Aplicații colaborative bazate pe ontologii

Vlad Posea
Facultatea de Automatică şi calculatoare,
Universitatea Politehnica Bucureşti
Splaiul Independenței 313, Bucureşti
vladposea@yahoo.com

Ovidiu Mara
Facultatea de Automatică şi calculatoare,
Universitatea Politehnica Bucureşti
Splaiul Independenței 313, Bucureşti
ovidiu.mara@gmail.com

REZUMAT
În această lucrare se descrie integrarea unor tehnologii bazate pe ontologii, specifice webului semantic, ca extensie într-o aplicație colaborativă existentă, de tip forum, cu scopul de a-i oferi funcționalități avansate de căutare şi clasificare a mesajelor, bazate pe semantica termenilor identificați în text.

Cuvinte cheie
Ontologie, forum, web semantic, OWL, adnotări.

Clasificare ACM
H5.2. Information interfaces and presentation (e.g., HCI): Miscellaneous.

INTRODUCERE

Această lucrare descrie realizarea unei extensii care îmbunătățește o aplicație de tip forum existentă, prin adăugarea de tehnologii specifice webului semantic.

Aplicațiile colaborative de tip forum sunt aplicații web care au ca scop punerea în comun a informațiilor de către utilizatori, sub formă de mesaje text, uneori cu imagini sau alte ataşamente. Ca structură, un forum este organizat în mai multe discuții. Fiecare discuție are un subiect definit de utilizatorul care a inițiat-o, şi un prim mesaj scris de acesta, în care de obicei el cere un răspuns de la ceilalți utilizatori ai forumului. Aceştia pot posta mesaje care vor fi incluse în discuția respectivă, în ordine cronologică.

Când un utilizator accesează forumul, i se oferă o pagină cu legături către discuțiile existente, de obicei ordonate după data ultimei modificări. Întrucât într-un forum folosit de mulți utilizatori, numărul de discuții poate deveni foarte mare, chiar de ordinul zecilor de mii, găsirea unei anumite discuții este practic imposibilă fără o funcție de căutare automată. În aplicațiile forum folosite pe larg în prezent, căutarea se face textual: utilizatorul introduce unul sau mai multe cuvinte cheie, iar aplicația îi afişează o listă cu discuțiile (sau cu mesajele individuale, în funcție de implementare) în care, în titlul sau în textul mesajelor, se găsesc exact cuvintele respective.

Acest mod de implementare a funcției de căutare este adesea ineficient pentru utilizator, neoferindu-i mesajele căutate, deşi ele există pe forum, pentru că nu conțin exact cuvintele cheie folosite în căutare. Utilizatorul trebuie să repete de mai multe ori căutarea, încercând diverse combinații de sinonime şi formulări înrudite ca sens ale cuvintelor despre care crede că s-ar găsi în mesajele respective, iar de fiecare dată trebuie să parcurgă o listă de rezultate în cea mai mare parte nefolositoare. Problema este că în tot acest timp informația căutată se găseşte în baza de date a forumului, dar aplicația nu este capabilă să o interpreteze decât ca pe o înşiruire de caractere, fără niciun sens.

O alternativă la căutarea textuală este adnotarea manuală a mesajelor. Când un utilizator publică un mesaj pe forum, pe lângă câmpurile obişnuite pentru titlu, text, ataşamente etc. pe care le are la dispoziție, poate completa şi o listă de adnotări – cuvinte cheie pe care le consideră reprezentative pentru mesajul respectiv, dar care nu sunt neapărat conținute textual în mesaj. Când un alt utilizator caută un mesaj cu o anumită informație, poate căuta în lista tuturor adnotărilor existente în acel moment. Acest tip de căutare poate fi util în anumite cazuri, când se realizează o căutare după un număr foarte mic de cuvinte cheie. Pentru căutari mai complexe şi mai specifice, poate fi chiar mai puțin eficient decât căutarea textuală, deoarece utilizatorii care postează mesaje au tendința de a folosi termeni generali în adnotări, un număr mic de cuvinte cheie, şi rareori mai multe cuvinte care sunt înrudite ca sens – dacă utilizatorul are nenorocul să caute după un cuvânt care e doar apropiat ca sens cu altul conținut în adnotare, nu va găsi ceea ce caută.

Soluția care depăşeşte considerabil în eficiență cele două abordări prezentate mai sus constă în integrarea în aplicația forum a unui sistem ce foloseşte o ontologie, cu dublu scop: adnotarea automată a mesajelor din forum cu termeni care fac parte din ontologie şi oferirea unei funcții avansate de căutare, nu după cuvinte cheie, ci după o combinație de noțiuni existente în ontologia respectivă.

WEBUL SEMANTIC

Webul semantic este o extensie evoluată din web în care conținutul paginilor web poate fi exprimat nu numai în limbaj natural, ci şi într-un format ce poate fi citit şi folosit de programe software. Acest lucru permite căutarea, distribuirea şi integrarea informațiilor într-un mod mult mai uşor.

Webul semantic constă într-o filozofie, un set de principii de proiectare şi o varietate de tehnologii folosite [1]. O parte dintre elementele webului semantic au fost definite în specificații formale, cum sunt RDF (Resource Description Framework), RDF Schema şi OWL (Web Ontology Language), toate trei oferind capacitatea de a descrie formal concepte, termeni şi relații între ele, într-un domeniu de cunoştințe dat – o ontologie.

Standarde existente pentru stocarea ontologiilor

RDF este o familie de specificații ale consorțiului W3C, proiectate original ca un model pentru metadate, dar care a ajuns să fie folosit ca metodă generală de a reprezenta informația, într-o varietate de sintaxe create ca extensii ale RDF (care, la rândul său, este bazat pe XML) [4], [8].

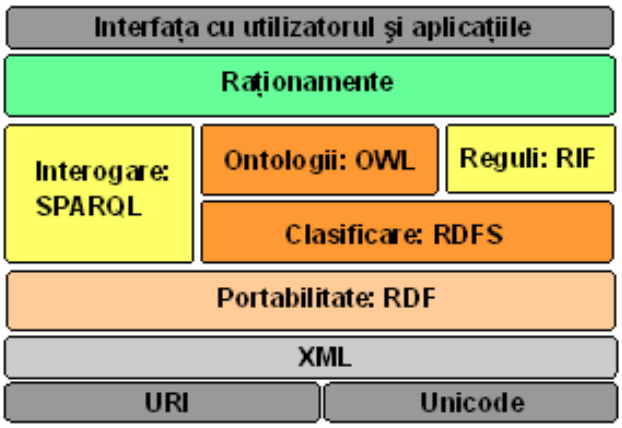

Figura 1 Modele pentru reprezentarea datelor în webul semantic, bazate pe XML

Modul în care sunt reprezentate informaţiile în RDF se bazează pe emiterea de propoziţii de forma subiect-predicat-obiect, numite triplete în terminologia RDF. Subiectul denotă o resursă, iar predicatul denotă trăsături sau aspecte ale resursei, şi exprimă o relaţie între subiect şi obiect. [11]

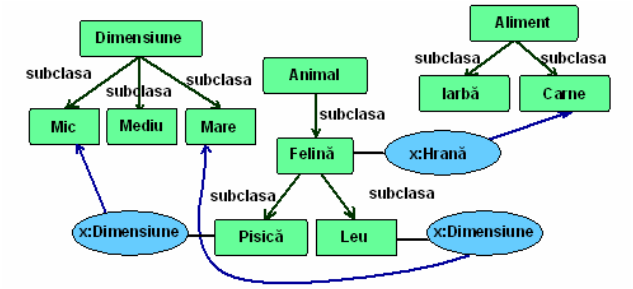

Figura 2 Reprezentare grafică a unui exemplu de ontologie descrisă prin RDFS; clasele au proprietăţi, cu valori reprezentate de alte clase

RDF Schema sau RDFS este un limbaj bazat pe RDF şi conceput în mod special pentru reprezentarea cunoştinţelor. Se introduc noţiunile de clasă – o resursă poate fi declarată ca fiind clasă pentru alte resurse, care sunt considerate indivizi ai clasei respective – şi moştenire – o clasă poate fi declarată ca subclasă a alteia mai cuprinzătoare. [5]

OWL (Web Ontology Language – [2], [3]) este un limbaj folosit pentru descrierea ontologiilor folosite pe web şi destinat în mod special prelucrării automate a informaţiilor de către calculator. OWL, în comparaţie cu RDFS, introduce numeroase noi elemente de sintaxă, pentru descrierea mai detaliată a proprietăţilor şi a claselor: printre altele, se introduc noi relaţii între clase (de exemplu, excluziune), constrângeri de cardinalitate (de exemplu, „exact unul"), egalitatea, caracteristici ale proprietăţilor (de exemplu, simetria sau tranzitivitatea) şi clase enumerate.

OWL cuprinde trei sublimbaje, OWL Lite, OWL DL (*description logic*) şi OWL Full, cele trei fiind diferenţiate prin expresivitate, dar şi prin gradul de complexitate computaţională. [10]

Figura 3 Definirea unei clase în OWL DL cu ajutorul operatorului reuniune

OWL DL oferă posibilitatea foarte utilă de a defini clase cu ajutorul operatorilor pe mulţimi: reuniune, intersecţie şi complement. Acest lucru permite o construcţie a ontologiilor relativ simplă, în anumite cazuri fiind utilă o abordare de la specific la general, prin descrierea unei superclase ca o enumerare de subclase.

Obţinerea de informaţii existente în ontologii

Ontologiile nu ar fi utile dacă informaţia conţinută în acestea nu ar putea fi extrasă într-un anumit fel. În acest scop au fost concepute mai multe limbaje de interogare a ontologiilor, mai mult sau mai puţin asemănătoare cu limbajele folosite pentru interogarea bazelor de date. În prezent, unul dintre aceste limbaje, SPARQL, a devenit standard de facto pentru interogarea ontologiilor. SPARQL permite realizarea de interogări folosind şabloane de triplete, conjuncţii şi disjuncţii. [12]

O altă posibilitate de a extrage informaţii este folosirea unui program numit reasoner. Un astfel de program primeşte o ontologie şi poate construi taxonomia acesteia (clasificarea completă a tuturor resurselor cuprinse în ontologie), în funcţie de implementare fiind uneori capabil şi de a verifica consistenţa ontologiei. Un reasoner poate fi folosit pentru a interoga o ontologie OWL DL în felul următor: în ontologia respectivă, se creează o clasă nouă, care reprezintă conceptul căutat; clasa se defineşte cu ajutorul operatorilor pe mulţimi (reuniune, intersecţie şi complement) pe baza claselor existente, eventual cu specificarea unor restricţii pentru proprietăţile sale, atât ca şi cardinalitate, cât şi ca valori luate de proprietăţile respective. După construcţia acestei descrieri, se apelează reasonerul şi, într-un protocol bine stabilit, i se transmite ontologia, împreună cu o cerere de enumerare, de exemplu, a claselor echivalente cu clasa nou definită. Se va obţine o listă a claselor existente în ontologie care se potrivesc şablonului căutat. [6], [7]

EXTINDEREA APLICAŢIILOR FORUM FOLOSIND ONTOLOGII

Odată cu creşterea popularităţii forumurilor, volumul de date disponibil în astfel de aplicaţii a devenit din ce în ce mai mare; în acelaşi timp, datorită structurii rigide a acestui tip de aplicaţii, bazate pe fire de discuţii, organizarea informaţiilor după conţinut este practic imposibilă. Funcţiile de căutare textuală reprezintă, în prezent, principala metodă de extragere a informaţiilor, dar sunt deficitare deoarece nu sunt tehnic capabile de a

găsi întotdeauna textele căutate, de fapt, de utilizator, dacă acestea nu conţin exact cuvintele cheie căutate. [10]

Pentru a evidenţia avantajul funcţiilor de clasificare şi căutare a mesajelor folosind ontologii faţă de cele textuale, dar şi pentru a stabili caracteristicile care vor trebui îndeplinite de aceste funcţii, am creat trei scenarii ce exemplifică folosirea unui forum:

1) Utilizatorul X postează pe un forum un mesaj despre nivelul preţurilor maşinilor italiene; el nu foloseste în text cuvântul „italian", ci „Italia"; totuşi, deoarece beneficiază de existenţa unei tehnologii de adnotare manuală, el îşi marchează mesajul cu cheile „maşini italiene" şi „preţuri".

2) Pe acelaşi forum, utilizatorul Y doreşte să afle care este preţul unui Fiat; caută textual „preţ Fiat" şi „preţuri Fiat", dar nu găseşte nimic; încearcă să caute cu aceleaşi cuvinte cheie după adnotări, dar din nou nu primeşte niciun rezultat.

3) Utilizatorul Z accesează un forum în care se discută reţete culinare. El caută o reţetă pentru o pizza vegetariană, dar nu îi place ceapa; singurele cuvinte cheie la care se poate gândi sunt „pizza vegetariană" şi „fără ceapă". Din păcate, deşi pe forum există mai multe reţete de acest tip, niciuna nu conţine exact remarca „fără ceapă". În schimb, dacă se caută numai „pizza vegetariană", se obţin câteva zeci de mesaje ca rezultat, pe care Z va trebui să le citească unul câte unul, neştiind dacă va găsi ceea ce caută. El decide să deschidă o discuţie nouă, la care cineva îi răspunde cu o legătură către mesajul care conţine o astfel de reţetă.

Toate aceste trei exemple descriu situaţii în care o tehnologie incapabilă de a procesa informaţiile pe baza sensului acestora eşuează în a oferi funcţionalitatea dorită de utilizatori.

Pentru ca integrarea unei ontologii într-o aplicaţie de tip forum să depăşească inconvenientele acestor metode, am stabilit că funcţia ce va fi dezvoltată trebuie să îndeplinească următoarele cerinţe:

1) adnotare automată a mesajelor postate pe forum, bazată pe cuvintele conţinute în textul mesajului; utilizatorul poate să vizualizeze marcajele create, dar nu este nevoie să intervină în niciun fel;

2) căutare semantică, nu textuală: căutarea după cuvinte cheie este înlocuită de căutarea după concepte;

3) posibilitatea unei căutări avansate, în care se folosesc cât mai multe dintre avantajele unui vocabular puternic ca OWL DL;

4) uşurinţa utilizării: un utilizator obişnuit, care nu este familiar cu sintaxa unei ontologii sau cu limbaje de interogare ca SPARQL, trebuie să poată folosi căutarea semantică cu un minim de efort de învăţare. Cea mai bună soluţie pentru această cerinţă este crearea unei interfeţe grafice pentru căutare.

5) un efort minim de învăţare.

Descrierea soluţiei

Soluţia creată presupune folosirea unei ontologii, în format OWL DL, creată cu o aplicaţie externă. Ontologia respectivă trebuie să conţină o structură de clase care reprezintă concepte semnificative pentru forumul în care va fi folosită (de exemplu, în cazul forumului despre reţete culinare, va fi folositoare o ontologie care descrie alimentele cele mai întâlnite şi diverse produse alimentare). Aceste clase trebuie să fie ierarhizate într-o formă logică, familiară utilizatorului. În plus, este recomandat să se cuprindă în ontologie cât mai multe date despre conceptele respective şi relaţiile dintre ele – cu cât ontologia este mai bogată în informaţii, cu atât funcţia de căutare va fi mai eficientă. Acest lucru înseamnă că cel care creează ontologia trebuie să aibă grijă să includă relaţii de excluziune între conceptele care sunt incompatibile în realitate, să folosească operatorul de reuniune unde este utilă enumerarea entităţilor dintr-o categorie şi să stabilească concis care sunt domeniile şi codomeniile proprietăţilor din ontologie.

O astfel de clasificare poate necesita destul de mult timp şi efort pentru realizare, dar beneficiile sale sunt valabile în permanenţă, micşorând timpul necesar găsirii informaţiei pentru utilizatori, şi efortul moderatorilor de a redirecţiona utilizatorii între discuţii cu subiecte identice (discuţii dublate).

Folosind soluţia propusă, cele trei scenarii prezentate anterior trebuie să evolueze în felul următor:

1) Utilizatorul X postează pe forum un mesaj despre nivelul preţurilor maşinilor italiene; el observă că mesajul său a fost adnotat automat cu marcajele „ontologie:maşină italiană", „ontologie:preţ", „ontologie:Euro".

2) Pe acelaşi forum, utilizatorul Y doreşte să afle care este preţul unui Fiat; accesează funcţia de căutare semantică; alege din ontologie conceptul „Fiat". Obţine ca rezultat şi mesajul scris de X, deoarece în ontologia folosită clasa „Fiat" este o subclasă a „Maşină italiană".

3) Utilizatorul Z accesează forumul în care se discută reţete culinare. Accesează funcţia de căutare semantică, şi descrie conceptul pe care îl caută exact după reprezentarea pe care şi-a format-o mental: alege din ontologie conceptul „Pizza", la care adaugă o restricţie: proprietatea „are ingredient" nu trebuie să ia valorile „Carne" şi „Ceapă". La trimiterea cererii, primeşte ca rezultat un mesaj în care se prezintă o reţetă corespunzătoare.

Implementarea soluţiei

Pentru implementarea soluţiei, am ales drept componentă pentru accesarea şi manipularea ontologiei, platforma Protégé 3.3. Aceasta cuprinde un set de biblioteci realizate în limbajul Java, bine documentate, este open-source şi implementează şi specificaţia limbajului OWL DL. În plus, este compatibilă cu orice reasoner care foloseşte protocolul DIG, creat de *DL Implementation Group*. [6]

Ca reasoner, am folosit o aplicaţie numită Pellet, versiunea 1.5. Acesta este realizat tot în Java, este open-source, şi suportă toate elementele sintactice ale OWL-DL. El implementează o varietate de modalităţi de conectare cu alte aplicaţii bazate pe ontologii, inclusiv protocolul DIG, ceea ce a permis conectarea sa cu platforma Protégé.

Aplicaţia forum pe care am ales-o pentru a o extinde a trebuit să fie implementată în limbajul Java, pentru a o putea integra cu bibliotecile Protégé. Am ales forumul JForum, o reimplementare a binecunoscutei aplicaţii phpBB, din mai multe motive: JForum beneficiază de

avantajul uşurinţei în utilizare a interfeţei din phpBB, este familiar pentru majoritatea utilizatorilor de forumuri din întreaga lume şi este un proiect open-source matur, cu codul sursă bine structurat în module şi template-uri – ceea ce facilitează crearea extensiilor. În plus, suportă numeroase formate pentru baza de date, printre care MySQL şi PostgreSQL.

La implementarea extensiei, am ţinut cont de principiile de organizare şi funcţionare după care a fost construit JForum.

În primul rând, am creat un pachet numit *ontology*, în clasele căruia am implementat toate funcţiile necesare manipulării ontologiei, folosint API-ul Protégé.

În schimb, ontologia în sine, ca structură de date, a fost încărcată în clasa *OntologyRepository* din pachetul *Repository* – pachet care cuprinde toate clasele care se ocupă de încărcarea datelor din medii externe aplicaţiei şi memorarea lor într-un cache. Am respectat acest format deoarece JForum are implementată o funcţie disponibilă administratorului forumului, de golire a întregului cache şi resetare a aplicaţiei, în cazul în care apar probleme în funcţionare sau se modifică setări în fişierele de configurare a aplicaţiei.

Ontologiile care pot fi folosite în forum sunt incluse ca fişiere OWL într-un director special destinat; forumul poate să lucreze cu un singur fişier la un moment dat, şi indicarea numelui acestuia se face într-o intrare din fişierul de configurare a aplicaţiei.

Extensia integrată în JForum are două roluri: în primul rând, adnotarea automată a mesajelor postate pe forum, imediat după acţiunea de postare; în al doilea rând, funcţia de căutare semantică.

Pentru adnotarea mesajelor, am observat cum sunt implementate funcţiile de indexare şi de căutare a mesajelor în JForum. Aplicaţia foloseşte o bază de date, în care stochează mesajele folosind patru tabele: tabelul jforum_posts conţine numai informaţii adiţionale despre mesaj (data postării, identificatorul autorului etc.), tabelul jforum_posts_texts conţine titlul şi textul mesajului, tabelul jforum_search_words conţine o listă cu toate cuvintele existente în forum, iar tabelul jforum_words este o listă de legături one-to-many de la cuvintele cuprinse în jforum_search_words la indicii posturilor din forum.

La postarea unui mesaj, aplicaţia prelucrează titlul şi textul pentru a obţine lista de cuvinte cuprinse în mesaj, după care adaugă cuvintele respective în tabelul de cuvinte cheie şi adaugă legături în tabelul de legături, între cuvintele cheie şi noul mesaj.

Am profitat de structura existentă a bazei de date pentru a realiza memorarea adnotărilor mesajelor. Aceasta funcţionează în felul următor: la primirea titlului şi textului unui mesaj nou, pe lângă obţinerea listei de cuvinte, se trimite textul către clasa *OntologyIndexer*, care întoarce o listă de marcaje de forma „numeOntologie:numeClasă”. Aceste marcaje sunt adăugate atât la lista de cuvinte cheie specifice mesajului, cât şi într-un câmp special al tabelei jforum_posts numit „Tags”, pentru a fi vizualizate în browserul clientului, odată cu celelalte informaţii despre mesaj.

Adnotarea automată se face prin extragerea din textul mesajului a tuturor cuvintelor şi a sintagmelor până la o lungime dată (în număr de cuvinte), specificată în setările aplicaţiei. Acest lucru este necesar pentru a putea găsi în ontologie clase descrise prin mai multe cuvinte, de exemplu „pizza vegetariană”.

Fiecare sintagmă din text este căutată între clasele ontologiei. Căutarea se face după marcajul *rdf:label* al claselor din ontologie, dacă există; dacă nu, se face chiar după numele claselor. În ambele cazuri, identificatorii textuali ai claselor sunt memoraţi la încărcarea ontologiei într-un HashMap, menţinut în cache, pentru a permite o căutare rapidă.

În ceea ce priveşte funcţia de căutare semantică, s-a creat o pagină web pentru aceasta, în care s-a implementat interfaţa grafică pentru căutare.

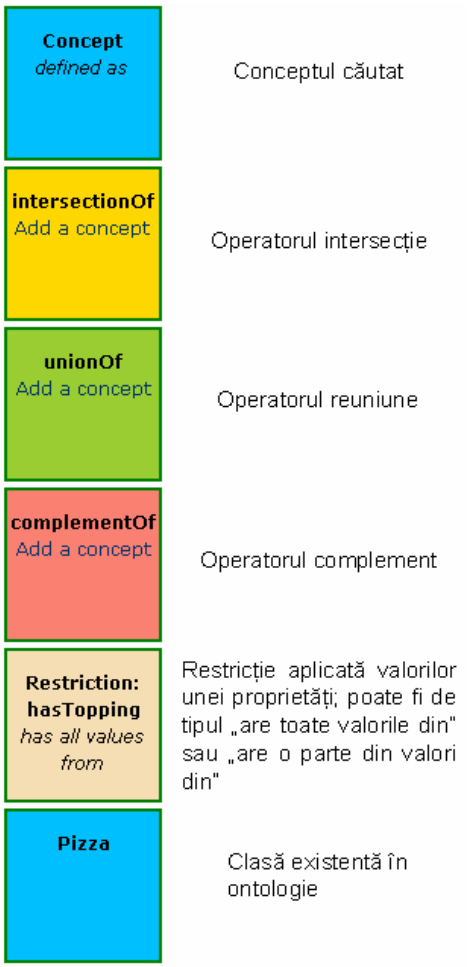

Figura 4 Elemente ale interfeţei de interogare grafică

Ca principiu, s-a dorit ca interfaţa să fie cât mai intuitivă şi mai uşor de folosit pentru un utilizator obişnuit. Din acest motiv s-a ales o abordare grafică, şi nu una textuală.

Pentru interogarea ontologiei, utilizatorul trebuie să descrie conceptul pe care îl caută. El poate face acest lucru construind un arbore în care, ca noduri, poate alege obiecte cu semnificaţii speciale (Figura 4).

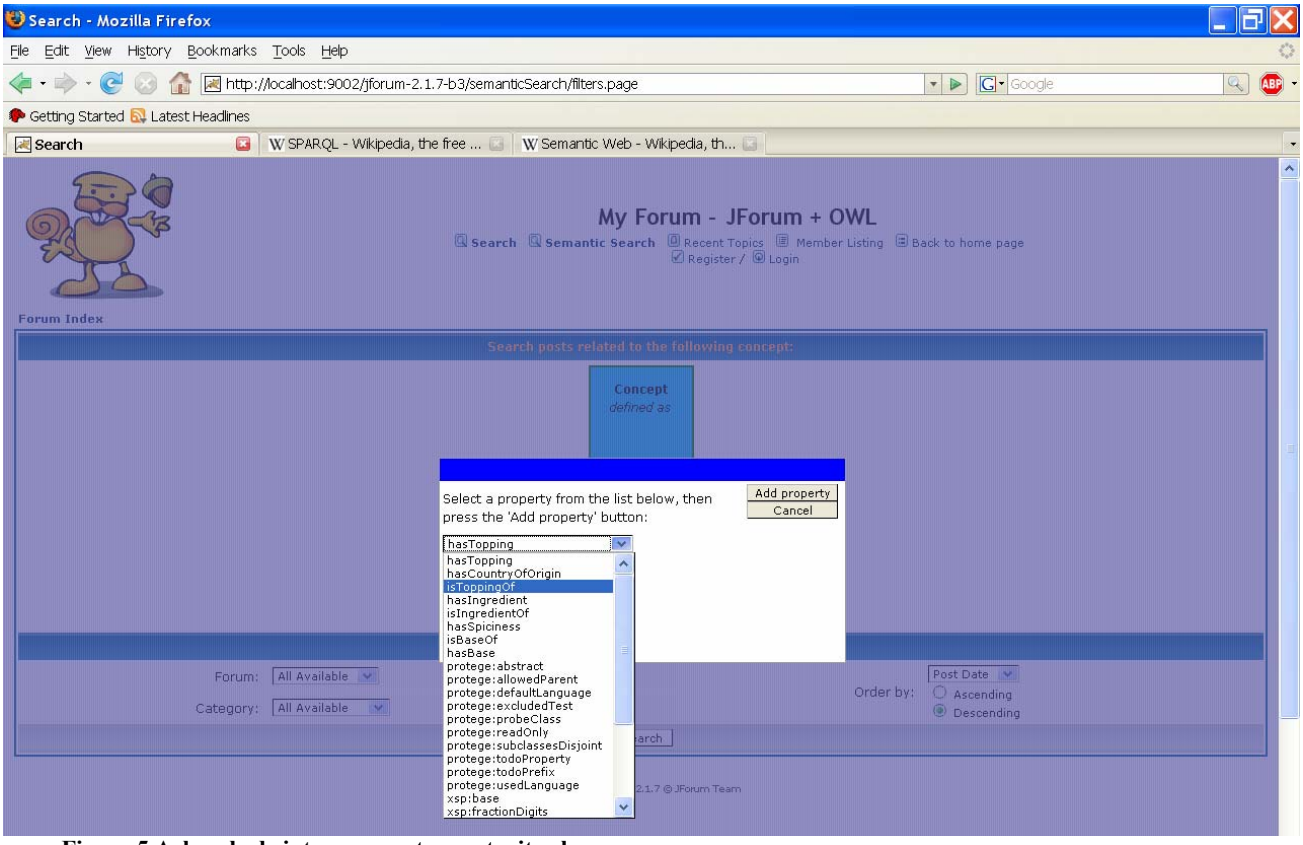

Figura 5 Arborele de interogare este construit sub îndrumarea calculatorului

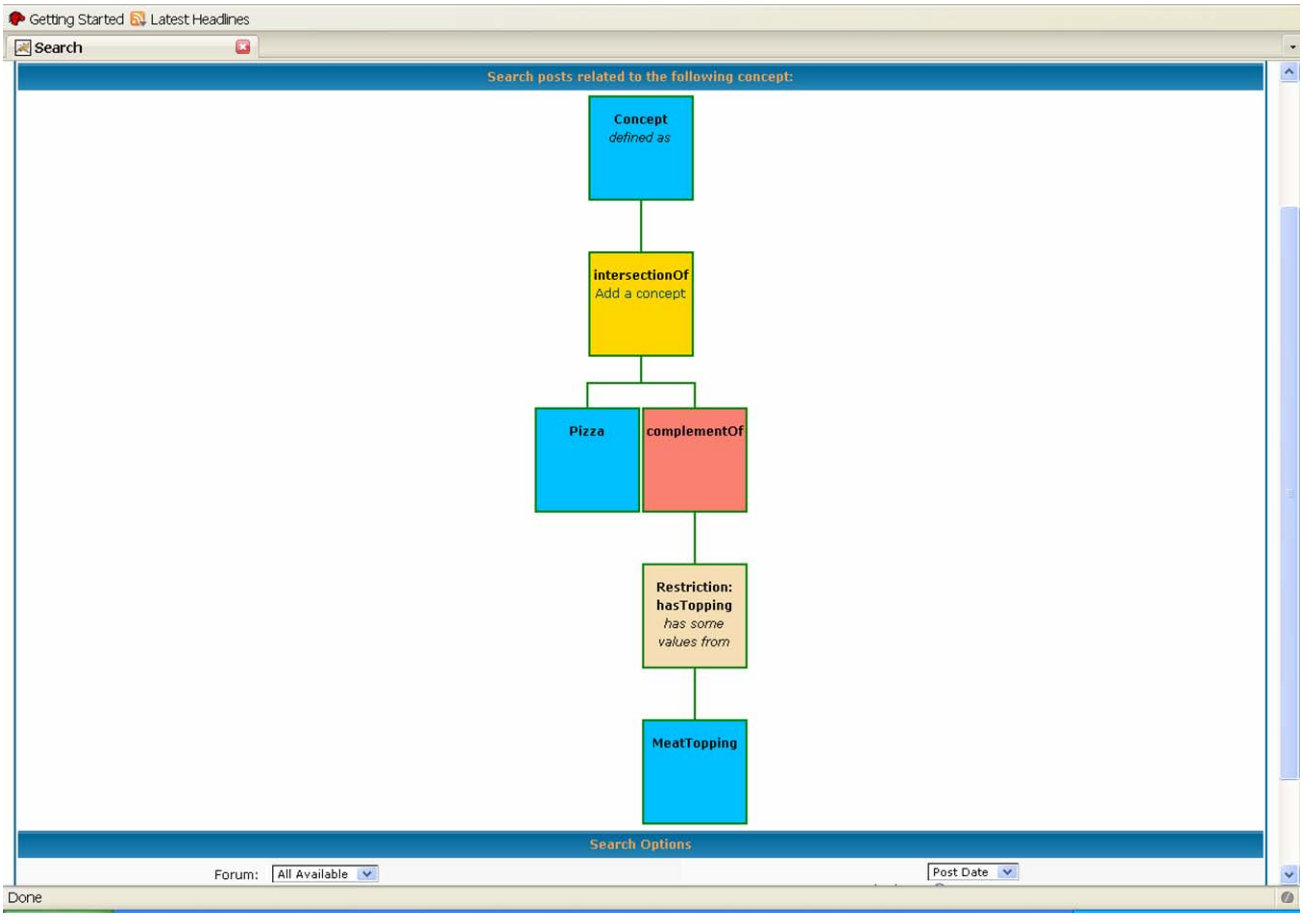

Figura 6 Arborele de interogare folosit pentru a defini conceptul „pizza vegetariană”, folosind vocabularul OWL-DL, va fi serializat sub formă de XML şi trimis către server

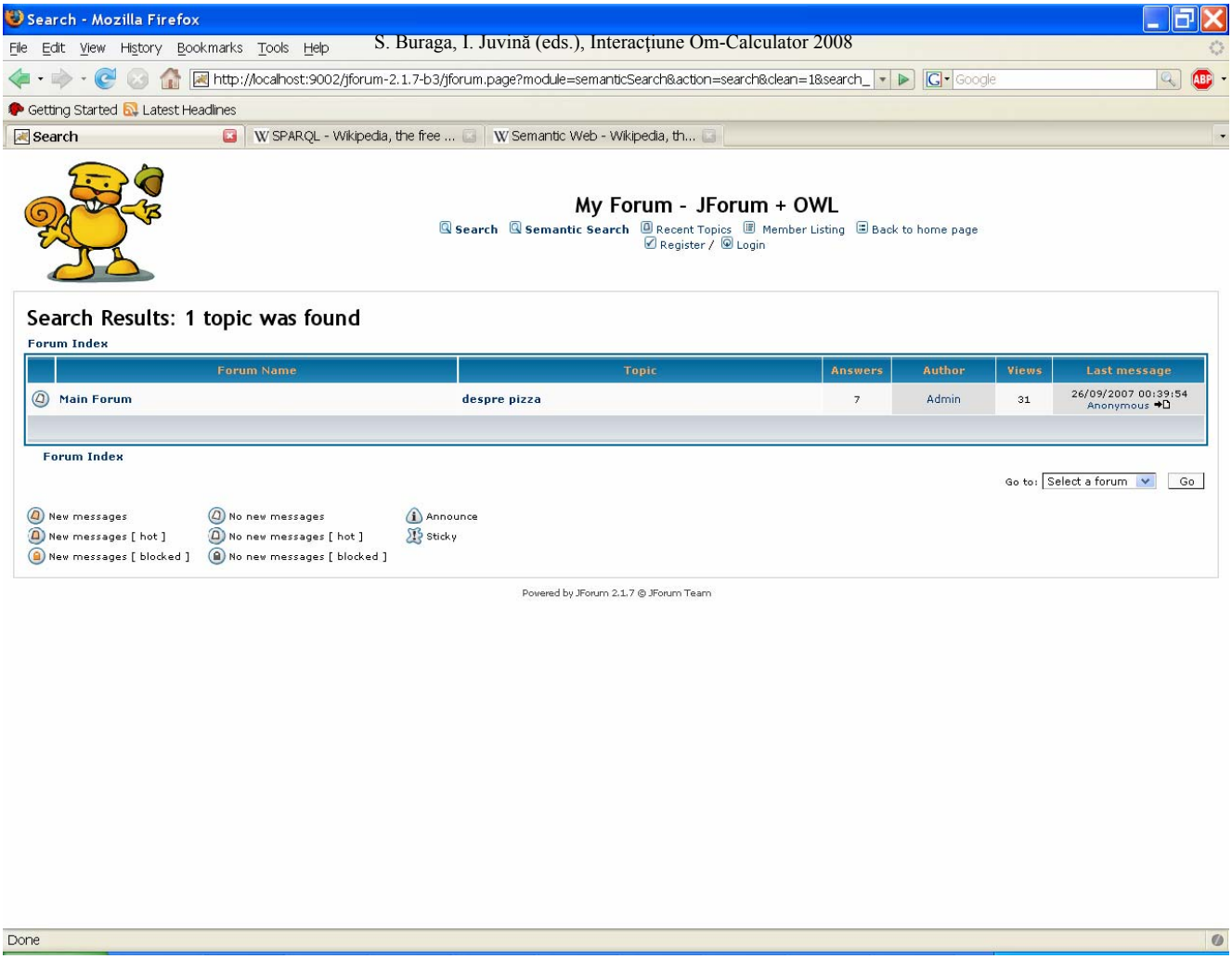

Figura 7 Rezultatul căutării

Construirea arborelui de interogare se face sub îndrumarea calculatorului: la adăugarea unui nou element, apare o fereastră de dialog în care utilizatorul specifică ce element doreşte să creeze, şi apoi caracteristicile acestuia. Programul nu permite utilizatorului să creeze o interogare greşită sintactic, nici să trimită o cerere incompletă.

La alegerea conceptelor de tip clasă este afişat un arbore al claselor existente în ontologie, ierarhizate corespunzător. La alegerea unei proprietăţi, se afişează o listă cu proprietăţile existente în ontologie. Aceste două structuri de date sunt preluate de la server asincron, folosind Ajax.

La trimiterea cererii, se verifică dacă arborele este completat în întregime, şi în caz afirmativ se construieşte o reprezentare în XML a acestuia, într-un format intern aplicaţiei. Când este primit de server, el este convertit în definiţia unei clase numită ForumSearchConcept. Serverul se foloseşte apoi de reasonerul instalat pentru a obţine o listă cu clasele echivalente şi subclasele lui ForumSearchConcept, după care acesta este eliminat din ontologie. Folosind lista de clase, se generează o listă de marcaje de forma „numeOntologie:cuvântCheieClasă”, care sunt apoi folosite pentru căutarea mesajelor.

REZULTATE

Aplicaţia web creată a fost testată folosind ca server web Tomcat 5.5, ca browser Firefox 2 şi Internet Explorer 6. S-au efectuat teste folosind atât o ontologie inclusă în Protégé OWL, numiă Pizza, cât şi o ontologie nou creată, Animal.

Rezultatele au fost satisfăcătoare şi au îndeplinit cerinţele propuse: aplicaţia creată a oferit posibilitatea căutării mesajelor fără a folosi cuvinte cheie conţinute în mesajele respective. În plus, posibilitatea definirii de concepte ca intersecţii sau reuniuni de concepte existente, sau impunerea de restricţii asupra proprietăţilor acestora au funcţionat şi s-au dovedit a fi mult superioare funcţionalităţilor textuale.

Interfaţa grafică pentru construcţia definiţiei conceptului care se caută este destul de uşor de folosit, chiar pentru utilizatori care nu au primit decât o explicaţie sumară despre modul de funcţionare.

CONCLUZII ŞI DEZVOLTĂRI ULTERIOARE

Din punct de vedere al aspectului interfeţei, extensia realizată s-a dovedit prezentabilă, uşor de folosit şi eficientă. Totuşi, ar fi util un mic tutorial vizual care nu numai să prezinte noilor utilizatori cum se foloseşte funcţia de căutare semantică, ci şi să îi convingă cât de puternică poate fi aceasta.

Din punct de vedere al implementării, s-au atins în întregime scopurile propuse. Căutarea semantică funcţionează corespunzător şi eficient, la fel ca şi adnotarea automată a mesajelor. În privinţa adnotării, se pot face îmbunătăţiri în cadrul funcţiei de identificare a claselor din ontologie pe baza textului; de exemplu, s-ar putea folosi un dicţionar de sinonime sau chiar o aplicaţie ca WordNet pentru a determina corespondenţe între sintagme din text şi clase din ontologie.

De asemenea, s-ar putea realiza o funcție de căutare care să combine căutarea semantică şi cea textuală, printr-o metodă de filtrare a rezultatelor.

Aceste obiective vor fi considerate la realizarea unor dezvoltări viitoare.

REFERINȚE

1. Semantic web - Wikipedia, the free encyclopedia. http://en.wikipedia.org/wiki/Semantic_web
2. OWL Web Ontology Language Overview. http://www.w3.org/TR/owl-features/
3. Owl - Wikipedia, the free encyclopedia. http://en.wikipedia.org/wiki/Owl
4. W3C's RDF at W3C. http://www.w3.org/RDF/
5. Resource Description Framework- Wikipedia, the free encyclopedia. http://en.wikipedia.org/wiki/Resource_Description_Framework
6. Protege Ontology Editor. http://protege.stanford.edu/
7. Protégé-OWL API Programmer's Guide. http://protege.stanford.edu/plugins/owl/api/guide.html
8. Introduction to Semantic Web, Ivan Herman, International Conference on Dublin Core and Metadata Applications, Singapore. http://www.dc2007.sg/
9. State of the Semantic Web, Ivan Herman, Semantic Days 2007, Stavanger, Norway. http://www.w3.org/2007/Talks/0424-Stavanger-IH/
10. Introduction to the Semantic Web, Ivan Herman, Semantic Days 2007, Stavanger, Norway. http://www.w3.org/2007/Talks/0423-Stavanger-IH/
11. SPARQL - Wikipedia, the free encyclopedia. http://en.wikipedia.org/wiki/SPARQL