Problematiche implementative del controllo supervisivo

Francesco Basile, Pasquale Chiacchio

Sebbene la teoria del controllo supervisivo abbia avuto una grossa attenzione da parte della letteratura scientifica nell'ultimo decennio, ne esistono poche applicazioni industriali. Uno dei motivi principali è costituito dai problemi implementativi di un supervisore sulle attuali macchine per l'elaborazione delle informazioni. Nella presente memoria si illustrano tali problematiche e viene presentato un algoritmo per l'implementazione di un supervisore su un elaboratore programmabile secondo lo standard IEC 61131.

Keyword

Discrete events systems, Supervisory control, IEC 61131 standard

Il controllore di un sistema di automazione industriale deve assicurare lo svolgimento di una o più sequenze di azioni la cui evoluzione è dettata dal verificarsi di eventi quali, ad esempio, la conclusione di una attività, l'insorgenza di anomalie, l'interazione con operatori o il risultato di un'operazione di conteggio. Questo tipo di problematica va sotto il nome di controllo logico/sequenziale. Si rimarca che le singole azioni possono poi essere realizzate da altri dispositivi di controllo che eseguono algoritmi anche complessi: dal punto di vista del controllo logico/sequenziale questi altri dispositivi di controllo sono visti come sottosistemi che devono solo essere abilitati o meno.

Dal momento che le specifiche di controllo per un sistema di automazione industriale sono costituite da sequenze di azioni, e sono per lo più espresse in linguaggio naturale, si pone un duplice problema: la stesura di queste specifiche in un linguaggio formale e la determinazione di metodologie di sintesi di un controllore che ne garantisca il soddisfacimento.

Una possibile risposta al primo problema è stata fornita dall'introduzione del linguaggio di programmazione Sequential Functional Chart (SFC). Lo SFC ben si presta allo scopo per la sua natura grafica e per avere i concetti di azione, fase e transizione tra i suoi elementi costitutivi. Nello stesso tempo, lo SFC utilizzato a livello funzionale per esprimere una certa specifica di controllo è immediatamente esprimibile come algoritmo di controllo, essendo lo SFC uno dei linguaggi ammessi dallo standard IEC 61131-3 [2,7].

Resta il problema di garantire che il controllore rispetti le specifiche. In questo caso si ha bisogno di metodologie di sintesi basate su strumenti matematici (teoremi) che garantiscano che il controllore ottenuto soddisfi le specifiche, così come avviene nella teoria del controllo classico.

Lo SFC non si presta ad essere oggetto di una trattazione matematica rigorosa che possa condurre ad una metodologia di sintesi.

Dal momento che l'evoluzione di un sistema di automazione può essere vista come una sequenza di eventi che indicano l'inizio e la terminazione di attività, è conveniente avvalersi, per lo studio di tali sistemi, delle metodologie fornite dalla teoria dei sistemi ad eventi discreti.

Un sistema ad eventi discreti è un sistema dinamico, un sistema, cioè, il cui comportamento dipende dallo stato in cui si trova, con le seguenti peculiarità: lo stato assume valori in un insieme discreto; l'evoluzione dello stato è dettata esclusivamente dall'occorrenza di eventi aventi una tempificazione asincrona, ovvero l'istante di occorrenza di tali eventi non presenta caratteristiche di regolarità.

In generale, la modellazione di tali sistemi risulta particolarmente complessa in quanto a causa della natura delle azioni, o più in generale delle attività, a cui gli eventi sono associati, è necessario rappresentare:

- la concorrenza di eventi, poiché più attività possono essere eseguite simultaneamente;
- la sincronizzazione di eventi, poiché alcune attività devono

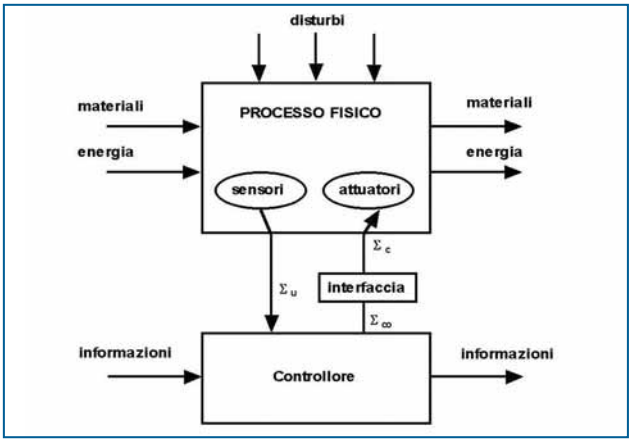

Figura 1 - Schema del controllo a forzamento di eventi

F. Basile, Università degli Studi di Salerno, Dipartimento di Ingegneria dell'Informazione e Ingegneria Elettrica; P. Chiacchio, Università degli Studi di Napoli, Dipartimento di Informatica e Sistemistica

essere completate prima di poter iniziare l'esecuzione di altre attività;
- il conflitto tra eventi, poiché alcune attività potrebbero richiedere l'utilizzo di risorse comuni.

In generale, un problema di controllo per un sistema ad eventi discreti può richiedere di attuare azioni in tempo reale sulla base della conoscenza dello stato del sistema, ed è questo il caso dell'automazione industriale. In tale ambito si presenta una ulteriore complicazione, la presenza di eventi non controllabili.
L'insieme dei possibili eventi può essere, infatti, suddiviso in due insiemi disgiunti, chiamati insieme degli eventi controllabili e insieme degli eventi non controllabili e denotati rispettivamente Σ_c e Σ_u.

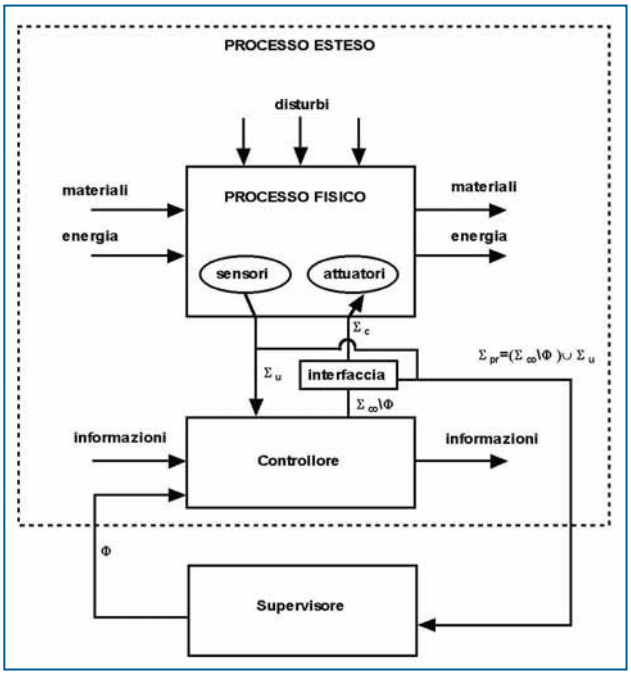

Figura 2 - Schema del controllo supervisivo

Un evento è detto non controllabile se la sua occorrenza non può essere in alcun modo influenzata da un agente esterno. Risultano essere eventi non controllabili, ad esempio, il guasto di una macchina o la terminazione di una attività.
Un evento è invece detto controllabile se la sua occorrenza dipende da un agente esterno. Il significato di questa definizione è diverso secondo l'approccio che si utilizza per il controllo di tali sistemi.
Nel cosiddetto *approccio a forzamento di eventi*, ogni evento che rappresenta l'inizio di una attività viene generato da un agente esterno, denominato controllore, e in tal senso si parla di eventi controllabili [1,3]. Con riferimento alla figura 1, il controllore, sulla base delle informazioni sugli eventi non controllabili rilevati dai sensori (insieme Σ_u) genera gli eventi controllabili in modo che vengano eseguite solo le sequenze desiderate. In realtà, il controllore genera degli eventi appartenenti ad un insieme Σ_{co} che comprende, oltre agli eventi appartenenti all'insieme Σ_c che sono inviati al processo tramite le opportune interfacce, anche eventi non controllabili generati al suo interno (ad esempio, il trascorrere di un determinato intervallo temporale).

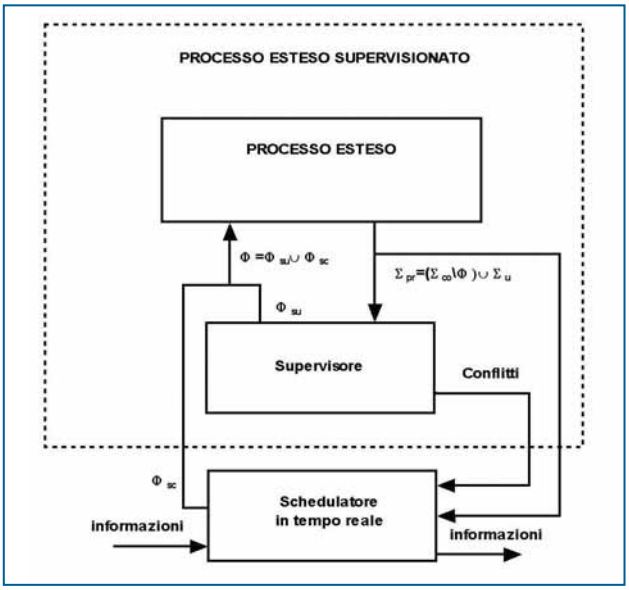

Figura 3 - Schema del controllo supervisivo con schedulatore

Nel cosiddetto *approccio del controllo supervisivo*, invece, il processo fisico viene esteso fino ad includere i dispositivi di controllo diretto con i relativi algoritmi (figura 2) [4]. In questo contesto sono i dispositivi di controllo diretto che eseguono le sequenze di azioni, alcune delle quali possono essere disabilitate da un agente esterno, denominato supervisore. Dal punto di vista del supervisore, il processo esteso viene ad essere un generatore di eventi, denotati Σ_{pr}, unione dell'insieme degli eventi non controllabili Σ_u e di quello degli eventi generati dal controllore (che appartengono a Σ_{co}). Gli eventi controllabili generabili dal controllore possono essere disabilitati dal supervisore attraverso l'invio al controllore dell'insieme ϕ degli eventi da disabilitare. Il problema del controllo supervisivo è quindi quello di disabilitare gli eventi controllabili in modo che il processo esteso rispetti le specifiche assegnate.
Come sarà mostrato in seguito l'approccio a forzamento di eventi, sebbene molto intuitivo, conduce alla sintesi di controllori non efficienti nel caso di sistemi molto complessi. Esso pertanto può essere utilizzato per controllare i singoli sottosistemi ma non è adatto per il coordinamento di più sottosistemi dove l'approccio del controllo supervisivo risulta migliore.
Si pone, inoltre, il problema della causalità, ovvero il problema di stabilire chi, tra processo e controllore, genera un certo evento. Non è sempre vero, infatti, come già accennato, che un evento non controllabile sia generato dal processo: nel caso in cui un'azione debba essere eseguita dopo che un intervallo temporale sia trascorso, è evidente che l'evento *tempo trascorso* dovrà essere generato dal dispositivo di controllo e che si tratta di un evento non controllabile. Quindi l'approccio a forzamento di eventi può condurre ad una non chiara distinzione tra eventi generati dal processo ed eventi generati dal controllore.
Alla luce di tali considerazioni risulta conveniente assumere,

per lo sviluppo della metodologia, l'architettura mostrata nella figura 2 in base alla quale è lasciato ad un controllore il compito di attuare le azioni sul processo, mentre al supervisore è lasciato il compito di abilitare o disabilitare alcune di esse.
Nell'ambito del controllo logico/sequenziale alle attività associate agli eventi non viene assegnata una durata temporale, sia essa determinata o aleatoria, per cui non possono essere presenti specifiche di controllo di tipo prestazionale che pure sono di notevole interesse. Si consideri il caso classico di una richiesta contemporanea di una risorsa da parte di due attività; ad esempio, due macchine che richiedono l'utilizzo di uno stesso robot. Il conflitto può essere risolto in maniera solo logica, per esempio con una priorità ma non è detto che questa soluzione sia la migliore dal punto di vista delle prestazioni dell'intero sistema. Sarebbe quindi auspicabile risolvere i conflitti in tempo reale secondo determinati indici di qualità.
Per tali ragioni è a volte presente un ulteriore livello nell'architettura di controllo, denominato schedulatore, il cui obiettivo è quello di risolvere in tempo reale i conflitti che vengono a determinarsi nel processo esteso soggetto a supervisione (figura 3). Lo schedulatore, sulla base degli eventi in conflitto, degli eventi generati dal processo esteso e delle informazioni provenienti dai sistemi di pianificazione aziendali, invia al processo esteso l'insieme di eventi disabilitati tra quelli in conflitto, che viene denotato Φ_{sc}. L'unione degli insiemi degli eventi disabilitati dal supervisore, ora denominato Φ_{su}, e degli eventi disabilitati dallo schedulatore forma l'insieme degli eventi disabilitati in un certo istante. Il problema della progettazione e della realizzazione dello schedulatore non sarà oggetto di trattazione di questo articolo.
Nella restante parte di questa memoria si assumerà la cosiddetta ipotesi di non concorrenza degli eventi, cioè si assumerà che due eventi non possano mai occorrere contemporaneamente. Tale ipotesi è assolutamente ragionevole nel mondo reale ma si scontra con la necessità di implementare i controllori e i supervisori su macchine sincrone; tali problematiche saranno affrontate in maggior dettaglio nel prosieguo, quando si parlerà della realizzazione dei controllori/supervisori.

Gli automi a stato finito come strumento di modello

Un automa a stati finiti è definito come la quintupla $(X,\Sigma,\delta,x_0,X_m)$ dove: X è un insieme finito di stati; Σ è un insieme finito di eventi (ingressi dell'automa); δ: $X\times\Sigma\rightarrow X$ è la funzione di transizione di stato dell'automa, che può non essere definita per ogni stato e ogni evento; si dirà che un evento e è *ammissibile* nello stato x se $\delta(x,e)$ è definita $x_0 \in X$ è lo stato iniziale; $X_m \subseteq X$ è il sottoinsieme degli stati finali, che rappresentano stati che hanno un significato particolare per colui che ha realizzato il modello.
Una rappresentazione alternativa dell'automa a stati finiti è quella di un grafo orientato i cui nodi, rappresentati da cerchi vuoti, rappresentano gli stati del sistema, e i cui archi sono associati agli eventi in seguito all'occorrenza dei quali si ha il passaggio da uno stato all'altro (nel caso gli eventi siano non controllabili si utilizzeranno archi tratteggiati). Lo stato iniziale viene contrassegnato da una freccia entrante nel cerchio che lo rappresenta mentre gli eventuali stati finali sono rappresentati da cerchi pieni.
A titolo di esempio, l'automa riportato nella figura 4b è descritto dalla quintupla: insieme degli stati X={I1,W1,A1,T1}; insieme degli eventi Σ= {s1,f1,t1,ft1}; funzione di transizione di stato le cui componenti sono δ(I1,s1)=W1, δ(W1,f1)=A1, δ(A1,t1)=T1, δ(T1,ft1),=I1; stato iniziale x_0={I1}; stato finale X_m={I1}; in questo caso il ritorno nello stato iniziale ha un significato particolare, quello di terminazione di un ciclo.

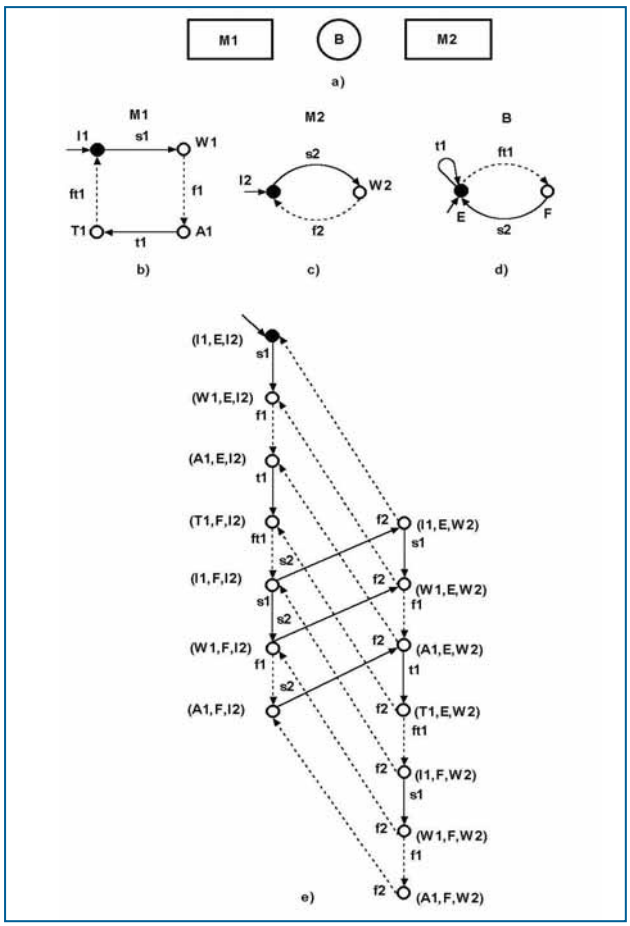

Figura 4 - Sistema formato da due macchine e da un buffer di capacità uno: a) schema del sistema; b) automa per la prima macchina; c) automa per la seconda macchina; d) automa per il buffer; e) automa per l'intero sistema

Nella figura 4a è mostrato un sistema costituito da due macchine generiche, M1 e M2, e un buffer B.
La macchina M1 ha quattro stati: I1 (libera), W1 (esecuzione lavorazione), A1 (attesa trasferimento pezzo semilavorato nel buffer), T1 (trasferimento di un pezzo semilavorato al buffer). In seguito all'occorrenza degli eventi s1 (inizio lavorazione), f1 (fine lavorazione), t1 (inizio trasferimento) e ft1 (fine trasferimento), si hanno le transizioni di stato indicate dal rispettivo automa nella figura 4b. Gli eventi f1 e ft1 sono non controllabili.

La macchina M2 ha soli due stati: I2 (libera) e W2 (prelievo dal buffer ed esecuzione lavorazione); in seguito all'occorrenza degli eventi s2 (inizio lavorazione) e f2 (fine lavorazione) si hanno le transizioni di stato indicate dal rispettivo automa nella figura 4c. L'evento f2 è non controllabile.
Il buffer B, che ha capacità pari ad uno, ha i due stati E (vuoto) e F (pieno), e in seguito all'occorrenza degli eventi s2, t1 e ft1 prima definiti si hanno le transizioni di stato indicate dal rispettivo automa nella figura 4d. Si noti che il buffer non è dotato di un sistema di controllo e pertanto bisogna impedire che la macchina M1 tenti di trasferirvi un pezzo quando esso è nello stato F o che la macchina M2 tenti di iniziare una lavorazione quando esso è nello stato E. Per cui l'automa della figura 4d è da interpretarsi come comportamento desiderato del buffer; tutti gli eventi che non compaiono in questo automa non sono soggetti a specifica.
Quando si vuole costruire il modello ad automi di un sistema a partire da moduli più semplici si deve tener conto che alcuni eventi possono essere comuni a più moduli e pertanto bisognerà, al fine di ottenere un modello del sistema complessivo, sincronizzare tali eventi. Nel caso dell'esempio nella figura 4, gli eventi ft1, t1 e s2 vanno sincronizzati in quanto ft1 e t1 sono presenti negli automi M1 e B, mentre s2 è presente negli automi M_2 e B.
Allo scopo si definisce composizione concorrente di due automi $A_1=(X_1,\Sigma_1,\delta_1,x_{01},X_{m1})$ ed $A_2=(X_2,\Sigma_2,\delta_2,x_{02},X_{m2})$ l'automa, denotato come $A_1||A_2$, definito dalla quintupla: $X\subseteq X_1\times X_2$, prodotto cartesiano degli insiemi di stato; $\Sigma=\Sigma_1\cup\Sigma_2$, unione degli insiemi degli eventi; δ: $X\times\Sigma\rightarrow X$ è definita come:
- $\delta((x_1,x_2),e)=(\bar{x}_1,x_2)$ se $e\in\Sigma_1$, $e\notin\Sigma_2$ e $\delta_1(x_1,e)=\bar{x}_1$,
- $\delta((x_1,x_2),e)=(x_1,\bar{x}_2)$ se $e\in\Sigma_2$, , $e\notin\Sigma_1$ e $\delta_2(x_2,\sigma)=\bar{x}_2$,
- $\delta((x_1,x_2),e)=(\bar{x}_1,\bar{x}_2)$ se $e\in\Sigma_1\cap\Sigma_2$, $\delta_1(x_1,e)=\bar{x}_1$, $\delta_2(x_2,e)=\bar{x}_2$,
- non definita negli altri casi;

$x_0=(x_{01},x_{02})$; $X_m=X_{m1}\,X_{m2}$.
Nel prodotto concorrente sono quindi ammissibili gli eventi "privati" ammissibili di ogni automa e gli eventi "comuni" ammissibili in tutti e due gli automi.
Il comportamento del sistema in esame è dato quindi dall'automa M1||B||M2 risultante dalla composizione concorrente dei tre automi, rappresentato nella figura 4e, avente 14 stati (nel rappresentare automi complessi, l'evento si indicherà il più vicino possibile alla partenza dell'arco associato).
Se in un automa si vuole anche tener conto della possibilità di avere delle uscite, esse possono essere definite in corrispondenza delle transizioni di stato (e si parlerà di macchina di Mealy) oppure degli stati (macchina di Moore). Si dovranno definire quindi l'insieme finito delle uscite Γ e la funzione di uscita dell'automa che sarà λ: $X\times\Sigma\rightarrow\Gamma$ nella macchina di Mealy oppure λ: $X\rightarrow\Gamma$ nella macchina di Moore.
Nel caso si voglia utilizzare un automa come modello di un controllore/supervisore di un sistema ad eventi discreti, le uscite dell'automa saranno costituite: dai forzamenti degli eventi controllabili, nel caso si utilizzi l'approccio a forzamento di eventi; dall'insieme degli eventi disabilitati, nel caso si utilizzi l'approccio del controllo supervisivo.
Si supponga di voler implementare un controllore/supervisore per il sistema della figura 4a. Gli eventi possono essere ripartiti negli insiemi $\Sigma_c=\{s1,t1,s2\}$ e $\Sigma_u=\{f1,ft1,f2\}$.

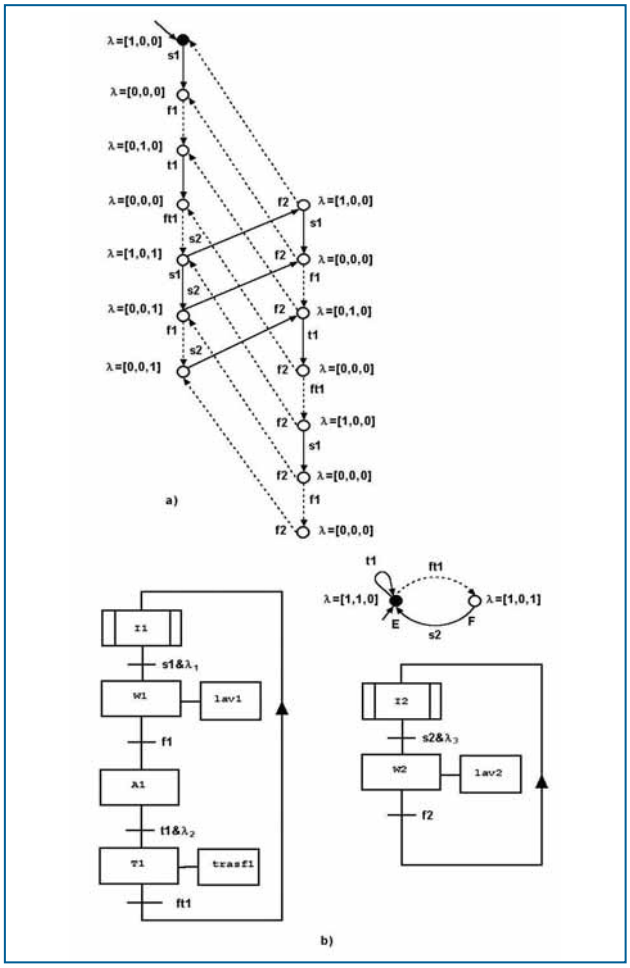

Figura 5 - Controllori per il sistema formato da due macchine e da un buffer di capacità uno: a) approccio a forzamento di eventi; b) approccio del controllo supervisivo

Nel caso si scelga l'approccio del forzamento degli eventi il controllore è riportato nella figura 5a e rappresenta la composizione concorrente degli automi delle macchine e dell'automa di specifica sul buffer; le uscite dell'automa, che risulta essere un automa di Moore, sono da interpretare come azioni effettuate direttamente sul processo. Si è scelto, per rappresentare le uscite, di utilizzare un vettore le cui componenti rappresentano il forzamento o meno dell'evento controllabile a loro associato. Per esempio, il vettore di uscita [1 0 0] associato allo stato iniziale denota il forzamento dell'evento s1 sul processo, mentre t1 e s2 non sono forzati. Si fa notare che quando in uno stato più di un evento dovrebbe essere forzato, si genera un conflitto che deve essere risolto dallo stesso controllore (ad esempio, con una priorità) o da uno schedulatore in tempo reale esterno.
Nel caso si scelga l'approccio del controllo supervisionato il controllore delle due macchine (modellato come SFC) e il supervisore (modellato con un automa) sono riportati nella figura 5b. In tal caso le uscite dell'automa del supervisore, che risulta essere lo stesso un automa di Moore, sono un vettore le

cui componenti rappresentano l'abilitazione o la disabilitazione dell'evento controllabile a loro associato. Per esempio, il vettore di uscita [1 1 0] associato allo stato iniziale dell'automa supervisore denota l'abilitazione degli eventi s1 e t1 e la disabilitazione di s2. Si noti la presenza, nel controllore descritto in SFC, di condizioni che dipendono dalle uscite del supervisore: ad esempio, la transizione in uscita dalla fase I1 può essere superata solo se l'evento s1 non è disabilitato dal supervisore.
Da tale esempio si evince come l'approccio del controllo supervisionato conduce ad una soluzione più efficiente, in quanto il supervisore ha appena due stati, mentre il controllore di figura 5 a) ne ha ben 14. Inoltre, vi è una chiara distinzione tra eventi generati dal processo ed eventi necessari per il controllo del sistema, ovvero una chiara risposta alla domanda "chi genera cosa?". Si noti infine che al variare della specifica di controllo lo SFC resta immutato, coerentemente con il fatto che il processo fisico per quanto riguarda i segnali generati resta immutato, e solo l'automa del supervisore si farà carico di imporre la nuova specifica sul sistema. Invece, se si adotta una soluzione a forzamento di eventi, sarà necessario calcolare un nuovo automa per tutto il sistema di controllo.
Per maggiori dettagli sulla teoria degli automi e del controllo supervisivo si rimanda a [5].

Problematiche di realizzazione del supervisore

Se il supervisore viene implementato su di un dispositivo di tipo sincrono, come le macchine per l'elaborazione delle informazioni attualmente in commercio, il dispositivo rimane "cieco" tra due letture successive degli ingressi. A causa di ciò possono verificarsi dei fenomeni che possono causare un erroneo comportamento del sistema a ciclo chiuso quando viene implementato il modello del supervisore ottenuto mediante le tecniche di sintesi offerte dalla teoria del controllo supervisivo [2,6].
Infatti, in un dispositivo sincrono, due eventi che accadono tra due letture successive degli ingressi saranno considerati concorrenti, ovvero accaduti nello stesso istante, e pertanto l'ipotesi di non concorrenza alla base della teoria esposta nel precedente paragrafo non può essere rispettata.
Si consideri ad esempio il supervisore rappresentato dall'automa nella figura 6a. Se esso si trova nello stato x1, può accadere che, quando si effettua la lettura degli ingressi, risultano occorsi gli eventi e1 e e2. Nel momento in cui si dovrà aggiornare lo stato dell'automa del supervisore, si dovrà necessariamente supporre un ordine di occorrenza tra gli eventi. Tale ordine potrebbe essere diverso da quello reale e determinare un erroneo aggiornamento dello stato dell'automa e una conseguente errata generazione dell'uscita. Infatti, se la sequenza reale è e1-e2, il nuovo stato del sistema sarà x4, altrimenti sarà x5.
Tale problema si presenta quando il supervisore è sensibile all'interlacciamento di eventi, come accade per l'automa rappresentato nella figura 6a, mentre nel caso del modello nella figura 6b, indipendentemente dall'ordine effettivo di occorrenza degli eventi e1 ed e2, il sistema raggiungerà sempre lo stato x4. Quindi prima di poter implementare un supervisore è necessaria un'analisi che garantisca l'assenza di situazioni del tipo di quella rappresentata nella figura 6a, altrimenti è necessario modificare, se possibile, il supervisore da implementare rendendolo non sensibile all'interlacciamento di eventi.
Un'altra problematica degna di attenzione è quella della possibile perdita di sincronizzazione tra supervisore e processo esteso. Per determinare l'uscita di controllo del supervisore, infatti, è richiesta l'esecuzione di un algoritmo che lavora sulla conoscenza dello stato del processo. Lo stato del processo potrebbe però cambiare, in seguito all'occorrenza di un evento non controllabile, durante il tempo necessario all'esecuzione dell'algoritmo. Se tale situazione si verifica, verrà inviato al processo un'uscita non coerente con lo stato in cui esso realmente si trova.

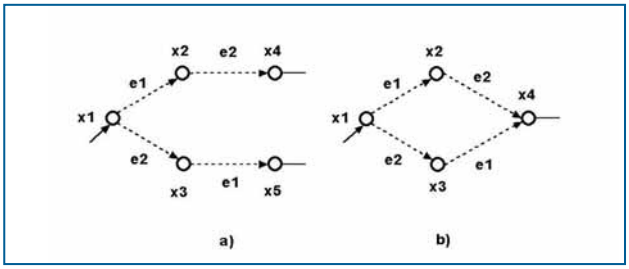

Figura 6 - Concorrenza di eventi: a) automa sensibile all'interlacciamento; b) automa non sensibile all'interlacciamento

Si supponga che il supervisore sia modellato dall'automa nella figura 7a, supposto nello stato x1, nel quale il supervisore abilita l'evento controllabile e2 se durante l'ultima lettura degli ingressi non era stata rilevata l'occorrenza di e1. Può accadere che durante l'esecuzione dell'algoritmo si abbia l'occorrenza dell'evento e1, di cui il dispositivo su cui è eseguito l'algoritmo si accorgerà solo alla successiva lettura degli ingressi. Quindi, il sistema reale si è, di fatto, portato nello stato x2 nel quale l'evento e2 non deve essere abilitato. Per cui alla successiva lettura degli ingressi la lettura dell'occorrenza di e1 non avrà alcun effetto sull'aggiornamento dello stato del supervisore che avrà definitivamente perso la sua sincronizzazione con il processo supervisionato. Il problema può essere risolto modificando il supervisore come indicato nella figura 7b.

Implementazione di supervisori

Nell'introduzione sono stati presentati due approcci per il controllo di sistemi ad eventi discreti, quello del controllo a forzamento di eventi e quello del controllo supervisivo, i quali conducono, rispettivamente, alla sintesi di un controllore o di un supervisore. Per l'approccio del controllo supervisivo sono state presentate delle metodologie di sintesi che consentono, in presenza di un certo numero di specifiche, di pervenire ad un supervisore modellabile con lo stesso formalismo utilizzato per il modello del processo (automa o rete di Petri). Per il controllo a forzamento di eventi, meno efficiente, è stato mostrato

attraverso esempi che il controllore è modellabile anch'esso tramite un automa o una rete di Petri.
Risulta pertanto utile illustrare come si possa realizzare un supervisore o un controllore, modellati come automi o reti di Petri, tramite una generica macchina per l'elaborazione delle informazioni. La procedura dovrebbe possedere i seguenti requisiti: assenza di sforzi progettuali una volta che il modello da implementare sia stato assegnato e presenza di relazioni biunivoche tra elementi del modello e istruzioni del programma. Nel presente paragrafo, per ragioni di brevità, si presenterà un possibile algoritmo per l'implementazione di supervisori; l'algoritmo, tuttavia, è facilmente adattatabile anche all'implementazione di controllori. Si assume che il modello del supervisore da implementare sia insensibile all'interlacciamento di eventi e non si possa avere mai perdita di sincronizzazione con il processo.
L'algoritmo di evoluzione presentato nella figura 8 rappresenta in forma algoritmica le regole di evoluzione del supervisore indipendentemente dal fatto che esso sia modellato ad automi o a rete di Petri.
Nella sezione di *inizializzazione* viene reso attivo lo stato iniziale dell'automa o assegnato a ciascun posto della rete di Petri il numero di gettoni previsto nella marcatura iniziale.

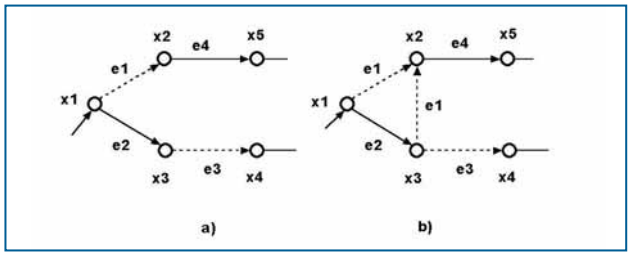

Figura 7 - Perdita di sincronizzazione: a) automa con possibile perdita di sincronizzazione; b) automa senza perdita di sincronizzazione

La *lettura delle informazioni dal processo esteso* consente di avere le informazioni necessarie a realizzare poi la *determinazione degli eventi accaduti*; in generale un evento per il supervisore, infatti, può essere generato da una elaborazione dei dati e degli eventi provenienti dal processo esteso.
È poi necessario l'*inserimento degli eventi in una lista*. Tra due letture successive è possibile, infatti, che più di un evento sia accaduto e bisogna *estrarre dalla lista un evento abilitato* alla volta per aggiornare lo stato (l'ordine di occorrenza non è determinabile e, per questo, è necessario che il supervisore non sia sensibile all'interlacciamento di eventi). L'aggiornamento dello stato, secondo l'ordine arbitrario di estrazione degli eventi, termina quando nella lista non sono più presenti eventi abilitati.
A questo punto è possibile determinare l'insieme degli eventi controllabili da abilitare nel processo esteso e inviare tali informazioni ad esso.
L'algoritmo della figura 8 può essere applicato anche per l'implementazione di un controllore, se si è utilizzato l'approccio del forzamento di eventi, ed in tal caso alla fase *abilitazione degli eventi controllabili* andrà sostituita la fase *forzatura degli eventi controllabili*. Dopo la forzatura di un evento si dovrà anche aggiornare lo stato del controllore, poiché l'evento generato è anche un evento accaduto. Infine, si dovrà più correttamente parlare di *lettura ingressi fisici* e *aggiornamento uscite fisiche* anziché di *lettura informazioni* provenienti dal processo esteso e invio informazioni provenienti dal processo esteso.

Codifica di un supervisore modellato ad automi

Per la codifica dell'algoritmo si ricorre al linguaggio di programmazione Testo Strutturato [7].
Si inizi con l'associare ad ogni stato dell'automa una variabile booleana (si indicheranno tali variabili con il nome del nodo dell'automa) il cui valore ne indicherà l'attivazione (uguale a 1 se attivo).
Si associ, poi, ad ogni arco dell'automa (evento) un blocco funzionale rilevatore di fronte di salita cui viene data in ingresso l'espressione logica associata all'evento. A ciascun evento viene associata una componente di un array di variabili booleane, il cui valore viene aggiornato dall'uscita dei rilevatori di fronte (uguale a 1 se l'evento corrispondente è accaduto). Inoltre a ciascun evento viene anche associata una variabile booleana il cui valore indicherà la sua ammissibilità nello stato attuale del supervisore (uguale a 1 se ammissibile).
Se sono accaduti eventi a partire dall'ultima lettura degli in-

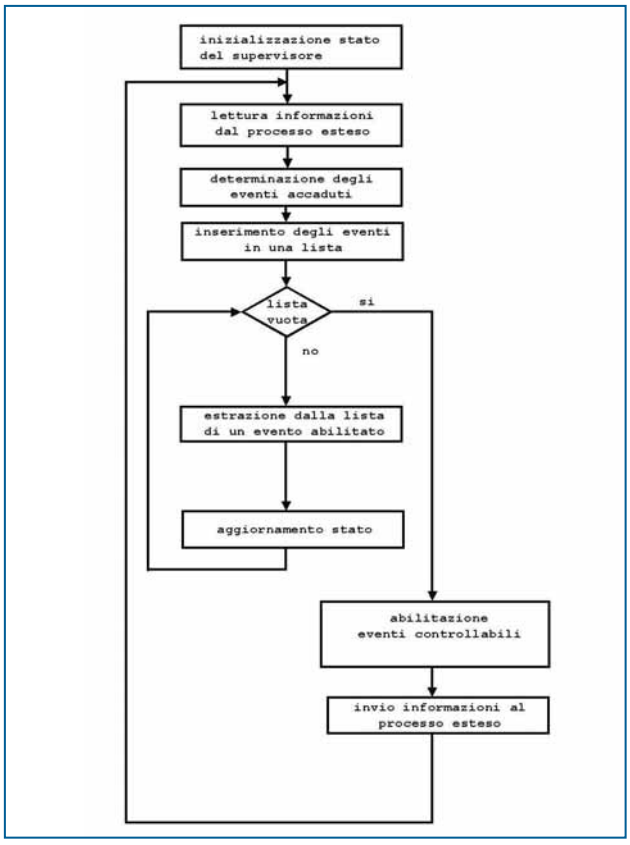

Figura 8 - Algoritmo di evoluzione del supervisore.

gressi, si esamina la lista (l'array delle variabili booleane) e appena viene trovato un evento ammissibile sotto lo stato corrente viene posto a 1 il valore della variabile booleana di ammissibilità (e la variabile dell'array viene posta al valore 0).
L'operazione viene ripetuta fino a che tutte le componenti dell'array hanno valore falso (nessun evento accaduto rimasto da esaminare).
Al fine di aggiornare lo stato del supervisore si deve, in corrispondenza di ciascun evento ammissibile accaduto, disattivare lo stato corrente e attivare lo stato futuro dell'automa in accordo alla sua funzione di transizione.
Infine, in funzione dello stato raggiunto si invieranno al processo esteso gli eventi abilitati.
Con riferimento all'automa del supervisore riportato nella figura 5b, se ne riporta la codifica in Testo Strutturato.

```
PROGRAM Automa_Supervisore
  VAR_INPUT (* informazioni in ingresso dal processo esteso*)
     E1,E2,E3: BOOL; (* segnali ft1, s2, t1 *)
  END_VAR

  VAR_OUTPUT (* informazioni in uscita verso processo este-
so*)
     ABS1,ABS2,ABT1: BOOL; (* abilitazioni di s1, s2 e t1 *)
  END_VAR

  VAR
     E,F: BOOL; (* stati dell'automa *)
     EFT1,ES2,ET1:R_TRIG; (* eventi dell'automa accaduti *)
     FT1,S2,T1:BOOL; (* eventi dell'automa ammissibili *)
     LISTA:ARRAY [1..3] OF BOOL; (* eventi accaduti *)
     CONT:INT; (* numero eventi accaduti *)
     PRIMO_PASSO, NON_ESTRATTO:BOOL:=TRUE;
  END_VAR

  IF PRIMO_PASSO=TRUE THEN (* inizializzazione *)
     E:=TRUE;
     PRIMO_PASSO:=FALSE;
  ELSE
              (*  determinazione eventi accaduti  *)
     EFT1(CLK:=E1);      (* ft1 *)
     ES2(CLK:=E2);      (* s2*)
     ET1(CLK:=E3);     (* t1*)
              (* costruzione lista degli eventi accaduti *)
     LISTA[1]:=EFT1.Q;
     IF LISTA[1] THEN CONT:=CONT+1; END_IF
     LISTA[2]:=ES2.Q;
     IF LISTA[2] THEN CONT:=CONT+1; END_IF
     LISTA[3]:=ET1.Q;
     IF LISTA[3] THEN CONT:=CONT+1; END_IF

     WHILE CONT>0 DO
        NON_ESTRATTO:=TRUE;
        (* estrazione di un evento ammissibile *)
        IF NON_ESTRATTO AND LISTA[1] AND E THEN
            FT1:=TRUE;
            CONT:=CONT-1;
            NON_ESTRATTO:=FALSE;
            LISTA[1]:=FALSE;
        END_IF
        IF NON_ESTRATTO AND LISTA[2] AND F THEN
            S2:=TRUE;
            CONT:=CONT-1;
            NON_ESTRATTO:=FALSE;
            LISTA[2]:=FALSE;
        END_IF
        IF NON_ESTRATTO AND LISTA[3] AND E THEN
            T1:=TRUE;
            CONT:=CONT-1;
            NON_ESTRATTO:=FALSE;
            LISTA[3]:=FALSE;
        END_IF
             (* aggiornamento dello stato *)
        IF FT1 THEN F:=TRUE; E:=FALSE; END_IF
        IF S2 THEN F:=FALSE; E:=TRUE; END_IF
        FT1:=FALSE;
        S2:=FALSE;
        T1:=FALSE;
     END_WHILE
         (* abilitazione eventi controllabili *)
     ABS2:=FALSE;
     ABT1:=FALSE;
     IF E THEN ABT1:=TRUE; END_IF
     IF F THEN ABS2:=TRUE; END_IF
     ABS1:=TRUE;
  END_IF
END_PROGRAM
```

Conclusioni

Si è mostrato che l'implementazione di un supervisore presenta varie problematiche, tutte in buona parte riconducibili al fatto che la teoria presuppone la non concorrenza degli eventi e la disponibilità di un dispositivo di tipo asincrono su cui implementare tali supervisori. Tali ipotesi sono quasi mai verificate dalle attuali tecnologie disponibili in commercio. Si è presentato un algoritmo di evoluzione e la sua codifica in Testo Strutturato per l'implementazione, sotto opportune ipotesi, di un supervisore modellato ad automi su un controllore a logica programmabile.

Bibliografia

[1] S. Balemi, G.J. Hoffmann, P. Gyugyi, H. wong-Toi, G.F. Franklin, "Supervisory Control of a Rapid Termal Multiprocessor", *Ieee Tran. On Automatic Control*, Vol. 38, n.7, pp. 1040-1059, 1993.
[2] P. Chiacchio, F. Basile, *Tecnologie informatiche per l'automazione 2*, Mc Graw Hill, Milano 2004.
[3] B.A. Brandin, "The Real-Time Supervisory Control of an Experimental Manufacturing Cell", *Ieee Trans. On Robotics and Automation*, Vol. 12, n.1, pp. 1-14, 1996.
[4] F. Charbonnier, H. Alla, R. David, "The Supervised Control of Discrete Event Systems", *Ieee Trans. On Control Systems technology*, Vol. 7, n. 2, pp. 175-187, 1999.
[5] A. Di Febbraro, A. Giua, *Sistemi ad eventi discreti*, Mc Graw Hill, Milano, 2002.
[6] M.Fabian, A. Hellegren, "PLC-based Implemetation of Supervisory Control for Discrere Event Systems", *Proc. of 37th Ieee Conference on Decision and Control* (Tampa, Florida, USA), December 1998.
[7] Norma Tecnica CEI EN 61131-3, Giugno, 1996. ■