CAP. 1

1. AUTOMAZIONE DI FABBRICA

- Sintetica presentazione del CIM, con funzioni, macchine e strumenti ai diversi livelli.
- Generalità sui processi industriali controllati mediante calcolatori.
- Breve analisi delle principali funzioni dei calcolatori nei processi industriali
  - ACQUISIZIONE
  - MONITORAGGIO
  - CONTROLLO
  - ALLARMI
  - VISUALIZZAZIONE
  - MEMORIZZAZIONE STORICA
  - STAMPA RAPPORTI
  - COMUNICAZIONE COL LIVELLO SUPERIORE
- Diversi ruoli tecnici e interfaccia uomo-macchina, con analisi delle caratteristiche attitudinali umane.
- Particolari requisiti dell'elaborazione negli ambienti di controllo di processo, negli aspetti architetturali e programmativi.
- Introduzione all'elaborazione in tempo reale e definizioni
- Cenni sull'affidabilità dei calcolatori industriali.

1.1 L’ATTIVITA’ PRODUTTIVA E IL RUOLO DEI CALCOLATORI

Visione dei ruoli dei calcolatori ad alto livello di astrazione e centrata sulla fabbrica.

1.2 L'AMBIENTE CIM

1.2.1 COS'E' IL CIM

CIM è l'acronimo di **Computer Integrated Manufacturing** e designa un'impostazione dei sistemi produttivi basata su un sistematico e organico impiego di strumenti informatici come supporto o ausilio nelle varie attività che concorrono agli obbiettivi finali di produzione.

L'idea di utilizzare dove e come possibile i calcolatori negli uffici amministrativi, negli uffici tecnici e nei reparti di produzione non è certo nuova, ma ha avuto un notevole impulso con l'evoluzione delle tecnologie, dette “abilitanti”, che sempre più consentono l'integrazione, tramite reciproche interazioni, di strumenti informatici con sempre migliori rapporti prestazioni / costi.

Lo scopo di questa sezione non è quello di approfondire l'analisi delle problematiche CIM, che tra l'altro comprendono complessi aspetti di organizzazione del lavoro, distribuzione di compiti e responsabilità, ecc., quanto quello di presentare in generale un ambiente in cui tipicamente si collocano la maggior parte delle applicazioni industriali dei calcolatori e dell'informatica.

E' molto importante rilevare che il CIM costituisce una trama che molto raramente è effettivamente realizzata con una buona copertura, tuttavia costituisce un contesto concettuale significativo. Infatti è sempre più importante pensare e progettare i sistemi basati su calcolatori come sottosistemi di un più ampio contesto, appunto di tipo CIM.

Aspetti caratteristici dell'integrazione prefigurata dall'approccio CIM sono:

- Basi di dati centralizzate ma accessibili dai vari tipi di processi.
- Comunicazione di informazioni tra i diversi sottosistemi di elaborazione.
- Coordinamento e sincronizzazione delle varie attività umane ed automatiche.
- Ottimizzazione dei tempi e della flessibilità di risposta ad esigenze di mercato rapidamente mutevoli.
- Livello di qualità dei prodotti elevato e controllato.

Alcune organizzazioni promotrici di approcci CIM ne hanno proposto modelli di riferimento che servissero a meglio capire e standardizzare le funzioni e le operazioni.
Solitamente l'ambiente CIM è rappresentato con una forma piramidale a **livelli** che descrive la gerarchia tra le diverse funzioni e l'architettura informatica su cui esse si possono naturalmente mappare.

1.2.2 LIVELLI DEL CIM

Vediamo ora brevemente la struttura a 7 livelli selezionata dall'ISO (TC184).

- - 6 - Azienda

- - 5 - Fabbrica / Stabilimento
- - 4 - Officina / Area
- - 3 - Reparto / Cella
- - 2 - Unità operatrice
- - 1 - Apparato / Sensore
- - 0 - Dispositivi meccanici

Ad ogni livello sono svolte delle funzioni di base:
- Scomposizione dei comandi ricevuti dal livello superiore in sottocompiti;
- Assegnazione dei sottocompiti e delle risorse al livello stesso ed al livello inferiore;
- Attuazione dei sottocompiti del livello;
- Analisi delle informazioni di risposta ricevute dal livello inferiore;
- Rapporto al livello superiore.

Si notino le due direzioni lungo le quali si sviluppano le integrazioni.
- Integrazione intralivello = orizzontale
- Integrazione interlivello = verticale

Ogni livello, oltre alle funzioni di base, ha specifiche responsabilità, orizzonti temporali di pianificazione e compiti da svolgere, che vengono sinteticamente elencati nel seguito.

1.2.2.1 Livello 6 - Azienda

Responsabilità: raggiungimento della missione aziendale.
Orizzonte: lungo termine (vari mesi)
Funzioni:
stabilire gli obbiettivi aziendali
stabilire le politiche e le strategie aziendali
pianificazione finanziaria
pianificazione dello sviluppo aziendale
coordinamento delle attività verso l'esterno
coordinamento delle attività degli stabilimenti
progettazione dei prodotti e dei processi

1.2.2.2 Livello 5 - Fabbrica / Stabilimento

Responsabilità: implementazione delle funzioni produttive e resoconto verso l'Azienda.
Orizzonte: medio termine (varie settimane)
Funzioni:
gestione della produzione
ottimizzazione e coordinamento
schedulazione
gestione delle informazioni sulla produzione

1.2.2.3 Livello 4 - Officina / Area

Responsabilità: coordinamento della produzione e supporto dei lavori dell'officina, gestendo le necessarie risorse.
Orizzonte: da alcuni giorni ad alcune settimane.
Funzioni:
schedulazione
supervisione
monitoraggio impianti

movimentazione
manutenzione
controllo qualità

1.2.2.4 Livello 3 - Reparto / Cella

Responsabilità: sequenziazione e supervisione delle lavorazioni di famiglie di pezzi e dei relativi servizi di supporto.
Orizzonte: da alcune ore ad alcuni giorni.
Funzioni:
analisi della necessità di risorse
rapide decisioni logistiche
sequenziazione e distribuzione dei compiti
raccolta dati di monitoraggio

1.2.2.5 Livello 2 - Unità operatrice

Responsabilità: automazione locale di un'apparecchiatura o di funzioni specifiche.
Orizzonte: da alcuni millisecondi ad alcune ore.
Funzioni:
regolazione e controllo
raccolta dei segnali di stato
diagnostica di guasti e anomalie

1.2.2.6 Livello 1 - Apparato / Sensore

Responsabilità: realizzazione di comandi.
Orizzonte: da alcuni microsecondi ad alcuni minuti.
Funzioni:
generazione di segnali
esecuzione di comandi

1.2.2.7 Livello 0 - Dispositivi meccanici

Responsabilità: lavorazioni.
Orizzonte: da alcuni microsecondi ad alcuni minuti.
Funzioni:
azioni e movimenti

1.2.3 PROCEDIMENTI DI SVILUPPO DEL CIM

La realizzazione di un ambiente in cui l’automazione di processo e le attività amministrative sono gestite in modo integrato con strumenti informatici non è certo cosa facile. Inoltre tale realizzazione costituisce un obbiettivo a cui tendere procedendo per gradi da situazioni già esistenti e spesso funzionanti in modo ancora soddisfacente.
Non si intende in questa sede entrare nell’analisi dei vari e complessi problemi di implementazione o introduzione del CIM, ma solo accennare a possibili percorsi evolutivi.

- APPROCCIO TOP-DOWN

L'approccio *top-down* consiste nell'impostare il progetto di una nuova fabbrica basandosi su un preciso riferimento di struttura CIM. La scelta e l'organizzazione delle varie componenti sono quindi fatte in modo da facilitare la loro integrazione.
Chiaramente si tratta di un approccio coerente ma piuttosto drastico, in quanto presuppone di partire da una situazione di *tabula rasa* e di imporre al personale e alle macchine stringenti vincoli di adeguamento ad una filosofia prefissata. Inoltre non è facile trovare sul mercato elementi (umani e di macchine) predisposti per una facile e completa integrazione.

- APPROCCIO BOTTOM-UP
L'approccio *bottom-up* prende le mosse da una situazione già esistente ed individua interventi locali di informatizzazione di macchine, procedure e attività di uffici che si prestano ad una facile realizzazione e che sono più promettenti in termini di vantaggi globali.
Questo percorso si presenta più semplice inizialmente, data la località degli interventi e delle modifiche, ma può portare a realizzazioni, pur singolarmente valide, che poi risulta però difficile integrare tra loro a causa di incompatibilità di vario tipo.

- APPROCCIO MISTO
Compromesso molto adottato, che procede per successive soluzioni locali e graduali integrazioni. Questo modo di procedere può essere considerato di tipo *top-down* per quanto riguarda il progetto che costituisce l'obbiettivo di riferimento, ma *bottom-up* per quanto riguarda il procedere delle realizzazioni.

- FASI
A tutti i livelli si può individuare la sequenza di fasi tipica dello sviluppo (ingegneristico) di sistemi, e cioè:

- posizione del problema
- analisi del problema
- stesura delle specifiche della soluzione
- analisi e conferma delle specifiche
- progetto della soluzione
- validazione del progetto rispetto alle specifiche
- realizzazione
- collaudo al banco e/o con simulazioni
- installazione
- collaudo in sede
- integrazione
- esercizio
- manutenzione.

Come è noto le fasi sopra elencate non sono da intendersi in stretta e singola sequenza, ma anzi è frequente che alcune di esse siano ripetute con un processo di successive approssimazioni verso soluzioni ottime o più soddisfacenti.
In ogni fase sono coinvolti specifici ruoli (finanziari, manageriali e tecnici) e vengono utilizzati opportuni strumenti e macchine.

1.2.4 STRUMENTI HW E SW DEL CIM

Vengono qui sinteticamente elencati strumenti, risorse e funzioni tipiche dei vari livelli. Gli aspetti più strettamente attinenti al corso saranno poi ripresi ed approfonditi nei capitoli successivi.

1.2.4.1 LIVELLI AZIENDA E FABBRICA

- MACCHINE
  - Potenti calcolatori gestionali
  - Number crunching
  - Memorie operative capaci
  - Memoria virtuale gerarchica
  - Memorie di massa molto capaci a dischi e nastri
  - Stampanti molto veloci
  - Terminali interattivi

- SOFTWARE DI BASE
  - Sistemi Operativi multiutente con elevato throughput
  - Grandi Data Base
  - Completa autonomia di sviluppo

- FUNZIONI APPLICATIVE
  - CAD, ottimizzazioni, simulazioni, decisioni Previsioni, marketing, clienti
  - Pianificazione, controllo qualità, manutenzione
  - Acquisti, magazzino, flusso materiali

- LINGUAGGI
  - Ad alto livello
    - COBOL, FORTRAN, PL/1
    - ADA

1.2.4.2 LIVELLO OFFICINA

- MACCHINE
  - Piccoli gestionali, grandi mini, supermicro
  - Elevate capacità di calcolo numerico
  - Memorie operative capaci (8..128 Mbyte)
  - Memorie virtuali
  - Memorie di massa capaci a dischi (e nastri)
  - Stampanti
  - Terminali interattivi

- SOFTWARE DI BASE
  - Sistemi Operativi con limitato real-time (foreground/background)
  - Data Base
  - Sviluppo autonomo e *Cross-software* per livelli inferiori

- FUNZIONI APPLICATIVE
  - Monitoraggio mediante interazione con i punti di lavoro e controllo di flusso dei materiali.
  - Diagnostica e gestione delle emergenze.
  - Acquisizione e compressione dei dati.
  - Controllo di qualità.
  - Controllo e ottimizzazione allocazione risorse.
  - Analisi storica e statistica dei dati.
  - Gestione dei satelliti: downloading, attivazione, diagnosi.

- LINGUAGGI
  FORTRAN, BASIC,
  C, ADA

1.2.4.3 LIVELLO REPARTO

- MACCHINE
  Microcalcolatori 16..32 bit
  Calcolatori industriali.
  Coprocessori matematici
  Memoria operativa (1..16 Mbyte)
    RAM, EPROM, RAM Tampone.
  Memorie di massa a dischi, se le condizioni ambientali lo consentono.
  Interfacce operatore con video semigrafici e grafici
  Piccole stampanti.
  Interfacce di processo (I/O)

- SOFTWARE DI BASE
  Sistema Operativo residente real-time
  Consente semplici modifiche delle applicazioni.

- FUNZIONI APPLICATIVE
  Avviamento ed arresto dell'impianto.
  Algoritmi di controllo “in grande”.
  Acquisizione e riduzione (medie) dei dati.
  Analisi di tendenza.

- LINGUAGGI
  Ad alto, medio e basso livello
    FORTRAN, BASIC
    ASSEMBLER
    C, ADA

1.2.4.4 LIVELLO UNITA' OPERATRICE

- MACCHINE
  - Microcalcolatori 8..16 bit, PLC, single-chip.
    - Sono detti **embedded**, che alla lettera significa “incastonati”, i calcolatori che sono considerati parte integrante di un sistema (macchina, strumento, apparecchiatura) più complesso.
    - Sono detti **CNC** (Computer Numerical Control) i calcolatori specificamente destinati a controllare le lavorazioni di macchine utensili.
  - Semplice ed efficiente set di istruzioni (logiche, I/O).
  - Memorie operative 8K..1M RAM, ROM, EPROM, RAM tampone.
  - Pannelli operatore con pulsanti, contraves, display, luci.
  - Interfacce di processo digitali e analogiche.

- SOFTWARE DI BASE
  - Semplice nucleo di S.O. real time (sincrono o a coroutine)
  - Consente solo modifiche semplici e previste dell'applicazione, che viene sviluppata su macchine separate o dei livelli superiori.

- FUNZIONI APPLICATIVE
  - Controllo di sequenze, temporizzazioni, consensi, blocchi.
  - Algoritmi di controllo “locali”.
  - Controllo movimenti.
  - Gestione sensori.
  - Rilievo di allarmi.
- LINGUAGGI
  - ASSEMBLER = linguaggio macchina in forma simbolica
  - Linguaggi grafici specifici
    - KOP, FUP, GRAFCET
  - C, PL/M, BASIC, FORTRAN

Nei **livelli superiori** si ha a che fare con informazioni più ricche e strutturate, gestite più direttamente dall'uomo.

Informatica gestionale
- basi di dati
- analisi statistica
- sistemi di supporto alle decisioni DSS
- gestione di documenti
- contabilità

Nei **livelli inferiori** la maggior parte delle informazioni e delle prestazioni sono correlate con l'interazione tra calcolatori e fenomeni fisici che hanno sede in opportune macchine.

Informatica industriale
- raccolta dati
- automazione
- regolazione
- sicurezza
- controllo qualità

Non sono diversi solo gli algoritmi, ma soprattutto sono diversi i requisiti sulle prestazioni richieste in termini di
- interazioni col mondo esterno

tempi di risposta
robustezza
reazione alle anomalie

Si noti che la suddivisione delle funzioni tra i vari livelli può essere considerata relativamente stabile, mentre invece si osserva una variazione, talora anche notevole, di risorse hardware (CPU, memorie, display, ecc.) e di strumenti software (Linguaggi, sistemi operativi, pacchetti applicativi, ecc.) in conseguenza dell'evoluzione tecnologica e di mercato.

1.3 I PROCESSI INDUSTRIALI

Chiamiamo Processo Industriale una combinazione, talora anche molto complessa, di funzioni destinate a gestire fenomeni che operano trasformazioni nel mondo fisico.
La casistica è troppo vasta per essere esemplificata in modo esaustivo, ma possiamo considerare i seguenti esempi.

- Trasformazioni di materia (impianti chimici), trasformazioni materia→energia (centrali termoelettriche), trasformazioni di energia (riscaldamento o raffreddamento di ambienti).
- Movimentazione di materiali sfusi (silos, miscelatori, nastri trasportatori), o di materiali confezionati (magazzini di pallets).
- Controllo di movimenti di macchine (macchine utensili, robot, macchine confezionatrici)
- Controllo di motori elettrici e termici.
- Gestione di impianti di distribuzione di energia.
- Gestione di strumenti di misura di vario tipo, anche complessi (inquinamento aria, acqua).

Si noti che il termine *industriale* è utilizzato con un significato molto allargato rispetto a quello suo proprio, designando anche sistemi per applicazioni su veicoli (automobili, velivoli, ecc) in laboratori, e addirittura di tipo domestico.
Il termine industriale è usato quindi più per designare una differenziazione da altri termini come informativo, matematico, ecc., che per riferirsi ad entità esclusive degli ambienti industriali.

1.3.1 MODELLO DEL PROCESSO

Un modello di processo, rappresentato ad elevato livello di astrazione nella fig. 1.1, viene normalmente descritto in termini delle **leggi fisiche** che governano i fenomeni rilevanti del processo stesso. In una visione ingegneristica non devono però essere dimenticati i fenomeni che producono effetti non desiderati. Per tali fenomeni non è facile, e spesso neppure necessario, individuare un accurato modello fisico, ma generalmente ci si limita a caratterizzare (ad es. in termini probabilistici) gli effetti che essi producono e che vengono detti **disturbi**.
La presenza di disturbi rende approssimate le descrizioni matematiche dei fenomeni, e richiede che non venga trascurata la problematica delle **tolleranze**, cioè degli scostamenti accettabili rispetto ai comportamenti nominali.

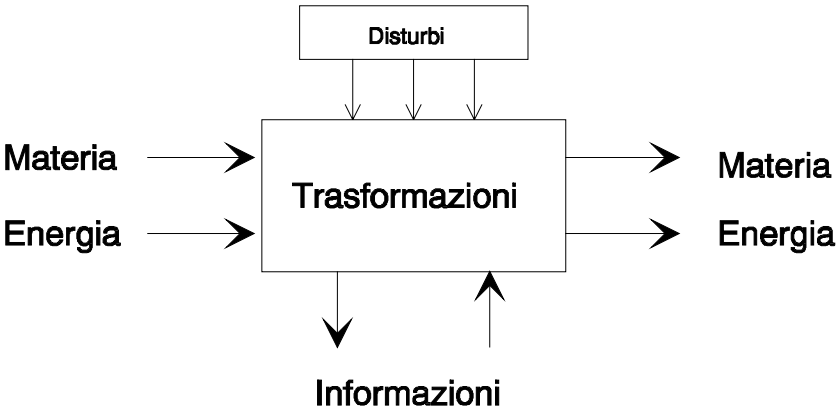


Fig. 1.1 - Modello generale di processo.

Nella fig. 1.1 le informazioni in ingresso sono i parametri controllabili che agiscono sul processo, mentre le informazioni in uscita sono variabili che descrivono lo stato del processo.

1.3.2 PROCESSO CONTROLLATO DA CALCOLATORE

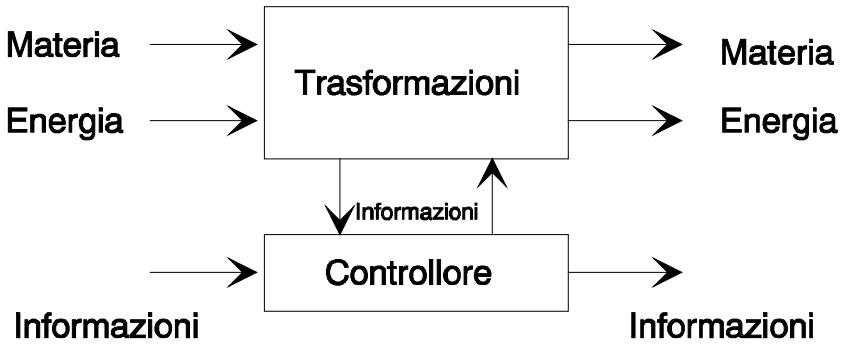


Fig. 1.2 - Modello di processo con controllore.

1.3.3 TIPI DI CONTROLLO

SEQUENZE

Le varie operazioni, che spesso vengono ripetute ciclicamente, sono attivate in base ad una temporizzazione oppure sono abilitate dal completamento delle operazioni precedenti.

MOVIMENTI

Per questi controlli assumono particolare rilievo le grandezze che rappresentano posizioni, velocità e accelerazioni. In alcuni casi si vuole seguire traiettorie predefinite, mentre in altri casi l'obbiettivo è di inseguire un riferimento imprevisto.

TRASFORMAZIONI FISICHE

Nei processi di trasformazione, che possono essere di tipo continuo o a lotti, sono in genere richiesti controlli (virtualmente) continui, basati sull'acquisizione di grandezze continue e sulla regolazione mediante la gestione di grandezze manipolabili continue oppure di tipo ON/OFF.

GESTIONE DI DATI

Anche le informazioni, analogamente a materia ed energia, possono richiedere meccanismi per la trasmissione ed elaborazione.

1.3.4 INTERAZIONI PROCESSO-CONTROLLORE

STATO DEL PROCESSO

Lo stato del processo è descritto dai valori di opportune grandezze caratterizzanti, che costituiscono le tipiche informazioni che il controllore deve ricevere, più o meno direttamente, dal processo stesso.

RILEVABILITA' (variabili osservabili)

Mentre alcune grandezze, o in generale informazioni, possono essere direttamente rilevate, altre possono essere ricavate solo indirettamente. Ciò può essere dovuto a difficoltà di accesso (ad es. temperature all'interno di un corpo solido) oppure alla natura stessa delle grandezze (ad es. le potenze meccaniche,

elettriche o termiche, calcolabili in genere come prodotto di due termini o come derivate del lavoro).

RILEVAMENTO (sensori)

Il rilevamento dei valori delle grandezze fisiche del processo si basa generalmente su dispositivi ausiliari, appositamente introdotti nel sistema, detti sensori e trasduttori.

In questi dispositivi hanno sede fenomeni fisici che “trasformano” le grandezze da rilevare in altre (dette segnali) più adatte ad essere trasportate e convertite in informazioni elaborabili.

MODIFICABILITA’ (variabili di controllo)

Le informazioni destinate ad influire sull’evoluzione del processo assumono la forma di comandi o variabili di controllo. Le leggi fisiche faranno in modo che queste variabili di controllo producano modifiche dello stato del processo, che è lo scopo ultimo del controllo..

MODIFICA (attuatori)

La trasformazione da informazione (generalmente rappresentata mediante segnali) a grandezza fisica di controllo viene effettuata da opportuni dispositivi detti attuatori. Si noti che dal punto di vista energetico sensori e attuatori non sono simmetrici. Dovendo modulare livelli energetici consistenti, gli attuatori sono generalmente più costosi ed ingombranti dei sensori.

1.3.5 ARCHITETTURA DI SISTEMA DI AUTOMAZIONE

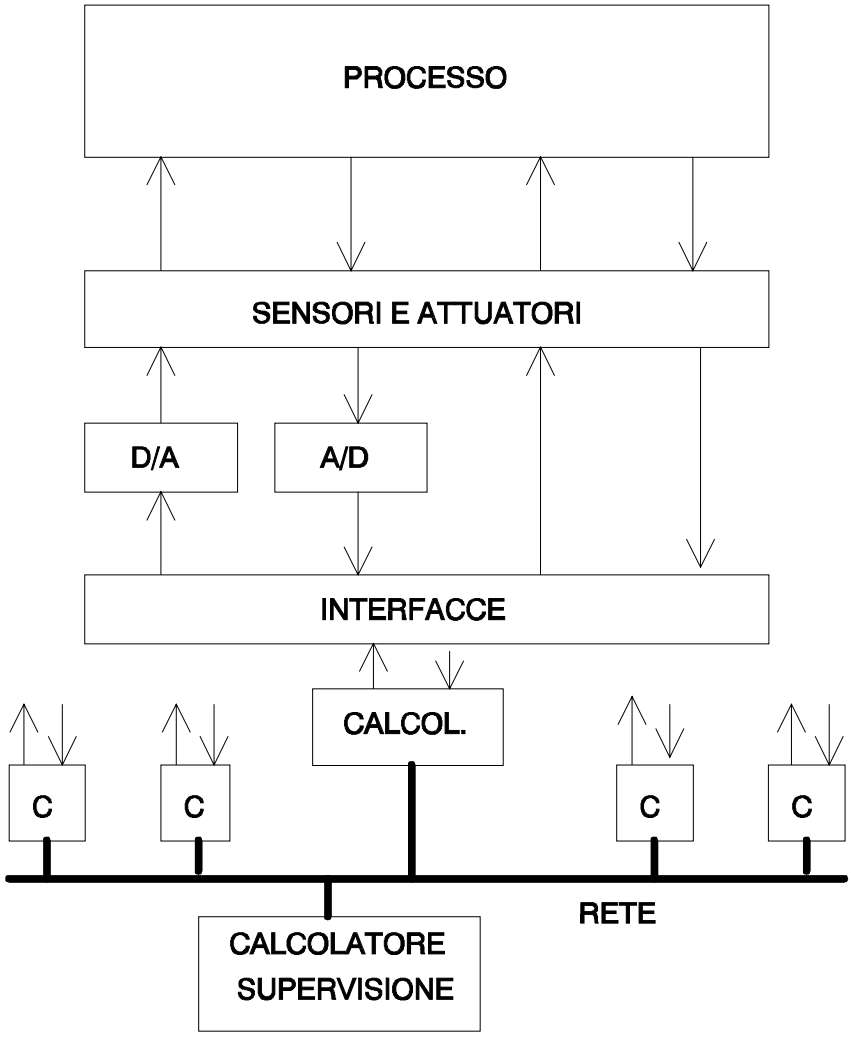


Fig. 1.3 - Architettura di massima di un sistema di automazione distribuito.
Per uno solo dei calcolatori di processo (C) è evidenziata in figura l'interazione con una parte del processo da controllare.

Nella fig. 1.3 sono rappresentati i principali componenti di un sistema di automazione basato su calcolatori.
Il caso più generale di automazione distribuita prevede che siano utilizzati diversi calcolatori collocati, anche relativamente lontani tra loro, nelle posizioni più adatte e collegati in modo da poter cooperare scambiandosi le opportune informazioni.
I collegamenti tra calcolatori possono essere dedicati (punto a punto) oppure, come avviene sempre più frequentemente, possono realizzare una rete locale. Nella fig. 1.3 la rete (in neretto) è realizzata con la topologia a bus, tipica delle reti locali per i sistemi di automazione.

1.3.6 SCHEMA FUNZIONALE DEL SISTEMA DI AUTOMAZIONE

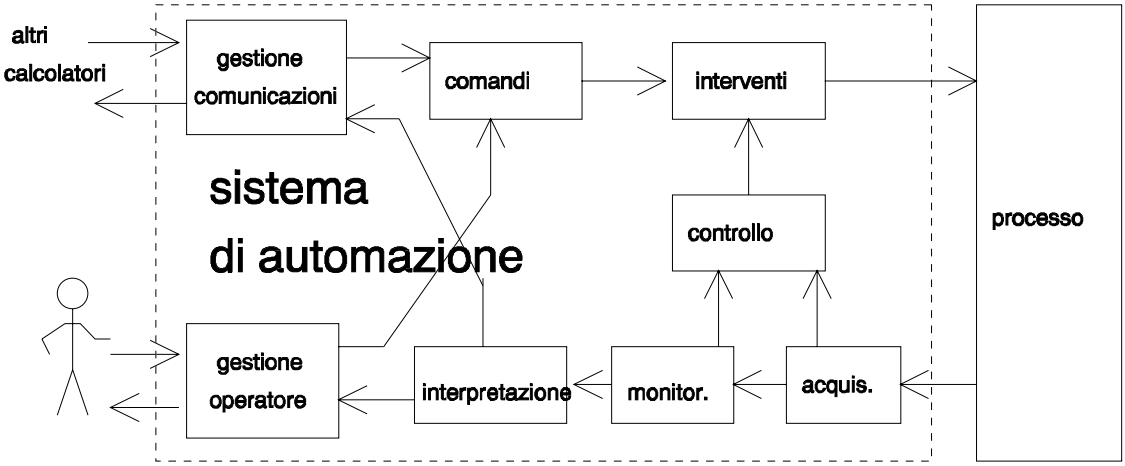


Fig. 1.4 - Principali blocchi funzionali e flussi di dati in un sistema di automazione di processo.

1.3.7 FUNZIONI DI UN CALCOLATORE DI AUTOMAZIONE

Analizziamo ora brevemente le funzioni che vengono sempre più spesso affidate ai calcolatori nei livelli inferiori del CIM, quelli cioè che costituiscono il cosiddetto sottosistema di automazione e che fungono anche da anello di collegamento con gli strati superiori, più orientati alle informazioni.
Le funzioni dei calcolatori di automazione possono essere raggruppate nelle seguenti categorie:

- ACQUISIZIONE
- MONITORAGGIO
- CONTROLLO
- ALLARMI
- VISUALIZZAZIONE
- MEMORIZZAZIONE STORICA
- STAMPA RAPPORTI
- COMUNICAZIONE COL LIVELLO SUPERIORE

1.3.7.1 ACQUISIZIONE

L'acquisizione di informazioni sugli stati ed eventi che caratterizzano l'evoluzione di macchine ed impianti, costituisce un aspetto a cui i calcolatori sono particolarmente adatti e risulta molto interessante anche nei casi in cui altre funzioni di automazione e controllo sono ancora affidate, più tradizionalmente, a dispositivi dedicati.

In alcuni casi l'acquisizione dati è impostata come primo passo per una migliore conoscenza del modello dell'impianto e per un inizio di controllo di qualità.
Ovviamente l'acquisizione dati costituisce in ogni caso una funzione di base anche per le funzioni più complesse (monitoraggio, controllo, ecc.), che vengono analizzate nei paragrafi seguenti.

E' importante tener presente che in ogni caso l'acquisizione delle informazioni deve essere impostata con politiche e tecniche che tengano ben conto della successiva destinazione di tali informazioni, che generalmente impone vincoli, anche molto stringenti, sul modo di estrarre le informazioni significative e sulle correlazioni temporali tra i fenomeni esterni che le generano e gli istanti in cui esse sono disponibili: tipico problema di tempo reale.

Come si vedrà, parlando di interfacciamento, le informazioni di **stato** sono acquisite con **campionamenti** effettuati ciclicamente con opportune frequenze, mentre le informazioni associate agli **eventi** vengono estratte dai **cambiamenti di stati** o, per gli eventi più "urgenti", tramite generazione e **servizio di interruzioni** (interrupt).

Le strutture dati relative a tali informazioni sono quindi costituite generalmente da **un'immagine di stati** globale e da **code di eventi**.

Molte delle problematiche tipiche delle funzioni di acquisizione verranno trattate, come si è ricordato, nei capitoli dedicati all'interfacciamento (cap. 5,6,7,8).

1.3.7.2 MONITORAGGIO

Per monitoraggio si intende normalmente l'insieme di funzioni di automazione di livello superiore, che non riguardano cioè gli aspetti più direttamente legati alle singole e rapide operazioni, ma operano su fasi a tempi più lunghi.
Spesso queste funzioni fanno soprattutto da tramite tra l'impianto e l'operatore che ne conduce l'esercizio.

Una particolare attenzione agli aspetti temporali è richiesta per le funzioni di rilievo delle anomalie e di intervento per ridurne gli effetti negativi. Spesso l'intervento consiste nel blocco, eventualmente coordinato, delle attività che hanno prodotto la situazione non voluta e di quelle che da tale situazione potrebbero essere dannosamente influenzate.

1.3.7.3 CONTROLLO

Per controllo e regolazione si intendono le funzioni tipicamente "attive" di un sistema di automazione, che consistono cioè nell'emettere **comandi** destinati ad influire sui fenomeni esterni per guidarne l'evoluzione secondo i criteri specificati.

Le **regolazioni**, generalmente ad anello chiuso, si propongono il mantenimento di singoli parametri fisici del processo nell'intorno dei valori voluti. Sono state sviluppate tecniche ed algoritmi che consentono ai calcolatori di realizzare (o approssimare) le funzioni di trasferimento che la teoria dei controlli consente di individuare come corrette o ottimali.
Questa funzionalità, nell'ambito discreto tipico dei calcolatori, viene ottenuta **virtualizzando il continuo** con una ripetizione ciclica, a cadenza sufficientemente frequente e regolare, di algoritmi di calcolo dei valori da emettere in uscita verso gli attuatori, in funzione dei valori acquisiti in ingresso e di parametri opportunamente "tarati".

Il termine **controllo** copre generalmente una funzionalità più ampia che riguarda la generazione di sequenze di comandi discreti, la gestione di transitori, il coordinamento di processi interagenti, ecc.

Spesso si utilizza, come modello implementativo dei controllori, un insieme di automi a stati finiti, ognuno corredato da una coda di simboli in ingresso di cui l'automa è consumatore per eseguire le transizioni di stato.
Il vocabolario di ingresso di questi automi è costituito da:
- eventi esterni
- eventi interni
- comandi generati da altri automi
- eventi temporali di varie categorie, come:
  - scadenza della durata di fasi
  - scadenza di orari
  - scadenza di *time-out* per mancati eventi

Implementativamente si può adottare un processo (*task*) per ogni automa se si devono differenziare diversi livelli di priorità (urgenza), mentre è sufficiente un unico *task* che periodicamente scandisce le code di ingresso dei vari automi e ne effettua le transizioni, nel caso in cui le urgenze siano indifferenziate.
Nel cap. 11 vengono presentate alcune tecniche implementative di automi.

1.3.7.4 ALLARMI

Una funzione strettamente legata sia al semplice monitoraggio che al controllo, è costituita dalla cosiddetta gestione degli allarmi.

Per **allarme** si intende la **segnalazione di un'anomalia**. Un'anomalia è generalmente associata al valore TRUE di un'espressione booleana (asserzione di anomalia) che rappresenta una condizione non desiderabile, come ad esempio:

(motore_in_moto) and (pressione_olio < 1.5)

Scopi degli allarmi sono la **segnalazione** ad operatori umani delle anomalie, l'eventuale azione di **blocco** (per gli allarmi detti "bloccanti") e l'eventuale **memorizzazione** storica come eventi significativi dell'esercizio dell'impianto.

Ad un allarme sono associati:
- un nome, cioè una stringa di caratteri che lo identificano univocamente da parte del personale dell'impianto;
- uno stato appartenente ad un opportuno insieme;
- una segnalazione audio-visiva dipendente dallo stato;
- un insieme di azioni associate alle transizioni di stato.
- un comando di tacitazione
- un eventuale comando di ripristino per gli allarmi bloccanti

Discutiamo brevemente questi aspetti.

1.3.7.4.1 Identificatori degli allarmi

Gli **identificatori** (nomi o frasi) sono scelti in sede di progetto. Spesso è utile associare ad ogni allarme anche un codice compatto (numerico) che facilita l'elaborazione informatica, mentre il nome in chiaro può essere composto da più parole concatenate per rispecchiare la gerarchia spaziale che localizza il fenomeno interessato. Ad esempio si potrà scegliere il nome

sovratemperatura.motore_3.macchina_X

1.3.7.4.2 Stati degli allarmi

Gli **stati** che un allarme può assumere dipendono dalle scelte di progetto e dalle normative adottate che propongono degli standard.
Senza qui affrontare il discorso in modo approfondito, portiamo come esempi due diagrammi stati-transizioni tipici di una gestione semplice e di una gestione più articolata di allarmi.

a) **Stati di automa per allarme semplice (Raccomandazione ISA1)**

STATO	LUCE	SIRENA
RIPOSO	spenta	tace
ALLARME	pulsa	suona
TACITATO	fissa	tace
RIENTRATO	pulsa	suona

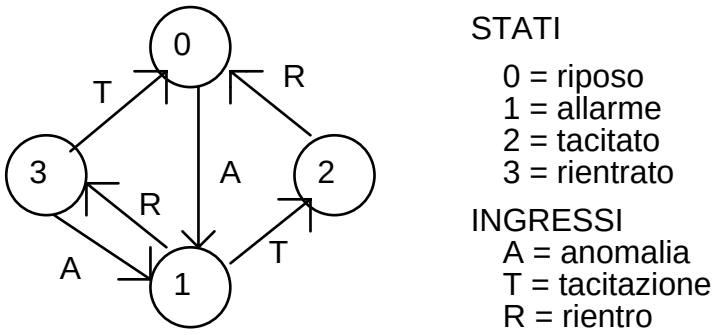
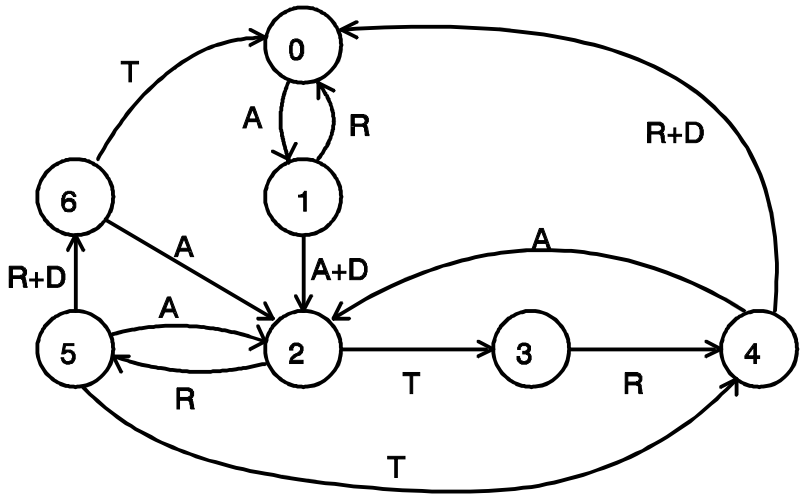


Fig. 1.5 - Diagramma Stati - Transizioni di un automa di allarme semplice.

Il comando di tacitazione indica il riconoscimento (*acknowledge*) dell'anomalia da parte dell'operatore. Poichè tale comando è necessario per tornare allo stato di riposo, nessun allarme, eventualmente anche rientrato, può passare inosservato dall'operatore.

b) **Stati di automa per allarme complesso**

N.S	Nome stato	Sirena	Luce	Messaggio
0	RIPOSO	tace	spenta	--
1	RITARDO ALL'USCITA	tace	spenta	--
2	ALLARME	suona	lamp. veloce	blinking
3	TACITATO	tace	fissa	fisso
4	IN RIENTRO TACITATO	suona	lamp. lento	blink controfase
5	IN RIENTRO	suona	lamp. lento	blink controfase
6	RIENTRATO	suona	fissa	fisso



A+D significa trascorso ritardo D in presenza di A

Fig. 1.6 - Possibile automa per gestione di allarme complesso.

Con lo schema più complesso ci si propone di evitare che brevi intervalli di tempo in cui l'anomalia si verifica (o rientra) siano sufficienti a provocare transizioni definitive come avverrebbe in caso, ad esempio, di segnali spurii.

1.3.7.4.3 Segnalazioni degli allarmi

Le **segnalazioni** degli allarmi avvengono generalmente a tre livelli:
1) **Sirena**, o comunque generatore di segnale sonoro, che richiama l'attenzione anche quando l'operatore non stia osservando il pannello di controllo.
La sirena viene zittita con il comando di tacitazione, in modo da non disturbare inutilmente e da rendere chiaramente recepibile il verificarsi di un eventuale successivo nuovo allarme.
2) **Luce**, o comunque segnalazione visiva, generica che indica la presenza di almeno un allarme non nello stato di riposo.
In genere questa luce lampeggia se esistono allarmi non tacitati, mentre rimane fissa se gli allarmi sono tacitati ma non rientrati.
In alcuni casi si utilizza un lampeggio lento per indicare la fase di rientro dell'allarme.
3) **Messaggi** specifici che identificano i vari allarmi e, con lampeggi (*blinking*) in fase tra loro o in controfase, ne evidenziano gli stati.
Generalmente questi messaggi sono raccolti per gruppi in apposite "pagine" (schermate) video che possono essere singolarmente richiamate su esplicito comando dell'operatore.

1.3.7.4.4 Azioni degli allarmi

In molti casi la gestione degli allarmi si limita alle segnalazioni appena citate, ma per alcune situazioni di allarme è specificato che vengano effettuate particolari **azioni** che di solito consistono nell'arresto di parti dell'impianto (blocco).
Senza voler approfondire il discorso ed esaurire la vasta casistica che si presenta nei progetti, vogliamo evidenziare alcuni aspetti molto importanti che devono essere attentamente considerati anche dal progettista informatico.
Il blocco di una macchina, e a maggior ragione di un intero impianto, costituisce un danno, sia in termini di mancata produzione, sia per problemi indotti a valle o a monte

della catena funzionale, sia, e spesso è l'aspetto più grave, per i possibili "guasti" indotti da arresti improvvisi.
Si pensi ad esempio a quanto poco desiderabile sarebbe che il rilievo di una bassa pressione dell'olio nella nostra auto producesse il blocco del motore, magari nel bel mezzo di un'autostrada affollata.
Si adotta quindi l'azione di blocco solo quando il costo degli effetti di blocco è minore di quello che si avrebbe col persistere dell'anomalia.

Alcune situazioni che costituiscono normalmente delle anomalie sono invece "fisiologiche" in alcuni particolari stati delle macchine (ad es. bassa pressione dell'olio di un motore fermo) e non devono quindi generare immediatamente allarmi bloccanti, pena l'impossibilità di eseguire transitori che mettano in moto la macchina.

Spesso le anomalie sono a loro volta causa di altre anomalie. Ciò comporta che spesso nelle situazioni di allarme sono numerosi gli allarmi presenti, soprattutto se l'impianto è andato in blocco, ma è importante mantenere evidenziato il cosiddetto "primo fuori", cioè l'anomalia originaria e probabilmente causa di tutte le altre. In sistemi a dinamica rapida gli intervalli tra i vari eventi di allarme possono essere anche molto brevi (pochi ms).

Da quanto detto si intuisce l'importanza di:
- accurata gestione dei valori acquisiti ed eliminazione degli errori introdotti dagli spurii;
- corretta scelta delle espressioni booleane che esprimono le condizioni di anomalia, con l'inserimento in "and" degli stati dell'impianto in cui tali anomalie sono significative;
- correttezza e accuratezza temporale delle operazioni di acquisizione ed elaborazione di stati ed eventi a cui sono associati allarmi, soprattutto se bloccanti.

1.3.7.4.5 Tacitazione degli allarmi

Il comando di **tacitazione** è fornito dall'operatore con diverse modalità (pulsanti, tastiere, mouse, ecc.) a seconda delle scelte implementative, ma sempre rispettando il criterio fondamentale che tale tacitazione deve rappresentare la presa di coscienza dell'allarme da parte dell'operatore che, con tale comando, si assume la responsabilità degli interventi necessari. In alcuni casi tale tacitazione assume un'importanza tale da costituire un evento da memorizzare nel "diario di bordo" storico dell'impianto.
Come conseguenza è buona regola non consentire la tacitazione di un allarme se la corrispondente "pagina" non è attualmente in vista sul video.
In genere la mancata tacitazione di allarmi disabilita le operazioni di avvio di un impianto, mentre per semplici ed innocue macchine si accetta talora che al comando di messa in moto sia associata un'implicita tacitazione degli eventuali allarmi salvo poi, ovviamente, gestire le anomalie che eventualmente persistessero.
Per gli allarmi bloccanti generalmente è previsto che oltre alla tacitazione sia effettuato anche un comando di ripristino (riarmo) prima che sia consentito il riavviamento dell'impianto.

Una tipica tecnica implementativa degli allarmi è ovviamente basata su istanze (quindi distinti contesti di variabili) di un automa (quindi un unico codice), eseguite da un apposito processo ciclico o attivato dagli eventi (uscita, rientro, tacitazione e scadenze temporali) oppure svolte come attività aggiuntive dai processi di acquisizione delle informazioni.

1.3.7.5 VISUALIZZAZIONE

La visualizzazione di informazioni di processo è sempre stata un aspetto importante, ma con le tecnologie elettroniche e poi informatiche ha trovato sempre più strumenti adatti ad una crescente ricchezza di informazioni.

Accanto ai LED e ai display a 7 (e 14) segmenti si trovano infatti sempre più spesso vari tipi di schermi alfanumerici e grafici, basati su tecnologie a raggi catodici (CRT a scansione, come i monitor televisivi), a cristalli liquidi (LCD Liquid Crystal Display) e a plasma.

Il destinatario di queste funzioni è l'operatore umano. Gli aspetti "ergonomici" quindi assumono una notevole rilevanza e vi dedicheremo l'intero paragrafo "Interazioni uomo-macchina".
Qui vogliamo però subito far notare che molte delle tecniche “fantasmagoriche” associate ai video grafici a colori, con finestre e mouse ecc., devono in genere subire un esame critico e talora un pesante adattamento alle esigenze di gestione di impianti, che sono molto diverse da quelle che un operatore informatico generalmente conosce bene.
In altre parole l’informatico che realizza il software di visualizzazione deve cercare di mettersi nei panni dell’operatore in esercizio.

Generalmente le funzioni di visualizzazione sono eseguite con l'attivazione periodica (ad es. ogni 0.5 s) di un *task* che spesso è lo stesso a cui è affidata la gestione del colloquio con l'operatore.
Un approccio interessante è basato, anche qui, su un automa a stati la cui evoluzione segue quella del "protocollo" di colloquio con l'operatore.

1.3.7.6 MEMORIZZAZIONE STORICA

La memorizzazione storica dei dati (stati ed eventi) di esercizio di una macchina o di un impianto è generalmente basata su una o più delle seguenti motivazioni:

- Tenere un diario degli **eventi significativi** della conduzione per
  - gestire la manutenzione preventiva
  - gestioni amministrative di produzione, consumi, ecc.
  - analisi di cause e responsabilità di malfunzionamenti

- Raccogliere **andamenti temporali** di grandezze fisiche significative per
  - "*tuning*" degli algoritmi di controllo
  - studio di modelli del processo
  - bilanci energia / materiali
  - controllo di qualità

A seconda dei casi si avranno diverse quantità di informazioni e diverse frequenze di scrittura sulla memoria di massa (non volatile e capiente) utilizzata per lo scopo.

Si noti che rispetto all’uso delle **memorie di massa** si presentano generalmente necessità di risolvere al meglio numerosi e diversi **compromessi** relativi a:
- frequenza di accesso al disco e carico di lavoro imposto alla CPU
- banda passante del canale verso il disco e conflitti di accesso per altre funzioni

- granularità temporale e di valore delle informazioni
- memorie non volatili di deposito temporaneo per mantenere la consistenza delle informazioni anche in caso di caduta rete
- mantenimento della sequenza temporale degli eventi
- dimensionamento della quantità di memoria di massa usata
- compatibilità di formato con i programmi di successiva elaborazione
- organizzazione che facilita ricerche selettive

Generalmente si adotta un singolo task “server del disco” che funge da consumatore per tutte le informazioni da memorizzare su una memoria di massa che esso gestisce, e che viceversa funge da produttore delle informazioni che da tale memoria di massa devono essere lette.

1.3.7.7 STAMPA RAPPORTI

La stampa di informazioni costituisce talvolta un'aggiunta e talaltra una sostituzione delle memorizzazioni, quando si voglia una memorizzazione "cartacea" (*hard copy*) per la consultazione umana delle informazioni salienti.
In genere si effettuano stampe **attivate** da:

- **Scadenza** periodica (oraria, di turno, giornaliera, ecc.) in occasione della quale si riportano dati consuntivi del periodo.
- **Richiesta** da parte dell'operatore che vuole una "fotografia" dei valori di stato significativi in quel momento.
- **Eventi** esterni significativi, come avviamento, cambiamento di produzione, arresto, ecc.
- **Allarmi** ed eventi correlati.

Ovviamente le stampe devono riportare data e ora a cui si **riferiscono** e le informazioni in chiaro, in modo facilmente consultabile ed interpretabile dall'uomo.

La tipica implementazione delle funzioni di stampa è basata su un task "server di stampante" per ogni stampante prevista, in modo da

- disaccoppiare i ritmi di stampa delle informazioni dai ritmi con cui esse sono prodotte
- garantire la necessaria mutua esclusione tra le diverse richieste di stampe.

Questi task saranno corredati di code per le richieste pendenti e utilizzeranno opportuni descrittori di formato per i vari blocchi di informazioni, da stampare come entità non interrompibili. Un descrittore di formato è una struttura (tipo record) contenente un campo per ogni elemento di informazione da stampare (o visualizzare), che a sua volta è una struttura i cui campi descrivono posizione (coordinate X e Y), attributi (colore, colore sfondo, lampeggi, grassetto, ecc.) e puntatore alla variabile che contiene il valore (o la stringa) dell’informazione.

1.3.7.8 COMUNICAZIONE COL LIVELLO SUPERIORE

Anche la comunicazione ad altri calcolatori, generalmente a livello superiore nella gerarchia CIM, costituisce una funzione complementare, o talvolta sostitutiva, della memorizzazione. Spesso infatti il calcolatore a cui sono affidate le funzioni di acquisizione dati e controllo non possiede, nè è adatto a gestire, una memoria di massa.

Oltre alla disponibilità di una certa potenza di calcolo, le memorie a disco richiedono infatti condizioni ambientali non molto ostili in termini di temperature, vibrazioni, polveri, aggressivi chimici, ecc.

Per queste funzioni di comunicazione valgono le considerazioni fatte per la memorizzazione, a cui si aggiungono i problemi di gestione di un protocollo di colloquio e delle relative disomogeneità attitudinali tra i diversi calcolatori.

Infatti uno dei problemi tipici, a questo proposito, è la soluzione del dilemma *colloquio gestito dal calcolatore di acquisizione o da quello di livello superiore*.
La soluzione di affidare al calcolatore di livello gerarchico superiore l'iniziativa di richiedere le informazioni è più consona all'impostazione gestionale tipica del software di questi calcolatori, i cui programmi sono abituati ad andare a leggersi i dati quando ne hanno bisogno..
D'altra parte il calcolatore di acquisizione costituisce il tipico produttore di informazioni, essendo quello che rileva direttamente eventuali eventi interessanti, e spesso non è dotabile di grandi capacità di memoria per le informazioni in attesa di essere "chieste" dal livello superiore.

1.3.8 INTERFACCIA UOMO-MACCHINA E RUOLI

Varie figure e ruoli (capacità responsabilità diritti)

Su una base attitudinale comune a tutti gli esseri umani, ogni persona innesta la propria cultura ed esperienza. A ciò si aggiunge un importante **effetto** dovuto al **ruolo** che una persona svolge, e al relativo contesto.
Si noti a tale proposito che una stessa persona può trovarsi, in momenti diversi, in ruoli diversi e quindi con attitudini diverse. Ad esempio la stessa persona che come programmatore ha una buona dimestichezza con una tastiera "qwerty" (ASCII), quando invece sia inserita come operatore di macchina utensile in un'officina, in posizione eretta e con grossi guanti di pelle di protezione, si troverà impacciato ad usare la stessa tastiera per effettuare i comandi alle macchine.

Quanto detto ci porta a considerare che gli aspetti di "ergonomia" nell'interazione uomo-macchina non possono essere analizzati facendo riferimento alle generiche attitudini umane, ma devono riferirsi anche a specifici ruoli, di cui nel seguito sono elencati i più rilevanti nel contesto che stiamo considerando.
Queste considerazioni sono molto importanti anche nell’informatica, dato che le interazioni uomo-macchina sono sempre più intensivamente basate sulla grande varietà e flessibilità degli strumenti informatici.

1.3.8.1 Ruoli

Committente

E' colui che, individuati gli obbiettivi di una iniziativa, stende le specifiche della realizzazione voluta.

Progettista

E' l'esperto in grado di individuare una soluzione al problema descritto nelle specifiche, conoscendo le tecnologie, metodologie e strumenti necessari per la realizzazione.

Dovrebbe avere competenze anche del dominio applicativo del problema da risolvere.

Costruttore

E' la persona che realizza il sistema descritto nella fase di progetto. Nel caso di software è detto programmatore.

I suoi compiti comprendono anche le verifiche, almeno "al banco", del prodotto.

Installatore

E' la persona che mette in opera il sistema realizzato, in modo che possa entrare in esercizio, cioè svolgere le funzioni che costituiscono l'obbiettivo della realizzazione.

Ha competenze interdisciplinari tra la tecnologia del sistema e le problematiche operative a cui esso è destinato

Supervisore

E' la persona con responsabilità di controllare che il sistema sia opportunamente utilizzato in esercizio.

Conduttore

E' l'operatore del sistema, che deve ben conoscere come utente e con cui interagisce direttamente.

E' esperto dei procedimenti di uso del sistema.

Manutentore

E' la persona con competenze sulla tecnologia implementativa del sistema, in grado di effettuare gli interventi che lo rendono utilizzabile lungo tutto il ciclo di vita (*life- cycle*).

1.3.8.2 Considerazioni sull'ergonomia

L'**ergonomia** studia le caratteristiche che deve possedere una macchina per meglio adattarsi all'uso da parte dell'uomo, tenendo quindi conto delle sue attitudini nello specifico ruolo.

Tradizionalmente l'ergonomia si è interessata degli aspetti **meccanici** che costituiscono la componente più evidente dell'interazione tra uomo e macchina, ma con i sistemi informatici, dotati di sempre crescente ricchezza informativa e funzionale, assumono notevole importanza anche gli aspetti **percettivi, comportamentali e psicologici**.

Da qualche tempo è di moda il termine "*user friendly*" per designare i sistemi di elaborazione che presentano all'operatore un comportamento facile da capire, immediato da utilizzare ed in grado di “perdonare” eventuali errori. Questo termine cattura una buona parte dei significati che attribuiamo al termine, più generale, di ergonomia.

Nel seguito vengono riportate alcune considerazioni, per lo più di buon senso, che non pretendono di affrontare scientificamente, e neppure organicamente, l'importante tema, ma solo costituire uno stimolo per riflessioni ed approfondimenti .

La difficoltà di realizzare un sistema ergonomico deriva dai seguenti fattori:

- difficoltà di conoscere e analizzare le attitudini dello specifico utente, tipicamente diverse da quelle del progettista;
- difficoltà di oggettivare (cioè prendere coscienza, formalizzare e quantificare) le attitudini dell'utente;
- problemi tecnologici e di costo nel realizzare le soluzioni eventualmente individuate.

Nessun'altra scusa, a parte queste difficoltà, giustifica un ingegnere nel trascurare nei suoi progetti le esigenze di ergonomia da parte degli utenti.

1.3.8.3 Caratteristiche attitudinali e percettive

Percezione di informazioni
Relazioni tra informazioni
Scelte tra possibili iniziative
Comandi di attuazione di operazioni
Errori nei comandi

- L'uomo ha bisogno di avere una percezione "sensoriale" di cosa sta succedendo. I calcolatori, contrariamente alle macchine meccaniche, non manifestano la loro attività con movimenti, suoni, odori, ecc., a meno che ciò non sia esplicitamente programmato. L'utente non deve vedere uno schermo immobile se sono in corso delle operazioni.

- L'uomo vuole vedere un riscontro immediato (entro circa 100 ms) dei suoi comandi. Sono infatti estremamente irritanti i ritardi nell'**eco** dei comandi. Se poi l'azione comandata impiega un tempo anche lungo per la sua esecuzione, ciò non crea ansia purchè si abbia avuta la percezione che il sistema ci ha “capito”.

- L'uomo ama costruirsi dei riferimenti spaziali fissi che gli consentano di riconoscere oggetti ed informazioni dalla loro posizione prima che dalla forma e prima che da scritte esplicative. Ogni contesto su video (pagina) in situazioni operative di esercizio, deve quindi essere statico, lasciando gli spostamenti di finestre, oggetti, messaggi, ecc. alla sola fase di personalizzazione iniziale.

- Le informazioni devono rispecchiare le relazioni logiche rispetto all'applicazione, e non, ovviamente, le strutture dati interne ai programmi. In altre parole il progettista deve mettersi nei panni dell’utente e non viceversa. Il tipo e il formato della presentazione delle informazioni deve quindi mapparsi il più possibile sugli schemi mentali dell'utente, creando analogie spaziali e di movimento con gli oggetti reali a cui si riferiscono.

- L’uomo ama costruirsi riferimenti simbolici fissi. I comandi principali devono avere un significato uniforme e indipendente dal contesto. Ciò vale in particolare per i comandi di “conferma” e di “annulla”.

- L'uomo preferisce indicare un comando in un menu che elenca le possibilità attivabili in quel contesto, piuttosto che doversi costruire un'immagine mentale astratta di tali possibilità di cui dover ricordare l'esatta dicitura.

- L'uomo è portato a commettere errori formali e/o logici per vari motivi:
  perchè non sa qualcosa
  perchè non ricorda ciò che sa
  perchè materialmente dà comandi diversi dal voluto
  esistono poi potenzialmente leggerezza e malafede.
  Un programma **deve** quindi prevedere verifiche su **tutti** gli errori materialmente possibili.
- In caso di errore l'uomo vuole vedere chiaramente:
  cosa ha fatto di errato
  perchè è errato o non attuabile
  cosa avrebbe dovuto o potuto fare
  cosa fare per uscire dalla situazione e porvi rimedio.

- L'uomo si sente angosciato se ha la sensazione che il comando che sta per dare diventa immediatamente ed irreversibilmente operativo. E' molto importante poter impostare una scelta, osservare con calma ciò che si è impostato potendolo modificare o annullare, e solo quando si è convinti dare la conferma. In particolare è importante che sia evidenziata la "portata" e la "reversibilità" degli effetti del comando che si sta per dare. Spesso si usano conferme in due tempi.

- Il cambiamento di contesto sul video (cambi di pagine) deve essere lasciato all'iniziativa dell'operatore. Eventuali situazioni (es. allarmi) che richiedono il passaggio a diverse pagine, devono produrre una segnalazione lampeggiante in un angolo dello schermo, che inviti l'operatore a prenderle in esame, dando volontariamente il comando di cambio pagina.

- L'uomo vuole poter scegliere tra diversi livelli di astrazione. Quindi è importante la facilità di commutazione tra viste di insieme sintetiche e viste di dettaglio locali.

- Ogni raggruppamento di informazioni (pagina o finestra) deve presentare elementi **omogenei** come livello di astrazione. Avremo così pagine generali con informazioni importanti, e pagine locali con informazioni dettagliate su una singola parte del processo.

- I colori, oltre ad aggraziare l'immagine su video, possono convogliare utili ed immediate informazioni se sono usati in modo consistente (lo stesso colore sempre con lo stesso significato) e coerentemente con le normali associazioni mentali (ad es. si può adottare la convenzione rosso = caldo, allarme; verde = OK; giallo = attenzione; ecc.). Ovviamente in presenza di normative, queste vanno seguite.

- I lampeggi danno un senso di movimento e richiamano l'attenzione. Frequenze diverse consentono di rappresentare aspetti più o meno urgenti. Lampeggi in controfase di alcuni elementi li distinguono dagli altri.
I periodi di lampeggio fisiologicamente significativi vanno da 0.2 a 2 secondi, con tempo di "spento" compreso tra 0.1 e 0.4 secondi (*duty-cycle* 60% . . 80%). Si noti infatti che ad una rapida occhiata gli elementi spenti troppo a lungo possono non apparire, mentre quelli a lampeggio troppo lento possono appairire come fissi.
Una volta esaurito il compito di richiamare l'attenzione, i lampeggi **devono essere sospesi** perchè è **estremamente** irritante ed affaticante fissare con attenzione qualcosa che lampeggia.

Periferiche di input da operatore

Tastiere dedicate e pannelli con pulsanti, commutatori, ecc.
Tastiere standard (qwerty)
Penna-ottica
Schermo tattile (*touch screen*)
Tavoletta grafica

Periferiche di output verso operatore

Lampade o LED
Sirene e avvisatori acustici
Video a scansione (CRT) a cristalli liquidi (LCD) a plasma
Display a segmenti per cifre (7 segmenti) o alfanumerici (14 segmenti)
Stampanti

Tecniche di colloquio (input/output) con operatore

Menu con elenchi di scelte o comandi
Maschere con campi da compilare (ad es. parametri)
Sinottici che rappresentano schematicamente parti di impianti e relativo stato
Barre con lunghezza proporzionale a valori di grandezze
Curve (*trend*) che rappresentano l'andamento nel tempo di grandezze
Pagine di allarmi con visualizzazione dello stato e comandi di tacitazione

Pacchetti applicativi - LabView, FIX

Sono pacchetti software che girano su PC o su Workstation industriali, che facilitano la configurazione e personalizzazione delle elaborazioni di informazioni tipiche dei sistemi di automazione e della gestione dell'interfaccia con l'operatore.

1.4 REQUISITI DELL'ELABORAZIONE NELL'INFORMATICA INDUSTRIALE

I calcolatori per le applicazioni industriali hanno in comune con quelli che chiameremo brevemente gestionali molte caratteristiche di base quali la funzionalità digitale programmabile, spesso la componentistica (CPU, memorie, ecc:) e talora anche buona parte della struttura costruttiva. I diversi impieghi e, spesso, i diversi tipi di ambienti di lavoro pongono però particolari requisiti che hanno impatto sia sulle tecniche di sviluppo del software che sulle caratteristiche architetturali.
Nel seguito si propone una serie di sintetiche considerazioni sulle peculiarità delle applicazioni industriali, evidenziandone le differenze da quelle delle applicazioni gestionali, che sono supposte meglio note al lettore.

1.4.1 ESECUZIONE DEI PROGRAMMI APPLICATIVI

Distinguiamo tre diverse modalità di esecuzione dei programmi, caratterizzati da un crescente livello di interattività col mondo esterno da cui sono ricevuti i dati e a cui vengono forniti i risultati.

1.4.1.1 Esecuzione BATCH

Si tratta di un'esecuzione, tipica dei centri di calcolo, che possiamo chiamare "differita", in quanto non c'è una particolare correlazione tra i tempi di raccolta e predisposizione dei dati, l'elaborazione e l'utilizzo dei risultati.
I programmi leggono dai file i dati quando ne hanno bisogno e scrivono su file o su stampante i risultati, quando li hanno calcolati.
L'uso di risorse potenti e costose (*mainframe*, capaci memorie di massa, veloci stampanti, ecc.) suggeriscono l'adozione di sistemi operativi che ne ottimizzino lo sfruttamento, cercando quindi di massimizzare il *throughput* più che la rapidità dei tempi di risposta.

1.4.1.2 Esecuzione ON-LINE

In questa modalità di esecuzione possiamo raggruppare le elaborazioni dette "interattive", in cui l'operatore, che generalmente è anche l'utente dei risultati, colloquia direttamente con il sistema operativo per attivare l'esecuzione delle funzioni che egli ritiene man mano necessarie per lo svolgimento del suo lavoro e **interagisce con i programmi applicativi** per fornire loro parte o tutti i dati da elaborare.
Analoga è anche la modalità di elaborazione che gestisce i protocolli di colloquio tra calcolatori per effettuare comunicazioni e transazioni.
Questa modalità può essere detta **reattiva** perchè l'esecuzione dei programmi è in gran parte stimolata come risposta ad eventi esterni (i comandi dell'operatore), ma è caratterizzata dal fatto che entrambi gli interlocutori (operatore e calcolatore) sono **disponibili a sincronizzarsi** l'uno con l'altro adeguandosi ai rispettivi ritmi.
Chiaramente è privilegiata una buona rapidità di risposta, ma l'utente è forzato ad adeguarsi ai tempi di esecuzione del calcolatore.
Le configurazioni tipiche sono costituite dai calcolatori (o da *workstation*) personali, oppure da potenti *server* con accesso in *time-sharing* tramite terminali. In ogni caso l'aumento del carico di lavoro provoca un degrado delle prestazioni locali in termini di tempi di risposta.

1.4.1.3 Esecuzione REAL-TIME

Se il sistema di elaborazione deve interagire con delle macchine non può in genere imporre i propri ritmi di esecuzione, ma deve adeguarsi a ben precisi vincoli temporali.
Infatti nelle macchine hanno sede fenomeni governati da leggi fisiche che determinano la dinamica di evoluzione dei fenomeni stessi da cui dipende la cadenza con cui possono essere recepiti i dati e a cui occorre adeguarsi per effettuare efficaci interventi di controllo.
Questa modalità è di tipo **reattivo e in tempo reale**.
Nei sistemi di questo tipo, che sono l'oggetto specifico di attenzione di questo corso, l'obbiettivo centrale non è la massimizzazione del *throughput*, e neppure la minimizzazione dei tempi di risposta, ma piuttosto la **predicibilità del comportamento temporale** nelle interazioni col mondo esterno.
Su questi aspetti si torna nel par. 1.4.6.

1.4.2 INTEGRAZIONE HW/SW

L'interazione con macchine, ed in generale con fenomeni fisici esterni, impone requisiti particolari e molto variabili da applicazione ad applicazione anche alla funzionalità di base dei calcolatori industriali, che si manifesta con una notevole varietà di interfacce e con particolari necessità di integrazione tra le funzioni affidate all'HW e quelle realizzate a livello di SW. Contrariamente a quanto è possibile ed avviene per i Personal Computer (PC) non è facile individuare ed adottare delle configurazioni e soluzioni standard, che siano adatte per la maggior parte dei contesti applicativi, anzi è frequente la necessità che il progettista entri in dettagli che si trovano sul cammino critico di una certa funzionalità per trovare una soluzione specializzata adeguata ai requisiti del problema.
La barriera di virtualizzazione tipica, cioè il Sistema Operativo, in alcune applicazioni addirittura non esiste, mentre in altre deve essere superata con particolari interventi specifici a livello HW o SW.
Ulteriori considerazioni sulle componenti HW e SW e sul loro contributo alle prestazioni del sistema di automazione, sono presentate nel cap. 4.

1.4.2.1 Classi di configurazioni

Sistemi Embedded.

Sono detti calcolatori *embedded* (alla lettera "incastonati") quelli che fanno parte integrante di una piccola macchina, apparato o strumento la cui funzionalità è ottenibile solo dalla stretta simbiosi delle parti tra cui è spesso difficile tracciare confini di separazione. I calcolatori *embedded* sono generalmente progettati, a livello di SW ma anche di HW, nell'ambito del progetto complessivo dell'apparato. Molto spesso i calcolatori sono del tipo *single chip controller*. La frequente disponibilità a bordo di questi controllori di un'interfaccia per bus seriale suggerisce soluzioni "distribuite" anche per applicazioni relativamente semplici e di ridotte dimensioni.

Sistemi di controllo di macchine.

Si tratta di calcolatori che possono assumere configurazioni anche piuttosto diverse, da singole schede di circuito stampato a controllori programmabili (PLC) a calcolatori specifici per controllo numerico (CNC) a sitemi modulari, che sono associati ad una macchina per controllarne il comportamento, spesso frapponendosi tra il pannello operatore e la macchina vera e propria.. In alcuni casi si adotta una configurazione a due calcolatori (ad esempio PC+PLC) più o

meno strettamente connessi, tra cui sono ripartiti i compiti di gestione dell'operatore e controllo della macchina.
Si hanno generalmente alcune decine di ingressi e uscite digitali e alcuni ingressi e (poche) uscite analogiche.

Sistemi di automazione di impianti.

Nell'automazione degli impianti la grande varietà di situazioni comporta una corrispondentemente ampia varietà di soluzioni realizzative. Localmente si avranno apparati con sistemi *embedded* e macchine con un proprio controllore, mentre si utilizzeranno sistemi (microcalcolatori modulari e minicalcolatori) ai livelli superiori per raccolta dati, monitoraggio ed in generale automazione globale.
Sono frequenti le situazioni con reti di interconnessione complesse e molte centinaia di I/O digitali e analogici.

1.4.2.2 Interfacce specifiche

La necessità di scambiare informazioni relative ad una grande varietà di fenomeni fisici richiede una gamma molto ampia di interfacce. Non è raro il caso in cui per alcune applicazioni si debba addirittura progettare e realizzare appositamente delle interfacce con requisiti particolari e quindi non reperibili in commercio.
In particolare si hanno esigenze anche molto variabili in termini di:

velocità di trasferimento delle informazioni
precisione e stabilità delle misure
insensibilità ad agenti perturbanti
resistenza a condizioni ambientali avverse
robustezza, affidabilità e tolleranza ai guasti
sicurezza intrinseca (ambienti con pericolo di esplosioni)

Le problematiche di interfacciamento costituiscono uno degli aspetti centrali, e ad esse sono dedicati i capitoli 5, 6, 7 e 8.

1.4.2.3 CPU e memorie

Altri aspetti peculiari dei sistemi di elaborazione per il controllo riguardano i processori e le memorie.
I processori adatti per applicazioni di controllo privilegiano spesso l'esecuzione in tempi ben predicibili di semplici operazioni orientate ai bit e ai numeri interi puttosto che l'esecuzione di operazioni complesse a velocità *mediamente* elevate ma meno predicibili.
Le memorie operative dei calcolatori di controllo hanno spesso capacità relativamente ridotte e sono organizzate in aree distinte per funzionalità e tecnologia.

- ROM o EPROM non volatili e non modificabili per l'area destinata ai programmi di base e applicativi.
- RAM volatile e inizializzata a software all'accensione del sistema, per i dati dinamici dell'elaborazione.
- RAM in *Tampone*, non volatile ma modificabile, per i dati semifissi, i cosiddetti parametri, che vengono impostati in fase di messa a punto iniziale (*tuning*) dell'applicazione e occasionalmente modificati in esercizio.
- Aree mappate su porte di I/O (*memory mapped I/O*) in cui con le normali istruzioni di accesso a memoria in realtà si leggono e scrivono speciali registri e porte di interfaccia con l'esterno.

1.4.2.4 Strumenti informatici e SW di base

La difficoltà a standardizzare i calcolatori, e ancor più le interfacce, dei sistemi per applicazioni industriali crea particolari problemi anche a livello di SW di base. Questi strumenti SW (Sistemi Operativi, Compilatori, Linker, Debugger, ecc.) nelle applicazioni **informatiche gestionali** sono generalmente standardizzati e si è abituati a considerarli parte quasi integrante del sistema, sviluppati a carico del costruttore del sistema o da terze parti, e comunque adatti senza particolari modifiche ma solo con qualche personalizzazione a tutte le normali applicazioni.
Questi strumenti tendono in genere, e giustamente ai fini per cui sono stati sviluppati, a nascondere (*hide*) tutti quei dettagli a cui si ritiene che il progettista di applicazioni che ne è l'utente non sia interessato e la cui gestione appesantirebbe lo sviluppo delle applicazioni stesse.
In particolare sono colpiti da "*incapsulamento opaco*" molti aspetti che invece riguardano anche molto da vicino l'attenzione del progettista di **applicazioni industriali**, tra cui i principali sono:

- le strutture dati in memoria ed i loro meccanismi di accesso
- la gestione della memoria centrale, le allocazioni di programmi e dati
- la gestione dei meccanismi di interrupt e DMA
- la gestione delle interfacce
- la gestione delle comunicazioni
- le politiche di gestione dei processi
- le temporizzazioni.

Nelle applicazioni di automazione e di elaborazione in tempo reale in genere la possibilità di **controllo delle risorse**, o almeno la **visibilità** e conoscenza dei meccanismi della loro gestione, è condizione necessaria per una corretta implementazione. Come conseguenza il progettista di applicazioni industriali non trova purtroppo la stessa ricchezza di strumenti informatici adatti che caratterizza altre categorie di applicazioni, e spesso deve ricorrere alla realizzazione in proprio anche del software di base, che in queste applicazioni si fonde molto con quello applicativo, e all'uso del linguaggio Assembler.

1.4.2.5 Architetture costruttive

Anche gli aspetti costruttivi, ovviamente, devono tener conto delle particolari esigenze di questa categoria di applicazioni.
Ad esempio **l'accessibilità** ai morsetti delle interfacce di I/O che in alcuni calcolatori (ad es. i PC) è considerata secondaria ed occasionale, nei calcolatori industriali deve essere agevole e ben evidenziata. E' importante la facilità di ispezione delle varie parti e talora sono previsti indicatori luminosi a LED che visualizzano lo stato binario dei morsetti di I/O.
La **robustezza** meccanica ed elettrica sono caratteristiche spesso richieste in misura ben maggiore negli ambienti industriali di fabbrica, rispetto a quanto lo siano nei laboratori e, naturalmente, rispetto agli ambienti di uffici.
Un altro aspetto molto importante, soprattutto nei sistemi di automazione di impianti, è la facilità di **manutenzione** ed eventuale sostituzione di parti. Sono quindi preferibili sistemi modulari e contenitori ad inserzione frontale dei moduli, connettori di dimensioni generose e di facile inserimento con minima possibilità di errori di posizione o di inversione. I cablaggi saranno ordinati e chiaramente contrassegnati con etichette di riconoscimento.

1.4.2.6 Collaudo e messa a punto (Test e Debug)

Il collaudo e la messa a punto delle applicazioni indistriali è un altro aspetto che le differenzia in modo drastico rispetto a quelle gestionali.
Innanzitutto si tratta di un collaudo integrato di funzioni HW e SW, che spesso pone problemi non trascurabili nell'attribuzione della responsabilità di eventuali errori o malfunzionamenti.
Le difficoltà probabilmente più gravi derivano dal fatto che non è agevole, e spesso neppure possibile, fornire al sistema di elaborazione dati e stimoli che riproducano tutti i casi che potranno verificarsi durante il reale esercizio.
In ogni caso l'ambiente di collaudo, anche nelle fasi di prova "al banco" nei laboratori del costruttore, richiede tutta una serie di apparecchiature e strumenti elettronici ed altro, che rendono il contesto ben diverso da quello "asettico" delle prove di programmi per normali elaborazioni.

1.4.3 ELABORAZIONE IN TEMPO REALE

Con la locuzione "**sistemi *real-time***" si intendono macchine adatte all'interazione con fenomeni che evolvono nel tempo in modo non completamente controllabile.
Mentre ci si propone nel seguito di individuare una definizione un po' più caratterizzante, è importante evidenziare che ci si trova in un terreno probabilmente molto "battuto" ma ancora poco "dissodato".

Si è infatti promossa una notevole attività di sviluppo dell'informatica ad alto livello e con notevoli progressi metodologici, che però privilegia decisamente l'invisibilità dei fenomeni che interagiscono con il calcolatore, soprattutto negli aspetti temporali, ma anche negli aspetti circuitali.
Nelle applicazioni tipicamente gestionali infatti, si focalizza l'attenzione sulle informazioni, mentre sono considerati dettagli implementativi da nascondere all'utente, ma anche al programmatore, tutti i meccanismi legati alla funzionalità circuitale del calcolatore ed alle sue interazioni con altre apparecchiature.
Dall'altro lato sono state sviluppate una miriade di applicazioni che possono fregiarsi della qualifica "real- time", spesso realizzate ed ancor più messe a punto in modo empirico, da persone più esperte dell'applicazione che degli strumenti informatici
Questa divaricazione (*gap*) ha probabilmente avuto il suo massimo a cavallo tra gli anno 70 e 80, e solo in tempi relativamente recenti mostra cenni di riduzione.
Si ritiene quindi importante che venga svolta un'attività di studio e sistematizzazione di concetti, formalismi e strumenti che utilizzi i risultati dell'informatica classica, ma che ne perfezioni gli aspetti legati al concetto di tempo e di interazione tra HW e SW e tra calcolatore e fenomeni esterni, sulla base di esigenze che emergono da varie classi di applicazioni.

Si tratta di un'attività con forti connotazioni **interdisciplinari**, non solo nel dualismo HW/SW, ma anche nella varietà di problemi logici, matematici, fisici e, perchè no, psicologici che ruotano attorno alle realizzazioni ingegneristiche di una certa complessità. In particolare gli aspetti umani, sia riguardo l'uomo progettista, che l'uomo utente, assumono un'importanza sempre crescente e pongono problemi talora non facilmente solubili con considerazioni puramente tecniche.

Naturalmente esistono nel mondo numerose sedi di attività di studio e realizzativa sugli aspetti citati, ma, soprattutto negli ambienti industriali, si nota ancora una scarsa convergenza verso l'unificazione e la divulgazione dei risultati, che anzi spesso sono considerati una proprietà da non condividere con altri concorrenti industriali.

E' importante notare che tutti i problemi presentano esigenze di tempo reale, più o meno stretto, anche quelli che possiamo classificare come "gestionali" in contrapposizione a quelli di "controllo": sono questi ultimi che ci interessano in modo particolare, dato che i primi possono ricondursi a questi (controllo di dischi, periferiche, ecc.) ma non viceversa.

Prima di procedere è importante precisare qual'è la visione dell'**elaborazione in tempo reale** che permea una buona parte del corso.

- I principali **elementi controllanti** sono costituiti da calcolatori elettronici digitali binari programmabili, sia di tipo *general purpose* che specializzati.

- Gli **elementi controllati** costituiscono, rispetto al calcolatore, un sottoinsieme del cosiddetto "mondo esterno" e sono generalmente caratterizzati dalle seguenti proprietà:

- Interagiscono col controllore fornendogli stimoli e ricevendone risposte.
- Dai comandi ricevuti (siano essi generati dai controllori autonomamente o come risposte a stimoli) dipende, talora anche in modo decisivo, la loro corretta evoluzione.
- Sono disomogenei dal punto di vista fisico e temporale rispetto al controllore, ed in particolare la loro evoluzione dipende da fenomeni fisici.
- Non sono sempre modellizzabili con accuratezza, per gli scopi perseguiti, soprattutto per quanto riguarda le situazioni anomale, cioè non sono abbastanza "prevedibili".
- Presentano tra loro un elevato livello di scorrelazione.

Gli elementi controllati costituiscono cioè dei generatori di eventi in istanti non prevedibili.
I sistemi in tempo reale caratterizzati come sono dalla correlazione tra gli stimoli esterni (uno dei quali è anche il fluire del tempo) e le loro azioni, sono ben rappresentabili come "*event driven*" (guidati dagli eventi), o anche, considerando che gli eventi sono in sostanza l'apparire di informazioni, "*data driven*". Alcuni esperti del settore preferiscono però adottare meccanismi implementativi cadenzati dal fluire del tempo (*time driven*). Questi aspetti sono approfonditi nel cap.10.

Le difficoltà che si incontrano nei sistemi in tempo reale sono di vario tipo: ne facciamo qui un breve cenno, rimandandone l'analisi a sedi più opportune.

- Difficoltà della mente umana a dominare il parallelismo, cioè situazioni che si sviluppano simultaneamente lungo dimensioni spaziali e temporali.
- Difficoltà ad individuare e ad utilizzare strumenti descrittivi del parallelismo, che siano sufficientemente semplici ma che evidenzino in modo completo e rigoroso le relazioni tra le varie entità concorrenti.
- Sequenzialità e discontinuità dei calcolatori programmabili che impongono di mappare su un unico asse temporale elaborazioni relative anche a fenomeni concomitanti e distribuiti che debbano essere trattati da un unico processore.
- Difficoltà nel valutare gli effettivi tempi di latenza e di esecuzione delle elaborazioni nell'ambito di un SW complesso e stratificato in livelli applicativi e livelli di base (Librerie, Sistema Operativo, Compilatori, ecc.)
- Problemi legati al non determinismo, cioè al fatto che alcune relazioni sono note solo in termini probabilistici. Non è possibile riprodurre esattamente la stessa sequenza di eventi, sia pure legati tra loro. Ciò in particolare pone gravi difficoltà per il *debug* e la verifica "in vivo" dei sistemi in tempo reale.

1.4.3.1 DEFINIZIONI DI TEMPO REALE (REAL-TIME)

In letteratura si trovano diverse definizioni di sistemi di elaborazione in tempo reale che in genere si differenziano solo per una diversa enfasi data alla capacità di garantire le prestazioni temporali richieste dalle specifiche.

a) -Da Oxford Dictionary of Computing:

Real-time system

> *Any system in which the time at which output is produced is significant. This is usually because the input corresponds to some movement in the physical world, and the output has to relate to that same movement. The lag from input time to output time must be sufficiently small for acceptable timeliness.*

b) - Young (1982) definisce sistema real-time:

> *Any information processing activity or system which has to respond to externally-generated input stimuli within a finite and specified period.*

c) - Altra definizione proposta.

> *Un sistema di elaborazione è detto in tempo reale se è richiesto che esso acquisisca dati, emetta dati ed interagisca con il suo ambiente circostante in istanti temporali precisati e generalmente correlati con eventi che in tale ambiente si verificano.*

Tutti concordano nell'affermare che:

> **la correttezza di un sistema in tempo reale dipende non solo dal valore dei risultati forniti (correttezza *logica*), ma anche dall'istante temporale in cui essi sono prodotti (correttezza *temporale*).**

Sono ancora oggetto di studio le tecniche per specificare le proprietà temporali da rispettare e per quantificare l'accettabilità degli scostamenti dal comportamento ideale.

Osservazioni sulle definizioni di tempo reale:

- Innanzittutto va osservato che la maggior parte dei sistemi di elaborazione, e non solo quelli di “controllo”, ricade nelle definizioni più generali; infatti per ogni problema di calcolo esiste una scadenza oltre la quale i risultati non sono più utili e la loro mancata produzione causa un certo danno. Le successive definizioni d) e e) tentano in parte di escludere i casi “banali”.
- In genere il comportamento temporale è specificato per il sistema **complessivo Processo + Controllore**. Da queste specifiche si devono poi dedurre quelle relative al comportamento del "**sottosistema di elaborazione**", e non sempre tale deduzione è semplice, univoca e realizzabile.

Ad esempio una specifica potrebbe richiedere che una parte in movimento si fermi "entro x metri", ponendo così *indirettamente* vincoli temporali sul comando degli organi frenanti, ma anche sulla loro energicità, ecc. In questo caso possono essere accettabili frenate sollecite (aspetto temporale) con una certa forza (aspetto quantitativo) oppure frenate ritardate ma effettuate con maggior forza. E’ evidente che un vincolo sulla forza frenante massima applicabile, o accettabile, pone anche un vincolo sul termine dell’intervallo temporale entro cui deve iniziare la frenata.

- Oltre a specifiche relative a **ritardi** e **scadenze**, sono necessarie generalmente anche specifiche di **frequenze** e **regolarità** temporali.

- Come in tutte le discipline ingegneristiche vanno previsti vari tipi di **perturbazioni** dovute a:
  - approssimazione dei modelli
  - imprecisioni costruttive
  - possibilità di malfunzionamenti.

- E' importante caratterizzare in qualche modo la "**criticità**" dei vari comportamenti temporali.
Generalmente si parla di "**hard real-time**" quando il mancato rispetto delle scadenze temporali è considerato un "**fallimento**" (alla stregua di un guasto) del sistema, che può comportare danni anche gravissimi a persone e/o cose.
Viene detto "**soft real-time**" una funzionalità per cui il mancato rispetto delle scadenze temporali costituisce un "**degrado** di prestazioni", con danni occasionalmente accettabili.
I sistemi di elaborazione in tempo reale avranno quindi un **sottoinsieme critico** di azioni che *devono* essere completate prima della loro scadenza, ed un sottoinsieme di azioni per cui è *desiderabile* il completamento prima della scadenza.

Può essere interessante (e forse un po' provocatoria) una definizione che distingue i sistemi ("banali") che non presentano particolari difficoltà a rispettare vincoli temporali da quelli (*real-time*) che invece richiedono tecniche di progetto ed implementative per ottenere e garantire la correttezza temporale:

d) -

> *Un sistema è classificabile come "sistema di elaborazione in tempo reale" se sono state assegnate specifiche di comportamento temporale e la sua realizzazione pone problemi per il rispetto di tali specifiche.*

Questa definizione è apparentemente tautologica e un pò vaga, ma operativa. Ovviamente va completata con la precisazione di quali specifiche temporali pongono problemi ad essere rispettate, e l'entità di tali problemi.

Una regola sintetica e pratica ("*thumb rule*") per individuare quali prestazioni temporali pongono dei problemi, potrebbe essere contenuta nella definizione seguente:

e) -

> *Un sistema è classificabile come "sistema di elaborazione in tempo reale" se le granularità dei tempi specificati per le risposte sono comparabili con quelle dei tempi di esecuzione delle funzioni necessarie per fornire tali risposte. (Cioè se occorre parallelismo tra le diverse attività.)*

Questa definizione traccia un confine sfumato e fortemente legato alla "potenza di calcolo" degli elaboratori considerati e agli algoritmi adottati.

Un aspetto trascurato dalle precedenti definizioni, ma che assume generalmente una notevole importanza, riguarda l'affidabilità del sistema applicativo, che in buona misura dipende dall'affidabilità del sottosistema di elaborazione.

- affidabilità HW
- correttezza SW
- robustezza SW (evitare "abort" causati da errori)
- elevata probabilità del rispetto di specifiche valore/tempo

Viene considerato **idealmente corretto** rispetto al comportamento in tempo reale un sistema che ricevendo stimoli con determinate caratteristiche è in grado di fornire **sempre** risposte **corrette** entro le **scadenze** temporali specificate.

Consideriamo (ingegneristicamente) **corretto**, e quindi degno della qualifica di "sistema di controllo in tempo reale" un sistema in cui ogni attività fornisca risultati entro le

tolleranze di **valori** e **temporali** specificate per quell'attività, e fallisca solo per una percentuale di volte inferiore ad una soglia di tolleranza precisata e specifica per quell'attività.

Specificando ad esempio una soglia di accettabilità di errori temporali dell'1%, un'attività è svolta correttamente se i suoi ritardi superano la scadenza prevista meno dell'1% delle volte. Naturalmente le funzioni di tipo *hard real-time* comportano soglie di accettabilità praticamente dello 0%, mentre le funzioni *soft real-time* avranno soglie superiori.

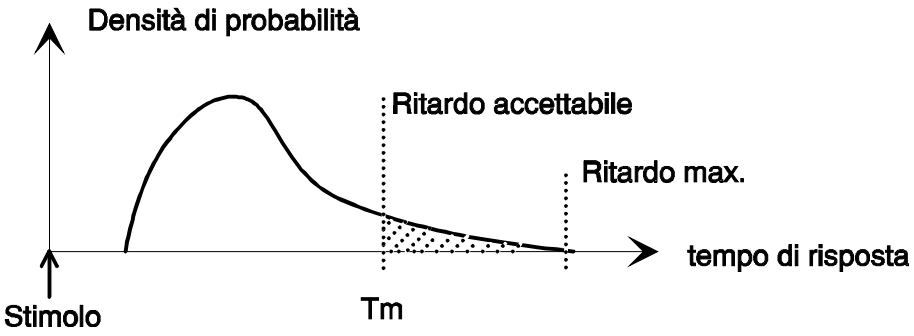


Fig. 1.7 - Curva di densità di probabilità che la risposta ad uno stimolo avvenga dopo un tempo di risposta. Tm è il ritardo massimo accettabile.
Il sistema è "corretto" se l'integrale di tale curva tra il tempo Tm e infinito è inferiore al valore di specifica.

Le motivazioni di questa posizione sono le stesse che guidano le problematiche sul riconoscimento e la correzione di errori in sistemi di elaborazione e trasmissione di informazioni, basate su ovvie considerazioni di fisica realizzabilità.
Come in tutte le realizzazioni ingegneristiche si cerca infatti di specificare la soglia in modo che il **costo associato all'accettazione degli errori sia inferiore al costo aggiuntivo necessario per eliminarli.**
Il costo degli errori si manifesta in termini di degrado della qualità, di maggior frequenza dei guasti e di maggiore probabilità e gravità di danni e catastrofi.

La proprietà che caratterizza un sistema RT (real-time) è quindi la correttezza temporale sovrapposta alla correttezza logica, e non l'efficiente sfruttamento delle risorse. Anzi spesso si ottiene la correttezza sottosfruttando sensibilmente un sistema in modo da averlo "*idle*", quindi disponibile, per la maggior parte del tempo.

Per ottenere la correttezza nel comportamento temporale è necessaria la predicibilità degli intervalli temporali in cui le varie azioni saranno terminate; ciò richiede che siano noti i limiti inferiori (*lower bound*) e soprattutto, perchè spesso sono i più critici, i limiti superiori (*upper bound*) dei **tempi di esecuzione** di tutte le azioni.
E' auspicabile l'uso di tecniche analitiche automatiche per garantire in anticipo che il **sottoinsieme critico** del sistema è **temporalmente corretto**, in modo da poter adottare provvedimenti correttivi, analogamente a quanto si fa per la correttezza logica.
Aspetto centrale nei sistemi real-time è il problema dello ***scheduling***, cioè del criterio di allocazione spaziale (tra diverse CPU e risorse) e temporale (alla stessa CPU) delle azioni da eseguire, ovviamente rispettando eventuali vincoli di precedenza e mutua

esclusione tra le diverse azioni, in modo che tali azioni soddisfino alle specifiche temporali.
Si noti che per una corretta progettazione e realizzazione di un sistema di elaborazione in tempo reale sono necessari:

- **criteri di verifica** che un sottoinsieme critico di azioni sia **schedulabile** (*feasible*)
- **algoritmi di scheduling** delle azioni appartenenti ad un sottoinsieme *feasible*.

E' molto importante notare che nessuna definizione di tempo reale fa riferimento a valori numerici assoluti di durate temporali: si possono avere attività in tempo reale che hanno tempi di risposta di 10 minuti, corretti in base alle loro specifiche, mentre altre attività che rispondono entro pochi millisecondi non possono essere considerate in tempo reale se le loro specifiche richiedono tempi di ritardo inferiori.
Infatti per qualunque tipo di fenomeno esiste una risoluzione temporale significativa, che potremo chiamare **risoluzione temporale caratteristica**, al di sotto della quale le differenze temporali possono essere considerate irrilevanti. Ciò consente di definire l'equivalenza tra comportamenti temporali le cui differenze cadono all'interno degli intervalli di indistinguibilità, corrispondenti alla risoluzione temporale.
Analogamente si ha una risoluzione caratteristica anche per ogni valore numerico, ad esempio di grandezze fisiche.

1.4.4 AFFIDABILITA'

Esiste poi l'importante aspetto dell'affidabilità, che assume caratterizzazioni anche molto diverse tra i sistemi di elaborazione di informazioni e i sistemi di automazione.
Mentre la **robustezza**, cioè la capacità di assorbire situazioni anomale preservando l'integrità del sistema (evitare "*crash*"), è una caratteristica importante per tutte le applicazioni di elaborazione, un requisito specifico dei sistemi RT è quello che potremmo chiamare **vitalità**, cioè la capacità di continuare l'esecuzione anche in presenza di anomalie (evitare "*abort*").

Quando un impianto o una macchina sono controllati da un calcolatore, tra le possibili cause di malfunzionamento va ovviamente annoverato il calcolatore stesso.

- I malfunzionamenti dovuti ad **elaborazioni non corrette** (nel valore o nel tempo di risposta) sono stati trattati nei punti precedenti e costituiscono errori di progetto, che eventualmente saranno evidenziati solo in situazioni critiche.
- Altri malfunzionamenti possono essere causati da **guasti del calcolatore**, e di questi ci occupiamo ora brevemente.

Si noti comunque che errori SW e malfunzionamenti HW all'esterno possono presentarsi in modo simile, così da essere talvolta difficili da distinguere.
I guasti dei calcolatori possono manifestarsi con la produzione di dati errati o con la cessazione della produzione di dati.
Chiamiamo **integrità** la proprietà di un calcolatore di non produrre dati errati, neppure in caso di guasto, eventualmente ricorrendo all'autospegnimento. In termini quantitativi la misura dell'integrità potrà essere data dalla percentuale dei dati correttamente calcolati, rispetto al totale.
Chiamiamo **persistenza** la proprietà di un calcolatore di fornire un servizio continuo che, in caso di guasto, può portare alla produzione di dati errati o a brevi arresti per una riconfigurazione. In termini quantitativi la persistenza è data dalla percentuale del tempo di funzionamento, rispetto al totale.

Quando l'effetto di guasti del calcolatore può comportare danni inaccettabili sull'impianto controllato, si ricorre a calcolatori affidabili (*dependable*) al posto dei normali calcolatori standard. I calcolatori affidabili presentano un comportamento predicibile in caso di un loro guasto, e per ottenere ciò contengono un certo grado di **ridondanza** strategicamente organizzata in base al tipo di comportamento desiderabile.
Si possono distinguere diverse classi di affidabilità che elenchiamo in ordine crescente di complessità, e quindi anche di costi:

- Calcolatori integri
- Calcolatori persistenti
- Calcolatori persistenti e integri, che assommano le doti di integrità e persistenza.
- Calcolatori altamente affidabili che **mascherano** completamente, rispetto all'impianto, l'effetto di ogni guasto.

Il grado di affidabilità da adottare ed i requisiti di comportamento in caso di guasto dipendono dalle caratteristiche del particolare impianto, che meritano quindi una pur breve analisi.
Per caratterizzare un impianto rispetto ai malfunzionamenti del suo controllore l'aspetto più importante è per quanto tempo l'impianto può tollerare senza danni tali malfunzionamenti, cioè il suo **tempo di tolleranza** (*grace time*). Si noti che ogni impianto presenta tempi di tolleranza diversi rispetto alla mancata integrità e alla mancata persistenza.
Il principale criterio discriminante è basato sull'analisi della **reversibilità** dell'impianto, cioè la sua possibilità di uscire da una situazione anomala e recuperare il comportamento normale.

Gli impianti **continui**, con la loro dinamica che li rende spesso modellabili come filtri passa-basso, sono in genere reversibili fintanto che il loro comportamento rimane nella fascia di sicurezza. Questi impianti possono quindi presentare un tempo di tolleranza anche relativamente lungo rispetto a dati errati (mancata integrità), mentre la cessazione del controllo (mancata persistenza) può essere tollerata indefinitamente solo se l'impianto ha un comportamento che lo porta spontaneamente in uno stato sicuro (*safe*).

Gli impianti a funzionamento **discreto** sono spesso caratterizzati da fasi a bassissima reversibilità: basti pensare a linee di assemblaggio, scaricamento del contenuto di silos su veicoli, ecc.
Per questo tipo di impianti prevale in genere l'esigenza di integrità, rispetto a cui i tempi di tolleranza sono anche molto brevi (pochi ms), mentre la mancanza di controllo porta solitamente ad un arresto sicuro.

In conclusione gli **obbiettivi di un controllore affidabile** sono:

- prevenire danneggiamenti dell'impianto (*safety*)
- ripristinare il funzionamento normale il più presto possibile (*availability*)
- massimizzare la disponibilità nel lungo periodo.

Come si è accennato le caratteristiche di sicurezza (*safety*) e disponibilità (*availability*) di un sistema automatizzato non dipendono solo dalle proprietà del calcolatore di controllo, ma soprattutto dal modo con cui queste proprietà sono "sintonizzate" sui gradi di tolleranza del particolare impianto controllato.

1.5 BIBLIOGRAFIA

E.G.R Gerelle, S. Stark
INTEGRATED MANUFACTURING - STRATEGY, PLANNING AND IMPLEMENTATION.
Ed. McGraw-Hill 1988

U. Rembold, C. Blume, R. Dillmann
COMPUTER INTEGRATED MANUFACTURING. TECHNOLOGY AND SYSTEMS.
Ed. Marcel Dekker 1985

Auslander D.M., Than C.H.
REAL-TIME SOFTWARE FOR CONTROL
Ed. Prentice-Hall
Bibliot.
Interessanti tecniche SW con esempi in C.
Poco formale e speculativo.

Malcolm Adamson
SMALL REAL-TIME SYSTEM DESIGN
Sigma Press (UK) 1990
Bibliot. AR-273
Piccoli sistemi, applicativo.
Interessante, con molte idee e considerazioni pratiche.
Uso di timer e counter.

NACHTIGAL C.L.
INSTRUMENTATION & CONTROL
John Wiley & Sons Inc. 1990
Bib. Dip. $ 120. 880 pagg.
Impostazione tecnico-industriale molto buono.

OLSSON G., PIANI G.
COMPUTER SYSTEMS FOR AUTOMATION AND CONTROL
Prentice-Hall 1992
Bib. Dip. L.88000 420 pagg.
Interessante. Cenni teoria controlli a campionamento.
Copre tutti gli argomenti ma un pò sinteticamente.

Julio Sanchez
ASSEMBLY LANGUAGE TOOLS AND TECHNIQUES FOR THE IBM MICROCOMPUTERS
Prentice Hall 1990

Karl J. Astrom, Bjorn Wittenmark
COMPUTER CONTROLLED SYSTEMS: THEORY AND DESIGN
Prentice-Hall 1984

John Billingsley
CONTROLLING WITH COMPUTERS
Controltheory and practical digital systems
McGraw-Hill 1989

Henri Nussbaumer
INFORMATIQUE INDUSTRIELLE
Vol. I Representation et traitement de l'information
Vol. II Introduction a l'informatique du temps reel
Vol. III Automates programmables. Commande et reglage. Capteurs.
Vol. IV Commande numerique. Teleinformatique. Securete, surete, fiabilite
Presses Polytechniques Romandes 1987

Gianguido Piani
MULTIPROGRAMMAZIONE E SISTEMI DIGITALI DI CONTROLLO
Con esempi in ambiente MS-DOS
Zanichelli 1988

H. Kopetz, M.G. Rodd
Distributed Computer Control Systems
Proc. 10th IFAC Wks. Sept 91 - Austria
Pergamon Press 1992
Bibl.CC-297
Articoli interessanti su Real-time, Fieldbus, Data-flow

H. W. Lawson
Parallel Processing in Industrial Real-Time Applications
Prentice Hall 1992
Molto interessante, con considerazioni formali, architetturali e di progetto.
Presuppone già una certa competenza.