Cap. 14

14. SISTEMI DISTRIBUITI

L'elaborazione distribuita, che prevede cioè l'attribuzione di diversi compiti elaborativi a diversi processori, è un'evoluzione relativamente recente delle architetture informatiche: nei primi anni ruggenti dei cosiddetti "cervelloni", infatti, si prefiguravano scenari in cui estese aree di attività avrebbero fatto capo ad un unico potente centro di elaborazione che si sarebbe fatto carico, possibilmente con un elevato livello di coordinamento, di tutte le funzioni di gestione e controllo.

Le tecnologie del tempo e le considerazioni di efficienza, gestione e manutenzione ad esse correlate portavano, non a torto, a ritenere la soluzione centralizzata la più praticabile, e di tale soluzione sono quindi permeati numerosi romanzi di fantascienza dell'epoca.

Con l'avvento dei microprocessori la capacità di elaborazione ha raggiunto livelli di economicità, compattezza e affidabilità un tempo impensabili e che hanno drasticamente modificato l'approccio all'elaborazione di informazioni nella maggior parte dei campi applicativi, promuovendo una crescente distribuzione delle risorse di calcolo.

In particolare nelle applicazioni industriali e di controllo tale distribuzione assume motivazioni e quindi connotazioni particolarmente spinte, come verrà sinteticamente presentato tra breve.

14.1 INTRODUZIONE AI SISTEMI DIGITALI DISTRIBUITI

14.1.1 GENERALITÀ

Una naturale conseguenza della distribuzione delle funzioni di elaborazione è la crescente necessità di interconnessioni che consentano la comunicazione, il più possibile “diretta”, tra i processori che in qualche modo siano coinvolti in elaborazioni tra loro correlate così da costituire un “sistema”.
Le tecniche che rendono disponibili le informazioni prodotte da un processore ad altri processori possono essere classificate nelle due seguenti categorie:
1) - Connessioni strette (*tightly coupled processors*)
2) - Connessioni lasche (*loosely coupled processors*)

Le connessioni strette sono tipicamente basate su meccanismi che consentono a diversi processori l'accesso ad aree di **memoria condivisa** (*shared*) a cui ognuno accede direttamente, salvo eventuali ritardi per l'arbitraggio di conflitti, come se si trattasse di una sua normale memoria.

Questo approccio, caratteristico dei sistemi multiprocessore, riguarda quella che possiamo considerare architettura interna dei calcolatori e traspare ben poco, se non nelle prestazioni generalmente elevate, ad un osservatore esterno e non verrà in questa sede ulteriormente approfondito.

Le connessioni lasche prevedono invece dei mezzi trasmissivi che collegano tra loro calcolatori che si scambiano informazioni mediante **operazioni esplicite di trasmissione** e rispettivamente di ricezione delle informazioni stesse, che generalmente vengono confezionate in messaggi.

Questo è l'approccio tipico dei sistemi effettivamente distribuiti e composti di macchine anche estremamente eterogenee, ed è quello a cui è rivolto il seguito di questo capitolo.

14.1.2 MOTIVAZIONI PER L'ELABORAZIONE DISTRIBUITA

Vi sono diverse motivazioni che suggeriscono la distribuzione di risorse attive di elaborazione, che assumono un diverso peso al variare del contesto applicativo e che quindi suggeriscono diverse scelte architetturali e topologiche. L'ordine con cui vengono presentate nel seguito non riflette quindi una graduatoria di inportanza con validità generale.

È comunque importante notare che la risposta alle esigenze sotto riportate non è sempre un beneficio “automatico” derivante da una soluzione distribuita, le cui promesse non sono ancora completamente realizzate a costi contenuti ed in forma standardizzata, ma costituiscono una tendenza tuttora in fase di sviluppo.

14.1.2.1 Costi dei cablaggi.

Soprattutto nelle applicazioni di monitoraggio e controllo di macchine ed impianti sono numerose le necessità di connessioni per i segnali che convogliano informazioni e comandi sulle grandezze fisiche in gioco.
I cablaggi costituiscono un costo, in taluni casi anche estremamente pesante, sia in termini di materiali (cavi più o meno speciali, canaline e morsettiere) sia per la loro installazione e manutenzione. Spesso la cadenza con cui le informazioni devono essere trasferite consente una loro **distribuzione nel tempo** su un unico cavo invece della distribuzione nello spazio su diversi cavi.
In questi casi un dispositivo di elaborazione che funga da concentratore può consentire anche notevoli risparmi sui cablaggi.

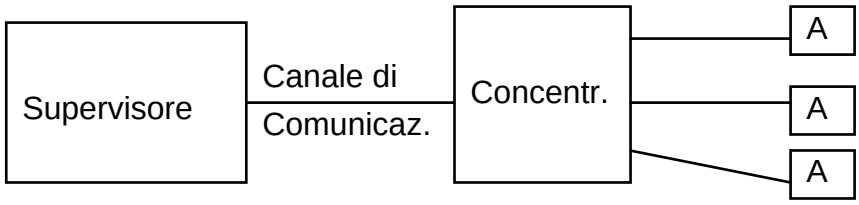


Fig. 14.1 Supervisore - Concentratore - Acquisitori.

Un esempio ormai diffusissimo è costituito da vari tipi di veicoli (automobili, aerei, navi) in cui la soluzione a microprocessori interconnessi è praticamente irrinunciabile.

14.1.2.2 Modularità.

La modularità costituisce un approccio estremamente interessante in molti e svariati settori applicativi per i quali siano richieste doti di estendibilità, riconfigurabilità, facilità di manutenzione, rapidità ed economicità nella riparazione di guasti.
Aspetti interessanti, a livello di mercato, sono poi anche la possibilità di utilizzo di moduli eguali in applicazioni diverse e di impiego in uno stesso sistema di moduli di diversi produttori, purchè ovviamente aderenti agli stessi standard ufficiali o di fatto.
La modularità è tipicamente basata su una struttura composta di parti facilmente collegabili tra loro in modo da formare un sistema integrato.
In termini strettamente inerenti alla quantità di dispositivi necessari la modularità comporta, rispetto alle soluzioni “monolitiche”, costi maggiori, che però sono spesso largamente compensati dai benefici globali.

14.1.2.3 Efficienza.

Per efficienza si intende qui lo sfruttamento ottimale delle risorse, cioè la collocazione del processore adatto alle funzioni da svolgere (specializzazione) nel luogo adatto a tali funzioni (distribuzione).

14.1.2.4 Tolleranza ai guasti.

La tolleranza ai guasti è la caratteristica dei sistemi in grado di svolgere in modo accettabile le loro funzioni, eventualmente con un degrado delle prestazioni, anche in presenza di uno o più guasti. In alcuni casi possiamo considerare, sia pure un pò impropriamente, tollerante ai guasti un sistema riparabile abbastanza facilmente e rapidamente da non provocare danni gravi durante il periodo di malfunzionamento.

In generale le tecniche per realizzare sistemi tolleranti ai guasti sono basate su forme di "**ridondanza**", cioè uso di entità in sovrannumero rispetto allo stretto necessario, e di "**confinamento degli effetti**" ottenuto conferendo la massima autonomia alle varie entità.
È facile vedere come un sistema distribuito, basato su moduli ad alto livello di autosufficienza e connessi tra loro lascamente sia particolarmente adatto a realizzare i requisiti citati e si possa anche prestare ad eventuali ridistribuzioni dinamiche dei compiti in caso di guasti.

14.1.3 TECNOLOGIE E CONDIZIONI ABILITANTI

La maggior parte delle potenzialità sopra citate si traduce in vantaggi effettivi solo se le tecnologie consentono una realizzazione di elementi con le adeguate caratteristiche di economicità ed affidabilità e se sono specificati e diffusi opportuni standard di interconnessione tra tali elementi.

Per quanto riguarda le tecnologie, dall'inizio degli anni '70 si è assistito ad una quasi travolgente evoluzione nel progetto e realizzazione di circuiti a sempre più elevati **livelli di integrazione**, soprattutto nell'ambito dell'elaborazione digitale, che hanno reso realizzabili dispositivi di notevole flessibilità e potenzialità a costi estremamente ridotti.
Un ruolo fortemente trainante in questo senso è stato, ed è tuttora, sostenuto dai **microprocessori** che hanno scatenato una notevole rivoluzione tecnologica in svariati settori ed in particolare in quelli della strumentazione e del controllo. Sono così nati i dispositivi (sensori, strumenti, attuatori, controllori, ecc.) gestiti con funzionalità "programmata" invece che "cablata".
Per distinguere tali dispositivi da quelli "tradizionali" si dice spesso che essi sono "computerizzati" (termine non molto bello) o "intelligenti" (termine alquanto improprio).
È interessante comunque notare che una comunicazione che sia un minimo articolata richiede generalmente una gestione con microprocessore, che possiamo quindi considerare il principale elemento abilitante delle tecniche attuali dei sistemi distribuiti.

Sul fronte delle **standardizzazioni** il processo è stato, ed è tuttora, più tormentato e spesso poco tempestivo, lasciando così adito alla ricerca di soluzioni proprietarie (e quindi incompatibili tra loro) da parte dei costruttori più intraprendenti.
In particolare questo processo è più complesso nell'ambito delle applicazioni industriali che in quelle di tipo gestionale, per vari motivi legati al mercato e alla maggior varietà di requisiti.
Comunque molto è stato fatto anche su questo fronte e alcune soluzioni si stanno ormai avviando alla maturità tecnologica e di mercato che consente un loro adeguato impiego in applicazioni industriali anche complesse.

14.1.4 ARCHITETTURE GERARCHICHE

La naturale organizzazione gerarchica delle funzioni di controllo coordinamento e gestione delle attività finalizzate alla trasformazione di energia o materia, ben si “mappa” su un'architettura gerarchica che connette diverse macchine, dispositivi e strumenti secondo le buone regole della modularità, che vogliono massimizzati coesione e disaccoppiamento.

Nelle strutture gerarchiche si possono in genere individuare dei “livelli”, cioè delle fasce che raggruppano elementi relativamente omogenei tra loro e piuttosto disomogenei rispetto a quelli dei livelli inferiore e superiore.

In una struttura rigorosamente gerarchica gli elementi di ogni livello non sono interconnessi tra loro, ma solo con quelli del livello superiore, da cui sono coordinati, e con quelli del livello inferiore, per i quali agiscono da coordinatori.
Ovviamente in molti casi è preferibile un minimo di coordinamento intra-livello, ottenibile con relative connessioni, senza dipendere completamente dal livello superiore, rinunciando in parte al rigore gerarchico.

Un approccio totale all'integrazione delle funzioni di produzione basate su calcolatori è proposto dal CIM (Computer Integrated Manufacturing), che prevede un continuo di supporti di elaborazione e di ausilio alle attività umane, dai livelli decisionali e organizzativi giù fino al controllo diretto delle macchine.

Una tipica gerarchia di sistemi di automazione distribuita è rappresentata nel modello di fig.14.2.

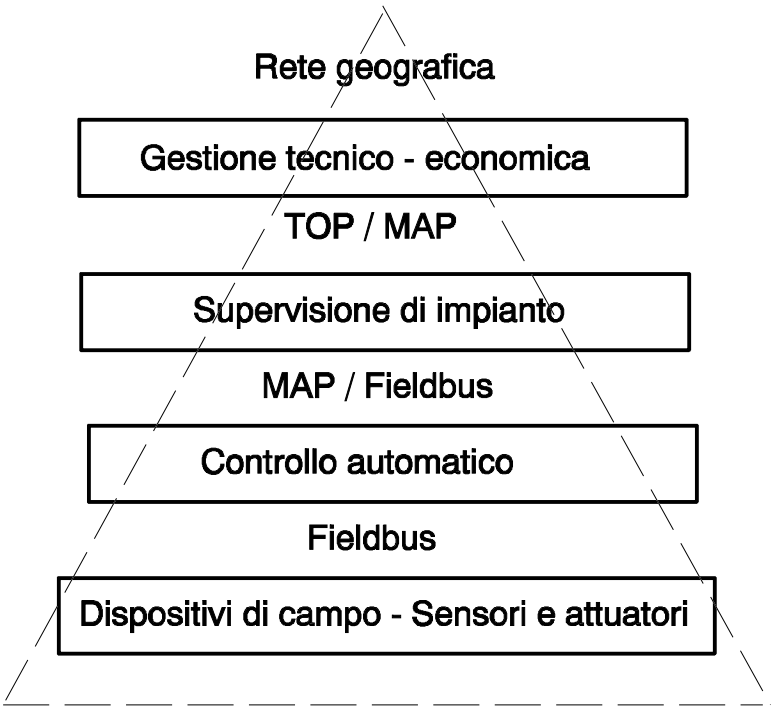


Fig. 14.2 - Modello di gerarchia nell’automazione distribuita

14.2 PROBLEMATICHE GENERALI DELLE COMUNICAZIONI

14.2.1 RAPPORTI TRA ENTITÀ COMUNICANTI

I rapporti tra due entità che interagiscono scambiandosi informazioni possono essere analizzati focalizzando l'attenzione su diversi punti di vista. Si noti che la corretta gestione di questi rapporti richiede l'uso di meccanismi che assumono aspetti diversi se le comunicazioni avvengono con una vera e propria trasmissione (sistemi a connessione lasca). In particolare devono essere considerati, perchè non sono più trascurabili, i **tempi** di comunicazione e gli **errori**.

CLIENT - SERVER
E' il rapporto che si riferisce ai **servizi**, cioè alle funzionalità ad alto livello, che l'entità cliente richiede e l'entità servitore fornisce. In genere il servitore è dotato di risorse particolari, come archivi di informazioni, programmi di elaborazione e processori, che risulta conveniente concentrare in opportuni siti ma che devono essere in qualche modo condivise.
Non tutte le comunicazioni richiedono che si configuri un esplicito e stabile rapporto client - server.

WRITER - READERS
E' il rapporto che evidenzia la **provenienza** di informazioni, considerate come **stati**, scambiate **senza sincronizzazione**.
Questo rapporto è generalmente instaurato per realizzare una memoria condivisa, eventualmente simulata se i processi risiedono su diversi processori.

PRODUCER - CONSUMER
E' il rapporto che evidenzia la direzione del **flusso** di informazioni principali, considerate come **eventi**, che l'ente produttore genera e l'ente consumatore utilizza coordinandosi con opportuni meccanismi di **sincronizzazione**.
È importante notare che spesso si hanno delle "catene" di produttori e consumatori in cascata, nelle quali i meccanismi di sincronizzazione assumono forme più articolate.

MASTER - SLAVE
E' il rapporto che riguarda i meccanismi di gestione delle comunicazioni a basso livello, dove le **iniziative** sono prerogativa dell'entità gestore (*master*), mentre l'ente asservito (*slave*) deve adeguarsi ed assecondare le richieste.
In genere il rapporto *master - slave* (che peraltro non è necessario) viene deciso per semplificare i meccanismi di comunicazione soprattutto per i piccoli processori dei nodi *slave*.

È interessante notare che le diverse visuali sopra citate sono in buona parte tra loro ortogonali: i ruoli ai diversi livelli sono cioè a priori indipendenti e sono dettati dalle varie esigenze applicative e dai vincoli implementativi, come nel seguito discusso in modo informale.

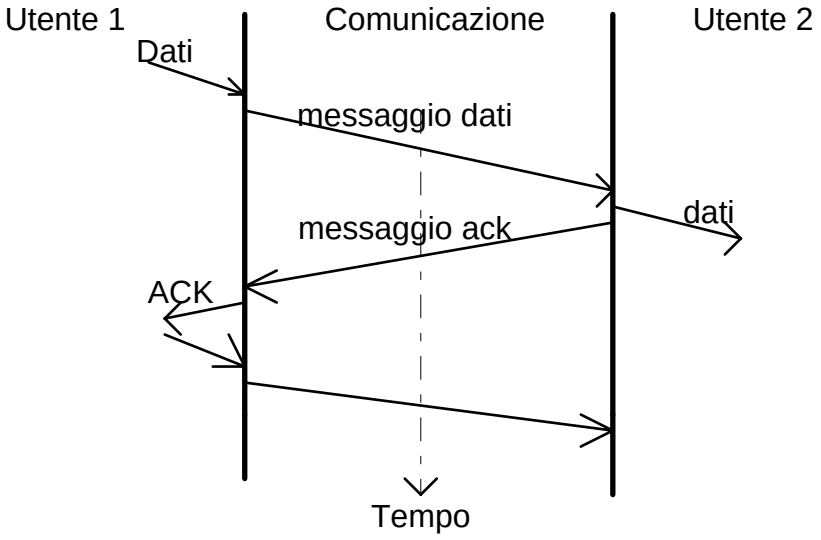
Ogni comunicazione prefigura l'instaurarsi di un rapporto “produttore - consumatore” tra due processi, attivi su macchine diverse, che rispettivamente generano le informazioni da trasmettere e utilizzano le informazioni ricevute.

È intuitivo notare come sia necessaria una **sincronizzazione** tra i due processi (non si possono utilizzare informazioni non ancora prodotte) ed eventualmente una **memorizzazione** se durante alcuni **intervalli limitati** di tempo la produzione è più veloce del consumo. In ogni caso la velocità media di produzione non dovrà essere superiore alla velocità massima di consumo.

Quando i processi produttore e consumatore non risiedono sullo stesso elaboratore i processi da considerare, e sincronizzare, sono in realtà almeno tre:

produttore
comunicazione
consumatore

Per inciso riportiamo una tipica tecnica di rappresentazione della successione nel tempo di interazioni tra diverse entità (fig.14.3), utilizzata sia per rappresentare comunicazioni tra i moduli componenti un calcolatore che, più spesso, per evidenziare le trasmissioni tra apparati di un sistema distribuito.
Questa rappresentazione prevede l'utilizzo di una colonna per ogni entità che si prende in considerazione, mentre l'asse dei tempi è orientato verticalmente e crescente verso il basso.
Le comunicazioni sono rappresentate mediante frecce orizzontali (o leggermente oblique verso il basso) che collegano la colonna del mittente a quella del destinatario e sono corredate di una sintetica descrizione a parole che spiega il significato delle informazioni trasferite.



Schemi descrittivi di interazioni

Fig. 14.3 - Esempio di schemi descrittivi di interazioni tra entità comunicanti.

Poichè le informazioni da trasferire devono in molti casi venir parcheggiate in opportune aree di memoria (*buffer*), non sono fuori luogo alcune note sulla

memorizzazione in buffer FIFO che preservano la sequenza temporale delle informazioni stesse.

- La memorizzazione non è necessaria se i processi produttore e consumatore possono essere rallentati quanto occorre per la sincronizzazione.
- La memorizzazione di un messaggio presso il trasmettitore è comunque richiesta dalla necessità di ripeterlo se si vuole ottenere la correzione degli errori rilevati in ricezione.
- La memorizzazione generalmente viene fatta a bordo del trasmittente, talvolta del ricevente e talvolta anche dai processi di comunicazione (ad es. nelle reti geografiche).
- In ogni caso la velocità potenziale media del consumatore deve essere maggiore di quella dei processi di comunicazione che a sua volta deve essere maggiore di quella del produttore.

Si noti l'asimmetria della situazione tra mittente e destinatario: l'operazione di trasmissione è iniziata “volontariamente” mentre la ricezione è “subita” dal destinatario che non sa quando arriverà il prossimo messaggio, ma deve essere pronto ad effettuarne la ricezione.
Instaurando un rapporto *master - slave* tra i dispositivi comunicanti si semplificano talora i problemi delle temporizzazioni al master, che controlla parzialmente i tempi delle comunicazioni.

Si noti che il ruolo di *master* può essere tenuto dal produttore o dal consumatore delle informazioni (o anche da un terzo elemento coordinatore), ed in genere la scelta di progetto cade su quello dei due che presenta vincoli più stringenti di coordinamento con altre attività.

Ad esempio uno strumento di acquisizione di dati (cioè un produttore) può essere vincolato ad una successione di campionamenti dalla tecnica adottata per le misure: in tal caso sarà probabilmente candidato al ruolo di *master* che invia i valori dei campioni quando questi sono acquisiti.

La situazione opposta si verifica quando il consumatore ha occasionalmente “bisogno” di informazioni (ad es. su richiesta dell'operatore) che tale consumatore potrà richiedere quando necessario se investito del ruolo di *master*.

Il rapporto *master-slave* si imposta tipicamente con una comunicazione di tipo *half-duplex* in cui il *master* può assumere l'iniziativa di mandare messaggi (contenenti informazioni o richieste), mentre lo *slave* è tenuto a rispondere sempre e solo ai messaggi del master, cogliendo tale occasione per inviare eventuali informazioni disponibili.

14.2.2 TIPI DI COMUNICAZIONI

Può essere interessante classificare i tipi di comunicazione nelle seguenti categorie:
- transazioni
- flusso di informazioni di stato
- flusso di informazioni di eventi.

14.2.2.1 TRANSAZIONI

Tipici di questa categoria sono i cosiddetti "*file transfer*", le interrogazioni di basi di dati remote e simili, ed in generale le funzionalità in cui si possano evidenziare rapporti *client-server*.
In genere il ruolo di master è inizialmente assegnato al sistema che effettua la richiesta (*client*) e successivamente è tenuto dal sistema (che può essere lo stesso) che funge da produttore.

È interessante notare che per queste funzioni spesso non si hanno problemi di tempo reale stretto e quindi i processi produttore e consumatore possono aspettare tutto il tempo necessario al partner, si possono effettuare memorizzazioni che preservino la sequenza e si devono prevedere tecniche di correzione (in genere con ripetizione) dei messaggi su cui in ricezione sono rilevati errori.

14.2.2.2 FLUSSO DI INFORMAZIONI DI STATO

Questa è la tipica situazione di collegamento tra un calcolatore di supervisione e controllo ed i sistemi di acquisizione che effettuano periodiche "fotografie" di grandezze di stato.

In questo caso la comunicazione è di tipo **ciclico** e non si hanno particolari problemi di temporizzazione se non che il periodo sia opportunamente breve e regolare in rapporto alla dinamica dei fenomeni in osservazione.

Gli errori devono essere **rilevati** per **scartare** i valori che ne sono affetti rinunciando però, in genere, alla loro correzione mediante ripetizioni. Infatti trattandosi di stati "continui" è accettabile mantenere i valori (o, più raramente, le loro derivate) precedentemente ricevuti, in attesa che poco dopo giungano i nuovi valori "freschi".

Il ruolo di master può essere assegnato (con problemi simmetrici) al produttore o al consumatore che si deve assumere la responsabilità di cadenzare i trasferimenti.
La memorizzazione temporanea, necessaria per gestire la comunicazione, riguarda esclusivamente le informazioni relative alla "fotografia" dello stato in corso di trasmissione e non si hanno code.
Questo flusso di informazioni riproduce il rapporto *writer-readers*.

14.2.2.3 FLUSSO DI INFORMAZIONI DI EVENTI

Nei sistemi di controllo si devono rilevare e generare anche degli eventi. Spesso si decide che tali eventi siano rilevati dal ricevente come variazioni degli stati e la

problematica relativa ricade nel caso precedente. Così facendo però si è costretti ad inserire nell'elenco delle informazioni di stato trasmesse ciclicamente anche tutte quelle relative a stati di scarso interesse se non per le loro **variazioni**, con un sovraccarico in molti casi non accettabile, o non conveniente, soprattutto per mezzi trasmissivi a banda limitata.

Quando il produttore genera informazioni di evento, in genere tutt'altro che cicliche e prevedibili e che non possono essere “perse” a causa di eventuali errori, si adotta una delle seguenti tecniche.

- - Chiunque sia il master, le informazioni di evento vengono associate alla prossima trasmissione di stati. Si deve accettare in tal caso un ritardo massimo pari al periodo di trasmissione ciclica e si deve prevedere un meccanismo di correzione degli eventuali errori sulle informazioni di evento.

- - Il produttore di eventi assume il ruolo di master per una immediata trasmissione aperiodica delle informazioni relative, ovviamente con meccanismi di correzione.

14.2.3 CONSISTENZA TEMPORALE

I meccanismi per realizzare e verificare le sincronizzazioni e le consistenze temporali tra processi, sono ovviamente basati sulla loro possibilità di comunicare. Nei sistemi distribuiti i tempi di comunicazione sono spesso non trascurabili e talora non noti (spesso noti solo come limite superiore). Ciò pone problemi più complessi di quelli riscontrabili nei sistemi con elaborazione centralizzata, come sinteticamente accennato nei punti seguenti.

14.2.3.1 Tempi assoluti

Perchè i diversi processori abbiano una consistente percezione dei tempi assoluti è necessario che i rispettivi orologi locali siano mantenuti sincronizzati tra loro. In letteratura sono stati proposti algoritmi che consentono di mantenere i vari tempi locali entro un limitato scostamento reciproco.
A grandi linee questi meccanismi sono basati sulle seguenti considerazioni.

- L’orologio di un sistema è assunto come riferimento a cui adeguare gli altri.
- La normale precisione degli orologi (oscillatori a quarzo) è usualmente adeguata per la correttezza nel breve periodo (cioè per durate di circa $\approx 10^5$ tick).
- Nel lungo periodo si adottano meccanismi di correzione basati su scambi di informazioni (i tempi locali) e **algoritmi** locali, che devono calcolare lo scostamento con segno tra gli orologi e devono **applicare le correzioni** in modo opportuno, cioè senza violare la regola di **monotonicità** del tempo e mantenendo il più possibile la **gradualità** per evitare errori inaccettabili nei calcoli di derivate e integrali.
- Per quanto riguarda lo scambio di informazioni va ricordato che i tempi di propagazione dei segnali elettrici sono di circa 10 ns/m, mentre i ritardi di trasmissione presentano le componenti additive Ta di accesso al mezzo trasmissivo e Tt = N*R di trasmissione di N bit alla velocità di R bit/s.

Mentre Tt è facilmente calcolabile, Ta è poco predicibile per alcune reti e in genere si è costretti a rilevarlo *on-line* mediante una doppia trasmissione di andata e ritorno, nella quale interviene anche il tempo (*Turnaround*) che il nodo impiega per rispondere. Il processo di aggiornamento orologi (Riferimento) calcola il tempo di comunicazione

come metà del tempo di andata e ritorno dei messaggi, e comunica ai nodi l'entità della correzione con l'algoritmo seguente.

Riferimento legge il suo tempo *Tr1*
Riferimento manda messaggio a nodo B
B risponde indicando il suo tempo locale *TLb1*
Riferimento legge il suo tempo *Tr2*
Riferimento calcola la correzione *Dtb = Tr2 - Tlb - (Tr2 - Tr1) / 2*
Riferimento manda a B il valore *Dtb*
B aggiunge (eventualmente in modo graduale) *Dtb* al suo tempo locale.

Nel caso particolare dei bus di campo con comunicazioni periodiche, il cui periodo è presumibilmente inferiore alla granularità temporale caratteristica, è comunque relativamente semplice mantenere gli errori entro il valore del periodo e quindi per definizione (di granularità caratteristica) accettabili.

14.2.3.2 Tempi relativi

Per quanto riguarda i tempi relativi, cioè le distanze temporali tra eventi rilevati (con *timestamping*) da diversi processori, il loro calcolo come differenza tra i tempi assoluti comporta un errore pari allo scostamento tra i due orologi locali.
Nel bus FIP, presentato nel seguito, sono previsti specifici meccanismi per correlare temporalmente gli eventi di produzione e di comunicazione di campioni di misure.

14.2.3.3 Sequenze

Si possono creare problemi di inconsistenza tra valori di variabili, di cui il progettista deve tener conto. In particolare i problemi si pongono quando in un nodo A un'informazione di stato (o evento) **causa** è ricevuta tramite comunicazione da un nodo remoto, mentre lo stato (o evento) **effetto** è letto direttamente come ingresso locale. Il nodo A "vede" l'effetto precedere erroneamente la causa, e questa situazione "anomala" perdura per tutto il tempo di comunicazione Tc, che può essere relativamentte lungo.

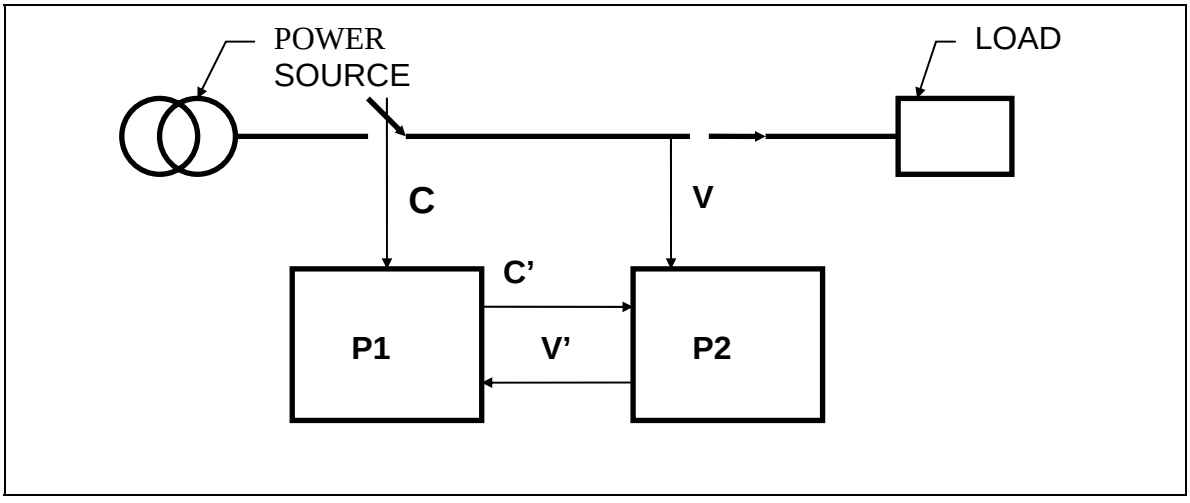


Fig. 14.4 - Esempio di situazione con potenziale inconsistenza causale tra variabili di stato con diversi percorsi di comunicazione.

Nella fig.14.4 si esemplifica la situazione citata con il seguente significato dei termini.

C - variabile booleana che rappresenta lo stato di chiusura di un contatto.

V - variabile booleana che rappresenta lo stato di presenza di tensione.
C' e **V'** sono i valori di aggiornamento delle variabili globali, effettuati con ritardi Tc dovuti alla comunicazione.
Tp - tempo, trascurabile, per la propagazione della tensione elettrica alla chiusura del contatto.

In assenza di guasti, la consistenza causale può essere espressa come:

V(t+Tp) = C(t) con $Tpmin \leq \mathbf{Tp} \leq Tpmax$

Il processore P2 vede **V** e **C'** che non rispettano la consistenza causale e potrebbe in particolare riportare l'anomalia **AN** che esprime presenza di tensione con contatto apparentemente aperto, data da:

AN = not C' and V

Tale anomalia rientra entro un tempo Tc, ma nel frattempo può causare corse critiche, ad esempio in caso di gestione di allarmi.

14.3 TOPOLOGIE DEI SISTEMI DISTRIBUITI

La topologia fisica delle interconnessioni tra vari dispositivi è generalmente imposta o suggerita dal tipo dei dispositivi stessi, dal tipo di mezzi trasmissivi adottati e da considerazioni di prestazioni, riconfigurabilità e tolleranza ai guasti.
È interessante notare che la funzionalità di comunicazione ai livelli più alti può prescindere in larga misura dalla topologia fisica, soprattutto se si adottano protocolli stratificati a livelli.

Nel seguito sono brevemente presentate e discusse le principali topologie adottate nelle comunicazioni tra dispositivi di controllo e acquisizione dati che però, per quanto detto sopra, si prestano in genere molto bene anche per applicazioni di tipo gestionale o misto.

14.3.1 COLLEGAMENTI SEMPLICI

14.3.1.1 COLLEGAMENTI PUNTO-PUNTO

Costituisce il caso più semplice e normalmente adotta una comunicazione di tipo *half-duplex.*
Generalmente uno dei due interlocutori gioca il ruolo di *master* permanente, ma talvolta il rapporto *master-slave* è instaurato di volta in volta a seconda del tipo di comunicazione da effettuare.

14.3.1.2 COLLEGAMENTI MULTI-PUNTO MASTER-SLAVE

È L'approccio adottato quando si abbia una marcata gerarchia tra un calcolatore, a cui si affida il compito di assumere l'iniziativa di tutte le comunicazioni e di gestirne lo svolgimento, e vari dispositivi asserviti.
Dal punto di vista fisico è possibile realizzare tale architettura con una configurazione a stella, come estensione del caso punto-punto, ma la situazione più interessante per economicità ed espandibilità prevede l'adozione di un bus, cioè un'interconnessione che

interessa in parallelo tutti i dispositivi, basato in molti casi sullo standard elettrico EIA RS 485 che consente tale connessione.

Si noti che il collegamento a stella richiede presso il nodo master un'interfaccia seriale per ogni dispositivo collegato consentendo un più breve tempo di risposta ed un più elevato volume di traffico globale a costo di una minore modularità rispetto alla soluzione a bus, che però costituisce un “collo di bottiglia”.

“**Polling**” è detta **l'interrogazione** (generalmente ciclica) che il *master* effettua su tutti gli slave per ricevere informazioni che essi abbiano eventualmente disponibili. Ciò avviene emettendo messaggi che specificano l'indirizzo dei dispositivi interrogati ed attendendo da essi una risposta che contiene le informazioni disponibili o che indica assenza di informazioni da inviare.

“**Selecting**” è invece l'operazione mediante la quale il *master* seleziona il particolare *slave* al quale vuole **inviare informazioni**. Lo slave indirizzato risponderà con un messaggio di ACK (= acknowledge = riconoscimento) se ha ricevuto correttamente il messaggio, mentre risponderà con NACK (= not acknowledge) per chiedere la ripetizione in caso di errore.

Con i collegamenti multipunto è possibile anche una comunicazione di tipo *broadcast* o di tipo *multicast*, cioè con un codice di indirizzo riconosciuto rispettivamente da tutti i - o da un sottoinsieme dei - dispositivi *slave*; in questi casi però non si richiede generalmente la conferma di ricezione corretta.

14.3.2 COLLEGAMENTI IN RETE

Sono dette reti (di calcolatori) i collegamenti tra vari dispositivi, detti nodi della rete, tra i quali sono possibili diverse funzioni di comunicazione, senza necessariamente assegnare ruoli permanenti di master e di slave.
Tra i problemi tipici delle reti segnaliamo i seguenti.

- - accesso al mezzo trasmissivo, in genere condiviso con allocazione
  statica (*time-slot*) degli intervalli
  dinamica
  a gestione centralizzata
  a gestione distribuita
- - tecniche di risoluzione dei conflitti di accesso contemporaneo di più nodi
- - instradamento dei messaggi dal mittente al destinatario.
- - gestione della correttezza (integrità e ordinamento) delle informazioni.

Si noti che nelle reti più complesse, soprattutto quando i nodi sono calcolatori utilizzati da operatori e si hanno collegamenti mediante "*bridge*" o "*gateway*" anche tra reti diverse, si presentano ulteriori problemi di **diritti di accesso**, ecc. Questo però non è generalmente il caso nei sistemi di acquisizione dati e controllo di processo.

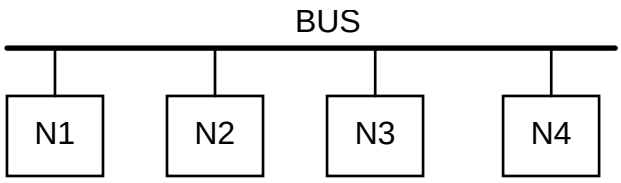
Una tipica distinzione delle reti prevede la loro classificazione in WAN e LAN (e SAN).
- **WAN** = *Wide Area Network* = reti geografiche, sono reti basate su canali trasmissivi pubblici, spesso di tipo telefonico, e collegano nodi di diversi enti anche a notevole distanza.
- **LAN** = *Local Area Network* = reti locali, collegano nodi della stessa organizzazione, in genere nello stesso edificio e comunque a distanze inferiori a circa un chilometro con mezzi trasmissivi privati e dedicati e con velocità anche molto elevate (10 Mbit/s).
- **SAN** = *Small Area Network* = reti ad estensione ridotta: si tratta di una sigla non molto usata che però designa le sempre più diffuse reti che restano confinate all'interno di un singolo apparato collegando semplici ed economici dispositivi.

Nella maggior parte delle applicazioni industriali si utilizzano reti LAN (e SAN) che consentono un miglior controllo dei tempi di comunicazione (tempo reale), mentre le WAN vengono in genere destinate solo ad applicazioni gestionali.

14.3.2.1 TOPOLOGIE DELLE RETI

Le topologie tipiche di interconnessione tra i nodi delle reti sono a bus, ad anello, a stella e a grafo generico.

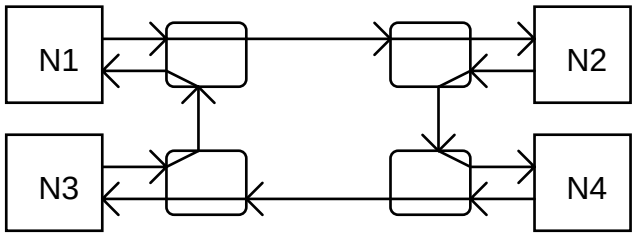
La struttura a bus è basata su un mezzo trasmissivo (solitamente cavo attorcigliato - *twisted pair* - oppure cavo coassiale) in comune per tutti i nodi che sono collegati quindi in parallelo.



Rete a BUS

Fig. 14.5 - Rete a bus.

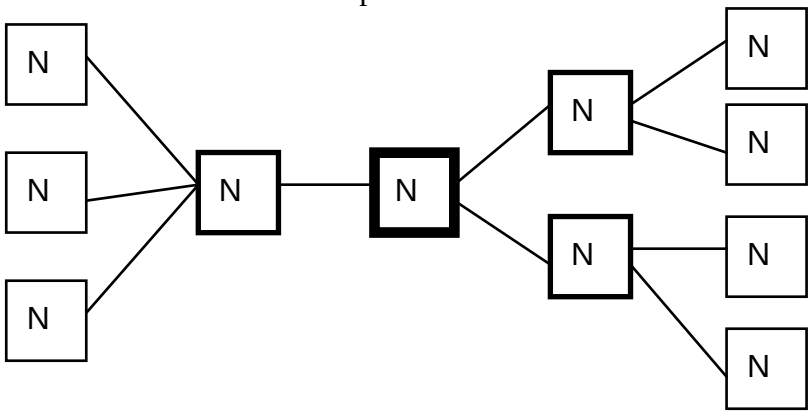
La struttura ad anello consiste in una connessione in cui ogni nodo riceve dal nodo a monte e trasmette a quello a valle, con una catena che alla fine si richiude sul primo nodo.



Rete ad ANELLO

Fig. 14.6 Rete ad anello

Il collegamento a stella (o albero) prevede un nodo centrale dotato di collegamenti dedicati ai nodi satelliti, che a loro volta possono essere collegati a nodi di livello inferiore. Si tratta di una struttura tipicamente gerarchica, in cui ogni nodo funge da concentratore / distributore per le comunicazioni con i livelli inferiori.



Rete ad ALBERO

Fig. 14.7 - Rete ad albero.

Infine si possono avere interconnessioni che possono essere descritte come grafi in cui ogni nodo è connesso con un certo numero di altri nodi, senza regole generali. Questa è la tipica struttura delle **reti geografiche**, nelle quali è tipica la distinzione tra i nodi di rete, detti IMP ed i calcolatori utenti delle connessioni, gli Host.

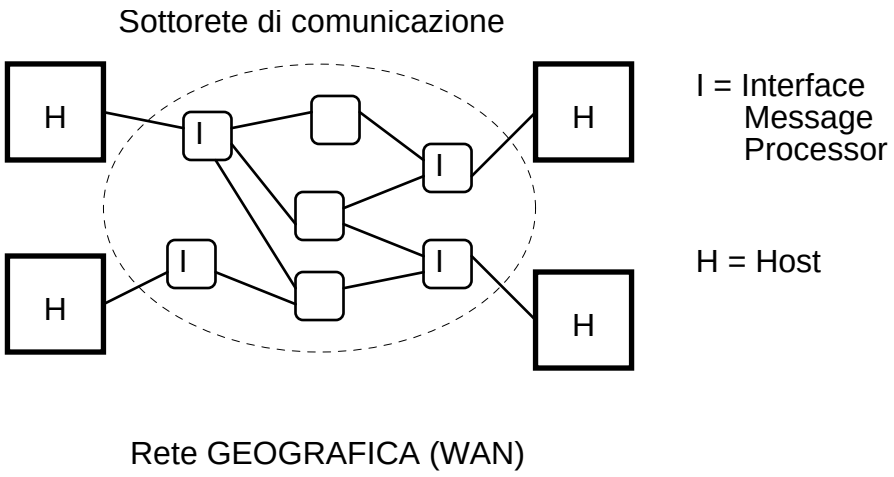


Fig. 14.8 -- Rete WAN con sottorete di comunicazione.

Nel seguito vengono presentate un pò più in dettaglio le architetture ad anello e a bus che sono le più diffuse per le reti locali, grazie alla loro maggiore regolarità e modularità.

14.3.2.1.1 CONNESSIONE AD ANELLO

La struttura delle reti ad anello (*ring*), come si è detto, è costituita da una catena chiusa di collegamenti punto a punto.
Poichè ogni nodo è l'unico responsabile della trasmissione verso il nodo a valle non vi sono conflitti di accesso fisico al mezzo trasmissivo. È però necessario regolamentare l'accesso logico, cioè attribuire ordinatamente ai vari nodi i turni di trasmissione delle proprie informazioni ed i periodi in cui funzionano come ricevitori e ripetitori.

La tecnica più diffusa per evitare conflitti logici è la cosiddetta "*token-ring*" specificata dallo standard IEEE 802.5. Si tratta di un metodo di accesso deterministico in cui un nodo non può trasmettere propri messaggi se non è in possesso dell'unico "gettone" (token) della rete.
Il gettone, rappresentato da un opportuno simbolo, circola continuamente in assenza di messaggi da trasmettere, mentre viene trattenuto dal nodo che usufruisce del suo diritto a "parlare" e rimesso in circolazione alla fine della trasmissione del messaggio.
L'accesso è quindi ordinato, di tipo "*round-robin*" (cioè a priorità ciclica) con un ottimo sfruttamento del mezzo trasmissivo anche al massimo volume di traffico.
La necessaria continuità dell'anello è ripristinata, in caso di caduta od esclusione di un nodo, mediante opportuni contatti che "scavalcano" l'interruzione.

Più complessi sono i meccanismi di recupero delle situazioni in cui, per errore, il gettone venga perso o duplicato.
Le connessioni ad anello sono abbastanza diffuse nelle reti per applicazioni gestionali ma pressochè assenti nelle applicazioni di automazione industriale.

14.3.2.1.2 CONNESSIONI A BUS

Sono connessioni tipicamente multipunto che pongono quindi il problema di arbitraggio dei conflitti di accesso fisico al mezzo trasmissivo condiviso.
L'arbitraggio può essere centralizzato (*traffic director*) o distribuito tra i nodi paritetici, come brevemente presentato nel seguito.

- **- TRAFFIC DIRECTOR**

Una possibile soluzione è basata sull'assegnazione ad un nodo della funzione di "*traffic director*" (*master*) che coordina le trasmissioni dei vari nodi (*slave*) con un'opportuna successione di interrogazioni. Recentemente nelle reti a bus si sono però imposte le tecniche presentate nel seguito, che assegnano ai vari nodi ruoli più paritetici, mentre l'arbitraggio centralizzato rimane tipico per alcuni Fieldbus (v. seguito).

- **- CSMA/CD**

Un approccio molto diffuso, tipico delle reti Ethernet, è il cosiddetto CSMA/CD cioè *Carrier Sense, Multiple Access, Collision Detect*, la cui funzionalità è la seguente:

- tutti i nodi sono in grado di rilevare la presenza di trasmissione in corso (bus occupato) o di bus libero;
- eventuali trasmissioni di messaggi vengono iniziate solo quando si rilevi lo stato di bus libero;
- durante la trasmissione vengono rilevate le eventuali collisioni confrontando il segnale in linea con quello trasmesso;
- in caso di collisione tutti i nodi in trasmissione cessano di trasmettere ed attendono un tempo casuale prima di ritentare;
- i nodi che non stanno trasmettendo sono in ricezione.

Questa tecnica (standard IEEE 802.3) consente un rapido accesso quando si ha un basso volume di traffico, mentre con volumi di traffico prossimi alla banda del mezzo trasmissivo porta a scarso determinismo nei tempi di accesso e a degrado di efficienza.
Per questi motivi la tecnica CSMA/CD, pur diffusa in ambito gestionale, è scarsamente utilizzata in applicazioni industriali.

- **- TOKEN BUS**

Utilizza una tecnica a passaggio di gettone (*token passing*) simile a quella vista per le strutture ad anello, con la sostituzione di un anello logico all'anello fisico. Il gettone è costituito da una trama (messaggio senza testo) che il possessore invia al suo successore, corredandola degli identificatori di mittente e destinatario.
Ogni nodo mantiene una tabella descrittiva dell'anello logico, in base alla quale conosce i suoi predecessori ed il suo successore, in modo da adottare opportuni algoritmi di recupero nel caso di errori o guasti che portino alla perdita del gettone.
Questa tabella viene compilata in una opportuna fase di inizializzazione della rete che va ripetuta ogni volta che un nodo venga inserito od escluso dalla rete e quindi dall'anello logico.

La tecnica *token-bus*, oggetto dello standard IEEE 802.4, è molto diffusa in applicazioni di automazione industriale e costituisce la base di reti ad alte prestazioni (MAP) come pure di alcuni tipi di bus di campo, per le sue caratteristiche che associano un determinismo di comportamento alla modularità ed economia di connessione tipica delle strutture a bus.

14.4 PROTOCOLLI E PROBLEMATICHE GENERALI

Le funzioni necessarie per la comunicazione tra calcolatori collegati in rete sono varie ed in genere piuttosto complesse; richiedono quindi una strutturazione basata su livelli che possono differire per i vari tipi di reti.

Ogni livello offre servizi al livello superiore, mascherando l'implementazione basata sui servizi ottenuti dal livello inferiore.
Una buona strutturazione prevede infatti che tra i vari moduli si abbiano interfacce chiare e semplici ed il più possibile indipendenti dall'implementazione interna dei singoli moduli.
Se ad ogni livello viene realizzata una comunicazione virtuale con un proprio protocollo di livello si realizza una significativa indipendenza tra gli aspetti logici e le tecniche realizzative.

L'insieme dei livelli è detto **Architettura di rete**, descritta dalle varie funzionalità a prescindere dalle implementazioni e dalla distribuzione dei compiti tra dispositivi circuitali (Hardware) e programmi (Software).
Queste esigenze portarono verso la fine degli anni '70 ad una proposta della ISO (*International Standards Organization*) di un modello di architettura di rete per sistemi aperti, cioè non proprietari, detto OSI (*Open Systems Interconnection*).

Il modello ormai ben noto come ISO-OSI è basato su 7 livelli: ai livelli inferiori sono assegnate funzionalità più legate ai mezzi trasmissivi, mentre a quelli superiori sono previsti servizi relativi alle applicazioni.
Il modello ISO-OSI è proposto come standard di riferimento ma non intende costituire una specifica realizzativa: è infatti previsto che esistano standard separati per le realizzazioni dei vari livelli in diversi contesti.
Nel seguito vengono presentati molto sinteticamente gli aspetti principali del modello ISO-OSI, mentre si rimanda ai testi specialistici (Ad es. Computer Networks di Tanenbaum) per un approfondimento dell'argomento ricco di problemi tuttora in buona parte non risolti.

14.4.1 I LIVELLI DEL MODELLO ISO-OSI

1 - **FISICO** - Physical layer
Si occupa della trasmissione di bit sul canale trasmissivo affrontando le problematiche relative ai segnali, alle temporizzazioni, alla direzione di trasmissione, come pure agli aspetti meccanici costruttivi dei connettori, ecc.

> ACCESSO AL MEZZO - Medium Access Control (MAC) Sublayer
> Nel caso di reti a mezzo trasmissivo condiviso, quindi con problemi di risoluzione dei conflitti di accesso, si è ritenuto opportuno aggiungere questo sotto-livello che può essere considerato come un'estensione del livello 1.

2 - **LINEA DATI** - Data Link layer
A questo livello le informazioni sono viste come trame (messaggi) e vengono risolti i problemi di riconoscimento di inizio e fine trama, rilievo ed eventuale correzione di errori, regolazione del flusso di informazioni in base alla capacità di gestirle dei nodi interessati.

3 - **RETE** - Network layer
Questo livello gestisce la sottorete di comunicazione ed eventualmente i passaggi tra diverse sottoreti realizzando le funzioni di instradamento dei pacchetti, di gestione dello stato di congestione del traffico, gestione degli addebiti agli utenti.
Con canali trasmissivi condivisi (bus, onde radio, ecc.) le funzioni di questo livello si riducono a ben poco.

4 - **TRASPORTO** - Transport layer
Il livello trasporto costituisce l'interfaccia tra i livelli sottostanti dipendenti dagli aspetti hardware dei mezzi trasmissivi ed i livelli superiori che sono prevalentemente condizionati dai Sistemi Operativi dei calcolatori Host (nodi della rete).
A questo livello le funzioni principali riguardano la creazione di collegamenti virtuali, la scomposizione in pacchetti dei dati e la ricomposizione dei dati corretti ed in sequenza, la gestione dei datagrammi (messaggi isolati), e le comunicazioni di tipo *broadcast*.
Costituisce il primo livello in cui si realizza un vero protocollo "*end to end*".

5 - **SESSIONE** - Session layer
È un livello con poche funzionalità, orientate essenzialmente alla instaurazione, gestione e chiusura di "sessioni" di scambio di dati in modo svincolato dalle operazioni relative ai collegamenti virtuali e, generalmente, con un protocollo *half-duplex*.

6 - **PRESENTAZIONE** - Presentation layer
Il livello presentazione è interessato agli aspetti di sintassi (struttura) e semantica (significato) delle informazioni da comunicare, svolgendo le operazioni di eventuale trascodifica, compressione, crittografia, ecc.
Molto importanti nelle reti geografiche, o comunque di uso pubblico, sono gli aspetti di protezione dei dati.

7 - **APPLICAZIONE** - Application layer
In questo livello si collocano le applicazioni di uso abbastanza diffuso da giustificare una standardizzazione che le renda utilizzabili senza doversi adeguare alle diverse implementazioni.
Tipici esempi sono l'accesso, il trasferimento e la gestione di file, i terminali virtuali, la posta elettronica (electronic mail), ecc.

14.4.2 CONSIDERAZIONI GENERALI SUI LIVELLI ISO-OSI

I livelli ISO-OSI possono essere suddivisi nei sottoinsiemi: livelli inferiori (1,2,3) e livelli superiori (4,5,6,7).

I livelli inferiori riguardano l'interfacciamento tra macchine adiacenti e concatenate nella sottorete di comunicazione, che sono spesso dette IMP (*Interface Message Processor*) dato che non svolgono altro che funzioni correlate con la gestione delle comunicazioni.

I livelli superiori realizzano protocolli *end to end*, cioè tra i processi di elaborazione che svolgono le funzioni tipicamente applicative e che girano sui calcolatori detti "host".

Ad ogni livello vengono gestiti protocolli virtuali tra partner di pari livello. Questi protocolli si basano in realtà sui servizi forniti dal livello inferiore, tranne che al livello fisico dove ovviamente il collegamento è diretto.

I vari protocolli sono gestiti con aggiunta ad ogni livello di intestazioni (*envelope* = buste) che vengono interpretate e tolte dal partner di corrispondente livello, mentre le informazioni rimanenti sono trattate come dati scambiati con il livello superiore.

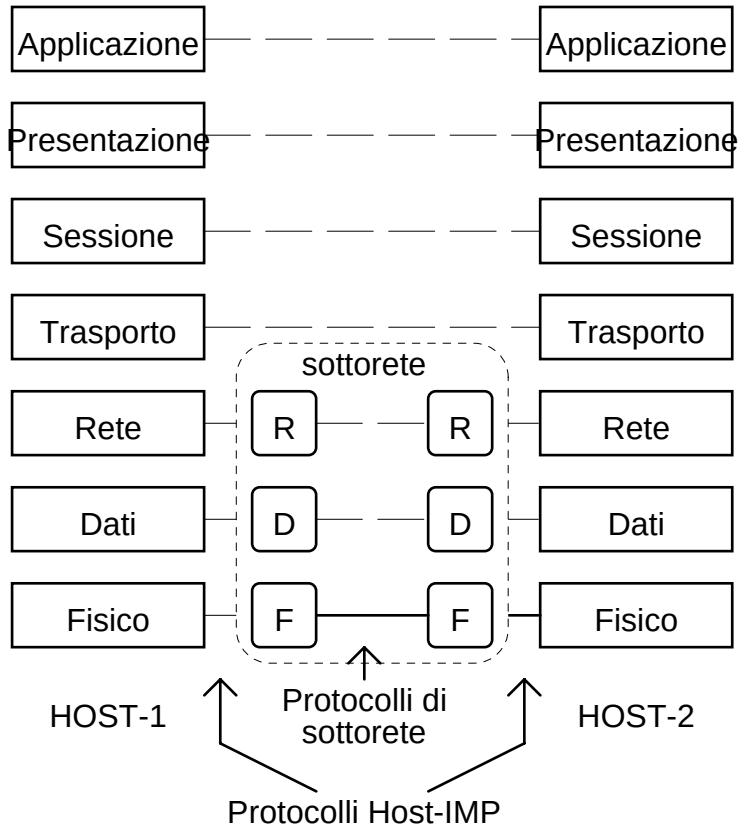


Fig. 14.9 Architettura di rete a livelli OSI

Ad ogni livello le funzioni sono realizzate da "entità" che in linea di principio possono essere di tipo hardware o software, con prevalenza delle entità software nei livelli

superiori o comunque per le funzioni che richiedono vincoli meno stringenti sui tempi di esecuzione.

Ai servizi richiesti al rispettivo livello inferiore si accede tramite i cosiddetti SAP (*Service Access Point*) a cui sono passate le IDU (*Interface Data Unit*) che contengono:

SDU (*Service Data Unit*) destinate al partner (*peer*) e
ICI (*Interface Control Information*) per il livello servente.

Il protocollo virtuale del livello avviene tramite le PDU (*Protocol Data Unit*) che sono gli elementi usati per la comunicazione e contengono porzioni di SDU e intestazioni con le informazioni di controllo.

I servizi di comunicazione vengono classificati nei tipi seguenti.

- Servizi **connection**, che possono essere paragonati alle telefonate, che supportano transazioni con sequenze ordinate di fasi intermedie, ognuna delle quali non viene attivata finchè la precedente non è correttamente completata;
- Servizi **connectionless**, mediante i quali si inviano messaggi isolati, detti anche "datagram", paragonabili a delle lettere che seguono ognuna una sua storia e non mantengono necessariamente la sequenza temporale.

I servizi *connectionless* possono a loro volta essere di tipo con o senza conferma (*acknowledge*) dal ricevente.

14.4.3 PRIMITIVE DI SERVIZI

I servizi sono specificati mediante un insieme di primitive corredate di eventuali parametri e classificabili nelle 4 seguenti categorie:

- **Request** per attivare un'azione
- **Indication** per segnalare un'azione del partner
- **Response** per rispondere ad iniziative del partner
- **Confirm** per segnalare una risposta del partner

Response e *Confirm* sono previste solo per servizi con conferma.

Ad esempio per un servizio di tipo *connection* si potranno avere le seguenti primitive:

CONNECTION.request	richiesta di stabilire una connessione
CONNECTION.indication	segnala al chiamato
CONNECTION.response	il chiamato accetta o rifiuta
CONNECTION.confirm	il chiamante è informato sull'esito dell'accettazione
DATA.request	richiesta di invio di dati
DATA.indication	segnalazione di ricezione di dati
- se non è prevista conferma mancano .response e .confirm	
DISCONNECT.request	richiede di rilasciare una connessione
DISCONNECT.indication	segnala al partner la richiesta

Una connessione si "**stabilisce**" e si "**rilascia**" (non si apre e si chiude) per non creare confusione tra tecnici elettrici ed informatici.

In conclusione possiamo dire che:

- - I servizi sono funzioni astratte scambiate tra entità poste a diversi livelli adiacenti
- - I protocolli sono l'implementazione tra partner di pari livello delle comunicazioni (virtuali) occorrenti per realizzare i servizi.

Lo standard OSI, come ammesso dagli stessi proponenti, non è perfetto perchè affrettato e risultante dalla convenienza ad adottare tecniche ormai già diffuse nelle applicazioni realizzate, nè è statico.
I protocolli effettivi, cioè le implementazioni dei servizi previsti ai vari livelli, come si è detto, sono oggetto di vari standard specifici.

14.4.4 STANDARD BASATI SUI LIVELLI ISO-OSI

Nel seguito vengono citati alcuni degli standard che precisano i protocolli per alcuni o tutti i livelli ISO-OSI mentre si rimanda alla letteratura in bibliografia il lettore interessato ad ulteriori informazioni.

- **- MAP (Manufacturing Automation Protocol)**

Il MAP, proposto dalla General Motors per organizzare le comunicazioni tra dispositivi (controllori, PLC, calcolatori di processo, ecc.) anche di diversi costruttori in ambito di automazione di fabbrica, copre tutti i livelli ISO-OSI ed è in continua evoluzione.
A livello fisico è prevista una connessione con canali a larga banda (tipo TV) o, per il più recente Mini-MAP, un collegamento a bus di tipo token-bus (IEEE-802.4).
Sono previste varianti su fibre ottiche.
Per i livelli superiori gli standard sono compatibili con il TOP (v. sotto) ed una particolare attenzione è stata posta nel definire l'interfaccia fra livello applicazione ed i processi utenti.

- **- TOP (Technical Office Protocol)**

Il TOP è stato proposto dalla Boeing per applicazioni ai livelli superiori del CIM orientate all'automazione di ufficio e si basa su un livello fisico tipo IEEE-802.3 (ethernet).
Nei protocolli MAP e TOP particolare attenzione è stata posta all'uso e connessione di diverse reti (*internetworking*) con la definizione di elementi repeater, bridge, router e gateway.

repeater	liv 1
bridge	liv 2 ethernet - token bus
router	liv 3 token bus - X.25
gateway	liv x (7) tra reti incompatibili

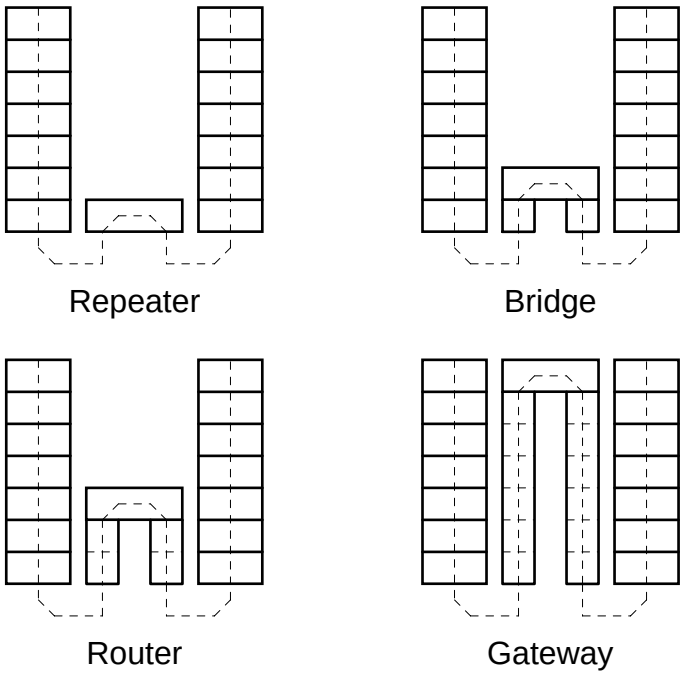


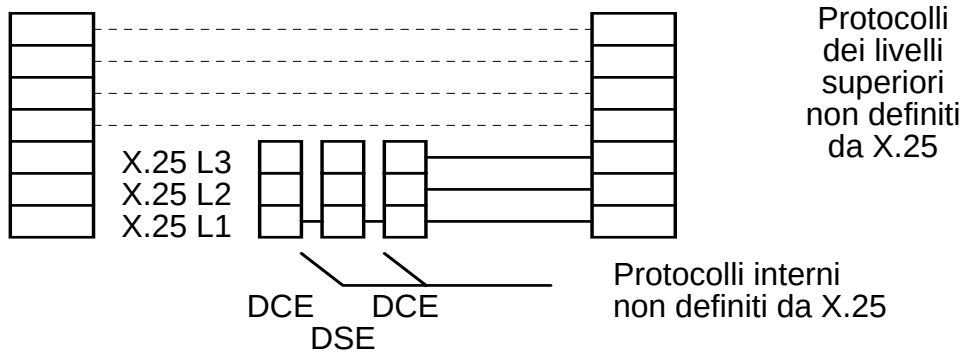
Fig. 14.10 - Interconnessioni tra reti.

- **- TCP/IP**

I protocolli a livello trasporto (TCP) e rete (IP) adottati dalla rete ARPA sono stati adottati anche da diversi sistemi commerciali. (V. pag. 429 Tanenbaum)

- **- X.25 CCITT**

La raccomandazione X.25 è molto usata internazionalmente per reti pubbliche e raccoglie standard per i livelli inferiori (1,2 e 3) e descrive l'interfaccia tra un DTE (Data Terminal Equipment = HOST) e un DCE (Data Circuit Equipment), mentre non specifica i protocolli tra DCE e DSE (Data Switching Exchange = IMP).
Gli standard usati per i livelli sono:
- livello fisico
  standard X.21 oppure RS-232 (X.21 bis)
- livello data link
  standard LAP (Link Access Procedure) simile a HDLC
- livello network
  tratta indirizzamento, flussi, conferme, ecc. supportando due forme di connessione:
  1. - circuito virtuale commutato che dopo la procedura di inizio per stabilire il circuito virtuale consente uno scambio full-duplex di dato organizzati in pacchetti
  2. - circuito virtuale permanente su cui si possono inviare direttamente dei pacchetti (datagrammi).



Collocazione di X.25 nella gerarchia di protocolli

Fig. 14.11 - Collocazione di X.25 nella gerarchia di protocolli.

14.5 RETI TIPICHE PER CONTROLLO DI PROCESSO

Le comunicazioni tra apparecchiature destinate alle funzioni di acquisizione dati e controllo di processi industriali presentano, come già accennato, problematiche particolari soprattutto per quanto concerne le temporizzazioni.

Inoltre la recente e crescente tendenza a corredare di microcalcolatori anche i dispositivi tipicamente terminali, come i sensori e gli attuatori ha portato alla necessità di sviluppare e standardizzare sistemi di comunicazione adatti a sostituire in modo efficace ed economico le tradizionali connessioni sul campo.

Si possono delineare due categorie principali di sistemi di comunicazione orientati alle applicazioni industriali:

- le reti industriali
- i bus di campo (*fieldbus*)

Non sempre la distinzione tra questi due tipi di reti è netta, anche perchè in entrambi i casi si preferiscono le strutture di **connessione a bus**, sia pure con diverse varianti, ma in generale i fattori discriminanti sono costituiti dalle prestazioni e dai costi.
Le prestazioni delle reti sono infatti più elevate, sia in termini di **distanze** copribili, che possono superare il chilometro, sia in termini di numero di nodi, ma soprattutto in termini di **velocità** di trasmissione che possono superare i 5 Mbit/s.

Un tipico esempio di rete per automazione industriale è rappresentato dallo standard MAP (Manufacturing Automation Protocol), di cui si è accennato, basato sulla tecnica *token-bus*, ma sono diffuse anche connessioni di tipo Ethernet.

14.5.1 BUS DI CAMPO (Fieldbus)

Particolarmente interessanti ed in fase di notevole sviluppo sono i bus di campo, di cui viene presentata nel seguito una breve rassegna, e che costituiscono un terreno di sfida per i tecnici che devono risolvere nel miglior modo il compromesso tra prestazioni, requisiti real-time, robustezza ed economicità.

In fig.14.12 è riportato lo schema logico di un sistema di controllo distribuito, composto da sensori, attuatori e uno o più dispositivi controllore (in figura sono due).

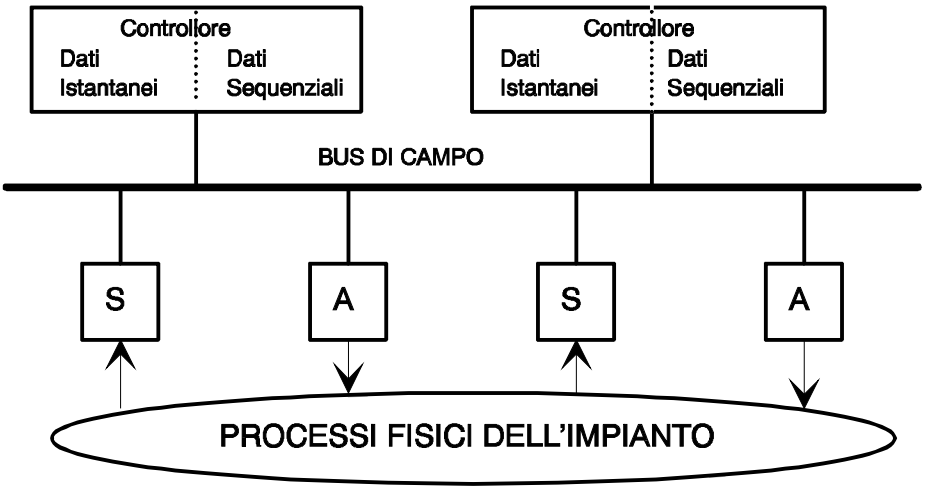


Fig. 14.12 - Struttura di un sistema di controllo distribuito. S e A sono microcontrollori che gestiscono rispettivamente Sensori e Attuatori.

In ognuno dei nodi controllori sono conservati due insiemi di dati:

- **Istantaneo:** contiene i valori delle variabili di **stato** del processo sotto controllo che vengono rinfrescate periodicamente dalle alle misure campionate sui sensori.
- **Sequenziale:** contiene le code degli **eventi** significativi per il processo.

14.5.1.1 Requisiti dei bus di campo.

Il fieldbus deve consentire lo scambio di informazioni tra i dispositivi coinvolti nello schema di fig.14.12 e deve garantire le seguenti funzionalità.

- I dati istantanei vengano rinfrescati con una **frequenza** pari alla frequenza di campionamento dei sensori.
- Il **ritardo** di trasmissione delle misure dei sensori e dei dati da inviare agli attuatori non sia tale da pregiudicare la stabilità del sistema.
- I dati istantanei relativi al perocesso si riferiscano tutti allo stesso periodo di tempo (**coerenza temporale**).
- Nel caso in cui più nodi controllori facciano uso di una stessa variabile essi devono in ogni istante disporre di identici valori di tale variabile (**coerenza spaziale**).
- Sia garantita **l'operatività** del sistema anche in caso di utilizzo di dispositivi di campo (S e A) provenienti da costruttori diversi.
- Il ritardo di trasmissione di messaggi di tipo **sporadico** (ad esempio allarmi) sia minimo.
- Sia possibile la connessione al bus usando **interfacce semplici ed economiche**.
- In alcuni casi vengono poi richiesti requisiti di particolare **affidabilità** e **sicurezza** (ad es. nel caso di impiego in ambienti a rischio di esplosione).

I servizi richiesti si possono ricondurre ai seguenti:

- comunicazione ciclica di informazioni di stato
- comunicazione aperiodica ad alta priorità di eventi
- comunicazione aperiodica a bassa priorità di informazioni di configurazione.

14.5.2 BITBUS

Intel

Proposto da Intel e basato sul protocollo SDLC di IBM con codifica NRZI e velocità standard di 64, 375 e 2400 Kbit/s. Le specifiche elettriche sono RS-485 su cavo attorcigliato e schermato.

Sono previsti tre tipi di nodi:
- master
- slave
- repeater

La stazione *master* controlla i trasferimenti indirizzando le stazioni secondarie (*slave*) che, con l'uso dei ripetitori, possono essere fino a 250.

A livello applicazione sono previste diverse primitive, senza però meccanismi espliciti per comunicazioni cicliche.

14.5.3 PHOEBUS

Svizzera

Sviluppato presso l'Istituto Federale di Tecnologia di Losanna per applicazioni di CNC, come supporto alle connessioni tra PLC e sensori ed attuatori.

Le transazioni sul bus prevedono una fase sincrona (per le informazioni di stato) ed una fase asincrona (per messaggi applicativi o servizi di rete).

A livello **fisico** si trasmettono bit a 375 Kbit/s con codifica NRZI su cavi attorcigliati e schermati.
A livello **data link** si formano trame di lunghezza prefissata per ogni stazione, con servizi di trasmissione con e senza riscontro (ack).
A livello **applicazione** sono previste le categorie di servizi:
- servizi di manipolazione di data base
- servizi di gestione di rete
- servizi di configurazione e controllo delle stazioni secondarie.

14.5.4 MIL-STD 1553

USA

Sviluppato in ambito militare da aeronautica e marina USA.
La comunicazione a 1 Mbit/s è basata su codifica Manchester bi-fase su un bus a cavo attorcigliato e schermato a cui possono essere collegate fino a 31 stazioni remote slave.
Le trame, sia di comando che di dati o di stato, hanno il formato fisso di 3 bit di sincronizzazione, 16 bit di dati e 1 bit di parità.
Viene data molta importanza ad una buona immunità ai disturbi con rigorose specifiche di interfaccia.

14.5.5 PROFIBUS

Germania

Il PROcess FIeld BUS è nato dagli sforzi congiunti di alcune industrie tedesche (inizialmente Bosch, Siemens e Kloekner- Moeller).
Le specifiche di PROFIBUS sono contenute nella norma DIN 19245 (Parti I e II), e sono state definite in base all'OSI Basic Reference Model con i livelli dal 3 al 6 vuoti.

Questo bus di campo è particolarmente adatto per un approccio *event-driven* dei vari processi di automazione.

14.5.5.1 Livello Physical.

Topologia.

La topologia adottata è quella a bus, e prevede anche l'uso di ripetitori. Il raggio di azione arriva a 1200 m (con ripetitori a 4800), e si possono collegare fino a 32 stazioni (127 con ripetitori).

Mezzo trasmissivo.

Il mezzo trasmissivo è il doppino attorcigliato, con standard elettrico RS-485. Per il futuro è previsto anche l'uso della fibra ottica. Sono previste le velocità di trasmissione: 9.6 Kbit/s, 93.75 Kbit/s, 187 Kbit/s e 500 Kbit/s.

Codifica.

La codifica è in banda base, del tipo NRZ.

14.5.5.2 Livello Data Link.

Controllo dell'accesso al canale trasmissivo (MAC).

I nodi possono essere di due tipi.

Stazioni attive: corrispondono a sistemi di media o elevata complessità (PC, PLC, Workstation,....) e si scambiano il diritto a iniziare trasmissioni con un meccanismo di tipo *token passing*.

Stazioni passive: in questa categoria vengono collocati i dispositivi più semplici che accedono al bus solo in risposta ad un'interrogazione da parte di una stazione attiva, dovendo così gestire solo una parte semplificata del protocollo.

Un nodo attivo in possesso del *token* può rivolgersi sia ad un nodo passivo sia ad un altro nodo attivo, instaurando così relazioni di comunicazione di tipo *master-slave* o *master-master*. Questo tipo di MAC consente alle stazioni attive di accedere alla rete in modo periodico e con tempo di attesa prevedibile (pari al tempo impiegato dal *token* per compiere un giro sull'anello logico) e contemporaneamente fa sì che sia possibile connettere al bus anche dispositivi con limitate capacità di elaborazione (come nodi passivi).

Servizi.

I servizi offerti al Livello Application, descritti nella tabella seguente, sono
di supporto al traffico aperiodico (SDA, SDN ed SRD),
e di gestione del traffico periodico (CSRD).

Servizi di Livello Data Link.

Servizio	Descrizione
SDA (Send Data with Acknowledge)	Trasmissione di dati da un nodo attivo verso un nodo attivo o passivo. Il destinatario dà conferma di corretta ricezione
SDN (Send Data with No Acknowledge)	Trasmissione di dati o messaggi da un nodo attivo verso: • un altro nodo (attivo o passivo). • un gruppo di nodi (servizio *multicast*). • tutti i nodi (servizio *broadcast*). Non c'è conferma da parte dei nodi riceventi.
SRD (Send and Request Data)	Un nodo attivo trasmette dati ad un secondo nodo attivo o passivo e contemporaneamente richiede l'invio di dati.
CSRD (Cyclic Send and Request Data)	Un nodo attivo trasmette ciclicamente dati ad un altro nodo attivo o passivo e contemporaneamente richiede l'invio di dati.

Per il servizio CSRD, ogni stazione attiva conserva una lista (**Lista di polling**) contenente l'elenco dei dispositivi a cui richiedere dati in occasione dei turni ciclici di possesso del *token*. Il meccanismo dello scambio ciclico dei dati viene svolto a Livello Data Link ed è quindi trasparente al Livello Application.
Il Livello Data Link si occupa anche di rilevare e/o correggere gli errori. La codifica usata permette una distanza di Hamming pari a 4 e permette di rilevare errori fino a tre bit e di correggere tutti gli errori su un bit.

14.5.5.3 Livello Application.

Il Livello Application consente la manipolazione di **oggetti di comunicazione**. Ogni stazione possiede un dizionario (OD = ***Object Dictionary***) che descrive tutti gli oggetti di sua pertinenza.
La struttura del dizionario OD è la seguente.

- Intestazione - contiene informazioni sulla struttura del dizionario stesso.
- Lista statica dei tipi - contiene l'elenco dei tipi supportati e relative strutture dati. I tipi gestiti da Profibus sono variabili semplici, array e record, aree di memoria, eventi.
- Dizionario statico degli oggetti - contiene l'elenco degli oggetti di comunicazione statici, definito in fase di configurazione e non modificabile in esercizio..
- Lista dinamica di variabili - contiene l'elenco di tutte le variabili conosciute. Questo elenco è modificabile anche in esercizio.
- Lista dinamica dei programmi - contiene l'elenco delle procedure richiamabili.

Esistono quindi due categorie fondamentali di oggetti di comunicazione.

- **Oggetti dinamici:** possono essere creati, cancellati o modificati durante il funzionamento del sistema. A questa categoria appartengono: Programmi e Liste di variabili (semplici o array o record).

- **Oggetti statici:** sono predefiniti in sede di configurazione della rete. A questa categoria appartengono: Variabili semplici, Array, Record, Aree di memoria ed Eventi.

Il Livello Application è diviso in due sottolivelli (*sublayer*).
FMS (*Fieldbus Message Specification*), che fornisce i servizi che consentono di manipolare gli oggetti di comunicazione.
LLI (*Lower Layer Interface*), che funge da interfaccia tra il Livello 2 ed il Livello 7 (sostituendo così i livelli dal 3 al 6).

Sottolivello FMS.

FMS mette a disposizione:
1. Servizi che consentono di stabilire e rilasciare connessioni logiche tra due processi residenti in nodi diversi.
2. Servizi di accesso a variabili semplici, record ed array.
3. Servizi che permettono il trasferimento di intere aree di memoria.
4. Servizi di controllo dell'esecuzione di processi.
5. Servizi di gestione degli eventi: rendono possibile l'invio di messaggi d'allarme e sono senza conferma.
6. Servizi che consentono l'identificazione delle stazioni collegate e l'invio di rapporti sul loro stato.
7. Servizi di gestione del dizionario degli oggetti: servono a creare, cancellare, leggere o modificare oggetti all'interno del dizionario contenuto in ogni nodo.

Sottolivello LLI.

LLI si occupa di:
1. Controllo del traffico (in sostituzione del Livello Transport).
2. Creazione, controllo e rilascio senza perdita di dati, di una connessione tra due processi di Livello Application.
3. Conversione dei Servizi di Livello Data Link in quelli fornibili da FMS, come la gestione *master-slave* dell'accesso al bus, il *polling* ciclico e le trasmissioni *multicast* e *broadcast*.

14.5.5.4 Relazioni di comunicazione.

PROFIBUS permette la distibuzione delle funzioni di controllo tra più nodi di rete.
Due processi residenti in nodi diversi possono comunicare tra loro attraverso canali logici detti **relazioni di comunicazione** (*communication relationship*). Ogni stazione conserva una lista (**CRL** = *Communication Relationship List*) che riporta le caratteristiche di tutte le connessioni che coinvolgono la stazione stessa. Tale lista viene caricata in fase di configurazione del sistema e non può più essere modificata in esercizio.
Le relazioni di comunicazione possono essere suddivise in due categorie fondamentali.

- **Relazioni connection oriented:** prima che possa aver luogo lo scambio di dati richiedono che sia stabilita una connessione logica fra due processi. Permettono l'invio di messaggi con o senza conferma di ricezione.
  Queste relazioni si suddividono a loro volta in:
  Connessioni aperte: il partner ed i parametri del collegamento vengono stabiliti dinamicamente durante il funzionamento della rete.

Connessioni definite: il partner ed i parametri del collegamento sono specificati in fase di configurazione. In questo modo per aprire un canale di scambio delle informazioni è sufficiente attivare la connessione predefinita, migliorando così l'efficienza del sistema.

- **Relazioni Connectionless:** non richiedono una fase di inizializzazione della connessione. Ogni *frame* di dati contiene infatti l'indirizzo del mittente e del destinatario, quindi maggiore flessibilità a costo di un maggiore *overhead*.

14.5.5.5 Modalità di implementazione

Esistono diversi modi di implementare le specifiche PROFIBUS. In linea di massima esistono tre possibilità, qui di seguito elencate in ordine crescente di costo, complessità e prestazioni ottenibili.

- Implementazione compatta: un unico microcontrollore o microprocessore svolge sia le funzioni applicative di elaborazione che quelle di comunicazione. E' questo il caso tipico delle stazioni passive.
- Le operazioni di comunicazione vengono affidate ad un processore dedicato (ad esempio Intel 80186 o NEC V25)
- Un ASIC o un microprocessore specializzato (ad es. Motorola 68302) si occupa dell'implementazione dei soli Livelli 1 e 2, mentre un secondo processore è dedicato al Livello 7.

14.5.6 FIP e WorldFIP

Il FIP (*Factory Instrumentation Protocol*) è nato dalla collaborazione di industrie e istituti accademici francesi a cui si sono affiancati anche enti italiani.

Per ragioni di semplicità e di economicità la specifica FIP richiede l'implementazione dei soli livelli Physical (1), Data Link (2) ed Application (7).
WorldFIP è una recente evoluzione di FIP, da cui si differenzia principalmente per l'adozione a livello fisico delle caratteristiche definite dalla norma IEC 1158-2 (*Fieldbus Standard for use in industrial control systems - Part 2: Physical layer specification and service definition*).

Caratteristiche salienti.

- La trasmissione di messaggi è compatibile con MMS (Manifacturing Message Service) dello standard MAP.
- Si utilizza un particolare schema produttore-distributore-utilizzatore.
- Adotta particolari meccanismi per verificare la consistenza temporale e spaziale dei dati, con impostazione *time-driven real-time*.

14.5.6.1 Livello Physical.

Topologia.

Viene utilizzata la topologia a bus. In fig.14.13 è riportato un esempio di realizzazione:

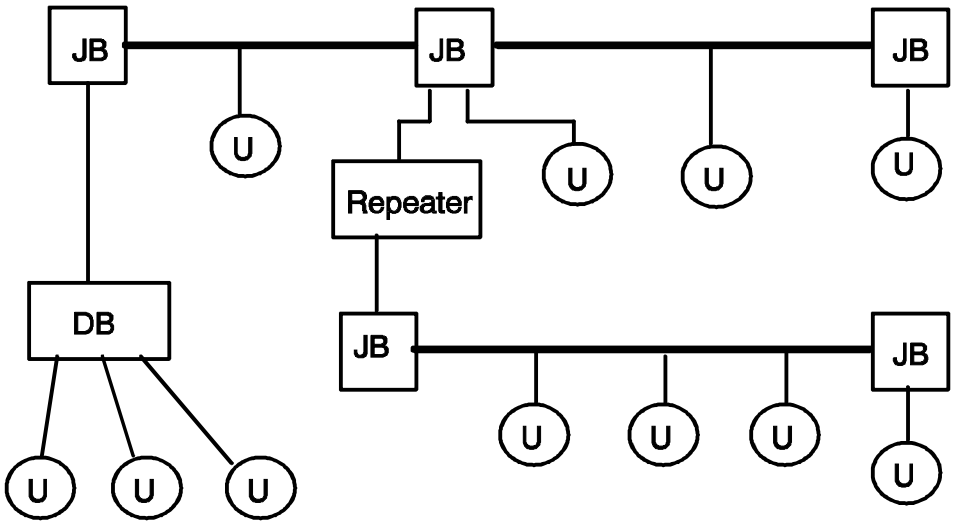


Fig. 14.13 - Esempio di architettura fisica di rete FIP.

Un fieldbus FIP è costituito da uno o più tratti principali (evidenziati in figura con la linea più spessa) collegati tramite ripetitori.

U sono i nodi utilizzatori del bus.

JB (*Junction Box*) indica una scatola di derivazione passiva del bus, con due o più "prese".

DB (*Diffusion Box*) è attiva e funge da concentratore con un collegamento a stella tra diversi utilizzatori **U**. I dispositivi DB consentono la connessione anche di sistemi non in grado di gestire direttamente la connessione al bus FIP (*retrofitting*).

Mezzo trasmissivo.

Il mezzo trasmissivo di FIP può essere il doppino attorcigliato schermato oppure la fibra ottica. Nel primo caso si possono utilizzare le velocità di trasferimento di 31.25kbit/s, 1Mbit/s (la più usata) e 2.5Mbit/s. Per le fibre ottiche la velocità è di 5Mbit/s.

Codifica.

FIP adotta il Codice Manchester (v. fig. 14.14), nel quale è prevista una commutazione del segnale al centro di ogni finestra temporale di bit. Consente quindi di effettuare particolari segnalazioni inserendo delle violazioni (mancate commutazioni)

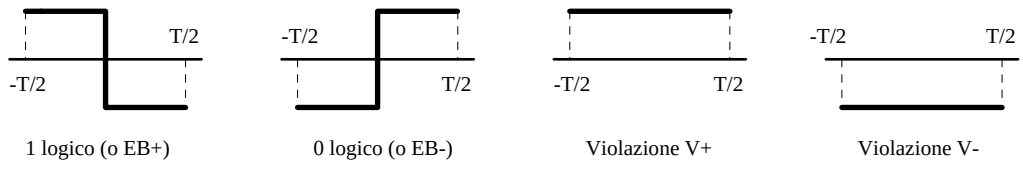


Fig. 14.14 - Finestre temporali di bit con codifica in banda base di FIP

Specifiche elettriche.

Per lo standard elettrico sono previste tre diverse classi.

1. CH (*High Power*, per lunghe distanze)
2. CM (*Medium Power*)
3. CL (*Low Power*, per brevi distanze e/o uso in ambienti a rischio di esplosione).

Nella tabella seguente sono riportate le specifiche fondamentali per ricevitori e trasmettitori (Tx):

Caratteristiche elettriche.

Classe	CH	CM	CL
Livello di uscita Tx - (V picco-picco)	5.5 ... 9	2.8 ... 9	1.5 ... 2
Livelli di uscita Tx a riposo (V picco-picco)	<30	<30	<30
Disturbi Tx (mV efficaci)	<10	<10	<5
Livello minimo per la ricezione (mV picco-picco)	700	700	200
Livello massimo per la non ricezione (mV picco-picco)	280	280	100
Livello massimo del rumore (mV picco-picco)	60	60	30

Formato di una trama WorldFIP

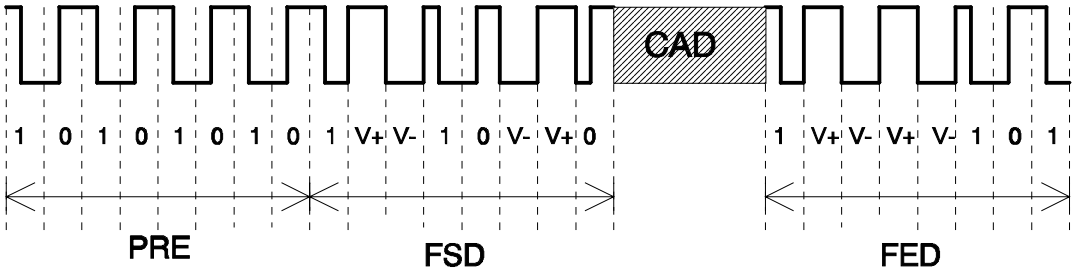


Fig.14.15 - Preambolo e delimitatori di una trama WorldFIP.

- **PRE** è il preambolo che serve ai ricevitori per agganciare in fase il loro clock a quello del trasmettitore.
- **FSD** (*Frame Start Delimiter*) contiene la sequenza che consente al Livello Data Link di riconoscere la parte dati (CAD). Le violazioni di codice, appositamente inserite, consentono un riconoscimento inequivocabile, diverso da qualsiasi sequenza di bit di dati.
- **CAD** è il testo del messaggio, senza violazioni.
- **FED** (*Frame End Delimiter*) consente al Livello Data Link di riconoscere la fine del campo dati.

Il livello fisico aggiunge quindi 24 simboli ai "dati utili" con un *overhead* temporale per ogni messaggio di 24 μs se la velocità è quella standard di 1 Mbit/s.

14.5.6.2 Livello Data Link.

Servizi trasmissivi.

Il Livello Data Link offre due categorie fondamentali di servizi:

- Scambio di variabili.
- Trasferimento di messaggi.

I servizi possono essere svolti

- ciclicamente (con periodicità prestabilita in fase di configurazione)
- oppure in modo asincrono su richiesta di uno dei nodi di rete.

Il traffico **periodico** viene gestito in modalità **Unacknowledged Connectionless**, mentre i messaggi **aperiodici** possono essere trasmessi anche in modalità **Acknowledged Connectionless.**

Indirizzamento.

Il FIP prevede due distinti spazi di indirizzamento.

Variabili.

Ogni variabile condivisa nell'ambito della rete, è individuata in modo univoco da un identificatore di 16 bit, che accompagna sempre il valore della variabile stessa. In questo modo il trasferimento di una variabile può avvenire tra un produttore e tutti i consumatori interessati al valore trasmesso sulla rete.

Messaggi.

Lo scambio di messaggi può essere di diversi tipi: punto a punto, uno-a-molti oppure uno-a-tutti. Ogni messaggio contiene l'indirizzo del nodo mittente e quello del destinatario

A Livello Data Link di ogni nodo sono allocati, in sede di configurazione, dei buffer contenenti i valori delle variabili prodotte e/o consumate dal nodo stesso.
Sono disponibili servizi che consentono al Livello Application operazioni di lettura e scrittura locali di tali buffer.
E' possibile accedere alle variabili memorizzate a Livello Data Link anche in modo remoto (operazioni di Buffer Transfer), sotto la regia del *Bus Arbitrator*, come descritto nel seguito.

Controllo dell'accesso al mezzo trasmissivo.

L'accesso alla rete è regolato da un **unico** nodo *master* detto **arbitro di bus** (BA = *Bus Arbitrator*). Questa stazione gestisce sia il traffico ciclico sia il traffico aperiodico. Ognuna delle stazioni connesse al bus può a priori svolgere la funzione di arbitro, ma non possono essere attivi nello istante due o più arbitri.
In fase di **configurazione** della rete nell'arbitro viene caricata la lista contenente gli identificatori delle variabili i cui valori devono essere rinfrescati in modo periodico. Durante il **funzionamento** l'arbitro di bus emette sulla rete, con la periodicità richiesta, dei messaggi di richiesta di tali identificatori. I produttori delle variabili così identificate rispondono trasmettendo i relativi valori, che possono essere prelevati da tutti i nodi consumatori interessati.
Nelle **finestre temporali** rimaste **libere** dal traffico periodico l'arbitro di bus inserisce poi i messaggi di richiesta per i trasferimenti asincroni di messaggi o di dati, di cui qualche stazione gli abbia fatto esplicita richiesta.

Nel seguito viene presentato un semplice esempio con 6 variabili di rete.

Tabella di arbitraggio

Variabile	Periodicità	tipo	Tempo di comunic.
A	5	`int_8`	170
B	10	`int_16`	178
C	15	`string_32`	418
D	20	`float_32`	194
E	20	`unsigned_32`	194
F	30	`string_16`	290

I tempi di comunicazione sono calcolati per la velocità di 1 Mbit/s, e si riferiscono alle transazioni complete, costituite dalle seguenti fasi.

1. L'arbitro manda la trama di richiesta della variabile.
2. Trascorre il tempo di *Turnaround*, che deve essere compreso tra 10 μs e 70 μs, e qui è ipotizzato di 20 μs.
3. Il nodo produttore emette la trama contenente il valore della variabile richiesta, e i livelli Data Link di tutti i nodi consumatori della variabile scrivono tale valore nel corrispondente buffer di consumo.

Nella figura seguente (fig.14.16)è rappresentata, su un asse temporale orizzontale, una possibile sequenza (macrociclo) di cicli elementari (in questo caso di 5 ms ognuno) con l'elenco delle variabili divulgate in ogni ciclo.
Il numero di cicli elementari che compongono un macrociclo è dato dal Minimo Comune Multiplo delle periodicità diviso per il Massimo Comun Divisore.

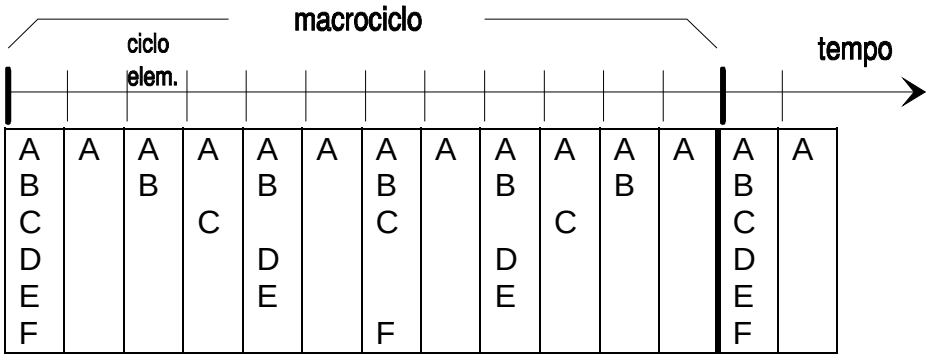


Fig. 14.16 - Sequenza temporale di cicli elementari di trasferimenti di variabili cicliche.

Si noti che i cicli elementari sono tutti temporalmente compatibili, dato che nessuno richiede comunicazioni che globalmente superino i 5 ms di durata del ciclo elementare stesso.
In ogni ciclo elementare tutto il tempo rimasto libero dalle comunicazioni periodiche è disponibile per il traffico aperiodico a richiesta, di variabili e di messaggi, come rappresentato in fig.14.17.

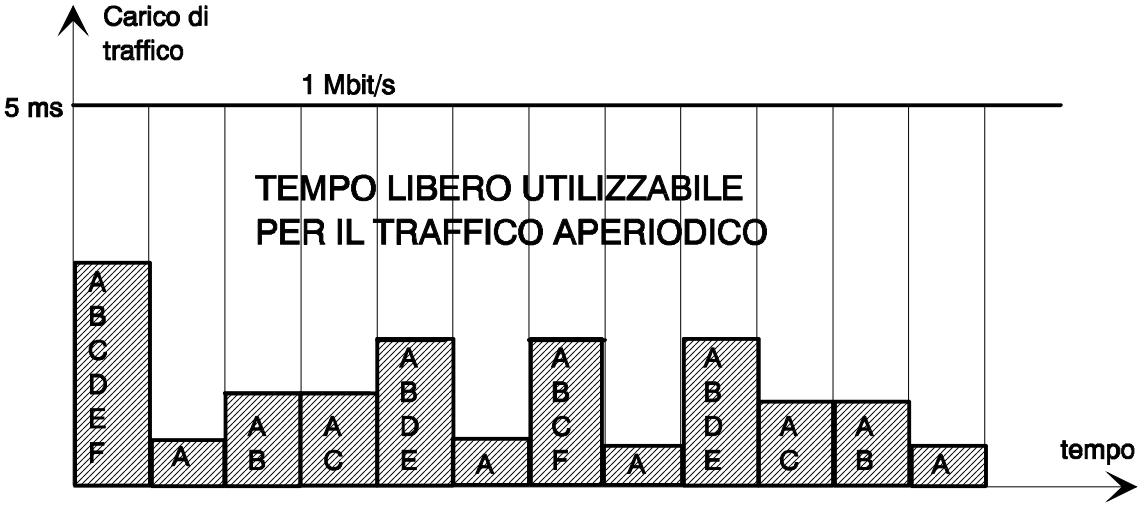


Fig. 14.17 - Esempio di andamento temporale del carico di traffico periodico (tratteggiato) in un macrociclo e tempi lasciati disponibili per il traffico aperiodico.

Richieste di trasferimenti aperiodici.

Questi trasferimenti corrispondono generalmente ad eventi sporadici prodotti dai vari nodi, ed utilizzano gli intervalli dei cicli elementari non utilizzati dal traffico periodico.
I trasferimenti si svolgono nelle seguenti fasi.

1. Il nodo arbitro chiama una variabile periodica, ad es. A.
2. Il nodo produttore di A abbia appena prodotto anche una variabile aperiodica da trasferire ai nodi consumatori. In occasione dell'invio del valore di A il nodo inserisce, nel campo di controllo della trama, anche una richiesta aperiodica, specificando priorità *Urgent* o *Normal*.
3. L'arbitro si annota l'identificatore di A (si noti, non il nome del nodo) nella coda (urgente o normale) di richieste aperiodiche.
4. Nella finestra temporale per trasferimenti aperiodici l'arbitro chiede al nodo produttore di A che esso trasmetta la sua richiesta.
5. Il nodo produttore di A trasmette una lista di identificatori (da 1 a max 64).
6. L'arbitro riceve ed inserisce in un'apposita coda tale lista.
7. Prima o poi, compatibilmente con il carico di rete (cioè sempre nei tempi lasciati liberi dal traffico periodico), l'arbitro chiamerà gli identificatori in coda.

Quindi perchè un nodo possa inoltrare richieste di trasferimenti aperiodici deve essere produttore di almeno una variabile inserita nella tabella dei trasferimenti periodici, e la periodicità di tale variabile determina il massimo ritardo dei trasferimenti.

Trasferimenti aperiodici di messaggi Unacknowledged

Il Livello Data Link è in grado di gestire questi trasferimenti sia nella modalità *point-to-point*, sia nella modalità *broadcast*. Il meccanismo è molto simile a quello visto per i trasferimenti di variabili, salvo che per i trasferimenti di messaggi si indirizzano i nodi..

Trasferimenti aperiodici di messaggi Acknowledged

Questi trasferimenti sono gestiti dal Livello Data Link solo nella modalità *point-to-point*. I messaggi contengono una numerazione modulo 2 (cioè di un bit), per consentire al ricevente il riconoscimento di perdite o duplicazioni di messaggi.
Quando il ricevente del messaggio emette la trama di *acknowledge*, il mittente informa l'arbitro con l'emissione di una trama di fine transazione.
In assenza di *acknowledge* entro un dato *time-out* il mittente ripete la trasmissione da 0 a 2 volte (parametro scelto in sede di configurazione).

14.5.6.3 Livello Application.

Servizi offerti ai processi applicativi

1. Lettura/scrittura locale o remota di valori di variabili: le operazioni locali accedono ai buffer di produzione e di consumo del Livello Data Link, e non richiedono ovviamente l'uso del canale trasmissivo.
Le operazioni remote, invece, comportano l'attivazione del Livello Data Link che esegue autonomamente i trasferimenti ciclici, e per quelli aperiodici chiede il trasferimento di dati sul bus durante le finestre temporali dedicate al traffico aperiodico, come visto precedentemente.

2. Sincronizzazione di produzione e consumo delle variabili:
Questi servizi sono utilizzabili opzionalmente per distribuire in modo sincrono (cioè con cadenza regolare) valori di veriabili di stato prodotte in modo asincrono (cioè con cadenza arbitraria e generalmente più elevata).

Per ottenere ciò, a Livello Application ogni variabile prodotta e consumata è immagazzinata in due aree di memoria di cui una è accessibile solo ai processi interni al nodo stesso (**buffer privato**), mentre l'altra (**buffer pubblico**) può essere manipolata solo dal *Data-Link service provider*.
L'invio sul bus FIP di un'apposita variabile di sincronizzazione, di tipo periodico, agisce come un "clock" che sincronizza la produzione effettuando la ricopiatura del buffer privato nel buffer pubblico, mentre effettua la sincronizzazione del consumo con l'operazione di copiatura opposta.
In sede di configurazione si decide quali variabili devono essere resincronizzate e a quale variabile ciclica di sincronizzazione si appoggiano.

3. Indicazione di avvenuta ricezione o trasmissione di variabili.

In sede di configurazione, per ogni variabile di rete, si può chiedere che il Livello Application sia avvertito della ricezione o della trasmissione da parte del Livello Data Link di tale variabile. Ciò consente di sincronizzare le parti di un processo distribuito, residenti in nodi di rete differenti, considerando l'arrivo o l'emissione di alcune variabili come eventi di sincronizzazione.

4. Informazione sulla freschezza (freshness) dei dati.

Quando un processo applicativo legge il valore di una variabile, esso può contemporaneamente essere informato sullo stato di "freschezza" (*freshness*) di tale valore.
A tal fine ad ogni variabile prodotta o consumata è associata anche l'informazione sulla sua qualità dal punto di vista del rispetto dei vincoli temporali prefissati in sede di configurazione. Tale informazione è costituita da variabili booleane appartenenti a due categorie fondamentali: gli stati di **refreshment** e di **promptness**.
Promptness e *refreshment* possono essere **asincroni, sincroni** o **puntuali**.

- **Refreshment asincrono.**
  Questo stato indica la tempestività del produttore di una variabile e viene gestito dal Livello Application presso il nodo **produttore**.
  Si supponga che in sede di configurazione si stabilisca che il valore di una certa variabile è considerabile "fresco" se è stato prodotto da meno di un tempo Tr.
  Il processo produttore produce con il suo ritmo i valori di tale variabile *Vi*. Il buffer pubblico relativo alla variabile stessa viene quindi aggiornato con operazioni di scrittura locale asincrona. Il Livello Application del nodo produttore ad ogni nuova produzione rilancia il timer a tempo Tr associato a *Vi*. Se il timer scade prima della nuova produzione il Livello Application pone a *false* il valore dello stato di *refreshment*. Lo stato di *refreshment* asincrono diventa quindi *false* ogni volta che tra due successive operazioni di scrittura nel buffer pubblico intercorre un tempo maggiore del periodo di produzione Tr previsto per *Vi*.
  Questa informazione è utile per variabili di stato oggetto di rapporti *writer-readers*.

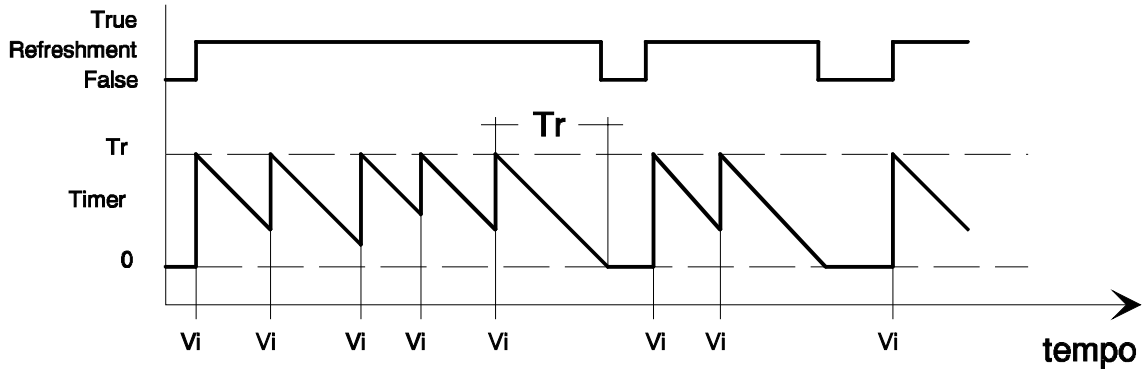


Fig. 14.18 - Andamento temporale del Refreshment Asincrono. **Vi** sono gli istanti di produzione di nuovi valori.

- **Refreshment sincrono.**
  Questo stato indica la tempestività del produttore di una variabile destinata a trasferimenti sincroni. In sede di configurazione si stabilisce il tempo Tr del periodo massimo previsto di aggiornamento di una variabile *Vi*, e si indica una variabile di sincronizzazione *Vs*, periodica (con periodo Ts > Tr) Ogni volta che *Vs* viene ricevuta (con ciò provocando la copiatura sul buffer pubblico del valore di *Vi* e del relativo stato di *refreshment*) un timer ad essa associato viene riportato al valore iniziale Tr e lo stato di *refreshment* diventa *false*. Lo stato di *refreshment* sincrono diventa *true* se il timer non si è ancora azzerato in corrispondenza della produzione di un nuovo valore per *Vi*.

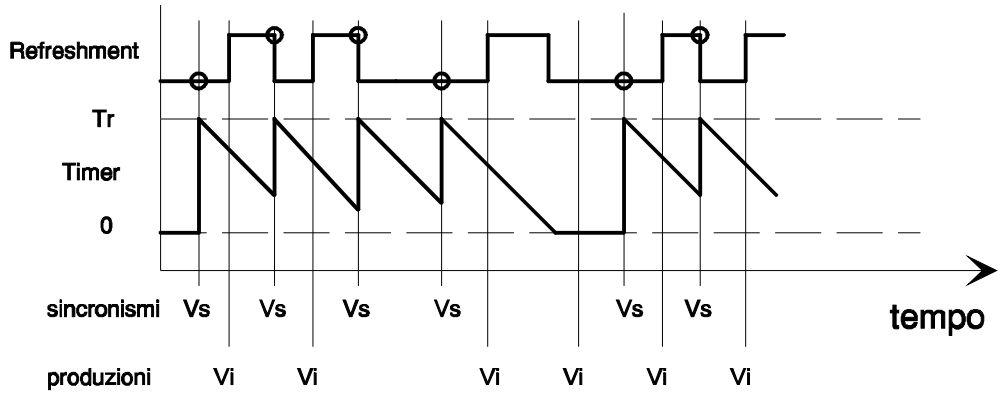


Fig. 14.19 - Andamento temporale di Refreshment Sincrono. **Vi** sono gli istanti di produzione di nuovi valori. **Vs** sono gli istanti di trasferimento sincrono, e lo stato di refreshment associato è indicato dai circoletti.

Lo stato di refreshment sincrono è *true* quando il nodo riesce a rinfrescare *Vi* con la cadenza corretta e in modo sincrono rispetto alla ricezione della variabile di sincronizzazione. Questa informazione è utile per eventi temporizzati, e da consumare in corrispondenza dell'invio di *Vs*.

- **Promptness asincrono.**
  Questo stato indica la tempestività del meccanismo di trasferimento dei valori prodotti, e viene elaborato dal Livello Application del nodo **consumatore** della variabile *Vi* a cui è associato. Anche in questo caso si suppone di aver definito in sede di configurazione il massimo ritardo Tr con la quale *Vi* deve essere aggiornata da una nuova trasmissione in rete. Ogni volta che *Vi* viene aggiornata dalla rete lo stato di *Promptness* asincrono diventa *true* e il timer a decremento associato viene riportato al valore iniziale Tr. Se non si riceve un nuovo valore di *Vi* prima della scadenza del periodo di produzione della variabile stessa (timer arrivato a zero), lo stato di Promptness asincrono viene posto uguale a *false*.

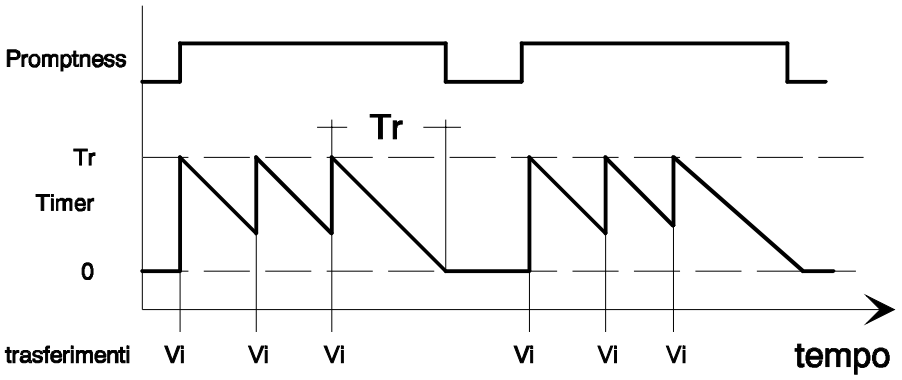


Fig. 14.20 - Andamento temporale del Promptness Asincrono.

5. Servizi per liste di variabili Sono disponibili servizi che danno informazioni sulla omogeneità e sulla validità dei dati contenuti nelle liste di variabili consumate o prodotte. Tali informazioni sono rappresentate da variabili booleane, di alcune delle quali qui di seguito sono riportate le descrizioni.

- **Coerenza temporale di produzione.** Il Livello Application fornisce anche il servizio di lettura di un'intera lista di variabili. Per una lista di variabili $\{v_1, v_2, .., v_n\}$ lo stato di coerenza temporale in produzione viene ottenuto come il prodotto logico (*and*) tra gli stati di ***refreshment*** relativi a ciascuna delle variabili, e dà quindi un'indicazione sulla validità delle stesse dal punto di vista del rispetto dei vincoli temporali in fase di produzione.
- **Coerenza temporale di consumo.** Questo stato viene invece calcolato come il prodotto logico tra gli stati di ***promptness*** relativi alle singole variabili della lista., e serve a stabilire se la rete abbia rinfrescato tali variabili alle frequenze corrette.
- **Coerenza spaziale.** Viene elaborato dai Livelli Application di tutti i nodi che possiedono le copie di una stessa lista $\{v_1, v_2, .., v_n\}$ di variabili, e vale *true* quando tali nodi dispongono di copie uguali di $\{v_1, v_2, .., v_n\}$.

14.5.7 LonWorks

LonWorks è un *trademark* della Echelon Corporation (USA), che designa un pacchetto di prodotti proposti per realizzare applicazioni di controllo distribuito in ambito "automazione di edifici" (*building automation*) e altre applicazioni industriali in cui i requisiti *real-time* non siano troppo spinti.
I componenti e gli strumenti di LONWORKS sono i seguenti.

NEURON CHIP. Sono circuiti integrati che costituiscono i nodi della rete e integrano un sistema a tre processori con il "firmware" per la gestione del protocollo LonTalk.

Componenti di controllo e di connettività che comprendono ricetrasmettitori (*transceiver*), moduli di controllo e *router*, adatti a gestire e collegare tratte di rete con diversi mezzi trasmissivi, quali doppini attorcigliati, onde convogliate, radio, ecc.

LONBUILDER. E' uno strumento SW che gira su PC e consente lo sviluppo e il debug di applicazioni.

NEURON C. E' un linguaggio di programmazione derivato da ANSI C, con alcune limitazioni ed alcune estensioni. In particolare le estensioni riguardano

l'introduzione di variabili ed eventi gestiti automaticamente dalla rete, e funzioni di libreria orientate a implementazioni *multitasking* distribuite.

LON MANAGER. E' un insieme di strumenti SW per istallazione, configurazione, diagnostica e manutenzione di reti LonWorks.

14.5.7.1 Struttura di rete

Un esempio di rete LonWorks è rappresentato in fig.14.21.

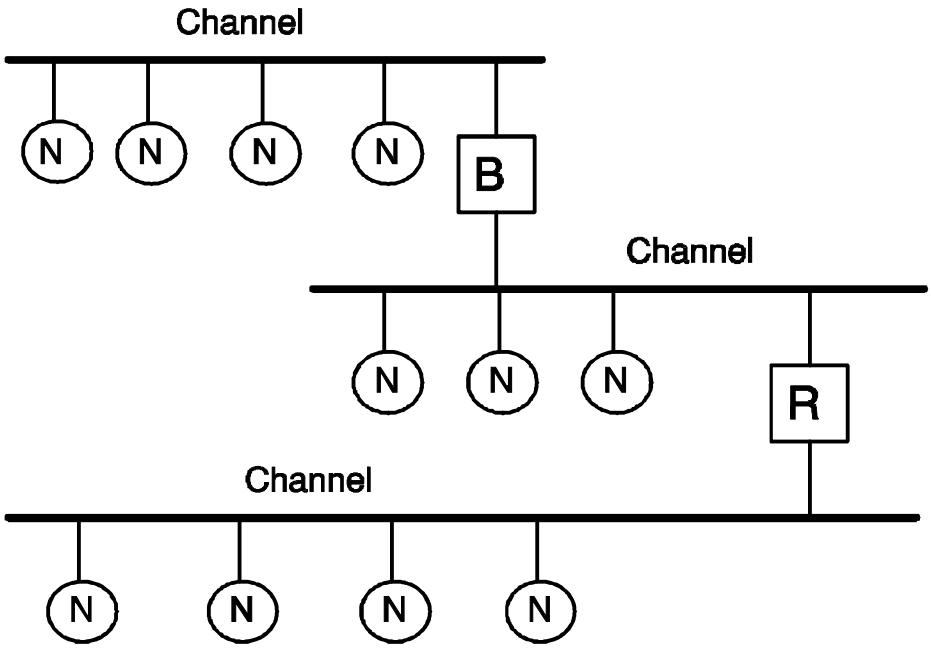


Fig. 14.21 - Esempio di rete LonWorks, con nodi N, Bridge B e repeater R.

Ogni **nodo N** è fisicamente connesso ad un ***channel***, cioè una tratta di rete basata su un determinato mezzo fisico (cavo, infrarossi, onde convogliate, ecc.).
Diversi *channel* sono collegati tra loro tramite dei ***router***, che sono dispositivi che possono assumere diverse funzionalità, tra cui quelle di *repeater* e *bridge*.
Un insieme di *channel* connessi tramite *bridge* **B** o *repeater* **R**, costituisce un ***segment***.
Tutti i nodi in un *segment* "vedono" tutti i pacchetti in esso trasmessi. Le velocità di trasmissione possibili dipendono dal mezzo fisico, mentre il numero di transazioni (pacchetti) al secondo (*throughput*) presenta una saturazione a circa 560 pacchetti/s dovuta alla potenza di calcolo dei nodi.

14.5.7.2 I Nodi di rete

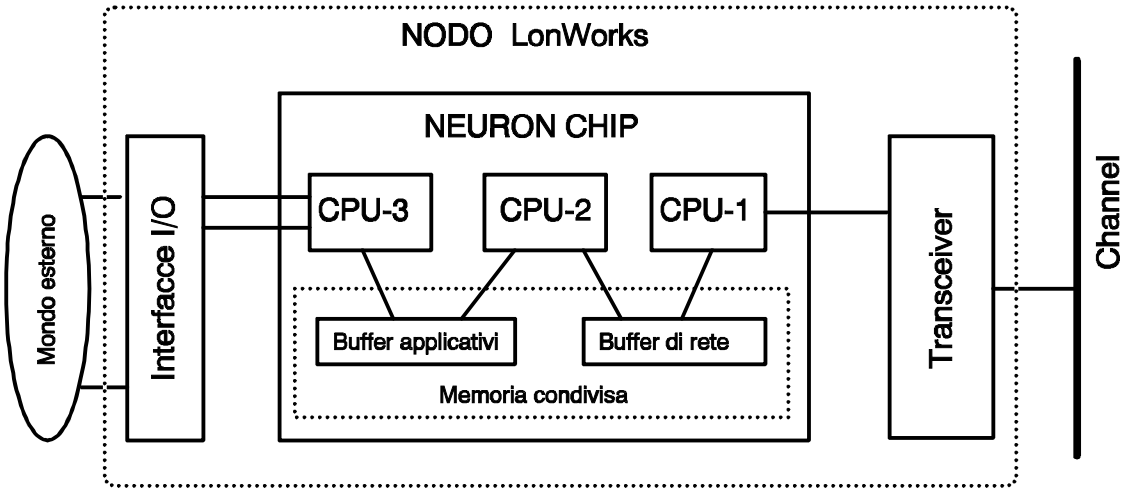


Fig. 14.22 - Struttura di un nodo LonWorks.

La struttura dei nodi di rete è rappresentata in fig.14.22, e comprende i seguenti elementi.

Interfacce di I/O applicativo. Queste possono ridursi a normali circuiti di interfacciamento, oppure possono essere basate su un microprocessore, più o meno convenzionale, che a sua volta si interfaccia con il mondo esterno.

Transceiver. (XCVR) E' un componente in grado di gestire il mezzo fisico di comunicazione. Nella "famiglia" LonWorks sono previsti circuiti per i seguenti mezzi fisici.

- Doppino attorcigliato, con standard elettrico RS-485
- Fibra ottica
- Cavo coassiale
- Comunicazione radio
- Comunicazioni a infrarossi, come i telecomandi domestici
- Onde convogliate. Si tratta di segnali a frequenza relativamente elevata (sul centinaio di KHz) sovrapposti alla tensione della rete di distribuzione della potenza elettrica, che assume il doppio ruolo di trasporto di potenza e di segnali.

NEURON CHIP. E' un integrato multiprocessore che contiene 3 CPU con registri separati ma che condividono ALU e indirizzamento alla memoria.
CPU-1 svolge le funzioni MAC (*Media Access Control*) interfacciandosi con il *transceiver*, e gestisce i livelli 1 e 2 del protocollo LonTalk, eseguendo programmi prefissati forniti dal costruttore (*firmware*).
CPU-2 svolge le funzioni *Network* coprendo i livelli 3, 4, 5 e 6 del protocollo LonTalk, eseguendo programmi prefissati forniti dal costruttore (*firmware*).
La CPU-2 scambia i dati con la CPU-1 tramite un'area di memoria condivisa che costituisce i *Network Buffers*, mentre comunica con la CPU-3 tramite gli *Application Buffers*.
CPU-3 svolge le funzioni *Application*, eseguendo i programmi scritti dall'utente (cioè il progettista applicativo) e le funzioni del sistema operativo da essi invocate. Il linguaggio proposto per questi programmi è Neuron C. Questa CPU gestisce anche l'I/O.

14.5.7.3 Il Protocollo LonTalk

Il protocollo è indipendente dal mezzo trasmissivo, consentendo quindi un'ampia scelta, anche nell'ambito della stessa rete, di diversi mezzi, come precedentemente accennato, ed utilizza tutti i 7 livelli ISO-OSI, come riportato sinteticamente nella tabella seguente.

	Livello OSI	Scopo	Servizi forniti	CPU
7	Application	Compatibilità applicativa	Gestione dei tipi standard di variabili di rete	CPU-3
6	Presentation	Interpretazione dei dati	Variabili di rete. Propagazione di trame esterne	CPU-2
5	Session	Azioni remote	Request/Response, autenticazione, gestione rete.	CPU-2
4	Transport	Affidabilità di comunicazione	Acknowledged e Unacknowledged. Unicast e multicast. Ordinamento e riconoscimento di duplicati.	CPU-2
3	Network	Indirizzamento della destinazione	Indirizzamento, informazioni di routing.	CPU-2
2	Link	Accesso al mezzo e composizione delle trame.	Trame, codifica dei dati, verifica errori con CRC, evitare e rilevare collisioni.	CPU-1
1	Physical	Connessione elettrica	Interfacciamento specifico col mezzo e tecniche di modulazione.	CPU-1 e transceivers

Sono previste le 12 **velocità** di trasmissione (ognuna il doppio della precedente) comprese tra 600 bit/s e 1250 Kbit/s.
I **pacchetti** sono mediamente di 10 . . 16 byte, e possono arrivare ad un massimo di 255 byte.

Indirizzamento in rete.

Sono previste le seguenti entità indirizzabili, che formano una gerarchia di insiemi di nodi.

Domain. - Collezione logica di nodi appartenenti ad uno o più *channel*. Costituisce una "rete virtuale", in quanto i nodi possono comunicare solo con altri nodi dello stesso *domain*. Ogni nodo può essere membro di uno o due *domain*. Un nodo appartenente a due domain può fungere da *gateway*.
L'identificatore di *domain* può essere costituito da 0 (*domain* unico che non ha quindi necessità di essere identificato), 1, 3 o 6 byte.

Subnet. - E' un gruppo logico di nodi (fino a 127), che possono appartenere anche a diversi *channel*. In un *domain* si possono avere fino a 255 *subnet*, il cui identificatore è quindi di 1 byte. Una *subnet* può attraversare dei *bridge*, ma non dei *router* (fig.14.23).

Node. - I nodi di un domain possono essere fino a 127*255=32.385.

I nodi, anche di diverse *subnet*, possono essere raggruppati in ***group***. Ogni nodo può essere configurato per appartenere fino a 15 diversi *group*.
Ad ogni nodo viene assegnato un indirizzo logico di 1 byte nell'ambito della *subnet*.
Ogni nodo possiede poi un proprio **identificatore fisico univoco** di 48 bit (6 byte) che gli viene assegnato in sede di fabbricazione, detto NEURONID.

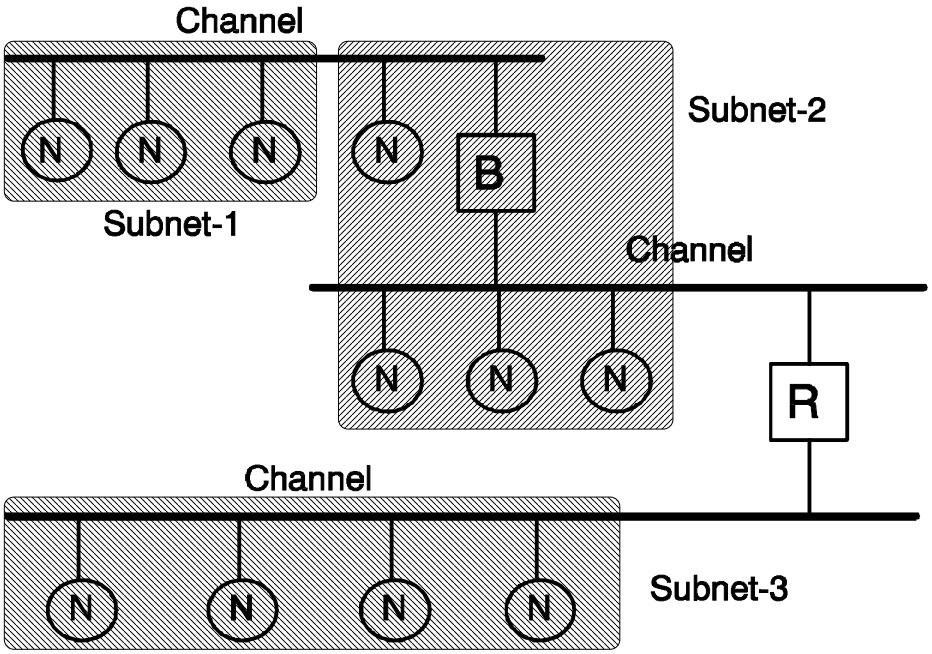


Fig. 14.23 - Esempio di rete con due *channel* collegati a formare un unico segmento dal *bridge* B, ed un terzo channel collegato al *domain* da un *repeater* R.

In questa struttura gerarchica gli indirizzamenti collettivi o singoli, sono espressi con una specie di "*path*", come esemplificato nella tabella seguente.

Indirizzo specificato	**Nodi indirizzati**
Domain, Subnet=0	Tutti i nodi nel Domain (broadcast)
Domain, Subnet	Tutti i nodi nella Subnet (multicast)
Domain, Subnet, Node	Singolo nodo logico (unicast)
Domain, Group	Tutti i nodi in Group (multicast)
Domain, Subnet, NeuronID	Specifico nodo fisico (unicast)

Servizi sui messaggi

I quattro tipi base di servizi sono i seguenti.

ACKD - *Acknowledged* - Il messaggio è inviato *unicast* o *multicast*, e si attende un *ack* da ogni nodo destinazione. Se entro un *time-out* non sono stati ricevuti tutti gli *ack*, il mittente ripete la trasmissione. Il tempo e il numero di tentativi sono stabiliti in sede di configurazione. Gli *ack* sono generati direttamente a livello rete, senza interessare il livello applicazione.

REQUEST - *Request/Response* - Simile al precedente, ma la risposta (*ack*) è generata a livello applicazione, e può quindi contenere anche dei dati, consentendo così di realizzare funzioni del tipo "*remote procedure call*" o applicazioni "*client/server*".

UNACKD_RPT - *Unacknowledged Repeated* - Il messaggio viene inviato *unicast* o *multicast* un numero prefissato di volte. In questo modo si ha bassa probabilità che i riceventi non ricevano correttamente almeno una copia del messaggio, senza il sovraccarico di rete di tutte le risposte *ack*.

UNACKD - *Unacknowledged* - Come il servizio precedente, ma con una sola trasmissione. Questo servizio minimizza il carico di rete, ma le applicazioni devono accettare occasionali perdite di messaggi (ad es. informazioni cicliche di stato).

Si tralasciano in questa sede i dettagli di formato delle trame e le tecniche per evitare/risolvere le collisioni.

14.5.7.4 Il modello di programmazione

La programmazione è basata sul linguaggio Neuron C, che costituisce un sottoinsieme di ANSI C, ottimizzato per la CPU-3 di Neuron Chip (ad es. `int` è di 8 bit e `long` è di 16 bit), con interessanti estensioni orientate alle applicazioni di automazione distribuita.
Timer SW - Il programma applicativo può vedere fino a 15 timer di tipo `unsigned long` (16 bit) con granularità selezionabile a 1 s o a 1 ms. Gli eventi generati dai timer possono attivare processi applicativi (v. seguito).
Variabili di rete - Il programma applicativo può dichiarare una speciale classe (`network`) di variabili statiche, dette variabili di rete, che possono essere di classe `input` (lette) o `output` (scritte). Queste variabili possono essere fino a 62 in ogni singolo nodo e su di esse la rete realizza automaticamente un rapporto *writer readers* distribuito. Il nodo che ha dichiarato una variabile di rete come `output` può assegnare ad essa nuovi valori che vengono comunicati a tutti i lettori che hanno dichiarato tale variabile come `input`.
Il collegamento (*binding*) delle variabili di rete viene effettuato mediante uno strumento di sviluppo (ad es. LonBuilder) che produce le tabelle di collegamento da inserire in EEPROM e usate in esercizio.
Messaggi espliciti - Messaggi contenenti fino a 228 byte di dati, con formato scelto dal programmatore applicativo, possono essere trasmessi in rete con semplici chiamate di funzioni di libreria, previste per i vari tipi di servizi (con o senza *ack*, ripetuti, con priorità, ecc.).
Oggetti di I/O - Sono previsti particolari tipi (in aggiunta ai normali tipi del C) adatti per dichiarare come variabili i valori dei segnali dei piedini di I/O. Esempi: `bit, byte, nibble`.
Varie funzioni di libreria consentono una facile gestione di operazioni di I/O.
Eventi - Sono previsti eventi predefiniti ed eventi definiti dal programmatore.
Esempi di eventi **predefiniti** sono:

cambiamento di stato di un pin di ingresso
cambiamento di valore di una variabile di rete
scadenza di un timer
ricezione di un messaggio.

Gli eventi **definibili** dal programmatore sono qualsiasi espressione che sia interpretabile come generatrice di un valore *true* o *false* (in C qualunque espressione aritmetica e/o logica).
Gli eventi sono citati in clausole con la sintassi:

```
when (<evento>) <task_body>
```

gestite dallo *scheduler* multitask.

14.5.7.5 Lo Scheduler

La gestione dei *task* è del tipo *event driven,* ed i task sono il corpo di clausole

```
when (<evento>)
```

con la normale sintassi delle istruzioni composte, senza la possibilità di ritornare valori con istruzioni `return ()`.
Le clausole non possono essere annidate, ma si può specificare che si tratta di task prioritari.
Lo scheduler valuta gli eventi citati nelle clausole di tipo `priority` ogni volta che ha il controllo della CPU, mentre tra le clausole di task non prioritari lo scheduling opera con la politica *round robin*.

14.5.8 Altri Bus di Campo

Per ulteriori approfondimenti ed informazioni su interessanti bus proprietari (CAN, D2B, I2C, ecc.) si rimanda ai testi citati in bibliografia, in particolare ANIPLA e IEEE-Network.

14.6 ALTRE LETTURE

D.M. Auslander, C.H. Tham
Real-time software for control. Program examples in C.
Ed. Prentice-Hall 1990

Baumgartner, Kuishewsky, Wieding (SIEMENS)
CIM - Considerazioni di base.
Ed. Tecniche nuove 1989

Inquadra le problematiche generali del CIM.
Il capitolo 4 è dedicato alle comunicazioni.

U. Rembold, C. Blume, R. Dillmann
Computer-Integrated Manufacturing Technology and Systems.
Ed. Marcel Dekker Inc. 1985

Copre i vari aspetti del CIM, approfondendo gli aspetti tecnici ed organizzativi dell'automazione di fabbrica, dal progetto alla lavorazione, assemblaggio, simulazione e controllo qualità, con riferimenti agli strumenti informatici.

C. Bin
La Trasmissione Dati.
Ed. Buffetti Editore 1991

Tecniche per la trasmissione di dati, modem, reti e riferimenti a standard per reti pubbliche, con attenzione prevalente alle applicazioni gestionali.

G. Saccardi
Trasmissione dati - Dispositivi standard e protocolli.
Ed. Jackson 1986

Contiene molte informazioni di dettaglio su standard e raccomandazioni nazionali ed internazionali, con particolare attenzione alle reti telefoniche ed alle applicazioni gestionali e a grandi distanze.

R. L. Krutz
Interfacciamento nella progettazione di sistemi digitali.
Ed. Jackson 1988

Sviluppa diversi aspetti dei sistemi digitali, con particolare attenzione alle interconnessione tra i circuiti (integrati) interni, ma con interessanti presentazioni di tecniche di collegamento.
Tratta sommariamente anche aspetti software e sistemi basati su processori della famiglia 80x86.
Ricca bibliografia nei vari capitoli.

A.S. Tanenbaum
Computer networks.
Ed. Prentice-Hall 1988

Ottima presentazione, chiara e completa, delle problematiche delle reti di calcolatori, con analisi delle funzionalità ai vari livelli ISO-OSI e riferimenti agli standard per reti locali e geografiche.

U.D. Black
Data networks: concepts, theory and practice.
Ed. Prentice-Hall 1989

H. Nussbaumer
Informatique Industrielle.
Vol. IV
Ed. Presses Polytechniques Romandes 1987

Insieme ai volumi I,II,III costituisce una interessante piccola enciclopedia dei problemi di uso degli strumenti informatici nelle applicazioni industriali.

ANIPLA - Raccolta di interventi 1987
La rete MAP nell'automazione di fabbrica.

Presentazione sintetica dei livelli ISO-OSI, problemi delle reti per automazione di fabbrica, esperienze applicative e standard MAP/TOP.

ANIPLA - Raccolta di interventi 1990
Le connessioni seriali di campo nell'automazione:applicazioni e nuove proposte.

Rassegna delle caratteristiche dei bus di campo, presentazione dei principali bus, proposte ed esempi di applicazioni.

IEEE NETWORKS
Vol.2 No.3 May 1988

Volume completamente dedicato alle comunicazioni, reti di calcolatori e bus di campo nell'ambito delle applicazioni industriali.

H.C. Folts
Data Communications Standards
McGraw-Hill 1982

Corposa raccolta di standard elettrici e funzionali per le comunicazioni seriali.

K.J. Astrom, B. Wittenmark
Computer-Controlled Systems: Theory and Design.
Prentice-Hall Int. - 2a ediz. 1990

K. Bender e altri
PROFIBUS: The Fieldbus for Industrial Automation
Prentice-Hall 1993