Sistem inteligent de instruire în programare centrat ontologic[1]

Ştefan Trăuşan-Matu

Universitatea "Politehnica" Bucureşti, Facultatea de Automatică şi Calculatoare

Centrul de Cercetări Avansate în Învăţare Automată, Prelucrarea Limbajului Natural şi Modelare Conceptuală al Academiei Române

email:trausan@cs.pub.ro, trausan@valhalla.racai.ro

URL: http://sunsite.pub.ro/people/trauşan, http://www.racai.ro/~trauşan

Rezumat

Lucrarea de faţă urmăreşte două scopuri. În primul rând sunt prezentate o serie de cercetări desfăşurate în ultimii ani şi materializate într-un sistem complex de instruire inteligentă asistată de calculator. În al doilea rând, se doreşte să se evidenţieze rolul deosebit de important jucat de utilizarea unei ontologii în dezvoltarea unui program complex : În sistemul de faţă toate modulele sistemului împart aceeaşi ontologie (bază de cunoştinţe).

Cuvinte cheie: *sistem inteligent de instruire, inteligenţă artificială, sisteme bazate pe cunoştinţe, instruire pe web, WWW*

1. INTRODUCERE

În lucrare este prezentat un sistem inteligent de instruire dezvoltat în jurul unei ontologii care include cunoştinţe atât ale domeniului considerat cât şi cunoştinţe pedagogice şi referitoare la studenţi. Sistemul este dezvoltat conform unor principii de asigurare a personalizării, a integrării mai multor modalităţi de instruire, inclusiv prin World Wide Web (WWW). Domeniul considerat este programarea calculatoarelor. În asigurarea unei ergonomii crescute au fost făcute şi cercetări în colaborare cu Institul de Psihologie al Academiei Române [5].

Programarea calculatoarelor devine din ce în ce mai mult o activitate cu puternice legături cu filosofia. De exemplu, achiziţia, reprezentarea şi prelucrarea cunoştinţelor umane sunt activităţi centrale atât în inteligenţa artificială cât şi în filosofie. Pe de altă parte, una din principalele probleme ale sistemelor cu inteligenţă artificială este problema implementării unor elemente specific umane, cum ar fi creativitatea, conştiinţa, intuiţia, înţelegerea. În acest context nu este de loc surprinzător că se discută din ce în ce mai mult astăzi de ontologii în ştiinţa calculatoarelor. Mai mult, ontologiile stau la baza şi a taxonomiilor de obiecte folosite în paradigma de programare orietată spre obiecte. De asemena, analiza şi specificarea programelor şi sistemelor informatice au ca scop definirea a ceea ce se presupune că există, adică a unei ontologii. În plus, în prezent există ontologii plasate pe web pentru a utilizate în aplicaţii (de exemplu, WordNet [24]),

Lucrarea începe prin discutarea problematicii ontologiilor, cu referire directă la ştiinţa calculatoarelor. Capitolul al doilea prezintă sistemul realizat. În final sunt trase câteva concluzii şi sunt precizat câteva direcţii pentru viitor.

2. ONTOLOGII

Programele de inteligenţă artificială simbolică prelucrează structuri simbolice, care reprezintă cunoştinţele referitoare la domeniul considerat. Aceste structuri simbolice formează o aşa numită bază de cunoştinţe, care constituie, de fapt, un model al domeniului respectiv. În ultimii ani se consideră că această bază de cunoştinţe trebuie văzută ca o ontologie, o conceptualizare, o teorie asupra ceea ce există în domeniul respectiv. O ontologie este o: “specificare a unei conceptualizări … Termenul este împrumutat din filosofie, unde însemna o considerare sistematică a existenţei. În inteligenţa artificială se referă la precizarea a ceea ce se consideră că <<există>>” [4].

O ontologie include categoriile, conceptele fundamentale, proprietăţile acestora, relaţiile şi distincţiile între ele. Un rol esenţial între relaţiile posibile îl joacă relaţia taxonomică, de la general la specific, prin care se pot “moşteni” proprietăţi de la conceptul mai general la cel mai particular, dacă aceste proprietăţi nu sunt redefinite la conceptul din urmă. Alte relaţii utile sunt “parte-întreg”, “opus” etc.

[1] Lucrarea a apărut în Lucrarile Conferintei de informatica teoretica si tehnologii informatice, CITTI 2000, ISBN 973-8082-10-2, 973-9286-59-3, Constanta, 2000, pag. 58-63.

Din perspectiva sistemelor bazate pe cunoştinţe se poate spune că o ontologie este o bază de cunoştinţe organizată astfel încât să permită accesarea şi manipularea unei mulţimi de concepte fundamentale într-un anumit domeniu şi a relaţiilor dintre ele. O ontologie trebuie să poată fi extinsă şi eventual restructurată. Ontologiile pot fi folosite în mai multe aplicaţii bazate pe cunoştinţe (de exemplu, în proiectare, diagnosticare, explicare, instruire etc.).

Ontologiile sunt definite (implicit) şi în aplicaţii care nu aparţin inteligenţei artificiale. Bibliotecile de obiecte din mediile de programare orientate spre obiecte conţin şi ele ontologii implicite: fiecare clasă este un concept. Este vorba, de această dată, nu de ceea ce există în realitate ci de o bibliotecă de obiecte ce reprezintă ceea ce există la dispoziţia programatorului, a entităţilor artificiale care pot fi asamblate şi combinate în sistemele create. Ierarhiile de interfeţe sunt ierarhii de comportări (sau meta-obiecte în sens XRL [1], de exemplu) şi pot fi considerate tot ontologii. Un set de adnotări XML [25] poate fi şi el considerat ca o ontologie de nivel 1.

O tendinţă de ultimă oră este plasarea pe web a unor microteorii pentru diverse domenii, adică a unor ontologii care se pot refolosi. O astfel de abordare este sprijinită de:

- limbaje de “interschimb” al cunoştinţelor cum ar fi KIF, CGIF [2],
- protocolul de acces la baze de cunoştinţe OKBC [7],
- limbaje dezvoltate în XML pentru descrierea ontologiilor (SHOE [10], OML [8]).

Efortul de a capta aspectele legate de semantică, de conţinutul documentelor face obiectul şi propunerii de standard RDF Schema [9], destinat a fi folosit împreună cu XML. RDF Schema se apropie ca idee de funcţionalitatea unui limbaj de descriere a ontologiilor.

Considerarea bazei de cunoştinţe ca o ontologie are, din punct de vedere tehnic, mai multe avantaje:

- În primul rând, adăugarea de noi cunoştinţe, aşa numita achiziţie de cunoştinţe precum şi învăţarea sau restructurarea bazei de cunoştinţe, sunt simplificate, sunt direcţionate în cazul în care cunoştinţele preexistente sunt structurate într-o ontologie. Este mult mai uşor să adaugi un concept nou la un eşafodaj existent, coerent, prin găsirea unuia sau mai multor concepte mai generale decât el, prin combinarea lor şi prin delimitarea unor diferenţe faţă de acestea.
- Un alt avantaj este faptul că se pot construi ontologii generale, care surprind conceptele, categoriile fundamentale, ontologii care sunt apoi extinse pentru fiecare aplicaţie în parte. Ontologiile pot fi astfel refolosite, în prezent ele fiind disponibile chiar pe Internet, unele din ele, cum ar fi WordNet [24] incluzând peste o sută de mii de concepte, această ontologie fiind curent folosită în aplicaţiile de înţelegere a limbajului uman.
- Ontologiile permit şi o interacţiune om-calculator mult mai prietenoasă, simplifică dialogul în limbaj uman prin particularizarea exprimărilor la proprietăţile fiecărui concept în parte.

O perspectivă mult folosită în inteligenţa artificială este cea în care sistemele bazate pe cunoştinţe, în afara faptului că îşi reprezintă mediul exterior în baza proprie de cunoştinţe, au şi o reprezentare a propriilor scopuri, alternative, alegeri etc., considerate în rezolvarea unei probleme. Elemente specifice conştienţei lui Chalmers [3], cum ar fi integrarea informaţiilor, raportabilitatea stărilor mentale, accesul la propriile stări, direcţionarea atenţiei şi controlul comportamentului necesită un model atât al mediului exterior cât şi unul al propriilor cunoştinţe, scopuri, raţiuni etc. Toate acestea se pot include şi ele în ontologia programului.

Pentru reprezentarea bazei de cunostinte, a ontologiei prezentului sistem de instruire asistata de calculator, s-a utilizat mediul XRL [1]. Acesta face parte din categoria mediilor de programare multiparadigma destinate aplicatiilor de inteligenta artificiala in general si sistemelor bazate pe cunoştinţe in particular. Aceste medii de programare sunt dezvoltate în ideea punerii la dispoziţia utilizatorilor a unei game largi de instrumente şi tehnici de prelucrare a cunoştinţelor astfel încât acesta să poata folosi pentru orice problemă particulară o reprezentare cât mai naturală şi directă. La baza sistemului stă paradigma de reprezentare a cunoştinţelor orientată spre obiecte structurate ceea ce face XRL adecvat pentru reprezentarea de ontologii. XRL este scris în COMMON LISP.

Fiecare concept din domeniul pentru care este dezvoltat sistemul de instruire este descris ca un obiect XRL. Conceptele au atribute (sloturi) care le caracterizează, meta-descrieri şi metode asociate mesajelor la care pot răspunde obiectele corespunzatoare. Atributele pot fi, la rindul lor, descrieri de obiecte XRL. Conceptele pot moşteni atribute sau metode de la alte concepte. De exemplu, conceptele funcţie anonimă, functională şi funcţie cu nume etc. vor fi descrise ca mai jos.

```
(unit function
  self
   (a unit
     supers
       (programming_concept
```

```
       procedural_abstraction)
    toC tocprocedure
    ml genfunml
    lisp genfuncl
    scheme genfunscheme
    paraph parapf)
  args    (x)
  vars    nil
  result  nil
  domain  nil
  range   nil
  name    nil
  nr_args 1
  body    nil)

(unit functionala
  self (a unit supers (function)))

(unit named_function
  self (a unit supers (function))
  name f)
```

Obiectul map descrie functionala care mapeaza o functie pe o lista. Acesta are drept corp (body) o particularizare a obiectului itereaza_pe_lista. O alta functionala este filter, care filtreza elementele unei liste care satisfac un anumit predicat.

```
(unit map
  self (a unit supers (functionala_pe_lista))
  name map
  body (a itereaza_pe_lista
         c2c (a cons 1st (a funcall name f)
                     rest (a reccall
                            name map
                            args (f l)))))

(unit filter
  self (a unit supers (functionala_pe_lista))
  name filter
  body (a itereaza_pe_lista
         c2c (a if_then_else
               test (a funcall name f
                               args (x))
               then (a cons
                       1st x
                       rest (a reccall
                              name filter
                              args (f l)))
               else (a reccall
                       name filter
                       args (f l)))))
```

3. SISTEMUL INTELIGENT DE INSTRUIRE

3.1 Principii

Sistemul inteligent de instruire a fost dezvoltat conform mai multor principii derivate din dorinţa de a obţine o comportare inteligentã, personalizatã şi cât mai cuprinzãtoare, incluzând toate activitãţile care apar în instruirea clasicã şi ţinând cont de noile realitãţi (de exemplu, WWW). Principiile sunt:

1. Sistemul este dezvoltat în jurul unei ontologii a domeniului considerat. Aceasta îndeplineşte mai multe roluri:

- regimuri de funcţionare personalizate, conform mai multor scenarii posibile,
- generare flexibilã de explicaţii,
- generare de hipertext in format HTML si XML, care poate fi parcurs de programele specializate pentru World Wide Web (WWW), de exemplu, Netscape sau InternetExplorer (versiunea 5 pentru XML),
- generare automatã de suport de curs,
- generare automatã de teste grila,
- analiza raspunsurilor date de studenţi la teste şi adaptarea desfaşurării ulterioare a lecţiei în funcţie de aceste rezultate,
- generare de cod într-un limbaj de programare,
- analizã (inginerie inversã) a programelor scrise de studenţi.

Centrarea ontologică a sistemului este, după părerea noastră esenţială. În primul rând, după cum se vede şi de mai sus, şi după cum s-a discutat în secţiunea anterioară, o ontologie, prin reprezentarea explicită a cunoştinţelor şi prin genericitatea ea, poate fi reutilizată pentru diverse aplicaţii. În al doilea rând, ea poate fi uşor modificată şi extinsă.

2. Baza de cunoştinţe referitoare la studenţi conţine atât informaţii asupra cunoştinţelor însuşite şi asupra greşelilor fãcute de student, cât şi asupra tipului de personalitate specificã fiecãrui student. În acest ultim scop, se foloseşte o taxonomie de tipuri de personalitãţi.

3. Explicaţiile date de sistem pot fi adaptate la fiecare student in parte, în funcţie de cunoştinţele sale şi de tipul de personalitate cãruia aparţine. Generarea explicaţiilor depinde şi de taxonomia de concepte a domeniului (în conexiune cu trãsãturile personale ale studenţilor).

4. Sistemul este destinat sprijinirii:

- studenţilor, prin lecţii care includ explicaţii, parcurgeri de hipertext WWW, teste, evaluări, reveniri etc.,
- profesorilor, prin instrumente pentru achiziţia de noi concepte, pentru generarea automată de texte (cursuri şi teste grilă) şi pentru gestiunea datelor referitoare la lecţii şi studenţi.

5. Sistemul poate folosi resurse web in format HTML sau XML.

După cum s-a discutat anterior, pentru reprezentarea bazei de cunoştinţe a prezentului sistem de instruire asistată de calculator, s-a decis utilizarea mediului XRL [1]. Baza de cunoştinţe are trei componente distincte [13, 16, 17]:

- Baza de cunoştinţe ale domeniului pentru care se dezvoltă sistemul. În aceasta au fost incluse mai multe concepte care formează o taxonomie pe baza relaţiei de moştenire, prezentată in faza anterioară.
- Baza de cunoştinţe referitoare la studenţi şi la tipuri de personalităţi de studenţi. Conceptele din baza de cunoştinţe asupra studenţilor sunt de trei categorii: personalităţi generice, studenţi şi faze de pregătire ale studenţilor.
- In baza de cunoştinţe pedagogice sunt descrise scenarii de desfăşurare a lecţiilor, paşi standard ai unei lecţii, descrierea parametrilor lecţiei curente precum şi informaţii referitoare la cursurile acoperite de sistem.

3.2 Generarea flexibilă de explicaţii

O problema centrală în instruire (student-profesor sau student-calculator) este problema comunicării cunoştinţelor. Pentru realizarea unei comunicări optime, instruirea trebuie să fie particularizată la student, la cunoştinţele sale, la trăsăturile proprii de personalitate. Una din componentele majore ale procesului comunicării în instruire este explicarea noţiunilor. De calitatea acestor explicaţii depinde foarte mult succesul procesului de instruire. De fapt, un profesor bun este un profesor care explică bine.

Pentru a adapta explicaţiile la particularităţile fiecărui student în parte, în abordarea de faţă s-a mers pe mai multe directii:

- particularizarea explicaţiilor la tipul de personalitate căreia aparţine studentul,
- introducerea mai multor perspective asupra bazei de cunoştinţe ale domeniului şi adaptarea explicaţiilor la aceste perspective,
- furnizarea de explicaţii multiple, complementare, unui aceluiasi student (de exemplu, prin prezentarea narativă a conceptului, prin textul unor programe de calculator ilustrative - in cazul conceptelor specifice domeniului programării calculatoarelor -, prin raportarea conceptului la alte concepte sau prin enumerarea atributelor definitorii),
- generarea de explicaţii in limbaj natural.

Calitatea explicaţiilor date (atât de om cât şi de un program de calculator) depinde foarte mult de:

- acurateţea limbajului folosit;
- utilizarea unor reguli retorice de organizare a textului;
- adaptarea prezentării la personalitatea interlocutorului.

În sistemul realizat [17] s-a cautat să se ţină cont de aceste idei. Trebuie aici precizată diferenţa între acurateţea limbajului în sens de generarea de propozitii coerente semantic şi corecte gramatical şi, pe de altă parte, de organizarea textului, a semanticii propoziţiilor acestuia pe baza unor reguli retorice.

Pentru realizarea dezideratelor de mai sus, in proiectarea sistemului de instruire s-a plecat de la o arhitectură în care conlucrează mai multi agenţi, fiecaruia din aceştia corespunzându-i un modul in sistem [17]:

- agent pentru analiză taxonomică;
- agent retor;
- agent lingvist;
- agent pentru generare de text;
- agent pentru afisare frumoasa a textelor.

Analiza taxonomica este folosita in urmatoarele scopuri :

- furnizarea de informatii pentru expertul in retorica,
- achizitia de cunostinte,
- generarea de teste grilă,
- restructurarea bazei de cunostinte.

Analiza este realizată de “agenti de analiză taxonomică”, care pleacă de la :

- un concept şi explorează împrejurimile sale (relaţiile hiper/hiponimice, meronimice, de similaritate etc.),
- doua concepte si exploreaza legatura intre ele (distanta intre ele, asemanari, deosebiri, discriminari - ultimele trei clase de informatii sunt folosite si de generatorul de teste si de analiza rezultatelor),
- intreaga baza de cunostinte.

Rolul structurarii unui text conform unor reguli de retorica este foarte important. In acest mod se evita

monotonia si se creste inteligibilitatea textului. Totodata, regulile de retorica constituie modalitati de comunicare a unor informatii suplimentare.

Pe baza analizei taxonomice, expertul in retorica planifica modul de prezentare a conceptelor. De exemplu, daca un obiect are doar doua subconcepte, ele pot fi prezentate prin antiteza unul fata de celalalt, ca în exemplul de mai jos, în care, plecând de la urmatoarea taxonomie de obiecte (un subset al taxonomiei obiectelor din chiar sistemul de instruire):

```
OBIECT_PEDAGOGIC
        PERSONALITATE
            PERSONALITATE_STANDARD
            PERSONALITATE_VARIANTA_1
        STUDENT
            . . .
        FAZA
        SCENARIU
            SCENARIU_VARIANTA_1
            SCENARIU_STANDARD
                SCENARIU_VARIANTA_2
        PAS
        . . .
```

s-a generat urmãtorul text:

```
       Scenariu_varianta_1 e un scenariu, in
plus fata de scenariu proprietatile lui
scenariu_varianta_1          sunt          pasi,
examinare_hipertext,        verificare        si
analiza_greseli.     Spre     deosebire     de
scenariu_varianta_1      proprietatile      lui
scenariu_standard  sunt  pasi,  initializare,
explicare, verificare si evaluare.     . . .
```

3.3 Generarea automată de hipertext în format HTML pentru WWW

Sistemul de faţã poate genera automat fişiere HTML, plecând de la descrierea conceptelor sub forma de obiecte XRL [17, 19, 20, 23]. Pentru fiecare concept este generatã o paginã de web. Structura de reţea a hipertextului urmeazã fidel structura impusã de relaţiile hiperonimice şi meronimice a bazei de concepte. Parcurgerea hipertextului generat se poate face cu programele specializate (Netscape, Internet Explorer, Lynx etc.).

În [22] este prezentat un experiment de a generare automatã de fişiere de stil în limbajul XSL (vezi [25]) pentru vizualizarea personalizatã a fişierelor XML.

3.4 Generarea de teste grilă

O componenta esenţialã a unui proces de instruire este verificarea însuşirii cunoştinţelor de cãtre studenţi. În acest scop se poate folosi un dialog în care profesorul (sistemul inteligent de instruire) pune întrebãri sau o serie de teste grilã. În sistemul dezvoltat [17] a fost implementat un modul care generează automat teste grilã. Acest modul are urmatoarele caracteristici:

- Sunt generate mai multe tipuri de teste grilã.
- Fiecare test grilã este generat automat plecând de la baza de cunoştinţe a domeniului.
- Succesiunea tipurilor de teste grilã este aleasã aleator pentru a evita monotonia.
- Conceptele considerate în generarea de teste grilã se incadrează în subiectul lecţiei curente consideratã.
- Succesiunea conceptelor care constituie subiectul testelor poate fi aleatoare (pentru a evita monotonia) sau poate fi focalizatã (în cazul în care testele sunt date într-o faza a unei lecţii, în care se doreşte analiza insuşirii unui anumit concept).
- Raspunsurile date de student la testele grila sunt memorate, ele fiind analizate pentru a construi un model al cunostintelor insusite de student.

Generarea aleatoare de teste grila este exemplificatã în continuare:

```
BUBBLE_SORT este caracterizata ca avind
COMPLEXITATE :
1 - N**2
2 - NLOGN

Care este raspunsul exact? 1

BINE

HEAP_SORT si INSERTION_SORT se diferentiaza
prin:

1 - STABILITATE
2 - IN_SITU

Care este raspunsul corect? 1

BINE

QUICK_SORT si MERGE_SORT se diferentiaza prin:

1 - IN_SITU
2 - NATURALETE

Care este raspunsul corect? 2

GRESIT
```

3.5 Generarea de programe sursă şi ingineria inversă a programelor scrise de studenţi

Plecând de la ideile din [12], au fost implementate metode de generare de programe sursã în limbajul ML

(de remarcat cã pot fi utilizate, dupa modificări minore, şi metodele de generare de cod C sau de pseudo-cod din [12]). Rezultatul acestei generari de cod este utilizat în explicarea notiunilor prezentate [16].

Tehnicile de inginerie inversã din [12] au fost utilizate şi pentru dezvoltarea unui modul de înţelegere a programelor studenţilor scrise în limbajul C [18, 21].

4. CONCLUZII

Toate facilităţile sistemului de instruire prezentate în lucrare au fost experimentate [11-23] şi s-a demonstrat fezabilitatea lor. Parametrii calitativi ai realizărilor practice au corespuns aşteptărilor. Din cauza lipsei de resurse umane, sistemul nu a fost încă dezvoltat într-o variantă disponibilă larg. În prezent se lucrează la plasarea unei variante a sistemului pe WWW pentru uzul curent al studenţilor de la Universitatea "Politehnica" Bucureşti.

Un rol central în sistem îl joacă ontologia, care include cunoştinţele domeniului considerat, cunoştinţele referitoare la studenţi şi cele pedagogice. Toate modulele sistemului sunt dezvoltate în jurul acestei ontologii, demonstrând importanţa centrării ontologice.

Până în momentul de faţă, ontologia a fost creată şi introdusă în sistem semi-automat (s-au scris manual descrieri XRL ale conceptelor şi s-au achiziţionat semi-automat noi concepte printr-o tehnică de tip grile repertoar). În continuarea cercetărilor se va trece şi la conectarea sistemului la ontologiile existente pe WWW.

O parte a tehnicilor experimentate în sistem sunt în prezent implementate într-un sistem de asistare a învăţării terminologiei financiare în limba engleză, care face obiectul unui proiect Copernicus al Comunităţii Europene [6].

4. BIBLIOGRAFIE

[1] M. Bărbuceanu, Şt. Trăuşan-Matu, Integrating Declarative Knowledge Programming Styles and Tools in a Structured Object Environment, in J. Mc.Dermott (ed.) Proceedings of 10-th International Joint Conference on Artificial Intelligence IJCAI'87, Italia, Morgan Kaufmann Publishers, Inc., 1987.

[2] Conceptual Graphs, http://concept.cs.uah.edu/cg/cg-standard.html

[3] D. Chalmers, Facing Up to the Problem of Consciousness, http://ling.ucsc.edu/~chalmers/papers/facing.html.

[4] Gruber, T., What is an Ontology, http://www.kr.org/top/definitions.html

[5] Gh. Iosif, I. Juvină, Ştefan Trăusan-Matu, A. Marhan, Aspecte ale achiziţiei de cunostinţe în modelarea studentului pentru un sistem inteligent de instruire, Revista de psihologie, Tomul 45, 1-2, p.31-50, 1999.

[6] Learning Foreign Language Terminology in a Scientific Domain, Proiect Copernicus, http://www-it.fmi.uni-sofia.bg/larflast

[7] Open Knowledge Base Connectivity, http://www.ai.sri.com/~okbc

[8] http://wave.eecs.wsu.edu/CKRMI/OML.html

[9] RDF Schema Specification, W3C, http://www.w3.org/TR/WD-rdf-schema, 1999.

[10] http://www.cs.umd.edu/projects/plus/SHOE/

[11] Şt. Trăuşan-Matu, L. Negreanu, Studiu asupra proceselor cognitive implicate in invatare precum si asupra abordarilor existente in reprezentarea acestora. Construirea unui cadru de reprezentare si prelucrare a cunostintelor implicate in procesele cognitive din instruire, Raport RACAI RR-6, dec.94

[12] Şt. Trăuşan-Matu, Sisteme de dezvoltare asistata de calculator a programelor; ingineria inversa a programelor. Teza de doctorat, UPB, 1994

[13] Şt. Trăuşan-Matu, Studiu asupra categoriilor de cunostinte implicate in procesele de instruire si a modalitatilor de reprezentare ale acestora, Raport RACAI RR-2, aug.94.

[14] Şt. Trăuşan-Matu, Lopătan, C., Specificaţii de proiectare a unui mediu de generare de sisteme hipertext pentru structurarea şi conceptualizarea cunoştinţelor din texte, Studiu. Grant CNCSU. 1995

[15] Şt. Trăuşan-Matu, Negreanu, L., Definirea unui formalism de reprezentare a cunostintelor unui student aflat intr-un proces de instruire, Raport RACAI RR-9, Centrul de Cercetari Avansate in Invatarea Automata, Prelucrarea Limbajului Natural si Modelare Conceptuala, Academia Romana, 1995.

[16] [Tra96] Şt. Trăuşan-Matu, Integrarea sistemului de asistare a instruirii cu un sistem de intelegere si generare a limbajului natural, raport RACAI RR-18, Academia Romana, CCAIAPLNMC, dec. 1996.

[17] Şt. Trăuşan-Matu, Negreanu, L., Sistem inteligent de asistare a instruirii, raport RACAI, RR-14, Academia Romana, iunie 96.

[18] Şt. Trăuşan-Matu, Lorina Negreanu, Roxana Spalatelu, Sistem experimental de intelegere a programelor scrise de studenti, Raport RACAI

RR-21, Academia Romana, CCAIAPLNMC, decembrie 1997.

[19] Şt. Trăuşan-Matu, Sistem bazat pe cunostinte pentru generarea de hipertext WWW in scop didactic, Grant CNCSU, noiembrie 1997.

[20] Şt. Trăuşan-Matu, Knowledge-Based, Automatic Generation of Educational Web Pages, in Proceedings of Internet as a Vehicle for Teaching Workshop, Ilieni, iun. 1997, pp.141-148.

[21] Şt. Trăuşan-Matu, Lorina Negreanu, Mihaela Spalatelu, Tricky Errors in C Programs and Their Detection by Knowledge-Based Reverse Engineering, ROSYCS'98, va apare in Scientific Annals of the "Alexandru Ioan Cuza" University of Iasi, Computer Science Section, 1998.

[22] Şt. Trăuşan-Matu, Produs program de generare dinamica de pagini de Web într-un sistem de instruire inteligent, Report RACAI RR-47, Dec 1999.

[23] Şt. Trăuşan-Matu, Web Page Generation Facilitating Conceptualization and Immersion for Learning Finance Terminology, RILW99, http://rilw.emp.paed.uni-muenchen.de/99/papers/Trausan.html

[24] WordNet, http://www.cogsci.princeton.edu/~wn/

[25] Extensible Markup Language (XML), www.w3.org/XML