

UN APPROCCIO FORMALE ED ETEROGENEO PER IL PROGETTO DI SISTEMI IBRIDI

L. Baresi ^(*), S. Carmeli ⁽⁺⁾, A. Monti ⁽⁺⁾ e M. Pezzè ^(*)

⁽⁺⁾ Dipartimento di Elettrotecnica - Politecnico di Milano
Piazza Leonardo da Vinci 32 - 20133 Milano
Tel: +39-2-23993714 - Fax: +39-2-23993703 - e-mail: [carmeli,anto]@etec.polimi.it

^(*) Dipartimento di Elettronica e Informazione - Politecnico di Milano
Piazza Leonardo da Vinci 32 - 20133 Milano
Tel: +39-2-23993400 - Fax: +39-2-23993411 - e-mail: [baresi,pezze]@elet.polimi.it

SOMMARIO

L'uso di metodi formali nello sviluppo del software in ambito industriale è tuttora limitato. La necessità di lavorare con teorie formali complesse e l'abitudine all'uso di tecniche di specifica informali sono gli ostacoli principali alla loro diffusione. In questo articolo, si propone un approccio innovativo per la progettazione di sistemi ibridi, che facilita l'uso di metodi formali complementandoli con notazioni informali solitamente utilizzate nella pratica industriale. La metodologia, che è sviluppata nell'ambito del progetto CEE ESPRIT INFORMA, è applicata alla realizzazione del controllo degli azionamenti elettrici.

PAROLE CHIAVE

Sistemi ibridi, Software in tempo reale, Reti di Petri ad alto livello temporizzate, Functional Block Diagram, Standard IEC 1131-3, Azionamenti elettrici.

1. I SISTEMI IBRIDI

Le applicazioni nel settore dell'automazione industriale sono caratterizzate da un legame stretto ed imprescindibile fra il *controllore* (il software di controllo) ed il *controllato* (l'impianto). Questi sistemi, che possono essere descritti in generale con anelli di controllo come quello mostrato in figura 1, sono spesso classificati come *sistemi ibridi*, per indicare la natura derivante dalla cooperazione di sottosistemi caratterizzati da un comportamento tempo continuo (l'impianto) e sottosistemi caratterizzati da un comportamento tempo discreto (il controllore software). La modellizzazione e verifica indipendente dei due componenti, e in particolare del software di controllo, è di aiuto limitato nella valutazione dell'intero sistema, il cui comportamento è caratterizzato dall'interazione stretta tra i sottosistemi componenti. La verifica del controllore di un sistema ibrido, non può prescindere dai dati prodotti dalla richiusura diretta dell'anello di controllo.

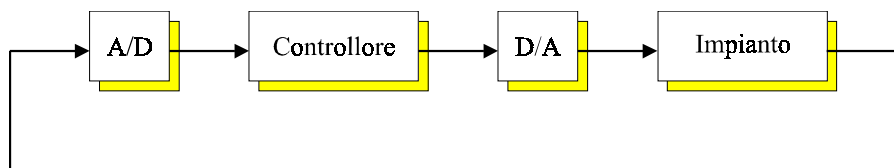


FIGURA 1: Struttura di un sistema di controllo a microprocessore

I sistemi ibridi presentano vincoli temporali critici, la cui verifica necessita di metodologie e processi di sviluppo più complessi rispetto alle applicazioni tradizionali. Nel caso di applicazioni di controllo, la correttezza del software determina la *sicurezza* dell'impianto. Malfunzionamenti del software di controllo possono provocare danni irreparabili alle strutture controllate e agli operatori. L'elevata criticità delle applicazioni in questione giustifica l'utilizzo di tecniche di specifica e di progettazione formali, che consentono di provare la correttezza del sistema in esame. Nella pratica industriale spesso però si preferiscono tecniche di specifica informali, anche per applicazioni critiche, perché di più semplice utilizzo e maggiormente radicate nella cultura aziendale [11]. Fino ad oggi, l'uso di metodi formali è limitato ad applicazioni pilota ed a studi di fattibilità [9, 10].

Anche nel caso dei sistemi ibridi, per cui sono stati studiati modelli formali [1, 2, 16] e tecniche di progettazione del software di controllo [25] ad hoc, lo studio accademico non corrisponde alla pratica industriale. Nel caso dei sistemi ibridi, lo scarso successo dei metodi formali in campo industriale è dovuto, oltre che alla limitata esperienza degli specialisti, alla difficoltà di modellare ed analizzare formalmente componenti radicalmente diversi (tempo continui e

tempo discreti), per i quali può essere difficile sia definire un modello formale omogeneo, sia analizzare modelli formali eterogenei.

Questo articolo propone un approccio innovativo per la progettazione di sistemi ibridi, che risolve i problemi d'uso di metodi formali coniugando una notazione informale, diffusa nell'ambito dell'automazione industriale (*Functional Block Diagram* (FBD) nello standard *IEC 1131-3* [18]) con un modello formale per la specifica di sistemi software (reti di Petri ad alto livello temporizzate). Il metodo risolve i problemi di eterogeneità dei vari componenti integrando metodi diversi per componenti tempo continuo e tempo discreto, e, per questi ultimi, per i componenti in tempo reale critico (*hard real-time*) e componenti con vincoli temporali non critici (*soft real-time*). Il metodo è illustrato con riferimento al settore applicativo degli azionamenti elettrici. La metodologia è sviluppata nell'ambito del progetto CEE ESPRIT INFORMA [20] ed è in corso di sperimentazione attraverso la produzione di uno ambiente di supporto che verrà utilizzato su applicazioni industriali pilota prima di un successivo utilizzo su larga scala.

La proposta di questo articolo si differenzia da altri studi che coniugano notazioni informali di specifica con metodi formali ([13, 23]) per l'ambito applicativo (il settore dell'automazione), le notazioni considerate (FBD) e lo studio dell'interazione con modelli eterogenei tempo discreti e tempo continui.

Il resto dell'articolo è organizzato come segue. Il capitolo 2 illustra brevemente i problemi relativi alla progettazione di azionamenti elettrici. Il capitolo 3 presenta la pratica industriale basata sull'uso di FBD. Il capitolo 4 illustra l'approccio INFORMA, indicando i principali componenti, descrivendone l'uso in un ambiente integrato e motivandone la scelta. Il capitolo 5 introduce le reti di Petri ad alto livello temporizzate (HLTPN) e definisce la corrispondenza fra FBD e HLTPN, illustrando come il progettista può trarre beneficio dal modello formale (HLTPN) interagendo con il *gergo* informale (FBD). Il capitolo 6 descrive l'interazione dei modelli eterogenei per modellare componenti diversi, mettendo a fuoco in particolare il problema dell'analisi del sistema mediante simulazione. Il capitolo 7 presenta un esempio. Infine, il capitolo 8 conclude riassumendo i principali contributi del lavoro ed indicando gli sviluppi futuri.

2. GLI AZIONAMENTI ELETTRICI

La struttura di controllo di un azionamento elettrico è composta da un insieme di anelli di controllo nidificati. In particolare, facendo riferimento alla figura 2, si hanno - dall'interno verso l'esterno - la regolazione di corrente o coppia, la regolazione di velocità e l'eventuale regolazione di posizione.

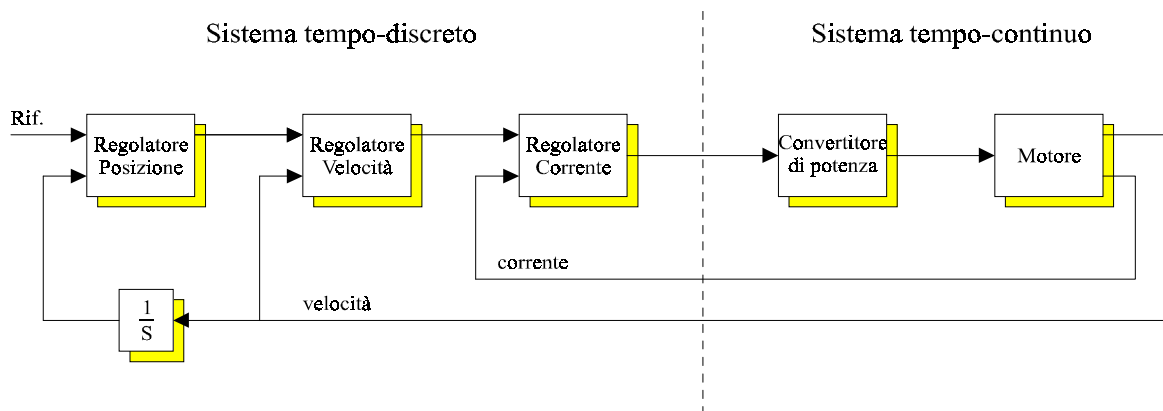


FIGURA 2: Schema di regolazione di un azionamento elettrico

I diversi anelli impongono vincoli temporali di granularità diversa, che richiedono soluzioni specifiche. I vincoli temporali sono a granularità crescente nei diversi anelli dall'interno verso l'esterno. È quindi più difficile gestire il controllo degli anelli interni, ed in particolare dell'anello di corrente, che presenta vincoli temporali molto stretti, che possono essere rispettati solo con processori ad elevata velocità e software particolarmente ottimizzato. Il problema tempo diventa ancora più rilevante se si considerano cicli più interni, come il ciclo di regolazione del convertitore di potenza, che comporta l'interfacciamento del sistema per il controllo diretto dell'accensione delle valvole (tiristori, transistor, IGBT). Spesso, i problemi vengono risolti utilizzando più processori con caratteristiche diverse: un microprocessore *general purpose* o un DSP, per l'anello di velocità e l'orientamento di campo, compresi, quindi, gli anelli di regolazione su assi di Park (tempo di ciclo pari ad 1 millisecondo) ed un microcontrollore per il controllo diretto delle valvole (tempo di ciclo pari a 400 microsecondi). Questa scelta è dovuta alle caratteristiche intrinseche del microcontrollore, che grazie alle sue elevate capacità di controllo dell'ingresso/uscita può agire direttamente sul sistema controllato in tempi assai ridotti.

I costi di progettazione possono essere ottimizzati creando librerie di componenti hardware e software riusabili. In questo caso, l'esperto di controllo può realizzare il progetto del regolatore per il sistema in esame facendo riferimento

a *schemi mentali* (blocchi funzionali) standardizzati, ad un livello d'astrazione in cui le specifiche possono essere definite connettendo blocchi funzionali predefiniti, prescindendo dai problemi di livello inferiore (hardware e software). I vantaggi di questo approccio sono:

- La rapidità nella produzione.
- La corrispondenza tra la specifica del controllore ed il codice realizzato.
- La qualità e la manutenibilità del codice garantita dal riuso di componenti *standard*.

Questo approccio, anche se ben radicato nella pratica industriale, presenta lacune che devono essere rimosse al fine di ottenere una metodologia di specifica completa ed affidabile. Le deficienze principali sono:

- L'impossibilità di compiere verifiche di consistenza temporale sulle specifiche prodotte.
- La mancanza di un approccio alla progettazione che integri il controllore (software) ed il controllato, sin dalle prime fasi del lavoro.

3. LA PRATICA INDUSTRIALE

In ambito industriale si è cercato di ovviare alle lacune evidenziate al paragrafo precedente introducendo strumenti specifici di supporto alla progettazione di software di controllo [27], spesso basati su FBD. Questi ambienti sono soluzioni parziali, non essendo ancora in grado di coprire in modo ottimale l'intero processo di sviluppo.

Una specifica (*programma*) FBD è organizzata in sezioni. Ogni sezione è composta da uno o più blocchi. L'ordine di esecuzione delle diverse sezioni è definito raggruppando le sezioni in gruppi (*task*). Una sezione può essere eseguita all'interno del ciclo principale (*main task*), che viene eseguito con una cadenza fissa programmabile dall'utente (tipicamente 100 millisecondi), oppure all'interno di altri *task* eseguiti anch'essi con una cadenza fissa (ad esempio ogni millisecondo). L'ordine di esecuzione dei *task* è definito assegnando priorità fisse. La figura 3 illustra un esempio: un programma FBD composto da tre *task*: *Main task* (per l'ingresso/uscita digitale e la logica di start/stop), *Task 1* (per l'anello di corrente, eseguito ogni millisecondo con priorità massima), *Task 2* (per l'anello di velocità, eseguito ogni 4 millisecondi con priorità inferiore). La figura indica i tre *task* e dettaglia l'anello di corrente (*Task 2*). Gli ambienti di sviluppo più completi, basati su FBD, permettono di generare automaticamente il codice dalla specifica, che può essere simulata sul microprocessore *target*, sul quale deve essere disponibile un sistema operativo *multitasking* in tempo reale per garantire la corretta esecuzione dei diversi *task*.

Metodologie di progetto, come quella descritte in questo capitolo, sono un passo importante verso un approccio *standardizzato* alla progettazione dei sistemi di azionamento. Sono però orientate soprattutto alle fasi terminali di progetto dei componenti software, in particolare a codifica e verifica del codice, ma non forniscono supporto sufficiente alle fasi iniziali di specifica ed analisi integrata dell'impianto. Queste metodologie non permettono né la specifica integrata controllore-impianto controllato, né la verifica della specifica integrata, se non simulando il codice generato. Un ambiente completo per il progetto integrato dell'azionamento dovrebbe estendere le funzionalità attuali per permettere:

- La specifica integrata di tutti i componenti (tempo continui, tempo discreti, critici e non).
- La verifica delle proprietà, ed in particolare delle proprietà temporali, dell'intero modello prima di produrre codice.
- La simulazione del codice, che resta una fase di verifica importante, ma non sostitutiva di analisi e simulazioni approfondite delle specifiche.

La metodologia proposta in questo articolo completa le metodologie esistenti definendo un ambiente che integra metodi formali ed informali diversi per i vari componenti per permettere la progettazione integrata dell'intero impianto e la verifica delle proprietà dell'impianto mediante simulazione ed analisi della specifica, complementando così la generazione di codice e la simulazione sulla macchina *target* fornite dai sistemi esistenti.

4. L'APPROCCIO INFORMA

Componenti di natura diversa possono essere modellati con formalismi diversi: componenti tempo continui, ad esempio motori elettrici, possono essere modellati con opportune equazioni differenziali che descrivono l'evoluzione delle quantità caratteristiche (velocità angolare, corrente) che modellano i componenti fisici; componenti tempo discreti critici, ad esempio i componenti software degli anelli di controllo, possono essere modellati con opportuni formalismi di specifica, quali i diagrammi FBD; i componenti tempo discreti non critici possono essere modellati con linguaggi generali per la specifica del software, ad esempio diagrammi di flusso dei dati oppure notazioni orientate agli oggetti. Ciascuna classe di formalismi, benché utile per l'uso specifico, mal si adatta alla modellizzazione di componenti di natura diversa.

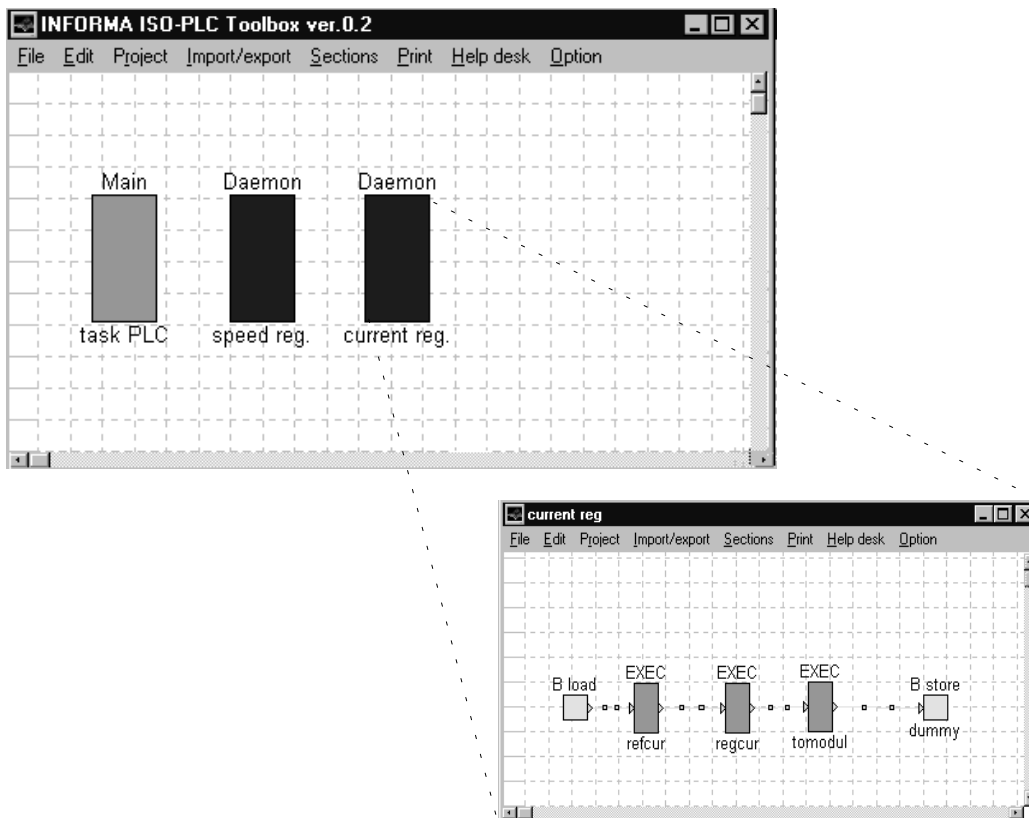


FIGURA. 3: Un esempio di specifica FBD

L'approccio INFORMA non si propone di sostituire le notazioni esistenti per i diversi componenti, nel tentativo di proporre un modello comune, ma intende integrare formalismi eterogenei in un unico ambiente di specifica e verifica. L'approccio INFORMA complementa formalismi ampiamente diffusi nella pratica industriale con modelli formali adeguati al fine di garantirne l'analizzabilità ed integra questi formalismi in un unico ambiente aperto di specifica e verifica. In particolare, INFORMA propone l'uso di equazioni differenziali opportunamente presentate in modo grafico per modellare i componenti tempo continui (l'impianto), lo standard FBD complementato con HLTPN per modellare componenti tempo discreti critici (software di controllo), e UML (Unified Modeling language) [12] complementato con VDM⁺⁺ (un'estensione orientata a oggetti di VDM - Vienna Development Method) [22] per modellare componenti tempo discreti non critici, come ad esempio la diagnostica di impianto.

Le equazioni differenziali sono utilizzate comunemente nella pratica industriale per modellare e simulare componenti tempo continui e sono ben supportate da strumenti commerciali sofisticati, come ad esempio MATLAB/SIMULINK. INFORMA non intende sostituire, ma integrare la pratica industriale corrente. In particolare, l'ambiente INFORMA si integra con MATLAB/SIMULINK per la modellizzazione, simulazione ed analisi dei componenti tempo continui. MATLAB/SIMULINK costituisce l'infrastruttura software del prodotto INFORMA, su cui si integrano tutti i componenti dell'ambiente.

Lo standard FBD, illustrato dall'esempio di figura 3, è ampiamente diffuso per la modellizzazione di componenti software di controllo per azionamenti elettrici ed è supportato da diversi strumenti industriali. Lo standard FBD è estremamente generale e deve essere completato con la definizione del comportamento funzionale dei singoli componenti o librerie di componenti per essere utilizzato in applicazioni specifiche. La maggior parte degli strumenti industriali oggi disponibili definiscono componenti e librerie mediante codice (spesso C) o pseudo-codice opportunamente commentato ed annotato. Tale specifica riflette l'uso limitato alle fasi finali di codifica e verifica del codice degli strumenti stessi. INFORMA propone di completare la definizione di componenti e librerie con HLTPN. Le reti di Petri sono state ampiamente sperimentate per la specifica ed analisi di sistemi concorrenti; le estensioni ad alto livello temporizzate permettono la modellizzazione e l'analisi anche di aspetti funzionali e temporali come quelli richiesti per il software di controllo per azionamenti elettrici [5]. Le reti di Petri ed il loro utilizzo in INFORMA, sia per completare la definizione dei moduli FBD, sia per descrivere formalmente la semantica e quindi per analizzare le specifiche dei componenti, sono descritti in dettaglio nel prossimo capitolo.

UML è una notazione meno nota in ambito industriale specifico (controllo di azionamenti elettrici), ma si sta rapidamente imponendo tra le notazioni orientate agli oggetti per la specifica del software, che attualmente rappresentano le

tecniche in maggior espansione. Nell'ambiente industriale specifico del controllo di azionamenti elettrici si è data poca rilevanza fino ad ora alla specifica e progettazione di componenti non tempo critici, che sono per lo più realizzati direttamente in codice C. Tale pratica, ampiamente giustificabile per la minor rilevanza di tali componenti rispetto al resto del sistema, riduce però l'insieme di funzionalità realizzabili a costi contenuti. In un ambiente integrato, quale INFORMA, l'uso di tecniche di specifica e progetto avanzate permette di produrre sistemi molto più efficienti e completi a costi inferiori. In questo caso, in INFORMA si è privilegiato l'uso di tecniche sperimentate e di successo nell'ambito software tradizionale come UML, in assenza di tecniche specifiche nell'ambiente di riferimento. UML si basa su una serie di diagrammi che permettono di definire la struttura del software in funzione delle classi componenti e delle relazioni tra classi, secondo la tecnologia ad oggetti [24]. Il comportamento delle singole classi può essere specificato in UML utilizzando Statecharts [17] un'estensione gerarchica delle macchine a stati finiti. Statecharts permettono di descrivere in dettaglio gli aspetti di controllo, ma risultano meno efficaci per la descrizione dei dati e degli aspetti funzionali. Per questo in INFORMA UML è integrato con VDM⁺⁺, un modello formale che ben si adatta alla descrizione di dati e aspetti funzionali in un paradigma orientato agli oggetti. In INFORMA, UML e VDM⁺⁺ possono essere considerati due viste diverse dei componenti non tempo critici che possono essere specificati indifferente in uno o entrambi i formalismi. In ogni caso, la semantica dei moduli specificati in questo modo è espressa completamente in VDM⁺⁺ che permette l'esecuzione, l'analisi e la simulazione di tali componenti.

L'ambiente INFORMA è progettato in modo da non richiedere l'uso di tutti i componenti per la specifica di un sistema di controllo, qualora non necessario. Opportune librerie FBD permettono all'utente di utilizzare solo la sintassi FBD senza dover utilizzare le reti di Petri (HLTPN). L'ortogonalità dei componenti UML/VDM⁺⁺ permette di ignorare questi modelli nel caso di sistemi con componenti non critici semplici che non richiedono l'uso di tecnologia sofisticata. In questo modo l'ambiente INFORMA è utilizzabile sia per applicazioni tradizionali, che possono essere risolte con l'uso di MATLAB/SIMULINK e librerie FBD standard, sia per applicazioni complesse, in cui può essere necessaria l'estensione dell'ambiente con librerie FBD ad-hoc, possibile mediante HLTPN, sia la specifica di componenti non critici complessi, che può essere semplificata con l'uso di UML/VDM⁺⁺.

La specifica di un sistema di controllo in INFORMA comprende quindi componenti MATLAB/SIMULINK, FBD/HLTPN e UML/VDM⁺⁺. Una specifica integrata e completa (aspetti funzionali e temporali) può essere verificata senza la necessità di ricorrere a simulazione sulla macchina target del codice generato. Questo è possibile perché il comportamento dell'impianto è modellato con MATLAB/SIMULINK e può quindi essere simulato senza attivare l'impianto stesso; il comportamento del software è modellato completamente dall'uso complementare di metodi informali grafici (FBD e UML) e metodi formali (HLTPN e VDM⁺⁺), che permettono di descrivere anche gli aspetti di funzionali e di temporizzazione. La simulazione integrata dell'intero impianto richiede la sincronizzazione dei diversi modelli semantici: MATLAB/SIMULINK, HLTPN che esprimono la semantica di FBD e VDM⁺⁺ che esprime la semantica di UML. I problemi relativi alla simulazione di un modello eterogeneo sono discussi in dettaglio nel capitolo 6 di questo articolo.

L'ambiente INFORMA è completato con strumenti di *debugging* e generazione di codice. I primi permettono di localizzare errori nelle specifiche rilevati con la simulazione. I secondi permettono di generare automaticamente codice C. La completezza dei modelli formali utilizzati in fase di specifica e la definizione della temporizzazione dei vari componenti permette di generare codice ottimizzato in modo completamente automatico. L'architettura generale dell'ambiente INFORMA è illustrata in figura 4, che mette in evidenza le interfacce utente (MATLAB/SIMULINK, FBD/HLTPN, UML/VDM⁺⁺), il ruolo delle HLTPN come *denominatore* semantico per la specifica del software di controllo critico, l'insieme di strumenti di analisi e verifica disponibili per INFORMA e l'interfaccia con strumenti tradizionali di simulazione del codice sulla macchina target.

In sintesi, l'ambiente INFORMA si presenta come un sistema multiformale in grado di coprire il ciclo di vita del software per applicazioni di controllo in tempo reale, in cui è possibile:

- Simulare completamente la specifica del sistema integrato (controllato-controllatore).
- Specificare il software con vincoli temporali *hard real-time* usando il formalismo FBD opportunamente complementato con HLTPN.
- Definire i componenti con requisiti temporali non stretti (*soft real-time*) con UML opportunamente complementato con VDM⁺⁺.
- Generare automaticamente il codice a partire dalla specifica eterogenea.

L'ambiente INFORMA non è legato ad una particolare processo di sviluppo, ma si adatta a diversi processi. Un possibile uso delle principali funzionalità di INFORMA per progettare un sistema di controllo è descritto nel capitolo 7.

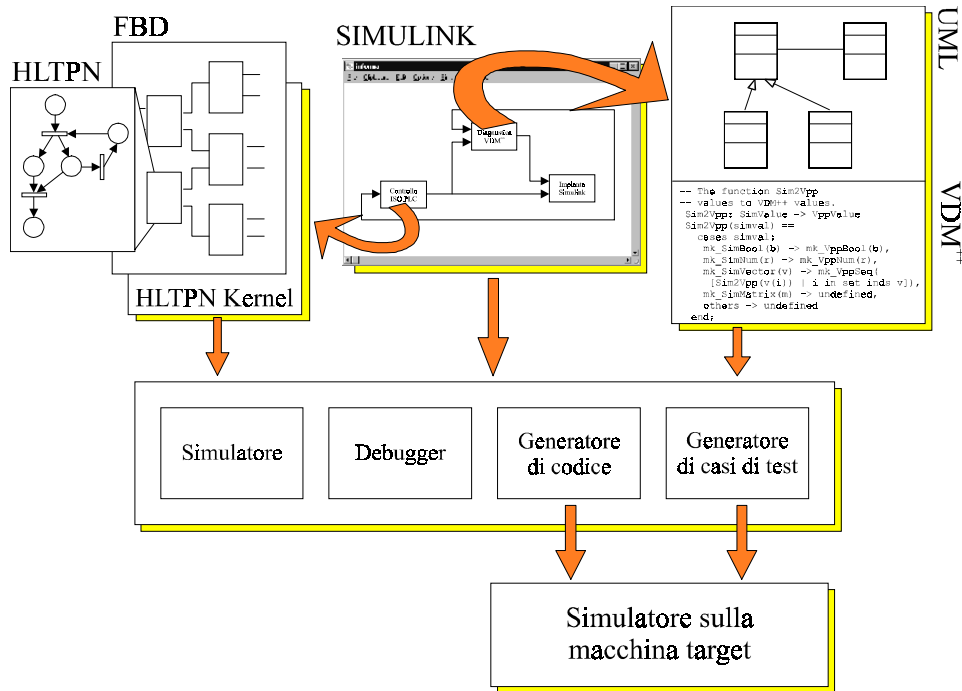


FIGURA 4: Architettura dell'ambiente INFORMA

5. I FUNCTIONAL BLOCK DIAGRAM E LE RETI DI PETRI

5.1 LE RETI DI PETRI

Al fine di facilitare il lettore non esperto, prima di illustrare il rapporto tra FBD e HLT PN, introduciamo brevemente il formalismo HLT PN. Una *rete di Petri* (figura 5) è un grafo bipartito orientato [28]: i nodi del grafo orientato sono cioè divisi in due gruppi, chiamati *posti* e *transizioni*, e gli archi non possono connettere nodi appartenenti allo stesso gruppo. Graficamente, i posti sono rappresentati con cerchi e le transizioni con barre o rettangoli. Informalmente, i posti rappresentano le condizioni necessarie perché si verifichino gli eventi (modellati da transizioni). La validità di una condizione è rappresentata da *token* (gettoni) che marcano i posti. Una particolare disposizione di gettoni nei posti è detta *marcatura*. L'insieme dei posti collegati con archi entranti ad una transizione t viene chiamato il *preset* di t ; l'insieme dei posti collegati con archi uscenti da t è detto il *postset* di t .

L'evoluzione di una rete di Petri è descritta dalla seguente *regola di scatto*, che definisce i concetti di abilitazione e scatto. In una marcatura una transizione è *abilitata* se esiste almeno un gettone in ogni posto del suo preset. Ogni transizione abilitata può scattare. Lo *scatto* di una transizione abilitata t comporta la rimozione di un gettone da ogni posto del preset di t e l'aggiunta di un gettone in ogni posto del postset di t . Ad esempio, lo scatto della transizione $T1$ di figura 5, rimuove l'unico gettone dal posto $P1$ e aggiunge un gettone nel posto $P3$.

Le reti di Petri permettono di rappresentare conflitti, concorrenza, indipendenza, sincronizzazione. Ad esempio, nella rete di figura 5, le transizioni $T1$ e $T2$ (così come $T2$ e $T3$) sono in conflitto: lo scatto di $T1$ rimuove il gettone dal posto $P1$ disabilitando così la transizione $T2$ e viceversa; le transizioni $T1$ e $T3$ sono concorrenti, possono cioè scattare a partire dalla stessa marcatura in un qualsiasi ordine; la transizione $T3$ rappresenta un possibile punto di sincronizzazione tra due processi distinti i cui stati sono rappresentati dai gettoni nei posti $P1$ e $P2$, rispettivamente. La *storia* di una rete di Petri è caratterizzata dalla sequenza di scatti delle transizioni, o, equivalentemente, dalla sequenza di marcature prodotte dagli scatti.

Le reti di Petri permettono di modellare bene gli aspetti di controllo, ma sono meno adatte per modellare gli aspetti funzionali e temporali. Le HLT PN permettono di superare questi limiti, arricchendo le reti di Petri con informazioni associate ai gettoni ed alle transizioni [14]. In una HLT PN, si associa una marca temporale e variabili con relativi valori ai gettoni e un predicato, un'azione ed una funzione temporale ad ogni transizione. Il *predicato* impone condizioni sui valori delle variabili dei gettoni nel *preset* della transizione. Solo le tuple di gettoni che verificano il predicato abilitano la transizione. L'*azione* indica la relazione funzionale tra i valori associati all'insieme di gettoni rimossi dallo scatto e i valori associati all'insieme di gettoni prodotti dallo scatto stesso. La *funzione temporale* indica il tempo minimo e massimo di scatto della transizione in funzione delle marche temporali e dei valori delle variabili dei

gettoni rimossi durante lo scatto. Il tempo di scatto della transizione deve essere compreso tra i valori minimo e massimo indicati dalla funzione temporale ed è il valore della marca temporale dei gettoni prodotti dallo scatto.

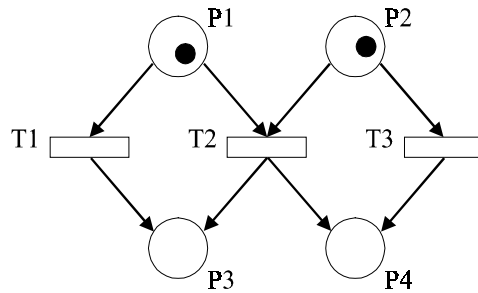


FIGURA 5: Un esempio di rete di Petri con marcatura

La semantica formale delle reti di Petri e delle estensioni ad alto livello temporali permette di risolvere ambiguità delle specifiche e di analizzarne la semantica. Calcolo degli invarianti [29], analisi di raggiungibilità temporale [15], grafo delle occorrenze [21], ma anche semplice esecuzione e simulazione permettono la verifica di importanti proprietà: assenza di deadlock, raggiungibilità di particolari situazioni, sicurezza (assenza di comportamenti a rischio).

5.2. FBD E HLTPN

La definizione formale della semantica dei blocchi FBD può essere facilmente definita in termini di HLTPN. Per ciascun blocco FBD è possibile definire una HLTPN corrispondente. La connessione di blocchi FBD corrisponde ad un'opportuna connessione delle HLTPN corrispondenti. Predicati, azioni e funzioni temporali associate alle transizioni delle HLTPN permettono di esprimere non solo gli aspetti di controllo, ma anche gli aspetti funzionali e temporali di una specifica FBD. L'analisi della HLTPN corrispondente ad una specifica FBD permette di provare importanti proprietà della specifica stessa. L'automatizzazione del processo di traduzione e la rappresentazione dei risultati dell'analisi di HLTPN in termini FBD, facilmente ottenibili con l'approccio descritto in [3] permettono all'esperto di azionamenti elettrici di trarre vantaggio dall'uso di un modello formale (HLTPN) senza richiederne la conoscenza specifica.

Ad esempio, ricordando la semantica delle HLTPN descritta in precedenza, è possibile definire la corrispondenza illustrata in figura 6. Si supponga di avere un blocco FBD caratterizzato da tre ingressi e due uscite. L'ipotesi che l'esecuzione inizi solo quando sono presenti tutti gli ingressi consente di rappresentare un singolo blocco attraverso una transizione con tre posti in ingresso. Il codice associato al blocco FBD è trasformato nell'azione della transizione. L'ipotesi, poi, che le uscite siano prodotte contemporaneamente permette di collegare direttamente alla transizione i due posti che corrispondono alle uscite del blocco.

Ponendosi ad una granularità inferiore, sarebbe possibile dettagliare ulteriormente la rappresentazione formale (HLTPN). Le funzionalità di un blocco potrebbero essere descritte da sottoreti di Petri composte da più transizioni, evidenziando le "sottoazioni" principali svolte dal blocco FBD. Un maggior livello di dettaglio fornisce un modello formale più rigoroso, ma comporta anche un notevole aumento delle dimensioni della HLTPN corrispondente. La maggiore analizzabilità ha come effetto negativo l'aumento della complessità della specifica prodotta.

L'effetto immediato della tecnica descritta in questo paragrafo è la possibilità di sfruttare tutti i benefici offerti dalle HLTPN senza conoscerle e senza dover modificare il proprio modo di lavorare, sia in termini di metodo di progettazione, che di notazioni usate.

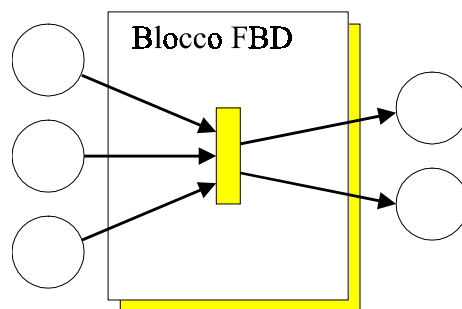


FIGURA 6: Blocco FBD e HLTPN corrispondente.

6. SIMULAZIONE DI MODELLI ETEROGENEI

Un modello formale - eseguibile ed analizzabile - del controllore software consentirebbe di valutarne la validità ed il soddisfacimento dei requisiti (temporali) richiesti prima della codifica e della validazione sul target. D'altra parte, per una verifica significativa del software di controllo è necessario produrre dati che siano logicamente, e matematicamente, compatibili con la chiusura dell'anello di regolazione. Questo implica un ambiente di simulazione in grado, da una parte, di gestire modelli tempo discreti e, dall'altra, modelli tempo continui caratterizzati da equazioni differenziali, magari non lineari e tempo varianti [6, 26].

Gli ambienti di simulazione tradizionali sono carenti nelle potenzialità di rappresentazione dei componenti discreti. Si limitano a fornire la possibilità di descrivere parti del sistema in termini di trasformata Z. La soluzione ideale, invece, dovrebbe consentire di:

- Integrare equazioni differenziali, anche complesse, che descrivono il sistema controllato.
- Gestire modelli del sistema tempo discreto - il controllore - utilizzando metodi formali quali le HLTPN.

Una soluzione di questo tipo può essere realizzata introducendo nuove funzionalità all'interno dell'ambiente MATLAB/SIMULINK. Questo strumento, grazie anche all'interfaccia grafica, è particolarmente adatto per la descrizione dei sistemi da controllare (componenti tempo continui), ma carente per quanto riguarda la definizione dei componenti tempo discreti. La sua struttura di ambiente aperto consente, però, l'aggiunta di nuovi elementi all'interno dei componenti standard. Interfaciando MATLAB/SIMULINK ed un esecutore di HLTPN è possibile attuare la seguente logica di simulazione:

- Ad ogni istante di integrazione del modello tempo continuo, si risolvono le equazioni differenziali del modello utilizzando l'algoritmo scelto dall'utente.
- Ad ogni istante di campionamento delle variabili tempo discrete, si campionano i valori e li si inviano all'esecutore di HLTPN.
- I risultati ottenuti dall'esecutore di HLTPN sono considerati validi per tutto il periodo di campionamento successivo.

Una logica di questo tipo è già stata realizzata per l'esecuzione tempo discreta di procedure di controllo scritte in linguaggio C o linguaggio di MATLAB/SIMULINK [7].

7. UN ESEMPIO

Al fine di chiarire le funzionalità, le potenzialità e i benefici legati all'utilizzo della metodologia INFORMA e di sottolineare come si articola l'intera fase di progettazione, si riporta nel seguito un esempio applicativo relativo al progetto di un sistema di controllo per un azionamento elettrico. Il sistema a cui si fa riferimento è un sistema di controllo di un motore a corrente continua alimentato da un convertitore DC-DC, caratterizzato da due anelli di regolazione (di velocità e di corrente).

La fase preliminare di specifica consiste nell'identificare i macro componenti:

- L'impianto che costituisce il sistema controllato nell'esempio è un motore a corrente continua e un convertitore DC-DC e sarà specificato in SIMULINK.
- I componenti *hard real-time*, che nell'esempio sono costituiti dai sistemi di controllo di regolazione della corrente e della velocità e saranno specificati in FBD.
- I componenti *soft real-time* che nell'esempio sono costituiti dalla diagnostica del sistema e saranno specificati con VDM⁺⁺

La figura 7 che rappresenta una delle finestre INFORMA illustra i principali componenti e indica la cardinalità delle interazioni.

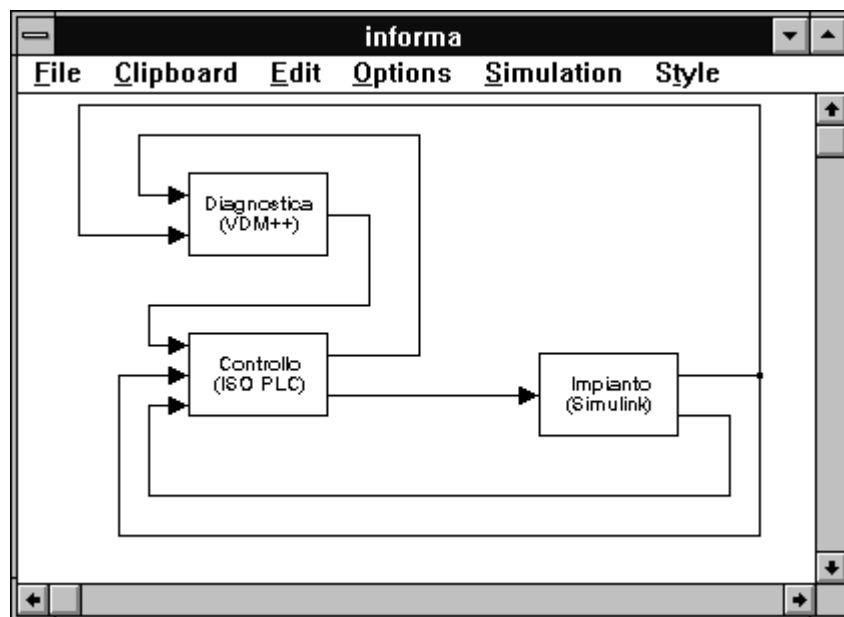


FIGURA 7: Schema integrato dell'ambiente INFORMA

Lo sviluppo prosegue con la specifica dettagliata dei singoli componenti. La specifica SIMULINK dell'impianto è riportata in figura 8, che descrive il comportamento dinamico del motore e del convertitore DC-DC attraverso la composizione di funzioni di trasferimento.

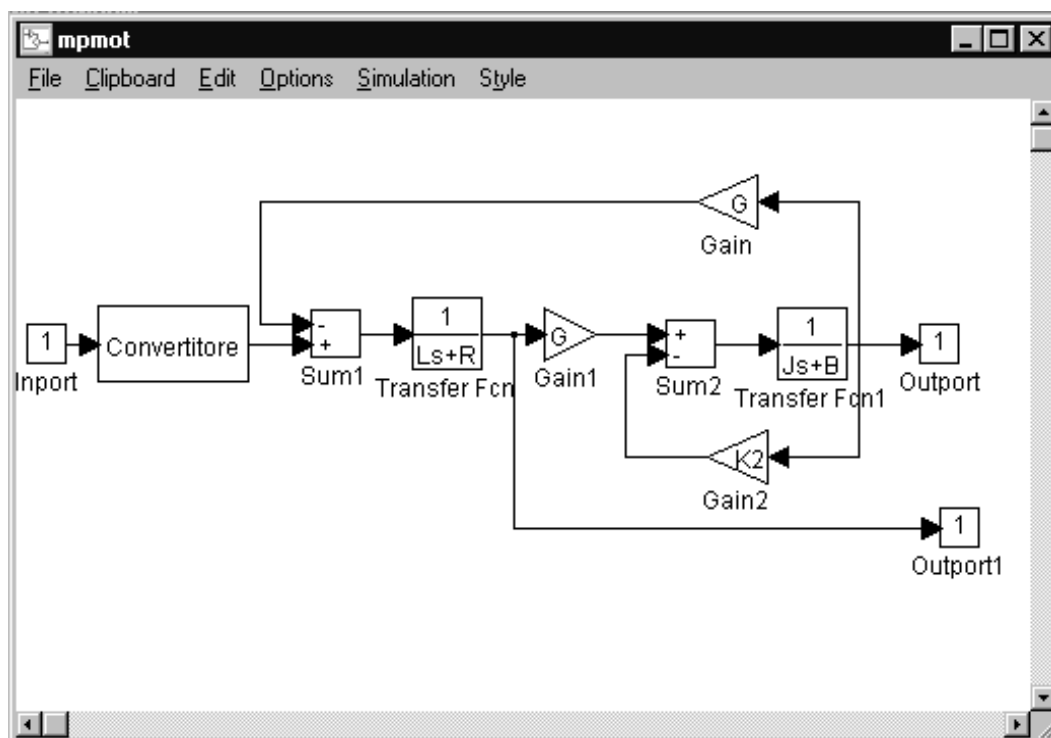


FIGURA 8: Modello, SIMULINK dell'impianto

La specifica prosegue con la definizione dettagliata del sistema di controllo. La figura 2 dettaglia le relazioni tra impianto e sistema di controllo, indicando i componenti dei due sottosistemi coinvolti nelle singole interazioni. La logica di controllo basata su due cicli annidati di controllo, pur non essendo l'unica scelta possibile è senz'altro quella più utilizzata in ambito industriale, in quanto consente di risolvere un problema di controllo multivariabile (controllo di

velocità e di corrente) come due problemi separati di regolazione. Inoltre la diversa dinamica che caratterizza i due anelli di velocità e corrente consente il progetto dei due regolatori in modo totalmente indipendente.

La specifica del sistema di controllo prosegue con l'identificazione dei task di controllo e il loro dettaglio in termini di FBD. La figura 3 descrive i task e presenta un primo livello di dettaglio del task di controllo di regolazione della corrente. Ogni blocco FBD viene tradotto in modo automatico in una HLTPTN in grado di garantire l'eseguibilità della specifica. Un ulteriore livello di dettaglio FBD del task di regolazione della corrente e la HLTPTN corrispondente ad uno dei blocchi FBD è illustrato in figura 9.

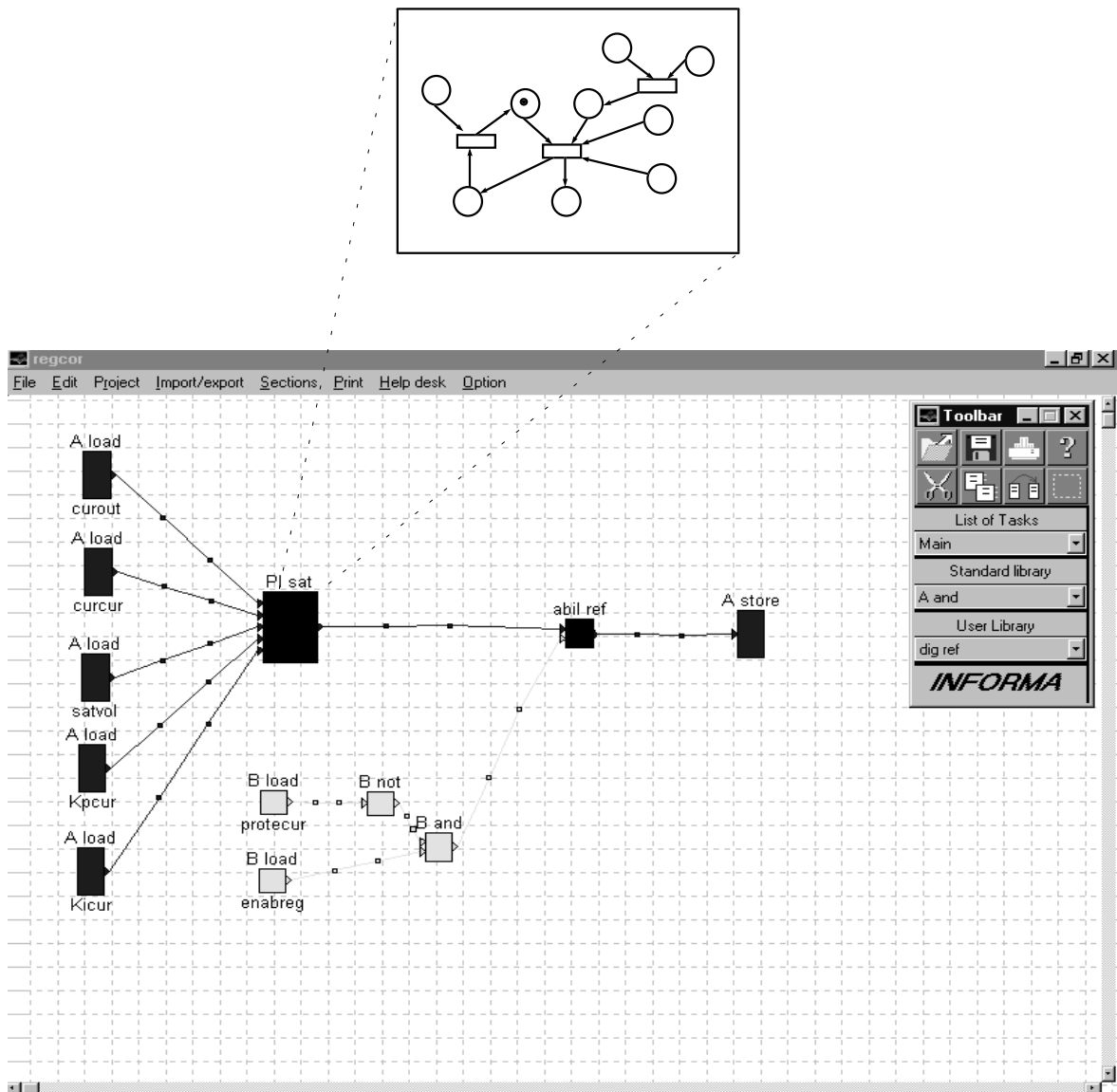


FIGURA 9: Esempio di specifica ISO PLC con HLTPTN sulla corrispondente ad uno dei blocchi FBD

La specifica dei componenti *soft real time* è mostrata in figura 10, che riporta una parte della specifica VDM⁺⁺ ed il suo corrispondente UML.

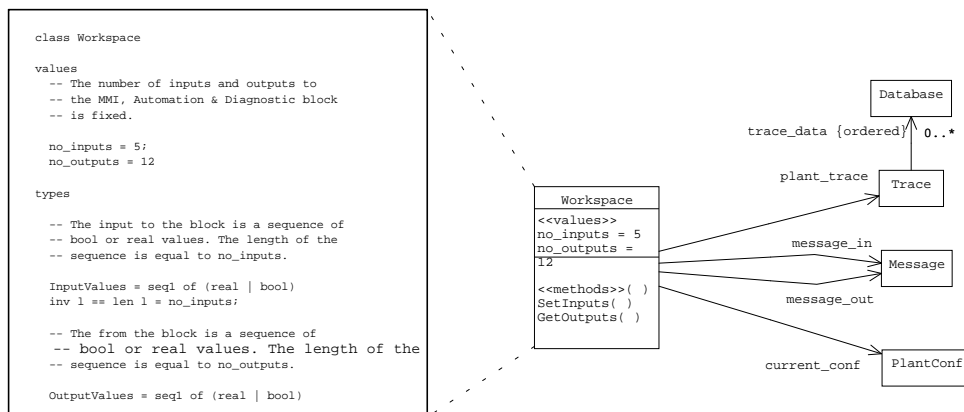


FIGURA 10: Struttura principale della specifica VDM⁺⁺

Una volta completata la fase di specifica, il modello può essere analizzato generando la specifica HLTPN corrispondente al modello FBD, che può essere eseguita in un ciclo di controllo comprendente anche l'esecuzione del modello SIMULINK dell'impianto e del modello VDM⁺⁺ dei componenti *soft real-time*. In tal modo è possibile osservare sulla scala del tempo simulato le interazioni tra sistema di controllo e sistema controllato. La traduzione delle specifiche realizzate in FBD e VDM⁺⁺ in nuovi blocchi SIMULINK consente un dialogo tra tali blocchi e il sistema controllato durante la fase di simulazione. Si approda così ad una nuova accezione del concetto di simulazione di sistema basata sulla duplice possibilità di verificare la correttezza funzionale del software e quella temporale grazie all'utilizzo di HLTPN.

Una volta verificata la correttezza della specifica mediante simulazione, si passa alla fase di generazione del codice. La verifica delle specifiche non garantisce la correttezza del codice generato, ma permette l'eliminazione di un numero notevole di errori e permette quindi di ridurre drasticamente la fase tradizionale di simulazione del codice su macchina target. Tale simulazione viene effettuata con gli strumenti tradizionalmente utilizzati in questa fase.

La conformità dell'ambiente FBD con lo standard IEC 1131 permette l'importazione e l'esportazione nei confronti di molti strumenti già in uso nella pratica industriale per il debugging real-time.

L'esempio applicativo presentato in questo paragrafo ha messo in luce i vantaggi legati all'utilizzo della metodologia INFORMA rispetto all'approccio tradizionale riassumibili nei seguenti punti:

- disponibilità di un ambiente integrato di sviluppo per le diverse fasi del progetto: l'uniformità dell'interfaccia utente e la standardizzazione delle procedure di progettazione facilitano l'esperto di controllo nel passaggio attraverso le diverse fasi del ciclo di vita del prodotto controllore;
- una nuova accezione del concetto di simulazione di sistema: si mira a fornire modelli, quanto più realistici possibile, del sistema digitale di controllo, consentendo di effettuare verifiche non solo algoritmiche, ma anche temporali grazie all'utilizzo delle reti di Petri d'alto livello;
- un elevato livello di automazione delle diverse fasi del processo di progettazione ed, in particolare, della fase di codifica;
- adeguamento a standard internazionali riconosciuti: il linguaggio *Functional Block Diagram* per quanto riguarda il PLC ed il linguaggio VDM⁺⁺, ed a standard "de facto", come MATLAB/SIMULINK, ampiamente diffuso in ambito industriale.

8. CONCLUSIONI

L'articolo ha descritto i problemi e le aspettative esistenti per quanto riguarda la progettazione del software di controllo nel campo dell'automazione. Come soluzione ai problemi tuttora irrisolti, si è descritta la metodologia proposta nell'ambito del progetto CEE ESPRIT INFORMA. Si sono illustrate le finalità del progetto e la sua possibile collocazione in ambito industriale. I punti di forza della metodologia possono essere riassunti in:

- Capacità, grazie alle corrispondenze tra FBD-HLTPN e UML-VDM⁺⁺, di *mascherare* i metodi formali agli utenti finali, consentendo loro di goderne i benefici senza avere alcuna conoscenza specifica.
- Possibilità di usare formalismi diversi e particolari per la definizione dei requisiti di componenti diversi: *hard e soft real-time*.
- Modularità dell'ambiente proposto in INFORMA per poter utilizzare - eventualmente - altre notazioni di progetto o altri metodi formali operazionali.

In conclusione, in questo articolo la metodologia è stata applicata alla definizione di azionamenti elettrici, ma l'approccio è da ritenersi valido per la progettazione di sistemi ibridi in generale.

BIBLIOGRAFIA

- [1] R. Alur, T.A. Henzinger e E.D. Sontag, "Hybrid Systems III". Volume 1066 di *Lecture Notes in Computer Science*. Springer-Verlag, 1996.
- [2] P.J. Antsaklis, "Hybrid Systems II". Volume 999 di *Lecture Notes in Computer Science*. Springer-Verlag, 1995.
- [3] L. Baresi, "Personalizzazione Formale di Notazioni Grafiche". Tesi di Dottorato. Dipartimento di Elettronica e Informazione - Politecnico di Milano, Gennaio 1997.
- [4] L. Baresi, A. Orso e M. Pezzè. "Introducing Formal Methods in Industrial Practice". Proceedings of the 19th International Conference on Software Engineering, ACM Press. Maggio 1997.
- [5] A. Caloini, G. Magnani e M. Pezzè, "A Technique for Designing Robotic Control Software Based on Petri Nets". IEEE Transactions on Control Systems Technology, Novembre 1997.
- [6] M. Carpita e A. Monti. "Voltage Source Converters and Drives Simulation at System Level for Control design Applications". EPE Journal, vol. 5 n. 3 e 4, pag. 49-55. Gennaio 1996.
- [7] F. Castelli Dezza e A. Monti. "Utilizzo di Matlab/Simulink come Sistema di Specifica per Controlli Digitali: Teoria ed Esempi Applicativi". I Conferenza Italiana Utenti Matlab - Bologna. Novembre 1995.
- [8] F. Castelli Dezza, E. Chiesa, A. Monti e M. Riva. "A New Interactive Matlab Toolbox for Teaching and Testing Electrical Drives". EPE97 - Trodheim (Norvegia). Settembre 1997.
- [9] E.M. Clarke e J.M. Wing, "Formal Methods: State of the Art and Future Directions". Rapporto Tecnico CMU-CS-96-178. Carnegie Mellon University, Settembre 1996.
- [10] J. Crow e B.L. de Vito, "Formalizing Space Shuttle Software Requirements". First Workshop on Formal Methods in Software Practice, ACM, Gennaio 1996.
- [11] K. Finney, "Mathematical Notation in Formal Specification: Too Difficult for the Masses?". IEEE Transactions on Software Engineering. vol. 9, n. 6, pag. 733-744. Giugno 1996.
- [12] M. Fowler e K. Scott, "UML Distilled: Applying the Standard Object Modeling Language", Addison-Wesley, 1997.
- [13] R. France e M. Larrondo-Petrie, "From Structured Analysis to Formal Specifications: State of the Theory". Proceedings of the ACM Computer Science Conference, pag. 249-256. Marzo 1994.
- [14] C. Ghezzi, D. Mandrioli, S. Morasca, M. Pezzè, "A Unified High-level Petri Net Formalism for Time Critical System". IEEE Transactions on Software Engineering. vol. 17, n. 2, pag. 160-172. Febbraio 1991.
- [15] C. Ghezzi, S. Morasca e M. Pezzè, "Validating Timing Requirements of Time Basic Net Specifications", Journal of Systems and Software, 27(7): 97-117, Novembre 1994.
- [16] R.L. Grossman, A. Nerode, A.P. Ravn e H. Rischel, "Hybrid Systems". Volume 736 di *Lecture Notes in Computer Science*. Springer-Verlag, 1993.
- [17] D. Harel, Statecharts: "A Visual Formalism for Complex Systems". Science of Computer Programming, vol. 8: 231-274, 1987
- [18] IEC, IEC1131-3 International Standard, 1993.
- [19] IFAD, "VDM++ Toolbox: User Manual", 1996
- [20] INFORMA, "Technical Annex", Gennaio 1997.
- [21] K. Jensen, "Coloured Petri Nets". *Advances in Petri Nets* (serie edita da W. Reisig e G. Rozemberg), Springer-Verlag, 1987 e 1997.
- [22] C. B. Jones, "Systematic Software Development Using VDM", Prentice-Hall, 1989.
- [23] M. Kronlöf, "Method Integration - Concepts and Case Studies". John Wiley & Sons, 1993.
- [24] B. Meyer, "Object-Oriented Software Construction", Prentice Hall, second edition, 1997.
- [25] A. Monti. "Circuits as Concurrent Systems: Through a New Approach to the Simulation". IEEE ISCAS 96 - Atlanta (USA).
- [26] A. Monti, F. Noli, "Towards a Flexible Environment for Developing and Testing Software-based electrical Drives". SPEEDAM 94, Taormina (Italy).
- [27] M. Matuonto, E. Mazzocut, A. Monti, "Graphic Software Designers: a Quick Way to Develop and Standardise Electrical Drives Digital Control". SPEEDAM 94, Taormina (Italy).
- [28] T. Murata, "Petri Nets: Properties, Analysis, and Applications". Proceedings of the IEEE, vol. 77, n. 4, pag. 541-580. Aprile 1989.
- [29] J. Peterson, "Petri Net Theory and the Modeling of Systems", Prentice-Hall, 1981.