3
CI, TTL 30 gate	20 ÷ 60
CI, TTL 100 gate	30 ÷ 100

4. Modelli per collegare l'affidabilità al valore di stress e quindi anche alle condizioni di impiego

Il documento più diffuso contenente tali modelli è il MIL - Hdbk - 217 aggiornato almeno parzialmente con frequenza quasi annuale. Lo

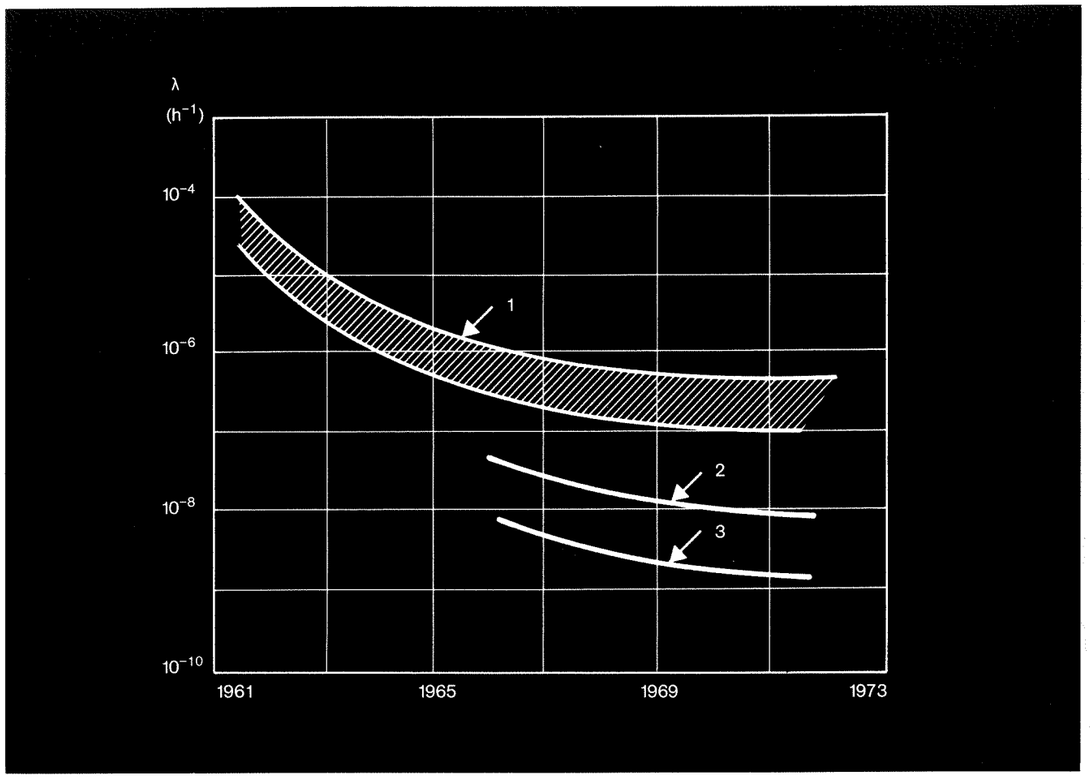


Fig. 8 - Evoluzione dell'affidabilità dei circuiti integrati bipolari (1 = dispositivi standard, 2 = dispositivi standard con setacciatura, 3 = linea di produzione speciale).

schema generale del modello è una relazione del tipo

$$\lambda = \lambda_b\ (T,S)\cdot\pi_E\cdot\pi_Q\cdot\pi_L\cdot\pi_{TE}\cdot\ldots\ldots\pi_n$$

dove λ_b è il valore specifico di ciascuna tecnologia e dipende dalla temperatura e dagli altri stress più rilevanti secondo tabelle riportate nel documento; π_E è il fattore di "ambiente" (fisso, mobile, benigno, severo, etc); π_Q è il fattore di qualità (diversi livelli di controllo e di screening); π_L è il fattore apprendimento; π_{TE} è il fattore "tecnologia"; π_n sta per fattori specifici relativi a ciascuna tecnologia.

Il modello generale per il calcolo di λ è rappresentato nella fig. 9. I fattori π sopra citati possono assumere un certo numero di valori discreti, con variabilità totale piuttosto ampia.

A titolo di esempio, la fig. 10 mostra l'andamento calcolato del tasso di guasto λ in rapporto alla temperatura di giunzione T_j di circuiti integrati digitali bipolari.

Questo richiamo serve qui a mettere in luce alcuni punti:

a) questo modello "spacca" il tasso di guasto "in uso" in numerosi fattori: alcuni di questi sono relativi al componente "in sè", altri dipendono da come viene applicato.

b) La variabilità che ne risulta è enorme: anche 5 o 6 decadi considerando i casi estremi e 3 decadi escludendo alcune combinazioni esterne.

c) Il progettista, oltre ad un importante riferimento convenzionale, ha uno strumento per usare al meglio un dato componente, in termini di affidabilità.

d) Il tasso di guasto è supposto costante in tutta la vita utile (distribuzione esponenziale).

e) Il fattore π_Q (determinato anche dagli screening) influenza tutta la vita, e non solo il periodo iniziale, ed inoltre ha una influenza rilevantissima (rapporto 1 : 70 nella ultima edizione, rapporto 1 : 300 nella precedente).

f) Il fattore apprendimento π_L viene stimato nel rapporto 1 : 10 e dura sei mesi.

Fig. 9 - Modello generale per il calcolo del tasso di guasto λ di un componente. I numeri 1, 2, 3, corrispondono a diversi livelli di stress (tensione, potenza, ecc.) applicati al dispositivo.

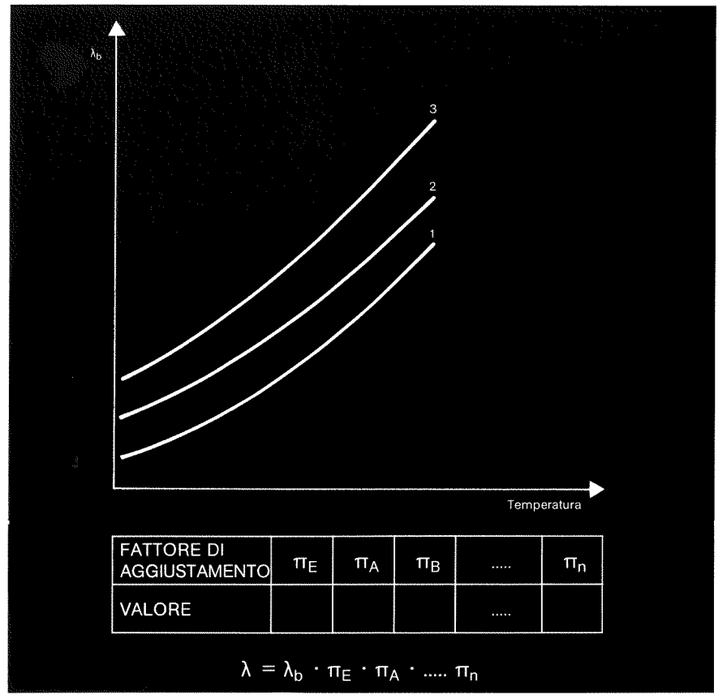
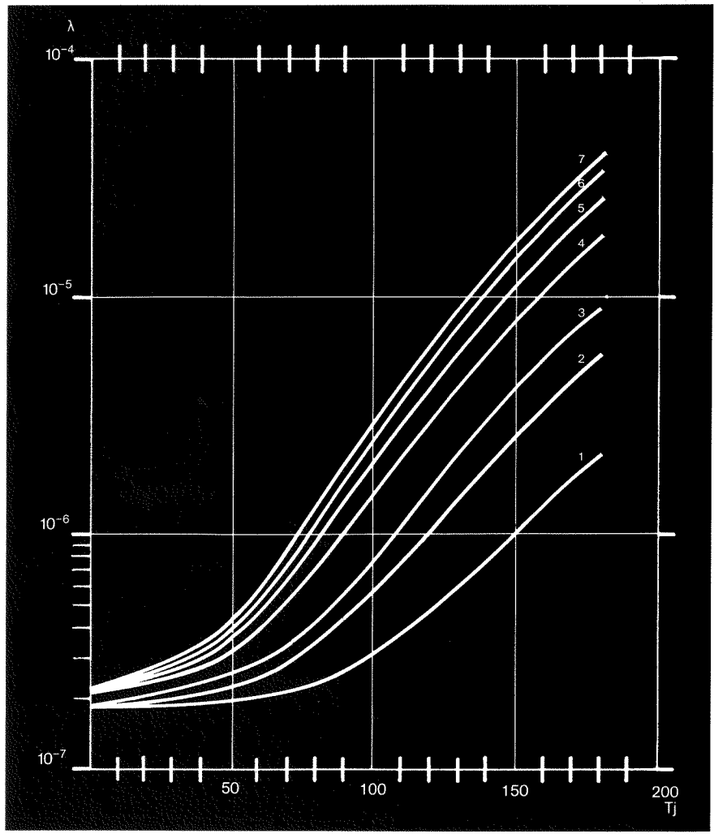


Fig. 10 - Andamento del tasso di guasto λ in funzione della temperatura di giunzione (T_j) per circuiti integrati bipolari (LTTL, LSTTL) per vari valori di complessità circuitale. (I numeri da 1 a 7 corrispondono a complessità tra 1 e 99 gate). Il valore di λ è calcolato con i seguenti valori dei parametri: $\pi_Q = 35$, $\pi_E = 1$, $\pi_L = 1$, Package DIP 14 pin.



5. L'affidabilità dei semiconduttori: (aspetti consolidati)

Esiste una buona concordanza sui seguenti punti:

a) *l'affidabilità di dispositivi a tecnologia consolidata* ed in contenitore ermetico *è molto buona,* dell'ordine della/delle decina di fit (1 fit = 1 guasto/10^9 compon. x ora). Essa è la risultante di azioni lungo tutto il processo produttivo per tenere sotto controllo e/o eliminare le possibili cause di guasto insite nel progetto, nel controllo in produzione di tutte le variabili e criticità del processo, negli eventuali screening per evidenziare quel residuo che può essere sfuggito. Ma è pressochè *impossibile "misurare" le decine di fit* in condizioni poco accelerate.

b) Questi valori di A., risultante cumulativa di tutto il processo produttivo, sono attribuiti alle due componenti della popolazione: la sottopopolazione "forte" e la piccola, ma determinante, popolazione "debole" (*ipotesi bimodale).*

c) *Non esiste un "modo di guasto" prevalente* ma esistono modi differenti di guasto, di cui almeno 3 - 5 con rilevanza similare.

d) Le *tecniche per controllare la robustezza* – diversa da lotto a lotto – *della popolazione "forte" sono diverse da quelle per ricercare i pochi deboli:* per la prima si ricorre a *prove* molto *accelerate* su campioni; per provocare il cedimento dei secondi occorre sottoporre tutti i pezzi del lotto ad opportuni set di *sollecitazioni.* Per i guasti di quest'ultimo tipo viene individuato un andamento decrescente nel tempo; per gli altri si ritiene più vera una distribuzione lognormale.

   Queste sollecitazioni ("screening") hanno naturalmente un costo non trascurabile (che può andare dal 20 - 30% fino al 300 - 400%).

   Sono state pubblicate molte stime sulla efficacia attesa.

e) Le affermazioni di cui al punto precedente valgono essenzialmente per i "chip". *I contenitori* ("plastici" oppure "a cavità", più o meno "ermetici") possono introdurre o favorire altre cause di guasto, ed inoltre possono essere più o meno idonei ai trattamenti di setacciatura e più o meno efficaci nel proteggere dalle sollecitazioni esterne.

f) Il fattore *"apprendimento"* può avere talvolta un'ampiezza di varie decadi ed una durata assai maggiore di quanto non si sia soliti ipotizzare.

Fig. 11 - Tassi di guasto, in esercizio, di circuiti integrati bipolari. Si nota la mortalità intantile e quella dovuta ai meccanismi di lungo periodo [3].

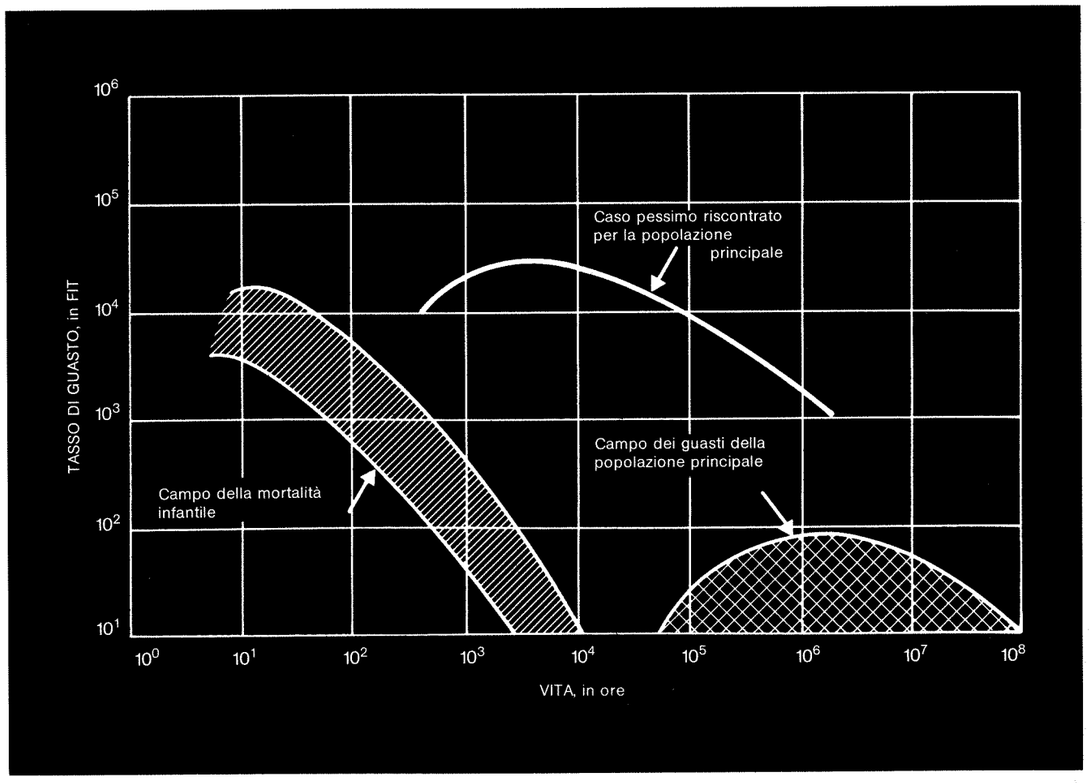


Fig. 12 - Relazioni tra le durate di vita osservate, per diverse temperature, in prove accelerate. Si nota il diverso andamento per la popolazione debole e per quella principale.

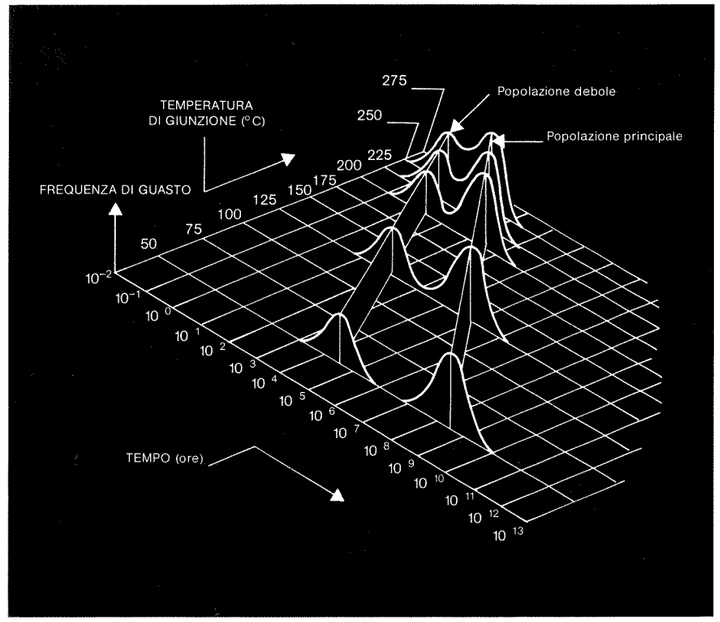


Fig. 13 - Evoluzione e linee di tendenza della complessità, costo e affidabilità dei circuiti integrati.

In fig. 11 è riportato un esempio di andamento del tasso di guasto durante la vita. Si nota la distribuzione bimodale, con andamento iniziale vistosamente decrescente e successivamente retto da distribuzione lognormale. In fig. 12 si osserva la dipendenza dalla temperatura in studi di dipendenza dalla tensione.

6. L'evoluzione tecnologica e le sue implicazioni sulla affidabilità e sulla qualità

È ben noto (senza entrare in tale argomento) che la tecnologia dei semiconduttori si evolve (o si rivoluziona!) con una velocità impressionante, che porta ad un continuo aumento del grado di integrazione, ad una diminuizione del costo, ad un atteso – a regime – miglioramento dell'A.; questa è ora misurata per unità di funzione, per porta logica o per bit di memoria (V. fig. 13). Per ottenerla cambiano i processi, cambiano i rapporti cliente-fornitore, cambiano i criteri di rischio. Si opera in regime altamente dinamico. Il fattore apprendimento gioca un ruolo sempre più importante.

Normalmente però, su nuove tecnologie, non si parte dai valori di affidabilità ottenuti sulle tecnologie precedenti, ma si può partire da valori sensibilmente peggiori.

Tutto ciò ha rilevanti implicazioni su A. e Qualità. Infatti i difetti iniziali, manifesti o latenti, possono essere non trascurabili (da qualche % ed anche il 10%). Diventa allora necessario provare all'unità in collaudo accettazione tutti i dispositivi, in un gran numero di loro configurazioni (il problema della "copertura" ottimale) a mezzo di macchine automatiche pilotate da un calcolatore e quindi comandate da un software specifico. Non è opportuno che arrivi sulle linee di montaggio una tale percentuale di difettosi che ridurrebbe drammaticamente la resa. Non pochi laboratori indipendenti sono sorti, nei vari paesi, per effettuare queste operazioni.

Inoltre, per cautelarsi dalla mortalità infantile, l'utilizzatore deve decidere che cosa comperare tra le diverse opzioni qualitative esistenti sul mercato per lo stesso componente; oppure può effettuare per proprio conto opportune setacciature, in genere combinandole con il controllo funzionale al 100%. Ma l'ottimizzazione delle setacciature dipende dalla presenza di specifici modi di guasto, per cui occorre che l'utilizzatore entri nella conoscenza tecnologica dei dispositivi ed effettui delle analisi di guasto, e ciò richiede la conoscenza delle nuove tecniche sofisticate della *fisica dei meccanismi di guasto* e la disponibilità delle non meno sofisticate attrezzature.

7. La fisica dei meccanismi di guasto ed i nuovi strumenti di valutazione

A partire dagli inizi degli anni '60 (il primo convegno annuale, poi ripetuto regolarmente, su "Physics of Failure in Electronics", è del 1962) si è cercato di indagare sulle possibili cause fisiche dei guasti, cercando i possibili effetti di non compatibilità dei materiali, di dimensionamento dei circuiti per evitare l'elettromigrazione, di protezione dalle scariche statiche, di effetti dell'umidità, per citarne solo alcune.

Si è così spaccato il dispositivo nelle sue ulteriori componenti (le metallizzazioni, i gradini di ossido, le connessioni, i contenitori, il "volume") e per ciascuno si sono studiati i fenomeni che provocano *qualche* cedimento, così pervenendo, tra l'altro, a conoscere meglio il comportamento a fatica oppure *marginale* di ciascuna fase tecnologica.

Per far ciò si sono messe a punto specifiche tecniche di attacco e di microanalisi, utilizzando ampiamente le varie tecniche della microscopia elettronica (es. in figg. 14 e 15).

Il conoscere i modi ed i meccanismi di guasto è diventato uno strumento per individuare tempestivamente i limiti del progetto del dispositivo e/o delle sue tecnologie e quindi per ridurre i rischi della innovazione veloce. Esso non avviene solo "post mortem", ma può essere fatto in tempo reale, per "spiare" in anticipo i punti deboli di una tecnologia attraverso la caratterizzazione tecnologica, o addirittura attraverso la ricostruzione dei criteri progettativi adottati dal costruttore ("progetto a ritroso").

Si tratta delle nuove tecniche, a forte circolarità e spiccata professionalità, per la valutazione continuativa, in tempo reale, della Affidabilità dei nuovi processi tecnologici.

Parte III – CONCLUSIONI

L'affidabilità può essere considerata oggi una disciplina avente un corpo di conoscenze e di metodi abbastanza ben assestato. Ai metodi generali si va aggiungendo un numero sempre maggiore di metodi associati a specifici campi tecnologici: si potrebbero citare per es. l'affidabilità strutturale, usata in meccanica, la tecnica dell'albero dei guasti, particolarmente importante nelle applicazioni orientate alla sicurezza, etc; non sempre facile risulta la scelta, dalla biblioteca di metodi, di quello più adatto a fronte di applicazioni specifiche, specialmente in campi nuovi.

Esistono poi filoni nuovi di particolare vitalità; fra i tanti, si possono segnalare i seguenti:

a) Il problema dei dati e delle banche di dati. Esso richiede la messa a punto di definizioni e procedure accurate, specialmente a fronte dell'obiettivo dello scambio fra banche o dell'accumulo dei dati.

b) Il problema del software e dei guasti provocati da errori e carenze del software.
Il software è ormai presente in quasi tutti i sistemi ma le metodologie per evitare gli errori, effettuare le verifiche, presentare i risultati, sono certamente diverse da quelle per l'hardware ma non ancora consolidate. Esistono vari modelli descrittivi assai diversi fra loro anche come impostazione di base.

c) I sistemi che sopportano i guasti (Fault Tolerant Systems). Questo filone, che trova numerose applicazioni nel campo di sistemi digitali, costituisce una nuova metodologia per affrontare i problemi di A. di sistemi complessi, caratterizzati da stringenti esigenze di continuità di servizio e prestazioni. Secondo tale metodologia i sistemi vengono strutturati secondo un'architettura per livelli, con ridondanze tali da conferire la proprietà di riconfigurazione dinamica volta ad eliminare l'effetto del guasto, oltre che di autodiagnostica;

d) Il problema della simulazione, nelle prove di A., delle condizioni ambientali.

Affronta la problematica delle prove combinate (CERT – Combined Environmental Testing) nonchè gli associati problemi della conoscenza e descrizione statistica delle sollecitazioni reali e relative ("trasformazioni" per combinarle con le distribuzioni statistiche delle robustezze).

e) Il problema delle setacciature a livello di apparati: si tratta di individuare il gruppo di prove e relative modalità e strumentazioni, che abbia la massima efficacia nello scoprire i difetti latenti.

È difficile, in una panoramica riassuntiva come la presente, dare l'idea della vastità multiforme di una disciplina come l'A.. Appare però ormai ben consolidato che l'A. è uno strumento essenziale per ottimizzare il progetto dei sistemi verso disponibilità, costi, sicurezza; ed anche che l'A. è una componente essenziale dell'avanzamento della tecnologia; infine che conservare o migliorare l'affidabilità dei sistemi, mentre questi diventano sempre più sofisticati e complessi, rappresenta un obiettivo di grande impegno.

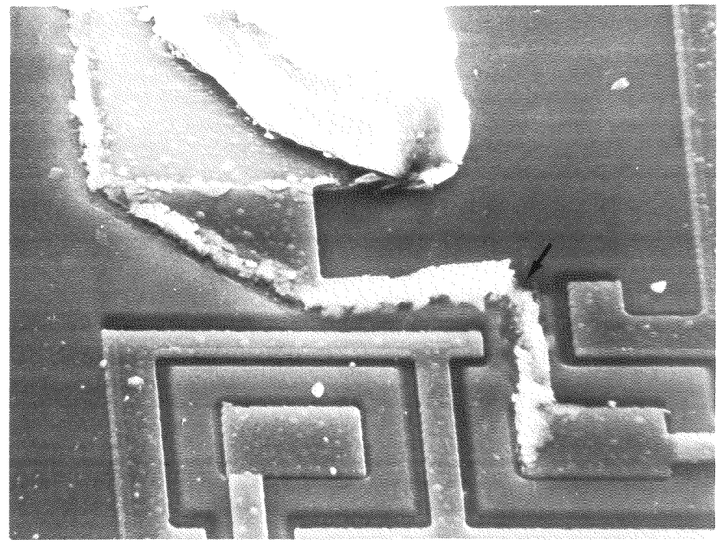


Fig. 14 – La foto (700 x) mostra un caso di corrosione su parte di un circuito integrato CMOS; la corrosione procede dal terminale di uscita e porta all'assottigliamento della pista di Alluminio fino all'interruzione indicata dalla freccia (larghezza tipica della pista 10 μm). [5]

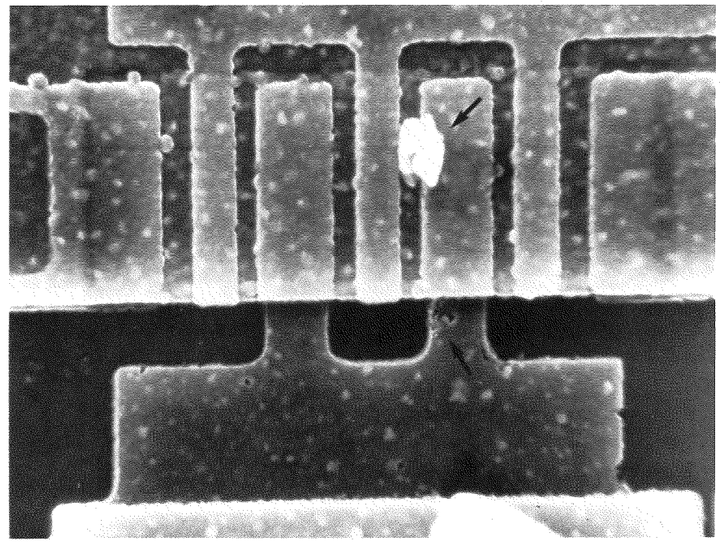


Fig. 15 – La foto (1400 x) presenta un caso di elettromigrazione in un circuito integrato ECL; l'eccessiva densità di corrente nel transistor di uscita porta alla creazione di lacune nella pista di metalizzazione in Alluminio all'inizio di un dito di emettitore (freccia in basso) ed all'accumulo di materiale all'altra estremità, fino al corto circuito con la sovrastante pista di collettore (freccia in alto). Larghezza delle piste da 5 a 20 μm. [5].

Bibliografia generale

1) Libri

[a] R.H. MYERS, K.L. WONG and H.M. GORDY, *"Reliability Engineering for Electronic Systems"*, John Wiley & Sons Inc, 1964

[b] M. L. SHOOMAN, *"Probabilistic Reliability: An Engineering Approach"*, Mc Graw-Hill Book Company, 1968

[c] W. G. IRESON, *"Reliability Handbook"*, Mc Graw-Hill Book Company, 1966

[d] W. HH. VON ALVEN, *"Reliability Engineering"*, ARING Research Corporation, Prentice Hall Inc, 1964

[e] R.E. BARLOW and F. PROSCHAN, *"Mathematical Theory of Reliability"*, John Wiley & Sons Inc, 1967

[f] R.E. BARLOW and F. PROSCHAN, *"Statistical Theory of Reliability and Life Testing"*, Holt Rinchart and Winston Inc, 1975

[g] R. MANN, E. SCHAFER and D. SINGPORWALLA, *"Methods for Statistical Analysis of Reliability and Life Data"*, John Wiley & Sons Inc, 1974

[h] K.C. KAPUR, L.R. LAMBERSON, *"Reliability in Engineering Design"*, John Wiley and Sons Inc, 1977

[i] P.D.T. O'CONNOR, *"Pratical Reliability Engineering"*, Heyden & Son Ltd, London, 1981

[l] J.E. ARSENAULT, J.A. ROBERTS (Editors), *"Reliability and Maintainability of Electronic Systems"*, PITMAN, London, 1980

[m] B.S. BLANCHARD and E.E. LOWERY, *"Maintainability: Principles and Practices"*, Mc Graw-Hill Book Company, 1969

[n] E. CARRADA, *"L'Affidabilità per l'elettronica"*, La Goliardica Editrice, 1975

[o] M. DECINA, *"Teoria dell'Affidabilità"*, La Goliardica Editrice, 1979

2) Convegni

Tra i più importanti si citano

- *"Annual Reliability and Maintainability Symposium"* (a partire dal 1954) - IEEE e Altri
- *"Annual Reliability Physics"*, (a partire dal 1962), IEEE e Altri

3) Riviste

- *IEEE Transactions on Reliability*
- *Microelectronics and Reliability*
- *Reliability Technology*

Riferimenti Bibliografici

[1] M.B. KLINE e AL., *"International Conference on Reliability and Maintainability"*, Paris, 1978 (tradotta con leggere modifiche)

[2] C. GORDON, Peattie e Al., Proceeding IEEE, 1978

[3] D.S. PECK, 1978 *Reliability Physics Symposium*

[4] Atti del Seminario su *"Affidabilità e fisica dei meccanismi di guasto nei semiconduttori"*, Lecce 1980

[5] Foto TELETTRA

[6] Rielaborazione da una proposta di A. Zaluadova al TC 56 dell'IEC

Strutture avanzate di elaborazione: le "data-flow machines"

GIANCARLO TESSERA

Honeywell Information Systems Italia
Centro di Ricerca e Progettazione
Pregnana Milanese

1. Introduzione

Il bisogno di elaboratori con architetture più adatte a sostenere i moderni linguaggi di programmazione nonchè la continua richiesta di una maggiore velocità di elaborazione, è sfociato nella realizzazione di sistemi in cui il controllo esecutivo delle istruzioni è affidato al "flusso dei dati" ("data-flow").

I sistemi data-flow, abbandonando il principio operativo ideato da Von Neumann nel 1946 ed applicato alla quasi totalità degli elaboratori commerciali sino ad ora prodotti, consentono di sfruttare totalmente ed automaticamente il parallelismo insito in un programma; unica condizione richiesta per l'esecuzione di una istruzione, è la sola presenza dei dati.

In questo articolo vengono descritte le caratteristiche operative fondamentali dei sistemi data-flow, ponendo in evidenza le principali differenze esistenti nei confronti degli elaboratori tradizionali.

2. La problematica

La necessità di disporre di elaboratori con potenza di calcolo sempre maggiore, ha portato alla realizzazione di sistemi operanti secondo schemi ed architetture di "tipo parallelo" [1].

L'esigenza di velocità di calcolo elevata è particolarmente sentita in alcune applicazioni quali la metereologia, il riconoscimento di forme (pattern recognition), di caratteri scritti, della voce, l'elaborazione dell'immagine e dei segnali.

L'elaborazione di tipo parallelo è caratterizzata dalla contemporanea esecuzione di più istruzioni appartenenti ad uno stesso programma. In genere, per consentire una elaborazione efficace di tipo parallelo, vengono adottati sistemi con architetture "multiprocessor", strutturati secondo una forma di parallelismo semplice (due o più unità centrali che operano su segmenti diversi dello stesso programma) o con strutture più complesse del tipo "array".

Solo da qualche anno, sotto l'impulso stimolante prodotto dai lavori di Hoare, Hansen, Dijkstra e Dennis, l'interesse verso l'elaborazione parallela ha acquisito crescente consenso.

Modelli architetturali e di programmazione, per vari aspetti del tutto nuovi, sono diventati argomento primario di studio e di analisi. La natura dei problemi che si presentano è tale da suggerire mutamenti abbastanza rivoluzionari di architetture hardware e software. Tuttavia si è fatto poco finora per la formulazione di appropriate procedure di progetto al fine di correlare i requisiti funzionali con le architetture di calcolo. In ogni caso, le nuove tendenze sia di costo che di tecnologia, sono tali per cui architetture innovative, per vari motivi improponibili fino a pochi anni fa, rappresentano ora una prospettiva attuabile.

Fra queste fioriture di nuove architetture, assumono particolare rilevanza per il modo del tutto peculiare di affrontare e risolvere il problema della elaborazione parallela, i sistemi conosciuti con il nome di "data-flow" o sistemi con controllo guidato dal "flusso dei dati" [2].

L'aspetto più rivoluzionario di questa architettura risiede nell'abbandono del concetto di esecuzione sequenziale di un programma (o di un suo segmento), concetto formulato da Von Neumann, Goldstine e Burks nel 1946 e applicato nella quasi totalità degli elaboratori commerciali fino ad ora prodotti.

L'architettura dei sistemi data-flow permette di adattare dinamicamente la struttura di macchina alla struttura della computazione da eseguire, permettendo quindi di esplicare il parallelismo in essa insito.

Mentre nell'architettura basata sullo schema ideato da Von Neumann, che può essere considerato del tipo a "controllo di flusso", lo svolgimento del lavoro è attuato mediante l'esecuzione sequenziale delle istruzioni, questi sistemi sono governati dal flusso dei dati. Questo significa che una certa operazione viene abilitata, ovvero che una istruzione viene eseguita, quando tutti gli operandi su cui questa opera sono "pronti"; gli operandi o dati, possono provenire dall'esterno del sistema (da unità periferiche di input) o da una precedente operazione eseguita dal sistema.

Poichè durante lo svolgimento di

un lavoro di computazione avviene in genere che più istruzioni sono contemporaneamente pronte per essere eseguite, gli elaboratori con architettura data-flow sono predisposti per parallelizzare l'esecuzione dei lavori e quindi in termini di produttività, ossia del lavoro svolto nella unità di tempo o "throughput", consentono di raggiungere brillanti risultati.

3. Differenze operative fra elaboratori a controllo di flusso ed elaboratori data-flow.

Gli elaboratori a controllo di flusso ed i rispettivi programmi (anche quelli capaci di sviluppare un certo grado di attività parallela), sono organizzati secondo una struttura di natura implicitamente sequenziale; comunque volendo procedere ad una elaborazione in forma parallela, richiedono una chiara esplicitazione delle fasi di programma eseguibili in modo concorrente.

I linguaggi convenzionali di programmazione, anche quelli definiti ad "alto livello", sono strutturati in accordo ad un modello computazionale che può essere considerato, per molti aspetti, un linguaggio a "basso livello". Ci si riferisce in particolare al modello di computazione definito del tipo processatore/memoria, in cui il processatore, o i processatori nel caso di elaboratori a struttura multiprocessor, eseguono operazioni sui dati contenuti nella memoria dell'elaboratore. La memoria è trattata come una risorsa comune condivisa dai programmi in corso di esecuzione; i dati in essa contenuti possono essere modificati secondo lo schema operativo definito nel programma stesso. Gli organi fondamentali che concorrono alla esecuzione di una operazione sono pertanto la memoria ed il processatore (chiamato anche CPU, central processor unit), connessi fra loro da un canale su cui transitano i dati e le istruzioni.

Pertanto in un elaboratore tradizionale a "controllo di flusso", le condizioni necessarie alla esecuzione di una istruzione sono: l'esistenza di almeno una cella di memoria a cui l'istruzione faccia riferimento e contenente il dato "variabile da usare"; la istruzione che opera una trasformazione sul dato in oggetto; le istruzioni di controllo che determinano l'ordine e la sequenza di esecuzione delle istruzioni stesse. Il controllo sequenziale delle operazioni è affidato al "program counter" che contiene l'indirizzo -o il puntatore- della istruzione da eseguire. Incrementando sequenzialmente il contenuto del "program counter", si perviene alla esecuzione in sequenza delle istruzioni costituenti il programma in oggetto.

Le istruzioni di controllo in genere alterano la normale esecuzione in sequenza delle istruzioni, modificando e variando il contenuto del "program counter" e quindi determinando un nuovo ordine di esecuzione delle istruzioni del programma.

La necessità di leggere dati dalla memoria e di riscriverli dopo averli operati come specificato dalle istruzioni, nonchè l'esigenza di dover leggere ed interpretare le istruzioni, fa si che una buona parte del traffico che transita sul collegamento tra memoria e CPU risulti, dal punto di vista dell'utilizzatore, completamente invisibile e quindi indifferente benchè questa parte del traffico sia indispensabile per il corretto funzionamento della computazione. In effetti, questo traffico crea una "strozzatura" nel sistema, la cui presenza condiziona le prestazioni del sistema stesso, i linguaggi di programmazione oggi usati, e infine il modo di inquadrare algoritmicamente i problemi [3].

Da un punto di vista concettuale, un programma viene scritto tenendo presente che, dopo l'esecuzione di una istruzione, usata per controllare il flusso delle istruzioni o semplicemente per operare sui dati, si procede alla esecuzione di una istruzione successiva, univocamente scelta in base alla logica di controllo contenuta nel programma stesso. Ciò consente di evitare computazioni indeterminate qualora più istruzioni appartenenti a programmi "concorrenti" condividano dati contenuti nelle stesse celle di memoria o quando risultati parziali modificano alcuni dati depositati nella memoria.

Però, come già detto, la necessità di aumentare la velocità di esecuzione di una computazione ed anche il tentativo di ottimizzare l'uso delle risorse di un elaboratore (spazio di memoria, unità periferiche, CPU, ecc.), ha determinato una spinta verso la ricerca e la definizione, nell'ambito di un programma, di "processi" o "task" che possono essere eseguiti in parallelo.

Inoltre le crescenti possibilità offerte dalla evoluzione tecnologica, ha reso disponibili elaboratori costituiti da una pluralità di CPU dotate di caratteristiche analoghe, operanti su di un'unica memoria (sistemi multiprocessor) e capaci di eseguire indipendentemente ed in parallelo processi o task diversi.

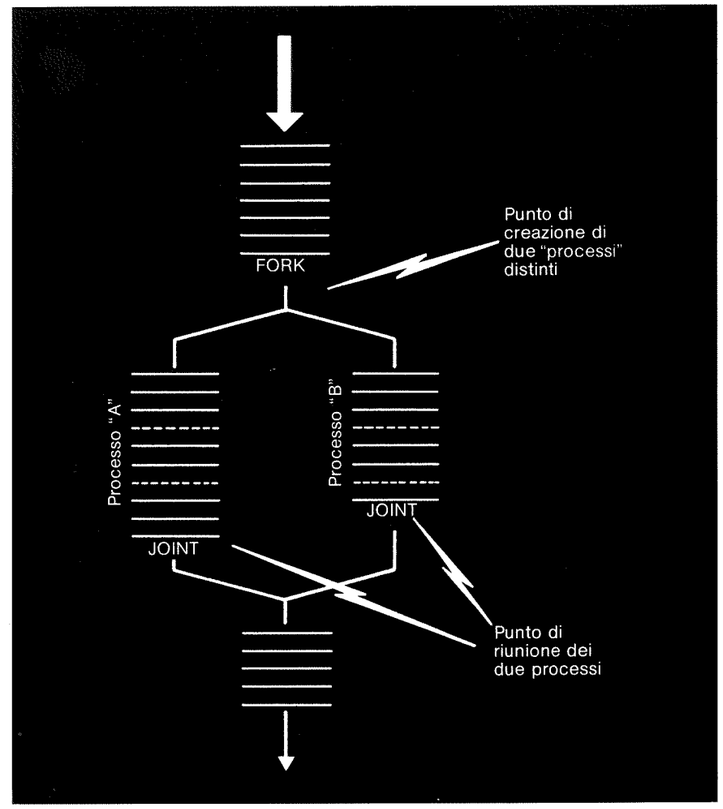
Un "processo" è da vedersi come una unità di computazione indivisibile, in contesa con altri processi per l'acquisizione delle risorse indispensabili alla conduzione dell'elaborazione (es. memoria) e con necessità di scambiare con essi messaggi (dati).

I linguaggi di programmazione devono ovviamente essere arricchiti con speciali costrutti per indicare l'inizio e la fine di più "task" concorrenti (es. mediante istruzioni di sincronizzazione quali la "Fork" e la "Joint"; vedi fig. 1). Sfortunatamente la necessità di indicare in qualche modo le sezioni di programma eseguibili in parallelo è causa dei seguenti inconvenienti:

- il lavoro di programmazione viene appesantito da una sovrastruttura di controllo fastidiosa e totalmente slegata dalla definizione dell'algoritmo di soluzione del problema.
- il programma risultante è di difficile lettura e quindi di difficile manutenzione.

Il sistema hardware e software deve essere complicato introducendo specifici meccanismi di protezione per evitare che un task interferisca in modo non voluto con il corretto svolgimento di un altro. L'insieme di tutte queste cause si riflette di fatto sulla velocità di esecuzione della computazione che può risultare pertanto insufficiente a fronte

Fig. 1 - Segmento di programma con due processi paralleli. L'istruzione "Fork" determina la formazione di due processi paralleli mentre le istruzioni "Joint" costituiscono il punto di riunione e confluenza della computazione in un unico processo.



delle crescenti esigenze di nuove applicazioni.

Gran parte dei problemi prima evidenziati, possono essere risolti mediante un elaboratore organizzato secondo una architettura "data-flow".

Come menzionato precedentemente, in questi sistemi la sequenza di controllo, in base a cui eseguire le istruzioni del programma, dipende solo dal flusso dei dati: una istruzione è abilitata per la esecuzione, "è processata", quando tutti i suoi operandi sono pronti, cioè le unità periferiche di ingresso-uscita hanno reso disponibili i dati al programma o questi sono stati ottenuti da altre operazioni precedentemente eseguite nel programma stesso.

L'elaborazione dei dati effettuata secondo il sistema "data-flow", comportando un controllo di flusso del programma di tipo implicito e distribuito a livello delle singole istruzioni, elimina la necessità di distinguere tra parallelismo "ad alto livello" (processi) e parallelismo "a basso livello" (istruzioni). Risultano dunque semplificate notevolmente anche quelle tecniche che sono in genere applicate nella realizzazione del nucleo del sistema operativo al fine di permettere la multiprogrammazione. Ogni istruzione porta esplicitamente con sè tutte le informazioni necessarie per essere posta in esecuzione e non occorre dunque effettuare operazioni supplementari per passare dalla esecuzione di un programma ad un altro.

Anche la semantica dei linguaggi ad alto livello può risultare notevolmente semplificata dalla introduzione del concetto di "data-flow" e ciò indubbiamente può contribuire alla riduzione dei costi relativi alla produzione del software, specialmente nei sistemi multiprocessor.

Per raggiungere pienamente gli obiettivi indicati, è di estrema importanza adottare una metodologia di progetto che tenga in debita considerazione la necessità di procedere al disegno del sistema in forma integrata sia dal punto di vista dello hardware che del software. In questo concetto rientra l'idea di definire innanzitutto un linguaggio ad alto livello, il linguaggio base di macchina, una adeguata struttura di protezione per il software dello utilizzatore, un sistema di gestione delle eccezioni di programmazione ad alto livello; il tutto dovrebbe essere correlato con i concetti semantici espressi nella computazione data-flow. Risulta quindi evidente che per addivenire ad una progettazione integrata hardware e software occorre fondere l'architettura del linguaggio con quella del sistema in modo omogeneo. Ciò porta al fatto che per sfruttare appieno i benefici derivanti da nuove impostazioni architetturali, diventa indispensabile disporre di un software nuovo. Per software nuovo intendiamo sia programmi scritti in un linguaggio ad alto livello adatto, sia strutture di macchina basate sui concetti informatori prima esposti.

4. Un mezzo espressivo più appropriato per la descrizione del problema.

Una metodologia che permette una razionale rappresentazione di un problema in accordo ai concetti data-flow, la ritroviamo in una vecchia idea matematica conosciuta con il nome di "grafo orientato" [4]. Il "grafo" permette di rappresentare in forma grafica le relazioni o l'assenza di relazioni che intercorrono fra gli elementi appartenenti ad un certo gruppo, nel nostro caso i dati di un programma. In un grafo orientato, i "nodi" corrispondono ad operatori funzionali e gli "archi" a canali lungo i quali vengono trasmessi i valori dei dati scambiati tra operatori. Il grafo è chiamato "diretto" in quanto il movimento dei dati fra nodo e nodo avviene in un'unica direzione lungo l'arco di congiunzione dei nodi.

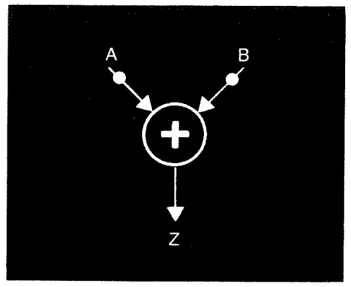
Le "reti di Petri" [5], tecnica sviluppata per concettualizzare i sistemi di logica dinamica e derivata dalla teoria dei grafi, con le proprietà di parallelismo, non determinismo e atemporalità, costituiscono uno strumento adatto a modellare diversi tipi di problemi rappresentabili secondo i concetti di data-flow.

Con questa tecnica è possibile descrivere sistemi all'interno dei quali coesistono o possono coesistere simultaneamente condizioni "statiche" e "dinamiche". Dinamicamente, l'esecuzione di un programma secondo questa rappresentazione, è da vedersi come un seguito di attivazioni degli operatori che abbiano i dati presenti sugli archi ad essi afferenti, con il conseguente assorbimento di un dato da ogni arco d'ingresso e l'emissione di un dato su ogni arco di uscita. L'attività di un operatore è quindi regolata dai dati in modo del tutto asincrono.

Consideriamo come esempio la seguente espressione:

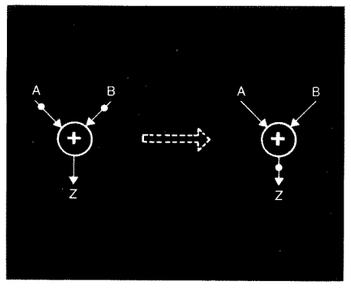
$$Z = A+B$$

Tradotta in forma di grafo assume la seguente struttura:



dove: il cerchio rappresenta un "operatore funzionale" capace di eseguire (in questo caso) l'operazione di somma ("+"); i due archi afferenti al cerchio portano rispettivamente i dati "A" e "B" rappresentati mediante due punti neri (i dati sono chiamati anche "tokens"); l'arco in uscita dal cerchio serve per convogliare ad altra destinazione il dato o risultato prodotto dall'operatore funzionale.

Il meccanismo di esecuzione della computazione, è il seguente: non appena i due dati sono presenti sugli archi d'ingresso di un operatore (rappresentati visivamente mediante "punti"), l'operatore può essere attivato e quindi eseguire la funzione che rappresenta. Questo processo comporta l'assorbimento dei dati in ingresso (i punti vengono assorbiti dall'operatore) e la generazione di un dato in uscita, che costituisce il risultato, sempre visualizzato mediante un punto.

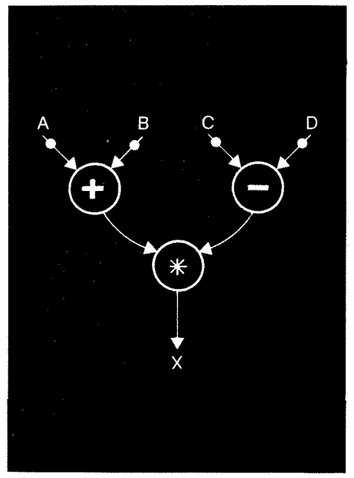


La rappresentazione di un problema secondo uno schema "data-flow", consiste nella definizione e nella interconnessione di quegli operatori funzionali richiesti per la soluzione del problema.

Per esempio volendo rappresentare la funzione:

$$X = (A + B) * (C - D)$$

basta comporre fra loro in modo semplice i tre seguenti operatori:



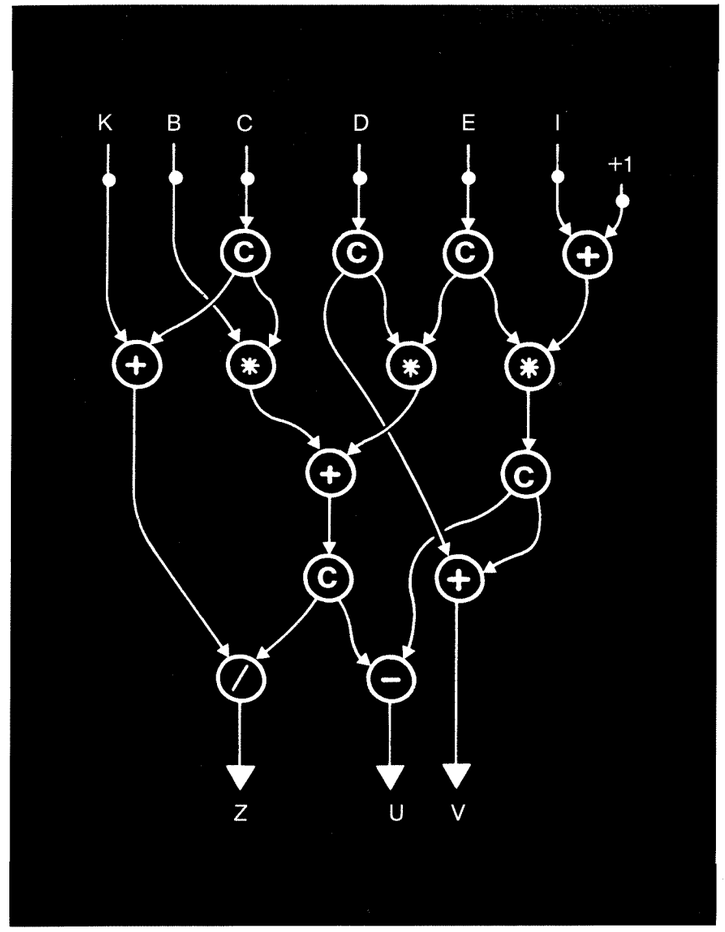
I principi operativi che informano un'attività data-flow, sono i seguenti:

1. Un operatore viene attivato, dando quindi luogo alla esecuzione di una funzione, quando e solo quando sono disponibili, ossia presenti, gli operandi (o l'operando).

2. Le operazioni sono considerate delle "funzioni", cioè non danno luogo a "effetti collaterali" (cioè non esplicitamente previsti).

In una ipotesi più restrittiva di funzionamento e per eliminare un problema di accumulo di dati su uno stesso arco, situazione che può verificarsi in programmi di natura interattiva, un operatore funzionale viene abilitato ad operare solo quando, oltrechè disporre dei dati sugli archi d'ingresso, detto operatore non contiene dati sull'arco di uscita. Cioè non si ammette la formazione di "code" di dati.

Fig. 2 - Rappresentazione data-flow del seguente segmento di programma:
A : = (B*C) + (D*E)
I : = I + 1
U : = A − (E*I)
V : = D + (E*I)
I : = K + C
Z : = I / A



○	operatore funzionale
*	moltiplicatore
+	addizionatore
/	divisore
-	sottrattore
Ⓒ	operatore di "copy"; produce due o più copie di uno stesso dato
+ 1	dato o funzione costante f(x)=1
—●	dato, detto anche "Token"
—→	arco su cui viaggia un dato

Da quanto detto, risulta evidente che l'ordine di esecuzione di due funzioni, ad esempio la somma e la sottrazione del precedente grafo, è irrilevante dal momento che le computazioni non presentano alcuna interazione fra loro. Quindi, nel nostro esempio, le due operazioni possono anche essere eseguite in "parallelo", cioè contemporaneamente.

Le interazioni tra i vari operatori di un grafo si riducono quindi al solo scambio di dati sull'arco che collega l'uscita di uno ad un ingresso dell'altro operatore e la esecuzione di un programma è da vedersi come un flusso di dati sugli archi del grafo e nelle attivazioni e disattivazioni degli operatori.

Nella figura 2 è riportata la rappresentazione data-flow del segmento di programma indicato di seguito:

```
A : = (B * C) + (D * E)
I : = I + 1
U : = A − (E * I)
V : = D + (E * I)
I : = K + C
Z : = I / A
```

Gli elementi del linguaggio utilizzati in questo esempio sono riassunti nella tabella allegata alla fig. 2.

Nella figura 3 è rappresentata la sequenza dinamica con cui avviene la computazione del programma riportato in figura 2.

L'analisi di un problema scritto secondo la tecnica del "grafo orientato", rispetto ad una scrittura di tipo tradizionale, permette di derivare molte informazioni relative alle caratteristiche del programma stesso.

Per esempio è noto che il tempo di esecuzione di un programma è strettamente legato alla lunghezza del "ramo critico" esistente in quel programma e ciò indipendentemente dal numero di "processor" usati nell'eseguire la computazione. Mentre la individuazione del "ramo critico" risulta molto semplice in un programma rappresentato mediante grafo, non si può altrettanto dire per un programma scritto secondo una rappresentazione convenzionale.

Altre utili informazioni che possono essere facilmente derivate dall'analisi di un problema rappresentato con un grafo orientato sono:

- eventuali strozzature (bottlenecks) esistenti nel programma: minimo numero di nodi (operatori funzionali) che si incontrano allo stesso livello del programma.

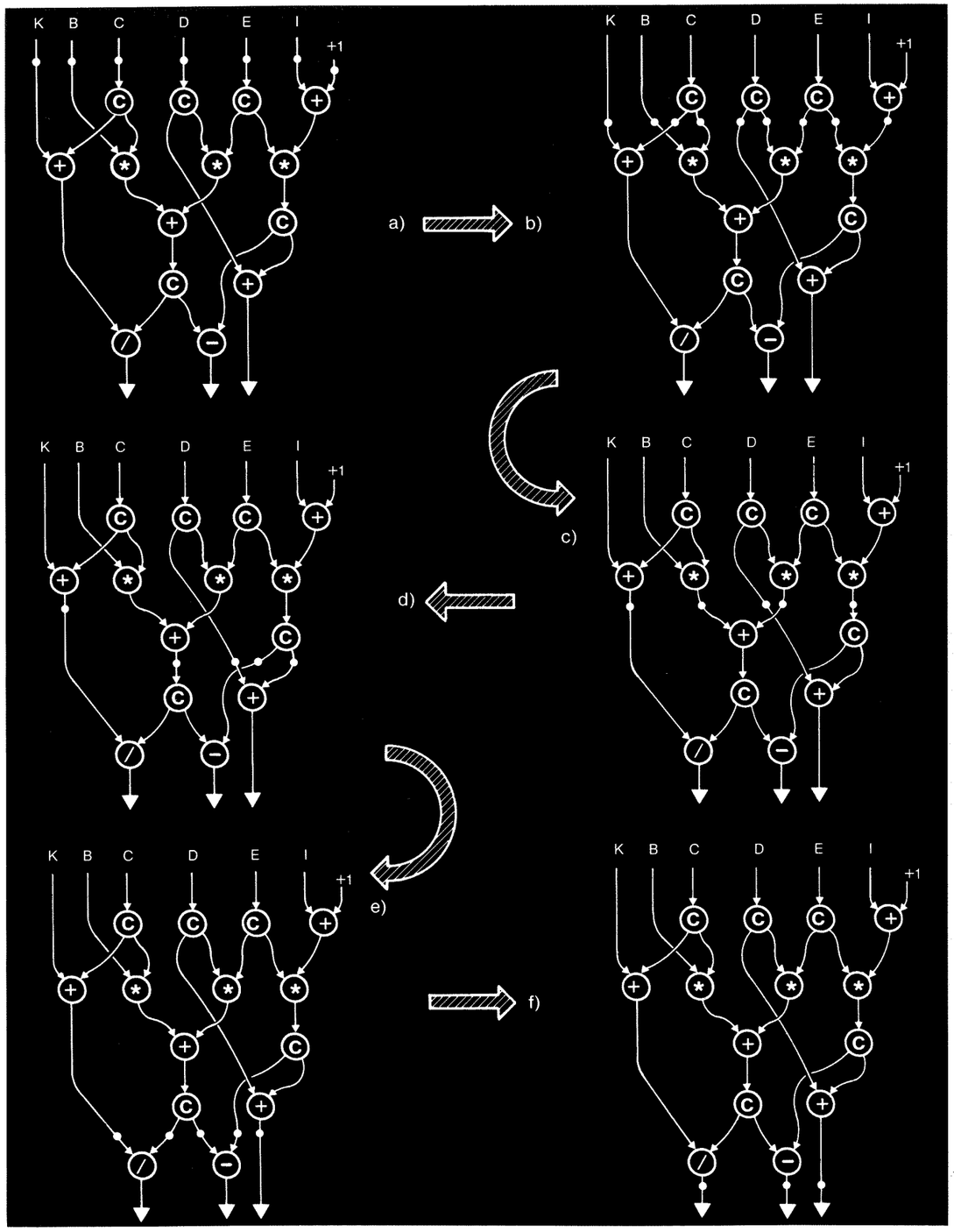


Fig. 3 - Meccanismo di esecuzione di una computazione mediante un sistema data-flow. I dati, contrassegnati con punti neri, transitano sugli archi del grafo secondo la sequenza operativa indicata nelle figure contraddistinte con le lettere a, b, c, d, e, f.

Inizialmente i dati sono presenti sugli archi d'ingresso agli operatori logici del 1° livello (fig. a) e dopo 5 fasi di calcolo si perviene alla generazione dei risultati (fig. f).

- parallelismo medio della computazione: definito come numero totale di nodi diviso per la lunghezza del "ramo critico".
- parallelismo massimo riscontrabile nella computazione; definito come il massimo numero di nodi (operatori funzionali) che si riscontrano nel diagramma ad uno stesso livello.

A titolo d'esempio supponiamo di dover risolvere il seguente sistema di equazioni:

$$U = (A*D)+(B*C)+A$$
$$T = (A*D)+(B*C)+(A*B)+(C*D)$$
$$V = (A*B)+(C*D)-C$$
$$R = (A*D)+(B*C)*V$$

Riscrivendo in forma più adeguata per la conversione in programma si ha:

$$P_0 = A * D$$
$$P_1 = B * C$$
$$P_2 = A * B$$
$$P_3 = C * D$$
$$P_4 = P_0 + P_1$$
$$P_5 = P_2 + P_3$$
$$U = P_4 + A$$
$$T = P_4 + P_5$$
$$V = P_5 - C$$
$$R = P_4 * V$$

La scrittura di questo programma secondo una struttura a grafo è riportata in fig. 4.

Per semplicità, dalla figura sono stati omessi gli operatori © di norma usati per eseguire "copie" plurime di dati.

Di seguito sono riassunte alcune informazioni facilmente desumibili da questo tipo di rappresentazione; ricavare le stesse informazioni da un programma rappresentato secondo una forma convenzionale di scrittura, risulta certamente più difficoltoso ed in qualche caso addirittura impossibile.

- Numero totale delle istruzioni : 10
- Lunghezza del ramo critico: 4
- Punto di strozzatura della computazione (parallelismo minimo) : 1
- Parallelismo massimo della computazione : 4
- Parallelismo medio della computazione : 2,5

Con riferimento all'esecuzione del programma, una ragionevole possibilità di parzializzazione della computazione potrebbe essere indicata nella fig. 4 mediante la linea spezzata. Gli operatori (istruzioni) a destra potrebbero essere eseguiti da un "processor" mentre un secondo "processor", operando in parallelo, potrebbe eseguire quelle di sinistra. Secondo questa ipotesi di partizione, i dati scambiati fra i due "processor" (indicati dal numero degli

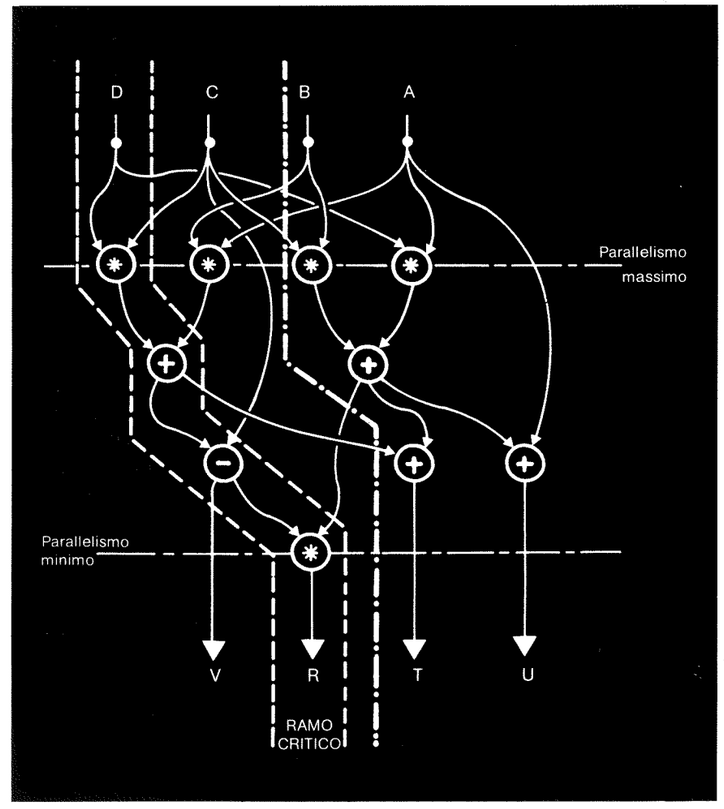


Fig. 4 - Un programma rappresentato secondo una struttura a grafo, permette di ricavare molteplici informazioni relative alla computazione da eseguire: parallelismo massimo, medio e minimo, lunghezza ramo critico, totale istruzioni, possibile ripartizione della computazione.

archi che attraversano la linea di separazione) sono 6 ed il tempo richiesto per l'esecuzione di questo programma su di un sistema bi-processor data-flow, potrebbe ridursi alla metà del tempo necessario per l'esecuzione dello stesso programma in un sistema tradizionale.

5. Architettura hardware di un sistema data-flow

Fra le diverse proposte di sistemi con architettura data-flow, alcune ormai in fase di avanzata sperimentazione, risulta particolarmente interessante la realizzazione sviluppata dal CERT di Tolosa [6].

Questo elaboratore, a struttura multiprocessor, opera utilizzando un linguaggio speciale ad alto livello detto LAU (dal francese Langage à Assignation Unique).

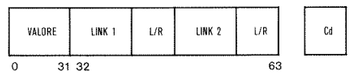
Gli elementi hardware principali che caratterizzano e contraddistinguono una architettura data-flow, sono:

1. Blocco di logica per l'identificazione delle istruzioni pronte all'esecuzione.
2. Sottosistema di memoria usato per registrare il programma e i dati su cui questo opera. Il blocco logico di memoria include una rete che provvede alla creazione e alla gestione della coda di istruzioni pronte per essere eseguite.
3. Unità di calcolo; può essere costituita da una sola CPU o da una pluralità di "processor" di tipo generalizzato connessi secondo uno schema di tipo simmetrico a cui vengono assegnate le istruzioni da eseguire. Secondo una realizzazione preferenziale, normalmente le CPU sono implementate con microprocessori fra loro connessi mediante una struttura a bus di tipo semplice.
4. Sottosistema per il controllo delle unità periferiche usate per l'ingresso e l'uscita dei dati dal sistema.

Le istruzioni usate dal sistema LAU, di lunghezza 64 + 3 bit, hanno il seguente formato:

- il primo campo contiene il codice operativo della istruzione cioè il tipo di operazione da eseguire (operatore funzionale secondo il linguaggio data-flow);
- il secondo campo esprime l'indirizzo di memoria a cui registrare il risultato prodotto dall'operazione;
- il terzo ed il quarto campo esprimono l'indirizzo di memoria in cui sono registrati i due operandi o dati su cui eseguire l'operazione;
- i bit C_0, C_1 e C_2 sono usati per definire e decidere se una istruzione è pronta o meno per essere eseguita. Precisamente C_1 e C_2 servono per indicare la presenza dei due operandi (dati pronti) e C_0, tenendo conto di alcune condizioni esterne, indica quando l'istruzione è pronta per essere eseguita.

Il formato di un operando o dato, è il seguente:



- 32 bit esprimono il valore del dato;
- LINK 1 e LINK 2 contengono l'indirizzo delle istruzioni che usano il dato in oggetto;
- C_d è un indicatore usato per notificare l'avvenuta assegnazione di un valore significativo al dato stesso (controllo di assegnazione singola).

I blocchi logici elementari di cui risulta costituito il sistema LAU, sono riportati in figura 5.

L'unità di controllo per l'esecuzione delle istruzioni, ispezionando i bit C_0, C_1 e C_2 di condizione, identifica le istruzioni pronte per essere poste in esecuzione (i tre bit devono avere valore 1). Queste vengono inserite in una coda FIFO (primo arrivato primo servito) all'interno del blocco di memoria.

Una rete di "distribuzione delle istruzioni", provvede ad assegnare ciascuna istruzione pronta ad un "processor" libero.

Un ciclo base di calcolo consiste:

- nella lettura dalla memoria dell'istruzione e dei valori degli operandi;
- nella scrittura in memoria del valore del risultato prodotto dall'unità di calcolo;
- nell'aggiornamento dei bit "C" di condizione associati a quelle istruzioni che useranno il risultato appena calcolato;
- nell'aggiornamento del bit C_d relativo al risultato.

Non appena un ciclo è stato completato, si procede alla esecuzione di un altro ciclo.

Poichè la sezione di calcolo può ospitare fino ad un massimo di 32 CPU, normalmente più istruzioni saranno contemporaneamente in fase di esecuzione.

Nel sistema LAU i processor hanno un parallelismo operativo di 16 bit e sono realizzati con microprocessori della famiglia AMD 2900; tempo base di ciclo 200 nsec.

Complessivamente il sistema LAU consta di 49 schede (nella versione con 32 CPU) per un totale di 11.900 circuiti integrati.

Attualmente il gruppo di ricerca di Tolosa sta conducendo su questo sistema un'attività sperimentale di programmazione basata sull'uso di linguaggi speciali al fine di misurare le prestazioni ottenibili da un sistema operante sulla base del concetto data-flow.

6. Conclusione

Le "data-flow machines" costituiscono un vero e proprio salto concettuale nella architettura dei sistemi di elaborazione, e presentano significativi vantaggi sulle strutture convenzionali.

Attualmente esistono solo prototipi di laboratorio per scopi di ricerca.

Circa il loro sviluppo pratico, occorre tener conto che se da un lato il rapido progresso della tecnologia circuitale (VLSI) offre prospettive favorevoli per quanto riguarda l'aspetto hardware, dall'altro la necessità di un nuovo software costituisce una grossa remora per una introduzione nell'uso corrente.

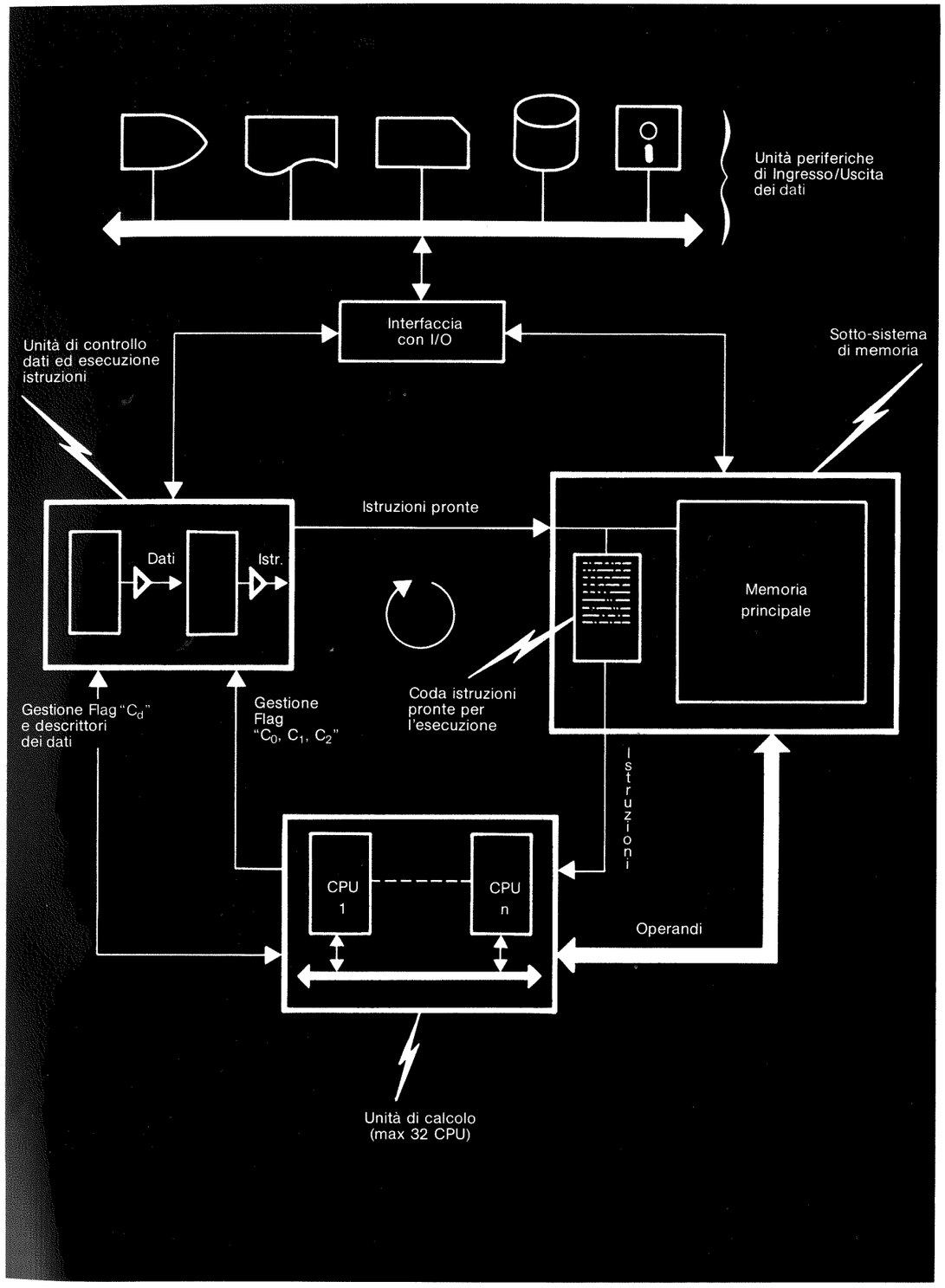


Fig. 5 - Schema a blocchi dell'elaboratore data-flow "LAU", sviluppato dai ricercatori del CERT di Tolosa.

7. Bibliografia

[1] LORIN H.; *"Parallelism in hardware and software"*; Prentice-Hall Inc.; 1972.

[2] DENNIS J.B., D.P. MISUNAS; *"A preliminary architecture for a basic data-flow processor"*; Second Annual Symposium on Computer Architecture - Conference Proceedings; Jan. 1975; pag. 126 ÷ 132.

[3] BACKUS J.; *"Can programming be liberated from the Von Neumann style? A functional style and its algebra of programs"*; Communication of ACM; August 1978; pag. 613 + 641.

[4] STIGALL P.D., O. TASAR; *"A review of directed graphs as applied to computers"*; Computer; Oct. 1974; pag. 39 + 47.

[5] PETERSON; *"Petri nets"*; Computing surveys; vol. 9, Sept. 1977; pag. 223 + 252.

[6] COMTE D., N. HIFDI, J. C. SYRE; "*LAU multiprocessor: microfuntional description and technological choices"*; First European Conference on parallel and distributed processing; Toulose, France, 1979; pag. 8 + 15.

[7] J.B. DENNIS; *"Data-flow supercomputers"*; Computer; Nov. 1980.

Nota - Questo articolo compare contemporaneamente su *"Informatica Oggi"*.

Un modello di automazione dell'ufficio

A. CICU
Honeywell Information Systems Italia
Pregnana Milanese

M. GUALZETTI
ETNOTEAM, Milano

R. POLILLO
Università di Milano, Istituto di Cibernetica

1. Introduzione

La presente nota descrive sinteticamente alcuni risultati di un progetto di ricerca volto alla definizione dei requisiti di un sistema per l'automazione integrata delle attività di ufficio, supportabile da un elaboratore di piccole-medie dimensioni.

Dopo una presentazione sommaria della collocazione funzionale della stazione di lavoro per l'operatore di ufficio, della struttura a grandi blocchi del sistema e dei requisiti dell'interfaccia utente, vengono tratteggiati i concetti e le funzioni principali del sistema studiato: la struttura degli oggetti trattati dalla stazione di lavoro, i concetti di archivio e di mailbox, le classi principali di funzioni.

2. La stazione di lavoro

Per definire un sistema per l'automazione delle attività di ufficio, risulta utile prendere le mosse dall'esame di una "stazione di lavoro" tradizionale (la scrivania dell'ufficio), per individuarne le possibili differenti configurazioni, gli strumenti abitualmente disponibili, le funzioni coinvolte, le interrelazioni fra le varie attività, le esigenze di automatizzazione (fig. 1).

Una stazione di lavoro automatizzata (fig. 2) dovrà consentire di effettuare elettronicamente (ovvero con il minimo movimento cartaceo e con i minimi interventi delle persone di ufficio) le stesse operazioni, o il loro equivalente, prima effettuate a mano, a voce o comunque con macchine senza memoria elettronica (creazione e/o aggiornamento e/o trasformazione, comunicazione, archiviazione, reperimento, riproduzione di informazione scritta o vocale) [1].

3. L'ambiente della stazione di lavoro

Alcune attività vengono svolte localmente alla stazione di lavoro (i dati e gli strumenti di elaborazione sono disponibili presso la stazione di lavoro stessa), altre richiedono interrelazioni che coinvolgono complessivamente tutta la struttura dell'ufficio, o l'ambiente esterno.

Un possibile modo di schematizzare tale complessa situazione è riportato nella figura 3, dove vengono mostrati i diversi sottosistemi in cui può essere pensata immersa ogni stazione di lavoro.

Componenti di ciò che è stato indicato, nella fig. 3a, come *sottosistema di ufficio,* possono essere considerati i seguenti:

- *comunicazioni interne*
  (gestione della corrispondenza e delle comunicazioni interne; comunicazioni informali; ecc.);
- *archivi d'ufficio*
  (archivi della documentazione d'ufficio, accessibili a più persone; comprende cataloghi, indirizzari, guide telefoniche; archivio personale);
- *procedure d'ufficio*
  (l'insieme delle norme, formalizzate o semi-formalizzate, che regolamentano le varie attività ed i flussi di informazioni).

In figura 3b si è evidenziato, come entità a sè stante, il sottosistema di *data processing,* per indicare quel complesso di attività e di informazioni che, nell'ambito dell'ufficio tradizionale, vengono attualmente gestite dai sistemi EDP e che nel modello studiato sono rese disponibili all'ufficio secondo opportuni criteri di pertinenza.

Con *ambiente esterno* (fig. 3c), si è voluto indicare, genericamente, il sistema in cui l'ufficio è immerso (l'azienda di cui l'ufficio fa parte, le procedure organizzative a livello aziendale, le norme di legge, la posta con enti esterni, gli archivi pubblici di informazioni, ecc.).

4. Struttura del sistema

Partendo dalle considerazioni precedenti, e pensando all'utilizzo di un elaboratore centrale (di piccole-medie dimensioni) per la gestione di un complesso di stazioni di lavoro in qualche modo interrelate, si può pensare ad un'architettura generale del tipo mostrato in Fig. 4.

Si sottolinea che si pensa ad un elaboratore in generale già utilizzato per procedure EDP.

In Fig. 4 possiamo immaginare che tutti i sottosistemi principali (dp, office, work station plug, communication e central printing service) siano in comunicazione fra di loro.

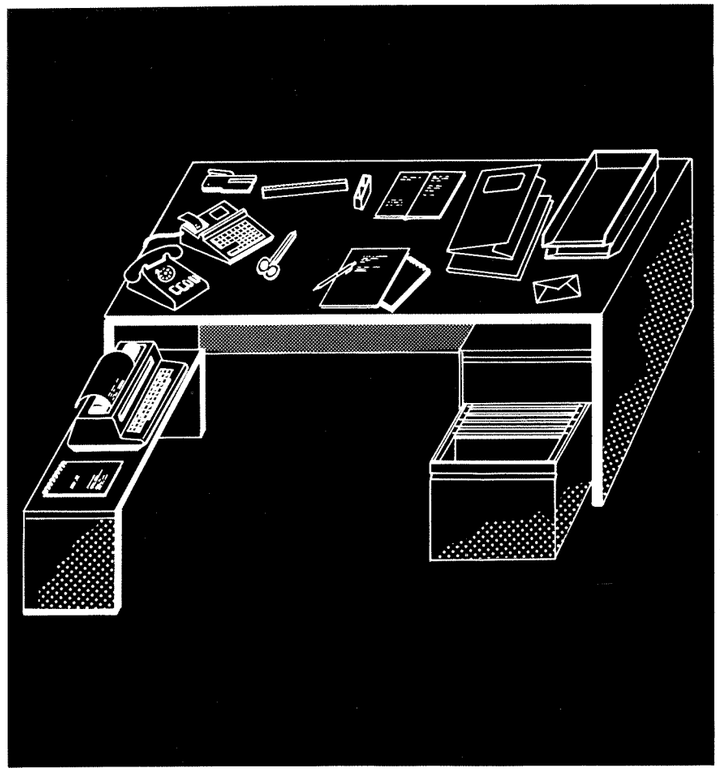


Fig. 1 - La stazione di lavoro tradizionale.

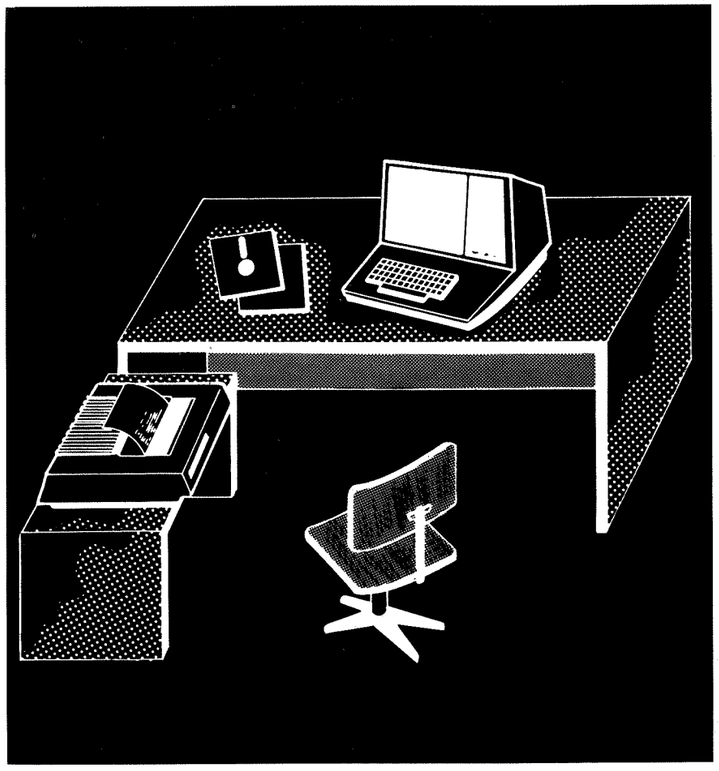
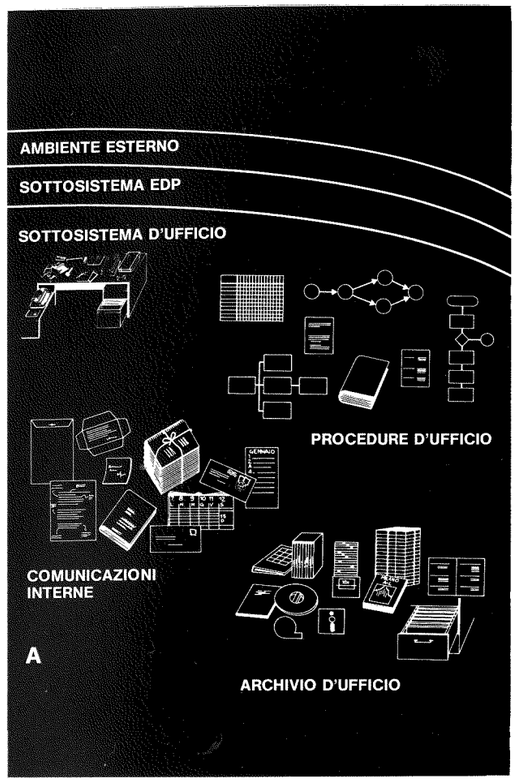


Fig. 2 - La stazione di lavoro automatizzata.



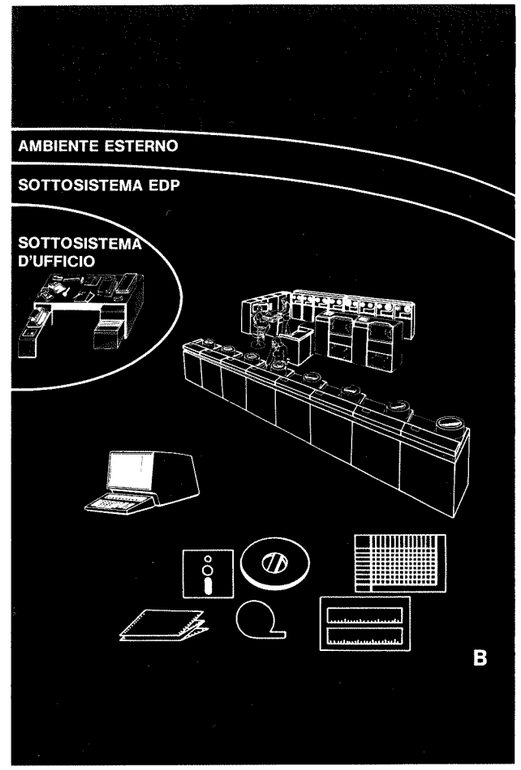
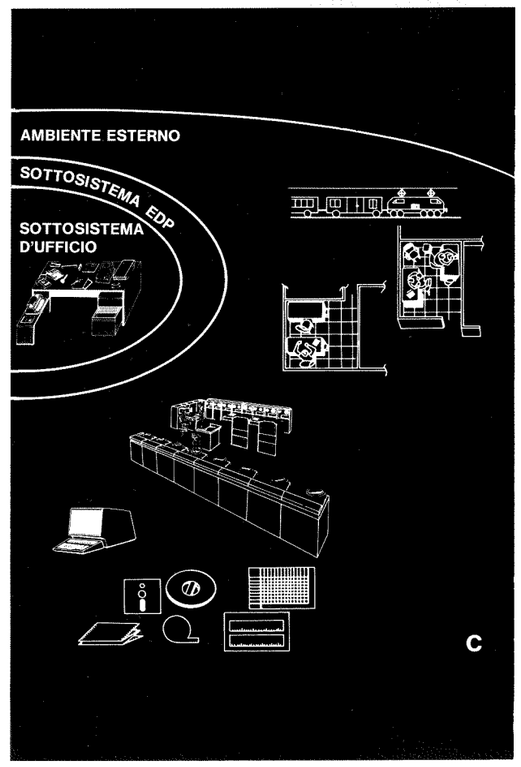


Fig. 3 - Queste figure schematizzano la complessa situazione in cui può essere pensato immerso ogni posto di lavoro.

Sono individuati tre sottosistemi: il sottosistema di ufficio (fig. 3a), il sottosistema EDP (fig. 3b), l'ambiente esterno (fig. 3c).



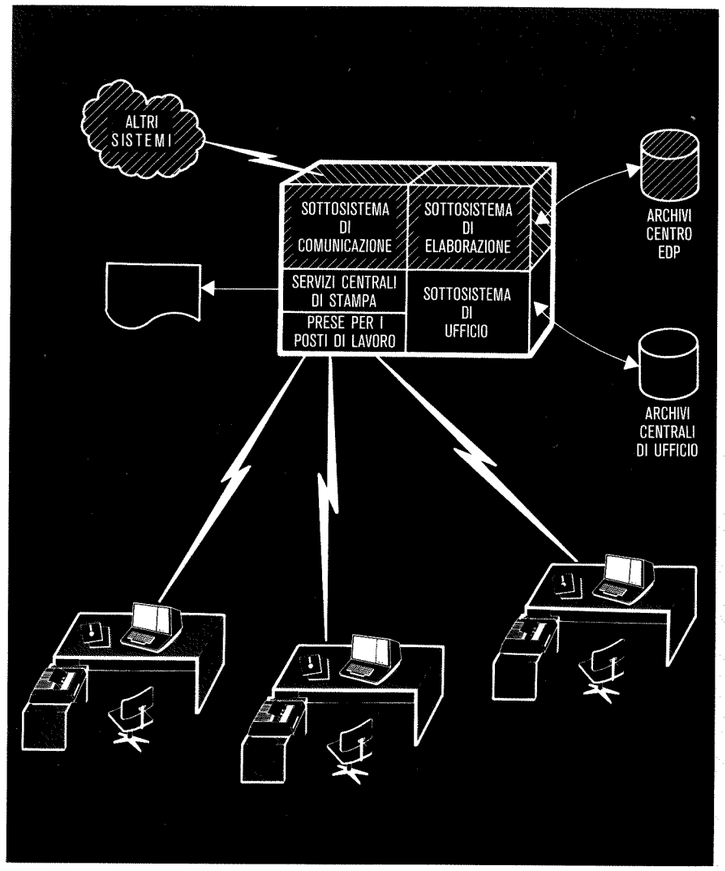


Fig. 4 - La struttura del sistema per la gestione delle attività d'ufficio.

5. L'interfaccia utente

L'interfaccia con l'utente al terminale deve soddisfare stringenti requisiti di semplicità e flessibilità operativa, fra cui:

- dialogo guidato (menu e prompt autoesplicativi, costante evidenziazione del contesto operativo, segnalazioni di errore efficaci, funzioni di help,....);
- massima flessibilità nel dialogo e nell'accesso all'informazione (l'operatore deve essere in grado di cambiare il contesto operativo ad ogni istante);
- linguaggio di comandi ridotto al minimo (ampio uso di tasti funzione, uso del cursore per operare selezioni,...);
- possibilità di utilizzare il sistema in toto o solo in alcune funzionalità base (subsetting in gruppi di funzioni "autonome", training base ridotto al minimo, uso di concetti e nomenclatura abituale nell'ambiente di ufficio,...);
- possibilità di personalizzazione dell'interfaccia con l'utente;
- "robustezza" dell'interfaccia.

Nel seguito verranno tratteggiati i concetti e le funzioni principali del sistema di automazione di ufficio studiato.

6. Gli oggetti dell'ufficio

Tutti gli oggetti maneggiati nel lavoro di ufficio sono costituiti, in varia forma, da informazione scritta.

Essi si differenziano tra loro per struttura, per modalità di costruzione, per modalità di uso.

Ciò che accomuna gli oggetti dell'ufficio tuttavia è la loro natura testuale, che consente, in larga misura:

- l'uso di strumenti identici per la loro costruzione;
- l'impiego di criteri informativi omogenei per la loro archiviazione e reperimento.

La struttura di detti oggetti può assumere varie forme poichè le regole di composizione non sono rigide: un foglio di appunti può diventare una lettera, più fogli possono essere "graffettati" insieme per comporre un documento, reports e liste possono divenire parte di un documento, ecc.

Un'analisi della struttura e delle modalità di uso degli oggetti dell'ufficio ha portato ad identificare cinque classi differenti di oggetti (fig. 5).

a) *Documenti*

Sono testi, considerati non strutturati, corredati da un certo insieme di "attributi" atti a facilitarne l'identificazione, la classificazione/archiviazione, il reperimento e l'accesso alle informazioni contenute.

Gli attributi principali sono: il nome del documento, la sua classificazione, il livello di riservatezza, la versione, l'autore la firma, il titolo, i commenti/annotazioni aggiuntive, le parole chiave, l'indice, le informazioni sulle dimensioni e sul formato di stampa (numero di pagine, linee per pagina, ecc.).

Non tutti tali attributi sono obbligatori; ove possibile, vengono compilati automaticamente dal sistema (ad esempio, l'identificatore che individua univocamente ogni oggetto nel sistema).

b) *Lettere*

Sono testi che posseggono attributi specifici: il mittente (al posto dell'autore del documento), l'oggetto (al posto del titolo), l'elenco dei destinatari (che può essere fornito anche assegnando il nome di una lista di distribuzione), alcuni dei quali possono essere "per conoscenza", il "riferimento Ns/Vs lettera", il livello di riservatezza, l'elenco degli eventuali documenti allegati, la data e l'ora di spedizione, ecc.

L'operatore deve obbligatoriamente compilare soltanto i campi mittente e destinatario/i. Il sistema compila automaticamente la data e l'ora di spedizione.

Si noti che queste lettere sono oggetti "formali": corrispondono alla corrispondenza formale di un ufficio tradizionale, differenziandosi, per questo, dai messaggi.

c) *Messaggi*

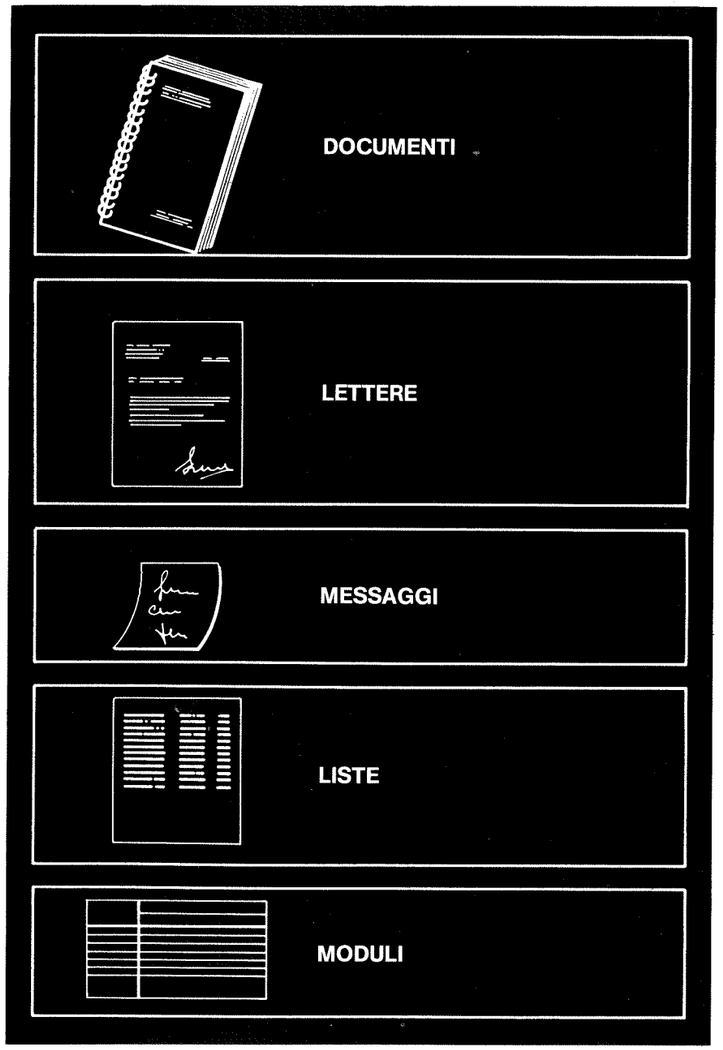
Sono oggetti generalmente "di consumo", corrispondono a brevi comunicazioni informali; vengono normalmente distrutti dal sistema dopo la lettura da parte del destinatario, che comunque, se lo desidera, può memorizzarli e riutilizzarli.

I loro attributi sono ridotti al minimo: da..., a..., data.

d) *Liste*

Sono oggetti strutturati, costitui-

Fig. 5 - Gli oggetti dell'ufficio.



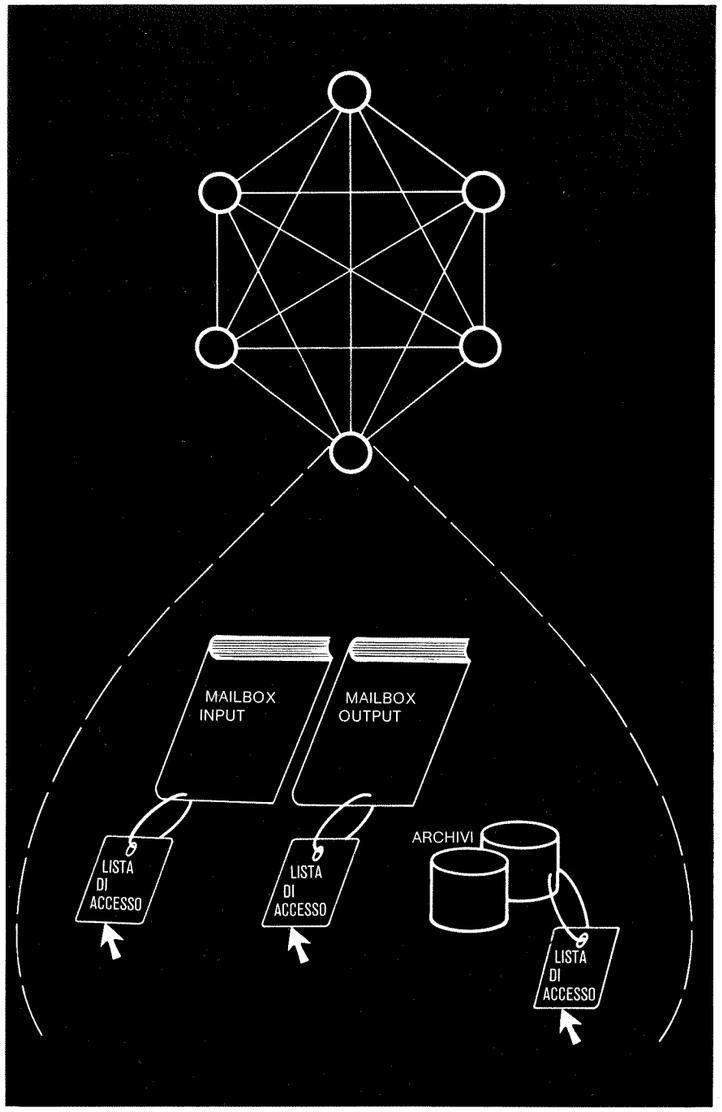


Fig. 6 - Archivi e mailbox.

ti da sequenze di record di struttura identica; possono essere utilizzati per scambio di dati da e verso gli archivi dp, per costruzione di liste di distribuzione, per la personalizzazione di lettere o documenti contenenti campi variabili (i cosiddetti "form documents"), ecc.

e) *Moduli*

Sono oggetti strutturati, più complessi delle liste. Corrispondono all'abituale modulistica di ufficio, e sono usati ad esempio, nell'ambito di procedure d'ufficio formalizzate.

Sono costituiti da campi dotati di nome e tipizzati (un certo campo può contenere uno specifico tipo di dati). Nella loro varietà di formati, sono dotati di un numero minimo di attributi fissi.

In generale, un oggetto viene creato, archiviato, via via aggiornato, composto con altri oggetti, spedito, copiato, annotato, ecc.

Per compiere tali operazioni su di un oggetto è necessario renderlo "disponibile" e visibile, solo o insieme ad altri oggetti.

Le operazioni sugli oggetti vengono riassunte più avanti.

Sottolineiamo, qui, la necessità ricorrente di "comporre" oggetti di natura diversa o di trasformare (cambiandone gli attributi) un oggetto di un tipo in un oggetto di tipo differente.

7. Archivi e mailbox

Tutti gli oggetti vengono memorizzati in "archivi". Un archivio può essere personale o collettivo, locale o centralizzato, e contenere oggetti di tipo diverso.

Ad ogni utente sono associate due mailbox (una di input ed una di output). Queste sono degli archivi adibiti a contenere la "posta" (in arrivo ed in partenza). Definiamo come "posta" tutti gli oggetti spediti (secondo le regole esposte più oltre nell'articolo) tra i vari utenti.

Onde consentire all'utente la massima flessibilità organizzativa, ad ogni archivio (o mailbox) è associata una "lista di accesso", che specifica quali utenti sono autorizzati ad accedere all'archivio stesso.

Sono possibili diversi gradi di accesso: in sola lettura, in lettura e scrittura, in lettura con possibilità di fare copie (ma non di modificare gli originali).

Dal punto di vista logico, il sistema può pertanto essere rappresentato come una rete di "nodi", ad ognuno dei quali è associata una coppia di mailbox e un insieme di archi, con le liste di accesso (fig. 6).

8. Le funzioni principali

Le funzioni che il sistema offre agli utenti sono raggruppate in classi:

1. CREAZIONE/REVISIONE DI OGGETTI
2. STAMPA
3. RICERCA NEGLI ARCHIVI
4. GESTIONE DELLA POSTA IN PARTENZA
5. GESTIONE DELLA POSTA IN ARRIVO
6. AGENDA PERSONALE
7. DEFINIZIONE DELLE PROCEDURE
8. DEFINIZIONE DEGLI ARCHIVI
9. AMMINISTRAZIONE DEL SISTEMA

La suddivisione in tali classi è stata effettuata sulla base di stretti criteri di omogeneità operativa: ogni classe ha associato un sottomenu di funzioni elementari, selezionabile a partire da un menu principale del sottosistema di gestione delle attività d'ufficio (che può essere, a sua volta, personalizzato sulla base delle necessità dei singoli utenti).

Si osservi che, benchè tutte le funzioni del sistema si presentino all'utente con un'interfaccia uniforme (il modello operativo del sistema è semplice; i comandi accettabili in contesti simili sono sempre gli stessi), l'operatore può apprendere l'uso del sistema con gradualità, ad esempio attraverso la seguente sequenza di training/uso:

a) *nucleo base di gestione testi:* creazione/revisione/composizione di testi; stampa; ricerca negli archivi (non tutte le funzionalità). Opera su archivi già definiti, e prodotti in modo opportuno. Non comprende le funzioni (più complesse) di creazione/ristrutturazione di interi archivi, nè funzioni di comunicazione con altre stazioni di lavoro.

b) *nucleo di comunicazione:* gestione della posta in arrivo, gestione della posta in partenza.

Tali funzioni si appoggiano completamente al concetto di archivio (con le relative operazioni base) utilizzato al punto a) (le mailbox di input/output sono archivi), con l'aggiunta di semplici operazioni specifiche.

c) *nucleo avanzato di gestione testi:* definizione degli archivi; ricerca negli archivi (tutte le funzionalità).

Permette operazioni di carattere globale sulle risorse accessibili al particolare utente: definizione o ristrutturazione di archivi; ricerca di informazioni nell'insieme degli archivi, ecc.

d) *nucleo di funzioni personali:* agenda personale.

Costituisce un nucleo indipendente dagli altri, che consente la gestione di impegni (con sollecito) ed appunti personali.

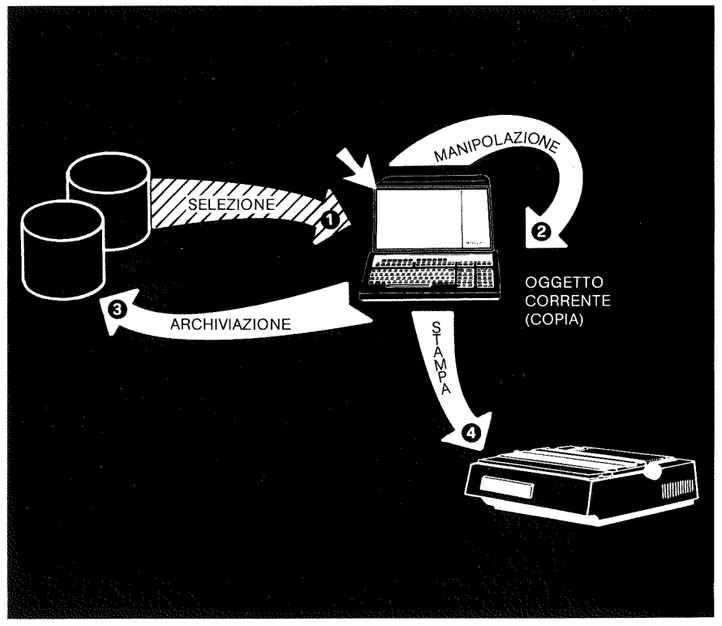


Fig. 7 - Interazione con gli oggetti.

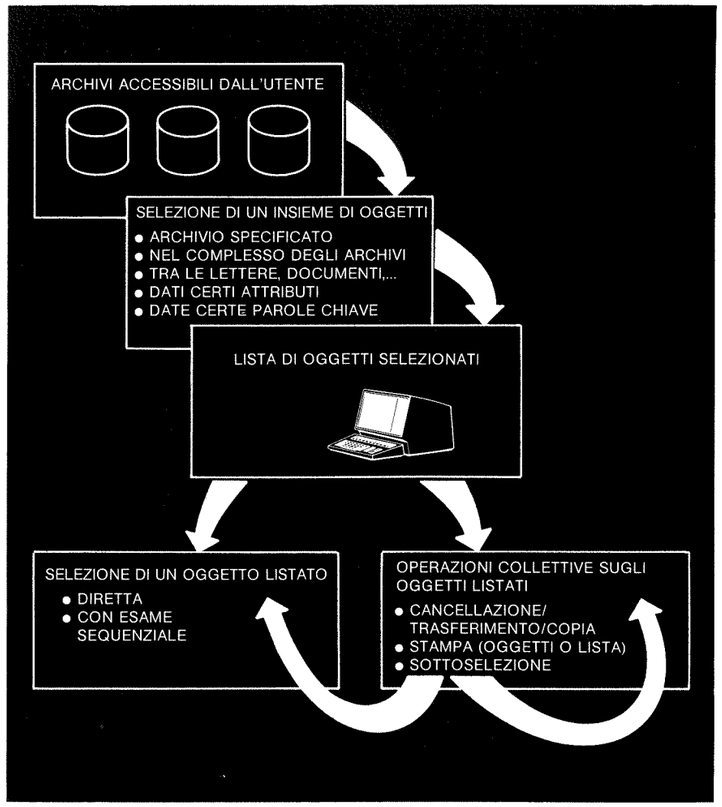


Fig. 8 - Interazione con gli archivi.

e) *nucleo procedurale locale:* definizione delle procedure.

Permette la definizione di macrocomandi denotanti insiemi di operazioni e procedure di uso corrente.

Richiede sensibilità ed impostazione di tipo "sistemistico" (a livello locale).

f) *nucleo di amministrazione del sistema:* amministrazione del sistema. Supporta la creazione di nuovi utenti e la definizione degli standard operativi ai quali si devono conformare i vari utenti "locali". Richiede competenze sistemistiche e procedurali a livello globale.

Nel seguito della presente nota, tratteggiamo sommariamente le varie funzioni.

9. Creazione/revisione/composizione di oggetti

Queste funzioni (vedi fig. 7) operano su un "oggetto corrente", che si trova fuori archivio; pertanto, la creazione e la revisione di un oggetto non modificano lo stato degli archivi. Per revisionare un oggetto archiviato, è necessario "afferrarlo". Il sistema provvede allora, automaticamente ed in modo invisibile all'utente, a crearne una copia di lavoro.

Al termine della sessione di revisione, l'utente potrà richiedere la sostituzione della versione in archivio.

Sono disponibili tutte le funzioni di creazione/revisione fornite da un potente word processing, includendo:

- funzioni di "cut and paste": assiemaggio di oggetti a partire da altri oggetti, memorizzazione di "ritagli" da riutilizzare in fasi successive, memorizzazione di "Form Documents";
- inserimento di note fuori testo;
- costruzione automatica di indici, con possibilità di accesso diretto ai "paragrafi" del documento;

Nel caso di oggetti formattati (liste e moduli):

- funzioni di definizione del formato (mediante dialogo guidato, in cui l'utente definisce l'immagine della lista o del modulo);
- funzioni di compilazione, con input guidato dal formato (tabulazioni verticali/orizzontali, validazione dei tipi di dati, compilazione automatica dei campi "computati" - es. totali, percentuali, ecc.);
- selezione su liste: ad es.: creazione di una sottolista di tutti i valori che godono di una certa proprietà in un'altra lista, ecc...;
- funzioni di merge (costruzione di documenti/lettere con inserimento di dati presenti in una lista);
- acquisizione/trasferimento di dati da/a archivi di Data Processing.

10. Stampa

Oltre alle normali funzioni di formattamento per la stampa su stampante locale o remota, il *merge printing* consente di fondere, direttamente in fase di stampa e senza conservare copie in archivio dei risultati della fusione, documenti contenenti parti variabili e liste (con possibilità di selezione dei valori in lista soddisfacenti a specificate condizioni).

11. Ricerca negli archivi

Questo gruppo di funzioni (vedi fig. 8) permette di condurre la ricerca di un oggetto all'interno di uno o più archivi.

a) *Criteri di selezione*

L'utente può selezionare (reperire) gruppi di oggetti utilizzando tre livelli di selezione:

1. selezione nell'ambito di *un* archivio specificato, o di *tutti* gli archivi di cui l'utente ha visibilità;
2. selezione sul tipo degli oggetti: lettere, documenti,...;
3. selezione basata su:
   - parole chiave specificate in fase di immissione dell'oggetto,
   - valori specificati degli attributi (titolo, autore, ecc.).

Tutti e tre i livelli di selezione possono essere utilizzati contemporaneamente.

b) *Lista degli oggetti selezionati*

La funzione di selezione visualizza la lista (sommario) di tutti gli oggetti selezionati.

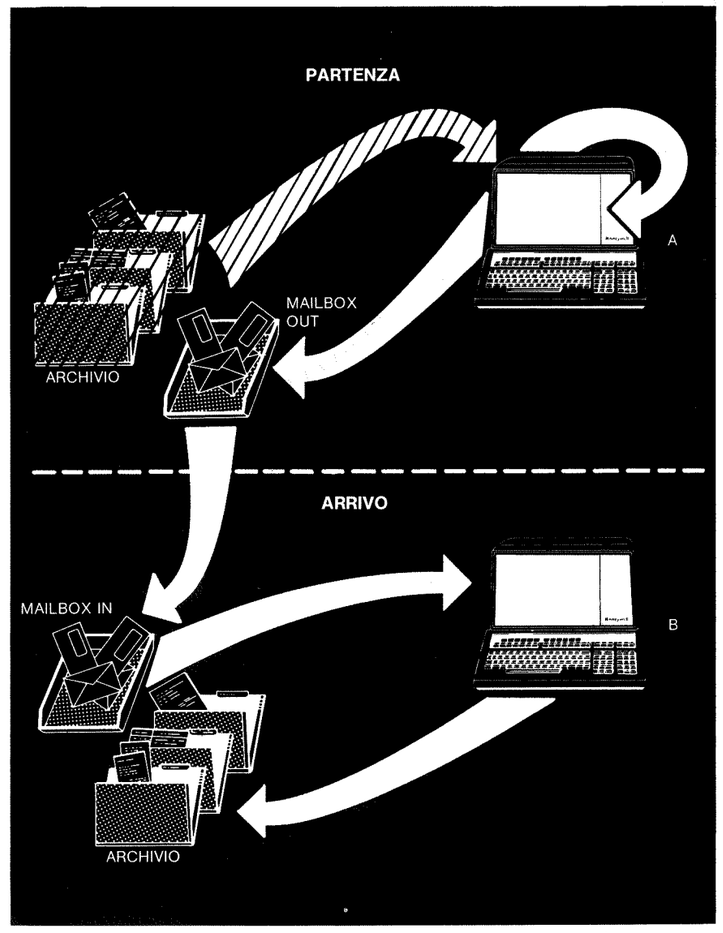
c) *Operazioni sugli oggetti selezionati*

- Selezione di un oggetto:
  A fronte di tale lista, l'utente può selezionare un singolo oggetto o esaminare sequenzialmente i vari oggetti in avanti o all'indietro (partendo da un elemento qualsiasi). L'oggetto selezionato viene visualizzato assieme a tutti i suoi attributi, in un formato analogo al formato di documenti e lettere tradizionali;
- Operazioni sull'oggetto visualizzato:

  Sono possibili tutte le operazioni dell'editor che non modificano il testo preesistente: "scroll" orizzontale e verticale, ricerca di brani di testo, ricerca di un paragrafo specificato, estrazione di "ritagli" per uso successivo, ecc....

  Inoltre sono possibili le seguenti operazioni: "grasp": l'oggetto viene considerato "l'oggetto corrente", per suc-

Fig. 9 - Interazioni con la posta.



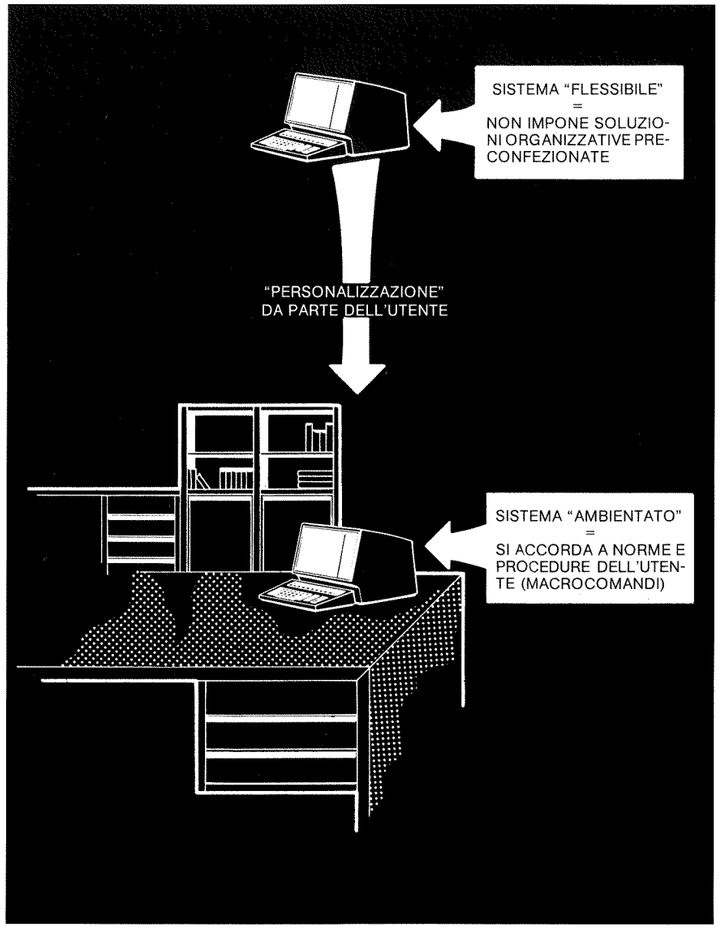


Fig. 10 - Procedure d'ufficio: l'obiettivo finale.

cessive operazioni di revisione; stampa, trasferimento o copia in altro archivio, cancellazione, annotazioni, invio in copia. Sebbene non sia possibile modificare oggetti esaminando un archivio (per far ciò occorre farne una copia nell'area di lavoro) è possibile apporre a qualsiasi oggetto delle annotazioni, che saranno comunque sempre distinguibili dal testo originale;

- Operazioni sull'intera lista:

Gli oggetti listati possono essere manipolati collettivamente con varie operazioni: cancellazione/trasferimento o copia in un altro archivio, stampa di tutti gli elementi listati, stampa della lista.

Si noti che, per permettere di operare sugli oggetti strettamente desiderati, è possibile effettuare ulteriori sottoselezioni sulla lista.

12. Gestione della posta in partenza

Ogni lettera, per poter essere spedita, tramite il servizio di "posta elettronica", deve essere deposta nella *mailbox di output* (vedi fig. 9). Su una mailbox di output sono possibili tutte le operazioni lecite per un generico archivio e, inoltre, le seguenti:

Firma: Il proprietario della mailbox (e nessun altro utente inserito nella lista di accesso) può apporre, con apposito comando, la propria "firma" sulla lettera correntemente visualizzata. Questa diviene un attributo della lettera, che verrà visualizzato con la lettera stessa ("lettera firmata").

Spedizione: Una lettera compilata correttamente può essere spedita, mediante un semplice comando.

Gli oggetti di classe diversa dalle lettere, sono spedibili in allegato ad una lettera.

13. Gestione della posta in arrivo

Le lettere in arrivo vengono depositate dal sistema nella *mailbox di input* (fig. 9). Alla richiesta dell'utente viene visualizzato il sommario del contenuto della mailbox: lettere, eventuali oggetti allegati.

Si noti che una mailbox può essere visionata da chiunque appartenga alla lista di accesso e non necessariamente solo dal suo proprietario. In ogni caso, l'oggetto ed il testo di una lettera dichiarata "riservata" dal mittente vengono visualizzati solo all'utente abilitato.

Le operazioni accettate sulla mailbox sono quelle possibili su un generico archivio.

Le lettere ricevute non possono essere modificate. Un comando opportuno provoca la costruzione automatica e l'invio immediato al mittente della lettera visualizzata, di una lettera di "ricevuta", senza che l'utente debba specificarne testo o attributi. Dopo la lettura di una lettera l'utente può archiviarla, specificando l'archivio desiderato. Altrimenti, la lettera non viene archiviata e viene riproposta alla visione dell'utente alla successiva riapertura della mailbox.

14. Agenda personale

Questo gruppo di funzioni permette di gestire una "agenda" di note e memorandum privati.

L'utente ha a disposizione un'agenda facilmente consultabile specificando il giorno (oggi, domani, una data precisa) o selezionando un'intera settimana.

E possibile istruire il sistema affinchè riproponga all'utente (all'atto del LOGIN), in date opportune, messaggi contenuti nell'agenda.

15. Definizione delle procedure

Il sistema fin qui descritto è stato concepito secondo il criterio della massima flessibilità operativa: esso, sostanzialmente, non impone all'utente nessuna procedura di lavoro prestabilita. Infatti, il sistema deve potersi inserire con la massima flessibilità in situazioni organizzative anche molto differenti, senza "perturbarle".

Ogni singola entità organizzativa, utente del sistema, avrà in generale, tuttavia, un insieme di norme operative e procedure, più o meno formalizzate.

Un sistema "avanzato" di automazione delle attività di ufficio dovrà, pertanto, essere adeguabile a tali norme e procedure. In altri termini, l'utente specifico dovrà avere la possibilità di personalizzare il sistema alle diverse procedure in uso (fig. 10). Ad esempio, dovrà essere possibile comunicare al sistema che, in un dato contesto organizzativo:

- "la spedizione di una lettera può avvenire soltanto se la lettera è firmata dal mittente";
- "ogni lettera arrivata con certi attributi, deve essere archiviata in uno specifico archivio ed inviata, con una specificata annotazione, ad uno specificato destinatario";
- "ogni variazione nella lista di accesso di certi archivi deve essere comunicata mediante una lettera standard a tutti gli utenti interessati";

eccetera.

Ciò può essere realizzato permettendo all'utente di definire dei "macrocomandi" (es.: SPEDISCI, SMISTA, ...), in termini di:

- comandi elementari (tutti i comandi trattati dal sistema e riassunti nelle precedenti pagine);
- condizioni che rendono possibile l'esecuzione di tali comandi (es.: "Firma presente", "attributo x= yyyy").

Tali possibilità sono attualmente allo studio; per un possibile linguaggio di definizione di tali procedure si veda [2].

16. Definizione degli archivi

Questo gruppo di funzioni supporta una normale gestione degli archivi (creazione/cancellazione/duplicazione) nonchè la gestione delle corrispondenti liste di accesso.

17. Amministrazione del sistema

Questo nucleo di funzioni è visibile esclusivamente ad uno specifico utente, che ha il compito di "amministrare" il sistema:

- creazione e gestione di tutti gli utenti e dei relativi profili;
- abilitazione dei diversi utenti all'uso delle diverse classi di funzioni offerte dal sistema;
- creazione e gestione degli archivi centrali e di "back-up";
- definizione delle procedure a livello globale.

18. Conclusioni

Nella presente nota sono stati descritti sinteticamente i requisiti funzionali di un sistema per l'automazione di ufficio che integri funzionalità di tipo "tradizionale" (per es.: word processing) con funzionalità più sofisticate (per es.: posta elettronica, integrazione con Data Processing, archiviazione e reperimento degli "oggetti" dell'ufficio), fino a funzionalità che permettono l'adeguamento del sistema a norme e procedure specifiche di un dato ufficio.

19. Bibliografia

[1] G. DEGLI ANTONI, R. POLILLO, M. POZZI, B. ZONTA: *"Dall'ufficio del futuro all'ufficio di oggi"*, Convegno annuale AICA 1980, Bologna.

[2] V. DE ANTONELLIS, G. DEGLI ANTONI, G. MAURI, B. ZONTA: *"Una notazione per la descrizione delle procedure e del loro comportamento basata sulle reti di Petri"*. Giornate di studio su "La programmazione strutturata: metodologia e strumenti per l'analisi ed il disegno" Milano 23-25 maggio 1979, Honeywell I.S.I.

Nota:
Il presente lavoro è stato svolto nell'ambito del "Communication and Programming Project", progetto di cooperazione fra Honeywell Information Systems Italia ed Istituto di Cibernetica dell'Università di Milano. Rappresenta un risultato di numerosi lavori di ricerca e sperimentazione sull'automazione di ufficio svolti da alcuni anni nell'ambito del CP Project.

Al presente lavoro hanno collaborato, principalmente, le seguenti persone: G. Torriani, G. Degli Antoni, V. De Antonellis, D. Biglino, S. Cappelli, R. Paulin.

La chimica computazionale: progettazione di molecole col calcolatore

GIANFRANCO PACCHIONI

Institut für Physikalische Chemie
Freie Universität
Berlin (BRD)

1. Introduzione

La seconda metà dell'800 ha visto impegnati nella lotta contro le malattie infettive gli ultimi esempi di scienziati in grado di cimentarsi con successo tra i banchi di un laboratorio di chimica come tra i letti di un ospedale, tra le pecore ammalate di carbonchio come tra le cavie da esperimento degli istituti biologici. I Pasteur, i Koch, i Roux, quel gruppo di uomini insomma passati alla storia, e in alcuni casi alla leggenda, come "cacciatori di microbi" rappresentano in un certo senso l'anello di congiunzione tra la concezione empirica della ricerca, quella, per intenderci, del provando e riprovando, e il tentativo di razionalizzare e di comprendere a fondo i fenomeni naturali che ha caratterizzato questo secolo.

Sotto questo aspetto la figura di Paul Ehrlich e la storia della sua lotta contro le spirochete sono emblematiche. Ehrlich nel 1896 dirigeva un laboratorio a Berlino chiamato "Regio Istituto Prussiano per il controllo dei sieri". Qui egli cercava di dare una dimostrazione convincente del fatto che per ogni germe doveva esistere un veleno, una antitossina capace di neutralizzarne l'azione. Non solo. Egli era un convinto assertore del fatto che tutti i complessi equilibri esistenti negli organismi viventi dovevano essere retti da leggi matematiche precise e che pure delle leggi matematiche dovevano governare l'azione dei farmaci contro i microbi. Si trattava di ipotesi davvero assurde per quel tempo tant'è vero che lo stesso Ehrlich non si preoccupò certo di cercare (ammesso che esistessero) tali leggi ma piuttosto si gettò con accanimento nella ricerca di un prodotto di sintesi in grado di uccidere le pericolose spirochete. Lo trovó, infine, il 31 agosto del 1909. Si trattava del "diossi-diamino-arsenobenzolo diidrocloruro", ma Ehrlich preferì chiamarlo col numero progressivo con cui aveva contrassegnato negli anni i vari farmaci sintetizzati: 606.

Ehrlich non poteva però immaginare che, mentre conduceva i suoi esperimenti, una nuova generazione di fisici costituita dai Planck, dagli Einstein, dai Bohr, stava costruendo le basi per arrivare a determinare quelle leggi matematiche che regolano i fenomeni microscopici della materia, vivente e non, in cui Ehrlich credeva più per fede irrazionale che per convinzione scientifica.

Una delle aspirazioni dell'uomo è sempre stata quella di poter conoscere più a fondo la natura e i suoi meccanismi, e l'obiettivo finale è quello di saper dare una risposta a domande tipo "qual'è il modo in cui l'aspirina fa passare il mal di testa", "come si originano le cellule cancerogene" o "quali sostanze sono in grado di uccidere le zanzare pur essendo innocue per l'uomo e perché".

A queste domande si è cercato sino ad oggi di dare una risposta tramite la tradizionale, e per ora insostituibile, sperimentazione, ma è chiaro che nell'era dell'informatica la possibilità di giungere a conclusioni altrettanto stringenti su pure basi teoriche con l'ausilio dei calcolatori non può essere scartata.

2. Molecole al calcolatore

Facciamo un esempio concreto. Da molti anni è noto che il piretro, un prodotto naturale di origine vegetale, possiede una azione tossica nei confronti degli insetti senza peraltro essere pericoloso per l'uomo. Nell'immediato dopoguerra si è provveduto a sostituire il piretro con il DDT, meno costoso in quanto prodotto sinteticamente, col risultato che molti insetti si sono assuefatti alla presenza del velenoso insetticida mentre i danni derivanti all'ambiente si sono dimostrati ingenti.

Così l'interesse per il piretro è rapidamente tornato in auge dando luogo a una richiesta sempre maggiore. Il problema è diventato il produrre il piretro, o un insetticida dalle proprietà simili, per via industriale, abbassando così notevolmente il costo del prodotto e rendendolo disponibile in grandi quantità. Molti centri di ricerca si sono dedicati allo studio della messa a punto di una sintesi economica in grado di partire da materie prime facilmente disponibili. Per prima cosa si è stabilita la relazione esistente tra l'attività biologica di questa classe di insetticidi e la struttura della molecola. In particolare si è accertato che tale attività insetticida è da attribuire all'acido crisantemico, una sostanza che si trova in molti piretroidi, e che tale acido crisantemico presenta nella sua struttura un anello di tre atomi di carbonio noto come ciclopropano.

Quindi si è stabilita una relazione diretta tra la presenza di tale anello e la capacità del piretro di agire come potente insetticida. Il problema che a questo punto si è presentato è stato quello di trovare un modo comodo per produrre industrialmente l'acido crisantemico. Purtroppo però proprio la presenza dell'anello ciclopropanico si è dimostrata un punto particolarmente difficile da superare dato che in natura gli atomi di carbonio tendono a dare strutture tetraedriche e che solo a prezzo di grossi sforzi si adattano a combinarsi in forme così poco "naturali" come un anello a tre atomi.

La fase successiva della ricerca è consistita quindi nell'individuare una nuova classe di molecole che, pur avendo più o meno la stessa forma e quindi le medesime caratteristiche dell'acido crisantemico, non contenessero tale anello e pertanto fossero più facili da preparare.

Ora, se Ehrlich per trovare il suo farmaco efficace dovette provarne e scartarne 605 prima di ottenere quello buono, oggi che i composti chimici noti e caratterizzati sono alcuni milioni, il numero di molecole da sintetizzare e studiare prima di determinarne una con le caratteristiche volute può essere enormemente superiore.

Tanto per dare qualche numero puramente indicativo, si può arrivare a preparare 10-20.000 molecole per avere un nuovo antiparassitario e sino a 50.000 se si tratta di mettere a punto un nuovo farmaco.

Ciò comporta ovviamente costi e tempi spaventosamente elevati dato che la via da percorrere in questi casi è composta di numerose fasi che vanno dalla pura e semplice sintesi della molecola, alla sua caratterizzazione, dai test che devono dimostrarne l'efficacia a quelli che ne devono garantire l'innocuità per l'uomo.

È chiaro quindi che la possibilità di ridurre a priori per via teorica il numero di eventuali molecole da studiare rappresenta una allettante prospettiva per la chimica d'oggi.

Anche nel campo della chimica quindi ci si è rivolti verso l'uso dei computer per limitare, almeno là dove possibile, le dispendiose ricerche di laboratorio. Per tornare all'esempio del piretro, o meglio dell'acido crisantemico, si è cercato di stabilire per via teorica quali molecole presentassero una conformazione simile a quella dell'acido crisantemico e quindi potessero "sovrapporsi" ad esso in modo da lasciar prevedere caratteristiche insetticide simili a quelle del piretro. Si tratta di un lavoro impossibile da eseguire a mano: esistono moltissime molecole la cui struttura potrebbe rispondere ai requisiti richiesti e per ognuna di esse le "torsioni" attorno ai legami liberi di muoversi producono a loro volta infinite possibili conformazioni.

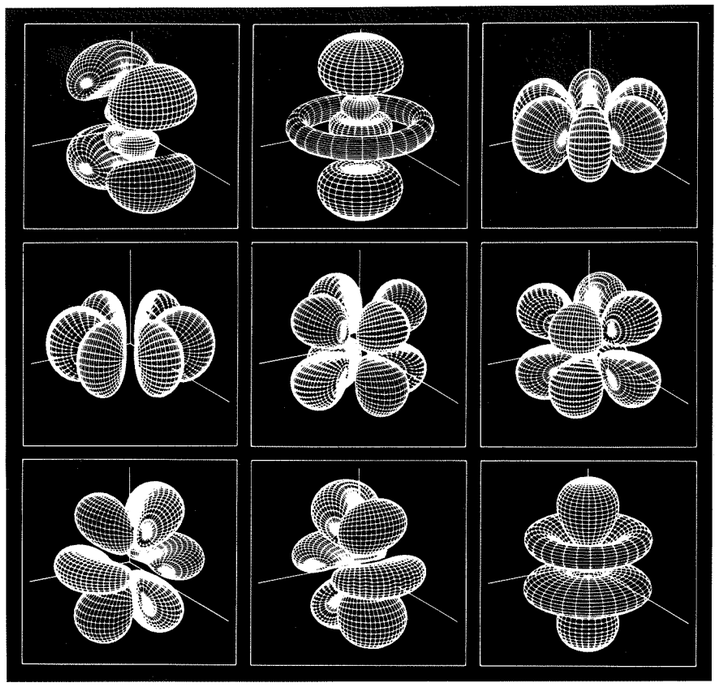


Fig. 1 - Forma spaziale di alcuni orbitali atomici disegnati con l'ausilio del calcolatore.

Viceversa, con l'uso del calcolatore ciò è divenuto accessibile, tanto che è stata recentemente messa a punto una nuova classe di insetticidi la cui struttura è simile a quella del piretro naturale ma la cui sintesi non è ostacolata dalla presenza dell'anello ciclopropanico.

In pratica con una selezione effettuata al calcolatore si è ristretto di molto il campo su cui concentrare le necessarie ricerche di laboratorio con notevole risparmio di tempo e di costi.

L'esempio del piretro è un po' particolare e non si deve pensare che l'uso di uno strumento come il calcolatore possa di colpo stravolgere la millenaria tradizione sperimentale della chimica. Esiste comunque un settore della ricerca chimica relativamente nuovo che tende a interpretare ma anche, almeno là dove possibile, a prevedere ex novo il comportamento di atomi e molecole nonché i vari modi con cui si combinano tra di loro. Tale disciplina viene generalmente indicata come chimica teorica o, in certi casi, *chimica computazionale,* sottointendendo con ciò come in tale approccio la parte più importante spetti allo stesso tempo alle teorie fisiche di base nonché al supporto dei computer.

3. La teoria quantistica

I primi esempi di calcoli teorici risalgono agli anni '30, al tempo in cui la meccanica quantistica era stata ormai formulata nelle sue grandi linee. Come noto, la meccanica quantistica affonda le sue radici nella teoria dei quanti enunciata da Max Planck all'inizio del secolo. Fu solo tra il 1920 e il 1930 comunque che la nuova e rivoluzionaria teoria, secondo cui l'energia viene emessa o assorbita dai sistemi atomici o molecolari secondo quantità ben definite dette "quanti", ebbe un suo pieno riconoscimento nonché un proprio apparato matematico grazie ai contributi di scienziati quali de Broglie, Pauli, Schrödinger, Heisenberg, Dirac.

Le leggi della meccanica quantistica si distinguono da quelle della fisica classica per il loro carattere fondamentale.

In parole povere, mentre le leggi quantistiche sono in grado di descrivere altrettanto bene il moto di un elettrone attorno a un nucleo atomico quanto quello di un pianeta attorno al sole, ciò non è più vero per le leggi classiche. Per questo quando si parla di "rivoluzione" determinata nella fisica dalla scoperta della teoria dei quanti si usa un termine non appropriato poiché tale teoria non toglie validità a quella classica (che può essere vista come limite della prima con un campo di applicabilità ristretto alla descrizione dei fenomeni macroscopici) ma ne costituisce, anzi, una generalizzazione.

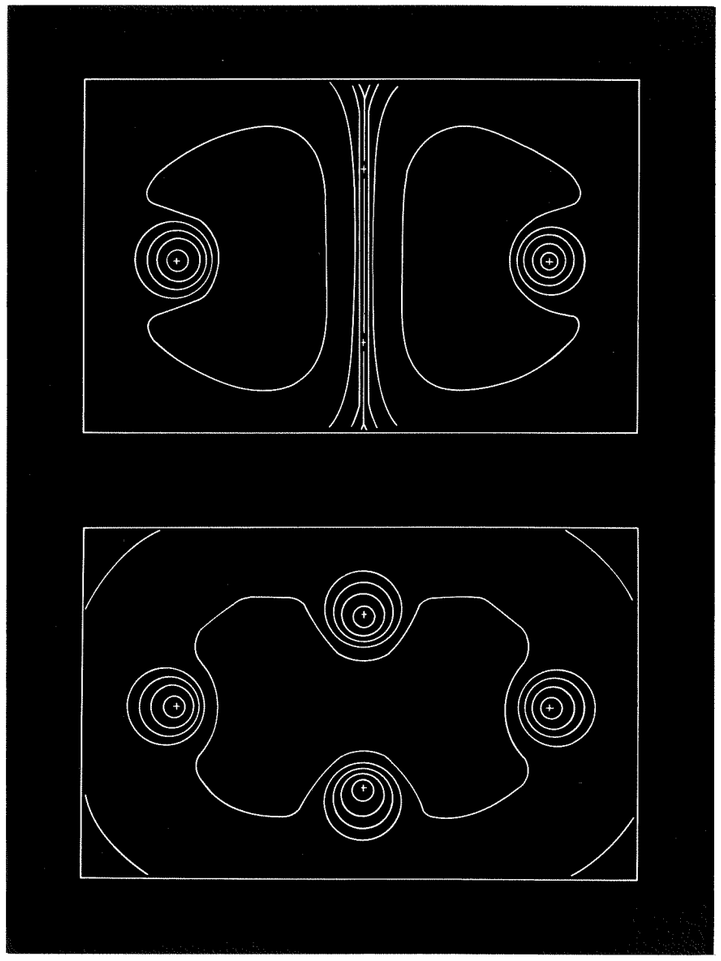


Fig. 2 - Mappe di densità elettronica in un sistema molecolare. I + indicano le posizioni occupate dai nuclei atomici.

Per mezzo delle leggi classiche si può descrivere il comportamento (moto) di un meccanismo composto da molle, leve, ecc., se si conoscono alcune costanti di materiali quali la densità, il calore specifico, l'elasticità. Però, se ci si chiede perché le densità sono quelle che sono, perché le costanti elastiche hanno i valori che hanno, perché una barra si spezza se la tensione cui è sottoposta supera un certo limite, e così via, la fisica classica non fornisce risposta. Essa non dice perché il rame fonde a 1083°C, perché il vapore di sodio emette luce gialla, perché l'idrogeno ha certe proprietà chimiche, perché il sole brilla, perché il nucleo dell'atomo di uranio si disintegra spontaneamente, perché l'argento conduce l'elettricità, perché lo zolfo è un isolante. A queste domande può dare una risposta la meccanica quantistica. Sulla base delle elementari interazioni tra le particelle costituenti la materia, si può via via risalire ai comportamenti più complessi che danno vita ai fenomeni macroscopici che ogni giorno osserviamo. Occorre però precisare: il fatto che la teoria dei quanti possa dare una risposta in teoria a tali domande non sempre significa che la possa dare anche in pratica. Qui entra in gioco infatti la complessità del problema e la necessità di servirsi dei calcolatori elettronici.

4. L'equazione di Schrödinger

Per tornare agli anni '30, la formulazione matematica della fisica quantistica proposta da Schrödinger con la sua funzione d'onda si prestava particolarmente per lo studio di sistemi atomici o molecolari grazie alla semplicità della trattazione e alla interpretazione fisica più diretta del significato stesso della funzione d'onda.

Come noto, alcune grandezze fisiche nei sistemi microscopici sono in relazione con l'energia totale del sistema, ossia con il lavoro che occorre compiere per distruggere il sistema stesso. Se parliamo di una molecola ad esempio, tale quantità corrisponde al lavoro necessario per "scollare" le une dalle altre tutte le particelle presenti (nuclei ed elettroni) e portarle a distanza infinita privandole della loro energia cinetica. Ad ogni sistema atomico o molecolare è possibile associare un operatore hamiltoniano, indicato con H, che descrive l'energia del sistema. Se questo ha energia costante (come un atomo che non assorba o non emetta radiazione, che non si ionizzi, ecc.) applicare l'operatore hamiltoniano alla funzione d'onda ψ porta alla celebre equazione di Schrödinger che nella forma più generale è nota come $H\Psi = E\Psi$. Più in dettaglio si può scrivere:

$$[-\nabla^2 + U(x,y,z)]\,\Psi\,(x,y,z) = E\,\Psi\,(x,y,z,)$$

omettendo alcune costanti moltiplicative. Il termine in parentesi quadra è l'operatore hamiltoniano ed E è l'energia totale del sistema. Schrödinger dimostrò che tale equazione è risolvibile solo per valori quantizzati di E. Nell'operatore hamiltoniano sono contenuti due termini che rappresentano l'energia cinetica e quella potenziale delle particelle presenti nel sistema.

Nel caso di un atomo in particolare, composto da un nucleo carico positivamente e da un certo numero di elettroni, l'energia totale del sistema è data dalla somma di alcuni contributi.

Innanzitutto va considerata l'energia cinetica degli elettroni che, essendo in continua rotazione attorno al nucleo, posseggono una propria energia di movimento; poi va considerata l'attrazione elettrostatica che il nucleo esercita sull'elettrone nonché la repulsione che ogni elettrone esercita sugli altri elettroni.

Quindi l'operatore hamiltoniano conterrà tre termini, ciascuno dei quali rappresenta uno di questi contributi all'energia totale del sistema. L'operatore di Laplace ∇^2, costituito da derivate parziali del secondo ordine rispetto alle coordinate, descrive l'energia cinetica, mentre le interazioni elettrostatiche, attrattive o repulsive, sono racchiuse nel potenziale U(x,y,z) dell'operatore hamiltoniano. Le informazioni relative al numero di elettroni, alla loro maggiore o minore distanza dal nucleo, alla loro disposizione spaziale sono invece contenute nella funzione d'onda Ψ. Il significato fisico di tale funzione è che essa, o meglio il suo quadrato, descrive la probabilità di trovare un elettrone (o un'altra particella) in una certa regione di spazio ad un dato istante.

Da questi "ingredienti" di partenza è possibile ricavare tutta una serie di informazioni sulla struttura dell'atomo e sulle sue proprietà. Perfettamente simile è il caso molecolare, dove l'operatore hamiltoniano conterrà nuovi termini relativi alle interazioni non presenti nell'atomo come, ad esempio, la repulsione reciproca dei nuclei atomici o l'attrazione che il nucleo A esercita sugli elettroni del nucleo B e viceversa.

L'equazione di Schrödinger comunque, esattamente risolvibile nel caso dell'atomo di idrogeno e, con grandi sforzi, di altri sistemi molto semplici, si trasforma in un complicato insieme di equazioni differenziali man mano che il sistema in esame aumenta le sue dimensioni. Questo comporta a sua volta la risoluzione di un certo numero di integrali il cui numero cresce grosso modo come n^4, dove n è il numero di orbitali atomici presenti nella molecola.

Sino alla fine degli anni '50 quindi la possibilità di studiare teoricamente il comportamento chimico-fisico della materia è stato limitato dalla scarsa diffusione di calcolatori sufficientemente potenti. Per dare un'idea di come l'analisi della struttura elettronica di una molecola richieda uno sforzo computazionale enorme basta dire che gli integrali da calcolare nel caso di una moleco-

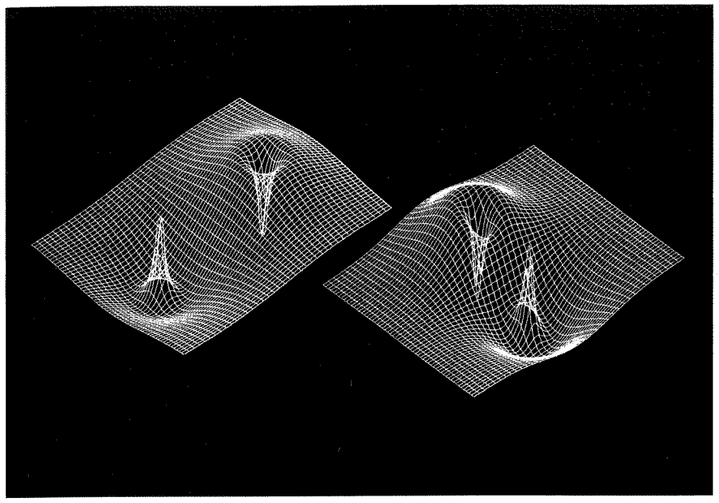


Fig. 3 - Forma degli orbitali naturali di un aggregato di quattro atomi di litio ottenuti sulla base di calcoli teorici e disegnati dal plotter del calcolatore.

la non particolarmente grande come il glucosio sono circa tre milioni e mezzo. Essi possono però diventare molti di più quando si vogliano ottenere dei risultati particolarmente affidati.

In questo caso, infatti, occorre usare per la descrizione della funzione d'onda un set di "funzioni di base" piuttosto esteso che può contenere due, tre o più volte il numero di orbitali atomici presenti nella molecola.

L'uso di "basi minime", cioé di un numero di funzioni pari al numero di orbitali, spesso produce risultati in cattivo accordo con i dati sperimentali a causa dell'approssimativa descrizione della funzione d'onda. In linea di principio, solo un set di base infinito può rappresentare la Ψ esatta del sistema, ma ciò ovviamente è fuori dalla portata di qualsiasi calcolatore. Il compromesso che si cerca di raggiungere è quello tra una soddisfacente descrizione della funzione d'onda e un calcolo non troppo oneroso.

Purtroppo lo studio delle proprietà elettroniche di una molecola di interesse biologico come la ferro-porfirina, una unità di base dell'emoglobina, anche quando si usi una base minima richiede il calcolo di circa 220 milioni di integrali, ognuno dei quali viene valutato mediante un buon numero di operazioni. Se poi l'integrazione, come avviene in certi casi, va fatta per via numerica anziché analiticamente, il numero di operazioni da svolgere cresce ulteriormente assieme al tempo di calcolo.

5. Primi calcoli quantomeccanici

Ciò ha limitato notevolmente lo sviluppo della chimica teorica almeno sin verso la metà degli anni '60, quando la diffusione dei primi programmi di calcolo di una certa efficienza nonché di elaboratori più potenti ha permesso a questa disciplina di staccarsi dalla fase puramente teorico-matematica per entrare in quella della interpretazione, della verifica e infine della previsione dei dati sperimentali.

In questa prima fase evolutiva della chimica computazionale, il tipo di calcoli prevedeva come unici dati di partenza le costanti fisiche del sistema, come la carica e la massa di elettroni e nuclei, e, ovviamente, il numero di elettroni e di nuclei presenti nella molecola nonché le posizioni spaziali dei nuclei stessi.

Fatta l'approssimazione che i nuclei siano fissi, dato che la loro velocità di movimento risulta estremamente più bassa rispetto a quella degli elettroni (approssimazione di Born-Oppenheimer), il calcolo veniva effettuato *ab initio*, senza introdurre cioé ulteriori approssimazioni. Secondo la teoria MO (dall'inglese Molecular Orbital), un orbitale molecolare è espresso come combinazione lineare di orbitali atomici e la forma nonché l'energia di tali orbitali molecolari viene determinata risolvendo una equazione matriciale ad autovalori-autovettori in cui compaiono matrici di ordine n, dove n è il numero di orbitali di base del sistema.

Tra i primi calcoli eseguiti in questo modo vi sono quelli che l'italiano Enrico Clementi fece negli anni '62-'64 negli Stati Uniti. Il programma di calcolo molecolare IBMOL, che venne messo a punto per questo genere di studi, è uno dei primi che sia circolato liberamente negli ambienti scientifici, e ancora oggi è usato nelle sue versioni più aggiornate.

Il primo obiettivo che ci si pose al momento di affrontare teoricamente lo studio delle interazioni chimiche fu quello di riprodurre i dati sperimentali esistenti al fine di verificare l'affidabilità del metodo. Un obiettivo che oggi si può considerare pienamente raggiunto dato che esistono addirittura esempi di come calcoli teorici sufficientemente accurati abbiano messo in evidenza l'inesattezza di alcune misure sperimentali.

Nonostante l'uso del computer abbia permesso di effettuare calcoli che, se fatti a mano, avrebbero richiesto l'impegno di parecchi ricercatori per qualche decina d'anni, l'uso dei metodi *ab initio* rimane sempre piuttosto oneroso dato che la diagonalizzazione delle matrici in gioco richiede tempi, ma soprattutto estensioni di memoria, spesso al di fuori della portata dei normali elaboratori. Anzi, proprio l'occupazione di memoria costituisce il limite fisico all'effettuazione di elaborati calcoli, dato che il numero di integrali da calcolare e organizzare in matrici cresce, come si è detto, in modo quasi esponenziale.

Per questo i metodi *ab initio* sono stati usati, e sono usati tuttora, per lo studio accurato di sistemi contenenti sino a 10-20 atomi, ma non sono applicabili quando si vogliano considerare reazioni che coinvolgono molecole più grandi.

Anche l'uso della teoria dei gruppi, che permette di evitare di calcolare molti integrali quando la molecola possegga una certa simmetria, riduce il calcolo ma non elimina i problemi di fondo.

D'altra parte la maggior parte delle reazioni di interesse chimico, fisico o biologico, coinvolge sistemi di grandi dimensioni. Basti pensare ai processi di sintesi di composti organici naturali, alla struttura dello stato solido o alle reazioni del DNA, per rendersi conto di come si sia lontani dal poter giungere a dare un contributo per via teorica servendosi di calcoli *ab initio*.

6. I metodi semiempirici

Nella seconda metà degli anni '60 si è intensificata la ricerca di nuovi metodi in grado di estendere le indagini teoriche a problemi realmente significativi.

Sono così comparsi i primi esempi di metodi *semiempirici*, così detti in quanto oltre alle informazioni di base già presenti nei metodi *ab initio* si introducono altri dati di carattere sperimentale (da cui il nome di metodi semiempirici) quali ad esempio le energie di ionizzazione sperimentale degli atomi.

In particolare, molti degli integrali da calcolare, corrispondenti ad altrettante interazioni all'interno della molecola, vengono trascurati se supposti molto piccoli o parametrizzati in modo da ridurre drasticamente le dimensioni del calcolo.

Inoltre, questi metodi tendono a considerare espressamente solo alcuni degli elettroni presenti nella molecola e in particolare quelli che determinano le proprietà chimiche.

L'effetto della restante parte degli elettroni, cioé il potenziale da essi generato, viene simulato in modo da semplificare il calcolo ma facendo attenzione che i risultati non si discostino molto dai corrispondenti dati sperimentali.

Accade quindi che mentre i metodi *ab initio*, dove tutte le interazioni sono espressamente introdotte, forniscono qualsiasi informazione sulle osservabili fisiche della molecola, i metodi semiempirici sono usati in modo diverso a seconda del problema da studiare. Con essi si è comunque riusciti a superare il problema tecnico dell'estensione del calcolo generato dai metodi *ab initio*, tant'è che i chimici, anche sperimentali, si sono molto avvalsi negli ultimi anni di tali metodi per interpretare dati o per ottenere risposte qualitative a problemi che la sperimentazione da sola non poteva chiarire.

Oggi circa la metà dei lavori scientifici di chimica teorica pubblicati ogni anno fa uso di tali metodi. È grazie ad essi infatti che si è potuto gettare un ponte tra teoria e sperimentazione, che comincia a dare i primi frutti concreti.

Di metodi semiempirici ne esistono di diverso tipo e recentemente ad essi si sono affiancati, soprattutto in campo industriale, i metodi di *drug design*, letteralmente progettazione di farmaci, in cui tutte le informazioni note sulle relazioni esistenti tra struttura chimica, proprietà fisiche e attività biochimica di una classe di composti vengono elaborate statisticamente allo scopo di giungere alla ottimizzazione teorica della struttura di un farmaco o di una molecola che risponda a certe caratteristiche volute (è ad esempio il caso del piretro di cui si è parlato all'inizio).

Con i metodi di *drug design* quindi si memorizzano grandi quantità di dati relativi alle relazioni tra struttura e reattività di una classe di composti, quindi si analizzano mo-

Fig. 4 - Densità degli stati di un cluster metallico ottenuti mediante calcoli teorici di tipo quantomeccanico. L'infittirsi dei picchi indica una struttura simile a quella del metallo.

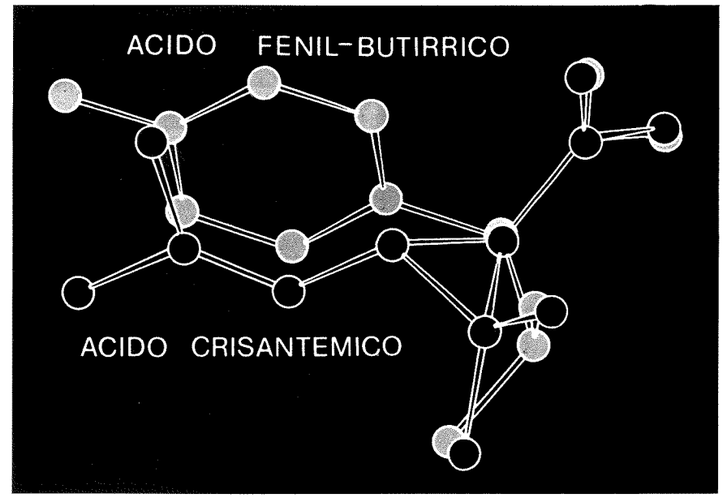


Fig. 5 - La struttura dell'acido fenil-butirrico è molto simile a quella dell'acido crisantemico, una molecola difficile da sintetizzare a causa della presenza di un anello ciclopropanico. L'acido fenil-butirrico, che ha proprietà simili a quelle dell'acido crisantemico, è stato individuato tra migliaia di molecole simili grazie all'ausilio di calcoli eseguiti con il calcolatore.

lecole strutturalmente simili allo scopo di individuare a tavolino il farmaco, l'antiparassitario, l'insetticida ideali per quel dato problema, limitando a pochi casi le definitive ricerche di laboratorio. Con tali metodi i sistemi studiati possono anche essere molto grandi, ma in generale la necessità di introdurre sempre nuove e sempre più drastiche semplificazioni a livello teorico finisce per limitare in parte l'attendibilità dei risultati.

7. La teoria dello pseudopotenziale

Una sorta di compromesso tra gli onerosi metodi *ab initio* e i meno rigorosi metodi semiempirici sembra essere stato raggiunto negli ultimi cinque-sei anni con la comparsa di una nuova metodologia di calcolo che si colloca tra le due precedenti e che va sotto il nome di *pseudopotenziale.*

Per la verità, la teoria dello pseudopotenziale era già nota da molti anni in fisica dove è stata con successo applicata allo studio dello stato solido, ma l'idea di utilizzare lo stesso principio anche nei problemi chimici è più recente.

Per capire meglio il presupposto fondamentale su cui si basa tale teoria è però necessario fare una breve premessa sulla struttura elettronica degli atomi. Prendiamo a mo' di esempio due atomi, il ferro e l'ossigeno, e il loro composto ossido di ferro FeO. Come noto, gli elettroni si dispongono attorno al nucleo atomico in orbitali dalle varie forme spaziali e dalla diversa energia: in particolare, più vicini al nucleo ci sono gli orbitali più energetici, ossia gli orbitali in cui gli elettroni si muovono con la massima velocità per compensare la forte attrazione esercitata su di essi dal nucleo. A seconda dei numeri quantici che li contraddistinguono, tali orbitali sono raggruppati in "gusci" o "livelli".

Nel ferro i livelli occupati sono quattro e contengono 2, 8, 8 e 8 elettroni rispettivamente andando dal guscio più interno a quello più esterno. Nell'ossigeno i gusci sono due e contengono 2 elettroni il primo e 6 il secondo.

Ora, quando passiamo a considerare la formazione della molecola, sappiamo che il legame chimico è determinato in pratica dai soli elettroni del guscio più esterno, elettroni *di valenza*, mentre quelli interni, *di cuore,* modificano di poco il loro assetto rispetto a quello che avevano nell'atomo libero.

Quindi nell'ossido di ferro gli elettroni importanti al fine del legame sono 14, e cioè 8 del ferro e 6 dell'ossigeno. Nei metodi *ab initio* il campo di potenziale generato dagli elettroni interni nella molecola viene calcolato esattamente con grande dispendio di tempo di calcolo. Nei metodi di pseudopotenziale invece tale effetto viene riprodotto con l'introduzione di un potenziale del tutto simile a quello che gli elettroni di cuore generano nell'atomo. Con questo potenziale, da cui il nome di metodi di pseudopotenziale, si tiene conto del fatto che gli elettroni di valenza nella molecola risentono dell'attrazione di un nucleo "rivestito" dagli elettroni dei gusci interni. In pratica si ricade in un metodo *ab initio* dove però si considerano i soli elettroni di valenza ai fini del calcolo delle interazioni elettroniche.

Nel caso banale della molecola di FeO si passa così da un totale di 34 a 14 elettroni e gli integrali da calcolare in base minima si riducono da 22155 a 1540.

I vantaggi offerti da tali metodologie sono evidenti: da un lato una affidabilità dei risultati pari a quella dei metodi *ab initio,* dall'altro una notevole riduzione delle dimensioni del problema che, se non rende i metodi di pseudopotenziale competitivi rispetto a quelli semiempirici, permette comunque di affrontare sistemi di interesse applicativo.

8. Le risposte della chimica teorica

Chiariti pregi e difetti dei principali metodi di calcolo di cui si avvale oggi la chimica teorica possiamo cercare di spiegare a quali domande l'uso combinato dei principi della meccanica quantistica e dei calcolatori elettronici può dare una risposta adeguata.

Per prima cosa è possibile uno studio conformazionale di molecole, ioni, radicali, ecc., ossia della disposizione geometrica ottimale dei nuclei atomici all'interno della struttura molecolare. Si tratta di un aspetto piuttosto importante in quanto dalla "forma" delle molecole spesso dipende la reattività e il più efficace metodo per individuare la geometria molecolare consiste nelle tecniche di diffrazione ai raggi X che, come noto, sono applicabili a sostanze cristalline o comunque solide. Viceversa la struttura di sostanze allo stato liquido o gassoso non può essere evidenziata in questo modo.

Per determinare la geometria di una molecola per via teorica, si spostano gli atomi dalle loro posizioni valutando le conseguenti variazioni dell'energia totale del sistema. La geometria corrispondente alla minima energia è quella più favorita, ossia è la geometria di equilibrio della molecola.

Una seconda importante risposta che si ottiene dai calcoli quantomeccanici riguarda la stabilità dei sistemi chimici. Per fare un semplice esempio, consideriamo la molecola di idrogeno, H_2, la cui energia è stata valutata per via teorica in 31.8 eV (elettronvolt). L'energia dell'atomo di idrogeno è di 13.6 eV, per cui dalla combinazione di due atomi a dare la molecola si ha un guadagno di 4.6 eV. Questa è l'energia di legame della molecola H_2 e corrisponde all'energia che occorre fornire per "rompere" la molecola e riottenere i due atomi separati.

L'importanza della valutazione delle energie di legame del sistema è notevole. Ogni reazione chimica, partendo da certi reagenti iniziali, conduce a nuovi prodotti finali coinvolgendo la rottura di alcuni legami e la formazione di altri. Se i legami di partenza sono deboli è evidente che la reazione sarà più facile in quanto tali legami si romperanno senza che si debba fornire troppa energia (ad esempio sotto forma di riscaldamento).

Non solo. Se il prodotto finale contiene nuovi legami particolarmente forti vuol dire che esso sarà stabile e che non tenderà a decomporsi o ad alterarsi.

Al contrario, la presenza di deboli interazioni sarà indice del fatto che la molecola tenderà con facilità a reagire con altre sostanze con cui venga a contatto, modificando il corso della reazione.

Dalla conoscenza delle energie in gioco in una reazione è possibile prevedere la fattibilità o meno di un processo chimico.

Senza contare che la conoscenza più approfondita dei meccanismi che governano le reazioni chimiche, e quindi gran parte dei fenomeni biologici, consente l'ottimizzazione di tali processi.

9. Conclusioni

Oggi come oggi la *chimica computazionale*, una disciplina relativamente giovane, è ancora in una fase interpretativa. Il passaggio alla fase di maggior interesse pratico della previsione a tavolino di fenomeni chimico-fisico-biologici è cominciato anche se sono stati compiuti solo pochi passi.

Per dare un'idea dello sviluppo di questa disciplina, si può dire che le note scientifiche pubblicate nel mondo nel corso del 1970 su questo argomento, quattrocento, sono decuplicate nel corso di un decennio (nel 1980 sono apparse circa 4000 pubblicazioni di chimica teorica).

La recente attribuzione del premio Nobel 1981 per la chimica a due chimici teorici, K. Fukui e R. Hoffmann, che fa seguito a quelli ottenuti da R.S. Mulliken nel 1966 e da I. Prigogine nel 1977, indica il crescente interesse del mondo scientifico verso questa branca della chimica.

Molti laboratori universitari e qualche grosso centro industriale (la Kodak, la General Electric, la Montedison in Italia) hanno sviluppato centri di ricerca teorica in cui l'analisi computazionale si sposa con quella sperimentale. Se ancora oggi molta attenzione viene prestata alla riproduzione e alla interpretazione dei dati sperimentali acquisiti, la chimica teorica ha comunque cominciato a spingersi più in là, sconfinando nel campo delle previsioni tanto che sempre più spesso si sente parlare di esperimenti effettuati col calcolatore.

Se la chimica computazionale potrà rappresentare un valido aiuto nel campo della sintesi chimica è forse presto per dirlo, ma è certo che se ciò avverrà una vera e propria rivoluzione si sarà prodotta rispetto alla millenaria tradizione sperimentale.

Il microcalcolatore "single-chip"

M. DE BLASI
Istituto di Scienze dell'Informazione
Università di Bari

A. GENTILE
P. PAPOFF
Centro di studio sulla Chimica Analitica Strumentale del C.N.R.
Università di Pisa.

1. Introduzione

L'avvento del microprocessore (μP), interamente realizzato su singola piastrina di silicio, sembrò rappresentare dieci anni fa un traguardo – più che un inizio – nella realizzazione di circuiti ad alta densità di integrazione. In realtà, proprio per il continuo ed impetuoso progresso tecnologico realizzato in questi ultimi anni nel campo dei circuiti integrati, è stato possibile includere nello stesso chip un numero sempre più rilevante di dispositivi, con il risultato che, accanto ai recenti potenti μP a 16 e a 32 bit, ha preso forma un componente affatto diverso, i cui connotati rassomigliano a quelli di un completo calcolatore, sia pure di caratteristiche particolari (microcalcolatore, μC).

La data di nascita del microcalcolatore single-chip risale al 1977, quando la Intel, utilizzando nuove tecnologie in grado di integrare circuiti con 17.000 transistor equivalenti per chip, cominciò a produrre l'8048 che racchiudeva su un'unica piastrina un processore, 1K byte di ROM (Ready Only Memory) e 64 byte di RAM (Random Access Memory).

Nell'8048 erano compresi inoltre un timer/contatore e delle porte d'I/O per un totale di 27 linee. Per questo insieme di caratteristiche, anche se alcune di esse di capacità estremamente ridotte, fu pienamente giustificato il termine di microcalcolatore per l'8048.

In corrispondenza alle sue elevate possibilità di colloquio, il μC venne orientato – come produzione – al controllo del funzionamento di tutti quei dispositivi di larga diffusione quali elettrodomestici, automobili, unità periferiche di calcolatori, in cui il tipo di elaborazione da compiere è estremamente semplice, i dati da trattare sono in numero piuttosto limitato, sì da richiedere ridotte capacità di memoria interne; i programmi, orientati verso un uso ben specifico e definiti una volta per sempre. Essi sono scritti su una ROM e diventano parte integrante e caratteristica del single-chip. In questo senso il single-chip diviene un dispositivo controllore rigidamente dedicato a scopo prefissato. Questa caratteristica lo differenzia particolarmente da un vero calcolatore.

Il costo di un μC, e per le sue caratteristiche intrinseche ed in quanto orientato verso un larghissimo mercato, risulta essere estremamente basso, quanto quello di un qualsiasi circuito integrato. Su esso però viene addizionalmente ad incidere il costo di mascheratura della ROM, tanto più quanto meno grande sarà il numero di pezzi prodotti.

Attualmente, i processi tecnologici sono in grado di fornire capacità d'integrazione molto superiori (50.000 ÷ 100.000 transistor) a quelle del primo μC single-chip. Questa maggiore capacità di integrazione si traduce, per gli attuali μC, rispetto ai predecessori, in maggiore capacità di memoria e maggiore potenza di elaborazione, permettendone l'utilizzazione in campi di applicazione sempre più vasti. Inoltre, alcuni tipi di essi sono espandibili tramite dispositivi esterni, per cui è possibile progettare "sistemi basati su μC single-chip" al posto di "sistemi basati su μP". Questa espandibilità verso l'esterno permette, fra l'altro, di scrivere programmi su ROM programmabili (PROM, EPROM e simili), e quindi di superare l'ostacolo della mascheratura delle ROM, non accettabile per la sperimentazione o per produzione di piccola serie. Molte aree applicative potranno pertanto essere ricoperte sia da sistemi basati sui μP che da quelli basati sui μC single-chip, con la differenza, per questi ultimi, di una sensibile riduzione di componenti separati.
In questo articolo chiameremo, appunto, sistemi a "pochi chip" i sistemi basati su μC single-chip e sistemi a "molti chip" quelli basati sui μP. Ciò detto, una visione generalizzata dei sistemi μC single-chip quale è quella che stiamo delineando, unitamente al fatto di avere integrato su di un unico chip un intero calcolatore, comportano una serie di considerazioni e pongono un insieme di domande che ancora una volta debbono tener conto della tecnologia attuale, della evoluzione futura, delle interazioni di questi dispositivi con l'ambiente circostante, del "modo nuovo" di vedere un calcolatore il cui limite più grosso sarà probabilmente il numero dei punti accessibili dall'esterno.

Questo articolo vuole evidenziare alcune caratteristiche peculiari dei μC in termini anche di prospezioni future. In questo senso, per ragioni di maggiore chiarezza, l'argomento verrà, sia pure schematicamente, suddiviso secondo alcuni aspetti scelti tra i più significativi.

2. Struttura generale dei μC single-chip

Un μC single-chip incorpora almeno 5 sub-unità (fig.1):

a) il processore
b) la memoria programmi
c) la memoria dati
d) le porte d'I/O
e) il timer/contatore

In alcuni casi, nel single-chip sono integrate anche altre unità, (fig. 2) come:

f) un'unità seriale d'I/O di tipo UART (Universal Asynchronous Receiver/Transmitter)
g) una circuiteria di "bus expansion"
h) un circuito per alimentazione di "stand-by"
i) un convertitore analogico/digitale
l) un multiplexer analogico.

Tenuto conto, inoltre, che attualmente sono in produzione [1] dei microsensori al silicio di vario tipo, non è difficile prevedere che essi possano trovar posto sullo stesso chip del μC. In fig. 2 abbiamo voluto anticipare questo impiego, inserendo nello schema generale, come standard, anche un sensore.

Poichè, in genere, le case produttrici forniscono non singoli tipi ma famiglie di μC single-chip, l'entità e la combinazione dei diversi componenti per ciascun tipo può variare largamente.

a) *Il processore*

Poichè per i μC single-chip non sono state previste applicazioni intensive in calcolo, il set d'istruzioni è in genere poco potente per l'elaborazione aritmetica dei dati, mentre è particolarmente potente nelle operazioni tipiche di acquisizione e controllo, quali trasferimento dati, elaborazione di bit, logiche, ecc.

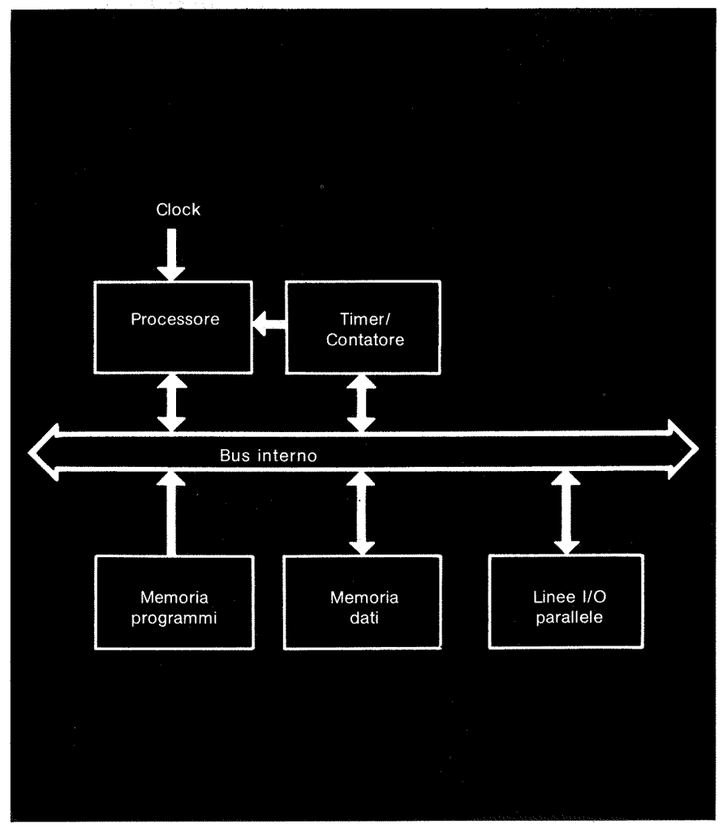
L'Intel 8051 per es., ultimo uscito della famiglia 8048, incorpora, oltre alla CPU, un altro processore chiamato booleano, che si incarica di tutte le operazioni su bit e permette di convertire in software le equazioni logiche di un circuito digitale [3].

b) *La memoria programmi*

Nei single-chip dedicati ad una particolare applicazione i programmi sono congelati in una memoria a sola lettura. Tale memoria deve essere sufficiente anche per contenere tabelle di look-up e decodifica necessarie, per es., per stampanti ed altre unità periferiche, per la generazione di caratteri.

Essa consiste di una ROM o EPROM (Erasable Programmable

Fig. 1 – Schema a blocchi di un semplice microcalcolatore single-chip.



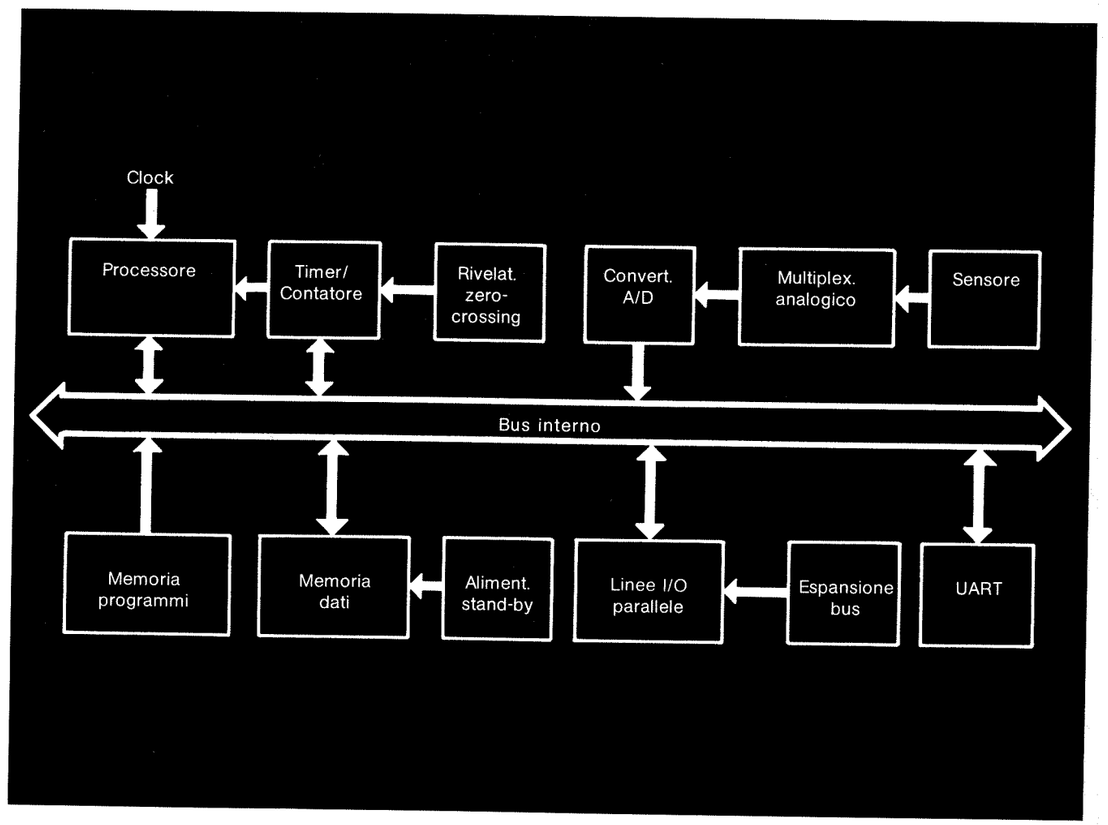


Fig. 2 - Schema a blocchi di un microcalcolatore single-chip complesso.

ROM) che può essere di 1K, 2K o 4K byte. I 4K byte attualmente raggiungibili sono adeguati per molte applicazioni di controllo di strumentazione.

Per quanto riguarda l'accesso a tabelle di dati, esso è una richiesta molto frequente nella elaborazione di dati in generale, ma ancor più nella elaborazione su single-chip: per questo troviamo, nella famiglia MK3870 della Mostek un registro "Data Counter" dedicato all'accesso di dati sequenziali. In questo tipo di applicazioni questo contatore diventa altrettanto importante del Contatore di Programma [2].

c) *La memoria dati*

Sul μC single-chip è fornita una RAM per contenere le variabili del programma. In particolare, nel controllo di unità periferiche, si tratta di immagazzinare buffer di 64 o 80 caratteri, quali richiesti, ad es., da stampanti o terminali. La capacità della RAM varia da 64 byte a 256 byte.

In alcuni casi, oltre alla RAM dati, esiste anche una RAM cosiddetta eseguibile. Per esempio, nel MK3872 la memoria programmi è ROM dalla posizione Ø alla 4031 ed è RAM dalla 4032 alla 4095.

Nell'INS 8070 della National, la memoria RAM di 64 byte è anche eseguibile, cioè può essere usata sia per dati che per programmi.

d) *Le porte parallele d'I/O*

Nel single-chip sono integrate generalmente da 3 a 4 porte parallele di 8 linee l'una, per cui si raggiungono da 24 a 32 linee parallele d'I/O.

Dato che i μC single-chip sono orientati alle funzioni di acquisizione e controllo, e che in essi vengono a mancare i bus di dati e di indirizzi, in quanto le memorie sono interne, la maggioranza dei piedini del chip sono utilizzati come piedini d'I/O.

e) *Il timer/contatore*

È presente un timer per la generazione di interrupt ad intervalli di tempo costanti.

f) *Unità UART*

Questa unità permette di collegare dispositivi che richiedono interfaccia di tipo seriale (ad es. stampanti, video, ecc.).

g) *Circuiteria di bus-expansion*

Avendo dedicato, come detto prima, la maggior parte dei piedini del μC alle porte d'I/O parallele, non è possibile portare, come accade sul μP, direttamente all'esterno dati e indirizzi.

Per connettere memorie esterne, alcuni single-chip incorporano una circuiteria che permette di condividere alcune linee d'I/O parallele con il bus interno di dati ed indirizzi.

h) *Circuito per alimentazione di "stand-by"*

Esso consente di commutare dalla tensione di alimentazione a quella fornita da una batteria di due celle al nichel-cadmio ricaricabili, sufficiente a fornire alimentazione di stand-by alla sola memoria dati. Così, al "power fail", si possono caricare in memoria quei dati ed informazioni che non si vogliono perdere e quindi ricommutare alla alimentazione normale quando la tensione di rete ritorna: da quel momento il calcolatore ricomincia a funzionare provvedendo a ricaricare automaticamente la batteria. Esempio di μC che adottano un circuito di alimentazione di stand-by sono Mostek MK3872, AMI-S4200, National INS 8048.

Alcuni μC single-chip hanno addirittura una istruzione che pone l'intera unità in stand-by quando non usata (μC CMOS AMI-S4200).

Quest'ultima caratteristica è molto utile in condizioni d'impiego in cui non sia possibile utilizzare l'alimentazione di rete.

i) *Convertitore analogico-digitale*

l) *Multiplexer analogico*

Una sezione interessante è anche quella, fornita su alcuni μC single-chip, che permette di effettuare degli input analogici (come Intel 8022, Motorola MC6805R2, Texas TMS2100). Oltre ad un convertitore AD, quasi sempre esistono un multiplexer analogico e un rivelatore di zero-crossing.

3. Famiglie di μC single-chip

A titolo esemplificativo, senza nessuna pretesa di completezza, sono presentate in fig. 3 alcune famiglie di μC single-chip. Invariabilmente, almeno un membro della famiglia è "ROM-less", cioè si basa su una EPROM o RAM esterna per la memoria programmi. Ciò è dovuto alla necessità di provare i programmi prima di procedere alla mascheratura definitiva in ROM.

Una famiglia rappresentata nella figura è la MK 3870 della Mostek, che è la versione μC single-chip della famiglia μP F8 della Fairchild. Come si vede, si passa da versioni con zero ROM e 64 byte di RAM a versioni con 4K byte di ROM e 64 byte di RAM. Per alcune combinazioni di ROM e RAM esistono modelli che hanno una porta seriale di I/O e altri che hanno il circuito di stand-by.

Altre famiglie riportate sono l'Intel 8048 e la National INS 8048. Quest'ultima ha avuto origine dalla prima ed è con essa compatibile. Degni di nota in questo ambito l'esistenza di un modello (8022) dotato di convertitore A/D e di un altro (8741) con un registro per comunicazione con un μP, che lo rende particolarmente idoneo come controllore di unità periferiche. Altri modelli più potenti si sono aggiunti successivamente a quelli indicati; si accennerà ad essi in altri paragrafi dell'articolo.

Un'altra famiglia rappresentata è la M6805 della Motorola. Essa pure presenta un modello con convertitore A/D. Inoltre, è disponibile una versione CMOS per quelle applicazioni per le quali è necessaria una bassa potenza di alimentazione.

Infine, in fig. 3 è rappresentata la famiglia INS 8070 della National il cui interesse maggiore consiste nel fatto di possedere un modello (INS 8073) il cui linguaggio di macchina è il BASIC. Infatti, nella ROM di programmazione di 2.5 K byte trova posto un interprete "TINY BASIC".

Le famiglie descritte si riferiscono a μC a 8 bit. Nel settore dei 4 bit esistono pure dei μC molto diffusi co-

Fig. 3 - Principali famiglie di microcalcolatori single-chip.
○ MK 3870
□ Intel 8048
Δ INS 8048
∇ INS 8070
◇ M 6805.
PS = Porta Seriale,
SB = RAM Stand-By,
MS = Master-Slave,
AD = Convert. A/D,
TB = Interpr. Tiny Basic

ROM \ RAM	0	64	128	256
4 K	○	○ SB		Δ
3 K	○	○		
2.5 K		∇ TB		
2 k	○ PS	○ PS, SB □ AD ◇ AD	□ Δ ◇ CMOS	
1 k	○ PS	○ PS □ MS Δ ◇		
0		○ PS □ Δ ∇	□ Δ ◇	Δ

me per esempio il COP della National e il TMS 2100 della Texas. La famiglia COP comprende, come ultimo nato, un modello estremamente interessante: il COP 2440. Si tratta di un μC con 2 CPU che possono elaborare parallelamente. Il TMS 2100, invece, ha un convertitore A/D ad 8 bit, una interfaccia ad alta tensione ed una PLA (Programmable Logic Array) a 32 termini.

Un cenno a parte dobbiamo fare per il TMS 9940 della Texas che non è normalmente riportato come un μC single-chip, ma fa parte di una famiglia, la TMS 9900, di μP a 16 bit. Il TMS 9940 incorpora 2K byte di ROM e 128 byte di RAM, il che non è sufficiente a farlo considerare un μC single-chip, ma è un indice dell'evoluzione futura che sicuramente porterà a μC single-chip con CPU potenti a 16 bit.

4. Allontanamento del μC dall'architettura di Von Neumann

Il modello di calcolatore proposto da Von Neumann nel 1945 ed adottato sostanzialmente fino ai nostri giorni, facente uso di un'unica memoria centrale contenente dati ed istruzioni, è parzialmente abbandonato in questi dispositivi. Infatti, in questi, sono integrate due memorie separate: una a sola lettura per i programmi ed una a lettura/scrittura per i dati. Ciò corrisponde anche ai principi attualmente adottati di evitare che le istruzioni siano modificate dal programma stesso, per ragioni di leggibilità del programma, manutenzione, ecc. Si noti che la modificabilità dell'istruzione era invece proclamata da von Neumann come uno dei punti di forza del suo modello, poichè si potevano trattare le istruzioni come dati.

La separazione della memoria dati da quella programmi richiama anche concetti di meccanismi di protezione, poichè i due spazi di indirizzi sono separati. Non saranno, ad es., permessi dall'hardware errori come salti a locazioni di memoria contenenti dati, come pure operazioni aritmetiche in locazioni di memoria contenenti istruzioni. Ma la caratterizzazione di queste nuove architetture non si esaurisce a questo punto. Abbiamo già visto l'introduzione di un nuovo registro, il Data Counter, che svolge, per la memoria dati, lo stesso ruolo che il Program Counter svolge per la memoria programmi. Così si viene a tener conto del frequente uso, che si fa nella programmazione, di dati organizzati sequenzialmente, cioè di tabelle.

Non mancano naturalmente i compromessi fra le due architetture. Per esempio, anche questo lo abbiamo già incontrato, in un particolare μC esiste oltre alla memoria dati, un'altra RAM così detta eseguibile.

È interessante notare come questi ed altri concetti, che solo ultimamente si stanno introducendo nel campo della "computer architecture", trovino già posto in questi modesti dispositivi.

Altre caratteristiche innovative, che utilizzano le crescenti possibilità della tecnologia, si stanno introducendo, nel rispetto dei vincoli architetturali di compatibilità posti dai modelli precedenti e dalle famiglie di μP da cui i single-chip provengono.

5. Configurazione interna

In dispositivi a così alta integrazione come i single-chip, assume particolare importanza la disposizione delle singole parti al loro interno, al fine di ridurre il più possibile l'area occupata dalle connessioni rispetto a quella utilizzata per le funzioni da svolgere.

L'utilizzazione del chip è per es. migliore nelle memorie che nella logica, poichè nelle prime le connessioni sono molto più regolari. Hughes e Chappel [5] affermano che una porta logica di CPU, con le sue interconnessioni, occupa 8 volte lo spazio di una cella di RAM statica e 80 volte quello di una cella di ROM.

Il problema dell'interconnessione diventa sempre più pressante all'aumentare della complessità del processor, ed in particolare della larghezza del "data-path" interno.

In fig. 4 è riportata una disposizione delle parti principali, e l'area da esse occupata nell'interno di un μC single-chip [5]. Come si può vedere da questa figura, anche i μC single-chip seguono la tendenza generale dei sistemi basati su μP, che è quella che vede la CPU svolgere un ruolo sempre più modesto rispetto alle altre unità funzionali.

L'aumento futuro di densità d'integrazione e di dimensione di area del chip verrà sempre di più destinato a potenziare la parte memoria ed I/O rispetto al processore. Hughes e Chappel [5] prevedono che sino all'80% dell'area dei futuri chip VLSI sarà dedicato a memoria.

6. Il problema dell'emulazione

Per lo sviluppo di sistemi basati sui μC single-chip si seguono metodi analoghi a quelli dei sistemi basati sui μP.

Come è noto, sviluppare un sistema basato su μP significa non soltanto scrivere e provare i programmi relativi, ma anche progettare ed assemblare il sistema hardware con i vari circuiti di supporto, di memoria, d'I/O, ecc.

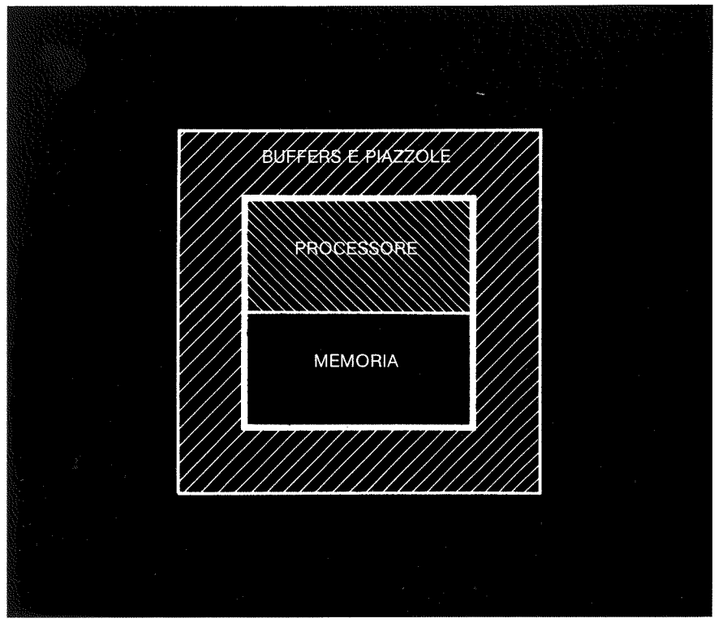
Nel caso dei μC single-chip, una parte di questi circuiti è compresa sul single-chip, per cui il problema dal punto di vista hardware risulta semplificato. D'altra parte, nel single-chip, un fattore fastidioso è che alcune di queste unità, come per es. le memorie, non sono accessibili dall'esterno.

Per questa ragione, le case costruttrici forniscono delle schede di "emulazione", cioè dei sistemi funzionalmente identici a un μC single-chip, ma con punti di accesso in numero sufficiente alle necessità su esposte. Inoltre, su queste schede sono montate delle RAM, che prendono il posto delle ROM interne al single-chip (nella fase di "debugging" e messa a punto del sistema).

Quando i programmi saranno completamente provati, l'utente li invierà alla casa costruttrice la quale provvederà alla mascheratura della ROM residente sul single-chip.

La scheda di emulazione fa eventualmente parte di un sistema di sviluppo dotato di unità periferiche quali floppy disk, stampanti, video, ecc., per fornire un ambiente di programmazione più confortevole.

Fig. 4 - Configurazione interna dei microcalcolatori single-chip



Un modo di realizzare gli emulatori è quello di "esplodere" il μC single-chip nelle corrispondenti unità della famiglia di μP con cui è compatibile, avendo cura, naturalmente, di sostituire le ROM con delle RAM. Tale approccio è considerato da alcuni Autori non completamente corretto (vedi per es. [7]), poichè non vengono usati nella fase di "debugging" e di sviluppo componenti dello stesso tipo di quelli che verranno usati successivamente nella fase di impiego effettivo. È per questa ragione che le case costruttrici includono normalmente nelle famiglie di μC single-chip un membro "ROM-less". In fase di "debugging" e messa a punto si potrà così usare un componente che è del tutto identico con quello che verrà usato in produzione, con l'unica differenza che il programma risiede su una RAM o una EPROM esterna.

Le soluzioni generalmente adottate lungo questa ultima direzione sono:

a) i dati ed indirizzi sono connessi all'esterno, multiplexati su linee d'I/O (l'Intel 8031 è la versione con RAM esterna dell'Intel 8051);

b) è presente una PROM di "piggy-back" cioè esterna, ma inseribile sullo stesso chip del μC. Ciò è molto utile perchè, senza effettuare connessioni, si mantengono le prerogative di un single-chip. È questo l'approccio della Mostek con il suo MK3874 nell'ambito della famiglia 3870;

c) La versione ROM-less è prodotta con un numero di piedini maggiore per ospitare il bus di dati ed indirizzi. E questo il caso del Rockwell 6500/1 che è prodotto in due versioni: a 40 piedini (standard) e a 64 piedini (per emulazione).

Di queste soluzioni, le più interessanti appaiono essere le ultime due, perchè più vicine alle caratteristiche del prodotto finale. In più, la versione b) risponde, oltre che alle necessità di emulazione, anche a quelle di realizzazione di prototipi o prodotti in piccola serie, per i quali, come abbiamo avuto modo di osservare, può non essere giustificata la spesa della mascheratura della ROM.

7. Necessità di autotesting

I μC single-chip tendono a divenire sempre più dei sistemi complessi senza aumentare proporzionalmente i loro terminali di accesso verso l'esterno. Mentre è aumentata notevolmente la densità d'integrazione dei circuiti integrati, ed ancora aumenterà, non è possibile aumentare di pari passo il numero di piedini di tali circuiti.

La conseguente inaccessibilità di un gran numero di nodi interni ha portato alla necessità di introdurre in questi dispositivi una capacità di autotesting [4]. Anche questa è una caratteristica che fino ad ora è stata predominio esclusivo di grossi e grossissimi calcolatori.

Il Motorola M6805 incorpora sul chip una ROM aggiuntiva che contiene un programma che controlla in continuazione, ad intervalli regolari di tempo, la RAM, la ROM, le linee di indirizzi e di dati, gli interrupt, le linee d'I/O ed il timer, cioè circa il 95% della funzionalità del μC.

Una considerazione siamo indotti a fare a questo punto di fronte a questi componenti: l'introduzione di certe caratteristiche strutturali innovative è stata resa possibile dall'elevato grado di integrazione raggiunto, ma al tempo stesso può essere una conseguenza necessaria senza la quale gli stessi dispositivi non potrebbero sopravvivere.

8. Come l'evoluzione della tecnologia influenza quella dei microcomputer single-chip.

Due fattori, il traguardo di un milione di transistor per chip, promessoci per i prossimi anni [4], e l'architettura dei μC single-chip, che già nell'attuale versione ben si presta ad utilizzare un tale eccesso di tecnologia, ci inducono a chiederci come saranno i μC single-chip del prossimo futuro.

Piuttosto che fare delle previsioni, per le quali si rimanda all'ottimo articolo già citato [4], ci è sembrato utile registrare in questo paragrafo alcuni esempi di evoluzione di μC single-chip avutasi in conseguenza della realizzazione di quelle che potevano essere considerate le promesse della tecnologia di qualche anno fa. Per inciso, il passato ha insegnato che in questo campo le promesse sono state sempre mantenute e spesso superate.

Evoluzione Intel 8048-8051

L'8051 è una versione in tecnologia 1980 dell'8048, uscito nel 1977. L'avanzamento può essere misurato da un unico dato molto significativo: 17.000 transistori sull'8048 contro i 60.000 dell'8051.

Come hanno utilizzato queste ulteriori potenzialità i progettisti della Intel? Innanzitutto la indirizzabilità della memoria è passata da 4Kbyte a 64 Kbyte. Questo è stato un progresso notevole che ha fatto uscire il μC single-chip da campi di applicazione molto limitati, per renderli di utilizzabilità sempre più generali.

In secondo luogo si è avuto, come c'era da aspettarsi, un incremento in ROM, da 1K a 4K, ed in RAM, da 64 a 128 byte. Inoltre, l'8051 è sino a 10 volte più veloce dell'8048 ed esibisce un'ulteriore caratteristica molto interessante, già citata precedentemente, quella cioè di possedere un processor booleano in aggiunta alla CPU.

Evoluzione MK3870-MK3872.

Il dato più significativo nel passare dalla tecnologia del 3870 a quella del 3872 è il raddoppio di ROM e di RAM ottenuto con un chip di area soltanto di poco superiore: 173x208 mil nel 3870 contro 173 x 268 mil del 3872.

Evoluzione COP 400-COP 2440

Qui le aumentate possibilità tecnologiche sono state sfruttate per produrre il μC 2440 con 2 CPU, che possono eseguire in parallelo la stessa istruzione su dati differenti, oppure 2 istruzioni diverse nella memoria programmi. Anche questo è un concetto rivoluzionario che porta architetture multiprocessor integrate su un singolo chip.

Evoluzione TMS 1000-TMS 2100

In questo caso, l'aumentata densità d'integrazione del 2100 rispetto al TMS1000 è stata usata per un convertitore A/D a 8 bit, una interfaccia ad alta tensione ed una PLA a 32 termini per conversioni di codici.

9. Conclusione

Da questa rapida panoramica, si può vedere come le linee di tendenza della tecnologia dei μC single-chip puntino ad aumentare la capacità di memoria interna, ad accrescere le potenzialità come la massima capacità di memoria indirizzabile e ad aumentare la potenza di calcolo portando i processori interni da 1 a 2. A ciò si aggiunge la presenza di caratteristiche come l'alimentazione di stand-by, modelli in versione CMOS e simili.

Possiamo dire, in conclusione, che tutto questo è solo un inizio, tenendo presente che dai 2.500 transistor del primo μP a 8 bit, l'INTEL 8008, si è passati ai 17.000 per l'8048 e agli attuali 60.000 per l'8051. Mentre i 17.000 transistori erano da considerarsi la soglia per la nascita del μC single-chip, i 60.000 sono solo una tappa intermedia verso il traguardo su menzionato del milione di transistori.

Il problema, a questo punto, sarà, per dirla con Gordon Moore [8], di sapere cosa faremo di tanta tecnologia.

10. Bibliografia

[1] S. MIDDELHOEK, J.B. ANGELL, D.J.W. NOORLAG, *"Microprocessors get integrated sensors"*, IEEE Spectrum, February 1980.

[2] H.W. DOZIER, R.S. GREEN, *"Single-chip microcomputer expands its memory"*, Electronics, May 11, 1978.

[3] B. KOEHLER, *"Microcontroller doubles as Boolean processor"*, Electronic Design, May 24, 1980.

[4] D.A. PATTERSON, C.H. SEQUIN, *"Design Considerations for Single-Chip Computers of the Future"*, IEEE Transactions on computers, February 1980.

[5] J. HUGHES, P. CHAPPELL, *"Memory-to-memory for future controllers"*, Electronic Design, November 8, 1980.

[6] P.M. RUSSO, *"VLSI Impact on Microprocessor Evolution, Usage and System Design"*, IEEE Journal of Solid State Circuits, August 1980.

[7] D. STARBUCK, D. PEETERS, K. HOGAN, R. EUFINGER, *"Expanded package speeds design with new, on-chip microcomputer"*, Electronics, June 8, 1978.

[8] G. MOORE *"VLSI: some fundamental challenges"*, IEEE spectrum, April 1979.