TATEA TEHNICĂ DIN TIMIŞOARA
TEA DE AUTOMATICĂ ŞI CALCULATOARE

CONTRIBUŢII PRIVIND STRUCTURAREA ŞI IMPLEMENTAREA SISTEMELOR DE CONDUCERE PE BAZA MODELELOR DE REŢEA PETRI

TEZĂ DE DOCTORAT

Autor:
Takács Balázs - Imre

CONDUCĂTOR ŞTIINŢIFIC:
Prof. dr.ing. Toma Leonida Dragomir

1998

MULȚUMIRI

Este o onoare deosebită pentru mine faptul că am avut posibilitatea să lucrez la această teză de doctorat sub conducerea domnului prof.dr.ing Toma Leonida DRAGOMIR. Mulțumesc pe această cale pentru generozitatea cu care, făcând uz de competența profesională deosebită, m- a îndrumat și m-a ajutat.

Aduc prinosul meu de recunoștință și cele mai calde mulțumiri domnilor prof.dr.ing Tiberiu LEȚIA, prof.dr.ing Octavian PĂSTRĂVANU și prof.dr.ing. Aurel Mihai STĂNESCU pentru analiza lucrării, observațiile făcute și susținerea tezei.

Mulțumesc deasemenea tuturor autorilor de publicații și lucrări științifice care, la solicitarea mea, m-au ajutat punându-mi la dispoziție lucrări valoroase legate de tematica prezentei lucrări.

Mulțumesc în mod deosebit soției mele și copiilor mei pentru răbdarea și înțelegerea de care au dat dovadă pe tot parcursul elaborării acestei lucrări.

B

CUPRINS

SINTEZA LUCRĂRII

Constructorii sistemelor de automatizări industriale moderne trebuie să creeze sisteme de înaltă performanţă ce vor evolua în scenarii insuficient cunoscute deorece investitorul nu poate preciza în mod obiectiv toate misiunile sistemului pe toată durata de viaţa a acestuia. Constructorii încearcă să facă faţă acestor cerinţe contradictorii prin crearea de sisteme caracterizate prin complexitate structurală deosebită, flexibilitate şi distribuire.

Dezvoltarea spectaculoasă a tehnologiei informatice face posibilă construirea de echipamente inteligente apte de prelucrări informaţionale sofisticate şi de comunicaţii în mai multe direcţii şi cu mai multe niveluri ierarhice fiind apte de a realiza diverse funcţionalităţi. Astfel de echipamente pot sta la baza sistemelor de automatizări industrale moderne cu condiţia ca proiectantul sistemului să dispună de metodele, procedeele şi instrumentele necesare.

Pentru stăpânirea complexităţii structurale a sistemelor, lucrarea de faţă propune următoarele trei principii de structurare, bazate pe noţiuni de modelare semantică:

1. *principiul decompoziţiei* respectiv modelarea orientată pe obiecte, prin care sistemele şi subsistemele se divizează în părţi componente;
2. *principiul abstractizării funcţionale*, prin care se realizează structurarea pe niveluri ierarhice a sistemelor;
3. *principiul transformării* sau modelarea orientată pe evenimente, prin care se descriu transformările de stare.

Lucrarea propune ca sistemele de automatizări industriale complexe să fie concepute, modelate, analizate şi implementate ca sisteme dinamice cu stări discrete pilotate de evenimente. Reţelele Petri oferă formalismul matematic necesar modelării şi analizei unor astfel de sisteme. Lucrarea prezintă, într-o formă inginerească, noţiunile de bază precum şi tehnicile de modelare şi analiză pe baza reţelelor Petri.

Lucrarea introduce noţiunile de *operator de tranziţie* şi *maşină virtuală de reţea Petri* şi demonstrează faptul că pentru a putea elabora mărimi de comandă în concordanţă cu relaţiile cauzale din procesul condus, orice sistem de conducere, realizat pe baza unui model de reţea Petri, trebuie să implementeze o maşină virtuală de reţea Petri.

Conceptul de maşină virtuală de reţea Petri permite structurarea clară a sistemului de conducere punând în evidenţă şi delimitând elementele funcţionale componente: (i) maşina virtuală de reţea Petri, (ii) observatorii de evenimente, (iii) strategia de conducere pe baza reacţiei după stare, şi (iv) operatorii de tranziţie.

Conceptul de maşină virtuală de reţea Petri permite enunţul unei condiţii de *controlabilitate primară* a proceselor modelate prin reţele Petri. Controlabilitatea în sensul clasic, relativ la comportamentul dorit al sistemului, poate fi asigurată doar dacă se asigură controlabilitatea primară exprimată în lucrare la paragraful 7.6.

Pe baza noţiunii de maşină virtuală de reţea Petri lucrarea propune soluţii pentru următoarele categorii de probleme:

1. Sinteza strategiei de conducere pentru cazul în care procesul este condus pe baza reacţiei după stare. Sinteza se realizează pe baza unor restricţii asupra stărilor realizabile ale procesului condus respectiv asupra marcajelor accesibile ale modelului de reţea Petri al procesului condus.

2. Definirea interfeţelor dintre subreţelele rezultate prin partiţionarea unei reţele Petri. Definiţia dată interfeţelor permite specificarea structurilor de conducere ale subreţelelor astfel încât controlul distribuit rezultat să se execute pe baza relaţiilor cauzale din procesul condus.
3. Definirea structurilor de conducere în cazul unei structuri ierarhice a modelului de reţea Petri al procesului condus respectiv a unei structuri ierarhice a sistemului de conducere.
4. Definirea interfeţelor şi a structurilor de conducere în cazul unor reţele Petri modulare.

INDEXUL PRINCIPALELOR ABREVIERI ŞI NOTAŢII UTILIZATE

a) Index de abrevieri

CPN	- reţea Petri colorată (Coloured Petri Net)
CtlPN	- reţea Petri controlată
FB	- funcţiune de bază
MMI	- interfaţa om-maşină (Man Mashine Interface)
MSOO	- modelarea semantică orientată pe obiecte
MVRP	- maşina virtuală de reţea Petri
OPT	- operator de tranziţie
POO	- programarea orientată pe obiecte
Reţea C/E	- reţea Petri Condiţii/Evenimente (Condition/Event Net)
Reţea P/T	- reţea Petri Locaţie/Tranziţie (Position/Transition Net)
Reţea mPTN	- reţea Petri Locaţie/Tranziţie marcată (marked Position/Transition Net)
SC	- sistem de conducere

b) Index de notaţii

Mulţimi

A, B, ...	- mulţimi
Z	- mulţimea numerelor întregi
N	- mulţimea numerelor naturale
N^+	- mulţimea numerelor naturale nenegative
R^+	- mulţimea numerelor reale nenegative
$\subseteq$	- submulţime
$\in$	- elementul aparţine mulţimii date
$\notin$	- elementul nu aparţine mulţimii date
$\cup$	- reuniune
$\cap$	- intersecţie
$\times$	- produs cartezian
$\setminus$	- diferenţa a două mulţimi
$\varnothing$	- mulţimea vidă
$\{ x \mid \ldots \}$	- mulţimea elementelor x pentru care este (sunt) valabile ...
$\|A\|$	- cardinalitatea mulţimii A
A^n	- mulţimea n-tuplurilor de elemente ale mulţimii A
$f : A \rightarrow B$	- funcţie

Operatori logici

$\forall$	- pentru oricare ...
$\exists$	- există un (o) ... pentru care ...
$/\exists$	- nu există un (o) ... pentru care ...
$\vee$	- disjuncţie (SAU logic)
$\wedge$	- conjuncţie (ŞI logic)
$\Leftarrow$	- implicaţie prin condiţii necesre
$\Rightarrow$	- implicaţie prin condiţii suficiente

$\Leftrightarrow$ - echivalenţă
□ - q.e.d.

Algebra liniară

$a, b, \ldots$ - mărimi scalare
$\mathbf{a}, \mathbf{b}, \ldots$ - vectori coloană
$\mathbf{a}^T, \mathbf{b}^T, \ldots$ - vectori linie
$\mathbf{a}[i]$ - elementul i al vectorului $\mathbf{a}$
$\mathbf{A}, \mathbf{B}, \ldots$ - matrici
$\mathbf{A}^T, \mathbf{B}^T, \ldots$ - matrici transpuse
$\mathbf{A}^{-1}, \mathbf{B}^{-1}, \ldots$ - inversele matricelor A respectiv B

Teoria grafurilor

D - graf orientat
D^K - condensata grafului orientat D
Q(D) - mulţimea nodurilor grafului orientat D
A(D) - mulţimea arcelor grafului orientat D
τ - funcţia de etichetare a arcelor grafului

Reţele Petri

N - reţea, reţea Petri
$P; p_i$ - mulţimea locaţiilor; locaţie
$T; t_i$ - mulţimea tranziţiilor; tranziţie
F - mulţimea arcelor (relaţia de flux)
W - funcţia de ponderare a arcelor
$K; \mathbf{k}$ - funcţia de capacitate; vectorul capacităţilor
$M_0; \mathbf{m}_0$ - marcajul iniţial; vectorul marcajului iniţial
$M; \mathbf{m}$ - marcaj; vectorul marcajului
$\mathbf{N}; \mathbf{N}^+; \mathbf{N}^-$ - matrici de incidenţe
$\mathbf{t}_j; \mathbf{t}_j^+; \mathbf{t}_j^-$ - matrici ale tranziţiilor
$\mathbf{i}_P$ - P-invariant
$\mathbf{i}_T$ - T-invariant
${}^\bullet x$ - premulţimea nodului x
$x^\bullet$ - postmulţimea nodului x
σ - secvenţă de tranziţii
$R_N(M_0)$ - mulţimea de accesibilitate a reţelei N
GA_N - graful de accesibilitate a reţelei N
GA^K - condensata grafului de accesibilitate
$C^j; \mathbf{c}^j$ - marcaj concesionant; vector concesionant
$D^j; \mathbf{d}^j$ - marcaj deconcesionant; vector deconcesionant
Δ^j - matricea diferenţă pentru tranziţia t_j
$p_i; p_i'$ - locaţii complementare
$t_j; t_j'$ - tranziţii congruente

1. Introducere

Motto:

"Abandonment of causality as a matter of principle should be permitted only in the most extreme emergency"

ALBERT EINSTEIN, 1924[1]

Acest capitol prezintă caracteristicile definitorii ale sistemelor de automatizări industriale precum şi contextul tehnic şi tehnologic în care se dezvoltă aceste sisteme. Se arată importanţa metodelor de structurare, modelare, analiză şi sinteză în implementarea sistemelor de automatizare distribuite. Se prezintă obiectivele lucrării şi structura pe capitole a lucrării.

1.1 Caracteristici definitorii ale sistemelor moderne de automatizări industriale

Experienţa istorică arată că bunăstarea societăţii omeneşti poate fi creată şi menţinută doar prin creşterea economică. Creşterea economică este însoţită de crearea şi extinderea sistemelor artificiale mari. Acestor sisteme artificiale mari li se impun cerinţe de înaltă performanţă pentru ca acestea să poată evolua în scenarii insuficient cunoscute, realizând însă nivele prescrise pentru indicatorii de performanţă.

Dimensiunea principală a sistemelor mari, care încearcă să răspundă acestor cerinţe, este ***complexitatea structurală***, caracterizată, în special, prin calitatea şi numărul legăturilor dintre elementele componente.

De cele mai multe ori constructorii unor astfel de sisteme nu dispun de informaţie apriorică completă asupra sistemului ce se doreşte a fi realizat. Această stare se datorează faptului că investitorul nu poate preciza în mod obiectiv toate misiunile sistemului pe toată durata de viaţă a acestuia.

Sistemele moderne de producţie încearcă să facă faţă cerinţelor ce li se impun prin ***flexibilitate***. În domeniul producţiei industriale, termenul de flexibilitate apare în legătură cu automatizarea fabricaţiei, ca trăsătură ce defineşte un sistem automat de fabricaţie, bazat pe maşini transformabile, atât pentru procesele de prelucrare, cât şi pentru cele de transport şi manipulare a materialelor. Flexibilitatea presupune în egală măsură şi capacitatea unui sistem de producţie de a trece la fabricaţia de produse de diferite tipuri – ca urmare a existenţei unei elasticităţi a structurilor tehnice – dar şi capacitatea de a reacţiona la cerinţele, în mereu schimbare, ale pieţii – ca urmare a unei elasticităţi comerciale.

Conform definiţiei date în lucrarea [Coj'90b], sistemele flexibile de fabricaţie sunt acele sisteme tehnice care asigură automatizarea fabricaţiei de serie datorită faptului că structura sistemului dispune ***de calitatea de integrabilitate***, posedă ***adaptabilitate*** faţă de

[1] A. Pais – "Subtle is the lord ..." – The science and the life of Albert Einstein. Oxford University Press, 1983

un domeniu de sarcini, este ***adecvabilă tehnic și economic*** fiecărei sarcini și este construită pe baza unei ***concepții dinamice***.

Calitatea de integrabilitate este necesară deoarece într-un sistem de fabricație automatizat, destinat mai multor tipuri de sarcini, există un număr important de sisteme parțiale ce trebuie integrate fizic, funcțional și temporal. Caracteristica de integrabilitate implică o corelare crecută între diferitele sisteme parțiale și diferitele funcțiuni, corelare ce trebuie avută în vedere încă de la conceperea acestora, ca o caracteristică calitativă specifică și definitorie.

Calitatea de adaptabilitate trebuie să asigure ca aceleași mijloace de fabricație să poată realiza sarcini diferite, necesitând operații diferite. Gradul de adaptabilitate trebuie să fie proporțional cu gradul de diversificare tehnologică și cu viteza cu care adaptarea trebuie să se producă. Adaptarea se realizează prin generarea (materializarea) diferitelor posibilități de lucru dintr-un spectru de posibilități potențiale, prin modificarea structurii cu sau fără ajutorul unor dispozitive auxiliare.

Adecvarea este calitatea prin care se asigură că dacă, pe baza adaptabilității, se selecționează o posibilitate de lucru, pentru realizarea unei operații (a unei acțiuni) atunci mijlocul de fabricație poate fi maximal adecvat, tehnic și economic, pentru realizarea acesteia.

Concepția dinamică trebuie să asigure posibilitatea ca structura unui sistem flexibil automatizat să poată fi modificată ca urmare a apariției unor noi sarcini de producție. Astfel de sisteme sunt alcătuite din ***elemente modulare*** care, la rândul lor, fac parte din ***familii de module***. Concepția dinamică va permite restructurarea unui astfel de sistem, prin înlocuirea unor module în vederea obținerii adecvării și adaptării, înlocuirea fiind posibilă pe baza calității de integrabilitate [Coj'90b].

Pentru realizarea efectivă a flexibilității sistemului de automatizare constructorul acestuia trebuie să dispună de metode de specificare, modelare, analiză și sinteză care să permită o inginerie efectivă a acestor sisteme [SVa]. Aceste metode trebuie să sprijine în mod efectiv comunicația dintre investitor, proiectant și operator.

1.2 Contextul în care se dezvoltă sistemele moderne de automatizări industriale

Flexibilizarea sistemelor de automatizări industriale este posibilă datorită dezvoltării spectaculoase a tehnologiei informatice. Microprocesoarele sunt utilizate în practic toate echipamentele și aparatele componente ale sistemelor moderne de automatizare. Aceste echipamente inteligente sunt apte de prelucrări informaționale sofisticate și de comuncații în mai multe direcții și cu mai multe niveluri ierarhice fiind apte de a realiza diverse funcționalități. Pe baza unor astfel de echipamente se realizează sisteme cu prelucrare distribuită sau pe scurt ***sisteme distribuite***.

Sistemele distribuite se cracterizează prin următoarele [PAG'93]:

1) ***Multitudinea resurselor fizice și logice***, prin care se realizează cerințele funcționale impuse sistemului dispuse într-o structură topologico-spațială oarecare. Modularitatea realizabilă prin echipamente inteligente precum și o arhitectură modulară pot contribui la asigurarea unor calități deosebite în privința extensibilității (incrementale), flexibilității și disponibilității.

2) ***Distribuirea fizică a componentelor sistemului***: impune existența unui suport de comunicație care să permită distribuirea fizică și intercomunicarea între componentele fizice și logice în interacțiune și înglobarea lor într-un sistem unitar. Cel mai important aspect privitor la distribuirea fizică și intercomunicarea dintre componente

îl constituie transferul explicit de mesaje, pe bază de protocoale de comunicaţie, unitatea de transfer în sistem fiind mesajul, chiar dacă entităţile sunt corezidente. O consecinţă importantă a acestui fapt o constituie caracterul variabil al întârzierilor de transfer al mesajelor, cu implicaţii esenţiale asupra ***obsevabilităţii şi cronologiei evenimentelor*** şi a realizării ***controlului global al sistemului*** [PAG'93].

3) ***Viziunea globală, unitară asupra sistemului***; exprimă faptul că funcţionarea unei multitudini de componente impune necesitatea existenţei unei strategii unificatoare globale, care să integreze componentele fizice şi logice distribuite, într-un întreg operaţional. O astfel de strategie trebuie materializată prin componente logice de control, care la rândul lor trebuie să dispună de informaţii de stare necesare. ***Este necesar un model de reprezentare a evoluţiei unui astfel de sistem în timp.*** Aceste modele se bazează pe ipoteza că este posibilă descrierea completă a stării in anumite momente determinate. O astfel de stare este numită ***punct observabil***, evoluţia sistemului materializându-se prin parcurgerea punctelor observabile. Un eveniment este atunci definit printr-o modificare a unei submulţimi a stării şi poate fi datat printr-un punct observabil.

4) ***Independenţa relativă, cooperatistă a sistemului;*** exprimă funcţionalitatea independentă (locală) a componentelor sistemului şi, în acelaşi timp, interacţiunea lor atât la nivel fizic cât şi la nivel logic. Interacţiunea se realizează prin intermediul suportului de comunicaţie. Interacţiunea dintre componentele logice este independentă de localizarea lor fizică. Coexistenţa calităţilor contradictorii de independenţă funcţională şi interacţiune se asigură printr-o strategie globală implementată printr-un nucleu integrator.

5) ***Transparenţa structurii fată de interfaţa de utilizare a acestuia.*** Este calitatea prin care se asigură transparenţa cererilor utilizatorului în raport cu structura concretă a sistemului. Acest criteriu presupune existenta unei interfete de acces la facilităţile sistemului, care să realizeze, la nivelul utilizatorului, abstractizarea de tip "procesor unic".

Implementarea sistemelor de automatizare distribuite .este efectiv sprijinită de protocoalele de comunicaţie standardizate specifice domeniului [Pfe'86], [Bûs'86], cum ar fi: PROFIBUS DP, PROFIBUS AMS, INTERBUS S, CAN Bus, ASI Bus şi chiar Ethernet cu TCP/IP. Sistemele MMI – Man Machine Interface – moderne oferă interfeţe utilizator grafice deosebit de ergonomice, cu multiple funcţionalităţi.

Modelarea semantică orientată pe evenimente şi formalismul reţelelor Petri pot constitui baza unei metodologii care, împreună cu tehnicile clasice înrădăcinate, să ofere instrumentele de inginerie necesare în realizarea sistemelor de automatizări industriale.

1.3 În cadrarea temei în actualitatea ştiinţifică

Lucrarea [Sch'91b] avansează ideea necesităţii unei teorii integratoare a automatizărilor industriale. O astfel de teorie integratoare este necesară din cauza caracterului interdisciplinar al problemelor cu care se confruntă automatizarea industrială modernă.

La baza unei astfel de teorii ar trebuie să stea următoarele două principii:

1. concepţie conştient sistemică bazată pe axiome ce caracterizează sistemele cu care se lucrează;
2. formularea cauzalităţii înaintea temporalităţii.

Concepţia sistemică bazată pe axiome sistem trebuie să sprijine în mod efectiv structurarea sistemlor de automatizare industrială. Pentru aceasta este necesară elaborarea

unor principii şi tehnici de structurare care să sprijine înţelegerea, modelarea şi analiza sistemelor. Este de dorit ca aceste tehnici să ofere o descriere formală, intuitivă a structurii sistemelor astfel încât să sprijine în mode eficient comunicaţia dintre diverşii subiecţi ce vin în contact cu sistemul (investitor, proiectant, operator, etc).

În cazul sistemelor mari, complexe şi spaţial distribuite ipoteza unai ordonări totale, obiective (independente de observator) a evenimentelor exclude posibilitatea unei modelări realiste. Din acest motiv în cazul unor astfel de sisteme cauzalitatea trebuie formulată înaintea temporalităţii ceea ce înseamnă că la baza modelării şi analizei sistemului vor sta relaţiile cauzale din cadrul acestuia. O astfel de abordare este posibilă pe baza conceptului de sistem dinamic cu evenimente discrete definit prin următoarele două proprietăţi:

(i) Spaţiul stărilor este o mulţime discretă;

(ii) Mecanismul de tranziţie a stărilor este pilotat de apariţia (asincronă) a evenimentelor.

Investigarea unor astfel de sisteme necesită instrumente matematice proprii, specifice. Formalismul reţelelor Petri netemporizate oferă unul din cele mai performante instrumente în acest scop.

1.4 Obiectivele lucrării

Mijloacele tehnice puse la dispoziţie de tehnologia informaţională permit realizarea de sisteme de automatizare distribuite şi flexibile. Pentru implementarea cu succes a unui astfel de sistem proiectantul trebuie să dispune de metode adecvate de structurare, modelare, analiză şi sinteză.

Lucrarea de faţă şi-a propus să sprijine elaborarea unor astfel de metode prin următoarele:

- să prezinte principiile de bază ale structurării sistemelor de automatizări industriale precum şi tehnicile formale de specificare a structurării;
- să prezinte importanţa modelării pe baza unei relaţii netranzitive dar reflexive şi a unei relaţii de ordonare parţială (relaţia cauzală);
- să introducă tehnica modelării prin reţele Petri într-o formă inginerească dedicată utilizării în modelarea sistemelor de automatizare;
- să prezinte tehnicile de bază de analiză a reţelelor Petri precum şi tehnici de bază de corecţie a reţelei;
- să introducă o nouă interpretare a reţelelor Petri astfel încât reţeaua interpretată să poată sta la baza implementării unui sistem de conducere;
- să introducă tehnica de determinare a stării procesului condus, pe baza modelului de reţea Petri;
- să prezinte structuri de conducere bazate pe modelul de reţea Petri împreună cu studiul caracteristicilor acestora;
- să prezinte aspectele de implementare ale sistemului de comandă, pe baza modelului de reţea Petri;
- să prezinte o metodă de control distribuit a sistemelor modelate prin reţele Petri;

- să prezinte tehnicile de structurare ierarhică a reţelelor Petri precum şi structurile de reţea pe baza cărora se vor implementa subsistemele de conducere ale unei structuri ierarhice de conducere;
- să prezinte principiile modularizării reţelelor Petri şi implicit a sistemului modelat respectiv să prezinte structurile de conducere ale unui astfel de sistem modular.

1.5 Sinopsis

Lucrarea este structurată după cum urmează:

Capitolul 2 prezintă, din punctul de vedere al ingineriei sistemelor, principiile modelării conceptuale şi trei principii de structurare a sistemelor de automatizări industriale: (1)-principiul decompoziţiei respectiv modelarea orientată pe obiecte, prin care sistemele şi subsistemele se divizează în părţi componente, (2)-principiul abstractizării funcţionale, prin care se realizează structurarea pe niveluri ierarhice a sistemelor, (3)-principiul transformării sau modelarea orientată pe evenimente, prin care se descriu transformările de stare. Se prezintă şi modul în care aceste principii se utilizează în modelarea sistemelor de atomatizări industriale.

Capitolul 3 prezintă, în prima parte, noţiunile fundamentale legate de modelarea discretă, orientată pe evenimente a sistemelor distribuite. Se prezintă rolul relaţiilor de ordonare parţială care stau şi la baza teoriei reţelelor Petri. În partea a doua a capitolului se prezintă, într-o formă inginerească adaptată nevoilor modelării sistemelor de automatizare, clasa reţelelor Petri Condiţii/Evenimente şi Locaţie/Tranziţie.

Capitolul 4 prezintă tehnicile de bază ale analizei reţelelor Petri Locaţie /Tranziţie marcate. Scopul analizei este de a pune în evidenţă proprietăţile de viabilitate, reversibilitate şi mărginire. Analiza poate fi efectuată pe două căi fundamental diferite: (i) analiza bazată pe graful accesibil şi (ii) analiza bazată pe invarianţi.

Capitolul 5 prezintă o metodă de realizare a corecţiilor de reţea. Corecţiile de reţea sunt necesare în cazul în care analiza reţelei nu a confirmat existenta proprietăţilor dinamice impuse reţelei. Corecţiile se realizează prin bucle de reţea suplimentare prin care se realizează restricţii suplimentare de concesionare. În acest sens, se prezintă o metodă de determinare a locaţiilor semnificative la care se vor conecta buclele de reţea. Se prezintă şi o metodă de verificare a corectabilităţii reţelei.

Capitolul 6 prezintă un studiu asupra efectelor tranziţiilor temporizate în modelarea cu reţele Petri. Ca şi concluzie a acestui studiu se enunţă principiul încapsulării proceselor de arbitrare a resurselor. Se introduce o metodă de interpretare a reţelelor Petri. Se introduce noţiunea de "maşină virtuală de reţea Petri" care stă la baza unei noi metode de determinare a stării procesului condus şi de implementare a sistemelor de comandă. Se prezintă un studiu asupra caracteristicilor sistemelor de comadă implementate pe baza maşinii virtuale de reţea Petri.

Capitolul 7 defineşte o structurare a vectorului de comandă şi prezintă două metode de sinteză a strategiei de comandă pentru cazul în care procesul este condus printr-o reacţie după stare. Sinteza se realizează pe baza unor restricţii asupra stărilor realizabile ale procesului condus respectiv asupra marcajelor accesibile ale modelului de reţea Petri al procesului condus.

Capitolul 8 prezintă o metodă de partiţionare a reţelelor Petri şi o definiţie a interfeţelor dintre subreţelele rezultate care permite, pe baza conceptului de maşină virtuală de reţea Petri, specificarea structurilor de conducere a subreţelelor rezultate.

Capitolul 9 prezintă o sinteză a procedeelor de structurare ierarhică a reţelelor Petri şi propune o nouă metodă de reprezentare a subreţelelor în reţeaua de bază. Această metodă prezintă unele avantaje, faţă de alte metode cunoscute, avantaje ce fac posibilă definirea structurii de reţea pe baza căreia, cu ajutorul conceptului de maşină virtuală de reţea Petri, poate fi sintetizată o structură de conducere necesară pentru implementarea echipamentului de conducere a subreţelei.

Capitolul 10 prezintă, pentru cazul modelării prin reţele Petri, conceptele modularizării şi interfaţării dintre module. Modularitaea este o caracteristică de bază a sistemelor flexibile de fabricaţie. Această modularitate trebuie să se reflecte şi asupra modelului de reţea Petri al sistemului. Sistemele flexibile de fabricaţie se modelează deosebit de eficient prin reţele Petri colorate. Se prezintă definiţia formală şi sintaxa reţelelor Petri colorate. Se arată că şi în acest caz sunt utilizabile conceptele maşinii virtuale de reţea Petri. Se prezintă modul cum se determină reţelele Petri necesare analizei diverselor module respectiv reţelele Petri pe baza cărora se implementează echipamentele de comandă ale acestor module.

Capitolul 11 prezintă o privire retrospectivă asupra lucrării precum şi principalele concluzii şi contribuţiile autorului. Se prezintă şi câteva consideraţii referitoare la posibilităţile de continuare a ideilor prezentate în această lucrare.

Anexa 1. Prezintă corespondenţa dintre elementele abstracte ale reţelei Petri şi obiecte ale sistemelor tehnice din diferite domenii

2. Structurarea sistemelor de conducere

Acest capitol prezintă, din punctul de vedere al ingineriei sistemelor, principiile modelarii conceptuale şi trei principii de structurare a sistemelor de automatizări industriale: (1)-principiul decompoziţiei respectiv modelarea orientată pe obiecte, prin care sistemele şi subsistemele se divizează în părţi componente, (2)-principiul abstractizării funcţionale, prin care se realizează structurarea pe niveluri ierarhice a sistemelor, (3)-principiul transformării sau modelarea orientată pe evenimente, prin care se descriu transformările de stare. Se prezintă şi modul în care aceste principii se utilizează în modelarea sistemelor de atomatizări industriale.

Crearea şi exploatarea sistemelor de automatizare industrială este determinată de cadrul general al organizării şi modelării intreprinderii ca sistem. Raporturile dintre intreprinderea industrială şi mediul său exterior sunt mijlocite prin categorii globale ca: *generozitatea în resurse a mediului, comanda socială, sisteme particulare de restricţii supraierarhizate resurselor şi gradul de satisfacere a comenzii sociale* [CPI'88]. Echilibrul dintre aceste categorii va implica o anumită structură a intreprinderii industriale, în cazul în care aceasta este considerată ca instrument de reglaj.

Particularizarea raporturilor dintre intreprinderea industrială şi mediul în care aceasta evoluează implică deopotrivă modelarea intrepriderii ca sistem, cât şi a mediului, astfel încât să se pună în evidenţă mulţimea legăturilor mediu-intreprindere, cît şi mulţimea determinărilor pe care aceste legături le prezintă asupra structurii intreprinderii. Utilitatea unei astfel de modelări va depinde de capacitatea de a le stăpâni şi a le adapta cerinţelor ce variază în raport cu timpul.

Pentru a face faţă cerinţelor ce li se impun, intreprinderile industriale apelează la serviciile sistemelor de automatizare, din ce în ce mai complexe din punct de vedere structural şi al numărului şi calităţii legăturilor dintre elementele componente. Complexitatea derivă din cerinţele de înaltă performanţă legate de flexibilitate, extensibilitate, şi mentenabilitate care trebuie să fie asigurate în scenarii insuficient cunoscute datorită informaţiei apriorice incomplete asupra misiunii şi mediului pe toată durata de viaţă, care trebuie să fie lungă, conform cerinţei investitorului.

Problemele ridicate de aceste sisteme de automatizare pot fi rezolvate doar printr-o viziune sistemică asupra obiectivelor, funcţiunilor şi mijloacelor de realizare, într-un cadru conceptual unitar oferit de disciplina ingineriei sistemelor, [Pol'89a][Pol'89b]. Ingineria sistemelor se defineşte ca disciplina tehnică care se ocupă cu determinarea şi aplicarea celor mai adecvate soluţii, metode, procedee şi instrumente care să conducă, în condiţii optime de rentabilitate şi eficienţă, la crearea respectiv exploatarea unui sistem tehnic care satisface toate cerinţele impuse prin specificaţia de definire.

La confruntarea cu sisteme mari, complexe, în mod instinctiv se încearcă să se descopere structura lor şi apoi se încearcă exploatarea unor caracteristici ale structurii pentru a găsi o cale - realizabilă - pentru a controla sistemul şi a proceda în consecinţă [Zab'85]. Structura înseamnă, de regulă, posibilitatea de a distinge subsisteme şi o varietate de legături sau interacţiuni între acestea, bine definite. Există un spectru larg de interconexiuni între subsisteme. Cele două extreme, de importanţă deosebită, sunt:

1. Structura pur ierarhizată, în care subsistemele sunt aranjate pe niveluri ierarhice, interacţiunile formând un graf de tip arbore pur. Aceasta reprezintă o structură verticală rigidă de comandă şi raportare.
2. Structura în reţea, constând din subsisteme de nivel ierarhic egale, fiecare interacţionând direct cu un număr limiat de subsisteme, de regulă cu cele vecine.

Există o gamă aproape continuă de structuri intermediare între aceste două extreme.

Stăpânirea complexităţii sistemelor nu poate fi concepută fără utilizarea tehnicilor informaţionale. Fără "liantul informatic" nu pot fi coordonate mijloacele necesare atingerii obiectivelor intreprinderii, chiar dacă aceste mijloace există în cantităţi suficiente. În aceste condiţii performanţele întregului sistem vor fi sensibil determinate de calitatea sistemelor de comunicaţie şi a celor de prelucrare a datelor dar, în mod deosebit, şi de structurile de date cu care acestea lucrează.

Metodele de structurare ale sistemelor de automatizare vor fi astfel strâns legate de metodele de structurare a datelor. În literatura de specialitate se prezintă mai multe metode de structurare respectiv modelare.

În cazul în care structurarea se realizează prin definirea de "obiecte" naturale sau abstracte ale sistemului se realizează structurarea sau modelarea după obiecte - aşa cum se prezintă în [Pol'89a], [Pol'89b] - sprijinită de limbajele de programare orientate pe obiecte [Jur'92]. Acest principiu de modelare utilizează, pentru descrierea caracteristicilor calitative şi cantitative ale sistemelor, reţele de clase şi reţele de agregare.

Sistemele tehnice pot fi modelate şi pe baza unei ierarhii funcţionale [Coj'90a], [Pol'89a], [Pol'89b]. În acest caz se definesc niveluri funcţionale - pe baza unei ierarhii - respectiv reţele de module funcţionale şi interacţiunile dintre aceste module.

În cazul modelului pe faze al producţiei se creează o reţea care scoate în evidenţă relaţiile cauzale în procesul de transformare a materialelor pe parcursul realizării produsului finit [Pol'89a], [Pol'89b]. Această tehnică de modelare se bazează pe metodele de modelare şi analiză oferite de teoria reţelelor Petri [Pol'89a], [Pol'89b], [CPI'88].

2.1 Preliminarii

2.1.1 Delimitarea problematicii

Automatizarea industrială poate fi definită ca o soluţie tehnică care, pentru procesele din sistemele tehnice, asigură o execuţie autonomă şi în conformitate cu cerinţele impuse. Prin soluţie tehnică se înţelege atât execuţia tehnică cât şi metodele prin care, plecând de la specificaţia de definire, se asigură implementarea sistemului tehnic în toate detaliile acestuia.

În cursul ciclului de viaţă al sistemului de automatizare vin în contact cu acesta un număr mare de subiecţi: proiectanţi, utilizatori, operatori. Toţi aceşti subiecţi îşi formează puncte de vedere personale îndreptăţite, mai mult sau mai puţin divergente prin conţinut şi formă de prezentare. Sistemul de automatizare va realiza obiectivele pentru care s-a proiectat doar dacă sistemul implementat va asigura consistenţa dintre aceste puncte de vedere.

Divergenţa punctelor de vedere se datorează:

- diversităţii obiectivelor automatizate;
- diversităţii aspectelor aparative (hardware);
- diversităţii aspectelor de programare (software);

- diversităţii aspectelor de proiectare;
- diversităţii aspectelor funcţionale, ş.a.

În cazul unui sistem concret aceste aspecte apar în diverse combinaţii caracteristice sistemului dat.

Prin *sistem de comandă* sau *sistem de conducere* se va înţelege un sistem prin care poate fi influenţat un proces tehnic în conformitate cu scopuri predefinite. Mijloacele tehnice utilizate (aparate, echipamente) se leagă de procesul condus prin senzori şi elemente de execuţie. Procesul condus împreună cu sistemul de comandă formează sistemul de automatizare. Figura 2-1 prezintă elementele fizice şi conceptuale de bază legate de sistemele de automatizarea precum şi legăturile dintre acestea.

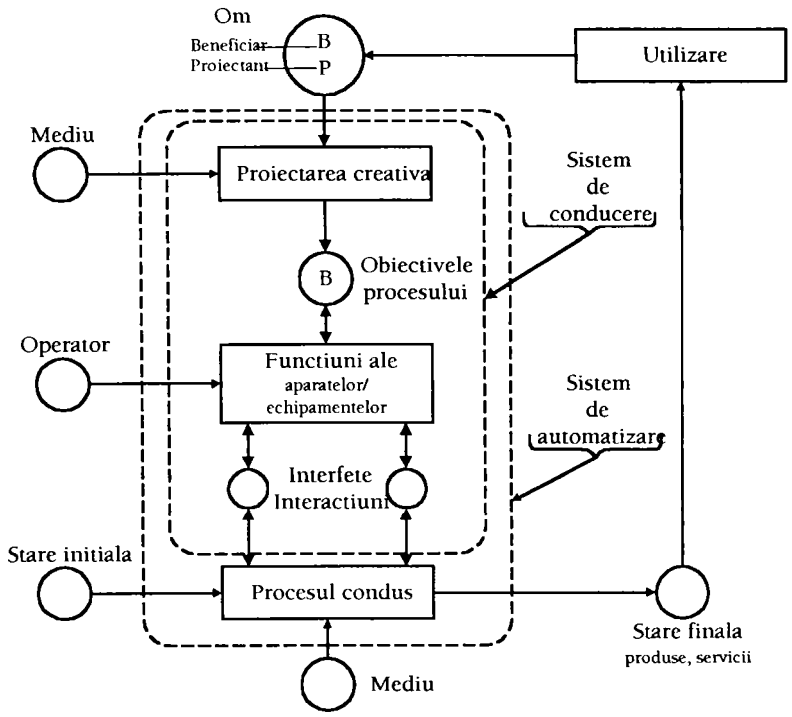


Fig. 2-1 Elementele de bază, fizice şi conceptuale, legate de sistemele de automatizare

Omul vine în contact cu sistemul de automatizare în calitate de Beneficiar, Proiectant sau Operator, în sensul cel mai larg al acestor cuvinte. Beneficiarul şi proiectantul determină caracteristicile sistemului de automatizare iar operatorul controlează funcţionarea corectă a acestuia.

Procesul de proiectare al sistemului este determinat de:

1. obiectivele formulate de către Beneficiar;
2. cunoştinţele, metodele de lucru şi experienţa Proiectantului;

3. mediul tehnic, economic şi social în care se realizează proiectul respectiv în care va evolua sistemul de automatizare.

Obiectivele procesului, furnizate de către Beneficiar, vor fi realizate cu ajutorul funcţiunilor puse la dispoziţie de către aparatele şi echipamentele sistemului de comandă sub supravegherea Operatorului. Sistemele de comandă moderne trebuie privite ca sisteme evolutive. Performanţele aparatelor şi echipamentelor pot fi continuu îmbunătăţite. Acest lucru va avea efect şi asupra obiectivelor formulate de către Beneficiar.

Sistemul de comandă interacţionează cu procesul condus, prin interfeţe bine definite, cu ajutorul senzorilor şi a elementelor de execuţie.

Datorită acţiunii sistemului de comandă, în conjuncţie cu acţiunea mediului, procesul condus va fi trecut din starea iniţială în starea finală.

Experienţa câştigată prin utilizarea/exploatarea sistemului de automatizare va determina formularea, de către Beneficiar, de noi obiective respectiv introducerea unor noi tehnici şi metode de proiectare ceea ce va determina crearea unor noi sisteme de comandă.

> **Automatizarea industrială**, ca disciplină tehnică, trebuie să asigure determinarea şi aplicarea celor mai adecvate soluţii, metode, procedee şi instrumente care să conducă, în condiţii optime de rentabilitate şi eficienţă, la crearea respectiv exploatarea unui sistem tehnic care satisface specificaţia de definire.

Problemele legate de sistemele de automatizare, din ce în ce mai complexe, pot fi rezolvate doar printr-o viziune sistemică asupra lor, respectiv printr-un cadru conceptual unitar. Pentru aceasta, în cele ce urmează se prezintă axiomele-sistem utilizate în cadrul sistemlor de automatizare, sintetizate pe baza lucrărilor [Sch '91b],[Sch'94].

2.1.2 Axiome-sistem

La confruntarea cu sisteme mari, complexe, în mod instinctiv se încearcă să se descopere structura lor şi apoi se încearcă exploatarea unor caracteristici ale structurii pentru a găsi o cale de a controla sistemul. Pentru aceasta se definesc următoarele două axiome structurale.

a) Axiome structurale

Axioma 1. *Principiul structural*

Un sistem este un complex unitar de părţi sau elemente ce au funcţii bine definite şi ocupă în cadrul sistemului poziţii bine determinate ceea ce permite să se afirme că sistemul se caracterizează printr-o *structură*.

Pentru un sistem este esenţial faptul că părţile sale componente sunt într-o anumită relaţie între ele şi cu mediul înconjurător.

Aceste interrelaţii imprimă sistemului calităţi specifice ce definesc identitatea sistemului şi ce constituie criteriul de delimitare faţă de mediul înconjurător. Sistemul opune rezistenţă faţă de acţiunile mediului înconjurător, în sensul conservării acestor calităţi.

Axioma 2. *Principiul decompoziţiei*

Părţile componente ale unui sistem pot să fie *subsisteme* în sensul că la un nou nivel de detaliere acestea prezintă caracteristici sistemice. Sistemul iniţial, ca reuniune a părţilor componente, prezintă calităţi ce nu pot fi identificate în părţile sale luate separat.

b) Axiomele schimbării

Din punct de vedere conceptual şi practic sunt interesante sistemele dinamice (sistemele ce evoluează în timp). Schimbările se raportează la *starea* sistemului, caracterizată prin *mărimi de stare*. Ceea ce se schimbă în timp este tocmai starea sistemului. Schimbarea se concretizează în procesarea substanţei, energiei şi informaţiei în conformitate cu legile generale ale naturii.

Axioma 3. *Principiul cauzalităţii*

Schimbările de stare din sistemele fizice (tehnice) au loc conform principiului general al cauzalităţii. Relaţia cauză→efect este o relaţie binară şi unidirecţională în sensul că nici un efect nu poate influenţa cauza care l-a produs (l-a determinat). Se porneşte de la prezumpţia că relaţia cauză→efect este şi stabilă în sensul că aceeaşi cauză va produce întotdeauna acelaş efect.

Axioma 4. *Principiul temporal*

Sistemul este alcătuit dintr-o mulţime de părţi componente a căror stare şi structură respectiv interacţiunea dintre ele se modifică în timp. Toate schimbările de stare sau structurale se ordonează în timp în sensul trecut→prezent→viitor.

În legatură cu un sistem se pot formula următoarele caracterizări:

1. Noţiunea de sistem este relativă, deorece una şi aceeaşi realitate fizică poate cuprinde diverse sisteme, corelate sau nu între ele.
2. Legăturile de cauzalitate pot fi astfel ordonate încât în cadrul sistemului să existe legături inverse, relaţii specifice sistemelor cibernetice (naturale sau tehnice).
3. În cazul unui sistem tehnic acţiunea comună a părţilor sistemului asigură realizarea unui anumit obiectiv respectiv al unui complex de obiective.
4. Obiectivul propus se poate realiza prin diverse procedee dar care au la bază aceeaşi structură a comunicaţiilor între părţile componente ale sistemului, respectiv diversele procedee sunt izomorfe.

Din punct de vedere practic, al ingineriei sistemelor, sistemele din natură dar în special cele tehnice prezintă utilitate dacă posedă următoarele proprietăţi principale:

1. *Stabilitatea* este o proprietate esenţială pentru evoluţia normală sau chiar pentru existenţa marii majorităţi a sistemelor. Un sistem este stabil dacă fiind dat un anumit regim de echilibru al mărimilor unui sistem, orice perturbare este urmată de revenirea naturală a sistemului, în timp, la regimul de echilibru care a precedat schimbarea.
2. *Controlabilitatea* este proprietatea conform căreia pentru orice evoluţie dorită a mărimilor-efecte există o evoluţie a mărimilor-cauze sub acţiunea cărora sistemul realizează respectiva evoluţie a mărimilor efecte.
3. *Observabilitatea* este proprietatea conform căreia dacă pentru o anumită evoluţie cunoscută a mărimilor-efecte, realizată de sistem sub acţiunea unor mărimi-cauze necunoscute, este posibilă determinarea evoluţiei respectivelor mărimi-cauze.
4. *Identificabilitatea* este proprietatea care exprimă faptul că pe baza cunoaşterii evoluţiei mărimilor-cauze şi a evoluţiei corespunzătoare a mărimilor-efecte se pot determina structura şi parametrii sistemului.
5. *Adaptabilitatea* este o proprietate legată de aprecierea calităţilor sistemelor, naturale sau tehnice. Acestea se apreciază fie pe baza proprietăţilor de mai sus, fie, mai ales, pe baza unor indicatori simpli sau sintetici care caracterizează relaţia

dintre mărimile-cauze şi mărimile-efecte. Dacă un sistem nu posedă o anumită calitate, dar prin modidificări structurale sau ajustări parametrice adecvate noul sistem evidenţiază calitatea respectivă, atunci sistemul iniţial are proprietatea de *adaptabilitate* (eventual autoadaptare).

6. *Robusteţea* caracterizează proprietatea unui sistem de a-şi conserva, între anumite limite precizate sau precizabile, o anumită calitate bine definită, în condiţiile în care parametrii şi/sau structura sistemului se modifică (în mod cunoscut sau nu) între anumite limite admisibile.

2.1.3 Modelarea sistemelor tehnice

Cunoaşterea sistemelor reale (naturale sau tehnice) se bazează pe construcţia - prin sistematizarea observaţiilor, sintetizarea rezultatelor măsurărilor şi cunoaşterea legilor generale ale naturii - a unei imagini, de regulă idealizate şi esenţializate, a fenomenelor reale. Această imagine a realităţii reprezintă ea însăsi un complex unitar, caracterizat printr-o structura internă, respectiv un *sistem abstract*. Descrierea sistemului abstract se face cu ajutorul unui model (verbal, grafic, informaţional, matematic) pe baza căreia se pot explicita proprietăţile cunoscute ale sistemului real şi predicta altele noi, neevidenţiate de observaţii şi măsurări, putându-se concepe experimente pentru punerea în evidenţă a respectivelor proprietăţi noi.

Validarea sistemului abstract (al modelului) este o etapă esenţială în procesul dinamic-iterativ al cunoaşterii sistemelor reale. Aceasta constă în regăsirea în realitate, prin experimente adecvate, a acelor proprietăţi care au fost evidenţiate de teoria care fundamentează sistemul abstract. Ca urmare a procesului de validare un sistem abstract poate fi acceptat sau respins.

Modelul se elaborează în funcţie de aspectele ce se doresc a fi surprinse. Oricare ar fi acestea cele patru axiome de mai sus rămân valabile. Spre exemplu, în cazul modelării discrete, orientate pe evenimente se iau în considerare doar valori caracteristice ale mărimilor de stare şi nu toate valorile pe care le iau în timpul variaţiei.

În cazul unui proces industrial diagramele (scheme bloc, diagrame procedurale, scheme de principiu) care descriu sistemul de automatizare reprezintă modele ale sistemului. Acestea se întocmesc conform unor normative specifice şi sunt mijloace de comunicaţie dintre personalul de proiectare, execuţie, exploatare şi întreţinere a sistemului de automatizare.

Modelele matematice sub forma unor sisteme de ecuaţii algebrice şi diferenţiale descriu partea specifică fizico-chimică şi/sau aparativă a proceselor. Ele sunt utilizate în procedeele de măsurare bazate pe modele, în reglarea proceselor, în diagnoză şi în procedeele de simulare.

Documentaţia unei instalaţii sau a unui agregat este o descriere fidelă a acestuia şi este un instrument de comunicaţie preferat al personalului de montaj, exploatare şi întreţinere. Crearea unor astfel de modele şi documentaţii este mult facilitată de sistemele CAD. Acestea automatizează elementele de rutină dar se bazează pe aceleaşi principii ca în cazul elaborării manuale.

Pe lângă matematică, în zilele noastre, şi informatica oferă noi principii de structurare cu ajutorul cărora sistemele de automatizare pot fi modelate, simulate, analizate respectiv optimizate. Aceste tehnici, numite şi ***modele de date informaţionale*** sau ***modele de date semantice***, sunt utilizate la realizarea băncilor de date relaţionale - numite în acest caz şi Entity-Relationship Modell, ER-Modell. Într-o formă adaptată aceste tehnici pot fi utilizate şi la modelarea şi realizarea sistemelor prin structurarea orientată pe obiecte.

Scopul principal al structurării informaţionale este de a stăpâni complexitatea sistemelor printr-o modelare adecvată. Modelele informaţionale sunt componente indispensabile ale unei tehnologii informaţionale care priveşte informaţia ca un factor de producţie decisiv.

Indiferent de forma concretă de implementare se conturează tot mai clar faptul că, datorită metodelor de prezentare grafică, modelele de date semantice vor constitui în viitor mijlocul de comunicaţie dintre constructorii de sistem şi beneficiarii respectiv operatorii sistemelor de automatizare.

În descrierea unui sistem dat, puncte de vedere diferite, adică criterii diferite, necesită modele diferite cu o prezentare diferită. Din acest motiv se impune a se lucra pe baza mai multor principii de modelare.

În continuare vor fi prezentate următoarele principii de structurare, utilizate în modelarea sistemelor de automatizare:

1. **principiul decompoziţiei** respectiv modelarea orientată pe obiecte, prin care sistemele şi subsistemele se divizeaza în părţi componente;
2. **principiul abstractizării funcţionale**, prin care se realizează structurarea pe niveluri ierarhice a sistemelor;
3. **principiul transformării** sau modelarea orientată pe evenimente, prin care se descriu transformările de stare.

2.2 Principii de structurare în modelarea sistemelor de automatizare

Ingineria sistemelor industriale este interesată în găsirea şi aplicarea acelor principii şi metode de structurare care permit descrierea şi modelarea sistemelor industriale astfel încât acestea să fie realizabile conform specificaţiei de definire. Fiecare principiu de structurare şi modelare descrie părţile componenete ale sistemelor precum şi relaţiile dintre ele şi cu mediul.

2.2.1 Modelarea conceptuală

Prin modelare conceptuală se înţelege descrierea formală a tuturor aspectelor relevante ale sistemului ce se realizează făcând abstracţie de orice aspecte legate de implementare. Se vor avea în vedere atât aspecte statice cât şi dinamice. Modelarea conceptuală permite realizarea de modele informatice pentru diversele componente ale sistemelor de automatizare, [Pol'94], [MOS'93]. Aceste modele constituie baza comunicaţiei interdisciplinare în cursul concepţiei, implementării şi exploatării sistemelor de automatizare.

2.2.1.1 Noţiuni de bază şi terminologia utilizată în modelarea conceptuală

Standardul ISO 82 ([ISO'82]) conţine propuneri de standardizare a terminologiei în domeniul modelării conceptuale. Unele din acestea vor fi reluate mai jos.

Pentru ca modelarea conceptuală să fie realizabilă trebuie acceptată premisa că este posibilă delimitarea unei părţi a lumii reale sau ipotetice şi că aceasta poate fi descrisă printr-un limbaj formal bine definit.

Noţiunea de bază în cadrul modelării conceptuale este entitatea.

Definiţia 2-1 (Entitate)

> Toate lucrurile interesante, concrete sau abstracte, din segmentul de realitate modelat care pot fi observate sau a căror existenţă este presupusă vor fi numite ***entităţi***.

În cursul modelării conceptuale, de cele mai multe ori, se lucrează cu obiecte. Noţiunea de obiect se defineşe după cum urmează.

Definiţia 2-2 (Obiect)

> Acele entităţi care au o existenţă de sine stătătoare, nelegată de existenţa vreunei alte entităţi a segmentului de realitate modelat, se vor numi ***obiecte***.

Obiectele se manifestă prin interacţiuni cu mediul şi/sau cu obiectele cu care se găsesc într-o anumită relaţie. Relaţiile dintre obiecte respectiv dintre obiecte şi mediu se definesc după cum urmează:

Definiţia 2-3 (Relaţie)

> ***Relaţiile*** dintre obiecte respectiv dintre obiecte şi mediu sunt entităţi ale segmentului de realitate modelat care specifică posibilităţile de interacţiune dintre obiecte respectiv dintre obiecte şi mediu.

Spre deosebire de obiecte, relaţiile nu au o existenţă de sine stătătoare în cdrul segmentului dc ralitate modelat. O relaţie există doar în măsura în care este posibilă o interacţiune între două sau mai multe obiecte respectiv între obiecte şi mediu.

O entitate dată a segmentului dc realitate modelat este fie un obiect fie o relaţie. Prin urmarea mulţimea obiectelor şi mulţimea relaţiilor sunt mulţimi disjuncte iar reuniunea lor formează mulţimea entităţilor.

Relativ la entităţi de bază, diverse entităţi, grupe de entităţi, etc., pot fi formulate afirmaţii.

Definiţia 2-4 (Afirmaţie, (proposition))

> O afirmaţie exprimă o calitate observabilă ce se referă la entităţi şi pentru care este posibil a se decide dacă este valabilă (dată) sau nu este valabilă (dată) pentru anumite entităţi.

În mod uzual se deosebesc afirmaţii despre starea actuală a entităţilor respectiv despre starea posibilă sau permisă a acestora (reguli, restricţii). În vederea schimbului de informaţii despre entităţi se vor utiliza descrieri ale afirmaţiilor.

Definiţia 2-5 (Descrieri ale afirmaţiilor)

> (a) Prin *obiect lingvistic* se înţelege o construcţie, permisă din punct de vedere gramatical, al unui limbaj.
>
> (b) Un obiect lingvistic utilizat pentru referirea unei entităţi se numeşte *termen*.
>
> (c) Un *predicat* este un obiect lingvistic care exprimă ceva (o proprietate, un atribut, o calitate) despre o entitate sau despre mai multe entităţi ce sunt referite prin termeni.
>
> (d) O *propoziţie* (sentence) este un obiect lingvistic care descrie o afirmaţie. Propoziţiile sunt alcătuite din termeni şi predicate.

În cadrul modelării conceptuale trebuie acordată o atenţie deosebită legăturilor dintre entităţi şi termeni. Se poate întâmpla ca pentru referirea unei entităţi să fie utilizaţi mai mulţi termeni diferiţi respectiv ca un anumit termen să fie utilizat pentru referirea mai multor entităţi. Pentru a desemna aceste situaţii se definesc următoarele.

Definiţia 2-6 (Sinonime, Omonime)

> (a) Termenii diferiţi utilizaţi pentru referirea unei aceleaşi entităţi se numesc *sinonime*.
>
> (b) Termenii identici utilizaţi pentru referirea de entităţi diverse se numesc *omonime*.

Deseori sunt grupate mulţimi de entităţi şi afirmaţii pentru a forma domenii. Domeniile caracterizează structura implicită sau indusă, prin acţiunea modelatorului, a mulţimii entităţilor. Domeniile sunt delimitate prin afirmaţiile ce sunt valabile pentru toate elementele unui domeniu de entităţi.

Definiţia 2-7 (Domenii)

(a) Se numeşte *domeniu de entităţi* (Entity Word) o submulţime a entităţilor ce formează un element structural într-o structură implicită sau indusă peste mulţimea entităţilor

(b) Se numeşte *domeniu de afirmaţii.* (Proposition Word) mulţimea afirmaţiilor valabile pentru elementele unui domeniu de entităţi

În cadrul modelării conceptuale raţionamentele se referă la toate entităţile concrete sau abstracte din segmentul lumii reale sau ipotetice luate în considerare. Pentru a exprima acest lucru s-a creat noţiunea, din ce în ce mai des utilizată, de "univers al discursului", conform următoarei definiţii.

Definiţia 2-8 (Universul Discursului (Universe of Discourse))

Prin *univers al discursului* se înţelege mulţimea tuturor entităţilor interesante ale segmentului de realitate luat în considerare. Ele includ toate entităţile existente şi acelea care vor exista sau ar putea să existe cândva.

Pentru a putea defini schema conceptuală ca rezultat al modelării conceptuale sunt necesare următoarele concepte.

Definiţia 2-9 (Afirmaţii neceare (Necessery Proposition))

Se numesc *afirmaţii necesare* afirmaţiile care trebuie să fie valabile pentru toate domeniile de entităţi, adică acele afirmaţii care trebuie să fie elemente ale tuturor domeniilor de afirmaţii.

Definiţia 2-10 (Clasă, tip, instanţiere)

(a) *O clasă* (Class) reprezintă o submulţime a mulţimii entităţilor universului discursului pentru care este valabilă o anumită afirmaţie. O anumită entitate poate să aparţină mai multor clase. Din această cauză, în general, clasele nu sunt disjuncte.

(b) Se numeşte *tipul* unei entităţi, afirmaţia care exprimă faptul că o anumită entitate este membră a unei clase date. O astfel de afirmaţie trebuie să se refere la o clasă există.

(c) Se numeşte *instanţiere* (Instance) a unei clasei, o entitate concretă sau concret referită pentru care este valabilă o afirmaţie specifică unui anumit tip.

În principiu o clasă poate avea oricâte instanţieri. Tipurile vor fi referite prin nume. Acestea pot să fie şi în acest caz omonime sau sinonime.

Pe baza noţiunilor definite mai sus pot fi definite noţiunile de schemă conceptuală şi modelare conceptuală.

Definiţia 2-11 (Schemă conceptuală, modelare conceptuală)

(a) *Schema conceptuală* este un ansamblu de propoziţii formale şi consistente care descriu afirmaţiile necesare ale universului discursului.

(b) Prin *modelare conceptuală* sau proiectare conceptuală se înţelege procesul de realizare a schemei conceptuale.

În cadrul modelării conceptuale se vor respecta următoarele două principii:

(1) Principiul completitudinii (100 Procent Principle)

În cadrul schemei conceptuale trebuie descrise toate aspectele relevante, din punct de vedere static sau dinamic, ale universului discursului. În funcţie de limbajul formal utilizat aspectele relevante se vor descrie într-o formă procedurală sau declarativă.

(2) Principiul conceptualizării (Conceptualization Principle)

O schemă conceptuală trebuie să conţină doar aspecte relevante din punct de vedere conceptual ale universului discursului. Se va face abstracţie de orice aspecte ale posibilelor implementării viitoare: aspecte legate de reprezentarea datelor prezentate utilizatorului, aspecte legate de eficienţa calculelor, aspecte legate de organizarea fizică a datelor precum şi aspecte organizatorice legate de sistemul ce se va realiza.

2.2.2 Principiul decompoziţiei, structurarea orientată pe obiecte

2.2.2.1 *Modelul semantic orientat pe obiecte*

În această secţiune se descrie un model informaţional, numit modelul semantic orientat pe obiecte (MSOO) prin care se realizează modelarea conceptuală.

A) Abstractizare

Abstractizarea este conceptul de bază în realizarea modelului informaţional. Printr-o abstractizare a unui sistem se înţelege un model al acestuia în care anumite detalii se neglijează în mod conştient. Modelele bazate pe abstractizare şi clasificare ierarhică sunt deosebit de utile în tratarea complexităţii, diversităţii şi evoluţiei atât în ştiinţă cât şi în viaţa de toate zilele. În lipsa unui efort de abstractizare şi de constituire a unei clasificări în categorii de obiecte, plecând de la aspecte esenţiale şi adăugând succesiv calităţi suplimentare, specifice, pentru a crea categorii noi, confuzia este inevitabilă.

Adeseori, în practică, se întâlnesc sisteme cu mult prea multe detalii relevante pentru a fi surprinse în mod eficient într-un singur model. Procedeul de modelare informaţională ce se prezintă mai jos permite divizarea unui sistem pe niveluri ierarhice de abstractizare. În acest fel modelul poate fi divizat pe diferite niveluri de abstractizare. Prin aceasta se poate asigura şi o bună stabilitate a modelului deoarece de cele mai multe ori modificări de detaliu într-un anumit nivel nu influenţează structura nivelurilor superioare.

Elementele de bază ale modelării informaţionale sunt obiecte concrete sau abstracte. Acestea pot să fie obiecte momentan existente sau obiecte ce ar putea exista cândva şi care trebuie luate în considerare în cadrul modelării.

În cazul MSOO atributele, obiecte ce pot exista doar prin legătura cu alte obiecte, sunt tratate ca obiecte de sine stătătoare, spre deosebire de modelul Entity-Relationship, unde atributele sunt tratate separat.

MSOO combină concepte relaţionale cu patru relaţii de bază ale reţelelor semantice.

B) Relaţiile de bază în reţelele semantice

1) Clasificare. Relaţia "...instanţiere a ..."

Clasificarea este mecanismul de abstractizare de bază în modelarea informaţională. Clasele se constituie pe baza afirmaţiilor valabile pentru toate obiectele clasei respective. Obiectele dintr-o clasă dată pot fi univoc identificate.

În schemele conceptuale, clasele de obiecte se reprezintă grafic printr-un cerc etichetat cu denumirea univocă a clasei, conform Figurii 2-2. Clasele vor fi întotdeauna denumite prin substantive.

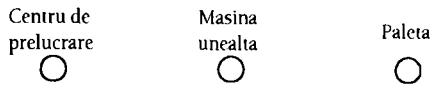


Fig. 2-2 Clasificare. Reprezentarea grafică a claselor.

În cadrul clasificării se folosec deseori noţiunile de *tip* şi *instanţiere*.

Afirmaţia "obiectul concret A este de tipul X" este echivalentă cu afirmaţia "obiectul concret A este membru al clasei X". De multe ori, pentru simplificarea exprimării, noţiunile "clasă" şi "tip" se vor utiliza ca sinonime. În sensul celor de mai sus obiectul A este o instanţiere a clasei X respectiv al tipului X.

2) Generalizare. Relaţia "...este un/o..."

Generalizarea este un mecanism de abstractizare utilizat în structurarea la nivelul claselor. Analiza realizată pe baza acestei relaţii sprijină în mod eficient reutilizarea claselor în diferitele scheme conceptuale. Acele afirmaţii care sunt valabile pentru mai multe clase de obiecte, definesc un ***supertip*** generic peste mulţimea claselor de obiecte. Clasele componente ale unui supertip generic se numesc ***subtipuri***. Afirmaţiile care definesc un supertip generic exprimă calităţi sau proprietăţi comune subtipurilor componente. Într-o exprimare complementară se va vorbi despre "moştenirea", de către subtipuri, a proprietăţilor supertipului. Relaţia de generalizare este o relaţie dintre clase de obiecte.

În schemele conceptuale, funcţia de generalizarea se reprezintă grafic printr-un pătrat în care s-a înscris simbolul disjuncţiei " ∨ ". Acest simbol se leagă de subtipurile generalizate şi de supertip prin arce orientate conform Figurii 2-3. Orientarea arcelor redă direcţia moştenirii proprietăţilor (property inheritance) pentru care sunt valabile afirmaţiile pe baza cărora se execută generalizarea.

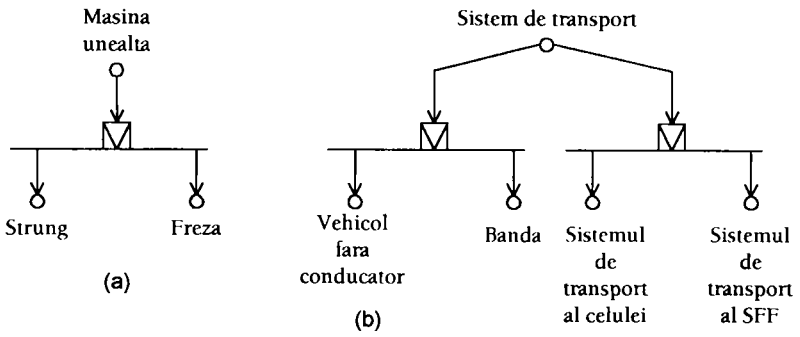


Fig. 2-3 Generalizare

Figura 2-3 prezintă două exemple pentru relaţia de generalizare. În Figura 2-3(a) afirmaţiile "freza este o maşină unealtă" respectiv "strungul este o maşină unealtă" definesc supertipul generic "Maşină unealtă" având ca şi subtipuri "Freza" şi "Strungul". Simbolul disjuncţiei din reprezantarea grafică a funcţiei de generalizare are semnificaţia de "sau inclusiv". Cu alte cuvinte, în exemplul din Figura 2-3, o maşină unealtă poate să fie o freză sau un strung.

În Figura 2-3(b) "Sistem transport" este un supertip pentru două relaţii de generalizare.

Observaţie: Nu ar fi corect ca subtipurile să participe la o singură relaţie de generalizare, deoarece s-ar pierde informaţia referitoare la afirmaţiile pe baza cărora s-a executat generalizarea.

3) Agregare. Relaţia "...este o parte a..."

Mecanismul de agregare permite abordarea relaţiilor dintre obiecte ca obiecte abstracte ale unui nivel ierarhic superior. Tipul unui astfel de obiect abstract se va numi *tip agregat* iar tipurile care participă la agregare se vor numi *tipuri componente*.

În schemele conceptuale funcţia de agregarea se reprezintă grafic printr-un pătrat în care s-a înscris semnul "×", simbolul operaţiei de conjuncţie logică. Acest simbol este legat, prin arce orientate, de tipurile componente şi de tipul agregat, aşa cum se prezintă în Figura 2-4. Orientarea arcelor corespunde în acest caz atât direcţiei moştenirii cât şi al ordonării ierarhice. În cazul relaţiei de agragare, tipul agragat moşteneşte proprietăţile tipurilor componente.

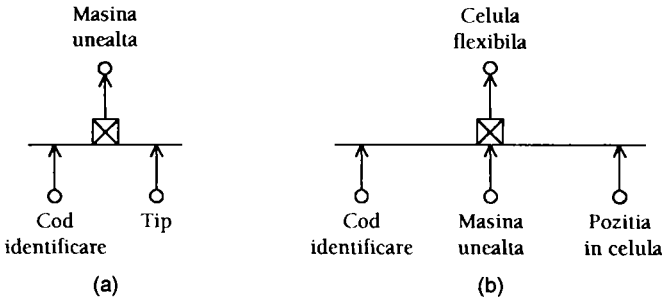


Fig. 2-4 Agregare

Figura 2-4(a) prezintă tipul agregat "Maşină unealtă" cu tipurile sale componente "cod identificare" şi "tip". În acest caz tipurile componente sunt tipuri elementare adică atribute.

Figura 2-4(b) prezintă o agregare în care sunt cuprinse atât tipurile componente elementare "cod identificare" şi "poziţii în celulă", cât şi un tip component care la rândul lui este un tip agregat, "Maşina unealtă", ca parte a celulei flexibile. La nivel conceptual "codul de identificare" şi "poziţia în celulă" sunt componentele tipului agregat "Celula Flexibilă".

4) Asociere. Relaţia "...este membru a..."

Asocierea permite modelarea tipurilor de obiecte a căror instanţe sunt mulţimi de instanţe ale tipurilor de obiecte de la nivelul ierarhic inferior. Tipul de obiect creat prin asociere se va numi *tip mulţime* iar tipul ierarhic inferior se va numi *tip element*.

În cadrul schemei conceptuale, funcţia de asociere se reprezintă grafic printr-un pătrat în care s-a înscris simbolul "✳". Simbolul rezultat se leagă, prin arce orientate, de tipurile element. Aceste arce specifică atât direcţia moştenirii cât şi ordonarea ierarhică a tipurilor de obiecte.

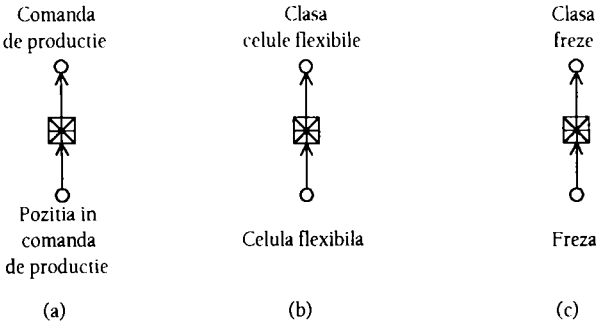


Fig. 2-5 Asociere

Figura 2-5(a) prezintă o asociere cu tipul mulţime "comandă de producţie" şi tipul element "poziţie în comanda de producţie".

Figura 2-5(b) prezintă o asociere de tipul mulţime "(Clasa) Celule Flexibile" având ca tip element "celula flexibilă" iar Figura 2-5(c) prezintă o asociere de tipul mulţime "(Clasa) Freze" având ca tip element "freza".

C) Interpretarea top-down respectiv bottom-up

În cele de mai sus mecanismele de abstractizare au fost prezentate într-o interpretare bottom-up. În mod special pentru necesităţile activităţilor de proiectare este deosebit de utilă o interpretare top-down. Corespondenţa dintre relaţiile celor două interpretări se prezintă în tabelul următor:

Bottom-up	Top-down
Clasificare	Instanţiere
Generalizare	Specializare
Agregare	Descompunere
Asociere	Dezmembrare

În cursul MSOO, realizată prin procedeul "top-down", se vor utiliza aceleaşi simboluri, se schimbă doar punctul de vedere al modelării. În unele lucrări, (ex. [Sub'96]), aceste funcţiuni sunt denumite chiar prin expresii simetrice: Gen-Spec pentru Generalizare-Specializare respectiv Whole-Part pentru Agregare-Descompunere, subliniind chiar prin denumiri reversibilitate acestor funcţiuni.

2.2.2.2 *MSOO şi programarea orientată pe obiecte*

Programarea orientată pe obiecte, abreviat POO, sprijină în mod direct modelarea orientată pe obiecte respectiv implementarea modelelor conceptuale orientate pe obiecte. POO este o tehnică de programare care abordează realitatea în concordanţă cu principiile MSOO enunţate mai sus şi oferă modele apropiate de obiectele reale şi de modul de reprezentare familiar programatorului.

Un obiect real este caracterizat atât prin structură cât şi prin funcţionalitate. POO se bazează pe conceptul de *obiect*, care reprezintă ansamblul alcătuit dintr-o structură de date şi procedurile (metodele) de operare cu aceste date. Aceste metode, care pot fi funcţii sau procese (în sensul programării structurate), determină caracteristicile comportamentale ale obiectelor. Se utilizează frecvent denumirea generică de membri

pentru date şi metode: *date membre* şi *funcţii membre*. Abordarea POO se bazează deci pe relaţia:

Date + Metode = Obiect.

În practică, de foarte multe ori este necesară ameliorarea structurală şi funcţională a obiectelor în prealabil definite. Este de dorit ca această ameliorare să fie transparentă pentru utilizatorul obiectului. Acest lucru poate fi asigurat prin principiul general al *încapsulării datelor*, confor căruia accesul la date se poate face numai prin intermediul setului de metode asociat. Acest principiu determină o abstractizare a datelor în sensul că un obiect este caracterizat complet de specificaţiile metodelor sale, detaliile de implementare fiind transparente pentru utilizator. Modificarea structurii datelor membre are ca efect numai modificări în setul de metode asociate şi, în măsura în care se păstrează specificaţiile metodelor, nu afectează utilizarea obiectului.

În cadrul MSOO respectiv POO, accentul este pus pe identificarea obiectelor şi pe modelarea acestor obiecte cu comportarea lor şi cu relaţiile dintre ele. Scopul modelării se deplasează astfel de la modelarea comportării segmentului de realitate la modelarea obiectelor care există în acest segment şi a comportării lor individuale. Caracteristica esenţială a limbajelor de programare orientate pe obiecte este faptul că permit construcţii care dau posibilitea distincţiei între proprietăţile generale şi proprietăţile specifice.

Conceptul fundamental este *clasa*. În POO clasa este o generalizare a noţiunii de tip de date. O clasă descrie un ansamblu de obiecte similare având aceeaşi structură a datelor şi aceleşi metode. În cadrul POO, un obiect este deci o variabilă de un anumit tip clasă. Clasele asigură încapsularea datelor, garantează iniţializarea datelor, conversii implicite de tip pentru tipuri definite de utilizator, tipizare dinamică, gestiunea memoriei controlată de utilizator şi mecanisme pentru supraîncărcarea operatorilor.

Evoluţia şi ierarhizarea claselor de obiecte sunt modelate pe baza conceptului de *moştenire*. Procedeul numit *derivare* permite definirea unei clase noi - clasa derivată - , pornind de la o clasă existentă - clasa de bază - prin adăugarea de date şi metode. Clasa derivată preia structura de date şi metodele clasei de bază (deci moşteneşte proprietăţile acesteia) şi adăugă altele suplimentare şi prin aceasta devine suportul unor funcţiuni noi.

Pornind de la un set de clase existent se pot realiza clase cu proprietăţi adecvate unei aplicaţii noi. Procesul de specializare a claselor se poate efectua în mai multe etape, rezultând o ierarhie de clase derivate. Pe de altă parte este posibilă derivarea unei clase din mai multe clase de bază prin procedeul de *moştenire multiplă*.

Moştenirea creează implicit o ierarhie de clase cu compatibilitatea unidirecţională "de sus în jos". În privinţa accesului la date sau metode încapsularea este o caracteristică mai "puternică" decât moştenirea în sensul că, în cazul accesului la date sau metode, încapsularea este prioritară moştenirii.

O caracteristică importantă a claselor de obiecte este cea de *polimorfism*. Conceptul de polimorfism se referă, în sens larg, la posibilitatea de a opera cu diverse variante ale unei funcţii care efectuează o anumită operaţie în mod specific pentru diferite obiecte. POO permite asocierea unei anumite activităţi, definirea variantelor şi selecţia automată a variantei corespunzătoare obiectului. În multe cazuri este util ca într-o clasă să se definească funcţii cu caracter general care prin derivare să poată fi înlocuite cu o variantă a acestora. Obiectele astfel create se numesc *obiecte polimorfice*.

Prin procedeele de moştenire şi de creare de obiecte polimorfice se poate asigura ca realizările noi să se bazeze pe eforturi anterioare respectiv soluţii deja existente pot fi preluate şi valorificate pentru o aplicaţie nouă, fapt deosebit de important din punctul de vedere al ingineriei sistemelor.

Pentru ca în cazul unei aplicaţii concrete obiectele declarate să devină funcţionale trebuie să fie iniţializate printr-un procedeu denumit ***instanţiere*** (creare de instanţe). Aceasta se realizează cu ajutorul unei metode speciale denumită *constructor*. Metoda complementară se numeşte *destructor* şi are rolul de a elibera resursele ocupate de o anumită instanţă la terminarea misiunii acesteia.

În cursul modelării, reţelele de clase ierarhice pot fi create în ambele sensuri. Prin "specializare" se creează obiecte dintr-o clasă de bază iar prin "generalizare", exprimând ceea ce este comun în atributele şi metodele unei grupe de obiecte modelate, se creează o clasă de bază nouă.

2.2.2.3 *MSOO în cadrul modelării sistemelor de automatizare*

În modelarera orientată pe obiecte a sistemelor de automatizare se folosesc conceptele MSOO. Din cauza faptului că implementarea acestor modele se realizează prin POO se va utiliza terminologia specifică acesteia dacă nu există pericolul confuziei. În cazul unui sistem de automatizare tehnica de modelare orientată pe obiecte înseamnă :

1. Modelarea este axată pe noţiunea de obiect. Un obiect poate fi o entitate naturală sau abstractă (de exemplu chiar şi o anumită funcţiune).
2. Obiectele sunt descrise prin proprietăţile lor, numite şi atribute (de exemplu proprietăţile unui produs dar şi ale unui proces de producţie). Atributele pot fi scalate (măsurate) atât metric cât şi topologic.
3. Prin procedeul de moştenire se creează *reţele de clase*, cu alte cuvinte se stabilesc relaţii de clasificare. În procesul sintezei sistemelor de automatizare se stabilesc şi *relaţii de agregare*. Prin agregare se descriu structuri de obiecte formate din obiecte de diverse clase. Structura astfel creată defineşte un nou obiect care poate fi integrat într-o reţea de clase.
4. Modelele obiectelor vor include şi metode sau procedee de prelucrare materială sau informaţională, de exemplu metode de calcul care descriu reacţia obiectului în cadrul interacţiunilor cu celelalte obiecte.
5. Prin clasificare şi agregare se creeaza reţele de obiecte - reţele de agregare respectiv reţele de clase. Reţelele de agregare se utilizează pentru descrierea constructivă utilă în cazul proiectării, exploatării şi întreţinerii sistemelor tehnice. O specificaţie de aparate şi echipamente este un exemplu de agregare.
6. O relaţie stabileşte o legătura între două sau mai multe obiecte. Spre exemplu, o dispoziţie de producţie defineşte relaţiile dintre agregate, termene, materiale şi reţete de realizare. Relaţiile se definesc prin clasificare, generalizare, agregare şi asociere.
7. În cazul sistemelor de automatizare instanţierea este un proces de parametrizare şi iniţializare.

Respectând definiţiile de mai sus, pentru obiect şi relaţie, un modelator le poate utiliza fără alte restricţii. Din acest motiv, în funcţie de punctul de vedere şi scopul urmărit, diverşi modelatori vor ajunge la modele diferite pentru acelaşi segment de realitate.

Modelele astfel realizate pot fi prezentate sub forma grafică şi sunt astfel uşor inteligibile şi acceptabile.

2.2.2.4 *Clase de obiecte ale sistemelor de automatizare*

Acceptând noţiunea de obiect în sensul celor de mai sus, pentru tratarea cu calculatorul a problemelor proiectării şi exploatării unui sistem de producţie (dar şi a unui aparat, a

unui agregat, a unui produs software) este necesară specificarea obiectelor adică delimitarea diverselor obiecte şi clase de obiecte.

În cazul unui sistem de producţie se disting următoarele clase de obiecte: echipamente de producţie, tehnologii de fabricaţie, echipamente de conducere, comunicaţie proces, logistică, întreţinere. Aceste clase de obiecte trebuie strict delimitate iar interacţiunile dintre ele complet specificate.

În continuare, pentru elaborarea modelelor informaţionale ale diverselor componente ale sistemelor de automatizare se va folosi MSOO. Această tehnică de modelare prezintă următoarele avantaje:

1. Permite o descriere constructivă completă şi precisă, a unui sistem tehnic, fără text (care adesea lasă loc ambiguităţilor) într-o formă de organizare la îndemâna utilizatorului.
2. Relaţiile dintre obiecte sunt, din punct de vedere logic, consistent definite adică atemporal valabile cu condiţia invariabilităţii definiţiilor de obiecte.
3. Permite identificarea sigură şi univocă a diverselor obiecte, de exemplu, în vederea intreţinerii sau documentării.

Figura 2-6 prezintă acest procedeu în cazul unei instalaţii de amestec cu agitator. Aceasta se compune din: acţionare, transmisii, reductor etc. Fiecare componentă se descrie prin atribute.

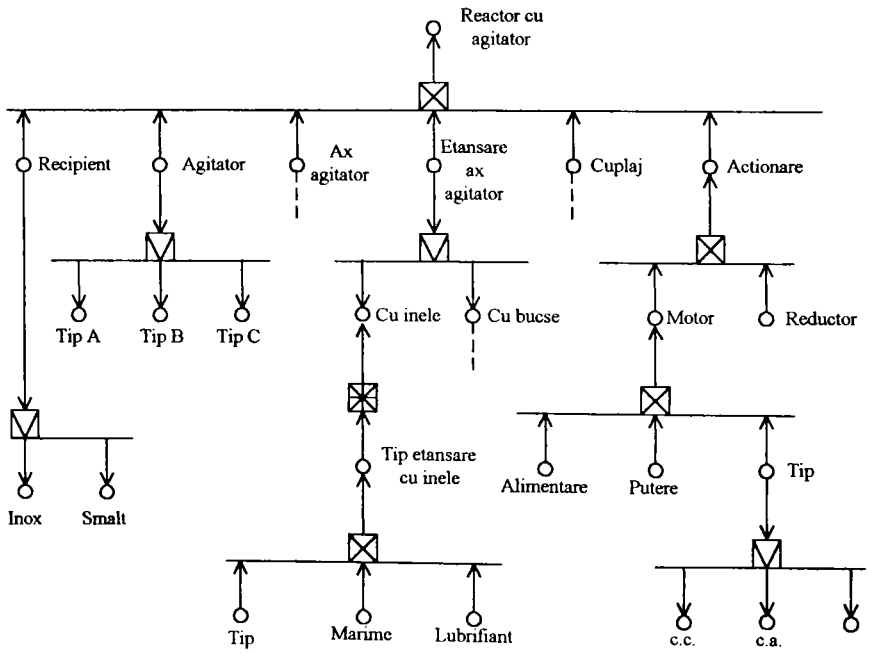


Fig. 2-6 Modelul informaţional al unui reactor cu agitator

Componentele instalaţiei de amestec cu agitator sunt prezentate în doua relaţii, cea de agregare (relaţia "... este o parte a ...") şi cea de generalizare (relaţia "... este un/o ...").

Aceasta din urmă se utilizează în situaţia în care componenta/subansamblul poate să fie de diferite tipuri.

Figura 2-7. prezintă un exemplu de reţea de clase a operaţiilor din tehnica conducerii proceselor.

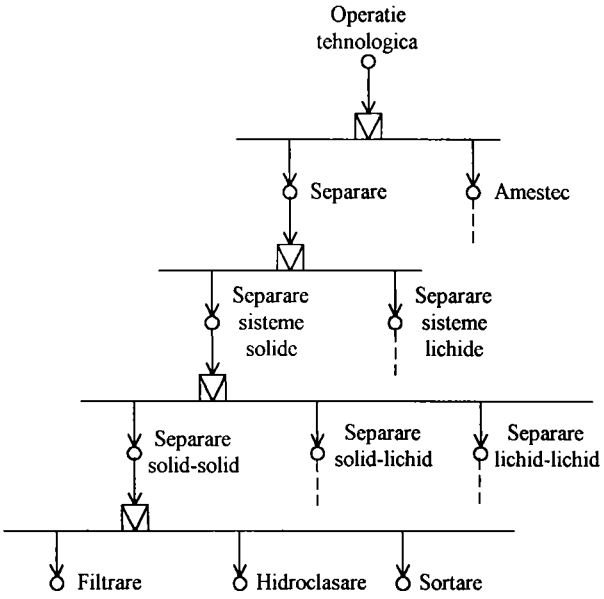


Fig. 2-7 Reţea de clase de operaţii tehnologice elementare

Având la bază mulţimea de operaţii tehnologice, definite conform principiilor de mai sus, se elaborează "o fişă tehnologică" sau reţetă de producţie pe baza căreia se realizează un anumit produs. Figura 2-7 prezintă o parte din structura unei astfel de reţete.

Pe baza acestei structuri, conducerea procesului de producţie trebuie să rezolve problema transpunerii, pe structura sistemului de producţie şi a sistemului de conducere, a prescripţiilor tehnologice cuprinse în reţetă de producţie, adică să asigure producerea unui anumit produs pe o anumită instalaţie, într-un anumit timp la cantitatea şi calitatea solicitată. Aceasta se poate realiza secvenţial pe o singură instalaţie (BATCH), sau pe mai multe instalaţii (continuu) în paralel pe baza unei "reţete de comandă".

O reţetă de producţie de comandă conţine toate informaţiile necesare şi suficiente care asigură reproductibilitatea unui produs dat şi se realizează pe baza unei ierarhii de clase ca în Figura 2-8.

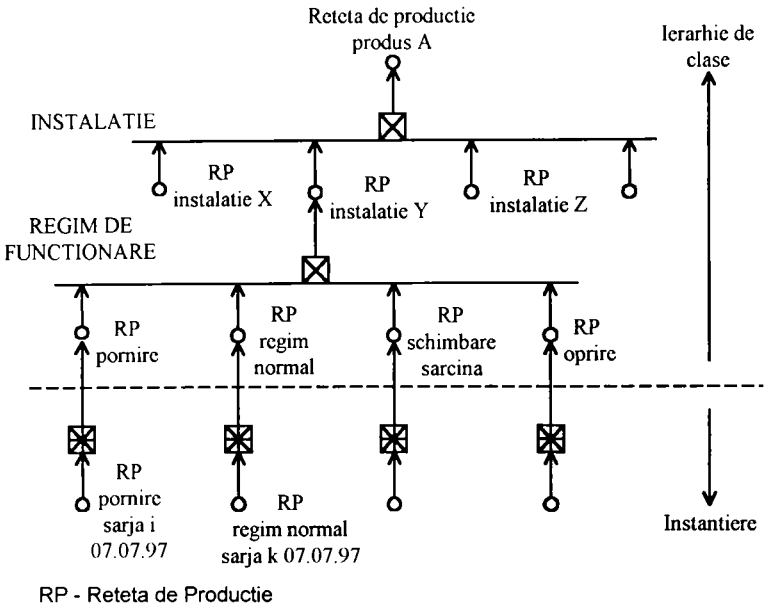


Fig. 2-8 Ierarhia de clase a unei reţete de producţie

Clasa superioară descrie, la modul general, realizarea unui produs. Informaţia conţinută în clasa superioară se consideră a fi reţeptura de bază. În clasa imediat următoare se concretizează această reţetă de producţie de bază prin proiecţia pe diferitele instalaţii (să se producă un produs A, cu calităţile B, prin procedeul C, pe instalaţia Y). Informaţia necesară este cuprinsă în aşa numita "reteptură de comandă a instalaţiei". Un nou pas spre concretizare poate însemna specificarea regimurior de funcţionare.

În Figura 2-8 se prezintă şi diferenţa dintre clase şi instanţiere. Clasele sunt descrieri valabile alr unei mulţimi de obiecte înrudite, instanţierile sunt copii ale descrierii clasei valabile la un moment dat sau cu un anumit parametru dat (de exemplu "să se producă pe instalaţia Y, produsul A, de calitate B, conform procedeului C, la momentul T"). Ierarhiile de clase (taxonomiile) stabilesc ordinea abstractă dintre clase. Instanţierile, în schimb, sunt copii ale descrierilor de clase, care sunt realizate la începerea ciclului de producţie şi controlează derularea proceselor şi în acelaşi timp preiau valori actuale ale parametrilor din proces.

2.2.3 Principiul abstractizării funcţionale, structurarea pe niveluri ierarhice

Structurarea prin funcţii reprezintă un grad înalt de abstractizare şi de definire al unui nivel stabil al fenomenelor materiale. Descrierea funcţională univocă a unui sistem care satisface obiectivele de proiectare nu impune soluţii la nivel procedeu/structură de realizare a funcţiilor. Una şi aceeaşi funcţie poate fi realizată printr-o diversitate de procedee/structuri. Funcţia apare ca o esenţa invariantă şi, aşa cum biologii arată, acest fenomen este unul dintre cele mai remarcabile ale vieţii însăşi.

Conform scopului şi destinaţiei, un sistem poate fi modelat mai mult sau mai puţin detaliat, mai mult sau mai puţin concret. Conform funcţiei şi responsabilităţii, diversele

persoane care vin în contact cu sistemul realizat, au puncte de vedere diferite asupra aceluiaşi sistem. Modelatorii de sistem realizează modele ale unor segmente ale lumii reale care acoperă unul sau mai multe dintre aceste puncte de vedere.

2.2.3.1 *Model de referinţă pentru structurarea funcţională a sistemelor de conducere*

Aspectele economice în realizarea sistemelor complexe impun proiectantului următoarele:

- Divizarea (funcţională) clară a sistemului pentru a asigura facilitatea implementării şi eficienţa şcolarizării personalului de exploatare;
- Evitarea dezvoltărilor paralele;
- Luarea în considerare a modificărilor în cerinţele impuse chiar şi în cursul proiectării şi exploatării;
- Tratarea unitară a sistemului şi integrarea acestuia în mediul existent pentru a putea aprecia beneficiile şi lămurii deficienţele respectiv tendinţele şi posibilităţile dezvoltării viitoare.
- Asigurarea accesului la serviciile sistemului a tuturor categoriilor de utilizatori.

Pentru a face faţă acestor cerinţe, în [Sch'94] s-a propus un concept de structurare prin care sistemele complexe pot fi divizate într-o manieră inteligibilă şi uşor acceptabilă.

Conceptul s-a elaborat pe baza principiilor de bază ale prelucrării informaţiei în procesele de decizie umană. Se poate astfel asigura, pe de o parte, independenţa faţă de aspectele de implementare, iar pe de altă parte, transpunerea facilă a divizării realizate în componentele de operare ale sistemului. Figura 2-9 prezintă, pe baza lucrărilor lui Rasmussen, structura procesului decizional uman, sub forma unei reţele Petri.

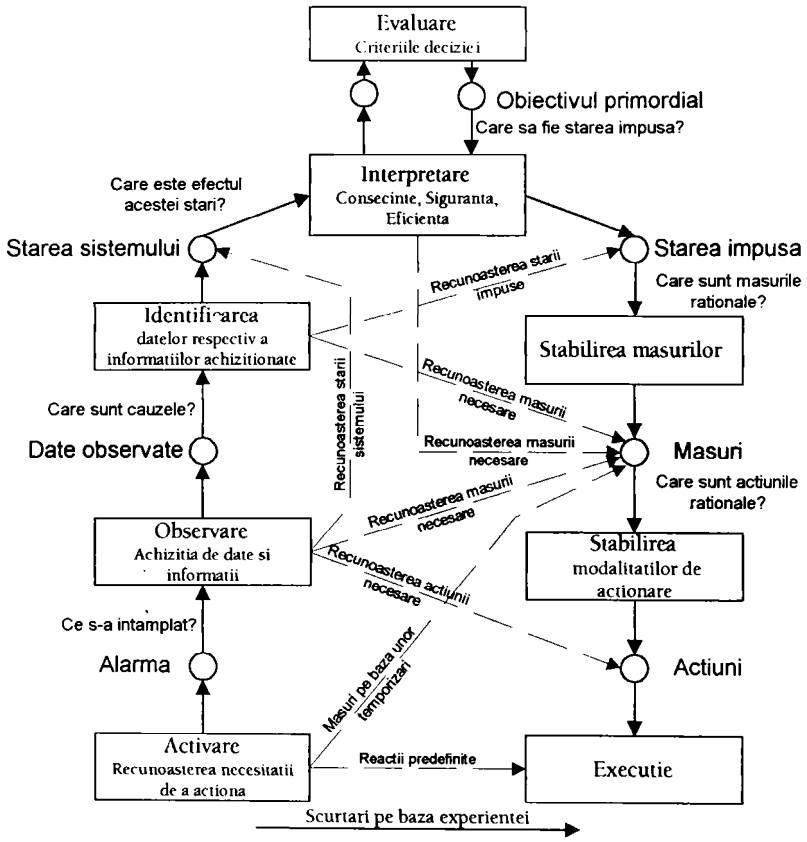


Fig. 2-9 Structura procesului decizional uman după Rasmussen

Figura 2-9 prezintă diversele stadii ce se parcurg din momentul unei solicitări de a reacţiona la un eveniment din sistem până la realizarea efectivă a acesteia.

În cursul analizei bazate pe cunoştinţele despre sistem, informaţiile sunt abstractizate în conformitate cu obiectivele de ansamblu ale sistemului, iar pe urmă, după definirea unei noi stări impuse, în cadrul unei planificări bazate pe cunoştinţele despre sistem, informaţiile se concretizează în elaborarea unor acţiuni realizabile.

Stările din cele două ramuri ale grafului reprezintă "stări de conştientizare standardizate". Aceste stadii intermediare ale procesului de decizie sunt deosebit de utile din mai multe puncte de vedere.

1. Procesele de prelucrare ale informaţiei în diversele faze au calităţi diferite deoarece se bazează pe metode diferite: observare, analiză, evaluare, planificare. Din acest motiv este util şi raţional de a le despărţii prin "stările de conştientizare standardizate". Acestea sunt stări de intrare ale unei faze respectiv stări de ieşire ale fazei precedente.
2. Această divizare permite reprezentarea unor decizii bazate pe experienţa acumulată. Astfel de decizii nu parcurg toate fazele de achiziţie-abstractizare-

planificare-concretizare. Traseele decizionale necesare elaborării unei acţiuni raţionale sunt scurtate. În Figura 2-9 astfel de decizii sunt reprezentate prin arce din linii întrerupte.

3. "Stările de conştientizare standardizate" pot fi văzute ca baze de cunoştinţe ce se îmbogăţesc cu fiecare parcurgere a ciclului decizional. Aceste cunoştinţe stau şi la baza comunicaţiei dintre operatorii unui domeniu dat.

Pe baza reprezentării grafice din Figura 2-9 se pot delimita diverse niveluri calitative cu aceeaşi complexitate a prelucrărilor informaţionale în cursul analizei respectiv planificării bazate pe cunoştinţele despre sistem.

Rezultă astfel, pentru descrierea unui proces decizional complex un model multinivel al evenimentelor caracteristice. Astfel de modele pe niveluri s-au utilizat deja în structurarea proceselor industriale sau a sistemelor de conducere. Exemple tipice sunt arhitecturile de calculatoare şi cele ale sistemelor de comunicaţie (modelul de referinţă ISO-OSI). Din păcate, acestea sunt însă atât de diferite încât comunicaţia dintre ele este de multe ori imposibilă.

2.2.3.2 *Structurarea funcţională pe patru niveluri*

În cele ce urmează se prezintă un concept de structurare pe patru niveluri, bazat pe structurile decizionale umane, care să fie universal aplicabil în descrierea sistemelor tehnice. Aceste patru niveluri sunt după cum urmează:

1. Nivelul strategic

Nivelul strategic conţine funcţiunile strategice, pentru toate domeniile de activitate, necesare conducerii sistemului (management, definire obiective etc). La acest nivel se formulează obiectivele întregului sistem.

Nivelul strategic trebuie să dea răspuns la întrebarea:

"Ce trebuie să fie realizat?"

2. Nivelul dispoziţional

Nivelul dispoziţional conţine funcţiunile logistice, economice şi tehnice necesare, de exemplu, pentru conducerea unei fabrici. Nivelul dispoziţional pune la dispoziţia nivelului inferior resursele necesare. Prin resurse se vor înţege şi informaţiile. Sarcina nivelului dispoziţional este de pune la dispoziţie resursele necesare.

Nivelul dispoziţional trebuie să dea răspuns la întrebarea:

"Care sunt resursele necesare pentru realizarea obiectivelor?"

3. Nivelul tactic

Nivelul tactic conţine pe lângă funcţiunile de conducere a grupelor de producţie şi funcţiuni de bază pentru reglarea respectiv conducerea domeniilor tehnice. Acest nivel coordonează resursele ce stau la dispoziţie.

Nivelul tactic trebuie să dea răspuns la întrebarea:

"Cum utilizez resusele ce-mi stau la dispoziţie?"

4. Nivelul operativ

Acest nivel conţine resursele ce stau la dispoziţie pentru a influenţa procesul condus. La acest nivel are loc execuţia efectivă cu ajutorul resurselor.

Nivelul operativ trebuie să dea răspuns la întrebarea:

"Cum se realizează funcţiunea dată?"

Procesul condus respectiv componentele sale fizice se structurează în aceeaşi manieră. Definiţia nivelurilor de conducere induce şi ordonarea componentelor procesului condus (al obiectivului automatizat), în vederea atingerii obiectivelor propuse. Altfel spus, obiectivele impuse sunt transformate prin sistemul de conducere în stări impuse procesului condus. Prin această structurare sistemul condus influenţează fluxul energetic iar prin aceasta fluxul de materiale şi prin urmare produsul finit. Cele de mai sus sunt sintetizate în Figura 2-10.

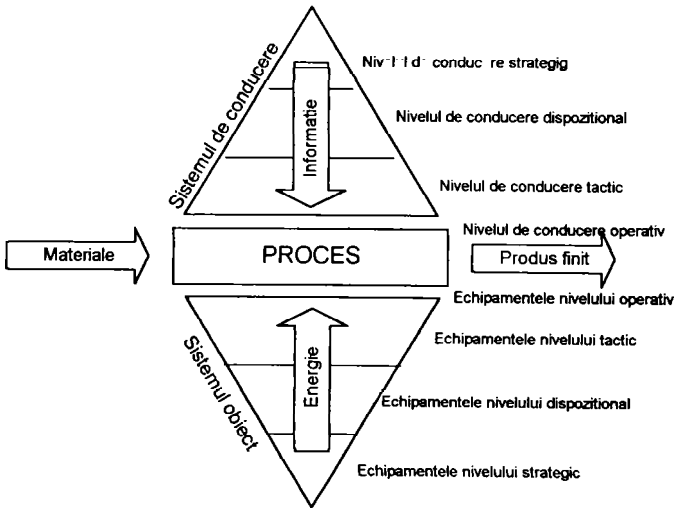


Fig. 2-10 Relaţia dintre sistemul de conducere şi sistemul obiect

Nivelurile de structurare ierarhică pot fi caracterizate prin indicatori de performanţă ale prelucrărilor informaţionale ce se realizează la diferite niveluri. Figura 2-11 prezintă aceste caracteristici sub forma unei histograme radiale utilizabilă atât în analiza unui sistem existent cât şi în sinteza unor noi sisteme de conducere [Sch'94]. Indicii de performanţă legaţi de capacitatea de memorare, viteza de comunicaţie, viteza de prelucrare etc. prezintă o relaţie logaritmică între diferitele niveluri. În Figura 2-11 valorile caracteristice ale fiecărui indicator de performanţă s-au specificat pentru fiecare nivel ierarhic şi s-au reprezentet pe câte o axă logaritmică. Aceste valori caracterizează performanţele funcţiunilor nivelului dat.

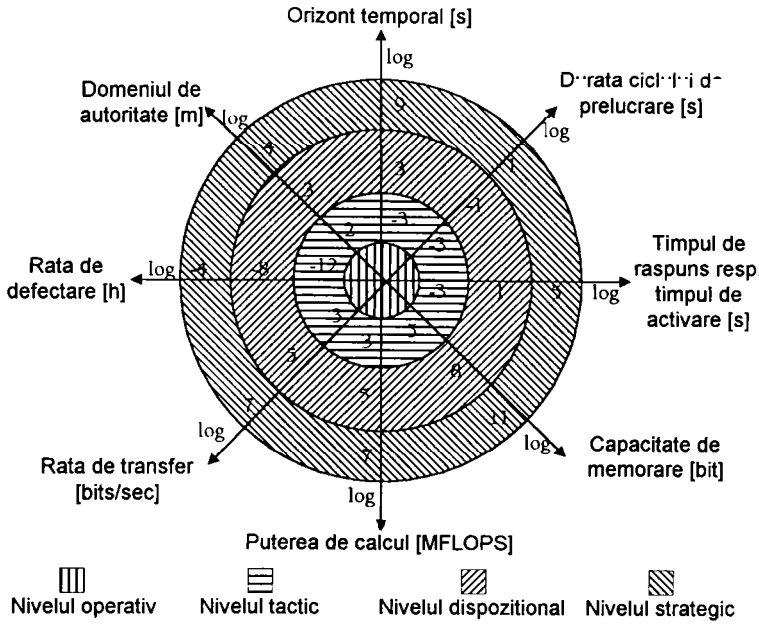


Fig. 2-11 Indicatori de performanţă pentru niveluri ierarhice.

În cazul sistemelor de automatizare structurarea pe niveluri creează o ierarhie funcţională care pune în evidenţă diversele categorii de "activităţi funcţionale" având semnificaţia de activităţi asiguratoare de condiţii pentru activităţile tehnologice direct productive necesare realizării transformărilor materialelor şi/sau informaţionale.

Funcţiunile se realizează prin operaţii, subactivităţi şi activităţi alocate ca misiuni (sarcini) oamenilor, maşinilor sau sistemelor om-maşină după reguli de adecvare tehnică şi economică. Modelul funcţional pune în evidenţă misiunile de indeplinit şi modelele necesare subiecţilor care paricipă la conducerea unui proces de producţie.

Conceptul de bază introdus este reprezentat în Figura 2-12 prin două exemple referitoare la două domenii de activitate distincte ale unei intreprinderi.

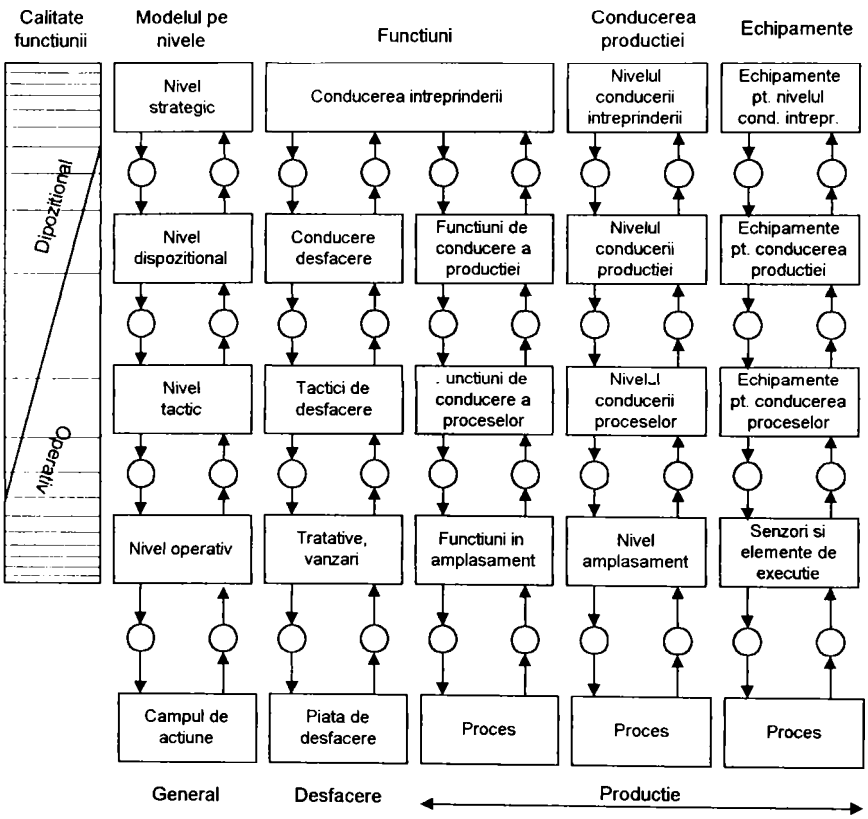


Fig. 2-12 Structura de bază a sistemului de conducere organizat pe patru niveluri ierarhice

Se poate observa cum funcţiunile sistemului de conducere, o dată cu apropierea de procesul condus, îşi schimbă caracterul de la cel dispoziţional la cel operativ. Nivelurile superioare asigură "baza normativă", adică definirea relaţiilor dintre scopurile urmărite şi stările impuse sistemului astfel încât să fie respectate legile, standardele şi normele de fabricaţie respectiv adaptarea la mediu conform unei strategii specifice intreprinderii.

Modelele nivelurilor de abstractizare superioare, intensionale, descriu scopul şi destinaţia sistemului. Modelele nivelurilor inferioare, extensionale, descriu componentele fizice ale sistemului în sensul că nivelurile inferioare asigură "baza fizică" a producţiei adică supravegherea proceselor tehnologice, a unor mărimi caracteristice procesului, localizarea şi eliminarea avariilor, întreţinerea profilactică etc.

Cu toate diferenţele principiale ale funcţiunilor celor două domenii prezentate - producţie respectiv desfacere - acestea pot fi structurate pe baza aceloraşi principii. Procesele din cele două domenii sunt total diferite dar obiectivele pe baza cărora acestea sunt influenţate sunt definite de către o aceeaşi funcţiune şi anume cea a conducerii intreprinderii. Cele de mai sus impun următoarele concluzii:

1. Modelul de structurare prezentat pentru sistemele tehnice poate fi aplicat, datorită generalităţii sale, şi sistemelor netehnice, de exemplu desfacerii produselor. Prin aceasta este posibilă reprezentarea interacţiunilor complexe dintr-un sistem de producţie (cerinţe tehnice, economice, ecologice) printr-un singur model.
2. Pentru stăpânirea complexităţii unui nivel ierarhic, este necesară divizarea funcţională suplimentară a acestuia.

2.2.3.3 *Granularitatea structurării funcţionale*

Granularitatea modelului poate fi făcută mai fină acţionând în două direcţii:

- pe orizontală, prin specializare funcţională;
- pe verticală, prin divizarea (descompunerea) suplimentară a nivelurilor.

Specializarea funcţională

Prin specializare funcţională se va înţelege divizarea unui nivel în diferite funcţiuni pe care acesta trebuie să le îndeplinească precum şi definirea modelelor informaţionale necesare. Specializarea funcţională se realizează pe baza unor atribute ce sunt moştenite de către funcţiunile nivelurilor inferioare.

Divizarea suplimentară a nivelurilor

Dacă complexitatea funcţională a unuia dintre nivelurile ierarhice o cere aceasta poate fi subdivizată pe baza aceloraşi patru niveluri. Nu toate nivelurile trebuie să fie neapărat prezente ci doar acelea ale căror caracteristici definitorii se regăsesc în funcţionalitatea nivelului respectiv subnivelului dat.

În acest sens definiţia modelului de structurare pe patru niveluri este relativă şi trebuie aplicată în contextul sistemului sau subsistemului modelat. Numărul nivelurilor de abstractizare variază de la sistem la sistem în funcţie de tipul sistemului şi scopul urmărit.

Această formă de prezentare ordonează procesul de modelare şi scoate în evidenţă faptul că, de cele mai multe ori, nu ajunge un singur model. Modelele ce se elaborează trebuie să fie acordate cu diversele puncte de vedere ale celor cărora li se adresează.

2.2.3.4 *Reguli pentru aplicarea principiilor de abstractizare funcţională*

În cadrul aplicării principiilor de abstractizare pe niveluri trebuie să se procedeze conform următoarelor reguli:

1. Diversele niveluri trebuie să ofere funcţiuni cu utilitate generală, adică să reprezinte o bună abstractizare;
2. Nivelurile trebuie să prezinte interfeţe bine definite şi legături minime către nivelul superior şi cel inferior şi să respecte principiul "secretului";
3. Nivelurile trebuie să realizeze funcţiuni complete şi închegate;
4. Un anumit nivel utilizează doar serviciile nivelului inferior adiacent. Trecerea peste niveluri nu este permisă.

Regulile de mai sus impun ca toate funcţiunile care au schimburi informaţionale intense să fie incluse în acelaş nivel.

Structurarea sistemelor complexe pe niveluri funcţionale şi comunicarea dintre acestea numai prin interfeţe bine precizate prezintă unele avantaje şi din punctul de vedere al simplificării şi creşterii gradului de testare [Ege'89]:

- Structurile, codurile şi algoritmii utilizaţi constituie o problemă strict internă a fiecărui nivel, ceea ce permite coexistenţa mai multor modalităţi de implementare ale aceluiaş nivel.
- Flexibilitatea sistemului cerşte, deoarece algoritmii şi mecanismele care implementează un anumit nivel se pot modifica fără să fie afectate interfeţele folosite. Funcţiunile pot fi implementate prin diverse tehnici, ceea ce asigură flexibilitate şi evoluţia sistemului.
- Un nivel poate fi simplificat sau eliminat în cazul în care unele respectiv toate serviciile sau funcţiunile sale nu sunt necesare.
- Validarea funcţională a unui sistem structurat pe niveluri funcţionale este mai uşor de realizat. Fiecare nivel în parte are o complexitate mai redusă, este mai uşor de analizat şi testat. Corelarea rezultatelor referitoare la comportarea tuturor nivelurilor permite obţinerea unor concluzii asupra comportării globale a sistemului.

Indiferent de metodele folosite pentru descompunerea orizontală şi/sau verticală, aprecierea unei modularizări se face pe baza unor criterii care s-au desprins din experienţa practică şi care sunt, în general, considerate acceptabile /ISO79/:

- Folosirea unui număr rezonabil de mic de niveluri. Fiecare nivel de abstractizare introduce un *cost-sistem* (overhead) suplimentar care, corelat şi cu o granularitate necorespunzătoare a obiectelor, poate afecta serios timpul de răspuns sau alţi indici de performanţă ai sistemului;
- Asigurarea unei interfeţe în punctul în care descrierea serviciului şi numărul de interacţiuni necesare utilizării lui sunt optime;
- Asigurarea unor graniţe la care interfeţele intra-nivel şi inter-niveluri să poată fi standardizate;
- Utilizarea unei metode de abstractizare adecvate pentru structurarea fiecărui nivel în parte, care să permită gruparea funcţiunilor înrudite şi separarea celor care prin natura lor sunt diferite, precum şi localizarea datelor şi a prelucrărilor;
- Utilizarea experienţei anterioare, câştigată prin măsurători şi studii statistice, simulare, experimentare sau în exploatare;
- Folosirea unor instrumente de evaluare, bazate pe metode analitice şi/sau de simulare, care să permită anticiparea calităţii modularizării obţinute.

2.2.3.5 *Modulul funcţional*

Structurarea funcţională se bazează pe conceptul de *modul funcţional*, privit ca o entitate logică de prelucrare, distribuire, control şi protecţie, fără constrângeri în privinţa realizării hardware. Un modul funcţional conţine următoarele elemente componente (Figura 2-13):

- algoritmul/algoritmele funcţionale (Problem Domain);
- datele specifice funcţiunii, într-o structură specifică (Data Management);
- interfeţe de operare şi supraveghere (Human Interaction);
- interfeţe către modulele funcţionale vecine din sistem (System Interaction)

În paranteză au fost date denumirile utilizate în [Sub'96]. Nivelurile funcţionale includ reţele de module funcţionale. Definiţia modulului funcţional nu impune soluţia tehnică de realizare.

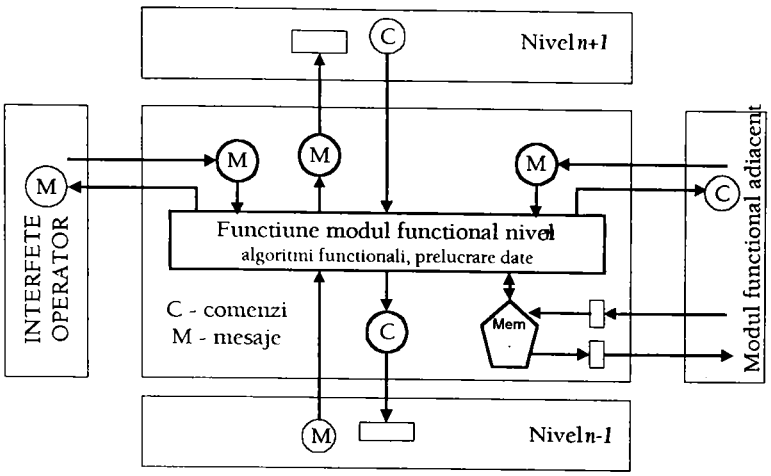


Fig. 2-13 Structura modulului funcţional

Modulul funcţional trebuie astfel conceput încât să permită, pe lângă integrarea funcţională, înţelegând prin aceasta funcţiunea strict legată de procesul producţiei, şi o integrare informaţională. Aceasta presupune asigurarea interfeţelor necesare în operare-supraveghere la nivel local şi central, precum şi a interfeţelor necesare proiectării şi diagnozei.

În Figura 2-13 prin "comenzi" se înţeleg, pe lângă comenzile propriu-zise, şi informaţiile necesare execuţiei comenzii date: argumente, parametrii etc. Prin "nivel" se va înţelege un nivel ierarhic conform celor de mai sus sau orice subdiviziune a acestuia. Schimbul de informaţii cu modulele funcţionale vecine se realizează prin mesaje sau prin accesul direct, prin partajarea memoriei, la variabile sau parametrii.

Nivelul superior utilizează serviciile nivelului inferior fără să aibă cunoştiinţă de felul cum acesta îşi realizează funcţiunile (principiul secretului). Serviciile care trebuie să fie realizate la fiecare nivel sunt: concentrare/sintetizare, prelucrare, evaluare, înregistrare/memorare, recepţie/transmisie. Deoarece aceste funcţiuni sunt prezente la fiecare nivel şi sunt utilizate la realizarea diverselor sarcini, acestea sunt considerate ca fiind funcţiuni generice. Aplicând consecvent principiul abstractizării se pot găsi diverse funcţiuni cu caracter general. Abstractizarea şi principiul secretului trebuie aplicate cu consecvenţă dacă se doreşte realizarea de niveluri autonome. Se asigură astfel posibilitatea schimbării - în cazul sistemelor tehnice - a unor componente, dacă progresul tehnic o cere, fără a periclita funcţionalitatea întregului sistem.

Nivelurile funcţionale se realizează prin reţele de module funcţionale, aşa cum se prezintă de exemplu în lucrarea [Krz'90]. Un modul funcţional, ca obiect conceptual al sistemului de automatizare, nu este legat de o anumită componentă hardware din sistem. Spre exemplu un regulator multifuncţional, ca echipament autonom, realizează mai multe funcţiuni, şi anume:

- prelucrarea rapidă a mărimilor analogice;
- realizarea diverselor structuri de regulatoare;
- prelucrarea mărimilor digitale;
- punerea la dispoziţie a bibliotecii de module funcţionale;
- configurarea şi parametrizarea unei structuri concrete (instanţiere).

Configurarea se realizează în mod interactiv, prin dialog, cu ajutorul pictogramelor, pe baza modulelor funcţionale ale bibliotecii regulatorului multifuncţional. Regulatorul multifuncţional asigură şi conexiunile cu componenetele funcţionale ale postului de conducere - nivelul conducerii proceselor - respectiv ale echipamentului electric şi ale procesului condus.

Dacă regulatorul multifuncţional este integrat într-o staţie de proces, prin magistrala de comunicaţie a staţiei, poate comunica cu toate modulele funcţionale localizate în aceeaşi staţie (se realizează reţeaua de module funcţionale la nivelul staţiei). Prin magistrala sistem se realizează legătura cu staţiile de conducere, configurare şi parametrizare respectiv de diagnoză.

2.2.3.6 *Avantajele structurării funcţionale*

a) Modulul funcţional şi POO

Se poate observa analogia dintre modulul funcţional şi obiectele POO. Această analogie este deosebit de importantă din punctul de vedere al implementării modulelor funcţionale deoarece POO poate fi folosită în mod direct atât la implementarea modulelor funcţionale cât şi a sistemelor de automatizare.

b) Modularizare

Modelarea conceptuală orientată pe obiecte, structurarea pe niveluri funcţionale şi specificarea funcţională pe baza modulelor funcţionale definite conform celor de mai sus şi a structurii din Figura 2-13 constituie mijloace puternice de modularizare. Calitatea modularizării este la originea obţinerii şi altor deziderate, cum ar fi:

a) comunicaţia eficientă dintre proiectantul, beneficiarul şi operatorul sistemului de automatizare;

b) siguranţa în realizare (costuri, termene) şi în exploatare;

c) stabilitatea funcţională;

d) dezvoltarea şi implementarea incrementală a sistemului;

e) valorificarea experienţei acumulate;

f) definirea şi implementarea unor metode de proiectare, elaborare şi validare asistate de calculator;

g) flexibilitatea funcţională, incluzând extensibilitatea şi adaptabilitatea;

h) flexibilitatea structurală prin distribuire;

i) fiabilitatea şi simplitatea utilizării;

j) definirea şi implementarea autonomă (standardizarea) a tehnicilor de comunicaţie între modulele funcţionale.

2.2.3.7 *Structura funcţională a sistemelor de conducere*

Un proces de producţie complex poate fi divizat în segmente de procesare cu prelucrări autonome care din materialele/semifabricatele de la intrare (produse de intrare) realizează produse de ieşire, eventual produsul finit [Stö'89], [Wag'90]. Acestei structuri, determinate de cerinţe tehnologice şi care este descrisă prin modelul pe faze al producţiei, trebuie să i se adapteze sistemul de conducere al producţiei [Pol'85], Pol'87], [Rep].

În Figura 2-14 sistemul de conducere al producţiei s-a structurat pe patru niveluri conform modelului de referinţă, astfel:

1. Nivelul conducerii intreprinderii
2. Nivelul conducerii producţiei
3. Nivelul conducerii proceselor
4. Nivelul funcţiunilor din amplasament,

Pentru un caz concret pot fi definite niveluri suplimentare conform principiului divizării suplimentare a nivelurilor. Spre exemplu, nivelul conducerii proceselor, în cazul fabricaţiei flexibile, se va subdiviza în nivelul conducerii celulelor flexibile respectiv nivelul conducerii maşinilor unelte. În Figura 2-14 acelaş nivel s-a subdivizat în "nivelul funcţiunilor centrale" respectiv :nivelul funcţiunilor de coordonare".

Prin funcţiunile sale, sistemul de conducere trebuie să asigure realizarea comenzii de producţie prin asigurarea resurselor necesare, crearea reţetei de comandă, transpunerea unor segmente ale reţetei pe instalaţii de producţie şi conducerea procesului de producţie prin funcţiuni de procesare elementare.

Se prezintă mai jos (Figura 2-14) funcţiunile necesare satisfacerii cerinţelor impuse sistemului de conducere. Aceste cerinţe, aliniate modelului ierarhic, vor fi structurate prin module funcţionale alocate unor componente ale sistemului de conducere.

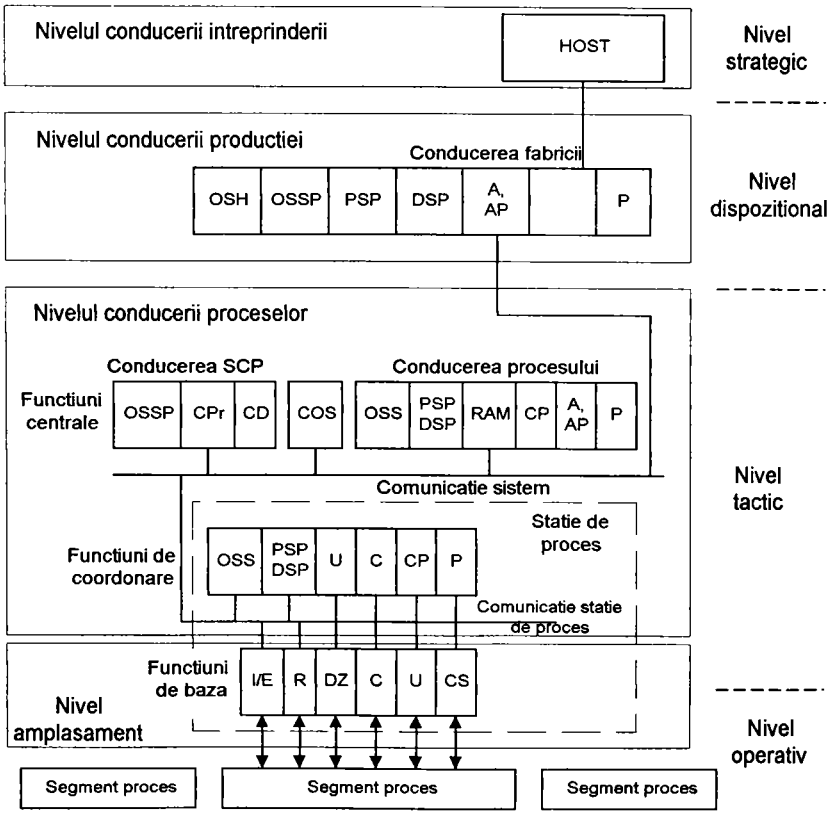


Fig. 2-14 Structura funcţională a unui sistem de conducere

Figura 2-14 prezintă structura funcţională pentru un sistem de conducere din domeniul industriei chimice. S-au delimitat următoarele categorii de funcţiuni:

Funcţiuni la nivelul amplasamentului:

a). Funcţiuni de comunicaţie locală

Într-o staţie de proces sunt cuprinse funcţiunile necesare conducerii unui utilaj tehnologic sau a unei părţi al acestuia. Aceste funcţiuni comunică între ele printr-un sistem de comunicaţii care, de cele mai multe ori, are un caracter proprietar (nestandardizat).

b). Funcţiuni de bază

Conducerea unui segment de producţie/procesare necesită funcţiuni de bază şi funcţiuni de coordonare. Funcţiunile de bază, prin intermediul interfeţelor de proces, au legături directe în instalaţiile de producţie prin senzori şi elemente de execuţie. Funcţiunile de bază cuprind şi funcţiunile de reglare şi comandă specifice ale echipamentelor de producţie. Pentru mărirea disponibilităţii, funcţiunile de bază trebuie să fie autonome şi să permită echiparea redondantă.

Conducerea funcţiunilor de bază se realizează prin intermediul funcţiunile de coordonare. Pentru claritatea în ordonarea ierarhică trebuie evitată comunicaţia directă dintre funcţiunile de bază. Funcţiunile de bază trebuie să fie autonome pentru a asigura, pe de o parte realizarea incrementală a sistemului, în sensul extensibilităţii etapizate iar pe de altă parte pentru a permite punerea în funcţiune şi operarea sistemului prin intermediul funcţiunilor de operare/supraveghere chiar la nivelul funcţiunilor de bază.

Funcţiunile de bază trebuie să se adapteze mediului hardware fără a-şi pierde identitatea şi autonomia. Trebuie asigurată activarea respectiv dezactivarea dinamică a funcţiunilor de bază.

În continuare se descriu unele funcţiuni de bază specifice industriei chimice.

- Funcţiuni intrare/ieşire (I/E)
  Sunt funcţiunile standard de intrare/ieşire prin semnale analogice şi binare. Este necesar a asigura şi posibilitatea acţionării/testării acestora prin funcţiunile de operare-supraveghere.
- Funcţiuni de reglare (R)
  Sunt necesare regulatoare autonome cu funcţiuni "AP" de automat programabil şi "Back up", algoritm PID şi opţional cu caracteristici de autoacordare. Funcţiunile AP şi "Back up" vor realiza comutările fără şocuri.
- Funcţiuni de dozare (DŽ)
  După preluarea parametrilor, funcţia de dozare, trebuie să lucreze autonom executând inclusiv comenzile secvenţiale şi achiziţia de date. Trebuie să ofere o interfaţă de comunicare a valorilor momentane (de exemplu debite pentru compensare).
- Funcţiuni de control (C)
  Sunt necesare module de control a acţionărilor de motoare şi ventile. Funcţiunile de control trebuie să fie programabile prin limbaje specializate. Viteza de prelucrare respectiv configuraţia funcţiunilor de control trebuie să fie proiectabile.
- Funcţiunile universale (U)
  Cuprind funcţiunile de măsură, control, reglare şi programare. Aceste funcţiuni pot fi realizate prin limbaje de comandă sau prin macromodule care asigură realizarea de operaţii tipice în conducerea proceselor (cântărire, dozare etc.). Prin limbaje de nivel înalt se vor realiza funcţiunile specifice proiectului dat.

- Funcţiuni de conectare subsisteme (CS)
  În situaţia în care anumite semnale din proces nu sunt aduse în mod direct la staţia de proces, acesta trebuie să asigure conectarea subsistemelor care preiau, transmit sau prelucrează local semnale de proces. Astfel de situaţii se ivesc atunci, când se doreşte reducerea necesarului de cabluri şi cablare, sau când se integrează funcţiuni speciale ca senzori inteligenţi, controloare de cântărire/dozare sau comenzi integrate ale unor maşini.

Este de dorit, ca o staţie de proces să dispună de funcţii de cuplare atât pentru subsistemele proprii cât şi pentru subsisteme străine.

Este de dorit realizarea unor sisteme deschise. În acest sens se vor folosi, pe cât posibil procedee standard de integrare respectiv sisteme de comunicaţii standardizate. Dacă acest lucru nu este posibil se va asigura flexibilitatea necesară (staţici de proces).

Funcţiunile de supraveghere-operare se vor extinde, prin funcţiunile de conectare (integrare), şi asupra acestor subsisteme. Tot aceste funcţiuni de integrare trebuie să asigure trecerea de la un sistem bazat pe regulatoare automate la un sistem de conducere integrat.

Funcţiuni la nivelul conducerii proceselor:
La acest nivel se deosebesc următoarele categorii de funcţiuni:

a) Funcţiuni de coordonare legate de un segment de producţie (procesare);
b) Funcţiuni centrale: funcţiuni pentru conducerea segmentului din procesul de producţie şi a subsistemului de conducere însăşi precum şi funcţiuni de diagnoză şi proiectare ale acestuia. Aceste susbsisteme de conducere sunt realizate, în tehnologiile actuale, printr-o catregorie de echipamente numite staţii de proces.
c) Funcţiuni de comunicaţie la nivelul subsistemului de conducere.

Funcţiuni de coordonare

Prin funcţiunile de coordonare se realizează coordonarea şi supravegherea funcţiunilor de bază necesare conducerii unui utilaj tehnologic sau a unei părţi al acestuia. Funcţiunile de coordonare trebuie să dispună de interfeţe de comunicaţie cu funcţiunile centrale de conducere. Aceste funcţiuni vor fi realizate modular astfel încât să permită, pe de o parte, o proiectare şi realizare incrementală iar pe de altă parte realizarea unui sistem tolerant la defectări. Aceste funcţiuni au menirea de a asigura, prin coordonarea funcţiunilor de bază sau a funcţiunilor specifice aparaturii, controlul, reglarea respectiv programabilitatea (prin diverse limbaje) cu scopul realizării unor funcţiuni specifice aplicaţiei concrete. Se deosebesc următoarele funcţiuni de coordonare:

- Funcţiuni de operare-supraveghere staţie de proces (OSSP)
- Funcţiuni de proiectare staţie de proces (PSP)
- Funcţiuni de diagnoză staţie de proces (DSP)
- Funcţiuni de coordonare universale (U). Aceste funcţiuni relizează funcţiunile generice de "reglare", "control" şi "proramare" la nuvelul funcţiunilor de bază respectiv al funcţiunilor de reglare sau control specifice unor echipamente ale utilajului tehnologic.
- Funcţiuni de conducere-coordonare (C). Prin aceste funcţiuni se realizează, pe baza funcţiunilor de bază şi a funcţiunilor de coordonare universale, funcţiunile de reglare şi control specifice unor părţi ale utilajului tehnologic. Aceste funcţiuni sunt dependente de reţetele ce se realizează pe utilajele tehnologice respective.

- Funcţiuni de conducere a proceselor (CP). Aceste funcţiuni trebuie să pună la dispoziţie un limbaj de programare care să permită, prin funcţiunile de coordonare universale şi funcţiunile de bază, implementarea sistemului de conducere, pe baza reţetei de comandă a unei instalaţii sau a unei părţi a acesteia
- Funcţiuni de programare (P). Modulul funcţional trebuie să fie astfel concepute încât să fie exploatabile printr-un program scris într-un limbaj de nivel înalt.

Funcţiunile centrale

Funcţiunile centrale sunt de două categorii:

1) Funcţiuni centrale pentru conducerea sistemului (ingineria de sistem)
- funcţiuni de operare-supraveghere staţie de proces (OSSP)
- funcţiuni centrale de proiectare (CPr)
- funcţiuni centrale de diagnoză (CD)

2) Funcţiunile centrale pentru conducerea proceselor.
- funcţiuni centrale de operare-supraveghere (COS)
- funcţiuni centrale de operare-supraveghere staţie (OSS)
- funcţiuni centrale de proiectare staţie de proces (PSP)
- funcţiuni centrale de diagnoză staţie de proces (DSP)
- funcţiuni centrale pentru generarea, recepţia şi analiza mesajelor (RAM)
- funcţiuni centrale pentru conducerea proceselor (CP)
- funcţiuni centrale de arhivare (A)
- funcţiuni centrale de analiză posteveniment (AP)
- funcţiuni centrale de programare (limbaj de nicel înalt) (P)

Funcţiuni la nivelul conducerii producţiei:

Nivelul conducerii producţiei integrează funcţiunile administrative, logistice, economice şi tehnice necesare conducerii unei fabrici. Tehnologiile informaţionale care sprijină realizarea funcţiunilor acestui nivel utilizează tehnica de calcul cu disponibilitate ridicată. La nivelul conducerii proceselor se realizează comenzile de producţie (solicitări referitoare la produse, cantităţi, calităţi, termene) prin procese de producţie specifice. Se presupune că instalaţiile şi echipamentele sunt funcţionale, reţeptura de comandă stă la dispoziţie, şi că resursele necesare stau la dispoziţie. Procesul de producţie se realizează prin segmente autonome. Acestei structuri orizontale trebuie să i se adapteze sistemul de conducere ca parte integrantă a echipamentului de producţie.

Funcţiunile la nivelul conducerii intreprinderii sunt următoarele:
- funcţiuni de operare-supraveghere host (OSH)
- funcţiuni de operare-supraveghere staţii de proces (OSSP)
- funcţiuni de proiectare staţii de proces (PSP)
- funcţiuni de diagnoză staţii de proces (DSP)
- funcţiuni de conducere a producţiei (CP)
- funcţiuni de arhivare (A)
- funcţiuni centrale de analiză posteveniment (AP)
- funcţiuni centrale de programare (limbaj de nicel înalt) (P)

Funcţiuni la nivelul conducerii intreprinderii.

Nivelul conducerii intreprinderii conţine funcţiunile strategice necesare conducerii tuturor compartimentelor: producţie, desfacere, dezvoltare produse. Sistemele de programe care sprijină realizarea funcţiunilor acestui nivel au fost realizate pe sisteme HOST, utilizând baze de date şi sisteme de comunicaţii ierarhice. Calculatoarele personale pot mări flexibilitatea în evaluarea datelor.

2.2.3.8 *Sisteme de conducere distribuite*

Sistemele cu microprocesoare (logica programată), sistemele de comunicaţii de date standardizate şi modulele funcţionale sprijină în mod eficient, printr-un efect sinergic, realizarea de sisteme de conducere distribuite şi flexibile. Cu ajutorul echipamentelor cu microprocesoare pot fi realizate funcţiunile tuturor nivelurilor ierarhice ale sistemelor de conducere începând cu senzorii şi elementele de execuţie inteligente şi terminând cu sistemele de conducere a producţiei. Aceste echipamente comunică între ele prin sisteme de comunicaţii de date.

S-au definit sisteme de comunicaţii de date standardizate pentru comunicaţia între respectiv în cadrul fiecărui nivel ierarhic. Figura 2-15 prezintă localizarea unor sisteme de comunicaţii de date reprezentative, în cadrul modelului pe niveluri ierarhice.

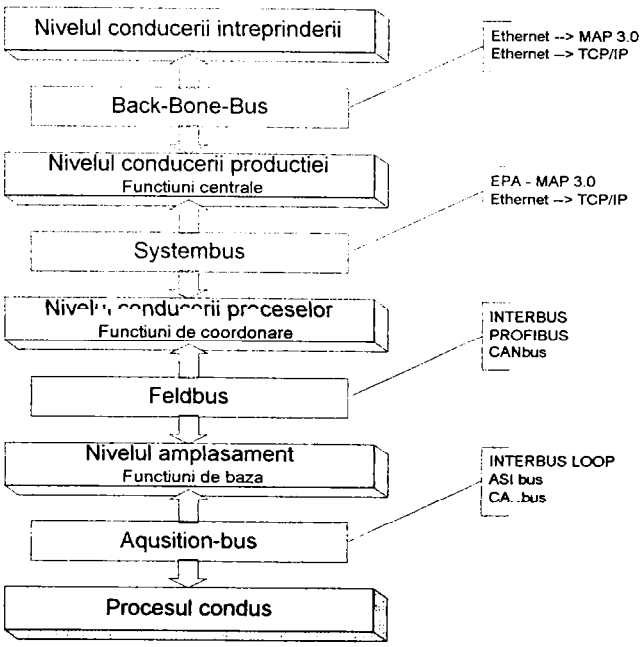


Fig. 2-15 Sisteme de comunicaţii standardizate

Echipamentele care implementează funcţiunile sistemului - numite şi staţii - sunt cuplate, din punct de vedere informaţional, cu:

- procesul şi/sau
- reţeaua de comunicaţii de date şi/sau

- operatorul sistemului.

Pentru a asigura adecvarea tehnică şi economică optimală sistemele de conducere conţin staţii şi reţele de comunicaţii de date cu diferite caracteristici de performanţă. Figura 2-11 prezintă cuantificarea, la nivelul ordinului de mărime, a caracteristicilor de bază - rata de transfer pentru reţeaua de comunicaţii de date respectiv durata ciclului de prelucrare pentru staţii - necesare în realizarea funcţiunilor de la diversele niveluri ierarhice.

Reţelele de comunicaţii de date şi staţiile formează o reţea informaţională sub forma unui sistem spaţial şi funcţional distribuit aşa cum se prezintă în Figura 2-16. În acestă figură s-au folosit următoarele notaţii:

CCI - calculator pentru conducerea intreprinderii;

CCPd - calculator pentru conducerea producţiei;

CCPr - calculator pentru conducerea proceselor;

SLCPr - staţie de lucru pentru conducerea proceselor;

SLCS - staţie de lucru pentru conducerea sistemului (de conducere a producţiei);

FC - funcţiuni de coordonare;

FB - funcţiuni de bază;

SI - senzor inteligent;

EEI - element de execuţie inteligent.

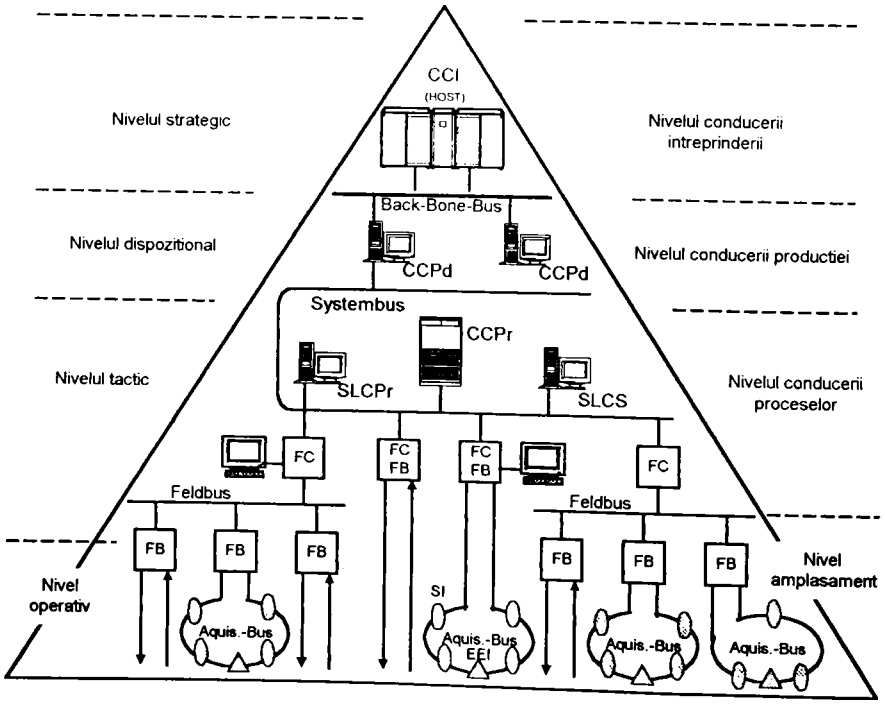


Fig. 2-16 Sistem de conducere distribuit

2.2.4 Principiul transformării, modelarea orientată pe evenimente

2.2.4.1 *Operatorul de tranziţie*

Principiul decompoziţiei şi al abstractizării funcţionale permite o structurare statică a sistemelor. Principiile de transformare permit structurarea sistemelor conform caracteristicilor lor dinamice şi crearea de modele care descriu procesele de transformare materială şi informaţională din sistem, adică dinamica sistemului.

În fiecare moment un sistem fizic (deci şi unul tehnic) se poate caracteriza, macroscopic, la nivelul fizicii clasice, printr-o mulţime finită de mărimi de stare. Prin variaţia mărimilor de stare de la un set de valori la un alt set de valori are loc trecera sistemului de la o stare dată spre o altă stare dată, adică se desfăşoară un proces fizic, al cărui mers depinde de starea iniţială şi de întreaga succesiune de stări ale sistemului până în starea finală.

Procesele dinamice din sistemele fizice (tehnice) corespund întotdeauna unor transferuri de substanţă, energie şi/sau informaţie de la unele obiecte ale sistemului spre alte obiecte ale acestuia.

În cazul unui sistem tehnic productiv, la nivelul conducerii producţiei, interesează mărimile de stare care caracterizează procesul tehnologic de producţie. Schimbările de stare se realizează datorită operaţiilor tehnologice. Pentru modelarea acestor sisteme se introduce noţiunea de *operator* (prin analogie cu noţiunea din matematică).

Din punctul de vedere al modelării, operatorul are rolul de a trece (tranzita) sistemul dintr-o stare în alta, de aceea putem să-l numim *operator de tranziţie*. Un sistem dat conţine un număr de operatori cu ajutorul cărora sistemul poate parcurge traiectoria de stare.

În exemplul din Figura 2-17, prin supunerea unui anumit produs procesului de transformare/prelucrare, după un anumit operator, se obţine un alt produs.

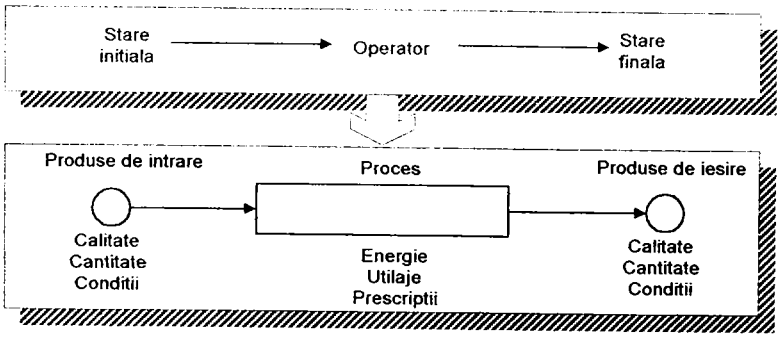


Fig. 2-17 O imagine idealizată a producţiei. Modelul pe faze.

În cazul unui sistem concret operatorul de tranziţie se activează dacă sunt satisfăcute anumite condiţii specifice, ca de exemplu:

1. Toate mijloacele necesare conducerii procesului (agregate, prescriptii de producţie, sisteme de conducere) stau la dispoziţie şi sunt funcţionale;
2. Produsul/produsele de intrare sunt disponibile la calitatea şi cantitatea necesară;
3. Stă la dispoziţie energia în toate formele necesare, la calitatea şi cantitatea necesară;
4. Toate echipamentele de protecţie indică stări funcţionale;
5. S-a validat pornirea activarea operatorului de tranziţie.

Odată cu validarea pornirii se iniţiază procesul de transformare a produselor de intrare rezultând produse de ieşire diferite de cele de intrare. O mulţime de astfel de procese de transformare interconectate formează un proces de producţie. În cazul general se realizează conexiuni complexe cuprinzând şi reacţii inverse. Se obţine astfel o reţea cauzală numită în cadrul teorici grafurilor "reţea Petri".

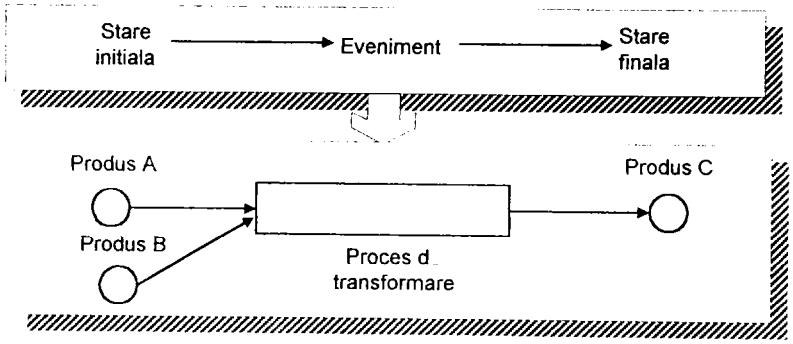


Fig. 2-18 Exemplu de reţea Petri

O reţea Petri este un model grafic de tipul grafurilor orientate, cu două categorii de noduri:

a) locaţii (care modelează condiţii)

b) tranziţii (care modelează acţiuni)

Relaţiile dintre evenimentele care pot avea loc şi condiţiile necesare pentru ca anumite evenimente să se producă efectiv sunt reprezentate prin arcele grafului, care stabilesc legăturile orientate dintre locaţii şi tranziţii, precum şi dintre tranziţii şi locaţii, întrucât prin producerea unui eveniment are loc modificarea atât a condiţiilor de apariţie a evenimentului, cât şi a celor care decurg din producerea evenimentului (şi care pot determina apariţia altui eveniment).

De regulă locaţiile se reprezintă grafic prin cercuri iar tranziţiile prin dreptunghiuri. Faptul că o condiţie este indeplinită se reprezintă grafic prin introducerea unui simbol în cercul poziţiei aferente condiţiei respective. Simblurile constituie marcarea locaţiei. Semnificaţia locaţiilor şi tranziţiilor se precizeaza prin "etichetare".

Reţelele Petri sunt deosebit de utile în modelarea sistemelor cu procese concurente. În practică se utilizează diverse tipuri de reţele Petri cu diverse puteri de modelare.

În cazul sistemelor de producţie se încearcă decuplarea fluxurilor de materiale prin intermediul unor elemente de acumulare tehnologică pentru a împiedica propagarea instantanee a perturbaţiilor prin tot sistemul. Segmentele de sistem decuplate pot fi considerate, între anumite limite, independente respectiv procesele de transformare din aceste segmente, concurente. Prin modelarea, pe baza reţelelor Petri, a unor astfel de sisteme decuplate se oferă posibilitatea conducerii pe baze logistice a procesului de producţie.

2.2.4.2 *Transformările de stare prin exemplul modelului pe faze al producţiei*

În scopul ilustrării principiilor de mai sus, se prezintă ideile de bază ale structurării proceselor prin intermediul producţiei într-o uzină chimică. Structurarea trebuie să ţină cont de criteriile de calitate şi de cele logistice de conducere.

Modelul pe faze, împrumutat din domeniul ingineriei de programe, joacă un rol important în analiza proceselor de producţie. Figura 2-19 prezintă un exemplu care scoate în evidenţă elementele grafice utilizate şi procedura de detaliere în "adâncime". Pe aceste diagrame dreptunghiurile reprezintă segmente de procesare (transformare materială) iar cercurile reprezintă stări respectiv condiţii. Segmentele de procesare pot fi concepute ca operatori care modifică proprietăţile produsului. Modelul pe faze satisface premisa posibilităţii modelării procesului la diferite niveluri de detaliere ("adâncime"). Aceasta permite, deasemenea, detalierea reţetei de comandă până la nivelul fizic elementar.

Modulul pe faze, aşa cum se prezintă în Figura 2-19 este un model semantic orientat pe obiecte. Cu ajutorul acestuia poate fi descrisă structura unui proces de producţie în cadrul unui model informaţional. Pentru un caz concret acest model informaţional permite, prin achiziţia de cunoştinţe din diverse resurse, crearea bazei de cunoştinţe pentru sistemul dat.

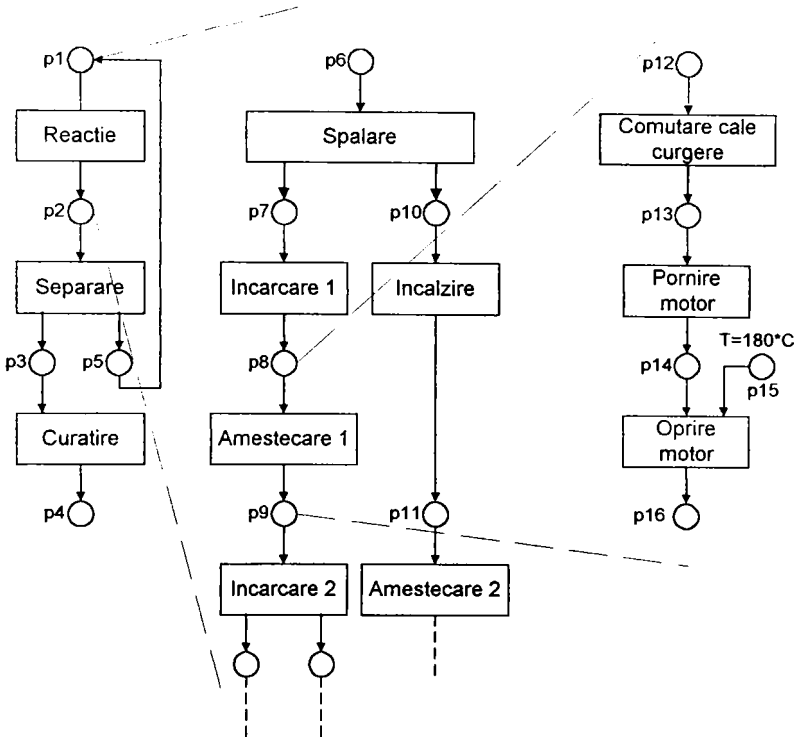


Fig. 2-19 Rafinarea modelului pe faze

Locaţiile din Figura 2-19 au semnificaţia de mai jos.

- p1 – condiţiile iniţiale pentru iniţierea reacţiei sunt satisfăcute;
- p2 – reacţie terminată;
- p3, p5 – separare terminată;
- p4 – curăţire terminată;
- p6 - condiţiile iniţiale pentru spălare sunt satisfăcute;
- p7, p10 - după terminarea spălării sunt îndeplinite condiţiile iniţiale pentru începerea "Încărcării 1" şi a "Încălzirii";

- p9, p11 - operaţiile de "Încărcare 2" şi "Amestecare 2" pot fi începute numai dacă "Amestecare 1" şi "Încălzire" sunt amândouă terminate (sincronizare);
- p12 - "Încărcare 2" terminată;
- p13 - cale curgere comutată;
- p14 - motor în funcţiune;
- p16 - motor deconectat.

Sarcina prioritară în industria chimică este asigurarea reproductibilităţii. Printr-o reţetă de producţie se înţelege mulţimea informaţiilor necesare şi suficiente care asigură reproductibilitatea unui produs dat. Aceasta va conţine următoarele componente:

- Specificarea proprietăţilor produselor de intrare, interfazice şi de ieşire (aspectul calităţii) precum şi bilanţurile cantitative ale acestora (atribute produs);
- Descrierea elementelor de procesare precum şi a interacţiunile lor statice şi dinamice (structura de reţea a procesului);
- Specificarea parametrilor calitativi, staţionari respectiv dinamici, impuşi pentru fiecare element/pas de procesare într-o formă de prezentare adecvată: histograme, grafice, tabele, etc.(atribute proces).

În cazul în care se impune optimizarea procesului de producţie conform anumitor mărimi de calitate, este necesară elaborarea unui model matematic care descrie interdependenţa parametrilor procesului precum şi dependenţa de aceştia a mărimilor de calitate a produselor. Modelele statice şi dinamice ale proceselor precum şi tehnicile de simulare bazate pe acestea constituie obiectul principal al informaticii tehnice. Aceste modele stau la baza metodelor moderne de reglare utilizate în conducerea proceselor şi a noilor tehnici de comunicaţie proces-operator uman.

Conducerea proceselor pe baza modelelor matematice creşte siguranţa în exploatare utilizând şi metode de măsurare bazate pe modele matematice care permit recunoaşterea timpurie a unor tendinţe periculoase. Elaborarea modelelor matematice în sensul celor de mai sus, precum şi simularea pe calculator constituie obiectul disciplinei informaticii tehnice. Automatizarea completă a procesului de producţie se realizează prin operaţii logistice realizabile în situaţia în care reţetele pot fi tratate cu calculatorul. Conducerea logistică a procesului de producţie trebuie să asigure ca, la pornire, oprire sau în cazul perturbaţiilor efectele să fie pe cât posibil localizate prin utilizarea acumulatoarelor tehnologice.

2.2.4.3 *Reţele Petri ca modele sistem*

Reţelele Petri reprezintă instrumentul de bază în modelarea sistemelor pe baza operatorului de tranziţie. Reţelele Petri, concepute să modeleze sisteme distribuite în care concurenţa, comunicarea şi paralelismul ocupă un loc central, au devenit un instrument de modelare de bază într-o asemnea topică. În acest sens reţelele Petri se bucură de trei atuuri fundamentale:

(i) **simplitatea:** Teoria reţelelor Petri face apel la un număr redus de concepte elementare dar care sunt combinate într-o mare varietate. Există doar o singură noţiune de bază de *nedeterminism*, o singură noţiune de bază de *concurenţă* şi o singură noţiune de bază de *secvenţă*.

(ii) **generalitatea:** Generalitatea reţelelor Petri poate fi explicată cel puţin din trei puncte de vedere:

a) Reţelele Petri pot servi ca intermediari în translatarea unor proprietăţi de la un anumit model (algebra proceselor) către mediul exterior prin utilizarea lor în descrierea semanticii respectivelor modele.

b) Reţelelor Petri li se pot asocia relativ uşor diverse tipuri de semantici: secvenţe de tranziţii finite sau infinite, secvenţe de mulţimi de tranziţii, procese.

c) Multe proprietăţi importante ale sistemelor modelate (viabilitate, mărginire, reversibilitate) pot fi exprimate cu destulă precizie prin intermediul reţelelor Petri.

(iii)**adaptabilitatea:** Modificări minore aduse modelului clasic de reţea Petri conduc la modele speciale ce pot surprinde aspecte ca temporizare, probabilitate şi incertitudine etc., capabile să le facă utile în domenii cât mai variate.

Următoarele caracteristici ale reţelelor Petri sunt esenţiale pentru modelarea sistemelor de automatizare:

1) reţelele Petri neinterpretate descriu o ordonare parţială dintre evenimente sau clase de evenimente;

2) reţelele Petri tratează atât evenimentele din sistemul modelat cât şi stările acestuia;

3) reţelele Petri pot fi executate pe baza mai multor strategii ce implementează diverse constrângeri. Prin urmare poate fi modelată şi analizată flexibilitatea sistemelor;

4) reţelele Petri pot fi interpretate în sensul limitării secvenţelor de tranziţii executabile permiţând astfel studiul caracteristicilor ce derivă din contradicţia "optimalitate versus flexibilitate".

a) Reţelele Petri şi axiomele-sistem

Reţelele Petri clasice se descriu matematic şi grafic ca în Tabelul 2. Acelaş tabel prezintă şi modul în care reţelele Petri descriu elementele definitorii surprinse în axiomele-sistem.

a.1) Principiul structural

Reţeaua Petri clasică ca un sistem abstract (matematic) constă din două mulţimi de elemente - mulţimea locaţiilor, P, şi mulţimea tranziţiilor, T, - şi relaţiile dintre acestea surprinse în relaţia de flux $F=(P\times T)\cup(T\times P)$. Elementele sistemului abstract sunt caracterizate prin mărimi (marcări) a căror valoare constituie marcarea, **m** (cu mulţimea marcărilor M), a reţelei şi care descrie starea sistemului. Elementele sistemului modelat corespund locaţiilor respectiv tranziţiilor reţelei iar relaţiile dintre acestea relaţiei de flux reprezentată grafic prin arce orientate.

a.2) Principiul de decompozabilitate

Elementele reţelei (locaţii, tranziţii), dacă reprezintă subsisteme, la rîndul lor pot fi desrise prin reţele (subreţele), astfel:

$N=(P,T;F,M)$

$P\rightarrow (P_P,T_P;F_P,M_P)$

$T\rightarrow (P_T,T_T;F_T,M_T)$

a.3) Principiul cauzalităţii

Marcările realizabile în reţele Petri, stările realizabile ale sistemului modelat (traiectoria de stare), sunt determinate de sintaxa reţelei. Aceasta asigură respectarea strictă a principiului cauzalităţii în sensul că marcările-efect sunt univoc determinate de marcările-cauză. Exprimat mai concret, marcările-cauză determină tranziţiile concesionate, cele ale

căror execuţie este permisă, iar tranziţiile efectiv executate determină marcările-efect. Într-o exprimare vectorială, dacă **m** este vectorul de marcare, **N** matricea de incidenţe a reţelei iar **p** vectorul tranziţiilor executate, relaţia dintre marcările cauză şi marcările-efect se exprimă astfel:

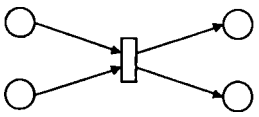
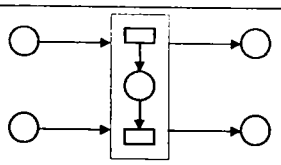
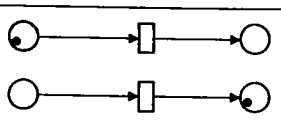
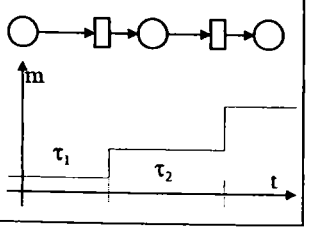
$$\mathbf{m}_{k+1}=\mathbf{m}_k+\mathbf{N}\bullet\mathbf{p}_k \ .$$

a.4) Principiul temporalităţii

Dimensiunea temporală a desfăşurării proceselor, a execuţiei acţiunilor (tranziţiilor) în reţele Petri este descrisă prin diverse concepte care specifică desfăşurarea temporală a fenomenelor în sistemul modelat prin alocarea de parametrii temporali - de natură deterministă sau stochastică - elementelor reţelei adică locaţiilor sau tranziţiilor.

Caracteristica esenţială a modelării prin reţele Petri este faptul că se descriu în mod explicit relaţiile cauzale din sistemul modelat. Analiza reţelei va pune în evidenţă proprietăţile sistemului imprimate de către relaţiile cauzale. Caracteristicile temporale ale sistemului se descriu prin proiecţia pe o axă temporală a desfăşurării fenomenelor (apariţia evenimentelor modelate prin execuţia tranziţiilor). Se asigură astfel ca rezultatele analizei să nu fie dependente de poziţia observatorului faţă de sistem respectiv de temporizările, întotdeauna instabile, din cadrul sistemului.

Tabelul 2

Axiome sistem	Reţele Petri	
	Notaţia matematică	Reprezentarea grafică
1. Principiul structural: - elemente, - interrelaţii	$P=\{p_1, p_2, \ldots, p_{|P|}\}$ $T=\{t_1, t_2, \ldots t_{|T|}\}$ $F=(P\times T)\cup(T\times P)$ $M=\{m_1, m_2, \ldots m_{|M|}\}$ $N=(P,T;F,M)$	
3. Principiul de decompozabilitate	$N=(P,T;F,M)$ $P\rightarrow (P_P,T_P;F_P,M_P)$ $T\rightarrow (P_T,T_T;F_T,M_T)$	
3. Principiul cauzal	$\mathbf{m}_{k+1}=\mathbf{m}_k+\mathbf{N}\bullet\mathbf{p}_k$	
4. Principiul temporal	$p_i(\tau)=\tau_i$	

b) Reţelele Petri şi diversitatea punctelor de vedere asupra sistemului de automatizări industriale

Diversitatea punctelor de vedere asupra sistemelor de automatizări industriale este surprinsă prin semantici asociate reţelelor Petri printr-un procedeu de "interpretare" a reţelei. Reţelele Petri sunt utilizabile în cele mai diverse domenii [Sch'91b] şi în practic toate etapele ciclului de viaţă a sistemelor de atumatizări industriale.

c) Tehnici şi instrumente de analiză

Diverse tehnici de analiză (invarianţi, mulţimea realizabilă, temporizare) permit studiul caracteristicilor dinamice, imprimate de structura reţelei, respectiv a caracteristicilor de performanţă. De cele mai multe ori, din cauza complexităţii reţelei, se impune utilizarea tehnicii de calcul. Stau la dispoziţie diverse instrumente de analiză [Sch'91b] pentru toate fazele modelării şi analizei.

d) Puterea de modelare. Clase de reţele Petri.

Pe lângă reţelele Petri de bază (reţele Condiţii/Evenimente, reţele Locaţie /Tranziţie, reţele "Free-Choice") s-au dezvoltat şi clase de reţele Petri speciale. Acestea au la bază aceleaşi principii dar se deosebesc prin funcţionalităţile specifice legate de concesionarea respectiv execuţia tranziţiilor şi a celor legate de individualizarea marcărilor. Multe dintre aceste clase se adresează unor domenii speciale. Clasele de reţele Petri speciale aplicabile la o gamă largă de probleme sunt următoarele:

a) reţelele Petri colorate (Coloured Petri Nets, C/P Nets), [Jen'92];

b) reţelele Petri Predicate/Tranziţii (Predicate/Transition Nets, P/T Nets), [MOS'93];

c) reţelele Petri Fuzzy (Vagi) (Fuzzy Petri Nets), [Lip'93], [Zim'93], [PrP'97];

d) reţelele Petri temporizate deterministe, [Han'92], [Clu'92];

e) reţelele Petri temporizate stochastice, [Bke'96], [Clu'92];

f) reţelele Petri hibride, [LeAş], LeA'96];

g) reţelele Petri orientate pe obiecte (object Petri nets), [Lak'91], [Lak'93], [Lak'95a], [Lak'95b], [Lak'95c], [Lak'96].

Reţelele Petri s-au dovedit deosebit de utile în cele mai diverse domenii din tehnica automatizărilor industriale:

(i) în proiectarea sistemelor de automatizare industrială, [Sch'91a];

(ii) în alegerea variantei optime pentru sistemul de conducere, [SES'94];

(iii) în specificarea grafică, intuitivă, fundamentată matematic, a sistemelor de automatizare, [Rei'91], [Win'86a], [Win'86b];

(iv) în dezvoltarea de programe pentru automate programabile, [Asp'93]' [JLB'95];

(v) în analiza proprietăţilor dinamice ale sistemelor în timp real, [FMD'94];

(vi) în modelarea schemelor de protecţie din sistemele de distribuţie a energiei electrice, [JKL'92];

(vii) în modelarea calitativă a sistemelor dinamice continue, [Lun'92], [Llu'96], [Lun'93a], [Lun'93b], [Lun'94], [Lun'95], [Lun'96].

2.3 Concluzii

1. Automatizările industriale, ca domeniu tehnic multidisciplinar, trebuie să determine cele mai adecvate soluţii, metode, procedee şi instrumente care, aplicate corect, asigură condiţii optime de rentabilitate şi eficienţă la realizarea respectiv exploatarea unui sistem tehnic.

Automatizarea industrială se confruntă atât cu complexitatea sistemului ce se realizează cât şi cu diversitatea punctelor de vedere asupra acestuia.

La confruntarea cu sisteme mari, complexe se încearcă descoperirea structurii lor (analiză) cu scopul de a exploata caracteristicile acestei structuri în definirea unei căi de a controla sistemul (sinteză).

Sistemele mari, complexe nu pot fi stăpânite fără o viziune sistemică asupra lor. Această viziune sistemică trebuie să se bazeze pe axiomele-sistem prezentate în secţiunea 2.1.2. Prin conţinut şi formă de prezentare secţiunea 2.1.2. reprezintă un element de originalitate, autorul, în bibliografia la care a avut acces, găsind doar elemente disparate ale acestei tematici.

2. Studiul sistemelor reale, naturale sau tehnice, se realizează pe baza unui model. Modelul este o imagine idealizată şi esenţializată a fenomenelor reale elaborată prin sistematizarea rezultatelor măsurilor şi cunoaşterea legilor generale ale naturii.

Datorită punctelor de vedere diferite, egal îndreptăţite, asupra sistemului modelat este necesară elaborarea de modele diferite, cu caracteristici diferite. Aceste modele se deosebesc în primul rând prin principiul pe baza căreia se realizează structurarea sistemului respectiv a modelului elaborat. Autorul propune trei astfel de principii de structurare:

a) principiul decompoziţiei respectiv modelarea orientată pe obiecte, prin care sistemele şi subsistemele se divizează în părţi componente (secţiuea 2.2.2);

b) principiul abstractizării funcţionale, prin care se realizează structurarea pe nivele ierarhice a sistemului (secţiunea 2.2.3);

c) principiul transformării sau modelarea orientată pe evenimente, prin care se descriu transformările de stare (secţiunea 2.2.4).

Cu toate că multe elemente ale acestor principii s-au regăsit în bibliografia accesibilă autorului, forma unitară şi conţinutul sintetic prezentat în acest capitol este o contribuţie originală a autorului.

3. Modelarea se realizează la nivel conceptual în sensul că modelul este o descriere formală a tuturor aspectelor relevante ale sistemului modelat făcând abstracţie de orice aspecte legate de implementare. Noţiunile de bază ale modelării conceptuale s-au definit în ISO 82. Având în vedere importanţa deosebită a noţiunii de "obiect" în modelarea conceptuală, autorul a considerat necesar să dea, în secţiunea 2.2.1.1. o definiţie mai precisă a acestei noţiuni, faţă de ISO 82.

4. Pe baza relaţiilor de bază utilizate în modelul semantic orientat pe obiecte, autorul defineşte, drept exemple: (i) modelul informaţional al unui reactor cu agitator, (ii) o reţea de clase de operaţii tehnologice elementare, (iii) ierarhia de clase a unei reţete de producţie.

3. Reţele Petri

Acest capitol prezintă, în prima parte, noţiunile fundamentale legate de modelarea discretă, orientată pe evenimente a sistemlor distribuite. Se prezintă rolul relaţiilor de ordonare parţiala care stau şi la baza teoriei reţelelor Petri. În partea a doua a capitolului se prezintă, într-o forma inginerească adaptată nevoilor modelării sistemelor de automatizare, clasa reţelelor Petri Condiţii/Evenimente şi Locaţie/Tranziţie.

3.1 Structuri de evenimente. Reţele

3.1.1 Cauzalitate şi timp

3.1.1.1 Efecte relativiste în sisteme distribuite

În modelarea sistemelor tehnice se acceptă premisa valabilităţii principiului general al cauzalităţii conform căreia relaţia cauză→efect este o relaţie binară unidirecţională în sensul că nici un efect nu poate influenţa cauza care l-a produs (l-a determinat). Se acceptă prezumpţia că relaţia cauză→efect este şi stabilă în sensul că aceeaşi cauză va produce întotdeauna acelaş efect. Ca urmare, orice tehnică de modelare s-ar adopta aceasta trebuie să trateze într-un fel sau altul relaţia dintre **cauzalitate şi timp.** Această relaţie este în mod fundamental determinată de considerarea sau neconsiderarea efectelor relativiste. Efectele relativiste trebuie considerate în situaţia în care caracterul finit al vitezei de propagare al interacţiunilor (sau al unor semnale) nu poate fi trecut cu vederea.

În cazul sistemelor de prelucrare a informaţici, efectele relativiste trebuie considerate dacă viteza de prelucrare în diversele componente este de acelaş ordin de mărime cu viteza de propagare a semnalelor între aceste componente. Aceasta este cazul, de exemplu:

(1) la proiectarea cu circuite logice dacă viteza de propagare a semnalelor prin porţile logice individuale este comparabilă cu viteza de propagare a semnalelor între porţi;

(2) la sistemele distribuite, bazate întotdeauna pe un anumit sistem de comunicaţii, dacă viteza de transport a mesajelor, determinată de canalul de comunicaţie şi sistemele hardware şi software din nodurile de comunicaţie, este comparabilă cu viteza de prelucrare a acestor mesaje de către programele de aplicaţie.

În cazul în care efectele relativiste nu pot fi neglijate, nu poate fi acceptat un model care presupune o ordonare temporală totală a evenimentelor, aşa cum se postulează în fizica clasică. Poziţia temporală relativă a fenomemnelor pe o axă temporală a unui anumit observator dat este determinată de relaţiile cauzale dintre acestea şi este subiectivă în sensul că este specifică acelui observator.

Conform teoriei relativităţii viteza luminii reprezintă viteza maximă de propagare a interacţiunilor. Cu ajutorul spaţiului cvadridimensional, Minkowski a dat teoriei relativităţii o interpretare geometrică extrem de elegantă [Vas70]. În raţionamentele sale Minkowski s-a bazat pe noţiunea de *eveniment*, care se defineşte, în cadrul relativităţii, ca un eveniment fizic observabil care este determinat de 3+1 coordonate - trei coordonate

spațiale, notate x, y respectiv z iar a patra coordonată se referă la timp, notat în continuare prin t.

Ansamblul evenimentelor determină un continuum cvadridimensional numit **univers.** Evenimentele sunt numite **puncte de univers. Elementul de linie de univers** exprimat prin relația:

$$ds^2 = dx^2 + dy^2 + dz^2 - c^2dt^2$$

este invariant față de transformările lui Lorentz. Această invarianță, prin hiperconul pe care îl determină, induce o anumită structură a spțiului Minkowski, relativ la relațiile cauzale dintre evenimente.

În funcție de valoarea elementului de linie de univers se definesc următoarele domenii:

(1) $ds^2 < 0$ - domeniul **gen temporal**. În acest caz este posibilă o **relație cauzală**, obiectivă între două evenimente dintre care unul este **eveniment cauză** iar celălalt **eveniment efect**;

(2) $ds^2 = 0$ – determină asimptota hiprconului. Relația cauzală este posibilă doar cu condiția ca viteza de propagare a acțiunii să fie egală cu viteza luminii.

(3) $ds^2 > 0$ - domeniul **gen spațial**. În acest caz nu este posibilă o relație cauzală obiectivă între două evenimente (în caz contrar viteza interacțiunii ar trebui să fie mai mare decât viteza luminii.). Două evenimente din intervalul spațial formează un **cuplu spațial.**

Relația care definește elementul de linie de univers descrie un hipercon în spațiul cvadridimensional Minkowski. Figura 3-1 prezintă intervalele definite mai sus, printr-o proiecție pe o axă spațială. Proiecția hiperconului s-a reprezentat relativ la evenimetul A. S-au reprezentat doar asimptotele hipeconului deoarece acestea sunt elementele de delimitare ale celor două domenii.

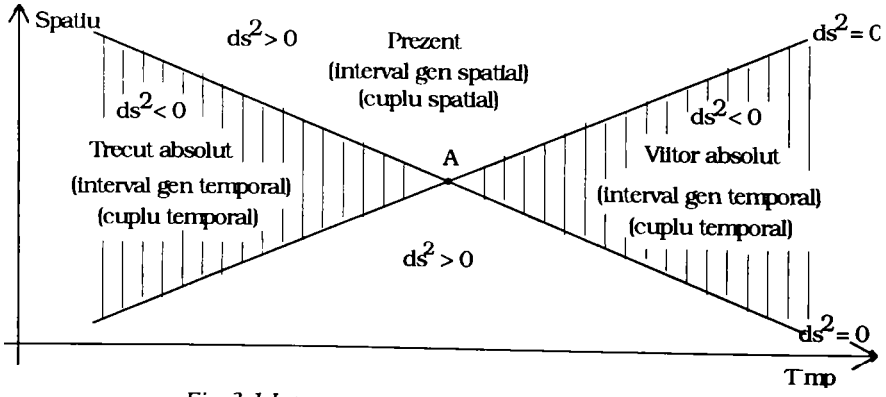


Fig. 3-1 Interpretarea geometrică a intervalului ds^2

Figura 3-2 prezintă relația cauzală și ordonarea evenimentelor în spațiul Minkowski.

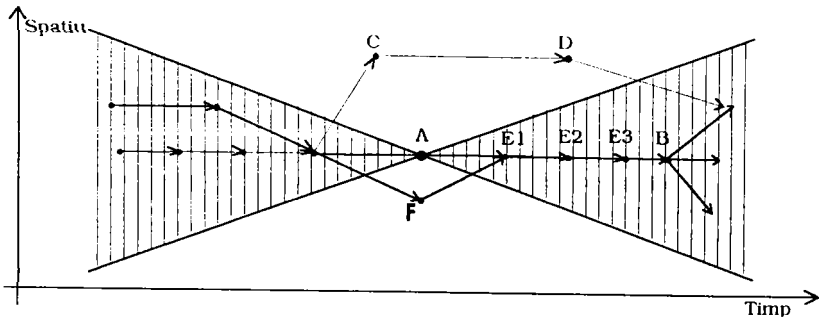


Fig. 3-2 Relația cauzală și ordonarea evenimentelor în spațiul Minkowski

Succesiunile de evenimente cauzal dependente s-au reprezentat prin traiectorii. Faptul că producerea evenimentului A este una dintre condițiile necesare producerii evenimentului B îl vom exprima prin "A este premisa lui B".

Dacă un eveniment A este premisa apariției (producerii) evenimentului B atunci aceasta din urmă trebuie să fie plasat în domeniul temporal "viitor absolut" doarece evenimentul A își poate exercita influența numai în acest domeniu. Cele două evenimente se găsesc în **relație temporală**. Spunem că evenimentul B este **cauzal dependent** de evenimentul A iar ordonarea temporală "*A înainte de B*" este **obiectivă** în sensul că este valabilă pentru toți observatorii posibili (nu poate exista vre-un observator pentru care ordinea temporală să se inverseze). Acest lucru se explică prin faptul că elementul de linie de univers ds^2 este invariant față de transformările lui Lorentz.

Evenimentele A și C din Figura 3-2 nu pot sta într-o ralație cauzală obiectivă. Din acest motiv aceste evenimente nu se ordonează, în mod natural, prin relații cauzale. Diverși observatori pot vedea "*A înainte de C*" sau "*C înainte de A*" sau "*C simultan cu* A". Ordonare evenimentelor din domeniul gen spațial nu poate fi decât **subiectivă** fiind dependentă de observator. Aspectul obiectiv relativ la aceste evenimente este neordonarea.

În Figura 3-2 apariția evenimentului E1 este condiționată atât de producerea evenimentului A cât și de producerea evenimentului F. Vom exprima acest lucru prin "A∧F este premisa lui E1".

Evenimentele A și F s-au produs simultan în puncte spațial distincte, prin urmare nu pot sta într-o relație cauzală. Evenimentele obiectiv simultane pot apare doar în același punct spațial în consecință nu pot fi distinse.

Întrucât toate situațiile de mai sus sunt plauzibile se impune următoarea concluzie:

> **Ipoteza unei ordonări totale, obiective (independente de observator) a evenimentelor exclude posibilitatea unei modelări realiste a sistemelor spațial distribuite.**

3.1.1.2 Relații dintre evenimente în spațiul Minkowski

Pentru a caracteriza interacțiunile dintre evenimente se definesc relațiile de cauzalitate "*cauzează*" și "*este premisa*", pe baza evenimentelor prezentate în Figura 3-2, astfel:

(1) "*A cauzează E1*" înseamnă că evenimentul A este cauza evenimentului E1.

(2) "*A este premisa lui B*" înseamnă că există o mulțime E1, ..., En astfel încât "*A cauzează E1, cauzează E2, ... ,cauzează En, cauzează B*".

Deasemenea, definim relațiile de ordonare temporală, "*pre*" și "*post*", bazate pe relațiile cauzale, astfel:

- Dacă relația "*A este premisa lui B*" este adevărată rezultă că ordinea temporală "*A înainte de B*" sau "*A pre B*" este obiectivă, la fel și "*B după A*" sau "*B post A*".

O mulțime de evenimente legate prin relația "*este premisa*", și care formează un lanț fără cicluri, se numește **ordonare cauzală**. Toate evenimentele X, pentru care este valabilă relația "*X este premisa lui A*" formează "trecutul absolut" al evenimentului A. Evenimentele pentru care este valabilă relația "*A este premisa lui X*" formează "viitorul absolut" al evenimentului A. Toate aceste evenimente prezintă o legătură temporală (gen temporal) cu A.

Mulțimea X a evenimentelor având o legătură spațială (gen spațial) cu A (inclusiv A) formează prezentul evenimentului A. Întrucât evenimentele mulțimii X nu pot sta într-o relație de cauzalitate cu evenimentul A se spune că evenimentele cu legătură spațială "sunt concurente", sau că "se găsesc în relație de concurență", notată prin "X *co* A".

Relația "*co*" nu poate crea vre-o ordonare, spre deosebire de relația "este premisa" care creează o ordonare parțială obiectivă. S-ar putea spune că relația "*co*" reprezintă neordonarea.

Relația "*co*" prezintă următoarele proprietăți:

i) este *reflexivă*, adică este valabilă: "A *co* A";

ii) este *simetrică*, adică este valabilă următoarea implicație: Dacă "A *co* C" atunci "C *co* A".

iii) este *netranzitivă*, adică: Dacă "A *co* D" și "D *co* B" nicidecum nu rezultă "A *co* B".

Este de remarcat deosebirea dintre relația "*co*", netranzitivă, și de simultaneitatea din fizica clasică care este tranzitivă.

Privitor la relațiile "pre" și "co" este evident că dacă este valabilă una dintre relațiile "*A este premisa lui X*" sau "*X este premisa lui A*" nu este valabilă relația "A *co* X" și reciproc, dacă este valabilă una dintre relațiile "A *co* X" sau "X *co* A" nu este valabilă relația "*A este premisa lui X*" sau "*X este premisa lui A*".

Având în vedere cele de mai sus se impune constatarea:

> **Deoarece modelarea poate fi executată doar pe baza obsevațiilor asupra lucrurilor (nu pe baza lucrurilor în sine) la baza acesteia trebuie să stea o relație netranzitivă dar reflexivă, așa cum este relația "*co*", împreună cu relațiile de ordonare parțială "pre" și "post"**

Modelarea pe baza relațiilor tranzitive (ca de exemplu simultaneitatea din fizica clasică) s-ar putea accepta doar în ipoteza unei viteze infinite de propagare a interacțiunilor, posibilitate infirmată de teoria relativității.

3.1.2 Structuri de evenimente

Pentru o caracterizare formală, abstractă a relațiilor cauzale dintre evenimentele ce apar în cursul evoluției unui sistem distribuit se definește o relație de ordine parțială pe mulțimea E a evenimentelor. Se va considera că evenimentele apar ca urmare a execuției unor acțiuni din sistemul distribuit. Mulțimea tuturor acțiunilor se va nota prin *A*. Legătura dintre acțiuni și evenimente se exprimă printr-o funcție de etichetare (labeling funcțion) notată prin λ.

Definiţia 3-1 (Mulţime parţial ordonată, etichetată)

Prin mulţime parţial ordonată, etichetată se înţelege tripletul (E, ≤, λ), alcătuit din:

- o mulţime de evenimente, E;
- o relaţie de ordine parţială pe mulţimea E, (≤)⊆ E×E;
- o funcţia de etichetare, λ: E→*A*.

Prin definiţie relaţia de ordine parţială (≤)⊆E×E este o relaţie *ireflexivă* şi *tranzitivă*. Aceste două proprietăţi implică *antisimetria*. Pentru reprezentarea unei mulţimi parţial ordonate, etichetate se adoptă următoarele convenţii:

- ε =(∅, ∅, ∅) este mulţimea parţial ordonată, etichetată vidă;
- mulţimea parţial ordonată, etichetată se reprezintă grafic. De exemplu ({e_a, e_b}, (≤), {(e_a, a), (e_b, b)})se reprezintă prin :

atunci când e_a şi e_b nu sunt în relaţie prin (≤) şi prin 

dacă e_a (≤) e_b.

Săgeata poate fi citită "cauzează" şi înseamnă că "e_a se produce înaintea lui e_b şi este totodată o premisă pentru producerea acestuia". Interpretare de mai sus justifică totodată şi proprietatea de ireflexivitate pe care o impunem acestor relaţii de ordine.

Exemplul 3-1

Drept exemplu se va considera următoarea mulţime parţial ordonată:

(E,(≤),λ)=({1,2,3,4,5,6}, (≤), {(e_1,1), (e_2,2), (e_3,3), (e_4,4), (e_5,5), (e_6,6)}.

Reprezentarea grafică a acestei mulţimi este dată în următoarea figură.

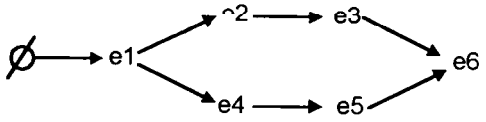


Fig. 3-3

În reprezentarea grafică din Figura 3-3 evenimentele e_2, e_3, e_4 şi e_5 sunt concurente şi prin urmare ordonarea temporală a acestora nu este obiectivă sau altfel exprimat acestea pot apare în orice combinaţie.

Definiţia 3-2

Definim relaţia *li* ⊆ E×E prin: e *li* e' ⇔ e (≤) e' ∨ e' (≤) e ∨ e≡e'.

Definim relaţia *co* ⊆ E×E prin: e *co* e' ⇔ ¬(e (≤) e' ∨ e' (≤) e).

Dacă o mulţime parţial ordonată surprinde apariţiile evenimentelor într-un sistem distribuit, atunci relaţia *li* surprinde apariţiile secvenţiale iar relaţia *co* apariţii concurente de evenimente.

În Exemplul 3-1:

li = {(e_1,e_2), (e_2,e_5), (e_5,e_6), (e_1,e_3), (e_3,e_4), (e_4,e_6)}

co = {(e_2,e_3), (e_2,e_4), (e_5,e_3), (e_5,e_4)}

Definiția 3-3.

Fie E o mulțime arbitrară. Se numește *relație de similaritate* pe E orice relație binară pe E, reflexivă ((e,e)∈E, ∀e∈E) și simetrică ((e,e')∈E⇒(e',e)∈E).

Este ușor de verificat că are loc:

Propoziția 3-1.

Fie E o mulțime parțial ordonată și e, e'∈E. Atunci:

(1) e *li* e' ∨ e *co* e';

(2) (e *li* e' ∧ e *co* e') ⇔ e = e' ;

(3) *li* și *co* sunt relații de similaritate.

Definiția 3-4

Fie E o mulțime arbitrară și ρ o relație de similaritate pe E. Numim *regiune* a relației ρ orice submulțime F⊆E pentru care au loc relațiile:

(i) eρe', ∀e,e'∈F;

(ii) (∀e∈E)(e∉F⇒∃e'∈F:¬(eρe')) (maximalitate).

Prin cea de a doua condiție s-a impus condiția de maximalitate a submulțimii F pentru ca aceasta să conțină toate elementele mulțimii E care se găsesc în relația ρ.

Se observă imediat că o regiune nu poate avea o submulțime proprie o altă regiune și, în cazul în care ρ este o relație de echivalență (satisface în plus și proprietatea de tranzitivitate), regiunile coincid cu clasele de echivalență.

Relațiile de similaritate ρ peste mulțimea E pot fi reprezentate grafic prin grafuri neorientate G=(E, U), unde U={(e,e') | e ≠ e' ∧ e ρ e' }.

În exemplul 3-1 regiunile relației *li* sunt următoarele:

$F_1 = \{(e_1,e_2), (e_2,e_5), (e_5,e_6)\}$

$F_2 = \{(e_1,e_3), (e_2,e_4), (e_4,e_6)\}$

iar regiunile relației *co* sunt:

$F_3 = \{(e_2,e_3)\}$, $F_4 = \{(e_2,e_4)\}$, $F_5 = \{(e_5,e_3)\}$, $F_2 = \{(e_5,e_4)\}$.

Definiția 3-5.

Fie E o mulțime parțial ordonată.

(1) Se numește *linie* (de univers) orice regiune a relației: *li* ⊆ E×E.

(2) Se numește *tăietură* orice regiune a relației: *co* ⊆ E×E.

Se observă că mulțimea E, parțial ordonată, etichetată din Exemplul 3-1 are două linii de univers F_1 și F_2 respectiv patru tăieturi F_3, F_4, F_5 și F_6.

Liniile reprezintă subprocese secvențiale maximale iar tăieturile observări globale la diferite momente de timp ale tuturor subproceselor secvențiale.

Propoziția 3-2

Fie E o mulţime parţial ordonată şi F⊆E.

(1) F este linie dacă şi numai dacă:

(a) $\forall e,e' \in F: e (\leq) e' \vee e'(\leq)e \vee e \equiv e'$;

(b) $\forall e \in E \setminus F, \exists e' \in F: \neg(e (\leq)e' \vee e' (\leq)e)$.

(2) F este tăietură dacă şi numai dacă:

(a) $\forall e,e' \in F: \neg(e (\leq) e' \vee e (\leq) e')$;

(b) $\forall e \in E \setminus F, \exists e' \in F: e (\leq) e' \vee e' (\leq) e$.

Demonstraţie. (1) cu cele două condiţii (a) şi (b) este echivalentă cu faptul că F este regiune pentru relaţia *li*. Analog, (2) cu cele două condiţii (a) şi (b) este echivalentă cu faptul că F este regiune pentru relaţia *co*. □

Definiţia 3-6

Fie E o mulţime parţial ordonată şi F,F'⊆E.

1. Spunem că E este mărginită dacă există $n \in N$ astfel ca pentru orice linie L a lui E are loc $|L| \leq n$, unde |L| reprezintă cardinalitate mulţimii L.
2. Spunem că F precede F', şi notăm F≤F', dacă pentru orice e∈F şi e'∈F' are loc e≤e' sau e *co* e'. Dacă F≤F' şi F≠F' atunci spunem că F precede strict F' şi notăm F<F'.

Notaţia 3-1

Fie E o mulţime parţial ordonată şi F⊆E. Notăm:

(a) $F^{-}=\{ e \in E \mid \{e\} \leq F \}$;

(b) $F^{+}=\{ e \in E \mid F \leq \{e\} \}$;

(c) $^{\circ}F=\{ e \in F \mid \forall e' \in F: e \; co \; e' \vee e \leq e' \}$

(d) $F^{\circ}=\{ e \in F \mid \forall e' \in F: e \; co \; e' \vee e' \leq e \}$

Se remarcă faptul că °F (F°) reprezintă mulţimea elementelor minimale (maximale) ale mulţimii F.

Propoziţia 3-3

Dacă E este o mulţime parţial ordonată mărginită atunci °E şi E° sunt tăieturi.

Demonstraţie. Fie $a,b \in {}^{\circ}E$. Conform definiţiei acestei mulţimi avem *a co b*. Rămâne să arătăm că °E este maximală. Fie $c \notin {}^{\circ}E$ şi L o linie ce conţine *c*. Deoarece L este finită (E este mărginită) urmează că $L \cap {}^{\circ}E \neq 0$ şi deci există $d \in L \cap {}^{\circ}E$. În plus, $d \leq c$ sau $c \leq d$ deoarece *d* este element minimal. Dar aceasta ne arată că °E este tăietură. Un raţionament similar se aplică şi pentru E° . □

Propoziţia 3-4

Fie E o mulţime parţial ordonată, L o linie şi D o tăietură a ei. Atunci $|L \cap D| \leq 1$.

Demonstraţie. Presupunem că L şi D au mai mult de un element în comun. Fie *e* şi *e'* două dintre acestea, $e,e' \in L \cap D$, atunci e *li* e' şi e *co* e' $\Rightarrow$ e=e'. □

Definiția 3-7.

O mulțime parțial ordonată este numită K-densă dacă orice linie a ei intersectează orice tăietură a ei.

Proprietate de K-densitate arată că orice subproces secvențial este observat în orice moment.

Fie $li(e)$ o linie a mulțimii parțial ordonate E astfel încât $e \in E$ și $e \in li(e)$. Reuniunea tuturor $li(e)$, $Li(e) = \bigcup_{e \in li(e)} li(e)$ formează *domeniul gen temporal* al evenimentului e. Acest domeniu corespunde viitorului absolut al evenimentului e în reprezentarea prin spațiul Minkowski.

Fie $co(e)$ o tăietură a mulțimii parțial ordonate E astfel încât $e \in E$ și $e \in co(e)$. Reuniunea tuturor tăieturilor $co(e)$, $Co(e) = \bigcup_{e \in co(e)} co(e)$ formează *prezentul evenimentului* e și corepunde aceleiași noțiuni al spațiului Minkowski. Oricărei observări a evenimentului e îi corespunde o tăietură $co(e)$.

3.1.3 Condiții și evenimente

O structură cauzală conține cu siguranță evenimente fenomene fizice observabile care reprezintă *evenimente*. Noțiunea de eveniment este în legătură cu noțiunea de schimbare în sensul că un eveniment poate fi observat doar dacă se produce și o schimbare. Cea ce este supus acțiunii unui eveniment, cea ce se schimbă sunt ***condițiile***. Noțiunile de eveniment și condiție sunt principial diferite deoarece un eveniment nu poate să se schimbe (un eveniment are loc, se produce) iar o condiție nu poate avea loc (o condiție este valabilă sau nu).

Condițiile caracterizează starea sistemului în care se produc evenimentele. Orice eveniment se poate produce doar într-o anumită configurație de condiții.

În modelarea discretă, orientată pe evenimente condițiile și evenimentele se consideră a fi atomice în sensul că un eveniment ori se produce ori nu se produce (nu există evenimente parțial produse) respectiv o condiție este ori validă ori invalidă (nu există condiții parțial valide).

Condițiile și evenimentele se găsesc în următoarea relație:

(1) O condiție se caracterizează prin acele evenimente care determină dacă aceasta este validă sau nu este validă;

(2) Un eveniment se caracterizează prin condițiile care înaintea producerii acestuia erau valabile iar după nu mai sunt valabile precum și prin condițiile care înainte de producerea evenimentului nu erau valabile iar după devin valabile.

Evenimentele exprimă modul în care condițiile interacționează între ele. Aceasta înseamnă ca, **în momentul** producerii evenimentului, condițiile care-l caracterizează trebuie să se găsească în **puncte spațiale vecine**.

In reprezentarea grafică a rețelelor condițiile se reprezintă prin cercuri iar evenimentele prin dreptunghiuri. O săgeată orientată de la o condiție la un eveniment semnifică faptul că apariția evenimentului este "pregătită" de condiția dată. O săgeată orientată de la un eveniment la o condiție semnifică faptul că aceea condiție este o condiție posterioară evenimentului, produsă de către evenimentul dat.

În Figura 3-4 se prezintă exemplul unui proces de transport (a unor obiecte sau a unor date) între două localități.

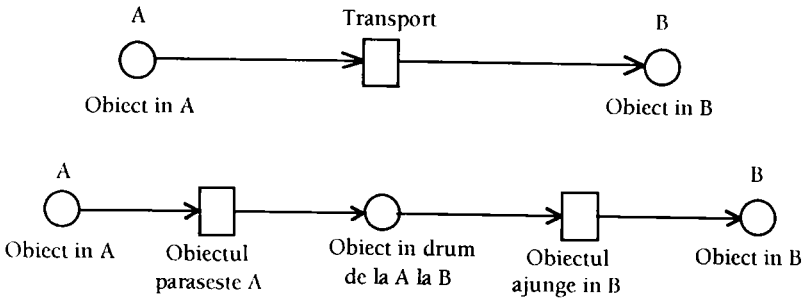


Fig. 3-4 Modelarea unui proces de transport

Prin modelarea procesului de transport ca **un** eveniment am presupus că starea [*(obiectul este în A)* ∧ *(obiectul nu este în B)*] trece **obectiv instantaneu** (eveniment atomic) în starea caracterizată prin [*(obiectul nu este în A)* ∧ *(obiecul este în B)*]. Cu alte cuvinte schimbarea de stare este instantanee. Acest lucru înseamnă, având în vedere conceptul relativist al cauzalității, că între cele două condiții (localități) nu poate exista nici un punct spațial. Cu alte cuvinte, la acest nivel de abstractizare, cele două localități se consideră a fi în puncte spațiale vecine. Numai în acest caz evenimentul corespunzător acțiunii de transport se poate produce instantaneu.

Schimbarea de stare este instantanee dar starea [*(obiectul este în A)* ∧ *(obiectul nu este în B)*] durează, are o durată de viață finită deoarece acțiunea de transport se realizează prin interacțiuni cu viteza de propagare finită. Durata stării [*(obiectul este în A)* ∧ *(obiectul nu este în B)*] este echivalentă cu durata procesului de transport.

La un alt nivel de abstractizare procesul de transport se poate explicita, de exemplu, prin introducerea unei noi condiții cu semnificația "*obiect în drum de la A la B*" încadrat între două evenimente conform Figurii 3-3.

O afirmație de genul "două puncte spațiale distincte sunt vecine, între ele nu poate exista nici un alt punct spațial" este specifică abordării (modelării) discrete a sistemelor. În cazul abordări continue punctele spațiale și temporale sunt proiectate pe o axă a numerelor reale având între oricare două puncte o infinitate de alte puncte care pot sau trebuie să fie integrate în model.

Modelare discretă ia în considerare doar obiectele observate sau observabile. În acest fel granularitatea (rezoluția) modelării este determinată de posibilitățile sau necesitățile de observare ale sistemelor. La nivelul de rezoluție maximă între două evenimente vecine observate nu se va găsi nici un alt eveniment dar se va găsi o condiție și reciproc între două condiții vecine nu se va găsi o altă condiție ci un eveniment. Cu alte cuvinte, în cazul unui model discret, orientat pe evenimente, evenimentele și condițiile alternează.

Oricare ar fi nivelul de abstractizare (de granularitate al modelului) între două evenimente întotdeauna se va găsi o condiţie iar între două condiţii întotdeauna se va găsi un eveniment.

Cu ajutorul noţiunilor de condiţii şi evenimente, în accepţiunea celor de mai sus, pot fi modelate sistemele discrete. Modelele se prezintă sub forma unui graf cu două tipuri de noduri. Arcele orientate reprezintă sensul interacţiunilor. Deoarece modelele astfel construite se prezintă întotdeauna sub forma unei reţele bipartite, orientate s-a încetăţenit noţiunea de modelare prin reţele iar ca disciplină ştiinţifică noţiunea de teoria reţelelor.

Cele de mai sus impun următoarea concluzie:

Teoria reţelelor bipartite orientate este o teorie a modelării discrete, relativiste. Elementele reţelei sunt condiţii şi evenimente. Condiţiile sunt încadrate doar prin evenimente iar evenimentele doar prin condiţii.

În practică s-au încetăţenit diverse tipuri de reţele bipartite orientate. De cele mai multe ori aceste reţele sunt referite prin denumirea de reţea Petri după numele celui care a fundamentat teoria reţelelor bipartite, orientate. Clasa reţelelor Petri reprezintă un instrument deosebit de eficient în modelarea sistemelor distribuite în care concurenţa, comunicarea şi paralelismul ocupă un loc central.

3.1.4 Reţele Petri.

Definiţia diferitelor tipuri de rerţele bipartite, orientate se bazează pe noţiunea de reţea Petri definită după cum urmează:

Definiţia 3-8 (Reţea Petri)

Se numeşte reţea Petri orice triplet N=(P,T;F), unde:

(1) P şi T sunt două mulţimi arbitrare ce satisfac condiţiile: $P \cap T = \emptyset$ şi $P \cup T \neq \emptyset$;

(2) $F \subseteq P \times T \cup T \times P$ este o relaţie binară numită *relaţia de flux* a reţelei.

Elementele mulţimii P se numesc P-elemente iar elementele mulţimii T se numesc T-elemente. P-elementele unei reţele N=(P,T;F) vor desemna stări atomice, iar T-elementele tranziţii atomice. Relaţia de flux F surprinde legătura (interrelaţia) dintre stări şi tranziţii. În reprezentarea grafică P-elementele vor fi reprezentate prin cercuri iar T-elementele prin pătrate sau dreptunghiuri. Relaţia de flux va fi reprezentată prin arce orientate de la x la y ori de câte ori $(x,y) \in F$ respectiv de la y la x ori de câte ori $(y,x) \in F$. Reţelele Petri sunt grafuri bipartite. Pornind de la graful bipartit al reţelei Petri putem reface reţeaua Petri conform Definiţiei 3-8.

Notaţia 3-2

Fie N=(P,T;F) o reţea Petri :

(1) În lucrul cu mai multe reţele, distincţia între elementele acestora se va realiza prin indexare inferioară cu simbolul asociat reţelei, astfel: $P=P_N$, $T=T_N$, $F=F_N$. Se va nota de asemenea $P_N \cup T_N = X_N$.

(2) Fie $x \in X_N$ şi $X \subseteq X_N$. Notăm:

(a) ${}^\bullet x = \{y \in X_N \mid (y,x) \in F\}$ - *premulţimea* elementului $x \in X_N$;

(b) $x^\bullet = \{y \in X_N \mid (x,y) \in F\}$ - *postmulţimea* elementului $x \in X_N$;

(c) ${}^\bullet X = \cup\{{}^\bullet x \mid x \in X\}$ - *premulțimea* submulțimii $X \in X_N$;

(d) $X^\bullet = \cup\{x^\bullet \mid x \in X\}$ - *postmulțimea* submulțimii $X \in X_N$.

Definiția 3-9

Fie N=(P,T;F) o rețea Petri.

(1) N se numește *finită* dacă X_N este mulțime finită.

(2) N se numește *pură* dacă pentru orice $x \in X_N$, ${}^\bullet x \cap x^\bullet = \varnothing$.

(3) N se numește *simplă* dacă $\forall x,y \in X_N$, ${}^\bullet x = {}^\bullet y \wedge x^\bullet = y^\bullet \Rightarrow x = y$.

(4) Un element $x \in X_N$ este *element izolat* dacă ${}^\bullet x \cup x^\bullet = \varnothing$.

Notația 3-3

(a) Vom nota prin |X| cardinalitatea mulțimii X.

(b) Fie N=(P,T;F) o rețea Petri. Mulțimea P-elementelor respectiv a T-elementelor le vom nota prin: $P=\{p_1,p_2, \ldots ,p_{|P|}\}$ respectiv $T=\{t_1,t_2, \ldots ,t_{|T|}\}$.

În cele ce urmează, având în vedere faptul că în cazul de față problematica rețelelor Petri este abordată în prisma aplicațiilor în domeniul sistemelor industriale, se vor considera doar rețele Petri finite.

Definiția 3-10

Fie $N_1=(P_1,T_1;F_1)$ și $N_2=(P_2,T_2;F_2)$ două rețele Petri.

(1) Vom spune că N_2 este subrețea a rețelei N_1 și notăm $N_2 \subseteq N_1$, dacă $P_2 \subseteq P_1$, $T_2 \subseteq T_1$ și $F_2=F_1 \cap ((P_2 \times T_2) \cup (T_2 \times P_2))$.

(2) Vom spune că N_2 este *duala* rețelei N_1 și notăm $N_2=\underline{N}_1$ dacă $S_2=T_1$, $T_2=S_1$ și $F_2=F_1^{-1}$.

3.1.5 Rețele de proces

Rețelele de proces se utilizează pentru descrierea comportării de tip proces a sistemelor orientate pe evenimente. În literatura de specialitate se folosește și terminologia de "rețea de apariții" sugerând faptul că această clasă de rețele surprinde aparițiile evenimentelor.

O rețea de proces reprezintă o structură cauzală formată din condiții si evenimente caracterizată prin faptul că fiecărei condiții i se alocă exact un eveniment care o realizează (validează) și exact un eveniment ce o invalidează. Faptul că o condiție este încadrată doar de două evenimente are următoarea explicație:

> **Rețelele de proces modelează desfășurarea reală a evenimentelor. Chiar dacă validitatea unei condiții este influențată de mai multe evenimente (de fapt de apariția acestora) validarea în sine este realizată în ultimă instanță doar de un anumit eveniment. Pentru invalidarea condiției este valabil același lucru.**

Evenimentele din rețelele de proces trebuie să fie legate la cel puțin la două precondiții și două postcondiții deoarece ele trebuie să descrie interacțiunea dintre condițiile situate în puncte puncte spațiale vecine precum și interacțiunile cu mediul care la rândul său condiționează desfășurarea procesului modelat respectiv care este condiționat de desfășurarea acestuia.

Definiția 3-11

O rețea K=(P,T;F) se numește *rețea de proces* dacă:

(1) $(\forall x,y \in P \cup T)\ (x F' y \Rightarrow \neg(y F' x))$ (nu există circuite închise);

(2) $\forall p \in P: |{}^{\bullet}p| \leq 1 \wedge |p^{\bullet}| \leq 1$ (nu există conflicte).

Exemplu: Reţeaua din Figura 3-5 este o reţea de proces deoarece nu prezintă circuite închise sau conflicte (fiecare locaţie are în pre- respectiv postmulţimea ei doar o singură tranziţie).

Definiţia 3-12

Fie K o reţea de proces. Se numeşte mulţime parţial ordonată indusă de K mulţimea $(P \cup T, (\leq))$, unde relaţia $(\leq)$ este dată prin:

$$x(\leq)y \Leftrightarrow x F' y.$$

Definiţia de mai sus este consistentă în sensul că proprietăţile reţelei de proces asigură că $(\leq)$ este în adevăr o relaţie de ordine parţială.

Definiţia 3-13

Fie K o reţea de proces.

O *tăietură* a mulţimii parţial ordonate indusă de K se numeşte:

- *caz* dacă este formată doar din P-elemente (specifică situaţia în care toate condiţiile tăieturii sunt valide).
- *tăietură temporală* dacă conţine şi un eveniment (specifică momentul în care s-au schimbat condiţiile în interacţiune cu acel eveniment).

O *linie* a mulţimii parţial ordonate indusă de K se numeşte :

- *secvenţă temporală* dacă evenimentele reprezintă schimbări (tranziţii) iar condiţiile momente de timp specifice.
- *proces secvenţial* dacă precondiţiile sunt interpretate ca premise iar postcondiţiile ca rezultate ale acţiunilor procesului.

Se poate demonstra următoarea teoremă [JŢi95]:

Teorema 3-1

Orice reţea de proces nevidă şi mărginită este K-densă adică orice linie a ei intersectează orice tăietură a ei.

Interacţiunea mai multor condiţii printr-un eveniment poate fi interpretată ca o sincronizare aşa cum se prezintă în exemplul din Figura 3-5. Alegem o linie a reţelei de proces şi o interpretăm ca un proces secvenţial (derularea unui proces secvenţial). Evenimentele liniei reprezintă operaţiile (tehnologice) executate în cursul derulării procesului. Condiţiilor le corespund, de exemplu, variabile din proces care sunt modificate (transformate) pas cu pas.

Precondiţiile unui eveniment din afara liniei alese sunt premisele externe ale apariţiei evenimentului dat. Postcondiţiile aceluiaşi eveniment sunt rezultate produse prin apariţia acestuia. Conform celor de mai sus un eveniment trebuie să aştepte validarea tuturor precondiţiilor pentru a se putea produce. Tocmai această posibilitate de sincronizare permite controlul procesului.

Pe baza celor de mai sus se impune următoarea concluzie:

Sincronizarea **este un concept de bază în descrierea sistemelor dinamice cu stări discrete, pilotate de evenimente.**

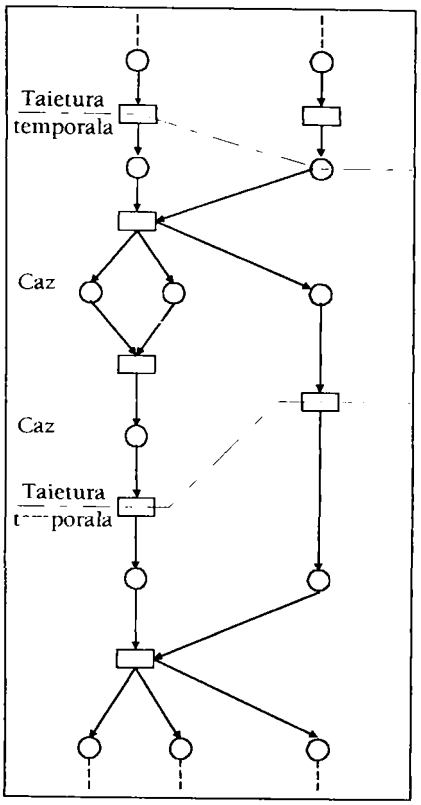


Fig. 3-5 Reţea de proces

În cazul în care o reţea de proces conţine evenimente cu o singură precondiţie aceasta nu modelează interacţiunea acelui eveniment cu mediul său, iar în cazul lipsei postcondiţiei nu se iau în considerare efectele procesului asupra mediului său. Un astfel de model nu permite analiza performanţelor sistemului modelat deoarece s-a renunţat în mod explicit la posibilitatea de sincronizare a procesului cu mediul său.

Analizând liniile unei reţele de proces modelatorul poate să-şi facă o idee despre *componentele* sistemului şi despre procesele de prelucrare (transformare) realizate de procesul dat. Pot fi conturate posibilităţile de modularizare respectiv poate fi detectat pericolul unor blocaje.

3.1.6 Reţele sistem

Reţelele de proces nu sunt apte pentru a descrie **comportamentul** sistemelor deoarece în ele nu pot apare alternative, conflicte sau cicluri. Reţelele sistem permit descrierea acestor situaţii.

În literatura de specialitate sunt definite mai multe clase de reţele sistem care se deosebesc prin complexitate şi putere de modelare [CPI'88], [Sei'92]. Acest capitol prezintă o sinteză despre reţelele Petri **Condiţii/Evenimente** şi **Locaţii/Tranziţii** bazată pe lucrările [Abe'90], [JŢi'95], [PĂS'97],[Sei'92], concepută pentru necesităţile prezentei lucrări.

Modelarea sistemelor industriale prin reţele Petri presupune parcurgerea a cel puţin două etape:

E1: stabilirea structurii statice a sistemului;

E2: stabilirea regulei de comportare dinamică a sistemului.

Prima etapă are ca obiectiv principal stabilirea componentelor pasive (stări, stări atomice), stabilirea componentelor active (tranziţii, tranziţii atomice) şi stabilirea interrelaţiilor dintre acestea (dependenţe cauzale). P-elementele modelează componente pasive, T-elementele modelează componente active, iar F-elementele interrelaţiile dintre ele.

Stabilirea regulei de comportare dinamică a sistemului este în strânsă legătură cu rezolvarea primei etape şi cu modul de definire a noţiunii de stare a sistemului. Se vor considera următoarele două cazuri:

(1) Dacă presupunem că P-elementele reprezintă *condiţii atomice* necesare producerii anumitor evenimente, condiţii ce pot fi îndeplinite sau neândeplinite, iar T-elementele reprezintă *acţiuni atomice* care, pentru a se produce, necesită îndeplinirea anumitor condiţii, atunci putem să spunem că starea sistemului este dată de mulţimea tuturor condiţiilor îndeplinite la un moment dat. Adică, o stare a sistemului este o submulţime $c \subseteq P$, $c=\{p \mid p$ condiţie îndeplinită$\}$. Uzual, aceste stări se numesc *cazuri* iar reţelele Petri, reţele Condiţii/Evenimente (abreviat, reţele C/E).

(2) Dacă presupunem că P-elementele reprezintă *locaţii* (elemente de acumulare, locaţii de memorie) sau *variabile* capabile să reţină anumite structuri de date, iar T-elementele reprezintă *acţiuni* (tranziţii, operaţii cu structuri de date), atunci putem spune că starea sistemului este dată de valoarea variabilelor (locaţiilor) la un moment dat. Deci, o stare în acest caz este o aplicaţie de la P la o mulţime de tipuri de date utilizate ca mai sus. Uzual, aceste stări se numesc *marcaje* iar reţelele Petri, *reţele Locaţie/Tranziţie* (abreviat reţele P/T).

3.2 Reţele Condiţii/Evenimente

3.2.1 Definiţii.

În continuare, reţelele Condiţii/Evenimente, abreviat reţele C/E, vor fi desemnate prin triplete (B,E;F). Elementele mulţimii B vor fi numite *condiţii* iar elementele mulţimii E vor fi numite *evenimente*. Dacă N=(B,E;F) este o reţea C/E şi $e\in E$, atunci elementele mulţimii ${}^\bullet e$ vor fi numite *precondiţii ale lui e* iar cele ale mulţimii $e^\bullet$ *poscondiţii ale lui e.*

Definiţia 3-14

Fie N=(B,E;F) o reţe Petri.

(1) Se numeşte *caz* al reţelei N orice submulţime $c\subseteq B$.

(2) Spunem că un eveniment $e\in E$ este *posibil (concesionat)* la cazul c, şi notăm $c[e\rangle_N$, dacă: ${}^\bullet e\subseteq c \wedge e^\bullet\cap c=\varnothing$.

(3) Dacă evenimentul e este posibil la cazul c atunci $c'=(c-{}^\bullet e)\cup e^\bullet$ spunem că este produs prin apariţia evenimentului e la cazul c. Notăm aceasta prin $c[e\rangle_N c'$. Acestă ultimă notaţie va fi simplificată la $[\bullet\rangle$, adică se omite scrierea evenimentului şi reţelei în care acesta se produce, ori de câte ori nu este pericol de confuzie.

Un caz c al unei reţele C/E surprinde ideea de mulţime a tuturor condiţiilor simultan satisfăcute (valabile) "de condiţii îndeplinite la un momemnt dat". Într-o reţea C/E cazurile vor fi reprezentate desenând câte un punct în cercurile corespunzătoare condiţiilor cazului respectiv, adică condiţiile îndeplinite se vor marca prin câte un punct. Schimbarea cazului prin producerea unui eveniment se face printr-o "deplasare" a punctelor în reţea.

Dacă e_1 şi e_2 sunt evenimente posibile la un caz c al reţelei N, şi în plus aceste evenimente au ori o precondiţie ori o postcondiţie comună, atunci conform Definiţiei 3-14 aplicabiliatea lui e_1 la cazul c exclude aplicabilitatea lui e_2 la cazul rezultat prin execuţia evenimentului e_1, sau invers. În cazul în care cele două evenimente nu au nici precondiţii şi nici poscondiţii comune atunci ele pot fi aplicate în ordine arbitrară.

Definiţia 3-15

Fie N=(B,E;F) o reţea C/E.

(1) O mulţime $G\subseteq E$ se numeşte *detaşată* dacă $\forall e_1,e_2\in G$, $e_1^\bullet\cap e_2^\bullet = \varnothing \wedge {}^\bullet e_1\cap {}^\bullet e_2 = \varnothing$.

(2) Fie c şi c' două cazuri şi G o mulţime detaşată. Mulţimea G se numeşte *pas* de la c la c', şi notăm $c[G\rangle c'$, dacă fiecare eveniment e din G este posibil (concesionabil) la cazul c şi $c'=(c-{}^\bullet G)\cup G^\bullet$.

Dacă $G=\{e\}$ este pas de la c la c' atunci notaţia $c[\{e\}\rangle c'$ se simplifică la $c[e\rangle c'$. Această notaţie se citeşte astfel: "Prin execuţia evenimentului e la cazul c se realizează cazul c'".

Notaţia 3-4

Fie N=(B,E;F) o reţea C/E. Notăm prin r_N relaţia binară $r_N\subseteq P(B)\times P(B)$ dată prin:

$c_1 r_N c_2 \Leftrightarrow \exists G \subseteq E$ pas: $c_1[G\rangle c_2$.

Prin $P(B)$ s-a notat mulțimea cazurilor adică mulțimea tuturor combinațiilor posibile realizate prin marcarea condițiilor.

Această relație va fi numită *relația de calcul înainte* (la dreapta) indusă de N. Relația r_N^{-1} va fi numită *relația de calcul înapoi* (la stânga) indusă de N.

În literatura de specialitate (de exemplu în [JȚi95]) se demonstrează următoarele două teoreme.

Teorema 3-2

Fie N o rețea C/E, G o mulțime detașată, c și c' două cazuri ale rețelei N. Mulțimea G este pas de la c la c' dacă și numai dacă ${}^\bullet G = c - c'$ și $G^\bullet = c' - c$.

Teorema 3-3

Fie N o rețea C/E și G un pas finit de la cazul c la cazul c'. Dacă $e_1, e_2, \ldots, e_n$ este o ordine arbitrară a elementelor pasului G, atunci există cazuri $c_1, c_2, \ldots, c_n$ astfel încât $c = c_0$, $c' = c_n$ și $c_{i-1}[e_i\rangle c_i$, $\forall 1 \le i \le n$.

Cu alte cuvinte, dacă un pas este finit atunci el poate fi realizat prin apariția evenimentelor care-l compun, în ordine arbitrară.

3.2.2 Situații fundamentale

Pe parcursul evoluției unui sistem real (distribuit) evenimentele acestuia se pot afla în diverse situații de dependență sau independență. Fie N o rețea C/E, e_1 și e_2 două evenimente distincte ale ei, și c un caz, principalele situații ce pot apare în legătură cu aceste elemente, numite situații fundamentale, sunt:

(1) **Secvența.** Spunem că e_1 și e_2 sunt în *secvență* la cazul c dacă

$$c[e_1\rangle c' \wedge \neg(c[e_2\rangle) \wedge c'[e_2\rangle.$$

Adică, e_1 este posibil la cazul c în timp ce e_2 nu este posibil, acesta din urmă devenind posibil abia după aplicarea lui e_1.

(2) **Conflictul.** Spunem că e_1 și e_2 sunt în *conflict* la cazul c dacă

$$c[e_1\rangle \wedge c[e_2\rangle \wedge \neg c[\{e_1, e_2\}\rangle$$

(3) **Concurența.** Spunem că e_1 și e_2 sunt *concurente* la cazul c dacă

$$c[\{e_1, e_2\}\rangle$$

adică $\{e_1, e_2\}$ este pas la cazul c.

(4) **Confuzia.** Numim *confuzie* a rețelei N orice triplet (c, e_1, e_2), unde:

(a) c este caz iar e_1 și e_2 sunt evenimente ale rețelei N;

(b) $c[\{e_1, e_2\}\rangle$;

(c) $\mathbf{cfl}(c_1, c) \neq \mathbf{cfl}(e_1, c_2)$, unde $c[\{e_2\}\rangle c'$.

În relaţia dc mai sus **cfl**(a,b) este definit prin:

Definiţia 3-16

Fie N=(B,E;F) o reţea C/E, e un eveniment şi c un caz al reţelei N astfel încât $c[e\rangle$. Notăm prin **cfl**(e,c) mulţimea tuturor evenimentelor care sunt în conflict cu e la cazul c. Adică,

$$\mathbf{cfl}(e,c)=\{e'\in E \mid c[e'\rangle \wedge \neg c[\{e, e'\}\rangle\}$$

Secvenţa conflictul, concurenţa şi confuzia sunt aspecte fundamentale ce apar în orice mod sau nivel de descriere al unui sistem distribuit. Cunoaşterea temeinică a acestor aspecte şi a interdependenţelor dintre ele constituie baza unei modelări corecte a sistemului prin reţele Petri.

3.2.3 Sisteme Condiţii/Evenimente

Precizarea unei reţele C/E, N=(B,E;F), ca model al unui sistem distribuit nu asigură o modelare completă a acestuia. Este necesară precizarea, într-un anumit mod, a mulţimii tuturor stărilor prin care a trecut sau va trece sistemul. Unei stări a sistemului îi corespunde în modelul reţea C/E un caz, deci o anumită distribuţie de puncte în reprezentarea grafică. Cunoaşterea stărilor sistemului revine deci la a preciza, pentru modelul C/E, a unei mulţimi C de cazuri. Reţeaua N împreună cu această mulţime trebuie să satisfacă următoarele proprietăţi:

(1) Reţeaua este simplă şi fără elemente izolate. Reţeaua nu conţine elemente distincte cu aceeaşi premulţime şi aceeaşi postmulţime.

(2) Mulţimea C este închisă la calculul înainte şi înapoi. Mulţimea C este închisă la calculul înainte şi înapoi dacă pentru orice $c\in C$ şi orice $c'\in B$ pasul realizat prin $r_N\cup r_N^{-1}$ produce cazul $c'\in C$.

$$(\forall c\in C)\ (\forall c'\in B)(c(r_N\cup r_N^{-1})c' \Rightarrow c'\in C).$$

(3) Relaţia $R_{N\Sigma}=(r_{N\Sigma}\cup r^{-1}_{N\Sigma})^*$ şi cazul iniţial $c_0\subseteq B$ generează o clasă de echivalenţă $C\subseteq P(B)$ numită *clasa cazurilor* sistemului Σ (a se vedea definiţia de mai jos).

(4) Mulţimea C este conexă. Pentru orice două cazuri din C există o secvenţă de paşi ce conduce de la un caz la celălalt.

(5) Reţeaua C/E şi mulţimea C trebuie să se găsească într-o relaţie de completitudine şi simplitate, în sensul:

(a) Completitudinea mulţimii C faţă de reţea: pentru orice eveniment e al reţelei există cel puţin un $c\in C$ la care este posibil e.

(b) Simplitatea reţelei faţă de mulţimea C: pentru orice condiţie b există cel puţin un caz c astfel încât $b\in c$.

Definiţia 3-17

Se numeşte *sistem C/E* orice 4-tuplu $\Sigma=(B,E;F,c_0)$, unde:

(1) (B,E;F) este o reţea simplă şi fără elemente izolate, numită *reţeaua suport a sistemului* Σ, şi notată N_Σ;

(2) $c_0 \subseteq B$ este un caz al rețelei N_Σ, numit *cazul inițial* al sistemului Σ;

(3) Pentru orice $e \in E$ există $c \in B$ astfel încât dacă $R_{N\Sigma}=(r_{N\Sigma} \cup r^{-1}_{N\Sigma})^*$, atunci $c_0 R_{N\Sigma} c$ și $c[e\rangle$.

Componentele sistemului Σ se vor nota prin: B_Σ, E_Σ, F_Σ, respectiv $C_\Sigma \subseteq B_\Sigma$.

În [JȚi95] cap. 2 pct.2.3 se demonstrează următoarea teoremă.

Teorema 3-4

Fie $\Sigma=(B, E; F, c_0)$ un sistem C/E. Atunci:

(1) $B \neq \emptyset$, $E \neq \emptyset$ și $F \neq \emptyset$;

(2) Pentru orice $c \in C$, $c' \subseteq B$ și $G \subseteq E$ are loc:

(a) $c[G\rangle c' \Rightarrow c' \in C$;

(b) $c'[G\rangle c \Rightarrow c' \in C$;

(3) Rețeaua suport (B, E; F) este pură;

(4) Pentru orice $b \in B$ există $c, c' \in C$ astfel încât $b \in c$ și $b \notin c'$.

3.2.4 Sisteme C/E ciclice și viabile

Definiția 3-18

Un sistem C/E Σ este numit *ciclic* dacă: $(\forall c_1, c_2 \in C_\Sigma)\ (c_1 r_\Sigma^* c_2)$.

Denumirea de sistem ciclic povine de la faptul că dacă $c_1 r_\Sigma^* c_2$ atunci $c_2 r_\Sigma^* c_1$, adică sistemul poate cicla în sens uzual.

Teorema 3-5

Fie Σ un sistem C/E ciclic și $c \in C_\Sigma$. Atunci $C_\Sigma=\{c' \mid c r_\Sigma^* c'\}$.

Definiția 3-19

Un sistem C/E Σ este numit *viabil* dacă:

$(\forall e \in E_\Sigma, \forall c \in C_\Sigma)\ (\exists c' \in C_\Sigma, \exists r_\Sigma^*$ a.î. $c r_\Sigma^* c' \wedge c'[e\rangle_\Sigma)$

Teorema 3-6

Orice sistem C/E ciclic este viabil.

Observație: Reciproca Teoremei 3-6 nu este adevărată adică, nu orice sistem viabil este și ciclic.

3.2.5 Descrierea vectorială a sistemelor C/E

În multe aplicații este utilă descrierea vectorială a sistemelor C/E. Sistemele C/E ce vor fi considerate în continuare vor fi finite. În plus, mulțimile $E=\{e_1, \ldots, e_{|E|}\}$ și $B=\{b_1, \ldots b_{|B|}\}$ vor

fi considerate total ordonate prin ordinea naturală pe indicii elementelor. Un sistem C/E poate fi descris prin procedee vectoriale, conform următoarelor definiții.

Definiția 3-20 (Descrierea vectorială a sistemelor C/E)

Un sistem C/E, N=(B,E,F,c_0) se descrie vectorial prin următoarele elemente:

(1) *vectorii de evenimente* $\mathbf{e}_j^+$ și $\mathbf{e}_j^-$ definiți pentru fiecare eveniment $e_j \in E$, $0<j\le|E|$, astfel

$$\mathbf{e}_j^- = \begin{cases} -1, & \text{daca } (b_i, e_j) \in F, \\ 0, & \text{daca } (b_i, e_j) \notin F, \end{cases} \qquad \forall b_i \in B,\ 0 < i \le |B|$$

$$\mathbf{e}_j^+ = \begin{cases} 1, & \text{daca } (e_j, b_i) \in F, \\ 0, & \text{daca } (e_j, b_i) \notin F, \end{cases} \qquad \forall b_i \in B,\ 0 < i \le |B|$$

(2) matricea de incidențe $\mathbf{N}=\mathbf{N}^-+\mathbf{N}^+$ unde:

$$\mathbf{N}^- = \left[e_1^-: \quad e_2^-: \quad e_{|E|}^-\right], \quad \mathbf{N}^+ = \left[e_1^+: \quad e_2^+: \quad e_{|E|}^+\right]$$

(3) *vectorul de marcaj* $\mathbf{m}$ specificat prin elementele $\mathbf{m}[i]=M(b_i)$ unde M este funcția $M:B\to\{0,1\}$ definită astfel:

$$M(b_i) = \begin{cases} 1, & \text{daca conditia b este valida} \\ 0, & \text{daca conditia b nu este valida} \end{cases} \qquad \forall b_i \in B,\ 0 < i \le |B|$$

(4) *vectorul marcjului inițial* $\mathbf{m}_0$ specificat prin elementele:

$$\mathbf{m}_0[i] = \begin{cases} 1, & \text{daca } b_i \in F, \\ 0, & \text{daca } b_i \notin F, \end{cases} \qquad \forall b_i \in B,\ 0 < i \le |B|$$

În continuare, descrierea vectorială a unui sistem C/E se va specifica prin N=(B,E,F,$\mathbf{m}_0$).

Regula de calcul într-un sistem C/E se formulează conform următoarei definiții.

Definiția 3-21 (Regula de calcul vectorială într-un sistem C/E)

Fie N=(B,E,F,$\mathbf{m}_0$) un sistem C/E, $\mathbf{m}$ un marcaj și e un eveniment al lui.

(1) *Regula de concesionare.* Spunem că evenimentul *e* are concesie (este posibil) la marcajul $\mathbf{m}$ în N și notăm $\mathbf{m}[e\rangle_N$ dacă are loc,

$$-\mathbf{e}_j^- \le \mathbf{m} \le \mathbf{1} - (\mathbf{e}_j^+ + \mathbf{e}_j^-)$$

(2) *Regula de calcul.* Spunem că marcajul $\mathbf{m}'$ este produsă prin apariția evenimentului e la marcajul $\mathbf{m}$ și notăm $\mathbf{m}[e\rangle_N\mathbf{m}'$ dacă $\mathbf{m}[e\rangle_N$. Marcajul $\mathbf{m}'$ este dată de realația:

$$\mathbf{m}' = \mathbf{m} + (\mathbf{e}_j^+ + \mathbf{e}_j^-)$$

3.3 Reţele Locaţie/Tranziţie

3.3.1 Definiţii.

În cazul reţelelor Locaţie/Tranziţie P-elementele sunt interpretate drept *locaţii* (variabile) capabile să reţină un număr natural iar T-elementele sunt interpretate drept *tranziţii* (operaţii asupra variabilelor). Clasa reţelelor Locaţie/Tranziţie reprezintă un bun compromis între puterea de modelare şi complexitatea respectiv disponibilitatea tehnicilor de analiză algoritmizate.

Definiţia 3-22

Se numeşte *reţea Locaţie/Tranziţie*, abreviat reţea P/T, orice 5-tuplu N=(P,T;F,K;W) unde:

(1) (P,T;F) este o reţea Petri;

(2) K este o funcţie K:P→*N*, numită *funcţia de capacitate* a reţelei N (K(p) este numită *capacitate* a locaţiei p∈P. Capacitatea unei locaţii poate să fie şi infinită.)

(3) W este o funcţie W:F→*N*, numită *funcţie ponderе* a reţelei N (W(*f*) este numită *ponderea* elementului *f*∈F).

Definiţia 3-23

Fie N=(P,T;F,K;W) o reţea P/T. Se numeşte *marcaj* al reţelei N orice aplicaţie M:P→*N* cu proprietatea M(p)≤K(p), ∀p∈P.

Notaţia 3-5

Fie N=(P,T;F,K;W) o reţea P/T.

(1) Se va nota prin N^P mulţimea tuturor marcajelor reţelei N. $N^P=\{M|M:P\to N \wedge \forall p\in P: M(p)\le K(p)\}$

(2) Pentru orice tranziţie *t* a reţelei N considerăm funcţiile t' , t⁻ :P→*N* şi Δt :P→Z definite prin:

(a) $t^-(p)=-W(p,t)$

(b) $t'(p)=W(p,t)$

(c) $\Delta t(p)= t'(p)+ t^-(p)$, ∀p∈P.

Se vor considera reţele P/T finite cu mulţimi finite de tranziţii şi locaţii, fără elemente izolate şi pentru care S≠∅ şi T≠∅. Proprietatea de finititudine a reţelei ne permite să identificăm marcajele $M\in N^P$ cu vectori |P|-dimensionali peste *N*, fixând o ordine totală pe P. Relaţiile şi operaţiile uzuale pe *N* le vom considera extinse asupra vectorilor din N^P pe componente.

Următoarea definiţie specifică *regula de tranziţie* într-o reţea P/T .

Definiţia 3-24

Fie N o reţea P/T, M un marcaj şi t o tranziţie a ei.

(i) *Regula de concesionare*. Spunem că tranziţia t *are concesie* (este posibilă) la marcajul M în reţeaua N, şi notăm $M[t\rangle_N$, dacă este adevărată următoarea relaţie:

(ii) $W(p,t) \le M(p) \wedge M(p)+W(t,p) \le K(p), \forall p \in P;$

(iii) *Regula de calcul.* Spunem că marcajul M' a reţelei N este *produsă* prin apariţia tranziţiei t la marcajul M, şi notăm $M[t\rangle_N M'$, dacă $M[t\rangle_N$. M' se determină prin următaorea relaţie:

(iv) $M'(p)= M(p)+\Delta t(p), \forall p \in P;$

Conform definiţiei de mai sus regula de tranziţie a reţelelor P/T este alcătuită din două reguli: ***regula de concesionare*** şi ***regula de calcul***. Prima vizează condiţiile în care o tranziţie t este posibilă la un marcaj M, iar a doua ne furnizează modul de calcul al noului martcaj în ipoteza în care tranziţia t este posibilă la M şi s-a executat.

Notaţia 3-6

Fie N o reţea Petri P/T şi $M \in \mathcal{N}^P$. Notăm prin T(N,M) mulţimea tuturor tranziţiilor reţelei N care sunt popsibile la M, adică: $T(N,M)=\{t \in T \mid M[t\rangle\}$. Această notaţie se va simplifica ori de câte ori N se subînţelege din context.

Prin regula de tranzţie am definit pentru orice tranziţie t a reţelei P/T o relaţie binară pe mulţimea N^P, notată $[t\rangle$, prin:

$$[t\rangle=\{(M,M') \mid M,M' \in \mathcal{N}^P \wedge M[t\rangle M'\}.$$

Dacă $t_1, \ldots t_n$ sunt tranziţii ale reţelei P/T N, putem realiza compunerea clasică a relaţiilor $[t_1\rangle, \ldots [t_n\rangle$, relaţia nou obţinută fiind notată $[t_1 \ldots t_n\rangle$. Definim în plus şi relaţia binară $[\lambda\rangle$ prin:

$$[\lambda\rangle=\{(M,M) \mid M \in \mathcal{N}^P\}.$$

Definiţia 3-25

Fie N o reţea P/T şi $M \in \mathcal{N}^P$.

(1) Spunem că $\sigma \in T^*$ este o *secvenţă de tranziţii* de la M (în N) dacă există $M' \in \mathcal{N}^P$ astfel încât $M[\sigma\rangle M'$.

(2) Spunem că marcajul $M' \in N^P$ este *accesibilă* de la M (în N) dacă există o secvenţă de tranziţii de la M (în N), σ, astfel încât $M[\sigma\rangle M'$.

Observaţie: Pentru orice marcaj M a unei reţele P/T N, λ este secvenţă de tranziţii de la M (în N).

Notaţia 3-7

Fie N o reţea P/T şi $M \in \mathcal{N}^P$.

(1) Mulţimea tuturor secvenţelor de tranziţie de la M (în N) o notăm prin TS(N,M). Este uşor de văzut că acestă mulţime are proprietatea prefixului, adică orice prefix al unei secvenţe de tranziţii este secvenţă de tranziţii.

(2) Mulţimea tuturor marcajelor accesibile de la M (în N) o notăm prin $R_N(M)$ sau $[M\rangle_N$.

3.3.2 Reţele Locaţie/Tranziţie marcate

Definiţia 3-26

Se numeşte reţea P/T marcată, abreviat mPTN, orice cuplu $\Lambda=(N,M_0)$ format dintr-o reţea P/T N, numită reţeaua suport a reţelei Λ, şi marcajul M_0 al reţelei N, numit *marcajul iniţial* al reţelei Λ.

În literatura de specialitate se foloseşte pe lângă terminologia de reţea P/T marcată şi terminologia de *sistem P/T*. În prezenta lucrare se va face distincţie între o reţea P/T marcată ca un abstract matematic, notat prin mPTN, şi un model de reţea P/T a unui sistem tehnic, ca o reţea mPTN interpretată (conform cap. 6). Aceasta din urmă se va numi *sistem P/T*.

Notaţia 3-8

(1) $[M_0\rangle$ denotă mulţimea tuturor marcajelor accesibile în Λ.

(2) $T_\Lambda(M)$ va denota mulţimea tuturor secvenţelor de tranziţii de la M în Λ.

Definiţia 3-27 (Reţea mPTN)

Se numeşte reţea P/T marcată, abreviat mPTN orice 6-tuplu $N=(P,T;F,K,W,M_0)$ unde:

(1) (P,T;F) este o reţea Petri;

(2) funcţia $K:P\rightarrow N^+$, este *funcţia de capacitate* care specifică capacităţile locaţiilor;

(3) funcţia $W:F\rightarrow N^+$, este *funcţia de ponderare* care specifică ponderea (multiplicitatea) fiecărui element al relaţiei de flux;

(4) funcţia $M_0:P\rightarrow N$, cu $M_0(p)\leq K(p)$ $\forall p\in P$, specifică marcajul iniţial al fiecărei locaţii.

Reţelele P/T marcate pot fi reprezentate prin grafuri bipartite, orientate, etichetate. Locaţiile şi tranziţiile mPTN apar ca noduri iar elementele relaţiei de flux ca arce orientate. Figura 3-6 prezintă un exemplu de mPTN care scoate în evidenţă elementele din Definiţia 3-27.

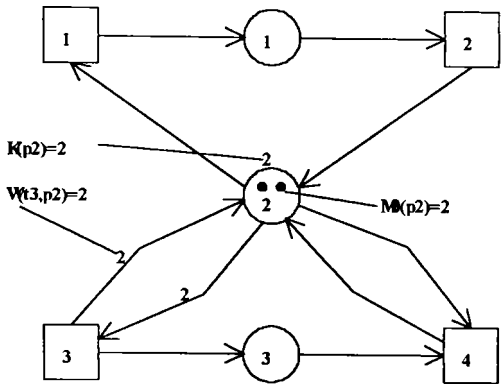


Fig. 3-6 Exemplu pentru reprezentarea grafică a unei mPTN

Locațiile p_i se simbolizează prin cercuri și sunt inscripționate cu valorile $K(p_i)$ ale capacităților acestora. Prin convenție, locațiile cu capacitatea unitară nu se inscripționează. Numărul maxim de puncte pe care pot să le conțină locațiile este determinat de valoarea capacității acestora conform Definiției 3-22.

Tranzițiile t_j se reprezintă prin dreptunghiuri. Acestea sunt elemente active deoarece determină, prin regula de tranziție, fluxul marcajelor în rețea. Tranzițiile pot fi interpretate ca evenimente locale ale căror apariție inițiază schimbările de stare.

Nodurile de rețea, de tip diferit, pot fi legate prin arce, (p_i, t_j) respectiv (t_j, p_i), reprezentate prin arce orientate, inscripționate prin valoarea ponderilor lor. Ponderea arcului determină numărul punctelor transferate prin arcul respectiv. Prin convenție, arcele cu ponderea unitară nu se inscripționează.

Se poate considera, fără restrângerea generalității, că o rețea Petri nu conține arce multiple între două noduri. Se acceptă arce multiple doar dacă sunt orientate în sens contrar (a se vedea în Figura 3-6 arcele (p2,t3) și (t3,p2) respectiv (p2,t4) și (t4,p2). Aceste arce formează o buclă autonomă de rețea, sau pe scurt buclă autonomă, conform următoarei definiții.

Definiția 3-28 (Buclă autonomă de rețea)

Fie N=(P,T;F) o rețea. Perechea $(p_i,t_j)\in PxT$ se numește buclă autonomă de rețea dacă locația p_i și tranziția t_j sunt legate prin arce în ambele sensuri adică dacă este valabilă relația

$$(p_i,t_j)\in F \wedge (t_j,p_i)\in F.$$

3.3.3 Tehnici de algebră liniară

Toate mPTN ce vor fi considerate în continuare vor fi finite. În plus, mulțimile $P=\{p_1, ..., p_{|P|}\}$ și $T=\{t_1, ..., t_{|T|}\}$ vor fi considerate total ordonate prin ordinea naturală pe indicii elementelor. O mPTN poate fi descrisă prin procedee vectoriale, conform următoarei definiții.

Definiția 3-29 (Descrierea vectorială a mPTN).

Fie $N=(P,T;F,K,W,M_0)$ o mPTN.

(i) Pentru fiecare tranziție $t_j \in T$ se definesc *vectorii de tranziție* $t_j^+, t_j^- \in Z^{|P|}$ astfel:

$$t_j^+[i] = \begin{cases} W(t_j,p_i), \text{ dacă } (t_j,p_i)\in F \\ 0 \quad , \text{dacă} (t_j,p_i)\notin F \end{cases}$$

și

$$t_j^-[i] = \begin{cases} -W(p_i,t_j), \text{ dacă } (p_i,t_j)\in F \\ 0 \quad , \text{dacă } (p_i,t_j)\notin F, \quad \text{cu } 0<i\le|P|,\ 0<j\le|T| \end{cases}$$

(ii) Din vectorii de tranziție se formează matricele:

$$N^+ = |t_1^+ : t_2^+ : \ldots : t_{|T|}^+| \text{ și } N^- = |t_1^- : t_2^- : \ldots : t_{|T|}^-|$$

numite și *matricea pozitivă* respectiv *matricea negativă* a rețelei. Suma $N=N^+ + N^-$ este *matricea de incidențe* a rețelei suport N.

(iii) Vectorul $\mathbf{k} \in N^{+|P|}$ având elementele $\mathbf{k}[i]=K(p_i)$ cu $p_i \in P$ se numește *vectorul capacităților.*

(iv) Vectorul $\mathbf{m_0} \in N$ având elementele $\mathbf{m_0}[i]=M_0(p_i)$ cu $p_i \in P$ se numește *vectorul marcajului inițial.*

Dacă mPTN este o rețea P/T marcată atunci prin matricea de incidență a acesteia vom înțelge matrice de incidență a rețelei suport. Se va aplica aceeași convenție și pentru matricea pozitivă respectiv matricea negativă a unei mPTN.

Vectorii definiți mai sus sunt prezentați, pentru rețeaua din Figura 3-6, în Figura 3-7.

$$\underline{N^+} = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 2 & 1 \\ 0 & 0 & 1 & 0 \end{vmatrix} \qquad \underline{N^-} = \begin{vmatrix} 0 & -1 & 0 & 0 \\ -1 & 0 & -2 & -1 \\ 0 & 0 & 0 & -1 \end{vmatrix}$$

$$\underline{N} = \begin{vmatrix} 1 & -1 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & 0 & 1 & -1 \end{vmatrix} \qquad \underline{K} = \begin{vmatrix} 1 \\ 2 \\ 1 \end{vmatrix} \qquad \underline{m0} = \begin{vmatrix} 0 \\ 2 \\ 0 \end{vmatrix}$$

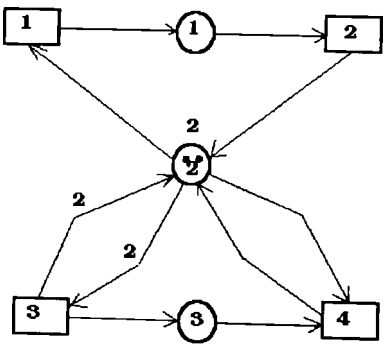


Fig. 3-7 Descrierea vectorială a unei mPTN

Se observă că matricea pozitivă a rețelei, N^+, conține capacitățile arcelor orientate de la tranziții la locații iar matricea negativă a rețelei, N^-, conține capacitățile arcelor orientate de la locații la tranziții. În acest fel se pot pune în evidență, în mod reversibil, buclele rețelei. În schimb matricea de incidență N nu poate pune în evidență buclele rețelei deoarece fiind dată o matrice N cu coeficienți în Z există o infinitate de rețele P/T marcate ce au matricea de incidență N, însă există o unică rețea pură cu matricea de incidență N.

Ca urmare a execuției tranzițiilor, din marcajul inițial, se generează noi marcaje. Marcajele pot fi specificate vectorial conform următoarei definiții:

Definiția 3-30 (Marcajul în forma matriceală).

(1) Marcajul unei mPTN $N=(P,T;F,K,W,M_0)$ este specificat prin funcția $M:P\rightarrow N$ cu $M(p)\leq K(p)$, $\forall p\in P$. Numărul natural astfel atașat fiecărei locații se numește *marcajul locației* p și specifică numărul marcajelor locației p.

(2) Vectorul $\mathbf{m}\in N^{|P|}$ format din marcajele locațiilor $p_i\in P$ se numește *vectorul marcajelor* și se definește prin: $\mathbf{m}[i]=M(p_i)$.

(3) Vectorul $\mathbf{m}_0\in N$ având elementele $\mathbf{m}_0[i]=M_0(p_i)$ cu $p_i\in P$ se numește *vectorul marcajului inițial.*

În continuare descrierea vectorială a unei mPTN se va specifica prin $N=(P,T;F,K,W,\mathbf{m}_0)$.

O tranziție t_j, executată, extrage fiecărei locații a premulțimii ei un număr de puncte egal cu ponderea arcului corespunzător și adaugă fiecărei locații a postmulțimii ei un număr de puncte egal cu ponderea arcului corespunzător. O tranziție este concesionată, adică poate să fie executată, dacă este satisfăcută relația de concesionare din Definiția 3-23.

Descrierea vectorială a rețelei permite o formulare vectorială a regulei de tranziție conform următoarei definiții:

Definiția 3-31 (Regula de tranziție exprimată vectorial).

Fie $N=(P,T;F,K,W,\mathbf{m}_0)$ o mPTN și $\mathbf{m},\mathbf{m}'$ două marcaje din rețeaua N.

(i) *Regula de concesionare.* O tranziție $tj\in T$ este concesionată prin $\mathbf{m}$ (adică este executabilă din marcajul $\mathbf{m}$) dacă este satisfăcută următoarea relație:

$$-\mathbf{tj}^- \leq \mathbf{m} \leq \mathbf{k} - (\mathbf{tj}^+ + \mathbf{tj}^-).$$

(ii) *Regula de calcul.* Prin execuția tranziției $t_j\in T$, concesionată prin marcajul $\mathbf{m}$, se generează un nou marcaj $\mathbf{m}'$, conform următoarei relații:

$$\mathbf{m}' = \mathbf{m} + (\mathbf{tj}^+ + \mathbf{tj}^-).$$

În felul în care s-a definit mai sus, condiția de concesionare are un caracter "tare" sau "strict". În anumite situații se utilizează și o regulă de tranziție "slabă" situație în care partea dreaptă a relației (i) din Definiția 3-31 nu se mai consideră a fi restrictivă. În continuare, dacă nu se specifică altfel, se va considera regula de tranziție "tare".

3.3.4 Proprietătile dinamice ale mPTN

Studiul sistematic al mPTN a scos în evidență anumite caracteristici ale acestora prin care pot fi specificate proprietățile dinamice ale rețelei. De o importanță majoră este problema accesibilității locațiilor. În acest context trebuie să se găsească un răspuns la întrebarea dacă și în ce fel, plecând de la un anumit marcaj inițial, și executând o anumită secvență de tranziții, se poate ajunge la un marcaj final. Mulțimea marcajelor realizate prin execuția tuturor secvențelor de tranziții posibile formează mulțimea marcajelor accesibile sau pe scurt *mulțimea accesibilă*.

Definiția 3-32 (Accesibilitate).

(i) O succesiune oarecare de tranziții $\sigma = t_{j1}, t_{j2}, t_{j3}, \ldots, t_{jn}$ cu $t_{jk} \in T$ și $0<j_k\leq|T|$ pentru $0<k\leq n$ se numește *secvență de tranziții de lungime n*.

(ii) O secvență de tranziții σ este aplicabilă la marcajul **m** dacă toate tranzițiile din σ sunt executabile în ordinea specificată în secvență, adică este satisfăcută regula de tranziție pentru k=1(1)n:

$$\mathbf{t}_{jk}^{-} - \sum_{i=1}^{k-1}\left(\mathbf{t}_{ji}^{+} + \mathbf{t}_{ji}^{-}\right) \leq \mathbf{m} \leq \mathbf{k} - \sum\left(\mathbf{t}_{ji}^{+} + \mathbf{t}_{ji}^{-}\right)$$

(iii) Un marcaj **m** a unei mPTN se numește *accesibil în N* dacă există o secvență de tranziții σ aplicabilă marcajului $\mathbf{m}_0$ și care asigură realizarea marcajului **m** din marcajul $\mathbf{m}_0$.

(iv) Mulțimea tuturor marcajelor accesibile $R_N(\mathbf{m}_0)=\{\mathbf{m} \mid \mathbf{m}$ accesibil în N$\}$ se numește *mulțimea marcajelor accesibile* sau pe scurt *mulțimea accesibilă* a mPTN. Notația $R_N(\mathbf{m})$ se va simplifica la $R(\mathbf{m})$ ori de câte ori nu există pericolul confuziei.

Numărul elementelor mulțimii accesibile crește exponențial cu mărimea rețelei (numărul de noduri) iar în cazul în care se aplică regula de tranziție "slabă" poate deveni chiar infinit. Mulțimea accesibilă finită poate fi reprezentă cu ajutorul unui graf bipartit, orientat, etichetat. Acest graf, numit în continuare *graf de accesibilitate*, va conține toate locațiile accesibile ca noduri (inscripționate cu transpusa vectorilor marcajelor) iar pașii efectuați ca arce (Figura 3-8).

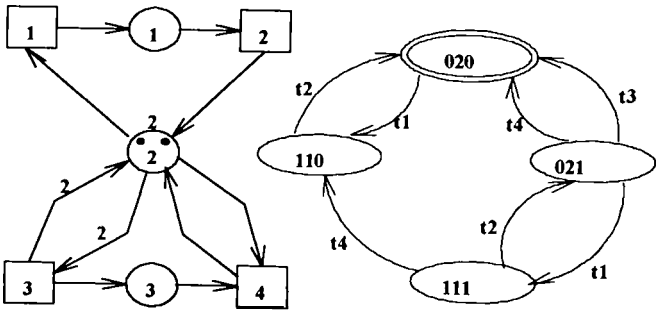


Fig. 3-8 O mPTN și graful ei de accesibilitate (marcajul inițial cu conturul dublat

Interpretând distribuția punctelor (vectorii marcajelor) ca stări globale ale sistemului modelat graful de accesibilitate poate fi interpretat ca o diagramă de stări care descrie un automat clasic echivalent rețelei Petri. Mărimea automatului este determinată de numărul marcajelor accesibile, mult mai mare decât numnărul locațiilor. Acest fapt probează capacitatea mPTN de a oferi, la același nivel de abstractizare, o descriere mult mai compactă a sistemului modelat decât ar oferi un automat clasic.

În funcţie de marcajele accesibile şi tranziţiile concesionabile într-o mPTN este posibilă exprimarea caracteristicilor de reţea.

O mPTN este *reversibilă* dacă orice marcaj al mulţimii accesibile este accesibilă din orice alt marcaj. În cazul în care în loc de accesibilitate a marcajelor interesează concesinabilitatea tranziţiilor, se are în vedere *viabilitatea* reţelei Petri adică posibilitatea de a concesiona respectiv executa fiecare tranziţie.

Pentru descrierea formală a acestor caracteristici se au în vedere următoarele definiţii:

Definiţia 3-33 (Reversibilitate).

O mPTN este *reversibilă* dacă pentru oricare două marcaje din mulţimea accesibilă se găseşte o secvenţă aplicabilă astfel încât dintr-unul dintre marcaje poate fi realizat cel de al doilea marcaj, adică dacă

$$\forall \mathbf{m}_1, \mathbf{m}_2 \in R(\mathbf{m}_0): \mathbf{m}_2 \in R(\mathbf{m}_1)$$

Din definiţia de mai sus rezultă că, plecând din orice marcaj al mulţimii accesibile, se poate realiza, din nou, marcajul iniţial.

Viabilitatea este determinată de concesionabilitatea tranziţiilor, conform următoarei definiţii:

Definiţia 3-34 (Viabilitate)

(i) O tranziţie $t_j \in T$ a unei mPTN N este viabilă dacă este concesionabilă din oricare marcaj accesibil, adică dacă:

$$\forall \mathbf{m}_1 \in R(\mathbf{m}_0), \exists\, \mathbf{m}_2 \in R(\mathbf{m}_1): t_j \text{ este concesionabilă prin } \mathbf{m}_2 .$$

(ii) O mPTN N se caracterizează printr-o *viabilitate slabă* dacă măcar o tranziţie a ei este viabilă, adică:

$$\exists t_j \in T: t_j \text{ este viabilă.}$$

(iii) O mPTN este *viabilă* (sau prezintă o *viabilitate tare*) dacă toate tranziţiile reţelei sunt viabile, adică:

$$\forall\, t_j \in T: t_j \text{ este viabilă.}$$

În literatura de specialitate se definesc şi alte grade de viabilitate (ex. în [PĂS'97]) utile în anumite aplicaţii.

În legătură cu noţiunea de viabilitate se definesc noţiunile de tranziţie moartă respectiv marcaj mort.

Definiţia 3-35 (Tranziţie moartă/Marcaj mort)

(1) O tranziţie $t_j \in T$ a unei mPTN este "moartă" dacă nu este concesionabilă din nici un marcaj al reţelei, adică dacă:

$$\neg\exists\, \mathbf{m} \in R(\mathbf{m}_0): t_j \text{ este concesionabilă prin } \mathbf{m} .$$

(2) Un marcaj **m** al unei mPTN este "mort" dacă nu există nici o tranziţie a reţelei concesionată de către **m** (marcajul **m** nu poate concesiona nici o tranziţie), adică dacă:

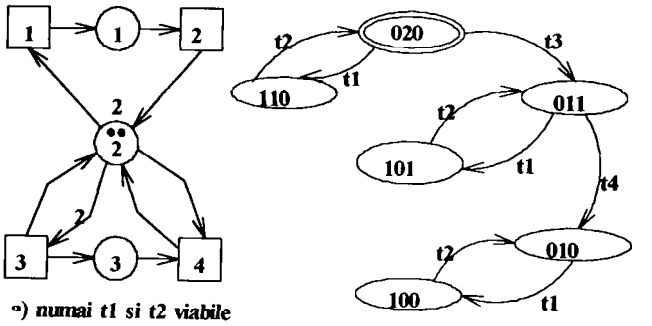
$$\neg\exists\, t_j \in T: t_j \text{ este concesionată prin } \mathbf{m}.$$

Noţiunile de mai sus derivă din situaţiile practice de blocaj în sistemele cu procese paralele cuplate prin resurse comune. Situaţia de blocaj apare dacă două sau mai multe procese active aşteaptă reciproc eliberarea resursei comune cu exploatare exclusivă şi

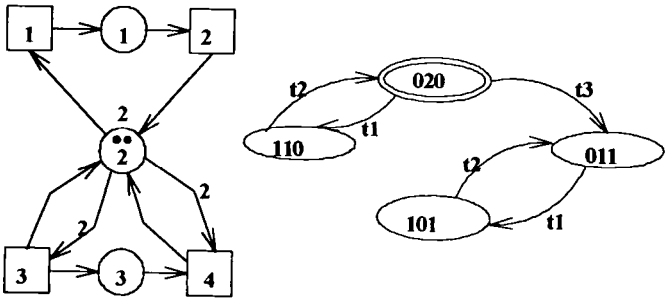
astfel desfăşurarea normală a proceselor este compromisă. În cazul în care se blochează toate procesele sistemului, blocare este totală altfel blocarea este parţială.

Desigur, blocarea totală este cauzată de un marcaj mort. Într-o mPTN caracterizată printr-o viabilitate slabă nu poate să apară blocarea totală deoarece măcar o tranziţie fiind viabilă nu poate exista vre-un marcaj mort în mulţimea accesibilă. În schimb pot să existe tranziţii neviabile şi chiar moarte. Desigur, definiţiile 3-34 şi 3-35 permit existenţa tranziţiilor care nu sunt nici viabile şi nici moarte. Acest lucru se întâmplă în situaţia în care o tranziţie este concesionabilă (deci nu este moartă) dar nu din orice marcaj (deci tranziţia nu este viabilă). Pentru viabilitatea reţelei toate tranziţiile trebuie să fie viabile şi în acest fel această caracteristică exclude nu numai blocarea totală dar şi pe cea parţială.

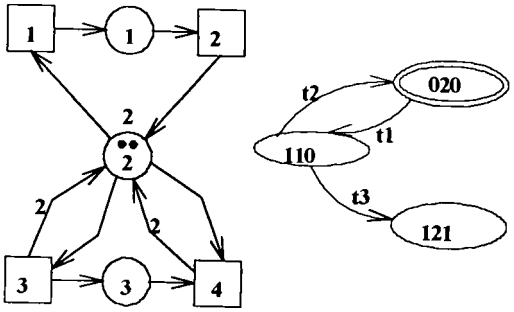
În Figura 3-9 se ilustrează aceste caracteristici prin intermediul unor reţele simple. Graful de accesibilitate al acestora fiind destul de simplu se verifică, prin analiză directă, caracteristicile specifice. Astfel, se observă că reţeaua din Figura 3-8 este reversibilă şi prezintă o viabilitate tare. În schimb Figura 3-9 prezintă trei variante nereversibile ale acestei reţele. Se observă şi pierderea, prin modificările efectuate, a caracteristicii de viabilitate.



a) numai t1 si t2 viabile



b) numai t1 si t2 viabile; t4 moarta



c) nici o tranzitie viabila; t4 moarta

Fig. 3-9 mPTN cu blocaj parţial (a), (b) şi blocaj total (c)

Grafurile de accesibilitate ale reţelelor din Figura 3-9(a) şi 3-9(b) arată că doar tranziţiile t1 şi t2 sunt întotdeauna concesionabile şi deci viabile. În a doua reţea tranziţia t4 nu poate fi executată, este o tranziţie moartă. Reţeaua din Figura 3-9(c) nu are nici o tranziţie viabilă; mai mult, secvenţa de tranziţii σ = t1,t3 produce marcajul mort $\mathbf{m}=(1,2,1)^T$, care exclude orice altă concesionare.

În reţelele Petri apare deseori situaţia în care tranziţiile sunt concesionate simultan. Dacă aceste tranziţii au premulţimile şi postmulţimile strict disjuncte tranziţiile pot fi executate independent, fără a se influenţa (concurenţă). În cazul unor locaţii comune în premulţime sau postmulţime se poate întâmpla ca după anumite execuţii să se piardă concesionarea unor tranziţii. În astfel de situaţii tranziţiile pot fi executate doar alternativ. Aceste tranziţii sunt într-o situaţie de *conflict* conform următoarei definiţii:

Definiţia 3-36 (Conflict)

Fie $\mathbf{m}_2$ marcajul rezultat din $\mathbf{m}_1$ ca urmare a tranziţiei t_k.

(1) Marcajul $\mathbf{m}_1$ este conflictual pentru tranziţiile t_k şi t_j, ($k \neq j$) dacă:

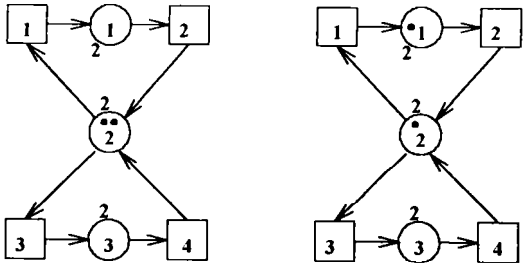
$\exists\, \mathbf{m}_1, \mathbf{m}_2 \in R(\mathbf{m}_0)$: t_j este concesionată prin $\mathbf{m}_1 \wedge t_j$ nu este concesionată prin $\mathbf{m}_2$.

(2) O mPTN este neconflictuală (fără vre-o situaţie conflictuală) dacă:

$\neg\exists \mathbf{m} \in R(\mathbf{m}_0)$: $\mathbf{m}$ este conflictual.

Pentru rezolvarea situaţiilor de conflict este nevoie de informaţie suplimentară din "mediul exterior" care să specifice tranziţia care se execută. Sintaxa reţelei nu specifică nici o ordine de execuţie a tranziţiilor concesionate.

Figura 3-10 ilustrează noţiunea de conflict .

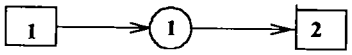


a)Mneconflictual b)Mxconflictual (t1 si t3)

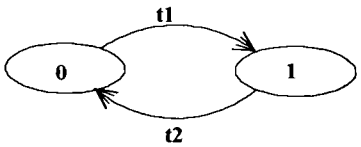
Fig. 3-10 mPTN conflictuală

Marcajul $\mathbf{m}_0$ activează în aceeaşi măsură tranziţiile t1 şi t3 . Fiecare dintre aceste tranziţii poate fi concesionată fără anularea concesionabilităţii celeilalte. Astfel, în Figura 3-10 după execuţia tranziţiei t1 ia naştere marcajul $\mathbf{m}_1$ din care sunt executabile ambele tranziţii concesionate. După o nouă execuţie a tranziţiei t1 tranziţia t3 devine neconcesionabilă şi deci la marcajul $\mathbf{m}_1$ tranziţiile t1 şi t3 sunt conflictuale.

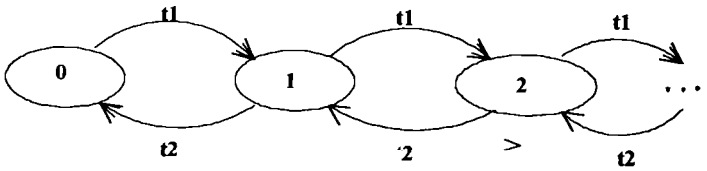
Aşa cum s-a amintit mai sus, în situaţia în care se aplică regula de tranziţie slabă, mulţimea accesibilă a reţelei poate deveni infinită, aşa cum se poate observa în Figura 3-11.



a) retea Petri



b) Multimea realizabila pentru regula de executie tare



c) ... respectiv pentru regula de executie slaba.

Fig. 3-11 mPTN mărginită

În cazul aplicării regulei de tranziţie slabe, dacă tranziţia t_1 este permanent concesionată poate transmite locaţiei p_1 oricâte marcaje.

Definiţia 3-37 (Mărginire).

(i) O locaţie $p_i \in P$ a unei mPTN N se numeşte k-mărginită la $\mathbf{m}_0$ dacă:

$$\exists k \in \mathcal{N},\ \forall \mathbf{m} \in R(\mathbf{m}_0):\ M(p_i) \le k.$$

(ii) O mPTN este mărginită la $\mathbf{m}_0$, dacă fiecare locaţie $p_i \in P$ este k-mărginită; adică dacă:

$\forall p_i \in P$: p_i este k - mărginită.

Mărgnirea este esenţială pentru majoritatea metodelor de analiză. Ea este *apriori* dată pentru regula de tranziţie tare. Se poate întâmpla însă ca modelarea să impună utilizarea regulei de tranziţie slabe. În acest caz este necesară studierea mărginirii şi dacă este cazul forţarea ei prin măsuri constructive.

3.4 Invarianţi de reţea

Proprietăţile dinamice ale unei reţele pot fi analizate dacă mulţimea accesibilă este explicit cunoscută. Găsirea fiecărui marcaj accesibile prin aplicarea regulei de tranziţie, la sisteme mari, este anevoiasă iar în cazul reţelelor nemărginite nici nu este posibilă.

Anumite proprietăţi ale mPTN pot fi determinate prin aşa numitul "calcul de accesibilitate" care stabileşte o relaţie între accesibilitate şi soluţia unui sistem de ecuaţii. În cazul în care, la aplicarea regulei de tranziţie, se neglijează condiţia de concesionare (aplicabilitate) şi se urmăreşte doar fluxul de puncte (marcaje), procesele de tranziţie pot fi reprezentate într-o formă concisă. Orice vector de marcaj rezultat dintr-o secvenţă de tranziţii se poate reprezenta printr-o combinaţie liniară de forma:

$$\mathbf{m}=\mathbf{m}_0+v_1\cdot(t_1^+ +t_1^-) + v_2\cdot(t_2^+ +t_2^-) + \ldots + v_{|T|}\cdot(t_{|T|}^+ +t_{|T|}^-) \qquad (3.4\text{-}1)$$

unde: $v_1, v_2, \ldots, v_{|T|} \in N$ reprezintă numărul execuţiilor tranziţiei date.

Dacă se formează vectorul: $\mathbf{v}^T=|v_1\ v_2\ \ldots\ v_{|T|}| \in N$, (3.4-2)

relaţia 2.2-1 se poate scrie sub forma:

$$\mathbf{m}=\mathbf{m}_0+[t_1^+ : t_2^+ : \ldots : t_{|T|}^+]\cdot\mathbf{v}+[t_1^- : t_2^- : \ldots : t_{|T|}^-]\cdot\mathbf{v} \qquad (3.4\text{-}3)$$

Conform Definiţiei 3-28, relaţia (3.4-3) se poate scrie sub forma concentrată:

$$\mathbf{m}=\mathbf{m}_0+\mathbf{N}^+\cdot\mathbf{v}+\mathbf{N}^-\cdot\mathbf{v} \qquad (3.4\text{-}4)$$

respectiv, în cazul reţelelor pure

$$\mathbf{m}=\mathbf{m}_0+\mathbf{N}\cdot\mathbf{v} \qquad (3.4\text{-}5)$$

unde **N** este matricea de incidenţă a reţelei care specifică conexiunile dintre locaţiile şi tranziţiile reţelei Petri.

Rezolvabilitatea ecuaţiei :

$$\mathbf{N}\cdot\mathbf{v}=\mathbf{m}-\mathbf{m}_0 \qquad (3.4\text{-}6)$$

în domeniul numerelor naturale, adică $\mathbf{v}[i] \in N$, este condiţia necesară pentru accesibiliatea marcajului **m**. Marcajele care nu sunt soluţii naturale ale acestei ecuaţii nu sunt marcaje accesibile. Condiţia exprimată prin ecuaţia (3.4-6) nu este şi suficientă deoarece în deducerea ei nu a fost inclusă şi condiţia de concesionare.

Proprietăţile dinamice ale mPTN pot fi studiate pe baza soluţiilor unor ecuaţii speciale deduse din ecuaţia (3.4-6). Aceste soluţii se numesc "invarianţi" ai reţelei Petri deoarece acestea sunt determinate doar de structura reţelei şi sunt invariante faţă de schimbările de marcaje din reţea. Se definesc P-invarianţi şi T-invarianţi

Definiţia 3-38 (T-invarianţi)

Un vector $\mathbf{i}_T \in Z^{|T|}$ este un T-invariant al mPTN, dacă: $\mathbf{N}\cdot\mathbf{i}_T=\mathbf{0}$.

Fiecare T-invariant pozitiv ($\mathbf{i}_T \in N^{|T|}$) specifică tranzițiile care trebuie să fie concesionate pentru realizarea unui anumit marcaj precum și numărul concesionărilor necesare. Relația din definiție se obține prin egalarea elementelor din membrul drept al relației (3.4-6), ceea ce exprimă faptul că vectorul $\mathbf{i}_T$ este invariant față de marcajul realizat.

P-invarianții caracterisează anumite categorii de locații ale unei mPTN.

Definiția 3-39 (P-invarianți)

Un vector $\mathbf{i}_P \in Z^{|P|}$ se numește P-invariant al unei mPTN, dacă: $\mathbf{N}^T\cdot\mathbf{i}_P=\mathbf{0}$.

Un P-invariant specifică faptul că, în procesele de tranziție, numărul marcajelor, ponderat prin acest P-invariant, rămâne constant. Această caracteristică poate fi dedusă din relația (3.4-6) și este utilă în studiul mărginirii rețelei. Ea poate fi exprimată prin următoarea relație de "conservare":

$$\mathbf{m}^T\cdot\mathbf{i}_P=\mathbf{m}_0{}^T\cdot\mathbf{i}_P \quad (3.4\text{-}7)$$

Prin calculul și utilizarea invarianților pot fi studiate caracteristicile dinamice ale rețelelor. Deși analiza bazată pe calculul invarianților este deosebit de elegantă ea nu oferă rezultate absolute. Datorită neglijării condiției de concesionare se obțin, fie condițiile necesare fie cele suficiente. Din acest motiv nu se poate renunța la determinare și studiul mulțimii accesibile.

3.5 Concluzii

Modelarea discretă, orientată pe evenimente a circuitelor logice de mare viteză sau a sistemelor distribuite complexe trebuie să țină cont de efectele relativiste ce apar în aceste sisteme. Modelarea realistă a acestora nu poate fi realizată pe baza ipotezei unei ordonări totale, obiective independentă de observator. Modelarea trebuie să aibă la bază relațiile cauzale care guvernează sistemul.

S-au definit două relații dintre evenimente care pot sta la baza modelării orientate pe evenimente a sistemelor distribuite. Aceste două relații sunt: relația de ordonare parțială, (notată în prezenta lucrare prin "*pre*") și relația de concurență, netranzitivă dar reflexivă (notată în prezenta lucrare prin "*co*"). Aceste relații permit definirea și studiul mulțimilor parțial ordonate respectiv a structurilor de evenimente.

Pentru modelarea discretă, orientată pe evenimente a sistemelor reale trebuie luate în considerare și condițiile în care pot apare evenimentele repectiv efectul producerii acestora asupra evoluției sistemului. Modelul care încorporează condiții și evnimente se prezintă sub forma unor rețele bipartite, orientate.

Teoria acestor rețele este o teorie a modelării discrete, relativiste, orientată pe evenimente. Elementele rețelei sunt condiții și evenimente. În cadrul rețelei condițiile sunt încadrate doar de evenimente iar evenimentele doar de condiții.

S-au elaborat, și în literatura de specialitate se prezintă, diverse tipuri de astfel de rețele care, de cele mai multe ori, sunt referite cu denumirea generică de rețele Petri. Acest capitol a tratat, într-o formă inginerească, adoptată nevoilor modelării în domeniul automatizărilor industriale, clasa rețelelor Petri Condiții/Evenimente și Locație/Tranziție.

Fundamentarea matematică riguroasă a tehnicilor de analiză bazate pe mPTN permit execuţia asistată de calculator a modelării şi analizei sistemelor tehnice modelate prin reţele P/T marcate.

Forma de prezentare aleasă în acest capitol clarifică noţiunile fundamentale legate de modelarea discretă, orientată pe evenimente a sistemelor distribuite şi implicit clarifică bazele conceptuale ale teoriei şi modelării prin reţele Petri. În acelaş timp prezintă teoria reţelelor Petri într-o formă inginerească care poate sta la baza utilizării efective a acestora în modelarea şi realizarea sistemelor de automatizare.

4. Analiza reţelelor Petri

Acest capitol prezintă, tehnicile de bază ale analizei reţelelor Petri Locaţie /Tranziţie marcate. Scopul analizei este de a pune în evidenţă proprietăţile de viabilitate, reversibilitate şi mărginire. Analiza poate fi efectuată pe două căi fundamental diferite: (i) analiza bazată pe graful de accesibilitate şi (ii) analiza bazată pe invarianţi.

4.1 Consideraţii preliminare

Analiza mPTN are ca scop punerea în evideţă a unor proprietăti pe baza cărora pot fi apreciate caracteristicile dinamice ale sistemului tehnic modelat. Procedeele ce se prezintă în acest capitol permit punerea în evidenţă a tuturor proprietăţilor definite in capitolul 3.3.4 şi permit formularea de sentinţe cu privire la accesibilitatea marcajelor, apariţia situaţiilor conflictuale, a marcajelor respectiv tranziţiilor moarte, punerea în evidenţă a proprietăţii de reversibilitate, viabilitate şi mărginire a mPTN.

Analiza mPTN poate fi realizată pe două căi fundamental diferite. În cazul analizei bazate pe grafuri este studiată mulţimea accesibilă şi este, din acest motiv, deosebit de sugestivă. Acest gen de analiză presupune disponibilitatea mulţimii accesibile. În cazul reţelelor mărginite mulţimea accesibilă este întotdeauna construibilă.

Pentru demonstrarea viabilităţii şi reversibilităţii reţelei este nevoie de o analiză a structurii mulţimii accesibile respectiv al grafului accesibil. În acestă analiză se utilizează noţiunea de graf condensat, univoc obtenabil din graful de accesibilitate. Procedeele de construcţie şi analiză ale grafului condensat sunt prezentate în paragraful 4.2.

Informaţii referitoare la comportamentul dinamic al reţelei sunt obtenabile direct prin P-invarianţii P şi T-invarianţi, construiţi pe baza matricii de incidenţe a reţelei. Criteriile necesare analizei pot fi deduse din ecuaţiile de definiţie ale invarianţilor şi din metodele de calcul ale marcajelor accesibile, conform paragrafului 4.3.

4.2 Analiza mulţimii accesibile

În cazul în care se cunoaşte mulţimea accesibilă precum şi reprezentarea ei formală - graful de accesibilitate - pot fi puse în evidenţă proprietăţile dinamice ale reţelei.

Pentru fiecare mPTN, mărginită, se poate construi graful de accesibilitate $GA_N=(Q,A,\tau)$. Graful de accesibilitate se defineşte astfel:

Definiţia 4-1 (Graful de accesibilitate)

Se numeşte *graf de accesibilitate al unei mPTN* graful orientat $GA_N=(Q,A,\tau)$ care are ca noduri marcajele accesibile în reţeaua mPTN:

$Q=R_N(\mathbf{m}_0)$

iar ca arce, tranziţiile realizate reprezentate prin perechi ordonate, astfel:

$A=\{(\mathbf{m},\mathbf{m}') \mid t_j$ este concesionat de $\mathbf{m} \wedge \mathbf{m} \in R_N(\mathbf{m}_0) \wedge \mathbf{m}' = \mathbf{m}+(t_j'+t_j^-)\}$.

Funcţia $\tau : A \rightarrow T$ alocă fiecărui arc $(\mathbf{m},\mathbf{m}') \in A$ tranziţia $t_j \in T$ care a realizat marcajul $\mathbf{m}'$ din marcajul $\mathbf{m}$. În acest fel sunt prezente în graful de accesibilitate toate secvenţele de tranziţii aplicabile în reţea.

Graful de accesibilitate definit mai sus deschide posibilitatea analizei pe baza teoriei grafurilor. Notaţia $GA_N=(Q,A,\tau)$ se va simplifica la $GA=(Q,A,\tau)$ ori de câte ori nu există pericolul confuziei.

Construcţia grafului accesibil presupune calculul tuturor marcajelor accesibile plecând de la marcajul iniţial prin aplicarea succesivă a regulei de tranziţie. În acelaş timp trebuie specificate şi tranziţiile ce realizează aceste marcaje.

Anumite proprietăţi ale reţelei sunt direct deductibile din graful de accesibilitate conform următoarei teoreme:

Teorema 4-1 (Analiza prin graful de accesibilitate)

Fie GA graful de accesibilitate al unei mPT N

(1) Dacă GA conţine un marcaj $\mathbf{m} \Leftrightarrow$ marcajul $\mathbf{m}$ este accesibil în mPTN.

(2) Dacă GA conţine un nod $\mathbf{m}$ fără vre-un arc care pleacă din acest nod $\Leftrightarrow \mathbf{m}$ este un marcaj mort al reţelei mPTN.

(3) Dacă $\exists t_j \in T$: GA nu conţine nici un arc etichetat cu $t_j \Leftrightarrow$ tranziţia t_j este o tranziţie moartă a mPTN.

(4) Fie $(\mathbf{m},\mathbf{m}')$ un arc al grafului accesibil etichetat cu $t_k \in T$, dacă $\exists t_j \in T$ $(j \neq k)$: din $\mathbf{m}$ pleacă un arc etichetat cu $t_j \wedge$ din $\mathbf{m}'$ nu pleacă un arc etichetat cu $t_j \Leftrightarrow$ în reţeaua mPTN, la marcajul $\mathbf{m}$, între tranziţiile t_j şi t_k există o relaţie de conflict.

Proprietăţile de viabilitate şi reversibilitate nu pot fi extrase în mod direct din graful de accesibilitate. Acestea se reflectă în proprietăţile structurale ale grafului accesibil. Pentru a caracteriza aceste proprietăţi se defineşte noţiunea de *graf cu coeziune puternică*, astfel:

Definiţia 4-2

Un graf orientat D se numeşte *graf cu coeziune puternică* dacă oricare două noduri ale sale sunt legate, prin căi orietate, în două direcţii.

În sensul definiţiei de mai sus, un graf trivial, care constă dintr-un singur nod, este un graf cu coeziune puternică.

În exemplul din Figura 4-1 graful D1 este cu coeziune puternică, deoarece toate nodurile pot fi legate prin căi orientate în două direcţii. D2, în schimb conţine un nod fără arce care intră în nod din acest motiv graful D2 nu prezintă o coeziune puternică.

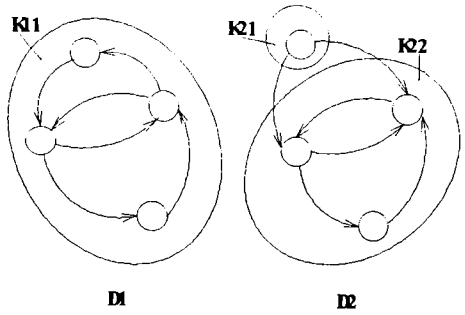


Fig. 4-1 Componente tari ale grafurilor D1 şi D2

Într-un graf se pot defini subgrafuri cu coeziune puternică. Dacă un subgraf cu coeziune puternică, după extragerea oricărui nod, îşi pierde această proprietate acesta este un subgraf cu coeziune maximală numit şi *componentă tare* a grafului D, conform următoarei definiţii:

Definiţia 4-3

> Se numeşte *subgraf cu coeziune maximală* un subgraf cu coeziune puternică care îşi pierde această proprietate, după extragerea oricărui nod.

În Figura 4-1 se poate observa că graful D2, care nu prezintă o coeziune puternică, se compune din două componente tari, K21 şi K22, iar graful D1, cu coeziune puternică, reprezintă în acelaşi timp singura componentă tare, K11.

Cu ajutorul componentelor tari ale unui graf D poate fi construit graful redus D^K al grafului D. Nodurile acestuia sunt componentele tari ale grafului D. Arcele realizează structurarea grafului redus. Graful redus D^K se mai numeşte şi ***condensata*** grafului D, conform următoarei definiţii:

Definiţia 4-4

> Se numeşte *condensata grafului D* graful ale cărui noduri sunt componentele tari ale grafului D iar arcele sunt acele arce ale grafului D care conectează componentele tari.

Figura 4-2 prezintă condensatele grafurilor D1 şi D2 din figura 4-1.

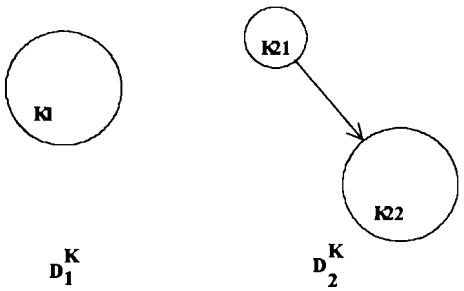


Fig. 4-2 Condensatele grafurilor D1 şi D2

Condensata unui graf D conţine un arc orientat între nodurile K_i şi K_j dacă şi numai dacă în graful original D există un arc care leagă un nod al lui K_i cu un nod al lui K_j.

Din definiţia condensatei rezultă că nodurile acesteia conţin două categorii speciale: *noduri sursă* - de la care pleacă arce şi la care nu soseşte nici un arc - şi *noduri de absorbţie* - la care sosesc arce şi de la care nu pleacă nici un arc. Condensata unui graf nu prezintă proprietăţi de coeziune puternică. Condensata D_1^K a grafului D_1 (Figura 4-1) cu coeziune puternică constă numai dintr-un singur nod care este în acelaş timp nod sursă şi nod de absorbţie.

Pe baza condensatei grafului accesibil pot fi analizate proprietăţile de viabilitate şi reversibilitate ale reţelei Petri în cauză. În acest sens sunt importante doar nodurile de absorbţie ale condensatei. Aceasta deoarece dacă la aplicarea regulei de tranziţie se ajunge la un marcaj conţinut într-un nod de absorbţie atunci sunt accesibile doar marcaje şi sunt executabile doar tranziţii conţinute în acest nod. Pe baza acestei observaţii pot fi formulate condiţii necesare şi suficiente pentru viabilitatea şi reversibilitatea reţelei Petri.

Înainte de formularea condiţiei de viabilitate se definesc noţiunile de componentă vie respectiv componentă moartă a condensatei unei mulţimi accesibile.

Definiţia 4-5 (Componentă vie/Componentă moartă)

Fie N o mPTN, GA graful ei accesibil şi GA' o componentă tare a lui GA.

(1) Componenta GA' se numeşte *componentă absolut vie* dacă pentru oricare tranziţie $t_j \in T$ conţine cel puţin un arc etichetat cu t_j, adică dacă:
$$\forall t_j \in T \ \exists a \in A : \tau(a)=t_j.$$

(2) Componenta GA' se numeşte *componentă relativ vie* dacă la cel puţin o tranziţie $t_j \in T$ conţine cel puţin un arc etichetat cu t_j, adică dacă:
$$\exists t_j \in T, a \in A : \tau(a)=t_j.$$

(3) Componenta GA' se numeşte *componentă moartă* dacă nu conţine, pentru nici o tranziţie $t_j \in T$, vre-un arc etichetat cu t_j, adică dacă:
$$\neg \exists t_j \in T, a \in A : \tau(a)=t_j.$$

Pe baza noţiunilor de mai sus pot fi formulate criteriile de analiză a viabilităţii şi reversibilităţii.

Teorema 4-2 (Analiza prin intermediul condensatei)

Fie N o mPTN, GA graful ei accesibil şi GA^K condensata lui GA.

(1) Dacă GA prezintă o coeziune puternică (adică constă dintr-un singur nod care este în acelaş timp şi nod sursă şi nod de absorbţie) $\Leftrightarrow$ mPTN este reversibilă.

(2) Dacă fiecare nod de absorbţie al lui GA^K este o componentă vie, mPTN este viabilă.

(3) Dacă nici un nod de absorbţie al lui GA^K nu este o componentă moartă $\Leftrightarrow$ mPTN prezintă o viabilitate slabă.

Pentru exemplificare Figura 4-3 prezintă o mPTN împreună cu graful ei accesibil şi cu condensata acestuia.

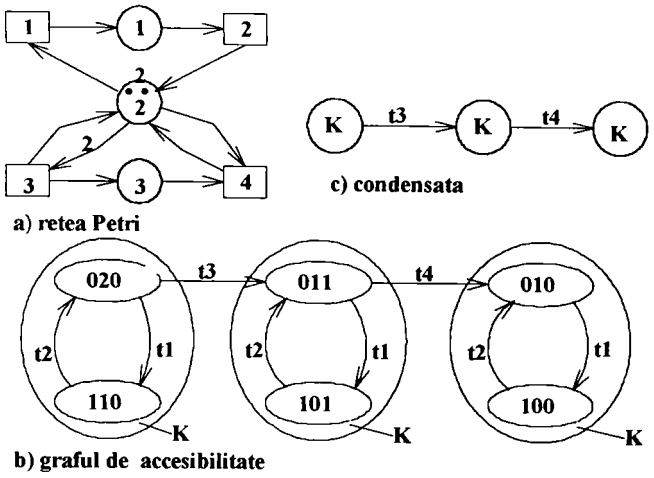


Fig. 4-3 mPTN nereversibilă, cu viabilitate slabă

Reţeaua nu este reversibilă deoarece condensata ei are mai multe componente tari. Condensata nu conţine vre-o compnentă moartă deoarece tranziţiile t_1 şi t_2 sunt executabile. Nici o componentă nu este vie, componenta K_3, singurul nod de absorbţie, nefiind vie rezultă că reţeaua prezintă o viabilitate slabă (conform definiţiei 4-5).

4.3 Analiza matricei de incidenţe a reţelei

Analiza matricei de incidenţe a reţelei se realizează prin metode algebrice şi are ca scop calculul şi utilizarea P-invarianţilor şi T-invarianţilor ţinând cont de legătura dintre anumite proprietăţi dinamice ale reţelei Petri şi anumite proprietăţi ale invarianţilor conform definiţiei acestora.

T-invarianţii pot fi utilizaţi în studiul reversibilităţii mPTN. În orice mPTN reversibilă se găseşte cel puţin o secvenţă aplicabilă care să reproducă marcajul iniţial. Secvenţele aplicabile care asigură reversibilitatea se găsesc printre secvenţele aplicabile în situaţia în care nu se impune satisfacerea condiţiei de aplicabilitate. Din Definiţia 3-37 rezultă că astfel de secvenţe sunt descrise tocmai de T-invarianţii reţelei. Poate fi formulată astfel următoarea teoremă:

Teorema 4-3 (Reversibilitate)

Pentru orice mPTN reversibilă se poate găsi un T-invariant nenegativ, adică:

Dacă mPTN este reversibilă $\Rightarrow \exists \mathbf{i}_T \in \mathcal{N}^{|T|} : \mathbf{N} \cdot \mathbf{i}_T = \mathbf{0}$.

Cu ajutorul T-invarianţilor se pot formula sentinţe asupra viabilităţii mPTN. Pentru fiecare mPTN se poate construi graful de accesibilitate şi condensata acestuia. Conform Teoremei 4-2 reţelele viabile conţin cel puţin o componentă tare care este, la rândul ei viabilă. Ca urmare a structurii componentelor tari există cel puţin un traseu închis de arce ale grafului accesibil care conţine, ca inscripţii, toate tranziţiile reţelei. Acestui traseu îi corespunde o secvenţă aplicabilă care conţine toate tranziţiile.

Teorema 4-4 (Viabilitate)

Pentru fiecare mPTN viabilă şi mărginită se poate găsi cel puţin un T-invariant pozitiv; adică:

Dacă mPTN este viabilă şi mărginită $\Rightarrow \exists \mathbf{i}_T \in \mathcal{N}^{|T|} : \mathbf{N} \cdot \mathbf{i}_T = \mathbf{0}$.

Relaţia de conservare (3.4-7) oferă posibilitatea utilizării P-invarianţilor pentru analiza mărginirii mPTN.

Următoarea inegalitate

$$\mathbf{m}[i] \cdot \mathbf{i}_p[i] \le \mathbf{m}[1] \cdot \mathbf{i}_p[1] + \mathbf{m}[2] \cdot \mathbf{i}_p[2] + \ldots + \mathbf{m}[|P|] \cdot \mathbf{i}_p[|P|]; \text{ cu } i=1(1)|P| \qquad (5.3\text{-}1)$$

este satisfăcută, în cazul P-invarinţilor nenegativi, pentru orice $p_i \in P$. Înlocuind, conform ecuatiei 3.4-7 , partea dreaptă a inegalităţii 5.3-1 cu expresia $\mathbf{m}_0^T \cdot \mathbf{i}_P$ se obţine:

$$\mathbf{m}[i] \cdot \mathbf{i}_p[i] \le \mathbf{m}_0^T \cdot \mathbf{i}_P \text{ cu } i = 1(1)|P|. \qquad (5.3\text{-}2)$$

Pentru invarianţi strict pozitivi, $\mathbf{i}_P[i] > 0$ se obţine aproximaţia:

$$m[i] \le \frac{m_0^T \cdot i_p}{i_p[i]} = k_i \quad \text{cu } i = 1(1)|P| \qquad (5.3\text{-}3)$$

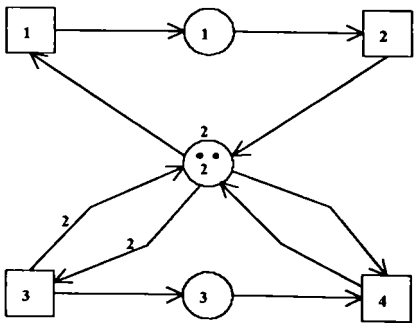
care, prin k_i, specifică o limită superioară pentru numărul de marcaje al fiecărei locaţii a reţelei şi conform definiţiei 3-37, însăşi reţeaua Petri este mărginită.

Teorema 4-5 (Mărginire)

O mPTN este mărginită dacă are invarainţi P pozitivi, adică dacă:

$\exists \mathbf{i}_p \in \mathcal{N}^{*|P|} : \mathbf{N}^T \cdot \mathbf{i}_P = \mathbf{0} \Rightarrow$ N este mărginită.

În Figura 4-4 se prezintă invarianţii P şi T pentru o mPTN mărginită, reversibilă şi viabilă.



$$\mathbf{i}_P = |0\,0\,0|^T + \lambda_1 \bullet |1\,1\,0|^T;\ \lambda_1 \in Z$$

$$\mathbf{i}_T = |0\,0\,0\,0|^T + \mu_1 \bullet |1\,1\,0\,0|^T + \mu_2 \bullet |0\,0\,1\,1|^T;\ \mu_1, \mu_2 \in Z$$

Fig. 4-4 Invarianţii P şi T ale unei mPTN

Se observă inmediat că pentru $\mu 1=\mu 2=1$ se obţine invariantul pozitiv $i_T=(1111)^T$ şi deci reţeaua este reversibilă şi viabilă. Invarianţii P în schimb nu sunt strict pozitivi deoarece pentru orice valoare a parametrului λ_1 un element al invariantului este nul. Conform teoremei 4-5 condiţia suficientă pentru mărginire nu este satisfăcută. Totuşi reţeaua este mărginită cea ce probează faptul că invarianţii P nu precizează şi condiţia necesară pentru mărginire.

Cu toate că invarianţii precizează doar condiţia necesară respectiv suficientă se dovedesc a fi un ajutor util în analiza mPTN.

4.4 Exemplul unei staţii de containere

Se prezintă exemplul, simplificat, al unei saţii de containere în care trei macarale portal transferă, simultan, containere între două trenuri (Figura 4-5). Macaralele portal A, B şi C se deplasează pe aceeaşi cale de rulare şi trebuie să transfere containere de pe trenul I pe trenul II. Pentru aceasta macaralele preiau câte un container se deplasează în poziţia de descărcare şi apoi descarcă containerul.

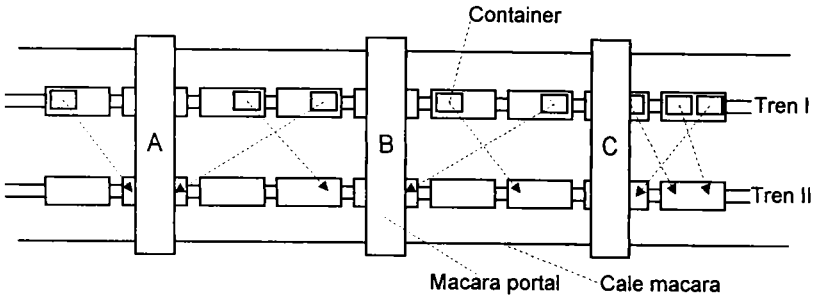


Fig. 4-5 Reprezentarea schematică a staţiei de containere

Sistemul descris poate prezenta blocaje deoarece conţine procese concurente (transport) şi resurse partajate cu utilizare exclusivă (segmente de cale, containere). Trebuie elaborat modelul de reţea Petri al acestui sistem, model pe baza căruia poate fi implementat sistemul de conducere.

4.4.1 Analiza bazată pe grafuri a unui subsistem

Modelarea şi analiza sistemului se va realiza în două etape:

1. se vor modela subsisteme componente, se vor analiza aceste modele şi, la nevoie, se vor corecta pentru a asigura caracteristicile dinamice impuse;
2. pe baza acestor modele, introducând şi resursele paratajate, se formează şi se analizează modelul întregului sistem.

Se va considera drept subsistem o singură macara portal care trebuie să execute transferul containerelor. Şina dealungul căreia se deplasează macaraua se consideră a fi divizată. Se obţin în acest fel segmente de cale. Fără a restrânge generalitatea raţionamentelor ce urmează se vor considera 12 segmente de cale. Se va considera că macaraua se poziţionează deasupra unui segment (Figura 4-6). Poziţia macaralei, exprimată prin numărul de segmente la stânga respectiv la dreapta acesteia, poate fi descrisă prin marcajul a două locaţii având capacitatea k=11.

Pentru a putea sintetiza modelul de rețea Petri al subsistemului trebuie stabilite toate acțiunile din sistem și toate condițiile în care acestea au loc.

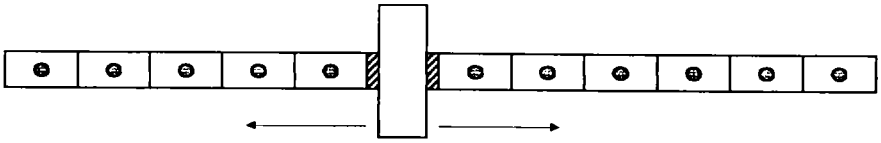


Fig. 4-6 Subsistemul unei sigure macarale portal

Acțiunile legate de funcționarea acestui subsistem sunt următoarele:

1 – începerea deplasării spre stânga;

2 – terminarea deplasării spre stânga;

3 – începerea deplasării spre dreapta;

4 – terminarea deplasării spre stânga;

5 – începerea depunerii unui container;

6 – începerea preluării unui container;

7 – terminarea depunerii respectiv a preluării unui container.

Condițiile în care acțiunile au loc sunt următoarele:

1 – macaraua așteaptă comenzi;

2 – macaraua încarcă respectiv descarcă un container;

3 – macaraua se deplasează spre stânga;

4 – macaraua se deplasează spre dreapta;

5 – macaraua este încarcată cu un container;

6 – macaraua este neîncăractă;

7 – numărul de segmente de cale din stânga macaralei;

8 – numărul de segmente de cale din dreapta macaralei;

Un prim model de rețea Petri al acestui subsistem se prezintă în Figura 4-7(a). Marcajul inițial precizează faptul că macaraua este neîncărcată și așteaptă comenzi respectiv faptul că se găsește pe popziția corespunzătoare Figurii 4-6.

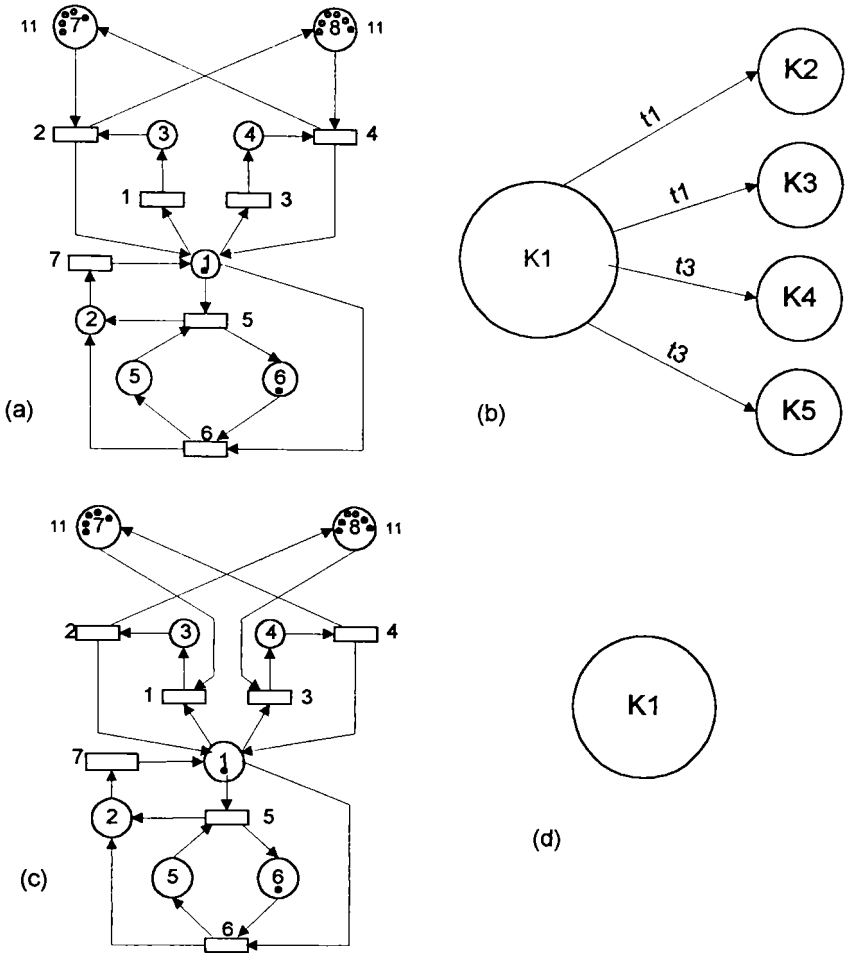


Fig. 4-7 Modelul de reţea Petri al subsistemului şi analiza bazată pe grafuri

Plecând din marcajul iniţial se pot găsi secvenţe de tranziţii, de exemplu $\sigma=t_1, t_2, t_6, t_7, t_3, t_4, t_5, t_7$, care readuc sistemul în starea iniţială. Cu toate acestea o analiză bazată pe grafuri arată că, condensata grafului de accesibilitate prezintă patru noduri de absorbţie K_2, ... K_5, (Fig. 4-7(b)). Acestor noduri le corespund următoarele patru marcaje moarte:

$$\mathbf{m}_2 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 11 \end{bmatrix}, \quad \mathbf{m}_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 11 \end{bmatrix}, \quad \mathbf{m}_4 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 11 \\ 0 \end{bmatrix}, \quad \mathbf{m}_5 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 11 \\ 0 \end{bmatrix},$$

O analiză a acestui rezultat arată că situaţiile de blocaj apar în cazul în care macaraua se găseşte într-o poziţie extremă (dreaptă sau stângă) şi primeşte o comandă nexecutabilă reprezentată prin tranziţiile t_1 sau t_3. Această greşeală de modelare se datorează faptului că un anumit proces (deplasare) poate fi iniţiat fără ca acesta să aibă rezervate resursele necesare (segemnte da cale).

În Figura 4-7(c) se prezintă modelul de reţea Petri corectat. Corecţia constă în faptul că de această dată deplasarea pote fi iniţiată doar dacă stă la dispoziţie măcar un segmnt de cale. O analiză a acestei reţele arată că ea este viabilă, condensata grafului de accesibilitate conţinând doar un singur nod cu coeziune puternică. Conform teoremei 4-2 reţeaua este şi reversibilă.

4.4.2 Modelul întregului sistem. Analiza bazată pe invarianţi.

Pe baza modelului de reţea Petri a unei singure macarale poate fi sintetizat modelul de reţea Petri a întregului sistem conţinând trei macarale. Cele trei subsisteme corespunzătoare celor trei macarale vor fi cuplate prin resurse partajate.

Într-un prim pas se va avea în vedere doar calea comună de rulare a celor trei macarale. Aceasta reprezintă o resursă paratajată a macaralelor. Reţeaua obţinută se prezintă în Figura 4-8.

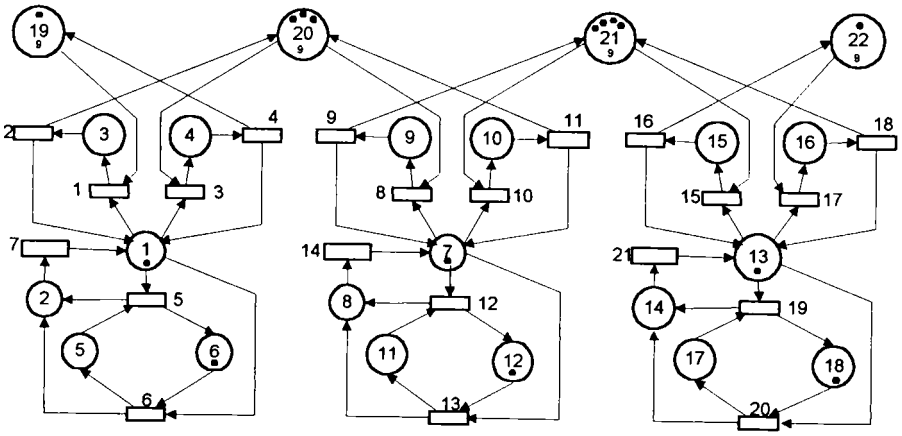


Fig. 4-8 Modelul de reţea Petri pentru trei macarale cu cale de rulare comună

Locaţiile 19, 20, 21 şi 22, având capacitatea k=9, modelează calea de rulare disponibilă. Caracteristicile dinamice ale sistemului se vor analiza pe baza invarianţilor de reţea. P-invarianţii reţelei sunt daţi de următoarea expresie:

$$\mathbf{i}_P = \begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\end{bmatrix} + P1\begin{bmatrix}0\\0\\1\\1\\0\\0\\0\\0\\1\\1\\0\\0\\0\\1\\1\\0\\0\\1\\1\\1\\1\end{bmatrix} + P2\begin{bmatrix}1\\1\\1\\1\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\end{bmatrix} + P3\begin{bmatrix}0\\0\\0\\0\\0\\0\\1\\1\\1\\1\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\end{bmatrix} + P4\begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\1\\1\\1\\1\\0\\0\\0\\0\\0\\0\end{bmatrix} + P5\begin{bmatrix}0\\0\\0\\0\\0\\1\\1\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\end{bmatrix} + P6\begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\1\\1\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\end{bmatrix} + P7\begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\1\\1\\0\\0\end{bmatrix}$$

T-invarianţii reţelei sunt daţi de următoarea expresie:

$$\mathbf{i}_T = \begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\end{bmatrix} + P1\begin{bmatrix}1\\1\\1\\1\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\end{bmatrix} + P2\begin{bmatrix}0\\0\\0\\0\\0\\0\\1\\1\\1\\1\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\end{bmatrix} + P3\begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\end{bmatrix} + P4\begin{bmatrix}0\\0\\0\\0\\1\\1\\2\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\end{bmatrix} + P5\begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\1\\1\\2\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\end{bmatrix} + P2\begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\1\\1\\2\\0\end{bmatrix}$$

Interpretarea acestor invarianți este mult facilitată de delimitarea acestora pe modelul de rețea Petri a sistemului așa cum se prezintă în Figura 4-9.

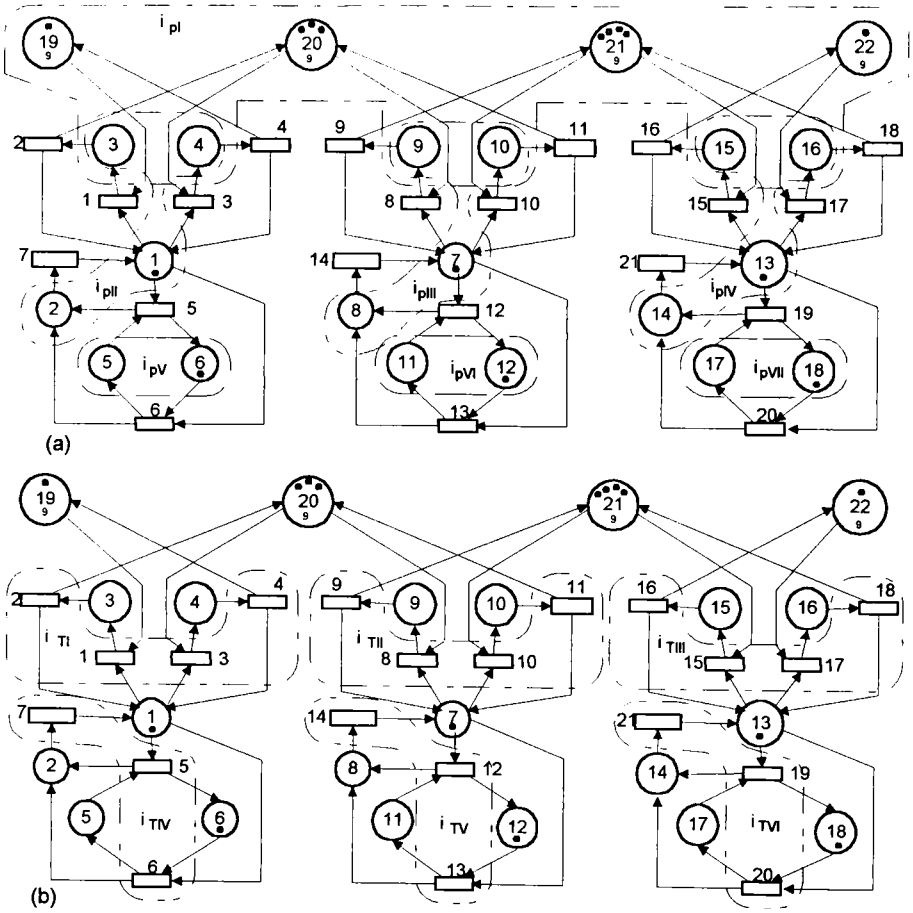


Fig. 4-9 Invarianții retelei din Figura 4-8

Se poate observa că P-invarianții rețelei caracterizează obiecte de aceeași natură. Locațiile care sunt prezente în mai mult de un singur invariant pot primi diverse interpretări. Astfel, în Figura 4-9(a):

- P-invariantul i_{PI} reprezintă *segmentele căii de rulare* comune ale macaralelor. Aceste segmente pot fi libere sau ocupate.
- P-invarianții i_{PII}, ..., i_{PIV} reprezintă *activitățile* macaralelor și anume așteptare, rulare sau încărcare/descărcare.
- P-invarianții i_{PV}, ..., i_{PVII} reprezintă *starea* încărcată/descărcată a macaralelor.

În aceeași manieră pot fi interpretați T-invarianții rețelei. Astfel, în Figura 4-9(b):

- T-invarianții i_{TI}, ..., i_{TIII} reprezintă *deplasarea* macaralelor.

- T-invarianţii $\mathbf{i}_{TIV}$, ..., $\mathbf{i}_{TVI}$ reprezintă acţiunile de încărcare/descărcare container.

Din expresia invarianţilor se poate observa că reţeaua posedă un T-invaraint positiv, condiţie necesară pentru viabilitatea reţelei. Deoarece această condiţie nu este şi suficientă trebuie procedat la o analiză suplimentară a reţelei. Această analiză este ajutată de subreţele induse prin T-invariaţi. În cazul de faţă se verifică uşor că fiecare dintre aceste subreţele, la marcajul iniţial dat, este reversibilă şi deoarece toate tranziţiile rămân concesionabile reţeaua este şi viabilă.

În afară de calea comună de rulare, *containerele* respectiv *locurile de depozitare libere* reprezintă deasemenea resurse partajate. Într-un prim pas acestea pot fi reprezentate în modelul de reţea Petri prin locaţii conectate la tranziţiile care modelează acţiunile de încărcare respectiv de descărcare. Astfel, având în vedere şi Figura 4-5:

- locaţia p_{23} reprezintă trenul nr. 1, încărcat cu 8 containere, iar
- locaţia p_{24} reprezintă trenul nr. 2, în care trebuie transferate containerele.

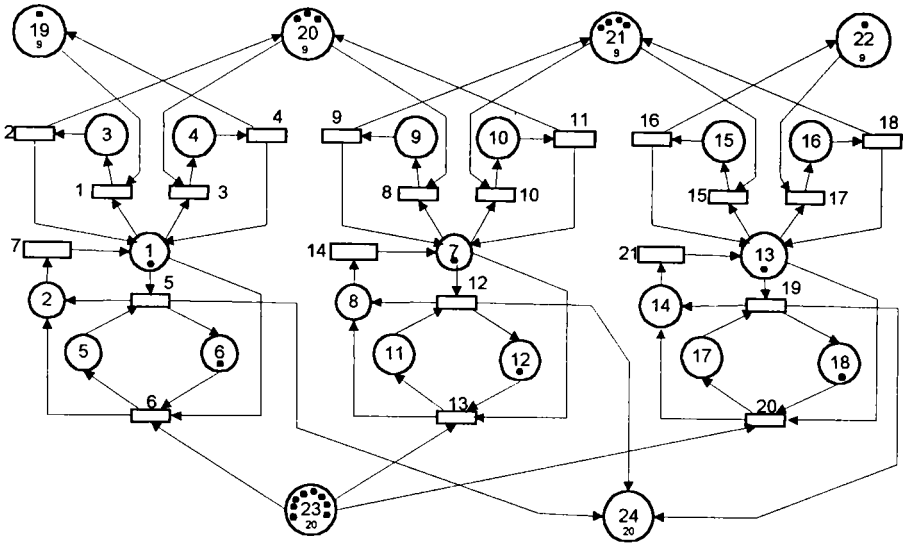


Fig. 4-10 Modelul de reţea Petri al staţiei de containere din Figura 4-5

Un ciclu complet de încărcare/descărcare a unei macarale transferă un marcaj din locaţia p_{23} în locaţia p_{24}. Modificarea de structură a reţelei va schimba şi invarianţii acesteia.

P-invarianţii reţelei sunt daţi de următoarea expresie:

$$i_P = \begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\end{bmatrix} + P1\begin{bmatrix}0\\0\\1\\1\\0\\0\\0\\0\\1\\1\\0\\0\\0\\1\\1\\0\\0\\1\\1\\1\\1\\0\\0\end{bmatrix} + P2\begin{bmatrix}1\\1\\1\\1\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\end{bmatrix} + P3\begin{bmatrix}0\\0\\0\\0\\0\\0\\1\\1\\1\\1\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\end{bmatrix} + P4\begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\1\\1\\1\\1\\0\\0\\0\\0\\0\\0\\0\\0\\0\end{bmatrix} + P5\begin{bmatrix}0\\0\\0\\0\\1\\1\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\end{bmatrix} + P6\begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\1\\1\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\end{bmatrix} + P7\begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\1\\1\\0\\0\\0\\0\\0\\0\end{bmatrix} + P8\begin{bmatrix}0\\0\\0\\0\\1\\0\\0\\0\\0\\0\\1\\0\\0\\0\\0\\0\\1\\0\\0\\0\\0\\0\\1\\1\end{bmatrix}$$

T-invarianții rețelei sunt dați de următoarea expresie:

$$i_T = \begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\end{bmatrix} + P1\begin{bmatrix}1\\1\\1\\1\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\end{bmatrix} + P2\begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\0\\1\\1\\1\\1\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\end{bmatrix} + P3\begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\end{bmatrix}$$

Se poate observa că integrarea în model a trenurilor are ca efect apariția unui P-invariant suplimentar și pierderea a trei T-invarianți. Pentru o interpretare sugestivă acești invarianți se reprezintă pe modelul de rețea Petri conform Figurii 4-11.

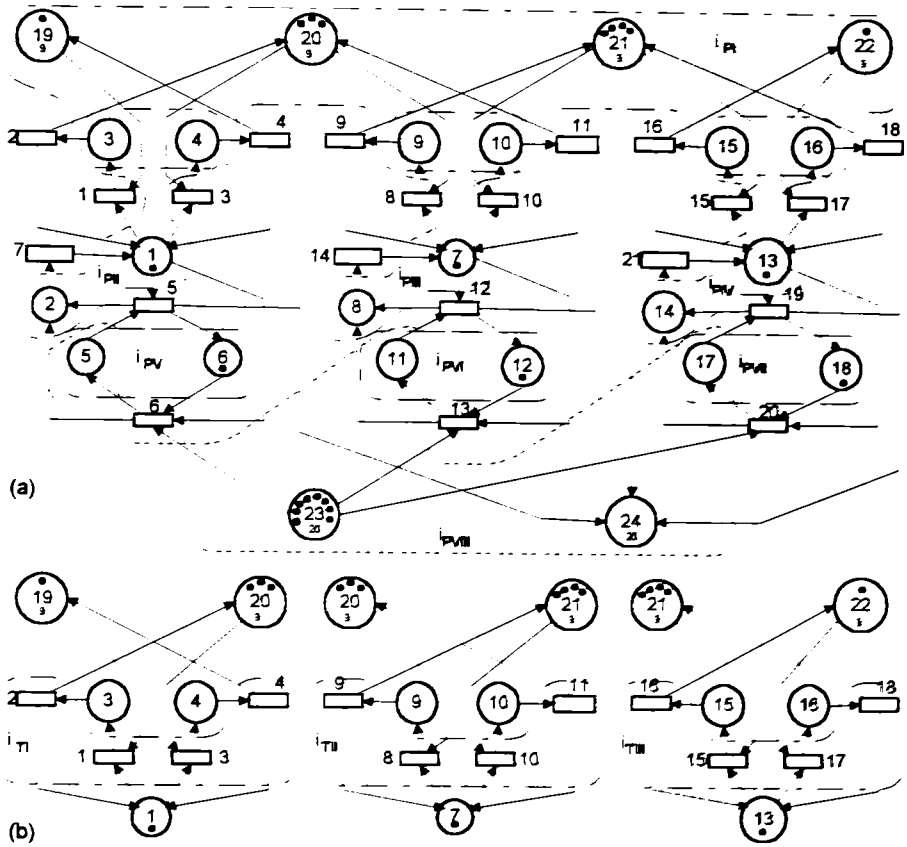


Fig. 4-11 Invarianții rețelei din Figura 4-10

Din Figura 4-11 se pot observa diferențele de comportament dinamic față de rețeaua din Figura 4-8. Aceste diferențe se manifestă prin:

- P-invariantul suplimentar i_{PVIII} corespunzător containerelor care se găsesc fie pe trenul nr. 1, fie pe o macara, fie pe trenul nr. 2.
- Pierderea T-invarianților i_{TIV}, ..., i_{TVI} și implicit pierderea subrețelelor reversibile corespunzătoare activităților de încărcare/descărcare.

Pe baza invarianților se poate decide inmediat că rețeaua nu mai este viabilă deoarece nu posedă nici un T-invariant pozitiv. Raționamente similare celor de mai sus arată că în rețeaua dată nu vor exista blocaje totale deoarece restul subrețelelor din Figura 4-11, la marcajul inițial dat, sunt reversibile. După transferul ireversibil al tuturor containerelor se instalează o situație de blocaj parțial deoarece, deși nu se mai pot executa acțiuni de transfer ale containerelor, deplasarea macaralelor este în continuare posibilă.

Acest model de rețea Petri este utilizabil cu condiția ca înaintea inițierii unui proces de transfer a containerelor să fie inițializate locațiile p_{23} și p_{24}.

4.5 Concluzii

S-a realizat o sinteză a principalelor metode de analiză a rețelelor Petri, analiză necesară pentru a pune în evidență caracteristicile dinamice definitorii ale unor rețele ce modelează sisteme de automatizări industriale.

Analiza mPTN poate fi realizată pe două căi fundamental diferite.

(i) *Analiza bazată pe graful de accesibilitate* presupune construirea, pe baza mulțimii accesibile, a grafului de accesibilitatel și studiul structurii acestuia. Structurarea se realizează pe baza noțiunilor de *graf cu coeziune puternică*, *subgraf cu coeziune maximală* și *condensata grafului accesibil*. Pe baza condensatei grafului accesibil se poate decide dacă rețeaua este reversibilă, viabilă sau dacă prezintă doar o viabilitate slabă.

(ii) *Analiza bazată pe invarianți* furnizează informații referitoare la comportamentul dinamic al rețelei, prin calcule efectuate pe baza matricei de incidențe a rețelei. Pe baza T-invarianților se pot formula condiții necesare pentru reversibilitatea și viabilitatea rețelei. Pe baza P-invarianților se pot formula condiții suficiente pentru mărginire. *Pe baza invarianților nu se pot formula condiții necesare și suficiente.*

Pentru exemplificarea tehnicilor de analiză s-a prezentat exemplul unei stații de containere.

5. Sinteza corecţiilor de reţea

Acest capitol prezintă o metodă de realizare a corecţiilor de reţea. Corecţiile de reţea sunt necesare în cazul în care analiza reţelei nu a confirmat existenta proprietăţilor dinamice impuse reţelei. Corecţiile se realizează prin bucle de reţea suplimentare prin care se realizează restricţii suplimentare de concesionare. În acest sens, e prezintă o metodă de determinare a locaţiilor semnificative la care se vor conecta buclele de reţea. Se prezintă şi o metodă de verificare a corectabilităţii reţelei.

5.1 Necesitatea corecţiilor de reţea

Prin analiza mPTN sunt investigate şi demonstrate anumite proprietăţi ale acestora. În special reversibilitatea şi viabilitatea sunt acele proprietăţi prin care se apreciază caracteristicile dinamice ale mPTN şi prin care se pot formula cerinţe în privinţa comportării dinamice a sistemului tehnic modelat prin reţeaua Petri. Lipsa reversibilităţii arată că sistemul poate ajunge într-o stare din care sunt accesibile doar o parte din stările iniţiale ale sistemului. În mod analog în cazul lipsei viabilităţii există pericolul ca sistemul să ajungă într-o stare din care să nu mai fie executabile toate tranziţiile.

În privinţa interpretării rezultatelor analizei, indiferent dacă s-a modelat doar procesul condus sau în model a fost cuprins şi sistemul de comandă, lipsa proprietăţilor de reversibilitate şi viabilitate arată faptul că nu poate fi garantată comportarea dorită în toate situaţiile posibile. Mai mult, din cauza lipsei sau a incompatibilităţii sistemului de comandă, sistemului modelat îi ramân prea multe grade de libertate şi prin urmare sistemul poate ajunge în stări nedorite. Comportarea nedorită a sistemului, recunoscută prin analiza reţelei Petri care-l modelează, trebuie corectată, prin restrângerea gradelor de libertate (prin restricţii), cu ajutorul unui sistem de comandă adecvat, astfel încât să se excludă ajugerea în anumite stări nedorite.

În cazul în care restricţiile necesare pot fi formulate prin sintaxa reţelelor Petri se ajunge la un model unitar al sistemului modelat care poate fi, din nou, supus unui proces de analiză. Dacă modificările necesare sunt *extinderi* ale reţelei originale elementele noi introduse pot fi concepute ca structuri de conducere elementare. Acestea asigură o comportare corectă a sistemului şi astfel, pot sta la baza altor procedee de analiză (de exemplu în privinţa optimizării).

În continuare se va prezenta un procedeu de corecţie a mPTN care permite asigurarea proprietăţilor de reversibilitate şi viabilitate.

5.2 Interpretarea rezultatelor analizei

5.2.1 Exemplul unor procese paralele

Procedeul prezentat utilizează analiza bazată pe grafuri deoarece numai aceasta pune la dispoziţie toate informaţiile legate de comportarea nedorită a sistemului modelat. Ideea de bază a procedeului se prezintă pe baza unei reţele care poate ajunge într-o situaţie de

blocare. Figura 5-1 prezintă reţeaua Petri a unui proces în care două subprocese A şi B, utilizează două resurse partajate I şi II, pe care le solicită, în sens contrar, pentru utilizare exclusivă.

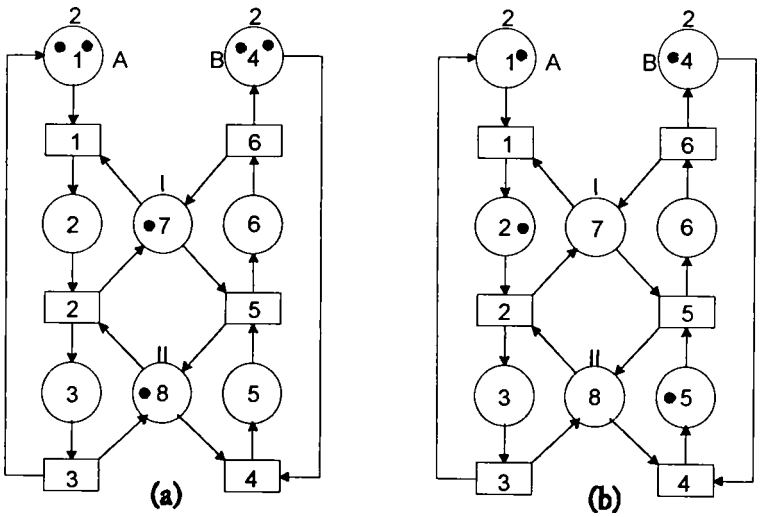


Fig. 5-1 Exemplu de mPTN a unor procese concurente

Se presupune în continuare, pentru a limita complexitatea grafului de accesibilitate, că fiecărui subproces îi stau la dispoziţie două obiecte de prelucrat (de exemplu semifabricate) şi că fiecare obiect complet prelucrat este instantaneu înlocuit cu un nou obiect de prelucrat.

Figura 5-1(a) prezintă starea iniţială a sistemului precum şi starea de blocaj total în care poate ajunge sistemul (Figura 5-1(b)). În această stare fiecare subproces aşteaptă după o resursă pe care o ocupă celălalt subproces.

Blocajul total apare în graful de accesibilitate ca un marcaj mort $\mathbf{m}_m$, care corespunde unui nod de absorbţie mort în condensata grafului accesibil aşa cum se prezintă în Figura 5-2.

Condensata grafului accesibil prezintă în mod sugestiv acţiunile ce se cer intreprinse pentru corectarea reţelei. În exemplul prezentat este iminent faptul că sunt nedorite acele tranziţii de stare care conduc sistemul în componenta tare K2. Aceste tranziţii corespund unor execuţii critice ale tranziţiilor t1 şi t4.

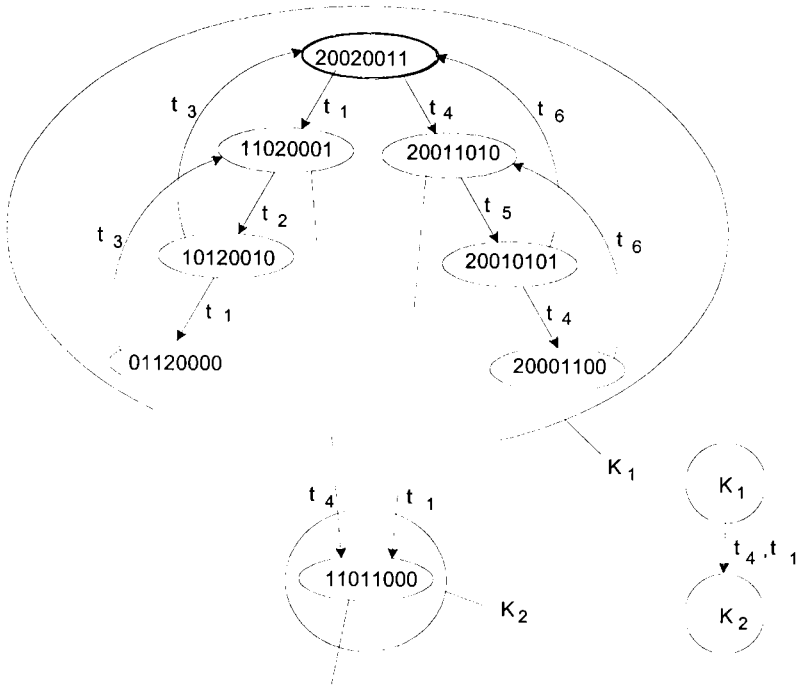


Fig. 5-2 Graful de accesibilitate şi condensata reţelei din Figura 5-1

Deoarece graful de accesibilitate conţine mai multe arce inscripţionate cu t_1 respectiv t_4 interzicerea generală a execuţiei acestor tranziţii ar fi prea restrictivă. Scopul sintezei corecţiilor de reţea este acela de a face distincţie clară între tranziţiile critice şi cele permise şi de a le interzice numai pe cele critice. O modificare corespunzătoare a structurii reţelei exclude marcajul mort din mulţimea accesibilă şi în acelaş timp toate celelalte marcaje rămân accesibile. Este necesar a se completa structura reţelei cu elemente care pot influenţa concesionabilitatea tranziţiilor fără a modifica fluxul de marcaje. Buclele autonome de reţea, aşa cum au fost introduse în capitolul 3.3, Definiţia 3-27, satisfac această cerinţă şi pot fi utilizate pentru a exclude anumite marcaje din mulţimea accesibilă.

5.2.2 Deducerea unor restricţii suplimentare de concesionare

Proprietăţile dinamice ale unei mPTN se reflectă în condensata GA^K a grafului accesibil şi prin urmare proprietăţile impuse sistemului modelat determină caracteristicile condensatei. Deoarece corecţia reţelei trebuie să se realizeze doar prin introducerea unor restricţii suplimentare de concesionare ca urmare a aplicării acestora se va ajunge la un subgraf GA'^K care se obţine din condensata originală GA^K prin eliminarea unor noduri (şi implicit a unor arce). Prin condensata modificată GA'^K se specifică şi graful de accesibilitate corespunzător GA' care la rîndul lui este un subgraf al grafului accesibil GA.

În concluzie trebuie să se găsească acea mPTN', notată N', care, în urma analizei prin metoda grafurilor, să furnizeze graful de accesibilitate GA' având condensata GA'^K.

Condensata modificată se specifică având în vedere scopurile urmărite prin sinteză. Dacă reţeaua corectată N' trebuie să prezinte o viabilitate tare atunci condensata GA'^K nu are voie să conţină noduri de absorbţie moarte sau cu viabilitate slabă. Dacă se doreşte o reţea cu viabilitate slabă se vor exclude doar nodurile de absorbţie moarte. Plecând de la condensata originală GA^K prin eliminarea succesivă a nodurilor problematice se ajunge la condensata GA'^K. Dacă se doreşte asigurarea reversibilităţii se va accepta doar nodul sursă al condensatei originale.

Tranziţiile a căror execuţie în reţeaua Petri originală poate să ducă sistemul în stări nedorite se vor numi tranziţii critice conform următoarei definiţii:

Definiţia 5-1 (Tranziţie critică)

Fie N o mPTN, GA=(Q,A,τ) graful de accesibilitate al reţelei N şi GA'=(Q',A',τ') un subgraf al grafului GA. O tranziţie $t_j \in T$ se numeşte tranziţie critică dacă GA conţine un arc etichetat cu t_j al cărui nod de absorbţie se găseşte în complementul submulţimii GA' faţă de GA iar nodul sursă în GA∩GA', adică dacă:

$$\exists a=(\mathbf{m},\mathbf{m}') \in A(GA): \tau(a)=t_j \wedge \mathbf{m} \in Q(GA') \wedge \mathbf{m}' \notin Q(GA').$$

Deci, pentru fiecare tranziţie critică se găseşte cel puţin un marcaj accesibil la care t_j este concesionată dar nu ar trebui să se execute. Pe de altă parte se găsesc şi marcaje pentru care t_j trebuie să se execute. Din acest motiv se impune o analiză suplimentară a grafurilor accesibile GA şi GA'. Trebuie selectate toate marcajele la care este concesionată o anumită tranziţie şi clasificate conform următoarei definiţii:

Definiţia 5-2 (Vector concesionant/Vector deconcesionant)

Fie N o mPTN, GA=(Q,A,τ) graful de accesibilitate a reţelei N şi GA'=(Q',A',τ') un subgraf al grafului GA şi fie $t_j \in T$ o tranziţie critică a reţelei N.

(1) Un marcaj $C^j \in Q(GA')$ se numeşte *concesionanta* tranziţiei t_j dacă tranziţia critică t_j este concesionată de către C^j şi marcajul rezultat este conţinut în GA∩GA' ; adică dacă:

$$\exists a=(C^j,C'^j) \in A(GA): \tau(a)=tj \wedge C^j \in Q(GA') \wedge C'^j \in Q(GA').$$

(2) Vectorul corespunzător acestui marcaj $\mathbf{c}^j \in N^{|P|}$ se numeşte *vector concesionant* pentru tranziţia t_j.

(3) Un marcaj $D^j \in Q(GA')$ se numeşte *deconcesionanta* tranziţiei t_j dacă tranziţia critică t_j este concesionată de către D^j şi marcajul rezultat nu este conţinut în GA'; adică dacă:

$$\exists a=(D^j,D'^j) \in A(GA): \tau(a)=t_j \wedge D^j \in Q(GA') \wedge D'^j \notin Q(GA').$$

(4) Vectorul corespunzător acestui marcaj $\mathbf{d}^j \in N^{|P|}$ se numeşte *vector deconcesionant* pentru tranziţia t_j.

Mulţimea marcajelor la care o tranziţie critică t_j este concesionată este subdivizată în două mulţimi disjuncte a marcajelor concesionante respectiv deconcesionante. Scopul corecţiei reţelei este de a sustrage concesionabilitatea unei tranziţii critice la apariţia unui marcaj deconcesionant şi de a păstra concesionabilitatea la apariţia unui marcaj concesionant. Celor de mai sus le corespund următoarele relaţii:

$\mathbf{m} = \mathbf{c}_1^j \vee \mathbf{m} = \mathbf{c}_2^j \vee \ldots \Rightarrow$ concesionează t_j

$\mathbf{m} = \mathbf{d}_1^j \vee \mathbf{m} = \mathbf{d}_2^j \vee \ldots \Rightarrow$ deconcesionează t_j. (5.2-1)

Marcajele care nu concesionează tranziţia critică dată respectiv marcajele neaccesibile pot fi alocate atât mulţimii marcajelor concesionante cât şi celor deconcesionante.

Pe baza definiţiilor 5-1 şi 5-2 poate fi formalizat procedeul de sinteză a corecţiilor de reţea. Acesta se va exemplifica pe reţeaua prezentată în paragraful 5.2.1.

În primul rând se vor specifica grafurile accesibile GA şi GA' conform scopului sintezei. Pentru reţeaua dată, care nu este nici viabilă şi nici reversibilă, dacă se doreşte asigurarea atât a viabilităţii cât şi a reversibilităţii va trebui eliminată componenta tare K2 (Figura 5-3(a)). Se obţine astfel subgraful GA' care conţine doar componenta tare K1 a grafului original (Figura 5-3(b)). O confruntare a grafurilor GA respectiv GA' arată existenţa a două tranziţii critice t1 şi t4 . Acestea vor fi tratate separat.

În continuare va fi tratată tranziţia t_1. Se observă că aceasta are un vector deconcesionant;

$\mathbf{d}_1^1=(2,0,0,1,1,0,1,0)^T$ (5.2-2)

şi doi vectori concesionanţi;

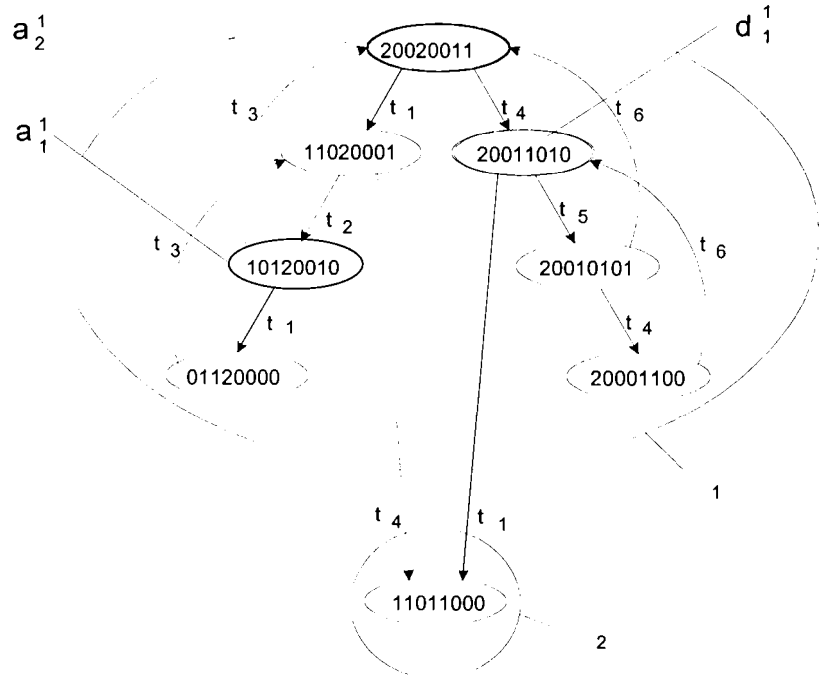
$\mathbf{c}_1^1=(2,0,0,2,0,0,1,1)^T$ şi $\mathbf{c}_2^1=(1,0,1,2,0,0,1,0)^T$ (5.2-3)

Nodurile şi arcele corespunzătoare din figura 5.2-3 au fost inscripţionate ca atare. Cele de mai sus conduc la următoarele prescripţii de corecţie:

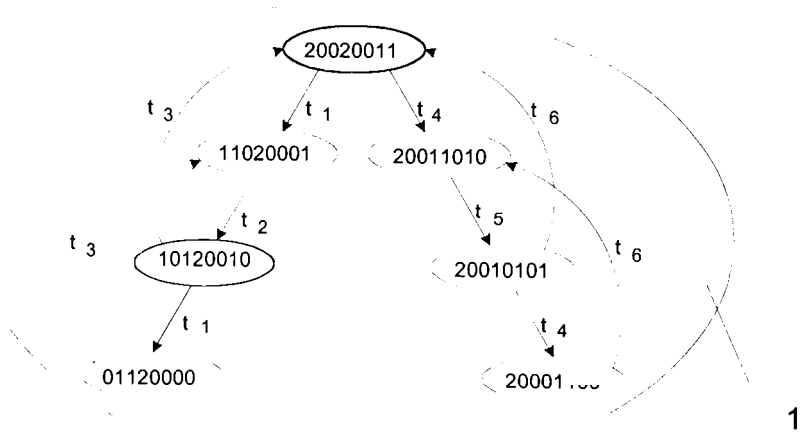
$\mathbf{m} = \mathbf{c}_1^1 \vee \mathbf{m} = \mathbf{c}_2^1 \Rightarrow$ concesionează t_1

$\mathbf{m} = \mathbf{d}_1^1 \Rightarrow$ deconcesionează t_1 . (5.2-4)

în care sunt conţinute condiţiile de concesionare pentru t_1. Pentru ca cele de mai sus să poată fi realizate prin elementele reţelei Petri relaţiile (5.2-4) trebuie exprimate prin marcajele locaţiilor reţelei Petri astfel:



a) Graful accesibil GA cu componentele tari K1 si K2



b) Graful accesibil GA'cu componenta tare K1

Fig. 5-3 Grafurile accesibile ale exemplului prezentat

$[M(p1)=2 \wedge M(p2)=0 \wedge M(p3)=0 \wedge M(p4)=2 \wedge$

$M(p5)=0 \wedge M(p6)=0 \wedge M(p7)=1 \wedge M(p8)=1] \vee$

$[M(p1)=1 \wedge M(p2)=0 \wedge M(p3)=1 \wedge M(p4)=2 \wedge$

$M(p5)=0 \wedge M(p6)=0 \wedge M(p7)=1 \wedge M(p8)=0] \Rightarrow$ concesionează t_1

$[M(p1)=2 \wedge M(p2)=0 \wedge M(p3)=0 \wedge M(p4)=1 \wedge$

$M(p5)=1 \wedge M(p6)=0 \wedge M(p7)=1 \wedge M(p8)=0] \Rightarrow$ deconcesionează t_1

Se poate observa că dacă se doreşte tratarea tranziţiilor critice conform scopului sintezei trebuie testat numărul curent al marcajelor $M(p_i)$ pentru fiecare locaţie p_i. Acest lucru presupune amendamente complexe la reţeaua originală. Dar, în situaţia în care răspunzătoare pentru tranziţia nedorită sunt doar unele stări locale acest lucru trebuie să se reflecte în existenţa unor *locaţii semnificative*.

Căutarea locaţiilor semnificative se reduce la o problemă clasică de minimizare în algebra Booleană. În acest sens se alocă fiecărei tranziţii critice t_j o variabilă binară B^j cu următoarea semnificaţie:

$B^j = 1 \Leftrightarrow$ se permite execuţia tranziţiei t_j

$B^j = 0 \Leftrightarrow$ nu se permite execuţia tranziţiei t_j (5.2-6)

În acest fel C^j specifică modul cum se tratează tranziţia t_j în reţeaua corectată. În continuare este necesară codificarea numărului de puncte în fiecare locaţie $p_i \in P$, $0 \le M(p_i) \le K(p_i)$ astfel încât fiecare marcaj posibil $M:P \rightarrow N$ să poată fi reprezentată ca o combinaţie a unor variabile binare $E_0, E_1, E_2, \ldots, E_k$. Este astfel posibilă transcrierea prescripţiei de corectare exprimată prin relaţia 5.2-1 într-o ecuaţie Booleană de forma:

$$B^j = f(E_0, E_1, E_2, \ldots, E_k) \qquad (5.2\text{-}7)$$

care trebuie minimizată (de exemplu prin diagrama KARNAUGH-VEITCH).

Deşi prin această metodă pot fi determinate locaţiile semnificative, în cazul unui număr mare de locaţii sau a unui număr mare de puncte pe locaţie metoda devine ineficienă (sunt cuprinse şi marcajele neaccesibile !). Din acest motiv se prezintă mai jos un procedeu de determinare a marcajelor semnificative care extrage informaţia necesară din concesionanţii şi deconcesionanţii tranziţiilor critice.

La baza procedeului stă o matrice-diferenţă care se obţine prin scăderea fiecărui vector concesionant din toţi vectorii deconcesionanţi, conform următoarei definiţii:

Definiţia 5-3 (Matrice-diferenţă)

Fie t_j o tranziţie critică şi fie $\{\mathbf{d}_1^j, \mathbf{d}_2^j, \ldots, \mathbf{d}_p^j\}$ şi $\{\mathbf{c}_1^j, \mathbf{c}_2^j, \ldots, \mathbf{c}_q^j\}$ mulţimea tuturor vectorilor deconcesionanţi respectiv concesionanţi pentru tranziţia t_j. Matricea $\Delta^j = [|\mathbf{d}_1^j - \mathbf{c}_1^j|: \ldots :|\mathbf{d}_1^j - \mathbf{c}_q^j|:|\mathbf{d}_2^j - \mathbf{c}_1^j|: \ldots |\mathbf{d}_p^j - \mathbf{c}_q^j|]$ se numeşte *matricea diferenţă a tranziţiei* t_j.

Un element nenul al liniei i al acestei matrici arată că pe locaţia p_i numărul punctelor în vectorul concesionant este diferit de cel al vectorului deconcesionant. Dacă toate elementele liniei i sunt diferite de zero locaţia p_i este o *locaţie semnificativă*. În caz contrar se va încerca formarea unei linii nenule prin combinaţia liniară a unui număr (minim) de linii. Indicii liniilor care participă la realizarea liniei nenule specifică locaţiile semnificative.

Pentru exemplificare se consideră aceaşi reţea. Pentru tranziţia critică t_1 având în vedere vectorii deconcesionanţi şi concesionanţi

$$\mathbf{d}_1^1 = [2,0,0,1,1,0,1,0]^T$$
$$\mathbf{c}_1^1 = [2,0,0,2,0,0,1,1]^T$$
$$\mathbf{c}_2^1 = [1,0,1,2,0,0,1,0]^T \qquad (5.2\text{-}8)$$

se obţine matricea diferenţă

$$\Delta^1 = [|\mathbf{d}_1^1 - \mathbf{c}_1^1| : |\mathbf{d}_1^1 - \mathbf{c}_2^1|] = \begin{bmatrix} 0 & 1 \\ 0 & 0 \\ 0 & 1 \\ 1 & 1 \\ 1 & 1 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \end{bmatrix} \qquad (5.2\text{-}9)$$

Se observă prezenţa a două linii nenule. Aceasta arată că locaţiile p4 şi p5 sunt locaţii semnificative. Ca urmare ajunge cunoaşterea uneia dintre marcajele M(p4) sau M(p5) pentru a diferenţia vectorul concesionant de cel deconcesionant (v. rel. 5.2-8). O altă posibilitate ar fi considerarea liniilor 1 şi 8 respectiv a marcajelor M(p1) şi M(p8).

Această metodă, a combinaţiilor liniare, poate fi aplicată în cazuri simple. În cazul unor reţele ample devine însă inaplicabilă. O soluţie elegantă constă în formularea problemei sub forma unui sistem de ingalităţi, astfel:

$$(\Delta^j)^T \bullet \mathbf{x} > \mathbf{0} \text{ cu } \Delta^j[k,l] \in N,\ \mathbf{x}[k] \in N \qquad (5.2\text{-}10).$$

Fiecare soluţie $\mathbf{x}$ reprezintă o sumă ponderată de vectori linie ai lui Δ^j a cărui rezultat nu mai are elemente nule (conform condiţiei >0). Vectorii linie care nu intră în această sumă, adică cei ponderaţi prin $x[i]=0$, specifică vectorii care corespund locaţiilor nesemnificative. În [Sei'91] se prezintă un procedeu de rezolvare a sistemului de inegalităţi (5.2-10).

Pentru exemplul de mai sus se obţine următoarea inegalitate:

$$\begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \end{bmatrix} \cdot \mathbf{x} > \begin{bmatrix} 0 \\ 0 \end{bmatrix},\ \mathbf{x}[i] \in N.$$

Vectorii

$$\mathbf{x}_1^T = [0\ 0\ 0\ 1\ 0\ 0\ 0\ 0]$$
$$\mathbf{x}_2^T = [0\ 0\ 0\ 0\ 1\ 0\ 0\ 0]$$
$$\mathbf{x}_3^T = [1\ 0\ 0\ 0\ 0\ 0\ 0\ 1]$$

sunt soluţii ale inegalităţii de mai sus. Elementele nenule ale acestor vectori arată că locaţiile p_4 şi p_5 sunt locaţii semnificative. Deasemnea combinaţia de locaţii p_1 şi p_8 este semnificativă. S-a obţinut rezultatul determinat pe calea elementară.

5.2.3 Testarea corectabilităţii

Cu toate că analiza bazată pe grafuri este absolut necesară în sinteza corecţiilor, metodele algebrice de analiză îşi au şi ele utilitatea lor. Cu ajutorul acestora se poate verifica dacă *corecţia prin restricţii* este realizabilă sau structura dată a reţelei o exclude din principiu.

Pentru exemplificare se va utiliza o variantă uşor modificată a reţelei din Figura 5-1. Sunt date două procese paralele care utilizează două resurse comune în ordine inversă. În acest caz însă fiecare proces prelucrează doar un singur obiect. Reţeaua Petri corespunzătoare şi graful de accesibilitate s-au reprezentată în Figura 5-4.

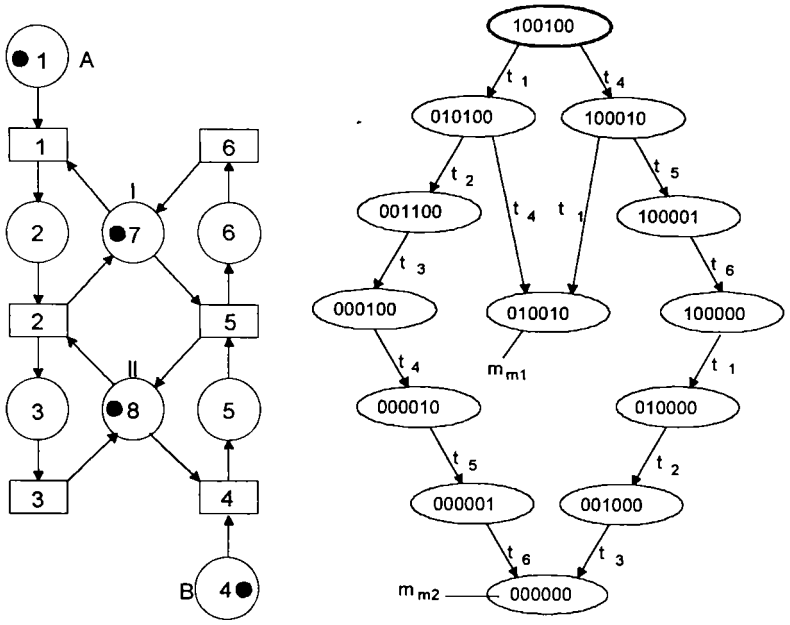


Fig. 5-4 mPTN şi graful ei accesibil

Graful de accesibilitate conţine două marcaje moarte $\mathbf{m}_{m1}$ şi $\mathbf{m}_{m2}$. Marcajul $\mathbf{m}_{m1}$ poate fi exclusă din mulţimea accesibilă prin introducerea de restricţii suplimentare pentru tranziţiile t_1 şi t_2. În cazul marcajului $\mathbf{m}_{m2}$ însă, acest procedeu ar conduce la noi marcaje moarte. Acest lucru scoate în evidenţă faptul că procesul modelat nu conţine procese repetabile şi prin urmare corecţia prin restricţii nu este aplicabilă.

Procedeul de corecţie bazat exclusiv pe restricţii suplimentare de concesionare nu este aplicabil în toate cazurile. În cazul unor modele incomplete acest procedeu poate produce noi marcaje moarte. Rezolvare ar aduce doar interzicerea totală a anumitor tranziţii. Astfel de cazuri pot fi puse în evidenţă prin metode algebrice de analiză. Se evită în acest fel execuţia unei analize nereuşite bazate pe grafuri. În locul acesteia se execută o verificare a modelului însăşi.

Prin calculul invarianţilor se poate extrage informaţie din structura reţelei Petri referitoare la posibilitatea corecţiei. Teoremele 3.4 şi 3.5 specifică condiţiile necesare pentru

asigurarea reversibilităţii respectiv viabilităţii reţelei. În acelaş fel pot fi formulate condiţii prin care se exclude existenţa proprietăţilor de reversibilitate şi viabilitate.

Teorema 5-1 (Ireversibilitate/Neviabilitate)

(1) Dacă o mPTN nu conţine nici un T-invariant nenegativ ea nu poate fi reversibilă, adică:

$\neg\exists \mathbf{i}_T \in \mathcal{N}^{|T|}$: $\mathbf{N} \bullet \mathbf{i}_T = \mathbf{0} \Rightarrow$ mPTN nu este reversibilă.

(2) Dacă o mPTN mărginită nu conţine nici un T-invariant strict pozitiv ea nu poate să fie viabilă, adică:

$\neg\exists \mathbf{i}_T \in \mathcal{N}'^{|T|}$: $\mathbf{N} \bullet \mathbf{i}_T = \mathbf{0} \Rightarrow$ mPTN nu este viabilă.

Teorema 5.1 se referă la reţeaua Petri originală. Pentru a verifica eficienţa unei proceduri de corecţie aceeaşi teoremă trebuie aplicată şi reţelei corectate N'. Deoarece corecţia reţelei se execută prin bucle suplimentare prezente doar în matricile $\mathbf{N}'^{+}$ respectiv $\mathbf{N}'^{-}$ dar care nu modifică matricea N a reţelei teorema 5.1 este aplicabilă fără restricţii şi reţelei corectate.

Teorema 5-2 (Transmiterea invarianţilor)

Fie N o mPTN şi N' o mPTN modificată doar prin bucle suplimentare $(p_i,t_j) \in PxT$ cu $W(p_i,t_j) = W(t_j,p_i)$

(1) Orice invariant al reţelei N este şi invariant a reţelei N' .

(2) Orice invariant al reţelei N' este şi invariant a reţelei N .

Din acest motiv teorema 5.1 poate fi aplicată direct reţelei corectate utilizând T-invarianţii reţelei originale. Teorema 5.1 se poate reformula astfel:

Teorema 5-3 (Ireversibilitate/Neviabilitate)

(1) O mPTN care nu conţine nici un T-invariant nenegativ nu este reversibilă şi nici nu poate fi făcută reversibilă prin introducerea de restricţii suplimentare de concesionare prin bucle de reţea.

$\neg\exists \mathbf{i}_T \in \mathcal{N}^{|T|}$: $\mathbf{N} \bullet \mathbf{i}_T = \mathbf{0} \Rightarrow$ N' nu este reversibilă

(2) O mPTN care nu conţine nici un T-invariant strict pozitiv nu este viabilă şi nici nu poate fi făcută viabilă prin introducerea de restricţii suplimentare de concesionare.

$\neg\exists \mathbf{i}_T \in \mathcal{N}'^{|T|}$: $\mathbf{N} \bullet \mathbf{i}_T = \mathbf{0} \Rightarrow$ N' nu este viabilă.

În concluzie, dacă o anumită proprietate dinamică nu este prezentă într-o reţea Petri ea nu poate fi realizată prin introducerea de restricţii de concesionare suplimentare realizate prin bucle de reţea. Prin restricţii suplimentare de concesionare nu poate fi asigurată viabilitatea reţelei. Din acest motiv trebuie verificată corectitudinea modelării.

Pentru reţeaua Petri modificată din Figura 5-4 se obţin următorii invarianţi:

$\mathbf{i}_T^T$ = [00000000] ;

$\mathbf{i}_P^T$ = [00000000] + $\lambda 1 \bullet$ [01000110] + $\lambda 2 \bullet$ [00101001], cu $\lambda 1, \lambda 2 \in Z$.

P-invarianţii arată că nu este satifăcută condiţia suficientă penru mărginirea reţelei deoarece componenta întâia şi a patra a fiecărui vector component este nulă. Pe baza

grafului accesibil se verifică că totuşi reţeaua este mărginită. Se observă prin aplicarea teoremei 5-3 că reţeaua nu este corectabilă şi deci modelul trebuie corectat.

Pentru reţeaua din Figura 5-1 se obţin invarianţii:

i_T^T = [000000] + $\mu 1$ • [111000] + $\mu 2$ • [000111] , cu $\mu 1, \mu 2 \in Z$,

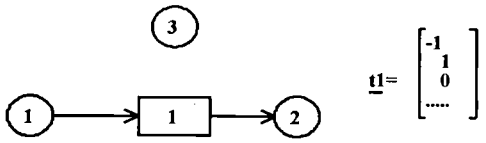
i_P^T = [00000000] + $\lambda 1$ • [01000110] + $\lambda 2$ • [00101001] + $\lambda 3$ • [11100000] +

+ $\lambda 4$ • [00011100] , cu $\lambda 1, \lambda 2, \lambda 3, \lambda 4 \in Z$.

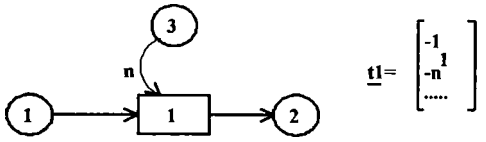
Printr-o simplă adunare a vectorilor componenţi se poate ajunge la invarianţi pozitivi P şi T. În consecinţă reţeaua este mărginită şi nu sunt excluse proprietăţile de reversibilitate şi viabilitate.

5.3 Realizarea restricţiilor suplimentare de concesionare

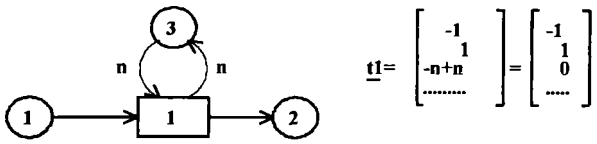
În ultima fază a realizării corecţiei de reţea, restricţiile suplimentare de concesionare deduse conform scopului urmărit trebuie transpuse în structuri de reţea. Realizarea acestora se bazează pe Definiţia 3-27 a buclelor autonome de reţea. În Figura 5-5 se prezintă o parte a unei mPTN împreună cu vectorul de tranziţie $t_1 = t_1^+ + t_1^-$ în diversele faze în procesul de completare a reţelei.



a) Nici o influenta a lui p3 asupra lui t1



b) p3 influenteaza concesionabilitatea si executia tranzitiei t1



c) p3 influenteaza numai concesionabilitatea

Fig. 5-5 Fazele de realizare ale unei bucle de reţea

În primul caz (Figura. 5-5(a)) se prezintă starea iniţială în care marcajul locaţiei p_3 nu are nici o influenţă asupra execuţiei tranziţiei t_1. În cazul în care această locaţie este introdusă în premulţimea tranziţiei t_1 aceasta se poate executa doar dacă este satisfăcută condiţia $M(p_3) \geq n$. Dealtfel, această modificare influenţează şi fluxul de marcaje. În vectorul tranziţiei t_1 apare o componentă nenulă pe poziţia elementului trei (Figura 5-5(b)). Pentru a evita acest lucru se completează nodul dintre p_3 şi t_1 (Figura 5-5(c)).

În concluzie, restricţii elementare de concesionare de forma:

$[M(p_i) \geq n] \Leftrightarrow$ activează t_j (5.3-1)

pot fi realizate printr-o buclă între p_i şi t_j. În reprezentarea vectorială a reţelei modificarea făcută se traduce printr-o nouă valoare a elementelor [i,j] astfel:

$$N^+[i,j]_{nou} = -n \quad (5.3\text{-}2)$$

$$N^-[i,j]_{nou} = n + N^+[i,j] + N^-[i,j] \quad (5.3\text{-}3)$$

Pentru realizarea condiţiei:

$[M(p_i) \leq n] \Leftrightarrow$ activează t_j (5.3-4)

se vor folosi locaţii complementare.

Definiţia 5-4 (Locaţii complementare)

Fie N=(P,T;F,K,W,$\mathbf{m}_0$) o mPTN. Două locaţii p,p'∈P se numesc *locaţii complementare* dacă pentru fiecare arc prin care p se leagă de o tranziţie t∈T exisă un arc de aceeaşi pondere dar cu orientarea inversă care leagă pe p' de t, adică dacă sunt valabile:

(a) $\bullet p = p'\bullet$

(b) $p\bullet = \bullet p'$

(c) $W(t,p) = W(p',t)\ \forall t \in T$ şi

(d) $W(t,p') = W(p,t)\ \forall t \in T$ şi

(e) $M_0(p) + M_0(p') = K(p) = K(p')$.

O mPTN poate fi întotdeauna completată cu o locaţie complementară aşa cum se prezintă în figura 5.2-3.

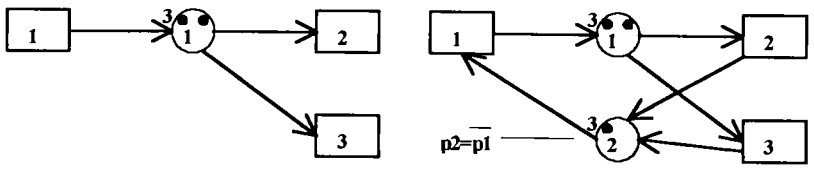


Fig. 5-6 Complementarea unei locaţii

O pereche de locaţii complementare se caracterizează prin faptul că suma marcajelor conţinute în ele M(p)+M(p') este constantă pentru toate marcajele accesibile. Conform definiţiei 5.4 acestă sumă are tocmai valoarea K(p).

Teorema 5-4 (Marcajul locaţiilor complementare)

Fie N=(P,T;F,K,W,M_0) o mPTN cu locaţiile complementare p, p'∈P şi fie $R_N(M_0)$ mulţimea accesibilă a reţelei N. Pentru oricare marcaj accesibil suma marcajelor locaţiilor p şi p' este egală cu capacitatea lui p respectiv p', adică este valabilă următoarea relaţie:

$\forall M \in R_N(M_0)$: $M(p) + M(p') = K(p) = K(p')$.

Demonstraţie: Se va demonstra această teoremă prin inducţie completă.

Fie M' marcajul rezultat după execuţia tranziţiei $t \in T$ din marcajul $M \in R_N(M_0)$. Conform regulei de tranziţie se obţin noile numere de marcaje în locaţiile p şi p', astfel:

$$M'(p) = M(p) - W(p,t) + W(t,p) \quad (5.3\text{-}5)$$

$$M'(p') = M(p') - W(p',t) + W(t,p') \quad (5.3\text{-}6)$$

Conform definiţiei 5.4 numărul marcajelor iniţiale ale locaţiilor p şi p' este:

$$M_0(p) + M_0(p') = K(p) = K(p') . \quad (5.3\text{-}7)$$

Deci teorema este valabilă pentru $M = M_0$.

Presupunem adevărată teorema pentru marcajul M. Deci este adevărată egalitatea:

$$M(p) + M(p') = K(p) = K(p') . \quad (5.3\text{-}8)$$

Trebuie demonstrat că în aceste condiţii este adevărată şi relaţia:

$$M'(p) + M'(p') = K(p) = K(p') . \quad (5.3\text{-}9)$$

Pentru locaţiile complementare sunt valabile, conform definiţiei 5.4,(iii),(iv), următoarele relaţii:

$$W(t,p) = W(p',t) \quad \text{şi} \quad (5.3\text{-}10)$$

$$W(p,t) = W(t,p') \quad (5.3\text{-}11)$$

După înlocuirea în ecuaţia (5.3-5) se obţine:

$$M'(p) = M(p) - W(t,p') + W(p',t) . \quad (5.3\text{-}12)$$

Iar după adunarea ecuaţiilor 5.3-12 şi 5.3-6 se obţine:

$$M'(p) + M'(p') = M(p) + M(p') . \quad (5.3\text{-}13)$$

După înlocuirea ipotezei inducţiei (ecuaţia 5.3-8) se obţine:

$$M'(p) + M'(p') = K(p) = K(p') \quad (5.3\text{-}14)$$

chiar ecuaţia din teormă. □

Constanţa numărului de marcaje în locaţii complementare permite realizarea restricţiei (5.3-4). La o creştere a numărului de marcaje în locaţia p se reduce numărul de marcaje în locaţa p' cu acelaş număr. Se consideră un nod construit cu locaţia p' în locul locaţiei p a cărui arce au ponderea etichetată $W(p_i',t_j)=W(t_j,p_i')=n'$. Influenţa acestui nod asupra execuţiei tranziţiei t_j relativ la $M(p_i')$ se formulează astfel:

$$[M(p_i') \geq n'] \Leftrightarrow \text{activează } t_j . \quad (5.3\text{-}15)$$

Aplicând teorema 5.4 se obţin următoarele:

$$K(p_i) - M(p_i) \geq n' \Leftrightarrow M(p_i) \leq K(p_i) - n' \quad (5.3\text{-}16)$$

Ultima inegalitate este o condiţie formulată prin $M(p_i)$ şi dacă se alege :

$$n' = K(p_i) - n \quad (5.3\text{-}17)$$

se ajunge la condiţia (5.3-4).

Pe baza relaţiilor (5.3-2) şi (5.3-4) pot fi construite condiţionări de concesionare mai complexe. Astfel, următoarele echivalenţe:

$$[M(p_i)=n] \Leftrightarrow [M(p_i)\geq n \wedge M(p_i)\leq n] \text{ şi} \qquad (5.3\text{-}18)$$

$$[M(p_i)\neq n] \Leftrightarrow [M(p_i)\geq n+1 \vee M(p_i)\leq n-1]$$

permit formularea de restricţii care testează egalitatea respectiv inegalitatea numărului de marcaje în ipoteza că se dispune de posibilitatea reprezentării condiţionărilor disjunctive respectiv conjunctive.

Reprezentarea condiţionărilor conjunctive se obţine prin aplicarea consecventă a condiţionărilor componente deoarece tranziţia în cauză se poate executa doar dacă toate condiţionările sunt satisfăcute (din cauza regulei de tranziţie). Figura 5-7 prezintă condiţionări conjuctive între locaţiile p_3 şi p_4.

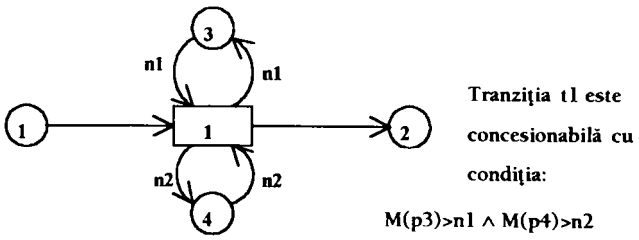


Fig. 5-7 Reprezentarea condiţionărilor conjunctive

În schimb, pentru realizarea condiţionărilor disjunctive este nevoie de secţionarea tranziţiei critice în cauză în două aşa numite tranziţii congruente.

Definiţia 5-5 (Tranziţii congruente)

Fie $N=(P,T;F,K,W,M_0)$ o mPTN. Tranziţiile $t',t''\in T$ se numesc congruente dacă pentru fiecare arc care leagă pe t' de o locaţie $p\in P$ există un arc la fel orientat şi de aceaşi pondere care leagă pe t" de p şi reciproc; adică dacă sunt valabile următoarele :

(i) $\bullet t' = \bullet t''$ şi

(ii) $t' \bullet = t''\bullet$ şi

(iii) $W(t',p) = W(t'',p) \quad \forall p\in P$ şi

(iv) $W(p,t') = W(p,t'') \quad \forall p\in P$.

Tranziţiile congruente se caracterizează prin faptul că realizează un flux identic al marcajelor în reţeaua Petri. Ele sunt legate de noduri diferite, aşa cum se prezintă în Figura 5-8, şi astfel dacă este satisfăcută una dintre condiţiile de execuţia se iniţiază fluxul de marcaje.

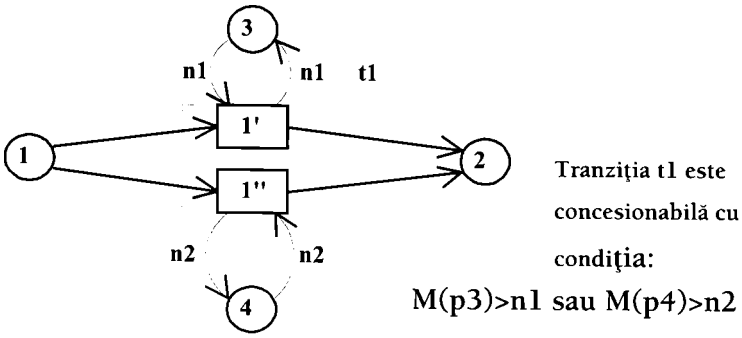


Fig. 5-8 Reprezentarea condiţionărilor disjunctive

Cu ajutorul acestor patru condiţionări elementare pot fi realizate şi condiţionările mai complexe după descompunerea acestora în aşa fel încât să conţină doar pe cele elementare.

Pe baza celor de mai sus poate fi realizată corecţia reţelei din figura 5.2-1. Aşa cum s-a văzut restricţiile suplimentare de concesionare sunt:

$[M(p_4)=2] \Leftrightarrow$ activează t_1 (5.3-19)

$[M(p_1)=2] \Leftrightarrow$ activează t_4 . (5.3-20)

Acestea se descompun după cum urmează:

$[M(p_4)\geq 2 \wedge M(p_4)\leq 2] \Leftrightarrow$ activează t_1 (5.3-21)

$[M(p_1)\geq 2 \wedge M(p_1)\leq 2] \Leftrightarrow$ activează t_4 (5.3-22)

Deoarece amândouă locaţiile în cauză au capacitatea $K(p_1)=K(p_2)=2$, condiţiile $M(p_4)\leq 2$ şi $M(p_1)\leq 2$ sunt întotdeauna satisfăcute relaţiile de mai sus se simplifică astfel:

$[M(p_4)\geq 2] \Leftrightarrow$ activează t_1 (5.3-23)

$[M(p_1)\geq 2] \Leftrightarrow$ activează t_4 (5.3-24)

Ca urmare, pentru realizarea corecţiei ajunge câte o buclă între p_4 şi t_1 respectiv între p_1 şi t_4 fără să fie necesară complementarea vreunei locaţii sau secţionarea vreunei tranziţii. Figura 5-9 prezintă reţeaua originală N împreună cu cea corectată N' . O analiză bazată pe grafuri a reţelei N' conduce la grafurile accesibile GA' respectiv condensata acestuia GA'^{K} aşa cum s-a impus prin condiţiile de sinteză.

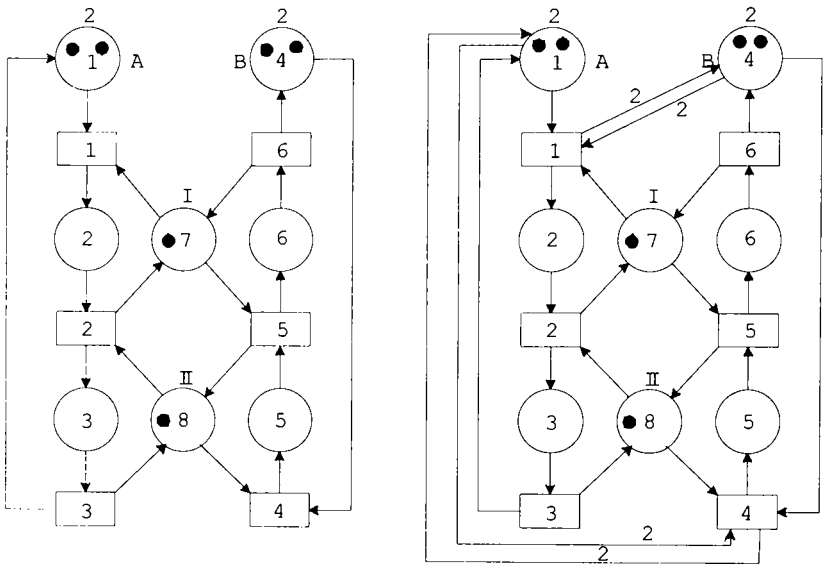


Fig. 5-9 Modelul procesului din Fig. 5-1 înainte (a) şi după corecţie (b)

5.4 Consideraţii de controlabilitate

O reţea Petri este controlabilă prin restricţii doar dacă toate tranziţiile critice din procesul condus sunt controlabile. Dacă această controlabilitate este exploatată de un controler atunci restricţiile pot fi realizate de către acest controler, fără a fi necesară schimbarea structurii reţelei care modelează procesul condus. Un astfel de controler trebuie să lucreze pa baza stării procesului condus, stare reprezentată de marcajul reţelei.

Vectorii concesionanţi respectiv deconcesionanţi determinaţi mai sus pot sta la baza unei strategii prin care se realizează conducerea procesului. Aspectele specifice unei astfel de abordări sunt prezentate în capitolele ce urmează.

Paragraful 6.4.6 prezintă un exemplu pentru o posibilă implementării corecţiilor de reţea determinate mai sus.

5.5 Concluzii

Corecţiile de reţea sunt necesare în situaţia în care, după analiza reţelei Petri care modelează sistemul, se ajunge la concluzia că anumite proprietăţi dinamice nu sunt satisfăcute. Aceste situaţii pot fi:

a) Reţeau prezintă grade de libertate care ar putea duce procesul modelat în stări nepermise sau nedorite (faţă de un posibil optim) din punct de vedere tehnologic. Pentru a limita numărul gradelor de libertate trebuie introduse restricţii de reţea – elemente structurale suplimentare prin care se limitează numărul gradelor de libertate.

b) Reţeaua nu satisface condiţiile de viabilitate şi/sau reversibilitate. În acest caz reţeaua poate fi corectată prin:

   (i) restricţii de reţea ca în cazul (a) de mai sus, sau

   (ii) extinderi ale reţelei originale cu elemente noi care, la nivelul sistemului modelat, se traduce prin introducerea de funcţionalităţi noi.

Metodele de sinteză a corecţiilor de reţea, prezentate în acest capitol, scot în evidenţă modificările de structură necesare a fi executate în reţeaua de originală pentru asigurarea funcţionării corecte. Restricţiile de reţea, conform metodelor prezentate, se realizează prin completarea reţelei originale cu bucle de reţea, locaţii complementare şi/sau tranziţii congruente.

Informatii referitoare la corectabilitatea reţelei se pot obţine prin analiza invarianţilor reţelei. S-a demonstrat că dacă o anumită proprietate dinamică nu este prezentă într-o reţea Petri ea nu poate fi realizată prin introducerea de restricţii suplimentare de concesionare realizate prin bucle de reţea.

În cazul în care reţeaua este corectabilă pot fi determinate, pe cale analitică, pe baza matricei diferenţă definită în acest capitol, locaţiile semnificative pe baza cărora pot fi realizate restricţiile suplimentare prin bucle de reţea. Matricea diferenţă se defineşte, respectiv locaţiile semnificative se determină, pentru fiecare tranziţie critică în parte.

Dacă locaţiile semnificative sunt disponibile atunci pot fi realizate restricţiile suplimentare de concesionare combinând următoarele patru condiţionări elementare:

1) $m(p) \geq n$ – realizabilă printr-o buclă de reţea;

2) $m(p) \leq n$ – realizabilă printr-o locaţie complementară;

3) $[\, m(p_1) \geq n_1 \,] \wedge [\, m(p_2) \geq n_2 \,]$ – realizabilă prin două bucle de reţea;

4) $[\, m(p_1) \geq n_1 \,] \vee [\, m(p_2) \geq n_2 \,]$ – realizabilă prin două bucle de reţea şi două tranziţii congruente.

Metodele de corecţie prezentate în acest capitol reprezintă o sinteză necesară în cadrul sintezei strategiei de comandă prezentată în capitolul 7, atât prin formularea restricţiilor cât şi prin schimbările de structură prezentate.

6. Structuri de conducere bazate pe modele de reţea Petri

Acest capitol prezintă un studiu asupra efectelor tranziţiilor temporizate în modelarea cu reţele Petri. Ca şi concluzie a acestui studiu se enunţă principiul încapsulării proceselor de arbitrare a resurselor. Se introduce o metodă de interpretare a reţelelor Petri. Se introduce noţiunea de "maşină virtuală de reţea Petri" care stă la baza unei noi metode de determinare a stării procesului condus şi de implementare a sistemelor de comandă. Se prezintă un studiu asupra caracteristicilor sistemelor de comadă implementate pe baza maşinii virtuale de reţea Petri.

6.1 Reţele Petri controlate

În continuare vom presupune că s-a optat pentru o abordare discretă orientată pe evenimente atât în modelarea şi analiza cât şi în conducerea proceselor. Presupunem deasemnea că s-a optat pentru formalismul reţelelor Petri ca metodă de modelare.

Prin urmare, suntem interesaţi în conducerea proceselor modelate prin reţele Petri. Acele reţele Petri care modelează procese controlabile le vom numi *reţele Petri controlate*, conform definiţiei de mai jos.

Definiţia 6-1 (Reţea Petri controlată)

Se numeşte reţea Petri controlată, abreviat CtlPN, 5-tuplul N=(P,T,F,C,B), unde:

(1) (P,T,F) este o reţea Petri;

(2) C este mulţimea finită a locaţiilor de control astfel, încât $C \cap P = \varnothing$

(3) $B \subseteq C \times T$ este mulţimea arcelor orientate care asociază locaţii de control cu tranziţiile reţelei.

În reprezentarea grafică locaţiile de control se vor reprezenta prin cercuri din linie dublă şi vor fi referite prin c_i. Aceste simboluri se vor utiliza în situaţia în care se doreşte reprezentarea explicită a punctelor în care se intervine asupra evoluţiei reţelei (a sistemului modelat) prin acţiunile de comandă a unui controler. Acţiunile de control se exercită prin locaţii deoarece tranziţiile sunt acelea prin care se schimbă starea reţelei şi prin urmare asupra lor trebuie intervenit dacă se doreşte influenţarea evoluţiei reţelei. Putem presupune, fără restrângerea generalităţii, că nu există vre-o tranziţie a reţelei originale care să aibă premulţimea vidă, prin urmare, prin locaţiile de control se vor executa acţiuni de sincronizare.

Mulţimea locaţiilor de control asociate tranziţiei t o vom nota cu ${}^{(c)}t$. Tranziţiile pentru care ${}^{(c)}t \neq \varnothing$ se vor numi *tranziţii controlabile*. Mulţimea tuturor tranziţiilor controlabile se va nota prin T_c.

Modelul de reţea Petri al procesului condus va conţine, cu siguranţă, tranziţii controlabile deoarece:

(i) datorită cerințelor de flexibilitate sistemul a fost astfel conceput încât să permită execuția diverselor strategii de conducere (controlabilitatea impusă prin specificație);

(ii) din motive tehnologice procesele de transformare materială și/sau informațională trebuie să fie (sau trebuie să fie făcute) controlabile.

Definiția 6-2 (Rețea P/T marcată, controlată)

Se numește rețea P/T marcată, controlată, abreviat mcPTN, 8-tuplul $N=(P,T;F,K,W,M_0,C,B)$, unde:

(1) $(P,T;F,K,W,M_0)$ este o mPTN;

(2) C este mulțimea finită a locațiilor de control, astfel încât $C\cap P=\varnothing$;

(3) $B\subseteq C\times T$ este mulțimea arcelor orientate care asociază locații de control cu tranzițiile rețelei.

Definiția 6-3 (Comandă)

Se numește *comandă* a unei mcPTN funcția $u:C\to\{0,1\}$ care alocă o valoare binară fiecărei locații de control $c\in C$ a rețelei.

Mulțimea tuturor comenzilor se va nota prin U. Comanda u_R se definește prin $u_R:=0$ pentru $\forall c\in C$ iar comanda u_P se definește astfel încât $u_P:=1$ pentru $\forall c\in C$. Fiind date comenzile $u,u'\in U$, spunem că u' este mai permisivă decât u dacă $u'\geq u$ pentru $\forall c\in C$ și $u'>u$ pentru cel puțin o locație de control $c\in C$. Prin urmare u_P este cea mai permisivă comandă iar u_R este cea mai puțin permisivă (cea mai restrictivă).

Se pot formula strategii de conducere în circuit deschis sau în circuit închis. Suntem interesați în formularea unor strategii de conducere în circuit închis. O strategie de conducere în circuit închis este o funcție care alocă fiecărui marcaj $\mathbf{m}$ o submulțime din mulțimea comenzilor $U(\mathbf{m})\subset U$. Dacă submulțimea $U(\mathbf{m})$ are mai mult decât un singur element, strategia de comandă este nedeterministă.

Strategia de control U_1 este mai permisivă decât strategia de control U_2 dacă $U_1(\mathbf{m})\supseteq U_2(\mathbf{m})$ pentru toate marcajele $\mathbf{m}$ și există un marcaj $\mathbf{m}'$ pentru care $U_1(\mathbf{m}')\supset U_2(\mathbf{m}')$.

Fie $\mathbf{m}$ un marcaj și u o comandă, spunem că $\mathbf{m}'$ este imediat accesibilă din $\mathbf{m}$ la comanda u, dacă există un pas $\mathbf{p}$ astfel încât $\mathbf{p}$ are concesie la marcajul $\mathbf{m}$ și comanda u și dacă după execuția pasului $\mathbf{p}$ (a tranzițiilor din pasul $\mathbf{p}$) se generează marcajul $\mathbf{m}'$. Mulțimea tuturor marcajele imediat accesibile la un marcaj $\mathbf{m}$ și o comandă u se va nota prin $R_1(\mathbf{m},u)$. Spunem că un marcaj $\mathbf{m}_k$ este accesibil de la marcajul $\mathbf{m}_0$ la comanda $u\in U$ dacă există o secvență $(\mathbf{m}_0\mathbf{m}_1\ldots\mathbf{m}_k)$ astfel încât $\mathbf{m}_i\in R_1(\mathbf{m}_{i-1},u)$ pentru $0<i\leq k$.

Mulțimea tuturor marcajelor accesibile din $\mathbf{m}$ la comanda u se va nota prin $R_\infty(\mathbf{m},u)$ cu convenția că $\mathbf{m}\in R_\infty(\mathbf{m},u)$ semnifică execuția unei secvențe de tranziții de lungime nulă.

Vom extinde notația de mai sus pentru mulțimea marcajelor accesibile la o strategie de control U. Vom nota prin $R_\infty(\mathbf{m},U)$ mulțimea marcajelor $\mathbf{m}'\in R_\infty(\mathbf{m},U)$ dacă există secvența de marcaje $\mathbf{m}_0,\mathbf{m}_1,\ldots,\mathbf{m}_n$ și secvența de comenzi $u_0,u_1,\ldots,u_{n-1}$, $u_i\in U(\mathbf{m}_i)$, $n\geq 0$, astfel încât $\mathbf{m}=\mathbf{m}_0$ și $\mathbf{m}_n=\mathbf{m}'$ și pentru $\forall 0\leq i\leq n$, $\mathbf{m}_{i+1}\in R_1(\mathbf{m}_i,u_i)$.

6.2 Tranziţii temporizate

6.2.1 Tranziţii în sisteme tehnice

În sistemele tehnice acţiunile nu pot fi executate instantaneu. Execuţia lor necesită o anumită durată. Tranziţiile care modelează astfel de acţiuni se vor numi tranziţii temporizate

Reţelele Petri, în felul în care au fost definite în capitolele anterioare, permit modelarea relaţiilor cauzale din sistemul tehnic modelat. În modelul de reţea Petri tranziţiile se execută instantaneu şi se produc la momente de timp nedefinite în model. Momentul executării poate fi impus printr-un semnal extern (informaţie din mediul înconjurător) sau prin condiţii suplimentare introduse în regula de tranziţie.

Evenimentele respectiv schimbările de stare care necesită o durată finită de realizare nu pot fi modelate în mod direct prin tranziţii (aşa cum s-a presupus în cazul reţelelor cauzale) deoarece sintaxa reţelelor Petri presupune o executare instantanee a tranziţiilor.

O analiză calitativă a proceselor cu durată finită (în continuare procese temporizate) se poate realiza şi pe baza reţelelor Petri cauzale fără a modifica regula de tranziţie, deoarece sintaxa reţelei nu impune vre-un moment de timp pentru execuţia unei tranziţii concesionate şi astfel nici nu limitează în vre-un fel durata de "şedere" a marcajelor în locaţiile marcate. Acest lucru permite ca fiecare proces temporizat să poată fi reprezentat prin două evenimente atemporale "pornire" respectiv "oprire" şi o stare "proces în derulare" cu durată finită, conform Figurii 6-1.

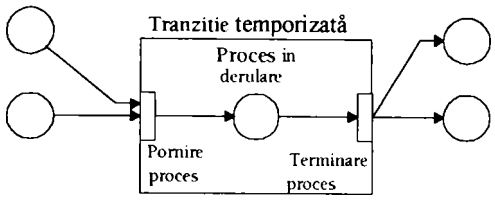


Fig. 6-1 Conversia unei tranziţii temporizate

Într-o reţea Petri la execuţia tranziţiei t_j marcajele se schimbă instantaneu de la marcajul **m** la marcajul **m'** conform relaţiei: $\mathbf{m'}=\mathbf{m}+(\mathbf{t}_j^+ + \mathbf{t}_j^-)$.

În cazul sistemelor tehnice, acţiunile modelate prin tranziţii nu vor fi instantaneu executate, ceea ce se traduce prin faptul că există posibilitate de a se genera şi alte marcaje decât cele din relaţia de mai sus. Plecând de la marcajul **m** se va ajunge la marcajul **m'** prin intermediul unui şir de marcaje. Ca prim exemplu se va considera cazul cel mai simplu din Figura 6-2.

Figura 6-2(a) prezintă execuţia tranziţiei t conform sintaxei modelului de reţea Petri. Figura 6-2(b) şi (c) prezintă cazurile posibile într-un sistem tehnic real. Se observă că sunt generate marcaje suplimentare faţă de model.

Figura 6-2(b) şi 6-2(c) diferă prin modul în care sunt utilizate resursele necesare executării tranziţiei. În cazul (b) resursa modelată prin p_1 este eliberată odată cu iniţierea executării tranziţiei iar în cazul (c) doar după terminarea executării tranziţiei.

Trebuie observat faptul că formalismul reţelelor Petri, aşa cum a fost definit în capitolul 3, nici nu permite descrierea corectă a unui astfel de comportament şi prin urmare nici rezultatele analizei, executate pe baza modelului de reţea Petri, nu pot fi corecte.

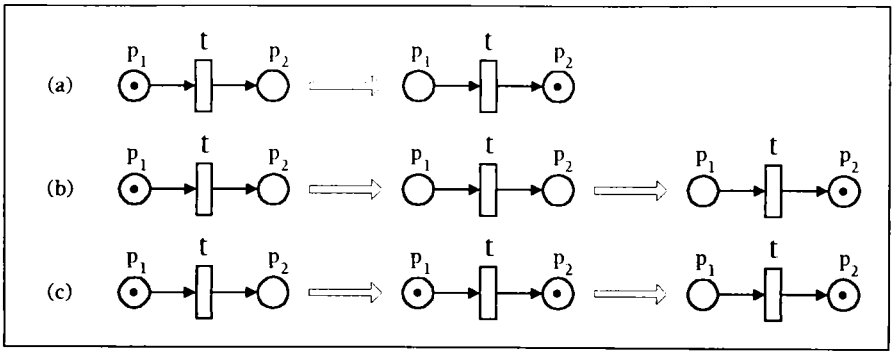


Fig. 6-2 Execuția temporizată a unei tranziții

Următorul exemplu, prezentat în Figura 6-3, arată că se pot produce chiar și secvențe de tranziții care nu sunt prezente în modelul de rețea Petri.

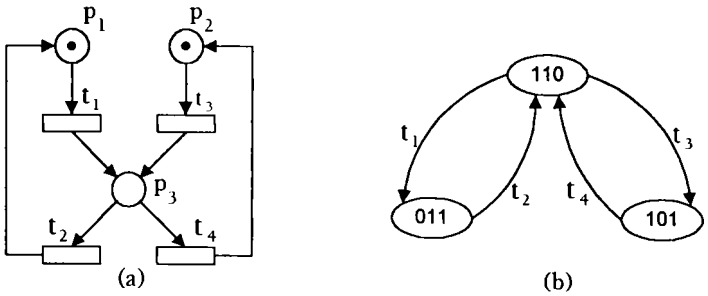


Fig. 6-3 Exemplu

În Figura 6-3 se prezintă o rețea Petri simplă împreună cu graful de accesibilitate generat conform sintaxei rețelei.

Vom considera pentru tranzițiile t_1 și t_3 comportamentul din Figura 5-2(b). Această alegere nu restrânge generalitatea raționamentelor ce urmează deoare sistemul tehnic poate fi astfel realizat încât să excludă comportamentul din Figura 5-2(c) impunând ordinea de interacțiune cu locațiile vecine (funcțiunile de bază vecine). Vom considera că durata tranzițiilor este τ_1 respectiv τ_3 astfel încât $\tau_1>\tau_3$. Figura 6-4 prezintă diagrama pentru o posibilă ordonare temporală a executării rețelei din Figura 6-3.

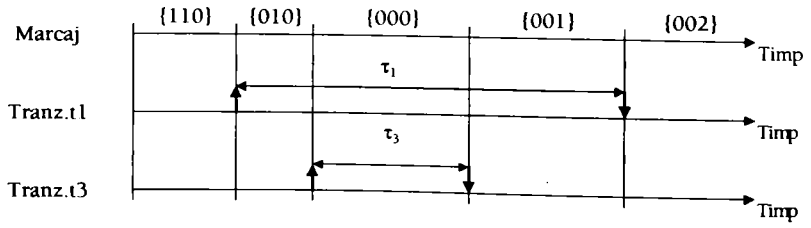


Fig. 6-4 Ordonarea temporală pentru exemplul anterior

Comparând această diagramă cu graful de accesibilitate din Figura 6-3 se observă că datorită executării temporizate a tranzițiilor t_1 și t_3 se pot produce:

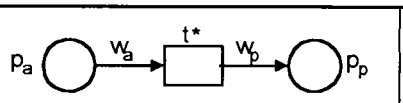
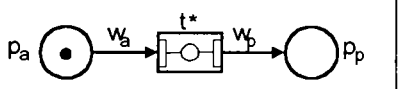
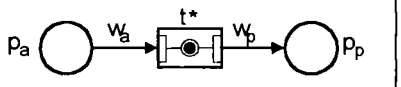
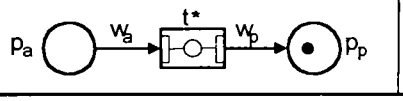
- marcajele nepermise [010],[000],[001],[002] şi
- secvenţa de tranziţii nepernisă $\sigma=t_1t_3$

Datorită tranziţiilor temporizate apar o mulţime de marcaje şi secvenţe de tranziţii care nu sunt prezente în graful de accesibilitate al reţelei Petri. Ca urmare în cazul sistemelor tehnice, aplicând regula de tranziţie clasică, rezultatele obţinute prin analiza grafului de accesibilitate sau a matricii de incidenţe a modelului de reţea Petri, îşi pierd valabilitatea.

6.2.2 Regula de tranziţie temporizată

Comportamentul deficitar prezentat mai sus se datorează faptului că tranziţiile temporizate nu pot rezolva în mod determinist situaţiile de conflict. Conform sintaxei reţelei tranziţiile t_1 şi t_3 s-ar putea executa doar alternativ. Diagrama din Figura 6-4 arată că tranziţiile temporizate permit şi o execuţie concurentă a acestora. Acest lucru este posibil deoarece t_3 nu este deconcesionat odată cu iniţierea execuţiei tranziţiei t_1. O soluţie posibilă ar fi aceea de a rezerva, în momentul iniţierii execuţiei, marcajele respectiv capacităţile necesare execuţiei în pre- resprectiv postmulţimea tranziţiei temporixate t^*. Pentru a rezolva şi problemele care pot apare în situaţia în care conflictul apare din cauza locaţiilor comune ale premulţimilor se vor rezerva capacităţile necesare şi în premulţimea tranziţiei temporizate. Tabelul 5.1 prezintă funcţionarea regulei de tranziţie "temporizate".

Tabelul 5.1

	K_a^*	m_a^*		K_p^*	m_p^*
(1)	K_a	m_{a0}		K_p	m_{p0}
(2)	K_a-w_a	m_{a0}-w_a		K_p-w_p	m_{p0}
(3)	K_a	m_{a0}-w_a		K_p	m_{p0}+w_p

Locaţiile p_a şi p_p cumulează locaţiile din premulţimea respectiv postmulţimea tranziţiei temporizate t^*. Mărimile indexate cu "*" reprezintă mărimi ce descriu execuţia temporizată a tranziţiei t^*.

Linia (1) în Tabelul 5-1 prezintă starea iniţială în care locaţia p_a este marcată astfel încât tranziţia t^* este concesionată. Capacitatea locaţiei p_a este K_a iar marcajul ei este m_{a0}. Capacitate locaţiei p_p este K_p iar marcajul ei este m_{p0}.

Linia (2) în Tabelul 5-1 corespunde situaţiei de acţiune (sau proces) în derulare. Iniţierea acţiunii se modelează prin iniţierea execuţiei tranziţiei soldată cu extragerea a w_a marcaje din locaţia p_a. În acelaşi moment se reduce capacitatea locaţiei p_a la valoarea K_a–w_a. În acest fel premulţimea locaţiei p_a "vede" tranziţie neexecutată. Capacitate locaţiei p_p se

reduce la valoarea K_p–w_p. În acest fel se rezervă capacitate în locația p_p pentru marcajul generat prin execuția tranziției t*.

Linia (3) din Tabelul 5-1 prezintă situația de după execuția tranziției t*. Se observă că se revine la capacitățile inițiale iar marcajele corespund efectului execuției tranziție t*.

Dacă temporizarea dispare, dispare și faza 2 din Tabelul 5.1 și se ajunge la sintaxa inițială a rețelei. Trebuie studiat dacă integrarea unei tranziții temporizate împreună cu regula de tranziție temporizată păstrează sintaxa (comportarea) în ansamblu a rețelei.

Figura 6-5. prezintă o tranziție temporizată t* integrată într-o rețea Petri.

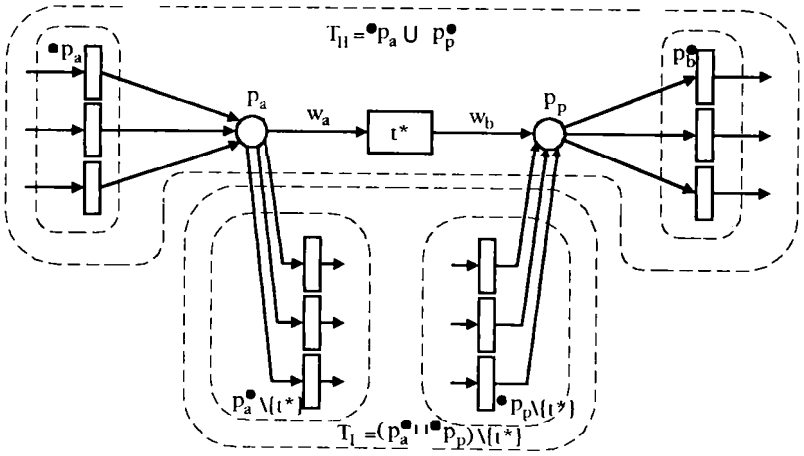


Fig. 6-5 Tranziția temporizată și mediul său

În Figura 6-5 p_a și p_p simbolizează întreaga pre- respectiv postmulțime a tranziției temporizate t*. Această tranziție își exercită influența asupra restului rețelei prin intermediul premulțimilor și postmulțimilor locațiilor p_a și p_p. Ajunge astfel să studiem influența regulei de tranziție temporizate asupra acestor mulțimi.

Pentru tranzițiile mulțimii $p_a^\bullet \setminus \{t^*\}$ marcajul său se va reduce conform $m_a^*=m_{a0}-w_a$ în momentul inițierii execuției tranziției t*. În același moment pentru mulțimea ${}^\bullet p_p \setminus \{t^*\}$ se reduce capacitatea p_p conform relației $K_p^*=K_p-w_p$. În acest fel mulțimii $T_I = (p_a^\bullet \cup {}^\bullet p_p) \setminus \{t^*\}$, tranziția t* apare ca fiind instantanee și anume în momentul inițierii execuției acesteia.

Pentru tranzițiile mulțimilor ${}^\bullet p_a$ capacitatea p_a se reduce conform relației $K_a^*=K_a-w_a$ iar împreună cu relația $m_a^*=m_{a0}-w_a$ starea p_a se conservă pentru mulțimea ${}^\bullet p_a$ până în momentul terminării execuției t* când se realizează situația $(K_a^*=K_a) \wedge (m_a^*=m_{a0}-w_a)$, conform sintaxei rețelei.

Pentru tranzițiile mulțimii $p_p^\bullet$ marcajul p_p se schimbă în momentul terminării execuției t* la valoarea $m_p^*=m_{p0}+w_p$, conform sintaxei rețelei.

În acest fel mulțimii $T_{II} = {}^\bullet p_a \cup p_p^\bullet$, tranziția t* apare ca fiind instantanee și anume în momentul încheierii execuției acesteia.

Trebuie verificat dacă nu cumva faptul că cele două mulțimi de tranziții T_I respectiv T_{II} sesizează execuția tranziției t* la momente de timp diferite, conduce la efecte nedorite în rețeaua de bază.

Tranziţiile mulţimii T_I pot fi privite ca alternative la tranziţia t*. Un potenţial conflict dintre aceste tranziţii se rezolvă în momentul iniţierii execuţiei t*. În schimb tranziţiile mulţimii T_{II} sunt temporal anterioare respectiv posterioare tranziţiei t*. Respectând regula de tranziţie temporizată aceste tranziţii aşteaptă terminarea execuţiei tranziţiei t*. Datorită structurii din Figura 6-5 după iniţierea execuţiei tranziţiei t*, o tranziţie din T_I poate fi concesionată doar după execuţia unei tranziţii din T_{II}. Deoarece aceasta din urmă poate fi executată doar după terminarea execuţiei tranziţiei t*, pe durata execuţiei toate activităţile în mulţimea T_I şi T_{II} sunt blocate. Astfel, aplicînd regula de tranziţie temporizată, o tranziţie temporizată, prin efectele sale, se manifestă ca o tranziţie instantanee.

Deci într-un sistem tehnic, o tranziţie concesionată trebuie să-şi rezerve resursele reprezentate de marcajele şi capacităţile locaţiilor prin care s-a realizat concesionarea, conform relaţiilor $K_a^*=K_a-w_a$ şi $m_a^*=m_{a0}-w_a$. În acest fel se garantează faptul că acţiunea modelată prin tranziţia concesionată să poate fi efectiv executată.

6.2.3 Efectul tranziţiilor temporizate asupra modelării

Analizând, în privinţa rezervărilor de resurse partajate, conexiunile elementare în reţele Petri, se observă următoarele (Figura 6-6):

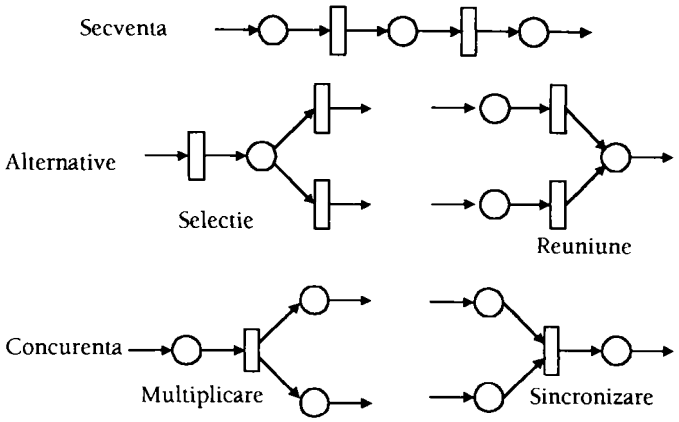


Fig. 6-6 Conexiuni elementare în reţele Petri

(i) În cazurile "Secvenţă", "Multiplicare" şi "Sincronizare" alocarea resurselor este dată apriori prin însăşi structura reţelei. Rezervarea nu este necesară deoarece locaţiile din aceste tipuri de conexiuni nu sunt partajate.

(ii) Cazurile "Selecţie" şi "Reuniune" conţin "locaţii partajate" locaţii ce modelează resurse partajate. Tranziţiile adiacente vor concura pentru obţinerea resurselor oferite de aceste locaţii. Sistemul tehnic care implementează o locaţie partajată (resursă partajată) trebuie să pună la dispoziţie mecanisme sigure de alocare (arbitrare a accesului) şi de rezervare (excludere mutuală).

(iii) În cazul "Reuniune" rezervarea resurselor locaţiei partajate duce la blocarea resurselor pe durata executării tranziţiei căreia i s-a alocat cu toate că tranziţia are nevoie de acea resursă doar în momentul terminării. Figura 6-7 prezintă câte un exemplu în acest sens.

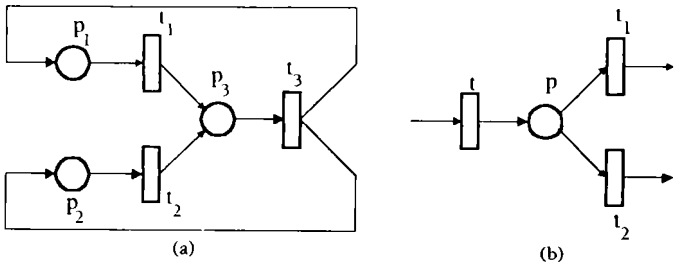


Fig. 6-7 Resurse partajate

În Figura 6-7(a) resursa partajată este reprezentată prin locația p_3. Dacă durata tranziției t_1 este τ_1 iar a tranziției t_2 este τ_2 și dacă $\tau_1=n\cdot\tau_2$ $n\in N$, în cazul în care p_3 este alocat tranziției t_1, tranziția t_2 rămâne blocată n ciclii proprii, ceea ce înseamnă o exploatare absolut ineficientă a resursei reprezentate de p_3. În Figura 6-7(b) resursa partajată este reprezentată prin locația p. Rezervarea acesteia de către una din tranziții o face inutilizabilă de către cealaltă.

Acest fenomen trebuie evitat încă din faza de modelare/proiectare prin "încapsularea" procesului de arbitrare a resurselor partajate, adică prin separarea procesului de arbitrare (de alocare a resursei) de procesele de prelucrare (informațională sau materială). Acest lucru se poate realiza printr-o simplă schimbare structurală a rețelei așa cum se prezintă în Figura 6-8, comparativ cu Figura 6-7.

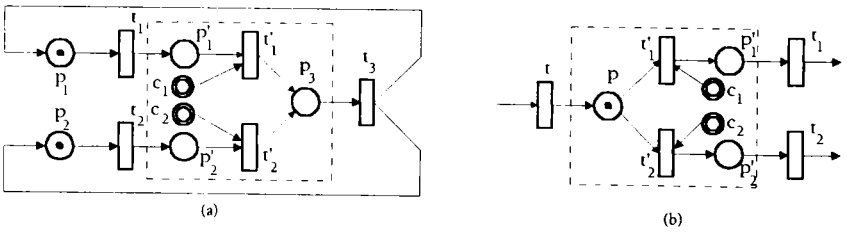


Fig. 6-8 Încapsularea proceselor de arbitrare a resurselor partajate

Zonele încadrate cu linie întreruptă modelează exclusiv procesul de arbitrare a resursei. Dacă strategia de conducere o cere arbitrarea poate să fie controlată. Din acest motiv s-au reprezentat și locațiile de control.

În Figura 6-8(a) locțiile suplimentare p_1' și p_2' modelează cererea de acces a tranzițiilor t_1 respectiv t_2 la resursa partajată iar tranzițiile suplimentare t_1'și t_2' modelează acțiunea de alocare a resursei partajate modelată prin p_3. În Figura 6-8(b) tranzițiile suplimentare t_1' și t_2' modelează acțiunea de alocare a resursei partajate modelată prin locația p iar locatiile p_1'și p_2', dacă sunt marcate, au semnificația de resursă alocată acțiunilor modelate prin tranzitiile t_1 sau t_2.

Având în vedere cele de mai sus trebuie formulată la modul imperativ următoarea regulă de modelare:

Modelul de rețea Petri al sistemelor tehnice trebuie astfel elaborat, încât procesele de arbitrare a resurselor partajate să fie "încapsulate" adică separate de procesele de prelucrare informațională sau materială.

Dacă se respectă această regulă şi dacă procesele de arbitrare a resurselor partajate lucrează corect (tehnica informatică pune la dispoziţie metodele necesare) nu mai este necesar a se apela la regula de tranziţie temporizată în modelarea prin reţele Petri. Dacă totuşi modelatorul consideră ca fiind util poate apela la transformarea tranziţiei temporizate conform Figurii 6-1. Această transformare conservă proprietăţile fundamentale de *viabilitate* şi *mărginire*, [PĂS'97], şi prin urmare nu este necesară din punctul de vedere al analizei în schimb ar putea fi utilă din punctul de vedere al controlului procesului tehnic modelat, aşa cum se prezintă mai jos în secţiunea 6.5.

6.3 Reţele Petri interpretate

Pentru ca o reţea Petri să modeleze un sistem tehnic ea trebuie să fie *interpretată*. Prin interpretare se înţelege operaţia de asociere a elementelor din sistemul tehnic modelat elementelor componente ale reţelei Petri. În acest fel reţeaua abstractă caracterizată prin mulţimi de elemente, structuri şi reguli sintactice primeşte o semantică specifică aplicaţiei concrete. În consecinţă, o interpretare pentru o reţea Petri se realizează prin definirea de corespondenţe dintre elementele reţelei şi obiectele sistemului tehnic modelat.

Pentru cazul unor controlere înglobate (embeded controller) şi a unor automate programabile cu număr mic de intrări/ieşiri în literatura de specialitate s-au propus diferite interpretări pentru reţele Petri.

În [KDS 95] şi [BSD 95] reţele C/E sunt utilizate pentru modelarea controlelelor paralele, sincronizate (denumite şi controlere digitale). Atributul de "paralel" sugerează faptul că sunt controlate procese paralele, cvasiindependente în interacţiune. Atributul de "sincronizat" specifică faptul că execuţia tranziţiilor este sincronizată cu un ceas global, adică toate tranziţiile concesionate la un moment dat sunt executate la acelaş tact al ceasului. Marcajele se determină o dată pe un ciclu al ceasului. Această clasă de controlere se realizează în tehnologie cablată.

Locaţiile reţelei C\E sunt interpretate ca stări locale ale controlerului. Marcajele locaţiilor specifică stările valide la un moment dat. Marcajul reţelei specifică starea globală a controlerului. Tranziţiile reprezintă schimbări de stare. Cu ajutorul semnalelor de intrare preluate din mediul înconjurător, controlerul generează semnalele de comandă care determină comportarea sistemului controlat. Semnalele de intrare influenţează executabilitatea tranziţiilor. Aceste semnale sunt variabilele unei funcţii logice numită *predicat al tranziţiei*. Un predicat al tranziţiei este o condiţie suplimentară impusă regulei de tranziţie din modelul de reţea Petri al controlerului. Semnalele de comandă generate sunt de două tipuri:

(1) semnale Mealy legate de execuţia tranziţiilor – un astfel de semnal este valid, dacă tranziţia corespunzătoare este concesionată şi predicatul ei este valid.

(2) semnale Moore legate de stări ale controlerului – aceste semnale sunt valide cât timp este realizată o anumită stare a controlerului.

În [Quă91b] o reţea Petri interpretată se defineşte prin 5-tuplul $NI = (P,T,F,q_p,q_T)$ unde:

(1) (P,T,F) este o reţea Petri;

(2) q_T este o funcţie logică, numită *predicat al tranziţiei*, definită prin $q_T: T \rightarrow \{0,1\}^{|X|}$ unde X este mulţimea semnalelor de intrare. Cu alte cuvinte funcţia q_T alocă fiecărei tranziţii o configuraţie a variabilelor de intrare.

(3) q_P este o funcţie care asociază locaţiilor reţelei reguli de calcul pentru determinarea semnalelor de ieşire pe baza semnalelor de intrare. Fiecărei locaţii i se asociază un sistem de funcţii logice parţial definite. Notând prin $EB = \{0,1\}^{|X|}$ mulţimea tuturor configuraţiilor posibile ale semnalelor de intrare, iar prin $AB = \{0,1,-\}$ mulţimea tuturor configuraţiilor posibile ale semnalelor de ieşire (cu "–" s-a notat situaţia "don't care"), mulţimea tuturor funcţiilor logice parţial definite poate fi reprezentată prin: AB^{EB} Având în vedere cele de mai sus funcţia q_P se defineşte prin: $q_P: P \rightarrow AB^{EB}$ În acest caz semnalele de ieşire ale NI la marcajul actual se obţin prin suprascrierea semnalelor existente cu cele determinate la marcajul dat. Nu se asociază semnale de ieşire tranziţiilor.

Interpretările de mai sus nu sunt utilizabile în cazul modelării unor sisteme de automatizare distribuită deoarece nu au în vedere faptul că, din cauza vitezei de propagare limitate a semnalelor respectiv a mesajelor, în general, nu se poate garanta livrarea acestora în ordinea apariţiei obiective a evenimentelor care le-au generat.

În cele ce urmează se prezintă o interpretare pentru reţele Petri pe baza căreia se elaborarează structuri de conducere care asigură atât respectarea relaţiilor cauzale cât şi tratarea corectă a tranziţiilor temporizate din procesul condus. Pe baza acesteia este posibilă realizarea sistemelor de conducere distribuite, orientate pe evenimente.

6.3.1 Evenimente şi acţiuni

Un proces tehnic se caracterizează prin activităţi executate atât simultan (concurent) cât şi secvenţial. Aceste activităţi sunt executate prin funcţiunile de bază ale sistemului tehnic modelat.

În cazul unei abordări discrete, orientate pe evenimente, sistemul tehnic modelat se structurează pe activităţi locale care mobilizează resursele disponibile în sensul realizării de scopuri locale, predefinite în specificaţia sistemului de conducere. Scopurile locale se realizează prin acţiuni locale. Acestea se pot găsi în două stări distincte:

(i) *starea activă* – starea în care mobilizează resursele disponibile în sensul realizării scopului local;

(ii) *starea inactivă* – starea în care nu mobilizează resursele disponibile în sensul realizării scopului local

Resursele locale la care apelează acţiunile locale se caracterizează prin mărimi de stare. În cazul unei abordări discrete spaţiul stărilor este o mulţime discrretă, numărabilă. Elementele acestei mulţimi vor fi mărimi discrete, exprimate prin valori discrete. Resursele pot fi referite ca funcţiuni de bază care pun la dispoziţie resusa dată.

Starea sistemului tehnic modelat este determinată de starea acţiunilor locale şi de starea funcţiunilor de bază.

Obiectele conceptuale de bază ale modelării orientate pe evenimente sunt *evenimentele* şi *acţiunile*.

Acţiunile reprezintă activităţi ca de exemplu: acţionarea unui buton, încărcarea unui program, prinderea unei piese pe o paletă, descărcarea unei piese de pe o maşină, montarea unei piese, etc. Acţiunile sunt atomice în sensul că, la nivelul de abstractizare

considerat, sunt indivizibile. Prin urmare acțiunile ori sunt executate ori nu sunt executate. La nivelul de abstractizare considerat ele nu pot fi executate parțial. La un alt nivel de abstractizare aceste acțiuni atomice eventual pot fi detaliate pe mai multe acțiuni la rândul lor atomice. De exemplu prinderea unei piese pe o paletă poate însemna: poziționarea piesei pe paletă, rigidizarea piesei, aducerea paletei în poziția de lucru. Acțiunile se identifică prin nume (sau etichetă). Presupunem existența unui univers al acțiunilor din sistemul tehnic modelat notat A având elementele notate prin a, b, c, ... Acțiunile sunt executate prin funcțiunile de bază ale sistemului tehnic modelat.

Execuția unei acțiuni este un eveniment în sistemul tehnic modelat. O anumită acțiune poate fi executată la diverse momente de timp din rațiuni și în contexte diferite. Un eveniment reprezintă o execuție specifică a unei acțiuni. Spre exemplu acțiunea de paletizare poate fi executată la diverse mașini ale unei celule flexibile de fabricație și pentru diverse piese în execuție. Fiecare execuție a acțiunii de paletizare este reprezentată printr-un eveniment. Legătura dintre acțiuni și evenimente este dată printr-o funcție de etichetare care asociază fiecărui eveniment acțiunea modelată de acel eveniment. Deoarece mai multe evenimente pot modela diverse execuții ale aceleiași acțiuni, respectiv pot exista acțiuni cărora nu le corespunde nici un eveniment, această funcție nu este nici injectivă și nici surjectivă. În principiu, evenimentele pot fi denumite în mod arbitrar cu condiția ca să se realizeze o identificare univocă.

Între evenimentele ce pot apare într-un sistem tehnic pot fi definite următoarele relații:

Cauzalitate: Cauzalitatea este o relație binară dintre evenimente. Într-o interpretare intuitivă a relației de cauzalitate afirmația - 'evenimentul **e** cauzează evenimentul **e'** - specifică faptul că dacă se produc atât **e** cât și **e'** atunci **e'** este cauzat de **e**. Sau altfel exprimat apariția evenimentului **e** este o condiție pentru ca **e'** să se poată produce. Nu neapărat este și o condiție suficientă pentru producerea evenimentului **e'** deoarece pot exista și alte evenimente a căror apariție este necesară pentru ca **e'** să se poată produce respectiv pot exista alte evenimente care dacă se produc, evenimentul **e'** nu se mai poate produce.

Cauzalitatea se bazează pe prezumpția existenței unei relații cauză→efect fixe (stabile) între execuția diverselor acțiuni (prin urmare a producerii diverselor evenimente). Relația de cauzalitate se descrie la nivelul evenimentelor și nu al acțiunilor deoarece diversele execuții ale unei acțiuni pot avea diverse cauze.

Conflict: Conflictul este o relație binară, ireflexivă și simetrică dintre evenimente cu semnificația că dacă **e** și **e'** sunt în conflict ele nu se pot produce amândouă în cursul unei execuții a sistemului tehnic modelat. Astfel dacă se produce **e(e')**, atunci **e'(e)** nu se poate produce. Dacă s-a executat **e**, atunci **e'** nu se poate produce, respectiv dacă s-a executat **e'**, atunci **e** nu se poate produce

Independența sau **concurența** este o relație binară simetrică între evenimente cu semnificația că dacă **e** și **e'** nu sunt nici intr-o relație de cauzalitate și nici într-o relație de conflict atunci ele se pot produce în mod independent. Prin urmare dacă sunt validate ele se pot produce în orice ordine.

Rețelele Petri reprezintă un instrument deosebit de eficient în modelarea acestor relații.

6.3.2 Interpretarea tranzițiilor

La nivel conceptual o tranziție a unei rețele Petri consemnează o acțiune în sistemul modelat. Execuția unei tranziții a rețelei Petri corespunde execuției acțiunii asociate ei. Datorită caracterului temporizat al acțiunilor din sistemele tehnice respectiv a regulei de

tranziție temporizate din modelul de rețea Petri, prin execuția unei astfel de tranziții se produc, în cazul general, două evenimente:

(i) activarea (inițierea) execuției;

(ii) terminarea (finalizarea) execuției tranziției.

Orice sistem de conducere ce se bazează pe modelul de rețea Petri al procesului condus trebuie să conțină module funcționale care sesizează, pe baza mărimilor măsurate în procesul condus, apariția acestor două evenimente. Prin funcționalitatea lor aceste module implementează câte un predicat pentru fiecare tip de eveniment. astfel:

(i) *Predicatul activării* care sesizează inițierea acțiunii, respectiv activarea funcțiunilor de bază care realizează acțiunea dată.

Predicatul activării se definește prin funcția A_j:$Y_a \rightarrow \{0,1\}$, unde Y_a este mulțimea mărimilor măsurate (a variabilelor) necesare în specificarea stării de activare a execuției tranziției date.

(ii) *Predicatul tranziției* care sesizează realizarea acțiunii respectiv sesizează evenimentul asociat realizării acțiunii.

Predicatul tranziției se definește prin funcția H_j:$Y_h \rightarrow \{0,1\}$, unde Y_h este mulțimea mărimilor măsurate pe baza cărora se determină, în mod univoc, dacă acțiunea dată s-a executat sau nu s-a executat.

În definițiile de mai sus prin indicele j se face legătura cu tranziția din modelul de rețea Petri al sistemului modelat.

Proiecția pe o axă temporală a evenimentelor legate de execuția unei tranziții temporizate se prezintă în Figura 6-9

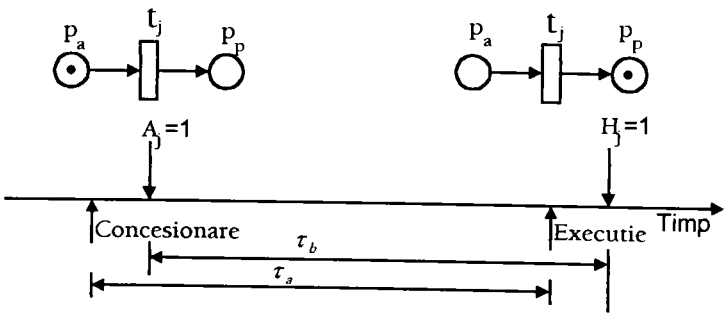


Fig. 6-9 Execuția temporizată și predicatul activării respectiv predicatul tranziției

A_j=1 specifică momentul în care predicatul activării devine valid respectiv momentul în care se generează semnalul sau mesajul care specifică activarea execuției. H_j=1 specifică momentul în care predicatul tranziției devine valid respectiv momentul în care se generază semnalul sau mesajul de execuție. Aceste semnale sau mesaje sunt destinate unui sistem de conducere și ajung la acesta prin canale de comunicație. Tranziția temporizată durează τ_a unități de timp iar sistemul de conducere va sesiza o durată de execuție de τ_b unități de timp.

La nivel tehnic o tranziție a unei rețele Petri reprezintă un subsistem al sistemului modelat capabil să execute o anumită acțiune. Vom numi acest subsistem *operator de tranziție*, abreviat OPT. Regula de concesionare a rețelei precizează condițiile în care execuția unei acțiuni poate să fie inițiată. OPT trebuie să fie capabil a realiza interacțiunile cu locațiile

vecine și, eventual, cu un sistem de conducere. Pentru aceasta OPT va pune la dispoziție următoarele funcțiuni:

(i) *Interacțiunea cu un sistem de conducere* prin preluarea unui semnal de comandă furnizat de acesta. Acest semnal va fi tratat astfel încât să se asigure funcționalitatea locației de control conform definiției rețelelor Petri controlate.

(ii) *Interacțiunea cu locațiile vecine* (funcțiunile de bază ce le realizează) trebuie să asigure atomicitatea executării tranzitiei. pentru aceasta OPT trebuie să realizeze:

- *Sesizarea concesionării.* OPT va sesiza concesionarea pe baza regulei de tranziție temporizate $(-\mathbf{t}_j^- \le \mathbf{m}_f \le \mathbf{k}_f - (\mathbf{t}_j^+ + \mathbf{t}_j^-)) \wedge (u_j = 1)$ unde prin $\mathbf{m}_f$ s-a notat vectorul marcajelor libere iar prin $\mathbf{k}_f$ vectorul capacităților libere. Funcțiunile de bază trebuie să fie capabile a pune la dispoziție informațiile care specifică marcajele respectiv capacitățile libere.
- *Rezervarea*, dacă este cazul, a marcajelor și a capacităților necesare execuției acțiunii. Funcțiunile de bază trebuie să permită rezervarea marcajelor și capacităților. Rezervare cu succes a marcajelor și capacităților trebuie să determine activarea funcțiunilor de bază prin care se execută acțiunea dată. Nu toate funțiunile de bază realizează locații de rețea. OPT trebuie să fie capabil a aștepta eliberarea resurselor necesare.
- *Execuția schimbărilor de stare locale* asociate acțiunii conform sintaxei rețelei, adică transferul marcajelor rezervate în capacitățile rezervate conform regulei de tranziție temporizate.

Deși definiția de mai sus a OPT este pur conceptuală, OPT trebuie să fie prezent sau să fie realizabil în sistemul de conducere modelat altfel modelul de rețea Petri nu poate fi implementat. OPT trebuie să implementeze un modul funcțional concret definit și bine delimitat conform structurii de modul funcțional prezentate în capitolul 2.

6.3.3 Interpretarea locațiilor

La nivel conceptual locațiile sunt purtătoare a informațiilor de stare. Dacă presupunem că P-elementele modelului de rețea Petri reprezintă *locații* (elemente de acumulare, locații de memorie) sau *variabile* capabile să rețină anumite structuri de date, iar T-elementele reprezintă *acțiuni* (tranziții, operații cu structuri de date), atunci putem spune că starea sistemului este dată de valoarea variabilelor (locațiilor) la un moment dat. Deci, o stare în acest caz este o aplicație de la mulțimea P a locațiilor la o mulțime de tipuri de date utilizate ca mai sus.

Ca urmare a apariției evenimentelor se schimbă stările locale conform regulei de tranziție temporizate. Structura rețelei și sintaxa acesteia precizează modul în care se realizează interacțiunea cu tranzițiile.

La nivel tehnic locațiile reprezintă obiecte ale sistemului modelat care sunt sau pot să fie suportul material al variabilelor rețelei. Funcțiunea de bază ca și concept general este un astfel de obiect al sistemului tehnic modelat. La nivel conceptual orice resursă poate fi echivalată cu o funcțiune de bază care pune la dispoziție aceea resursă. Prin marcajele locațiilor se specifică starea funcțiunilor de bază pe care acestea le modelează.

Anexa 1 prezintă corespondența dintre obiectele abstracte ale rețelei Petri și obiecte ale sistemelor tehnice din diferite domenii.

6.3.4 Interpretarea relației de flux

La nivel conceptual relația de flux specifică structura cauzală a interacțiunilor din sistemul modelat (a interacțiunilor dintre evenimente). Din punctul de vedere al modelatorului interacțiunile pot fi de natură logică, temporală, materială sau energetică.

La nivel tehnic relația de flux reprezintă canale de comunicație (legături pentru semnale) prin care se realizează interacțiunea dintre tranziții și locații respectiv medii de transport al materialelor sau energiei. Proiectantul trebuie să asigure realizarea interacțiunilor astfel încât să se realizeze acțiunile predefinite în specificația sistemului de conducere

Exemplu:

Figura 6-10 prezintă un exemplu simplu care pune în evidență elementele definite mai sus. În realizarea unui proces tehnologic este necesară încălzirea unui lichid dintr-un recipient până la o anumită temperatură cu ajutorul instalației din Fig.6-10, unde: K- este un cotactor prin care se alimentează rezistența de încălzire R iar $\vartheta°$- este un termostat calibrat.

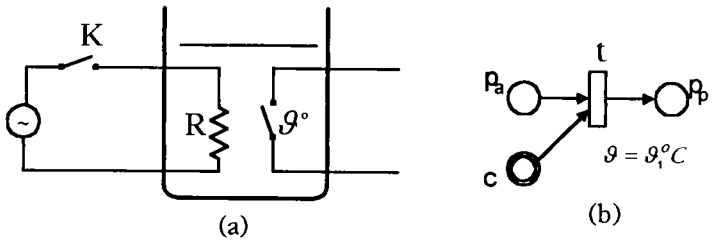


Fig. 6-10 Exemplul unui proces de încălzire a unui lichid

Modelăm acest *proces de încălzire* prin tranziția t din Fig.6-10(b). Marcajul locației p_a semnifică faptul că în procesul condus sunt date toate condițiile necesare pentru începerea încălzirii. Dacă strategia de comandă o cere, sistemul de conducere validează execuția tranziției. În modelul din Fig.6-10(b) *comanda* de validare este reprezentată prin marcajul locației c. În acest moment OPT poate închide contactorul K. Semnalul măsurat necesar *predicatului activării* poate fi un contact auxiliar al contactorului sau, de preferință, un senzor al curentului prin rezistența R. Modulul funcțional care realizează predicatul va semnaliza sistemului de conducere starea acestuia printr-un semnal sau mesaj. Valoarea măsurată (variabila) *predicatului tranziției*, în acest caz, este ieșirea termostatului $\vartheta°$. *Predicatul tranziției* devine valid în momentul în care temperatura a ajuns la valoarea $\vartheta_1°C$. Modulul funcțional care implementează predicatul tranziției semnalizează sistemului de conducere execuția tranziției printr-un semnal sau mesaj.

6.3.5 Rețele Petri interpretate

Conform definițiilor de mai sus mulțimile de obiecte din procesul condus pe baza cărora se realizează interpretarea rețelelor Petri care, în acest fel, va modela procesul dat sunt următoarele:

(i) mulțimea predicatelor activării, notată prin A;

(ii) mulțimea predicatelor tranziției, notată prin H;

(iii) mulțimea operatorilor de tranziție, notată prin OP;

(iv) mulțimea obiectelor care sunt suportul material pentru variabilele rețelei (mărimile de stare), notată prin SV.

Interpretarea rețelei Petri constă în definirea următoarelor aplicații:

1) pentru interpretarea locațiilor

$IP:SV \rightarrow P$

2) pentru interpretarea tranzițiilor

$IT_A:A \rightarrow T$

$IT_H:H \rightarrow T$

$IT_{OP}:OP \rightarrow T$

Pe baza acestor aplicații se definește rețeaua Pentri interpretată.

Definiția 6-4(rețea Petri interpretată)

O rețea Petri interpretetă este 5-tuplul $NI=(N,IP,IT_A,IT_H,IT_E)$ unde:

- N este o rețea Petri;
- IP - este aplicația pentru interpretarea locațiilor;
- IT_A,IT_H,IT_{OP} - sunt aplicațiile pentru interpretarea tranzițiilor.

6.4 Structuri de conducere

6.4.1 Sistemul de conducere. Schema bloc.

Conform interpretărilor de mai sus sistemul tehnic modelat constă din următoarele obiecte:

(1) funcțiuni de bază ca obiecte purtătoare ale variabilelor rețelei;

(2) subsistemele care realizează operatorii de tranziție;

(3) subsistemele care implementează predicatele activării;

(4) subsistemele care implementează predicatele tranziției.

Interacțiunile dintre aceste elemente sunt surprinse prin structura rețelei care modelează sistemul tehnic. Acest sistem este controlat prin *sistemul de coonducere*, abreviat SC, care exercită controlul conform unei *strategii de conducere*. SC poate interveni la nivelul tranzițiilor (al OPT) printr-o acțiune de validare/invalidare a execuției acestora. De fapt rețeaua Petri care descrie sistemul tehnic condus (în continuare desemnat prin *proces condus* sau *proces*) este o rețea Petri controlată.

Problematica conducerii unui proces pe baza unei abordări discrete, orientate pe evenimente, modelat prin rețele Petri, constă în a asigura îndeplinirea următoarelor condiții de funcționare (după [PĂS'97]:

(i) Asigurarea succesiunii corecte a acțiunilor (a operațiilor tehnologice) în procesul condus. Pentru aceasta trebuie elaborate comenzi în conformitate cu starea actuală și cu relațiile cauzale din procesul condus.

(ii) Corectitudinea alocării și eliberării resurselor necesitate de fiecare operație în parte.

(iii) Execuția tranzițiilor în procesul condus prin operatorii de tranziție;

(iv) Repetabilitatea operațiilor, fără blocaje circulare datorate utilizării partajate a unora dintre resurse.

Figura 6-11 prezintă o primă schemă bloc a unui sistem de automatizare care pune în evidență schimbul de informații (semnale, mesaje) dintre SC și mediul său (procesul condus, alte sisteme de conducere, sistem MMI–Man Mashine Interface.

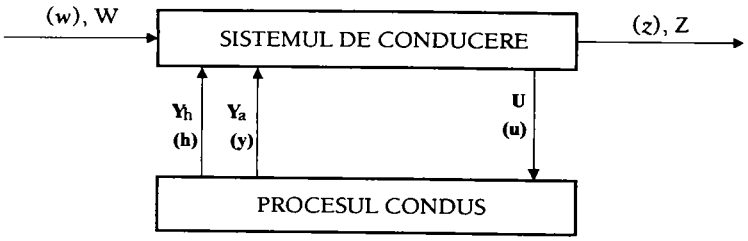


Fig. 6-11 Schema bloc a sistemului de conducere

În cadrul sistemului de conducere schimbul de informații se realizează prin următoarele mărimi:

(i) Y_h - mulțimea mărimilor măsurate în procesul condus, necesare predicatelor tranziției;

(ii) Y_a - mulțimea mărimilor măsurate în procesul condus, necesare predicatelor activării; În cazul în care modulele funcționale care implementează predicatele tranziției și predicatele activării se consideră a fi incluse în sistemul de conducere, mărimile Y_h și Y_a sunt *mărimi de reacție.*

(iii) W - mulțimea mărimilor de conducere; Mărimile de conducere sunt mărimi binare prin care SC interacționează cu alte sisteme de conducere. Această mulțime se reprezentă vectorial astfel: $\mathbf{w}^T=|w_1\ w_2\ \ldots\ w_{|W|}|$, $w_i \in W$, $0 \le i \le |W|$. Mărimile de conducere pot fi de două categorii:

1) mărimi de comandă în cazul în care sunt furnizate de către un sistem de conducere de la un nivel ierarhic superior;

2) mărimi de concesionare dacă sunt furnizate de către un OPT de la nivelul ierarhic al SC.

(iv) U - mulțimea mărimilor de comandă; Mărimile de comandă sunt mărimi binare, generate de către SC pe baza unei stragii de conducere, mărimi care determină comportarea (evoluția) procesului condus. Această mulțime se reprezentă vectorial astfel: $\mathbf{u}^T=|u_1\ u_2\ \ldots\ u_{|T|}|$, $u_i \in U$, $0 \le i \le |T|$. În ipoteza că rețeaua este controlată, corespondența dintre elementele vectorului **u** și tranzițiile rețelei este dată prin indexarea din reprezentarea vectorială. Fiecare mărime de comandă va fi determinată pe baza unei funcții de comandă $U_j: X \times W \to \{0,1\}$, astfel încât $u_j=U_j$. X este mulțimea mărimilor de stare, definită mai jos.

(i) Z - mulțimea mărimilor de ieșire; Mărimile de ieșire sunt mărimi binare prin care se realizează interacțiunea cu alte sisteme de conducere sau cu un operator uman. Într-o reprezentare vectorială vectorul corespunzător se va nota prin z. Prin mărimile de ieșire pot fi realizate următoarele categorii de interacțiuni:

1) interacțiuni de concesionare cu OPT de pe același nivel ierarhic prin mărimi de concesionare;
2) interacțiuni de reacție - în care caz SC furnizează mărimi de reacție spre SC de pe un nivel ierarhic superior
3) interacțiuni prin mărimi de apreciere pentru un operator uman prin intermediul unui sistem MMI (Man Mashine Interface). SC prezintă către MMI mărimile care descriu fenomenele de la SC "în jos", într-o structură ierarhică. Mărimile de apreciere vor conține mărimile de reacție, mărimile de comandă și *mărimile de stare*.

Mărimile de stare, în cazul modelării prin rețele Petri, sunt specificate prin marcajul rețelei. Mărimile de stare vor fi referite în continuare prin vectorul $\mathbf{x}^T=|x_1\ x_2\ \dots\ x_{|T|}|$, $x_i \in X$, $0 \le i \le |T|$. Pentru o mPTN mărimile de stare sunt date de relațiile $x_i=M(p_i)$. Corespondența dintre elementele vectorului **x** și locațiile rețelei este specificată prin indexarea dată de reprezentarea vectorială.

6.4.2 Mărimi de execuție și mărimi de activare. Observator de evenimente.

Module funcționale specifice ale sistemului de conducere implementează predicatele tranziției și predicatele activării, notate prin H_j respectiv A_j. În continuare se va considera că modulele funcționale care implementează predicatele tranziției și predicatele activării nu sunt incluse în sistemul de conducere. În acest caz, din punctul de vedere al sisetmului de comandă, aceste predicate furnizează mărimi de reacție denumite *mărimi de execuție* repectiv *mărimi de activare*.

Mărimile de execuție sunt mărimi binare generate în momentul execuției acțiunii modelate de către tranziția dată. Vectorul $\mathbf{h}^T=|h_1\ h_2\ \dots\ h_{|T|}|$ specifică mulțimea mărimilor de execuție. Elementele vectorului **h** sunt tocmai predicatele tranzițiilor: $\mathbf{h}[j]=H_j$, unde $H_j:Y_h \to \{0,1\}$ este funcția care implementează predicatul tranzitiei. Corespondența dintre elementele vectorului **h** și tranzițiile rețelei este specificată prin indexarea dată de reprezentarea vectorială.

Mărimile de activare sunt mărimi binare generate în momentul activării funcțiunilor de bază care realizează acțiunea modelată de către tranziția dată. Vectorul $\mathbf{a}^T=|a_1\ a_2\ \dots\ a_{|T|}|$ specifică mulțimea mărimilor de activare. Elementele vectorului **a** sunt tocmai predicatele activării: $\mathbf{a}[j]=A_j$, unde $A_j:Y_a \to \{0,1\}$ este funcția ce implementează predicatul activării. Corespondența dintre elementele vectorului **a** și tranzițiile rețelei este specificată prin indexarea dată de reprezentarea vectorială.

Punând în evidență și modulele funcționale care furnizează mărimile de execuție și de activare, schema bloc a sistemului de automatizare, în cazul unei abordări discrete orientate pe evenimente, se modifică conform Figurii 6-12

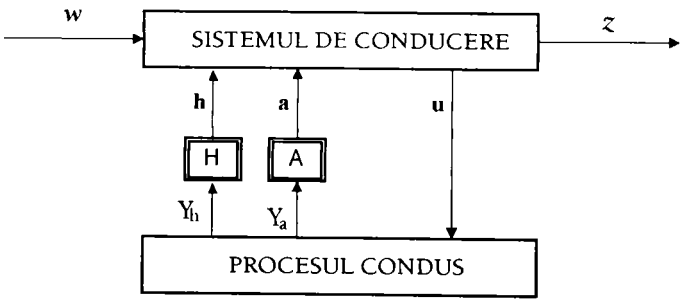


Fig. 6-12 Schema bloc cu observatori de evenimente

Blocurile neliniare, reprezentate prin derptunghiuri desenate cu linie dublă şi notate prin H respectiv A, implementează predicatele tranziţiei respectiv ale activării şi sunt de fapt nişte *observatori de evenimente.*

Structurile prezentate prin schemele bloc din Figura 6-11 respectiv 6-12 arată că şi în acest caz se aplică principiul de bază al reacţiei inverse. Spre deosebire de cazul buclelor de reglare în acest caz se lucrează cu semnale binare în număr mare şi nu există o lege de comandă aplicabilă pe scară largă, prin parametrizare, ca de exemplu în cazul regulatoarelor PID.

Reacţia inversă poate fi realizată pe baza pe baza modelului de reţea Petri al sistemului de conducere. Acest model trebuie astfel elaborat încât să asigure o cât mai mare flexibilitate ceea ce se traduce prin asigurarea unei controlabilităţi cât mai largi. Un anumit comportament particular al sistemului tehnic se va asigura prin strategia de comandă ca parte integrantă a SC.

SC generează mărimile de comandă pe baza mărimilor de reacţie şi a mărimilor de conducere în conformitate cu o strategia de conducere şi starea sistemului. Pentru ca această strategie să fie corect realizabilă SC trebuie să asigure următoarele:

(1) determinarea stării procesului condus;

(2) refacerea ordonării cauzale a mărimilor de reacţie;

(3) tratarea corectă, în conformitate cu modelul, a tranziţiilor temporizate din procesul condus.

În principiu OPT ar putea furniza SC marcajul locaţiilor pre- şi post mulţimii tranziţiei în cauză. Structurile de conducere bazate pe acest pricipiu ar avea dezavantajul unei comunicaţii interne cu vehicularea unor cantităţi mari de informaţie, problematic de realizat în cazul sistemelor de timp real.

În cazul unor sisteme distribuite, sistemul de comunicaţii de date nu poate garanta livrarea mesajelor în ordinea cauzală de producere a evenimentelor care iniţiază transmisia. Dacă aceste mesaje conţin informaţie de stare pe baza cărora sistemul de conducere elaborează comenzi, nu mai poate fi garantat comportamentul cauzal al sistemului de conducere. Din acest motiv sistemul de conducere trebuie să lucreze pe baza informaţiei apriorice despre structura cauzală a procesului condus.

Dacă SC se foloseşte de informaţia a priori cunoscută, cuprinsă în modelul de reţea Petri comunicaţia dintre SC şi procesul condus se poate realiza prin mărimile de reacţie **h** şi **a** care sunt evenimente în procesul condus. În acest caz comunicaţia este minimală deoarece se transmite informaţia minimală (identificatorul evenimentului care a avut loc) numai în momentul apariţiei unui eveniment. Cu alte cuvinte SC lucrează orientat pe evenimente,

conducerea procesului este orientată pe evenimente iar sistemul de automatizare este pilotat de evenimente.

6.4.3 Determinarea stării procesului condus

Dacă considerăm un sistem distribuit în care trebuie controlat un proces pe baza unei strategii de comandă, specificată într-un anumit fel, atunci această strategie trebuie să dispună de starea reală a procesului condus.

Marcajele reţelei care modelează procesul condus reprezintă stări globale ale acestuia. SC poate determina starea procesului condus după execuţia tranziţiei t_j pe baza relaţiei de calcul din regula de tranziţie astfel:

$$\mathbf{m'=m+(t_j^+ + t_j^-)} \tag{6.1}$$

În relaţia de mai sus **m** şi **m'** sunt marcaje ale reţelei care modelează procesul condus iar $\mathbf{t}_j^+$ şi $\mathbf{t}_j^-$ sunt vectori de tranziţie. Pentru a face distincţia dintre marcajul reţelei şi starea procesului determinată de către SC aceasta din urmă se va nota cu x. La recepţionarea informaţiei asupra producerii de evenimente în procesul condus, informaţie prezentă în vectorul **h** prin elementele nenule ale acestuia, SC trebuie să execute următoarele:

(i) preluarea vectorului h (a mărimilor măsurate);

(ii) execuţia prelucrărilor informaţionale necesare determinării stării procesului condus şi a vectorului de concesionare;

(iii) actualizare mărimilor de comandă.

Aceste acţiuni ale SC au o anumită durată caracteristică. Din acest motiv SC trebuie să accepte, în vectorul **h**, prezenţa a mai multor tranziţii executate (a mai multor evenimente apărute în procesul condus). Din punctul de vedere al SC, în acest caz, în procesul condus s-a executat un *pas*. Se defineşte vectorul **p** al pasului realizat prin elementele

$$\mathbf{p}[j] = \begin{cases} 1 & \text{daca tranzitia } \mathbf{t}_j \text{ s-a executat} \\ 0 & \text{daca tranzitia } \mathbf{t}_j \text{ nu s-a executat} \end{cases} \tag{6.2}$$

Având în vedere relaţia de calcul din regula de tranziţie, dacă la starea x s-a produs pasul **p** atunci starea x^+, rezultată ca urmare a execuţiei pasului **p** se determină cu relaţia:

$$\mathbf{x}^+ = \mathbf{x} + \sum_{\mathbf{p}[j]=1} (\mathbf{t}_j^+ + \mathbf{t}_j^-) \tag{6.3}$$

Relaţia (6.3) poate fi exprimată în mod echivalent cu ajutorul matricei de incidenţe astfel:

$$\mathbf{x^+=x+N \cdot p} \tag{6.4}$$

Interpretarea corectă a acestei relaţii, în contextul unei abordări orientate pe evenimente, are în vedere următoarele considerente:

- Un eveniment este o schimbare de stare într-un sistem fizic, schimbare de stare perceptibilă de către un observator al acelui sistem.
- Obsevatorul poate obţine informaţie binară referitoare la eveniment (evenimentul s-a produs / nu s-a produs) adică asupra schimbării de stare (producerii unui fenomen fizic).

- Deoarece fenomenele fizice se produc în timp, între două apariții succesive ale aceluiași eveniment se va găsi întotdeauna un interval de timp.
- În sisteme reale evenimente distincte pot fi observate ca fiind simultane.

Relația (6.4) este ecuația de stare a procesului modelat și exprimă faptul că dacă SC cunoaște structura rețelei care modelează procesul condus – conform definiției 3-29 – atunci, la producerea oricărui/oricăror evenimente, poate determina starea rezultată a procesului condus pe baza stării de dinaintea apariției acestor eenimente. Figura 6-13 prezintă schema bloc structurală a elementului din SC care determină starea procesului condus pe baza relației (6.4).

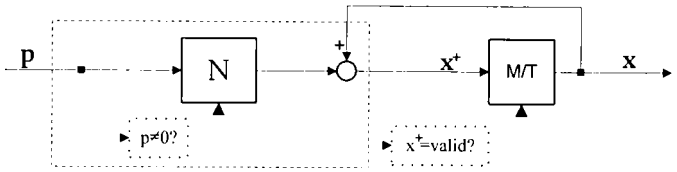


Fig. 6-13 Schema bloc structurală pentru determinarea mărimilor de stare

În Figura 6-13 vectorul $\boldsymbol{x}$ reprezintă starea procesului după producerea ultimului eveniment. Această stare este memorată de către modulul M/T.

Partea încadrată cu linie întreruptă cuprinde cunoștințele SC despre modelul de rețea Petri al procesului condus – prin elementele definiției 3-29 – respectiv modul în care se determină starea rezultată după producerea de evenimente – conform relației 6.4.

Informația asupra producerii de evenimente în procesul condus este recepționată de către SC în vectorul **p**. Această informație are un caracter binar, adică $\mathbf{p}[i] \in \{0,1\}$.

Procesul de determinare a noii stări, notată prin $\boldsymbol{x}^+$, este inițiat odată cu recepționarea unui/unor evenimente în vectorul **p**, adică odată cu realizarea condiției $\mathbf{p} \neq \mathbf{0}$. Acest fapt este simbolizat în Figura 6-13 prin modulul funcțional notat prin "p≠0?" reprezentat prin linie punctată.

După ce s-a determinat, noua stare este transferată în elementul de "transfer și memorare" notat prin M/T în momentul în care $\boldsymbol{x}^+$ devine valid, moment determinat de de modulul funcțional notat prin "$\boldsymbol{x}^+$=valid" și reprezentat în Figura 6-13 prin linie punctată.

În scopul simplificării reprezentărilor grafice, în cele ce urmează, se vor reprezenta doar elementele fluxului informațional iar cele de control ale acestuia ("p≠0?", "$\boldsymbol{x}^+$=valid?") se vor omite înțelegând însă modul de funcționare prezentat mai sus.

Pentru a putea sta la baza unor acțiuni (calcule), informația asupra producerii de evenimente, trebuie, într-un fel sau altul, "materializată". Pentru aceasta se presupune că fiecare eveniment ce se poate produce în procesul condus are propriul observator – prevăzut prin proiectare și instalat chiar în procesul condus – care emite un semnal (impuls) sau un mesaj ori de câte ori apare (se produce) evenimentul dat. În schema bloc din Figura 6-12 acești observatori s-au notat prin H respectiv A.

Din cauza acestei materializări nu mai pot fi detectate apariţii oricât de apropiate ale aceluiaşi eveniment. Cu alte cuvinte caracteristicile observatorilor de evenimente sunt limitative în privinţa dinamicii maxime a sistemului ce se doreşte a fi controlat.

Semnalele sau mesajele corespunzătoare producerii unor evenimente se transmit cu întârzieri variabile la sistemul de conducere care include un observator ce determină starea (discretă) a sistemului controlat conform celor prezentate mai sus.

Sistemulde conducere efectuează calculele din relaţia (6.4) pe baza unor procese fizice cu durată finită. Acest fapt este alt factor limitativ în privinţa capabilităţii de control a acestuia relativ la dinamica maximă a sistemului controlat.

Sistemul de conducere trebuie să asigure interpretarea corectă a evenimentelor în sensul că o apariţie dată a unui eveniment să fie luată în considerare la o singură operaţie de înmulţire în relaţia (6.4). Structura din Figura 6-13 satisface această condiţie. Se observă faptul că determinarea stării procesului condus este orientată pe evenimente în sensul că este iniţiată odată cu detectarea de evenimente în pasul **p**.

6.4.4 Refacerea ordinii cauzale a mărimilor de reacţie

Din cauza întârzierilor variabile în transmiterea informaţiei asupra producerii de evenimente nu se poate garanta livrarea în ordinea de apariţie cauzală proprie procesului condus. Această ordine trebuie refăcută de către controler deoarece în caz contrar controlerul nu ar obţine starea corectă. Ordinea cauzală corectă se poate reface pe baza informaţiei conţinute în modelul de reţea Petri al procesului cu ajutorul vectorului de concesionare **g** din maşina virtuală de reţea Petri.

Relaţia (6.4) furnizează starea corectă a procesului condus doar dacă paşii pe baza cărora se face calculul conţin tranziţiile executate (evenimentele) în ordinea cauzală în care s-au produs. În cazul evenimentelor ordonate cauzal (neconcurente) ordinea de apariţie a lor în pasul **p** trebuie să respecte ordinea reală de apariţie indiferent de valoarea întârzierilor suferite în cursul transmisiei mărimilor de execuţie, deci indiferent de ordinea de apariţie a acestora în vectorul **h**.

SC nu poate utiliza direct mărimile de execuţie din vectorul **h** deoarece din cauza întârzierilor în canalele de comunicaţie ordinea de apariţie în vectorul **h** a mărimilor de execuţie poate să nu respecte ordinea efectivă, cauzală în care s-au produs în procesul condus. Din acest motiv este nevoie de refacerea ordinii cauzale.

SC poate executa aceasta pe baza relaţiei de concesionare din regula de tranziţie. Conform acestei reguli tranziţia $\mathbf{t}_j$ este concesionată la marcajul **m** dacă este satisfăcută relaţia

$$-\mathbf{t}_j^- \le \mathbf{m} \le \mathbf{k} - (\mathbf{t}_j^+ + \mathbf{t}_j^-) \tag{6.5}$$

În cazul SC această relaţie se modifică astfel

$$-\mathbf{t}_j^- \le \mathbf{x} \le \mathbf{k} - (\mathbf{t}_j^+ + \mathbf{t}_j^-) \tag{6.6}$$

În concluzie, cunoscând structura reţelei, adică matricea de incidenţe N şi vectorul k al capacităţilor, SC poate determina tranziţiile concesionate. Definim vectorul de concesionare **g** prin elementele

$$g[j] = \begin{cases} 1 & \text{daca tranzitia } \mathbf{t}_j \text{ este concesionata la starea data} \\ 0 & \text{daca tranzitia } \mathbf{t}_j \text{ nu este concesionata la starea data} \end{cases} \tag{6.7}$$

Acest vector precizează tranziţiile concesionate în procesul condus. Cu alte cuvinte vectorul **g** precizează evenimentele care urmează să apară conform relaţiilor cauzale încorporate în modelul de reţea Petri al procesului. O operaţie de conjuncţie logică dintre vectorii **g** şi **h** va furniza un vector **p** care specifică ***pasul cauzal corect*** în reţeaua care modelează sistemul. Deci:

$$\mathbf{p}=\mathbf{g}\wedge\mathbf{h}, \quad \text{cu } \mathbf{p}[i]=\mathbf{g}[i]\wedge\mathbf{h}[i],\ 0<i\leq|T| \tag{6.8}$$

Definind o funcţie G(x) care determină elementele vectorului de concesionare, pe baza relaţiei (6.6), schema bloc structurală din Figura 6-13 se completează conform Figurii 6-14 în care blocul notat "&" realizează operaţia de conjuncţie logică a doi vectori potrivit relaţiei (6.8).

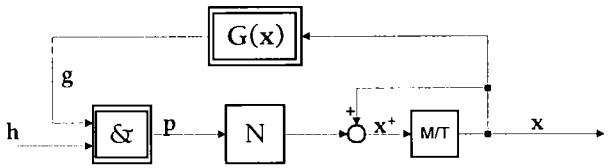


Fig. 6-14 Schema bloc structurală pentru maşina virtuală de reţea Petri

Pentru diferenţierea funcţională, blocurile nou introduse în Figura 6-14 s-au reprezentat prin dreptunghiuri dublu încadrate. Structura funcţională din Figura 6-14 se va numi "***maşină virtuală de reţea Petri***" (abreviat MVRP).

Rolul maşinii virtuale de reţea Petri este tocmai de a pune la dispoziţia unei strategii de comandă starea corectă a procesului condus refăcută pe baza informaţiilor asupra relaţiilor cauzale din procesul condus. În acest sens maşina virtuală de reţea Petri este un observator de stare al procesului condus. Acest fapt ar justifica denumirea de "*observator de stare Petri*". Optăm totuşi pentru prima denumire de "maşină virtuală de reţea Petri" din următoarele două considerente:

(a) Maşina virtuală de reţea Petri este un "*sistem generic*" pentru o anumită clasă de reţele Petri (reţele Petri Locaţie/Tranziţie, ca şi în cazul de faţă, Reţele Petri colorate (v. cap 10), reţele C/E, ş,a.).

(b) Maşina virtuală de reţea Petri este *virtuală* atât din punctul de vedere al proiectantului de sistem cât şi al celui care o implementează întrucât aceştia o abordează doar prin intermediul matricei de incidenţe şi a structurii funcţionale din Figura 6-14. Aşa cum a fost definită, MVRP permite realizarea de sisteme tehnice (hardware sau software), cu caracter general, nededicate unei anumite aplicaţii, a unui anumit model de reţea Petri, ci utilizabile pentru orice model de reţea Petri dintr-o clasă dată. Odată implemenată pe baza definiţiei date în prezenta lucrare proiectantul sistemului de automatizare nu se va confrunta cu alte detalii de implementare.

Cele de mai sus impun următoarea concluzie importantă:

Folosind o implementare a maşinii virtuale de reţea Petri, structurată funcţional ca în Figura 6-14, un sistem de conducere poate genera mărimi de comandă în concordanţă cu relaţiile cauzale din procesele conduse descrise prin modele de reţele Petri.

Din cauza duratelor finite de execuție a calculelor, mașina virtuală de rețea Petri poate observa ca simultane evenimente care de fapt nu s-au produs simultan. Cu toate acestea ea va furniza strategiei de comandă starea finală corectă rezultată în urma execuției tranzițiilor ce au generat evenimentele date.

Proiectantul care realizează sistemul trebuie să fie conștient de limitările rezultate din acest fapt și să-și definească performanțele controlerului în acord cu dinamica procesului condus.

Mașina virtuală de rețea Petri, prin faptul că reface ordinea cauzală corectă a evenimentelor ce s-au produs în procesul condus și prin faptul că pune la dispoziție starea corectă a acestuia, nu este pur și simplu implementarea unei rețele Petri. Ea trebuie privită în contextul structurii sistemului de automatizare. Așa cum se va arăta mai jos, pe baza ei se realizează o structurare clară, foarte bine definită a sistemului de automatizare, structurare deosebit de importantă din punctul de vedere al ingineriei sistemului dat.

Capitolele 8, 9 și 10 vor demonstra faptul că mașina virtuală de rețea Petri permite aceeași structurare clară a *sistemelor distribuite*, în acord cu principiile de structurare pe niveluri ierarhice expuse în capitolul 2 al prezentei lucrări. Este deosebit de important, din punctul de vedere al ingineriei sistemului, faptul că se obțin interfețe bine definite între diversele sisteme de comandă, interfețe ce permit și specificarea performanțelor necesare sistemelor de comandă respectiv a vitezelor de comunicație necesare.

6.4.5 Schema bloc structurală a sistemului de automatizare

Notând cu U($\mathbf{x}$,$\mathbf{w}$) o strategie de conducere care realizează o reacție după stare, se obține, pentru sistemul de conducere, schema bloc structurală din Figura 6-15. S-a reprezentat și modulul funcțional, notat cu H, care implementează predicatele tranzițiilor.

Pentru ca procesul să fie corect controlat acesta nu trebuie să devanseze controlerul. Având în vedere faptul că tranzițiile din procesul condus sunt atomice, execuția tranzițiilor controlabile poate fi sincronizată cu evoluția controlerului cu condiția ca semnalul de comandă să transporte informația de comandă pe front respectiv mesajul de comandă să fie interpretat în aceeași manieră. În cazul structurii de conducere bazate pe mașina virtuală de rețea Petri sincronizarea se realizează deosebit de simplu printr-o operație de conjuncție între vectorul de comandă $\mathbf{u}_S$=U($\mathbf{x}$,$\mathbf{w}$) furnizat de către strategia de conducere și vectorul de concesionare $\mathbf{g}$. Se obține astfel vectorul de comandă $\mathbf{u}$, astfel:

$$\mathbf{u}=\mathbf{u}_S \wedge \mathbf{g}=U(\mathbf{x},\mathbf{w}) \wedge \mathbf{g}.$$

În acest fel comenzile vor fi emise doar în momentul în care sunt disponobile componentele corespunzătoare ale vectorului de concesionare.

Figura 6-15 scoate în evidență elementele funcționale de bază ale sistemului de conducere: mașina virtuală de rețea Petri, observatorii de evenimente și strategia de conducere.

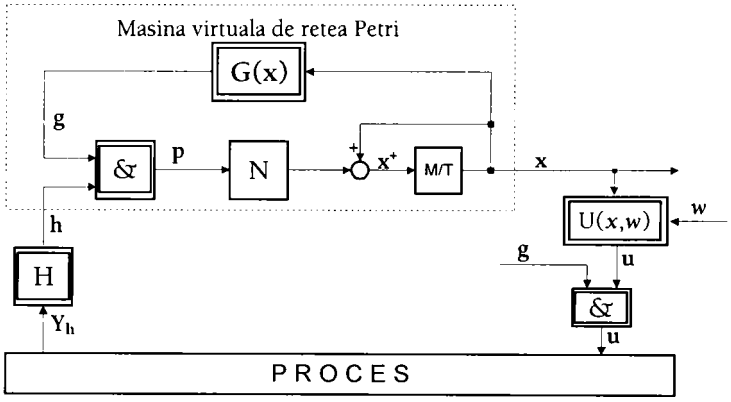


Fig. 6-15 Schema bloc structurală a sistemului de automatizare

Proiectantul sistemului trebuie să implementeze şi operatorii de tranziţie (OPT). Prin urmare schema bloc structurală a sistemului de conducere cu care se confruntă proiectantul este cea din Figura 6-16.

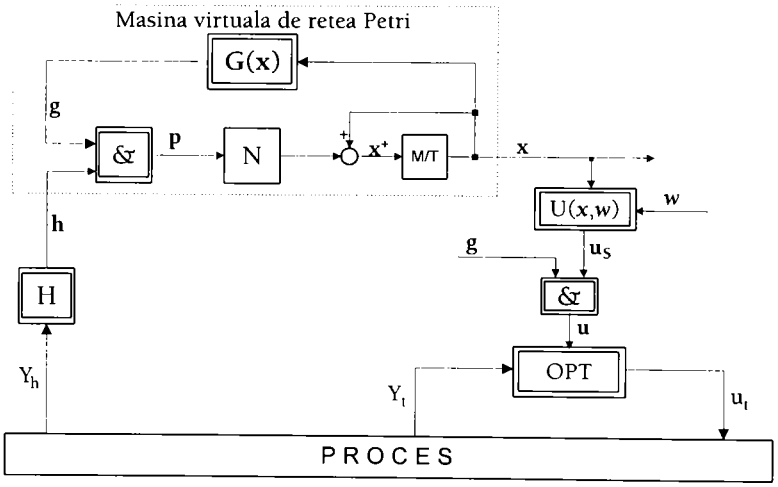


Fig. 6-16 Schema bloc structurală completă a sistemului de automatizare

În Figura 6-16 OPT realizează reacţii inverse locale prin mărimile de reacţie locale Y_t şi mărimile de comandă locală u_t. Acestea din urmă acţionează asupra funcţiunilor de bază din procesul condus.

OPT poate să fie un (sub)sistem dinamic continuu. Prin urmare sistemul ca întreg poate să fie un sistem hibrid.

În practică se poate întâmpla ca să nu fie disponibile toate funcţiunile de bază necesare iar proiectantul trebuie să le realizeze şi pe acestea. Proiectarea respectiv realizarea acestor funcţiuni de bază este mult facilitată de delimitarea funcţională clară rezultată din structurarea pe baza modelului de reţea Petri.

În cele ce urmează, pentru a simplifica reprezentările grafice, în cadrul schemelor bloc structurale nu se vor mai reprezinta OPT şi blocul de conjuncţie logică care realizează u. Primul se va considera inclus în structura procesului condus iar cel de al doilea în structura strategiei de conducere.

6.4.6 STUDIU DE CAZ

Pentru exemplificarea modului în care maşina virtuală de rețea Petri se integrează în structura unui sistem de automatizare, se reconsideră exemplul din paragrful 5.2.1.

- Analiza bazată pe graful de accesibilitate (Fig. 5-2) scoate în evidență stările critice, adică stările din care, dacă se execută anumite tranziții, numite "tranziții critice", sistemul ajunge într-o stare de blocaj. Pentru a evita acest lucru trebuie invalidate tranzițiile critice din stările critice. În capitolul 5 invalidarea s-a realizat prin modificări structurale în modelul inițial de rețea Petri.

  Această modificare structurală are menirea de a demonstra, printr-o nouă analiză, faptul că prin corecția de rețea realizată se asigură proprietățile dinamice dorite (viabilitate, reversibilitate). Aşa cum s-a arătat în capitolul 5 corecția rețelei se realizează prin restricții asupra stărilor realizabile.

- În capitolul 5 nu s-a făcut nici o referire la vre-o modalitatea practică de implementrare a restricțiilor de rețea. În principiu, aceste restricții pot fi implementate prin două metode diametral opuse:

  (a) printr-un controler ce utilizează informația asupra stării globale a procesului condus;

  (b) prin reconfigurări ale procesului prin legături directe între componentele acestuia.

  O analiză comparativă a acestor metode trebuie să aibă în vedere următoarele:

  1. Situațiile conflictuale nu pot fi rezolvate prin modificări structurale ale rețelei Petri. Ele se rezolvă prin informație sosită din mediul exterior ce conține o decizie de validare a uneia dintre acțiuni. În cazul unui sistem de automatizare această informație este furnizată de către un controler (sistem de conducere). Prin urmare, prezența unui controler este şi în acest caz obligatorie.
  2. Schimbările de structură ale rețelei Petri pot fi implementate în procesul condus respectând "relații cauzale" de realizabilitate fizică dar aceasta are ca rezultat o rigidizare a sistemului precum şi asumarea de schimbări în structura inițială a sistemului.

  Această rigidizare este rezultatul faptului că elementele în interacțiune trebuie să fie *în puncte "spațial-vecine"* (v. Figra 3-2), ceea ce presupune legături directe prin canale de comunicație proprii fiecărei interacțiuni. Fără astfel de legături directe sistemul şi-ar pierde caracterul de "sistem orientat pe evenimente" deoarece prin mediul (mediile) de comunicație s-ar transmite şi informații de stare.

  Prin urmare, atât *flexibilitatea* cât şi *integritatea conceptuală*, în sensul orientării pe evenimente, *impun ca restricțiile să fie implementate printr-un controler ce lucrează pe baza unei reacții după stare.*

- În cadrul sistemelor flexibile de fabricaţie este de dorit ca sistemul de automatizare să se bazeze pe un sistem de producţie cu structură cauzală stabilă iar flexibilitatea să fie realizată printr-un controler supervizor pregătit şi optimizat pentru aplicaţii de automatizare.

  Desigur, în ciclul de viaţă al unui sistem de automatizare se poate întâmpla, chiar de mai multe ori, să fie necersare schimbări de structură. Acestea trebuie însă văzute ca schimbări calitative ce se disting de cele operative proprii automatizărilor flexibile.

  Prin urmare sistemul de comandă trebuie să implementeze corecţiile de reţea prin restricţii asupra stărilor realizabile. Restricţiile pot fi realizate cu condiţia ca tranziţiile critice să fie tranziţii controlabile.

- Totodată, sistemul de conducere trebuie să implementeze şi o anumită strategie de control. Deci vectorul de comandă $\boldsymbol{u}$ va avea cel puţin două componente: componenta de corecţie $\boldsymbol{u}_C$ şi componenta de strategie de control $\boldsymbol{u}_S$.

  $$\boldsymbol{u}=\boldsymbol{u}_C \wedge \boldsymbol{u}_S$$

- În cazul exemplului din Figura 5-1, reţeaua Petri având 6 tranziţii numerotate de la 1 la 6, vectorul de corecţie va prezenta, corespunzător, 6 componente, fiind de forma:

  $$\boldsymbol{u}_C=|u_1\ u_2\ u_3\ u_4\ u_5\ u_6|^T.$$

  Componenta de corecţie, $\boldsymbol{u}_C$, a vectorului de comandă trebuie să intervină doar asupra tranziţiilor critice. Tranziţiile critice fiind tranziţiile t_1 şi t_4, dacă se lucrează pe baza relaţiei (1), elementele "don't care" în vectorul $\boldsymbol{u}_C$, pot fi înlocuite cu "1" logic şi se obţine vectorul:

  $$\boldsymbol{u}_C=|u_1\ 1\ 1\ u_4\ 1\ 1|^T,$$

  pentru care elementele $\boldsymbol{u}_C[1]$ şi $\boldsymbol{u}_C[4]$ se determină astfel:

  $$\boldsymbol{u}_C[1]=\begin{cases} 0 & \text{dacă } \boldsymbol{m}=|1\ 1\ 0\ 2\ 0\ 0\ 0\ 1|^T \\ 1 & \text{altfel.} \end{cases}$$

  $$\boldsymbol{u}_C[4]=\begin{cases} 0 & \text{dacă } \boldsymbol{m}=|2\ 0\ 0\ 1\ 1\ 0\ 1\ 0|^T \\ 1 & \text{altfel.} \end{cases}$$

  Se observă că stările critice din sistemul (modelul) iniţial sunt şi stări conflictuale între tranziţiile t_1 şi t_5 respectiv t_2 şi t_4. Strategia de comandă care implementează funcţiile de mai sus desfiinţează aceste stări conflictuale.

  Pentru ca funcţiile $\boldsymbol{u}_C[1]$ şi $\boldsymbol{u}_C[4]$ să fie "on-line" evaluabile strategia de comandă trebuie să dispună de starea globală a sistemului controlat. Această stare globală, exprimată prin vectorul $\boldsymbol{x}$, este furnizată de maşina virtuală de reţea Petri.

- Anexa 2 prezintă o posibilă implementare a aspectelor de mai sus sub forma unui program MATLAB. Programul pune la dispoziţie toate componenetele sistemului: modelul de reţea Petri al procesului condus, mărimile de reacţie (vectorul $\boldsymbol{h}$), maşina virtuală de reţea Petri şi strategia de comandă.

  Programul prezintă o interfaţă grafică interactivă simplă şi este definit pentru tratarea a şase modele de reţea Petri. Modelul cu care se lucrează trebuie în prealabil selectat cu

ajutorul butonului de selecție "***Select Net***". Acționând butonul "***Netdef***" programul oferă spre editare matricele și vectorii ce descriu modelul de rețea Petri selectat.

Acționând butonul "***Control Def***" programul oferă spre editare modulul de program care implementează strategia de conducere corespunzătoare modelului de rețea Petri selectat.

Toate acțiunile ce urmează se aplică rețelei selectate prin butonul "***Select Net***".

Acționând butonul "***Init***" programul va inițializa modelul de rețea Petri al procesului respectiv mașina virtuală de rețea Petri (MVRP).

Acționând butonul "***Control***" se generează vectorul de comadă, ***u***, conform strategiei de conducere și a tranzițiilor concesionate din cadrul situațiilor conflictuale. Prin acționarea tastei "***Edit u***" programul oferă spre editare vectorul de comandă permițând astfel o intervenție directă asupra acestuia. Prin acționarea butonului "**H**" se generează vectorul ***h*** într-o manieră cvasialeatoare în sensul că dintre tranzițiile concesionate se consideră a fi executate doar unele, alegerea făcându-se cvasialeator. Prin acționarea tastei "***Edit h***" programul oferă spre editare vectorul de execuție permițând astfel o intervenție directă asupra acestuia.

Pentru a rezolva sitațiile de conflict prin mărimi de conducere s-au prevăzut butoane de selecție notate "***CONFLICT XX***" prin care se specifică tranziția ce se va executa, pentru fiecare situație conflictuală în parte. În cazul exemplului prezentat în lucrarea de față sunt trei situații conflictuale. S-au prevăzut totuși șase butoane de selecție în ideea unor aplicații mai complexe ce ar putea să le solicite.

Prin acționarea butoanelor "***Pas proces***" respectiv "***Pas MVRP***" se execută pasul corespunzător vectorului ***h*** în modelul de rețea Petri al procesului respectiv în MVRP.

Pentru a putea urmării evoluția sistemului, programul afișează, după fiecare, acțiune următoarele:

a) vectorul de stare al procesului (al modelului de rețea Petri), (XP)

b) vectorul de stare furnizat de MVRP, (X);

c) vectorul de concesionare al procesului, (pg);

d) vectorul de concesionare al MVRP, (g);

e) vectrorul de comandă, (u);

f) vectorul de execuție, (h).

Modulele ***Init, Control, H, Pas Proces*** și ***Pas MVRP*** sunt module genrice și încorporează funcții generice privind controlul proceselor bazat pe modelul de rețea Petri. În acest sens creează o mașină virtuală.

Elementele specifice unei rețele date sunt tratate prin modulele ***Netdef*** (definirea rețelei), ***Control Def*** (definirea strategiei de comandă) și modulele ***CONFLICT XX*** (rezolvarea conflictelor prin mărimi de conducere). Modulele ***Edit u*** și ***Edit h*** oferă o alternativă de intervenție directă, "manuală" asupra derulării proceselor, utilă în situații de simulare.

- Programul permite studiul efectului strategiilor de comadă și a mărimilor de conducere asupra comportamentului în buclă închisă. Bunăoară, pentru exemplul din paragraful 5.2.1 "Net1" utilizează matricea de incidențe a rețelei corectate iar "Net2" utilizează

matricea de incicenţe a reţelei necorectate. În cazul "Net2" corecţia se realizează prin strategia de comandă implementată prin modulul "control.m".

Validarea programului este imediată rezultând prin prin compararea cu graful accesibil din Figura 5-3b. Se obţine aceeaşi succesiune de stări şi comenzi.

6.5 Variante ale structurii de conducere

6.5.1 Utilizarea mărimilor de activare şi de execuţie în generarea comenzilor

Mărimile de activare se utilizează în situaţia în care strategia de conducere generează comenzi care depind de faptul dacă un eveniment, legat de iniţierea unei acţiuni, s-a produs sau nu s-a produs. Cu alte cuvinte strategia surprinde o anumită istorie a evoluţiei sistemului. Dacă strategia de conducere utilizează şi mărimile de activare SC trebuie să refacă ordinea cauzal-corectă a acestora. Procedeul este acelaşi cu cel al generării mărimilor de execuţie, adică se va determina un pas de activare q conform relaţiei:

$$q=g\wedge a, \qquad \text{cu } q[i]=g[i]\wedge a[i],\ 0<i\leq|T| \tag{6.9}$$

Pentru a furniza strategiei de conducere informaţii care să specifice acţiunile iniţiate, reţeaua dată se completează cu câte o locaţie pentru fiecare eveniment ce se utilizează în generarea comenzilor. Aceste locaţii suplimentare se leagă de tranziţiile reţelei prin arce orientate şi sunt purtătoare a mărimilor de reacţie suplimentare ce formează vectorul **d** conform următoarei relaţii:

$$\mathbf{d}^{+}=\mathbf{d}+\begin{bmatrix}\mathbf{C}_p & \mathbf{C}_q\end{bmatrix}\cdot\begin{bmatrix}\mathbf{p}\\ \mathbf{q}\end{bmatrix} \tag{6.10}$$

Figura 6-17 prezintă schema bloc structurală a sistemului de conducere rezultat unde blocul notat prin ":" semnifică operaţia de agregare a vectorilor **p** şi **q** conform relaţiei de mai sus. Matricea $\begin{bmatrix}\mathbf{C}_p & \mathbf{C}_q\end{bmatrix}$ este parte integrantă a strategiei de conducere. Locaţiile suplimentare marcate semnifică evenimente apărute în procesul condus. Din acest motiv în starea iniţială ele nu sunt marcate, prin urmare nu sunt semnificative din punctul de vedere al stării iniţiale.

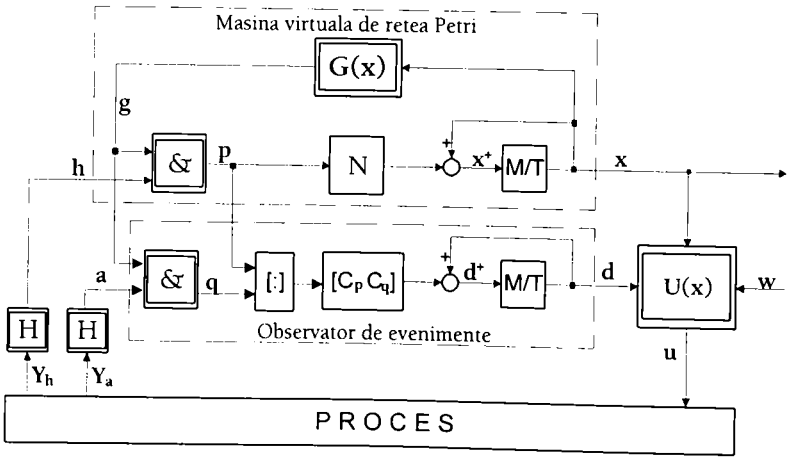


Fig. 6-17 Schema bloc structurală cu utilizarea mărimilor de activare şi de execuţie

Mărimile de activare pot fi utilizate în acest fel doar dacă prin utilizarea lor strategia de conducere nu creează conflicte, adică nu utilizează o aceeaşi locaţie suplimentar introdusă pentru validarea a două sau mai multor tranziţii. Dacă o astfel de situaţie poate să apară este mai bine să se preia mărimile de activare în structura modelului de reţea Petri.

6.5.2 Preluarea mărimilor de activare în modelul de reţea Petri

În situaţia în care în modelul de reţea Petri al procesului toate mărimile de activare sunt reprezentate prin tranziţii acestea vor fi înlocuite prin mărimi de execuţie (predicatele activării se transformă în predicate ale tranziţiilor) şi deci vectorul a nu mai apare în relaţii şi scheme bloc structurale. Această modificare nu va influenţa proprietăţile reţelei (viabilitate, reversibilitate, invarianţi) dar creşte dimensiunile acesteia cu câte o tranziţie şi o locaţie pentru fiecare mărime de activare respectiv cu câte o linie şi o coloană în matricea de incidenţe.

Modificarea poate fi justificată prin următorul argument stilistic: *orice eveniment semnificativ din procesul modelat se reprezintă printr-o tranziţie!*

Dacă se adoptă această regulă de modelare, relaţia (6.10) devine

$$\mathbf{d}^+ = \mathbf{d} + \mathbf{L} \cdot \mathbf{p} \tag{6.11}$$

iar vectorul de stare obţinut prin agregarea vectorilor x şi d se exprimă prin relaţia

$$\begin{bmatrix} \mathbf{x}^+ \\ \mathbf{d}^+ \end{bmatrix} = \begin{bmatrix} \mathbf{x} \\ \mathbf{d} \end{bmatrix} + \begin{bmatrix} \mathbf{N} \\ \mathbf{L} \end{bmatrix} \cdot \mathbf{p} \tag{6.12}$$

unde L este o matrice ce surprinde locaţiile suplimentare a căror marcaj semnifică producerea unui anumit eveniment. Matricea **L** este parte integrantă a strategiei de conducere. Schema bloc structurală devine cea din Figura 6-18.

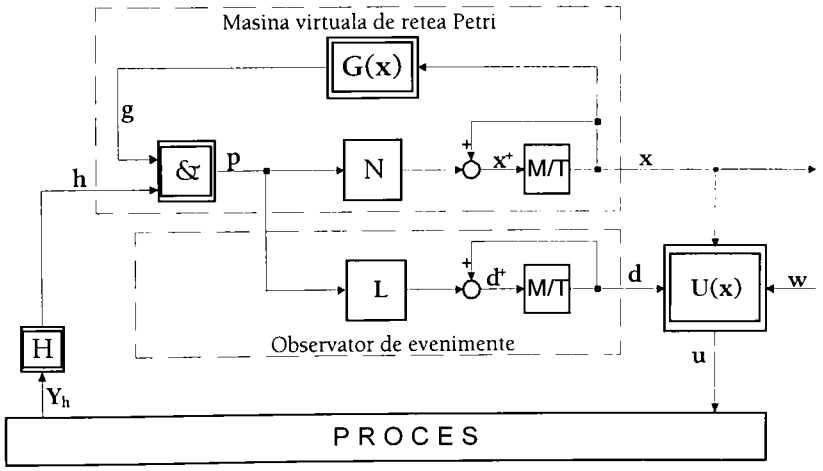


Fig. 6-18 Schema bloc structurală după preluarea mărimilor de activare în structura reţelei Petri

În principiu locaţiile suplimentare pot fi integrate în maşina virtuală de reţea Petri dacă vectorul mărimilor de stare se exprimă prin $[\mathbf{x}\ \mathbf{d}]^T$. În acest caz relaţia (6.12) devine:

$$\begin{bmatrix}\mathbf{x}^+\\ \mathbf{d}^+\end{bmatrix}=\begin{bmatrix}\mathbf{x}\\ \mathbf{d}\end{bmatrix}+\begin{bmatrix}\mathbf{N} & \mathbf{0}\\ \mathbf{C}_p & \mathbf{C}_q\end{bmatrix}\cdot\begin{bmatrix}\mathbf{p}\\ \mathbf{q}\end{bmatrix} \tag{6.13}$$

Iar dacă se notează $\begin{bmatrix}\mathbf{x}^+\\ \mathbf{d}^+\end{bmatrix}=\mathbf{x}_C^+,\begin{bmatrix}\mathbf{x}\\ \mathbf{d}\end{bmatrix}=\mathbf{x}_C,\begin{bmatrix}\mathbf{N} & \mathbf{0}\\ \mathbf{C}_p & \mathbf{C}_q\end{bmatrix}=\mathbf{N}_C,\begin{bmatrix}\mathbf{p}\\ \mathbf{q}\end{bmatrix}=\mathbf{p}_C$ (6.14)

se ajunge la relaţia

$$\mathbf{x}^+_C=\mathbf{x}_C+\mathbf{N}_C\cdot\mathbf{p}_C \tag{6.15}$$

respectiv la schema bloc structurală din Figura 6-15.

Structurile de conducere prezentate în Figurile 6-15, 6-16 şi 6-17 sunt echivalente, în sensul că pe baza lor poate fi implementat un acelaş sistem de conducere. Proiectantul va trebui să se decidă asupra structurii pe care o adoptă. Această decizie este determinată de flexibilitatea impusă prin specificaţia sistemului de conducere.

În continuare se va lucra cu structura din Figura 6-15 cea ce nu restrînge generalitatea raţionamentelor ce urmează dar simplifică reprezentările grafice şi relaţiile matematice.

6.5.3 Sisteme de conducere cu comunicaţii ce livrează mărimile de reacţie în ordinea cauzală

În cazul în care prin construcţia sistemului se poate asigura furnizarea mărimilor (semnalelor) din vectorul **h** în ordinea cauzală se poate renunţa la blocurile de conjuncţie. Astfel de sisteme pot fi realizate mai ales în domeniul controlerelor înglobate (embeded controller) sau a controlerelor paralele, sincronizate (sincronized parallel controller). În acest caz se ajunge la structura din Figura 6-19

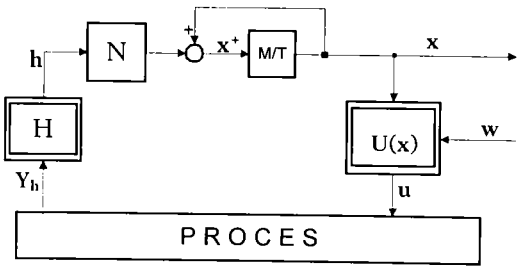


Fig. 6-19 Schema bloc structurală pentru cazul comunicaţiei cauzale

6.6 Aspecte de implementare ale sistemului de conducere

6.6.1 Comunicaţia cu procesul condus

Comunicaţia SC cu mediul său se poate realiza prin *mesaje* sau *semnale*. Comunicaţia cu procesul condus se execută prin mărimile de execuţie **h**, mărimile de activare **a** şi mărimile de comandă **u**.

6.1.1.1 Comunicaţia prin semnale

Mărimile de execuţie semnalizează apariţia evenimentelor corespunzător execuţiei tranziţiilor (a execuţiei acţiunilor corespunzătoare tranziţiilor). În cazul în care sunt transmise prin semnale electrice acestea vor aduce informaţia codificată pe un front.

Fiecare semnal va fi preluat de către un element de memorare (latch). Acesta va fi resetat în momentul în care informaţia a fost preluată pentru calculul stării $\mathbf{x}^*$. Înainte de utilizare se va realiza reordonarea cauzală prin vectorul de concesionare **g**, conform relaţiei: $\mathbf{p}[j] = \mathbf{h}[j] \wedge \mathbf{g}[j]$ aşa cum se prezintă în Figura 6-20

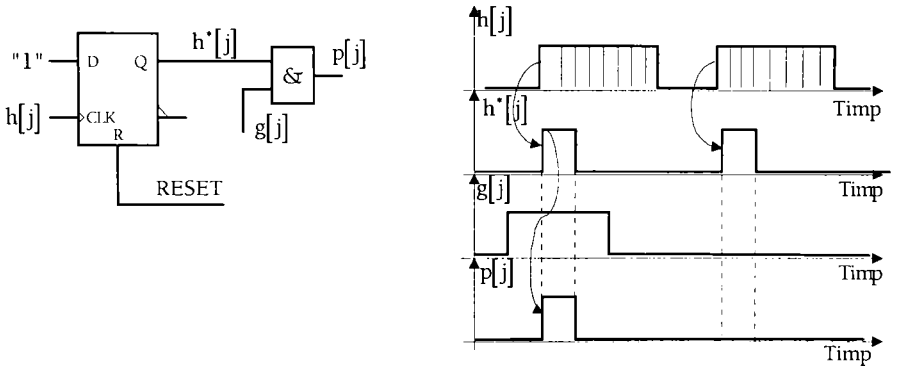


Fig. 6-20 Preluarea mărimilor de execuţie

Mărimile de activare semnalizează iniţierea execuţiei tranziţiilor în urma concesionării şi validării prin comanda **u**. Sunt valabile caracteristicile din cazul mărimilor de execuţie. cu observaţia, că în acest caz elementele de memorare vor fi resetate în momentul apariţiei semnalului **p**[j] corespunzător, aşa cum se prezintă în Figura 6-21.

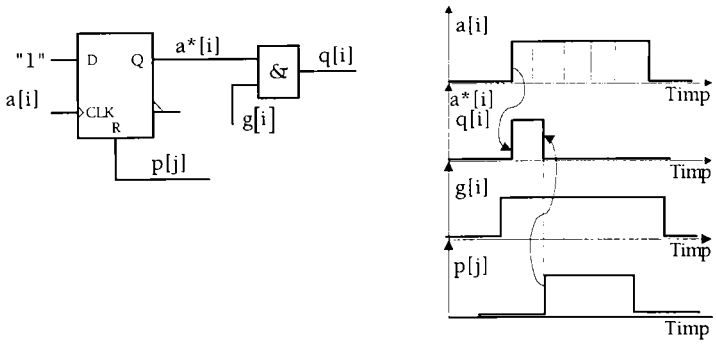


Fig. 6-21 Preluarea mărimilor de activare

În cazul în care mărimile de activare şi de execuţie s-ar transmite prin semnale tip nivel (Moore) ar fi necesar un semnal suplimentar de comunicaţie aşa cum se poate vedea din Figura 6-22.

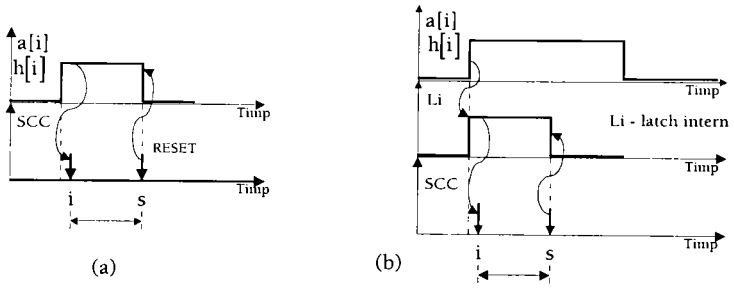


Fig. 6-22 Posibilităţi de comunicare dintre procesul condus şi SC

În cazul (a) la apariţia unuia (unora) dintre semnalele **h** sau **a** în SC se iniţiază un proces de prelucrare în momemtul *i* . După terminarea acestui proces, în momentul s, SC trebuie să emită un impuls de reset către modulul observator al evenimentului dat.

În cazul (b) la apariţia semnalului de reacţie se setează un latch intern din SC după care se iniţiază procesul de prelucrare asociat. La terminarea acestuia latch-ul intern este resetat. Această operaţie fiind internă SC, nu apare necesitatea comunicaţiei suplimentare dintre SC şi observatorul de evenimente.

Marimile de comandă fiind generate printr-o reacţie după stare vor fi de tipul Moore adică informaţia de validare a tranziţiei controlate este transmisă pe nivelul semnalului. Trebuie să se asigure că din aceat motiv nu vor apare validări false. Acestea ar putea apare în situaţia în care tranziţia dată este concesionată înainte ca să fie ridicată comanda unei tranziţii tocmai executate (Figura 6-23).

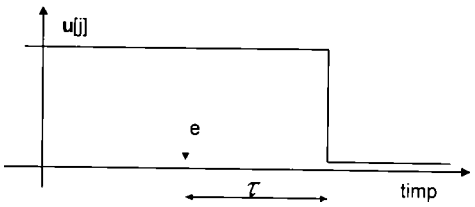


Fig. 6-23 Relativă la posibilitatea concesionărilor nedorite

În Figura 6-23, prin e s-a notat evenimentul care cauzează invalidarea comenzii iar τ este întârzierea cu care se invalidează comanda **u**[j]. Pot fi utilizate semnale Moore pentru transmiterea comenzilor doar dacă se poate asigura ca tranziţia corespunzătoare t_j să nu fie concesionată în intervalul τ.

Acest interval poate fi minimizat dacă semnalul de comandă se emite conform schemei de principiu din Figura 6-24.

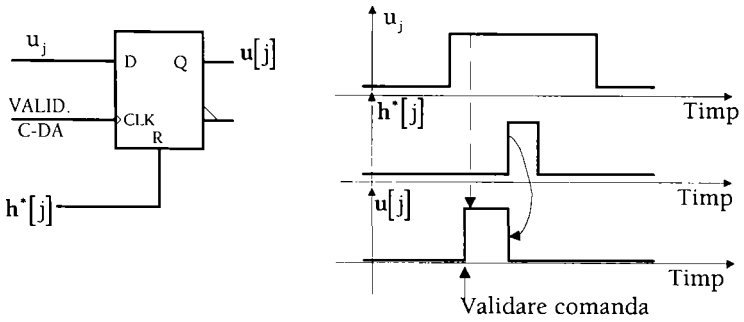


Fig. 6-24 Generarea semnalului de comandă

În această figură u_j este mărimea de comandă elaborată de către strategia de comandă $U(x,w)$ şi se inserează la fiecare pas realizat de către maşina virtuală de reţea Petri printr-un impuls de validare care înscrie valoarea dată a mărimii u_j în elementul de memorare. La ieşirea acestuia apare semnalul de comandă **u**[j]. După execuţia tranziţiei, din mărimea de execuţie **h**[j], maşina virtuală de reţea Petri formează impusul **h***[j] conform Figurii 6-20. Acest impuls va reseta elementul de memorare al semnalului de comandă înainte ca strategia să determine o nouă comandă.

În situaţia în care o comandă **u**[j] trebuie sincronizată cu un eveniment **p**[i], în schema bloc din Figura 6-24, semnalul "VALIDARE COMANDĂ" se va obţine printr-o conjuncţie logică dintre semnalul de validare a comenzii şi semnalul corespunzător elementului **p**[i] al vectorului **p**.

Procedeul prezentat în Figura 6-24 are următoarele avantaje:

(i) se minimizează intervalul de timp cât ar putea apare o validare falsă a tranziţiei în cauză;

(ii) OPT poate extrage mărimea de comandă atât din frontul cât şi din nivelul semnalului de comandă;

(iii) funcţionează corect şi în cazul în care sunt necesare execuţii multiple ale tranziţiei date pentru o singură valoare stabilă a mărimii de comandă u_j.

Caracterul atomic al acţiunilor din procesul condus poate justifica o regulă care impune ca mărimea de comandă să fie codificată pe frontul semnalului de comandă. Prin respectarea acestei reguli se asigură următoarele proprietăţi:

(i) interacţiunea SC cu mediul său se realizează în mod uniform prin codificarea pe frontul semnalelor de interacţiune pentru toate mărimile de interacţiune, adică se păstrează caracterul orientat pe evenimente;

(ii) în cazul comunicaţiei prin mesaje este mult simplu de implementat deoarece o comandă este legată de emiterea doar a unui singur mesaj;

(iii) strategia de comandă poate utiliza, în generarea comenzilor, pe lângă mărimile de stare, şi elemente ale vectorului **p** (evenimente din proces). Această posibilitate simplifică mult eventualele sincronizări necesare în generarea comenzilor.

În acest caz OPT va prelua mărimea de comandă codificată pe frontul semnalului de comandă pe baza schemei de principiu din Figura 6-25. Pe frontul ridicător al semnalului de comandă elementul de memeorare este setat. După execuţia acţiunii asociate tranziţiei date elementul de memorare va fi resetat de către OPT. În acest fel se asigură ca la un front ridicător al impulsului de comandă acţiunea dată să fie executată o singură dată.

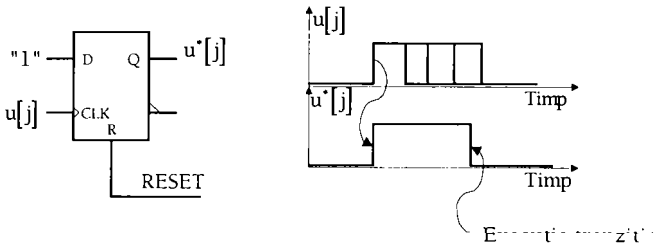


Fig. 6-25 Preluarea semnalului de comandă

Pentru a asigura funcţionarea corectă şi în situaţia în care semnalul de comandă generat prin reacţia după stare trebuie să permită şi execuţii multiple ale acţiunilor, acestea se vor "ridica" şi reînnoii la execuţia fiecărui pas ca în Figura 6-24.

Sincronizarea dintre vectorul de comandă $\boldsymbol{u}$ şi vectorul de concesionare $\boldsymbol{g}$, realizată printr-o conjuncţie logică aşa cum se prezintă în Figura 6-16, va sigura reînnoirea automată a comenzii la terminarea fiecărui ciclu de lucru a maşinii virtuale de reţea Petri.

Observaţie importantă:

> *Acţiunile în procesul condus fiind atomice concesionarea unei tranziţii şi validarea ei prin semnalul de comandă va iniţia execuţia acţiunii. Aceasta nu mai poate fi întreruptă prin schimbarea stării semnalului de comandă. Dacă totuşi, din motive de securitate, trebuie prevăzute posibilităţi de întrerupere ale acţiunilor iniţiate acestea se vor realiza prin mecanisme specifice care nu sunt prezente în modelul de reţea Petri. Prin invalidarea comenzii încă în timpul execuţiei acţiunii în cauză se invalidează o nouă execuţie a ei.*

6.1.1.2 Comunicaţia prin mesaje

În cazul comunicaţiei prin mesaje mărimile de reacţie şi de comandă vor fi reprezentate prin variabile. Se va asigura aceeaş funcţionalitate ca în cazul comunicaţiei prin semnale.

Pentru mărimile de execuţie **h** şi mărimile de activare **a** este de ajuns a se transmite un identificator al tranziţiei şi unul al tipului de semnal (execuţie/activare). În SC aceste mărimi vor fi reprezentate ca variabile ce vor fi tratate astfel:

- Variabila **h***[j] va fi setată, **h***[j]=1, la recepţia mesajului corespunzător şi va fi resetată, **h***[j]=0, la preluarea pentru calculul x^*.
- Variabila **a***[j] va fi setată, **a***[j]=1, la recepţia mesajului corespunzător şi va fi resetată, **a***[j]=0, o dată cu validarea mărimii **p**[j].

Mesajul transmis pentru mărimile de comandă specifică tranziţia care va fi validată. În OPT mărimea de comandă va fi tratată ca o variabilă, astfel:

- Variabila **u***[j] va fi setată, **u***[j]=1, la recepţia mesajului **u**[j] şi va fi resetată, **u***[j]=0, la execuţia acţiunii corespunzătoare tranziţiei t_j.

6.6.2 Determinarea mărimilor de stare

SC trebuie să determine starea procesului condus după fiecare pas **p** realizat conform uneia dintre relaţiile de mai jos:

$$\mathbf{x}^+ = \mathbf{x} + \mathbf{N} \cdot \mathbf{p} \tag{6.16}$$

$$\mathbf{x}^+ = \mathbf{x} + \sum_{\mathbf{p}[j]=1} \left(\mathbf{t}_j^+ + \mathbf{t}_j^- \right) \tag{6.17}$$

$$\mathbf{x}^+ = \mathbf{x} + \sum_{\mathbf{p}[j]=1} \mathbf{t}_j \tag{6.18}$$

Al doilea termen din membrul drept al fiecărei relaţii de mai sus poate fi determinat în două moduri distincte:

(1) pasul **p** este preluat "paralel", ca vector şi apoi se execută operaţia matriceală **N**·**p**;

(2) pasul **p** se prelucrează "secvenţial" element cu element, ţinând cont de faptul că $\mathbf{p}[j] \in [0,1]$ şi executând doar sumele conform relaţiei (6.17) pentru reţelele care conţin bucle autonome sau conform relaţiei (6.18) pentru reţelele care nu conţin bucle autonome.

A doua metodă este mai eficientă atât în cazul unei realizări hardware cât şi a unei realizări software. În momentul în care s-a determinat complet vectorul x^+ acesta, împreună cu un semnal de validare, se pune la dispoziţia modulelor care realizează funcţiile $G(x)$ respectiv $U(x,w)$.

Figura 6-26 prezintă schema bloc a algoritmului pentru determinarea vectorului de stare.

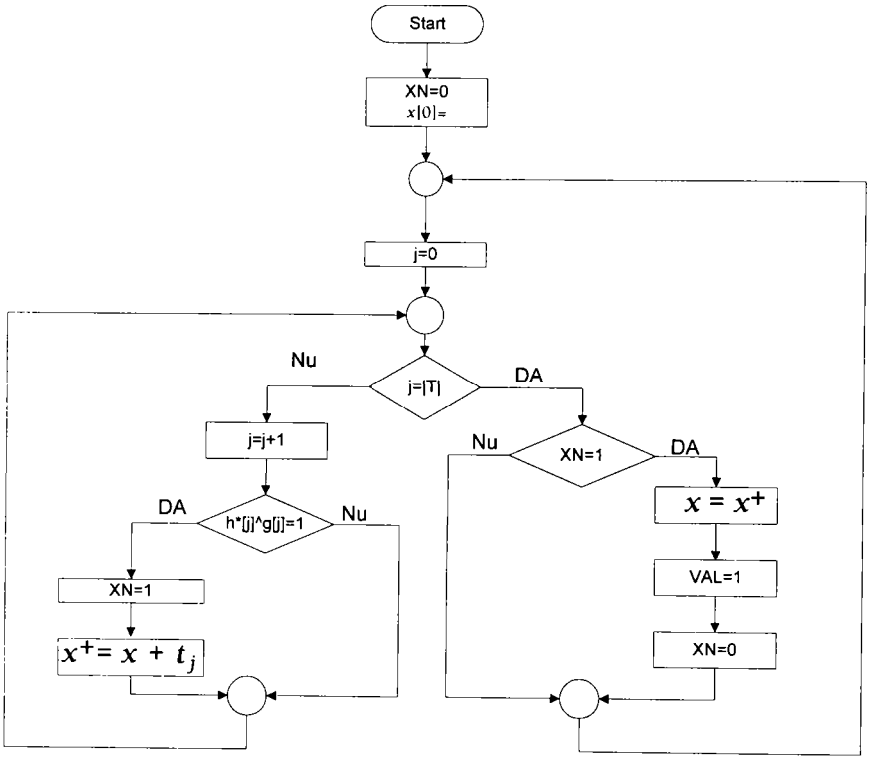


Fig. 6-26 Schema logică a algoritmului pentru determinarea vectorului de stare

În Figura 6-26, XN este variabila care prin valoarea adevărat specifică faptul că x este o valoare nouă a vectorului de stare.

VAL este variabila care, prin valoarea adevărat, specifică faptul că este disponibilă valoarea nouă a vectorului de stare, adică faptul că x^+ este valid.

6.6.3 Determinarea elementelor vectorului de concesionare

Elementele vectorului de concesionare se determină pentru fiecare tranziţie în parte conform următoarei relaţii:

$$\mathbf{g}[j] = \begin{cases} 1 & \text{daca } -\mathbf{t}_j^- \le \mathbf{x} \le \mathbf{k} - (\mathbf{t}_j^+ + \mathbf{t}_j^-) \\ 0 & \text{altfel} \end{cases} \qquad (6.19)$$

Introducând notaţia $\mathbf{d}_j = \mathbf{k} - (\mathbf{t}_j^+ + \mathbf{t}_j^-)$ relaţia de mai sus devine:

$$\mathbf{g}[j] = \begin{cases} 1 & \text{daca } -\mathbf{t}_j^- \le \mathbf{x} \le \mathbf{d}_j \\ 0 & \text{altfel} \end{cases} \qquad (6.20)$$

Această relaţie poate fi exprimată în mod echivalent astfel:

$$\mathbf{g}[j] = \begin{cases} 1 & \text{daca } (-\mathbf{t}_j^- \le \mathbf{x}) \wedge (\mathbf{x} \le \mathbf{d}_j) \\ 0 & \text{altfel} \end{cases} \qquad (6.21)$$

Figura 6-27(a) este reprezentarea grafică pentru schema structurală a modulului funcţional care determină elementele vectorului de concesionare. Acest modul funcţional poate fi explicitat conform Figurii 6-27(b) definind următoarele funcţii:

$$F_{1j}(\mathbf{x}, \mathbf{t}_j^-) = \begin{cases} 1 & \text{daca } \mathbf{x} + \mathbf{t}_j^- \ge 0 \\ 0 & \text{altfel} \end{cases} \qquad (6.22)$$

$$F_{2j}(\mathbf{x}, \mathbf{d}_j) = \begin{cases} 1 & \text{daca } \mathbf{x} + (-\mathbf{d}_j) \le 0 \\ 0 & \text{altfel} \end{cases} \qquad (6.23)$$

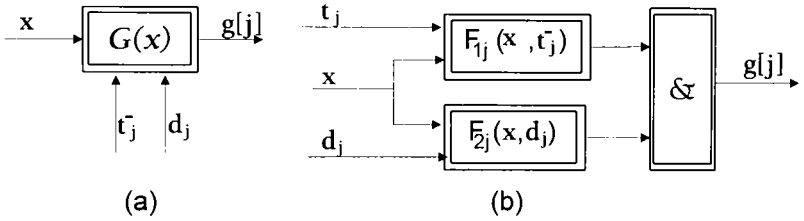


Fig. 6-27 Schema bloc funcţională pentru determinarea vectorului de concesionare

Figura 6-28 prezintă schema logică a algoritmului pentru determinarea vectorului de concesionare.

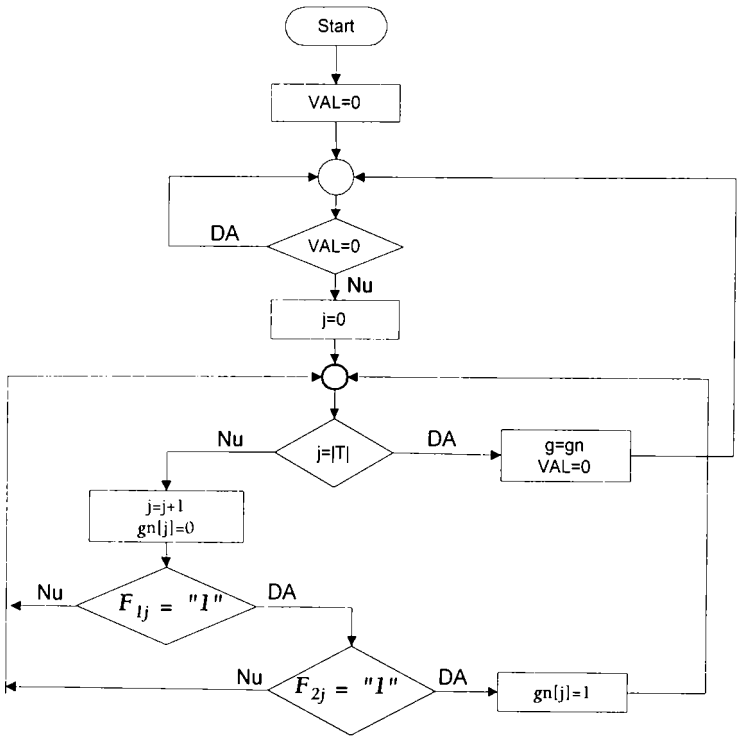


Fig. 6-28 Schema logică a algoritmului pentru determinarea vectorului de concesionare

În Figura 6-28, VAL este o variabilă care validează vectorul de stare x iar g_n este vectorul de concesionare nou, determinat în ciclul curent.

În cazul mPTN se lucrează cu numere întregi deoarece marcajele locaţiilor respectiv ponderile arcelor se exprimă prin numere întregi. Procesoare de 8 biţi permit reprezentarea unor capacităţi a locaţiilor de până la 255, satisfăcătoare pentru cele mai multe aplicaţii practice. Viteza de execuţie este critică deoarece de ea depind în mod direct caracteristicile de timp real ale maşinii virtuale de reţea Petri.

6.6.4 Exemplu de implementare a unei maşini virtuale de reţea Petri

Dacă rezultatele analizei calitative (şi eventual cantitative) ale modelului de reţea Petri sunt în acord cu caracteristicile impuse prin specificaţie atunci se poate trece la implementarea sistemului de conducere şi în cadrul acesteia la implementarea maşinii virtuale de reţea Petri. Dacă se dispune de metode de implementare asistate de calculator acestea asigură o productivitate ridicată şi o implementare fără erori. În cele ce urmează se prezintă o posibilă implementare hardware a unei maşini virtuale de reţea Petri modelată printr-o reţea C/E.

În acest caz cea mai simplă tehnică de implementare a locaţiilor este acea de a aloca câte un bistabil fiecărei locaţii. Se obţine astfel un registru de stare cu un număr de biţi egal cu numărul locaţiilor. În principiu, numărul bistabilelor poate fi redus dacă locaţiilor li se asociază coduri din aceşti bistabili.

Conţinutul registrului de stare se schimbă ca urmare a execuţiei tranziţiilor, conform relaţiei

$$x^+=x+N^+\cdot a+N^-\cdot a \qquad (6.24)$$

unde a este vectorul tranziţiilor executate. Acest vector este de fapt pasul p încărcat într-un registru de intrare (Figura 6-29). În cazul unei implementări secvenţiale, aşa cum este cea de faţă, acest registru este necesar pentru a asigura valori stabile ale vectorului a pe durata prelucrărilor. Cu toate că maşina virtuală de reţea Petri nu poate determina în mod instantaneu nici vectorul de stare şi nici vectorul de concesionare, în limita performanţelor de timp real, implementarea se poate executa în mod corect cu condiţia ca să nu se genereze stări cărora nu le corespund marcaje ale reţelei. Acest lucru poate fi asigurat prin registrul de intrare. Nu se poate însă evita ca maşina virtuală de reţea Petri să rămână în urma procesului condus (decalaj temporal). De aici derivă şi limitările de lucru în timp real.

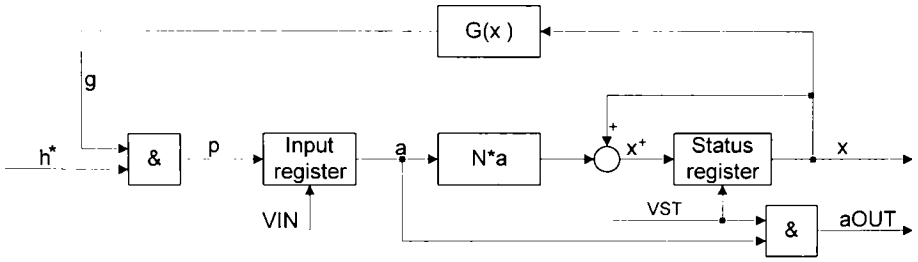


Fig. 6-29 Schema bloc funcţională pentru implementarea maşinii virtuale de reţea Petri

Maşina virtuală de reţea Petri preia informaţiile din proces prin predicatele tranziţiilor prezente în vectorul h^*. Elementele acestui vector sunt variabile binare. Acestea sunt validate cu ajutorul vectorului de concesionare g, printr-o operaţie de conjuncţie logică. Rezultatul acestei operaţii este înscris în registrul de intrare prin semnalul de validare VIN, aplicat în momentul în care toate semnalele din vectorul g sunt valide şi stabile.

Semnalul de validare a stării, VST, este inserat în momentul în care starea nou determinată x^+ este validă şi stabilă.

Maşina virtuală de reţea Petri lucrează ciclic cu perioada $T_C=t_1+t_2$ (Figura 6-30). Prin t_1 s-a notat durata necesară determinării vectorului de concesionare, g, iar prin t_2 durata necesară determinării vectorului de stare, x^+ . Perioada ciclului de prelucrare a maşinii virtuale de reţea Petri este o mărime ce caracterizează capabilităţile de lucru în timp real ale acesteia. Maşina virtuală de reţea Petri poate controla procese în care frecvenţa de apariţie a oricărui eveniment este mai mică decât $1/T_C$.

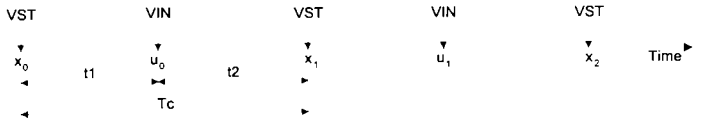


Fig. 6-30 Ciclul de lucru al maşinii virtuale de reţea Petri

Vectorul de stare se determină pe baza relaţiei (6.24). Elementele vectorului x^+, notate prin $x^+[i]$, pot fi exprimate prin relaţia

$$\mathbf{x}^+[i] = \mathbf{x}[i] + \sum_{q=1}^{|T|} \mathbf{N}^+[i,q] \bullet \mathbf{a}[q] - \sum_{q=1}^{|T|} (-\mathbf{N}^-[i,q]) \bullet \mathbf{a}[q] \ .$$

unde prin N[i,q] s-au desemnat elementele matricei N.

Situaţiile de conflict, prezente în modelul de reţea C/E al sistemului, se rezolvă prin strategia de comandă.

La un moment dat, locaţiile dintr-o reţea C/E pot să fie ori valide ori invalide. Din acest motiv modelul de reţea C/E trebuie să fie sigură. O reţea C/E este sigură dacă, pentru orice marcaj realizabil, nici o locaţie nu conţine mai mult de un marcaj. Pentru a asigura acest lucru, nici o tranziţie nu poate fi concesionată dacă are vreo locaţie marcată în postmulţimea ei sau vreo locaţie nemarcată în premulţimea ei. Regula de tranziţie exprimă acest lucru prin relaţia

$$-\mathbf{e}_j^- \le \mathbf{x} \le 1 - (\mathbf{e}_j^+ + \mathbf{e}_j^-)$$

Într-o reţea C/E sunt valabile următoarele relaţii:

i) dacă $x[i] = 1 \Rightarrow \sum_{q=1}^{|T|} \mathbf{N}^+[i,q] \bullet \mathbf{a}[q] = 0$;

ii) dacă $x[i] = 0 \Rightarrow \sum_{q=1}^{|T|} (-\mathbf{N}^-[i,q]) \bullet \mathbf{a}[q] = 0$;

iii) $\sum_{q=1}^{|T|} \mathbf{N}^+[i,q] \bullet \mathbf{a}[q] \in \{0,1\}$, $\sum_{q=1}^{|T|} (-\mathbf{N}^-[i,q]) \bullet \mathbf{a}[q] \in \{0,1\}$.

În cazul unei reţele C/E sigure sumele de mai sus pot fi exprimate prin funcţii logice. Acestea se vor nota prin B^+_F pentru sumele ce conţin elementele matricii N^+ respectiv prin B^-_F pentru sumele ce conţin elementele matricii N^-, şi se exprimă prin relaţiile

$$B^+_{Fi} = (N^+[1,i] \wedge \mathbf{a}[1]) \vee (N^+[2,i] \wedge \mathbf{a}[2]) \vee \ ... \ \vee (N^+[|T|,i] \wedge \mathbf{a}[|T|])$$

$$B^-_{Fi} = (N^-[1,i] \wedge \mathbf{a}[1]) \vee (N^+[1,i] \wedge \mathbf{a}[2]) \vee \ ... \ \vee (N^+[1,i] \wedge \mathbf{a}[|T|])$$

şi prin urmare pentru i=1(1)|T| se obţine:

$$x^+[i] = \begin{cases} x_i & \text{if } (B^+_{Fi}=0) \wedge (B^-_{Fi}=0) \vee (B^+_{Fi}=1) \wedge (B^-_{Fi}=1) \\ 1 & \text{if } B^+_{Fi} = 1 \\ 0 & \text{if } B^-_{Fi} = 1. \end{cases}$$

Elementele vectorului de concesionare se determină pe baza regulei de concesionare prin următoarea relaţie logică:

$$g[j] = ((x[1] \wedge N^+[j,1]) \vee !N^+[j,1])) \wedge ((x[2] \wedge N^+[j,2]) \vee !N^+[j,2])) \wedge ... \wedge ((x[|P|] \wedge N^+[j,|P|]) \vee !N^+[j,|P|])) \wedge ((x[1] \wedge N^-[j,1]) \vee !N^-[j,1])) \wedge ((x[2] \wedge N^-[j,2]) \vee !N^-[j,2])) \wedge ... \wedge ((x[|P|] \wedge N^-[j,|P|]) \vee !N^-[j,|P|]))$$

unde prin "!" s-a notat operaţia de negaţie logică.

Considerentele de mai sus sunt valabile atât pentru o implementare software cât şi pentru o implementare hardware. O altă formă de prezentare a relaţiilor de mai sus este dată în [Tak'96a], [Tak'96b] respectiv [Tak'96c].

Flexibilitatea şi viteza de lucru sunt caracteristicile de bază ale unei implementări. Proiectantul va trebui să opteze pentru o implementare hardware sau software în funcţie

de caacteristicile de timp real ale procesului condus şi în funcţie de complexitate modelului de reţea Petri al acestuia.

Un controler industrial realizat prin implementarea directă a modelului de reţea Petri trebuie să prezinte aceeaşi flexibilitate ca un automat programabil clasic. În cazul unei implementări hardware acest lucru este realizabil doar prin circuite logice programabile în sistem ("in system programmable logic devices"), [Vol'92].

În cazul controlerelor modelate printr-o reţea C/E cu mai puţin de 32 de locaţii şi tranziţii este posibilă o implementare prin circuite logice combinaţionale şi deci întreaga maşină virtuală de reţea Petri poate fi realizată printr-un circuit logic programabil în sistem.

Pentru sisteme mai mari trebuie utilizată logica secvenţială. În Figura 6-31 se prezintă o implementare hardware ce lucrează în mod secvenţial, realizabilă printr-un circuit logic LSI programabil în sistem.

Structura reţelei este programată în memorii de tip EEPROM notate prin EEPROM N+ respectiv EEPROM N-. Presupunând o organizare matriceală, coloanele memoriilor EEPROM N+ respectiv EEPROM N- implementează elementele vectorilor de evenimente e^+_j respectiv e^-_j.

O linie de |P| elemente din EEPROM N+ memorează starea iniţială x_0 adică marcajul iniţial a reţelei. O linie de |P| elemente din EEPROM N- se utilizează, prin interacţiunea cu unitatea de control, pentru a specifica numărul de locaţii din reţeaua ce se implementează. Se poate astfel genera un semnal de "sfârşit de secvenţă".

Prin semnalele INIT şi WR vectorul x_0 al stării iniţiale este transferat în mod serial în bistabilele x'_i. Procesul secvenţial care determină elementele vectorului de concesionare este pornit prin semnalul "Load1". Acest semnal setează toate bistabilele g_j. Acestea rămân setate doar dacă funcţia logică,

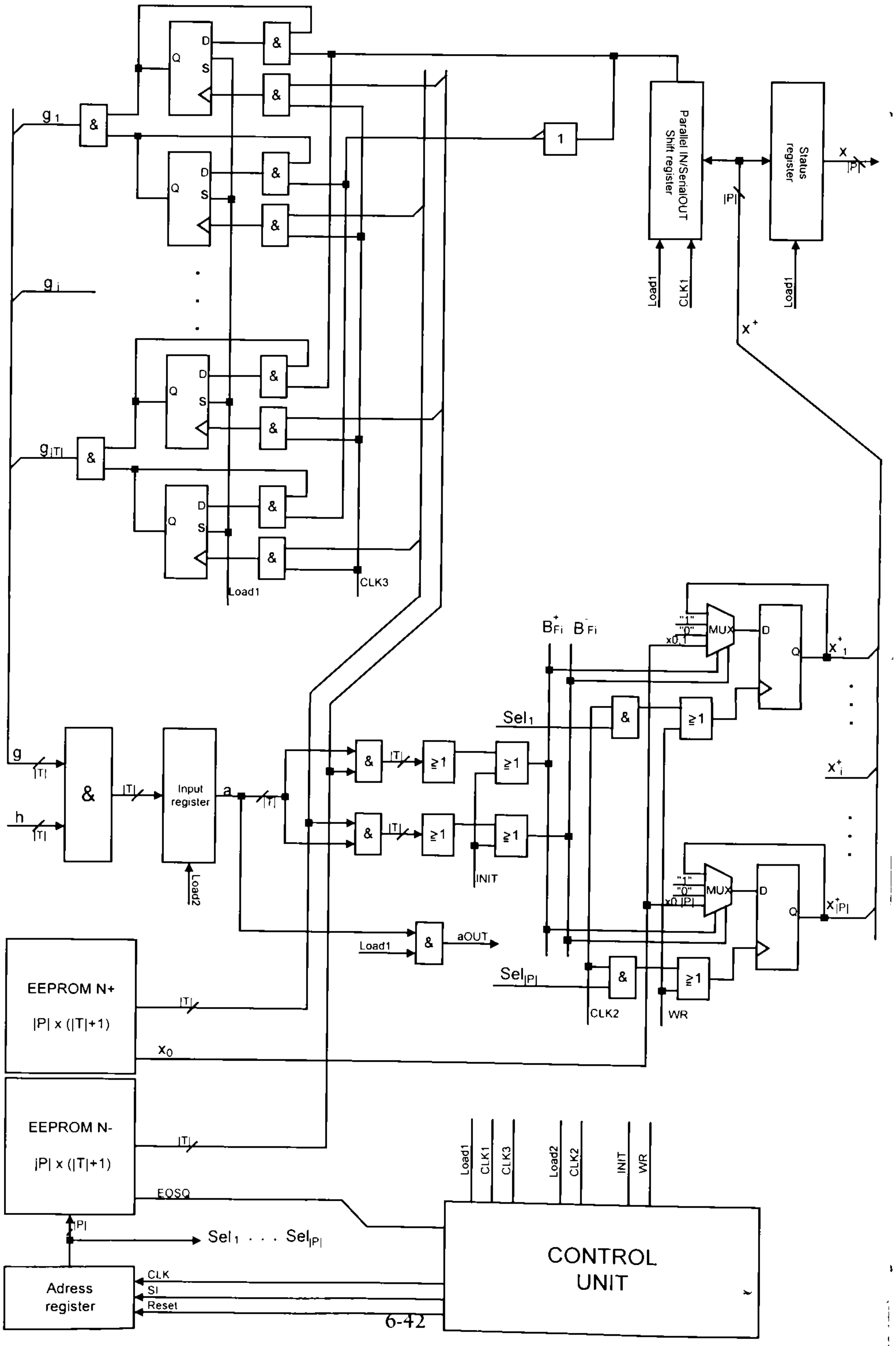
$g[j]=g^+[j]\wedge g^-[j]$, unde

$g^+[j]=((x[1]\wedge N^+[j,1])\vee !N^+[j,1]))\wedge((x[2]\wedge N^+[j,2])\vee !N^+[j,2]))\wedge...\wedge((x[|P|]\wedge N^+[j,|P|])\vee !N^+[j,|P|]))$

$g^-[j]=((x[1]\wedge N^-[j,1])\vee !N^-[j,1]))\wedge((x[2]\wedge N^-[j,2])\vee !N^-[j,2]))\wedge...\wedge((x[|P|]\wedge N^-[j,|P|])\vee !N^-[j,|P|])),$

ia valoarea "adevărat". La fiecare tact al ceasului CLK1 se testează câte un termen al relaţiilor de mai sus.

Cu ajutorul semnalului "Load2" rezultatul conjuncţiei logice dintre elementele vectorului de concesionare şi ale vectorului de tranziţie este trecut în registrul de intrare. Într-un proces secvenţial de |P| paşi sunt determinate funcţiile logice $x^+[i]$ şi înscrise în bistabilele x^+_i prin circuite de multiplexare cu ajutorul semnalelor "Sel_i" . Vectorul de stare este transferat în registrul de stare prin semnalul "Load1". Prin acelaş semnal, elementele vectorului a sunt prezentate spre exterior, sub forma unor semnale Mealy, pentru eventuale sincronizări.



g 1
g i
g |T|
g
h
|T|
Input register
Load2
a
Load1
CLK3
1
Parallel IN/Serial OUT Shift register
Status register
Load1
CLK1
|P|
x+
x |P|
Sel1
Sel|P|
INIT
aOUT
CLK2
WR
MUX
x+ 1
x+ i
x+ |P|
EEPROM N+
|P| x (|T|+1)
EEPROM N-
|P| x (|T|+1)
x0
EOSQ
Sel1 . . . Sel|P|
Adress register
CLK
SI
Reset
CONTROL UNIT
Load1
CLK1
CLK3
Load2
CLK2
INIT
WR

Fig. 6-31 Exemplu de implementare harware a unei maşini virtuale de reţea Petri

Considerând cazul |P|=|T|=64 se obţine:

- EEPROM N+=EEPROM N- = 65x65 = 4225 biţi;
- numărul de bistabile necesare: aprox.500;
- numărul porţilor logice necesare: aprox.1000;
- numărul terminalelor necesare: aprox.200.

Un circuit integrat de această complexitate este executabil cu tehnologiile actuale.

6.7 Caracteristicile sistemelor de conducere realizate pe baza maşinii virtuale de reţea Petri

6.7.1 Caracteristici de lucru în timp real.

SC lucrează în timp real dacă poate reacţiona la apariţiile succesive ale oricărui eveniment. Frecvenţa maximă de apariţie a evenimentelor pe care SC o poate trata este limitată de întârzierile datorate transmiterii semnalelor prin canalele de comunicaţie şi a întârzierilor datorate prelucrării acestora.

În cazul maşinii virtuale de reţea Petri întârzierea se datorează *timpului de insensibilitate* al intrărilor pentru mărimile de execuţie. În Figura 6-32 timpul de insensibilitate este notat cu τ_{in} şi se compune din timpul τ_1 necesar determinării vectorului de stare şi din timpul τ_2 necesar determinării vectorului de concesionare. Astfel, timpul de insensibiliate se determină prin $\tau_{in}=\tau_1+\tau_2$. Mărimea $F_{in}=1/\tau_{in}$ este o mărime de calitate a maşinii virtuale de reţea Petri. Valoarea acestuia depinde de caracteristicile echipamentului prin care se realizează şi trebuie determinată pentru cazul cel mai defavorabil în care se determină toate elementele vectorilor x şi g.

O altă mărime de calitate a SC este timpul necesar generării mărimilor de comandă **u**. Figura 6-32 prezintă duratele necesare în diversele faze de generare şi transmitere a mărimilor de comandă. Aceasta este dată de $\tau=\tau_e+\tau_{in}+\tau_u+\tau_c$, unde: τ_e este timpul necesar determinării şi transmiterii mărimii de execuţie, τ_{in} este timpul necesar determinării stării procesului, τ_u este timpul necesar generării mărimii de comandă, τ_c este timpul necesar transmiterii semnalului de comandă.

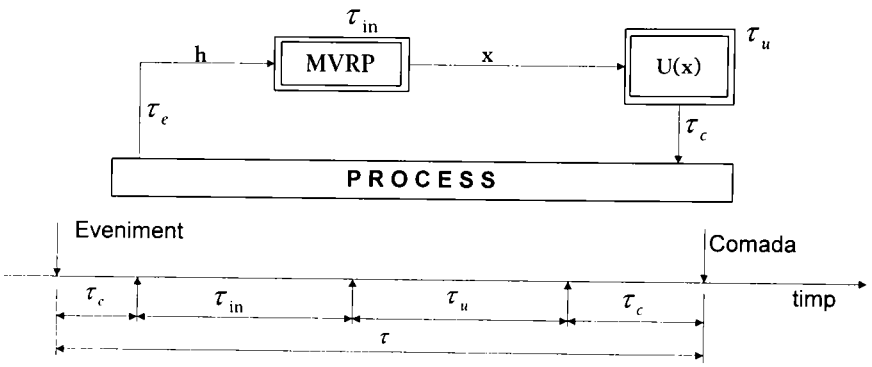


Fig. 6-32 Timpul de generare a mărimilor de comandă şi componentele sale

SC poate comanda doar evenimente ce apar cu o întârziere mai mare decât τ faţă de evenimentul pe baza căreia se elaborează comanda. Această mărime depinde atât de echipamentul care-l realizează cât şi de strategia de comandă adoptată.

6.7.2 Modularizare. Distribuire

Funcţionalitatea SC şi OPT definită mai sus permite distribuirea totală a modulelor funcţionale care realizeză modelul de reţea Petri. Interacţiunea dintre OPT şi dintre OPT şi SC, se realizeză prin canale de comunicaţie. Acestora li s-a impus doar livrarea sigură a informaţiilor, nu s-a impus ca livrarea să se realizeze în ordinea cauzală. Capacitatea canalelor de comunicaţie poate influenţa în mod considerabil performanţele sistemului de conducere. Proiectantul este ajutat prin faptul că standardele de comunicaţie deterministe garantează timpul maxim de aşteptare pentru accesul la serviciile de comunicaţie.

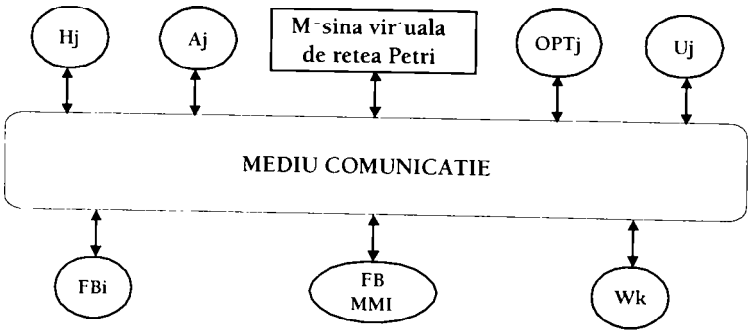


Fig. 6-33 Comunicaţia dintre elementele funcţionale ale sistemului de conducere

În sistemul de conducere s-au definit următoarele module funcţionale (Figura 6-33):

(i) Maşina virtulă de reţea Petri;

(ii) Funcţiuni de bază, abreviat FB, cu ajutorul cărora OPT pot executa acţiunile corespunzătoare tranziţiilor din modelul de reţea Petri;

(iii) Operatori de tranziţie, abreviat OPT, care pe baza interacţiunilor cu sistemul de conducere şi cu ajutorul funcţiunilor de bază realizează tranziţiile în procesul condus;

(iv) Observatori de evenimente care realizează predicatele tranziţiei respectiv predicatele activării. Fiecare dintre aceşti observatori poate fi realizat ca modul funcţional distinct (H_j respectiv A_j);

(v) Strategia de comandă, în principiu, poate fi realizată din module funcţionale distincte (U_j) care determină fiecare mărime de comandă (u_j) în parte. În cazul în care se impun caacteristici de timp real deosebite, aceste module funcţionale vor fi implementate de acelaşi echipament care realizează si maşina virtuală de reţea Petri.

(vi) Module funcţionale pentru comunicaţia cu un sistem MMI.

Modulele funcţionale care furnizează mărimi de conducere, notate prin w, sunt externe sistemului de conducere dar trebuie să fie accesibile prin sistemul de comunicaţie.

Interacţiunile necesare realizării tranziţiilor sunt "ascunse" observatorului de la nivelul de abstractizare al reţelei Petri. OPT este elementul de bază al sistemului distribuit şi astfel determină granularitatea distribuţiei. În situaţii practice se poate întâmpla ca mai multe OPT să fie realizate de acelaşi echipament de calcul. Se poate întâmpla ca unele OPT să fie realizate chiar de către SC. Deşi în practică este de evitat o astfel de soluţie, interpretarea dată mai sus reţelelor Petri permite acest lucru.

6.7.3 Scalabilitate

Modularizarea prezentată mai sus este semnificativă şi în privinţa scalabilităţii în sensul că, în funcţie de cerinţele concrete diversele module pot fi implementate pe echipamente distincte sau mai multe module pe acelaş echipament. Interesează mai mult scalabilitatea maşinii virtuale de reţea Petri deoarece performanţele de timp real sunt determinate în mod direct de viteza de lucru a acestuia.

O primă modalitate de a implementa maşina virtuală de reţea Petri este acea de a trata procesele de prelucrare ca fiind strict secvenţiale aşa cum se prezintă în Figura 5-34 .

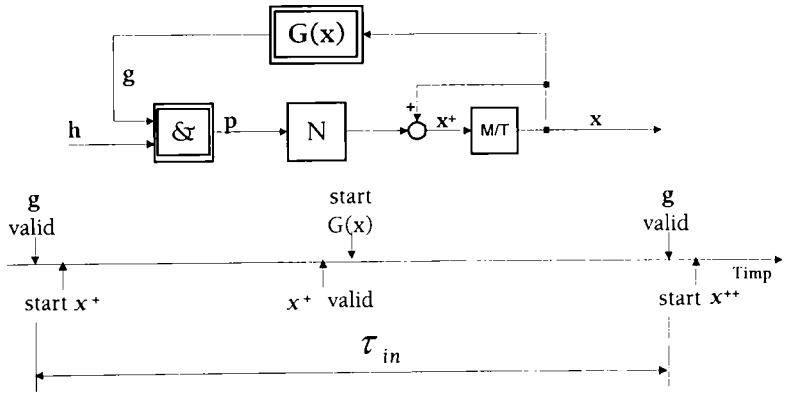


Fig. 6-34 Funcţionarea strict secvenţială a maşinii virtuale de reţea Petri

Intervalul τ_{in} este durata de insensibilitate a maşinii virtuale de reţea Petri faţă de apariţiile de evenimente din procesul condus şi caracterizează în mod direct performanţele de timp real ale acesteia. Echipamentul care realizează maşina virtuală de reţea Petri trebuie să minimizeze intervalul τ_{in}.

Caracteristica acestei metode este stricta secvenţialitate în privinţa determinării vectorului de stare x şi a vectorului de concesionare g. Într-adevăr vectorul de concesionare poate fi determinat doar dacă se dispune de un vector de stare valid. În schimb există posibilitatea de a efectua prelucrările necesare determinării vectorului de stare x' în paralel cu determinarea vectorului de concesionare.

În acest sens observăm faptul că, în ipoteza unei reţele fără bucle autonome, relaţia $\mathbf{x}^+ = \mathbf{x} + \mathbf{N} \cdot \mathbf{p}$ poate fi scrisă succesiv:

$$\mathbf{x}^+ = \mathbf{x} + \sum_{\mathbf{p[j]=1}} \mathbf{t}_j \qquad (6.24)$$

$$\mathbf{x}^+ = \mathbf{x} + \sum_{\mathbf{j=1}}^{|\mathbf{T}|} \mathbf{g[j]} \cdot \mathbf{h}^*\mathbf{[j]} \cdot \mathbf{t}_j \qquad (6.25)$$

Înmulţirile din termenul doi al membrului doi din ecuţia de mai sus conduc la înmulţirea unui vector cu un scalar în sens uzual iar adunările sunt adunări vectoriale. Introducând notaţia $\mathbf{b}_i = \sum_{j=1}^{|i|} \mathbf{g[j]} \cdot \mathbf{h}^*\mathbf{[j]} \cdot \mathbf{t}_j$ relaţia de mai sus devine

$$\mathbf{x}^+ = \mathbf{x} + \mathbf{b}_{|\mathbf{T}|} \text{ , cu } 0 \le i \le |T| \text{ şi } \mathbf{b}_0 = 0. \qquad (6.26)$$

Deasemnea se poate observa faptul că $\mathbf{b}_j$ poate fi calculat în mod recursiv astfel:

$$\mathbf{b}_j = \mathbf{b}_{j-1} + \mathbf{g[j]} \cdot \mathbf{h}^*\mathbf{[j]} \cdot \mathbf{t}_j \qquad (6.27)$$

Relaţiile de mai sus arată că termenul $\mathbf{b}_j$ poate fi procesat în paralel cu determinarea elementelor g[j] din vectorul de concesionare, aşa cum se prezintă în Figura 6-35.

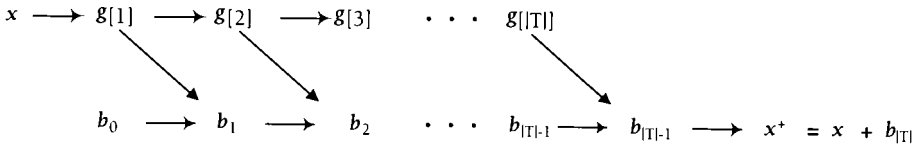


Fig. 6-35 Procesarea paralelă a elementelor g[j] şi b_i

Schema bloc structurală a modulului funcţional pentru determinarea vectorului de stare conform procedeului de mai sus se prezintă în Figura 6-36

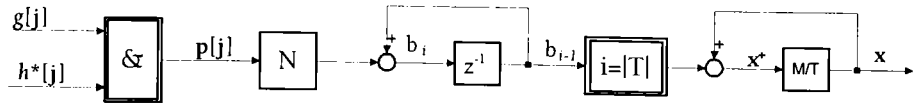


Fig. 6-36 Schema bloc structurală pentru determinarea vectorului de stare

Fiecare pas de prelucrare se iniţiază în momentul în care s-a recepţionat elementul g[j] validat al vectorului de concesionare. Modulul notat i=|T| transferă valoarea $\mathbf{b}_{|T|}$ pentru calculul x'. Elementul g[j] al vectorului de concesionare se obţine prin relaţia

$$g[j]=F_{1j} \wedge F_{2j}. \qquad (6\text{-}28)$$

Funcţiile F_{1j} şi F_{2j} pot fi implementate prin module funcţionale care lucrează concurent. Elementul g[j] va fi furnizat împreună cu un semnal de validare.

Se poate observa că toate cele trei module funcţionale definite mai sus efectuează adunarea a doi vectori de |T| elemente şi prin urmare sunt foarte echilibrat încărcate. Deasemnea se poate observa că suprastructura necesară asigurării funcţionării în paralel a celor trei module este minimală. Din acest motiv funcţionarea lor concurentă este deosebit de eficientă asigurând efectiv creşterea de trei ori a vitezei de prelucrare faţă de cazul secvenţial, desigur cu preţul a trei echipamente de calcul.

6.8 Concluzii

Încapsularea proceselor de alocare a resurselor

Analiza asupra efectelor tranziţiilor temporizate în modelarea cu reţele Petri a condus la următorul principiu de modelare:

> **Modelul de reţea Petri al sistemelor tehnice trebuie astfel elaborat, încât procesele de arbitrare a resurselor partajate să fie "încapsulate" adică separate de procesele de prelucrare informaţională sau materială.**

Maşina virtuală de reţea Petri

Dacă sistemul de conducere se realizează pe baza maşinii virtuale de reţea Petri atunci acesta poate elabora comenzi în concordanţă cu structura cauzală a procesului condus.

> **Folosind o implementare a maşinii virtuale de reţea Petri, structurată funcţional ca în Figura 6-14, un sistem de conducere poate genera mărimi de comandă în concordanţă cu relaţiile cauzale din procesele conduse descrise prin modele de reţele Petri.**

Structurare

Modelul de reţea Petri, împreună cu intrerpretarea prezentată, realizează o structurare clară a sistemului de conducere. Pe baza acestui model, funcţiunile fiecărei componente pot fi univoc delimitate şi eficient încapsulate.

Interfaţare

Interfaţa dinte modulele funcţionale, definită în acest capitol, este cea mai simplă posibilă. Ea este în acelaş timp şi uniformă în sensul că toate mărimile sunt interfaţate în acelaş mod. Specificarea acestei interfeţe este imediată atât în cazul comunicaţiei prin semnale cât şi în cazul comunicţiei prin mesaje. Din acest motiv se poate realiza şi o standardizare eficientă şi uşor acceptabilă.

Sistem deschis

Încapsularea clară şi interfaţarea simplă permit crearea de sisteme efectiv deschise. Fiecare modul funcţional poate fi realizat prin echipamente distincte de la diverşi producători.

* * *

Faţă de literatura de specialitate la care autorul a avut acces, acest capitol prezintă următoarele elemente originale:

a) Definirea, pe baza unei analize asupra efectului tranziţiilor temporizate în modelarea cu reţele Petri, a principiului încapsulării proceselor de arbitrare a resurselor partajate.

b) Definirea unei metode de interpretare a reţelelor Petri şi definirea obiectelor semantice din procesul modelat, pe baza cărora se realizează interpretarea.

c) Definirea noţiunii de ***operator de tranziţie*** ca element de bază în interpretarea reţelelor Petri şi în implementarea sistemelor de automatizare modelate prin reţele Petri.

d) Definirea mărimilor de interacţiune dintre sistemul de conducere, implementat pe baza modelului de reţea Petri, şi procesul condus respectiv mediu.

e) Elaborarea metodelor de determinare a stării procesului condus modelat prin reţele Petri.

f) Elaborarea metodei de refacere a ordinii cauzale a mărimilor de reacţie şi definirea conceptului de "*maşină virtuală de reţea Petri*".

g) Elaborarea structurii sistemului de conducere bazat pe conceptul de maşină virtuală de reţea Petri.

h) Studiul asupra variantelor acestei structuri de conducere.

i) Studiul asupra aspectelor de implementare a sistemului de conducere. Elaborarea algoritmului pentru determinarea *vectorului de stare* şi a *vectorului de concesionare*.

j) Prezentarea unui exemplu de implementare harware a maşinii virtuale de reţea Petri.

k) Studiu asupra caracteristicilor sistemului de conducere implementat pe baza maşinii virtuale de reţea Petri

7 Sinteza strategiei de conducere pe baza modelului de reţea Petri

În acest capitol se defineşte o structurare a vectorului de comandă şi se prezintă două metode de sinteză a strategiei de conducere pentru cazul în care procesul este condus printr-o reacţie după stare. Sinteza se realizează pe baza unor restricţii asupra stărilor realizabile ale procesului condus respectiv asupra marcajelor accesibile ale modelului de reţea Petri al procesului condus.

7.1 Preliminarii

Din punctul de vedere al proiectării unui sistem de automatizare pe baza modelului de reţea Petri, soluţia cea mai simplă şi cea mai elegantă ar fi aceea de a modela în mod direct comportamentul dorit al sistemului (sistem de conducere+proces condus), deoarece în acest caz modelul ar încorpora şi strategia de conducere iar vectorul de comandă ar fi identic cu vectorul de concesionare: **u =g** în Figura 7.1.

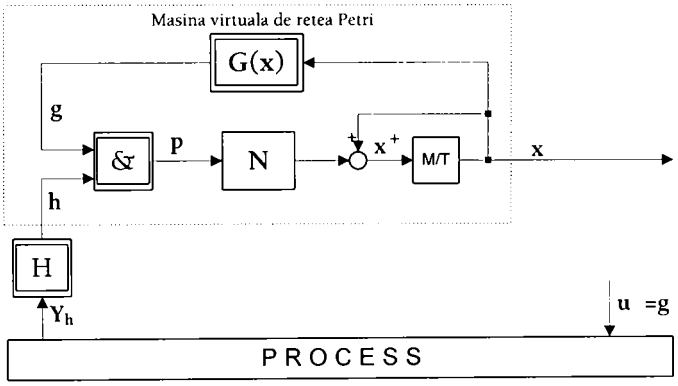


Fig. 7-1 Structura de conducere pentru cazul modelării comportamentului dorit al sistemului de automatizare

Această metodă are însă următoarele dezavantaje majore:

(i) Aplicând această metodă dispare delimitarea clară dintre structura cauzală proprie procesului condus şi strategia de conducere prin care se realizează conducerea procesului.

(ii) În implementarea sistemului de automatizare nu se poate face o delimitare clară între sistemul de comandă şi utilajele tehnologice, în sensul că, pentru a implementa modelul de reţea Petri, nu se poate garanta că nu trebuie intervenit şi la nivel de utilaj tehnologic cu modificări de structură şi/sau de funcţionalitate.

În cele ce urmează se va presupune existenţa unui model de reţea Petri al procesului condus având destule grade de libertate astfel încât comportmentul dorit al procesului să fie realizabil prin restricţii asupra marcajelor accesibile.

Procesul condus respectiv modelul de reţea Petri al acestuia trebuie să ofere aceste grade de libertate pentru:

(i) a asigura flexibilitate sistemului de automatizare;

(ii) a oferi posibilitatea corecţiei comportamentului dinamic al sistemului cu scopul de a asigura viabilitatea şi/sau reversibilitatea sistemului.

7.2 Componentele vectorului de comandă

Vectorul de comandă, **u** , se determină pe baza strategiei de conducere, notată prin **U(x,w)** in Figura 7.2. Se va lucra cu un vector de comandă având |T| elemente ordonate în conformitate cu vectorul **t** al tranziţiilor. Tranziţiile modelează acţiuni atomice, din acest motiv elementele vectorului de comandă au valori binare: $\mathbf{u}[j] \in \{0,1\}$.

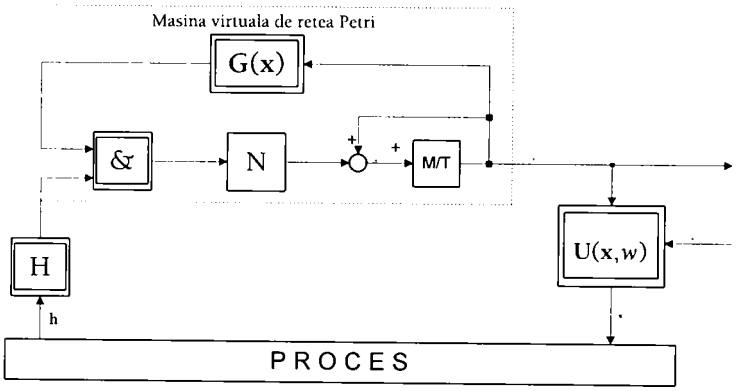


Fig. 7-2 Structura de conducere cu strategia de conducere implementată printr-un modul distinct

Elementele vectorului de comandă se determină cu ajutorul unor funcţii specifice fiecărui element al vectorului de comandă. Aceste funcţii se specifică prin strategia de conducere. O analiză atentă arată că, în cazul în are procesul este condus pe baza unui model de reţea Petri, strategia de conducere înglobează următoarele componente:

a) Componenta de controlabilitate specifică tranziţiile controlabile. Deoarece nu toate tranziţiile sunt controlabile, elementele corespunzătoare tranziţiilor necontrolabile vor avea valoarea "0-logic" (fals). Acest lucru se specifică prin componenta de controlabilitate, $\mathbf{u}_C$, a vectorului de comandă astfel:

$$\mathbf{u}_C[j] = \begin{cases} 1 \text{ daca } t_j \in T_C \\ 0 \text{ daca } t_j \notin T_C \end{cases}, \quad 0<j\leq|T|. \tag{7.1}$$

unde T_C este mulţimea tranziţiilor controlabile. Desigur, vectorul $\mathbf{u}_C$ specifică o caracteristică a procesului respectiv al modelului de reţea Petri al acestuia. Cu toate acestea, deoarece nici o strategia de conducere nu poate face abstracţie de această caracteristică, este justificată înglobarea acestui vector în strategia de conducere.

b) Componenta de sincronizare, $\mathbf{u}_S$, este o funcţie definită pe mulţimea mărimilor de strare, **x** , şi a mărimilor de conducere, **w**.

$\mathbf{u}_S[j]=S_j(\mathbf{x}_k,\mathbf{w}):(P\times N_P)\times(W\times N_W)\rightarrow\{0,1\}$, $0<j\leq|T|$. (7.2)

unde N_P este o submulţime a mulţimii numerelor naturale reprezentând domeniul de definiţie al funcţiei de marcare. S-a presupus, pentru omogenitatea reprezentării, că şi mărimile de conducere se specifică prin numere naturale din submulţimea N_W. Această prezumpţie este justificată deoare $\mathbf{u}_S[j]$ fiind o funcţie logică, valorile mărimilor de conducere trebuie oricum discretizate. Dacă această discretizare este necesară ea se executa în afara modelului prezentat.

Prin această componentă se realizează sincronizarea execuţiei tranziţiilor cu anumite stări ale procesului respectiv cu anumite valori ale vectorului de conducere. Situaţiile conflictuale se rezolvă întotdeauna prin componenta de sincronizare a mărimilor de comandă.

c) Componenta de temporizare este prezentă din cauza comenzilor de temporizare. Comenzile de temporizare sunt comenzi care întârzie execuţia unei tranziţii faţă de concesionarea realizată prin sintaxa şi structura reţelei. Întârzierea se realizează în însăşi utilajul tehnologic sau prin elemente de temporizare realizate de strategia de conducere, conform Figurii 7-3.

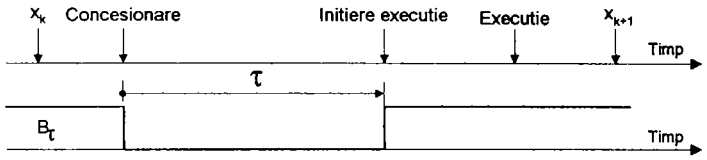


Fig. 7-3 Relativă la execuţia temporizată a tranziţiilor

În Figura 7-3 prin τ s-a notat durata temporizării realizate din momentul concesionării iar prin B_τ variabila logică definită astfel:

$$B_\tau = \begin{cases} 0 & \text{- in cursul (pe durata) temporizarii, respectiv} \\ 1 & \text{- daca temporizarea nu este activata} \end{cases}$$

O astfel de variabilă logică se defineşte pentru fiecare tranziţie controlabilă în parte, variabilele logice corespunzătoare fiind notate prin: $B_{\tau j}$. În cazul tranziţiilor netemporizate $B_{\tau j}=1$. Componenta de temporizare a vectorului de comandă se specifică astfel:

$\mathbf{u}_T[j]=\mathbf{g}_k[j]\cdot B_{\tau j}$, $0<j\leq|T|$. (7.3)

Elementele de temporizare, pe cât posibil, trebuie evitate. În locul lor trebuie utilizate mărimi măsurate în procesul condus, controlul acestuia realizându-se prin reacţia după stare (sincronizarea) pe baza acestor mărimi. Figura 7-4 prezintă schimbarea de structură necesară dacă în locul temporizării se recurge la reacţia după stare.

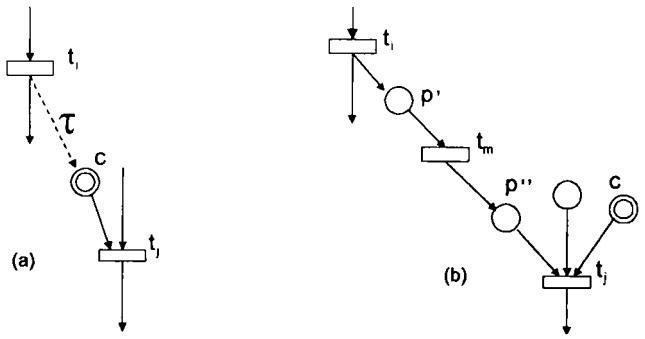


Fig. 7-4 Înlocuirea temporizării printr-o mărime măsurată în procesul condus

În Figura 7-4(a) comanda tranziţiei t_j se validează după τ unităţi de timp de la producerea evenimentului t_i. În Figura 7-4(b) tranziţia t_j se va concesiona doar după producerea evenimentului t_m, a cărui apariţie este condiţionată de producerea evenimemtului t_j. Pe baza predicatului tranziţiei t_m, maşina virtuală de reţea Petri va determina marcajul (starea) corespunzătoare locaţiei p" şi poate elabora o comandă de sincronizare.

d) Componenta anticipativă este determinată de restricţiile impuse asupra stărilor realizabile ale procesului condus. Denumirea este justificată de faptul că restricţia poate fi respectată doar dacă se anticipă efectul comenzii asupra comportării sistemului şi se permite doar aplicarea acelor comenzi care conduc sistemul în stări care nu violează restricţiile impuse. Componenta anticipativă a vectorului de comandă se specifică prin următoarele funcţii:

$$\mathbf{u}_A[j]=A_j(\mathbf{x}_k,\mathbf{w},\aleph,N):(P\times N_P)\times(W\times N_P)\times(P\times N_P)\times(P\times T)\rightarrow\{0,1\},\ 0<j\leq|T|. \quad (7.4)$$

unde: $\aleph$ este mulţimea restricţiilor ce trebuie respectate iar N este matricea de incidenţe a reţelei. Aceasta din urmă apare ca argument al funcţiei deoarece un studiu anticipativ se poate efectua doar pe baza structurii modelului de reţea Petri al procesului condus.

Având în vedere cele de mai sus elementele componentelor vectorului de comandă se determină astfel:

$$\mathbf{u}[j]=\mathbf{u}_C[j]\wedge\mathbf{u}_S[j]\wedge\mathbf{u}_T[j]\wedge\mathbf{u}_A[j]. \quad (7.5)$$

Într-o formă vectorială relaţia de mai sus devine:

$$\mathbf{u}=\mathbf{u}_C\wedge\mathbf{u}_S\wedge\mathbf{u}_T\wedge\mathbf{u}_A. \quad (7.5bis)$$

Înacest ultim caz operaţia de conjuncţie logică se referă la elementele corespeondente ale vectorilor de aceeaşi dimensiune (în cazul de faţă |T|).

Fiecare componentă a vectorului de comandă presupune metode proprii de specificare, analiză şi implementare. În cele ce urmează se va prezenta câte o metodă pentru determinarea componentei anticipative respectiv a componentei de sincronizare.

7.3 O metodă generală pentru determinarea componentei anticipative a vectorului de comandă

Metoda ce se prezintă se bazează pe conceptul de maşină virtuală de reţea Petri introdusă în capitolul 6.

7.3.1 *Restricții*

Componentele anticipative ale vectorului de comandă se determină pe baza restricțiilor asupra stărilor procesului condus respectiv asupra marcajelor accesibile ale modelului de rețea Petri. Marcajele interzise se definesc ca submulțimi ale mulțimii, $R_N(\mathbf{m}_0)$, a marcajelor accesibile. Vom numi aceste submulțimi "restricții de marcaj", "restricții de stare" sau pe scurt "restricții".

Restricțiile vor fi notate prin X. Fiecare restricție $X \subseteq R_N(\mathbf{m}_0)$ trebuie definită printr-o metodă oarecare, spre exemplu așa cum s-a prezentat în capitolul 4. Mulțimea locațiilor condiționate prin restricția X se va nota prin P_X iar mulțimea tuturor restricțiilor prin $\aleph$, $X \subseteq \aleph$, $\aleph = \{X_1, X_2, \ldots\}$.

Strategia de conducere trebuie să asigure ca nici un marcaj al restricției să nu se realizeze, deci :

$$X \notin R_N(\mathbf{m},\mathbf{u}),\ \forall \mathbf{m} \in R_N(\mathbf{m}_0) \wedge \forall \mathbf{u} \in U \qquad (7.6)$$

unde prin $R_N(\mathbf{m},\mathbf{u})$, s-a notat mulțimea marcajelor accesibile din marcajul **m** la comanda **u**.

În cazul în care toate tranzițiile sunt controlabile, realizarea fiecărei marcări este controlabilă și asigurarea condiției de mai sus este realizabilă. Mulțimea tranzițiilor controlabile se va nota prin T_C.

Vom presupune că toate tranzițiile conflictuale sunt controlabile. Această prezumpție se justifică prin cerința elementară ca un sistem tehnic controlat să prezinte un comportament determinist. Celelalte tranziții pot fi controlabile sau nu.

Pentru a asigura realizarea restricției X, strategia de conducere trebuie să anticipeze efectul fiecărei comenzi vis-a-vis de restricția X, respectiv vis-a-vis de marcajele locațiilor $P_X \subseteq P$, condiționate prin restricția dată. Pentru aceasta trebuie determinate toate căile rețelei Petri prin care, plecând din starea inițială dată, pot fi influențate marcajele locațiilor $p \in P_X$.

7.3.2 *Cale de precedență. Domeniu de precedență*

O cale este o secvență orientată formată din locațiile și tranzițiile rețelei Petri. Definim mulțimea Π_N a tuturor căilor aciclice formate din locații și tranziții în alternanță, astfel încât pentru orice cale $\pi \in \Pi_N$, și pentru fiecare segment de cale *tp* din π, $p \in t^\bullet$, respectiv pentru fiecare segment de cale *pt* din π, $p \in {}^\bullet t$. Prin aciclic înțelegem faptul că nici o locație sau tranziție nu se regăsește de mai multe ori într-o cale. În schimb, un ciclu rupt poate să constituie o cale. În cazul operațiilor cu mulțimi o cale va fi considerată ca fiind o submulțime a mulțimii $P \cup T$.

În principiu, căile unei rețele Petri pot fi orientate în sensul arcelor rețelei sau în sensul opus acestora.

În continuare se va considera orientarea căilor ca fiind opusă orientării arcelor rețelei. Acestă convenție este avantajoasă în determinarea căilor de precedență ale unei locații deoarece în acest fel fiecare cale de precedență începe chiar cu locația dată. O cale de precedență a unei locații se definește astfel:

Definiția 7-1 (Cale de precedență)

Se numește ***cale de precedență*** a unei locații $p \in P$ orice cale de forma $\pi(p) = (p_1, t_1, \ldots, p_{n-1}, t_{n-1}, p_n)$ sau de forma $\pi(p) = (p_1, t_1, \ldots, p_{n-1}, t_n)$, orientată în sensul contrar arcelor rețelei și care satisface următoarele condiții:

1. calea începe chiar cu locaţia dată, adică $p_1 \equiv p$;
2. calea se termină la locaţia p_n dacă este valabilă afirmaţia:

$$\left({}^{\bullet}p_n \equiv \varnothing\right) \vee \left(\pi \cap {}^{\bullet}p_n \neq \varnothing\right)$$

3. calea se termină la tranziţia t_n dacă este valabilă afirmaţia:

$$\left(t_n \in T_c\right) \vee \left({}^{\bullet}t_n \equiv \varnothing\right) \vee \left(\pi \cap {}^{\bullet}t_n \neq \varnothing\right)$$

Condiţia 2. din definiţia de mai sus specifică faptul că locaţia terminală a căii de precedenţă este fie o locaţie ce nu are nici o tranziţie în premulţimea ei, Figura 7-5(a), fie o locaţie ce are tranziţii în premulţimea ei ce sunt elemente ale căii însăşi, Figura 7-5(b).

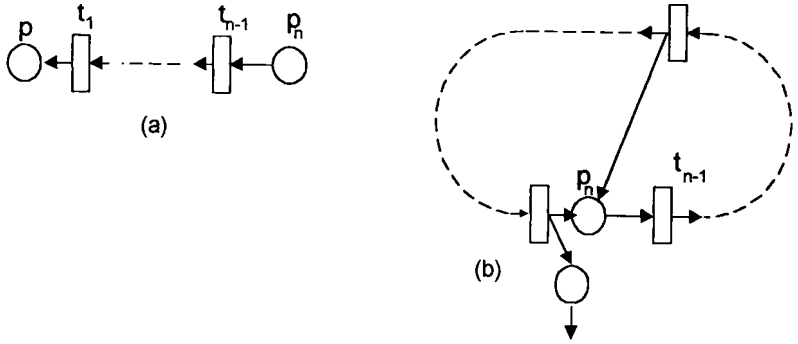


Fig. 7-5 Căi de precedenţă terminate prin locaţii

Condiţia 3. din definiţia de mai sus specifică faptul că dacă calea de precedenţă se termină într-o tranziţie aceasta poate să fie o tranziţie controlată, Figura 7-6(a), o tranziţie având premulţimea vidă (tranziţie tip sursă), Figura 7-6(b), sau o tranziţie ce are în premulţimea ei locaţii ce aparţin căii însăşi, Figura 7-6(c). Figura7-6(d) prezintă situaţia în care bucla se închide chiar pe locaţia dată.

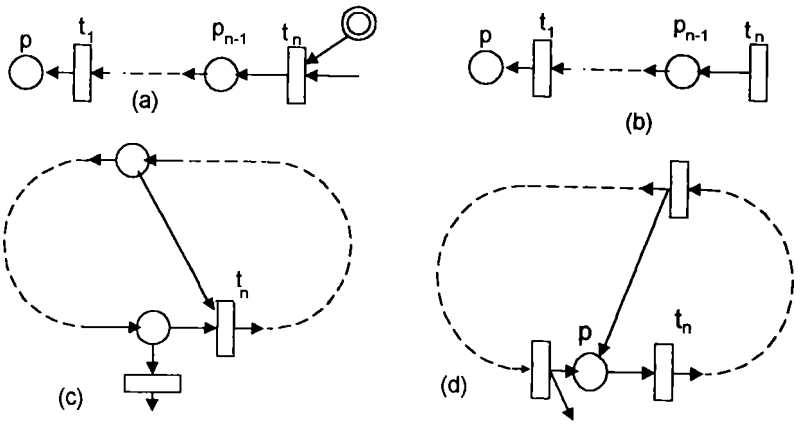


Fig. 7-6 Căi de precedenţă terminate prin tranziţii

Următoarea lemă rezultă imediat din Definiția 1.

Lema 1: *Pentru orice mulțime de locații* P_X *și orice* $p \in P_X$, *mulțimea șirurilor de precedență* $\Pi(p)$ *este finită.*

În acest fel este posibilă, determinarea mulțimii căilor de precedență, pentru fiecare locație $p \in P_X$.

Definiția 7-2 (Domeniu de precedență)

Mulțimea căilor de precedență ale unei locații reunită cu premulțimea tranzițiilor terminale t_n se va numi ***domeniul de precedență*** al locației și se exprimă prin relația:

$$\Pi(p) = \left(\bigcup_i \pi_i(p)\right) \bigcup_i {}^{\bullet} t_{\pi i} \qquad (7.7)$$

unde $t_{\pi i}$ este ultima tranziție a căii π.

Domeniul de precedență al restricției X se definește ca reuniunea domeniilor de precedență ale locațiilor condiționate prin restricția X și se exprimă prin relația:

$$D_X = \bigcup_{p \in P_X} \Pi(p). \qquad (7.8)$$

O cale de precedență este controlabilă dacă ultima tranziție a ei este controlabilă. În continuare se va presupune că toate căile de precedență ale locației $p \in P_X$ sunt controlabile, adică, dacă t_π este ultima tranziție a căii de precedență atunci $\forall t_\pi \in T$: $t_\pi \in T_C$. Domeniul de precedență al unei restricții X este controlabil dacă toate căile de precedență ale acestuia sunt controlabile. Controlabilitatea se referă, de fapt, la controlul accesului marcajelor în domeniul de precedență și prin acesta implicit asupra realizabilității restricției date.

Domeniul de precedență al unei restricții controlabile nu conține cicluri necontrolate. Acest lucru este deosebit de important deoarece marcajele ciclurilor necontrolate nu pot fi influențate prin comenzi. Ele pot realiza toate marcajele permise prin structura și sintaxa rețelei. Prin urmare, locații aparținând ciclurilor necontrolate nu pot fi elemente din domeniul de precedență al unor restricții iar dacă ele apar atunci proiectantul va trebui să asigure controlabilitatea prin măsuri constructive suplimentare. Din punctul de vedere al căilor de precedență, un ciclu controlabil (având cel puțin o tranziție controlabilă) este "rupt" la prima tranziție controlabilă. Prin urmare este adevărată următoarea lemă:

Lema 2: *Domeniul de precedență nu conține cicluri.*

7.3.3 Determinarea domeniului de precedență a unei restricții

Figura 7-7 prezintă organigrama pentru determinarea domeniului de precedență a unei restricții.

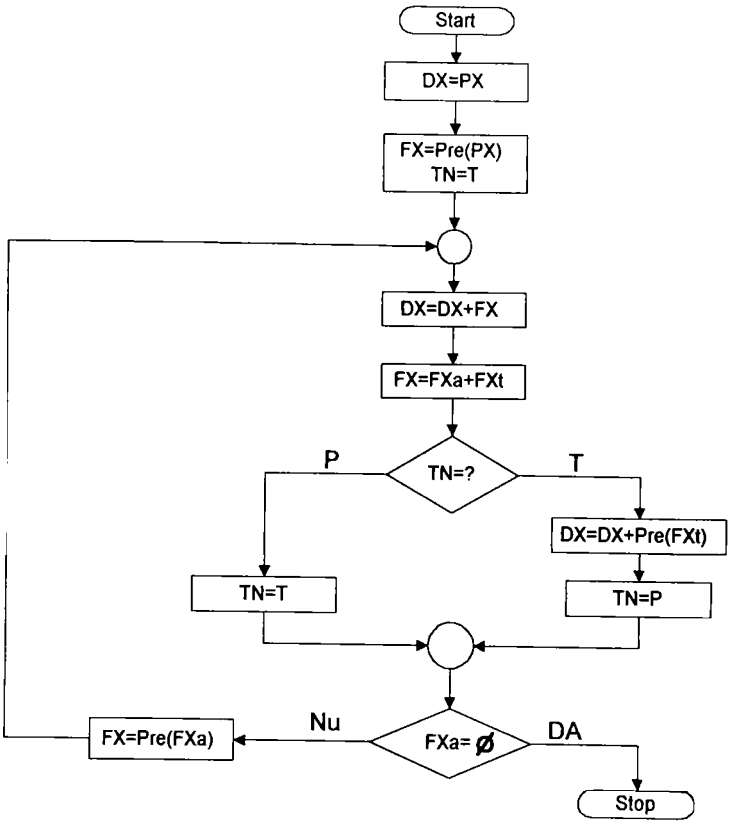


Fig. 7-7 Organigramă pentru determinarea domeniului de precedență a unei restricții

Notațiile din Figura 7-7 au următoarea semnificație:

- PX – este mulțimea locațiilor condiționate prin restricția X;
- DX – este domeniul de precedență a restricției X;
- FX – este mulțimea nodurilor rețelei ce formează limita domeniului de precedență la pasul curent, numit în continuare *frontul* domeniului de precedență al restricției X;
- FXt – *front terminal* este mulțimea nodurilor care, la pasul curent, termină căi de precedență din domeniul de precedență;
- FXa – *front activ* este mulțimea nodurilor care, la pasul curent, nu termină căi de precedență din domeniul de precedență;
- TN – este o variabilă ce specifică tipul nodului din frontul FX. Această variabilă va lua, alternativ, valoarea T pentru tranziție respectiv valoarea P pentru locație;
- Pre(Z) – premulțimea mulțimii nodurilor $Z \subseteq P \cup T$, este o funcție ce alocă mulțimii nodurilor Z nodurile (mulțimea nodurilor) din premulțimea tuturor nodurilor mulțimii Z, Pre(Z): $P \cup T \rightarrow P \cup T$.

Domeniul de precedență se determină printr-un procedeu iterativ. Se observă faptul că mulțimea locațiilor condiționate de restricția dată, PX, respectiv mulțimea tranzițiilor din premulțimea acestora fac parte din domeniul de precedență. Prima iterație începe chiar cu această mulțime, FX=Pre(PX).

Pentru continuarea iterației se definește o funcție care, pe baza definiției 7-1, determină frontul terminal respectiv frontul activ la pasul curent, astfel încât FX=FXa+FXt. În cazul în care nodurile frontului curent sunt tranziții, confrom definiției 7-2, Pre(FXt) aparține domeniului de precedență.

Iterația continuă pe baza nodurilor FXa astfel încât noul front va fi FX=Pre(FXa). Iterațiile se termină în cazul în care FXa=∅.

Într-o posibilă implementare a organigramei din Figura 7-7, domeniul de precedență se descrie printr-o matrice de incidențe N_X^0 având aceleași dimensiuni ca și matricea de incidențe N a rețelei de bază dar având linii și coloane nule în locul locațiilor și tranzițiilor care nu sunt prezente în domeniul de precedență.

Matricea de incidențe a domeniului de precedență se va utiliza pentru determinarea, în timp real, a comenzii celei mai permisive relativ la restricția dată. Din acest motiv, această matrice trebuie minimizată. Minimizarea se va realiza prin extragerea liniilor și coloanelor nule obținând astfel matricea de incidențe minimizată, notată N_X, a domeniului de precedență. Matricea N_X definește o subrețea a rețelei de bază.

Deoarece comanda cea mai permisivă a unei restricții trebuie determinată în funcție de starea actuală a procesului condus și deoarece comanda determinată se aplică rețelei de bază, trebuie definită legătura dintre rețeaua de bază și subrețelele induse prin restricții.

7.3.4 Legătura dintre rețeaua controlată și subrețelele induse prin restricții

Legătura dintre rețeaua de bază și subrețele se definește prin corespondența dintre nodurile respectiv comenzile rețelei de bază și nodurile respectiv comenzile subrețelelor.

În continuare se va proceda la o abordare vectorială. În acest sens se fac următoarele convenții de notare:

- $\mathbf{t}$ – este vectorul tranzițiilor rețelei de bază;
- $\boldsymbol{x}$ – este vectorul de stare a rețelei de bază;
- $\boldsymbol{u}$ – este vectorul de comandă a rețelei de bază;
- $\mathbf{t}_X$ – este vectorul tranzițiilor subrețelei N_X;
- $\boldsymbol{x}_X$ – este vectorul de stare a subrețelei N_X;
- $\boldsymbol{u}_X$ – este vectorul de comandă a subrețelei N_X;

Corespondența dintre elementele acestor vectori este complet definită prin matricea de incidențe, N, a rețelei de bază, matricea de incidențe, N_X^0, a subrețelei și procedeul de minimizare prin care se obține matricea de incidențe minimizată, N_X, a subrețelei.

Cel mai simplu procedeu de minimizare constă în a extrage liniile și coloanele nule din matricea N_X^0 și definirea a doi vectori "mască", t_m pentru tranziții respectiv p_m pentru locații, care reține configurația inițială a coloanelor și liniilor matricei N_X^0. Elemntele vectorilor mască au valoarea nulă (0) pe pozițiile liniilor respectiv a coloanelor nule și valoare unitară (1) pe pozițiile liniilor respectiv a coloanelor nenule.

Pe baza acestor doi vectori mască pot fi definite și implementate funcții de conversie care stabilesc legătura dintre rețeaua de bază și subrețeaua indusă prin restricția dată. Aceste funcții se vor definii într-o formă vectorială, în sensul că aceste funcții vor avea vectori drept argumente. Acest fapt se va notifica prin utilizarea de caractere îngroșate în numele de funcții și argumente respectiv în numele domeniilor de definiție (mulțimilor de vectori).

Se definesc următoarele funcții:

(i) Funcția de compresie a vectorului tranzițiilor, $C_{TX}:T\rightarrow T_X$, astfel încât:

$$\mathbf{t}_X=C_{TX}(\mathbf{t}).$$

(ii) Funcția de compresie a vectorului de stare, $C_{PX}:P\rightarrow P_X$, astfel încât:

$$\mathbf{x}_X=C_{PX}(\mathbf{x}).$$

(iii) Funcția de compresie a vectorului de comandă, $C_{CX}:U\rightarrow U_X$, astfel încât:

$$\mathbf{u}_X=C_{CX}(\mathbf{u}).$$

(iv) Funcția de expandare a vectorului tranzițiilor, $E_{TX}:T_X\rightarrow T$, astfel încât:

$$\mathbf{t}_X^E=E_{TX}(\mathbf{t}_X).$$

(v) Funcția de expandare a vectorului de stare, $E_{PX}:P_X\rightarrow P$, astfel încât:

$$\mathbf{x}_X^E=E_{PX}(\mathbf{x}_X).$$

(vi) Funcția de expandare a vectorului de comandă, $E_{CX}:U_X\rightarrow U$, astfel încât:

$$\mathbf{u}_X^E=E_{CX}(\mathbf{u}_X).$$

În definițiile de mai sus:

- T și T_X reprezintă mulțimea vectorilor tranzițiilor;
- P și P_X reprezintă mulțimea vectorilor de stare;
- U și U_X reprezintă mulțimea vectorilor de comandă;
- Compresia realizează vectorii specifici subrețelei în concordanță cu minimizarea matricei de incidențe a subrețelei.
- Expandarea realizează vectori în concordanță cu ordinea elementelor în matricea N_X^0.

Relativ la vectorii de comandă se vor avea în vedere următoarele:

Mulțimea tranzițiilor controlabile din domeniul de precedență al locației $p\in P$ se va nota prin $T_C(p)$. Mulțimea locațiilor de control din domeniul de precedență al locației $p\in P$ este $C(p)=\{c \mid c\in C, c\in {}^\bullet t_\pi, t_\pi\in T_C(p)\}$, unde C este mulțimea locațiilor de control al rețelei.

Mulțimea tranzițiilor controlate ale domeniului de precedență a restricției X este dată prin:

$$T_{CX} = \bigcup_{p\in P_X} T_C(p)$$

iar mulțimea locațiilor de control ale domeniului de precedență al restricției X prin:

$$C_X = \bigcup_{p\in P_X} C(p)$$

Comanda domeniului de precedenţa a restricţiei X se defineşte prin funcţia:

$$u_X : C_X \rightarrow \{0,1\}$$

Într-o reprezentare vectorială elementele vectorilor se definesc astfel:

$$\mathbf{u}[j] = \begin{cases} u(j) & \text{daca } t_j \in T_C \\ 0 & \text{daca } t_j \notin T_C \end{cases}$$

respectiv,

$$\mathbf{u}_X[j] = \begin{cases} u_X(j) & \text{daca } t_j \in T_{CX} \\ 0 & \text{daca } t_j \in D_X \setminus T_{CX} \end{cases}$$

În expresiile de mai sus s-a avut în vedere faptul că reprezentarea vectorială induce o ordonare strictă a mulţimii tranziţiilor. Acestă ordonare permite utilizarea indecşilor drept variabile.

7.3.5 *Determinarea componentei anticipative a vectorului de comandă*

Domeniul de precedenţă, D_X, a restricţiei X, împreună cu relaţiile de flux din reţeaua iniţială formează o subreţea a acesteia. Matricea de incidenţe a subreţelei, notată prin N_X, conţine doar acele linii şi coloane ale matricii de incidenţe originale ale căror elemente corespunzătoare (locaţii respectiv tranziţii) sunt incluse în domeniul de precedenţă D_X.

Cunoscând starea reţelei se cunoaşte implicit şi marcajul subreţelei N_X. Prin urmare, evoluţia acesteia poate fi studiată pe baza schemei bloc structurale din Figura 7-8.

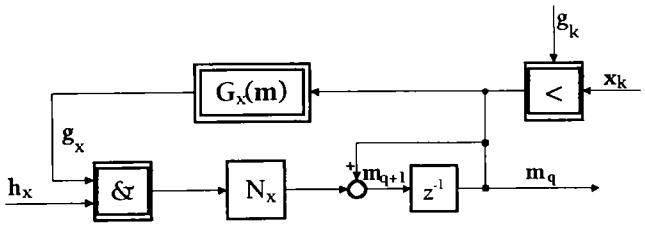


Fig. 7-8 Schema bloc structurală pentru determinarea celei mai permisive comenzi

În Figura 7-8, $\mathbf{h}_X$ este vectorul predicatelor tranziţiilor subreţelei N_X şi se defineşte astfel:

$$\mathbf{h}_X[j] = \begin{cases} u_X(j) & \text{daca } t_j \in T_{CX} \\ 1 & \text{daca } t_j \in D_X \setminus T_{CX} \end{cases} \qquad (7.9)$$

Vectorul $\mathbf{h}_X$ are două categorii de elemente:

1. elemente de comandă $u_X(j)$, corespunzătoare tranziţiilor controlabile;
2. elemente de execuţie, având valoarea "1-logic", deoarece în acest caz scopul execuţiei subreţelei N_X este de a anticipa comportarea ei şi nu de a controla vre-un proces.

Vectorul $\mathbf{h}_X$ poate fi determinat prin următoarea relaţie:

$$\mathbf{h}_X = \mathbf{i} \vee \mathbf{u}_X = \mathbf{i} \vee C_{CX}(\mathbf{u}), \qquad (7.10)$$

unde **i** este vectorul unitate de dimensiune potrivită iar operația de disjuncție logică a doi vectori se definește ca fiind disjuncția logică între elementele corespondente ale vectorilor.

Starea inițială a subrețelei, de la care se inițiază studiul, este o submulțime a stării $\mathbf{x}_k$ a rețelei de bază. În Figura 7-8 modulul notat prin "<" realizează preluarea stării inițiale la începutul fiecărui ciclu de studiu al subrețelei N_x. În Figura 7-8, starea inițială $\mathbf{m}_0$, pentru pornirea studiului la pasul k din rețeaua de bază, se obține prin relația:

$$\mathbf{m}_0=C_{PX}(\mathbf{x}_k). \qquad (7.11)$$

Ciclul de studiu se inițiază în momentul în care cel puțin o tranziție terminală, t_π, a domeniului de precedență, este concesionată. Concesionarea se determină pe baza unei funcții logice $V(g_{Xk})$ ce conține disjuncția logică între elementele vectorului de concesionare, g_{Xk}, ale subrețelei. Acest vector se determină astfel:

$$g_{Xk}=C_{TX}(g_k). \qquad (7.12)$$

Evoluția subrețelei este determinată de structura sa, de starea inițială și de vectorul $\mathbf{h}_X$. Structura subrețelei N_X precum și starea inițială $\mathbf{m}_0$ - starea rețelei în momentul inițierii studiului - sunt constante, variabile fiind acele elemente ale vectorului $\mathbf{h}_X$ care sunt definite prin funcția u_X respectiv prin elementele vectorului de comandă $\mathbf{u}_X$ așa cum se poate observa în relația (7.10).

Comportarea subrețelei N_X trebuie studiată pentru diverse comenzi $\mathbf{u}_X$. Comenzile pentru care $X\in R_N(\mathbf{x}_k,\mathbf{u}_X)$ sunt comenzi nepermise. În continuare, scopul studierii subrețelei N_X este de a găsi cea mai premisivă comandă pentru care rețeaua respectă restricția X. În acest scop se presupune existența unui modul de control care funcționează pe baza diagramei logice din Figura 7-9.

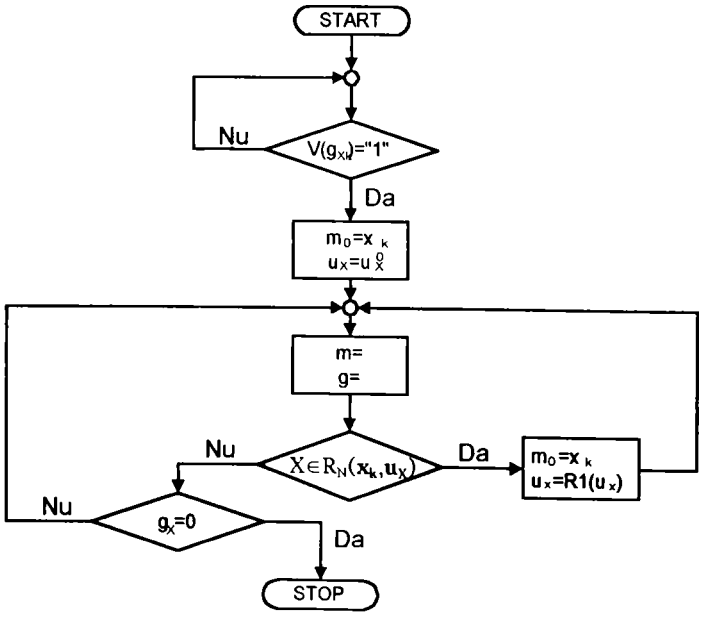


Fig. 7-9 Diagrama logică a modulului de control

Studiul subrețelei se inițiază în cazul în care funcția logică $V(g_k)$ ia valoarea "1-logic". Comanda u_X^0 este cea mai permisivă comadă posibilă a domeniului de precedență a restricției X. În organigrama din Figura 7-9 funcția $R1(\mathbf{u}_X)$ este "funcția de restricție unitară" a comenzii $\mathbf{u}_X$ restricție realizată prin schimbarea în zero a valorii unitare a unui elemnt al vectorului argument.

Cea mai permisivă comandă la starea dată este prima comandă, cu care se ajunge în situația $g_X=0$, caz în care nici o tranziție a subrețelei N_X nu mai este concesionată. Adică, s-au realizat toate marcajele posibile la comanda dată, fără a viola restricția, adică $X\notin R_N(\mathbf{x}_k,\mathbf{u}_X)$.

Deoarece toate căile de precedență sunt controlate, în cazul unei comenzi pentru care $X\notin R_N(\mathbf{x}_k,\mathbf{u}_X)$ se va ajunge cu siguranță în situația $g_X=0$. Comanda $\mathbf{u}_X=\mathbf{0}$ cu siguranță realizează condiția $X\notin R_N(\mathbf{x}_k,\mathbf{u}_X)$. În caz contrar aceasta nu s-ar fi realizat nici pentru starea $\mathbf{x}_0$. Pentru ca restricția să fie respectată la starea inițială x_0 trebuie asigurată consistența restricțiilor cu structura și starea inițială a rețelei.

Prin același procedeu se determină comanda cea mai permisivă pentru fiecare restricție $X\in\aleph=\{X_1,X_2,...\}$. Aceste comenzi sunt funcții $u_{Xi}:C_{Xi}\rightarrow\{0,1\}$, unde $C_{Xi}\subseteq C$ sunt submulțimi ale mulțimii locațiilor de control.

Condiția de bază a controlabilității prin restricții asupra stărilor realizabile ale sistemului este ca la starea inițială, x_0, să se realizeze restricția X, adică să existe o comandă $u_X\geq u_R$ astfel încât $X\notin R_N(\mathbf{x}_0,\mathbf{u}_X)$.

> **Lema 3.** *Dacă restricția controlabilă $X\subseteq M$ este realizabilă la starea $\mathbf{x}_0$ a rețelei Petri atunci ea este realizabilă pentru fiecare marcaj accesibil al rețelei.*

Condiția exprimată în lema 3 este necesară deoarece marcajul inițial al rețelei nu poate fi influențat prin comenzi. În schimb stările ulterioare sunt controlabile iar analiza anticipativă va asigura ca, pentru orice marcaj x_k a rețelei, restricția să fie respectată.

Componenta anticipativă a vectorului de comandă, $\mathbf{u}_A$, se determină cu ajutorul funcției de expandare a comenzilor $E_{CX}(\bullet)$. Având la dispoziție cele mai permisive comenzi, $\mathbf{u}_{X1}$, $\mathbf{u}_{X2}$, $\mathbf{u}_{X3}$, ... $\mathbf{u}_{X|\aleph|}$ corespunzătoare restricțiilor X_1, X_2, X_3 ...$X_{|\aleph|}$ componenta anticipativă a vectorului de comandă se obține astfel:

$$\mathbf{u}_A = \mathbf{E}_{CX1}(\mathbf{u}_{X1})\wedge \mathbf{E}_{CX2}(\mathbf{u}_{X2})\wedge\ldots\wedge\mathbf{E}_{CX|\aleph|}(\mathbf{u}_{X|\aleph|}). \qquad (7.13)$$

Pentru a asigura permisivitatea maximă a comenzii, calculul $\mathbf{u}_{Xi}$ se execută la fiecare pas cât timp $V_{Xi}(g_{Xik})$ are valoarea logică "adevărat".

* * *

În concluzie, sinteza componentei anticipative a vectorului de comandă se realizează prin următorii pași:

a) Analiza "off-line":

   (i) Se determină domeniile de precedență D_{xi} ale restricțiilor $X_i\in\aleph$, conform procedurii prezentate în secțiunea 7.3.3.

   (ii) Se determină funcțiile de compresie și de expandare conform secțiunii 7.3.4.

   (iii) Se verifică, pe baza procedeului prezentat în secțiunea 7.3.5, dacă aceste restricții sunt realizabile la marcajul inițial.

b) Analiza "on-line":

(i) Determinarea valorii funcțiilor $V_{Xi}(g_{Xk})$ corespunzătoare domeniilor de precedență, D_{xi}, și preluarea stării la care se inițiază studiul, pe baza procedeului prezentat în secțiunea 7.3.5;

(II) Realizarea studiului, pe baza organigramei din Figura 7-9, în vederea determinării comenzii celei mai permisive care mai asigură respectarea restricției;

(iii) Determinarea componentei anticipative a vectorului de comandă pe baza relației (7.13).

În comparație cu metoda prezentată în [HGZ'96] această metodă este deosebit de simplă și elegantă. În același timp, dispunând de o implementare a mașinii virtuale de rețea Petri, implementarea metodei este și deosebit de simplă deoarece aceasta este de fapt o restricție a implementării inițiale. Modulul de control necesar analizei "on-line" este deasemenea ușor de implementat și în același timp, după o parametrizare adecvată, este aplicabil tuturor restricțiilor.

7.4 Metodă bazată pe invarianți pentru determinarea componentei de sincronizare a vectorului de comandă

În cazul acestei metode restricțiile se formulează prin inegalități liniare compuse din elementele vectorului de marcaj și a unor constante și variabile întregi. Se va arăta că este posibilă transformarea în astfel de inegalități și a altor forme de prezentare a restricțiilor ca de exemplu a celor formulate prin expresii Booleene sau a celor formulate cu elemente ale vectorului de concesionare. Metoda se bazează pe ideile prezentate în [Yam95] dar propune o implementare, realizabilă pe baza conceptului de rețea Petri, care lasă intact modelul de rețea Petri al procesului condus și care are ca rezultat un modul ce implementează însăși strategia de conducere.

7.4.1 Specificarea restricțiilor

Restricțiile pe care trebuie să le respecte sistemul controlat, se presupun a fi de forma:

$$\sum_{i=1}^{n} \alpha_i \mu_i \le \beta \qquad (7.15)$$

unde μ_i este marcajul locației p_i iar α_i și β sunt constante întregi. Spre exemplu, în cazul unei rețele C/E se poate impune ca $\mu_1 + \mu_2 \le 1$, ceea ce însemnă că doar una dintre locațiile p_1 și p_2 pot fi marcate, altfel spus nu pot fi marcate simultan amândouă locațiile. Introducînd o variabilă auxiliară nenegativă μ_c, inegalitatea (7.15) poate fi transformată într-o egalitate. Condiția de mai sus devine $\mu_1 + \mu_2 + \mu_c = 1$, sau în general:

$$\sum_{i=1}^{n} \alpha_i \mu_i + \mu_c = \beta \qquad (7.16)$$

Având în vedere faptul că, relația (7.16) se referă la un sistem modelat printr-o rețea Petri cu numărul locațiilor |P|, pozitiv și finit, rezultă n≤|P|. Dacă n_c este numărul restricțiilor de forma (7.15) pe care trebuie să le satisfacă sistemul, **m** vectorul de marcaj (|P|×1-dimensional) al rețelei Petri care modulează sistemul, **b** un vector cu elemente întregi de dimensiune n_c×1, iar A o matrice de dimensiune n_c×|P|, atunci restricțiile de forma (7.15) pot fi exprimate vectorial astfel:

$$\mathbf{A} \cdot \mathbf{m} \le \mathbf{b} \qquad (7.17)$$

Restricţiile de forma (7.17) pot fi exprimate în formă vectorială introducând vectorul $\mathbf{m}_c$, $n_c|\times 1$–dimensional, al variabilelor auxiliare, sub forma:

$$\mathbf{A}\cdot\mathbf{m}+\mathbf{m}_c=\mathbf{b} \tag{7.18}$$

Relaţia de mai sus poate fi pusă în forma:

$$\begin{bmatrix}\mathbf{A} & \mathbf{I}\end{bmatrix}\cdot\begin{bmatrix}\mathbf{m}\\ \mathbf{m}_c\end{bmatrix}=\mathbf{b} \tag{7.19}$$

unde $\mathbf{I}$ este matricea unitate $n_c\times n_c$ dimensională. Relaţia (7.16) poate fi pusă şi sub forma echivalentă:

$$\begin{bmatrix}\mathbf{m}\\ \mathbf{m}_c\end{bmatrix}^T\cdot\begin{bmatrix}\mathbf{A} & \mathbf{I}\end{bmatrix}^T=\mathbf{b}^T \tag{7.20}$$

Interpretând variabilele auxiliare $\mathbf{m}_c$ ca marcări ale unor locaţii suplimentare în reţeaua de bază, relaţia (7.20) este o relaţie de invarianţă, $\begin{bmatrix}\mathbf{A} & \mathbf{I}\end{bmatrix}$ fiind un invariant în reţeaua completată cu aceste locaţii suplimentare. Se poate determina structura reţelei în care este valabilă relaţia (7.20) observând că fiecare locaţie suplimentară introduce o nouă linie în matricea de incidenţe a reţelei completate. Dacă $\mathbf{N}$ este matricea de incidenţe a reţelei de bază, iar $\mathbf{N}_c$ este matricea formată din liniile corespunzătoare locaţiilor suplimentare, condiţia care exprimă ca $\begin{bmatrix}\mathbf{A} & \mathbf{I}\end{bmatrix}$ să fie invariant al reţelei completate este de forma:

$$\begin{bmatrix}\mathbf{N}\\ \mathbf{N}_c\end{bmatrix}^T\cdot\begin{bmatrix}\mathbf{A} & \mathbf{I}\end{bmatrix}^T=0 \tag{7.21}$$

Matricea $\mathbf{N}_c$ specifică arcele dintre tranziţiile reţelei de bază şi locaţiile suplimentare, de control. Astfel, fiind dată matricea de incidenţe $\mathbf{N}$ a procesului condus, se poate determina matricea $\mathbf{N}_c$ prin care se realizează restricţiile de forma (7.15):

$$\mathbf{N}_c=-\mathbf{A}\cdot\mathbf{N} \tag{7.22}$$

Marcajul iniţial al locaţiilor suplimentare se determină pe baza relaţiei (7.18), care trebuie să fie satisfăcută la marcajul iniţial, şi se obţine:

$$\mathbf{m}_{c0}=\mathbf{b}-\mathbf{A}\cdot\mathbf{m}_0 \tag{7.23}$$

Structura de conducere obţinută este prezentată în Figura 7-10.

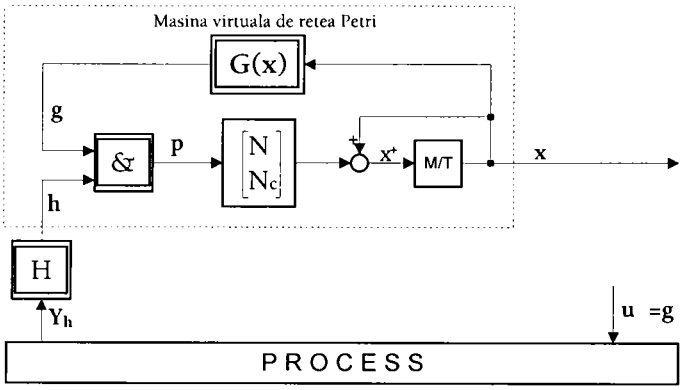


Fig. 7-10 Structura de conducere obţinută pentru restricţiile date

În Figura 7-10, SC lucrează pe baza matricii de incidenţe completată a reţelei de bază. Mărimile de comandă se obţin imediat fiind identice cu elementele vectorului de concesionare g . Această simplificare se plăteşte, pe lângă dezavantajele prezentate la începutul acestui capitol, şi prin următoarele dezavantaje:

(i) structura SC se amplifică proporţional cu numărul de restricţii impuse. Dezavantajul constă în faptul că operaţia de determinare a vectorului de stare x^+ este indivizibilă respectiv paralelizarea ei este dificilă.

(ii) schimbarea de structură a reţelei Petri impune şi schimbarea OPT din procesul condus. Schimbarea de structură se "propagă" în tot sistemul.

Cu alte cuvinte, ocolind sinteza directă a strategiei de control, se dispersează schimbările structurale. Cu toate acestea în anumite situaţii poate să fie avantajoasă această metodă de control, mai ales în situaţii în care OPT se poate reconfigura simplu şi eventual automatizat (pentru a evita erorile de reconfigurare).

Dezavantajele de mai sus pot fi evitate în cazul în care restricţiile se realizează prin strategia de control H(•) aşa cum s-a definit în capitolul 6. Pentru a obţine comportarea echivalentă, elementele vectorului de comandă, **u** , se vor determina conform următoarei relaţii, prin modulul notat U(**x**,**c**,**w**) în Figura 7-11:

$$\mathbf{u}[j]=\mathbf{H}_j(\mathbf{x},\mathbf{c})=\begin{cases}1, \textit{daca} -\begin{bmatrix}\mathbf{t}_j^-\\ \mathbf{t}_{cj}^-\end{bmatrix}\le\begin{bmatrix}\mathbf{x}\\ \mathbf{c}\end{bmatrix}\le\mathbf{k}-\left(\begin{bmatrix}\mathbf{t}_j^+\\ \mathbf{t}_{cj}^+\end{bmatrix}+\begin{bmatrix}\mathbf{t}_j^-\\ \mathbf{t}_{cj}^-\end{bmatrix}\right)\\ 0, \textit{altfel}\end{cases}, \quad 0<j\le|T| \quad (7.24)$$

unde prin **x** s-a notat starea sistemului de bază (vectorul de marcaj al reţelei de bază), iar prin **c** vectorul de marcaj a locaţiilor de control. Vectorii $\mathbf{t}_{cj}^+$ şi $\mathbf{t}_{cj}^-$ specifică conexiunile dintre locaţiile suplimentare şi tranziţiile reţelei originale. Aceşti vectori sunt parte componentă a stragiei de comandă. Vectorii $\begin{bmatrix}\mathbf{x}\\ \mathbf{c}\end{bmatrix}, \begin{bmatrix}\mathbf{t}_j^-\\ \mathbf{t}_{cj}^-\end{bmatrix}$ şi $\begin{bmatrix}\mathbf{t}_j^+\\ \mathbf{t}_{cj}^+\end{bmatrix}$ se obţine prin agregarea vectorilor componenţi.

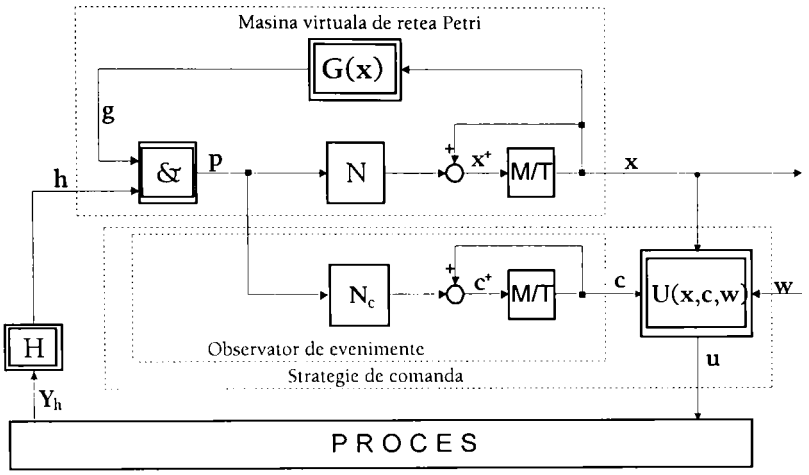


Fig. 7-11 Structura de conducere cu modul distinct (controler) pentru implementarea strategiei de conducere

Vectorul de marcaj al reţelei de bază se determină prin relaţia de mai jos (notaţiile fiind cele uzuale):

$$\mathbf{x}^+ = \mathbf{x} + \mathbf{N} \cdot \mathbf{p}\,, \qquad \mathbf{x}_0 = \mathbf{m}_0 \tag{7.25}$$

Vectorul marcajelor de control şi marcajul iniţial al cestora se determină prin relaţiile:

$$\mathbf{c}^+ = \mathbf{c} + \mathbf{N}_C \cdot \mathbf{p} \tag{7.26}$$

$$\mathbf{c}_0 = \mathbf{b} - \mathbf{A} \cdot \mathbf{m}_0 \tag{7.27}$$

Structura de control care realizează restricţiile impuse conform celor de mai sus se prezintă în Figura 7-11.

În acest caz structura iniţială a reţelei se păstrează intactă atât la nivelul SC cât şi la nivelul OPT. Apare suplimentar modulul care realizează strategia de conducere care prin relaţiile (7.22), (7.26), (7.27) este complet definită. Modulul care determină vectorul $\mathbf{c}^+$ este de fapt un observator de evenimente în sensul că valoarea acestuia este determinată de valoarea anterioară $\mathbf{c}$ şi de evenimentele apărute în sistem.

În cele ce urmează, modulul software sau echipamentul (hardware+software) care implementează strategia de conducere se va numi *controler*. În Figura 7-11 s-au prezentat şi mărimile de conducere prin vectorul w. Influenţa acestora asupra vectorului de comandă nu se tratează în acest capitol.

Observaţie: *Această metodă de sinteză a strategiei de conducere presupune implicit că toate tranziţiile afectate sunt sau pot fi făcute controlabile.*

7.4.2 Permisivitatea strategiei de conducere bazată pe invarianţi

Fie $\mathbf{X}$ o matrice cu elemente întregi având coloanele liniar independente, matrice ce reprezintă o bază pentru P–invarianţii reţelei de bază. În acest caz matricea $\mathbf{X}$ satisface ecuaţia:

$$\mathbf{X}^T \cdot \mathbf{N} = 0 \tag{7.28}$$

Numărul coloanelor matricii **X** (deci numărul P–invarianţilor reţelei de bază) este |P|–rangN, deoarece N este o matrice |P|×|T| –dimensională şi **X** este o bază pentru spaţiul nul al N. În cazul în care rangN=|P| reţeaua de bază nu are P–invarianţi.

Matricea de incidenţă a sistemului controlat este:

$$C = \begin{bmatrix} N \\ N_c \end{bmatrix} = \begin{bmatrix} N \\ -AN \end{bmatrix} \tag{7.29}$$

Având în vedere faptul că liniile matricii N_c sunt combinaţii liniare ale liniilor matricii N, se obţine rangC=rangN. Astfel, numărul invarianţilor sistemului controlat este $|P|+n_c$-rangN.

Fiind valabilă relaţia $\begin{bmatrix} X^T & 0 \end{bmatrix} \cdot \begin{bmatrix} N \\ N_c \end{bmatrix} = X^T \cdot N = 0$, rezultă că P–invarianţii reţelei de bază sunt şi P–invarianţi ai sistemului controlat. Acest lucru este adevărat pentru orice strategie de control care introduce doar locaţii şi arce în reţeaua de bază. Datorită modului de formulare a restricţiilor este valabilă şi relaţia:

$$\begin{bmatrix} A & I \end{bmatrix} \cdot \begin{bmatrix} N \\ N_c \end{bmatrix} = \begin{bmatrix} A & I \end{bmatrix} \cdot \begin{bmatrix} N \\ -AN \end{bmatrix} = A \cdot N - A \cdot N = 0. \tag{7.30}$$

Prin urmare toţi P–invarianţii reţelei controlate (în număr de $|P|+n_c$-rangN) sunt daţi de relaţia $X_c \cdot N = 0$, unde:

$$X_c = \begin{bmatrix} X & A^T \\ 0 & I \end{bmatrix},$$

şi prin urmare:

$$X_c^T \cdot \begin{bmatrix} N \\ N_c \end{bmatrix} = \begin{bmatrix} X & 0 \\ A & I \end{bmatrix} \cdot \begin{bmatrix} N \\ N_c \end{bmatrix} = \begin{bmatrix} X^T & 0 \\ A & I \end{bmatrix} \cdot \begin{bmatrix} N \\ -AN \end{bmatrix} = X^T \cdot N + A \cdot N - A \cdot N = X^T \cdot N = 0$$

Rangul matricii X_c se exprimă prin rangX=n+n_c-rangN, deoarece rangX=n-rangN, iar **I** este matricea unitară $n_c \times n_c$–dimensională.

Deci nu apar invarianţi suplimentari forţaţi în sistem datorită strategiei de control. Aceasta este maximal permisivă deoarece nu sunt inhibate alte acţiuni decât cele rezultate din structura de bază şi restricţiile formulate în strategia de conducere.

* * *

Metoda de specificare a strategiei de control prezentată mai sus impune formularea restricţiilor pe baza sumelor ponderate ale marcajelor locaţiilor. Pentru ca această metodă să fie aplicabilă şi altor forme de prezentare a restricţiilor acestea trebuie să fie transformate în forma acceptată de această metodă. Se prezintă mai jos diverse forme de reprezentare a restricţiilor împreună cu posibilităţile de transformare ale acestora în forma cerută de metodă.

7.4.3 Restricţii formulate prin expresii logice.

Dacă o restricţie poate fi scrisă în forma normală conjunctivă din algebra Booleană:

$$A \rightarrow \Phi_1 \wedge \Phi_2 \wedge \ldots \Phi_q \tag{7.31}$$

unde A este o variabilă logică şi fiecare dintre disjuncţiile Φ_i este de forma:

$$\Phi_i = \Psi_{i1} \vee \Psi_{i2} \vee \ldots \Psi_{ihi} \qquad (7.32)$$

şi fiecare Ψ_{ij} este o variabilă logică, atunci ea poate fi exprimată în mod echivalent printr-o mulţime de inegalităţi de forma:

$$(1 - \Psi_{i1}) + (1 - \Psi_{i2}) + \ldots + (1 - \Psi_{ihi}) + A \le h_i,\ 0 < i \le q \qquad (7.33)$$

Mulţimile acestor inegalităţi reprezintă restricţiile pe care sistemul trebuie să le satisfacă. Din eventualele apariţii multiple ale acestor inegalităţi se va păstra una singură. Aceste inegalităţi vor fi prelucrate aşa cum se prezintă mai jos.

7.4.4 Restricţii de forma "mai mic sau egal".

7.4.4.1 Restricţii conţinând doar elemente ale vectorului de marcaj.

Presupunem că, restricţiile pot fi exprimate printr-o inegalitate de forma:

$$\sum_{i=1}^{r} \mu_i \le k \qquad (7.34)$$

Aceasta semnifică faptul că suma marcajelor locaţiilor $p_1, p_2, \ldots, p_r$ ale reţelei Petri nu trebuie să depăşească în nici un caz valoarea întreagă k. Această formă este cea cerută de metodă. Introducând variabila auxuliară μ_c, condiţia poate fi scrisă sub forma unei egalităţi, astfel:

$\sum_{i=1}^{r} \mu_i + \mu_c = k$. Această ecuaţie are din nou forma cerută de metodă.

7.4.4.2 Restricţiile formulate sub forma unor sume ponderate ale elementelor vectorului de marcaj

Astfel de restricţii pot fi specificate prin expresii de forma:

$$\sum_{i=1}^{r} a_i \mu_i \le k \qquad (7.35)$$

şi se tratează ca la în secţiunea 7.4.1.

Constanta k nu influenţează în nici un fel structura reţelei rezultate (adică numărul locaţiilor de control sau pe cel al arcelor). Ea specifică doar numărul de marcări din invarianţi respectiv marcajul iniţial al locaţiilor de control.

7.4.4.3 Restricţii conţinând elemente ale vectorului de marcaj şi ale vectorului de concesionare.

În cazul în care în formularea restricţiei se iau în considerare atât elemente ale vectorului de marcaj cât şi ale vectorului de concesionare, expresia trebuie adusă la o formă echivalentă, în care apar doar elemente ale vectorului de marcaj. În cele de mai jos se vor prezenta metode de transformare pentru reţele C/E.

a) Se vor considera restricţii în care intră un singur element al vectorului de concesionare, având forma:

$$\sum_{i=1}^{r} \mu_i + g_j \le r \qquad (7.36)$$

unde numărul locaţiilor care intră în expresie este egal cu constanta r din partea dreaptă a expresiei. Restricţia exprimă faptul că tranziţia t_j nu poate fi executată în cazul în care toate locaţiile μ_i sunt marcate. Utilizând regula de tranziţie în reţele C/E care specifică

faptul că o tranziție poate fi executată dacă și numai dacă toate locațiile premulțimii sunt valide (marcate), putem înlocui g_j prin suma elementelor premulțimii tranziției t_j, modificând în mod corespunzător partea dreaptă a expresiei. Dacă q_j este numărul locațiilor din premulțimea tranziției t_j, expresia (7.30) devine:

$$\sum_{i=1}^{r} \mu_i + \sum_{j=1}^{qj} \mu_j \leq r + q_j - 1 \qquad (7.37)$$

În partea dreaptă a expresiei (7.37) pot să fie simultan valide (marcate) maxim $r+q_j-1$ locații. Astfel, dacă toate locațiile $p_1,...,p_r$ sunt marcate, atunci nu pot fi valide toate locațiile din premulțimea tranziției t_j, ca urmare t_j nu poate să fie concesionată.

Pe de altă parte, dacă toate locațiile premulțimii tranziției t_j sunt valide, pot să fie valide maxim r-1 locații din mulțimea $p_1,...,p_r$. Expresia restricției, în forma (7.37) conține doar elemente ale vectorului de marcaj. Trebuie să observăm că se tratează corect și cazul în care există locații care participă în amândouă sumele expresiei (7.37).

b) În cazul în care restricția se exprimă prin:

$$\sum_{i=1}^{r} \mu_i + g_j \leq k \qquad (7.38)$$

unde k<r, trebuie să distingem două cazuri:

(i) premulțimea tranziției t_j are un singur element;

(ii) premulțimea tranziției t_j are mai multe elemente.

În cazul (i) g_j poate fi înlocuit prin marcajul locației μ_j din premulțimea tranziției t_j și expresia devine :

$$\sum_{i=1}^{r} \mu_i + \mu_j \leq k \qquad (7.39)$$

Forma (7.39) este echivalentă formei (7.38) dar conține doar elemente ale vectorului de marcaj.

În cazul (ii) (premulțimea tranziției t_j conține mai mult decât un element) expresia (7.38) se va înlocui printr-o mulțime echivalentă de restricții care exclud toate cazurile nepermise de (7.38).

Mulțimea de restricții echivalente (dacă toate sunt simultan satisfăcute) este de forma:

$$\sum_{i=1}^{r} \mu_i \leq k$$

$$\mu_1 + \mu_2 + ... + \mu_{k-1} + \mu_k + g_j \leq k \wedge$$

$$\mu_1 + \mu_2 + ... + \mu_{k-1} + \mu_{k+1} + g_j \leq k \wedge$$

$$\mu_1 + \mu_2 + ... + \mu_{r-1} + \mu_r + g_j \leq k \wedge$$

$$\mu_2 + \mu_3 + ... + \mu_k + \mu_{k+1} + g_j \leq k \wedge$$

$$\mu_2 + \mu_3 + ... + \mu_k + \mu_r + g_j \leq k \wedge$$

$$\mu_{t-k+1} + \mu_{t-k+2} + \ldots + \mu_{t-1} + \mu_t + g_j \le k \qquad (7.40)$$

Numărul de linii este egal cu numărul de combinaţii ale lui r luate câte k. Astfel, fiecare expresie conţine k elemente de marcaj din cele r ale expresiei (7.38) pe lângă g_j şi este de forma (7.15).

c) Restricţia poate fi exprimată într-o formă mai generală prin următorea expresie:

$$\sum_{i=1}^{r} \mu_i + \sum_{j=1}^{qj} g_j \le k \quad \text{cu } k < q_j + r \qquad (7.41)$$

Expresia (7.41) trebuie transformată untr-un set de restricţii echivalente adică un set de inegalităţi care conţin toate restricţiile cuprinse în relaţia (7.41). Aceste inegalităţi vor fi de forma uneia dintre expresiile descrise în această secţiune şi vor conţine cel mult un element de marcaj g_j, j=1, ..., q_j. Fiecare expresie poate fi transformată aşa cum s-a prezentat mai sus.

7.4.4.4 Restricţii conţinând doar elemente ale vectorului de concesionare

Aceste restricţii se referă la concesionări simultane a mai multor tranziţii. Se vor considera toate cazurile posibile. Transformările ce se vor prezenta sunt valabile în cazul reţelelor C/E.

a) Cazul restricţiilor de forma:

$$\sum_{j=1}^{r} g_j \le r - 1 \qquad (7.42)$$

Aceste restricţii specifică faptul că nu toate tranziţiile $g_1, g_2, \ldots, g_r$ pot să fie concesionate simultan. Regula de concesionare sugerează faptul că se pot obţine forme echivalente prin înlocuirea fiecărei tranziţii cu suma locaţiilor din premulţimea acestora. Expresia echivalentă va fi de forma:

$$\sum_{j=1}^{r} (\mu_{j1} + \mu_{j2} + \ldots + \mu_{cj}) \le \sum_{j=1}^{r} c_j - 2 \qquad (7.43)$$

unde $\mu_{j1}, \mu_{j2}, \ldots, \mu_{jc}$ sunt marcajele locaţiilor premulţimii tranziţiei t_j iar c_j este numărul acestor locaţii. În această formă expresia nu permite ca toate locaţiile din premulţimile tuturor tranziţiilor să fie simultan valide. Astfel, se asigură ca să nu fie simultan concesionate toate tranziţiile şi expresiile conţin doar elemente ale vectorilor de marcaj.

b) Orice alte tipuri de restricţii conţinând doar elemente ale vectorilor de concesionare pot fi exprimate în forma:

$$\sum_{j=1}^{r} g_j \le k \quad \text{cu } k \le r - 2 \qquad (7.44)$$

Aceasta trebuie să fie exprimată într-o formă echivalentă sub forma unei mulţimi de inegalităţi de forma (7.42), fiecare conţinând k+1 elemente din vectori de concesionare:

$$g_1 + g_2 + \ldots + g_k + g_{k+1} \le k \quad \wedge$$

$$g_1 + g_2 + \ldots + g_k + g_{k+2} \le k \quad \wedge$$

$$g_1 + g_2 + \ldots + g_k + g_r \le k \quad \wedge$$

$$g_2 + g_3 + \ldots + g_{k+1} + g_{k+2} \le k \quad \wedge$$

$$g_2 + g_3 + \ldots + g_{k+1} + g_r \le k \quad \wedge$$

$$g_{r-k} + g_{r-k+1} + \ldots + g_{r-1} + g_r \le k \qquad (7.45)$$

Fiecare dintre aceste inegalităţi este dată în forma (7.42) şi poate fi transformată aşa cum s-a prezentat mai înainte.

7.4.5 Restricţii de forma "mai mare sau egal"

7.4.5.1 Restricţii conţinând doar elemente ale vectorilor de marcaj.

În unele cazuri este util să se impună ca un set de locaţii să conţină un minim de k marcări. Aceasta se exprimă prin:

$$\sum_{i=1}^{r} a_i \mu_i \ge k \qquad (7.46)$$

Această expresie impune ca suma ponderată a marcajelor celor *r* locaţii să fie mai mare sau egal cu constanta *k*. Această inegalitate poate fi transformată într-o egalitate prin introducerea variabilei auxiliare de exces μ_e în partea stângă:

$$\sum_{i=1}^{r} a_i \mu_i - \mu_e = k \qquad (7.47)$$

Prin această relaţie se forţează un P–invariant în reţeaua controlată formată din locaţiile $\mu_1, \mu_2, \ldots, \mu_r$ şi μ_e. Restricţia în forma (7.47) poate fi tratată similar cu (7.16), doar că în acest caz coeficintul variabilei auxiliare este -1.

Considerând notaţiile din secţiunea 7.4.1 structura de conducere obţinută se specifică astfel:

(i) matricea de incidenţe se determină pe baza relaţiei de invarianţă, astfel:

$$[A \quad I] \cdot \begin{bmatrix} N \\ N_c \end{bmatrix} = 0 \Rightarrow A \cdot N - N_c = 0 \qquad \text{adică:}$$

$$N_c = A \cdot N \qquad (7.48)$$

(ii) vectorul marcajelor iniţiale ale locaţiilor de control se obţin pe baza relaţiei (7.47) astfel:

$$A \cdot m_0 - m_{c0} = k \Rightarrow m_{c0} = A \cdot m_0 - k \qquad (7.49)$$

7.4.5.2 Restricţii conţinând elemente ale vectorului de marcaj şi ale vectorului de concesionare.

Transformările ce se vor prezenta sunt valabile doar pentru reţele C/E.

a) În primul caz considerat restricţia se referă doar la un singur element al vectorului de marcaj şi un singur element al vectorului de concesionare:

$$\mu_i + g_i \ge 1 \qquad (7.50)$$

Prin această expresie se impune ca ori de câte ori p_i nu este marcată, adică μ_i=0, tranziţia t_j trebuie să fie concesionată, adică g_j=1, respectiv ori de câte ori tranziţia tj nu este concesionată, adică g_j=0, locaţia p_i trebuie să fie marcată, adică μ_i=1. Pentru a transforma această expresie într-una care conţine doar elemente ale vectorului de marcaj, va trebui să aducem această expresie la o formă normată de funcţii logice. Expresia (7.50) poate fi înlocuită de expresia logică echivalentă:

$$\neg\mu_i \rightarrow g_j \qquad (7.51)$$

Acestă expresie specifică faptul că, dacă variabila logică μ_i nu este adevărată (nu este validă), variabila logică g_j trebuie să fie adevărată, adică t_j trebuie să fie concesionată, şi faptul că ori de câte ori g_j nu este adevărată (t_j nu este concesionată), μ_i trebuie să fie adevărată (adică p_i trebuie să fie validă, marcată). Acest lucru ne permite să înlocuim g_j în (7.29.) printr-o conjuncţie a tuturor locaţiilor premulţimii sale. Se obţine astfel:

$$\neg\mu_i \rightarrow \mu_{j1} \wedge \mu_{j2} \wedge \ldots \wedge \mu_{cj} \qquad (7.52)$$

unde c_j este numărul locaţiilor din premulţimea tranziţiei t_j. Această expresie conţine doar elemente ale vectorului de marcaj. Ea este o expresie în forma conjunctivă normală şi, aşa cum s-a arătat în secţiunea 7.4.3 este echivalentă şi poate fi înlocuită prin următoarea mulţime de inegalităţi:

$$(1-\mu_{j1})+(1-\mu_i)\le 1 \Rightarrow \mu_{j1}+\mu_i \ge 1 \cdot \wedge$$

$$(1-\mu_{j2})+(1-\mu_i)\le 1 \Rightarrow \mu_{j2}+\mu_i \ge 1 \cdot \wedge$$

$$(1-\mu_{jcj})+(1-\mu_i)\le 1 \Rightarrow \mu_{jcj}+\mu_i \ge 1 \qquad (7.53)$$

Termenul $\neg\mu_i$ s-a putut înlocui prin 1-μ_i,deoarece într-o reţea C/E $\neg\mu_i$ şi (1-μ_i) au aceeaşi valoare:

(i) pentru μ_i=1 $\neg\mu_i$ =0 şi 1-μ_i=0

(ii) pentru μ_i=0 $\neg\mu_i$ =1 şi 1-μ_i=1

Toate inegalităţile din (7.53) sunt de forma (7.50) şi pot fi tratate ca atare.

b) Forma generală a restricţiilor acestei categorii este:

$$\sum_{i=1}^{r}\mu_i + \sum_{j=1}^{q} g_j \ge k. \qquad (7.54)$$

Acestă expresie trebuie adusă la forma normală a unei expresii logice iar pe urmă fiecare parte a expresiilor logice se va trata ca mai sus. Expresiile logice în forma normală trebuiesc minimizate pentru a evita redondanţa.

7.4.5.3 Restricţii conţinând doar elemente ale vectorului de concesionare.

Aceste restricţii se exprimă în următoarea formă generală:

$$\sum_{j=1}^{q} g_j \ge k \qquad (7.55)$$

Această expresie impune ca în orice moment cel puţin k din cele q tranziţii $t_1,t_2,\ldots,t_q$ să fie concesionate. Şi în acest caz se va înlocui această expresie prin expresii logice, în forma

normală. Metodele de transformare ce se vor prezenta sunt valabile pentru reţelele C/E. Pentru exemplu se va considera cazul simplu de mai jos:

$$g_1 + g_2 \geq 1 \tag{7.56}$$

Acesta expresie impune ca în orice moment cel puţin una dintre cele două tranziţii trebuie să fie concesionată. Printr-o expresie logică aceasta se exprimă astfel:

$$\neg g_1 \rightarrow g_2 \tag{7.57}$$

Fiecare element al vectorului de concesionare, g_j, din expresia de mai jos poate să fie înlocuită printr-o conjuncţie a marcajelor locaţiilor premulţimii ei. Se obţine astfel:

$$\neg\left[\mu_{11} \wedge \mu_{12} \wedge \ldots \wedge \mu_{c1}\right] \rightarrow \mu_{21} \wedge \mu_{22} \wedge \ldots \wedge \mu_{c2} \tag{7.58}$$

unde $\mu_{11},\mu_{12},\ldots,\mu_{c1}$ si $\mu_{21},\mu_{22},\ldots,\mu_{c2}$ sunt marcajele locaţiilor din premulţimea tranziţiei t_1 respectiv t_2. Expresia (7.37) poate fi scrisă astfel:

$$\begin{aligned}&\left(\neg\mu_{11} \rightarrow \mu_{21} \wedge \mu_{22} \wedge \ldots \wedge \mu_{c2}\right) \wedge \\ &\left(\neg\mu_{12} \rightarrow \mu_{21} \wedge \mu_{22} \wedge \ldots \wedge \mu_{c2}\right) \wedge \\ &\left(\neg\mu_{1c1} \rightarrow \mu_{21} \wedge \mu_{22} \wedge \ldots \wedge \mu_{c2}\right) \wedge\end{aligned} \tag{7.59}$$

Fiecare expresie este de forma (7.30) şi fiecare poate fi înlocuită de o mulţime de inegalităţi aşa cum s-a arătat în secţiunea 7.2.1.

7.4.6 Restricţii sub forma unor egalităţi.

7.4.6.1 Restricţii conţinând doar elemente ale vectorului de marcaj.

Aceste restricţii se scriu sub forma:

$$\sum_{i=1}^{r} \mu_i = k \tag{7.60}$$

Această ecuaţie exprimă faptul că locaţiile $p_1,p_2,\ldots,p_r$ formează un invariant. Aceasta este o condiţie formulată prin elementele reţelei de bază şi ca atare ar trebui să fie asigurată prin structura acesteia. Dacă totuşi nu este, trebuie procedat la modificarea matricii de incidenţe N a reţelei Petri, astfel încât să fie satisfăcută relaţia:

$$\mathbf{X}^T \cdot \mathbf{N} = \mathbf{0} \tag{7.61}$$

unde X conţine P-invariantul specificat prin (7.60).

Noile elemente ale matricii de incidenţe N reprezintă arce suplimentare prin care se realizează P-invariantul suplimentar.

7.4.6.2 Restricţii în forma unor egalităţi conţinând atât elemente ale vectorului de marcaj cât şi elemente ale vectorului de concesionare.

Presupunem că această categorie de restricţii se exprimă prin:

$$\sum_{i=1}^{k} \mu_i + \sum_{j=1}^{q} g_j = k + q \tag{7.62}$$

Această ecuaţie exprimă faptul că în orice moment $\mu_1,\mu_2,\ldots,\mu_k$ trebuie să fie marcate şi toate tranziţiile $g_1,g_2,\ldots,g_q$ trebuie să fie concesionate.

Probabilitatea de a impune o astfel de restricţie unui sistem este destul de mică dar se poate întâmpla să fie totuşi necesară.

Expresia poate fi transformată înlocuind g_j prin suma marcajelor locaţiilor premulţimii ei şi modificând în consecinţă partea dreaptă. Transformarea este valabilă doar pentru reţele C/E. Forma transformată devine:

$$\sum_{i=1}^{k} \mu_i + \sum_{j=1}^{q} \sum_{p=1}^{cj} \mu_{jp} = k + \sum_{j=1}^{q} c_j \qquad (7.63)$$

unde c_j este numărul locaţiilor din premulţimea tranziţiei t_j. S-a ajuns la forma (7.60).

7.4.6.3 Restricţii sub forma unor egalităţi conţinând doar elemente ale vectorilor de concesionare.

Pot fi formulate restricţii doar pe baza elementelor vectorului de concesionare. Un exemplu simplu ar fi:

$$g_i + g_j = 1 \qquad (7.64)$$

Această ecuaţie specifică faptul că una dintre cele două tranziţii trebuie să fie întotdeauna concesionată. Cu alte cuvinte dacă g_i nu este concesionată, atunci g_j trebuie să fie concesionată, şi viceversa. Aceasta poate fi scrisă sub forma unei expresii logice:

$$(g_i \rightarrow \neg g_j) \wedge (\neg g_i \rightarrow g_j) \qquad (7.65)$$

În cazul unor reţele C/E aceeaşi condiţionare poate fi scrisă prin utilizarea marcării locaţiilor premulţimilor, sub forma:

$$([\mu_{i1} \wedge \mu_{i2} \wedge \ldots \wedge \mu_{ci}] \rightarrow \neg\mu_{j1} \wedge \mu_{j2} \wedge \ldots \wedge \mu_{cj}) \wedge$$
$$(\neg[\mu_{i1} \wedge \mu_{i2} \wedge \ldots \wedge \mu_{ci}] \rightarrow \mu_{j1} \wedge \mu_{j2} \wedge \ldots \wedge \mu_{cj}) \qquad (7.66)$$

Aceste expresii trebuie separate şi aduse la forma (7.32), şi apoi înlocuite prin inegalităţi. Toate restricţiile de acest tip trebuie analizate aşa cum s-a prezentat mai sus.

7.4.7 Transformări ale grafului reţelei Petri.

Transformările grafului reţelei Petri reprezintă o altă cale de a transforma restricţii ce conţin elemente ale vectorului de concesionare. Considerăm un sistem modelat printr-o reţea Petri care trebuie să satisfacă restricţia:

$$\mu_i + g_j \leq 1 \qquad (7.67)$$

Această relaţie specifică faptul că tranziţia t_j nu poate fi concesionată dacă locaţia p_i este marcată şi viceversa. Pentru a transforma această restricţie într-o formă în care să conţină doar elemente ale vectorului de marcaj se va proceda la o transformare a grafului reţelei. Tranziţia t_j este expandată în două tranziţii t_j şi t_j' şi o locaţie p_j' între ele aşa cum se arată în Figura 7-12.

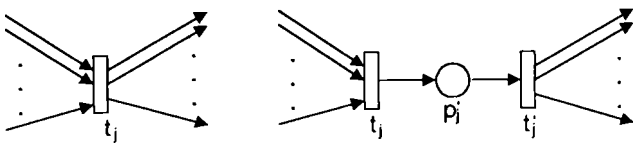


Fig. 7-12 Transformări ale grafului reţelei

Matricea de incidenţe a reţelei de bază se completează cu o linie şi o coloană, deoarece numărul total al tranziţiilor şi locaţiilor creşte cu unu. Această transformare nu modifcă nici o caracteristică a reţelei originale, scopul ei este de a monitoriza execuţia tranziţiei t_j.

După această transformare, în relaţia (7.67) g_j poate fi înlocuită prin marcajul μ_j' a locaţiei p_j' şi astfel devine de forma:

$$\mu_i + \mu_i' \leq 1 \qquad (7.68)$$

În această nouă formă restricţia conţine doar elemente ale vectorului de marcaj şi structura de conducere poate fi calculată aşa cum s-a arătat în secţiunea 7.4.1. Deoarece metoda produce o strategie de conducere, constând doar din locaţii şi arce, nu se va crea nici un arc care se leagă de locaţia p_j' creată prin transformare. După ce s-a sintetizat controlerul, cele două tranziţii şi locaţia suplimentară pot fi integrate în tranziţia t_j originală.

7.5 Exemplul unei celule flexibile de fabricaţie

Se consideră exemplul unei celule flexibile de fabricaţie constând din trei staţii de lucru: două staţii pentru prelucrarea semifabricatelor (staţia de lucru #2 şi staţia de lucru #3) şi o staţie de montaj (staţia de lucru #3). Semifabricatele respectiv produsul finit sunt transportate cu ajutoul a cinci vehicule ghidate automat (automated gided vehicules – AGVs). Modelul de reţea Petri al sistemului se prezintă în Figura 7-12.

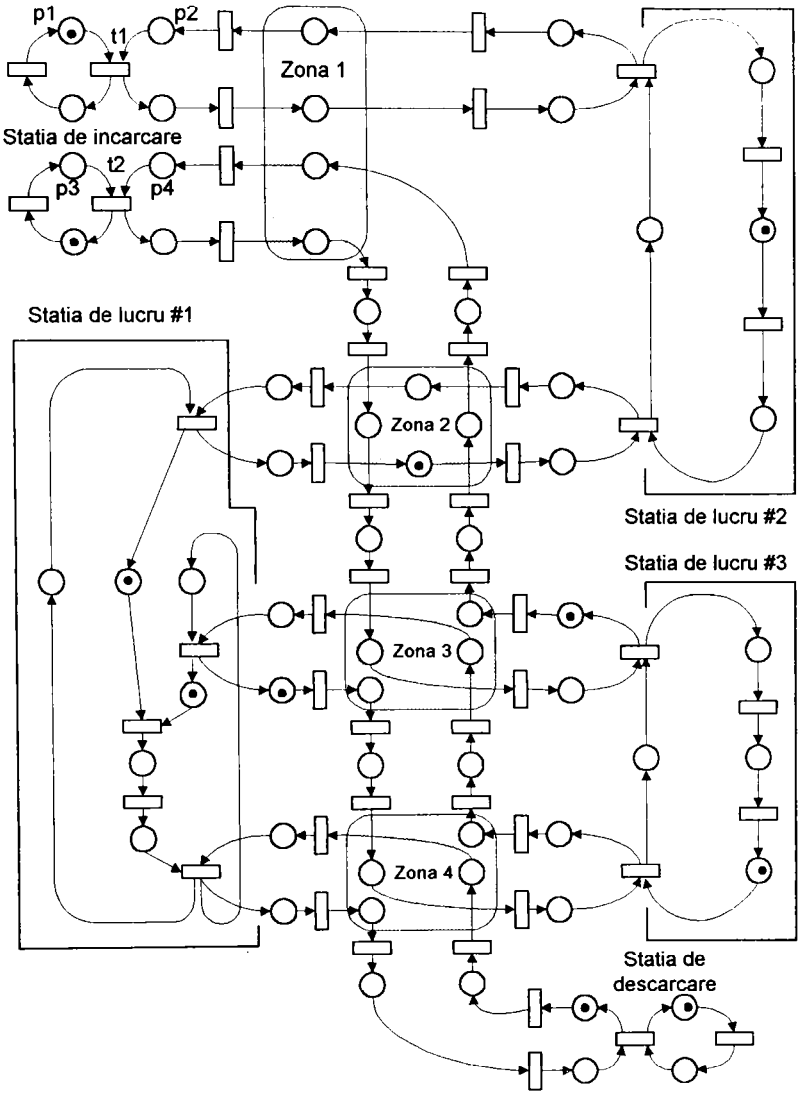


Fig. 7-13 Modelul de reţea Petri al celulei flexibile de fabricaţie

Vehiculele şi prefabricatele se reprezintă prin jetoane şi în acest fel marcajul reţelei Petri corespunde stării actuale a sistemului. Zonele haşurate reprezintă zonele în care se pot întâlni vehicule. Nu este permis ca două sau mai multe vehicule să se întâlnească în aceeaşi zonă. Această interdicţie reprezintă restricţii asupra stărilor realizabile ale sistemului respectiv asupra marcajelor accesibile ale modelului de reţea Petri, restricţii a căror realizare cade în sarcina sistemului de conducere. Pe baza tehnicilor prezentate mai sus se poate ajunge la formalizarea acestor restricţii.

Restricţiile care specifică interdicţia asupra prezenţei a două sau mai multe vehicule în zonele periculoase se exprimă prin următoarele inegalităţi:

$$\sum_{i\in Z_1}\mu_i \le 1, \quad \sum_{i\in Z_2}\mu_i \le 1, \quad \sum_{i\in Z_3}\mu_i \le 1, \quad \sum_{i\in Z_4}\mu_i \le 1, \tag{7.69}$$

unde Z_j reprezintă mulţimea indicilor nodurilor care formează zona *j*. Aceste restricţii conţin doar elemente ale vectorului de marcaj şi astfel pot fi introduse variabile auxiliare iar inegalităţile de mai sus devin egalităţi, după cum urmează:

$$\sum_{i\in Z_1}\mu_i + \mu_{C1} = 1, \quad \sum_{i\in Z_2}\mu_i + \mu_{C2} = 1, \quad \sum_{i\in Z_3}\mu_i + \mu_{C3} = 1, \quad \sum_{i\in Z_4}\mu_i + \mu_{C4} = 1 \tag{7.70}$$

Cele patru variabile auxiliare pot fi interpretate ca patru locaţii suplimentare legate prin arce de tranziţii în modelul de reţea Petri al sistemului. În acest caz fiecare locaţie suplimentară controlează accesul într-unul din zonele intezise.

Restricţiile asupra accesului în zona de încărcare se specifică cu ajutorul elementelor vectorului de concesionare, prin următoarea relaţie:

$$g_1+g_2 \le 1, \tag{7.71}$$

unde t_1 şi t_2 sunt tranziţiile care modelează acţiunea de preluarea de către un vehicul ghidat automat, a unui semifabricat din staţia de încărcare 1 respectiv din staţia de încărcare 2. Expresia de mai sus conţine doar elemente ale vectorului de concesionare şi trebuie transformată conform celor prezentate la secţiunea 7.4.4.4. Cele două elemente ale vectorului de concesionare se vor înlocui prin suma marcajelor locaţiilor din premulţimea tranzţtiilor corespunzătoare, conform Figurii 7-12. Notând aceste marcaje prin μ_1, μ_2, μ_3 şi μ_4 se obţine expresia echivalentă:

$$\mu_1+\mu_2+\mu_3+\mu_4 \le 3. \tag{7.72}$$

Această condiţie va impiedica marcarea simultană a locaţiilor p_1, p_2, p_3 şi p_4. Cu ajutorul unei variabile auxiliare expresia de mai sus transformă într-o egalitate, astfel:

$$\mu_1+\mu_2+\mu_3+\mu_4+\mu_{C5} \le 3. \tag{7.73}$$

Această a cincea variabilă auxiliară poate fi interpretată ca o a cincea locaţie suplimentară legată prin arce de tranziţii în modelul de reţea Perti al sistemului, locaţie ce controlează accesul pieselor în celula flexibilă de fabricaţie.

Matricea de incidenţe N_C observatorului de evenimente (a se vedea şi Figura 7-11) se obţine pe baza ecuaţiei,

$$N_C = -A \cdot N,$$

unde N este matricea de incidenţe a modelului de reţea Petri al sistemului iar matricea A este matricea formată, conform procedeului descris în secţiunea 7.4.1, din coeficienţii relaţiilor (7.70) şi (7.73).

Marcajul iniţial al subreţelei pe baza căreia lucrează obsevatorul de evenimente se obţine prin relaţia:

$$\mathbf{c}_0 = \mathbf{b} - \mathbf{A} \cdot \mathbf{m}_0,$$

unde **b** este vectorul variabilelor auxiliare format conform secţiunii 7.4.1 iar $\mathbf{m}_0$ este marcajul iniţial al modelului de reţea Petri al sistemului.

Structura de control care realizează restricţiile impuse conform celor de mai sus se prezintă în Figura 7-11.

7.6 Controlabilitate primară

În sistemele reale funcţiile **u**[j] ce specifică comanda aplicată procesului condus nu pot fi evaluate instantaneu. Aşa cum s-a precizat în paragraful 6.7.1 sistemul de conducere are nevoie de un timp τ pentru a genera mărimile de comandă şi a le transmite procesului condus. Aceasta are ca rezultat necesitatea ca elementele vectorului de stare al procesului condus – în limbajul reţelelor Petri marcajele locaţiilor – implicate în funcţiile **u**[j] trebuie să persiste cel puţin pe durata τ.

În mod natural ne putem găsi în două situaţii:

(a) când procesul procesul condus satisface această necesitate, fiind desemnat în continuare ***„ τ-liber primar controlabil“,*** şi

(b) când procesul nu satisface de la sine această necesitate.

În acest al doilea caz sunt, deasemenea posibile două variante, după cum procesului i se poate induce, prin comandă, proprietatea de a respecta condiţia necesară, spunându-se că este ***„ τ-forţat primar controlabil“,*** sau după cum procesului nu i se poate induce proprietatea necesară spunîndu-se că procesul nu este controlabil.

Posibilitatea de a controla procesul condus în accepţiunea mai sus prezentată este de tip *primar* întrucât ea nu garantează reuşita elaborării unei strategii de conducere în sens clasic ci doar condiţia primară pe baza căreia se poate pune problema elaborării unei astfel de strategii. În acest sens vorbim despre *controlabilitatea primară a procesului modelat prin reţele Petri* şi putem enunţa următoare condiţie de controlabilitate primară:

> *Un proces este controlabil pe baza modelului de reţea Petri dacă pentru orice comandă* **u**[j]=U(***x***) *starea* ***x*** *se conservă pe durata* τ *de generare a comenzii ca urmare a faptului că procesul este τ-liber primar controlabil sau a faptului că procesul este τ-forţat primar controlabil*

Condiţia exprimată mai sus este o condiţie suficientă. Ea este implicit satisfăcută în cazul proceselor a căror modele de reţea Petri conţin doar tranziţii controlabile.

Pentru cazu în care procesul este *τ-forţat primar controlabil* condiţia de controlabilitate exprimată mai sus implică ca o submulţime $T_{CX} \subseteq T$ a mulţimii tranziţiilor să fie controlabilă. Această submulţime este determinată de strategia de comandă concretă adoptată. Determinarea ei implică o analiză suplimentară a reţelei.

Cardinalul mulţimii T_{CX} depinde de sistemul de conducere (prin valoarea duratei τ) şi de caracteristicile procesului condus (durata de persistenţă implicită a marcajelor datorată inerţiilor inerente din procesul condus). Pentru a determina submulţimea T_{CX}, în modelul de reţea Petri locaţiile trebuie să fie caracterizate şi din punctul de vedere al persistenţei marcajelor. Trebuie precizat aici că această caracterizare nu este identică cu cea utilizată în cazul reţelelor Petri temporale. În cazul analizei de controlabilitate caracterizarea poate să fie binară şi relativă la timpul de reacţie al sistemului de conducere. Caracterizarea poate fi

făcută printr-un predicat care exprimă dacă o anumită locaţie este sau nu este persistentă relativ la timpul de reacţie al sistemului de conducere dat.

Metodele de corecţie sau restricţie prezentate sunt valabile numai în conjuncţie cu o analiză de controlabilitate bazată pe cele prezentate mai sus.

Controlabilitatea în sensul clasic, relativ la comportamentul dorit al sistemului, poate fi asigurată doar dacă se asigură controlabilitatea primară exprimată mai sus. Semnificativă în acest sens este Figura 7-11 care scoate în evidenţă faptul că, chiar dacă s-a reuşit modelarea print-o reţea Petri a comportamentului dorit al sistemului, implementarea efectivă a restricţiilor necesare trebuie realizată printr-un sistem de conducere care la rândul lui trebuie să satisfacă *condiţia de controlabilitate primară.*

7.7 Concluzii

S-au definit componentele de sincronizare, de temporizare şi anticipativă ale vectorului de comandă iar pe baza acestora, expresia pentru determinarea vectorului de comandă. Această structurare este utilă deoarece fiecare dintre aceste componente presupune metode proprii de specificare, analiză şi implementare.

S-a prezentat o metodă simplă şi elegantă de sinteză a componentei anticipative a vectorului de comandă. Metoda se bazează pe conceptul de maşină virtuală de reţea Petri. Sinteza se realizează pe baza restricţiilor asupra stărilor procesului condus. Dispunând de o implementare a maşinii virtuale de reţea Petri, implementarea metodei este deosebit de simplă deoarece aceasta este de fapt o restricţie a implementării iniţiale. Modulul de control necesar este deasemenea uşor de implementat şi în acelaş timp, după o parametrizare adecvată, este aplicabil tuturor restricţiilor.

S-a prezentat o metodă de sinteză a componentei de sincronizare a vectorului de comandă bazată pe restricţii ce se formulează prin inegalităţi liniare compuse din elementele vectorului de marcaj şi a unor constante şi variabile întregi. S-a arătat că este posibilă transformarea în astfel de inegalităţi şi a altor forme de prezentare a restricţiilor ca de exemplu a celor formulate prin expresii Booleene sau a celor formulate cu elemente ale vectorului de concesionare. Metoda prezentată propune o implementare, realizabilă pe baza conceptului de reţea Petri, care lasă intact modelul de reţea Petri al procesului condus şi care are ca rezultat un modul ce implementează însăşi strategia de conducere.

Oricare ar fi strategia de comandă care asigură comportamentul dorit sistemul de automatizare trebuie să satisfacă condiţia de controlabilitate primară exprimată la punctul 7.6.

7.7.1 CONSIDERAŢII BIBLIOGRAFICE

1.

- În lucrarea [HZG'96] respectiv în lucrarea precedentă acesteia (Bruce H. Krogh and Lawrence E. Holloway: *Synthesis of Feedback Control Logic for Discrete Manufacturing Systems*, Automatica, Vol.27, No.4, pp.641-651, 1991) este tratată problematica restricţionării stărilor accesibile presupunând restricţii specificate prin mulţimi de stări $F \subseteq P$ ce satisfac condiţia de restricţie specificată astfel: $M_F = \{ m \mid m(p) \geq 1, \forall p \in P \}$.

  Pentru această categorie de restricţii se prezintă o metodă de determinare a comenzilor celor mai permisive. Analiza "off-line" respectiv "on-line" este şi în acest caz prezentă. În aceste lucrări scopul analizei "off-line" este de a determina un număr de predicate prin

evaluarea cărora poate fi anticipată evoluţia marcajelor locaţiilor condiţionate prin restricţia dată. Aceste predicate sunt evaluate în cadrul analizei "on-line". Informaţia încorporată în modelul de reţea Petri al procesului este utilizată doar în cadrul analizei "off-line".

- Spre deosebire de cele de mai sus, metoda prezentată în această lucrare nu impune o formă specifică de descriere a restricţiilor. Analiza "off-line" are drept scop determinarea subreţelei controlabile afectate de restricţia dată. Analiza "on-line" se realizează pe baza unei metode ce utilizează informaţia încorporată în modelul de reţea Petri al procesului condus, metodă bazată pe mecanismul maşinii virtuale de reţea Petri, care, odată implementată, poate fi reutilizată pentru analiza "on-line". Nu este necesară specificarea şi mai ales implementarea unui număr mare de predicate. Proiectantul se poate concentra asupra problemei de automatizare – specificarea restricţiilor.

2.

- Problema restricţionării stărilor accesibile ale unei reţele Petri este tratată şi în lucrarea [GDi'92] prin definirea unui supervizor. Un supervizor este considerat ca fiind un agent care acţionează asupra tranziţiilor controlabile ale unui sistem, cu scopul de a restricţiona stările accesibile ale acestuia astfel încât să rezulte un comportament dorit.

  Ca şi în cazul lucrării de faţă şi în această lucrare supervizorul este un observator al secvenţelor de evenimente generate de către sistemul controlat. Mărimile de comandă se consideră a fi tranziţiile concesionate ale supervizorului. Ca urmare a sintezei supervizorului se ajunge la o nouă structură de reţea obţinută prin amendarea structurală a reţelei iniţiale. Lucrarea precizează condiţiile în care o astfel de sinteză este posibilă.

- Aceeaşi problematică a fost tratată şi în cadrul prezentei lucrări, într-o manieră practică şi cu o altă terminologie, în capitolele 5 şi 7. În capitolul 5 s-a prezentat o metodă testare a corectabilităţii reţelei respectiv s-a prezentat o metodă de restricţionare a stărilor accesibile prin bucle autonome. În capitolul 7 s-a prezentat o metodă de restricţionare a stărilor accesibile prin locaţii şi arce suplimentare.

  Astfel, se propune o modalitate practică de a utiliza structura de supervizor obţinută având în vedere faptul că în în sisteme reale nu există un observator ideal şi se tratatează aspectele practice legate de preluarea evenimentelor generate în sistemul condus precum şi cele legate de generarea semnalelor de comadă.

  Prin structura de conducere prezentată în Figura 7-10 lucrarea prezintă o soluţie practică ce exploatează rezultatele teoretice legate de supervizare.

* * *

Faţă de literatura de specialitate la care autorul a avut acces, acest capitol prezintă următoarele elemente originale:

a) Definirea componentelor vectorului de comandă.

b) Introducerea noţiunii de domeniu de precedenţă şi elaborarea metodei de determinare a acestuia.

c) Definirea funcţiilor care realizează legătura dintre reţeaua de bază şi subreţelele induse prin restricţii.

d) Elaborarea unei metode generale pentru determinarea componentei anticipative a vectorului de comandă, pe baza restricţiilor impuse reţelei.

e) Definirea structurii de conducere pentru metoda de determinare a componentei de sincronizare, bazată pe invarianţi.

f) Exprimarea condiției de controlabilitate primară.

8. Controlul distribuit pe baza modelului de reţea Petri

Acest capitol prezintă o metodă de partiţionare a reţelelor Petri şi o definiţie a interfeţelor dintre subreţelele rezultate care permite, pe baza conceptului de maşină virtuală de reţea Petri, specificarea structurilor de conducere a subreţelelor rezultate.

8.1 Preliminarii

În cazul sistemelor mari, complexe poate să apară necesitatea divizării sistemelor mari pe mai multe subsisteme şi a alocării controlului acestora mai multor procesoare sau Echipamente de conducere. Dacă sistemul s-a modelat printr-o reţea Petri aceasta trebuie subdivizată astfel încât diverselor subsisteme să le corespundă subreţele ale reţelei iniţiale, de bază. Fiecare echipament de conducere va lucra pe baza unei astfel de subreţele.

Divizarea se realizează prin partiţionare. Scopul partiţionării este de a asigura controlabilitatea sistemului modelat prin reţeaua Petri. Subreţelele rezultate au legături între ele realizate prin interfeţe. Posibilitatea controlului precum şi caracteristicile echipamentelor prin care se realizează sunt determinate în mod direct de caracteristicile metodei de partiţionare şi a celei de definire a interfeţelor.

8.2 Partiţionarea reţelelor Petri

Înainte de a defini o partiţie a unei reţele Petri se defineşte noţiunea de subreţea, astfel:

Definiţia 8-1 (Subreţea)

O reţea $N_i=(P_i,T_i;F_i,\mathbf{m}_{0i})$ este o subreţea a reţelei $N=(P,T;F,\mathbf{m}_0)$ dacă:

(i) Mulţimile de locaţii, P_i, şi tranziţii, T_i, ale subreţelei N_i sunt submulţimi ale mulţimii locaţiilor respectiv tranziţiilor reţelei N:

$$(P_i \subseteq P) \wedge (T_i \subseteq T)$$

(ii) Mulţimile arcelor subreţelelor N_i sunt restricţii la mulţimile P_i şi T_i:

$$F_i = F \cap ((T_i \times P_i) \cup (P_i \times T_i))$$

(iii) Marcarea iniţială a reţelei N_i este o restricţie la P_i a marcării iniţiale a reţelei N:

$$\forall p \in P,\ \mathbf{m}_{0i}(p) = \mathbf{m}_0(p).$$

Conform definiţiei de mai sus subreţelele pot fi disjuncte sau nu.

Subreţelele definite mai sus dau posibilitatea definiţiei unei partiţii a reţelei.

Definiţia 8-2 (Partiţie de reţea)

O partiţie Q a reţelei N este o colecţie de subreţele ale reţelei N: $Q=\{N_1, N_2, ..., N_n\}$, astfel încât subreţelele să acopere toată reţeaua N, adică:

$$\bigcup_{i=1}^{n} P_i = P$$

$$\bigcup_{i=1}^{n} T_i = T$$

$$\bigcup_{i=1}^{n} F_i = F$$

Partiţia realizată trebuie să asigure şi comunicaţia dintre subreţele. Comunicaţia trebuie realizată prin elemente structurale de reţea Petri astfel încât funcţionarea în ansamblu a partiţiei realizate să corespundă în totalitate funcţionării reţelei iniţiale.

În lucrarea de faţă se propune o soluţie care rezolvă comunicaţia dintre subreţele prin specificarea unei partiţii ce alocă subreţelelor domenii de reţea comune în reţeaua partiţionată. Aceste domenii se vor numi domenii de interfaţare.

Definiţia 8-3 (Domeniul de interfaţare)

Domeniul de interfaţare dintre două subreţele N_i şi N_j este o subreţea N_{ij} comună celor două reţele, astfel încât:

$N_{ij}=(P_{ij},T_{ij};F_{ij},\mathbf{m}_{0ij})$

$P_{ij}=P_i \cap P_j$;

$T_{ij}=T_i \cap T_j$;

$F_{ij}=F_i \cap F_j$;

$\forall p \in P_{ij}$, $\mathbf{m}_{0ij}(p)=\mathbf{m}_0(p)$.

O anumită subreţea, N_i , poate să aibă domenii de interfaţare cu mai multe alte subreţele. Din acest motiv, relativ la această subreţea, trebuie considerată o mulţime a domeniilor de interfaţare.

Definiţia 8-4 (Mulţimea domeniilor de interfaţare)

Mulţimea domeniilor de interfaţare a subreţelei N_i este reuniunea tuturor domeniilor de interfaţare şi se exprimă prin relaţia:

$$S_i = \bigcup_{i \neq j} N_{ij}$$

8.3 Interfaţarea subreţelelor

8.3.1 Structurarea domeniilor de interfaţare

Definiţia dată mai sus domeniilor de interfaţare nu le concretizează în sensul că nu le specifică structura sau numărul de noduri. Pentru a putea implementa structuri de conducere distribuite, bazate pe o anumită partiţie dată trebuie definite interfeţe care să permită o comunicaţie eficientă între subreţele şi care să se alinieze conceptelor de bază ale reţelelor Petri.

În primul rând trebuie specificată o structură concretă pentru domeniile de interfaţare. În acest sens, trebuie specificat, în primul rând o regulă de delimitare a subreţelelor. În principiu este posibilă o delimitare la oricare element de reţea: tranziţie, locaţie sau arc de reţea. În această lucrare se propune ca delimitarea să se realizeze la nivelul tranziţiilor. Tranziţiile care delimitează o anumită subreţea se vor numi tranziţii de graniţă şi se definesc astfel:

Definiţia 8-5 (Tranziţie de graniţă)

La partiţionarea unei reţele Petri, N, tranziţia $t_j \in N$, este o tranziţie de graniţă a subreţelei N_i dacă:

$(\exists p \in {}^\bullet t_j \subset N_i \text{ a.î. } p \notin N_i) \vee (\exists p \in t_j^\bullet \subset N_i \text{ a.î. } p \notin N_i)$

Definiţia 8-5 exprimă faptul că, dacă o tranziţie $t_j \in N$ este o tranziţie de graniţă a subreţelei N_i, atunci există cel puţin o locaţie a premulţimii sau postmulţimii ei care nu aparţine subreţelei N_i.

8.3.2 Delimitarea domeniilor de interfaţare

Domeniile de interfaţare se delimitează prin următoarea *regulă de delimitare a domeniilor de interfaţare*:

> Oricare ar fi tranziţia de graniţă $t_m \in N_i$ şi tranziţia de graniţă $t_n \in N_j$, astfel încât t_m, $t_n \in N_{ij}$, între aceste două tranziţii nu poate exista nici o altă tranziţie.

Cu alte cuvinte, elementele structurale de bază ale domeniilor de interfaţare vor fi triplete de forma (t_m, p_q, t_n).

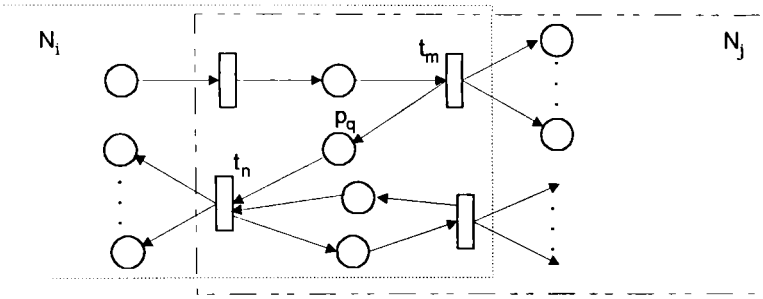


Fig. 8-1 Delimitarea domeniilor de interfaţare

Figura 8-1 ilusrează această regulă de delimitare a domeniilor de interfaţare. Tripletele (t_m, p_q, t_n) sunt orientate prin orientarea arcelor reţelei. Această orientare poate constitui un criteriu suplimentar de subdivizare a domeniilor de interfaţare. În continuare se convine ca notaţia N_{ij} să se refere la un domeniu format din triplete orientate de la subreţeaua N_i la subreţeaua N_j şi reciproc, notaţia N_{ji} se va referi la un domeniu format din triplete orientate de la subreţeaua N_j la subreţeaua N_i. Domeniile orientate conform convenţiei de mai sus se vor numi interfeţe în reţele Petri sau pe scurt interfeţe.

Observaţie: Din definiţiile de mai sus rezultă că partiţia are sens doar dacă nu creează subreţele mai mici decât tripletul (t_m, p_q, t_n).

Acceptând regula de structurare de mai sus domeniile de interfaţare vor fi formate din triplete de forma (t_m, p_q, t_n) astfel încât: $((t_m, p_q) \in F_{ij}) \wedge ((p_q, t_n) \in F_{ij})$, unde indecşii utilizaţi sunt specificaţi conform ordonării din reţeaua de bază. Domeniile de interfaţare vor fi formate din structuri de forma celei din Figura 8-2.

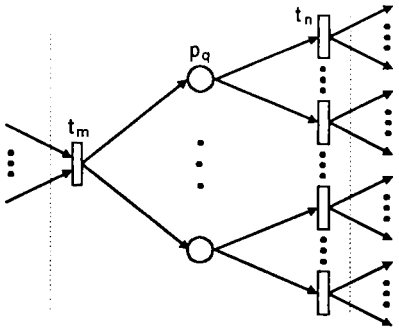


Fig. 8-2 Structuri ce formează domeniile de interfaţare

8.4 Structura de conducere pentru controlul distribuit

În cazul cel mai simplu, interfaţa dintre subreţelele N_i şi N_j este formată dintr-un singur triplet, (t_m, p_q, t_n), aşa cum se prezintă în Fig.8-3(a). Subsistemele de comandă lucrează pe baza acestor subreţele respectiv acţionează asupra subsistemelor modelate prin aceste subreţele. Subreţelele rezultate sunt prezentate în Fig.8-3(b). În cadrul acestora tranziţiile t_m şi t_n primesc câte un atribut. Atributele s-au notat aici prin câte un index superior.

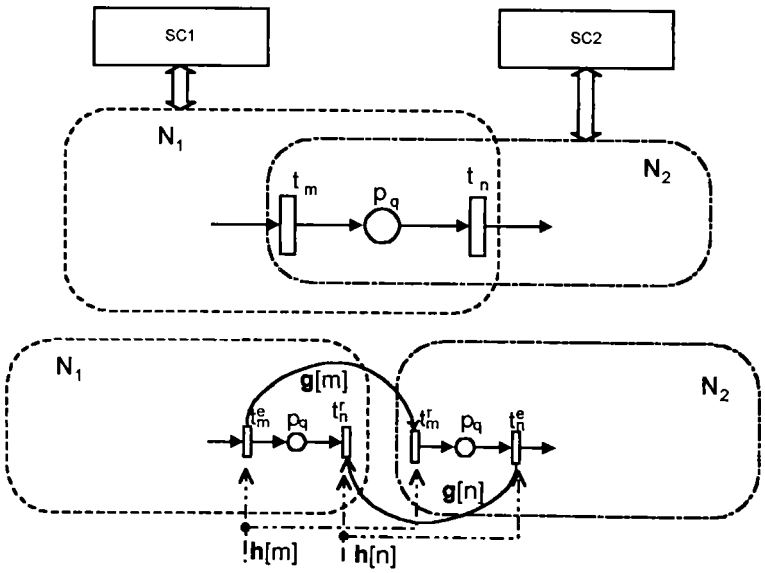


Fig. 8-3 Controlul distribuit al proceselor modelate prin reţele Petri

Dacă indexul superior este "e" acesta semnifică faptul că echipamentul de comandă care controlează subreţeaua dată emite un semnal sau mesaj, în momentul concesionării tranziţiei indexate, către echipamentul de comandă a subreţelei cu care se interfaţează. Dacă indexul superior este "r" acesta semnifică faptul că echipamentul de comandă care controlează subreţeaua dată determină momentul concesionării prin recepţia unui semnal sau mesaj de la echipamentul de comandă a subreţelei cu care se interfaţează.

Subreţeaua N_i nu conţine postmulţimea tranziţiei t_n^r şi prin urmare SC_i nu poate observa concesionarea acesteia. În mod simetric, subreţeaua N_j nu conţine premulţimea tranziţiei t_m^r şi prin urmare SC_j nu poate observa concesionarea acesteia. În schimb, N_i poate determina concesionarea tranziţiei t_m^e iar N_j pe cea a tranziţiei t_n^e. Prin urmare, fiecare SC poate furniza celeilalte informaţia necesară unei funcţionări corecte. În Fig. 8-3(b) prin g[m] s-a notat mesajul sau semnalul emis de SC_i în momentul concesionării tranziţiei t_m^e iar prin g[n] s-a notat mesajul sau semnalul emis de către SC_j la concesionarea tranziţiei t_n^e. Dacă se presupune că prin sistemul de comunicaţii predicatele tranziţiilor **h**[m] şi **h**[n] sunt disponibile fiecărui echipament de conducere, acesta poate lucra corect, aşa cum s-a prezentat în capitolul 6.

Funcţionarea va fi corectă dacă mesajul sau semnalul emis în momentul concesionării este recepţionat înainte de a recepţiona semnalul sau mesajul corespunzător predicatului tranziţiei (Figura 8-4). Aceasta deoarece numai în acest caz, la nivelul subreţelei care recepţionează aceste semnale sau mesaje, este observată în mod corect relaţia cauză→efect, în cazul de faţă concesionare→execuţie.

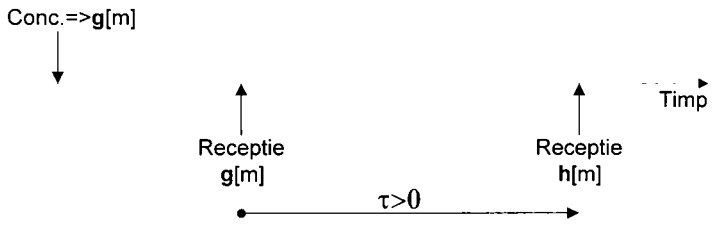


Fig. 8-4 Condiţia necesară în comunicaţia elementelor g[m] *şi* **h**[m]

În acest fel dinamica sistemului controlat determină în mod direct viteza de comunicaţie necesară. Această condiţie trebuie să fie satisfăcută pentru fiecare triplet (t_m, p_q, t_n) în parte.

În cazul general o interfaţă conţine mai multe triplete respectiv o subreţea este interfaţată cu mai multe alte subreţele. Partiţia realizată trebuie să specifice corespondenţa dintre tranziţiile interfeţelor şi tranziţiile reţelei de bază. În acest fel comunicaţia are loc cu referire la ordonarea tranziţiilor în reţeaua de bază.

Notaţiile de până acum au făcut referire directă la ordonarea tranziţiilor în reţeaua de bază N.

Deoarece fiecare subreţea se va trata separat acest lucru se va reflecta într-o ordonare proprie a tranziţiilor şi locaţiilor. În acest caz partiţia realizată trebuie să specifice aplicaţii care fac legătura cu ordonarea din reţeaua de bază N.

$\Pi_i : P_i \rightarrow P$

$\theta_i : T_i \rightarrow T$

Având în vedere structurarea introdusă, tranziţiile subreţelei N_i sunt de trei categorii, astfel încât $T_i = T_{ai} \cup T_{ei} \cup T_{ri}$, unde

T_{ai} -este mulţimea tranziţiilor ce nu participă la interfaţare;

T_{ei} -este mulţimea tranziţiilor la concesionarea cărora se emite un semnal sau un mesaj;

T_n -este mulţimea tranziţiilor a căror concesionare este observată prin recepţia unui mesaj sau semnal.

În structura de comandă prezentată în Fig. 8-5 acest lucru se reflectă în componentele vectorului de concesionare, astfel:

1. Componenta g_a a vectorului de concesionare corespunde submulţimii T_a a tranziţiilor;
2. Componenta g_e a vectorului de concesionare corespunde submulţimii T_e a tranziţiilor;
3. Componenta g_r a vectorului de concesionare corexpunde mulţimii Tr a tranziţiilor.

Primele două componente pot fi determinate de către maşina virtuală a subreţelei N_i. Componenta g_r se recepţionează de la alte echipamente de conducere. Prin agregare se obţine un vector de concesionare complet pe baza căruia poate fi determinată starea subreţelei respectiv poate fi realizată o reacţie după stare.

Figura 8-5 prezintă structura de conducere a unei subreţele. Maşina virtuală de reţea Petri se implementează pe baza structurii subreţelei respectiv a matricii ei de incidenţe N_i.

Dacă se respectă condiţia exprimată prin Figura 8-4, structura de conducere prezentată în Figura 8-5 va lucra corect, în concordanţă cu reţeaua iniţială. Această structură constituie elementul de distribuţie al sistemului care controlează reţeaua iniţială. Pe baza acesteia pot fi implementate echipamentele de comandă ale subreţelelor. Comunicaţia dintre acestea se poate realiza prin semnale sau mesaje aşa cum s-a prezentat în capitolul 6.

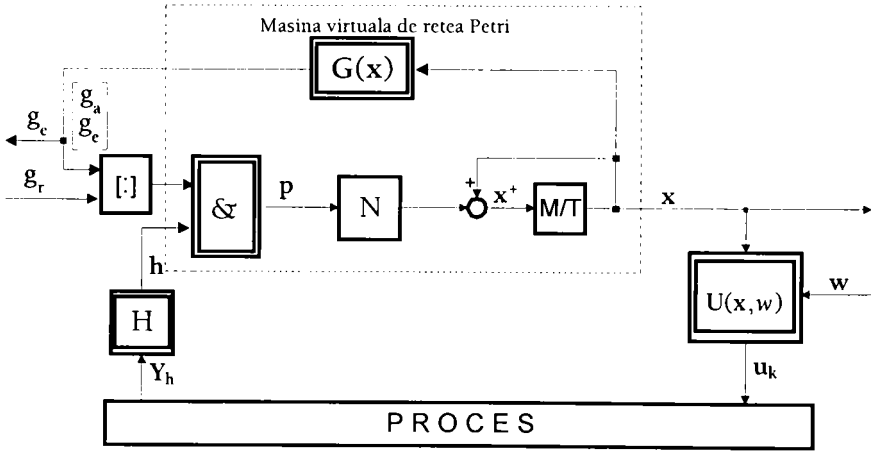


Fig. 8-5 Structra de conducere pentru controlul distribuit

În Figura 8-5 modulul notat prin "[:]" realizează operaţia de agregare a vectorilor de intrare.

În capitolul 10 se prezintă un exemplu pentru controlul distribuit al unei linii de asamblare. În cazul acestui exemplu s-a aplicat prezenta metodă de control distribuit. În paragraful 10.5.3 se tratează problematica implementării echipamentului de comandă. Prin Fig. 10-18 se definesc domeniile de interfaţare a staţiei de montaj cu modulele de transport cu care aceasta comunică.

8.5 Concluzii

S-a prezentat o metodă de partiționare a rețelei Petri și o definiție a interfețelor dintre subsistemele rezultate care permite, pe baza conceptului de mașină virtuală de rețea Petri, definirea structurilor de conducere a subrețelelor rezultate. Acestea constituie elementele de distribuire în sistemul de conducere a rețelei Petri de bază. Structurile de conducere sunt complet definite astfel încât prin integrarea lor se obține un comportament echivalent cu un sistem de conducere ce ar implementa întreaga rețea. Integrarea se realizează prin sistemul de comunicații care trebuie să respecte condiția formulată prin Figura 8-4.

* * *

Față de literatura de specialitate la care autorul a avut acces, acest capitol prezintă următoarele elemente originale:

a) Definirea domeniilor de interfațare într-o concepție proprie. În [DAL'93] se definește un domeniu de interfațare constituit dintr-un singur nod de rețea (locație sau tranziție). Acest singur nod nu aduce informație suficientă pentru a putea implementa un sistem de comandă distribuit ce lucrează strict pe baza relațiilor cauzale din procesul modelat și din acest motiv se recurge la tehnici speciale în scopul administrării timpului și sincronizării ceasurilor.

b) Definirea structurii de conducere pentru controlul distribuit. În literatura de psecialitate la care a avut acces, autorul nu a găsit definită vre-o structură de conducere general aplicabilă în controlul distribuit al sistemelor.

c) Specificarea condițiilor ce se impun sistemului de comunicații, în privința vitezei, pentru ca sistemul de control distribuit să fie corect implementabil.

9. Structurarea ierarhică a sistemelor de conducere pe baza modelului de reţea Petri

Acest capitol prezintă o sinteză a procedeelor de structurare ierarhică a reţelelor Petri şi propune o nouă metodă de reprezentare a subreţelelor în reţeaua de bază. Această metodă prezintă unele avantaje, faţă de alte metode cunoscute, avantaje ce fac posibilă definirea structurii de reţea pe baza căreia, cu ajutorul conceptului de maşină virtuală de reţea Petri, poate fi sintetizată o structură de conducere necesară pentru implementarea echipamentului de conducere a subreţelei.

9.1 Organizarea ierarhică a reţelelor Petri

9.1.1 Consideraţii preliminare

Lucrul cu reţele Petri devine dificil o dată cu creşterea complexităţii acestora. Reprezentarea grafică a unei reţele complexe poate ocupă o "suprafaţă" atât de mare încât să devină practic inutilizabilă. Numărul elementelor mulţimii realizabile creşte exponenţial cu numărul de noduri din reţea. În consecinţă, conform [Sci'92], pentru a testa realizabilitatea unui marcaj, necesarul de memorie, şi în mod implicit necesarul de timp de calcul, creşte exponenţial cu mărimea reţelei. Din acest motiv, peste un anumit număr de noduri, atât reprezentarea cât şi analiza reţelei devin practic nerealizabile. Se impune divizarea pe subreţele.

Divizarea poate fi realizată atât prin procedeul "top-down" cât şi prin procedeul "bottom-up". În situaţia în care divizarea se realizează pe mai multe niveluri, se ajunge la o structură piramidală, adică la o organizare ierarhică a reţelei, organizare care permite stăpânirea complexităţii acesteia. Într-un caz ideal, aşa cum se arată în paragraful următor, subsistemele pot fi reprezentate prin câte un nod al reţelei supraordonate.

Oricare ar fi procedeul de divizare, subreţelele ce se realizează trebuie să fie "încapsulate" astfel încât să fie posibilă o analiză corectă a reţelei ce le cuprinde, fără a uza de informaţii asupra structurii interne sau comportamentului dinamic al subreţelelor. Trebuie definite structuri de subreţele care să fie integrabile în reţeaua de bază dar care, la rândul lor, să poată integra alte subreţele.

9.1.2 Interfaţa dintre reţeaua de bază şi subreţea

Cea mai simplă modalitate de reprezentare a subreţelelor într-o reţea de bază este reprezentarea fiecărei subreţele printr-un nod - tranziţie sau locaţie - a reţelei de bază. Se vor analiza, în continuare, condiţiile pe care trebuie să le îndeplinească subreţelele pentru ca reprezentarea prin noduri să fie posibilă respectiv se va analiza puterea de modelare a subreţelelor care satisfac condiţiile impuse unor subreţele ce pot fi reprezentate prin noduri. În acest sens tipul nodului nu este semnificativ. Tipul va determina doar felul nodului prin care subreţeaua se integrează în reţeaua de bază.

Cele mai cunoscute definiţii de subreţele se bazează pe forma "bloc". În cazul formei "bloc" fiecare subreţea comunică cu reţeaua de bază printr-o singură tranziţie de intrare t_{in} respectiv o singură tranziţie de ieşire t_{out} (Figura 9-1). În reţeaua de bază N din Figura 9-1(a), tranziţia t_v reprezintă o subreţea, N', Figura 9-1(b).

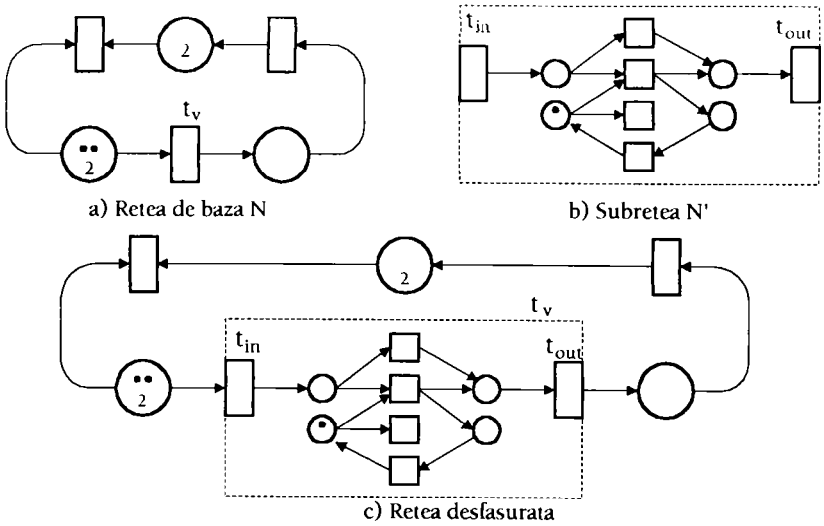


Fig. 9-1 Reprezentarea unei subrețele printr-o tranziție

Toate arcele, din rețeaua de bază, orientate către t_v, se leagă de tranziția t_{in} a subrețelei, repectiv toate arcele, din rețeaua de bază, care pleacă de la tranziția t_v, se leagă de tranziția t_{out} a subrețelei: ${}^\bullet t_v \equiv {}^\bullet t_{in} \wedge t_v^\bullet \equiv t_{out}^\bullet$. În continuare se va numi rețea de bază rețeaua care conține tranziția/tranzițiile care reprezintă subrețele. Rețeaua ce se obține din rețeaua de bază după integrarea subrețelelor în locul tranzițiilor care le reprezintă se va numi rețea desfășurată .

9.1.3 Conservarea proprietăților rețelei de bază

Pentru a păstra proprietățile dinamice ale rețelei de bază subrețelele componente trebuie să fie realizate în așa fel încât acestea să poată simula toate execuțiile tranziției t_v. Această cerință minimală se realizează dacă sunt satisfăcute următoarele două condiții [Sei'92]:

(i) Dacă t_v este concesionată la un anumit marcaj al rețelei de bază atunci tranziția t_{in} din rețeaua desfășurată trebuie să fie concesionată la același marcarj al rețelei de bază astfel încât simularea execuției tranziției t_v să poată fi începută. (9-1)

(ii) La o execuție a tranziției t_{in} trebuie să corespundă exact o execuție a tranziției t_{out}, astfel încât o simulare a execuției tranziției t_v să fie întotdeauna încheiată la nivelul subrețelei. (9-2)

La trecerea de la rețeaua de bază la rețeaua desfășurată, proprietățile dinamice ale rețelei de bază se mențin doar în situația în care după încheierea tuturor execuțiilor simulate ale tranziției t_v (adică, după ce plecând de la marcajul inițial, t_{in} și t_{out} au avut un număr egal de execuții) se realizează doar acele marcaje ale locațiilor rețelei de bază care sunt prezente și în rețeaua desfășurată. În [Sei'92] se demonstrează că, în cazul regulei de execuție slabe, condiția de mai sus este îndeplinită de subrețelele care satisfac condițiile (9-1) și (9-2).

În cazul regulei de execuție tari (stricte) proprietățile dinamice ale rețelei de bază nu pot fi păstrate doar prin satisfacerea condițiilor (9-1) și (9-2). Aceste condiții sunt insuficiente

deoarece în acest caz concesionabilitatea tranziției t_{out} în rețeaua desfășurată depinde nu numai de marcajul subrețelei ci și de marcajul rețelei de bază. Din acest motiv se poate ajunge ca după execuția tranziției t_{in} rețeaua desfășurată să evolueze astfel încât tranziția t_{out} să nu mai fie niciodată concesionată și drept urmare simularea tranziței t_v să nu se mai încheie.

Se poate arăta, [Sei91], că această problemă se rezolvă prin complementarea locațiilor din premulțimea tranziției de intrare respectiv a celor din postmulțimea tranziției de ieșire.

În literatura de specialitate se prezintă mai multe definții asupra criteriilor de structurare a subrețelelor.

9.1.4 Structurarea subrețelelor după Valette

În cazul criteriilor de structurare propuse de Valette, următoarele două cerințe structurale sunt impuse subrețelei:

(i) subrețeaua trebuie să respecte forma bloc;

(ii) subrețeaua, completată printr-o locație de extensie p_e conform Figurii 9-2, trebuie să satisfacă proprietățile dinamice prezentate mai jos.

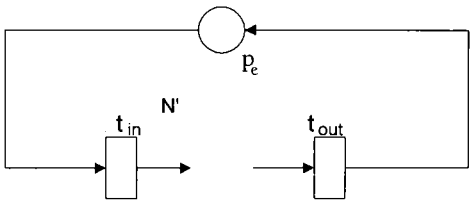


Fig. 9-2 Rețea extinsă pentru structurarea subrețelelor după Valette

Locația de extensie p_e se leagă printr-un singur arc de tranziția de intrare t_{in} respectiv tranziția de ieșire t_{out} și, în plus, în starea inițială locația p_e este marcată printr-un singur marcaj. Locația de extesie nu aparține subrețelei. Ea a fost introdusă în sprijinul analizei și poate fi privită ca o substituție a rețelei de bază. Din acest motiv în rețeaua de bază tranziția de intrare va fi concesionată cel mult o dată, așa cum se întâmplă în rețeaua extinsă din Figura 9-2. Rețeaua extinsă trebuie să satisfacă următoarele condiții:

(i) rețeaua extinsă trebuie să fie viabilă;

(ii) locația de extensie p_e trebuie să fie marcată doar la marcajul inițial;

(iii) tranziția de intrare trebuie să fie concesionată doar la marcajul inițial.

Un bloc N' care satisface condițiile de mai sus se numește *bloc bine format*. Datorită condiției (ii) un bloc bine format este mărginit. Deoarece prin execuția tranziției t_{out} locația de extensie primește un singur marcaj și deoarece locația de extensie, conform definiției, este marcată doar la marcajul inițial, execuția tranziției t_{out} reproduce marcajul inițial. Din acest motiv rețeaua care înglobează un bloc bine format este reversibilă. Blocul bine format N' s-a definit cu scopul de a transmite proprietățile de viabilitate și mărginire ale rețelei de bază N și rețelei desfășurate N". În acest fel, dacă aceste proprietăți sunt prezente în rețeaua de bază, demonstrate prin analiza rețelei, ele vor fi prezente și în rețeaua desfășurată.

Condițiile din definiția blocului bine format restrâng puterea de modelare a acestuia în comparație cu o rețea Petri generală. Spre exemplu, se exclude posibilitatea de a modela diferite procese în funcție de numărul execuției, deoarece după fiecare execuție trebuie

reprodus marcajul iniţial. De asemenea, este exclusă concesionarea multiplă, adică după realizarea unei concesionări, o nouă concesionare este posibilă doar după execuţia tranziţiei t_{out}.

Concesionabilitatea multiplă a subreţelei prezintă interes deoarece de cele mai multe ori aceasta conţine mai multe locaţii care pot reprezenta şi elemente de acumulare. Problematica concesinabilităţii multiple a fost tratată de Suzuki şi Murata.

9.1.5 Structurarea subreţelelor după Suzuki - Murata

Suzuki şi Murata definesc subreţele în forma bloc multiplu concesionabilă. O subreţea în forma bloc de k ori concesionabilă se defineşte astfel:

Definiţia 9-1 (Concesionabilitate multiplă)

Fie N o mPTN şi R_N mulţimea ei de accesibilitate. O tranzitie $t_j \in T$ este de k ori concesionabilă în N dacă există un marcaj la care t_j poate fi executată succesiv de k ori adică dacă:

$$\exists \mathbf{m} \in R_N(\mathbf{m}_0): -k \cdot \mathbf{t}_j^- \le \mathbf{x} \le \mathbf{k} - k \cdot (\mathbf{t}_j^+ + \mathbf{t}_j^-), \text{ unde } k \in \mathbf{N}^+$$

Locaţia de extensie p_e, în concordanţă cu concesionabilitatea muliplă, este marcată de k ori la marcajul iniţil. Suzuki-Murata nu impun reţelei extinse condiţiile de viabilitate şi mărginire. Acest lucru are efect asupra reţelei desfăşurate. În general, un bloc de k ori concesionabil nu transmite reţelei desfăşurate proprietăţile reţelei de bază. În schimb întotdeauna realizează funcţionalitatea unei tranziţii în reţeau de bază.

O subreţea în forma bloc este de k-bine-formată dacă îndeplineşte următoarele condiţii:

(i) tranziţia de intrare t_{in} este viabilă în reţeaua desfăşurată;

(ii) pentru fiecare secvenţă de tranziţii din reţeaua desfăşurată care conţine pe t_{in} de mai multe ori decât pe t_{out} există o secvenţă de tranziţii care conţine pe t_{out} de mai multe ori decât pe t_{in} astfel încât după aplicarea tuturor secvenţelor de tranziţii t_{in} şi t_{out} sunt executate de acelaşi număr de ori;

(iii) în reţeaua desfăşurată nu există nici o secvenţă de tranziţii aplicabilă din marcajul iniţial care să conţină pe t_{out} de mai multe ori decât pe t_{in}.

Pentru transmiterea proprietăţilor reţelei de bază Suzuki-Murata formulează condiţii suplimentare, specifice proprietăţii ce se doreşte a fi transmisă. Condiţiile formulate de Suzuki-Murata sunt mai puţin restrictive decât cele formulate de Valette. În schimb formularea acestor condiţii nefiind universală (este dependentă de cazul concret) conduce la complicaţii şi la pierderea transparenţei.

9.1.6 Structurarea subreţelelor după Abel

În [Abe'90] se propune schimbarea sintaxei reţelei astfel încât după execuţia tranziţiei t_{in} se reduce capacitatea locaţiilor din domeniul anterior al tranziţiei t_{in} cu numărul de marcaje pe care le a extras tranziţia t_{in} din aceste locaţii respectiv se reduce capacitatea locaţiilor din domeniul posterior al tranziţiei t_{out} cu numărul de marcaje pe care le va introduce tranziţia t_{out}, după execuţia ei, în aceste locaţii. După execuţia tranziţiei t_{out} se reface capacitatea fiecărei locaţii. Se asigură în acest fel executabilitate tranziţiei t_{out} în reţeaua de bază.

În [Sei'91] se propune, pentru a evita schimbarea sintaxei reţelei, utilizarea locaţiilor complementare astfel încât să se producă acelaş fenomen de rezevare a capacităţilor locaţiilor din domeniul anterior al tranzţiei t_{in} respectiv al domeniului posterior al tranziţiei t_{out}. În acest caz, creşterea complexităţii reţelei este preţul plătit pentru

rezolvarea problemei. Cu toate acestea o astfel de soluție pare mai acceptabilă decât schimbarea sintaxei rețelei.

Prin această schimbare structurală Abel impune condiția ca rețeaua de bază (comportarea ei) să nu se schimbe în cadrul rețelei defășurate. Cele două procedee de mai sus satisfac o astfel de condiție. Conform acestui procedeu subrețelele trebuie să satisfacă următoarele condiții:

(i) subrețelele trebuie să corespundă formei bloc;

(ii) să existe o locație p_k complementară în raport cu restul subrețelei;

(iii) la marcajul inițial tranziția t_{in} este concesionată și este singura tranziție concesionată;

(iv) prin execuția tranziției t_{out} se realizează marcajul inițial;

(v) subrețeaua este reversibilă.

Spre deosebire de primele două procedee în acest caz nu se impun condiții de concesionabilitate în rețeaua de bază. În schimb apare o cerință structurală care impune prezența unei locații complementare față de restul subrețelei. Spre deosebire de locația de extensie de la Valette și Suzuki-Murata, locație care s-a introdus în sprijinul analizei, locația complementară este parte integrantă a subrețelei.

Datorită condiției de reversibilitate impusă subrețelei acesta este și viabilă dacă nu conține tranziții moarte. O subrețea după Abel care nu conține tranziții moarte satisface definiția subrețelelor dată de Valette.

Tabelul 9-1 prezintă o comparație a celor trei procedee, comparație ce scoate în evidență avantajele și dezavantajele acestora.

Tabelul 9.1

	Valette	**Suzuki-Murata**	**Abel**
Regula de tranziție	slabă	slabă	tare
Forma bloc impusă	da	da	da
Necesitatea analizei rețelei de bază	da	da	nu
Conservarea proprietăților rețelei de bază	da	da	da
Transmiterea proprietăților rețelei de bază	da	în anumite condiții	da
Activabilitatea multiplă, simultană	nu	da	nu
Numărul activărilor permise	1	k	∞

Tipul regulei de tranziție nu are consecințe principiale deorece, introducînd locații complementare într o rețea cu regulă de tranziție tare și instaurând regula de tranziție slabă se obține o rețea echivalentă. Întrucât toate procedeele de structurare se bazează pe forma bloc și respectă condițiile impuse în definiția acesteia, fiecare procedeu asigură conservarea proprietăților rețelei de bază.

În cazul subrețelelor după Valette și Abel proprietățile de viabilitate, mărginire și reversibilitate ale rețelei de bază sunt transmise și rețelei desfășurate. În cazul subrețelelor după Suzuki Murata, pentru a nu impune condiții inutil de restrictive, transmiterea proprietăților rețelei de bază se specifică pentru fiecare caz concret în parte.

Definițiile de subrețele conform Abel și Valette sunt foarte restrictive. Rețelele Petri care înglobează astfel de subrețele au o structură clară și transparentă deoarece comportarea subrețelelor este foarte strict reglementată. Definiția dată de Abel pentru subrețele permite decuplarea totală a acesteia față de rețeaua de bază deoarece nu s-a impus nici o condiție de concesionare. Definiția rigidă are dezavantajul de a nu putea modela situații care apar în sisteme tehnice (ireversibilități și blocaje parțiale) și prin urmare acestea trebuie modelate indiferent de însemnătatea lor în rețeaua de bază. Se pierde astfel posibilitatea realizării unor structuri ierarhice raționale.

În situația în care rețeaua de bază modelează blocaje parțiale, justificarea definițiilor restrictive pentru subrețele prin argumentul transmiterii proprietăților rețelei de bază în rețeaua desfășurată își pierde valabilitatea.

Interzicerea concesionării multiple restrânge semnificativ puterea de modelare a subrețelelor după Valette și Abel. Aceasta deoarece presupunând, de exemplu, un proces complex de prelucrare a unor piese, subrețeaua desfășurată poate să conțină o secvență de diferite prelucrări executate pe mașini distincte. Într-un astfel de șir se pot afla mai multe piese simultan, în diferite stadii de prelucrare. O astfel de situație poate fi modelată doar printr-o tranziție multiplu concesionabilă.

Din cauza următoarelor dezavantaje:

1. În anumite situații, rezervarea capacităților unor locații poate duce la blocarea evoluției rețelei de bază (subsistemul care generează condițiile pentru activarea tranziției t_{in} este blocat în mod artificial iar faptul că rețeaua de bază trebuie să aștepte după tranziția t_{out} este "ascuns" în sintaxa rețelei în cazul Abel respectiv în structura complementară în cazul Valette cu regula de concesionare tare).
2. Rețeaua de bază, pe baza sintaxei nemodificate, nu mai descrie în mod corect comportamentul procesului modelat din cauza faptului că nu mai redă consumul de marcaje din premulțimea tranziției t_v și reținerea acestora, respectiv nu furnizează informații despre acest fapt;

procedeele de mai sus nu dau rezultate corecte în următoarele situații:

(i) tranziția care modelează subrețeaua concurează la obținerea unei resurse concurate;

(ii) pentru conducerea procesului modelat se utilizează o reacție după stare care are nevoie de informații referitoare la consumul de marcaje din premulțimea tranziției t_{in} respectiv asupra șederii acestora în structura subrețelei;

(iii) se dorește sinteza unui echipament de conducere pentru subrețea, caz în care este necesară definirea interfeței dintre echipamentul de conducere a subrețelei și a rețelei de bază.

9.2 Aspecte specifice de implementare a sistemului de conducere pe baza structurilor ierarhice de rețea Petri

Presupunând faptul că sopul modelării prin rețele Petri este nu doar analiza ci și implementarea unui sistem de conducere a procesului modelat, apar aspecte noi specifice, ce trebuie avute în vedere chiar în cursul modelării prin structuri ierarhice de rețea Petri, aspecte ce au repercusiuni și asupra modului de reprezentare a subrețelei în rețeaua de bază. Aceste aspecte specifice trebuiesc avute în vedere doarece:

(i) se impune definirea interfețelor dintre subsistemele de conducere de pe diferitele niveluri ierarhice, subsisteme corespunzătoare subrețelelor de pe aceste niveluri;

(ii) optimizarea conducerii poate necesita informație de stare despre subsistemele modelate prin subrețele.

9.2.1 Interfațarea (sub)sistemelor de conducere

Așa cum s-a arătat în capitolul 8, pentru ca sistemele de conducere să poată lucra pe baza relațiilor cauzale din procesul condus modelat printr-o rețea partiționată, interfețele trebuie să conțină tranziții comune modelelor de rețea Petri interfațate. Aceste tranziții au asociate predicate ale tranzițiilor implementate prin observatori de evenimente.

Deoarece între rețeaua de bază și o anumită subrețea există cel puțin două interfețe, rețeaua de bază trebuie să "vadă" cel puțin două tranziții ale subrețelei. Din acest motiv, structura minimală de reprezentare a unei subrețele în rețeaua de bază trebuie să fie cea din Figura 9-3. Această structură pune în eveidență, în mod efectiv, tranzitia de intrare și tranziția de ieșire iar prin locația p_v dintre ele încorporează și informație de stare.

Această reprezentare poate fi justificată și prin faptul că subrețelele conțin cu siguranță și locații. Acestea, din punctul de vedere al marcajelor, sunt elemente de acumulare care preiau și reține marcaje din premulțimea tranziției care reprezintă subrețeaua în rețeaua de bază. Locația p_v încorporează informație de stare reprezentată de marcajul acesteia. Marcajul depinde de numărul execuțiilor tranziției t_{in} și t_{out} respectiv de ponderea w_{in} și w_{out} a arcelor conectate la locația p_v.

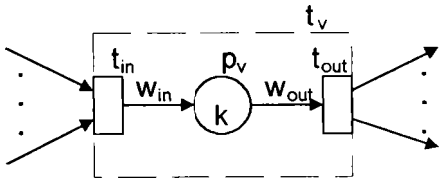


Fig. 9-3 Structură minimală cu informație de stare

Prezenta lucrare propune această structură pentru reprezentarea subrețelelor. Se pleacă de la aceeași ipoteză în privința interfeței dintre rețeaua de bază și subrețea: subrețeaua comunică cu rețeaua de bază printr-o singură tranziție de intrare t_{in} și o singură tranziție de ieșire t_{out}. Condițiile pe care trebuie să le satisfacă subrețeaua devin:

1. Subrețeaua trebuie să asigure concesionarea tranziției t_{in}. (9.3)
2. Subrețeaua reprezentată de structura din Figura 9-3 trebuie să fie viabilă și reversibilă. (9.4)

Prima condiție are drept consecință faptul că locația p_v este nemarcată în starea inițială a rețelei de bază. A doua condiție are drept consecință faptul că marcajele (jetoanele) acumulate în locația p_v, ca urmare a execuțiilor repetate a tranziției t_{in}, vor fi golite în totalitate prin execuții repetate ale tranziției t_{out}. În cazul în care $w_{in}=w_{out}$ a doua condiție este echivalentă cu a doua condiție din definiția formei bloc. În acest caz, în cadrul subrețelei, la o concesionare a tranziției t_{in} va corespunde exact o concesionare a tranziției t_{out}.

Observație: Tranzițiile t_{in} și t_{out} trebuie să fie tranziții efective în procesul condus având implementate observatorii de evenimente corespunzători predicatelor tranzițiilor.

Dacă aceste două condiții sunt îndeplinite subrețeaua va simula corect execuția structurii din Figura 9-3 și nu va provoca blocajul rețelei desfășurate.

Această reprezentare are următoarele avantaje:

1. Pune la dispoziție tranzițiile necesare definirii interfeței dintre sistemul de conducere implementat pe baza rețelei de bază și a sistemului de conducere implementat pe baza subrețelei.
2. Reprezentând subrețeaua prin acestă structură, rețeaua de bază descrie comportamentul real al procesului modelat. De fapt, față de reprezentarea printr-un singur nod, această reprezentare pune în acord sintaxa rețelei cu comportamentul real al procesului modelat. În cazul reprezentării printr-o singură tranziție este necesară ori o modificare a sintaxei rețelei, așa cum procedează Abel, ori o modificare a structurii rețelei, așa cum se procedează în cazul Valette pentru regula de concesionare tare, ori definirea unui nou tip de tranziție în cazul Suzuki-Murata. În fiecare dintre aceste cazuri se pierde informația referitoare la preluarea și menținerea de marcaje din premulțimea tranziției care reprezintă subrețeaua.
3. Concesionabilitatea multiplă este reprezentată în mod natural, fără a defini un nou tip de tranziție așa cum s-a procedat în cazul Suzuki-Murata. Multiplicitatea concesionabilității este dată de raportul k/w_{in} dintre capacitatea locației p_v și multiplicitatea w_{in} a arcului (t_{in},p_v).
4. Se pune la dispoziție informație suplimentară, față de reprezentarea printr-o singură tranziție, ce poate fi utilizată la realizarea reacției după stare în rețeaua de bază. Aceasta informație poate fi utilizată și trebuie să fie utilizată în cursul analizei de performanță.

9.2.2 Informația de stare asupra subrețelei pusă la dispoziția rețelei de bază

Structura minimală din Figura 9-3, prin macajul locației p_v, pune la dispoziția rețelei de bază (a sistemului de conducere implementat pe baza acesteia) informație de stare asupra procesului modelat prin subrețea. Având în vedere încapsularea subrețelei între două tranziții, dacă există o ordonare cauzală, obiectivă între anumite tranziții ale subrețelei, ordonare determinabilă printr-un procedeu oarecare, atunci se poate ajunge la o structură secvențială de forma celei din Figura 9-4.

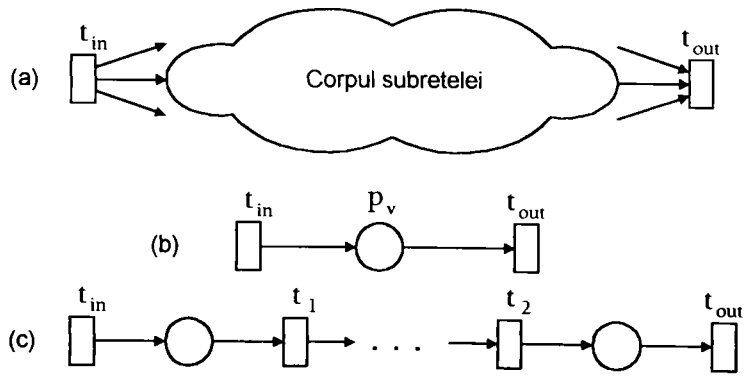


Fig. 9-4 Informație de stare obtenabilă asupra subrețelei

Figura 9-4(a) prezintă încapsularea subrețelei între tranzițiile t_{in} și t_{out}. În Figura 9-4(b) marcajul locației p_v pune la dispoziție informație doar asupra caracterului acumulativ al subrețelei. Figura 9-4(c) pune la dispoziție informație de stare utilizabilă în conducerea optimală cu condiția ca tranzițiile t_1 și t_2 să fie tranziții efective ale subrețelei și între tranzițiile t_{in}, t_1, ... t_2, și t_{out} să existe relația cauzală obiectivă modelată prin această structură secvențială.

9.2.3 Subrețele reprezentate prin structuri secvențiale cu informație de stare și analiza rețelei de bază

Dezavantajul acestei forme de reprezentare este faptul că introduce suplimentar în structura rețelei de bază o locație, p_v, și o tranziție, t_{out} respectiv o structură secvențială dacă se doresc informații de stare suplimentare. Consecințele acestei modificări structurale depind de contextul în care ea se realizează. Astfel:

a) Dacă toate subrețelele se reprezintă în acest fel încă din momentul elaborării modelului de rețea Petri de bază, dezavantajul constă doar în însăși numărul crescut al nodurilor căci analiza va scoate în evidență toate proprietățile rețelei de bază ce include nodurile suplimentare.

b) Este mult mai interesantă însă situația în care rețeaua de bază o dată sintetizată, analizată și corectată este detaliată prin aceste structuri secvențiale. Detalierea se poate executa identificând tranzițiile ce urmează a fi detaliate și înlocuindu-le prin structurile secvențiale corespondente. Consecințele acestei înlocuiri sunt determinate de structura rețelei. Două situații distincte pot fi puse în evidență:

   (i) tranziția detaliată concurează pentru obținerea unei resurse partajate;

   (ii) tranziția detaliată nu concurează pentru obținerea vre-unei resurse partajate.

Pentru a ilustra cazul (i) de mai sus în Figura 9-5(a) se prezintă o rețea Petri simplă împreună cu graful ei accesibil, generat conform sintaxei rețelei. Acest exemplu arată că se pot produce marcaje care nu sunt prezente în modelul de rețea Petri de bază.

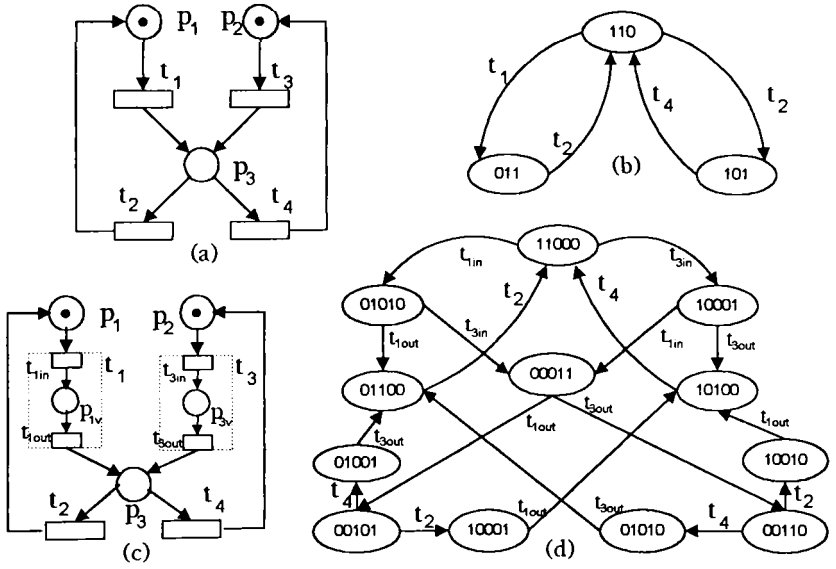


Fig. 9-5 Exemplu

Tranzițiile t_1 și t_3 se înlocuiesc prin structura prezentată în Figura 9-3. Se va considera că durata de reținere a marcajelor în locația p_{1v} este τ_1 iar în locația p_{3v} este τ_3 astfel încât

$\tau_1>\tau_3$. Figura 9-6 prezintă diagrama pentru o posibilă ordonare temporală a execuției rețelei din Figura 9-5.

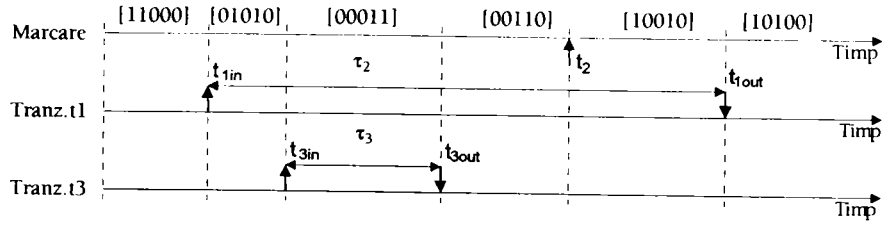


Fig. 9-6 Ordonarea temporală pentru exemplul anterior

Comparând această diagramă cu graful de accesibilitate din Figura 9-5(b) se observă că se pot produce marcajele suplimentare [010xx],[000xx],[001xx],[101xx], nerealizabile în rețeaua de bază.

La nivelul tranzițiilor de ieșire t_{1out} respectiv t_{2out} excluderea mutuală în utilizarea resursei modelate de locația p_3 se realizează corect spre deosebire de situația în care tranzițiile t_1 și t_3 se consideră a fi tranziții temporizate ce consumă marcaje din locațiile premulțimii și le reține o anumită perioadă, așa cum s-a prezentat în capitolul 6.

Datorită nodurilor suplimentare apar o mulțime de marcaje și secvențe de tranziții care nu sunt prezente în graful accesibil al rețelei de bază. Ca urmare în cazul sistemelor tehnice, aplicând regula de tranziție clasică, rezultatele obținute prin analiza grafului accesibil sau a matricii de incidențe își pierd valabilitatea. În cazul de față deosebirea esențială, față situația inițială, este prezența marcajului [00011] adică posibilitatea ca locațiile p_1 și p_2 să fie simultan nemarcate, ceea ce în rețeaua inițială este exclusă. Acest lucru se datorează faptului că resursa reprezentată de locația p_3 este partajată de către tranzițiile t_1 și t_3. Acest fenomen poate crea probleme de exemplu în cazul în care reacția după stare sau partea de rețea nereprezentată aici dar conectată la aceste locații, se bazează pe această proprietate.

Această problemă se rezolvă dacă încă din faza de modelare/proiectare se procedează la "încapsularea" procesului de arbitrare a resurselor, adică la separarea procesului de arbitrare (de alocare) a resursei de procesele de prelucrare informațională sau materială conform principiului enunțat în capitolul 6.

În practică acest principiu înseamnă că resursele partajate modelate în cadrul unui anumit nivel ierarhic vor fi arbitrate chiar la acest nivel. Arbitrarea nu va depinde de procesele unui nivel ierarhic inferior. Această regulă de modelare respectă întru-totul principiul abstractizării pe niveluri funcționale. De fapt numai în acest fel organizarea ierarhică a modelelor de rețea Petri va respecta principiile de bază ale abstractizării funcționale pe niveluri ierarhice.

Se poate observa că în acest caz marcajele [000xx] pot să apară în rețeaua de bază chiar înaite de detalierea acesteia și deci analiza efectuată va ține cont de acest marcaj.

Respectând acest principiu toate tranzițiile care reprezintă subrețele se încadrează în cazul (ii) de mai sus, adică tranziția dată nu concurează pentru obținerea vre-unei resurse partajate. Astfel de tranziții vor fi încadrate în structuri de rețea secvențiale de tipul celei prezentate în Figura 9-7, locațiile premulțimii tranziției t_{in} respectiv locațiile postmulțimii tranziției t_{out} fiind locații neconcurate.

Structura secvențială care reprezintă subrețeaua în rețeaua de bază nu va influența caracteristicile dinamice ale rețelei de bază (viabilitate, reversibilitate, mărginire)[PĂS'97]. Stările, respectiv marcajele suplimentare ce apar sunt legate doar de marcajul locației p_v.

nici un fel de prezenţa locaţiei p_v şi a tranziţie t_{out}. Aceasta va evolua neschimbat doar că evoluţia ei poate fi observată la diverse valori ale $m(p_v)$. Evoluţia reţelei de bază poate fi însă influenţată printr-o reacţie după stare în măsura în care conţine tranziţii controlabile.

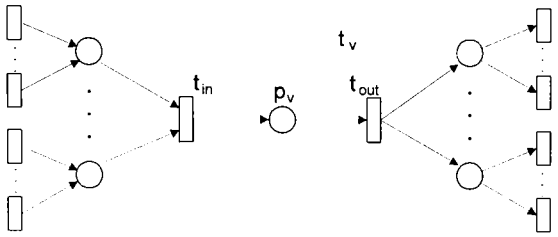


Fig. 9-7 Încadrarea subreţelei în reţeaua de bază după încapsularea proceselor de arbitrare a resurselor

În cazul structurilor secvenţiale la care participă, această structură va introduce doar un pas (paşi) suplimentari în calea fluxului de marcaje. Fluxul de marcaje nu va fi blocat, nu se vor crea marcaje suplimentare şi nu se distrug marcaje faţă de situaţia iniţială.

Dacă se respectă acest principiu detalierea trebuie efectuată doar în situaţia în care reacţia după stare are nevoie de informaţia încorporată în structura secvenţială iar analiza reţelei de bază poate fi efectuată reprezentând subreţeaua printr-o tranziţie.

Strategia de conducere are nevoie de informaţia de stare conţinută în structurile secvenţiale prezentate mai sus şi ca urmare maşina virtuală de reţea Petri se va implementea pe baza reţelei de bază completată cu aceste structuri. Respectând condiţiile formulate mai sus maşina virtuală de reţea Petri este definită şi implementabilă.

9.3 Structuri de conducere pentru subreţele

Pentru a putea realiza un echipament de conducere prin care se poate controla o subreţea într-o structură ierarhică de modele de reţea Petri trebuie determinată structura reţelei pe baza căreia poate fi realizată implementarea. Interfeţele dintre reţeaua de bază şi subreţea pot fi definite conform conceptelor introduse în capitolul 8. Procedând în acest fel se obţin structuri de interfeţe relativ complexe aşa cum se prezintă în Figura 9-7.

În cazul structurilor ierarhice de modele de reţea Petri este avantajos a modela şi implementa interfeţele în mod explicit pe baza structurii minimale (t, p, t). Procedând în acest fel structura modelului de reţea Petri a reţelei de bază devine cea din Figura 9-8(a) iar cea a subreţelei cea din Figura 9-8(b).

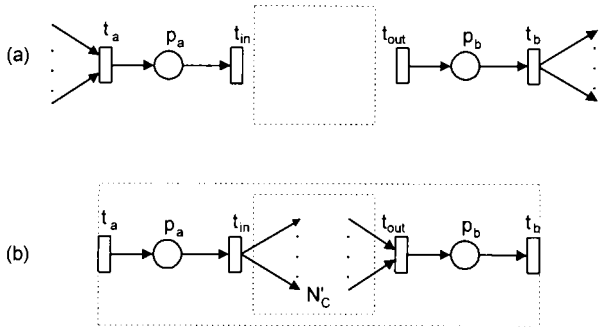


Fig. 9-8 Structura de reţea pentru implementarea echipamentului de conducere a subreţelei

Cele două interfeţe prin care se realizează încapsularea subreţelei sunt modelaţe prin tripleţii (t_a, p_a, t_{in}) şi (t_{out}, p_b, t_b). Elementele de reţea care modelează interfeţele sunt comune celor două reţele. Corpul subreţelei s-a notat prin N'_C. Echipamentul de conducere a subreţelei se va implementa pe baza structurii de reţea din Figura 9-8(b).

Matricea de incidenţe pe baza căreia se sintetizează echipamentul de conducere va conţine toate elementele de reţea din Figura 9-8(b). Interacţiunea dintre echipamentul de conducere a reţelei de bază şi echipamentul de conducere a subreţelei se realizează prin mecanismele prezentate în capitolul 8 pentru cazul controlului distribuit. Structura de conducere pe baza căreia se va implementa echipamentul de conducere a subreţelei se prezintă în Figura 9-9.

În acest caz, interfaţarea se realizează doar între subreţea şi reţeaua de bază. Componentele g_e şi g_r ale vectorului de concesionare vor avea câte un singur element.

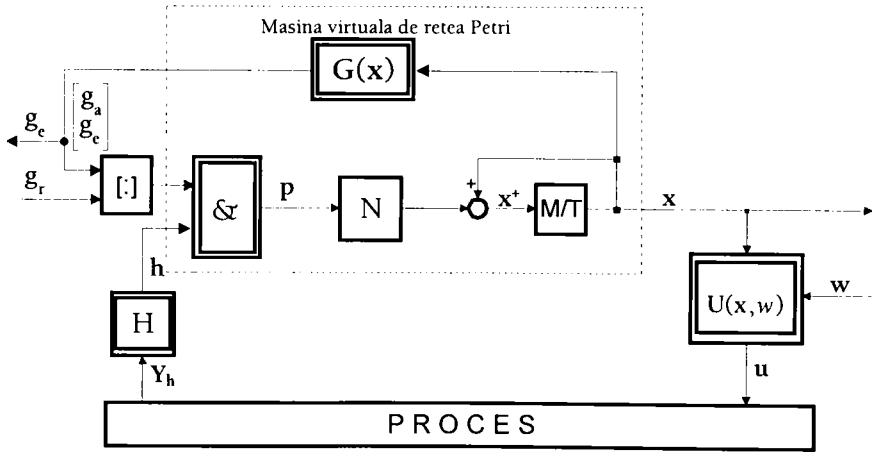


Fig. 9-9 Structra de conducere pentru subreţea şi reţeaua de bază

Aceeaşi structură de conducere se va utiliza şi pentru implementarea sistemului de conducere a reţelei de bază doar că în acest caz vectorii g_e şi g_r vor avea atâtea elemente câte subreţele contine reţeaua de bază.

Dacă se au în vedere și interfețele, subrețeaua trebuie analizată pe baza structurii de rețea din Figura 9-10. Dacă această rețea este viabilă subrețeaua va simula corect funcționarea structurii din Figura 9-3 și în același timp va asigura transmiterea proprietăților rețelei de bază și rețelei desfășurate.

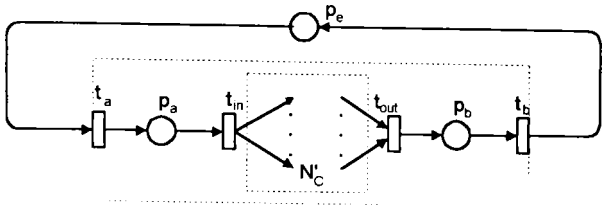


Fig. 9-10 Structura pentru analiza subrețelei

În Figura 9-10 corpul subrețelei s-a notat prin N_C'. Pentru a defini subrețeaua din punct de vedere dinamic se impun condițiile de **viabilitate** și **mărginire** rețelei din Figura 9-10.

Pentru efectuarea analizei trebuie definit marcajul inițial al subrețelei. Marcajul inițial al structurii de rețea din Figura 9-10 se compune din:

(i) marcajul inițial implicit al rețelei N_C';

(ii) marcajul inițial al locației p_e.

În acest fel se asigură că dacă sunt îndeplinite condițiile pentru realizarea funcțiunii modelate prin subrețea, aceasta se realizează și tranziția de ieșire t_{out} se va executa, cu condiția să fie concesionată în rețeaua de bază. În acest fel se asigură ca proprietățile rețelei de bază să fie transmise rețelei desfășurate. Pentru rețeaua din Figura 9-10 condiția de viabilitate este echivalentă cu condiția de reversibilitate.

Această formă de reprezentare este în acord cu conceptul de structurare pe niveluri ierarhice conform căruia funcțiunile nivelului superior apelează la serviciile oferite de către nivelul inferior. Acesta din urmă răspunde printr-un rezultat al acțiunii proprii serviciului solicitat. În prima fază a acestei interacțiuni nivelul superior este elementul activ (are inițiativa în solicitarea serviciului iar nivelul inferior este elementul pasiv care preia solicitarea). Din acest motiv locația p_a este elementul natural pentru modelarea acestei interacțiuni.

În interacțiunea de furnizare a rezultatelor acțiunii asociate serviciului solicitat rolurile se inversează și este normal ca și caracterul elementului de interfațare să se inverseze aceasta fiind în acest caz tranziția t_{out}.

Elementele de interfață aparțin în egală măsură rețelei de bază și subrețelei. Elementul strict propriu al subrețelei este corpul acesteia, N'^{C} în Figura 9-10.

În acest sens denumirea de locație de intrare respectiv tranziție de ieșire este relativă, deoarece în rețeaua de bază rolul acestora se inversează. Această "proprietate comună" se manifestă și în cursul analizei deoarece elementele de interfață sunt prezente, în cursul anlizei, atât în rețeaua de bază cât și în subrețea.

Acestă reprezentare este în acord și cu specificarea interfețelor rezultate prin interpretarea din capitolul 6. Conform acestei specificații interacțiunile dintre elementele funcționale ale sistemelor modelate prin rețele Petri se realizează prin informație transportată pe frontul unui semnal sau printr-un mesaj interpretat în mod similar.

9.4 Concluzii

În cadrul acestui capitol s-a prezentat o sinteză a procedeelor de detaliere a subrețelelor într-o structură ierarhică de rețele Petri și s-a propus o nouă metodă de reprezentare a subrețelelor în rețeaua de bază.

Metodele de structurare după Valette, Suzuki-Murata și Abel se bazează pe forma bloc a subrețelei care permite reprezentarea acesteia, în rețeaua de bază, printr-o singură tranziție. Diferențele dintre aceste procedee rezultă din modalitatea în care, prin restricții structurale, se asigură transmiterea proprietăților rețelei de bază și rețelei desfășurate. În cazul Suzuki-Murata se prezintă și o definiție a concesionabilității multiple.

S-a propus a nouă modalitate de reprezentare a subrețelelor în rețeaua de bază printr-o structură minimală care încorporează informație de stare ce depinde de numărul de execuții a tranziției de intrare și a celei de ieșire. Această formă de reprezentare are următoarele avantaje:

1. Pune la dispoziție tranzițiile necesare definirii interfeței dintre sistemul de conducere implementat pe baza rețelei de bază și a sistemului de conducere implementat pe baza subrețelei.
2. Pune în acord comportamentul rețelei de bază, determinată de sintaxa acesteia, cu comportamentul procesului modelat.
3. Concesionabilitatea multiplă este reprezentată în mod natural.
4. Pune la dispoziție informație suplimentară pentru reacția după stare.
5. Face posibilă definirea structurii de rețea pe baza căreia se poate implementa un echipament de conducere a subrețelei în așa fel încât să fie definită și interfața cu echipamentul de conducere a rețelei de bază.

S-a arătat că dacă modelul de rețea Petri al sistemelor tehnice este astfel elaborat încât procesele de arbitrare a resurselor să fie "încapsulate" adică separate de procesele de prelucrare informațională și materială atunci detalierea rețelei de bază prin structura propusă nu schimbă caracteristicile dinamice de bază (viabilitate, reversibilitate) ale rețelei de bază.

S-a definit structura de rețea pentru implementarea echipamentului de conducere a subrețelei și, pe baza conceptului de mașină virtuală de rețea Petri, s-a prezentat o structură de conducere pentru implementare echipamentului de conducere. S-a definit deasemnea structura de rețea necesară analizei subrețelei. Aceasta este în acord cu principiile de structurare pe niveluri ierarhice și cu interpretarea dată rețelelor în capitolul 6.

* * *

Față de literatura de specialitate la care autorul a avut acces, acest capitol prezintă următoarele elemente originale:

a) Definirea structurii minimale cu informație de stare (Fig. 9-3) pentru reprezentarea subrețelelor în rețeaua de bază și stabilirea condițiilor pe care trebuie să le îndeplinească subrețelele astfel reprezentate, pentru a asigura transmiterea proprietăților rețelei de bază în rețeaua desfășurată.

b) Autorul a avansat ideea unei analize suplimentare a subrețelei, analiză efectuată cu scopul de a determina structuri secvențiale care modelează relații cauzale obiective dintre tranziții efective ale subrețelei. Informația de stare furnizată de această structură poate fi utilizată pentru optimizarea strategiei de conducere bazate pe reacția după stare. Această soluție oferă avantajul unei optimizări bazate pe mărimi măsurate în procesul condus, strategia nefiind nevoită să recurgă la temporizări.

Lucrările viitoare trebuie să determine metode practice de a obține o astfel de structură secvențială.

c) Demonstarea faptului că dacă, în rețeaua de bază, procesele de arbitrare a resurselor sunt încapsulate, subrețelele pot fi reprezentate prin tranziții fără ca rafinarea lor ulterioară să modifice proprietățile de viabilitate și reversibilitate. În schimb rafinarea aduce informația suplimentară de stare cuprinsă în structura din Fig. 9-3 sau Figura 9-4.

d) Definirea structurii de rețea pe baza căruia poate fi implementat un echipament de conducere a subrețelei (a procesului modelat prin aceea subrețea) și prezentarea unei structuri de conducere, pe baza conceptului de mașină virtuală de rețea Petri și a conceptului de domeniu de interfațare introdus în capitolul 8.

10. Reţele Petri modulare

Modularitaea este o caracteristică de bază a sistemelor flexibile de fabricaţie. Această modularitate trebuie să se reflecte şi asupra modelului de reţea Petri al sistemului. Acest capitol prezintă, pentru cazul modelării prin reţele Petri, conceptele modularizării şi interfaţării dintre module. Sistemele flexibile de fabricaţie se modelează deosebit de eficient prin reţele Petri colorate. Se prezintă definiţia formală şi sintaxa reţelelor Petri colorate. Se arată că şi în acest caz sunt utilizabile conceptele maşinii virtuale de reţea Petri. Se prezintă modul cum se determină reţelele Petri necesare analizei diverselor module respectiv reţelele Petri pe baza cărora se implementează echipamentele de comandă ale acestor module.

10.1 Preliminarii

În cazul structurării pe niveluri ierarhice a reţelelor Petri, subreţelele pot fi privite ca module funţionale. Acestea reprezintă rezultatul unei abstractizări funcţionale şi prin urmare, în mod declarat, fac abstracţie de orice element de structurare constructivă a sistemului modelat. În acest fel se asigură nu doar un grad înalt de abstractizare ci şi un grad înalt de generalitate.

În practică însă, deseori apare situaţia în care sistemul modelat se compune din elemente funcţionale şi constructive bine definite şi delimitate. Un exemplu tipic în acest sens sunt sistemele flexibile de fabricaţie. Acestea au în componenţă staţii de lucru şi sisteme de transport bine definite atât din punct de vedere funcţional cât şi din punct de vedere constructiv.

Astfel de sisteme au un caracter pronunţat modular [Coj'90a], [Coj'90b]. Dacă se modelează prin reţele Petri, este avantajos, din punctul de vedere al ingineriei sistemului, ca această modularitate să se reflecte şi la nivelul modelului. Practic, acest lucru înseamnă că se vor elabora module de reţea Petri. Modelul sistemului se va obţine prin asamblarea acestor module cu ajutorul unor tehnici de interfaţare sau de comunicaţie dintre modulele de reţea Petri.

Considerentele de mai sus justifică denumirea de "reţele Petri modulare".

Scopul acestui capitol este de a prezenta module reprezentative şi tehnici de interfaţare între aceste module precum şi modul de determinare a reţelelor Petri necesare analizei diverselor module respectiv reţelele Petri pe baza cărora se implementează echipamentele de comandă. Se vor prezenta şi structurile de conducere pe baza cărora pot fi implementate echipamentele de comandă necesare conducerii sistemului modelat.

10.2 Modularizare. Interfaţare.

Caracteristica definitorie a modulelor din sistemele flexibile de fabricaţie este ciclicitatea operaţiilor efectuate. Din acest motiv structura caracteristică în modelarea prin reţele Petri este cea a unei bucle, aşa cum se prezină Figura 10-1.

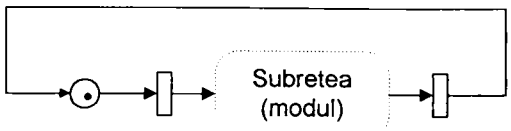


Fig. 10-1 Structura unui modul de rețea Petri

În continuare prin "modul" se va înțelege subrețeaua din Figura 10-1.

Pentru ca ciclicitatea să fie efectiv realizabilă modulul trebuie să fie viabil. În sistemele tehnice nu sunt admise acumulări infinite sau întârzieri infinite; din acest motiv modulele trebuie să fie și mărginite. Reversibilitate modulelor, respectiv necesitatea acestei calități, se va studia pentru fiecare caz în parte.

Modulele se conectează între ele prin interfețe. În cele ce urmează se definesc câteva interfețe tipice.

10.2.1 Interfațarea modulelor

a) Interfețe de sincronizare (Figura 10-2(a))

"Modulul M_1 sincronizează modulul M_2 printr-o solicitare (REQest) și achitarea corespunzătoare (ACKnowlege)". Modulul M_2 așteaptă o solicitare, REQ, din partea modulului M_1 pentru a realiza o acțiune ce se inițiază prin execuția tranziției T_m. Solicitarea se transmite printr-o cale de comunicație. Finalizarea acțiunii sincronizate este semnalizată de către modulul M_2 printr-un semnal sau mesaj transmis pe calea "ACK".

b) Interfața "Consumator-Producător" (Figura10-2(b))

"M_1 consumă (CONSumer) un marcaj produs de M_2 respectiv M_1 produce (PRODuce) un marcaj pentru M_2". PRODuce reprezintă o solicitare emisă de către modulul M_1 către modulul M_2. În funcție de cazul concret CONS poate reprezenta un mesaj care certifică disponibiltatea produsului sau produsul însuși.

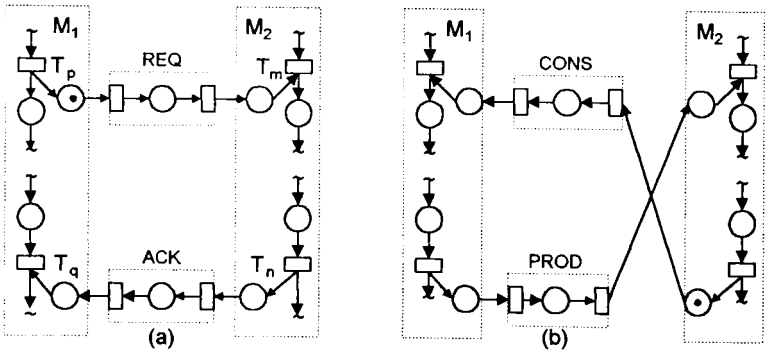


Fig. 10-2 Interfața de sincronizare (a) și interfața "consumator-producător" (b)

În funcție de gradul de detaliere necesar, interfețele pot fi modelate prin una din structurile prezentate în Figura 10-3. Interfețele au rolul de a transporta informație (comenzi, semnalizări) sau substanță (prefabricate, semifabricate, produse finite) între module.

Structura din Figura 10-3(a) are dezavantajul relativei complexități. Are însă următoarele avantaje majore:

1. Delimitează în mod clar elementele componente ale modulului de cele ale interfeței;
2. În cazul unui control distribuit, conform capitolului 8, domeniile de interfațare sunt chiar interfețele naturale.
3. Modelează întârzierea introdusă de procesul de transport (al informației sau substanței).

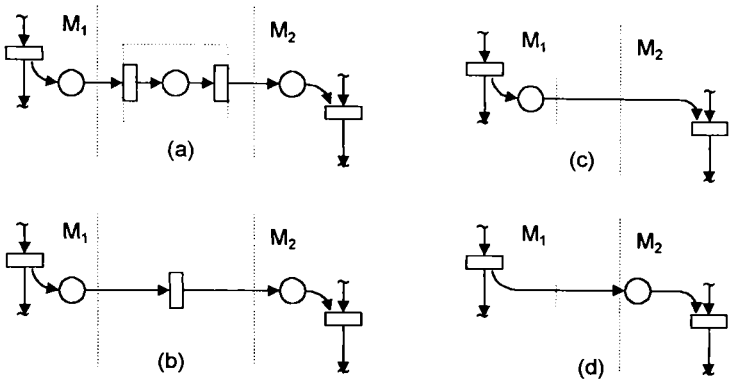


Fig. 10-3 Alternative de reprezentare a interfețelor

Structura din Figura 10-3(b) modelează procesul de transport (al informației sau substanței) printr-un singur nod de rețea, o singură tranziție, făcând astfel abstracție de întârzierea introdusă. Conform acestui model informația sau materialele sunt instantaneu trecute de la modulul M_1 la modulul M_2.

Structurile din Figura 10-3(c) și (d) modelează legătura dintre module doar printr-un arc al rețelei, prin urmare nu modelează comunicația dintre module, nu iau în considerare caracteristicile acesteia și nu scot în evidență existența sau necesitatea unui subsistem de comunicații sau transport dintre module.

Într-un sistem de fabricație flexibil sistemul de comunicații de date și sistemul de transport sunt elemente esențiale ale integrării modulelor componente și ale asigurării flexibilității. Prin urmare nici un model al unui sistem flexibil de fabricație nu poate face abstracție de existența acestora.

Prin urmare cea mai simplă structură care modelează în mod realist interfața dintre module este structura din Figura 10-3(a). Nu este corect a simplifica modelul de rețea Petri pe seama acestei structuri. Proiectantul trebuie să se concentreze asupra definirii și modelării corecte a unor module funcționale bine conturate, definite pe baza principiior generale ale abstractizării funcționale prezentate în capitolul 2.

c) Alocarea resurselor prin semafoare

Structura de rețea care modelează acest proces este prezentată în Figura 10-4. "Resursa comună R(Resource) este alocată prin excludere mutuală fie modulului M_1 fie modulului M_2." Se poate observa că structura prezentată în Figura 10-4 asigură încapsularea arbitrării

resursei partajate în sensul că arbitrarea este realizabilă, prin tranzițiile T_1 și T_2, independent de procesele de prelucrare.

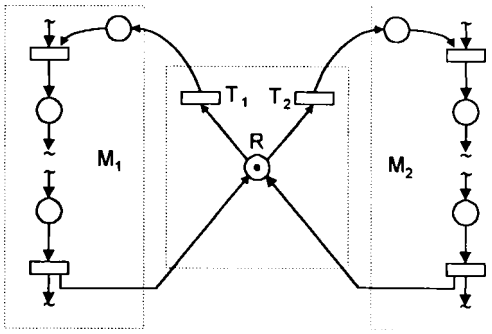


Fig. 10-4 Arbitrarea unei resurse partajate

Se prezintă în continuare două module reprezentative ale unui sistem flexibil de fabricație.

10.2.2 Module tipice într-un sistem flexibil de fabricație

a) Modulul de montaj

Acest modul de rețea Petri descrie activitatea unei stații de montaj destinate montării a două piese. Modelul de rețea Petri al modulului se prezintă în Figura 10-5.

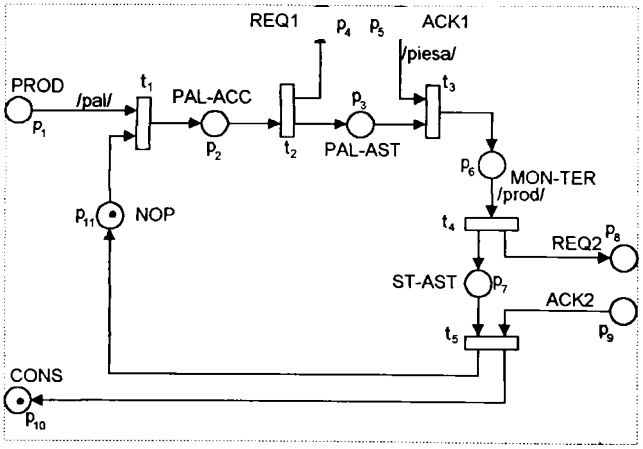


Fig. 10-5 Modelul de rețea Petri al unei stații de montaj

Conform Figurii 10-5 acest modul prezintă trei interfețe:

1. Interfața PROD/CONS pentru preluarea unui obiect, în cazul de față cu ajutorul unei palete /pal/.
2. Interfața REQ_1/ACK_1 pentru solicitarea obiectului ce se va monta pe paleta /pal/, referit în acest caz prin /piesa/. /piesa/ trebuie să ajungă la locul operației de montare doar după sosirea paletei.

3. Interfața REQ_2/ACK_2 pentru interacțiunea cu sistemul de transport în vederea descărcării stației de montaj de produsul, /prod/, rezultat din montarea piesei pe paletă.

Locația PROD modelează un buffer de intrare prin care un marcaj /pal/ ajunge în stația de montaj propriu-zisă. Dacă stația este disponibilă, locația NOP este marcată, sunt date toate condițiile pentru execuția tranziției t_1 (concesionare). Marcajul /pal/ este trecut în locația PAL-ACC (paletă acceptată). După ce paleta a fost acceptată se execută tranziția t_2 care solicită prin REQ_1 piesa ce se va monta pe paletă. Dacă /piesa/ este disponibilă, locația ACK_1 marcată, poate fi executat montajul propriu-zis prin tranziția t_3. Se realizează astfel produsul iar în modelul de rețea Petri se marchează locația MON-TER (montajul terminat). Prin execuția tranziției t_4 se solicită descărcarea stației de montaj (REQ_2) de paleta care conține produsul. Îndepărtarea paletei la o distanță suficient de mare, din punctul de vedere al securității, se semnalizează prin marcarea locației ACK_2.

Prin execuția tranziției t_5 stația ajunge în starea NOP, apță de a accepta o nouă paletă fapt semnalizat prin marcarea locației CONS.

Prin perechea de interfețe PROD/CONS și REQ_2/ACK_2 se poate asigura ca în interiorul modelului să nu se acumuleze marcaje, în stația de montaj să nu se acumuleze palete respectiv produse. Acest lucru este rațional atât din punct de vedere tehnic cât și al analizei modelului.

Utilizarea mai multor interfețe REQ/ACK permite modelarea unor stații de montaj pentru mai multe piese. Această posibilitate, la prima vedere avantajoasă, trebuie utilizată cu atenție deoarece se poate ajunge la pierderea clarității modelării.

b) Modulul de transport

Funcționarea unei instalații de transport care poate realiza transportul, de exemplu, de la stația de prelucrare A la stația de montaj B, poate fi modelată conform rețelei Petri din Figura 10-6.

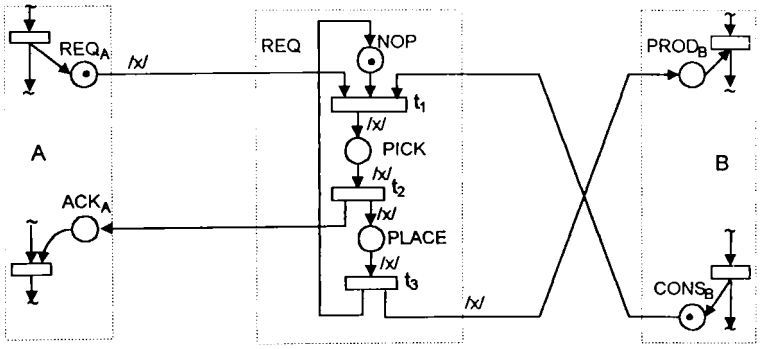


Fig. 10-6 Modelul de rețea Petri al unei instalații de transport

Transportul poate fi inițiat doar în momentul în care sunt satisfăcute următoarele condiții:

a) stația A solicită acest lucru printr-un marcaj în locația REQ_A.

b) modulul de transport este liber, fapt semnalizat prin marcajul locației NOP;

c) stațai B este apță de a prelua obiectul transportat, fapt semnalizat prin marcajul locației $CONS_B$;

În prima fază a transportului, notat prin PICK, obiectul este preluat din staţia A. După ce obiectul a fost îndepărtat peste o anumită distanţă de siguranţă se emite semnalul de achitare a acţiunii de preluare, notat prin ACK_A. Acest semnal se emite prin execuţia tranziţiei t_2 prin care se marchează şi locaţia PLACE. După aceasta obiectul este depus în bafferul de intrare $PROD_B$ a staţiei B prin execuţia tranziţiei t_3. În acelaş moment se marchează şi locaţia NOP semnalizând prin aceasta eliberarea echipamentului de transport.

În practică, de cele mai multe ori, faza de preluare, PICK, respectiv faza de transport efectiv, PLACE, sunt suplimentar condiţionate de către caracteristicile instalaţiei de transport, aşa cum se prezintă în Figura 10-7. În cazul unei benzi transportoare aceste condiţionări pot reprezenta poziţia unor elemente de oprire/imobilizare a obiectului transportat.

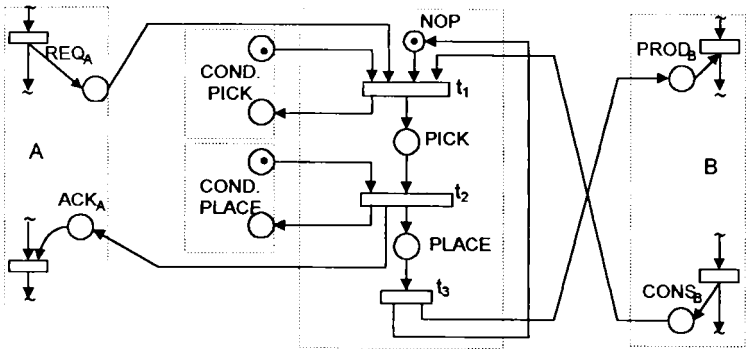


Fig. 10-7 Condiţionări suplimentare în instalaţia de transport

În cazul în care modulele de reţea Petri modelează echipamente ale unui sistem flexibil de fabricaţie ele însele trebuie să fie flexibile respectiv modelul de reţea Petri al modulelor trebuie să fie "flexibil". Altfel exprimat, modelul de reţea Petri al modulelor trebuie să încorporeze informaţia despre flexibilitatea modului respectiv asupra gamei de funcţiuni pe care le poate realiza. Această gamă de funcţiuni determină în mod direct *flexibilitatea de adaptare* a sistemului de fabricaţie. Fiecare sarcină nouă pe care acesta trebuie s-o realizeze implică o modificare a configuraţiei curente a sistemului. În raport cu diferenţierea faţă de sarcina anterioară, intensitatea modificărilor va fi diferită. Pentru adaptare sunt practicate trei strategii de modificare a stării sistemului:

1. *transformarea sistemului* atunci când elementele sau modulele funcţionale sunt înlocuite;
2. *modificarea sistemului*, atunci când dintr-un număr de funcţii parţiale permanent existente şi alternativ utilizabile se selecţionează una anume;
3. *reglarea sistemului* atunci când din mulţimea valorilor pe care le pot lua parametrii de funcţionare se alege pentru fiecare parametru o anumită valoare.

Reţelele Petri Colorate (Coloured Petri Nets, CP-Nets) permit crearea de modele ce sprijină modelarea sistemelor flexibile. În cele ce urmează reţelele Petri colorate vor fi desemnate prin abrevierea CPN.

10.3 Rețele Petri colorate

Rețelele Petri colorate se definesc, pe baza lucrării [Jen'92a]. Înainte de a defini rețelele Petri colorate se introduce noțiunea de "multiset" și se prezintă proprietățile de bază ale acestora.

10.3.1 Multiseturi

În cele ce urmează N va desemna mulțimea întregilor nenegativi iar $[A \rightarrow B]$ mulțimea tuturor funcțiilor de la A la B.

Definiția 10-1

Un **multiset** m peste o mulțime S este o funcție $m \in [S \rightarrow N]$. Întregul nenegativ $m(s) \in N$ specifică **numărul aparițiilor** elementului $s \in S$ în multisetul m.

În mod uzual multisetul m se reprezintă printr-o sumă formală, astfel:

$$\sum m(s)`s .$$

Prin S_{MS} se va nota mulțimea tuturor multiseturilor peste S. Întregii nenegativi $\{m(s) \mid s \in S\}$ se numesc **coeficienții** multisetului m, iar m(s) este numit **coeficientul** lui s. Se spune că elementul $s \in S$ aparține la multisetul m dacă $m(s) \neq 0$ și se scrie în acest caz $s \in m$.

Semnul " ` " din suma formală de mai sus denotă un operator de multiplicitate cu ajutorul căruia se specifică numărul de apariții ale unui element în multisetul dat. În expresiile ce urmează, acest operator va fi întotdeauna explicitat. Coeficienții unitari vor și ei întotdeauna explicitați. Multisetul vid se va nota prin "empty". De fapt fiecare mulțime de elemente are un multiset vid. În cele ce urmează acest fapt se va explicita doar în situația în care utilizarea multisetului generic "empty" ar crea confuzii.

Mulțimea tuturor submulțimilor mulțimii S se va nota prin S_S. Pe baza definiției de mai sus este evident că există o corespondență unu la unu între S_S și multiseturile $m \in S_{MS}$ pentru care $0 \leq m(s) \leq 1$, $\forall s \in S$. Din acest motiv în cele ce urmează se va considera că fiecare mulțime este un multiset (și fiecare multiset $m \in S_{MS}$ pentru care $0 \leq m(s) \leq 1$, $\forall s \in S$, este o mulțime). Aceasta înseamnă că pentru fiecare mulțime $A \subseteq S$ se va utiliza notația A pentru a referi multisetul care corespunde mulțimii A, adică multisetul în care fiecare element al mulțimii A apare o singură dată (și nu apare nici un alt element). În mod analog pentru fiecare element $s \in S$ prin s se va nota multisetul "s", adică multisetul ce corespunde mulțimii {s} sau altfel exprimat multisetul 1`s. În acest sens dacă S={a,b,c,d} și A={a,b,c} scrierea echivalentă, în termeni multiset, a submulțimii $A \subseteq S$ este A=1`a+1`b+1`c.

Operațiile cu multiseturi se definesc astfel:

Definiția 10-2 (Operații cu multiseturi)

Se definesc operațiile de **adunare**, **înmulțire cu un scalar**, **comparare**, și **determinare cardinalitate** pentru oricare $m, m_1, m_2 \in S_{MS}$ și oricare $n \in N$, astfel:

(i) $m_1 + m_2 = \sum_{s \in S} (m_1(s) + m_2(s))`s$ (adunare)

(ii) $n * m = \sum_{s \in S} (n * m(s))`s$ (înmulțirea cu un scalar)

(iii) $m_1 \neq m_2 = \exists s \in S: m_1(s) \neq m_2(s)$ (comparare; ≥ şi = sunt definite
$m_1 \leq m_2 = \forall s \in S: m_1(s) \leq m_2(s)$ în mod analog cu ≤)

(iv) $|m| = \sum_{s \in S} m(s)$ (determinare cardinalitate)

Dacă $|m| = \infty$ se spune că m este **infinit**. Altfel multisetul m este finit. Dacă $m_1 \leq m_2$ se poate defini operaţia de **scădere**, astfel:

(v) $m_2 - m_1 = \sum_{s \in S} (m_2(s) - m_1(s))`s$ (scădere)

Se prezintă mai jos proprietăţile operaţiilor cu multiseturi. Demonstrarea acestora este imediată şi ea nu se prezintă aici. În cele ce urmează N_+ va desemna mulţimea întregilor pozitivi şi se definesc următoarele: $n + \infty = \infty + n = \infty, \forall n \in N \cup \infty$ iar şi $n * \infty = \infty * n = \infty$, $\forall n \in N_+ \cup \{\infty\}$ şi $0 * \infty = \infty * 0 = 0$.

Lema 1: Următoarele ecuaţii sunt satisfăcute pentru orice multist $m, m_1, m_2, m_3 \in S_{SM}$ şi orice întreg nenegativ $n, n_1, n_2, n_3 \in N_+$:

(i) $m_1 + m_2 = m_2 + m_1$ (+ este comutativ);

(ii) $m_1 + (m_2 + m_3) = (m_1 + m_2) + m_3$ (+ este asociativ);

(iii) $m + empty = m$ (empty este element nul);

(iv) $1 * m = m$ (1 este element unitate);

(v) $0 * m = empty$;

(vi) $n * (m_1 + m_2) = (n * m_1) + (n * m_2)$;

(vii) $(n_1 + n_2) * m = (n_1 * m) + (n_2 * m)$;

(viii) $n_1 * (n_2 * m) = (n_1 * n_2) * m$;

(ix) $|m_1 + m_2| = |m_1| + |m_2|$;

(x) $|n * m| = n * |m|$.

Având în vedere proprietăţile (i) ... (iv) de mai sus rezultă $(S_{MS}, +)$ este un grup comutativ. În acest grup se definesc sumele peste multiseturi, astfel:

Definiţia 10-3 (Sume peste multiseturi)

Fie un multiset finit $m \in S_{MS}$, un grup comutativ (R,+) şi o funcţie $F \in [S \to R]$. Se va folosi notaţia:

$$\sum_{s \in m} F(s)$$

pentru a semna **suma** ce conţine ca termeni câte un termen pentru fiecare element al multisetului m. Dacă un element al mulţimii S apare de mai multe ori în multisetul m atunci apare o operaţie de adunare pentru fiecare dintre aceste apariţii.

Ca un exemplu pentru această notaţie: pentru fiecare multiset m, este valabilă egalitatea $m = \sum_{s \in m} s$.

10.3.2 Reţele Petri colorate

Scopul acestei secţiuni este de a da o definiţie formală reţelelor Petri colorate. Acestea vor fi definite prin n-tupluri aşa cum se procedează în cazul grafurilor sau a automatelor finite. Pentru a da o definiţie reţelelor Petri colorate nu este necesar a specifica o sintaxă prin care modelatorul îşi scrie expresiile. Vom presupune doar că o astfel de sintaxă există împreună cu o semantică bine definită astfel încât se poate vorbi în mod univoc, fără ambiguităţi despre următoarele:

- *Elementul unui tip*, T. Mulţimea tuturor elementelor tipului se va nota prin numele T al tipului însăşi.
- *Tipul unei variabile*, v – notat prin Type(v).
- *Tipul unei expresii*, expr – notat prin Type(expr).
- *Mulţimea variabilelor unei expresii*, expr – notat prin Var(expr).
- *O legătură (binding) a variabilelor*, V – obţinută prin asocierea unei valori $b(v) \in Type(v)$ variabilei $v \in V$.
- *Valoarea obţinută prin evaluarea unei expresii*, expr, *pentru o legătură*, b – notată prin expr<b>. Var(expr) trebuie să fie o submulţime a mulţimii variabilelor lui b, şi evaluarea are loc prin înlocuirea fiecărei variabile $v \in Var(expr)$ prin valoarea $b(v) \in Type(v)$ dată de legătura b.

O expresie fără variabile este o ***expresie închisă*** ce poate fi evaluată pentru orice set de valori şi va da întotdeauna aceeaşi valoare - fapt ce va fi notat prin numele expresiei însăşi. Ceea ce înseamă că se va scrie "expr" în locul mult mai pedantului "expr<b>".

Înainte de a da definiţia reţelelor Petri colorate se convin următoarele notaţii:

- *B* va desemna tipul Boolean, având elementele {false, true} şi operaţiile logice uzuale;
- dacă prin Vars s-a notat o mulţime de variabile atunci se convine ca Type(Vars) să desemneze mulţimea tipurilor, astfel: $Type(Vars)=\{Type(v) \mid v \in Vars\}$.

Definiţia 10-4 (Reţea Petri Colorată, CPN)

O **reţea Petri colorată** este un tuplu $CPN=(\Sigma,P,T,A,N,C,G,E,I)$ ce satisface următoarele condiţii:

(i) Σ este o mulţime finită şi nevidă de tipuri, numită **mulţimea seturilor de culori**;

(ii) P este o mulţime finită de **locaţii**;

(iii) T este o mulţime finită de **tranziţii**;

(iv) A este o mulţime finită de **arce** astfel încât: $P \cap T = P \cap A = T \cap A = \emptyset$;

(v) N este o **funcţie nod**. Ea este definită de la A la $P \times T \cup T \times P$; $N: A \rightarrow P \times T \cup T \times P$;

(vi) C este o **funcţie culoare**. Ea este definită de la P la Σ; $C: P \rightarrow \Sigma$;

(vii) G este o **funcţie de gardă**. ea este definită în T având ca valori expresii astfel încât: $t \in T: [Type(G(t))=B \wedge Type(Var(G(t))) \subseteq \Sigma]$.

(viii) E este o **funcţie expresie-arc**. Ea este definită în A, având ca valori expresii, astfel încât: $a \in A: [Type(E(a))=C(p(a))_{MS} \wedge Type(Var(E(a))) \subseteq \Sigma]$, unde p(a) este locaţia funcţiei nod N(a).

(ix) I este o **funcție de inițializare**. Ea este definită de la P la expresii închise astfel încât: $p \in P: [Type(I(p))=C(p)_{MS}]$.

(i) **Mulțimea seturilor de culori** determină tipurile, operațiile și funcțiile ce pot fi utilizate în inscripționarea rețelei. Presupunem că fiecare set de culori are cel puțin un element.

(ii)+(iii)+(iv) **Locațiile, tranzițiile și arcele** sunt descrise prin trei mulțimi P, T și A care trebuie să fie finite și disjuncte.

(v) **Funcția nod**, N, alocă fiecărui arc o pereche de noduri în care primul nod este nodul sursă iar cel de al doilea este nodul destinație. Cele două noduri trebuie să fie de tip diferit. Definiția permite exestența mai multor arce între aceleași două noduri. Definiția permite de asemnea existența nodurilor izolate.

(vi) **Funcția culoare**, C, alocă fiecărei locații, p, un set de culori C(p). În mod intuitiv, acest lucru înseamnă că fiecare marcare a locației p trebuie să aibă o culoare ce aparține tipului C(p).

(vii) **Funcția de gardă**, G, alocă fiecărei tranziții, t, o expresie având tipul boolean, adică un predicat. Pe deasupra, fiecare variabilă a lui G(t) trebuie să fie de tipul care aparține mulțimii Σ. Funcția de gardă poate să și lipsească ceea ce este echivalent cu faptul că funcția de gardă este o expresie închisă având valoarea logică "true".

(viii) **Funcția expresie arc**, E, alocă fiecărui arc, a, o expresie care trebuie să fie de tipul $C(p(a))_{MS}$. Acest lucru înseamnă că fiecare evaluare a expresie trebuie să dea ca rezultat un multiset peste culorile atașate locației legate de arcul dat. Este permis ca arcul să nu aibă nici o expresie caz în care se consideră că aceasta dă ca rezultat multisetul "empty".

(ix) **Funcția de inițializare**, I, alocă fiecărei locații, p, o expresie închisă ce trebuie să fie de tipul $C(p)_{MS}$, adică trebuie să fie un multiset peste C(p). Expresia de inițializare poate să și lipsească.

În continuare prin X se va nota mulțimea tuturor nodurilor, astfel încât: $X=P \cup T$. Deasemenea se vor defini un număr de funcții pentru a descrie relațiile dintre elementele învecinate ale rețelei. Denumirea funcției va indica și domeniul de valabilitate a ei. Spre exemplu, p este o aplicație în mulțimea locațiilor iar A este o aplicație în mulțimea de arce (litera mare indică faptul că domeniul se compune din mulțimi). De multe ori se va folosi aceeași literă pentru a desemna diverse funcții sau mulțimi dar argumentul/argumentele vor clarifica univoc obiectul desemnat. Spre exemplu, mulțimea A sau funcția $A \in [X \rightarrow A_S]$ sau funcția $A \in [P \times T \cup T \times P \rightarrow A_S]$:

$\mathbf{p \in [A \rightarrow P]}$ alocă fiecărui arc, a, locația lui N(a), adică aceea componentă a lui N(a) care este o locație;

$\mathbf{t \in [A \rightarrow T]}$ alocă fiecărui arc, a, tranziția lui N(a), adică aceea componentă a lui N(a) care este o tranziție;

$\mathbf{s \in [A \rightarrow X]}$ alocă fiecărui arc, a, sursa lui a, adică prima componentă a lui N(a);

$\mathbf{d \in [A \rightarrow X]}$ alocă fiecărui arc, a, destinația lui a, adică ceea de-a doua componentă a lui N(a);

$\mathbf{A \in [(P \times T \cup T \times P) \rightarrow A_S]}$ alocă fiecărei perechi de noduri (x_1,x_2) mulțimea arcelor ce le conectează, adică arcele a căror sursă este nodul x_1 și a căror destinație este x_2:

$$A(x_1,x_2)=\{a \in A \mid N(a)=(x_1,x_2)\}$$

$\mathbf{A \in [X \rightarrow A_S]}$ alocă fiecărui nod x mulțimea arcelor conexe, adică mulțimea acelor arce pentru care x este nod sursă sau destinație:

$$A(x)=\{a\in A \mid \exists x^{\bullet}\in X:[N(a)=(x,x^{\bullet})\vee N(a)=(x^{\bullet},x)\}$$

$^{\bullet}x\in[X\rightarrow X_S]$ alocă fiecărui nod x mulţimea nodurilor din premulţimea lui, adică mulţimea nodurilor ce sunt conectate de x printr-un arc de intrare:

$$^{\bullet}x=\{x^{\bullet}\in X \mid \exists a\in A:N(a)=(x^{\bullet},x)\}$$

$x^{\bullet}\in[X\rightarrow X_S]$ alocă fiecărui nod, x, mulţimea nodurilor din postmulţimea lui, adică mulţimea nodurilor ce sunt conectate la x printr-un arc de ieşire:

$$x^{\bullet}=\{x^{\bullet}\in X \mid \exists a\in A:N(a)=(x,x^{\bullet})\}$$

$X\in[X\rightarrow X_S]$ alocă fiecărui nod, x, mulţimea nodurilor ce îl înconjoară, adică mulţimea nodurilor ce sunt conectate de el printr-un arc:

$$X(x)=\{x^{\bullet}\in X \mid \exists a\in A:[N(a)=(x,x^{\bullet})\vee N(a)=(x^{\bullet},x)]\}$$

Toate funcţiile definite mai sus pot fi modificate astfel încât să aibă ca intrare o mulţime. În acest caz ele vor întoarce tot mulţimi ce se vor nota prin majuscule.

10.3.3 Exemplu de modelare prin CPN

Pentru a ilustra noţiunile de mai sus se prezintă următorul exemplu:

Exemplul 10-1:

Un număr de procese concurează un set de resurse. Sistemul este compus din două tipuri de procese (numite p-procese respectiv q-procese) şi din trei tipuri de resurse (r-resurse, s-resurse şi t-resurse). Procesele se consideră a fi ciclice. În cursul execuţiei unui ciclu fiecare proces are nevoie de un anumit număr de resurse pentru utilizare exclusivă.

Se deosebesc patru stări distincte în care se pot găsi p-procesele. Resursele solicitate în aceste stări în ordinea în care sunt accesate sunt după cum urmează:

Starea I. = nu solicită nici o resursă;

Starea II. = solicită două resurse s: 2`s;

Starea III. = solicită două resurse s şi o resursă t: 2`s+1`t;

Starea IV. = solicită două resurse s şi două resurse t: 2`s+2`t;

Procesele q se pot găsi în cinci stări distincte. Resursele solicitate în aceste stări în ordinea în care sunt accesate sunt după cum urmează:

Starea I. = nu solicită nici o resursă;

Starea II. = solicită o resursă r şi o resursă s: 1`r+1`s;

Starea III. = solicită o resursă r şi două resurse s: 1`r+2`s;

Starea IV. = solicită două resurse s: 2`s;

Starea V. = solicită două resurse s şi o resursă t: 2`s+1`t.

În starea iniţială sistemul conţine două procese tip p, trei procese tip q, o resursă de tip r, trei resurse tip s şi două resurse tip t. Toate procesele pornesc din prima stare, în care nu solicită nici o resursă.

Figura 10-8 prezintă CPN care descrie (modelează) funcţionarea acestui sistem.

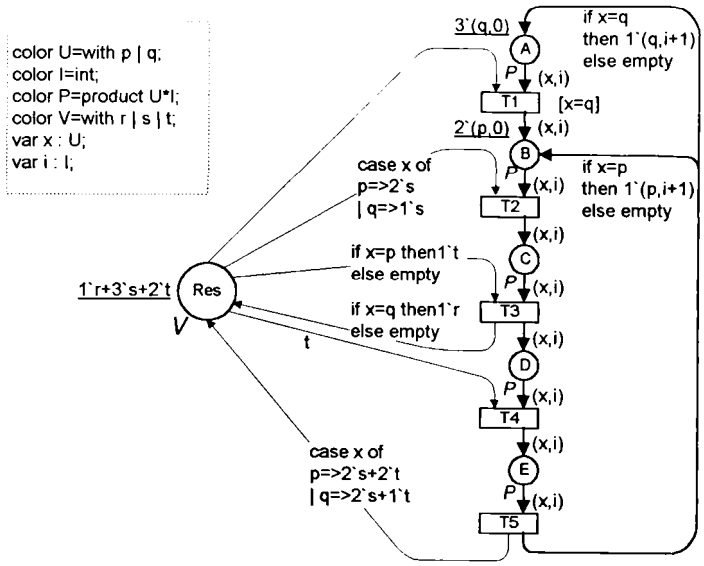


Fig. 10-8 Modelarea procesului de alocare a resurselor printr-o CPN

În Figura 10-8 seturile de culori s-au reprezentat prin majuscule înclinate (italic). Locaţiile A, B, C, D şi E au setul de culori *P* iar locaţia Res, setul de culori *V*.

Figura conţine şi un câmp de declaraţii reprezentat prin dreptunghiul din linii punctate din colţul stânga sus. Din aceste declaraţii se poate vedea că locaţiile A, B, C, D, E posedă un set de culori ce sunt o pereche de elemente ale mulţimii U respectiv I (produs cartezian dintre elemetele mulţimii U şi I). Primul element va preciza procesul, p sau q, iar al doilea element numărul de execuţii al procesului dat.

Setul de culori al locaţiei *V* are trei elemente r, s şi t. Deci o marcare a acestei locaţii poate să fie un multiset peste această mulţime. Expresiile de iniţializare au fost subliniate.

Figura 10-9 prezintă aceeaşi CPN sub forma unor n-tupluri, fără specificarea detaliată a seturilor de culori U, I, P şi V (acestea au fost date în Figura 10-8).

$\sum$ ={U,I,P,E}

P={A,B,C,D,E,Res}

T={T1,T2,T3,T4,T5}

A={A→T1, T1→B, B→T2, T2→C, C→T3, T3→D, D→T4, T4→E, E→T5, T5→A, T5→B, Res→T1, Res→T2, Res→T3, Res→T4, T3→Res, T5→Res}

N(a)=(SURSĂ,DEST) dacă a este în forma SURSĂ→DEST

$$C(p)=\begin{cases} P \text{ daca } p \in \{A,B,C,D,E\} \\ V \text{ altfel} \end{cases}$$

$$G(t)=\begin{cases} x = q \text{ daca } t = T1 \\ \text{true altfel} \end{cases}$$

$$E(a)=\begin{cases} 1`r+1`s & \text{daca } a = Res \rightarrow T1 \\ \text{case } x \text{ of } p \Rightarrow 2`s \mid q \Rightarrow 1`s & \text{daca } a = Res \rightarrow T2 \\ \text{if } x = p \text{ then } 1`t \text{ else empty} & \text{daca } a = Res \rightarrow T3 \\ \text{if } x = q \text{ then } 1`r \text{ else empty} & \text{daca } a = T3 \rightarrow Res \\ t & \text{daca } a = Res \rightarrow T4 \\ \text{case } x \text{ of } p \Rightarrow 2`s+2`t \mid q \Rightarrow 2`s+1`t & \text{daca } a = Res \rightarrow T2 \\ \text{if } x = q \text{ then } 1`(q,i+1) \text{ else empty} & \text{daca } a = T5 \rightarrow A \\ \text{if } x = p \text{ then } 1`(p,i+1) \text{ else empty} & \text{daca } a = T5toB \\ (x,i) & \text{altfel} \end{cases}$$

$$I(p)=\begin{cases} 3`(q,0) & \text{daca } p = A \\ 2`(p,0) & \text{daca } p = B \\ 1`r+3`s+2`t & \text{daca } p = Res \end{cases}$$

Fig. 10-9 Reprezentarea reţelei din Figura 10-8 prin n-tupluri

Din Figura de mai sus se poate vedea că, în lipsa unei reprezentări grafice, forma de prezentare prin n-tupluri este inacceptabilă, (inutilizabilă) pentru un subiect uman.

10.3.4 Sintaxa CPN

S-a definit mai sus structura CPN. În cele ce urmează se va defini comporatmentul CPN pe baza sintaxei reţelei. Se introduc următoarele notaţii:

$$\forall t \in T: Var(t)=\{v \mid v \in Var(G(t)) \vee \exists a \in A: v \in Var(E(a))\}$$

$$\forall (x_1,x_2) \in (P \times T \cup T \times P): E(x_1,x_2)= \sum_{a \in A(x_1,x_2)} E(a)$$

Var(t) se va numi mulţimea variabilelor tranziţiei t iar $E(x_1,x_2)$ se va numi expresia arcului (x_1,x_2). Suma indică operaţia de adunare a expresiilor şi este bine definită deoarece toate expresiile participante la sumă aparţin aceluiaş tip de multiset. Argumentele vor indica întotdeauna, dacă este vorba despre funcţia $E \in [A \rightarrow Expr]$ sau despre funcţia $E \in [(P \times T \cup T \times P) \rightarrow Expr]$.

Este de notat faptul că $A(x_1,x_2)$=empty implică $E(x_1,x_2)$=empty (în acest ultim caz "empty" semnifică o expresie ce are ca rezultat un multiset vid).

Se va defini în continuare ***legătura unei tranziţii***. În mod intuitiv, legătura unei tranziţii se obţine prin substituţia fiecărei variabile a tranziţiei printr-o evaluare a ei. Se impune, ca fiecare culoare să aibă tipul corect iar funcţia de gardă, prin evaluare, să dea valoarea "true". Aşa cum s-a prezentat mai sus G(t)<b> denotă evaluarea expresiei funcţiei de gardă pentru legătura b.

Definiţia 5 (Legătura unei tranziţii)

O legătură a tranziţiei t este o funcţie b definită pe Var(t), astfel încât:

$$\forall v \in Var(t): b(v) \in Type(v)$$

$$G(t)<b>$$

Prin B(t) se notează mulţimea tuturor legăturilor tranziţiei t.

O legătură se scrie sub forma:<$v_1=c_1,v_2=c_2,...v_n=c_n$> unde Var(t)={$v_1,v_2,...v_n$}. Nu se impune vre-o ordonare a elementelor mulțimii Var(t).

Se definesc în continuare noțiunile de : element de marcare, element al legăturii, marcări și pași.

Definiția 10-6

Un element de marcare este o pereche (p,c) unde p∈P și c∈C(p), iar un element al legăturii este o pereche (t,b) unde t∈T și b∈B(t). Mulțimea tuturor elementelor de marcare se notează prin TE iar mulțimea tuturor elementelor unei legături prin BE.

O marcare este un multiset peste TE, iar un pas este un multiset nevid și finit peste BE.

Marcarea inițială M_o este marcarea ce se obține prin evaluarea expresiilor de inițializare:

$$\forall(p,c)\in TE: M_o(p,e)=(I(p))(c)$$

Mulțimea tuturor marcărilor și pașilor se notează prin *M* respectiv *Y*.

Fiecare marcare $M\in TE_{MS}$ determină o funcție univocă M^* definită pe P, astfel încât $M^*(p)\in C(p)_{MS}$:

$$\forall p\in P\forall c\in C(p):(M^*(p))(c)=M(p,c)$$

Pe de altă parte, fiecare funcție M^*, definită pe P astfel încât $M^*(p)\in C(p)_{MS}$ pentru orice p∈P, determină o marcare unică M:

$$\forall(p,c)\in TE: M(p,c)=(M^*(p))(c)$$

În mod analog, există o corespondență unică între un pas Y și o funcție Y^* definită pe T astfel încât $Y^*(t)\in B(t)_{MS}$ este finită pentru orice t∈T și nevidă pentru cel puțin un t∈T.

Spre exemplu, marcarea inițială a CPN din Figura 10-8 poate fi reprezentată fie ca o funcție:

$$M_o(p)=\begin{cases}3`(q,0) & \text{daca } p=A\\ 2`(p,0) & \text{daca } p=B\\ 1`r+3`s+2`t & \text{daca } p=Res\\ \text{empty} & \text{altfel}\end{cases}$$

fie ca un multiset:

$$M_o=3`(A,(q,0))+2`(B,(p,0))+1`(Res,(1`r+3`s+2`t))$$

Pe baza celor de mai sus, poate fi dată definiția formală a concesionării. Prin evaluarea expresiei E(p,t)<b> se obține multisetul de culori extrase din locația p la execuția tranziției t, cu legătura b. Considerând suma tuturor legăturilor (t,b)∈Y, obținem toate marcările extrase din locația p la execuția pasului Y. Acest multiset trebuie să fie mai mic sau egal cu marcarea locației p. Altfel exprimat, fiecare element al setului de valori (t,b)∈Y trebuie să obțină marcările specificate de E(p,t)<b>, fără să fie nevoit a le împărți cu alte elemente ale legăturii din pasul Y. Este de reamintit faptul că, legătura tranziției t satisface funcția de gardă a tranziției t.

Definiția 10-7 (Concesionare)

Pasul Y este concesionat la marcarea M, dacă următoarea condiție este satisfăcută:

$$\forall p\in P:\ \sum_{(t,b)\in Y} E(p,t)<b> \le M(p)$$

Fie pasul Y concesionat la marcarea M. Dacă $(t,b)\in Y$, se spune că tranziţia t este concesionată la marcarea M, pentru setul de valori b. Se spune, deasemenea, că (t,b) este concesionat la M respectiv t este concesionat la M.

Dacă (t_1,b_1), $(t_2,b_2)\in Y$ şi $(t_1,b_1)\neq(t_2,b_2)$, se spune că (t_1,b_1) şi (t_2,b_2) sunt concurent concesionate. În mod analog t_1 şi t_2 sunt concurent concesionate. Dacă $|Y(t)|\geq 2$, se spune că t este concurent concesionat cu sine însuşi. Dacă $Y(t,b)\geq 2$, se spune că (t,b) este concurent concesionat cu sine însuşi.

Dacă un pas este concesionat, el poate fi executat. Prin execuţia pasului se extrag marcări din locaţiile de intrare şi se adună marcări la locaţiile de ieşire a tranziţiilor executate. Numărul şi culoarea marcărilor extrase respectiv transferate sunt determinate de expresiile arcelor, evaluate la legătura concesionată:

Definiţia 10-8

Dacă pasul Y este concesionat la marcarea M_1, el poate fi executat, schimbând marcarea M_1 la o marcare M_2, definită prin:

$$\forall p\in P: M_2(p)=(M_1(p)-\sum_{(t,b)\in Y} E(p,t)<b>)+\sum_{(t,b)\in Y} E(t,p)<b>$$

Prima sumă din partea dreapta a ecuaţiei de mai sus reprezintă marcările extrase, iar cea de a doua marcările transferate. Deasemenea spunem că, M_2 este direct realizat din M_1 prin execuţia pasului Y. Notăm acest fapt astfel $M_1[Y\rangle M_2$.

Execuţia unui pas este un eveniment atomic, indivizibil. Deşi expresia de mai sus presupune efectuarea scăderii înaintea adunării, acest lucru nu înseamnă prezenţa unei marcări intermediare în care marcările extrase s-au scăzut iar cele transferate încă nu s-au depus în locaţii. De notat faptul că, pasul nu trebuie să fie maximal, adică este posibil ca doar unele tranziţii ale pasului să fie executate.

10.3.5 Maşina virtuală de reţea Petri în cadrul CPN

Se va analiza în continuare aplicabilitatea conceptelor maşinii virtuale de reţea Petri în cazul CPN. Figura 10-10 prezintă schema bloc structurală a maşinii virtuale de reţea Petri.

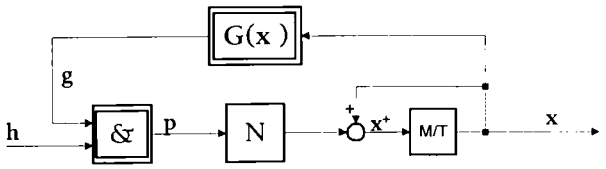


Fig. 10-10 Maşina virtuală de reţea Petri. Schema bloc structurală.

Pentru ca maşina virtuală de reţea Petri să fie utilizabilă şi în cazul CPN trebuie să fie satisfăcute următoarele condiţii:

(i) să fie în mod univoc definită starea sistemului, reprezentată prin vectorul *x*;

(ii) să fie determinabil vectorul de concesionare definit prin elementele sale, $g[j]$, $0<j\leq|T|$, astfel:

$$g[j]=\begin{cases}0 & \text{daca } t_j \text{ nu este concesionata la marcarea data}\\ 1 & \text{daca } t_j \text{ este concesionata la marcarea data}\end{cases};$$

(iii) să fie determinabilă matricea de incidenţe, N, a sistemului;

(iv) sistemul să fie observabil, adică elementele vectorului de execuţie **h** să fie determinabile în procesul condus.

Se vor analiza mai jos aceste condiţii în cazul reţelelor Petri colorate.

Pasul definit mai sus este un pas concesionat în sensul că nu neapărat va fi şi executat. Definiţia 10-6 nici nu a impus ca pasul să fie maximal. În Figura 10-10 pasul **p** este un pas executat în sensul că va conţine tranziţii executate (elementele nenule reprezintă tranziţii executate). Apariţia unei valori nenule a acestui vector, **p**[i]≠0, este un eveniment, la nivelul maşinii virtuale de reţea Petri, care semnifică execuţia unui pas. Indexul *k* este de fapt numărul de ordine al ultimului pas executat.

(i) Prin x s-a notat vectorul de stare k. Elementele sale sunt specificate astfel:

$x[i]=M(p_i)$

(ii) În cazul CPN elementele vectorului de concesionare la pasul k se determină astfel:

a.) pentru cazul regulei de tranziţie slabe:

$$g[j]=\begin{cases}1 & \text{daca } (G(t_j)=true)\wedge(\forall p\in {}^\bullet t_j : E(p,t_j)\le M(p))\\ 0 & \text{altfel}\end{cases}$$

b.) pentru cazul regulei de tranziţie tari:

$$g(j)=\begin{cases}1 & \text{daca } (G(t_j)=true)\wedge(\forall p\in {}^\bullet t_j : E(p,t_j)\le M(p))\wedge(\forall p\in t_j^\bullet : E(t_j,p)\le K-M(p))\\ 0 & \text{altfel}\end{cases}$$

Predicatele tranziţiei, reprezentate prin vectorul **h**, sunt distincte de funcţiile de gardă. Acestea din urmă sunt elemente sintactice, de modelare, ale CPN, pe când predicatele tranziţiei reflectă evenimentele din procesul condus.

Prin operaţia de conjuncţie logică se obţine pasul **p**, al tranziţiilor executate. Pe baza acestuia, a structurii reţelei, a sintaxei reţelei şi a stării actuale trebuie determinată noua stare.

(iii) Matricea de incidenţe, în acest caz, se defineşte prin elementele sale astfel:

$$N[i\,j]=\begin{cases}0 & \text{daca } (t_j p_i)\cup(p_i t_j)=\varnothing\\ E(t_j,p_i) & \text{daca } t_j\in {}^\bullet p_i\\ -E(p_i,t_j) & \text{daca } p_i\in {}^\bullet t_j\\ E(t_j,p_i)-E(p_i,t_j) & \text{daca } (t_j\in {}^\bullet p_i)\wedge(p_i\in {}^\bullet t_j)\end{cases}$$

Conform Definiţiei 10-8:

$$M^+(p_i)=M(p_i)-\sum_{(t\in p_i^\bullet)\wedge G(t)=1}(E(p_i,t)<b>+\sum_{(t\in {}^\bullet p_i)\wedge G(t)=1}E(t,p_i)<b>$$

Sumele din ecuaţia de mai sus se pot obţine prin produsul vectorial: N[i]·p. Prin urmare:

$M^+(p_i)=M(p_i)+N[i]\cdot p$

de unde rezultă:

$x^+=x+N\cdot p$

De notat faptul că expresiile E(p,t) şi E(t,p) din matricea de incidenţe se consideră a fi evaluate la legătura rezultată din starea x (M(p),p∈P). De fapt aceste expresii se evaluează oricum în cursul determinării vectorului de concesionare.

(iv) Se va considera că toate evenimentele în sistemul controlat sunt observabile la nivelul de detaliere al modelului de reţea Petri. Acest lucru înseamnă că sunt disponibile elementele vectorului de execuţie **h**.

În concluzie, maşina virtuală de reţea Petri este utilizabilă şi în cazul CPN.

10.3.6 Modelul CPN al staţiei de montaj

Modelul CPN al staţiei de montaj se prezintă în Figura 10-11.

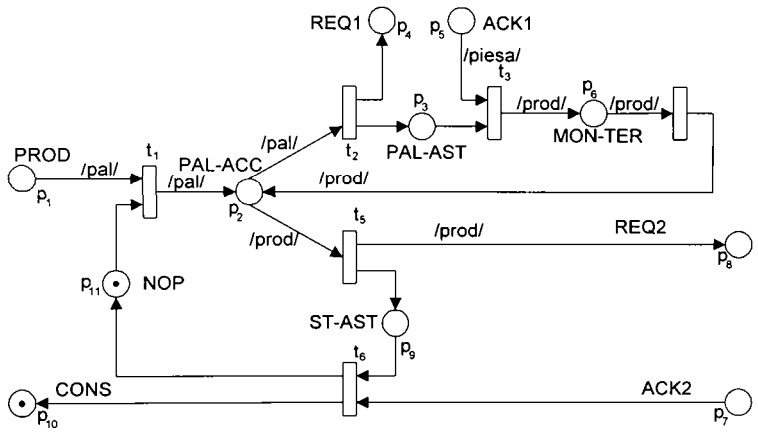


Fig. 10-11 Modelul CPN al staţiei de montaj

În acest caz /pal/ poate să însemne diverse tipuri de piese respectiv piese în diverse stadii de prelucrare. La fel /piesa/ poate să fie de diverse tipuri respectiv produsul /prod/ rezultat în urma montajului poate să fie de diferite tipuri.

Staţia de montaj primeşte şi funcţiuni noi, faţă de cea prezentată în Fig. 10-5, datorită expresiilor arcelor (p_2,t_2), respectiv (p_2,t_5). Tranziţia t_2 va fi concesionată prin expresia /mon/ doar dacă paleta din locaţia p_2 conţine o piesă destinată montajului. În cazul în care ajunge aici un produs, /prod/, rezultat din operaţia de montare, şi care trebuie evacuat din staţia de montaj, tranziţia t_5 va fi concesionată prin expresia /evac/.

Privit strict din punctul de vedere al montajului, acest test suplimentar pare inutil. Având însă în vedere faptul că, paleta /pal/ poate accepta diverse piese, această locaţie de test poate constitui un buffer suplimentar, ceea ce creşte flexibilitatea staţiei, acest buffer fiind exploatabil în cele mai diverse situaţii.

În ceea ce priveşte instalaţia de transport aceasta are un caracter universal. Gama de obiecte transportabile se va specifica, în cazul modelului CPN, prin expresia /x/.

10.4 Exemplul unei linii de montaj

Pentru a ilustra utilizarea modelelor prezentate mai sus se va elabora modelul unei linii de montaj formate din trei staţii de montaj, o presă şi o bandă transportoare (Figura 10-12).

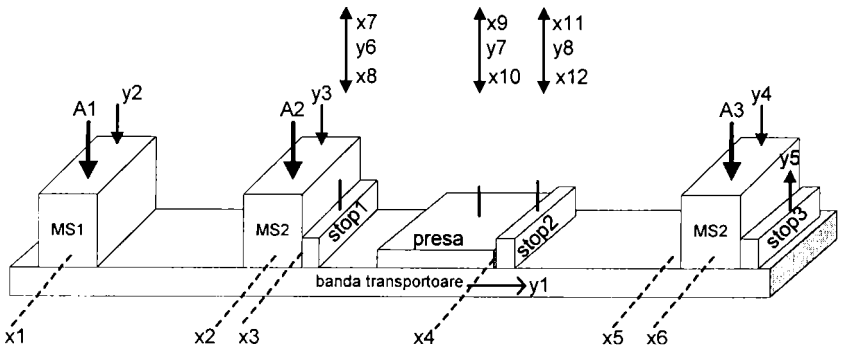


Fig. 10-12 Structura constructivă a liniei de montaj din exemplul prezentat

Staţiile de montaj MS1, MS2, MS3 sunt alimentate prin punctele de alimentare A1, A2, A3. După efectuarea operaţiei de montaj aferente, semifabricatele sunt preluate de banda transportoare.

MS1 fixează de o paletă o piesă de bază (P1). Aceasta este preluată de bandă şi dusă la staţia MS2 unde, peste piesa de bază se montează o a doua piesă (P2). După montare, ansamblul este dus la o presă pentru a rigidiza cele două piese. În MS3 ansamblul este împachetat în ambalajul preluat de MS3.

În Figura 10-12 y_j reprezintă semnalele de comandă emise de către echipamentele de comandă către elementele de execuţie iar x_i reprezintă semnalele furnizate de către senzorii cu care s-a echipat linia de montaj.

Sistemul trebuie să satisfacă următoarele cerinţe minime de flexibilitate:

1. Sistemul trebuie să fie apt a produce trei tipuri diferite de produse.
   1.1. TIP1 = produs complet constând din P1, P2, presat, împachetat;
   1.2. TIP2 = produs constând doar din P1, nepresat, împachetat;
   1.3. TIP3 = produs constând doar din P2, nepresat, împachetat.
2. Sistemul să accepte comenzi cu diverse cantităţi pentru fiecare dintre aceste tipuri.
3. Sistemul să accepte prescrierea de priorităţi în efectuarea comenzilor. Dacă condiţiile pentru execuţia unei comenzi cu prioritate ridicată nu sunt satisfăcute (îndeplinite), sistemul să treacă automat la comanda cu următoarea prioritate. În cazul de faţă se presupune că TIP1 are prioritatea cea mai mare iar TIP3 prioritatea cea mai mică.
4. Produsul asamblat să poată fi trecut într-o staţie de verificare, neprezentată aici, şi la nevoie să fie din nou presată.

Modelul de reţea Petri al sistemului se prezintă în Figura 10-13, inserat la sfârşitul acestui capitol.

Tranziţiile TIP1, TIP2 respectiv TIP3 fac posibilă selecţia tipului de comandă ce se execută. A1 reprezintă echipamentul de încărcare a piesei de bază, (P1), iar A2 reprezintă echipamentul de încărcare (alimentare) cu piesa ce se montează, (P2). Procesul de transport între staţiile de montaj, MS1, MS2, MS3, staţia de ambalare, AMB, respectiv staţia de descărcare, DESC, s-a modelat conform Figurii 10-13. Echipamentele STOP1, STOP2, STOP3 sunt parte componentă a instalaţiei de transport.

Echipamentul de alimentare A3 şi modulul de ambalare AMB sunt astfel conectate încât ambalajul ajunge în locul ambalării înaintea produsului finit, deoarece ambalajul este solicitat înaintea acestuia, prin interfaţa PROD-CONS.

În continuare vom presupune că fiecare staţie de lucru este controlată printr-un echipament de comandă propriu. Întregul sistem este condus printr-un sistem de comandă de coordonare ce va controla şi instalaţia de transport (Figura 10-14).

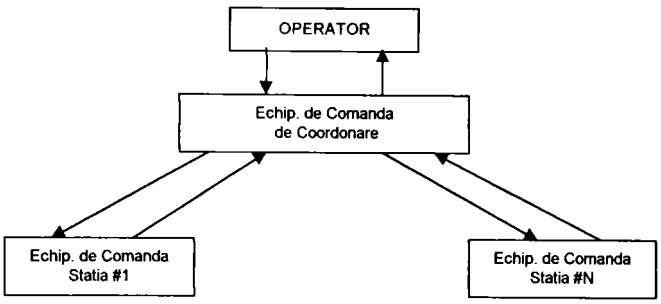


Fig. 10-14 Structura ierarhică a sistemului de conducere

Pentru a putea implementa acest sistem de conducere trebuie specificate şi implementate atât echipamentele de comandă cât şi interfeţele dintre ele. În acest sens trebuie găsit răspuns la următoarele întrebări:

1. Cum se analizează un astfel de model?
2. Care este structura reţelelor Petri pe baza cărora se implementează diversele sisteme de comandă?

10.5 Analiza reţelelor Petri modulare

10.5.1 Sistemul de comandă de coordonare

Următoarele două observaţii fac posibilă determinarea reţelei Petri ce modelează sistemul din punctul de vedere al sistemului de comandă de coordonare şi care este analizabilă prin metodele cunoscute:

1. Calităţile modelului sunt determinate în mod direct de calităţile modulelor componente. Acestea sunt astfel concepute încât să fie viabile deoarece, în caz contrar, ar fi inutilizabile într-un sistem de producţie.
2. Modelul de reţea Petri al staţiilor de lucru a fost astfel conceput, încât locaţiile de interfaţă să nu reprezinte resurse partajate. Având în vedere considerentele prezentate în capitolul 6 şi capitolul 9, acest lucru permite ca, în modelul de reţea Petri al sistemului de coordonare, o staţie de montaj să fie echivalată conform Figurii 10-15.

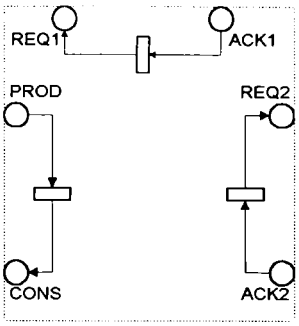


Fig. 10-15 Echivalarea modelului de reţea Petri al unei staţii de montaj

Înlocuind modulele cu această structură se obţine, pentru sistemul de coordonare, modelul de reţea Petri din Figura 10-16, inserat la sfârşitul acestui capitol. Prin analiza acestei reţele se determină calităţile dinamice ale sistemului la nivelul de coordonare.

10.5.2 Analiza modelului de reţea Petri al modulelor componente

În cazul modulelor, analiza se va realiza pe baza reţelei obţinute după înlocuirea reţelei de coordonare printr-o structură echivalentă de subreţea, aşa cum s-a prezentat în capitolul 9. Această alegere este justificată de faptul că reţeaua de conducere va exploata în mod raţional toate capabilităţile funcţionale ale modulelor, ceea ce se traduce prin faptul că tranziţiile externe modulelor vor fi executate conform aceleiaşi raţionalităţi.

Prin această înlocuire, modelul de reţea Petri pentru analiza funcţională a staţiei de montaj, devine cea din Figura 10-17.

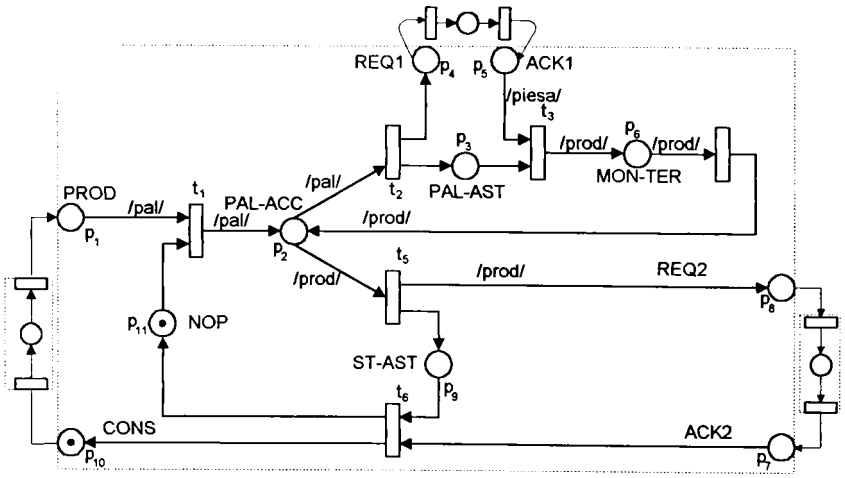


Fig. 10-17 Reţea Petri echivalentă pentru modelarea staţiei de montaj

În situaţia în care se doreşte doar efectuarea unei analize calitativ, mediul extern interfeţelor poate fi modelat printr-o singură tranziţie deoarece locaţiile de interfaţare nu reprezintă resurse partajate (capitolul 9).

10.5.3 Implementarea echipamentelor de comandă

În determinarea modulelor de reţea Petri, pe baza cărora se implementează sistemul de conducere se vor avea în vedere principiile prezentate în capitolul 8. În acest caz, interfeţele trebuie să includă tranziţii efective ale modulelor. Pentru a ilustra acest lucru, Figura 10-18 prezintă modelul staţiei de montaj MS1 împreună cu elementele de reţea la care se conectează.

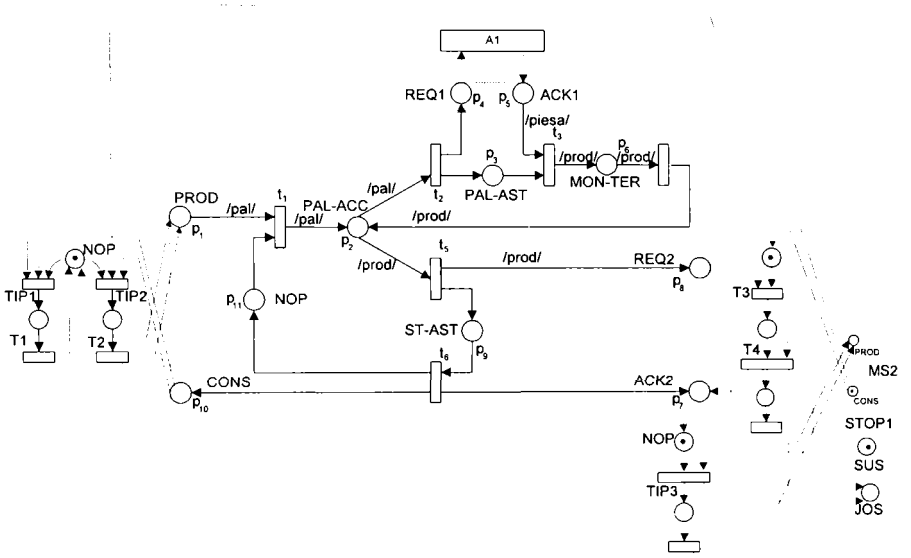


Fig. 10-18 Model de reţea Petri pentru determinarea domeniilor de interfaţare ale staţiei de montaj

Domeniile de interfaţare, în sensul definiţiilor din capitolul 8, sunt sunt după cum urmează:

$N_{1a}=\{(t_6, p_{10}, TIP1), (t_6, p_{10}, TIP2)\}$

$N_{1b}=\{(t_5, p_8, T_3)\}$

$N_{a1}=\{(T_1, p_1, t_1), (T_2, p_1, t_1)\}$

$N_{b1}=\{(T_4, p_7, T_6)\}$

În acelaş fel se va proceda în cazul interfaţării cu instalaţia de alimentare A1.

Tranziţiile din aceste domenii sunt tranziţii concrete din modelul iniţial şi astfel este posibilă interfaţarea conform principiilor prezentate în capitolul 8. Figura 10-19 prezintă modelul de reţea Petri pe baza căruia se va implementa echipamentul de comandă a staţiei de montaj.

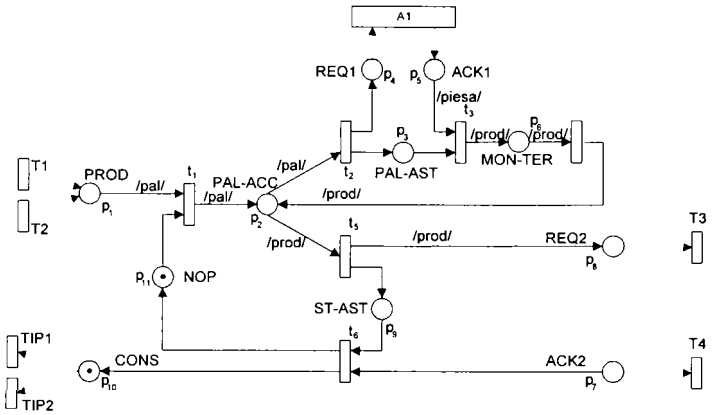


Fig. 10-19 Modelul de reţea Petri pe baza căruia se va implementa echipementul de comandă al staţiei de montaj

10.5.4 Structura de conducere pentru implementarea echipamentelor de comandă

Structura de conducere pentru implementarea echipamentelor de comandă este identică cu ce prezentată în capitolul 8 pentru cazul controlului distribuit şi se prezintă în Figura 10-20.

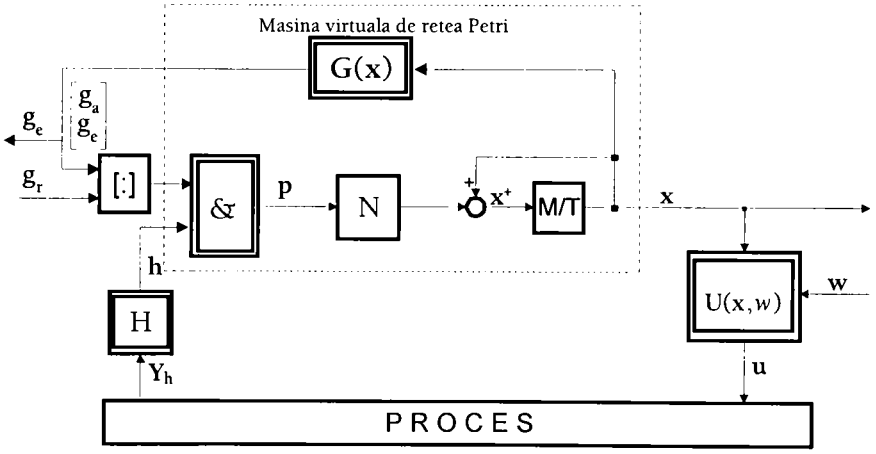


Fig.10-20 Structura de conducere pentru implementarea echipamentelor de comandă

Elementele componente ale vectorului de concesionare g_a, g_e, g_r, au semnificaţia definită în capitolul 8. Strategia de coordonare este încorporată în modulul U(*x*,*w*). Vectorul de comandă *w* va specifica tipul, cantitatea şi prioritatea de execuţie a produselor.

10.6 Concluzii

Modularitatea este o caracteristică de bază a sistemelor flexibile de fabricație. Această modularitate trebuie să se reflecte și asupra modelului de rețea Petri al sistemului.

În paragraful 10-2 s-au prezentat, prin module și interfețe tipice, conceptele modularizării și a interfațării dintre module în cazul modelării prin rețele Petri. S-au prezentat trei tipuri de interfețe și două tipuri de module.

Pentru interfațarea dintre modulele de rețea Petri este necesară o structură de rețea care să asigure:

1. Delimitarea clară dintre elementele componente ale modulului și cele ale interfeței;
2. Modelarea întârzierii introduse de procesul de transport (al informației sau substanței).
3. Realizarea domeniilor de interfatare, conform capitolului 8, astfel încât, în cazul unui control distribuit, acestea să fie chiar interfețele naturale dintre modul și sistemul de transport;

Pentru a asigura flexibilitatea de adaptare a sistemului, modulele componente trebuie să ofere o gamă cât mai largă de funcțiuni pe o structură dată. Modelarea unor astfel de module este în mod eficient realizabilă prin rețele Petri colorate. Din acest motiv, în acest capitol, s-a definit sructura și sintaxa acestui tip de rețea Petri.

Conceptele mașinii virtuale de rețea Petri s-au dovedit deosebit de utile în implementarea echipamentelor de comandă pentru sistemele modelate. În paragraful 10.3.5 s-a demonstrat că aceste concepte sunt utilizabile și în cazul rețelelor Petri colorate.

Pentru ilustrarea conceptelor introduse s-a prezentat exemplul unei linii de montaj. Aceasta realizează funcțiunile impuse prin coordonarea activităților modulelor componente. Necesitatea coordonării impune o structură ierarhică sistemului de conducere.

În paragraful 10.5 se prezintă structurile de rețea pe baza cărora poate fi efectuată analiza comportării dinamice a modulelor respectiv a sistemului de coordonare. Analiza poate să fie "modulară" cu condiția respectării cu strictețe a principiilor încapsulării resurselor partajate. Altfel exprimat, pentru a obține rezultate concludente în urma analizei independente a modulelor este necesară respectarea cu strictețe, în cursul elaborării modelului, a principiului încapsulării resurselor partajate.

Rețele Petri, care stau la baza implementării echipamentelor de comandă, trebuie să conțină domenii de interfațare bazate pe tranziții efective, a căror execuție este observabilă, și nu tranziții echivalente ca în cazul analizei. Tehnica de interfațare elaborată pentru controlul distribuit este și în acest caz utilizabilă.

Pentru a obține o implementare modulară, structura de conducere din Fig. 10-20, care stă la baza implementării echipamentelor de comandă, trebuie să specifice componentele de interfațare ale vectorului de concesionare pe baza domeniilor de interfațare. Acestea sunt definite pe baza modelelor de rețea Petri ale modulelor componente și a sistemului de coordonare.

* * *

Față de literatura de specialitate la care autorul a avut acces, acest capitol prezintă următoarele elemente originale:

a) Definirea unei structuri de interfațare dintre modulele de rețea Petri.

b) Demonstrarea faptului că, conceptul de mașină virtuală de rețea Petri este utilizabil și în cazul modelării prin rețele Petri colorate.

c) Stabilirea condițiilor în care un model modular de rețea Petri poate fi analizat tot "modular" adică stabilirea condițiilor în care o analiză pe module va da rezultate concludente cu privire la funcționalitatea întregului sistem.

d) Specificarea condițiilor în care sistemul de conducere de coordonare poate fi modelat printr-o rețea Petri modulară.

e) Definirea structurii de rețea pentru implementarea sistemului de conducere de coordonare.

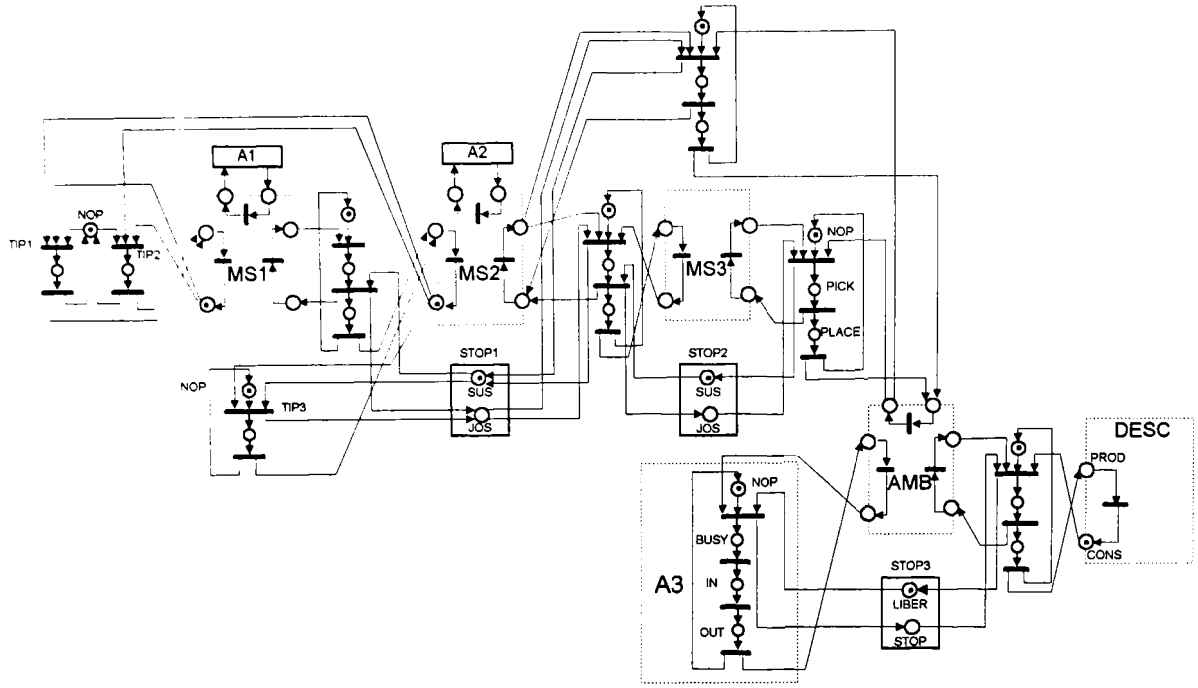


Fig. 10-16 Modelul de rețea Petri al sistemului la nivelul de coordonare

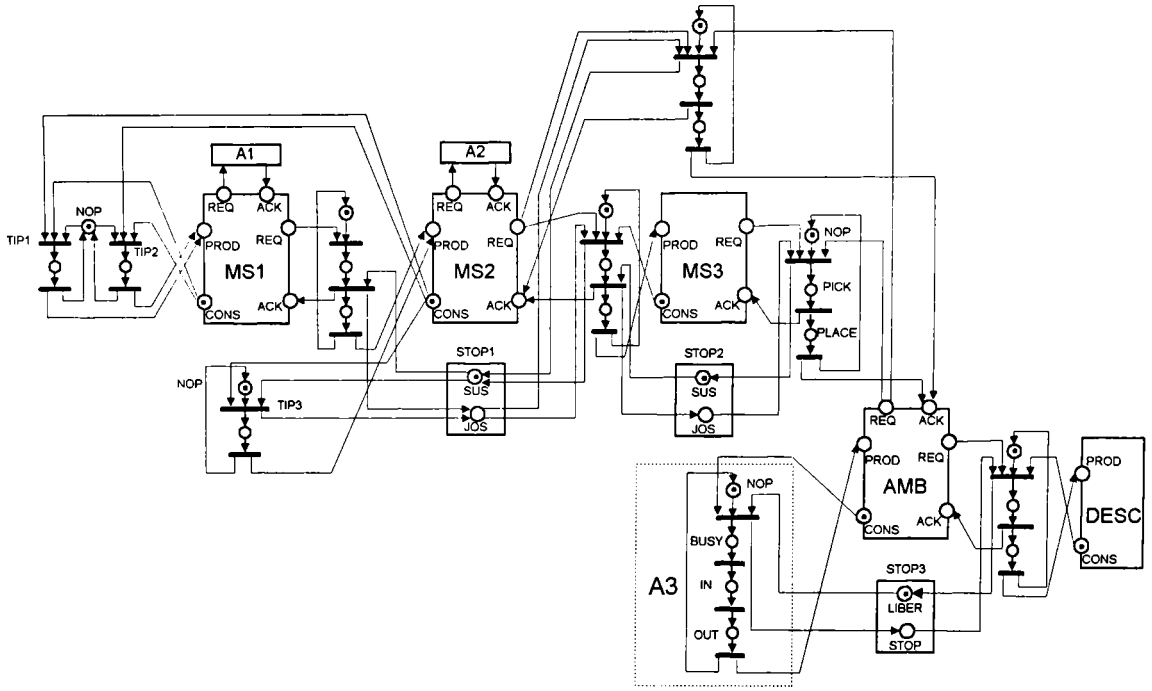


Fig. 10-13 Modelul de rețea Petri al liniei de transport

11. CONCLUZII ŞI CONTRIBUŢII

Acest capitol prezintă o sinteză restrospectivă a conţinutului lucrării precum şi concluziile generale ce se desprind. Deasemnea prezintă contribuţiile originale ale autorului şi consideraţii asupra posibilităţilor de continuare a ideilor prezentate în lucrare.

11.1 Sinteza lucrării

Constructorii sistemelor de automatizări industriale moderne trebuie să creeze sisteme de înaltă performanţă ce vor evolua în scenarii insuficient cunoscute deorece investitorul nu poate preciza în mod obiectiv toate misiunile sistemului pe toată durata de viaţa a acestuia. Constructorii încearcă să facă faţă acestor cerinţe contradictorii prin crearea de sisteme caracterizate prin complexitate structurală deosebită, flexibilitate şi distribuire.

Dezvoltarea spectaculoasă a tehnologiei informatice face posibilă construirea de echipamente inteligente apte de prelucrări informaţionale sofisticate şi de comunicaţii în mai multe direcţii şi cu mai multe niveluri ierarhice fiind apte de a realiza diverse funcţionalităţi. Astfel de echipamente pot sta la baza sistemelor de automatizări industrale moderne cu condiţia ca proiectantul sistemului să dispună de metodele, procedeele şi instrumentele necesare.

Pentru stăpânirea complexităţii structurale a sistemelor, lucrarea de faţă propune următoarele trei principii de structurare, bazate pe noţiuni de modelare semantică:

1. ***principiul decompoziţiei*** respectiv modelarea orientată pe obiecte, prin care sistemele şi subsistemele se divizează în părţi componente;
2. ***principiul abstractizării funcţionale***, prin care se realizează structurarea pe niveluri ierarhice a sistemelor;
3. ***principiul transformării*** sau modelarea orientată pe evenimente, prin care se descriu transformările de stare.

Lucrarea propune ca sistemele de automatizări industriale complexe să fie concepute, modelate, analizate şi implementate ca sisteme dinamice cu stări discrete pilotate de evenimente. Reţelele Petri oferă formalismul matematic necesar modelării şi analizei unor astfel de sisteme. Lucrarea prezintă, într-o formă inginerească, noţiunile de bază precum şi tehnicile de modelare şi analiză pe baza reţelelor Petri.

Lucrarea introduce noţiunile de *operator de tranziţie* şi *maşină virtuală de reţea Petri* şi demonstrează faptul că pentru a putea elabora mărimi de comandă în concordanţă cu relaţiile cauzale din procesul condus, orice sistem de conducere, realizat pe baza unui model de reţea Petri, trebuie să implementeze o maşină virtuală de reţea Petri.

Conceptul de maşină virtuală de reţea Petri permite structurarea clară a sistemului de conducere punând în evidenţă şi delimitând elementele funcţionale componente: (i) maşina virtuală de reţea Petri, (ii) observatorii de evenimente, (iii) strategia de conducere pe baza reacţiei după stare, şi (iv) operatorii de tranziţie.

Pe baza noţiunii de maşină virtuală de reţea Petri lucrarea propune soluţii pentru următoarele categorii de probleme:

1) Sinteza strategiei de conducere pentru cazul în care procesul este condus pe baza reacţiei după stare. Sinteza se realizează pe baza unor restricţii asupra stărilor realizabile ale procesului condus respectiv asupra marcajelor accesibile ale modelului de reţea Petri al procesului condus.

2) Definirea interfeţelor dintre subreţelele rezultate prin partiţionarea unei reţele Petri. Definiţia dată interfeţelor permite specificarea structurilor de conducere ale subreţelelor astfel încât controlul distribuit rezultat să se execute pe baza relaţiilor cauzale din procesul condus.

3) Definirea structurilor de conducere în cazul unei structuri ierarhice a modelului de reţea Petri al procesului condus respectiv a unei structuri ierarhice a sistemului de conducere.

4) Definirea interfeţelor şi a structurilor de conducere în cazul unor reţele Petri modulare.

Cele ce urmează detaliază, pe capitole, contribuţiile şi concluziile lucrării.

11.2 Concluzii şi contribuţii

Capitolul 1:

Concluzii

Creşterea economică este însoţită de crearea şi extinderea sistemelor artificiale mari. Acestor sisteme li se impun cerinţe de înaltă performanţă pentru ca să poată evolua în scenarii insuficient cunoscute, realizând însă nivele prescrise pentru indicatorii de performanţă.

Dimensiunea principală a sistemelor mari, care încearcă să răspundă acestor cerinţe, este *complexitatea structurală*, caracterizată, în special, prin calitatea şi numărul legăturilor dintre elementele componente.

Sistemele moderne de producţie încearcă să facă faţă cerinţelor ce li se impun prin *flexibilitate*. Sistemele flexibile de fabricaţie sunt acele sisteme tehnice care asigură automatizarea fabricaţiei de serie datorită faptului că structura sistemului dispune *de calitatea de integrabilitate*, posedă *adaptabilitate* faţă de un domeniu de sarcini, este *adecvabilă tehnic şi economic* fiecărei sarcini şi este construită pe baza unei *concepţii dinamice*.

Flexibilizarea sistemelor de automatizări industriale este posibilă datorită dezvoltării spectaculoase a tehnologiei informatice. Microprocesoarele sunt utilizate în practic toate echipamentele şi aparatele componente ale sistemelor moderne de automatizare. Aceste echipamente inteligente sunt apte de prelucrări informaţionale sofisticate şi de comuncaţii în mai multe direcţii şi cu mai multe niveluri ierarhice fiind apte de a realiza diverse funcţionalităţi. Pe baza unor astfel de echipamente se realizează sisteme cu prelucrare distribuită sau pe scurt *sisteme distribuite*.

Sistemele distribuite se cracterizează prin:

1) *Multitudinea resurselor fizice şi logice*,

2) *Distribuirea fizică a componentelor sistemului*;.

3) *Viziunea globală, unitară asupra sistemului;*

4) *Independenţa relativă, cooperatistă a sistemului;*

5) *Transparenţa structurii fată de interfaţa de utilizare a acestuia.*

Într-un sistem distribuit funcţionarea unei multitudini de componente impune necesitatea existenţei unei strategii unificatoare globale, care să integreze componentele fizice şi logice distribuite, într-un întreg operaţional. O astfel de strategie trebuie materializată prin componente logice de control, care la rândul lor trebuie să dispună de informaţii de stare necesare. ***Este necesar un model de reprezentare a evoluţiei unui astfel de sistem în timp.*** Aceste modele se bazează pe ipoteza că este posibilă descrierea completă a stării in anumite momente determinate. O astfel de stare este numită punct observabil, evoluţia sistemului materializându-se prin parcurgerea punctelor observabile. Un eveniment este atunci definit printr-o modificare a unei submulţimi a stării şi poate fi datat printr-un punct observabil.

Pentru realizarea efectivă a distribuirii şi flexibilităţii sistemului de automatizare constructorul acestuia trebuie să dispună de metode de specificare, modelare, analiză şi sinteză care să permită o inginerie efectivă a acestor sisteme. Aceste metode trebuie să sprijine în mod efectiv comunicaţia dintre investitor, proiectant şi operator. Modelarea semantică orientată pe evenimente şi formalismul reţelelor Petri pot constitui baza unei metodologii care, împreună cu tehnicile clasice înrădăcinate, să ofere instrumentele de inginerie necesare în realizarea sistemelor de automatizări industriale.

Capitolul 2:

Concluzii

1. Automatizarea industrială ca disciplină tehnică multidisciplinară, trebuie să determine cele mai adecvate soluţii, metode, procedee şi instrumente care, aplicate corect, asigură condiţii optime de rentabilitate şi eficienţă la realizarea respectiv exploatarea unui sistem tehnic.

Automatizarea industrială se confruntă atât cu complexitatea sistemului ce se realizează cât şi cu diversitatea punctelor de vedere asupra acestuia.

La confruntarea cu sisteme mari, complexe se încearcă descoperirea structurii lor (analiză) cu scopul de a exploata caracteristicile acestei structuri în definirea unei căi de a controla sistemul (sinteză).

Sistemele mari, complexe nu pot fi stăpânite fără o viziune sistemică asupra lor. Această viziune sistemică trebuie să se bazeze pe axiomele-sistem prezentate în secţiunea 2.1.2.

2. Studiul sistemelor reale, naturale sau tehnice, se realizează pe baza unui model. Modelul este o imagine idealizată şi esenţializată a fenomenelor reale elaborată prin sistematizarea rezultatelor măsurătorilor şi cunoaşterea legilor generale ale naturii.

Datorită punctelor de vedere diferite, egal îndreptăţite, asupra sistemului modelat este necesară elaborarea de modele diferite, cu caracteristici diferite. Aceste modele se deosebesc în primul rând prin principiul pe baza căreia se realizează structurarea sistemului respectiv a modelului elaborat. Autorul a propus trei astfel de principii de structurare:

a) principiul decompoziţiei respectiv modelarea orientată pe obiecte, prin care sistemele şi subsistemele se divizează în părţi componente (secţiuea 2.2.2);

b) principiul abstractizării funcţionale, prin care se realizează structurarea pe niveluri ierarhice a sistemului (secţiunea 2.2.3);

c) principiul transformării sau modelarea orientată pe evenimente, prin care se descriu transformările de stare (secţiunea 2.2.4).

3. Modelarea se realizează la nivel conceptual în sensul că modelul este o descriere formală a tuturor aspectelor relevante ale sistemului modelat făcând abstracţie de orice aspecte legate de implementare. Noţiunile de bază ale modelării conceptuale s-au definit în standardul ISO 82, [ISO'82].

Contribuţii

(i) Prin conţinut şi formă de prezentare secţiunea 2.1.2. "Axiome-sistem" reprezintă un element de originalitate, autorul, în literatura de specialitate la care a avut acces, găsind doar elemente disparate ale acestei tematici.

(ii) Cu toate că multe elemente ale principiilor prezentate la punctul 2 s-au regăsit în literatura de specialitate accesibilă autorului, forma unitară şi conţinutul sintetic prezentat în acest capitol este o contribuţie originală a autorului.

(iii) Având în vedere importanţa deosebită a noţiunii de "obiect" în modelarea conceptuală, autorul a considerat necesar să dea, în secţiunea 2.2.1.1. o definiţie mai precisă a acestei noţiuni, faţă de ISO 82.

(iv) Pe baza relaţiilor de bază utilizate în modelul semantic orientat pe obiecte, autorul defineşte, drept exemple: (i) modelul informaţional al unui reactor cu agitator, (ii) o reţea de clase de operaţii tehnologice elementare, (iii) ierarhia de clase a unei reţete de producţie.

Capitolul 3:

Concluzii

Modelarea discretă, orientată pe evenimente a circuitelor logice de mare viteză sau a sistemelor distribuite complexe trebuie să ţină cont de efectele relativiste ce apar în aceste sisteme. Modelarea realistă a acestora nu poate fi realizată pe baza ipotezei unei ordonări totale, obiective independentă de observator. Modelarea trebuie să aibă la bază relaţiile cauzale care guvernează sistemul.

S-au definit două relaţii dintre evenimente care pot sta la baza modelării orientate pe evenimente a sistemelor distribuite. Aceste două relaţii sunt: relaţia de ordonare parţială, (notată în prezenta lucrare prin "*pre*") şi relaţia de concurenţă, netranzitivă dar reflexivă (notată în prezenta lucrare prin "*co*"). Aceste relaţii permit definirea şi studiul mulţimilor parţial ordonate respectiv a structurilor de evenimente.

Pentru modelarea discretă, orientată pe evenimente a sistemelor reale trebuie luate în considerare şi condiţiile în care pot apare evenimentele repectiv efectul producerii acestora asupra evoluţiei sistemului. Modelul care încorporează condiţii şi evenimente se prezintă sub forma unor reţele bipartite, orientate.

Teoria acestor reţele este o teorie a modelării discrete, relativiste, orientată pe evenimente. Elementele reţelei sunt condiţii şi evenimente. În cadrul reţelei condiţiile sunt încadrate doar de evenimente iar evenimentele doar de condiţii.

S-au elaborat, şi în literatura de specialitate se prezintă, diverse tipuri de astfel de reţele care, de cele mai multe ori, sunt referite cu denumirea generică de reţele Petri. Acest capitol a tratat, într-o formă inginerească, adoptată nevoilor modelării în domeniul automatizărilor industriale, clasa reţelelor Petri Condiţii/Evenimente şi Locaţie/Tranziţie.

Contribuţii

Forma de prezentare aleasă în acest capitol clarifică noţiunile fundamentale legate de modelarea discretă, orientată pe evenimente a sistemelor distribuite şi implicit clarifică bazele conceptuale ale teoriei şi modelării prin reţele Petri. În acelaş timp prezintă teoria reţelelor Petri într-o formă inginerească care poate sta la baza utilizării efective a acestora în modelarea şi realizarea sistemelor de automatizare.

Capitolul 4:

Concluzii

Analiza mPTN poate fi realizată pe două căi fundamental diferite.

(i) *Analiza bazată pe graful accesibil* presupune construirea, pe baza mulţimii accesibile, a grafului accesibil şi studiul structurii acestuia. Structurarea se realizează pe baza noţiunilor de *graf cu coeziune puternică*, *subgraf cu coeziune maximală* şi *condensata grafului accesibil*. Pe baza condensatei grafului accesibil se poate decide dacă reţeaua este reversibilă, viabilă sau dacă prezintă doar o viabilitate slabă.

(ii) *Analiza bazată pe invarianţi* furnizează informaţii referitoare la comportamentul dinamic al reţelei, prin calcule efectuate pe baza matricei de incidenţe a reţelei. Pe baza T-invarianţilor se pot formula condiţii necesare pentru reversibilitatea şi viabilitatea reţelei. Pe baza P-invarianţilor se pot formula condiţii suficiente pentru mărginire. Pe baza invarianţilor **nu** se pot formula condiţii necesare **şi** suficiente.

Contribuţii

S-a realizat o sinteză a principalelor metode de analiză a reţelelor Petri. S-au ales metodele cu utilitate generală pentru evidenţierea caracteristicile dinamice definitorii ale reţelelor Petri ce modelează sisteme de automatizări industriale. S-a prezentat exemplul unei staţii de containere.

Capitolul 5:

Concluzii

Corecţiile de reţea sunt necesare în situaţia în care, după analiza reţelei Petri care modelează sistemul, se ajunge la concluzia că anumite proprietăţi dinamice nu sunt satisfăcute. Astfel de situaţii pot fi următoarele:

a) Reţeau prezintă grade de libertate care ar putea duce procesul modelat în stări nepermise sau nedorite, faţă de un posibil optim, din punct de vedere tehnologic. Pentru a limita numărul gradelor de libertate trebuie introduse restricţii de reţea – elemente structurale suplimentare prin care se limitează numărul gradelor de libertate.

b) Reţeaua nu satisface condiţiile de viabilitate şi/sau reversibilitate. În acst caz reţeaua poate fi corectată prin:

   (i) restricţii de reţea ca în cazul (a) de mai sus, sau

   (ii) extinderi ale reţelei originale cu elemente noi care, la nivelul sistemului modelat, se traduce prin introducerea de funcţionalităţi noi.

Metodele de sinteză a corecţiilor de reţea, prezentate în acest capitol, scot în evidenţă modificările de structură necesare a fi executate în reţeaua de originală pentru asigurarea funcţionării corecte. Restricţiile de reţea, conform metodelor prezentate, se realizează prin

completarea reţelei originale cu bucle autonome de reţea, locaţii complementare şi/sau tranziţii congruente.

Informatii referitoare la corectabilitatea reţelei se pot obţine prin analiza invarianţilor reţelei. S-a demonstrat că dacă o anumită proprietate dinamică nu este prezentă într-o reţea Petri ea nu poate fi realizată prin introducerea de restricţii suplimentare de concesionare realizate prin bucle autonome de reţea.

În cazul în care reţeaua este corectabilă pot fi determinate, pe cale analitică, pe baza matricei diferenţă definită în acest capitol, locaţiile semnificative pe baza cărora pot fi realizate restricţiile suplimentare prin bucle autonome de reţea. Matricea diferenţă se defineşte, respectiv locaţiile semnificative se determină, pentru fiecare tranziţie critică în parte.

Contribuţii

Metodele de corecţie din în acest capitol s-au structurat în conformitate cunecesităţile sintezei strategiei de comandă prezentată în capitolul 7, atât prin formularea restricţiilor cât şi prin schimbările de structură prezentate.

Capitolul 6:

Concluzii

Încapsularea proceselor de alocare a resurselor

Analiza asupra efectelor tranziţiilor temporizate în modelarea cu reţele Petri a condus la următorul principiu de modelare:

> **Modelul de reţea Petri al sistemelor tehnice trebuie astfel elaborat, încât procesele de arbitrare a resurselor partajate să fie "încapsulate" adică separate de procesele de prelucrare informaţională sau materială.**

Maşina virtuală de reţea Petri

Dacă sistemul de coducere se realizează pe baza maşinii virtuale de reţea Petri atunci acesta poate elabora comenzi în concordanţă cu structura cauzală a procesului condus.

> **Pentru a putea elabora mărimi de comandă în concordanţă cu relaţiile cauzale din procesul condus orice sistem de conducere realizat pe baza unui model de reţea Petri trebuie să implementeze o maşină virtuală de reţea Petri conform schemei bloc structurale din Figura 6-14.**

Structurare

Modelul de reţea Petri, împreună cu intrerpretarea prezentată, realizează o structurare clară a sistemului de conducere. Pe baza acestui model, funcţiunile fiecărei componente pot fi univoc delimitate şi eficient încapsulate.

Interfaţare

Interfaţa dinte modulele funcţionale, definită în acest capitol, este cea mai simplă posibilă. Ea este în acelaş timp şi uniformă în sensul că toate mărimile sunt interfaţate în acelaş mod. Specificarea acestei interfeţe este imediată atât în cazul comunicaţiei prin semnale cât şi în cazul comunicţiei prin mesaje. Din acest motiv se poate realiza şi o standardizare eficientă şi uşor acceptabilă.

Sistem deschis

Încapsularea clară şi interfaţarea simplă permit crearea de sisteme efectiv deschise. Fiecare modul funcţional poate fi realizat prin echipamente distincte de la diverşi producători.

Contribuţii

Faţă de literatura de specialitate la care autorul a avut acces, acest capitol prezintă următoarele elemente originale:

a) Definirea, pe baza unei analize asupra efectului tranziţiilor temporizate în modelarea cu reţele Petri, a principiului încapsulării proceselor de arbitrare a resurselor partajate.

b) Definirea unei metode de interpretare a reţelelor Petri şi definirea obiectelor semantice din procesul modelat, pe baza cărora se realizează interpretarea.

c) Definirea noţiunii de ***operator de tranziţie*** ca element de bază în interpretarea reţelelor Petri şi în implementarea sistemelor de automatizare modelate prin reţele Petri.

d) Definirea mărimilor de interacţiune dintre sistemul de conducere, implementat pe baza modelului de reţea Petri, şi procesul condus respectiv mediu.

e) Elaborarea metodelor de determinare a stării procesului condus modelat prin reţele Petri.

f) Elaborarea metodei de refacere a ordinii cauzale a mărimilor de reacţie şi definirea conceptului de "***maşină virtuală de reţea Petri***".

g) Elaborarea structurii sistemului de conducere bazat pe conceptul de maşină virtuală de reţea Petri.

h) Studiul asupra variantelor acestei structuri de conducere.

i) Studiul asupra aspectelor de implementare a sistemului de conducere. Elaborarea algoritmului pentru determinarea *vectorului de stare* şi a *vectorului de concesionare*.

j) Prezentarea unui exemplu de implementare harware a maşinii virtuale de reţea Petri.

k) Studiu asupra caracteristicilor sistemului de conducere implementat pe baza maşinii virtuale de reţea Petri.

Capitolul 7:

Concluzii

S-au definit componentele de sincronizare, de temporizare şi anticipativă a vectorului de comandă iar pe baza acestora, expresia pentru determinarea vectorului de comandă. Această structurare este utilă deoarece fiecare dintre aceste componente presupune metode proprii de specificare, analiză şi implementare.

S-a prezentat o metodă simplă şi elegantă de sinteză a componentei anticipative a vectorului de comandă. Metoda se bazează pe conceptul de maşină virtuală de reţea Petri. Sinteza se realizează pe baza restricţiilor asupra stărilor procesului condus. Dispunând de o implementare a maşinii virtuale de reţea Petri, implementarea metodei este deosebit de simplă deoarece aceasta este de fapt o restricţie a implementării iniţiale. Modulul de control necesar este deasemenea uşor de implementat şi în acelaş timp, după o parametrizare adecvată, este aplicabil tuturor restricţiilor.

S-a prezentat o metodă de sinteză a componentei de sincronizare a vectorului de comandă bazată pe restricţii ce se formulează prin inegalităţi liniare compuse din elementele vectorului de marcaj şi a unor constante şi variabile întregi. S-a arătat că este posibilă transformarea în astfel de inegalităţi şi a altor forme de prezentare a restricţiilor ca de exemplu a celor formulate prin expresii Booleene sau a celor formulate cu elemente ale vectorului de concesionare. Metoda prezentată propune o implementare, realizabilă pe baza conceptului de reţea Petri, care lasă intact modelul de reţea Petri al procesului condus şi care are ca rezultat un modul ce implementează însăşi strategia de comandă.

Contribuţii

Faţă de literatura de specialitate la care autorul a avut acces, acest capitol prezintă următoarele elemente originale:

a) Definirea componentelor vectorului de comandă.

b) Introducerea noţiunii de domeniu de precedenţă şi elaborarea metodei de determinare a acestuia.

c) Definirea funcţiilor care realizează legătura dintre reţeaua de bază şi subreţelele induse prin restricţii.

d) Elaborarea unei metode generale pentru determinarea componentei anticipative a vectorului de comandă, pe baza restricţiilor impuse reţelei.

e) Definirea structurii de conducere pentru metoda de determinare a componentei de sincronizare, bazată pe invarianţi.

f) Enunţarea condiţiei de controlabilitate primară.

Capitolul 8:

Concluzii

S-a prezentat o metodă de partiţionare a reţelei Petri şi o definiţie a interfeţelor dintre subsistemele rezultate care permite, pe baza conceptului de maşină virtuală de reţea Petri, definirea structurilor de conducere a subreţelelor rezultate. Acestea constituie elementele de distribuire în sistemul de conducere a reţelei Petri de bază. Structurile de conducere sunt complet definite astfel încât prin integrarea lor se obţine un comportament echivalent cu un sistem de conducere ce ar implementa întreaga reţea. Integrarea se realizează prin sistemul de comunicaţii care trebuie să respecte condiţia formulată prin Figura 8-4.

Contribuţii

Faţă de literatura de specialitate la care autorul a avut acces, acest capitol prezintă următoarele elemente originale:

a) Definirea domeniilor de interfaţare într-o concepţie proprie. În [DAL'93] se defineşte un domeniu de interfaţare constituit dintr-un singur nod de reţea (locaţie sau tranziţie). Acest singur nod nu aduce informaţie suficientă pentru a putea implementa un sistem de conducere distribuit ce lucrează strict pe baza relaţiilor cauzale din procesul modelat şi din acest motiv se recurge la tehnici speciale cu scopul administrării timpului şi sincronizării ceasurilor.

b) Definirea structurii de conducere pentru controlul distribuit. În literatura de specialitate la care a avut acces autorul nu a găsit definită vre-o structură de conducere general apilicabilă în controlul distribuit al sistemelor.

c) Specificarea condiţiilor ce se impun sistemului de comunicaţii, în privinţa vitezei, pentru ca sistemul de control distribuit să fie corect implementabil.

Capitolul 9:

Concluzii

În cadrul acestui capitol s-a prezentat o sinteză a procedeelor de detaliere a subreţelelor într-o structură ierarhică de reţele Petri şi s-a propus o nouă metodă de reprezentare a subreţelelor în reţeaua de bază.

Metodele de structurare după Valette, Suzuki-Murata şi Abel se bazează pe forma bloc a subreţelei care permite reprezentarea acesteia, în reţeaua de bază, printr-o singură tranziţie. Diferenţele dintre aceste procedee rezultă din modalitatea în care, prin restricţii structurale, se asigură transmiterea proprietăţilor reţelei de bază şi reţelei desfăşurate. În cazul Suzuki-Murata se prezintă şi o definiţie a concesionabilităţii multiple.

S-a propus a nouă modalitate de reprezentare a subreţelelor în reţeaua de bază printr-o structură minimală care încorporează informaţie de stare ce depinde de numărul de execuţii a tranziţiei de intrare şi a celei de ieşire. Această formă de reprezentare are următoarele avantaje:

1) pune în acord comportamentul reţelei de bază, determinată de sintaxa acesteia, cu comportamentul procesului modelat;
2) concesionabilitatea multiplă este reprezentată în mod natural;
3) pune la dispoziţie informaţie suplimentară pentru reacţia după stare;
4) face posibilă definirea structurii de reţea pe baza căreia se poate implementa un echipament de conducere a subreţelei în aşa fel încât să fie definită şi interfaţa cu echipamentul de comandă a reţelei de bază.

S-a arătat că dacă modelul de reţea Petri al sistemelor tehnice este astfel elaborat încât procesele de arbitrare a resurselor să fie "încapsulate" adică separate de procesele de prelucrare informaţională şi materială atunci detalierea reţelei de bază prin structura propusă nu schimbă caracteristicile dinamice (viabilitate, reversibilitate) ale reţelei de bază.

S-a definit structura de reţea pentru implementarea echipamentului de comandă a subreţelei şi, pe baza conceptului de maşină virtual de reţea Petri, s-a prezentat o structură de conducere pentru implementare echipamentului de comandă. S-a definit deasemnea structura de reţea necesară analizei subreţelei. Aceasta este în acord cu principiile de structurare pe niveluri ierarhice şi cu interpretarea dată reţelelor în capitolul 6.

Contribuţii

Faţă de literatura de specialitate la care autorul a avut acces, acest capitol prezintă următoarele elemente originale:

a) Definirea structurii minimale cu informaţie de stare (Fig. 9-3) pentru reprezentarea subreţelelor în reţeaua de bază şi stabilirea condiţiilor pe care trebuie să le îndeplinească subreţelele astfel reprezentate, pentru a asigura transmiterea proprietăţilor reţelei de bază în reţeaua desfăşurată.

b) Autorul a avansat ideea unei analize suplimentare a subreţelei cu scopul determinării unei structuri secvenţiale care modelează relaţiile cauzale reale între tranziţii efective ale subreţelei. Dacă o astfel de structură există, informaţia de stare furnizată de către aceasta poate fi utilizată pentru optimizarea unei strategii de conucere bazate pe reacţia după stare.

c) Demonstarea faptului că dacă, în reţeaua de bază, procesele de arbitrare a resurselor sunt încapsulate, subreţelele pot fi reprezentate prin tranziţii fără ca rafinarea lor ulterioară să modifice proprietăţile de viabilitate şi reversibilitate. În schimb rafinarea aduce informaţia suplimentară de stare cuprinsă în structura din Fig. 9-3 sau Figura 9-4.

d) Definirea structurii de reţea pe baza căruia poate fi implementat un echipament de conducere a subreţelei (a procesului modelat prin aceea subreţea) şi prezentarea unei structuri de conducere, pe baza conceptului de maşină virtuală de reţea Petri şi a conceptului de domeniu de interfaţare introdus în capitolul 8.

Capitolul 10:

Concluzii

Modularitatea este o caracteristică de bază a sistemelor flexibile de fabricaţie. Această modularitate trebuie să se reflecte şi asupra modelului de reţea Petri al sistemului.

În paragraful 10-2 s-au prezentat, prin module şi interfeţe tipice, conceptele modularizării şi a interfaţării dintre module în cazul modelării prin reţele Petri. S-au prezentat trei tipuri de interfeţe şi două tipuri de module.

Pentru interfaţarea dintre modulele de reţea Petri este necesară o structură de reţea care să asigure:

1. Delimitarea clară dintre elementele componente ale modulului şi cele ale interfeţei;
2. Modelarea întârzierii introduse de procesul de transport (al informaţiei sau substanţei).
3. Realizarea domeniilor de interfatare, conform capitolului 8, astfel încât, în cazul unui control distribuit, acestea să fie chiar interfeţele naturale dintre modul şi sistemul de transport;

Pentru a asigura flexibilitatea de adaptare a sistemului, modulele componente trebuie să ofere o gamă cât mai largă de funcţiuni pe o structură dată. Modelarea unor astfel de module este în mod eficient realizabilă prin reţele Petri colorate. Din acest motiv, în acest capitol, s-a definit sructura şi sintaxa acestui tip de reţea Petri.

Conceptele maşinii virtuale de reţea Petri s-au dovedit deosebit de utile în implementarea echipamentelor de comandă pentru sistemele modelate. În paragraful 10.3.5 s-a demonstrat că aceste concepte sunt utilizabile şi în cazul reţelelor Petri colorate.

Pentru ilustrarea conceptelor introduse s-a prezentat exemplul unei linii de montaj. Aceasta realizează funcţiunile impuse prin coordonarea activităţilor modulelor componente. Necesitatea coordonării impune o structură ierarhică sistemului de conducere.

În paragraful 10.5 se prezintă structurile de reţea pe baza cărora poate fi efectuată analiza comportării dinamice a modulelor respectiv a sistemului de coordonare. Analiza poate să fie "modulară" cu condiţia respectării cu stricteţe a principiilor încapsulării resurselor partajate. Altfel exprimat, pentru a obţine rezultate concludente în urma analizei independente a modulelor este necesară respectarea cu stricteţe, în cursul elaborării modelului, a principiului încapsulării resurselor partajate.

Reţele Petri, care stau la baza implementării echipamentelor de comandă, trebuie să conţină domenii de interfaţare bazate pe tranziţii efective, a căror execuţie este observabilă, şi nu tranziţii echivalente ca în cazul analizei. Tehnica de interfaţare elaborată pentru controlul distribuit este şi în acest caz utilizabilă.

Pentru a obţine o implementare modulară structura de conducere din Fig. 10-20, care stă la baza implementării echipamentelor de comandă, trebuie să specifice componentele de interfaţare ale vectorului de concesionare pe baza domeniilor de interfaţare. Acestea sunt definite pe baza modelelor de reţea Petri ale modulelor componente şi a sistemului de coordonare.

Contribuţii

a) Definirea unei structuri de interfaţare dintre modulele de reţea Petri.

b) Demonstrarea faptului că, conceptul de maşină virtuală de reţea Petri este utilizabil şi în cazul modelării prin reţele Petri colorate.

c) Stabilirea condiţiilor în care un model modular de reţea Petri poate fi analizat tot "modular" adică stabilirea condiţiilor în care o analiză pe module va da rezultate concludente cu privire la funcţionalitatea întregului sistem.

d) Specificarea condiţiilor în care sistemul de conducere de coordonare poate fi modelat printr-o reţea Petri modulară.

e) Definirea structurii de reţea pentru implementarea sistemului de conducere.

11.3 Direcţii de dezvoltare viitoare

După opinia autorului ideile prezentate în această lucrare pot constitui baza unei metodologii de abordare a sistemelor de automatizare distribuite şi flexibile. Pentru ca această metodologie să ofere instrumente efective, de inginerie lucrarea de faţă trebuie continuată pe următoarele direcţii principale:

a) Elaborarea de noi metode pentru specificarea şi implementarea strategiei de conducere. Foarte interesante în acest sens sunt ideile prezentate în [PGL'97] şi [PGH'94], idei legate de sinteza bazată pe reguli. Informaţia de stare furnizată prin vectorul de stare de către maşina virtuală de reţea Petri furnizează chiar informaţia de stare necesară determinării stării controlerului, informaţie de stare cuprinsă, în cazul lucrării [PGH'94], în vectorul v_C al taskurilor, vectorul r_C al resurselor respectiv vectorul u_C al pieselor preluate în celula flexibilă de fabricaţie.

b) Studiu asupra posibilităţilor de reprezentare grafică, în cadrul reţelelei, a restricţiilor. În acest sens sunt interesante ideile prezentate în lucrarea [Obe'92].

c) Elaborarea unor metode de optimizare a strategiei de comandă. După opinia autorului trebuie căutate metode, altele decât cele bazate pe stări globale (ex. lanţuri Markov), metodă care, în schimb, tratează concurenţa într-un mod natural. În acest sens, pentru cazul sistemelor ierarhice modelate prin reţele Petri, în capitolul 9, autorul propune o analiză suplimentară a subreţelei, analiză efectuată cu scopul de a determina structuri secvenţiale care modelează relaţii cauzale obiective dintre tranziţii efective ale subreţelei. Informaţia de stare furnizată de această structură poate fi utilizată pentru optimizarea strategiei de conducere bazate pe reacţia după stare. Această soluţie oferă avantajul unei optimizări bazate pe mărimi măsurate în procesul condus, strategia nefiind nevoită să recurgă la temporizări. Lucrările viitoare trebuie să determine metode practice de a obţine o astfel de structură secvenţială.

d) Elaborarea de produse program pentru:

   (i) implementarea maşinii virtuale de reţea Petri conform capitolului 6;

   (ii) implementarea predicatelor tranziţiei şi ale activării conform capitolului 6;

   (iii) specificarea restricţiilor impuse reţelei conform capitolelor 5 şi 7;

   (iv) determinarea componentei anticipative a vectorului de comandă conform capitolului 7;

   (v) realizarea controlului distribuit pe baza modelului de reţea Petri conform capitolelor 8, 9 şi 10.

I. Anexa 1

Corespondenţa dintre elementele abstracte ale reţelei Petri şi obiecte ale sistemelor tehnice din diferite domenii

Obiecte ale teoriei reţelelor	Dezvoltare	Calculatoare de proces	Sisteme de programe	Fiabilitate	Sistem de fabricaţie	Sistem de prod. în flux cont.	Conversia energiei	Sisteme de transport
Locaţii: - anonime - definite	Faze de dezvoltare Documente Programe Caiete de sarcini Proiecte tehnice Alocare lucrări	Memorie Registru Ecran Plotter Magistrală Stări de avarie	Memorie Variabile Semafoare Stări ale programului Operanzi Stări de excepţie	Stare inoperantă Stare operantă	Container Buffer Încărcare maşină Scule Semifabricate	Rezervor Conductă Stări substanţe Stări elemente execuţie Stări elemente de măsură Stări reactori	Combustibil Stare energetică Conductoare Conexiuni	Container, Cisternă Buffer Magazie Paletă Staţie încărcare-descărcare Cadre transport
Tranziţii determi-niste stochas-tice	Specificare Modelare Analiză Simulare Programare PIF	Procesor CAN CNA Transmisii date	Taskuri Proceduri Comenzi Procese	Defectare Reparare Punerea în funcţiune	Montare (piesă) Prelucrare Deformare plastică(ambuti-sare) Separare	Reacţii Transport Compresie Amestec Separare	Turbine Generatoare Supraîncăl-zitoare Pompare Transformare	Macaz Încărcare Descărcare
Relaţii Ponderare arce	Logice Temporale Materiale	Logice Temporale Materiale Energetice	Logice Temporale Materiale	Logice Temporale	Logice Temporale Materiale Energetice	Logice temporale Materiale Energetice	Logice Temporale Materiale Energetice	Logice Temporale Materiale
Marcaje: anonime individua-lizate	Faza de dezvoltare actuală Rezultate obţinute	Date Parametrii Instrucţiuni Intreruperi Căderi ale sistemului	Date Valori Stări de excepţie	Stare de defect Stare funcţională	Stări ale substanţelor Stări energetice	Stări ale materiilor prime Stări energetice	Stări ale materiilor prime Stări energetice	Vehicule fără conducător Încărcătură Vagon Container

Anexa 2

Se prezintă mai jos o posibilă implementare a MVRP sub forma unui program MATLAB. Programul pune la dispoziție toate componenetele sistemului: modelul de rețea Petri Locație/Tranziție al procesului condus, mărimile de reacție (vectorul ***h***), mașina virtuală de rețea Petri și strategia de comandă.

Programul prezintă o interfață grafică interactivă simplă și este definit pentru tratarea a șase modele de rețea Petri. Modelul cu care se lucrează trebuie în prealabil selectat cu ajutorul butonului de selecție "***Select Net***". Acționând butonul "***Netdef***" programul oferă spre editare matricele și vectorii ce descriu modelul de rețea Petri selectat.

Acționând butonul "***Control Def***" programul oferă spre editare modulul de program care implementează strategia de conducere corespunzătoare modelului de rețea Petri selectat.

Toate acțiunile ce urmează se aplică rețelei selectate prin butonul "***Select Net***".

Acționând butonul "***Init***" programul va inițializa modelul de rețea Petri al procesului respectiv mașina virtuală de rețea Petri (MVRP).

Acționând butonul "***Control***" se generează vectorul de comadă, ***u***, conform strategiei de conducere și a tranzițiilor concesionate din cadrul situațiilor conflictuale. Prin acționarea tastei "***Edit u***" programul oferă spre editare vectorul de comandă permițând astfel o intervenție directă asupra acestuia. Prin acționarea butonului "H" se generează vectorul ***h*** într-o manieră cvasialeatoare în sensul că dintre tranzițiile concesionate se consideră a fi executate doar unele, alegerea făcându-se cvasialeator. Prin acționarea tastei "***Edit h***" programul oferă spre editare vectorul de execuție permițând astfel o intervenție directă asupra acestuia.

Pentru a rezolva sitațiile de conflict prin mărimi de conducere s-au prevăzut butoane de selecție notate "***CONFLICT XX***" prin care se specifică tranziția ce se va executa, pentru fiecare situație conflictuală în parte. În cazul exemplului prezentat în lucrarea de față sunt trei situații conflictuale. S-au prevăzut totuși șase butoane de selecție în ideea unor aplicații mai complexe ce ar putea să le solicite.

Prin acționarea butoanelor "***Pas proces***" respectiv "***Pas MVRP***" se execută pasul corespunzător vectorului ***h*** în modelul de rețea Petri al procesului respectiv în MVRP.

Pentru a putea urmării evoluția sistemului, programul afișează, după fiecare, acțiune următoarele:

a) vectorul de stare al procesului (al modelului de rețea Petri), (XP)

b) vectorul de stare furnizat de MVRP, (X);

c) vectorul de concesionare al procesului, (pg);

d) vectorul de concesionare al MVRP, (g);

e) vectrorul de comandă, (u);

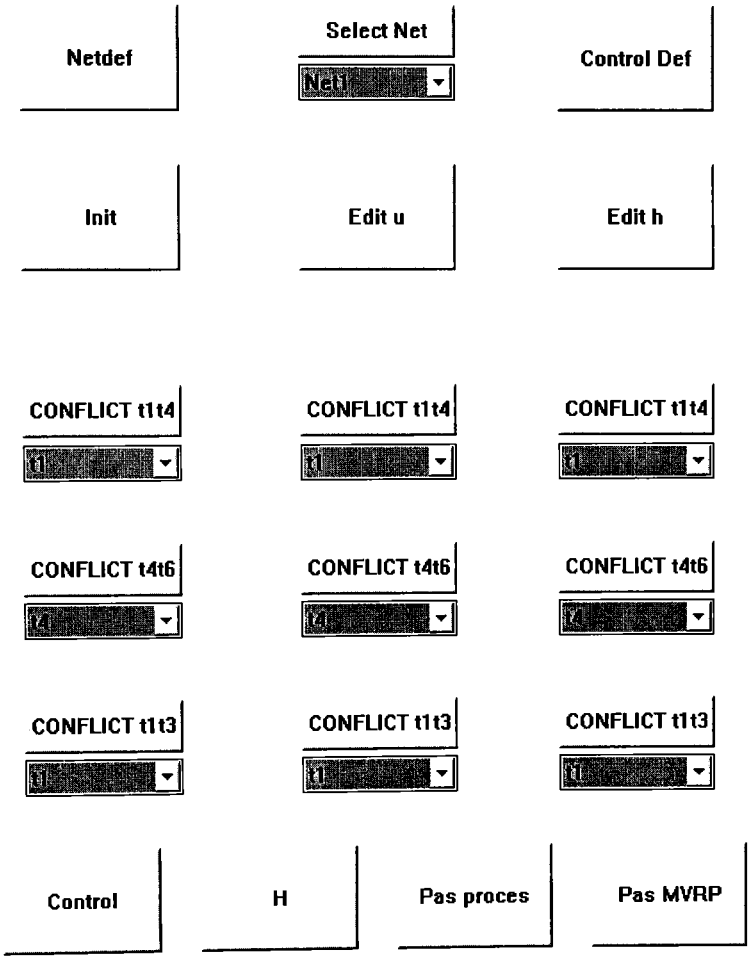
f) vectorul de execuție, (h).

Programul permite studiul efectului strategiilor de comadă asupra comportamentului în buclă închisă prin compararea cu graful accesibil.

Modulele ***Init, Control, H, Pas Proces*** şi ***Pas MVRP*** sunt module genrice şi încorporează funcţii generice privind controlul proceselor bazat pe modelul de reţea Petri. În acest sens creează o maşină virtuală.

Elementele specifice unei reţele date sunt tratate prin modulele ***Netdef*** (definirea reţelei), ***Control Def*** (definirea strategiei de comandă) şi modulele ***CONFLICT XX*** (rezolvarea conflictelor prin mărimi de conducere). Modulele ***Edit u*** şi ***Edit h*** oferă o alternativă de intervenţie directă, "manuală" asupra derulării proceselor, utilă în situaţii de simulare.

Interfața grafică, interactivă



%Selectia retelei ce se doreste a fi definita (inetdef.m)

```
val=get(hpm10,'Value')
if val==1
        !edit netdef1.m
elseif val==2
        !edit netdef2.m
elseif val==3
        !edit netdef3.m
elseif val==4
        !edit netdef4.m
elseif val==5
        !edit netdef5.m
elseif val==6
        !edit netdef6.m
end
```

%Definirea retelei Petri 1 (netdef1.m)

```
function [N,NP,NM,K,W,c,M0] = netdef1

%Matricea pozitiva a retelei
NP=[0 0 1 2 0 0
   1 0 0 0 0 0
   0 1 0 0 0 0
   2 0 0 0 0 1
   0 0 0 1 0 0
   0 0 0 0 1 0
   0 1 0 0 0 1
   0 0 1 0 1 0]

%Matricea negativa a retelei
NM=[-1 0 0 -2 0 0
   0 -1 0 0 0 0
   0 0 -1 0 0 0
   -2 0 0 -1 0 0
   0 0 0 0 -1 0
   0 0 0 0 0 -1
   -1 0 0 0 -1 0
   0 -1 0 -1 0 0]

%Matricea de incidente a retelei
N=NP+NM

%Vectorul capacitatilor
K=[2
   1
   1
```

```
  2
  1
  1
  1
  1]

%Vectorul de multiplicitate a arcelor
W=[1
  1
  1
  1
  1
  1
  1
  1]

%Vectorul tranzitiilor controlabile
c=[1
  1
  1
  1
  1
  1
  1]

%Vectorul marcajului initial
M0=[2
   0
   0
   2
   0
   0
   1
   1]

home
```

%Definirea retelei Petri 2 (netdef2.m)

```
function [N,NP,NM,K,W,c,M0] = netdef2

%Matricea pozitiva a retelei
NP=[0 0 1 0 0 0
   1 0 0 0 0 0
   0 1 0 0 0 0
   0 0 0 0 0 1
   0 0 0 1 0 0
   0 0 0 0 1 0
   0 1 0 0 0 1
```

```
  0 0 1 0 1 0]

%Matricea negativa a retelei
NM=[-1 0 0 0 0 0
  0 -1 0 0 0 0
  0 0 -1 0 0 0
  0 0 0 -1 0 0
  0 0 0 0 -1 0
  0 0 0 0 0 -1
  -1 0 0 0 -1 0
  0 -1 0 -1 0 0]

%Matricea de incidente a retelei
N=NP+NM

%Vectorul capacitatilor
K=[2
  1
  1
  2
  1
  1
  1
  1]

%Vectorul de multiplicitate a arcelor
W=[1
  1
  1
  1
  1
  1
  1
  1]

%Vectorul tranzitiilor controlabile
c=[1
  1
  1
  1
  1
  1
  1
  1]

%Vectorul marcajului initial
M0=[2
  0
  0
  2
  0
  0
  1
```

```
   1]

home
```

%Definirea retelei Petri 3 (netdef3.m)

```
function [N,NP,NM,K,W,c,M0] = netdef

%Matricea pozitiva a retelei
NP=[3 0 1 2 0 0
   1 0 0 0 0 0
   0 1 0 0 0 0
   2 0 0 0 0 1
   0 0 0 1 0 0
   0 0 0 0 1 0
   0 1 0 0 0 1
   0 0 1 0 1 0]

%Matricea negativa a retelei
NM=[-1 0 0 -2 0 0
   0 -1 0 0 0 0
   0 0 -1 0 0 0
   -2 0 0 -1 0 0
   0 0 0 0 -1 0
   0 0 0 0 0 -1
   -1 0 0 0 -1 0
   0 -1 0 -1 0 0]

%Matricea de incidente a retelei
N=NP+NM

%Vectorul capacitatilor
K=[2
   1
   1
   2
   1
   1
   1
   1]

%Vectorul de multiplicitate a arcelor
W=[1
   1
   1
   1
   1
   1
   1
   1]
```

```
%Vectorul tranzitiilor controlabile
c=[1
   1
   1
   1
   1
   1
   1]

%Vectorul marcajului initial
M0=[2
   0
   0
   2
   0
   0
   1
   1]

home
```

%Initializarea retelei selectate (init.m)

```
val=get(hpm4,'Value')
if val==1
        [N,NP,NM,K,W,c,M0] = netdef1
elseif val==2
        [N,NP,NM,K,W,c,M0] = netdef2
elseif val==3
        [N,NP,NM,K,W,c,M0] = netdef3
elseif val==4
        [N,NP,NM,K,W,c,M0] = netdef4
elseif val==5
        [N,NP,NM,K,W,c,M0] = netdef5
elseif val==6
        [N,NP,NM,K,W,c,M0] = netdef6
end

X=M0
PX=M0
save x.net X -ascii
save px.net PX -ascii
[g]=conces(X,N,NP,NM,K)
save g.net g -ascii
[pg]=conces(PX,N,NP,NM,K)
save pg.net pg -ascii
u=zeros(size(g),1)
save u.net u -ascii
h=zeros(size(g),1)
save h.net h -ascii
```

```
q=[1]
save q.net q -ascii

d1=[X,PX]
d2=[g,pg,u,h]
home
disp (' ')
disp (' ')
disp (' _____')
disp (' ')
disp ('    X    PX  ')
disp (' -------------------------')
disp (d1)
disp (' _____')
disp (' ')
disp ('    g    pg    u    h')
disp (' -------------------------')
disp (d2)
```

%Determinarea vectorului de concesionare (conces.m)

```
function [g]=conces(X,N,NP,NM,K)
 [a,b]=size(N)
for j=1:1:b
        g(j,1)=0
                if -NM(:,j)<=X
                        if X<=K-N(:,j)
                                g(j,1)=1
                        end
                end
end
```

%Selectia retelei pentru care se defineste strategia de comanda (cntrdef.m)

```
val=get(hpm4,'Value')
if val==1
        !edit control1.m
elseif val==2
        !edit control2.m
elseif val==3
        !edit control3.m
elseif val==4
        !edit control4.m
elseif val==5
        !edit control5.m
elseif val==6
        !edit control6.m
end
```

%Determinarea vectorului de comanda (control.m)

```
val=get(hpm4,'Value')
if val==1
        control1
elseif val==2
        control2
elseif val==3
        control3
elseif val==4
        control4
elseif val==5
        control5
elseif val==6
        control6
end
```

%Determinarea vectorului de comanda 1 (control1.m)

```
load g.net -ascii
load px.net -ascii
u=g
if PX==[2;0;0;2;0;0;1;1]
        load ct1t4.net -ascii
        u=g&ct1t4
elseif PX==[2;0;0;1;0;1;0;1]
        load ct4t6.net -ascii
        u=g&ct4t6
elseif PX==[1;0;1;2;0;0;1;0]
        load ct1t3.net -ascii
        u=g&ct1t3
end
save u.net u -ascii

d1=[X,PX]
d2=[g,pg,u,h]
home
disp (' ')
disp (' ')
disp (' _____')
disp (' ')
disp ('    X    PX  ')
disp (' -------------------------')
disp (d1)
disp (' _____')
disp (' ')
disp ('    g    pg    u    h')
disp (' -------------------------')
disp (d2)
```

%Determinarea vectorului de comanda 2 (control2.m)

```
load g.net -ascii
load px.net -ascii
u=g
if PX==[2;0;0;2;0;0;1;1]
        load ct1t4.net -ascii
        u=g&ct1t4
end
if PX==[2;0;0;1;0;1;0;1]
        load ct4t6.net -ascii
        u=g&ct4t6
end
if PX==[1;0;1;2;0;0;1;0]
        load ct1t3.net -ascii
        u=g&ct1t3
end
if PX==[1;1;0;2;0;0;0;1]
        u(4)=0
end
if PX==[2;0;0;1;1;0;1;0]
        u(1)=0
end

save u.net u -ascii
d1=[X,PX]
d2=[g,pg,u,h]
home
disp (' ')
disp (' ')
disp (' _____')
disp (' ')
disp ('    X    PX  ')
disp (' -------------------------')
disp (d1)
disp (' _____')
disp (' ')
disp ('    g    pg    u    h')
disp (' -------------------------')
disp (d2)
```

%Determinarea vectorului de comanda 3 (control3.m)

```
load g.net -ascii
load px.net -ascii

u=g
if PX==[2;0;0;2;0;0;1;1]
        load ct1t4.net -ascii
        u=g&ct1t4
end
```

```
if PX==[2;0;0;1;0;1;0;1]
        load ct4t6.net -ascii
        u=g&rct4t6
end
if PX==[1;0;1;2;0;0;1;0]
        load ct1t3.net -ascii
        u=g&rct1t3
end
if PX==[1;1;0;2;0;0;0;1]
        u(4)=0
end
if PX==[2;0;0;1;1;0;1;0]
        u(1)=0
end

save u.net u -ascii
d1=[X,PX]
d2=[g,pg,u,h]
home
disp (' ')
disp (' ')
disp (' _____')
disp (' ')
disp ('    X     PX   ')
disp (' -------------------------')
disp (d1)
disp (' _____')
disp (' ')
disp ('    g     pg    u     h')
disp (' -------------------------')
disp (d2)
```

%Specificarea tranzitiilor ce se executa (hc.m)

```
load q.net -ascii
load pg.net -ascii
load c.net -ascii
load u.net -ascii
%Se determina tranzitiile executabile
hp=(pg&rc&u)|(~c&pg)
n=[0]
for i=1:1:size(pg)
        if hp(i)~=0
                n=n+1
        end
end
if rem(n,2)~=0
        h=hp
else
        h=zeros(size(pg),1)
        for i=q:1:size(h)
```

```
                if hp(i)==1
                        h(i)=1
                        p=i
                        m=i
                end
        end
        if h==0
                for i=1:1:p
                        if hp(i)==1
                                h(i)=1
                                m=i
                        end
                end
        end
        save q.net m -ascii
end
save h.net h -ascii

d1=[X,PX]
d2=[g,pg,u,h]
home
disp (' ')
disp (' ')
disp (' _____')
disp (' ')
disp ('    X    PX   ')
disp (' -------------------------')
disp (d1)
disp (' _____')
disp (' ')
disp ('    g    pg    u    h')
disp (' -------------------------')
disp (d2)
```

%Executia unui pas MVRP (pas.m)

```
load x.net -ascii
load g.net -ascii
load h.net -ascii
p=g&h
X=X+N*p
save x.net X -ascii
[g]=conces(X,N,NP,NM,K)
save g.net g -ascii
d1=[X,PX]
d2=[g,pg,u,h]
home
disp (' ')
disp (' ')
disp (' _____')
disp (' ')
disp ('    X    PX   ')
```

```
disp (' -------------------------')
disp (d1)
disp (' _____________________________')
disp (' ')
disp ('   g    pg   u    h')
disp (' -------------------------')
disp (d2)
```

%Executia unui pas Proces (pasp.m)

```
load px.net -ascii
load h.net -ascii
PX=PX+N*h
save px.net PX -ascii
[pg]=conces(PX,N,NP,NM,K)
save pg.net pg -ascii
d1=[X,PX]
d2=[g,pg,u,h]
home
disp (' ')
disp (' ')
disp (' _____________________________')
disp (' ')
disp ('    X     PX  ')
disp (' -------------------------')
disp (d1)
disp (' _____________________________')
disp (' ')
disp ('    g    pg   u    h')
disp (' -------------------------')
disp (d2)
```

%Selectia tranzitiei ce se va executa: Conflict t1t3 (sett1t3.m)

```
load g.net -ascii
ct1t3=zeros(size(g),1)
val=get(hpm3,'Value')
if val==1
        ct1t3(1)=1
elseif val==2
        ct1t3(3)=1
end
save ct1t3.net ct1t3 -ascii
```

%Selectia tranzitiei ce se va executa: Conflict t1t4 (sett1t4.m)

```
load g.net -ascii
ct1t4=zeros(size(g),1)
val=get(hpm1,'Value')
```

```
if val==1
        ct1t4(1)=1
elseif val==2
        ct1t4(4)=1
end
save ct1t4.net ct1t4 -ascii
```

%Selectia tranzitiei ce se va executa: Conflict t4t6 (sett4t6.m)

```
load g.net -ascii
ct4t6=zeros(size(g),1)
val=get(hpm2,'Value')
if val==1
        ct4t6(4)=1
elseif val==2
        ct4t6(6)=1
end
save ct4t6.net ct4t6 -ascii
```

%Interfata grafica interactiva (aplmvrp.m)

```
h=figure ('Name','Reactia dupa stare pe baza MVRP',...
  'Position',[.5 .9 .5 .9],'units','normalized','Color',[1,1,1])

hpb1=uicontrol ('Style','Pushbutton','Units','normalized',...
    'Position',[.05 .7 .2 .1],'Callback','init',...
    'String','Init')
hpb2=uicontrol ('Style','Pushbutton','Units','normalized',...
    'Position',[.02 .05 .2 .1],'Callback','control',...
    'String','Control')
hpb3=uicontrol ('Style','Pushbutton','Units','normalized',...
    'Position',[.27 .05 .2 .1],'Callback','hc',...
    'String','H')
hpb4=uicontrol ('Style','Pushbutton','Units','normalized',...
    'Position',[.52 .05 .2 .1],'Callback','pasp',...
    'String','Pas proces')
hpb5=uicontrol ('Style','Pushbutton','Units','normalized',...
    'Position',[.77 .05 .2 .1],'Callback','pas',...
    'String','Pas MVRP')
hpb6=uicontrol ('Style','Pushbutton','Units','normalized',...
    'Position',[.05 .85 .2 .1],'Callback','inetdef',...
    'String','Netdef')
hpb7=uicontrol ('Style','Pushbutton','Units','normalized',...
    'Position',[.73 .85 .2 .1],'Callback','cntrdef',...
    'String','Control Def')

hpb8=uicontrol ('Style','Pushbutton','Units','normalized',...
    'Position',[.4 .7 .2 .1],'Callback','!edit u.net',...
    'String','Edit u')
```

```
hpb9=uicontrol ('Style','Pushbutton','Units','normalized',...
   'Position',[.73 .7 .2 .1],'Callback','!edit h.net',...
   'String','Edit h')


hpq1=uicontrol ('Style','Pushbutton','Units','normalized',...
   'Position',[.05 .54 .2 .05],'String','CONFLICT t1t4')
hpm1=uicontrol('Style','Popup','String','t1|t4|','Units','normalized',...
   'Position',[.05 .50 .2 .1],'Callback','sett1t4')
hpq2=uicontrol ('Style','Pushbutton','Units','normalized',...
   'Position',[.05 .39 .2 .05],'String','CONFLICT t4t6')
hpm2=uicontrol('Style','Popup','String','t4|t6|','Units','normalized',...
   'Position',[.05 .35 .2 .1],'Callback','sett4t6')
hpq3=uicontrol ('Style','Pushbutton','Units','normalized',...
   'Position',[.05 .24 .2 .05],'String','CONFLICT t1t3')
hpm3=uicontrol('Style','Popup','String','t1|t3|','Units','normalized',...
   'Position',[.05 .20 .2 .1],'Callback','sett1t3')

hpq4=uicontrol ('Style','Pushbutton','Units','normalized',...
   'Position',[.4 .54 .2 .05],'String','CONFLICT t1t4')
hpm4=uicontrol('Style','Popup','String','t1|t4|','Units','normalized',...
   'Position',[.4 .50 .2 .1],'Callback','sett1t4')
hpq5=uicontrol ('Style','Pushbutton','Units','normalized',...
   'Position',[.4 .39 .2 .05],'String','CONFLICT t4t6')
hpm5=uicontrol('Style','Popup','String','t4|t6|','Units','normalized',...
   'Position',[.4 .35 .2 .1],'Callback','sett4t6')
hpq6=uicontrol ('Style','Pushbutton','Units','normalized',...
   'Position',[.4 .24 .2 .05],'String','CONFLICT t1t3')
hpm6=uicontrol('Style','Popup','String','t1|t3|','Units','normalized',...
   'Position',[.4 .20 .2 .1],'Callback','sett1t3')

hpq7=uicontrol ('Style','Pushbutton','Units','normalized',...
   'Position',[.73 .54 .2 .05],'String','CONFLICT t1t4')
hpm7=uicontrol('Style','Popup','String','t1|t4|','Units','normalized',...
   'Position',[.73 .50 .2 .1],'Callback','sett1t4')
hpq8=uicontrol ('Style','Pushbutton','Units','normalized',...
   'Position',[.73 .39 .2 .05],'String','CONFLICT t4t6')
hpm8=uicontrol('Style','Popup','String','t4|t6|','Units','normalized',...
   'Position',[.73 .35 .2 .1],'Callback','sett4t6')
hpq9=uicontrol ('Style','Pushbutton','Units','normalized',...
   'Position',[.73 .24 .2 .05],'String','CONFLICT t1t3')
hpm9=uicontrol('Style','Popup','String','t1|t3|','Units','normalized',...
   'Position',[.73 .20 .2 .1],'Callback','sett1t3')

hpq10=uicontrol ('Style','Pushbutton','Units','normalized',...
   'Position',[.4 .9 .2 .05],'String','Select Net')
hpm10=uicontrol('Style','Popup','String','Net1|Net2|Net3|Net4|Net5|Net6|',...
   'Units','normalized','Position',[.4 .86 .2 .1])
```

BIBLIOGRAFIE

[Abe'87] Abel, Dirk: *Modellbildung und Analyse ereignisorientierter Systeme mit Petri-Netzen.* Fortschritt-Berichte VDI Reihe 8 Nr. 142. Düsseldorf: VDI- Verlag 1987.

[Abe'90] Abel, Dirk: *Petrinetze für Ingenieure. Modellbildung und Analyse diskret gesteuerter Systeme.* Springer Verlag, Berlin, 1990.

[Asp'93] Aspern, Jens v.: *SPS-Software-Entwicklung mit Petrinetzen.* Hüthig Buch Verlag, Heidelberg, 1993.

[BFG'96] Baccelli, F.; Foss, S.; Gaujal, B.: *Free-Choice Petri Nets. An Algebraic Approach.* IEEE Trans. On Automatic Control, Dec. 1996.

[Bke'96] Bause, F.; Kritzinger, P.-S.: *Stochastik Petri Nets. An Introduction to the Theory.* Verlag Vieweg/Wiesbaden, 1996.

[Bni'92] Budde, R., Nieters, H.: *Einführung in die Netztheorie (Theorie der Petrinetze).* În volumul "Petrinetze in der Automatisierungstehnik" R. Oldenburg Verlag München Wien 1992, pp.23-41.

[BSD'95] Bilinski, K.; Saul, J.M.; Dagless, E.L.: *Efficient functional verification algorithm for Petri-net-based parallel controller designs.* IEEE Proc.-Comput. Digit. Tech., Vol. 142, No. 4, July 1995, pp.255-262.

[Bla'91] Bruce H. Krogh and Lawrence E. Holloway: *Synthesis of Feedback Control Logic for Discrete Manufacturing Systems.* Automatica, Vol.27, No.4, pp.641-651, 1991)

[Büs'86] Büsing, W.: *Datenkommunikation in der Leittechnik.* atp - Auromatisierungs-technische Praxis, 28. Jahrgang. Heft 5/1986, pp.228-237.

[Clu'92] Cramer, Andreas; Luttenberger, Norbert: *Messung, Modellierung und Bewertung von Echtzeitsystemen: Methodik und Fallstudie.* În volumul "Petrinetze in der Automatisierungstechnik", R, Oldenburg Verlag München Wien 1992, pp.180-211.

[Coj'90a] Cojocaru, G.: *O viziune sistemică pentru modelarea intreprinderilor viitorului.* În volumul: Ingineria industrială. Prezent şi perspective. Editura Academiei Române, Bucureşti, 1990.

[Coj'90b] Cojocaru G.: *Structurile flexibile ca bază a sistemelor de producţie cu automatizare extinsă.* În volumul: Ingineria industrială. Prezent şi perspective. Editura Academiei Române, Bucureşti, 1990.

[CPI'88] Călin, S., Popescu,Th., Jora, B., Sima, V.: *Conducerea adaptivă şi flexibilă a proceselor industriale.* Editura tehnică, 1988.

[DAL'93] Duenas, J.-C.; Alonso, A.; Leon, G.; Puente, J.-A.: *Distributed Execution of Specifications.* Real Time Systems 5 (1993) pp. 213-234. Kluwer Academic Publishers 1993.

[Dra'86a] Dragomir, T.-L.: *Regulatoare automate.* Vol.I, Curs IPTVT Timişoara, Facultatea de Electrotehnică,1986.

[Dra'86a] Dragomir, T.-L.: *Regulatoare automate.* Vol.II, Curs IPTVT Timişoara, Facultatea de Electrotehnică,1986.

[Ege'89] Eckelmann, W.; Geibig, K.-F.: *Produktionsnache Informatinsverarbeitung - Basis für CIP.* Nachdruck aus CIM-Management, Heft 5/1989.

[EKS'92] Epple, U.; Kopec, H.; Schmidt, R.: *Strukturierung von Prozeßführungsaufgaben und Leitsystemsofware.* atp - Automatisierungstechnische Praxis 34 (1992) 2 pp.59-67.

[FMD'94] Felder, Miguel; Mandrioli, Dino; Morzenti, Angelo: *Proving Properties of Real-Time Systems Through Logical Specifications and Petri Net Models.* IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. 20, NO. 2, FEBRUARY 1994.

[Fra'92] Franke, Dieter: *A new representation of Boolean functions with applications to binary process control.* IFAC workshop on automatic control for quality and productivity, ACQP'92, june 3-5, 1992, Istanbul, Türkiye. Preprints, Volume 1, pp.225-233.

[Fra'93a] Franke, D.: *Global Linearization of Finite State Machines via Feedback Control.* Preprints of 12th IFAC World Congress. Sydney, Australia 18-23 July, 1993, Vol. 4. pp.157-160.

[Fra'93b] Franke, D.: *Observability and Observer Design for Arithmetically Finite Automata.* Proceedings, 2nd International Conference on "Modelling & Simulation" Victoria University of technology, Melbourne, Australia, July 12-14, 1993, Vol. 1, pp.171-178.

[Fra'94a] Franke, D.: *Arithmetical logic - a bridge between various fields of systems engineering.* Integrated Systems Engineering, Preprints of the IFAC Conference, Baden-Baden, Germany, 27-29 September 1994, pp.437-440.

[Fra'94b] Franke, D.: *Rule-based boolean control of discrete-event systems - an arithmetic approach.* Second International Conference on "Intelligent Systems Engineering"' 5-9 September 1994. Conference Publication Number 395, pp.218-222.

[Fra'94c] Franke, D.: *Feedback Control of Arithmetically Linear Discrete-Event Systems.* Proceedings, First Asian Control Conference, July 27-30, 1994, Tokyo, Vol.2 of 3, pp.885-887.

[Fra'94d] Franke, D.: *A linear state space approach to a class of discrete-event systems.* Proceedings of the IMACS Symposium on Mathematical Modelling 2-4 February, 1994, at Technical University Vienna, Austria, Vol.3, pp.492-496.

[FraH6] Franke, D.: *Neue Wege in der diskreten Steuerungstechnik.* e& i 111. Jg. H.6, pp.254-257.

[Fri'94] Frick, Klaus: *Zur beschreibung von Petri-Netzen als Abtastsystem.* at-Automatisierungstechnik 42(1994) 9, pp.385-390.

[GDi'92] Giua, Alessandro; DiCesare, Frank: *On the Existence of Petri Net Supervisors.* Proceedings of the 31st Conference on Decision and Control, Tucson, Arizona – December 1992.

[Gre'95] Greim, Thomas: *Ein beitrag zur steuerungstechnischen Anwendung von Petri-Netzen.* at - Automatisierungstechnik 44 (1995) 6 pp.274-280.

[Han'90] Hanisch, Hans-Michael: *Dynamik von Koordinierungssteuerungen in diskontinuierlichen verfahrenstechnischen Systemen.* at- Automatisierungstechik 38 (1990) 11. pp.399-405.

[Han'91] Hanisch, H.-M.: *Modellierungskonzept zur operativen Steuerung diskontinuierlicher Produktionssysteme.* msr, Berlin 34 (1991) 1.

[Han'92] Hanisch, Hans-Michael: *Petri-Netze in der Verfahrenstechnik: Modellierung und Steuerung verfahrenstehnischer Systeme.* Oldenburg Verlag München Wien 1992.

[HGZ'96] Holloway, L.-E.; Guan, X.; Zhang, L.: *A Generalization of State Avoidance Policies for Controlled Petri Nets.* IEEE Transactions on Automatic Control, Vol. 42, No. 6, June 1996.

[ISO'82] Griethuysen, J.J., Ed.: *Concepts and Terminology for the Conceptual Schema and the Information Base*, Report of the ISO/TC97/SC5/WG3, Publ. No. ISO/TC97/SC5-N695, 1982

[Jen'92] Jensen, Kurt: Coloured *Petri Nets. Basic Concepts, Analysis Methods and Practical Use.* Volume 1: Basic Concepts; Version 5 - April 1992. Computer Science Department, Aarhus University.

[Jen'92] Jensen, Kurt: *Coloured Petri Nets. Basics Concepts, Analysis Methods and Practical Use. Volume 2:* Analysis Methods; Version 5 - April 1992. Computer Science Department, Aarhus University.

[JKh'92] Jenkins, L.; Khincha, H.-P.: *Deterministic and Stochastik Petri Net models of Protection Schemes.* Transactions on Power Delivery, Vol.7, No.1, January 1992, pp.84-90.

[JLB'95] Jörns, Carsten; Litz, Lothar; Bergold, Stefan: *Automatische Erzeugung von SPS-Programmen auf der Basis von Petri-Netzen.* atp -Automatisierunstechnische Praxis 37 (1995) 3, pp.10-14.

[JŢi'95] Jucan, T.; Ţiplea, F.-L.: *Reţele Petri.* Editura Universităţii „Al. I. Cuza", Iaşi - 1995.

[Jur'92] Jurca, I.: *Programarea orientată pe obiecte în limbajul C++.* Editura Eurobit, Timişoara, 1992.

[Kat'95] Katoen, J.-P.: *Causal Behaviours and Nets.* În volumul "Application and Theory of Petri Nets" 1995. 16th International Conference Turin, Italy, June 26-30, 1995. Proceedings, pp.258-277.

[Kat'96] Katoen, J.-P.: *Quantitative and Qualitative Extensions of Event Structures.* Enschede: Centre for Telematics and Informatin Technology, Ph. D.-thesies series No. 96-09, 1996.

[Kec'81] Kecs, Wolfgang: *Complemente de matematici cu aplicaţii în tehnică.* Editura Tehnică, Bucureşti, 1981.

[KDS'95] Kozlowski, T.; Dagless, E.L.; Saul, J.M.; Adamski, M.; Szajna, J: *Parallel controller synthesis using Petri nets.* IEE Proc.-Comput. Digit. Tech., Vol. 142, No. 4, July 1995, pp.263-271.

[KKL'95] Kluwe, M.; Krebs, V.; Lunze, J.; Richter, H.: *Rekonstruktion qualitativer Prozeßzustände durch ereignisdikrete Beobachter.* at - Automatisierungstechnik 43 (1995) 6, pp.289-295.

[KQu'88] König, R., Quäck L.: *Petri-Netze in der Steuerungs- und Digitaltechnik.* R.Oldenbourg Verlag, München, Wien 1988.

[Krz'90] Kurz, Hans: *Realisierung gehobener Methoden der Regelungstechnik auf Prozeßleitsystemen - Ein Diskussionsbeitrag.* Automatisierungstehnische Praxis atp 32 (1990) 10, pp.489-494.

[Lak'91] Lakos, Charles: *LOOPN - Language for Object-Oriented Petri Nets.* Proceedings of the SCS Multiconference on Object-Oriented Simulation, 23-25 January 1991, Anaheim, California, pp.22-30.

[Lak'93] Lakos, Charles: *Applying Invariant Analysis to Modular Petri Nets.* Proceedings of the Sixteenth Australian Computer Science Conference, 3-5 February, 1993, Brisbane, Qeensland, pp.765-772.

[Lak'95a] Lakos, Charles: *From Coloured Petri Nets to Object Petri Nets.* Application and Theory of Petri Nets 1995. 16th International Conference Turin, Italy, June 26-30, 1995. Proceedings, pp.278-297.

[Lak'95b] Lakos, Charles: *The Object Orientation of Object Petri Nets.* Object-Oriented Programming and Models of Concurrency. A workshop within the 16th International Conference Turin, Italy, June 26-30, 1995.

[Lak'95c] Lakos, Charles: *Pragmatic Inheritance Issues for Object Petri Nets.* Proceedings of the fifteenth International Conference TOOLS PACIFIC, MELBOURNE, 1995, pp.309-321.

[Lak'96] Lakos, Charles: *The Consistent Use of Names and Polimorphism in the Definition of Object Petri Nets.* Application and Theorie of Petri Nets 1996. 17th International Conference Osaka, Japan, June 24-28. 1996. Proceedings. pp.380-399.

[LeAş] Leţia, T.; Aştilean, A.: *A Hybrid Petri Net Model for an Electric Furnace.*

[LeA'96] Leţia, T.; Aştilean A.: *Control method for an electric furnace based on a high-level Petri Net.* The International Conference of Technical Informatics. Proceedings. Automation and Industrial Informatics. November 14-15, 1996, Timişoara, România, Vol.I, pp.177-182.

[Lip'93] Lipp, Hans-Peter: Wissensbasierte Produktionsfürung für flexible Fertigungs-prozesse auf der Basis von zeitbewerteten Fuzzy-Petri-Netzen. at-Auromatisierungstechnik 41 (1993) 8, pp.281-287.

[LLu'96] Lichtenberg, G.; Lunze, J.: Identification of discrete event models for continuous- variable systems. UKACC International Conference on CONTROL '96, 2-5 September 1996, Conference Publication No. 427, pp.711-715.

[Lun'92] Lunze, Jan: A Petri-net approach to qualitative modelling of continuous dynamical systems. SAMS, 1992, Vol. 9, pp.89-101. 1992 Gordon and Breach Science Publishers S. A.

[Lun'93a] Lunze, Jan: Ein Ansatz zur qualitativen Modellierung und Regelung dynamischer Systeme. at - Automatisierungstechnik 42 (1993) 12, pp.451-460.

[Lun'93b] Lunze, Jan: On the stability of qualitative models. Systems & Control Letters 21 (1993) 137-142.

[Lun'94] Lunze, Jan: Qualitative Modelling of Linear Dynamical Systems with Quantized State Measurements. Automatica, Vol. 30, No. 3, pp.417-431, 1994.

[Lun'95] Lunze, Jan: Stabilization of nonlinear systems by qualitatve feedback controllers. International Journal of Control, 1995, Vol. 62, No. 1, pp.109-128.

[Lun'96] Lunze, J.; Nixdorf, B.; Richter, H.: Eine Methode zur Prozeßführung kontinuierlicher Systeme auf der Basis eines qualitativen Prozeßmodells. atp Automatisierungstechnische Praxis 38 (1996) 7 pp.46-54.

[Mat'93] Mattern, Friedemann: Efficient Algorithms for Distributed Snapshots and Global Virtual Time Approximation. Journal of Parallel and Distributed Computing 18, 423-434 (1993), pp.423-433.

[Mat'94] Mattern, Fr.; Schwartz, R.: Detecting causal relationships in distributed computations: in search of the holly grail. Distributed Computing (1994) 7: pp.149-174.

[Mol'82] Molloy, K. Michael: Performance Analysis Using Stochastic Petri Nets. IEEE TRANSACTIONS ON COMPUTERS, VOL. C-31, NO. 9, September 1982, pp.913-917.

[MOS'93] Mochel, Thomas; Oberweis, Andreas; Sanger, Volker: INCOME/STAR the Petri net simulation concepts. Reprint SAMS, 1993, Vol. 13 pp.21-36, Gordon and Breach Science Publishers S.A. , 1993.

[Moß'96] Moßig Kai: Steuerungsentwurf für ereignisdiskrete Systeme durch Vorgabe eines Eigenvektors der Max-Plus-Algebra. at - Automatisierungstechnik 44 (1996) 2 pp.80-86.

[Obe'90] Oberweis, Andreas: Zeitstrukturen für Informationssysteme. Inauguraldissertation zur Erlangung des akademischen Grades eines Doktors der Naturwissenschaften der Universität Mannheim. Mannheim 1990.

[Obe'92] Oberweis, Andreas: Spezifikation von Mechanismen zur Ausnahmebehandlung mit Petri-Netzen. at-Auromatisierungstechnik 40 (1992) 1, pp.21-30.

[PAG'93] Păunescu, Florin; Goteşteaunu, Dănuţ Petre: Sisteme cu prelucrare distribuită şi aplicaţiile lor. Editura Tehnică Bucureşti, 1993.

[PĂS'97] Păstrăvanu, Octavian: Sisteme cu evenimente discrete. Tehnici calitative bazate pe formalismul reţelelor Petri. MATRIX ROM Bucureşti 1997.

[PGH'94] Păstrăvanu, O.C.; Gürel, A.; Huang, H.H.; Lewis, F.L.: "Rule-based controller design algorithm for discrete event manufacturing systems", Proc. American Control Confference, pp. 299-305, Baltimore, 1994.

[PGL'97] Păstrăvanu, O.C.; Gürel, A.; Lewis, F.L.: "Teaching discrete event control of manufacturing systems" Proc. 4th Symp. Advances in Control Education, pp. 307-312, Istanbul.

[Pfe'86] Pfleger, J.: Kommunikationssystem Feldbus. atp - Auromatisierungs-technische Praxis, 28. Jahrgang. Heft 5/1986, pp.223-227.

[Pol'85] Polke, M.: Prozeßleittechnik in der chemischen Industrie. Elektronische Regelanlagen 27 (1985), heft 3, Seite 166-173.

[Pol'87] Polke, M.; Körner, W.; Moll, P.: Zur funktionalen Gliederung von Leitsystemen. Nachdruck aus Automatisierungstechnische Praxis Heft 9/1987.

[Pol'89a] Polke, M.; Ahrens, W.: Netzmodelle als systemtechnische Informationsbasis für die Prozeßleittechnik. at- Automatisierungstechnik 37 (1989) 3 pp.94-103 şi 4 pp.138-144.

[Pol'89b] Informationsstrukturen in der Automatisierungstechnik. Manuskript, Plenarvortrag INTERKAMA 1989, Mai 1989.

[Pol'94] Polke, M.; Bucher, H.; Lauber, J.: Das Informationsmodell: Basis für die interdisziplinäre Prozeßbeschreibung. at - Automatisierungstechnik 42 (1994) 1 pp.5-10.

[PrP'97] Preitl, Ştefan; Precup, Radu Emil: Introducere în conducerea fuzzy a proceselor. Editura Tehnică Bucureşti – 1997.

[Quä'91a] Quäck, Lothar: Aspekte der Modellierung und Realisierung der Steuerung technologischer Prozesse mit Petri-Netzen. at-Automatisierungstechnik 39 (1991) 4, pp.116-120. Partea I.

[Quä'91a] Quäck, Lothar: Aspekte der Modellierung und Realisierung der Steuerung technologischer Prozesse mit Petri-Netzen. at-Automatisierungstechnik 39 (1991) 5, pp.158-64. Parte II.

[Quä'91b] Quäck, Lothar: Modellbildung und Realisierung von Automatisierungssystemen mit Petri-Netzen. În volumul "Efizientes Engineering komplexer Automatisierungssyteme. Methoden, Anwendungen und Tools auf der Basis von Petri-Netzen. Technische Universität Braunschweig, 1991, pp.27-42.

[Reh'95] Rehkopf, Andreas: Gesteuerte ereignisdiskrete Prozesse: Weiterführende Überlegungen zur "Quäck'schen Waschmaschine". atp - Automatisierungstechnische Praxis 43 (1995) 5 pp.242-248.

[Reh'96] Rehkopf, Andreas: Optimale Steuerung ereignisdiskreter Prozesse der Fertigungstechnik mit der Max-Plus-Algebra. at - Automatisierungstechnik 44 (1996) 2 pp.87-93.

[Rei'91] Reisig, Wofgang: Petrinetze als anschauliches graphisches Beschreibungsmittel auf matematischer grundlage. În volumul "Efizientes Engineering komplexer Automatisierungssyteme. Methoden, Anwendungen und Tools auf der Basis von Petri-Netzen.Technische Universität Braunschweig,1991, pp.1-14.

[Rep] Report on "Distributed Automation". CIRED Ad hoc Working Group 2 "Distribution Automation". Functions and Data.

[Rmo'96] Rehkopf, A.; Moßig, K.: Einführung in die „Max-Plus"-Algebra zur Beschreibung ereignisdiskreter dynamischer Prozesse. at-Automatisierungstechnik 44 (1996) 1 pp.3-9.

[Sch'91a] Schnieder, E.: Methodischer entwurf von Automatisierungssystemen mit Petrinetzen. În volumul "Efizientes Engineering komplexer Automatisierungssyteme. Methoden, Anwendungen und Tools auf der Basis von Petri-Netzen. Technische Universität Braunschweig, 1991, pp.27-42.

[Sch'91b] Schnieder, Eckehard: Braucht die automatisierungstechnik eine Theorie ? at - Automatisierungstechnik 39 (1991) 11, pag. 391-401.

[Sch'94] Schnieder, E.; Erdmann, L.; Schielke, A.-G.: Referenzmodell zur Strukturierung von Leitsystemen. at- Automatisierungstechnik 42 (1994) 5 pp.187-197.

[SLD'91] Seiche, Werner; Abel, Linnich; Abel Dirk: Entwurf verklemmungsfreier Steuerungen auf der Grundlage einer graphentheoretischen Petri-Netz-Analyse. at-Automatisierungstechnik 41 (1993) 3, pp.88-93.

[SES'94] Schnieder, Eckehard; Shansagimov, Baurshan: Analyse und Wahl optimaler Varianten von Automatisierungssytemen mit Hilfe von Petrinetzdarstellungen. at-Automatisierungstechnik 40 (1992) 6 pag.228- 234 şi at-Automatisierungstechnik 40 (1992) 7 pag.257-262.

[Sei'92] Seiche, Werner: Analyse und Synthese diskret gesteuerter Systeme mit Petri-Netzen. Frtschritt-Berichte VDI, Reihe 8: Meß-, Steuerungs- und Regelunstechnik, Nr. 269. VDI Verlag GmbH Düsseldorf 1991.

[SVa] Silva, Manuel; Valette, Robert: Petri Nets and Flexible Manufacturing.

[Şab'81] Şabac, Ion Gh.: Matematici speciale. Vol. 1. Editura Didactică şi Pedagogică, Bucureşti, 1981.

[Stö'89] Stöckler, H.-P.: Fortschrittliche Strukturen von Leitsystemen. Nachdruck aus der Automatisierungstechnische Praxis, Heft 7/1989.

[Sub'96] Substation Integrated Protection, Control and Data Aquisition. Phase 1, Task 2, Requirements Specification. Preliminary Report, Version 0.4, March, 1996.

[Tak'96a] Takács, Balázs-Imre: Petri-Netze in der entwurf der Steuerung ereignisdiskreter technologischer Prozeße. Buletinul Ştiinţific UPT 1996 Tom 41(55), Seria Automatică si Calculatoare, pp.56-68.

[Tak'96b] Takács, Balázs-Imre: Implementation aspects of a virtual Petri-Net-Mashine. The International Conference of Technical Informatics. Proceedings. Automation and Industrial Informatics. November 14-15, 1996, Timişoara, România, Vol.I, pp.139-148.

[Tak'96c] Takács, Balázs-Imre: Petri-Net-Based Controller Design. SINTES 8-th edition, "International Sympsium on Systems Theory, Robotics, Computers and Process Informatics" Section "Automation and Robot Control", Craiova, România, 6-7 June 1996, pp.279-280.

[Tak'98] Takács, B.-I., Stoica, I., Coroiu, N., Cucu, A.: DA/SCADA Implementation Strategy in the Conditions of a Romanian Utility. DistribuTECH DA/DSM Europe 98, 26-29 October 1998 London, Session DT 3.4: Operational Management and Automation. Proceedings.

[Vas'70] Vasiu, Mircea: Fizica teoretică. Editura didactică şi pedagogică, Bucureşti - 1970.

[Vol'92] Volda, W.: Im System programmierbare komplexe PLDs. elektronik industrie 6-1992, pp84-89.

[Wag'90] Wagner, S.: Informationstechnische Aspekte von CIM. atp - Automatisierungstechnische Praxis 32 (1990) 1 pp.7-22

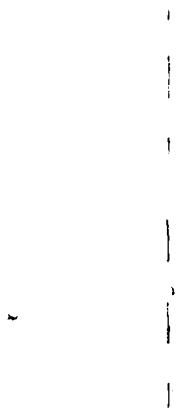
[Win'86a] Winkler, P.: Anforderungsbeschreibungen mit Netzmodellen. Automatisierungstehnische Praxis atp, 28. Jahrgang, Heft 1/1986, pp.32-39, Partea I.

[Win'86b] Winkler, P.: Anforderungsbeschreibungen mit Netzmodellen. Automatisierungstehnische Praxis atp, 28. Jahrgang, Heft 2/1986 pp.94-98, Partea II.

[Zab'85] Zaborsky, M.: Dezvoltarea ştiinţei sistemelor automate: trecut, prezent şi viitor. AMC 48, Editura Tehnică, Bucureşti, 1985. p.59-90.

[Zim'93] Zimmermann, H.-J.: Fuzzy Set Theory and Its Applications. Second, Revised Edition. Kluwer Academic Publishers, Boston/Dordrecht/London, 1993.

[Yam'95] Yamalidou, K.; Moody, J.; Lemmon, M.; Antsaklis, P.: Feedback Control of Petri Nets Based on Place Invariants. Automatica, Vol. 32, No. 1, 1995, pp.15-28.



CURRICULUM VITAE

TAKÁCS Balázs-Imre, director comercial Schrack Energietechnik srl, Oradea

DATE GENERALE:

Data şi locul naşterii:	10 ianuarie 1955, PETREU, judeţul Bihor
Starea civilă:	căsătorit, doi copii
Locul de muncă actual:	Schrack Energietechnik srl, Oradea
Funcţia:	director comercial, Schrack Energietechnik srl, Oradea
Limbi cunoscute:	româna, maghiara, germana, engleza, franceza

FORMAŢIE:

1974	Bacalaureat (Liceul de Cultură Generală Nr. 4 din Oradea)
1980	Inginer Electronică Industrială (I.P.T.V. Timişoara)
1982	Specialist acţionări electrice pentru maşini unelte
1984	Cercetător ştiinţific la I.P.A. Filiala din Oradea
1986	Cercetător ştiinţific gradul III la I.P.A. Filiala din Oradea
1988	Cercetător ştiinţific gradul II la I.P.A. Filiala din Oradea
1990	Responsabilul Consiliului Tehnico-Economic pentru Filiala din Oradea a I.P.A
1991	Inginer de sistem, specialist SCADA la FTDEE Oradea.
1998	S.C. Schrack Energitechnik srl, director comercial.

EXPERIENŢĂ:

1980 ... 1984	Inginer electronică industrială la "Electrotehnica" Bucureşti, experienţă în domeniul acţionărilor electrice pentru maşini unelte.
1984 ... 1992	Cercetător ştiinţific la I.P.A. Filiala din Oradea în domeniul sistemelor CNC pentru maşini unelte speciale, conducător de proiect l;a 3 tipuri de sisteme CNC.
1992 ... 1998	Inginer de sistem, specialist SCADA la FTDEE Oradea, elaborarea caietelor de sarcini pentru sistemul de automatizare a distribuţiei în FTDEE Oradea respectiv pentru sistemul integrat de protecţie control a staţiilor de transformare.
1998 ... ????	Schrack Energirtechnik srl: echipamente şi sisteme de distribuţie a energiei electrice, sisteme de supraveghere, automate programabile.
1997	Predarea unui curs de "Electronică şi Automatizări" la Universitatea din Oradea
1998	Predarea unui curs de "Reţele Petri" la Universitatea din Oradea.

Lucrări publicate: 9

Inovaţii: 1

Lucrări simpozioane internaţionale: 3