VPNS SISTEM DE MODELARE ŞI ANALIZĂ A PERFORMANŢELOR MODELELOR DE REŢEA PETRI

DIANA Palii, EMILIAN Guţuleac

Universitatea Tehnică a Moldovei

Abstract: În lucrarea de faţă ne propunem, utilizînd limbajul de modelare vizual (UML), să analizăm şi să dezvoltăm un software, ce ne-ar permite modelarea şi analiza performanţelor modelelor de reaţea Petri. Cu ajutorul diagramelor cazurilor de utilizare au fost specificate cerinţele din perspectiva utilizatorului final iar prin intermediul diagramei de activităţi au fost modelate fluxurile de activităţi aşa cum sunt vazute ele de actorii care comunică cu sistemul VPNS.

Cuvinte cheie: *Reţele Petri, graf de marcaje accesibile, matrice de incidenţă, simulare animată, analiza performanţelor, diagrama cazurilor de utilizare, diagrama de activitate*

1. Introducere

Reţelele Petri reprezintă un formalizm de descriere şi analiză a sistemelor paralele/distribuite cu evenimente discrete. În prezent ele au numeroase aplicaţii şi sunt utilizate în diverse domenii: sistemelor de producţie, inginerie, telecomunicaţii, modelarea proceselor de afaceri şi în învăţământ, deoarece dispun de o reprezentare grafică foarte accesibilă, au o semantică bine definită care permite o analiză formală a comportamentului şi proprietăţilor acestora

Utilizarea eficientă a resurselor de calcul joacă un rol important în reducerea cheltuielilor de elaborare şi producţie ale sistemelor hardware/software reconfigurabile, iar producătorii acestora devin mai competitivi folosind formalisme şi instrumente adecvate pentru a palia aceste probleme.

Reţelele Petri prezintă, de asemenea, un mare interes datorită clarităţii grafice şi intuitive de reprezentare a fluxului controlului într-un sistem cu activităţi interdependente, iar extensiile lor la RP stocastice permit studiul sistemelor de calcul sub aspectul evaluării performanţelor.

2. Reţelele Petri noţiuni

2.1. Noţiuni introductive

O *reţea Petri* (eng. *Petri net*) se compune dîntr-un tip particular de *graf orientat* notat N şi o *stare iniţială* M_0, denumită *marcaj iniţial* (eng. *iniţial marking*). Graful N al reţelei Petri este orientat, ponderat şi bipartit, constînd din două tipuri de noduri, denumite *poziţii* sau *locaţii* (eng. *place*) şi respectiv *tranziţii* (eng. *transition*); *arcele* pleacă fie de la o pozite la o tranziţie, fie de la o tranziţie la o poziţie. (Nu exista arce care sa conecteze doua poziţii între ele, sau doua tranziţii între ele.) Ca simbolizare grafică, poziţiile se reprezintă prin cercuri, iar tranziţiile prin bare sau dreptunghiuri. Arcele sunt etichetate cu ponderile lor (valori întregi, pozitive); un arc cu ponderea k poate fi privit ca o mulţime de k arce paralele cu o pondere unitară. Etichetele pentru pondere unitară se omit în reprezentările grafice uzuale. Un *marcaj* sau o *stare* atribuie fiecare pozitii un număr întreg mai mare sau egal cu 0. Dacă un marcaj atribuie poziţiei p întregul $k > 0$, se spune că p este marcat cu k *jetoane* (eng. *token*). Din punct de vedere grafic, în cercul corespunzător poziţiei p se vor plasa k discuri. Orice marcaj M este *un vector* m-dimensional, unde m notează numărul total al poziţiilor; componenta p a lui M, notată $M(p)$ semnifică numărul de jetoane din poziţia p.

În problemele de modelare ce utilizează conceptele de *condiţii* şi *evenimente*, poziţiile reprezintă conditii şi tranziţiile reprezintă evenimente. O tranzitie (eveniment) posedă un număr de pozitii de intrare şi ieşire, care reprezintă *pre-condiţii* şi respectiv *post-condiţii* pentru evenimentul în cauză. Prezenta unui jeton într-o pozitie trebuie înţeleasă ca valoare logică "adevărat" pentru condiţia asociată respectivei pozitii.

2.2. Definiții de bază

O retea Petri este un cvintuplu, $PN = (P, T, F, W, M_0)$ în care:

P={p1,p2,…,pm} este o multime finită de poziții;
T={t1,t2,…,tn} este o multime finită de tranziții;
F⊆(PxT)∪(TxP) este o multime de arce;
W:F→{1,2,3,…} este o funcție de ponderare a arcelor;
M_0:P→{0,1,2,3,…} este o funcție de marcaj inițial.

Multimile P şi T sunt disjuncte P∩T={∅} şi satisfac condiția P∪T≠{∅}. O retea Petri cu un marcaj initial M_0 se va nota prin (N, M_0) iar o rețea Petri cu un marcaj oarecare M se va nota prin (N, M).

Marcajul unei retele Petri are semnificatie de stare a retelei şi se poate modifica în conformitate cu următorul procedeu denumit regula tranzitiei (validare şi executare):

a) O tranziție t se spune că este validată (eng. enabled) dacă fiecare poziție de intrare p a lui t este marcată cu cel putin W(p, t) jetoane, unde W(p, t) notează ponderea arcului de la p la t.

b) O tranzitie validată poate sau nu să fie executată sau declansată (eng. fire), după cum evenimentul asociat tranziției are sau nu loc.

c) Executarea unei tranziții validate t, îndepărtează W(p, t) jetoane din fiecare poziție de intrare p a lui t şi adaugă W(t, p) jetoane la fiecare poziție de ieşire p a lui t, unde W(t, p) este ponderea arcului de la t la p.

O tranzitie fără nici o poziție de intrare se numeste tranziție sursă (eng. source). O tranziție fără nici o poziție de ieşire se numeste tranziție receptor (eng. sink). Modul de operare al acestor tranziții este următorul:

a) O tranziție sursă este necondiționat validată (fără a fi obligatoriu ca să se execute). Executarea ei produce jetoane.

b) Executarea unei tranziții receptor consumă jetoane, fără a produce.

Dacă o poziție p este atît poziție de intrare, cît şi poziție de ieşire pentru o tranziție t, atunci p şi t formează o buclă autonomă (eng. self-loop). O rețea Petri care nu conține bucle autonome se numeste pură. O retea Petri se numeste ordinară dacă toate arcele sale au pondere unitară.

2.3. Proprietăți

Ca instrument matematic, rețelele Petri posedă un număr de proprietăți. Aceste proprietăți, atunci cînd sunt interpretate în contextul sistemului modelat, permit designer-ului de sistem să identifice prezența sau absența proprietăților funcționale specifice domeniului de aplicare al sistemului proiectat. Astfel, pot fi distinse două tipuri de proprietăți: *comportamentale* şi *structurale*. Proprietățile comportamentale sunt acelea care depind de starea inițială, sau marcajul, unei rețele Petri iar proprietățile *structurale* depind numai de topologia acesteia. Proprietățile comportamentale sunt accesibilitate, mărginire, viabilitate, reversibilitate pe cînd cele structurale sunt viabilitate structurală, controlabilitate, mărginire structurală, conservativitate, repetitivitate, consistență.

3. Analiza şi proiectarea sistemului VPNS

Utilizînd limbajul de modelare vizual (UML), ne propunem să concepem şi să dezvoltăm un software, ce ne-ar permite modelarea şi analiza performanțelor modelelor de reațea petri.

Prin intermediul cazurilor de utilizare, pentru a crea o imagine generală asupra sistemului vom specifica cerințele din perspectiva utilizatorului final. Pe baza unui singur use case, este posibilă generarea de scenarii multiple, fiecăruia dintre ele corespunzându-i o singură modalitate de a atinge obiectivul. În continuare vom folosi diagramele use case pentru prezentarea cazurilor de utilizare ale sistemului propus.

Interacțiunea cu sistemul o realizează utilizatorul, ca parte componentă a sistemului. Această interacțiune prevede crearea/editarea modelelor de rețele petri, posibilitatea de a seta parametrii rețelei şi simularea acesteia avînd la dispoziție un set vast de proprietăți a componentelor modelului de rețelea creată. Utilizatorul mai dispune de posibilitatea de a salva datele statistice ale rezultatului simulării precum şi unele diagrame funcționale care primesc ca parametri propietățile obectelor din rețea.

Sistemul (VPNS) realizează interacțiunea cu utilizatorul oferindui instrumentariu necesar creării rețelelor şi totodată posibilitatea de a seta proprietățile acestora. La nivelul de prezentare ca răspuns la acțiunile utilizatorului sistemul reflectă orice schimbare a modelului rețelei, inclusiv simularea modelului rețelei.

Descrierea comportamentului sistemului în raport cu o cerere din partea unui actor din afara acestui sistem este prezentată în fig.1.

Figura 1 Diagrama Use Case ce reprezintă interacțiunea dintre User şi System (VPNS)

Actorul în cazul nostru este utilizatorul. Din diagrama prezentată se observă toate funcționalitățile sistemului, care sunt bazate pe schimb de mesaje dinte utilizator şi sistem, adică ce trebuie un sistem să facă pentru un actor în scopul de a atinge un anumit obiectiv.

Funcționalitatea Salvare/Export Rețea

Scenariu de bază – Salvarea rețea:

1. Utilizatorul alege comanda "Save" din meniul File sau de pe meniul principal al aplicației; 2. Conținutul fişierului rețelei este suprascris.

Scenariu de bază – Export rețea:

1. Utilizatorul alege comanda "Export" din meniul File sau de pe meniul principal al aplicației.
2. Conținutul fişierului (modelul de rețea) este salvat în format grafic „*.png”.

Funcționalitatea Crearea / Editarea Rețelei

Scenariu de bază – Crearea:

1. Utilizatorul alege comanda "New" din meniul File al formei de baza a aplicației sau pictograma New de pe meniul principal al programei.
2. În fereastra de dialog apărută utilizatorul alege calea şi introduce denumirea noului fişier (rețelei noi).
3. Rețeaua este creata şi salvată la denumirea rețelei de bază se adaugă extensia fişierului rețelei ".pn".

Scenariu alternativ - CreareaAnulare:

a.2. În fereastra de dialog apărută utilizatorul alege butonul Cancel.
3. Fereastra de dialog se închide şi nu se produce nici o schimbare.

Funcționalitatea Analiza proprietăților structurale

Scenariu de bază - Analiza proprietăților structurale:

1. Utilizatorul alege comanda "Structural Analyse" de pe meniul principal al aplicației.
2. Utilizatorul selecteză funcționalitatea generării matricii de incidență sau determinarea P,T invarianților rețelei.
3. La selectarea opțiunii dorite sistemul calculează şi afişează rezultatele.
4. Utilizatorul selecteză opțiunea save şi rezultatele sunt salvate intr-un fişier.

Scenariu alternativ –Analiza proprietăților structurale Anulare:

a.2 În fereastra de dialog apărută utilizatorul alege butonul Cancel.
3. Fereastra de dialog se închide şi nu se produce nici o schimbare.

Funcționalitatea Simulare Animată

Scenariu de bază – Simulare Animată:

1. Utilizatorul alege comanda "Run" de pe meniul principal al aplicației.
2. Utilizatorul efectueaza setările necesare simulării.
3. Utilizatorul startează simularea animată.
4. Utilizatorul opreşte simularea.
5. Utlizatorul închide forma de dirijare a simulării.
6. Aplicația revine la forma inițială.

Scenariu alternativ – Simulare Animată Anulare:

a.2. Utlizatorul închide forma de dirijare a simulării.
3. Fereastra de dialog se închide şi nu se produce nici o schimbare.

Funcționalitatea Generarea Grafului de marcaje accesibile (GMA)

Scenariu de bază – Generarea Grafului de marcaje accesibile:

1. Utilizatorul alege comanda "Reachability graph" de pe meniul principal al aplicației.
2. Utilizatorul efectuează setările necesare.
3. Utilizatorul startează build graph.
4. Sistemul afişează graful de marcaje accesibile.
5. Utilizatorul alege opțiunea de analiză a proprietăților comportamentale „Behavioural analysis”.
6. Sistemul afişează proprietăților comportamentale ale modelului analizat.
7. Utlizatorul închide forma de dirijare a simulării.
8. Aplicația revine la forma inițială.

Scenariu alternativ – Generarea Grafului de marcaje accesibile Anulare:

a.2. Utlizatorul închide forma de dirijare a grafului de marcaje accesibile.
3. Fereastra de dialog se închide şi nu se produce nici o schimbare.

Funcționalitatea Analiza performanțelor

Scenariu de bază – Analiza performanțelor:

1. Utilizatorul alege comanda "Performance analyze" de pe meniul principal al aplicației. Sunt disponibile doua forme de analiză (Steady şi Transient). În dependență de matoda selectată sistemul afişează forma necesară pentru setarea paramentrilor.
2. Utilizatorul efectueaza setările necesare.
3. Utilizatorul startează analiza.
4. Sistemul afişează rezultatele obținute în urma analizei.
5. Utlizatorul închide forma de dirijare a simulării.
6. Aplicația revine la forma inițială.

Scenariu alternativ – Analiza performanțelor Anulare:

a.2. Utlizatorul închide forma de dirijare a analizei performanțelor.

Funcționalitatea *Crearea / Editarea Rețelei* este una compusă. Ea conține următoarele subfuncționalități:

-*Add Shape* - adăugă elemete ce compun rețeaua petri;

-*Copy subnet* - copie elementele selectate a rețelei/subrețelei;

-*Cut subnet* - copie elementele selectate a rețelei în memorie şi şterge aceste elemente din rețeaua activă;

-*Paste subnet* - înserează elementele copiate în rețeaua activă;

-*Remove subnet* - şterge elementele selectate ale rețelei.

-*Select elemente* – selectează elementele rețelei/ selectează rețeaua.

Diagrama de activități reprezintă o modalitate de modelare vizuală a fluxurilor. Cu ajutorul diagramelor de activități vom modela cazurile de utilizare prezentate anterior.

Pentru a modela aspectele dinamice la nivel de sistem vom pune accentul pe activități aşa cum sunt vazute ele de actorii care comunică cu sistemul.

Accentuînd în diagramă informația privind responsabilitatea executării acțiunilor se vor folosi elemente swimlanes, plasîndu-se fiecare acțiune pe "culoarul" actorului care execută acea acțiune.

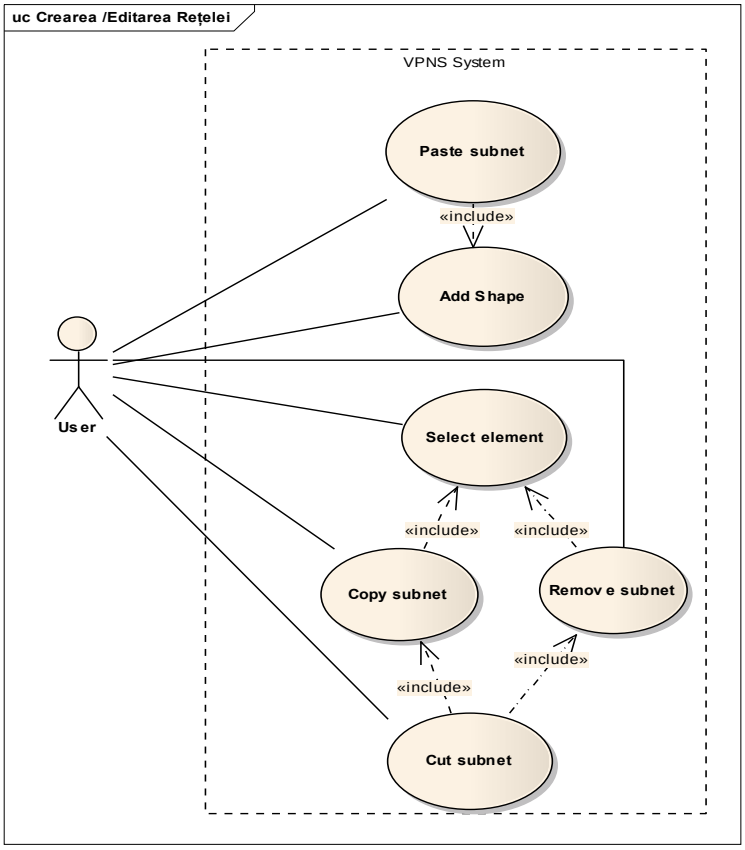

Figura 2 Diagrama Use Case ce reprezintă funcționalitatea *Crearea / Editarea Rețelei*

Userul este cel care inițiază procesul prin intermediul punctului de start al diagramei, după care au loc activitățile de bază. Punctele de decizie sunt specificate printr-un romb cu un flux de intrare şi mai multe fluxuri de ieşire. Fluxurile de ieşire includ condiții care verifica evenimente (listenere) ale obiectelor GUI. Starea finală este atinsă la alegerea ramurii [exit], [close] sau dacă se dorește ieşirea din program.

Bibliografie

1. Alain Abran , James W. Moore, *Guide to the Software Engineering Body of Knowledge*, 2004 Version.
2. Fowler, M. *UML Distilled*, *A Brief Guide to the Standard Object Modeling Languag*e, Addison-Wesley(2000).
3. Gheorghieş O., Apetrei A. *Ingineria Programării*, Universitatea "Al. I. Cuza", Iaşi, 2003.