MODELAREA SISTEMELOR ORIENTATE PE SERVICII PRIN REȚELE PETRI RECONFIGURABILE CU ATRIBUTE MATRICEALE

Iu. Țurcanu, drd., E. Guțuleac, dr. hab, prof. univ., D. Palii, drd.
Universitatea Tehnică a Moldovei

INTRODUCERE

Actualmente, sistemele de calcul cu arhitecturi orientate pe servicii (SCOS) în timp real cunosc o dezvoltare rapidă, atât sub aspectul complexității şi/sau performanțelor, cât şi al ariei de răspândire [1, 4, 8]. Acest tip de sisteme trebuie să aibă o flexibilitate, disponibilitate şi siguranță în funcționare deosebită. Un astfel de sistem poate fi considerat şi implementat drept fiind o colecție de configurații, unde fiecare din acestea este o rețea de componente ce comunică între ele. Diferite configurații pot fi folosite pentru alocarea şi procesarea a diferitor servicii sau condiții de operare ale aplicațiilor. Ca urmare, facilitățile trecerii în timp real de la o configurație către alta pe parcursul rulării, duc la creşterea s*iguranței în funcționare* şi a flexibilității utilizării sistemului [2], asigurarea cărora în timpul reconfigurării dinamice este dificilă din cauza interacțiunii între serviciile de aplicație ale sistemului.

Unul dintre cele mai răspândite formalisme moderne, folosite pentru modelarea şi analiza sistemelor paralele/distribuite cu evenimente discrete, sunt rețelele Petri (*RP*) de diferite extensii [1, 3, 6, 7]. Totodată, apare necesitatea de a dezvolta aceste formalisme pentru a descrie mai adecvat, mai flexibil şi mai comod sisteme SCOS cu structuri complexe dinamic restructurabile. În [4] au fost introduse *RP* generalizate *reconfigurabile* (*RGR*), pentru simularea, verificarea şi analiza sistemelor concurente care în mod dinamic sunt supuse schimbărilor de structură.

Însă pentru sisteme reale într-o simulare vizuală prin *RGR*, este posibil ca unele variabile ale atributelor obiectelor acestor tip de rețele trebuie să ia valori specifice multiple, care fie că nu pot fi descrise compact într-un mediu *RGR* sau că modelarea lor va creşte în mod semnificativ complexitatea grafică a modelului sistemului. De exemplu, în scopul de a efectua evaluarea indicatorilor de performanță ale unor procese de calcul orientate pe servicii pentru o anumită sarcină de simulare cu un profil de câteva mii de servicii cu valori diferite, este necesar ca un număr mare de locații, tranziții şi arce să fie adăugate la modelul *RGR* pentru a obține valoarea de încărcare dorită pentru fiecare interval de timp specificat. Aceasta duce la o creştere considerabilă a complexității structurale a modelului *RGR* şi deci apar probleme semnificative de analiză, cum ar fi explozia spațiului de stări, ceia ce induce la creşterea complexității globale de calcul, precum şi la creşterea timpului de simulare al funcționării sistemului analizat. Pentru a aborda acest tip de probleme, în lucrare sunt introduse modele *RGR interpretate* cu atribute matriceale de calcul, notată *RGIM* , în mod similar cum au fost folosite în [5] prin introducerea *arcelor database* cu ponderi matriceale. În acelaşi context, sunt considerate unele aspecte de modelare şi analiză a proceselor de calcul orientate pe servicii reconfigurabile.

Un avantaj important al demersului propus constă în faptul că modelul *RGIM* este foarte concis şi flexibil, deoarece mai multe dintre variabilele sale sunt parametrizate ce pot lua valori alternative.

1. REȚELE RGR CU ATRIBUTE MATRICEALE

Un atribut matriceal, dependent de marcajul curent, al unui obiect rețele *RGR* cu atribute matriceale, abreviat *RGRM* , (arc, locație, tranziție obişnuită, tranziție de procesare, regulă de rescriere, etc.) este definit de către o matrice $\boldsymbol{A} = [a_{i,j}(M)]_{k \times n}$, $a_{i,j}(M) \in IN_+$, $\boldsymbol{A} \in \mathbf{A}$ de tipul specificat şi de către un set de locații de control $P^A \subset P$, determinat de dimensiunile acestei matrice $k \times n$, astfel încât elementele acestei matrice sunt constante, variabile sau nişte funcții (expresii) de tipul specificat care, eventual, pot fi dependente de marcajul curent M al *RGRM* . De exemplu, un vector coloană are o dimensiune egală cu unu, deci $P^A = \{p_l\}$; o matrice 2-D are o dimensiune egală cu doi, deci $P^A = \{p_l, p_v\}$, etc. Capacitatea locațiilor de control este respectiv egală cu: $M(p_l) = k$ şi $M(p_v) = n$. Marcajul curent $m_l = M(p_l)$ şi $m_v = M(p_v)$ al locațiilor de control $P^A = \{p_l, p_v\}$ arată poziția

elementului în matricea $\boldsymbol{A}$, mărimea căruia este necesar să fie importată şi luată în considerare la funcționarea şi analiza modelului.

Un atribut matriceal al *RGRM* va fi reprezentat astfel încât el va conține între paranteze pătrate numele matricei specificate în prealabil. Astfel, cardinalitatea arcelor matriceale *directe,* simbolizate respectiv cu  şi a arcelor *inhibitor* sau *test* [2, 3, 5], simbolizate în mod similar pot lua valori care sunt conținute într-o matrice specificată $\boldsymbol{A}$. Pentru a ilustra acest fapt în figura 1 este prezentat un exemplu de rețea $RGRM1$, cu marcajul curent $M=(3,4,1,1)=(3p_1 4p_2 p_3 p_4)$, locații de control $P^A=\{p_1, p_2\}$ şi un arc matriceal direct (t_1, p_3) ce ilustrează modelarea grafica şi operarea acestui arc care importă mărimea ponderii sale curente din matricea $\boldsymbol{A}$, redată de relația (1):

$$\boldsymbol{A}=\begin{bmatrix} 3 & 2m_1+m_3 & 1 & 1+m_3 \\ 2 & 7 & 5 & 3+m_2 \\ 1 & 4 & 8 & m_2+4m_3 \end{bmatrix}, \qquad (1)$$

unde $m_i=M(p_i), i=1,2,3,4$ este numărul de jetoane în locația p_i pentru marcajul curent.

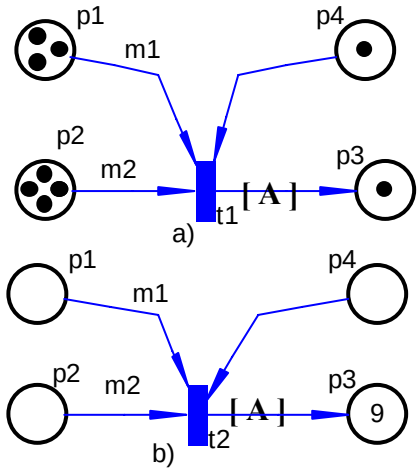

Figura 1. Acționarea unui arc matriceal direct în $RGRM1$: a) starea inițială; b) starea finală.

Ponderea arcului (p_1, t_1) este egală cu m_1, iar a arcului (p_2, t_1) este egală cu m_2. Deoarece matricea $\boldsymbol{A}$ este bidimensională, sunt necesare două locații de control. În cazul rețelei $RGRM1$ avem $P^A=\{p_1, p_2\}$. Locația de control p_1 are capacitatea $K_p(p_1)=3$, iar capacitatea lui p_2 este $K_p(p_2)=4$. Aceste locații sunt incidente înainte la tranziția t_1 de intrare la arcul matriceal direct (t_1, p_3), simbolizat cu $[\boldsymbol{A}]$. Poziția elementului $a_{i,j}$ al matricei $\boldsymbol{A}$, mărimea ponderii căreia este de dorit să fie importată, este realizată de informațiile despre numărul respectiv de jetoane conținute în aceste locații şi anume: elementul selectat al acestei matrice se află în rândul $i=m_1$ şi coloana $j=m_2$.

În figura 1a este prezentată starea $RGRM1$ înainte de declanşarea lui t_1, din care se poate observa că $m_1=3$ (deoarece, locația p_1 are trei jetoane) şi $m_2=4$ (deoarece, locația p_2 are patru jetoane), iar în figura 1b este prezentată starea $RGRM1$ după declanşarea lui t_1, din care se poate observa faptul că locația p_3 are un marcaj egal cu valoarea 9, deoarece a fost selectat elementul $a_{3,4}=m_2+4m_3=8$. Ca rezultat al declanşării t_1, obținem $M[t_1>M'$, $M'=M+a_{2,4}=(9p_3)$.

În mod similar pot fi matricial parametrizate toate atributele rețelei *RGRM* cum ar fi, de exemplu, următoarele atribute matriceale [3]:

- de gardă ale tranzițiilor $\boldsymbol{B}\in\mathbf{B}$ pentru care $\boldsymbol{B}=[b_{i,j}(M)]_{k\times n}$, $b_{i,j}(M)\in\{true, false\}$;
- de capacitate ale locațiilor $\boldsymbol{KP}\in\mathbf{KP}$ pentru care $\boldsymbol{KP}=[kp_{i,j}(M)]_{k\times n}$, $kp_{i,j}(M)\in IN_+$ cu $K_p(p_i)=\max\limits_{\forall i}(m_i, kp_{i,j}(M))$;
- a regulilor de rescriere validate $\boldsymbol{R}\in\mathbf{R}$ $\boldsymbol{R}=[r_{i,j}(M)]_{k\times n}$, $r_{i,j}(M):\Gamma\zeta^L_{i,j}\triangleright\Gamma\zeta^W_{i,j}$ ale rețelei curente $\Gamma\zeta$.

Menționăm că la declanşarea $r_{i,j}(M)$, operatorul binar $\triangleright$ va produce o modificare a structurii $\Gamma\zeta$ astfel încât subrețeaua $\Gamma\zeta^L_{i,j}\subseteq\Gamma\zeta$ va fi eliminată din $\Gamma\zeta$, obținându-se o nouă rețea $\Gamma\bar{\zeta}=\Gamma\zeta\setminus\Gamma\zeta^L_{i,j}$ la care se va adăuga apoi subrețea specificată $\Gamma\zeta^W_{i,j}$ [3]. Ca rezultat al aplicării acestei reguli se va obține rețeaua $\Gamma\tilde{\zeta}=\Gamma\bar{\zeta}\cup\Gamma\zeta^w_{i,j}$, adică $\Gamma\zeta\ [r_{i,j}(M)>\Gamma\tilde{\zeta}$. La eliminarea locațiilor şi/sau tranzițiilor (regulilor de rescriere) ale $\Gamma\zeta^L_{i,j}$, arcele respective ce le conectează se vor elimina în mod implicit. De asemenea, la adăugarea lui $\Gamma\zeta^w_{i,j}$ în $\Gamma\bar{\zeta}$ toate locațiile cu marcaje noi şi/sau tranzițiile (regulile de rescriere) respective ce au aceleaşi nume şi aceleaşi atribute vor fi contopite. La contopirea locațiilor cu acelaşi nume jetoanele se vor aduna. Implicit, regulile $r_{i,j}(M):\Gamma\zeta^L_{i,j}\triangleright\varnothing$ şi $r_{i,j}(M):\varnothing\triangleright\Gamma\zeta^W_{i,j}$ descriu respectiv $\Gamma\tilde{\zeta}=\Gamma\zeta\setminus\Gamma\zeta^L_{i,j}$ şi $\Gamma\tilde{\zeta}=\Gamma\zeta\cup\Gamma\zeta^w_{i,j}$. Menționăm, de asemenea, că

$\Gamma\zeta_{i,j}^{L}$ şi $\Gamma\zeta_{i,j}^{w}$ pot fi considerate aparte şi ca submulțimi de: 1) locații P_L şi/sau P_w cu marcaje respective; 2) tranziții (respectiv reguli de rescriere) T_L (respectiv R_L) şi/sau T_W (respectiv R_W) şi 3) arce A_L şi/sau A_w de diferite tipuri.

Cel mai important avantaj al folosirii modelelor *RGRM* la descrierea şi verificarea funcționării proceselor de calcul orientate pe servicii constă în faptul că structura acestor tip de modele este foarte *concisă* şi poate fi *flexibil de modificat* în timp real, deoarece mai multe dintre atributele sale sunt *parametrizate*. Aceasta permite de a asocia atributelor *RGRM* valori alternative în mod controlat de starea curentă a modelului. De asemenea, prin utilizarea atributelor matriceale propuse, datele reale pot fi cu uşurință importate în procesul de simulare, asigurând astfel coerența şi valabilitatea rezultatelor obținute.

Modelarea acestor tipuri de procese prin *RGR* sau *RP* obişnuite [1, 3] duce la necesitatea de a folosi pentru fiecare atribut matriceal încă $2k \times n$ arce şi $k \times n$ tranziții (locații) suplimentare, ceea ce face ca complexitatea structurii modelului să crească în mod considerabil. Pentru a ilustra avantajele folosirii rețelelor de tipul *RGRM* în figura 2a este prezentat un exemplu simplu de rețea *RGRM*2 cu un arc direct matriceal (t_5, p_6) cu ponderea **A**, redată de relația (2) :

$$\mathrm{A}=\begin{bmatrix} 5+m_1 & 3 & 2 \\ 4 & 2+m_2 & 1+2m_2 \end{bmatrix} \qquad (2)$$

În acest model locațiile de control sunt $P^A = \{p_2, p_3\}$ cu $K_p(p_2) = 2$ şi $K_p(p_3) = 3$ respectiv. Selectarea rândului respectiv $i = m_2$ şi a coloanei $j = m_3$ ale elementului $a_{i,j}(M)$, $i = 1, 2;\ \ j = 1, 2, 3$ al matricei **A** este efectuată în mod dinamic de către marcajul curent al $\{p_2, p_3\}$.

Desfăşurarea acestui model în formă de rețea *RP*2 obişnuită, funcționarea căreia este echivalentă lui *RGRM*2, este prezentată în figura 2b. Pentru a efectua aceasta desfăşurare este necesar de a substitui în *RGRM*2 următoarele operații:

1) tranziția t_5 este substituită prin $k \times n = 6$ tranziții $t_{5,l}$, $l = 1,..., k \times n$, funcțiile de gardă ale cărora sunt respectiv:

$$g_{5,1}(M) = (m_2 = 2)\,\&\,(m_3 = 3),$$
$$g_{5,2}(M) = (m_2 = 2)\,\&\,(m_3 = 2),$$
$$g_{5,3}(M) = (m_2 = 2)\,\&\,(m_3 = 1),$$
$$g_{5,4}(M) = (m_2 = 1)\,\&\,(m_3 = 2),$$
$$g_{5,5}(M) = (m_2 = 1)\,\&\,(m_3 = 3),$$
$$g_{5,6}(M) = (m_2 = 1)\,\&\,(m_3 = 1);$$

2) arcul direct matricial (t_5, p_6) cu ponderea **A** este substituit prin $k \times n$ arce directe $(t_{5,l}, p_6)$, $l = 1,...,6$ cu ponderile respective ale elementelor matricei A din relația (2);

3) de a conecta locația p_5 cu fiecare tranziție introdusă $t_{5,l}$ prin arce $(p_5, t_{5,l})$;

4) locația p_2 (respectiv p_3) este conectată cu fiecare tranziție introdusă $t_{5,l}$ prin arce test $(p_2, t_{5,l})$ (respectiv $(p_3, t_{5,l})$).

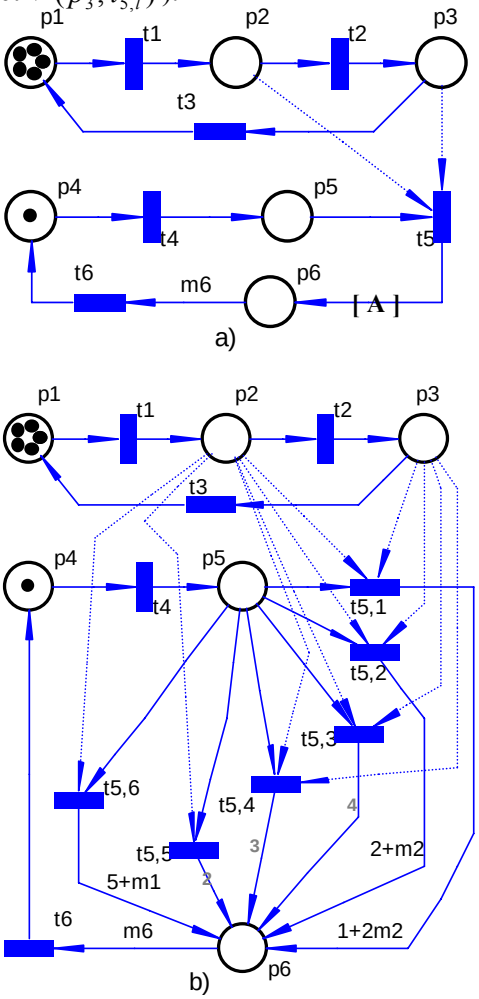

Figura 2. Modelul *RGRM*2 (a) şi modelul *RP*2 echivalent celui *RGRM*2 (b).

2. RGRM INTERPRETATE

Pentru a trata unele probleme menționate la modelarea şi analiza proceselor de calcul orientate pe servicii reconfigurabile, în acest compartiment definim un model de *RGRM interpretate* dinamic

reconfigurabile, notate *RGIM* , în care introducem o mulțime de tranziții de procesare $\zeta = \{\varsigma_1, ..., \varsigma_n\}$ a expresiilor matriceale ale atributelor şi o mulțime de reguli matriceale de rescriere $R = \{r_1, ..., r_k\}$ a rețelei *RGIM* curente ce pot modifica atât marcajul curent, cât şi structura ei la ocurența unor evenimente specificate. O tranziție de procesare $\varsigma_j \in \zeta$ şi o regulă de rescriere $r_j \in R$ sunt o generalizare a noțiunii de tranziție obişnuită $t_j \in T = \{t_1, ..., t_{|T|}\}$, folosită în sens clasic. Condiția de validare de către marcajul curent M a unei tranziții de procesare şi/sau a unei reguli de rescriere $r_j \in R$ este similară cu cea a unei tranziții obişnuite $t_j \in T$ în *RGR* .

Definiția 1**.** O multimulțime μ de elemente ale unei mulțimi finite nevide X este orice aplicație a lui X în mulțimea numerilor întregi naturale IN_0. Exprimăm $\mu = \sum_{x \in X} \mu(x).x$, unde $\mu(x)$ denotă numărul de apariții ale lui x în μ , $\mu(x) \in N_0$. Mulțimea tuturor multimulțimilor ale lui X este notată $Bag(X)$. Mulțimea de submulțimi ale lui X este notată prin $\wp(X)$. Pentru două multimulțimi μ_1, $\mu_2 \in Bag(X)$ operațiile adunarea (+), înmulțirea la un scalar (*), compararea ($\leq$), diferența (+) şi dimensiunea $|\mu|$ sunt definite în modul următor:

1. $(\mu_1 + \mu_2)(x) = \mu_1(x) + \mu_2(x), \forall x \in X$;
2. $(n * \mu)(x) = n * \mu(x), \forall n \in IN_0, \forall x \in X$;
3. $\mu_1 \leq \mu_2 \Leftrightarrow \mu_1(x) \leq \mu_2(x), \quad \forall x \in X$;
4. $(\mu_2 - \mu_1)(x) = \mu_2(x) - \mu_1(x), \ \forall x \in X$, $\mu_1 \leq \mu_2$; 5)$|\mu| = \sum_{x \in X} \mu_1(x)$. ■

Exemplul 1. Fie $X = \{a, b, d, e\}$, iar $\mu_1 = \{a, a, a, b, b, e, e\}$ şi $\mu_2 = \{a, a, b, d, d, d, e\}$ sunt respectiv două multimulțimi pe X astfel încât: $\mu_1(a) = 3, \mu_1(b) = 2, \ \mu_1(d) = 0, \mu_1(e) = 2$ şi $\mu_2(a) = 2, \mu_2(b) = 1, \ \mu_2(d) = 3, \mu_2(e) = 1$. Aceste multimulțimi μ_1 şi μ_2 sunt notate şi ca o sumă formală redate astfel: $\mu_1 = 3a + 2b + 2e$ şi $\mu_2 = 2a + b + 3d + e$, iar suma lor este $\mu_1 + \mu_2 = 5a + 3b + 3d + 3e$.

Cu fiecare *tranziție de procesare* $\varsigma_j \in \zeta$ sunt asociate aserțiuni, expresii, mesaje, funcții marcaj dependente de tipul specificat, expresii descriptive ale subrețelelor ce trebuie să fie procesate de ς_j, la declanşarea acesteia obținându-se rezultatul specificat. Tipul de date este descris prin declarații care este similar cu cel al unei rețele Petri colorate şi al unui limbaj de programare orientat pe obiecte. Pe aceste tipuri de date se definesc o mulțime de operații, rezultatul fiind o algebră multisortată având ca domenii aceste tipuri. De exemplu, mulțimea finită, utilizator-extensibilă, de tipuri de date de bază ar putea fi:

y::= *bool* | *integer*| *real*| *char*| *string*| *XML* | *net*|...,

unde *bool* conține valorile booleene adevărat sau fals, *integer* şi *real* conțin toate numerele întregi şi respectiv cele reale, *string* conține toate şirurile de caractere *char* şi XML conține toate documentele XML bine formate, iar *net* expresii descriptive ale subrețelelor posibile tip $RGIM_j$, $j = 1,...,n_y$ [3]. Deşi acest set de tipuri de date poate fi ales în mod arbitrar, vom cere ca el, cel puțin, să le conțină pe cele menționate. Din aceste tipuri de bază, putem construi şi tipuri de date mai complexe.

Fie $X \rho Y$ este o relație binară ρ între două mulțimi X şi Y, în care *domeniul* lui ρ este $Dom(\rho) = \rho Y$, iar *codomeniul* este $Cod(\rho) = X\rho$.

Definiția 2**.** O rețea *RGIM* , abreviat $\Gamma\zeta$, este o structură de obiecte constituită dintr-un 13-tuplu:

$\Gamma\zeta = < P, E, \theta$, *Pre*, *Post*, *Test*, *Inh*, *Pri*, G_E, G_R, Kp , ϕ , M_0 >, unde :

- $P = \{p_1, ..., p_{|P|}\}$ este mulțimea nevidă de locații; E este mulțimea nevidă de evenimente discrete, constituită din $E = T \cup \zeta \cup R \neq \varnothing$, $T \cap \zeta \cap R = \varnothing$, astfel încât $P \cap E = \varnothing$, unde $T = \{t_1, ..., t_{|T|}\}$ (respectiv $\zeta = \{\varsigma_1, ..., \varsigma_{n_\varsigma}\}$) este mulțimea *tranzițiilor obişnuite* (resp. *de procesare*), declanşarea cărora pot să modifice numai marcajul curent (resp. să determine noi date de tipul specificat, obținute la *procesarea expresiilor, de tipul respectiv, asociate cu* ζ şi să modifice marcajul curent) al rețelei, iar $R = \{r_1, ..., r_k\}$, este mulțimea *regulilor de rescriere*, care poate să modifice în mod dinamic marcajul curent şi/sau structura cu toate atributele rețelei curente. Grafic (vezi figura 3), tranzițiile obişnuite sunt reprezentate prin bare groase, tranzițiile *de procesare* prin dreptunghiuri cu o săgeată în ele, iar regulile de rescriere sunt reprezentate prin dreptunghiuri imbricate;
- $\theta = \{\vartheta_1, \vartheta_2, ..., \vartheta_{n_\varsigma}\}$ este mulțimea finită nevidă a tipurilor de date, definită pentru funcții

matriceale marcaj dependente θ_ζ ale mulțimii ζ, astfel încât $\theta_\zeta : \zeta \to \wp(\theta)$;

- Arcele sunt redate de funcții matriceale marcaj - dependente de *inhibiție Inh, Test* şi cele de incidență *înainte* (respectiv *înapoi*) *Pre*: $E \times P \times IN_+^{|P|\times|P|} \to IN_+$ (respectiv *Post*: $P \times E \times IN_+^{|P|\times|P|} \to IN_+$). IN_+ este mulțimea numerilor întregi naturale nenule;
- *Pri*: $E \times IN_+^{|P|\times|P|} \to IN_+$ este funcția matriceală de ordonare parțială a lui *E*, care introduce priorități dinamice de declanşare a evenimentelor validate de marcajul curent. Implicit, prioritățile ce nu sunt menționate ale unor evenimente $e_j \in E$ sunt considerate nule;
- Funcțiile $G_E : E \times IN_+^{|P|\times|P|} \to \{true, false\}$ şi $G_R : (\zeta \cup R) \times IN_+^{|P|\times|P|} \to \{true, false\}$ sunt respectiv nişte funcții de gardă matriceale (eng. *Guard-function*), care pentru orice eveniment $e_j \in E$ şi respectiv $\varsigma_j \in \zeta$, $r_k \in R$ determină respectiv funcții Booleene $g_j^E(M)$, $g_j^\zeta(M)$ şi $g_k^R(M)$ în marcajul curent *M*. Astfel, dacă evenimentul e_j este validat de marcajul curent M, notat $M[e_j>$, relativ la arce şi $g_j^E(M)$='*true*', atunci evenimentul e_j rămâne validat şi, eventual, el poate fi declanşat, iar dacă $g_j^E(M)$='*false*' - acest eveniment nu este validat. Implicit $g_j^E(M)$='*true*'. În cazul în care evenimentul e_j validat este o tranziție *obişnuită* sau o tranziție de procesare cu $g_j^\zeta(M)$='*false*' sau o regulă de rescriere cu $g_j^R(M)$='*false*', atunci declanşarea acestui eveniment va *shimba numai* marcajul curent al rețelei $\Gamma\zeta$. În cazul în care $e_j = \varsigma_j$ este validată şi $g_j^\zeta(M)$='*true*' atunci tranziția de procesare ς_j la declanşare va schimba atât marcajul curent cât şi datele curente, de tipul respectiv, ca rezultat al procesării expresiilor şi aserțiunilor asociate cu acest eveniment. Însă dacă $e_j = r_j$ şi $g_j^R(M)(c)$='*true*' acesta va modifica atât sructura cu unele atribute curente ale $\Gamma\zeta$, cât şi marcajul ei curent în conformitate cu specificațiile acestei reguli. Implicit, funcțiile de gardă respective ce nu sunt menționate ale unor evenimente $E' \subseteq E$ şi $E'' \subseteq \zeta \cup R$ sunt considerate ca constanta '*true*';
- $Kp: P \times IN_+^{|P|\times|P|} \to (IN_+ \cup +\infty)$ este funcția matriceală de capacitate a locațiilor, încât $\forall p_i \in P$ aceasta este redată de capacitatea maximă de jetoane care pot să se afle în locația p_i, $0 < Kp(p_i) < +\infty$. Implicit, $Kp(p_i)$ este nelimitată;
- $\phi : E \to \{T, \zeta, R\}$ este funcția care indică tipul de eveniment validat de către marcajul curent al rețelei, adică el este de tipul *t*, $e \in T$ sau $e \in \zeta$, declanşarea căruia modifică *numai marcajul curent* sau de tipul $r \in R$, care va modifica atât *structura rețelei cu atributele sale*, cât şi *marcajul ei curent*;
- $M_0 : P \to Bag(P)$ este marcajul inițial ce determină o funcție de marcare definită pe mulțimea locațiilor *P*, astfel încât:

$\forall p \in P,\ M(p) \in Bag(P)$. ■

În figura 3 sunt prezentate primitivele rețelelor de tipul $\Gamma\zeta$.

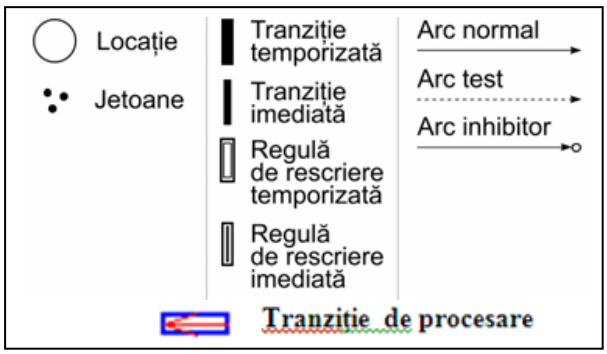

Figura 3. Primitivele rețelelor de tipul $\Gamma\zeta$.

3. MODELAREA SCOS PRIN RGIM STOCASTICE

SCOS este o paradigmă de organizare şi utilizare a capacităților resurselor de calcul distribuite, care pot fi aplicate şi controlate in domenii cu diferite proprietăți. În pofida faptului că implementările sunt specifice acestea sunt întotdeauna construite în baza următoarelor concepte majore: servicii (referitor la implementări); descrierea serviciilor; operațiuni (interacțiuni) cu servicii.

Serviciile SCOS sunt efectuate printr-un schimb de mesaje de interacționare între solicitanții şi furnizorii de servicii. Furnizorii de servicii publică descrierile lor într-un registru de servicii. Clienții foc cunoştință cu aceste descrieri, apoi în baza condițiilor invocate accesează aceste servicii. Arhitectură SCOS este bazată pe interdependența a trei entități distincte: furnizorul de servicii, consumatorul de servicii şi registrul de servicii.

Rețelele Petri, fiind un formalism cu o semantica formală bine fundamentată, sunt folosite cu succes pentru modelarea proceselor cu

evenimente discrete [1, 2, 6, 8]. Ele sunt deosebit de potrivite pentru modelarea proceselor cu proprietăți de concurență, sincronizare şi non-determinism.

Serviciile furnizate de SCOS sunt exact procese de acest tip, deoarece fluxurile de execuție pot fi văzute ca seturi de stări parțial ordonate ale serviciilor şi operații ale acestor service. Mai mult, în astfel de modele de servicii stările ar trebui să alterneze cu operațiile. O structură de serviciu SCOS include două tipuri de seturi cu elemente care pot fi conectate numai cu elementele ce aparțin altui tip de set şi vice-versa. Evident, astfel de proprietăți structurale sunt exact cele pe care le oferă formalismul de rețele $RGIM$.

Definiția 3**.** O rețea *RGIM stocastică extinsă*, abreviat *RIMS,* este sistemul redat de către tripletul de obiecte $S\Gamma\zeta =< \Gamma\zeta, \Lambda, \omega >$, unde:

- $\Gamma\zeta$ este o rețea *RGIM* temporizată, definită în conformitate cu Definiția 2, în care $E = E_\tau \cup E_0$, $E_0(M) \cap E_\tau(M) = \varnothing$. Aici E_τ este mulțimea evenimentelor temporizate cu o durată aleatorie de declanşare ce are o distribuție exponențial–negativă (grafic sunt reprezentate prin dreptunghiuri groase), iar E_0 este mulți-mea evenimentelor imediate cu o durată de declanşare nulă (grafic sunt reprezentate prin bare subțiri), astfel încât $\Pr i(E_0) > \Pr i(E_\tau)$. Acestea descriu selectoare probabilistice;
- $\Lambda : E_\tau \times IN_+^{|P|\times|P|} \to IR^+$ este funcția matriceală ce determină rata $0 \le \lambda(t, M) < +\infty$ de declanşare a evenimentului validat în marcajul curent *M*. IR^+ este mulțimea mărimilor reale nenegative.
- $\omega : E_0 \times IN_+^{|P|\times|P|} \to IR^+$ este funcția matriceală de ponderе $0 \le \omega(t, M) < +\infty$ ce determină probabilitatea de declanşare a evenimentelor imediate validate de marcajul curent *M*. ■

Astfel, fluxurile de prelucrare ale unui modul de serviciu simplu pot fi descrise de către o subrețea Petri membranală $SM_i = [_i\ S\Gamma\zeta_i\ ]_i^{Q_i}, i = 1,..., ks$ [3], numită *modul rețea de servire*, în care atributul $Q_i = \{q_1,.., q_{|Q|}\}$ reprezintă mulțimea de indicatori ai calității de serviciu QoS furnizați de SM_i, de exemplu: q_1 - fiabilitatea ; q_2 - disponibilitatea; q_3 - durata medie de răspuns; q_3 -costul serviciului, etc.

In continuare, pentru a facilita descrierea proceselor de servire ale SCOS, locațiile şi tranzițiile respective ale fiecărui SM_i vor fi redate în modul următor: $p_{i,l} \in P_i$, $l = 1,..., ni$, $ni = |P_i|$ şi $e_{i,j} \in E_i$, $j = 1,..., ki$, $ki = |E_i|$ cu $i = 1,..., ks$. De asemenea, vom considera SM_i ce are numai o singură locație de intrare (locație start) $p_{i,1}$ care corespunde stării inițiale a serviciului şi numai o singură locație de ieşire (locație finală) $p_{i,\,ni}$ care corespunde stării finale a acestui serviciu. Locația $p_{i,1}$ nu are arce de intrare, iar cea $p_{i,\,ni}$ nu are nici un arc de ieşire asociat cu această locație. De obicei, o locație de intrare este asociată cu starea serviciului în care el este gata să înceapă derularea lui, adică pentru $M(p_{i,1}) = 1$, $M(p_{i,\,l}) = 0$, $l = 2,..., ni$. În mod obişnuit, o locație de ieşire reflectă starea în care serviciul este finalizat, adică pentru $M(p_{i,ni}) = 1$, $M(p_{i,\,j}) = 0$, $j = 1,..., ni-1$. Locația de intrare şi cea de ieşire trebuie să fie totdeauna conectate prin intermediul unor stări tranzitorii, fie direct sau indirect, pentru a se asigura că serviciul este realizabil structural. De asemenea, vom cere ca *rețeaua de servire* complementară $\overline{SM}_i$, obținută prin adăugarea la serviciul SM_i a unei tranziții $t_{i,ni+1}$ şi a două arce $(t_{i,ki+1}, p_{i,1})$ şi $(p_{i,\,ni}, t_{i,\,ki+1})$ cu $M_0^i = (p_{i,1})$ trebuie să fie mărginită (sigură), viabilă şi reversibilă [3]. O astfel de organizare a unui SM_i este numită formă standard, care este deosebit de potrivită pentru a descrie funcționarea unui SCOS cu aplicații reconfigurabile.

O stare a serviciului curent este activată (finalizată), atunci când există un jeton în locația sa corespunzătoare a unui SM_i. Prezența unui jeton în locația de intrare sau a celei de ieşire a serviciului SM_i reflectă respectiv inițierea şi finalizarea acestui serviciu. Dacă locația de intrare la un eveniment are un număr necesar de jetoane, acesta este validat şi el poate să declanşeze. La declanşarea acestui eveniment el va elimină (va depune) respectiv un număr specificat de jetoane din locația sa de intrare (în locația sa de ieşire) sau îl va reconfigura.

În figura 4 este prezentat un model de rețea $S\Gamma\zeta 1$ care descrie funcționarea unui SCOS ce furnizează un set de servicii simple, redate de către o matrice $SM = [SM_j]_{1\times n}$, $j = 1,..., n$.

Semnificațiile nodurilor modelului $S\Gamma\zeta 1$ sunt:

- p_1 - utilizatori de servicii; p_2 - cereri de servicii formate; p_3 - cerere de serviciu recepționată de către server; p_4 - activarea mecanismului de formare a următorului SM_i; p_5 - activarea verificării schimbării serviciului curent; p_6 - semaforul ce asigură deservirea ordonată a cererilor; p_7 -

activarea regulii de schimbare a SM_j în SM_i; p_8 - răspunsul serviciului SM_j; p_9 - serverul este liber; p_{10} - indicator sfârșit al serviciului furnizat de $SM1$; p_{11} - inițiere controlor al mecanismului de generare a cererilor de servicii; p_{12} - indicator de selectare al următorului tip de serviciu; $p_{j,1}$ - inițierea prelucrării serviciului curent SM_j; $p_{j,nj}$ - sfârșit al prelucrării serviciului curent SM_j.

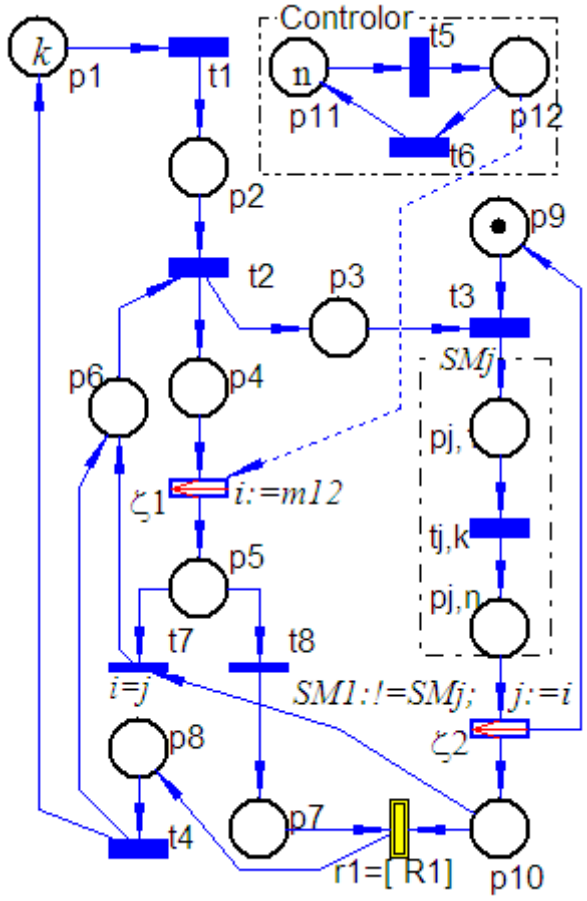

Figura 4. Modelul $S\Gamma\zeta 1$ funcționării unui SCOS ce furnizează un set de servicii simple.

- t_1 - generarea cererilor de servicii; t_2 - transmiterea cererii de servicii către server și procesarea pregătirii următorului tip de serviciu; t_3 - cererea de serviciu SM_j este recepționată și verificată de către server; t_4 - finalizarea procesării răspunsului serviciului SM_j; t_5 - generarea tipului de serviciu cerut; t_6 - setarea controlorului de generare a cererilor de servicii; t_7 - verifică dacă utilizatorul cere iarăși serviciul ce este în curs de procesare (funcția de gardă $g_7^E = (i = j)$); t_8 - verifică dacă utilizatorul cere un serviciu diferit de cel în curs de procesare (funcția de gardă $g_7^E = (i \neq j)$); $t_{j,k}$ - prelucrarea curentă a serviciului redat de SM_j.

- ς_1 - procesarea expresiei $i := m_{12}$ de tipul integer; ς_2 - finalizarea procesării serviciului SM_j, eliberarea serverului și transmiterea confirmării; procesarea expresiilor: $j := i$ de tipul integer și $SM1 :!= SM_j$ de tipul *rețea de servire* în care rețeaua fiind redată de expresia descriptivă a SM_j [3] este atribuită lui $SM1$.

- $r_1 = [R1]$, regula de rescriere matriceală a $S\Gamma\zeta 1$ cu $R1 = [r_i^1]_{1\times n}$, elementele matricei căreia sunt: $r_i^1 : SM1 \triangleright (|_{t_3}\ p_{i,1} \vee SM_i \vee p_{i,ni}\ |_{t_4})$, $i = 1,..., n$ cu $g_i^{R1} =$"*true*" (vezi [3]). La declanșarea lui r_i^1 din rețeaua curentă $S\Gamma\zeta 1$ se va *elimina* subrețeaua redată de expresia descriptivă $SM1$, obținându-se astfel o nouă rețea $S\Gamma\zeta 1' = S\Gamma\zeta 1 \setminus SM1$, iar la aceasta se va adăuga subrețeaua redată de expresia descriptivă $SM' = |_{t_3}\ p_{i,1} \vee SM_i \vee p_{i,ni}\ |_{t_4}$, luându-se în considerație faptul că nodurile ce au același nume și atribute se vor contopi, iar numărul de jetoane în locațiile respective se vor aduna. În rezultatul rescrierii $S\Gamma\zeta 1$ de către r_i^1 se va obține o nouă configurație de rețea $S\Gamma\zeta 1[r_i^1 > S\Gamma\zeta 1''$, unde $S\Gamma\zeta 1'' = S\Gamma\zeta 1' \cup SM'$.

Atributele locațiilor $S\Gamma\zeta 1$ sunt: $Kp(p_i) = 1$, $i = 3,..., 10$; $M_0 = (kp_1, p_8, p_9, np_{11})$.

Ratele de declanșare ale evenimentelor temporizate (tranziții obișnuite, tranziții de procesare și reguli de rescriere) sunt redate în mod standard.

Mulțimea de evenimente validate de marcajul inițial al modelului de rețea $S\Gamma\zeta 1$ este $E(M_0) = \{t_1, t_5\}$. La declanșarea tranziției obișnuite t_1 se va genera o cerere de serviciu, redată de apariția unui jeton în locația p_1, care la rândul său validează tranziția obișnuită t_2. Declanșarea tranziției obișnuite t_5 a controlorului va selecta următorul modul de serviciu SM_i, indicele $i = m_{12}$ căruia este procesat de ς_1. La declanșarea lui t_2 se vor depune câte un jeton în locațiile p_3 și p_4 care vor valida respectiv tranziția obișnuită t_3 și tranziția de procesare ς_1. Declanșarea lui t_3 va iniția începutul prelucrării modulului de serviciu curent SM_i prin apariția unui

jeton in locația $p_{j,1}$. Terminarea serviciului SM_i este indicată de apariția unui jeton în locația $p_{j,nj}$ care va valida ς_2. În cazul declanşării tranziției de procesare ς_2 ea va procesa expresiile: $j := i$ de tipul integer şi $SM1 :!= SM_j$ de tipul *rețea de servire* în care rețeaua curentă, fiind redată de expresia descriptivă SM_j, este atribuită lui $SM1$. De asemenea, la declanşarea lui ς_2 ea va depune câte un jeton în locațiile p_9 şi p_{10} care vor valida regula matriceală de rescriere $r_1 = [R1]$ a modulului de servire SM_j curent prelucrat prin următorul SM_i, $i := m_{12}$ selectat în prealabil de către ς_1 cu $g_i^{R1} =$"*true*". Tranzițiile imediate t_7 şi t_8 redau selectorul probabilistic care determină dacă este necesar de furniza acelaşi modulul de servire SM_j în cazul în care $g_7^E = (i = j)$.

Rețeaua $S\Gamma\zeta 1$ este mărginită, viabilă şi reinițializabilă.

4. CONCLUZII

Rețelele *RIMS* propuse sunt foarte eficiente la redarea, modelarea, verificarea şi analiza performanțelor sistemelor SCOS, deoarece folosirea acestora au următoarele avantaje: 1) au caracteristici suplimentare de vizualizare a procedurilor de modelare şi simulare, care permit de a realiza un mediu puternic de validare şi evaluare; 2) permit de a reprezintă în acelaşi model atât schimbarea dinamică a atributelor şi stărilor sistemului, precum şi schimbarea dinamică a structurii sale; 3) cu utilizarea atributelor matriceale propuse, datele reale pot fi cu uşurință importate în procesul de simulare, asigurând astfel coerența şi valabilitatea rezultatelor obținute.

Metoda propusă este generală şi poate fi aplicată la o gamă largă de tipuri de sisteme cu evenimente discrete. În plus, cu destul de puține modificări şi completări, utilizarea abordării descrise poate fi în continuare generalizată pentru a studia sisteme din domenii cu caracteristici similare. Din analiza prezentată sa arătat că *RIMS* definite şi studiate în această lucrare pot fi folosite ca un instrument foarte promițător pentru modelarea, evaluarea fiabilității şi performanțelor SCOS.

Aplicabilitatea acestui demers este ilustrată printr-un exemplu al proceselor de calcul orientat pe servicii reconfigurabile în care sunt prelucrate diferite tipuri de cereri ale diferitor servicii.

În lucrările pe viitor, vom elabora şi dezvolta un produs program instrumental pentru simularea vizuală şi analiză a modelelor de rețele *RIMS* ce descriu funcționarea proceselor de calcul orientate pe servicii reconfigurabile.

Bibliografie

1. Ding, Z. J., Wang, J. L., Jiang, C. J*. An Approach for Synthesis Petri Nets for Modeling and Verifying Composite Web Service. Journal of information science and engineering 24, pp. 1309-1328, 2008.*

2. Guțuleac, E., Țurcanu, Iu., Palii, D. *Aspecte de modelare şi analiză a proceselor de calcul orientate pe servicii reconfigurabile// Proceedings of the 6-th International Conference on Microelectronics and Computer Science, ICMCS 2011 Chişinău, R. Moldova, September 22-24, p. 187-19,12011.*

3. Guțuleac, E. *Descriptive Timed Membrane Petri Nets for Modeling of Parallel Computing. International Journal of Computers, Communications & Control, No. 3, Vol. I, Agora University Editing House, Oradea, România, pp. 33-39, 2006.*

4. Hamadi, R., Benatallah, B. *A Petri net-based model for Web service Composition// In Proceedings of 14th Australian Database Conference on Database Technologies, Vol. 17, Adelaide, Australia, pp. 191–200, 2003.*

5. Katsigiannis, Y. A., Georgilakis, P. S. Tsinarakis, G.J. *Introducing a coloured fluid stochastic Petri net-based methodology for reliability and performance evaluation of small isolated power systems including wind turbines. IET Renewable Power Generation, Vol. 2, No. 2, pp. 75–88, 2008.*

6. Li, B., Xu, Y., Wu, J., Zhu, J. *A Petri-net and QoS Based Model for Automatic Web Service Composition. Journal of Software, Vol. 7, No. 1, pp. 149-155, 2012.*

7. Mtibaa, S., Tagina, M. *A Petri Nets-based Conceptual Framework for Web Service Composition in a Healthcare Service Platform. Journal of Telecommunications, Vol. 2, No. 4, pp.836-840, 2012.*

8. Todica, V., Cremene, M., Vaida, M. A *Framework for Developing Complex Systems of Services, Coping with Complexity// COPCOM2011, Cluj-Napoca, 19-20 oct. pp. 77-88, 2011.*

Recomandat spre publicare: 28.11.2013.