TEHNOLOGII SEMANTIC WEB ÎN MEDIUL EDUCAȚIONAL

Teză destinată obţinerii
titlului ştiinţific de doctor inginer
la
Universitatea ”Politehnica” din Timişoara
în domeniul INGINERIE ELECTRONICĂ
ŞI TELECOMUNICAŢII
de către

Ing. Bogdan-Cătălin Drăgulescu

Conducător ştiinţific:	prof.univ.dr.ing. Radu VASIU
Referenţi ştiinţifici:	prof.univ.dr.ing. Aurel VLAICU
	prof.univ.dr.ing. Mihai ROMANCA
	prof.univ.dr.ing. Vladimir-Ioan CREȚU

Ziua susţinerii tezei: 15.12.2012

Seriile Teze de doctorat ale UPT sunt:

1. Automatică
2. Chimie
3. Energetică
4. Ingineria Chimică
5. Inginerie Civilă
6. Inginerie Electrică
7. Inginerie Electronică şi Telecomunicaţii
8. Inginerie Industrială
9. Inginerie Mecanică
10. Ştiinţa Calculatoarelor
11. Ştiinţa şi Ingineria Materialelor

Universitatea „Politehnica” din Timişoara a iniţiat seriile de mai sus în scopul diseminării expertizei, cunoştinţelor şi rezultatelor cercetărilor întreprinse în cadrul şcolii doctorale a universităţii. Seriile conţin, potrivit H.B.Ex.S Nr. 14 / 14.07.2006, tezele de doctorat susţinute în universitate începând cu 1 octombrie 2006.

România, 300159 Timişoara, Bd. Republicii 9,
tel. 0256 403823, fax. 0256 403221
e-mail: editura@edipol.upt.ro

Cuvânt înainte

Teza de doctorat a fost elaborată pe parcursul activităţii mele în cadrul Departamentului de Comunicaţii al Facultăţii de Electronică şi Telecomunicaţii, Universitatea ”Politehnica” din Timişoara.

Lucrarea este dedicată unui domeniu de interes şi cu un potențial de dezvoltare semnificativă pentru următorii ani: utilizarea tehnologiilor semantic web în învăţământul electronic, cunoscut sub numele de e-Learning. Se acordă o atenţie deosebită metodologiei de proiectare și implementare a unei ontologii în scopuri educaționale. În ultimii ani e-Learning-ul a prins un avânt considerabil, existând din ce în ce mai mult material educațional disponibil studenților. Acest lucru impune găsirea unei soluții pentru a ușura accesul la materialele educaționale, prin filtrarea si relaționarea acestora.

Prezenta lucrare cuprinde o serie de studii critice cu privire la stadiul actual al tehnologiilor semantic web, obiecte educationale utilizate in sisteme e-Learning, respectiv a diferitelor unelte semnatic web pentru platforme educaționale. Rezultatele obţinute au fost utilizate pentru a stabili direcţia ulterioară în cadrul activităţii doctorale. Aceasta s-a concretizat prin propunerea unui vocabular pentru dezvoltarea ontologiei educaționale, modele de unelte bazate pe tehnologii semantic web (publicarea de conținut în format RDF, partajarea de informații între două aplicații educaționale, generarea automată de teste folosind o bancă de întrebări în format RDF). Tehnologiile abordate, uneltele corespunzătoare şi interpretarea rezultatelor sunt detaliate pe parcursul tezei.

În încheiere doresc să aduc muţumiri familiei, colegilor şi conducătorului de doctorat, prof.dr.ing. Radu Vasiu, pentru răbdarea, sprijinul şi înţelegerea arătate în perioada anilor aferenţi cercetării şi elaborării prezentei lucrări.

Timişoara, 15.12.2012 Bogdan Drăgulescu

Pentru Alex.

Drăgulescu, Bogdan-Cătălin

Tehnologii Semantic Web în mediul educațional

Teze de doctorat ale UPT, Seria 7, Nr. 57, Editura Politehnica, 2012, 130 pagini, 31 figuri, 6 tabele.

ISSN: 1842-7014

ISBN: 978-606-554-585-4

Cuvinte cheie:
Semantic web, e-Learning, obiecte educaționale, ontologii semantic web, LMS.

Rezumat,

Teza de doctorat este dedicată unui domeniu de interes şi cu o dezvoltare semnificativă în ultimii ani - învăţământul prin intermediul mijloacelor electronice şi al Internet-ului, cunoscut sub numele de e-Learning. Se acordă o atenţie deosebită tehnologiilor semantic web care îmbunătăţesc în mod direct sau indirect calitatea educaţiei online.

Rezultatele obţinute dintr-o serie de studii critice cu privire la stadiul actual şi de perspectivă al e-Learning-ului şi al tehnologiilor semantic web în e-Learning sau cotidian au fost utilizate pentru a stabili direcţia ulterioară în cadrul activităţii doctorale.

Aceasta s-a concretizat prin propunerea unui vocabular pentru dezvoltarea ontologiei educaționale, utilizarea ontologiei în diferite modele de unelte bazate pe tehnologii semantic web si implementarea acestor modele în cadrul campusului virtual din cadrul UPT.

Testarea implementărilor menţionate anterior a fost realizată din diferite perspective, demonstrând gradul de utilitate al noilor tehnologii în activitatea didactică și a studenţilor.

Tehnologiile abordate, aplicaţiile corespunzătoare, analizele efectuate, contribuţiile teoretice şi aplicative sunt menţionate pe parcusul tezei.

CUPRINS

LISTĂ DE FIGURI

LISTĂ DE TABELE

LISTĂ DE NOTAŢII ŞI ACRONIME

HTML	- Hypertext Markup Language
SW	- Semantic Web
CMS	- Content Management System
LMS	- Learning Management System
LCMS	- Learning Content Management System
VLE	- Virtual Learning Environment
CEL	- Centrul de eLearning
XML	- Extensible Markup Language
RSS	- Really Simple Syndication
W3C	- World Wide Web Consortium
RDF	- Resource Description Framework
OWL	- Web Ontology Language
RDFS	- Resource Description Framework Schema
API	- Application Programming Interface
ODBC	- Open Database Connectivity
JDBC	- Java Database Connectivity
SQL	- Structure Query Language
URI	- Uniform Resource Identifier
SPARQL	- Simple Protocol and RDF Query Language
RIF	- Rule Interchange Format
SWRL	- Semantic Web Rule Language
SMW	- Semantic Media Wiki
GO	- Gene Ontology
RDFa	- Resource Description Framework in attributes
FOAF	- Friend of a Friend
SIOC	- Semantically Interlinked Online Communities
SKOS	- Simple Knowledge Organization System
RDQL	- RDF Data Query Language
SeRQL	- Sesame RDF Query Language
WSDL	- Web Services Description Language
SPARUL	- SPARQL Update
GRDDL	- Gleaning Resource Descriptions from Dialects of Languages
eRDF	- Embedded RDF
MOODLE	- Modular Object-Oriented Dynamic Learning Environment
SCORM	- Shareable Content Object Reference Model
SCO	- Shareable Content Object
ADL	- Advanced Distributed Learning
LOM	- Learning Object Metadata
IMS	- Instructional Management System
CC	- Common Cartrige
LTI	- Learning Tools Interoperability
DC	- Dublin Core
DCMI	- Dublin Core Metadata Initiative
LTSC	- Learning Technology Standard Committee
ARIES	- Advanced Research in Intelligent Educational Systems
P2P	- Peer to Peer
QR	- Quick Response

UML - Unified Modeling Language
WSML - Web Service Modeling Language
LOD - Linked Open Data

1 Motivația

In capitolul introductiv am realizat o prezentare a principalelor considerente cu privire la tematica de cercetare aleasă. Am argumentat actualitatea temei, am descris premisele inițiale de cercetare şi am prezentat pe scurt structura tezei pe capitole, dar şi lista lucrărilor ştiinţifice publicate pe parcursul activităţii de cercetare.

1.1 Considerații generale cu privire la tema aleasă

Dezvoltarea și utilizarea pe scară largă a tehnologiilor internet a permis dezvoltarea de variante electronice pentru majoritatea serviciilor clasice (poștă electronică, publicații disponibile online, magazine de vânzare cu amănuntul, etc.). Era natural ca acest proces să cuprindă și activitățile educaționale. Astfel, au fost dezvoltate sisteme electronice care să ofere suport în procesul didactic, folosind ca mediu de comunicare Internetul.

Accelerarea dezvoltării sistemelor de acest tip este încurajată de către organizații care în mod tradițional oferă programe educaționale în regim la distanță. Pentru acestea, încorporarea sistemelor electronice educaționale este un pas logic pentru a extinde activitățile educaționale la distanță din repertoriul lor. De asemenea, companiile manifestă un interes crescut pentru tehnologiile e-Learning, pentru a reduce costurile cu instruirea personalului. Universitățile care oferă programe educaționale față în față, utilizează tehnologii de tip e-Learning pentru a îmbunătăți accesul cursanților la materialele educaționale, pentru a îmbunătăți suportul personalizat acordat studenților în vederea creșterii ratei de promovabilitate, precum și pentru a se promova pe piețe internaționale [1].

Pentru a oferi suport electronic în procesul didactic este nevoie de anumite unelte software care să fie capabile să ofere serviciile necesare. Acestea pot fi împachetate în aceeași platformă, oferind o soluție completă, sau pot fi o serie de unelte electronice disponibile gratuit sau contra cost, pe care tutorele decide să le utilizeze în procesul educațional. Pentru o organizație ce oferă un volum mare de cursuri este pretabilă prima variantă. Astfel au fost introduse platformele de administrare educaționale **LMS** (Learning Management System) [2]. Există variante în sursă deschisă (cel mai cunoscut fiind Moodle), și variante comerciale (monopolul fiind deținut de BlackBoard). Fiecare dintre aceste două abordări prezintă avantaje și dezavantaje, platforma ideală pentru o organizație fiind determinată de nevoile proprii ale acesteia.

Dintre beneficiile aduse prin utilizarea platformelor educaționale amintesc următoarele [3]: reducerea costurilor, inclusiv pentru întâlnirile față în față;

reducerea costurilor cu echipamentele didactice; îmbunătățirea reputației instituției; studenți mai mulțumiți; îmbunătăţirea procesului didactic; creșterea numărului de cursanți ce pot fi înscriși într-un program; distribuția mai rapidă a materialelor educaționale; etc.

În cadrul activității mele de cercetare, am fost implicat în utilizarea de platforme educaționale în proiecte de cercetare (ViCaDiS – Virtual Campus for Digital Students), iar mai apoi în realizarea Campusului Virtual al Universității „Politehnica" din Timişoara, care oferă suport pentru toate programele oferite la distanță, pentru ciclul de master și o parte din cursanții de la licență, învățământ cu frecvență. De asemenea, am dezvoltat și utilizat în procesul didactic o unealtă de laborator virtual pentru cursul de Baze de Date.

Pe parcursul acestor activități am observat următoarele neajunsuri: datele disponibile în platformă sunt publicate în formate destinate doar oamenilor; agenții software care vor să citească informațiile vor pierde semnificația datelor; partajarea informațiilor între diferite platforme și unelte educaționale este îngreunată datorită necesității dezvoltării de module specifice care să îndeplinească această funcție. Datorită acestor limitări anumite scenarii educaționale și procese administrative sunt greu de implementat, fiind necesară reproiectarea lor în cazul actualizării platformelor educaționale.

În acest context, în cadrul activității de doctorat mi-am propus să identific o metodă de partajare a informațiilor fără a pierde semnificația acestora. Mi-au atras atenția noile tehnologii informaționale, care transformă Internetul actual într-o versiune în care datele sunt păstrate nu doar pentru utilizatorii umani, ci și pentru a asigura *lizibilitatea* acestora de către agenți software. Aceste tehnologii poartă numele de Semantic Web.

Pentru a mă asigura că este o soluție viabilă, mi-am propus să evaluez gradul de standardizare și maturitate a acestor tehnologii pentru a putea fi utilizate în platforme educaționale. Am analizat structura sistemelor educaționale pentru a identifica datele care ar fi utile pentru modelare. De asemenea, am evaluat principalele specificații utilizate pentru descrierea obiectelor educaționale și pentru introducerea de meta-date destinate agenților software.

A urmat apoi procesul de transpunere a conceptelor identificate în modele semantice, urmărind specificațiile unei metodologii și validarea acestora prin utilizarea în aplicații practice.

Tehnologiile Web-ului Semantic sunt în curs de dezvoltare, având un potențial mare pentru introducerea de facilități noi Internetului actual. Din acest motiv, mulți cercetători și-au dedicat activitatea științifică acestui domeniu, în direcții precum: introducerea și standardizarea conceptelor de bază [4] [5], construcția de unelte necesare în aplicații semantic web [6] [7] sau în aplicații specifice cum este învățământul electronic [8] [9] [10].

În concluzie, premisele inițiale de cercetare le pot sintetiza sub forma unor întrebări de cercetare, în modul următor:

1. Care este nivelul actual de standardizare a tehnologiilor Web-ului Semantic? Cum pot fi utilizate în realizarea de aplicații web îmbunătățite?
2. Ce tipuri de date sunt păstrate în platformele educaționale? Se pot reutiliza specificațiile de inserare de meta-date în interiorul obiectelor educaționale?
3. Cum se pot insera meta-date direct în structura existentă a LMS-urilor?

4. Ce concepte sunt necesare în modelarea datelor educaționale și cum pot fi ele formalizate?
5. În ce măsură se pot utiliza aceste modele pentru a crește calitatea serviciilor educaționale?

1.2 Articole publicate

Activitatea mea de cercetare s-a concretizat într-o serie de lucrări științifice publicate sau în curs de publicare, ca autor sau co-autor:

Iasmina Ermalai, **Bogdan Dragulescu**, Andrei Ternauciuc, Radu Vasiu, "Building a Module for Inserting Microformats into Moodle", Advances in Electrical and Computer Engineering Journal, ISSN: 1582-7445 (revistă ISI în curs de publicare)

Bogdan Drăgulescu, Marian Bucos, Radu Vasiu, "A Semantic Web approach for automated test generation", Proceedings of IADIS International Conference WWW/INTERNET 2012, Madrid, Spania, 2012, ISBN: 978-989-8533-09-8, pp. 235-241

Bogdan Drăgulescu, Iasmina Ermalai, Marian Bucos, Radu Vasiu, "Metadata Methods for Improving Usability in Moodle", International Journal of Web Engineering, 2012, doi: 10.5923/j.web.20120101.02, pp. 6–10

Mugur Mocofan, Iasmina Ermalai, Marian Bucos, Mihai Oniţa, **Bogdan Drăgulescu**, "Supervised tree content based search algorithm for multimedia image databases", Proceedings of 6th IEEE International Symposium on Applied Computational Intelligence and Informatics (SACI), Timișoara, România, 2011, ISBN: 978-1-4244-9107-0, pp. 469–472 (în curs de indexare ISI Proceedings)

Bogdan Drăgulescu, Iasmina Ermalai, Marian Bucos, Mugur Mocofan, "Using hCard and vCard for improving usability in Moodle", Proceedings of 6th IEEE International Symposium on Applied Computational Intelligence and Informatics (SACI), Timișoara, România, 2011, ISBN: 978-1-4244-9107-0, pg. 473–476 (în curs de indexare ISI Proceedings)

Marian Bucos, **Bogdan Drăgulescu**, Marius Velțan, "Designing a semantic web ontology for E-learning in higher education", Proceedings of 9th International Symposium on Electronics and Telecommunications (ISETC), Timișoara, România, 2010, ISBN: 978-1-4244-8460-7, pp. 415–418 (indexat ISI Proceedings)

Iasmina Ermalai, **Bogdan Drăgulescu**, "The usefulness and functionality of microformats in a particular eLearning system", Proceedings of Computational Cybernetics and Technical Informatics (ICCC-CONTI), Timișoara, România, 2010, ISBN: 978-1-4244-7431-8, pp. 387–390

Diana Andone, Radu Vasiu, Andrei Ternauciuc, **Bogdan Drăgulescu**, "The use of social media tools in ViCaDiS Virtual Campus", Proceedings of International Joint Conference on Computational Cybernetics and Technical Informatics (ICCC-CONTI), Timișoara, România, 2010, pp. 305–310

Mihai Oniţa, Iasmina Ermalai, Andrei Ternauciuc, **Bogdan Dragulescu**, *"Media Streaming in Higher Education"*, Proceedings of Iadis International Conference "Cognition and Exploratory Learning in Digital Age", Roma, Italia, 2009, ISBN: 978-972-8924-95-9, pp. 373-377

Mihai Onita, Andrei Ternauciuc, **Bogdan Dragulescu**, Iasmina Ermalai, *"Streaming Solutions at UPT",* Proceedings of the 5th International Scientific

Conference ELSE - E-Learning and Software for Education, Editura „Universitatea Nationala de Aparare Carol I", București, România, 2009, ISSN 2066-026X, pp. 151-157

Andrei Ternauciuc, **Bogdan Dragulescu**, Mihai Oniță, Radu Vasiu, "*Single sign-on solutions for Moodle",* Proceedings of the 5th International Scientific Conference ELSE - E-Learning and Software for Education, Editura „Universitatea Națională de Apărare Carol I", București, România, 2009, ISSN 2066-026X, pp. 217-225

Marian Bucos, **Bogdan Drăgulescu**, Andrei Ternauciuc, "Developing virtual labs at 'Politehnica' University of Timișoara", Proceedings of International Conference "Virtual University" 2008, Bratislava, Slovakia, 2008, ISBN: 978-80-89316-10-6

1.3 Structura tezei de doctorat

Am structurat teza de doctorat în șase capitole, la care se adaugă bibliografia utilizată.

Capitolul 1 reprezintă capitolul introductiv, în care am stabilit motivația care a determinat realizarea activităților de cercetare descrise în prezenta teză de doctorat. Am menționat considerentele generale cu privire la tema aleasă, subliniind actualitatea temei și definind întrebările fundamentale pentru stabilirea direcției de cercetare. Capitolul cuprinde și lista de lucrări publicate, sau în curs de publicare, precum și o prezentare succintă a capitolelor cuprinse în prezenta lucrare.

Capitolul 2 introduce conceptele de bază în Web-ul Semantic, subliniind necesitatea apariției acestei tehnologii. Sunt identificate componentele și tehnologiile necesare implementării acestor concepte, precizând gradul lor de standardizare. Chiar dacă straturile superioare sunt momentan doar la stadiul de proiect, am argumentat totuși posibilitatea construirii de platforme educaționale îmbogățite semantic, prin identificarea de aplicații care au început să pună în practică paradigmele Web-ului Semantic. Scopul acestui capitol a fost de a identifica limitările curente și direcțiile viitoare de dezvoltare, precum și maturitatea acestor tehnologii, pentru a putea fi utilizate în platforme educaționale unde calitatea serviciilor oferite trebuie să fie ridicată.

Capitolul 3 conține o analiză a principalelor concepte utilizate în învățământul electronic și implicit în platformele de gestionare a conținutului educațional. Este prezentată structura de organizare a unui LMS, identificând patru secțiuni: cursuri, activități, evaluare și utilizatori. Sunt descrise structurile obiectelor educaționale de tip SCORM și IMS, precum și specificațiile de introducere de meta-date (IEEE LOM). Capitolul cuprinde o primă abordare de a îmbunătăți serviciile oferite într-un LMS, prin utilizarea microformatelor.

Capitolul 4 cuprinde o propunere de model de ontologie educațională și mecanismele prin care sunt legate elementele pe care acest model le conține. Am analizat metodologiile de proiectare ontologică, evaluând diferențele dintre acestea pentru a o alege pe cea mai potrivită situației de față. Termenii incluşi în ontologie au fost trecuţi printr-un proces de conceptualizare, definire a specificațiilor și formalizare, având în vedere pe cât posibil reutilizarea de vocabulare și ontologii

standardizate. Implementarea termenilor a fost realizată utilizând limbajele RDFS și OWL.

Capitolul 5 prezintă trei aplicații practice de utilizare a ontologiei propuse în capitolul precedent. Prima aplicație reprezintă o metodă de republicare a datelor păstrate în baza de date Moodle în format RDF, utilizând un fișier de mapare. Astfel se asigură partajarea datelor pentru utilizarea lor de agenți software semantici. A doua aplicație propusă este *fuzionarea* datelor disponibile într-o platformă educațională cu datele dintr-o aplicație de tip laborator virtual, permițând astfel construcția de servicii și automatizarea extragerii datelor. În final am prezentat un sistem de generare automată de teste de evaluare folosind o bancă de întrebări modelată în formatul RDF.

Capitolul 6 conține principalele concluzii rezultate din prezenta teză de doctorat, o sinteză a principalelor contribuții teoretice și aplicative, precum și posibile direcții viitoare de cercetare.

2 Web-ul Semantic

În această parte am prezentat conceptele și tehnologiile fundamentale necesare înțelegerii lucrării de față. Capitolul face referire la stadiul actual al dezvoltării web-ului semantic: conceptele de bază, tehnologiile utilizate în implementare, arhitectura unui sistem semantic şi gradul lor de utilizare.

2.1 Introducere

Pentru a putea defini termenul de Web Semantic, trebuie sa definim cuvântul semantic. Conform dicționarului explicativ al limbii române, cuvântul semantic are următoarea definiție [11]:

"SEMÁNTIC, -Ă, semantici, -ce, s. f., adj. I. S. f. 1. Ramură a lingvisticii care se ocupă cu studierea sensurilor cuvintelor și a evoluției acestor sensuri; semasiologie, semantism. 2. (Log.) Teoria interpretării unui anumit sistem formalizat prin alt sistem formalizat. II. Adj. Care ține de semantică (I 1), care se referă la sensurile cuvintelor; semasiologic. – Din fr. sémantique."

Astfel, prin semantic înțelegem sensul unui cuvânt. Putem considera Web-ul ca o sursă vastă de informații de cele mai multe ori necatalogate într-un mod ușor de înțeles pentru un agent software. Web-ul Semantic se referă la o modalitate de a publica informațiile într-o formă structurată în care un agent software ar putea ușor să identifice "sensul" datelor respective.

De exemplu, în momentul de față informația publicată în paginile web este formatată utilizând tag-uri HTML. Semantic, putem să deducem faptul că informația încapsulată de tag-ul <H1> prezintă o importanță sporită pentru cititor, față de restul informațiilor, datorită sensului tag-ului respectiv. În plus, unele pagini web adaugă informație suplimentară destinată motoarelor de căutare, numită meta-informație, dar aceasta conţine doar cuvinte cheie fără a defini "sensul" și contextul

informației conținute de documentele HTML. Într-un mod similar, bazele de date pot sugera sensul informației conținute dacă denumirile tabelelor și coloanelor sunt alese corect [12].

Aceste metode de publicare a informației nu sunt suficiente pentru a permite descrierea semnificației datelor conținute în documente. Astfel, a apărut necesitatea utilizării unui nou concept, Web-ul Semantic. Acesta a fost introdus de către Tim Berners Lee în articolul *The Semantic Web* din anul 2001 [4], fiind conceput ca o extensie a Web-ului actual, care permite în plus descrierea sensului datelor publicate.

În continuarea acestui subcapitol, voi prezenta modul de percepție a informației de către oameni și cum este aceasta utilizată de către agenți software, subliniind dificultățile pe care aceștia din urmă le au în extragerea informației, dar și evoluția Web-ului actual către un Web inteligent,precum și modelele de reprezentare a datelor.

2.1.1 Perceperea informației

În comunicarea dintre două persoane, limbajul natural este extraordinar. Fără un efort considerabil putem solicita o informație unei persoane, să ne expunem ideile cu privire la calitatea unui film sau să învățăm dintr-o carte scrisă în urmă cu 20 de ani. Este greu de imaginat o unealtă mai bună pentru a distribui informația.

Cea mai simplă formă gramaticală de a transmite informația în limbaj natural este "subiect – predicat - obiect". Să luăm ca și exemplu două propoziții care respectă forma precedentă:

Anca savurează tomatele.
Tomatele îl dezgustă pe Andrei.

Fiecare dintre aceste două fraze transmit o informație. Cuvintele Anca și Andrei se referă la anumiți oameni, cuvântul tomate se referă la un fruct, iar cuvintele savurează și dezgustă definesc relația dintre persoană și fruct. Datorită faptului că pe baza cunoștințelor anterioare înţelegem sensul verbelor a savura și a dezgusta, precum și datorită faptului că fiecare dintre noi a văzut o tomată, putem să înțelegem informația conținută în cele două propoziții. O dată citite cele două propoziții ne furnizează informație nouă despre lumea înconjurătoare. Acesta este un exemplu de semantică: simbolurile se referă la obiecte sau concepte, iar o înșiruire de simboluri transmit o semnificație. Folosind semnificația dedusă din cele două propoziții putem răspunde la o întrebare simplă, cum ar fi: *Cui ii plac tomatele?*

Acest model de funcționare în comunicarea dintre oameni, folosind limbaj natural, s-ar putea aplica și la schimbul de informație într-o rețea de calculatoare, fie că aceasta este o rețea locală sau globală (World Wide Web). Folosind logica descrisă mai sus, programele dezvoltate într-un astfel de sistem ar fi capabile să "fabrice" informație nouă pe baza seturilor de date disponibile [13].

Odată cu apariția World Wide Web ca și metodă principală de schimb de date între două calculatoare, structurarea informației transmise între client și server este ascunsă utilizatorului. Acest lucru se datorează decuplării transmisiei datelor de interfața aplicației, ducând la dezvoltarea sistemelor de gestiune a datelor.

Cu toate că această separare a dus la dezvoltarea accelerată a aplicațiilor web, ea vine și cu efecte secundare negative; datele publicate într-o aplicație nu pot fi partajate, interconectate sau integrate în alte aplicații.

Să presupunem că investigăm sistemul nostru solar și am găsit o pagină web cu informații cuprinzătoare despre obiectele din sistemul solar: stele, planete, sateliți, asteroizi, comete. Fiecare obiect are propria pagină web, cu fotografii și informații esențiale (masă, distanța de la soare, formă, mărime, ce obiecte se învârt în jur, perioada de rotație, perioada de revoluție, etc.). În capul paginii se găsește categoria obiectului: planetă, lună, asteroid, cometă. Altă pagină conține liste de obiecte interesante: sateliți ai lui Jupiter, obiecte din centura de asteroizi, planete care se învârt în jurul Soarelui. Această ultimă pagină conține cele 9 planete cu legătura către paginile proprii. Într-o zi citim în ziar că International Astronomical Union (IAU) a decis că Pluto nu va mai fi considerată planetă, ci va fi inclusă în noua categorie de "planete pitice". Verificăm informația pe pagina lui Pluto și constatăm că modificarea a fost făcută: Pluto este o planetă pitică. Dar când mergem la pagina planetelor din sistemul Solar, constatăm ca Pluto este încă listatată ca o planetă. Informația în acest caz este inconsecventă, nesincronizată și deconectată. Ce trebuie să facem pentru a avea un web mai inteligent? Avem nevoie de aplicații inteligente sau infrastructură web inteligentă?

Internetul este plin de aplicații inteligente: motoare de căutare care returnează rezultate profunde și intuitive, site-uri de comerț electronic care fac recomandări inteligente personalizate folosind șabloanele de cumpărare ale fiecărui client, site-uri de cartografiere care includ informații geografice extinse și pot planifica rute și calcula distanțe. Orice tehnologie informațională se poate utiliza într-o aplicație web într-un mod inteligent.

Chiar și cea mai "perspicace" și inteligentă aplicație este limitată de calitatea informației accesibilă ei. Chiar și cele mai inteligente aplicații, dacă utilizează informație de intrare inconsecventă sau contradictorie, va genera date confuze, deconectate, stupide. De aici vine necesitatea utilizării unei infrastructuri web "inteligente" din punct de vedere al integrării informațiilor pe Web. Aceasta este provocarea ce trebuie rezolvată de Semantic Web - să obțină date corecte și să le distribuie la locul potrivit, astfel încât aplicațiile inteligente să poată face munca lor.

Revenind la exemplul de mai sus, pagina web cu informații astronomice, am dori ca informația să fie actualizată într-un mod consecvent. Dacă declarăm că Pluto nu mai este planetă, atunci pagina cu lista de planete ar trebui să reflecte acest lucru. O soluție în acest caz este utilizarea unei baze de date care să păstreze informațiile. În schimb, dacă dorim ca această informație să fie disponibilă și altor aplicații, problema reapare datorită limitării accesului la baza de date. Deci, se păstrează necesitatea unei infrastructuri web capabile să obțină și să distribuie date corecte [14][p 1-6].

Web-ul Semantic, așa cum este el imaginat de W3C, va permite în plus construirea unui graf de date global prin interconectarea tuturor datelor semantice. Pentru a realiza acest tip de structură, a apărut comunitatea Linked Open Data, cu scopul declarat de a dezvolta cele mai bune practici pentru publicare și distribuire de date semantice.

În timp ce o comunitate de publicare este necesară pentru a realiza graful global, la fel de importantă este comunitatea dezvoltatorilor de aplicații care folosesc această colecție de date distribuite pentru a demonstra valoarea unui astfel de graf. Astfel de aplicații vor putea interoga surse multiple de date, obținând o bucată critică de informație de la fiecare, care le va permite să interogheze următoarea sursă. Navigând prin această structură, până la sursa de date finală, aceste aplicații vor fi capabile să furnizeze informații pe care nu le-am putea obține de la nici unul din site-urile inițiale [13][p 105 - 116].

2.1.2 Modelarea datelor

Un model este o descriere simplificată a unor anumite aspecte din realitate, utilizate pentru a înțelege, structura sau prezice anumite părți din lumea reală. Crearea de modele face parte din capacitatea ființelor umane de a comunica și raționa. Modelele utilizate în comunicarea umană au avantajul capacității oamenilor de a interpreta simboluri și de a le înțelege sensul, aceste modele putând fi definite într-o gamă variată de forme folosind limbaj natural sau chiar imagini. Un model poate fi definit de o persoană, modificat de către o alta și interpretat de o a treia. Modele de comunicare au fost dezvoltate și pentru interacțiunea om-calculator sau calculator-calculator.

În arhitectura Internetului există modele standardizate de comunicare între client și server. Dacă un server web care ascultă protocolul HTTP primește o cerere GET, el știe să trimită datele găsite la adresa specificată în cerere către client. Această metodă de comunicare a permis dezvoltatorilor de aplicații web să se concentreze pe modul în care aplicațiile sunt vizualizate de utilizatori, ducând în acest mod la o decuplare între date și interfețele web. Astfel, datele sunt ascunse în spatele unor interfețe, împiedicând partajarea și integrarea acestora în alte aplicații internet, capabile să ofere o utilitate crescută.

Odată cu creșterea cantității informației utilizate în aplicații web, a fost necesară dezvoltarea unor modele de structurare a informaţiei. Sunt multe moduri în care se poate păstra informația, fiecare dintre ele optimizate pentru a îndeplini o anumită sarcină.

În cazul Web-ului Semantic trebuie utilizat un model care să faciliteze partajarea ușoară și interoperabilitatea datelor. Această problemă are două părți: sintactică și semantică. Din punct de vedere sintactic, partajarea datelor implică accesul la date, adică modul standardizat în care un sistem trimite o cerere și primește un răspuns. A doua parte, cea semantică, se referă la capacitatea de a încorpora datele extrase în structura informațională a sistemul care le consumă.

În continuare voi prezenta cele mai comune metode de modelare a datelor, reliefând punctele tari și slabe cu privire la integrarea datelor în aplicații web.

Date tabelare

Cele mai simple seturi de date, cunoscute aproape de toată lumea, sunt datele care utilizează modelul tabelar. Datele tabelare sunt date păstrate într-un tabel, cum ar fi un fișier de tip Excel sau un tabel HTML. Avantajele modelului tabelar reies din simplitatea cu care acest model de date a fost conceput. Datele cuprinse într-o astfel de structură sunt ușor de citit și manipulat.

Tabel 1. Exemplu model tabelar

Nume	Prenume	Anul nașterii	Materii studiate
Popescu	Ion	1984	Fizică, Matematică, Baze de date
Ionescu	Andrei	1986	Informatică, Baze de date

Nu există variații de modelare de la caz la caz, datele fiind structurate în linii și coloane. Chiar și această structurare simplistă a datelor păstrează informații despre date (meta-date). Poziționarea informației într-o anumită coloană sau anumit rând permite persoanei care citește datele să deducă sensul informației. De exemplu,

1986 prin faptul că se găsește pe același rând cu Ionescu ne spune că "persoana cu numele Ionescu s-a născut în anul 1986".

Datele păstrate în această formă au limitări evidente. În coloana *Materii studiate* se păstrează lista materiilor pe care persoana respectivă le studiază. Această structură păstrează corect datele până când dorim să introducem informații suplimentare, cum ar fi notele obținute la evaluare de respectivii studenți. În această situație, datele se pot introduce între paranteze în dreptul fiecărei materii, iar citirea informației se face cu dificultate atât de către oameni cât și de către agenții software.

Fișierele Excel, construite pe model de date tabelar, permit procesarea datelor. Chiar și datele expuse în exemplul precedent pot fi procesate prin utilizarea comenzilor macro. Modelul tabelar este rigid, greu de modificat și limitat din punct de vedere al partajării informației.

Date relaționale

Bazele de date relaționale au evoluat rapid in ultimele trei decenii, devenind modelul de bază de date cel mai utilizat în aplicațiile comerciale [15][p - 1]. Odată cu evoluția aplicațiilor web, de la simple pagini HTML spre aplicații cu conținut dinamic, datele au trebuit să fie separate de interfața grafică. Modelul relațional s-a impus datorită maturității sistemelor de gestiune bazate pe acest model, cum ar fi Oracle DB, MySQL sau PostgreSQL. Bazele de date relaționale sunt capabile să stocheze cantități imense de informație, oferind în același timp rapiditate în rularea interogărilor, scalabilitate, tranzacții, etc.

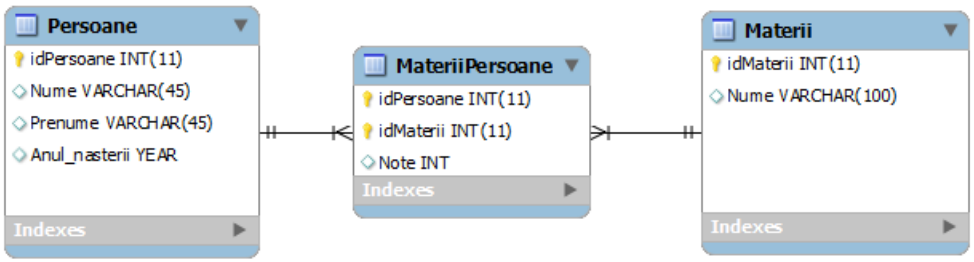
O bază de date relațională permite să se *unească* printr-o operație de tip JOIN informație din mai multe tabele într-un mod standardizat. Exemplul precedent se poate modela relațional ca în figura 1, datele fiind reprezentate într-un mod mai util. Astfel, față de modelul tabelar, unde doar operațiile de filtrare după conținutul coloanei se puteau implementa simplu, într-o bază de date relațională se pot utiliza interogări mai sofisticate, cum ar fi selectarea tuturor persoanelor care au promovat examenele cu note mai mari de opt. În plus, se observă că prin separarea tabelului inițial în entități s-au introdus meta-date suplimentare.

Proiectarea unei baze de date relaționale presupune cunoașterea a priori a datelor ce vor fi stocate și modul în care acestea vor fi utilizate. Dacă după implementare se modifică structura datelor, aceasta implică automat o reproiectare a bazei de date, migrarea datelor de pe schema veche pe cea nouă și modificarea interogărilor.

Odată ce structura datelor evoluează, schemele bazelor de date devin din ce în ce mai complicate, pentru a putea păstra diferitele entități de date. Astfel, în cazul în care dorim să adăugăm date externe pentru a putea construi interogări mai avansate, normalizarea bazei de date va fi din ce în ce mai complicată.

Din punct de vedere sintactic, partajarea informației este posibilă prin utilizarea de API-uri standardizare de interacțiune cu bazele de date (Open Database Connectivity – ODBC, Java Database Connectivity – JDBC, Structure Query Language - SQL). De exemplu, interogările scrise utilizând SQL pentru o bază de date pot fi ușor adaptate pentru un alt sistem de baze de date relaționale. Totuși, există încă dificultăți deoarece fiecare producător de sisteme de baze de date are altă implementare pentru API-urile respective, făcând necesară cunoașterea a priori a tipului de sistem interogat. Din punct de vedere semantic, problema încă persistă.

Bazele de date relaționale sunt indicate pentru a păstra cantități mari de informații într-un model de date care nu se modifică foarte des, partajarea și integrarea informației fiind dificilă.



Persoane

IdPersoane	Nume	Prenume	Anul_nasterii
1	Popescu	Ion	1984
2	Ionescu	Andrei	1986

MateriiPersoane

IdPersoane	IdMaterii	Note
1	1	7
1	2	8
1	3	9
2	4	10
2	3	9

Materii

idMaterii	Nume
1	Fizică
2	Matematică
3	Baze de date
4	Informatică

Figura 1. Exemplu de model relațional

Date RDF

În Web-ul Semantic datele sunt reprezentate sub forma unor afirmații compuse din trei părți: subiect, predicat și obiect. Datorită acestor trei părți, datele mai poartă numele de triplete. Cele trei elemente au sens asemănător în triplete cu elementele din limbajul natural. Subiectul este lucrul descris în afirmație, în timp ce predicatul descrie legătura între subiect și obiect.

Datele din exemplul precedent pot fi transpuse în triplete în următoarea formă:

Popescu *are prenumele* Ion.
Popescu *are anul nașterii* 1984.
Popescu *are materia* Fizică.

Aceste afirmații se pot reprezenta sub forma unui graf ca în figura 2, unde nodurile reprezintă subiecte sau obiecte, iar predicatele sunt muchiile. Direcția muchiei este dinspre subiect înspre obiect. Fiecare din cele trei componente ale unei triplete are ca și identificator o adresă URI, cu excepția obiectelor reprezentate printr-o valoare literală. În cazul în care obiectul este o valoare, el este reprezentat grafic sub forma unui dreptunghi. Acesta este modelul de date utilizat în Web-ul Semantic și este standardizat în limbajul Resource Description Framework (RDF) [16].

Grafurile nu au elemente rădăcină ca și în cazul documentelor XML, care sunt construite sub o structură ierarhică. Dacă vrem să unim două documente XML este dificil de decis care element va deveni rădăcina noului document. În grafurile RDF aceasta nu este o problemă, deoarece nu există o rădăcină, iar suprapunerea nodurilor se face pe baza identificatorului URI. Astfel, se pot uni grafuri din diferite surse de informație fără a pierde sensul datelor respective. Modelul RDF rezolvă problema semantică a partajării informației.

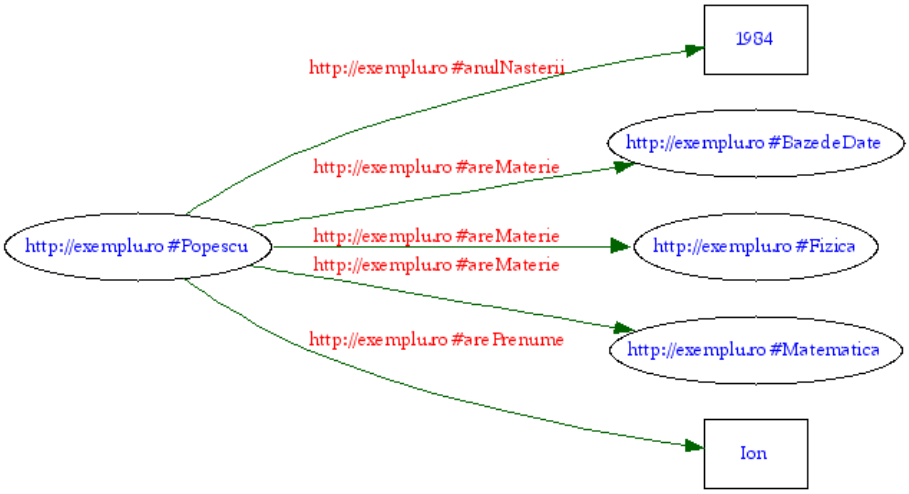


Figura 2. Exemplu de model graf RDF

Pentru cealaltă parte a problemei, partea sintactică, s-au definit standarde de extragere a datelor din grafuri RDF, și anume limbajul de interogare SPARQL. Modelul de date RDF rezolvă problema partajării ușoare și interoperabilitatea datelor, propunând un sistem relativ simplu de modelare a datelor, dar complet diferit de modelele clasice.

În concluzie, modelele de date clasice (tabelar și relațional) nu sunt utilizabile în Web-ul Semantic, deoarece nu permit partajarea ușoară și/sau interoperabilitatea datelor. Din acest motiv a fost introdus modelul grafului RDF, care rezolvă problema semantică a partajării informației.

2.1.3 Componentele sistemelor SW

Pentru a construi o aplicație SW se utilizează o serie de componente distincte. Pentru a putea înțelege arhitectura unui astfel de sistem aceste componente trebuie definite. Conform [12][p 10 – 15] și [14][p 59 - 77] acestea sunt împărțite în două categorii: componente de bază, utilizate pentru descrierea datelor în formate specifice SW, și unelte asociate SW, pentru interacțiunea cu datele respective. Din prima categorie fac parte afirmațiile SW, identificatorii URI, limbajele SW, ontologiile și datele.

Afirmațiile. Acestea formează fundația Web-ului Semantic. Fiecare afirmație este compusă din subiect, predicat și obiect, formând astfel o tripletă. Folosind aceste afirmații se pot defini structura informației, instanțele și constrângerile.

URI. Acronimul provine de la *Uniform Resource Identifier*, utilizat în Internet pentru a furniza un identificator unic pentru o resursă (ex: pagină web, fișier, etc.). Este folosit în SW pentru a identifica unic elementele tripletelor în tot spațiul web, eliminându-se astfel conflictele de nume pe baza unui standard deja existent.

Limbajele. Tripletele sunt exprimate folosind limbajele SW. Acestea conțin instrucțiuni pentru uneltele SW. Sunt mai multe limbaje standardizate cu complexitate și expresivitate variabilă. Dacă se alege un limbaj mai expresiv se pot construi structuri de date mai complexe, dar această abordare necesită putere de calcul mai ridicată.

Ontologii. Acestea conțin afirmații care definesc concepte. De exemplu, în afirmația *Popescu este student,* obiectul student reprezintă un concept, iar Popescu este o instanță a acestuia. O ontologie definește concepte, relațiile între acestea și constrângeri. Făcând o analogie cu bazele de date relaționale, o ontologie poate fi privită ca schema bazei de date. Este indicat să se utilizeze sau să se extindă ontologii deja existente cu concepte specifice aplicației curente, deoarece astfel este mai ușor de implementat schimbul de cunoștințe între mai multe sisteme.

Date. Sunt tripletele care conțin informație despre anumite entități, în loc de concepte. Acestea pot fi instanțe ale anumitor concepte definite în ontologii, dar nu este un lucru obligatoriu. Datele formează cea mai mare parte a SW. O ontologie care descrie conceptul student poate fi utilizată în milioane de instanțe de date.

Odată definite cunoștințele și ontologiile necesare, este nevoie de unelte SW pentru a putea construi și stoca tripletele, pentru a interoga datele, a infera date suplimentare, a defini reguli de producție pentru a extinde capacitatea de inferențiere. Toate aceste unelte pot fi integrate în pachete software.

Unelte de construcție. Aceste unelte permit definirea și integrarea de triplete în ontologii sau instanțe. Pot fi interfețe grafice care permit unui utilizator să descrie informațiile în triplete printr-un editor sau care permit stocarea acestor triplete pentru utilizări ulterioare.

Unelte de interogare. Asigură navigarea prin graful de date SW pentru a extrage datele cerute printr-o interogare. Sunt soluții care permit navigare de grafuri, căutare sau implementare de limbaje complete pentru interogare.

Motor de inferențe. Se bazează pe conceptele, relațiile și constrângerile definite în ontologii pentru a deduce informații noi, a clasifica instanțele și proprietățile. Sunt mai multe tipuri de motoare de inferențiere, implementând reguli mai simple sau mai complexe, și care de obicei fac parte din pachete software.

Reguli de producție. Permit implementarea de reguli de inferențiere de un nivel superior, permițând fuzionare de ontologii și alte operații logice de dimensiuni mari (ex: căutare în șir de caractere).

Toate aceste unelte pot fi integrate în pachete software care să ofere o soluție integrată pentru dezvoltarea de aplicații SW. Un astfel de pachet poate fi structurat ca în figura 3, în care părțile lui componente reprezintă:

- *Convertoare* – fac parte din categoria de unelte de construcție și permit transformarea datelor din baze de date, pagini web, tabele, etc. în triplete RDF.
- *Procesare și serializare* – permit citirea datelor din fișiere RDF pentru stocare într-o bază de date RDF și procesul invers de export al datelor într-un format de serializare a tripletelor. Fac parte din uneltele de construcție.
- *Baza de date RDF* – formă optimizată de păstrare a tripletelor pentru utilizare în interogări și inferențe. În aceasta se păstrează toate tripletele, fie că sunt instanțe sau ontologii care definesc conceptele sau regulile de producție.
- *Motor de interogări și inferențe* – parte a sistemului de gestiune care permite interacțiunea cu datele. De obicei conține și un motor de generare a inferențelor.

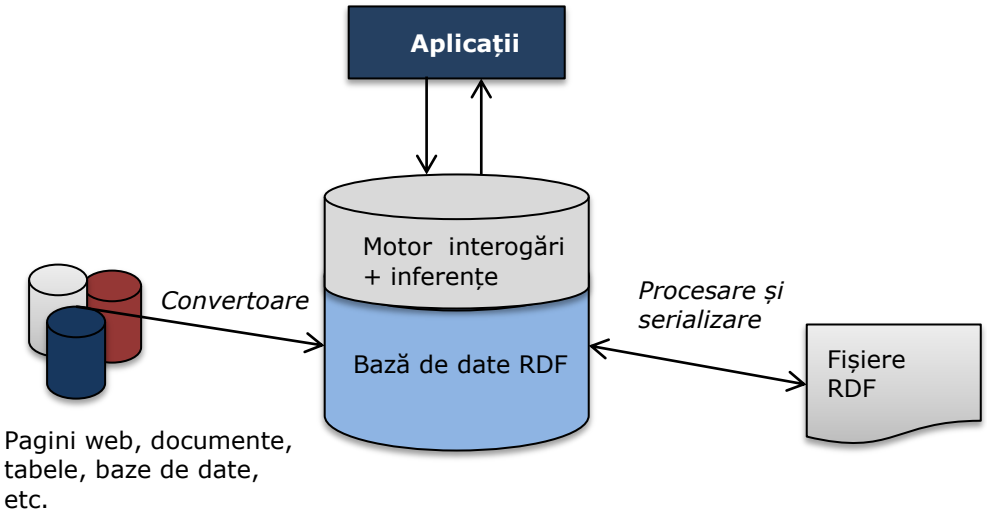


Figura 3. Structura pachet software SW

Având în vedere modelul de date utilizat în Web-ul Semantic și arhitectura unui pachet software SW, în continuarea acestui capitol introductiv mi-am propus să identific stadiul actual de definire și de implementare a standardelor (subliniind posibilitățile de utilizare în aplicații reale), uneltele software necesare proiectării și dezvoltării aplicațiilor. Acest pas este necesar pentru validarea fiabilității acestor tehnologii în construcția de modele de aplicații avansate pentru învățământul electronic.

2.2 Tehnologii ale Web-ului Semantic

Web-ul Semantic a fost introdus la scară largă în articolul "The Semantic Web" [4] publicat de Tim Berners-Lee, James Hendler și Ora Lassila în mai 2001 în *Scientific America*. Autorii schițează conceptele fundamentale: tripletele, limbajul

RDF, ontologiile și rolul URI-urilor. De asemenea prezintă un studiu de caz cu privire la un agent software de programări, care folosește informații adnotate semantic pentru a găsi furnizori de servicii medicale care îndeplinesc anumite criterii.

De la apariția articolului și până în prezent așteptările de la această tehnologie au continuat să crească atât în comunitatea tehnică, cât și în exteriorul ei. Chiar dacă majoritatea tehnologiilor necesare au fost schițate în articolul din 2001 și apoi extinse în cel din 2006 [17], până în momentul de față nu au apărut agenții software preconizați în cele două articole. Totuși, Web-ul Semantic nu poate fi privit ca un singur obiect, având la bază o serie de standarde, un set de unelte și, cel mai important, o comunitate care publică și partajează date pentru a îmbogăți aplicațiile.

W3C (The World Wide Web Consortium – comunitate internațională care se ocupă de dezvoltarea standardelor deschise pentru Web) prezintă Web-ul Semantic ca o stivă de tehnologii. A fost introdusă prima oară de către Tim Berners-Lee într-o prezentare despre Web-ul Semantic și XML [18] și prezintă modul în care tehnologii diferite sunt construite una peste alta pentru a extinde capabilitățile semantice ale Web-ului. De-a lungul timpului, stiva a evoluat pentru a reflecta corect evoluția standardelor și a încorpora noi situații de utilizare, pâna la forma prezentată în figura 4.

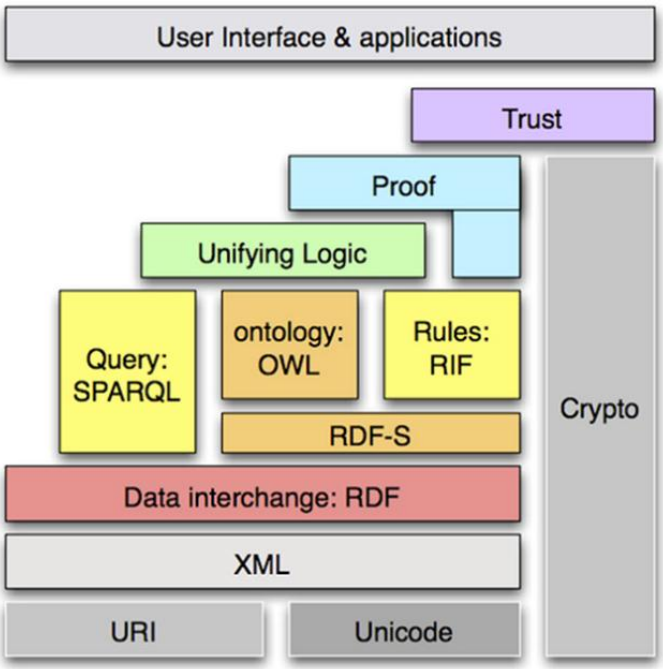


Figura 4. Stiva Web-ului Semantic

Tehnologii din partea de jos a stivei, până la OWL, sunt în prezent standardizate şi acceptate pentru a construi aplicaţii web semantice. Momentan nu este încă clar modul în care tehnologiile din vârful stivei vor fi puse în aplicare. Straturile de jos conţin tehnologii care sunt bine cunoscute, fiind utilizate în Web-ul actual. Utilizarea acestor tehnologii permite dezvoltarea Web-ului Semantic ca o extensie a Web-ului actual. Acestea sunt:

- **Uniform Resource Identifier** (URI), prevede mijloace pentru a facilita identificarea în mod unic a resurselor web. Web-ul Semantic are nevoie de identificare unică pentru a permite manipularea resurselor în straturile superioare.
- **Unicode** este utilizat pentru a reprezenta și manipula textul în mai multe limbi. Web-ul Semantic ar trebui să permită legături între resurse în diferite limbi.
- **XML** este un limbaj de marcare care permite crearea de documente de date structurate. Web-ul Semantic are nevoie de o metodă pentru a structura datele, ca apoi să poată construi legături între ele. De asemenea, pentru a facilita relaționarea între date provenite din surse diferite se utilizează XML Namespaces (spații de nume), identificând unic elementele din documentele XML.

Straturile din mijlocul stivei conţin tehnologiile standardizate de W3C, utilizabile în construirea de aplicații web semantice:

- **Resource Description Framework** (RDF), este primul strat specific Web-ului Semantic. După cum am precizat în subcapitolul precedent, este un limbaj care permite crearea de declaraţii în formă de triplete. Acestea permit reprezentarea datelor sub formă de grafuri.
- **RDF Schema** (RDFS), este un vocabular de bază pentru construcția de clase și proprietăți în RDF. Utilizând RDFS se pot construi vocabulare cu o complexitate redusă.
- Pentru construcția de vocabulare/ontologii mai complexe a fost standardizat limbajul **Web Ontology Language** (OWL). OWL extinde RDFS, aducând restricții suplimentare asupra structurii și conținutului fișierelor RDF și permițând construcția de algoritmi de raționare.
- La fel cum SQL este un limbaj de interogare standardizat pentru baze de date relaționale, **SPARQL** este standardizat ca limbaj de interogare pentru RDF. Poate fi utilizat pentru a interoga orice date RDF, inclusiv declarații în RDFS sau OWL.

Straturile superioare conţin tehnologii care se află în curs de standardizare sau doar în stadiu de idee, dar care ar trebui să fie puse în aplicare pentru a realiza Web-ul Semantic:

- **RIF** (Rule Interchange Format) sau **SWRL** (Semantic Web Rule Language) vor trebui să ofere suport pentru construirea de reguli în descrierea de legături. Vor fi utilizate în cazuri neacoperite de logica descriptivă din OWL.
- **Cryptography** și **Trust**, sunt importante pentru a asigura securitatea și calitatea datelor. Momentan nu există standarde implementate în Web-ul Semantic, ci doar o viziune de cum ar trebui să funcționeze sistemul.
- Stratul superior, **User interfaces and applications**, reprezintă de fapt aplicațiile care vor permite accesul la datele modelate pentru Web-ul Semantic.

Acest mod de a privi Web-ul Semantic, ca un întreg, sugerează că implementarea de aplicații utile utilizatorilor este condiționată de existența tehnologiilor standardizate pentru toate *straturile*. Având în vedere că tehnologiile de nivel superior sunt definite doar la nivel de concept, iar cercetarea în acest

domeniu este încă la început, am putea presupune că nu se pot construi aplicații web semantice. Totuși, au început să apară aplicații care au la bază tehnologii și concepte ale Web-ului Semantic. Dintre acestea, cele mai importante sunt:

- **Semantic MediaWiki (SMW)**, o extensie la MediaWiki, platforma care stă la baza Wikipedia.org. Utilizând wiki-urile colaborative s-au construit site-uri web foarte utile, permițând crearea de conținut ca și pagini web și asigurând legături între ele. Legătura între diferite pagini depinde puternic de intervenția umană. Dacă nu există nici o legătură către o anumită pagină, aceasta devine izolată. O soluție pentru a îmbunătăți wiki-urile, ca și platformă de publicare de conținut, este utilizarea tehnologiilor semantic web. **Wiki-urile semantice** permit colaboratorilor să introducă și să interogheze date semantice în interiorul unei pagini. Această abordare va permite utilizatorului să interogheze wiki-ul sau să navigheze semantic spre o anumită informație, în loc să depindă de link-urile native. În plus, aplicațiile pot să interogheze conținutul și să îl refolosească.
- Un alt tip de aplicații în care Web-ul Semantic are un impact puternic sunt rețelele sociale. **Facebook,** spre exemplu, utilizează pentru descrierea persoanelor și legăturilor între ele vocabularul FOAF. La baza Facebook se găsește un graf social de oameni și conexiuni ale acestora cu tot ceea ce îi interesează. Acest graf a fost extins prin dezvoltarea Open Graph protocol, care permite păstrarea informațiilor de interacțiune a utilizatorilor cu diferite pagini și site-uri web externe. Prin acest protocol, Facebook permite dezvoltatorilor de aplicații să realizeze legături persistente cu utilizatorii și să extragă informații relevante.
- **DBpedia** este o comunitate care dorește să extragă informaţii structurate de la Wikipedia şi să facă disponibile aceste informaţii pe Web. DBpedia permite rularea de interogări complexe asupra datelor din Wikipedia şi realizarea de legături cu alte seturi de date de pe Web. Scopul declarat al acestei mișcări este realizarea unor mecanisme noi de navigare pentru a îmbunătăți Wikipedia și a reutiliza datele construite deja de această comunitate.
- **GoPubMed**. Are la bază ontologia GO (Gene Ontology), utilizând-o pentru a structura articolele din baza de date Medline. Folosind această aplicație, căutarea este mult mai rapidă decât varianta "non-semantică" PubMed.
- **Freebase** este o bază colaborativă de date semantice, cu informaţii despre milioane de subiecte. În cadrul Freebase se pot găsi date de la organizații internaţionale și agenţii guvernamentale, fundații private, grupuri de cercetare universitare, proiecte open source, companii private şi utilizatori individuali, cu alte cuvinte, cineva care a publicat datele lor gratuit. Freebase este un graf de date, făcut din noduri şi legături. Dar, spre deosebire de grafurile RDF, permite dezvoltatorilor să trateze aceste structuri ca simple obiecte care sunt asociate cu unul sau mai multe tipuri Freebase.
- **Google**, cel mai important motor de căutare la ora actuală, a început să ofere suport pentru o serie de vocabulare. De exemplu, acesta extrage meta-datele produselor definite folosind Goodrelations și RDFa, afișând informații suplimentare în momentul afișării paginii respective în pagina de căutare.

După cum se poate vedea din lista de mai sus, utilizarea tehnologiilor Web-ului Semantic pentru îmbunătățirea serviciilor oferite este viabilă, având aplicabilitate în publicarea datelor pentru interconectare și reutilizare, în rețele sociale sau în agenți de căutare.

Dacă avem în vedere figura 3, dezvoltarea unei aplicaţii web semantice necesită implementarea mai multor componente (bază de date RDF, motor interogări și inferențe, convertoare, procesare și serializare date). În continuarea acestui capitol mi-am propus să identific standardele utilizate pentru stocare și schimbul de date, limbajele utilizate pentru construcția de vocabulare și ontologii, limbajele de interogare și modurile de publicare a datelor modelate semantic.

2.2.1 Formate de serializare a datelor RDF

După cum am spus și în subcapitolul precedent, modelul de descriere a datelor în Web-ul Semantic este dat de către limbajul RDF. Reprezentarea datelor sub formă de graf este un mod abstract foarte bun pentru analiza datelor de către oameni, dar neutilizabil în schimbul de informație între aplicații. Serializarea permite transmiterea datelor între aplicații, furnizând o metodă de conversie a grafurilor într-un formă concretă, cum ar fi un fișier text sau un flux de date.

Chiar dacă modelul RDF este foarte simplu, formele de serializare utilizate pentru partajarea informației tind să devină complicate datorită mecanismelor utilizate pentru comprimarea datelor. Aceste mecanisme implică de obicei utilizarea unor scurtături pentru identificarea de referințe multiple către un nod din graf (de exemplu, în figura 2, nodul Popescu).

Cum toate grafurile RDF au aceeași structură, orice format de serializare trebuie sa poată reprezenta aceleași construcţii (afirmații compuse din identificatori URI și valori literale). Diferitele forme de serializare au impus metode de reprezentare a acestor construcţii în forme mai convenabile, dar toate descriu de fapt aceeași informație.

RDF/XML

În prima recomandare W3C cu privire la RDF [19] au fost descrise atât modelul de date RDF cât și o metodă de a reprezenta aceste informații în RDF. Din această cauză, în anumite surse, descrierea datelor serializate RDF se referă la formatul RDF/XML, dar trebuie subliniat faptul că este doar o altă formă de serializare.

RDF/XML este singurul standard de serializare a datelor RDF [20] pentru schimbul de informație, motiv pentru care acesta trebuie să fie suportat de toate aplicațiile Web-ului Semantic. Sunt și alte sintaxe foarte utilizate, dar pentru a asigura interoperabilitatea între aplicații s-a ales RDF/XML ca şi sintaxă oficială.

Conceptual, un document RDF/XML este construit dintr-o serie de descrieri, fiecare dintre ele urmărind o muchie. Pornind dintr-un nod (subiect), muchia (predicat) descrie legătura cu un alt nod (obiect).

Dacă sunt mai multe muchii care pleacă din același nod, aceste afirmații se pot grupa în elemente <rdf:Description>. Dacă sunt mai multe grupuri, acestea trebuie să aibă ca și părinte elementul XML *rdf:RDF;* în caz contrar, avem un singur grup, iar acest element nu este necesar. Fiecare element de descriere conține un atribut *rdf:about*, care definește subiectul pentru toate afirmațiile din interiorul elementului. În mod similar, fiecare din elementele copil definesc predicatul și subiectul afirmațiilor.

Graful descris în figura 2 va avea următoarea formă în RDF/XML:

```
<rdf:RDF xmlns="http://exemplu.ro#"
  xmlns:log="http://www.w3.org/2000/10/swap/log#"
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">

   <rdf:Description rdf:about="http://exemplu.ro#Popescu">
      <anulNasterii  rdf:datatype="http://www.w3.org/2001/XMLSchema#integer">
                1984</anulNasterii>
      <areMaterie rdf:resource="http://exemplu.ro#BazedeDate"/>
      <areMaterie rdf:resource="http://exemplu.ro#Fizica"/>
      <areMaterie rdf:resource="http://exemplu.ro#Matematica"/>
      <arePrenume>Ion</arePrenume>
   </rdf:Description>
</rdf:RDF>
```

Standardul RDF-XML oferă o multitudine de metode de simplificare a sintaxei de declarare a datelor RDF. Acest fapt duce la imposibilitatea de a compara două fișiere RDF/XML generate folosind unelte de serializare diferite; informația va fi identică, dar exprimată în forme diferite. Tot din această maleabilitate a sintaxei, documentele rezultate prin serializare sunt dificil de citit de către un utilizator uman. Chiar și cu aceste dezavantaje, standardul RDF-XML este unul din cele mai utilizate formate pentru scrierea de date RDF.

N-triples

Reprezintă cea mai simplă formă de serializare. O afirmație este reprezentată în N-triple printr-o singură linie conținând subiectul, predicatul și obiectul. Datorită simplității limbajului, această formă a fost utilizată de către W3C Core Working Group pentru a exprima diverse studii de caz cu privire la modelarea datelor [21].

Fiecare linie dintr-un fișier N-triples reprezintă o afirmație formată din subiect, predicat și obiect urmată de un punct. Cu excepția nodurilor anonime sau valorilor literale, fiecare subiect, predicat, respectiv obiect este reprezentat printr-un URI complet. Nodurile anonime sunt reprezentate prin *_:nume,* iar obiectele cu valori literale sunt introduse între ghilimele.

Graful descris în figura 2 va avea următoarea formă în N-triples:

```
<http://exemplu.ro#Popescu> <http://exemplu.ro#anulNasterii>
                    "1984"^^<http://www.w3.org/2001/XMLSchema#integer> .
<http://exemplu.ro#Popescu> <http://exemplu.ro#areMaterie>
                                          <http://exemplu.ro#BazedeDate> .
<http://exemplu.ro#Popescu> <http://exemplu.ro#areMaterie>
                                                <http://exemplu.ro#Fizica> .
<http://exemplu.ro#Popescu> <http://exemplu.ro#areMaterie>
                                          <http://exemplu.ro#Matematica> .
<http://exemplu.ro#Popescu> <http://exemplu.ro#arePrenume> "Ion" .
```

Datorită simplității limbajului, este util în construcția manuală a seturilor de date pentru testarea și depanarea aplicațiilor. De asemenea, este recomandat în aplicațiile care au la bază fluxuri de date.

Terse RDF Triple Language (Turtle)

Turtle este o sintaxă de serializare mai ușor de utilizat și citit de către oameni. Nu este un limbaj XML, ci un limbaj special proiectat pentru date RDF. Acest limbaj a fost propus W3C în 2008 de către David Beckett și Tim Berners-Lee [22].

Se observă că datele serializate în format N-triples conțin informații repetitive. Această informație redundantă va necesita mai mult timp pentru transmiterea și procesarea datelor, ridicând o problemă în aplicațiile care produc și consumă cantități mari de informație. Prin adăugarea câtorva structuri suplimentare, limbajul Turtle reușește să reducă masiv repetiția informației.

Fiecare conexiune dintre două noduri dintr-un graf RDF va deveni o tripletă. Cum un nod poate participa în mai multe conexiuni, și de obicei așa se și întâmplă, se poate reduce numărul de caractere utilizate în N-Triples dacă utilizăm un simbol pentru a reprezenta nodurile repetitive. Astfel s-a împrumutat conceptul de XML Namespace și s-a introdus prefixul URI declarat la începutul documentului. Utilizând acest prefix putem scurta URI-urile.

O altă prescurtare introdusă în Turtle este posibilitatea de a lega afirmațiile despre același subiect. Utilizarea caracterului ";" ca terminator al unei triplet ne spune că următoarele două elemente sunt predicat și obiect ale aceluiași subiect.

Printr-o metodă similară, utilizând "," putem simplifica tripletele care au același subiect și predicat.

Graful descris în figura 2 va avea următoarea formă în Turtle:

```
@prefix : <http://exemplu.ro#>.
:Popescu :arePrenume "Ion"; :anulNasterii 1984;
:areMaterie :Fizica; :areMaterie :Matematica; :areMaterie :BazedeDate.
```

Similar RDF/XML, Turtle oferă o prescurtare pentru precizarea tipului unei resurse, litera *a* este utilizată în loc de forma mai greoaie rdf:type. Ca și celelalte trăsături Turtle, această prescurtare ajută la scrierea mai ușoară a tripletelor și crește lizibilitatea informației.

Tabel 2. Biblioteci în sursă deschisă pentru manipularea datelor RDF

Nume	Limbaj	Adresă web
RDFLib	python	https://github.com/RDFLib
Redland	C	http://librdf.org/
dotNetRDF	.Net	http://www.dotnetrdf.org/
JRDF	JAVA	http://jrdf.sourceforge.net/
ActiveRDF	Ruby	http://www.activerdf.org/
EasyRdf	PHP	http://www.aelius.com/njh/easyrdf/
Oroboro	JAVA	http://site.oroboro.googlecode.com/hg/about.html
RDF.rb	Ruby	http://rdf.rubyforge.org/
RAP	PHP	http://sourceforge.net/projects/rdfapi-php/

După cum se observă din cele trei forme de serializare expuse, Turtle reprezintă forma cea mai ușor de utilizat pentru scrierea și citirea tripletelor de către oameni, oferind metode de reducere a informației redundante și creștere a lizibilității. Cu toate acestea, RDF/XML este singurul limbaj standardizat și este indispensabil pentru comunicarea corectă între aplicații. În anumite cazuri de transmitere a informației (sub formă de flux de date, unde sintaxa este necesar să fie minimizată) N-triples este indicat.

Totuși, există foarte multe biblioteci în sursă deschisă pentru aproape toate limbajele de programare moderne, care asigură interacțiunea cu oricare din formatele de mai sus. De asemenea, o serie de unelte permit conversia datelor între cele trei sintaxe.

2.2.2 Vocabulare și ontologii

Având la dispoziţie metode de transpunere a unui graf RDF într-o formă serializată, putem considera că este asigurat primul pas pentru partajarea informației între mai multe aplicații. Aceasta rezolvă problema sintactică amintită în subcapitolul precedent. Pentru a păstra și sensul informației respective (sau cunoștințele), trebuie definite conceptele, relațiile și constrângerile. Acest lucru este posibil, după cum am amintit deja, prin definirea unei construcții asemănătoare unei scheme din sistemele de gestiune a datelor relaționare. Acestea se pot întâlni sub diferite nume: vocabulare, taxonomii sau ontologii. Fiecare dintre ele conține un set definit de termeni, critic pentru capacitatea de a exprima *cunoștințele.* Cea mai bună definire a lor am întâlnit-o în cartea lui John Hebeler [12][p 99 - 100]:

- **Vocabular –** este o colecție de termeni definiți, non-ambigui, utilizați în comunicare. Termenii introduși în vocabular trebuie să nu fie redundanți și să aibă același sens în orice context.
- **Taxonomie –** este de fapt un vocabular în care termenii sunt organizați ierarhic. Fiecare termen poate să aibă mai multe relații părinte-copil cu orice alt termen din taxonomie. Relația părinte-copil poate avea paritatea mai mulți la mai mulți, dar majoritatea taxonomiilor adoptă restricția că fiecare termen poate avea un singur părinte. În aceste cazuri taxonomia are o structură arborescentă.
- **Ontologie –** folosește un vocabular predefinit de termeni pentru a defini conceptele și relațiile dintre ele pentru un anumit domeniu de interes. Se poate referi la un vocabular, taxonomie sau concepte suplimentare, având de obicei un model logic avansat pentru descrierea unui domeniu de cunoștințe. Folosind ontologii se poate exprima semantica din spatele termenilor unui vocabular, relațiile și contextul în care se folosesc.

Web-ul Semantic folosește un limbaj de definire a schemelor și unul de definire a ontologiilor, pentru a permite dezvoltatorilor să construiască vocabulare, taxonomii și ontologii. RDF Schema (RDFS) se poate utiliza pentru definirea de taxonomii de clase și proprietăți, dar și pentru domeniul și gama de definire a proprietăților. Limbajul de definire a ontologiilor, OWL (Web Ontology Language), furnizează un limbaj foarte expresiv de definire a ontologiilor pentru domenii de cunoștințe.

Pentru definirea ontologiilor se utilizează, la fel ca în cazul datelor semantice, limbajul RDF, iar tripletele rezultate se pot păstra în aceeași bază de date RDF ca și datele în sine. Tripletele din ontologii, respectiv vocabulare, nu sunt de fapt decât o altă sursă de date, care descrie doar concepte, relații și constrângeri. Utilizând

ontologii bine definite, aplicațiile pot partaja datele păstrând sensul informației regăsită în triplete.

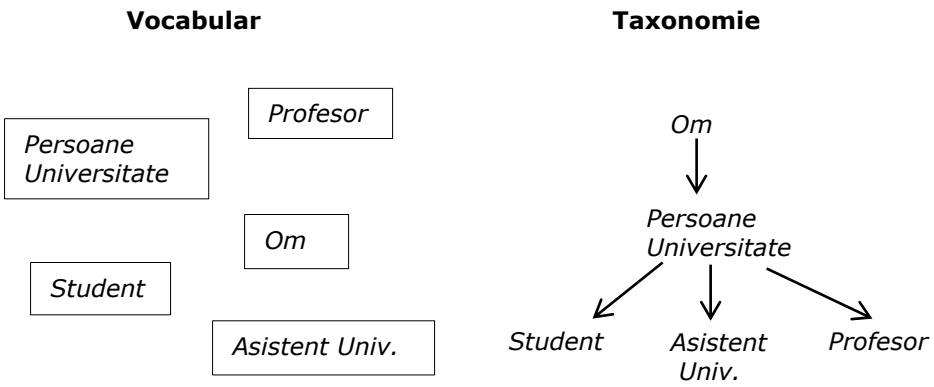


Figura 5. Vocabulare și Taxonomii

În plus, aplicațiile software semantice pot deduce, pe baza conceptelor, relațiilor și constrângerilor definite în ontologii, informații noi printr-un proces de inferență. Informația dedusă poate fi considerată ca și o concluzie extrasă prin evaluarea datelor existente în baza de date RDF. Aceste inferențe sunt generate sub aceeași formă de triplete ca și datele inițiale, oferind astfel posibilitatea ca această nouă informație să fie adăugată la cunoștințele inițiale. Ontologiile oferă un mod de a introduce reguli, pe baza cărora se pot extinde cunoștințele inițiale.

În cele ce urmează voi reliefa principalele diferențe între RDFS și OWL, subliniind limitările fiecăruia și cazurile în care este indicată utilizarea unui limbaj în locul celuilalt.

RDFS

Primul pas spre capacitatea de a exprima sensul informațiilor din datele RDF este de a construi un vocabular comun, care are termeni bine definiți și cunoscuți, fiind utilizat coerent în descrierea resurselor (subiect, predicat, obiect). RDFS nu încearcă să definească aceste vocabulare, ci un limbaj pentru descrierea de vocabulare comune.

A fost introdus în 2004 ca o recomandare a W3C [23], descriind metode de utilizare a limbajului RDF pentru definirea de vocabulare. La rândul ei, această specificație este introdusă ca un vocabular de termeni. Conține șase clase și șapte proprietăți de bază. Pe lângă acestea mai conține patru sub-vocabulare care descriu colecții, liste, reificare (afirmație despre o afirmație) și unelte.

Vocabularele RDFS descriu clasele de resurse și proprietățile utilizate în modele RDF. Folosind RDFS, se pot construi ierarhiile de specializare, defini domeniile și codomeniile unui predicat, afirma apartenențe la anumite clase și specifica tipul de date al unui obiect cu valoare literală.

În RDFS toate elementele tripletelor sunt considerate instanțe ale clasei resurselor. Se pot descrie suplimentar aceste instanțe prin clasele definite în RDFS

sau de către proiectantul ontologiei. În același mod se pot extinde și proprietățile din RDFS cu cele definite de dezvoltator.

RDFS nu este un limbaj foarte expresiv, dar este un prim pas important în construcția și definirea de ontologii. Datorită acestei limitări este ideal pentru utilizarea în vocabulare și ontologii care nu introduc concepte foarte complicate, sau în cazul în care nu este nevoie de capabilități de inferență avansate.

OWL

Limbajul de definire a ontologiilor, OWL (Web Ontology Language), extinde vocabularul RDFS cu resurse suplimentare care se pot utiliza în construcția de ontologii mai expresive. OWL introduce restricții suplimentare cu privire la structura și conținutul documentelor RDF pentru a permite procesarea și deducerea mai uşoară a informaţiei. A fost introdus prima oară în 2004, iar ultima versiune, OWL2, în 2009 [24]. La fel ca și RDFS, ontologiile declarate folosind OWL sunt scrise utilizând limbajul RDF.

OWL2 conține trei subseturi. În funcție de nivelul de complexitate al ontologiei modelate, se utilizează unul din cele trei subseturi. OWL2 EL este potrivit pentru utilizarea în cazul unor ontologii foarte complexe; limbaj foarte expresiv, dar necesită putere de calcul mare, prin urmare performanțe scăzute. OWL2 QL este ideal pentru cazurile în care avem ontologii relativ reduse, care sunt utilizate în organizarea unui set mare de date, optimizate pentru interogări relaționale. OWL2 RL, la fel ca și subsetul precedent, este indicat pentru utilizarea în definirea de ontologii reduse cu un set mare de date, dar optimizate pentru interogări rulate direct pe triplete RDF.

În momentul definirii de ontologii se poate utiliza în același timp RDFS și OWL, folosind un subset oricât de mic din conceptele introduse de ele. Limbajele sunt foarte flexibile și se pot utiliza într-un mod rezonabil și pragmatic pentru fiecare caz în parte.

În momentul de faţă există un număr destul de mare de vocabulare şi ontologii care se utilizează pe scară largă:

- **FOAF** (“friend of a friend”), este un vocabular utilizat pentru a reprezenta informații despre oameni, cum ar fi: nume, zile de naștere, poze, blog-uri și, în special, legătura cu alți oameni. Astfel, fișierele FOAF sunt deosebit de utile pentru a reprezenta datele care apar pe reţelele sociale. Mai multe reţele sociale permit accesarea datelor despre utilizatorii lor ca și fișiere FOAF. Informații suplimentare se găsesc la pagina web a proiectului [http://www.foaf-project.org/].
- **GeoNames** este o bază de date geografice, disponibilă online, care conține peste opt milioane de intrări. Datele sunt disponibile în diverse formate, inclusiv RDF. Informații suplimentare asupra utilizării [http://www.geonames.org/].
- **SIOC** (Semantically-Interlinked Online Communities Project) este un vocabular definit cu ajutorul RDFS, care oferă metode de interconectare a discuțiilor în diferite formate: bloguri, forumuri, liste de mail-uri. Este utilizat pentru a realiza concepte de tip “social media”. În acest scop, documentele SIOC pot folosi alte ontologii pentru a descrie mai bine informația. De exemplu, informația suplimentară despre autorul unui comentariu poate fi descrisă utilizând vocabularul FOAF. Informații suplimentare se pot găsi la pagina web a proiectului [http://sioc-project.org/].

- **SKOS** (Simple Knowledge Organization System) este un model de date care facilitează partajarea și conectarea datelor din sisteme de organizare a informației folosind Web-ul. Multe sisteme de organizare a informațiilor, precum: tezaurele, taxonomiile, schemele de clasificare, au o structură similară și sunt folosite în aplicații similare. SKOS utilizează aceste similitudini pentru a permite schimburi de date între diverse aplicații. Informații suplimentare se pot găsi la pagina web a proiectului [http://www.w3.org/2004/02/skos/].
- Pentru a descrie date din magazinele virtuale a fost dezvoltată ontologia de nivel inferior **GoodRelations**. Este momentan singura ontologie bazată pe OWL DL susținută oficial atât de Google cât și de Yahoo. Aceasta oferă un vocabular standard pentru a exprima toate detaliile comerciale şi funcționale ale unei aplicații e-Commerce, ca de exemplu: opțiuni de plată și livrare, cantitate, reduceri, program de lucru, modalitatea de livrare, etc. Informații suplimentare asupra utilizării vocabularului puteți găsi la adresa următoare [http://www.heppnetz.de/projects/goodrelations/].

În acelaşi fel în care OWL extinde RDFS, asigurând compatibilitatea datelor declarate folosind RDFS, este indicat ca un vocabular sau ontologie nouă să extindă pe cât posibil concepte deja definite în ontologiile populare, pentru aplicațiile semantice. Astfel se permite partajarea mai ușoară a datelor între sisteme diferite, permițând alinieri de grafuri RDF mult mai rapid și intuitiv.

2.2.3 Stocarea și interogarea datelor

La fel cum SQL este un limbaj de interogare standardizat pentru bazele de date relaționale, a fost necesară introducerea unui standard pentru a putea utiliza datele din grafuri RDF. SPARQL (Simple Protocol and RDF Query Language) a fost introdus ca și recomandare W3C în 2008 [25], iar în momentul de față este în curs de standardizare versiunea 1.1 care aduce o serie de îmbunătățiri [26].

Sunt și alte limbaje de interogare pentru date RDF, cele mai cunoscute fiind RDQL (RDF Data Query Language) și SeRQL (Sesame RDF Query Language), dar datorită standardizării SPARQL și adoptării pe scară largă în servicii deschise de date, acesta este mai indicat. Cele mai importante abordări în dezvoltarea limbajelor de interogare RDF sunt descrise pe larg în articolul lui James Baily publicat în 2005, anterior standardizării SPARQL [27].

SPARQL oferă o serie de funcții avansate pentru a construi șabloane de interogare, respectiv pentru a adăuga condiții suplimentare de filtrare, dar și metode de formatare a rezultatelor. Structura unei interogări SPARQL este similară cu o interogare SQL ca și sintaxă, dar cele două limbaje sunt fundamental diferite operând pe structuri de date complet diferite.

În plus față de standardizarea limbajului de interogare, SPARQL definește protocolul de comunicare între un client SPARQL și un server de stocare și rulare de interogări. Pentru aceasta folosește un alt standard W3C, și anume WSDL 2.0.

Pentru construcția șabloanelor de extragere a tripletelor se utilizează sintaxa Turtle. SPARQL adaugă în plus variabile de interogare pentru a specifica părțile din șablonul de interogare care se doresc a fi extrase ca și rezultat. Având exemplul de graf RDF din figura 2, pentru a selecta toate persoanele, respectiv materiile pe care le studiază, trebuie rulată interogarea următoare:

```
@prefix : <http://exemplu.ro#>.
select ?persoana ?materie
where {
  ?persoana :areMaterie ?materie
}
```

Sunt două componente principale în această interogare, SELECT și WHERE. Prima identifică ce variabile vor fi utilizate pentru selectarea datelor, iar a doua specifică șablonul de interogare și, eventual, informații suplimentare de filtrare.

Sunt patru tipuri de interogări suportate de SPARQL:

- SELECT – sunt returnate rezultate care nu fac parte dintr-un graf RDF;
- CONSTRUCT – permite atât transformarea datelor dintr-un graf RDF în altul, cât si combinări de grafuri;
- ASK – returnează adevărat sau fals în funcție de existența șablonului căutat;
- DESCRIBE – permite returnarea unui graf RDF fără specificarea unui șablon de la client; utilizat în cazuri în care nu se cunoaște structura informației.

Pentru modificarea datelor a fost dezvoltat inițial un limbaj separat denumit SPARUL (SPARQL/Update) [28], care permite operații de adăugare, ștergere, mutare, etc. SPARQL 1.1 redenumește acest limbaj în SPARQL Update și este anexat la recomandarea SPARQL. Față de această modificare minoră, SPARQL 1.1 introduce în plus subinterogări, căi de proprietăți de-a lungul grafului, calcul de expresii, grupare, interogări federative [29][30].

Am urmărit în continuare să identific un pachet software în sursă deschisă care să permită stocarea grafurilor RDF și să aibă implementată ultima versiune de SPARQL pentru motorul de interogare. Sunt multe unelte specializate care ating doar unul din pașii necesari unui sistem complet funcțional:

- Stocare – datele descrise folosind RDF trebuie stocate fie în memoria RAM a calculatorului, o bază de date sau un motor de stocare specializat pentru triplete;
- Încărcare a datelor – trebuie să fie capabil să recunoască cât mai multe formate de serializare a datelor;
- Deducere a informației – în caz de utilizare a RDFS sau OWL trebuie să aibă capabilități de inferare;
- Interogare – permite rularea de interogări asupra datelor stocate;
- Export – salvarea datelor în mai multe formate.

Unele unelte construiesc doar motoare de stocare optimizate pentru volume mari de date, altele sunt biblioteci care permit manipularea datelor în diferite formate de serializare. Am identificat unelte care să permită toate cele cinci facilități descrise mai sus, pornind de la colecțiile construite de comunitatea Web-ului Semantic [31][32]. Cele mai utilizate și stabile soluții gratuite sunt: Jena, dezvoltată inițial de Hewlett-Packard [33], Sesame, dezvoltată de compania olandeză Aduna [7], iar pentru dezvoltatorii PHP, ARC2 [34].

Din tabelul 3 rezultă că cele mai puternice sunt Jena și Sesame, ARC2 având deficiențe pe motoarele de stocare implicite și capabilitatea de inferare. Totuși, indic utilizarea acestei unelte în aplicații de mici dimensiuni, datorită bibliotecilor puternice pentru manipularea datelor RDF.

Conform datelor publicate în [35][36], Jena este mult mai rapidă la încărcarea datelor RDF în baze de cunoștințe folosind motoarele de stocare native (recomandând-o în aplicații în care datele se modifică foarte des), în timp ce Sesame returnează rezultatele interogărilor mult mai rapid (indicat a fi utilizată pentru seturi de date de până în 100 de milioane de triplete).

Atât Jena cât și Sesame permit rularea de interogări federative. În cazul în care este nevoie de capacitate mare de stocare (de ordinul miliardelor de triplete) se poate apela la motoare de stocare externe comerciale sau în sursă deschisă, cum ar fi: AllegroGraph, Virtuoso, BigOWLIM, Bigdata, etc.

Tabel 3. Comparație unelte semantic web

Cerință	**Sesame**	**Jena**	**ARC2**
Licență	LGPL	BSD	GPL
Dezvoltat în	JAVA	JAVA	PHP
Limbaje de interogare	RDQL SPARQL 1.1	SeRQL SPARQL 1.1	SPARQL
Motor de stocare	Memorie BD Fișier Nativ	Memorie BD Fișier	BD
Formate de serializare suportate	RDF/XML N-Triples N3 Turtle TriX JSON	RDF/XML N-Triples N3 Turtle	RDF/XML N-Triples N3 Turtle SPOG
Capabilități de inferare	RDFS OWL-Lite	RDFS OWL-Lite DIG 1.1 Interface	-
Server RDF	Protocol SPARQL	Protocol SPARQL	Protocol SPARQL
Adresă web	jena.apache.org	www.openrdf.org	arc.semsol.org

2.2.4 Publicarea de cunoștințe

Publicarea datelor în formate compatibile Web-ului Semantic duce la producerea unui efect de rețea datorită interconectării între seturile de date. De exemplu, informațiile publicate în format semantic cu privire la o instituție publică (orarul de funcționare, datele de contact, serviciile oferite, etc.) pot fi utilizate în aplicații externe pentru a construi servicii conexe împreună cu alte seturi de date. Cu cât sunt publicate mai multe date semantice, cu atât mai mult crește valoarea seturilor de date semantice curente, oferind posibilitatea dezvoltării de aplicații mai puternice și mai utile [13][p 155].

Principala problemă în adoptarea de către furnizorii de conținut a acestor principii a fost lipsa unor standarde și unelte ușor de utilizat. Formatul RDF/XML

este complicat de implementat, în plus, majoritatea informației se găsește în format HTML, astfel că administratorii aplicațiilor web trebuie să mențină două pagini pentru aceeași informație.

Pentru a se rezolva problema publicării informației în două formate, au fost dezvoltate microformatele, eRDF și RDFa care permit inserarea de etichete semantice direct în documentele HTML. Astfel, un program de procesare sau indexare a paginilor va fi capabil să extragă cunoștințe folosind etichetele definite. Aceste formate sunt utilizate de către motoare de indexare pentru a îmbunătăți calitatea serviciilor oferite prin afișarea de fragmente bogate în cunoștințe [37][38]. Acest fapt demonstrează utilitatea tehnologiilor Web-ului Semantic în construcția Web-ului de mâine.

Microformate

Au fost concepute pentru a oferi dezvoltatorilor o metodă simplă de inserare a etichetelor semantice direct în documentele HTML, folosind atributul *class.* Reprezintă un set de formate simple dezvoltate folosind standarde de comunicare des utilizate (ex: vCard – folosit în agende electronice) [39]. Unul din cele mai cunoscute astfel de cazuri îl constituie adaptarea standardului vCard pentru utilizarea în documente HTML, denumit hCard; este asemănător unei cărți de vizită, etichetând informații cu privire la o persoană sau organizație.

```
<div class=”vcard”>
    <div class=”fn”>Popescu Ion</div>
<div class=”org”>UPT</div>
    <div calss=”rel”>025640******</div>
    <a href=”url” href=http://www.upt.ro>Adresa web</a>
</div>
```

Microformatele au apărut din dorința de a implementa o parte din obiectivele Web-ului Semantic: structurarea conținutului web, sistem descentralizat de management al cunoștințelor și dezvoltarea de standarde pentru a încuraja reutilizarea structurilor. Microformatele prezintă o serie de dezavantaje: nu sunt dezvoltate de o singură entitate (Web-ul Semantic este centrat pe W3C); se adresează în primul rând oamenilor şi abia apoi aplicațiilor; încurajează utilizarea etichetării directe în documente HTML față de dezvoltarea de standarde noi (ex: RDF, OWL) [12][p 391-293].

Totuși, microformatele au reprezentat primii pași în atingerea conceptelor Web-ului Semantic, fiind indicată utilizarea lor pentru dezvoltarea de structuri simple [40]. Prin utilizarea de metode de conversie GRDDL, microformatele se pot integra în seturi de date semantice [41].

Cele mai utilizate microformate sunt:

- **hCard –** pentru informații despre persoane, locații, organizații și companii. Are la bază standardul vCard RFC 2426;
- **hCalendar –** pentru descrierea de evenimente dependente de calendar. Are la bază standardul iCalendar RFC 2445;
- **XFN –** descrie diferite relații între prieteni;
- **XOXO –** utilizat pentru a îmbogăți listele cu informații suplimentare cum ar fi titlu, rezumate, adrese URL;
- **hReview –** pentru descrierea de recenzii.

eRDF

eRDF reprezintă prescurtarea de la embedded RDF. Este un subset al limbajului RDF care se poate utiliza în pagini XHTML sau HTML. Prin subset se înțelege faptul că o parte a recomandării RDF nu este suportată în eRDF (noduri anonime, liste, subclase, etc.).

A fost dezvoltat în principal de către Ian Davis între anii 2005 și 2006 în cadrul companiei Talis [12][p 392-395]. În momentul de faţă, există biblioteci pentru extragerea cunoștințelor ce conțin structuri eRDF, dar având în vedere faptul că eRDF nu este un standard W3C este puțin probabil să fie adoptat pe scară largă.

Primul pas în utilizarea acestui format este declararea faptului că documentul HTML conține eRDF, folosind atributul *profile* în capul documentului (x)HTML. Apoi, trebuie specificate spațiile de nume utilizate în structurarea datelor. Al treilea pas implică efectiv inserarea tripletelor.

Un exemplu de accesare a vocabularului FOAF, utilizând eRDF, este următorul:

```
<html>
<head profile=”http://purl.org/NET/erdf/profile”>
  <link rel=”schema.rdf” href=”http://www.w3.org/1999/02/22-rdf-syntax-ns#”>
  <link rel=”schema.rdfs” href=”http://www.w3.org/2000/01/rdf-schema#”>
  <link rel=”schema.foaf” href=”http://xmlns.com/foaf/0.1/”>
</head>
<body>
    <div id=”Popescu” class=”-foaf-Person”>
         Popescu este prieten cu
    <span class=”-foaf-Person foaf-knows” id=”Ionescu”> Ionescu</span>.
</body>
</html>
```

RDFa

Este o recomandare W3C care permite inserarea de etichete semantice în interiorul documentelor xHTML [42]. RDFa a fost dezvoltat după apariția microformatelor și încearcă să reutilizeze cele mai bune concepte din acestea. RDFa este interoperabil cu limbajul RDF și suportă spațiile de nume XML și sintaxele de compactare a URL-urilor.

Deoarece dezvoltatorii de cunoștințe vor dori să creeze propriile vocabulare, taxonomii și ontologii, RDFa le permite acestora publicarea de date semantice fără restricții.

Pentru definirea subiectului unei triplete se utilizează atributul *about*, iar pentru definirea predicatului unul din atributele *property, rel* sau *rev.* Obiectele care sunt reprezentate prin URI se declară folosind unul din atributele *href, resource* sau *src*, în timp ce obiectele cu valori literale se declarăfolosind atributul *content* sau textul delimitat de etichete.

Exemplul precedent rescris folosind RDFa ia următoarea formă:

```
<html xmlns=http://www.w3.org/1999/xhtml
            xmlns:foaf=http://xmlns.com/foaf/0.1/
        xmlns:ex=”http://exemplu.ro#”>
<body>
         Popescu este prieten cu
    <div about=”http://exemplu.ro#Popescu”
          rel=”foaf:knows”
          resource=”[ex:Ionescu]”>
     Ionescu
    </div>
</body>
</html>
```

Dacă comparăm cele două abordări de a încorpora etichete semantice în interiorul paginilor HTML, microformatele sunt direcționate spre a rezolva problemele curente cu care se confruntă dezvoltatorii de aplicații web, în timp ce RDFa este proiectat pentru a integra complet limbajul RDF, asigurând în acest fel interoperabilitate și o expresivitate ridicată [43][44].

Conform celor de mai sus, sugerez utilizarea microformatelor stabile pentru introducerea de etichete semantice cu privire la structuri de date simple, standardizate deja, iar pentru dezvoltări viitoare utilizarea RDFa, datorită posibilității etichetării folosind vocabulare și ontologii specifice domeniului respectiv.

O altă metodă de a publica conținut semantic, în acest caz date RDF, este utilizarea unei aplicații care să permită interogări directe pe o bază de cunoștințe. Această facilitate este integrată în unelte pe care le-am prezentat în partea referitoare la stocarea și interogarea cunoștințelor. Atât Jena cât și Sesame permit acest lucru utilizând protocolul SPARQL. Există biblioteci pentru toate limbajele de programare moderne care să permită comunicarea între client (aplicația consumatoare de date RDF) și serverul de stocare a datelor. Această metodă se utilizează pentru a publica seturi de date de dimensiuni mari, fără caracter confidențial, și pentru a face posibilă implementarea Linked Data [45].

2.3 Contribuții și concluzii

Scopul acestui capitol a fost familiarizarea cu conceptele de bază din Web-ul Semantic, subliniind necesitatea apariției acestei tehnologii, identificarea componentelor unui sistem care să implementeze aceste concepte, descrierea pe scurt a tehnologiilor componente (precizând gradul lor de standardizare) și, nu în ultimul rând, identificarea de unelte și limbaje suficient de stabile pentru implementarea de aplicații semantice.

După descrierea modelului de date utilizat în Web-ul Semantic și a arhitecturii unei pachet software destinat lui, am identificat stadiul actual de standardizare a fiecărui strat din stiva Web-ului Semantic propusă de Tim Berners-Lee (inițiatorul web-ului semantic). Chiar dacă straturile superioare sunt momentan doar la stadiul de proiect, am argumentat totuși posibilitatea construirii de platforme educaționale

îmbogățite semantic, prin identificarea de aplicații care să pună în practică paradigmele Web-ului Semantic.

Pentru a putea trece la pasul următor, și anume propunerea de metode de integrare a acestei noi tehnologii informaționale, în primă fază a trebuit să identific limitările curente și direcțiile de dezvoltare ale acesteia. Am studiat formatele de serializare a datelor RDF. Turtle este cel mai ușor de utilizat pentru scrierea și citirea tripletelor de către oameni, oferind metode de reducere a informației redundante și creștere a lizibilității. Cu toate acestea, RDF/XML este singurul limbaj standardizat și este indispensabil pentru comunicarea corectă între aplicații. În anumite cazuri de transmitere a informației, unde sintaxa este necesar să fie minimizată (fluxuri de date), este indicată utilizarea N-triples.

Modelarea datelor în format RDF se face folosind vocabulare, taxonomii sau ontologii. Ca și limbaje standardizate de W3C pentru definirea acestora, se poate utiliza în același timp RDFS și OWL, folosind un subset oricât de mic din conceptele introduse de ele. Limbajele sunt foarte flexibile și se pot utiliza într-un mod rezonabil și pragmatic pentru fiecare caz în parte.

Având datele modelate și serializate corespunzător, pentru a le face disponibile în aplicații este nevoie de un sistem de stocare și interogare. Pentru interogarea datelor, W3C a standardizat SPARQL. Pasul următor a fost să identific, prin comparație, cele mai utile platforme în sursă deschisă din punct de vedere al stabilității, gradului de implementare al standardelor, capacităţii de lucru cu volume mari de date. Jena este mult mai rapidă la încărcarea datelor RDF în baze de cunoștințe folosind motoarele de stocare native, în timp ce Sesame returnează rezultatele interogărilor mult mai rapid. Recomand utilizarea Jena pentru volume de date mari modificate des, iar Sesame pentru situații în care se rulează preponderent interogări de selecție.

Pentru publicarea datelor am studiat formatele care permit o utilizare ușoară de către dezvoltatori. Sugerez utilizarea microformatelor, pentru introducerea de etichete semantice cu privire la structuri de date simple, standardizate deja, iar pentru dezvoltări viitoare utilizarea RDFa, datorită posibilității etichetării folosind vocabulare și ontologii specifice domeniului respectiv. De asemenea, sugerez utilizarea capabilităților de publicare integrate în platformele menționate mai sus.

În concluzie, prin cele expuse în acest capitol, consider tehnologiile Web-ului Semantic suficient de mature pentru utilizarea lor în platformele educaționale, unde calitatea ridicată a serviciilor oferite este obligatorie. De asemenea, consider aceste studii şi concluzia care derivă din ele, ca o primă contribuţie teoretică importantă a prezentei teze.

3 Învățământul electronic în contextul Web-ului Semantic

Acest capitol cuprinde introducerea conceptelor de e-Learning, studiul platformelor de management al conținutului educațional și exemple de utilizare de meta-date pentru îmbunătățirea serviciilor oferite. Principalul obiectiv pe care am vrut să îl îndeplinesc în această parte a tezei a fost acela de a identifica structura datelor gestionate în platformele educaționale, pentru a defini în pasul următor modelul ontologic necesar pentru modelarea lor în formate compatibile Web-ului Semantic.

3.1 Sisteme de e-Learning

Termenul de e-Learning este cunoscut ca utilizarea intenționată a tehnologiilor informaționale și de comunicare în procesele de predare și învățare. O serie de alți termeni sunt utilizați pentru a descrie acest mod de desfășurare a procesului didactic: învățământ online, învățământ virtual, învățământ distribuit, învățământ bazat pe rețea sau web. Toate aceste tehnologii referă de fapt utilizarea tehnologiilor informaționale în procesul educațional pentru a media activități asincrone sau sincrone [46].

Semnificația termenului e-Learning s-a schimbat în decursul timpului. Unele definiții restricționează utilizarea termenului doar la sistemele care utilizează tehnologiile internet. Cea mai simplă și cuprinzătoare definiție a fost dată de consiliul de evaluare a învățământului deschis și la distanță din Marea Britanie (Open and Distance Learning Quality Council UK) [47]:

”E-Learning-ul este procesul eficient de învățare creat prin combinarea livrării digitale de conținut, cu suport și servicii.”

Interes crescut pentru tehnologii e-Learning a fost manifestat de organizații care au oferit programe de educație la distanță sau cu o formă mixtă (blended learning). Încorporarea tehnologiilor învățământului online în activitățile lor didactice a fost un pas logic. În sectorul corporațiilor, tehnologiile e-Learning au fost adoptate pentru a reduce costurile de instruire a angajaților. De asemenea, universitățile tradiționale au recurs la utilizarea acestor tehnologii pentru a îmbunătăți accesul studenților la programele lor educaționale și pentru a accesa anumite piețe de nișă.

Astfel, creșterea e-Learning-ului este direct relaționată cu accesul la tehnologiile informaționale și scăderea costurilor de implementare a sistemelor electronice destinate acestui proces [46].

Evoluția e-Learning-ului (învățământului electronic) este strict legată de cea a calculatoarelor și Internetului. Scăderea costului, odată cu creșterea puterii de calcul disponibilă publicului larg, concomitent cu dezvoltarea tehnologiilor internet și accesul de mare viteză la rețeaua globală au încurajat dezvoltarea sistemelor educaționale electronice.

Inițial au existat două tipuri fundamentale de e-Learning [48]:

- **Sincron –** presupune că interacțiunea între participanți și tutor se întâmplă în timp real folosind Internetul. Participanții interacționează între ei sau cu tutorele folosind unelte de tip chat, conferință audio video, etc. Principalele avantaje sunt: posibilitatea monitorizării activităților de învățare, conectivitate la nivel global între participanți, abilitatea de a personaliza procesul de instruire pentru fiecare participant.
- **Asincron –** participanții pot parcurge materiale de curs în ritmul propriu, fără interacțiune în timp real cu tutorele. Informația este disponibilă 24 de ore din 24. Avantajul acestui tip de e-Learning este dat de faptul că oferă studenților informația când au nevoie de ea. Aceste sisteme conțin de asemenea forumuri de discuții, bloguri și alte unelte de comunicare. Alte avantaje ale sistemelor asincrone sunt: flexibilitate, se poate adresa unui număr mare de studenți, cost de producție redus.

În plus față de acestea, a apărut conceptul de învățământ mixt sau blended learning. Acesta presupune utilizarea de concepte din învățământul sincron și asincron. Pentru partea sincronă se pot utiliza unelte electronice (în cazul învățământului la distanță) sau întâlniri față în față (învățământul clasic). În acest context, partea asincronă vine ca o formă de suport suplimentar oferită cursanților, permițând accesul la materialele educaționale, la unelte de comunicare, precum și la alte unelte conexe [49].

Principalele motive pentru care se alege implementarea unui sistem de tip mixt sunt conform lui Graham [50]:

- Îmbunătățirea practicilor pedagogice – cu scopul de a crește eficiența procesului didactic, de a schimba strategiile de transmitere a informației în strategii interactive;
- Creșterea accesului și flexibilității – pentru a permite studenților să acceseze materialele educaționale de la distanță și să parcurgă activitățile în ritmul lor, dar păstrând interacțiunea cu tutorele prin întâlniri față în față;
- Eficiența costului – pentru a diminua costul total al procesului didactic; un sistem educațional de tip mixt ar putea duce la reducerea numărului necesar de tutori.

Pentru a permite implementarea oricărui tip de e-Learning, este nevoie de anumite unelte software care să fie capabile să ofere serviciile necesare. Acestea pot fi împachetate în aceeași platformă, oferind o soluție completă, sau pot fi o serie de unelte electronice disponibile gratuit sau contra cost pe care tutorele decide să le utilizeze în procesul educațional. Prima variantă este utilizată de obicei de organizații care oferă un volum mare de cursuri, iar a doua este utilizată de un tutore individual pentru a îmbunătăți procesul pedagogic. În cele ce urmează voi

introduce conceptul de LMS și voi identifica structura unei platforme dedicate procesului educațional.

3.1.1 Platforme e-Learning

Un **LMS** (Learning Management System) este un pachet software proiectat pentru a facilita desfășurarea procesului de învățare folosind ca mediu de comunicare Internetul [2]. Rulează pe un server web, de obicei gestionat de către universitatea în care se desfășoară cursul, studenții având acces la facilitățile oferite de acesta prin intermediul unui navigator web. Orice tip de conținut ce poate fi prezentat prin intermediul Internetului (text, grafică, sunet și video) poate fi disponibil în LMS. Diferența principală între un LMS și alte situri web este că instituțiile pot restricționa accesul studenților înrolați într-un curs, pot prezenta materialele într-un mod automat și pot monitoriza accesarea lor.

În plus față de furnizarea de conținut educațional, un LMS permite ca procesul educațional să se desfășoare colaborativ. Un astfel de sistem include forumuri de discuții și aplicații software sociale, cum ar fi blog-uri și wiki-uri. Aceste unelte sunt asincrone, permițând studenților să participe fără a avea un program impus. Uneltele sincrone (conferințe video și audio, mesagerie instant) sunt de asemenea integrate în LMS-uri, cu toate că unele dintre ele rămân ca sisteme exterioare platformelor educaționale.

Orice organizație care dorește să utilizeze un LMS pentru a îmbunătăți procesul educațional are trei posibilități: utilizarea unei variante comerciale, a unei variante gratuite în sursă deschisă sau dezvoltarea unei platforme educaționale interne.

Soluțiile comerciale sunt de obicei stabile, asigurând anumite facilități de bază pe o structură proprie. Principalul dezavantaj îl constituie costurile pentru achiziționarea licenței de utilizare și suport. Flexibilitatea redusă a unui astfel de sistem nu permite adaptarea modului de funcționare, deoarece nu este oferit accesul la codul sursă.

Comparativ cu soluțiile comerciale, cele în sursă deschisă oferă aproximativ aceleași facilități, dar și adaptarea platformei la nevoile universității. Cu toate că nu sunt costuri pentru achiziționarea pachetelor software, totuși există costuri pentru dezvoltarea, administrarea și suportul acestor soluții. O altă limitare este dată de faptul că orice produs software dezvoltat folosind ca și bază un produs în sursă deschisă nu poate fi comercializat, deoarece licența originală trebuie păstrată [51]. În concluzie, produsele în sursă deschisă sunt o soluție pentru universitățile care vor să aibă costuri reduse pentru achiziționarea platformei educaționale, păstrând în același timp posibilitatea adaptării soluției la nevoile proprii.

A treia posibilitate constă în dezvoltarea unei soluții proprii, care să aibă aceleași facilități ca și variantele comerciale sau în sursă deschisă. Presupune un proiect software complex, cu un număr mare de angajați specializați și o perioadă de timp mare pentru implementare. Costurile vor fi bineînțeles mai mari decât în cazul adaptării unei soluții în sursă deschisă la nevoile proprii. Avantajul ar fi că platforma astfel obținută ar fi complet pliată pe nevoile universității. Totuși, datorită complexității proiectului și costurilor, nu recomand această abordare decât în cazurile în care nu există o platformă în sursă deschisă care să îndeplinească majoritatea cerințelor.

Ca și exemple de soluții comerciale, cele mai utilizate sunt Blackboard și WebCT, aceasta din urmă fiind achiziționată ulterior de către Blackboard, iar ca soluții în sursă deschisă Moodle, Sakai și OCW. Gradul de utilizare a variantelor

comerciale raportat la cele în sursă deschisă în universitățile de top din lume, comparativ cu cele din România, este complet diferit. Conform studiului realizat de colegul meu Ternauciuc Andrei în teza sa de doctorat [52], universitățile străine preferă soluții comerciale, în timp ce în România predomină Moodle (figura 6), variantă în sursă deschisă. În cazul universităților străine este totuși interesant că în primele zece din cele 50 studiate, doar una folosește soluții comerciale.

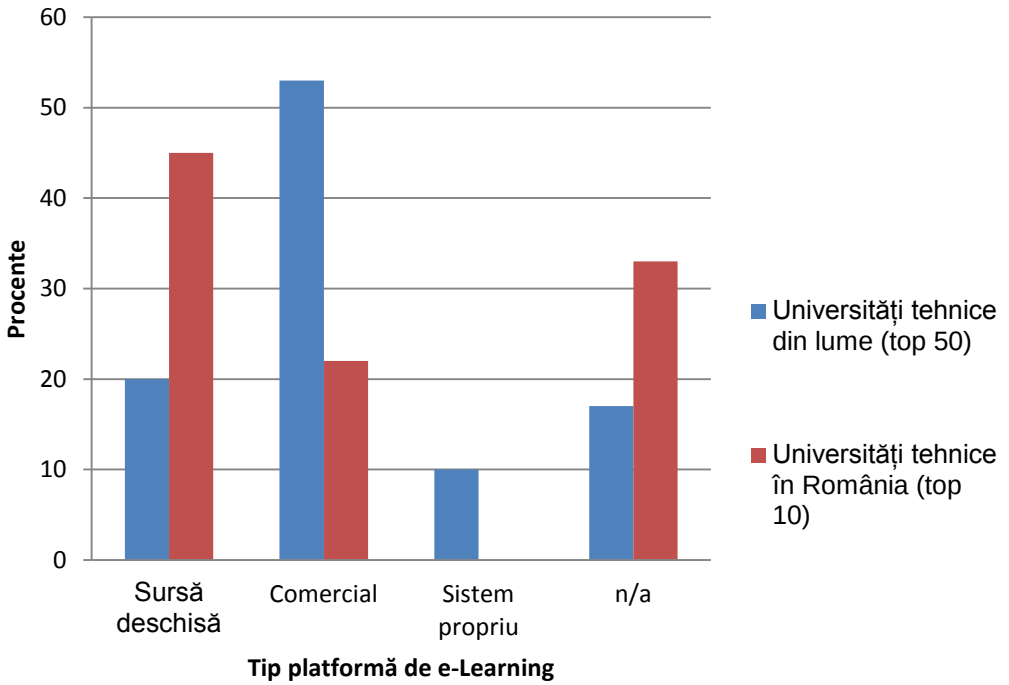


Figura 6. Utilizare tipuri de soluții LMS în universități tehnice (țară și străinătate)

Din studiul colegului meu rezultă doi jucători mari pe piață, Blackboard, ca și variantă comercială, și Moodle, ca și variantă în sursă deschisă. O comparație între cele două sisteme a fost realizată de către Dave Bremer și Reuben Bryant [53], pe trei direcții diferite: din punctul de vedere al administratorilor, al tutorilor și al studenților. La momentul realizării studiului, aceștia utilizau Blackboard ca și platformă educațională, dar vroiau să evalueze posibilitatea trecerii la o variantă în sursă deschisă.

Din perspectiva administrării platformei educaționale, în soluția în sursă deschisă sunt mai greu de rezolvat eventuale erori, deoarece nu este asigurat suport. Soluțiile la eventuale probleme trebuie identificate în cadrul comunității folosind forumurile de discuții. Din perspectiva unui administrator, curba de învățare este mai lungă. Ca și avantaj, pot menționa consumul mai redus de resurse și stabilitatea platformei.

Din punctul de vedere al unui tutor, trecerea de la o platformă la alta presupune adaptarea persoanei la noua platformă. Unele facilități diferă, astfel că tutorii sunt mai reticenți la utilizarea Moodle. Studenții, pe de altă parte, au fost mai înclinați către Moodle.

Pentru a studia structura unui LMS m-am axat pe Moodle, deoarece aceasta este platforma utilizată și în cadrul Universității "Politehnica" din Timișoara pentru a oferi capabilități de învățământ mixt. Pe situl Moodle [54] sunt declarate aproximativ 68.000 de instanțe active în 220 de țări, având 40 de milioane de utilizatori. În România sunt 243 de instanțe, demonstrând gradul mare de utilizare. La sfârșitul anului 2010 a fost lansat Moodle 2, având o serie de facilități suplimentare față de versiunea precedentă.

Moodle (Modular Object Oriented Dynamic Learning Environment) este construit, după cum ii spune și numele, pe un concept modular, făcând posibilă în acest fel o integrare ușoară de module externe pachetului de bază. În cazul în care o organizație consideră că este necesară o funcționalitate neinclusă în pachetul standard, aceasta poate dezvolta un modul suplimentar, iar integrarea este relativ simplă. Modulele dezvoltate de comunitatea Moodle sunt disponibile pe situl oficial al platformei, permițând astfel reutilizarea gratuită a lor. În continuare voi reliefa modulele centrale disponibile implicit în pachetul software.

Organizarea materialelor educaționale se face în funcție de cursul de care aparțin, astfel că o parte importantă a platformei este destinată definirii cursului, împreună cu datele necesare pentru descrierea lui. El reprezintă spațiul comun de lucru al unei grupe educaționale formată de obicei din studenți și tutori. Cursurile pot fi organizate în categorii, folosind un sistem ierarhic. O abordare corectă în descrierea cursurilor este dată de realizarea structurii astfel încât acestea să reflecte curriculum-ul. Există patru tipuri de cursuri:

- Formatul subiecte, care permite organizarea pe module de studiu. Aceste module pot fi capitole dintr-un material de curs sau subiecte de studiu. Se pot utiliza toate uneltele de comunicare și evaluare disponibile.
- Formatul săptămânal, în care materialul este organizat pe săptămâni de studiu. Aceasta este singura diferență față de formatul precedent.
- Formatul social, structurat în jurul unui forum, materialele și activitățile fiind în plan secund. Procesul de învățare constă din discuții libere purtate între participanți. În plus față de comentarii, se pot posta fișiere, imagini, sau alte resurse considerate utile.
- Ultima opțiune este formatul SCORM, care permite importul de obiecte educaționale realizate în exteriorul aplicației. Modulul din Moodle este un interpretor al pachetului SCORM, păstrând evoluția studenților în parcurgerea lui.

Odată creat un curs, se pot adăuga materiale de curs, cunoscute in interiorul platformei sub numele de resurse, sau module de activități. Ca resurse de curs se pot adăuga fișiere (care pot fi orice tip de fișier electronic, de la documente pdf la prezentări power point), un dosar care să conțină mai multe documente, o locație url, o pagină HTML care să conțină materialul formatat folosind acest limbaj de marcare sau un pachet IMS.

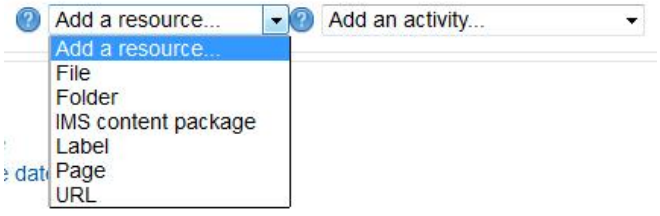


Figura 7. Tipuri de resurse care pot fi adăugate într-un curs

În plus fată de resursele de curs, Moodle permite tutorului să adauge diferite activități la un curs, de la teme de îndeplinit de către studenți, la unelte de colaborare și comunicare. Datorită sistemului modular, Moodle permite integrarea facilă de noi unelte, care să ofere o funcționalitate îmbunătățită sau una nouă. Activitățile de tip temă pe care un tutore le poate adăuga sunt următoarele:

- **Urcare fișier** – un student poate urca un singur fișier. Acesta poate fi un document text sau orice document digital. Mai multe fișiere se pot arhiva într-un document zip. După ce studenții urcă tema, tutorele poate să deschidă respectivele documente, să acorde o notă și să publice un comentariu, dacă este cazul.
- **Urcare avansată de fișiere** – permite urcarea mai multor fișiere, împreună cu un mesaj pentru fiecare document.
- **Text online** – această activitate cere utilizatorilor să compună și să editeze text, folosind editorul integrat în platformă. Tutorele poate să noteze activitatea, să adauge comentarii în interiorul temei sau să editeze tema.
- **Activitate offline** – utilizată pentru a comunica cerințele unei teme către studenți, dacă activitatea se desfășoară față în față. Sistemul de notare funcționează la fel ca și în cazurile precedente.

Modulul **glosar** permite participanților să creeze și să administreze o listă de definiții. Intrările din glosar pot fi vizualizate sau căutate în diferite formate. O activitate de tip glosar poate fi colaborativă sau restricționată la intrările făcute de tutore. Definițiile pot fi organizate în categorii.

Activitatea **SCORM** permite tutorelui să urce și să includă în curs orice pachet care respectă acest format. SCORM este o colecție de specificații care permit resurselor web educaționale să fie reutilizate.

Modulul **workshop** permite realizarea unei activități în care evaluarea este făcută de către colegi. Este similar cu modulul de teme, adăugând o serie de facilități. Fiecare participant la activitate urcă propria temă în format text sau documente. Este premis și modul de lucru pe grupuri, în acest caz tema urcată este pentru toți membrii grupului. Temele pot fi evaluate de către alți participanți la activitate. La final, studenții vor primi două note: una pentru materialul urcat, alta pentru calitatea evaluărilor date pentru colegii lor.

Activitate de sugestii (**Feedback**) permite rularea de chestionare pentru a colecta informații de evaluare. Un alt modul asemănător, dar care este utilizat în procesul didactic este **Choice**. Acesta permite tutorelui să adreseze întrebări participanților la curs pentru a deschide subiectul pe o anumită temă, pentru a permite realizarea unui proces de votare cu privire la activitățile desfășurare sau pentru a primi aprobarea din partea participanților.

O parte importantă din aceste activități este reprezentată de **uneltele de comunicare**: wiki, forum, chat, blog. Utilizarea uneltelor de comunicare duce la un angajament mai mare în procesul de învățare și la un nivel mai ridicat de satisfacție a studenților [55]. Moodle a fost proiectat și construit să urmeze filozofia constructivismului social [56]. Scenariul ideal de învățare în Moodle este ca grupurile să construiască conținut unul pentru celălalt, într-un mod colaborativ, creându-se astfel o comunitate care produce conținut educațional, în timp ce fiecare membru poate să își formeze propria metodă de învățare. Rolul tutorelui în acest model este de a supraveghea activitatea educațională și de a îndruma corect participanții. Un rol foarte important în acest scenariu îl reprezintă uneltele de comunicare: forumurile pot fi utilizare în comunicarea asincronă, discuțiile fiind disponibile și celorlalți participanți; unealta de chat permite colaborarea într-un mod

sincron; unealta wiki este utilă în realizarea de conținut colaborativ ca și grup; blogul poate fi utilizat pentru a realiza conținut individual și pentru a primi sugestii de la ceilalți participanți.

Pentru evaluarea participanților se poate utiliza modulul **Quiz,** care permite definirea mai multor tipuri de elemente de test (întrebare de tip eseu, cu răspuns scurt, cu răspuns numeric, cu răspunsuri multiple, cu răspuns adevărat/fals, etc.), păstrarea lor într-o bancă de întrebări și reutilizarea lor în diferite teste. În cazul în care acest modul nu satisface nevoile tutorului, se poate utiliza unul din modulele dezvoltate de comunitate.

Având conținutul educațional asigurat, utilizatorii trebuie restricționați la anumite facilități ale platformei, în funcție de rolul pe care aceștia îl au de îndeplinit în interiorul platformei sau cursului. Moodle permite organizarea privilegiilor în roluri și atribuirea acestora utilizatorilor, în funcție de context. Atribuirea rolurilor în interiorul unui curs se face prin procesul de înrolare a utilizatorilor ca și participanți la curs. Pentru ca procesul de înrolare să se desfășoare mai ușor pentru administrator, utilizatorii pot fi organizați în cohorte, iar aceste cohorte pot fi înrolate în cadrul unui curs sau categorie de cursuri. Se pot utiliza și module ce asigură legătura cu servicii externe de administrare a conturilor de utilizator.

Moodle cuprinde și alte facilități suplimentare neprezentate în textul de mai sus. Am urmărit să identific doar structura de bază a organizării conținutului educațional și a uneltelor conexe des utilizate pentru a identifica posibilitățile de modelare RDF a datelor disponibile în platformă.

Facilitățile prezentate mai sus se pot structura în patru secțiuni: cursuri (organizarea și materialele educaționale), activități (toate modulele ce ajută în procesul educațional), evaluare (uneltele de evaluare, pot fi considerate și ca activitate de curs) și utilizatori (administrarea participanților și privilegiilor pe care aceștia le au în sistem).

Voi prezenta în continuare forma de organizare a pachetelor de obiecte educaționale, pentru a identifica meta-datele disponibile în aceste formate.

3.1.2 Obiecte educaționale

Dezvoltarea tehnologiilor internet a influențat dezvoltarea educației moderne prin integrarea noilor tehnologii informaționale și de comunicare în procesul educațional. Au apărut platforme LMS care să asigure administrarea procesului educațional, a resurselor și utilizatorilor implicați. Pentru a îmbunătăți aceste platforme a fost necesară introducerea unor standarde de realizare a resurselor electronice educaționale [57]. Acestea trebuie să asigure:

- Accesibilitatea – posibilitatea ca obiectele educaționale să fie accesate din mai multe locații.
- Interoperabilitatea – capacitatea de a utiliza o componentă educațională pe mai multe platforme
- Durabilitatea – capacitatea de a fi refolosite o perioadă lungă de timp fără reproiectare sau reconfigurare.
- Reutilizarea – utilizarea componentelor în mai multe aplicații și contexte.

Există două standarde utilizate la scară largă în platformele educaționale: SCORM și IMS. După cum am prezentat și în subcapitolul precedent, Moodle oferă suport pentru ambele standarde. Voi prezenta pe scurt structura pachetelor educaționale construite cu aceste standarde, identificând ce meta-informații se păstrează și în ce formă.

SCORM

Shareable Content Object Reference Model (SCORM) este o colecție de standarde și specificații pentru împachetarea și definirea succesiunii materialului educațional într-o formă partajabilă și reutilizabilă. Pentru a atinge acest deziderat, SCORM propune utilizarea de obiecte partajabile de conținut (SCO) care conțin instrucțiuni și materiale educaționale definite folosind terminologia standardului. Obiectele astfel definite vor fi rulate în aplicații care citesc informația conținută și le redau utilizatorilor. Pot fi integrate în platforme educaționale sau într-o bază de date de obiecte educaționale, care permit o căutare adecvată. Meta-informația conținută în aceste pachete este utilizată pentru localizarea obiectelor conținute [58].

Este susținut și menținut de ADL (Advanced Distributed Learning) o agenție sponsorizată de guvernul Statele Unite, care cercetează și dezvoltă specificații pentru a încuraja adoptarea învățământului electronic.

Un pachet SCORM este de fapt o arhivă zip care conține obiecte livrabile folosind Internetul. Folosind standarde pentru organizarea obiectelor se permite interoperabilitatea între diferite LMS-uri sau baze de date de conținut. Componentele conținute de pachetele SCORM sunt [59]:

- Resurse – sunt reprezentări electronice de conținut media, text, imagini, sunet, pagini HTML, obiecte de evaluare sau alte bucăți de date. Sunt cele mai reutilizabile elemente; pot fi rearanjate, li se poate schimba scopul sau pot fi reutilizate în diferite contexte și aplicații.
- SCO (obiecte partajabile) – cea mai mică unitate logică de informație care poate fi livrată cursantului prin intermediul unui LMS. Utilizează resursele disponibile în pachet. Este singurul element din SCORM care poate comunica cu platforma LMS, obținând informații despre cursant și păstrând datele referitoare la parcurgerea cursului în baza de date a LMS-ului.
- Agregare – este o colecție de activități relaționate. Poate conține obiecte SCO sau alte elemente de agregare. Procesul de agregare este utilizat pentru a grupa conținutul relaționat pentru a putea fi livrat cursanților în ordinea stabilită de tutore. Agregarea nu este un fișier fizic, este o structură în fișierul manifest SCORM.
- Organizare – obiectele SCO sunt structurate arborescent. În acest fel este definită ordinea de livrare. Organizarea într-un pachet SCORM este considerată agregarea de tip rădăcină.

Mai jos este prezentat un exemplu de declarare a unui document manifest SCORM. Pentru definirea lui se utilizează limbajul de marcare XML, în timp ce pentru organizarea conținutului se utilizează specificațiile IMS. Se observă trei secțiuni principale în interiorul nodului părinte: *organization*, în care sunt descrise agregarea și organizarea obiectelor, *resource,* în care sunt introduse resursele disponibile și locația lor, *metadata*, unde se specifică versiunea de SCORM utilizată și alte meta-informații. Aceste meta-date sunt utile pentru căutarea pachetelor SCORM și oferă un mecanism prin care se pot descrie caracteristicile acelui pachet. În cazul cercetării mele, aceste meta-informații prezintă interes pentru a stabili modul de colectare și aliniere a lor cu alte date în format RDF.

```
<manifest>
  <metadata> ... </metadata>
  <organization>
    <item identifier="Agregare1">
      <item identifier="SCO1" identifierref="resursa1">...</item>
      <item identifier="SCO2" identifierref="resursa2">...</item>
    </item>
    <item identifier="SCO3" identifierref="resursa3">...</item>
  </organization>
  <resource>
    <resource identifier="resursa1">...</resource>
    <resource identifier="resursa2">...</resource>
    <resource identifier="resursa3">...</resource>
  </resource>
</manifest>
```

Momentan nu sunt definite cerințe în SCORM pentru introducerea de meta-date și pentru asocierea acestora cu componentele modelului SCORM, aceste informații nefiind necesare realizării unui pachet. Pentru definirea meta-datelor este recomandată utilizarea standardului IEEE LOM [60].

Modelul informațional IEEE LOM descrie elementele care pot fi utilizate pentru a construi meta-date. Modelul este structurat în nouă categorii de elemente [60]:

- Categoria *generală,* poate fi utilizată pentru a descrie informația generală despre pachetul SCORM, ca întreg;
- *Ciclul de viață*, se poate utiliza pentru a descrie istoricul și forma curentă a pachetului SCORM;
- Categoria *meta-metadate,* se poate utiliza pentru a descrie informații despre componenta de meta-date;
- Categoria *tehnică,* se utilizează pentru a descrie caracteristicile și cerințele tehnice ale modelului utilizat;
- Categoria *educațională,* descrie caracteristicile pedagogice ale pachetului SCORM;
- Categoria *drepturi,* utilizată pentru a specifica drepturile de autor și condițiile de utilizare;
- Categoria *adnotări,* utilizată pentru a introduce comentarii cu privire la modul de folosire al componentei și informații cu privire la persoanele care au adăugat comentarii;
- Categoria *clasificare,* descrie modul în care componentele fac parte dintr-un anumit sistem de clasificare.

Toate instanțele de meta-date vor avea un nod părinte <lom> și încapsulează toate categoriile descrise mai sus, fără a fi necesară apariția nodurilor în ordinea descrisă mai sus. Este recomandată declararea spațiilor de nume în elementul lom. În exemplul de mai jos este prezentată structura unui nod de tip lom, pentru care este definit spațiul de nume și pentru care nodurile copil sunt goale.

Fiecare din cele nouă categorii are definit un set de termeni de bază. În cazul în care o organizație consideră că termenii respectivi nu sunt suficient de expresivi pentru modelarea meta-informațiilor, se permite adăugarea de noi termeni.

```
<lom xmlns="http://ltsc.ieee.org/xsd/LOM">
  <general/>
  <classification/>
  <annotation/>
  <lifeCycle/>
  <technical/>
  <metaMetadata/>
  <educational/>
  <relation/>
  <rights/>
</lom>
```

Pentru a introduce meta-datele, SCORM permite declararea lor în fișierul manifest. Se pot adăuga meta-date la orice componentă descrisă în interiorul fișierului manifest. Acestea pot fi:

- Meta-date de agregare a conținutului - acestea sunt informații generale despre organizarea componentelor și se introduc în nodul <metadata> din fișierul manifest. Sunt destinate pentru identificarea resurselor de aplicații din exterior.
- Meta-date de organizare a conținutului - sunt informații despre organizarea componentelor și se introduc în nodurile <organization> din fișierul manifest. Sunt utilizate pentru a identifica resursele în interiorul pachetului.
- Meta-date de activitate - descriu o activitate individuală în nodurile copil <item>, din noul părinte <organization>.
- Meta-date ale elementelor SCO - sunt utilizate pentru a descrie conținutul acestor elemente și pentru a facilita reutilizarea lor.
- Meta-date de resurse - sunt utilizate pentru a introduce informații despre resurse, independent de contextul în care acestea sunt utilizate.

SCORM permite declararea meta-datelor și în fișiere externe. În acest caz, legătura se realizează în fișierul manifest prin nodul <adlcp:location> în care se furnizează locația exterioară a meta-datelor.

```
<manifest>
  <metadata>
  <schema>ADL SCORM</schema>
  <schemaversion>2004 4th Edition</schemaversion>
  <adlcp:location>metadate.xml</adlcp:location>
  </metadata>
</manifest>
```

IMS

IMS (Instructional Management System) Global Consortium este un consorțiu de furnizori și dezvoltatori de resurse educaționale care doresc să stabilească specificații bazate pe XML pentru a descrie caracteristicile cheie ale cursurilor, temelor, cursanților și grupurilor.

Una din cele mai cunoscute specificații propuse de *IMS Global Consortium* este *Common Cartrige*, care definește un format deschis pentru distribuția de conținut în Internet. Este proiectat pentru a asigura instalarea și operarea corectă a conținutului în orice platformă compatibilă IMS CC. Se bazează pe mai multe specificații definite de consorțiu sau de alte organizații [61]:

- *IEEE LOM* – utilizat și în pachetele SCORM.
- *IMS Content Packaging* – utilizat pentru descrierea unei structuri de date pentru a facilita exportul, importul, agregarea de conținut între două sisteme. Utilizat și în SCORM pentru definirea fișierului manifest.
- *IMS Question & Test Interoperability* – este un model de reprezentare a elementelor de test și rezultatelor obținute de cursanți. Permite schimbul de elemente de test și rezultate între diferite unelte de administrare a testelor: bănci de întrebări, unelte de construcție de teste, platforme educaționale, etc.
- *IMS Authorization Web Service* – definește doar modul de comunicare între LMS și distribuitorul pachetului CC.
- *IMS Basic Learning Tools Interoperability* – permite integrare de conținut și unelte externe în LMS-uri.

Ca și în cazul SCORM, resursele sunt împachetate într-o arhivă zip împreună cu fișierul manifest care descrie structura internă de fișiere. O unitate de organizare a resurselor este un dosar. Acesta este o colecție de elemente și dosare copil care sunt așezate într-o anumită ordine. Un dosar este o paradigmă de prezentare a conținutului și poate fi utilizat pentru organizarea și prezentarea conținutului către cursant.

Dosarele pot conține mai multe tipuri de resurse. Resursele de tip web sunt orice fișiere suportate pe scară largă pentru furnizarea de conținut pe Internet. Acestea pot fi fișiere HTML, imagini, audio, video, documente MS Ofice, documente PDF, Flash, etc. Resursa poate fi marcată pentru utilizare într-un plan de lecție, temă, etc., astfel încât platforma educațională să țină cont de această intenție de utilizare. Resursele pot fi și legături web, specificându-se titlu, fereastra în care să fie deschise paginile și dimensiunile acestora.

Activitățile de tip topic de discuții au un tip de resursă proprie. Conțin informații cu privire la titlu, descriere, fișiere atașate și sunt folosite pentru a specifica platformei educaționale ce topic să deschidă în uneltele de comunicare proprii.

Resursele de tip evaluare sunt utilizate pentru a descrie testele și elementele conținute, numărul de încercări permise, limitările de timp și dacă este permisă depășirea acestui timp cu penalizare.

Dosarele mai pot conține și resurse sub forma unor documente XML, care descriu informația necesară integrării de unelte externe folosind specificația IMS LTI. Platforma educațională care citește acest fișier apelează unealta externă pe baza informațiilor conținute în acest document.

Meta-datele se păstrează folosind un profil redus al specificației IEEE LOM, doar cu termenii cei mai utilizați. Acest profil este descris folosind termeni din vocabularul Dublin Core. Spațiul de nume pentru acest profil LOM diferă față de original, prin inserția segmentului *imscc.* Elementele de meta-date pot fi introduse în fișierul manifest sau în alte locații din pachetul IMS CC. Trebuie specificat în fișierul manifest, înainte de introducerea nodurilor de meta-informație, schema și versiunea IMS utilizată, ca în exemplul de mai jos.

```
<metadata>
  <schema>IMS Common Cartridge</schema>
  <schemaversion>1.2.0</schemaversion>
  <lomimscc:lom>
  <lomimscc:general>
  <lomimscc:title>
  </lomimscc:title>
  <lomimscc:description>
  …..
  </lomimscc:general>
  </lomimsc:lom>
</metadata>
```

Analizând cele două cele mai cunoscute specificații pentru realizarea de pachete de resurse educaționale, SCORM și IMS CC, am observat că pentru inserarea meta-informațiilor amândouă converg spre utilizarea unui standard comun IEEE LOM, chiar dacă IMS folosește un profil mai restrictiv. De asemenea, organizarea resurselor se face într-un mod asemănător: împachetează resursele și fișierele de configurare într-o arhivă zip, utilizează un fișier manifest pentru descrierea structurii și modului de furnizare a conținutului către cursanți.

În concluzie, se poate reutiliza standardul IEEE LOM pentru a modela și extrage meta-informații din obiecte educaționale pentru aplicații Semantic Web.

3.2 Utilizarea meta-datelor în educație

În prima parte a acestui subcapitol mi-am propus să analizez metodele necesare pentru a extrage și pune în valoare meta-datele disponibile în platformele educaționale. După descrierea platformelor LMS și a obiectelor educaționale conținute de acestea, este evident rolul meta-informațiilor pentru o bună utilizare a pachetelor SCORM și IMS. Astfel, o direcție de urmărit este dată de modul în care aceste informații pot fi disponibile și pentru aplicații semantic web.

Un prim pas pentru a îmbunătăți marcarea conținutului cu meta-date o reprezintă utilizarea microformatelor. În a doua parte a acestui subcapitol am propus o metodă în care am utilizat aceste specificații pentru a ușura extragerea informațiilor despre utilizatori de către agenți software.

3.2.1 Meta-date educaționale în formate RDF

Specificațiile de declarare de meta-date au fost dezvoltate pentru a permite atât interoperabilitate între diferiți agenți software, cât și o mai bună identificare a resurselor educaționale de către oameni. Sunt două standarde acceptate pe scară largă în mediul educațional, IEEE LOM și DC. Acestea pot fi catalogate ca și meta-date standard [62], fiind limitate doar la definirea și structurarea termenilor, ierarhic. O categorie intermediară este reprezentată de meta-datele semi-semantice, acestea fiind profile de IEEE LOM extinse cu anumite componente semantice. Din ultima categorie fac parte meta-datele semantice, descrise complet în RDF; acestea folosesc modele derivate din IEEE LOM sau modele noi.

O primă abordare pentru a transforma meta-datele existente în format XML LOM în triplete RDF o constituie realizarea unui fișier de legătură pentru definirea

termenilor în RDF [63]. Folosind limbajul RDFS, se definesc termenii din IEEE LOM utilizând pe cât posibil vocabulare deja existente (DC, vCard).

Un grup de lucru format din membri ai celor două organizații care administrează standardele DC și IEEE, DCMI (Dublin Core Metadata Initiative) și IEEE LTSC (Learning Technology Standard Committee) au semnat un acord prin care se angajează să realizeze un vocabular RDF pentru elementele LOM [64].

În alte abordări, autorii doresc să renunțe la IEEE LOM. Una dintre acestea este cea desfășurată în laboratoarele ARIES (Advanced Research in Intelligent Educational System) din Canada, în care se renunță la specificațiile existente pentru o abordare mai "ecologică" (deducerea meta-datelor prin urmărirea interacțiunilor utilizatorilor cu obiectele educaționale) [65]. Informațiile cu privire la cursant, interacțiunea cu conținutul educațional, evaluarea cursantului, etc., sunt atașate în interiorul obiectului educațional pentru instanța cursantului respectiv. Datele strânse de la toți utilizatorii formează un raport colectiv al experiențelor cursanților în interacțiunea cu obiectul educațional. Acest raport poate fi inspectat pentru identificarea tiparelor utilizatorilor care nu au rezultate satisfăcătoare. Informaţiile se păstrează folosind modele RDF. Autorii văd această abordare ca o extensie a pachetului SCORM.

Proiectul EDUTELLA a avut ca scop realizarea infrastructurii necesare în sisteme educaționale P2P (Peer to Peer), folosind RDF [66]. Structura infrastructurii propuse conține: serviciu de interogare; serviciu de replicare; serviciu de mapare, care asigură interoperabilitatea între diferite vocabulare de meta-date; serviciu de adnotare. Pentru a oferi aceste servicii de meta-date se utilizează variante RDF ale specificaților IEEE LOM și IMS.

Un exemplu de utilizare a meta-datelor obiectelor educaționale în format RDF o reprezintă realizarea integrării resurselor educaționale partajate din diferite surse, folosind principiile Linked Data [67]. Aceasta presupune convertirea datelor LOM extrase din resursele partajate în format RDF, stocarea lor într-o bază de cunoștințe, aplicarea de mecanisme de îmbogățire a datelor prin interogări pe datele Linked Open Data.

Utilizarea datelor LOM RDF, în contextul unei model de LMS semantic, este argumentată de R. Balog-Crisan și I. Roxin în lucrarea [68]. Sunt argumentate beneficiile aduse cursanților: o posibilă platformă educațională care să integreze meta-date RDF ar crește adaptabilitatea acesteia la nevoile utilizatorilor, având un potențial de personalizare ridicat.

3.2.2 Meta-date în LMS. Caz particular Moodle

Ca şi primă abordare pentru introducerea de meta-date în context educațional, am utilizat microformatele pentru a structura bucăți mici de informație, care totuși să ajute la îmbunătățirea serviciilor oferite.

Extragerea de informații din pagini web se bazează în prezent pe utilizatorul uman. În ultimul timp, tendința a fost de a veni în ajutorul acestuia prin diverse soluții automatizate de prelucrare a datelor. Acestea se bazează pe standarde de publicare a datelor (de exemplu microformatele și RDFa), pentru a păstra semnificația informației și structura conținutului. Extragerea de informații corecte (datele de contact, evenimente de calendar, locații) este la fel de importantă, dacă nu chiar mai importantă în învățământului electronic. În acest subcapitol voi prezenta metode de publicare a conținutului în LCMS-ul Moodle. Abordarea aleasă presupune construcția unui bloc Moodle, în care sunt utilizate pentru publicare standardele hCard, vCard și codurile QR de marcare pentru dispozitive mobile. Acest

model permite extensii viitoare pentru alte tipuri de informații (evenimente calendar, locații, etc.).

Dintre LCMS-urile în sursă deschisă, Moodle este foarte utilizat, permițând universităților să implementeze cursuri electronice rapid și ieftin. Pachetul software se poate descărca și instala pe orice calculator, având capacitatea de a se adapta ușor diferitelor cerințe, de la sisteme cu un singur tutore, la universități cu mii de studenți. Studiile întocmite în universitatea noastră, cu privire la gradul de utilizare al acestui LCMS [69], relevă faptul că este printre cele mai utilizate astfel de soluții software, atât în țară, cât și în străinătate.

Moodle, ca și orice alt sistem de management de conținut, folosește Web-ul pentru a administra și distribui conținut. După cum am arătat în capitolul precedent, Web-ul evoluează lent, dar constant, spre a adăuga cunoștințe în conținut; astfel, pare firesc să se urmeze această tendință și în mediile educaționale. Microformatele și RDFa sunt un prim pas important spre acest deziderat. În continuare, am prezentat metode de integrare a acestora în platforma educațională utilizată în Universitatea "Politehnica" din Timișoara.

Microformatele sunt standarde simple pentru publicarea inteligentă a informației pe Web. Numărul de pagini web care utilizează hCard – unul dintre microformate – a depășit 2 miliarde [70]. Predecesorul hCard, formatul vCard, este cel mai utilizat în cărțile de vizită electronice și aplicațiile de stocare a datelor cu caracter personal.

Codurile QR sunt coduri de bare bidimensionale, care au început să fie din ce în ce mai utilizate în ultima vreme, datorită numărului mare de dispozitive mobile inteligente. Deși inițial au fost utilizate pentru urmărirea pieselor în producția de mașini, au început să fie utilizate într-un context mai larg, atât în aplicații comerciale, cât și ca etichete citibile de dispozitivele mobile.

Atât codurile QR, cât și vCard-urile, sunt acceptate de majoritatea sistemelor de operare mobile, direct sau prin intermediul unor aplicații software.

Tabel 4. Suportul pentru coduri QR și formatul vCard în sisteme de operare pentru dispozitive mobile

Sistem de operare pentru mobil	Cotă de piață 2010 [71]	Suport QR	Suport vCard
Symbian	37.6 %	Da	Da
Android	22.7 %	Da	Da
RIM	16 %	Da	Da
IOS	15.7%	Da	Da
Microsoft	4.2 %	Da	Da
Alte SO	3.8 %	x	x

Procesul de extragere de informații din aplicația Moodle este destul de laborios. De exemplu, dacă se dorește extragerea datelor de contact ale unui tutor (nume, adresă, e-mail, telefon), utilizatorul trebuie să navigheze până la informația respectivă și să o introducă manual într-o agendă electronică, cum ar fi Microsoft Outlook sau Google Contacts. Acesta înseamnă că aplicația nu este foarte ușor de utilizat. Pentru a îmbunătăți utilizabilitatea, mi-am propus să definesc o metodă de publicare a datelor de contact ale tutorilor folosind tehnologiile de mai sus (vCard,

hCard și coduri QR). Cu o extensie de navigator web, de exemplu Operator pentru Mozilla sau un telefon mobil cu cameră și cititor QR instalat, studenții pot salva aproape instantaneu datele în aplicații precum Outlook.

Există două abordării complementare cu privire la utilizabilitate [72]:

- "Bottom-up" – este o viziune orientată spre produs, care identifică utilizabilitatea cu ușurința în folosire;
- "Top-down" – unde utilizabilitatea reprezintă capacitatea de a folosi un produs pentru scopul propus.

În acest caz, am folosit prima variantă, care descrie utilizabilitatea din punct de vedere al ingineriei software, în timp ce a doua o descrie din punct de vedere a factorilor umani.

În realizarea procesului de proiectare am luat în calcul evaluarea primelor microformate implementate în portalul CSID (Centrul de Studii prin Învățământ la Distanță), care atestă utilitatea acestora, dar le consideră dificil de utilizat și integrat în rutina zilnică [73]. Pentru a corecta neajunsurile identificate, am utilizat pe lângă microformatele propriu zise și imagini QR, pentru a permite importul rapid al datelor în telefoane mobile.

vCard. hCard. RDFa

Microformatele apar în acest moment ca o modalitate de a atenua decalajul dintre Web-ul prezent și cel din viitor. Acestea sunt concepute atât pentru oameni, cât și pentru mașini, funcționând cu unelte și navigatoare existente. Microformatele sunt metode simple, bazate pe standarde existente (HTML și CSS), folosite pentru a adăuga mai multe cunoștințe în paginile web, cu scopul de a indica mai bine oameni, companii, evenimente, recenzii, etichete și altele [74]. hCard este cel mai utilizat microformat, fiind o reprezentare 1 la 1 a vCard în format HTML.

vCard-ul este o carte de vizită electronică; automatizează schimbul de informații cu caracter personal (nume, adresă, numere de telefon, adrese de e-mail, elemente grafice, informații de localizare) între diferite aplicații [75].

O serie de aplicații au implementat microformate [76]: Digg, Drupal, Eventful, Facebook, Flickr People și Photos, Google (Chrome, Search, Blogger, Creative Commons Search, Maps), Internet Explorer, LinkedIn, Magnolia, phpMicroformats, Technorati (Contact Feed Service, Events Feed Service, Microformats Search, Search, Tags), Upcoming, WordPress, Yahoo (Creative Com-mons Search, Local, Tech, UK Movies), Pingerati, etc.

Ultimele versiuni de navigatoare web, de exemplu Firefox începând de la versiunea 3.0, au suport integrat pentru microformate, iar versiunile mai vechi se pot adapta folosind extensii de genul Operator, BlueOrganizer, Tails sau Tails Export. Internet Explorer 7 și 8 are disponibil Oomph pentru microformate, în timp ce Chrome are extensia Michromeformats. Alte navigatoare care oferă suport sunt Flock și Safari [77]. De obicei, extensiile convertesc hCard în vCard, oferind astfel posibilitate de import în aplicații desktop.

RDFa a fost descris pe larg în capitolul precedent. Dezavantajul acestui format este dat de limitarea la utilizarea în documente xHTML 1.1.

Coduri QR

Codurile QR sunt coduri de bare bidimensionale care pot fi citite de telefoane mobile inteligente sau un calculator. Au apărut din necesitatea de încapsulare a unei cantități de informații mai mari decât posibilitatea oferită de codurile de bare standard și au fost lansate în 1994.

Primii pași au fost făcuți de Bernard Silver în 1948, după ce a auzit o conversație între un decan al Drexel Institute of Tehnology din Philadelphia și președintele unui lanț alimentar, în care cel din urmă argumenta necesitatea cercetării unui mod de captare a informațiilor produselor în mod automat la finalizarea comenzii. Patentul pentru coduri de bare a fost acordat în 1952 lui Woodland și Silver, iar în 1962 a fost vândut către Philco. Primul sistem cu coduri de bare a fost instalat de Computer Identics în 1969, iar în 1974 a fost vândut primul produs cu ajutorul unui cititor de coduri de bare [78].

Figura 8. Exemplu de cod QR

Codul de bare clasic pe o singură dimensiune, conține informații pe o singură direcție, în timp ce un cod de bare pe două dimensiuni, conține informații pe verticală cât și pe orizontală. Printre cele mai cunoscute coduri de bare bidimensionale sunt QR (Quick Respons), Data Matrix și codurile Aztec. Codurile QR sunt foarte utilizate în Asia (panouri publicitare, stații de autobuz, reclame), dar au început să fie răspândite și în Europa [79].

Codurile QR sunt capabile să codeze orice tip de date (caractere numerice sau litere, simboluri, date binare, coduri de control, etc.) în aproximativ o zecime din spațiul pe care l-ar utiliza codurile de bare clasice. Au capacitatea de corectare a erorilor și sunt ușor de citit din orice direcție.

Tehnologia codurilor de bare bidimensionale a început să fie utilizată și în mediul educațional pentru activități specifice unui anumit context [80] sau adnotări în materiale video pe dispozitive mobile [81].

Utilizarea hCard, vCard și coduri QR în Moodle

După cum am mai precizat, în Moodle procesul de salvare a datelor de contact ale unui tutor într-o agendă electronică este dificil. În primul rând, utilizatorul trebuie să selecteze tutorele în cauză, apoi este redirecționat într-o pagină cu informația dorită, pe care trebuie să o extragă manual și să o introducă în aplicația dorită. Procesul este descris în figura 9.

Acest proces este laborios și poate duce la pierderea de informații. Distribuția corectă a informațiilor referitoare la datele de contact ale tutorilor și evenimentele de calendar este cea mai importantă parte dintr-o platformă de eLearning.

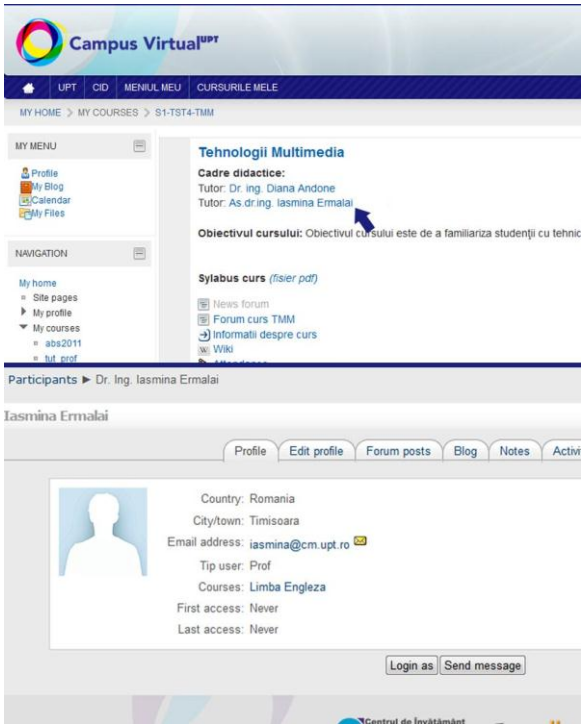


Figura 9. Publicarea informațiilor de contact în sistemul inițial

Am propus și implementat un bloc Moodle care extrage informațiile din baza de date Moodle și le face accesibile utilizatorilor și aplicațiilor în trei forme diferite: folosind hCard, vCard și coduri QR (figura 10). Am folosit stratul de legătură cu baza de date a platformei Moodle, în acest fel asigurând portabilitate spre versiuni ulterioare ale LMS-ului în sursă deschisă. Informația este apoi procesată și exportată în formatele hCard și vCard.

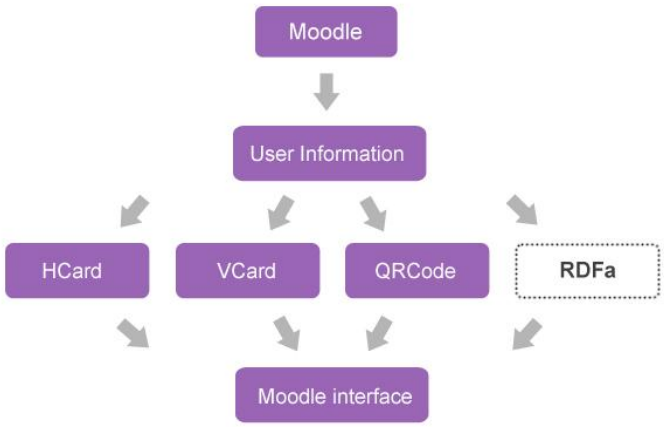


Figura 10. Diagrama de procesare a informației

La prima vedere nu este nici o diferență între varianta inițială și cea nouă – în dreptul numelui tutorelui apare cuvântul vCard cu o legătură pe el (Figura 11). Prin utilizarea legăturii apare o fereastră conținând informațiile despre tutore, etichetate folosind hCard, o legătură pentru descărcarea datelor în format vCard și codul QR. Informația este încărcată folosind o procedură AJAX (JavaScript asincron și XML), pentru a nu consuma resurse de pe server decât dacă informația este cerută.

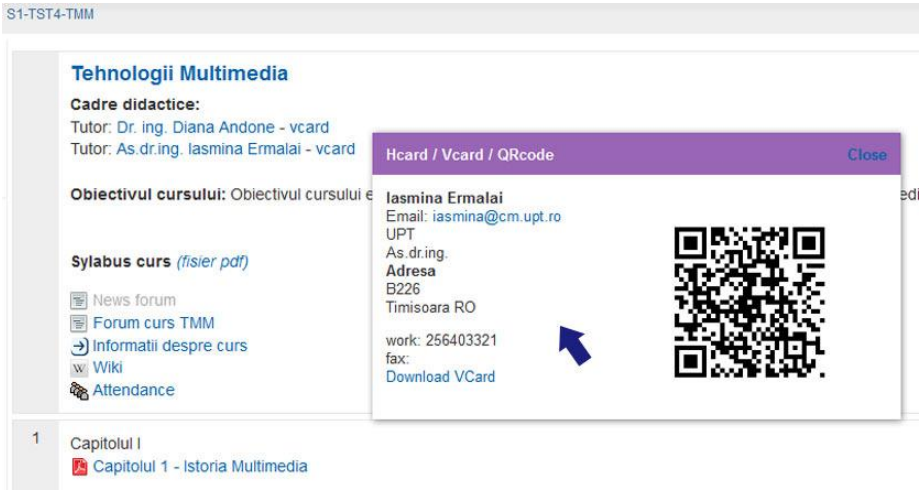


Figura 11. Publicarea informației de contact a tutorului

Generarea codurilor QR

Codurile QR pot fi citite de telefoane mobile cu cameră foto și un cititor de coduri QR instalat. Rezultatul acestui proces este o carte de vizită electronică a tutorelui, conținând informația extrasă din baza de date Moodle, informație care acum se poate ușor salva în agenda electronică a dispozitivului mobil.

Pentru a îndeplini această sarcină am avut două posibilități: documentul vCard generat se putea coda direct în imaginea QR sau se putea coda doar legătura spre acest document. Dacă în viitor se dorește adăugarea mai multor informații în vCard, imaginea QR ar deveni mai mare, existând o limită de dimensiune care nu ar mai putea fi afișată pe ecran. Din această cauză, am decis să utilizez a doua soluție pentru implementare.

Pentru a coda informația dorită în imagini QR, se poate utiliza unealta Google Chart. Este un API gratuit (Application Programming Interface), care generează grafice din date furnizate în URL. Principalul avantaj în utilizarea unui API extern este reducerea puterii de procesare necesare.

Google API este soluția ideală la momentul implementării acestui bloc, deoarece limita de accesări este de 250.000 pe zi, iar platforma Moodle din universitatea noastră are doar 5.000 de accesări pe zi. Când numărul de vizite va depăși limita expusă mai sus sau dacă se va dori o independență față de serviciile Google, se poate utiliza următoarea metodă: se generează o imagine QR pentru fiecare persoană, imagine care apoi poate fi stocată în baza de date sau direct pe server. Acest proces este posibil datorită faptului că informația păstrată în imaginea QR este o legătură cu parametri specifici fiecărui utilizator.

Generarea vCard și hCard

Documentele vCard pot fi exportate dinamic din sistem, prin accesarea unei legături interne care conține variabile necesare obținerii informației de contact a utilizatorului din baza de date. Structura aplicației îi permite să se comporte ca un API pentru alte aplicații software, asigurând astfel schimbul ușor de date între Moodle și alte aplicații web.

O clasă PHP a fost utilizată pentru generarea de documente vCard. Aceasta generează formatul dorit prin procesarea unui tablou asociativ care conține informațiile de contact ale tutorului, extrase din baza de date. Clasa conține două metode: build(), folosită pentru generarea vCard-ului pe baza informației furnizate, și download(), pentru a genera fișierul cu extensia .vcf și anteturile corecte.

De asemenea, am folosit formatul hCard pentru a afișa informații de contact ale tutorelui în fereastra deschisă. Beneficiile utilizării acestui format sunt:

- Motoarele de căutare pot extrage mult mai ușor informație structurată
- Folosind extensii precum Operator pentru Mozilla (figura 12), sunt posibile o serie de acțiuni – salvarea informației în Outlook (figura 13), Yahoo! Contacts, localizarea poziției folosind hărți online, etc.

Pentru a publica informațiile extrase din baza de date în format hCard, am folosit un șablon standard. Pentru automatizare am construit o clasă PHP, conținând toate metodele necesare generării de microformate în viitor. Momentan am utilizat doar metoda createHCard(), care evident generează hCard.

Folosind AJAX pentru a încărca și genera fereastra, extensia Operator nu poate accesa informațiile din fereastra curentă. La o a doua cerere, hCard-ul vechi va fi înlocuit cu cel curent.

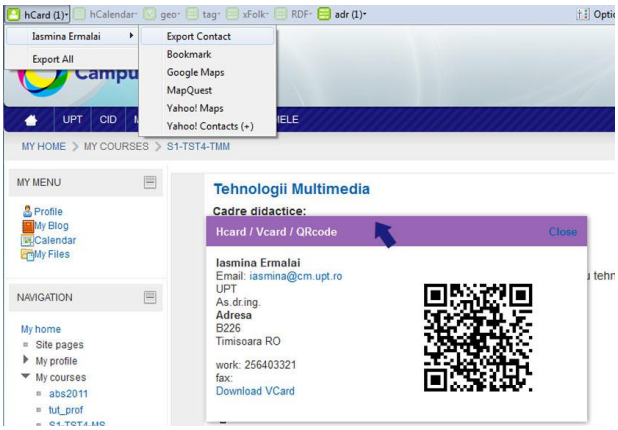


Figura 12. Citirea hCard-ului de către Operator

Interfața standard Moodle nu permite o extragere ușoară a informațiilor de contact, ducând la posibile alterări sau pierderi de informații. Pentru a rezolva această problemă și pentru a îmbunătăți astfel utilizabilitatea aplicației, propun utilizarea următorilor pași:

- Utilizarea stratului de abstractizare a legăturii cu baza de date Moodle pentru extragerea informației de contact; în acest fel se asigură portabilitatea blocului spre versiunile ulterioare.

- Generarea de vCard-uri și hCard-uri pentru acces ușor și rapid la informația de contact a tutorelui.
- Utilizarea codurilor QR pentru importul rapid al datelor în agenda electronică a telefoanelor mobile
- Compactarea acestor facilități într-un bloc, pentru a nu modifica structura aplicației.

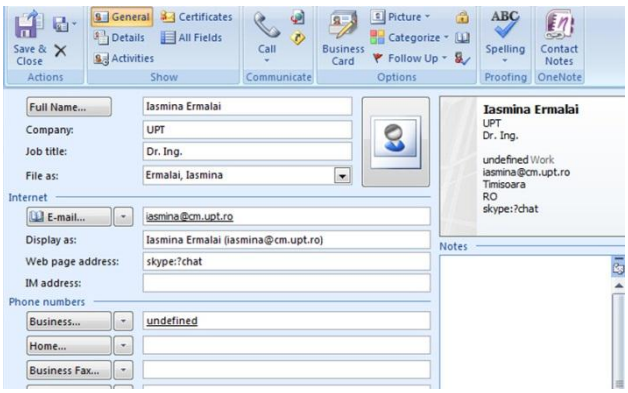


Figura 13. Salvarea informației în Outlook

O altă metodă de publicare de informații, atât pentru oameni cât și pentru mașini, este utilizarea RDFa. Din păcate nu se poate utiliza decât în documente xHTML 1.1, acesta constituind un impediment, deoarece momentan interfața utilizată în Moodle este xHMTL 1.0. O soluție ar fi modificarea temei Moodle pentru a respecta acest standard.

Chiar dacă microformatele permit etichetarea ușoară a informațiilor din documente HTML, acestea nu se pot extinde (tehnic se pot declara microformate proprii, dar nu este recomandat). Dacă în viitor trebuie inserate date care nu sunt definite în microformatul hCard, va trebui utilizat RDFa.

Rezultatele acestor cercetări (metoda de îmbogățire a datelor de contact din Moodle folosind microformate și coduri QR, descrierea implementării blocului Moodle, discuții cu privire la posibile extinderi a metodei), au fost publicate în [82][83]. Această metodă de publicare se poate reutiliza și pentru evenimente și informații de localizare, utilizând microformatele adecvate.

3.3 Contribuții și concluzii

Am studiat principalele concepte utilizate în învățământul electronic, axându-mă pe platformele de gestionare a conținutului educațional. Am identificat structura datelor gestionate în aceste platforme. Acestea se pot structura în patru secțiuni: cursuri (organizarea materialelor educaționale), activități (toate modulele implicate în procesul educațional), evaluare (uneltele de evaluare pot fi considerate și ca activitate de curs) și utilizatori (administrarea participanților și privilegiilor pe care aceștia le au în sistem). Acest studiu reprezintă de asemenea şi prima contribuţie din acest capitol, fiind necesar în pasul următor de proiectare a unui model ontologic.

Studiul asupra specificaţiilor care permit introducerea de meta-date în obiecte educaţionale a fost necesar pentru a identifica eventualele meta-informaţii deja disponibile în platformele educaţionale şi reprezintă a doua contribuţie teoretică din acest capitol.

Analizând cele mai utilizate specificații pentru realizarea de pachete de resurse educaționale, SCORM și IMS CC, am observat că pentru inserarea meta-informațiilor ambele converg spre utilizarea unui standard comun IEEE LOM, chiar dacă IMS folosește un profil mai restrictiv. De asemenea, organizarea resurselor se face într-un mod asemănător: împachetează resursele și fișierele de configurare într-o arhivă zip, utilizează un fișier manifest pentru descrierea structurii și modului de furnizare a conținutului către cursanți.

Specificațiile IEEE LOM au variante de definire a termenilor folosind limbaje specifice Web-ului Semantic. Datorită acestui fapt și a gradului mare de utilizare, nu consider necesară o reproiectare a structurii meta-datelor din interiorul obiectelor educaționale SCORM sau IMS, aceste informații fiind ușor de extras în format RDF.

Datele care nu pot fi ușor utilizate în aplicații semantice sunt cele conținute în cele patru categorii prezentate mai sus. Pentru a folosi aceste informații păstrate în platformele LMS, în capitolul următor voi propune un model de bază pentru modelarea acestor date folosind termeni definiți în RDF.

De asemenea, am prezentat o primă încercare de a îmbunătăți serviciile oferite într-un LMS, utilizând microformate pentru publicarea de structuri mici de date. Astfel, am realizat un modul destinat LMS-ului Moodle care să permită publicarea informațiilor de contact ale utilizatorilor în format hcard, ușor de citit de agenți software care extrag sau indexează date. De asemenea, am propus utilizarea codurilor QR pentru a facilita importul datelor de contact direct în agenda telefonului mobil. Această metodă de *îmbogăţire* a datelor, a fost publicată în două articole - [82] [83] - şi reprezintă de asemenea şi o primă contribuţie aplicativă a prezentei teze.

Construcția de servicii avansate necesită publicarea unor structuri de date mai complexe şi posibilitatea interogării datelor respective. Microformatele nu permit acest deziderat, astfel încât este necesară modelarea datelor folosind o ontologie Semantic Web, iar publicarea datelor se poate face folosind standardul RDFa. Pentru a permite partajarea și interoperabilitatea datelor între mai multe surse, voi utiliza tehnologiile Web-ului Semantic. Prin urmare, este necesară stabilirea ontologiei utilizate în modelarea datelor educaționale. Acest proces este descris în capitolul următor.

4 Studiul și implementarea unei ontologii educaționale

În acest capitol am prezentat modelul de ontologie propus pentru a modela datele din sisteme electronice educaționale, pentru a le face disponibile în Web-ul Semantic. Am identificat în primă fază metodologia necesară proiectării unei ontologii, având în vedere limitările impuse de sistemele educaționale. Urmând această metodologie, am încercat să ajung la un model de ontologie de bază, ușor extensibilă și adaptabilă pentru necesități de publicare viitoare.

4.1 Modelare semantic web

În ultimii ani, popularitatea crescută a Internetului a dus la apariția de noi modalități de învățare, bazate pe unelte și aplicații educaționale. În acest context este necesară găsirea unor soluții pentru reutilizarea resurselor, fiind necesare specificații pentru încorporarea de meta-date pentru a putea reprezenta conținutul educațional, resursele dintr-un LMS sau diferitele scenarii pedagogice.

Sistemele educaționale electronice deseori iau forma unor platforme online, care permit utilizatorilor accesul la date generale și materiale educaționale. Această alternativă la practicile educaționale tradiționale s-a dovedit de un real succes, prin îmbogățirea mediului de lucru cu aplicații online și comunități de utilizatori care colaborează în scopuri educaționale.

Specificațiile pentru modelarea conținutului educațional reprezintă modele de reprezentare și agregare a informației, care descriu din punct de vedere pedagogic conținutul și activitățile educaționale [8]. Aceste elemente sunt de obicei organizate în unități de învățare cu scopul de a permite reutilizarea și interoperabilitatea lor [84].

Printre cele mai cunoscute limbaje pentru modelarea conținutului educațional se găsesc:

- **IMS LD** [http://www.imsglobal.org/learningdesign/] – Descrie o metodă compusă dintr-o serie de activități care permit elevului și tutorelui să atingă anumite obiective educaționale. Permite combinarea mai multor tehnici educaționale (tradițională, colaborativă, etc.) și facilitează descrierea altora noi.
- **IEEE LOM** [http://ltsc.ieee.org/wg12/] – Este o specificație care descrie aspecte tehnice referitoare la e-Learning, creată și sprijinită de IEEE Learning Technologies Standards Committee. LOM este aproape identică cu specificația IMS și compatibilă cu meta-datele Dublin Core [85]
- **SCORM** [http://www.adlnet.gov/] – Este o colecție de specificații propuse de Advanced Distributed Learning Initiative (ADL), care permite partajarea, împachetarea și distribuția obiectelor educaționale [86].
- **DC** - Standardul de meta-date Dublin Core este un set simplu, dar eficient, de elemente, pentru a descrie o gamă largă de resurse partajate în rețea. Standardul cuprinde cincisprezece elemente, care au fost stabilite prin consens de către o organizație internațională, un grup de profesioniști din mai multe discipline, bibliotecari, informaticieni, specialiști în codarea textului, precum și alte domenii conexe [87]. Datorită versatilității și simplității cu care a fost conceput, este indicat în utilizarea și adăugarea de meta-date în obiecte educaționale.

Aceste specificații sunt utile în descrierea resurselor educaționale (conform studiului din capitolul 3), astfel facilitând interoperabilitatea și reutilizarea obiectelor între diferite platforme educaționale. Totuși, aceste specificații sunt în limbaj natural, fiind ușor de interpretat de o ființă umană, dar dificil de procesat automat printr-un sistem software. Pentru a se rezolva această problemă, se pot utiliza ontologii pentru a descrie într-un mod formal și explicit structura și sensul elementelor care conțin meta-date.

Pe lângă obiectele educaționale, o platformă LMS va mai păstra o serie de resurse necesare gestionării utilizatorilor și pentru controlul accesului, organizarea conținutului, informații referitoare la statistica de utilizare a platformei, legătura cu alte aplicații software, etc. O parte din aceste date ar fi util să fie disponibile într-un format ușor de partajat și prelucrat, pentru a putea permite reutilizarea informației în diferite aplicații sau realizarea unor agenți software care să furnizeze facilități suplimentare.

Ontologiile pot fi foarte utile când vine vorba de organizare de conținut. Datorită absenței structurării materialelor educaționale disponibile pe Web, conținutul educațional este mai greu accesibil. Prin crearea unor ontologii bine definite și a sistemelor aferente, identificarea unei anumite resurse educaționale poate fi o sarcină ușoară pentru o aplicație care interoghează date semantice.

Ontologiile au o serie de potențiale beneficii și aplicații în învățământul superior, cum ar fi: partajarea informației între diferite sisteme educaționale, reutilizarea obiectelor educaționale, suport inteligent și personalizat pentru studenți, etc. Dificultățile inerente în crearea unui model de domeniu sunt abordate de comunitățile implicate în dezvoltarea ontologiilor. Acestea comunităţi conlucrează pentru a atinge viziunea Web-ului Semantic. [88]

Web-ul Semantic se bazează pe ontologiile formale, care structurează datele cu scopul de a le face ușor de transportat și înțeles de către mașini. De aceea, succesul Web-ului Semantic depinde puternic de proliferarea ontologiilor, care necesită o proiectare rapidă și simplă şi în același timp evitarea efectului de sugrumare în achiziția cunoștințelor. [89]

Având în vedere cele expuse mai sus, mi-am propus să proiectez o ontologie Semantic Web, care să permită formalizarea meta-datelor și a datelor dorite spre partajare într-un sistem educațional. Pentru a îndeplini această sarcină, inițial am stabilit metodologia de proiectare, identificând cea mai bună abordare pentru situația de față.

4.1.1 Clasificare ontologii

În momentul de faţă există mai multe tipuri de ontologii. Cuvântul ontologie poate reprezenta diferite obiective, în funcție de context. De exemplu, poate fi un tezaur de termeni în domeniul extragerii informației, un model reprezentat în OWL în contextul linked-data sau o schemă XML în domeniul bazelor de date. Pentru a putea stabili metodologia cea mai potrivită pentru proiectare, în prima fază este necesară identificarea tipului ontologiei.

Conform Roussey și alții [90] ontologiile pot fi clasificate după nivelul de expresivitate și formalitate (în funcție de capacitatea de a descrie concepte, proprietăți, instanțe, axiome, etc.) și după scopul ontologiei.

După nivelul de expresivitate și formalitate, ontologiile pot fi [90]:

- **Ontologii de informații** – sunt compuse din diagrame și schițe utilizate pentru a clarifica și organiza ideile colaboratorilor în dezvoltarea unui proiect. Sunt ușor de modificat, scalabile, schematice și sunt utilizate cu preponderență în procesul de design dintr-un proiect. Ele introduc concepte, instanțe și relațiile dintre acestea, cu scopul de crea o privire de ansamblu asupra stadiului curent dintr-un anumit proiect. Acest tip de ontologie este descris de obicei folosind limbaje vizuale, pentru a fi ușor de înțeles de către oameni (exemplu: Mind Map). Sunt utilizabile în colaborarea între oameni, nu pentru a permite ”colaborarea” între mașini.
- **Ontologii lingvistice** – pot fi glosare de termeni, dicționare, vocabulare controlate, taxonomii, folksonomii, tezaure sau baze de date lexicale. Termenii pot fi ambigui, de exemplu ”bancă” poate reprezenta un obiect de mobilier sau o instituție financiară. Astfel, rolul ontologiilor ligvistice constă în a defini vocabularul utilizat și a stabili un acord (în comunitatea care le utilizează) asupra termenilor utilizați pentru a descrie un concept. Acest al doilea proces se numește normalizarea vocabularului. În cazul în care un concept poate fi descris de doi termeni sinonimi, prin procesul de normalizare se alege doar unul, pentru a nu exista confuzii. Taxonomiile și tezaurele își organizează conceptele folosind relații de echivalență, ierarhizare și asociere. Cel mai cunoscut limbaj pentru declararea de tezaure este SKOS (Simple Knowlege Organization System). Acest tip de ontologii se pot utiliza în cazuri care nu necesită introducerea de constrângeri și reguli logice avansate.

- **Ontologii software** – utilizate în activități de dezvoltare de soft, pentru a garanta consistența, stocarea și manipularea datelor. Introduc concepte care conțin un set de proprietăți, iar conceptele au relații între ele pe baza anumitor constrângeri. Ontologiile software sunt descrise folosind limbaje de modelare conceptuală utilizate în ingineria software și a bazelor de date. Cel mai cunoscut limbaj este UML (Unified Modeling Language). Acest tip de ontologie se poate utiliza pentru descrierea unui anumit sistem, impunând restricțiile necesare formalizării modului de lucru în interiorul acestuia.
- **Ontologii formale** – necesită o semantică clară a limbajului utilizat pentru definirea conceptelor, o motivație clară în adoptarea distincțiilor între concepte și reguli clare în definirea relațiilor dintre concepte. Pentru a îndeplini aceste limitări se utilizează logica descriptivă. Astfel, bazele de cunoștințe sunt sisteme formale care păstrează sensul vocabularului adoptat prin definiți logice. Scopul acestor sisteme nu este doar păstrarea și extragerea informației, ci și posibilitatea de a deduce informație nouă din cea furnizată. Ontologiile formale nu sunt focusate pe definirea termenilor, cu toate că aceştia pot fi definiţi. Termenii sunt utilizați ca simboluri pentru a ajuta utilizatorul în procesul de manipulare a formulelor logice. Cel mai cunoscut limbaj pentru implementarea ontologiilor formale este OWL, prezentat în capitolul 2 din această lucrare. Aceste ontologii sunt indicate în construcția de modele complexe care să permită pe lângă standardizarea modului de stocare și partajare a informației, capacități de inferențiere.

După al doilea criteriu de clasificare, și anume scopul ontologiei, acestea se împart în [90]:

- **Ontologii locale** – sunt ontologii specializate, care nu permit partajarea cunoștințelor. Sunt modele particulare realizate prin perspectiva unui utilizator sau dezvoltator. Acestea conțin cunoștințe necesare pentru a îndeplini o sarcină într-o aplicație.
- **Ontologii de domeniu** – sunt aplicabile doar pe un anumit domeniu, cu o perspectivă specifică. Ontologiile de domeniu sunt de obicei legate de o anumită aplicație: un exemplu ar fi administrarea unei platforme educaționale.
- **Ontologii de bază** – este un standard utilizat de diferite grupuri de utilizatori. Acest tip de ontologie este legată de un anumit domeniu, dar integrează puncte de vedere ale diferitelor grupuri de utilizatori. O astfel de ontologie rezultă prin integrarea mai multor ontologii de domeniu, încercând să descrie conceptele și relațiile de bază.
- **Ontologii generale** – nu sunt dedicate unui anumit domeniu, conțin cunoștințe generale dintr-o arie mare.
- **Ontologii fundamentale** – sunt ontologiile generice aplicabile în diferite domenii. Ele definesc cunoștințe de bază, cum ar fi: obiecte, relații, procese, etc.

Într-o platformă educațională pe lângă stocarea și furnizarea obiectelor educaționale, sistemul trebuie să permită o serie de alte facilități: unelte de comunicare, controlul accesului, monitorizarea procesului educațional, generarea de statistici, etc. Unele din aceste componente ar putea fi modelate folosind ontologii lingvistice, altele ontologii formale.

Conform celui de-al doilea criteriu, ontologiile pretabile pentru modelare sunt cele de domeniu. Acestea pot fi utilizate pentru a descrie concepte specifice platformelor educaționale, în timp ce ontologiile de bază, generale sau fundamentale pot fi utilizate pentru a descrie concepte care nu sunt specifice. Respectând această regulă, datele modelate folosind această ontologie vor fi mult mai ușor de partajat și aliniat cu datele din alte sisteme.

Un exemplu elocvent pentru cazul în care este recomandată utilizarea unei ontologii generale este descrierea conceptului de persoană, cu toate proprietățile adiacente. În acest caz se poate utiliza o parte din ontologia FOAF, standardizată, descriind suplimentar doar conceptele de domeniu.

Pentru ca sistemele care vor utiliza această ontologie să poată interacționa cu alte aplicații, este necesară reutilizarea vocabularelor sau ontologiilor dintr-un anumit domeniu, dacă acestea există. Multe ontologii sunt disponibile în formă electronică și pot fi importate și reutilizate [91]. Curentul actual în proiectarea ontologiilor este de a reutiliza pe cât posibil resursele deja existente, permițând în acest fel o partajare mai ușoară a datelor [92].

Pe baza analizei de mai sus am stabilit următorii paşi în procesul de proiectare a unei ontologii: împarţire ontologie în sub-ontologii, identificare vocabulare/ ontologii care să descrie majoritatea conceptelor necesare, extindere sau proiectare sub-ontologie.

Pentru a putea realiza proiectarea și implementarea, este necesară stabilirea metodologiei utilizate în acest proces.

4.1.2 Metodologia de proiectare.

Până în anul 2009 cele mai utilizate metodologii de proiectare a unei ontologii erau METHONTOLOGY, On-To-Knowledge și DILIGENT [93]. Aceste metodologii conțin ghiduri pentru construcția unei ontologii, de la specificații, la implementare. În contrast cu acestea, metodologia NeOn sugerează direcții de abordare pentru scenarii diferite, comparativ cu sistemul rigid al celor trei metodologii precizate mai sus.

METHONTOLOGY permite construcția de ontologii la nivel de cunoștințe [94]. Presupune identificarea etapelor din procesul de dezvoltare a ontologiei; ciclul de viață al ontologiei; tehnici pentru management, dezvoltare și suport. În plus, conține o listă de activități ce trebuie efectuate în timpul reutilizării sau reproiectării ontologiei, dar nu specifică ghiduri fixe pentru aceste activități.

Metodologia **On-To-Knowledge** [95] propune ca în construcția ontologiei să se aibă în vedere modul în care aceasta va fi utilizată în aplicații cu cunoștințe. Procesul de proiectare propus prin această metodologie constă din: studiu de fezabilitate, identificarea cerințelor ontologiei, finisarea ontologiei pentru a produce o variantă matură orientată spre aplicații, evaluare și mentenanță. Pentru a asigura reutilizarea vocabularelor sau ontologiilor deja existente, în pasul doi al proiectării (identificarea cerințelor ontologiei) este menționat faptul că dezvoltatorii trebuie să aibă în vedere ontologiile ce pot fi reutilizate. Totuși, nu este specificat și un ghid prin care astfel de ontologii pot fi identificate și reutilizate.

Metodologia **DILIGENT** [96][97] este orientată spre un mod de lucru distributiv și colaborativ. Procesul cuprinde cinci activități principale: construcția, adaptarea locală, analiza, revizia și modificarea locală. După dezvoltarea unei ontologii de bază, utilizatorii o pot utiliza și adapta local pentru nevoile proprii. Astfel, rezultă două tipuri de ontologii: una partajată (cea de bază), disponibilă

tuturor, și mai multe ontologii locale. Metodologia nu include detalii cu privire la reutilizarea altor ontologii.

După cum spuneam anterior, o abordare diferită o reprezintă metodologia **NeOn,** bazată pe scenarii. Suportă reutilizarea ontologiilor existente, mod de lucru colaborativ și rețele de ontologii în sisteme distributive [98]. Principalele caracteristici ale acestei metodologii sunt: un set de nouă scenarii pentru construcția de ontologii, cu accent pe reutilizarea ontologiilor; reproiectare, colaborare; glosar de procese și activități întâlnite în procesul de proiectare; ghiduri pentru diferite procese și activități.

Setul de nouă scenarii pentru construcția de ontologii poate fi rezumat în modul următor:

- **Scenariul 1**: **De la specificații la implementare –** Ontologia este dezvoltată de la zero fără a reutiliza resurse.
- **Scenariul 2**: **Reutilizarea si reproiectarea de resurse non-ontologice** – Dezvoltatorii trebuie să identifice aceste resurse, să decidă care dintre acestea pot fi reutilizate, iar apoi trebuie reproiectate pentru a putea fi incluse în ontologie.
- **Scenariul 3**: **Reutilizarea de resurse ontologice –** Dezvoltatorii utilizează module sau ontologii întregi pentru a construi rețele de ontologii.
- **Scenariul 4**: **Reutilizarea și reproiectarea resurselor ontologice –** Ontologiile sunt reutilizate după un proces de reproiectare.
- **Scenariul 5**: **Reutilizarea și fuzionarea resurselor ontologice –** Acest scenariu este valabil când au fost selectate mai multe ontologii din același domeniu, pentru reutilizare.
- **Scenariul 6**: **Reutilizarea, fuzionarea și reproiectarea resurselor ontologice –** Similar cu scenariul anterior, dar ontologiile selectate necesită și reproiectare.
- **Scenariul 7**: **Reutilizarea șabloanelor de proiectare –** Dezvoltatorii accesează baze de date cu șabloane reutilizabile.
- **Scenariul 8: Restructurarea resurselor ontologice –** Dezvoltatorii restructurează resursele prin modularizare, extindere, specializare, pentru a introduce rețele de ontologii.
- **Scenariul 9: Localizarea resurselor ontologice** – Dezvoltatorii adaptează ontologia la alte limbi sau comunități culturale.

O altă abordare în dezvoltarea unei metodologii este acea de a împrumuta concepte din metodologiile de dezvoltare software și a le adapta pentru dezvoltarea de ontologii. Una dintre aceste abordări este **eXtreme Design** care împrumută concepte din AGILE [99]. Suportă reutilizarea șabloanelor de proiectare, având dezvoltată metodologia și uneltele necesare pentru a o utiliza.

DOGMA este o abordare care susține necesitatea proiectării unei ontologii de bază, care să cuprindă conceptele generale de domeniu. Metodologia cuprinde apoi doi pași de specializare: primul introduce concepte pentru anumite categorii de aplicații, iar al doilea introduce concepte specifice unei aplicații particulare [100].

Majoritatea metodologiilor descrise se rezumă la parcurgerea anumitor pași în procesul de proiectare. Acești pași trebuie să respecte anumite ghiduri, specifice fiecărei metodologii în parte. Ei se pot generaliza și rezuma la [101] [102] [103] [104]:

- **Stabilirea specificațiilor ontologiei** – obiectivul acestui pas îl reprezintă identificarea scopului ontologiei. De asemenea, trebuie

stabilit domeniul care urmează a fi modelat, cui se adresează ontologia şi care este nivelul de expresivitate necesar.

- **Conceptualizarea –** în acest pas, specificațiile stabilite la pasul precedent sunt transpuse într-un model conceptual. Se dezvoltă un vocabular de termeni și relațiile dintre concepte.
- **Formalizarea –** conceptele introduse în pasul precedent și relațiile dintre acestea trebuie formalizate, pentru a avea definiții non-ambigue. În acest pas se introduc constrângeri asupra proprietăților și conceptelor.
- **Implementarea –** este procesul de transpunere a ontologiei formalizate într-un limbaj pe care un agent software poate să îl înțeleagă (ex: OWL).
- **Evaluarea –** reprezintă procesul de validare a ontologiei.
- **Mentenanța -** în timpul testării ontologiei, într-un sistem bazat pe cunoștințe, se corectează deficiențele acesteia.

Pentru proiectarea ontologiei am utilizat pașii descriși mai sus. Datorită limitării deduse din clasificare si flexibilităţii ridicate, consider că cea mai bună metodologie de proiectare a unei ontologii o reprezintă **NeON**. Voi utiliza unul din scenariile de reutilizare și / sau reproiectare, în funcție de conceptele care urmează a fi modelate și având în vedere vocabularele sau ontologiile care se potrivesc pentru adaptare în ontologia educațională.

4.2 Model de ontologie educațională

Ontologiile oferă o metodă standardizată de reprezentare a cunoștințelor, care poate fi extinsă la nesfârșit, pe baza unui set definit de relații între componentele de bază ale ontologiei.

Termenul de ontologie provine din filozofia greacă și reprezintă o ramură a metafizicii preocupată cu definirea (în mod general) naturii lucrurilor care aparțin unui univers de discurs. Universul de discurs reprezintă setul de entități reprezentate formal. [105]

Ontologiile joacă un rol fundamental în dezvoltarea de agenți de comunicare, precum și în contextul Web-ul Semantic, în curs de dezvoltare. Ideea care stă la baza Web-ului Semantic este, după cum sugerează și numele, adăugarea unui nivel de sens în Web, astfel încât datele să fie mai ușor manipulate de către programele de calculator și să devină mai uşor accesibile pentru om. [88]

Domeniul educațional poate beneficia foarte mult de posibilitățile oferite de Web-ul Semantic (conform studiilor din capitolele anterioare). Sistemele semantice pot afecta pozitiv accesul persoanelor la materialele educaționale. Ele pot oferi un set bogat de servicii care să permită personalizarea modului în care conținutul este disponibil utilizatorilor, prin furnizarea unei baze de date bine structurate, care permite o mai bună manipulare a cunoștințelor de către mașini.

Ontologiile și Web-ul Semantic oferă o nouă perspectivă asupra sistemelor inteligente de e-Learning, prin reprezentarea adecvată a conceptelor utilizate. Astfel de sisteme de e-Learning oferă secvențierea curriculum-ului, analiza soluțiilor studentului [106] și trebuie să fie construite cu o serie de funcționalități, printre care următoarele ar putea fi incluse:

- Construirea unui model al planului de învățământ al studentului, prin colectarea de date despre student și interesele sale;
- Furnizarea materialelor de curs în succesiunea corectă, cu adnotările corespunzătoare, conform datelor colectate și a modelului studentului;

- Urmărirea intereselor studenților, pentru a stabili curriculum;
- Monitorizarea evoluției studenților în cadrul grupurilor de studiu și identificarea celor cu o rată diferită de progres.

În continuare, voi prezenta modelul de ontologie pe care l-am propus pentru utilizare în sisteme electronice educaționale pentru învățământul superior. Obiectivul principal pe care mi-am propus să îl ating a fost acela de a crea o ontologie de domeniu capabilă să reprezinte principalele concepte din învățământul superior și care să ofere suport sistemelor specializate de e-Learning.

Dezvoltarea de ontologii pentru învățământul superior se află încă într-un stadiu incipient. În domeniul e-Learning majoritatea ontologiilor sunt create pentru a descrie elemente de învățare și nu pentru a asista sisteme specializate de suport.

Un exemplu de utilizare a unor ontologii într-un sistem de e-Learning este furnizat în cadrul proiectului "*Scholarly Ontologies*", al institutului *Knowlege Media,* din cadrul *Open University*. Scopul acestui proiect este acela de a dezvolta o abordare computațională pentru a sprijini extragerea sensului academic, prin interpretare și argumentare, permițând cercetătorilor să facă afirmații: să descrie și să dezbată punctul lor de vedere asupra principalelor contribuții dintr-un document și a relațiilor cu literatura de specialitate. [107]

O altă ontologie prezentată de Verdejo, Read și Mizoguchi oferă o reprezentare conceptuală a nivelului de cunoștințe pentru a descrie sistemele colaborative de învățare. Această ontologie conține cunoștințe care nu sunt reprezentate explicit în alte ontologii destinate învățământului colaborativ, cunoștințe cu privire la studiul și analiza procesului de învățare. [108]

Sistemul de învățare ISAR, un proiect Leonardo da Vinci, finanțat de Uniunea Europeană, oferă un exemplu de sistem educațional cu ontologii, folosind concepte SCORM. Sistemul cuprinde un set de ontologii care descriu tipul produselor care rezultă din procesul de fabricație, ontologii de administrare și ontologii de învățare, care vizează structurarea materialelor educaționale. [109]

Alte ontologii au fost create pentru a descrie conținutul educațional, interacțiunile dintre entitățile implicate în procesul de e-Learning sau scenariile educaționale ale mediilor colaborative.

Ontologiile existente, utilizate în învăţământul superior, nu au fost dezvoltate având în vedere necesitățile sistemelor educaționale. Am propus un model de ontologie pentru structurarea unui sistem de e-Learning destinat învățământului superior, oferind posibilitatea de a descrie caracteristicile sistemului, prin componentele sale. Primul pas a fost să identific ce date sunt necesare pentru modelare, adică ce date ar fi relevante pentru partajare cu aplicații externe.

4.2.1 Stabilirea specificațiilor ontologiei

Primul pas în proiectarea ontologiei educaționale a fost acela de a stabili domeniul ontologiei. Această ontologie va fi folosită pentru a reprezenta entități din învățământul superior, cum ar fi universitate, facultate, student, profesor, curs sau seminar. Toate aceste concepte au fost definite având în vedere apartenența lor la un sistem de e-Learning. Am considerat faptul că domeniul principal al acestei ontologii educaționale îl constituie reprezentarea conceptelor existente într-un sistem web de e-Learning.

Ontologiile reprezintă un produs extraordinar al Web-ului Semantic. Semantica asigură utilizatorul uman că performanțele mașinii sunt ridicate la un nivel superior, furnizând uneltele necesare nu doar pentru prelucrarea informațiilor

disponibile, dar şi pentru înțelegerea și dezvoltarea de noi șabloane de deducere a informației, în scopul de a extinde capacitățile sale de extragere a cunoștințelor.

Având în vedere conceptele prezentate anterior, construcţia unei ontologii educaționale se rezumă la efectuarea următorilor paşi: realizarea unei cercetări cu privire la sistemul de e-Learning; extragerea elementelor de bază care compun structura pe care sistemul o poate dezvolta.

Conform studiilor realizate de colegul meu Andrei Ternauciuc în cadrul tezei sale de doctorat [52], Moodle este cea mai utilizată platformă educațională în cadrul universităților de top din România. Principalul său avantaj este disponibilitatea în sursă deschisă a acestui pachet software, astfel fiind reduse semnificativ costurile. Comparativ cu alte platforme educaționale importante, disponibile pe piaţă (Blackboard, Sakai), Moodle oferă aceleași facilități (urcare și partajare fișiere, unelte de comunicare, obiecte educaționale, etc.), prin pachete de bază sau prin extensii dezvoltate de comunitatea Moodle.

Conform celor de mai sus, consider că un studiu al platformei Moodle va reliefa caracteristicile principale ale oricărui LMS, fie că este comercial sau în sursă deschisă.

Astfel, pentru a putea identifica elementele care pot fi modelate semantic, am ales să studiez îndeaproape platforma educațională Moodle, platformă care este utilizată şi în cadrul Universității ”Politehnica” din Timișoara în cadrul Campusului Virtual (cv.upt.ro). Studiul a fost realizat pe ultima variantă a Campusului Universitar Virtual, pentru anul universitar 2011 - 2012.

Moodle își păstrează datele, ca majoritatea aplicaților web moderne, în baze de date relaționale. În figura 14 se observă schema bazei de date Moodle cu pachetele de bază. Se observă complexitatea ridicată a acestei scheme, ea devenind din ce în ce mai complicată cu fiecare extensie instalată. În cazul particular al instanței din universitatea noastră, baza de date conține 349 de structuri. În decursul unui an de studiu, având în vedere că momentan sunt găzduite preponderent materialele aferente cursurilor de la ciclul de master și de la învățământ la distanță (aproximativ 750 de cursuri și 4.600 de utilizatori), au fost generate 2,5 milioane de înregistrări în aproximativ 7 Gb de informație. O parte din această informație nu prezintă interes pentru publicarea ei într-un format partajabil, fiind destinată exclusiv administrării platformei.

Având în vedere cantitatea mare de informație disponibilă în platformă, a trebuit să identific ce informație ar fi utilă pentru republicare. Pentru a face acest lucru, am identificat anumite scenarii în care accesul la informația din platforma educațională ar fi folositoare în procesul educațional:

- **Informațiile de utilizator –** Ar fi utilă partajarea lor dacă se dorește implementarea unui mecanism de transfer a datelor utilizatorilor, pentru a permite autentificarea, fără a fi necesară crearea unui cont nou sau reintroducerea utilizatorului și parolei [110]. De asemenea, ar fi utilă partajarea datelor de contact ale utilizatorilor pentru a permite construcția de agenți software care să definească și să îmbunătățească liste de contacte automat [73].
- **Materiale educaționale –** Permițând accesul la materiale educaționale (cursuri, documente, obiecte educaționale) într-un format semantic web, este facilitată construcția de agenți software care să îmbunătățească aceste resurse prin intermediul unor procese, precum: culegerea de referințe bibliografice printr-un proces automat, reutilizarea obiectelor educaționale, integrarea de resurse video [111][112], generarea de rapoarte, statistici, etc.

Figura 14. Schema bazei de date Moodle [113]

- **Activități –** Partajarea datelor referitoare la teme, unelte de comunicații (blog, wiki, forum, im), proiecte, permite colectarea informațiilor din mai multe aplicații utilizate în procesul didactic. Evaluând aceste informații se poate observa calitatea răspunsurilor, nivelul de implicare în anumite direcții de studii, obținându-se în acest mod date relevante pentru recomandarea de materii opționale studenților. Discuțiile pot fi "strânse" din mai multe surse; astfel, la identificarea răspunsului la o întrebare pot fi extrase informații din mai multe aplicații. Temele alocate pot fi disponibile în aplicații externe, în care studenții vor îndeplini cerințele activității respective. Acest ultim aspect îl voi prezenta pe larg în capitolul următor.
- **Evaluări –** Procesul de generare de teste, de evaluare sau auto-evaluare, ar putea fi îmbunătățit prin generarea dinamică de teste dintr-o bancă de întrebări. Datele păstrate în aceasta pot fi colectate din mai multe surse. De exemplu, Moodle are mai multe module de gestionare a testelor, cele mai importante fiind Feedback și Questionnaire. O descriere pe larg a acestui scenariu și studiul cu privire la implementarea lui practică va fi prezentat în capitolul 5.

Datorită cantității mari de date, dar şi datorită faptului că majoritatea aplicațiilor care utilizează date semantice vor avea nevoie doar de o parte din ontologie, este indicată împărțirea în sub-ontologii. Astfel, se limitează accesul doar la datele necesare, permițând în același timp o dezvoltare ulterioară mai ușoară datorită modularității.

În funcție de domeniul diferitelor date și de modul de structurare a acestora în cadrul platformei Moodle, am identificat următoarele subseturi care vor deveni sub-ontologii: *persoane, cursuri, activități și evaluare*. Fiecare dintre aceste ontologii conține elemente care sunt în strânsă legătură.

La identificarea limitărilor fiecărei sub-ontologii, am ținut cont de modelul din viața reală. În tabelul următor sunt listate principalele limitări identificate în fiecare sub-ontologie în parte, precizând și gradul de expresivitate necesar în limbajul de modelare pentru implementarea acelui deziderat.

Tabel 5. Limitări identificate pentru proiectarea ontologiei educaționale

Sub-ontologie	Expresivitate	Limitare
Persoane	Scăzută	+ Utilizatorii sunt împărțiți în profesori, studenți si administratori, pe baza rolurilor pe care le au în sistem
	Scăzută	+ Pot să dețină mai multe informații de contact
	Scăzută	+ Un utilizator poate să aibă un alt rol în fiecare curs, deci drepturile utilizatorului într-un curs se pot schimba în funcție de context
	Scăzută	+ Pot exista, între utilizatori, relații student - student, tutore - student sau tutore - tutore

Cursuri	Scăzută	+ Structurate sub forma unei curriculum al unui an de studiu
	Scăzută/Medie	+ Pot avea unul sau mai mulți tutori și mai mulți studenți
	Scăzută	+ Un curs poate fi structurat în curs, laborator, seminar, proiect și alte activități
	Medie/Ridicată	+ Participarea la un anumit curs poate fi condiționată
	Scăzută	+ Descriere curs; să conțină și grad de dificultate (în special opționale)
Activități	Scăzută	+ Pot fi teme, proiecte, unelte de comunicare, unelte conexe
	Scăzută	+ Unele activități necesită evaluare din partea tutorelui
	Scăzută	+ Activitățile pot fi de curs sau platformă
	Scăzută	+ Pot participa toți utilizatorii înscriși în curs (studenți și tutori) sau pot fi deschise (în special cazul uneltelor de comunicare)
Evaluare	Scăzută	+ Conține mai multe tipuri de elemente de test
	Scăzută	+ Se pot genera teste pentru evaluare de către tutore sau teste de auto-evaluare de către studenți
	Scăzută	+ Fiecare element de test are arondată o dificultate (utilizat în generarea testelor, pentru a păstra suma dificultăților într-un anumit interval)
	Scăzută	+ Întrebările pot fi relaționate între ele, astfel încât la generarea testelor să nu se extragă două întrebări care abordează același subiect

Din tabelul de mai sus se observă că gradul de expresivitate al limbajului de modelare, necesar pentru implementarea limitărilor, este în marea majoritate a cazurilor scăzut, dar folosind un limbaj mai expresiv se vor putea introduce constrângeri suplimentare care vor putea deduce date suplimentare printr-un proces de inferențiere. După cum am precizat în capitolul 2 pentru modelare se poate utiliza RDFS, un limbaj mai puțin expresiv, sau OWL, limbaj dedicat definirii ontologiilor.

În momentul definirii de ontologii se poate utiliza în același timp RDFS și OWL, folosind un subset oricât de mic din conceptele introduse de ele. Limbajele sunt foarte flexibile și se pot utiliza într-un mod rezonabil și pragmatic pentru fiecare caz în parte.

Momentan soluțiile software nu au implementat toate conceptele din OWL, iar ca și motor de inferențe este disponibilă gratuit doar o versiune OWL-Lite, cu o mică parte din specificațiile OWL implementate.

În proiectarea ontologiei voi utiliza pe cât posibil concepte care se pot utiliza în mediile de dezvoltare existente, iar odată cu dezvoltările viitoare ale acestora, ontologia se poate extinde păstrând compatibilitatea între diferite sisteme.

Pe baza specificațiilor identificate, în pasul următor voi defini conceptele din fiecare din cele patru sub-ontologii, voi identifica proprietățile care le leagă, căutând în același timp vocabulare/ontologii care se pot reutiliza conform unui scenariu din metodologia **NeOn**.

4.2.2 Conceptualizarea

În dezvoltarea ontologiei, procesul de conceptualizare presupune construirea unei liste cuprinzătoare de termeni. Pentru a obține această listă de termeni, este important să nu luăm în considerare suprapunerea dintre conceptele pe care le reprezintă, relațiile dintre termeni sau orice proprietăți pe care conceptele le pot avea, sau dacă conceptele sunt clase sau proprietăți. [114]

O ontologie conține o listă finită de termeni și relațiile stabilite între ele. Termenii sunt de fapt o serie de concepte (obiecte si clase), care aparțin unui anumit domeniu. Relațiile dintre obiecte și clase pot fi modelate ca o ierarhie. Am folosit un proces combinat de dezvoltare (evaluarea structurilor de date relaționale ale platformei Moodle, corelate cu structurile de date din alte aplicații educaționale, identificarea unor concepte comune), pentru a structura ierarhic clasele ontologiei.

Persoane

Informațiile unei persoane, cum ar fi nume, date de contact, ultima autentificare, etc., nu sunt diferit organizate pentru diferitele tipuri de utilizatori. Pentru a introduce conceptul de utilizator putem folosi același termen. Am introdus pentru acest lucru conceptul de ***Persoană,*** din care fac parte toți utilizatorii platformei aplicației.

Pentru a descrie o persoană este nevoie de utilizarea unor concepte care să introducă: *nume, prenume, data nașterii, poză*. De asemenea, este necesar să fie modelate informațiile de contact existente în baza de date a platformei Moodle, astfel că trebuie introduse concepte cu privire la: *email, telefon, adresă, oraș, adresă skype, adresă aim, pagină web, adresă msn*. Dacă vor fi introduse alte unelte de comunicare externe, va fi necesară introducerea în ontologie a conceptelor care descriu aceste unelte. Aceste concepte au legătură cu conceptul *Persoană*.

La introducerea unui utilizator nou, acesta poate avea în sistem rolul de *administrator, tutore*, *student* sau *vizitator*. Aceste concepte sunt subdiviziuni ale conceptului *Persoană*. Dacă un utilizator este descris prin intermediul conceptului student, el este descris automat și de conceptul Persoană și de toate conceptele care au legătură cu acesta. După cum am stabilit în pasul precedent, în specificații, un utilizator poate avea rol diferit în cadrul unui curs, față de rolul general din platformă.

Informații specifice conceptelor de *tutore* și *student* sunt afilierea cu universitatea, respectiv facultatea/departamentul, la care îndeplinesc aceste roluri. Conceptul de *universitate* nu va fi util în modelarea datelor din platforma de e-

Learning; el va fi necesar doar în momentul partajării lor cu alte universități sau organizații externe.

Pentru a descrie limitările de acces dintr-un sistem de e-Learning, voi defini posibilitate de a atribui drepturi unui anumit utilizator. La definirea drepturilor am utilizat principiul minimului de privilegii necesare. Prin urmare, trebuie să fie definite mai multe profiluri de utilizatori, care să asigure faptul că un anumit utilizator nu are mai multe drepturi decât este necesar.

Am definit opt operații la care un utilizator poate avea acces în funcție de profilul de utilizator atribuit lui: crearea de conținut, crearea de agenți, accesarea agenților, citirea conținutului public, citirea conținutului nepublic, editare de conținut, ștergerea de conținut, reproducerea sau copierea de conținut.

Am definit trei tipuri de utilizatori:

- *Editor* – are acces la toate operațiile;
- *Deponent* – poate doar crea, edita și citi conținut public sau privat;
- *Cititor* – poate doar citi conținut public și poate accesa agenți web. Aceste activități pot fi reprezentate de navigarea prin documente, realizarea temelor sau participarea în procesul de evaluare;
- *Fără acces* – restricționează accesul utilizatorului doar la vizualizarea conținutului public.

Aceste profiluri sunt arondate implicit, în funcție de tipul utilizatorilor, după cum urmează: administrator – editor, tutore – editor, student – cititor, vizitator – fără acces. Pentru a putea acorda drepturi unui utilizator voi introduce un concept intermediar, numit *profil*, care va primi drepturi specifice; drepturile pot fi alocate direct conceptelor de administrator, tutore, student sau oricărui alt tip de utilizator care va fi introdus ulterior. A doua variantă este mai ușor de implementat și simplifică construcția ulterioară a interogărilor, deoarece se reduce numărul de noduri intermediare.

Un utilizator se poate găsi într-una din relațiile student – student, student – tutor, tutor – tutor, în funcție de rolul jucat de aceștia în interiorul unui curs. Activând în mai multe contexte (cursuri), doi utilizatori se pot găsi în mai multe relații. Pentru a modela această legătură trebuie să introduc conceptele de *are tutor pe*, *are coleg tutore pe*, *are coleg student pe*. Acestea vor fi concepte care vor fi legate de conceptul *persoană* și *curs*.

Cursuri

Pentru a organiza materiale de curs trebuie introdus în ontologie conceptul de *curs.* Fiecare instanță a acestui concept va avea definite anumite proprietăți care să definească informații despre cursul respectiv: numele cursului, descriere, structura cursului, precizându-se numărul de săptămâni de predare și tipul activităților care se desfășoară (laborator, seminar, proiect). Astfel, trebuie introduse conceptele: *nume curs, descriere, părți curs.* De asemenea, pentru a preciza domeniul cursului se pot adăuga etichete. Introducerea conceptului *domeniu* ar fi util pentru a putea relaționa cursurile din mai multe platforme și a partaja resurse între ele.

Planurile de învățământ cuprind un set de activități pe care un student înscris într-un anumit an de studiu trebuie să le urmeze. Se formează liste de cursuri pe care un student înscris într-un anumit ciclu de studii trebuie să le urmeze.

Conceptul de *Curriculum* îndeplinește acest obiectiv. Fiecare individ care aparţine conceptului *curs* reprezintă o instanţă a conceptului *curriculum*. Această legătură între cursuri și curriculum poate fi de două feluri: cursuri obligatorii, pe care studenții sunt obligați să le urmeze pentru a termina studiile, și cursuri

opționale, unde pot alege dintr-o listă. Astfel, pentru a defini legătura dintre cursuri și curriculum avem nevoie să introducem conceptul de *curs obligatoriu* și *curs opțional*. Un curs poate fi legat de mai multe instanțe ale conceptului *curriculum,* de unele ca și curs opțional, iar de altele obligatoriu. De asemenea, un curriculum se poate organiza în secțiuni, de exemplu anii de studiu.

Fiecare curs poate avea stabiliți mai mulți participanți, cu rol de tutore sau student. Astfel, introducem conceptul de legătură între persoană și curs: *tutore la* și *student la,* care stabilesc în acest mod și rolul pe care acea persoană îl are în interiorul acelui context.

Pentru a modela condiționarea participării la un anumit curs, am introdus conceptul de *curs legat,* care nu permite înscrierea studenților care nu au urmat sau nu au promovat cursul sau cursurile anterioare necesare.

Este necesară astfel modelarea informației cu privire la promovarea cursului de către studenți. Se poate păstra fie doar starea de promovat/nepromovat sau se poate extinde acest concept pentru a cuprinde inclusiv nota obținută la fiecare din părțile componente ale cursului și media lor.

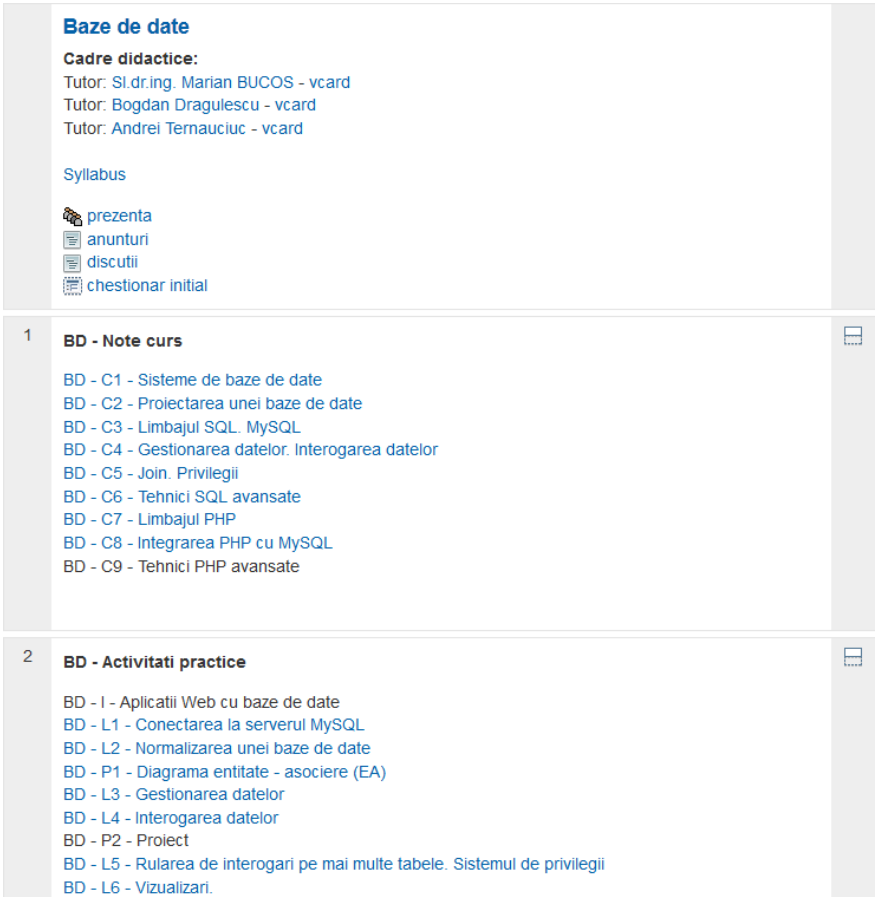


Figura 15. Structură curs Moodle (fragment exemplu curs cv.upt.ro)

O parte importantă din organizarea unui curs o constituie materialele de curs. Acestea pot fi în diferite formate: simple documente (pdf, word, html), obiecte educaționale (SCORM, IMS), resursă externă. În cadrul cv.upt.ro majoritatea cursurilor conțin materialele de curs sub formă de documente. Structura tipică a unui curs poate fi observată în figura 15. Documentele pot fi arondate unui anumit curs sau pot fi integrate într-o anumită parte de curs. Fiecare din aceste resurse de curs poate deține următoarele informații: nume, domeniu, autor, etc.

Activități

Ca prim pas trebuie introdus conceptul de *activități*, care să conțină conceptele de *teme*, *unelte de comunicare, unelte conexe.* Fiecare din instanțele acestor concepte poate fi descrisă folosind: nume, durata de desfășurare (pe tot parcursul cursului sau un interval prestabilit), cerințele activității, autor.

Unele activități necesită evaluare din partea tutorelui, pentru notare sau îndrumare. În acest context am introdus conceptele *necesar evaluare* și *necesar notare.*

Activitățile de comunicare sau uneltele conexe pot fi activități de platformă sau de curs, în timp ce temele pot fi doar activităţi de curs. Pentru a modela acest lucru este util conceptul *legat de curs.*

Un utilizator poate participa la una sau mai multe activități. În platforma Moodle, temele constau în urcarea unuia sau a mai multor documente, editarea unui text sau realizarea unei activităţi externe, după cum se poate vedea din figura 16. Trebuie păstrate: data predării temei, documentul sau textul urcat, respectiv nota (dacă este cazul).

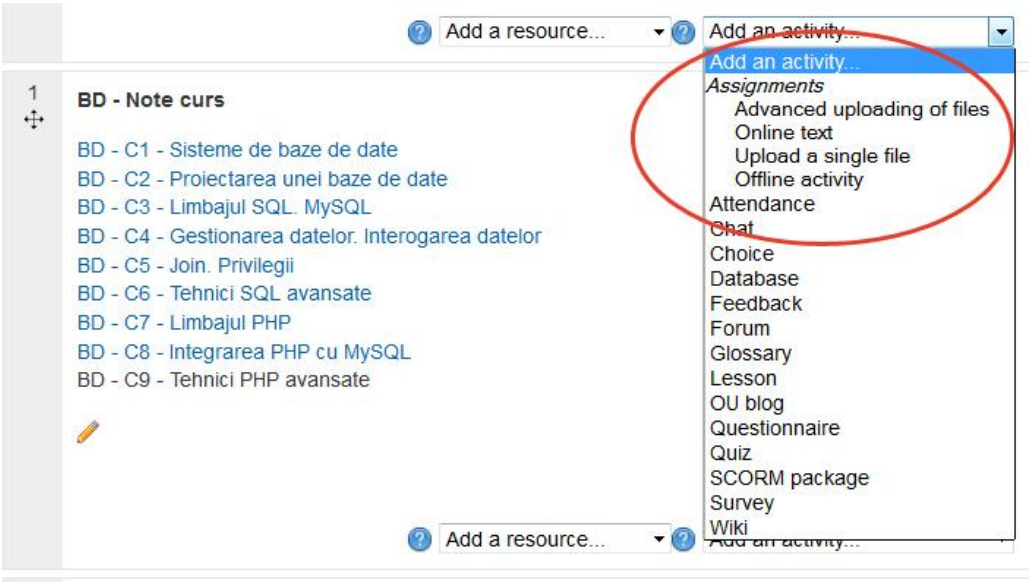


Figura 16. Structura activităților în Moodle (temele sunt încercuite)

În cazul uneltelor de comunicare trebuie introduse subdiviziunile disponibile: forum, wiki, blog, chat. Aceste concepte sunt deja modelate extins pentru Web-ul Semantic, trebuie doar aplicate și în interiorul acestei ontologii. În plus, este

necesar să asigur legătura spre curs (concept prezentat mai sus) şi legătura utilizatorului cu mesajele introduse de către acesta (prin intermediul conceptului *scris de*).

Evaluare

Fiecare test de evaluare este compus din *întrebări*, acesta fiind și primul concept introdus. Întrebările fiind de mai multe tipuri, există subdiviziuni ale conceptului introdus: *răspuns multiplu, răspuns scurt, adevărat/fals, gradare.*

Pentru a organiza elementele de testare, două concepte sunt folosite: *secțiune de curs* și *curs.* Conceptul *secțiune de curs* asociază o secțiune la un anumit curs. În acest fel, putem limita selecția întrebărilor la un anumit subiect și / sau la un anumit curs.

Pentru a asigura că întrebările extrase din baza de date nu abordează același subiect, trebuie să fie declarate toate întrebările relaționate. Acest lucru se face prin utilizarea proprietății *întrebări relaționate,* care leagă toate întrebările relaționate de același nod gol.

Pentru întrebările cu selecţie multiplă și pentru întrebările adevărat-fals se folosesc conceptele *răspuns greșit* și *răspuns corect* pentru a defini răspunsurile. Pentru a limita numărul maxim de cuvinte acceptate pentru răspunsul la întrebările scurte, am introdus conceptul *număr de cuvinte*.

Atunci când o întrebare este introdusă pentru prima dată în baza de întrebări, este atribuit un nivel de dificultate de către profesor, folosind conceptul *grad de dificultate*. Această informație este utilizată în generarea testelor pentru a păstra dificultatea de ansamblu a testului într-un anumit interval. Gradul de dificultate al unei întrebări poate fi schimbat prin evaluarea rezultatelor obținute de către studenți, la acea întrebare, în decursul timpului.

4.2.3 Formalizarea

Conceptele introduse în pasul precedent și relațiile dintre acestea trebuie formalizate pentru a avea definiții non-ambigue. În acest pas se introduc constrângeri asupra proprietăților și conceptelor.

Ontologiile sunt în general construite respectând o anumită structură, care conține mai multe componente. Aceste componente sunt: clase, instanțe, atribute, relații, restricții, reguli și axiome.

Voi căuta vocabulare/ontologii care să descrie majoritatea conceptelor dorite spre modelare, astfel încât să pot să le refolosesc (reproiectându-le sau extinzându-le dacă este necesar) sau, în caz contrar, să formalizez conceptele descrise în pasul precedent.

În formalizarea conceptelor voi folosi limba engleză, pentru a asigura caracterul internaţional al ontologiei.

Persoană

Cel mai cunoscut vocabular pentru modelarea datelor care descriu persoane, relațiile dintre acestea și datele lor de contact este FOAF (Friend of a Friend). Este unul din cele mai mari proiecte ale Web-ului Semantic, fiind un standard acceptat la scară largă pentru a reprezenta rețele sociale și multe din aplicațiile web sociale îl folosesc pentru a-și descrie utilizatorii [115]. Popularitatea standardului îl recomandă pentru o posibilă utilizare și în modelarea utilizatorilor din cadrul unei platforme educaționale. Astfel, modelarea datele utilizatorilor, folosind clase și proprietăți definite în FOAF, permite dezvoltarea de agenți software care să

îmbogățească datele despre persoane cu informații disponibile în platformele sociale.

Pentru a putea identifica clasele și proprietățile din FOAF care se potrivesc pentru conceptele introduse, am analizat structura vocabularului FOAF descris în documentul de standardizare [116]. În figura 17 se pot observa clasele și proprietățile definite în vocabular. Am eliminat etichetele și elementele definite în limbajele de modelare, pentru a putea avea o privire de ansamblu.

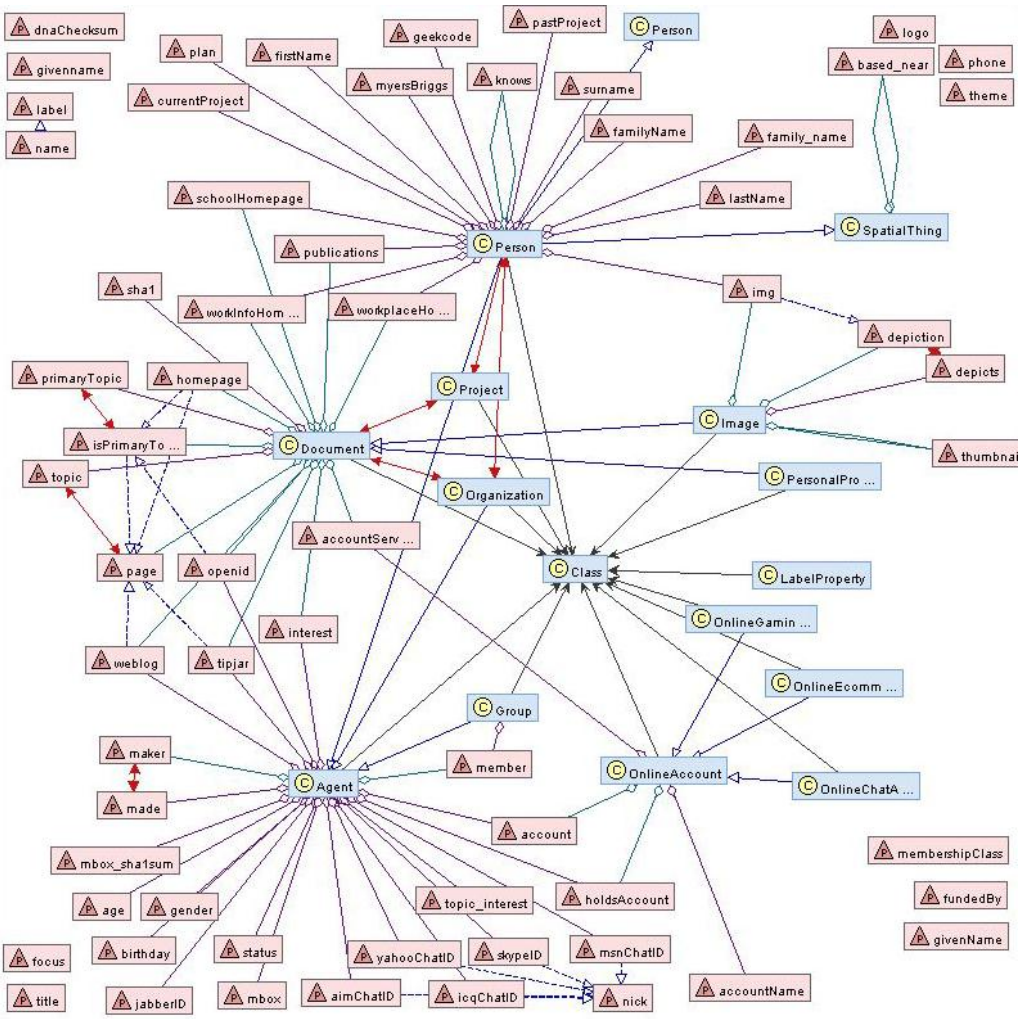


Figura 17. Structura vocabularului FOAF (fragment)

Se observă clasa **foaf:Person**, clasă utilizată în vocabular pentru a reprezenta persoane. Este o subclasă a **foaf:Agent**, introdusă deoarece clasa *foaf:Person* ar fi fost prea specifică în anumite situații. De exemplu, anumite conturi de aplicații (jabberID, IM) pot fi asociate unei persoane sau unei organizații. Datorită acestui fapt, orice instanță a clasei *foaf:Person* va fi, printr-un proces de

inferență, o instanță a clasei *foaf:Agents*, având acces în acest fel la proprietățile specifice agenților.

Conceptele *nume, prenume, data nașterii, poză* pot fi formalizate utilizând proprietățile *foaf:lastname, foaf:firstname, foaf:birthday, foaf:img*. De asemenea, este necesar să fie formalizate informațiile de contact existente în baza de date a platformei Moodle. Acestea pot fi modelate folosind proprietățile: *foaf:mbox* (adresă email*), foaf:phone (telefon, este în curs de standardizare), foaf:skypeID (*adresă skype), *foaf:aimChatID* (adresă aim), *foaf:homepage* (pagină web), *foaf:msnChatID (adresă msn).* Pentru *adresă, oraș* și *țară* voi utiliza vocabularul vCard, care permite modelarea microformatului hCard în format RDF. Astfel, voi utiliza proprietățile *vcard:street-address, vcard:locality, vcard:postal-code* și *vcard:country-name*.

Pentru a formaliza rolurile pe care utilizatorii le au în cadrul platformei educaționale, voi introduce clasele **edu:Admin**, **edu:Tutor**, **edu:Student** și **edu:Guest**. Acestea vor fi subclase ale clasei *foaf:Person.* Instanțele claselor *edu:Tutor* și *edu:Student* vor putea fi legate de o instanță a clasei **edu:University**, prin proprietatea *edu:hasUniversity,* și de o instanță a clasei **edu:Faculty**, prin proprietatea *edu:hasFaculty.* Clasa *edu:Faculty* este o subclasă a clasei *edu:University*.

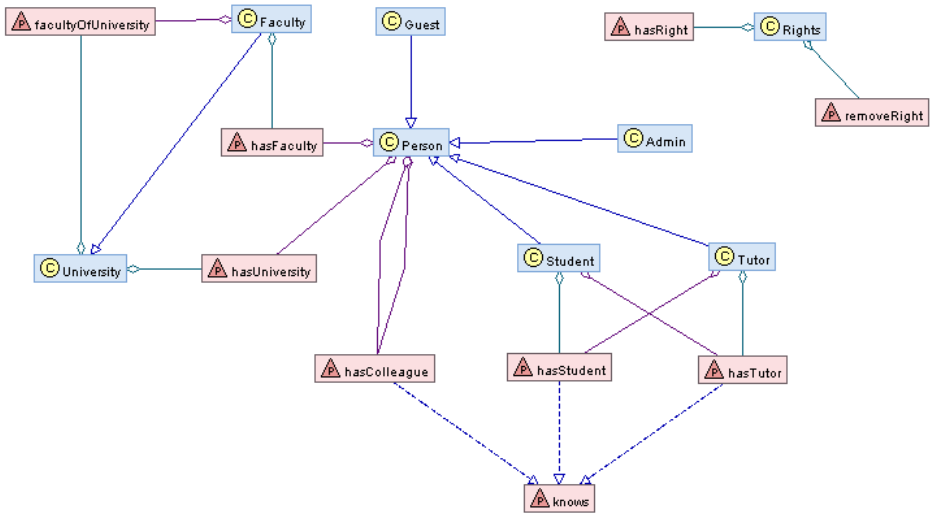


Figura 18. Elementele introduse pentru sub-ontologia Person (fragment)

Pentru atribuirea privilegiilor disponibile în sistem am introdus clasa **edu:Rights**, permițând în acest mod introducerea unui număr flexibil de drepturi (acestea sunt specifice unui anumit sistem). Pentru a lega o instanță a acestei clase (adică un privilegiu) de o clasă sau o instanță, se folosește proprietatea *edu:hasRight*. În cazul în care proprietatea se folosește având ca subiect clase (Admin, Tutor, etc.), orice instanță a acestor clase va deține și ea drepturile acordate. Pentru a acorda drepturi în cadrul unui context, (de exemplu, în interiorul unui curs) trebuie să avem în vedere faptul că drepturile vor fi valabile doar dacă există legătura între student și curs, respectiv tutore și curs. În caz contrar, vor fi valabile drepturile specifice clasei tipului de utilizator. În anumite situații este

necesară reducerea privilegiilor pentru anumite contexte. Din acest motiv, am introdus proprietatea *edu:removeRight*.

Un concept foarte util în vocabularul FOAF este proprietatea foaf:knows, care definește faptul că o persoană cunoaște o altă persoană. În cazul mediului educațional această legătură poate să fie descrisă mai specific, de aceea am introdus sub-proprietăți ale foaf:knows, care modelează legătura dintre utilizatori: *edu:hasTutor* (legătura student - tutor), *edu:hasColleague* (pentru student – student sau tutore - tutore) și proprietatea inversă *edu:hasStudent* (pentru tutore - student).

Conceptele formalizate suplimentar, față de cele descrise de vocabularele foaf și vcard, sunt reprezentate grafic în figura 18. În plus, față de clasele și proprietățile descrise mai sus, se pot utiliza și restul de elemente introduse de vocabularul FOAF sau se pot extinde cu concepte noi. Eu am ales un număr cât mai mic de concepte care să satisfacă nevoile de modelare a datelor. Dacă în timpul dezvoltării de aplicații care utilizează date semantice se observă necesitatea de introducere de noi concepte, se repetă pașii de proiectare.

Cursuri

Formalizarea conceptului *Curs* am făcut-o prin introducerea clasei **edu:Course.** Instanțele acestei clase pot avea atașate informații suplimentare, folosind una din proprietățile: *edu:courseName* (pentru introducerea numelui cursului), *edu:courseDescription* (informații despre curs, cum ar fi un scurt rezumat), *edu:domain* (cuvine cheie care să descrie cursul).

În fiecare curs se pot desfășura diferite activități introduse prin proprietatea *edu:hasSectionOfCourse.* Pentru fiecare tip de activitate trebuie create o sub-proprietate a *edu:hasSectionOfCourse*; astfel am introdus *edu:hasLecture* (unde tutorele expune materialul studenților), *edu:hasSeminar* (întâlnire de seminar), *edu:hasLaboratory* (întâlnire de laborator), *edu:hasProject* (întâlnire pe proiect). Obiectele din triplete definite folosind aceste proprietăți vor fi instanțe ale claselor **edu:Lecture, edu:Seminar, edu:Laboratory, edu:Project**. Aceste patru clase sunt subclase ale clasei **edu:SectionOfCourse.**

Pentru a organiza cursurile am introdus clasa **edu:Curriculum.** Pentru a adăuga un curs la un anumit curriculum se poate folosi proprietatea *edu:mandatoryCourse,* pentru cursurile obligatorii, și *edu:optionalCourse*, pentru cele opționale. Un curriculum se poate organiza în secțiuni, pentru a putea organiza cursurile pe semestre sau ani de studiu, motiv pentru care am introdus clasa **edu:SectionOfCurriculum.** Pentru a adăuga un curs la o secțiune de curriculum se utilizează proprietatea *edu:belongsToSectionOfCurriculum.* Este necesară modelarea legăturii cu facultatea pentru care este format curriculumul. Această legătură este realizată cu ajutorul proprietății *edu:curriculumFor.*

Utilizatorii care participă la un curs pot fi specificați prin proprietățile *edu:studentOf,* pentru studenți, și *edu:tutorOf,* pentru tutori. Subiectul tripletelor care utilizează aceste predicate trebuie să fie instanțe ale clasei persoane, deoarece există situații când rolul de sistem se schimbă în cadrul unui curs (de exemplu, un tutore de sistem poate fi student în cadrul unui curs de formare de formatori).

Pentru a putea condiționa înscrierea într-un curs de urmarea și/sau promovarea altui curs, am introdus proprietatea *edu:conditionBy.* Astfel, este necesară și stabilirea promovabilității unui student. Acest pas se face folosind proprietățile *edu:hasGrade* și *edu:onCourse*, legate printr-un nod gol împreună cu proprietatea *edu:studentOf*. Folosind proprietatea *edu:onSectionOfCourse*, în loc de *edu:onCourse*, se pot păstra notele individuale obținute la secțiuni diferite dintr-un

curs. Pentru a permite calcularea notelor, este necesară introducerea unor concepte și reguli de inferență proprii, ridicând mult nivelul de complexitate al ontologiei. Momentan nu consider necesară această facilitate; dacă va fi necesară, ontologia poate fi extinsă sau se poate crea o sub-ontologie pentru a putea permite această facilitate.

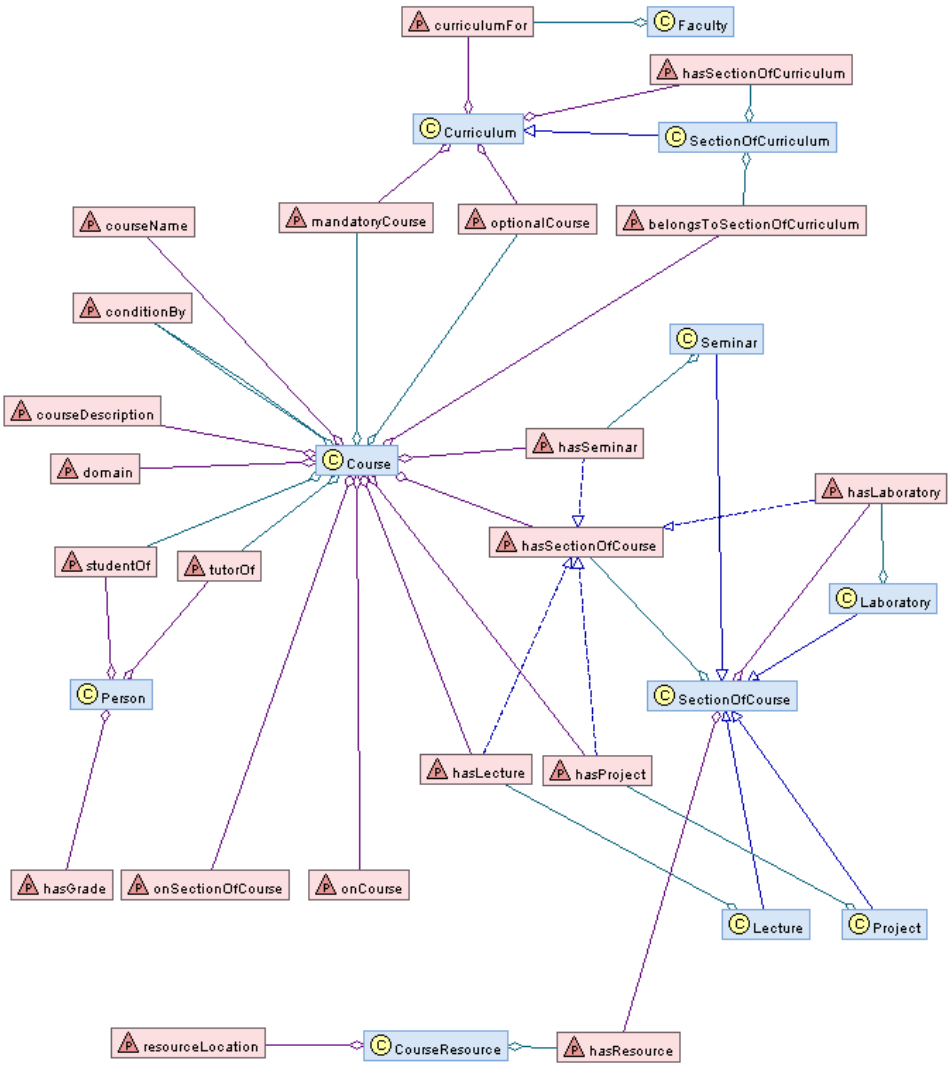


Figura 19. Elementele introduse pentru sub-ontologia Course (fragment)

Materialele de curs le-am considerat ca resurse de curs și instanțe ale clasei **edu:CourseResource.** Acestea pot fi documente sau pachete de obiecte educaționale (este necesară păstrarea legăturii către acestea și existența de informații despre ele). Astfel, am introdus proprietățile *edu:resourceLocation* (locația url a obiectului educațional), *edu:hasResource* (asigură legătura între o secțiune de curs și un document). Pentru a descrie suplimentar documentele (pdf,

doc, html) se poate utiliza vocabularul Dublin Core [117]. Pentru pachete de obiecte educaționale (Scorm, IEEE Lom, IMS) se poate extrage informația structurată conform fiecărui standard în parte și converti la formatul RDF. Un exemplu de cum se poate realiza acest pas este utilizarea limbajului WSML [118]. După cum am mai precizat în acest capitolul, în cazul cv.upt.ro se utilizează cu precădere documente sau pagini HTML, astfel că nu consider momentan necesară urmărirea acestei direcții.

Clasele, proprietățile și relațiile dintre ele se pot observa în figura 19.

Activități

Clasa care încorporează toate activitățile care se pot desfășura într-un curs am formalizat-o folosind termenul **edu:Activity.** Aceasta conține următoarele subclase: **edu:Assignment** (introduce temele), **edu:CommunicationTools** (introduce uneltele de comunicare), **edu:ExtraTools** (introduce unelte suplimentare utilizate). Instanțele acestei clase pot avea atașate informații suplimentare, folosind una din proprietățile: *edu:activityName* (introduce numele activității), *edu:activityRequirements* (informații despre cerințele activității), *edu:activityStartDate* și *edu:activityStopDate* (dacă este necesară introducerea unui interval de timp în care activitatea este activă), *edu:activityAuthor* (persoana care a introdus activitatea).

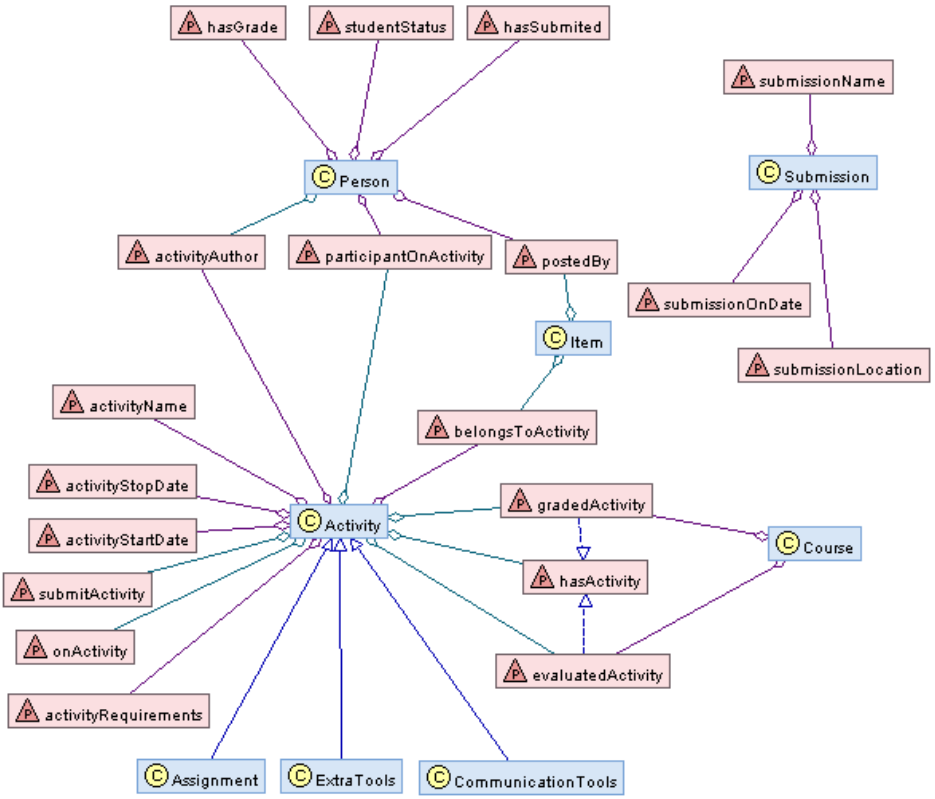


Figura 20. Elementele introduse pentru sub-ontologia Activity (fragment)

Datorită faptului că unele activități sunt utilizate în procesul de evaluare, trebuie precizat dacă tutorele trebuie să evalueze activitatea și dacă trebuie să acorde o notă. Acest lucru se face folosind *edu:evaluatedActivity* și *edu:gradedActivity.*

Dacă activitatea este evaluată sau notată, este necesară stabilirea promovabilității unui student. Acest pas se face folosind proprietățile *edu:hasGrade* (introdusă în sub-ontologia precedentă) și *edu:onActivity*, legate printr-un nod gol împreună cu proprietatea *edu:studentStatus*.

Activitățile pot fi legate de un curs. Conceptul formalizat care îndeplinește acest deziderat este proprietatea *edu:hasActivity.* Această proprietate este obligatorie doar în descrierea activităților de tip temă.

Un utilizator poate participa la mai multe activități în cadrul unui curs. În acest caz legătura este asigurată prin proprietatea *edu:participantOnActivity*, iar dreptul de a putea participa este asigurat prin înscrierea persoanei ca student sau tutore în cadrul cursului.

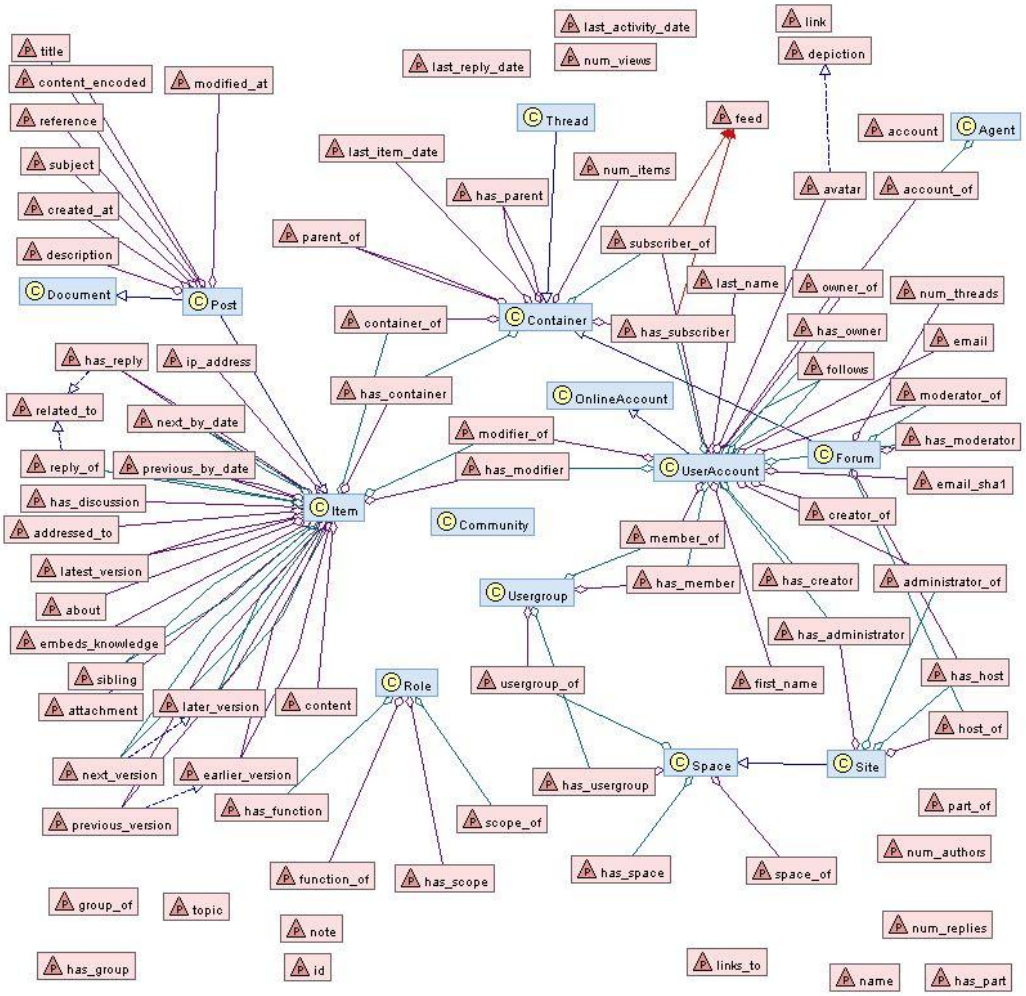


Figura 21. Ontologia SIOC (fragment)

Activitățile de tip temă presupun urcarea unui document într-un termen stabilit. Pentru a modela procesul de trimitere a temei, documentul se poate lega printr-un nod gol (ca și în cazul notării) prin intermediul proprietăţilor *edu:submitActivity* și *edu:hasSubmited.* Subiectul corespunzător acestui predicat este o instanță a clasei **edu:Submission** și dispune de următoarele proprietăți pentru a introduce informații: *edu:submissionName* (numele documentului), *edu:submissionLocation* (locația URL a documentului). Mai este necesară introducerea datei la care a fost adăugat documentul, în scenariile în care activitatea se desfășoară într-un interval de timp; pentru aceasta se utilizează proprietatea *edu:submissionOnDate.*

Pentru modelarea discuțiilor din cadrul uneltelor de comunicare propun spre utilizare ontologia SIOC (Semnatically-Interlinked Online Communities) [119]. Scopul acestei ontologii este de a furniza o platformă pentru modelarea activităților din comunitățile online (blog, forum, rss, etc.), fiind adoptată și utilizată în multe aplicații sociale [120]. Astfel, constituie un avantaj modelarea datelor din uneltele de comunicare folosind SIOC, deoarece permite fuzionarea mai multor surse de date și construcția de agenți software complecși care să proceseze aceste date. Un fragment din această ontologie este prezentat în figura 21. Se pot modela discuțiile folosind clasele **sioc:Item, sioc:Post, sioc:Container, sioc:Thread** și proprietățile care descriu instanțele acestor clase. În plus, trebuie asigurată legătura cu activitatea din curs, folosind *edu:belongsToActivity,* și cu persoana care a introdus articolul, folosind *edu:postedBy*.

Evaluare

Un element de testare este o instanță a uneia dintre următoarele clase: **edu:MultipleAnswer**, **edu:TrueFalse**, **edu:TextAnswer** sau **edu:Rank**. Aceste clase reprezintă tipul de întrebări disponibile în cadrul unui test si sunt subclase ale clasei **edu:Questions**. În cazul în care un element de testare este o instanță a clasei *edu:MultipleAnswer*, într-o bază de date de triplete cu capabilități de inferență (ex: baza RDFS nativă Sesame), se deduce că elementul de testare este, de asemenea, o instanță a clasei *edu:Questions*.

Pentru a organiza elementele de testare, sunt folosite două clase: ***edu:*** **edu:SectionOfCourse** și **edu:Course**. Proprietatea *edu:questionSectionOfCourse* asociază o secțiune la un anumit curs, iar *edu:questionBelongsTo* asociază o întrebare la o secțiune de curs. În acest fel, ne putem limita selecția de întrebări la un anumit subiect și / sau la un anumit curs.

Proprietățile *edu:wrongAnswer* și *edu:corectAnswer* sunt folosite pentru a defini răspunsurile pentru întrebările cu alegere multiplă și adevărat-fals, și sunt sub-proprietăți ale *edu:hasAnwser*. Pentru a limita numărul maxim de cuvinte acceptate ca răspuns la întrebări scurte, este utilizată proprietatea *edu:numberOfWords*. Toate răspunsurile sunt instanțe ale clasei **edu:Anwser**.

Atunci când o întrebare este introdusă pentru prima dată în baza de date RDF, este atribuit un nivel de dificultate de către profesor, folosind proprietatea *edu:difficultyLevel*. Această informație (nivelul de dificultate) este utilizată în generarea testelor, pentru a păstra dificultatea de ansamblu a testului într-un anumit interval. Gradul de dificultate al unei întrebări poate fi schimbat prin evaluarea rezultatelor obținute de studenți la acea întrebare în decursul timpului.

Modelarea opțiunilor studenților se realizează folosind un nod gol şi proprietățile: *edu:studentAnwser* (leagă studentul de nod), *edu:onQuestion* (specifică întrebarea la care s-a răspuns), *edu:choseAnswer* (ce răspuns a ales studentul), *edu:onTest* (la ce test). Testele vor fi instanțe ale clasei **edu:Test** și vor

avea proprietățile: *edu:testName* (numele testului), *edu:testDate* (data testării), *edu:testGraded* (specifică dacă testul este de auto-evaluare sau evaluare). Nota la un test se poate modela printr-un nod gol, folosind *edu:hasGrade* și *edu:onTest.*

Pentru a asigura că întrebările extrase din baza de date nu abordează același subiect, trebuie să fie declarate toate întrebările relaționate. Acest lucru se face prin utilizarea proprietății *edu:relatedQuestion,* care leagă toate întrebările relaționate de același nod gol.

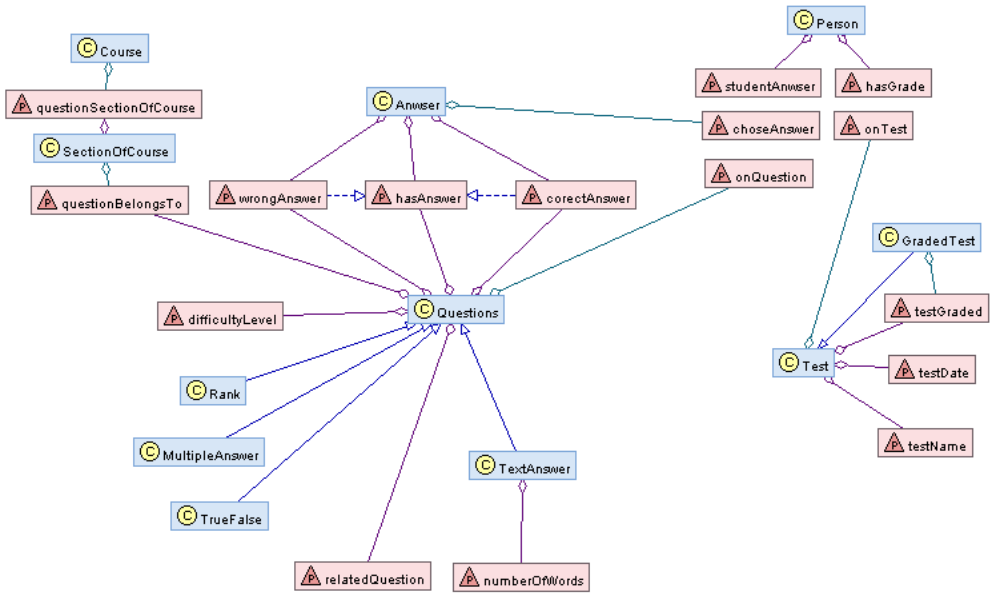


Figura 22. Elementele introduse pentru sub-ontologia Quiz (fragment)

4.2.4 Implementarea

Având conceptele formalizate mai sus, am folosit RDFS (RDF Schema), pentru a descrie clasele și proprietățile. Pentru a reduce numărul de triplete necesare pentru a descrie o întrebare, am utilizat următoarele proprietăți rdfs:subClassOf, rdfs:subPropertyOf, rdfs:domain, rdfs:range. Prin folosirea acestor proprietăți unele informații pot fi deduse. [14]

Pentru a adăuga o expresivitate mai mare se poate folosi limbajul OWL. Subclase pot fi declarate disjuncte, după cum este necesar, sau definite prin condițiile necesare și suficiente, lăsând sarcina de clasificare a instanțelor aparținătoare motoarelor de inferențiere.

Câteva exemple de situații în care OWL poate adăuga restricții suplimentare sunt: anumite instanțe de clase sunt autorizate să stabilească mai multe relații, în timp ce altele sunt limitate la una singură; clasele pot fi primitive sau definite, simple sau enumerate; proprietățile pot fi funcționale, invers funcționale sau simetrice și pot avea proprietăți inverse; intervalele proprietăților pot fi instanțe, clase sau uniuni de clase. Având în vedere lipsa unui motor de inferențe care să respecte toate regulile OWL, am decis să folosesc doar acele proprietăți care se pot infera în OWL-Lite.

Doar obiectele pot lua valori literale în triplete. Nu am considerat necesară impunerea tipului de date pe care obiectele anumitor proprietăți le pot avea, deoarece acestea pot varia de la sistem la sistem, iar bazele de date RDF nu validează oricum tipul datelor. Această responsabilitate este asigurată de agenţii software care procesează datele. Documentele rezultate se pot găsi la adresa [http://elearning.upt.ro/ontology/], iar domeniul de definiție al conceptelor este [http://elearning.upt.ro/ontology/edu#].

Modelul ontologiei educaționale prezentate a fost implementat utilizând formatele de serializare Turtle și RDF/XML. Graficele au fost generate utilizând unealta RDF Gravity [121], dezvoltată în cadrul Universității din Salzburg, Austria.

4.2.5 Evaluarea și mentenanța

Pentru evaluarea ontologiei și validarea posibilității utilizării ei, în capitolul următor voi descrie moduri de utilizare a acestui model în construcția de aplicații educaționale inteligente.

În procesul implementării acestor aplicații am ținut cont și de posibile corecții care trebuie efectuate asupra ontologiei pentru a îndeplini specificațiile fiecărui caz în parte (realizarea fișierului de mapare, partajarea de informații între aplicații, generarea automată de teste). În cazul în care am identificat lipsa unor concepte, proprietăți sau expresivitate prea scăzută, am revizuit toți pașii parcurși în construcția ontologiei. Acest proces de mentenanță trebuie repetat de fiecare dată când există inadvertențe între ontologie și datele ce se doresc a fi modelate.

Versiunea curentă prezentată în acest capitol îndeplinește cel puțin nevoile celor trei aplicații practice descrise în capitolul 5 (republicarea datelor folosind un fișier de mapare, partajarea de informații, generarea automată de teste).

4.3 Contribuții și concluzii

În acest capitol am descris modelul de ontologie educațională propus și mecanismele prin care sunt legate elementele pe care aceasta le conține. Am definit ontologia pentru a îndeplini cerințele unui sistem ontologic educațional bazat pe tehnologiile Web-ului Semantic.

Am analizat utilizarea ontologiilor în sisteme de e-Learning pentru învățământul superior. Am subliniat necesitatea de a dezvolta modele educaționale care îndeplinesc așteptările comunității învățământului superior, în ceea ce privește adaptarea și eficiența sistemelor de e-Learning, prin ontologii care utilizează tehnici specifice Web-ului Semantic.

Prezint în continuare, punctual, contribuţiile ce rezultă din acest capitol, atât cele teoretice, cât şi cele cu caracter aplicativ:

Studiul asupra metodologiilor de proiectare a unei ontologii SW

Pentru a putea proiecta ontologia, în prima fază a trebuit să definesc clar domeniul în care aceasta se încadrează și restricțiile care se deduc din această categorisire. Odată stabilit domeniul, am evaluat cele mai populare metodologii de proiectare ontologică, evaluând diferențele dintre acestea și am ales-o pe cea mai potrivită pentru situația de față. Am ajuns la concluzia că metodologia NeOn este cea mai potrivită, datorită flexibilității oferite de scenariile propuse. De asemenea, am sintetizat pașii necesari proiectării și implementării ontologiei.

Stabilirea specificaţilor ontologiei

Elementele ontologiei propuse sunt alese pentru a îndeplini cerințele unui sistem de e-Learning pentru învățământul superior și sunt menite să ofere: posibilitatea de a distinge între diferite tipuri de utilizatori și de a furniza date despre aceștia; informațiile necesare pentru o comunicare adecvată în rândul utilizatorilor; accesul corespunzător la materiale de studiu și activități educative; structurarea corectă a documentelor; o relație bine definită între activități și alte materiale disponibile. Datorită faptului că de obicei în aplicații se utilizează doar anumite părți din ontologii, am împărțit modelul în patru părți: persoane, cursuri, activități, evaluare.

Formalizarea și implementarea conceptelor ontologiei educaționale

După stabilirea specificațiilor și conceptualizare, am formalizat termeni din ontologie încercând să reutilizez pe cât posibil vocabulare sau ontologii deja standardizate. Astfel, am utilizat FOAF pentru a defini persoane, VCARD pentru anumite elemente de contact neincluse în FOAF, DC pentru a defini documente, SIOC pentru a descrie elementele din uneltele de comunicare. Extinzând aceste ontologii populare am asigurat compatibilitate cu sursele de date externe care le utilizează.

După formalizare am implementat conceptele folosind limbajul RDFS, limbaj mai puțin expresiv, dar suportat pe deplin de către motoare de inferențiere existente la ora actuală. Pentru situațiile în care am considerat necesară introducerea de expresivitate mai ridicată am utilizat OWL.

Ontologia educațională, cu clasele sale și relațiile stabilite între instanțele lor, reprezintă componenta centrală a unui sistem care ar permite publicarea datelor educaționale în format RDF. Toate structurile care conlucrează pentru construcția unui sistem de e-Learning le-am definit în interiorul ontologiei.

O parte din cercetarea prezentată în acest capabil a fost prezentată în lucrarea [122].

În concluzie, am definit o ontologie care poate fi utilizată pentru modelarea datelor provenite dintr-o platformă educațională, iar în capitolul următor voi demonstra utilitatea ei prin intermediul unor modele de aplicații practice.

5 Modele de aplicații semantic web pentru platforme educaționale

Având ontologia proiectată în capitolul precedent, în această parte a lucrării îmi propun să demonstrez utilitatea practică a tehnologiilor Web-ului Semantic în platformele educaționale și să evaluez calitatea modelului propus. Capitolul este compus din trei părți, care descriu: o metodă de conversie și publicare a datelor în format RDF, partajarea de informații între platforma Moodle și o aplicație externă utilizată într-un laborator virtual, precum și o metodă de generare de teste de evaluare sau autoevaluare folosind o bază de întrebări modelată semantic.

5.1 D2RQ și publicarea informației în format RDF

O dată cu maturizarea tehnologiilor Web-ului Semantic, necesitatea accesului la date non-RDF de către aplicații RDF va deveni din ce în ce mai stringentă [6]. Principalele surse de date non-RDF sunt bazele de date relaționale, principalul model de structurare a informațiilor în aplicațiile web. Pentru a permite agenților software semantici să acceseze datele din structuri relaționale, se pot utiliza două abordări:

- **Conversia datelor** în format RDF, folosind programe dedicate pentru fiecare bază de date în parte. În acest caz, dezavantajul principal îl reprezintă necesitatea repetării procesului de conversie, atunci când datele din baza de date se modifică.
- **Maparea structurii bazei de date relaționale,** folosind un server web care să permită rularea de interogări SPARQL direct pe bazele de date, folosind un fișier de mapare a structurilor conținute de baza de date relațională.

Platforma educațională utilizată în cadrul universității noastre este bazată pe Moodle, LCMS care păstrează datele utilizând sistemul de gestiune a bazelor de date MySQL. Datele se actualizează foarte des, astfel încât dacă aș utiliza conversia datelor pentru a face disponibile datele în format RDF este necesară repetarea procesului de conversie foarte des (consum ridicat de resurse de calcul) sau oferirea datelor aplicațiilor RDF în format incomplet sau eronat (situație inadmisibilă pentru construcția de aplicații ce au nevoie de date corecte în timp real). Maparea structurii permite accesul la datele relaționale direct în format RDF sau salvarea lor în baze de date RDF.

Datele din platforma educațională, publicate sub format RDF, pot fi utilizate de către agenți software inteligenți în diferite scenarii educaționale; două dintre acestea vor fi prezentate pe larg în următoarele subcapitole.

Un avantaj important corespunzător accesării datelor în format RDF îl constituie construcția de aplicații care utilizează cantitatea mare de informație disponibilă ca și Linked Open Data (LOD). Pentru a evidenția cantitatea de informații și domeniile din care aceasta a fost extrasă am atașat diagrama LOD (figura 23) [123]. Nodurile de dimensiune mare au cel puțin un miliard de triplete. Se estimează că în 2011 erau disponibile aproximativ 30 de miliarde de triplete, majoritatea obținute printr-un proces de mapare a bazelor de date relaționale [124]. Publicarea datelor compatibile LOD permite o partajare ușoară a acestora, ușurință în integrarea cu alte surse de date și construcția de interogări federative [125].

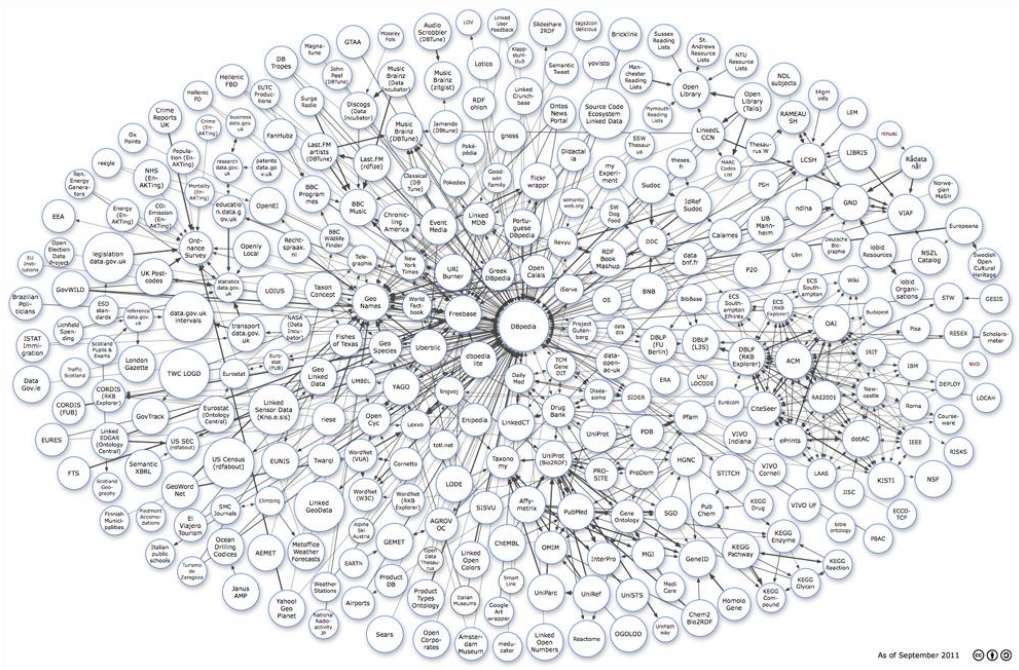


Figura 23. Diagrama Linked Open Data

Ca și soluții de mapare există mai multe metode și limbaje specifice sau adaptate. O soluție propusă de C. Perez și S. Conrand presupune introducerea

schemei *Relational.OWL* [126]. Aceștia propun pentru mapare utilizarea unui limbaj definit în RDF și OWL, care să formalizeze conceptele de tabel, cheie primară, cheie externă, relațiile dintre ele, etc., (din bazele de date relaționale) și utilizarea acestora pentru crearea fișierului de mapare.

Unealta *RDB2Onto* convertește datele relaționale în RDF, pe baza unor șabloane. Documentele generate prin completarea acestor șabloane se păstrează într-un motor de stocare semantic [127]. Chiar dacă este o soluție mai simplă decât utilizarea unui limbaj de mapare, nu se poate utiliza decât în situația în care dorim conversia datelor.

O soluție simplă de generare a fișierului de mapare, fără să fie necesară învățarea unui nou limbaj, o reprezintă unealta *RDOTE* [128]. Aceasta permite generarea mapării prin intermediul unei interfețe grafice, fiind foarte ușor de folosit de către utilizatori neexperimentați. Dezavantajul, la fel ca și la soluția precedentă, este dat de generarea datelor sub formă de fișiere, care apoi trebuie stocate pentru a putea fi interogate.

VisAVis [129] permite construcția unui fișier de mapare folosind unealta Protégé [130]. După generarea fișierului de mapare, datele se pot interoga folosind limbaje specifice Web-ului Semantic. Din păcate, proiectul a fost abandonat.

Ontograte cuprinde un set de unelte, proiectate pentru maparea automată a bazelor de date prin transformarea schemei într-o ontologie. Folosește inferențe pentru a rula interogări între diferite ontologii de mapare [131]. Nu este indicată utilizarea Ontograte în acest caz, datorită faptului că am o ontologie bine stabilită pe care trebuie executată maparea.

Platforma *D2RQ* permite accesarea bazelor de date relaționale utilizând grafuri RDF virtuale, create prin fișiere de mapare [6]. Limbajul de mapare d2rq este utilizat pentru crearea mapării tabelelor și coloanelor cu clase și proprietăți RDF, iar serverul D2R permite rularea de interogări SPARQL. Ca alternativă la utilizarea serverului D2R, oferă posibilitatea utilizării unei aplicații Java care oferă aceleași facilități, pe un server Tomcat. De asemenea, permite legătura cu medii de dezvoltare semantice, cum ar fi Jena și Sesame. O altă facilitate pe care nu am întâlnit-o la nici o altă soluție de mapare o reprezintă implementarea SPARQL/Update pentru a rula interogări de modificare SPARQL și conversia în interogări SQL update, permițând astfel sincronizări între o bază de date RDF și una relațională [132].

Din cele expuse mai sus, consider D2RQ cel mai adecvat pentru situația de față, având un set de funcționalități indispensabil unei mapări corecte a bazei de date Moodle. În procesul de mapare voi utiliza ontologia descrisă în capitolul 4.

5.1.1 Realizarea fișierului de mapare

Limbajul de mapare D2RQ este un limbaj declarativ, care permite descrierea legăturilor între o bază de date relațională și vocabulare / ontologii construite folosind RDFS sau OWL. Fișierul de mapare definește un graf virtual, care conține informații din baza de date. Este un concept similar cu vizualizările din limbajul SQL, diferența fiind dată de faptul că structura datelor virtuale este un graf RDF, în loc de un tabel virtual relațional. Graful RDF poate fi accesat în următoarele moduri: prin rularea de interogări SPARQL, ca și server Linked Data, se pot exporta datele într-un fișier, prin intermediul unei interfețe HTML sau din platformele de dezvoltare Jena sau Sesame.

Fișierul de mapare reprezintă un document RDF, care utilizează formatul de serializare Turtle pentru a păstra tripletele. Maparea este realizată utilizând termeni

din spațiul de nume D2RQ [http://www.wiwiss.fu-berlin.de/suhl/bizer/D2RQ/0.1#]. Maparea se poate face prin scrierea directă, într-un editor, a tripletelor necesare sau prin utilizarea uneltei de generare a fișierului de mapare, pentru a genera un schelet. Acesta este apoi editat pentru a utiliza ontologii formalizate. A doua variantă este de obicei mai ușor de utilizat, deoarece este mult mai rapidă.

Generarea fișierului de mapare, folosind unealta dump-rdf inclusă în pachetul D2RQ, presupune furnizarea datelor de acces la baza de date care se dorește a fi mapată. Aceasta generează fișierul de mapare pentru toate structurile la care are acces. Având în vedere numărul mare de structuri conținute în baza de date Moodle (350), precum și faptul că o mare parte dintre acestea nu prezintă interes pentru modelarea RDF, am creat un utilizator special pentru acest proces, cu drepturi limitate asupra structurilor care se doresc a fi mapate.

Folosind noul utilizator, am generat scheletul fișierului de mapare utilizând dump-rdf. Primul pas în procesul de editare a fișierului schelet a fost să asigur legătura cu baza de date. Aceasta se face prin inserarea tripletelor de mai jos, triplete care definesc driverul jdbc, adresa serverului de baze de date și datele de conectare utilizate.

```
map:database a d2rq:Database;
d2rq:jdbcDriver "com.mysql.jdbc.Driver";
d2rq:jdbcDSN "jdbc:mysql://adresăserver/numebd";
d2rq:username "utilizator";
d2rq:password "parolă";
jdbc:autoReconnect "true";
jdbc:zeroDateTimeBehavior "convertToNull";
.
```

Pentru mapare am utilizat o parte din ontologia definită în capitolul 4. Nu am utilizat subontologia de evaluări din două motive: băncile de întrebări stocate de către cele două module disponibile în Moodle, Feedback și Questionnaire, păstrează elementele de test în aceași coloană, separate prin virgulă, ceea ce ar presupune o procesare suplimentară sql asupra acestora, pentru a putea fi utilizabile; sunt foarte puține elemente de test disponibile în băncile de întrebări. Totuși, utilitatea acestei părți din ontologie o voi sublinia în al treilea subcapitol. De asemenea, nu am mapat momentan unelte de comunicare (wiki, blog, forum), folosind SIOC, deoarece fișierul ar deveni foarte complex, iar datele respective nu ar fi momentan utilizate în nici un agent software extern.

```
map:mdl_user a d2rq:ClassMap;
d2rq:dataStorage map:database;
d2rq:uriPattern "http://cv.upt.ro/cvdata/person/@@mdl_user.id@@";
d2rq:class foaf:Person;
.
```

Tabelul mdl_user din baza de date Moodle conține informații despre utilizatorii platformei. Astfel, definesc utilizând **d2rq:ClassMap** că orice tuplu din acest tabel este o instanța a clasei foaf:Person (clasă utilizată în subontologia persoane). Proprietatea *d2rq:uriPattern* precizează șablonul după care vor fi generate URI-urile instanțelor clasei *foaf:Person*. Secțiunea de cod de mai sus cuprinde triplete care

descriu cele expuse mai sus. Pentru celelalte declarări de clase, tripletele vor avea aceeași structură și, din acest motiv, nu voi mai introduce secțiunile respective de cod.

Pentru a mapa proprietățile care definesc legătura între o coloană a unui tabel și subiect, se utilizează clasa **d2rq:PropertyBridge**. Coloana *mdl_user.firstname* reprezintă prenumele utilizatorului, acesta fiind introdus în ontologie prin proprietatea *foaf:firstName*. Proprietatea din ontologie utilizată pentru mapare este precizată prin intermediul *d2rq:property,* iar coloana din tabel utilizată ca și obiect în triplet, prin *d2rq:column.*

```
map:mdl_user_firstname a d2rq:PropertyBridge;
d2rq:belongsToClassMap map:mdl_user;
d2rq:property foaf:firstName;
d2rq:column "mdl_user.firstname";
.
```

Într-un mod asemănător cu cel descris mai sus am mapat:

- coloana mdl_user.lastname la proprietatea *foaf:lastName*, reprezentând numele utilizatorului;
- coloana mdl_user.email la proprietatea *foaf:mbox*, reprezentând adresa de email a utilizatorului;
- coloana mdl_user.icq la proprietatea *foaf:icqChatID*, reprezentând adresa de icq a utilizatorului;
- coloana mdl_user.skype la proprietatea *foaf:skypeID*, reprezentând adresa de skype a utilizatorului;
- coloana mdl_user.yahoo la proprietatea *foaf:yahooChatID*, reprezentând adresa de yahoo a utilizatorului;
- coloana mdl_user.aim la proprietatea *foaf:aimChatID*, reprezentând adresa de aim a utilizatorului;
- coloana mdl_user.msn la proprietatea *foaf:msnChatID*, reprezentând adresa de msn a utilizatorului;
- coloana mdl_user.phone1 la proprietatea *foaf:phone*, reprezentând primul număr de telefon al utilizatorului;
- coloana mdl_user.phone2 la proprietatea *foaf:phone*, reprezentând a doilea număr de telefon al utilizatorului;
- coloana mdl_user.url la proprietatea *foaf:homepage*, reprezentând adresa web a utilizatorului;
- coloana mdl_user.address la proprietatea *vcard:street-address*, reprezentând adresa utilizatorului;
- coloana mdl_user.city la proprietatea *vcard:locality*, reprezentând orașul utilizatorului;

Tuplurile din tabela mdl_course reprezintă informații despre cursurile disponibile în platformă, motiv pentru care le-am mapat ca și instanțe ale clasei *edu:Course*. Coloanele mdl_course.fullname și mdl_course.shortname conțin numele scurt și lung al cursului, astfel încât le-am definit ca obiecte, în triplete în care predicatul este *edu:courseName.* Descrierea cursului este păstrată în coloana mdl_course. summary, iar pentru mapare am folosit *edu:courseDescription*.

În cazul în care trebuie definite legăturile între tabele, tripletele nu vor mai avea la obiect valori literale, ci alte instanțe de clase. Pentru a realiza acest lucru se

utilizează proprietățile *d2rq:refersToClassMap* (prin care se precizează cu ce clasă trebuie asigurată legătura) și *d2rq:join* (prin care se precizează direcția de legătură). Astfel, pentru coloana mdl_course.category se face legătura cu tabelul mdl_course_categories, pe care îl voi utiliza în definirea claselor curriculum și secțiune din curriculum.

```
map:mdl_course_category a d2rq:PropertyBridge;
d2rq:belongsToClassMap map:mdl_course;
d2rq:property edu:belongsToSectionOfCurriculum;
d2rq:refersToClassMap map:mdl_course_categories;
d2rq:join "mdl_course.category => mdl_course_categories.id";
.
```

În tabela mdl_course_categories se păstrează organizarea cursurilor. Astfel, dacă administratorul platformei Moodle păstrează cursurile organizate în curriculum, se pot mapa din acest tabel triplete care să definească curriculum și secțiunile pe care le conține. Am considerat toate tuplurile din acest tabel instanțe ale clasei *edu:SectionOfCurriculum*. Pentru păstrarea numelui secțiunii am utilizat proprietatea *rdfs:label*.

Pentru a descrie ierarhia de organizare a secțiunilor de curriculum trebuie realizată o legătură de tip join între câmpurile mdl_course_categories.id și mdl_course_categories.parent, pentru a introduce proprietatea edu:hasSection OfCurriculum. Aceasta trebuie condiționată, astfel încât părintele rădăcină să nu fie modelat. Condiția se introduce folosind proprietatea *d2rq:condition.*

```
map:mdl_course_categories_parent a d2rq:PropertyBridge;
d2rq:belongsToClassMap map:mdl_course_categories;
d2rq:property edu:hasSectionOfCurriculum;
d2rq:refersToClassMap map:mdl_course_categories;
d2rq:join "mdl_course_categories.parent =>
      mdl_course_categories_parents.id";
  d2rq:alias "mdl_course_categories AS mdl_course_categories_parents";
  d2rq:condition "mdl_course_categories_parents.id <> 0";
.
```

Secțiunile de curs se păstrează în tabela mdl_course_sections, acestea fiind instanțe ale clasei *edu:SectionOfCourse.* Pentru a specifica cărui curs aparține secțiunea respectivă, am introdus proprietatea *edu:hasSectionOfCourse,* care leagă o instanță a tabelului mdl_course de una a tabelului mdl_course_sections, printr-o operație de join. Pentru a păstra numele și descrierea secțiunii am mapat proprietățile *rdfs:label* la coloana mdl_course_section.name și *rdfs:comment* la coloana mdl_course_sections.summary.

Activitățile de tip temă sunt păstrate în tabelul mdl_assignment, tuplurile acestuia fiind instanțe ale clasei *edu:Assignment.* Numele temei este definit prin proprietatea *edu:activityName*, care mapează coloana mdl_assignment.name. Cerințele se găsesc în coloana mdl_assignment.intro și sunt mapate prin *edu:activityRequirements.* Informațiile referitoare la data de predare sunt modelate folosind *edu:activityStopDate* și *edu:activityStartDate.*

Temele sunt legate de un anumit curs printr-o legătură de tip join, iar maparea am făcut-o folosind trei proprietăți:

- *edu:hasActivity* – pentru toate tuplurile
- *edu:gradeActivity* – doar acele tupluri care au setată în coloana mdl_assignment.grade, o notă diferită de zero.
- *edu:evaluatedActivity* – tuplurile care au setată în coloana mdl_assignment.grade, valoarea zero.

Informațiile referitoare la predările de teme se păstrează în tabelul mdl_assignment_submissions, tuplurile acestuia fiind instanțe ale clasei *edu:Submission*. Data predării, nota și comentariile sunt mapate utilizând proprietățile *edu:submissinOnDate*, *edu:hasGrade* și *rdfs:comment*. Pentru a preciza la ce temă a avut loc predarea, am utilizat proprietatea *edu:onActivity,* care leagă printr-un join instanțe din clasa mdl_assignment_submissions cu mdl_asignment. Asemănător este precizată persoana care a predat tema, prin proprietatea *edu:hasSubmited*.

Pentru a preciza ce rol au utilizatorii într-un anumit curs, ar fi trebuit construit un join între cinci tabele. Pentru a simplifica maparea, am construit o vizualizare SQL care conține informațiile necesare, denumită cvmap_enroled. Rolul pe care un utilizator îl are într-un curs se precizează prin proprietatea *edu:hasStudent* sau *edu:hasTutor*. Pe lângă operația de join care asigură legătura între curs și utilizator mai trebuie precizată condiția care să ateste rolul utilizatorului. Acest lucru este implementat prin testarea coloanei cvmap_enroled.roleid; dacă aceasta conține cifra trei, rolul este de tutore, iar dacă conține cifra cinci, rolul este de student.

```
map:cvmap_enroled_students a d2rq:PropertyBridge;
d2rq:belongsToClassMap map:cvmap_enroled;
d2rq:property edu:hasStudent;
d2rq:refersToClassMap map:mdl_user;
d2rq:join "mdl_user.id <= cvmap_enroled.iduser";
d2rq:condition "cvmap_enroled.roleid = 5";
.
```

5.1.2 Publicarea datelor în format RDF

Având fișierul de mapare, acesta se poate utiliza în publicarea în format RDF, fie prin exportul acestora într-un fișier utilizând unealta rdf-dump, fie prin oferirea unor servicii de interogare (server linked data, aplicație war pentru Tomcat sau API în interiorul Jena sau Sesame).

După cum am precizat și la începutul subcapitolului, generarea unui fișier cu date prezintă un dezavantaj major: trebuie repetat procesul de export în cazul în care datele vor fi modificate în baza de date. Totuși, dacă se dorește renunțarea la baza de date relațională, atunci se pretează doar conversia datelor și încărcarea lor într-un motor de stocare specializat pentru triplete.

În cazul mapării bazei de date Moodle, aceasta va rămâne principala metodă de stocare și organizare a datelor, astfel încât este indicată folosirea fișierului de mapare într-un serviciu de interogare. Pentru aceasta, platforma D2RQ oferă serverul D2R, care permite navigarea și interogarea datelor în format RDF. De asemenea, permite interogarea ca și serviciu din utilitare externe (de ex: Sesame), folosind limbajul de interogare SPARQL 1.1. Acest lucru permite contruirea de interogări din mai multe surse de date.

Serverul D2R se poate rula din linie de comandă, pe calculatorul pe care a fost descărcat pachetul software, sau ca și aplicație web J2EE într-un servlet container, cum ar fi Apache Tomcat. A doua variantă este mai indicată pentru cazurile în care dorim să asigurăm o disponibilitate crescută a serviciului, cum ar fi furnizarea de date RDF pentru agenți software. Astfel, am considerat necesară realizarea arhivei war și urcarea ei pe serverul Tomcat. Pentru detalii de realizare a pachetului war se pot consulta specificațiile serverului D2R [133]. În documentul de mapare trebuie introdusă o secțiune de configurare a serverului, ca în exemplul următor:

```
# server config
@prefix d2r: <http://sites.wiwiss.fu-berlin.de/suhl/bizer/d2r-server/config.rdf#> .
@prefix meta: <http://www4.wiwiss.fu-berlin.de/bizer/d2r-server/metadata#> .

<> a d2r:Server;
 rdfs:label "D2R Server CV";
 d2r:baseURI <http://adresa_server/locație_d2r/>;
 d2r:port 8080;
 d2r:vocabularyIncludeInstances true;

 d2r:sparqlTimeout 300;
 d2r:pageTimeout 5;

 meta:datasetTitle "CV dataset" ;
 meta:datasetDescription "CV data." ;
 meta:datasetSource "cv" ;
 meta:operatorName "Bogdan" ;
 meta:operatorHomepage <http://bogdan.cm.upt.ro> ;
 .
# end server config
```

După urcarea aplicației pe serverul Tomcat, pentru a avea acces la date se pot rula interogări federative din orice aplicație care conține motor de interogare SPARQL 1.1, folosind clauza service în interiorul interogării select, sub forma prezentată mai jos:

```
select * where {
service < http://adresa_server/locație_d2r /sparql>
{?x ?y ?z.}
} limit 50
```

În plus, serverul D2R oferă acces la o interfață web pentru a naviga prin conceptele și instanțele mapate sau rularea de interogări direct din navigatorul web. Un exemplu de rulare de interogare din interfața grafică este cel prezentat în figura 24, unde se obțin toate tripletele legate de o anumită activitate.

Prin procesul de mapare și oferirea accesului la date ca și serviciu SPARQL, am permis accesul la informații disponibile în baza de date Moodle pentru agenți software semantici care ar putea oferi servicii foarte utile în mediul educațional. Printre acestea amintesc: sincronizarea cursurilor, sincronizarea utilizatorilor și

înscrierea studenților în cursuri, prin fuziunea datelor din mai multe surse (Moodle, baza de date a secretariatelor facultăților, etc.); partajarea datelor pentru aplicații conexe platformei educaționale, cum ar fi laboratoare virtuale (exemplu detaliat în subcapitolul următor); construcția de agenți care să îmbunătățească datele prin interogarea lor din mai multe surse (de ex: datele de contact din fișiere FOAF ar fi extinse); agenți software care să genereze agenda de contacte ale unui utilizator, pe baza cunoștințelor pe care le are; etc.

Momentan, prin fișierul de mapare implementat, am permis accesul la aproximativ 3,9 milioane de triplete RDF. Prin extinderea mapării către conceptele neincluse din ontologie și prin adăugarea de altele noi pot fi publicate mult mai multe triplete, dar ar scădea rapiditatea executării interogărilor. Din acest motiv procesul de mapare trebuie făcut rezonabil, în funcție de necesități.

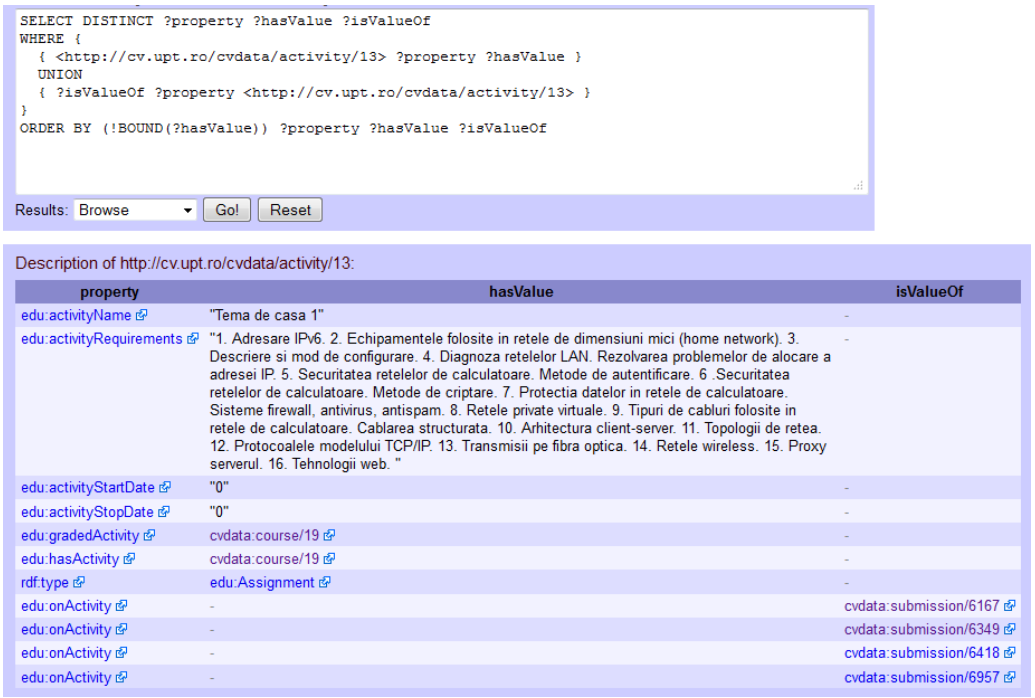


Figura 24. Rularea de interogări din interfața web D2R

5.2 Partajarea de informații cu aplicații externe

Începutul unui nou an școlar este de obicei o perioadă dificilă pentru persoanele implicate în activități didactice, în special în cazurile în care acestea necesită instalarea, configurarea și utilizarea unei soluții software complexe.

Evoluția tehnologiei informației și comunicațiilor a modificat metodele de predare și învățare, deschizând noi posibilități pentru comunități educaționale într-un sistem educațional global. [134]

În prima parte a activității mele doctorale, am încercat să proiectez și să implementez un sistem web care să permită simularea laboratoarelor care au loc pentru cursul de baze de date. Acest laborator virtual permite transferul informației

referitoare la baze de date, limbaje de programare și servere web prin intermediul unei aplicații web. În plus, sistemul permite studenților să implementeze și să testeze module pentru aplicații web dinamice, utilizând soluții în sursă deschisă [135].

Dezvoltarea sistemelor de învățământ virtual primește din ce în ce mai multă atenție în cercetare, într-o eră a Web-ului și a tehnologiilor virtuale, atât global cât și în România. Universitățile implementează platforme de e-Learning și utilizează e-Learning pentru a moderniza sistemul educațional. Majoritatea universităților din România încearcă să facă accesibile cursurile și în format electronic, pe Web. Totuși, infrastructura necesară dezvoltării și utilizării laboratoarelor virtuale este departe de a fi finalizată.

În domeniile inginerești, unde întâlnirile de laborator sunt indispensabile, studenții nu ar fi capabili să îndeplinească cerințele necesare obținerii diplomei fără să participe la acestea, fizic sau virtual.

Un laborator virtual este un sistem de calcul care permite partajarea resurselor fizice disponibile într-un laborator, între utilizatori conectați de la distanță prin intermediul Internetului. [136]

Cele mai comune alternative la laboratoarele reale, în învățământul la distanța, sunt:

- Laboratoare simulate;
- Laboratoare virtuale;
- Laboratoare cu control la distanță.

Principalele avantaje și dezavantaje ale acestor trei tipuri de laboratoare sunt prezentate în tabelul 6 [137]. După cum se observă din tabel, laboratoarele virtuale și cu control la distanță au un avantaj considerabil față de cele clasice, accesibilitatea.

Tabel 6. Avantaje și dezavantaje pentru laboratoarele reale, virtuale și cu control la distanță

Tip laborator	Avantaje	Dezavantaje
Real	- date reale - interacțiune directă cu echipamentul - muncă colaborativă - interacțiune cu tutorele	- restricții de timp și spațiu - necesită un orar fix - preț ridicat - necesită supervizare
Virtual	- bun pentru introducerea conceptelor - nu sunt restricții de timp și spațiu - mediu interactiv - cost scăzut	- date idealizate - lipsa colaborării - nu există interacțiune cu echipamentul real
Control de la distanță	- interacțiune cu echipamentul - date reale - fără restricții de timp și spațiu - cost mediu	- asigură doar prezența virtuală în laborator

5.2.1 Structura aplicației inițiale. Avantaje și dezavantaje.

Pentru a dezvolta laboratorul virtual, este recomandată utilizarea unei structuri modulare ca şi metodă de implementare, permițând reutilizarea componentelor în dezvoltarea de alte aplicații și posibilitatea de a extinde ușor serviciile oferite de aceasta. LEFTI, denumirea atribuită acestei aplicații, este compusă din două părți principale: o parte pentru administrarea conținutului și managementul aplicației, iar cea de-a doua destinată studenților.

Prin înregistrarea unui cont, de student sau administrator, aplicația va crea mediul de lucru alocat pentru un utilizator. Acesta constă dintr-un dosar, cu acces restricționat, precum și o bază de date accesibilă prin intermediul aceluiaşi cont (utilizator și parolă).

După validarea contului prin e-mail, studenții au dreptul de a gestiona fișiere în directorul propriu (creare, redenumire, editare, mutare) cu restricții cu privire la tipul fișierului și la dimensiunea lui. Au acces la baza lor de date, având posibilitatea de a rula interogări și de a vizualiza starea curentă a bazei lor de date.

Pentru scrierea sau editarea de scripturi, este încorporat un editor de text având următoarele funcționalități: afișează numărul liniei curente, folosește coduri de culoare pentru structura scriptului în mai multe limbaje de programare, suportă scurtături de editare a textului gen copiere / mutare.

Sistemul de management permite tutorelui: crearea conturilor studenților, monitorizarea activităților, monitorizarea cererilor de utilizator și a bazelor de date.

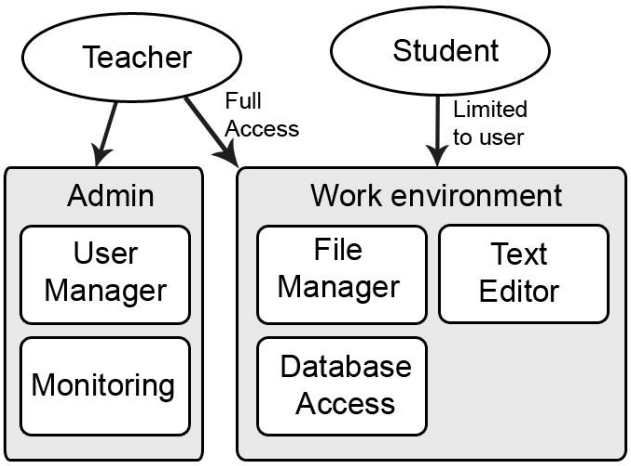


Figura 25. Structura aplicației

Prin monitorizarea activităților studenților în aplicație, profesorul poate folosi datele pentru a analiza procesul de predare și pentru a îmbunătăți sistemul de notare. Prin monitorizarea aplicației/bazei de date și prin acordarea accesului la fișierele studenților, profesorul are posibilitatea de a urmări evoluția acestora, dar şi posibilitatea de a interveni atunci când este necesar.

În cei patru ani în care această aplicație a fost utilizată de către studenți din sistemul de învățământ tradițional, precum și de către cei de la învățământ la distanță, am identificat următoarele avantaje și dezavantaje.

Beneficii. Laboratorul virtual exclude nevoia de instalare si configurare a serverelor și a pachetelor software necesare pentru cursul de baze de date. Aplicația oferă studenților oportunitatea de a pune în aplicare și a testa aplicații web dinamice printr-o interfață web intuitivă. Nu este nevoie de a instala serverul web Apache, serverul de baze de date MySQL sau limbajul de programare PHP, pentru fiecare participant la curs. Acest software este pre-instalat pe serverul care găzduiește aplicația de laborator virtual si este partajat de către toți participanții [1].

Cu toate acestea studenții sunt încurajați să creeze propriul lor mediu de dezvoltare AMP (Apache, MySQL, PHP), pentru a se familiariza cu procedura de instalare, dar succesul acestei operațiuni nu influențează capacitatea lor pentru a finaliza restul de sarcini.

Dezavantaje. Deși aplicația oferă unele informații despre activitățile de monitorizare a utilizatorilor, aceasta nu furnizează date cu privire la erorile pe care studenții le-au întâmpinat în sarcinile lor. Cu astfel de informații profesorul poate îmbunătăți metodele de predare sau materialele de laborator.

O altă problemă este lipsa de informație partajată între laboratorul virtual și platforma educațională a universității, bazată pe Moodle. Scopul este ca cele două aplicații să fie capabile de schimb de informații cu privire la temele, activitățile și notele studenților.

De asemenea, aplicația nu permite colaborarea între studenți pe o anumită sarcină. Acest lucru ar putea fi o problemă pentru profesor, în unele scenarii de învățare (tema este prea mare pentru un singur student, un exercițiu în care munca în echipă este încurajată).

5.2.2 Structura aplicației modificate

O soluție pentru neajunsurile descrise mai sus o reprezintă utilizarea tehnologiilor Web-ului Semantic. Am ales această abordare, deoarece Web-ul Semantic permite schimbul de informații între aplicații, într-un mod automat, fără intervenție umană. Lizibilitatea semantică a informației de către mașini este mai mult decât un simplu schimb de biți. Ea permite de asemenea transferul sensului informației.

În figura 26 este reprezentată structura modificată a aplicaţiei. În centru este baza de date RDF, cu rolul de a stoca și combina informații din surse diferite. Un convertor este necesar pentru obținerea de informații în format RDF despre contul studentului, temele, notele și activitatea sa.

Blocul de monitorizarea a erorilor va avea sarcina de a colecta informații cu privire la problemele întâmpinate de studenți în sarcinile lor. Aceste date sunt convertite în RDF și stocate în baza de date RDF.

Blocul de statistică utilizează date colectate din Moodle și din blocul de monitorizare a erorilor pentru a genera rapoarte despre activitățile studenților. Utilizând blocul de notare, profesorul poate evalua o temă a unui student. Nota este convertită în RDF și trimisă pentru stocare.

Un bloc nou trebuie să fie construit în Moodle pentru a permite interacțiunea cu baza de date RDF și sincronizarea statisticilor și notelor.

Pentru a asigura accesul și păstrarea datelor în format RDF, am ales să utilizez platforma Sesame, care oferă implicit motoare de stocare cu capabilități de inferențiere RDFS și motor de interogare SPARQL 1.1. Procesul de instalare consistă din urcarea arhivei WAR openrdf-sesame pe un server Apache Tomcat. Pentru a controla accesul la datele stocate, am configurat serverul Tomcat astfel încât să nu

permită accesul la aplicația Sesame folosind metodele GET, POST, PUT, DELETE (utilizate pentru a trimite comenzi aplicației) decât prin validarea drepturilor printr-un proces de autentificare.

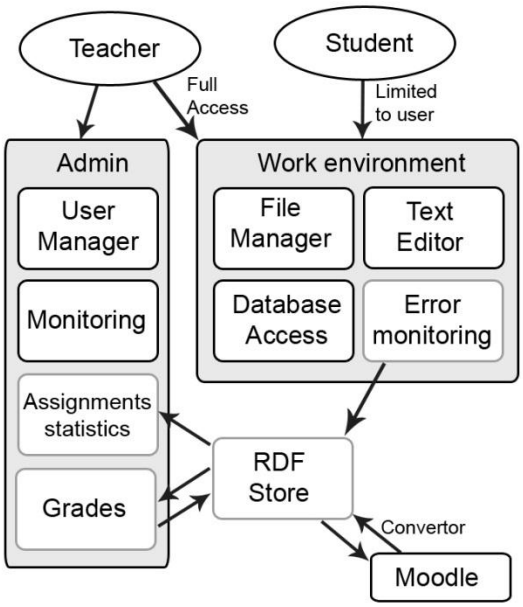


Figura 26. Structura modificată a aplicației

Accesul la datele păstrate în baza de date Moodle se face interogând serverul D2R, prezentat în subcapitolul precedent. Din motive de securitate, interogările pe acest server au fost limitate la o listă restrânsă de adrese IP. Nu am putut să folosesc un proces de autentificare, deoarece SPARQL nu are integrată o procedură de autentificare.

Pentru a avea acces și la datele din aplicația LEFTI trebuie realizat și pentru aceasta un fișier de mapare a tabelelor din baza de date. Tabelele mapate în acest caz sunt lefti_users și lefti_users_trafic.

Tuplurile din tabela lefti_users reprezintă informații despre conturile de utilizatori disponibile în platformă, motiv pentru care le-am mapat ca și instanțe ale clasei *edu:Person,* dar cu un URI de identificare diferit de cel utilizat pentru Moodle. Coloana lefti_users.nume am definit-o ca obiect în triplete în care predicatul este *foaf:name.* Adresa de email este păstrată în coloana lefti_users.email, iar pentru mapare am folosit *foaf:mbox*.

Tabelul cms_users_trafic conține informațiile de utilizare a aplicației de către utilizatori. În acest caz tuplurile le-am declarat ca instanțe ale clasei *lefti:Trafic.* Pentru a păstra legătura între instanța de trafic și utilizator, am introdus proprietatea lefti:hasTrafic printr-o operație de join. Adresa IP de pe care a fost accesată aplicația se mapează utilizând proprietatea *lefti:hasIp*, iar data prin proprietatea *lefti:hasDate.*

Pentru a folosi fișierul de mapare rezultat, am urmat aceiași pași ca şi în cazul platformei Moodle, pentru a asigura accesul la datele din aplicația LEFT în format

RDF (prin intermediul unei noi instanțe a serverului D2R). La fel ca și în cazul precedent, din motive de siguranță, am limitat accesul la o listă de adrese IP.

Pentru a obține date din cele două surse, se pot rula interogări federative din interiorul Sesame, având o bază de date RDF în care să fie declarate spațiile de nume utilizate în fișierele de mapare.

Alinierea datelor, pentru a afla ce persoană din baza de date Lefti corespunde unei persoane din baza de date Moodle, se poate face utilizând câmpul care conține adresa de email. În acest caz, trebuie specificat utilizatorilor aplicației Lefti, faptul că pentru a avea date legate trebuie să utilizeze aceeași adresă de email ca în platforma Moodle. O altă abordare este implementarea unei soluții de reutilizare a contului de Moodle pentru autentificare, asemănătoare cu cele descrise în lucrarea [110], sau implementarea unei soluții care utilizează datele mapate prin serverul D2R. Chiar și în cazul în care utilizatorii nu au fost atenționați să utilizeze aceeași adresă de email, din 97 de studenți înscriși la cursul de Baze de date, 94 au folosit aceeași adresă.

Pentru a lega identificatorii utilizatorilor celor două sisteme după adresa de email se poate utiliza următoarea interogare federativă, în care se filtrează câmpurile de email necompletate. Această interogare este descrisă utilizând algebra SPARQL.

```
join
(service <http://adresă_server/cvmap/sparql>
  (filter (!= (str ?emailcomun) ""))
  (bgp
  (triple <http://cv.upt.ro/cvdata/course/768> edu:hasStudent ?personmoodle)
  (triple ?personmoodle foaf:mbox ?emailcomun)
)))
  (service <http://adresă_server/leftimap/sparql>
  (filter (!= (str ?emailcomun) ""))
  (bgp (triple ?personlefti foaf:mbox ?emailcomun)
)))
```

Obținerea informațiilor de contact disponibile în baza de date Moodle, pentru un anumit utilizator din aplicația LEFTI, se realizează precizând adresa URI a persoanei, în serverul D2R destinat aplicației LEFTI, iar în partea de interogare federativă, a datelor din Moodle care specifică șablonul de extragere a informațiilor de contact dorite. Un exemplu de implementare, în algebra limbajului SPARQL, este prezentat în cele ce urmează:

```
join
(service <http://adresă_server/cvmap/sparql>
  (filter (!= (str ?emailcomun) ""))
  (bgp
  (triple <http://cv.upt.ro/cvdata/course/768> edu:hasStudent ?personmoodle)
  (triple ?personmoodle foaf:mbox ?emailcomun)
  (triple ?personmoodle foaf:firstName ?prenume)
  (triple ?personmoodle foaf:lastName ?nume)
  (triple ?personmoodle vcard:locality ?oras)
)))
```

```
(service <http://adresă_server/leftimap/sparql>
  (filter (!= (str ?emailcomun) "")
  (bgp (triple <http://cv.upt.ro/leftidata/person/66> foaf:mbox ?emailcomun)
)))
```

Pentru a obține lista de teme predate de către un student, se specifică adresa URI de identificare a persoanei în serverul D2R destinat aplicației LEFTI, iar în partea de interogare federativă a datelor din Moodle, se specifică șablonul de extragere a informațiilor referitoare la temele predate în contextul cursului de Baze de date. Această cerință se poate implementa în algebra limbajului SPARQL în modul următor:

```
join
  (service <http://193.226.10.115:8080/cvmap/sparql>
  (filter (!= (str ?emailcomun) "")
  (bgp
  (triple <http://cv.upt.ro/cvdata/course/768> edu:hasStudent ?personmoodle)
  (triple ?personmoodle foaf:mbox ?emailcomun)
  (triple ?activitate edu:hasActivity <http://cv.upt.ro/cvdata/course/768>)
  (triple ?submission edu:onActivity ?activitate)
  (triple ?submission edu:hasSubmited ?personmoodle)
  (triple ?submission edu:submissionOnDate ?datapredare)
)))
  (service <http://193.226.10.115:8080/leftimap/sparql>
  (filter (!= (str ?emailcomun) "")
  (bgp (triple <http://cv.upt.ro/leftidata/person/66> foaf:mbox ?emailcomun)
)))
```

Acestea sunt doar câteva exemple care permit obţinerea de date îmbogățite, folosind cele două surse de informații. Folosind informația de trafic din Lefti şi Moodle, tutorele poate să aibă o imagine de ansamblu cu privire la interesul studentului. De asemenea, păstrarea erorilor generate în implementarea cerințelor de laborator sau a temelor, poate atrage atenția tutorelui asupra deficiențelor în cunoștințele studenților. Pentru a păstra aceste date este nevoie de un bloc care să monitorizeze și să convertească erorile în triplete RDF. Datele astfel obținute se vor păstra în baza de date RDF din Sesame, utilizată și în interogările precedente.

Notele acordate folosind aplicația LEFTI se vor păstra în aceeași bază de date RDF Sesame. Pentru a urca aceste note și în baza de date Moodle se poate utiliza limbajul D2RQ/Update. Acesta este o extensie a limbajului de mapare D2RQ, care permite conversia unui fișier de mapare, din date RDF, în instrucțiuni SQL de inserare [132]. Același limbaj se poate utiliza și pentru celelalte sincronizări de date necesare.

În acest subcapitol, am prezentat o soluție pentru îmbunătățirea laboratorului virtual utilizat în laboratoare de programare. Am descris structura aplicației și modul de funcționare, identificând punctele sale forte și punctele slabe. Am propus o metodă de îmbunătățire a aplicației, utilizând tehnologii ale Web-ului Semantic pentru a corecta neajunsurile identificate.

5.3 Generarea automată de teste

În acest subcapitol voi descrie abordarea propusă privind generarea automată de teste. Pentru ca sistemul să funcționeze, am organizat întrebările legate de un anumit subiect astfel încât să fie disponibile pentru aplicații. Pentru a face acest lucru, am folosit tehnologii ale Web-ului Semantic pentru a construi un vocabular, care definește toate conceptele referitoare la întrebări în format RDF (eticheta întrebării, tipul de întrebări, răspunsul corect, răspunsuri greșite, relațiile dintre întrebări, etc.). Datele rezultate trebuie să fie stocate într-o bază de date RDF. Pentru a genera un test din banca de întrebări (în acest caz o bază de date RDF), am definit modele pentru extragerea de întrebări aleatorii și independente. Evaluarea vocabularului și a modelelor de interogare am realizat-o prin rularea algoritmului de generare a testelor de un număr de ori, în diferite scenarii, asigurându-mă că întrebările au o distribuție de utilizare normală în teste. Această metodă de generare de teste ar putea fi utilizată de către tutori pentru a reduce volumul de muncă al acestora, precum și de către studenți pentru auto-evaluare.

Pentru a asigura măsurarea corectă a performanței academice a studenților, cadrele didactice trebuie să proiecteze teste de evaluare cât mai științific și profesional cu putință. Cu toate acestea, pregătirea atentă a elementelor de testare și evaluare a rezultatelor este consumatoare de timp, prin urmare fiind o parte scumpă a producției unui curs.

În teoria clasică a testelor, scorul obținut prin testare este format din mai multe elemente. Scorul brut obținut de un student este alcătuit dintr-o componentă adevărat și o componentă de eroare aleatorie. Scorul adevărat al unei persoane poate fi găsit prin media scorurilor tuturor testelor luate de către o persoană, dacă ar putea să participe la un număr infinit de sesiuni de evaluare [138]. Deoarece acest lucru nu este posibil, trebuie redusă la minim eroarea prin proiectarea testelor.

Metoda prin care un profesor proiectează elementele unui test, este legată de contextul și materia pentru care acesta este construit. Principalele etape necesare în scopul proiectării testelor de evaluare sunt [90]:

- Identificarea scopului (evaluare / auto-evaluare);
- Stabilirea specificațiilor testului (numărul de elemente de testare, tip întrebări, timpul estimat pentru finalizare, etc.);
- Construirea bazei inițiale de elemente de testare;
- Revizuirea elementelor de testare;
- Evaluarea preliminară a elementelor;
- Verificarea elementelor de testare pe un eșantion mare al populației evaluate;
- Determinarea proprietăților statistice ale scorurilor, pe elemente de testare;
- Publicarea dovezii de fiabilitate și de valabilitate a testului final;
- Elaborarea de ghiduri pentru administrarea, notarea și interpretarea rezultatelor testelor.

Calitatea elementelor de testare poate fi ușor evaluată de către profesor. El poate construi elemente de testare extrem de concentrate și detaliate, bazate pe cursul sau unitățile de învățare. Știind ce material a fost prezentat studenților, profesorul poate identifica elemente de testare cu care ar trebui să fie familiarizați aceştia și care implică o înțelegere mai aprofundată a cursului [139]. Acest lucru face ca profesorul să fie candidatul perfect pentru crearea de elemente de testare.

Din pașii de mai sus putem presupune că generarea de teste va consuma o cantitate mare de timp profesorului. Deși nu putem interveni în proiectarea elementelor testului, ne putem asigura că procesul de revizuire, testare a elementelor, generare a testelor, evaluare a testelor se realizează într-un sistem automatizat.

Un alt factor care m-a motivat în proiectarea unui sistem automat de generare de teste a fost acela de a îmbunătăți distribuția statistică a notelor. Figura 27 arată distribuția statistică a notelor pentru cursurile *Programare Orientată pe Obiecte* și *Sisteme de Gestiune a Datelor*, din cadrul universității nostre, din 2009 până în 2011. Aceste cursuri au o evaluare distribuită, ceea ce implică un test pentru fiecare jumătate de semestru. Graficele unu și doi arată distribuția statistică a notelor pentru *Sisteme de Gestiune a Datelor*, pentru prima (ED1) și a doua (ED2) evaluare, în timp ce graficele trei si patru arată distribuția statistică a notelor pentru *Programare Orientată pe Obiecte*, pentru prima (ED1) și a doua (ED2) evaluare. Pe fiecare grafic sunt prezentate curbele de distribuție a notelor pentru evaluare (ED) și re-evaluare (RED), între 2009 și 2011. Putem observa că unele dintre curbele de distribuție nu seamănă cu distribuția Gaussiană, dar şi faptul că acestea sunt foarte diferite de la o evaluare la alta.

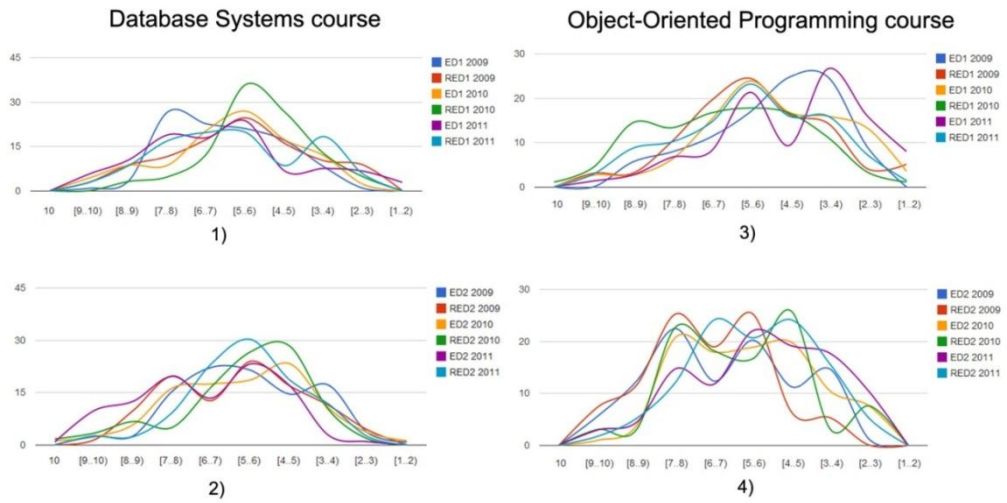


Figura 27. Distribuția notelor pentru Sisteme de Gestiune a Datelor (1. și 2.) și Programare Orientată pe Obiecte (3. și 4.) între anii 2009 - 2011

Aceste teste au fost generate manual de către profesor, prin alegerea unui număr cuprins între 18 - 24 de elemente de testare independente, dintr-o bancă de întrebări, pe baza experienţei sale. Întrebările pot fi cu răspunsuri multiple, de tip adevărat-fals sau întrebare cu răspuns scurt.

Scopul cercetării mele a fost acela de a construi un sistem care va reduce volumul de muncă al cadrelor didactice, prin furnizarea unui mijloc de construcție automată a testelor, evaluarea dificultății testelor, generarea de statistici și în același timp partajarea unei părți din date cu agenți software sau aplicații. Pentru a realiza operaţiile enumerate mai sus, am utilizat tehnologii ale Web-ului Semantic care permit stocarea informațiilor pentru consumul acestora în aplicații. Datele sunt bine definite și organizate pentru a forma expresii bogate, integrarea și partajarea

este simplificată. Datele au capabilități inferențiale și permit extragerea de informații semnificative în timp ce rămân distribuite. [12]

O abordare semantică în generarea de teste este prezentată în articolele [140] [141], unde se propune o ontologie care să formalizeze termenii-cheie în dezvoltarea de teste și să definească funcționalitatea și componentele unui sistem de generare de teste. Un alt exemplu este utilizarea de Linked Open Data (LOD) în generarea semi-automată a artefactelor de învățare [142]. În cazul meu, testele sunt concepute de profesor, pe baza cunoștințelor predate. LOD nu pot fi utilizate pentru a evalua studenţii, deoarece răspunsurile întrebărilor trebuie să fie în materialele de curs.

Primul pas în construcția sistemului este utilizarea unui mediu de stocare care să permită agenților software sau aplicaților să acceseze și să interacționeze cu datele. Banca de întrebări poate fi convertită în RDF și inserată într-o bază de date de triplete. Pentru a introduce conceptele, relațiile și constrângerile elementelor de testare, am definit o mică ontologie. După conversia și stocarea întrebărilor în format RDF, am creat un prototip de generare a testelor.

5.3.1 Ontologia Test

Pentru a proiecta ontologia necesară pentru a formaliza concepte, relații și constrângeri, a trebuit să identific tipul de elemente de testare. La evaluarea băncilor de întrebări pentru cursurile Programare Orientată pe Obiecte și Sisteme de Gestiune a Datelor am observat următoarele tipuri de elemente de testare:

- Alegere multiplă, în cazul în care studentul trebuie să selecteze toate răspunsurile corecte din lista de opțiuni;
- Adevărat-fals, similar cu alegere multiplă, cu doar două răspunsuri posibile și doar una corectă;
- Răspuns scurt, caz în care studentul va trebui să dea un răspuns scurt.

Pe lângă cazurile de mai sus, studenții pot avea o întrebare cu variante multiple, unde logica de răspuns este de a selecta un singur răspuns corect. În unele scenarii se pot utiliza întrebări de gradare, caz în care studentul trebuie să compare diferite elemente.

Pentru organizarea întrebărilor avem următoarele constrângeri:

- Întrebările aparțin de o anumită secțiune a cursului și pot fi legate de alte întrebări pe aceeași temă;
- Secțiunile aparțin unui anumit curs.

Având constrângerile prezentate mai sus, am utilizat din ontologia proiectată în capitolul 4 conceptele prezentate în figura 24. Am folosit RDFS (RDF Schema) pentru a descrie clasele și proprietățile. Pentru a reduce numărul de triplete necesare pentru a descrie o întrebare, am utilizat următoarele proprietăți rdfs:subClassOf, rdfs:domain, rdfs:range. Prin folosirea acestor proprietăți, unele informații pot fi deduse. [14]

Un element de testare este o instanță a unuia dintre următoarele clase *MultipleAnswer, TrueFalse, TextAnswer* sau *Rank*. Aceste clase reprezintă tipul de întrebări disponibile în cadrul unui test şi sunt subclase ale clasei *Questions*. În acest fel, dacă un element de testare este o instanță a clasei *MultipleAnswer*, într-o bază de date de triplete cu capabilități de inferență (ex: baza RDFS nativă Sesame), se deduce că elementul de testare este, de asemenea, o instanță a clasei *Questions*.

Pentru a organiza elementele de testare sunt folosite două clase: *SectionOfCourse* și *Course*. Proprietatea *questionSectionOfCourse* asociază o

secțiune la un anumit curs. Astfel, am putut limita selecția de întrebări la un anumit subiect și / sau la un anumit curs.

Proprietățile *wrongAnswer* și *corectAnswer* sunt folosite pentru a defini răspunsurile pentru întrebările cu alegere multiplă și adevărat-fals. Pentru a limita numărul maxim de cuvinte acceptate ca răspuns la întrebări scurte, poate fi utilizată proprietatea *numberOfWords*.

Atunci când o întrebare este introdusă pentru prima dată în baza de date RDF, este atribuit un nivel de dificultate de către profesor, folosind proprietatea *difficultyLevel*. Această informație este utilizată în generarea testelor pentru a păstra dificultatea de ansamblu a testului într-un anumit interval. Gradul de dificultate al unei întrebări poate fi schimbat prin evaluarea rezultatelor obținute de studenți la acea întrebare, în decursul timpului.

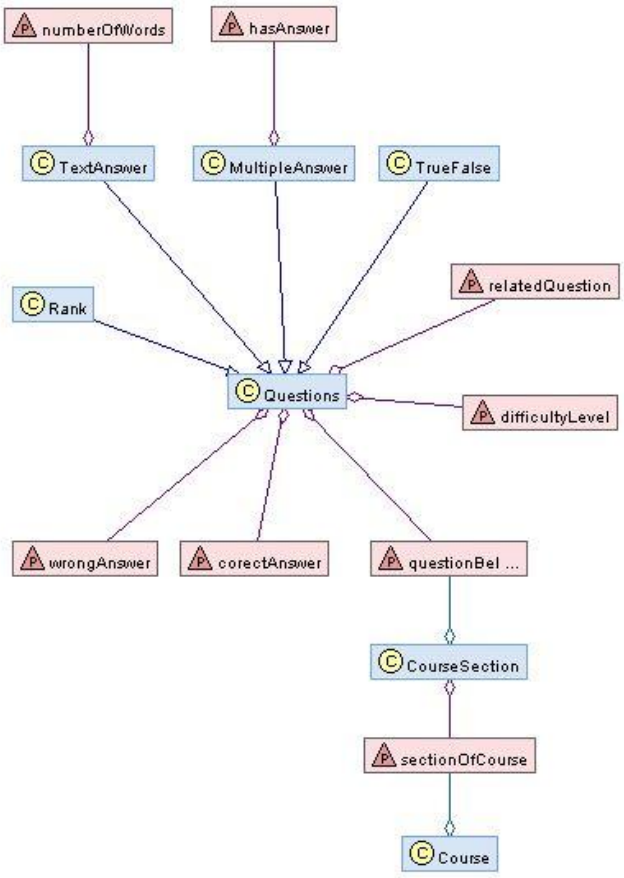


Figura 28. Graficul ontologiei Test (fragment)

Pentru a asigura că întrebările extrase din baza de date nu abordează același subiect, trebuie să fie declarate toate întrebările relaționate. Acest lucru se face prin

utilizarea proprietății *relatedQuestion,* care leagă toate întrebările relaționate de același nod gol.

5.3.2 Generarea testelor

Pentru a genera teste, în primul rând avem nevoie să stabilim specificațiile testului: numărul de întrebări și distribuția după tipul de întrebare, din ce secțiuni și cursuri sunt selectate elementele de testare, precum și dificultatea de ansamblu a testului. Fiecare tip de întrebare poate apărea un număr de ori, dintr-un interval stabilit sau un număr fix, iar suma tuturor elementelor de testare trebuie să fie un număr specificat de către tutore. Tutorele trebuie să specifice dacă elementele de testare sunt necesare pentru evaluare distribuită, evaluarea unui anumit subiect sau auto-evaluare.

Pentru a selecta elementele de testare și pentru a extrage toate informațiile necesare, am construit o interogare SPARQL, care satisface formula de mai jos. Semnificația parametrilor este: tipul de întrebare pe care am dorit să o selectez (t), o listă de identificatori pentru elementele de testare deja selectate (R) și o listă de secțiuni de curs de care aparține elementul de testare. Elementul de testare este selectat aleatoriu, și trebuie să fie de tip specificat (t), fără legătură cu elementele deja selectate (R) și să fie dintr-o secțiune selectată (S). Procesul de selecție se repetă pentru fiecare tip de întrebare, în funcţie de numărul specificat de către tutore (număr fix sau număr dintr-un interval).

$$Question(t,R,S) = \{\, q \mid \mathrm{q} \in \mathrm{Q} \text{ and } \mathrm{q} \notin \mathrm{R} \text{ and } \mathrm{q} \in \mathrm{S} \text{ and } \mathrm{q}(t)\ returns\ true\}$$

Unde:
t – tipul întrebării
R – denotă elementele de testare deja extrase
S – denotă secțiunile de curs de care aparțin elementele de testare
q – întrebarea selectată

Dacă întrebarea selectată este de tip alegere multiplă sau adevărat-fals, trebuie selectate răspunsurile pentru acel element. Acest proces se realizează prin construirea unei interogări SPARQL care satisface formula de mai jos. Pentru întrebările cu variante multiple de răspuns, tutorele poate specifica un număr fix de opțiuni prin parametrul *n*. Pentru întrebările adevărat-fals numai răspunsul corect poate fi selectat (adevărat sau fals); în acest caz n este egal cu unu.

$$Answers(q,n) = \{\, a \mid a \in \mathrm{A} \text{ and } a(q)\ returns\ true\}$$

Unde:
q – întrebarea pentru care au fost selectate răspunsuri
n – numărul de răspunsuri selectate

Atunci când procesul de selecție a fost încheiat (toate elementele necesare generării testului, întrebări și răspunsuri, au fost selectate), testul trebuie să fie validat pentru un anumit nivel de dificultate generală și datele trebuie puse la dispoziția unei aplicații externe care furnizează testul studentului (modul generare PDF, modul de testare on-line).

5.3.3 Evaluare

Pentru a testa ontologia și metoda de generare, am convertit banca de întrebări pentru cursul Sisteme de Gestiune a Datelor în format RDF, folosind clasele și proprietăți menționate mai sus pentru a defini datele. Banca de întrebări conține 215 întrebări (164 de tip alegere multiplă, 21 cu răspuns scurt și 30 de tip adevărat-fals), împărțite în zece secțiuni. Prin conversie s-au produs 1817 triplete, din care 430 reprezintă relațiile dintre întrebări (așa cum se vede în figura 29, secțiunea 2.), 215 reprezintă legătura dintre o întrebare și o secțiune, iar cea mai mare parte din restul tripletelor reprezintă răspunsurile (după cum se vede în figura 29, secțiunea 1.) și etichetele.

Tripletele RDF au fost încărcate într-o bază de date nativă RDFS Sesame alături de tripletele care definesc ontologia, pentru a face uz de capacitățile de inferență. Sesame este o platformă în sursă deschisă scrisă în Java, care oferă posibilitatea de a stoca și interoga date RDF, având capabilități de inferențiere RDFS fără a fi necesare configurări suplimentare. [143]

Rularea unei interogări, pentru a număra toate tripletele cu activarea procesului de inferențiere, va produce 4429 triplete. Există șase clase specifice ontologiei de test care nu au instanțe declarate în datele originale RDF, dar au fost deduse: clasa *Course* cu o singură instanță (deoarece datele sunt pentru un singur curs); clasa *CourseSection* cu zece triplete; clasa *Questions* are 215 instanțe și reprezintă numărul total de întrebări din bancă; clasa *MultipleAnswer* are 164 instanțe și reprezintă numărul de întrebări de tip alegere multiplă; clasa *TrueFalse* are 30 de instanțe și reprezintă numărul de întrebări de tip adevărat-fals; clasa *TextAnswer* are 21 de instanțe și reprezintă numărul de întrebări cu răspuns scurt.

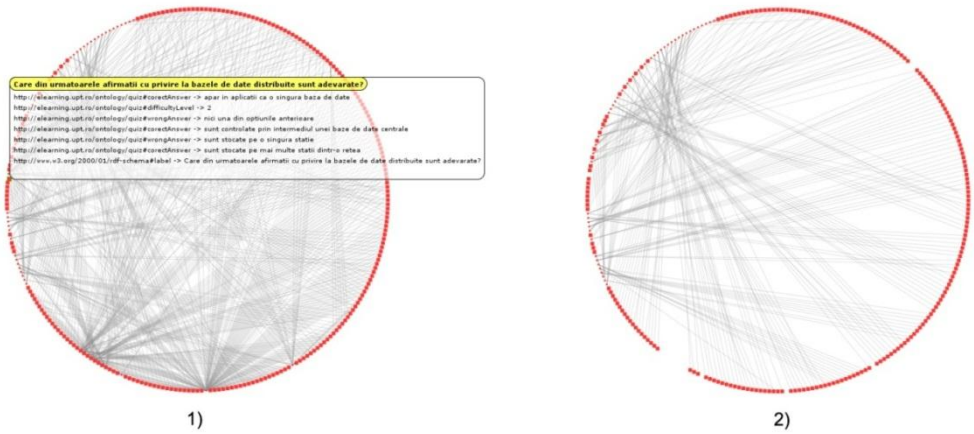


Figura 29. Toate nodurile și muchiile (1.) și nodurile și muchiile ce descriu relațiile între întrebări (2.) pentru cursul Sisteme de Gestiune a Datelor.

Proprietățile din ontologia de testare și numărul de triplete în care acestea sunt utilizate sunt: *questionBelongsTo, dificultyLevel* și *relatedQuestion*, apar în 215 triplete (numărul exact de întrebări descrise, deoarece acestea sunt specifice pentru fiecare întrebare); *wrongAnswer* apare în 164 triplete; *corectAnswer* apare în 194 triplete (descrie răspunsuri pentru întrebări cu răspunsuri multiple); *numberOfWords* apare în 21 de triplete (numărul exact de întrebări cu răspuns scurt).

Pentru a confirma generarea corectă a testelor folosind metoda descrisă în secțiunea anterioară, am implementat algoritmul folosind limbajul de programare PHP. Pentru a finaliza această sarcină am folosit biblioteca *phpSesame* pentru a interacționa cu RDF Sesame [144]. Folosind această bibliotecă, am generat statisticile prezentate mai sus și am rulat interogările SPARQL care au adunat elementele de testare. Pentru a mă asigura că elementele de testare sunt distribuite uniform în procesul de selecție (în acest fel selecția aleatoare a elementelor de testare este corectă) și fiecare test nu conține întrebări relaționate, am setat sistemul să producă 7000 de teste pentru a fi evaluate. Pentru aceasta, am ales să generez teste de 18 întrebări, cu două sau trei elemente adevărat-fals, unu la trei elemente cu răspuns scurt și restul elemente cu alegere multiplă. Întrebările au fost selectate din toate cele zece secțiuni ale cursului.

Pentru a produce toate testele cu cantitatea corectă de întrebări, sistemul trebuie să valideze parametrii stabiliți pentru respectivul test. De exemplu, numărul maxim de întrebări cu răspuns scurt nu poate fi mai mare decât grupurile de întrebări legate de acest tip. Dacă acest lucru nu este satisfăcut, sistemul poate produce teste cu întrebări mai puține.

Distribuția elementelor după tip, pentru cele 7000 de teste generate, este prezentată în figura 30, unde *tf* reprezintă elementele de testare adevărat-fals, *sa* pe cele cu răspuns scurt, iar *mc* pe cele cu răspunsuri multiple. De exemplu, *tf 3* reprezintă numărul de teste în care au fost selectate trei elemente adevărat-fals. Se observă ușor că elementele cu răspuns scurt și adevărat-fals sunt aproape perfect împărțite între numărul de posibilități (trei pentru răspuns scurt și doi pentru adevărat-fals). Numărul de elemente cu alegere multiplă se obține prin eliminarea celorlalte două tipuri de întrebări. Am obținut o distribuție normală, în care testele au șanse mai mari de a avea 13 sau 14, mai degrabă decât 12 sau 15, elemente de tip alegere multiplă.

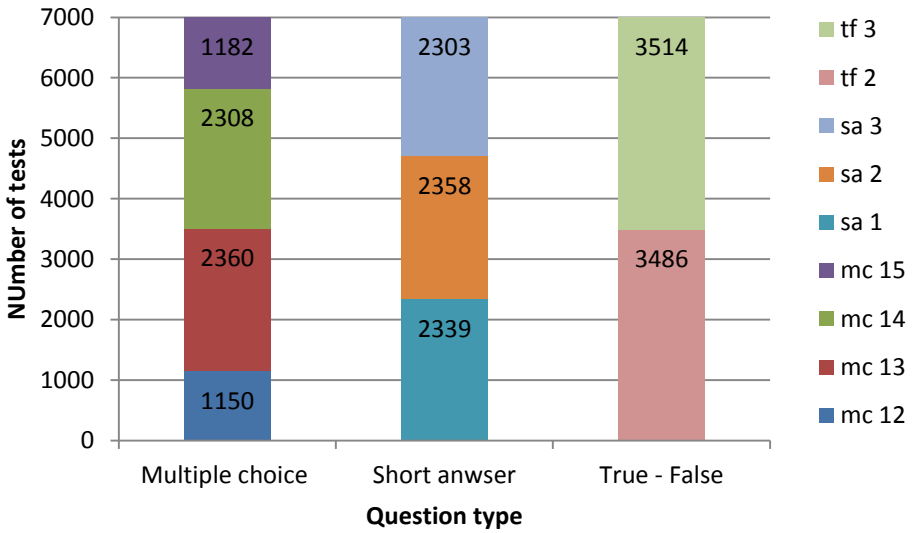


Figura 30. Distribuția elementelor în funcție de tip

Fiecare întrebare apare între 500 și 700 de teste, cu unele excepții. Numărul de apariții a unei întrebări în teste este prezentată în figura 31 de linia albastră,

corelat cu numărul de întrebări relaționate reprezentat de linia roșie. În figura 31, secțiunea 1, construită folosind datele pentru toate elementele de testare, putem vedea că numărul de apariții ale unui element de test scade atunci când numărul de întrebări relaționate crește. Cu toate acestea se observă unele vârfuri. Motivul pentru acest efect este reprezentat de numărul relativ mic de întrebări de tip adevărat-fals (30) și întrebări cu răspuns scurt (21). Acest lucru se observă în figura 31, secțiunea 2. Datele prezentate în aceasta figură sunt doar pentru elementele cu răspunsuri multiple; în acest caz, majoritatea vârfurilor au dispărut. Cele care rămân sunt pentru întrebările cu răspunsuri multiple, care sunt legate numai de întrebări de tip adevărat-fals sau cu răspuns scurt. În concluzie, distribuția elementelor de testare selectate din banca de întrebări garantează că fiecare element de testare este utilizat și numărul de apariții variază ușor în generarea de cantități mari de teste.

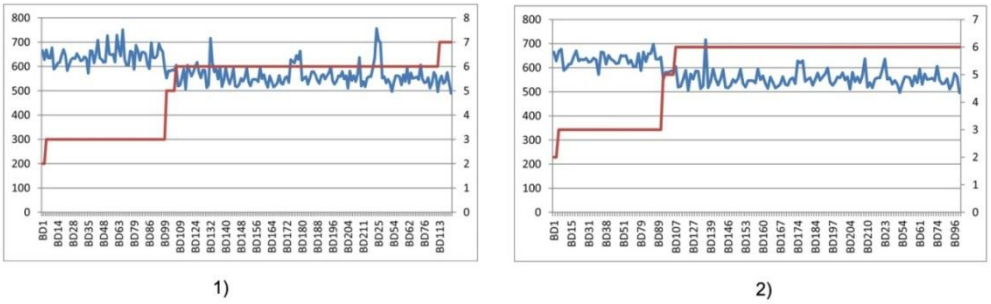


Figura 31. Distribuția tuturor elementelor (1.) și distribuția elementelor cu alegere multiplă (2.)

Am evaluat fiecare test pentru a mă asigura că nu apar duplicate. În cele 7000 de teste generate nu am găsit un astfel de caz.

În acest subcapitol am descris un sistem de stocare a băncilor de întrebări și o modalitate de generare a testelor, folosind tehnologii ale Web-ului Semantic. Am folosit un vocabular minimal care ar putea fi ușor extins pentru a include și alte concepte necesare în diferite scenarii de testare. Am convertit banca de întrebări pentru cursul Sisteme de Gestiune a Datelor în format RDF. De asemenea, am introdus toate întrebările într-o bază de date Sesame RDFS și am construit o metodă de generare a testelor. Prin generarea a 7000 de teste, am validat că ontologia și metoda de extracție produc teste bine formate.

Sistemul poate fi folosit de către profesori pentru a minimiza cantitatea de timp necesară pentru a produce teste și de către studenți în scenarii de auto-evaluare. Pentru a asigura corectitudinea evaluării, se poate calcula și verifica dificultatea de ansamblu a fiecărui test, care trebuie să fie într-un interval specificat pentru acel examen.

Pentru a verifica dacă această metodă îmbunătățește distribuția statistică a notelor, testele generate ar putea fi formatate pentru evaluare on-line sau formate pentru imprimare. Am implementat un modul de imprimare care generează teste în format PDF. Am utilizat acest modul pentru o evaluare. Pentru a evalua corect dacă distribuția statistică a notelor poate fi îmbunătăţită, trebuie să colectez mai multe date, pe o perioadă mai lungă de timp.

Această metodă se poate integra în platforma de e-Learning a universității noastre, bazată pe Moodle. Această integrare presupune un proces în două etape: primul pas constă din convertirea băncii de întrebări din blocurile Moodle Feeback și Questionaire, prin utilizarea unei unelte de cartografiere a bazelor de date relaționale (D2R platforma [145]), iar al doilea ar fi construcția unui bloc Moodle, care generează teste din baza de date RDF și permite utilizatorului să interacționeze cu datele.

Unul din avantajele folosirii tehnologiilor Web-ului Semantic îl constituie posibilitatea de a partaja cu ușurință datele stocate în format RDF cu mai mulți agenți. O altă direcție de studiu o reprezintă utilizarea Linked Data pentru îmbunătățirea calității testelor generate.

O parte din aceste cercetări au fost acceptate spre publicare în lucrarea [146].

5.4 Contribuții și concluzii

În acest capitol am încercat să demonstrez utilitatea ontologiei propuse în capitolul 4 și a tehnologiilor Web-ului Semantic în îmbunătățirea uneltelor software utilizate în procesul educațional.

Principale contribuţii aduse în acest capitol vor fi descrise punctual în cele ce urmează:

Maparea bazei de date Moodle pentru republicare de date sub format RDF

În prima aplicație practică am realizat fișierul de mapare a bazei de date Moodle, folosind conceptele și proprietățile definite într-o ontologie; fișierul realizat poate fi utilizat în serverul D2R. Prin acest proces am permis accesul la informațiile păstrate în format RDF, în platforma Moodle. Agenți software ar putea utiliza aceste date pentru construcția de aplicații utile cum ar fi: sincronizarea cursurilor, sincronizarea utilizatorilor și înscrierea studenților în cursuri prin fuziunea datelor din mai multe surse (Moodle, baza de date de gestiune a studenţilor facultăţilor, etc.); partajarea datelor pentru aplicații conexe platformei educaționale; construirea de agenți care să îmbunătățească datele prin interogarea lor din mai multe surse (de ex: datele de contact din fișiere FOAF ar fi extinse); agenți software care să genereze agenda de contacte a unui utilizator, pe baza cunoștințelor pe care le are; etc.

Momentan prin fișierul de mapare implementat am permis accesul la aproximativ 3,9 milioane de triplete RDF. Fișierul de mapare se poate extinde pentru a cuprinde restul de concepte propuse în ontologie și prin utilizarea altor ontologii externe. Totuși, recomand limitarea mapării doar la datele strict necesare în aplicații, deoarece o mapare excesivă ar crește timpul de răspuns la interogări.

Proiectarea şi implementarea laboratorului virtual, asigurând partajarea datelor cu platforma educaţională pe baza unui model de interogări federativ pentru îmbogăţirea datelor

Am prezentat apoi modul de utilizare a datelor publicate printr-un proces de mapare, pentru îmbunătățirea calității serviciilor oferite de o aplicație de laborator virtual. Astfel, modelul de îmbunătățire ar permite un schimb de date între cele două sisteme (aplicația Lefti și platforma Moodle), care ar ușura munca tutorelui și ar îmbunătăți procesul educațional.

Propunerea unei metode de generare a testelor folosind o bancă de întrebări în format RDF

Sistemul de stocare a băncilor de întrebări și modalitatea de generare a testelor folosind tehnologii ale Web-ului Semantic pot fi utilizate de către profesori pentru a minimiza cantitatea de timp necesară pentru a produce teste, iar de către studenți în scenarii de auto-evaluare. Pentru a asigura corectitudinea evaluării, se poate calcula și verifica dificultatea de ansamblu a fiecărui test, care trebuie să fie într-un interval specificat pentru acel examen.

Dezvoltarea şi implementarea aplicaţiei software de generare de teste

Pentru validarea metodei, testele generate ar putea fi formatate pentru evaluare on-line sau pentru imprimare. Am implementat un modul de imprimare, care generează teste în format PDF și am utilizat acest modul într-o evaluare. Pentru a valida calitatea sistemului trebuie colectate date pe o perioadă mai lungă de timp, dar testele inițiale sunt încurajatoare.

Pasul următor ar fi integrarea acestei metode în platforma de e-Learning a universității noastre, bazată pe Moodle. Această integrare presupune un proces în două etape: primul pas constă din conversia băncilor de întrebări disponibile în baza de date Moodle, iar al doilea constă în construirea unui bloc Moodle, care generează teste din baza de date RDF și permite utilizatorului să interacționeze cu datele. O altă direcție care merită exploatată este utilizarea Linked Data pentru îmbunătățirea calității testelor generate.

În concluzie, am prezentat în acest capitol, trei modele de aplicații practice ale ontologiei educaționale propuse, validând în acest mod utilitatea ei.

6 Contribuții și concluzii

Teza de doctorat a fost elaborată pe parcursul activității de cercetare desfășurate în cadrul Centrului Multimedia, Facultatea de Electronică și Telecomunicații a Universității ”Politehnica” din Timișoara. În această lucrare am abordat un domeniu de interes, care prezintă o dezvoltare continuă în ultimii ani, e-Learning-ul (învățământul electronic), din perspectiva partajării informațiilor între diferite sisteme fără a pierde semnificația lor.

În cadrul activității mele de cercetare am fost implicat în utilizarea aplicațiilor educaționale în diferite proiecte (platforme educaționale: ViCaDiS, realizarea campusului virtual al Universității „Politehnica” din Timişoara; laborator virtual pentru cursul de Sisteme de Gestiune a Datelor). Pe parcursul acestor activități am observat limitările datorate partajării informațiilor între diferite aplicații educaționale. Ca o posibilă soluție, mi-au atras atenția noile tehnologii informaționale (tehnologiile Web-ului Semantic), care au ca scop păstrarea datelor nu numai pentru ”consum” uman, ci și pentru agenți software.

În prima parte a acestei lucrări, am introdus conceptele de bază din spatele acestor tehnologii, prezentând componentele centrale conținute de un astfel de sistem. Urmărind modelul propus de inițiatorul acestei direcții, Tim Berners-Lee, am evaluat gradul de standardizare a straturilor conținute. Chiar dacă straturile de nivel superior sunt încă la stadiul de concept, am identificat o serie de aplicații web construite doar pe straturile inferioare, care oferă servicii și funcționalități noi sau îmbunătățite. Acest lucru denotă fiabilitatea tehnologiilor Web-ului Semantic, care mi-a permis testarea utilizării lor în mediul educațional.

Am studiat modelul RDF și formatele de serializare a datelor; ca și concluzie, Tutle este indicat pentru declararea tripletelor de către oameni (fiind mai ușor de utilizat), RDF/XML pentru comunicarea între aplicații (fiind singurul limbaj standardizat). Ca și sistem de stocare și interogare a datelor RDF, am recomandat utilizarea aplicațiilor în sursă deschisă, cele mai stabile soluții fiind Jena și Sesame. Pentru publicarea datelor cu păstrarea semnificației lor, pe baza studiilor realizate, am recomandat microformatele pentru structuri de date simple, iar pentru cele nestandardizate în microformate sau structuri complexe, RDFa.

Pentru a stabili ce date trebuie modelate, am studiat principalele concepte utilizate în platformele educaționale. De asemenea, am evaluat specificațiile de realizare a obiectelor educaționale, din punct de vedere al organizării datelor și declararea de meta-date. Specificația IEEE LOM este utilizată pe scară largă pentru definirea de meta-informații în obiecte educaționale. Fiind suficiente abordări pentru conversia acestora la RDF, nu am considerat necesară propunerea unui model suplimentar.

Pe baza structurilor de date identificate și a metodologiei de proiectare studiate, am propus un model de ontologie educațională care să permită republicarea datelor disponibile în platformă, în format RDF. După ce am proiectat specificațiile ontologiei, am formalizat conceptele și le-am implementat folosind limbajele RDFS și OWL. Am propus împărțirea ontologiei în subontologii, pentru a permite restricționarea accesului la date și o dezvoltare ulterioară mai facilă.

Evaluarea ontologiei am realizat-o prin implementarea a trei aplicații practice: publicarea datelor din baza de date Moodle în format RDF, realizând un fișier de mapare a conceptelor introduse în ontologie; îmbunătățirea laboratorului virtual prin partajarea de informații și interogări federative; generarea automată de teste dintr-o bancă de întrebări în format RDF.

Rezultatele cercetării cuprinse în prezenta teză se pot sintetiza ca şi răspunsuri la întrebările de cercetare stipulate în capitolul introductiv al prezentei lucrări:

1. *Care este nivelul actual de standardizare a tehnologiilor Web-ului Semantic? Cum pot fi utilizate în realizarea de aplicații web îmbunătățite?*
   Analiza extinsă, realizată în capitolul 1, relevă gradul mediu de standardizare al tehnologiilor SW. Cu toate acestea, am identificat o serie de aplicaţii practice care utilizează aceste tehnologii, uneltele necesare construcţiei lor şi avantajele pe care acestea le aduc.

2. *Ce tipuri de date sunt păstrate în platformele educaționale? Se pot reutiliza specificațiile de inserare de meta-date în interiorul obiectelor educaționale?*
   În capitolul al doilea am identificat aceste tipuri şi le-am organizat în patru structuri importante. De asemeana, am identificat şi evaluat specificaţiile existente, care permit introducerea de metadata în obiecte educaţionale. În concluzie, meta-informaţiile introduse de acestea se pot reutiliza, dar nu sunt suficiente pentru modelarea completă a unui sistem educaţional din punct de vedere semantic.

3. *Cum se pot insera meta-date direct în structura existentă a LMS-urilor?*
   Studiul asupra tehnologiilor SW au reliefat modurile de inserare de meta-date în structura unui LMS. Acestea pot fi microformate sau date în format RDF. Prima metodă este utilă pentru date care au o structură simplă. A doua metodă permite o expresivitate mare, dar necesită utilizarea de modele ontologice. Pentru a demonstra utilitatea acestor meta-informaţii am propus în subcapitolul 3.3 o metodă de republicare a datelor unei persoane în LMS-ul Moodle, folosind microformate şi coduri QR.

4. *Ce concepte sunt necesare în modelarea datelor educaționale și cum pot fi ele formalizate?*
   Pe baza studiilor asupra arhitecturii platformelor educaţionale şi a metodologiilor de proiectare a modelelor ontologice, am identificat şi formalizat principalele structuri de date din platformele educaţionale. Mai multe detalii se găsesc în capitolele 3.1 şi 4.

5. *În ce măsură se pot utiliza aceste modele pentru a crește calitatea serviciilor educaționale?*
   Odată stabilit modelul ontologic, am realizat trei experimente pentru a demonstra utilitatea acestuia. Pentru republicarea datelor şi pentru oferirea acestora spre partajare, am realizat un fişier de mapare care permite în primă fază accesul la aproximativ 4 milioane de triplete RDF. Aceste triplete se pot utiliza de agenţi software educaţionali pentru a produce servicii superioare (interogări federative, statistică, partajare facilă de date). Un astfel de agent software este prezentat în subcapitolul 5.2. Acesta oferă servicii educaţionale superioare prin partajarea de date între platforma educaţională şi o aplicaţie de laborator virtual. Al treilea exemplu prezintă o metodă de generare de teste folosind ca şi bancă de întrebări date modelate semantic.

Consider modelul de ontologie propus și aplicațiile practice prezentate drept un pas util spre integrarea tehnologiilor Web-ului Semantic în platformele educaționale. În continuare, voi sintetiza contribuțiile proprii, structurate în teoretice și aplicative.

6.1 Contribuţii teoretice

(1) Studiu asupra gradului de standardizare și nivelul de maturitate al tehnologiilor semantic web

Pentru a demonstra fiabilitatea noilor tehnologii informaționale, am identificat stadiul actual de standardizare a fiecărui strat din stiva Web-ului Semantic propusă de Tim Berners-Lee. Chiar dacă straturile superioare sunt momentan doar la stadiul de concept, am argumentat posibilitatea utilizării acestor tehnologii în platformele educaționale, prin identificarea de aplicații care pun în practică paradigmele Web-ului Semantic (subcapitolul 2.2). Am identificat limbajele necesare pentru modelarea datelor (formate de serializare – subcapitolul 2.2.1, limbaje de definire a modelelor – subcapitolul 2.2.2), sisteme de stocare și interogare (subcapitolul 2.2.3), metode de publicare de cunoștințe (subcapitolul 2.2.4).

(2) Analiza și sintetizarea principalelor tipuri de date administrate într-un LMS

Am studiat principalele concepte utilizate în cadrul platformelor de gestionare a conținutului educațional, identificând structura datelor administrate. Am grupat aceste tipuri de date în secțiuni, pentru a permite organizarea conceptelor care vor fi modelate în ontologia propusă. Această analiză, împreună cu concluziile deduse, pot fi găsite în subcapitolul 3.1.1.

(3) Analiza specificațiilor utilizate în introducerea de meta-date în obiecte educaționale

Am analizat două dintre cele mai utilizate specificații pentru realizarea de pachete de resurse educaționale, SCORM și IMS CC. Am observat că pentru inserarea meta-informațiilor, amândouă converg spre utilizarea unui standard comun IEEE LOM, chiar dacă IMS folosește un profil mai restrictiv. Specificațiile IEEE

LOM au variante de definire a termenilor folosind limbaje specifice Web-ului Semantic. Datorită acestui fapt și a gradului mare de utilizare, nu consider necesară o reproiectare a structurii meta-datelor din interiorul obiectelor educaționale SCORM sau IMS, aceste informații fiind ușor de extras în format RDF. Toate aceste considerente pot fi găsite în cadrul subcapitolului 3.1.2.

(4) Studiul metodologiilor de proiectare a unei ontologii semantic web

Pentru proiectarea corectă a ontologiei educaționale a trebuit să efectuez un studiu asupra metodologiilor specializate pentru acest proces. Am identificat principalele metodologii, comparându-le din punct de vedere al reutilizării și reproiectării de vocabulare / ontologii standardizate. Am formalizat pașii care trebuie parcurși în procesul de proiectare, iar ca metodologie am propus utilizarea soluției NeOn, datorită flexibilității ridicate. Descrierea metodologiilor și comparația între ele este descrisă în subcapitolul 4.1.2.

(5) Stabilirea specificaților de proiectare a unei ontologii educaționale

Am evaluat cantitatea de date păstrată în baza de date Moodle și am propus spre modelare doar informațiile necesare în anumite scenarii educaționale. Pentru a permite accesul aplicațiilor doar la o parte din informații, am propus împărțirea în subontologii. Am identificat limitările conceptelor care trebuie modelate în ontologie, stabilind în același timp gradul de expresivitate al limbajului de modelare, necesar pentru implementarea acestora. Descrierea pe larg a evaluării datelor disponibile în platforma Moodle, a scenariilor utilizate în identificarea datelor propuse spre modelare și lista specificațiilor se găsesc în subcapitolul 4.2.1.

(6) Descrierea modelului de interogări federative pentru obținerea de informații îmbogățite

În subcapitolul 5.2.2, am propus o metodă de a obține date îmbogățite folosind interogări federative între mai multe surse de date RDF. Pentru alinierea grafurilor RDF am propus spre utilizare nodurile care descriu adresele de email ale utilizatorilor celor două aplicații din care se obțin datele. Această abordare permite implementarea de scenarii educaționale care pot îmbunătăți procesul pedagogic.

(7) Propunerea metodei pentru generarea automată de teste dintr-o bancă de întrebări în format RDF

Pentru minimizarea cantității de timp necesare unui tutore pentru a produce teste de evaluare am identificat etapele din acest proces. În proiectarea elementelor testelor nu pot interveni, deoarece profesorul este cel mai în măsură să realizeze acest proces, dar pot asigura că procesul de revizuire, de testare a elementelor, de generare și evaluare a testelor se realizează într-un sistem automatizat. Pentru realizarea acestor deziderate am propus modelarea elementelor testelor în format RDF și am descris procesul de generare a testelor. Descrierea pe larg a acestei metode se găsește în subcapitolul 5.3.

6.2 Contribuții aplicative

(1) Proiectarea și implementarea unei metode de introducere de microformate în cadrul LMS-ului Moodle

Din analiza funcționării platformei Moodle am constat că procesul de salvare a datelor de contact într-o agendă electronică este laborios și poate duce la pierdere de informații.

Astfel, am realizat un modul destinat LMS-ului Moodle care să permită publicarea informațiilor de contact ale utilizatorilor în format hcard, ușor de citit de agenți software care extrag sau indexează datele. De asemenea, am propus utilizarea codurilor QR pentru a facilita importul datelor de contact direct în agenda telefonului mobil. Se îmbunătățesc astfel serviciile oferite într-un LMS, utilizând microformate pentru publicarea de structuri mici de date.

Această soluție este detaliată în subcapitolul 3.2.2.

(2) Formalizarea și implementarea conceptelor ontologiei educaționale, utilizând limbajul RDFS

Pentru implementarea de facilități mai complexe a fost necesară proiectarea unei ontologii care să modeleze datele, microformatele nefiind suficient de expresive.

Urmărind metodologia stabilită prin analiza efectuată și conform specificațiilor trasate, am urmat un proces de conceptualizare și formalizare a termenilor definiți în ontologie, încercând să reutilizez pe cât posibil vocabulare sau ontologii standardizate. Extinzând ontologii populare (FOAF, DC, SIOC) am asigurat o aliniere ușoară cu grafurile de date externe. Folosind limbajul RDFS am implementat conceptele introduse.

Descrierea pe larg a acestui proces, precum și considerentele legate de utilizare, se găsesc în subcapitolul 4.2.

(3) Proiectarea și implementarea fișierului de mapare a bazei de date Moodle pentru republicare date în format RDF

Pentru a putea construi aplicații semantice, datele trebuie să fie disponibile în format RDF. Am realizat acest pas pe baza ontologiei propuse în capitolul 5. Am ales să utilizez o soluție de mapare a bazei de date Moodle la conceptele din ontologie, permițând astfel accesul direct la informații, fără a dubla spațiul ocupat de datele și fără a elimina toate neajunsurile datorate sincronizării. În acest fel, am permis accesul la aproximativ 4 milioane de triplete. Recomand limitarea mapării strict la conceptele necesare în aplicație, din motive de siguranță și de performanță.

Descrierea detaliată a fișierului de mapare se găsește în subcapitolul 5.1.

(4) Proiectarea și implementarea unui laborator virtual, având date partajate cu LMS-ul în format RDF

Prin evaluarea aplicației de tip laborator virtual utilizată în cadrul activităților legate de cursuri de programare, am constatat necesitatea schimbului de informații între această aplicație și platforma educațională utilizată pentru a oferi suport de curs (Moodle).

Pentru rezolvarea acestei deficiențe, am reproiectat aplicația astfel încât să conțină module care să asigure partajarea informației în format RDF și construcția de interogări federative, care să pună în valoare aceste date.

Aceste soluții sunt prezentate pe larg în subcapitolul 5.2.

(5) Dezvoltarea unei soluții de generare automată de teste, în care întrebările sunt modelate folosind limbajul RDF

În activitatea mea didactică am observat cât timp este necesar pentru stabilirea testelor de evaluare. De asemenea, prin analiza rezultatelor studenților la testele de evaluare, am constat variația mare a curbelor de distribuție de la o prezentare la alta și față de distribuția standard.

Ca și soluție, am ales să realizez un sistem de generare automată de teste, în care elementele de test să fie în format RDF. Pentru a asigura corectitudinea evaluării, se poate calcula și valida gradul de dificultate, de ansamblu, al testului. Am implementat un modul de generare de teste în format PDF. Am evaluat procesul de generare pe un eșantion de 7000 de teste.

Descrierea detaliată a aplicației propuse se găsește în subcapitolul 5.3.

6.3 Direcţii de cercetare viitoare

Modelul ontologic propus se poate extinde pentru a cuprinde și alte concepte necesare în anumite scenarii educaționale. Odată cu implementarea specificațiilor OWL în motoarele de stocare, care să susțină capabilitățile de inferențiere propuse de acest standard, se pot adăuga în ontologie constrângeri suplimentare care vor permite deduceri de informații noi.

Datele publicate prin fișierul de mapare se pot extinde folosind și restul de concepte. Trebuie studiată posibilitatea restricționării accesului la anumite secțiuni de informații, pentru a putea partaja anumite date fără caracter personal în sursă deschisă. În aplicația de laborator virtual trebuie proiectată și implementată o metodă de sincronizare a datelor în format RDF cu cele din baza de date relaționale.

Pentru validarea îmbunătățirii curbei de distribuție a notelor, sistemul de generare de întrebări trebuie testat pe un număr mare de studenți. De asemenea, trebuie implementată o interfață de administrare a băncii de întrebări.

În plus față de scenariile de utilizare prezentate în exemple, se pot identifica alte situații în care acest tip de abordare ar fi benefică. Una din posibile aplicații care m-ar interesa în mod deosebit ar fi sincronizarea conturilor de utilizator, curriculum și înscrierile în cursuri, între baza de date a platformei educaționale și cea a secretariatelor facultăților care se ocupă de gestiunea studenților. Implementarea acestei sincronizări ar minimiza munca necesară pentru administrarea platformei la fiecare început de an.

O altă direcție de cercetare ar fi learning analytics. Acest domeniu presupune analiza datelor statistice cu privire la interacțiunea cursanților cu resursele educaționale, unelte de comunicare sau orice alt mediu / unealtă din cadrul procesului educațional [147][148]. Aceste metode de analiză se pot implementa pe date colectate în format RDF, permițând astfel punerea în valoare a interogărilor federative.

Bibliografie

[1] S. Naidu, *E-learning: A guidebook of principles, procedures and practices*. Commonwealth Educational Media Centre for Asia (CEMCA), 2003.

[2] N. Sclater, "Large-scale open source e-learning systems at the Open University UK," *Educause Centre for Applied Research, Research Bulleting*, vol. 2008, no. 12, 2008.

[3] M. Nichols, "The financial benefits of eLearning," *Journal of Open, Flexible and Distance Learning*, vol. 8, no. 1, pp. 25–33, 2012.

[4] T. B. Lee, J. Hendler, O. Lassila, and others, "The semantic web," *Scientific American*, vol. 284, no. 5, pp. 34–43, 2001.

[5] S. Decker, F. Van Harmelen, J. Broekstra, M. Erdmann, D. Fensel, I. Horrocks, M. Klein, and S. Melnik, "The Semantic Web-on the respective Roles of XML and RDF," *IEEE Internet Computing*, vol. 4, no. 5, pp. 1–19, 2000.

[6] C. Bizer and A. Seaborne, "D2RQ-treating non-RDF databases as virtual RDF graphs," in *Proceedings of the 3rd International Semantic Web Conference (ISWC2004)*, 2004, p. 26.

[7] J. Broekstra and A. Kampman, "RDF (S) manipulation, storage and querying using Sesame," in *Demo Proc. of the 3rd Intl. Semantic Web. Conf*, 2004.

[8] R. R. Amorim, M. Lama, E. Sánchez, A. Riera, and X. A. Vila, "A learning design ontology based on the IMS specification," *JOURNAL OF EDUCATIONAL TECHNOLOGYAND SOCIETY*, vol. 9, no. 1, p. 38, 2006.

[9] M. Poveda-Villalón, "A reuse-based lightweight method for developing linked data ontologies and vocabularies," *The Semantic Web: Research and Applications*, pp. 833–837, 2012.

[10] L. Aroyo and D. Dicheva, "The new challenges for e-learning: The educational semantic web," *Educational Technology & Society*, vol. 7, no. 4, pp. 59–69, 2004.

[11] Academia Română, "Dicționarul explicativ al limbii române, ediția a II-a." Editura Univers Enciclopedic, 1998.

[12] J. Hebeler, M. Fisher, R. Blace, and A. Perez-Lopez, *Semantic Web Programming*, 1st ed. Wiley, 2009.

[13] T. Segaran, C. Evans, and J. Taylor, *Programming the Semantic Web*, 1st ed. O'Reilly Media, 2009.

[14] D. Allemang and J. Hendler, *Semantic Web for the Working Ontologist: Effective Modeling in RDFS and OWL*, 1st ed. Morgan Kaufmann, 2008.

[15] T. J. Teorey, S. S. Lightstone, T. Nadeau, and H. V. Jagadish, *Database Modeling and Design, Fifth Edition: Logical Design*, 5th ed. Morgan Kaufmann, 2011.

[16] "RDF Primer." [Online]. Available: http://www.w3.org/TR/rdf-primer/. [Accessed: 02-Jul-2012].

[17] N. Shadbolt, W. Hall, and T. Berners-Lee, "The Semantic Web Revisited," *IEEE Intelligent Systems*, vol. 21, no. 3, pp. 96–101, Feb. 2006.

[18] T. Berners-Lee, "Semantic Web - XML2000 - slide 'Architecture'."[Online]. Available: http://www.w3.org/2000/Talks/1206-xml2k-tbl/slide10-0.html. [Accessed: 26-Jul-2012].

[19] "Resource Description Framework (RDF) Model and Syntax Specification," 1999. [Online]. Available: http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/. [Accessed: 30-Jul-2012].
[20] "RDF/XML Syntax Specification (Revised)." [Online]. Available: http://www.w3.org/TR/rdf-syntax-grammar/. [Accessed: 30-Jul-2012].
[21] "RDF Test Cases." [Online]. Available: http://www.w3.org/TR/rdf-testcases/#ntriples. [Accessed: 30-Jul-2012].
[22] D. Beckett and T. Berners-Lee, "Turtle - Terse RDF Triple Language." [Online]. Available: http://www.w3.org/TeamSubmission/turtle/. [Accessed: 30-Jul-2012].
[23] "RDF Vocabulary Description Language 1.0: RDF Schema." [Online]. Available: http://www.w3.org/TR/rdf-schema/. [Accessed: 02-Jul-2012].
[24] "OWL 2 Web Ontology Language Document Overview." [Online]. Available: http://www.w3.org/TR/owl2-overview/. [Accessed: 02-Jul-2012].
[25] "SPARQL Query Language for RDF." [Online]. Available: http://www.w3.org/TR/rdf-sparql-query/. [Accessed: 01-Aug-2012].
[26] "SPARQL 1.1 Query Language." [Online]. Available: http://www.w3.org/TR/2012/WD-sparql11-query-20120724/. [Accessed: 01-Aug-2012].
[27] J. Bailey, F. Bry, T. Furche, and S. Schaffert, "Web and semantic web query languages: A survey," *Reasoning Web*, pp. 95–95, 2005.
[28] A. Seaborne, G. Manjunath, C. Bizer, J. Breslin, S. Das, I. Davis, S. Harris, K. Idehen, O. Corby, K. Kjernsmo, and others, "SPARQL/Update: A language for updating RDF graphs," *W3C Member Submission*, vol. 15, 2008.
[29] M. Arenas and J. Pérez, "Federation and Navigation in SPARQL 1.1," *Reasoning Web. Semantic Technologies for Advanced Query Answering*, pp. 78–111, 2012.
[30] A. Polleres, "SPARQL1. 1: New features and friends (OWL2, RIF)," *Web Reasoning and Rule Systems*, pp. 23–26, 2010.
[31] "Colecţie de unelte semanticweb.org." [Online]. Available: http://semanticweb.org/wiki/Tools. [Accessed: 01-Aug-2012].
[32] C. Bizer and D. Westphal, "Developers Guide to Semantic Web Toolkits for different Programming Languages," 2007. [Online]. Available: http://www4.wiwiss.fu-berlin.de/bizer/toolkits/. [Accessed: 01-Aug-2012].
[33] B. McBride, "Jena: A semantic web toolkit," *Internet Computing, IEEE*, vol. 6, no. 6, pp. 55–59, 2002.
[34] B. Nowack, "ARC2." [Online]. Available: https://github.com/semsol/arc2/wiki. [Accessed: 01-Aug-2012].
[35] C. Schönberg and B. Freitag, "Evaluating RDF querying frameworks for document metadata," Technical Report MIP-0903, Univ. of Passau, 2009.
[36] C. Bizer and A. Schultz, "The berlin sparql benchmark," *International Journal on Semantic Web and Information Systems (IJSWIS)*, vol. 5, no. 2, pp. 1–24, 2009.
[37] T. Steiner, R. Troncy, and M. Hausenblas, "How Google is using Linked Data Today and Vision For Tomorrow," *Future Internet Assembly, Ghent, Belgium*, 2010.
[38] P. Karanth and K. Mahesh, "Integrating Knowledge Base Retrieval with Web Search using Semantic Roles," *Proceedings of the International MultiConference of Engineers and Computer Scientists*, vol. 1, 2012.
[39] "Despre Microformate, Pagină oficială." [Online]. Available: http://microformats.org/about. [Accessed: 02-Aug-2012].

[40] R. Khare and T. Çelik, "Microformats: a pragmatic path to the semantic web," in *Proceedings of the 15th international conference on World Wide Web*, 2006, pp. 865–866.
[41] A. Ben, "hGRDDL: Bridging microformats and RDFa," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 6, no. 1, pp. 54–60, Feb. 2008.
[42] "RDFa 1.1 Primer." [Online]. Available: http://www.w3.org/TR/xhtml-rdfa-primer/. [Accessed: 02-Aug-2012].
[43] A. Graf, "RDFa vs. Microformats," *Digital Enterprise Research Institute, Innsbruck*, 2007.
[44] V. Tomberg and M. Laanpere, "RDFa versus Microformats: Exploring the Potential for Semantic Interoperability of Mash-up Personal Learning Environments," in *nd International Workshop on Mashup Personal Learning Environments (MUPPLE). Ed. by Fridolin Wild, Marco Kalz, Matthias Palmér, and Daniel Müller. CEUR Workshop Proceedings. Sept..: http://CEUR-WS. org*, 2009, vol. 506.
[45] C. Bizer, T. Heath, K. Idehen, and T. Berners-Lee, "Linked data on the web (LDOW2008)," in *Proceeding of the 17th international conference on World Wide Web*, 2008, pp. 1265–1266.
[46] S. Naidu, "E–Learning A Guidebook of Principles," *Procedures and Practices. 2nd Revised Edition. New Delhi: The Commonwealth Educational Media Center for Asia*, 2006.
[47] R. Mason and F. Rennie, *Elearning: The key concepts*. Taylor & Francis, 2006.
[48] N. Sawhney, "E-learning: global education without walls," *Educational Quest-An International Journal of Education and Applied Social Sciences*, vol. 3, no. 1, pp. 35–41, 2012.
[49] D. R. Garrison and H. Kanuka, "Blended learning: Uncovering its transformative potential in higher education," *The internet and higher education*, vol. 7, no. 2, pp. 95–105, 2004.
[50] C. R. Graham, "Blended learning systems," *Handbook of blended learning: Global Perspectives, local designs. Pfeiffer Publishing, San Francisco, http://www. publicationshare. com/graham_intro. pdf*, 2006.
[51] S. E. Lakhan and K. Jhunjhunwala, "Open source software in education," *Educause Quarterly*, vol. 31, no. 2, p. 32, 2008.
[52] A. Ternauciuc, "Contribuții la dezvoltarea uneltelor de comunicare în cadrul platformelor web educaționale," Universitatea "Politehnica" Timisoara, 2011.
[53] D. Bremer and R. Bryant, "A Comparison of two learning management Systems: Moodle vs Blackboard," in *Proceedings of the 18th Annual Conference of the National Advisory Committee on Computing Qualifications. NACCQ, New Zealand. Retrieved February*, 2005, vol. 21, p. 2008.
[54] "Instanțe declarate de Moodle." [Online]. Available: http://moodle.org/stats. [Accessed: 20-Sep-2012].
[55] D. Andone, R. Vasiu, A. Ternauciuc, and B. Dragulescu, "The use of social media tools in ViCaDiS Virtual Campus," in *Computational Cybernetics and Technical Informatics (ICCC-CONTI), 2010 International Joint Conference on*, 2010, pp. 305–310.
[56] C. Mary, *Moodle 2.0 First Look*. Packt Publishing, 2010.
[57] G. Totkov, C. Krusteva, and N. Baltadzhiev, "About the standardization and the interoperability of e-learning resources," in *Proceedings of the 5th*

international conference on Computer systems and technologies, 2004, pp. 1–6.
[58] V. Gonzalez-Barbone and L. Anido-Rifon, "Creating the first SCORM object," *Computers & Education*, vol. 51, no. 4, pp. 1634–1647, 2008.
[59] "SCORM Users Guide for Programmers." ADLNet.gov, 2011.
[60] P. Jesukiewicz, "SCORM® 2004 4th Edition Content Aggregation Model (CAM) Version 1.1." ADLNet.gov, 2009.
[61] "IMS GLC Common Cartridge Profile: Implementation." 2011.
[62] H. S. Al-Khalifa and H. C. Davis, "The evolution of metadata from standards to semantics in E-learning applications," in *Proceedings of the seventeenth conference on Hypertext and hypermedia*, 2006, pp. 69–72.
[63] M. Nilsson, M. Palmér, and J. Brase, "The LOM RDF binding: principles and implementation," in *Proceedings of the Third Annual ARIADNE conference, Leuven Belgium, 2003*, 2003.
[64] "Grupul de lucru DCMI / IEEE LTSC." [Online]. Available: http://dublincore.org/educationwiki/DCMIIEEELTSCTaskforce. [Accessed: 25-Sep-2012].
[65] C. Brooks and G. McCalla, "Towards flexible learning object metadata," *International Journal of Continuing Engineering Education and Life Long Learning*, vol. 16, no. 1, pp. 50–63, 2006.
[66] W. Nejdl, B. Wolf, C. Qu, S. Decker, M. Sintek, A. Naeve, M. Nilsson, M. Palmér, and T. Risch, "EDUTELLA: a P2P networking infrastructure based on RDF," in *Proceedings of the 11th international conference on World Wide Web*, 2002, pp. 604–615.
[67] H. Q. Yu, S. Dietze, N. Li, C. Pedrinaci, D. Taibi, N. Dovrolls, T. Stefanut, E. Kaldoudi, and J. Domingue, "A linked data-driven & service-oriented architecture for sharing educational resources," 2011.
[68] R. Balog-Crisan and I. Roxin, "Semantic Learning Content Management System," *Proceedings of e-Learning vol. II*, pp. 85–89, 2008.
[69] M. Onita, "Contributii la utilizarea noilor tehnologii informationale in invatamantul electronic," Universitatea "Politehnica" Timisoara, 2011.
[70] C. Tatek, "microformats.org at 5: Two Billion Pages With hCards, 94% of Rich Snippets," 2010. [Online]. Available: http://microformats.org/2010/07/08/microformats-org-at-5-hcards-rich-snippets. [Accessed: 06-Aug-2012].
[71] Inc. Gartner, "Gartner Says Worldwide Mobile Device Sales to End Users Reached 1.6 Billion Units in 2010; Smartphone Sales Grew 72 Percent in 2010," 2011. [Online]. Available: http://www.gartner.com/it/page.jsp?id=1543014. [Accessed: 06-Aug-2012].
[72] N. Bevan, "Measuring usability as quality of use," *Software Quality Journal*, vol. 4, no. 2, pp. 115–130, 1995.
[73] I. Ermalai and B. Dragulescu, "The usefulness and functionality of microformats in a particular eLearning system," in *Computational Cybernetics and Technical Informatics (ICCC-CONTI), 2010 International Joint Conference on*, 2010, pp. 387–390.
[74] E. P. Lewis, *Microformats made simple*. New Riders Pub, 2009.
[75] Internet Mail Consortium, "vCard Overview." [Online]. Available: http://www.imc.org/pdi/vcardoverview.html. [Accessed: 06-Aug-2012].
[76] Microformats.org, "Microformats Implementations." [Online]. Available: http://microformats.org/wiki/implementations. [Accessed: 06-Aug-2012].

[77] Microformats.org, "Browsers." [Online]. Available: http://microformats.org/wiki/browsers. [Accessed: 06-Aug-2012].
[78] T. Seideman, "Barcodes Sweep the World," 1998. [Online]. Available: http://www.barcoding.com/information/barcode_history.shtml. [Accessed: 07-Aug-2012].
[79] M. Ebling and R. Cáceres, "Bar Codes Everywhere You Look," *Pervasive Computing, IEEE*, vol. 9, no. 2, pp. 4–5, 2010.
[80] T. Y. Liu, T. H. Tan, and Y. L. Chu, "QR Code and Augmented Reality-Supported Mobile English Learning System," *Mobile Multimedia Processing*, pp. 37–52, 2010.
[81] T. Y. Liu, T. H. Tan, and Y. L. Chu, "2D barcode and augmented reality supported english learning system," in *Computer and Information Science, 2007. ICIS 2007. 6th IEEE/ACIS International Conference on*, 2007, pp. 5–10.
[82] B. Dragulescu, I. Ermalai, M. Bucos, and M. Mocofan, "Using hCard and vCard for improving usability in Moodle," in *Applied Computational Intelligence and Informatics (SACI), 2011 6th IEEE International Symposium on*, 2011, pp. 473–476.
[83] B. Dragulescu, I. Ermalai, M. Bucos, and R. Vasiu, "Metadata Methods for Improving Usability in Moodle," *International Journal of Web Engineering*, pp. 6–10, 2012.
[84] A. Rawlings, P. van Rosmalen, R. Koper, M. Rodríguez-Artacho, and P. Lefrere, "Survey of Educational Modelling Languages (EMLs) Version 1," 2002.
[85] M. H. Kadivar and C. S. Lee, "Evaluating an ontology based E-learning data model using the TAM model," in *Proceedings of 18th International Conference on Computers in Education*, 2010.
[86] V. Gonzalez-Barbone and L. Anido-Rifon, "From SCORM to Common Cartridge: A step forward," *Computers & Education*, vol. 54, no. 1, pp. 88–102, 2010.
[87] A. Hopkinson, "International standards for global information.," 2004.
[88] R. Wilson, "The role of ontologies in teaching and learning," *TechWatch Reports*, 2004.
[89] A. Maedche and S. Staab, "Ontology learning for the semantic web," *Intelligent Systems, IEEE*, vol. 16, no. 2, pp. 72–79, 2001.
[90] L. Crocker and J. Algina, *Introduction to classical and modern test theory.* Holt, Rinehart and Winston, 6277 Sea Harbor Drive, Orlando, FL 32887, 1986.
[91] M. Sreenivasan, "Ontology Engineering," COCHIN UNIVERSITY OF SCIENCE AND TECHNOLOGY, 2010.
[92] E. Simperl, "Reusing ontologies on the Semantic Web: A feasibility study," *Data & Knowledge Engineering*, vol. 68, no. 10, pp. 905–925, 2009.
[93] M. C. Suárez-Figueroa, R. García-Castro, B. Villazón-Terrazas, and A. Gómez-Pérez, "Essentials In Ontology Engineering: Methodologies, Languages, And Tools," 2011.
[94] A. Gómez-Pérez, O. Corcho, and M. Fernández-López, *Ontological Engineering*. Springer-Verlag, London, Berlin, 2002.
[95] S. Staab, R. Studer, H. P. Schnurr, and Y. Sure, "Knowledge processes and ontologies," *Intelligent Systems, IEEE*, vol. 16, no. 1, pp. 26–34, 2001.

[96] H. S. Pinto, C. Tempich, and S. Staab, "Ontology engineering and evolution in a distributed world using DILIGENT," *Handbook on Ontologies*, pp. 153–176, 2009.
[97] Y. Sure, S. Staab, and R. Studer, "Ontology engineering methodology," *Handbook on Ontologies*, pp. 135–152, 2009.
[98] M. C. Suárez-Figueroa, "NeOn methodology for building ontology networks: Specification, scheduling and reuse," Informatica, 2010.
[99] E. Blomqvist, V. Presutti, E. Daga, and A. Gangemi, "Experimenting with extreme design," *Knowledge Engineering and Management by the Masses*, pp. 120–134, 2010.
[100] M. Jarrar and R. Meersman, "Ontology engineering–the DOGMA approach," *Advances in Web Semantics I*, pp. 7–34, 2009.
[101] D. Gašević, N. Kaviani, and M. Milanović, "Ontologies and software engineering," *Handbook on Ontologies*, pp. 593–615, 2009.
[102] C. Roussey, F. Pinet, M. A. Kang, and O. Corcho, "An Introduction to Ontologies and Ontology Engineering," *Ontologies in Urban Development Projects*, pp. 9–38, 2011.
[103] S. Lamparter and Y. Sure, "An interdisciplinary methodology for building service-oriented systems on the web," in *Services Computing, 2008. SCC'08. IEEE International Conference on*, 2008, vol. 2, pp. 475–478.
[104] A. Soares and F. Fonseca, "Building Ontologies for Information Systems: What we have, what we need," 2009.
[105] G. A. and F. van Harmelen, "A Semantic Web Primer, 2nd Edition," Mar. 2008.
[106] V. Devedzic, "Education and the semantic web," *International Journal of Artificial Intelligence in Education*, vol. 14, no. 2, pp. 165–191, 2004.
[107] V. Uren, S. Buckingham Shum, M. Bachler, and G. Li, "Sensemaking tools for understanding research literatures: Design, implementation and user evaluation," *International journal of human-computer studies*, vol. 64, no. 5, pp. 420–445, 2006.
[108] B. Barros, M. F. Verdejo, T. Read, and R. Mizoguchi, "Applications of a collaborative learning ontology," *MICAI 2002: Advances in Artificial Intelligence*, pp. 103–118, 2002.
[109] Z. Pilat, M. Słowikowski, and J. Zieliński, "Application of Modern E-Learning Techniques in the Vocational Training in Automation and Robotics," *Solid State Phenomena*, vol. 144, pp. 106–111, 2009.
[110] A. Ternauciuc, B. Dragulescu, M. Onita, and R. Vasiu, "SINGLE SIGN-ON SOLUTIONS FOR MOODLE," in *Conference proceedings of" eLearning and Software for Education,"* 2009, p. 217.
[111] M. Onita, A. Ternauciuc, B. Dragulescu, and I. Ermalai, "Streaming Solutions at UPT," in *Proceedings of the 5th International Scientific Conference ELSE'E-Learning and Software for Education,"" Universitatea Nationale de Aparare Carol I"" Publishing House, ISSN*, Roma, Italia, 2009, pp. 151–157.
[112] M. Onita, A. Ternauciuc, B. Dragulescu, and I. Ermalai, "Media Streaming in Higher Education," in *Proceedings of Iadis International Conference "Cognition and Exploratory Learning in Digital Age,"* 2009, pp. 373–377.
[113] "Schema bazei de date Moodle." [Online]. Available: http://docs.moodle.org/dev/images_dev/5/5a/Moodle2erd.png. [Accessed: 10-Sep-2012].
[114] N. F. Noy, D. L. McGuinness, and others, *Ontology development 101: A guide to creating your first ontology*. Stanford knowledge systems laboratory

technical report KSL-01-05 and Stanford medical informatics technical report SMI-2001-0880, 2001.

[115] J. Golbeck and M. Rothstein, "Linking social networks on the web with foaf: A semantic web case study," in *Proceedings of the 23rd national conference on Artificial intelligence*, 2008, vol. 2, pp. 1138–1143.

[116] D. Brickley and L. Miller, "FOAF vocabulary specification 0.91," Tech. rep. ILRT Bristol, Nov. 2007. ur l: http://xmlns. com/foaf/spec/20071002. html, 2000.

[117] DCMI Usage Board, "Recomandarea Dublin Core." [Online]. Available: http://dublincore.org/documents/2012/06/14/dcmi-terms/. [Accessed: 12-Sep-2012].

[118] S. Sánchez-Alonso, M. A. Sicilia, and M. Pareja, "Mapping IEEE LOM to WSML: an ontology of learning objects," *Proceedings of the ITA*, pp. 92–101, 2007.

[119] U. Bojars and J. G. Breslin, *SIOC Core Ontology Specification. Namespace document, DERI, NUI Galway*. 2009.

[120] J. Breslin, U. Bojars, A. Passant, S. Fernandez, and S. Decker, "SIOC: Content Exchange and Semantic Interoperability Between Social Networks," 2009.

[121] S. Goyal and R. Westenthaler, "RDF-Gravity (RDF Graph Visualization Tool)." [Online]. Available: http://semweb.salzburgresearch.at/apps/rdf-gravity/. [Accessed: 13-Sep-2012].

[122] M. Bucos, B. Dragulescu, and M. Veltan, "Designing a semantic web ontology for E-learning in higher education," in *2010 9th International Symposium on Electronics and Telecommunications (ISETC)*, 2010, pp. 415–418.

[123] "The Linking Open Data cloud diagram." [Online]. Available: http://richard.cyganiak.de/2007/10/lod/. [Accessed: 14-Sep-2012].

[124] C. A. Knoblock, P. Szekely, J. L. Ambite, S. Gupta, A. Goel, M. Muslea, K. Lerman, and P. Mallick, "Interactively mapping data sources into the semantic web," in *Proceedings of the Workshop on Linked Science (Submitted for review)*, 2011.

[125] M. S. Marshall, R. Boyce, H. F. Deus, J. Zhao, E. L. Willighagen, M. Samwald, E. Pichler, J. Hajagos, E. Prud'hommeaux, and S. Stephens, "Emerging practices for mapping and linking life sciences data using RDF—A case series," *Web Semantics: Science, Services and Agents on the World Wide Web*, 2012.

[126] C. de Laborda and S. Conrad, "Database to Semantic Web Mapping using RDF query languages," *Conceptual Modeling-ER 2006*, pp. 241–254, 2006.

[127] M. Šeleng, M. Laclavík, Z. Balogh, and L. Hluchỳ, "RDB2Onto: Approach for creating semantic metadata from relational database data," in *INFORMATICS 2007: proceedings of the ninth international conference on informatics. Bratislava, Slovak Society for Applied Cybernetics and Informatics*, 2007, pp. 113–116.

[128] K. N. Vavliakis, T. K. Grollios, and P. A. Mitkas, "Rdote-transforming relational databases into semantic web data," in *Proc. 9th Int. Semantic Web Conf*, 2010.

[129] N. Konstantinou, D. Spanos, M. Chalas, E. Solidakis, and N. Mitrou, "VisAVis: An approach to an intermediate layer between ontologies and relational database contents," in *International Workshop on Web Information Systems Modeling (WISM2006) Luxembourg*, 2006.

[130] H. Knublauch, M. Horridge, M. Musen, A. Rector, R. Stevens, N. Drummond, P. Lord, N. F. Noy, J. Seidenberg, and H. Wang, "The Protégé OWL Experience," in *Proc. OWL: Experiences and Directions Workshop*, 2005, vol. 2005.
[131] D. Dou, H. Qin, and P. Lependu, "OntoGrate: Towards Automatic Integration for Relational Databases and the Semantic Web through an Ontology-based Framework," *International Journal of Semantic Computing*, vol. 4, no. 1, p. 123, 2010.
[132] V. Eisenberg and Y. Kanza, "D2RQ/update: updating relational data via virtual RDF," in *Proceedings of the 21st international conference companion on World Wide Web*, 2012, pp. 497–498.
[133] "D2R Server specificatii." [Online]. Available: http://d2rq.org/d2r-server. [Accessed: 16-Sep-2012].
[134] M. Bucos, "Developing E-learning Informational Systems and Educational Virtual Organizations," Timisoara, 2007.
[135] M. Bucos, B. Dragulescu, and A. Ternauciuc, "Developing virtual labs at 'Politehnica' University of Timisoara," in *International Conference "Virtual University" 2008, Bratislava, Slovakia*, 2008.
[136] G. Carnevali and G. Buttazzo, "A virtual laboratory environment for real-time experiments," in *Intelligent components and instruments for control applications 2003 (SICICA 2003): a proceedings volume from the 5th IFAC International Symposium, Aveiro, Portugal, 9-11 July 2003*, 2003.
[137] Z. Nedic, J. Machotka, and A. Nafalski, "Remote laboratories versus virtual and real laboratories," in *Frontiers in Education, 2003. FIE 2003. 33rd Annual*, 2003, vol. 1, p. T3E–1.
[138] L. W. Anderson, D. R. Krathwohl, P. W. Airasian, K. A. Cruikshank, R. E. Mayer, P. R. Pintrich, J. Raths, and M. C. Wittrock, *A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives, Abridged Edition*. Allyn & Bacon, 2000.
[139] R. J. Mislevy and H. I. Braun, "Intuitive test theory," in *Annual Dinner Meeting of the Princeton Association for Computing Machinery (ACM) and Institute of Electrical and Electronics Engineers (IEEE) Computer Society Chapters, Kingston, NJ, May*, 2003, vol. 22.
[140] L. Soldatova and R. Mizoguchi, "Ontology of test," *Proc. Computers and Advanced Technology in Education*, pp. 173–180, 2003.
[141] D. Dicheva, R. Mizoguchi, and J. E. Greer, *Semantic Web Technologies for E-Learning*. IOS Press, 2009.
[142] G. Á. Rey, I. Celino, P. Alexopoulos, D. Damljanovic, M. Damova, N. Li, and V. Devedzic, "Semi-Automatic Generation of Quizzes and Learning Artifacts from Linked Data," *Proceedings of the 2nd International Workshop on Learning and Education with the Web of Data (LiLe2012)*, 2012.
[143] Aduna, "Sesame framework." [Online]. Available: http://www.openrdf.org/.
[144] A. Latchford, "phpSesame." [Online]. Available: https://github.com/alexlatchford/phpSesame.
[145] C. Bizer, "D2RQ platform." [Online]. Available: http://d2rq.org/.
[146] B. Dragulescu, M. Bucos, and R. Vasiu, "A Semantic Web approach for automated test generation," in *IADIS International Conference WWW/INTERNET 2012*, 2012, pp. 235–241.
[147] E. Duval, "Attention please! Learning analytics for visualization and recommendation," in *Proceedings of LAK11: 1st International Conference on Learning Analytics and Knowledge*, 2011, pp. 9–17.

[148] S. Retalis, A. Papasalouros, Y. Psaromiligkos, S. Siscos, and T. Kargidis, "Towards Networked Learning Analytics–A concept and a tool," in *Proceedings of the fifth international conference on networked learning*, 2006.