UNIVERSITATEA „DUNĂREA DE JOS" DIN GALAȚI

CONTRIBUȚII LA ACHIZIȚIA ŞI STRUCTURAREA CUNOŞTINȚELOR ÎN SISTEME INTELIGENTE PENTRU DIAGNOZA DEFECTELOR.

Teză de doctorat

Rezumat

Conducător ştiinţific,
Prof.dr.ing. Severin BUMBARU

Doctorand,
Florin POSTOLACHE

~ 2011 ~

II 39.853

ROMÂNIA
MINISTERUL EDUCAȚIEI, CERCETĂRII, TINERETULUI ȘI SPORTULUI
UNIVERSITATEA DUNĂREA DE JOS DIN GALAȚI

Strada Domnească nr. 47, cod poștal 800008
Galați, România
E-mail: rectorat@ugal.ro

UNIVERSITAS
GALATIENSIS

Tel.: (+4) 0336-130.109; 0336-130.108; 336-130.104
Fax: (+4) 0236 - 461.353
www.ugal.ro

C4963/3.11.2011

Către

_____

_____

Universitatea " Dunărea de Jos " din Galați vă face cunoscut că în data de 21.09.2011 ora 12.00, în sala SENATULUI, va avea loc susținerea publică a tezei de doctorat intitulată : ***" Contributii la achizitia și structurarea cunoștințelor în sisteme inteligente pentru diagnoza defectelor "***, elaborată de domnul/doamna ***ing. POSTOLACHE FLORIN,*** în vederea conferirii titlului științific de doctor în Domeniul de doctorat - ***Știinta calculatoarelor.***

Comisia de doctorat are urmatoarea componență :

Președinte:	***Prof.dr.ing. Dorel AIORDĂCHIOAIE*** ***Decan – Facultatea de Automatică, Calculatoare, Inginerie Electrică și Electronică*** ***Universitatea " Dunărea de Jos " din Galați***
Conducător de doctorat:	***Prof.dr.ing. Severin BUMBARU*** ***Universitatea " Dunărea de Jos " din Galați***
Referent 1:	***Prof.dr.ing. Mihaela OPREA*** ***Universitatea Petrol-Gaze din Ploiești***
Referent 2:	***Prof.dr.ing. Viorel ARITON*** ***Universitatea DANUBIUS din Galați***
Referent 3:	***Prof.dr.ing. Cristina SEGAL*** ***Universitatea " Dunărea de Jos " din Galați***

Cu această ocazie vă transmitem rezumatul tezei de doctorat și vă invităm să participați la susținerea publică. În cazul în care doriți să faceți eventuale aprecieri sau observații asupra conținutului lucrării, vă rugăm să le transmiteți în scris pe adresa Universității, str. Domnească nr. 47, 800008 - Galați, Fax - 0236 / 461353.

RECTOR,
Prof.dr.ing. Viorel MÎNZU

SECRETAR DOCTORAT,
Ing. Luiza AXINTE

„Trebuie să încerci necontenit să urci foarte sus daca vrei să vezi foarte departe"

Constantin Brâncuși

Familiei mele și celor care au fost alături de mine,

Prof.dr.ing. Severin Bumbaru,

Conf.dr. Cătălin Angelo Ioan,

Prof.dr. Dan Caragea,

Colectivul Catedrei.

Cuprins

1. INTRODUCERE

1.1. Domeniul de investigaţie şi actualitatea temei

Virtualizarea mediilor fizice (hardware), subiect din ce în ce mai dezbătut, acceptat şi implementat în prezent, în principal datorită creşterii disponibilităţii şi gradului de utilizare a resurselor hardware disponibile, devine o necesitate absolută atunci când planificăm o reconfigurare a infrastructurii IT şi vizăm totodată o alocare dinamică şi echilibrare a încărcării resurselor fizice disponibile, fără a avea o creştere proporţională a costurilor, ca în cazul unei extinderi obişnuite a infrastructurii. Deşi verificabilă şi tratată de cele mai multe ori doar ca pe o extensie a infrastructurii deja existente, virtualizarea comportă multiple discuţii de cele mai multe ori contradictorii, deoarece anumiţi specialişti manifestă o reticenţă nejustificată referitor la implementarea ei.

Datorită comportamentelor distincte ale entităţilor eterogene din componenţa unui sistem, sistemele virtuale se caracterizează prin prezenţa unor interacţiuni puternice, de regulă neliniare, între entităţi, prin reacţii ciclice complexe, care fac dificilă separarea cauzei de efect, prin discontinuităţi (întârzieri) semnificative în spaţiu şi timp, precum şi printr-o serie de praguri şi limitări I/O. Toate acestea îngreunează munca cercetătorilor şi din această cauză a apărut opţiunea noastră pentru o cercetare legată de necesitatea virtualizării infrastructurii IT şi administrării eficiente a mediilor virtualizate astfel încât să avem o funcţionare a sistemului în parametri optimi.

Ipoteza de la care ne propunem să pornim cercetarea noastră este aceea conform căreia în mediul abstractizat, comparativ cu cel fizic, achiziţia şi structurarea cunoştinţelor pentru diagnoza defectelor dobândeşte noi valenţe, datorită faptului că infrastructura hardware devine un serviciu, maşinile virtuale instalate pe cele hardware devin preponderente în cadrul infrastructurii iar agentul (inteligent sau uman) si experienţele anterioare joacă un rol major în soluţionarea cazurilor.

Necesitatea cercetării este impusă şi de evoluţia rapidă a tehnologiei informaţiei, care conduce la inaugurarea de noi metode de organizare a infrastructurii hardware, a aplicaţiilor, precum şi la noi modalităţi de efectuare a operaţiilor, indiferent de complexitatea lor. Datorită virtualizării, procesele capătă un caracter dinamic şi distribuit, structurile statice şi ierarhice devin din ce în ce mai adaptabile şi mai flexibile iar software-ul sporeşte semnificaţia interacţiunilor dintre om şi computer. Astfel, pentru diagnoza defectelor, se impune, pe de o parte, cunoaşterea manifestărilor sistemelor complexe la defect, iar pe de altă parte, achiziţia cunoştinţelor cu privire la anomalii, manifestări, granularitatea defectelor, relaţiile dintre acestea în varii contexte de funcţionare, toate limitându-se acum la un nou mediu abstractizat. Cercetarea întreprinsă în vederea achiziţiei şi structurării cunoştinţelor, nu intenţionează rezolvarea completă a problematicii diagnozei, chiar dacă, deţinerea acestor cunoştinţe pot soluţiona problema pentru o anumită tehnică de diagnoză.

Prezenta lucrare se adresează comunităţii specialiştilor din domeniul diagnozei şi din domeniul sistemelor virtualizate, cărora ne propunem să le oferim o posibilă soluţie privind achiziţia şi structurarea cunoştinţelor prin elaborarea de noi metode prin care incidentele din interiorul infrastructurii să fie gestionate rapid. Soluţia, testată si validată, urmează a fi oferită comunităţii, în condiţiile în care aceste demersuri au reprezentat mereu o sarcină dificilă, în raport cu cvasi-imposibilitatea de a beneficia de o opinie majoritară.

1.2. Obiectivele cercetării

Prezenta lucrare își propune ca, pe fundamentul unei elaborări teoretice, să pună bazele unei metodologii care are drept scop funcționarea în parametri optimi ai unui sistem, indiferent de problemele care apar pe durata funcționării lui. În domeniul diagnozei defectelor urmărim elaborarea unui sistem de asistare a diagnosticianului uman (SADU) care să ofere soluții pentru administrarea și gestionarea sistemelor în vederea creșterii eficienței, siguranței în funcționare și exploatare, având totodată capacitatea de a semnaliză și eventual interveni în cazul în care anumite echipamentele IT nu funcționează conform specificațiilor.

Virtualizarea infrastructurii implică noi provocări în ceea ce privește alocarea dinamică și echilibrarea încărcării resurselor hardware disponibile din infrastructură, de aceea, vizând cel mai important aspect al măsurii funcționalității unui sistem, care este dat de către utilizatorii finali precum și de gradul lor de satisfacere ca urmare a funcționării serviciilor, achiziția și structurarea cunoștințelor pentru diagnoza defectelor implică o nouă modalitate de abordare, pe care ne-o asumăm în prezenta lucrare.

Aceasta vizează, înainte de toate, modalitatea de prezentare a proceselor, procedurilor și etapelor ce au loc pe tot parcursul unui proiect de achiziție de cunoștințe, fiind necesară totodată o bună cunoaștere a sistemului, cu toate particularitățile sale. În viziunea noastră, pentru diagnoza defectelor mediilor abstractizate este necesară cunoașterea și achiziția cunoștințelor referitoare la comportarea la defect (anomalii/simtoame și manifestări, granularitatea defectelor, relații între entități în contexte de funcționare diferite, etc.).

Propunem o nouă viziune asupra modelului de structurare a infrastructurii abstractizate prin prisma grupării blocurilor/straturilor ce deservesc componentele hardware (servere și spațiu de stocare), componentele hardware abstractizate (serverele abstractizate și spațiul de stocare abstractizat) și rețelele care asigură transferul de informație între acestea (rețelele spațiului de stocare și rețeaua din DataCenter) în jurul conceptuli de stivă, lucru care permite o mai ușoară introducere în cunoașterea și înțelegerea domeniului virtualizării, a componentelor sale și a relațiilor dintre acestea. Astfel devine indispensabilă elaborarea unei ontologii a domeniului vizat, pe care o considerăm necesară și oportună la momentul actual pentru comunitatea specialiștilor. Alegerea metodologiei corespunzătoare privind clasificarea sistemelor virtualizate în vederea definirii unei noi ontologii se va face prin prisma straturilor ce compun modelul propus pentru structurarea infrastructurii IT, permițând captarea relațiilor dintre entitățile care intră în componența unui strat cât și relațiile dintre componentele diferitelor straturi.

Diagnoza defectelor în sistemele IT abstractizate conduce la noi provocări pentru inginerii de sistem în înțelegerea și depanarea posibilelor probleme ce apar, mai ales dacă acestea implică anumite modificări ale parametrilor funcționali. În consecință, vom proceda la o depistare a manifestărilor, observabile pe fiecare nivel al sistemului țintă, în mod pasiv (fișiere de log, înregistrări, etc.) sau activ (agenți inteligenți), cu scopul localizării și remedierii defectului conform ipotezelor și investigațiilor suplimentare. Schimbarea modelului de la o orientare centrată pe defect la una bazată pe urme pentru a facilita sau amplifica propriile capacități , atât ale experților umani cât și ale utilizatorilor constituie punctul nostru de plecare în modelarea interacțiunilor dintre resursele fizice și cele virtuale. Vom putea astfel să punem bazele unui sistem instruibil care să sprijine deciziile diagnosticianului uman în rezolvarea problemelor apărute.

Reprezentarea stării funcționale a unui sistem, în contextul încărcării procentuale pe care o au resursele abstractizate prin prisma disponibilității lor, precum și modalitatea de alocare dinamică și echilibrare a încărcării resurselor vor fi tratate utilizând un model matematic și logica fuzzy.

Pentru validarea investigării pe care ne-o asumăm, am ales două modalități de reprezentare a stării funcționale, tratate în secțiunile studiului de caz.

1.3. Structura lucrării

Având în vedere obiectivele formulate și metodologia asumată, am considerat că lucrarea ar trebui să aibă o structură organizată pe capitole, care se configurează ca un demers de la teoretic către practic, prin realizarea de analize comparative, de aplicații, de validări, de interpretări și evaluări.

Capitolul *Abordări în diagnoza sistemelor complexe* tratează sistemele IT abstractizate. Pentru a înțelege mai bine complexitatea sistemului, vom realiza o clarificare conceptuală a sistemelor abstractizate (ce sunt și cum se virtualizează aceste sisteme). Propunem o nouă metodologie privind abstractizarea infrastructurii IT scoțând în evidență avantajele abordării. Totodată, vor fi vizate beneficiile și problemele întâlnite pe tot parcursul procesului de virtualizare, monitorizare și funcționare în timp a sistemelor abstractizate.

Capitolul *Achiziția și structurarea cunoștințelor* tratează noțiunile de bază și stadiul actual al cercetării privind achiziția și structurarea cunoștințelor pentru sistemele IT virtualizate. Ne propunem să analizăm metodele utilizate, problemele privind reprezentarea și stocarea cunoștințelor precum și formalismul de reprezentare pentru achiziția de cunoștințe cu scopul de a popula o bază de cunoștințe. De asemenea, ne vom opri asupra sistemelor bazate pe cunoștințe, reliefând ce reprezintă ele, care sunt caracteristicile lor, modalitățile de funcționare, procesul prin care este dezvoltat un sistem bazat pe cunoștințe și modul lor de utilizare. Unul din scopurile noastre este să identificăm instrumentele și tehnologiile disponibile, dar și care sunt problemele ce pot apărea din punct de vedere al costurilor și al impactului pe care-l au aceste sistemele bazate pe cunoștințe. O atenție deosebită o vom acorda utilizării și beneficiilor, dar și problemelor și dificultăților apărute la achiziția și captarea de cunoștințe. De asemenea, în acest capitol tratăm modalitățile de achiziție de cunoștințe cu ajutorul agenților (inteligenți sau umani) cât și structurarea și reprezentarea cunoștințelor, punând accentul asupra procedurii și modului de lucru dar și asupra aspectelor esențiale privind captarea, analiza și modelarea lor. Propunem o nouă ontologie vizând domeniul virtualizării infrastructurii IT pe care o supunem spre validare comunității.

Capitolul *Diagnoza defectelor* abordează identificarea naturii sau cauzei unui fenomen în sistemele complexe virtualizate. Pornind de la conceptul de diagnoza, prin descrierea tehnicilor de localizare a defectului cele mai utilizate, ne vom opri asupra modalității de aplicabilitate în sistemele complexe virtualizate,a raționamentului bazat pe urme, pornind de la cel bazat pe cazuri (CBR) detaliind aplicația (sistemul instruibil care sprijină deciziile diagnosticianului uman în rezolvarea problemelor apărute) și rezultatele obținute.

În capitolul *Abordări aproximative privind alocarea și încărcarea resurselor pentru SADU* realizăm o descriere a metodelor științifice de cercetare unde propunem, pe lângă o modalitate de atenționare și notificare a stării funcționale, o modalitate de alocare dinamică și echilibrare a încărcării resurselor disponibile în cazul în care sistemul intră în colaps. În cadrul acestui capitol supunem validării studiul nostru de caz, folosindu-ne de mai multe modalități de apreciere a stării funcționale a sistemului privind echilibrarea încărcării, printr-o descriere în detaliu a aplicațiilor și a rezultatelor obținute.

Lucrarea se încheie prin formularea concluziilor, contribuțiilor și a viitoarelor direcții de cercetare.

2. ABORDĂRI ÎN DIAGNOZA SISTEMELOR COMPLEXE

Recent, evoluția tehnologiei informațiilor a inaugurat noi metode de structurare a infrastructurii hardware, a aplicațiilor și, implicit, a modului de efectuare a operațiilor. Astfel, în ultima perioadă, infrastructura unor organizații a devenit parțial sau complet virtualizată, iar datorită acestui fapt, procesele au căpătat un caracter dinamic și distribuit, iar structurile statice și ierarhice au devenit din ce în ce mai adaptabile și flexibile.

Un **sistem complex** (secțiunea 2.1) este un sistem compus din multiple și variate entități independente interconectate, aflate în interacțiune prin schimburi de energie, materie și informație (Rastetter et al. 1992; von Bertalanffy 1968), care se manifesta ca un întreg, prezentând una sau mai multe proprietăți care nu sunt în mod cert proprietăți ale părților individuale ce compun sistemul.

Sistemele inteligente, într-un nou mod de abordare științifică, studiază cum relațiile dintre entități dau naștere la noi comportamente colective ale sistemului și la cum ele interacționează și formează legături cu mediul înconjurător astfel încât să-și poată atinge obiectivele vizate. Astfel, acestea învață pe tot parcursul existenței lor și acționează pentru fiecare situație apărută „simțind" mediul înconjurător datorită interacțiunii și comunicării cu acesta. Totodată, sistemele moderne conduc la provocări pentru inginerii de sistem în a înțelege și depana posibilele probleme ce apar în sistem, mai ales dacă aceste sisteme implică anumite modificări ale parametrilor funcționali.

Virtualizarea și norul informatic (secțiunea 2.2) reprezintă subiecte intens dezbătute în ultima perioadă care conduc către strategii numeroase, destul de diferite. Virtualizarea implică transformarea unui număr de componente hardware și dispozitive de rețea în software și încărcarea lui pe o platformă hardware puternică, cu alte cuvinte, serviciile oferite de o componenta hardware virtualizată sunt abstractizate din componenta fizică.

Virtualizarea se aplică serverelor, mediilor de stocare, rețelei și *desktop*-urilor astfel că, mare parte din discuțiile privind virtualizarea si norul (*cloud*) evidențiază accesul nelimitat la resursele de procesare, stocare și în rețea la lățimea de bandă. Dintre modalitățile diferite de virtualizare (Figura 2.2), virtualizarea serverelor este cea mai familiară, de aceea, de cele mai multe ori termenul de „virtualizare", face referire de obicei la virtualizarea serverelor. Astfel, pe un mediu „bare metal"[1] cu ajutorul unui soft de virtualizare[2], se pot crea mai multe mașini virtuale, distincte ca entități. Concret, virtualizarea și norul informatic nu pot exista fără un *DataCenter* abstractizat.

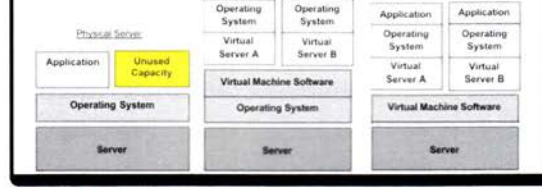

Figura 2.2. Tipuri de virtualizare a serverului (Postolache F. et al., 2010)

[1] Un mediu „bare metal" este un sistem de calcul sau o rețea unde o mașină virtuală sau o rețea virtuală este instalată direct pe hardware.
[2] Software-ul utilizat în studiul nostru de caz a fost oferit de VMware prin licență academică.

Virtualizarea rețelei, ca și alte forme de virtualizare, presupune mai multe servicii de rutare și de switch-ing, iar virtualizarea spațiului de stocare (SV[3]) implică „abstractizarea la orice nivel în software-ul de stocare și stiva hardware"[4]., nu doar posesia unui SAN[5]

În abordarea noastră ne vom opri în mod special asupra modelului infrastructurii abstractizate și implicit la consolidarea serverelor din *DataCenter B*, asupra mașinilor virtuale și a aplicațiilor instalate pe acestea.

Într-un proiect de virtualizare, blocurile hardware vizate sunt: spațiul de stocare, servere fizice și rețelele care le deservesc. Pentru a beneficia pe deplin de avantajele virtualizării este necesar să dispunem de o infrastructură care întradevăr să poată suporta abstractizarea fiecărei componente fizice din cadrul infrastructurii IT și implicit a centrului de date (Figura 2.7).

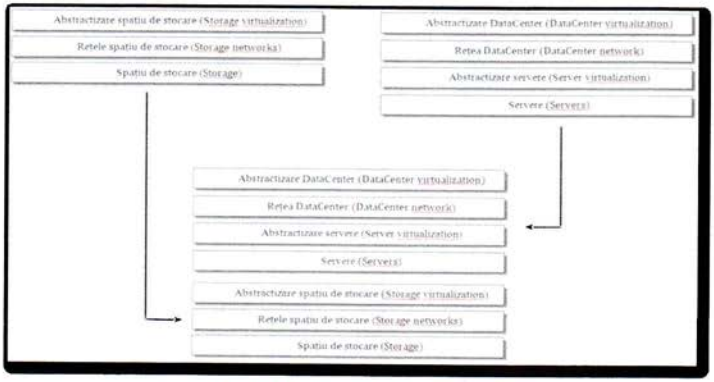

Figura 2.7. Abstractizare blocuri hardware

Astfel, în mod tradițional, o infrastructura IT virtualizată (Figura 2.8) ce deservește întreg spectrul de utilizatori constă din straturi/blocuri care abstractizează componentele echipamentelor hardware utilizate. Administrarea resurselor disponibile ce intră în componența straturilor/blocurilor, fie că sunt hardware sau software, se face cu ajutorul unor interfețe separate care permit cu ușurință gestionarea, monitorizarea și configurarea resurselor disponibile.

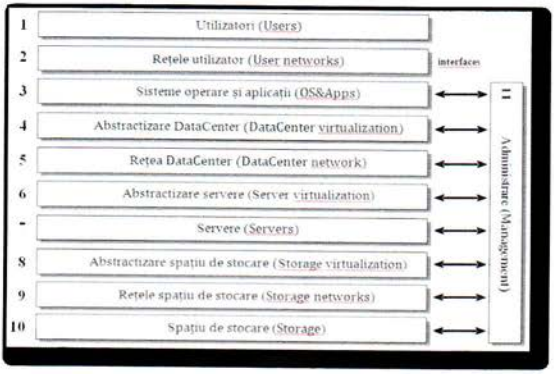

Figura 2.8. Abstractizare infrastructură IT (VMware)

[3] SV – engl. *Storage Virtualization*
[4] http://en.wikipedia.org/wiki/Storage_virtualization
[5] SAN – engl. *Storage Area Network*

2.3. Abordarea propusă

În cadrul studiului întreprins, propunem o nouă viziune asupra modelului tradițional descris, bazată pe experiența din depanarea/utilizarea sistemelor virtuale, concretizată printr-o grupare a straturilor (Figura 2.9) în jurul conceptului de stivă. Acest lucru ne va ajuta ulterior la stabilirea competențelor agenților plasați în sistem în vederea desfășurării proiectului de achiziție de cunoștințe și în al doilea rând pentru o mai bună diagnoză a defectelor.

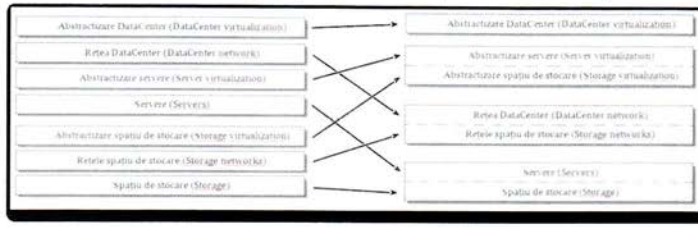

Figura 2.9. Grupare blocuri

Din rațiuni pur economice și practice, pentru a eficientiza achiziția de cunoștințe cu ajutorul agenților, propunem astfel spre analiză, dezbatere, testare și validare o nouă abordare privind abstractizarea infrastructurii IT fizice, materializată printr-o grupare a straturilor ce deservesc componentele hardware (servere și spațiu de stocare), componentele hardware abstractizate (serverele abstractizate și spațiul de stocare abstractizat) și a rețelelor care asigură transferul de informație între acestea (rețelele spațiului de stocare și rețeaua din DataCenter) în jurul conceptului de stivă (Figura 2.10).

Această nouă abordare privind abstractizarea infrastructurii IT, pe lângă faptul că nu înlătură nici un bloc din componența modelului standard, a permis gruparea anumitor blocuri fără modificarea interfețelor de administrare prevăzute pentru gestionarea, monitorizarea și configurarea serviciilor, asigurând totodată o funcționare în parametri optimi, conform Anexa A.

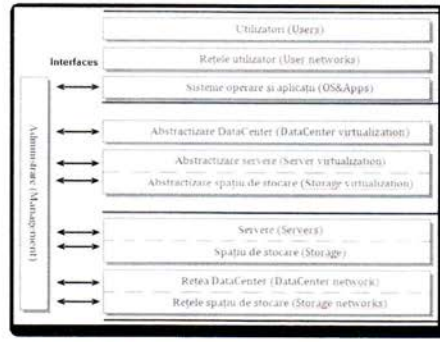

Figura 2.10. Abstractizare infrastructură IT propusă

2.3.1. Implementarea abordării propuse

S-a procedat la virtualizarea rețelei, a serverelor, a spațiului de stocare și a unui laborator de informatică compus din 32 de calculatoare, formând astfel un *cloud* - nor informatic -, structurat (Figura 2.11). În abordarea propusă, interfețele de administrare ce deservesc blocurile rămân independente, lucru indicat prin săgețile stratului

de administrare. Modelul propus a necesitat în schimb îndepărtarea stratului de separație dintre *SAN*-uri, lucru care a presupus îmbinarea într-o singură rețea a celor două rețele fizice independente.

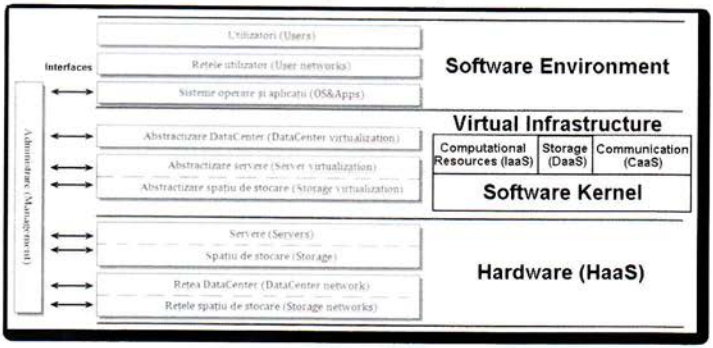

Figura 2.11. Model implementat

Astfel, pe infrastructura supusă testării, norul informatic și virtualizarea serverului se completează reciproc datorită următoarelor considerente:

- Ambele necesită o infrastructură fizică robustă, bazându-se puternic pe rețea și implicând o regândire a infrastructurii tradiționale.
- Serviciile norului informatic sunt implementate după ce *DataCenter*-ul este virtualizat, deoarece doar stratul de virtualizare suportă arhitectura Cloud.
- Norul informatic adaugă un nou strat de virtualizare între utilizatorul final și întregul mediu informatic abstractizat.

În concluzie, abordarea propusă prin prisma grupării straturilor funcție de apartenența/tipul lor (hardware, infrastructură virtuală, software) a condus la o delimitare a responsabilității agenților inteligenți în vederea achiziției de cunoștințe pentru diagnoza defectelor și a permis o mai bună înțelegere a domeniului abstractizării, concretizată prin dezvoltarea unei noi ontologii.

2.3.2. Validarea abordării

Implementată inițial în mai 2008, abordarea propusă (secțiunea 2.3) a schimbat fundamental arhitectura sistemului, în special la nivel de LAN și SAN, concretizată printr-o mai ușoară monitorizate a încărcării resurselor și traficului în rețea, o creștere a securității și disponibilității resurselor, precum și printr-o diminuare a costurilor din punct de vedere economic (Anexa A).

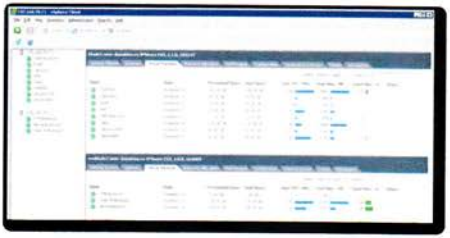

Figura 2.14. Consolidarea serverelor în DataCenter B

Figura A.9. Încarcare rețea Blade 1

Datorită grupării straturilor conform noii abordări, virtualizarea spațiului de stocare, a rețelei cât și abstractizarea și consolidarea serverelor a condus la simplificarea operațiilor de configurare, monitorizare și

administrare asigurând o funcționare în parametri normali a sistemului. Virtualizarea rețelei a constat din plasarea „pe rețea" a serviciilor tradiționale bazate pe client - server. În urma monitorizării traficului s-a constatat că funcționarea rețelei se încadrează în parametrii optimi, chiar dacă aveam tot traficul pe „un singur cablu", iar administrarea echipamentelor a devenit mai convenabilă datorită faptului că acestea aparțin unei singure rețele (Anexa A). Componentele hardware ce deserveau mediile de stocare au fost plasate într-un fond comun de resurse, și funcție de cerere, acestea pot fi alocate în mod dinamic, fără o întrerupere sau dereglare în modul de funcționare a sistemului. Consolidarea serverelor s-a concretizat cu ajutorul produselor software oferite de VMware, procedându-se la o virtualizare a mașinilor fizice existente. Infrastructura virtuală creată ne-a permis nu numai o menținere a numărului fizic de servere ci și creșterea numărului de servere virtuale, concomitent cu creșterea gradului de utilizare al resurselor hardware disponibile. În plus, soluția ne-a oferit posibilitatea izolării încărcării severelor și controlul tuturor resurselor sistemului.

Cu alte cuvinte, infrastructura hardware existentă, datorită virtualizării, a suferit modificări care ne-au asigurat o mai bună performanță în utilizarea resurselor disponibile și a permis fără investiții suplimentare o dezvoltare ulterioară, concretizată prin creșterea numărului de mașini virtuale care deservesc cerințe tot mai mari, cu costuri relativ mici. Tocmai acest lucru a demonstrat că în condițiile existenței pe o singură mașină fizică a 8 mașini virtuale, gradul de încărcare a resurselor este totuși destul de mic (Anexa A), iar per ansamblu a implicat un grad crescut de utilizare a resurselor fizice existente și implicit, din punct de vedere financiar, o amortizare mai eficientă a cheltuielilor. Totuși, pe parcurs, abordarea propusă a confirmat avantajele obținute evidențiind în schimb și posibile probleme, pe care le-am eliminat odată cu asigurarea redundanței la nivelul blocurilor (secțiunea 2.3.3).

2.4. Concluzii și contribuții

Datorită conjuncturii economice actuale dar și problemelor impuse de securitatea sistemelor informatice, companiile au început să utilizeze tehnologii de virtualizare pentru a-și proteja cât mai bine cele mai valoroase active: datele stocate. Virtualizarea schimbă modul în care un centru de date funcționează, este gestionat și administrat. De exemplu, înainte de a implementa virtualizarea, se știa în orice moment care sunt aplicațiile cele mai utilizate, pe ce mașini fizice sunt instalate, cât din resursele mașinii se folosesc, care sunt momentele de încărcare maximă, etc. Datorită virtualizării, aceasta legătură tradițională dintre hardware și software este ruptă, iar acest lucru creează o mai bună perspectivă privind funcționalitatea și rezolvarea conflictelor. Totodată, datorită profilului ciclic de funcționare a aplicațiilor, administratorii au o privire de ansamblu globală asupra sistemului, cunoscând modul de funcționare a acestuia vizualizând care sunt aplicațiile și momentele ce încetinesc performanța sistemului. Punctul forte îl constituie administrarea întregului sistem abstractizat folosind o singură consolă.

Contribuția majoră este legată de modalitatea grupării straturilor din componența infrastructurii abstractizate în jurul conceptului de stivă funcție de apartenența sau tipul acestora (hardware, infrastructură virtuală, software), abordare care a condus la o funcționare în parametri optimi a infrastructurii abstractizate, fără o investiție suplimentară în echipamente hardware, la minimizarea timpilor de nefuncționare a echipamentelor precum și simplificarea și raționalizarea proceselor de administrare, asigurând în final o mai bună înțelegere a domeniului virtualizării. Din punct de vedere economic, implementarea a contribuit la micșorarea costurilor implicate de investițiile în hardware, software și de acordare a licențelor precum și reducerea facturilor la utilități.

Această abordare ne va ajuta în derularea proiectului de achiziție și structurare a cunoștințelor conducând în final la dezvoltarea unei noi ontologii specifice domeniului.

3. ACHIZIȚIA ȘI STRUCTURAREA CUNOȘTINȚELOR

Achiziția de cunoștințe este o etapă necesară în realizarea sistemului de asistare a diagnosticianului uman. Modalitățile de achiziție de cunoștințe pentru diagnoza defectelor în cazul sistemelor virtualizate, în vederea populării bazei de cunoștințe, se rezumă atât la metoda clasică de achiziție (de la expertul uman), cât și la metoda automată de achiziție a cunoștințelor folosind agenții inteligenți (secțiunea 3.3.2), arătând cum, aceștia din urmă, pot acționa și interacționa cu sistemul abstractizat. Metoda manuală implică interviul, colectarea și sesiunea de extragere a cunoștințelor cu ajutorul experților în domeniu, iar metoda bazată pe calculator încearcă să automatizeze procesul achiziției de cunoștințe, utilizând tehnici interactive (semi-automate) sau tehnici bazate pe învățare (automate).

Ne propunem crearea unei baze de cunoștințe, a instrumentelor necesare achiziției de cunoștințe, elemente ce vor fi puse la dispoziția experților, urmărind astfel unificarea, actualizarea și validarea lor. Intenția este de a demonstra că tot ce afirmăm se poate realiza prin tehnici de achiziție de cunoștințe ușor de utilizat, dovedind astfel performanța procedurii pe care o propunem și care să conducă în final la o pertinentă diagnoză a defectelor. Scopul nostru este de a înainta experților spre dezbatere și validare întreaga cazuistică, care, după cum se știe, astăzi nu este unificată, nu este făcută publică, constituind doar un fond de cunoștințe umane, tacite sau care lipsesc cu desăvârșire. Modalitatea prin care cunoștințele sunt stocate în baza de cunoștințe precum și legăturile dintre cele patru componente ale ei :concepte, atribute, valori și relații este dată de tehnologiile de cunoștințe și de sistemele bazate pe cunoștințe (Knowledge Based System - KBS) utilizate (secțiunea 3.2).

Pentru structurarea și reprezentarea cunoștințelor considerăm necesară dezvoltarea unei ontologii proprii (secțiunea 3.6.1) deoarece domeniul de investigație este relativ nou și din acest considerent optăm pentru adoptarea metodologiei (secțiunea 2.3) care permite captarea relațiilor dintre entitățile/serviciile ce intră în componența unui strat cât și a relațiilor dintre componentele diferitelor straturi.

Construcția unui sistem bazat pe cunoștințe (secțiunea 3.2.1) reprezintă o etapă necesară în realizarea sistemului de asistare a diagnosticianului uman (secțiunea 4.6). Sistemul bazat pe cunoștințe (KBS - *Knowledge Based System*), reprezintă un sistem care utilizează metode și tehnici de inteligență artificială (AI) pentru a rezolva probleme complexe care în mod normal ar fi efectuate de către o persoană cu experiență în domeniu. Sunt descrise caracteristicile generale, modalitatea de funcționare și modelele de rezolvare a problemei de către un KBS (secțiunea 3.2). Insistăm asupra procedurilor achiziției de cunoștințe și a manierei de lucru folosite, prin aprofundarea celor trei aspecte esențiale:

- captarea cunoștințelor (tehnici utilizate de către experți, agenți inteligenți);
- analiza și structurarea cunoștințelor (identificare tuturor elementelor necesare pentru construirea bazei de cunoștințe, ontologiei);
- modelarea cunoștințelor (crearea diferitelor metode de editare și vizualizare a bazei de cunoștințe).

În concluzie, cele trei aspecte importante ale unui proiect de achiziție de cunoștințe pentru diagnoza defectelor: captarea, analiza și modelarea cunoștințelor se referă la modelul normativ, (de funcționare normală – cel puțin în cazul diagnozei bazată pe model), la modelul de comportare la defect și la corespondența „context de funcționare"- „manifestări" – „defect", pentru a obține un produs final coerent având ca scop utilitatea și ușurința folosirii lui de către specialiști.

3.2.1. Dezvoltarea KBS. Instrumentele și tehnologii disponibile

Etapele dezvoltării sistemului bazat pe cunoștințe (KBS) din cadrul proiectului sunt:

1. Dobândirea cunoștințelor:
   a. Manual, în care un inginer de cunoștințe captează și modelează cunoștințele de la un expert în domeniu pentru a construi o bază de cunoștințe. Această etapă include, de asemenea, fazele: (i) de definire ale sistemului și (ii) de precizare a cunoștințelor care vor fi captate de către experții umani (denumit domeniul de competență),
   b. Cu ajutorul agenților inteligenți.
2. Etapa de implementare: programatorul ia informațiile și baza de cunoștințe și le transformă într-un program funcțional. Aceasta etapa include, de asemenea, etapele de testare și corectare ale erorilor (rafinament).

În studiul nostru de caz, propunem o abordare realizabilă în trei faze de lucru, astfel:

Faza 1: Domeniul de investigație, modelul sarcină și modelul agent.

Această fază clarifică aspectele legate de necesitatea unui sistem bazat pe cunoștințe, căror tip de probleme le este adresată soluția, avantajele, costurile și efectele privind domeniul de investigație vizat. De asemenea, înțelegerea pe deplin a contextului și mediului în care sistemul va fi dezvoltat și pus în aplicare se face analizând în detaliu: (i) domeniul vizat, (ii) sarcinile relevante și (iii) agenții (inteligenți sau experții din domeniu). Astfel:

- **Domeniul de investigație** include cunoștințele orientate spre probleme și oportunități, o defalcare a procesului și a cunoștințelor acumulate implicate pentru crearea ontologiei;
- **Modelul sarcină** scoate în evidența sarcinile relevante și caracteristicile lor;
- **Modelul agent** include agenții, sarcinile pe care le efectuează, cunoștințele experților pentru acele sarcini (inclusiv cunoașterea blocajelor/ gâtuirilor), precum și o evaluare a modului și impactului în care sistemul bazat pe cunoștințe produce schimbări în rolul utilizatorilor.

Faza 2: Modelul de cunoștințe și modelul comunicare

Aspectele importante din această fază fac referire la natura și structura cunoștințelor implicate dar și la natura și structura corespunzătoare comunicării. Aceste aspecte sunt clarificate întocmind o descriere conceptuală a cunoștințelor utilizate într-o sarcină, realizată prin construirea modelului de comunicare și modelului de cunoștințe.

Modelul de cunoștințe este o descriere independentă a modului de utilizare a componentelor cunoștințelor implicate în efectuarea unei activități/sarcini. Pentru a ajuta la crearea modelului de cunoștințe, este selectată o sarcină șablon și apoi populată cu domeniul de cunoștințe. Sarcina șablon se bazează pe un model de soluționare a problemelor, care descrie cum un expert constată o greșeală/eroare într-un sistem.

Principalele modele de soluționare a problemelor sunt:

- *Clasificarea* – stabilește corect categoria pentru un obiect;
- *Evaluarea* – găsește o categorie de decizie pentru un caz, bazată pe set de norme specifice domeniului;
- *Monitorizarea* – analizează procesele continuu pentru a afla dacă acestea se comportă conform așteptărilor;
- *Sinteza* – proiectează o structură care îndeplinește integral un set de cerințe;

- *Proiectarea configurației* – găsește un ansamblu de componente care satisface un set de cerințe și se supune tuturor constrângerilor;
- *Atribuirea* – creează un raport între două grupuri de obiecte, care satisfac cerințele și se supun la toate constrângerile;
- *Planificarea* - generează un plan care constă dintr-un set ordonat de activități, care îndeplinesc un obiectiv sau un set de obiective;

Fiecare model de soluționare a problemelor este folosit pentru a conduce la extragerea, analiza și modelarea activităților necesare pentru a construi modelul de cunoștințe. Modelul de comunicare, realizat printr-o implementare conceptuală independentă, descrie interacțiunile între diferiți agenți implicați într-o sarcină.

Faza 3: Proiectarea sistemului

Această fază clarifică aspectele referitoare la modalitatea de implementare a cunoștințelor într-un sistem computațional, arhitectura software și mecanismele de calcul care trebuiesc folosite.

Modelele create în fazele anterioare constituie caietul de sarcini al cerințelor pentru sistemul bazat pe cunoștințe, defalcate pe diferite aspecte. Pe baza acestuia este creat modelul proiectului care ne oferă specificațiile tehnice din punct de vedere al arhitecturii platformei, specificațiile modulelor software precum și mecanismele computaționale constructive și de reprezentare.

3.4. Abordarea asumată

Pe parte experimentală, achiziția și structurarea cunoștințelor pentru diagnoza defectelor în sisteme virtuale se derulează pe infrastructura IT a unei instituții de învățământ. Infrastructura virtualizată este împărțită în trei zone fizice, fiecare corespunzând unei locații distincte. Locațiile sunt integrate printr-un inel de fibră optică redundant, fiecare dispunând de un DataCenter individual. Virtualizarea rețelei fizice s-a făcut pe echipamentele Cisco, iar abstractizarea resurselor hardware ale sistemelor de calcul cu ajutorul software-ului VMware vSphere 4 ne-a oferit oportunitatea transformării componentelor hardware în software și astfel „încărcarea" lor pe platforma hardware existentă. Identic s-a procedat si cu mediul de stocare, printr-o abstractizare la orice nivel în software-ul de stocare și stiva hardware din *Storage Area Network* (SAN).

În urma abstractizării, fiecare mașină hardware existentă virtualizată conține minim trei mașini virtuale eterogene care vor dispune de sistemele de operare și aplicațiile dedicate. Pe parcursul studiului ne vom îndrepta atenția în mod special asupra unei mașini hardware (blade1) virtualizate care suporta pe perioada studiului între 5 și 10 mașini virtuale, pe care sunt instalate sisteme de operare și aplicații mari consumatoare de resurse, amintind aici aplicația Danubius Online[6].

3.4.1. Modul de lucru

Agenții inteligenți care operează în cadrul sistemului supus testării sunt agenți reflex simpli, agenți bazați pe utilitate și agenți statici de învățare. Aceștia sunt plasați pe nivelurile de interes urmărite din cadrul infrastructurii abstractizate: hardware(mașină fizică, rețele de calculatoare, etc.), infrastructură abstractizată (stratul de virtualizare, mașina virtuală) și mediul software (sisteme de operare, aplicații, baze de date, etc.). Tehnologia aflată în spatele

[6] Danubius Online este un LMS integrat pe un sistem virtualizat, accesat de peste 10000 clienți.

agenților inteligenți este o combinație de tehnici din domeniul inteligentei artificiale și de metodologii de dezvoltare a sistemelor, care le va permite sa învețe și să reacționeze la mediul înconjurător. Agenții inteligenți vor interacționa cu mediul din care fac parte prin criterii de selecție a datelor bazate pe reguli. Vom urmări ca pe parcurs agenții inteligenți să își dezvolte reguli corespunzătoare datorită instrucțiunilor explicite furnizate de utilizator, prin imitarea utilizatorului, prin feedback-ul pozitiv sau negativ primit de la utilizator și prin indicațiile obținute în urma interacționării cu alți agenți.

3.4.2. Modul de comunicare

Agenții inteligenți plasați în sistem vor monitoriza și măsura constant activitatea din cadrul sistemului, pe nivelurile corespunzătoare competențelor, raportând (inserând în baza de cunoștințe) și semnalizând problemele de natură hardware sau software ce survin pe parcursul desfășurării proceselor. Urmărim crearea unei interfeței standard pentru toate platformele PC, indiferent de protocolul de comunicare, astfel încât agenții să poată opera nu doar în cadrul unui anumit sistem de operare. În cele mai multe cazuri, tipul interfeței pe care o are un agent inteligent depinde de tipul resurselor sistemului de operare cu care acesta comunică, de aceea, tipurile de agenți inteligenți disponibili comunică doar cu resursele sistemului din mediul software aflat în imediata sa vecinătate. Încercăm să schimbăm acest lucru, astfel încât limitarea impusă de o anumita platforma să dispară

3.4.3. Modelul mașinii hardware abstractizate

În cadrul sistemului supus testării, pentru o mai bună monitorizare și administrare a sistemul virtualizat cât și pentru o mai bună distribuire a sarcinilor și competențelor agenților în vederea achiziției de cunoștințe, considerăm necesară o împărțire pe straturi (*layers*) a mașinii hardware abstractizate. În consecință, aveam stratul fizic (PL), stratul de virtualizare (VL), stratul mașină virtuală (VM), stratul sistem de operare (SO) și stratul aplicații (*Apps*) (Figura 3.4).

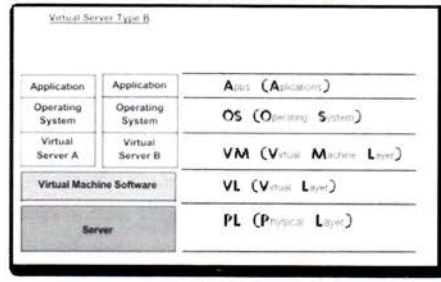

Figura 3.4. Straturile mașinii hardware abstractizate (Postolache et al., 2010)

3.4.4. Plasarea și impactul agenților

În cadrul studiului întreprins ne-am lovit de probleme noi, neîntâlnite până acum, care au implicat o monitorizare atentă a resurselor disponibile utilizate, iar datorită dimensiunii sistemului, structurarea cunoștințelor a constituit o provocare în sistemul complex analizat datorită virtualizării. Achiziția de cunoștințe cu ajutorul agenților inteligenți dar și a celor umani privind comportarea la defect a necesitat o bună cunoaștere a sistemului, cu toate particularitățile sale, pe fiecare nivel propus și supus studiului (hardware, infrastructură virtuală, mediu software, etc.). Abordarea nu se oprește doar la cele menționate mai sus, ci se concretizează și cu modalitatea de acțiune și

interacțiune a agenților cu sistemul virtualizat și aplicațiile instalate pe acesta. Este firesc să se cunoască algoritmul care stă la baza funcționării agentului cu ajutorul căruia se face achiziția, insistând asupra securității și încrederii acordată agentului, în condițiile în care aceștia pot fi considerați viruși.

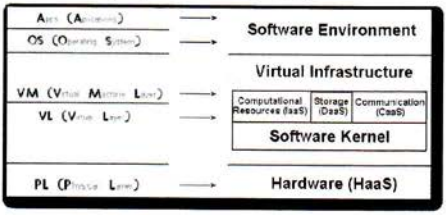

Figura 3.5. Structura sistemelor virtualizate

Considerând necesară o descriere a sistemul informatic pe care are loc testarea, dar și modalitatea abordată, pentru a ne însuși terminologia dar și entitățile prezente în sistemul informatic, sistemul abstractizat este analizat prin prisma straturilor care îl compun (Figura 3.5), pornind de la cel hardware (stratul fizic) la cel de abstractizare a resurselor (infrastructura virtuală), de la alocarea si monitorizarea resurselor abstractizate la funcționalitatea aplicațiilor.

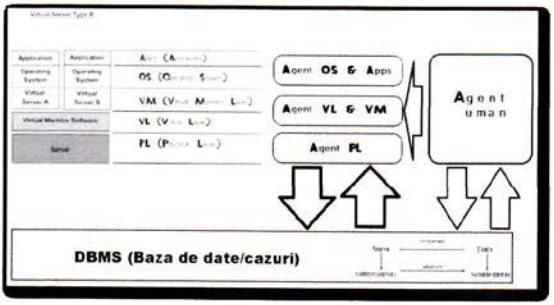

Figura 3.6. Distribuția și rolul agenților în achiziția de cunoștințe

Distribuția agenților inteligenți în cadrul sistemului este conform Figura 3.6, aceștia având rolul de a monitoriza, supraveghea procesele și insera în baza de cazuri evenimentele întâlnite. În urma inserării de către agentul inteligent în baza de cazuri a evenimentelor care conduc la o funcționare care nu se încadrează în parametri normali (defectuoasă) sau chiar conduc la o posibilă defectare a unor componente (nu facem aici referire doar la cele fizice), agentul uman continuă acțiunea, stabilind ce operație sau acțiune trebuie întreprinsă.

Urmare a operațiilor întreprinse de către agentul uman pentru fiecare caz întâlnit, pe baza similarității se caută o posibilă soluție în baza de cazuri, care este adaptată noului caz. Dacă sunt mai multe cazuri identice, se aplică soluția care a produs cele mai bune rezultate. În acest caz, dacă se consideră necesar, responsabilitatea se poate transfera asupra agentului inteligent. Astfel, la apariția aceluiași tip de eveniment, agentul uman nu mai intervine, toată responsabilitatea căzând în sarcina agentului. Acesta inserează o nouă înregistrare în baza de cazuri, corespunzătoare evenimentului, extrage acțiunea care trebuie să o întreprindă și o execută, concomitent cu inserarea acțiuni întreprinse de el pentru completarea înregistrării.

Drept urmare, agentul uman este înștiințat de operațiile efectuate de către agentul inteligent și ca atare poate să constate dacă s-a acționat corect, iar în urma consultării cazurilor identice sau similare poate decide dacă acțiunea viitoare pe care o va lua agentul rămâne ultima decizie considerată ca fiind cea mai bună sau trebuie schimbată.

Inițial agenții inteligenți sunt implicați în observarea funcționalității resurselor fizice și a celor abstractizate, verificarea stării de conectivitate IP a mașinii fizice și a celei virtuale, citirea și afișarea încărcării resurselor dar și consultarea fișierelor de log. Reamintim faptul că în studiul nostru de caz, achiziția de cunoștințe privind structura fizică și logică de componente se va face în funcției de straturile pe care le avem în componența sistemului.

Granularitatea componentelor (nivelul de adâncime a discriminării componentelor) este arbitrară și se alege funcție de nivelul de profunzime vizat al diagnozei astfel:

- la granularitate mare (discriminare la nivel macro) diagnoza vizează blocuri funcționale mari în scopul înlocuirii – restartării componentelor ce prezintă defect sau rezolvării ulterioare a problemelor (a depanării) când condițiile sunt propice.
- la granularitate mică diagnosticul se poate da în detaliu pentru o intervenție completă în depanare.

Cele două abordări prin prisma granularității au avantaje și dezavantaje, raportate la durata diagnozei și la complexitatea sistemului expert de diagnoză. Pentru a face o diagnoză corectă a defectelor ne interesează să depistăm cauzele utile depanării, fiind necesară o clasificare a cauzelor pe niveluri (secțiunea 3.4.4), acestea din urmă alcătuind diagnosticul.

Astfel, cauzele provoacă manifestări care conduc la diverse defecte regăsite în diferite stări funcționale ale unui sistem. În consecință, avem două modalități de lucru:

1. inventarierea tuturor cauzelor posibile,
2. inventarierea situațiilor de nefuncționare pentru a depista cauzele.

În cazul nostru, varianta aleasă este cea de-a doua, în care depistăm cauzele după care facem maparea cauză – efect. Pentru aceasta este necesar ca situațiile să fie clar stabilite pe fiecare nivel în parte, de aici rezultând simptoamele. De asemenea, este necesară inventarierea pe straturi a situațiilor de defect cât și cauzele care le-au provocat.

Abordarea achiziției de cunoștințe pe o structură arborescentă aduce avantajul că structurează cunoștințele până la nivelul de dată (valoare măsurată) venind în ajutorul diagnosticianului uman pentru a-și plasa cunoștințele din practică, cât și pentru expertul uman, care analizează funcționarea normală și la defect a ansamblului *top-down* a sistemului. Exista și o a doua abordare, utilizată de diagnosticianul uman care pe baza experienței practice „culege" cauze și efecte într-o maniera *bottom-up* (adică știe manifestările.....). Prima abordare o putem caracteriza tip *Deep Knowledge* iar cealaltă *Shallow Knowledge* (cunoștințe superficiale).

În cadrul monitorizării sistemului informatic abstractizat, cele mai importante variabilele de care vom ține cont sunt: conectivitatea IP, timpul de răspuns și încărcarea resurselor (secțiunea 3.4.5). Funcție de gradul de încărcare al resurselor, fie că este o mașină fizică sau mașină virtuală, folosind ecuația planului determinăm formula care ne asigură o modalitate de notare a sistemului în ceea ce privește buna funcționare a lui.

Ca atare, vom considera starea încărcării fiecărei variabile a sistemului în intervalul [0,1], unde valoarea 0 reprezintă o funcționare în parametri normali ai sistemului iar valoarea 1 reprezintă starea de funcționare în parametri nedoriți sau defect (nefuncționalitate).

3.5. Structurarea și reprezentarea cunoștințelor

În derularea proiectului de achiziție și structurare a cunoștințelor am situat pe primele poziții utilitatea produsului pentru utilizatorii finali și calitatea cunoștințelor cuprinse (corecte, complete și relevante) depozitate într-

un mod structurat. Pentru structurarea și reprezentarea cunoștințelor domeniului vizat considerăm necesară dezvoltarea unei ontologii proprii (secțiunea 3.6.1) care să conceptualizeze domeniul de cunoaștere, modelând relații și axiome, entități și atribute într-un anumit format, destinat procesării cu ajutorul calculatorului, având ca scop principal o conceptualizare complexă a domeniului nostru de interes. Rolul ontologiei este acela de a capta cunoștințele dintr-un anumit domeniu de interes și de modul în care entitățile pot fi grupate, ierarhizate, sau împărțite în funcție de asemănările și deosebirile lor.

În ciuda celor 30 de ani de investiții în acest domeniu, s-a constatat că există puține sfaturi practice și de orientare pentru cei care se implică în încercarea de a organiza și structura cunoștințele din orice sursă și prea puțină atenție i-a fost acordată procesului actual cu privire la implicarea utilizatorilor finali, contextului de aplicare și bineînțeles a cerințelor viitoare.

3.6. Abordarea propusă

Vizând crearea ontologiei domeniului virtualizării, optăm pentru adoptarea metodologiei prin prisma grupării straturilor (secțiunea 2.3) ce compun modelul întrucât permite captarea relațiilor dintre entitățile/serviciile care intră în componența unui strat cât și relațiile dintre componentele diferitelor straturi. Ne propunem să evidențiem anumite provocări privind modalitatea de reprezentare și structurare prin prisma straturilor (*Layers*) din componența modelului propus (secțiunea 2.3.1) și modul în care aceste provocări din domeniul IT vizat au fost abordate anterior.

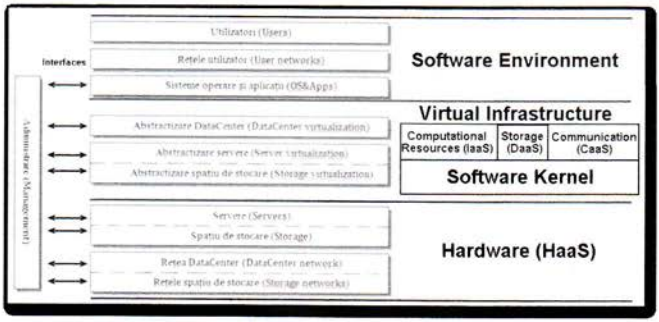

Figura 3.28. Modelul propus pentru abstractizarea infrastructurii IT

În dezvoltarea ontologiei vizăm *criteriile de stabilire a claselor, specificarea proprietăților și atributelor* și *a relațiilor*, funcție de straturile din componența modelului (Figura 3.28). Amintim aici că, în cadrul infrastructurii abstractizate, gruparea straturilor s-a făcut în funcție de mediul în care acestea sunt plasate (hardware, software, infrastructură virtuală, etc.), iar fiecare strat, datorită virtualizării, cuprinde unul sau mai multe servicii cu niveluri echivalente de abstractizare.

Urmărind diversele alternative viabile de modelare a domeniului în vederea dezvoltării unei ontologii, am procedat la o concepere în etape ținând cont de următorii pași: definirea claselor, aranjarea lor într-o ierarhie taxonomică (subclase și superclase), definirea atributelor și descrierea valorilor permise de acele atribute și umplerea cu valori a atributelor și instanțelor.

Ținând cont de stadiul actual al cercetării și de concluziile rezultate în urma analizei ontologiilor existente oferite de diverși dezvoltatori cu privire la crearea și utilizarea unei ontologii pentru domeniul virtualizării, rezultatul nu a fost pe măsura așteptărilor în ceea ce privește domeniul de interes întrucât mare parte din studiile efectuate au un caracter general și se limitează la infrastructura *hardware* sau superficial la *Cloud Computing*.

Distribuit de Semantic-Knowledge[*], *Tropes* este un procesor al limbajului natural și de clasificare semantică conceput pentru informatica, cercetare si studii științifice, care în urma analizei textului garantează pertinent calitatea rezultatelor. Cu ajutorul acestui editor freeware de ontologii și luând în calcul încercările precedente, vom înainta spre testare, validare și dezbatere o ontologie proprie concepută în urma analizei aprofundate a domeniului virtualizării. Suntem motivați să utilizăm acest software deoarece înglobează mai multe scenarii predefinite bogate în clasificări,care corespund diferitelor abordări privind clasificarea cuprinzătoare, detaliată și înalt specializată a documentelor.

Analiza domeniului se va face ținând cont de stadiul actual al cercetării privind ontologia existentă cu privire la virtualizare, a documentelor oferite de către dezvoltatorii de produse specifice virtualizării, a cărților,articolelor, discursurilor sau oricăror alte forme de text existente în literatura de specialitate. Astfel, scenariul inițial este îmbogățit prin adăugarea documentelor, cărților, manualelor de utilizare și articolelor oferite spre studiu de către dezvoltatorii de software, site-urile conferințelor, bazele de date internaționale și platformele inteligente de cercetare (ISI Web of Knowledge, IEEE Xplore, etc) din domeniul virtualizării.

3.6.1. Crearea ontologiei (scenariului)

Ontologia propusă se referă la structurarea ierarhică a cunoștințelor printr-o sub-categorisire a lor conform calităților esențiale pe care le au (relevante și/sau cognitive) și de asemenea prin construirea de structuri de cunoștințe care să permită diferitelor sisteme informatice să înțeleagă și să folosească reciproc conținutul celuilalt.

Pentru crearea ontologiei domeniului vizat ne bazăm pe următoarele componente principale: clase, atribute (proprietăți ale claselor), valori și relații (modul în care clasele sunt asociate între ele).

În cadrul cercetării întreprinse, structura este determinată de modul și scopul în care cunoștințele trebuie să fie reprezentate atât altor sisteme IT cât și utilizatorului. Avem două modalități de bază pentru reprezentarea lor: una bazată pe relații, alta pe atribute și valori. Prima poate fi vizualizată ca o rețea de concepte cu legături între ele, fiecare link reprezentând o legătură. Acesta este formatul de bază al unei scheme (concept harta) și este, de asemenea, baza celor logice. Cealaltă, bazată pe atribute și valori, este vizualizată cel mai bine ca un cadru (*frame*). Amintim aici faptul că unele baze de cunoștințe utilizează un format pur relațional, altele utilizează cadre, iar altele combina cele două formate.

Dispunând oficial de un număr impresionant de documente care vizează domeniul de investigație (Figura 3.29), îmbunătățim scenariul existent în vederea actualizării și filtrării claselor echivalente având ca scop definirea unei clasificări personalizate proprii domeniului vizat, practic crearea ontologiei domeniului.

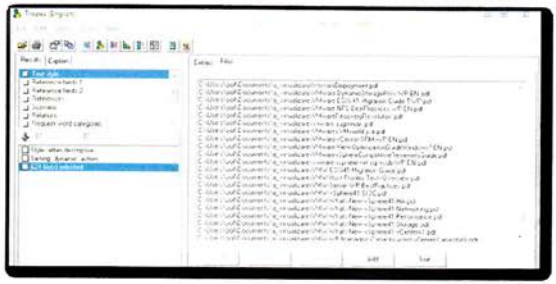

Figura 3.29. Încărcare documente în Tropes

[*] http://www.semantic-knowledge.com/

Detectarea imediată a contextului, izolarea temei și identificarea principalilor actori (clasele) se face prin aplicarea a trei niveluri de clasificări semantice. Este identificat stilul de text cu scopul de al plasa rapid în context și de a-l compara cu alte texte. Meta-categoriile semantice sunt utilizate pentru gruparea verbelor, adjectivelor, adverbelor, pronumelor personale si conjuncțiilor iar prin efectuarea unei analize analogice a textului izolăm principalele episoade și blocuri vizualizate de discuție prin identificarea unui mic grup de propuneri fundamentale unice regăsite în text (informații relevante pentru analiza discursului).

Utilizând instrumentul Scenariu (*Scenario*), care este un manager inteligent de lexicon bazat pe rețele semantice și tehnologii de analiza a textului limbajului natural, alături de care regăsim mai multe clasificări existente și pregătite pentru utilizare, evidențiem posibile grafice în urma analizei textului și sortăm automat lista de cuvinte cheie, ajutând practic imediat la vizualizarea și verificarea interpretării ipotetice. Scenariile sunt concepute pentru a îmbogăți și filtra clasele echivalente conform analizei strategice prin: definirea unei clasificări personalizate proprii (ontologii); modificarea sau restructurarea dicționarelor software-ului; înlocuirea lexiconului și personalizarea funcțiilor de recuperare a informațiilor dar și prin definirea caroiajului rezultat în urma analizei pentru generarea automată a statisticilor și raportului analizei în vederea indexării documentelor.

Indiferent unde este plasat textul, fie carte, articol, discurs sau orice altă formă, prin analiza de context aplicăm un set de tehnici pentru a reliefa (scoate în evidență) și înțelege contextul textului cu scopul determinării principalilor actori, a relațiilor mutuale în care se află aceștia cât și ierarhia și evoluția relațiilor (Figura 3.30). Contextul grupează substantivele principale în clase echivalente urmare a analizei textului. În studiul nostru a fost esențial să utilizăm instrumentul *Scenario* deoarece ne oferă o analiză corectă a textului chiar dacă acest lucru impune o refacere a clasificărilor, o corectare a oricăror anomalii întâlnite dar și o personalizare a propriei analize.

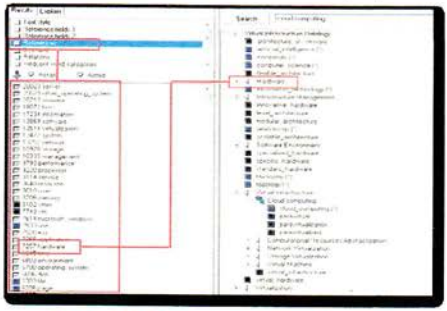

Figura 3.30. Creare scenariu funcție de context

De asemenea, unealta Scenariu a fost proiectată să rezolve ambiguitățile manual, acest lucru manifestându-se prin utilizarea claselor echivalente. Dacă eronat plasăm o clasă sau un atribut în două grupuri diferite, anomalia este semnalată și a doua înregistrare nu va fi luată în calcul.

În concluzie, pentru dezvoltarea scenariului am luat în calcul abordările precedente, iar în urma analizei colecției de documente reprezentative pentru domeniul supus investigației, am adăugat clasele, relațiile dintre acestea și atributele care au relevanță. După schițarea ontologiei, afișarea relațiilor și expresiilor propuse de către extractorul de terminologie cât și studierea listei cu verbe și adjective ne ajută la o mai bună clasificare a componentelor relevante din cadrul scenariului.

3.6.2. Extragerea terminologiei

În cadrul abordării propuse, cu ajutorul instrumentului de extragere a terminologiei am identificat cele mai semnificative expresii și combinații ale acestora, obținând astfel o clasificare mai precisă concomitent cu o îmbogățire rapidă a scenariului propus prin gruparea acronimelor și a expresiilor corespondente.

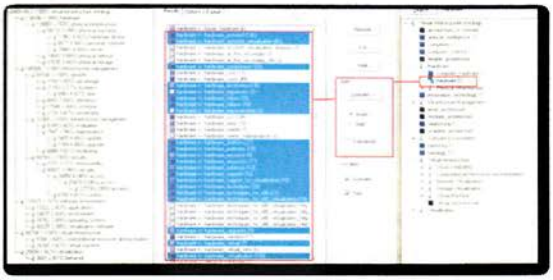

Figura 3.33. Selectare expresii pentru inserare în Scenariu

Termenii sau expresiile rezultate în urma extracției sunt precedate, funcție de cât de utilizate sunt în context, de un cod al culorilor, care variază pentru cele mai utilizate de la o nuanță întunecată, la una deschisă pentru cele nesemnificative. Această unealtă de extragere a terminologiei este legată semantic de instrumentul Scenariu deoarece la selectarea mai multor expresii corespondente unui termen, *Tropes* încearcă să poziționeze automat scenariul pe grupul care pare cel mai în măsură să înglobeze selecția.

3.6.3. Clasele echivalente și reprezentarea relațiilor

Pentru reprezentarea relațiilor avem la dispoziție patru tipuri de grafice: [*Arie*], [*Stea*], [*Distribuție*] și [*Episod*]. Cele mai utilizate grafice sunt primele două tipuri enumerate (arie și stea) acestea indicând clasele echivalente și relațiile dintre acestea. Graficul de distribuție afișează distribuția clasei echivalente sau a relațiilor dintre clasele echivalente iar graficul episoadelor afișează în ordine cronologică episoadele

Graficul arie (Figura 3.34) reprezintă clasele echivalente sub formă de sfere, suprafața lor fiind proporțională cu relevanța entității. Distanța dintre clasa centrală și celelalte clase este proporțională cu numărul de relații care le conectează, astfel, când două clase sunt apropiate una de cealaltă între ele sunt multe relații, pe când dacă sunt depărtate, acestea împart puține relații.

Figura 3.34. Grafic Arie

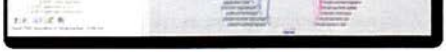

Figura 3.35. Grafic Stea

Graficul stea (Figura 3.35) afișează relațiile dintre clasele echivalente și numărul relațiilor existente dintre diferite clase echivalente. Putem astfel urma relațiile afișate pe grafic selectând printr-un click direct pe clasa care ne interesează. Această funcție ne permite o navigare între clase pentru o analiză a conexiunilor dintre diferite clase.

Graficul actorilor (Figura 3.36) prezintă concentrarea relațiilor dintre clase (principalii actori) și ne arată o comparație vizuală a „greutății" relațiilor dintre principalele grupuri ale scenariului. Concentrarea relațiilor este calculată pentru fiecare grup în parte ca fiind numărul total de relații divizat cu numărul de relații diferite.

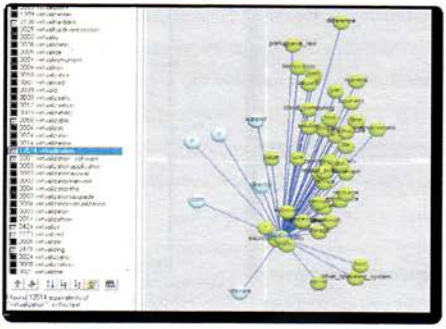

Figura 3.36. Grafic actori

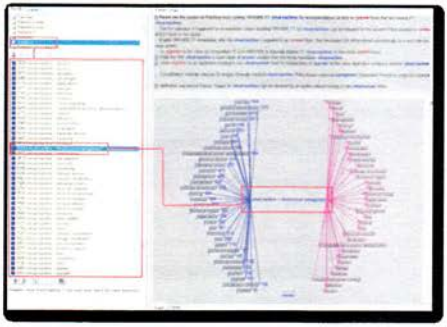

Figura 3.37. Relații dintre clase

În cazul relațiilor dintre diverse clase (Figura 3.37), graficul stea ne indică cele două clase centrale cu relațiile proprii precum și numărul de relații dintre entitățile claselor aflate în relație.

3.6.4. Validare ontologie

Organizând domeniul de interes în jurul conceptului de stivă, am determinat principalele clase pe care le-am structurat (aranjat) într-o ierarhie taxonomică (subclase și superclase), bazându-ne pe experiența din domeniu, pe cărțile, ghidurile de utilizare, articolele oferite spre studiu de către dezvoltatorii de software, site-urile conferințelor, bazele de date internaționale și platformele inteligente de cercetare existente (Figura 3.38, Figura 3.40).

Figura 3.38. Ierarhie taxonomică a claselor

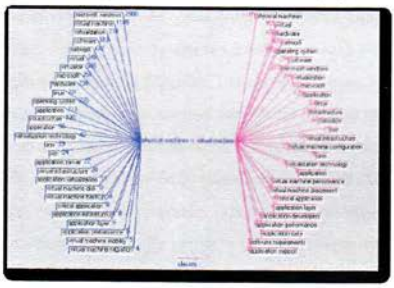

Figura 3. 39. Relațiile dintre clase

Specificarea relațiilor mutuale dintre aceste clase echivalente și numărul relațiilor existente dintre diferite clase echivalente este redat în Figura 3.39. Relațiile afișate pe grafic pot fi astfel urmate printr-o simplă selecție (printr-un click), direct pe clasa care ne interesează. Această funcție ne permite o analiză aprofundată a conexiunilor dintre diferitele clase datorită capacității de navigare între clase.

Figura 3. 40. Ontologia propusă

Specificarea atributelor (Figura 3.41), descrierea valorilor permise de acele atribute și umplerea cu valori a atributelor în scenariu se face utilizând instrumentele *Referințe1* și *Referințe 2* care nuanțează termenii specifici domeniului.

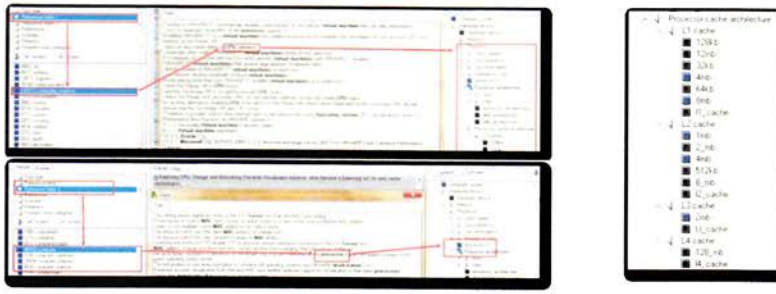

Figura 3. 41. Inserare atribute și valori

În concluzie, ontologia propusă conduce către stabilirea domeniilor de interes necesare Sistemului de Asistare a Diagnosticianului Uman – SADU – , subiect tratat în capitolul următor. Astfel sunt înaintate comunității specialiștilor spre testare și validare principalele componente ale ontologiei: clase (actori), proprietățile claselor (atributele), valorile și relațiile dintre clase (modul în care clasele sunt asociate între ele).

3.7. Concluzii și contribuții

Achiziția de cunoștințe implică crearea unui fond comun de cunoștințe, folosit pentru a oferi un produs final care înglobează mai multe aplicații. Problema reală cu care ne-am confruntăm în cazul achiziției de cunoștințe și anume extragerea unui imens volum de cunoștințe de la experții umani, a reprezentat-o numărul de resurse disponibile implicate pentru a face acest lucru cât mai eficient posibil. Acest lucru nu a fost ușor, în special atunci când a trebuit să ajungem la cunoștințele tacite („adânci") pe care le au experți sau în luarea deciziei corecte atunci când există un dezacord între opiniile lor. Pentru aceasta am considerat că avem nevoie de o abordare sistematică în care să dispunem de metodologii de urmărit și de utilizare a cadrelor de lucru (*frameworks*). Acestea ne-au ajutat să ne concentrăm pe cunoștințele de care avem nevoie și pe modalitatea de a le obține în cel mai eficient mod posibil.

Am reutilizat ori de câte ori a fost posibil, cunoștințele din proiectele anterioare sau de la modelele generice de cunoștințe, cum ar fi taxonomiile (practica și știința clasificării) generice. Datorită acestui fapt, nu am început cu o baza de cunoștințe goală, astfel folosind, îmbinând și adaptând „scheletele" existente am dezvoltat o nouă structură care ne-a oferit indicații privind sortarea cunoștințelor capturate. Pentru aceasta am folosit un software special care ne-a ușurat efortul, oferindu-ne un set de instrumente și tehnici pe care să le folosim atunci când achiziționăm, analizăm și modelăm cunoștințele. Astfel am fost nevoiți să învățăm de la alte persoane și la rândul nostru să diseminăm propria noastră experiență în achiziția de cunoștințe.

Contribuțiile aduse în acest capitol achiziției și structurării cunoștințelor fac referire la:

1. Metodologia corespunzătoare privind clasificarea sistemelor virtualizate prin prisma straturilor ce compun modelul în jurul conceptului de stivă (secțiunea 2.3). Permite ontologiei propuse captarea relațiilor dintre entitățile/serviciile care intră în componența unui strat cât și relațiile dintre componentele diferitelor straturi. De asemenea,ne permite o mai ușoară introducere în cunoașterea și înțelegerea domeniului virtualizării, a componentelor sale și a relațiilor dintre acestea

2. Structurarea domeniului virtualizării privind facilitarea achiziției de cunoștințe. Prin structurarea domeniului obținem avantajul că fiecare agent (uman sau inteligent) specializat pe un anumit strat va identifica și insera în baza de cazuri strict acele cazuri unde acesta este plasat, permițând o mai bună localizare, un nivel de adâncime a discriminării componentelor cât mai precis implicit o granularitate mică, diagnosticul în astfel de cazuri fiind în detaliu facilitând o intervenție completă în depanare.
3. Conceperea unei ontologii pentru domeniul nostru de interes contribuind astfel la:
   - Partajarea unui vocabular comun și al unei înțelegeri a structurii informațiilor în cadrul comunității și al agenților software.
   - Reutilizarea domeniului de cunoștințe pentru dezvoltări ulterioare.
   - Formularea într-un mod explicit a ipotezelor privind domeniul de interes.
   - Separarea cunoștințelor de domeniu de cunoștințele operaționale.
   - O analiză mai profundă a cunoștințelor domeniului de interes.
   - Utilizarea bazei de cunoștințe pentru a împărtăși cunoștințele cu alte sisteme de calcul sau ca parte din dezvoltarea unui sistem inteligent computațional.

4. DIAGNOZA DEFECTELOR.

Diagnoza reprezintă identificarea naturii sau cauzei unui fenomen fie prin procesul de eliminare, fie prin metode analitice. În informatică, diagnoza se ocupă cu dezvoltarea de algoritmi şi tehnici care sunt în măsură să stabilească dacă comportarea funcţională a sistemului se încadrează în parametri nominali.

Datorită virtualizării, infrastructura hardware comportă stări diferite de cele cunoscute datorită creşterii gradului de utilizare a resurselor, puterii de calcul necesară dispozitivelor fizice ce suportă multiple maşini virtuale, dispozitivelor I/O precum şi capacităţii reţelei de a satisface cererile de transfer dintre maşinile virtuale si cele fizice. Cunoscând beneficiile virtualizării (secţiunea 2.3.3) şi vizând scopul de a pune bazele unui sistem instruibil care să sprijine deciziile diagnosticianul uman în rezolvarea problemelor apărute, sistemele IT abstractizate conduc la provocări pentru inginerii de sistem în a înţelege şi depana posibilele probleme ce pot surveni pe tot parcursul funcţionării dar mai ales în a depista manifestările pe fiecare nivel ce intră în componenţa sistemului ţintă în vederea localizării şi remedierii defectului conform ipotezelor şi investigaţiilor suplimentare.

Astfel, din punct de vedere a preciziei, complexităţii, performanţei şi adaptării la schimbări nu există o tehnică unică care să fie considerată cea mai bună în rezolvarea problemelor generice din diagnosticarea defectelor, de aceea, mulţi cercetători încearcă să combine diferite tehnici pentru a elabora o soluţie mai bună la problema apărută (S. Katker, M. Paterok, 1997).

În general, abordările bazate pe reguli (secţiunea 4.4.3) pot fi folosite pentru un sistem simplu, care este rar schimbat pe când sistemele bazate pe model (secţiunea 4.4.2) prezintă un model suplimentar al sistemului cu privire la reguli, ceea ce le face superioare sistemelor bazate pe reguli dar nu le face mai atractive datorită obţinerii şi actualizării modelului cu dificultate.

Reţelele neuronale si arborii de decizie se bazează pe o perioadă lungă de formare şi nu pot lucra în afara zonei de formare (secţiunea 4.4.3).

Tehnica cărţii de criptare - *Codebook* (secţiunea 4.4.2) este interesantă datorită performanţei şi robusteţii sale, dar ridică probleme deoarece necesită o procedură suplimentară privind gestionarea schimbărilor din sistem. În plus, nu poate lucra atunci când survine simultan mai mult de o defecţiune odată cu suprapunerea mai multor simptome.

Tehnica CFG (gramatica fără context) este atractivă datorită capacităţii de a modela ierarhic sistemul dar inconvenientul rezidă din faptul că toţi algoritmii disponibili, aplicabili pentru modelul construit sunt prea complicaţi pentru a fi folosiţi în sistemele reale (secţiunea 4.4.2).

Tehnicile de traversare a modelului (secţiunea 4.4.1), deşi sunt rezistente în cazul schimbărilor frecvente privind configurarea sistemului, acestea nu pot modela situaţia în care defectul unui obiect poate depinde de defectul sau de o combinaţie de defecte ale altui obiect. Cea mai accesibilă idee aflată în spatele multor tehnici de localizare a defectelor este considerată corespondenţa alarmă/eveniment datorită capacităţii mari de a restabili relaţii între alarme-evenimente.

Cu toate acestea, tehnicile descrise în acest capitol sunt destul de mulţumitoare în cazul monitorizării sistemelor care nu au o complexitate crescută, de aceea în cazul sistemelor inteligente virtualizate vom aborda o strategie distribuită. Astfel, sistemul complex va fi divizat în niveluri de interes (subsisteme) unde anumite tehnici prezentate mai sus vor putea fi aplicate cu succes.

Deși sistemele bazate pe cazuri (secțiunea 4.4.3) sunt mai puțin sensibile la schimbările din sistem, schimbarea modelului de la o orientare centrată pe defect la una bazată pe urme (secțiunea 4.5) pentru a facilita sau amplifica propriile capacități ale diverșilor experți umani sau utilizatori constituie punctul forte în diagnoza sistemelor virtualizate datorită modelării interacțiunilor dintre resursele fizice și cele virtuale.

Multe cercetări tratează urmele (secțiunea 4.5) pentru diagnoza privind comportamentul software (Diekert V. and Rozenberg G., 1995) sau pentru diagnoza comportamentului uman (Penelope S. and Fisher C., 1994). Dacă unele dintre aceste cercetări se concentrează pe o secvență de căutare în urme (M. Steinder and A. S. Sethi, 2004), pentru adaptarea contextului propriu nici una dintre ele nu utilizează urmele în vederea reutilizării experiențelor anterioare. Etapa de elaborare a raționamentului bazat pe urme a căpătat o importantă deosebită studiind noțiunea de „puncte de vedere" (Karacapilidis N. et al., 1997), precum și conceptul de „conversație CBR" (Aha D.W. et al., 2001). Acest pas se bazează pe sistemul de cunoștințe, având scopul de a ajuta utilizatorii să-și descrie propriile lor problemele. Cu toate acestea, sistemele CBR nu utilizează urmele pentru a ajuta utilizatorii în această sarcină complexă, cu toate că ar putea fi util, de exemplu pentru a extinde cu ușurință contextul de studiu.

Intr-adevăr, sistemele CBR (secțiunea 4.4.3) dobândesc noi cunoștințe prin acumularea de cazuri reprezentând episoade de rezolvare a problemelor, pe care le memorează într-un mod predefinit, astfel încât acestea să poate fi refolosite pentru rezolvarea problemelor viitoare. Propunerea este să mergem mai departe de colectarea si codarea episoadelor de rezolvare a problemelor si să considerăm urmele interacțiunii unei activități, ca un fond comun (*pool*) în care diferitele situații de rezolvare a problemei pot fi găsite. Construim astfel o variantă extinsă a celei dezvoltate și dezbătută de Alain Mille (2006), din punct de vedere al reprezentării cunoștințelor de către CBR, pe care a numit-o raționament bazat pe experiența urmelor (*traced experience based reasoning TBR*). În abordarea propusă nu urmărim să înlăturăm raționamentul bazat pe cazuri sau cel bazat pe urme, ci dorim o îmbinare (Figura 4.12) a celor două forme de raționament (secțiunea 4.5) în care rolul central revine utilizatorilor deoarece ei sunt implicați în elaborarea unei probleme, în regăsirea unei experiențe anterioare, precum și în adaptarea soluției. Noutatea abordării constă în stocarea modelelor de rezolvare a problemei într-o bază de date nu doar în urme, permițând astfel fiecărui utilizator să aibă propriul său punct de vedere cu privire la o problemă, care va fi împărtășită comunității utilizatorilor.

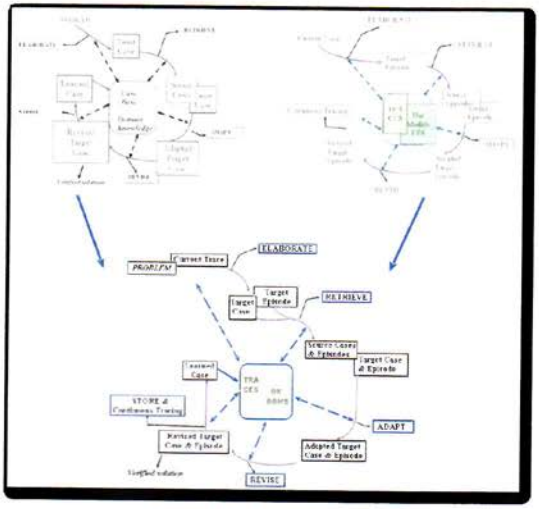

Figura 4. 12. Ciclul CBR + TBR

4.6. Sistemul de asistare a diagnosticianului uman (SADU)

După cum am menționat anterior, am urmărit realizarea unui **sistem de asistare a diagnosticianului uman** (secțiunea 4.6), în sensul că acest sistem va dispune de cunoștințele diagnosticianului și apoi de informații preluate prin interacțiune cu omul la solicitarea SADU.

Datorită virtualizării și posibilității migrării mașinii virtuale sau aplicației în cadrul infrastructurii, contextul activității urmei este deosebit de important în abordarea asumată. Urmărim ca fiecare entitate (mașină virtuală, aplicație, etc.) să conțină acel istoric al defectelor (istoric locație, strat, situație - ce se poate întâmpla -, cauză, mărime, soluții similare, etc.) indiferent de locația pe care o poate avea la un anumit moment dat.

Crearea contextului activității urmei cât și a instrumentelor necesare colectării, transformării, depozitării, manipulării și vizualizării cazurilor le realizăm prin:

- Prezentarea și exploatarea redundanței episoadelor la niveluri de virtualizare diferite: mașina virtuală, stratul de virtualizare, mașina fizică, mecanismele de administrare ale inginerului de sistem/diagnosticianul uman dar și de identificat, discutat și exemplificat specificul și conținutul episoadelor pe niveluri de abstractizare;
- Prezentarea și exploatarea „serializării" episoadelor ca evenimente care apar sau sunt culese secvențial în funcționarea sub-sistemelor componente ale sistemului virtualizat, prin care secvența de mesaje în timp oferă informație redundantă de valoare pentru rafinarea diagnozei.
- Exploatarea „serializării" episoadelor este îmbogățită cu informații preluate la momente (contexte) prestabilite, atât a valorilor numerice (de la senzori și parametri de performanță sau capacitate a resurselor), cât și a valorilor calitative de la diagnosticianul uman (preluate sau măsurate la inițiativa omului).
- Secvența de evenimente preluate și completate cu informații de la senzori și diagnosticianul uman (toate serializate implicit de sistemul virtualizat sau prin culegerea de către SADU a informațiilor de la senzori/om) – reprezintă apoi „semnături" sau „urme" care vor fi folosite la diagnoză.
- Semnăturile (secvențele) vor fi folosite parțial sau integral (de exemplu se așteaptă apariția unui eveniment cheie care încheie „lanțul" de manifestare (lanț de derulare a proceselor) suficient pentru aplicarea diagnozei.

Au fost prezentate arhitectura și funcționarea SADU (secțiunea 4.6.2), funcție de contextul activității urmei (secțiunea 4.6.1) precum și o descriere amănunțită a aplicației (secțiunile 4.6.3 – 4.6.7), de unde au rezultat concluziile (secțiunea 4.7).

Figura 4.27. Pagina principala de acces

Figura 4.26. Rezolvare eveniment

4.8. Contribuții

Contribuția majoră din acest capitol s-a concretizat în dezvoltarea unui sistem de asistare a diagnosticianului uman. Contribuția importantă adusă de SADU provine din faptul că sistemul de diagnoză este funcțional din primul moment, fără a necesita întreaga cazuistică inserată legată de contextul activității urmei, deoarece aceasta se completează pe parcursul desfășurării proceselor, iar în cazul apariției defectelor necunoscute, fără rezolvare imediată, problema este lăsată în discuție/analiză pentru rezolvare comunității specialiștilor din domeniu.

De altfel, la construirea SADU îmbinând cele două tehnici CBR și TBR, regăsirea și adaptarea într-o astfel de abordare diferă, pornind de la caz la urme (secțiunea 4.5). Prin urmare, înainte de a efectua orice raționament bazat pe cazuri, este necesar să construim „cazurile" pornind de la urme. Acest lucru subliniază importanța elaborării pasului de regăsire în TBR dar și marele avantaj, care permite unui utilizator să vadă rezultatul direct al cercetării sale. Procesul oferă posibilitatea identificării imediate în cazul unei nepotriviri a măsurii similitudinii.

În acest caz, noul sistem de raționament lasă utilizatorului posibilitatea să aleagă modalitatea de perfecționare a măsurii, de a adopta alta, sau de continuare a ciclului cu alte tipuri de cunoștințe (de adaptare sau de context) dacă problema pare să aibă o rezolvare bună. De asemenea, el poate decide, dacă oprește procesul aici sau îl continuă.

Stocarea în baza de date a contextului specific și a cunoștințelor de adaptare, în special atunci când acestea sunt folosite în comun de către mai mulți utilizatori, conduce la rezolvarea problemei privind automatizarea adaptării atunci când știm cum să adaptăm contextul. Mai mult decât atât, diferitele tipuri de cunoștințe sunt interconectate în noul sistem de raționament de aceea trebuie să le lăsăm să se schimbe într-un mod cât mai natural.

Atunci când o problemă nu poate fi rezolvată cu un episod regăsit, acest episod trebuie extins, prin extinderea cazului urmărind la ce s-a întâmplat „înainte de" sau „în timpul".

Interacțiunea om-calculator cât și abilitățile și implicarea utilizatorului joacă un rol important în determinarea cu precizie a momentului în care contextul trebuie extins. Propriile noastre urme sunt ușor de vizualizat. În tot acest context, „cunoaștem" acțiunile întreprinse și suntem în măsură să înțelegem sensul acțiunilor noastre. Cu toate acestea, folosirea urmelor cu scopul de a „învăța" din experiență reprezintă o sarcină destul de complexă ce implică extinderea conceptului prin partajarea experiențelor utilizatorilor, nu limitarea lui doar la experiența unuia, dacă ținem cont că o urmă ar putea fi rezultatul mai multor interacțiuni ale diferiților utilizatori. SADU permite unui utilizator să beneficieze de urmele provenite de la alți utilizatori. Urmele, datorită „reținerii" întregului context al experienței anterioare, ne permit mai multe modalități de interacțiune în sisteme și în special combinarea mai multor modalități de interacțiune într-un singur sistem. Avantajul rezidă din faptul că utilizatorii diferiți, cu diverse abilități sau obiceiuri, vor putea să-și împărtășească experiența.

Prin asocierea dintre CBR și TBR oferim rolul central utilizatorilor deoarece ei sunt implicați în elaborarea unei probleme, în regăsirea unei experiențe anterioare, precum și în adaptarea soluției. Ca rezultat, ei oferă sistemului abilități de achiziție a cunoștințelor continue prin elaborarea de cunoștințe în timpul fiecărei interacțiuni. Practic, sistemul oferă utilizatorilor urme reformulate, fiecare utilizator putând transforma urmele folosind propriile sale cunoștințe pentru problema ce trebuie rezolvată. Cu acest mod de abordare, fiecare utilizator poate avea propriul său punct de vedere cu privire la o problemă.

Credem că acest tip de raționament mixt (Figura 4.12), bazat pe cazuri și urme, va avea un impact semnificativ asupra experienței de partajare a aplicațiilor, în special atunci când acestea sunt bazate pe web și sistemul de raționament bazat pe experiență va fi partajat de către comunitatea utilizatorilor.

5. ABORDĂRI APROXIMATIVE PRIVIND ALOCAREA ŞI ÎNCĂRCAREA RESURSELOR PENTRU SADU

Propunem două abordări privind încărcarea şi alocarea resurselor din cadrul unui sistem informatic, în cazul nostru abstractizat. Abordarea aproximativă (fuzzy) vine în sprijinul diagnosticianului uman pentru o cunoaştere şi interpretare mai eficientă în contextul diagnozei şi al deciziei, iar abordarea cantitativă şi deterministă este o abordare matematică (computaţională) de notificare şi apreciere a stării funcţionale a unui sistem în vederea alocării dinamice şi echilibrării încărcării resurselor.

5.1. Alocarea dinamică şi echilibrarea încărcării resurselor din sistemele fizice abstractizate, utilizând logica fuzzy.

Studiul priveşte optimizarea utilizării resurselor disponibile dintr-un sistem, implicit a proceselor, ţinând cont de cererile sosite şi eliminând tot ce ar putea produce o stare de nefuncţionalitate.

Dacă în mod tradiţional prezenţa stării de nefuncţionalitate dintr-un sistem se reduce la intervenţia asupra nodului respectiv, prin virtualizare defectarea unui nod poate conduce la o cascadă de defecte, de aceea, este indicat să luăm în considerare ca aceste sisteme să fie tolerante la defect şi recuperare.

În cazul virtualizării, alocarea dinamică şi echilibrarea încărcării resurselor sistemului abstractizat reprezintă procesul de îmbunătăţire a performanţelor unui sistem ce dispune de un număr de „m" resurse de acelaşi tip, prin distribuirea sarcinii între acestea (Andrews G. R., 1982), având ca scop minimizarea timpului de răspuns, fără a afecta stabilitatea funcţională a sistemului (Sharma S. *et al.*, 2008). Astfel, studiile anterioare cu privire la echilibrarea încărcării resurselor nu iau în considerare incertitudinea şi imprecizia în ceea ce priveşte informaţiile referitoare la starea sistemului, fapt ce constituie un punct de plecare în vederea utilizării intrărilor *crisp* în modelare, datorită logicii *fuzzy*.

Prin folosirea acestui tip de logică, propunem o nouă abordare privind algoritmul de alocare dinamică a resurselor fizice şi de echilibrare a încărcării resurselor abstractizate ale sistemului, demonstrând astfel faptul că, raţionamentul prezentat va face faţă incertitudinii şi impreciziei, dar şi faptul că algoritmul nostru ne poate spune cu precizie starea funcţională a întregului sistem. Un motiv plauzibil pentru care utilizăm abordarea fuzzy este acela că, la ieşirea controlerului, dorim să oferim informaţii/cunoştinţe „calitative" diagnosticianului uman privind situaţia încărcării resurselor – ca să înţeleagă şi să facă diagnoza unor situaţii speciale, ca să le folosească la politici de management ale sistemului virtual (de exemplu pentru upgrade sau în cazuri de cădere sau de întreţinere ale sistemului etc.). Clarificăm aspectele legate de *eterogenitatea nodurilor, încărcarea nodului şi reţelei, defectarea nodului* precum şi *conceptele de bază ale teoriei mulţimilor vagi* (secţiunea 5.1).

Raţionamentul fuzzy (aproximativ) reprezintă o procedură de inferenţă care furnizează concluzii pe baza unui set de reguli fuzzy „dacă – atunci" şi a unui set de fapte cunoscute (Jang and Sun, 1995; Jang, et al., 1997). În logica binară tradiţională, raţionamentele sunt asemănătoare celor umane şi au la bază metoda de inferenţă „Modus Ponens" (MP) conform căreia se poate deduce valoarea de adevăr a unei propoziţii B cunoscând valoarea de adevăr a propoziţiei A care o implică. Construirea comenzilor ce vor fi aplicate va rezulta în urma operaţiei de inferenţa.

Inferența este operația logică prin care se obțin concluzii valabile pe baza unor premize. (A. Charuk, 1936) a demonstrat că logica predicatelor nu este o teorie complet decidabilă și deci nu există o metodă universală prin care într-un număr finit de pași să se decidă dacă o formulă bine compusă (formulată) este validă sau infirmată. Există totuși mai multe metode de inferență cum ar fi Modus Ponens, Modus Tollens, raționamentul ipotetic sau tranzitivitatea. În logica fuzzy se utilizează principiul rezoluției care necesită aducerea propozițiilor la o formă standard numită formă clauzală.

Echilibrarea încărcării se reduce la partajarea volumul de încărcare computațional între multiple calculatoare sau de a lucra practic ca un singur calculator virtual dacă dispunem de o infrastructură alcătuită din computere *standalone* sau se referă la redistribuirea/realocarea resurselor, de la mașinile care nu prezintă o încărcare peste un anumit prag către cele care necesită o putere mare de calcul la un moment dat sau când situația o cere datorită supraîncărcării (depășirii unui prag) în cazul sistemelor a căror componente fizice sunt abstractizate.

Uzual, algoritmii de echilibrare a încărcării resursei au ca referință un indice al încărcării, care oferă o măsură a volumului de muncă prezent într-un nod, relativ la media globală, și de asemenea, anumite politici care reglementează acțiunile întreprinse din momentul detectării unei încărcări echilibrate a resursei (Shivaratri N. G. et al., 1992). Mai mult, indicele de încărcare este folosit pentru a detecta prezența unei stări de dezechilibru în ceea ce privește încărcarea resursei. Calitativ, o stare de dezechilibru privind încărcarea unei resurse se produce atunci când indicele de încărcare al unui nod este mult mai mare (sau mai mic) decât indicele altor noduri. În cazul resurselor multiple (procesor, memorie, disc, etc.), o măsură îmbunătățită a performanței privind timpul de răspuns poate fi dată doar de ponderea pe care o au resursele respective în modalitatea de apreciere a stării funcționale a sistemului (Ferrari D. and Zhou S., 1987), (Leinberger W. et al, 2000).

Un aspect dificil al echilibrării încărcării îl reprezintă alegerea algoritmului (secțiunea 5.1.1) pe care trebuie să-l folosim, ținând cont de faptul că de-a lungul timpului s-au propus multe variante, fiecare cu un scop și aplicabilitate bine definite și care de cele mai multe ori nu are aplicabilitate generală. Algoritmii de echilibrare a încărcării (statici sau dinamici) au ca principiu de funcționare o realocare a sarcinilor suplimentare de lucru ce depășesc un anumit prag în raport cu volumul total de muncă.

Secțiunea 5.1.1 descrie cele patru politici (informația, transferul, locația și selecția) care guvernează acțiunea algoritmului de echilibrare a încărcării atunci când este detectat un dezechilibru privind încărcarea unei resurse.

5.1.3. Algoritmul de echilibrare a încărcării

Sistemele IT virtualizate sau distribuite au în componența lor entități eterogene, cu performanțe diferite, fapt ce implică un studiu atent privind echilibrarea încărcării, care trebuie să fie direct proporțională cu performanța entității. Vom utiliza termenul de „nod" folosit în rețelele de calculatoare pentru o unitate din multitudinea de resurse disponibile de același tip (aplicație, resursă, mașină virtuală, mașină fizică, *switch*, etc.).

Algoritmul propus se adresează în primul rând infrastructurilor care au în componență aplicații accesate de numeroși clienți, care se execută simultan/paralel pe mașini diferite. Fără să ținem seama de faptul că mașinile care suportă aplicațiile din cadrul infrastructurii sunt *standalone* sau abstractizate, modelul își propune să indice în timp optim gradul de încărcare și starea funcțională globală a întregului sistem dar și o echilibrare, în același timp, a încărcării nodurilor/resurselor, prin redistribuirea sarcinilor/clienților de la un nod încărcat (expeditor) către un nod eligibil (receptor) ce prezintă o încărcare sub prag. Sunt descrise metodologia propusă (secțiunea 5.1.2) și noțiunile utilizate în proiectarea algoritmului (secțiunea 5.1.3).

Indice relativ de performanță a nodului– IRP – reprezintă valoarea relativă a performanței unui nod, funcție de IN (încărcarea procentuală a nodului) și IP (indicele de performanță).

Calcul IRP: pentru fiecare nod „i", i∈ [1,N]:

$$IRP_i = \frac{IP_i}{IP_{max}}$$

Indice încărcare nod – IIN – reprezintă o măsură a încărcării efective a nodului raportată la întregul sistem, funcție de IRP

$$IIN = IN * IRP$$

Indice încărcare globală – IIG – reprezintă un indicator sintetic pe ansamblul sistemului calculat ca fiind media ponderată a încărcării tuturor nodurilor, acesta fiind considerat și pragul de încărcare.

$$IIG = \frac{1}{N}\sum IIN\,; N = numărul\ total\ de\ noduri$$

Indice supraîncărcare – IS – reprezintă capacitatea unui nod de a primi noi sarcini adică de a fi Receptor

$$IS = IIN - IIG : \begin{Bmatrix} > 0\,; potential\ expeditor \\ \leq 0; potential\ receptor \end{Bmatrix}$$

Actualizarea IIN se poate face în două moduri și anume, la perioade fixe de timp, sau la apariția unui eveniment, adică în momentul în care un nod depășește nivelul de dubiu al încărcării critice.

Considerăm că în cadrul unei infrastructuri dispunem de „m" noduri, unde fiecare nod poate fi o combinație complexă alcătuită din mai multe tipuri de resurse (CPU, RAM, HDD, etc.) dar și faptul că, pentru fiecare nod, configurația fizică a resurselor poate fi eterogenă. Cuantumul resurselor alocate unui nod poate fi destul de diferit față de alt nod atât ca număr de resurse dar și ca mod de configurare. În plus, alocarea resurselor identice pe fiecare nod poate fi diferită. De asemenea, fiecare nod are o partiționare fuzzy relativă la gradul global de încărcare a sistemului.

Toate aceste considerente conduc la performanțe diferite ale nodurilor. De exemplu, un nod (mașină virtuală) poate avea alocată o cantitate de memorie (relativ) mare, raportată la numărul de procesoare pe care le posedă respectivul nod în timp ce, un alt nod poate avea un număr mai mare de procesoare, respectiv cu o alocare de memorie mai mică (Darbha S. and Agrawal D. P., 1998), (Munir S. A. et al., 2007). Identic exemplului descris anterior, la un moment dat, într-un nod pot sosi foarte multe cereri simultan, provenite de la mulți clienți, lucru ce conduce la o funcționare în parametri nedoriți, sau chiar o blocare a aplicației, mașinii virtuale sau chiar a infrastructurii.

Modelul supus atenției (Figura 5.1) are în componență următoarele blocuri: tabelul de distribuire, indicele de încărcare nod, tabelul de costuri și controlerul fuzzy care gestionează echilibrarea încărcării resurselor de același tip din cadrul infrastructurii IT.

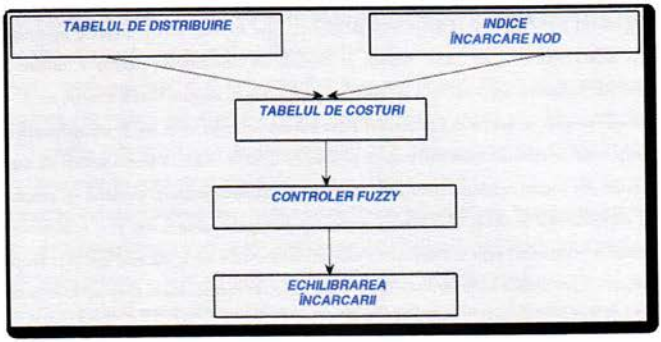

Figura 5.1. Modelul propus

Tabelul de distribuire conține legăturile de comunicare dintre nodurile compatibile ale sistemului (algoritmul de distribuție este tratat pe larg de Ioan, C.A., 2008).

Adoptăm „strategia proximității", reducând pe cât posibil transportul de sarcini sau clienți, astfel un client se poate conecta la un nod îndepărtat doar dacă nu dispunem de un nod eligibil mai apropiat. Totodată, când resursa (nodul) servește scopul în sine și nu disponibilitatea limitată pe care administratorul o decide ajungem la noțiunea de „acces strategic". Această strategie reprezintă fundația pentru sistemul nostru de cerere/distribuție care oferă garanția că putem satisface cererile clienților oferind acces pentru fiecare, atunci când dorește.

Tabelul de costuri ne oferă informații cu privire la costurile de transport dintre noduri dar și numărul de noduri ce prezintă o încărcare ce depășește pragul mediu (IIG). Tabelul de costuri este obținut prin utilizarea indicelui de încărcare și tabelului de distribuire. Statutul unui nod se stabilește pe baza tabelului de costuri, utilizând un controler și reguli fuzzy.

Sunt detaliate caracteristicile și structura de bază a sistemului fuzzy utilizat. Acesta prezintă două intrări și o singură ieșire pentru variabilele lingvistice conform diagramei sistemului (Figura 5.2).

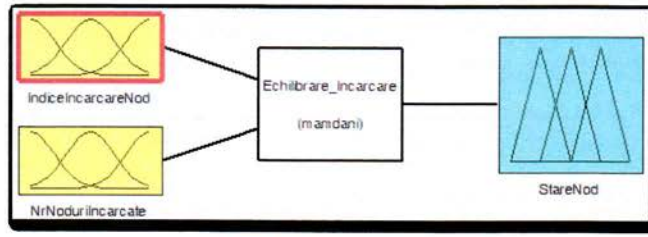

Figura 5. 2. Diagrama sistemului fuzzy

Echilibrarea încărcării este declanșată de către un nod care are calitatea de Expeditor. Astfel, dacă dorim o echilibrare corectă a încărcării, avem următoarele situații:

În cazul în care se urmărește transferarea unei sarcini selectate către orice Receptor compatibil urmăm regulile:

- R1:Dacă Nod este Expeditor atunci selectează un nod compatibil ca partener de migrare (din tabelul de distribuire) care să prezinte calitatea de Receptor (IIN mic) la un cost cât mai mic (conform tabelului de costuri);
- R2:Dacă Nod este Expeditor și nu găsește Receptor (partener de migrare) atunci STOP;
- R3: Dacă Nod este Expeditor și găsește Receptor atunci selectează o sarcină în vederea transferului;
- R4: Dacă Nod Expeditor eșuează în a transfera o sarcină Receptorului selectat atunci selectează un alt Receptor.
- R5: Dacă sarcina selectată de Nod Expeditor nu poate fi transferată către un Receptor compatibil atunci selectează altă sarcină.

Sau, dacă urmărim ca celui mai apropiat Receptor compatibil sa-i fie transferată orice sarcină avem regulile:

- R1:Dacă Nod este Expeditor atunci selectează un nod compatibil ca partener de migrare (din tabelul de distribuire) care să prezinte calitatea de Receptor (IIN mic) la un cost cât mai mic (conform tabelului de costuri);
- R2:Dacă Nod este Expeditor și nu găsește Receptor (partener de migrare) atunci STOP;

- R3: Dacă Nod este Expeditor și găsește Receptor atunci selectează o sarcină în vederea transferului;
- R4: Dacă Nod Expeditor eșuează în a transfera o sarcină Receptorului selectat atunci selectează altă sarcină.
- R5: Dacă sarcina selectată de Nod Expeditor nu poate fi transferată către un Receptor compatibil atunci selectează alt Receptor.

5.1.4. Metodologia propusă

Fuzzyficarea este o operație prin care se aleg mărimile de intrare și de ieșire după care se definesc mulțimile fuzzy care permit descrierea acestora prin variabile lingvistice, rezultând cadrul cognitiv sau partiția fuzzy. Procedăm la o clasificare a valorii indicelui (gradului) de încărcare a nodului (IIN) în cinci categorii, bazat pe un anumit prag (IIG). Acest indice de încărcare a resursei este definit între 0 și 1, având pragul d.

Intrarea 1 din controlerul fuzzy, și anume, variabila lingvistică ce descrie valoarea indicelui de încărcare a nodului (IIN), folosește cinci mulțimi fuzzy (Figura 5.3), iar gradul de încărcare a resursei prezintă următorii termeni lingvistici: încărcare minimă (min), încărcare moderata, încărcare medie (prag), încărcare mare, încărcare critică (max).

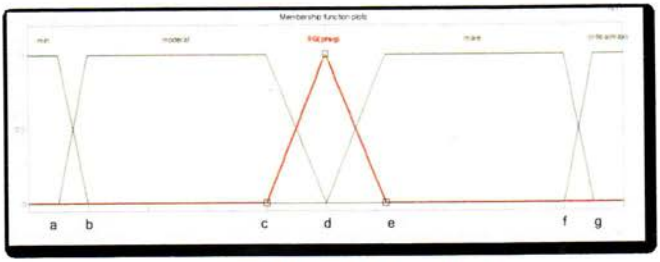

Figura 5.3. Indicele de încărcare fuzzy

Pentru intrarea 2 utilizăm notația NNI pentru variabila lingvistică care descrie numărul de noduri încărcate peste pragul IIG din totalul nodurilor prezente în sistem.

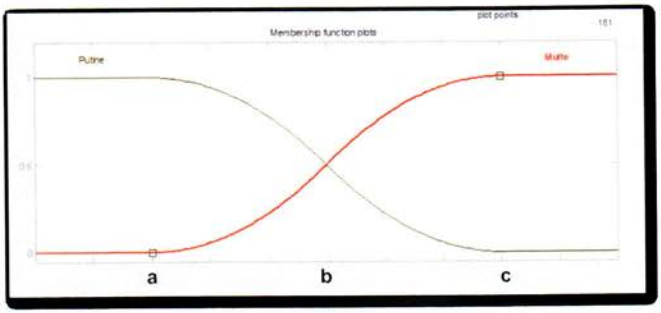

Figura 5.4. Numărul de noduri NNI

Mulțimile fuzzy ce reprezintă numărul nodurilor încărcate peste pragul IIG au termenii lingvistici: puține și multe (Figura 5.4). În acest caz, funcția de apartenență (sigmoidă Z și S) pentru intrarea 2 devine:

$$\mu_{(putine)}(NNI) = \begin{cases} 1 & ; \quad NNI \le a \\ 1 - 2\left(\frac{NNI - a}{c - a}\right)^2 & ; a \le NNI \le b \\ 2\left(\frac{NNI - c}{c - a}\right)^2 & ; b \ge NNI \ge c \\ 0 & ; \quad NNI \ge c \end{cases}$$

$$\mu_{(multe)}(NNI) = \begin{cases} 0 & ; \quad NNI \le a \\ 2\left(\frac{NNI - c}{c - a}\right)^2 & ; \quad a \le NNI \le b \\ 1 - 2\left(\frac{NNI - a}{c - a}\right)^2 & ; \quad b \le NNI \le c \\ 1 & ; \quad NNI \ge c \end{cases} ; b = \frac{a+c}{2}$$

În cadrul modelului propus avem o singură ieșire care formează cadrul cognitiv, termenii lingvistici fiind modelați prin mulțimi fuzzy de formă Z și S pentru termenii externi, conform Figura 5.7. Se observă că o valoare de ieșire poate însă aparține mai multor mulțimi fuzzy.

Scopul fuzzyficării este să permită construirea unei baze de reguli înglobând cunoștințele noastre referitoare atât la starea sistemului cât și la metodele de echilibrare a încărcării pe care vrem să le aplicăm. Utilizăm în modelului nostru doar clauze cu premise complexe și consecință unică, astfel încât, orice formulă bine formată și închisă poate fi adusă la forma clauzală. Putem astfel materializa o mare varietate de reguli de inferență logică utilizând diferiți operatori logici (conjuncție ∧, disjuncție ∨, echivalență, negație, implicație, etc.) în diferite poziții ale clauzelor.

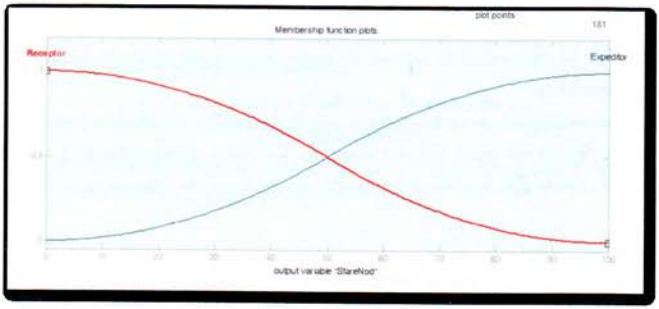

Figura 5.7. Ieșire 1 din controlerul fuzzy (echilibrarea încărcării)

Din modul de alcătuire a cadrului cognitiv se cunoaște faptul că, din setul de reguli de conducere, mai multe pot fi activate în același timp, fiecare dintre ele putând induce câte o comandă proprie. Totodată, inițializarea algoritmului de echilibrare a încărcării o face nodul care are statutul de Expeditor.

Baza de reguli (cunoștințe) propusă cuprinde următoarele reguli (Figura 5.8):

- R1. If (IndiceIncarcareNod is min) then (StareNod is Receptor)
- R2. If (IndiceIncarcareNod is moderat) and (NrNoduriIncarcate is Putine) then (StareNod is Expeditor)
- R3. If (IndiceIncarcareNod is moderat) and (NrNoduriIncarcate is Multe) then (StareNod is Receptor)
- R4. If (IndiceIncarcareNod is IIG(prag)) and (NrNoduriIncarcate is Putine) then (StareNod is Expeditor)
- R5. If (IndiceIncarcareNod is IIG(prag)) and (NrNoduriIncarcate is Multe) then (StareNod is Receptor)
- R6. If (IndiceIncarcareNod is mare) and (NrNoduriIncarcate is Putine) then (StareNod is Expeditor)
- R7. If (IndiceIncarcareNod is mare) and (NrNoduriIncarcate is Multe) then (StareNod is Receptor)
- R8. If (IndiceIncarcareNod is critica(max)) then (StareNod is Expeditor)

În acest punct putem ști cu certitudine gradul de încărcare și eticheta fiecărui nod în parte, raportată la indicele de încărcare globală a sistemului.

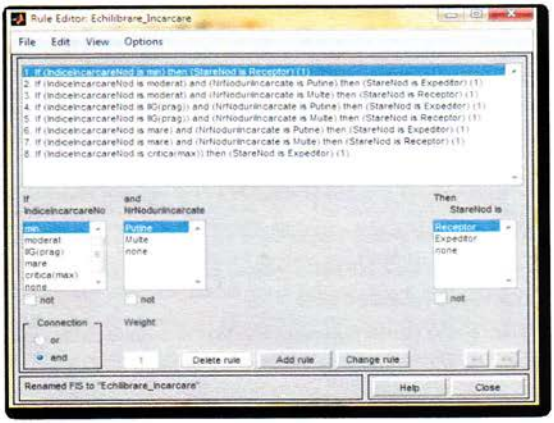

Figura 5.8. Editare reguli din baza de cunoștințe

În cazul în care sistemul prezintă un IIG mare, implicit inexistența unui partener de migrare, trebuie să existe o modalitate de semnalizare a stării de colaps a sistemului astfel încât procesele să rămână într-o coadă de așteptare până în momentul apariției unui partener de migrare. De asemenea, sistemul trebuie să fie perfectibil dacă constatăm că prezintă defecte de raționament în sistemul de reguli care implicit conduc la decizii eronate în legătură cu supraîncărcarea sistemului.

Bazându-se pe diagrama de inferență fuzzy, interfața grafică de vizualizare a regulilor permite interpretarea întregului proces de inferență fuzzy indicând modul în care forma anumitor funcții de apartenență influențează rezultatul general. Vizualizarea regulilor de parcursul întregului proces de inferență fuzzy este redată în Figura 5.9.

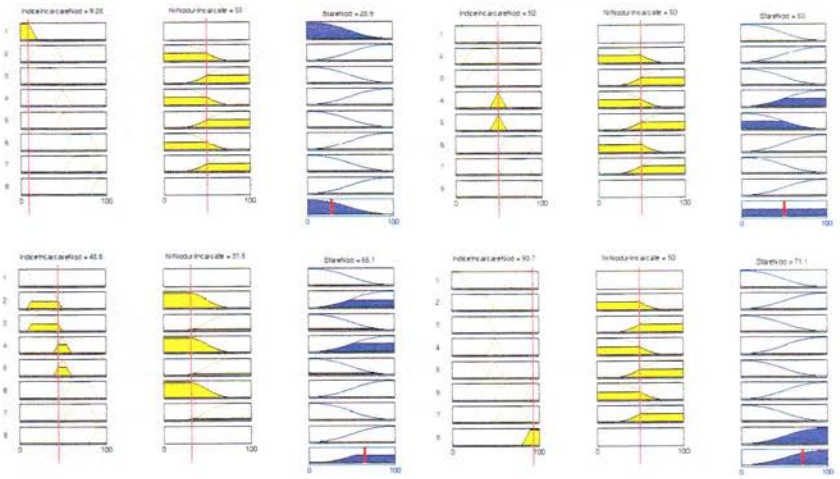

Figura 5.9. Interfața vizualizare reguli

Interfața de vizualizare a regulilor prezintă în detaliu un singur calcul la un moment dat prezentând o vizualizare la nivel micro a sistemului de inferență fuzzy. Pentru obținerea unei mărimi ferme de ieșire va fi prin urmare necesară operația de defuzzyficare.

Defuzzyficarea este ultima dintre problemele principiale ridicate de controlerul fuzzy, prin care mulțimea fuzzy obținută prin inferență este transformată într-o mărime fermă.

Defuzzyficarea se referă la modul în care se asociază unei mulțimi fuzzy o mărime exactă, reprezentativă pentru acea mulțime fuzzy. Sunt detaliate cele cinci modalități de defuzzyficare a unei mulțimi fuzzy definită pe un univers de discurs și având o funcție de apartenență precum și criteriile care stau la baza alegerii metodei folosite.

În modelul propus am utilizat pentru defuzzyficare metoda centrelor de greutate. Metoda centrelor de greutate COG (engl. *Center of Gravity*) este metoda de defuzzyficare cea mai sensibilă, care ține seama, într-o manieră ponderată, de influența fiecărui termen lingvistic al ieșirii, considerând toate valorile posibile pentru gradele de apartenență. Prin COG domeniul valorilor discrete de ieșire devine continuu.

5.1.5. Validare model

Pentru simulare am utilizat software-ul Matlab, oferit de compania MathWorks. În cadrul simulării am utilizat un set de valori pentru IIN și NNI, care reprezintă datele de intrare, generate aleatoriu printr-o distribuție discretă uniformă în intervalul [0, 100], utilizând funcția RANDI.

```
input = randi([0,100], 1000,2);
```

Încărcarea din fișier a sistemului de inferență fuzzy și simularea utilizând datele de intrare cu scopul obținerii variabilelor de ieșire în Matlab s-a făcut utilizând comenzile:

```
fis = readfis('Echilibrare_Incarcare.fis');
y=evalfis(input,fis);
```

Astfel, în simularea noastră funcție de setul de valori de intrare am obținut valoarea corespunzătoare variabilei de ieșire, după cum se observă în Figura 5.10.

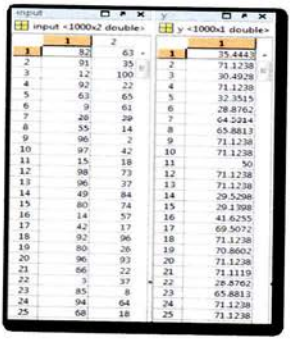

Figura 5.10. Datele simulării

Dispunând de întregul set de date cunoscut (date de intrare și ieșire) putem vizualiza valoarea prezisă/calculată de model. Astfel, valoarea prezisă de model a suprafeței redată de datele de ieșire ale sistemului (pe durata întregului set de intrare) este redată de Figura 5.11.

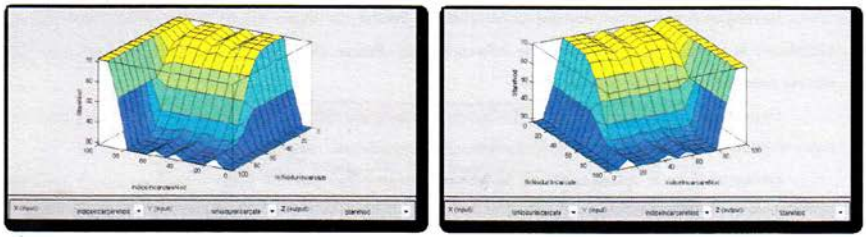

Figura 5.11. Afișare suprafața funcție de setul de date disponibil

5.1.6. Contribuții ale modelului

Contribuția adusă de modelul fuzzy permite modelarea interacțiunii dintre resursele fizice sau virtuale disponibile datorită echilibrării încărcării lor, conducând la o mai bună gestionare a cererilor clienților și implicit la o creștere a randamentului sistemului IT concomitent cu creșterea fiabilității lui.

Acest model, bazat pe eficiență conferă un acces mai facil la resursele neocupate nu doar pentru un procent din clienți ci pentru toți, semnalizând totodată gradul de încărcare și disponibilitatea resurselor. De asemenea, clienții conectați nu întâmpină dificultăți în accesarea serviciilor dorite și asta datorită minimizării timpilor de așteptare blocajelor sistemului deoarece cererile lor sunt direcționate către servere cu grad mic de încărcare.

Raportată la întregul sistem, aceasta abordare este definitorie în stabilirea gradului de încărcare a resurselor existente disponibile, stabilind cu exactitate, funcție de ponderea pe care o are resursă, dacă sistemul funcționează în parametri optimi. Din punct de vedere economic metoda aduce mari beneficii concretizate printr-o diminuare a timpilor de răspuns, a capitalului investit și a consumului de energie, ținând cont că toate acestea implică un înalt grad de utilizare a resurselor.

5.2. Metodă matematică de alocare dinamică și apreciere a stării funcționale a unui sistem, funcție de încărcarea resurselor disponibile

În a doua parte a acestui capitol propunem o nouă modalitate de alocare dinamică, notificare și apreciere a stării funcționale a unui sistem, prin modelarea stării comportamentale a sistemului care dispune de „m" resurse, ce pot fi monitorizate în orice moment, și care în funcție de procentul lor de încărcare pot conduce către un anumit comportament al stării de funcționare. Această stare de funcționare a unui sistem se poate încadra sau nu în anumiți parametrii optimi de funcționare definiți, de aceea, în continuare vom demonstra utilitatea metodei pentru expertul/diagnosticianul uman.

Prin această modelare matematică (dezvoltată în colaborare cu conf.dr. Ioan Cătălin Angelo) propunem, pe lângă o modalitate de atenționare și notificare a stării funcționale, o modalitate de alocare dinamică a unei resurse suplimentare disponibile în cazul în care sistemul intră în colaps, ținând cont de experiența anterioară a sistemului.

Stabilim în cadrul modelului ponderea pe care o are o resursă R_j, datorită faptului că același grad de încărcare a diferitelor resurse conduce la o notare diferită a stării de funcționare. Cu alte cuvinte, dacă luăm pe rând suplimentarea cu 1 a fiecărei resurse R_j monitorizabilă, j= $\overline{1,m}$ din cadrul sistemului la un anumit moment t_i, în urma retestării funcționalității sistemului la momentul t_{i+1} acest lucru ne va conduce la „m" notări diferite ale stărilor de funcționare ale sistemului în momentul „i". Astfel, ponderea cea mai mare în aprecierea gradului de funcționalitate a sistemului o are resursa care a condus la cea mai mare creștere pe scara funcționalității, funcție de sistemul de apreciere. Implicit, în acest caz, stabilim ponderea pe are o are fiecare tip de resursă din cadrul sistemului. În funcție de aceste „m" rezultate obținute, vom suplimenta doar acea resursă care a condus la o notare superioară a stării funcționale a sistemului. De asemenea, este posibil ca numărul absolut de resurse R_j utilizat să fie maxim și să nu mai dispunem de încă o unitate disponibilă a acelei resurse. Atunci nu mai este posibilă o suplimentare și sistemul intră în starea de colaps, lucru care este semnalizat prin afișarea numărului corespunzător de unități cu care trebuie suplimentat sistemul. Ca o consecință a acestui fapt, va trebui să suplimentăm hardware acea resursă.

În concluzie, indiferent dacă suplimentarea se face din resursele disponibile în urma abstractizării celor fizice, fie prin adăugarea de resurse hardware, sistemul va fi funcțional și va semnaliza permanent starea sa operațională.

5.2.1. Ipoteze de lucru

Să considerăm un sistem complex ce dispune de „m" resurse pe care le notăm „R_j, j= $\overline{1,m}$", resurse care în timpul funcționării sistemului pot avea diferite grade de încărcare și care pot fi supuse monitorizării. Încărcarea procentuală a unei resurse devine:

$$\textit{încărcare procentuală} = \frac{\text{încărcare resursă}}{100};$$

încărcare resursă reprezintă nivelul procentual de încărcare în intervalul [0,100].

Vom considera o durată de monitorizare pe întreaga perioadă de funcționare a sistemului egală cu T și un interval de timp [0,T] în care se fac operațiile de monitorizare pentru a nu încărca resursele. Împărțim această perioadă de monitorizare în „n" părți egale în scopul monitorizării sistemului la intervale de timp regulate și predeterminate.

$t_0<t_1<...<t_n=T$; unde $t_0=0$, $t_i - t_{i-1} = \frac{T}{n}$, i= $\overline{1,n}$.

Vom considera, de asemenea, că fiecare resursă R_j se poate multiplica, potențial, în N_j exemplare. De exemplu, N_j reprezinta numarul maxim de resurse virtulale pe care le poate avea un sistem hardware abstractizat. Astfel, daca dispunem de 8 procesoare pe o masina fizica, numarul maxim de procesoare disponibile în urma abstractizării este multiplu de 8, adica de 8 ori câte masini virtuale sunt create pe mașina hardware respectivă. De asemenea, pot exista mai multe procesoare care au proprietatea că în momentul încărcării maxime a unuia va conduce la intrarea în execuție a celui de-al doilea sau în cazul în care memoria RAM disponibilă funcționează la o încărcare maximă, acest lucru poate conduce la o atenționare a situației create, implicit la o suplimentare a memoriei atât fizic prin inserarea unei resurse hardware suplimentare de memorie sau printr-o alocare suplimentară din memoria virtuală pe care o are sistemul la dispoziție.

Vom nota deci resursele sub forma: R_j, $j=\overline{1,m}$ și vom atribui lui R_j la momentul „t" un grad de importanță notat $q_j(t)>0$ ce va semnifica importanța (ponderea) acestei resurse în cadrul funcționării sistemului la momentul „t". Vom presupune acest lucru din motive tehnice evidente. Normalizând aceşti coeficienți de importanță, ponderea resursei R_j în cadrul sistemului la momentul t va fi:

$$P_j(t)=\frac{q_j(t)}{\sum_{k=1}^{m} q_k(t)},\ j=\overline{1,m}.$$

având evident faptul că suma acestor indicatori este 1, conform relației : $\sum_{k=1}^{m} p_k(t)=1$.

Vom considera la momentul t că există în funcțiune $k_j(t)\in N$ (unde, evident, $1\le k_j(t)\le N_j$) resurse de tipul R_j, $j=\overline{1,m}$. Nivelul de utilizare al resurselor este dat de variabilele: $x_j\in[0,k_j(t)]$, $j=\overline{1,m}$ ce semnifică gradul de încărcare a resurselor corespunzătoare.

Pentru formalizare, introducem matricea $U(t)=(v_{ji}(t))$, $i=\overline{1,N_j}$, $j=\overline{1,m}$ unde:

$$v_{ji}(t)=\begin{cases}1, \text{dacă resursa } R_{ji} \text{ este activă la momentul t}\\ 0, \text{dacă resursa } R_{ji} \text{ nu este activă la momentul t}\end{cases}$$

unde R_{ji} reprezintă exemplarul „i" a resursei R_j.

Considerăm o partiție a intervalului de notare [0,1] ce va permite cuantificarea stărilor sistemului,de forma:

$I_1=[\alpha_0,\alpha_1)$, $I_2=[\alpha_1,\alpha_2)$, $I_p=[\alpha_{p-1},\alpha_p)$,...,$I_r=[\alpha_{r-1},\alpha_r]$ unde $\alpha_0=0$ și $\alpha_r=1$

unde r este numărul de intervale din cuantificarea stărilor sistemului iar α_p reprezintă un set de valori care vor fi determinate ulterior (secțiunea 5.2.3).

Vom considera acum o modalitate de cuantificare a bunei funcționări a sistemului prin atribuirea unui număr real din intervalul [0,1] a cărui semnificație va consta în faptul că o valoare apropiată de 1 va conduce la o bună funcționare a sistemului, iar una apropiată de 0 la o comportare defectuoasă a acestuia.

Fie deci funcția

$$f:\{0,1,\dots,m\}\times[0,k_1(t_i)]\times\dots\times[0,k_m(t_i)]\to[0,1],\ (i,x_1,\dots,x_m)\to f(i,x_1,\dots,x_m)\in[0,1]$$

funcția de atribuire a „notei procentuale" sistemului la momentul de monitorizare t_i, $i=\overline{0,m}$ atunci când sunt active $k_1(t_i)$ resurse R_1,..., $k_m(t_i)$ resurse R_m.

5.2.2. Considerații matematice de geometrie a hiperplanelor

Fie în spațiul afin m-dimensional $\mathbf{R}^m$, având coordonatele $x_1,\dots,x_m$, ecuația hiperplanului (analogul planului din spațiu sau al dreptei din plan): $H: a_1x_1+\dots+a_mx_m-\beta=0$

Hiperplanul H împarte spaţiul $\mathbf{R}^m$ în două regiuni deschise (*numite semispaţii deschise*) caracterizate prin:

$$H_1=\{(x_1,...,x_m)\in\mathbf{R}^m \mid a_1x_1+...+a_mx_m-\beta<0\} \text{ şi } H_2=\{(x_1,...,x_m)\in\mathbf{R}^m \mid a_1x_1+...+a_mx_m-\beta>0\}$$

sau în:

$$H_1=\{(x_1,...,x_m)\in\mathbf{R}^m \mid a_1x_1+...+a_mx_m-\beta\le0\} \text{ şi } H_2=\{(x_1,...,x_m)\in\mathbf{R}^m \mid a_1x_1+...+a_mx_m-\beta\ge0\}$$

numite semispaţii închise ale căror intersecţie este hiperplanul H.

Determinarea uneia sau alteia dintre regiuni se face considerând un punct (de regulă originea, iar dacă hiperplanul nu trece prin ea, atunci $\beta\neq0$) al unei regiuni arbitrare şi stabilirea semnului expresiei: $a_1x_1+...+a_mx_m-\beta$. În acest caz, toate punctele aflate de aceeaşi parte cu cel considerat vor satisface aceeaşi inegalitate, semispaţiul opus satisfăcând inegalitatea contrară. În cazul în care $\beta>0$ este evident că semispaţiul ce va conţine originea va avea drept inecuaţie: $a_1x_1+...+a_mx_m-\beta<0$, iar cel opus: $a_1x_1+...+a_mx_m-\beta>0$.

Să considerăm acum funcţia signum (abr. sign), definită prin:

$$\text{sign}(x)=\begin{cases}1 \text{ dacă } x>0;\\ 0 \text{ dacă } x=0;\\ -1 \text{ dacă } x<0\end{cases}$$

Funcţia pe care o vom nota (de la semn al produsului negativ) snsign, definită prin:

$$\text{snsign}(x,y)=\text{sign}(1-\text{sign}(x)\cdot\text{sign}(y))=\begin{cases}0 \text{ dacă } xy>0;\\ 1 \text{ dacă } x=0 \text{ sau } y=0;\\ 1 \text{ dacă } xy<0\end{cases}=\begin{cases}0 \text{ dacă } xy>0;\\ 1 \text{ dacă } xy\le0;\end{cases}$$

alocă punctelor (x,y) din cadranele II şi IV valoarea 1.

Analog, funcţia pe care o vom nota (de la semn al produsului pozitiv) spsign, definită prin:

$$\text{spsign}(x,y)=\text{sign}(1+\text{sign}(x)\cdot\text{sign}(y))=\begin{cases}1 \text{ dacă } xy>0;\\ 1 \text{ dacă } x=0 \text{ sau } y=0;\\ 0 \text{ dacă } xy<0\end{cases}=\begin{cases}1 \text{ dacă } xy\ge0;\\ 0 \text{ dacă } xy<0\end{cases}$$

alocă punctelor (x,y) din cadranele I şi III valoarea 1.

Considerând un semispaţiu închis $H_1=\{(x_1,...,x_m)\in\mathbf{R}^m \mid a_1x_1+...+a_mx_m-\beta\le0\}$ vom putea să-l caracterizăm, cu ajutorul funcţiei snsign, prin: $\text{snsign}(a_1x_1+...+a_mx_m-\beta,\ 1)=1$ şi analog, pentru $H_2=\{(x_1,...,x_m)\in\mathbf{R}^m \mid a_1x_1+...+a_mx_m-\beta\ge0\}$ avem: $\text{spsign}(a_1x_1+...+a_mx_m-\beta,\ 1)=1$.

De asemenea, să remarcăm că distanţa de la origine la un hiperplan: H: $a_1x_1+...+a_mx_m-\beta=0$ este:

$$\text{dist}(0,H)=\frac{|\beta|}{\sqrt{a_1^2+...+a_m^2}}$$

5.2.3. Modelul matematic

Să considerăm acum paralelipipedul m-dimensional (conform noţiunii Coxeter) la momentul t:

$$[0,k_1(t)]\times[0,k_2(t)]\times...\times[0,k_m(t)]=\{(x_1,...,x_m) \mid x_1\in[0,k_1(t)],...,x_m\in[0,k_m(t)]\}\subset\mathbf{R}^m$$

Considerăm diagonala principală a paralelipipedului:

$$\frac{x_1}{k_1(t)}=...=\frac{x_m}{k_m(t)}$$

Sau în termeni parametrici: $\begin{cases}x_1=k_1(t)\lambda\\ \quad ...\\ x_m=k_m(t)\lambda\end{cases},\lambda\in\mathbf{R}$

Vom divide segmentul [0,A], unde 0 reprezintă punctul de origine şi $A(k_1(t),...,k_m(t))$ este colţul diagonal al paralelipipedului, în „r" subintervale $I_1,...,I_r$, corespunzătoare $I_1=[\alpha_0,\alpha_1),...,I_p=[\alpha_{p-1},\alpha_p),...,I_r=[\alpha_{r-1},\alpha_r]$.

În reprezentarea noastră, intervalul de notare I_k, $k=\overline{1,r}$ reprezintă diagonala paralelipipedului, împărţită în intervale inegale, funcţie de modul de reprezentare a stării funcţionale a sistemului. Acest lucru ne arată că un anumit interval din gradul de încărcare procentuală al unei resurse corespunde unei anumite stări funcţionale.

Deoarece $OA=\sqrt{k_1^2(t)+...+k_m^2(t)}$ avem $J_p=[B_p,C_p)$ unde:

$$B_p\left(\frac{k_1(t)}{\sqrt{k_1^2(t)+...+k_m^2(t)}}\alpha_{p-1},...,\frac{k_m(t)}{\sqrt{k_1^2(t)+...+k_m^2(t)}}\alpha_{p-1}\right)$$

$$C_p\left(\frac{k_1(t)}{\sqrt{k_1^2(t)+...+k_m^2(t)}}\alpha_{p},...,\frac{k_m(t)}{\sqrt{k_1^2(t)+...+k_m^2(t)}}\alpha_{p}\right)$$

Prin urmare $x\in[B_p,C_p)$ de unde rezultă că:

$$x=\left(\frac{k_1(t)}{\sqrt{k_1^2(t)+...+k_m^2(t)}}\left((1-\lambda)\alpha_{p-1}+\lambda\alpha_p\right),...,\frac{k_m(t)}{\sqrt{k_1^2(t)+...+k_m^2(t)}}\left((1-\lambda)\alpha_{p-1}+\lambda\alpha_p\right)\right),\ \lambda\in[0,1)$$

Deoarece fiecare resursă R_j are normalizată ponderea (nivel de importanţă) $p_j(t),j=\overline{1,m}$, la momentul t, este firesc să adopte un efect combinat $p_j(t)x_j$, unde x_j reprezintă nivelul de utilizare al resursei respective.

Să considerăm aşadar hiperplanul: $p_1(t)x_1+...+p_m(t)x_m-\beta=0$.

Condiţia ca hiperplanul să treacă prin C_p satisface:

$$p_1(t)\frac{k_1(t)}{\sqrt{k_1^2(t)+...+k_m^2(t)}}\alpha_p+...+p_m(t)\frac{k_m(t)}{\sqrt{k_1^2(t)+...+k_m^2(t)}}\alpha_p-\beta=0$$

Prin urmare:

$$\beta=\frac{p_1(t)k_1(t)+...+p_m(t)k_m(t)}{\sqrt{k_1^2(t)+...+k_m^2(t)}}\alpha_p$$

Ca o concluzie, la momentul de monitorizare t_i, $i=\overline{0,n}$ vom considera hiperplanele G_{ij}, $i=\overline{0,n}$, $j=\overline{0,r}$: $G_{ij}: p_1(t_i)x_1+...+p_m(t_i)x_m-\frac{p_1(t_i)k_1(t_i)+...+p_m(t_i)k_m(t_i)}{\sqrt{k_1^2(t_i)+...+k_m^2(t_i)}}\alpha_j=0$

unde $p_1(t_i)x_1$ reprezintă ponderea pe care o are resursa x_1 la momentul t_i

Fie regiunea S_{ij} cuprinsă între semispaţiile G_{ij} şi G_{ij+1} la momentul t_i şi delimitată de feţele m-paralelipipedului. Aceasta va fi caracterizată de condiţiile:

$$S_{ij}:\begin{cases}p_1(t_i)x_1+...+p_m(t_i)x_m-\frac{p_1(t_i)k_1(t_i)+...+p_m(t_i)k_m(t_i)}{\sqrt{k_1^2(t_i)+...+k_m^2(t_i)}}\alpha_j>0\\ p_1(t_i)x_1+...+p_m(t_i)x_m-\frac{p_1(t_i)k_1(t_i)+...+p_m(t_i)k_m(t_i)}{\sqrt{k_1^2(t_i)+...+k_m^2(t_i)}}\alpha_{j+1}\le 0\end{cases}, i=\overline{0,n},\ j=\overline{0,r-1}$$

sau altfel, din cele de mai sus definim regiunea prin ecuația:

$$S_{ij}: snsign\left(\sum_{v=1}^{m} p_v(t_i)x_v - \frac{\sum_{v=1}^{m} p_v(t_i)k_v(t_i)}{\sqrt{\sum_{v=1}^{m} k_v^2(t_i)}}\alpha_j \cdot \sum_{v=1}^{m} p_v(t_i)x_v - \frac{\sum_{v=1}^{m} p_v(t_i)k_v(t_i)}{\sqrt{\sum_{v=1}^{m} k_v^2(t_i)}}\alpha_{j+1}\right)=1$$

Dacă un punct $x=(x_1,....x_m)\in S_{ij}$ atunci $x\notin S_{ik}\ \forall k\neq j$ deoarece hiperplanele nu se intersectează în interiorul m-paralelipipedului (intervalele I_p, $p=\overline{1,r}$ fiind disjuncte două câte două) fiind paralele.

În acest caz, dacă avem:

$$\begin{cases} p_1(t_i)x_1 + ... + p_m(t_i)x_m - \dfrac{p_1(t_i)k_1(t_i) + ... + p_m(t_i)k_m(t_i)}{\sqrt{k_1^2(t_i) + ... + k_m^2(t_i)}}\alpha_j > 0 \\ p_1(t_i)x_1 + ... + p_m(t_i)x_m - \dfrac{p_1(t_i)k_1(t_i) + ... + p_m(t_i)k_m(t_i)}{\sqrt{k_1^2(t_i) + ... + k_m^2(t_i)}}\alpha_{j+1} > 0 \end{cases}$$

sau:

$$\begin{cases} p_1(t_i)x_1 + ... + p_m(t_i)x_m - \dfrac{p_1(t_i)k_1(t_i) + ... + p_m(t_i)k_m(t_i)}{\sqrt{k_1^2(t_i) + ... + k_m^2(t_i)}}\alpha_j < 0 \\ p_1(t_i)x_1 + ... + p_m(t_i)x_m - \dfrac{p_1(t_i)k_1(t_i) + ... + p_m(t_i)k_m(t_i)}{\sqrt{k_1^2(t_i) + ... + k_m^2(t_i)}}\alpha_{j+1} < 0 \end{cases}$$

obținem:

$$snsign\left(\sum_{v=1}^{m} p_v(t_i)x_v - \frac{\sum_{v=1}^{m} p_v(t_i)k_v(t_i)}{\sqrt{\sum_{v=1}^{m} k_v^2(t_i)}}\alpha_j \cdot \sum_{v=1}^{m} p_v(t_i)x_v - \frac{\sum_{v=1}^{m} p_v(t_i)k_v(t_i)}{\sqrt{\sum_{v=1}^{m} k_v^2(t_i)}}\alpha_{j+1}\right)=0$$

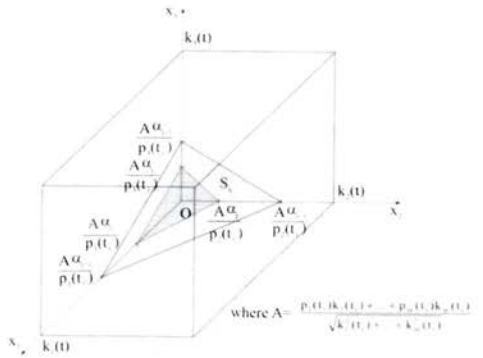

Figura 5.12. Reprezentarea sistemului pentru trei resurse la un moment de timp t_i (Postolache et al., 2010)

Totodată, modelul permite o realocare dinamică a hiperplanelor în funcție de încărcarea sistemului cu scopul rezolvării problemei într-un mod automat, în fiecare moment, ținând cont de experiența anterioară a sistemului.

Modelul matematic permite dezvoltări ulterioare, în primul rând datorită abordării cantitative unde resursele se realocă în mod automat pentru a asigura o distribuire uniformă a încărcării. Astfel, modelul nu se limitează doar

la a semnaliza o stare de nefuncţionalitate ci rezolvă întreaga problematică a echilibrării încărcării resurselor din cadrul sistemului. Modelul permite astfel înlăturarea automată a stării de nefuncţionalitate, învăţând din experienţa anterioară, lucru ce asigură o perfecţionare continuă a modalităţii de alocare a resurselor.

Definim deci funcţia: $f:\{0,1,....,m\}\times[0,k_1(t_i)]\times...\times[0,k_m(t_i)]\to[0,1]$, $(i,x_1,....,x_m)\to f(i,x_1,....,x_m)=$

$$(r+1)-\sum_{j=0}^{r-1}(j+1)\,\mathrm{snsign}\left(\sum_{v=1}^{m}p_v(t_i)x_v-\frac{\sum_{v=1}^{m}p_v(t_i)k_v(t_i)}{\sqrt{\sum_{v=1}^{m}k_v^2(t_i)}}\alpha_j,\sum_{v=1}^{m}p_v(t_i)x_v-\frac{\sum_{v=1}^{m}p_v(t_i)k_v(t_i)}{\sqrt{\sum_{v=1}^{m}k_v^2(t_i)}}\alpha_{j+1}\right)$$

ce va lua următoarele valori la momentul t_i:

$$f(i,x_1,....,x_m)=\begin{cases} r \text{ dacă } (x_1,x_2,....,x_m)\in S_{i0};\\ r-1 \text{ dacă } (x_1,x_2,....,x_m)\in S_{i1};\\ \quad ...\\ 1 \text{ dacă } (x_1,x_2,....,x_m)\in S_{ir-1}\end{cases}$$

Să considerăm acum ansamblul de variabile aleatoare:

$$Z_{ij}=\begin{pmatrix}1 & 2 & ... & r\\ C_1 & C_2 & ... & C_r\end{pmatrix},\ i=\overline{1,n}\ ,\ j=\overline{1,m}$$

unde 1, 2, ..., r reprezintă modalitatea de notare a funcţionalităţii sistemului iar C_1, C_2, ..., C_r reprezintă media încărcărilor (cât de încărcată) unei componente pentru a ajunge la notarea 1, 2, ..., r.

Astfel de la momentul de start „j" până la momentul „i", considerăm suma stărilor încărcării reperului respectiv la momentul „i" pentru care ia nota k.

$$C_k=\frac{\sum_{j=0}^{i}\sum_{f(j,x_1,....,x_m)=k}x_j}{card\{x_j\,|f(i,x_1,....,x_m)=k\}}$$

adică media aritmetică a valorilor procentuale ale resursei R_j ce au condus la o stare a sistemului ce a obţinut „nota" k, determinată de la începutul funcţionării procesului până la momentul t_i, unde $card\{x_j\,|f(i,x_1,....,x_m)=k\}$ reprezintă cardinalul , şi este numărul de elemente ale unei mulţimi.

Media M_{ij} a variabilei aleatoare Z_{ij}, $M_{ij}=\sum_{k=1}^{r}k\cdot C_k$ va reprezenta „nota" medie pe care o are resursa R_j în cadrul contribuţiei ei la funcţionarea sistemului.

Vom considera deci la momentul t_{i+1} ponderea resursei R_j în cadrul sistemului ca fiind:

$$p_j(t_{i+1})=\frac{M_{ij}}{\sum_{k=1}^{m}M_{ik}}$$

adică acea „cotă" ce îi revine în cadrul „notei" generale a sistemului. Dacă $f(i,x_1,....,x_m)=1$ acest lucru înseamnă că sistemul tinde să intre în colaps. În acest caz, vom suplimenta pe rând cu o unitate fizică (dacă este

posibil) fiecare resursă și vom retesta sistemul. Acea resursă care va obține cea mai mare „notă" va fi cea care va intra în funcțiune efectiv.

Fie deci resursa R_j ce va fi suplimentată cu $\min\{1, N_j - k_j(t_i)\}$ (adică dacă numărul absolut de resurse R_j este utilizat la maximum deci $Nj - k_j(t_i)=0$ atunci nu se mai poate face nicio suplimentare și sistemul rămâne în starea de colaps, în caz contrar suplimentarea fiind de o unitate.

Se calculează valoarea funcției de notare f de mai sus pentru toate aceste situații. În final se reține suplimentarea ce conduce la „nota" maximă.

Cu alte cuvinte, dacă luăm pe rând suplimentarea cu 1 a fiecărei resurse R_j, $j= \overline{1, m}$ din cadrul sistemului, acest lucru ne va conduce la „m" notări ale stărilor de funcționare. În consecință, în funcție de rezultatele obținute, vom suplimenta doar acea resursă care a condus la o notare superioară a stării funcționale a sistemului. Ca o consecință a acestui fapt, dacă nu mai dispun de o suplimentare cu încă o resursă, sistemul ne atenționează că intră în colaps și va trebui să suplimentăm hardware acea resursă. Astfel, indiferent că suplimentarea se face din resursele disponibile în urma abstractizării celor fizice, fie prin adăugare de hardware, sistemul meu va fi funcțional și va semnaliza permanent starea sa funcțională.

Observație

Pentru aplicarea algoritmului, propunem ca la momentul de start t_0 să considerăm gradele de importanță egale $q_j(t_0)=1 \ \forall j=\overline{1.m}$. Acești parametri se vor modifica dinamic pe parcursul funcționării sistemului.

În cazul nostru, indiferent de modul de reprezentare pe care dorim să îl avem în cazul funcționalității sistemului, în momentul apariției defectului, agentul observă dacă a apărut o diferență în modalitatea de notare a funcționalității și o asignează la rândul sau codului de eroare respectiv. Cu alte cuvinte, dacă respectiva eroare a dus la o scădere în modul de notare al gradului de funcționalitate, acesteia i se atașează o variabilă care indică gradul în care a fost afectată funcționalitatea sistemului.

5.2.4. Validare studiu de caz

Funcție de reprezentarea pe care dorim sa o dăm funcționalității sistemului privitor la încărcarea resurselor, avem mai multe posibilități. Ne vom opri asupra unui sistem de notate cât și asupra unui sistem de reprezentare cu ajutorul codului culorilor.

În cazul sistemului de notare se acordă o notă funcționalității sistemului de la 1 la 10, valoarea 1 corespunzând funcționări defectuoase iar 10 pentru o funcționare bună. Similar avem aprecierea funcționalității sistemului printr-un cod al culorilor. Pentru aceasta avem nevoie de valori aleatoare pentru variabilele analizate, în consecință vom proceda după cum urmează.

Modalitate de notificare 1

Dacă se dorește o evaluare a stării funcționale a sistemului prin metoda notării (acordarea unei note), trebuie sa ne decidem asupra sistemului de notare. Vom considera sistemul de notare în intervalul [1, 10], unde valoarea 1 corespunde stării de funcționare în afara parametrilor nominali(sau unei stări de funționalitate corespunzătoare stării de defect), iar valoarea 10 corespunde unei funcționări normale.

Rezultă că funcția de notare a stării funcționale a sistemului are următoarele valori:

$$f(i,x_1,\ldots,x_m)=\begin{cases}1, & functionare\ anormala\\ \ldots & \\ r, & functionare\ normala\end{cases}$$

În cazul acordării de note sistemului de la 1 la 10, avem:

$$r=10 \quad \Rightarrow f(i,x_1,\ldots,x_m) = \begin{cases} 1, & functionare\ anormala \\ \ldots & \\ 10, & functionare\ normala \end{cases}$$

Tabelul 5. 1 Simulare metodă de notare

Număr disponibil resursa R1=	2
Număr disponibil resursa R2=	1
Număr disponibil resursa R3=	2
Grad de modificare de la momentul t la t+1=	10%
Încărcare inițială resursa R1=	0.1
Încărcare inițială resursa R2=	0.2
Încărcare inițială resursa R3=	0.1

Nota	Total=	884
1	0	0%
2	0	0%
3	0	0%
4	0	0%
5	4	0%
6	223	25%
7	491	56%
8	162	18%
9	75	8%
10	29	3%

Resursa R1	Resursa R2	Resursa R3	Ponderea R1	Ponderea R2	Ponderea R3	Număr resurse R1 în funcțiune	Număr resurse R2 în funcțiune	Număr resurse R3 în funcțiune
10.00%	20.00%	10.00%	0.333333333	0.333333333	0.333333333	1	1	1
17.00%	27.00%	2.00%	0.037974684	0.398734177	0.563291139	1	1	1
25.00%	31.00%	11.00%	0.040378863	0.411843769	0.540877368	1	1	1
20.00%	22.00%	20.00%	0.042588043	0.436527437	0.520884521	1	1	1
27.00%	17.00%	17.00%	0.087648483	0.425095145	0.487256372	1	1	1
29.00%	20.00%	13.00%	0.087932247	0.442238811	0.469828943	1	1	1
36.00%	22.00%	19.00%	0.087407764	0.452924275	0.45966796	1	1	1
26.00%	23.00%	28.00%	0.100446818	0.45627847	0.443279713	1	1	1
32.00%	14.00%	20.00%	0.098749493	0.457399217	0.443851129	1	1	1
42.00%	5.00%	23.00%	0.098825765	0.464935104	0.436239131	1	1	1
34.00%	6.00%	18.00%	0.098827441	0.478034593	0.423137966	1	1	1
42.00%	16.00%	12.00%	0.099408326	0.485410255	0.415181419	1	1	1
44.00%	18.00%	14.00%	0.098538154	0.490787863	0.410673983	1	1	1
37.00%	9.00%	18.00%	0.097565554	0.494989578	0.407444868	1	1	1
40.00%	2.00%	14.00%	0.097772985	0.499394362	0.402832653	1	1	1
35.00%	7.00%	8.00%	0.098150899	0.505291195	0.396557906	1	1	1
39.00%	4.00%	3.00%	0.098479689	0.508482846	0.393037465	1	1	1
37.00%	10.00%	2.00%	0.098743827	0.5126711	0.388585074	1	1	1
30.00%	2.00%	1.00%	0.098779102	0.51485026	0.386370639	1	1	1
27.00%	3.00%	9.00%	0.099493311	0.517439695	0.383066993	1	1	1
18.00%	5.00%	16.00%	0.100212169	0.519121541	0.380666291	1	1	1
9.00%	2.00%	26.00%	0.101165069	0.519102743	0.379732187	1	1	1
12.00%	11.00%	29.00%	0.102575803	0.518391789	0.379032414	1	1	1
18.00%	5.00%	20.00%	0.103401861	0.516409964	0.380186176	1	1	1
22.00%	8.00%	11.00%	0.104207141	0.516402553	0.379390306	1	1	1
15.00%	18.00%	8.00%	0.104889088	0.516470168	0.378840744	1	1	1
18.00%	22.00%	16.00%	0.105048435	0.514137168	0.380814397	1	1	1
9.00%	21.00%	13.00%	0.105048435	0.514137168	0.380814397	1	1	1
14.00%	19.00%	12.00%	0.105516715	0.510686542	0.383796743	1	1	1
14.00%	11.00%	10.00%	0.105834324	0.508279441	0.385886236	1	1	1
7.00%	19.00%	15.00%	0.106451329	0.507207129	0.386341543	1	1	1
1.00%	20.00%	24.00%	0.107021981	0.50413502	0.388842999	1	1	1
2.00%	17.00%	26.00%	0.095466111	0.506894488	0.397639401	1	1	1
0.00%	8.00%	33.00%	0.096006091	0.503682351	0.400311557	1	1	1
3.00%	1.00%	28.00%	0.09689031	0.501508597	0.401601093	1	1	1
6.00%	7.00%	31.00%	0.097921283	0.500764455	0.401314262	1	1	1
12.00%	14.00%	26.00%	0.098629803	0.49950021	0.401869987	1	1	1
11.00%	17.00%	16.00%	0.098900571	0.498052471	0.403046958	1	1	1
8.00%	25.00%	18.00%	0.099115105	0.496239659	0.404645236	1	1	1
10.00%	29.00%	12.00%	0.093684451	0.496141443	0.410174106	1	1	1

Întrucât variabilele se modifică dinamic pe tot parcursul funcționării sistemului, inițial am considerat ponderile egale și o încărcare procentuală mică a resurselor (între 10 și 20%) în aprecierea stării funcționale a sistemului.

După cum se observă în Tabelul 5. 1, am atribuit valori variabilelor R_1, R_2 și R_3 pentru a simula comportarea parametrilor sistemului, ca apoi generarea valorilor reprezentând gradul de încărcare procentuală a resurselor sistemului să se producă aleator. De asemenea, am luat în calcul și numărul de unități identice de care poate dispune o resursă în cadrul sistemului. În cazul nostru dispunem de 2 resurse de tipul R_1 și R_3 și de o resursă pentru R_2.

În acest caz pentru aprecierea stării funcționale a sistemului dispunem de 10 intervale. Simularea s-a derulat pentru un număr de 1000 de triplete de valori ce reprezintă aleator gradele de încărcare ale resurselor, pentru a avea o mai bună distribuire a notării stării de funcționalitate.

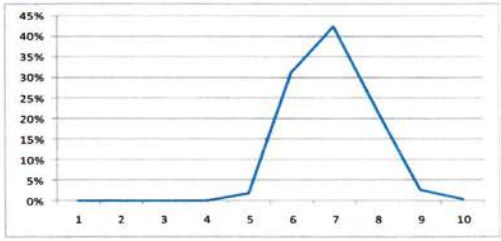

Figura 5.13. Distribuția funcționalității sistemului prin metoda de notare

După cum se observă în Figura 5.13, funcția are o reprezentare similară Legii distribuției normale (Gauss), cu o deplasare spre modul de funcționare bun. Din grafic se observă că sistemul a avut în peste 40% din cazuri o funcționare de nota 7. Pentru modalitatea de evaluare a stării funcționale prin acordarea de note, se observa o deplasare a graficului spre modul de funcționare bun, nota 1 reprezentând starea de colaps iar 10 o funcționare în parametri optimi a sistemului.

Modalitate de notificare 2.

Dacă se dorește o reprezentare a stării funcționale a sistemului printr-un cod al culorilor, trebuie sa ne decidem asupra numărului culorilor, care sunt și ce semnificație au acestea. Considerăm următoarele culori asignate valorilor: verde, galben, orange, roșu și negru, iar în reprezentarea stării funcționale verde corespunde stării funcționale normale iar negru stării anormale sau nefuncționale (colaps) a sistemului monitorizat.

Considerăm o reprezentare a stării funcționale prin cinci culori, rezultă ca r=5, astfel funcția noastră devine:

$$f(i,x_1,\ldots,x_m) = \begin{cases} 5, & verde \\ 4, & galben \\ 3, & orange \\ 2, & rosu \\ 1, & negru \end{cases}$$

Identic metodei de notare, în reprezentarea stării funcționale cu ajutorul culorilor inițial am considerat egale ponderile pe care le au resursele și un grad de încărcare procentuală mic deoarece aceste valori se vor modifica dinamic pe tot parcursul simulării.

Am atribuit valori variabilelor R_1, R_2 și R_3, luînd în calcul și numărul de unități identice de care dispune o resursă în cadrul sistemului (Tabelul 5. 2).

Tabelul 5. 2. Simulare metodă codul culorilor

Număr disponibil resurse R1=	2
Număr disponibil resurse R2=	1
Număr disponibil resurse R3=	2
Grad de modificare de la momentul t la t+1=	10%
Încărcare inițială resursa R1=	0.1
Încărcare inițială resursa R2=	0.2
Încărcare inițială resursa R3=	0.1

	Culoare	Total	Procent
1	VERDE	523	52%
2	GALBEN	183	18%
3	PORTOCALIU	49	5%
4	ROSU	58	6%
5	NEGRU	120	12%
6	resursa	69	7%
		1000	

Resursa R1	Resursa R2	Resursa R3	Pondere_R1	Pondere_R2	Pondere_R3	Nr resurse R1 in funcțiune	Nr resurse R2 in funcțiune	Nr resurse R3 in funcțiune
10.00%	20.00%	10.00%	0.3333333	0.3333333	0.3333333	1	1	1
17.00%	17.00%	12.00%	0.0722892	0.3975904	0.5301205	1	1	1
27.00%	8.00%	16.00%	0.0834192	0.4325438	0.4840371	1	1	1
26.00%	17.00%	12.00%	0.0958864	0.4714413	0.4326724	1	1	1
21.00%	25.00%	16.00%	0.0976801	0.4774115	0.4249084	1	1	1
25.00%	32.00%	25.00%	0.1764614	0.4300177	0.3935209	1	1	1
31.00%	24.00%	21.00%	0.1755528	0.4240863	0.400361	1	1	1
28.00%	20.00%	20.00%	0.1809395	0.4232909	0.3957696	1	1	1
24.00%	29.00%	28.00%	0.1821396	0.4240701	0.3937903	1	1	1
16.00%	25.00%	36.00%	0.1782565	0.4223809	0.3993626	1	1	1
10.00%	16.00%	32.00%	0.1727124	0.4195061	0.4077815	1	1	1
8.00%	18.00%	23.00%	0.1682316	0.4173038	0.4144646	1	1	1
3.00%	15.00%	19.00%	0.1639946	0.4155203	0.420485	1	1	1
2.00%	13.00%	9.00%	0.1661142	0.4107652	0.4231207	1	1	1
4.00%	5.00%	7.00%	0.1686903	0.4066886	0.4246211	1	1	1
2.00%	7.00%	4.00%	0.1722046	0.4050482	0.4227473	1	1	1
8.00%	1.00%	3.00%	0.1755413	0.4024838	0.4219749	1	1	1
2.00%	3.00%	4.00%	0.1790927	0.4032048	0.4177025	1	1	1
10.00%	0.00%	6.00%	0.1827939	0.401412	0.4157941	1	1	1
5.00%	4.00%	16.00%	0.1859365	0.4028278	0.4112357	1	1	1
3.00%	6.00%	18.00%	0.1887353	0.4014676	0.4097971	1	1	1
10.00%	6.00%	23.00%	0.1913459	0.3990855	0.4095686	1	1	1
1.00%	7.00%	26.00%	0.1929676	0.3988995	0.4081329	1	1	1
9.00%	0.00%	26.00%	0.1953675	0.3955735	0.409059	1	1	1
2.00%	4.00%	21.00%	0.1976953	0.3959529	0.4063518	1	1	1
10.00%	6.00%	25.00%	0.2003226	0.3936839	0.4059935	1	1	1
9.00%	11.00%	15.00%	0.2015472	0.3935748	0.404878	1	1	1
16.00%	16.00%	8.00%	0.2023346	0.3927862	0.4048792	1	1	1
23.00%	18.00%	12.00%	0.2017347	0.3936359	0.4046293	1	1	1
24.00%	8.00%	6.00%	0.2001272	0.3959731	0.4038997	1	1	1
31.00%	10.00%	0.00%	0.1997594	0.3997085	0.4005321	1	1	1
41.00%	11.00%	8.00%	0.1986097	0.4046898	0.3967004	1	1	1
47.00%	21.00%	2.00%	0.1965011	0.4109883	0.3925105	1	1	1
42.00%	27.00%	2.00%	0.206163	0.4108362	0.3830008	1	1	1
41.00%	35.00%	9.00%	0.2104259	0.4115158	0.3780583	1	1	1
36.00%	35.00%	5.00%	0.2119813	0.4120132	0.3760056	1	1	1
42.00%	25.00%	0.00%	0.2117028	0.4129255	0.3753717	1	1	1
37.00%	24.00%	1.00%	0.2134456	0.414269	0.3722855	1	1	1
39.00%	14.00%	2.00%	0.2138318	0.4156195	0.3705488	1	1	1
35.00%	19.00%	3.00%	0.2126248	0.4183816	0.3689936	1	1	1

Ca și în cazul metodei de notare, în aprecierea stării funcționale a sistemului, simularea s-a derulat pentru un număr de 1000 de triplete de valori ce reprezintă aleator diferite grade de încărcare ale resurselor R_1, R_2 și R_3. Având 5 intervale de apreciere, fiecare corespunzând unei culori și vizând o mai bună distribuire a notării stării de funcționalitate simularea a reliefat un aspect cu care până acum nu ne-am confruntat și anume semnalizarea unui necesar de resursă.

După cum se observă în Figura 5.14, funcția nu prezintă o reprezentare similară Legii distribuției normale (Gauss). Distribuția, în cazul în care alegem o notificare a stării funcționale cu ajutorul unui cod al culorilor, este diferită de cea a sistemului de notare (r=10). Totuși graficul prezintă o deplasare spre modul de funcționare bun, cu un maxim funcțional către starea corespunzătoare culorii galben (peste 35% din rezultatele simulării), din aprecierea stării funcționale.

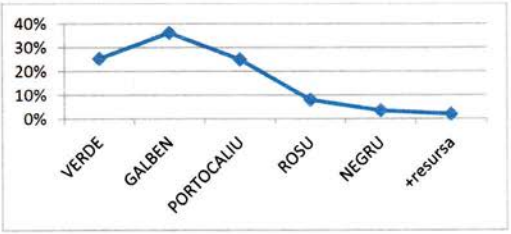

Figura 5.14. Distribuția funcționalității sistemului prin codul culorilor

Semnalizarea necesarului de resursă

Datorită posibilității realocării hiperplanelor într-un mod dinamic în funcție de încărcarea și numărul resurselor de care dispune sistemul, ținând cont de ponderea pe care o au resursele în momentul apariției stării de colaps, modelul indică într-un mod automat necesarul de resurse cu care trebuie suplimentat sistemul pentru rezolvarea problemei.

Modelul matematic, datorită abordării cantitative, realocă resursele într-un mod automat în vederea asigurării unei distribuții cât mai uniforme a încărcării lor, iar în cazul în care sunt insuficiente, pentru a preîntâmpina o viitoare stare de nefuncționalitate notifică necesarul de resursă cu care trebuie suplimentat sistemul (Figura 5.15).

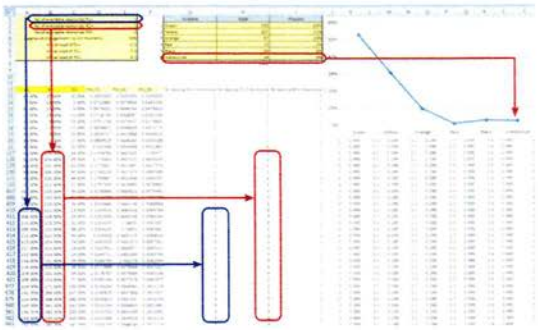

Figura 5. 15. Semnalizare suplimentare resurse

Din simulare rezultă clar ca sistemul tinde către o funcționalitate bună, unde resursele disponibile sunt alocate în mod corespunzător, modelul înlăturând automat starea de nefuncționalitate.

5.2.5. Contribuţii ale modelului

1. Modelarea interacţiunii dintre resursele fizice şi/sau virtuale disponibile.

Contribuţia adusă de modelul matematic se concretizează în modelarea interacţiunii dintre resursele fizice sau virtuale disponibile, în primul rând printr-o echilibrare a încărcării lor şi în al doilea rând permiţând o alocare dinamică a resurselor în cadrul sistemului. Această abordare conduce la o uniformizare a gradului de încărcare a resurselor eterogene distribuite în sistem, funcţie de caracteristicile de performanţă (tehnice) pe care acestea le au.

Totodată, modelul permite o realocare dinamică a hiperplanelor în funcţie de încărcarea sistemului cu scopul rezolvării problemei într-un mod automat, în fiecare moment, ţinând cont de experienţa anterioară a sistemului.

Modelul nu se limitează doar la a semnaliza o stare de nefuncţionalitate ci rezolvă întreaga problematică a echilibrării încărcării resurselor din cadrul sistemului, înlăturînd automat stărea de nefuncţionalitate, învăţând din experienţa anterioară. Modelarea permite astfel sistemului să se „auto-repare" prin înlăturarea situaţiei care conduce la o stare de nefuncţionalitate a sistemului printr-o re-alocare dinamică a resurselor disponibile în vederea echilibrării încărcării lor.

Supralicitarea, potrivit modelării ne permite o mai bună echilibrare a nevoilor pe care le poate avea o aplicaţie, la un moment dat, în utilizarea resurselor disponibile. Modalitatea de a suplimenta şi accesa cu încă o unitate o anumită resursă conduce la o deblocare a proceselor care au loc în momentul unei cereri suplimentare dar şi la o funcţionare în parametri normali ai sistemului. Acest lucru poate conduce la o alocare dinamică a resursei în funcţie de nevoile pe care le poate avea sistemul la un anumit moment dat.

2. Stabilirea unei ponderi de influenţă a unei „situaţii" în funcţionarea globală a sistemului

Modelul matematic permite o atenţionare a ponderii unei resurse în timpul funcţionării sistemului. Deoarece iniţial am considerat (la momentul t_0) ponderea celor trei parametri egală din cadrul sistemului, acest lucru se poate schimba pe parcursul evoluţiei stării funcţionale prin acordarea de ponderi diferite resurselor sistemului în funcţie de comportarea sistemului la momentele anterioare apariţiei evenimentului. Datorită modalităţii de notificare, funcţie de nota obţinută de sistem la momentul apariţiei defectului (t_1) şi nota din momentul anterior (t_0), rezultă ponderea pe care o are defectul în aprecierea/deprecierea stării funcţionale a sistemului..Astfel sistemul va avea posibilitatea de a învăţa din experienţa anterioară şi de a face o prognoza mai precisă a viitoarelor stări comportamentale.

3. Semnalizarea iminenţei stării de colaps a sistemului.

O astfel de situaţie poate să apară atunci când încărcarea procentuală a unei resurse depăşeşte limitele prestabilite şi nu au fost luate măsuri în vederea creşterii disponibilităţii resursei. Pentru modelul propus este determinantă modalitatea de semnalizare şi monitorizare a viitoarelor necesităţi din sistem, pentru ca acesta să funcţioneze în parametri optimi. Prin urmare, dacă un sistem semnalizează o posibilă stare de colaps, avem indicii sigure privind o anume nevoie de resurse ce trebuie asigurată, prevenind situaţia de nefuncţionalitate.

4. Receptarea soluţiei indicate pentru evitarea stării de colaps

În urma semnalizării stării de colaps, soluţia indicată de model ne spune viitoarele necesităţi pe care le are sistemul în vederea unei funcţionări în parametri optimi. Astfel, soluţia este inovatoare deoarece indică numărul de unităţi cu care trebuie suplimentată resursa R_j, în cazul în care aceasta este supraîncărcată cu diferite cereri ale sistemului.

6. CONCLUZII, CONTRIBUȚII ȘI DIRECȚII VIITOARE DE CERCETARE

6.1. Concluzii

În cadrul acestui capitol vom face o trecere în revistă a modului în care a fost structurat studiul abordat, concluziile, contribuțiile și viitoarele direcții de cercetare ce au decurs ca rezultat al studiului.

Astfel, în cadrul tezei, atenția este preponderent îndreptată către sistemele IT abstractizate în vederea asigurării unui tablou de ansamblu cât mai clar schițat în ceea ce privește funcționarea și utilizarea întregii capacități a sistemului virtualizat în parametrii optimi doriți. Capitolul *Abordări în diagnoza sistemelor complexe* tratează, gradual, terminologia folosită, noțiunile de sistem inteligent și complexitate, precum și modalitățile care conduc la virtualizarea infrastructurii, evident pentru înțelegerea cât mai corectă a complexității sistemelor. Am evidențiat cum această tehnologie, în special datorită ruperii tradiționalei legături dintre hardware și software, schimbă modul în care aceste sisteme funcționează, sunt gestionate, configurate și monitorizate prin realizarea unei imagini cât mai complete care să redea funcționalitatea și rezolvarea conflictelor.

Ținând cont de stadiul actual al cercetării și de modalitățile de abstractizare existente am propus spre analiză, dezbatere, testare și validare un nou model privind abstractizarea infrastructurii, derivat prin gruparea anumitor straturi ce deservesc rețelele, componentele hardware și software din cadrul unei infrastructuri abstractizate. Afirmăm că modelul este definitoriu, deoarece încă din momentul implementării, acesta a fost supus testării, monitorizării, verificării cât și validării performanțelor, lucru care a condus la nunțarea și confirmarea anumitor avantaje, evidențiind în schimb și riscurile la care acesta poate fi supus.

Virtualizarea având un impact asupra infrastructurii fizice, datorită necesarului de putere de calcul precum și asupra capacității rețelei de a satisface cererile de transfer dintre mașinile virtuale si cele fizice, conduce în final la avantaje semnificative privind migrația aplicațiilor, securitatea și fiabilitatea sistemelor.

Am argumentat și punctat importanța etapei de evaluare într-un mediu virtualizat a funcționalității și performanței aplicației în vederea asigurării planificării capacității și instrumentelor de monitorizare necesare unei funcționări în parametrii optimi a sistemului.

De asemenea, infrastructura IT abstractizată, structurată conform modelului propus ne-a permis o mai ușoară introducere în cunoașterea și înțelegerea domeniului virtualizării, a componentelor sale și a relațiilor dintre acestea, lucru care a condus în capitolul *Achiziția și structurarea cunoștințelor* la o redefinire a ontologiei domeniului vizat, pe care am considerat-o necesară și oportună la momentul actual pentru comunitatea specialiștilor.

Cu ce scop ontologia propusă?

Tratând noțiunile de baza și stadiul actual al cercetării privind achiziția de cunoștințe, analizând metodele utilizate, probleme privind reprezentarea și stocarea cunoștințelor precum și formalismul de reprezentare pentru achiziția de cunoștințe, vizăm popularea unei baze de cunoștințe cu acele cunoștințe necesare pentru o diagnoză a defectelor. Facem referire asupra sistemelor bazate pe cunoștințe, punctând ce reprezintă ele, care sunt caracteristicile lor, cum funcționează și cum este dezvoltat un sistem bazat pe cunoștințe dar și modul de utilizare a lor.

O atenție și o abordare deosebită am acordat-o achiziției automate de cunoștințe cu ajutorul agenților inteligenți în vederea creării unui fond comun de cunoștințe dar și modalității de reprezentare și structurare a lor.

Pentru aceasta am considerat necesară dezvoltarea unei ontologii proprii deoarece domeniul de investigație este relativ nou și din acest considerent avem nevoie de un set comun de constrângeri și de un cadru conceptual comun având ca scop principal relațiile dintre clasele de informație și modelarea cunoștințelor.

În definirea ontologiei domeniului virtualizării am folosit un software dedicat (*Tropes*) care ne-a pus la dispoziție un set de instrumente și tehnici pe care le-am utilizat la achiziția, analiza și modelarea cunoștințelor.

Metodologia corespunzătoare privind clasificarea sistemelor virtualizate s-a concretizat prin prisma straturilor ce compun modelul deoarece a permis captarea relațiilor dintre entitățile ce intră în componența stratului cât și relațiile dintre componentele diferitelor straturi/blocuri.

Însă, ce rol au toate acestea?

Cunoscând beneficiile virtualizării și vizând scopul de a pune bazele unui sistem instruibil care să sprijine deciziile diagnosticianul uman (SADU) în rezolvarea problemelor apărute, schimbarea modelului de la o orientare centrată pe defect la una bazată pe urme pentru a facilita sau amplifica capacitățile proprii ale diverșilor experți umani sau utilizatori a constituit punctul forte în diagnoza sistemelor virtualizate datorită modelării interacțiunilor dintre resursele fizice și cele virtuale.

Plecând de la conceptul de diagnoza, prin descrierea și analizarea celor mai utilizate tehnici,, în capitolul *Diagnoza defectelor* am evidențiat modalitatea de implementare a raționamentului bazat pe urme, firește, pornind de la raționamentul bazat pe cazuri. Vizând aplicabilitatea în sistemele complexe virtualizate, am sugerat o îmbinare a acestor două tehnici.

Contribuția majoră a SADU provine din faptul că sistemul de diagnoză este funcțional din primul moment, fără a necesita întreaga cazuistică inserată legată de contextul activității urmei, deoarece aceasta se completează pe parcursul desfășurării proceselor, iar în cazul apariției defectelor necunoscute, fără rezolvare imediată, problema este lăsată în discuție/analiză pentru rezolvare comunității specialiștilor din domeniu.

SADU permite unui utilizator să beneficieze de urmele provenite de la alți utilizatori. În acest caz, noul sistem de raționament lasă posibilitatea utilizatorului să aleagă modalitatea de perfecționare a măsurii, de a adopta alta, sau de continuare a ciclului cu alte tipuri de cunoștințe (de adaptare sau de context) dacă problema pare să aibă o rezolvare bună. De asemenea, el poate decide, dacă oprește procesul aici sau îl continuă.

Capitolul *Abordări aproximative privind alocarea și încărcarea resurselor pentru SADU* argumentează necesitatea utilizării unei noi abordări unde propunem, pe lângă o modalitate de atenționare și notificare a stării funcționale, o nouă modalitate de echilibrare și un procedeu de alocare dinamică a unei resurse suplimentare disponibile în cazul în care sistemul ar putea intra în colaps.

Această abordare privind reutilizarea resurselor disponibile prin echilibrarea încărcării conduce la o creștere a randamentului utilizării lor concomitent cu creșterea fiabilității sistemului și implicit la o mai bună gestionare a cererilor clienților.

Acest proces, bazat pe eficiență și echilibrarea utilizării resurselor disponibile conferă un acces mai facil la resursele neocupate nu doar pentru un procent din clienți ci pentru toți.

Această modalitate de abordare responsabilă a resurselor sistemului, a managementului resurselor și proceselor proiectate pentru nimic altceva decât pentru o funcționalitate în cel mai eficient și inteligent mod posibil poate fi definită ca o „funcționalitate bazată pe resurse".

Ca o consecință a elementelor de noutate aduse de teză au rezultat, în mod logic, contribuții specifice și noi direcții de cercetare în domeniu, sistematizate în cele ce urmează:

6.2. Contribuții

1. ***Propunerea unui nou model de abstractizare a infrastructurii***

   Conceput prin gruparea blocurilor ce deservesc componentele hardware/software și rețelele din cadrul infrastructurii în jurul conceptului de stivă funcție de apartenența sau tipul acestora (hardware, infrastructură virtuală, software) (secțiunea 2.3.1), modelul ne-a permis o mai ușoară introducere în cunoașterea și înțelegerea domeniului virtualizării, a componentelor sale și a relațiilor dintre acestea.

   Datorită grupării straturilor, abstractizarea spațiului de stocare și abstractizarea (consolidarea) serverelor a devenit mai ușor de configurat, monitorizat și administrat (secțiunea 2.3.2).

2. ***Propunerea unui nou model de abstractizare a mașinii hardware***

   Conceput prin prisma straturilor ce compun mașina hardware (secțiunea 3.4.3), acesta este definitoriu în stabilirea competențelor agenților inteligenți. Aici definim straturile ce alcătuiesc modelul: stratul fizic (P L), stratul de virtualizare (VL), stratul mașină virtuală (VM), stratul sistem de operare (SO) și stratul aplicații (*Apps*).

3. ***Stabilirea algoritmului care stă la baza funcționării agenților***

   Stabilirea algoritmului care stă la baza funcționării agentului (secțiunea 3.4) cu ajutorul căruia se face achiziția de cunoștințe s-a făcut prin prisma competenței, securității și încrederii acordată agenților. De asemenea, agenții au modul de lucru și comunicare bine stabilit și competențe funcție de rolul și stratul din cadrul infrastructurii pe care sunt plasați.

4. ***Crearea ontologiei proprii domeniului vizat***

   a) Modelul conceput prin gruparea straturilor ce alcătuiesc infrastructura abstractizată (secțiunea 2.3) a contribuit semnificativ la derularea proiectului de achiziție și structurare a cunoștințelor (secțiunea 3.2.1) conducând în final la dezvoltarea unei noi ontologii specifice domeniului vizat (secțiunea 3.6.1) cu scopul captării relațiilor dintre entitățile ce intră în componența stratului cât și relațiile dintre componentele diferitelor straturi (secțiunea 3.6.3).

   b) A permis conceperea unui set comun de constrângeri și a unui cadru conceptual comun având ca scop principal relațiile dintre clasele de informație și modelarea cunoștințelor în vederea obținerii ontologiei domeniului (secțiunea 3.6.4).

   c) Alegerea metodologiei, a contribuit la îmbunătățirea tehnicilor și instrumentelor pe care le-am utilizat la achiziția, analiza și modelarea cunoștințelor în vederea:

      - Partajării unui vocabular comun și a înțelegerii structurii informațiilor din cadrul comunității și al agenților software.
      - Reutilizării domeniului de cunoștințe pentru dezvoltări ulterioare.
      - Formulării într-un mod explicit a ipotezelor privind domeniul de interes.
      - Separării cunoștințelor de domeniu de cunoștințele operaționale.
      - Unei analize mai profunde a cunoștințelor domeniului de interes.

5. ***Structurarea domeniului virtualizării privind facilitarea achiziției de cunoștințe.***

   Prin structurarea domeniului obținem avantajul că fiecare agent (uman sau inteligent) specializat pe un anumit strat va identifica și insera în baza de cazuri strict acele cazuri unde acesta este plasat, permițând o mai bună localizare, un nivel de adâncime a discriminării componentelor cât mai precis implicit o

granularitate mică, diagnosticul în astfel de cazuri fiind în detaliu facilitând o intervenție completă în depanare (secțiunea 3.4.4).

Abordarea a vizat în primul rând utilitatea produsului pentru utilizatorii finali și calitatea cunoștințelor cuprinse (corecte, complete și relevante) depozitate într-un mod structurat.

De altfel, o structurare a domeniului permite partajarea bazei de cunoștințe cu alte sisteme de calcul sau utilizarea ei ca parte din dezvoltarea unui sistem inteligent computațional.

6. ***Dezvoltarea unui sistem de asistare a diagnosticianului uman***

a) Contribuția importantă adusă de SADU provine din faptul că sistemul de diagnoză este funcțional din primul moment, fără a necesita întreaga cazuistică inserată legată de contextul activității urmei, deoarece aceasta se completează pe parcursul desfășurării proceselor, iar în cazul apariției defectelor necunoscute, fără rezolvare imediată, problema este lăsată în discuție/analiză pentru rezolvare comunității specialiștilor din domeniu (secțiunea 4.6).

b) Contextul activității urmei, datorită virtualizării și posibilității migrării mașinii virtuale sau aplicației în cadrul infrastructurii, este deosebit de important în abordarea asumată deoarece urmele conțin acel istoric al defectelor indiferent de locația pe care o poate avea mașina virtuală sau aplicația la un anumit moment dat (secțiunea 4.6.1).

c) Prin asocierea dintre CBR și urme oferim rolul central utilizatorilor deoarece ei sunt implicați în elaborarea unei probleme, în regăsirea unei experiențe anterioare, precum și în adaptarea soluției. Ca rezultat, ei oferă sistemului abilități de achiziție a cunoștințelor continue prin elaborarea de cunoștințe în timpul fiecărei interacțiuni. Practic, sistemul oferă utilizatorilor urme reformulate, fiecare utilizator putând transforma urmele folosind propriile sale cunoștințe pentru problema ce trebuie rezolvată. Cu acest mod de abordare, fiecare utilizator poate avea propriul său punct de vedere cu privire la o problemă (secțiunea 4.5). Urmele, datorită „reținerii" întregului context al experienței anterioare, ne permit mai multe modalități de interacțiune în sisteme și în special combinarea mai multor modalități de interacțiune într-un singur sistem. Avantajul rezidă din faptul că utilizatorii diferiți, cu diverse abilități sau obiceiuri, vor putea să-și împărtășească experiența.

7. ***Impactul raționamentului SADU***

Noul sistem de raționament lasă utilizatorului posibilitatea să aleagă modalitatea de perfecționare a măsurii, de a adopta alta, sau de continuare a ciclului cu alte tipuri de cunoștințe (de adaptare sau de context) dacă problema pare să aibă o rezolvare bună. De asemenea, el poate decide, dacă oprește procesul aici sau îl continuă (secțiunea 4.5).

Sistemul de diagnoză este funcțional din primul moment, fără a necesita întreaga cazuistică inserată legată de contextul activității urmei, deoarece aceasta se completează pe parcursul desfășurării proceselor, iar în cazul apariției defectelor necunoscute, fără rezolvare imediată, problema este lăsată în discuție/analiză pentru rezolvare comunității specialiștilor din domeniu.

Credem că acest tip de raționament mixt (bazat pe cazuri și urme) va avea un impact semnificativ asupra experienței de partajare a aplicațiilor, în special atunci când acestea sunt bazate pe web și sistemul de raționament „bazat pe experiență" va fi partajat de către comunitatea utilizatorilor.

8. ***Propunerea unui nou model fuzzy de alocare dinamică și echilibrare a încărcării resurselor***

Contribuția adusă de modelul fuzzy (secțiunea 5.1) permite modelarea interacțiunii dintre resursele fizice sau virtuale disponibile datorită echilibrării încărcării lor. Această abordare privind reutilizarea resurselor

disponibile prin echilibrarea încărcării ne conduce la o mai bună gestionare a cererilor clienților și implicit la o creștere a randamentului sistemului IT concomitent cu creșterea fiabilității lui.

Algoritmul de echilibrare a încărcării resurselor disponibile, implicit modelul, bazat pe eficiență utilizării, conferă un acces mai facil la resursele neocupate nu doar pentru un procent din clienți ci pentru toți, semnalizând totodată gradul de încărcare și disponibilitatea resurselor (secțiunea 5.1.3). De asemenea, clienții conectați nu întâmpină dificultăți în accesarea serviciilor dorite și asta datorită minimizării timpilor de așteptare blocajelor sistemului deoarece cererile lor sunt direcționate către servere cu grad mic de încărcare.

Această modalitate de abordare asigură o funcționalitate în cel mai eficient și inteligent mod posibil deoarece este responsabilă de managementul resurselor și proceselor sistemului (secțiunea 5.1.4).

Abordarea este definitorie în stabilirea gradului de încărcare a resurselor disponibile, stabilind cu exactitate, funcție de ponderea pe care o are încărcarea resursei, dacă sistemul funcționează în parametri optimi (secțiunea 5.1.5).

9. Modelarea matematică a stării funcționale a unui sistem

Modelul permite unui sistem complex ce are în componență „m" resurse cu ponderi și încărcări procentuale diferite ale resurselor (secțiunea 5.2.1) să semnalizeze în două moduri (notare și reprezentare printr-un cod al culorilor) starea funcțională globală a sistemului la un anumit moment dat (secțiunea 5.2.4).

Noutatea constă în modelarea interacțiunii dintre resursele fizice și cele virtuale disponibile cu ajutorul geometriei hiperplanelor (secțiunea 5.2.2), în primul rând pentru o atenționare a stării funcționala a sistemului și în al doilea rând permițând o notificare a încărcării resurselor în cadrul sistemului (secțiunea 5.2.3) în vederea unei alocări dinamice a lor. Această abordare conduce la o uniformizare a gradului de încărcare a resurselor eterogene distribuite în sistem, funcție de caracteristicile de performanță (tehnice) pe care acestea le au.

Totodată, modelul permite o realocare dinamică a hiperplanelor în funcție de încărcarea sistemului (secțiunea 5.2.2), cu scopul rezolvării problemei într-un mod automat, în fiecare moment, ținând cont de experiența anterioară a sistemului. Modelarea permite astfel sistemului să se „auto-repare" prin înlăturarea situației care conduce la o stare de nefuncționalitate a sistemului printr-o re-alocare dinamică a resurselor disponibile în vederea echilibrării încărcării lor.

Supralicitarea, potrivit modelării ne permite o mai bună echilibrare a nevoilor pe care le poate avea o aplicație, la un moment dat, în utilizarea resurselor disponibile. Modalitatea de a suplimenta și accesa cu încă o unitate o anumită resursă conduce la o deblocare a proceselor care au loc în momentul unei cereri suplimentare dar și la o funcționare în parametri normali ai sistemului. Acest lucru poate conduce la o alocare dinamică a resursei în funcție de nevoile pe care le poate avea sistemul la un anumit moment dat.

Modelul nu se limitează doar la a semnaliza o stare de nefuncționalitate ci rezolvă întreaga problematică a echilibrării încărcării resurselor din cadrul sistemului, învățând din experiența anterioară, înlăturînd astfel automat stărea de nefuncționalitate.

10. Stabilirea unei ponderi de influență a unei „situații" în funcționarea globală a sistemului.

Modelul matematic permite o atenționare a punctelor slabe din cadrul sistemului funcție de ponderea pe care o au resursele, permițând doar suplimentarea acelei resurse care a dus la micșorarea performanțelor sistemului (secțiunea 5.2.3). Deoarece inițial am considerat (la momentul t_0) ponderea celor trei parametri egală din cadrul sistemului, acest lucru se poate schimba pe parcursul evoluției stării funcționale prin acordarea de ponderi diferite resurselor sistemului în funcție de comportarea sistemului la momentele anterioare apariției evenimentului.

Datorită modalității de notificare, funcție de nota obținută de sistem la momentul apariției defectului (t_1) și nota din momentul anterior (t_0), rezultă ponderea pe care o are defectul în aprecierea/deprecierea stării funcționale a sistemului. Astfel sistemul va avea posibilitatea de a învăța din experiența anterioară și de a face o prognoza mai precisă a viitoarelor stări comportamentale (secțiunea 5.2.3).

Astfel, la apariția defectului, agentul inteligent va eticheta defectul respectiv, funcție de gravitatea situației, dată de modalitatea de apreciere a stării funcționale a sistemului.

11. Semnalizarea iminenței stării de colaps a sistemului.

O astfel de situație poate să apară atunci când încărcarea procentuală a unei resurse depășește limitele prestabilite și nu au fost luate măsuri în vederea creșterii disponibilității resursei. Pentru modelul propus este determinantă modalitatea de semnalizare și monitorizare a viitoarelor necesități din sistem, pentru ca acesta să funcționeze în parametri optimi. Prin urmare, dacă un sistem semnalizează o posibilă stare de colaps, avem indicii sigure privind o anume nevoie de resurse ce trebuie asigurată, prevenind situația de nefuncționalitate (secțiunea 5.2.3).

Permite evaluarea stării globale a sistemului ce intervine în alocarea punctuală a resurselor către un factor critic datorită:

a. Situației (stare de defect)- conectare la rețea, client, servicii background
b. Componentelor (procesor, RAM, HDD)
c. Direcțiilor de criticizare a resurselor
d. Constrângerilor I/O
e. Partajării resursei

12. Receptarea soluției indicate pentru evitarea stării de colaps

În urma semnalizării stării de colaps (secțiunea 5.2.3), soluția indicată de model ne spune viitoarele necesități pe care le are sistemul în vederea unei funcționări în parametri optimi. Astfel, soluția este inovatoare deoarece indică numărul de unități cu care trebuie suplimentată resursa R_j, în cazul în care aceasta este supraîncărcată cu diferite cereri ale sistemului

Studiul de față a dovedit că abordarea noastră este posibilă, bazându-ne pe cercetările efectuate în domeniu și rezultatele obținute pe parcursul studiului, concretizate prin publicarea de lucrări indexate în baze de date internaționale (BDI) dar și articole susținute la diferite manifestări științifice naționale și internaționale, cotate ISI, după cum urmează:

1. **Florin POSTOLACHE, Viorel ARITON, TUREAC Cornelia Elena, Filip Alin CONSTANTIN**, *LOADING AND DYNAMIC ALLOCATION MATHEMATICAL METHOD OF COMPLEX SYSTEM RESOURCES*, The 15th "International Business Information Management Association" Conference - IBIMA 2010, Section: **Software Development and Performance Measurement**, November 6-7, 2010, Cairo, Egypt, pp. 1420-1430, ISBN: 978-0-9821489-4-5, http://www.ibima.org/. Articolul cotat ISI propune o modelare matematică care urmărește optimizarea utilizării resurselor disponibile dintr-un sistem, implicit a proceselor, ținând cont de cererile sosite și eliminând tot ce ar putea produce o stare de nefuncționalitate, conștientizând totodată conceptele „funcționalității maxime" și „durabilității maxime" pentru tehnica de calcul, care este expusă celei mai rapide faze de învechire din punct de vedere tehnologic. Propune metoda matematică de alocare dinamică și echilibrare a încărcării resurselor în sistemele fizice abstractizate tratată în capitolul *Abordări aproximative privind alocarea și încărcarea resurselor pentru SADU (secțiunea 5.2)*.

2. **Florin POSTOLACHE, Severin BUMBARU, Filip Alin CONSTANTIN**, *FRAMEWORK ON VIRTUALISATION APPLICATIONS AND BENEFITS*, The 4th International Workshop on „Soft Computing Applications" - SOFA2010, July 15-17, 2010 – Arad, Romania, IEEE Catalog Number: CFP1028D-CDR, ISBN: 978-1-4244-7983-2, pp. 83 – 88, **Print ISBN:** 978-1-4244-7985-6 **INSPEC Accession Number:** 11516913, **Digital Object Identifier:** 10.1109/SOFA.2010.5565620, http://sofa2010.org/index.html, http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=5565620&abstractAccess=no&userType=. Articolul cotat ISI subliniază faptul că orice sistem complex poate fi virtualizat atunci când cerințele de lucru impun acest fapt, conducând la o consolidare a serverului și la beneficii pe măsură. Tratează multiplele aplicații ale virtualizării, modalitatea de virtualizare a serverului, rețelei și spațiului de stocare precum și tranziția aplicațiilor de la standalone spre Cloud. Descrie primii pași făcuți în validarea implementării modelului propus privind abstractizarea infrastructurii conform capitolului *Sisteme inteligente*.

3. **Filip Alin CONSTANTIN, Florin POSTOLACHE, Valentin CURTEANU, Cornelia Elena TUREAC, Liviu Mihail MARINESCU, Emanuel Stefan MARINESCU**, *FRAMEWORK OF DANUBIUS ONLINE COLLABORATION AND LEARNING ENVIRONMENT*, The 6th International Seminar on „Quality Management in Higher Education"- QMHE 2010, July 8-9, 2010, Tulcea, The Center for Continuing Education and Training (CETEX) at "Gheorghe Asachi" Technical University of Iasi, Romania, Vol II Quality Management in Higher Education: pp. 447-450, ISBN 978-973-662-566-4 (VOL I: IDS Number-BTW64, ISBN: 978-973-662-567-1, **VOL II**: IDS Number-BTW63, ISBN 978-973-662-568-8), http://www.cetex.tuiasi.ro/qmhe2010/index.php. Articolul cotat ISI descrie modalitatea de lucru într-un mediu virtual colaborativ pe portalul Danubius Online, instalat la rândul lui pe o mașină virtuală din cadrul infrastructurii IT abstractizate.

4. **Florin POSTOLACHE**, *FRAMEWORK ON ECONOMICAL IMPLICATION AND ISSUES OF SADU IMPLEMENTATION*, ACTA UNIVERSITATIS DANUBIUS. ŒCONOMICA No 2(7)/2011, pp. 139-154, Print ISSN: 2065-0175, Online ISSN: 2067-340X, http://journals.univ-danubius.ro/index.php/oeconomica/article/view/893. Lucrarea, indexată BDI, tratează pe larg etapele dezvoltării sistemului de asistare a diagnosticianului uman, abordarea asumată, arhitectura și funcționarea SADU precum și descrierea SADU. Lucrarea se încheie cu detalierea rezultatelor în urma implementării.

5. **Florin POSTOLACHE, Severin BUMBARU, Viorel ARITON**, *COMPLEX SYSTEMS VIRTUALIZATION IN THE CURRENT'S ECONOMICAL CONTEXT*, EuroEconomica Nr. 3(26)/2010 - ISSN 1582-8859, pp. 29-50, http://journals.univ-danubius.ro/index.php/euroeconomica/article/view/714. Lucrarea, indexată BDI, tratează în amănunt domeniul abstractizării resurselor fizice (rețea, mediu de stocare, servere), planificarea, obiectivele și beneficiile aduse de virtualizarea infrastructurii IT. Propunem spre dezbatere un nou model al mașinii abstractizate hardware prin prisma straturilor ce o compun. Reliefăm importanța straturilor pentru achiziția și structurarea cunoștințelor cu ajutorul agenților în vederea diagnozei defectelor.

6. **Florin POSTOLACHE, Viorel ARITON, Florentina Loredana TACHE, Cătălin NĂCHILĂ, Alin Constantin FILIP,** *INTELIGENT AGENTS IN KNOWLEDGE ACQUISITION AND STRUCTURING FOR VIRTUAL SISTEMS DIAGNOSIS*, ACTA UNIVERSITATIS DANUBIUS. ŒCONOMICA No 3(8)/2010, pp. 140-160, Print ISSN: 2065-0175, Online ISSN: 2067-340X, http://journals.univ-danubius.ro/index.php/oeconomica/article/view/706. Articolul, indexat BDI, tratează achiziția și structurarea cunoștințelor pentru diagnoza sistemelor virtualizate cu ajutorul agenților inteligenți. Lucrarea analizează sistemul prin prisma straturilor care îl compun, pornind de la cel *hardware* (physical layer) la cel de abstractizare a resurselor (*virtual layer*), de la alocarea si monitorizarea resurselor pe mașinile virtuale (*virtual machines layer*) la starea funcțională a sistemelor de operare și a aplicațiilor (*OS&Apps layer*). Agenților li se pot atribui diferite grade de libertate în ceea ce privește modul lor de acționare iar posibilei soluții la cazul problemă i se pot atribui diferite rating-uri.

7. **Severin BUMBARU, Andy PUSCA, Florin POSTOLACHE,** *TEACHING WITH TECHNOLOGY: DANUBIUS UNIVERSITY CASE STUDY*, The 2nd International Conference on „Institutional Strategic Quality Management in Higher Education"- ISQM 2010, 14-16 October 2010, Sinaia – Romania, ISBN: 978-606-8154-11-4, pp. 55-64 http://proiecte.aracis.ro/academis/promovare-diseminare-transfer/conferinte-internationale/. Articolul susține posibilitatea că o aplicație bine consolidată, mare consumatoare de resurse și accesată de un mare număr de clienți, poate funcționa în parametri optimi pe o mașină virtuală în cadrul infrastructurii abstractizate.

8. **Florin POSTOLACHE, Severin BUMBARU, Viorel ARITON**, *KNOWLEDGE ACQUISITION AND STRUCTURING FOR DIAGNOSIS IN COMPLEX VIRTUAL SYSTEMS*. The 1st International Conference "Advances in Engineering & Management" – ADEM 2010 Drobeta Turnu-Severin, May 19-21, 2010, Faculty of Engineering and Management of Technological Systems, University of Craiova, ISBN 978-606-510-899-8, http://www.imst.ro/adem/adem.htm. Articolul conturează etapele dezvoltării unui proiect de achiziție de

cunoștințe prin descrierea modalităților de captare, analiză și modelare a cunoștințelor în vederea realizării unei diagnoze a defectelor, particular asupra sistemului virtualizat. Articolul descrie în amănunt pașii necesari realizării unui proiect de achiziție de cunoștințe și este indexat BDI.

9. **Viorel ARITON, Vasile PALADE, Florin POSTOLACHE**, *COMBINED DEEP AND SHALLOW KNOWLEDGE IN A UNIFIED MODEL FOR DIAGNOSIS BY ABDUCTION* - EuroEconomica Nr. 1(20)/2008 - ISSN 1582-8859, pp. 33-42, http://journals.univ-danubius.ro/index.php/euroeconomica/article/view/297. Relevă importanța experienței specialistului uman în procesul de diagnoză a defectelor. Articolul propune o abordare unificată a diagnozei bazată pe criterii de plauzibilitate și relevanță aplicate printr-o implementare conexionistă. De altfel, în cadrul achiziției de cunoștințe, granularitatea influențează diagnoza în raport cu mărimea ei. Articolul este indexat BDI.

10. **Viorel ARITON, Florin POSTOLACHE**, *THE DIAGNOSIS BY ABDUCTION USING HUMAN EXPERT KNOWLEDGE*- ACTA UNIVERSITATIS DANUBIUS. ŒCONOMICA Nr 1(2)/2006 ISSN: 2065-0175, pp. 161-178, http://journals.univ-danubius.ro/index.php/oeconomica/article/view/36. Articolul nuanțează valoarea cunoștințelor procedurale versus cunoștințe conceptuale, a cunoștințelor explicite versus cunoștințe tacite, dobândite de către expertul uman. Lucrarea oferă sugestii privind proiectarea și construirea unui sistem de diagnoză bazat pe aceste cunoștințe dobândite de către expert, precum și modalitatea de izolare ierarhică a erorilor. Demonstrează rolul determinant pe care îl ocupă inițial agentul uman în diagnoza defectelor. Articolul este indexat BDI.

6.3. Direcții viitoare de cercetare.

În viitor, atenția va fi îndreptată spre o continuare a cercetării întreprinse în această lucrare.

Pe de o parte, din punct de vedere teoretic, pe lângă o analiza detaliată se va încerca identificarea punctelor sensibile/vulnerabile din sistem, pe baza informațiilor culese de la fiecare unitate de calcul, în vederea îmbunătățirii modalității de alocare a sarcinilor într-un mod cât mai optim posibil. Astfel, cererea tot mai mare de resurse de către procesele active din cadrul unui sistem conducând de cele mai multe ori la o încărcare excesivă a resurselor disponibile, la o funcționare anormală sau blocare a anumitor subsisteme din componența lui, va trebui analizată și modelată astfel încât sistemul să evite inconvenientele amintite.

În acest caz, vom recurge la o posibilă generalizare, care va consta în luarea în calcul a unei cascade de sisteme astfel încât la o prognoză de funcționare defectuoasa la „n" sisteme, să intre în funcționare noul sistemul „n+1".

Într-o infrastructură abstractizată, în momentul accesării sistemului de către un număr mare de clienți sau în cazul unei cereri suplimentare de resurse, acesta poate identifica punctele vulnerabile și preîntâmpina neajunsurile printr-o alocare dinamica a resurselor existente.

Particular, în momentul accesării unei aplicații de către un număr mare de clienți, mașina virtuală care suportă respectiva aplicație se poate clona, permițând astfel o redistribuire a clienților, respectiv sarcinilor, conducând la o decongestionare a traficului existent și implicit la o diminuare a încărcării resurselor mașinii respective.

Acest lucru, în viziunea noastră determină eliminarea blocării în funcționare sau conduce la o micșorare a timpul de răspuns ca urmare a cererilor sosite din partea clienților.

BIBLIOGRAFIE SELECTIVĂ

[1]. Aha, D.W., Breslow, L. A., & Munoz- Avila, H. (2001). Conversational CBR. Applied Intelligence.

[2]. Ariton, V., Palade, V., & Postolache, F. (2008). Combined deep and shallow knowledge in a unified model for diagnosis by abduction. *EUROECONOMICA* Nr. 1(20)/2008 - ISSN 1582-8859. http://EconPapers.repec.org/RePEc :dug:journl:y:2007:i:19:p:33-42

[3]. Arthur, W.B. (2009). What is Technology and How does it Evolve? Book excerpt from The Nature of Technology, NY. Acad. of Sciences.

[4]. Brachman, R. J. & Levesque, H. J. (2004): Knowledge representation and reasoning, *Morgan Kaufmann Publishers*, New York.

[5]. Cooper, D. & La Rocca, G. (2007). Knowledge-based Techniques for Developing Engineering Applications in the 21st Century, *7Th AIAA Aviation Technology, Integration and Operations Conference*, Belfast, Northern Ireland.

[6]. Cordier, A.; Fuchs, B.; Lieber, J.; & Mille, A. (2007). Interactive Knowledge Acquisition in CBR. *Proceedings of the workshop on Knowledge Discovery and Similarity* (at ICCBR'07), 85–94. Workshop of the seventh International Conference on CBR Workshop.

[7]. Deleuze, G. (2002). The Actual and the Virtual, in: *Dialogues*, Second Edition, trans. Eliot Ross Albert, *Columbia UP*.

[8]. Diekert, V., & Rozenberg, G. (1995). The Book of Traces. *World Scientific Publishing*, Singapore.

[9]. Doguc, O. & Ramirez-Marquez, J. E. (2009). An Efficient Fault Diagnosis Method for Complex System Reliability. *7th Annual Conference on Systems Engineering Research (CSER 2009)*. Loughborough University, UK

[10]. El-Abd, A.E. (2002). Load balancing in distributed computing systems using fuzzy expert systems. *International Conference on Modern Problems of Radio Engineering, Telecommunications and Computer Science* (TCSET 2002), Lviv-Slavsko, Ukraine.

[11]. Gertler, J.J. (1998). Survey of Model-Based Failure Detection and Isolation in Complex Plants, *IEEE Control Systems Magazine*, vol. 8, pp. 3–11.

[12]. Gordon, J.L. (2000). Creating knowledge maps by exploiting dependent relationships, *Knowledge Based Systems* 13(2-3): pp71-79.

[13]. Gruber, T. R. (1993). A translation approach to portable ontology specification. *Knowledge Acquisition*, 5, pp.199-220.

[14]. Hey, J. (2004). The Data, Information, Knowledge, Wisdom Chain:The Metaphorical link, *Intergovernmental Oceanographic Commission - OceanTeacher: a training system for ocean data and information management*.

[15]. Honavar, V.,& Caragea, D., (2008), Towards Semantics-Enabled Distributed infrastructure for Knowledge Acquisition, Association for the Advancement of Artificial Intelligence (AIIDE 2008) Conference, Chicago, SUA.

[16]. Ioan, C. A., (2008). Matematica aplicata în economie, *Editura Universitara Danubius*, Galati, ISBN 978-973-1746-81-4

[17]. Karacapilidis, N., Trousse, B., & Papadias, D. (1997). Using CBR for argumentation with multiple viewpoints. *CBR Research and Development* (ICCBR'97), volume 1266 of Lecture Notes in AI, pp. 541–552. Springer.

[18]. Kasabov, N., (2000), Future Directions for Intelligent Systems and Information Sciences, Heidelberg, *Physica-Verlag* (Springer Verlag).

[19]. Kumar, A., Singhal, M., & Ming, T. L. (1987). A model for distributed decision making: An expert system for load balancing in distributed systems. *11th Symposium on Operating Systems*.

[20]. Leake, D., and Dial, S. A. (2008). Using case provenance to propagate feedback to cases and adaptations. *Proceedings of 9th European Conference on Advances in CBR*, pp. 89–103, Springer-verlag., Trier, Germany.

[21]. Leinberger, W., Karypis, G., & Kumar, V., (2000). Load Balancing Across Near-HomogeneousMulti-Resource Servers. *Proceedings. 9thHeterogeneous Computing Workshop* (HCW 2000) Cancun, Mexico

[22]. Liu, B. (2007). Web Data Mining: Exploring Hyperlinks, Contents and Usage Data. *Springer*, ISBN 3540378812

[23]. Luger, G., & Stubblefield, W. (2004). Artificial Intelligence: Structures and Strategies for Complex Problem Solving (5th ed.). *The Benjamin/Cummings Publishing Company Inc.* ISBN 0-8053-4780-1, pp. 35–77.

[24]. Malik, S. (2000). "Dynamic Load Balancing in a Network of Workstation, *95.515F Research Report*.

[25]. Massumi, B. (2002). Parables for the Virtual. Movement, Affect, Sensation, *Duke University Press.*
[26]. Merali, Y., & Davies, J. (2001). Knowledge Capture and Utilization in Virtual Communities, *Proc. of the First International Conference on Knowledge Capture*, Canada.
[27]. Michalski, R.S., Carbonell, J.G., & Mitchell T.M. (1983), Machine Learning: An Artificial intelligence Approach, *Tioga Publishing Company*, ISBN 0-935382-05-4.
[28]. Milton, N. R. (2007). Knowledge Acquisition in Practice: A Step-by-step Guide. London: *Springer.*
[29]. Milton, N. R. (2008). Knowledge Technologies,Polimetrica ® *S.a.s. Milano*
[30]. Munir, S. A., Bin, Y. W., Biao, R., & Man, M. (2007). Fuzzy Logic based Congestion Estimation for QoS in Wireless Sensor Network. *Wireless Communications and Networking Conference*, WCNC 2007. IEEE. Kowloon, pp. 4336-4341.
[31]. Patton, R. J., Lopez-Toribio, C. J. & Uppal, F. J. (1999). Artificial intelligence approaches to fault diagnosis for dynamic systems. *International Journal of Applied Mathematics and Computer Science* 9 (3): 471-518.
[32]. Popper, F. (2007). From Technological to Virtual Art. *Leonardo Books*, MIT Press.
[33]. Postolache, F., Bumbaru, S., & Arinton, V. (2010). Knowledge acquisition and structuring for diagnosis in complex virtual systems. *The 1st International Conference "Advances in Engineering & Management"* ADEM 2010 – Drobeta Turnu-Severin, ISBN 978-606-510-899-8.
[34]. Postolache, F., Bumbaru, S., & Constantin, F.A. (2010). Framework on virtualisation applications and benefits, *The 4th International Workshop on Soft Computing Applications* - SOFA2010, Arad, Romania, IEEE Catalog Number: CFP1028D-CDR, ISBN: 978-1-4244-7983-2, http://sofa2010.org/index.html.
[35]. Roberts J., (2000). From Know-how to Show-how? Questioning the Role of Information and Communication Technologies in Knowledge Transfer". *Technology Analysis & Strategic Management*; Vol. 12, No. 4.
[36]. Rowley, J., (2007). The wisdom hierarchy: representations of the DIKW hierarchy. Journal of Information Science, 33(2), pp: 163-180.
[37]. Russell, S.J., & Norvig, P. (2003). Artificial Intelligence: A Modern Approach (2nd ed.), Upper Saddle River, New Jersey: Prentice Hall, ISBN 0-13-790395-2.
[38]. Settouti, L., Prié, Y., Champin, P.-A., Marty, J.-C., & Mille, A. (2009). A TBMS framework: Models, languages and semantics. *Research Report*, http://hal.inria.fr
[39]. Sharma S., Singh S., and Sharma M., (2008). Performance Analysis of Load Balancing Algorithms, *World Academy of Science*, Engineering and Technology, vol. 38.
[40]. Shivaratri G., Krueger P., and Singhal M. (1992). Load Distributing for Locally Distributed Systems, Computer, vol. 25, pp. 33-44.
[41]. Steinder, M., & Sethi, A. S. (2004). A Survey of Fault Localization Techniques in Computer Networks. [Elsevier] Science of Computer Programming, S.I. on Network and System Administration.
[42]. Wooldridge, M. (2002). An Introduction to Multiagent Systems, *John Wiley & Sons Ltd*, Baffins Lane, Chichester, West Sussex P O 19 IUD, England.
[43]. Xu, Z. & Huang, R. (2009). Performance Study of Load Balancing Algorithms in Distributed Web Server Systems. *Parallel and Distributed Processing Project Report.*
[44]. Zimmermann, H. (2001). Fuzzy set theory and its applications. Boston: *Kluwer Academic Publishers.* ISBN 0-7923-7435-5.

Anexa A

Cuprinde situația monitorizării sistemului supus testării, demonstrând totodată o funcționare în parametri optimi și o disponibilitate destul de ridicată a resurselor disponibile dincadrul sistemului abstractizat.

Figurile A.1. – A.17. indică încărcarea și alocarea resurselor fizice disponibile pe mașinile virtuale gazdă ale mașinii hardware Blade 1 din cadrul infrastructurii.

Anexa B

Prezintă modalitatea de generare a numerelor aleatoare în calculatoarele numerice folosind distribuția uniformă (*uniform deviate*) în intervalul [0,1) și modalitatea de semnalizare a sistemului pentru un necesar suplimentar de resursă.