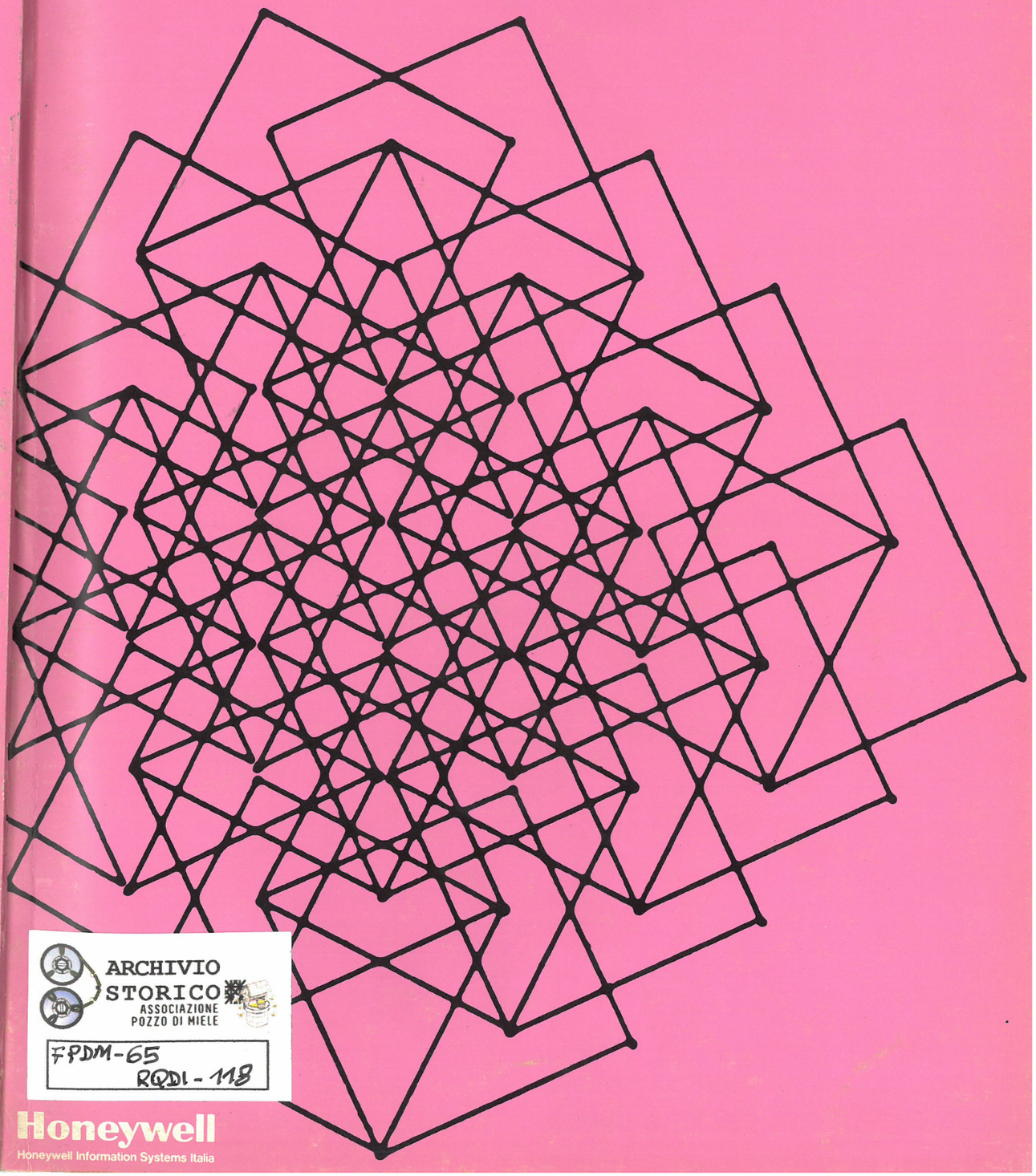
Quaderni di informatica 20

Rassegna di tecnologia ed applicazioni degli elaboratori elettronici – Anno X - Numero 2 - 1983



Honeywell
Honeywell Information Systems Italia

Honeywell

Honeywell Information Systems Italia

Quaderni di informatica

Rassegna di tecnologia ed applicazioni degli elaboratori elettronici
Anno X - Numero 2 - 1983

Sommario

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.

Le basi di dati relazionali

VIRGINIO CHIODINI

Honeywell Information Systems Italia
Centro di Ricerca e Progettazione
Pregnana Milanese

1. Introduzione

A partire dal 1970, anno in cui E.F. Codd pubblicò il famoso articolo "A relational Model of Data for Large Shared Data Banks", le basi di dati relazionali sono state oggetto di un crescente interesse. Ristretta all'inizio nell'ambito del mondo accademico e della ricerca teorica, l'idea del modello relazionale cominciò ben presto a materializzarsi nelle prime realizzazioni a livello di prototipi. Dopo circa dodici anni, i sistemi di basi di dati relazionali cominciano ad uscire dall'ambito sperimentale e ad essere installati commercialmente sui sistemi di elaborazione dei dati.

Questo articolo, oltre a descrivere i concetti base di quello che è chiamato "modello relazionale", fornisce alcune caratteristiche di talune realizzazioni commerciali, e inoltre notizie su alcuni campi oggetto di ulteriore ricerca teorica o tecnologica.

Verranno inoltre fornite alcune valutazioni sui benefici potenziali del modello relazionale in confronto con i tradizionali modelli di basi di dati "gerarchico" o "reticolare".

All'interno del modello relazionale verrà infine illustrato un esempio specifico ("Interactive Data Base System", installato sul sistema Honeywell DPS-4).

2. Cos'è un sistema di base di dati relazionale

La proprietà fondamentale del Modello Relazionale di Base di Dati è costituita dal formato tabellare con cui i dati sono descritti e presentati all'utilizzatore.

Ogni tabella è costituita da righe e colonne in cui le righe corrispondono ai records e le colonne rappresentano i campi all'interno dei records. La figura 1 è un esempio di una struttura di dati relazionali che descrive l'informazione "impiegato" in una tabella che ha per nome *"Impiegati"*.

Questa tabella contiene solo sei righe o records, uno per ogni impiegato.

Un'effettiva tabella, per un'installazione tale da richiedere una base di dati, conterrà ovviamente un numero di records molto superiore.

Per ogni impiegato la tabella fornisce quattro informazioni (campi): il codice, il nome, la qualifica e il dipartimento.

Fig. 1 Struttura relazionale dei records "*Impiegati*".

MATRICOLA	NOME	QUALIFICA	DIPARTIMENTO
03856 Y	MAGGIONI	PROGRAMMATORE	G 210
01987 K	SCOTTI	PROGRAMMATORE	K 540
04328 W	POMINI	ANALISTA	G 310
02318 H	COSTA	PROGRAMMATORE	G 210
00842 W	AGLIERI	ANALISTA	K 540
02954 W	CASATI	SISTEMISTA	K 560

CODICE	MANAGER	LOCALITÀ
G 210	MARTINELLI	FIRENZE
K 560	MAURI	MILANO
G 310	GUIZZARDI	MILANO
K 540	CESANA	TORINO

Fig. 2 - Tabella *"Dipartimenti"*.

La fig. 2 illustra invece la tabella *Dipartimenti,* strutturata in tre tipi di informazioni. Si noti che la colonna *Dipartimento* in *Impiegati* e la colonna *Codice* in *Dipartimenti* contengono informazioni omogenee.

In termine di modello relazionale si dice che le due colonne sono definite su "domini" di valori omogenei.

Il formato fisico in cui sono registrati i dati è nel modello relazionale totalmente irrilevante, a differenza dei modelli gerarchico e reticolare in cui la collocazione dei records è un vettore essenziale di informazione.

Questo non vuol dire che la struttura fisica dei dati non abbia un suo senso nei sistemi relazionali, specie per il fatto che essa condiziona pesantemente la velocità di accesso e registrazione.

Il problema delle prestazioni del sistema di base di dati è però totalmente enucleato e indipendente dall'aspetto di disegno dei dati.

Ciò che importa è che il modo in cui i dati sono singolarmente definiti coincide con la visibilità che di questi ha l'utente.

Se il modello relazionale si basa su strutture dati costituite da records dello stesso formato, in cosa esso differisce dal tradizionale modello di dati a flussi sciolti? La differenza consiste in alcune norme precise che qualificano un modello relazionale di base di dati:

- Ogni tabella contiene un solo tipo di record
- Ogni record (riga) è composto da un numero fisso di campi, ad ognuno dei quali è assegnato un nome esplicito, noto al sistema
- I campi sono tutti distinti e definiti a livello elementare (atomico); sono vietati campi a vettori (Repeating groups)
- In una tabella non possono esistere due records con lo stesso contenuto. Per questo può essere definito un campo, o un insieme di campi, sul record, il cui contenuto individua univocamente il record stesso. Tale insieme di campi viene definito "chiave primaria" della tabella
- La successione dei records all'interno della tabella è irrilevante ai fini dell'informazione che la base di dati fornisce
- Ogni campo assume il suo contenuto estraendolo da un "dominio" di possibili valori
- Lo stesso dominio può essere comune a diversi tipi di campi, divenendo una sorgente di valori di campo per più colonne appartenenti allo stesso tipo di record o a records diversi.
- Nuove tabelle possono essere continuamente generate dal sistema di base di dati, congiungendo tabelle diverse che abbiano in comune dei valori di campi appartenenti ad ognuna di esse e definiti sullo stesso dominio.

La possibilità di generare continuamente una nuova tabella a partire da due o più tabelle esistenti é realmente l'elemento più qualificante di una base di dati relazionale.
Con i metodi di accesso tradizionale a flussi sciolti, tale combinazione e composizione di nuovi tipi di record a partire dai records esistenti è totalmente a carico della logica dei programmi.

3. I fondamenti teorici del modello relazionale

Il modello relazionale è derivato dallo sviluppo teorico della teoria algebrica delle relazioni. Tale derivazione teorica ha fortemente influenzato la terminologia e le notazioni, rendendo la materia spesso inutilmente complessa e spesso generando ambiguità a causa della varietà di definizioni usate nelle pubblicazioni della comunità accademica. Questo risulta tanto più evidente dal confronto con i concetti circolanti nell'ambiente dell'elaborazione dati commerciale.

La seguente tabella traduce in termini di comune linguaggio EDP una lista di termini usati nelle letteratura tecnica.

- Relazione = Tabella o tipo di record
- N-upla (Tupla) = riga o record
- Attributo = nome di colonna o tipo di campo
- Elemento = campo
- Grado di una relazione = numero di colonne nella tabella
- Cardinalità di una relazione = numero di righe nella tabella
- Relazioni binarie = Tabella con due colonne
- Relazione N-aria = Tabella con N colonne
- N-upla = record in una tabella con N colonne

Il termine "dominio" non trova corrispondenti nel tradizionale linguaggio; esso può essere definito come l'insieme di tutti i possibili valori di un certo tipo da cui uno specifico tipo di campo può assumere i suoi contenuti.

La definizione formale di relazione è la seguente:

"Dati degli insiemi S_1, S_2, ...S_n (non necessariamente distinti), R è una relazione su questi "n" insiemi se esiste un insieme di n-uple, ciascuna delle quali ha il primo elemento contenuto in S_1, il secondo elemento contenuto in S_2 e così via. In altri termini R è un sottoinsieme del prodotto cartesiano
$S_1 \times S_2 \times ....\times S_n$.

S_j è lo j-esimo dominio di R; R ha grado "n".

Operazioni su tabelle

La definizione dei dati in forma tabellare, consente di definire le operazioni di manipolazione in termini di operatori algebrici sulle tabelle. Elemento qualificante delle operazioni relazionali è che la risultante dell'applicazione di essi alle tabelle è sempre una tabella. In questo caso si parla di proprietà di "chiusura" del modello relazionale.

Vengono qui presentati i tre operatori più importanti applicabili alle tabelle:

Selezione

La selezione è la più semplice delle operazioni relazionali.

Quando è applicata ad una tabella, la selezione genera una nuova tabella contenente tutte le colonne della prima ma solo un sottoinsieme delle righe della prima. Tale sottoinsieme viene determinato individuando le righe in cui il valore di uno o più campi soddisfa ad una condizione di selezione. La fig. 3 ad esempio mostra l'effetto dell'applicazione dell'operatore di selezione alla tabella *Impiegati.* In questo caso l'operatore immette nella tabella risultante solo le righe della tabella per cui la colonna *Dipartimento* ha il valore = G210.

Proiezione

L'operazione di proiezione applicata ad una tabella genera una tabella risultante contenente tutte le righe della prima ma solo un sottoinsieme delle colonne. Va notato che la tabella risultante può contenere alcune righe tra loro identiche. Poichè non è consentito ad una tabella avere due o più righe tra loro uguali, una sola di esse viene conservata. La figura 4 mostra l'effetto della proiezione della tabella *Impiegati* sulle colonne *Nome* e *Dipartimento.* In questo caso la risultante non contiene duplicati.

Gli operatori di Selezione e Proiezione possono essere applicati in successione alla stessa tabella. Il primo selezionerà le righe che soddisfano alla condizione espressa, il secondo proietterà la risultante su un sottoinsieme delle colonne.

Congiunzione

L'operatore di "congiunzione" agisce su due tabelle, fondendo ogni riga della prima con una riga della seconda in base ai valori assunti in esse da una colonna che è detta colonna di congiunzione.

La fig. 5 mostra la risultante dell'operatore di congiunzione applicato alle tabelle *Impiegati* e *Dipartimenti* in base alla colonna *Di-*

Fig. 3 - Selezione degli impiegati che lavorano nel dipartimento *G210.*

CODICE	NOME	QUALIFICA	DIPARTIMENTO
03856 Y	MAGGIONI	PROGRAMMATORE	G 210
02318 H	COSTA	PROGRAMMATORE	G 210

Fig. 4 - Proiezione delle colonne *Nome* e *Dipartimento* della tabella *Impiegati.*

NOME	DIPARTIMENTO
MAGGIONI	G 210
SCOTTI	K 540
POMINI	G 310
COSTA	G 210
AGLIERI	K 540
CASATI	K 560

MATRICOLA	NOME	QUALIFICA	DIPARTIMENTO	MANAGER	LOCALITÀ
03856 Y	MAGGIONI	PROGRAMMATORE	G 210	MARTINELLI	FIRENZE
02318 H	COSTA	PROGRAMMATORE	G 210	MARTINELLI	FIRENZE
02954 W	CASATI	SISTEMISTA	K 560	MAURI	MILANO
04328 W	POMINI	ANALISTA	G 310	GUIZZARDI	MILANO
01987 K	SCOTTI	PROGRAMMATORE	K 540	CESANA	TORINO
00842 W	AGLIERI	ANALISTA	K 540	CESANA	TORINO

Fig. 5 - Congiunzione delle tabelle *Impiegati* e *Dipartimento* sulle colonne *Dipartimento* in *Impiegati* e *Codice* in *Dipartimento*.

partimento presente in *Impiegati* e alla colonna *Codice* in *Dipartimento*.

L'operazione di congiunzione avviene nel seguente modo:

- Si prende la prima riga della prima tabella e si cerca nella seconda una riga che assuma un valore uguale alla prima in corrispondenza alla colonna di congiunzione.
- Ogni volta che una riga così è individuata, essa viene fusa con la prima formando una nuova riga contenente la somma delle colonne delle partecipanti meno 1 (la colonna di congiunzione che non viene ripetuta)
- Si prosegue fino all'esaurimento delle righe della seconda tabella
- Si prende la riga successiva della prima tabella e si ricercano nella seconda le righe per cui combaciano i valori della colonna di giunzione, ripetendo l'operazione descritta prima
- Il ciclo si ripete fino all'esaurimento delle righe della prima tabella. In tale modo, per esaurire l'operazione di congiunzione, la seconda tabella viene letta tante volte quante sono le righe della prima.

È ovvio che affinchè l'operazione di congiunzione abbia senso, i valori assunti dalle due colonne di giunzione debbono essere omogenei.

In altre parole, non avrebbe alcun senso congiungere due tabelle in base ad una colonna definita su un dominio di valori di "data" e un'altra definita su un dominio di valori di "prezzo".

L'operatore di congiunzione può a sua volta essere combinato con gli operatori di selezione e proiezione. All'operazione di congiunzione parteciperanno solo le righe selezionate e la tabella risultante conterrà solo le colonne prescelte.

Il modello relazionale fa uso anche di altri operatori quali:

Unione, Intersezione e Differenza che operano su tabelle omogenee e *Divisione* di tabelle aventi una colonna definita su un dominio comune.

Tutti questi operatori sono utilizzati nella elaborazione delle informazioni. L'illustrazione di essi va tuttavia oltre gli scopi di questa descrizione, che tende semplicemente ad illustrare la logica con cui opera il modello relazionale.

I cammini di accesso ai dati

Elemento qualificante del modello relazionale è che in esso la richiesta di accesso ai dati è espressa solo indicando "quali" sono i dati desiderati e non "come" ad essi si accede.

Non esiste nelle basi di dati relazionali la possibilità di indicare il cammino d'accesso, quali ad esempio i verbi "FIND LAST WITHIN SET" del sistema reticolare CODASYL o "GET NEXT WITHIN PARENT" del DL/1.

Nei modelli gerarchico e reticolare di basi di dati i cammini di accesso fanno parte integrante della struttura dei dati definita dall'utente. Un programma operante su una struttura gerarchica di dati utilizza esplicitamente questo cammino d'accesso per navigare dal record padre ai records figli. Qualora tale cammino d'accesso non esista, la logica di programmazione deve essere alterata.

Nel modello relazionale, la definizione dei dati non include alcuna specificazione di cammini d'accesso. Per reperire i dati parecchie vie d'accesso sono potenzialmente disponibili. Questo significa che la definizione dei dati è invariante rispetto alla ristrutturazione degli stessi, mentre le funzioni di manipolazione sono veramente ad alto livello.

Un'altra caratteristica deducibile da quanto detto è l'assoluta simmetria dell'accesso ai dati qualunque sia il cammino disponibile.

Nei modelli gerarchico e reticolare la logica di programmazione è funzione del cammino d'accesso seguito. Ad esempio nel modello reticolare "CODASYL" la sintassi del verbo usato per accedere ad un record è diversa a seconda che il cammino di accesso sia "VIA SET" oppure diretto. Questo vincola un programma alla struttura fisica dei dati e al cammino d'accesso.

Ogni ristrutturazione che alteri tali elementi ha impatti pesanti sulla logica stessa dei programmi esistenti.

Riassumendo, gli aspetti essenziali delle operazioni relazionali sono:

- Le operazioni relazionali agiscono su intere tabelle ossia su insiemi di records
- La risultante dell'applicazione di uno o più operatori ad una o più tabelle è sempre una tabella (ossia un insieme di records)
- Le operazioni sulle tabelle qualificano solo quali sono le tabelle oggetto e quali righe e colonne di esse sono desiderate.

Per contro le caratteristiche dei modelli gerarchico e reticolare sono così riassumibili:

- I sistemi gerarchico e reticolare operano su singoli records e consentono l'elaborazione di uno di essi alla volta
- Le operazioni sono esplicitamente qualificate dalla indicazione del cammino d'accesso da seguire. Tale cammino d'accesso deve essere predefinito al momento della descrizione della struttura dei dati e ne è parte integrante. Ogni alterazione di questo induce cambiamenti nella logica dei programmi.

4. I linguaggi relazionali

Dopo aver descritte le principali operazioni relazionali parleremo ora dei formalismi linguistici con cui tali operazioni possono essere espresse dall'utente. La descrizione di questi linguaggi permetterà una valutazione più precisa della facilità d'uso e della potenzialità espressiva del modello relazionale. Un numero notevole di linguaggi sono stati definiti fino ad oggi. Solo alcuni di essi tuttavia sono usciti da un ambito di realizzazione prototipale e hanno cominciato ad essere commercialmente diffusi. Anche se gli esempi che verranno presentati si riferiscono ad operazioni di lettura di dati, occorre notare che i linguaggi relazionali offrono l'intera gamma di funzioni di lettura, aggiornamento, inserimento, e cancellazione di dati.

I linguaggi relazionali sono raggruppabili in quattro classi principali:

Linguaggi algebrici

Questi linguaggi forniscono al programmatore gli operatori relazionali di selezione, proiezione, divisione, unione, intersezione e differenza. La tabella risultante è ottenuta tramite la applicazione successiva degli operatori sulle tabelle oggetto. La successione delle operazioni viene definita anche tramite l'uso di parentesi di raggruppamento.

La fig. 6a fornisce un esempio di tali operazioni. Il più noto dei sistemi di base di dati relazionali che utilizzano linguaggi algebrici è il "NOMAD".

Linguaggi predicativi

Questi hanno una forma meno procedurale dei precedenti. Essi sono sostanzialmente costituiti da espressioni che definiscono le tabelle risultanti in base a dichiarative che utilizzano le notazioni della logica predicativa di primo livello con l'aggiunta delle operazioni sugli insiemi.

Le espressioni dei linguaggi predicativi sono del tipo:

<Oggetto> WHERE <Predicato>

in cui:

<Oggetto> individua l'insieme delle tabelle da cui la tabella risultante è derivata.

<Predicato> specifica le condizioni cui debbono soddisfare le righe delle relazioni oggetto che parteciperanno alla formazione della tabella risultante.

In questa categoria di linguaggi il più noto è QUEL *(Query Language)* installato sul sistema di base di dati INGRES.

Un esempio di esso è illustrato nella fig. 6b.

Linguaggi rappresentativi

Contengono tutte le funzioni dei linguaggi predicativi. In aggiunta ad esse i linguaggi rappresentativi consentono di inserire espressioni di interrogazione dentro altre espressioni di interrogazione.

L'inserimento correlato di espressioni di interrogazione dentro altre è consentito fino ad un livello qualsiasi. I linguaggi rappresentativi forniscono inoltre strumenti espressivi per confrontare insiemi di valori con altri appartenenti allo stesso dominio, permettendo la correlazione della risultante di un'interrogazione con quella in cui è inserita.

In qualche modo si può affermare che i linguaggi rappresentativi combinino funzioni di tipo algebrico con funzioni di tipo predicativo.

Questo consente una maggiore flessibilità ed accresce la gamma delle possibilità espressive a disposizione del programmatore, anche se rende non univoca la modalità con cui un tipo di richiesta può essere espresso.

Il più noto dei linguaggi rappresentativi è l'S.Q.L. (Structured Query Language) supportato dai due sistemi di base di dati *SOL/Data System e Oracle.*

Un esempio è illustrato in fig. 6c.

Linguaggi visivi

Questa classe di linguaggi esprime le operazioni relazionali non in termini di frasi descrittive ma utilizzando simboli grafici visivi forniti sui video-terminali.

L'esempio classico è il Query By Example (Q.B.E.). Con esso il terminalista può formulare l'interrogazione "compilando" su un modulo visivo che descrive la tabella un esempio di una possibile risposta.

Gli strumenti espressivi disponibili sono:

- la digitazione di elementi che servono come esempio (variabili) che debbono essere sottolineati. Questi sono utilizzati per esprimere gli operatori di proiezione e congiunzione
- la digitazione di elementi costanti (non sottolineati) per esprimere gli operatori di selezione

Inoltre la funzione "P" (print) indica le colonne della relazione risultante.

Nell'esempio presentato in fig. 6d. FIRENZE è un elemento costante, ROSSI e ANALISTA sono digitati come esempio di una possibile risposta. L'elemento G210 che appare in entrambi i moduli di tabella, indi-

(IMPIEGATI [DIPARTIMENTO = CODICE]
(DIPARTIMENTO [LOCALITÀ = "FIRENZE"])) [NOME, QUALIFICA]

Funzionalmente:

SELEZIONA la tabella DIPARTIMENTO per LOCALITÀ = FIRENZE

↓

CONGIUNGI la risultante con IMPIEGATI sulla base delle colonne DIPARTIMENTO e CODICE

↓

PROIETTA la risultante sulle colonne NOME e QUALIFICA

a) Linguaggio algebrico

```
RANGE OF I IS IMPIEGATI
RANGE OF D IS DIPARTIMENTO
RETRIEVE (I. NOME, I. QUALIFICA)
WHERE    I. DIPARTIMENTO = D. CODICE AND
         D. LOCALITÀ     = "FIRENZE"
```

b) Linguaggio di calcolo (QUEL)

```
SELECT    NOME, QUALIFICA
FROM      IMPIEGATI, DIPARTIMENTI
WHERE     DIPARTIMENTO = CODICE AND
          LOCALITÀ     = "FIRENZE"

oppure:
SELECT    NOME, QUALIFICA
FROM      IMPIEGATI
WHERE     DIPARTIMENTO IN
                  (SELECT CODICE
                  FROM DIPARTIMENTI
                  WHERE LOCALITÀ = "FIRENZE");
```

c) Linguaggio di rappresentazione (SQL)

CODICE	NOME	INDIRIZZO	QUALIFICA	DIPARTIMENTO
	P. ROSSI		ANALISTA	G 100

MANAGER	LOCALITÀ	DIPARTIMENTO
	FIRENZE	G 100

d) Linguaggio grafico (Query by example)

Fig. 6 - Formulazione in vari linguaggi relazionali della interrogazione: *"Leggi il nome e la qualifica di tutti gli impiegati che lavorano nei dipartimenti situati a Firenze".*

ca la richiesta di congiunzione tra le due tabelle *Impiegati* e *Dipartimenti* sulla base delle rispettive colonne *Dipartimento* e *Codice.*

5. Tabelle primarie e derivate

Nella descrizione fatta finora si sono implicitamente definiti due tipi di tabelle:

Le "Tabelle Primarie" che corrispondono ai dati effettivamente registrati.

Le "Tabelle Derivate" che sono definite come risultante dell'applicazione degli operatori relazionali alle tabelle primarie.

Le Tabelle Derivate possono essere logicamente suddivise in tre classi:

- **Le Liste** – sono le tabelle derivate che in genere non vengono fisicamente registrate. Ogni aggiornamento delle tabelle primarie da cui una lista è generata non si riflette su di essa dopo che è stata formata.
- **Le Istantanee** – sono le tabelle derivate che vengonó fisicamente registrate, qualificate dalla data e dall'ora in cui la tabella derivata è stata materializzata. Dopo che la materializzazione è avvenuta, la tabella derivata è insensibile a modifiche apportate alle tabelle primarie da cui è originata.
- **Le Viste** – sono delle "Tabelle Primarie Virtuali" in quanto appaiono come dati effettivamente registrati in tabelle primarie. Esse sono invece costituite dalla sola formula espressiva che le descrive come risultanti di operazioni su tabelle primarie effettive.

  Ad esempio l'espressione riportata in fig. 6 può essere usata per definire una "vista" cha dà l'impressione che esista una tabella con due colonne (*Nome* e *Qualifica*) e con tante righe quanti sono gli impiegati la cui sede di *Dipartimento* è *Firenze.*

La vista rappresenta, in altri termini, una "finestra dinamica" aperta sulle tabelle primarie. Ogni aggiornamento apportato a queste è continuamente riflesso dalla "vista".

In alcuni linguaggi relazionali le "viste logiche", come le tabelle primarie, possono essere oggetto dell'applicazione di operazioni relazionali e "congiunte" con altre tabelle primarie o altre "viste".

Esse possono inoltre essere usate per definire altre "viste".

Le "viste" relazionali hanno una potenzialità di impiego vastissima.

Ad esempio esse permettono di filtrare preventivamente i dati di base a cui un utilizzatore può accedere ed inoltre di fornire a questi una visione semplice di dati che possono essere in realtà l'esito di un'interrogazione estremamente complessa sulle tabelle primarie.

La vastità di impiego potenziale delle "viste logiche" è tuttavia limitata da alcune restrizioni pesanti esistenti sulle possibilità di utilizzo di esse per l'aggiornamento dei dati. Infatti, in generale, le operazioni di aggiornamento possono avere per oggetto una vista logica solo se le righe di questa tabella virtuale sono derivate in corrispondenza biunivoca da righe di una tabella primaria.

Questa limitazione ha un impatto rilevante specie sull'utilizzo delle vedute per applicazioni programmatiche, a causa della disimmetria generata tra operazioni di lettura e aggiornamento.

Notevoli sforzi tecnici vengono spesi attualmente per fornire strumenti espressivi in grado di rimuovere questa limitazione.

Una particolare soluzione è quella fornita dall'IDBS, come verrà più avanti illustrato.

6. Linguaggi relazionali e interfacce di programmazione tradizionali

I linguaggi relazionali offrono una indubbia potenza espressiva e sono indubbiamente il veicolo naturale con cui esprimere interrogazioni interattive immesse via video terminale. È tuttavia evidente che la facilità di programmazione potrebbe trarre un notevolissimo vantaggio dalla disponibilità di un linguaggio di accesso ai dati potente e ad alto livello.

Il problema maggiore derivante dall'accoppiamento di linguaggi relazionali con i tradizionali linguaggi di programmazione: COBOL, FORTRAN, PL/1 sta nel fatto che questi operano su un record alla volta, mentre la risultante di un'espressione relazionale è sempre un insieme di records.

Le soluzioni finora adottate tendono a distinguere due momenti:

- la definizione della tabella da derivare
- la presentazione della tabella derivata al programma ospite

Una prima possibilità di esecuzione consiste nel materializzare comunque la tabella derivata in un flusso che può essere letto record per record dai tradizionali verbi di accesso ai flussi.

L'alternativa è quella di mantenere la restituzione dei records sotto il controllo del sistema di base di dati relazionale.

Ad esempio un'espressione di interrogazione S.Q.L. può essere ospitata in un programma COBOL o PL/1 avendo associato ad essa un "cursore". Le righe della tabella derivata sono presentate al programma una alla volta dietro comando esplicito di lettura emesso sul cursore.

L'integrazione tra linguaggi tradizionali e linguaggi relazionali di accesso ai dati è comunque indubbiamente complessa e talvolta presenta soluzioni alquanto artificiose. Queste difficoltà sono frutto del divario creatosi tra la potenza espressiva delle espressioni relazionali e la capacità espressiva alquanto limitata dei linguaggi tradizionali.

Alcune ricerche in corso tendono a favorire un'integrazione più naturale tra i linguaggi relazionali ed alcuni dei linguaggi di programmazione più evoluti come PASCAL e PROLOG.

7. Il Disegno delle tabelle primarie nelle basi di dati relazionali

Un altro campo oggetto di intensa analisi teorica in seguito all'introduzione del modello relazionale riguarda il disegno della composizione delle tabelle primarie. Obbiettivo principale in quest'area è il supporto alla definizione di formati records che rimangano stabili nella fase di evoluzione della base di dati.

Le strutture delle tabelle esistenti non dovrebbero presentare necessità di ristrutturazione in seguito all'introduzione di nuove applica-

zioni, pur essendo soggette a possibili estensioni.

Queste teorie di disegno tendono alla definizione di strutture di records normalizzate attraverso l'analisi delle dipendenze funzionali tra i campi.

Sono stati identificati cinque stadi di normalizzazione delle tabelle.

La "prima forma normale" concerne la "forma" di un tipo di record.

Per soddisfare la prima forma normale ogni riga della tabella deve contenere lo stesso numero di campi, escludendo quindi campi vettoriali a numero di elementi variabile (Repeating Groups).

La "prima forma normale" più che una tecnica di disegno è un elemento qualificante delle basi di dati relazionali secondo quanto detto prima.

La "seconda forma normale" è violata quando un campo non chiave in una tabella dipende funzionalmente non dall'intera chiave ma da un sottoinsieme dei campi chiave.

Ad esempio nel seguente record;

PARTE	MAGAZZINO	QUANTITÀ	INDIRIZZO-MAGAZZINO

CHIAVE

l'Indirizzo Magazzino è funzione solo di *Magazzino*. Una struttura di record di questo tipo può generare i seguenti problemi:

- l'indirizzo del magazzino è ripetuto in ogni record che descrive una parte a magazzino
- la modifica dell'indirizzo di un magazzino, comporta la modifica di tutti i records in cui tale indirizzo appare
- Questa ridondanza può causare inconsistenza di informazioni
- Qualora un magazzino non contenesse alcuna parte, non ci sarebbe modo di conservare l'informazione sul suo indirizzo.

Per normalizzare l'informazione al secondo livello, il record precedente deve essere scomposto in due records distinti:

PARTE	MAGAZZINO	QUANTITÀ

CHIAVE

MAGAZZINO	INDIRIZZO MAGAZZINO

CHIAVE

La "terza forma normale" è violata quando un campo non chiave dipende da un altro campo non chiave. Ad esempio nel record:

IMPIEGATO	DIPARTIMENTO	LOCALITÀ

CHIAVE

se ciascun dipartimento ha sede in un certo luogo, il campo *Località* dipende non solo dal campo chiave *Impiegato* ma anche del campo non chiave *Dipartimento*. I problemi che tale struttura crea sono analoghi ai precedenti:

- la località sede di dipartimento è ripetuta in ogni record impiegato
- se la sede cambia, occorrerà modificare tutti i records che descrivono impiegati con quella sede di lavoro
- se un dipartimento fosse senza impiegati non esisterebbe l'informazione sulla località in cui ha sede.

Portare l'informazione alla terza forma normale consiste nello scomporre il record precedente in due records diversi:

IMPIEGATO	DIPARTIMENTO

CHIAVE

DIPARTIMENTO	LOCALITÀ

CHIAVE

Oltre le tre forme normali "classiche" del modello relazionale sono state identificate da lavori teorici successivi una quarta e una quinta forma normale che consentono di individuare i raggruppamenti atomici di informazioni da cui ogni informazione più complessa può essere ottenuta utilizzando gli operatori di congiunzione.

Sul processo di normalizzazione va però detto che solo la prima forma normale è essenziale al modello relazionale. Non sempre è utile raggiungere le forme di normalizzazione successive.

Le forme di normalizzazione successive infatti, pur prevedendo anomalie in fase di aggiornamento, penalizzano la velocità di reperimento delle informazioni in quanto un insieme di colonne, anzichè essere raggruppato in un record e quindi estraibile con una sola lettura, deve invece essere composto operando congiungimenti di più tabelle.

È perciò necessario bilanciare i pro e i contro del processo di normalizzazione. Recenti pubblicazioni suggeriscono dei processi di "denormalizzazione" delle tabelle, in modo da ottenere almeno migliori prestazioni nel congiungimento di più tabelle.

8. Le prestazioni dei sistemi di base di dati relazionali

Un dubbio di fondo pesa sull'efficienza dei sistemi relazionali.

Esso riguarda la velocità con cui il sistema è in grado di rispondere ad un'interrogazione. Tale perplessità deriva soprattutto dall'impossibilità per il programma di "indicare" al sistema la esistenza di cammini di accesso rapido ai dati, quali indici che abbiano per chiave l'argomento di ricerca.

L'impressione è che il sistema, in assenza di tale "guida", scateni sempre scansioni sequenziali sui dati con effetti pesantissimi sul carico di lavoro del sistema.
In realtà i sistemi relazionali, prima di iniziare l'esecuzione di un'interrogazione, determinano sempre la strategia di navigazione tra i dati che minimizzi l'utilizzo di risorse della macchina sia come quantità di calcolo interno che come numero di accessi a disco.

Questa fase di ottimizzazione viene svolta a caldo in caso di lancio di interrogazioni estemporanee oppure è eseguita "una tantum" in fase di compilazione, quando le interrogazioni sono immerse in un programma ospite.

La fase di ottimizzazione si serve di informazioni catalogate nel dizionario della base dati quali:

- l'esistenza di indici che consentano di ritrovare i records cercati senza scandire l'intera tabella
- informazioni statistiche sulla dimensione delle tabelle e su eventuali criteri di ordinamento fisico dei records

Ad esempio nell'operazione di congiunzione di fig. 5 l'esistenza di un indice sulla seconda tabella, avente come chiave la colonna di congiunzione con la prima, avrebbe consentito di individuare immediatamente quali righe della seconda tabella combaciavano con una riga estratta dalla prima.

Se per di più un indice sulla colonna di congiunzione fosse esistito anche sulla prima tabella, la ricerca di valori coincidenti avrebbe potuto essere eseguita operando solo sugli indici.

Ancora si potrebbe calcolare, in assenza di indici, se non sia preferibile ordinare (se già non lo sono) fisicamente le righe delle due tabelle in base alla colonna di congiunzione e poi operare la fusione.

Un ulteriore elemento di intelligenza del sistema relazionale consiste nello scomporre un'espressione di interrogazione in una sequenza ottimale di operazioni sulle tabelle.

In questo l'analisi delle operazioni da parte del sistema è supportata in modo determinante dai fondamenti teorici del modello relazionale, che consentono di prevedere in modo esatto l'esito dell'applicazione di una successione qualsiasi di operatori.

Alcuni sistemi ottimizzano ulteriormente la fase esecutiva, traducendo preventivamente le sequenze ad alto livello in linguaggio macchina in modo da evitare a caldo ricicli interpretativi.

Non esiste alcuna ragione tecnica per cui i sistemi relazionali siano condannati a prestazioni peggiori di quelle delle tradizionali basi di dati.

L'unico pericolo è che l'estrema facilità con cui le interrogazioni, anche complesse, possono essere formulate, renda l'utilizzatore poco cosciente della mole di lavoro da cui la macchina è sollecitata.

L'utilizzatore deve curare la messa a punto del sistema dal punto di vista delle prestazioni, creando cammini di accesso ad indice per le vie di ricerca più usate e bilanciando il beneficio ottenuto da un accesso immediato col sovraccarico causato dalla necessità di mantenere aggiornati tali indici. Tale messa a punto può essere eseguita gradualmente, senza minimamente alterare la struttura delle informazioni.

L'interesse crescente accentrato sulle basi di dati relazionali ha inoltre stimolato la ricerca e lo sviluppo di nuove tecnologie software e hardware focalizzate all'ottenimento di migliori livelli di prestazioni, specializzate per funzioni dei sistemi relazionali.

9. Le "Data Base Machines"

Le "Data Base Machines" possono essere definite come processatori specializzati nella esecuzione ad alta velocità di interrogazioni sulla base di dati. Connesse ad uno o più sistemi ospiti, esse concentrano in se stesse tutte le operazioni sulla base di dati.
Già studiate per i sistemi di base di dati gerarchici e reticolari, esse non hanno mai offerto su questi miglioramenti decisivi di prestazioni, in quanto la granularità a livello record delle richieste rendeva le linee di connessione alla macchina ospite delle strozzature insormontabili per l'alto flusso di richieste.
Il modello relazionale offre invece la possibilità di inviare alla "Data Base Machine" una richiesta ad alto livello, scaricando su di essa l'intera esecuzione della richiesta e ricevendo come risposta un insieme di record.
In questo modo il sovraccarico delle linee di comunicazione tra il sistema ospite e la "Data Base Machine" viene ridimensionato.
La centralizzazione della base di dati, dà alla "Data Base Machine" la possibilità di servire una grande varietà di sistemi ospiti, offrendo un'interfaccia di accesso ai dati ad alto livello utilizzabile anche da microcomputers o terminali intelligenti.

La fig. 7 illustra un tipico schema architetturale di accoppiamento "Data Base Machine" - sistema ospite.

10. Alcune valutazioni complessive

Si è molto discusso sulla possibilità effettiva per i sistemi relazionali di rimpiazzare i modelli reticolari o gerarchici. La conclusione non è nettamente favorevole o contraria. Pur essendo notevolissime, le potenzialità del modello relazionale non sono integralmente sfruttabili da tutte le applicazioni e da tutti gli utilizzatori.

Gli utenti che non sono professionisti EDP sono sicuramente coloro che possono trarre più vantaggio dal modello relazionale, in quanto essi eseguono tipicamente interrogazioni estemporanee sulla base di dati.
Con i sistemi tradizionali il modo di formulare tali interrogazioni dipendeva dai cammini di accesso disponibili, costringendo le espressioni di interrogazione ad essere in un certo senso asimmetriche. Il modello relazionale offre invece una interfaccia simmetrica, semplice e ad alto livello per qualsiasi interrogazione, anche se tali modi espressivi potrebbero essere più semplici e soprattutto personalizzati per le diverse categorie di persone non esperte EDP.

Nella programmazione delle normali applicazioni il modello relazionale può offrire vantaggi o svantaggi a seconda del tipo di applicazione.

In alcune il modello relazionale offre indubbi vantaggi, specie là dove sono richieste letture e aggiornamenti di insiemi di records. In questi casi la logica di programmazione ne risulta semplificata e con essa risulta semplificata la comprensione e la manutenzione dei programmi.

Nei casi in cui la logica del programma deve essere legata alla elaborazione di un record alla volta, lo sforzo di programmazione nell'ambiente relazionale è uguale se non superiore a quello necessario nell'ambito delle basi di dati tradizionali. Un esempio è la gestione materiali in cui i cicli di implosione/esplosione delle parti debbono essere percorsi individualmente.

In altre applicazioni, la minore proceduralità dei sistemi relazionali è compensata dallo svantaggio derivante dalla necessità di manipolare diverse tabelle tra cui esiste una potenziale molteplicità di interconnessioni.

I modelli reticolari e gerarchici invece, con i loro cammini di accesso

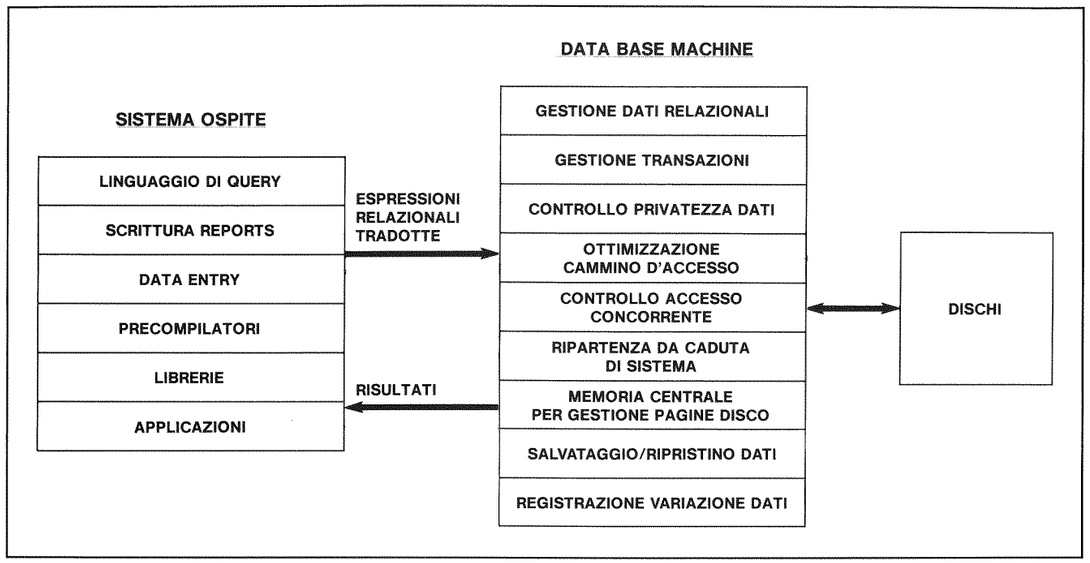


Fig. 7 - Connessione Sistema Ospite - Data Base Machine.

predefiniti, riescono a far cogliere immediatamente alcune correlazioni complesse.

Per gli amministratori della base di dati, la problematica posta dal modello relazionale è simile a quella posta dai modelli tradizionali.

Rimangono a carico dell'amministratore la scelta della struttura dei dati su disco, la messa a punto di queste per l'ottimizzazione delle prestazioni e la loro manutenzione.

Rimane a carico dell'amministratore anche la scelta delle procedure di salvataggio/ripristino dei dati.

Col modello relazionale, tuttavia, la semplicità delle interfacce tra struttura logica e fisica riduce notevolmente la quantità di opzioni e alternative disponibili e le semplifica considerevolmente.

La fase di disegno della base di dati risulta notevolmente semplificata dal modello relazionale, specialmente in seguito all'assunzione da parte del sistema del compito di organizzare, mantenere e ottimizzare i cammini di accesso e di navigazione tra i dati. La descrizione dei dati può essere concentrata quindi sulla loro struttura logica senza necessità di prevedere tutte le interconnessioni tra di essi, presenti e future.

Per quanto riguarda il disegno della struttura logica dei records, le modalità di analisi di essa sono simili in tutti i modelli di base di dati in quanto delle tecniche di disegno, quali la normalizzazione, sono comunque necessarie.

11. L'"Interactive Data Base System" (I.D.B.S.)

L'I.D.B.S., installato sul sistema *Honeywell DPS-4,* presenta un modello di base di dati che combina la semplicità di disegno e la flessibilità del modello relazionale con le esigenze di programmi applicativi scritti in linguaggio tradizionale.

Per l'I.D.B.S., come per il modello relazionale, i dati sono fisicamente registrati in formato tabellare.

Da ogni tabella fisica (formato record fisico) è possibile derivare una vista logica semplice (tabella logica) ottenuta per proiezione e selezione della tabella fisica.

Più viste logiche semplici possono essere fuse in una vista logica composta, definendo dei cammini di accesso che sequenziano tra loro le righe delle tabelle logiche in base a uno o più campi di concatenamento. Tali cammini di accesso consentono inoltre di sequenziare tra loro le righe di ogni tabella logica a parità di valore del campo di concatenamento. Le fig. 8 e 9 illustrano un esempio di definizione di viste logiche semplici e composte.

La fusione generata dalla vista logica composta è sostanzialmente diversa da quella risultante da un operatore di congiunzione. Infatti ogni vista logica composta conserva l'"identità" delle viste logiche semplici; in altre parole, ogni record della vista logica composta conterrà campi proiettati da una sola tabella fisica.

La fusione produce perciò in sostanza delle gerarchie di tabelle logiche. La fusione viene fisicamente realizzata mediante indici a cammino d'accesso generalizzato. Essi sequenziano i diversi valori dei campi di giunzione e di ordinamento secondario derivati dalle tabelle logiche e indirizzano, per ognuno di essi, la riga fisica da cui è proiettata la riga logica corrispondente.

Dal punto di vista dei programmi applicativi, una vista logica composta può essere elaborata come un tradizionale flusso indiciato con più tipi di records.

Le tabelle fisiche e le vedute logiche semplici possono essere elaborate come flussi tradizionali con un solo tipo di record.

Fig. 8a - Vista logica *"Dipendenti"*, proiettata dalla tabella fisica *"Impiegati"*.

NOME	CODICE DIPARTIMENTO
MAGGIONI	G 210
SCOTTI	K 540
POMINI	G 310
COSTA	G 210
AGLIERI	K 540
CASATI	K 560

Fig. 8b - Vista logica *"Sezioni"*, proiettata dalla tabella fisica *"Dipartimenti"*.

CODICE	LOCALITÀ
G 210	FIRENZE
K 560	MILANO
G 310	MILANO
K 540	TORINO

Fig. 9 - Fusione delle viste logiche *"Dipendenti"* e *"Sezioni"* sul campo di concatenamento *"Codice dipartimento"* e ordinamento ulteriore secondo *"Nome"* dipendenti.

G 210	FIRENZE
COSTA	G 210
MAGGIONI	G 210
G 310	MILANO
POMINI	G 310
K 540	TORINO
AGLIERI	K 540
SCOTTI	K 540
K 560	MILANO
CASATI	K 560

L'utilizzo di tecniche software avanzate e di nuovi strumenti hardware e firmware disegnati ad hoc per l'I.D.B.S. consente di ottenere, nell'elaborazione delle viste logiche complesse, livelli di prestazioni superiori a quelli ottenibili tradizionalmente dai flussi indiciati contenenti più tipi record.

Un elemento fondamentale dell'I.D.B.S. è la completa processabilità in lettura e in aggiornamento delle viste logiche semplici e composte, senza le limitazioni poste dalle viste logiche relazionali.

Nuove viste logiche semplici e composte possono essere definite in qualsiasi momento sopra le tabelle fisiche esistenti, senza alterare la struttura e l'operabilità sia di queste che delle viste logiche già esistenti.

In sostanza l'I.D.B.S., pur fornendo solo un sottoinsieme delle capacità operative sulle tabelle del modello relazionale, fornisce un insieme di funzionalità di base di dati completo e coerente con le capacità operative degli attuali linguaggi di programmazione.

Pianificazione dei sistemi di trasmissione dati

TREVOR HOUSLY*

1. Introduzione

La pianificazione e la progettazione sono aspetti preliminari importanti dello sviluppo di tutti i progetti, sia che si tratti di un sistema basato su elaboratore con rete di trasmissione dati che della semplice installazione di una nuova serratura sulla dispensa di cucina.

I primi sistemi di trasmissione dei dati basati sull'elaboratore presentavano molte "cadute". Con caduta di sistema si voleva indicare la condizione in cui il sistema non funzionava o non riusciva a raggiungere gli obiettivi prefissati dal progetto. Generalmente la caduta del sistema era imputabile 1) al fatto che hardware e software non erano in realtà molto adatti a costruire sistemi con trasmissione dei dati e 2) alla mancanza di esperienza per individuare le aree in grado di creare potenzialmente dei problemi. Oggi le cose sono cambiate: abbiamo accumulato una notevole esperienza alle nostre spalle e disponiamo di strumenti analitici che ci possono aiutare a prevedere le prestazioni della rete.

La progettazione e l'analisi di un sistema di grandi dimensioni sono attività assai complesse e costose che talvolta possono richiedere l'impiego di pacchetti di programmi di simulazione e di altri aiuti che richiedono tempo di elaborazione. Per i sistemi più piccoli (e quindi la maggior parte dei sistemi in linea e delle reti di comunicazione dati del mondo) sono state messe a punto diverse tecniche semplici che possono essere applicate per verificare la ragionevolezza o fattibilità delle ipotesi di partenza e controllare se il sistema immaginato ha molte probabilità di lavorare o se invece è prevedibile che si verifichino colli di bottiglia o che si creino problemi. Ciò consente al progettista di esaminare diverse alternative di sistema e di rete prima di proporre un progetto realizzabile come nella fig. 1. Un sistema realizzabile non solo soddisfa i requisiti funzionali richiesti nelle specifiche applicative, ma è in grado di smaltire il carico di lavoro richiesto con un tempo di risposta adeguato, ha una interfaccia efficiente con le persone che lo usano, è facilmente espandibile e il suo costo non è irragionevole.

Esaminiamo subito alcuni elementi tra i più noti che devono essere esaminati quando si pianifica e si sviluppa un sistema in linea.

(*)Il materiale di questo articolo è tratto dal libro dello stesso Autore «Sistemi di trasmissione dati e di teleelaborazione», collana dei Quaderni di Informatica, ed F. Angeli (1983).

2. Problemi di direzione

Uno dei problemi maggiori che si incontra nello sviluppo dei sistemi di trasmissione dati non è di tipo tecnico, ma direzionale. Il progetto e l'attuazione dei sistemi in linea differisce grandemente dal progetto e dallo sviluppo dei sistemi in batch. In un'installazione ben avviata che funziona in batch, il servizio di elaborazione dati e il servizio utente possono di solito unire le forze, specificare una applicazione, progettare e sviluppare senza far ricorso alle risorse del fornitore di *mainframe.*

Invece sviluppando un sistema in linea è necessario coinvolgere nello sforzo di progettazione e sviluppo organizzazioni esterne come le seguenti:

Fornitori di terminali. Il mercato offre centinaia di terminali diversi e può valer la pena esaminare le possibilità di impiegare terminali diversi da quelli offerti dal fornitore di *mainframe.* È possibile, così facendo, trovare terminali migliori, più versatili, più economici, di consegna più pronta degli analoghi prodotti offerti dal fornitore del *mainframe.*

Fornitori di mainframe. È probabile che si debba prendere contatto con il fornitore di *mainframe* per molte ragioni. Per esempio, se si vuole usare terminali di un terzo, ci si deve assicurare che quei terminali siano compatibili con lo hardware e il software fornito dal costruttore di *mainframe.* E su questa strada si possono incontrare immediatamente dei problemi, perchè costruttore di *mainframe* e fornitore di terminali possono non voler dialogare direttamente. Inoltre se è la prima volta che si affrontano sistemi in linea, è probabile che non si abbia una conoscenza approfondita dello hardware e del software di trasmissione dati e quindi sarà necessario fare affidamento sull'assistenza del costruttore di *mainframe.*

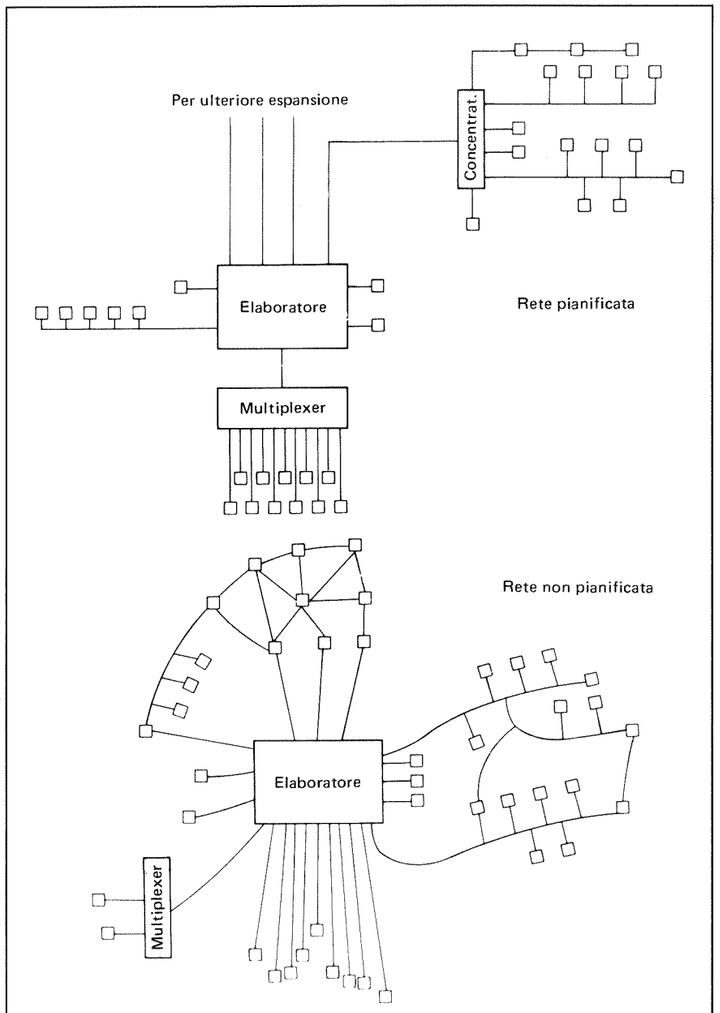


Fig. 1 - La pianificazione della progettazione della rete.

Consulente. Se si sta studiando un sistema in linea per la prima volta può essere utile richiedere i servizi di un consulente esterno per avere la necessaria assistenza durante le fasi di progettazione e sviluppo.

Software house. Per le stesse ragioni per cui si ricorre ai consulenti, si possono acquistare i servizi di una *software house* (cioè di un'azienda specializzata nella produzione di software). Nella fase di progettazione e sviluppo in linea è spesso necessario disporre di specialisti di capacità maggiori rispetto a quelli di cui abbiamo bisogno per mantenere in funzione il sistema una volta che esso sia stato reso operativo. Poiché gli specialisti di progettazione e sviluppo di sistemi in linea sono assai costosi, può essere conveniente affidare il lavoro a breve termine ad una *software house* o ad una casa di consulenza piuttosto che assumere persone così qualificate nell'organico aziendale. E anche in quest'ultima ipotesi, sarà poi difficile riuscire ad offrire loro una sufficiente varietà di lavoro in grado di mantenere un elevato interesse e un buon livello di motivazione anche dopo il completamento del progetto; del resto queste persone hanno una notevole mobilità: se tutte quelle condizioni non vengono rispettate essi si porranno rapidamente alla ricerca di nuove opportunità professionali.

Compagnie telefoniche. Se si intende installare terminali al di fuori dei locali dell'azienda, è necessario richiedere ed ottenere servizi di comunicazione dalla compagnia telefonica locale. Il numero delle compagnie telefoniche con le quali sarà necessario trattare dipenderà dal paese nel quale ci si trova e se si sta studiando o meno una rete internazionale.

Infrastrutture (energia elettrica, condizionamento dell'aria, ecc...). Il sistema in linea viene impiegato per applicazioni che hanno requisiti di affidabilità e ripristino molto più rigorosi di quelli dei sistemi in batch.

Un sistema di elaborazione in batch può anche non funzionare per diverse ore senza che nessuno, al di fuori del centro Edp, se ne accorga. Non così invece sui sistemi in linea: appena questo si ferma, tutti coloro che vi sono coinvolti, compresi gli utenti, lo sanno. Spesso si procede a duplicare in gran parte lo hardware in modo da non compromettere il funzionamento dell'intero sistema in caso di guasto di qualche unità. Naturalmente ciò vale anche per i servizi ausiliari e si dovranno predisporre delle alternative per il condizionamento dell'aria e l'erogazione dell'energia elettrica.

Ognuna delle organizzazioni elencate in precedenza ha qualche interesse da perseguire che può essere divergente da quello di altre. Il direttore del centro di elaborazione dati dovrà impegnarsi a fondo nell'attività di direzione richiesta per coordinare le organizzazioni esterne. Può capitare che il responsabile del centro non abbia l'esperienza necessaria per gestire questi problemi e che non si renda conto dei problemi fino a quando non è troppo tardi. Si sono verificati molti esempi in cui non sono stati rispettati i piani fissati per il progetto e sono stati mancati obiettivi concreti a causa dello scarso coordinamento svolto dal responsabile del progetto.

Per gestire i progetti di questo tipo e per coordinare le attività delle varie parti coinvolte si possono seguire procedure ben definite e collaudate. Se il direttore del centro di elaborazione dati o il responsabile del progetto capiranno sin dall'inizio che si possono verificare dei problemi, sarà possibile minimizzarne la frequenza e le dimensioni.

3. Problemi di progettazione

Abbandoniamo ora i problemi manageriali e limitiamoci ad esaminare gli aspetti tecnici che devono essere valutati con attenzione nella costruzione di un sistema di comunicazione dei dati: l'elaboratore, i programmi, le linee, i terminali e gli altri dispositivi ancora richiesti dal sistema. Durante la pianificazione di dettaglio del sistema si devono tener costantemente presenti alcuni criteri generali che si applicano a tutta l'attività di progettazione dell'intero sistema.

Questi criteri fondamentali offrono un punto di appoggio sul quale si può costruire l'intero progetto e che possono aiutare a scoprire ed articolare meglio i requisiti del sistema. Esaminiamo questi aspetti generali nei paragrafi seguenti.

1. Prestazioni del sistema

È un elemento la cui importanza è ovvia. Il sistema viene costruito per uno scopo e da esso ci si aspettano certe funzioni e certi servizi. Di solito i requisiti di prestazioni sia generali che specifici sono documentati come *specifiche funzionali* che spesso sono allegate, costituendone parte integrante, ad un contratto tra cliente e fornitore.

Le specifiche funzionali dovrebbero definire esattamente come deve essere elaborata ogni transazione, quali archivi siano richiesti, quali informazioni devono essere contenute negli archivi, quali devono essere le maschere video dell'input e dell'ouput e così via. Oltre che a rispondere a questi requisiti, al sistema si richiede anche di offrire certe prestazioni standard: se il sistema non è in grado di offrire le prestazioni richieste, di solito lo si considera fallito. Le prestazioni del sistema sono di solito specificate facendo riferimento al carico di lavoro che il sistema può gestire - cioè la quantità di lavoro che può essere eseguita dal sistema (throughput del sistema) e dalla velocità con la quale il sistema lavora.

Su un sistema con interrogazione *(enquiry)* in linea, la misura della velocità è di solito il tempo di risposta che se troppo lungo può creare problemi all'organizzazione utente.

Per esempio un sistema per la prenotazione dei posti di una compagnia aerea deve rispondere alle seguenti specifiche: il tempo di risposta medio deve essere inferiore o uguale a 3 secondi con un carico di 30.000 transazioni l'ora. Il sistema non sarà considerato soddisfacente se non riuscirà a smaltire il carico di lavoro e darà dei lunghi tempi di risposta.

In tutte le fasi durante le quali si elaborano le specifiche di un progetto si dovrà tener presente l'effetto che qualsiasi decisione del progettista ha sulle prestazioni del sistema. Per esempio, se si sta disegnando un grosso programma paghe per un sistema di elaborazione in batch e si sottostima della metà il numero di accessi a disco per ogni transazione ci si può trovare a gestire una procedura la cui esecuzione richiede di continuare anche per parte del turno successivo di lavoro. Tuttavia se si compie lo stesso errore progettando un sistema in linea potrebbe accadere che il tempo di risposta aumenti da 3 a 10 secondi, il che potrebbe comportare effetti rilevanti per la gestione del sistema.

2. Interfaccia con l'utente

La maggior parte dei sistemi di comunicazione vengono oggi impiegati da persone che interagiscono con il sistema in qualche modo. Gli utenti possono essere programmatori e sistemisti culturalmente preparati, operatori addestratissimi oppure direttori o altri che impiegano il terminale solo occasionalmente o ancora addirittura l'uomo della strada che ignora come si fa funzionare un terminale. Il sistema deve essere progettato in modo da offrire un'interfaccia efficiente a misura di quel particolare tipo di persona che sarà l'utente del sistema.

Fino a tempi recenti le persone che dovevano interfacciare un elaboratore in linea dovevano avere una formazione specialistica o erano programmatori o analisti o operatori ben addestrati a cui veniva insegnato esattamente come comunicare con il sistema. In questi casi gli utenti si adattavano alle esigenze dell'elaboratore per vedere i dati piuttosto che chiedere di essere serviti dall'elaboratore secondo i propri desideri. Man mano però che si riduce il prezzo dei sistemi di elaborazione, questi trovano applicazioni sempre più vicine all'utente finale e vengono integrati processi di produzione nei processi di distribuzione e nelle attività di vendita.

Quando mettiamo un terminale di un elaboratore in fabbrica, le persone che lo impiegheranno per interfacciare il sistema saranno certa-

mente privi di una lunga formazione specifica sugli elaboratori.

Si tratta di operai che dopo una spiegazione assai rapida e sommaria si troveranno ad operare sulla tastiera del terminale il giorno dopo. Per valutare attentamente questa limitazione è necessario anche verificare quale sia l'effettiva cultura delle persone che lavoreranno sul terminale, cultura che persino nei paesi sviluppati potrebbe, per varie ragioni, essere assai modesta, fino a non superare il livello dell'analfabetismo. In tal modo l'utente del sistema può essere incapace di leggere e comprendere istruzioni semplici come quelle che indicano come usare il telefono pubblico o come compilare un modulo. Il numero delle persone adulte che si trovano in queste condizioni, veri analfabeti di ritorno, varia da paese a paese, ma non è infrequente, nel mondo cosiddetto sviluppato, contare sino al 14% della popolazione. Inoltre nei paesi in cui si registra un'elevata proporzione di immigrati occorrerà fare i conti non con la loro intelligenza o la loro cultura di base ma con la loro comprensione della lingua locale che deve essere sufficiente per garantire una comunicazione efficace.

Ad esempio in una fabbrica in Australia, il direttore Edp ha analizzato i lavoratori dello stabilimento ed ha trovato che vi erano occupate persone di 37 diverse nazionalità e che la maggior parte di esse non sapeva leggere o scrivere in inglese. Ebbene creare un'interfaccia efficace al sistema per queste persone può costituire un grosso problema non facilmente solubile; è stato sollevato solo di recente e si è compiuta solo poca strada verso la sua soluzione. È un problema ormai comune a molti paesi e merita più attenzione di quanta non gliene sia stata dedicata fino ad ora. Purtroppo, come è successo in altre aree dell'elaborazione dei dati, si può essere sicuri che si moltiplicheranno le proposte banali e scontate.

3. Espandibilità

È stato detto che un sistema in linea di successo cresce fino a diventare ingestibile. Ciò può, o può non, essere vero. In ogni caso il sistema dovrebbe essere progettato in modo tale da poter essere esteso senza richiedere un'importante riprogettazione o una degradazione del sistema.

Si installano molti sistemi in linea per offrire un servizio laddove prima non esisteva nulla. In tal caso è difficile stimare il carico che dovrà essere sopportato dal sistema perchè non si dispone di un metro per misurare la domanda. È possibile tentare di risolvere questi problemi costruendo un sistema pilota per osservare come si comporta l'utente ed estrapolare quei dati per prevedere le dimensioni reali della domanda di quel servizio. Se il carico effettivo del sistema reale dovesse essere maggiore di quello per il quale il sistema è stato progettato, si dovrebbe in ogni caso essere in grado di ampliare il sistema stesso con rapidità e semplicità.

In un mondo così mutevole occorre tener presenti altri requisiti di espandibilità. È difficile, per la maggior parte delle aziende prevedere quale sarà il livello di attività economica dopo 2 o 3 anni e i sistemi di elaborazione dovranno essere progettati in modo da poter far fronte ad eventuali aumenti del carico di lavoro che può anche variare in modo considerevole rispetto alle previsioni iniziali. Un cambiamento di governo o l'introduzione di nuova legislazione può modificare l'ambiente in modo tale da imporre cambiamenti improvvisi al carico del sistema di elaborazione.

Finchè è possibile, si dovrebbero identificare nel progetto iniziale i punti che saranno oggetto di successive probabili espansioni in modo che il piano generale del sistema riporti i dispositivi e i servizi che dovranno essere impiegati ed aggiunti per far fronte alle nuove esigenze.

4. Modularità o progetto specifico dell'applicazione

Hardware e software di sistema possono essere disegnati in modo modulare (cioè in modo che si possano modificare o riordinare piccoli segmenti o moduli del sistema senza modificare l'intero sistema). La progettazione modulare consente di far fronte ad eventuali nuovi sviluppi del sistema o a modifiche delle interfacce con altri sistemi con flessibilità. Però sarà sempre maggiore l'efficienza di un sistema disegnato specificamente per quella particolare applicazione (a spese della flessibilità). Occorre valutare attentamente, nella scelta fra modularità e progettazione specifica, quali sono i requisiti generali o le eventuali limitazioni poste al sistema. Al limite si è deciso di progettare parte di alcuni sistemi secondo il primo criterio e parte secondo l'altro.

5. Costo

Non si deve perdere mai di vista il costo delle realizzazioni che di solito costituisce un criterio che si sovrappone a tutti gli altri nella progettazione del sistema. Impianti e linee sono dispositivi che comportano spese rilevanti: vale quindi la pena di cercare la combinazione economicamente più vantaggiosa. Talvolta si può desiderare di disporre di certe pregevoli caratteristiche o di ottenere certe prestazioni complessive del sistema, ma il loro costo può risultare proibitivo. Può anche capitare di riuscire con intuizioni intelligenti a realizzare progetti di sistemi che svolgono il lavoro richiesto ad un costo inferiore a quello preventivato, ma altre volte si sarà costretti a far quadrare, con qualche compomesso, budget e specifiche del sistema. Si dovranno analizzare le soluzioni che rispondono a ciascuno dei primi quattro criteri di progettazione alla luce dei costi in gioco.

Se si seguono questi criteri di progettazione, si compiono alcuni passi sulla via dell'effettiva progettazione del sistema. Nascono alcune domande spontanee: da dove cominciare? e come valutare le possibilità per arrivare ad un sistema equilibrato e ragionevole? Si dovranno fissare alcune variabili:

- dimensione e velocità dell'unità centrale;
- dimensione e velocità dei dischi e dei tamburi;
- linee di comunicazione;
- punto-a-punto;
- multidrop;
- multiplexer o concentratori;
- full-duplex o half-duplex;

- velocità di trasmissione;
- tecniche di controllo degli errori;
- interrogazione o mancanza di controllo (*polling o freewheeling).*

Alcuni di questi parametri interagiscono fra loro e potrà anche essere necessario trovare un compromesso per arrivare ad un sistema gestibile.

4. Criteri di misurazione delle prestazioni

Nella maggior parte dei sistemi in linea o in tempo reale occorre soddisfare due criteri che ne definiscono le prestazioni: il *tempo di risposta* e la *quantità di lavoro svolto* (throughput). Il tempo di risposta misura la velocità con cui il sistema offre le sue prestazioni e la quantità di lavoro svolto (throughput) è la misura del volume dei dati che il sistema gestirà.

In un sistema di interrogazione in linea *(enquiry),* la quantità di lavoro svolta (o throughput) è pari generalmente al numero di transazioni all'ora che il sistema può gestire durante l'ora di picco dell'attività. In un sistema di smistamento dei messaggi (*message switching)* il throughput è misurato con il numero di messaggi per ora che il sistema può gestire nell'ora di picco. In un sistema di raccolta dati *(data collection)* il «throughput» è il numero di caratteri per ora o per minuto che possono essere gestiti durante il momento in cui si verifica il carico di picco.

La velocità di esecuzione del sistema può essere misurata in modi diversi per sistemi diversi. Nel caso di sistemi per lo smistamento dei messaggi *(message switching),* si parla di *«ritardo di attraversamento di ufficio»* o *«ritardo di transito» (cross office delay o transit delay)* che è il tempo che passa tra il momento in cui l'ultimo carattere del messaggio in ingresso viene ricevuto fino al momento in cui il messaggio viene posto in coda sulla linea in uscita. In un sistema di interrogazione e risposta (*enquiry-and-response system*) la misura della velocità dell'attività è il *tempo di risposta* cioè il tempo che passa tra il momento in cui l'operatore completa l'ultima azione richiesta per inviare l'interrogazione e il momento in cui vede il primo carattere della risposta sul terminale.

Questo è schematizzato nella fig. 2 che riporta un semplice sistema di interrogazione in linea. L'operatore impiega un terminale con memoria di transito e quindi può introdurre i dati nel terminale e iniziare la comunicazione sulla linea di trasmissione premendo il tasto *transmit* (o di trasmissione). A questo punto supponendo che il terminale sia privo di controlli e senza ritardi di interrogazione *(polling),* il messaggio in input viene avviato lungo la linea. Il tempo che esso richiede per essere trasmesso dipende dalla velocità di trasmissione e dalla lunghezza del messaggio. Quando l'elaboratore riceve il messaggio, questi elabora la transazione svolgendo delle azioni. Esso probabilmente accederà all'archivio e preparerà una risposta per l'operatore. La risposta sarà quindi trasmessa al terminale che, in funzione del modo con cui è stata costruita, inizierà a visualizzare il messaggio che proviene dalla linea o aspetterà a presentare all'operatore l'intero messaggio fino a quando non lo avrà ricevuto completamente. Supponendo che il terminale visualizzi i caratteri così come gli giungono dalla linea, il primo carattere apparirà poco dopo l'inizio della trasmissione in output dell'elaboratore. Il tempo che trascorre tra il momento in cui l'elaboratore inizia a trasmettere e il momento in cui l'operatore vede il primo carattere dipenderà dai ritardi subiti sulla linea di comunicazione e dal numero di caratteri di servizio all'inizio del blocco di messaggio che appunto potrebbe iniziare con un certo numero di caratteri di sincronizzazione e di caratteri di intestazione prima del primo carattere di testo. Il tempo di risposta, in questo caso, verrà calcolato come è indicato nella fig. 2.

Il tempo di risposta e la quantità di lavoro svolto sono di solito correlate, ma la loro relazione non è lineare. La curva che mostra il variare del tempo di risposta al variare del carico del sistema ha un andamento simile a quello indicato nella fig. 3. Infatti, man mano che si aumenta il carico del sistema, aumenterà il numero di transazioni che competono per le risorse del sistema e quindi in vari punti del sistema si possono verificare delle strozzature che possono causare congestione, cioè formazione di code di transazioni che devono essere elaborate. Le curve che esprimono la lunghezza delle file di attesa hanno un andamento generale simile a quello indicato nella fig. 3. Più avanti si analizzeranno alcune tecniche che aiutano ad individuare i punti dove è probabile che si verifichino strozzature del sistema.

Di solito si specifica il tempo di risposta voluto a fronte di una certa quantità di lavoro (throughput). Immaginiamoci, per esempio che per un dato sistema si sia richiesto un tempo di risposta di 3 secondi o meno con un carico di 10.000 transazioni per ora. Supponiamo che si sia in grado di rispondere a quella richiesta operando al punto B della curva di prestazioni della fig. 3. Fintantochè il carico del sistema non supererà le 10.000 transazioni per ora, il sistema opererà molto bene, ma se il carico aumentasse anche di poco, il tempo di risposta si allungherà assai rapidamente muovendosi sulla parte ripida della curva. Se tuttavia si è riusciti a rispondere a quelle richieste operando sul punto A sulla curva, oltre a mantenere i tempi di risposta entro i limiti richiesti, si potrà affrontare un notevole incremento del carico del sistema senza subire un marcato scadimento delle prestazioni.

Questa analisi ha lo scopo di tentare di prevedere con un certo anticipo dove si collocherà il sistema sulla curva delle prestazioni. Per la maggior parte dei sistemi oggi in funzione non sono stati fatti calcoli del genere e gran parte dei direttori Edp non ha un'idea di dove si trovi sulla curva il proprio sistema informativo. Eppure nella maggior parte dei casi non è difficile eseguire quei pochi semplici calcoli in modo da riuscire a prevedere con sufficiente anticipo il comportamento del sistema al variare delle condizioni di carico.

5. Impegno del sistema

Per semplificare un po' la matematica del problema, esprimeremo in generale il carico del sistema come frazione del carico massimo che il sistema può gestire. Chiameremo

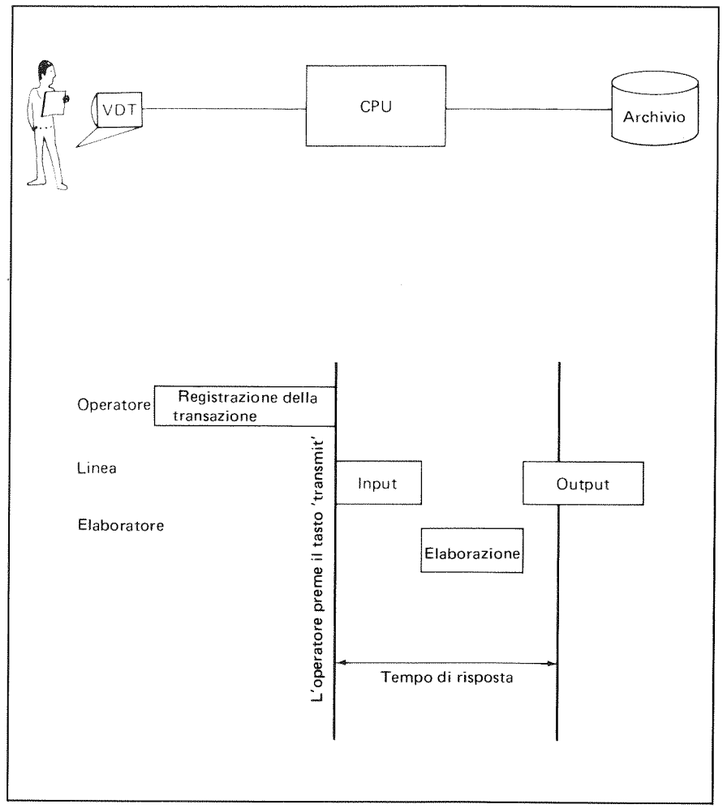


Fig. 2 - Tempo di risposta per un sistema d'interrogazione in linea.

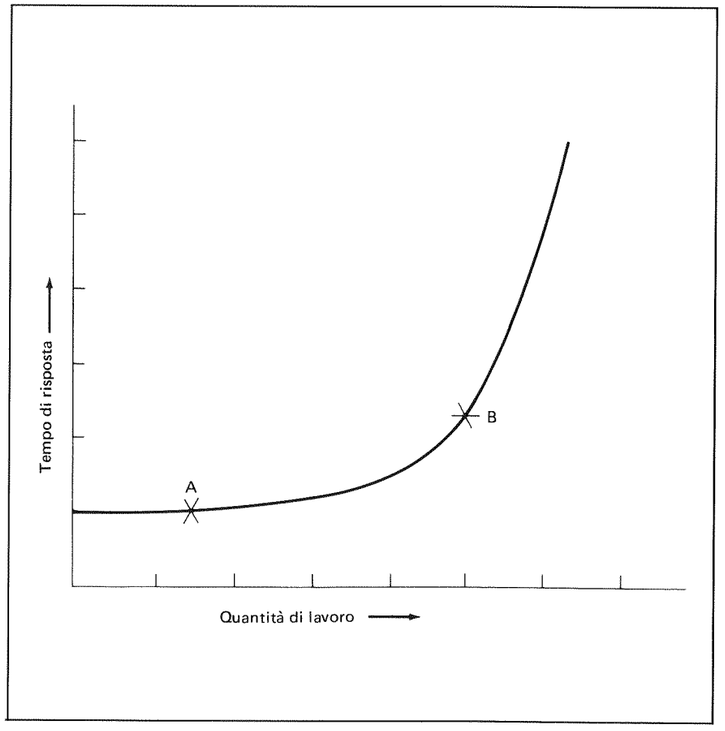


Fig. 3 - La relazione tra quantità di lavoro (throughput) e tempo di risposta.

quella frazione *impegno del sistema* ('system utilization') o anche *occupazione del sistema* ('system occupancy') come si legge frequentemente.

Si può definire l'impegno del sistema in modi diversi, i due più frequenti sono:

impegno del sistema = (carico effettivo del sistema) / (carico massimo che il sistema può gestire)

oppure

(tempo durante il quale il sistema è occupato) / (tempo disponibile)

L'impegno del sistema si indica di solito con la lettera greca ρ (ro).

Come si può dedurre immediatamente dalle equazioni, un sistema completamente occupato ha un impegno pari ad uno, mentre un sistema completamente scarico ha un impegno pari a zero. L'impegno quindi è compreso tra 0 e 1. Per esprimersi in percentuali basterà moltiplicare il valore dell'impegno per 100 ottenendo quindi la percentuale di impegno o la percentuale di occupazione.

Esiste una regola approssimativa ma di efficacia pratica che stabilisce che la curva delle prestazioni si alza assai rapidamente quando l'impegno aumenta oltre l'80%. In effetti, progettando un sistema si dovrebbe mirare a far si che il carico normale e continuo sul sistema non porti ad un impegno medio superiore al 60-70%. Questo lascia spazio alle repentine variazioni del traffico che portano l'impegno all'80% senza compromettere le prestazioni. Se si progetta il sistema mirando ad un impegno continuo e stabile dell'80%, le fluttuazioni istantanee del carico possono richiedere un impegno del 90-100% che comporta una seria degradazione delle prestazioni del sistema.

Proviamo a calcolare l'impegno di un sistema assai semplice, costituito da un negozio dove si vendono panini, anzi solo panini al prosciutto. Dopo essersi fatto una lunga esperienza, il proprietario del negozio di panini si è organizzato in modo tale che quando entra un cliente che chiede un panino al prosciutto, egli può farlo, confezionarlo e darglielo in 30 secondi. Supponiamo che arrivino 60 clienti all'ora. Calcoliamo l'impegno del cameriere (operatore) ai panini nel modo seguente:

Impegno dell'operatore = tempo occupato / tempo disponibile

Il tempo occupato nella preparazione del panino è:

(30 secondi per panino) x (60 panini l'ora) = 1800 secondi/ora.

Il numero dei secondi in un'ora è 3600, cosicchè l'impegno dell'operatore è

ρ = 1800/3600 = 0,5.

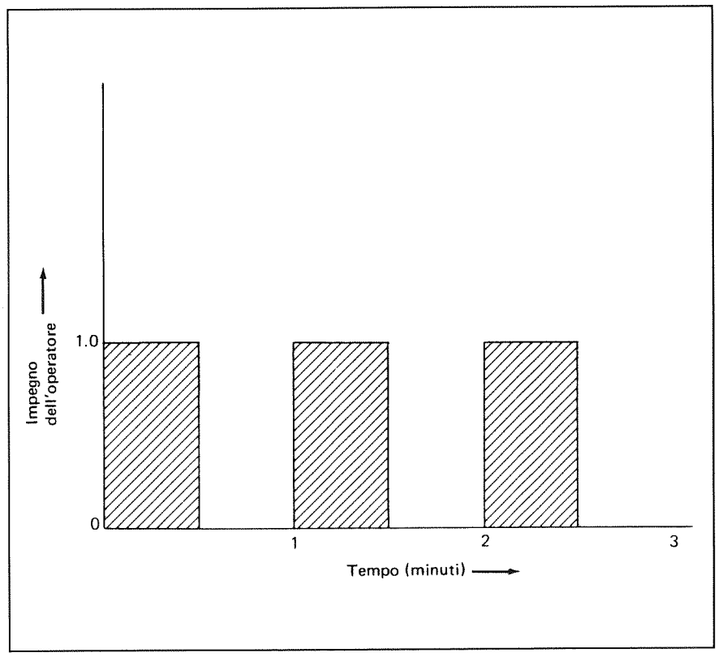
In alternativa avremmo potuto dire che alla frequenza di un panino ogni 30 secondi, la capacità massima di lavoro per quel bar con un solo cameriere è di 120 panini l'ora. Il carico effettivo del sistema è 60 panini l'ora, perciò potremmo dire che

impegno dell'operatore = carico effettivo/carico massimo

= 60/120 = 0,5.

Se i clienti arrivano regolarmente - cioé uno ogni minuto, proprio alla scadenza del sessantesimo secondo - allora si può disegnare il diagramma della fig. 4 che mostra l'impegno istantaneo dell'operatore in un certo intervallo. Poichè per fare un panino ci vogliono solo 30 secondi, quando arriva il nuovo cliente l'operatore sarà libero e ciò significa che il nuovo cliente sarà servito immediatamente dall'operatore che sarà quindi occupato per i successivi 30 secondi, al

Fig. 4 - Impegno dell'operatore con frequenza costante di arrivi dei clienti.



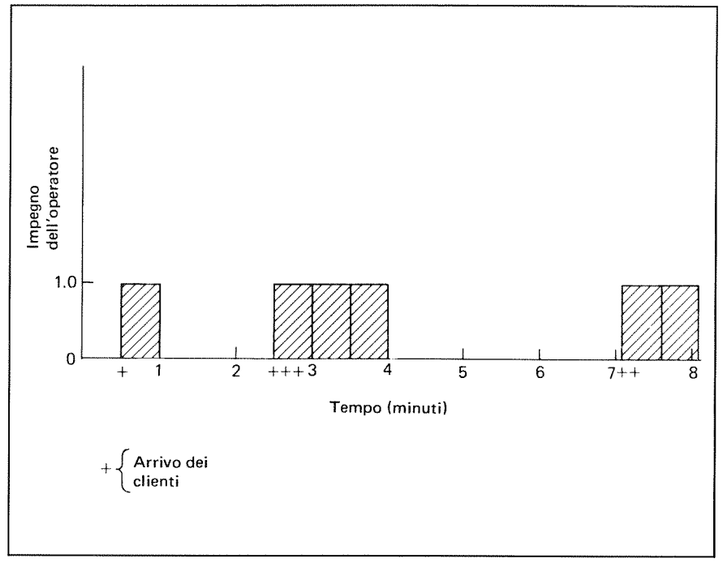


Fig. 5 - Impegno istantaneo dell'operatore quando i clienti arrivano in modo casuale.

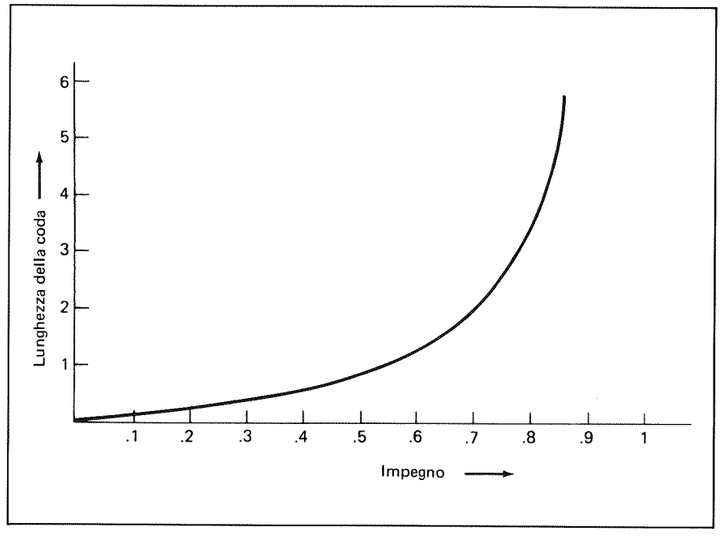


Fig. 6 - Andamento della curva di lunghezza della coda.

termine dei quali poi attenderà semplicemente l'arrivo del cliente successivo. Ci si trova in una situazione ideale, dove il tempo richiesto per fare un panino - o come spesso diciamo, il *tempo di servizio* - offerto dal sistema è costante e la transazione da servire arriva con frequenza costante.

6. Code

In realtà, le transazioni in genere arrivano in modo casuale. Se i clienti arrivano ancora con la stessa frequenza media di 60 all'ora, ma si dispongono in modo casuale, si potrebbe verificare la situazione descritta nella fig. 5. Leggendo la figura si vede l'arrivo del primo cliente in un momento di inattività dell'operatore: questo cliente sarà servito immediatamente. Dopo un periodo di inattività, arrivano in rapida successione tre clienti. Il primo di essi viene servito immediatamente, ma il secondo deve aspettare la fine del servizio al primo cliente e il terzo deve rimanere in attesa fino a quando non è stato ultimato il servizio ai due arrivati prima. Segue ancora un periodo di inattività dell'operatore e l'arrivo di due clienti: uno di questi dovrà aspettare fino a che l'altro non sia stato servito e così via.

Nella fig. 5 vediamo come si forma una coda in attesa di servizio e che la lunghezza della coda varia nel corso dell'ora. Osservando costantemente la coda si può calcolarne la lunghezza media che è funzione alquanto complessa dell'impegno medio dell'impianto. In genere l'andamento della curva che dà lunghezza della coda in funzione dell'impegno dell'impianto è quello indicato nella fig. 6.

Analogamente si potrebbe misurare il tempo che ogni persona spende nella coda, cioè il *tempo di coda* (si noti che in alcuni libri sulla teoria delle code i termini «tempo di attesa» e «tempo di coda» vengono impiegati con significato opposto) e potremmo calcolare su un certo intervallo il tempo medio che ogni persona spende in coda. Si definisce tempo di coda il tempo che trascorre dal momento in cui la transazione arriva in coda, e dopo averla percorsa tutta, fino al momento in cui esce dall'altro estremo dopo essere stata servita. Perciò il tempo di coda è costituito da un tempo di attesa che è il tempo che la transazione passa in attesa del servizio e dal tempo di servizio: pertanto si può scrivere una relazione che dice che il tempo di coda è uguale al tempo di attesa più il tempo di servizio. Ancora una volta il tempo di attesa dipende con una relazione alquanto complessa dallo impegno del servente. In generale, la curva che indica il tempo di coda in funzione dell'utilizzazione dello operatore ha un andamento assai simile alla curva della fig. 6 che dà la lunghezza della coda (il tempo minimo di coda sarà pari al tempo di servizio e la lunghezza minima della coda sarà zero).

Come si è già visto in precedenza analizzando la relazione tra quantità di lavoro (throughput) e tempo di risposta, la curva si alza assai rapidamente quando l'utilizzazione supera l'80%. Come regola generale, perché il sistema presenti caratteristiche soddisfacenti dal punto di vista della teoria delle code, è necessario che il carico continuo non superi il 60-70% circa della capacità del sistema.

Fin qui nel nostro esempio abbiamo offerto soltanto un prodotto (il panino al prosciutto) la cui confezione richiedeva un tempo di servizio costante. Se il proprietario del nostro negozio di panini deciderà di allargare la gamma dei prodotti offerti come panini di diverso tipo, caramelle, chewing-gum, bibite e così via, modificherà, così facendo, i termini del problema perchè clienti diversi richiederanno tempi di servizio diversi in funzione della complessità del loro ordine. Per esempio, il cliente che entra nel negozio per acquistare un pacchetto di gomme da masticare sarà probabilmente servito in pochi secondi, mentre altri che ordinano tre panini di tipo diverso e una bevanda saranno serviti in tempi assai piu lunghi dei 30 secondi necessari per preparare il panino al prosciutto.

Analizzando l'esempio, dopo aver introdotto tempi di servizio variabili, si può vedere che per una data frequenza di arrivo dei clienti e per un dato impegno medio dell'operatore, le prestazioni del sistema in situazione di code diventano peggiori rispetto a quelle possibili quando il tempo di servizio è costante. La fig. 7, dove abbiamo sovrapposto le curve calcolate per tempi di servizio costanti e per tempi di servizio assai variabili, da un'idea del fenomeno.

Nella maggior parte dei nostri sistemi di elaborazione, il tempo di servizio tende a fluttuare un po' invece di rimanere costante. Ancora una volta vale la regola generale che la curva si inerpica bruscamente quando l'impegno supera l'80% della capacità del sistema.

All'aumentare del carico, in più punti del sistema di comunicazione con l'elaboratore si formano code che si allungano con tempi di coda sempre maggiori. Come con il nostro negozio di panini la relazione tra lunghezza della coda o tempo di coda e impegno del servizio ha l'andamento indicato nella fig. 7. Diremo *servizi* le diverse parti del sistema che vengono impiegate per gestire una transazione. Poiché si può sovraccaricare qualsiasi servizio del sistema, è necessario calcolare l'impegno di ognuno di essi.

I servizi impiegati variano da sistema a sistema, ma vi potremo contare generalmente un operatore che può interagire con il cliente e che introduce i dati usando un terminale. L'operatore può essere sovraccarico, facendo quindi nascere una coda di persone che desiderano essere servite. L'operatore impiega un terminale collegato ad una linea di comunicazione alla quale faranno capo anche altri terminali. Ciò significa che ogni operatore compete per impegnare la linea. All'aumentare del carico di sistema si può creare una coda di transazioni in attesa di essere trasmesse sulla linea di comunicazione con l'elaboratore. Una volta che la transazione è giunta all'elaboratore, essa deve competere con tutte le altre transazioni in arrivo all'elaboratore per lo spazio di memoria e il tempo di elaborazione. Man mano che aumenta il carico, si possono creare code di transazioni che si allungano in attesa di questi servizi. E probabile che la maggior parte delle transazioni debba essere servita utilizzando un sottosistema di memoria di massa costituito da dischi o tamburi. Quelle transazioni quindi competeranno perchè venga loro assegnato lo spazio di memoria di massa previsto, ma il meccanismo di accesso richiederà tempo per posizionare le testine di lettura/scrittura sulla traccia voluta e per assegnare il canale di input/output tra l'elaboratore e il sottosistema di memoria di massa. Ancora, man mano che aumenta il carico, si possono creare code di transazioni in attesa di questi servizi. Anche gli eventuali concentratori o multiplexer presenti nella rete possono essere causa di strozzature.

È importante riuscire ad identificare su qualsiasi sistema i diversi servizi impiegati per elaborare le transazioni in modo da calcolare l'impegno di ciascuno di essi per il carico di lavoro previsto, individuando quindi per tempo il servizio o i servizi che hanno probabilità di diventare sovraccarichi per primi e determinare quindi delle strozzature. Analizziamo ora un semplice sistema in linea osservando ciò che accade quando si aumenta il carico del sistema.

7. Analisi di un sistema semplice

Analizziamo ora un sistema semplice di interrogazione (*enquiry*) in linea costituito da un elaboratore, un sottosistema a disco e una linea di comunicazione con un terminale video (VDT ovvero video-terminale) come è indicato nella fig. 8. Il sistema è stato creato per un ufficio di informazioni sul credito ed ha lo scopo di offrire a chi formula le richieste, ad esempio un negoziante, una indicazione del credito di cui gode la persona che presenta un carta di credito per acquistare qualche cosa. Il

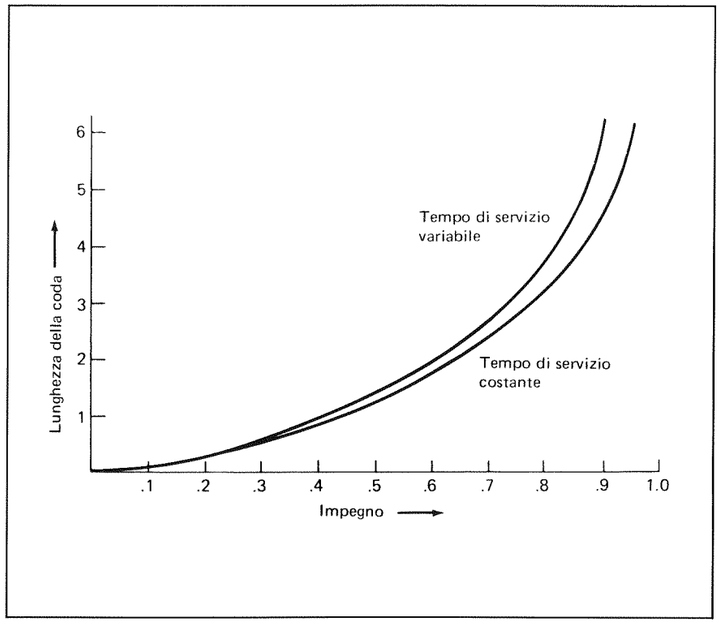


Fig. 7 - Andamento delle curve di lunghezza della coda per tempi di servizio diversi.

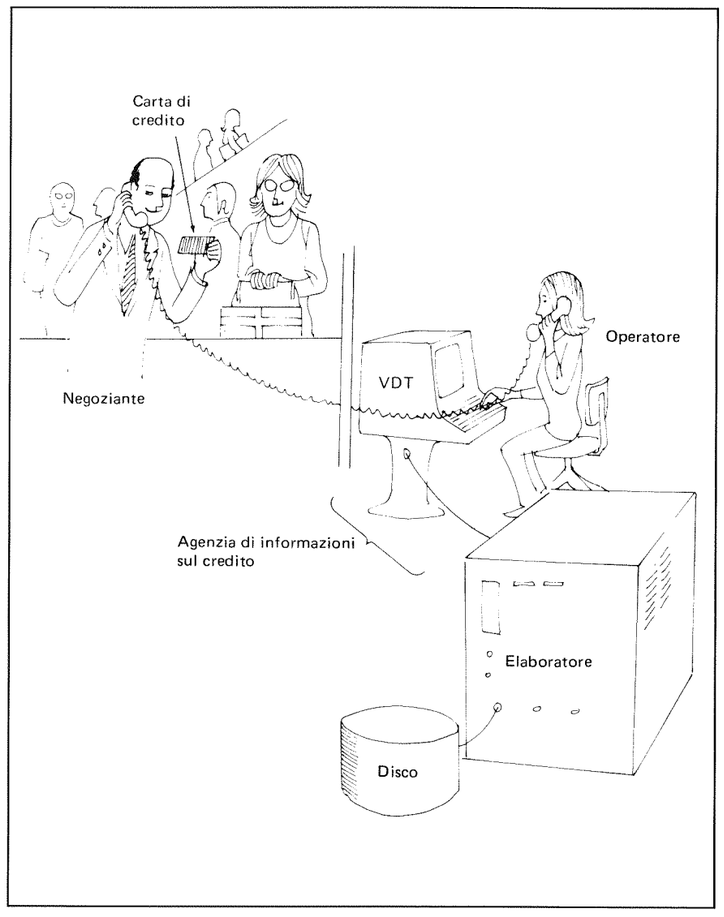


Fig. 8 - Un sistema semplice.

sistema permette di scomporre la transazione nelle seguenti operazioni elementari:

1. la persona entra nel negozio e sceglie un articolo che desidera acquistare con una carta di credito;
2. il negoziante telefona all'ufficio informazioni sul credito e parla con l'operatore al video terminale, il quale registra sul terminale il numero della carta di credito del cliente. Diremo, ai fini di questo esempio, che l'operazione richiede 10 secondi;
3. l'operatore preme il tasto di trasmissione *(transmit)* del video terminale ed invia il messaggio lungo la linea all'elaboratore. Il numero della carta di credito è sempre lungo 15 caratteri e il terminale trasmette in modo asincrono a 150 caratteri al secondo con il codice Ascii. IL video terminale è dotato di memoria di transito , cosicché il messaggio viene trasmesso in un unico blocco. La trasmissione richiede quindi complessivamente un secondo. Ai fini di questo esempio, supporremo che vi sia un solo bit di stop e che il carattere sia quindi formato da dieci bit:

   1 bit di start + 7 bit di informazione + 1 bit di parità + 1 bit di stop.

   Perciò:

   150 bit/secondo: 10 bit/carattere = 15 caratteri/secondo;
4. l'elaboratore riceve il messaggio in arrivo e lo passa ai programmi applicativi per le successive elaborazioni. Il programma applicativo calcola le chiavi di accesso all'archivio su disco magnetico a partire dal numero della carta di credito. Questa applicazione è veramente semplice ed in essa si richiede di trattare solo un tipo di interrogazione. Così il programma che elabora l'interrogazione *(enquiry)* è sempre residente nella memoria dell'elaboratore e non deve essere letto dal disco prima di iniziare l'elaborazione della transazione. Diciamo che l'elaborazione iniziale richiede 3 millisecondi:
5. ad ogni numero di carta di credito è associato un record che contiene tutte le informazioni rilevanti di quel conto. Quei record sono disposti su tutto lo spazio disponibile di disco. Poiché le interrogazioni *(enquiry)* sono introdotte in modo casuale, si deve spostare il braccio del disco per posizionare la testina di lettura sul cilindro che contiene il record desiderato eseguendo una istruzione di *seek* (ricerca) del cilindro voluto: il tempo medio di seek per questo disco è di 50 millisecondi.
6. quando il braccio è posizionato correttamente, si deve subire un ulteriore ritardo medio di 12,5 millisecondi durante i quali il disco che ruota porta il record voluto sotto la testina di lettura. Questo ritardo ha il nome di *latenza (latency)* ed è in media pari alla metà del tempo necessario per una rivoluzione del disco. Il trasferimento del record del disco all'elaboratore richiede un altro secondo;
7. i programmi applicativi elaborano i dati del record e formulano una risposta che verrà trasmessa all'operatore seduto al videoterminale. L'elaborazione del calcolatore richiede 5 millisecondi;
8. l'elaboratore trasmette un messaggio di risposta di 150 caratteri al videoterminale. Con una trasmissione asincrona a 150 caratteri al secondo in modo Ascii, sono necessari 67 millisecondi prima che il primo carattere appaia sullo schermo del videoterminale (1000 millisecondi/secondo : 15 caratteri/secondo = 67 caratteri/secondo) e 10 secondi perchè venga ricevuto l'intero messaggio (150 secondi: 15 caratteri/secondo = 10 secondi);
9. appena riceve il messaggio di risposta, l'operatore al videoterminale informa il negoziante dell'affidabilità del cliente che chiede credito. Questa conversazione dura 5 secondi al termine della quale l'operatore appende il microfono del telefono e azzera lo schermo preparandosi alla successiva interrogazione e ciò richiede altri 3 secondi.

Nella fig. 9 riportiamo l'intera sequenza di eventi su una scala temporale: il diagramma indica come si giunge al tempo totale richiesto per gestire la transazione, cioè il tempo compreso tra il momento in cui squilla il telefono presso l'agenzia di informazioni sul credito e il momento in cui l'operatore al terminale appende il microfono del telefono e azzera lo schermo. Il tempo di risposta dal punto di vista del negoziante è il tempo che intercorre tra la fine della prima conversazione con l'operatore al videoterminale (in cui indica il numero della carta di credito) e la prima sillaba della risposta dell'operatore all'interrogazione. Il tempo di risposta dal punto di vista dell'operatore è il tempo che passa tra l'inizio della trasmissione del messaggio in ingresso e l'apparizione del primo carattere della risposta sullo schermo video. Per il momento, supponiamo che tutti i tempi delle varie operazioni siano fissi e che rimangano identici per ogni transazione, anche se, come vedremo più avanti, si tratta di una semplificazione un po' eccessiva di quanto accade in realtà.

Conoscere i tempi necessari ad ogni segmento di elaborazione della transazione ci permette di determinare come viene utilizzato ogni servizio del nostro semplice sistema. Si possono identificare i seguenti servizi o serventi:

- operatore e terminale;
- linee di comunicazione;
- elaboratore;
- braccio del disco;
- canale del disco.

Si può calcolare l'impegno dell'operatore calcolando il tempo complessivo che l'operatore spende a gestire transazioni nel corso di un'ora. Un tempo che potrà essere stimato moltiplicando il numero medio di transazioni per ora (cioé la frequenza media di arrivo delle transazioni) e il tempo medio impiegato per ogni transazione. Chiamiamo il tempo impiegato dal servizio per elaborare le transazioni, tempo di occupazione del servizio (*facility holding time).*

Con riferimento alla fig. 9 stimiamo il tempo di occupazione dell'operatore per transazione che è pari alla somma del tempo richiesto da ogni singola operazione che costituisce la transazione. Il tempo di occupazio-

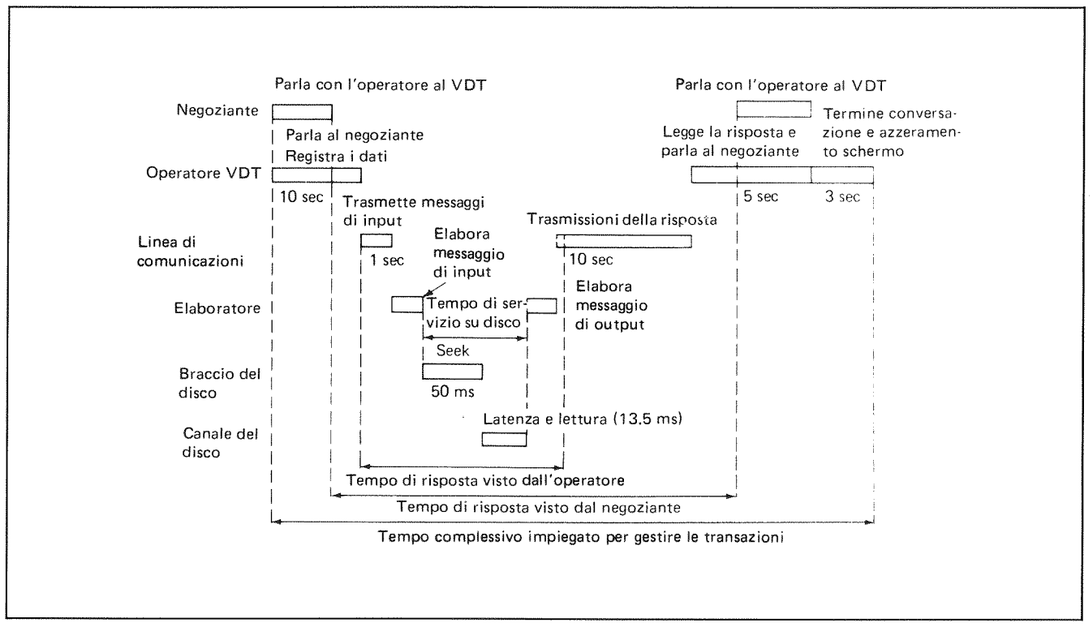


Fig. 9 - Scala temporale degli eventi del sistema di elaborazione delle trasmissioni.

ne dell'operatore è pertanto di 29,0715 secondi (10+1+0,003+0,005+0,0125+0,001+0,005+1+5+3 secondi). L'equazione che dà l'impegno dell'operatore è pertanto:

$$\rho \text{ operatore} = \frac{\text{(frequenza oraria transazione)} \times \text{(tempo di impegno operatore)}}{\text{(3600 secondi/ora)}}$$

$$= \text{(frequenza transazione)} \times \text{(29,0715 secondi)/(3600 secondi/ora)}$$

Nel nostro semplice esempio, l'impegno dell'operatore è di fatto anche l'impegno del terminale.

Si può calcolare l'impegno della linea di trasmissione in modo analogo al modo con cui è stato stimato l'impegno dell'operatore. Il tempo totale nell'ora durante il quale la linea è impegnata è uguale al prodotto della frequenza delle transazioni e del tempo di occupazione di linea. Il tempo di occupazione di linea è costituito da due elementi: il tempo di trasmissione dell'input, (cioè del messaggio che va dal videoterminale all'elaboratore) e il tempo di trasmissione dell'output (cioè del messaggio che va dall'elaboratore al videoterminale). Input e output di solito vengono riferiti all'elaboratore. Nel semplice esempio che stiamo analizzando, supporremo che la linea sia half-duplex: così il tempo di occupazione della linea per una transazione è la somma del tempo per l'input e del tempo per l'output. Se invece si trattasse di linee full-duplex, si dovrebbe calcolare il tempo di occupazione del canale *(channel holding time)* in ognuna delle direzioni. Se per esempio il nostro modello avesse una linea in full-duplex, il canale di ingresso sarebbe utilizzato assai meno del canale di uscita poichè il tempo di trasmissione per il messaggio di output è di 10 secondi, mentre il tempo di trasmissione del messaggio di input è di 1 secondo.

Le equazioni che danno l'occupazione delle linee di trasmissione sono:

$$\text{Tempo di occupazione della linea} = \text{tempo di input} + \text{tempo di output} = 1 \text{ secondo} + 10 \text{ secondi} = 11 \text{ secondi}$$

$$\rho \text{ linea} = \frac{\text{tempo di impegno/tempo disponibile}}{\text{(frequenza oraria transazioni)} \times \text{(tempo di impegno linea)}} = \frac{\text{(frequenza oraria transazioni)} \times \text{(tempo di impegno linea)}}{\text{(3600 secondi/ora)}} = \frac{\text{(frequenza oraria delle transazioni)} \times 11 \text{ secondi}}{\text{(3600 secondi/ora)}}$$

Le equazioni che danno l'impegno dell'elaboratore sono analoghe alle precedenti equazioni di impegno del servizio:

$$\textit{tempo di occupazione dell'elaboratore per transazione} = 3 \text{ msec} + 5 \text{ msec} = 8 \text{ msec}$$

$$\rho \text{ CPU} = \frac{\text{(frequenza oraria trans.)} \times \text{(tempo di occupazione CPU)}}{\text{(3600x1000 msec/ora)}} = \frac{\text{(frequenza oraria delle transazioni)} \times (8 \text{ msec})}{\text{(3600x1000 msec/ora)}}$$

Si può vedere immediatamente che l'elaboratore sarà impegnato solo limitatamente rispetto alle altre stazioni di servizio. Anche se la frequenza delle transazioni arrivasse a 1000 all'ora (limite fisicamente irrangiungibile perchè il sistema è servito da una sola persona), la CPU rimarrebbe utilizzata per meno dell'1 per cento del tempo.

$$\rho \text{ elaboratore} = (1000 \times 8) \,/\, (3600 \times 1000) = 8/3600 = 0{,}25\%$$

Analogamente l'impegno del braccio del disco può essere calcolato con le seguenti equazioni:

$$\textit{tempo di occupazione del braccio per transazione} = \text{tempo di ricerca} + \text{latenza} + \text{tempo di trasferimento} = 50 \text{ msec} + 12{,}5 \text{ msec} + 1 \text{ msec} = 63{,}5 \text{ msec.}$$

$$\rho \text{ Braccio} = \text{tempo occupato/tempo disponibile} = \frac{\text{(frequenza oraria transazioni)} \times \text{(tempo di occupazione braccio)}}{\text{(3600 x 1000 msec/ora)}} = \frac{\text{(frequenza oraria delle transazioni)} \times (63{,}5 \text{ msec})}{\text{(3600 x 10000 msec/ora)}}$$

L'impegno del canale di disco dipenderà dallo hardware e dal sistema operativo installato. Lo hardware del disco può essere costruito secondo uno dei tre criteri architettonici fondamentali. Nei sistemi più semplici, il canale di disco è dimensionato per iniziare la ricerca su disco e rimane occupato per tutta la durata dell'operazione composta da ricerca, latenza e trasferimento. Ciò impone che gli accessi a disco compiuti dal sottosistema siano eseguiti serialmente.

I sistemi più recenti sfruttano il fatto che i tempi di ricerca su disco possono essere relativamente lunghi: questi sistemi impegnano il canale per lanciare il comando di ricerca e quindi non lo rilasciano sino a quando non è stata completata l'operazione di seek. Il sistema quindi occupa il canale per il tempo di latenza e la fase di trasferimento dei dati. In tal modo è possibile sovrapporre le operazioni effettuate su due o più unità a dischi collegate ad una unità di governo. Se si iniziassero in rapida successione diverse operazioni di seek su unità diverse, la prima che finirà l'operazione occuperà il canale. In questa situazione è certamente possibile che una delle altre unità trovi il canale occupato e sia costretta ad attendere.

I sistemi più moderni sono dotati di sensore in grado di rilevare la posizione del disco in rotazione e con questo mezzo essi sono in grado, una volta che è iniziata la seek, di lasciar libero il canale fino a quando i dati non si trovino sotto la testina in lettura. Ciò consente di sovrapporre ulteriormente le operazioni su diverse unità e consente di ottenere migliori prestazioni del sistema.

Nel modello abbiamo ipotizzato unità del secondo tipo il cui canale sarà effettivamente impegnato soltanto durante il periodo di latenza e di trasferimento dei dati. Quindi le equazioni per l'impegno del canale di disco sono:

$$\textit{tempo di occupazione del canale per transazione} = \text{latenza} + \text{tempo di trasferimento} = 12{,}5 \text{ msec} + 1 \text{ msec} = 13{,}5 \text{ msec}$$

$$\rho \text{ canale} = \text{tempo occupato/tempo disponibile} = \frac{\text{(frequenza oraria transazioni)} \times \text{(tempo di occupazione canale)}}{\text{(3600 x 1000 msec/ora)}} = \frac{\text{(frequenza oraria delle transazioni)} \times (13{,}5 \text{ msec})}{\text{(3600 x 1000 msec/ora)}}$$

In tutte le operazioni di utilizzazione dei servizi che abbiamo presentato fin qui, la variabile incognita è la frequenza oraria delle transazioni. Possiamo analizzare il variare del carico sui vari servizi al variare della frequenza delle transazioni costruendo una tavola che indica l'impegno di ciascuna stazione di servizio alle diverse frequenze di arrivo delle transazioni. Ovviamente all'aumento della frequenza delle transazioni aumenterà il carico delle stazioni di servizio. Si veda la tab. 1, la quale mostra che nel nostro semplice sistema le stazioni di servizio si saturano nel seguente ordine:

- operatore;
- linea di trasmissione;
- braccio del disco;
- canale dei dischi;
- elaboratore.

Il numero teorico massimo di transazioni per ora che può essere gestito da ciascun servizio è:

$$\text{- Operatore} = \frac{3600 \text{ sec/ora}}{29{,}07 \text{ sec/transazione}} = 123{,}83 \text{ transazioni/ora}$$

- linea = 3600/11 = 327,27
- braccio del disco = 3600/0,0635 = 56.693
- canale del disco = 3600/0,0135 = 266.666
- CPU = 3600/0,008 = 450.000

La frequenza di interrogazione che l'agenzia di informazioni può gestire è limitata in questo caso dall'operatore.

La tab. 1 mostra che il carico massimo che il sistema può gestire è approssimativamente di 120 transazioni l'ora; si ha un'utilizzazione dell'80% a circa 100 transazioni l'ora, il limite massimo di sicurezza cade attorno alle 80 transazioni l'ora, poichè a questo livello l'impegno del sistema è del 65%.

Sulla base di queste informazioni potremmo disegnare la curva che esprime la relazione tra i volumi di transazioni e il tempo di coda del negoziante. Supponiamo che l'agenzia di informazioni sul credito abbia un sistema per accodare le chiamate telefoniche: in questo modo se il negoziante chiama mentre l'operatore sta parlando con qualche altra persona, il chiamante viene avvisato da un suono particolare anziché con il segnale di occupato. Quindi appena l'operatore finisce la comunicazione con il negoziante troverà il successivo che attende di essere servito. Il sistema è quindi in grado di creare una coda di negozianti al telefono che attendono il servizio man mano che il carico sul sistema aumenta. La fig. 10 mostra il tipo generale di relazioni che corre tra il tempo di coda per il negoziante e la frequenza oraria delle transazioni che stanno per essere applicate al sistema.

Se si vuol aumentare la quantità di lavoro che può essere svolta dal sistema in modo da aumentare anche il numero di transazioni per ora che possono essere elaborate, è necessario eliminare le strozzature che impongono al sistema il limite di 120 transazioni l'ora. Nel caso esaminato, il problema è costituito dall'operatore la cui utilizzazione supera il limite di sicurezza di frequenza delle transazioni che è di circa 80 all'ora.

In ogni sistema esistono due metodi principali per migliorare la quantità di lavoro svolta (throughput). Il primo mira a ridurre il tempo complessivo richiesto per elaborare ciascuna transazione in modo da poter svolgere più transazioni nel tempo disponibile. Il secondo metodo tende ad aumentare il numero di transazioni che possono essere gestite in parallelo aumentando quindi il numero totale di transazioni che si possono inviare al sistema.

Osservando la tabella di impegno del sistema si può notare che elaboratore, braccio del disco e canale del disco sono assai scarichi anche se le transazioni raggiungono una frequenza elevata. Ai fini di questa analisi si può quindi immaginare che l'elaboratore sia in grado di rispondere a qualsiasi carico gli si voglia ragionevolmente affidare. Detto questo, le principali strozzature del sistema sono l'operatore e la linea di trasmissione. Vediamo ora come si può aumentare il lavoro svolto dal sistema (throughput) riducendo il tempo complessivo necessario per elaborare una transazione.

Dalla fig. 9 si nota immediatamente che non ha molto senso ridurre il tempo che l'operatore impiega a parlare con il negoziante sia all'inizio che alla fine della transazione, poiché questi tempi sono già minimi. Ancora, si può guadagnare poco riducendo il tempo richiesto dall'elaboratore per trattare il messaggio o per migliorare l'accesso al di-

Tab. 1 - Impegno dei servizi.

Frequenza oraria delle transazioni	*Operatore*	*Linea*	*Elaboratore*	*Canale*	*Braccio*
10	0,08	0.03	–	–	–
20	0,16	0,06	–	–	–
30	0,24	0,09	–	–	–
40	0,32	0,12	–	–	–
50	0,40	0,15	–	–	–
60	0,48	0,18	–	–	–
70	0,56	0,21	–	–	–
80	0,65	0,24	–	–	–
90	0,73	0,27	–	–	–
100	0,80	0,30	0,0002	0,0017	0,00037
110	0,89	–	–	–	–
120	0,96	–	–	–	–
200	1,61	0,61	–	–	–
300	2,42	0,91	–	–	–
400	3,24	1,22	–	–	–
600	4,75	1,83	–	–	–
1000	8,00	3,05	0,002	0,017	0,0037

Nota: un servizio non può essere utilizzato oltre il 100% del tempo ($\rho = 1,00$) ma le cifre riportate mostrano come si dovrebbe affrontare il sovraccarico su operatore e linea quando i servizi CPU e dischi sono ancora assai poco utilizzati.

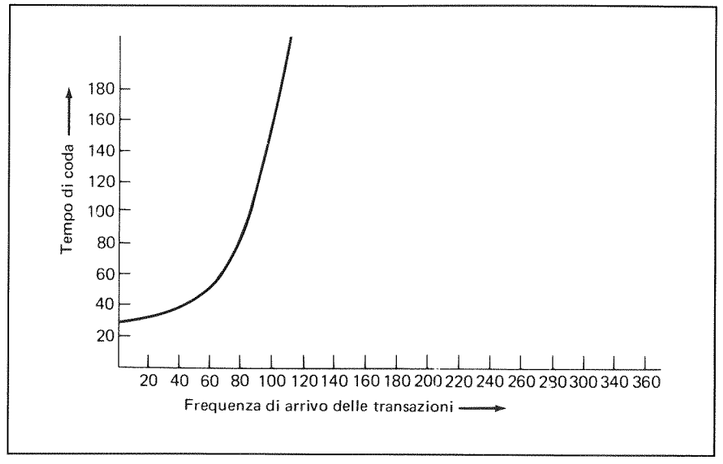


Fig. 10 - Tempo di coda per il sistema con un operatore.

sco, poiché questi tempi vengono misurati in millisecondi, mentre gli altri tempi che si possono leggere sul diagramma sono dell'ordine di secondi.

La linea di trasmissione opera a solo 150 bit al secondo ed è impegnata per ogni transazione in totale per 11 secondi: varrebbe la pena aumentare la velocità della linea a 1200 bit al secondo riducendo il tempo di occupazione della linea a 1,37 secondi.

Ciò consentirebbe quindi di ridurre a 19,44 secondi il tempo complessivo necessario per gestire una transazione portando la frequenza oraria teorica massima di transazioni a 3600/19,44 = 185,18 l'ora, e ciò rappresenta un aumento significativo rispetto alle 123,83 transazioni l'ora possibili con la configurazione precedente.

Dovendo analizzare il carico dei servizi si possono ricalcolare l'impegno dell'operatore e della linea, i due punti critici del sistema al variare del carico di lavoro del sistema, come nella tavola seguente:

Frequenza di arrivo (transazioni/ora)	*Impegno della stazione di servizio*	
	operatore	*Linea*
37	0,2	0,113
74	0,4	0,226
111	0,6	0,339
148	0,8	0,452
185	1,0	0,565

Si vede quindi che con una frequenza di 111 transazioni l'ora si raggiunge un impegno del 60%, che con 148 transazioni l'ora si tocca un impegno dell'80%, mentre 185 transazioni l'ora rappresentano il carico massimo: 100%. Possiamo ora tracciare una curva che rappresenta con una certa approssimazione l'andamento del tempo di coda del negoziante all'aumentare della frequenza delle transazioni. Nella fig. 11 sono state sovrapposte la curva già tracciata nella fig. 10 per il sistema con una velocità di linea di 150 bit al secondo e la curva per il sistema con linea a 1200 bit al secondo. Dalla fig. 11 si nota immediatamente che il nuovo sistema offre prestazioni migliori poichè non solo sostiene un carico superiore ma riduce anche il tempo di coda essendo diminuito il tempo di servizio.

Altro punto da notare è che in questo sistema non si otterrà un risultato di molto migliore aumentando la velocità di linea oltre i 1200 bit al secondo. La riduzione del tempo di coda del negoziante che si può ottenere aumentando la velocità della linea è assai modesto poiché questo ha un effetto assai piccolo sul tempo totale richiesto per elaborare ogni transazione.

Lasciamo per il momento la velocità di linea a 150 bit al secondo e verifichiamo se ci sono altre possibilità per aumentare la quantità di lavoro (throughput) svolto dal sistema. Per aumentare il numero di transazioni che il sistema è in grado di gestire, si può ovviamente aggiungere un secondo operatore, come indicato nella fig. 12. Se il sistema continua ad avere il sistema di gestione delle code delle telefonate, ciascuno dei due operatori può servire la prima transazione in testa alla coda. La tempificazione indicata nella fig. 9 è ancora valida, poiché l'aggiunta del secondo operatore non modifica l'intervallo di tempo richiesto dai vari servizi per gestire la transazione. Poichè gli operatori si dividono tra loro il carico di lavoro, l'impegno di ogni operatore sarà ora pari alla metà dell'impegno dell'operatore

Fig. 11 - Tempo di coda con velocità di linee diverse.

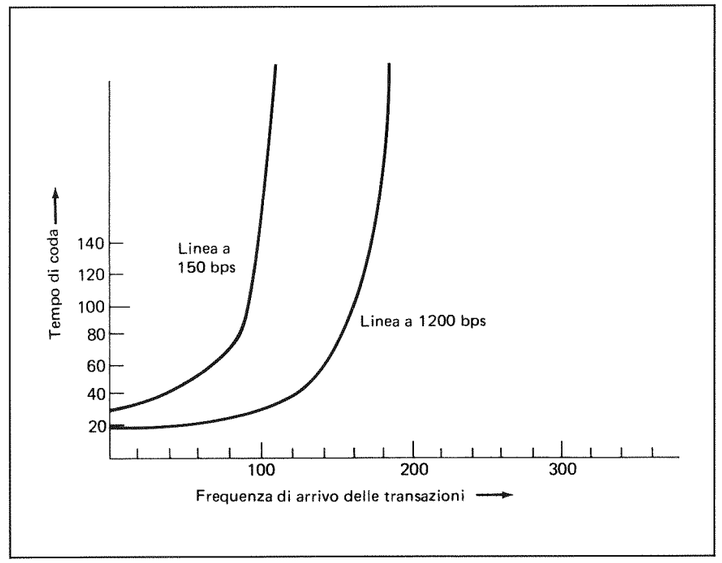
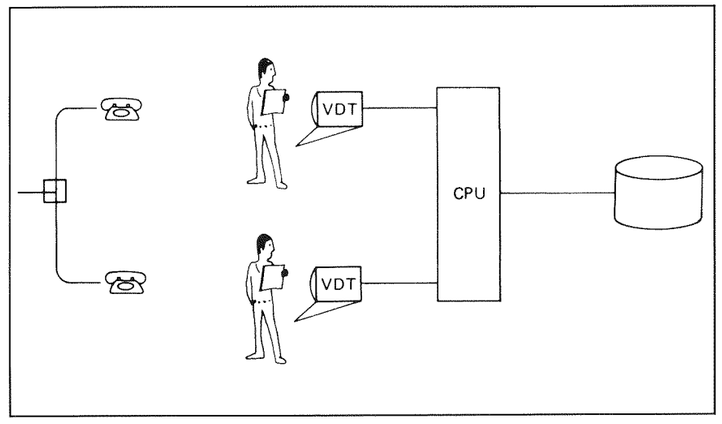


Fig. 12 - Sistema con due operatori.



dell'esempio precedente. Anche l'impegno della linea è pari alla metà dell'impegno della linea che si aveva nell'esempio precedente. Invece elaboratore e memoria di massa continuerebbero a gestire tutte le transazioni, poichè l'impegno dello elaboratore, del braccio del disco e del canale del disco rimangono sugli stessi livelli del primo esempio.

Calcoliamo ancora una volta il carico sui vari servizi all'aumentare della frequenza delle transazioni nel sistema compilando la tabella seguente:

Frequenza di arrivo (transazioni/ora)	*Impegno del servizio*	
	Operatore	*Linea*
160	0,65	0,24
200	0,80	0,30
240	0,96	0,36

In questo caso il massimo carico che il sistema può gestire è raddoppiato e i punti critici di utilizzazione del 65% e dell'80% giungono grosso modo a carico doppio rispetto a quello precedente.

La fig. 13 riporta la curva che mostra l'andamento del tempo di coda del negoziante rispetto al numero di transazioni l'ora che giungono al sistema, rispettivamente nel caso in cui il sistema abbia uno o due operatori.

Si potrebbe aggiungere ancora un terzo operatore alla configurazione indicata nella fig. 12: calcolando l'impegno dei servizi all'aumentare

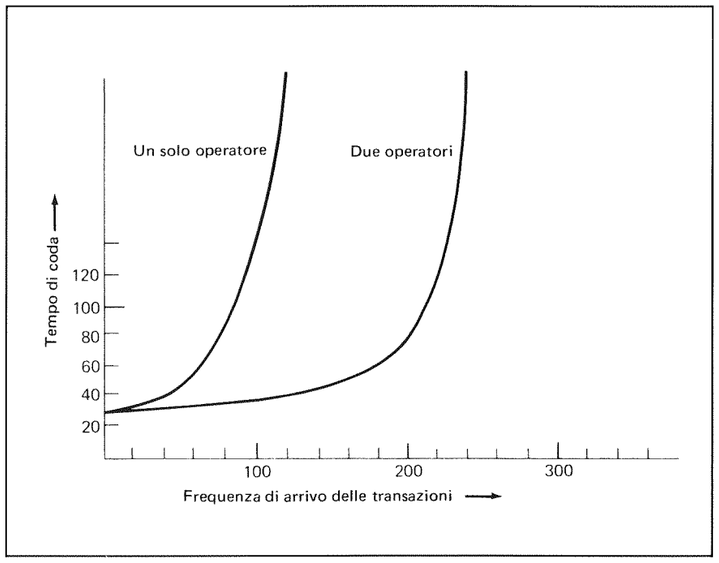


Fig. 13 - Tempo di coda di un sistema con due operatori.

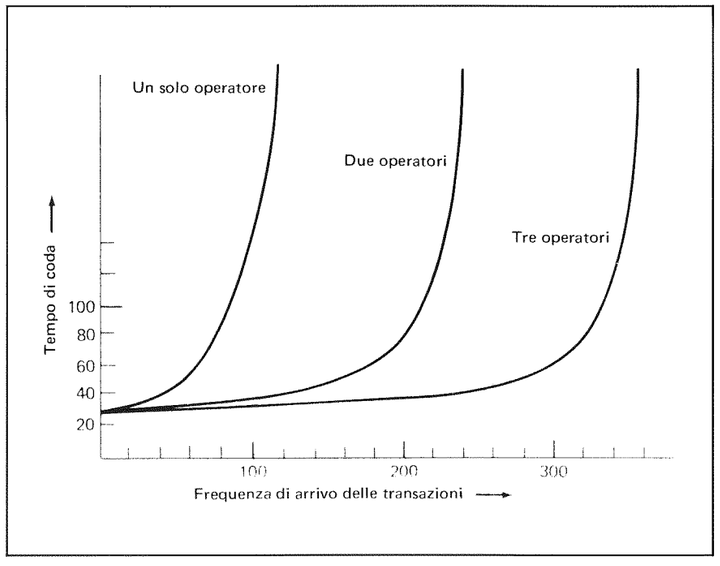


Fig. 14 - Tempo di coda di un sistema con tre operatori.

Fig. 15 - Sistema con tre operatori e una sola linea di trasmissione.

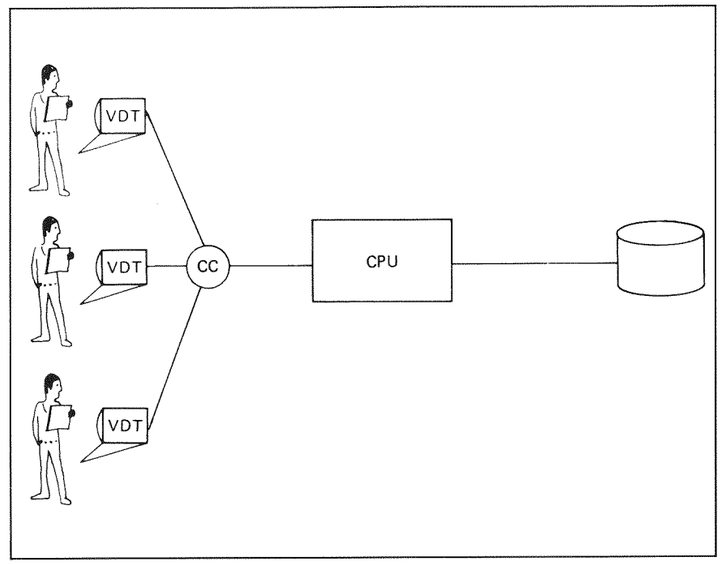
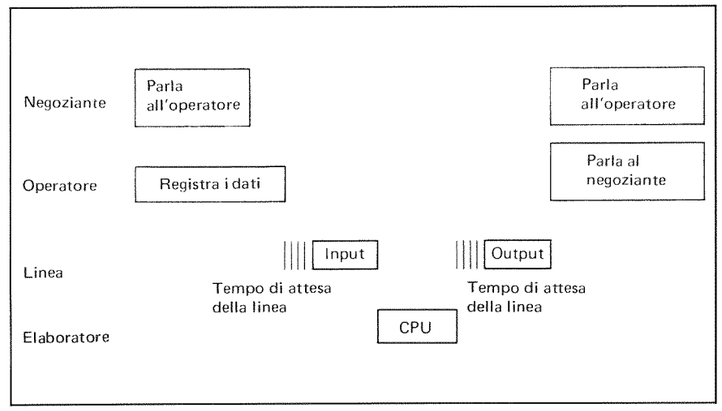


Fig. 16 - Schema temporale delle attività sul sistema con tre operatori e una sola linea.



del carico del sistema si determinano i punti critici della curva:

Frequenza di arrivo (transazioni/ora)	*Impegno del servizio*	
	Operatore	*Linea*
240	0,65	0,24
300	0,80	0,30
360	0,96	0,36

Nella fig. 14 si raffrontano le curve che danno l'andamento dei tempi di coda all'aumentare del carico per il sistema con tre operatori con le analoghe curve calcolate per i sistemi con due e con un solo operatore.

In alternativa all'espansione del sistema si potrebbero collegare altri terminali sulla stessa linea di trasmissione impiegando un governo di grappolo di terminali per evitare le eventuali contese di linea. La configurazione è schematizzata nella fig. 15. Quindi si ha un'unica linea di trasmissione che sostiene tutto il traffico proveniente dai terminali e ciò significa che l'impegno della linea aumenterà all'aumentare del traffico. Ciò farà formare delle code di messaggi in attesa del governo del grappolo di terminali, man mano che gli operatori entrano in competizione per impegnare la capacità della linea. Questo fenomeno è descritto dal diagramma a scala temporale semplificata riportato nella fig. 16. Nel riquadro indicato con «elaboratore» sono comprese l'elaborazione vera e propria, l'operazione di 'seek' e le operazioni di canale di disco riportate analiticamente nel diagramma della fig. 9.

Il tempo richiesto per elaborare ogni transazione comprenderà anche l'attesa in coda per la linea al momento dell'input e l'altra attesa per la stessa linea di trasmissione al momento dell'output. Questo tempo è effettivamente speso soltanto per superare le code per la linea anche se è distribuito fisicamente con la transazione in ingresso da una parte e le transazioni in uscita dall'altra. Questi tempi di attesa aumenteranno esponenzialmente all'aumentare del carico sulla linea di trasmissione: ciò, a sua volta, causerà un aumento assai rilevante del tempo di occupazione dell'operatore appena

il carico del sistema si avvicina al punto critico di utilizzazione della linea. Ciò farà aumentare l'impegno dell'operatore più velocemente che in altre soluzioni e questo farà aumentare il tempo di coda del negoziante in modo assai repentino. Questo è, se volete, la traduzione in concreto di un risultato dell'analisi e cioè che il tempo di attesa del negoziante è funzione esponenziale di una funzione esponenziale: all'aumentare del carico del sistema si avrà un rapido declino delle sue prestazioni.

Con tre terminali, sulla linea esiste un limite intrinseco al tempo che la transazione deve attendere in coda per la linea perchè la coda non avrà mai più di tre transazioni per volta. Se invece i terminali collegati fossero molti, il tempo di coda per la linea potrebbe essere significativo. Si può migliorare la situazione ritornando all'ipotesi originale ed aumentare la velocità di linea a 1200 bit per secondo. Questo consentirebbe di ridurre l'utilizzazione della linea, anche a frequenze di transazioni relativamente elevate, superando il precedente grosso problema di code per la linea: così facendo anche l'impegno dell'operatore non aumenterebbe così velocemente e le prestazioni del sistema risulterebbero assai più stabili.

Questa stessa analisi può essere applicata al modo con cui viene svolta l'elaborazione interna del calcolatore. Si possono immaginare code di transazioni che si allungano in attesa di entrare nei diversi segmenti elaborativi così che vengano eseguite le operazioni di seek e si liberi il canale di disco. All'aumentare del carico del sistema le code possono diventare troppo lunghe. Quando un servizio viene utilizzato per oltre l'80% questo fa degradare le prestazioni del sistema.

Per evitare di complicare l'esposizione abbiamo semplificato grossolanamente il sistema preso ad esempio. In realtà sui sistemi possono essere inviate decine o centinaia di tipi di transazioni diverse, ognuno dei quali presenta caratteristiche sue proprie di elaborazione, con messaggi di input ed ouput di varia lunghezza. Inoltre l'operatore non impegnerà lo stesso tempo parlando ad ogni persona che lo chiama. Alcune chiamate verranno elaborate velocemente, altre richiederanno un tempo assai lungo dovuto alle interazioni tra operatore e chiamate. All'interno del sistema, transazioni di tipo diverso richiederanno impegno diverso di elaboratore e di sottosistema a disco: mentre alcune transazioni potranno essere soddisfatte senza alcun accesso a disco, altre ne richiederanno venti, trenta, forse addirittura cento.

Studiando i sistemi reali si dovranno perciò applicare metodi statistici per calcolare la lunghezza media dei messaggi, il tempo medio di trasmissione, il tempo medio di elaborazione, i tempi delle operazioni su disco e così via. Non era nostro scopo analizzare con maggiore dettaglio questi problemi e pertanto rimandiamo il lettore interessato ai lavori di James Martin, *Systems Analysis for Data Transmission e Design of Real Time Computer Systems* i cui estremi sono indicati nella «Bibliografia».

8. Bibliografia

[1] Per un esame generale della trasmissione dei dati, per ritrovare alcuni dettagli tecnici sul funzionamento delle reti telefoniche e Telex; per una descrizione sintetica dei servizi di trasmissione dei dati disponibili in Gran Bretagna: U.K. Post Office, *Handbook of Data Communications,* Ncc Publications, Londra, 1975.

[2] Per un valido esame generale della trasmissione dei dati e dei sistemi in linea, con capitoli particolarmente interessanti sulla teoria delle code e sulla statistica: J. Martin, *Systems Analysis for Data Trasmission,* Prentice-Hall, Englewood Cliffs, 1972; J. Martin, *Design of Real Time Computer Systems,* Prentice-Hall, Englewood Cliffs, 1967.

[3] Per un'analisi dei dettagli tecnici delle comunicazioni con l'elaboratore, la trasmissione dei dati, il controllo del flusso di informazioni, la codifica, il multiplexing, il message switching, la commutazione di pacchetto e il routing: D.W. Davies, D.L.A. Barber, *Communications Network for Computers,* Wiley, Londra, 1973.

[4] Per il dettaglio tecnico dei servizi di telecomunicazione presenti e futuri: J. Martin, *Telecommunication and Computers,* Prentice-Hall, Englewood Cliffs, 2ª ed., 1976; J. Martin, *Future Developments in Telecommunications,* Prentice-Hall, Englewood Cliffs, 2ª ed., 1977.

[5] Per dettagli sugli standard di comunicazione e le raccomandazioni relative alla trasmissione dei dati sulle reti telefoniche (Serie V) e sulle reti dati pubbliche digitali (Serie X): Ccitt Sixth Plenary Assembly, *Orange Book VIII.1 Data Trasmission over the Telephone Network,* International Telecommunication Union, Genève, 1977; Ccitt Sixth Plenary Assembly, *Orange Book Vol. VII.2, Public Data Networks,* International Telecommunication Union, Genève, 1977.

Opportunità di investimento in "office automation": un modello di valutazione economico finanziaria

DANIELA NAN - ENRICO PARAZZINI

Direzione Amministrazione e Piani Finanziari
Honeywell Information Systems Italia
Milano

1. Introduzione

Lo scopo di questo studio è quello di determinare l'ammontare che una impresa potrebbe ragionevolmente investire in "office automation" a fronte di un previsto incremento di produttività.

Il risultato di questa analisi è significativo per due ordini di motivi: sia per stabilire l'entità dell'investimento possibile per ogni azienda gestionalmente avanzata, che per valutare le opportunità di mercato che si offriranno in un immediato futuro alle aziende che si collocano nel settore della produzione e commercializzazione di "office automation" identificando quindi la domanda potenziale di mercato.

Come per ogni altra decisione finanziaria, quella di impiegare capitali in questo tipo di investimento sarà privilegiata dalle imprese solo nel caso in cui essa garantisca un adeguato ritorno del capitale investito.

D'altra parte è affermazione analoga sostenere che il capitale che una azienda investirà deve essere direttamente proporzionale al risultato in termini di incremento di produttività.

Il problema è quindi stabilire quanto un'azienda è disposta ad investire in "office automation" in previsione di realizzare miglioramenti nel livello di produttività in un prestabilito arco temporale.

2. Struttura del modello - le variabili

Le variabili coinvolte in questa analisi possono essere ricondotte alle seguenti: [1]

1) COSTO DEL PERSONALE
2) COSTO DELL'INVESTIMENTO
3) TASSO DI ATTUALIZZAZIONE DEI FLUSSI FINANZIARI
4) PRESSIONE FISCALE
5) INCREMENTO DI PRODUTTIVITÀ ATTESO
6) LIVELLO DELL'INVESTIMENTO IN "OFFICE AUTOMATION"

1) Il costo del personale si intende comprensivo degli oneri e degli accantonamenti. Il tasso di sviluppo dei salari per gli anni successivi al primo viene assunto nella misura del 16% annuo sulla base del salario unitario dell'anno 1; naturalmente altre ipotesi possono essere ragionevolmente formulate al riguardo e si può considerare consigliabile, in tempi caratterizzati da accentuata inflazione, sottoporre a valutazione più ipotesi ritenute possibili.

2) La vita utile degli impianti in questione viene stabilita in cinque anni, con un valore di recupero ipotizzato nullo per comodità di calcolo. [2]
Il criterio di ammortamento utilizzato prevede l'imputazione annua al rendiconto economico di quote costanti.

3) Il fattore di attualizzazione da utilizzare nel modello per omogenizzare ad una unica data tutti i valori evidenziati, può essere individuato tra tassi di varia natura: è comunque possibile restringere il campo di scelta tra il costo del capitale, inteso come costo del capitale di finanziamento a breve e a medio - lungo termine al netto delle imposte, e il cosiddetto costo d'opportunità [3], ossia quel tasso di rendimento medio normale che l'azienda ritiene di poter trarre dai propri investimenti, con particolare riguardo alle occasioni abbandonate per mancanza di capitale, tasso che può essere ben identificato con il R.O.I. (Return on Investiment) dell'azienda [4].

(1) Il modello impiegato nella valutazione economico-fianziaria di un investimento in "Office Automation" risulta di possibile applicazione per decisioni di investimento di qualsiasi genere.

(2) In occasione della presentazione del modello al convegno organizzato dalla A.I.-C.A. a Milano il 18/19 novembre 1982 sul tema "Modelli d'Ufficio" venne obiettato come 5 anni rappresentassero una durata eccessiva; restiamo tuttavia dell'avviso che tale arco temporale non sia eccessivo ai fini della valutazione dell'opportunità dell'investimento, poichè non è realistico pensare che l'investimento venga eliminato prima. Altre stazioni verrano aggiunte nel frattempo, e più moderne, ma non per questo si distruggerà l'investimento fatto: lo si permuterà forse, ma si avrà in tal caso un significativo valore di recupero.

(3) Si veda al riguardo G. BRUGGER - Gli investimenti industriali - vol. 1° - Criterio di inquadramento e di valutazione degli investimenti.
Ed. Giuffré.

(4) La definizione di R.O.I. utilizzata è quella del seguente rapporto:
Utile netto + Interessi passivi (al netto delle imposte)/Investimento netto (capitale netto + finanziamenti).

Tuttavia per le aziende italiane risulta più appropriato l'impiego del costo del capitale data la situazione di elevato costo del denaro, di bassa profittabilità e di elevato indebitamento con il sistema bancario che finisce con il dare un significato del tutto distorto alla nozione di ritorno sull'investimento definita precedentemente.

4) Infine, per poter effettuare un'analisi oggettiva dei valori inseriti in bilancio, è opportuno rivedere tutti i costi interessanti questa analisi al netto delle imposte gravanti sugli stessi.

   Infatti, poichè costi del personale ed ammortamenti vengono esposti tra i componenti negativi di reddito, è corretto evidenziarli, nella valutazione economica del modello, al netto del recupero fiscale che può essere effettuato: nel caso particolare è stata utilizzata una aliquota fiscale pari al 40%.

5) Nel determinare il livello di incremento di produttività richiesto per giustificare l'investimento in "office automation", si assume che durante il primo anno si manifesti solo il 50% dell'obiettivo stabilito, mentre nei rimanenti quattro anni di vita utile dell'impianto si realizzi al 100% l'incremento di produttività ipotizzato.

6) Il livello dell'investimento in "office automation", infine, é la variabile incognita che verrà definita in funzione delle altre variabili indicate, ed espressa quantitativamente come percentuale del costo del personale.

3. Struttura del modello - le relazioni

Un investimento di qualsiasi natura è economicamente giustificato se il valore attuale dell'incremento di produttività economicamente espresso in termini di reddito[5] che potrà generare, è uguale al valore attuale del capitale investito al netto del valore attuale del recupero fiscale a seguito della procedura di ammortamento:

V.A.[6] (PRODUTTIVITÀ) = V.A. (CAPITALE INVESTITO) - V.A. (RECUPERO FISCALE)

La misura dell'incremento di produttività può essere definita come minor costo del personale.

L'obiettivo dello studio è quindi individuare la relazione lineare che lega investimento e produttività, entrambi rapportati al costo del personale. Si vuole perciò identificare una curva del tipo:

$$y = mx + q$$

dove y coincide con la possibile spesa per "office automation", x è rappresentato dalla percentuale di incremento di produttività che ne consegue, m, coefficiente angolare della retta, dal fattore di attualizzazione impiegato e q è posto pari a zero in quanto si ipotizza che per un incremento di produttività nullo anche la somma che potrà essere investita in "office automation" sarà nulla, e che quindi la curva abbia origine nel punto di intersezione degli assi.

L'equazione adattata al modello diventerà conseguentemente:

$$c_k \, CI = \Delta \, P_1$$

c_k = Costo del capitale

CI = Capitale investito

P_1 = Produttività del fattore lavoro

Sulla base di tale relazione e della curva costruita è possibile determinare quale percentuale del costo annuo del personale può essere investita in "office automation" nel presupposto di un dato incremento di produttività.

4. Applicazione del modello

L'indagine effettuata sulla base di dati rilevati da Mediobanca nel 1981 [7] su un campione di 1176 società italiane, (e su informazioni relative ad aziende di credito elaborate dagli autori) ci consente di suddividere il campione di aziende considerate in diversi segmenti:

- campione delle 1176 società
- imprese di grandi dimensioni
- medie imprese
- imprese pubbliche
- imprese private
- aziende di credito

Per ciascuna di tali imprese è stato possibile identificare una curva adeguata alle specifiche realtà di settore ed individuare quindi le potenzialità di mercato che è possibile riferire ad esse.

D) Costo netto dell'investimento

$$1.000 - 0.303 = 0.697$$

E) Relazione produttività - Investimento

Introducendo i dati così ottenuti nell'equazione del modello per un incremento di produttività pari al 10% si avrà:

$$x = \frac{0.2709}{(1-0.303)}$$

da cui

$$x = 0.39$$

passando alle percentuali di miglioramento produttivo del 15 e del 20% si otterrà conseguentemente:

$$x = \frac{0.4063}{(1-0.303)}$$

$$x = 0.58$$

e

$$x = \frac{0.5417}{(1-0.303)}$$

$$x = 0.78$$

Sinteticamente è possibile esprimere le relazioni individuate per ogni campione nella seguente tabella.

(5) Si vuole sottolineare che il concetto usato non fa riferimento a produttività fisiche, ma a produttività economiche, in cui cioè all'incremento di produttività fisica sia associato un'incremento del profitto

(6) V.A. = valore attuale

(7) FONTE: Dati cumulativi di 1176 società italiane (1968-1981) a cura di Mediobanca.

Applicazione del modello al campione di 1176 società

A) Valore attuale dell'incremento di produttività

ANNO	1	2	3	4	5
Costo del personale annuo	1.000	1.160	1.350	1.570	1.820
Tasse (40%)	.400	.464	.540	.628	.728
Costo del personale dopo le tasse	.600	.696	.810	.942	1.092
Incremento di produttività	.05	.10	.10	.10	.10
Risparmio di costi	.0300	.0696	.0810	.0942	.1092
Fattore di attualizzazione (c_k = 16%) (*)	1.0000	.8621	.7432	.6407	.5523
Valore attuale del risparmio annuo	.0300	.0600	.0602	.0604	.0603
$\sum_{i=1}^{5}$ V.A.	0.2709				

(*) La relazione di "Mediobanca" ha evidenziato che il costo medio del capitale per il campione di aziende considerate era del 27.6% che al netto delle tasse si riduce al 16%.

B) Valore attuale dell'investimento in "Office Automation"

ANNO	1	2	3	4	5
Costo	1.000				
Uscita effettiva di cassa	1.000				

C) Valore attuale del recupero fiscale dell'ammortamento

ANNO	1	2	3	4	5
Quota d'ammortamento	.200	.200	.200	.200	.200
Tasse (40%)	.080	.080	.080	.080	.080
Coefficiente di attualizzazione	1.0000	.8621	.7432	.6407	.5523
Valore attuale del recupero fiscale dell'ammortamento	.080	.0689	.0594	.0510	.0440
$\sum_{i=1}^{5}$ V.A.	0.303				

Illustrando graficamente la situazione precedentemente esposta, è possibile identificare diverse curve, ciascuna determinata sulla base delle differenti caratteristiche del settore di appartenenza delle imprese. Ogni curva correla un determinato incremento di produttività all'ammontare di spesa possibile per l'azienda esaminata (fig. 1).

5. Conclusione

Sulla base della relazione individuata per ciascuna categoria di imprese, è possibile affermare che, nella situazione particolare in cui in seguito all'impiego di apparecchi di "Office Automation" si realizzi un incremento di produttività pari al 10%, è economicamente giustificata una spesa per tale investimento pari al 39% del costo annuo del personale, e cioè una somma media per il campione delle 1176 società, di circa 12 285 000 lire.

Inoltre utilizzando la relazione espressa dal grafico è possibile verificare come per ogni diverso incremento di produttività si modifichi la percentuale di costo del personale che può essere investita: infatti, sempre mantenendo il campione delle 1176 imprese di Mediobanca, si può stabilire che se l'incremento di produttività passa dal 10 al 15 o ancora al 20%, la spesa economicamente possibile subisce un incremento proporzionale al miglioramento produttivo stesso.

Nel primo caso sarà perciò pari al 58% del costo del personale (18 270 000 circa pro-capite), mentre nel secondo al 78% (circa 24 570 000).

Relativamente agli altri campioni esaminati, per i quali sono stati impiegati valori di costo del capitale diversi, la percentuale di spesa possibile per investimenti in "Office Automation" varia, ed in particolare aumenta al diminuire del tasso stesso ma secondo un andamento che non rispecchia la proporzionalità della variazione. (9)

Può essere interessante, a questo punto, completare il discorso esaminando come varierebbero le conclusioni raggiunte qualora si considerasse un arco temporale di tre anni, anzichè di cinque come si è finora fatto. I risultati di questo ulteriore approfondimento sono sintetizzati nel grafico di fig. 2 (curva A1) dove, per semplicità, si è ragionato solo sul campione totale di 1176 aziende. Come era prevedibile, al ridursi dell'arco temporale di riferimento si abbassa l'inclinazione della curva rivelando che, in questo caso, in presenza di un incremento di produttività pari al 10% il 23% del costo del personale (7 245 000 lire circa) può essere investito, mentre quando l'incremento di produttività passa al 20% l'investimento fattibile sale al 46% del costo del personale (14 490 000 lire circa).

Concludendo, i risultati dell'analisi svolta su un campione sufficientemente esteso, e quindi ben rappresentativo, consentono di individuare ampi spazi di opportunità di investimento in "Office Automation" in presenza di realistici miglioramenti di produttività; sarà quindi su quest'area che dovrà concentrarsi l'attenzione del "management" delle imprese, essendo stata accertata, sul piano economico, la credibilità dell'investimento.

(9) Va ricordato che i valori esposti in questa analisi vengono, per semplicità di calcolo, arrotondati, conseguentemente la proporzionalità potrà non essere verificata in modo puntuale in alcuni campioni esaminati.

TABELLA RIEPILOGATIVA

Δ PRODUTTIVITÀ / SPESA PER "OFFICE AUTOMATION" (8)	10%	15%	20%
1176 SOCIETÀ	39% (12 285 000)	58% (18 270 000)	78% (24 570 000)
GRANDI IMPRESE	39% (13 104 000)	58% (19 480 000)	78% (26 208 000)
MEDIE IMPRESE	37% (11 026 000)	56% (16 688 000)	74% (22 052 000)
IMPRESE PUBBLICHE	37% (12 876 000)	56% (19 488 000)	75% (26 100 000)
IMPRESE PRIVATE	40% (11 880 000)	60% (17 820 000)	80% (23 761 000)
AZIENDE DI CREDITO	44% (13 200 000)	66% (19 800 000)	88% (26 400 000)

(8) La spesa é espressa in percentuale del costo del personale ed in valore assoluto.

Fig. 1 - Relazione produttività-investimento per diversi settori di impresa. (Ammortamento in cinque anni).

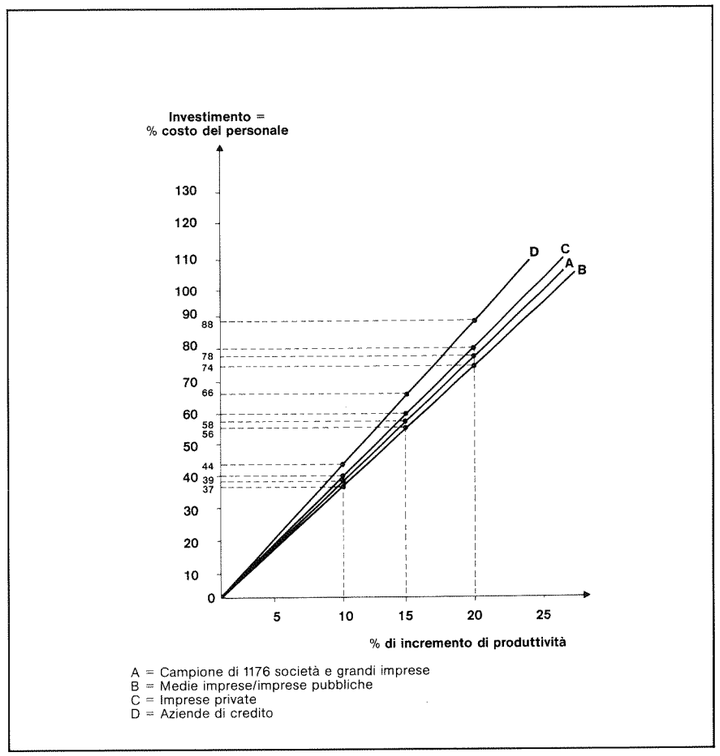
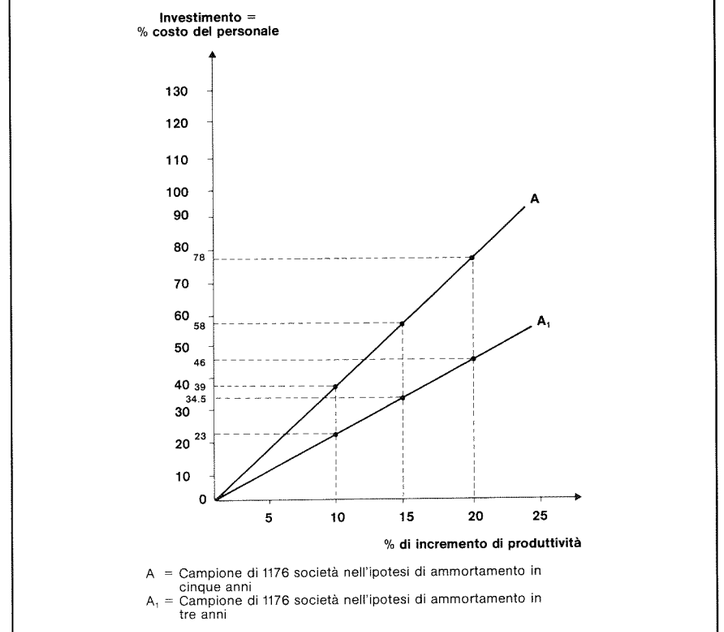


Fig. 2 - Relazione produttività-investimento: confronto tra due diverse ipotesi di periodo di ammortamento (cinque e tre anni).



Informatica e processi cognitivi

BUNI ZELLER

Consulente di Informatica Educativa
Milano

1. I sistemi di istruzione basati sull'elaboratore.

Il ruolo che l'informatica può svolgere nei processi di apprendimento è materia ormai ampiamente riconosciuta di interesse, tanto nel mondo dell'educazione che in quello dell'informatica.

I calcolatori sono oggi utilizzati nel quadro di attività di insegnamento e di *training* in aziende e grandi enti amministrativi, università e centri di ricerca, laboratori di ricerca pedagogica e istituti scolastici di vario ordine e grado. Negli ultimi decenni sono stati progettati, sperimentati e in gran parte anche costruiti e venduti su scala industriale, sistemi di istruzione basati sul ruolo del calcolatore secondo un ampio ventaglio di applicazioni:

a) Sistemi che utilizzano il calcolatore come strumento per insegnare o come sussidio didattico in ambiente di tipo tradizionale (attività di formazione guidata, sia essa di tipo scolastico o universitario, oppure in azienda). Questi sistemi vengono generalmente identificati con il nome di *Computer Assisted Instruction* (CAI) e derivano dalle ricerche e dagli investimenti intrapresi a partire dalla seconda metà degli anni '50, soprattutto nelle università americane. Nel quadro delle attività CAI sono stati impiegati sistemi del tipo: istruzione programmata, addestramento ed esercitazione, *problem solving,* simulazione, giochi educativi, *testing.*

b) Sistemi di integrazione multimediale, che utilizzano oltre al calcolatore, mezzi audiovisivi e televisivi, banche di dati, collegamenti in teleconferenza. Sistemi di questo tipo vengono usati soprattutto per attività di educazione a distanza, con il supporto in molti casi di programmi appositamente diffusi dalle reti televisive.

c) Sistemi di tipo "intelligente", basati sulle ricerche sviluppate nel campo dell'intelligenza artificiale, che gestiscono un'interazione "aperta" con lo studente sulla base di linguaggi elementari e creativi, che valorizzano l'apprendimento delle capacità logiche attraverso la costruzione e la correzione di programmi per il calcolatore. Questi sistemi sono stati sperimentati soprattutto nell'educazione infantile e nel recupero di soggetti handicappati.

Si può dunque parlare ormai di un campo assai vasto di sistemi di istruzione basati sul calcolatore, al quale contribuiscono le ricerche e i progetti di istituti di ricerca pubblici e privati, di aziende, di consorzi educativi, con il concorso di competenze e discipline diversificate che vanno dall'ingegneria elettronica, all'informatica, alla pedagogia, alla linguistica, alla psicologia, alla sociologia.

Soprattutto a partire dagli anni settanta, la diffusione di calcolatori nelle scuole di ogni ordine e grado ha assunto dimensioni di vasta scala nei paesi maggiormente industrializzati, nei quali sono state perseguite politiche nazionali di finanziamento e di sperimentazione di questi sistemi.

Negli Stati uniti il governo federale ha investito, specialmente attraverso la National Science Foundation, circa 300 milioni di dollari fino alla fine degli anni '60, a partire dalla quale tali responsabilità sono state demandate a livello statale.

In Giappone, Gran Bretagna, Germania e Francia sono state sviluppate politiche nazionali di sperimentazione a partire dai primi anni '70, soprattutto sotto l'influenza del Centro di ricerche sull'educazione (il CERI) dell'OCSE. A livello europeo la CEE, attraverso il Fondo sociale europeo, finanzia progetti in questo campo orientati soprattutto alla formazione professionale.

In Italia le attività di ricerca sono state sviluppate essenzialmente da Istituti del CNR e delle università, da associazioni e consorzi educativi.

2. L'elaboratore come strumento per programmare l'apprendimento.

Tracciato sommariamente questo quadro, occorre però sottolineare che l'interesse verso i sistemi di istruzione basati sul calcolatore sembra oggi rispondere a fenomeni più complessi e ad esigenze più profonde rispetto all'ambiente dal quale avevano preso le mosse i primi grandi progetti di "macchine per insegnare" agli albori della storia del CAI. Alla fine degli anni '50, quando il computer "mette piede" nelle scuole e nelle università americane, l'ambiente educativo è fortemente influenzato dalle correnti della scienza dell'educazione e delle teorie dell'istruzione programmata.

Il dominante senso di ottimismo attorno alle possibilità offerte dai grandi strumenti di calcolo in campo pedagogico, si associa in quegli anni ad argomenti di natura economica, tecnica e commerciale. L'impiego di tecnologie educative viene promosso come estensione di un mezzo di razionalizzazione, ottimizzazione e risparmio di risorse umane ed economiche, con l'obiettivo di migliorare l'efficacia e la produttività del sistema scolastico.

In questo contesto - dalle ricerche sviluppate da Patrick Suppes e Richard Atkinson all'Istituto di studi matematici nelle scienze sociali all'Università di Stanford, da Donald Bitzer per il sistema PLATO nei laboratori dell'Università dell'Illinois, e dai laboratori dell'IBM per il sistema CAI 1500 - nasce la prima generazione di sistemi CAI.

Le esperienze sviluppate nel corso degli anni '60 risentono fortemente delle caratteristiche tecnologiche di questa prima generazione. Le strategie basate sui concetti dell'istruzione programmata comportano rigidi controlli del dialogo programmato. La dimensione dei sistemi e la specificità dei diversi progetti li rendono ancora troppo costosi e difficilmente universalizzabili. Le risorse tecnologiche adottate sono limitate, sia dal punto di vista delle possibilità di interattività e immediatezza delle risposte, che della trasferibilità dei programmi didattici.

Il modello di apprendimento che questi sistemi seguono è dunque fortemente centralizzato e mira a fornire sequenze, più o meno lineari di informazioni, nel quadro di un processo conoscitivo sostanzialmente individuale e basato sull'accumulazione progressiva di conoscenze. Il calcolatore risponde, anche se a costo di grossi investimenti, a questa esigenza presentandosi come strumento capace di fornire informazioni soprattutto di tipo testuale, costruendo un rapporto basato sulla sequenza informazione/stimolo/risposta, e capace di misurare quantitativamente i risultati per gestire la prosecuzione del processo formativo.

3. "Lasciamo che siano i bambini a programmare l'elaboratore".

Tuttavia, già a partire dalla seconda metà degli anni '60, comincia a svilupparsi un nuovo approccio teorico e sperimentale all'impiego del calcolatore nei processi di apprendimento. All'uso del calcolatore come *electronic page turner,* cioè nel modello del CAI come mera automazione del tradizionale apprendimento cumulativo e lineare, si oppone un nuovo approccio il cui motto emblematico è "lasciamo che siano i bambini a programmare i calcolatori e non i calcolatori a programmare i bambini". Le parole sono di Seymour Papert, un cibernetico che dopo aver lavorato alcuni anni al fianco dello psicologo cognitivo svizzero Jean Piaget ha assunto, assieme a Marvin Minsky, la direzione del Laboratorio di Intelligenza Artificiale del MIT.

Dalla pedagogia piagetiana viene l'orientamento a mettere il soggetto dell'apprendimento nelle condizioni di costruire autonomamente il proprio processo cognitivo. L'elaboratore viene allora concepito come mezzo attraverso il quale esplicitare il processo conoscitivo e formalizzarlo. Gli assiomi che stanno alla base di questo uso del calcolatore si possono schematizzare nel modo che segue:

a) L'elaboratore è, storicamente, il primo strumento di automazione dei processi intellettuali.

b) Solo gli aspetti dei processi mentali completamente analizzati e compresi possono essere (attualmente) eseguiti da un elaboratore.

c) La lettura di un programma, che modellizza meccanismi particolari o conoscenze particolari, permette di dedurre i meccanismi o le conoscenze modellizzate.

d) La costruzione di un programma impone la comprensione del campo di applicazione del programma stesso.

e) Un programma è la formalizzazione di un programma e della sua soluzione. Questa formalizzazione è operativa, cioè sperimentabile, eseguibile e verificabile. Inoltre, questa formalizzazione è dinamica, cioè soggetta a modificazioni continue parallelamente allo sviluppo delle conoscenze.

f) La programmazione, in un ambiente adeguato, permette di prendere coscienza dei meccanismi del pensiero.

Se dal punto di vista della macchina il programma è un insieme di ordini comprensibili ed eseguibili, dal punto di vista di chi programma esso è l'espressione della comprensione di un determinato problema, e siccome questa comprensione può svilupparsi e modificarsi, il programma non è quasi mai un prodotto finito.

Questa è una via per realizzare una "ricostruzione delle conoscenze" in contrapposizione a quello che nella terminologia piagetiana è chiamato "apprendimento dissociato", una scissione tra ciò che si impara e il processo di apprendimento. In questa "ricostruzione" dell'unità del processo cognitivo possiamo rinvenire un fondamento epistemologico delle grandi rivoluzioni scientifiche del ventesimo secolo. In particolare, in fisica la meccanica quantistica opera una rottura con la forma meccanicistico-classica della conoscenza e postula un'integrazione tra *oggetto* e *strumenti* della conoscenza.(*)

Il programmatore è, in questo senso, il "bambino costruttore" dell'immagine piagetiana, secondo la quale i bambini sono i costruttori delle loro stesse strutture intellettuali. Il bambino non impara dall'elaborato-

(*) "In fisica classica si può stabilire una netta distinzione tra il sistema studiato e i mezzi di osservazione, e conseguentemente fare astrazione da questi ultimi nella concezione del fenomeno. L'esistenza del 'quanto di azione' rende questa distinzione impossibile, perchè essa impone un limite all'analisi dell'interazione tra il sistema e l'apparecchiatura che fissa le condizioni secondo le quali noi lo osserviamo: è dunque ora il tutto indivisibile formato dal sistema e dagli strumenti di osservazione che definisce il 'fenomeno'". (Léon Rosenfeld, cit. in Prigogine I., *La nuova allenza. Uomo e natura in una scienza unificata,* Longanesi, Milano, 1979, p. 214).

re, bensì programmando l'elaboratore, che lo mette nelle condizioni (anzi lo costringe) di esplicitare le sue "intuizioni". Tradurre un'intuizione sotto forma di programma, significa concretizzarla, renderla palpabile e più accessibile alla riflessione. Sulla base di questo ragionamento, Papert scrive in "Mindstorms", il libro che esprime la sua filosofia educativa: "...noi consideriamo le idee tratte dall'informatica non solo come strumenti che permettono di *spiegare* i meccanismi dell'apprendimento e del pensiero, ma anche come degli strumenti che permettono di *agire* su questi processi, di *modificarli,* e se è possibile di migliorare il modo in cui l'essere umano pensa e apprende".

4. Intelligenza artificiale e modelli cognitivi.

Questo approccio - come del resto è evidente dalla biografia di Papert - si basa sull'integrazione dei modelli delle scienze cognitive con le ricerche svolte nel campo dell'intelligenza artificiale, e si sviluppa in ambienti sperimentali con l'impiego di linguaggi fortemente interattivi.

Dal punto di vista storico si può dire che il BASIC sia stato il primo linguaggio a rendere possibile un apprendimento basato sulla programmazione. Tuttavia quello che il BASIC consente nei calcoli di tipo matematico, nuovi linguaggi interattivi lo consentono anche in campo non numerico. Il LOGO e lo SMALLTALK, in particolare, sviluppano capacità di manipolazione di oggetti simbolici, grafici e testuali.

In questo articolo prenderemo in considerazione soprattutto il linguaggio LOGO e ne descriveremo le procedure fondamentali.

Prima riteniamo però necessario introdurre alcuni concetti teorici e metodologici che appaiono particolarmente significativi per ricostruire l'approccio sviluppato attraverso linguaggi progettati e sperimentati nel campo dell'intelligenza artificiale e in ambienti di programmazione. Del resto si può ritenere che i contenuti dei processi cognitivi siano indipendenti dalla *sintassi* di un linguaggio particolare, ma dipendano dalla *semantica* delle istruzioni del linguaggio, e cioè dalla sua capacità espressiva. In questo senso si può cominciare col dire in termini generali che sono necessari linguaggi di alto livello in grado di consentire elevate capacità descrittive, semplici comandi operativi, estendibilità delle procedure a livelli superiori di complessità.

Il primo concetto a cui facciamo riferimento è quello di *proceduralità* nel linguaggio di programmazione.

Ogni linguaggio è dotato di un insieme di comandi primitivi.

Con un'opportuna sequenza di questi comandi si può scrivere un programma. La caratteristica di un linguaggio procedurale, come il LOGO, è quella di consentire procedure o programmi dotati di un nome e di uno o più argomenti che, una volta definite, possono essere invocate in modo non distinguibile (per colui che scrive il programma) dai comandi primitivi. In altre parole, un linguaggio come il LOGO è capace di aumentare il numero delle parole a cui sa dare significato operativo.

In questo modo, in base ad un'analisi attenta dei problemi, è possibile descrivere un'operazione complessa come composta in modo semplice attraverso operazioni di complessità appena inferiore. Questo modo di "scomporre" i problemi per livelli di complessità decrescente è, a ben vedere, l'arte fondamentale della scoperta scientifica, ovvero il più importante dei metodi euristici.

Conoscere significa infatti ridurre l'ignoto al noto mediante strutture logiche dominabili ad ogni passo della scomposizione. Questo procedimento vale sia per la conoscenza scientifica che per l'apprendimento, ad ogni livello di età e in ogni campo. Al tempo stesso la scomposizione procedurale di un'azione complessa è l'abilità fondamentale richiesta per progettare azioni rivolte al raggiungimento di scopi non immediatamente raggiungibili.

Un secondo concetto fondamentale in questa strategia cognitiva ci pare sia legato al sistema della *programmazione parallela* (o trattamento simultaneo). Questo è un elemento fondamentale dello SMALLTALK, un linguaggio sviluppato da Alan Kay presso il Centro ricerche Xerox di Palo Alto, orientato alla creazione di ambienti per la programmazione, che è stato sperimentato con finalità didattiche e si presta alla programmazione per disegni animati, giochi e composizioni musicali.

Alla base dello SMALLTALK sta una visione del processo intellettivo descrivibile come cooperazione e concorrenza di procedure, che deriva da una teoria fondamentale dell'intelligenza artificiale proposta da Minsky e rielaborata da Carl Hewitt con il nome di "teoria degli attori". Questa concezione interattiva del processo cognitivo porta a concepire un sistema di elaborazione in cui si possa facilmente interrompere un'attività, per svilupparne altre che concorrano al raggiungimento dello scopo previsto, e poi riprenderla successivamente. Se, ad esempio, l'obiettivo è quello di comporre un disegno con due tipi di simmetria fondamentale, si può creare una procedura che realizza il primo tipo di simmetria, farla partire e pensare alla seconda. Quando la seconda procedura è stata scritta, si può riprendere l'esecuzione della prima e vedere che risultati ha dato. Nel frattempo la seconda può essere mandata in esecuzione. Alla fine le due procedure dovranno essere eseguite in parallelo per formare il prodotto finale, cioè il disegno previsto.

È evidente che la programmazione parallela facilita la programmazione di sistemi complessi e la rende più chiara da concepire.

Il terzo concetto che intendiamo mettere in evidenza è il *ruolo dell'errore*. La relazione tra questo e gli altri due concetti evidenziati è rintracciabile in una delle teorie più importanti dell'intelligenza artificiale, sviluppata presso il MIT da Minsky, Papert ed altri collaboratori: la teoria dell'apprendimento come processo di costruzione e messa a punto di un sistema complesso di procedure che si richiamano. Questa teoria che dagli autori è chiamata teoria procedurale dell'acquisizione delle competenze, ovvero in termini abbreviati 'teoria del *debugging*', ha chiaramente origine nell'analisi di come un programmatore esperto

costruisce un programma complesso.

Questo processo può essere sommariamente descritto come un continuo raffinamento di un'idea centrale che si articola via via in una struttura organizzata, di tipo gerarchico, di idee secondarie. Se questa teoria è vera, dicono Minsky e Papert, gli errori di apprendimento possono essere spiegati come errori di applicazione di alcuni metodi generali che servono alla costruzione delle attività complesse.

5. Il linguaggio LOGO.

L'impiego di linguaggi interattivi in ambienti sperimentali, soprattutto con bambini, è stato realizzazo da Papert e Minsky ispirati dalla filosofia piagetiana dell'intelligenza come adattamento all'ambiente e della educazione come sviluppo spontaneo attraverso una serie quasi biologica di stadi intellettuali.

Da questa premessa discende l'idea che in questo processo occorra inserire il soggetto dell'apprendimento (tipicamente il bambino) in un ambiente fertile. Nasce così il Laboratorio di Intelligenza Artificiale del MIT, e al suo interno un laboratorio sperimentale sull'apprendimento basato sul calcolatore, attraverso un linguaggio di programmazione estremamente semplice, il LOGO.

Nel laboratorio del MIT hanno "giocato" persone di tutti i tipi. Dai bambini di quattro anni ad alcune intere classi di ragazzi provenienti da zone culturalmente svantaggiate dei dintorni di Boston; dai liceali superdotati, agli studenti di scienza dei calcolatori del MIT; dai ricercatori di alcune università dell'America latina in cui il LOGO è stato trasferito, a quelli di altri dipartimenti di Intelligenza Artificiale, come quello di Edimburgo. Con il LOGO e la sua filosofia di apprendimento si sono poi cimentati molti dei più brillanti ricercatori del Laboratorio di Intelligenza Artificiale. La traccia di questo lavoro è rappresentata da oltre duecento memorie tecniche, che affrontano problemi del tipo più disparato, dalla creazione automatica di poesie, alla simulazione di fenomeni biologici e comportamentali.

Il linguaggio LOGO si presenta come un sistema per dialogare con un calcolatore, che consente di impartire ordini scomponendo un determinato processo in una sequenza di azioni logiche.

In altri termini rappresenta un automa, cioè un esecutore fedele di ordini ben formati. Può dunque essere un modello astratto del comportamento sia di una macchina che di un uomo che, con un certo repertorio di azioni, agisce per realizzare un determinato stato di cose.

L'automa LOGO non esegue qualsiasi ordine. All'inizio riconosce un insieme di parole primitive mediante le quali si possono comporre gli ordini. Ma il LOGO è un automa capace di apprendere: il suo linguaggio può essere cioè esteso, nel senso che mediante le parole primitive l'utente può definire il significato operativo di altre parole, che entrano così a far parte degli ordini eseguibili dall'automa.

Partendo da alcune delle materie fondamentali insegnate a scuola (lingua, matematica, scienze, musica), le esperienze fatte con il LOGO hanno presto travalicato i confini disciplinari, proponendo ai bambini esperienze di tipo globale, non realizzabili altro che con il calcolatore.

Tra queste esperienze la più conosciuta, e implementata anche su altri linguaggi (ad esempio su SMALLTALK, o su LISP che è il linguaggio di base dell'intelligenza artificiale), è quella della "Tartaruga". La Tartaruga è un piccolo veicolo, un vero e proprio robot telecomandato dal calcolatore attraverso il linguaggio LOGO. L'obiettivo è quello di esplorare uno spazio per mezzo della Tartaruga che si sposta sul terreno lasciando una traccia scritta sul pavimento, in grado di avvertire la presenza di ostacoli sul suo cammino attraverso cellule fotoelettriche.

La Tartaruga è stata poi anche "trasferita" sul video di un calcolatore dove si presenta come un "oggetto", un punto o un triangolino. I suoi spostamenti consentono di tracciare qualsiasi figura, grazie a semplicissimi ordini comunicati attraverso la tastiera: AVANTI, INDIETRO, DESTRA, SINISTRA, accompagnati dai valori di distanza lineare e variazione di direzione dello spostamento in gradi. I singoli comandi sono così organizzabili in sequenze più o meno complesse, che vengono a costituire vere e proprie procedure, scoperte e costruite autonomamente, confrontando intuizioni e operazioni logiche, a forza di prove e di errori.

Questo metodo porta Papert a chiamare la sua Tartaruga un "oggetto per pensare", un "animale cibernetico assistito dal calcolatore", uno "strumento capace di programmare". Il calcolatore diventa, attraverso il linguaggio LOGO, un mezzo per la riflessione, l'analisi e la conoscenza. Struttura il pensiero e il ragionamento, fornendo una logica e una metodologia intellettuale, euristica e scientifica.

Oltre alla Tartaruga, le esperienze educative realizzate per mezzo del LOGO sono state: 1) la produzione di poesie con il calcolatore, sulla base di una ricostruzione "produttiva" della grammatica e del concetto di generazione casuale di parole di una certa classe; 2) lo studio empirico delle regolarità dei numeri e delle operazioni aritmetiche; 3) le esperienze di composizione musicale attraverso una scatola musicale controllata dal calcolatore per mezzo del linguaggio LOGO, che consente di scomporre la musica nei suoi elementi fondamentali (melodia, ritmo, armonia, colore); attraverso la variazione, eseguita mediante comandi LOGO, di ciascuno di questi elementi si arriva a capirne l'importanza e il valore effettivo.

6. L'informatica tra teoria e tecnica.

Il punto di vista cognitivo e interattivo dell'informatica, che sta dietro a linguaggi come LOGO e SMALLTALK, propone un uso del calcolatore come strumento che consente di fare dell'attività di programmazione un metodo per la formalizzazione delle conoscenze e l'acquisizione di capacità di comprensione e manipolazione di problemi e situazioni complesse. Si tratta di qualcosa che va ben oltre il tradizionale concetto di CAI e che, a ben vedere,

rivela un possibile approccio per portare l'informatica in ambienti educativi, anche laddove l'obiettivo formativo non è principalmente orientato all'acquisizione di una professionalità specifica.

Il problema si è rivelato consistente laddove si è cercato di sviluppare progetti di introduzione su vasta scala dell'informatica nelle scuole. In Francia, ad esempio, attorno alla sperimentazione programmata allo inizio degli anni '70 per l'introduzione dell'informatica nei licei, si sono manifestati due punti di vista differenti: da un lato i sostenitori dell'insegnamento disciplinare dell'informatica, dall'altro i fautori di una "disseminazione" dell'informatica nell'insegnamento delle discipline tradizionali. Il dibattito è stato animato da due personalità autorevoli, due "pionieri" dell'informatica nell'educazione.

Da una parte Jacques Arsac si richiama alle conclusioni di un convegno promosso dall'Ocse, nel 1970 a Sévres, per ribadire che "ciò che è importante non è tanto l'introduzione dell'elaboratore, quanto la via informatica che si può definire come algoritmica, operativa e organizzativa". E aggiunge: "L'informatica è soprattutto un linguaggio, un sistema di segni che permette di comunicare allo stesso livello degli altri linguaggi, come la matematica e le lingue". Perciò Arsac sostiene che l'informatica deve diventare disciplina di insegnamento nelle scuole.

Dall'altra parte Jacques Hebenstreit replica che "non c'è, non c'è mai stato un 'linguaggio informatico'. L'informatica è una scienza, la scienza del trattamento delle informazioni e non un linguaggio". In questi termini si rifà ad una "interpretazione pragmatica della 'società informatizzata'", dove "...gli elaboratori, in quanto tali, spariscono, non sono altro, nella schiacciante maggioranza dei casi, che una parte dei dispositivi più o meno complicati di cui l'utente non percepisce che l'aspetto 'servizio reso' dal dispositivo, e quest'ultimo sarà accettato solo se il modo d'impiego è sufficientemente semplice cioè adatto ai comportamenti abituali degli individui di una società". Concepire l'informatica come disciplina generale significherebbe addirittura - secondo Hebenstreit - creare un corpo di insegnanti specializzati, sui quali "l'insieme del corpo insegnante avrà una tendenza naturale a scaricarsi di tutto ciò che riguarda l'informatica e quindi a disinteressarsi del problema". Si tratta invece di far sì che "*ogni* insegnante nella *sua* disciplina, oltre alle altre impostazioni insegni il procedimento informatico proprio alla sua disciplina".

Dietro a questa *querelle* si può riconoscere un antico dibattito epistemologico. Per concludere questo accenno al dibattito, riferiamo quanto scrive Jean-Claude Simon, autore di un importante rapporto su Educazione e informatizzazione della società: "Un conflitto, lungo quanto la storia umana, esiste tra teoria e tecnica, Minerva e Vulcano, il teorico e il realizzatore, il sapiente e l'ingegnere. Ora... l'informatica è ben altra cosa che la tecnica dei calcolatori... È al tempo stesso una tecnica delle teorie e una teoria delle tecniche. Una tecnica delle teorie: essa meccanizza il discorso, permette di rendere concreti e calcolabili i modelli teorici, rende esplicite le nostre rappresentazioni. Una teoria delle tecniche: meglio del discorso umano, o dei linguaggi precedenti, essa permette di formalizzare le tecniche, di renderle operative e algoritmiche".

In questa sede ci pare di poter sostenere che i progetti cognitivi che stanno dietro a un linguaggio come LOGO contribuiscono a spostare l'attenzione sul fatto che fini e mezzi educativi cambiano in relazione ai modelli cognitivi. Ed è nostra convizione che fini e mezzi sociali cambiano in relazione ai modelli di interazione sociale.

Fintanto che l'educazione viene concepita sulla base di modelli di apprendimento cumulativi e lineari, funzionali a modelli di interazione sostanzialmente rigidi e gerarchici, il ruolo dell'informatica rimane stretto tra il "governo" dello strumento e la generica educazione dell'utente. Laddove invece si valorizzano gli elementi euristici e cognitivi dell'informatica, si può disporre di modelli interpretativi e di linguaggi operativi che diventano essenziali in una strategia educativa orientata alla partecipazione all'interno di sistemi organizzativi e sociali, i cui requisiti di autonomia, consapevolezza e coordinamento sono funzionali a comportamenti fortemente interattivi.

7. Note e citazioni bibliografiche

[1] SUPPES P. and MORNINGSTAR M., *Computer-Assisted Instruction at Standford, 1966-68,* Academic Press, New York, 1972.

[2] ATKINSON R.C. and WILSON H.A., eds., *Computer-Assisted Instruction: A Book of Readings,* Academic Press, New York, 1969.

[3] BITZER D.L. and BRAUNFELD P. and LICHTENBERGER, W., *"PLATO: An Automatic Teaching Device"*, IRE Trans. on Education, Vol. E-4, Dec. 1961, pp. 157-161.

Per un'analisi storica delle tendenze principali del CAI, cfr. gli articoli di Jack A. Chambers e Jerry W. Sprecher, *"Computer Assisted Instruction: Current Trends and Critical Issues"*, Communication of the ACM, Vol. 23, n. 6, June 1980, pp. 332-342 e di Jurg Niegervelt, *"A Pragmatic Introduction of Courseware Design"*, Computer, Sept. 1980, pp. 7-21.

[4] PAPERT S., *Mindstorms: Children, Computer and Powerful Ideas,* Basic Books, New York, 1980.

[5] PIAGET J., *L'epistemologia genetica,* Laterza, Bari, 1970, e *Psicologia e pedagogia,* Loescher, Torino, 1972.

Per un'analisi delle teorie di Piaget cfr. anche: Trisciuzzi L., Poropat M.T., Cargnello M., *Cibernetica e strutture operatorie del pensiero,* Loescher, Torino, 1979.

[6] MINSKY M., *A Framework for Representing Knowledge,* in Proceedings of "5th International Joint Conference on Artificial Intelligence", Cambridge, MIT, 20-26 agosto 1977.

[7] KAY A., "Microelectronics and Personal Computer", *Microelectronics. A Scientific American Book,* W.H. Freeman & Company, San Francisco, 1977.

[8] KAY A. and GOLDENBERG A., *Personal Dynamic Media,* Palo Alto Calif., Xerox Palo Alto Research Center, 1976.

[9] SIMON J.-C., *L'éducation et l'informatisation de la société. Rapport au président de la Republique,* Fayard-La Documentation Francaise, 1981.

In un volume annesso sono contenuti gli interventi di Jacques Arsac e Jacques Hebenstreit.

[10] LARICCIA G., ORLETTI F., BARRESE R. e ZAGO E., *Sistemi formativi multimediali per l'educazione degli adulti,* Rapporto tecnico preparato per il Censis, Roma, 1981.

Il rapporto offre una vasta rassegna informativa e bibliografica sul settore.

APPENDICE

La geometria della Tartaruga

(un esempio tratto da "Mindstorms" di Seymour Papert)

I comandi AVANTI e INDIETRO provocano uno spostamento della Tartaruga lungo la retta che passa per il suo asse.

Per modificarne l'orientamento sono necessari altri comandi:

DESTRA e SINISTRA comportano una rotazione sul posto della Tartaruga. Tutti questi comandi devono essere accompagnati da un numero, nel messaggio di entrata, che indica alla Tartaruga la lunghezza dello spostamento lineare o di quanto deve ruotare. È evidente per un adulto che il numero, nel caso della rotazione, rappresenta la misura in gradi dell'angolo di rotazione. Ma per la maggioranza dei bambini questo è materia di ricerca.

Per ottenere un quadrato si dovranno dare i seguenti comandi:

```
PER QUADRATO
AVANTI 100
DESTRA 90
AVANTI 100
DESTRA 90
AVANTI 100
DESTRA 90
AVANTI 100
DESTRA 90
FINE
```

oppure

```
PER QUADRATO
RIPETI 4
   AVANTI 100
   DESTRA 90
FINE
```

oppure

```
PER QUADRATO: N
RIPETI 4
   AVANTI: N
   DESTRA 90
FINE
```

Nello stesso modo si può programmare un triangolo equilatero:

```
PER TRIANGOLO
AVANTI 100
DESTRA 120
AVANTI 100
DESTRA 120
AVANTI 100
DESTRA 120
FINE
```

oppure

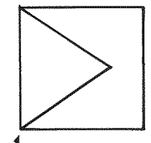
```
PER TRIANGOLO: N
RIPETI 3
   AVANTI: N
   DESTRA 120
FINE
```

Non ci sarebbe bisogno di un calcolatore per disegnare un quadrato o un triangolo. Una matita e della carta sarebbero sufficienti. Ma questi programmi, una volta elaborati, costituiscono per un bambino elementi di costruzione che gli permettono di stabilire una gerarchia di conoscenze. Si sviluppa, a partire da questo piccolo gioco, un'agilità intellettuale che si troverà ad impiegare in seguito per un'infinità di circostanze, come risulta chiaramente dai progetti intrapresi dai bambini dopo qualche seduta di lavoro con la Tartaruga.

Un esempio, tra i più elementari ma significativi, è quello del programma PER CASA. Dopo aver insegnato al calcolatore i nomi QUADRATO e TRIANGOLO, si può intuire che per disegnare una casa sarà sufficiente collocare un triangolo sopra un quadrato. Ad un primo tentativo questo sarà il programma:

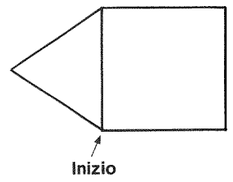
```
PER CASA
QUADRATO
TRIANGOLO
FINE
```

Ma il comando così concepito darà questo risultato:

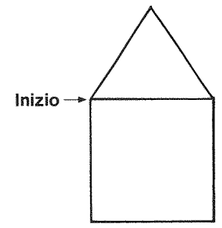


Il triangolo è stato disegnato all'interno del quadrato, invece che sopra!

Ci sono diversi modi per scoprire e correggere l'errore. Un tentativo sarà quello di ripercorrere il cammino della Tartaruga. Si scoprirà che il triangolo è stato disegnato dentro al quadrato perchè si era insegnato alla Tartaruga a disegnare il triangolo ruotando verso destra. Si potrà allora tentare di risolvere il problema programmando il triangolo con rotazioni orientate verso sinistra. Oppure si potrà inserire l'istruzione SINISTRA 60 tra QUADRATO e TRIANGOLO. In entrambi i casi si otterrà l'immagine seguente:



L'apprendista programmatore fa dei progressi. Constata che le cose non sono spesso né completamente corrette, né completamente sbagliate, e che semmai passano da un caso all'altro senza soluzione di continuità. Questa casa è più convincente della prima, ma non è ancora perfetta, bisogna raddrizzarla. Per venire a capo di quest'ultimo problema, sarà sufficiente dare il comando DESTRA 90 come primo passo del programma.



Ruolo dell'elaboratore nella diagnosi clinica

MARSDEN S. BLOIS

University of California
Section of Medical Information Science
San Francisco (USA)

1. Introduzione

L'applicazione sempre più frequente di metodi formali, inclusi algoritmi e programmi di calcolatori, a processi che sono visti di solito come richiedenti un giudizio, sembra una sorgente sia di promesse che di disagio per i medici.

L'esame di alcuni di questi metodi suggerisce che può essere di aiuto il cercare di distinguere con cura il giudizio dal calcolo. Le cure mediche riguardano un complesso di processi deduttivi, alcuni dei quali possono essere eseguiti come giudizi e altri che possono essere svolti come calcoli.

Il mio scopo è di identificare qui il secondo caso. L'evidenza empirica suggerisce che una tale demarcazione è fattibile. La domanda più importante sembra essere non *"Dove possiamo usare i calcolatori?"* ma *"Dove dobbiamo usare gli esseri umani?"*

Finchè questo argomento non sarà stato esplorato più a fondo, la tensione tra i medici e i sostenitori dei calcolatori è destinata a durare.

2. Il processo diagnostico

Ogni volta che l'argomento del giudizio medico viene discusso tra medici ed esperti di calcolatori, avviene che alcuni dei medici facciano notare che la medicina è tanto arte quanto scienza e che le regole non possono sostituire l'intuizione e l'istinto. Gli esperti di calcolatori possono replicare che a meno che la medicina sia pura magia, il giudizio clinico (con cui indico i processi decisionali che usiamo in assenza di regole esplicite o di strumenti appropriati) deve basarsi sulle deduzioni tratte dai dati del paziente e da conoscenze precedenti. Essi faranno notare correttamente che certi tipi di conoscenza possono essere rappresentati in un calcolatore, e che, sia che operi come un motore logico o come una macchina da calcolo, il calcolatore si comporta più accuratamente, e spesso più rapidamente, dell'essere umano. Il dibattito si arresta di solito a questo punto, il che è un male, perchè è proprio qui che occorre affrontare alcune questioni importanti.

Quando questi stessi dottori si ritrovano con i loro studenti di medicina, si troveranno probabilmente a descrivere le regole per fare questo o quello, insistendo che la medicina è una scienza e non un'arte. L'esperto di calcolatori, di ritorno nel suo ufficio, ricomincerà a cercare di rappresentare nel suo programma la differenza tra i significati delle parole "up" e "down" così che il programma possa "capire" la frase: "The baby threw up on the down comforter".

Argomenti seri ed interessanti sono alla base di questo dibattito. Molti medici sono imbarazzati perchè non sanno cosa attendersi esattamente dai calcolatori. E, cosa forse più importante, il sentirsi minacciati dal calcolatore può aver portato alcuni medici a sottovalutare le proprie svariate capacità.

L'analisi della nozione di giudizio clinico e l'analisi di alcuni dei contributi proposti per i calcolatori nei processi di diagnosi, riguardano in generale problemi a diversi livelli [1].

Non si possono esaminare qui tutti questi problemi. Vi sono, per esempio, le domande senza risposta: quanto il giudizio possa essere oggetto di calcolo e se il pensiero stesso sia governato da regole. Questi problemi sono stati recentemente analizzati nei termini di cosa un calcolatore non può fare [2] e di cosa il calcolatore non deve fare [3]. Altri scrittori trovano l'intero argomento semplicemente deprimente [4]. Per restringere la portata di questa discussione, consideriamo soltanto le prime fasi del processo di cura medica: quelle cioè che intercorrono tra il primo incontro paziente-medico fino alla determinazione di una diagnosi di tentativo o di lavoro.

Quando un nuovo paziente entra nello studio, l'attenzione del medico deve essere pienamente in funzione. In quel momento è richiesta al medico la massima estensione conoscitiva. Il medico generico deve essere preparato ad avere a che fare con una o più tra alcune migliaia di malattie o "condizioni" mediche e con un numero enormemente grande di sintomi che non rappresentano una malattia. I risultati di una conversazione preliminare, l'ascoltare la storia del paziente, l'esame fisico e forse qualche test di laboratorio o qualche esame speciale, verranno usati per ridurre l'enorme insieme di possibilità a un piccolo numero di stati probabili: la diagnosi differenziale.

A quale di questi atti separati, possiamo applicare il termine di "giudizio clinico"? Alla selezione della domanda successiva, alla ricerca di un segno particolare, all'ordine di un test, o alla scelta finale tra le varie alternative? Il termine può sembrare ugualmente applicabile a ciascuno di questi elementi.

Rappresentiamo il processo di giudizio come un imbuto o un corno (Fig. 1) con il diametro più largo all'inizio del processo e la parte più piccola alla conclusione. Il diametro descrescente dell'imbuto può rappresentare il restringersi della estensione di cognizioni richiesta al medico durante questo processo. All'inizio, l'ampiezza di comprensione del medico deve estendersi alla totalità del mondo di ogni giorno. Cioè, egli deve avere almeno una conoscenza media del mondo e deve essere in grado di esercitare il senso comune. La necessità di questa ampiezza di comprensione è spesso trascurata come ingrediente del giudizio medico. Tuttavia nel trattare con i programmi dei calcolatori, nulla può essere dato per scontato. E il giudizio clinico conta poco se non si basa fermamente sull'ordinario giudizio umano.

Mentre quasi tutto è possibile all'inizio di una visita al paziente, il campo di possibilità si restringe poi progressivamente. Ottenendo più informazioni, le possibilità si riducono, finchè rimane soltanto un piccolo numero di malattie potenziali.

L'universo conoscitivo, in cui ha luogo un'ulteriore ricerca dei fatti, si dà spazio alle deduzioni e nel quale va presa una decisione finale, diviene più piccolo, più dettagliato e più specifico. Cosa più importante, emerge un senso di struttura. Le alternative non solo diminuiscono, ma vengono affinate ed emergono le relazioni tra esse.

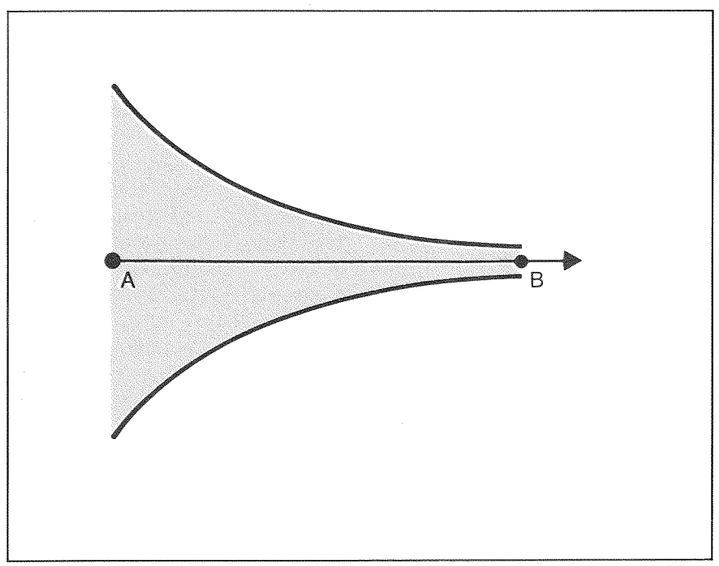
Benchè sembri appropriato applicare il termine "giudizio clinico" ai processi deduttivi che si verificano tra il punto A ed il punto B della Figura 1, questi processi appaiono di natura molto diversa [5]. Alcuni medici, se interrogati circa queste differenze, potrebbero rispondere che i processi di ragionamento richiesti attorno al punto B sono quelli che richiedono di più dalle loro capacità, che cioè in tale area del processo complessivo si richiedono maggiormente le loro abilità uniche e le loro speciali conoscenze e che qui i calcolatori sarebbero meno utili. Io dimostrerò che dovrebbe essere vero il contrario.

3. Processi diagnostici deterministici

Cominciamo con due esempi che sono indicabili come eventi medici nella vicinanza del punto B e rappresentano problemi spesso affrontati dai medici. Uno è il trattamento di un paziente con anormalità acido-basiche o elettrolitiche. Questi problemi particolari si trovano in un'area della medicina in cui i meccanismi causali sono ben compresi e in cui la gestione clinica di tale condizione è stata in parte ridotta alla soluzione di alcuni problemi di biochimica applicata. I compiti necessari sono l'esame della storia precedente, l'osservazione di alcuni segni clinici, la determinazione di speciali dati di laboratorio, la sostituzione di tali valori in formule appropriate e la soluzione di tali formule. Il paziente può anche avere altri problemi e sintomi che non sono descritti da questa formula e il medico deve occuparsi anche di quelli. Ma un medico che vuole seriamente correggere i disturbi acido-basici o elettrolitici farà bene ad usare le formule (almeno come guida generale alla diagnosi e alla cura), poichè esse sembrano rappresentare al meglio l'attuale comprensione medica del problema.

Per facilitare l'uso accurato di queste formule e regole, Bleich [6], [7] ha sviluppato dei programmi su calcolatore che richiedono e accettano certi dati di laboratorio e clinici, risolvono il problema (con l'uso delle

Fig. 1 - Ampiezza delle conoscenze richieste durante il processo diagnostico.



formule) e poi forniscono al medico consigli appropriati. I programmi di Bleich vanno oltre la pura aritmetica: essi possono richiedere dati aggiuntivi, suggerire una diagnosi differenziale, dare suggerimenti terapeutici e fornire la base per i suggerimenti. Ci si può chiedere se i programmi del calcolatore si comportano altrettanto bene del dottore. Poichè sia i programmi che il dottore impiegano la stessa formula (anche se non sempre approcciano il problema allo stesso modo) forse potremmo porre il problema in modo diverso e domandare "Sanno i medici risolvere queste formule (seguire queste regole) così bene come il calcolatore?". Poichè nè matematici nè ingegneri pretenderebbero di essere capaci di calcolare con maggior affidabilità di un calcolatore, sembrerebbe strano trovare dei medici che facessero questa affermazione.

Un altro problema clinico è quello di scegliere l'opportuna dose di medicine, particolarmente se relativamente tossiche. Modelli cinetici della distribuzione delle medicine (rappresentati da insiemi di equazioni) sono stati sviluppati e incorporati in programmi [8], i quali, quando vengono alimentati con dati d'ingresso appropriati, come il peso del paziente, la funzione renale e la quantità e temporizzazione di medicine precedenti, calcoleranno la dose necessaria per ottenere il livello desiderato nel sangue. Tale programma è stato migliorato con l'uso di tecniche addizionali, quali il feedback, nel tentativo di trattare le idiosincrasie dei pazienti [9]. Quando questi sistemi vengono valutati, le loro prestazioni appaiono di solito migliori del semplice giudizio di un medico. Questo non dovrebbe essere una sorpresa se si ammette che, a parte certe limitazioni che discuteremo più tardi, i calcolatori eseguono i calcoli estremamente bene. Programmi di questo tipo sono conosciuti come sistemi "esperti" o "consulenti". Altri programmi simili, che impiegano tecniche diverse, forniscono consigli clinici nel gestire malattie renali acute [10] e terapia antimicrobica [11].

4. Processi diagnostici stocastici

I due esempi citati sopra riguardano processi che sono deterministici nel senso che il loro comportamento può essere predetto mediante leggi conosciute, chimiche o fisiche. Se una legge rilevante è conosciuta e può essere rappresentata da una formula appropriata, il calcolo sembrerebbe fornire una predizione più accurata della stima o dell'intuizione. Ma ci sono anche processi non-deterministici (stocastici) che, possono essere predetti da formule solo in termini probabilistici, il che rende profittevoli le compagnie di assicurazione e le case da gioco.

Nel 1954, Paul Meehl [12], uno psicologo, passò in rassegna un certo numero di studi in cui le prestazioni di un clinico (di solito uno psicologo) erano confrontate con quelle di una certa procedura statistica. Questi studi riguardavano una varietà di argomenti, includendo la diagnosi psicologica, la predizione delle prestazioni di studenti all'inizio dell'università o di allievi aviatori, la predizione di recidività tra delinquenti in libertà provvisoria.

In ogni caso, certi dati di test o dati storici erano disponibili e una procedura statistica (spesso semplice come un modello di regressione lineare) veniva applicata dopo essere stata adattata empiricamente a informazioni precedenti sull'argomento particolare. Il risultato complessivo della ricerca di Meehl fu che in ognuno dei 24 casi (più tardi estesi a 35) [13], il metodo statistico, o come Meehl lo chiamava, "attuariale", si comportava altrettanto bene o meglio delle previsioni umane. I risultati dello studio di Meehl apparvero sconcertanti a molti clinici e resero perplesso lo stesso Meehl. Noi sembriamo per istinto avere un profondo rispetto per un giudizio accurato. Come può un'arida formula competere con esso?

Secondo il commento di Elstein [14], questi risultati appaiono sconcertanti, contrari alla nostra intuizione e controversi. I metodi attuariali di Meehl stabilivano degli standard di prestazioni che potevano essere avvicinate o eguagliate, ma generalmente non sorpassate dalle previsioni di gruppi di persone. Una volta ottenuta un'ottima formula di previsione, ottenere la corretta soluzione sembra il meglio che si possa fare. Nel calcolo non c'è livello di prestazione che superi il raggiungimento della correttezza.

Più recentemente sono stati incorporati in programmi per calcolatori gli algoritmi diagnostici medici basati su un certo numero di tecniche (metodi statistici come il confronto di modelli, procedure di raggruppamento, regole decisionali e regole di produzione). Questi programmi sono stati progettati per usare dati di osservazioni e per scegliere la corretta alternativa da un insieme fisso e generalmente piccolo.

Tali programmi diagnostici sono stati usati per malattie congenite del cuore [15], malattie della tiroide [16], glaucoma [23], disturbi medici [24] e dolori acuti addominali [25]. Un programma dell'ultimo tipo è stato sviluppato da De Dombal e valutato nel pronto soccorso di vari ospedali. È stato affermato che esso si comporta meglio del personale medico, quando la stessa informazione è messa a disposizione sia del programma che del medico [26]. Prestazioni di questo tipo sono state valutate con scetticismo o come una possibile minaccia da molti medici. Questi risultati sollevano ancora il problema del ruolo del giudizio clinico.

Nel valutare questo argomento, è essenziale tenere conto che queste previsioni o programmi decisionali (come quelli citati da Meehl) trattano attività collocate vicino al punto B di Figura 1. (Il più ambizioso di questi programmi, INTERNIST [24], seleziona un insieme di qualche centinaio di disturbi medici e così si posizionerebbe un po' a sinistra degli altri, se rappresentato in Figura 1, ma ancora starebbe un bel po' a destra del punto A). Tutti questi programmi operano in domini piccoli e ben strutturati che sono stati organizzati (formalizzati) in precedenza da un giudice umano operante nel punto A. Questo aspetto è quasi sempre trascurato nella discussione sulla diagnosi via calcolatore.

Domande come "Eseguiamo il programma dei dolori acuti addominali o il programma dei dolori al torace?" oppure addirittura "Dobbiamo eseguire un programma diagnostico?", devono prima essere esaminate dall'uomo. Una grande quantità di elaborazione umana delle informazioni deve aver luogo prima di affidare il lavoro al calcolatore. Quando Scriven scrive (dei risultati di Meehl), "L'ignominia più grave è sicuramente quella di scoprire che la vasta esperienza e l'addestramento formale del medico si risolvono in giudizi non migliori di quelli della formula più semplice possibile" [27], egli ripete questo errore. Trascurando di riconoscere dove comincia il processo del giudizio clinico, egli confonde una parte del processo (la scelta finale tra alternative ben definite) con l'intero processo.

5. Un problema di complessità

Ma torniamo indietro e vediamo lo stato delle informazioni al punto A. Quanti fatti sono presenti qui? Questa domanda è equivalente a chiedersi quanti fatti dovrebbero essere elencati se uno volesse dare una completa descrizione dell'universo. Ricercatori di intelligenza artificiale, psicologi della conoscenza e filosofi hanno studiato questo problema in molto maggior dettaglio di quanto possa essere riassunto qui. Tra gli ottimisti, ricercatori di intelligenza artificiale, come Minsky [28] hanno stimato che qualcosa come un milione di "fatti" sarebbero necessari per ottenere "grande intelligenza" in un sistema di intelligenza-artificiale. Panker et al. [29]. hanno stimato che il nucleo di conoscenza centrale della medicina interna implichi un milione di "fatti" e che aggiungendo le specialità mediche, questa cifra possa salire a due milioni. In precedenza Russel e Wittgenstein (nel Tractatus) avevano proposto che il mondo si potesse descrivere in termini di "affermazioni atomiche", frasi così primitive da non richiedere ulteriori spiegazioni. Questa speranza (di una atomicità logica) fu in seguito abbandonata dalla maggioranza dei filosofi, incluso Wittgenstein, e si è ora d'accordo che non c'è modo di evitare un cammino all'indietro infinito, se si cerca di descrivere il mondo in questo modo. In altre parole, ogni frase o affermazione solleva ulteriori domande che richiedono altre spiegazioni a queste altre domande e così via all'infinito. Il mondo di ogni giorno, è chiaro, non può essere descritto da un numero di affermazioni "primitive" (poichè queste non esistono) o da ogni numero finito di affermazioni di qualsiasi tipo [30].

Il numero di cose che può dirsi con esattezza di ogni oggetto del mondo è grande al di là di ogni immaginazione, e l'unico modo di parlare utilmente delle cose è di limitarsi ad argomenti rilevanti. Con "rilevanti" indico qualcosa di connesso al soggetto o allo scopo con cui siamo impegnati in un certo momento. È soltanto attraverso la scelta di attributi rilevanti che riusciamo a comunicare veramente. Al punto A di Figura 1 quasi tutti i fatti che si possono affermare di un paziente sarebbero irrilevanti per il problema di trovare cos'è che non funziona. È compito del medico scegliere dal numero incredibilmente vasto di fatti indifferenti e neutrali quelli che risultano essere rilevanti. Questa situazione è del tutto diversa da quella al punto B, dove il processo di selezione è in gran parte terminato e quindi può bastare una conoscenza specializzata, e un calcolo può risolvere il problema. Una volta raggiunto il punto B nel processo di giudizio clinico, i fatti irrilevanti sono stati filtrati e il problema medico è divenuto relativamente ben strutturato. Ma inizialmente l'imbuto delle conoscenze necessarie è molto aperto e interfaccia con il mondo in tutta la sua complessità.

Come gestisce il medico questa complessità e quale successo hanno i programmi dei calcolatori in questo compito? Molti abili medici hanno insegnato che indagare la storia del paziente è il passo più critico dell'intero processo diagnostico e che questa fase è quella che separa il medico eccezionale dagli altri. Dal punto di vista della teoria dell'informazione, questa idea appare del tutto plausibile. Consideriamo il modo in cui possiamo misurare la differenza tra il valore di un fatto esposto volontariamente dal paziente e un fatto che è ottenuto in risposta ad una domanda. In una compilazione standard di malattie [31], sono listate 3262 malattie e "condizioni mediche", ognuna definita in termini di attributi clinici e di laboratorio. Nel nostro studio di questo elenco, il termine "sangue da naso" (o i suoi sinonimi) è stato trovato nella descrizione di 27 diverse malattie.

Se un nuovo paziente visitato riferisse di "sangue da naso", la nostra attenzione sarebbe concentrata su queste 27 malattie e non sulle 3235 per cui il "sangue da naso" non è considerato una caratteristica. Se d'altra parte chiedessimo sempre a tutti i pazienti se hanno "sangue da naso" la maggioranza delle risposte sarebbe negativa; questo approccio permetterebbe di escludere soltanto 27 malattie invece di 3235. La potenza "diagnostica" o di "selezione" di una risposta positiva è pertanto più di cento volte maggiore di una risposta negativa per questo attributo. E quando i pazienti forniscono volontariamente dei fatti lo fanno quasi sempre in modo affermativo. I pazienti non riferiscono dei sintomi che non hanno. La quantità di informazione ottenuta da domande casuali sui sintomi (che forniscono per lo più risposte negative [32]) sarà quindi molto piccola. Se, al contrario, il paziente è incoraggiato a fornire spontaneamente dei sintomi positivi, il valore dell'informazione è certamente più grande.

Ogni medico conosce questo approccio e lo pratica istintivamente anche se generalmente non sa che questa pratica poggia su una salda base teorica. Cominciando a questo modo (cioé dalla "lamentela principale"), il medico dotato sia di senso comune che di conoscenze mediche ha modo di sfruttare i fatti rilevanti. Questo è il metodo del medico per gestire situazioni estremamente complesse. Come si comporterebbero qui dei programmi di calcolatore?

Forse i soli sistemi di calcolatori appropriati in questo caso sono quelli destinati a raccogliere la storia medica del paziente [33]. Quando furono introdotti alcuni anni fa, questi programmi sembravano offrire grandi promesse, ma in seguito il loro interesse sembra essere diminuito.

Friedman e Guastafson [34] commentano in proposito:

"Un esempio della mancanza di relazioni sulle ragioni dell'insuccesso di un approcio, può trovarsi nell'area dell'applicazione dei calcolatori alla raccolta di Dati Storici in Medicina. Numerosi gruppi di varie nazioni hanno lavorato e pubblicato in quest'area, spesso con duplicazione di sforzi precedenti. Tuttavia, benchè nella grande maggioranza questi tentativi siano stati abbandonati, non siamo riusciti a trovare alcuna pubblicazione sulle ragioni del loro fallimento".

Eppure la causa di questi insuccessi non sembra difficile da localizzare. Al fine di porre al paziente domande significative, il medico usa il suo senso della situazione e conoscenze mediche generali. Ma per essere significativo, un fatto deve essere significativo per qualcosa e questo qualcosa è ordinariamente una diagnosi. Non possiamo raccogliere dati significativi senza avere in mente una teoria o una diagnosi [5], anche se c'è chi pensa che la raccolta dei dati e la diagnosi possano aver luogo come attività distinte. Gorry [35] ha posto in evidenza i limiti dei programmi basati sull'analisi delle decisioni, ed alluso al ruolo delle conoscenze di senso comune. Dopo di lui, altri hanno proposto di usare le tecniche dell'intelligenza artificiale per superare questa limitazione [36].

Per le parti più importanti della medicina, gli attuali programmi su calcolatore, benchè seguano un cammino sistematico, possono al più porre delle domande meccaniche e, per la maggior parte, ottenere risposte negative, di poco valore [32]. Di più, anche quando con questi sistemi si ottengono risposte positive, i medici si lamentano di aver bisogno di ulteriori interpretazioni e che le informazioni così ottenute spesso risultano di poca importanza [32], [38]. In effetti, ci si deve aspettare questo tipo di prestazioni, dal momento che si è ignorato l'argomento della significatività o rilevanza. Vari studi [37], [39] hanno confermato che questi sistemi automatizzati per raccogliere la storia del paziente ottengono di solito informazioni "valide". Ma il nocciolo del problema sta nella mancanza di rilevanza, non di validità, e la determinazione di ciò che è rilevante in una situazione richiede senso comune e conoscenze mediche generali.

6. Intelligenza artificiale e diagnosi clinica

I problemi di ogni giorno nell'area attorno al punto A sembrano quelli in cui i programmi dei calcolatori sono meno utili. In verità c'è da dubitare che essi abbiano mai agito con successo in questo punto. Per dire le cose diversamente, perchè un programma agisca con successo in questa regione, sarebbe necessario che il programmatore avesse sviluppato i mezzi per rappresentare l'universo delle possibilità relative a questo punto. E nel punto A questo richiederebbe una descrizione del mondo. I ricercatori di intelligenza artificiale che lavorano con programmi che cercano di "capire" il linguaggio naturale [40] o di fornire consigli agli esperti [11], [41] hanno inventato sistemi ingegnosi e apparentemente impressionanti. Tuttavia questi sistemi sembrano funzionare in modo soddisfacente soltanto quando operano nei cosiddetti micro-mondi, cioè in domini estremamente ristretti che sono vicini al punto B e che sono stati quindi già accuratamente strutturati e circoscritti. Alcuni di questi sistemi del punto B si sono rivelati molto utili [41], e altri [11], [24], hanno dato buone speranze. (Le differenze tra le applicazioni relative al punto A e quelle relative al punto B possono servire a stabilire una demarcazione tra i due tipi di ricerca sull'intelligenza artificiale. Le altre applicazioni che empiricamente hanno avuto il maggiore successo vengono sempre più descritte con il termine di Feigenbaum, "ingegneria della conoscenza" [42]). L'idea che il mondo reale al punto A possa essere ricostruito mediante un grande insieme di micromondi non ha ricevuto alcun supporto e appare in serio dubbio per ragioni epistemologiche [2], [43]. Prima che diventi accettabile questo tentativo di porre insieme dei "micromondi" per formare un mondo, sarà necessaria una procedura per suddividere il mondo in più parti. Questa impresa deve ancora essere compiuta, e il suo successo appare dubbio [44].

Sembra che ci siano due approcci principali aperti per lo sviluppo di programmi che si comportino intelligentemente. Per comportarsi "intelligentemente" voglio dire operare con successo più vicino al punto A che al punto B e trattare una varietà di situazioni della vita di tutti i giorni. Il primo di questi approcci richiede un mezzo per rappresentare la conoscenza del senso comune in un programma, così da poterlo applicare a situazioni di partenza. Finora, i progressi per raggiungere questo obiettivo sono stati scarsi e il problema può, come è stato suggerito, ritenersi insolubile [2], [43]. Un'alternativa alla memorizzazione di queste conoscenze a priori, sarebbe quella di escogitare un programma capace di "apprendere", nel vero senso della parola. Ma il tipo di apprendimento include la generalizzazione di ciò che è stato appreso. Questo non sembra sia stato ottenuto, dopo circa 30 anni di sforzi nel campo della intelligenza artificiale.

Può darsi che esista un terzo approccio che rimetterebbe in discussione l'intero argomento. Questo sarebbe la scoperta di tecnologie ancora sconosciute, ossia hardware, organizzazione e software così radicalmente diversi da alcunchè oggi conosciuto che una macchina così costruita possa chiamarsi in un certo senso "viva". Vivendo nel mondo (sopravvivere, competere, riuscire, fallire e così via) una simile macchina potrebbe cominciare a raccogliere conoscenza da sola. Se lo sviluppo di una tale macchina sia possibile e come la chiameremo se venisse inventata, non può essere discusso in questa sede.

L'elemento essenziale è distinguere tra la natura della situazione che esiste al punto A, dove c'è da affrontare la totalità del mondo, e la situazione al punto B, dove il dominio in esame è stato strutturato attraverso un precedente sforzo umano, dove un'astrazione è già disponibile e dove è richiesto poco senso comune. La mia ragione per sottolineare questa differenza è che il medico, che dispone di conoscenze mediche per formazione ed esperienza, e di senso comune per essere vissuto nel mondo, può facilmente gestire la prima situazione. I medici ovviamente forniscono buone prestazioni anche al punto B, ma nel trattare un numero crescente di processi specifici (e oggetti di calcolo) identificabili in questa regione, i medici non possono comportarsi meglio di un calcolatore. È quasi certo che questa tendenza continuerà.

Sarebbe quindi particolarmente spiacevole se i medici considerassero erroneamente le loro capacità professionali come adatte a quegli aspetti del processo diagnostico che sono per natura oggetto di calcolo, e quindi minimizzassero il loro ruolo indispensabile nelle situazioni generali. Quest'ultimo processo è certamente al di là delle capacità attuali per un calcolatore. Sembra che la scelta tra esseri umani e macchine esista soltanto per soluzioni altamente strutturate e formalizzate; i nostri problemi più interessanti hanno quindi a che fare con la scelta di ruoli appropriati per l'uomo e per la macchina, per il giudizio e per il calcolo [46]. E in questa scelta non sembra presuntuoso suggerire di usare entrambi nei ruoli che svolgono meglio, specialmente poichè questi ruoli appaiono molto diversi.

7. Bibliografia

[1] Fifth Trans-Disciplinary Symposium on Philosophy and Medicine, Los Angeles, California, April 14-16, 1977. In: Englehardt HT Jr, Spicker SF, Towers B, eds. Clinical judgment: a critical appraisal. Dordrecht, Hollans: Reidel, 1979.

[2] Dreyfus HL. *What computers cant' do: the limits of artificial intelligence.* Revised ed. New York: Harper & Row, 1979.

[3] Weizenbaum J. *Computer power and human reason: from judgment to calculation.* San Francisco: WH Freeman, 1976.

[4] Thomas L. *On Artificial intelligence.* N Engl J Med. 1980; 302:506-8.

[5] Elstein AS, Shulman LS, Sprafka SA. *Medical problem solving: an analysis of medical reasoning.* Cambridge, Mass.: Harvard University Press, 1978.

[6] Bleich HL. *Computer evaluation of acid-base disorders.* J Clin Invest. 1969; 48:1689-96.

[7] *Idem. Computer-based consultation: electrolyte and acid-base disorders.* Am J Med. 1972; 53:285-91.

[8] Jelliffe RW, Buell J, Kalaba R. *Reduction of digitalis toxicity by computer-assisted glycoside dosage regimens.* Ann Intern Med. 1972; 77:891-906.

[9] Gorry GA, Silverman H, Pauker SG. *Capturing clinical expertise: a computer program that considers clinical responses to digitalis.* Am J Med. 1978; 64:452-60.

[10] Gorry GA, Kaissirer JP, Essig A, Schwartz WB. *Decision analysis as the basis for computer-aided management of acute renal failure.* Am J Med. 1973; 55:473-84.

[11] Shortliffe EH. *Computer-based medical consultations:* MYCIN. New York: Elsevier, 1976.

[12] Meehl PE. *Clinical versus statistical prediction: a theoretical analysis and a review of the evidence.* Minneapolis: University of Minnesota Press, 1954.

[13] *Idem. Psychodiagnosis: selected papers.* Minneapolis: University of Minnesota Press, 1973.

[14] Elstein AH. *Human factors in clinical judgment: discussion of Scriven's 'clinical judgment.'* In: Englehardt HT Jr, Spicker SF, Towers B, eds. Clinical judgment: a critical appraisal. Dordrecht, Holland: Reidel, 1979:17-28.

[15] Warner H, Toronto AF, Veasy LG. *Experience with Bayes' Theorem for computer diagnosis of congenital heart disease.* Ann NY Acad Sci. 1964; 115:558-67.

[16] Crooks J, Murray IPC, Wayne EJ. *Statistical Methods applied to the clinical diagnosis of thyrotoxicosis.* Q J Med. 1959; 28:211-34.

[17] Overall JE, Williams CM. *Conditional probability programs for diagnosis of thyroid function.* JAMA. 1963; 183:307-13.

[18] Fitzgerald LT, Overall JE, Williams CM. *A computer program for diagnosis of thyroid disease.* Am J Roentgenol Radium Ther Nucl Med. 1966; 97:901-5.

[19] Winkler C, Reichertz P, Kloss G. *Computer diagnosis of thyroid diseases: comparison of incidence data and considerations on the problem of data-collection.* Am J Med Sci. 1967; 253:27-33.

[20] Nordyke RA, Kulikowski CA, Kulikowski CW. *A comparison of Methods for the automated diagnosis of thyroid dysfunction.* Comput Biomed Res. 1971; 4:374-89.

[21] Oddie TH, Hales IB, Stiel JN, et al. *Prospective trial of computer program for the diagnosis of thyroid disorders.* J Clin Endocrinol Metab. 1974; 38:876-82.

[22] Alperovitch A, Fragu P.A *suggestion for an effective use of computer aided diagnosis system in screening for hyperthyroidism.* Methods Inf Med. 1977; 16:93-5.

[23] Weiss S, Kulikowski CA, Safir A. *Glaucoma consultation by computer.* Comput Biol Med. 1978; 8:25-40.

[24] Pople HE. *The formation of composite hypotheses in diagnostic problem solving; an exercise in synthetic reasoning.* Proceedings of the 5th International Joint Conference on Artificial Intelligence. Cambridge, MA, August 22 to 25, 1977.

[25] de Dombal FT, Leaper DJ, Staniland JR, Mc Cann AP, Horrocks JC. *Computer-aided diagnosis of acute abdominal pain.* Br Med J. 1972; 2:9-13.

[26] Lusted L.B. *Twenty years of medical decision making studies.* Presented at the third Annual Symposium on Computer Application in Medical care, Washington, D.C., October 14-16, 1979.

[27] Scriven M. *Clinical judgment.* In: Englehardt HT Jr, Spicker SF, Towers B,eds. Clinical judgment: a critical appraisal. Dordrecht, Holland: Reidel, 1979:3-16.

[28] Minsky ML, ed. *Semantic information* processing. Cambridge, Mass.: MIT Press, 1969.

[29] Pauker SG, Gorry GA, Kassirer JP, Schwartz WB. *Toward the simulation of clinical cognition: taking a present illness by computer.* Am J Med. 1976; 60:981-96.

[30] Pratt V. *Numerical taxonomy – a critique.* J Theor Biol. 1972; 36:581-92.

[31] Gordon B, Ed. *Current medical information and terminology.* 4th ed. Chicago: American Medical Association, 1971.

[32] Grossman JH, Barnett GO, McGuire MT, Swedlow DB. *Evaluation of computer-acquired patient histories.* JAMA. 1971; 215:1286-91.

[33] Slack WV, Hicks GP, Reed CE, Van Cura LJ. *A computer-based medical history system.* N Engl J Med. 1966; 274:194-8.

[34] Friedman RB, Gustafson DH. *Computers in clinical medicine: a critical review.* Comput Biomed Res. 1977; 10:199-204.

[35] Gorry GA. *Computer-assisted clinical decision making.* Methods Inf Med. 1973; 12:45-51.

[36] Szlovits P, Pauker SG. *Computers and clinical decision making: whether, how, and for whom?* Proc IEEE. 1979; 67:1224-6.

[37] Lucas RW, Card WI, Knill-Jones RP, Watkinson G, Crean GP. *Computer interrogation of patients.* Br Med J. 1976; 2:623-5.

[38] Mayne JG, Martin MJ Taylor WF, O'Brien PC, Fleming PJ, *A health questionnaire based on paper-and-pencil medium, individualized and produced by computer.* III. Usefulness and acceptability to physicians. Ann Intera. Med. 1972; 76:923-30.

[39] Chun RWM, Van Cura LJ, Spencer M, Slack WV. *Computer interviewing of patients with epilepsy.* Epilepsia. 1976; 17:371-5.

[40] Winograd T. *Understanding natural language.* New York: Academic Press, 1972.

[41] Buchanan BG, Feigenbaum EA. *Dendral and metadendral: their applications dimensions.* Artif Intell. 1978; 11:5-24.

[42] Shortliffe EH, Buchanan BG, Feigenbaum EA. *Knowledge engineering for medical decision making; a review of computer-based clinical decision aids.* Proc IEEE. 1979. 67:1207-24.

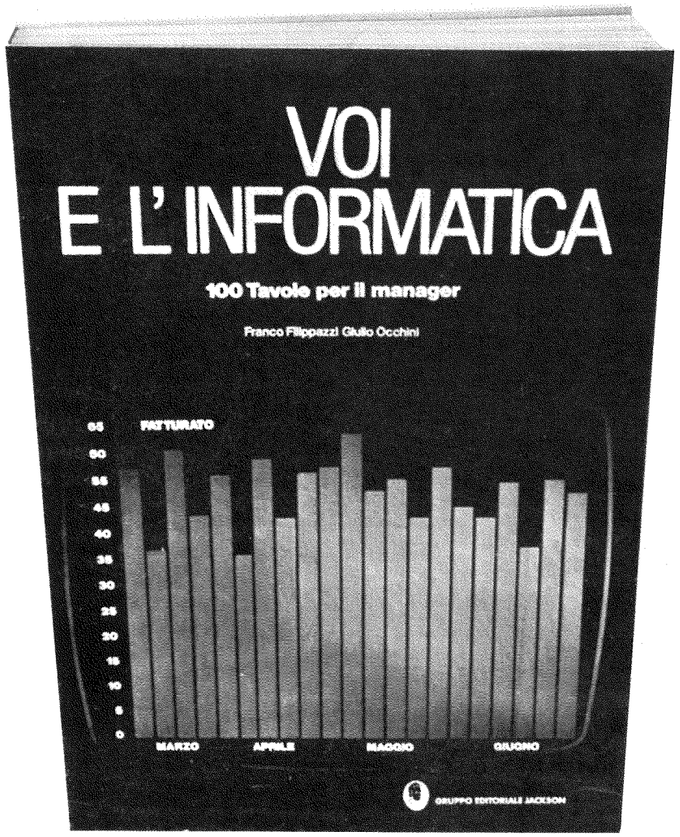
[43] Boden MA. *Artificial intelligence and natural man.* New York: Basic Books, 1977.

[44] Bronowski J. *The origins of knowledge and imagination.* New Haven: Yale University Press, 1978.

[45] Mitchell TM. *Version spaces: a candidate elimination approach to rule learning.* Proceedings of the 5th International Joint Conference on Artificial Intelligence. Cambridge, MA, August 22 to 25, 1977.

[46] Reiser SJ. *Medicine and the reign of technology.* Cambridge: Cambridge University Press, 1978.

BIBLIOTECA
NOVITÀ



VOI E L'INFORMATICA
100 Tavole per il manager

di Franco Filippazzi e Giulio Occhini
Gruppo Editoriale Jackson

L'informazione è oggi una risorsa aziendale della stessa importanza di quelle classiche sinora prese in considerazione dal management, quali le risorse umane, quelle finanziarie, ecc.

Si impone quindi la necessità da parte dei quadri direttivi di acquisire una conoscenza a livello concettuale di questa nuova risorsa, in modo sufficiente per poterne impostare e controllare una razionale strategia di impiego.

A ciò intende contribuire questo volume, che presenta un quadro sintetico, ma tuttavia completo e rigoroso, della materia. La forma di presentazione adottata (per "tavole commentate") si caratterizza per incisività e immediatezza.

Gli Autori sono due noti esperti di informatica ed entrambi fanno parte della redazione dei *"Quaderni di Informatica"*.

BIBLIOTECA
NOVITÀ

Sistemi di trasmissione dati e di teleelaborazione

di Trevor Housley

È uscito il numero 9 della "Collana dei Quaderni di Informatica" edita dalla Franco Angeli e curata dalla Honeywell Information Systems Italia. Si tratta di "Sistemi di trasmissione dati e teleelaborazione" di Trevor Housley.

La diffusione dei sistemi di teleelaborazione e delle reti di trasmissione dei dati ha subito un enorme incremento in questi ultimi anni. Il fenomeno è stato reso possibile dai grandi progressi tecnologici ottenuti in molti settori. Da un lato gli elaboratori dispongono di una velocità, potenza e affidabilità sempre maggiori a parità di costo. Dall'altro i mezzi di comunicazione che possono essere utilizzati per la trasmissione dei dati si sono moltiplicati ed avvolgono l'intero globo in una rete capillare costituita da linee telefoniche e telex, satelliti e cavi coassiali, sistemi a microonde e fibre ottiche. Il software dal canto suo è divenuto sempre più ricco e complesso, ma anche più affidabile e sicuro. Non sono mancati infine progressi verso la standardizzazione degli elementi che compongono quei sistemi. Eppure, nonostante l'enorme attività svolta in questo settore dell'informatica, l'impegno rivolto alla divulgazione dei concetti che stanno alla base della trasmissione dei dati e della teleelaborazione è rimasto finora piuttosto inadeguato.

Trevor Housley intende colmare questo vuoto con il presente volume. Australiano, consulente di trasmissione dati, Housley ha una lunga esperienza didattica che indubbiamente traspare da queste pagine nelle quali è riuscito ad esprimere l'intera materia in modo sistematico, in forma semplice, comprensibile e priva di ambiguità. Un impegno da cui può trarre vantaggio anche l'esperto.

Dopo una parte relativa all'esame dei concetti fondamentali della trasmissione dei dati in cui vengono riproposte in modo piano definizioni, codici e modalità di trasmissione dei dati, il volume analizza gli elementi che compongono le reti, i metodi di individuazione e correzione degli errori, i protocolli di trasmissione e le procedure di controllo di linea, i servizi offerti dalle compagnie telefoniche e i passi fondamentali da seguire nella pianificazione dei sistemi. Il lettore può così acquisire la preparazione necessaria ad affrontare i problemi tipici che si incontrano nella progettazione dei sistemi di teleelaborazione o che comunque ricorrono nella trasmissione dei dati.