



УНИВЕРЗИТЕТ У НОВОМ САДУ
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА У
НОВОМ САДУ



Стефан Пијетловић

ЈЕДНО РЕШЕЊЕ ОНТОЛОГИЈЕ ЗА ОПИС ПОДАТАКА У СИСТЕМУ ЗА ДОБАВЉАЊЕ РАЗНОРОДНОГ САДРЖАЈА СА ИНТЕРНЕТА

ДИПЛОМСКИ РАД
- Основне академске студије -



УНИВЕРЗИТЕТ У НОВОМ САДУ • ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА
21000 НОВИ САД, Трг Доситеја Обрадовића 6

КЉУЧНА ДОКУМЕНТАЦИЈСКА ИНФОРМАЦИЈА

Редни број, РБР :			
Идентификациони број, ИБР :			
Тип документације, ТД :	Монографска документација		
Тип записа, ТЗ :	Текстуални штампани материјал		
Врста рада, ВР :	Завршни (Bachelor) рад		
Аутор, АУ :	Стефан Пијетловић		
Ментор, МН :	Др Илија Башичевић		
Наслов рада, НР :	Једно решење онтологије за опис података у систему за добављање разнородног сарджаја са Интернета		
Језик публикације, ЈП :	Српски / латиница		
Језик извода, ЈИ :	Српски		
Земља публикавања, ЗП :	Република Србија		
Уже географско подручје, УГП :	Војводина		
Година, ГО :	2014.		
Издавач, ИЗ :	Ауторски репринт		
Место и адреса, МА :	Нови Сад; трг Доситеја Обрадовића 6		
Физички опис рада, ФО : (поглавља/страна/ цитата/табела/слика/графика/прилога)			
Научна област, НО :	Електротехника и рачунарство		
Научна дисциплина, НД :	Рачунарска техника		
Предметна одредница/Кључне речи, ПО :	Дигитална телевизија, Интернет, онтологија		
УДК			
Чува се, ЧУ :	У библиотеци Факултета техничких наука, Нови Сад		
Важна напомена, ВН :			
Извод, ИЗ :	У раду је описана једна примена онтологије као модел података у систему за добављање разнородног садржаја са Интернета. Такође, у раду је описана и имплементација онтологије у програмском језику Јава као и проблеми и компромиси који су направљени како би решење било применљиво у пракси.		
Датум прихватања теме, ДП :			
Датум одбране, ДО :			
Чланови комисије, КО :	Председник:	Др Јелена Ковачевић, доцент	
	Члан:	Др Иштван Пап, доцент	Потпис ментора
	Члан, ментор:	Др Илија Башичевић, доцент	



UNIVERSITY OF NOVI SAD • FACULTY OF TECHNICAL SCIENCES
21000 NOVI SAD, Trg Dositeja Obradovića 6

KEY WORDS DOCUMENTATION

Accession number, ANO :			
Identification number, INO :			
Document type, DT :	Monographic publication		
Type of record, TR :	Textual printed material		
Contents code, CC :	Bachelor Thesis		
Author, AU :	Stefan Pijetlović		
Mentor, MN :	Ilija Bašičević, PhD		
Title, TI :	A solution of an ontology for the description of data in a system which collects various content from the Internet		
Language of text, LT :	Serbian		
Language of abstract, LA :	Serbian		
Country of publication, CP :	Republic of Serbia		
Locality of publication, LP :	Vojvodina		
Publication year, PY :	2014.		
Publisher, PB :	Author's reprint		
Publication place, PP :	Novi Sad, Dositeja Obradovica sq. 6		
Physical description, PD : (chapters/pages/ref./tables/pictures/graphs/appendixes)			
Scientific field, SF :	Electrical Engineering		
Scientific discipline, SD :	Computer Engineering, Engineering of Computer Based Systems		
Subject/Key words, S/KW :	Digital Television, Internet, ontology		
UC			
Holding data, HD :	The Library of Faculty of Technical Sciences, Novi Sad, Serbia		
Note, N :			
Abstract, AB :	This paper describes a way of using an ontology as a data model in a system which acquires various content from the Internet. Also, a Java implementation of the ontology is described along with the problems and compromises which have been made in order to make the solution usable.		
Accepted by the Scientific Board on, ASB :			
Defended on, DE :			
Defended Board, DB :	President:	Jelena Kovačević, PhD	
	Member:	Ištván Pap, PhD	Menthor's sign
	Member, Mentor:	Ilija Bašičević, PhD	

Zahvalnost

Zahvaljujem se mentoru dr Iliji Bašičeviću i stručnom saradniku Nenadu Jovanoviću na savetima i ukazanoj pomoći pri realizaciji ovog rada.

Posebno se zahvaljujem porodici, prijateljima i kolegama koji su bili tu da pruže podršku tokom čitavog školovanja.

SADRŽAJ

1. Uvod	1
2. Teorijske osnove.....	3
2.1 Osnovni pojmovi.....	4
2.2 Primer jednostavne ontologije	5
3. Koncept rešenja	7
3.1 Pregled klasa ontologije.....	8
3.1.1 Klasa Content	8
3.1.1.1 Klase TimeRestrictedContent, ContentOnDemand i LiveContent	10
3.1.1.2 Klase TimeUnrestrictedContent, ServiceRelatedContent i WebPage ...	11
3.1.2 Klasa Metadata	11
3.1.2.1 Klase Person, Organization i Location.....	12
3.1.2.2 Klase koje predstavljaju zanimanja.....	13
3.1.3 Klasa Tag.....	15
4. Programsko rešenje.....	16
4.1 Modul CMoreCloudEngineComponents	17
4.1.1 Klasa ContentItem	17
4.1.2 Klasa ContentMetadata	17
4.1.3 Klasa ContentTimeRestriction	17
4.1.4 Klasa ContentFilter.....	18
4.2 Modul CMoreKnowlegdeBase	18
4.2.1 Sprega CMoreKnowledgeBaseApi	18
4.2.2 Klasa KnowledgeBaseQuery.....	19
4.2.3 Klase Entity i Result.....	19
4.2.4 Klasa Person	20
4.2.5 Klasa Actor.....	20
4.2.6 Klasa FreebaseActor.....	20
5. Rezultati.....	22
6. Zaključak	24
7. Literatura	25

SPISAK SLIKA

Slika 1. Primer jednostavne ontologije fakulteta	5
Slika 2. Hijerarhija sadržaja.....	9
Slika 3. Klasa Content i njene veze sa ostalim klasama	10
Slika 4. Klasa Content i njeni potomci	11
Slika 5. Klasa Metadata, njeni potomci i veza sa ostalim klasama	12
Slika 6. Klasa Person, njene veze sa ostalim klasama i potomci.....	13
Slika 7. Instanca filma i sve veze koje se formiraju u ontologiji.....	14
Slika 8. Primer komunikacije između našeg servera i Freebase-a.....	19

SPISAK TABELA

Tabela 1. Odnos broja klasa u teorijskom i konkretnom rešenju.....	23
--	----

SKRAĆENICE

DVB – **D**igital **V**ideo **B**roadcasting: skup standarda za digitalnu televiziju

RDF – **R**esource **D**escription **F**ramework: familija specifikacija za opis ontologija i metapodataka

W3C – **W**orld **W**ide **W**eb **C**onsortim: glavna internacionalna organizacija zadužena za standarde o svetskoj mreži

AI – **A**rtificial **I**ntelligence: veštačka inteligencija

HbbTV - **H**ybrid **B**roadcast **B**roadband **T**V: industrijski standard (ETSI TS 102 796) i inicijativa da se omogući realizacija hibridnih interaktivnih digitalnih TV aplikacija koje koriste i prenosni tok (Broadcast) i vezu sa Internetom (Broadband)

POJO – **P**lain **O**ld **J**ava **O**bject: objekat klase koja je obična i ne prati neke od Java modela i šablona

1. Uvod

Televizija kao primarni izvor informacija od značaja i zabave postepeno gubi trku sa Internetom koji se razvija munjevitom brzinom i gde količina dostupnih informacija postaje nezamisljivo velika. Korisnici danas žele kvalitetan sadržaj koji odgovara isključivo njihovom ukusu, a takav sadržaj je mnogo lakše pronaći na Internetu nego na televiziji.

U poslednjih nekoliko godina, usled potrebe tržišta i većih mogućnosti standardnih TV prijemnika, postoji trend povezivanja televizije i Interneta na jednom uređaju. Tehnologija to dozvoljava ali postoji određeni broj problema: naime, format podataka koji se emituje na televiziji je unapred definisan i postoji veliki broj standarda koji regulišu ovu oblast (DVB, ATSC, ISDB, DTMB...). Sa druge strane, sadržaj koji dolazi sa Interneta može biti veoma raznolik po svojoj prirodi: video, tekst, web stranice itd. i većinom je prilagođen za gledanje na računaru, tako da nije svaki jednako pogodan za prikaz na TV prijemniku. Samim tim dolazimo do problema: kako modelovati podatke koji dolaze sa različitih izvora u različitim formatima a pri tom omogućiti jedinstveno rukovanje sadržajem bez obzira na njegovo poreklo? U rešenju problema se koristi ontologija. Raznorodni podaci koji se dobavljaju sa Interneta se preslikavaju na ontologiju i potom se dalje obrađuju u sistemu i dostavljaju krajnjem korisniku.

U radu je prikazano formalno rešenje problema u vidu ontologije, modelovane uz pomoć alata Protégé, verzija 4.3. Prikazana je realizacija ontologije u vidu Java klasa koje sačinjavaju poseban modul u arhitekturi poslužioca koji obavlja akviziciju sadržaja sa Interneta i njegovu obradu [1]. Objašnjeni su razlozi zbog kojih je došlo do odstupanja u odnosu na teorijsko rešenje kao i veza sa bazom podataka.

Rad je podeljen u sedam celina:

- 1.) Uvod – izložen je početni problem.
- 2.) Teorijske osnove – dat je opis ontologije kao pojma i razlozi za korišćenje iste prilikom rešavanja početnog problema
- 3.) Koncept rešenja – prikazano je jedno teorijsko i formalno rešenje problema, i izvršena klasifikacija podataka od značaja.
- 4.) Programsko rešenje – opisana je implementacija u programskom jeziku Java i dat pregled najvažnijih klasa.
- 5.) Rezultati – prikaz rezultata
- 6.) Zaključak – dat je kratak pregled rešenja i mogući pravci daljeg razvoja
- 7.) Literatura – sadrži spisak korišćene literature

2. Teorijske osnove

Reč ontologija je grčkog porekla i predstavlja filozofsku disciplinu koja se bavi bićem i njegovim opštim, fundamentalnim i konstitutivnim određenjima. U računarstvu i informatici, ontologija formalno predstavlja znanje unutar nekog domena koje je predstavljeno kao hijerarhija klasa, tipova, osobina i njihovih međusobnih veza. Kako je domen Interneta i digitalne televizije poprilično veliki, potrebno je na neki način klasifikovati i kategorizovati znanje i odvojiti bitno od nebitnog, kada su upitanju sadržaji koji se isporučuju krajnjem korisniku, bilo da se on nalazi ispred računara ili televizora.

Postavlja se pitanje: zašto razvijati ontologiju? Neki od odgovora, pored klasifikacije i kategorizacije znanja, datih u [2] uključuju: analizu znanja unutar domena, ponovno korišćenje znanja, razdvajanje domenskog od operativnog znanja i mnogi drugi. Definisanje ontologije nekog sistema u najčešćem slučaju nije cilj sam po sebi već jedan od koraka u dizajnu kvalitetnije programske podrške. Razvoj ontologije je srodan razvoju skupa podataka i njihovih struktura koje će se koristiti razne komponente unutar sistema.

U računarstvu se ontologija koristi kao okosnica za razvoj i struktuiranje podataka u oblastima poput veštačke inteligencije, semantičkog Web-a, biomedicine, rukovanja bibliotekama i mnogim drugim [3]. Još sredinom sedamdesetih godina dvadesetog veka, istraživači na polju veštačke inteligencije (Artificial Intelligence, AI) su otkrili da je modelovanje znanja ključno za velike i moćne AI sisteme. Jednu od prvih definicija ontologije kao discipline u oblasti računarstva dao je Tom Gruber u svom radu [4] u prvoj polovini devedesetih godina. Gruber je ontologiju definisao kao specifikaciju konceptualizacije, odnosno on vidi ontologiju kao opis pojmova i veza među njima. Pored veštačke inteligencije, ontologije su našle najveću primenu u ideji semantičkog Web-a. Ovo predstavlja pokret koji predvodi W3C (World Wide Web Consortium) čiji je cilj da trenutnu

svetsku mrežu pretvori u strukturiranu mrežu podataka koje će mašine biti same u stanju da obrade i razumeju. Henry Kim u svom radu [5] predviđa da je semantički Web budućnost i da će razvoj tog koncepta zavisiti u najvećoj meri od razvoja mnoštva ontologija koje su sastavni deo projekta. U poslednje vreme, ontologije se često koriste i u oblasti multimedije [6], [7] u sklopu arhitektura sistema i ispitivanja kvaliteta usluge.

2.1 Osnovni pojmovi

Osnovni pojmovi koji čine jednu ontologiju su: klase, osobine klase (engl. data property), veze između klasa (engl. object property) i ograničenja. Ontologija zajedno sa individuumima (instancama klasa) čini bazu znanja nekog domena.

Klase predstavljaju glavni deo svake ontologije zato što svaka klasa predstavlja jedan pojam unutar domena. U našem sistemu, koji će kasnije u radu biti detaljnije opisan, klasa *Content* predstavlja svaki vid digitalnog sadržaja koji se dostavlja krajnjem korisniku. Emisija koja se upravo prikazuje predstavlja jednu instancu klase *Content*. Svaki video klip i svaka web stranica takođe predstavljaju instance ove klase. Unutar ontologije se klase organizuju hijerarhijski, kao stablo, gde svaka klasa može imati svoje pod klase. Važno je napomenuti da u ovoj hijerarhiji važe principi nasleđivanja ustanovljeni u objektno-orijentisanoj paradigmi, odnosno svaka pod klasa sadrži sve osobine nad klase sa dodatkom specifičnih proširenja. Jedna suštinska razilka je da u ontologiji određena individua može istovremeno biti instanca proizvoljnog broja klasa, ukoliko ograničenja to dozvoljavaju.

Međutim sama hijerarhija klasa nam ne daje dovoljno informacija o sistemu. Ovo je naročito tačno kada se radi o velikim sistemima koji imaju puno veza između koncepata. Zato su osobine klasa, veze i ograničenja neizostavni deo svake ontologije.

Osobine klasa (data property dalje u tekstu) predstavljaju polja klase koja su najčešće neki od osnovnih tipova: celobrojne vrednosti, realni brojevi, logičke vrednosti, znakovni nizovi ili neki složeni tipovi poput datuma.

Veze između klasa (object property dalje u tekstu) predstavljaju polja klase koja su instance drugih klasa ili reference na njih. One su te koje razbacane informacije unutar domena grupišu u skladnu celinu. Ograničenja se primenjuju na klase i tu su da uvedu logičke zabrane koje važe u sistemu.

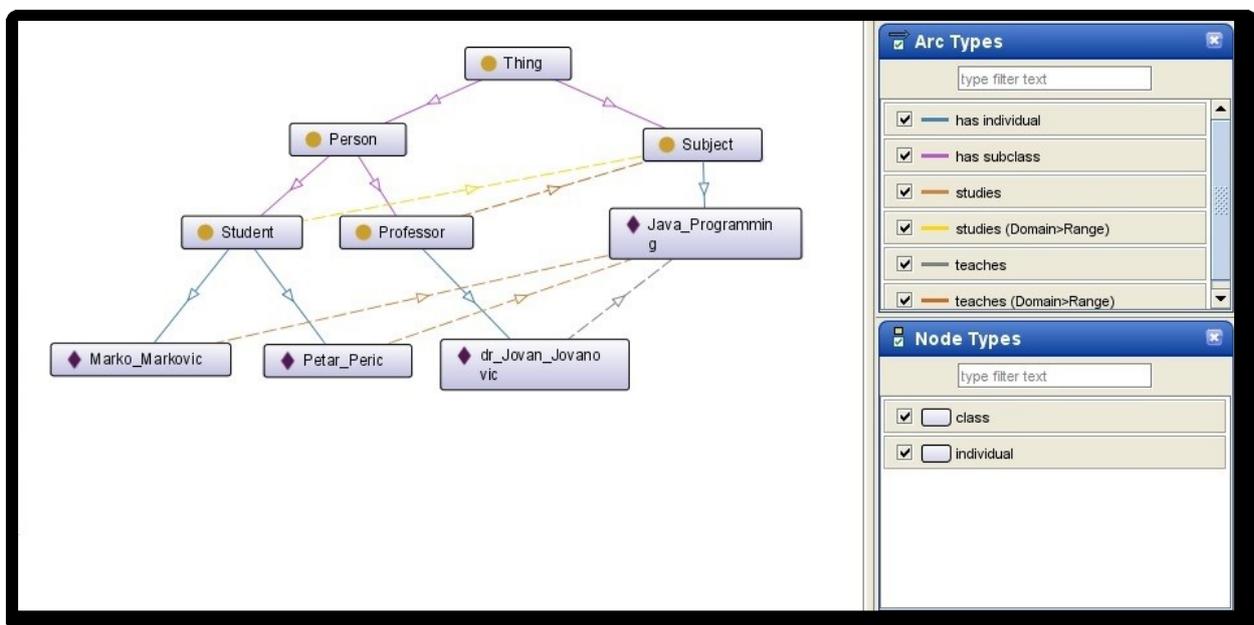
Ontologije se najčešće opisuju uz pomoć RDF-a (Resource Description Framework). RDF je familija W3C specifikacija primarno dizajniranih kao model podataka za metapodatke. Koncept je baziran na iskazima o resursima u formi subjekat-predikat-objekat. Subjekat predstavlja resurs, predikat vezu između subjekta i objekta, dok je objekat u

najčešćem slučaju takođe resurs. Kolekcija RDF iskaza suštinski predstavlja graf podataka koji je pogodna struktura za predstavljanje znanja.

2.2 Primer jednostavne ontologije

Sve osnovne pojmove ćemo razjasniti na veoma pojednostavljenoj ontologiji fakulteta. Pojmovi koji su nam od interesa su: profesor, student i predmet, te će se te tri klase naći u hijerarhiji. Možemo primetiti da profesor i student imaju nekoliko zajedničkih osobina kao što su ime, prezime i broj godina. Pored toga imaju i veliki broj zajedničkih osobina koje nam nisu od interesa kao što su na primer boja očiju, kose, visina, težina, omiljena knjiga itd... Iz tog razloga ima smisla da klasa *Professor* i klasa *Student* imaju zajedničkog pretka, klasu *Person* koja bi u sebi sadržala data property-e *firstName*, *lastName* i *age*. Obe klase nasleđuju klasu *Person*, gde svaka klasa u sebi sadrži neke posebne odredbe. Tako na primer klasa *Professor* sadrži i polje *staffID* dok klasa *Student* ima polje *studentID*.

Logička veza između profesora i studenta jeste predmet koji profesor predaje a student sluša. Ova veza se opisuje uz pomoć object property-a. Unutar klase *Professor* imamo object property *teaches* koji je povezan sa klasom *Subject*. Analogno, u klasi *Student* imamo object property *studies* koji je takođe povezan sa klasom *Subject*.



Slika 1. Primer jednostavne ontologije fakulteta

Kada su u pitanju ograničenja u ontologiji, ona se primenjuju na dva mesta u hijerarhiji. U našem pojednostavljenom slučaju, logično je da neki pojam ne može biti istovremeno i osoba i predmet izučavanja. Zato se prvo ograničenje stavlja na najvišem hijerarhijskom

nivou i označava da su klase *Person* i *Subject* disjunktne. Isto tako, pretpostavljamo da ni jedan student nije istovremeno i profesor na fakultetu i obrnuto, tako da se drugo ograničenje uvodi na tom hijerarhijskom nivou. Ograničenja su generalno veoma korisna u slučaju kada imamo izuzetno kompleksne ontologije sa mnogo veza, jer nas samo okruženje sprečava da pravimo greške prilikom dodavanja individua u sistem.

Nakon definisanih ograničenja, dodavanjem individua izlazimo izvan granica ontologije i dobijamo bazu znanja. U našoj vrlo jednostavnoj bazi znanja, imamo jednog profesora, dva studenta i jedan predmet. Kada se njihova polja popune, okruženje je u stanju da celokupnu bazu znanja eksportuje u RDF formatu koji se standardno koristi za opis ontologija.

Primer izgenerisanog RDF-a za jednog studenta:

```
<owl:NamedIndividual rdf:about="&untitled-ontology-19;Petar_Peric">
  <rdf:type rdf:resource="&untitled-ontology-19;Person"/>
  <rdf:type rdf:resource="&untitled-ontology-19;Student"/>
  <untitled-ontology-19:studentID rdf:datatype="&xsd;integer">12345</untitled-ontology-19:studentID>
  <untitled-ontology-19:age rdf:datatype="&xsd;integer">20</untitled-ontology-19:age>
  <untitled-ontology-19:lastName rdf:datatype="&xsd;string">Peric</untitled-ontology-19:lastName>
  <untitled-ontology-19:firstName rdf:datatype="&xsd;string">Petar</untitled-ontology-19:firstName>
  <untitled-ontology-19:studies rdf:resource="&untitled-ontology-19;Java_Programming"/>
</owl:NamedIndividual>
```

Primer RDF-a za predmetnog profesora:

```
<owl:NamedIndividual rdf:about="&untitled-ontology-19;dr_Jovan_Jovanovic">
  <rdf:type rdf:resource="&untitled-ontology-19;Person"/>
  <rdf:type rdf:resource="&untitled-ontology-19;Professor"/>
  <untitled-ontology-19:age rdf:datatype="&xsd;integer">42</untitled-ontology-19:age>
  <untitled-ontology-19:staffID rdf:datatype="&xsd;integer">987654</untitled-ontology-19:staffID>
  <untitled-ontology-19:firstName rdf:datatype="&xsd;string">Jovan</untitled-ontology-19:firstName>
  <untitled-ontology-19:lastName rdf:datatype="&xsd;string">Jovanovic</untitled-ontology-19:lastName>
  <untitled-ontology-19:teaches rdf:resource="&untitled-ontology-19;Java_Programming"/>
</owl:NamedIndividual>
```

3. Koncept rešenja

Kao što je napomenuto ranije u radu, domen digitalne televizije i Interneta je izuzetno veliki. Da bi ontologija uopšte imala smisla, potrebno je izdvojiti podskup podataka koji su od značaja za rad našeg sistema i samo njega opisati. Naš sistem treba da bude u stanju da na zahtev klijenata dostavi metapodatke uz sadržaj koji se trenutno emituje od strane televizijskih operatera, kao i da preporuči raznorodni sadržaj sa Interneta povezan sa trenutnom tematikom. Iako su DVB transportni tok i Internet disjunktni izvori, potreban je mehanizam koji omogućuje rukovanje podacima sa oba izvora. Glavnu prepreku predstavljaju veoma raznoliki formati podataka – u DVB standardu [8] je format veoma precizno definisan za razliku od Interneta gde gotovo da ne postoji pravilnost u smislu dodatnih informacija koje su pridružene nekom sadržaju.

Inicijalno rešenje sistema, pre početka razvoja ontologije, se u velikoj meri oslanjalo na informacije koje pristižu iz transportnog toka. U ove informacije spadaju: vreme početka i kraja emisije, odnosno dužina trajanja, žanr, kraći i duži opis emisije, podatak o tome za koji je uzrast prikladan sadržaj i mnoge druge. Iz tog razloga je logično da postoji hijerarhija u koju se mogu uklopiti sadržaji vezani za DVB. Međutim, u sadržaj spadaju i svi video klipovi i sve web stranice koji najčešće nemaju nikakve veze sa DVB standardom. Treba napomenuti da neka emisija može biti dostupna i preko Interneta i to sa više različitih izvora. Zato je potrebna neka korenska klasa koja predstavlja svaki vid sadržaja koji se može dostaviti krajnjim korisnicima.

Za prvobitnu klasifikaciju sadržaja je korišćen podatak o žanru koji pristiže u DVB transportnom toku. Ipak ovakav vid podele na samo nekoliko kategorija je bio previše opšti i nije bio primenljiv za veliki broj servisa iz prostog razloga što žanr nije bio definisan. Sledeći korak je bilo uvođenje takozvanih tagova koji su malo bliže objašnjavali u koju kategoriju

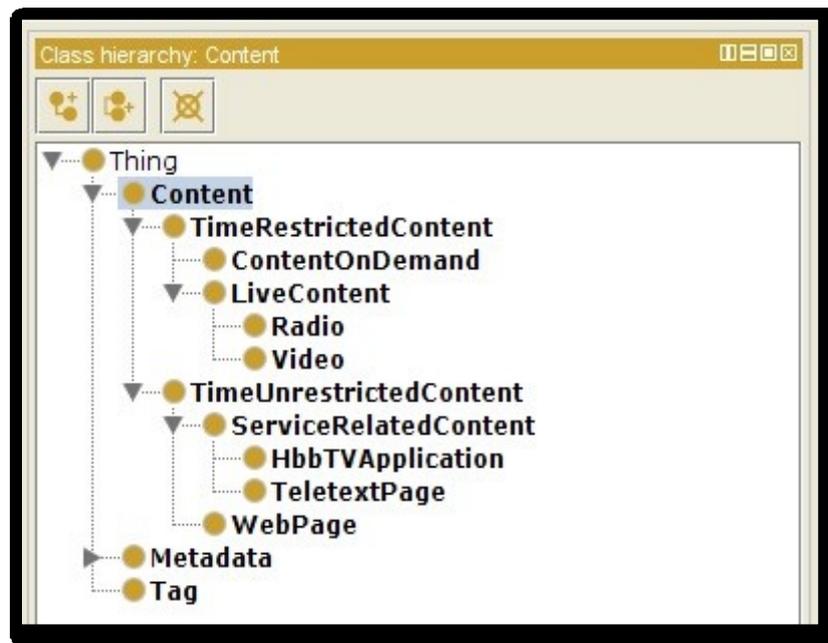
spada određena emisija. Tag predstavlja reč ili pojam koji je pridružen nekom sadržaju i nosi neki vid informacija o njemu i omogućuje lakši pronalazak svih sadržaja koji su označeni istim tagom. Inicijalni skup tagova je bio preslikan skup žanrova sa dodatkom nekoliko pojmova poput fudbala, emisija o kuvanju i slično. Rešenje se pokazalo za nijansu bolje, međutim nije bilo moguće uključiti u taj koncept ništa od sadržaja sa Interneta a neki vid kategorizacije je morao biti uveden da zameni dosadašnji. Po uzoru na web stranicu Freebase [9] koji predstavlja jednu od najvećih baza znanja, započet je razvoj hijerarhije metapodataka. Koncept tagova je zadržan samo kao pomoćni.

3.1 Pregled klasa ontologije

3.1.1 Klasa Content

Ova klasa modeluje svaki vid sadržaja koji se može predstaviti krajnjem korisniku. Kao vrste sadržaja koje dolaze u obzir izdvojeni su: sadržaj na zahtev (CoD – Content on Demand), sadržaj koji se emituje uživo (Live Content) koji može biti video ili radio, web stranice, teletekst stranice i HbbTV aplikacije.

Jedan način kategorizacije koji se nama ukazao kao odgovarajući je podela na vremenski ograničen i vremenski neograničen sadržaj, odnosno na sadržaj koji ima neko trajanje i na onaj koji ga nema (na primer emisija na televiziji ili video klip spadaju u vremenski ograničen sadržaj dok teletekst ili web stranica predstavljaju vremenski neograničen sadržaj jer je njihova dinamika menjanja znatno manja).

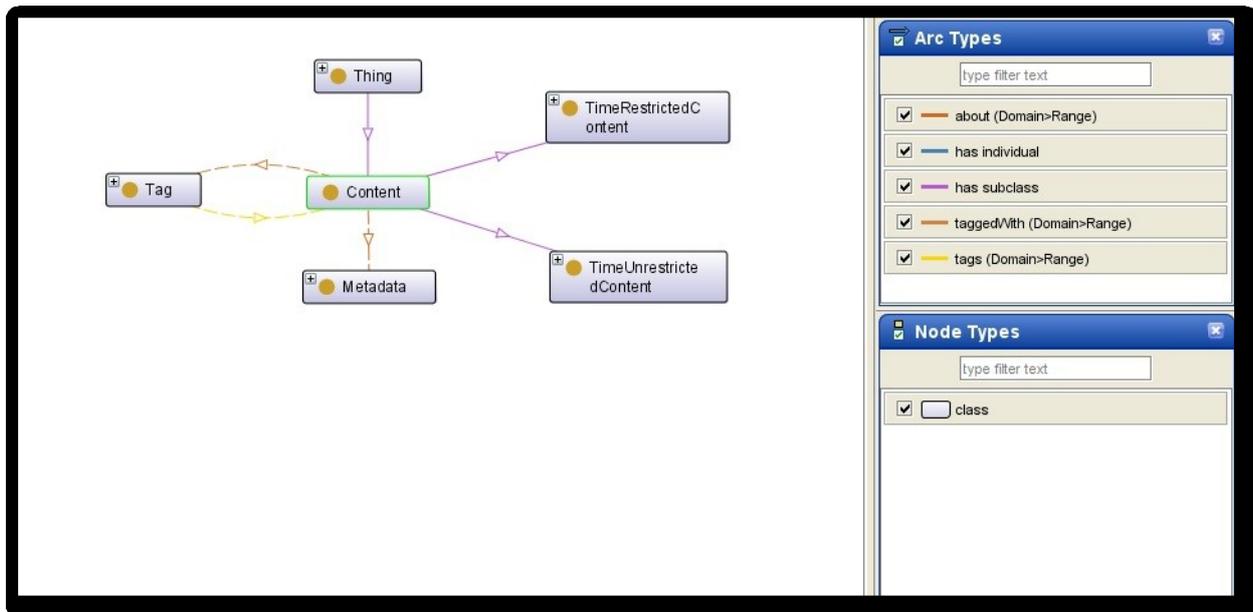


Slika 2. Hijerarhija sadržaja

Zajedničke informacije za sav sadržaj predstavljaju odgovarajuće data property-e:

- jedinstveni identifikator: *contentURI*
- opis sadržaja: *contentDescription*
- jezik: *contentLanguage*
- naziv sadržaja: *contentName*
- uzrast za koji je sadržaj prikladan: *contentParentalRate*
- cena: *contentPrice*
- ocena sadržaja: *contentRating*

Veze sa ostalim klasama su modelovane korišćenjem object property-a i najbolje se vide na sledećoj slici:



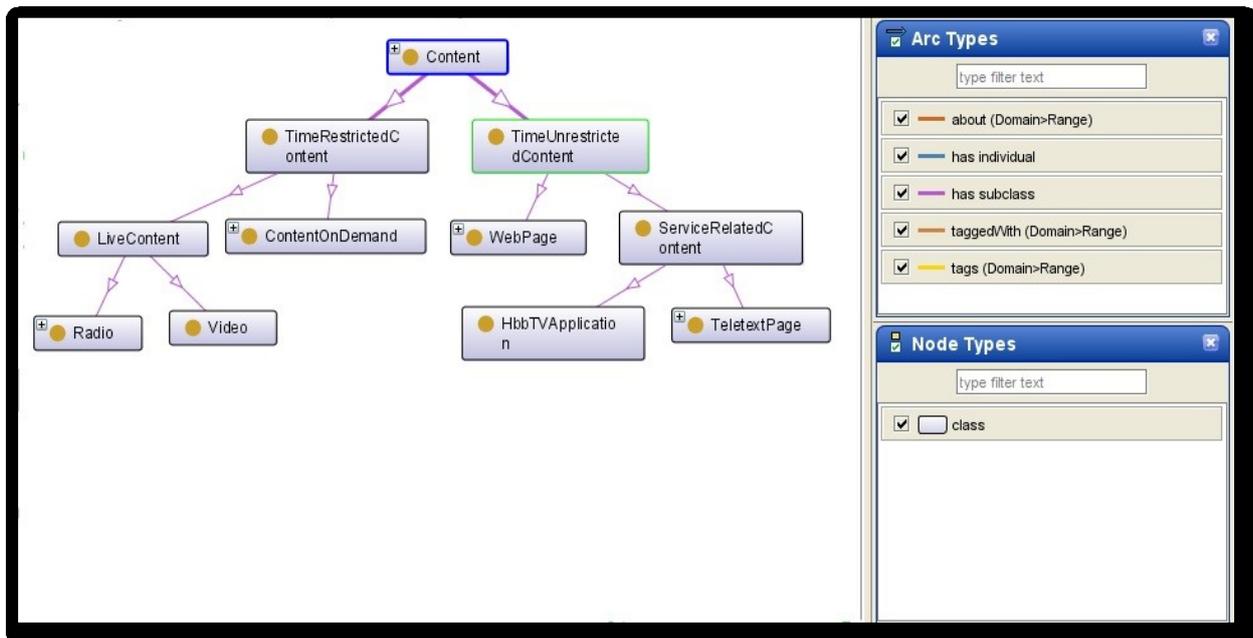
Slika 3. Klasa Content i njene veze sa ostalim klasama

Vidimo da je klasa *Content* potomak klase *Thing* koja predstavlja najopštiji pojam u okviru ontologije, analogno klasi *Object* u programskom jeziku Java. Ima dva potomka koji predstavljaju vremenski ograničen sadržaj, odnosno sadržaj koji ima neko trajanje, i vremenski neograničen sadržaj koji dalje imaju svoje potomke koji nisu prikazani na slici radi bolje preglednosti. Object property *about* nam govori da je sadržaj povezan za neku temu unutar hijerarhije *Metadata* odnosno da se u okviru instance neke klase *Content* pojavljuje neka od instance klase *Metadata*. Vidimo da postoji i dvostruka veza sa pojmom *Tag* radi dodatnog opisa sadržaja.

3.1.1.1 Klase *TimeRestrictedContent*, *ContentOnDemand* i *LiveContent*

Klasa *TimeRestrictedContent* predstavlja vremenski ograničen sadržaj odnosno sadržaj koji ima neko trajanje. Suštinska razlika između ovakve vrste sadržaja i web stranica je što je ovakav sadržaj veoma promenljiv u vremenu za razliku od stranice čija je dinamika menjanja znatno manja te se može zanemariti u našem slučaju – drugim rečima, korisnik mora biti prisutan kako ne bi propustio neki deo. Ukoliko nije, moraće da premeta sadržaj do mesta kada je prestao da ga gleda ili će morati da gleda sadržaj ponovo u nekom drugom vremenskom periodu u slučaju da ne postoji opcija premotavanja. Na slici 2. vidimo da ova klasa ima dve klase potomka – sadržaj na zahtev (*ContentOnDemand*) i sadržaj koji se emituje uživo (*LiveContent*). Zajednička osobina oba potomka je da imaju dužinu trajanja, tako da se data property *contentDuration* definiše na ovom nivou hijerarhije. U slučaju

“živog” sadržaja imamo i vreme početka i kraja emitovanja sadržaja, te se na nivou klase *LiveContent* dodaju i odgovarajući data property-i *contentStartTime* i *contentEndTime*.



Slika 4. Klasa Content i njeni potomci

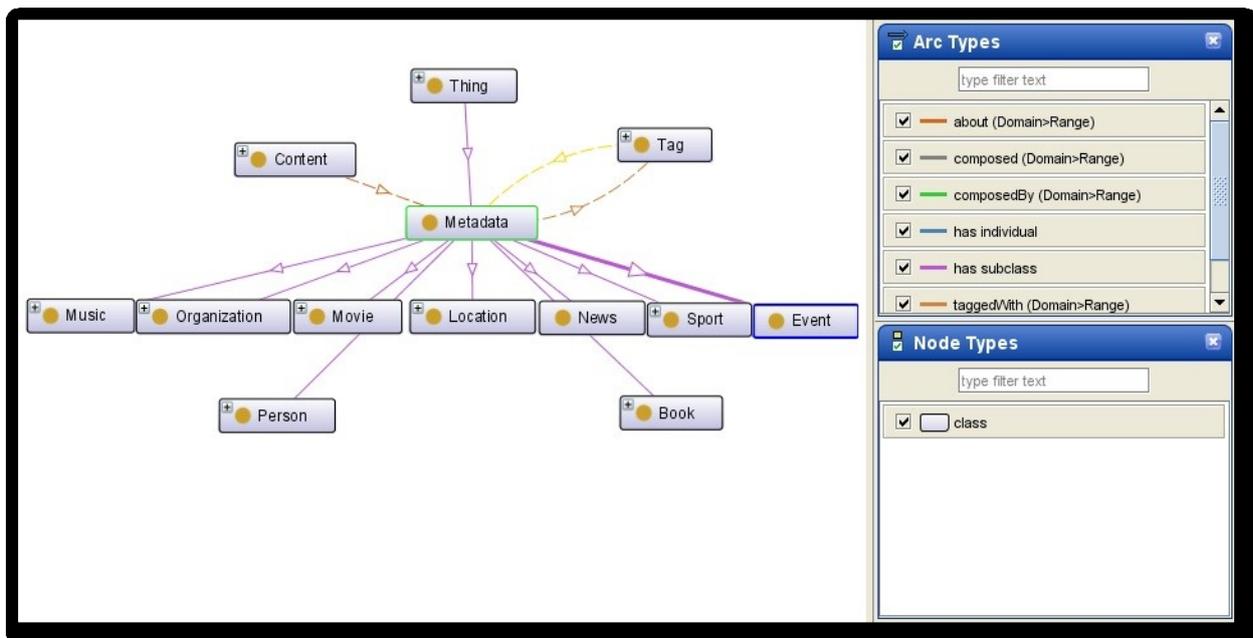
3.1.1.2 Klase *TimeUnrestrictedContent*, *ServiceRelatedContent* i *WebPage*

Klasa *TimeUnrestrictedContent* predstavlja uslovno rečeno vremenski neograničen sadržaj, odnosno sadržaj koji kada se jednom pribavi verovatno neće imati mnogo promena pa nije potrebno da korisnik sve vreme bude prisutan. Ovde možemo primetiti dve vrste sadržaja: jedan vezan za određene DVB servise i drugi koji uglavnom nije. Samim tim klasa *TimeUnrestrictedContent* ima dve klase potomaka – jedni predstavljaju web stranice (klasa *WebPage*) dok u drugu grupu spadaju teletext stranice i HbbTV aplikacije (klasa *ServiceRelatedContent*).

Od data property-a koji su ovde novi, imamo pre sve identifikatore koji su definisani DVB standardom i to: identifikator mreže (*networkID*), zatim identifikator multipleksa (*transponderID*) i identifikator servisa (*serviceID*).

3.1.2 Klasa Metadata

Ova klasa predstavlja sam vrh hijerarhije metapodataka. Metapodaci predstavljaju podatke o podacima i kao takvi se veoma često koriste u web pretraživačima. U našem slučaju metapodaci služe da bliže opišu sadržaj koji se isporučuje korisniku i za kategorizaciju sadržaja kako bi podsistem za preporuku sadržaja imao bolji skup ulaznih podataka i samim tim mogao preciznije da napravi preporuku.



Slika 5. Klasa Metadata, njeni potomci i veza sa ostalim klasama

Svi potomci klase *Metadata* predstavljaju pojmove koji su nama od interesa. Ontologija koja postoji u okviru Freebase-a je mnogo složenija i održavana od strane velikog broja korisnika ali je takođe i previše detaljna za potrebe našeg sistema. Iz tog razloga smo izdvojili mali podskup nama relevantnih pojmova koji predstavljaju znanje koje je nama od značaja.

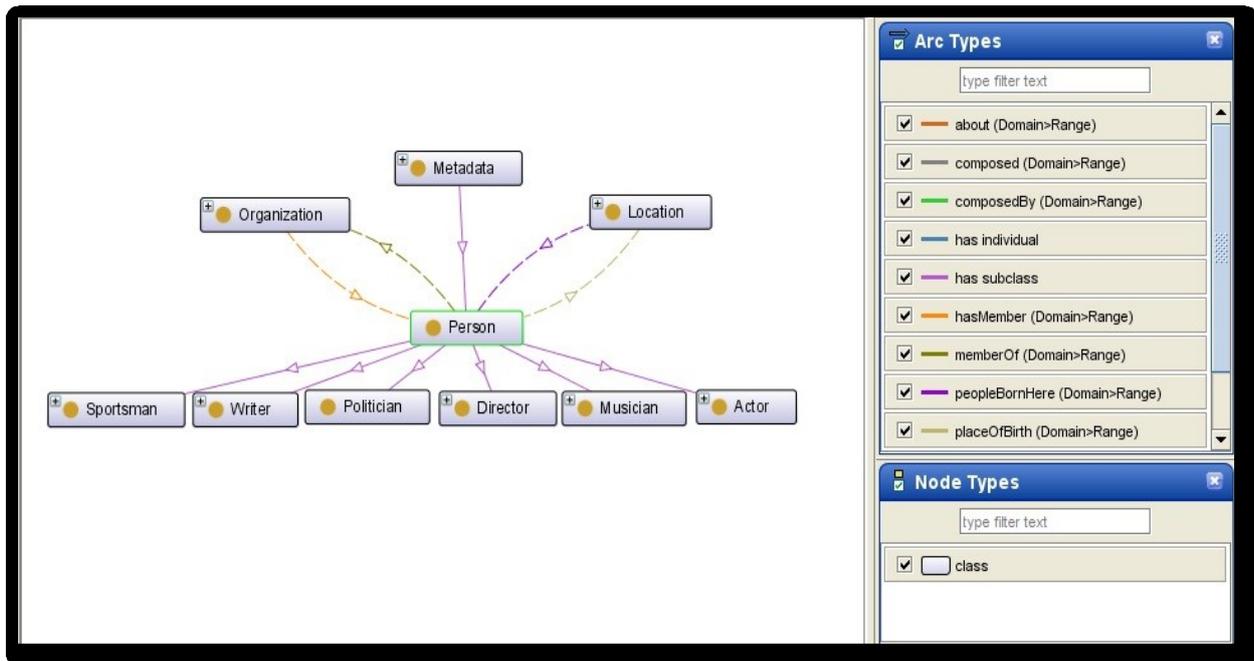
3.1.2.1 Klase Person, Organization i Location

Klasa *Person* je od velikog značaja jer su ljudi neizostavni deo skoro svih sadržaja, bilo da se radi o filmovima (glumci), sportskim događajima (igrači na terenu), vestima (poznate ličnosti su često tema vesti) i tako dalje. Ova klasa je zajednički predak svih osoba i svaka osoba predstavlja po jednu instancu ove klase. Zajedničke osobine svih osoba koje su nama od interesa su:

- ime: *firstName*
- prezime: *lastName*
- datum rođenja: *dateOfBirth*
- datum smrti (ukoliko postoji): *dateOfDeath*
- mesto rođenja: *placeOfBirth*
- mesto smrti (ukoliko postoji): *placeOfDeath*

Ostale osobine zajedničke za sve osobe (poput visine, težine, boje očiju) nama nisu od interesa ali ukoliko korisnika zanimaju, on relativno jednostavno može doći do njih time što će posetiti link koji će voditi do stranice o toj osobi i koja će biti dostavljena kao dodatni sadržaj.

Problem kada nekoga klasifikujemo kao osobu je što mi zapravo imamo relativno malo pojmova sa kojima ga možemo povezati. Mnogo veći broj veza možemo imati ukoliko znamo čime se ta osoba bavi u životu, odnosno po čemu je poznata – tada tu osobu možemo povezati sa drugim osobama koje se bave istim zanimanjem. Iz tog razloga, klasa *Person* ima nekoliko klasa potomaka koje predstavljaju zanimanja. Ovaj skup nije konačan i u trenutku definisanja ontologije se sastojao samo od šest klasa. Ukoliko se pojavi potreba, skup se vrlo jednostavno može proširiti bez promena u strukturi sistema.



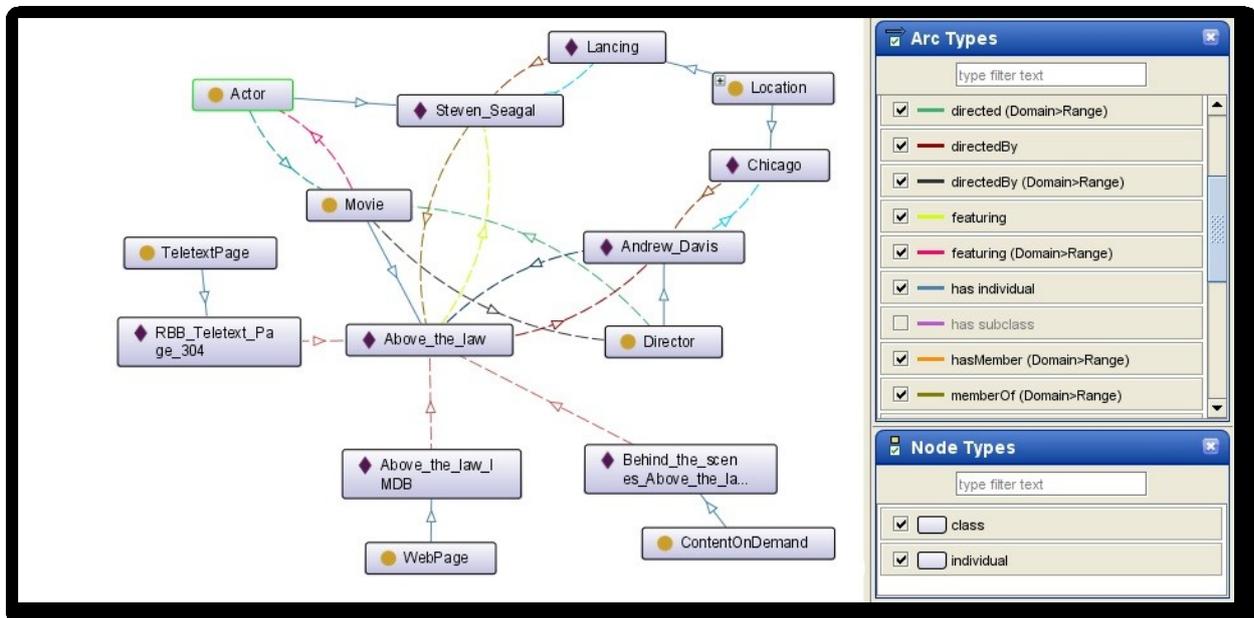
Slika 6. Klasa *Person*, njene veze sa ostalim klasama i potomci

Sa slike 6 vidimo da je klasa *Person* povezana i sa klasom *Organization* koja predstavlja grupu od više osoba. Takođe vidimo da povezana i sa klasom *Location* kako bi omogućili pretragu osoba koja su poreklom ili su na neki drugi način povezani sa istim mestom. Klasa *Location* sadrži tri data property-a kako bi jedinstveno opisala mesta na svetu i to: *country*, *region* i *city*.

3.1.2.2 Klase koje predstavljaju zanimanja

U ovu grupu spada šest klasa: *Sportsman*, *Writer*, *Politician*, *Director*, *Actor* i *Musician*. Svaka od ovih klasa u sebi sadrži neko proširenje u odnosu na nad klasu *Person* koje je bitno za dato zanimanje. Tako na primer glumac sadrži podatke o filmovima u kojima glumi, režiser podatke o filmovima koje je režirao, muzičar je povezan sa kompozicijom koju izvodi i žanrom muzike, sportista sa događajima kao što su svetsko prvenstvo ili olimpijada i slično.

Svaki pojam koji se može pridružiti nekom od zanimanja je takođe zasebna klasa. Tako imamo klase kao što su: *Book*, *Event*, *Movie*, *Music*, *News* itd. Na kraju se individue ovih klasa međusobno sprežu, praveći tako bazu znanja.



Slika 7. Instanca filma i sve veze koje se formiraju u ontologiji

Na ovom primeru vidimo sve veze koje jedan sadržaj poput filma pravi sa ostalim individuama. Ovakve informacije su veoma korisne sistemu za preporuku sadržaja jer samo spram jednog entiteta (pojma) korisniku mogu veoma brzo dostaviti veliku količinu povezanog sadržaja. Ovim mehanizmom se obezbeđuje da korisnik na jednom mestu i brzo može dobiti veliku količinu srodnog sadržaja, prilagođenu njegovom ukusu.

Primer RDF-ova koji opisuju pojedine individue sa slike:

```
<owl:NamedIndividual rdf:about="&cmore_ontology;Above_the_law">
  <rdf:type rdf:resource="&cmore_ontology;Metadata"/>
  <rdf:type rdf:resource="&cmore_ontology;Movie"/>
  <cmore_ontology:movieReleaseYear rdf:datatype="&xsd;integer">1988</cmore_ontology:movieReleaseYear>
  <cmore_ontology:movieName rdf:datatype="&xsd:string">Above the law</cmore_ontology:movieName>
  <cmore_ontology:directedBy rdf:resource="&cmore_ontology;Andrew_Davis"/>
  <cmore_ontology:featuring rdf:resource="&cmore_ontology;Steven_Seagal"/>
</owl:NamedIndividual>
```

```
<owl:NamedIndividual rdf:about="&cmore_ontology;Above_the_law_IMDB">
  <rdf:type rdf:resource="&cmore_ontology;Content"/>
  <rdf:type rdf:resource="&cmore_ontology;TimeUnrestrictedContent"/>
  <rdf:type rdf:resource="&cmore_ontology;WebPage"/>
  <cmore_ontology:contentParentalRate rdf:datatype="&xsd;integer">18</cmore_ontology:contentParentalRate>
  <cmore_ontology:contentRating rdf:datatype="&xsd;float">8.4</cmore_ontology:contentRating>
```

```

    <cmore_ontology:contentName rdf:datatype="&xsd:string">Above the law</cmore_ontology:contentName>
    <cmore_ontology:contentDescription          rdf:datatype="&xsd:string">Super          awesome          and
realistic</cmore_ontology:contentDescription>
    <cmore_ontology:contentURI
rdf:datatype="&xsd:string">http://www.imdb.com/above_the_law</cmore_ontology:contentURI>
    <cmore_ontology:about rdf:resource="&cmore_ontology;Above_the_law"/>
</owl:NamedIndividual>

```

```

<owl:NamedIndividual rdf:about="&cmore_ontology;Steven_Seagal">
  <rdf:type rdf:resource="&cmore_ontology;Actor"/>
  <rdf:type rdf:resource="&cmore_ontology;Metadata"/>
  <rdf:type rdf:resource="&cmore_ontology;Person"/>
  <cmore_ontology:dateOfBirth rdf:datatype="&xsd:string">10.04.1952.</cmore_ontology:dateOfBirth>
  <cmore_ontology:lastName rdf:datatype="&xsd:string">Seagal</cmore_ontology:lastName>
  <cmore_ontology:firstName rdf:datatype="&xsd:string">Steven</cmore_ontology:firstName>
  <cmore_ontology:starsIn rdf:resource="&cmore_ontology;Above_the_law"/>
  <cmore_ontology:placeOfBirth rdf:resource="&cmore_ontology;Lancing"/>
</owl:NamedIndividual>

```

3.1.3 Klasa Tag

Ova klasa nema nikakve potomke već samo individue. Svaka individua predstavlja jednu reč odnosno jedan pojam koji na neki način kategorizuje sadržaj. Klasa *Tag* je povezana dvosmernom vezom i sa klasama *Metadata* i sa klasom *Content* i omogućuje jednostavniju pretragu i jedne i druge hijerarhije.

4. Programsko rešenje

Nakon što se ontologija definiše, Protégé je u stanju da sam generiše Java klase. Ipak ovako mašinski generisane klase nisu efikasne sa stanovišta rukovanja resursima. Generičnost je izuzetno velika te je za pristup konkretnom objektu potrebno najpre nekoliko pristupa generičkim klasama koje nisu predviđene ontologijom. Pored toga, na dosta mesta u kodu je potrebno generičke entitete iz Protégé-a zameniti konkretnim koji već postoje u sistemu od ranije. Tip baze podataka koji odgovara ontologijama jesu graf bazirane baze dok naš sistem iz istorijskih razloga i performansi ima relacionu bazu. Iz datih razloga su klase napisane iz početka po uzoru na formalni model ali je došlo do određenih odstupanja i izmena zarad boljih performansi u budućnosti.

Ontologija je rasplinuta po celom sistemu, ali je glavni deo koncentrisan u dva dela. Prvi deo se zove *CMoreCloudEngineComponents* i predstavlja biblioteku koja u sebi sadrži definicije entiteta i veliki broj sprega u sistemu. U okviru ove biblioteke se nalazi klasa *ContentItem*, koja je pandan klasi *Content* iz formalnog modela, i sve prateće klase koje su deo te hijerarhije. Drugi deo predstavlja modul koji se zove *CMoreKnowlegdeBase*, i u okviru njega su realizovane klase koje su deo hijerarhije o metapodacima.

Važno je napomenuti da sve klase, ukoliko to nije drugačije napomenuto, predstavljaju POJO (Plain Old Java Object) klase odnosno sadrže samo polja, odgovarajuće get i set metode za njihovo dobavljanje i menjanje i konstruktor.

4.1 Modul CMoreCloudEngineComponents

U okviru ovog modula se pored klase vezane za sadržaj nalaze i paketi u kojima su realizovane klase neophodne za rukovanje akcijama korisnika, za rukovanje profilima, sistemima za preporuku i mnogi drugi.

Sve klase od interesa za ovaj rad se nalaze u okviru paketa *com.rtrk.cmore.contentcollector*.

4.1.1 Klasa ContentItem

Pandan klasi *Content* iz formalnog opisa sistema, ova klasa predstavlja svaki vid sadržaja koji se potencijalno može dostaviti krajnjim korisnicima. Najveći deo data property-a je direktno preslikan na odgovarajuća polja klase, a neka nova polja su dodata sa pojavom novih zahteva. Tako da u okviru klase imamo polja kao što su: *URI, name, description, language, parentalRate, lifespan, rating*.

Object property, *about*, je realizovan kao set klasa *ContentMetadata* radi efikasnijeg sprezanja sa bazom podataka.

Unutar klase postoji i enumeracija tipova sadržaja čije su vrednosti šest vrsta sadržaja predviđena ontologijom: *CONTENT_ON_DEMAND, LIVE_VIDEO, LIVE_RADIO, HBBTV_APPLICATION, TELETEXT_PAGE* i *WEBPAGE*. Ova enumeracija se koristi kako bi se različit sadržaj prosledio odgovarajućim komponentama u sistemu koje su namenjene za rukovanje sa njim.

4.1.2 Klasa ContentMetadata

Ova klasa sadrži minimalan skup podataka koji će se čuvati u našoj bazi podataka jednom kada neki sadržaj u potpunosti identifikujemo i povežemo sa odgovarajućom bazom znanja negde na Internetu.

Sadrži polja *ID, entity, dataSourceKey, dataSource* kao i skup *contentItem*-a. *Entity* predstavlja ime entiteta kod nas u bazi, dok *dataSourceKey* predstavlja ključ u odgovarajućoj bazi na Internetu.

4.1.3 Klasa ContentTimeRestriction

U ovoj klasi vidimo prvo odstupanje od formalne ontologije definisane u Protégé-u. Radi efikasnijeg rukovanja bazom, umesto podele na vremenski ograničen i vremenski neograničen sadržaj, imamo samo sadržaj na koji se, ukoliko ima potrebe, dodaje vremensko ograničenje. U ovom slučaju se u istoj tabeli nalazi sav sadržaj što znatno pojednostavljuje celokupnu bazu podataka.

Polja koja su od interesa su *duration*, *startTime* i *endTime*, koja predstavljaju, respektivno, trajanje sadržaja, početak i kraj emitovanja. U slučaju sadržaja na zahtev, polja *startTime* i *endTime* neće imati nikakvu vrednost, dok u slučaju web ili teletekst stranica će sva tri polja biti prazna.

4.1.4 Klasa **ContentFilter**

Ova klasa predstavlja filter koji se primenjuje prilikom dobavljanja sadržaja iz baze. Takođe nije predviđena ontologijom ali je koncept nasleđen iz prvobitne implementacije sistema. Parametri po kojima se može filtrirati su ujedino i polja ove klase i to su: *uri*, *name*, *language*, *entity*, *sourceName*, *duration*, *rating*, *parentalRate*, *containsEntities*.

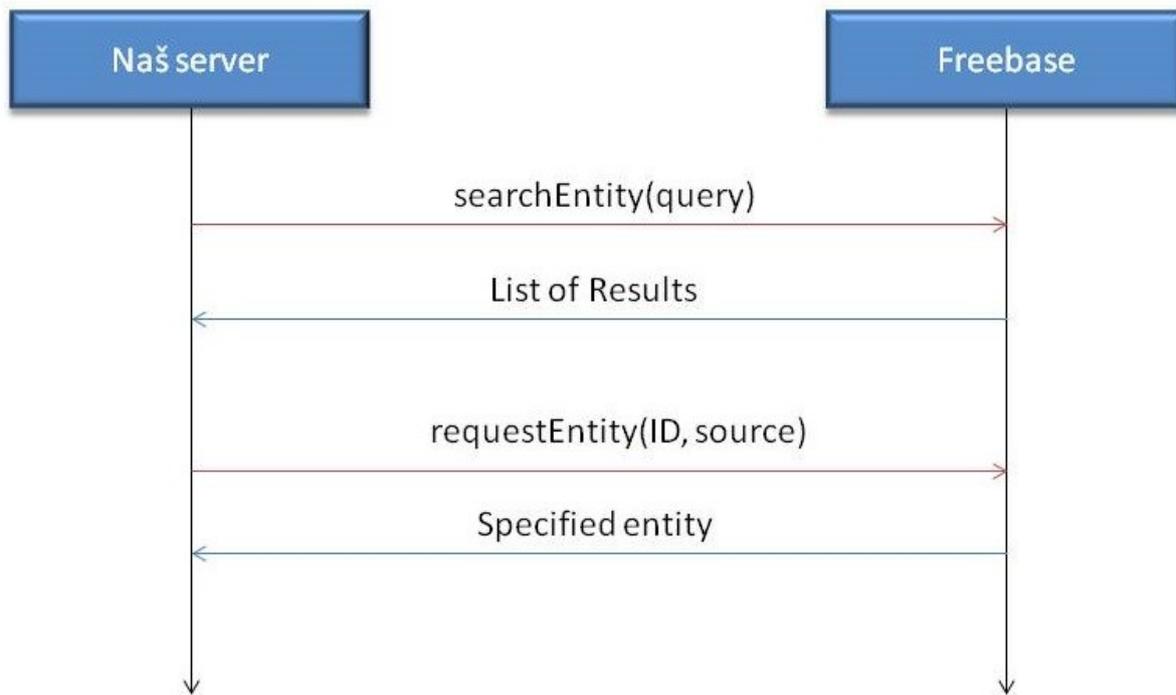
4.2 Modul **CMoreKnowledgeBase**

U okviru ovog modula je realizovana hijerarhija meta podataka. Modul ima dva paketa *com.rtrk.knowledgebase.api* i *com.rtrk.knowledgebase.entities*. Prvi sadži spregu (engl. interface) prema bazama znanja, a drugi sadži inicijalni skup klasa koje modeluju entitete nama od značaja.

4.2.1 Sprega **CMoreKnowledgeBaseApi**

Ova sprega sadži dve metode: *searchEntity* i *requestEntity*. Metoda *searchEntity* ima kao parameter objekat klase *KnowledgeBaseQuery* a kao povratnu vrednost ima listu objekata klase *Result*. Metoda *requestEntity* kao parameter uzima tačno određeni entitet odnosno njegov ključ u bazi znanja i identifikator same baze znanja.

Komunikacija je ilustrovana na slici 8. Pre nego što korisnik zatraži dodatni sadržaj, ka bazama znanja se šalje odgovarajući upit koji se formira spram naziva emisije, kategorije u koju je smeštena i dodatnih informacija. Kada takav upit stigne na stranicu kao što je Freebase, kao rezultat imamo listu rezultata koji odgovaraju datom opisu. Svaki od rezultata predstavlja vezu sa entitetom u njihovoj bazi. Kako bi dobili baš sam entitet, neophodno je u zahtev uključiti link do samog entiteta odnosno ključ. Iz tog razloga da kako bi došli do određenog entiteta je potrebno uputiti inicijalno dva zahteva. Jednom kada dobijemo entitet, njegov ključ se čuva kod nas u bazi kako ubuduće ne bi pravili nepotrebne zahteve jer se ključ neće menjati. Kada korisnik pritisne dugme na svom daljinskom upravljaču za dodatni sadržaj, on je već unapred pripremljen i čekanje je time znatno smanjeno a samim tim je i celokupan doživljaj bolji.



Slika 8. Primer komunikacije između našeg servera i Freebase-a

4.2.2 Klasa KnowledgeBaseQuery

Klasa *KnowledgeBaseQuery* modeluje upit koji se upućuje ka raznim bazama znanja. Cilj ove klase je da u malom broju reči što preciznije (idealno jednoznačno) opiše entitet koji je nama od interesa. Pored teksta samog upita, od interesa su i neke pomoćne informacije kao što su pre svega izvor i jezik na kom će pretraga biti obavljena, kao i tip entiteta u slučaju da imamo, na primer, više osoba sa istim imenom i prezimenom. Polja ove klase su: *queryText*, *language*, *source* i *type*.

4.2.3 Klase Entity i Result

Klasa *Entity* predstavlja klasu pretka za sve klase iz hijerarhije metapodataka. Sama klasa sadrži skup zajedničkih informacija za sve entitete u našem sistemu. Ima veliki broj apstraktnih potomaka koje imaju konkretne potomke u zavisnosti od baza znanja iz kojih se dobavljaju podaci.

Primer koji će biti objašnjen u narednih par poglavlja jeste primer glumca: svaki glumac je ujedino i osoba a osoba predstavlja jedan od entiteta u našem sistemu. Zato klasa *Entity* ima apstraktnog potomka, klasu *Person* koja u sebi sadrži bitna obeležja osoba za naš sistem. Takođe, klasa *Person* ima apstraktnog potomka, klasu *Actor* koja predstavlja sve glumce i sadrži relevantne informacije za nekog glumca. Na kraju hijerarhije nasleđivanja, imamo

konkretnu klasu *FreebaseActor* koja nasleđuje klasu *Actor*, i konkretizuje sve apstraktne metode svojih predaka. Sadrži veliki broj polja koja predstavljaju podskup informacija koje se mogu dobiti za određenog glumca sa sajta Freebase. Polja klase *Entity* su ime sadržaja (*name*), izvor sa koga pristižu informacije (*source*), ključ u bazi znanja (*foreignID*) i enumeracija koja sadrži sve tipove entiteta.

Klasa *Result* nasleđuje klasu *Entity* ali ne dodaje nova polja. Svrha ove klase je da omogući prikaz najosnovnijih informacija o nekom entitetu pre nego što se on dobavi. Ako se vratimo na sliku 8. vidimo da prilikom prvog upita za neki entitet na Freebase-u, dobijamo listu potencijalnih rezultata a ne samih entiteta. Ove rezultate je moguće prikazati i spram tih osnovnih informacija dobiti samo jedan entitet koji nam je od interesa umesto da dobivamo informacije o svim entitetima. Na ovaj način se performanse drastično povećavaju jer je dovoljno napraviti samo jedan upit za listu rezultata i jedan za neki konkretan entitet umesto jedan za listu rezultata i po jedan za svaki od entiteta u datoj listi.

4.2.4 Klasa Person

Ova klasa je realizovana kao apstraktna i pandan je klasi *Person* iz formalnog rešenja ontologije. Ipak, postoje razlike u odnosu na klasu predviđenu teorijskim rešenjem, radi optimizovanijeg rukovanja bazom podataka. Kako nasleđuje klasu *Entity*, koja u sebi već ima polje *name*, ime i prezime osobe je modelovano kao jedan znakovni niz. Ovakav vid rešenja je praktičan iz prostog razloga što neke osobe imaju veći broj imena i prezimena. Pored toga uvodi svoja polja zajednička za sve osobe kao što su *dateOfBirth*, *dateOfDeath*, *placeOfBirth*, *currentResidence* i *placeOfDeath*.

4.2.5 Klasa Actor

Pandan klasi *Actor* iz formalnog rešenja, ova klasa predstavlja zajedničkog pretka za sve glumce. Takođe je apstraktna i nasleđuje klasu *Person*. Sadrži minimalni skup informacija od interesa za sve glumce, u vidu liste filmova u kojima je glumac učestvovao (*starsIn*) i liste nagrada koje je glumac osvojio (*awards*).

4.2.6 Klasa FreebaseActor

Klasa *FreebaseActor* predstavlja konkretnu implementaciju klase koja modeluje glumca. Prilagođena je odgovoru koji se dobija sa stranice Freebase. U ovoj klasi je izdvojen podskup od dvadesetak obeležja, dok je njihov ukupan broj znatno veći i sadrži mnogo više detalja nego što je nama potrebno. Isto tako, u klasi su implementirane i sve metode klase *Actor* i *Person*.

Polja ove klase su mnogobrojna i formirana spram odgovora koji je rezultat upita ka Freebase-u. Neka od polja uključuju drugi nazivi za isti entitet (*alias*), lista nagrada koje je glumac osvojio (*awardsWon*), kratka biografija glumca (*description*), podaci o obrazovanju glumca (*education*), lista filmova u kojima je glumac glumio (*filmsActedIn*), poznate ličnosti sa kojima se glumac druži (*friends*), slike glumca (*image*), profile na raznoraznim socijalnim mrežama (*socialMediaPresence*), i mnoga druga.

Ostale klase predviđene ontologijom prate sličnu hijerarhiju – sve klase imaju svoju apstraktnu i konkretnu realizaciju. Tako imamo apstraktne klase kao što su *Director*, *Writer*, *Sportsman* i njihove konkretne realizacije *FreebaseDirector*, *Freebase Writer* i *WriterSportsman*.

5. Rezultati

Sama ontologija je realizovana kroz nekoliko modula koji su deo većeg sistema. Sistem se sastoji iz mnoštva međusobno spregnutih modula tako da nije moguće obaviti ispitivanje same ontologije bez ispitivanja celokupnog sistema koji u trenutku pisanja rada nije bio u celosti implementiran. Verifikaciju realizovanog dela sistema smo obavili korišćenjem FaCT++ rasuđivača (engl. reasoner) [10].

Rasuđivači predstavljaju primere programske podrške koji su u stanju da izvedu logičke posledice spram definisanih pravila i kao takvi se veoma često koriste u aplikacijama veštačke inteligencije i semantičkog Web-a. Oni kao ulaz koriste bazu znanja ili ontologiju, odnosno pravila koja su definisana unutar njih, i kao izlaz daju model u RDF-u ili nekom drugom jeziku/formatu. Za naše potrebe su korišćene samo osnovne mogućnosti rasuđivača koje se svode na proveru konzistentnosti ontologije. Napredne mogućnosti uključuju raznorazne logičke optimizacije (svode se na zakone Bulovalgebre poput apsorpcije) zarad generisanja što jednostavnijeg modela. Ove optimizacije nama nisu bile neophodne pošto je implementacija realizovana u Javi. Samim tim što je alat uspešno izgenerisao RDF model koji je u skladu sa zamišljenim, imamo neki rezultat koji ukazuje da ontologija ne sadrži semantičke greške.

Osnovna prednost i doprinos ontologije predstavlja semantičko bogatstvo, preciznost modelovanja i bolje mogućnosti pretrage. Odnos između broja klasa u formalnom i konkretnom rešenju dat je sledećom tabelom:

/	Teorijsko rešenje	Konkretno rešenje
Hijerarhija sadržaja (skup klasa naslednica klase Content/ContentItem)	7	5
Hijerarhija metapodataka (skup klasa naslednica klase Metadata/Entity)	15	16

Tabela 1. Odnos broja klasa u teorijskom i konkretnom rešenju

Možemo primetiti da je, radi optimizovanijeg rukovanja pre svega bazom podataka, broj klasa koji opisuje sadržaj u konkretnom rešenju manji. Realizacija hijerarhije metapodataka je analogna konceptualnom rešenju, sa izuzetkom klase *Result* čije postojanje nije ni predviđeno ontologijom jer nema nikakvo semantičko značenje. Ipak, u konkretnoj realizaciji, ova klasa predstavlja bitan element jer eliminiše nepotrebnu komunikaciju prilikom ponovnog dobavljanja određenog entiteta.

6. Zaključak

U radu je prikazano jedno rešenje ontologije za opis podataka u sistemu za dobavljanje raznorodnog sadržaja sa Interneta. Prikazano je formalno rešenje u vidu modela podataka realizovanog uz pomoć alata Protégé kao i implementacija ontologije u programskom jeziku Java. Ontologija je realizovana tako da izdvoji optimalan skup podataka potreban za rad našeg sistema u trenutku pisanja rada. Programsko rešenje je realizovano modularno a skalabilnost sistema će biti naknadno ispitano kada se realizuju i svi ostali moduli. Konzistentnost ontologije je proverena korišćenjem FaCT++ rasuđivača. U radu su prikazane prednosti razvijenog ontološkog modela, pored ostalog i kroz pretrage koje je moguće realizovati korišćenjem datog modela, a nije moguće realizovati korišćenjem ranijeg rešenja.

Što se tiče planova za dalji razvoj, ontologija će verovatno trpeti izmene kako bi ispratila razvoj sistema i nove zahteve. Inkrementalno poboljšanje ontologija bez velikih intervencija predstavlja jednu od njenih prednosti i samim tim opravdava razlog za njeno korišćenje. Dodavanje novih podataka i entiteta u ontologiju i njihovo povezivanje sa drugim entitetima ne bi trebalo da predstavlja problem u smislu drastičnih izmena u strukturi sistema.

7. Literatura

- [1] S. Pijetlovic, N. Jovanovic, N. Jovanov, S. Ocovaj, "One solution of a system for data acquisition and storage from the digital television transport stream and its exposure to the clients", 21st Telecommunications Forum TELFOR, Belgrade, November 2013.
- [2] Natalya F. Noy, Deborah L. McGuinness, "Ontology Development: A guide to creating your first ontology"
http://protege.stanford.edu/publications/ontology_development/ontology101-noy-mcguinness.html loaded 06.06.2014.
- [3] Ontology (Information science), Wikipedia
http://en.wikipedia.org/wiki/Ontology_%28information_science%29 loaded 06.06.2014.
- [4] Tom Gruber (1995). "Toward Principles for the Design of Ontologies Used for Knowledge Sharing", International Journal of Human-Computer Studies
- [5] Henry Kim, "Predicting how ontologies for the Semantic Web will evolve", Communications of the ACM February 2002/Vol. 45, No. 2
- [6] Myriam Lamolle, Chan Leduc, Ludovic Menet, "An Ontology-driven Architecture Approach for Open Multimedia Service Oriented Architectures", 2010. Third International Conference on Communication Theory, Reliability, and Quality of Service
- [7] Jorge Gómez-Montalvo, Ernesto Exposito, "A Semantic Approach to User-based QoS Provision for Multimedia Services in Home Networks", 2010. Third International Conference on Communication Theory, Reliability, and Quality of Service
- [8] TSI Digital Video Broadcasting (DVB), Specification for service information (SI) in DVB systems:

http://www.etsi.org/deliver/etsi_en/300400_300499/300468/01.11.01_60/en_300468v011101p.pdf loaded 06.06.2014.

[9] Freebase, <https://www.freebase.com/>, loaded 06.06.2014.

[10] Dmitry Tsarkov, Ian Horrocks, “FaCT++ Description Logic Reasoner: System Description”, School of Computer Science, University of Manchester, UK
<http://www.cs.ox.ac.uk/ian.horrocks/Publications/download/2006/TsHo06a.pdf> loaded 07.06.2014.