UNIVERZITET U NOVOM SADU
TEHNIČKI FAKULTET „MIHAJLO PUPIN“, ZRENJANIN

ONTOLOŠKI ZASNOVANA ANALIZA SEMANTIČKE KOREKTNOSTI MODELA PODATAKA PRIMENOM SISTEMA AUTOMATSKOG REZONOVANJA

D O K T O R S K A D I S E R T A C I J A

KANDIDAT
MR ZOLTAN KAZI

MENTOR
PROF. DR BILJANA RADULOVIĆ

ZRENJANIN, 2014. GODINE

UNIVERZITET U NOVOM SADU
TEHNIČKI FAKULTET „MIHAJLO PUPIN“, ZRENjANIN

KLJUČNA DOKUMENTACIJSKA INFORMACIJA

Redni broj: **RBR**	
Identifikacioni broj: **IBR**	
Tip dokumentacije: **TD**	Monografska publikacija
Tip zapisa: **TZ**	Tekstualni štampani materijal
Vrsta rada: **VR**	Doktorska disertacija
Autor: **AU**	Mr Zoltan Kazi
Mentor: **MN**	Prof. dr Biljana Radulović
Naslov rada: **NR**	Ontološki zasnovana analiza semantičke korektnosti modela podataka primenom sistema automatskog rezonovanja
Jezik publikacije: **JP**	srpski (latinica)
Jezik izvoda: **JI**	srpski i engleski
Zemlja publikovanja: **ZP**	Srbija
Uže geografsko područje: **UGP**	Vojvodina
Godina: **GO**	2014.
Izdavač: **IZ**	autorski reprint

Mesto i adresa: **MA**	Đure Đakovića bb Zrenjanin, 23000
Fizički opis rada: (broj poglavlja/strana /literaturnih referenci /citata/slika/tabela /listinga/priloga) **FO**	10/199/125/8/52/14/21/3
Naučna oblast: **NO**	Informacione tehnologije
Naučna disciplina: **ND**	Baze podataka, Informacioni sistemi, Veštačka inteligencija
Predmetna odrednica/Ključne reči: **PO**	Ontologije, korektnost modela podataka, sistemi automatskog rezonovanja
UDK	
Čuva se: **ČU**	Biblioteka Tehničkog fakulteta „Mihajlo Pupin“, Zrenjanin
Važna napomena: **VN**	
Izvod: **IZ**	U radu je izvršeno teoretsko istraživanje i analiza postojećih stavova i rešenja u oblasti validacije i provere kvaliteta modela podataka. Kreiran je teorijski model ontološki zasnovane analize semantičke korektnosti modela podataka primenom sistema automatskog rezonovanja i izvršena praktična implementacija teorijskog modela, što je potvrđeno i sprovedenim eksperimentalnim istraživanjem. Razvijena je softverska aplikacija za formalizaciju modela podataka i mapiranje ontologije u oblik Prolog klauzula. Formirana su pravila zaključivanja na predikatskom računu prvog reda, koja su integrisana sa modelom podataka i domenskom ontologijom. Upitima u okviru Prolog sistema, vrši se provera semantičke korektnosti modela podataka. Definisana je i metrika ontološkog kvaliteta modela podataka koja se bazira na odgovorima sistema automatskog rezonovanja.

Datum prihvatanja teme od NN veća: **DP**	29.03.2010.
Datum odbrane: **DO**	
Članovi komisije (naučni stepen/ime i prezime/zvanje/ fakultet) **KO**	
Član, predsednik:	Prof. dr Ivana Berković, redovni profesor, Tehnički fakultet „Mihajlo Pupin“, Zrenjanin
Član:	Prof. dr Dušan Malbaški, redovni profesor, Fakultet tehničkih nauka, Novi Sad
Član:	Prof. dr Dragica Radosav, vanredni profesor, Tehnički fakultet „Mihajlo Pupin“, Zrenjanin
Član:	Prof. dr Vladimir Brtka, vanredni profesor, Tehnički fakultet „Mihajlo Pupin“, Zrenjanin
Član, mentor:	Prof. dr Biljana Radulović, redovni profesor, Tehnički fakultet „Mihajlo Pupin“, Zrenjanin

UNIVERSITY OF NOVI SAD
TECHNICAL FACULTY „MIHAJLO PUPIN“, ZRENJANIN

KEY WORDS DOCUMENTATION

Accession number: **ANO**	
Identification number: **INO**	
Document type: **DT**	Monographic publicitation
Type of record: **TR**	Textual material, printed
Contents code: **CC**	Ph.D. Thesis
Author: **AU**	Zoltan Kazi, M.Sc.
Mentor: **MN**	Biljana Radulović, Ph.D., full professor
Title: **TI**	Ontology based semantic analyses of data model correctness by using automated reasoning system
Language of text: **LT**	Serbian (Latin letters)
Language of abstract: **LA**	Serbian and English
Country of publication: **CP**	Serbia
Locality of publication: **LP**	Vojvodina
Publication year: **PY**	2014.
Publisher: **PB**	Author reprint

Publication place: **PP**	Zrenjanin, 23000 Djure Djakovića bb
Physical description: (chapters/pages /references/citations /pictures/tables /listings/appendixes) **PD**	10/199/125/8/52/14/21/3
Scientific field: **SF**	Information Technologies
Scientific discipline: **SD**	Databases, Information systems, Artificial intelligence
Subject/Key words: **S/KW**	Ontologies, data model correctness, automated reasoning system
UC	
Holding data: **HD**	Library of Technical faculty „Mihajlo Pupin“, Zrenjanin
Note: **N**	
Abstract: **AB**	Work presents a theoretical study and analysis of existing theories and solutions in the area of data model validation and quality checking. It is created a theoretical model of ontology based analysis of data model semantic correctness by applying automated reasoning system which is practicaly implemented and confirmed by the conducted experimental research. A software application is developed for data model formalization and ontology mapping in Prolog clauses form. Reasoning rules are formed the in first-order predicate logic, which are integrated with the data model and domain ontology. Semantic correctness of the data model is checked with queries within Prolog system. Metrics of ontological quality of the data model are defined which are based on automated reasoning system replies.

Accepted by the Scientific Board on: **ASB**	2010-03-29
Defended on: **DE**	
Thesis defended board: (name/degree/title/ faculty) **DB**	
Member, President:	Ivana Berković, Ph.D., full professor, Technical faculty „Mihajlo Pupin“, Zrenjanin
Member:	Dušan Malbaški, Ph.D., full professor, Faculty of technical sciences, Novi Sad
Member:	Dragica Radosav, associate professor, Technical faculty „Mihajlo Pupin“, Zrenjanin
Member:	Vladimir Brtka, associate professor, Technical faculty „Mihajlo Pupin“, Zrenjanin
Member, Mentor:	Biljana Radulović, Ph.D., full professor, Technical faculty „Mihajlo Pupin“, Zrenjanin

Zahvaljujem se mentoru prof.dr Biljani Radulović,
prof. dr Ivani Berković, prof. dr Milanu Pavloviću,
prof. dr Petru Hotomskom, prof.dr Vladimiru Brtki,
prof. dr Dušanu Malbaškom, prof.dr Dragici Radosav
i mr Ljubici Kazi na korisnim sugestijama,
pomoći i podršci prilikom izrade doktorske disertacije!

SADRŽAJ

1. UVODNA RAZMATRANJA

Razvoj informacionih sistema predstavlja složen i dugotrajan proces, uz niz aktivnosti i rezultata koji nastaju u interakciji razvojnog tima i budućih korisnika sistema, a uz upotrebu i razvoj informacionih tehnologija. Kritične oblasti aktivnosti koje utiču na kvalitet i troškove razvoja predstavljaju procesi modelovanja. Ovi procesi se odnose na preslikavanje realnog sveta u skup apstraktnih koncepata, kao i prevođenje jednih vrsta modela u druge. U segmentu poslovnog modelovanja koristi se znanje o organizacionom sistemu, njegovom načinu poslovanja i funkcionisanja, strateškim planovima razvoja i informacionim potrebama. Daljim razvojem se modeli poslovnog domena prevode u modele dizajna budućeg rešenja informacionog sistema, na osnovu kojih se vrši programiranje i testiranje aplikativnog softvera, kreiranje izveštaja, implementacija baza podataka u nekom od softvera za upravljanje bazama podataka, dokumentovanje svih faza razvoja, modelovanja i sl.

Pored navedenog, može se primetiti da razvoj baza i modela podataka danas predstavlja značajan problem jer postoje veoma dinamična i agilna poslovna okruženja koja brzo menjaju zahteve i potrebe za podacima. Zbog toga, konceptualni modeli podataka koji su osnova i temelj dizajna baze podataka, treba da budu dovoljno fleksibilni kako bi omogućili jednostavnu i brzu ugradnju promena u baze podataka. U cilju razumevanja faktora koji određuju održivost konceptualnih modela podataka i poboljšanje procesa modelovanja konceptualnih modela podataka potrebno je znati procenjivati osobine konceptualnih modela podataka i njihov kvalitet na objektivan način.

Modelovanje podataka je jedna od najvažnijih aktivnosti u procesu razvoja informacionih sistema koju vrše projektanti baza podataka na osnovu domenskog znanja iz određene oblasti. Ova znanja mogu biti predstavljena u različitim formama: izvorni štampani ili elektronski dokumenti, specifikacija zahteva korisnika, opisi poslova, poslovna pravila, zakonski okviri, različiti poslovni pravilnici, modeli procesa, rečnici podataka sistemske analize, pseudo kodovi, biznis proces modeli, dijagrami slučajeva korišćenja i dr. Međusobna povezanost modela u procesu razvoja doprinosi povećanju složenosti, ali i integraciji procesa razvoja. Korišćenjem softverskih alata za projektovanje, kao što su CASE alati, modeli se usklađuju i povezuju, a pružaju mogućnost delimične automatske evaluacije sintaksnog aspekta modela i generisanja drugih modela, kao i izvršnih verzija softvera. Nakon završetka izgradnje logičkog modela podataka, u CASE alatima je moguće izvršiti proveru ispravnosti modela tako da softver može upozoriti projektante baza podataka da li čine greške u kreiranju modela, ali samo sa aspekta specifične sintakse nekog modela podataka. Ugradnja semantike u model podataka zavisi od sposobnosti i iskustva projektanta da kroz određene koncepte kao što su struktura dijagrama, ograničenja ili operacije preslika relevantne osobine realnog poslovnog sistema.

S obzirom da je baza podataka jezgro svakog informacionog sistema ili složenijeg softverskog proizvoda koji organizovano čuva i manipuliše podacima, postavljaju se određena pitanja:

- Da li baza podataka odgovara specifičnim osobinama poslovnog sistema iz određene problemske oblasti?
- Koliko kvalitet implementirane baze podataka zavisi od iskustva i znanja projektanta i dizajnera?

Na ova pitanja CASE alati ne mogu dati odgovore i usklađenost modela podataka i domenske oblasti za koju se isti gradi zavisi, u velikoj meri, od iskustva i znanja projektanta. Sa druge strane, entiteti kao logičke celine koje se kreiraju u postpuku logičkog projektovanja modela podataka jesu produkti mišljenja. Opšte je poznato da različiti umovi mogu videti iste stvari na različit način, što u postupku izradnje modela podataka može rezultovati različitim modelima ili njihovim verzijama i varijacijama, a samim tim i različitim bazama podataka, koje u većoj ili manjoj meri odgovaraju ili ne korisniku, tj. oblasti u kojoj se koriste.

Problematika disertacije jeste teoretsko i empirijsko istraživanje modaliteta provere kvaliteta modela podataka sa aspekta semantike i domenske valjanosti za oblast u kojoj se želi kreirati konceptualni model, sa ciljem da se izgrade kvalitetnije i adekvatnije baze podataka, koje u manjoj meri zavise od iskustva projektanta.

Metodološki okvir rada je prikazan u drugom poglavlju. Definisani su ciljevi i zadaci istraživanja, formulisana glavna hipoteza istraživanja i pomoćne hipoteze. Navedeni su očekivani rezultati istraživanja, kao i naučna i društvena opravdanost istraživanja.

Treće poglavlje prikazuje osnovne teoretske koncepte, saznanja, pojmove i definicije koje se odnose na tematiku istraživanja u sledećim oblastima: ontologije, modeli podataka sa posebnim osvrtom na model entiteta poveznika i automatsko rezonovanje.

Pregled vladajućih stavova u području istraživanja je prikazan u četvrtom poglavlju. Navedena su istraživanja u oblasti evaluacije modela podataka, testiranja korektnosti modela podataka, veza ontologija, predikatskog računa prvog reda i modela podataka, a na samom kraju poglavlja navedena su istraživanja iz oblasti automatizacija evaluacije modela podataka.

Peto poglavlje prikazuje model ontološki zasnovane analize semantičke korektnosti modela podataka primenom sistema automatskog rezonovanja. Detaljno je prikazana formalizacija modela podataka u oblik rečenica predikatskog računa prvog reda i njihovu transformacija u činjenice Prolog jezika. Opisan je način kreiranja ontologije i njeno mapiranje na predikatski račun i Prolog jezik. Definisana su pravila zaključivanja na osnovu kojih se u okviru Prolog sistema, čije jezgro predstavlja sistem za automatsko rezonovanje, vrši provera sematičke korektnosti modela podataka.

Implementacija teoretskog modela prikazana je u okviru šestog poglavlja. Opisani su i korišteni softverski alati u realizaciji rada. Detaljno je prikazana aplikacija Data Model Validator koja je programirana u cilju integracije pojedinih koraka i automatizacije postupka semantičke analize modela podataka.

Empirijsko istraživanje, sprovedeno na Univerzitetu u Novom Sadu, na Tehničkom fakultetu „Mihajlo Pupin“, sa studentima smera Informacione tehnologije je prikazano u sedmom poglavlju. Opisan je i primer koji je poslužio za istraživanje. Prikazano je detaljno testiranje modela na dva test primera. U ovom poglavlju su dati i rezultati empirijskog istraživanja sa podacima, statistikom i odgovarajućim dijagramima. Izvršena je i analiza rezultata empirijskog istraživanja.

Zaključak i zaključna razmatranja su navedena kao osmo poglavlje, u kome su date i smernice za dalja istraživanja i urađena je analiza rezultata sa stanovišta hipoteza istraživanja.

Deveto poglavlje sadrži spisak literaturnih referenci koje su korištene u izradi disertacije, spsiak objavljenih radova autora iz oblasti doktorske disertacije i korištene literaturne reference sa Interneta.

Poslednje, deseto poglavlje čine prilozi: biografija kandidata, pojmovnik osnovnih koncepata sadržanih u radu i listing programa Data Model Validator koji predstavlja jedan od rezultata istraživanja i rada na disertaciji.

2. METODOLOŠKI OKVIR RADA

2.1. CILJ I ZADACI ISTRAŽIVANJA

Cilj istraživanja je:

- Kreiranje teorijskog modela ontološki zasnovane analize semantičke korektnosti modela podataka primenom sistema automatskog rezonovanja.
- Praktična implementacija ovakvog sistema, čija bi upotrebljivost, efekti, pozitivne i negativne karakteristike bile utvrđene kroz eksperimentalno istraživanje u obrazovanju studenata u oblasti informatike.

Zadaci istraživanja:

- Definisanje teorijskog modela ontološki zasnovane analize semantičke korektnosti modela podataka primenom sistema automatskog rezonovanja.
- Analiza, projektovanje i implementacija softverske podrške za ontološki zasnovanu analizu semantičke korektnosti modela podataka.
- Priprema implementiranog sistema za korišćenje u obrazovnom procesu u nastavnom radu, u okviru časova nastave i izrade seminarskih radova studenata.
- Empirijsko istraživanje efikasnosti implementirane softverske podrške upotrebom eksperimentalne metode. Uzorak će predstavljati modeli podataka u okviru nastavnog procesa, u radu studenata i u okviru časova nastave i učenja za nastavne predmete: Informacioni sistemi, Baze podataka i Kompleksne baze podataka.
- Analiza rezultata empirijskog istraživanja i sinteza zaključaka, koji se odnose na pozitivne i negativne karakteristike, odnosno prednosti i nedostatke implementiranog sistema.
- Razmatranje perspektiva i mogućih pravaca daljeg razvoja sistema.

2.2. HIPOTEZE ISTRAŽIVANJA

2.2.1. Glavna hipoteza

Moguće je kreirati teorijski model za ontološki zasnovanu analizu semantičke korektnosti modela podataka primenom sistema automatskog rezonovanja.

2.2.2. Podhipoteze

1. Upotreba ontološki zasnovane analize semantičke korektnosti modela podataka primenom sistema automatskog rezonovanja povećava kvalitet kreiranih baza podataka i adekvatniju primenljivost pratećih aplikativnih softvera, u okviru određene problemski orijentisane domenske oblasti.

2. Ontološki zasnovana analiza semantičke korektnosti modela podataka primenom sistema automatskog rezonovanja može da se koristi u okviru nastavnog procesa, čime se povećava efikasnost nastave u oblasti baza podataka, informacionih sistema i sistema veštačke inteligencije, kao i kvalitet softverskih projekata u nastavnom procesu.
3. Moguće je softverski automatizovati ontološki zasnovanu analizu semantičke korektnosti modela podataka primenom sistema automatskog rezonovanja.

2.3. OČEKIVANI REZULTATI ISTRAŽIVANJA

- Teorijski model ontološki zasnovane analize semantičke korektnosti modela podataka primenom sistema automatskog rezonovanja, razrađen na osnovu teorijske analize problemskih oblasti istraživanja, analize postojećih rešenja i empirijskog istraživanja.
- Projektovanje i implementacija softvera za podršku ontološki zasnovane analiza semantičke korektnosti modela podataka primenom sistema automatskog rezonovanja, zasnovanog na prethodno razvijenom teorijskom modelu. Softver treba da automatizuje procese mapiranja domenske ontologije i izabranog modela podataka na predikatski račun prvog reda.
- Rezultati eksperimentalnog istraživanja karakteristika primene implementiranog sistema u praksi, odnosno u nastavnom procesu koji se odnosi na projektovanje i implementaciju modela podataka.
- Prikaz perspektiva i mogućih pravaca daljeg razvoja.

2.4. NAUČNA I DRUŠTVENA OPRAVDANOST ISTRAŽIVANJA

Istraživanjem u oblasti analize i verifikacije modela podataka primenom sistema automatskog rezonovanja i primenom rezultata istraživanja omogućio bi se doprinos u sledećim oblastima:

- Naučni doprinos istraživanja se ogleda u kreiranju teorijskog modela ontološki zasnovane analize semantičke korektnosti modela podataka primenom sistema automatskog rezonovanja.
- Praktični doprinos ogleda se u implementaciji softverske podrške za ontološki zasnovanu analizu semantičke korektnosti modela podataka. Implementacijom teorijskog modela bi se proširila oblast primene sistema automatskog rezonovanja.
- Društveni značaj ogleda se u omogućavanju unapređivanja nastavnog procesa u oblasti baza podataka i razvoja informacionih sistema, korišćenjem postupka u okviru nastavnog rada studenata.

3. TEORIJSKA OSNOVA

3.1. ONTOLOGIJE

Ontologije se definišu na različite načine u zavisnosti od naučne oblasti u kojoj se koriste. U računarskom inženjerstvu ontologija se odnosi na predstavljanje znanja (*Knowledge Representation*). U oblasti informacionih sistema postoji problem postojanja različitih klasa podataka koje imaju svoje vlastite termine kojima se predstavljaju informacije. Kada se takve informacije nalaze na jednom mestu zajedno, terminološke i konceptualne različitosti mogu se prevazići upotrebom odgovarajućih ontologija. Ontologija se definiše kao skup termina koji se koriste da bi se opisao domen, tj. oblast znanja. Ontologije koriste ljudi, baze podataka i softverske aplikacije koje dele informacije iz određenog domena. Domen predstavlja specifičnu predmetnu oblast znanja. Prema istraživanjima koja se sprovode, u oblasti veštačke inteligencije, pojam ontologija se vezuje za:

- ponovnu upotrebu znanja (*Reusability*), i
- deljenje znanja iz određenog domena (*Sharing*).

Najpopularniju definicija ontologije je dao [Gruber 1995] kao "formalnu, eksplicitnu specifikaciju deljene konceptualizacije" ("…*a formal, explicit specification of a shared conceptualization*"). Pojam formalna se odnosi na činjenicu da ontologija treba da je razumljiva, tj. čitljiva za mašine. Eksplicitna znači to da su eksplicitno definisani svi koncepti i ograničenja koja se koriste. Deljena ukazuje na to da ontologija treba da «uhvati» znanje prihvaćeno od strane zajednice u kojoj se uvodi i koristi. Konceptualizacija se odnosi na apstraktan model fenomena iz stvarnog sveta koji se dobija identifikovanjem relevantnih koncepata ovih fenomena [El-Ghalayini et al., 2005]. Još jednu relevantnu definiciju ontologije je predstavio [Guarino 1998]: «Ontologija je skup logičkih aksioma dizajniran tako da se odnosi na planirano značenje nekog rečnika» ("*a set of logical axioms designed to account for the intended meaning of a vocabulary*"). U ovoj definiciji je istaknuta uloga teorije logike u značenju i predstavljanju ontologije.

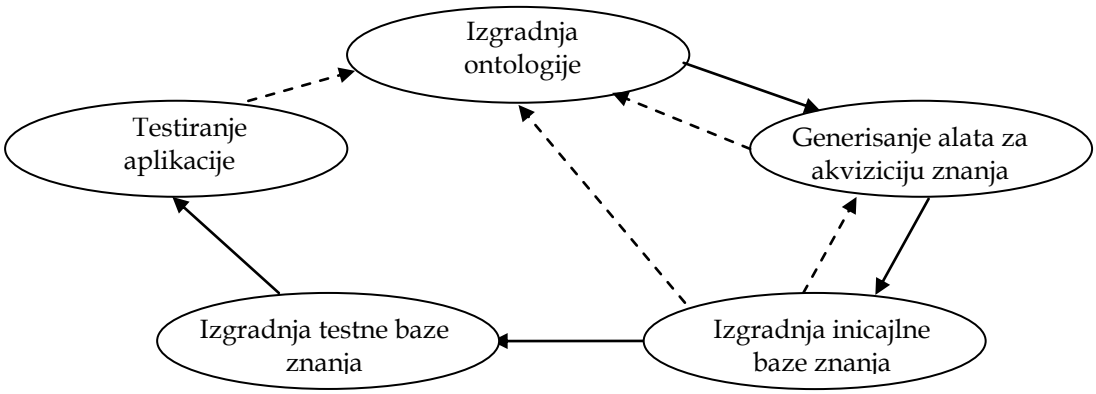
Formalna definicija, prema [Bergamaschi et al., 2004], određuje ontologiju kao skup prikazan u definiciji 3.1.

Definicija 3.1. Ontologija *O* je skup sastavljen od pet elemenata (*C, I, R, F, A*), gde je:

C – konačan skup koncepata, apstrakcija kojima se opisuju objekti realnog sveta,
I – konačan skup instanci koncepata, tj. objekata stvarnog sveta,
R – konačan skup relacija između elemenata skupa *I*,
F – konačan skup funkcija definisanih nad objektima realnog sveta,
A – konačan skup aksioma na predikatskom računu prvog reda koji određuju značenje klasa objekata, relacija između objekata i funkcija definisanih nad objektima realnog sveta.

Prema istraživanjima u različitim oblastima, zaključeno je da osnovna namena ontologija jeste omogućavanje predstavljanja, prenošenja i razmene znanja iz neke oblasti. Zaključak koji se može izvesti iz različitih definicija je taj da ontologije formalizuju semantiku domena eksplicitno opisujući njihove elemente i s obzirom da se sastoje od koncepata, opisuju mogućnosti tih koncepata i osobina koje opisuju relacije između ovih koncepata.

Koraci u iteratvnom postupku izgradnje ontologije, prema [Devedžić 2004]:



Slika 3.1. – Postupak izgradnje ontologije

3.1.1. Ontološki jezici i alati

Sa razvojem Semantičkog Web-a u prethodnih desetak godina pojavio se veliki broj jezika za modelovanje ontologija. Neki od njih su razvijani još od početka 90-tih godina 20. veka, naročito u oblasti veštačke inteligencije. Prvobitni jezici pripadaju eri pre XML-a, dok su najnoviji zasnovani na XML standardu. [Gomez-Perez et al., 2002]

Neki od najpoznatijih ranih ontoloških jezika su:

- KIF – zasnovan na predikatskom računu prvog reda,
- Loom – bazira se na deskriptivnoj logici,
- Shoe – predstavlja ekstenziju HTML jezika,

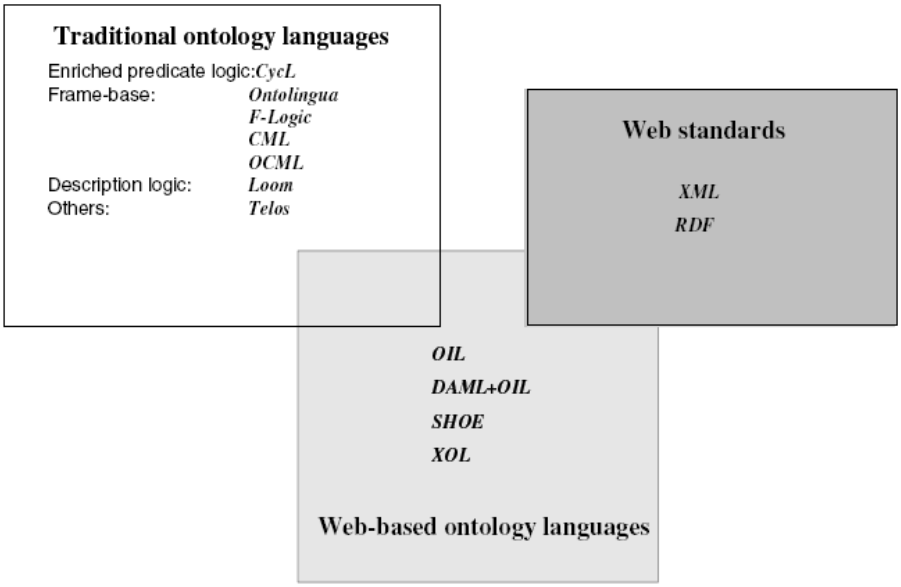
W3C (*World Wide Web Consortium*) je prihvatio sledeće ontološke jezike [Gomez-Perez et al., 2002]:

- RDF - *Resource Descriptiom Framework* jezik za semantičke mreže koji služi za opis resursa na Web-u,
- RDFS - *RDF Schema* je proširenje RDF jezika,
- OIL – *Ontology Interchange Language*, jezik zasnovan na deskriptivnoj logici,
- DAML+OIL – poslednja verzija ranijeg *DARPA Agent Markup Language* jezika koji je je nastao kao kombinacija dva jezika,
- OWL - *Web Ontology Language* nastao razvojem iz DAML+OIL i trenutno predstavlja najpopularniji ontološki reprezentacioni jezik,
- XML - *EXtensible Markup Language*.

Ontologije se mogu, prema [Luo 2005], svrstati u tri kategorije, u zavisnosti od domena, tj. područja koji modeluju:

1. Ontologije za predstavljanje znanja (*Knowledge Representation Ontology*) su sistemi za predstavljanje znanja koji sjedinjuju ontološke radne okvire (*Framework*). Znanje se predstavlja ili u KIF formatu (*Knowledge Interchange Format*) koji je logički sveobuhvatan i obezbeđuje definicije objekata, funkcija i relacija ili, pak, na kompjuterski orjentisanom jeziku dizajniranom za razmenu znanja između različitih programa;
2. Ontologije o opštem svetskom znanju (*Upper Level Ontology*);
3. Ontologije specifičnih domena (*Domain Specific Ontology*) sa primarnim fokusom na povezivanju struktura i ponašanja kroz koncept ovlašćenja.

Ontološki jezici se zasnivaju na upotrebi XML jezika. Oni moraju biti kompatibilni sa drugim industrijskim i Web standardima, što je u prošlosti olakšalo razvoj alata i jezika za razvoj ontologija. Ontološki jezici se mogu klasifikovati kao što je prikazano na sledećoj šemi na slici 3.2:

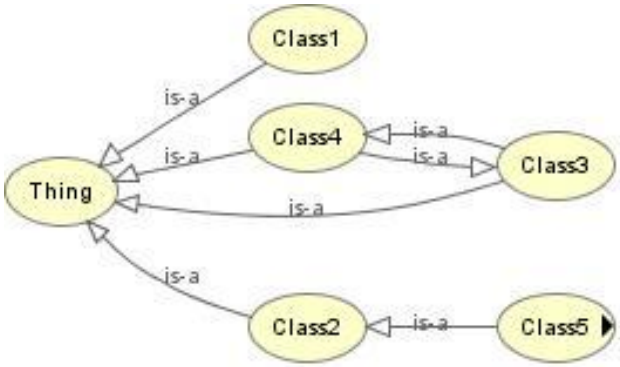


Slika 3.2. – Klasifikacija ontoloških jezika [Su et al., 2002]

Formalne osnove svih vodećih jezika Semantičkog Web-a (OWL, RDF i RDF(S)) su zasnovane na klasičnoj predikatskoj logici. [Antoniou et al., 2005]

Protégé [1] je trenutno jedno od najznačajnijih, ako ne i vodeće okruženje za razvoj ontologija. Razvijen je na Univerzitetu Stanford i prošao je kroz nekoliko modifikacija i verzija. Omogućuje jednostavno definisanje koncepata (klasa) u ontologiji, osobina, taksonomija, različitih ograničenja kao i instance klasa (podatke u bazi znanja). Poseduje grafički alat za sakupljanje i akviziciju znanja u bazu znanja koji odgovara ontologiji i koja se može koristiti za rešavanje različitih zadataka. *Protégé* podržava nekoliko jezika za predstavljanje ontologija: OWL, RDF(S), DAML+OIL, XMI (*XML Metadata Interchange*) i druge.

Prema [Lammari et al., 2004], osnovna karakteristika svih ontologija je definisanje hijerarhije koncepata/objekata, između kojih se uspostavljaju različite semantičke veze. Primer apstraktne ontologije koji je prikazan na slici 3.3. opisuje koncept, stvar, tj. objekat iz realnog sveta (*Thing*) koja ostvaruje semantičke veze (*Is-a*) sa različitim klasama/objektima (*Class*) koje su opisane određenim osobinama (*Properties*) i ostalim konceptima iz oblasti ontologija. Koncepti u realnom svetu predstavljaju konkretne objekte i pojmove.



Slika 3.3. – Grafički prikaz apstraktnih klasa ontologije

Ontologije, kao što je već navedeno, se kreiraju specifičnim softverskim alatima koji generišu XML elemente, oznake i zapis u OWL jeziku, kao u sledećem primeru koji predstavlja opis ontologije sa slike 3.3.:

Listing 3.1:

```
<?xml version="1.0"?>
<!DOCTYPE Ontology [
    <!ENTITY owl "http://www.w3.org/2002/07/owl#" >
    <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
    <!ENTITY owl2xml "http://www.w3.org/2006/12/owl2-xml#" >
    <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
    <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
    <!ENTITY Ontology
"http://www.semanticweb.org/ontologies/2009/9/Ontology.owl#" >]>
<Ontology xmlns="http://www.w3.org/2006/12/owl2-xml#"
     xml:base="http://www.w3.org/2006/12/owl2-xml#"
     xmlns:owl2xml="http://www.w3.org/2006/12/owl2-xml#"
xmlns:Ontology="http://www.semanticweb.org/ontologies/2009/9/Ontology.owl#"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:owl="http://www.w3.org/2002/07/owl#"
     URI="http://www.semanticweb.org/ontologies/2009/9/Ontology.owl">
    <Declaration>
        <Class URI="&Ontology;Class1"/>
    </Declaration>
    <Declaration>
        <Class URI="&Ontology;Class2"/>
    </Declaration>
    <SubClassOf>
        <Class URI="&Ontology;Class3"/>
        <Class URI="&Ontology;Class4"/>
    </SubClassOf>
```

Listing 3.1(nastavak):

```
    <Declaration>
        <Class URI="&Ontology;Class3"/>
    </Declaration>
    <SubClassOf>
        <Class URI="&Ontology;Class4"/>
        <Class URI="&Ontology;Class3"/>
    </SubClassOf>
    <Declaration>
        <Class URI="&Ontology;Class4"/>
    </Declaration>
    <SubClassOf>
        <Class URI="&Ontology;Class5"/>
        <Class URI="&Ontology;Class2"/>
    </SubClassOf>
    <Declaration>
        <Class URI="&Ontology;Class5"/>
    </Declaration>
    <SubClassOf>
        <Class URI="&Ontology;Class6"/>
        <Class URI="&Ontology;Class5"/>
    </SubClassOf>
    <Declaration>
        <Class URI="&Ontology;Class6"/>
    </Declaration>
    <FunctionalObjectProperty>
        <ObjectProperty URI="&Ontology;Propertie1"/>
    </FunctionalObjectProperty>
    <Declaration>
        <ObjectProperty URI="&Ontology;Propertie1"/>
    </Declaration>
    <InverseFunctionalObjectProperty>
        <ObjectProperty URI="&Ontology;Propertie2"/>
    </InverseFunctionalObjectProperty>
    <Declaration>
        <ObjectProperty URI="&Ontology;Propertie2"/>
    </Declaration>
</Ontology>
```

3.1.2. Karakteristike i elementi OWL ontologija

OWL ontologija se, prema [Horridge 2009], sastoji od jedinki (pojedinaca, individua), osobina i klasa. Jedinke su instance klasa dok su osobine binarne relacije između jedinki. Za neke relacije mogu postojati i inverzne relacije. Osobine mogu biti ograničene na samo jednu vrednost, tj. mogu biti funkcionalne. Takođe, osobine mogu biti tranzitivne ili simetrične.

OWL klase se predstavljaju kao skupovi koji sadrže jedinke. Opisane su pomoću formalnih opisa koji sadrže precizne uslove za pripadnost klasi. Klase se mogu organizovati u hijerarhije odnosa nadklasa, tj. superklasa i njihovih podklasa koje su još poznate i kao taksonomije. Podklase su specijalizacija njihovih nadklasa.

OWL osobine predstavljaju relacije. Postoje tri osnovne vrste osobina:
- osobine objekata (*Object Properties*),
- osobine tipova podataka (*Datatype Properties*),
- osobine beleški (*Annotation Properties*).

Osobine objekata povezuju jednu jedinku sa drugom, mogu imati domen i specificirani opseg. Osobine povezuju jedinke iz domena sa jedinkama iz opsega.

Osobine tipova podataka opisuju relacije između jedinki i vrednosti podataka. Ove osobine mogu biti predefinisani skupovi tipova podataka ili se mogu formirati specifična ograničenja za moguće vrednosti.

Osobine beleški se mogu koristiti kako bi se preko meta podataka dodale informacije klasama, jedinkama i osobinama objekata ili osobinama tipova podataka.

U OWL jeziku osobine mogu imati podosobine, tako da je moguće formirati i hijerarhije osobina. Podosobine specijalizuju svoje nadosobine na isti način kao i podklase njihove nadklase.

Karakteristike osobina mogu biti sledeće:
- funkcionalne – može postojati samo jedna jedinka koja je povezana sa tom osobinom;
- inverzne funkcionalne – ukoliko je inverzna osobina funkcionalna;
- tranzitivne – osobina povezuje jedinku A sa jedinkom B i jedinku B sa jedinkom C, tada je jedinka A povezana sa jedinkom C;
- simetrične – osobina P je simetrična ukoliko povezuje jedinku A sa jedinkom B a jedinka B je takođe povezana sa jedinkom A preko osobine P;
- antisimetrične – osobina P povezuje jedinku A sa jedinkom B, tada jedinka B nije povezana sa jedinkom A preko osobine P;
- refleksivne – osobina P je refleksivna kada povezuje neku jedinku sa istom tom jedinkom;
- irefleksivne – osobine koje se povezuju jedinku A sa jedinkom B, ako jedinka A nije identična sa jedinkom B.

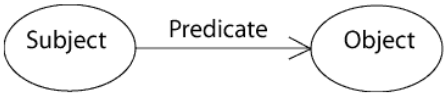
U OWL jeziku se ograničenja mogu svrstati u tri osnovne kategorije:
- kvantifikatorska ograničenja (*Quantifier Restrictions*),
- ograničenja kardinaliteta (*Cardinality Restrictions*),
- ograničenja tipa «ima vrednost» (*hasValue Restric*tions).

3.1.3. RDF

RDF (*Resource Descriptiom Framework*) je radni okvir za opis i predstavljanje informacija na Web-u. Namenjen je za predstavljanje metapodataka o Web resursima, kao što su podaci o naslovu, autoru, datum modifikacije stranice na Internetu, informacije o licenci i pravima i sl. Struktura svakog izraza u RDF je kolekcija trojki, od kojih se svaka sastoji od subjekta, predikata i objekta. Skup takvih trojki se zove RDF graf gde su

čvorovi grafa subjekti i objekti. Ceo graf je konjukcija svih trojki subjekata, predikata i objekata. Činjenice koje se predstavljaju RDF trojkom ukazuju na relaciju između dve stvari gde je ime predikata relacija ili osobina, a subjekat i objekat predstavljaju te dve stvari. [Antoniou et al., 2005], [6]

Tabela 3.1: Struktura RDF izraza [6]

Forma grafa	
Uređena trojka	`subject predicate object`
Relaciona forma	`predicate(subject,object)`

RDF/XML	`<rdf:Description rdf:about="subject">`
	`  <ex:predicate>`
	`  <rdf:Description rdf:about="object"/>`
	`  </ex:predicate>`
	`</rdf:Description>`

Forma "*Turtle*"	`subject ex:predicate object.`

RDF koristi sledeće osnovne koncepte:

- Grafički model podataka,
- Rečnik baziran na URI,
- Tipove podataka,
- Literale (npr. <xsd:boolean, "true">),
- XML sintaksu (*XML Serialization Syntax*),
- Predstavljanje jednostavnih činjenica (*Simple facts*),
- Semantičke veze između izraza (*Entailment*).

Komponente RDF trojki, tj. tripleta su i URI (*Uniform Resource Identifier*) – kompaktni nizovi znakova za identikovanje apstraktnog ili fizičkog resursa na Web-u. Komponente URI-ja su obično prilično dugi stringovi, koji se u RDF dokumentu razlikuju na samom kraju XML elementa, gde se navode subjekti, predikati i objekti. Zbog toga se koriste entiteti za konstantne delove elementa koji se nazivaju imenovani prostori (*Namespace*). Naredni primeri [7] ilustruju URI-je koji se često koriste:

```
ftp://ftp.is.co.za/rfc/rfc1808.txt
      (ftp šema za File Transfer Protocol servise)
http://www.math.uio.no/faq/compression-faq/part1.html
      (http šema za Hypertext Transfer Protocol servise)
```

RDF obezbeđuje sintaksu zasnovanu na tekstu (*Turtle*) za čuvanje i razmenu grafova [8]. Primer:

Listing 3.2:

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
@prefix contact: <http://www.w3.org/2000/10/swap/pim/contact#>.

<http://www.w3.org/People/EM/contact#me>
  rdf:type contact:Person;
  contact:fullName "Eric Miller";
  contact:mailbox <mailto:em@w3.org>;
  contact:personalTitle "Dr.".
```

3.2. MODELI PODATAKA

Modeli podataka su specifične teorije pomoću kojih se vrši specifikacija i projektovanje konkretnih baza podataka i informacionih sistema. Model podataka je formalna apstrakcija putem koje se realni svet preslikava u bazu podataka. Modelom podataka se, preko skupa podataka i njihovih veza, prikazuje stanje realnog sistema u jednom određenom trenutku. U ovoj oblasti rezultati koji su poslužili kao literaturni izvori, objavljeni su u [Chen 1976], [Elmasri et al., 2007], [Ullman et al., 2002], [Mogin i sar. 2000], [Mogin i sar. 1996], [Lazarević i sar. 2003], [Devedžić 2004] i [Light 1997].

U istoriji razvoja modela podataka razlikuju se sledeće generacije modela podataka:

1. generacija - klasični programski jezici, sa relativno jednostavnim tipovima podataka i siromašnom semantikom, ne mogu predstaviti realan sistem.
2. generacija - modeli konvencionalnih sistema za upravljanje bazom podataka: hijerarhijski, mrežni i relacioni, poseduju znatno bogatiju semantiku od 1. generacije, strukture podataka i složenije tipove podataka, ali nepotpuno opisuju realan sistem.
3. generacija - semantički bogati modeli – ER (*Entity Relationship Attribute*), SDM (*Semantic Data Model*) i drugi, poseduju specifične koncepte za detaljan opis realnog sistema, sa nedostatkom u pogledu potpune softverske realizacije (*CASE* alati).
4. generacija - objektni model podataka (OOM), nastao je početkom 90-tih godina 20. veka, kao implementacija struktura podataka iz objektno orijentisanih programskih jezika (*C++, SmallTalk*) u sisteme za upravljanje bazama podataka, u oblasti inženjerskog projektovanja (*CAD* sistemi), ili kao implementacija ugnježdenih tabela u okviru klasičnog relacionog modela podataka.

Najpoznatiji način za prikazivanje modela podataka je ER model ili MOV (Model Objekti-Veze-obeležja). Autor je *Peter Chen*, sa Univerziteta MIT u Bostonu, 1978. godine. Osnovni elementi ER modela podataka su: entiteti, atributi, identifikacioni atributi, veze između entiteta, sa definisanim ograničenjima – kardinalitetom. ER model podataka je, zbog koncepata koji nisu u potpunosti mogli kvalitetno da predstave osobine realnih poslovnih sistema, proširen konceptima kao što su: generalizacija/specijalizacija, gerund, kategorija, jak i slab tip entiteta. Ovako modifikovani model je nazvan EER (*Enhanced Entity Relationship Data Model*) i u prethodnih 20-ak godina se pojavio znatan broj softvera (*Computer Aided Software Engineering - CASE* alata) koji omogućuju brži proces logičkog i fizičkog projektovanja modela baze podataka. Najpoznatiji su: *OracleCase, BPWin/ERWin, Sybase Power Designer* i dr.

Objektna orijentacija je pristup 4. generacije modela podataka u kome se neki sistemi organizuju kao kolekcije međusobno povezanih objekata koji sarađujući ostvaruju postavljene ciljeve. Pod objektom se podrazumeva entitet koji je sposoban da čuva svoja stanja i koji stavlja okolini na raspolaganje skup operacija preko kojih se ta stanja

prikazuju ili menjaju. Svi objekti istog tipa imaju isti skup stanja (osobina) i jedinstveno ponašanje (isti skup operacija). Konkretni objekat je pojavljivanje ili instanca datog tipa objekta, tj. klase objekta. Stanje objekta je određeno vrednostima njegovih osobina (atributa) kao i vezama između tog objekta i drugih objekata u nekom sistemu. Ove osobine objekata se menjaju u vremenu. Ponašanje objekta se opisuje skupom operacija koje taj objekat izvršava ili koje se nad njim izvršavaju. Svaka operacija ima kao argument objekat kome je pridružena, može da ima listu ulaznih i izlaznih parametara, a takođe može da vrati tipizovani rezultat.

Rešenja koja se danas koriste u projektovanju baza podataka jesu ta da se specifikacija sistema uradi korišćenjem alata četvrte ili treće generacije, a zatim se vrši prevođenje takve specifikacije u Relacioni model koji pripada drugoj generaciji i za koji postoji veliki broj sistema za upravljanje bazama podataka. Najpoznatiji su: *Oracle, DB2, Ingres, Progress, SyBase, DBase, Paradox, SQL Server, FoxPro, MySQL, Access* i dr. Sistem za upravljanje bazom podataka (SUBP, engl. *Database Management System - DBMS*) je softverski sistem za kreiranje, čuvanje, ažuriranje i pretraživanje podataka. Najstariji modeli podataka kao što su klasični programski jezici, mrežni i hijerarhijski model više nemaju ni praktičan ni teorijski značaj.

Za logičko i konceptualno modelovanje podataka dva najčešće korišćena modela su EER i objektni model podataka. Od 90-tih godina prošlog veka su urađene razne studije koje valorizuju i porede različite modele podataka. Zaključeno je da je EER model superioran u odnosu na modele prve i druge generacije, posebno zbog načina na koji se modeluju relacije između entiteta. Objektni model podataka je široko rasprostranjen u programiranju i softverskom inženjerstvu uopšte. U poslednjih desetak godina je ostvario značajan prodor u oblasti projektovanja baza podataka zbog dobre povezanosti sa elementima programskih rešenja i brze implementacije nakon izgradnje modela, što je izuzetno bitan faktor u razvoju softverskih rešenja. Neka eksperimentalna istraživanja [De Lucia et al., 2009], [Shoval 1997], pokazala su da je, stariji, EER model ipak podesniji od objektnog modela podataka za logičko projektovanje baze podataka, jer je jednostavniji za razumevanje, bolji je za predstavljanje kompleksnijih relacija između entiteta (unarne i ternarne veze), zahteva manje vremena za kreiranje šeme i projektanti ga više koriste. Preporuka istraživača iz ove oblasti je da se dizajn baze podataka uradi u EER modelu, pa da se zatim izvrši njegovo mapiranje u OOM, u kom je potrebno dodati metode i poruke koje opisuju ponašanje objekata.

3.2.1. ER model podataka

Model entiteta poveznika (*Entity Relationship Data Model*) se bazira na osnovnoj ideji da se realan svet i njegovi delovi opisuju pomoću dva osnovna koncepta: entiteta i poveznika. Entitet je «nešto» što se može jednoznačno identifikovati. To je osnovna jedinica posmatranja u nekom sistemu. Entitet je apstraktna predstava nekog objekta, subjekta, događaja, pojma u ljudskom intelektu. Entiteti se mogu klasifikovati u skupove sličnih entiteta.

Definicija 3.2. „Neka je e entitet, tada je skup entiteta $E=\{e|P(e)\}$, gde je $P(e)$ predikat čija istinitosna vrednost ukazuje da li e pripada skupu E. Ako e poseduje osobinu $P(e)$, tada e prpada E. Isti entitet e može pripadati raličitim skupovima entiteta.“ [Mogin i sar. 1996, strana 12]

Skupovi sličnih entiteta nazivaju se klasama entiteta, gde svi entiteti jedne klase poseduju bar jednu zajedničku osobinu, na osnovu koje su svrstani u istu klasu. Broj zajedničkih osobina jedne klase je uglavnom veći od jedan, a nazivaju se obeležjima ili atributima.

Definicija 3.3. „Obeležje koje se ne može dekomponovati na komponente, koje takođe predstavljaju obeležja naziva se elementarno obeležje.“ [Mogin i sar. 1996 , strana 13]

Definicija 3.4. „Složeno obeležje je $X=\{A_1,..., A_k\}$, gde su A_i, $1 \le i \le k$, elementarna obeležja.“ [Mogin i sar. 1996, strana 13]

Svakom atributu, tj. obeležju odgovara jedan skup svih mogućih vrednosti koje to obeležje može imati. Ovakav skup vrednosti naziva se domenom obeležja. Domen obeležja A označava se sa $dom(A)$. Domen obeležja može posedovati posebno ime i isti domen se može pridružiti većem broju različitih obeležja.

Sa tačke gledišta razvoja informacionog sistema, prilikom modelovanja podataka, nisu sva obeležja klase entiteta podjednako važna. Obeležja koja su relevantna za ostvarivanje ciljeva razvoja informacionog sistema grade model klase entiteta. Model klase entiteta naziva se tipom entiteta.

Definicija 3.5. „Izraz oblika $N(A_1,..., A_n)$ predstavlja model skupa entiteta $E=\{e|P(e)\}$ i naziva se tipom entiteta, ako i samo ako N predstavlja ime skupa $\{e|P(e)\}$, a $A_1,...,A_n$ obeležja entiteta skupa $\{e|P(e)\}$.“ [Mogin i sar. 1996, strana 15]

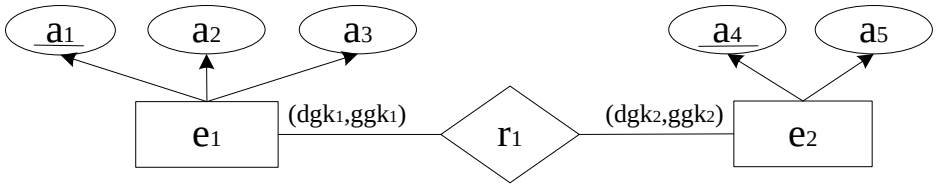
Naziv tipa entiteta treba da odražava semantiku i smisao apstraktnog opisa klase entiteta iz realnog sveta. Svaka klasa entiteta ima konačan broj osobina koji je zajednički svim realnim entitetima. Ako je $\{A_1,..., A_m\}$ skup osobina entiteta E, tada je skup obeležja $\{A_1,..., A_n\}$ izabranih za izgradnju tipa entiteta kao modela klase E podskup skupa obeležja $\{A_1,..., A_m\}$. Tip entiteta je skup entiteta koji imaju ista obeležja. Svaki tip entiteta mora biti opisan svojim imenom i atributima.

Definicija 3.6. „Skup poveznika R predstavlja relaciju, u matematičkom smislu, između $n(\ge 2)$ skupova entiteta, gde je $R=\{(e_1,..., e_n) \mid e_i \in E_i, i=1,\ldots, n\}$. Pri tome skupovi E_i ne moraju biti različiti. Svaka n-torka predstavlja jedan poveznik.“ [Mogin i sar. 1996, strana 14]

Između dva ista skupa entiteta može postojati više različitih skupova poveznika, tj. relacija. Ako poveznik povezuje entitete istog skupa naziva se rekurzivnim.

Definicija 3.7. „Izraz oblika $N(E_1,...,E_n; B_1,...,B_k)$ predstavlja model skupa skupa poveznika R i naziva se tipom poveznika, ako i samo ako N predstavlja naziv skupa poveznika R, $E_1,...,E_n$ su povezani skupovi entiteta, a $B_1,...,B_k$ obeležja poveznika skupa $R=\{(e_1,...,e_n) \mid e_i \in E_i,\ i=1,\ldots,\ n\}$." [Mogin i sar. 1996, strana 15]

Model statičke strukture realnog sistema se predstavlja pomoću ER dijagrama. Tip entiteta se crta kao pravougaonik u koji je upisan naziv tipa entiteta. Tip poveznika se predstavlja rombom u koji je, takođe, upisan naziv tipa poveznika. Ukoliko tip poveznika R povezuje tipove entiteta $E_1, ...,E_n$ tada se crta po jedan neusmereni poteg od romba do svakog od entiteta E_i. Domeni obeležja se predstavljaju kao ovali sa upisanim nazivom domena i povezuju se preko usmerenog potega sa odgovarajućim tipom entiteta ili tipom poveznika. ER dijagrami se mogu crtati na dva nivoa detaljnosti u zavisnosti od toga kolika preglednost se želi postići: nivo detaljnosti naziva i nivo detaljnosti obeležja.



Slika 3.4. – Dijagram ER modela podataka

Model entiteta zove se pojava odgovarajućeg tipa entiteta. U razvoju informacionih sistema postoji jasna potreba da se entiteti jedne klase mogu razlikovati, što podrazumeva da i modeli dva entiteta budu različiti. Ako su e_1 *i* e_2 entiteti klase E, a $N(A_1,...,A_n)$ tip entiteta, tada mora važiti $(A_1,...,A_n)(e_1)\neq(A_1,...,A_n)(e_2)$. Znači mora postojati neprazan skup obeležja $X\subseteq\{A_1,...,A_n\}$ takav da je $X(e_1)\neq X(e_2)$.

Definicija 3.8. „Neka je $P=\{p_i \mid i=1,\ldots,\ k\}$ skup pojava tipa entiteta $N(A_1,...,A_n)$ a $P[X]$ restrikcija pojave p na obeležje X. Tada obeležje X predstavlja ključ tipa entiteta $N(A_1,...\ A_n)$ ako za svaki podskup P pojave tipa entiteta N, važe sledeća dva uslova:

1° $(\forall p_i, p_j \in P)(p_i \neq p_j \Rightarrow p_i[X] \neq p_j[X])$ i

2° $(\forall X' \subseteq X)(\neg 1°)$

Ako X zadovoljava samo uslov 1° naziva se superključem tipa entiteta N." [Mogin i sar. 1996, strana 19]

Uslov 1° ukazuje na jedinstvenu vrednost ključa. U skupu tipa entiteta ne smeju postojati dve pojave sa istom vrednošću ključa. Uslov 2° je uslov minimalnosti ključa koji ne poseduje suvišna obeležja. Svaki tip entiteta mora posedovati bar jedan ključ.

Jedan tip entiteta može posedovati više ključeva, koji se nazivaju ekvivalentni, a samo jedna se bira za primarni. U okviru notacije obeležja jednog ključa se podvlače sa kontinualnom linijom. Na slici 3.4. su to obeležja a_1 i a_4.

Ključ tipa entiteta u literaturi ima i drugi naziv – identifikacioni atribut, na nivou ER modela podataka. Od identifikacionih atributa se, nakon prevođenja ER modela u Relacioni model podataka, grade primarni ključevi relacionih šema.

Integritet domena atributa predstavljaju tri karakteristike: predefinisani tip podatka, dužina podatka (maksimalan broj znakova) i uslov (regularni izraz ili funkcija). Prve dve karakteristike domena atributa su obavezne komponente, a uslov nije. Specijalno ograničenje domena predstavlja nula vrednost (*Null value*) čime se određuje da je vrednost obeležja nepoznata.

Kardinalnost tipa poveznika predstavlja informaciju o prirodi odnosa između entiteta povezanih klasa. Kardinalnost poveznika je binarna relacija R između skupova dva tipa entiteta E_1 i E_2, koja se predstavlja putem dva preslikavanja: R_1: $E_1 \rightarrow P(E_2)$ i R_2: $E_2 \rightarrow P(E_1)$, gde je $P(E)$ partitivni skup skupa E. Preslikavanja R_1 i R_2 mogu da imaju semantičku interpretaciju gde je R_1 uloga entiteta iz skupa E_1 a R_2 je uloga entiteta iz skupa E_2. Ovaj pojam se odnosi na brojnost (kardinalnost) elemenata partitivnog skupa u koji se preslikava jedan element originala. Kardinalnost preslikavanja R_1 se označava $R_1(E_2(a_2,b_2))$, gde je a_2 minimalni, a b_2 maksimalni kardinalitet. Kardinalnost relacije (tipa poveznika) R se označava sa $R(E_1(a_1,b_1)$: $(E_2(a_2,b_2))$, gde je a_2 minimalni, a b_2 maksimalni kardinalitet. Parametru a se dodeljuje vrednost 0, ako se bar jedan element skupa originala preslikava u prazan skup, inače mu se dodeljuje vrednost 1. Parametru b se dodeljuje vrednost 1, ako kardinalitet slike svakog originala preslikava nije veći od 1, inače mu se dodeljuje vrednost N, gde je $1<N,M\leq|E|$. [Mogin i sar. 1996]

Ukoliko posmatramo maksimalne vrednosti kardinaliteta, tipovi poveznika se mogu podeliti u tri grupe: 1:1, 1:M i M:N. Na ER dijagramima kardinalitet tipa poveznika se predstavlja navođenjem para (a_1,b_1) i (a_2,b_2) ili samo maksimalnim vrednostima b_1 i b_2 uz grafičku predstavu odgovarajućeg tipa entiteta.

Slabi tip entiteta se na dijagramu uvodi tako što se u dvostruki pravougaonik upisuje naziv tipa entiteta, a uz grafički simbol poveznika se navodi slovo E za egzistencijalnu vezu ili I(ID) za identifikacionu vezu određenih tipova entiteta, uz poteg koji ukazuje na slabi tip entiteta. Slabi tip entiteta je takav da ne može postojati nezavisno od pojave nekog drugog tipa entiteta koji se u tom slučaju naziva jaki tip entiteta. Identifikaciona zavisnost ukazuje na to da se slabi tip entiteta ne može jedistveno identifikovati preko vrednosti nekog svog obeležja, već preko povezanosti sa jakim tipom entiteta. Kod egsiztencijalne zavisnosti slabi tip entiteta, pak, poseduje neko svoje obeležje kao primarno, tj. identifikaciono, ali će prilikom prevođenja u Relacioni model preuzeti i primarni ključ relacione šeme jakog tipa entita, tako da će se formirati složeni primarni ključ.

3.2.2. Proširenje ER modela podataka

Krajem 70-tih i početkom 80-tih godina prošlog veka, razni autori su uveli određeni broj novih, složenijih koncepata u ER model podataka, sa osnovnim ciljem da se postigne povećanje semantike pri izgradnji modela realnog sistema. Novi koncepti su: specijalizacija, generalizacija, agregacija, asocijacija i kategorizacija, a model se naziva *Enhanced Entity Relationship Data Model* (EER).

Specijalizacija i generalizacija su postupci kojim se definišu osobine skupova sličnih entiteta gde se zajednička obeležja grupišu u superklasu, a specifična obeležja, shodno različitim ulogama entiteta u odgovarajuće potklase. Veza između superklase i potklasa označavamo na dijagramu sa IS_A simbolom u okviru poveznika, a pri entitetu koji je superklasa se navodi klasifikaciono obeležje. Potklasa nasleđuje, preko vrednosti ključa, sve osobine nadklase i ne može se posmatrati izdvojeno i nezavisno od nadklase. Kardinalnost preslikavanja potklasa na skup pojave superklase je uvek (1,1), jer svakoj pojavi bilo koje potklase odgovara samo jedna pojava nadklase i ovaj kardinalitet se na dijagramima ne navodi. Ako svakoj pojavi tipa nadklase odgovara najmanje jedna pojava potklase minimalni kardinalitet je 1, u suprotnom je 0. Ukoliko neka potklasa nema jasno izdvojena specifična obeležja mora biti povezana relacijom sa nekim trećim tipom entiteta u odnosu na koju nije potklasa. Potklasa, takođe, može biti konceptom nasleđivanja (IS_A hijerarhije) povezana sa tipovima entiteta koji predstavljaje njene potklase.

Ukoliko potklasa objedinjuje pojave potpuno različitih tipova entiteta sa različitim obeležjima, tada se potklasa naziva kategorijom.

Gerund ili glagolska imenica je tip entiteta dobijen transformacijom tipa poveznika, u cilju povećanja semantike modela. Gerund se još zove i agregacija, tj. mešoviti objekat-veza i služi prevazilaženju problema u slučaju da je potrebno povezati dva poveznika koji se, prema autorima modela, ne mogu direktno povezati. Grafički se predstavlja rombom koji je upisan u pravougaonik. Koristi se u situacijama kada se pojave jednog tipa poveznika moraju povezati sa pojavama drugog tipa poveznika. Tada se neka od pojava tipa poveznika mora transformisati u gerund, tj. mešoviti objekat-veza. Gerundi ukazuju i na vremenski sled poslova, aktivnosti ili događaja iz realnog sveta koji imaju za posledicu formiranje niza međusobno povezanih tipova poveznika. Preko gerunda se, takođe, mogu prevazići greške u modelovanju koje mogu nastati u vezama ranga tri ili većim, tako što će se svesti na više relacija ranga dva.

3.3. XML I MODELI PODATAKA

XML (*eXtensible Markup Language*) je *World Wide Web Consortium* standard za razmenu podataka na Web-u. Osnovna ideja vezana za XML jeste ta da se definiše standard kojim će se razmenjivati podaci na Web-u, ali tako da se sva pažnja usmeri na sadržaj, a ne na prikaz. XML je jezik koji nema predefinisan skup ključnih reči (elemenata i atributa), već je to jezik za definisanje drugih jezika, tj. metajezik. Međutim, sam po sebi

XML nije dovoljan, već je u okviru XML tehnologije definisan skup standarda koji omogućuju njegovo korišćenje.

XML je definisan kao jezik za označavanje strukture dokumenta unutar njegovog sadržaja, čija je prvenstvena namena standardizacija strukture i sadržaja dokumenata radi njihove efektivne razmene u različitim vrstama poslovnih i drugih elektronskih komunikacija. XML je jednostavan i fleksibilan jezik koji je zasnovan na standardnom uopštenom jeziku za označavanje (*Standard Generalized Markup Language* - SGML). Organizacija za standardizaciju Web tehnologija, W3C je 1996. godine počela razvoj XML-a sa osnovnim ciljem da se dobije jezik sa kombinacijom fleksibilnosti SGML-a i jednostavnosti HTML-a. Prva verzija XML-a je definisana 1998. godine. Za razliku od HTML-a, koji prvenstveno opisuje način prikazivanja podataka u Internet stranici, XML opisuje podatke ali bez načina prikazivanja istih.

XML dokumenti su samoopisujuće platformski nezavisne tekstualne datoteke sa hijerarhijski uređenom strukturom koja se sastoji od elemenata, atributa i podataka tipa niza karaktera. Za specifikaciju tipova dokumenta se koristi *XML Schema* - XML dokument koji sadrži definicije prostih i složenih tipova, deklaracije elemenata, deklaracije atributa, modela grupa, grupa atributa, deklaracije notacija i drugo.

XML se sve više tretira kao model podataka koji sistem predstavlja kao skup međusobno povezanih tipova dokumenata, a bazu kao kolekciju međusobno povezanih dokumenata koji su pojavljivanja definisanih tipova. Koristi se za povezivanje baza podataka sa ostalim izvorima podataka dostupnim preko Interneta. Takođe može da posluži za transformaciju jednog modela podataka u drugi, jer poseduje alate i metode preko kojih se to može izvršiti.

3.4. AUTOMATSKO REZONOVANJE

Automatsko rezonovanje je jedna od osnovnih oblasti veštačke inteligencije. U ovoj oblasti rezultati koji su poslužili kao literaturni izvori, objavljeni su u [Gallaire et al 1984], [Hotomski 2003], [Berković 1997], [Tošić i sar.], [Antoniou et al., 2005].

Sistemi automatskog rezonovanja su računarski programi koji poseduju određene komponente inteligentnog ponašanja i koji se mogu primenjivati kao ljuske ekspertnih sistema, dijaloški sistemi, prevodioci prirodnih jezika, obrazovni računarski softveri, informacioni sistemi i dr. Izvođenje zaključka u ovakvim sistemima najčešće se bazira na rezolucijskoj metodi opovrgavanja, tj. negiranju tvrđenja koji se dokazuje u sistemu za automatsko dokazivanje teorema (ADT) i izvođenju zaključka iz određenog skupa pravila i činjenica. Prvobitna oblast primene ovih programa je bila isključivo matematika, ali se vremenom uvidelo da se mogu koristiti za rešavanje određenih problema i u drugim oblastima kao što su:

- sistemi bazirani na znanju;
- ekspertni sistemi;
- korektnost programa;

- upitni jezici nad relacionim bazama podataka;
- projektovanje elektronskih kola;
- generisanje programa.

Prve rezultate u ovoj oblasti su imali logičari kao što su: *Čerč, Tjuring, Markov, Erbran* i dr. Njihovi radovi, vezani za formalizaciju matematičkih teorija i logičkog izvođenja, teorije dokaza i teoriju algoritama, čine teorijsku osnovu koja je omogućila razvoj sistema automatskog rezonovanja. Prvi program ove vrste, autora *Newell, Shaw* i *Siomon*, se zvao LT (*Logic Theory Machine*) i vršio je dokazivanje teorema iskaznog računa. Sledeći program je bio *Wang Hao* i pomoću ovog programa mogle su se dokazivati i teoreme predikatskog računa. Slede 60-tih godina prošlog veka programi za dokazivanje teorema na jeziku predikatskog računa prvog reda autora: *Gilmora, Davisa, Putnama i Abrahamsa.* Ovi programi su dokazivali teorema na jeziku predikatskog računa iz «*Principia Mathematica*».

Kada je *Robinson*, 1965. godine, formulisao pravilo rezolucije, otvoren je put ka izradi još boljih programa za dokazivanje teorema. *Chang* je 1967. godine napisao progam za izvođenje posledica iz datog skupa aksioma koristeći Robinsonovo pravilo rezolucije. Važne programe u ovoj oblasti su uradili još i *Robinson, Wos*, *Allen, Luckham*. U razvoju programa za dokazivanje teorema koji se baziraju na pravilu rezolucije značajan napredak je predstavljala pojava programa za vođenje dijaloga, *Greena i Raphela* 1968. godine. Drugi važan pravac primene pravila rezolucije je bio u vezi sa proverom korektnosti programa i sa automatskim programiranjem u radovima *Waldinger, Lee*, *Manna*. Povećanje efikasnosti ovih programa postignuto je početkom 70-tih godina 20. veka uvođenjem semantičke informacije, zatim mogućnošću da se u procesu dokazivanja izvrše intervencije spolja na proces dokazivanja, ugrađivanjem specifičnih svojstava konkretnih teorija u pravila izvođenja i algoritme unifikacije. Osim metoda za automatsko dokazivanje teorema koji se baziraju na pravilu rezolucije, postoje i druge metode kao što su: metoda zasnovana na obratnom metodu *Maslova*, metoda zasnovana na prirodnom izvođenju *Šanina*, metode zasnovane na matematičkoj indukciji, sistemi zasnovani na logikama višeg reda itd.

Jezik koji se najviše koristi u formalizaciji problema koji se dokazuje sistemom za ADT jeste predikatski račun prvog reda, čiji je tvorac matematičar *Gottlob Frege*, 1879. godine. Da bi korisnici ovakvih programa mogli lakše da komuniciraju sa sistemom ne polazi se u rešavanju problema uvek od kvantifikatorske formule, već se formule predstavljaju na visokom nivou, tj. na prirodnom jeziku, a sve u cilju prilagođavanja komunikacije čoveku. Programi za dokazivanje teorema rade sa formulama predikatskog računa i u toku rada sistema formule se moraju transformisati iz jednog oblika u drugi. Pravilo rezolucije, podrazumeva da se formula iz predikatskog računa prvog reda prevodi skolemizacijom na oblik sastavaka, čime se znatno gubi na prirodnom obliku formule koji je blizak čoveku. Rezolucija je vrlo značajno pravilo izvođenja koje uz pretpostavku neograničenih prostorno vremenskih resursa, može da posluži za izvođenje dokaza svake formule koja jeste teorema. Jedan deo istraživanja u dokazivanju teorema rezolucijom jeste uvođenje dodatnih heuristika, tj. inteligentnih strategija u organizaciji pretrage, kao što je pravilo prirodne dedukcije Gentzenovog tipa koja ne zahteva

transformisanje formule na oblik sastavaka, a tvrđenje se ne negira. U sistemima za ADT se koriste i sistemi sa prirodnim izvođenjem u kojima postoji mnoštvo pravila, tako da problem predstavlja kreiranje programa koji kontroliše redosled primene tih pravila. U izvođenju se koriste empirijske metode koje su rezultat posmatranja i imitacije rada matematičara. Sistemi sa prirodnim izvođenjem zadržavaju implikaciju u formuli za razliku od rezolucije i vrše razbijanje cilja na dva ili više jednostavnijih, te koriste razne heuristike za dati domen.

Prema prilazu programi za ADT se dele na one koji se zasnivaju:

- na studiranju i simuliranju ljudskog mišljenja (predstavnici su *Newell, Shaw, Simon, Gelernter, Pastre, Brown* i *Bledsoe*);
- na čisto logičkim osnovama (predstavnici su *Herbrand, Robinson, Wos, Henchen, Kovalski, Slagle, Overbeek, Carson, Loveland* i *Luckham*).

Istraživanja prema ovim prilazima pokazala su da je oblast izuzetno kompleksna i sami programi nemaju dovoljno razvijenu strategiju za vođenje dokaza kojom bi omogućili generisanje što više pravih izvođenja, a što manje nekorisnih. Velik broj istraživanja bio je vezan za nalaženje restrikcija za primenu rezolucije, kako bi se izvođenje dokaza moglo sprovesti u raspoloživom vremenu i prostoru. U novije vreme sve se više istražuju sistemi sa nepotpunim znanjem za izvođenje zaključaka koji slede iz pretpostavki sa određenom verovatnoćom i pod određenim pretpostavkama o svetu.

Osnovni problem svih pristupa u dokazivanju teorema i izgradnji programa za ADT leži u kontrolisanju prostora pretrage, odnosno u vođenju programa ka pravim zaključcima, jer ljudi u toku dokazivanja obrađuju vrlo mali deo prostora pretrage, a dosta koriste intuiciju. Prenošenjem ovih osobina ljudi na računar znatno su unapređeni dokazivači teorema, a mogućnost obrade nepreciznog znanja dovela je do razvoja ekspertnih sistema, sistema baziranih na znanju i neuro-fuzzy sistema.

3.4.1. Predikatski račun prvog reda

U predikatskom računu prvog reda se, prema [Gallaire et al 1984], koriste sledeći primitivni simboli:

- kvantifikatori: univerzalni (∀) i egzistencijalni (∃),
- simboli za konstante (a, b, c, ...), funkcije (f, g, h, ...), promenljive (u, v, w, x, y, z, ...) i predikate (P, Q, R, ...),
- logički veznici: negacija (¬), konjukcija (∧), disjunkcija (∨), implikacija (⇒) i ekvivalencija (⇔).

Definicija 3.9. Term se definiše rekurzivno kao konstanta ili promenljiva ili ako je f n-arna funkcija i t_1, t_2,... t_n su termi, tada je i f(t_1, t_2,... t_n) term.

Definicija 3.10. Ako je R n-arni predikatski simbol i t_1, t_2,... t_n su termi, tada je R(t_1, t_2, ... t_n) atomarna formula.

Definicija 3.11. Atomarna formula ili njena negacija je literal. Dobro formirana formula je atomarna formula. Ako su W_1 i W_2 dobro formirane formule, tada su i $\neg(W_1)$, $(W_1)\wedge(W_2)$, $(W_1)\vee(W_2)$, $(W_1)\Rightarrow(W_2)$, $(W_1)\Leftrightarrow(W_2)$ dobro formirane formule.

Zatvorena dobro formirana formula je ona koja ne sadrži nijednu slobodnu promenljivu, već sadrži samo kvantifikovane promenljive, kao što je prikazano u sledećoj formuli:

$$(\forall x)\,(P(x) \Rightarrow (\exists y)\,Q(x,y)) \tag{3.1}$$

Dobro formirana formula je u preneksnoj normalnoj formi ako se svi kvantifikatori pojavljuju na početku formule:

$$(\forall x)\,(\exists y)\,(\neg P(x) \vee Q(x,y)) \tag{3.2}$$

Preneksna formula je u Skolemovoj normalnoj formi kada su svi egzistencijalni kvantifikatori eliminisani tako što su zamenjeni funkcijama koje kao argumente imaju promenljive na koje utiče egzistencijalni kvantifikator. Ove funkcije se nazivaju Skolemove funkcije. Ukoliko ove funkcije nemaju argumenata nazivaju se Skolemove konstante. Primer:

$$(\forall x)\,(\neg P(x) \vee Q(x,f(x))) \tag{3.3}$$

Klauzula je disjunkcija literala u kojoj su sve promenljive implicitno univerzalno kvantifikovane. Kada je dobro formirana formula u Skolemovoj normalnoj formi, svi kvantifikatori ispred formule mogu biti obrisani. Na primeru formule (3.3) to je:

$$\neg P(x) \vee Q(x,f(x)) \tag{3.4}$$

Postupak Skolemizacije formule predikatskog računa prvog reda se vrši u sledećim koracima:

1. Preoznačavanje promenljivih tako da različiti kvantifikatori utiču na različite promenljive.
2. Eliminisanje implikacije i ekvivalencije zamenama:
   a. $P \Rightarrow Q \leftrightarrow \neg P \vee Q$
   b. $P \Leftrightarrow Q \leftrightarrow (P \Rightarrow Q) \wedge (Q \Rightarrow P) \leftrightarrow (\neg P \vee Q) \wedge (\neg Q \vee P)$
3. Negacija se dovodi neposredno ispred predikatskih simbola primenom zamena:
   a. $\neg(\neg P) \leftrightarrow P$
   b. $\neg(P \vee Q) \leftrightarrow \neg P \wedge \neg Q$
   c. $\neg(P \wedge Q) \leftrightarrow \neg P \vee \neg Q$
   d. $\neg(\forall x)P \leftrightarrow (\exists x)\,\neg P$
   e. $\neg(\exists x)P \leftrightarrow (\forall x)\,\neg P$
   f. $(P \wedge Q) \vee R \leftrightarrow (P \vee R) \wedge (Q \vee R)$

g. $(P \vee Q) \wedge R \leftrightarrow (P \wedge R) \vee (Q \wedge R)$

4. Skolemizacija preneksne forme.
5. Formula bez kvantifikatora se dovodi u konjuktivnu normalnu formu.
6. Eliminacija simbola konjukcije i zamena tipa:

   $P \wedge Q \leftrightarrow P, Q$

Ako imamo sastavak:

$$\neg A_1 \vee \ldots \vee \neg A_m \vee B_1 \vee \ldots \vee B_n \qquad (3.5)$$

gde za n>=1, važi uslov da je jedan literal pozitivan i tada se ovakav sastavak zove klauzula, te se tada može zapisati u sledećoj formi:

$$A_1 \wedge \ldots \wedge A_m \Rightarrow B_1 \wedge \ldots \wedge B_n \qquad (3.6)$$

U klauzuli (3.6), ako je n=0 ili n=1, ista se naziva Hornova klauzula. Svaka zatvorena dobro formirana formula može se predstaviti u formi klauzula.

3.4.2. PROLOG

Najpoznatiji softver iz oblasti sistema automatskog rezonovanja je **Prolog**, autor je *Alan Kalmero,* 1971. godine. Naziv Prolog je skraćenica od engleskih reči "*PRO*(*gramming*) in *LOG*(*ic*)", što znači da je reč o programskom jeziku koji je prvenstveno namenjen logičkom programiranju.

U osnovi Prologa se nalazi predikatski račun prvog reda. Bilo je potrebno gotovo sto godina da se ovaj račun namenjen analizi formalne strukture mišljenja predstavi na računaru. Francuz *Alan Kalmero* je u Marselju 1971. godine predstavio prvu verziju Prologa pod nazivom SYSTEMQ. Nešto kasnije je usvojen i današnji naziv. Prvi interpreter je napisan na programskom jeziku Fortran IV i od 1973. godine *Kowalski* i *Kalmero* šire ideju o programiranju na Prologu. Na kongresu IFIP 1974. je šira naučna javnost upoznata sa postojanjem programskog jezika Prolog i idejom logičkog programiranja. Godine 1977. je *Warren* u Edinburgu predstavio kompajler za Prolog namenjen računaru DEC-10. Ova verzija je postala nezvanični standard, a sam programski jezik se počinje koristiti za razvoj ekspertnih sistema, raznih drugih softvera i u oblasti obrazovanja. Zbog nepostojanja zvaničnog standarda postoje razni dijalekti: PROLOG-1, micro-PROLOG, turbo-PROLOG, aritty-PROLOG itd. Prolog spada u grupu deskriptivnih programskih jezika, pri čemu programer opisuje "ŠTA" program treba da radi, a ne i "KAKO" to treba uraditi, za razliku od proceduralnih programskih jezika.

Postupak traženja rešenja u Prologu se zasniva na Robinsonovom pravilu rezolucije za ADT iz 1965. godine, sa modifikacijom ovog principa na Hornove sastavke. To su sastavci koji sadrže bar jedan pozitivan literal (literal koji ne sadrži negaciju). Ova modifikacija je izvršena zbog nedostatka rezolucijske metode pobijanja koja radi sa

sastavcima, pa je postupak izvođenja i čitanja dokaza izuzetno neprirodan čoveku, kojem je daleko bliži rad sa implikativnim zapisima.

U Prologu postoje tri vrste rečenica: činjenice, pravila i ciljevi. Pravila su formule koje sadrže implikaciju, kao u primeru (3.7):

$$P(x) \wedge P(y) => P(z) \quad (3.7)$$

Gde je P – naziv predikata; x, y, z – nazivi promenljivih. Ova formula se može zapisati u sledećem obliku:

$$P(z) :- P(x), P(y). \quad (3.8)$$

Ovakav zapis se zove klauzula, tj. pravilo oblika:

zaključak :- pretpostavka1, pretpostavka2, ..., pretpostavkaN.

gde je zaključak pozitivan literal, a pretpostavke literali sa ili bez negacije. Zaključak je glava pravila, a pretpostavke čine telo klauzule. Činjenica je prema ovome klauzula sa praznim telom, koji se sastoji samo od glave pravila. Npr. to je sledeći zapis:

$$P(a). \quad (3.9)$$

Azbuku Prologa čine velika i mala slova, brojevi i posebni znaci. Term je sintaksno prihvatljiv niz znakova. Termi mogu biti:

- konstante, koje mogu biti atomi (počinju (_), malim slovom azbuke ili brojem);
- promenljive (počinju velikim slovom);
- strukture – složeni termi koji se dobijaju vezivanjem nekoliko terma pomoću funktora.

Osnovni operatori Prologa su: konjunkcija terma (, and), disjunkcija terma (; or), vrat pravila (:-), rez (!) operator odsecanja, true – cilj koji uvek uspeva, false – cilj koji uvek propada, fail – predikat koji predstavlja konačan neuspeh, a upotrebljava se kada je potrebno dobiti sva željena rešenja; not(P) – negacija , aritmetički operatori: +, -, *, / i relacije poređenja: =, <, >, >=, <=, <>.

Tvrđenje koje se želi dokazati zove se cilj i ima oblik:

$$?-P(z). \quad (3.10)$$

Ciljevi koji sadrže samo jedan literal zovu se upiti. Ukoliko cilj ima više literala, pojedini literali kao elementi cilja moraju biti razdvojeni zarezom. Elementi ovakvog cilja se obrađuju jedan po jedan uz primenu zamene na sve ostale podciljeve. Zadovoljenje elemenata cilja svodi se na unifikaciju tog elementa sa glavom neke klauzule. Zadovoljenje početnog cilja postiže se uzastopnim generisanjem podciljeva sve dok i poslednji element podcilja ne bude zadovoljen generisanjem praznog sastavka.

Ograničenje rada Prologa sa Hornovim sastavcima, koje zahteva da glava klauzule bude pozitivan literal znači da neki rezolucijski dokazi ne mogu da se prevedu na ovaj oblik klauzula. To su dokazi pobijanjem koji bitno zavise od sastavaka koji sadrže bar dva negativna literala i ne sadrže ni jedan pozitivan literal.

Skup klauzula predstavlja bazu znanja za zadovoljenje cilja. Procedura pretraživanja omogućuje nalaženje više različitih načina zadovoljenja cilja, a ponekad i svih zadovoljenja cilja, uz intenzivnu primenu principa rekurzije.

Prolog program se piše u skladu sa sintaksom i semantikom Prolog jezika. Opisni programski jezici, poput Prologa, predstavljaju nadgradnju procedurnih programskih jezika tipa Fortran, Pascal i sl. Njihov razvoj je bio moguć zahvaljujući velikom broju već urađenih procedura. U slučaju Prolog jezika to su procedure rezolucijskog pretraživanja u procesu pobijanja. Sintaksa i semantika Prolog jezika usaglašene su sa tim procedurama, ali su i prilagođene korisniku u smislu da omogućuju pisanje Prolog programa i bez detaljnog poznavanja procedurnog deduktivnog sistema, koji na osnovu opisa datih u programu izvršava potrebne transformacije.

Bitne osobine Prologa su i poseban režim pretraživanja sa vraćanjem i rezom, rad sa strukturama podataka: listama i binarnim stablima. Prolog jezik sadrži i izvesne elemente procedurnih jezika, kao što su: aritmetičke i logičke operacije, naredbe ulaza-izlaza, rukovanje datotekama i dr. Većina verzija Prologa tretira negaciju na poseban način, koji nije u skladu sa logičkom negacijom.

Rad Prologa bazira se na principu konačnog neuspeha koji glasi: "element cilja koji sadrži negaciju, oblika notC, je zadovoljen ako svaki pokušaj zadovoljenja cilja C trpi konačan neuspeh, tj. ne postoji za C konačno stablo pretraživanja sa granom uspeha. U suprotnom, ako za C postoji konačno stablo pretraživanja sa granom uspeha, onda cilj notC trpi konačan neuspeh i ne može se zadovoljiti" [Hotomski 2003, strana 141-142]. Dakle, ako za cilj C postoji dokaz, onda je C tačan, a njegova negacija netačna, što je logički korektno. Međutim, ako se cilj C ne može dokazati, onda se negacija tog cilja C usvaja kao tačna! Ovo poslednje ne odgovara logičkom tretmanu negacije i dovodi do nekorektnih odgovora Prolog sistema. Da bi se izbegli nekorektni odgovori, uvode se razna ograničenja na upotrebu negacije koja ipak ne eliminišu u potpunosti mogućnost generisanja nekorektnih odgovora.

3.4.2.1. Kvantifikatori u Prologu

Pravila zaključivanja u Prologu su klauzule koje se sastoje od glave i tela pravila, povezanih vratom (3.8). Glava pravila je zaključak, a telo su pretpostavke koje impliciraju zaključak. Pošto su činjenice klauzule sa praznim telom koje ne sadrže promenljive (3.9), već predikate sa konstantama kao argumentima, kao takvi nisu podložni kvantifikaciji, koja se pak odnosi na promenljive u rečenicama.

Univerzalni (∀) kvantifikator se, u Prologu, predstavlja na sledeći način – Ukoliko se tvrdi da je za svaku vrednost promenljive *x* osobina *P* tačna, tada se ova izjava zapisuje u formi predikatskog računa prvog reda kao:

$$(\forall x)\ P(x) \tag{3.11}$$

što je ekvivalentno Prolog rečenici:

$$P(X). \tag{3.12}$$

U slučaju da u izjavi postoji implikacija osobina promenljivih – za svako *x*, ako *x* ima osobinu *P*, sledi da *x* ima i osobinu *Q*, tada predikatska forma glasi:

$$(\forall x)\ (\ P(x) \Rightarrow Q(x)\ ) \tag{3.13}$$

i tada Prolog klauzula ima sledeći oblik, prema (3.8):

$$Q(X) :- P(X). \tag{3.14}$$

Svaka promenljiva koja se pojavljuje u glavi pravila, kao *x* u primeru *Q*(*x*), je univerzalno kvantifikovana prema [9], [10].

Primeri upotrebe univerzalnog kvantifikatora u predikatskim i Prolog rečenicama:

$$(\forall x)(\forall y)(\forall z)\ (\ R(x,z) \wedge R(z,y) \Rightarrow P(x,y,z)\ ) \leftrightarrow P(X,Y,Z) :- R(X,Z),\ R(Z,Y). \tag{3.15}$$

Egzistencijalni (∃) kvantifikator se koristi u Prolog jeziku tako što se u telu pravila nalazi promenljiva koja nije prisutna u glavi, te je tada takva promenljiva kvantifikovana egzistencijalno [9], [10], [11].

Ukoliko se tvrdi da postoji promenljiva *y* takva da za nju važi osobina *Q* koja je tačna, tada se piše:

$$(\exists y)\ Q(y) \tag{3.16}$$

U slučaju rečenice koja sadrži logičku implikaciju – postoji *y* sa osobinom *Q* takvo da iz toga sledi da za svako *x* važi osobina *P*. Tada dobijamo sledeću predikatsku rečenicu:

$$(\exists y)\ Q(y) \Rightarrow (\forall x)\ P(x) \leftrightarrow P(X) :- Q(Y). \tag{3.17}$$

U ovoj klauzuli je promenljiva *y* egzistencijalno kvantifikovana.

Primeri upotrebe egzistencijlnog kvantifikatora u predikatskim i Prolog rečenicama:

$$1.\ (\forall x)(\exists y)\ (\ Q(x,y) \Rightarrow P(x)\ ) \leftrightarrow P(X) :- Q(X,Y). \tag{3.18}$$

$$2.\ (\forall x)(\forall y)\ (\exists z)\ (\ R(x,z) \wedge R(z,y) \Rightarrow P(x,y)\ ) \leftrightarrow P(X,Y) :- R(X,Z),\ R(Z,Y). \tag{3.19}$$

4. PREGLED VLADAJUĆIH STAVOVA I REŠENJA U PODRUČJU ISTRAŽIVANJA

Korektnost modela podataka je jedan od aspekata kvaliteta podataka (*Data Quality* - DQ) kao generalnog koncepta. Loš model podataka utiče na efikasnost i efektivnost finkcionalnosti organizacionih sistema. Jedan od primera uticaja kvaliteta podataka dat je u izveštaju *Data Warehousing* Instituta iz 2002. godine u kojem je prikazano da su problemi kvaliteta podataka koštali poslovanje u Sjedinjenim Američkim Državama više od 600 milijardi dolara godišnje.

Razlika u tumačenjima i primeni vezanoj za kvalitet podataka u mnogim organizacijama podstakla je razvoj istraživanja u ovoj oblasti. Doprinosi koje se odnose na kvalitet podataka realizovani su u okviru teorijskih istraživanja i praktičnih rešenja u oblasti baza podataka i informacionih sistema [Batini et al., 2006]. Nekoliko međunarodnih konferencija bave se kvalitetom podataka kao glavnom temom, počev od *The International Conference on Information Quality*, održanoj na MIT-u 1996. godine. Istraživanja u ovoj oblasti su pokazala da postoji potreba za razvojem odgovarajućih tehnika, metodologija i alata. Literatura prikazuje fragmentirane rezultate pre svega uz formalni i istraživačko-orjentisani pristup, često uz primenu u ograničenim specifičnim oblastima. Istraživači [Batini et al., 2006] i [Wang et al., 2004] opisuju opšti pristup kvalitetu podataka, kombinujući rezultate različitih projekata i istraživačkih grupa. Mnogi alati koji se odnose na kvalitet podataka su dostupni na tržištu i koriste se od strane raznih "*Data Driven*" aplikacija, kao što je "*Data Warehousing*", kako bi unapredili kvalitet modela podataka i poslovnih procesa. Često je obuhvat tih alata ograničen i domenski zavisan.

Kvalitet podataka je kompleksna i multidisciplinarna oblast istraživanja, koja se zasniva na drugim istraživačkim oblastima kao što su: statistika, prezentovanje znanja i rezonovanje, data mining, menadžment informacioni sistemi, integracija podataka. U [Batini et al., 2006] su izdvojeni problemi istraživanja u ovoj oblasti. Neki od najvažnijih dimenzija kvaliteta podataka koje se koriste u metrikama uključuju:

- tačnost/ ispravnost (*Accuracy*),
- kompletnost (*Completeness*),
- vremenski zavisne dimenzije:
    - opšta primenljivost/proširenost/protok (*Currency*),
    - pravovremenost (*Timeliness*),
- izmenljivost (*Volatility*),
- konzistentnost (*Consistency*),
- pristupačnost (*Accessibility*),
- kvalitet izvora podataka (*Quality of Information Sources*),
- dimenzije kvaliteta šeme
  (čitljivost - *Readability*, normalizacija - *Normalization*).

Glavni problemi istraživanja u ovoj oblasti odnose se na: dimenzije kojima bi se merio nivo kvaliteta modela podataka, modele merenja kvaliteta podataka, tehnike merenja i unapređenja (algoritmi, heuristike, procedure bazirane na znanju i proces učenja), metodološke okvire za razvoj alata za merenje i unapređenje, koji treba da obezbede uputstva za izbor najefektivnijih tehnika i alata za merenje i unapređenje kvaliteta podataka. Otvoreni problemi u oblasti metodologija kvaliteta podataka odnose se na: validaciju metodologija, razvoj alata koji omogućavaju primenljivost metodologija, eksperimentalnu validaciju metodologija, proširenje metodoloških uputstava na širi skup dimenzija koje se mere i zavisnosti između dimenzija, kompoziciju kvaliteta, gde se podacima upravlja na različitim nivoima agregacije, razvoj efektivnije metodologije vrednovanja, gde se kvalitativni i kvantitativni indikatori uzimaju u obzir. [Batini et al., 2006]

Kvalitativne metrike se određuju za ključne faktore koji utiču na kvalitet modela. Moody navodi faktore: kompletnost, integritet, fleksibilnost, razumljivost, korektnost, jednostavnost, integracija, mogućnost implementacije [Moody 1998]. Autori [Piprani et al., 2008] navode faktore: tačnost, kompletnost, konzistentnost, preciznost, pouzdanost, vremenska usklađenost, pravovremenost, jedinstvenost, validnost.

Kesh definiše metrike kvaliteta modela u odnosu na ontološku i bihejvioralnu komponentu. Ontološke komponente se odnose na strukturu (pogodnost, stabilnost, konzistentnost) i sadržaj (kompletnost, kohezivnost, validnost). Bihejvioralna komponenta se odnosi na: upotrebljivost sa korisničkog i dizajnerskog aspekta, mogućnost održavanja, tačnost i performanse. [Kesh 1995]

Autori [Yücesan et al., 2002] navode praktične tehnike i smernice za verifikaciju i validaciju modela. Cilj verifikacije i validacije je model koji je korektan kada se koristi za predviđanje performansi u realnom svetu sistema koji se predstavlja, ili da predvidi razliku u performansama između dva scenarija ili dve konfiguracije modela. Dati su primeri velikog broja tipičnih situacija u kojima projektanti modela mogu da daju neodgovarajuće ili netačne pretpostavke i nude smernice i tehnike za izvođenje verifikacije i validacije. Upotreba jednostavne tehnike navedene u ovom radu može da pomogne u izbegavanju velikih i ozbiljnih grešaka u modelovanju.

Van Belle daje okvir za ocenu raznih vrsta modela sa sintaksnog, semantičkog i pragmatičkog aspekta kroz kriterijume i predložene metrike za ocenu modela u odnosu na dati kriterijum (Tabela 4.1). [Van Belle 2006]

Tabela 4.1: Van Belleov okvir za evaluaciju modela [Van Belle 2006]

ASPEKT	KRITERIJUM (METRIKE)
Sintaksni aspekt	Veličina (broj koncepata)
	Korektnost, nepostojanje grešaka, integritet, konzistentnost, usklađenost sa standardima
	Modularnost (broj grupa dijagrama, broj nivoa dijagrama)
	Struktura, hijerarhija (stepen ponovne iskoristivosti - *reuse ratio*)
	Kompleksnost, gustina (*density*)
	Arhitekturni stil (estetika izgleda modela)
Semantički aspekt	**Generičnost-*genericity* (usklađenost sa domenom)**
	Pokrivenost (pokrivenost domena i osnovnih koncepata)
	Kompletnost (stepen pokrivenosti rečnika - *lexicon*)
	Efikasnost, konciznost
	Ekspresivnost
	Sličnost i preklapanje sa drugim sličnim modelima
	Razumljivost, čitljivost
	Dokumentacija (kompletnost, proširivost, čitljivost)
Pragmatički aspekt	Validnost, prihvaćenost od strane klijenta i korisnika
	Fleksibilnost, proširivost, adaptibilnost
	Zrelost, usklađenost sa sadašnjim stanjem (*currency*)
	Usklađenost sa svrhom, ciljem, relevantnost, da odgovara sistemu
	Dostupnost (*availability*)

4.1. EVALUACIJA KVALITETA MODELA PODATAKA

Kvantitativne metrike konceptualnih modela podataka [Piattini et al., 2000] se zasnivaju na broju entiteta, atributa i relacija. Razlikuju atribute sa različitim karakteristikama: funkcionalno zavisnih atributa, izvedenih atributa, redundanse, kompozitnih atributa, kao i različite vrste relacija (M:N, IS-A) itd.

Moody je definisao skup metrika, prikazan u tabeli 4.2, za evaluaciju kvaliteta dijagrama ER modela podataka. [Moody 1998]

Tabela 4.2: Metrike za evaluaciju faktora kvaliteta dijagrama ER modela podataka [Moody 1998]

FAKTOR KVALITETA	METRIKA
Kompletnost	Broj stavki modela ne odgovara zahtevima korisnika. Broj zahteva koji su prisutni u modelu podataka. Broj stavki modela odgovara zahtevima korisnika ali nije tačno definisan. Broj nekonzistentosti sa modelom procesa.
Integritet	Broj poslovnih pravila koji nisu primenjeni u modelu podataka. Broj ograničenja integriteta uključenih u model podataka koji ne odgovaraju poslovnoj politici.
Fleksibilnost	Broj elemenata u modelu koji su predmet izmena u budućnosti. Očekivani troškovi izmena. Strateška vežnost izmena.
Razumljivost	Korisničko vrednovanje razumljivosti modela. Sposobnost korisnika da interpretiraju model korektno. Razumljivost kod razvoja aplikacija.
Korektnost	Broj narušavanja konvencija modelovanja podataka. Broj narušavanja normalnih formi. Broj slučajeva redundanse u modelu.
Jednostavnost	Broj entiteta. Broj entiteta i relacija. Suma izraza: (NE + NR + NA), gde je NE broj entiteta, NR je broj relacija, and NA je broj atributa.
Integracija	Broj konflikata sa drugim modelima podataka. Broj konflikata sa postojećim sistemom. Rejting predstavnika svih poslovnih oblasti.
Primenljivost	Rejting tehničkog rizika. Rejting rizika rasporeda. Procena troškova razvoja. Broj elemenata uključenih u fizički nivo modela podataka.

Istraživači [Gray et al., 1991] su predložili ciljeve i metrike za evaluaciju kvaliteta ER dijagrama. Cilj ovih metrika je bio da dizajnerima omogući podršku za pomoć u poređenju alternativa u dizajnu kako bi doneli najbolju odluku. Ove metrike, takođe, omogućuju utvrđivanje problema u projektovanju baza podataka, kako bi se povećao njihov kvalitet. Metrike su podeljenje u tri grupe: kompleksnost ER dijagrama, kompleksnost entiteta i kompleksnost arhitekture podataka, tj. atributa nekog entiteta.

1. Kompleksnost ER dijagrama se definiše kao:

$$E = \sum_{i=1}^{n} (E_i)^c \tag{4.1}$$

gde je: $c>1$, a n je broj entiteta.

2. Kompleksnost entiteta *Ei* se definiše kao:

$$E_i = D_i * F_i \tag{4.2}$$

gde je: *Di* – kompleksnost arhitekture podataka (atributa), a *Fi* – funkcionalna kompleksnost.

3. Kompleksnost arhitekture podataka nekog entiteta *i* je:

$$D_i = R_i * (a * FDA_i + b * NFDA_i) \tag{4.3}$$

gde je: $0<a<=b$, *Ri* - broj relacija, *FDAi* - broj funkcionalno zavisnih atributa, *NFDAi* - broj atributa koji nisu funkcionalno zavisni.

Autori [Piattini, Polo et al., 2000] su predložili skup objektivnih metrika za evaluaciju strukturne kompleksnosti dijagrama ER modela podataka, sa ciljem da izgrade sistem za predviđanje održivosti ovog modela.

1. RvsE metrika za merenje relacija koje postoje između entiteta:

$$RvsE = \left(\frac{N^R}{N^R + N^E} \right)^2 \tag{4.4}$$

gde je: N^R broj relacija u ER dijagramu, N^E je broj entiteta, $N^R + N^E > 0$. U N^R spadaju i relacije IS_A hijerarhije, po jedna ka svakom podtipu entiteta.

2. DA metrika za određivanje broja deljenih atributa u ER dijagramu:

$$DA = \frac{N^{DA}}{N^A - 1} \tag{4.5}$$

gde je: N^{DA} je broj deljenih atributa u ER dijagramu, N^A je broj atributa u ER dijagramu, $N^A>1$. U N^A su uključeni jednostavni atributi, kompozitni atributi, ali i atributi sa višestrukim atributima.

3. CA metrika za ocenjivanje broja kompozitnih atributa u poređenju sa ukupnim brojem atributa u ER dijagramu:

$$CA = \frac{N^{CA}}{N^A} \tag{4.6}$$

gde je: N^{CA} je broj kompozitnih atributa u ER dijagramu, N^A je broj atributa u ER dijagramu, $N^A>1$. U N^A su ključeni jednostavni atributi, kompozitni atributi, ali i atributi sa višestrukim atributima.

4. RR metrika za broj redundantnih relacija u ER dijagramu:

$$RR = \frac{N^{RR}}{N^R - 1} \tag{4.7}$$

gde je: N^{RR} broj redundantnih relacija u ER dijagramu, N^R broj relacija u ER dijagramu, $N^R>1$. U N^R spadaju i relacije IS_a hijerarhije, po jedna ka svakom podtipu entiteta.

5. M:NRel metrika za broj relacija kardinaliteta M:N u ER dijagramu:

$$M:N\,\mathrm{Re}l = \frac{N^{M:NR}}{N^{R}} \tag{4.8}$$

gde je: $N^{M:NR}$ broj relacija kardinaliteta M:N u ER dijagramu, N^{R} broj relacija u ER dijagramu, $N^{R}>0$. U N^{R} spadaju i relacije IS_A hijerarhije, po jedna ka svakom podtipu entiteta.

6. IS_ARel metrika ocenjivanje kompleksnosti veze generalizacije/specijalizacije u ER dijagramu:

$$FLeaf = \frac{N^{Leaf}}{N^{E}} \tag{4.9}$$

gde je: N^{Leaf} broj podklasa u ER dijagramu, N^{E} broj nadklasa u ER dijagramu, $N^{E}>0$. U N^{R} spadaju i relacije IS_A hijerarhije, po jedna ka svakom podtipu entiteta.

7. IS_ARel metrika se koristi za ocenjivanje kompleksnosti svih IS-A hijerarhija i izračunava se sledećom formulom:

$$Is\_A\,\mathrm{Re}l = FLeaf - \frac{FLeaf}{ALLSup} \tag{4.10}$$

Nakon kreiranja ER modela podataka, CASE alati omogućavaju automatsko transformisanje modela u relacioni model podataka na osnovu definisanih pravila za prevođenje. Ali, dok transformišu ER u relacioni model, nisu u stanju da pronađu greške koje dovode do problema normalizacije šema [Bock 1997], te je neophodno da se vodi računa o određenim tipovima grešaka u ER modeliranju, u cilju da se izbegnu buduće greške normalnih formi u relacionom modelu podataka.

S obzirom da su istraživanja vezana za kvalitet modela podataka [Gray et al., 1991], [Moody 1998], [Piattini et al., 2000], [Piattini, Polo et al., 2000] teoretskog karaktera, grupa istraživača [Genero et al., 2000] je izvršila analizu i validaciju ovih metrika u kontrolisanom eksperimentu. Zaključili su da većina metrika nije dovoljno teoretski i empirijski proverena i da autori nisu definisali nijednu metodologiju za njihovu primenu. Oblast merenja kvaliteta konceptualnog modela podataka još uvek nije dovoljno čvrsto utemeljena. Potrebno je definisati validne metrike koje mogu biti od koristi projektantima baza podataka.

Kao što je navedeno, u cilju razumevanja faktora koji određuju održivost konceptualnih modela podataka i poboljšanje procesa modelovanja konceptualnih modela podataka potrebno je znati procenjivati osobine konceptualnih modela podataka i njihov kvalitet na objektivan način. Nedostatak rano dostupnih i temeljno proverivih instrumenata za proveru metrika konceptualnog modela podataka motivisalo je istraživače [Genero et al., 2003] da definišu skup metrika za ER dijagrame. Prikazali su ciljeve i lako izračunljive metrike, merljiva unutrašnja svojstva ER dijagrama koja su povezana sa

njihovom strukturnom kompleksnošću i koja mogu biti upotrebljena kao mere za održavanje modela.

Dva glavna pristupa u obučavanju konceptualnom modelovanju su poređena u istraživanju [Batra et al., 2004]. Prvi je zasnovan na pravilima, dok je drugi zasnovan na obrascima. Eksperimentalno je pokazano da složeniji zadaci utiču na slabije performanse dizajna. Takođe je pokazano da pristup zasnovan na pravilima nije značajno bolji od modela zasnovanog na obrascima, ali pristup zasnovan na pravilima za početnike dizajnere daje znatno bolje performanse u dva od tri nivoa složenosti.

Kvalitet modela podataka odražava kvalitet podataka u bazama podataka i aplikacijama poslovne inteligencije. Što se tiče kvaliteta modela podataka, postoje dva različita pristupa: evaluacija u toku procesa kreiranja modela podataka i evaluacija već gotovih modela podataka. Modeli podataka koje su [Batini et al., 2006] uključili u evaluaciju kvaliteta su: strukturirani (ER, Relacioni, Objektno-orijentisani) i polu-strukturirani modeli podataka (XML). Posebno su istraženi aspekti opisa porekla elementarnog podatka i procesa putem kojih je neki podatak dospeo u bazu podataka.

Aktivnost evaluacije kvaliteta podataka predstavlja svaki proces koji se pomoću različitih tehnika izvršava direktno nad podacima kako bi se unapredio njihov kvalitet, odnosno da bi se izmerila i unapredila dimenzija kvaliteta podataka. Ove aktivnosti obuhvataju:

- preuzimanje/akviziciju podataka,
- standardizaciju/normalizaciju,
- identifikaciju objekata,
- integraciju podataka,
- proveru poverenja u izvora podataka,
- kompoziciju kvaliteta, lokalizaciju grešaka (koristeći skup semantičkih pravila),
- korekciju grešaka,
- optimizaciju troškova,
- usklađivanje šeme,
- čišćenje šeme,
- profilisanje,
- izdvajanje osobina iz podataka.

Metodologije određivanja kvaliteta podataka mogu se klasifikovati u skladu sa različitim kriterijumima [Batini et al., 2006]:

- *Data-driven vs. Process-driven* metodologije,
- Metodologije orjentisane na merenje kvaliteta podataka naspram metodologija za unapređenje (*Measurement vs. Improvement*),
- Metodologije opšte namene (širok spektar aktivnosti, dimenzija itd.) prema metodologijama specijalne namene (specifične aktivnosti ili dimenzije),
- Intra-organizacione naspram inter-organizacionim metodologijama – merenje i unapređenje odnose se na konkretnu organizaciju ili grupu organizacija, proces ili bazu podataka i svaki od ovih tipova metodologija je domenski zavisna.

Jedan od načina da se poboljšaju mogućnosti informacionog sistema je da se razmotri kvalitet konceptualnog modela, kao i njegovo funkcionalno ponašanje. Kvalitet konceptualnog modela može da se definiše kao skup razumljivih karakteristika izraženih kroz merljive parametre koji mogu biti objektivni i/ili subjektivni. Cilj empirijskog istraživanja [Cherfi et al., 2007] je bio da se proceni i uporedi kvalitet različitih verzija konceptualnih modela istog univerzuma. Ovo istraživanje opisuje:

a) skup pokazatelja (jasnost, jednostavnost, izražajnost, minimalizacija) primenjenih na različite verzije konceptualne šeme ER modela,
b) okvir koji omogućava sveobuhvatno poređenje konceptualnih šema,
c) eksperimentalna vođenja koja dovode do procene istih šema od strane korisnika informacionog sistema, kao što su dizajneri, krajnji korisnici i studenti,
d) poređenje objektivne i subjektivne procene zasnovane na uzorku od oko 120 zapažanja koristeći različite statističke metode.

U oblasti evaluacije procesa kreiranja modela podataka, neka eksperimentalna istraživanja su sprovedena u vezi sa poređenjem ER modelovanja i objektno-orijentisanog modelovanja [Shoval 1997], [De Lucia et al., 2009] i [Aguirre-Urreta et al., 2008]. Pokazan je značaj ER modelovanja i istaknuta njegova prednost u odnosu na objektno-orijentisani pristup. Pokazano je da ER modelovanje ima neke prednosti u odnosu na objektno-orijentisani pristup:

- Veze između entiteta ranga većeg od dva se lakše razumeju i formiraju u ER modelu.
- Međusobno povezane kompleksne relacije se lakše kreiraju u ER modelu.
- Potrebno je značajno manje vremena za dizajniranje ER šema nego dijagrama klasa.
- Projektani i dizajneri baza podataka jednostavno više koriste ER model.

Referencijalni integritet je ograničenje od suštinskog značaja u relacionim bazama podataka, koje održava bazu podataka u kompletnom i konzistentnom stanju. Autori [Ordonez et al., 2008] pretpostavljaju da baze podataka mogu da krše referencijalni integritet i relacije mogu biti denormalizovane. Predlaže se skup kvalitetnih metrika, definisanih u četiri nivoa: baza podataka, relacija, atribut i vrednost, koja meri kompletnost i konzistenciju referencijalnog integriteta. Kvalitet pokazatelja se efikasno izračunava sa standardnim SQL upitima, koji sadrže dve optimizacije upita: levi spoljašnji spoj na stranim ključevima i grupisanje stranih ključeva. Autori su u eksperimentalnom istraživanju predložili metrike za optimizacija SQL upita na stvarnim i sintetičkim bazama podataka. Pokazana je mogućnost otkrivanja i objašnjavanja grešaka u referencijalnom integritetu.

Sveobuhvatno istraživanje [Moody 2005] prikazuje analizu predloženih rešenja za evaluaciju konceptualnog modela podataka:

- Istraživanje je obuhvatilo 50 različitih pristupa evaluaciji konceptualnog modela podataka koji su objavljeni, od kojih je manje od 20% empirijski provereno.
- Nijedno od predloženih rešenja nije prihvaćeno u praksi (van okvira istraživanja).

- Predložena rešenja su na različitom stepenu opštosti i teška su za praktičnu primenu, dok se malobrojna praktična rešenja fokusiraju na pojedinim elementima modelovanja.
- Predložena rešenja pokazuju da postoji nedostatak utemeljene terminologije, povezanosti oblasti istraživanja i primene standarda, metrika za merenje i procedura za evaluaciju, uputstava za poboljšanje procesa evaluacije. Rešenja su usmerena uglavnom na otkrivanje grešaka u modelovanju, a nedostaje utvrđivanje kvaliteta softverskog proizvoda, kao i empirijske studije koje su realizovane u praksi.
- Empirijska testiranja su uglavnom vršena u laboratorijskim uslovima sa studentima, što povećava dilemu njihove generalizacije na praktičnom polju. Ostala empirijska istraživanja uključuju saradnju istraživača i stručnjaka iz ove oblasti u okviru praktičnih projekata u evaluaciji konceptualnog modela podataka.

4.2. TESTIRANJE KOREKTNOSTI MODELA PODATAKA

U istraživanju [Formica et al., 2004], sprovedenom u okviru projekta «*Progetto Finalizzato Sistemi Informatici e Calcolo Parallelo*», u Italiji, izvršena je provera korektnosti IS-A hijerarhija u šemama objektno-orijentisanih baza podataka. IS-A hijerarhija je koncept modelovanja koji obezbeđuje mehanizam nasleđivanja. U IS-A hijerarhiju je uključena semantika, kao skup objekata koji predstavljaju proširenje ekstenzija datog tipa koji je podskup svih ekstenzija nadtipova. U rešavanju problema je upotrebljen objektno-orijentisani jezik za definisanje fizičke strukture *TQL*.

U istraživanju [Sotnykova 2006], sprovedenom u Francuskoj, razmatran je problem integracije baza podataka tako što je uveden nov metod koji kombinuje funkcionalnost tehnika konceptualnog modelovanja baza podataka sa funkcionalnošću logički zasnovanih dokazivača. Razrađen je postupak za prostorno-vremensku integraciju informacija na nivou relacione šeme.

Prema istraživanju koje je urađeno u Brazilu [Emer et al., 2008], testiranje šema relacione baze podataka izvršeno je sa ADIA (*Alternative Data Instance Analyses*) sistemom. ADIA je pristup testiranju relacione šeme baziran na greškama u ograničenjima koja se odnose na elemente, tj. entitete i atribute. U ADIA sistem testiranja se unose: model podataka, klase sa greškama, formalna prezentacija modela, proces testiranja i kriterijumi za testiranje. Model podataka se predstavlja metamodelom *M* opisanim u XML notaciji. Prednosti ovog pristupa su ta što šema može biti testirana i ako baza podataka nije dostupna, a test podaci se generišu automatski i generisana instanca baze podataka se može koristiti za testiranje aplikacija baze podataka.

Formalna prezentacija relacionog modela podataka, prema [Emer et al., 2008], je *S = (E, A, R, P),* gde je:

E – konačan skup entiteta,
A – konačan skup atributa,
R – konačan skup ograničenja koja se tiču domena, definicija, relacija i semantike pridružene entitetima i atributima,
P – konačan skup relacija između entiteta, atributa i ograničenja.

Istraživanje [Batra et al., 2001], predstavlja greške konceptualnog modelovanja kao ljudske greške na tri nivoa: bazirane na veštinama, zasnovane na pravilima i zasnovane na znanju. Poseban softverski prototip - CODA je realizovan za potrebe konsultantske podrške konceptualnom dizajnu baze podataka za dizajnere početnike. U toku eksperimentalnih istraživanja sa studentima, pokazano je da se korišćenjem ovog softvera može unaprediti kvalitet modela podataka. Ovaj softverski sistem sadrži heuristike i pravila za prepoznavanje tipičnih grešaka u ER modelovanju.

Metod za podršku odlučivanju za evaluaciju dizajna baze podataka je objavljen u radu [Eessaar et al., 2012]. Metod je zasnovan na modelu "*Analytic Hierarchy Process*" koji omogućuje donošenje odluka u modelovanju složenih problema i hijerarhijskih struktura. Autori su demonstrirali upotrebu metode tako što su poredili četiri dizajna baze podataka na niovu modela uz pomoć PostgreSQL sistema za upravljanje bazama podataka. Dodatni rezultat jeste skup kriterijuma koji se može koristiti za evaluaciju dizajna SQL baza podataka.

4.3. ONTOLOGIJE I MODELI PODATAKA

Ontologija, koja više i u potpunosti [Ullman et al., 2002] opisuje znanja za konkretan problemski domen, se koristi za poređenje sa dizajniranim modelom podataka, tako da se model može ocenjivati i sa semantičkog aspekta. Glavna razlika između ontologije i šeme baze podataka je u tome što je šema obično ograničena na opis manjeg podskupa «mini-sveta» iz realnog sveta u cilju memorisanja i upravljanja podacima. Ontologija se najčešće smatra opštijom, jer nastoji da opiše deo realnosti ili oblast od interesa što je moguće potpunije.

Razlike između ontologija i različitih modela za projektovanje softverskih rešenja za organizacije i sisteme su izložena u nizu istraživanja sumiranih u [Atkinson et al., 2006]:

- Modeli su fokusirani na realizaciji, dok ontologije nisu. [Noy et al., 2001]
- Ontologije mogu podržavati rezonovanje, što ne važi za modele. [Cali et al., 2002]
- Ontologije su formalne, dok modeli to nisu. [Baclawski et al., 2002], [Brockmans et al., 2004], [Đurić et al., 2005]
- Ontologije predstavljaju informacije na Web-u, dok modeli ne. [Kifer et al., 1995], [Frankel et al., 2004]
- Modeli se koriste u „zatvorenom svetu“, a ontologije u „otvorenom svetu“.

- Ontologije se mogu koristiti za eksploataciju znanja u izvršnom režimu rada, a modeli ne. [17], [Hoessler et al., 2004]

Osnovna razlika između ontologije i šeme baze podataka je ta da je šema baze podataka obično ograničena na opisivanje malih podskupova mini-sveta iz stvarnosti sa ciljem skladištenja i upravljanja podacima. Ontologije pokušavaju da opišu deo realnog sveta ili domena interesovanja. [Formica et al., 2006]

[Kesh 1995] je razvio metode za vrednovanje kvaliteta ER dijagrama. Autor smatra da kvalitet modela podataka zavisi od ontoloških i bihejviorističkih komponenti. Ceo metod se može sumirati u tri koraka:

1. Izračunavanje rezultata/vrednosti za individualne ontološke komponente, kako za strukturne komponente (relacije između elemenata koje čine model), tako i za sadržinske komponente (atributi entiteta).
   Strukturne komponente su:
   a. pogodnost (O1),
   b. korektnost (O2),
   c. konzistentnost (O3),
   d. konciznost (O4).
   Sadržinske komponente:
   a. kompletnost (O5),
   b. povezanost (O6),
   c. proverljivost (O7).
2. Izračunavanje rezultata/vrednosti za relevantne ontološke komponente za svaku bihejviorističku komponentu:
   a. upotrebljivost sa aspekta korisnika (S1),
   b. upotrebljivost sa aspekta projektanta (S2),
   c. održivost (S3),
   d. preciznost (S4),
   e. performanse (S5).
3. Kombinovanje rezultata za bihejviorističke i ontološke komponente i izračunavanje kvaliteta modela podataka:

   $$Q = W1*S1 + W2*S2 + W3*S3 + W4*S4 + W5*S5 \quad (4.4)$$

   gde su: W1, W2, ..., W5 težine bihejviorističkih faktora a S1, S2, ..., S5 bazirani na kombinaciji vrednosti ontoloških faktora:

   S1 = (O1 + O3 + O4 + O5) / 4
   S2 = (O2 + O3 + O5 + O6 + O7) / 5
   S3 = (O2 + O4 + O6) / 3
   S4 = (O3 + O5) / 2
   S5 = (O4 + O5) / 2

U Kesh-ovom pristupu, ontološki faktori imaju vrednosti od 1 do 5. Priroda ovih metrika je takva da zahteva interakciju dizajnera baze podataka, korisnika i menadžera

koji vrednuju model. Nakon praktične primene modela, Kesh je zaključio da model obezbeđuje koristan okvir za analizu i reviziju dijagrama ER modela podataka, koji po njemu mogu biti čak i veliki dijagrami ER modela.

U okviru istraživanja [El-Ghalayini et al., 2005] su ispitane mogućnosti generisanja domenskog konceptualnog modela na osnovu date ontologije sa vizijom da se konceptualni model transformiše u globalni model koji integriše podatake iz većeg broja informacionih izvora. Izvedena su pravila, bazirana na ontološki izvedenoj semantici BWW modela, koja mapiraju elemenete ontološkog jezika (DAML+OIL) u elemente domenskog konceptualnog modela. Prezentovana je mogućnost kreiranja konceptualnog modela podataka na osnovu ontologije.

Vysniauskas sa koautorima opisuje generisanje relacione šeme na osnovu ontološke OWL specifikacije [Vysniauskas et al., 2006]. Međutim, kreiranje ontologije i generisanje modela podataka na osnovu ontologije u okviru ontoloških alata nezavisno je od rezultata rada CASE alata kojim se integriše proces razvoja informacionog sistema. Ovaj proces obuhvata sve aktivnosti od modelovanja poslovnih procesa i modelovanja podataka do dizajna arhitekture sistema i softvera. Vysniauskas dovodi u pitanje zasnovanost kreiranih ontologija i modela podataka. Javlja se potreba za povezanošću ontoloških alata i CASE alata, čime bi se prevazišao navedeni problem.

Mapiranje ontologije u elemente konceptualnog modela podataka su izvršili [El-Ghalayini et al., 2005] tako što su predložili inicijalni skup pravila za mapiranje kako bi se generisao konceptualni model podataka na osnovu ontologije korišćenjem BWW ontološke konstrukcije za povezivanje elemenata. Transformaciona pravila su:

1. pravilo: "stvar" predstavlja pojavu objekta u ontološkom jeziku (*Individual*), kao i entitet u konceptualnom modelu. Znači objekat iz ontološkog jezika se mapira u entitet u konceptualnom modelu.
2. pravilo: pošto klasa u ontološkom jeziku i tip entiteta u konceptualnom modelu, oba predstavljaju klasu, tj. vrstu neke prirodne stvari, autori su zaključili da svaka imenovana klasa predstavljena u ontološkom jeziku se mapira u tip entiteta pod određenim ograničenjima.
3. pravilo: svaka osobina (*daml: DatatypeProperty*) koja se koristi za definisanje klase u ontološkom jeziku predstavlja atribut entiteta u konceptualnom modelu podataka u skladu sa ograničenjima (*Global/Local Property Constraints*) koja se odnose na osobinu.
4. pravilo: relacija nadklasa/podklasa u ontološkom jeziku može se definisati kao relacija između entiteta u konceptualnom modelu podataka. Pošto se svaka ontološka klasa mapira u tip entiteta u konceptualnom modelu, tada svaka osobina klase se preslikava u relaciju između dva entiteta uz zadovoljenje neophodnih ograničenja.
5. pravilo: svako ograničenje osobine (*Property Constraint*) u ontološkom jeziku se preslikava u ograničenje relacije u konceptualnom modelu podataka koje se odnosi na vrstu relacije, broj entiteta ili tipova entiteta u relaciji. Konkretno daml:intersectionOf, se preslikava u {or}

ograničenje, daml:unionOf se preslikava u {and} ograničenje, daml:hasClass (*Existential Quantifier*), daml:toClass (*Universal Quantifier*) određuju kada je minimalni kardinalitet relacije 0 (nula) i maksimalni M (*Many*, tj. više).

6. pravilo: relacija celina/deo u ontološkom jeziku se preslikava u kompozitnu relaciju u konceptualnom modelu podataka.

Kreiranje ontologije za veće sisteme predstavlja dugotrajan proces. S obzirom da se smatra da ontologija „sveobuhvatnije" predstavlja realan svet od modela podataka, autori rada [Elmasri et al., 2007] postavljaju pitanje da li se na osnovu ontologija mogu generisati modeli podataka?

Prema istraživanju [Su et al., 2002], u kojem je izvršena komparacija ontoloških alata, zaključeno je da su najznačajniji alati pogodni za razvoj ontologija: *Ontolingua, WebOnto, WebODE, Protégé, OntoEdit* i *OilEd*.

4.4. ONTOLOGIJE I PREDIKATSKI RAČUN PRVOG REDA

Antoniou i saradnici su predložili proširenje sloja ontologija semantičkog Web-a sa pravilima logičkog jezika. Definisali su prevođenje elemenata RDFS ontologija na predikatski račun [Antoniou et al., 2005]:

- klasa:
  `rdfs : type(x,C)` $C(x)$
- podklasa:
  `rdfs : subClassOf(C,D)` $C(x) \rightarrow D(x)$
- podosobina:
  `rdfs : subPropertyOf(P,Q)` $P(x, y) \rightarrow Q(x, y)$
- domena:
  `rdfs : domain(P,C)` $P(x, y) \rightarrow C(x)$
- opsega:
  `rdfs : range(P,C)` $P(x, y) \rightarrow C(y)$

gde su: *C, D* – nazivi klasa; *P, Q* – nazivi osobina; x, y - promenljive.

Grupa istraživača sa Univerziteta Karlsrue i Univerziteta u Južnoj Kaliforniji je, u okviru *Bubo* projekta [Volz et al.], istražila strategije implementacije OWL jezika u sistemima za logičko programiranje. Formirani su skupovi elemenata OWL jezika – primitiva koji su mapirani na predikatski račun. Takođe, formiran je skup komandi za deklarativni upitni jezik koji bi se koristio u implementaciji sistema. U ovom projektu su definisane još i relacije između ontoloških komponenti OWL jezika u okviru softverskih alata za izgradnju ontologija i elemenata predikatskog računa koji su prikazani u tabeli 4.3.

Tabela 4.3: OWL primitive interpretirane u predikatskom računu

OWL Primitives	Interpretation
Thing	$\triangle^{\mathcal{I}}$
Nothing	$\oslash$
C subClassOf D	$C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
C unionOf D	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
C intersectionOf D	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
complementOf C	$\triangle^{\mathcal{I}} \backslash C^{\mathcal{I}}$
C disjointWith D	$C^{\mathcal{I}} \cap D^{\mathcal{I}} = \oslash$
P subPropertyOf Q	$P^{\mathcal{I}} \subseteq Q^{\mathcal{I}}$
C sameClassAs D	$C^{\mathcal{I}} = D^{\mathcal{I}}$
P samePropertyAs Q	$P^{\mathcal{I}} = Q^{\mathcal{I}}$
x sameIndividualAs y	$x^{\mathcal{I}} = y^{\mathcal{I}}$
x differentIndividualFrom y	$x^{\mathcal{I}} \in \triangle^{\mathcal{I}} \backslash \{y\}$
TransitiveProperty P	$\forall p, q, r[(p, q) \in P^{\mathcal{I}} \wedge (q, r) \in P^{\mathcal{I}} \rightarrow (p, r) \in P^{\mathcal{I}}]$
SymmetricProperty P	$\forall p, q[(p, q) \in P^{\mathcal{I}} \leftrightarrow (q, p) \in P^{\mathcal{I}}]$
P inverseOf Q	$P^{\mathcal{I}} = Q^{\mathcal{I}^{-1}}]$
P minimumCardinality C n	$\{x \in \triangle^{\mathcal{I}} \mid \sharp\{y \mid (x, y) \in P^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\} \geqslant n\}$
P maximumCardinality C n	$\{x \in \triangle^{\mathcal{I}} \mid \sharp\{y \mid (x, y) \in P^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\} \leqslant n\}$
P allValuesFrom C	$\forall x, y[(x, y) \in P^{\mathcal{I}} \rightarrow y \in C^{\mathcal{I}}]$
P someValuesFrom C	$\forall x \exists y[(x, y) \in P^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}]$
P hasValue b	$\forall x[(x, b) \in P^{\mathcal{I}}]$
oneOf($b_1, b_2, ..., b_i, b_n$)	$\{b_1^{\mathcal{I}}, ..., b_n^{\mathcal{I}}\}$

4.5. AUTOMATIZACIJA EVALUACIJE MODELA PODATAKA

Istraživanja u oblasti automatizacije validacije modela podataka rezultovala su kreiranjem alata koji imaju za cilj merenje kvaliteta (uz detekciju grešaka/anomalija) i korekcije (transformacije) nad podacima ili modelima podataka. Tehnologije implementacije navedenih rešenja baziraju se na [Batini et al., 2006]:

- korišćenju deklarativnih jezika i operatorima logičkih transformacija (alat «*Ajax*», autori Galhardas et al., 2001);
- korišćenje formalnih pravila (alat «*Intelliclean*», autori [Low et al., 2001] za identifikaciju i integrisanje objekata);
- korišćenje meta modela (alat «*Artkos*», autori Vasiliadis et al., 2001);
- prezentovanju semantike podataka u okviru identifikacije objekata i deduplikacije objekata (alat «*Choice Maker*», autori Buechi et al., 2003);
- korišćenje sistema baziranih na znanju u procesu izrade konceptualnog modela podataka [Antony et al., 2004].

Alati razvijeni u okviru istraživanja korektnosti modela podataka odnose se na različite vrste modela podataka:

a) Konceptualni model: [Sugumaran et al., 2006] su razvili prototip softvera koji koristi domenske ontologije u kreiranju i validaciji dizajna konceptualnog modela podataka, odnosno ER modela podataka.

b) Relacioni model: [Emer et al., 2008] su razvili sistem ADIA (*Alternative Data Instance Analyses*), koji služi za testiranje šeme relacionog modela podataka. Ovo testiranje je bazirano na pronalaženju grešaka koje se odnose na elemente šeme, nekorektne ili izostavljene definicije ograničenja. Šema podataka je predstavljena putem meta modela koristeći UML notaciju, a zatim je izvršena transformacija u formalni oblik, pogodan za testiranje. Model podataka se predstavlja metamodelom M opisanim u XML notaciji. Prednosti ovog pristupa su ta što šema može biti testirana i ako baza podataka nije dostupna, a test podaci se generišu automatski i generisana instanca baze podataka se može koristiti za testiranje aplikacija baze podataka.

c) Objektni model: problem dizajna objektno-orjentisanih baza podataka istražen je u okviru rada autora Formica i Missikoff [Formica et al., 2006]. Posebno semantički značajan problem je modelovanje IS-A hijerarhije. Softverski prototip koristi elemente objektno orijentisanog jezika za definisanje fizičke strukture baze podataka korišćenjem TQL jezika.

Autori rada [Choppella et al., 2007] su predstavila svoja iskustva sa istraživanjem upotrebe PVS sistema da formalno specificiraju i rezonuju sa ER modelom podataka na različitim nivoima apstrakcije. Entiteti i veze su navedeni kao korisnički definisani tipovi, dok se ograničenja izražavaju kroz aksiome. Choppella i saradnici su pokazali kako je ispravnost mapiranja, od apstraktnog do konceptualnog ER modela i od konceptualnog ER modela do šeme modela, formalno zasnovano korišćenjem provere tipova. Dokazi većine uslova korektnosti tipa su prilično mali, sa četiri koraka ili manje.

Rad [Brdjanin et al., 2012] prikazuje pristup automatizovanom dizajnu konceptualnog modela podataka. UML dijagram aktivnosti, koji se često koristi u modelovanju poslovnih procesa i u dizajnu softvera, je upotrebljen za automatsko generisanje dijagrama klasa koji prezentuje konceptualni model baze podataka. Formalna pravila za automatsko generisanje klasa i njihovih asocijacija pokrivaju automatsko ekstrahovanje poslovnih objekata i učesnika poslovnih procesa. Na bazi ovih pravila je implementiran automatski generator koji je verifikovan na stvarnom poslovnom modelu.

5. MODEL ONTOLOŠKI ZASNOVANE ANALIZE SEMANTIČKE KOREKTNOSTI MODELA PODATAKA PRIMENOM SISTEMA AUTOMATSKOG REZONOVANJA

Model ontološki zasnovane analize semantičke korektnosti modela podataka primenom sistema automatskog rezonovanja je formiran na osnovu teoretskog istraživanja i pregleda vladajućih stavova u oblasti istraživanja.

5.1. ZASNOVANOST MODELA NA PRETHODNIM ISTRAŽIVANJIMA

Za opis i modelovanje znanja iz određene problemske oblasti je izabran mehanizam ontologija koji je opšte prihvaćen način za opis znanja specifičnih domena [Ullman et al., 2002]. "Glavna razlika između ontologije i šeme baze podataka je u tome što je šema obično ograničena na opis manjeg podskupa "mini-sveta" iz realnosti u cilju memorisanja i upravljanja podacima. Ontologija se najčešće smatra opštijom, jer nastoji da opiše deo realnosti ili oblast od interesa što potpunije." [Shoval 1997]

Do sličnog zaključka u pogledu razlika između ontologije i šeme baze podataka su došli i autori [Formica et al., 2006]. Prema [Elmasri et al., 2007] smatra se da ontologije sveobuhvatnije predstavljaju realan svet od modela podataka. Autori [El-Ghalayini et al., 2005] su utvrdili da se na osnovu ontologija može generisati konceptualni model podataka. Vysniauskas sa koautorima opisuje generisanje relacione šeme na osnovu ontološke OWL specifikacije [Vysniauskas et al., 2006]. Međutim, kreiranje ontologije i generisanje modela podataka na osnovu ontologije u okviru ontoloških alata nezavisno je od rada u CASE alatu kojim se integriše proces razvoja informacionog sistema, što dovodi u pitanje zasnovanost kreiranih ontologija i modela podataka. Javlja se potreba za povezanošću ontoloških alata i CASE alata, čime bi se prevazišao navedeni problem [Vysniauskas et al., 2006].

Za kreiranje ontologije, u disertaciji je upotrebljen ontološki alat Protégé, sa univerziteta Stanford. Ovaj alat je izabran jer je prema istraživanju [Su et al., 2002], u kojem je izvršena komparacija ontoloških alata, zaključeno je da ovaj alat spada u nekoliko najznačajniji alata pogodnih za razvoj ontologija, a takođe je dostupan preko Interneta.

S obzirom da ontologija opštije i kompletnije opisuje znanje problemske oblasti, odnosno poslovnog domena, predložen je model sistema koji je baziran na ontologijama i koji se koristi za evaluaciju semantike konceptualnih modela podataka.

Modeli podataka se najčešće kreiraju, u procesu razvoja informacionog sistema, pomoću CASE alata, u kojima se u proces modelovanja podataka integrišu i koriste rezultati prethodno izvršenog modeliranja poslovnih procesa. Ontologija opisuje kompletno znanje konkretnog problemskog domena i koristi se radi upoređivanja sa

modelom podataka koji je dizajniran, kako bi kreirani model bio vrednovan sa semantičkog aspekta.

Pronađeni su modaliteti za uspostavljanje korelacija ontologija i formalnog logičkog jezika i izvršeno mapiranje ontologije u neki logički jezik, poput predikatskog računa prvog reda.

Antoniou i saradnici su definisali prevođenje elemenata RDFS ontologija na predikatski račun [Antoniou et al., 2005]. Mapiranje ontologije u elemente konceptualnog modela podataka su izvršili [El-Ghalayini et al., 2005]. Grupa istraživača sa Univerziteta Karlsrue i Univerziteta u Južnoj Kaliforniji je, u okviru Bubo projekta [Volz et al.], istražila strategije implementacije OWL jezika u sistemima za logičko programiranje. Formirani su skupovi elemenata OWL jezika – primitiva koji su mapirani na predikatski račun.

Na osnovu navedenih radova je izvršeno mapiranje domenske ontologije na predikatski račun prvog reda, a zatim i u rečenice sistema za automatsko rezonovanje.

Istraživanja [De Lucia et al., 2009] i [Shoval 1997] su pokazala da je stariji ER model podesniji za projektovanje baza podtaka od mlađeg objektno orijentisanog modela podataka. Preporuka istraživača iz ove oblasti je da se dizajn baze podataka uradi u ER modelu, pa da se zatim izvrši njegovo mapiranje u OOM, u kom je potrebno dodati metode i poruke koje opisuju ponašanje objekata iz realng sveta.

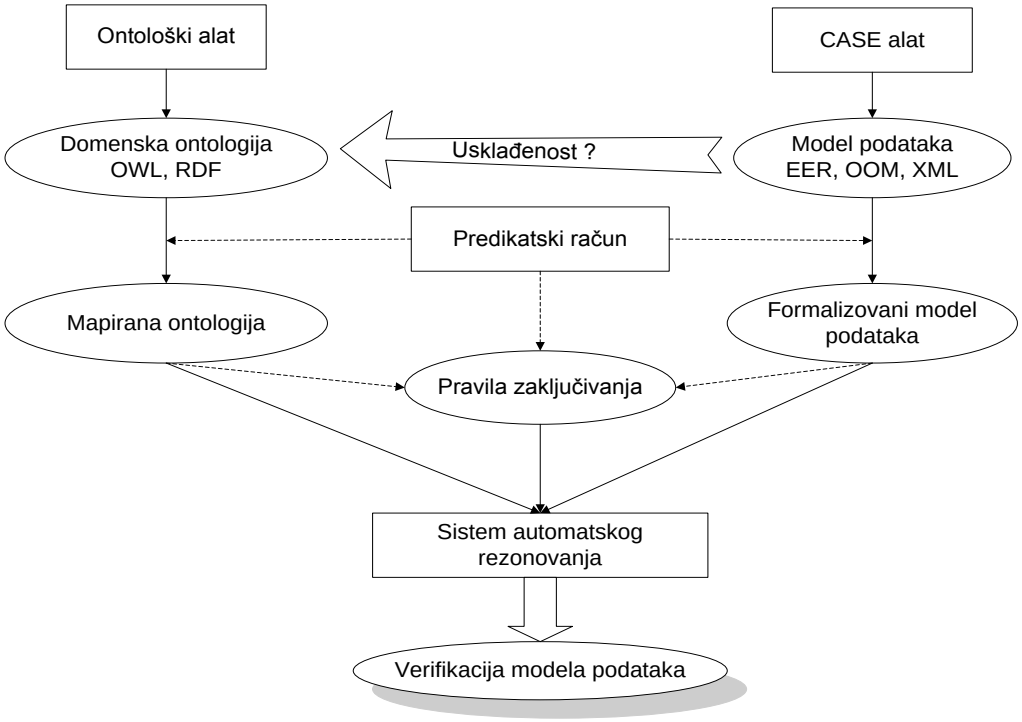
U oblasti evaluacije procesa kreiranja modela podataka, neka eksperimentalna istraživanja su sprovedena u vezi sa poređenjem ER i objektno orijentisano modelovanje [Shoval 1997], [De Lucia et al., 2009], [Aguirre-Urreta et al., 2008], gde je pokazan značaj ER modelovanja i njegova prednost u odnosu na objektno-orijentisani pristup. ER model je, ipak, podesniji od objektnog modela podataka za logičko projektovanje baze podataka, jer je jednostavniji za razumevanje, bolji je za predstavljanje kompleksnijih relacija između entiteta (unarne i ternarne veze), zahteva manje vremena za kreiranje šeme i projektanti ga više koriste.

Zbog navedenih razloga je ER(EER) model podataka, tj. konceptualni model podataka izabran kao jedan od dva najčešće upotrebljavana modela za projektovanje baza podataka.

Formalnu prezentacija relacionog modela podataka su izvršili [Emer et al., 2008]. Na osnovu njihovog rada zaključeno je da provera korektnosti modela podataka zahteva formalnu specifikaciju modela podataka, da bi se kreirale rečenice u formalnom jeziku, tj. elementi specifikacije koji se mogu predstaviti kao aksiome sistema automatskog rezonovanja.

Istraživanja u oblasti automatizacije validacije modela podataka rezultovala su kreiranjem alata koji imaju za cilj merenje kvaliteta, uz detekciju grešaka i anomalija, kao i korekcije i transformacije nad podacima ili modelima podataka [Batini et al., 2006].

[Sugumaran et al., 2006] su razvili prototip softvera koji koristi domenske ontologije u kreiranju i validaciji dizajna konceptualnog modela podataka. Problem dizajna objektno-orjentisanih baza podataka istražen je u okviru rada autora Formica i Missikoff [Formica et al., 2006]. S obzirom da su [Vysniauskas et al., 2006] uočili potrebu za povezanošću ontoloških alata i CASE alata, čime bi se prevazišao problem nezavisnog rada CASE alata, poput Sybase Power Deisgner-a i Protégé-a, u cilju integracije procesa razvoja informacionog sistema, izvršeno je programiranje aplikacije za mapiranje RDF/OWL ontologije, formalizaciju konceptualnog modela podataka i njihovu integraciju sa semantičkim pravilima zaključivanja u oblik Prolog programa. Semantička verifikacija modela podataka se vrši postavljanjem odgovarajućih upita Prolog sistemu.



Slika 5.1. – Model ontološki zasnovane analize semantičke korektnosti modela podataka

Dijagram na slici 5.1. prikazuje šemu modela sistema i proces korišćenja ontološki zasnovane evaluacije modela podataka uz pomoć sistema automatskog rezonovanja. Da bi sistem automatskog rezonovanja, u ovom slučaju Prolog, mogao dati odgovore na upite korisnika, definisana su pravila zaključivanja na osnovu kojih se vrši semantički zasnovana analiza korektnosti modela podataka pomoću denisanih metrika. Centralna tačka predloženog modela analize je objedinjavanje mapirane ontologije, formalizovanog konceptualng modela podataka i pravila zaključivanja u oblik klauzula koje su razumljive sistemu automatskog rezonovanja. Elementi predloženog modela i teoretskog istraživanja su prikazani u [Kazi et al., 1], [Kazi et al., 3], [Kazi et al., 7], [Kazi et al., 8] i [Kazi et al., 9].

5.2. FORMALIZACIJA MODELA PODATAKA

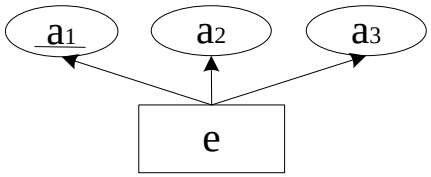
Pošto provera korektnosti modela podataka, prema [Emer et al., 2008], zahteva formalnu specifikaciju modela podataka u cilju formiranja rečenica u formalnom jeziku i predstavljanja u formi aksioma sistema automatskog rezonovanja, formalizacija modela podataka Emera i koautora je proširena elementom *R* koji predstavlja skup relacija, tj. poveznika između entiteta.

Definicija 3.9. *F*ormalna specifikacija ***S EER* modela podataka** je uređena petorka (*E, A, R, S, P*), gde je:

E - konačan skup **entiteta,**
A - konačan skup **atributa,**
R - konačan skup **poveznika između entiteta,**
S - konačan skup **ograničenja** koja se tiču domena, definicija, relacija i semantike pridružene entitetima i atributima,
P - konačan skup **relacija između entiteta, atributa, poveznika i ograničenja**.

Prikazivanje elemenata EER modela podataka i dijagrama je urađeno prema [Chen 1976], [Elmasri et al., 2007], [Ullman et al., 2002], [Mogin i sar. 2000], [Mogin i sar. 1996].

Na slici 5.2. je prikazan **tip entiteta *e*** koji je opisan svojim imenom i obeležjima entiteta, tj. **atributa** a_1, a_2, a_3, od kojih je a_1 indentifikacioni atribut i domeni atributa su $a_1 \rightarrow dom_1$, $a_2 \rightarrow dom_2$ i $a_3 \rightarrow dom_3$. Za atribute a_1 i a_2 važi ograničenje da ne mogu biti bez vrednosti.



Slika 5.2. – Grafički prikaz entiteta i atributa

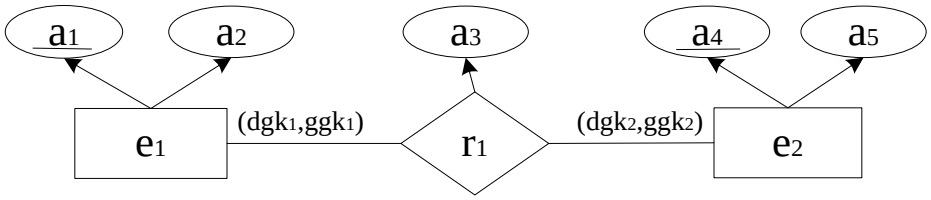
Formalizacija entiteta i atributa se predstavlja preko skupova:

$E = \{e\}$
$A = \{a_1, a_2, a_3\}$
$R = \{\ \}$
$S = \{id\_atr, ob\_vr\_atr, dom_1, dom_2, dom_3\}$
$P = \{p(e, a_1), p(e, a_2), p(e, a_3), p(a_1, dom_1), p(a_2, dom_2), p(a_3, dom_3),$
$p(a_1, ob\_vr\_atr), p(a_2, ob\_vr\_atr), p(a_1, id\_atr)\}$

gde je:
id_atr – identifikacioni atribut,
ob_vr_atr – obavezna vrednost atributa,
dom_i – domen atributa.

Na slici 5.3. su prikazani **tipovi entiteta** e_1 i e_2 koji su povezani **relacijom, tj. poveznikom** r_1. Tip entiteta e_1 je opisan svojim imenom i atributima a_1 i a_2, od kojih je a_1 indentifikacioni atribut, a domeni atributa su: $a_1 \rightarrow dom_1$, $a_2 \rightarrow dom_2$. Tip entiteta e_2 je

opisan svojim imenom i atributima a_4 i a_5, od kojih je a_4 indentifikacioni atribut i domeni atributa su $a_4 \rightarrow dom_4$ i $a_5 \rightarrow dom_5$. Za atribute a_1, a_2, a_4 i a_5 važi ograničenje da ne mogu biti bez vrednosti. Poveznik r_1 je opisan svojim imenom, atributom a_3 i kardinalnošću $((dgk_1,ggk_1):(dgk_2,ggk_2))$. Domen atributa a_3 je dom_3 i za njega ne postoji ograničenje da ne može biti bez vrednosti.



Slika 5.3. - Grafički prikaz poveznika između entiteta

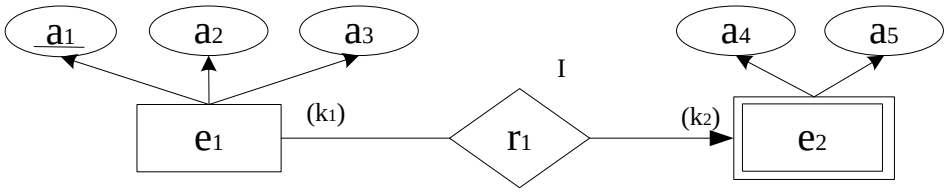
Formalizacija poveznika između entiteta se predstavlja preko skupova:

$E = \{e_1, e_2\}$
$A = \{a_1, a_2, a_3, a_4, a_5\}$
$R = \{r_1\}$
$S = \{id\_atr, ob\_vr\_atr, dom_1, dom_2, dom_3, dom_4, dom_5, dgk_1, dgk_2, ggk_1, ggk_2\}$
$P = \{p(e_1, a_1), p(e_1, a_2), p(r_1, a_3), p(e_2, a_4), p(e_2, a_5), p(a_1, dom_1), p(a_2, dom_2),$
$p(a_3, dom_3), p(a_1, dom_4), p(a_1, dom_5), p(a_1, ob\_vr\_atr), p(a_2, ob\_vr\_atr),$
$p(a_4, ob\_vr\_atr), p(a_5, ob\_vr\_atr), p(a_1, id\_atr), p(a_4, id\_atr), p(e_1, r_1),$
$p(r_1, e_2), p(r_1, dgk_1), p(r_1, ggk_1), p(r_1, dgk_2), p(r_1, ggk_2)\}$

gde je:

id_atr – identifikacioni atribut,
ob_vr_atr – obavezna vrednost atributa,
dom_i – domen atributa,
dgk_i – donja granica kardinaliteta,
ggk_i – gornja granica kardinaliteta.

Na slici 5.4. su prikazani **jak tip entiteta** e_1 i **slab tip entiteta** e_2 koji su povezani relacijom r_1. Slabi tip entiteta e_2 **identifikaciono** zavisi od jakog tipa entiteta e_1. Tip entiteta e_1 je opisan svojim imenom i atributima a_1, a_2 i a_3, od kojih je a_1 indentifikacioni atribut, a domeni atributa su: $a_1 \rightarrow dom_1$, $a_2 \rightarrow dom_2$ i $a_3 \rightarrow dom_3$. Tip entiteta e_2 je opisan svojim imenom i atributima a_4 i a_5, a domeni atributa su $a_4 \rightarrow dom_4$ i $a_5 \rightarrow dom_5$. Za atribute a_1, a_2, a_4 i a_5 važi ograničenje da ne mogu biti bez vrednosti, dok ovo ograničenje ne važi za atribut a_3. Poveznik r_1 je opisan svojim imenom i kardinalnošću $((ggk_1):(ggk_2))$, dok je $dgk_1=dgk_2=1$, te se ne navodi na dijagramu.



Slika 5.4. - Grafički prikaz veze između jakog i slabog tipa entiteta (identifikaciona zavisnost)

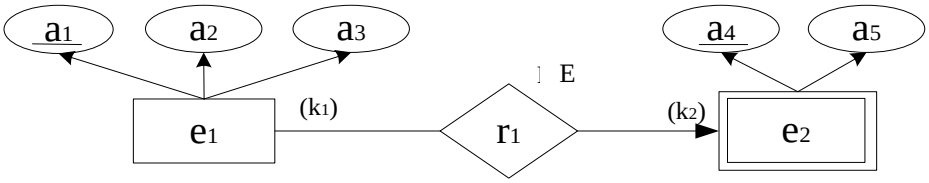
Formalizacija veze jak-slab tip entiteta, identifikaciona zavisnost, se predstavlja preko skupova:

E = {e_1, e_2}
A = {a_1, a_2, a_3, a_4, a_5}
R = {r_1}
S = {k_1, k_2, id_atr, ob_vr_atr, dom_1, dom_2, dom_3, dom_4, dom_5, id_zav}
P = {p(e_1, a_1), p(e_1, a_2), p(e_1, a_3), p(e_2, a_4), p(e_2, a_5), p(a_1, id_atr), p(e_1, r_1), p(e_2, r_1),
p(r_1, k_1), p(r_1, k_2), p(a_1, dom_1), p(a_2, dom_2), p(a_3, dom_3), p(a_4, dom_4),
p(a_5, dom_5), p(a_1, ob_vr_atr), p(a_2, ob_vr_atr), p(a_4, ob_vr_atr),
p(a_5, ob_vr_atr), p(a_1, id_atr), p(id_zav,r_1)}

gde je:

id_atr – identifikacioni atribut,
ob_vr_atr – obavezna vrednost atributa,
dom_i – domen atributa,
id_zav – identifikaciona zavisnost,
k_i – kardinalitet poveznika.

Na slici 5.5. su prikazani **jak tip entiteta** e_1 i **slab tip entiteta** e_2 koji su povezani relacijom r_1. Slabi tip entiteta e_2 **egzistencijalno** zavisi od jakog tipa entiteta e_1. Tip entiteta e_1 je opisan svojim imenom i atributima a_1, a_2 i a_3, od kojih je a_1 identifikacioni atribut, a domeni atributa su: $a_1 \rightarrow dom_1$, $a_2 \rightarrow dom_2$ i $a_3 \rightarrow dom_3$. Tip entiteta e_2 je opisan svojim imenom i atributima a_4 i a_5, od kojih je a_4 indentifikacioni atribut i domeni atributa su $a_4 \rightarrow dom_4$ i $a_5 \rightarrow dom_5$. Za atribute a_1, a_2, a_4 i a_5 važi ograničenje da ne mogu biti bez vrednosti, dok ovo ograničenje ne važi za atribut a_3. Poveznik r_1 je opisan svojim imenom i kardinalnošću ($(ggk_1):(ggk_2)$), dok je $dgk_1=dgk_2=1$, te se ne navodi na dijagramu.



Slika 5.5. - Grafički prikaz veze između jakog i slabog tipa entiteta (egzistencijalna zavisnost)

Formalizacija veze jak-slab tip entiteta, egzistencijalna zavisnost, se predstavlja preko skupova:

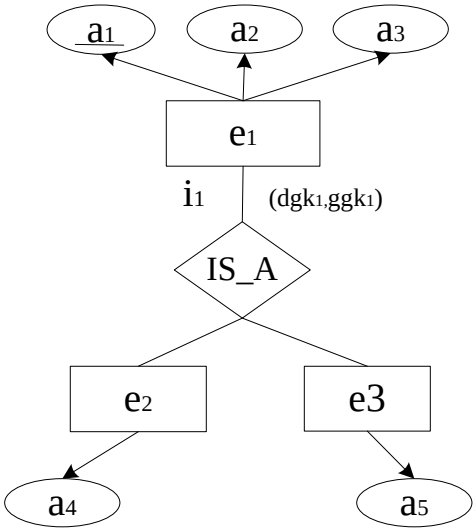
E = {e_1, e_2}
A = {a_1, a_2, a_3, a_4, a_5}
R = {r_1}
S = {k_1, k_2, id_atr, ob_vr_atr, dom_1, dom_2, dom_3, dom_4, dom_5, eg_zav}
P = {p(e_1, a_1), p(e_1, a_2), p(e_1, a_3), p(e_2, a_4), p(e_2, a_5), p(a_1, id_atr), p(e_1, r_1), p(e_2, r_1),
p(r_1, k_1), p(r_1, k_2), p(a_1, dom_1), p(a_2, dom_2), p(a_3, dom_3), p(a_4, dom_4),
p(a_5, dom_5), p(a_1, ob_vr_atr), p(a_2, ob_vr_atr), p(a_4, ob_vr_atr),
p(a_5, ob_vr_atr), p(a_4, id_atr), p(eg_zav,r_1)}

gde je:

id_atr – identifikacioni atribut,

ob_vr_atr – obavezna vrednost atributa,
dom_i – domen atributa,
eg_zav – egzistencijalna zavisnost,
k_i – kardinalitet poveznika.

Na slici 5.6. su prikazani **nadtip entiteta** (nadklasa) e_1 i **podtipovi entiteta** (podklase) e_2 i e_3 koji su povezani **is_a hijerarhijom** i_1. Nadtip entiteta e_1 je opisan svojim imenom i atributima a_1, a_2 i a_3, od kojih je a_1 identifikacioni atribut, a domeni atributa su: $a_1 \rightarrow dom_1$, $a_2 \rightarrow dom_2$ i $a_3 \rightarrow dom_3$. Podtip entiteta e_2 je opisan svojim imenom i atributom a_4, a domen atributa je $a_4 \rightarrow dom_4$. Podtip entiteta e_3 je opisan svojim imenom i atributom a_5, a domen atributa je $a_5 \rightarrow dom_5$. Za atribute a_1, a_2, a_4 i a_5 važi ograničenje da ne mogu biti bez vrednosti, dok ovo ograničenje ne važi za atribut a_3. IS_A hijerarhija i_1 je opisana svojim imenom i kardinalnošću (dgk_1,ggk_1), dok je $dgk_2=ggk_2=1$ kod svakog podtipa entiteta e_2 i e_3, te se ne navodi na dijagramu.



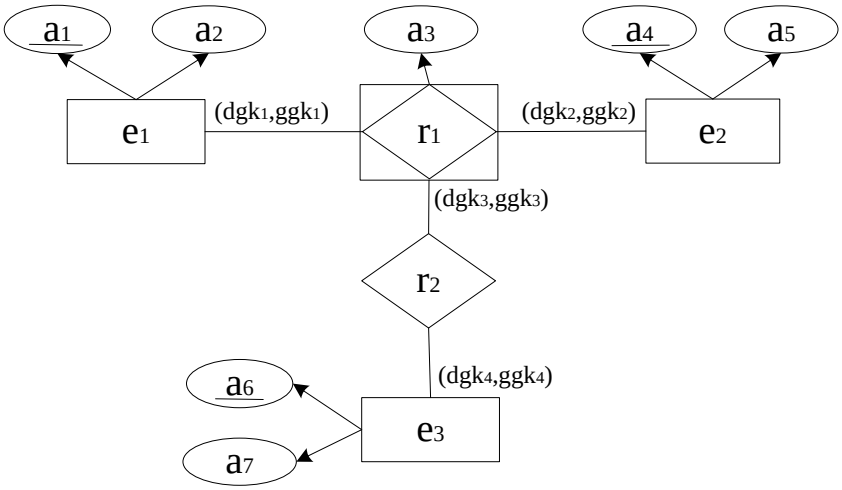
Slika 5.6. - Grafički prikaz IS_A hijerarhije

Formalizacija veze jak-slab tip entiteta, egzistencijalna zavisnost, se predstavlja preko skupova:

$E = \{e_1, e_2, e_3\}$
$A = \{a_1, a_2, a_3, a_4, a_5\}$
$R = \{i_1\}$
$S = \{dgk_1, ggk_1, id\_atr, ob\_vr\_atr, nadtip, dom_1, dom_2, dom_3, dom_4, dom_5\}$
$P = \{p(e_1, a_1), p(e_1, a_2), p(r_1, a_3), p(e_2, a_4), p(e_2, a_5), p(a_1, id\_atr), p(e_1, i_1), p(e_2, i_1),$
$p(e_3, i_1), p(i_1, dgk_1), p(i_1, ggk_1), p(i_1, nadtip), p(a_1, dom_1), p(a_2, dom_2),$
$p(a_3, dom_3), p(a_4, dom_4), p(a_5, dom_5), p(a_1, ob\_vr\_atr), p(a_2, ob\_vr\_atr),$
$p(a_4, ob\_vr\_atr), p_{21}(a_5, ob\_vr\_atr)\}$

gde je:
id_atr – identifikacioni atribut,
ob_vr_atr – obavezna vrednost atributa,
dom_i – domen atributa,
nadtip – veza nadklase i podklasa,
dgk_i – donja granica kardinaliteta,
ggk_i – gornja granica kardinaliteta.

Na slici 5.7. su prikazani tipovi entiteta e_1 i e_2 koji su povezani **mešovitim objektom vezom, tj. gerundom** r_1, koji je poveznikom r_2 povezan sa tipom entiteta e_3. Tip entiteta e_1 je opisan svojim imenom i atributima a_1 i a_2, od kojih je a_1 indentifikacioni atribut, a domeni atributa su: $a_1 \rightarrow dom_1$, $a_2 \rightarrow dom_2$. Tip entiteta e_2 je opisan svojim imenom i atributima a_4 i a_5, od kojih je a_4 identifikacioni atribut i domeni atributa su $a_4 \rightarrow dom_4$ i $a_5 \rightarrow dom_5$. Tip entiteta e_3 je opisan svojim imenom i atributima a_6 i a_7, gde je a_6 identifikacioni atribut, a domeni atributa su $a_6 \rightarrow dom_6$ i $a_7 \rightarrow dom_7$. Za atribute a_1, a_2, a_4, a_5, a_6 i a_7 važi ograničenje da ne mogu biti bez vrednosti. Domen atributa a_3 je dom_3 i za njega ne postoji ograničenje da ne može biti bez vrednosti. Gerund r_1 je opisan svojim imenom, atributom a_3 i kardinalnošću $((dgk_1,ggk_1):(dgk_2,ggk_2))$. Poveznik r_2 je opisan svojim imenom i kardinalnošću $((dgk_3,ggk_3):(dgk_4,ggk_4))$.



Slika 5.7. - Grafički prikaz mešovitog objekta-veza (gerund)

Formalizacija mešovitog objekta-veza (gerund) se predstavlja preko skupova:

$E = \{e_1, e_2, e_3, r_1\}$

$A = \{a_1, a_2, a_3, a_4, a_5, a_6, a_7\}$

$R = \{r_1, r_2\}$

$S = \{dgk_1, dgk_2, dgk_3, dgk_4, ggk_1, ggk_2, ggk_3, ggk_4, id\_atr, ob\_vr\_atr, dom_1, dom_2, dom_3, dom_4, dom_5, dom_6, dom_7\}$

$P = \{p(e_1, a_1), p(e_1, a_2), p(r_1, a_3), p(e_2, a_4), p(e_2, a_5), p(a_1, id\_atr), p(a_3, id\_atr), p(a_6, id\_atr), p(e_1, r_1), p(e_2, r_1), p(r_1, r_2), p(e_1, r_2), p(r_1, dgk_1), p(r_1, ggk_1), p(r_1, dgk_2), p(r_1, ggk_2), p(r_2, dgk_3), p(r_2, ggk_3), p(r_2, dgk_4), p(r_2, ggk_4), p(a_1, dom_1), p(a_2, dom_2), p(a_3, dom_3), p(a_4, dom_4), p(a_5, dom_5), p(a_6, dom_6), p(a_7, dom_7), p(a_1, ob\_vr\_atr), p(a_2, ob\_vr\_atr), p(a_4, ob\_vr\_atr), p(a_5, ob\_vr\_atr), p(a_6, ob\_vr\_atr), p(a_7, ob\_vr\_atr)\}$

gde je:

id_atr – identifikacioni atribut,

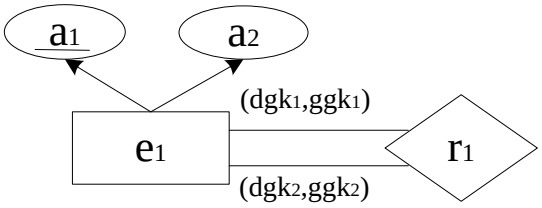
ob_vr_atr – obavezna vrednost atributa,

dom_i – domeni atributa.

dgk_i – donja granica kardinaliteta,

ggk_i – gornja granica kardinaliteta.

Na slici 5.8. je prikazan **tip entiteta** e_1 koji je povezan sa samim sobom preko **unarnog (rekurzivnog) poveznika** r_1. Tip entiteta e_1 je opisan svojim imenom i atributima a_1 i a_2, od kojih je a_1 indentifikacioni atribut, a domeni atributa su: $a_1 \rightarrow dom_1$, $a_2 \rightarrow dom_2$. Za atribute a_1 i a_2 važi ograničenje da ne mogu biti bez vrednosti. Poveznik r_1 je opisan svojim imenom i kardinalnošću $((dgk_1,ggk_1):(dgk_2,ggk_2))$.



Slika 5.8. - Grafički prikaz entiteta i unarnog poveznika

Formalizacija mešovitog objekta-veza (gerund) se predstavlja preko skupova:

$E = \{e_1\}$
$A = \{a_1, a_2\}$
$R = \{r_1\}$
$S = \{dgk_1, dgk_2, ggk_1, ggk_2, ob\_vr\_atr, id\_atr, dom_1, dom_2\}$
$P = \{p(e_1, a_1), p(e_1, a_2), p(a_1, id\_atr), p(e_1, r_1), p(r_1, dgk_1), p(r_1, ggk_1), p(r_1, dgk_2), p(r_1, ggk_2), p(a_1, dom_1), p(a_2, dom_2), p(a_1, ob\_vr\_atr), p(a_2, ob\_vr\_atr)\}$

gde je:

id_atr – identifikacioni atribut,
ob_vr_atr – obavezna vrednost atributa,
dom_i – domeni atributa.
dgk_i – donja granica kardinaliteta,
ggk_i – gornja granica kardinaliteta.

5.2.1. Kreiranje Prolog rečenica za model podataka

Prilikom formalizacije modela podataka i predstavljanja elemenata skupova *E, A, R, S, P* u obliku zapisa na predikatskom računu prvog reda i u formi Prolog klauzula, korištene su sledeće konvencije u cilju povećanja čitljivosti budućih Prolog rečenica:

- Elementi skupa *E* se predstavljaju predikatom «ent».
- Elementi skupa *A* se predstavljaju predikatom «atr».
- Elementi skupa *R* se predstavljaju predikatom «rel».
- Elementi skupa *S* se predstavljaju predikatom «res».
- Elementi skupa *P* se predstavljaju predikatom «p».

Na osnovu [Gallaire et al 1984], (3.9), (3.11) i (3.12) elementi ovih skupova se transformišu u Prolog činjenice, kao klauzule koje se sastoje samo od glave pravila, dok je telo pravila prazno prema [Hotomski 2003] i [Tošić i sar.]. Vrši se preoznačavanje simbola predikata prema navedenoj konvenciji i dodavanje tačke kao oznake za kraj rečenice.

Primer transformacije formalizovanog modela podataka, prikazanog na slici 5.7., u oblik Prolog činjenica, za tipove entiteta e_1 i e_2 koji su povezani gerundom r_1, koji je poveznikom r_2 povezan sa tipom entiteta e_3.

$E = \{e_1, e_2, e_3, r_1\}$ ⇒

```
ent(e1).
ent(e2).
ent(e3).
ent(r1).
```

$A = \{a_1, a_2, a_3, a_4, a_5, a_6, a_7\}$ ⇒

```
atr(a1).    atr(a5).
atr(a2).    atr(a6).
atr(a3).    atr(a7).
atr(a4).
```

$R = \{r_1, r_2\}$ ⇒

```
rel(r1).
rel(r2).
```

$S = \{dgk_1, dgk_2, dgk_3, dgk_4, ggk_1, ggk_2, ggk_3, ggk_4, id\_atr, ob\_vr\_atr, dom_1, dom_2, dom_3, dom_4, dom_5, dom_6, dom_7\}$ ⇒

```
res(dgk1).    res(ggk3).    res(dom3).
res(dgk2).    res(ggk4).    res(dom4).
res(dgk3).    res(id_atr).  res(dom5).
res(dgk4).    res(ob_vr_atr). res(dom6).
res(ggk1).    res(dom1).    res(dom7).
res(ggk2).    res(dom2).
```

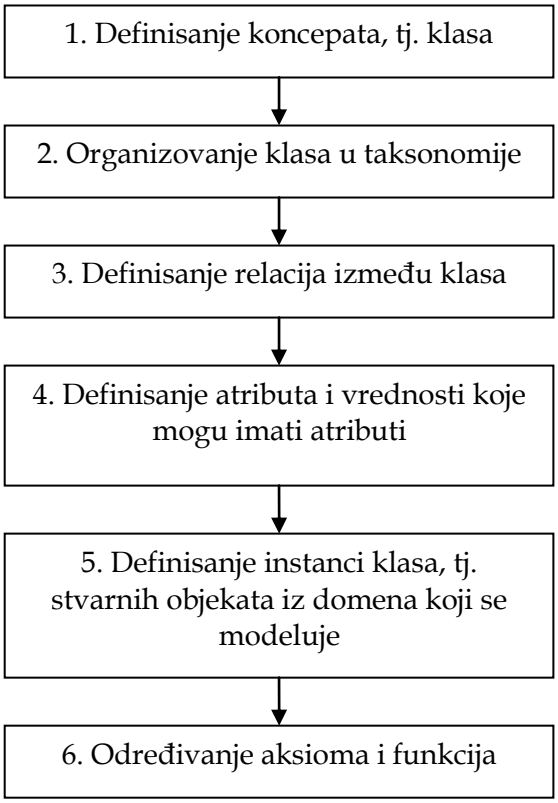
$P = \{p(e_1, a_1), p(e_1, a_2), p(r_1, a_3), p(e_2, a_4), p(e_2, a_5), p(a_1, id\_atr), p(a_3, id\_atr), p(a_6, id\_atr), p(e_1, r_1), p(e_2, r_1), p(r_1, r_2), p(e_1, r_2), p(r_1, dgk_1), p(r_1, ggk_1), p(r_1, dgk_2), p(r_1, ggk_2), p(r_2, dgk_3), p(r_2, ggk_3), p(r_2, dgk_4), p(r_2, ggk_4), p(a_1, dom_1), p(a_2, dom_2), p(a_3, dom_3), p(a_4, dom_4), p(a_5, dom_5), p(a_6, dom_6), p(a_7, dom_7), p(a_1, ob\_vr\_atr), p(a_2, ob\_vr\_atr), p(a_4, ob\_vr\_atr), p(a_5, ob\_vr\_atr), p(a_6, ob\_vr\_atr), p(a_7, ob\_vr\_atr)\}$ ⇒

```
p(e1, a1).    p(e1, r2).    p(a3, dom3).
p(e1, a2).    p(r1, dgk1).  p(a4, dom4).
p(r1, a3).    p(r1, ggk1).  p(a5, dom5).
p(e2, a4).    p(r1, dgk2).  p(a6, dom6).
p(e2, a5).    p(r1, ggk2).  p(a7, dom7).
p(a1, id_atr). p(r2, dgk3). p(a1, ob_vr_atr).
p(a3, id_atr). p(r2, ggk3). p(a2, ob_vr_atr).
p(a6, id_atr). p(r2, dgk4). p(a4, ob_vr_atr).
p(e1, r1).    p(r2, ggk4).  p(a5, ob_vr_atr).
p(e2, r1).    p(a1, dom1).  p(a6, ob_vr_atr).
p(r1, r2).    p(a2, dom2).  p(a7, ob_vr_atr).
```

5.3. KREIRANJE ONTOLOGIJE

Ontologije predstavljaju mehanizam za formalizaciju semantike određenog domena eksplicitno opisujući njegove elemente. S obzirom da se sastoje od konačnog skupa koncepata i njihovih instanci, tj. konkretnih objekata (individua) koji su međusobno povezani relacijama, ontologije opisuju mogućnosti i funkcije tih koncepata iz realnog sveta i njihove osobine na što je moguće prirodniji način. [Bergamaschi et al., 2004], [Gruber 1995], [El-Ghalayini et al., 2005], [Lammari et al., 2004], [Horridge 2009]

Metodologija za razvoj i kreiranje ontologija podrazumevaju sledeće korake [5]:



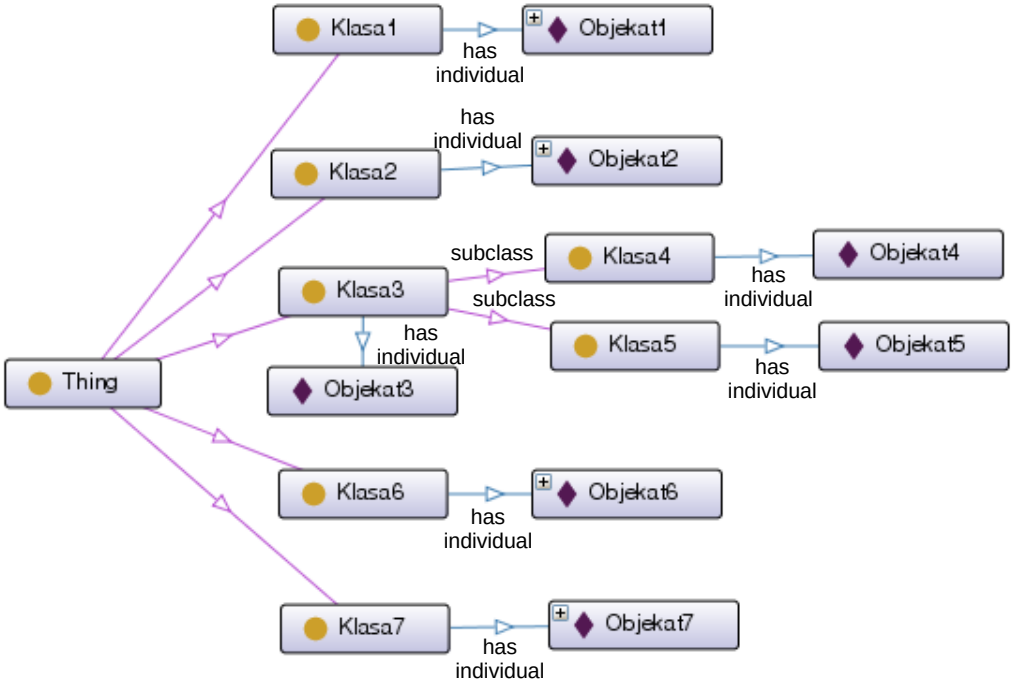
Slika 5.9. – Koraci u kreiranju ontologije [5]

Prvo je potrebno imenovati klase (*Classes*). Za svaku klasu se mogu odrediti objekti koji pripadaju toj klasi, ekvivalentne klase, disjunktne klase i superklase. Sledi organizacija klasa u taksonomije, određivanje podklasa neke nadklase. Na slici 5.10., koja prikazuje apstraktnu ontologiju, definisano je nekoliko klasa i za „Klasu 3“ je izvršena specijalizacija na dve podklase (*Subclasses*): „Klasa 4“ i „Klasa 5“.

Sledi kreiranje objekata kao instanci klasa (*Named Individual*) iz domena koji se modeluje. Za objekte se formira lista članova koji pripadaju istoj klasi. Opciono, moguće je definisati tipove objekata, identične ili različite individue. Za objekte se određuju osobine objekata (*Object Property*) i osobine podataka (*Data Property*) koje isti poseduju ili ne poseduju. U modelu prikazanom na slici je formirano sedam klasa, od kojih svaka ima po jednu instancu: Klasa 1 → Objekat 1, Klasa 2 → Objekat 2, Klasa 3 →

Objekat 3, Klasa 4 → Objekat 4, Klasa 5 → Objekat 5, Klasa 6 → Objekat 6 i Klasa 7 → Objekat 7.

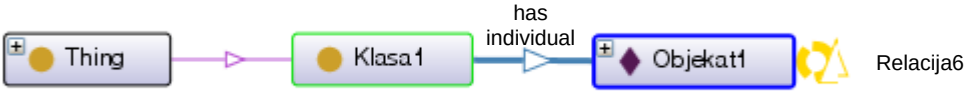
U narednom koraku se definišu osobine podataka, tj. atributi. Dodeljuje se naziv, domen (*Domain*), tip podatka i opseg (*Data Range*), a moguće je odrediti identične osobine podataka (*Equivavlent Data Properties*), nadređene osobine u slučaju da se formiraju strukture podataka (*Super Properties*) i osobine podataka sa kojima definisana osobina nema zajedničkih karakteristika (*Disjoint Properties*). Na kraju ovog koraka je potrebno povezati objekte sa odgovarajućom osobinom podataka (*Data Property Assertion*).



Slika 5.10. Klase ontologije sa instancama i taksonomijom

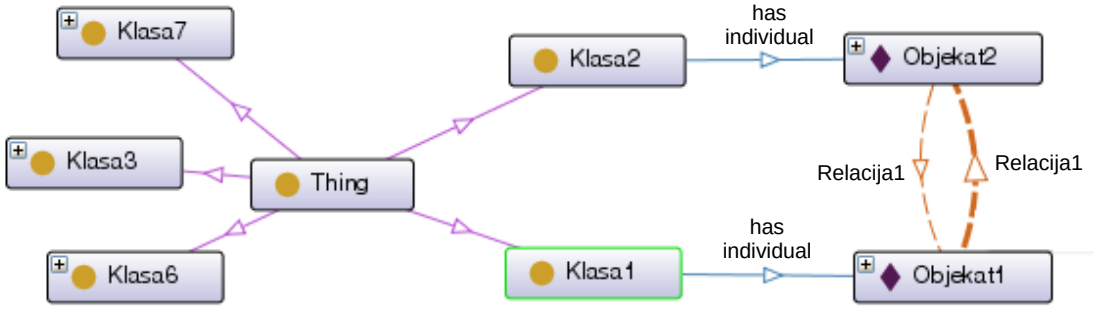
Relacije (*Object Property*) su odnosi između objekata kao instanci klasa. Uspostavljaju su između objekata, kao i veze objekata i atributa (*Class Assertions*). Ograničenja relacija se iskazuju kroz minimalni i maksimalni kardinalitet, egzistencijalno ili univerzalno kvantifikovanje osobina između objekata (*Ranges*), kao i restrikcije na osobine podataka ili domene. Ostale, opcione karakteristike relacija su: identične osobine objekata, inverzne nadređene, disjunktne osobine objekata ili pak lance međusobno povezanih osobina objekata. Ove karakteristike ontologije su deo rečnika i ne prikazuju se na dijagramu.

Na slikama 5.11., 5.12. i 5.13 prikazane su relacije između jednog, dva i tri objekta. Objekat može biti u relaciji sa samim sobom, kao što je to moguće i u drugim modelima (Slika 5.11.).



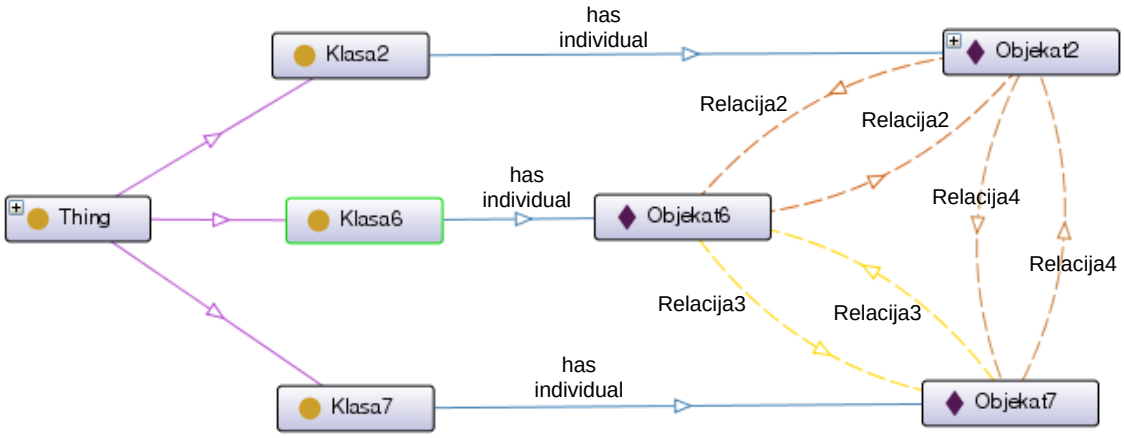
Slika 5.11. – Veza objekta sa samim sobom

Na slici 5.12. je prikazana veza između dva objekta: „Objekat 1“ i „Objekat 2“, dok je naziv relacije između ovih objekata „Relacija 1“. U grafu koji prikazuje deo ontologije su prikazana oba smera binarnih relacija.



Slika 5.12. – Relacija između dva objekta ontologije

Na slici 5.13. su prikazane binarne veze između tri objekta: „Objekat 2“ i „Objekat 6“ su povezani preko „Relacija 2“, Objekat 6“ i „Objekat 7“ preko „Relacija 3“, a sa „Relacija 4“ se ostvaruje veza između „Objekat 2“ i „Objekat 7“.



Slika 5.13. – Relacija između tri objekta u ontologiji

5.3.1. Mapiranje ontologije na predikatski račun prvog reda

Jedan od najčešće korišćenih načina predstavljanja ontologija jeste RDF radni okvir. Kao što je u teoretskom istraživanju navedeno u [Antoniou et al., 2005] i [6], RDF izraz je kolekcija trojki, od kojih se svaka sastoji od subjekta, predikata i objekta. Elementi skupa S su subjekti, skupa P su predikati, a skupa O objekti RDF ontologije. Članovi skupova S, P i O se mogu predstaviti kao termi atomarne formule $R(S, P, O)$. Mapiranje ontologije na predikatski račun se sastoji iz transformacije XML zapisa ontologije u cilju identifikacije elemenata skupova S, P i O. Na osnovu definicije 3.9, elementi ova tri skupa predstavljaju terme koji se sastoje od konsanti. Ako je R n-arni predikatski simbol (definicija 3.10) i elementi S, P i O su termi, tada je $R(S, P, O)$ atomarna formula, tj. literal (prema definiciji 3.11). Ovakva atomarna formula se ne kvantifikuje, s obzirom da ne sadrži promenljive.

Mapiranje OWL klase na predikatski račun prvog reda:

S = subj:'Klasa1',
P = rdf:type,
O = owl:'Class' ;

R(S, P, O) ⇒ R(Klasa1, type, Class)

Mapiranje OWL podklase na predikatski račun prvog reda:

S = subj:'Klasa4',
P = rdfs:subClassOf,
O = subj:'Klasa3' ;

R(S, P, O) ⇒ R(Klasa4, subClassOf, Klasa3)

Mapiranje OWL relacije na predikatski račun prvog reda:

S = subj:Relacija1,
P = rdf:type,
O = owl:'ObjectProperty' ;

R(S, P, O) ⇒ R(Relacija1, type, ObjectProperty)

Definisanje ekvivalentnih OWL relacija u formi predikatskog računa:

S = subj:Relacija4,
P = owl:equivalentProperty,
O = subj:Relacija5 ;

R(S, P, O) ⇒ R(Relacija4, equivalentProperty, Relacija5)

Mapiranje OWL atributa na predikatski račun prvog reda:

S = subj:Atribut1,
P = rdf:type,
O = owl:'DatatypeProperty' ;

R(S, P, O) ⇒ R(Atribut1, type, DatatypeProperty)

Transformacija OWL tipova podataka na predikatski račun:

S = subj:Atribut1,
P = rdfs:range,
O = xsd:integer ;

R(S, P, O) ⇔ R(Atribut1, range, integer)

Mapiranje OWL instanci klasa na predikatski račun:

S = subj:Objekat1,
P = rdf:type,
O = subj:'NamedIndividual';

R(S, P, O) ⇔ R(Objekat1, type, NamedIndividual)

Mapiranje OWL relacije između objekata na predikatski račun:

S = subj:Objekat1,
P = subj:Relacija1,
O = subj:Objekat2 ;

R(S, P, O) ⇔ R(Objekat1, Relacija1, Objekat2)

Mapiranje OWL ograničenja relacija između objekata na predikatski račun:

S = subj:Objekat1,
P = subj:Relacija1,
O = rdfs:mincard;

R(S, P, O) ⇔ R(Objekat1, Relacija1, mincard)

S = subj:Objekat1,
P = subj:Relacija1,
O = rdfs:maxcard;

R(S, P, O) ⇔ R(Objekat1, Relacija1, maxcard)

S = subj:Objekat2,
P = subj:Relacija1,
O = rdfs:mincard;

R(S, P, O) ⇔ R(Objekat2, Relacija1, mincard)

S = subj:Objekat2,
P = subj:Relacija1,
O = rdfs:maxcard;

R(S, P, O) ⇔ R(Objekat2, Relacija1, maxcard)

5.3.2. Kreiranje Prolog rečenica za mapiranu ontologiju

Dalja transformacija elemenata ontologije u oblik rečenica Prolog jezika (na osnovu 3.7 i 3.9) je izvršena tako što se predikat *R* preslikava u „rdf" predikat Prolog klauzula. Nazivi elemenata skupa subjekata *S*, predikata *P* i objekata *O* su argumenti literala koji sadrže konstante, te se moraju navesti malim početnim slovima. Dodavanjem tačke na kraju literala, kao sintaksnom oznakom za kraj rečenice, formirane su Prolog činjenice. To su, kao i u slučaju formalizovanog modela podataka, klauzule sa praznim telom, koje se sastoje isključivo od glave pravila (3.9), (3.11), (3.12), [Hotomski 2003], [Tošić i sar.]:

R(Klasa1, type, Class)⇒ rdf(klasa1, type, class).

R(Klasa4, subClassOf, Klasa3) ⇒ rdf(klasa4, subclassof, klasa3).

R(Relacija1, type, ObjectProperty) ⇒ rdf(Relacija1, type, ObjectProperty).

R(Relacija4, equivalentProperty, Relacija5) ⇒ rdf(relacija4, equivalentproperty, relacija5).

R(Atribut1, type, DatatypeProperty) ⇒ rdf(atribut1, type, datatypeproperty).

R(Atribut1, range, integer) ⇒ rdf(atribut1, range, integer).

R(Objekat1, type, NamedIndividual) ⇒ rdf(objekat1, type, namedindividual).

R(Objekat1, Relacija1, Objekat2) ⇒ rdf(objekat1, relacija1, objekat2).

R(Objekat1, Relacija1, mincard) ⇒ rdf(objekat1, relacija1, mincard).

R(Objekat1, Relacija1, maxcard) ⇒ rdf(objekat1, relacija1, maxcard).

R(Objekat2, Relacija1, mincard) ⇒ rdf(objekat2, relacija1, mincard).

R(Objekat2, Relacija1, maxcard) ⇒ rdf(objekat2, relacija1, maxcard).

Nakon postupka mapiranja OWL/RDF ontologije na predikatski račun prvog reda i prevođenja literala u Prolog činjenice, formira se skup Prolog rečenica (Listing 5.1) koji opisuje ontologiju i prikazuje elemente OWL/RDF ontologije:

- Klase (*Class*),
- Podklase (*Subclass*),
- Osobine objekata (*Object Property*),
- Ekvivalentne osobine objekata (*Equivalent Object Properties*),
- Atribute (*Data Property*),
- Objekte kao instance klasa (*Named Individual*),
- Tipove podataka (*Data Range*),
- Veze objekata i klasa (*Class Assertion*),
- Veze između atributa i tipova podataka (*Data Property Assertion*),
- Veze između različith objekata,
- Ograničenja veza između objekata (*Object Property Range*).

Listing 5.1:

```
rdf(klasa1, type, class).
rdf(klasa2, type, class).
rdf(klasa3, type, class).
rdf(klasa4, type, class).
rdf(klasa5, type, class).
rdf(klasa6, type, class).
rdf(klasa7, type, class).
rdf(relacija1, type, objectproperty).
rdf(relacija2, type, objectproperty).
rdf(relacija3, type, objectproperty).
rdf(relacija4, type, objectproperty).
rdf(relacija5, type, objectproperty).
rdf(relacija6, type, objectproperty).
rdf(atribut1, type, dataproperty).
rdf(atribut2, type, dataproperty).
rdf(atribut3, type, dataproperty).
rdf(atribut4, type, dataproperty).
rdf(atribut5, type, dataproperty).
rdf(atribut6, type, dataproperty).
rdf(atribut7, type, dataproperty).
rdf(atribut8, type, dataproperty).
rdf(objekat1, type, namedindividual).
rdf(objekat2, type, namedindividual).
rdf(objekat3, type, namedindividual).
rdf(objekat4, type, namedindividual).
rdf(objekat5, type, namedindividual).
rdf(objekat6, type, namedindividual).
rdf(objekat7, type, namedindividual).
```

Listing 5.1 (nastavak):

```
rdf(relacija4, equivalentobjectproperties, relacija5).
rdf(klasa4, subclass, klasa3).
rdf(klasa5, subclass, klasa3).
rdf(atribut1, range, integer).
rdf(atribut2, range, string).
rdf(atribut3, range, datetime).
rdf(atribut4, range, integer).
rdf(atribut5, range, string).
rdf(atribut6, range, int).
rdf(atribut7, range, string).
rdf(atribut8, range, long).
rdf(objekat1, classassertion, klasa1).
rdf(objekat2, classassertion, klasa2).
rdf(objekat3, classassertion, klasa3).
rdf(objekat4, classassertion, klasa4).
rdf(objekat5, classassertion, klasa5).
rdf(objekat6, classassertion, klasa6).
rdf(objekat7, classassertion, klasa7).
rdf(atribut1, datapropertyassertion, objekat1).
rdf(atribut2, datapropertyassertion, objekat1).
rdf(atribut3, datapropertyassertion, objekat1).
rdf(atribut4, datapropertyassertion, objekat2).
rdf(atribut5, datapropertyassertion, objekat2).
rdf(atribut1, datapropertyassertion, objekat3).
rdf(atribut2, datapropertyassertion, objekat3).
rdf(atribut3, datapropertyassertion, objekat3).
rdf(atribut7, datapropertyassertion, objekat4).
rdf(atribut6, datapropertyassertion, objekat5).
rdf(atribut1, datapropertyassertion, objekat6).
rdf(atribut2, datapropertyassertion, objekat6).
rdf(atribut7, datapropertyassertion, objekat7).
rdf(atribut8, datapropertyassertion, objekat7).
rdf(objekat2, relacija1, objekat1).
rdf(objekat1, relacija6, objekat1).
rdf(objekat1, relacija1, objekat2).
rdf(objekat6, relacija2, objekat2).
rdf(objekat7, relacija4, objekat2).
rdf(objekat2, relacija2, objekat6).
rdf(objekat7, relacija3, objekat6).
rdf(objekat6, relacija3, objekat7).
rdf(objekat2, relacija4, objekat7).
rdf(objekat1, relacija1, mincard).
rdf(objekat1, relacija1, maxcard).
rdf(objekat2, relacija1, mincard).
rdf(objekat2, relacija1, maxcard).
rdf(objekat6, relacija2, mincard).
rdf(objekat6, relacija2, maxcard).
rdf(objekat2, relacija2, mincard).
rdf(objekat2, relacija2, maxcard).
rdf(objekat6, relacija3, mincard).
rdf(objekat6, relacija3, maxcard).
rdf(objekat7, relacija3, mincard).
rdf(objekat7, relacija3, maxcard).
rdf(objekat2, relacija4, mincard).
rdf(objekat2, relacija4, maxcard).
rdf(objekat7, relacija4, mincard).
rdf(objekat7, relacija4, maxcard).
rdf(objekat1, relacija6, mincard).
rdf(objekat1, relacija6, maxcard).
```

5.4. PRAVILA ZAKLJUČIVANJA

Na osnovu teoretskog istraživanja, pregleda vladajućih stavova u oblasti istraživanja, prethodno prikazanih modela formalizacije ER modela podataka i mapiranja ontologije u formu predikatskih i Prolog rečenica, formirana su pravila zaključivanja koja definišu korektnost modela podataka sa aspekta specifične semantike u određenom domenu.

[Van Belle 2006] daje okvir za semantičku ocenu modela kroz kriterijume i predložene metrike za ocenu modela u odnosu na dati kriterijum. [Batini et al., 2006] su naveli najvažnije dimenzije kvaliteta koje se koriste u metrikama modela. Na osnovu ovih istraživanja može se tvrditi da je konceptualni model podataka korektan ako su zadovoljeni uslovi: postoji usklađenost modela podataka i ontologije - generičnost, pokrivenost domena i osnovnih koncepata, kompletnost rečnika podataka, konzistentnost, ekspresivnost, čitljivost, efikasnost i konciznost modela.

[Moody 1998] je definisao skup metrika u kojima su precizirani faktori kvaliteta: kompletnost, integritet, fleksibilnost, razumljivost, korektnost, jednostavnost, integracija i primenljivost. [Gray et al., 1991] su predložili metrike za evaluaciju kvaliteta ER dijagrama: kompleksnost ER dijagrama, kompleksnost entiteta i kompleksnost arhitekture podataka nekog entiteta.

[El-Ghalayini et al., 2005] su predložili skup pravila za mapiranje kako bi se generisao konceptualni model podataka na osnovu ontologije. Stvar, tj. pojavu objekta u ontološkom jeziku mapiraju u entitet u konceptualnom modelu podataka, ontološka klasa u tip entiteta, osobina klase u ontološkom jeziku predstavlja atribut entiteta, svako ograničenje osobine u ontološkom jeziku se preslikava u ograničenje relacije u konceptualnom modelu podataka koje se odnosi na vrstu relacije, broj entiteta ili tipova entiteta u relaciji. Na osnovu istraživanja [El-Ghalayini et al., 2005], [Gray et al., 1991], [Moody 1998], [Batini et al., 2006], [Volz et al.] i [Van Belle 2006] formirana je semantička veza između elemenata ER modela podataka i domenske ontologije.

Tabela 5.1: Pravila zaključivanja za semantičku verifikaciju konceptualnog modela podataka

Oznaka pravila	Prolog pravilo	Vrste objekata u okviru ontologije	Vrsta objekata u modelu podataka	Semantički kriterijum validacije
R 5.1	PR 5.1	klase	entiteti	pokrivenost osnovnih koncepata
R 5.2	PR 5.2	klase	entiteti	pokrivenost osnovnih koncepata
R 5.3	PR 5.3	osobine podataka	atributi	pokrivenost osnovnih koncepata
R 5.4	PR 5.4	osobine podataka	atributi	pokrivenost osnovnih koncepata
R 5.5	PR 5.5	osobine objekata klasa	relacije	pokrivenost osnovnih koncepata
R 5.6	PR 5.6	osobine objekata klasa	relacije	pokrivenost osnovnih koncepata

R 5.7	PR 5.7	osobine podataka i opsezi	atributi i tipovi podataka	pokrivenost domena i osnovnih koncepata, kompletnost rečnika, generičnost
R 5.8	PR 5.8	osobine podataka i opsezi	atributi i tipovi podataka	pokrivenost domena i osnovnih koncepata, kompletnost rečnika, generičnost
R 5.9	PR 5.9	klase i osobine podataka	entiteti i atributi	pokrivenost domena i osnovnih koncepata, kompletnost rečnika, generičnost
R 5.10	PR 5.10	osobine objekata klasa	relacije	generičnost
R 5.11	PR 5.11	osobine objekata klasa	relacije	generičnost
R 5.12	PR 5.12	osobine objekata klasa	relacije	generičnost
R 5.13	PR 5.13	osobine objekata klasa	relacije	generičnost
R 5.14	PR 5.14	osobine objekata klasa	relacije	generičnost
R 5.15	PR 5.15	osobine objekata klasa	relacije, entiteti, mešoviti objekat veza	generičnost
R 5.16	PR 5.16	osobine objekata klasa	relacije, entiteti, mešoviti objekat veza	generičnost
R 5.17	PR 5.17	nadklase i podklase	is_a hijerarhija	pokrivenost osnovnih koncepata
R 5.18	PR 5.18	nadklase i podklase	is_a hijerarhija	generičnost
R 5.19	PR 5.19	nadklase i podklase	is_a hijerarhija	generičnost
R 5.20	PR 5.20	nadklase i podklase	is_a hijerarhija	pokrivenost osnovnih koncepata
R 5.21	PR 5.21	klase	entiteti	generičnost
R 5.22	PR 5.22	osobine objekata klasa	relacije	generičnost
R 5.23	PR 5.23	opsezi osobina objekata klasa	kardinalitet poveznika/relacija	generičnost
R 5.24	PR 5.24	opsezi osobina objekata klasa	kardinalitet poveznika/relacija	generičnost

Pravilo R 5.1 - Za svaku klasu ontologije mora, u konceptualnom modelu podataka, da postoji odgovarajući tip entiteta opisan svojim imenom, prema definicijama 3.2 i 3.5:

$$(\forall x)\,(RDF(x, type, class) \wedge Ent(x) \Rightarrow OntoClassEnt(x)) \qquad (R\ 5.1)$$

gde je: x promenljiva, «type» i «class» oznake konstanti iz ontologije, «RDF» je ontološki predikat, «Ent» predikat iz modela podataka i «OntoclassEnt» oznaka predikata. Prolog forma pravila R 5.1 formirana je transformacijom na osnovu (3.7), (3.8), (3.11), (3.12), (3.13) i (3.14) :

```
ontoclassent(X):-rdf(X,type,class),ent(X).        (PR 5.1)
```

Pravilo R 5.2 – Služi za određivanje klasa ontologije koje nisu «pokrivene», tj. za koje ne postoje definisani tipovi entiteta u konceptualnom modelu podataka:

$$(\forall x)\,(RDF(x, type, class) \wedge \neg Ent(x) \Rightarrow OntoClassNoEnt(x)) \qquad (R\ 5.2)$$

gde je: x promenljiva iz ontologije, «type» i «class» oznake konstanti iz ontologije, «RDF» je ontološki predikat, «Ent» predikat iz modela podataka i «OntoClassNoEnt» oznaka predikata. Prolog forma za R 5.2, prema (3.7), (3.8), (3.11), (3.12), (3.13) i (3.14):

```
ontoclassnoent(X):-rdf(X,type,class),not ent(X).  (PR 5.2)
```

Semantička pravila R 5.1 i R 5.2 koja se koriste za proveru pokrivenosti klasa tipovima entiteta nisu uvek dovoljan i potpun uslov za davanje odgovora da li su sve klase ontologije pokrivene entitetima, s obzirom da projektanti baza podataka mogu koristiti različite ili slične nazive za pojedine tipove entiteta. Ukoliko se tipovi entiteta razlikuju u nazivu koji treba da odražava semantiku sistema koji se projektuje, prilikom zaključivanja u Prolog sistemu neće se izvršiti unifikacija, pa pojedine ontološke klase mogu ostati nepokrivene tipovima entiteta u konceptualnom modelu, iako predstavljaju usklađene vrste objekata na nivou oba modela.

Pravilo R 5.3 – Svaka osobina podataka koja se definiše u ontološkom jeziku predstavlja atribut entiteta u konceptualnom modelu podataka [El-Ghalayini et al., 2005]:

$$(\forall x)\,(RDF(x, type, dataproperty) \wedge Atr(x) \Rightarrow OntoDataAtrib(x)) \qquad (R\ 5.3)$$

gde je: x promenljiva, «type» i «dataproperty» oznake konstanti iz ontologije, «RDF» je ontološki predikat, «Atr» predikat iz modela podataka i «OntoDataAtrib» oznaka predikata. Prolog forma za R 5.3, prema (3.7), (3.8), (3.11), (3.12), (3.13) i (3.14):

```
ontodataatrib(X):-rdf(X,type,dataproperty),atr(X).  (PR 5.3)
```

Pravilo R 5.4 – Služi za određivanje osobina podataka iz ontologije koje nisu pokrivena atributima u konceptualnom modelu podataka:

$$(\forall x)\,(RDF(x, type, dataproperty) \wedge \neg Atr(x) \Rightarrow OntoDataNoAtrib(x)) \qquad (R\ 5.4)$$

gde je: x promenljiva, «type» i «dataproperty» oznake konstanti iz ontologije, «RDF» je ontološki predikat, «Atr» predikat iz modela podataka i «OntoDataNoAtrib» oznaka predikata. Prolog forma za R 5.4, prema (3.7), (3.8), (3.11), (3.12), (3.13) i (3.14):

```
ontodatanoatrib(X):-rdf(X,type,dataproperty),not atr(X).  (PR 5.4)
```

Pravila R 5.3 i R 5.4, predstavljaju vezu ontoloških osobina podataka i atributa entiteta u konceptualnom modelu podataka, koja je ustanovljena na osnovu naziva, kao što je to urađeno u pravilima R 5.1 i R 5.2. Pri tome se ne uzimaju u obzir adekvatni domeni atributa, tipovi podataka i ograničenja, a kako [El-Ghalayini et al., 2005] navode da se u obzir se moraju uzeti i ograničenjima koja se odnose na ontološke osobine, tj. atribute u modelu podataka. Zbog toga je formirano pravilo R 5.7. koje se odnosi na proveru tipova podataka.

Pravilo R 5.5 – Svaka ontološka osobina objekta se preslikava u relaciju modela podataka [El-Ghalayini et al., 2005]:

$$(\forall x)\,(RDF(x, type, objectproperty) \wedge Rel(x) \Rightarrow OntoObjPropRel(x)) \qquad (R\ 5.5)$$

gde je: x promenljiva, «type» i «objectproperty» oznake konstanti iz ontologije, «RDF» je ontološki predikat, «Rel» predikat iz modela podataka i «OntoObjPropRel» oznaka predikata. Prolog forma za R 5.5, prema (3.7), (3.8), (3.11), (3.12), (3.13) i (3.14):

```
ontoobjproprel(X):-rdf(X,type,objectproperty),rel(X).        (PR 5.5)
```

Pravilo R 5.6 – Služi za izdvajanje ontoloških osobina objekata koje se ne preslikavaju u relacije modela podataka:

$$(\forall x)\,(RDF(x, type, objectproperty) \wedge \neg Rel(x) \Rightarrow OntoObjPropNoRel(x)) \qquad (R\ 5.6)$$

gde je: x promenljiva, «type» i «objectproperty» oznake konstanti iz ontologije, «RDF» je ontološki predikat, «Rel» predikat iz modela podataka i «OntoObjPropNoRel» oznaka predikata. Prolog forma za pravilo R 5.6, prema (3.7), (3.8), (3.11), (3.12), (3.13) i (3.14):

```
ontoobjpropnorel(X):-rdf(X,type,objectproperty),not rel(X).(PR 5.6)
```

Ista karakteristika koja je navedena u slučaju ontološke povezanosti entiteta i atributa (pravila R 5.1, R 5.2, R 5.3 i R 5.4), važi i za relacije, tako da je problem sinonima izuzetno prisutan. Ovo može značajno uticati na krajnju procenu semantičke korektnosti modela podataka, tako da su definisana posebna pravila zaključivanja koja prevazilaze u određenoj meri problem i koja se odnose na složeniju semantičku vezu ER modela podataka i domenske ontologije.

Pravilo R 5.7 – Svaka osobina podataka koja se definiše u ontološkom jeziku predstavlja atribut entiteta u konceptualnom modelu podataka u skladu sa ograničenjima koja se odnose na osobinu. Za svaku ontološku osobinu podataka je definisan opseg podataka koji treba da je usklađen sa tipom podatka koji je pridružen atributu u modelu podataka preko elemenata skupa ograničenja *S*:

$$(\forall x)(\exists y)(\exists z)\,(RDF(x, type, dataproperty) \wedge RDF(x, range, y) \wedge Atr(x) \wedge Res(z) \wedge P(x,z) \wedge DataType(y,z) \Rightarrow OntoDataAtribType(x)) \qquad (R\ 5.7)$$

gde su: x, y, z promenljive, «type», «dataproperty» i «range» oznake konstanti iz ontologije, «RDF» je ontološki predikat, «Res», «P» i «Atr» predikati iz modela podataka, «DataType» je predikat za povezivanje tipova podataka u ontologiji i modelu podataka, a «OntoDataAtribType» oznaka predikata. Prolog forma pravila R 5.7, prema (3.7), (3.8), (3.11), (3.12), (3.13) i (3.14), je:

```
ontodataatribtype(X,Y):-rdf(X,type,dataproperty),
      rdf(X,range,Y),atr(X),res(Z),p(X,Z),datatype(Y,Z).      (PR 5.7)
```

Pravilo R 5.8 – Prikaz opsega ontoloških osobina podataka koji nemaju identičan naziv atributu u modelu podataka, ali imaju odgovarajući tip podatka (sinonimi za nazive atributa):

$$(\forall x)(\forall y)(\forall x_1)(\forall y_1)\,(RDF(x, type, dataproperty) \wedge RDF(x, range, y) \wedge Atr(x_1) \wedge Res(y_1) \wedge P(x_1, y_1) \wedge DataType(y, y_1) \wedge \neg x=x_1 \Rightarrow OntoDataAtribTypeSin(x,y,x_1,y_1)) \quad (R\ 5.8)$$

gde su: x, y, x_1, y_1 promenljive, «type», «dataproperty» i «range» oznake konstanti iz ontologije, «RDF» je ontološki predikat, «Res», «P» i «Atr» predikati iz modela podataka, «DataType» je predikat za povezivanje tipova podataka u ontologiji i modelu podataka, a «OntoDataAtribTypeSin» oznaka predikata. Prolog forma za pravilo R 5.8, prema (3.7), (3.8), (3.11), (3.12), (3.13) i (3.14), je:

```
ontodataatribtypesin(X,Y,X1,Y1):-rdf(X,type,dataproperty),
      rdf(X,range,Y),atr(X1),res(Y1),p(X1,Y1),
      datatype(Y,Y1),not X=X1.                                (PR 5.8)
```

Pravilo R 5.9 – Izdvajanje svih onih skupova ontoloških osobina podataka objekata neke klase koji odgovaraju atributima entiteta koji semantički odgovaraju ontološkoj klasi:

$$(\forall x)(\forall y)(\exists z)\,(RDF(x, type, class) \wedge RDF(y,type,dataproperty) \wedge RDF(Z,type,namedindividual) \wedge RDF(Z,classassertion,X) \wedge RDF(Y,datapropertyassertion,z) \wedge Ent(x) \wedge Atr(y) \wedge P(x,y) \Rightarrow OntoClassEntDataAtrib(x,y)) \quad (R\ 5.9)$$

gde su: x, y, z promenljive, «type», «class», «namedindividual», «classassertion» i «datapropertyassertation» oznake konstanti iz ontologije, «RDF» je ontološki predikat, «Ent», «P» i «Atr» predikati iz modela podataka, a «OntoClassEntDataAtrib» oznaka predikata. Prolog forma pravila R 5.9, prema (3.7), (3.8), (3.11), (3.12), (3.13) i (3.14):

```
ontoclassentdataatrib(X,Y):-rdf(X,type,class),
      rdf(Y,type,dataproperty),rdf(Z,type,namedindividual),
      rdf(Z,classassertion,X),rdf(Y,datapropertyassertion,Z),
      ent(X),atr(Y),p(X,Y).                                   (PR 5.9)
```

Pravilo R 5.10 – Izdvajanje svih ontoloških osobina objekata klase koji odgovaraju relacijama u modelu podatka, uspostavljenim između entiteta koji semantički odgovaraju ontološkim klasama:

$$(\forall xc_1)(\forall xc_2)(\forall yop)(\forall yr)(\forall xe_1)(\forall xe_2)(\exists xo_1)(\exists xo_2)\,(RDF(xc_1,type,class) \wedge RDF(xo_1,classassertion,xc_1) \wedge RDF(xo_1,type,namedindividual) \wedge RDF(xc_2,type,class) \wedge RDF(xo_2,classassertion,xc_2) \wedge RDF(xo_2,type,namedindividual) \wedge RDF(yop,type,objectproperty) \wedge RDF(xo_1,yop,xo_2) \wedge Ent(xe_1) \wedge Ent(xe_2) \wedge Rel(yr) \wedge P(xe_1,yr) \wedge P(yr,xe_2) \wedge xe_1=xc_1 \wedge xe_2=xc_2 \wedge yr=yop \Rightarrow OntoRel(xc_1,yop,xc_2,xe_1,yr,xe_2)) \quad (R\ 5.10)$$

gde su: xc_1, xc_2, yop, yr, xe_1, xe_2, xo_1, xo_2 promenljive, a «type», «class», «namedindividual», «classassertion» i «objectproperty» oznake konstanti iz ontologije, «RDF» je ontološki predikat, «Ent», «P» i «Rel» predikati iz modela podataka, a

«OntoRel» je oznaka predikata. Prolog forma za pravilo R 5.10, prema (3.7), (3.8), (3.11), (3.12), (3.13) i (3.14), je:

```
ontorel(XC1,YOP,XC2,XE1,YR,XE2):-rdf(XC1,type,class),
      rdf(XO1,classassertion,XC1),rdf(XO1,type,namedindividual),
      rdf(XC2,type,class),rdf(XO2,classassertion,XC2),
      rdf(XO2,type,namedindividual),rdf(YOP,type,objectproperty),
      rdf(XO1,YOP,XO2),ent(XE1),ent(XE2),rel(YR),p(XE1,YR),
      p(YR,XE2),XE1=XC1,XE2=XC2,YR=YOP.                    (PR 5.10)
```

Za pronalaženje sinonima relacija koje su uspostvaljene između određenih tipova entiteta su definisana pravila 5.11, 5.12, 5.13 i 5.14.

Pravilo R 5.11 – Izdvajanje svih ontoloških osobina objekata klase koji se po nazivu razlikuju od relacija uspostavljenih između entiteta koji semantički odgovaraju ontološkim klasama, pri čemu se ne izdvajaju relacije već pronađene pravilom R 5.10 (sinonimi za relacije u modelu podataka):

$$(\forall xc_1)(\forall xc_2)(\forall yop)(\forall yr)(\forall xe_1)(\forall xe_2)(\exists xo_1)(\exists xo_2)\ (RDF(xc_1,type,class) \wedge RDF(xo_1,classassertion,xc_1) \wedge RDF(xo_1,type,namedindividual) \wedge RDF(xc_2,type,class) \wedge RDF(xo_2,classassertion,xc_2) \wedge RDF(xo_2,type,namedindividual) \wedge RDF(yop,type,objectproperty) \wedge RDF(xo_1,yop,xo_2) \wedge Ent(xe_1) \wedge Ent(xe_2) \wedge Rel(yr) \wedge P(xe_1,yr) \wedge P(yr,xe_2) \wedge xe_1=xc_1 \wedge xe_2=xc_2 \wedge \neg OntoRel(xc_1,yop,xc_2,xe_1,yr,xe_2) \Rightarrow OntoRelSinRel(xc_1,yop,xc_2,xe_1,yr,xe_2)) \quad (R\ 5.11)$$

gde su: xc_1, xc_2, yop, yr, xe_1, xe_2, xo_1, xo_2 promenljive, a «type», «class», «namedindividual», «classassertion» i «objectproperty» oznake konstanti iz ontologije, «RDF» je ontološki predikat, «Ent», «P» i «Rel» predikati iz modela podataka, «OntoRel» i «OntoRelSinRel» oznake predikata. Prolog forma pravila R 5.11, prema (3.7), (3.8), (3.11), (3.12), (3.13) i (3.14), je:

```
ontorelsinrel(XC1,YOP,XC2,XE1,YR,XE2):- rdf(XC1,type,class),
      rdf(XO1,classassertion,XC1),rdf(XO1,type,namedindividual),
      rdf(XC2,type,class),rdf(XO2,classassertion,XC2),
      rdf(XO2,type,namedindividual),rdf(YOP,type,objectproperty),
      rdf(XO1,YOP,XO2),ent(XE1),ent(XE2),rel(YR),
      p(XE1,YR),p(YR,XE2),XE1=YC1,XE2=XC2,
      not ontorel(XC1,YP,XC2,XE1,YR,XE2).                  (PR 5.11)
```

Pravilo R 5.12 – Izdvajanje svih ontoloških osobina objekata klase koji odgovaraju relacijama uspostavljenim između entiteta od kojih jedan nije pokriven ontološkom klasom po nazivu a drugi jeste, pri čemu se ne izdvajaju relacije već pronađene pravilima R 5.10 i R5.11 (sinonimi za jedan od entiteta):

$$(\forall xc_1)(\forall xc_2)(\forall yop)(\forall yr)(\forall xe_1)(\forall xe_2)(\exists xo_1)(\exists xo_2)\ (RDF(xc_1,type,class) \wedge RDF(xo_1,classassertion,xc_1) \wedge RDF(xc_1,type,namedindividual) \wedge RDF(xc_2,type,class) \wedge RDF(xo_2,classassertion,xc_2) \wedge RDF(x_2,type,namedindividual) \wedge RDF(yop,type,objectproperty) \wedge RDF(xo_1,p,xo_2) \wedge Ent(xe_1) \wedge Ent(xe_2) \wedge Rel(yr) \wedge P(xe_1,yor) \wedge P(yr,xe_2) \wedge (xe_1=xc_1 \vee xe_2=xc_2) \wedge yr=yop \wedge \neg OntoRel(xc_1,yop,xc_2,xe_1,yr,xe_2) \Rightarrow OntoRelSinEnt(xc_1,yop,xc_2,xe_1,yr,xe_2)) \quad (R\ 5.12)$$

gde su: xc_1, xc_2, yop, yr, xe_1, xe_2, xo_1, xo_2 promenljive, a «type», «class», «namedindividual», «classassertion» i «objectproperty» oznake konstanti iz ontologije, «RDF» je ontološki predikat, «Ent», «P» i «Rel» predikati iz modela podataka, dok su «OntoRel», «OntoRelSinRel» i «OntoRelSinEnt» oznake predikata. Prolog forma pravila R 5.12, prema (3.7), (3.8), (3.11), (3.12), (3.13) i (3.14):

```
ontorelsinent(XC1,YOP,XC2,XE1,YR,XE2):- rdf(XC1,type,class),
      rdf(XO1,classassertion,XC1),rdf(XO1,type,namedindividual),
      rdf(XC2,type,class),rdf(XO2,classassertion,XC2),
      rdf(XO2,type,namedindividual),rdf(YOP,type,objectproperty),
      rdf(XO1,YOP,XO2),ent(XE1),ent(XE2),rel(YR),p(XE1,YR),
      p(YR,XE2),(XE1=XC1;XE2=XC2),YR=YOP,
      not ontorel(XC1,YOP,XC2,XE1,YR,XE2),
      not ontorelsinrel(XC1,YOP,XC2,XE1,YR,XE2).          (PR 5.12)
```

Pravilo R 5.13 – Izdvajanje svih ontoloških osobina objekata klase koji odgovaraju relacijama uspostavljenim između entiteta koji nisu pokriveni ontološkim klasama po nazivu, pri čemu se ne izdvajaju relacije već pronađene pravilima R 5.10, R 5.11 i R5.12 (sinonimi za oba entiteta):

$$(\forall xc_1)(\forall xc_2)(\forall yop)(\forall yr)(\forall xe_1)(\forall xe_2)(\exists xo_1)(\exists xo_2)\ (RDF(xc_1,type,class) \wedge RDF(xo_1,classassertion,xc_1) \wedge RDF(xo_1,type,namedindividual) \wedge RDF(xc_2,type,class) \wedge RDF(xo_2,classassertion,xc_2) \wedge RDF(xo_2,type,namedindividual) \wedge RDF(yop,type,objectproperty) \wedge RDF(xo_1,yop,xo_2) \wedge Ent(xe_1) \wedge Ent(xe_2) \wedge Rel(yr) \wedge P(xe_1,yr) \wedge P(yr,xe_2) \wedge yr=yop \wedge \neg OntoRel(xc_1,yop,xc_2,xe_1,yr,xe_2) \wedge \neg OntoRelSinEnt(xc_1,yop,xc_2,xe_1,yr,xe_2) \Rightarrow OntoRelSinEnt2(xc_1,yop,xc_2,xe_1,yr,xe_2))$$ (R 5.13)

gde su: xc_1, xc_2, yop, yr, xe_1, xe_2, xo_1, xo_2 promenljive, a «type», «class», «namedindividual», «classassertion» i «objectproperty» oznake konstanti iz ontologije, «RDF» je ontološki predikat, «Ent», «P» i «Rel» predikati iz modela podataka, dok su «OntoRel», «OntoRelSinRel», «OntoRelSinEnt», «OntoRelSinEnt2» oznake predikata. Prolog forma za pravilo R 5.13, prema (3.7), (3.8), (3.11), (3.12), (3.13) i (3.14), je:

```
ontorelsinent2(XC1,YOP,XC2,XE1,YR,XE2):- rdf(XC1,type,class),
      rdf(XO1,classassertion,XC1),rdf(XO1,type,namedindividual),
      rdf(XC2,type,class),rdf(XO2,classassertion,XC2),
      rdf(XO2,type,namedindividual),rdf(YP,type,objectproperty),
      rdf(XO1,YOP,XO2),ent(XE1),ent(XE2),rel(YR),p(XE1,YR),
      p(YR,XE2),(XE1=XC1;XE2=XC2),YR=YOP,
      not ontorel(XC1,YOP,XC2,XE1,YR,XE2),
      not ontorelsinrel(XC1,YP,XC2,XE1,YR,XE2),
      not ontorelsinent(XC1,YOP,XC2,XE1,YR,XE2).          (PR 5.13)
```

Pravilo R 5.14 – Izdvaja sve ontološke osobine objekata klase koji se po nazivu razlikuju od relacija uspostavljenih između entiteta, od kojih je samo jedan jeste pokriven odgovarajućom ontološkom klasom po nazivu, dok se drugi entitet razlikuje, pri čemu se ne izdvajaju relacije već pronađene pravilima R 5.10, R 5.11, R 5.12 i R5.13 (sinonimi za relaciju i jedan entitet):

$$(\forall xc_1)(\forall xc_2)(\forall yop)(\forall yr)(\forall xe_1)(\forall xe_2)(\exists xo_1)(\exists xo_2)\ (RDF(xc_1,type,class) \wedge RDF(xo_1,classassertion,xc_1) \wedge RDF(xo_1,type,namedindividual) \wedge$$

$$RDF(xc_2,type,class) \wedge RDF(xo_2,classassertion,xc_2) \wedge RDF(xo_2,type,namedindividual) \wedge RDF(yop,type,objectproperty) \wedge RDF(xo_1,yop,xo_2) \wedge Ent(xe_1) \wedge Ent(xe_2) \wedge Rel(yr) \wedge P(xe_1,yr) \wedge P(yr,xe_2) \wedge (xe_1=xc_1 \vee xe_2=xc_2) \wedge \neg OntoRel(xc_1,yop,xc_2,xe_1,yr,xe_2) \wedge \neg OntoRelSinEnt(xc_1,yop,xc_2,xe_1,yr,xe_2) \wedge \neg ntoRelSinEnt2(xc_1,yop,xc_2,xe_1,yr,xe_2) \Rightarrow OntoRelSinEntRel(xc_1,yop,xc_2,xe_1,yr,xe_2)) \quad (R\ 5.14)$$

gde su: xc_1, xc_2, yop, yr, xe_1, xe_2, xo_1, xo_2 promenljive, a «type», «class», «namedindividual», «classassertion» i «objectproperty» oznake konstanti iz ontologije, «RDF» je ontološki predikat, «Ent», «P» i «Rel» predikati iz modela podataka, a «OntoRel», «OntoRelSinRel», «OntoRelSinEnt», «OntoRelSinEnt2», «OntoRelSinEntRel» su oznake predikata. Prolog forma za pravilo R 5.13, prema (3.7), (3.8), (3.11), (3.12), (3.13) i (3.14), je:

```
ontorelsinentrel(XC1,YOP,XC2,XE1,YR,XE2):- rdf(XC1,type,class),
      rdf(XO1,classassertion,XC1),rdf(XO1,type,namedindividual),
      rdf(XC2,type,class),rdf(XO2,classassertion,XC2),
      rdf(XO2,type,namedindividual),rdf(YOP,type,objectproperty),
      rdf(XO1,YOP,XO2),ent(XE1),ent(XE2),rel(YR),p(XE1,YR),
      p(YR,XE2),(XE1=XC1;XE2=XC2),not
      ontorel(XC1,YOP,XC2,XE1,YR,XE2),
      not ontorelsinrel(XC1,YOP,XC2,XE1,YR,XE2),
      not ontorelsinent(XC1,YOP,XC2,XE1,YR,XE2),
      not ontorelsinent2(XC1,YOP,XC2,XE1,YR,XE2).          (PR 5.14)
```

Pravilo R 5.15 – Izdvajanje svih ontoloških osobina objekata klase koji odgovaraju mešovitim objektima-veze, tj. gerundima u modelu podataka:

$$(\forall xc_1)(\forall xc_2)(\forall yop)(\forall yer)(\forall xe_1)(\forall xe_2)(\exists xo_1)(\exists xo_2)(\exists yr_1)(\exists yr_2)(\exists yid) (RDF(yop,type,objectproperty) \wedge RDF(xc_1,type,class) \wedge RDF(xo_1,classassertion,xc_1) \wedge RDF(xc_2,type,class) \wedge RDF(xo_2,classassertion,xc_2) \wedge RDF(xo_2,type,namedindividual) \wedge RDF(xo_1,yop,xo_2) \wedge Ent(xe_1) \wedge Ent(yer) \wedge Ent(xe_2) \wedge Rel(yr_1) \wedge Rel(yr_2) \wedge P(xe_1,yr_1) \wedge P(yr_1,yer) \wedge P(yer,yr_2) \wedge P(yr_2,xe_2) \wedge yop=yer \wedge \neg EntWithID(yer,yid) \Rightarrow OntoObjPropGer(xc_1,yop,xc_2,xe_1,yer,xe_2)) \quad (R\ 5.15)$$

gde su: xc_1, xc_2, yop, yer, xe_1, xe_2, xo_1, xo_2, yr_1, yr_2 i yid promenljive, a «type», «class», «namedindividual», «classassertion» i «objectproperty» oznake konstanti iz ontologije, «RDF» je ontološki predikat, «Ent», «P» i «Rel» predikati iz modela podataka, a «OntoObjPropGer» jeste oznaka predikata. Prolog forma za pravilo R 5.15, prema (3.7), (3.8), (3.11), (3.12), (3.13) i (3.14):

```
ontoobjpropger(XC1,YOP,XC2,XE1,YER,XE2):-
      rdf(YOP,type,objectproperty),rdf(XC1,type,class),
      rdf(XO1,classassertion,XC1),rdf(XC2,type,class),
      rdf(XO2,classassertion,XC2),rdf(XO2,type,namedindividual),
      rdf(XO1,YOP,XO2),ent(XE1),ent(YER),ent(XE2),rel(YR1),
      rel(YR2),p(XE1,YR1),p(YR1,YER),p(YER,YR2),p(YR2,XE2),
      YOP=YER,not entwithid(YER,YID).                      (PR 5.15)
```

S obzirom da se za predstavljanje gerunda (mešovitog objekta-veza) u skupu entiteta *E* nalaze nazivi ovih objekata iz modela podataka, pošto oni ne mogu imati svoj identifikacioni atribut, bilo je neophodno definisati pomoćno pravilo zaključivanja koje

nije semantičkog karaktera, a koje služi za identifikaciju entiteta koji imaju identifikacioni atribut. Negacijom ovog predikata izdvajaju se samo oni objekti tipa entiteta koji su nastali transformacijom običnog poveznika u mešoviti objekat-veza.

Pravilo R 5.15a – Izdvajanje svih entiteta u modelu podataka koji imaju definisan identifikacioni atribut:

$$(\forall x)(\forall y)\,(Ent(x) \wedge Atr(y) \wedge Res(idatr) \wedge P(x,y) \wedge P(y,idatr) \Rightarrow EntWithID(x,y)) \qquad \text{(R 5.15a)}$$

gde su: x, y promenljive, «idatr» oznaka konstante iz modela podataka, «Ent», «Atr», «Res» i «P» predikati iz modela podataka, a «EntWithID» oznaka predikata. Prolog forma pravila R 5.15a formirana je transformacijom na osnovu (3.7), (3.8), (3.11), (3.12), (3.13) i (3.14) :

```
entwithid(X,Y):-ent(X),atr(Y),res(idatr),p(X,Y),p(Y,idatr).
```

(PR 5.15a)

Pravilo R 5.16 – Izdvajanje svih ontoloških osobina objekata klase koji po nazivu ne odgovaraju mešovitim objektima-veze, tj. gerundima u modelu podataka, ali su uspostavljeni između entiteta za koje su definsane ontološke klase (sinonimi za gerunde), pri čemu se ne izdvajaju mešoviti objekti-veze već pronađeni pravilom R 5.15:

$$\begin{aligned}&(\forall xc_1)(\forall xc_2)(\forall yop)(\forall yer)(\forall xe_1)(\forall xe_2)(\exists xo_1)(\exists xo_2)(\exists yr_1)(\exists yr_2)\,(\exists yid)\\&\quad(RDF(yop,type,objectproperty) \wedge RDF(xc_1,type,class) \wedge\\&\quad RDF(xo_1,classassertion,xc_1) \wedge RDF(xc_2,type,class) \wedge RDF(xo_2,classassertion,xc_2)\\&\quad \wedge RDF(xo_1,type,namedindividual) \wedge RDF(xo_2,type,namedindividual) \wedge\\&\quad RDF(xo_1,yop,xo_2) \wedge Ent(xe_1) \wedge Ent(yer) \wedge Ent(xe_2) \wedge Rel(yr_1) \wedge Rel(yr_2) \wedge\\&\quad P(xe_1,yr_1) \wedge P(yr_1,yer) \wedge P(yer,yr_2) \wedge P(yr_2,xe_2) \wedge (xe_1=xc_1 \vee xe_2=xc_1) \wedge\\&\quad \neg EntWithID(yer,yid) \wedge \neg OntoObjPropGer(xc_1,yop,xc_2,xe_1,yer,xe_2) \Rightarrow\\&\quad OntoObjPropGerSin(xc_1,yop,xc_2,xe_1,yer,xe_2)) \qquad \text{(R 5.16)}\end{aligned}$$

gde su: xc_1, xc_2, yop, yer, xe_1, xe_2, xo_1, xo_2, yr_1, yr_2 i yid promenljive, a «type», «class», «namedindividual», «classassertion» i «objectproperty» oznake konstanti iz ontologije, «RDF» je ontološki predikat, «Ent», «P» i «Rel» predikati iz modela podataka, a «OntoObjPropGer» i «OntoObjPropGerSin» oznake predikata. Prolog forma za pravilo R 5.16, prema (3.7), (3.8), (3.11), (3.12), (3.13) i (3.14), je:

```
ontoobjpropgersin(XC1,YOP,XC2,XE1,YER,XE2):-
      rdf(YOP,type,objectproperty),rdf(XC1,type,class),
      rdf(XO1,classassertion,XC1),rdf(XC2,type,class),
      rdf(XO2,classassertion,XC2),rdf(XO1,type,namedindividual),
      rdf(XO2,type,namedindividual),rdf(XO1,YOP,XO2),ent(XE1),
      ent(YER),ent(XE2),rel(YR1),rel(YR2),p(XE1,YR1),p(XR1,XER),
      p(YER,YR2),p(YR2,XE2),(XE1=XC1;XE2=XC1),
      not entwithid(YER,YID),
      not ontoobjpropger(XC1,YOP,XC2,XE1,YER,XE2).
```

(PR 5.16)

Pravilo R 5.17 – Svaka relacija nadklasa/podklasa u ontologiji može se definisati kao IS_A hijerarhija između entiteta u modelu podataka [El-Ghalayini et al., 2005]:

$$\begin{aligned}&(\forall x)(\forall x1)(\forall x2)(\exists y)\,(RDF(x, type, class) \wedge RDF(x1, subclass, x) \wedge RDF(x2, subclass, x) \wedge\\&\quad Ent(x) \wedge Ent(x1) \wedge Ent(x2) \wedge P(x, y) \wedge P(y, x1) \wedge P(y, x2) \Rightarrow\\&\quad OntoSubclassIsa(x,x1,x2)) \qquad \text{(R 5.17)}\end{aligned}$$

gde su: x, x_1, x_2, y promenljive, «type», «class» i «subclass» oznake konstanti iz ontologije, «RDF» je ontološki predikat, «Ent» i «P» predikati iz modela podataka, a «OntoSubclassIsa» je oznaka predikata. Prolog forma pravila R 5.17 dobijena je transformacijom, na osnovu (3.7), (3.8), (3.11), (3.12), (3.13) i (3.14):

```
ontosubclassisa(X,X1,X2):-
        rdf(X,type,class),rdf(X1,subclass,X),rdf(X2,subclass,X),
        ent(X),ent(X1),ent(X2),p(X,Y),p(Y,X1),p(Y,X2),not X1=X2.
```

(PR 5.17)

Pravilo R 5.18 – Izdvajanje svih ontoloških nadklasa i podklasa za koje postoji odgovarajuća IS_A hijerarhija između entiteta u konceptualnom modelu podataka i ukoliko nazivi nadklase ontologije neodgovaraju nazivima entiteta, pri čemu se ne izdvajaju IS_A hijerarhije već pronađene pravilom R 5.17 (sinonimi za nadklasu):

$$(\forall xp)(\forall xc_1)(\forall xc_2)(\forall xe)(\forall xe_1)(\forall xe_2)(\exists y)\ (RDF(xp,type,class) \wedge RDF(xc_1,subclass,xp) \wedge RDF(xc_2,subclass,xp) \wedge Ent(xe) \wedge Ent(xe_1) \wedge Ent(xe_2) \wedge P(xe,y) \wedge P(y,xe_1) \wedge P(y,xe_2) \wedge \neg xc_1=xc_2 \wedge \neg xe_1=xe_2 \wedge \neg OntoSubclassIsa(xp,xc_1,xc_2) \wedge (\neg xe=xc_1 \vee \neg xe=xc_2) \Rightarrow OntoSubclassIsaSin(xp,xc_1,xc_2,xe,xe_1,xe_2)) \quad (R\ 5.18)$$

gde su: xp, xc_1, xc_2, xe, xe_1, xe_2, y promenljive, «type», «class» i «subclass» su konstante iz ontologije, «RDF» je ontološki predikat, «Ent» i «P» predikati iz modela podataka, a «OntoSubclassIsa» i «OntoSubclassIsaSin» su oznake predikata. Prolog forma pravila R 5.18 dobijena je transformacijom, na osnovu (3.7), (3.8), (3.11), (3.12), (3.13) i (3.14) :

```
ontosubclassisasin(XP,XC1,XC2,XE,XE1,XE2):-
        rdf(XP,type,class),rdf(XC1,subclass,XP),
        rdf(XC2,subclass,XP),ent(XE),ent(XE1),ent(XE2),p(XE,Y),
        p(Y,XE1),p(Y,XE2),not XC1=XC2,not XE1=XE2,
        not ontosubclassisa(XP,XC1,XC2),(not XE=XC1;not XE=XC2).
```

(PR 5.18)

Pravilo R 5.19 – Izdvajanje svih ontoloških nadklasa i podklasa za koje postoji odgovarajuća IS_A hijerarhija između entiteta u konceptualnom modelu podataka, pri čemu nazivi podklase ontologije nisu identični nazivima entiteta u modelu podataka (sinonimi za podklase):

$$(\forall x)(\forall x1)(\forall x2)(\exists y)(\exists z)\ (RDF(x1, subclass, z) \wedge RDF(x2, subclass, z) \wedge Ent(x) \wedge Ent(x1) \wedge Ent(x2) \wedge P(x, y) \wedge P(y, x1) \wedge P(y, x2) \wedge \neg x1=x2 \Rightarrow OntoSubclassIsaSinSub(x,x1,x2)) \quad (R\ 5.19)$$

gde su: x, x1, x2, y i z promenljive, «subclass» oznaka konstante iz ontologije, «RDF» je ontološki predikat, «Ent» i «P» predikati iz modela podataka, a «OntoSubclassIsaSinSub» je oznaka predikata. Prolog forma pravila R 5.19 dobijena je transformacijom na osnovu (3.7), (3.8), (3.11), (3.12), (3.13) i (3.14) :

```
ontosubclassisasinsub(X,X1,X2):-rdf(X1,subclass,Z),
        rdf(X2,subclass,Z),ent(X1),ent(X2),ent(X),p(X,Y),
        p(Y,X1),p(Y,X2),not X1=X2.
```

(PR 5.19)

Pravilo R 5.20 – Izdvajanje svih ontoloških nadklasa i podklasa za koje ne postoji odgovarajuća IS_A hijerarhija između entiteta u konceptualnom modelu podataka:

$$(\forall x)(\forall x1)(\forall x2)(RDF(x, type, class) \wedge RDF(x1, subclass, x) \wedge RDF(x2, subclass, x) \wedge \neg x1=x2 \wedge \neg OntoSubclassIsa(x,x1,x2) \Rightarrow OntoSubclassNoIsa(x,x1,x2)) \quad (R\ 5.20)$$

gde su x, x1, x2 promenljive, «type», «class» i «subclass» oznake ontoloških konstanti, «RDF» je ontološki predikat, a «OntoSubclassIsa» i «OntoSubclassNoIsa» su oznake predikata. Prolog forma pravila R5.20 dobijena je transformacijom, na osnovu (3.7), (3.8), (3.11), (3.12), (3.13) i (3.14):

```
ontosubclassnoisa(X,X1,X2):-rdf(X,type,class),
        rdf(X1,subclass,X),rdf(X2,subclass,X),not X1=X2,not
        ontosubclassisa(X,X1,X2).                          (PR 5.20)
```

Pravilo R 5.21 – Izdvajanje svih ontoloških klasa za koje ne postoje odgovarajući tipovi entiteta sa istim nazivom ali postoje entiteti sa sličnim nazivom pa se mogu odrediti sinonimi entiteta:

$$(\forall x)(\forall y)\ (\text{OntoClassNoEnt}(x) \wedge \text{Ent}(y) \wedge \neg\text{OntoClassEnt}(y) \Rightarrow \text{OntoClassEntSin}(x,y)) \qquad \text{(R 5.21)}$$

gde su: x, y promenljive, «type» i «class» oznake ontoloških konstanti, «RDF» je ontološki predikat, «Ent» predikat iz modela podataka, a «OntoClassNoEnt», «OntoClassEnt» i «OntoClassEntSin» i oznake predikata. Prolog forma pravila R5.21 formirana je transformacijom na osnovu (3.7), (3.8), (3.11), (3.12), (3.13) i (3.14) :

```
ontoclassentsin(X,Y):-ontoclassnoent(X),ent(Y),
        not ontoclassent(Y).                               (PR 5.21)
```

Pravilo R 5.22 – Izdvajanje svih ekvivalentnih ontoloških osobina objekata:

$$(\forall x)(\forall y)\ (\text{RDF}(x,\text{equivalentobjectproperties},y) \Rightarrow \text{OntoEqvObjprop}(x,y)) \qquad \text{(R 5.22)}$$

gde su: x, y promenljive, «equivalentobjectproperties» oznaka konstante iz ontologije, «RDF» je ontološki predikat, a «OntoEqvObjprop» oznaka predikata. Prolog forma pravila R 5.22 formirana je transformacijom prema (3.7), (3.8), (3.11), (3.12), (3.13) i (3.14):

```
ontoeqvobjprop(X,Y):-rdf(X,equivalentobjectproperties,Y).  (PR 5.22)
```

Pravilo R 5.23 –Izdvajanje svih ontoloških opsega osobina objekata sa ograničenjima koja odgovaraju kardinalitetu poveznika uspostavljenih između entiteta u modelu podataka za koje postoje odgovarajuće ontološke klase:

$$\begin{aligned}&(\forall xc)(\forall yop)(\forall zcd_1)(\forall zcd_2)(\exists xe_1)(\exists xe_2)(\exists yr)\ (\text{RDF}(yop,\text{type},\text{objectproperty}) \wedge \\ &\quad \text{RDF}(xc,yop,zcd_1) \wedge \text{RDF}(xc,yop,zcd_2) \wedge \text{Ent}(xe_1) \wedge \text{Ent}(xe_2) \wedge \text{Rel}(yr) \wedge \\ &\quad \text{P}(xe_1,yr) \wedge \text{P}(yr,xe_2) \wedge ((\text{P}(zcd_1,yr) \wedge \text{P}(zcd_2,yr)) \vee (\text{P}(yr,zcd_1) \wedge \text{P}(yr,zcd_2))) \wedge \\ &\quad (xe_1=xc \vee xe_2=xc) \wedge yr=yop \wedge \neg xcd_1=xcd_2 \Rightarrow \text{OntoCard}(xc,yop,xcd1,xcd_2)) \qquad \text{(R 5.23)}\end{aligned}$$

gde su: xc, yop, zcd_1, zcd_2, xe_1, xe_2, yr promenljive, «type» i «objectproperty» oznake ontoloških konstanti, «RDF» je ontološki predikat, «Ent», «Rel» i «P» oznake predikata iz modela podataka, «OntoCard» oznaka predikata. Prolog forma pravila R 5.23 formirana je transformacijom, na osnovu (3.7), (3.8), (3.11), (3.12), (3.13) i (3.14):

```
ontocard(XC,YOP,ZCD1,ZCD2):-rdf(YOP,type,objectproperty),
      rdf(XC,YOP,ZCD1),rdf(XC,YOP,ZCD2),ent(XE1),ent(XE2),
      rel(YR),p(XE1,YR),p(YR,XE2),((p(ZCD1,YR),p(ZCD2,YR));
      (p(YR,ZCD1),p(YR,ZCD2))),(XE1=XC;XE2=XC),YR=YOP,not
      ZCD1=ZCD2.                                           (PR 5.23)
```

Pravilo R 5.24 – Izdvajanje svih ontoloških opsega osobina objekata sa ograničenjima koja odgovaraju kardinalitetu sinonima poveznika uspostavljenih između entiteta u modelu podataka za koje postoje odgovarajuće ontološke klase:

$$(\forall xc)(\forall yop)(\forall zcd_1)(\forall zcd_2)(\exists xe_1)(\exists xe_2)(\exists xc_1)(\exists xc_2)(\exists yr)\ (RDF(yop,type,objectproperty) \wedge RDF(xc,yop,zcd_1) \wedge RDF(xc,yop,zcd_2) \wedge Ent(xe_1) \wedge Ent(xe_2) \wedge Rel(yr) \wedge P(xe_1,yr) \wedge P(xr,xe_2) \wedge ((P(zcd_1,yr) \wedge P(zcd_2,yr)) \vee (P(yr,zcd_1) \wedge P(yr,zcd_2))) \wedge (xe_1=xc \vee xe_2=xc) \wedge OntoRelSinRel(xc_1,yop,xc_2,xe_1,yr,xe_2) \wedge \neg zcd_1=zcd_2 \wedge \neg OntoCard(xc,yop,zcd_1,zcd_2) \Rightarrow OntoCardSin(xc,yop,zcd_1,zcd_2)) \quad (R\ 5.24)$$

gde su: xc, yop, zcd1, zcd2, xe1, xe2, yr promenljive, «type» i «objectproperty» oznake ontoloških konstanti, «RDF» je ontološki predikat, «Ent», «Rel» i «P» oznake predikata predikata iz modela podataka, a «OntoCard» i «OntoCardSin» su oznake predikata. Prolog forma pravila R 5.24 formirana je transformacijom, prema (3.7), (3.8), (3.11), (3.12), (3.13) i (3.14):

```
ontocardsin(XC,YOP,ZCD1,ZCD2):-rdf(YOP,type,objectproperty),
      rdf(XC,YOP,ZCD1),rdf(XC,YOP,ZCD2),ent(XE1),ent(XE2),rel(YR),
      p(XE1,YR),p(YR,XE2),((p(ZCD1,YR),p(ZCD2,YR));(p(YR,ZCD1),
      p(YR,ZCD2))),(XC=XE1;XC=XE2),ontorelsinrel(XC1,YOP,XC2,XE1,Y
      YR,XE2),not ZCD1=ZCD2,not ontocard(XC,YOP,ZCD1,ZCD2).(PR 5.24)
```

Tabela 5.2 prikazuje prethodno definisana pravila zaključivanja za semantičku verifikaciju modela podataka grupisane prema vrstama elemenata ER modela čija se semantika proverava.

Tabela 5.2: Pregled pravila zaključivanja za semantičku verifikaciju prema vrstama elemenata

Vrste elemenata u konceptualnom modelu podataka	**Oznaka pravila zaključivanja**	**Osobine modela za proveru**
eniteti	R 5.1, R 5.2, R 5.9, R 5.21	usklađenost sa ontološkim klasama, pripadnost atributa entitetima
atributi	R 5.3, R 5.4, R 5.7, R 5.8, R 5.9	usklađenost sa ontološkim osobinama podataka, provera tipova podataka, pripadnost atributa entitetima
poveznici	R 5.5, R 5.6, R 5.10, R 5.11, R 5.12, R 5.13, R 5.14, R 5.15, R 5.22, R 5.23, R 5.24	usklađenost sa ontološkim osobinama objekata kao instancama klasa, povezanost sa entitetima koji odgovaraju ontološkim klasama, sinonimi za poveznike i entitete, kardinaliteti poveznika koji odgovaraju opsezima ontoloških osobina objekata
mešoviti objekti-veze	R 5.15, R 5.16	usklađenost sa ontološkim osobinama objekata kao instancama klasa, sinonimi za gerunde
is_a hijerarhija	R 5.17, R 5.18, R 5.19, R 5.20	usklađenost sa ontološkim nadklasama i podklasama, sinonimi za superklasu i podklase

Spisak pravila zaključivanja za proveru semantičke korekntosti ER modela podataka sastoji se iz sledećeg skupa Prolog pravila zaključivanja (Listing 5.2):

Listing 5.2:

```
ontoclassent(X):-rdf(X,type,class),ent(X).

ontoclassnoent(X):-rdf(X,type,class),not ent(X).

ontodataatrib(X):-rdf(X,type,dataproperty),atr(X).

ontodatanoatrib(X):-rdf(X,type,dataproperty),not atr(X).

ontoobjproprel(X):-rdf(X,type,objectproperty),rel(X).

ontoobjpropnorel(X):-rdf(X,type,objectproperty),not rel(X).

ontodataatribtype(X,Y):-
rdf(X,type,dataproperty),rdf(X,range,Y),atr(X),res(Z),p(X,Z),datatype(Y,Z).

ontodataatribtypesin(X,Y,X1,Y1):-
rdf(X,type,dataproperty),rdf(X,range,Y),atr(X1),res(Y1),p(X1,Y1),
datatype(Y,Y1),not X=X1.

ontoclassentdataatrib(X,Y):-
rdf(X,type,class),rdf(Y,type,dataproperty),rdf(Z,type,namedindividual),
rdf(Z,classassertion,X),rdf(Y,datapropertyassertion,Z),ent(X),atr(Y),p(X,Y).

ontorel(XC1,YOP,XC2,XE1,YR,XE2):-rdf(XC1,type,class),
rdf(XO1,classassertion,XC1),rdf(XO1,type,namedindividual),
rdf(XC2,type,class),rdf(XO2,classassertion,XC2),
rdf(XO2,type,namedindividual),rdf(YOP,type,objectproperty),
rdf(XO1,YOP,XO2),ent(XE1),ent(XE2),rel(YR),p(XE1,YR),
p(YR,XE2),XE1=XC1,XE2=XC2,YR=YOP.

ontorelsinrel(XC1,YOP,XC2,XE1,YR,XE2):- rdf(XC1,type,class),
rdf(XO1,classassertion,XC1),rdf(XO1,type,namedindividual),
rdf(XC2,type,class),rdf(XO2,classassertion,XC2),
rdf(XO2,type,namedindividual),rdf(YOP,type,objectproperty),
rdf(XO1,YOP,XO2),ent(XE1),ent(XE2),rel(YR),
p(XE1,YR),p(YR,XE2),XE1=YC1,XE2=XC2,
not ontorel(XC1,YP,XC2,XE1,YR,XE2).

ontorelsinent(XC1,YOP,XC2,XE1,YR,XE2):- rdf(XC1,type,class),
rdf(XO1,classassertion,XC1),rdf(XO1,type,namedindividual),
rdf(XC2,type,class),rdf(XO2,classassertion,XC2),
rdf(XO2,type,namedindividual),rdf(YOP,type,objectproperty),
rdf(XO1,YOP,XO2),ent(XE1),ent(XE2),rel(YR),p(XE1,YR),
p(YR,XE2),(XE1=XC1;XE2=XC2),YR=YOP,not ontorel(XC1,YOP,XC2,XE1,YR,XE2),
not ontorelsinrel(XC1,YOP,XC2,XE1,YR,XE2).

ontorelsinent2(XC1,YOP,XC2,XE1,YR,XE2):- rdf(XC1,type,class),
rdf(XO1,classassertion,XC1),rdf(XO1,type,namedindividual),
rdf(XC2,type,class),rdf(XO2,classassertion,XC2),
rdf(XO2,type,namedindividual),rdf(YP,type,objectproperty),
rdf(XO1,YOP,XO2),ent(XE1),ent(XE2),rel(YR),p(XE1,YR),
p(YR,XE2),(XE1=XC1;XE2=XC2),YR=YOP, not ontorel(XC1,YOP,XC2,XE1,YR,XE2),
not ontorelsinrel(XC1,YP,XC2,XE1,YR,XE2),
not ontorelsinent(XC1,YOP,XC2,XE1,YR,XE2).
```

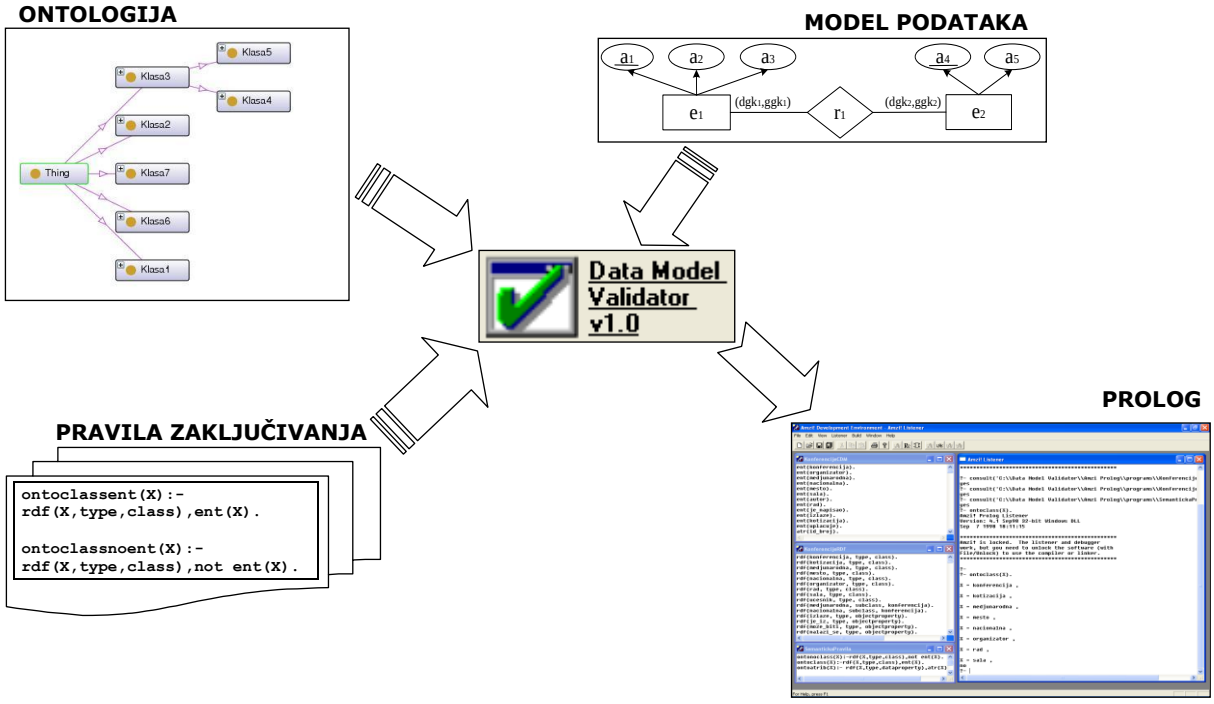
Listing 5.2 (nastavak):

```
ontorelsinentrel(XC1,YOP,XC2,XE1,YR,XE2):- rdf(XC1,type,class),
rdf(XO1,classassertion,XC1),rdf(XO1,type,namedindividual),
rdf(XC2,type,class),rdf(XO2,classassertion,XC2),
rdf(XO2,type,namedindividual),rdf(YOP,type,objectproperty),
rdf(XO1,YOP,XO2),ent(XE1),ent(XE2),rel(YR),p(XE1,YR),
p(YR,XE2),(XE1=XC1;XE2=XC2),not ontorel(XC1,YOP,XC2,XE1,YR,XE2),
not ontorelsinrel(XC1,YOP,XC2,XE1,YR,XE2),
not ontorelsinent(XC1,YOP,XC2,XE1,YR,XE2),
not ontorelsinent2(XC1,YOP,XC2,XE1,YR,XE2).

entwithid(X,Y):-ent(X),atr(Y),res(idatr),p(X,Y),p(Y,idatr).

ontoobjpropger(XC1,YOP,XC2,XE1,YER,XE2):-
rdf(YOP,type,objectproperty),rdf(XC1,type,class),
rdf(XO1,classassertion,XC1),rdf(XC2,type,class),
rdf(XO2,classassertion,XC2),rdf(XO2,type,namedindividual),
rdf(XO1,YOP,XO2),ent(XE1),ent(YER),ent(XE2),rel(YR1),
rel(YR2),p(XE1,YR1),p(YR1,YER),p(YER,YR2),p(YR2,XE2), YOP=YER,not
entwithid(YER,YID).

ontoobjpropgersin(XC1,YOP,XC2,XE1,YER,XE2):-
rdf(YOP,type,objectproperty),rdf(XC1,type,class),
rdf(XO1,classassertion,XC1),rdf(XC2,type,class),
rdf(XO2,classassertion,XC2),rdf(XO1,type,namedindividual),
rdf(XO2,type,namedindividual),rdf(XO1,YOP,XO2),ent(XE1),
ent(YER),ent(XE2),rel(YR1),rel(YR2),p(XE1,YR1),p(XR1,XER),
p(YER,YR2),p(YR2,XE2),(XE1=XC1;XE2=XC1),not entwithid(YER,YID),
not ontoobjpropger(XC1,YOP,XC2,XE1,YER,XE2).

ontosubclassisa(X,X1,X2):-
rdf(X,type,class),rdf(X1,subclass,X),rdf(X2,subclass,X),ent(X),ent(X1),
ent(X2),p(X,Y),p(Y,X1),p(Y,X2),not X1=X2.

ontosubclassisasin(XP,XC1,XC2,XE,XE1,XE2):-
rdf(XP,type,class),rdf(XC1,subclass,XP),
rdf(XC2,subclass,XP),ent(XE),ent(XE1),ent(XE2),p(XE,Y), p(Y,XE1),p(Y,XE2),
not XC1=XC2,not XE1=XE2,not ontosubclassisa(XP,XC1,XC2),
(not XE=XC1;not XE=XC2).

ontosubclassisasinsub(X,X1,X2):-
rdf(X1,subclass,Z),rdf(X2,subclass,Z),ent(X1),ent(X2),ent(X),p(X,Y),p(Y,X1),
p(Y,X2),not X1=X2.

ontosubclassnoisa(X,X1,X2):-
rdf(X,type,class),rdf(X1,subclass,X),rdf(X2,subclass,X),not X1=X2,
not ontosubclassisa(X,X1,X2).

ontoclassentsin(X,Y):-ontoclassnoent(X),ent(Y),not ontoclassent(Y).

ontoeqvobjprop(X,Y):-rdf(X,equivalentobjectproperties,Y).

ontocard(XC,YOP,ZCD1,ZCD2):-rdf(YOP,type,objectproperty),
rdf(XC,YOP,ZCD1),rdf(XC,YOP,ZCD2),ent(XE1),ent(XE2),
rel(YR),p(XE1,YR),p(YR,XE2),((p(ZCD1,YR),p(ZCD2,YR));
(p(YR,ZCD1),p(YR,ZCD2))),(XE1=XC;XE2=XC),YR=YOP,not ZCD1=ZCD2.

ontocardsin(XC,YOP,ZCD1,ZCD2):-rdf(YOP,type,objectproperty),
rdf(XC,YOP,ZCD1),rdf(XC,YOP,ZCD2),ent(XE1),ent(XE2),rel(YR),p(XE1,YR),
p(YR,XE2),((p(ZCD1,YR),p(ZCD2,YR));(p(YR,ZCD1),
p(YR,ZCD2))),(XC=XE1;XC=XE2),ontorelsinrel(XC1,YOP,XC2,XE1,YYR,XE2),not
ZCD1=ZCD2,not ontocard(XC,YOP,ZCD1,ZCD2).
```

5.5. INTEGRACIJA MAPIRANE ONTOLOGIJE, FORMALIZOVANOG MODELA PODATAKA I PRAVILA ZAKLJUČIVANJA

Mapirana ontologija i formalna specifikacija modela podataka su prikazanim i opisanim modelom predstavljeni u logičkom formalnom jeziku, tj. u rečenicama predikatskog računa prvog reda. Zatim su transformacijama prevedene u oblik klauzula, koji je pogodan za procesiranje u izabranom sistemu automatskog rezonovanja, tj. u Prologu.

Definisana pravila rezonovanja, iz odeljka 5.4., se primenjuju nad semantikom sistema prezentovanom ontologijom i modelom podataka koji treba da bude vrednovan. Ova pravila se koriste za proveru semantičke komponente kvaliteta modela podataka, tako što se odgovori sistema automatskog rezonovanja uključuju u odgovarajuću metriku.



Slika 5.14 – Šema integracije komponenti modela

Mapiranje OWL/RDF ontologije, formalnu specifikaciju modela podataka i njihovu integraciju sa pravilima zaključivanja vrši program pod nazivom «Data Model Validator» (DMV) koji je nastao kao rezultat istraživanja u okviru disertacije.

Nakon ove integracije formira se jedinstvena datotetka sa svim elementima modela koja predstavlja ulaz u Prolog sistem. U okviru Prolog editora se vrši postavljanje upita (ciljeva, pitanja prikazanih u narednom odeljku 5.6.), na koje se dobijaju odgovori sistema.

5.6. UPITI ZA PROVERU KOREKTNOSTI MODELA

1. (Testiranje pravila R 5.1) Koje klase ontologije jesu pokrivene tipovima entiteta u konceptualnom modelu podataka ?

```
?-ontoclassent(X).
```

2. (Testiranje pravila R 5.2) Koje klase ontologije nisu pokrivene tipovima entiteta u konceptualnom modelu podataka?

```
?-ontoclassnoent(X).
```

3. (Testiranje pravila R 5.3) Koje osobine klasa ontologije jesu pokrivene atributima u konceptualnom modelu podataka?

```
?-ontodataatrib(X).
```

4. (Testiranje pravila R 5.4) Koje osobine klasa ontologije nisu pokrivene atributima u konceptualnom modelu podataka?

```
?-ontodatanoatrib(X).
```

5. (Testiranje pravila R 5.5) Koje osobine objekata klasa ontologije jesu pokrivene relacijama u konceptualnom modelu podataka?

```
?-ontoobjproprel(X).
```

6. (Testiranje pravila R 5.6) Koje osobine objekata klasa ontologije nisu pokrivene relacijama u konceptualnom modelu podataka?

```
?-ontoobjpropnorel(X).
```

7. (Testiranje pravila R 5.7) Za koje ontološke osobine podataka postoje odgovarajući atributi u konceptualnom modelu podataka takvi da su usklađeni i njihovi tipovi podataka?

```
?-ontodataatribtype(X,Y).
```

8. (Testiranje pravila R 5.8) Za koje ontološke osobine podataka postoje atributi u konceptualnom modelu podataka takvi da su im nazivi različiti (sinonimi) ali su usklađeni tipovi podataka?

```
?-ontodataatribtypesin(X,Y,X1,Y1).
```

9. (Testiranje pravila R 5.9) Za koje skupove ontoloških osobina podataka objekata klase postoje definisani odgovarajući atributima u okviru entiteta?

```
?-ontoclassentdataatrib(X,Y).
```

10. (Testiranje pravila R 5.10) Koje ontološke osobine objekata klasa odgovaraju poveznicima, tj. relacijama između tipova entiteta pokrivenih ontološkim klasama?

```
?-ontorel(XC1,YOP,XC2,XE1,YR,XE2).
```

11. (Testiranje pravila R 5.11) Koje ontološke osobine objekata klasa se, po nazivu, razlikuju od naziva poveznika/relacija uspostavljenih između tipova entiteta pokrivenih ontološkim klasama (sinonimi za relacije)?

```
?-ontorelsinrel(XC1,YOP,XC2,XE1,YR,XE2).
```

12. (Testiranje pravila R 5.12) Koje ontološke osobine objekata klasa odgovaraju poveznicima, tj. relacijama uspostavljenim između entiteta koji nisu pokriveni odgovaraju ontološkim klasama po nazivu (sinonimi za jedan entitet)?

```
?-ontorelsinent(XC1,YOP,XC2,XE1,YR,XE2).
```

13. (Testiranje pravila R 5.13) Koje ontološke osobine objekata klasa odgovaraju poveznicima/relacijama uspostavljenim između entiteta koji nisu pokriveni odgovaraju ontološkim klasama po nazivu (sinonimi za oba entiteta)?

```
?-ontorelsinent2(XC1,YOP,XC2,XE1,YR,XE2).
```

14. (Testiranje pravila R 5.14) Koje ontološke osobine objekata klasa se po nazivu razlikuju od relacija uspostavljenih između entiteta, od kojih je samo jedan entitet pokriven odgovarajućom ontološkom klasom po nazivu, dok se drugi može razlikovati (sinonimi za relaciju i jedan od entiteta)?

```
?-ontorelsinentrel(XC1,YOP,XC2,XE1,YR,XE2).
```

15. (Testiranje pravila R 5.15) Da li postoje ontološke osobine objekata klasa koji odgovaraju mešovitim objektima-veze, tj. gerundima u modelu podataka:

```
?-ontoobjpropger(XC1,YOP,XC2,XE1,YER,XE2).
```

16. (Testiranje pravila R 5.16) Da li postoje ontološke osobine objekata klase koji po nazivu ne odgovaraju mešovitim objektima veze, tj. gerundima u modelu podataka, ali su uspostavljene između entiteta za koje su definsane ontološke klase (sinonimi za gerunde):

```
?-ontoobjpropgersin(XC1,YOP,XC2,XE1,YER,XE2).
```

17. (Testiranje pravila R 5.17) Za koje ontološke nadklase i podklase postoji definisana odgovarajuća IS_A hijerarhija između entiteta u konceptualnom modelu podataka ?

```
?-ontosubclassisa(X,X1,X2).
```

18. (Testiranje pravila R 5.18) Za koje ontološke nadklase i podklase postoji IS_A hijerarhija između entiteta u konceptualnom modelu podataka, čak i ukoliko nazivi nadklase ontologije neodgovaraju nazivima entiteta (sinonimi za nadklasu)?

```
?-ontosubclassisasin(YOP,XC1,XC2,XE,XE1,XE2).
```

19. (Testiranje pravila R 5.19) Za koje ontološke nadklase i podklase postoji IS_A hijerarhija između entiteta u konceptualnom modelu podataka, čak i ukoliko nazivi nadklase ontologije neodgovaraju nazivima entiteta (sinonimi za podklase)?

```
?-ontosubclassisasinsub(X,X1,X2).
```

20. (Testiranje pravila R 5.20) Za koje ontološke nadklase i podklase ne postoje definisane odgovarajuće IS_A hijerarhije između entiteta u konceptualnom modelu podataka ?

```
?-ontosubclassnoisa(X,X1,X2).
```

21. (Testiranje pravila R 5.21) Da li postoje ontološke klase koje nisu pokrivene tipovima entiteta sa istim nazivom ali postoje tipovi entiteta sa sličnim nazivom pa se mogu odrediti sinonimi?

```
?-ontoclassentsin(X,Y).
```

22. (Testiranje pravila R 5.22) Koliko ima parova ekvivalentnih ontoloških osobina objekata?

```
?-ontoeqvobjprop(X,Y).
```

23. (Testiranje pravila R 5.23) Koji opsezi ontoloških osobina objekata klasa odgovaraju kardinalitetima poveznika, tj. relacija između tipova entiteta pokrivenih ontološkim klasama?

```
?-ontocard(XC,YOP,ZCD1,ZCD2).
```

24. (Testiranje pravila R 5.24) Koji opsezi ontoloških osobina objekata klasa odgovaraju kardinalitetima sinonima poveznika između tipova entiteta pokrivenih ontološkim klasama?

```
?-ontocardsin(XC,YOP,ZCD1,ZCD2).
```

5.7. VREDNOVANJE POJEDINAČNIH MODELA PODATAKA

Za vrednovanje pojedinačnih radova, tj. konceptualnih modela podataka je definisana metrika po ugledu na istraživanja [Gray et al., 1991], [Moody 1998] , [Piattini, Polo et al., 2000] i [Kesh 1995] . Ukupna ocena semantičke usklađenosti sa ontologijom je prikazana u formuli 5.1:

$$OM = \frac{K_E \cdot OM_E + K_A \cdot OM_A + K_R \cdot OM_R + K_{SC} \cdot OM_{SC}}{4} \tag{5.1}$$

gde je:

- OM – ukupna ocena semantičke usklađenosti modela podataka sa ontologijom,
- OM_E – ontološka ocena za entitete,
- OM_A – ontološka ocena za atribute,
- OM_R – ontološka ocena za poveznike i gerunde,
- OM_{SC} – ontološka ocena za nadklase i podklase,
- K_E, K_A, K_R, K_{SC} - težinski koeficijenti pojedinih ocena.

Težinski koeficijenti su definisani prema ontološkim metrikama navedenim u (4.4) i u predloženoj metrici modela iznose $K_E=K_A=K_R=K_{SC}=1$. U prethodno kreiranoj formuli (5.1) se, u imeniocu razlomka, nalazi broj četiri zbog toga što postoji isto toliko elemenata modela podataka koji se sematnički vrednuju, a sve vrednosti OM_E, OM_A, OM_R i OM_{SC} predstavljaju, svaka pojedinačno, procentualnu usaglašenost elementa modela podataka sa ontologijom. Krajnji cilj je bila aritmetička sredina ove četiri semantičke ocene. Zbog prethodno navedenog procentualnog iskazivanja vrednosti ocena, se u formulama (5.2), (5.3), (5.4) i (5.5), u brojiocu razlomaka vrši množenje zbira dobijenih broja elemenata iz Prolog upita sa brojem 100.

Ontološka ocena za entitete - $\boldsymbol{OM_E}$ se izračunava na sledeći način:

$$OM_E = \frac{(\sum_{n_E}^{i_E=1} E(R5.1)_{i_E} + \sum_{n_E}^{i_E=1} E(R5.21)_{i_E}) \cdot 100}{\sum_{n_E}^{i_E=1} E(R5.1)_{i_E} + \sum_{n_E}^{i_E=1} E(R5.2)_{i_E}} \tag{5.2}$$

gde su:

- $E(R\ 5.1)$, $E(R\ 5.2)$ i $E(R\ 5.21)$ vrednosti dobijene testiranjem pravila zaključivanja R 5.1, R 5.2 i R 5.21.
- Zbir $E(R\ 5.1)$ i $E(R\ 5.21)$ predstavlja ukupan broj ontoloških klasa pokrivenih entitetima u modelu podataka.
- Zbir $E(R\ 5.1)$ i $E(R\ 5.2)$ predstavlja ukupan broj klasa u ontologiji.

Ontološka ocena za atribute - OM_A se izračunava na sledeći način:

$$OM_A = \frac{((\sum_{n_A}^{i_A=1} A(R5.3)_{i_A} + (\sum_{n_A}^{i_A=1} A(R5.7)_{i_A} + \sum_{n_A}^{i_A=1} A(R5.8)_{i_A}) + \sum_{n_A}^{i_A=1} A(R5.9)_{i_A})/3) \cdot 100}{\sum_{n_A}^{i_A=1} A(R5.3)_{i_A} + \sum_{n_A}^{i_A=1} A(R5.4)_{i_A}} \quad (5.3)$$

gde su:

- $A(R\ 5.3)$, $A(R\ 5.4)$, $A(R\ 5.7)$, $A(R\ 5.8)$ i $A(R\ 5.9)$ vrednosti dobijene testiranjem pravila zaključivanja R 5.3, R 5.4, R 5.7, R 5.7 i R 5.9.
- Zbir $A(R\ 5.3)$, $A(R\ 5.7)$, $A(R\ 5.8)$ i $A(R\ 5.9)$ predstavlja ukupan broj ontoloških osobina podataka pokrivenih atributima u modelu podataka.
- Zbir $A(R\ 5.3)$ i $A(R\ 5.4)$ predstavlja ukupan broj osobina podataka u ontologiji.

Ontološka ocena za poveznike i gerunde - OM_R se izračunava na sledeći način:

$$OM_R = (\frac{\begin{array}{l}(\sum_{n_R}^{i_R=1} R(R5.10)_{i_R} + (\sum_{n_R}^{i_R=1} R(R5.11)_{i_R} + \sum_{n_R}^{i_R=1} R(R5.12)_{i_R} + \sum_{n_R}^{i_R=1} R(R5.13)_{i_R} \\ + \sum_{n_R}^{i_R=1} R(R5.14)_{i_R}) + (\sum_{n_R}^{i_R=1} R(R5.15)_{i_R} + \sum_{n_R}^{i_R=1} R(R5.16)_{i_R}))\cdot 100\end{array}}{\sum_{n_R}^{i_R=1} R(R5.5)_{i_R} + \sum_{n_R}^{i_R=1} R(R5.6)_{i_R} - \sum_{n_R}^{i_R=1} R(R5.22)_{i_R}} + \frac{(\sum_{n_R}^{i_R=1} R(R5.23)_{i_R} + \sum_{n_R}^{i_R=1} R(R5.24)_{i_R})\cdot 100}{(\sum_{n_R}^{i_R=1} R(R5.5)_{i_R} + \sum_{n_R}^{i_R=1} R(R5.6)_{i_R} - \sum_{n_R}^{i_R=1} R(R5.22)_{i_R})*2})/2 \quad (5.4)$$

gde su:

- $R(R\ 5.5)$, $R(R\ 5.6)$, $R(R\ 5.10)$, $R(R\ 5.11)$, $R(R\ 5.12)$, $R(R\ 5.13)$, $R(R\ 5.14)$, $R(R\ 5.15)$ i $R(R\ 5.16)$ vrednosti dobijene testiranjem pravila zaključivanja R 5.5, R 5.6, R 5.10, R 5.11, R 5.12, R 5.13, R 5.14, R 5.15 i R 5.16.
- Zbir $R(R\ 5.10)$, $R(R\ 5.11)$, $R(R\ 5.12)$, $R(R\ 5.13)$, $R(R\ 5.14)$, $R(R\ 5.15)$ i $R(R\ 5.16)$ predstavlja ukupan broj ontoloških osobina objekata pokrivenih relacijama i gerundima u modelu podataka.
- Zbir $R(R\ 5.23)$ i $R(R\ 5.24)$ predstavlja ukupan broj ograničenja ontoloških osobina objekata pokrivenih kardinalitetima tipa poveznika u modelu podataka.
- $R(R\ 5.5)+R(R\ 5.6)-R(R5.22)$ predstavlja ukupan broj osobina objekata u ontologiji.

Ontološka ocena za nadklase i podklase - OM_{SC} se izračunava na sledeći način:

$$OM_{SC} = \left(\frac{\left(\sum_{n_{SC}}^{i_{SC}=1} SC(R5.17)_{i_{sc}} + \sum_{n_{SC}}^{i_{SC}=1} SC(R5.18)_{i_{sc}}\right)\cdot 100}{\sum_{n_{SC}}^{i_{SC}=1} SC(R5.17)_{i_{sc}} + \sum_{n_{SC}}^{i_{SC}=1} SC(R5.20)_{i_{sc}}} + \frac{\left(\sum_{n_{SC}}^{i_{SC}=1} SC(R5.17)_{i_{sc}} + \sum_{n_{SC}}^{i_{SC}=1} SC(R5.19)_{i_{sc}}\right)\cdot 100}{\sum_{n_{SC}}^{i_{SC}=1} SC(R5.17)_{i_{sc}} + \sum_{n_{SC}}^{i_{SC}=1} SC(R5.20)_{i_{sc}}}\right)/2 \quad (5.5)$$

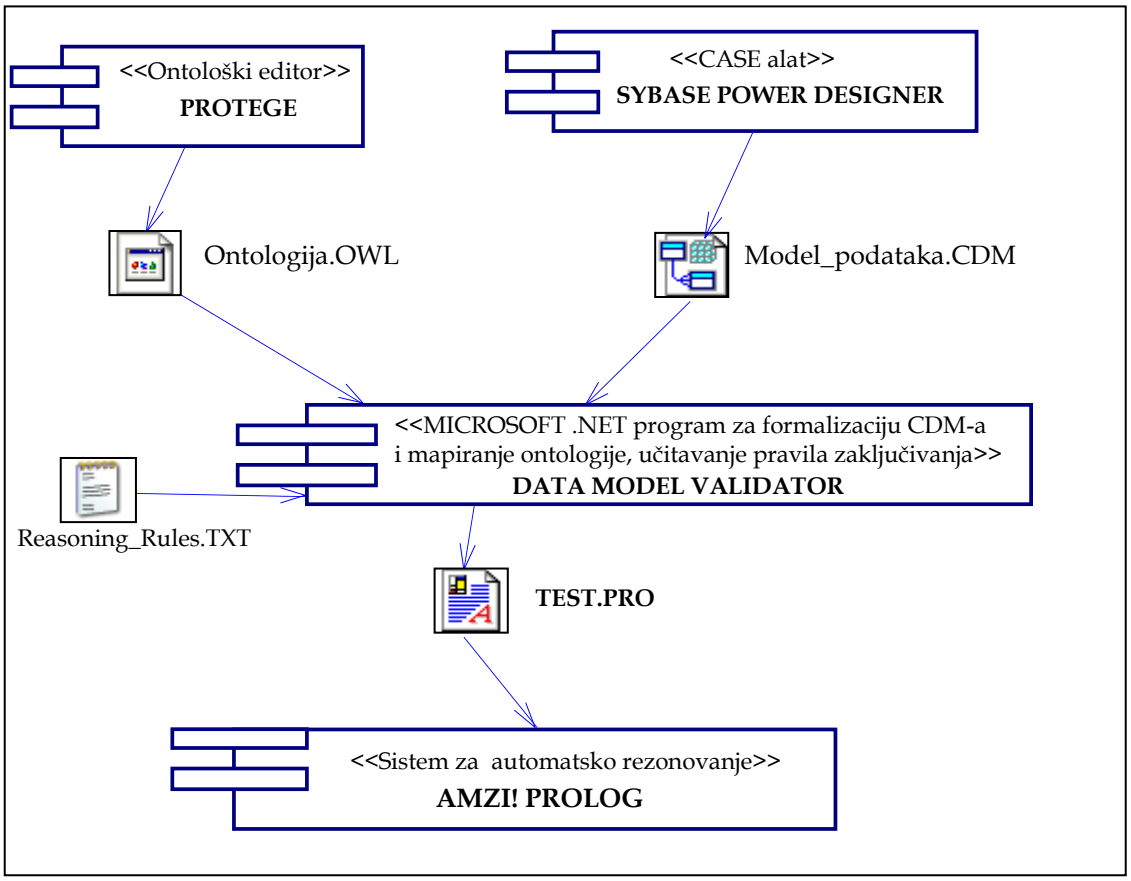
gde su:

- $SC(R\ 5.17)$, $SC(R\ 5.18)$, $SC(R\ 5.19)$ i $SC(R\ 5.20)$ vrednosti dobijene testiranjem pravila zaključivanja $R\ 5.17$, $R\ 5.18$, $R\ 5.19$ i $R\ 5.20$.
- Zbir $SC(R\ 5.17)$ i $SC(R\ 5.18)$ predstavlja ukupan broj ontoloških nadklasa pokrivenih supertipovima tipa entiteta u modelu podataka.
- Zbir $SC(R\ 5.17)$ i $SC(R\ 5.19)$ predstavlja ukupan broj ontoloških podklasa pokrivenih podtipovima tipa entiteta u modelu podataka.
- Zbir $SC(R\ 5.17)$ i $SC(R\ 5.20)$ predstavlja ukupan broj nadklasa u ontologiji.
- Zbir $SC(R\ 5.17)$ i $SC(R\ 5.20)$ predstavlja ukupan broj podklasa u ontologiji.

6. IMPLEMENTACIJA ONTOLOŠKI ZASNOVANE ANALIZE SEMANTIČKE KOREKTNOSTI MODELA PODATAKA PRIMENOM SISTEMA AUTOMATSKOG REZONOVANJA

6.1. OPIS KORIŠTENIH SOFTVERSKIH ALATA

U procesu semantičke evaluacije modela podataka, korišteni su različiti softverski alati za potrebe kreiranja domenske ontologije, modela podataka, formalizacije modela i ontologije. U šemi integracije komponenti modela prikazanoj na slici 6.1. centralno mesto zauzima DMV program, koji integriše softverske alate čime se automatizuje deo procesa. Zbog toga je unapređen kvalitet procesa semantičke provere i vrednovanja modela podataka. Slika 6.1. prikazuje UML dijagram komponenti, koji opisuje softverske alate koji se su uključeni u proces, izlazne rezultate primene ovih alata i način njihove integracije.



Slika 6.1. - Dijagram komponenti sistema upotrebljenih softverskih alata

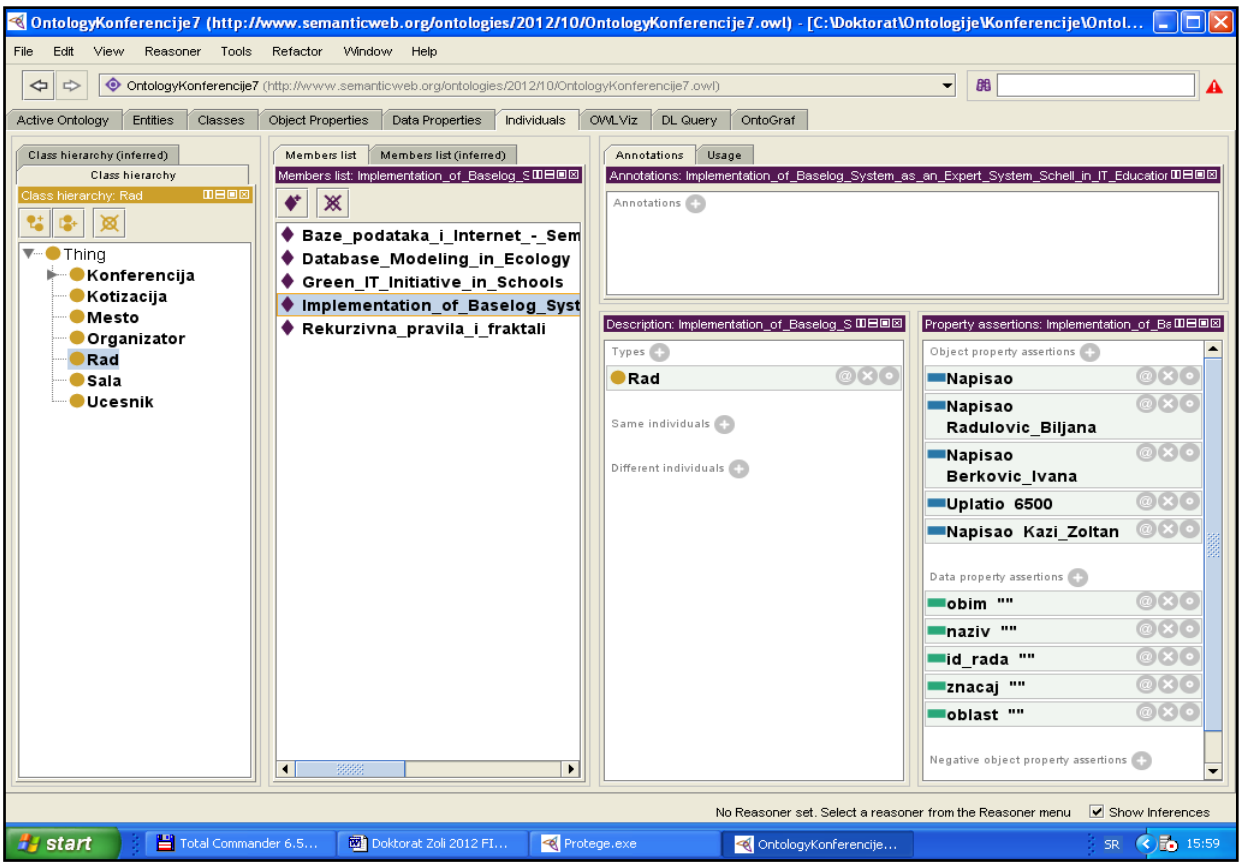
Osnovna tehnologija koja je korištena u DMV programu, u cilju integracije ontologije i modela podtaaka, je XML, kao format izlaznih datotetka koje ontološki i CASE alat kreiraju.

Predloženi sistem uključuje sledeće korištene alate:

- Protégé – ontološki alat. Daje rezultat modeliranja ontologija u formi izlaznog fajla sa ekstenzijom OWL i čija struktura je čitljiva kao i svaki drugi XML fajl, specifične sintakse OWL jezika.
- Sybase Power Designer – CASE alat korišten za modelovanje podataka. Daje rezultat u formi datoteke sa ekstenzijom CDM za konceptualni model podataka (EER model), ekstenzijom PDM za fizički model podataka (*Relational Data Model*) i ekstenzijom OOM za UML dijagrame, koji uključuje i dijagram klasa objektno orjentisanog modela. Svaki od navedenih fajlova su zapravo XML dokumenti koji sadrže opise elemenata odgovarajućeg modela podataka.
- AMZI PROLOG, SWI PROLOG – jezici logičkog programiranja koji se koriste kao alati za automatsko rezonovanje. Ulaz u PROLOG je fajl ima ekstenziju PRO. To je posebno formatirana tekstualna datoteka sa rečenicama koje zadovoljavaju sintaksu Prolog jezika.

6.1.1. Protégé ontološki editor

Protégé je *„open source"* ontološki editor koji se koristi i kao razvojni okvir sistema baziranih na znanju [1]. Razvijen je na Stanford univerzitetu, inicijalno kao softver u oblasti medicine i biohemijskih nauka i trenutno je jedan od vodećih ontoloških editora u svetu. Prošao je nekoliko modifikacija i verzija. Baziran je na programskom okruženju Java, omogućuje proširenja, što ga čini pogodnim i kao bazu za brz razvoj prototipa i aplikacija baziranih na znanju. Protégé ima značajnu podršku od strane akademske zajednice i vlade SAD, kao i poslovnih korisnika.



Slika 6.2. – Protégé ontološki alat (editor)

Protégé platforma podržava dva osnovna načina modelovanja ontologija:

- Protégé okviri (*Frames*) – implementira model baziran na znanju kompatibilan sa *Open Knowledge Base Connectivity protocol* (*OKBC*).
- Protégé OWL editor – integrisan sa Jena i ima „*open source*" API za razvoj komponenti korisničkog interfejsa i rad sa servisima semantičkog weba.

Protégé OWL editor omogućuje korisnicima da kreiraju, učitavaju i snimaju OWL i RDF ontologije, da vrše izmene i vizuelizaciju klasa i njihovih osobina, zatim da definišu logičke karakteristike klasa i OWL izraza, te pokreću i izvršavaju razne sisteme za rezonovanje. Omogućuje definisanje koncepata (klasa) ontologije, osobina, taksonomija i različitih ograničenja klasa, kao i njihovo instanciranje. Korisnički interfejs se sastoji od podešljivih formi organizovanih preko kartica. Grafičko prezentovanje ontologija se ostvaruje dodacima koji se posebno preuzimaju i instaliraju sa Interneta, kao što su: *ezOWL (Visual OWL) Editor, OWL-S editor, Jess, UML, XML Metadata, Interchange (XMI), DAML+OIL* i mnogi drugi. U izradi disertacije je korišten *GraphViz* editor za prikaz otologije i njenih komponenti.

Ontologije mogu biti izvezene u različite formate:

- RDF(S)/XML,
- OWL/XWL,
- OWL funkcionalna sintaksa,
- MANCHESTER OWL sintaksa,
- OBO 1.2 datoteku,
- KRSS sintaksa
- LATEX,
- TURTLE.

6.1.2. CASE alat Sybase Power Designer

Power Designer je CASE alat firme *Sybase* za računarski podržano softversko inženjerstvo, projektovanje i i mplementaciju baza podataka, programskih rešenja, XML modela, modela za IT projekte i sl. Nastao je početkom 90-tih godina prošlog veka i prošao je kroz više od 15 verzija.

Power Designer podržava kreiranje projekata iz sledećih oblasti:

- Modelovanje poslovnih procesa (*Business Process Model*, BPM): omogućuje analizu sistema od strane analitičara i menadžera u cilju racionalizacije i optimizacije novog sistema, kreiranje dijagrama toka podataka, generisanje koda za *Sybase WorkSpace Business Process*, *ebXML BPSS, ebXML CPA, BPEL4WS* ili *WS-BPEL*, reverzno inženjerstvo iz *ebXML BPSS, BPEL4WS ili WS-BPEL*, učitavanje *Process Analyst Model* (PAM) verzije 6.
- Modelovanje baza podataka: konceptualno modelovanje, logičko modelovanje baze podataka (*Conceptual and Logical Diagrams*), fizičko modelovanje podataka na relacionom nivou (*Physical Diagrams*), osmišljavanje okidača, kreiranje Web servisa, generisanje baze podataka, generisanje SQL koda, migraciju drugih

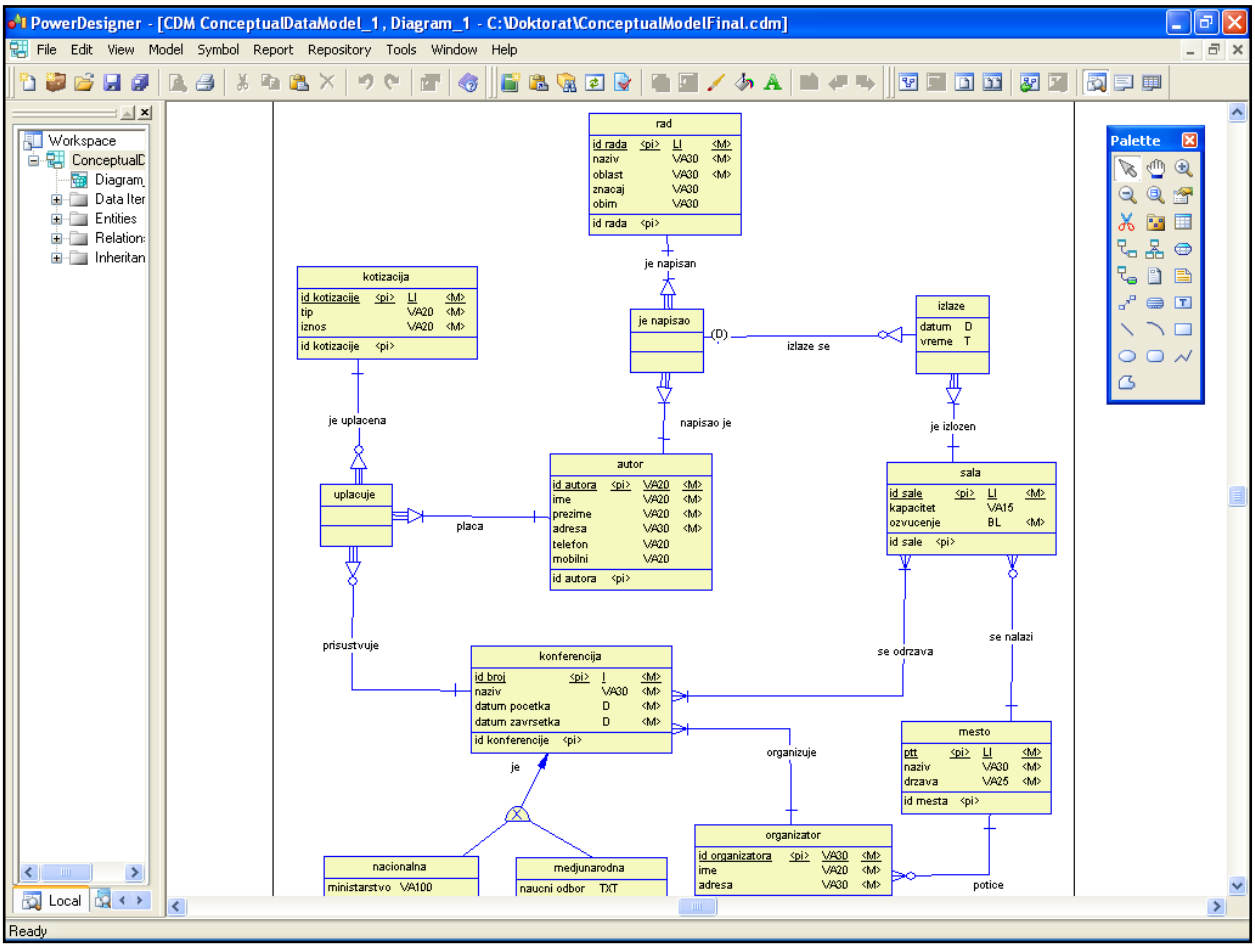
modela, poput *ERWin*-a u Power Designer, mapiranje konceptualnog u objektni model.

- Objektno orijentisano modelovanje (*Object Oriented Model* - OOM) pomoću *Unified Modeling Language* – UML, originalno osmišljenog od strane *Grady Boocha, Jamesa Rumbaugha* i *Ivara Jacobsona*, a trenutno kontrolisano od strane OMG (*Object Management Group*). Podržana je dobro definisana sintaksa i semantika u modelovanju. U tabeli 6.1. su prikazani dijagrami UML modela koji se mogu formirati u okviru objektno orijentisanog modelovanja. Podržava UML 2.0 standard za kreiranje dijagrama.

Tabela 6.1: UML dijagrami koje podržava Power Designer

Grupa	**Dijagram**
Use Case dijagrami	Use case dijagram
Dijagrami strukture	Class dijagram
	Composite structure dijagram
	Object dijagram
	Package dijagram
Dinamički dijagrami	Communication dijagram
	Sequence dijagram
	Statechart dijagram
	Activity dijagram
	Interaction overview dijagram
Dijagrami implementacije	Component dijagram
	Deployment dijagram

- XML modele za opis podataka i njihove strukture, koji može biti razmenjivan između nekompatibilnih sistema, rad sa XML *Schema Definition* datotekama (.XSD), a *Document Type Definition* datotekama (.DTD) ili *XML-Data Reduced* datotekama (.XDR) sa vizuelizacijom u formi dijagrama i šeme.
- *Free Model* (FEM) - bilo koja vrsta dijagrama u konteksno slobodnom okruženju, prikazivanje povezanosti dokumenata i relacija unutar organizacije, kreiranje dijagrama hijerarhije, korisnika budućeg rešenja, grupa i njihovih veza.
- Modelovanje zahteva klijenta i korisnika (*Requirements Model*),
- *Information Liquidity Model* (ILM) je zasnovan na kretanju podataka kroz organizaciju, omogućuje replikaciju podataka za *Replication Server* i *MobiLink*, transformaciju podataka iz različitih izvora i učitavanje u izlazni izvor kao što su ETL (*Extract Transform and Load*) i EII (*Enterprise Information Integration*).
- *Enterprise Architecture* modelovanje: analiza i dokumentovanje sistema i organizacije, usklađenost sa standardima i regulativama, modelovanje kooperacije. Dijagrami se mogu raditi za poslovni, aplikacioni i tehnološki sloj.



Slika 6.3. – Sybase Power Designer CASE alat

Prilikom kreiranja modela i baza podataka podržano je automatsko generisanje SQL koda za preko 50 softvera za upravljanje bazama podataka, Javu i Microsoft .Net. Softver ima podršku za inverzno inženjerstvo, dokumentovanje modela i projekata, izmenu i održavanje postojećih sistema. Osnovna jedinica rada je model koji mora sadržati najmanje jedan dijagram sa nekoliko objekata. Modeli se mogu organizovati u radne prostore (*Workspace*) i mogu biti podeljeni u veći broj paketa, zbog organizacionih razloga.

Takođe, značajna je podrška CASE alata u proveri ispravnosti različitih tipova modela sa aspekta sintakse i pravila za izgradnju dijagrama i modela.

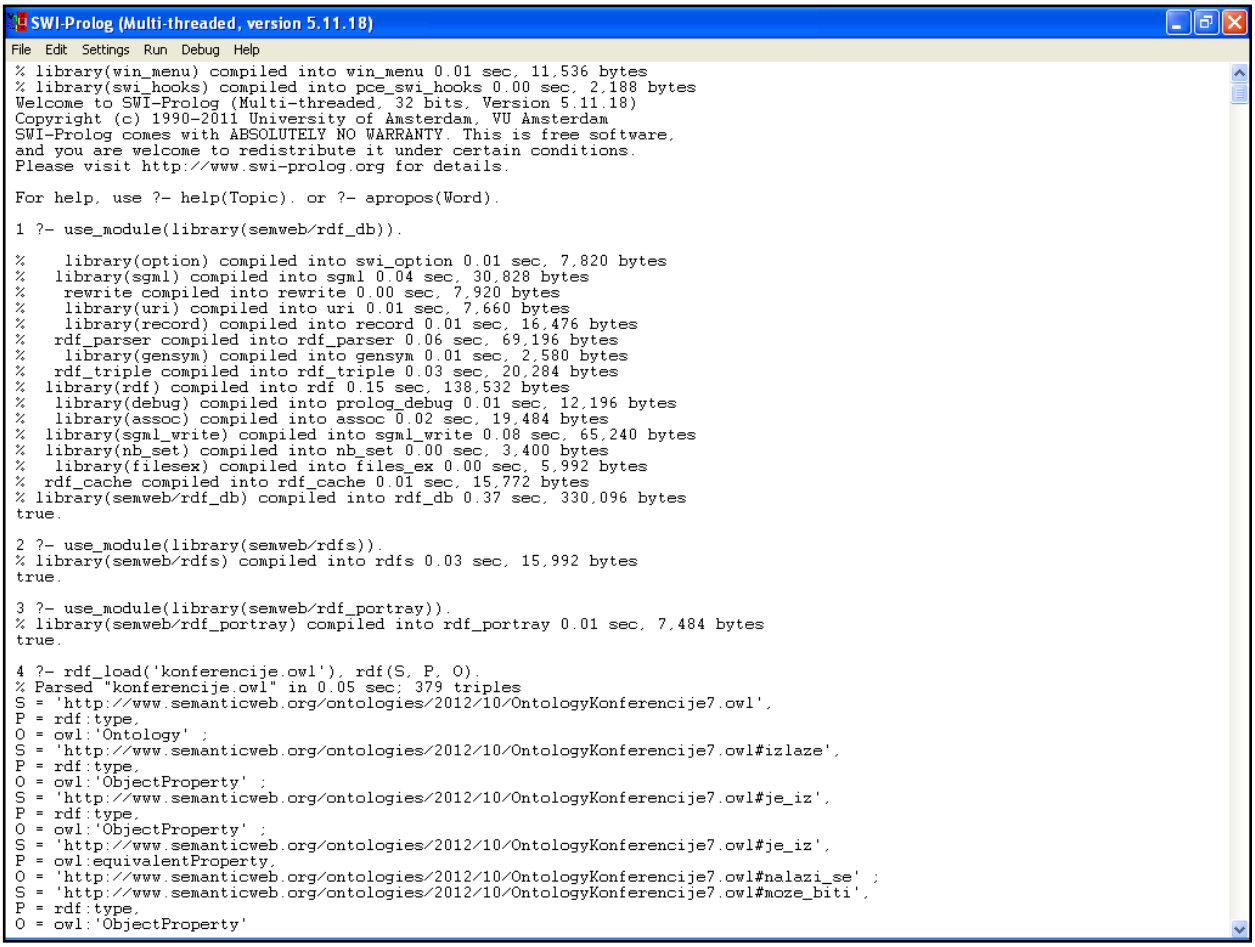
6.1.3. SWI Prolog

SWI Prolog je jezik logičkog programiranja koji spada u kategoriju besplatnih implementacija Prologa koji se distibuira slobodno preko Interneta [13]. Razvija se, od 1990. godine, na univerzitetu u Amsterdamu. Korisnici jezika su istraživači i studenti širom sveta, programeri kojima treba velik stepen prenosivosti i proširivosti aplikacija koje poseduju i karakteristike za rad u računarskim mrežama.

SWI Prolog poseduje RDF biblioteku i «*Reasoning*» modul koji su osnova infrastrukture za semantički Web, koja se koristi kada korisnici žele da rade sa ontologijama (Slika 6.3.), RDF i RDFS datotekama. Poseduje HTTP server biblioteku koja omogućuje da predikati budu raspoređeni na različitim HTTP lokacijama. Interesantan modul je još i «*Presentation*», koji služi za definisanje HTML pravila za kreiranje elemenata korisničkog interefejsa. [14]

Ostali alati i karakteristike SWI Prologa [15] su:

- Interaktivna izmena i učitavanje programa u toku izvršavanja.
- Automatsko učitavanje programa koje programer koristi.
- DWIM („*Do What I Mean*“) sistem za otkrivanje i ispravljanje grešaka.
- Editovanje programa iz komandne linije.
- Poseduje kompajler.
- Poseduje biblioteku «*Check*» za proveru kompletnosti učitanih Prolog programa.
- Ima sistem za pomoć.
- Ima grafičke alate, poput Editor programa, Prolog navigatora itd.



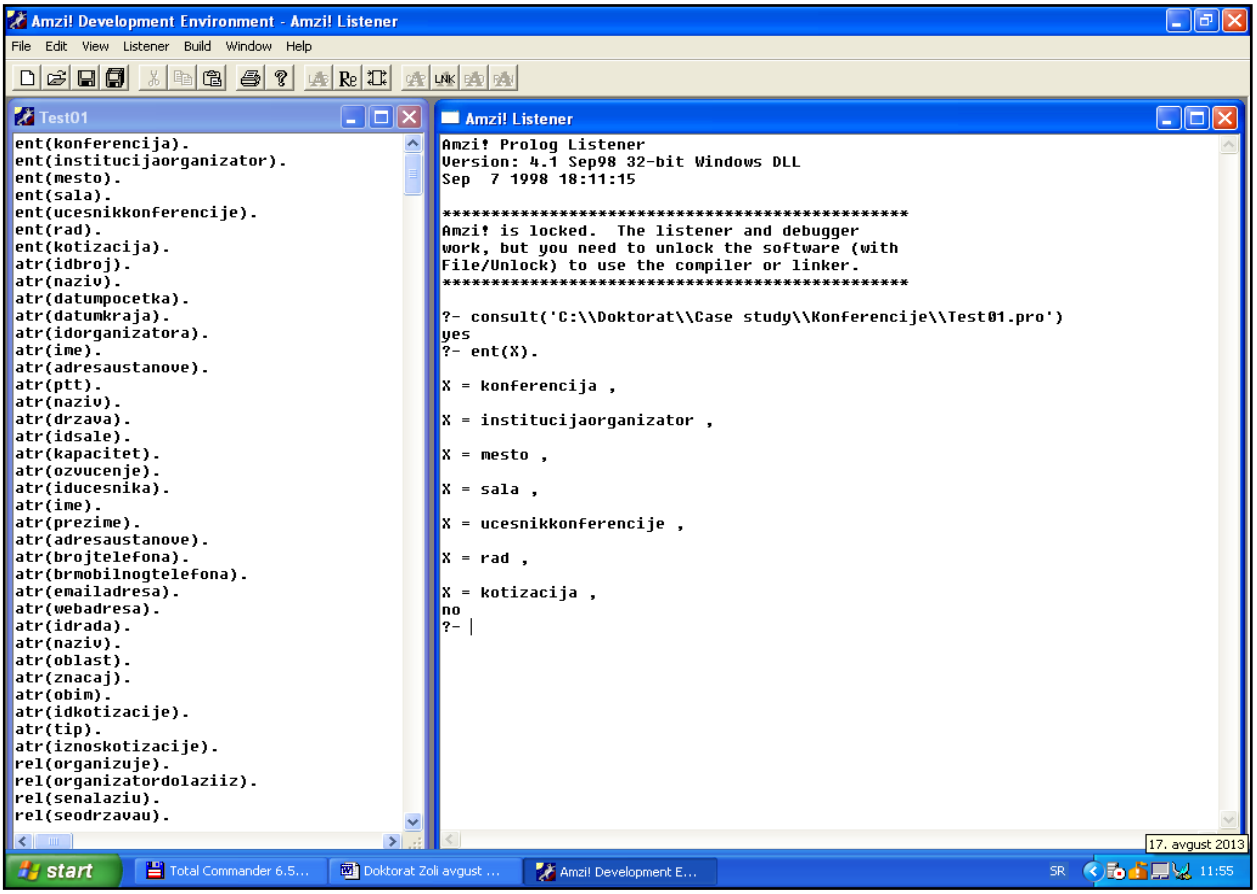
Slika 6.3. – Parsiranje elemenata ontologije u SWI Prologu

6.1.4. AMZI! Prolog

AMZI! Prolog je jezik logičkog programiranja koji se nudi korisnicima u dve verzije: besplatna (*Shareware*) i komercijalna. Verzija *Amzi!® Logic Explorer* se može izvršavati na svim Windows platformama i besplatna je za ličnu upotrebu, te se kao takva može preuzeti sa Interneta [16]. Prva verzija se pojavila 1994. godine.

Neke karakteristike AMZI! Prolog jezika:

- Komponente se mogu integrisati sa Windows, Linux, Sun Solaris, HP/UX operativnim sistemima.
- Poseduje podršku za programske jezike: C, *C++, Java, Web Servers (ASP.NET, JSP, Java Servlets, CGI), Delphi, VB.NET, C#.NET, PowerBuilder, VBA u Access, VBA u Excel* i druge.
- Moguće je konsultovati izvorne i kompajlirane datoteke u okviru istog programa.
- Omogućuje rad sa relativnim putanjama projekata.
- Ima podršku za ODBC tipove podataka i baze podataka (samo pod Windows okruženjem).
- Amzi! komunikaciju sa korisnikom ostvaruje preko komandne linije ili IDE alata koji integriše editor koda, «*listener*», «*debugger*», «*compiler*», «*linker*» i «*runtime engine*» (slika 6.4).
- Poseduje podršku za izgradnju Internet aplikacija.
- Ugrađen je sistem za otkrivanje i ispravku grešaka u programima.

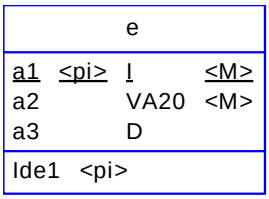


Slika 6.4. - Testiranje korektnosti modela podataka u AMZI! Prologu

6.2. KREIRANJE MODELA PODATAKA U CASE ALATU I TRANSFORMACIJA U PROLOG REČENICE

U cilju realizacije empirijskog istraživanja bilo je potrebno formalizovati modele podataka urađene u Power Designer CASE alatu i prevesti ih u oblik Prolog klauzula, kako je opisano u modelu.

Prikaz dijagrama entiteta konceptualnog modela podataka u CASE alatu:



Slika 6.5. – Prikaz entiteta i atributa u CASE alatu

Ograničenja modela podataka, u okviru CASE alata:

a_1 (dom_1): ceo broj (integer - I), obavezna vrednost atributa (mandatory=true),

a_2 (dom_2): string (20) (variable character(20) – VA), obavezna vrednost atributa (mandatory=true),

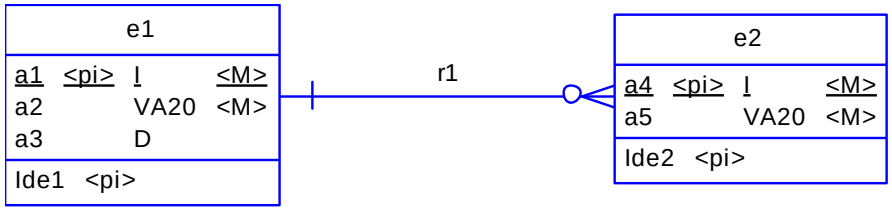
a_3 (dom_3): datum (date - D), neobavezna vrednost atributa (mandatory=false).

Prolog klauzule, prema (3.9), su:

```
ent(e).
atr(a1).
atr(a2).
atr(a3).
res(idatr).
res(mandatory).
res(i).
res(va).
res(d).
p(a1, i).
p(a2, va).
p(a3, d).
p(e, a1).
p(e, a2).
p(e, a3).
p(e, idatr).
p(a1, mandatory).
p(a2, mandatory).
p(a1, idatr).
```

Ukoliko je vrednost atributa neobavezna (mandatory=false), tada se u skupu *P* ne pojavljuje element koji povezuje odgovarajuće elemente skupa *S* i *A*, što znači da i u formi Prolog klauzula postoje samo one činjenice koje ukazuju na obaveznu vrednost atributa (npr. p(a1, mandatory) ili p(a2, mandatory)).

Prikaz dijagrama entiteta i poveznika konceptualnog modela podataka u CASE alatu:



Slika 6.6. - Prikaz poveznika, entiteta i atributa u CASE alatu

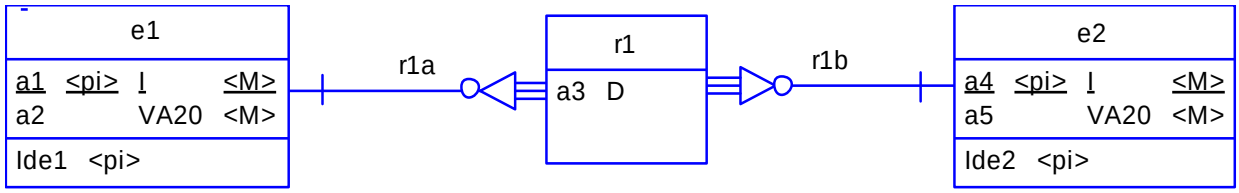
Ograničenja modela podataka:

a_1 (dom_1): ceo broj (integer - I), obavezna vrednost atributa (mandatory=true),
a_2 (dom_2): string (20) (variable character(20) – VA), obavezna vrednost atributa (mandatory=true),
a_3 (dom_3): datum (date - D), neobavezna vrednost atributa (mandatory=false),
a_4 (dom_4): ceo broj (integer - I), obavezna vrednost atributa (mandatory=true),
a_5 (dom_5): string (20) (variable character(20) – VA), obavezna vrednost atributa (mandatory=true),
dgk_1=0 (mincard0),
ggk_1=m (maxcardm),
dgk_2=1 (mincard1),
ggk_2=1 (maxcard1).

Prolog klauzule, prema (3.9), su:

```
ent(e1).          res(maxcard1).      p(e1, a1).          p(e1, r1).
ent(e2).          res(mincard0).      p(e1, a2).          p(r1, e2).
atr(a1).          res(maxcardm).      p(e1, a3).          p(mincard1, r1).
atr(a2).          res(i).             p(e2, a4).          p(maxcard1, r1).
atr(a3).          res(va).            p(e2, a5).          p(r1, mincard0).
atr(a4).          res(d).             p(e1, idatr).       p(r1, maxcardm).
atr(a5).          p(a1, i).           p(e2, idatr).       p(a1, idatr).
rel(r1).          p(a2, va).          p(a1, mandatory).   p(a4, idatr).
res(idatr).       p(a3, d).           p(a2, mandatory).
res(mandatory).   p(a4, i).           p(a4, mandatory).
res(mincard1).    p(a5, va).          p(a5, mandatory).
```

Prikaz dijagrama poveznika sa atributom konceptualnog modela podataka u CASE alatu:



Slika 6.7. - Prikaz entiteta sa atributima i poveznika sa atributom u CASE alatu

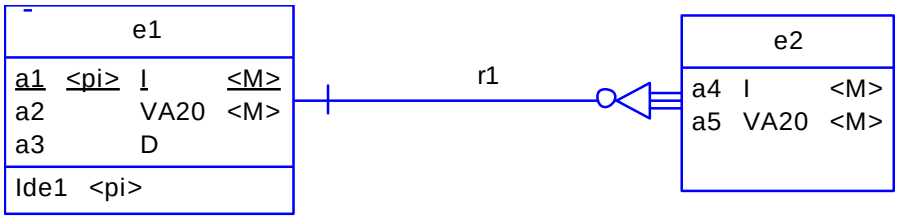
Ograničenja modela podataka:

a_1 (dom_1): ceo broj (integer - I), obavezna vrednost atributa (mandatory=true),
a_2 (dom_2): string (20) (variable character(20) – VA), obavezna vrednost atributa (mandatory=true),
a_3 (dom_3): datum (date - D), neobavezna vrednost atributa (mandatory=false),
a_4 (dom_4): ceo broj (integer - I), obavezna vrednost atributa (mandatory=true),
a_5 (dom_5): string (20) (variable character(20) – VA), obavezna vrednost atributa (mandatory=true),
dgk_1=1 (mincard1),
ggk_1=1 (maxcard1),
dgk_2=0 (mincard0),
ggk_2=m (maxcardm).

Prolog klauzule, prema (3.9), su:

```
ent(e1).            res(maxcardm).      p(e1, idatr).          p(maxcardm, r1b).
ent(e2).            res(dependent).     p(e2, idatr).          p(r1b, mincard1).
ent(r1).            res(i).             p(a1, mandatory).      p(r1b, maxcard1).
atr(a1).            res(va).            p(a2, mandatory).      p(dependent, r1a).
atr(a2).            res(d).             p(a4, mandatory).      p(r1b, dependent).
atr(a3).            p(a1, i).           p(a5, mandatory).      p(a1, idatr).
atr(a4).            p(a2, va).          p(r1, r1a).            p(a4, idatr).
atr(a5).            p(a4, d).           p(r1a, e1).
rel(r1a).           p(a5, i).           p(e2, r1b).
rel(r1b).           p(a3, va).          p(r1b, r1).
res(idatr).         p(e1, a1).          p(mincard1, r1a).
res(mandatory).     p(e1, a2).          p(maxcard1, r1a).
res(mincard1).      p(e2, a4).          p(r1a, mincard0).
res(maxcard1).      p(e2, a5).          p(r1a, maxcardm).
res(mincard0).      p(r1, a3).          p(mincard0, r1b).
```

Prikaz dijagrama identifikacione zavisnosti entiteta u konceptualnom modelu podataka:



Slika 6.8. - Prikaz veze između jakog i slabog tipa entiteta (identifikaciona zavisnost)

Ograničenja modela podataka:

a_1 (dom_1): ceo broj (integer - I), obavezna vrednost atributa (mandatory=true),

a_2 (dom_2): string (20) (variable character(20) – VA), obavezna vrednost atributa (mandatory=true),

a_3 (dom_3): datum (date - D), neobavezna vrednost atributa (mandatory=false),

a_4 (dom_4): ceo broj (integer - I), obavezna vrednost atributa (mandatory=true),

a_5 (dom_5): string (20) (variable character(20) – VA), obavezna vrednost atributa (mandatory=true),

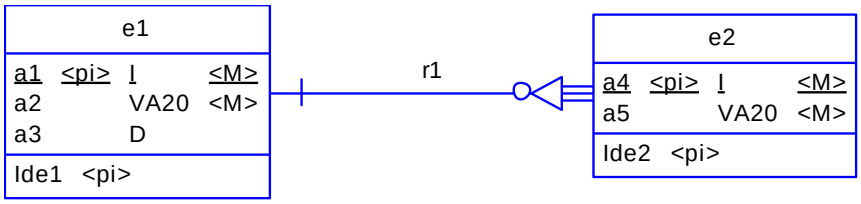
k_1=ggk_1=m (maxcardm), dgk_1=0 (mincard0),

k_2=ggk_2=1 (maxcard1), dgk_2=1 (mincard1).

Prolog klauzule, prema (3.9), su:

```
ent(e1).            res(maxcard1).      p(a4, i).              p(a5, mandatory).
ent(e2).            res(mincard0).      p(a5, va).             p(e2, r1).
atr(a1).            res(maxcardm).      p(e1, a1).             p(r1, e1).
atr(a2).            res(dependent).     p(e1, a2).             p(mincard0, r1).
atr(a3).            res(inherit).       p(e1, a3).             p(maxcardm, r1).
atr(a4).            res(i).             p(e2, a4).             p(r1, mincard1).
atr(a5).            res(va).            p(e2, a5).             p(r1, maxcard1).
rel(r1).            res(d).             p(e1, idatr).          p(dependent, r1).
res(idatr).         p(a1, i).           p(a1, mandatory).      p(a1, idatr).
res(mandatory).     p(a2, va).          p(a2, mandatory).
res(mincard1).      p(a3, d).           p(a4, mandatory).
```

Prikaz dijagrama egzistencijalne zavisnosti entiteta u okviru konceptualnog modela podataka:



Slika 6.9. - Prikaz veze između jakog i slabog tipa entiteta (egzistencijalna zavisnost)

Ograničenja modela podataka:

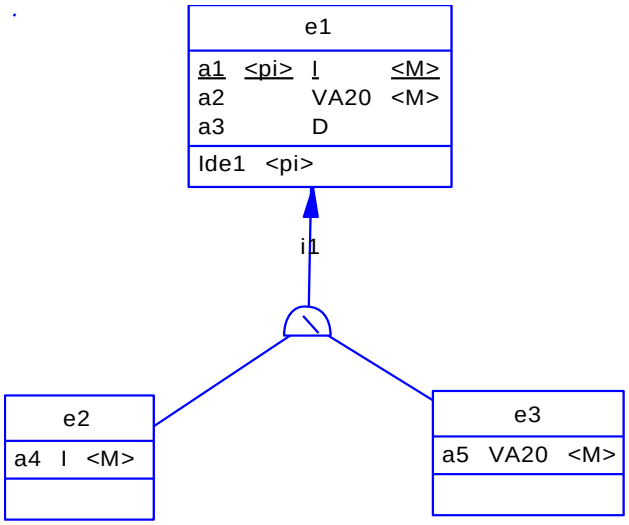
a_1 (dom_1): ceo broj (integer - I), obavezna vrednost atributa (mandatory=true),
a_2 (dom_2): string (20) (variable character(20) – VA), obavezna vrednost atributa (mandatory=true),
a_3 (dom_3): datum (date - D), neobavezna vrednost atributa (mandatory=false),
a_4 (dom_4): ceo broj (integer - I), obavezna vrednost atributa (mandatory=true),
a_5 (dom_5): string (20) (variable character(20) – VA), obavezna vrednost atributa (mandatory=true),
k_1=ggk_1=m (maxcardm),
dgk_1=0 (mincard0),
k_2=ggk_2=1 (maxcard1),
dgk_2=1 (mincard1).

Prolog klauzule , prema (3.9), su:

```
ent(e1).            res(maxcard1).      p(a4, i).           p(a4, mandatory).
ent(e2).            res(mincard0).      p(a5, va).          p(a5, mandatory).
atr(a1).            res(maxcardm).      p(e1, a1).          p(e2, r1).
atr(a2).            res(dependent).     p(e1, a2).          p(r1, e1).
atr(a3).            res(inherit).       p(e1, a3).          p(mincard1, r1).
atr(a4).            res(i).             p(e2, a4).          p(maxcard1, r1).
atr(a5).            res(va).            p(e2, a5).          p(r1, mincard0).
rel(r1).            res(d).             p(e1, idatr).       p(r1, maxcardm).
res(idatr).         p(a1, i).           p(e2, idatr).       p(dependent, r1).
res(mandatory).     p(a2, va).          p(a1, mandatory).   p(a1, idatr).
res(mincard1).      p(a3, d).           p(a2, mandatory).   p(a4, idatr).
```

Prema [Mogin i sar. 1996] kardinalnost preslikavanja potklasa na skup pojave superklase je uvek (1,1), jer svakoj pojavi bilo koje potklase odgovara samo jedna pojava nadklase i ovaj kardinalitet se na dijagramima ne navodi. Zbog toga je kardinalnost dgk_2=ggk_2=1. Oznaka ovog preslikavanja u skupu ograničenja je *nadtip*, tj. *inherit* u CASE alatu.

Prikaz dijagrama IS_A hijerarhije konceptualnog modela podataka u CASE alatu je dat na slici 6.10.



Slika 6.10. - Prikaz IS_A hijerarhije u CASE alatu

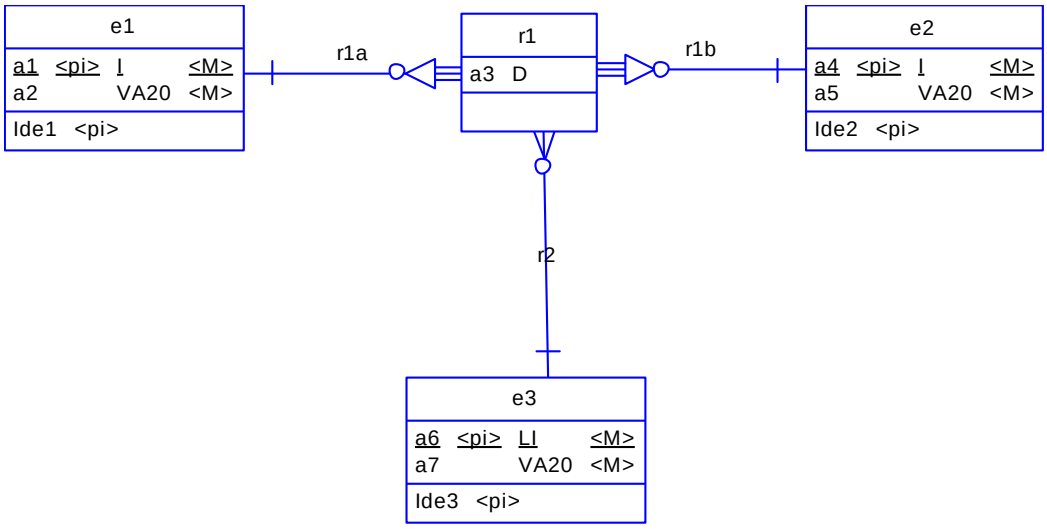
Ograničenja modela podataka:

a_1 (dom_1): ceo broj (integer - I), obavezna vrednost atributa (mandatory=true),
a_2 (dom_2): string (20) (variable character(20) – VA), obavezna vrednost atributa (mandatory=true),
a_3 (dom_3): datum (date - D), neobavezna vrednost atributa (mandatory=false),
a_4 (dom_4): ceo broj (integer - I), obavezna vrednost atributa (mandatory=true),
a_5 (dom_5): string (20) (variable character(20) – VA), obavezna vrednost atributa (mandatory=true),
nadtip: inherit,
dgk_1=1 (mincard1),
ggk_1=m (maxcardm).

Prolog klauzule, prema (3.9), su:

```
ent(e1).          res(mincard1).       p(a3, va).          p(a2, mandatory).
ent(e2).          res(maxcard1).       p(a4, d).           p(a4, mandatory).
ent(e3).          res(maxcardm).       p(a5, i).           p(a5, mandatory).
atr(a1).          res(dependent).      p(e1, a1).          p(a1, idatr).
atr(a2).          res(inherit).        p(e1, a2).          p(i1, inherit).
atr(a3).          res(i).              p(e1, a3).          p(i1, mincard1).
atr(a4).          res(va).             p(e2, a4).          p(i1, maxcardm).
atr(a5).          res(d).              p(e3, a5).          p(e1, i1).
res(idatr).       p(a1, i).            p(e1, idatr).       p(i1, e2).
res(mandatory).   p(a2, i).            p(a1, mandatory).   p(i1, e3).
```

Prikaz entiteta i mešovitog objekta-veze (gerunda) konceptualnog modela podataka u CASE alatu je prikazan na slici 6.11.



Slika 6.11. - Prikaz entiteta i gerunda u CASE alatu

Ograničenja modela podataka:

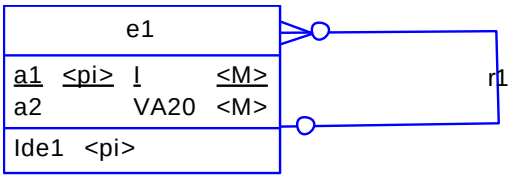
a_1 (dom_1): ceo broj (integer - I), obavezna vrednost atributa (mandatory=true),
a_2 (dom_2): string (20) (variable character(20) – VA), obavezna vrednost atributa (mandatory=true),
a_3 (dom_3): datum (date - D), neobavezna vrednost atributa (mandatory=false),
a_4 (dom_4): ceo broj (integer - I), obavezna vrednost atributa (mandatory=true),
a_5 (dom_5): string (20) (variable character(20) – VA), obavezna vrednost atributa (mandatory=true),
a_6(dom_6): ceo broj većeg opsega (long integer - LI), obavezna vrednost atributa (mandatory=true),
a_7(dom_7): string (20) (variable character(20) – VA), obavezna vrednost atributa (mandatory=true),
dgk_1=1 (mincard1),
ggk_1=1 (maxcard1),
dgk_2=0 (mincard0),
ggk_2=m (maxcardm).

Prolog klauzule, prema (3.9), su:

```
ent(e1).            res(mincard1).      p(e1, a1).          p(r1, r1a).
ent(e2).            res(maxcard1).      p(e1, a2).          p(r1a, e1).
ent(r1).            res(mincard0).      p(e2, a4).          p(e2, r1b).
ent(e3).            res(maxcardm).      p(e2, a5).          p(r1b, r1).
atr(a1).            res(dependent).     p(r1, a3).          p(r1, r2).
atr(a2).            res(i).             p(e3, a6).          p(r2, e3).
atr(a4).            res(va).            p(e3, a7).          p(mincard1, r1a).
atr(a5).            res(d).             p(e1, idatr).       p(maxcard1, r1a).
atr(a3).            res(li).            p(e2, idatr).       p(r1a, mincard0).
atr(a6).            p(a1, i).           p(e3, idatr).       p(r1a, maxcardm).
atr(a7).            p(a2, va).          p(a1, mandatory).   p(mincard0, r1b).
rel(r1a).           p(a4, d).           p(a2, mandatory).   p(maxcardm, r1b).
rel(r1b).           p(a5, i).           p(a4, mandatory).   p(r1b, mincard1).
rel(r2).            p(a3, va).          p(a5, mandatory).   p(r1b, maxcard1).
res(idatr).         p(a6, li).          p(a6, mandatory).   p(mincard1, r2).
res(mandatory).     p(a7, va).          p(a7, mandatory).   p(maxcard1, r2).
```

```
p(r2, mincard0).   p(dependent, r1a).   p(r1b, dependent).    p(a4, idatr).
p(r2, maxcardm).                        p(a1, idatr).         p(a6, idatr).
```

Prikaz entiteta i unarnog poveznika konceptualnog modela podataka u CASE alatu:



Slika 6.12. - Grafički prikaz entiteta i unarnog poveznika u CASE alatu

Ograničenja modela podataka:

a_1 (dom_1): ceo broj (integer - I), obavezna vrednost atributa (mandatory=true),
a_2 (dom_2): string (20) (variable character(20) – VA), obavezna vrednost atributa (mandatory=true),
dgk_1=0 (mincard0),
ggk_1=m (maxcardm),
dgk_2=0 (mincard0),
ggk_2=1 (maxcard1).

Prolog klauzule, prema (3.9), su:

```
ent(e1).            res(maxcard1).      p(a2, va).            p(e1, r1).
atr(a1).            res(mincard0).      p(e1, a1).            p(r1, e1).
atr(a2).            res(maxcardm).      p(e1, a2).            p(mincard0, r1).
rel(r1).            res(i).             p(e1, idatr).         p(maxcard1, r1).
res(idatr).         res(va).            p(a1, mandatory).     p(r1, mincard0).
res(mandatory).     p(a1, i).           p(a2, mandatory).     p(r1, maxcardm).
```

6.2.1. Usklađivanje tipova podataka u CASE alatu i ontologiji

U pravilu zaključivanja R 5.7, koje se koristi za proveru ograničenja osobine podataka na nivou ontologije i tipa podatka nekog atributa u ER modelu podataka, proverava se da li postoje ontološka osobina podataka i atribut istog naziva, sa istim tipom podataka, pri čemu nisu razmatrana dodatna ograničenja koja se odnose na broj karaktera, cifara, decimalnih mesta i sl. Prilikom definisanja ovog pravila potrebno je bilo uskladiti različite oznaka tipova podataka u ontologiji i modelu podataka. Oznaka za celobrojni tip podatka u RDF ontologiji je «integer», dok je u modelu podataka implementiranom u Power Designer CASE alatu oznaka «i», stringovni tip podatka u ontologiji se predstavlja oznakom «string» a u CASE alatu su to «va» i «txt» za veće opise. Zbog toga je formirana baza činjenica, prikazana u listingu 6.1., koja služi usklađivanju ovih sintaksnih različitosti. Definisan je predikat «datatype» koji ima dva argumenta – prvi je oznaka tipa podatka u ontologiji, a drugi u modelu podataka.

Listing 6.1:

```
datatype(integer,i).
datatype(integer,no).
datatype(integer,n).
datatype(positiveinteger,i).
datatype(positiveinteger,no).
datatype(positiveinteger,n).
datatype(short,si).
datatype(short,no).
datatype(short,n).
datatype(long,li).
datatype(long,no).
datatype(long,n).
datatype(byte,bt).
datatype(byte,no).
datatype(byte,n).
datatype(decimal,dc).
datatype(decimal,mn).
datatype(boolean,bl).
datatype(string,txt).
datatype(string,va).
datatype(string,la).
datatype(string,a).
datatype(datetime,dt).
datatype(datetime,d).
datatype(datetime,t).
```

6.3. MAPIRANJE ONTOLOGIJE POMOĆU PROLOGA

Ontologija se kreira u odgovarajućem ontološkom editoru, poput *Protégé-a,* u kom se definisani elementi ontologije mogu predstavljati u različitim formatima, između ostalog u OWL i RDF jezicima. Mapiranje elemenata ontologije na predikatski račun prvog reda i Prologolike klauzule se može izvršiti pomoću SWI Prolog jezika. Na osnovu rezultata rada SWI Prologa su, u disertaciji, izdvojeni RDF tripleti i utvrđeno je da se OWL datoteka ontologije može parsirati u oblik literala na predikatskom zapisu. Da bi se ovo postiglo, prvo se u okviru SWI Prolog okruženja, moraju uključiti biblioteke za rad sa RDF ontologijama:

```
?- use_module(library(semweb/rdf_db)).
```

Zaitm se uključuje i biblioteka za rad sa RDF šemama:

```
?- use_module(library(semweb/rdfs)).
```

Na kraju se uključuje i treća biblioteka - za kreiranje i rad sa imenovanim prostorima u RDF datotekama:

```
?- use_module(library(semweb/rdf_portray)).
```

Nakon ukjučenih biblioteka za rad sa ontologijama, tj. semantičkim Web-om, postavlja se upit za učitavanje RDF ontologije u Prolog i parsiranje RDF trojki, tj. triplet-a prikazanih u listingu 6.2.

Kao što je u teorijskom istraživanju istaknuto, elementi RDF triplet-a su URI nizovi karaktera za identikovanje abstraktnog ili fizičkog resursa na Web-u [Antoniou et al.,

2005], [6], [7]. Pošto su komponente URI-ja potencijalno prilično dugački stringovi, koji se u samom RDF dokumentu razlikuju na samom kraju XML elementa, gde se navode subjekat, predikat i objekat. Zbog toga se koriste entiteti za konstantne delove elementa koji se nazivaju imenovani prostori (*Namespace*).

Listing 6.2:

```
?-rdf_load('onto.rdf'), rdf(S, P, O).
% Parsed "onto.rdf" in 0.01 sec; 82 triples
S = 'http://www.semanticweb.org/ontologies/2013/0/Onto.owl',
P = rdf:type,
O = owl:'Ontology' ;
S = 'http://www.semanticweb.org/ontologies/2013/0/Onto.owl#Relacija1',
P = rdf:type,
O = owl:'ObjectProperty' ;
S = 'http://www.semanticweb.org/ontologies/2013/0/Onto.owl#Relacija2',
P = rdf:type,
O = owl:'ObjectProperty' ;
...
S = 'http://www.semanticweb.org/ontologies/2013/0/Onto.owl#Atribut1',
P = rdf:type,
O = owl:'DatatypeProperty' ;
S = 'http://www.semanticweb.org/ontologies/2013/0/Onto.owl#Atribut1',
P = rdfs:range,
O = xsd:integer ;
...
S = 'http://www.semanticweb.org/ontologies/2013/0/Onto.owl#Klasa3',
P = rdfs:subClassOf,
O = owl:'Thing' ;
S = 'http://www.semanticweb.org/ontologies/2013/0/Onto.owl#Klasa4',
P = rdf:type,
O = owl:'Class' ;
S = 'http://www.semanticweb.org/ontologies/2013/0/Onto.owl#Klasa4',
P = rdfs:subClassOf,
O = 'http://www.semanticweb.org/ontologies/2013/0/Onto.owl#Klasa3' ;
S = 'http://www.semanticweb.org/ontologies/2013/0/Onto.owl#Klasa5',
P = rdf:type,
O = owl:'Class' ;
S = 'http://www.semanticweb.org/ontologies/2013/0/Onto.owl#Klasa5',
P = rdfs:subClassOf,
...
```

Prilikom učitavanja ontologije, formiraju se tri imenovana prostora, kako bi se povećala čitljivost i olakšao rad sa predikatima koje formira SWI Prolog:

```
?- rdf_register_ns(subj,'http://www.semanticweb.org/
   ontologies/2013/0/Onto.owl#').
?- rdf_register_ns(pred,'http://www.w3.org/1999/02/22
   -rdf-syntax-ns#').
?- rdf_register_ns(obj,'http://www.w3.org/2002/07/owl#').
```

Postavljanje upita SWI Prologu za parsiranje RDF-a, rezultuje, u apstraktnoj ontologiji koja je prikazana u teoretskom modelu (Slike 5.10., 5.11., 5.12. i 5.13.), formiranjem približno 100 tripleta. Upit za formiranje RDF tripleta glasi:

```
?- rdf(S, P, O).
```

SWI Prolog, ipak, nije upotrebljen u implementaciji modela semantičke analize modela podataka za testiranje pravila zaključivanja, s obzirom da je za primer ontologije koja je prikazana u eksperimentalnom istraživanju disertacije, postavljen upit `?-rdf(S,P,O)`

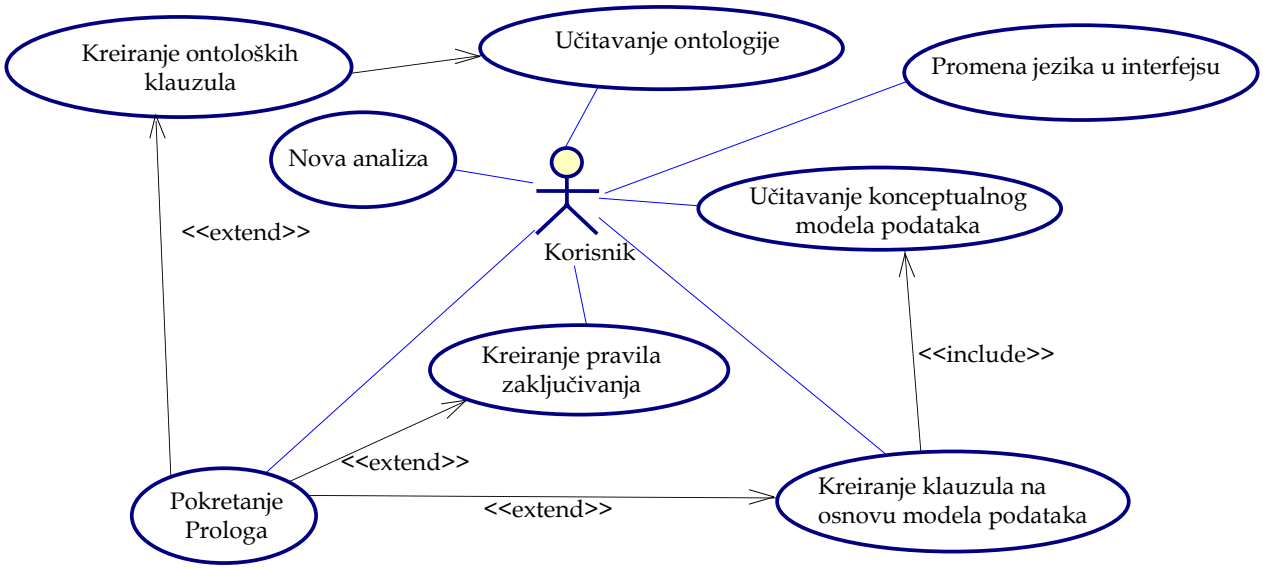
rezultovao sa preko 300 RDF tripleta. Od ovog broja uređenih trojki, izvestan broj nema dobro diferencirane subjekte, predikate i objekte, te zbog toga, dalje istraživanje i praktična primena SWI Prologa za rad sa ontologijama nije izvršena. Umesto toga, urađen je deo DMV programa koji vrši parsiranje ontologije i formira RDF triplete.

6.4. OPIS I IMPLEMENTACIJA APLIKACIJE DATA MODEL VALIDATOR - DMV

Windows aplikacija Data Model Validator (DMV) je programirana u Microsoft Visual Studio .Net okruženju, u programskom jeziku *Visual Basic*, sa ciljem da se izvrši integracija nekoliko koraka u procesu provere semantičke korektnosti modela.

Ovaj softver omogućuje:

1. Učitavanje konceptualnog modela podataka kreiranog u Power Designer CASE alatu i njegovo transformisanje iz XML oblika, u oblik Prolog rečenica.
2. Učitavanje RDF ontologije kreirane u Protégé ontološkom alatu i njenu transformaciju iz XML oblika, u oblik RDF tripleta kao Prolog činjenica.
3. Učitavanje, tj. dodavanje pravila zaključivanja Prolog programu.
4. Pokretanje Prolog okruženja u cilju testiranja korektnosti modela podataka.
5. Dvojezični korisnički interfejs, na srpskom i engleskom jeziku.



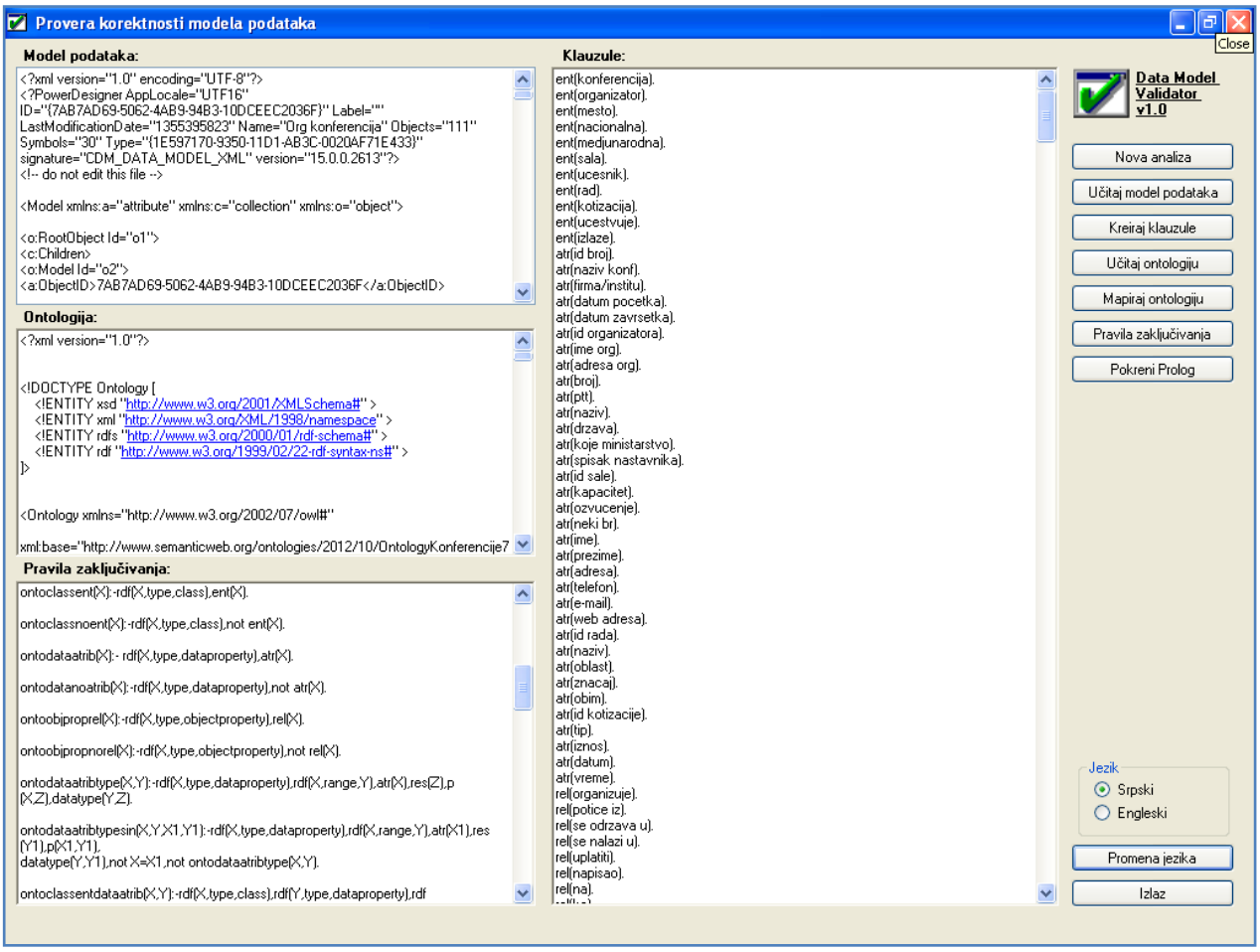
Slika 6.13. – Dijagram slučajeva korišćenja Data Model Validator softvera

Kao što se na slici 6.13. može videti, korisnik DMV programa ima na raspolaganju nekoliko softverskih funkcija navedenih pod stavkama 1, 2, 3, 4 i 5.

Slučaj korišćenja programa "Kreiranje klauzula na osnovu modela podataka" obuhvata sledeće podfunkcije: prikaz entiteta, atributa, relacija, tipova podataka, atributa u entitetima, entiteta sa identifikacionim atributima, obaveznih atributa, identifikacionih atributa, relacija između entiteta, kardinaliteta relacija, zavisnih relacija i IS_A hijerarhija.

Slučaj korišćenja programa "Kreiranje ontoloških klauzula" obuhvata sledeće podfunkcije:

- Izdvajanje klasa,
- Izdvajanje osobina objekata,
- Izdvajanje osobine podataka,
- Izdvajanje objekata,
- Izdvajanje istih osobina objekata,
- Izdvajanje podklasa,
- Izdvajanje tipova podataka,
- Izdvajanje veza objekata i klasa,
- Izdvajanje veza objekata i tipova podataka,
- Izdvajanje veza osobina objekata,
- Izdvajanje ograničenja osobina objekata.



Slika 6.14. – Aplikacija Data Model Validator (DMV)

6.4.1. Implementacija softverske funkcije za učitavanje konceptualnog modela podataka

Kao što je već ranije istaknuto rezultat konceptualnog modelovanja podataka, u Power Designer CASE alatu, je datoteka sa ekstenzijom .CDM, koja je zapravo XML dokument koji sadrži oznake elemenata modela podataka. Učitavanje ove datoteke u odgovarajuće strukture podataka i promenljive zahtevalo je uključivanje biblioteke klasa za rad sa

ulazno-izlaznim uređajima u okviru *Visual Basic* programskog jezika. Zatim je, na formu korisničkog interfejsa, postavljena kontrola »*OpenFileDialog*« koja se koristi za selektovanje datoteka u prozoru koji se tom prilikom otvara. Instancira se globalni objekat na nivou aplikacije *fileModel* tipa klase »*StreamReader*«, koja omogućuje čitanje tekstualne XML datoteke metodom »*ReadToEnd*« i dodeljivanje njenog sadržaja promenljivoj *VCeoModel*, stringovnog tipa.

Listing 6.3:

```
Imports System.IO

Dim fileModel As System.IO.StreamReader
Dim vCeoModel As String = ""

OpenFileDataModel.ShowDialog()
filePath = OpenFileDataModel.FileName
fileModel = New System.IO.StreamReader(filePath)
vCeoModel = fileModel.ReadToEnd
fileModel.Close()
txtModel.Text = vCeoModel.ToString
```

6.4.2. Implementacija softverske funkcije za kreiranje klauzula na osnovu modela podataka

Realizacija kreiranja Prolog klauzula na osnovu odgovarajućih elemenata modela podataka bazirana je na višestrukom parsiranju promenljive *vCeoModel*. Prvo su identifikovani entiteti, zatim atributi, relacije između entiteta, identifikacioni atributi itd. Formirana je lista entiteta (Listing 6.4), što je zahtevalo uključivanje biblioteke *System.Collections*:

Listing 6.4:

```
Imports System.Collections

Dim listaEntiteta As ArrayList
listaEntiteta = New ArrayList
```

Na formu su postavljene i ostale komponente, tj. kontrole tipa: *TextBox, Label, RichTextBox, CommandButton, OptionButton*, u kojima se prikazuju međurezultati celokupnog postupka.

Listing 6.5 prikazuje XML elemente konceptualnog modela podataka na osnovu kojih je određen element skupa *E*={konferencija} i na osnovu koje je formirana Prolog činjenica: ent(konferencija).

Listing 6.5:

```
<o:Entity Id="o46">
<a:Name>konferencija</a:Name>
<a:Code>KONFERENCIJA</a:Code>
...
</o:Entity>
```

Implementacija ovog dela programa realizovana je pomoću Visual Basic funkcije INSTR(string, podstring) koja locira poziciju početka podstringa u okviru nekog stringa. Zatim se funkcijom MID(string, lokacijapočetka, dužinaodsecanja) formira novi string koji sadrži odgovarajući podstring početnog stringa, tj. celog modela podataka, a koji počinje XML elementom `<o:Entity>` i završava se sa zatvarajućom oznakom objekta `</o:Entity>`.Naziv entiteta je određen na osnovu `<a:Code>` elementa u okviru

elementa <o:Entity>, zbog toga što u CASE alatu kod objekta inicijalno predstavlja jedinstveni identifikator u modelovanju, te se kao takav ne može duplirati i navoditi više puta. Deklarisane su odgovarajuće pomoćne promenljive tipa *String* i *Integer* za formiranje podstringova modela na osnovu XML oznaka elemenata do samog naziva entiteta, u ovom slučaju.

Kada se na kraju izdvoji sam naziv entiteta, isti se memoriše u promenljivoj *vEntityIDAtributName*, čija se vrednost dodaje *Text* svojstvu odgovarajućeg polja za prikaz naziva svih entiteta i listi entiteta koja se koristi za formiranje Prolog klauzula. Promenljiva koja sadrži ceo model se zatim smanjuje MID funkcijom, tako što se odbacuje već izanalizirani deo stringa, tj. modela. Pronalazi se sledeći <o:Entity> XML element i ceo postupak se ponavlja *While* ciklusom dok se ne isparsiraju svi elementi tipa <o:Entity>, da bi se pronašli svi entiteti u modelu podataka (Listing 6.6).

Listing 6.6:

```
Dim vModel As String = ""
Dim vEntity As String = ""
Dim vEntityIDAtributCode As String = ""
Dim vEntityIDAtributName As String = ""
Dim pozicijaPocetkaEntiteta As Integer = 0
Dim pozicijaKrajaEntiteta As Integer = 0
Dim pozicijaPocetkaIDEntiteta As Integer = 0
Dim pozicijaKrajaIDEntiteta As Integer = 0
vModel = vCeoModel

While Not vModel = ""
   pozicijaPocetkaEntiteta = InStr(vModel, "<o:Entity Id")
   pozicijaKrajaEntiteta = InStr(vModel, "</o:Entity>")
   vEntity = Mid(vModel, pozicijaPocetkaEntiteta, pozicijaKrajaEntiteta -
   pozicijaPocetkaEntiteta + 11)
   pozicijaPocetkaIDEntiteta = InStr(vEntity, "<a:Code>")
   pozicijaKrajaIDEntiteta = InStr(vEntity, "</a:Code>")
   vEntityIDAtributCode = Mid(vEntity, pozicijaPocetkaIDEntiteta,
   pozicijaKrajaIDEntiteta - pozicijaPocetkaIDEntiteta + 9)
   pozicijaPocetkaIDEntiteta = InStr(vEntityIDAtributCode, ">")
   pozicijaKrajaIDEntiteta = InStr(vEntityIDAtributCode, "</")
   vEntityIDAtributName = Mid(vEntityIDAtributCode, pozicijaPocetkaIDEntiteta
   + 1, pozicijaKrajaIDEntiteta - pozicijaPocetkaIDEntiteta - 1)
   txtEntities.Text = txtEntities.Text + vEntityIDAtributName + ";  "
   listaEntiteta.Add(vEntityIDAtributName)
   vModel = Mid(vModel, pozicijaKrajaEntiteta + 1, Len(vModel) -
pozicijaKrajaEntiteta)
End While
...
```

Kreiranje liste atributa je nešto složenije nego formiranje liste entiteta zbog toga što se opisuju većim brojem XML elemenata koji se nalaze na različtim lokacijama stringa modela podataka. U okviru <o:Entity> elementa se nalazi jedan ili više <c:DataItem> elemenata. Ovaj element, koji opisuje elementarni podatak, tj. atribut ER modela podataka, ima svoju referencu već u narednom elementu <o:DataItem Ref="o60"/>. Na osnovu ove reference atributa, pronalazi se drugi element <o:DataItem Id="o60">, u kojem se u okviru narednog elementa <a:Code>ID_BROJ</a:Code> izdvaja naziv atributa, na sličan način kako je to urađeno i u slučaju entiteta.

Listing 6.7:

```
<c:DataItem>
<o:DataItem Ref="o60"/>
</c:DataItem>
</o:EntityAttribute>
<o:EntityAttribute Id="o61">
<a:BaseAttribute.Mandatory>1</a:BaseAttribute.Mandatory>
<c:DataItem>
...
<o:DataItem Id="o60">
<a:Name>id broj</a:Name>
<a:Code>ID_BROJ</a:Code>
<a:DataType>I</a:DataType>
</o:DataItem>
```

Listing 6.7 prikazuje XML elemente datoteke modela podataka na osnovu kojih je formiran član skupa *A*={id_broj} i Prolog činjenica atr(id_broj).

U prethodno navedenim XML elementima se mogu odrediti i ograničenja vezana za osobinu obaveznosti atributa, preko osobine «Mandatory», čija je vrednost u elementu `<a:BaseAttribute.Mandatory>1</a:BaseAttribute.Mandatory>`. Na osnovu ove karakteristike se, uz prethodno formirane elemente skupa ograničenja *R*={i, mandatory}, (u Prolog formi: res(i) i res(mandatory), gde je «i» oznaka Integer tipa podatka u CASE alatu), mogu kreirati elementi skupa *P* koji povezuju entitet i atribut, tip podatka i atribut, kao i atribut i osobinu obavezan atribut:

P={P(konferencija, id_broj), P(id_broj, i), P(id_broj, mandatory)}

U Prolog formi to su sledeće tri činjenice:

p(konferencija, id_broj).
p(id_broj, i).
p(id_broj, mandatory).

Ukoliko element `<a:BaseAttribute.Mandatory>1</a:BaseAttribute.Mandatory>` ne postoji, osobina atirbuta «Mandatory» je postavljena na netačno, tj. false, što znači da atribut ne mora uvek imati vrednost u kasnijoj eksploataciji baze podataka.

Listing 6.8 ilustruje niz Visual Basic naredbi sa deklaracijama promenljivih potrebnih za parsiranje delova modela kojima se izdvajaju: naziv atributa, tip podatka, obaveznost i pripadnost atributa odgovarajućem entitetu, lista atributa i *While* ciklus u kojem se parsira deo XML zapisa modela podataka koji sadrži kod, tj. naziv atributa. Blok naredbi ima istu logiku kao i deo za određivanje skupa entiteta, sa tom razlikom da su traženi drugačiji XML elementi za formiranje podstringova, sve do onog najmanjeg <a:Code>ID_BROJ</a:Code>, koji sadrži sam naziv atributa.

Listing 6.8:

```
Dim vEntity As String = ""
Dim vEntityIDAtributCode As String = ""
Dim vEntityIDAtributName As String = ""
Dim vEntityAtribut As String = ""
Dim pozicijaPocetkaAtr As Integer = 0
Dim pozicijaKrajaAtr As Integer = 0
Dim pozicijaPocetkaRef As Integer = 0
...
listaAtributa = New ArrayList
vModel = vCeoModel
...
While Not vEntity = ""
   pozicijaPocetkaAtr = InStr(vEntity, "<c:DataItem>")
   If pozicijaPocetkaAtr = 0 Then GoTo Izlaz
   pozicijaKrajaAtr = InStr(vEntity, "</c:DataItem>")
   vEntityAtribut = Mid(vEntity, pozicijaPocetkaAtr, pozicijaKrajaAtr -
   pozicijaPocetkaAtr)
   pozicijaPocetkaRef = InStr(vEntityAtribut, "Ref=")
   vEntityAtribut = Mid(vEntityAtribut, pozicijaPocetkaRef,
   pozicijaPocetkaRef)
   vEntityAtribut = OdrediNazivAtributa(vEntityAtribut)
   txtAtributes.Text = txtAtributes.Text + vEntityAtribut + "; "
   listaAtributa.Add(vEntityAtribut)
   vEntity = Mid(vEntity, pozicijaKrajaAtr + 1, Len(vEntity) -
   pozicijaKrajaAtr)
End While
Izlaz:...
```

Relacije između entiteta su određene na osnovu XML elemenata koji su prikazani u listingu 6.9. U ovom slučaju od interesa su bili XML elementi: `<o:Relationship>` i `<a:Code>`, na osnovu kojih je formirana lista relacija. Formiran je član skupa *R*={organizuje} i Prolog činjenica rel(organizuje).

Listing 6.9:

```
<c:Relationships>
<o:Relationship Id="o8">
<a:Name>organizuje</a:Name>
<a:Code>ORGANIZUJE</a:Code>
<a:Entity1ToEntity2RoleCardinality>1,1</a:Entity1ToEntity2RoleCardinality>
<a:Entity2ToEntity1RoleCardinality>1,n</a:Entity2ToEntity1RoleCardinality>
<c:Object1>
<o:Entity Ref="o47"/>
</c:Object1>
<c:Object2>
<o:Entity Ref="o46"/>
</c:Object2>
</o:Relationship>
...
<o:Entity Id="o47">
<a:Name>organizator</a:Name>
<a:Code>ORGANIZATOR</a:Code>
...
<o:Entity Id="o46">
<a:Name>konferencija</a:Name>
<a:Code>KONFERENCIJA</a:Code>
```

Osim određivanja liste sa nazivima relacija, pomoću elemenata `<c:Object1>` i `<c:Object2>`, utvrđeni su entiteti koji su u relaciji. Za svaki objekat je određena referenca, tj. jedinstveni broj koji ga identifikuje u XML-u. Na osnovu ovog broja, pronađeni su odgovarajući elementi `<o:Entity>` sa istom vrednošću osobine ID i u

okviru kojih je izdvojen naziv, tj. kod entiteta u podstringu: `<a:Code> ... </a:Code>`. U formi Prolog rečenice to je niz činjenica:

P(organizator, organizuje).
P(organizuje, konferencija).
P(mincard1, organizuje).
P(maxcardm, organizuje).
P(organizuje, mincard1).
P(organizuje, maxcard1).

Kardinalitet poveznika je određen na osnovu sledeća dva XML elementa:

```
<a:Entity1ToEntity2RoleCardinality>1,1
   </a:Entity1ToEntity2RoleCardinality>
<a:Entity2ToEntity1RoleCardinality>1,n
   </a:Entity2ToEntity1RoleCardinality>
```

Implementacija ostalih elemenata ER modela podataka, kao što su identifikacioni atributi entiteta, ograničenja, veze jak-slab tip entiteta (zavisna relacija sa osobinom *Dependent*) i IS_A hijerarhija, se ostvaruju na identičan način kao što je i opisano već za enitete, atribute i poveznike, uz napomenu da su od interesa, u parsiranju stringa *vCeoModel,* bili drugi XML elementi. Za identifikacioni atribut entiteta je to `<c:PrimaryIdentifier>` element, za ograničenja kod obaveznih vrednosti atributa `<a:Mandatory>` element u okviru strukture `<c:DataItem>`, za IS_A hijerarhiju `<o:Inheritance></o:Inheritance>` i na kraju za vezu jak-slab tip entiteta je to `<a:DependentRole>` oznaka, u okviru `<o:Relationship></o:Relationship>` XML strukture.

Ispis formalizovanog modela podataka u kontrolama na formi je ostvaren tako što se pristupa svakom elementu lista entiteta, atributa i relacija, pomoću kolekcije *Item*, uz konverziju vrednosti u *String* tip podatka:

```
listaEntiteta.Item(i).ToString.ToLower
```

Listing 6.10:

```
fileClauses = New System.IO.StreamWriter("test.pro")
Dim i As Integer = 0
Dim brSlogova As Integer = 0
brSlogova = listaEntiteta.Count - 1

For i = 0 To brSlogova
richtxtClauses.Text = richtxtClauses.Text + "ent(" +
listaEntiteta.Item(i).ToString.ToLower + ")." + Chr(13)
fileClauses.WriteLine("ent(" + listaEntiteta.Item(i).ToString.ToLower + ").")
Next i
brSlogova = listaAtributa.Count - 1
For i = 0 To brSlogova
richtxtClauses.Text = richtxtClauses.Text + "atr(" +
listaAtributa.Item(i).ToString.ToLower + ")." + Chr(13)
fileClauses.WriteLine("atr(" + listaAtributa.Item(i).ToString.ToLower + ").")
Next i
brSlogova = listaRelacija.Count - 1
For i = 0 To brSlogova
richtxtClauses.Text = richtxtClauses.Text + "rel(" +
listaRelacija.Item(i).ToString.ToLower + ")." + Chr(13)
fileClauses.WriteLine("atr(" + listaRelacija.Item(i).ToString.ToLower + ").")
Next i
```

6.4.3. Implementacija softverske funkcije za učitavanje ontologije

Rezultat kreiranja ontologije u ontološkom editoru je datoteka sa ekstenzijom .OWL, koja je takođe XML dokument koji sadrži oznake elemenata ontologije. Učitavanje ove datoteke u odgovarajuće promenljive zahtevalo je uključivanje biblioteke klasa za rad sa ulazno-izlaznim uređajima u Visual Basic-u. Zatim je na formu postavljena kontrola »*OpenFileDialog*« koja se koristi za selektovanje datoteka, u prozoru koji se tom prilikom otvara. Kao i kod učitavanja modela podataka, instancira se globalni objekat na nivou aplikacije tipa klase »*StreamReader*« koja omogućuje čitanje tekstualne XML datoteke metodom »*ReadToEnd*« i njeno dodeljivanje promenljivoj *vCelaOntologija,* stringovnog tipa. Blok naredbi koji realizuje učitavanje ontologije u promenljivu *vCelaOntologija* je prikazan u listingu 6.11.

Listing 6.11:

```
Imports System.IO
Imports System.Xml

OpenFileOntology.ShowDialog()
filePath = OpenFileOntology.FileName
fileOntologija = New System.IO.StreamReader(filePath)
vCelaOntologija = fileOntologija.ReadToEnd
fileOntologija.Close()
richtxtOntology.Text = vCelaOntologija.ToString
XMLDoc.Load(filePath)
```

6.4.4. Implementacija softverske funkcije za kreiranje ontoloških klauzula

U OWL datoteci koju generiše ontološki editor, za svaku klasu, formirani su odgovarajući XML elementi. Objekti ontologije i grupe međusobno povezanih elemenata koji su analizirani DMV programom:

- Za klase: `<Declaration>` i `<Class>`.
- Za osobine objekata: `<Declaration>` i `<ObjectProperty>`.
- Za osobine podataka: `<Declaration>` i `<DataProperty>`.
- Za objekte: `<Declaration>` i `<NamedIndividual>`.
- Za iste osobine objekata: `<EquivalentObjectProperties>` i `<ObjectProperty>`.
- Za podklase: **<SubClassOf>** i **<Class>**.
- Za tipove podataka: `<DataPropertyRange>`, `<DataProperty>` i `<Datatype>`.
- Za veze objekata i klasa: `<ClassAssertion>`, `<Class>` i `<NamedIndividual>`.
- Za veze objekata i tipova podataka: `<DataPropertyAssertion>`, `<DataProperty>`, `<NamedIndividual >` i `<Literal datatype>`.
- Za međusobne veze objekata: `<ObjectPropertyAssertion>`, `<ObjectProperty>` i `<NamedIndividual >`.
- Za ograničenja osobina objekata: `<ObjectPropertyRange>`, `<ObjectProperty>`, `<ObjectAllValuesFrom>`, `<ObjectMinCardinality>` i `<ObjectMaxCardinality>`.

U slučaju ontoloških klasa traženi elementi su <Declaration> i <Class>, kao što je prikazano u listingu 6.12. Parsirana je OWL datoteka, tj. promenljiva *vCelaOntologija* i izdvojen je podstring koji sadrži celokupan element za opis klase i na osnovu toga je formiran predikat R(Konferencija, type, Class). Prvi argument je naziv klase, drugi je vrsta elementa ontologije, a treći je službena reč za naziv klase. Predikat R se, u Visual Basic-u, naredbom koja vrši spajanje stringova transformiše u Prolog činjenicu: rdf(konferencija, type, class).

Listing 6.12:

```
<Declaration>
       <Class IRI="#Konferencija"/>
</Declaration>
```

Implementacija ove sosftverske funkcije realizovana je, kao i u slučaju formalizacije modela podataka, pomoću funkcija: INSTR(string, podstring) za određivanje pozicija početka podstringa u okviru nekog stringa i MID(string, lokacijapočetka, dužinaodsecanja) za formiranje podstringova. Traženi elementi su <Declaration>, <Class> i završni element </Declaration>.

Listing 6.13:

```
Dim vEntity As String = ""
Dim vEntityName As String = ""
Dim pozicijaPocetkaEntiteta As Integer = 0
Dim pozicijaKrajaEntiteta As Integer = 0
Dim pozicijaPocetkaNaziva As Integer = 0
Dim pozicijaKrajaNaziva As Integer = 0

vModel = vCelaOntologija

While Not vModel = ""
   pozicijaPocetkaEntiteta = InStr(vModel, "<Class IRI=")
   pozicijaKrajaEntiteta = InStr(vModel, "</Declaration>")
   vEntity = Mid(vModel, pozicijaPocetkaEntiteta, pozicijaKrajaEntiteta –
   pozicijaPocetkaEntiteta + 14)
   pozicijaPocetkaNaziva = InStr(vEntity, "#")
   pozicijaKrajaNaziva = InStr(vEntity, "/>")
   vEntityName = Mid(vEntity, pozicijaPocetkaNaziva + 1,
   pozicijaKrajaNaziva – pozicijaPocetkaNaziva - 2)
   richtxtClauses.Text = richtxtClauses.Text + "rdf(" +
   vEntityName.ToString.ToLower + ", type, class)." + Chr(13)
   vModel = Mid(vModel, pozicijaKrajaEntiteta + 20, Len(vModel) –
   pozicijaKrajaEntiteta)
End While
```

Naziv ontološke klase je određen na osnovu <Class IRI="#Konferencija"/> elementa. Deklarisane su promenljive tipa *String* i *Integer* za formiranje podstringova modela na osnovu XML oznaka elemenata do samog naziva klase, kao i u slučaju entiteta modela podataka. Kada se izdvoji naziv klase, isti se memoriše u promenljivoj *vEntityName*, čija se vrednost dodaje *Text* svojstvu odgovarajućeg polja za prikaz naziva svih klasa u korisničkom interfejsu. Promenljiva *vModel,* koja sadrži XML kod cele ontologije, pomoću MID funkcije, se transformiše u podstring cele ontologije iz koje je izostavljen element već obrađene, tj. identifikovane klase. Pronalazi se sledeći par <Declaration> i <Class> elemenata i ceo postupak se ponavlja *While* ciklusom dok se ne isparsiraju svi elementi ovog tipa.

Parsiranjem stringa XML elementa <SubClassOf> se definiše taksonomija klasa, kao u primeru „Međunarodna konferencija" koji je prikazan u listingu 6.14:

Listing 6.14:

```
<SubClassOf>
       <Class IRI="#Medjunarodna"/>
       <Class IRI="#Konferencija"/>
</SubClassOf>
```

Formira se predikat R(Medjunarodna, subClassOf, Konferencija), koji u Prologu ima oblik: rdf(medjunarodna, subclass, konferencija).

Sledeći primer dela implementacije softverske podrške je primer izdvajanja osobina objekata (*ObjectProperty* svojstvo) ontologije iz OWL elementa (Listing 6.15).

Listing 6.15:

```
<Declaration>
       <ObjectProperty IRI="#Odrzava_se"/>
</Declaration>
```

Formira se predikat R(odrzava_se, type, ObjectProperty), koji u Prologu ima oblik: rdf(odrzava_se, type, objectproperty). Izdvajanje osobina podataka ontologije vrši se iz elemenata <DataPropertyRange>, <DataProperty> i <Datatype>. Oznaka <DataProperty> određuje naziv osobine podataka, a <Datatype> tip podatka.

Listing 6.16:

```
<DataPropertyRange>
      <DataProperty IRI="#adresa"/>
      <Datatype abbreviatedIRI="xsd:string"/>
</DataPropertyRange>
```

Za osobine podataka ontologije formiraju se predikat: R(adresa, range, string) i Prolog činjenica: rdf(adresa, range, string). Opšti oblika tripleta je: rdf(naziv, range, tip_podatka).

Listing 6.17 prikazuje elemente za povezivanje individua, tj. konkretnih objekata pomoću osobina objekata. Rezultujući RDF triplet, dobijen parsiranjem elementa <ObjectPropertyAssertion> je: R(Odrzava_se, AIIT, Hotel_Vojvodina), što se u Prologu predstavlja činjenicom: rdf(odrzava_se, aiit, hotel_vojvodina).

Listing 6.17:

```
<ObjectPropertyAssertion>
        <ObjectProperty IRI="#Odrzava_se"/>
        <NamedIndividual IRI="#AIIT"/>
        <NamedIndividual IRI="#Hotel_Vojvodina"/>
</ObjectPropertyAssertion>
```

Kada su u pitanju kardinaliteti kao ograničenja osobina objekata klase, u listingu 6.18, se može primetiti da su elementi skupa P u modelu određeni na osnovu <ObjectPropertyRange> XML elementa. Za svaku relaciju u konceptualnom modelu podataka kardinalitet se određuje, prema [Mogin i sar. 1996], kao binarna relacija R između skupova dva tipa entiteta E_1 i E_2, koja se predstavlja putem dva preslikavanja: R_1: $E_1 \rightarrow P(E_2)$ i R_2: $E_2 \rightarrow P(E_1)$, a kardinalnost relacije (tipa poveznika) R se označava sa $R(E_1(a_1,b_1)$: $(E_2(a_2,b_2))$, gde je a_1,a_2 minimalni (0 ili 1), a b_1,b_2 je maksimalni kardinalitet (1 ili više od 1, tj. N ili M). U ontologiji su definisana četiri ograničenja osobina objekta koja

odgovaraju vrednostima a_1,b_1,a_2,b_2. Element <ObjectAllValuesFrom> predstavlja gornju granicu kardinaliteta koja je veća od 1, <ObjectMaxCardinality cardinality="1"> gornju granicu kardinaliteta sa vrednošću 1, dok su elementi <ObjectMinCardinality cardinality="0"> i <ObjectMinCardinality cardinality="1"> elementi koji ukazuju na vrednosti donje granice kardinaliteta.

Listing 6.18:

```
<ObjectPropertyRange>
        <ObjectProperty IRI="#Izlaze"/>
        <ObjectAllValuesFrom>
            <ObjectProperty IRI="#Izlaze"/>
            <Class IRI="#Ucesnik"/>
        </ObjectAllValuesFrom>
</ObjectPropertyRange>
<ObjectPropertyRange>
        <ObjectProperty IRI="#Izlaze"/>
        <ObjectMinCardinality cardinality="0">
            <ObjectProperty IRI="#Izlaze"/>
            <Class IRI="#Rad"/>
        </ObjectMinCardinality>
</ObjectPropertyRange>
<ObjectPropertyRange>
        <ObjectProperty IRI="#Izlaze"/>
        <ObjectMinCardinality cardinality="0">
            <ObjectProperty IRI="#Izlaze"/>
            <Class IRI="#Ucesnik"/>
        </ObjectMinCardinality>
</ObjectPropertyRange>
<ObjectPropertyRange>
        <ObjectProperty IRI="#Izlaze"/>
        <ObjectMaxCardinality cardinality="1">
            <ObjectProperty IRI="#Izlaze"/>
            <Class IRI="#Rad"/>
        </ObjectMaxCardinality>
</ObjectPropertyRange>
```

Parsiranjem XML elemenata koja se odnose na ograničenja kardinaliteta, formirana su, za svaku osobinu objekta, četiri rdf predikata, kao što se može videti u primeru:

rdf(rad,izlaze,mincard0).
rdf(rad,izlaze,maxcard1).
rdf(ucesnik,izlaze,mincard0).
rdf(ucesnik,izlaze,maxcardm).

Prvi argument predstavlja naziv klase, drugi je osobina objekta kao instance te klase, a treći je vrednost ograničenja kardinaliteta.

Svi ostali elementi ontologije izdvojeni su procedurama koje imaju istu aplikacionu logiku kao i prethodno opisane, a odnose se na ontološke klase, podklase, osobine objekata i osobine podataka.

6.4.5. Kreiranje pravila zaključivanja

Pravila zaključivanja na osnovu kojih se vrši provera semantičke korektnosti modela podataka su definisana i detaljno opisana u odeljku 5.4. disertacije. Formirana je tekstualna datotetka u koju su upisana pravila zaključivanja za semantičku proveru modela (listing 5.2). Metodom *StreamReader* se, iz tekstualne datoteke, čitaju pravila

zaključivanja i upisuju u promenljivu *vRules*, čija se vrednost, nakon konverzije tipa podatka, dodeljuje odgovarajućoj kontroli tipa *RichTextBox* u okviru forme korisničkog interfejsa programa. Nakon spiska Prolog rečenica koje sadrže formalizovani model podataka i mapiranu ontologiju, dodaje se i ovaj spisak pravila zključivanja.

Listing 6.19:

```
OpenFileRules.ShowDialog()
filePath = OpenFileRules.FileName
fileRules = New System.IO.StreamReader(filePath)
vRules = fileRules.ReadToEnd
fileRules.Close()
richtxtRules.Text = vRules.ToString
richtxtClauses.Text = richtxtClauses.Text + vRules
```

6.4.6. Nova analiza

Slučaj korišćenja softvera „Nova analiza“ poništava vrednosti promenljivih koje sadrže stringove XML zapisa modela podataka: `vModel` i `vCeoModel`, ontologije: `vOntologija` i `vCelaOntologija`, kao i pravila zaključivanja `vRules`, kako bi se moglo izvršiti učitavanje novog modela podataka i ontologije.

Vrednosti kontrola u korisničkom interfejsu koje prikazuju ceo XML modela podataka `txtModel`, ontologije `richtxtOntology`, Prolog klauzula `richtxtClauses` i pravila zaključivanja `richtxtRules` , takođe, dobijaju vrednost praznog stringa (listing 6.20).

Listing 6.20:

```
txtModel.Text = ""
richtxtOntology.Text = ""
richtxtRules.Text = ""
richtxtClauses.Text = ""
vModel = ""
vCeoModel = ""
vOntologija = ""
vCelaOntologija = ""
vRules = ""
```

6.4.7. Pokretanje Prologa

AMZI! Prolog sistem je postavljen u direktorijum (*folder*) u kom se nalazi izvršna verzija DMV aplikacije, te je samo iniciran početak procesa za pokretanje editora programa, tj. datoteke „a4ideA.exe“:

```
Process.Start("C:\Data Model Validator\DataModelValidator\Bin\Debug\
            Prolog\a4ideA.exe")
```

Završetkom rada u Prologu, nakon postavljanja svih upita kreiranih na osnovu pravila zaključivanja, ovaj proces se automatski završava, te se aktivira prozor DMV aplikacije, u kom se može uraditi ili nova analiza ili završiti korišćenje programa.

7. EMPIRIJSKO ISTRAŽIVANJE

Empirijsko istraživanje je izvršeno kao eksperimentalno istraživanje sa jednom grupom ispitanika. Sprovedeno je na Univerzitetu u Novom Sadu, na Tehničkom fakultetu „Mihajlo Pupin“, 2012. godine, sa studentima smera Informacione tehnologije osnovnih akademskih studija, na studijskom programu Informacione tehnologije, u okviru modula: inženjerstvo, u poslovnim sistemima i u obrazovanju. Za proveru osnovne hipoteze disertacije i testiranje teoretskog modela izvršena je analiza semantičke korektnosti konceptualnih modela podataka koje su studenti radili na kolokvijumu iz nastavnog predmeta Baze podataka 1, u okviru jedne od četiri grupe koliko je imalo praktičan kolokvijuma na računaru.

7.1. PRIMER „ORGANIZACIJA KONFERENCIJA”

Kreiranje ontologije i modela podataka su pokazani na primeru «Organizacija konferencija». Ovaj primer je izabran za istraživanje zato što su studenti imali najmanje znanja i informacija o domenu koji je bilo potrebno modelovati.

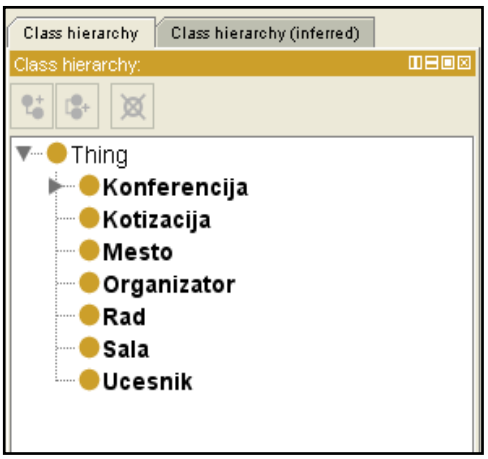
Opis i karakteristike sistema potrebne za konceptualno modelovanje baze podataka su data u sledećem opisu:

„Konferenciju, koja se odlikuje svojim identifikacionim brojem, nazivom, datumima početka i završetka, organizuje firma ili institucija organizator sa sledećim relevantnim osobinama: id organizatora, ime, adresa. Konferencija mora biti nacionalnog ili međunarodnog karaktera. Tada se za međunarodne konferencije upisuje i spisak nastavnika u naučnom odboru iz inostranstva, a za nacionalne se navodi naziv ministarstva koje podržava organizaciju skupa. Jednu konferenciju organizuje samo jedan organizator i taj organizator može učestvovati u organizaciji većeg broja konferencija, a mora najmanje jednu da organizuje kako bi se našao u bazi podataka. Organizator je iz jednog i samo jednog mesta, sa sledećim relevantnim osobinama: ptt, naziv mesta, država. Ne mora u svakom mestu biti organizatora konferencija, ali može biti i više institucija koje organizuju različite konferencije u jednom mestu. Konferencija se uvek održava u nekoj sali, jednoj ili više njih, koja ima svoju jedinstvenu oznaku (id sale), kapacitet i podatak o tome da li postoji ozvučenje ili ne. Tokom vremena, u sali se može održati i više konferencija, a najmanje mora jedna. Sala se, takođe, nalazi u nekom mestu, jednom i jedinom. U mestu, sa druge strane, ne mora postojati nijedna sala, ali ih može biti više. Učesnik konferencije ima osobine poput identifikacionog broja, imena, prezimena, adrese, naziva ustanove, broja telefona, broja mobilnog telefona, e-mail adrese i web adrese. Učesnik može napisati rad (id rada, naziv, oblast, značaj, obim) – ali ne mora, a može ih napisati i više. Rad piše jedan ili više učesnika konferencije. Da bi učesnik izlagao rad u nekoj sali određenog datuma i u određeno vreme, mora biti ispunjen uslov da je napisao taj rad. Učesnik, napisan rad ne mora ni izlagati u nekoj sali, ali ga može izlagati samo u jednoj sali. Obrnuto, u nekoj sali se nijedan učesnik sa napisanim radom ne mora pojaviti, a može ih biti više. Konačno, evidentira se da li je učesnik uplatio kotizaciju (id kotizacije, tip i iznos kotizacije). Kotizacija ne mora biti

uplaćena, ali ako jeste, uplaćena je samo jednom. Kotizacija može biti uplaćena od strane najmanje jednog, ali i od strane više učesnika koji su koautori na radu. Uslov da učesnik uopšte prisustvuje konferenciji jeste taj da je uplatio jednu kotizaciju i ne mora učestvovati čak i ako je uplatio kotizaciju. Sa druge strane, kao što je rečeno, da bi učesnik prisustvovao konferenciji mora uplatiti tačno jednu kotizaciju."

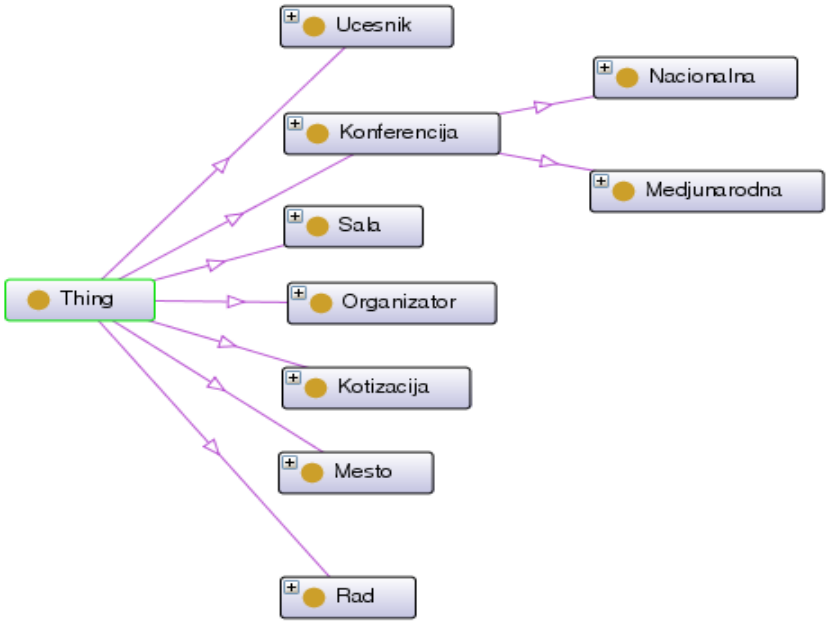
7.2. ONTOLOGIJA ZA PRIMER «ORGANIZACIJA KONFERENCIJA»

Prema metodologiji za kreiranje ontologija prikazanim na slici 5.9., u prvom koraku su definisane klase: «Konferencija», «Kotizacija», «Mesto», «Organizator», «Rad», «Sala» i «Ucesnik». Slika 7.1. prikazuje prozor ontološkog alata u kojem se definišu klase.



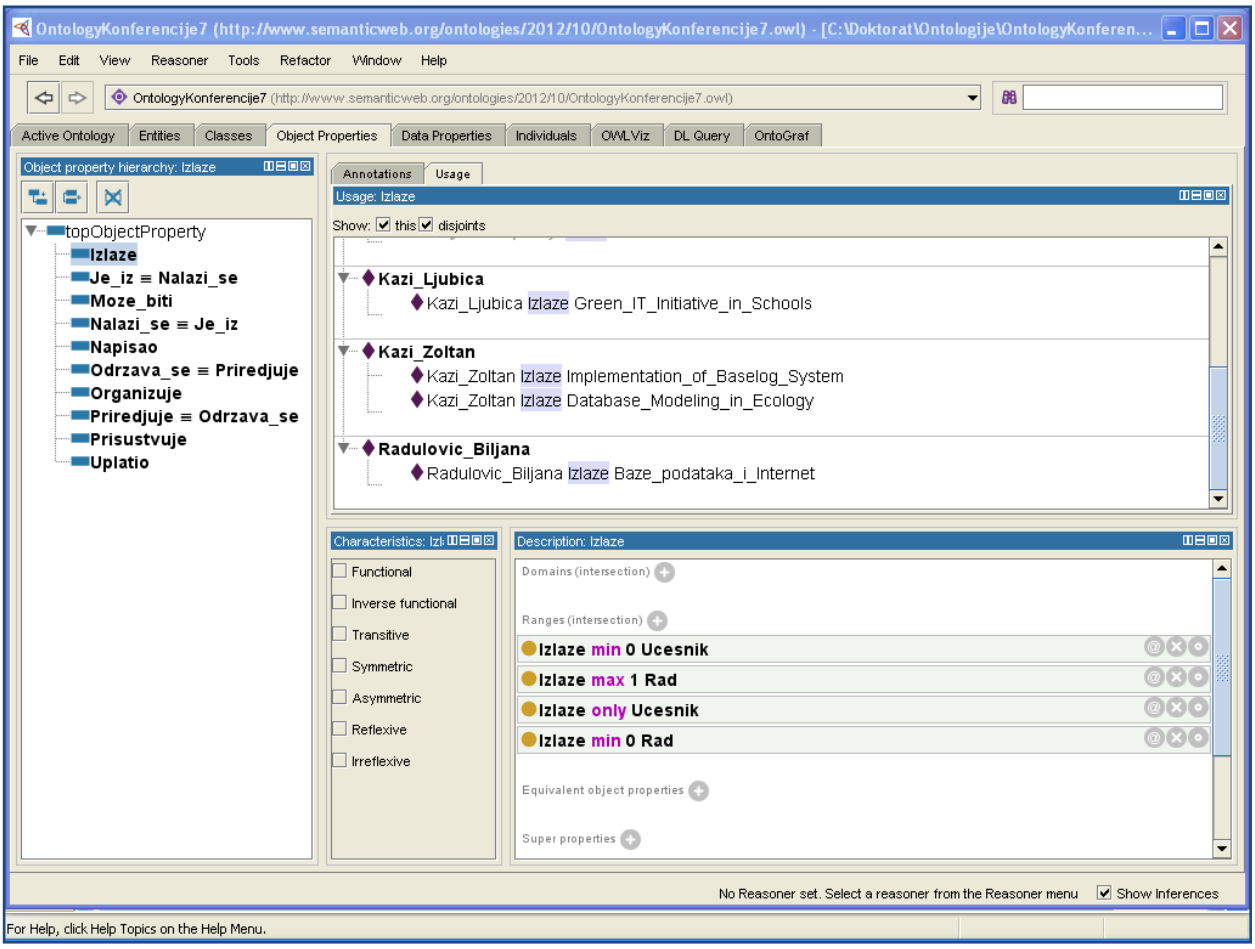
Slika 7.1. – Klase ontologije

Zatim su, u drugom koraku, organizovane klase u taksonomije tako što je za klasu «Konferencija» izvršena specijalizacija na dve podklase: «Medjunarodna» i «Nacionalna» (Slika 7.2.).

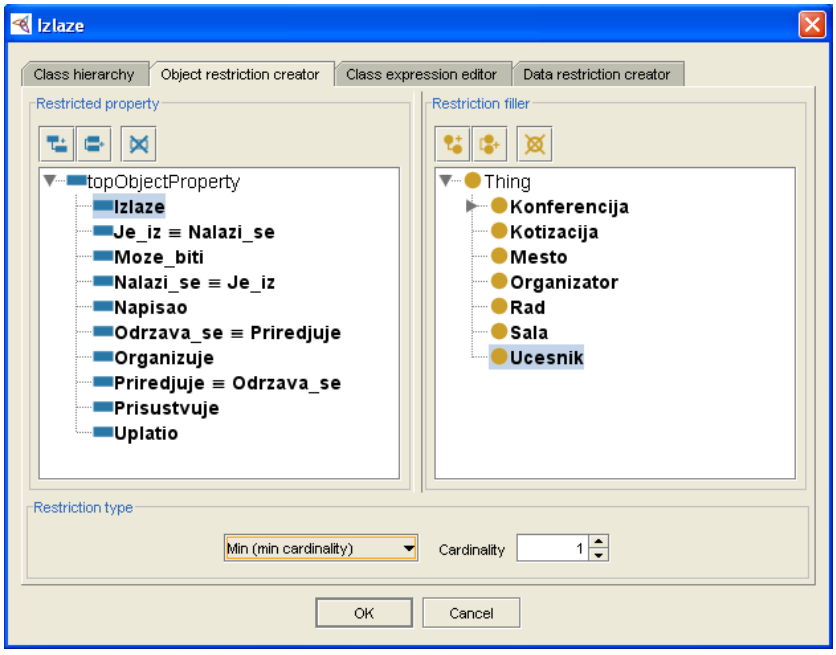


Slika 7.2. – Taksonomija klasa

U trećem koraku su definisane relacije između klasa. Prvo su određene sledeće osobine objekata: «Izlaze», «Je iz», «Moze biti», «Nalazi se», «Napisao», «Odrzava se», «Organizuje», «Priredjuje», «Prisustvuje» i «Uplatio».

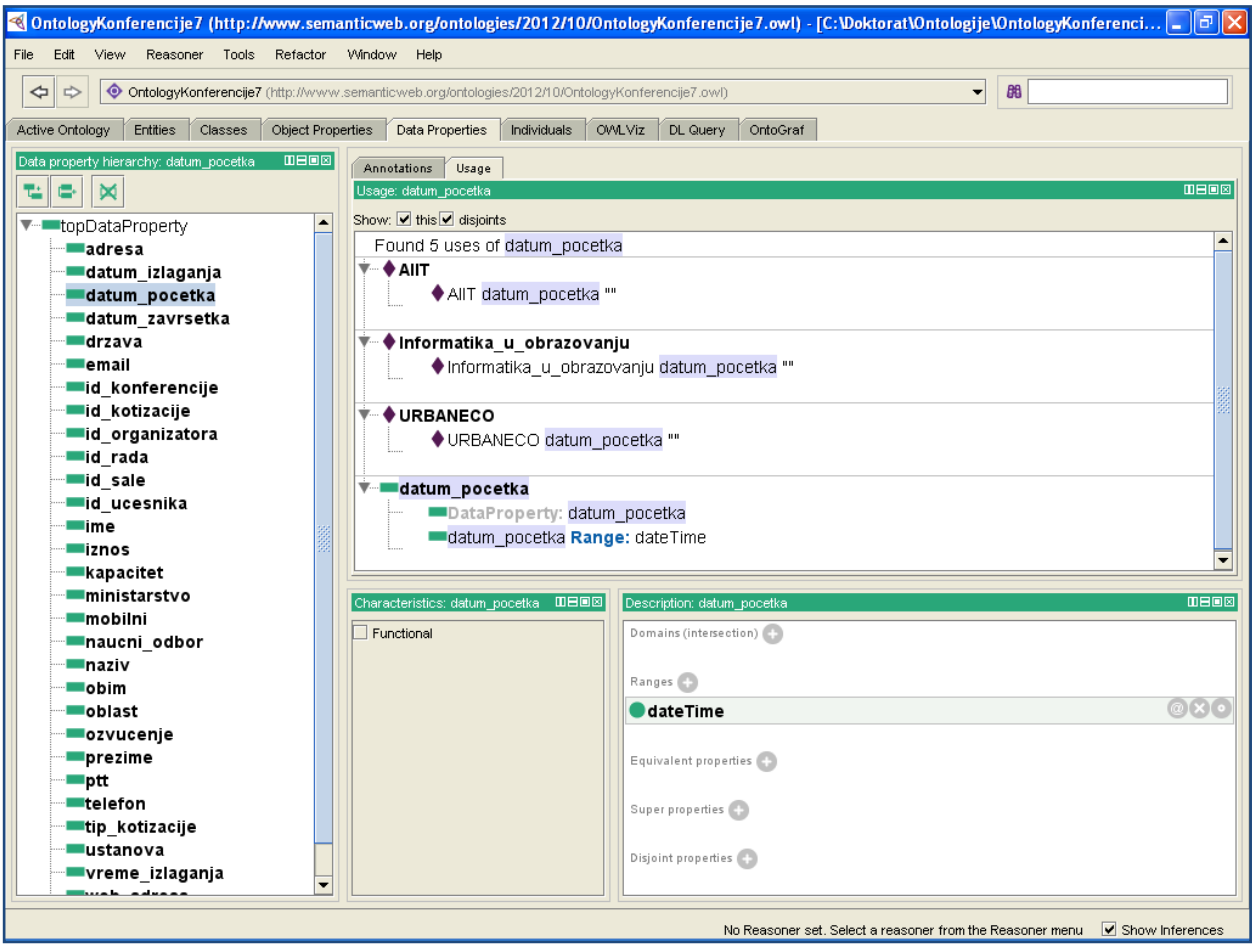


Slika 7.3. – Osobine objekata



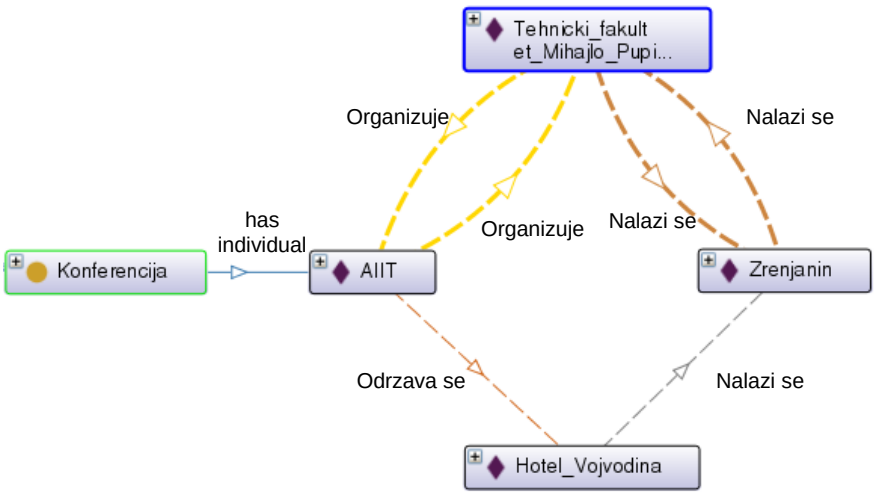
Slika 7.4. - Grafički prikaz hijerarhije klasa ontologije iz OWL editora

U četvrtom koraku su definisani atributi, tj. osobine podataka: «naziv», «adresa», «drzava», «email», «datum izlaganja», «datum pocetka», «datum zavrsetka», «id konferencije», «id kotizacije», «id konferencije», «id organizatora», «id rada», «id ucesnika», «id sale», «prezime», «ime», «telefon», «ptt», «iznos», «kapacitet», «obim», «ozvucenje», «ministarstvo», «mobilni», «naucni odbor», «tip kotizacije», «ustanova», «kapacitet», «oblast», «tip kotizacije», «vreme izlaganja», «web adresa» i «znacaj».



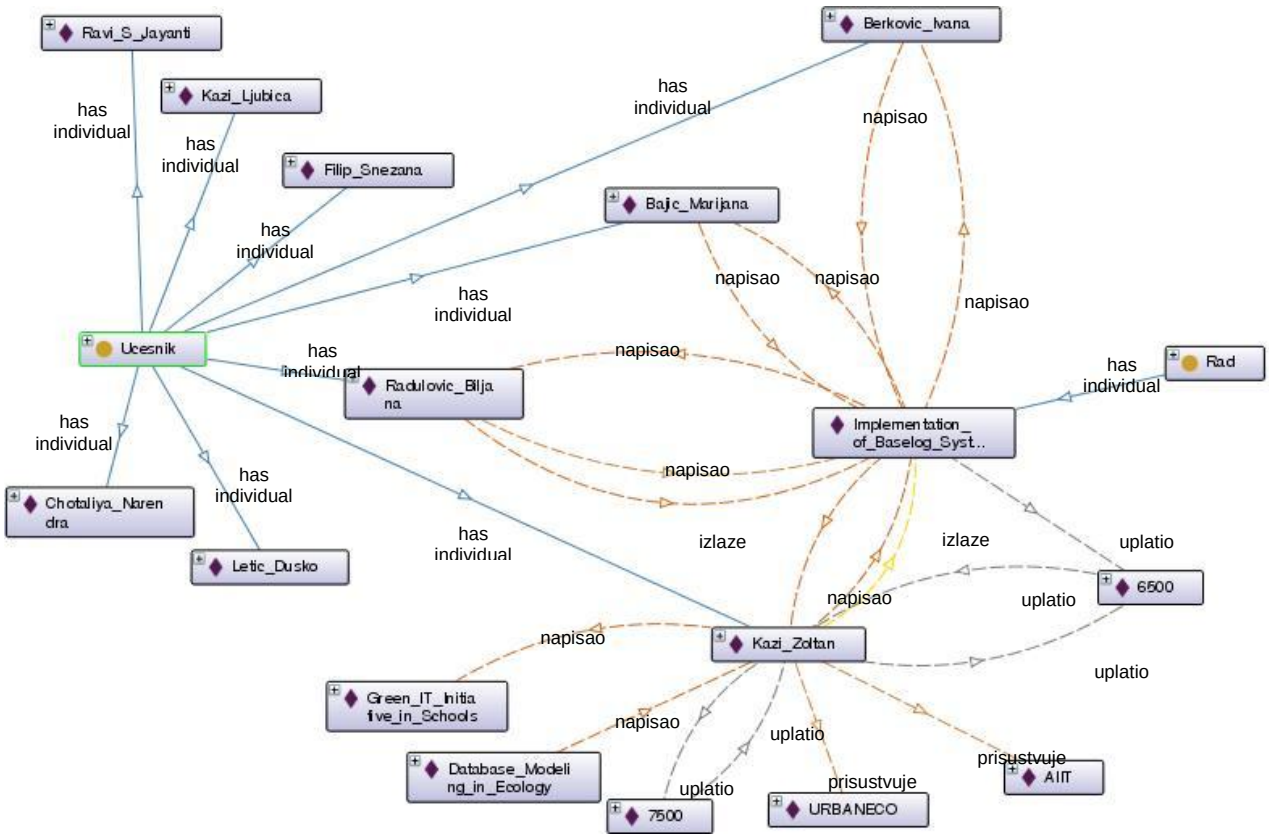
Slika 7.5. – Atributi ontologije kao osobine podataka

U petom koraku su određene instance klasa. Kreirani su objekti iz domena koji je modelovan (Slika 7.6.).



Slika 7.6. - Grafički prikaz instance klase «Konferencija» u OntoGraf editoru

Objekti, kao instance klasa, nisu sadržani u tekstu zadatka koji opisuje deo sistema koji se konceptualno modeluje, pa su vrednosti određene na osnovu zbornika radova I međunardone konferencije „Applied Interent and Information Technologies“, održane u Zrenjaninu, u organizaciji Tehničkog fakulteta „Mihajlo Pupin“. Slika 7.7. prikazuje samo instance klasa «Ucesnik» i «Rad» u OntoGraf editoru, s obzirom da je ceo graf ove jednostavne ontologije prevelik da bi se prikazao u celosti.



Slika 7.7. - Grafički prikaz dela instanci klasa «Ucesnik» i «Rad» u OntoGraf editoru

7.2.1. Mapiranje ontologije

Primer ontologije «Organizacija konferencija», koji je kreiran u *Protégé* alatu, snimljen je kao OWL/RDF ontologija. Mapiranje elemenata ontologije na predikatski račun uz pomoć SWI Prolog okruženja, vrši se u nekoliko koraka. Prvo se moraju učitati biblioteke za rad sa RDF ontologijama:

```
?- use_module(library(semweb/rdf_db)).
```

Zatim su učitavaju biblioteke za rad sa RDF šemama, kao i biblioteka za kreiranje i rad sa imenovanim prostora u RDF datotekama:

```
?- use_module(library(semweb/rdfs)).
?- use_module(library(semweb/rdf_portray)).
```

Nakon učitanih biblioteka za rad sa ontologijama, postavlja se upit za učitavanje RDF ontologije u Prolog i parsiranje RDF trojki, tj. triplet-a:

```
?-rdf_load('konferencije.owl'), rdf(S, P, O).
% Parsed "konferencije.owl" in 0.05 sec; 379 triples
S = 'http://www.semanticweb.org/ontologies/2012/10/OntologyKonferencije7.owl',
P = rdf:type,
O = owl:'Ontology' ;
S = 'http://www.semanticweb.org/ontologies/2012/10/OntologyKonferencije7.owl#
izlaze',
P = rdf:type,
O = owl:'ObjectProperty' ;
S = 'http://www.semanticweb.org/ontologies/2012/10/OntologyKonferencije7.owl#
je_iz',
P = rdf:type,
O = owl:'ObjectProperty' ;
S = 'http://www.semanticweb.org/ontologies/2012/10/OntologyKonferencije7.owl#
je_iz',
P = owl:equivalentProperty,
O = 'http://www.semanticweb.org/ontologies/2012/10/OntologyKonferencije7.owl#
nalazi_se' ;
```

Kao što je u teorijskom istraživanju istaknuto elementi RDF triplet-a su URI nizovi karaktera za identikovanje abstraktnog ili fizičkog resursa na Web-u. Pošto su komponente URI-ja prilično dugački stringovi, formirana su tri imenovana prostora kako bi se povećala čitljivost i olakšao rad sa predikatima koje formira SWI Prolog:

```
?- rdf_register_ns(subj, 'http://www.semanticweb.org/ontologies/2012/10/
OntologyKonferencije7.owl#').
?- rdf_register_ns(pred, 'http://www.w3.org/1999/02/22-rdf-syntax-ns#').
?- rdf_register_ns(obj, 'http://www.w3.org/2002/07/owl#').
```

Postavljanje upita za parsiranje RDF ontologije rezultuje, u primeru «Organizacija konferencija», formiranjem 379 tripleta. Prikazani su samo karakteristični RDF tripleti, koji su poslužili kao osnova za prevođenje ontologije i njenih elemenata iz XML oblika u formule predikatskog računa.

Mapiranje OWL klase na predikatski račun prvog reda:

```
S = subj:'Konferencija',
P = rdf:type,
O = owl:'Class' ;
```

R (S, P, O) ⇔ R (`Konferencija, type, Class`)

```
S = subj:'Medjunarodna',
P = rdf:type,
O = owl:'Class' ;
```

R (S, P, O) ⇔ R (`Medjunarodna, type, Class`)

Mapiranje OWL podklase na predikatski račun:

```
S = subj:'Medjunarodna',
P = rdfs:subClassOf,
O = subj:'Konferencija' ;
```

R (S, P, O) ⇔ R (`Medjunarodna, subClassOf, Konferencija`)

Mapiranje OWL relacija na predikatski račun:

```
S = subj:odrzava_se,
P = rdf:type,
O = owl:'ObjectProperty' ;
```

R (S, P, O) ⇔ R (odrzava_se, type, ObjectProperty)

Definisanje ekvivalentnih OWL relacija u formi predikatskog računa:

```
S = subj:odrzava_se,
P = owl:equivalentProperty,
O = subj:priredjuje ;
```

R (S, P, O) ⇔ R (odrzava_se, equivalentProperty, priredjuje)

Mapiranje OWL atributa na predikatski račun:

```
S = subj:datum_izlaganja,
P = rdf:type,
O = owl:'DatatypeProperty' ;
```

R (S, P, O) ⇔ R (datum_izlaganja, type, DatatypeProperty)

Mapiranje OWL tipova podataka na predikatski račun:

```
S = subj:datum_izlaganja,
P = rdfs:range,
O = xsd:dateTime ;
```

R (S, P, O) ⇔ R (datum_izlaganja, range, dateTime)

Mapiranje OWL instanci klasa na predikatski račun:

```
S = subj:aiit,
P = rdf:type,
O = subj:'NamedIndividual';
```

R (S, P, O) ⇔ R (aiit, type, NamedIndividual)

Mapiranje OWL veza i relacija između različith objekata na predikatski račun:

```
S = subj:aiit,
P = subj:odrzava_se,
O = subj:hotel_vojvodina ;
```

R (S, P, O) ⇔ R (aiit, odrzava_se, hotel_vojvodina)

```
S = subj:aiit,
P = subj:organizuje,
O = subj:tehnicki_fakultet_mihajlo_pupin ;
```

R (S, P, O) ⇔ R (aiit, organizuje, tehnicki_fakultet_mihajlo_pupin)

Dalja transformacija elemenata ontologije u oblik rečenica Prolog jezika (na osnovu 3.7 i 3.9), prikazana samo za karakteristične oblike RDF tripleta:

R (Konferencija, type, Class) ⇔ RDF(konferencija, type, class).

R (Medjunarodna, type, Class) ⇔ RDF(medjunarodna, type, class).

R(Medjunarodna, subClassOf, Konferencija)
⇔ RDF(medjunarodna, subclass, konferencija).

R(odrzava_se, type, ObjectProperty)
⇔ RDF(odrzava_se, type, objectproperty).

R(odrzava_se, equivalentProperty, priredjuje)
⇔ RDF(odrzava_se, equivalentobjectproperties, priredjuje).

R(datum_izlaganja, type, DatatypeProperty)
⇔ RDF(datum_izlaganja, type, dataproperty).

R(datum_izlaganja, range, dateTime)
⇔ RDF(datum_izlaganja, range, datetime).

R(aiit, type, NamedIndividual)
⇔ RDF(aiit, type, namedindividual).

R(aiit, odrzava_se, hotel_vojvodina)
⇔ RDF(aiit, odrzava_se, hotel_vojvodina).

R(aiit, organizuje, tehnicki_fakultet_mihajlo_pupin)
⇔ RDF(aiit, organizuje, tehnicki_fakultet_mihajlo_pupin).

Nakon postupka mapiranja OWL/RDF ontologije na predikatski račun prvog reda i prevođenja literala u oblik Prolog činjenica, formiran je sledeći skup Prolog rečenica:

Klase (*Class*):

```
rdf(konferencija, type, class).
rdf(kotizacija, type, class).
rdf(medjunarodna, type, class).
rdf(mesto, type, class).
rdf(nacionalna, type, class).
rdf(organizator, type, class).
rdf(rad, type, class).
rdf(sala, type, class).
rdf(ucesnik, type, class).
```

Podklase (*Subclass*):

```
rdf(medjunarodna, subclass, konferencija).
rdf(nacionalna, subclass, konferencija).
```

Osobine objekata (*Object Property*):

```
rdf(izlaze, type, objectproperty).
rdf(je_iz, type, objectproperty).
rdf(moze_biti, type, objectproperty).
rdf(nalazi_se, type, objectproperty).
rdf(napisao, type, objectproperty).
rdf(odrzava_se, type, objectproperty).
```

```
rdf(organizuje, type, objectproperty).
rdf(priredjuje, type, objectproperty).
rdf(prisustvuje, type, objectproperty).
rdf(uplatio, type, objectproperty).
```

Ekvivalentne osobine objekata (*Equivalent Object Properties*):

```
rdf(je_iz, equivalentobjectproperties, nalazi_se).
rdf(odrzava_se, equivalentobjectproperties, priredjuje).
```

Atributi (*Data Property*):

```
rdf(adresa, type, dataproperty).
rdf(datum_izlaganja, type, dataproperty).
rdf(datum_pocetka, type, dataproperty).
rdf(datum_zavrsetka, type, dataproperty).
rdf(drzava, type, dataproperty).
rdf(email, type, dataproperty).
rdf(id_konferencije, type, dataproperty).
rdf(id_kotizacije, type, dataproperty).
rdf(id_organizatora, type, dataproperty).
rdf(id_rada, type, dataproperty).
rdf(id_sale, type, dataproperty).
rdf(id_ucesnika, type, dataproperty).
rdf(ime, type, dataproperty).
rdf(iznos, type, dataproperty).
rdf(kapacitet, type, dataproperty).
rdf(ministarstvo, type, dataproperty).
rdf(mobilni, type, dataproperty).
rdf(naucni_odbor, type, dataproperty).
rdf(naziv, type, dataproperty).
rdf(obim, type, dataproperty).
rdf(oblast, type, dataproperty).
rdf(ozvucenje, type, dataproperty).
rdf(prezime, type, dataproperty).
rdf(ptt, type, dataproperty).
rdf(telefon, type, dataproperty).
rdf(tip_kotizacije, type, dataproperty).
rdf(ustanova, type, dataproperty).
rdf(vreme_izlaganja, type, dataproperty).
rdf(web_adresa, type, dataproperty).
rdf(znacaj, type, dataproperty).
```

Objekti kao instance klasa (*Named Individual*):

```
rdf(k1000, type, namedindividual).
rdf(k6500, type, namedindividual).
rdf(k7500, type, namedindividual).
rdf(aiit, type, namedindividual).
rdf(bajic_marijana, type, namedindividual).
rdf(baze_podataka_i_internet, type, namedindividual).
rdf(berkovic_ivana, type, namedindividual).
rdf(chotaliya_narendra, type, namedindividual).
rdf(database_modeling_in_ecology, type, namedindividual).
rdf(ecka, type, namedindividual).
rdf(filip_snezana, type, namedindividual).
rdf(green_it_initiative_in_schools, type, namedindividual).
rdf(hotel_vojvodina, type, namedindividual).
rdf(implementation_of_baselog_system, type, namedindividual).
rdf(informatika_u_obrazovanju, type, namedindividual).
rdf(kazi_ljubica, type, namedindividual).
```

```
rdf(kazi_zoltan, type, namedindividual).
rdf(letic_dusko, type, namedindividual).
rdf(lovacki_dvorac, type, namedindividual).
rdf(radulovic_biljana, type, namedindividual).
rdf(ravi_s_jayanti, type, namedindividual).
rdf(rekurzivna_pravila_i_fraktali, type, namedindividual).
rdf(sala_skupstine_opstine, type, namedindividual).
rdf(tehnicki_fakultet_mihajlo_pupin, type, namedindividual).
rdf(urbaneco, type, namedindividual).
rdf(zrenjanin, type, namedindividual).
```

Tipovi podataka (*Data Range*):

```
rdf(adresa, range, string).
rdf(datum_izlaganja, range, datetime).
rdf(datum_pocetka, range, datetime).
rdf(datum_zavrsetka, range, datetime).
rdf(drzava, range, string).
rdf(email, range, string).
rdf(id_konferencije, range, positiveinteger).
rdf(id_kotizacije, range, positiveinteger).
rdf(id_organizatora, range, positiveinteger).
rdf(id_rada, range, positiveinteger).
rdf(id_sale, range, positiveinteger).
rdf(id_ucesnika, range, positiveinteger).
rdf(ime, range, string).
rdf(iznos, range, decimal).
rdf(kapacitet, range, short).
rdf(ministarstvo, range, string).
rdf(mobilni, range, string).
rdf(naucni_odbor, range, string).
rdf(naziv, range, string).
rdf(obim, range, byte).
rdf(oblast, range, string).
rdf(ozvucenje, range, boolean).
rdf(prezime, range, string).
rdf(ptt, range, long).
rdf(telefon, range, string).
rdf(tip_kotizacije, range, string).
rdf(ustanova, range, string).
rdf(vreme_izlaganja, range, string).
rdf(web_adresa, range, string).
rdf(znacaj, range, string).
```

Veze objekata i klasa (*Class Assertion*):

```
rdf(k1000, classassertion, kotizacija).
rdf(k6500, classassertion, kotizacija).
rdf(k7500, classassertion, kotizacija).
rdf(aiit, classassertion, konferencija).
rdf(aiit, classassertion, medjunarodna).
rdf(bajic_marijana, classassertion, ucesnik).
rdf(baze_podataka_i_internet, classassertion, rad).
rdf(berkovic_ivana, classassertion, ucesnik).
rdf(chotaliya_narendra, classassertion, ucesnik).
rdf(database_modeling_in_ecology, classassertion, rad).
rdf(ecka, classassertion, mesto).
rdf(filip_snezana, classassertion, ucesnik).
rdf(green_it_initiative_in_schools, classassertion, rad).
rdf(hotel_vojvodina, classassertion, sala).
rdf(implementation_of_baselog_system, classassertion, rad).
rdf(informatika_u_obrazovanju, classassertion, konferencija).
```

```
rdf(informatika_u_obrazovanju, classassertion, nacionalna).
rdf(kazi_ljubica, classassertion, ucesnik).
rdf(kazi_zoltan, classassertion, ucesnik).
rdf(letic_dusko, classassertion, ucesnik).
rdf(lovacki_dvorac, classassertion, sala).
rdf(radulovic_biljana, classassertion, ucesnik).
rdf(ravi_s_jayanti, classassertion, ucesnik).
rdf(rekurzivna_pravila_i_fraktali, classassertion, rad).
rdf(sala_skupstine_opstine, classassertion, sala).
rdf(tehnicki_fakultet_mihajlo_pupin, classassertion, organizator).
rdf(urbaneco, classassertion, konferencija).
rdf(urbaneco, classassertion, medjunarodna).
rdf(zrenjanin, classassertion, mesto).
```

Veze između atributa i tipova podataka (*Data Property Assertion*):

```
rdf(id_kotizacije, datapropertyassertion, k1000).
rdf(iznos, datapropertyassertion, k1000).
rdf(tip_kotizacije, datapropertyassertion, k1000).
rdf(id_kotizacije, datapropertyassertion, k6500).
rdf(iznos, datapropertyassertion, k6500).
rdf(tip_kotizacije, datapropertyassertion, k6500).
rdf(id_kotizacije, datapropertyassertion, k7500).
rdf(iznos, datapropertyassertion, k7500).
rdf(tip_kotizacije, datapropertyassertion, k7500).
rdf(datum_pocetka, datapropertyassertion, aiit).
rdf(datum_zavrsetka, datapropertyassertion, aiit).
rdf(id_konferencije, datapropertyassertion, aiit).
rdf(naucni_odbor, datapropertyassertion, aiit).
rdf(naziv, datapropertyassertion, aiit).
rdf(adresa, datapropertyassertion, bajic_marijana).
rdf(email, datapropertyassertion, bajic_marijana).
rdf(id_ucesnika, datapropertyassertion, bajic_marijana).
rdf(ime, datapropertyassertion, bajic_marijana).
rdf(mobilni, datapropertyassertion, bajic_marijana).
rdf(prezime, datapropertyassertion, bajic_marijana).
rdf(telefon, datapropertyassertion, bajic_marijana).
rdf(ustanova, datapropertyassertion, bajic_marijana).
rdf(web_adresa, datapropertyassertion, bajic_marijana).
rdf(id_rada, datapropertyassertion, baze_podataka_i_internet).
rdf(naziv, datapropertyassertion, baze_podataka_i_internet).
rdf(obim, datapropertyassertion, baze_podataka_i_internet).
rdf(oblast, datapropertyassertion, baze_podataka_i_internet).
rdf(znacaj, datapropertyassertion, baze_podataka_i_internet).
rdf(adresa, datapropertyassertion, berkovic_ivana).
rdf(email, datapropertyassertion, berkovic_ivana).
rdf(id_ucesnika, datapropertyassertion, berkovic_ivana).
rdf(ime, datapropertyassertion, berkovic_ivana).
rdf(mobilni, datapropertyassertion, berkovic_ivana).
rdf(prezime, datapropertyassertion, berkovic_ivana).
rdf(telefon, datapropertyassertion, berkovic_ivana).
rdf(ustanova, datapropertyassertion, berkovic_ivana).
rdf(web_adresa, datapropertyassertion, berkovic_ivana).
rdf(adresa, datapropertyassertion, chotaliya_narendra).
rdf(email, datapropertyassertion, chotaliya_narendra).
rdf(id_ucesnika, datapropertyassertion, chotaliya_narendra).
rdf(ime, datapropertyassertion, chotaliya_narendra).
rdf(mobilni, datapropertyassertion, chotaliya_narendra).
rdf(prezime, datapropertyassertion, chotaliya_narendra).
rdf(telefon, datapropertyassertion, chotaliya_narendra).
rdf(ustanova, datapropertyassertion, chotaliya_narendra).
rdf(web_adresa, datapropertyassertion, chotaliya_narendra).
rdf(id_rada, datapropertyassertion, database_modeling_in_ecology).
```

```
rdf(naziv, datapropertyassertion, database_modeling_in_ecology).
rdf(obim, datapropertyassertion, database_modeling_in_ecology).
rdf(oblast, datapropertyassertion, database_modeling_in_ecology).
rdf(znacaj, datapropertyassertion, database_modeling_in_ecology).
rdf(drzava, datapropertyassertion, ecka).
rdf(naziv, datapropertyassertion, ecka).
rdf(ptt, datapropertyassertion, ecka).
rdf(adresa, datapropertyassertion, filip_snezana).
rdf(email, datapropertyassertion, filip_snezana).
rdf(id_ucesnika, datapropertyassertion, filip_snezana).
rdf(ime, datapropertyassertion, filip_snezana).
rdf(mobilni, datapropertyassertion, filip_snezana).
rdf(prezime, datapropertyassertion, filip_snezana).
rdf(telefon, datapropertyassertion, filip_snezana).
rdf(ustanova, datapropertyassertion, filip_snezana).
rdf(web_adresa, datapropertyassertion, filip_snezana).
rdf(id_rada, datapropertyassertion, green_it_initiative_in_schools).
rdf(naziv, datapropertyassertion, green_it_initiative_in_schools).
rdf(obim, datapropertyassertion, green_it_initiative_in_schools).
rdf(oblast, datapropertyassertion, green_it_initiative_in_schools).
rdf(znacaj, datapropertyassertion, green_it_initiative_in_schools).
rdf(id_sale, datapropertyassertion, hotel_vojvodina).
rdf(kapacitet, datapropertyassertion, hotel_vojvodina).
rdf(ozvucenje, datapropertyassertion, hotel_vojvodina).
rdf(id_rada, datapropertyassertion, implementation_of_baselog_system).
rdf(naziv, datapropertyassertion, implementation_of_baselog_system).
rdf(obim, datapropertyassertion, implementation_of_baselog_system).
rdf(oblast, datapropertyassertion, implementation_of_baselog_system).
rdf(znacaj, datapropertyassertion, implementation_of_baselog_system).
rdf(datum_pocetka, datapropertyassertion, informatika_u_obrazovanju).
rdf(datum_zavrsetka, datapropertyassertion, informatika_u_obrazovanju).
rdf(id_konferencije, datapropertyassertion, informatika_u_obrazovanju).
rdf(ministarstvo, datapropertyassertion, informatika_u_obrazovanju).
rdf(naziv, datapropertyassertion, informatika_u_obrazovanju).
rdf(adresa, datapropertyassertion, kazi_ljubica).
rdf(email, datapropertyassertion, kazi_ljubica).
rdf(id_ucesnika, datapropertyassertion, kazi_ljubica).
rdf(ime, datapropertyassertion, kazi_ljubica).
rdf(mobilni, datapropertyassertion, kazi_ljubica).
rdf(prezime, datapropertyassertion, kazi_ljubica).
rdf(telefon, datapropertyassertion, kazi_ljubica).
rdf(ustanova, datapropertyassertion, kazi_ljubica).
rdf(web_adresa, datapropertyassertion, kazi_ljubica).
rdf(adresa, datapropertyassertion, kazi_zoltan).
rdf(email, datapropertyassertion, kazi_zoltan).
rdf(id_ucesnika, datapropertyassertion, kazi_zoltan).
rdf(ime, datapropertyassertion, kazi_zoltan).
rdf(mobilni, datapropertyassertion, kazi_zoltan).
rdf(prezime, datapropertyassertion, kazi_zoltan).
rdf(telefon, datapropertyassertion, kazi_zoltan).
rdf(ustanova, datapropertyassertion, kazi_zoltan).
rdf(web_adresa, datapropertyassertion, kazi_zoltan).
rdf(adresa, datapropertyassertion, letic_dusko).
rdf(email, datapropertyassertion, letic_dusko).
rdf(id_ucesnika, datapropertyassertion, letic_dusko).
rdf(ime, datapropertyassertion, letic_dusko).
rdf(mobilni, datapropertyassertion, letic_dusko).
rdf(prezime, datapropertyassertion, letic_dusko).
rdf(telefon, datapropertyassertion, letic_dusko).
rdf(ustanova, datapropertyassertion, letic_dusko).
rdf(web_adresa, datapropertyassertion, letic_dusko).
rdf(id_sale, datapropertyassertion, lovacki_dvorac).
rdf(kapacitet, datapropertyassertion, lovacki_dvorac).
rdf(ozvucenje, datapropertyassertion, lovacki_dvorac).
```

```
rdf(adresa, datapropertyassertion, radulovic_biljana).
rdf(email, datapropertyassertion, radulovic_biljana).
rdf(id_ucesnika, datapropertyassertion, radulovic_biljana).
rdf(ime, datapropertyassertion, radulovic_biljana).
rdf(mobilni, datapropertyassertion, radulovic_biljana).
rdf(prezime, datapropertyassertion, radulovic_biljana).
rdf(telefon, datapropertyassertion, radulovic_biljana).
rdf(ustanova, datapropertyassertion, radulovic_biljana).
rdf(web_adresa, datapropertyassertion, radulovic_biljana).
rdf(adresa, datapropertyassertion, ravi_s_jayanti).
rdf(email, datapropertyassertion, ravi_s_jayanti).
rdf(id_ucesnika, datapropertyassertion, ravi_s_jayanti).
rdf(ime, datapropertyassertion, ravi_s_jayanti).
rdf(mobilni, datapropertyassertion, ravi_s_jayanti).
rdf(prezime, datapropertyassertion, ravi_s_jayanti).
rdf(telefon, datapropertyassertion, ravi_s_jayanti).
rdf(ustanova, datapropertyassertion, ravi_s_jayanti).
rdf(web_adresa, datapropertyassertion, ravi_s_jayanti).
rdf(id_rada, datapropertyassertion, rekurzivna_pravila_i_fraktali).
rdf(naziv, datapropertyassertion, rekurzivna_pravila_i_fraktali).
rdf(obim, datapropertyassertion, rekurzivna_pravila_i_fraktali).
rdf(oblast, datapropertyassertion, rekurzivna_pravila_i_fraktali).
rdf(znacaj, datapropertyassertion, rekurzivna_pravila_i_fraktali).
rdf(id_sale, datapropertyassertion, sala_skupstine_opstine).
rdf(kapacitet, datapropertyassertion, sala_skupstine_opstine).
rdf(ozvucenje, datapropertyassertion, sala_skupstine_opstine).
rdf(adresa, datapropertyassertion, tehnicki_fakultet_mihajlo_pupin).
rdf(id_organizatora, datapropertyassertion,
tehnicki_fakultet_mihajlo_pupin).
rdf(ime, datapropertyassertion, tehnicki_fakultet_mihajlo_pupin).
rdf(datum_pocetka, datapropertyassertion, urbaneco).
rdf(datum_zavrsetka, datapropertyassertion, urbaneco).
rdf(id_konferencije, datapropertyassertion, urbaneco).
rdf(naucni_odbor, datapropertyassertion, urbaneco).
rdf(naziv, datapropertyassertion, urbaneco).
rdf(drzava, datapropertyassertion, zrenjanin).
rdf(naziv, datapropertyassertion, zrenjanin).
rdf(ptt, datapropertyassertion, zrenjanin).
```

Veze između različith objekata:

```
rdf(kazi_zoltan, uplatio, k6500).
rdf(kazi_zoltan, uplatio, k7500).
rdf(kazi_ljubica, uplatio, k7500).
rdf(hotel_vojvodina, odrzava_se, aiit).
rdf(tehnicki_fakultet_mihajlo_pupin, organizuje, aiit).
rdf(implementation_of_baselog_system, napisao, bajic_marijana).
rdf(aiit, prisustvuje, bajic_marijana).
rdf(k6500, uplatio, bajic_marijana).
rdf(radulovic_biljana, napisao, baze_podataka_i_internet).
rdf(k1000, uplatio, baze_podataka_i_internet).
rdf(rekurzivna_pravila_i_fraktali, izlaze, berkovic_ivana).
rdf(implementation_of_baselog_system, napisao, berkovic_ivana).
rdf(rekurzivna_pravila_i_fraktali, napisao, berkovic_ivana).
rdf(aiit, prisustvuje, berkovic_ivana).
rdf(informatika_u_obrazovanju, prisustvuje, berkovic_ivana).
rdf(k1000, uplatio, berkovic_ivana).
rdf(k6500, uplatio, berkovic_ivana).
rdf(green_it_initiative_in_schools, napisao, chotaliya_narendra).
rdf(urbaneco, prisustvuje, chotaliya_narendra).
rdf(k7500, uplatio, chotaliya_narendra).
rdf(letic_dusko, napisao, database_modeling_in_ecology).
rdf(radulovic_biljana, napisao, database_modeling_in_ecology).
```

```
rdf(kazi_zoltan, napisao, database_modeling_in_ecology).
rdf(filip_snezana, napisao, database_modeling_in_ecology).
rdf(k7500, uplatio, database_modeling_in_ecology).
rdf(lovacki_dvorac, nalazi_se, ecka).
rdf(urbaneco, odrzava_se, ecka).
rdf(database_modeling_in_ecology, napisao, filip_snezana).
rdf(urbaneco, prisustvuje, filip_snezana).
rdf(7500, uplatio, filip_snezana).
rdf(kazi_ljubica, napisao, green_it_initiative_in_schools).
rdf(chotaliya_narendra, napisao, green_it_initiative_in_schools).
rdf(ravi_s_jayanti, napisao, green_it_initiative_in_schools).
rdf(kazi_zoltan, napisao, green_it_initiative_in_schools).
rdf(letic_dusko, napisao, green_it_initiative_in_schools).
rdf(k6500, uplatio, green_it_initiative_in_schools).
rdf(zrenjanin, nalazi_se, hotel_vojvodina).
rdf(aiit, odrzava_se, hotel_vojvodina).
rdf(berkovic_ivana, napisao, implementation_of_baselog_system).
rdf(bajic_marijana, napisao, implementation_of_baselog_system).
rdf(radulovic_biljana, napisao, implementation_of_baselog_system).
rdf(kazi_zoltan, napisao, implementation_of_baselog_system).
rdf(k6500, uplatio, implementation_of_baselog_system).
rdf(sala_skupstine_opstine, odrzava_se, informatika_u_obrazovanju).
rdf(tehnicki_fakultet_mihajlo_pupin, organizuje,
informatika_u_obrazovanju).
rdf(green_it_initiative_in_schools, izlaze, kazi_ljubica).
rdf(green_it_initiative_in_schools, napisao, kazi_ljubica).
rdf(urbaneco, prisustvuje, kazi_ljubica).
rdf(7500, uplatio, kazi_ljubica).
rdf(implementation_of_baselog_system, izlaze, kazi_zoltan).
rdf(database_modeling_in_ecology, izlaze, kazi_zoltan).
rdf(implementation_of_baselog_system, napisao, kazi_zoltan).
rdf(green_it_initiative_in_schools, napisao, kazi_zoltan).
rdf(urbaneco, prisustvuje, kazi_zoltan).
rdf(aiit, prisustvuje, kazi_zoltan).
rdf(k7500, uplatio, kazi_zoltan).
rdf(k6500, uplatio, kazi_zoltan).
rdf(green_it_initiative_in_schools, napisao, letic_dusko).
rdf(urbaneco, prisustvuje, letic_dusko).
rdf(k7500, uplatio, letic_dusko).
rdf(ecka, nalazi_se, lovacki_dvorac).
rdf(urbaneco, odrzava_se, lovacki_dvorac).
rdf(baze_podataka_i_internet, izlaze, radulovic_biljana).
rdf(implementation_of_baselog_system, napisao, radulovic_biljana).
rdf(database_modeling_in_ecology, napisao, radulovic_biljana).
rdf(urbaneco, prisustvuje, radulovic_biljana).
rdf(implementation_of_baselog_system, izlaze, radulovic_biljana).
rdf(informatika_u_obrazovanju, prisustvuje, radulovic_biljana).
rdf(k7500, uplatio, radulovic_biljana).
rdf(k6500, uplatio, radulovic_biljana).
rdf(k1000, uplatio, radulovic_biljana).
rdf(green_it_initiative_in_schools, napisao, ravi_s_jayanti).
rdf(urbaneco, prisustvuje, ravi_s_jayanti).
rdf(k7500, uplatio, ravi_s_jayanti).
rdf(berkovic_ivana, napisao, rekurzivna_pravila_i_fraktali).
rdf(k1000, uplatio, rekurzivna_pravila_i_fraktali).
rdf(zrenjanin, nalazi_se, sala_skupstine_opstine).
rdf(informatika_u_obrazovanju, odrzava_se, sala_skupstine_opstine).
rdf(zrenjanin, je_iz, tehnicki_fakultet_mihajlo_pupin).
rdf(urbaneco, organizuje, tehnicki_fakultet_mihajlo_pupin).
rdf(aiit, organizuje, tehnicki_fakultet_mihajlo_pupin).
rdf(informatika_u_obrazovanju, organizuje,
tehnicki_fakultet_mihajlo_pupin).
rdf(lovacki_dvorac, odrzava_se, urbaneco).
rdf(tehnicki_fakultet_mihajlo_pupin, organizuje, urbaneco).
```

```
rdf(sala_skupstine_opstine, nalazi_se, zrenjanin).
rdf(tehnicki_fakultet_mihajlo_pupin, nalazi_se, zrenjanin).
rdf(hotel_vojvodina, nalazi_se, zrenjanin).
rdf(informatika_u_obrazovanju, odrzava_se, zrenjanin).
rdf(aiit, odrzava_se, zrenjanin).
```

Ograničenja osobina objekata (*Object Property Range - Cardinality*):

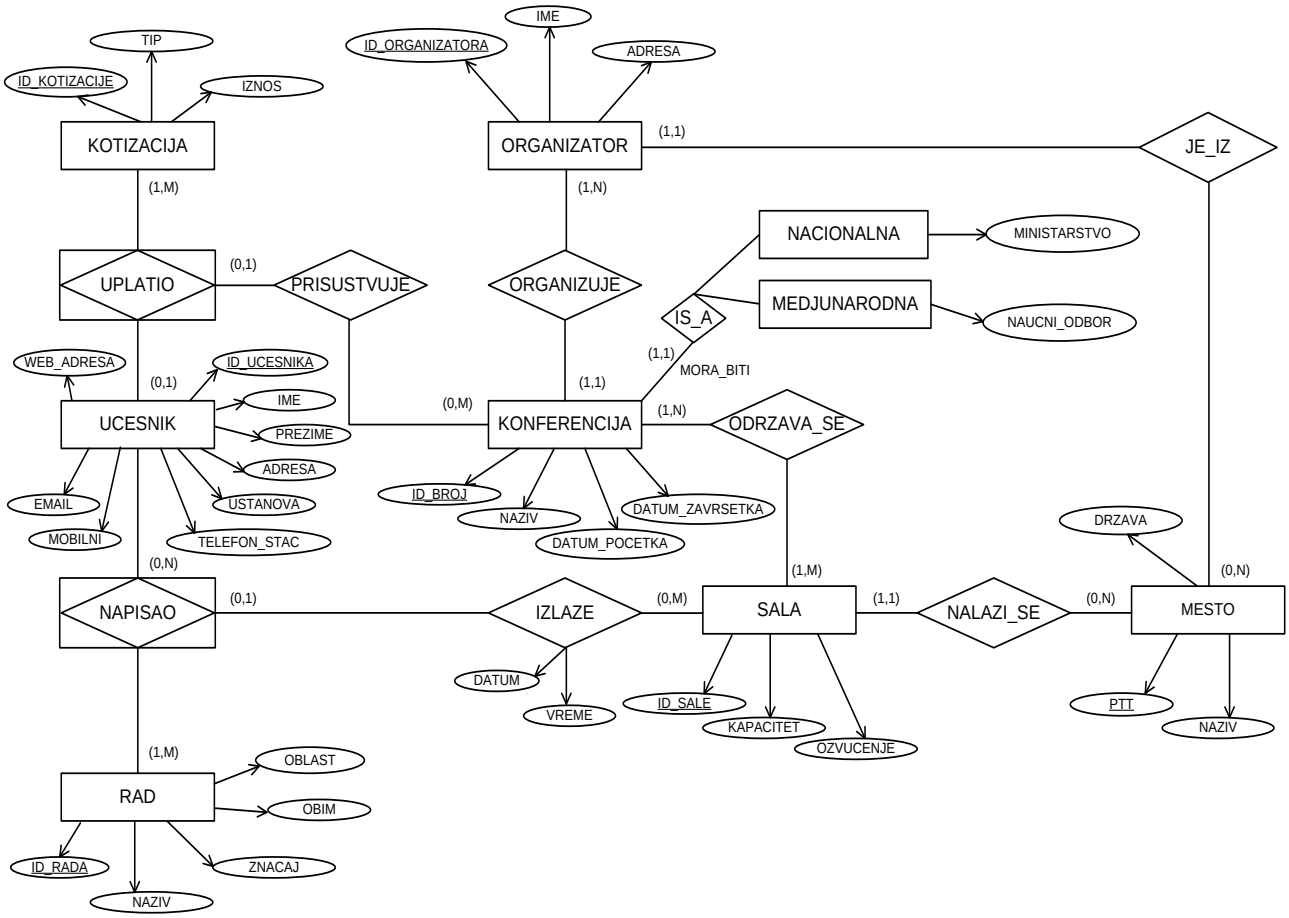
```
rdf(ucesnik, izlaze, maxcardm).
rdf(rad, izlaze, mincard0).
rdf(ucesnik, izlaze, mincard0).
rdf(rad, izlaze, maxcard1).
rdf(mesto, je_iz, maxcardm).
rdf(mesto, je_iz, mincard0).
rdf(organizator, je_iz, mincard1).
rdf(organizator, je_iz, maxcard1).
rdf(konferencija, moze_biti, mincard1).
rdf(konferencija, moze_biti, maxcard1).
rdf(mesto, nalazi_se, maxcardm).
rdf(mesto, nalazi_se, mincard0).
rdf(sala, nalazi_se, mincard1).
rdf(sala, nalazi_se, maxcard1).
rdf(rad, napisao, maxcardm).
rdf(ucesnik, napisao, maxcardm).
rdf(rad, napisao, mincard1).
rdf(ucesnik, napisao, mincard1).
rdf(konferencija, odrzava_se, maxcardm).
rdf(sala, odrzava_se, maxcardm).
rdf(konferencija, odrzava_se, mincard1).
rdf(sala, odrzava_se, mincard1).
rdf(organizator, organizuje, maxcardm).
rdf(konferencija, organizuje, mincard1).
rdf(organizator, organizuje, mincard1).
rdf(konferencija, organizuje, maxcard1).
rdf(konferencija, priredjuje, mincard1).
rdf(sala, priredjuje, mincard1).
rdf(konferencija, prisustvuje, maxcardm).
rdf(ucesnik, prisustvuje, maxcardm).
rdf(konferencija, prisustvuje, mincard0).
rdf(ucesnik, prisustvuje, mincard0).
rdf(ucesnik, uplatio, maxcardm).
rdf(ucesnik, uplatio, mincard0).
rdf(kotizacija, uplatio, mincard1).
rdf(kotizacija, uplatio, maxcard1).
```

7.3. MODEL PODATAKA ZA PRIMER «ORGANIZACIJA KONFERENCIJA»

7.3.1. Dijagram ER modela podataka

ER model podataka je kreiran prema tekstualnoj specifikaciji sistema. Na osnovu definicija 3.2, 3.3, 3.4, 3.5, 3.6, i 3.7 iz [Mogin i sar. 1996], određeni su skupovi entiteta, atributa, identifikacioni/primarni atributi, domeni atributa, definisane su osobine poveznika/relacija između tipova entiteta, gerundi (mešoviti objekti-veze) i IS_A hijerarhija, tj. specijalizacija tipa entiteta.

Dijagram ER modela podataka je prikazan na slici 7.8.



Slika 7.8. – Dijagram ER modela podataka

7.3.2. Domeni atributa

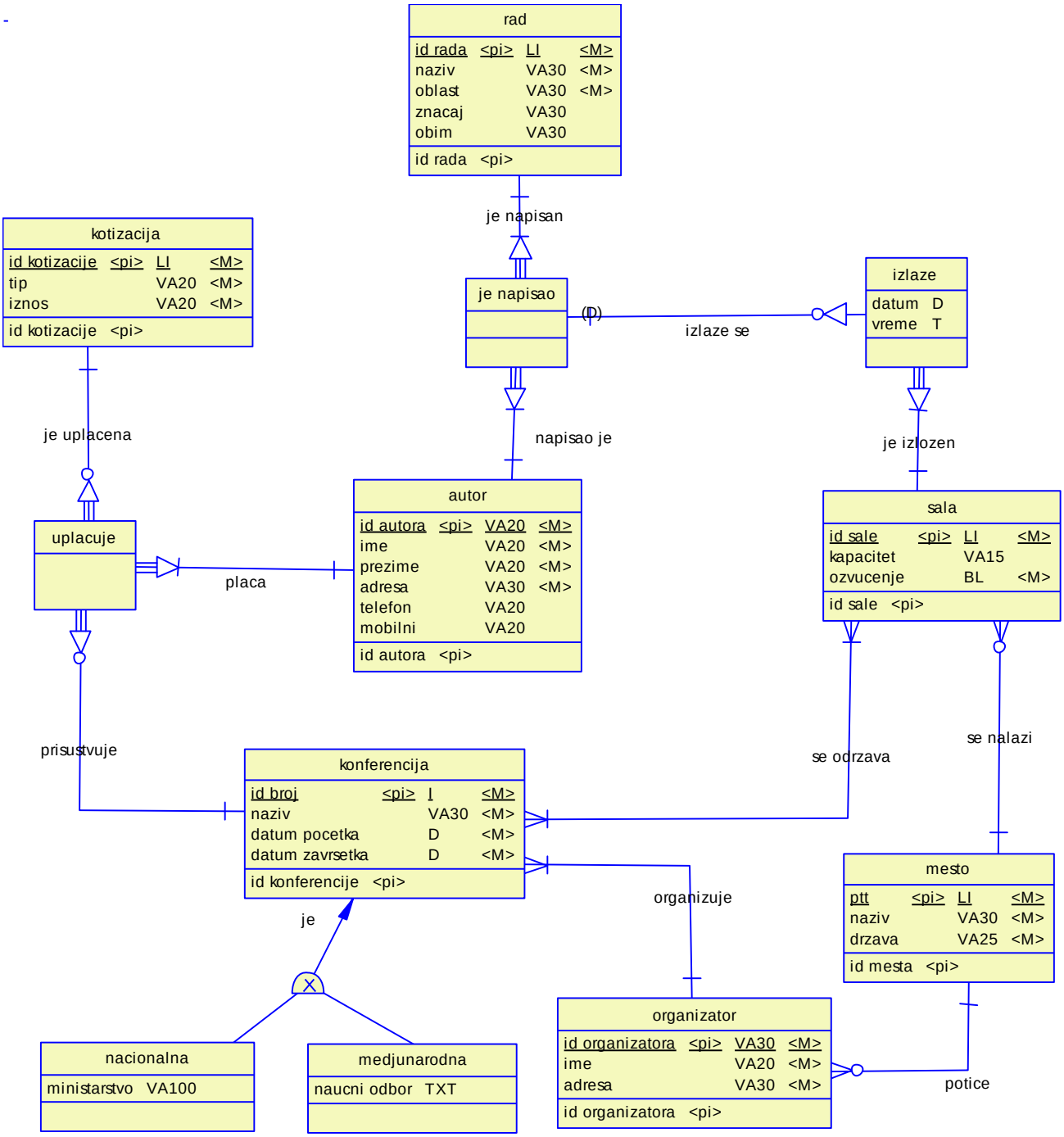
Kao što je u teoretskom istraživanju navedeno, prema [Mogin i sar. 1996], svakom obeležju odgovara jedan skup mogućih vrednosti koje to obeležje može imati koji se naziva se domenom obeležja. Domen obeležja može ali ne mora posedovati posebno ime u odnosu na atribut. Isti domen se može pridružiti većem broju različitih obeležja, kao što se može videti u primeru atributa i njihovih domena, u tabeli 7.1. kolona „Pripada entitetu/vezi" sadrži nazive tipova entiteta ili tipova poveznika/gerunda, kojima su pridruženi atributi određenog tipa podatka.

Tabela 7.1: Domeni atributa konceptualnog modela podataka

Naziv	**Kod**	**Tip podatka**	**Pripada entitetu/vezi**
adresa	ADRESA	STRING	Organizator;Autor
datum	DATUM	DATE	Izlaze
datum pocetka	DATUM_POCETKA	DATE	Konferencija
datum zavrsetka	DATUM_ZAVRSETKA	DATE	Konferencija
drzava	DRZAVA	STRING	Mesto
id autora	ID_AUTORA	STRING	Autor
id broj	ID_BROJ	INTEGER	Konferencija
id kotizacije	ID_KOTIZACIJE	LONG INTEGER	Kotizacija
id organizatora	ID_ORGANIZATORA	STRING	Organizator
id rada	ID_RADA	LONG INTEGER	Rad
id sale	ID_SALE	LONG INTEGER	Sala
ime	IME	STRING	Organizator;Autor
iznos	IZNOS	STRING	Kotizacija
kapacitet	KAPACITET	STRING	Sala
ministarstvo	MINISTARSTVO	STRING	Nacionalna
mobilni	MOBILNI	STRING	Autor
naucni odbor	NAUCNI_ODBOR	TEXT	Medjunarodna
naziv	NAZIV	STRING	Konferencija;Mesto;Rad
obim	OBIM	STRING	Rad
oblast	OBLAST	STRING	Rad
ozvucenje	OZVUCENJE	BOOLEAN	Sala
prezime	PREZIME	STRING	Autor
ptt	PTT	LONG INTEGER	Mesto
telefon	TELEFON	STRING	Autor
tip	TIP	STRING	Kotizacija
vreme	VREME	TIME	Izlaze
znacaj	ZNACAJ	STRING	Rad

7.3.3. Konceptualni model podataka u CASE alatu

Dijagram EER modela podataka, tj. konceptualnog modela u CASE alatu, prikazuje strukturu modela: entitete, poveznike, atribute, identifikacione atribute kao i određene elemente rečnika podataka, kao što su ograničenja domena atributa i kardinaliteta tipa poveznika (Slika 7.9.).



Slika 7.9. – Dijagram ER modela podataka u CASE alatu

7.3.4. Formalizacija konceptualnog modela podataka

Na osnovu teoretskog istraživanja i predložene formalizacije konceptualnog modela na predikatskom računu i transformacije u Prolog klauzule, rezultat postupka su sledeći skupovi entiteta *E,* atributa *A,* relacija *R,* ograničenja *S*, kao i relacija između entiteta, atributa, poveznika i ograničenja *P* formirani DMV programom:

E = {

ent(konferencija),	ent(mesto),	ent(je_napisao),
ent(organizator),	ent(sala),	ent(izlaze),
ent(medjunarodna),	ent(autor),	ent(kotizacija),
ent(nacionalna),	ent(rad),	ent(uplacuje)
}		

A = {

atr(id_broj),	atr(drzava),	atr(id_rada),
atr(naziv),	atr(id_sale),	atr(naziv),
atr(datum_pocetka),	atr(kapacitet),	atr(oblast),
atr(datum_zavrsetka),	atr(ozvucenje),	atr(znacaj),
atr(id_organizatora),	atr(id_autora),	atr(obim),
atr(ime),	atr(ime),	atr(datum),
atr(adresa),	atr(prezime),	atr(vreme),
atr(naucni_odbor),	atr(adresa),	atr(id_kotizacije),
atr(ministarstvo), atr(ptt),	atr(telefon),	atr(tip),
atr(naziv),	atr(mobilni),	atr(iznos)
}		

R = {

rel(organizuje),	rel(napisao_je),	rel(placa),
rel(potice),	rel(je_napisan),	rel(je_uplacena),
rel(se_odrzava),	rel(izlaze_se),	rel(prisustvuje)
rel(se_nalazi),	rel(je_izlozen),	
}		

S = {

res(idatr),	res(cardm),	res(txt),
res(mandatory),	res(dependent),	res(li),
res(card01),	res(inherit),	res(va),
res(card11),	res(i),	res(va),
res(card0m),	res(va),	res(bl),
res(card1m),	res(d),	res(t)
res(card0),	res(va),	
res(card1),	res(va),	
}		

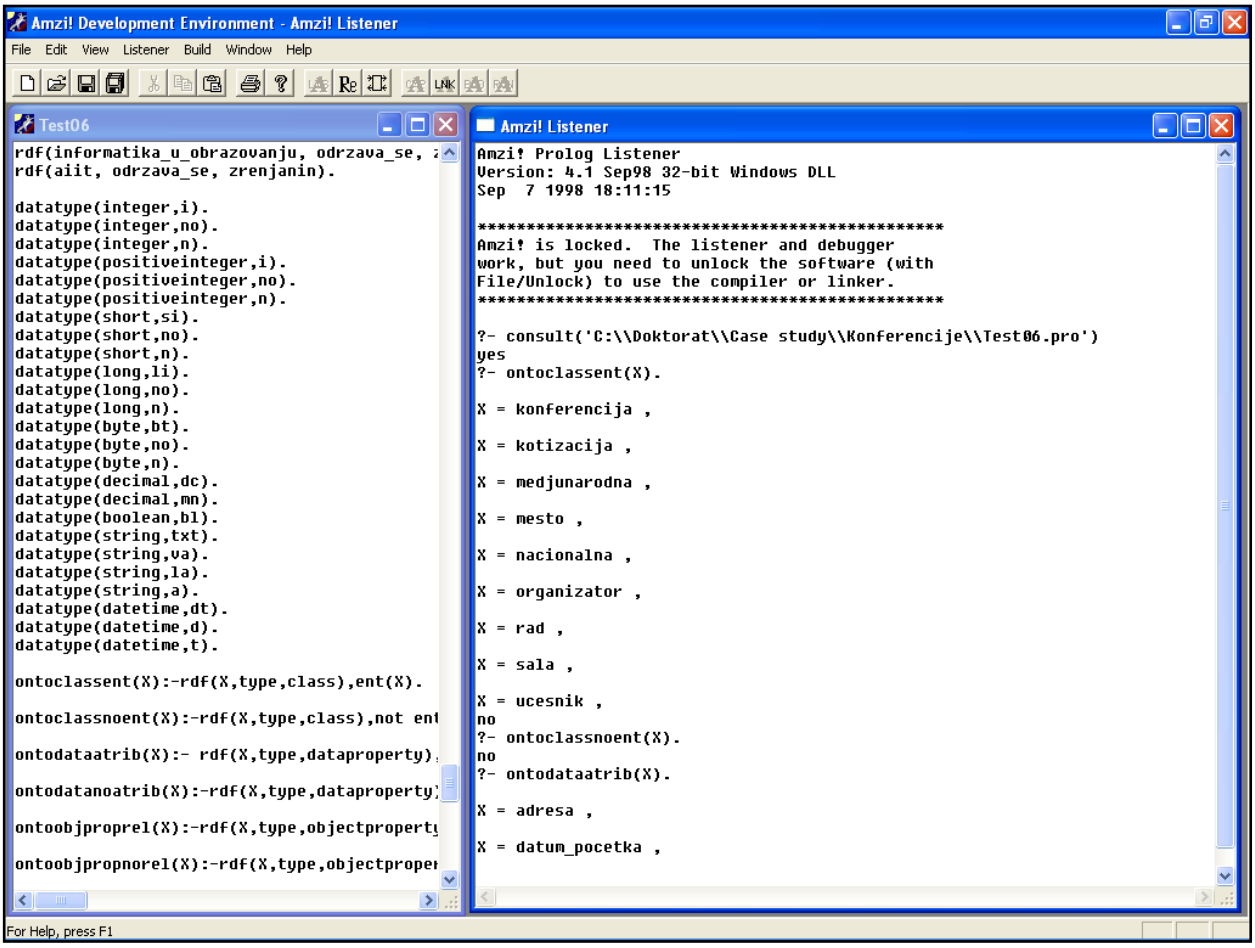
P = {

p(id_broj, i),	p(naziv, va),	p(mobilni, va),
p(naziv, va),	p(drzava, li),	p(id_rada, va),
p(datum_pocetka, d),	p(id_sale, va),	p(naziv, d),
p(datum_zavrsetka, d),	p(kapacitet, bl),	p(oblast, t),
p(id_organizatora, va),	p(ozvucenje, va),	p(znacaj, li),
p(ime, va),	p(id_autora, va),	p(obim, va),
p(adresa, va),	p(ime, va),	p(datum, va),
p(naucni_odbor, va),	p(prezime, va),	p(konferencija, id_broj),
p(ministarstvo, txt),	p(adresa, li),	p(konferencija, naziv),
p(ptt, li),	p(telefon, va),	

p(konferencija, datum_pocetka),
p(konferencija, datum_zavrsetka),
p(organizator, id_organizatora),
p(organizator, ime),
p(organizator, adresa),
p(medjunarodna, naucni_odbor),
p(nacionalna, ministarstvo),
p(mesto, ptt),
p(mesto, naziv),
p(mesto, drzava),
p(sala, id_sale),
p(sala, kapacitet),
p(sala, ozvucenje),
p(autor, id_autora),
p(autor, ime),
p(autor, prezime),
p(autor, adresa),
p(autor, telefon),
p(autor, mobilni),
p(rad, id_rada),
p(rad, naziv),
p(rad, oblast),
p(rad, znacaj),
p(rad, obim),
p(izlaze, datum),
p(izlaze, vreme),
p(kotizacija, id_kotizacije),
p(kotizacija, tip),
p(kotizacija, iznos),
p(konferencija, idatr),
p(organizator, idatr),
p(mesto, idatr),
p(sala, idatr),
p(autor, idatr),
p(rad, idatr),
p(kotizacija, idatr),
p(id_broj, mandatory),
p(naziv, mandatory),
p(datum_pocetka, mandatory),
p(datum_zavrsetka, mandatory),
p(id_organizatora, mandatory),
p(ime, mandatory),
p(adresa, mandatory),
p(ptt, mandatory),
p(naziv, mandatory),
p(drzava, mandatory),
p(id_sale, mandatory),
p(kapacitet, mandatory),
p(ozvucenje, mandatory),
p(id_autora, mandatory),
p(ime, mandatory),
p(prezime, mandatory),
p(adresa, mandatory),
p(id_rada, mandatory),
p(naziv, mandatory),
p(oblast, mandatory),
p(id_kotizacije, mandatory),
p(tip, mandatory),
p(iznos, mandatory),
p(organizator, organizuje),
p(organizuje, konferencija),
p(mesto, potice),
p(potice, organizator),
p(sala, se_odrzava),
p(se_odrzava, konferencija),
p(mesto, se_nalazi),
p(se_nalazi, sala),
p(je_napisao, napisao_je),
p(napisao_je, autor),
p(rad, je_napisan),
p(je_napisan, je_napisao),
p(izlaze, izlaze_se),
p(izlaze_se, je_napisao),
p(sala, je_izlozen),
p(je_izlozen, izlaze),
p(uplacuje, placa),
p(placa, autor),
p(kotizacija, je_uplacena),
p(je_uplacena, uplacuje),
p(konferencija, prisustvuje),
p(prisustvuje, uplacuje),
p(organizuje, mincard1),
p(organizuje, maxcard1),
p(mincard0, potice),
p(maxcardm, potice),
p(potice, mincard1),
p(potice, maxcard1),
p(mincard1, se_odrzava),
p(maxcardm, se_odrzava),
p(se_odrzava, mincard1),
p(se_odrzava, maxcardm),
p(mincard0, se_nalazi),
p(maxcardm, se_nalazi),
p(se_nalazi, mincard1),
p(se_nalazi, maxcard1),
p(mincard1, napisao_je),
p(maxcard1, napisao_je),
p(napisao_je, mincard1),
p(napisao_je, maxcardm),
p(mincard1, je_napisan),
p(maxcardm, je_napisan),
p(je_napisan, mincard1),
p(je_napisan, maxcard1),
p(mincard1, izlaze_se),
p(maxcard1, izlaze_se),
p(izlaze_se, mincard0),
p(izlaze_se, maxcard1),
p(mincard1, je_izlozen),
p(maxcardm, je_izlozen),
p(je_izlozen, mincard1),
p(je_izlozen, maxcard1),
p(mincard1, placa),
p(maxcard1, placa),
p(placa, mincard1),
p(placa, maxcardm),
p(mincard0, je_uplacena),
p(maxcardm, je_uplacena),
p(je_uplacena, mincard1),
p(je_uplacena, maxcard1),
p(mincard0, prisustvuje),
p(maxcardm, prisustvuje),
p(prisustvuje, mincard1),
p(prisustvuje, maxcard1),
p(dependent, napisao_je),
p(je_napisan, dependent),
p(dependent, izlaze_se),
p(je_izlozen, dependent),
p(dependent, placa),
p(je_uplacena, dependent),
p(prisustvuje, dependent),
p(id_broj, idatr),
p(id_organizatora, idatr),
p(ptt, idatr),
p(id_sale, idatr),
p(id_autora, idatr),
p(id_rada, idatr),
p(id_kotizacije, idatr),
p(je, inherit),
p(je, card1),
p(konferencija, je),
p(je, medjunarodna),
p(je, nacionalna)
}

7.5. TESTIRANJE PRIMERA

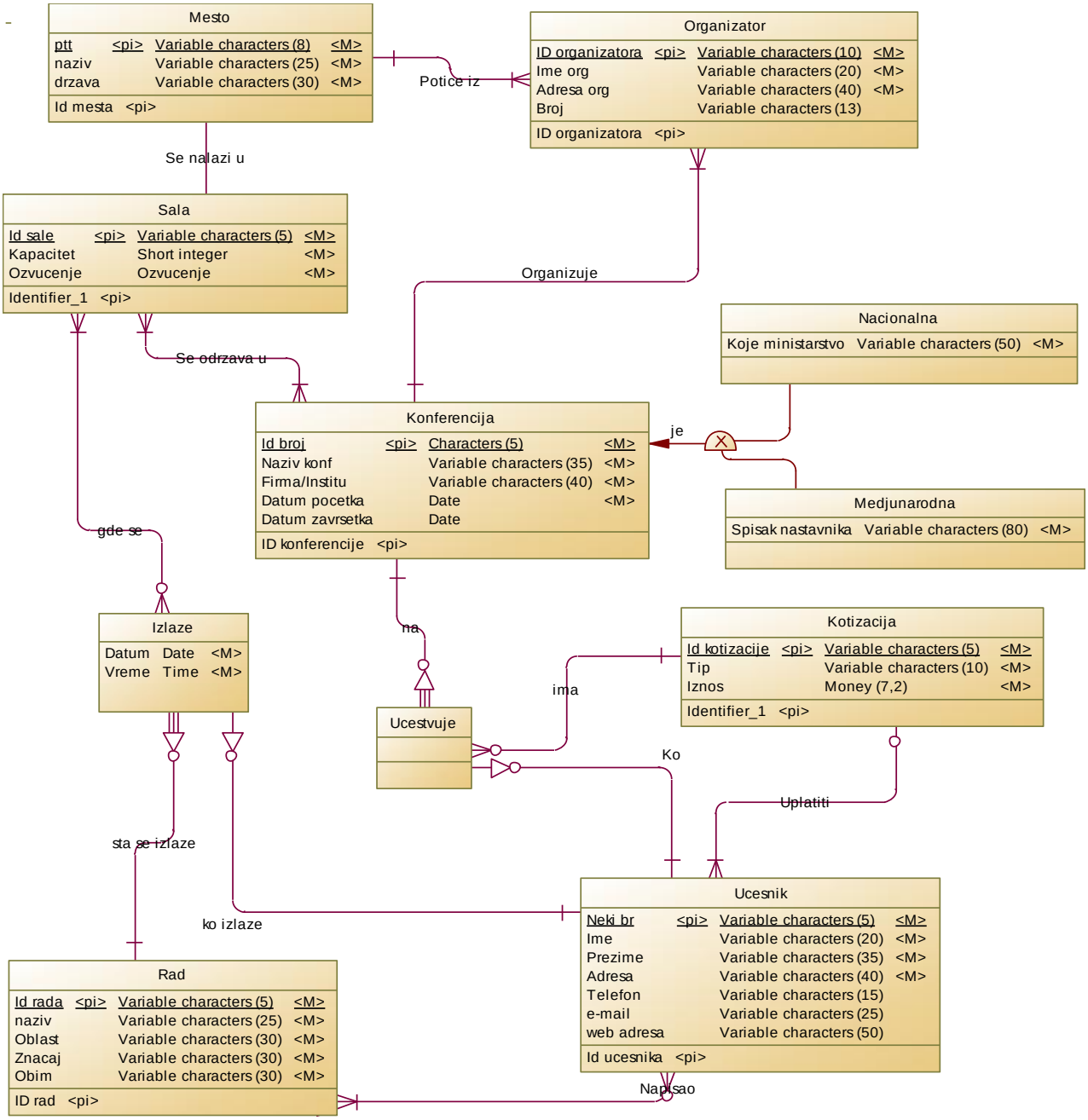
Na slici 7.10. se može videti izgled ekrana Amzi! Prolog sistema sa učitanim integrisanim modelom podataka, ontologijom i pravilima zaključivanja za model K06.cdm (sadržaj prozora levo), kao i set pitanja koji se preko Amzi!Listenera postavljaju (sadržaj prozora desno) i na koja se dobijaju odgovori sistema, a na osnovu kojih se dolazi do vrednosti elemenata za propisanu metriku (takođe u desnom prozoru).



Slika 7.10. - Testiranje korektnosti modela podataka K06.CDM u Prologu

Testiranje primera je pokazano na studentskim konceptualnim modelima podataka pod oznakama K06.cdm (pikazan šemom modela na slici 7.11) i K01.cdm (pikazan šemom modela na slici 7.12).

Ovi modeli su izabrani kao primer semantički korektnijeg – K06.cdm i jednog od radova koji su najslabije semantički usklađeni sa domenskom ontologijom – K01.cdm.



Slika 7.11. – Dijagram konceptualnog modela podataka K06.cdm

1. (Testiranje pravila R 5.1) Koje klase ontologije jesu pokrivene tipovima entiteta u konceptualnom modelu podataka?

```
?- ontoclassent(X).
X = konferencija ,
X = kotizacija ,
X = medjunarodna ,
X = mesto ,
X = nacionalna ,
X = organizator ,
X = rad ,
X = sala ,
X = ucesnik ,
no
```

Ukupno entiteta za ontološku ocenu: **9**

2. (Testiranje pravila R 5.2) Koje klase ontologije nisu pokrivene tipovima entiteta u konceptualnom modelu podataka?

```
?- ontoclassnoent(X).
no
```

Ukupno klasa: **0**

3. (Testiranje pravila R 5.3) Koje osobine podataka klasa ontologije jesu pokrivene atributima u konceptualnom modelu podataka?

```
?- ontodataatrib(X).
X = adresa ,
X = datum_pocetka ,
X = datum_zavrsetka ,
X = drzava ,
X = id_kotizacije ,
X = id_organizatora ,
X = id_rada ,
X = id_sale ,
X = ime ,
X = iznos ,
X = kapacitet ,
X = naziv ,
X = obim ,
X = oblast ,
X = ozvucenje ,
X = prezime ,
X = ptt ,
X = telefon ,
X = web_adresa ,
X = znacaj ,
no
```

Ukupno atributa za ontološku ocenu: **20**

4. (Testiranje pravila R 5.4) Koje osobine podataka klasa ontologije nisu pokrivene atributima u konceptualnom modelu podataka?

```
?- ontodatanoatrib(X).
X = datum_izlaganja ,
X = email ,
X = id_konferencije ,
X = id_ucesnika ,
X = ministarstvo ,
X = mobilni ,
X = naucni_odbor ,
X = tip_kotizacije ,
X = ustanova ,
X = vreme_izlaganja ,
no
```

Ukupno osobina podataka klasa: **10**

5. (Testiranje pravila R 5.5) Koje osobine objekata klasa ontologije jesu pokrivene relacijama u konceptualnom modelu podataka?

```
?- ontoobjproprel(X).
X = napisao ,
X = organizuje ,
no
```

Ukupno poveznika/relacija za ontološku ocenu: **2**

6. (Testiranje pravila R 5.6) Koje osobine objekata klasa ontologije nisu pokrivene relacijama u konceptualnom modelu podataka?

```
?-ontoobjpropnorel(X).
X = izlaze ,
X = je_iz ,
X = moze_biti ,
X = nalazi_se ,
X = odrzava_se ,
X = priredjuje ,
X = prisustvuje ,
X = uplatio ,
no
```

Ukupno osobina objekata klasa: **8**

7. (Testiranje pravila R 5.7) Za koje ontološke osobine podataka postoje odgovarajući atributi u konceptualnom modelu podataka takvi da su usklađeni i njihovi tipovi podataka?

```
?-ontodataatribtype(X,Y).
X = adresa
Y = string ,
X = datum_pocetka
Y = datetime ,
X = drzava
Y = string ,
X = ime
Y = string ,
X = naziv
Y = string ,
X = oblast
Y = string ,
X = prezime
Y = string ,
X = telefon
Y = string ,
X = web_adresa
Y = string ,
X = znacaj
Y = string ,
no
```

Ukupno atributa i tipova podataka za ontološku ocenu: **10**

8. (Testiranje pravila R 5.8) Za koje ontološke osobine podataka postoje atributi u konceptualnom modelu podataka takvi da su im nazivi različiti (sinonimi za atribute) ali su odgovarajućih tipova podataka?

```
?-ontodataatribtypesin(X,Y,X1,Y1).
```

```
X = adresa
Y = string
X1 = id_broj
Y1 = a ,
```

NE

```
X = adresa
Y = string
X1 = naziv_konf    NE
Y1 = va ,
...
X = adresa
Y = string
X1 = adresa_org    DA
Y1 = va ,
...
X = drzava
Y = string
X1 = neki_br       NE
Y1 = va ,
X = drzava
Y = string
X1 = prezime       NE
Y1 = va ,
...
```

Ukupno sinonima atributa u entitetima za ontološku ocenu: **1**

9. (Testiranje pravila R 5.9) Za koje skupove ontoloških osobina podataka objekata klasa postoje definisani odgovarajući atributi u okviru entiteta modela podataka?

?-ontoclassentdataatrib(X,Y).

```
X = konferencija
Y = datum_pocetka ,
X = konferencija
Y = datum_zavrsetka ,
X = kotizacija
Y = id_kotizacije ,
X = kotizacija
Y = iznos ,
X = mesto
Y = drzava ,
X = mesto
Y = naziv ,
X = mesto
Y = ptt ,
X = organizator
Y = id_organizatora ,
X = rad
Y = id_rada ,
X = rad
Y = naziv ,
X = rad
Y = obim ,
X = rad
Y = oblast ,
X = rad
Y = znacaj ,
X = sala
Y = id_sale ,
X = sala
Y = kapacitet ,
```

```
X = sala
Y = ozvucenje ,

X = ucesnik
Y = adresa ,

X = ucesnik
Y = ime ,

X = ucesnik
Y = prezime ,

X = ucesnik
Y = telefon ,

X = ucesnik
Y = web_adresa ,

no
```

Ukupno atributa u entitetima za ontološku ocenu: **21**

10. (Testiranje pravila R 5.10) Koje ontološke osobine objekata klasa odgovaraju poveznicima, tj. relacijama između tipova entiteta pokrivenih ontološkim klasama?

?-ontorel(XC1,YOP,XC2,XE1,YR,XE2).

```
XC1 = konferencija
YOP = organizuje
XC2 = organizator
XE1 = konferencija
YR = organizuje
XE2 = organizator ,

XC1 = rad
YOP = napisao
XC2 = ucesnik
XE1 = rad
YR = napisao
XE2 = ucesnik ,

no
```

Ukupno poveznika/relacija za ontološku ocenu: **2**

11. (Testiranje pravila R 5.11) Koje ontološke osobine objekata klasa se po nazivu razlikuju od naziva poveznika/relacija uspostavljenih između tipova entiteta pokrivenih ontološkim klasama (sinonimi za relacije)?

?-ontorelsinrel(C1,P,C2,E1,R,E2).

```
XC1 = kotizacija
YOP = uplatio
XC2 = ucesnik
XE1 = kotizacija
YR = uplatiti
XE2 = ucesnik ,
```

DA

```
XC1 = mesto
YOP = nalazi_se
XC2 = sala
XE1 = mesto
YR = se_nalazi_u
XE2 = sala ,
```

DA

```
XC1 = mesto
YP = je_iz
XC2 = organizator
XE1 = mesto
YR = potice_iz
XE2 = organizator ,
```

DA

```
XC1 = rad
YOP = izlaze
XC2 = ucesnik        NE
XE1 = rad
YR = napisao
XE2 = ucesnik ,
```

```
XC1 = sala
YOP = odrzava_se
XC2 = konferencija    DA
XE1 = sala
YR = se_odrzava_u
XE2 = konferencija ,
```

```
no
```

Ukupno poveznika/relacija koji imaju sinonime za ontološku ocenu: **4**

12. (Testiranje pravila R 5.12) Koje ontološke osobine objekata klasa odgovaraju poveznicima, tj. relacijama uspostavljenim između entiteta koji nisu pokriveni odgovaraju ontološkim klasama po nazivu (sinonimi za entitet)?

?-ontorelsinent(XC1,YOP,XC2,XE1,YR,XE2).

```
XC1 = medjunarodna
YOP = organizuje
XC2 = organizator
XE1 = konferencija
YR = organizuje
XE2 = organizator ,
...
no
```

Ukupno poveznika/relacija gde postoje sinonimi kod jednog entiteta za ontološku ocenu: **0**

13. (Testiranje pravila R 5.13) Koje ontološke osobine objekata klasa odgovaraju poveznicima/relacijama uspostavljenim između entiteta koji nisu pokriveni odgovaraju ontološkim klasama po nazivu (sinonimi za oba entiteta)?

?-ontorelsinent2(XC1,YOP,XC2,XE1,YR,XE2).

```
no
```

Ukupno poveznika/relacija gde postoje sinonimi kod oba entiteta za ontološku ocenu: **0**

14. (Testiranje pravila R 5.14) Koje ontološke osobine objekata klasa se po nazivu razlikuju od relacija uspostavljenih između entiteta od kojih je samo jedan entitet pokriven odgovarajućom ontološkom klasom po nazivu, dok se drugi može razlikovati (sinonimi za relaciju i jedan od entiteta)?

?-ontorelsinentrel(XC1,YOP,XC2,XE1,YR,XE2).

```
XC1 = konferencija
YOP = odrzava_se
XC2 = mesto
XE1 = konferencija    NE
YR = organizuje
XE2 = organizator ,
...
no
```

Ukupno poveznika/relacija gde postoje sinonimi kod jednog entiteta i poveznika za ontološku ocenu: **0**

15. (Testiranje pravila R 5.15) Da li postoje ontološke osobine objekata klasa koji odgovaraju mešovitim objektima veze, tj. gerundima u modelu podataka?

```
?-ontoobjpropger(XC1,YOP,XC2,XE1,YER,XE2).
XC1 = rad
YOP = izlaze
XC2 = ucesnik
XE1 = rad
YER = izlaze
XE2 = sala ,
XC1 = rad
YOP = izlaze
XC2 = ucesnik
XE1 = rad
YER = izlaze
XE2 = ucesnik ,
...
no
```

Ukupno mešovitih objekata-veza za ontološku ocenu: **1**

16. (Testiranje pravila R 5.16) Da li postoje ontološke osobine objekata klase koje po nazivu ne odgovaraju mešovitim objektima veze, tj. gerundima u modelu podataka, ali su uspostavljene između entiteta za koje su definsane ontološke klase (sinonimi za gerunde)?

```
?-ontoobjpropgersin(XC1,YOP,XC2,XE1,YER,XE2).
XC1 = sala
YOP = nalazi_se
XC2 = mesto          NE
XE1 = rad
YER = izlaze
XE2 = sala ,
XC1 = rad
YOP = napisao
XC2 = ucesnik        NE
XE1 = rad
YER = izlaze
XE2 = sala ,
...
no
```

Ukupno sinonima mešovitih objekata-veza za ontološku ocenu: **0**

17. (Testiranje pravila R 5.17) Za koje ontološke nadklase i podklase postoji definisana odgovarajuća IS_A hijerarhija između entiteta u modelu podataka?

```
?-ontosubclassisa(X,X1,X2).
X = konferencija
X1 = medjunarodna
X2 = nacionalna ,
X = konferencija
X1 = nacionalna
X2 = medjunarodna ,
no
```

Ukupno ontoloških nadklasa za ontološku ocenu: **1**, podklasa za ontološku ocenu: **2**

18. (Testiranje pravila R 5.18) Za koje ontološke nadklase i podklase postoji IS_A hijerarhija između entiteta u modelu podataka ukoliko nazivi nadklase ontologije neodgovaraju nazivima entiteta (sinonimi za nadklasu)?

```
?-ontosubclassisasin(XP,XC1,XC2,XE,XE1,XE2).
no
```

Ukupno sinonima ontoloških nadklasa za ontološku ocenu: **0**

19. (Testiranje pravila R 5.19) Za koje ontološke nadklase i podklase postoji IS_A hijerarhija između entiteta u konceptualnom modelu podataka i ukoliko nazivi nadklase ontologije neodgovaraju nazivima entiteta (sinonimi za podklase)?

```
?-ontosubclassisasinsub(X,X1,X2).

X = konferencija
X1 = medjunarodna
X2 = nacionalna ,    NE

X = konferencija
X1 = nacionalna
X2 = medjunarodna ,
no
```

Ukupno sinonima ontoloških podklasa za ontološku ocenu: **0**

20. (Testiranje pravila R 5.20) Za koje ontološke nadklase i podklase ne postoje definisane odgovarajuće IS_A hijerarhije između entiteta u konceptualnom modelu podataka?

```
?-ontosubclassnoisa(X,X1,X2).
no
```

Ukupan broj nepokrivenih ontoloških klasa/podklasa: **0/0**

21. (Testiranje pravila R 5.21) Da li postoje ontološke klase koje nisu pokrivene tipovima entiteta sa istim nazivom ali postoje tipovi entiteta sa sličnim nazivom pa se mogu odrediti sinonimi?

```
?-ontoclassentsin(X,Y).
no
```

Ukupno sinonima entiteta za ontološku ocenu: **0**

22. (Testiranje pravila R 5.22) Koliko ima parova ekvivalentnih ontoloških osobina objekata?

```
?-ontoeqvobjprop(X,Y).

X = je_iz
Y = nalazi_se ,

X = odrzava_se
Y = priredjuje ,
no
```

Ukupno ekivavelntnih osobina ontoloških objekata: **2**

23. (Testiranje pravila R 5.23) Koji opsezi ontoloških osobina objekata klasa odgovaraju kardinalitetima poveznika između tipova entiteta pokrivenih ontološkim klasama?

```
?-ontocard(XC,YOP,ZCD1,ZCD2).
XC = rad
YOP = napisao
ZCD1 = mincard1
ZCD2 = maxcardm ,
XC = ucesnik
YOP = napisao
ZCD1 = mincard1
ZCD2 = maxcardm ,
XC = konferencija
YOP = organizuje
ZCD1 = mincard1
ZCD2 = maxcard1 ,
XC = organizator
YOP = organizuje
ZCD1 = mincard1
ZCD2 = maxcardm ,
no
```

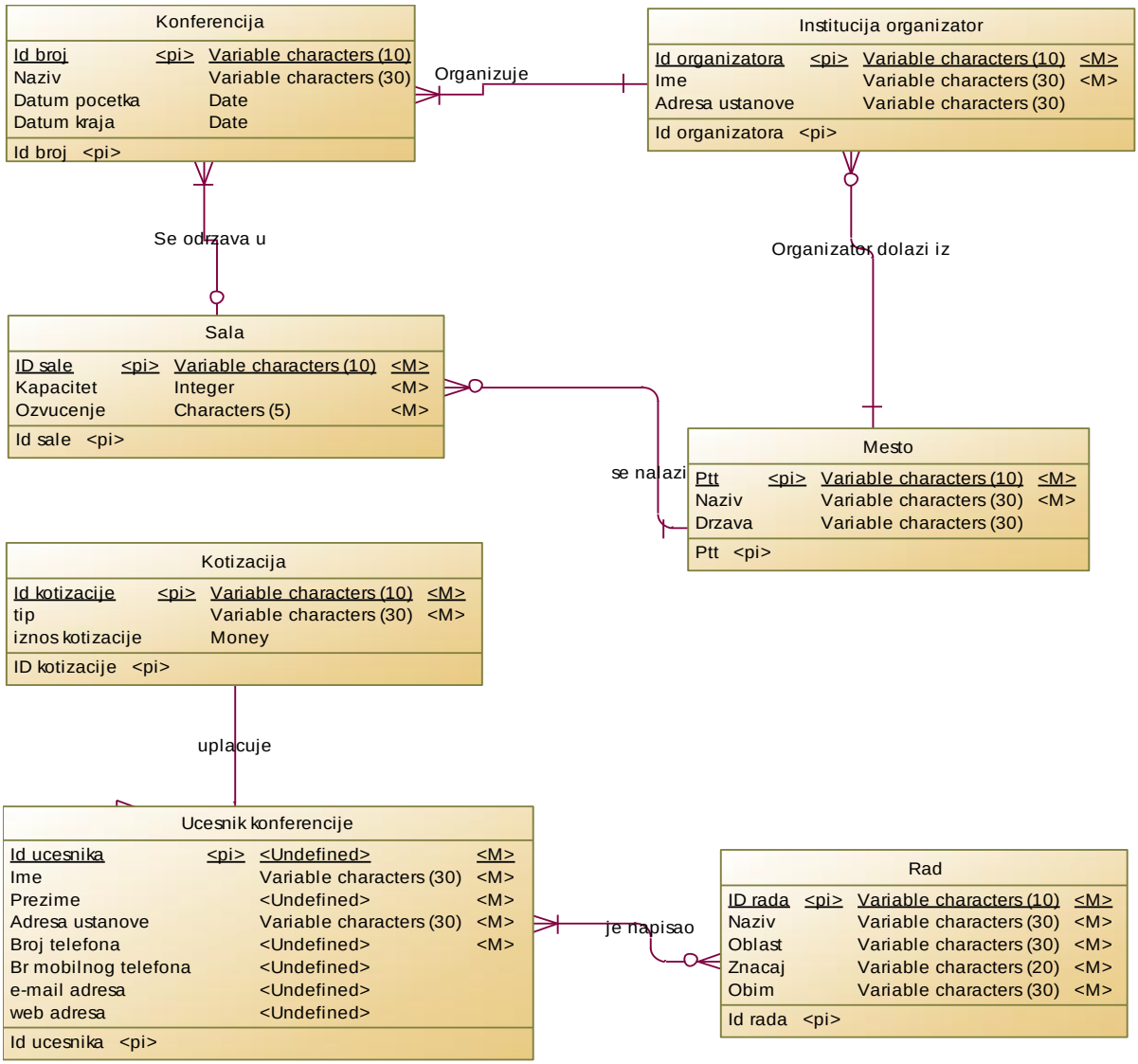
Ukupno ograničenja kardinaliteta poveznika za ontološku ocenu: 4

24. (Testiranje pravila R 5.24) Koji opsezi ontoloških osobina objekata klasa odgovaraju kardinalitetima sinonima poveznika između tipova entiteta pokrivenih ontološkim klasama?

```
?-ontocardsin(XC,YOP,ZCD1,ZCD2).
XC = ucesnik
YOP = izlaze
ZCD1 = mincard0
ZCD2 = maxcardm ,
XC = organizator
YOP = je_iz
ZCD1 = mincard1
ZCD2 = maxcard1 ,
XC = konferencija
YOP = odrzava_se
ZCD1 = mincard1
ZCD2 = maxcardm ,
XC = sala
YOP = odrzava_se
ZCD1 = mincard1
ZCD2 = maxcardm ,
no
```

Ukupno ograničenja kardinaliteta poveznika za ontološku ocenu: 4

Testiranje primera modela podataka - K01.CDM, za koji je šema modela podataka prikazana na slici 7.12.:



Slika 7.12. – Dijagram ER modela podataka K01.cdm

1. (Testiranje pravila R 5.1) Koje klase ontologije jesu pokrivene tipovima entiteta u konceptualnom modelu podataka?

```
?- ontoclassent(X).
X = konferencija ,
X = kotizacija ,
X = mesto ,
X = rad ,
X = sala ,
no
```

Ukupno entiteta za ontološku ocenu: **5**

2. (Testiranje pravila R 5.2) Koje klase ontologije nisu pokrivene tipovima entiteta u konceptualnom modelu podataka?

```
?- ontoclassnoent(X).
X = medjunarodna ,
X = nacionalna ,
X = organizator ,
X = ucesnik ,
no
```

Ukupno klasa: **4**

3. (Testiranje pravila R 5.3) Koje osobine podataka klasa ontologije jesu pokrivene atributima u konceptualnom modelu podataka?

```
?- ontodataatrib(X).
X = drzava ,
X = ime ,
X = kapacitet ,
X = naziv ,
X = obim ,
X = oblast ,
X = ozvucenje ,
X = prezime ,
X = ptt ,
X = znacaj ,
no
```

Ukupno atributa za ontološku ocenu: **10**

4. (Testiranje pravila R 5.4) Koje osobine podataka klasa ontologije nisu pokrivene atributima u konceptualnom modelu podataka?

```
?- ontodatanoatrib(X).
X = adresa ,
X = datum_izlaganja ,
X = datum_pocetka ,
X = datum_zavrsetka ,
X = email ,
X = id_konferencije ,
X = id_kotizacije ,
X = id_organizatora ,
X = id_rada ,
X = id_sale ,
X = id_ucesnika ,
X = iznos ,
X = ministarstvo ,
X = mobilni ,
X = naucni_odbor ,
X = telefon ,
X = tip_kotizacije ,
X = ustanova ,
X = vreme_izlaganja ,
X = web_adresa ,
no
```

Ukupno osobina podataka klasa: **20**

5. (Testiranje pravila R 5.5) Koje osobine objekata klasa ontologije jesu pokrivene relacijama u konceptualnom modelu podataka?

```
?- ontoobjproprel(X).
X = organizuje ,
no
```

Ukupno poveznika/relacija za ontološku ocenu: **1**

6. (Testiranje pravila R 5.6) Koje osobine objekata klasa ontologije nisu pokrivene relacijama u konceptualnom modelu podataka?

```
?-ontoobjpropnorel(X).

X = izlaze ,
X = je_iz ,
X = moze_biti ,
X = nalazi_se ,
X = napisao ,
X = odrzava_se ,
X = priredjuje ,
X = prisustvuje ,
X = uplatio ,
no
```

Ukupno osobina objekata klasa: **9**

7. (Testiranje pravila R 5.7) Za koje ontološke osobine podataka postoje odgovarajući atributi u konceptualnom modelu podataka takvi da su usklađeni i njihovi tipovi podataka?

```
?-ontodataatribtype(X,Y).

X = drzava
Y = string ,

X = ime
Y = string ,

X = naziv
Y = string ,
no
```

Ukupno atributa i tipova podataka za ontološku ocenu: **3**

8. (Testiranje pravila R 5.8) Za koje ontološke osobine podataka postoje atributi u konceptualnom modelu podataka takvi da su im nazivi različiti (sinonimi za atribute) ali su odgovarajućih tipova podataka?

```
?-ontodataatribtypesin(X,Y,X1,Y1).
```

```
X = adresa
Y = string
X1 = ime
Y1 = va ,
```

NE

```
X = adresa
Y = string
X1 = adresaustanove
Y1 = va ,
```

DA

```
...
X = datum_izlaganja
Y = datetime              NE
X1 = datumkraja
Y1 = d ,

X = datum_pocetka
Y = datetime              DA
X1 = datumpocetka
Y1 = d ,
...
X = znacaj
Y = string                NE
X1 = naziv
Y1 = va ,
no
```

Ukupno sinonima atributa u entitetima za ontološku ocenu: **6**

9. (Testiranje pravila R 5.9) Za koje skupove ontoloških osobina podataka objekata klasa postoje definisani odgovarajući atributi u okviru entiteta modela podataka?

```
?-ontoclassentdataatrib(X,Y).

X = konferencija
Y = naziv ,
X = mesto
Y = drzava ,
X = mesto
Y = naziv ,
X = mesto
Y = ptt ,
X = rad
Y = naziv ,
X = rad
Y = obim ,
X = rad
Y = oblast ,
X = rad
Y = znacaj ,
X = sala
Y = kapacitet ,
X = sala
Y = ozvucenje ,
no
```

Ukupno atributa u entitetima za ontološku ocenu: **10**

10. (Testiranje pravila R 5.10) Koje ontološke osobine objekata klasa odgovaraju poveznicima, tj. relacijama između tipova entiteta pokrivenih ontološkim klasama?

```
?-ontorel(XC1,YOP,XC2,XE1,YR,XE2).

no
```

Ukupno poveznika/relacija za ontološku ocenu: **0**

11. (Testiranje pravila R 5.11) Koje ontološke osobine objekata klasa se po nazivu razlikuju od naziva poveznika/relacija uspostavljenih između tipova entiteta pokrivenih ontološkim klasama (sinonimi za relacije)?

```
?-ontorelsinrel(XC1,YOP,XC2,XE1,YR,XE2).
```

```
XC1 = konferencija
YOP = odrzava_se
XC2 = sala
XE1 = konferencija
YR = seodrzavau
XE2 = sala ,
```

```
XC1 = sala
YOP = nalazi_se
XC2 = mesto
XE1 = sala
YR = senalaziu
XE2 = mesto ,
```

```
no
```

Ukupno poveznika/relacija koji imaju sinonime za ontološku ocenu: **2**

12. (Testiranje pravila R 5.12) Koje ontološke osobine objekata klasa odgovaraju poveznicima, tj. relacijama uspostavljenim između entiteta koji nisu pokriveni odgovaraju ontološkim klasama po nazivu (sinonimi za entitet)?

```
?-ontorelsinent(XC1,YOP,XC2,XE1,YR,XE2).
```

```
XC1 = konferencija
YOP = organizuje
XC2 = organizator
XE1 = konferencija
YR = organizuje
XE2 = institucijaorganizator ,
no
```

Ukupno poveznika/relacija gde postoje sinonimi kod jednog entiteta za ontološku ocenu: **1**

13. (Testiranje pravila R 5.13) Koje ontološke osobine objekata klasa odgovaraju poveznicima/relacijama uspostavljenim između entiteta koji nisu pokriveni odgovaraju ontološkim klasama po nazivu (sinonimi za oba entiteta)?

```
?-ontorelsinent2(XC1,YOP,XC2,XE1,YR,XE2).
```

```
no
```

Ukupno poveznika/relacija gde postoje sinonimi kod oba entiteta za ontološku ocenu: **0**

14. (Testiranje pravila R 5.14) Koje ontološke osobine objekata klasa se po nazivu razlikuju od relacija uspostavljenih između entiteta od kojih je samo jedan entitet pokriven odgovarajućom ontološkom klasom po nazivu, dok se drugi može razlikovati (sinonimi za relaciju i jedan od entiteta)?

```
?-ontorelsinentrel(XC1,YOP,XC2,XE1,YR,XE2).
```

```
XC1 = konferencija
YOP = odrzava_se
XC2 = mesto
XE1 = konferencija
YR = organizuje
XE2 = institucijaorganizator ,
```

NE

```
...
```

```
XC1 = konferencija
YOP = prisustvuje
XC2 = ucesnik
XE1 = konferencija              NE
YR = organizuje
XE2 = institucijaorganizator,
...
no
```

Ukupno poveznika/relacija gde postoje sinonimi kod jednog entiteta i poveznika za ontološku ocenu: **0**

15. (Testiranje pravila R 5.15) Da li postoje ontološke osobine objekata klasa koji odgovaraju mešovitim objektima veze, tj. gerundima u modelu podataka?

```
?-ontoobjpropger(XC1,YOP,XC2,XE1,YER,XE2).
no
```

Ukupno mešovitih objekata-veza za ontološku ocenu: **0**

16. (Testiranje pravila R 5.16) Da li postoje ontološke osobine objekata klase koje po nazivu ne odgovaraju mešovitim objektima veze, tj. gerundima u modelu podataka, ali su uspostavljene između entiteta za koje su definsane ontološke klase (sinonimi za gerunde)?

```
?-ontoobjpropgersin(XC1,YOP,XC2,XE1,YER,XE2).
no
```

Ukupno sinonima mešovitih objekata-veza za ontološku ocenu: **0**

17. (Testiranje pravila R 5.17) Za koje ontološke nadklase i podklase postoji definisana odgovarajuća IS_A hijerarhija između entiteta u modelu podataka?

```
?-ontosubclassisa(X,X1,X2).
no
```

Ukupno ontoloških nadklasa za ontološku ocenu: **0**, podklasa za ontološku ocenu: **0**

18. (Testiranje pravila R 5.18) Za koje ontološke nadklase i podklase postoji IS_A hijerarhija između entiteta u modelu podataka ukoliko nazivi nadklase ontologije neodgovaraju nazivima entiteta (sinonimi za nadklasu)?

```
?-ontosubclassisasin(XP,XC1,XC2,XE,XE1,XE2).
no
```

Ukupno sinonima ontoloških nadklasa za ontološku ocenu: **0**.

19. (Testiranje pravila R 5.19) Za koje ontološke nadklase i podklase postoji IS_A hijerarhija između entiteta u konceptualnom modelu podataka i ukoliko nazivi nadklase ontologije neodgovaraju nazivima entiteta (sinonimi za podklase)?

```
?-ontosubclassisasinsub(X,X1,X2).
no
```

Ukupno sinonima ontoloških podklasa za ontološku ocenu: **0**.

20. (Testiranje pravila R 5.20) Za koje ontološke nadklase i podklase ne postoje definisane odgovarajuće IS_A hijerarhije između entiteta u konceptualnom modelu podataka?

?-ontosubclassnoisa(X,X1,X2).

```
X = konferencija
X1 = medjunarodna
X2 = nacionalna ,

X = konferencija
X1 = nacionalna
X2 = medjunarodna ,

no
```

Ukupan broj nepokrivenih ontoloških klasa/podklasa: **1/2**

21. (Testiranje pravila R 5.21) Da li postoje ontološke klase koje nisu pokrivene tipovima entiteta sa istim nazivom ali postoje tipovi entiteta sa sličnim nazivom pa se mogu odrediti sinonimi?

?-ontoclassentsin(X,Y).

```
X = medjunarodna                 NE
Y = institucijaorganizator ,
...
X = organizator                  DA
Y = institucijaorganizator ,
X = organizator                  DA
Y = ucesnikkonferencije ,
...
X = ucesnik                      NE
Y = institucijaorganizator ,
no
```

Ukupno sinonima entiteta za ontološku ocenu: **2**

22. (Testiranje pravila R 5.22) Koliko ima parova ekvivalentnih ontoloških osobina objekata?

?-ontoeqvobjprop(X,Y).

```
X = je_iz
Y = nalazi_se ,

X = odrzava_se
Y = priredjuje ,
no
```

Ukupno ekivavelntnih osobina ontoloških objekata: **2**.

23. (Testiranje pravila R 5.23) Koji opsezi ontoloških osobina objekata klasa odgovaraju kardinalitetima poveznika između tipova entiteta pokrivenih ontološkim klasama?

?-ontocard(XC,YOP,ZCD1,ZCD2).

```
XC = konferencija
YOP = organizuje
ZCD1 = mincard1
ZCD2 = maxcard1 ,
no
```

Ukupno ograničenja kardinaliteta poveznika za ontološku ocenu: 1

24. (Testiranje pravila R 5.24) Koji opsezi ontoloških osobina objekata klasa odgovaraju kardinalitetima sinonima poveznika između tipova entiteta pokrivenih ontološkim klasama?

```
?- ontocardsin(XC,YOP,ZCD1,ZCD2).
XC = mesto
YOP = nalazi_se
ZCD1 = mincard0
ZCD2 = maxcardm ,
XC = sala
YOP = nalazi_se
ZCD1 = mincard1
ZCD2 = maxcard1 ,
XC = konferencija
YOP = odrzava_se
ZCD1 = mincard1
ZCD2 = maxcardm ,
XC = sala
YOP = odrzava_se
ZCD1 = mincard1
ZCD2 = maxcardm ,
no
```

Ukupno ograničenja kardinaliteta poveznika za ontološku ocenu: 4

7.5.1. Izračunavanje ontološke ocene za pojedinačan model podataka

Izračunavanje ukupne ontološke ocene semantičke korektnosti modela podataka vrši se prema metrici (5.1) i formulama (5.2), (5.3), (5.4) i (5.5), opisanim u poglavlju 5.7.

Testirani primer modela podatka **K06.CDM:**

Ontološka ocena entiteta:
OM_E = ((9+0)*100)/(9+0) = 100,0

Ontološka ocena atributa:
OM_A = ((20+(10+4)+24)/3*100)/(20+10) = 64,4

Ontološka ocena poveznika i gerunda:
OM_R = ((2+(4+0+0+0)+(1+0))*100/(1+9-2) +(4+4)*100/(1+9-2) *2)/2= 68,8

Ontološka ocena nadklasa i podklasa:
OM_{SC} = ((1+0)*100/(1+0)) + ((2+0)*100/(2+0)) = 100

Ukupna ontološka ocena modela podataka:
OM= (K_E*OM_E+ K_A*OM_A+ K_R*OM_R + K_{SC}*OM_{SC})/4

K_E=K_A=K_R=K_{SC}=1

OM=(100,0+64,4+68,8+100,0)/4

***OM*=83,3**

Testirani primer modela podatka **K01.CDM:**

Ontološka ocena entiteta:
OM_E = ((5+2)*100)/(5+4) = 77,8

Ontološka ocena atributa:
OM_A = ((10+(3+6)+10)/3*100)/(10+20) = 32,2

Ontološka ocena poveznika i gerunda:
OM_R = ((0+(2+1+0+0)+(0+0))*100/(1+9-2) +(1+4)*100/(1+9-2) *2)/2=34,4

Ontološka ocena nadklasa i podklasa:
OM_{SC} = ((0+0)*100/(1+0)) + ((0+0)*100/(2+0)) = 0

Ukupna ontološka ocena modela podataka:
OM= (K_E*OM_E+ K_A*OM_A+ K_R*OM_R + K_{SC}*OM_{SC})/4

K_E=K_A=K_R=K_{SC}=1

OM=(77,8+32,2+34,4+0)/4

***OM*=36,1**

7.6. REZULTATI EMPIRIJSKOG ISTRAŽIVANJA

U tabeli 7.2 su prikazani rezultati testiranja pravila zaključivanja koji se odnose na semantiku entiteta i njihove usklađenosti sa ontološkim klasama.

Tabela 7.2: Ontološko vrednovanje entiteta

Oznaka modela podataka	1.	2.	3.	4.	5.	6.	Ontološka ocena entiteta u modelu podataka (%)
K01	5	4	2	2	7	9	**77,8**
K02	7	2	1	1	8	9	**88,9**
K03	7	2	2	0	9	9	**100,0**
K04	7	2	0	2	7	9	**77,8**
K05	9	0	0	0	9	9	**100,0**
K06	9	0	0	0	9	9	**100,0**
K07	5	4	2	2	7	9	**77,8**
K08	8	1	1	0	9	9	**100,0**
K09	9	0	0	0	9	9	**100,0**
K10	9	0	0	0	9	9	**100,0**
K11	7	2	1	1	8	9	**88,9**
K12	8	1	0	1	8	9	**88,9**
K13	9	0	0	0	9	9	**100,0**
K14	5	4	4	0	9	9	**100,0**
K15	5	4	4	0	9	9	**100,0**
K16	9	0	0	0	9	9	**100,0**
K17	9	0	0	0	9	9	**100,0**
K18	6	3	3	0	9	9	**100,0**
K19	6	3	3	0	9	9	**100,0**
K20	5	4	4	0	9	9	**100,0**
K21	6	3	0	3	6	9	**66,7**
K22	5	4	4	0	9	9	**100,0**
K23	8	1	1	0	9	9	**100,0**
K24	7	2	2	0	9	9	**100,0**
K25	5	4	4	0	9	9	**100,0**
K26	7	2	0	2	7	9	**77,8**
K27	6	3	3	0	9	9	**100,0**
K28	6	3	3	0	9	9	**100,0**
K29	7	2	2	0	9	9	**100,0**
K30	5	4	2	2	7	9	**77,8**
K31	7	2	2	0	9	9	**100,0**
K32	6	3	1	2	7	9	**77,8**
K33	6	3	1	2	7	9	**77,8**
K34	8	1	0	1	8	9	**88,9**
K35	8	1	0	1	8	9	**88,9**
K36	5	4	2	2	7	9	**77,8**
K37	5	4	1	3	6	9	**66,7**
K38	8	1	1	0	9	9	**100,0**
K39	5	4	4	0	9	9	**100,0**
K40	5	4	4	0	9	9	**100,0**
K41	5	4	4	0	9	9	**100,0**
K42	9	0	0	0	9	9	**100,0**
K43	6	3	1	2	7	9	**77,8**
K44	5	4	4	0	9	9	**100,0**
Prosek:	**6,68**	**2,32**	**1,66**	**0,66**	**8,34**	**9,00**	**92,68**

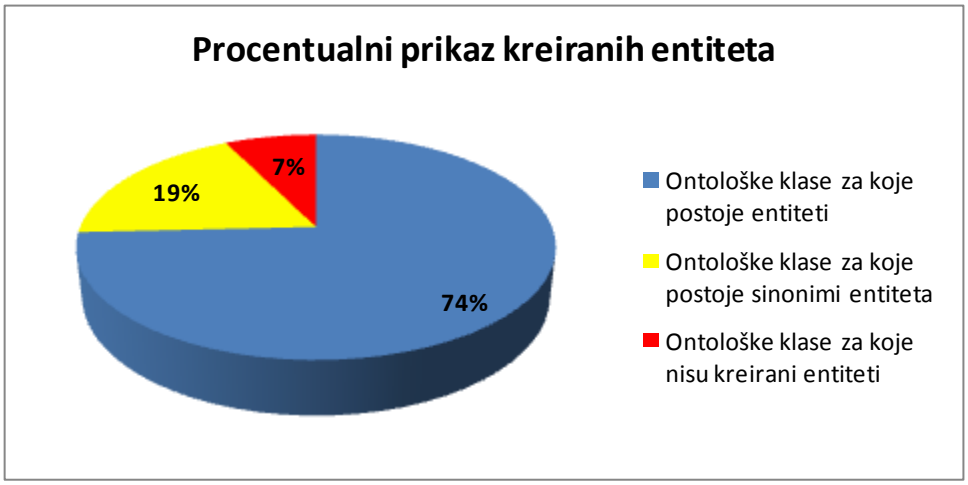
Značenje pojedinih kolona tabele:

1. Broj ontoloških klasa za koje su kreirani entiteti (na osnovu pravila R 5.1)
2. Broj ontoloških klasa za koje nisu kreirani entiteti (na osnovu pravila R 5.2)
3. Broj ontoloških klasa koje su pokrivene sinonimima entiteta (na osnovu pravila R 5.21)
4. Ukupan broj ontoloških klasa nepokrivenih entitetima (2.-3.)
5. Ukupan broj ontoloških klasa pokrivenih entitetima (1.+3.)
6. Ukupno klasa u ontologiji (1.+2.)

Na slici 7.13. prikazana je ukupna pokrivenost ontoloških klasa entitetima u svim testiranim modelima podataka.

Slika 7.13. – Ukupna pokrivenost ontoloških klasa entitetima

Procentualni prikaz kreiranih entiteta u modelu podataka u odnosu na ontološke klase je dat na slici 7.14.



Slika 7.14. – Procentualni prikaz kreiranih entiteta u modelu podataka

U tabeli 7.3 su prikazani rezultati testiranja pravila zaključivanja koji se odnose na semantiku atributa i njihovu usklađenost sa ontološkim osobinama podataka objekata kao instanci klasa.

Tabela 7.3: Ontološko vrednovanje atributa

Oznaka modela podataka	1.	2.	3.	4.	5.	6.	7.	8.	Ontološka ocena atributa u modelu podataka
K01	10	20	3	6	9	10	21	30	**32,2**
K02	11	19	4	13	17	12	13	30	**44,4**
K03	17	13	8	8	16	18	14	30	**56,7**
K04	21	9	11	6	17	22	13	30	**66,7**
K05	18	12	6	7	13	20	17	30	**56,7**
K06	20	10	10	4	14	24	16	30	**64,4**
K07	17	13	7	4	11	14	19	30	**46,7**
K08	17	13	14	9	23	11	7	30	**56,7**
K09	24	6	10	1	11	26	19	30	**67,8**
K10	20	10	9	3	12	22	18	30	**60,0**
K11	18	12	8	6	14	13	16	30	**50,0**
K12	15	15	9	7	16	17	14	30	**53,3**
K13	16	14	5	4	9	19	21	30	**48,9**
K14	22	8	10	2	12	17	18	30	**56,7**
K15	11	19	5	11	16	6	14	30	**36,7**
K16	14	16	4	1	5	12	25	30	**34,4**
K17	20	10	12	5	17	19	13	30	**62,2**
K18	21	9	11	4	15	24	15	30	**66,7**
K19	22	8	10	4	14	23	16	30	**65,6**
K20	15	15	2	4	6	10	24	30	**34,4**
K21	15	15	8	4	12	11	18	30	**42,2**
K22	22	8	10	3	13	15	17	30	**55,6**
K23	17	13	4	2	6	16	24	30	**43,3**
K24	16	14	6	1	7	13	23	30	**40,0**
K25	11	19	5	13	18	6	12	30	**38,9**
K26	21	9	2	3	5	21	25	30	**52,2**
K27	12	18	4	8	12	7	18	30	**34,4**
K28	13	17	4	9	13	7	17	30	**36,7**
K29	20	10	10	3	13	20	17	30	**58,9**
K30	14	16	7	5	12	14	18	30	**44,4**
K31	18	12	8	6	14	18	16	30	**55,6**
K32	12	18	4	4	8	12	22	30	**35,6**
K33	16	14	7	2	9	14	21	30	**43,3**
K34	18	12	11	2	13	18	17	30	**54,4**
K35	19	11	11	2	13	18	17	30	**55,6**
K36	14	16	5	7	12	14	18	30	**44,4**
K37	17	13	5	2	7	14	23	30	**42,2**
K38	22	8	11	2	13	21	17	30	**62,2**
K39	16	14	8	6	14	10	16	30	**44,4**
K40	12	18	5	9	14	6	16	30	**35,6**
K41	12	18	1	5	6	6	24	30	**26,7**
K42	20	10	12	5	17	19	13	30	**62,2**
K43	14	16	11	3	14	18	16	30	**51,1**
K44	17	13	8	7	15	10	15	30	**46,7**
Prosek:	**16,75**	**13,25**	**7,39**	**5,05**	**12,43**	**15,16**	**17,57**	**30,00**	**49,27**

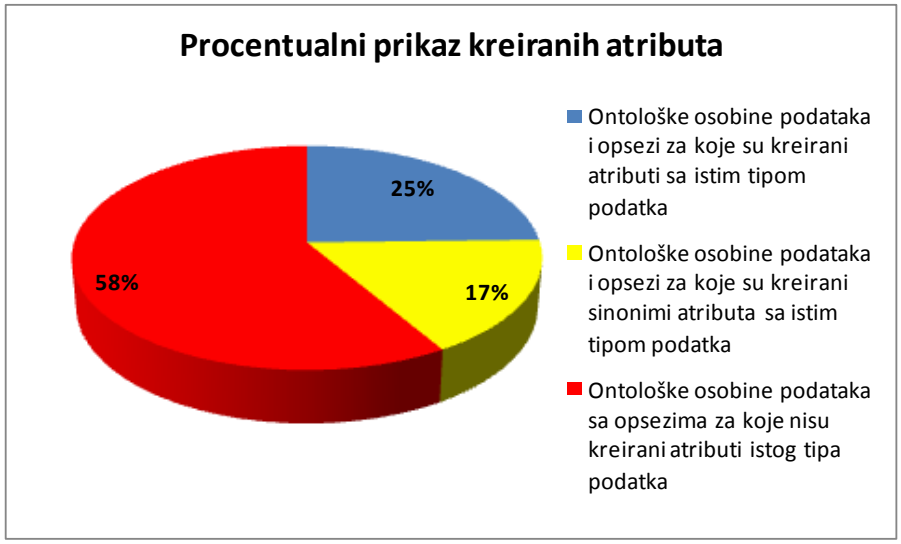
Značenje pojedinih kolona tabele:

1. Broj ontoloških osobina podataka za koje su kreirani atributi (na osnovu pravila R 5.3)
2. Broj ontoloških osobina podataka za koje nisu kreirani atributi (na osnovu pravila R 5.4)
3. Ontološke osobine podataka i opsezi za koje su kreirani atributi sa istim tipom podatka (na osnovu pravila R 5.7)
4. Ontološke osobine podataka i opsezi za koje su kreirani sinonimi atributa sa istim tipom podatka (na osnovu pravila R 5.8)
5. Ukupno ontoloških osobina podataka i opsega za koje su kreirani atributi sa istim tipom podatka (3.+4.)
6. Ontološke osobine podataka u okviru klasa za koje su kreirani atributi u entitetima (na osnovu pravila R 5.9)
7. Ontološke osobine podataka sa opsezima za koje nisu kreirani atributi odgovarajućeg tipa podatka (1.+2.-5.)
8. Ukupno ontoloških osobina podataka u ontologiji (1.+2.)

Na slici 7.15. prikazana je ukupna pokrivenost ontoloških osobina podataka atributima u svim testiranim modelima podataka.

Slika 7.15. Ukupna pokrivenost ontoloških osobina podataka atributima

Procentualni prikaz kreiranih atributa u modelu podataka u odnosu na ontološke osobine podataka je dat na slici 7.16.



Slika 7.16. – Procentualni prikaz kreiranih atributa u modelu podataka

U tabeli 7.4 su prikazani rezultati testiranja pravila zaključivanja koji se odnose na semantiku relacija između entiteta i mešovitih objekata-veza, tj. gerunda i njihove usklađenosti sa ontološkim osobinama objekata kao instanci klasa.

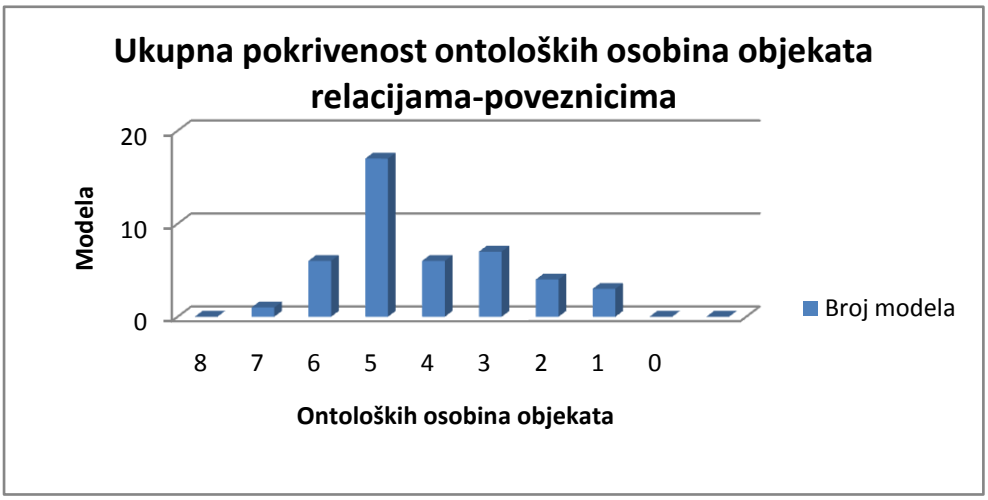
Tabela 7.4: Ontološko vrednovanje relacija i gerunda (mešovitih objekata-veza)

Oznaka modela podataka	1.	2.	3.	4.	5.	6.	7.	8.	9.	10.	11.	Ontološka ocena relacija u modelu podataka
K01	1	9	0	3	0	3	5	1	4	2	8	**34,4**
K02	2	8	2	4	0	6	2	4	3	2	8	**59,4**
K03	6	4	5	2	0	7	1	10	2	2	8	**81,3**
K04	2	8	1	3	1	5	3	2	4	2	8	**50,0**
K05	1	9	1	3	2	6	2	2	6	2	8	**62,5**
K06	2	8	2	4	1	7	1	4	4	2	8	**68,8**
K07	3	7	0	5	0	5	3	1	2	2	8	**40,6**
K08	2	8	1	3	2	6	2	2	6	2	8	**62,5**
K09	4	6	4	2	1	7	1	8	2	2	8	**75,0**
K10	4	6	4	3	1	8	0	4	5	2	8	**78,1**
K11	4	6	1	4	0	5	3	0	0	2	8	**31,3**
K12	2	8	2	4	0	6	2	3	1	2	8	**50,0**
K13	0	10	0	6	0	6	2	0	2	2	8	**43,8**
K14	2	8	1	2	2	5	3	2	1	2	8	**40,6**
K15	3	7	2	3	2	7	1	4	2	2	8	**62,5**
K16	5	5	4	2	0	6	2	1	2	2	8	**46,9**
K17	3	7	2	4	0	6	2	2	4	2	8	**56,3**
K18	1	9	0	2	0	2	6	0	3	2	8	**21,9**
K19	2	8	1	3	0	4	4	3	3	2	8	**43,8**
K20	0	10	0	4	0	4	4	0	1	2	8	**28,1**
K21	0	10	0	2	0	2	6	0	3	2	8	**21,9**
K22	2	8	1	4	2	7	1	2	1	2	8	**53,1**
K23	3	7	1	5	0	6	2	4	3	2	8	**59,4**
K24	5	5	1	3	2	6	2	4	2	2	8	**56,3**
K25	3	7	1	3	2	6	2	4	2	2	8	**56,3**
K26	2	8	1	3	0	4	4	2	4	2	8	**43,8**
K27	2	8	2	3	1	6	2	4	3	2	8	**59,4**
K28	2	8	2	2	1	5	3	4	4	2	8	**56,3**
K29	1	9	1	3	3	7	1	2	4	2	8	**62,5**
K30	1	9	0	4	0	4	4	1	7	2	8	**50,0**
K31	5	5	3	3	0	6	2	5	4	2	8	**65,6**
K32	2	8	1	2	3	6	2	2	2	2	8	**50,0**
K33	1	9	1	3	0	4	4	2	3	2	8	**40,6**
K34	0	10	0	3	0	3	5	0	0	2	8	**18,8**
K35	0	10	0	3	0	3	5	0	0	2	8	**18,8**
K36	1	9	0	4	0	4	4	1	7	2	8	**50,0**
K37	1	9	0	3	1	4	4	1	4	2	8	**40,6**
K38	2	8	1	3	2	6	2	3	6	2	8	**65,6**
K39	0	10	0	5	0	5	3	0	1	2	8	**34,4**
K40	3	7	1	3	2	6	2	4	2	2	8	**56,3**
K41	3	7	1	3	2	6	2	4	2	2	8	**56,3**
K42	3	7	2	4	0	6	2	2	4	2	8	**56,3**
K43	0	10	0	3	0	3	5	0	0	2	8	**18,8**
K44	0	10	0	2	0	2	6	0	1	2	8	**15,6**
Prosek:	**2,07**	**7,93**	**1,20**	**3,23**	**0,75**	**5,18**	**2,82**	**2,36**	**2,86**	**2,00**	**8,00**	**48,72**

Značenje pojedinih kolona tabele:

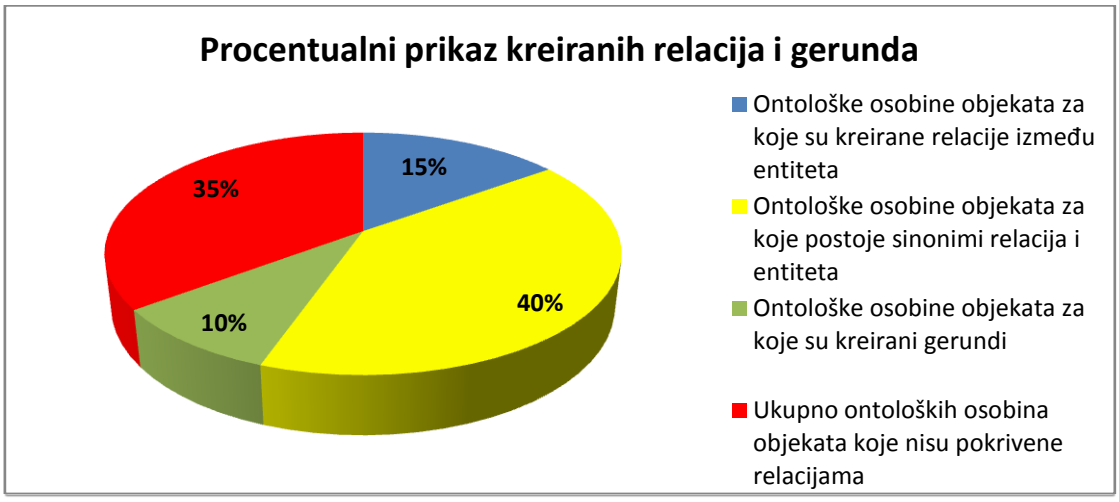
1. Broj ontoloških osobina objekata za koje su kreirane relacije (na osnovu pravila R 5.5)
2. Broj ontoloških osobina objekata za koje nisu kreirane relacije (na osnovu pravila R 5.6)
3. Broj ontoloških osobina objekata za koje su kreirane relacije između entiteta koje odgovaraju ontološkim klasama (na osnovu pravila R 5.10)
4. Broj ontoloških osobina objekata za koje postoje sinonimi relacija i entiteta (na osnovu pravila R 5.11, R 5.12, R 5.13 i R 5.14)
5. Broj ontoloških osobina objekata za koje su kreirani gerundi (na osnovu pravila R 5.15 i R 5.16)
6. Ukupno relacija/gerunda koji semantički odgovaraju ontologiji (3.+4.+5.)
7. Ukupno ontoloških osobina objekata koje nisu pokrivene relacijama (1.+2.-8.-6.)
8. Broj opsega ontoloških osobina objekata koje odgovaraju kardinalitetima poveznika (pravilo R.23)
9. Broj opsega ontoloških osobina objekata koje odgovaraju kardinalitetima sinonima poveznika (pravilo R.24)
10. Broj parova ekvivalentnih ontoloških osobina objekata (na osnovu pravila R 5.22)
11. Ukupno osobina objekata u ontologiji (1.+2.-8.)

Na sledećem dijagramu (Slika 7.17.) prikazana je ukupna pokrivenost ontoloških osobina objekata poveznicima i gerundima u svim testiranim modelima podataka.



Slika 7.17. Ukupna pokrivenost ontoloških osobina objekata poveznicima

Procentualni prikaz kreiranih poveznika i gerunda u modelu podataka u odnosu na ontološke osobine objekata je dat na slici 7.18.



Slika 7.18. - Procentualni prikaz kreiranih relacija i gerunda u modelu podataka

U tabeli 7.5 su prikazani rezultati testiranja pravila zaključivanja koji se odnose na semantiku IS_A hijerarhije u modelu podataka i usklađenosti supertipova i podtipova entiteta sa ontološkim nadklasama i podklasama.

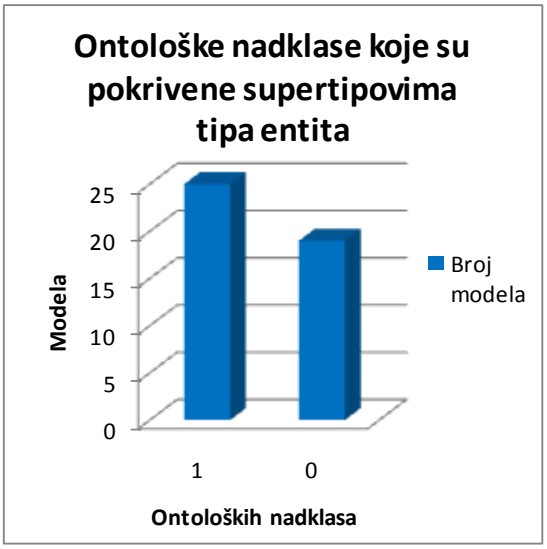
Tabela 7.5: Ontološko vrednovanje nadklasa/podklasa

Redni broj modela podataka	1.	2.	3.	4.	5.	6.	7.	8.	9.	10.	Ontološka ocena nadklasa i podklasa u modelu podataka
K01	0	0	1	2	0	0	0	0	1	2	**0,0**
K02	0	1	1	1	0	0	0	1	1	2	**25,0**
K03	1	0	0	2	0	2	1	2	1	2	**100,0**
K04	0	0	1	2	0	0	0	0	1	2	**0,0**
K05	1	2	0	0	0	0	1	2	1	2	**100,0**
K06	1	2	0	0	0	0	1	2	1	2	**100,0**
K07	0	0	1	2	0	0	0	0	1	2	**0,0**
K08	1	2	0	0	0	0	1	2	1	2	**100,0**
K09	1	2	0	0	0	0	1	2	1	2	**100,0**
K10	1	2	0	0	0	0	1	2	1	2	**100,0**
K11	0	0	1	2	0	0	0	0	1	2	**0,0**
K12	1	2	0	0	0	0	1	2	1	2	**100,0**
K13	0	0	1	2	0	0	0	0	1	2	**0,0**
K14	0	0	1	2	1	2	1	2	1	2	**100,0**
K15	1	0	0	2	0	0	1	0	1	2	**50,0**
K16	1	2	0	0	0	0	1	2	1	2	**100,0**
K17	1	2	0	0	0	0	1	2	1	2	**100,0**
K18	0	0	1	2	1	2	1	2	1	2	**100,0**
K19	0	0	1	2	1	2	1	2	1	2	**100,0**
K20	0	0	1	2	0	0	0	0	1	2	**0,0**
K21	1	2	0	0	0	0	1	2	1	2	**100,0**
K22	0	0	1	2	0	0	0	0	1	2	**0,0**
K23	1	2	0	0	0	0	1	2	1	2	**100,0**
K24	1	2	0	0	0	0	1	2	1	2	**100,0**
K25	0	0	1	2	1	2	1	2	1	2	**100,0**
K26	0	0	1	2	0	0	0	0	1	2	**0,0**
K27	1	0	0	2	0	2	1	2	1	2	**100,0**
K28	0	0	1	2	1	2	1	2	1	2	**100,0**
K29	0	0	1	2	1	2	1	2	1	2	**100,0**
K30	0	0	1	2	0	0	0	0	1	2	**0,0**
K31	1	1	0	1	0	1	1	2	1	2	**100,0**
K32	0	0	1	2	0	0	0	0	1	2	**0,0**
K33	0	0	1	2	0	0	0	0	1	2	**0,0**
K34	0	0	1	2	0	0	0	0	1	2	**0,0**
K35	0	0	1	2	0	0	0	0	1	2	**0,0**
K36	0	0	1	2	0	0	0	0	1	2	**0,0**
K37	0	0	1	2	0	0	0	0	1	2	**0,0**
K38	1	2	0	0	0	0	1	2	1	2	**100,0**
K39	0	0	1	2	0	0	0	0	1	2	**0,0**
K40	0	0	1	2	1	2	1	2	1	2	**100,0**
K41	1	2	0	0	0	0	1	2	1	2	**100,0**
K42	1	2	0	0	0	0	1	2	1	2	**100,0**
K43	0	0	1	2	0	0	0	0	1	2	**0,0**
K44	0	0	1	2	0	0	0	0	1	2	**0,0**
Prosek:	**0,41**	**0,68**	**0,59**	**1,30**	**0,16**	**0,43**	**0,57**	**1,11**	**1,00**	**2,00**	**56,25**

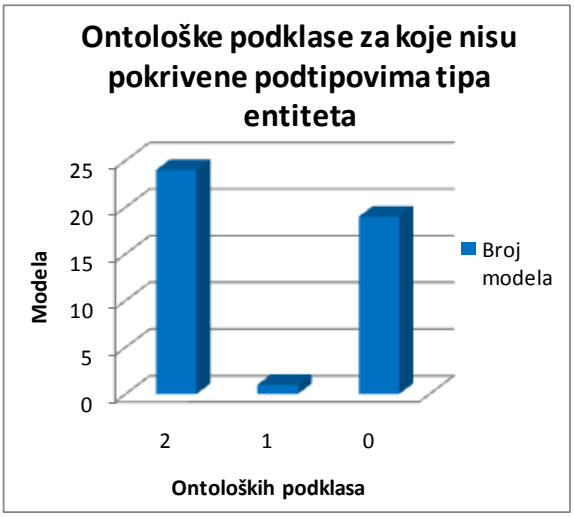
Značenje pojedinih kolona tabele:

1. Broj ontoloških nadklasa za koje su kreirani supertipovi entiteta IS_A hijerarhije (na osnovu pravila R 5.17)
2. Broj ontoloških podklasa za koje su kreirani podtipovi entiteta (na osnovu pravila R 5.17)
3. Broj nadklasa za koje nije kreirana superklasa IS_A hijerarhije (na osnovu pravila R 5.20)
4. Broj podklasa za koje nije kreirana podklasa IS_A hijerarhije (na osnovu pravila R 5.20)
5. Broj ontoloških nadklasa za koje postoje sinonimi supertipa entiteta IS_A hijerarhije (na osnovu pravila R 5.18)
6. Broj ontoloških podklasa za koje postoje sinonimi podtipa entiteta IS_A hijerarhije (na osnovu pravila R 5.18)
7. Ukupno ontoloških nadklasa koje su pokrivene supertipom entiteta IS_A hijerarhije (1.+5.)
8. Ukupno ontoloških podklasa koje su pokrivene podtipom entiteta IS_A hijerarhije (2.+6.)
9. Ukupno nadklasa u ontologiji (1.+3.)
10. Ukupno podklasa u ontologiji (2.+4.)

Procentualni prikaz kreiranih elemenata IS_A hijerarhije u modelu podataka u odnosu na ontološke nadklase i podklase je dat na dijagramima u okviru slika 7.19. i 7.20.



Slika 7.19. - Ukupna pokrivenost ontoloških nadklasa odgovarajućim supertipovima tipa entiteta



Slika 7.20. - Ukupna pokrivenost ontoloških podklasa odgovarajućim podtipovima tipa entiteta

U tabeli 7.6 su prikazani rezultati testiranja svih modela podataka i njihove usklađenosti sa domenskom ontologijom. Ukupna ontološka ocena svakog pojedinačnog modela podataka izračunata je kao prosečna vrednost ontoloških ocena pojedinih elemenata modela podataka (iz tabela 7.2, 7.3, 7.4 i 7.5), a prema formuli (5.1).

Tabela 7.6: Ukupna ontološka ocena semantičke korektnosti modela podataka

Oznaka modela podataka	Ontološka ocena entiteta u modelu podataka	Ontološka ocena atributa u modelu podataka	Ontološka ocena poveznika/gerunda u modelu podataka	Ontološka ocena IS_A hijerarhije u modelu podataka	UKUPNA ontološka ocena modela podataka
K01	77,8	32,2	34,4	0,0	**36,1**
K02	88,9	44,4	59,4	25,0	**54,4**
K03	100,0	56,7	81,3	100,0	**84,5**
K04	77,8	66,7	50,0	0,0	**48,6**
K05	100,0	56,7	62,5	100,0	**79,8**
K06	100,0	64,4	68,8	100,0	**83,3**
K07	77,8	46,7	40,6	0,0	**41,3**
K08	100,0	56,7	62,5	100,0	**79,8**
K09	100,0	67,8	75,0	100,0	**85,7**
K10	100,0	60,0	78,1	100,0	**84,5**
K11	88,9	50,0	31,3	0,0	**42,5**
K12	88,9	53,3	50,0	100,0	**73,1**
K13	100,0	48,9	43,8	0,0	**48,2**
K14	100,0	56,7	40,6	100,0	**74,3**
K15	100,0	36,7	62,5	50,0	**62,3**
K16	100,0	34,4	46,9	100,0	**70,3**
K17	100,0	62,2	56,3	100,0	**79,6**
K18	100,0	66,7	21,9	100,0	**72,1**
K19	100,0	65,6	43,8	100,0	**77,3**
K20	100,0	34,4	28,1	0,0	**40,6**
K21	66,7	42,2	21,9	100,0	**57,7**
K22	100,0	55,6	53,1	0,0	**52,2**
K23	100,0	43,3	59,4	100,0	**75,7**
K24	100,0	40,0	56,3	100,0	**74,1**
K25	100,0	38,9	56,3	100,0	**73,8**
K26	77,8	52,2	43,8	0,0	**43,4**
K27	100,0	34,4	59,4	100,0	**73,5**
K28	100,0	36,7	56,3	100,0	**73,2**
K29	100,0	58,9	62,5	100,0	**80,3**
K30	77,8	44,4	50,0	0,0	**43,1**
K31	100,0	55,6	65,6	100,0	**80,3**
K32	77,8	35,6	50,0	0,0	**40,8**
K33	77,8	43,3	40,6	0,0	**40,4**
K34	88,9	54,4	18,8	0,0	**40,5**
K35	88,9	55,6	18,8	0,0	**40,8**
K36	77,8	44,4	50,0	0,0	**43,1**
K37	66,7	42,2	40,6	0,0	**37,4**
K38	100,0	62,2	65,6	100,0	**82,0**
K39	100,0	44,4	34,4	0,0	**44,7**
K40	100,0	35,6	56,3	100,0	**73,0**
K41	100,0	26,7	56,3	100,0	**70,7**
K42	100,0	62,2	56,3	100,0	**79,6**
K43	77,8	51,1	18,8	0,0	**36,9**
K44	100,0	46,7	15,6	0,0	**40,6**
Prosek:	**92,68**	**49,27**	**48,72**	**56,25**	**61,73**

Tabela 7.7: Ukupna semantička korektnost pojedinih elemenata modela podataka

Opis elementa analize	Prosečno elemenata po modelu	Ukupno elementa u ontologiji	Procenat ontološke pokrivenosti (%)
Entiteti			
Broj ontoloških klasa pokrivenih entitetima	8,34	9	**92,68**
Broj ontoloških klasa nepokrivenih entitetima	0,66	9	**7,32**
Atributi			
Broj ontoloških osobina podataka i opsega za koje su kreirani atributi sa istim tipom podatka	12,43	30	**41,43**
Broj ontoloških osobina podataka i opsega za koje su kreirani atributi u entitetima	15,16	30	**50,53**
Broj ontoloških osobina podataka i opsega za koje nisu kreirani atributi odgovarajućeg tipa podatka	17,57	30	**58,57**
Relacije (poveznici) i gerundi			
Broj relacija/gerunda koji semantički odgovaraju ontologiji	5,18	10-2=8	**64,75**
Broj ontoloških osobina objekata koje nisu pokrivene relacijama	2,82	10-2=8	**35,25**
Broj opsega ontoloških osobina objekata sa odgovarajućim kardinalitetima poveznika	5,23	8*2=16	**32,69**
IS_A hijerarhija			
Broj ontoloških nadklasa koje su pokrivene supertipom entiteta is_a hijerarhije	0,57	1	**57,00**
Broj ontoloških nadklasa koje nisu pokrivene supertipom entiteta is_a hijerarhije	0,43	1	**43,00**
Broj ontoloških podklasa koje su pokrivene podtipom entiteta is_a hijerarhije	1,11	2	**55,50**
Broj ontoloških podklasa koje nisu pokrivene podtipom entiteta is_a hijerarhije	0,89	2	**44,50**

7.6.1. Analiza rezultata empirijskog istraživanja

Iz prethodno prikazanih rezultata empirijskog istraživanja vidi se da je ono podeljeno u šest celina. Prva četiri dela, prikazanih u tabelama 7.2, 7.3, 7.4 i 7.5, prikazuju rezultate testiranja pravila zaključivanja integrisanog formalizovanog modela podataka i mapirane domenske ontologije na osnovnim elementima ER modela podataka: entitetima, poveznicima i atributima, kao i gerundima i IS_A hijerarhijom, konceptima iz proširenja ovog modela. Tabela 7.7 sadrži ukupne ontološke ocene svakog prethodno navedenog elementa ER modela podataka i ukupnu ontološku ocenu svih testiranih modela podataka. Tabela 7.6 prikazuje ukupnu semantičku korektnost pojedinih elemenata modela podataka.

Uzorak na kom je obavljeno testiranje modela se sastoji od 44 rada, tj. konceptualnih modela podataka studenata koji su označeni sa K01, K02, ... K44.

Na osnovu podataka iz tabele 7.2, koja prikazuje ontološko vrednovanje entiteta ER modela podataka, vidi se da je prosečan broj entiteta koji po nazivu odgovaraju semantičkim klasama 6,68 od 9, koliko ukupno ima klasa u ontologiji. Ovo čini približno 74% ontoloških klasa za koje postoje entiteti u modelu podataka (slika 7.14). Prosečno 2,32 od 9 klasa (25,78%) nema odgovarajuće entitete, ali se može uočiti da je za 1,66 klasa (≈19%) utvrđeno da postoje sinonimi entiteta, što ukupno iznosi prosečno 8,34 od 9 klasa, po modelu, koje su semantički usklađene i pokrivene (92,68%). 0,66 ontoloških klasa ili prosečno 7,33% nema svoj par u formi tipa entiteta u modelu podataka. Iz dijagrama na slici 7.13. vidi se da je u 27 radova, svih 9 klasa pokriveno entitetima, u 5 radova je to 8 klasa, u 9 radova je 7, dok je u 2 rada to 6 klasa. Nijedan student nije formirao korektno manje od 6 entiteta. Najniža ontološka ocena entiteta iznosi 66,7% usklađenosti sa domenskim klasama ontologije (u dva rada), dok je u čak 27 modela ova vrednost maksimalna i iznosi 100%. Ukupna prosečna ontološka ocena entiteta svih modela podataka je 92,68% usklađenosti sa domenskim klasama ontologije.

Podaci iz tabele 7.3 ilustruju ontološko vrednovanje atributa ER modela podataka. Može se videti da je prosečan broj atributa koji po nazivu odgovaraju semantičkim osobinama podataka objekata klasa 16,75 od 30, koliko ukupno ima osobina podataka u ontologiji. Procentualno, ovo iznosi 55,83% ontoloških osobina podataka za koje postoje atributi u modelu podataka. Prosečno 13,25 od 30 osobina podataka nema odgovarajuće atribute (44,17%). Kada se uporede ontološke osobine podataka i opsezi za koje su kreirani atributi sa istim tipom podatka, vidimo da 7,39 od 30 ontoloških osobina objekata jeste pokriveno atributima sa istim tipom podatka (24,63%). Za 5,05 od 30 ontoloških osobina objekata (16,83%) postoje sinonimi atributa sa odgovarajućim tipom podatka, dok prosečno 12,43 ontoloških osobina objekata odgovara atributima u okviru entiteta kojima isti pripadaju (41,43%). Ukupan broj ontoloških osobina podataka i opsega koji nisu pokriveni atributima odgovarajućeg tipa podatka je prosečno 17,57 po modelu od maksimalno 30 ili približno 58%, kao što je i prikazano na slici 7.16. Za približno 42% atributa i njihovih tipova podataka se može reći da semantički odgovaraju ontologiji, tj. osobinama podataka ojekata klasa. Iz dijagrama prikazanog na

slici 7.15. vidi se da je u 4 rada 20 ontoloških osobina podataka pokriveno atributima, što je najveća vrednost. Zatim su u još 4 rada 19 osobina podataka pokrivena atributima, u 2 rada je to 18 itd. sve do samo jednog modela podataka u kojem je samo 8 atributa semantički pokriveno, što je i najniža vrednost u analizi. Nijedan student nije formirao, u modelu podataka, korektno manje od 8 atributa ali ni više od 20. Sve vrednosti se kreću između 8 i 20, od maksimalno 30 osobina podataka pokrivenih atributima. Najniža ontološka ocena atributa, u tabeli 7.3, iznosi 27% usklađenosti sa domenskim osobinama objekata, u okviru samo jednog rada, dok je najviša ocena 68%, takođe, u samo jednom studentskom radu. Ukupna prosečna ontološka ocena atributa svih modela podataka je 49,27% usklađenosti sa domenskim osobinama podataka ontologije.

Kada su u pitanju poveznici, tj. relacije uspostavljene između entiteta i mešoviti objekti-veze (gerundi), na osnovu podataka iz tabele 7.4, zaključuje se da je za prosečno 2,07 ontoloških osobina objekata od ukupno 8 kreirana odgovarajuća relacija, što čini 25,87%. Prosečan broj ontoloških osobina objekata za koje su kreirane relacije između entiteta koje odgovaraju ontološkim klasama je 1,20 (15%), za još 3,23 postoje sinonimi relacija i entiteta (40,37%), a za prosečno 0,75 ontoloških osobina objekata po modelu postoje odgovarajući mešoviti objekti-veze (9,37%). Kada se sumiraju ove tri vrednosti, dolazimo do broja od 5,18 relacija koje su semantički korektne (64,75%), a prosečan broj ontoloških osobina objekata koje nisu pokrivene relacijama iznosi 2,82, ili približno 35%. Što se tiče ograničenja u pogledu kardinaliteta osobina objekata, iz kolone 8 tabele 7.4 može se uočiti da prosečno 2,36 (14,75%) kardinaliteta u ontologiji i modelu podataka jesu identični, dok je još 2,86 (17,87%) kardinaliteta od ukupno 16 (po dva preslikavanja za svaki tip poveznika) određeno za poveznike uspostavljene između odgovarajućih tipova entiteta, ali za sinonime poveznika. Ukupno 5,22 ograničenja osobina objekata ima odgovarajuće kardinalitete tipova poveznika, što iznosi 32,62% dobro definisanih kardinaliteta u modelu podataka.

Iz dijagrama na slici 7.17. vidi se da je maksimalnih 8 ontoloških osobina objekata pokriveno relacijama i to u samo jednom modelu. U najviše radova, 17, je 6 ontoloških osobina objekata pokriveno relacijama i gerundima, dok je najmanje samo 2 ontološke osobine objekata od 8 pokriveno relacijama u 3 rada, tj. modela podataka. Iz tabele 7.4 se vidi da je najviša ontološka ocena poveznika, gerunda i kardinaliteta 81,3% usklađenosti sa domenskim ontološkim osobinama objekata (u samo jednom radu), dok je najniža ocena za poveznike 18,8% domenske usklađenosti (u tri konceptualna modela). Ukupna prosečna ontološka ocena poveznika i gerunda svih modela podataka je 48,72% usklađenosti sa domenskim osobinama objekata ontologije.

Kada se posmatra veza ontoloških nadklasa i podklasa sa IS_A hijerarhijom modela podataka, na osnovu podataka iz tabele 7.5, zaključuje se da je prosečan broj ontoloških nadklasa za koje su kreirani supertipovi entiteta IS_A hijerarhije 0,41 od maksimalnih 1 (41%). Za još 16% ontoloških nadklasa postoje sinonimi. Broj ontoloških podklasa za koje su kreirani podtipovi tipa entiteta iznosi 0,68 od maksimalno 2, što čini 34%, dok je sinonima podklasa još 21,5%. Ontoloških nadklasa koje su pokrivene supertipom entiteta IS_A hijerarhije ima ukupno 57%, a podklasa 55,75%. Iz dijagrama 7.19 može se

primetiti da je 25 studenata korektno formiralo nadklase, a 19 to nije dobro uradilo. Kada su u pitanju podklase, 19 studenata nije dobro kreiralo podklase, dok je 24 ispravno modelovalo ovaj segment konceptualnog modela podataka. Ukupna prosečna ontološka ocena IS_A hijerarhije svih modela podataka je 56,25% usklađenosti sa domenskim osobinama klasa i podklasa ontologije.

U analizi tabele 7.6, sa podacima za ukupnu ontološku ocenu korektnosti testiranih modela podataka, vidi se da najveću ocenu za semantičku korektnost modela podataka ima jedan rad sa 85,7% ontološke usklađenosti, sledeći najbolji ima skor 84,5% usklađenosti, dok je najniža vrednost ontološke ocene 36,1%.

Iz prethodne analize rezultata, kao i iz tabele 7.7, može se zaključiti da su studenti najbolje rezultate u projektovanju konceptualnog modela podataka ostvarili za tipove entiteta, sa 92,68% semantički pokrivenih. Zatim sledi IS_A hijerarhija sa 57,00% za nadklase i 55.5% za podklase, pa atributi sa 49,27% semantičke usklađenosti sa ontologijom. Najslabiji rezultat je ostvaren kod modelovanja i formiranja poveznika između entiteta i gerunda sa 48,72% semantičke usklađenosti sa ontologijom.

Ukupna prosečna ontološka ocena semantičke korektnosti svih modela podataka iznosi 61,73% semantičke usklađenosti sa domenskom ontologijom.

8. ZAKLJUČNA RAZMATRANJA

Na osnovu izloženog u disertaciji može se zaključiti da su ostvareni osnovni ciljevi i zadaci istraživanja pošto je kreiran teorijski model ontološki zasnovane analize semantičke korektnosti modela podataka primenom sistema automatskog rezonovanja i izvršena praktična implementacija teorijskog modela, što je potvrđeno sprovedenim eksperimentalnim istraživanjem.

Postavljeni zadaci istraživanja koji se odnose na analizu, projektovanje i implementaciju softverske podrške za ontološki zasnovanu analizu semantičke korektnosti modela podataka su realizovani kroz analizu postojećih stavova i rešenja u oblasti istraživanja, projektovanje i implementaciju softverske podrške za ontološki zasnovanu analizu semantičke korektnosti modela podataka. Rezultat je aplikacija «Data Model Validator» koja vrši automatizaciju postupaka iz teorijskog modela. Ovaj implementirani softverski sistem se može, već od školske 2013/2014. godine koristiti u obrazovnom procesu, u nastavnom radu, u okviru časova laboratorijskih vežbi i za izradu seminarskih radova studenata iz nastavnih predmeta Baze podataka 1 i 2, Informacioni sistemi 1 i 2, Projektovanje informacionih sistema. Pojedini delovi sistema mogu se koristiti i u okviru nastave predmeta Sistemi veštačke inteligencije.

Izvršeno je empirijsko istraživanje efikasnosti teoretskog modela i implementirane softverske podrške upotrebom eksperimentalne metode na jednoj grupi i uzorku od 44 rada, tačnije konceptualnih modela podataka urađenih od strane studenata u okviru praktične nastave predmeta Baze podataka 1. Rezultati empirijskog istraživanja pokazuju da studenti najbolje rezultate u projektovanju konceptualnog modela podataka imaju u modelovanju entiteta (92,68% semanatički pokrivenih), nešto slabije rezultate imaju u projektovanju IS_A hijerarhije (56,25%), pa atributa sa 49,27% semantičke usklađenosti sa ontologijom, dok je najslabiji rezultat ostvaren kod modelovanja poveznika i gerunda (48,72%). Ovaj rezultat može značajno poboljšati kvalitet nastavnog procesa navedenih predmeta, tako što će se posebna pažnja posvetiti elementima modela podataka u kojima su rezultati bili slabiji.

Postavljena glavna hipoteza disertacije glasi: **«Moguće je kreirati teorijski model za ontološki zasnovanu analizu semantičke korektnosti modela podataka primenom sistema automatskog rezonovanja»** je u prethodno navedenim rezultatima istraživanja i potvrđena.

Prva podhipoteza jeste: «Upotreba ontološki zasnovane analize semantičke korektnosti modela podataka primenom sistema automatskog rezonovanja povećava kvalitet kreiranih baza podataka i adekvatniju primenljivost pratećih aplikativnih softvera, u okviru određene problemski orijentisane domenske oblasti.» Potvrda ove podhipoteza se ogleda prvenstveno u rezultatima empirjskog istraživanja, izvedenim zaključcima u okviru analize istraživanja jer predloženi sistem analize modela podataka ukazuje na greške u modelovanju čime se povećava kvalitet modela podataka, što za rezultat može imati kvalitetnije kreirane baza podataka i adekvatniju primenljivost aplikativnih softvera, naravno, u određenoj problemskoj i domenskoj oblasti.

Druga podhipoteza glasi: «Ontološki zasnovana analiza semantičke korektnosti modela podataka primenom sistema automatskog rezonovanja može da se koristi u okviru nastavnog procesa, čime se povećava efikasnost nastave u oblasti baza podataka, informacionih sistema i sistema veštačke inteligencije, kao i kvalitet softverskih projekata u nastavnom procesu.» Ova podhipoteza je potvrđena izvršenim empirijskim istraživanjem nad studentskim radovima, kao i definisanim težinskim koeficijentima metrike za ontološko vrednovanje modela podataka, čime nastavnik može formirati ocenu usklađenu sa pedagoškim ciljevima i kriterijumima u okviru nastavnih predmeta u kojima će se primenjivati.

Treća podhipoteza je formulisana na sledeći način: «Moguće je softverski automatizovati ontološki zasnovanu analizu semantičke korektnosti modela podataka primenom sistema automatskog rezonovanja». Podhipoteza je potvrđena projektovanjem i implementacijom softverske podrške za ontološki zasnovanu analizu semantičke korektnosti modela podataka, kroz aplikaciju «Data Model Validator» koja vrši automatizaciju najvećeg dela postupka.

Naučni doprinos istraživanja se ogleda u kreiranju teorijskog modela ontološki zasnovane analize semantičke korektnosti modela podataka primenom sistema automatskog rezonovanja. Praktični doprinos se ogleda u implementaciji softverske podrške celog procesa. Implementacija teorijskog modela sasvim sigurno proširuje oblast primene sistema automatskog rezonovanja. Društveni značaj rada se ogleda u omogućavanju unapređivanja nastavnog procesa u oblasti projektovanja baza podataka, kao i razvoja informacionih sistema, korišćenjem predloženog modela i DMV softvera u okviru nastavnog rada sa studentima. Takođe, društveni značaj rada predstavlja i mogućnost njegove primene u naučno-istraživačkom radu i projektima iz oblasti informacionih tehnologija.

Prednosti implementiranog sistema se odnose na izuzetno jednostavan način transformacije konceptualnog modela i ontologije aplikacijom «Data Model Validator» u oblik Prolog rečenica i klauzula, koji se koristi u postupku provere korektnosti modela podataka. Predloženi model i rešenje može značajno smanjiti troškove održavanja i naknadnih izmena delova informacionih sistema, što se naravno mora dodatno istražiti i empirijski proveriti.

Nedostaci ovog pristupa se ogledaju u neophodnosti kreiranja ontologije, kao još jednog modela više u procesu razvoja informacionih sistema, što može produžiti proces razvoja istog. Drugi nedostatak jeste u setu upita koje korisnik, tj. projektant baze podataka treba da postavi. Treći problem može predstavljati vreme potrebno za dedukovanje odgovora od strane sistema automatskog rezonovanja. Poslednji nedostatak predstavlja nedovoljna sinteza seta pravila zaključivanja u jedno jedinstveno pravilo ili nekoliko pravila koje će izračunati ontološku ocenu modela podataka ili bar pojedinih elemenata i tako olakšati korišćenje softvera i ubrzati dobijanje odgovora u Prologu.

Perspektive i pravci daljeg razvoja sistema se odnose modifikacije teoretskog modela i delova softvera, kako bi se omogućila primena na modelima iz oblasti projektovanja komponenti softverskih rešenja i aplikacija, kao i na različite UML modele i dijagrame.

9. LITERATURA

[Aguirre-Urreta et al., 2008] Aguirre-Urreta M.I., Marakas G.M. *Comparing Conceptual Modeling Techniques: A Critical Review of the EER vs. OO Empirical Literature*, ACM SIGMIS Database , Volume 39 Issue 2, April 2008.

[An et al., 2004] An Y., Borgida A., Mylopoulos J. *Refining Semantic Mappings from Relational Tables to Ontologies,* Semantic Web and Databases, International Workshop, SWDB 2004, Toronto, Canada, pp. 84-90, 2004.

[An et al., 2006] An Y., Mylopoulos J., Borgida A. *Building Semantic Mappings from Databases to Ontologies*, AAAI NECTAR 2006.

[Antoniou et al., 2005] Antoniou G., Dam´asio C.V., Grosof B., Horrocks I., Kifer M., Maluszynski J., Patel-Schneider P.F. *Combining Rules and Ontologies. A survey*. EU FP6 Project: Reasoning on the Web with Rules and Semantics, 2005.

[Atkinson et al., 2006] Atkinson C., Gutheil M., Kiko K. *On the Relationship of Ontologies and Models*, Proceedings of the 2nd International Workshop on Meta Modelling WoMM, pg. 37-40, 2006.

[Avison et al., 2003] Avison D., Fitzgerald G. *Information Systems Development: Methodologies, Techniques, and Tools*, McGraw Hill, UK, 2003.

[Baclawski et al., 2002] Baclawski K., Kokar M., Kogut P., Hart L., Smith JE., Letkowski J., Emery P. *Extending the Unified Modeling Language for ontology development*, Software and System Modeling, vol. 1 , pages 142-156, 2002.

[Batini et al., 1986] Batini C., Lenzerini M., Navathe S.B. *A Comparative Analysis of Methodologies for Database Schema Integration*, ACM Computing Surveys, Vol. 18, No. 4, ACM 0360-0300/86/1200-0323, 1986.

[Batini et al., 2006] Batini C., Scannapieco M. *Data Quality*, Springer, 2006.

[Batra et al., 2001] Batra D., Antony S.R. *Consulting Support During Conceptual Database Design in the Presence Of Redundancy in Requirements Specifications: An Empirical Study*, International Journal of Human-Computer Studies, Elsevier, 2001.

[Batra et al., 2004] Batra D., Wishart N.A. *Comparing a rule-based approach with a pattern–based approach at different levels of complexity of conceptual modelling tasks*, International Journal of Human-Computer Studies, Elsevier, 2004.

[Bergamaschi et al., 2004] Bergamaschi S., Guerra F., Vincini M. *Critical Analysis of the emerging ontology languages and standards*, Project: Web Intelligent Search based on DOMain ontologies, Italy, 2004. (http://www.dbgroup.unimo.it/ wisdom/deliverables/fase_1/ d1r1.pdf)

[Berković et al., 1997] Berković I., Hotomski P. *The Concept of Logic Programming Language Based on the Resolution Theorem Prover*, Proceedings of VIII International Conference on Logic and Computer Science LIRA'97, Institute of Mathematics Novi Sad, 1997.

[Berković 1997] Berković I. *Deduktivne osnove za razvoj opisnih jezika logičkog programiranja*, Doktorska disertacija, Univerzitet u Novom Sadu, Tehnički fakultet "Mihajlo Pupin", Zrenjanin 1997.

[Berković 1999] Berković I. *Elementi veštačke inteligencije - kroz primere i zadatke*, Univerzitet u Novom Sadu, Tehnički fakultet "Mihajlo Pupin", Zrenjanin, 1999.

[Berners-Lee et al., 2001] Berners-Lee T., Hendler J., Lassila O. *The Semantic Web*, Scientific American, May 2001.

[Bonifati et al., 2008] Bonifati A., Mecca G., Pappalardo A., Raunich S., Summa G. *Schema Mapping Verification: The Spicy Way*, EDBT '08: Proceedings of the 11th international conference on Extending database technology: Advances in database technology, Publisher: ACM, March 2008.

[Bock 1997] Bock D.B. *Entity-Relationship Modelling and Normalization Errors*, Journal of Database Management, 1987.

[Booch et al., 1999] Booch G., Rumbaugh J., Jacobson I. *The Unified Modelling Language User Guide*, Addison-Wesley, 1999.

[Booch et al., 2002] Booch G., Naiburg E.J., Maksimchuk R.A. *UML za projektovanje baza podataka*, CET, Beograd, 2002.

[Borgida 1995] Borgida A. *Description Logics in Data Management*, IEEE Transactions on Knowledge and Data Engineering, vol. 7, no. 5, October 1995, pp. 671-682.

[Brdjanin et al., 2012] Brdjanin D., Maric S. *An Approach to Automated Conceptual Database, Design Based on the UML Activity Diagram*, ComSIS Vol. 9, No. 1, January 2012., pg. 250-283.

[Brockmans et al., 2004] Brockmans S., Volz R., Eberhart A., Löffler P. *Visual Modeling of OWL DL Ontologies Using UML*, Int'l Semantic Web Conference, 2004, pp. 198-213.

[Cali et al., 2002] Cali A., Calvanese D., De Giacomo G., Lenzerini M. *A Formal Framework for Reasoning on UML Class Diagrams*, Lecture Notes in Computer Science, Vol. 2366, p. 503-512, 2002.

[Chen 1976] Chen P. *The Entity-Relationship Data Model – Toward a Unified View of Data*, ACM Transactions on Database Systems, vol. 1, no 1, 1976, pp. 9-36.

[Cherfi et al., 2007] Cherfi S.S., Akoka J., Comyn-Wattiau I. *Perceived vs. Measured Quality of Conceptual Schemas: An Experimental Comparison*, Twenty-Sixth International Conference on Conceptual Modeling - ER 2007.

[Choppella et al., 2007] Choppella V., Sengupta A., Robertson E.L., Johnson S.D. *Preliminary Explorations in Specifying and Validating Entity-Relationship Models in PVS*, AFM'07, November 6, Atlanta, GA, USA, ACM ISBN 978-1-59593-879-4/07/11, 2007.

[Curino et al., 2009] Curino C., Moon H.J., Zaniolo C. *Automating Database Schema Evolution in Information System Upgrades*, HotSWUp '09: Proceedings of the Second International Workshop on Hot Topics in Software Upgrades, October 2009.

[De Lucia et al., 2009] De Lucia A., Gravino C., Oliveto R., Tortora G. *An Experimental Comparison of ER And UML Class Diagrams for Data Modelling*, Journal of Empirical Software Engineering, ISSN: 1382-3256, Springer Science+Business Media, LLC, 2009.

[Devedžić 2004] Devedžić V. *Tehnologije inteligentnih sistema*, Univerzitet u Beogradu, Fakultet organizacionih nauka, Beograd, 2004.

[Ding et al., 2002] Ding Y., Fensel D., Klein M., Omelayenko B. *The Semantic Web: Yet Another Hip?*, Elsevier Journal Data & Knowledge Engineering 41 (2002), Pages 205–227, 2002.

[Dou et al., 2006] Dou D., Pan J.Z., Qin H., LePendu P. *Towards Populating and Querying the Semantic Web*, In Proc. 2nd Int'l workshop on Scalable Semantic Web Knowledge Base Systems (SSWS 2006).

[Dou, LePendu 2006] Dou D., LePendu P., *Ontology-based Integration f or Relational Databases*, In Proc. ACM Symposium on Applied computing (SAC'06). pp. 461-466, 2006.

[Dou, LePendu et al., 2006] Dou D., LePendu P., Kim S., Qi P. 2006. *Integrating Databases into the Semantic Web Through an Ontology-Based Framework*, In Proc. 3rd Int'l workshop on Semantic Web and Databases (SWDB'06). pp. 54, 2006.

[Đurić 2004] Đurić D. *MDA-Based Ontology Infrastructure*, Computer Science and Information Systems (ComSIS) 1 (1) (2004) 91.

[Đurić et al., 2005] Đurić D., Gašević D., Devedžić V. *Ontology Modeling and MDA*, Journal of Object Technology, vol. 4, no. 1, pages 109-128, 2005.

[Eessaar et al., 2012] Eessaar E., Soobik M. *Decision Support Method for Evaluating Database Designs*, ComSIS Vol. 9, 82 No. 1, January 2012, pg. 81-106.

[El-Ghalayini et al., 2005] El-Ghalayini H., Odeh M., McClatchey R., Solomonides T. *Reverse Engineering Ontology to Conceptual Data Models*, IASTED International Conference on Databases and Applications, DBA 2005.

[Elmasri et al., 2007] Elmasri R., Navathe S. B. *Fundamentals of Database Systems*, Addison Wesley, 2007.

[Emer et al., 2008] Emer M.C., Vergilio S.R., Jino M. *Testing Relational Database Schemas with Alternative Instance Analysis*, 20th International Conference on Software Engineering & Knowledge Engineering (SEKE'2008), San Francisco, USA, 2008.

[Fisher et al., 2006] Fisher K., Mandelbaum Y., Walker D. *The Next 700 Data Description Languages*, POPL '06: Conference record of the 33rd ACM SIGPLAN-SIGACT symposium on Principles of programming languages, January 2006.

[Formica 2006] Formica A. *Ontology-based concept similarity in Formal Concept Analysis*, Information Sciences, Volume 176, Issue 18, Pages 2624-2641 September 2006.

[Formica et al., 2004] Formica A., Michele Missikoff, *Inheritance Processing and Conflicts in Structural Generalization Hierarchies*, ACM Computing Surveys (CSUR), Volume 36, Issue 3, September 2004.

[Formica et al., 2006] Formica A., Missikoff M. *Correctness of ISA hierarchies in Object-Oriented database schemas*, Advances in Database Technology, Springer Berlin/Heidelberg, 2006.

[Frankel et al., 2004] Frankel D., Hayes P., Kendall E., McGuinness D. *The Model Driven Semantic Web*, The Model-Driven Semantic Web Workshop (MDSW 2004), September, Monterey CA 2004.

[Gallaire et al., 1984] Gallaire H., Minker J., Nicolas J.M. *Logic and Databases: A Deductive Approach*, ACM Computing Surveys, Vol. 16, No. 2, June 1984, pg 153-188.

[Gašević et al., 2006] Gašević D., Djurić D., Devedžić V. *Model Driven Architecture and Ontology Development*, Springer-Verlag Berlin Heidelberg, Germany 2006.

[Genero et al., 2000] Genero M., Jiménez L., Piattini M. *Measuring the Quality of Entity Relationship Diagrams,* In: Proceedings of the 19th International Conference on Conceptual Modeling (ER 2000), Salt Lake City, USA, October 2000, 513-522.

[Genero et al., 2003] Genero M., Poels G., Piattini M. *Defining and Validating Metrics for Assessing the Maintainability of Entity-Relationship Diagrams*, Faculteit Economie en Bedrijfskunde Hoveniersberg, Gent, Working Paper Series 11 03/199, 2003.

[Gomez-Perez et al., 2002] Gomez-Perez A., Corcho O. *Ontology Languages for the Semantic Web*, IEEE Intelligent Systems *17 (1),* 2002, 54-60.

[Gray et al., 1991] Gray R., Carey B., McGlynn N., Pengelly A. *Design Metrics for Database Systems*, *BT Technology J.,* 9(4), 1991, 69-79.

[Gruber 1995] Gruber T.R., *Toward Principles for the Design of Ontologies Used for Knowledge Sharing*, Human Computer Studies, 43 (5-6), 1995, 907-928.

[Guarino 1998] Guarino N. *Formal ontology and information systems*. In: N. Guarino, ed. Formal Ontology in Information Systems. Amsterdam, 1998. Netherlands: IOS Press, pp. 3-15.

[Hoessler et al., 2004] Hoessler J., Soden M. *OCL Support in MOF Repositories*, First European Workshop on Model Driven Architecture with Emphasis on Industrial Application, Enschede, The Netherlands. March, 2004.

[Horridge 2009] Horridge M. *A Practical Guide to Building OWL Ontologies Using Protégé 4 and CO-ODE Tools* - Edition 1.2, The University Of Manchester, 2009.

[Hotomski 1982] Hotomski P. *Metode i pravila za mehaničko dokazivanje teorema u teorijama I reda sa matematičkom indukcijom*, Doktorska disertacija, Univerzitet u Beogradu, Prirodno-matematički fakultet, Beograd, 1982.

[Hotomski 2003] Hotomski P. *Sistemi veštačke inteligencije*, Univerzitet u Novom Sadu, Tehnički fakultet "Mihajlo Pupin", Zrenjanin, 2003.

[Hotomski i sar. 1988] Hotomski P., Pevac I. *Matematički i programski problemi veštačke inteligencije u oblasti automatskog dokazivanja teorema*, Naučna knjiga, Beograd, 1988.

[Kibble et al., 2002] Kibble R., *Natural Language Semantics Computational Linguistics,* Book reviews, Volume 28, Issue 1, Publisher: MIT Press, March 2002.

[Kazi i sar. 2007] Kazi Z., Hotomski P., Radulović B., Kazi Lj. *Automated Reasoning Systems and Remote Databases in Distributed Information Systems*, MIPRO XXX International Symposium Computers in Education, IEEE Region 8, 21-25 May, 2007, Opatija, Croatia, Proceedings Vol. III CTS & CIS pg 151-156.

[Kazi i sar. 2008] Kazi Lj., Radulović B., Kazi Z. *Predicate Logic BASELOG – A Business Rules Management System,* International conference „Sustainable development of Romania and its convergence to EU", section: Knowledge society within the space of unived Europe, may 16-17th, 2008, Tibiscus University, Timisoara, Romania, Proceedings, ISSN 1582-6333.

[Kesh 1995] Kesh S. *Evaluating the Quality of Entity Relationship Models*, Information and Software Technology, 37(12), 1995, pg. 681-689.

[Kifer et al., 1995] Kifer M., Lausen G., Wu J. *Logical Foundations of Object Oriented and Frame Based Languages*, Journal of the Association for Computing Machinery, May 1995.

[Lazarević i sar. 2003] Lazarević B., Marjanović Z., Aničić N., Babrogić S. *Baze podaaka*, Fakulet organizacionih nauka, Beograd, 2003.

[Light 1997] Light R. *Presenting XML*, Sams Publishing, 1997.

[Lopes et al., 2008] Lopes N., Fernandes C., Abreu S. *Representing and querying multiple ontologies with contextual logic programming*, Computer Science and Information Systems/ComSIS, 2008.

[Luger et al., 1993] Luger G.F., William A. Stubblefield, *Artifical Intelligence – Structures and strategies for complex problems solving*, The Benjamin/Cummings Publishing Company, Inc. 1993.

[Luo 2005] Luo D. *The Information Systems Modeling with an Ontology-Based ERD*, The Ninth Pacific Asisa COnference on Information Systems, Bangkok, Thailand, 2005., pg. 1447-1455.

[Marjanović 1996] Marjanović Z. *Logička specifikacija i logička prototipska implementacija baze podataka i aplikacija*, Doktorska disertacija, Fakultet Orgnizacionih nauka, Beograd, 1996.

[Markoski 2000] Markoski B. *Testiranje programa i deduktivna provera njihove korektnosti*, Magistarski rad, Fakultet tehničkih nauka, Katedra za računarske nauke i informatiku, Novi Sad, 2000.

[Markoski 2007] Markoski B. *Rezolucijska metoda za jednoprolazno simboličko testiranje razgranatih programskih struktura*, Doktorska disertacija, Fakultet tehničkih nauka, Institut za računarstvo i automatiku, Novi Sad, 2007.

[Mernik et al., 2005] Mernik M., Heering J., Sloane A.M. *When and How to Develop Domain-Specific Languages,* ACM Computing Surveys (CSUR) , Volume 37, Issue 4, December 2005.

[Mihajlović 1993] Mihajlović D. *Informacioni sistemi*, Univerzitet u Novom Sadu, Fakultet tehničkih nauka, Novi Sad, 1993.

[Mizoguchi 2005] Mizoguchi. *The Role of Ontological Engineering for AIED Research*, Computer Science and Information Systems/ComSIS, 2005.

[Mogin i sar. 1996] Mogin P., Luković I. *Principi baza podataka*, Univerzitet u Novom Sadu, Fakultet tehničkih nauka, Stylos, Novi Sad, 1996.

[Mogin i sar. 2000] Mogin P., Luković I., Govedarica M. *Principi projektovanja baza podataka*, Univerzitet u Novom Sadu, Fakultet tehničkih nauka, Stylos, Novi Sad, 2000.

[Moody 1998] Moody D. *Metrics for Evaluating the Quality of Entity Relationship Models*, Proceedings of the Seventeenth International Conference on Conceptual Modelling (ER ´98), Singapore, November 1998, 213-225.

[Moody 2005] Moody D. *Theoretical and Practical Issues in Evaluating the Quality of Conceptual Models: Current State and Future Directions*, Data & Knowledge Engineering (55), 243-276, 2005.

[Lammari et al., 2004] Lammari N., Elisabeth M. *Building and maintaining ontologies: a set of algorithms,* Elsevier Journal Data & Knowledge Engineering 48 (2004), page 155–176, 2004.

[Choi et al., 2006] Choi N., Song I.L., Han H. *A survey on ontology mapping*, ACM SIGMOD Record, Volume 35 Issue 3, September 2006.

[Noy et al., 2001] Noy N.F., McGuinness D.L. *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, March 2001.

[Ordonez et al., 2008] Ordonez C., García J. *Referential integrity quality metrics*, Decision Support Systems 44, pg. 495–508, 2008.

[Piattini et al., 2000] Piattini M., Genero M., Calero C. *Data model metrics*, UK Academy of Information Systems Annual Conference 2000, Cardiff, 26-28 April 2000.

[Piattini,Polo et al., 2000] Piattini M., Genero M., Calero C., Polo M., Ruiz F. *Database Quality*, In Chapter 14: Advanced Database Technology and Design, Eds. Mario Piattini and Oscar Díaz, Artech House, 2000, 485-509.

[Piprani et al., 2008] Piprani B., Ernst D. *A Model for Data Quality Assessment*, Proceedings of OTM Workshops 2008, pp. 750-759.

[Radulović 1997] Radulović B. *Projektovanje baza podataka u oblasti obrazovnog računarskog softvera*, Doktorska disertacija, Univerzitet u Novom Sadu, Tehnički fakultet "Mihajlo Pupin" Zrenjanin, 1997.

[Radulović 2000] Radulović B. *Baze podataka i Internet - Semistrukturirani podaci*, IX Međunarodna konferencija "Informatika u obrazovanju, kvalitet i nove informacione tehnologije", Zrenjanin, 2000.

[Radulović i sar. 1997] Radulović B., Hotomski P. *Sistem BASELOG za projektovanje baza podataka u obrazovanju*, VII Međunarodna konferencija "Informatika u obrazovanju i nove informacione tehnologije", Novi Sad, 1997.

[Radulović i sar. 2006] Radulović B., Hotomski P., Kazi Z. *Korišćenje udaljenih baza podataka u sistemima automatskog rezonovanja,* časopis za informacionu tehnologiju i multimedijalne sisteme - InfoM, godina 5, kvartal 2, volumen 18/2006, ISSN 1451-4397, str. 28-35.

[Radulović i sar. 2008] Radulović B., Berković I., Hotomski P., Kazi Z. *The Development of Baselog System and Some Applications*, International Review on Computers and Software (IRECOS), Praise Worthy Prize, Vol. 3 N. 4, ISSN 1828-6003, July 2008, pp 390-395.

[Radulović et al., 2000] Radulović B., Hotomski P. *Projecting of Deductive Databases with CWA Management in BASELOG System*, III International Conference on Theoretical Aspects of Computer Science, Novi Sad Journal of Mathematics, Vol 30, N2, 2000.

[Rahm et al., 2001] Rahm E., Bernstein P.A. *A Survey of Approaches to Automatic Schema Matching*, The VLDB Journal - The International Journal on Very Large Data Bases , Volume 10 Issue 4, Springer-Verlag New York, December 2001.

[Rinaldi et al., 2009] Rinaldi A.M. *An ontology-driven approach for semantic information retrieval on the Web*, ACM Transactions on Internet Technology (TOIT) , Volume 9 Issue 3, July 2009.

[Shoval 1997] Shoval P. *Experimental Comparisons of Entity-Relationshipand and Object-Oriented Data Models*, AJIS vol. 4 No. 2, 1997.

[Sotnykova 2006] Sotnykova A. *Semantic validation in spatio-temporal schema integration*, Thèse No 3423 (2006), École Polytechnique Fédérale De Lausanne, Lausanne, EPFL, 2006.

[Storey et al., 1997] Storey V.C., Roger H., Chiang L., Dey D., Goldstein R.C., Sudaresan S. *Database design with common sense business reasoning and learning*, ACM Transactions on Database Systems (TODS) , Volume 22 Issue 4, December 1997.

[Su et al., 2002] Su X., Ilebrekke L. *A Comparative Study of Ontology Languages and Tools*, Advanced Information Systems Engineering, Volume 2348/2002, Springer Berlin/Heidelberg, 2002.

[Sugumaran et al., 2006] Sugumaran V., Storey V.C. *The Role of Domain Ontologies in Database Design: An Ontology Management and Conceptual Design Environment*, ACM Transactions on Database Systems, Vol. 31, No. 3., 2006.

[Tošić i sar.] Tošić D., Protić R. *PROLOG kroz primere*, Tehnička knjiga, Beograd.

[Ullman 1988] Ullman J. *Principles of Database And Knowledge - Base Systems*, Stanford University, Computer Science Press, Rockville, Maryland, 1988.

[Ullman et al., 2002] Ullman J., Garcia–Molina H., Widom J. *Database Systems: The Complete Book*, Department of Computer Science, Stanford University, Prentice Hall, New Jersey, 2002.

[Ullman] Ullman J. *Assigning An Appropriate Meaning To Database Logic With Negation*, Stanford University, CA, USA.

[Van Belle 2006] Van Belle J.P. *A Framework for the Evaluation of Business Models and its Empirical Validation*, The Electronic Journal Information Systems Evaluation 2006, Volume 9, Issue 1, pp. 31-44, www.ejise.com

[Volz et al.] Volz R., Decker S., Oberle D. *Bubo - Implementing OWL in rule-based systems*, DARPA Agent Markup Language Program, http://www.daml.org/listarchive/joint-committee/att-1254/01-bubo.pdf.

[Vysniauskas et al., 2006] Vysniauskas E., Nemuraite L. *Transforming Ontology Representation from OWL to Relational Database*, ISSN 1392 – 124X Information Technology and Control, Vol.35, No.3A, 2006.

[Wang et al., 2004] Wang J., Jin B., Li J. *An Ontology-Based Publish/Subscribe System,* Middleware '04: Proceedings of the 5th ACM/IFIP/USENIX international conference on Middleware, Springer-Verlag New York, Inc., 2004.

[Yücesan et al., 2002] Yücesan E., Chen C.H., Snowdon J.L., Charnes J.M., Carson J.S.II *Model Verification and Validation*, Proceedings of the 2002 Winter Simulation Conference.

[Zamperoni et al., 1994] Zamperoni A., Loehr-Richter P. *Enhancing the Quality of Conceptual Database Specifications Through Validation*, Lecture Notes in Computer Science Springer Verlag KG Germany, ISSN 0302-9743, 1994.

9.1. LITERATURNE REFERENCE SA INTERNETA

[1] *Protégé - open source ontology editor and knowledge-base framework*, Stanford University, Salifornia, http://protege.stanford.edu

[2] *OWL Web Ontology Language Guide*,W3C Recommendation 10 February 2004, (http://www.w3.org/TR/2004/REC-owl-guide-20040210/)

[3] *OWL Web Ontology Language Overview*,W3C Recommendation 10 February 2004, (http://www.w3.org/TR/2004/REC-owl-features-20040210/)

[4] *RDF/XML Syntax Specification*,W3C Recommendation 10 February 2004, (http://www.w3.org/TR/2004/REC-rdfl-syntax-gramar-20040210/)

[5] Julita Bermejo, *A Simplified Guide to Create an Ontology*, http://tierra.aslab.upm.es/documents/controlled/ASLAB-R-2007-004.pdf

[6] W3C *Resource Description Framework (RDF): Concepts and Abstract Syntax*

http://www.w3.org/TR/2004/REC-rdf-concepts-20040210

[7] W3C *RFC 2396 - Uniform Resource Identifiers (URI): Generic Syntax*, T. Berners-Lee, R. Fielding and L. Masinter, IETF, August 1998.

http://www.isi.edu/in-notes/rfc2396.txt

[8] W3C *RDF Primer – Turtle Version,*

http://www.w3.org/2007/02/turtle/primer/

[9] *Creating & Altering the Database Prolog & Logic Programming First-Order Predicate Calculus, Paradigms of Computation*, Michael J. Ciaraldi and David Finkel, 2001. -2003.

[10] http://www.j-paine.org/students/prolog/lesson4

[11] *Jesse Hoey - Prolog*, School of Computing, University of Dundee, September 2008.

[12] http://protege.stanford.edu/publications/ontology_development/ontology101-noy-mcguinness.html

[13] http://www.swi-prolog.org

[14] Jan Wielemaker, Michiel Hildebrand and Jacco van Ossenbruggen *Using Prolog as the fundament for applications on the semantic web* University of Amsterdam, http://hcs.science.uva.nl/projects/SWI-Prolog/articles/mn9c.pdf.

[15] Jan Wielemaker *An Overview of the SWI-Prolog Programming Environment,* University of Amsterdam, http://pdf.aminer.org/000/327/818/prolog_programming_environments_architecture_and_implementation.pdf.

[16] AMZI! Prolog, http://www.amzi.com.

[17] Ontology Driven Architectures and Potential Uses of the Semantic Web in Systems and Software Engineering, http://www.w3.org/2001/sw/BestPractices/SE/ODA/060211.

9.2. LITERATURNE REFERENCE - OBJAVLJENI RADOVI AUTORA IZ OBLASTI DOKTORSKE DISERTACIJE

[Kazi et al., 1] Kazi Lj., Kazi Z., Radulović B., Letic. D.: *Using Automated Reasoning System for Data Model Evaluation*, 8th International Symposium on Intelligent Systems and Informatics SISY 2010, September 2010, Subotica, Serbia, ISBN 978-1-4244-7394-6, pp. 246-250.

[Kazi et al., 2] Kazi Z., Radulovic B. *Software Tool for Automated Analysis of Conceptual Data Model*, 34th International convention on information and communication technology, electronics and microelectronics MIPRO 2011, May 2011, Opatija, Croatia, CTS&CIS Proceedings Vol. 3, pp. 328-333.

[Kazi et al., 3] Kazi Lj., Kazi Z., Radulovic B., Stanciu O. *Integration of Conceptual Data Modeling Methods in Information System Development*, 1st International Symposium Engineering Management and Competitiveness EMC 2011, Zrenjanin, Serbia, ISBN 978-86-7672-135-1, pp. 451-456.

[Kazi et al., 4] Kazi Lj., Kazi Z., Radulovic B., Stanciu O. *Evaluation of Students' Work on Data Modeling - Teaching Improvement Implications*, ITRO journal 2011, Vol. 1, No. 1, pp. 24-30, ISSN 2217-7930.

[Kazi et al., 5] Kazi Lj., Kazi Z., Radulović B., Radosav D. *Evaluation of Models in Information Systems Development*, 14th International Conference Dependability and Quality Management ICDQM, Belgrade, Serbia, 2011, ISBN 978-86-86355-05-8, pp. 589-595.

[Kazi 6] Kazi Z. *Review of Research in Area of Data Model Corectness Analyses*, Univerzitet u Novom Sadu, Tehnički fakultet »Mihajlo Pupin«, Pupin almanah 2011, Vol. 2, No. 1, ISSN 2217-6063, str. 180-188.

[Kazi et al., 7] Kazi Z., Kazi Lj., Radulović B. *Semantička analiza korektnosti modela podataka primenom sistema automatskog rezonovanja*, XVII Simpozijum o računarskim naukama i informacionim tehnologijama - YU Info 2011, Kopaonik.

[Kazi et al., 8] Kazi Z., Kazi Lj., Radulovic B. *Automated Data Model Evaluation*, International Scientific Publications Journals - Journal Materials, Methods and Technologies, www.science-journals.eu, ISSN 1313-2539, Vol 6, 2012.

[Kazi et al., 9] Kazi Z., Kazi Lj., Radulovic B. *Analysis of Data Model Correctness by Using Automated Reasoning System*, Technics Technologies Education Management, Vol. 7, No. 3, pp. 1090-1100, ISSN 1840-1503, 2012.

[Kazi et al., 10] Kazi Z., Kazi Lj., Radulovic B. *Building Ontologies in Protégé*, 2nd Interntational Conference on Applied Internet and Information Technologies - ICAIIT, Zrenjanin, Serbia, 2013, ISBN 978-86-7672-211-2, pp. 26-29.

10. PRILOZI

10.1. BIOGRAFIJA KANDIDATA

Kazi (Peter) Zoltan je rođen 04.12.1971. godine u Zrenjaninu. Osnovnu školu „2. oktobar" i srednju školu, smer informatika, završio je u Zrenjaninskoj gimnaziji „Koča Kolarov", sa prosečnim uspehom 5,00. Za osnovno i srednješkolsko obrazovanje dobitnik je tri diplome „Vuk Karadžić". Tehnički fakultet „Mihajlo Pupin", Univerziteta u Novom Sadu, upisao je 1993. godine na smeru: Profesor informatike. Diplomirao septembra 1998. godine, sa diplomskim radom: „Informacioni podsistem finansija – Glavna knjiga", pod mentorstvom prof. dr Đorđa Nadrljanskog. Prosečna ocena tokom studija je bila 9,00. U toku studija, 1997. godine, nagrađen Univerzitetskom nagradom za tematski rad pod nazivom "Informacioni sistem Atletskog saveza".

Zaposlen na Univerzitetu u Novom Sadu, na Tehničkom fakultetu "Mihajlo Pupin" u Zrenjaninu, od 20. oktobra 1998. godine, prvo kao stručni saradnik, a zatim i kao asistent pripravnik i asistent. Izvodio vežbe iz sledećih predmeta: Baze podataka, Baze podataka 1 i 2, Informaciono-upravljački sistemi, Informacioni sistemi 1 i 2, Informacioni sistem škole, Projektovanje obrazovnog računarskog softvera, Sistemi veštačke inteligencije, Ekspertni sistemi, Poslovna inteligencija i Kompleksne baze podataka.

Postdiplomske studije upisao na Tehničkom fakultetu „Mihajlo Pupin", 1998. godine u okviru naučne oblasti: „Informatika u obrazovanju". Magistarsku tezu pod nazivom „Korišćenje udaljenih baza podataka u sistemima automatskog rezonovanja", odbranio juna 2005. godine na Tehničkom fakultetu „Mihajlo Pupin", pod mentorstvom prof. dr Biljane Radulović. Prosečna ocena tokom magistarskih studija je bila 10,00.

Temu za izradu doktorske disertacije pod nazivom: „Ontološki zasnovana analiza semantičke korektnosti modela podataka primenom sistema automatskog rezonovanja" prihvatilo je 29. marta 2010. godine stručno veće za Tehničko-tehnološke nauke Univerzitetu u Novom Sadu. Mentor za izradu disertacije je prof. dr Biljana Radulović.

Učestvovao u razvoju i realizaciji nekoliko idejnih, glavnih i izvođačkih projekata iz oblasti informacionih sistema u Zrenjaninu, Vršcu i Novom Sadu. Održao je, u organizaciji Tehničkog fakulteta „Mihajlo Pupin", 16 kurseva za obuku nastavnika, pedagoga, nezaposlenih osoba, službenika i programera iz oblasti informacionih tehnologija, programiranja, baza podataka, informacionih sistema i Microsoft Office paketa. Učestvovao je u organizaciji četiri međunarodne konferencije.

Objavio je 63 naučna i stručna rada u časopisima i na međunarodnim i domaćim naučno-stručnim skupovima i konferencijama. Autor je ili koautor 15 softvera, koautor 3 udžbenika, 1 zbirke zadataka, 1 praktikuma, učesnik u 9 naučno-istraživačkih i stručnih projekata.

Osim srpskog jezika čita, piše i govori engleski i mađarski jezik.

Oženjen je i ima dve ćerke.

10.2. POJMOVNIK

ER model podataka - (*Entity Relationship Aribute Data Model*) semantički bogat model treće generacije. Poseduje specifične koncepte za detaljan opis realnog sistema.

CASE alat (*Comupter Aided Software Engineering*) - paket programa za projektovanje softverskih rešenja, između ostalog i konceptualnog, tj. ER modela podataka.

Model podataka - Specifična teorija pomoću kojih se vrši specifikacija i projektovanje konkretnih baza podataka i informacionih sistema. Model podataka je formalna apstrakcija putem koje se realni svet preslikava u bazu podataka.

Ontologija - Skup termina koji se koriste da bi se opisao domen, tj. oblast znanja. Ontologije koriste ljudi, baze podataka i softverske aplikacije koje dele informacije iz određenog domena.

Ontološki jezik - jezik za modelovanje ontologija koji takođe predstavlja i rezultat rada ontoloških softverskih okruženja i editora.

OWL – (*Web Ontology Language*) je trenutno najpopularniji ontološki reprezentacioni jezik koji se bazira na XML jeziku.

Predikatski račun prvog reda – Logički sistem namenjen analizi formalne strukture mišljenja, pogodan za predstavljanje na računaru.

Prolog – Jezik logičkog programiranja. Najpoznatiji softver iz oblasti sistema automatskog rezonovanja.

Power Designer – CASE alat firme Sybase namenjen logičkom, konceptualnom, fizičkom modelovanju baze podatka, modelovanju poslovnih procesa, zahteva klijenata, UML i objektnih modela, XML modela itd.

Protégé – ontološki alat/editor sa Stanford univerziteta, baziran na programskom okruženju Java. Inicijalno je bio softver u oblasti medicine i biohemijskih nauka, a trenutno je jedan od vodećih ontoloških editora u svetu.

RDF – (*Resource Definition Framework*) je jezik za semantičkih mreža, služi za opis resursa na Web-u i bazira se na XML jeziku. Namenjen je za predstavljanje metapodataka o Web resursima.

Sistem automatskog rezonovanja - Računarski program koji poseduje određene komponente inteligentnog ponašanja i koji se može primenjivati kao ljuska ekspertnih sistema, dijaloških sistema, prevodioc prirodnih jezika, obrazovni računarski softver, informacioni sistem i dr.

XML – (*Extensible Markup Language*) je jezik za označavanje strukture dokumenta unutar njegovog sadržaja čija je prvenstvena namena standardizacija strukture i sadržaja dokumenata radi njihove efektivne razmene u različitim vrstama poslovnih i drugih elektronskih komunikacija. Postao je *World Wide Web Consortium*-ov standard za razmenu podataka na Internetu.

10.3. LISTING PROGRAMA DATA MODEL VALIDATOR

```
Imports System.IO
Imports Microsoft.VisualBasic
Imports System.Collections
Imports System.Xml

Public Class frmErModelValidation

    Dim fileModel As System.IO.StreamReader
    Dim fileOntologija As System.IO.StreamReader
    Dim fileClauses As System.IO.StreamWriter
    Dim fileRules As System.IO.StreamReader
    Dim filePath As String = ""
    Dim vModel As String = ""
    Dim vCeoModel As String = ""
    Dim vOntologija As String = ""
    Dim vCelaOntologija As String = ""
    Dim vRules As String = ""
    Dim XMLDoc As New XmlDocument

    Dim listaEntiteta As ArrayList
    Dim listaAtributa As ArrayList
    Dim listaAtributaUEntitetima As ArrayList
    Dim listaIdentEntiteta As ArrayList
    Dim listaTipova As ArrayList
    Dim listaTipovaPoJedan As ArrayList
    Dim listaRelacija As ArrayList
    dim listaRelacijaIEntiteta as ArrayList
    Dim listaObaveznihAtributa As ArrayList
    Dim listaCardRelacija As ArrayList
    Dim listaDepenRelacija As ArrayList
    Dim listaISa As ArrayList
    Dim listaISaEntitet As ArrayList
    Dim listaISaCard As ArrayList
    Dim listaISaPodklasa As ArrayList
    Dim listaIdentAtributa As ArrayList

    Private Function OdrediNazivAtributa(ByVal pAtribut As String) As String
        Dim pozicijaPocetkaAtributa As Integer = 0
        Dim pozicijaKrajaAtributa As Integer = 0
        Dim vAtributId As String = ""
        Dim vAttribute As String = ""
        Dim vAttributeCode As String = ""
        Dim vAttributeName As String = ""
        Dim pozicijaPocetkaAtributauModelu As Integer = 0
        Dim pozicijaKrajaAtributauModelu As Integer = 0

        vAtributId = "Id"
        vAtributId = vAtributId + Mid(pAtribut, 4, Len(pAtribut) - 7)
        pozicijaPocetkaAtributa = InStr(vCeoModel, vAtributId)
        pozicijaKrajaAtributa = pozicijaPocetkaAtributa + 200
        vAttribute = Mid(vCeoModel, pozicijaPocetkaAtributa, pozicijaKrajaAtributa -
         pozicijaPocetkaAtributa + 11)
        pozicijaPocetkaAtributauModelu = InStr(vAttribute, "<a:Code>")
        pozicijaKrajaAtributauModelu = InStr(vAttribute, "</a:Code>")
        vAttributeCode = Mid(vAttribute, pozicijaPocetkaAtributauModelu,
         pozicijaKrajaAtributauModelu - pozicijaPocetkaAtributauModelu + 9)
        pozicijaPocetkaAtributauModelu = InStr(vAttributeCode, ">")
        pozicijaKrajaAtributauModelu = InStr(vAttributeCode, "</")
        vAttributeName = Mid(vAttributeCode, pozicijaPocetkaAtributauModelu + 1,
         pozicijaKrajaAtributauModelu - pozicijaPocetkaAtributauModelu - 1)

        Return vAttributeName
    End Function
```

```
Private Function OdrediNazivIDAtributa(ByVal pAtribut As String) As String
        Dim pozicijaPocetkaAtributa As Integer = 0
        Dim pozicijaKrajaAtributa As Integer = 0
        Dim vAtributId As String = ""
        Dim vAttribute As String = ""
        Dim vAttributeCode As String = ""
        Dim vAttributeName As String = ""
        Dim pozicijaPocetkaAtributauModelu As Integer = 0
        Dim pozicijaKrajaAtributauModelu As Integer = 0

        vAtributId = "Id="
        vAtributId = vAtributId + pAtribut
        pozicijaPocetkaAtributa = InStr(vCeoModel, vAtributId)
        pozicijaKrajaAtributa = pozicijaPocetkaAtributa + 150
        vAttribute = Mid(vCeoModel, pozicijaPocetkaAtributa, pozicijaKrajaAtributa -
         pozicijaPocetkaAtributa + 11)
        pozicijaPocetkaAtributauModelu = InStr(vAttribute, "<a:Code>")
        pozicijaKrajaAtributauModelu = InStr(vAttribute, "</a:Code>")
        vAttributeCode = Mid(vAttribute, pozicijaPocetkaAtributauModelu,
         pozicijaKrajaAtributauModelu - pozicijaPocetkaAtributauModelu + 9)
        pozicijaPocetkaAtributauModelu = InStr(vAttributeCode, ">")
        pozicijaKrajaAtributauModelu = InStr(vAttributeCode, "</")
        vAttributeName = Mid(vAttributeCode, pozicijaPocetkaAtributauModelu + 1,
         pozicijaKrajaAtributauModelu - pozicijaPocetkaAtributauModelu - 1)

        Return vAttributeName
    End Function

    Private Function OdrediNazivEntiteta(ByVal pEnt As String) As String
        Dim pozicijaPocetkaEnt As Integer = 0
        Dim pozicijaKrajaEnt As Integer = 0
        Dim vEntId As String = ""
        Dim vEnt As String = ""
        Dim vEntCode As String = ""
        Dim vEntName As String = ""
        Dim pozicijaPocetkaEntModelu As Integer = 0
        Dim pozicijaKrajaEntuModelu As Integer = 0

        vEntId = "Id"
        vEntId = vEntId + Mid(pEnt, 4, Len(pEnt) - 7)
        pozicijaPocetkaEnt = InStr(vCeoModel, vEntId)
        pozicijaKrajaEnt = pozicijaPocetkaEnt + 150
        vEnt = Mid(vCeoModel, pozicijaPocetkaEnt, pozicijaKrajaEnt -
         pozicijaPocetkaEnt + 11)
        pozicijaPocetkaEntModelu = InStr(vEnt, "<a:Code>")
        pozicijaKrajaEntuModelu = InStr(vEnt, "</a:Code>")
        vEntCode = Mid(vEnt, pozicijaPocetkaEntModelu, pozicijaKrajaEntuModelu -
         pozicijaPocetkaEntModelu + 9)
        pozicijaPocetkaEntModelu = InStr(vEntCode, ">")
        pozicijaKrajaEntuModelu = InStr(vEntCode, "</")
        vEntName = Mid(vEntCode, pozicijaPocetkaEntModelu + 1, pozicijaKrajaEntuModelu
         - pozicijaPocetkaEntModelu - 1)

        Return vEntName
    End Function

    Private Sub btnLoadModel_Click(ByVal sender As System.Object, ByVal e As
     System.EventArgs) Handles btnLoadModel.Click
        On Error Resume Next

        txtModel.Text = ""
        richtxtClauses.Text = ""
        richtxtRules.Text = ""
        richtxtOntology.Text = ""
        txtEntities.Text = ""
        txtAtributes.Text = ""
        txtDataTypes.Text = ""
        txtRelationships.Text = ""
        txtAtrEnt.Text = ""
        txtEntityID.Text = ""
```

```
        txtEntRelations.Text = ""
        txtRelatCard.Text = ""
        txtMandatory.Text = ""
        txtDependent.Text = ""
        txtIDAtr.Text = ""
        txtISA.Text = ""

        OpenFileDataModel.ShowDialog()
        filePath = OpenFileDataModel.FileName
        fileModel = New System.IO.StreamReader(filePath)
        vCeoModel = fileModel.ReadToEnd
        fileModel.Close()
        txtModel.Text = vCeoModel.ToString
    End Sub

    Private Sub btnExit_Click(ByVal sender As System.Object, ByVal e As
     System.EventArgs) Handles btnExit.Click
        End
    End Sub

    Private Sub btnShowEntities_Click(ByVal sender As System.Object, ByVal e As
     System.EventArgs) Handles btnShowEntities.Click
        On Error GoTo Izlaz

        Dim vEntity As String = ""
        Dim vEntityIDAtributCode As String = ""
        Dim vEntityIDAtributName As String = ""
        Dim pozicijaPocetkaEntiteta As Integer = 0
        Dim pozicijaKrajaEntiteta As Integer = 0
        Dim pozicijaPocetkaIDEntiteta As Integer = 0
        Dim pozicijaKrajaIDEntiteta As Integer = 0

        txtEntities.Text = ""
        listaEntiteta = New ArrayList
        vModel = vCeoModel

        While Not vModel = ""
            pozicijaPocetkaEntiteta = InStr(vModel, "<o:Entity Id")
            pozicijaKrajaEntiteta = InStr(vModel, "</o:Entity>")
            vEntity = Mid(vModel, pozicijaPocetkaEntiteta, pozicijaKrajaEntiteta -
             pozicijaPocetkaEntiteta + 11)
            pozicijaPocetkaIDEntiteta = InStr(vEntity, "<a:Code>")
            pozicijaKrajaIDEntiteta = InStr(vEntity, "</a:Code>")
            vEntityIDAtributCode = Mid(vEntity, pozicijaPocetkaIDEntiteta,
             pozicijaKrajaIDEntiteta - pozicijaPocetkaIDEntiteta + 9)
            pozicijaPocetkaIDEntiteta = InStr(vEntityIDAtributCode, ">")
            pozicijaKrajaIDEntiteta = InStr(vEntityIDAtributCode, "</")
            vEntityIDAtributName = Mid(vEntityIDAtributCode, pozicijaPocetkaIDEntiteta
             + 1, pozicijaKrajaIDEntiteta - pozicijaPocetkaIDEntiteta - 1)
            txtEntities.Text = txtEntities.Text + vEntityIDAtributName + ";  "
            listaEntiteta.Add(vEntityIDAtributName)
            vModel = Mid(vModel, pozicijaKrajaEntiteta + 1, Len(vModel) -
             pozicijaKrajaEntiteta)
        End While
    Izlaz:
    End Sub

    Private Sub btnCreateClauses_Click(ByVal sender As System.Object, ByVal e As
    System.EventArgs) Handles btnCreateClauses.Click
        On Error Resume Next

        btnShowEntities.PerformClick()
        btnShowAttributes.PerformClick()
        btnRelatonships.PerformClick()
        btnShowTypes.PerformClick()
        btnEntityAttribute.PerformClick()
        btnIDAttributes.PerformClick()
        btnMandatory.PerformClick()
        btnIDAtr.PerformClick()
        btnEntRelations.PerformClick()
```

```
btnRelatCard.PerformClick()
btnDependent.PerformClick()
btnISA.PerformClick()

fileClauses = New System.IO.StreamWriter("c:\test.pro")
Dim i As Integer = 0
Dim brSlogova As Integer = 0
brSlogova = listaEntiteta.Count - 1
For i = 0 To brSlogova
    richtxtClauses.Text = richtxtClauses.Text + "ent(" +
     listaEntiteta.Item(i).ToString.ToLower + ")." + Chr(13)
    fileClauses.WriteLine("ent(" + listaEntiteta.Item(i).ToString.ToLower +
     ").")
Next i
brSlogova = listaAtributa.Count - 1
For i = 0 To brSlogova
    richtxtClauses.Text = richtxtClauses.Text + "atr(" +
     listaAtributa.Item(i).ToString.ToLower + ")." + Chr(13)
    fileClauses.WriteLine("atr(" + listaAtributa.Item(i).ToString.ToLower +
     ").")
Next i
brSlogova = listaRelacija.Count - 1
For i = 0 To brSlogova
    richtxtClauses.Text = richtxtClauses.Text + "rel(" +
     listaRelacija.Item(i).ToString.ToLower + ")." + Chr(13)
    fileClauses.WriteLine("atr(" + listaRelacija.Item(i).ToString.ToLower +
     ").")
Next i
richtxtClauses.Text = richtxtClauses.Text + "res(idatr)." + Chr(13)
fileClauses.WriteLine("res(idAtr).")
richtxtClauses.Text = richtxtClauses.Text + "res(mandatory)." + Chr(13)
fileClauses.WriteLine("res(mandatory).")
richtxtClauses.Text = richtxtClauses.Text + "res(mincard0)." + Chr(13)
fileClauses.WriteLine("res(mincard0).")
richtxtClauses.Text = richtxtClauses.Text + "res(mincard1)." + Chr(13)
fileClauses.WriteLine("res(mincard1).")
richtxtClauses.Text = richtxtClauses.Text + "res(maxcard1)." + Chr(13)
fileClauses.WriteLine("res(maxcard1).")
richtxtClauses.Text = richtxtClauses.Text + "res(maxcardm)." + Chr(13)
fileClauses.WriteLine("res(maxcardm).")
richtxtClauses.Text = richtxtClauses.Text + "res(dependent)." + Chr(13)
fileClauses.WriteLine("res(dependent).")
richtxtClauses.Text = richtxtClauses.Text + "res(inherit)." + Chr(13)
fileClauses.WriteLine("res(inherit).")
For i = 0 To listaTipovaPoJedan.Count - 1
    richtxtClauses.Text = richtxtClauses.Text + "res(" +
     listaTipovaPoJedan.Item(i).ToString.ToLower + ")." + Chr(13)
    fileClauses.WriteLine("res(" + listaTipovaPoJedan.Item(i).ToString.ToLower
     + ").")
Next i
brSlogova = listaAtributa.Count - 1
For i = 0 To brSlogova
    richtxtClauses.Text = richtxtClauses.Text + "p(" +
     listaAtributa.Item(i).ToString.ToLower + ", " +
     listaTipova.Item(i).ToString.ToLower + ")." + Chr(13)
    fileClauses.WriteLine("p(" + listaAtributa.Item(i).ToString.ToLower + ", "
     + listaTipova.Item(i).ToString.ToLower + ").")
Next i
For i = 0 To listaAtributaUEntitetima.Count - 1
    richtxtClauses.Text = richtxtClauses.Text +
     listaAtributaUEntitetima.Item(i).ToString.ToLower + "." + Chr(13)
    fileClauses.WriteLine(listaAtributaUEntitetima.Item(i).ToString.ToLower +
     ".")
Next i
For i = 0 To listaIdentEntiteta.Count - 1
    richtxtClauses.Text = richtxtClauses.Text +
     listaIdentEntiteta.Item(i).ToString.ToLower + "." + Chr(13)
    fileClauses.WriteLine(listaIdentEntiteta.Item(i).ToString.ToLower + ".")
Next i
For i = 0 To listaObaveznihAtributa.Count - 1
```

```
            richtxtClauses.Text = richtxtClauses.Text +
             listaObaveznihAtributa.Item(i).ToString.ToLower + "." + Chr(13)
            fileClauses.WriteLine(listaObaveznihAtributa.Item(i).ToString.ToLower +
             ".")
        Next i
        For i = 0 To listaRelacijaIEntiteta.Count - 1
            richtxtClauses.Text = richtxtClauses.Text +
             listaRelacijaIEntiteta.Item(i).ToString.ToLower + "." + Chr(13)
            fileClauses.WriteLine(listaRelacijaIEntiteta.Item(i).ToString.ToLower +
             ".")
        Next i
        For i = 0 To listaCardRelacija.Count - 1
            richtxtClauses.Text = richtxtClauses.Text +
             listaCardRelacija.Item(i).ToString.ToLower + "." + Chr(13)
            fileClauses.WriteLine(listaCardRelacija.Item(i).ToString.ToLower + ".")
        Next i
        For i = 0 To listaDepenRelacija.Count - 1
            richtxtClauses.Text = richtxtClauses.Text +
             listaDepenRelacija.Item(i).ToString.ToLower + "." + Chr(13)
            fileClauses.WriteLine(listaDepenRelacija.Item(i).ToString.ToLower + ".")
        Next i
        For i = 0 To listaIdentAtributa.Count - 1
            richtxtClauses.Text = richtxtClauses.Text +
             listaIdentAtributa.Item(i).ToString.ToLower + "." + Chr(13)
            fileClauses.WriteLine(listaIdentAtributa.Item(i).ToString.ToLower + ".")
        Next i
        For i = 0 To listaISa.Count - 1
            richtxtClauses.Text = richtxtClauses.Text +
             listaISa.Item(i).ToString.ToLower + "." + Chr(13)
            fileClauses.WriteLine(listaISa.Item(i).ToString.ToLower + ".")
        Next i
        For i = 0 To listaISaCard.Count - 1
            richtxtClauses.Text = richtxtClauses.Text +
             listaISaCard.Item(i).ToString.ToLower + "." + Chr(13)
            fileClauses.WriteLine(listaISaCard.Item(i).ToString.ToLower + ".")
        Next i
        For i = 0 To listaISaEntitet.Count - 1
            richtxtClauses.Text = richtxtClauses.Text +
             listaISaEntitet.Item(i).ToString.ToLower + "." + Chr(13)
            fileClauses.WriteLine(listaISaEntitet.Item(i).ToString.ToLower + ".")
        Next i
        For i = 0 To listaISaPodklasa.Count - 1
            richtxtClauses.Text = richtxtClauses.Text +
             listaISaPodklasa.Item(i).ToString.ToLower + "." + Chr(13)
            fileClauses.WriteLine(listaISaPodklasa.Item(i).ToString.ToLower + ".")
        Next i
    End Sub

    Private Sub btnShowAttributes_Click(ByVal sender As System.Object, ByVal e As
     System.EventArgs) Handles btnShowAttributes.Click
        On Error GoTo Izlaz

        Dim vEntity As String = ""
        Dim vEntityIDAtributCode As String = ""
        Dim vEntityIDAtributName As String = ""
        Dim vEntityAtribut As String = ""
        Dim pozicijaPocetkaEntiteta As Integer = 0
        Dim pozicijaKrajaEntiteta As Integer = 0
        Dim pozicijaPocetkaIDEntiteta As Integer = 0
        Dim pozicijaKrajaIDEntiteta As Integer = 0
        Dim pozicijaPocetkaAtr As Integer = 0
        Dim pozicijaKrajaAtr As Integer = 0
        Dim pozicijaPocetkaRef As Integer = 0

        txtAtributes.Text = ""
        listaAtributa = New ArrayList
        vModel = vCeoModel

     Dalje:  While Not vModel = ""
            pozicijaPocetkaEntiteta = InStr(vModel, "<o:Entity Id")
```

```
        If pozicijaPocetkaEntiteta = 0 Then GoTo Izlaz
        pozicijaKrajaEntiteta = InStr(vModel, "</o:Entity>")
        vEntity = Mid(vModel, pozicijaPocetkaEntiteta, pozicijaKrajaEntiteta -
         pozicijaPocetkaEntiteta + 11)
        pozicijaPocetkaIDEntiteta = InStr(vEntity, "<a:Code>")
        pozicijaKrajaIDEntiteta = InStr(vEntity, "</a:Code>")
        vEntityIDAtributCode = Mid(vEntity, pozicijaPocetkaIDEntiteta,
         pozicijaKrajaIDEntiteta - pozicijaPocetkaIDEntiteta + 9)
        pozicijaPocetkaIDEntiteta = InStr(vEntityIDAtributCode, ">")
        pozicijaKrajaIDEntiteta = InStr(vEntityIDAtributCode, "</")
        vEntityIDAtributName = Mid(vEntityIDAtributCode, pozicijaPocetkaIDEntiteta
         + 1, pozicijaKrajaIDEntiteta - pozicijaPocetkaIDEntiteta - 1)

        While Not vEntity = ""
            pozicijaPocetkaAtr = InStr(vEntity, "<c:DataItem>")
            If pozicijaPocetkaAtr = 0 Then GoTo Izlaz
            pozicijaKrajaAtr = InStr(vEntity, "</c:DataItem>")
            vEntityAtribut = Mid(vEntity, pozicijaPocetkaAtr, pozicijaKrajaAtr -
              pozicijaPocetkaAtr)
            pozicijaPocetkaRef = InStr(vEntityAtribut, "Ref=")
            vEntityAtribut = Mid(vEntityAtribut, pozicijaPocetkaRef,
              pozicijaPocetkaRef)
            vEntityAtribut = OdrediNazivAtributa(vEntityAtribut)
            txtAtributes.Text = txtAtributes.Text + vEntityAtribut + "; "
            listaAtributa.Add(vEntityAtribut)
            vEntity = Mid(vEntity, pozicijaKrajaAtr + 1, Len(vEntity) -
              pozicijaKrajaAtr)
        End While

    End While
 Izlaz:
    If (Len(vModel) - pozicijaKrajaEntiteta) > 0 Then
        vModel = Mid(vModel, pozicijaKrajaEntiteta + 1, Len(vModel) -
         pozicijaKrajaEntiteta)
        GoTo Dalje
    End If
End Sub

Private Sub btnShowTypes_Click(ByVal sender As System.Object, ByVal e As
 System.EventArgs) Handles btnShowTypes.Click
    On Error GoTo Izlaz

    Dim vAttribute As String = ""
    Dim vTipPodatka As String = ""
    Dim vAttributeIDAtributCode As String = ""
    Dim vAttributeIDAtributName As String = ""
    Dim vTipPodatkaCode As String = ""
    Dim pozicijaPocetkaAtributa As Integer = 0
    Dim pozicijaKrajaAtributa As Integer = 0
    Dim pozicijaPocetkaIDAtributa As Integer = 0
    Dim pozicijaKrajaIDAtributa As Integer = 0
    Dim pozicijaPocetkaTipaPodatka As Integer = 0
    Dim pozicijaKrajaTipaPodatka As Integer = 0
    Dim i As Integer = 0

    txtDataTypes.Text = ""
    listaTipova = New ArrayList
    listaTipovaPoJedan = New ArrayList
    vModel = vCeoModel

    While Not vModel = ""
        pozicijaPocetkaAtributa = InStr(vModel, "<o:DataItem")
        pozicijaKrajaAtributa = InStr(vModel, "</o:DataItem>")
        vAttribute = Mid(vModel, pozicijaPocetkaAtributa, pozicijaKrajaAtributa -
        pozicijaPocetkaAtributa + 11)
        pozicijaPocetkaIDAtributa = InStr(vAttribute, "<a:Code>")
        pozicijaKrajaIDAtributa = InStr(vAttribute, "</a:Code>")
        pozicijaPocetkaTipaPodatka = InStr(vAttribute, "<a:DataType>")
        pozicijaKrajaTipaPodatka = InStr(vAttribute, "</a:DataType>")
```

```
            vTipPodatkaCode = Mid(vAttribute, pozicijaPocetkaTipaPodatka,
            pozicijaKrajaTipaPodatka - pozicijaPocetkaTipaPodatka + 9)
            pozicijaPocetkaTipaPodatka = InStr(vTipPodatkaCode, ">")
            pozicijaKrajaTipaPodatka = InStr(vTipPodatkaCode, "</")
            vTipPodatka = Mid(vTipPodatkaCode, pozicijaPocetkaTipaPodatka + 1,
            pozicijaKrajaTipaPodatka - pozicijaPocetkaTipaPodatka - 1)
            If vTipPodatka.Substring(0, 1) = "A" Then
                vTipPodatka = Mid(vTipPodatka, 1, 1)
            End If
            If vTipPodatka.Substring(0, 1) = "N" Then
                vTipPodatka = Mid(vTipPodatka, 1, 1)
            End If
            If InStr(vTipPodatka, "VA") Then
                vTipPodatka = Mid(vTipPodatka, 1, 2)
            End If
            If InStr(vTipPodatka, "LA") Then
                vTipPodatka = Mid(vTipPodatka, 1, 2)
            End If
            If InStr(vTipPodatka, "TXT") Then
                vTipPodatka = Mid(vTipPodatka, 1, 3)
            End If
            If InStr(vTipPodatka, "NU") Then
                vTipPodatka = Mid(vTipPodatka, 1, 2)
            End If
            If InStr(vTipPodatka, "MN") Then
                vTipPodatka = Mid(vTipPodatka, 1, 2)
            End If
            listaTipova.Add(vTipPodatka)
            For i = 0 To listaTipova.Count - 1
                If Not listaTipovaPoJedan.Contains(vTipPodatka) Then
                    listaTipovaPoJedan.Add(vTipPodatka)
                End If
            Next i
            txtDataTypes.Text = txtDataTypes.Text + vTipPodatka + ";  "
            vModel = Mid(vModel, pozicijaKrajaAtributa + 1, Len(vModel) -
             pozicijaKrajaAtributa)
        End While
     Izlaz:
    End Sub

    Private Sub btnRelatonships_Click(ByVal sender As System.Object, ByVal e As
     System.EventArgs) Handles btnRelatonships.Click
        On Error GoTo Izlaz

        Dim vEntity As String = ""
        Dim vEntityIDAtributCode As String = ""
        Dim vEntityIDAtributName As String = ""
        Dim pozicijaPocetkaEntiteta As Integer = 0
        Dim pozicijaKrajaEntiteta As Integer = 0
        Dim pozicijaPocetkaIDEntiteta As Integer = 0
        Dim pozicijaKrajaIDEntiteta As Integer = 0

        txtRelationships.Text = ""
        listaRelacija = New ArrayList
        vModel = vCeoModel

        While Not vModel = ""
            pozicijaPocetkaEntiteta = InStr(vModel, "<o:Relationship Id")
            pozicijaKrajaEntiteta = InStr(vModel, "</o:Relationship>")
            vEntity = Mid(vModel, pozicijaPocetkaEntiteta, pozicijaKrajaEntiteta -
            pozicijaPocetkaEntiteta + 11)
            pozicijaPocetkaIDEntiteta = InStr(vEntity, "<a:Code>")
            pozicijaKrajaIDEntiteta = InStr(vEntity, "</a:Code>")
            vEntityIDAtributCode = Mid(vEntity, pozicijaPocetkaIDEntiteta,
            pozicijaKrajaIDEntiteta - pozicijaPocetkaIDEntiteta + 9)
            pozicijaPocetkaIDEntiteta = InStr(vEntityIDAtributCode, ">")
            pozicijaKrajaIDEntiteta = InStr(vEntityIDAtributCode, "</")
            vEntityIDAtributName = Mid(vEntityIDAtributCode, pozicijaPocetkaIDEntiteta
            + 1, pozicijaKrajaIDEntiteta - pozicijaPocetkaIDEntiteta - 1)
```

```
            txtRelationships.Text = txtRelationships.Text + vEntityIDAtributName + ";
            "
            listaRelacija.Add(vEntityIDAtributName)
            vModel = Mid(vModel, pozicijaKrajaEntiteta + 1, Len(vModel) -
            pozicijaKrajaEntiteta)
        End While
     Izlaz:
    End Sub

    Private Sub btnChangeLanguage_Click(ByVal sender As System.Object, ByVal e As
     System.EventArgs) Handles btnChangeLanguage.Click
        If radbtnSerbian.Checked = True Then
            Me.Text = " Provera korektnosti modela podataka"
            grpbxLanguage.Text = "Jezik"
            radbtnSerbian.Text = "Srpski"
            radbtnEnglish.Text = "Engleski"
            btnChangeLanguage.Text = "&Promena jezika"
            btnExit.Text = "&Izlaz"
            btnLoadModel.Text = "Učitaj &model podataka"
            btnLoadOntology.Text = "&Učitaj ontologiju"
            btnRules.Text = "Pravila &zaključivanja"
            btnCreateClauses.Text = "&Kreiraj klauzule"
            btnRunProlog.Text = "Pokreni &Prolog"
            btnNewAnalyses.Text = "Nova &analiza"
            lblClauses.Text = "Klauzule:"
            lblDataModel.Text = "Model podataka:"
            lblOntology.Text = "Ontologija:"
            lblReasoningRules.Text = "Pravila zaključivanja:"
            btnOntologyMapping.Text = "Mapiraj ontologiju"
        End If
        If radbtnEnglish.Checked = True Then
            Me.Text = " Data Model Validation"
            grpbxLanguage.Text = "Language"
            radbtnSerbian.Text = "Serbian"
            radbtnEnglish.Text = "English"
            btnChangeLanguage.Text = "&Change language"
            btnExit.Text = "&Exit"
            btnLoadModel.Text = "&Load data model"
            btnLoadOntology.Text = "Load &ontology"
            btnRules.Text = "Add reasoning &rules"
            btnCreateClauses.Text = "&Create clauses"
            btnRunProlog.Text = "&Run Prolog"
            btnNewAnalyses.Text = "New &analyses"
            lblClauses.Text = "Clauses:"
            lblDataModel.Text = "Data model:"
            lblOntology.Text = "Ontology:"
            lblReasoningRules.Text = "Reasoning rules:"
            btnOntologyMapping.Text = "Ontology &mapping"
        End If
    End Sub

    Private Sub btnRules_Click(ByVal sender As System.Object, ByVal e As
     System.EventArgs) Handles btnRules.Click
        On Error Resume Next

        richtxtRules.Text = ""
        OpenFileRules.ShowDialog()
        filePath = OpenFileRules.FileName
        fileRules = New System.IO.StreamReader(filePath)
        vRules = fileRules.ReadToEnd
        fileRules.Close()
        richtxtRules.Text = vRules.ToString
        richtxtClauses.Text = richtxtClauses.Text + vRules
        fileClauses.WriteLine(vRules)
        fileClauses.Close()
    End Sub

    Private Sub btnLoadOntology_Click(ByVal sender As System.Object, ByVal e As
     System.EventArgs) Handles btnLoadOntology.Click
        On Error Resume Next
```

```
        OpenFileOntology.ShowDialog()
        filePath = OpenFileOntology.FileName
        fileOntologija = New System.IO.StreamReader(filePath)
        vCelaOntologija = fileOntologija.ReadToEnd
        fileOntologija.Close()
        richtxtOntology.Text = vCelaOntologija.ToString
        XMLDoc.Load(filePath)
    End Sub

    Private Sub btnEntityAttribute_Click(ByVal sender As System.Object, ByVal e As
     System.EventArgs) Handles btnEntityAttribute.Click
        On Error GoTo Izlaz

        Dim vEntity As String = ""
        Dim vEntityIDAtributCode As String = ""
        Dim vEntityIDAtributName As String = ""
        Dim vEntityAtribut As String = ""
        Dim pozicijaPocetkaEntiteta As Integer = 0
        Dim pozicijaKrajaEntiteta As Integer = 0
        Dim pozicijaPocetkaIDEntiteta As Integer = 0
        Dim pozicijaKrajaIDEntiteta As Integer = 0
        Dim pozicijaPocetkaAtr As Integer = 0
        Dim pozicijaKrajaAtr As Integer = 0
        Dim pozicijaPocetkaRef As Integer = 0

        txtAtrEnt.Text = ""
        listaAtributaUEntitetima = New ArrayList
        vModel = vCeoModel

     Dalje:  While Not vModel = ""
            pozicijaPocetkaEntiteta = InStr(vModel, "<o:Entity Id")
            If pozicijaPocetkaEntiteta = 0 Then GoTo Izlaz
            pozicijaKrajaEntiteta = InStr(vModel, "</o:Entity>")
            vEntity = Mid(vModel, pozicijaPocetkaEntiteta, pozicijaKrajaEntiteta -
            pozicijaPocetkaEntiteta + 11)
            pozicijaPocetkaIDEntiteta = InStr(vEntity, "<a:Code>")
            pozicijaKrajaIDEntiteta = InStr(vEntity, "</a:Code>")
            vEntityIDAtributCode = Mid(vEntity, pozicijaPocetkaIDEntiteta,
            pozicijaKrajaIDEntiteta - pozicijaPocetkaIDEntiteta + 9)
            pozicijaPocetkaIDEntiteta = InStr(vEntityIDAtributCode, ">")
            pozicijaKrajaIDEntiteta = InStr(vEntityIDAtributCode, "</")
            vEntityIDAtributName = Mid(vEntityIDAtributCode, pozicijaPocetkaIDEntiteta
            + 1, pozicijaKrajaIDEntiteta - pozicijaPocetkaIDEntiteta - 1)

            While Not vEntity = ""
                pozicijaPocetkaAtr = InStr(vEntity, "<c:DataItem>")
                If pozicijaPocetkaAtr = 0 Then GoTo Izlaz
                pozicijaKrajaAtr = InStr(vEntity, "</c:DataItem>")
                vEntityAtribut = Mid(vEntity, pozicijaPocetkaAtr, pozicijaKrajaAtr -
                pozicijaPocetkaAtr)
                pozicijaPocetkaRef = InStr(vEntityAtribut, "Ref=")
                vEntityAtribut = Mid(vEntityAtribut, pozicijaPocetkaRef,
                pozicijaPocetkaRef)
                vEntityAtribut = OdrediNazivAtributa(vEntityAtribut)
                txtAtrEnt.Text = txtAtrEnt.Text + vEntityIDAtributName + "(" +
                vEntityAtribut + "); "
                listaAtributaUEntitetima.Add("p(" + vEntityIDAtributName + ", " +
                vEntityAtribut + ")")
                vEntity = Mid(vEntity, pozicijaKrajaAtr + 1, Len(vEntity) -
                pozicijaKrajaAtr)
            End While

        End While
     Izlaz:
        If (Len(vModel) - pozicijaKrajaEntiteta) > 0 Then
            vModel = Mid(vModel, pozicijaKrajaEntiteta + 1, Len(vModel) -
            pozicijaKrajaEntiteta)
            GoTo Dalje
        End If
```

```
End Sub

Private Sub btnRunProlog_Click(ByVal sender As System.Object, ByVal e As
 System.EventArgs) Handles btnRunProlog.Click
    On Error GoTo Izlaz

    Process.Start("C:\Data Model Validator\Data Model
    Validator\Bin\Debug\Prolog\a4ideA.exe")

    Exit Sub
Izlaz:
    MessageBox.Show("Prolog nije moguće pokrenuti!", "Upozorenje",
    MessageBoxButtons.OK, MessageBoxIcon.Error)
End Sub

Private Sub btnIDAttributes_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnIDAttributes.Click
    On Error GoTo Izlaz

    Dim vEntity As String = ""
    Dim vEntityIDAtributCode As String = ""
    Dim vEntityIDAtributName As String = ""
    Dim vEntityAtribut As String = ""
    Dim pozicijaPocetkaEntiteta As Integer = 0
    Dim pozicijaKrajaEntiteta As Integer = 0
    Dim pozicijaPocetkaIDEntiteta As Integer = 0
    Dim pozicijaKrajaIDEntiteta As Integer = 0
    Dim pozicijaPocetkaID As Integer = 0
    Dim pozicijaKrajaID As Integer = 0
    Dim pozicijaPocetkaRef As Integer = 0

    txtEntityID.Text = ""
    listaIdentEntiteta = New ArrayList
    vModel = vCeoModel

    While Not vModel = ""
        pozicijaPocetkaEntiteta = InStr(vModel, "<o:Entity Id")
        If pozicijaPocetkaEntiteta = 0 Then GoTo Izlaz
        pozicijaKrajaEntiteta = InStr(vModel, "</o:Entity>")
        vEntity = Mid(vModel, pozicijaPocetkaEntiteta, pozicijaKrajaEntiteta -
        pozicijaPocetkaEntiteta + 11)
        pozicijaPocetkaIDEntiteta = InStr(vEntity, "<a:Code>")
        pozicijaKrajaIDEntiteta = InStr(vEntity, "</a:Code>")
        vEntityIDAtributCode = Mid(vEntity, pozicijaPocetkaIDEntiteta,
        pozicijaKrajaIDEntiteta - pozicijaPocetkaIDEntiteta + 9)
        pozicijaPocetkaIDEntiteta = InStr(vEntityIDAtributCode, ">")
        pozicijaKrajaIDEntiteta = InStr(vEntityIDAtributCode, "</")
        vEntityIDAtributName = Mid(vEntityIDAtributCode, pozicijaPocetkaIDEntiteta
        + 1, pozicijaKrajaIDEntiteta - pozicijaPocetkaIDEntiteta - 1)

        pozicijaPocetkaID = InStr(vEntity, "<c:PrimaryIdentifier>")
        If pozicijaPocetkaID <> 0 Then
            txtEntityID.Text = txtEntityID.Text + vEntityIDAtributName + "; "
            listaIdentEntiteta.Add("p(" + vEntityIDAtributName + ", idatr)")
        End If
        vModel = Mid(vModel, pozicijaKrajaEntiteta + 1, Len(vModel) -
        pozicijaKrajaEntiteta)
      End While
Izlaz:
End Sub

Private Sub btnMandatory_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnMandatory.Click
    On Error GoTo Izlaz

    Dim vEntity As String = ""
    Dim vEntityIDAtributCode As String = ""
    Dim vEntityIDAtributName As String = ""
    Dim pozicijaPocetkaEntiteta As Integer = 0
    Dim pozicijaKrajaEntiteta As Integer = 0
```

```
        Dim pozicijaPocetkaIDEntiteta As Integer = 0
        Dim pozicijaKrajaIDEntiteta As Integer = 0
        Dim vEntityAtribut As String = ""
        Dim pozicijaPocetkaAtr As Integer = 0
        Dim pozicijaKrajaAtr As Integer = 0
        Dim pozicijaPocetkaRef As Integer = 0

        txtMandatory.Text = ""
        listaObaveznihAtributa = New ArrayList
        vModel = vCeoModel

    Dalje:  While Not vModel = ""
            pozicijaPocetkaEntiteta = InStr(vModel, "<o:Entity Id")
            If pozicijaPocetkaEntiteta = 0 Then GoTo Izlaz
            pozicijaKrajaEntiteta = InStr(vModel, "</o:Entity>")
            vEntity = Mid(vModel, pozicijaPocetkaEntiteta, pozicijaKrajaEntiteta -
            pozicijaPocetkaEntiteta + 11)
            pozicijaPocetkaIDEntiteta = InStr(vEntity, "<a:Code>")
            pozicijaKrajaIDEntiteta = InStr(vEntity, "</a:Code>")
            vEntityIDAtributCode = Mid(vEntity, pozicijaPocetkaIDEntiteta,
            pozicijaKrajaIDEntiteta - pozicijaPocetkaIDEntiteta + 9)
            pozicijaPocetkaIDEntiteta = InStr(vEntityIDAtributCode, ">")
            pozicijaKrajaIDEntiteta = InStr(vEntityIDAtributCode, "</")
            vEntityIDAtributName = Mid(vEntityIDAtributCode, pozicijaPocetkaIDEntiteta
            + 1, pozicijaKrajaIDEntiteta - pozicijaPocetkaIDEntiteta - 1)

            While Not vEntity = ""
                pozicijaPocetkaAtr = InStr(vEntity, "<c:DataItem>")
                If pozicijaPocetkaAtr = 0 Then GoTo Izlaz
                pozicijaKrajaAtr = InStr(vEntity, "</c:DataItem>")
                vEntityAtribut = Mid(vEntity, pozicijaPocetkaAtr, pozicijaKrajaAtr -
                 pozicijaPocetkaAtr)
                pozicijaPocetkaRef = InStr(vEntityAtribut, "Ref=")
                vEntityAtribut = Mid(vEntityAtribut, pozicijaPocetkaRef,
                 pozicijaPocetkaRef)
                vEntityAtribut = OdrediNazivAtributa(vEntityAtribut)
                If InStr(vEntity, "<a:Mandatory>") <> 0 Or InStr(vEntity,
                 "<a:BaseAttribute.Mandatory>") <> 0 Then
                    txtMandatory.Text = txtMandatory.Text + vEntityAtribut + "; "
                    listaObaveznihAtributa.Add("p(" + vEntityAtribut + ", mandatory)")
                End If
                vEntity = Mid(vEntity, pozicijaKrajaAtr + 1, Len(vEntity) -
                 pozicijaKrajaAtr)
            End While

        End While
     Izlaz:
        If (Len(vModel) - pozicijaKrajaEntiteta) > 0 Then
            vModel = Mid(vModel, pozicijaKrajaEntiteta + 1, Len(vModel) -
            pozicijaKrajaEntiteta)
            GoTo Dalje
        End If
    End Sub

    Private Sub btnEntRelations_Click(ByVal sender As System.Object, ByVal e As
    System.EventArgs) Handles btnEntRelations.Click
        On Error GoTo Izlaz

        Dim vEntity As String = ""
        Dim vEntityIDAtributCode As String = ""
        Dim vEntityIDAtributName As String = ""
        Dim vEntityRel As String = ""
        Dim vRelEntity As String = ""
        Dim vEntRel1 As String = ""
        Dim vEntRel2 As String = ""
        Dim pozicijaPocetkaEntiteta As Integer = 0
        Dim pozicijaKrajaEntiteta As Integer = 0
        Dim pozicijaPocetkaIDEntiteta As Integer = 0
        Dim pozicijaKrajaIDEntiteta As Integer = 0
        Dim pozicijaPocetkaPrvogEnt As Integer = 0
```

```
        Dim pozicijaPocetkaDrugogEnt As Integer = 0
        Dim pozicijaKrajaPrvogEnt As Integer = 0
        Dim pozicijaKrajaDrugogEnt As Integer = 0
        Dim pozicijaPocetkaRef As Integer = 0

        txtEntRelations.Text = ""
        listaRelacijaIEntiteta = New ArrayList
        vModel = vCeoModel

        While Not vModel = ""
            pozicijaPocetkaEntiteta = InStr(vModel, "<o:Relationship Id")
            pozicijaKrajaEntiteta = InStr(vModel, "</o:Relationship>")
            vEntity = Mid(vModel, pozicijaPocetkaEntiteta, pozicijaKrajaEntiteta -
            pozicijaPocetkaEntiteta + 11)
            pozicijaPocetkaIDEntiteta = InStr(vEntity, "<a:Code>")
            pozicijaKrajaIDEntiteta = InStr(vEntity, "</a:Code>")
            vEntityIDAtributCode = Mid(vEntity, pozicijaPocetkaIDEntiteta,
            pozicijaKrajaIDEntiteta - pozicijaPocetkaIDEntiteta + 9)
            pozicijaPocetkaIDEntiteta = InStr(vEntityIDAtributCode, ">")
            pozicijaKrajaIDEntiteta = InStr(vEntityIDAtributCode, "</")
            vEntityIDAtributName = Mid(vEntityIDAtributCode, pozicijaPocetkaIDEntiteta
            + 1, pozicijaKrajaIDEntiteta - pozicijaPocetkaIDEntiteta - 1)

            pozicijaPocetkaPrvogEnt = InStr(vEntity, "<c:Object1>")
            pozicijaKrajaPrvogEnt = InStr(vEntity, "</c:Object1>")
            vEntityRel = Mid(vEntity, pozicijaPocetkaPrvogEnt, pozicijaKrajaPrvogEnt -
            pozicijaPocetkaPrvogEnt)
            pozicijaPocetkaRef = InStr(vEntityRel, "Ref=")
            vEntityRel = Mid(vEntityRel, pozicijaPocetkaRef, pozicijaPocetkaRef)
            vEntityRel = OdrediNazivEntiteta(vEntityRel)

            pozicijaPocetkaDrugogEnt = InStr(vEntity, "<c:Object2>")
            pozicijaKrajaDrugogEnt = InStr(vEntity, "</c:Object2>")
            vRelEntity = Mid(vEntity, pozicijaPocetkaDrugogEnt, pozicijaKrajaDrugogEnt
            - pozicijaPocetkaDrugogEnt)
            pozicijaPocetkaRef = InStr(vRelEntity, "Ref=")
            vRelEntity = Mid(vRelEntity, pozicijaPocetkaRef, pozicijaPocetkaRef)
            vRelEntity = OdrediNazivEntiteta(vRelEntity)

            txtEntRelations.Text = txtEntRelations.Text + vEntityRel + "-" +
            vEntityIDAtributName + "-" + vRelEntity + "; "
            vEntRel1 = "p(" + vEntityRel + ", " + vEntityIDAtributName + ")"
            listaRelacijaIEntiteta.Add(vEntRel1)
            vEntRel2 = "p(" + vEntityIDAtributName + ", " + vRelEntity + ")"
            listaRelacijaIEntiteta.Add(vEntRel2)
            vModel = Mid(vModel, pozicijaKrajaEntiteta + 1, Len(vModel) -
            pozicijaKrajaEntiteta)
        End While
    Izlaz:
    End Sub

    Private Sub btnRelatCard_Click(ByVal sender As System.Object, ByVal e As
     System.EventArgs) Handles btnRelatCard.Click
        On Error GoTo Izlaz

        Dim vEntity As String = ""
        Dim vEntityIDAtributCode As String = ""
        Dim vEntityIDAtributName As String = ""
        Dim vEntityRel As String = ""
        Dim vEntityRel2 As String = ""
        Dim vRelEntity As String = ""
        Dim vRelEntity2 As String = ""
        Dim vEntRel1 As String = ""
        Dim vEntRel2 As String = ""
        Dim pozicijaPocetkaEntiteta As Integer = 0
        Dim pozicijaKrajaEntiteta As Integer = 0
        Dim pozicijaPocetkaIDEntiteta As Integer = 0
        Dim pozicijaKrajaIDEntiteta As Integer = 0
        Dim pozicijaPocetkaPrvogEnt As Integer = 0
        Dim pozicijaPocetkaDrugogEnt As Integer = 0
```

```
Dim pozicijaPocetkaRef As Integer = 0

txtRelatCard.Text = ""
listaCardRelacija = New ArrayList
vModel = vCeoModel

While Not vModel = ""
    pozicijaPocetkaEntiteta = InStr(vModel, "<o:Relationship Id")
    pozicijaKrajaEntiteta = InStr(vModel, "</o:Relationship>")
    vEntity = Mid(vModel, pozicijaPocetkaEntiteta, pozicijaKrajaEntiteta -
    pozicijaPocetkaEntiteta + 11)
    pozicijaPocetkaIDEntiteta = InStr(vEntity, "<a:Code>")
    pozicijaKrajaIDEntiteta = InStr(vEntity, "</a:Code>")
    vEntityIDAtributCode = Mid(vEntity, pozicijaPocetkaIDEntiteta,
    pozicijaKrajaIDEntiteta - pozicijaPocetkaIDEntiteta + 9)
    pozicijaPocetkaIDEntiteta = InStr(vEntityIDAtributCode, ">")
    pozicijaKrajaIDEntiteta = InStr(vEntityIDAtributCode, "</")
    vEntityIDAtributName = Mid(vEntityIDAtributCode, pozicijaPocetkaIDEntiteta
    + 1, pozicijaKrajaIDEntiteta - pozicijaPocetkaIDEntiteta - 1)

    pozicijaPocetkaPrvogEnt = InStr(vEntity,
    "<a:Entity1ToEntity2RoleCardinality>")
    vEntityRel = Mid(vEntity, pozicijaPocetkaPrvogEnt +
    Len("<a:Entity1ToEntity2RoleCardinality>"), 3)

    pozicijaPocetkaDrugogEnt = InStr(vEntity,
    "<a:Entity2ToEntity1RoleCardinality>")
    vRelEntity = Mid(vEntity, pozicijaPocetkaDrugogEnt +
    Len("<a:Entity2ToEntity1RoleCardinality>"), 3)

    If vEntityRel = "0,1" Then
        vEntityRel = "mincard0"
        vEntityRel2 = "maxcard1"
    End If

    If vEntityRel = "1,1" Then
        vEntityRel = "mincard1"
        vEntityRel2 = "maxcard1"
    End If

    If vEntityRel = "0,n" Then
        vEntityRel = "mincard0"
        vEntityRel2 = "maxcardm"
    End If

    If vEntityRel = "1,n" Then
        vEntityRel = "mincard1"
        vEntityRel2 = "maxcardm"
    End If

    If vRelEntity = "0,1" Then
        vRelEntity = "mincard0"
        vRelEntity2 = "maxcard1"
    End If

    If vRelEntity = "1,1" Then
        vRelEntity = "mincard1"
        vRelEntity2 = "maxcard1"
    End If

    If vRelEntity = "0,n" Then
        vRelEntity = "mincard0"
        vRelEntity2 = "maxcardm"
    End If

    If vRelEntity = "1,n" Then
        vRelEntity = "mincard1"
        vRelEntity2 = "maxcardm"
    End If
```

```
            txtRelatCard.Text = txtRelatCard.Text + vRelEntity + "-" +
            vEntityIDAtributName + "-" + vEntityRel + "; "
            vEntRel1 = "p(" + vRelEntity + ", " + vEntityIDAtributName + ")"
            listaCardRelacija.Add(vEntRel1)
            vEntRel1 = "p(" + vRelEntity2 + ", " + vEntityIDAtributName + ")"
            listaCardRelacija.Add(vEntRel1)
            vEntRel2 = "p(" + vEntityIDAtributName + ", " + vEntityRel + ")"
            listaCardRelacija.Add(vEntRel2)
            vEntRel2 = "p(" + vEntityIDAtributName + ", " + vEntityRel2 + ")"
            listaCardRelacija.Add(vEntRel2)
            vModel = Mid(vModel, pozicijaKrajaEntiteta + 1, Len(vModel) -
            pozicijaKrajaEntiteta)
        End While
    Izlaz:
    End Sub

    Private Sub btnISA_Click(ByVal sender As System.Object, ByVal e As
     System.EventArgs) Handles btnISA.Click
        On Error GoTo Izlaz

        Dim vEntity As String = ""
        Dim vEntityIDAtributCode As String = ""
        Dim vEntityIDAtributName As String = ""
        Dim vEntityIdentifikatori As String = ""
        Dim vEntityIdentifikator As String = ""
        Dim vEntityAtribut As String = ""
        Dim vEntitySec As String = ""
        Dim pozicijaPocetkaEntiteta As Integer = 0
        Dim pozicijaKrajaEntiteta As Integer = 0
        Dim pozicijaPocetkaISa As Integer = 0
        Dim pozicijaKrajaISa As Integer = 0
        Dim pozicijaPocetkaIDEntiteta As Integer = 0
        Dim pozicijaKrajaIDEntiteta As Integer = 0
        Dim pozicijaIsaCard As Integer = 0
        Dim pozicijaPocetkaISaPodklasa As Integer = 0
        Dim pozicijaKrajaISaPodklasa As Integer = 0
        Dim vISa As String = ""
        Dim vISaName As String = ""
        Dim vModelEnt As String = ""
        Dim vIsaCard As String = ""
        Dim vISaPodklasa As String = ""

        listaISa = New ArrayList
        listaISaEntitet = New ArrayList
        listaISaCard = New ArrayList
        listaISaPodklasa = New ArrayList
        vModel = vCeoModel
        txtISA.Text = ""

        While Not vModel = ""
            'Naziv ISa
            pozicijaPocetkaISa = InStr(vModel, "<o:Inheritance Id=")
            If pozicijaPocetkaISa = 0 Then GoTo Izlaz
            pozicijaKrajaISa = InStr(vModel, "</o:Inheritance>")
            vISa = Mid(vModel, pozicijaPocetkaISa, pozicijaKrajaISa -
            pozicijaPocetkaISa)
            vEntity = vISa
            pozicijaPocetkaIDEntiteta = InStr(vISa, "<a:Code>")
            pozicijaKrajaIDEntiteta = InStr(vISa, "</a:Code>")
            vEntityIDAtributCode = Mid(vEntity, pozicijaPocetkaIDEntiteta,
            pozicijaKrajaIDEntiteta - pozicijaPocetkaIDEntiteta + 9)
            pozicijaPocetkaIDEntiteta = InStr(vEntityIDAtributCode, ">")
            pozicijaKrajaIDEntiteta = InStr(vEntityIDAtributCode, "</")
            vISaName = Mid(vEntityIDAtributCode, pozicijaPocetkaIDEntiteta + 1,
            pozicijaKrajaIDEntiteta - pozicijaPocetkaIDEntiteta - 1)
            pozicijaIsaCard = InStr(vISa,
            "<a:MutuallyExclusive>1</a:MutuallyExclusive>")

            'Kardinalitet ISa
            listaISa.Add("p(" + vISaName + ", inherit)")
```

```
            txtISA.Text = txtISA.Text + vISaName
            If pozicijaIsaCard > 0 Then
                listaISaCard.Add("p(" + vISaName + ", card1)")
                txtISA.Text = txtISA.Text + "(card1); "
            Else
                listaISaCard.Add("p(" + vISaName + ", cardM)")
                txtISA.Text = txtISA.Text + "(cardM); "
            End If

            'Naziv entiteta
            pozicijaPocetkaEntiteta = InStr(vEntity, "<o:Entity Ref=")
            vEntitySec = Mid(vEntity, pozicijaPocetkaEntiteta + 14, 5)
            vModelEnt = vCeoModel
            pozicijaPocetkaEntiteta = InStr(vModelEnt, "<o:Entity Id=" + vEntitySec)
            If pozicijaPocetkaEntiteta = 0 Then GoTo Izlaz
            vEntity = Mid(vModelEnt, pozicijaPocetkaEntiteta, 200)
            pozicijaPocetkaIDEntiteta = InStr(vEntity, "<a:Code>")
            pozicijaKrajaIDEntiteta = InStr(vEntity, "</a:Code>")
            vEntityIDAtributCode = Mid(vEntity, pozicijaPocetkaIDEntiteta,
            pozicijaKrajaIDEntiteta - pozicijaPocetkaIDEntiteta + 9)
            pozicijaPocetkaIDEntiteta = InStr(vEntityIDAtributCode, ">")
            pozicijaKrajaIDEntiteta = InStr(vEntityIDAtributCode, "</")
            vEntityIDAtributName = Mid(vEntityIDAtributCode, pozicijaPocetkaIDEntiteta
            + 1, pozicijaKrajaIDEntiteta - pozicijaPocetkaIDEntiteta - 1)
            txtISA.Text = txtISA.Text + vEntityIDAtributName + ";  "
            listaISaEntitet.Add("p(" + vEntityIDAtributName + ", " + vISaName + ")")

            vModel = Mid(vModel, pozicijaKrajaISa + 1, Len(vModel) - pozicijaKrajaISa)
        End While
    Izlaz:
        Dim vISaCeo As String = ""
        Dim pomvEntityIDAtributName As String = ""
        Dim pomDeoISa As String = ""
        vModel = vCeoModel
        While Not vModel = ""
            pozicijaPocetkaISa = InStr(vModel, "<o:InheritanceLink Id=")
            If pozicijaPocetkaISa = 0 Then GoTo Kraj
            pomDeoISa = Mid(vModel, pozicijaPocetkaISa, Len(vModel) -
            pozicijaPocetkaISa)
            pozicijaKrajaISa = InStr(pomDeoISa, "</o:InheritanceLink>")
            vISa = Mid(vModel, pozicijaPocetkaISa, 500)
            vISaCeo = vISa
            pozicijaPocetkaISa = InStr(vISa, "<c:Object1>")
            If pozicijaPocetkaISa = 0 Then GoTo Kraj
            pozicijaKrajaISa = InStr(vISa, "</c:Object1>")
            vISa = Mid(vISa, pozicijaPocetkaISa + 32, 5)
            pozicijaPocetkaISaPodklasa = InStr(vISaCeo, "<c:Object2>")
            If pozicijaPocetkaISaPodklasa = 0 Then GoTo Kraj
            pozicijaKrajaISaPodklasa = InStr(vISaCeo, "</c:Object2>")
            vISaPodklasa = Mid(vISaCeo, pozicijaPocetkaISaPodklasa + 27, 5)

            vModelEnt = vCeoModel
            pozicijaPocetkaEntiteta = InStr(vModelEnt, "<o:Entity Id=" + vISaPodklasa)
            If pozicijaPocetkaEntiteta = 0 Then GoTo Kraj
            vEntity = Mid(vModelEnt, pozicijaPocetkaEntiteta, 200)
            pozicijaPocetkaIDEntiteta = InStr(vEntity, "<a:Code>")
            pozicijaKrajaIDEntiteta = InStr(vEntity, "</a:Code>")
            vEntityIDAtributCode = Mid(vEntity, pozicijaPocetkaIDEntiteta,
            pozicijaKrajaIDEntiteta - pozicijaPocetkaIDEntiteta + 9)
            pozicijaPocetkaIDEntiteta = InStr(vEntityIDAtributCode, ">")
            pozicijaKrajaIDEntiteta = InStr(vEntityIDAtributCode, "</")
            vEntityIDAtributName = Mid(vEntityIDAtributCode, pozicijaPocetkaIDEntiteta
            + 1, pozicijaKrajaIDEntiteta - pozicijaPocetkaIDEntiteta - 1)
            pomvEntityIDAtributName = vEntityIDAtributName

            vModelEnt = vCeoModel
            If InStr(vISa, "/") Then
                vISa = Microsoft.VisualBasic.Left(vISa, 3) + """"
            End If
            pozicijaPocetkaEntiteta = InStr(vModelEnt, "<o:Inheritance Id=" + vISa)
```

```
            If pozicijaPocetkaEntiteta = 0 Then GoTo Kraj
            vEntity = Mid(vModelEnt, pozicijaPocetkaEntiteta, 200)
            pozicijaPocetkaIDEntiteta = InStr(vEntity, "<a:Code>")
            pozicijaKrajaIDEntiteta = InStr(vEntity, "</a:Code>")
            vEntityIDAtributCode = Mid(vEntity, pozicijaPocetkaIDEntiteta,
            pozicijaKrajaIDEntiteta - pozicijaPocetkaIDEntiteta + 9)
            pozicijaPocetkaIDEntiteta = InStr(vEntityIDAtributCode, ">")
            pozicijaKrajaIDEntiteta = InStr(vEntityIDAtributCode, "</")
            vEntityIDAtributName = Mid(vEntityIDAtributCode, pozicijaPocetkaIDEntiteta
            + 1, pozicijaKrajaIDEntiteta - pozicijaPocetkaIDEntiteta - 1)
            txtISA.Text = txtISA.Text + "(" + vEntityIDAtributName + ", " +
            pomvEntityIDAtributName + "); "
            listaISaPodklasa.Add("p(" + vEntityIDAtributName + ", " +
            pomvEntityIDAtributName + ")")

            pozicijaKrajaISa = InStr(vModel, "</o:InheritanceLink>")
            vModel = Mid(vModel, pozicijaKrajaISa + 1, Len(vModel) - pozicijaKrajaISa)
        End While
    Kraj:
    End Sub

    Private Sub btnDependent_Click(ByVal sender As System.Object, ByVal e As
    System.EventArgs) Handles btnDependent.Click
        On Error GoTo Izlaz

        Dim vEntity As String = ""
        Dim vEntityIDAtributCode As String = ""
        Dim vEntityIDAtributName As String = ""
        Dim vEntityRel As String = ""
        Dim vRelEntity As String = ""
        Dim vEntRel1 As String = ""
        Dim vEntRel2 As String = ""
        Dim pozicijaPocetkaEntiteta As Integer = 0
        Dim pozicijaKrajaEntiteta As Integer = 0
        Dim pozicijaPocetkaIDEntiteta As Integer = 0
        Dim pozicijaKrajaIDEntiteta As Integer = 0
        Dim pozicijaPocetkaPrvogEnt As Integer = 0
        Dim pozicijaPocetkaDrugogEnt As Integer = 0
        Dim pozicijaPocetkaRef As Integer = 0
        Dim pozicijaDependent As Integer = 0
        Dim vrstaDependent As String = ""
        Dim redniDependent As String = ""

        txtDependent.Text = ""
        listaDepenRelacija = New ArrayList
        vModel = vCeoModel

        While Not vModel = ""
            pozicijaPocetkaEntiteta = InStr(vModel, "<o:Relationship Id")
            pozicijaKrajaEntiteta = InStr(vModel, "</o:Relationship>")
            If pozicijaPocetkaEntiteta = 0 Then GoTo Izlaz

            vEntity = Mid(vModel, pozicijaPocetkaEntiteta, pozicijaKrajaEntiteta -
            pozicijaPocetkaEntiteta + 11)
            pozicijaPocetkaIDEntiteta = InStr(vEntity, "<a:Code>")
            pozicijaKrajaIDEntiteta = InStr(vEntity, "</a:Code>")
            vEntityIDAtributCode = Mid(vEntity, pozicijaPocetkaIDEntiteta,
            pozicijaKrajaIDEntiteta - pozicijaPocetkaIDEntiteta + 9)
            pozicijaPocetkaIDEntiteta = InStr(vEntityIDAtributCode, ">")
            pozicijaKrajaIDEntiteta = InStr(vEntityIDAtributCode, "</")
            vEntityIDAtributName = Mid(vEntityIDAtributCode, pozicijaPocetkaIDEntiteta
            + 1, pozicijaKrajaIDEntiteta - pozicijaPocetkaIDEntiteta - 1)

            pozicijaDependent = InStr(vEntity, "<a:DependentRole>")
            redniDependent = Mid(vEntity, pozicijaDependent + 17, 1)

            pozicijaPocetkaPrvogEnt = InStr(vEntity,
            "<a:Entity1ToEntity2RoleCardinality>")
            vEntityRel = Mid(vEntity, pozicijaPocetkaPrvogEnt +
            Len("<a:Entity1ToEntity2RoleCardinality>"), 3)
```

```
            pozicijaPocetkaDrugogEnt = InStr(vEntity,
            "<a:Entity2ToEntity1RoleCardinality>")
            vRelEntity = Mid(vEntity, pozicijaPocetkaDrugogEnt +
            Len("<a:Entity2ToEntity1RoleCardinality>"), 3)

            If pozicijaPocetkaPrvogEnt < pozicijaPocetkaDrugogEnt And redniDependent =
                "B" Then
                vrstaDependent = "p(dependent, " + vEntityIDAtributName + ")"
            End If
            If pozicijaPocetkaPrvogEnt < pozicijaPocetkaDrugogEnt And redniDependent =
                "A" Then
                vrstaDependent = "p(" + vEntityIDAtributName + ", dependent)"
            End If
            If pozicijaPocetkaPrvogEnt > pozicijaPocetkaDrugogEnt And redniDependent =
                "B" Then
                vrstaDependent = "p(dependent, " + vEntityIDAtributName + ")"
            End If
            If pozicijaPocetkaPrvogEnt > pozicijaPocetkaDrugogEnt And redniDependent =
                "A" Then
                vrstaDependent = "p(" + vEntityIDAtributName + ", dependent)"
            End If
            If pozicijaDependent > 0 Then
                listaDepenRelacija.Add(vrstaDependent)
                txtDependent.Text = txtDependent.Text + vEntityIDAtributName + "-
                dependent; "
            End If

            vModel = Mid(vModel, pozicijaKrajaEntiteta + 1, Len(vModel) -
             pozicijaKrajaEntiteta)
        End While

    Izlaz:
    End Sub

    Private Sub btnIDAtr_Click(ByVal sender As System.Object, ByVal e As
    System.EventArgs) Handles btnIDAtr.Click
        On Error GoTo Izlaz

        Dim vEntity As String = ""
        Dim vEntityIDAtributCode As String = ""
        Dim vEntityIDAtributName As String = ""
        Dim vEntityIdentifikator As String = ""
        Dim vEntityAtribut As String = ""
        Dim vEntityPart As String = ""
        Dim pozicijaPocetkaEntiteta As Integer = 0
        Dim pozicijaKrajaEntiteta As Integer = 0
        Dim pozicijaPocetkaIdentifikatora As Integer = 0
        Dim pozicijaKrajaIdentifikatora As Integer = 0
        Dim pozicijaPocetkaID As Integer = 0
        Dim pozicijaKrajaID As Integer = 0
        Dim pozicijaPocetkaRef As Integer = 0
        Dim pozicijaPocetkaAtr As Integer = 0
        Dim pozicijaKrajaAtr As Integer = 0
        Dim pozicijaPocetkaIDAtr As Integer = 0
        Dim pozicijaKrajaIDAtr As Integer = 0
        Dim imaIDAtr As Integer = 0

        txtIDAtr.Text = ""
        listaIdentAtributa = New ArrayList
        vModel = vCeoModel

        While Not vModel = ""
            pozicijaPocetkaEntiteta = InStr(vModel, "<o:Entity Id")
            If pozicijaPocetkaEntiteta = 0 Then GoTo Izlaz
            pozicijaKrajaEntiteta = InStr(vModel, "</o:Entity>")
            vEntity = Mid(vModel, pozicijaPocetkaEntiteta, pozicijaKrajaEntiteta -
            pozicijaPocetkaEntiteta + 11)
            While Not vEntity = ""
                imaIDAtr = InStr(vEntity, "<c:Identifier.Attributes>")
```

```
 Dalje:     pozicijaPocetkaIdentifikatora = InStr(vEntity, "<o:EntityAttribute
           Ref=")
           If pozicijaPocetkaIdentifikatora = 0 Then GoTo Nastavak
           vEntityIdentifikator = Mid(vEntity, pozicijaPocetkaIdentifikatora +
           23, 5)
           'Odredjivanje naziva atributa
           pozicijaPocetkaIDAtr = InStr(vEntity, "<o:EntityAttribute Id=" +
            vEntityIdentifikator)
           vEntityPart = Mid(vEntity, pozicijaPocetkaIDAtr, 400)
           pozicijaPocetkaAtr = InStr(vEntityPart, "<o:DataItem Ref")
           If pozicijaPocetkaAtr = 0 Then GoTo Nastavak
           vEntityAtribut = Mid(vEntityPart, pozicijaPocetkaAtr + 16, 5)
           vEntityAtribut = OdrediNazivIDAtributa(vEntityAtribut)
           listaIdentAtributa.Add("p(" + vEntityAtribut + ", idatr)")
           txtIDAtr.Text = txtIDAtr.Text + vEntityAtribut + "; "
           vEntity = Mid(vEntity, pozicijaPocetkaIdentifikatora + 28,
            Len(vEntity) - pozicijaPocetkaIdentifikatora + 28)
           GoTo Dalje
       End While

Nastavak:  vModel = Mid(vModel, pozicijaKrajaEntiteta + 1, Len(vModel) -
           pozicijaKrajaEntiteta)
    End While
Izlaz:
End Sub

Private Sub btnCreateOntoClauses_Click(ByVal sender As System.Object, ByVal e As
 System.EventArgs)
    On Error GoTo Izlaz

    Dim vEntity As String = ""
    Dim vEntityName As String = ""
    Dim pozicijaPocetkaEntiteta As Integer = 0
    Dim pozicijaKrajaEntiteta As Integer = 0

    listaEntiteta = New ArrayList
    vModel = vCelaOntologija

    While Not vModel = ""
        pozicijaPocetkaEntiteta = InStr(vModel, "<ObjectProperty rdf:about=")
        vEntity = Mid(vModel, pozicijaPocetkaEntiteta, pozicijaPocetkaEntiteta +
        49)
        pozicijaKrajaEntiteta = InStr(vModel, "</o:Entity>")
        listaEntiteta.Add(vEntityName)
        vModel = Mid(vModel, pozicijaKrajaEntiteta + 1, Len(vModel) -
        pozicijaKrajaEntiteta)
    End While
Izlaz:
End Sub

Private Sub btnClasses_Click(ByVal sender As System.Object, ByVal e As
 System.EventArgs) Handles btnClasses.Click
    On Error GoTo Izlaz

    Dim vEntity As String = ""
    Dim vEntityName As String = ""
    Dim pozicijaPocetkaEntiteta As Integer = 0
    Dim pozicijaKrajaEntiteta As Integer = 0
    Dim pozicijaPocetkaNaziva As Integer = 0
    Dim pozicijaKrajaNaziva As Integer = 0

    vModel = vCelaOntologija

    While Not vModel = ""
        pozicijaPocetkaEntiteta = InStr(vModel, "<Class IRI=")
        pozicijaKrajaEntiteta = InStr(vModel, "</Declaration>")
        vEntity = Mid(vModel, pozicijaPocetkaEntiteta, pozicijaKrajaEntiteta -
        pozicijaPocetkaEntiteta + 14)
        pozicijaPocetkaNaziva = InStr(vEntity, "#")
        pozicijaKrajaNaziva = InStr(vEntity, "/>")
```

```
            vEntityName = Mid(vEntity, pozicijaPocetkaNaziva + 1, pozicijaKrajaNaziva
            - pozicijaPocetkaNaziva - 2)
            richtxtClauses.Text = richtxtClauses.Text + "rdf(" +
            vEntityName.ToString.ToLower + ", type, class)." + Chr(13)
            fileClauses.WriteLine("rdf(" + vEntityName.ToString.ToLower + ", type,
            class).")
            vModel = Mid(vModel, pozicijaKrajaEntiteta + 20, Len(vModel) -
            pozicijaKrajaEntiteta)
        End While
    Izlaz:
    End Sub

    Private Sub btnObjProp_Click(ByVal sender As System.Object, ByVal e As
    System.EventArgs) Handles btnObjProp.Click
        On Error GoTo Izlaz

        Dim vEntity As String = ""
        Dim vEntityName As String = ""
        Dim pozicijaPocetkaEntiteta As Integer = 0
        Dim pozicijaKrajaEntiteta As Integer = 0
        Dim pozicijaPocetkaNaziva As Integer = 0
        Dim pozicijaKrajaNaziva As Integer = 0

        vModel = vCelaOntologija
        pozicijaPocetkaEntiteta = InStr(vModel, "<ObjectProperty IRI=")
        vModel = Mid(vModel, pozicijaPocetkaEntiteta)

        While Not vModel = ""
            pozicijaPocetkaEntiteta = InStr(vModel, "<ObjectProperty IRI=")
            vEntity = Mid(vModel, pozicijaPocetkaEntiteta)
            pozicijaKrajaEntiteta = InStr(vModel, "</Declaration>")
            vEntity = Mid(vModel, pozicijaPocetkaEntiteta, pozicijaKrajaEntiteta -
            pozicijaPocetkaEntiteta + 14)
            pozicijaPocetkaNaziva = InStr(vEntity, "#")
            pozicijaKrajaNaziva = InStr(vEntity, "/>")
            vEntityName = Mid(vEntity, pozicijaPocetkaNaziva + 1, pozicijaKrajaNaziva
            - pozicijaPocetkaNaziva - 2)
            richtxtClauses.Text = richtxtClauses.Text + "rdf(" +
            vEntityName.ToString.ToLower + ", type, objectproperty)." + Chr(13)
            fileClauses.WriteLine("rdf(" + vEntityName.ToString.ToLower + ", type,
            objectproperty).")
            vModel = Mid(vModel, pozicijaKrajaEntiteta + 20, Len(vModel) -
            pozicijaKrajaEntiteta)
        End While
    Izlaz:
    End Sub

    Private Sub btnDataProp_Click(ByVal sender As System.Object, ByVal e As
    System.EventArgs) Handles btnDataProp.Click
        On Error GoTo Izlaz

        Dim vEntity As String = ""
        Dim vEntityName As String = ""
        Dim pozicijaPocetkaEntiteta As Integer = 0
        Dim pozicijaKrajaEntiteta As Integer = 0
        Dim pozicijaPocetkaNaziva As Integer = 0
        Dim pozicijaKrajaNaziva As Integer = 0

        vModel = vCelaOntologija
        pozicijaPocetkaEntiteta = InStr(vModel, "<DataProperty IRI=")
        vModel = Mid(vModel, pozicijaPocetkaEntiteta)

        While Not vModel = ""
            pozicijaPocetkaEntiteta = InStr(vModel, "<DataProperty IRI=")
            vEntity = Mid(vModel, pozicijaPocetkaEntiteta)
            pozicijaKrajaEntiteta = InStr(vModel, "</Declaration>")
            vEntity = Mid(vModel, pozicijaPocetkaEntiteta, pozicijaKrajaEntiteta -
            pozicijaPocetkaEntiteta + 14)
            pozicijaPocetkaNaziva = InStr(vEntity, "#")
            pozicijaKrajaNaziva = InStr(vEntity, "/>")
```

```
            vEntityName = Mid(vEntity, pozicijaPocetkaNaziva + 1, pozicijaKrajaNaziva
            - pozicijaPocetkaNaziva - 2)
            richtxtClauses.Text = richtxtClauses.Text + "rdf(" +
            vEntityName.ToString.ToLower + ", type, dataproperty)." + Chr(13)
            fileClauses.WriteLine("rdf(" + vEntityName.ToString.ToLower + ", type,
            dataproperty).")
            vModel = Mid(vModel, pozicijaKrajaEntiteta + 20, Len(vModel) -
            pozicijaKrajaEntiteta)
        End While
    Izlaz:
    End Sub

    Private Sub btnIndividual_Click(ByVal sender As System.Object, ByVal e As
    System.EventArgs) Handles btnIndividual.Click
        On Error GoTo Izlaz

        Dim vEntity As String = ""
        Dim vEntityName As String = ""
        Dim pozicijaPocetkaEntiteta As Integer = 0
        Dim pozicijaKrajaEntiteta As Integer = 0
        Dim pozicijaPocetkaNaziva As Integer = 0
        Dim pozicijaKrajaNaziva As Integer = 0

        vModel = vCelaOntologija
        pozicijaPocetkaEntiteta = InStr(vModel, "<NamedIndividual IRI=")
        vModel = Mid(vModel, pozicijaPocetkaEntiteta)

        While Not vModel = ""
            pozicijaPocetkaEntiteta = InStr(vModel, "<NamedIndividual IRI=")
            vEntity = Mid(vModel, pozicijaPocetkaEntiteta)
            pozicijaKrajaEntiteta = InStr(vModel, "</Declaration>")
            vEntity = Mid(vModel, pozicijaPocetkaEntiteta, pozicijaKrajaEntiteta -
            pozicijaPocetkaEntiteta + 14)
            pozicijaPocetkaNaziva = InStr(vEntity, "#")
            pozicijaKrajaNaziva = InStr(vEntity, "/>")
            vEntityName = Mid(vEntity, pozicijaPocetkaNaziva + 1, pozicijaKrajaNaziva
            - pozicijaPocetkaNaziva - 2)
            richtxtClauses.Text = richtxtClauses.Text + "rdf(" +
            vEntityName.ToString.ToLower + ", type, namedindividual)." + Chr(13)
            fileClauses.WriteLine("rdf(" + vEntityName.ToString.ToLower + ", type,
            namedindividual).")
            vModel = Mid(vModel, pozicijaKrajaEntiteta + 20, Len(vModel) -
            pozicijaKrajaEntiteta)
        End While
    Izlaz:
    End Sub

    Private Sub btnEquivProp_Click(ByVal sender As System.Object, ByVal e As
    System.EventArgs) Handles btnEquivProp.Click
        On Error GoTo Izlaz

        Dim vEntity As String = ""
        Dim vEntityName1 As String = ""
        Dim vEntityName2 As String = ""
        Dim pozicijaPocetkaEntiteta As Integer = 0
        Dim pozicijaKrajaEntiteta As Integer = 0
        Dim pozicijaPocetkaNaziva As Integer = 0
        Dim pozicijaKrajaNaziva As Integer = 0

        vModel = vCelaOntologija
        pozicijaPocetkaEntiteta = InStr(vModel, "<EquivalentObjectProperties>")
        vModel = Mid(vModel, pozicijaPocetkaEntiteta)

        While Not vModel = ""
            pozicijaPocetkaEntiteta = InStr(vModel, "<ObjectProperty IRI=")
            vEntity = Mid(vModel, pozicijaPocetkaEntiteta)
            pozicijaKrajaEntiteta = InStr(vModel, "</EquivalentObjectProperties>")
            If (pozicijaKrajaEntiteta < pozicijaPocetkaEntiteta) And
                (pozicijaKrajaEntiteta <> 0) And (pozicijaPocetkaEntiteta <> 0) Then
```

```
            pozicijaPocetkaEntiteta = InStr(vModel,
            "<EquivalentObjectProperties>")
            vModel = Mid(vModel, pozicijaPocetkaEntiteta)
            pozicijaPocetkaEntiteta = 1
            pozicijaKrajaEntiteta = InStr(vModel, "</EquivalentObjectProperties>")
        End If
        vEntity = Mid(vModel, pozicijaPocetkaEntiteta, pozicijaKrajaEntiteta -
        pozicijaPocetkaEntiteta + 30)
        pozicijaPocetkaNaziva = InStr(vEntity, "#")
        pozicijaKrajaNaziva = InStr(vEntity, "/>")
        vEntityName1 = Mid(vEntity, pozicijaPocetkaNaziva + 1, pozicijaKrajaNaziva
        - pozicijaPocetkaNaziva - 2)
        vEntity = Mid(vEntity, pozicijaKrajaNaziva)
        pozicijaPocetkaNaziva = 0
        pozicijaKrajaNaziva = 0
        pozicijaPocetkaEntiteta = InStr(vEntity, "<ObjectProperty IRI=")
        If pozicijaPocetkaEntiteta = 0 Then
            GoTo Dalje
        End If
        pozicijaKrajaEntiteta = InStr(vEntity, "</EquivalentObjectProperties>")
        vEntity = Mid(vEntity, pozicijaPocetkaEntiteta, pozicijaKrajaEntiteta -
        pozicijaPocetkaEntiteta + 30)
        pozicijaPocetkaNaziva = InStr(vEntity, "#")
        pozicijaKrajaNaziva = InStr(vEntity, "/>")
        vEntityName2 = Mid(vEntity, pozicijaPocetkaNaziva + 1, pozicijaKrajaNaziva
        - pozicijaPocetkaNaziva - 2)

        richtxtClauses.Text = richtxtClauses.Text + "rdf(" +
        vEntityName1.ToString.ToLower + ", equivalentobjectproperties, " +
        vEntityName2.ToString.ToLower + ")." + Chr(13)
        fileClauses.WriteLine("rdf(" + vEntityName1.ToString.ToLower + ",
        equivalentobjectproperties, " + vEntityName2.ToString.ToLower + ").")
Dalje:
        vModel = Mid(vModel, pozicijaKrajaEntiteta + 20, Len(vModel) -
        pozicijaKrajaEntiteta)
    End While
Izlaz:
End Sub

Private Sub btnSubclasses_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnSubclasses.Click
    On Error GoTo Izlaz

    Dim vEntity As String = ""
    Dim vEntityName1 As String = ""
    Dim vEntityName2 As String = ""
    Dim pozicijaPocetkaEntiteta As Integer = 0
    Dim pozicijaKrajaEntiteta As Integer = 0
    Dim pozicijaPocetkaNaziva As Integer = 0
    Dim pozicijaKrajaNaziva As Integer = 0

    vModel = vCelaOntologija
    pozicijaPocetkaEntiteta = InStr(vModel, "<SubClassOf>")
    vModel = Mid(vModel, pozicijaPocetkaEntiteta)

    While Not vModel = ""
        pozicijaPocetkaEntiteta = InStr(vModel, "<Class IRI=")
        vEntity = Mid(vModel, pozicijaPocetkaEntiteta)
        pozicijaKrajaEntiteta = InStr(vModel, "</SubClassOf>")
        If (pozicijaKrajaEntiteta < pozicijaPocetkaEntiteta) And
          (pozicijaKrajaEntiteta <> 0) And (pozicijaPocetkaEntiteta <> 0) Then
            pozicijaPocetkaEntiteta = InStr(vModel, "<SubClassOf>")
            vModel = Mid(vModel, pozicijaPocetkaEntiteta)
            pozicijaPocetkaEntiteta = 1
            pozicijaKrajaEntiteta = InStr(vModel, "</SubClassOf>")
        End If
        vEntity = Mid(vModel, pozicijaPocetkaEntiteta, pozicijaKrajaEntiteta -
        pozicijaPocetkaEntiteta + 14)
        pozicijaPocetkaNaziva = InStr(vEntity, "#")
        pozicijaKrajaNaziva = InStr(vEntity, "/>")
```

```
            vEntityName1 = Mid(vEntity, pozicijaPocetkaNaziva + 1, pozicijaKrajaNaziva
            - pozicijaPocetkaNaziva - 2)
            vEntity = Mid(vEntity, pozicijaKrajaNaziva)
            pozicijaPocetkaNaziva = 0
            pozicijaKrajaNaziva = 0
            pozicijaPocetkaEntiteta = InStr(vEntity, "<Class IRI=")
            If pozicijaPocetkaEntiteta = 0 Then
                GoTo Dalje
            End If
            pozicijaKrajaEntiteta = InStr(vEntity, "</SubClassOf>")
            vEntity = Mid(vEntity, pozicijaPocetkaEntiteta, pozicijaKrajaEntiteta -
            pozicijaPocetkaEntiteta + 14)
            pozicijaPocetkaNaziva = InStr(vEntity, "#")
            pozicijaKrajaNaziva = InStr(vEntity, "/>")
            vEntityName2 = Mid(vEntity, pozicijaPocetkaNaziva + 1, pozicijaKrajaNaziva
            - pozicijaPocetkaNaziva - 2)

            richtxtClauses.Text = richtxtClauses.Text + "rdf(" +
            vEntityName1.ToString.ToLower + ", subclass, " +
            vEntityName2.ToString.ToLower + ")." + Chr(13)
            fileClauses.WriteLine("rdf(" + vEntityName1.ToString.ToLower + ",
            subclass, " + vEntityName2.ToString.ToLower + ").")
Dalje:
            vModel = Mid(vModel, pozicijaKrajaEntiteta + 20, Len(vModel) -
            pozicijaKrajaEntiteta)
        End While
Izlaz:
End Sub

Private Sub btDtTpRange_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btDtTpRange.Click
        On Error GoTo Izlaz

        Dim vEntity As String = ""
        Dim vEntityName1 As String = ""
        Dim vEntityName2 As String = ""
        Dim pozicijaPocetkaEntiteta As Integer = 0
        Dim pozicijaKrajaEntiteta As Integer = 0
        Dim pozicijaPocetkaNaziva As Integer = 0
        Dim pozicijaKrajaNaziva As Integer = 0

        vModel = vCelaOntologija
        pozicijaPocetkaEntiteta = InStr(vModel, "<DataPropertyRange>")
        vModel = Mid(vModel, pozicijaPocetkaEntiteta)

        While Not vModel = ""
            pozicijaPocetkaEntiteta = InStr(vModel, "<DataProperty IRI=")
            vEntity = Mid(vModel, pozicijaPocetkaEntiteta)
            pozicijaKrajaEntiteta = InStr(vModel, "</DataPropertyRange>")
            vEntity = Mid(vModel, pozicijaPocetkaEntiteta, pozicijaKrajaEntiteta -
            pozicijaPocetkaEntiteta + 14)
            pozicijaPocetkaNaziva = InStr(vEntity, "#")
            pozicijaKrajaNaziva = InStr(vEntity, "/>")
            vEntityName1 = Mid(vEntity, pozicijaPocetkaNaziva + 1, pozicijaKrajaNaziva
            - pozicijaPocetkaNaziva - 2)
            pozicijaPocetkaEntiteta = InStr(vModel, "<Datatype abbreviatedIRI=")
            vEntity = Mid(vModel, pozicijaPocetkaEntiteta, pozicijaKrajaEntiteta -
            pozicijaPocetkaEntiteta + 14)
            pozicijaPocetkaNaziva = InStr(vEntity, ":")
            pozicijaKrajaNaziva = InStr(vEntity, "/>")
            vEntityName2 = Mid(vEntity, pozicijaPocetkaNaziva + 1, pozicijaKrajaNaziva
            - pozicijaPocetkaNaziva - 2)

            richtxtClauses.Text = richtxtClauses.Text + "rdf(" +
            vEntityName1.ToString.ToLower + ", range, " +
            vEntityName2.ToString.ToLower + ")." + Chr(13)
            fileClauses.WriteLine("rdf(" + vEntityName1.ToString.ToLower + ", range, "
            + vEntityName2.ToString.ToLower + ").")
            vModel = Mid(vModel, pozicijaKrajaEntiteta + 20, Len(vModel) -
            pozicijaKrajaEntiteta)
```

```
    End While
Izlaz:
End Sub

Private Sub btnDtPropAss_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnDtPropAss.Click
    On Error GoTo Izlaz

    Dim vEntity As String = ""
    Dim vEntityName1 As String = ""
    Dim vEntityName2 As String = ""
    Dim pozicijaPocetkaEntiteta As Integer = 0
    Dim pozicijaKrajaEntiteta As Integer = 0
    Dim pozicijaPocetkaNaziva As Integer = 0
    Dim pozicijaKrajaNaziva As Integer = 0

    vModel = vCelaOntologija
    pozicijaPocetkaEntiteta = InStr(vModel, "<DataPropertyAssertion>")
    vModel = Mid(vModel, pozicijaPocetkaEntiteta)

    While Not vModel = ""
        pozicijaPocetkaEntiteta = InStr(vModel, "<DataProperty IRI=")
        vEntity = Mid(vModel, pozicijaPocetkaEntiteta)
        pozicijaKrajaEntiteta = InStr(vModel, "</DataPropertyAssertion>")
        vEntity = Mid(vModel, pozicijaPocetkaEntiteta, pozicijaKrajaEntiteta -
        pozicijaPocetkaEntiteta + 18)
        pozicijaPocetkaNaziva = InStr(vEntity, "#")
        pozicijaKrajaNaziva = InStr(vEntity, "/>")
        vEntityName1 = Mid(vEntity, pozicijaPocetkaNaziva + 1, pozicijaKrajaNaziva
        - pozicijaPocetkaNaziva - 2)
        pozicijaPocetkaEntiteta = InStr(vModel, "<NamedIndividual IRI=")
        vEntity = Mid(vModel, pozicijaPocetkaEntiteta, pozicijaKrajaEntiteta -
        pozicijaPocetkaEntiteta + 18)
        pozicijaPocetkaNaziva = InStr(vEntity, "#")
        pozicijaKrajaNaziva = InStr(vEntity, "/>")
        vEntityName2 = Mid(vEntity, pozicijaPocetkaNaziva + 1, pozicijaKrajaNaziva
        - pozicijaPocetkaNaziva - 2)

        richtxtClauses.Text = richtxtClauses.Text + "rdf(" +
        vEntityName1.ToString.ToLower + ", datapropertyassertion, " +
        vEntityName2.ToString.ToLower + ")." + Chr(13)
        fileClauses.WriteLine("rdf(" + vEntityName1.ToString.ToLower + ",
        datapropertyassertion, " + vEntityName2.ToString.ToLower + ").")
        vModel = Mid(vModel, pozicijaKrajaEntiteta + 20, Len(vModel) -
        pozicijaKrajaEntiteta)
    End While
Izlaz:
End Sub

Private Sub btnClsAss_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnClsAss.Click
    On Error GoTo Izlaz

    Dim vEntity As String = ""
    Dim vEntityName1 As String = ""
    Dim vEntityName2 As String = ""
    Dim pozicijaPocetkaEntiteta As Integer = 0
    Dim pozicijaKrajaEntiteta As Integer = 0
    Dim pozicijaPocetkaNaziva As Integer = 0
    Dim pozicijaKrajaNaziva As Integer = 0

    vModel = vCelaOntologija
    pozicijaPocetkaEntiteta = InStr(vModel, "<ClassAssertion>")
    vModel = Mid(vModel, pozicijaPocetkaEntiteta)

    While Not vModel = ""
        pozicijaPocetkaEntiteta = InStr(vModel, "<Class IRI=")
        vEntity = Mid(vModel, pozicijaPocetkaEntiteta)
        pozicijaKrajaEntiteta = InStr(vModel, "</ClassAssertion>")
```

```
            vEntity = Mid(vModel, pozicijaPocetkaEntiteta, pozicijaKrajaEntiteta -
            pozicijaPocetkaEntiteta + 18)
            pozicijaPocetkaNaziva = InStr(vEntity, "#")
            pozicijaKrajaNaziva = InStr(vEntity, "/>")
            vEntityName1 = Mid(vEntity, pozicijaPocetkaNaziva + 1, pozicijaKrajaNaziva
            - pozicijaPocetkaNaziva - 2)
            pozicijaPocetkaEntiteta = InStr(vModel, "<NamedIndividual IRI=")
            vEntity = Mid(vModel, pozicijaPocetkaEntiteta, pozicijaKrajaEntiteta -
            pozicijaPocetkaEntiteta + 18)
            pozicijaPocetkaNaziva = InStr(vEntity, "#")
            pozicijaKrajaNaziva = InStr(vEntity, "/>")
            vEntityName2 = Mid(vEntity, pozicijaPocetkaNaziva + 1, pozicijaKrajaNaziva
            - pozicijaPocetkaNaziva - 2)

            richtxtClauses.Text = richtxtClauses.Text + "rdf(" +
            vEntityName2.ToString.ToLower + ", classassertion, " +
            vEntityName1.ToString.ToLower + ")." + Chr(13)
            fileClauses.WriteLine("rdf(" + vEntityName2.ToString.ToLower + ",
            classassertion, " + vEntityName1.ToString.ToLower + ").")
            vModel = Mid(vModel, pozicijaKrajaEntiteta + 20, Len(vModel) -
            pozicijaKrajaEntiteta)
        End While
Izlaz:
End Sub

Private Sub btnIndProp_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnIndProp.Click
        On Error GoTo Izlaz

        Dim vEntity As String = ""
        Dim vEntityName1 As String = ""
        Dim vEntityName2 As String = ""
        Dim vEntityName3 As String = ""
        Dim pozicijaPocetkaEntiteta As Integer = 0
        Dim pozicijaKrajaEntiteta As Integer = 0
        Dim pozicijaPocetkaNaziva As Integer = 0
        Dim pozicijaKrajaNaziva As Integer = 0

        vModel = vCelaOntologija
        pozicijaPocetkaEntiteta = InStr(vModel, "<ObjectPropertyAssertion>")
        vModel = Mid(vModel, pozicijaPocetkaEntiteta)

        While Not vModel = ""
            pozicijaPocetkaEntiteta = InStr(vModel, "<ObjectProperty IRI=")
            vEntity = Mid(vModel, pozicijaPocetkaEntiteta)
            pozicijaKrajaEntiteta = InStr(vModel, "</ObjectPropertyAssertion>")
            vEntity = Mid(vModel, pozicijaPocetkaEntiteta, pozicijaKrajaEntiteta -
            pozicijaPocetkaEntiteta + 25)
            pozicijaPocetkaNaziva = InStr(vEntity, "#")
            pozicijaKrajaNaziva = InStr(vEntity, "/>")
            vEntityName1 = Mid(vEntity, pozicijaPocetkaNaziva + 1, pozicijaKrajaNaziva
            - pozicijaPocetkaNaziva - 2)

            pozicijaPocetkaEntiteta = InStr(vModel, "<NamedIndividual IRI=")
            vEntity = Mid(vModel, pozicijaPocetkaEntiteta, pozicijaKrajaEntiteta -
            pozicijaPocetkaEntiteta + 25)
            pozicijaPocetkaNaziva = InStr(vEntity, "#")
            pozicijaKrajaNaziva = InStr(vEntity, "/>")
            vEntityName2 = Mid(vEntity, pozicijaPocetkaNaziva + 1, pozicijaKrajaNaziva
            - pozicijaPocetkaNaziva - 2)

            pozicijaPocetkaEntiteta = InStr(vModel, "<NamedIndividual IRI=")
            vEntity = Mid(vModel, pozicijaPocetkaEntiteta + 60, pozicijaKrajaEntiteta
            - pozicijaPocetkaEntiteta + 25)
            pozicijaPocetkaNaziva = InStr(vEntity, "#")
            pozicijaKrajaNaziva = InStr(vEntity, "/>")
            vEntityName3 = Mid(vEntity, pozicijaPocetkaNaziva + 1, pozicijaKrajaNaziva
            - pozicijaPocetkaNaziva - 2)
```

```
            richtxtClauses.Text = richtxtClauses.Text + "rdf(" +
            vEntityName3.ToString.ToLower + ", " + vEntityName1.ToString.ToLower + ",
            " + vEntityName2.ToString.ToLower + ")." + Chr(13)
            fileClauses.WriteLine("rdf(" + vEntityName3.ToString.ToLower + ", " +
            vEntityName1.ToString.ToLower + ", " + vEntityName2.ToString.ToLower +
            ").")
            vModel = Mid(vModel, pozicijaKrajaEntiteta + 20, Len(vModel) -
            pozicijaKrajaEntiteta)
        End While
    Izlaz:
    End Sub

    Private Sub btnOntologyMapping_Click(ByVal sender As System.Object, ByVal e As
    System.EventArgs) Handles btnOntologyMapping.Click
        btnClasses.PerformClick()
        btnObjProp.PerformClick()
        btnDataProp.PerformClick()
        btnIndividual.PerformClick()
        btnEquivProp.PerformClick()
        btnSubclasses.PerformClick()
        btDtTpRange.PerformClick()
        btnClsAss.PerformClick()
        btnDtPropAss.PerformClick()
        btnIndProp.PerformClick()
        btnObjPropCard.PerformClick()
    End Sub

    Private Sub btnObjPropCard_Click(ByVal sender As System.Object, ByVal e As
    System.EventArgs) Handles btnObjPropCard.Click
        On Error GoTo Izlaz

        Dim vEntity As String = ""
        Dim vEntity2 As String = ""
        Dim vEntity3 As String = ""
        Dim vEntity4 As String = ""
        Dim vEntityName1 As String = ""
        Dim vEntityName2 As String = ""
        Dim vEntityName3 As String = ""
        Dim vEntityName4 As String = ""
        Dim pronadjencard = False
        Dim pozicijaPocetkaEntiteta As Integer = 0
        Dim pozicijaKrajaEntiteta As Integer = 0
        Dim pozicijaPocetkaNaziva As Integer = 0
        Dim pozicijaKrajaNaziva As Integer = 0

        vModel = vCelaOntologija
        pozicijaPocetkaEntiteta = InStr(vModel, "<ObjectPropertyRange>")
        vModel = Mid(vModel, pozicijaPocetkaEntiteta)

        While Not vModel = ""
            pronadjencard = False
            pozicijaPocetkaEntiteta = InStr(vModel, "<ObjectProperty IRI=")
            vEntity = Mid(vModel, pozicijaPocetkaEntiteta)
            pozicijaKrajaEntiteta = InStr(vModel, "</ObjectPropertyRange>")
            vEntity = Mid(vModel, pozicijaPocetkaEntiteta, pozicijaKrajaEntiteta -
            pozicijaPocetkaEntiteta + 25)
            pozicijaPocetkaNaziva = InStr(vEntity, "#")
            pozicijaKrajaNaziva = InStr(vEntity, "/>")
            vEntityName1 = Mid(vEntity, pozicijaPocetkaNaziva + 1, pozicijaKrajaNaziva
            - pozicijaPocetkaNaziva - 2)

            pozicijaPocetkaEntiteta = InStr(vEntity, "<Class IRI=")
            vEntity2 = Mid(vEntity, pozicijaPocetkaEntiteta, pozicijaKrajaEntiteta -
            pozicijaPocetkaEntiteta + 14)
            pozicijaPocetkaNaziva = InStr(vEntity2, "#")
            pozicijaKrajaNaziva = InStr(vEntity2, "/>")
            vEntityName2 = Mid(vEntity2, pozicijaPocetkaNaziva + 1,
            pozicijaKrajaNaziva - pozicijaPocetkaNaziva - 2)
```

```
            pozicijaPocetkaEntiteta = InStr(vEntity, "<ObjectMinCardinality
            cardinality=")
            If pozicijaPocetkaEntiteta = 0 Then
                pozicijaPocetkaEntiteta = InStr(vEntity, "<ObjectMaxCardinality
                cardinality=")
                If pozicijaPocetkaEntiteta = 0 Then
                    pozicijaPocetkaEntiteta = InStr(vEntity, "<ObjectAllValuesFrom")
                    vEntityName3 = "maxcardm"
                    pronadjencard = True
                End If
                If pronadjencard = False Then
                    vEntity3 = Mid(vEntity, pozicijaPocetkaEntiteta,
                    pozicijaKrajaEntiteta - pozicijaPocetkaEntiteta + 37)
                    pozicijaPocetkaNaziva = 35
                    pozicijaKrajaNaziva = 37
                    vEntityName3 = Mid(vEntity3, 36, 1)
                    vEntityName3 = "maxcard" + vEntityName3
                    pronadjencard = True
                End If
            End If
            If pronadjencard = False Then
                vEntity3 = Mid(vEntity, pozicijaPocetkaEntiteta,
                pozicijaPocetkaEntiteta + 37)
                pozicijaPocetkaNaziva = 35
                pozicijaKrajaNaziva = 37
                vEntityName3 = Mid(vEntity3, 36, 1)
                vEntityName3 = "mincard" + vEntityName3
            End If
            richtxtClauses.Text = richtxtClauses.Text + "rdf(" +
            vEntityName2.ToString.ToLower + ", " + vEntityName1.ToString.ToLower + ",
            " + vEntityName3.ToString.ToLower + ")." + Chr(13)
            fileClauses.WriteLine("rdf(" + vEntityName2.ToString.ToLower + ", " +
            vEntityName1.ToString.ToLower + ", " + vEntityName3.ToString.ToLower +
            ").")
            vModel = Mid(vModel, pozicijaKrajaEntiteta + 20, Len(vModel) -
            pozicijaKrajaEntiteta)
        End While
    Izlaz:
    End Sub

Private Sub btnNewAnalyses_Click(ByVal sender As System.Object, ByVal e As
    System.EventArgs) Handles btnNewAnalyses.Click
        txtModel.Text = ""
        richtxtOntology.Text = ""
        richtxtRules.Text = ""
        richtxtClauses.Text = ""
        vModel = ""
        vCeoModel = ""
        vOntologija = ""
        vCelaOntologija = ""
        vRules = ""
    End Sub

End Class
```