SVEUČILIŠTE U ZAGREBU
FAKULTET ORGANIZACIJE I INFORMATIKE
V A R A Ž D I N

Bogdan Okreša Đurić

SEMANTIČKO MODELIRANJE POSLOVNIH PRAVILA

DIPLOMSKI RAD

Varaždin, 2013.

SVEUČILIŠTE U ZAGREBU
FAKULTET ORGANIZACIJE I INFORMATIKE
V A R A Ž D I N

Bogdan Okreša Đurić
Matični broj: 39651/10–R
Studij: Baze podataka i baze znanja

SEMANTIČKO MODELIRANJE POSLOVNIH PRAVILA

DIPLOMSKI RAD

Mentor:
Doc.dr.sc. Markus Schatten

Varaždin, lipanj 2013.

Prije samog rada

zahvala obitelji, za bodrenje i podršku,

i prijateljima, a posebno:
Ozano Rokov,
Zrinka Šajn,
Zvonko Podbojec,
Iva Gregurec
i Maja Hornung.

Sadržaj

Sadržaj I
1 Uvod 1
2 Poslovna pravila 3
2.1 Model činjenica 4
2.2 Osnovna načela poslovnih pravila 8
2.3 Pravila ponašanja i pravila određivanja 8
2.4 RuleSpeak 10
2.5 Poslovni procesi? 13
3 Modeliranje poslovnih pravila 15
3.1 UML 15
3.2 ORM 20
3.3 Usporedba 25
4 Semantičko modeliranje 26
4.1 Ontologija 26
4.2 Semantički web 27
4.3 RDF 28
4.4 RDFS 30
4.5 OWL 32
4.5.1 Kardinalnost 33
4.5.2 Sinonimi 34
4.5.3 Svojstva 35
4.5.4 Skupni operatori 38
4.5.5 Ekstenzijsko određivanje i ograničavanje vrijednošću 40
4.5.6 Negacija 41
5 Semantičko modeliranje poslovnih pravila 42
5.1 SWRL (O'Connor, 2012) 42
5.1.1 SWRL atomi 42
5.1.2 Ugrađena svojstva 44
5.1.3 Disjunkcija? 46
5.1.4 OWL/XML zapis 46
5.2 RIF 48
5.2.1 Odnos RIF i SWRL 48
5.2.2 RIF sintaksa 49

5.3 Postupak prevođenja ... 51
6 Praktični primjer ... 53
6.1 Uvod u primjer ... 53
6.2 Primjer ... 53
7 Zaključak ... 66
Literatura ... 67
Dodatni elementi ... 69
Popis slika ... 69
Popis tablica ... 70
Popis poslovnih pravila ... 70
Popis isječaka koda ... 71
Popis SWRL pravila ... 73
Popis RIF pravila ... 73
Popis OCL izraza ... 73

1 Uvod

Tema ovog diplomskog rada nastala je kao rezultat zanimanja na području ontologija i poslovnih pravila, pojačana dodatnim utjecajem upravljanja znanjem, sustavima temeljenim na znanju, kao i višeagentnim sustavima. Spoj ovih područja informacijsko komunikacijskih tehnologija, uz odgovarajuću zastupljenost, prema interesu, uspješno je pronađen upravo u semantičkom modeliranju poslovnih pravila. Osim interesa u područjima informacijsko komunikacijskih tehnologija, ova je tema motivirana i interesom u području vođenja projekata i proučavanja poslovnih sustava. Prema tome, poslovna pravila dolaze iz poslovnog svijeta, gdje predstavljaju temelj poslovanja, jer svaki sustav, pa i onaj poslovni, mora biti temeljen na određenim pravilima koja usmjeruju poslovanje, u pozitivnom ili negativnom smjeru, a koja mogu poslužiti i za ispravljanje smjera kretanja takvog sustava.

Poslovna pravila temeljni su koncepti u poslovanju, stoga što određuju cjelokupno poslovanje i ponašanje sustava. S druge strane stoji semantičko modeliranje kojim se dobivaju semantički modeli, tj. ontologije. Takvo modeliranje, pak, svoje temelje traži u informatičkim granama te nije izravno povezano sa poslovanjem ili poslovnim sustavima.

Modeliranje uz pomoć računala oduvijek je prikazivano kao modeliranje za čovjeka, gdje računalo pomaže svojim memorijskim mogućnostima, brzinom obrade podataka te činjenicom da ga ne može savladati umor, glad ili slične ljudske osobine. Ontologijama je omogućeno da model koji razvije čovjek, računalo može shvatiti i interpretirati, čime je neusporedivo povećana moć računala. Mogućnost shvaćanja odnosa koncepata, definicije određenog koncepta i njegovih svojstava, sve do definiranja domena iznimnog opsega na računalu shvatljiv način, otvara novu paletu mogućih primjena računala.

Semantičko modeliranje poslovnih pravila objedinjuje ova dva opisana koncepta, jedan iz domene primarno poslovnih sustava, drugi iz domene primarno informacijsko komunikacijske tehnologije. Spojem obaju koncepata moguće je proizvesti sustav koji nije izrazito kompleksan za korisnika, omogućava računalno definiranje poslovnih pravila na način koji je razumljiv i korisniku i računalu, a samim time jača ulogu računala u poslovnom sustavu, gdje računalo može dodatno pomoći u donošenju odluka.

Upravo je taj spoj poslovnog i informatičkog sadržaja glavna zanimljivost teme ovog diplomskog rada. Naime, uvriježeno je mišljenje da je student pretežito informatičkog područja informatičar te da kao takav ima, gotovo uvijek, isključivo znanje programiranja, te da svaki informatičar poznaje sve tajne bilo kojeg problema koji je, čak i marginalno, povezan s računalom. Jedan od ciljeva ovog rada je prikazati širinu polja informatičkih znanosti i raznovrsnost primjene informatike te samim time i specijalizacije studenata studija informatičkog područja. Upravo zbog navedene širine i raznovrsnosti, važno je prikazati koliko je znanje informatike moguće primijeniti i na ostala područja, a ne isključivo programiranje, dizajn i ostala uobičajena područja.

Također, ova tema prikazuje koliko je duboko moguća kvalitetna bliska suradnja informatičke struke sa poslovnim sustavima. Dobro je poznato da se u poslovnim organizacijama informatička struka ograničava na baze podataka, obradu i analizu podataka, izradu softvera i web-stranica, no to nije sve.

U sljedećim poglavljima bit će obrađeni temelji i primjeri poslovnih pravila u poglavlju 2, zatim načini modeliranja poslovnih pravila i njihova usporedbu u poglavlju 3, nakon čega slijedi upoznavanje ontologija i semantičkog modeliranje kroz definicije i primjere u poglavlju 4, dok će poglavlje 5 obuhvatiti razradu teorije i primjera semantičkog modeliranja poslovnih

pravila. Poglavlje 6 prikazat će primjer obrade poslovnog pravila i aplikacije razvijene za definiranje poslovnih pravila i rezoniranje nad njima.

Za potrebe izrade ovog rada korišteni su sekundarni izvori u obliku članaka i opisa standarda, kako je to prikazano u poglavlju Literatura. Korišteni su izvori dostupni putem mrežnih stranica udruga (poput *BRCommunity*) ili baza članaka. Svi prikazani primjeri, a posebno primjer na kraju rada, predstavljaju primarni izvor podataka, tj. autorsko su djelo autora ovog rada, nastali uz više ili manje utjecaja iz navedenih sekundarnih izvora.

2 Poslovna pravila

Za početak, a u svrhu kasnijeg razmatranja, potrebna je definicija pojma „pravilo“:

> *stalan, čvrst, nepromjenljiv, uzajaman odnos kakvih pojava, u kojem dolazi do izražaja određena zakonitost; princip*[1]
>
> *princip ili regulativa koja određuje aktivnosti, akciju, postupak, događaj i sl.*[2]

Poslovna pravila predstavljaju koncept koji se koristi od kad je čovjek počeo timski i sustavno djelovati, no prva suvisla i objavljena definicija, sa sintagmom „poslovno pravilo“, javlja se 1984. godine:

> *[...] eksplicitna izjava ograničenja koja postoji unutar poslovne ontologije [...]*[3]

Istraživanje i definiranje poslovnih pravila razvijalo se kroz godine, pa nastaje definicija koja predstavlja prvi pokušaj da se poslovna pravila bilježe i kategoriziraju deklarativno, temeljem modela podataka:

> *[...] određena pravila (ili poslovna politika) koja uređuju [...] ponašanje [organizacije] i razlikuju je od ostalih [...] takva pravila uređuju promjene u stanju organizacije [...]*[4]

U jednom od najcitiranijih izvora u slučaju rasprave o poslovnim pravilima, *GUIDE Business Rules Project Reportu*, poslovno je pravilo definirano kako slijedi:

> *[...] izjava koja određuje ili ograničava neke aspekte poslovanja [...] [koja] bi trebala odrediti poslovnu strukturu, ili kontrolirati utjecaj ponašanja na poslovanje. [Poslovno pravilo] ne može se razbiti [na manje dijelove] ili dodatno dekomponirati u detaljnija poslovna pravila [...] ukoliko ih dodatno smanjimo, dolazi do gubitka podataka važnih za poslovanje. (Business Rules Group, 2000, pp. 4-5)*

Nadalje, koncept poslovnih pravila se većinom razvijao u informatičkom kontekstu, gdje je poslovni kontekst zapostavljen utjecajem informatičkog konteksta te činjenicom da se poslovna pravila zapisuju pomoću računala i da računala sadrže podatke važne za poslovanje organizacije. 2000. godine Ross u svojim materijalima za seminar održan u Bostonu, MA, od 19. do 21. lipnja, definira poslovna pravila kako slijedi:

> *Atomarni dio ponovno iskoristive [eng. re-usable] poslovne logike određen deklarativno.*

Pa ipak, što poslovna pravila zapravo jesu? Ross (2000) pojašnjava da su poslovna pravila ono što organizaciju vodi u svakodnevnim aktivnostima, iz čega proizlazi da poslovna pravila služe za standardizaciju poslovanja i kao pomoć u donošenju poslovnih odluka. Ukoliko poslovna pravila ne bi postojala, svatko tko ima mogućnost donošenja odluka, neovisno o hijerarhiji unutar organizacije, mogao bi odluku donijeti temeljem vlastitog iskustva, stava i mišljenja. Takav način donošenja odluka doveo bi do kaosa i različitih rezultata na jednake

[1] http://hjp.novi-liber.hr/index.php?show=search_by_id&id=eVhkWBQ%3D&keyword=pravilo, pristup 2.10.2012.
[2] http://dictionary.reference.com/browse/rule, pristup 2.10.2012.
[3] Daniel S. Appleton (15. listopada 1984.). Business Rules: The Missing Link. *Datamation*, str. 145-150.
[4] Ronald G. Ross (1987.). Entity Modeling: Techniques and Application. *Database Research Group, Inc.*

ulaze diljem organizacije. Prema tome, Ross napominje da svaki organizirani poslovni proces ima poslovna pravila.

Važno je shvatiti da poslovna pravila, sama po sebi, ne rade ništa. Ona su samo izjave koje uređuju kako će se nešto odvijati. Roger Burlton (Ross, 2000) lijepo je sažeo svrhu poslovnih pravila ovako: „Odvojiti odvijanje od znanja.“ (eng. *Separate the flow from the know*.) Ova jednostavna rečenica kazuje da poslovna pravila (znanje) uređuju i upravljaju odvijanjem npr. poslovnih procesa. Koncept poslovnog pravila, suprotno potencijalnom prvom dojmu, a u skladu sa činjenicom da predstavlja element znanja, uvijek ima oblik uvjeta (definicije), činjenice (tipa) ili pravila. U pravilu, poslovno pravilo mora biti nešto od ta tri koncepta, gdje je uvjet temeljni oblik, na koji se nadovezuje činjenica, na koju se nadograđuje pravilo. Sukladno navedenom pravilu (Business Rules Group, 2000, pp. 4-5), cilj definiranja poslovnog pravila je da ono bude, na određeni način, minimalno, tj. da ga nije moguće razbiti na više manjih pravila. Primjer poslovnog pravila je sljedeće:

> *PP 2.1: Redoviti student pri upisu više godine studija, sa, uključivo, više od 55 ECTS bodova položenih na prethodnoj godini studija, ne plaća školarinu.*

Poslovno pravilo predstavlja onaj dio znanja organizacije koji je zapisan, tj. koji je eksplicitno prikazan. Implicitno znanje, koje zaposlenici imaju kao dio svog iskustva i osobnog znanja, a koje nije nigdje fizički izraženo, neovisno o utjecaju na poslovanje, ne predstavlja poslovna pravila zato što nije dostupno svima. Takvo znanje potrebno je izraziti eksplicitno, kako bi ono postalo dio poslovnih pravila organizacije. Razlog tome je činjenica da implicitno znanje zaposlenici „nose“ sa sobom, čime se iznimno ograničava iskoristivost tog znanja, kao i rizik od nestajanja istog. Primjerice, ekspert iz područja javne nabave, u nekoj organizaciji, koji znanje drži za sebe, bez njegove eksplikacije, može svoju organizaciju napustiti, čime povlači i sve svoje znanje potrebno za kvalitetan rad organizacije. Da je takvo znanje eksplicitno izraženo, npr. zapisano u obliku nekih pravila, iskustva ili uputa, znanje bi lakše ostalo u organizaciji čak i kad je ekspert napusti. Prema tome, navedeno poslovno pravilo predstavlja eksplicitni dio znanja koje određuje upise na fakultetu. Drugim riječima, poslovna su pravila „kodirano znanje poslovnog djelovanja.“ (Ross, 2000) Pritom je riječ upravo o znanju koje je temelj upravljanja znanjem.

2.1 Model činjenica

Kao što je spomenuto, poslovna pravila moraju biti shvatljiva sama po sebi, bez dodatnih pojašnjenja pojmova, svakom korisniku. Kako bi takvo definiranje bilo moguće, organizacija uz poslovna pravila mora definirati i model činjenica (eng. *fact model*). Model činjenica predstavlja strukturirani poslovni rječnik koji opisuje sve korištene pojmove i njihova značenja, u svrhu lakšeg shvaćanja poslovnih pravila. Takav rječnik u sebi sadrži imenice, glagole, pridjeve i sve ostale oblike koji pomažu u definiranju sadržaja određenih koncepata koje organizacija treba za sklapanje poslovnih pravila, kao i odnose tih koncepata. Organizacija u danom trenutku može imati iznimno velik broj poslovnih pravila, gdje se ona obično broje u stotinama, pa je iznimno važno uskladiti značenje gotovo svake pojedine riječi koja se u njima koristi. Model činjenica pomaže upravo u tome, a služi i za razvijanje poslovnih pravila, tj. potrebe razvijanja skupa poslovnih pravila, kao i za usklađivanje njihovog značenja. Važno je napomenuti da poslovna pravila trebaju biti jasnog sadržaja čak i izvan konteksta u kojem su prvotno zamišljena, (Ross & Lam, 2011) čemu pridonosi upravo model činjenica. Štoviše, poslovno pravilo mora biti jednoznačno i određeno opsegom svojeg sadržaja, čime se isključuje mogućnost da isto poslovno pravilo bude interpretirano na više različitih načina. Izjava PP 2.2 predstavlja nekakav slabi oblik pravila.

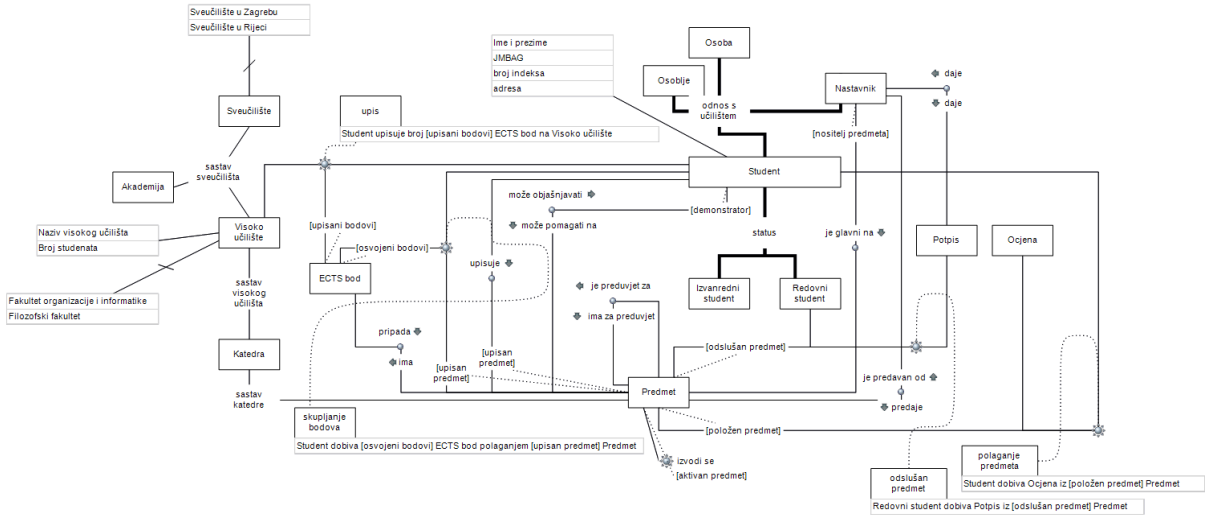
PP 2.2: Srebrni kupac kupuje često.

PP 2.2 određuje status *srebrnog kupca*, koji je u praksi definiran kao kupac koji *obavi dvije kupnje mjesečno*. Navedena izjava nije dovoljno određena: koji raspon vremena obuhvaća pojam *često*; koliko kupac puta treba kupovati; koliki iznos kupac treba potrošiti?

Kao što je navedeno, model činjenica sadrži sve koncepte koji su potrebni za poslovna pravila organizacije, kao i odnose među njima. S obzirom na velik broj koncepata i ovisno o veličini organizacije, moguće je da se u modelu činjenica pojave sinonimi. U takvom je slučaju potrebno odrediti simbole (riječi) koji su poželjniji za korištenje i one koji su manje poželjni. Dakako, u modelu činjenica mogu se pojaviti i homonimi, pri čemu je korisno osigurati da se uz homonim uvijek navodi i kontekst u kojem se isti koristi, kako bi bilo jasno točno značenje iskorištenog simbola. Osim toga, korisno je pokušati djelovati na članove organizacije da se usklade u korištenju pojmova i njihovih značenja, čime bi se izbjegli nesporazumi i krive interpretacije. Kao što je već spomenuto, a prema (Gerrits, 2012), model činjenica, osim koncepata, sadrži i veze među njima, koje mogu biti:

- kategorizacija, npr. *student* je kategorija od *osoba*; *redoviti student* je kategorija od *student* i sl.;
- svojstva, npr. *student* ima *adresu*; *osoba* ima *spol* i sl.;
- klasifikacija ili izbor, npr. FOI je instanca od *fakultet*;
- objektifikacija (eng. *objectification*), npr. „Osoba upisuje fakultet“ objektificirano je kao „upis.“;
- uloge, npr. Osoba [student] upisuje fakultet;
- kompozicija, npr. *kuća* se sastoji od *zida* (*zid* je sastavni dio *kuće*) i sl.

Model činjenica uobičajeno se prikazuje grafički kao dijagram, kao na Slika 2.1, gdje su prikazani svi navedeni koncepti modela činjenica.

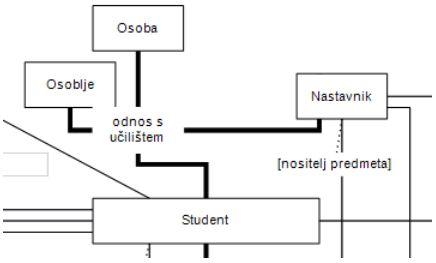


Slika 2.1: Dijagram dijela modela činjenica sustava visokog školstva Hrvatske. (izvor: izrada autora)

Model činjenica na Slika 2.1 opisuje dio sustava visokog školstva u Hrvatskoj, prikazom odnosa koncepata studenta, nastavnika, predmeta i visokih učilišta, zajedno sa svim potrebnim

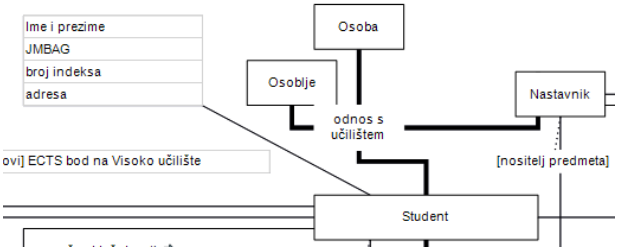
pratećim konceptima i njihovim vezama. Od navedenih elemenata modela činjenica, moguće je jasno uočiti sljedeće:

- kategorizacija – koncepti *Student*, *Nastavnik* i *Osoblje* kategorije su koncepta *Osoba*, kroz shemu *odnos s učilištem*[5], prikazano na Slika 2.2;



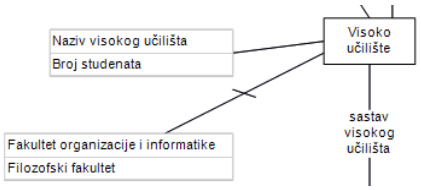
Slika 2.2: Kategorizacija u primjeru modela činjenica. (izvor: izrada autora)

- svojstva – *Student* ima pridružena određena svojstva (*Ime i prezime*, *JMBAG*, *broj indeksa* i *adresa*), prikazano na Slika 2.3;



Slika 2.3: Svojstva koncepta *Student* u primjeru modela činjenica. (izvor: izrada autora)

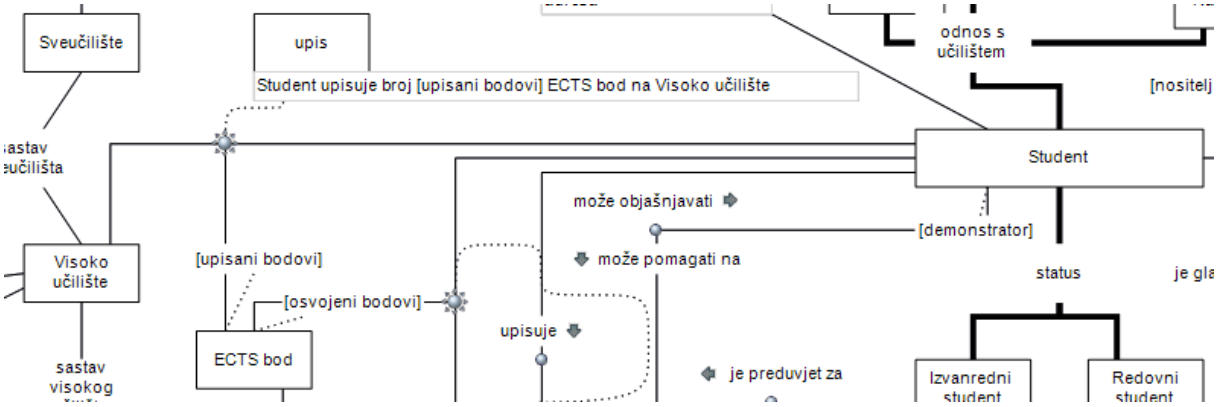
- klasifikacija – koncept *Visoko učilište* ima instance *Fakultet organizacije i informatike* i *Filozofski fakultet*, prikazano na Slika 2.4;



Slika 2.4: Klasifikacija koncepta *Visoko učilište* u primjeru modela činjenica. (izvor: izrada autora)

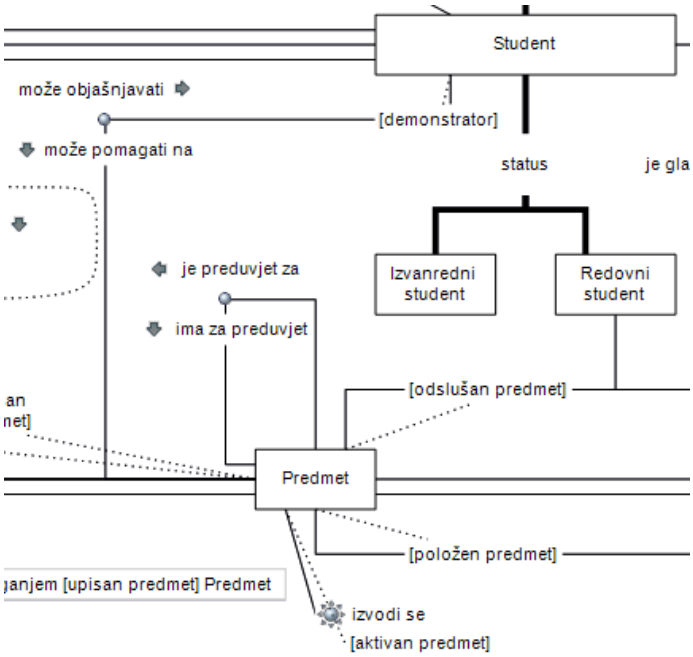
- objektifikacija – vezu koja opisuje upis studenta na visoko učilište, a koja je opisana riječima *Student upisuje broj [upisani bodovi] ECTS bod na Visoko učilište* pretvorena je u zasebni izraz (eng. *term*) *upis*, prikazano na Slika 2.5;

[5] shema kategorizacije obilježava skup koncepata; moguće je imati više različitih shema kategorizacije



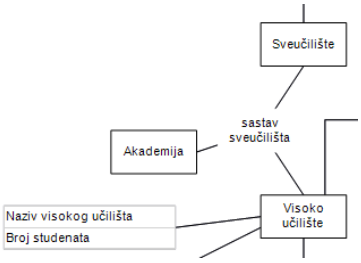
Slika 2.5: Objektifikacija veze *Student – Visoko učilište – ECTS bod* u primjeru modela činjenica. (izvor: izrada autora)

- uloge – student koji pomaže na nekom predmetu ima ulogu *[demonstrator]*, i obrnuto, neki predmet može (službeno) objašnjavati student u ulozi demonstratora; također, neki predmet koji se izvodi ima ulogu *[aktivan predmet]*; uloga se u programu FactXpress označava uglatim zagradama, prikazano na Slika 2.6;



Slika 2.6: Uloge u primjeru modela činjenica. (izvor: izrada autora)

- kompozicija – *Sveučilište* se sastoji od koncepata *Akademija* i *Visoko učilište*, prikazano na Slika 2.7.



Slika 2.7: Kompozicija koncepta *Sveučilište* u primjeru modela činjenica. (izvor: izrada autora)

2.2 Osnovna načela poslovnih pravila

Izjava PP 2.2 može biti dio poslovne politike, no sama po sebi ne predstavlja poslovno pravilo. Poslovno pravilo koje opisuje navedenu izjavu je temeljno poslovno pravilo, zato što je izvedeno izravno iz poslovne politike, temeljem poslovnog djelovanja. Poslovna politika pruža temeljne smjernice poslovanja, no iste nisu dovoljno određene da bi bilo moguće predstaviti ih zaposlenicima ili računalima bez izmjena.

Ross i Lam odredili su nekoliko osnovnih načela poslovnih pravila (2011), po potrebi dodatno pojašnjena kako slijedi:

- Niti jedno poslovno pravilo nije vječno.
    - Poslovna pravila podložna su izmjenama. Činjenica je da se poslovanje organizacije kroz vrijeme mijenja, pa je potrebno sukladno tome mijenjati i poslovna pravila
- Poslovno pravilo mora imati globalnog smisla, kroz cijeli arhitekturni okvir.
- Nema poslovnih pravila sve dok se ne kaže da ona postoje.
- Poslovno pravilo znači upravo ono što je određeno korištenim riječima – ništa više, ništa manje.
- Poslovno je pravilo uvijek primjenjivo.
    - Poslovno pravilo tako je postavljeno da je uvijek moguće točno odrediti njegovu primjenjivost, tj. uvijek je jasno što ono znači i na što se odnosi.

U ovim je načelima spomenuto da poslovna pravila ne postoje sve dok se ne kaže da ona postoje, tj. dok se ne odrede, te da ih je potrebno mijenjati. U skladu s time, a prema (Ross, 2010), poslovno pravilo je pravilo koje je u nadležnosti poslovne organizacije. Prema tome, pravilo koje nije u nadležnosti organizacije, nije poslovno pravilo. Ranije je spomenuta početna definicija pojma pravilo koja uključuje odnos pojava i izražavanje određene zakonitosti. Prema takvoj definiciji, pravilo je bilo kakva izjava koja izražava odnos određenih koncepata, gdje taj odnos može odražavati nekakvu zakonitost. Takve definicije, pravila, moguće je prepoznati u svim područjima života, znanosti i prirode, pa je pravilo i poznati Pitagorin poučak ili tzv. zlatno pravilo u umjetnosti, zatim zakon gravitacije i sl. Razlog zašto navedena i ostala pravila nisu poslovna pravila je činjenica da navedeno nije u nadležnosti neke organizacije, već, općenito gledano, van utjecaja čovjeka, pa time i bilo koje organizacije.

2.3 Pravila ponašanja i pravila određivanja

U poslovnom svijetu, pravila služe kako bi unijela određena ograničenja. Bilo koje pravilo u svom sadržaju svakako postavlja ograničenja, tj. može se reći da umanjuje ili ograničava slobodu. Kad pravilo ne bi djelovalo na takav način, onda isto ne bi bilo pravilo, već bi svojim usmjeravanjem aktivnosti, bez postavljanja ograničenja, iako vjerojatno korisno, služilo isključivo upravo usmjeravanju, davanju savjeta, čime postaje svojevrsni savjet. Prema

(Ross, 2010), sva pravila mogu se podijeliti u dvije velike kategorije: **pravila ponašanja** ili operativna pravila i **pravila određivanja** ili strukturalna pravila.

Određena poslovna pravila moguće je automatizirati korištenjem moderne tehnologije, čime se olakšava proces donošenja odluka ili jednostavnog svakodnevnog rada organizacije. S druge strane, postoje poslovna pravila koja nije moguće automatizirati, a koja se često odnose na ponašanje osoba te ih je potrebno uključiti u aktivnosti osoba organizacije. Poslovno pravilo PP 2.3 predstavlja jedan takav primjer.

PP 2.3: Za vrijeme zime, kaput se mora ostaviti u garderobi.

Poslovno pravilo PP 2.3 nije moguće jednostavno automatizirati pomoću računalne tehnologije. Jasno, bilo bi moguće postaviti senzore pri ulazu ili osigurati da osobe prođu pored garderobe te da ih sustav prepozna ukoliko nisu ostavili kaput, no, u jednostavnim uvjetima, ovo je pravilo teško provoditi zato što se odnosi isključivo na ponašanje ljudi.

Kao što je spomenuto, poslovno pravilo općenito uvodi određeno ograničenje u poslovanje. Ukoliko izjava okarakterizirana kao poslovno pravilo nema to svojstvo ograničavanja, ista je, kao što je ranije spomenuto, savjet. Izjava PP 2.4 predstavlja savjet.

PP 2.4: Na predavanje student može nositi bilježnicu.

Navedena izjava PP 2.4 ne uvodi nikakvo novo ograničenje niti obvezu, no ne postavlja niti ograničenje nad znanjem dane aktivnosti. Prema tome, takva je izjava savjet, koji nije nužno zapisati uz poslovna pravila. Uobičajeno je da se savjeti zapisuju, čime ih se dovodi gotovo na istu razinu sa poslovnim pravilima, osim činjenice da ništa izravno ne ograničavaju. Dakako, nije svaki savjet potrebno zapisati uz poslovna pravila, no neki savjeti imaju određenu težinu te su npr. nastali kao rješenje konflikta unutar radne okoline, pa ih je korisno zapisati (Ross, 2010).

Kao što je ranije navedeno, poslovna pravila se općenito dijele u dvije velike kategorije, od kojih jedna obuhvaća pravila ponašanja. Pravila ponašanja takva su poslovna pravila koja je moguće prekršiti određenim postupkom, kao pravilo PP 2.5.

PP 2.5: Pozvani predavač mora dobiti pratnju do dvorane za predavanje.

Poslovno pravilo PP 2.5 predstavlja pravilo ponašanja. Ovo je konkretno pravilo moguće prekršiti jednostavnim postupkom nedavanja pratnje pozvanom predavaču u trenutku kad se isti pojavi na recepciji. Kao što je to vidljivo u navedenom pravilu, pravila ponašanja uobičajeno djeluju preventivno. U ovom slučaju, pokušava se spriječiti situacija u kojoj se pozvani predavač nalazi izgubljen; ili se pokušava spriječiti nepokazivanje poštovanja prema pozvanom predavaču. Ross (2010) za pravilo ponašanja navodi da takva pravila omogućavaju organizacijama da obavljaju aktivnosti „na način koji je procijenjen kao prikladan, optimalan ili najbolje usklađen sa njenim ciljevima.“ S obzirom na činjenicu da je pravila ponašanja moguće prekršiti, u poslovnom je sustavu potrebno svakom takvom pravilu pridodati razinu važnosti, tj. intenzitet primjene tog pravila.

Osim savjeta, slabiji oblik poslovnog pravila su i smjernice. Zanimljivo je da se intenzitet poslovnog pravila (savjet, smjernica, pravilo) iznimno lako određuje riječima koje su korištene pri konstrukciji pravila. Takav sustav zahtijeva iznimno točno korištenje danih riječi, kao i dobro definiran model činjenica spomenut ranije. Smjernica je izjava u obliku poslovnog pravila, no nema toliko izraženu zabranu, kao PP 2.6.

PP 2.6: Nakon upisnog roka, student bi pri upisu trebao platiti zakasninu 300,00 kn.

U PP 2.6 moguće je primijetiti korištenje sintagme „trebao bi“ kojom se ukazuje na određeno ograničenje, no ono nema intenzitet kao u slučaju korištenja riječi „mora.“ Prema tome, PP 2.6 predstavlja smjernicu koje bi se trebalo pridržavati; većinom je istinita, no ne postoji njeno sustavno primjenjivanje.

Pravila određivanja pak uvijek odaju smisao nužnosti ili nemogućnosti (Ross, 2010). Upravo stoga se preporuča korištenje ključnih riječi „uvijek“ i „nikad.“ Bitno je napomenuti da, za razliku od pravila ponašanja, gdje je svako pravilo ponašanja poslovno pravilo, svako pravilo određivanja nije poslovno pravilo, već samo ono koje je u izravnoj nadležnosti neke poslovne organizacije, koja dano pravilo može u danom trenutku izmijeniti. Nadalje, pravilo određivanja gotovo uvijek klasificira ili računa nešto temeljem svih dostupnih činjenica. Pravila određivanja „ukazuju kako poslovna organizacija organizira (strukturira) svoje osnovno znanje.“ (Ross, 2010) Bitno je ovdje napomenuti da pravilo određivanja (eng. *definitional rule*) ili definiranja nije isto što i definicija (eng. *definition*): definicija opisuje osnovne značajke nekog koncepta, dok pravilo određivanja postavlja točne granice kojima je opisani koncept definiran, čime se omogućava točno određivanje elemenata ekstenzije danog koncepta. Dakle, definicija predstavlja intenziju koncepta, dok pravilo određivanja postavlja uvjete za elemente ekstenzije.

Poslovna su pravila u svojim temeljima određena manifestom o poslovnim pravilima (Business Rules Group, 2003).

2.4 RuleSpeak

Grupacija *Business Rule Solutions* održava RuleSpeak, upute za standardizirani zapis poslovnih pravila u skladu sa SBVR (eng. *Semantics of Business Vocabulary and Business Rules*, u slobodnom prijevodu *Semantika poslovnog rječnika i poslovnih pravila*). Upute se bave poželjnim korištenjem pojmova prilikom definiranja poslovnih pravila, kako bi se poslovna pravila, čak i u prirodnom jeziku, donekle standardizirala. Glavni razlog standardizacije izričaja poslovnih pravila je i unapređenje komunikacije između osoba iz poslovanja, poslovnih analitičara i informatičara, što bi dovelo do izbjegavanja prepreka koje se najčešće javljaju u jeziku koji je korišten za izražavanje poslovnih pravila. Lako je razumljivo da je za kvalitetnu komunikaciju bitno izraziti smisao poslovnog pravila na jasan način, nedvosmisleno, a ipak korištenjem dobro strukturiranih rečenica. Bitno je ovdje naglasiti da poslovno pravilo, i prema uputama RuleSpeak, uobičajeno ograničava ili smanjuje slobodu djelovanja.

Prema razini specifičnosti, poznata su tri oblika izraza (Business Rule Solutions, 2009, pp. 2-3):

- upravljajuća izjava – obično izjava korištena u pravu, statutima i ostalim regulama, kao i poslovnoj politici, općenito, u situacijama kada je potrebno poslovne aktivnosti uskladiti sa nekim općim ciljevima;
- primjenjiva izjava – uobičajeno je deklarativna izjava jasne strukture u prirodnom jeziku, jasno razumljiva i jednostavno iskoristiva od strane zaposlenika prilikom provođenja poslovnih procesa ili odlučivanja;
- implementacijska izjava – izjava koja je oblikovanjem spremna za izradu ili implementaciju nekog automatiziranog sustava.

Nekoliko je osnovnih uputa u RuleSpeaku, kojima se postavljaju temelji dobrih izjava poslovnih pravila (Business Rule Solutions, 2009, pp. 4-5):

- Izjava poslovnog pravila ne bi trebala biti proceduralnog oblika, zato što su takvi oblici daleko od prirodnog jezika, ubrzo postaje teško pratiti ih i otežano je njihovo ponovno korištenje u sklopu poslovnog sustava.

- Izjave poslovnih pravila moraju biti jasne, zbog jednostavnosti njihove primjene, izjava poslovnog pravila mora biti jedinstvenog značenja i uvijek iste interpretacije.
- Ograničenje i provođenje su dvije odvojene stvari, zato što izraz poslovnog pravila mora svojim značenjem odgovoriti isključivo na „što?“ dio pitanja samog poslovnog pravila, a ne i dodatna opisna pitanja koja služe provođenju pravila, kao „tko? (je odgovoran za provođenje pravila)“, „kada? (treba pravilo koristiti)“, „kako? (treba procjenjivati pravilo i provoditi ga)“ i slično; primjer je opisan kao PP 2.7 u kojem nije definirano *kako* će pravilo biti provođeno, u *kojem* dijelu poslovnog sustava će ono biti implementirano, *tko* je odgovoran za njegovo provođenje ili *kada* ga je točno potrebno provoditi, pa ipak, ima jasnu poruku o tome *što* je potrebno uraditi.

  PP 2.7: Astronautsko odijelo mora biti nošeno prilikom šetnje svemirom.

Osim temeljnih uputa o sadržaju izraza poslovnih pravila, RuleSpeak, u svrhu standardizacije i jednostavnijeg shvaćanja značenja izraza poslovnih pravila, predlaže i određene riječi za korištenje u izrazima poslovnih pravila (Business Rule Solutions, 2009, p. 6):

- Izraz poslovnog pravila trebao bi sadržavati *ključnu riječ pravila*, tj. budući da pravilo po svojoj prirodi ograničava slobodu, svako bi pravilo trebalo sadržavati oblik riječi „morati“ ili „samo“, kao što je to u PP 2.9, koje predstavlja unapređenje pravila PP 2.8 koje nije standardizirano.

  PP 2.8: Indeks upisanog studenta trebao bi biti potpisan.

  PP 2.9: Indeks upisanog studenta mora biti potpisan.

- Kao što je navedeno, poslovno pravilo uklanja određen stupanj slobode, pa nije dobro koristiti riječ „može“ kojom se jednostavno ističe činjenica, kao što je u PP 2.10, gdje se standardizacijom dobiva puno jasnije pravilo PP 2.11 koje unosi određeno ograničenje (radno vrijeme).

  PP 2.10: Student može dobiti potvrdu o upisu u studentskoj referadi.

  PP 2.11: Student može dobiti potvrdu o upisu u studentskoj referadi samo za vrijeme rada sa strankama.

- Nije potrebno dodavati nepotrebne riječi kako bi se pojačao utjecaj osnovnih potrebnih riječi u izrazu, kao u pravilu PP 2.12, gdje se uklanjanjem suvišnih riječi zadržava značenje izraza pravila PP 2.13.

  PP 2.12: Student mora potpisati ugovor prilikom upisa, bez iznimaka.

  PP 2.13: Student mora potpisati ugovor prilikom upisa.

Nadalje, izrazi poslovnih pravila temelje se na određenim činjenicama, poput *ECTS bodovi pripadaju predmetu*. Upravo zbog toga iznimno je korisno da poslovna organizacija ima dobro strukturiran i organiziran poslovni rječnik. Navedena činjenicu može se iskoristiti kao u pravilu PP 2.14 gdje se koristi ista ključnu riječ kao u izrazu činjenice. Pritom je bitno paziti da

u izrazu poslovnog pravila budu navedene sve činjenice koje su potrebne za njegovu nedvosmislenu interpretaciju.

> *PP 2.14: Student polaganjem predmeta osvaja samo ECTS bodove koji pripadaju tom predmetu.*

U svrhu povećanja jasnoće značenja izraza poslovnog pravila, preporuča se korištenje riječi koje što bolje opisuju idejno značenje, a ne izrazi koji omogućavaju pogrešnu interpretaciju (Business Rule Solutions, 2009, p. 8), kao u izrazu pravila PP 2.15, gdje nije jasno točno koju ulogu Studentski zbor mora imati u projektu: Financira li Studentski zbor projekt? Vodi li Studentski zbor projekt? Ima li Studentski zbor predstavnike u organizacijskom odboru projekta? Pravilo PP 2.16 osigurava točno tumačenje točnim izrazom.

> *PP 2.15: Projekt mora podržati Studentski zbor.*
>
> *PP 2.16: Projekt mora voditi Studentski zbor.*

Nadalje, RuleSpeak predlaže da izrazi poslovnih pravila ne počinju riječju *ako*, usprkos činjenici da je većinu poslovnih pravila moguće izraziti u *ako-onda* obliku (Business Rule Solutions, 2009b). Glavni razlog ove upute je jednostavnost korištenja i činjenica da je rečenica koja započinje subjektom prirodnija od rečenice oblika ako-onda. Štoviše, ne potiče se korištenje subjekta u množini, već isključivo u jednini, zato što se za imenicu u jednini, naročito ako ista ima funkciju subjekta, podrazumijeva da se odnosi na svaku instancu tog koncepta, dok je, po potrebi, moguće dodati i riječ „svaki“ ispred subjekta. Pravilo PP 2.17 predstavlja nestandardizirani izraz poslovnog pravila, dok je PP 2.18 unaprijeđen, pojednostavnjen, standardiziran oblik istog značenja.

> *PP 2.17: Ako su studenti počinili stegovni prekršaj, onda moraju pristupiti stegovnom sudu.*
>
> *PP 2.18: (Svaki) Student koji je počinio stegovni prekršaj mora pristupiti stegovnom sudu.*

Vremensko određivanje izraza poslovnog pravila poželjno je stavljati na kraj rečenice, a ne na početak, iako to u nekim slučajevima možda izgleda jednostavnije (Business Rule Solutions, 2009, pp. 10-11). U slučaju numeričkog određivanja, tj. ograničenja iskazanog brojem, dobro je u ulogu subjekta staviti koncept koji je prebrojiv, koji realno može za ograničenje imati navedeni broj. Izraz PP 2.19 ima subjekt koji nije prebrojiv i na koji se ne može primijeniti brojčano ograničenje, što je ispravljeno u PP 2.20.

> *PP 2.19: Laboratorijske vježbe moraju imati više od 15+1 računalnih mjesta.*
>
> *PP 2.20: Broj računalnih mjesta na laboratorijskim vježbama mora biti više od 15+1.*

U izražavanju poslovnih pravila važno je da bude točno poznato na što se koja od korištenih riječi odnosi, naročito kad takve riječi vremenski ograničavaju značenje pravila.

Iskazi poslovnih pravila po mogućnosti su prilično jednostavni. Konjunkciju ili disjunkciju ili bilo kakvo nabrajanje poželjno je napisati u obliku popisa uvjeta, zato što se time povećava mogućnost ponovnog korištenja pravila, a dobiva se i na lakoći izmjene tog pravila, npr. dodavanje ili isključivanje pojedinog uvjeta uvelike je olakšano (Business Rule Solutions, 2009, p. 14). Popisom uvjeta uklanja se i potreba za riječima koje se u svom osnovnom značenju odnose na dva ili više pojmova, poput oblika riječi „oboje“. Popis uvjeta uobičajeno je najaviti

oblikom riječi „navedeno“. PP 2.21 prikazuje nabrajanje uvjeta koje djeluje nezgrapno, dok PP 2.22 predstavlja poboljšanje u obliku popisa uvjeta.

> *PP 2.21: Student za prijavu na Natječaj mora ispuniti online prijavu i predati motivacijsko pismo.*
>
> *PP 2.22: Student za prijavu na Natječaj mora obaviti sve navedeno:*
>
> - *Ispuniti online prijavu.*
> - *Predati motivacijsko pismo.*

Izričito zapisane brojeve poželjno je izbjegavati (Business Rule Solutions, 2009, p. 16), naročito u slučaju kad su to brojevi koji su podložni promjenama ili se učestalo ponavljaju u većem broju izraza poslovnih pravila. Ukoliko se broj često ponavlja, korisno ga je definirati unutar zasebnog izraza poslovnog pravila, te se u ostalim izrazima pozivati na upravo taj definiran pojam. PP 2.23 sadrži numerički uvjet izražen izravno, iako je on podložan promjeni. Izraz poslovnog pravila PP 2.24 taj broj definira posebnim pojmom, čime se dobiva novi pojam, i pravilo više, koji se može koristiti u svim ostalim pravilima, gdje je isti potrebno primjenjiv. Tim je načinom olakšano i mijenjanje tog broja. Pravilo PP 2.25 ima isto značenje kao i PP 2.23, ali je izraženo na kvalitetniji način.

> *PP 2.23: Redovni student mora upisati više od 50 ECTS bodova godišnje.*
>
> *PP 2.24: Redovni broj ECTS bodova mora biti više od 50 ECTS bodova godišnje.*
>
> *PP 2.25: Redovni student mora upisati redovni broj ECTS bodova.*

Dakako, u situaciji koja je objašnjena gore, potrebno je s ostalim osobama poslovne organizacije dobro utvrditi novouvedene pojmove, poput onog *redovni broj ECTS bodova* u PP 2.24. Analogno rješenje predlaže se u situaciji kad je umjesto jednostavnog broja formula po kojoj se računa neka vrijednost.

Konačno, poželjno je da se izraz poslovnog pravila ne oslanja na neki proces, već da je ograničen stanjem, tj. da se poziva na stanje određenog koncepta, čime se poboljšava ponovna iskoristivost (Business Rule Solutions, 2009, pp. 19-20). Pritom pravilo određuje da stanja ne mogu istovremeno biti aktivna. PP 2.26 oslanja se na proces, dok je PP 2.27 definiran stanjima.

> *PP 2.26: Predmet nije moguće položiti ako je neocijenjen.*
>
> *PP 2.27: Položen predmet mora biti ocijenjen.*

2.5 Poslovni procesi?

Uzimajući u obzir činjenicu da su poslovni procesi kao koncept, u odnosu na poslovna pravila, znatno češće korišteni u opisu poslovanja i u opisu neke organizacije, bitno je ovdje naglasiti da poslovna pravila i poslovni procesi nisu sinonimi, već se uvelike razlikuju. Štoviše, razlika poslovnih pravila i poslovnih procesa vidljiva je već u samim temeljnim opisima istih: poslovni proces uvijek transformira nešto, dok poslovna pravila, sama po sebi, nikad ne dovode do transformacije, iako njihova primjena može dovesti do određene transformacije u drugom smislu, no to je nešto potpuno drugačije. (Ross, 2012) Poslovna pravila u kontekstu modela poslovnih procesa najčešće je moguće uočiti pri donošenju odluka, tj. u uvjetovanim tokovima,

gdje postoji uvjet tipa *ako* (eng. *if*). Upravo se u tome krije česta pogreška u modeliranju poslovnih procesa, gdje nastali model može biti preopsežan, suviše složen i nečitljiv. Ross predlaže da se u samom modelu poslovnog procesa, u slučaju uvjetovanog toka, ne prikazuju kriteriji po kojima se izvršava evaluacija uvjeta, već da se takvi kriteriji uvrste u skup poslovnih pravila. Također, Ross predlaže da se poslovna pravila, iako usko vezana za poslovni proces koji se modelira, ne prikazuju u modelu poslovnog procesa, kako bi se povećala preglednost, ali i zato što prikazivanje poslovnih pravila nije u skladu s osnovnom zadaćom modela poslovnog procesa – skicom načina obavljanja određenog posla. Konačno, poslovni procesi najčešće ne predstavljaju iznimno kompleksne strukture, već se složenost dobiva uputama o izvršavanju tog određenog procesa, koje je korisno isključiti iz modela poslovnog procesa i prikazati isključivo kao poslovna pravila.

3 Modeliranje poslovnih pravila

Prepoznavanje poslovnih pravila u poslovnom kontekstu, pa zatim njihovo zapisivanje i organizacija dugotrajan je posao. Stoga je poslovna pravila, kao rezultat tog procesa, iznimno korisno zapisati u digitalnom obliku. U tu se svrhu koriste mnoga rješenja, gdje, u nedostatku profesionalnosti ili stručnosti, poslovne organizacije pronalaze vlastita rješenja u obliku jednostavnih tabličnih zapisa ili tekstualnih dokumenata. Na organiziran je način poslovna pravila moguće formalizirati korištenjem različitih metoda modeliranja, od kojih će biti spomenut UML (eng. *Unified Modeling Language*, u slobodnom prijevodu standardizirani jezik za modeliranje) kao jezik koji je iznimno raširen među poslovnom populacijom te postavljen za svojevrsni standard u modeliranju poslovnih koncepata. Druga prikazana metoda modeliranja poslovnih pravila koristit će ORM (eng. *Object Role Modeling*) dijagram.

Formalizacija poslovnih pravila modelima koji koriste UML ili ORM dijagrame koristi se za računalni zapis koji je kvalitetniji i omogućava veću ponovnu iskoristivost, u odnosu na zapise pomoću jednostavnih tablica ili tekstualnih dokumenata. Također, takav formalizirani zapis omogućava bolju integraciju s računalnim sustavima i softverskim rješenjima koja poslovna pravila mogu dalje iskoristiti za npr. automatizaciju poslovanja. Osim toga, jasno je da rastuća organizacija ima koristi od strukturiranih i formaliziranih podataka. Bitno je napomenuti i da je modeliranje izraz koji dopušta rad na višoj razini apstrakcije te dizajn softvera prije same izrade koda, čime se umanjuje vjerojatnost neuspjeha softverskog projekta (npr. informacijski sustav).

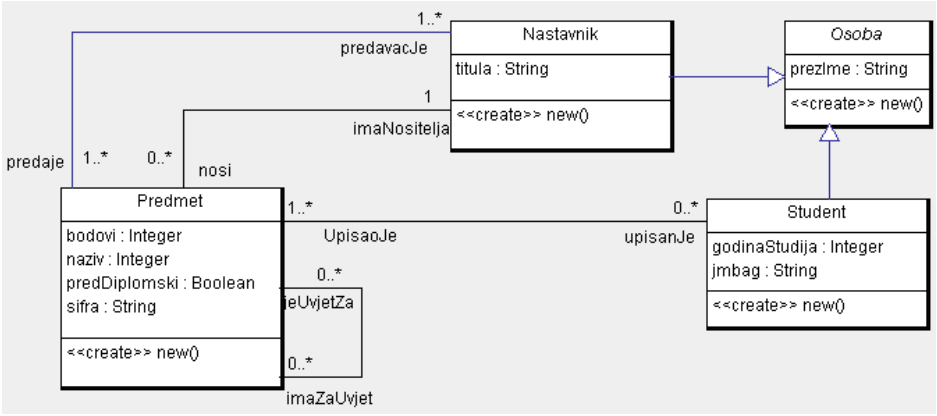
3.1 UML

UML je standard modeliranja koji je uvela organizacija *Object Management Group* (OMG), a koji „pomaže u specifikaciji, vizualizaciji i dokumentiranju modela softverskih sustava, uključujući njihovu strukturu i dizajn.“ (Object Management Group, 2012.) Može se koristiti za modeliranje gotovo bilo koje aplikacije, a temelji se na objektno orijentiranim konceptima poput klasa i operacija, dok neki alati omogućavaju i donekle uspješan prijevod modela u objektno orijentirane jezike i okruženja, poput C++ ili Java, čime se, kvalitetnim modelom, dobiva kostur za daljnju izradu aplikacije. Zahvaljujući mogućnosti korištenja još jednog OMG standarda za razmjenu XML metapodataka, XMI (eng. *XML Metadata Interchange*), UML model je moguće prebaciti iz jednog alata u drugi, bez gubitka podataka. Štoviše, UML je moguće koristiti za modeliranje i prikaz rezultata neovisno o metodologiji korištenoj za analizu ili dizajn.

U sklopu UML-a definirane su tri kategorije dijagrama (Object Management Group, 2012.): dijagrami strukture, dijagrami ponašanja i dijagrami interakcije. Svakoj od tih kategorija pripada jedan od trinaest tipova dijagrama. Za prikaz poslovnih pravila koriste se: dijagram klasa iz kategorije dijagrama strukture te dijagram stanja iz kategorije dijagrama ponašanja.

Za potpuni prikaz poslovnih pravila pomoću UML dijagrama bit će korišten i OCL (eng. *Object Constraint Language*), formalni jezik kojim se pobliže određuju koncepti UML dijagrama, što najčešće uključuje definiranje nepromjenjivih uvjeta koji moraju biti istiniti u svakom trenutku za sustav koji je modeliran. Bitno je napomenuti da prilikom procjene istinitosti OCL pravila ne dolazi do promjena unutar modeliranog sustava, tj. ona nemaju tzv. nuspojava. OCL se koristi za postavljanje funkcija koje prilikom izvođenja mijenjaju stanje sustava, zatim ograničenja koja su specifična za modeliranu aplikaciju, postavljanje upita nad UML modelom i slično.

Modeliranje poslovnih pravila pomoću UML-a i OCL-a bit će prikazano kroz *opensource* alat ArgoUML[6] koji omogućava izradu UML dijagrama te dodavanje ograničenja u obliku OCL izraza. Činjenica da je alat tek u razvojnoj inačici v0.34 objašnjava određena ograničenja na koja je moguće naići prilikom njegovog korištenja. Naime, OCL izrazi, po svojoj definiciji, omogućavaju referenciranje veza među klasama, kao i atributa povezanih klasa, kao što će biti prikazano kasnije. Na žalost, u trenutnoj verziji alata nije moguće OCL izraze koristiti na takav način, već su oni ograničeni na prostor klase nad kojom se uvode ograničenja te izravno njene atribute i funkcije. Unatoč tome, alat provjerava sintaksu OCL izraza i upozorava na pogreške, dok je moguće zadati da ignorira potencijalne pogreške u izrazu, a koje se odnose upravo na tip koji se naznačuje (atribut, funkcija, veza, …).



Slika 3.1: Konačni UML dijagram klasa primjera modeliranja poslovnih pravila pomoću UML-a. (izvor: izrada autora)

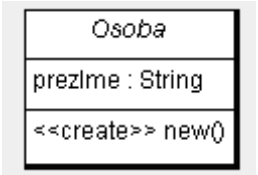
Kroz nekoliko primjera bit će prikazano kako je na temelju poslovnih pravila moguće dobiti neke dijelove modela sa Slika 3.1. Prvo poslovno pravilo izraženo je kao PP 3.1, a odnosi se na koncept `Osoba`. Nadalje, PP 3.2 daje osnovni opis koncepta `Student`, dok PP 3.3 postavlja temelje koncepta `Nastavnik`.

PP 3.1: Osoba mora imati ime i prezime.

PP 3.2: Student mora biti osoba.

PP 3.3: Nastavnik mora biti osoba.

Poslovnim pravilom PP 3.1 stvara se klasa `Osoba` atributa `prezIme`, kao na Slika 3.2. Svaka od klasa koje će ovdje biti spomenute ima funkciju `<<create>> new()` koja predstavlja konstruktor klase, a definiranje koje zahtijeva korišteni alat.



Slika 3.2: Apstraktna klasa *Osoba* s atributom *prezIme*.

[6] ArgoUML dostupan je na http://argouml.tigris.org

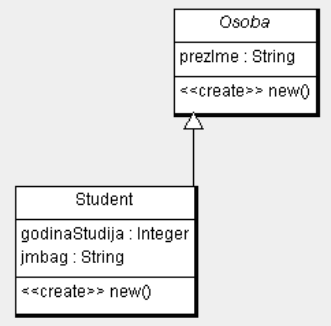
Poslovno pravilo PP 3.2 uvodi, u ovom primjeru, još nekorišten koncept `Student`, a `Student` i `Osoba` su u odnosu kao na Slika 3.3. Ovdje se uvode dodatna poslovna pravila, PP 3.4, PP 3.5 i PP 3.6.

PP 3.4: Student mora imati JMBAG.

PP 3.5: JMBAG mora jednoznačno identificirati akademskog građanina.

PP 3.6: Student mora biti na jednoj od sljedećih godina studija:

- *prva,*
- *druga,*
- *treća,*
- *četvrta.*



Slika 3.3: Klasa *Student* je podklasa klase *Osoba* i ima atribute *godinaStudija* i *jmbag.* (izvor: izrada autora)

Osim što je iz PP 3.4 jasno da student ima atribut `jmbag`, pravilom je određeno da vrijednost tog atributa mora uvijek biti definirana. Uz već prikazani UML dijagram, ovo ograničenje moguće je dodatno opisati OCL izrazom, kao u OCL 1.

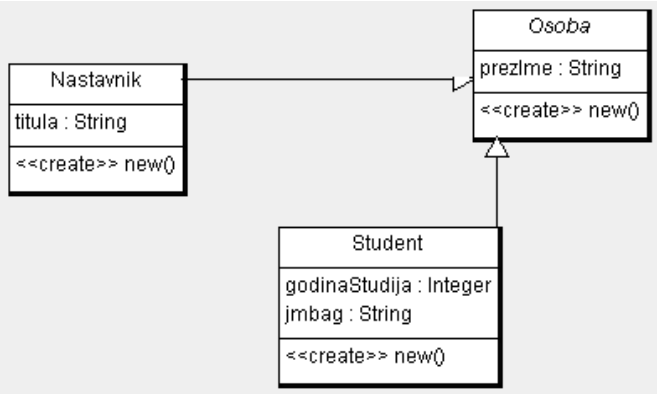
```
Context Student inv : self.jmbag <> ''
```

OCL 1: Atribut *jmbag* ne smije biti prazan.

Nadalje, poslovno pravilo PP 3.3 određuje koncept `Nastavnik` slično kao i PP 3.2 koncept `Student`, no dodano je ovdje još jedno pravilo, PP 3.7. Proširen model prikazan je na Slika 3.4.

PP 3.7: Nastavnik mora imati jednu od sljedećih titula:

- *asistent,*
- *viši asistent,*
- *znanstveni novak,*
- *docent,*
- *izvanredni profesor,*
- *redovni profesor,*
- *vanjski suradnik,*
- *profesor emeritus.*



Slika 3.4: Klasa *Nastavnik* je podklasa klase *Osoba* i ima atribut *titula.* (izvor: izrada autora)

Koncept predmet, prema početnom dijagramu, Slika 3.1, definiran je većim brojem poslovnih pravila. U nastavku će biti ispisano samo nekoliko pravila, koja su zanimljiva za razmatranje s obzirom na ograničenja koja postavljaju. Poslovna pravila PP 3.8 i PP 3.9 pobliže određuju atribute klase `Predmet`, dok PP 3.10, PP 3.11 i PP 3.12 pobliže određuju veze klase `Predmet` sa klasama `Nastavnik` i `Predmet`.

PP 3.8: Predmet mora imati naziv.

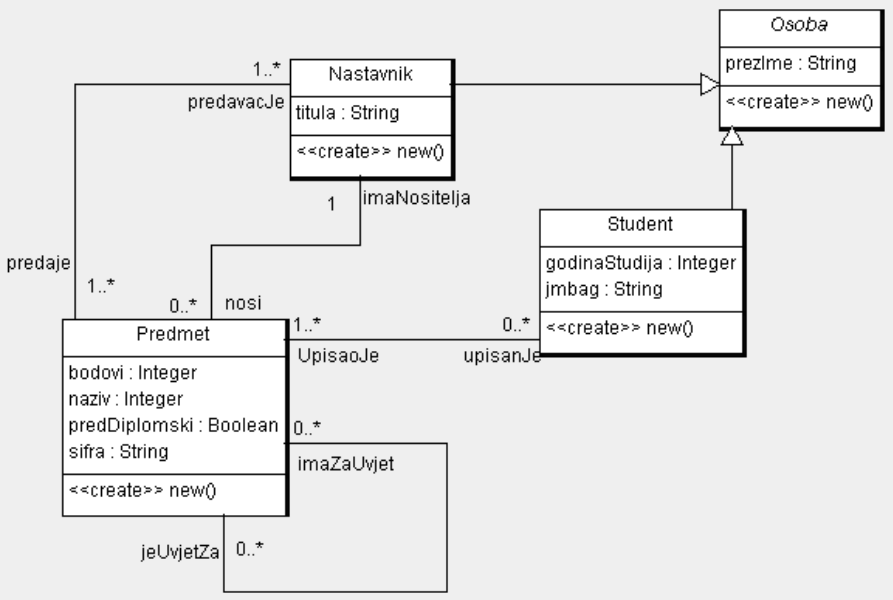
PP 3.9: Predmet mora imati broj ECTS bodova veći od 0.

PP 3.10: Predmet mora voditi jedan nastavnik nositelj predmeta.

PP 3.11: Predmet mora predavati jedan ili više nastavnika.

PP 3.12: Upis predmeta može biti uvjetovan drugim predmetima.

Pravilo PP 3.8 određuje da svaka instanca klase `Predmet` mora uvijek imati definiranu vrijednost atributa `naziv`. Nadalje, pravilo PP 3.9 ograničava vrijednost atributa `bodovi`, koji označava broj ECTS bodova koji odgovaraju opterećenju predmeta, na brojeve veće od 0. Pravilo PP 3.12 na prvi pogled izgleda dovoljno dobro izraženo, no lako je uočiti da ne osigurava potpunu informaciju: Što znači da je upis predmeta uvjetovan drugim predmetom ili predmetima? Potrebno li je te predmete položiti prije upisa promatranog predmeta ili je dovoljno samo prikupiti potpis iz tih predmeta koji predstavljaju uvjete? Ipak, odluka je da je ovo pravilo dovoljno informativno da određuje svojstvo uvjetovanja upisa predmeta, no početna ideja i dalje ostaje: svaki predmet ima posebno definirane preduvjete i njihovo stanje u trenutku upisa promatranog predmeta. Također, ovo pravilo koristi ključnu riječ „može“ umjesto riječi „mora“, što je moguće opravdati opcionalnošću, tj. činjenicom da promatrani predmet može, ali i ne mora imati uvjete. Ažurirani UML model prikazan je na Slika 3.5.



Slika 3.5: Konačni UML dijagram opisanog primjera. (izvor: izrada autora)

Dijagram na Slika 3.5 jasno prikazuje odnos klase `Predmet` i klase `Nastavnik`. Uzimajući u obzir kontekst veze koja opisuje nositeljstvo predmeta, iz dijagrama je vidljivo da se radi o vezi koju je moguće izraziti kako slijedi: svaki predmet ima točno jednog nositelja koji je nastavnik; svaki nastavnik može biti nositelj na više predmeta. Navedeni je odnos, sa strane klase `Predmet`, izražen pravilom PP 3.10. Ograničenje vidljivo samo u UML dijagramu nije dovoljno jasno izraženo. Naime, nije sigurno koliki je to točno broj „više“, koji označava broj predmeta kojima promatrani nastavnik može biti nositelj. Kako ne bi došlo do preopterećenja danog nastavnika, uvodi se dodatno poslovno pravilo. Takvo dodatno ograničenje, temeljeno na pravilu PP 3.13, može se u model unijeti korištenjem OCL izraza, kao OCL 2, koji se dodaje klasi `Nastavnik`.

PP 3.13: Nastavnik ne smije biti nositelj više od 3 predmeta.

```
Context Nastavnik inv : self.nosi -> size() < 4
```

OCL 2: Nastavnik može biti nositelj najviše 3 predmeta.

Pravilo PP 3.13 u skladu je sa svime prije navedenim, pa i s UML modelom, zato što postavlja gornje ograničenje, tj. najveći broj predmeta, dok ne postavlja donju granicu, dakle još je uvijek moguće da nastavnik nije nositelj niti jednog predmeta.

Pravilo PP 3.12 koje uređuje odnos uvjetovanog upisa, korisno je dodatno ograničiti, s obzirom na primijećen propust koji dopušta da bude određeno da predmet ima samog sebe kao uvjetovani predmet. U svrhu uklanjanja primijećenog problema dodaje se još jedno poslovno pravilo, PP 3.14 koje je u UML model moguće uvesti OCL izrazom OCL 3.

PP 3.14: Upis predmeta ne smije biti uvjetovan istim predmetom.

```
Context Predmet inv : self.imaZaUvjet -> reject(self)
```

OCL 3: Predmet ne može za uvjet imati samog sebe.

Na proteklih nekoliko stranica prikazan je moguć pristup modeliranju poslovnih pravila korištenjem jezika UML za iskazivanje modela i odnosa osnovnih koncepata, zajedno s njihovim atributima, dok je za pojašnjenje logike i dodatne potrebe izražavanja značenja poslovnih pravila korišten jezik OCL. Oba korištena jezika standardizirana su te je UML smatran za standard modeliranja poslovnih procesa u obujmu većem od onog koji je iskorišten u primjerima ovdje.

Prednost modeliranja korištenjem UML modela i OCL izraza je činjenica da su UML modeli iznimno dobro prihvaćeni u široj zajednici te nisu poznati isključivo populaciji informatičara, već su prisutni i u poslovnom svijetu[7]. OCL izrazi učinkovito nadopunjuju UML modele zbog svoje jednostavnosti pisanja, kao i čitanja, te niti oni ne predstavljaju opterećenje u komunikaciji informatike i poslovanja.

S obzirom da je ovdje prikazan primjer modeliranja jednostavnih poslovnih pravila, u svrhu kratkog prikaza metode, nije bilo moguće vidjeti nedostatke OCL izraza u izražavanju poslovnih pravila, koji ipak postoje. Izražavanje napredne logike poslovnih pravila nije moguće korištenjem OCL izraza ili je ono prilično ograničeno.

3.2 ORM

ORM (eng. *Object Role Modeling*, modeliranje uloga objekata) konceptualni je pristup modeliranju i upitima semantici informacija poslovnih domena, gdje sve činjenice i pravila mogu biti pretočeni u jezik koji razumiju i obični korisnici tih poslovnih domena (Halpin, 2009a). ORM je metoda dizajniranja modela baze podataka i provođenja upita nad njime, a sve na konceptualnoj razini, na kojoj je sve potrebno, cijela aplikacija, opisano korištenjem pojmova koji su lako shvatljivi korisniku, a ne suviše tehnički i povezani izravno uz implementaciju (Halpin, 1996).

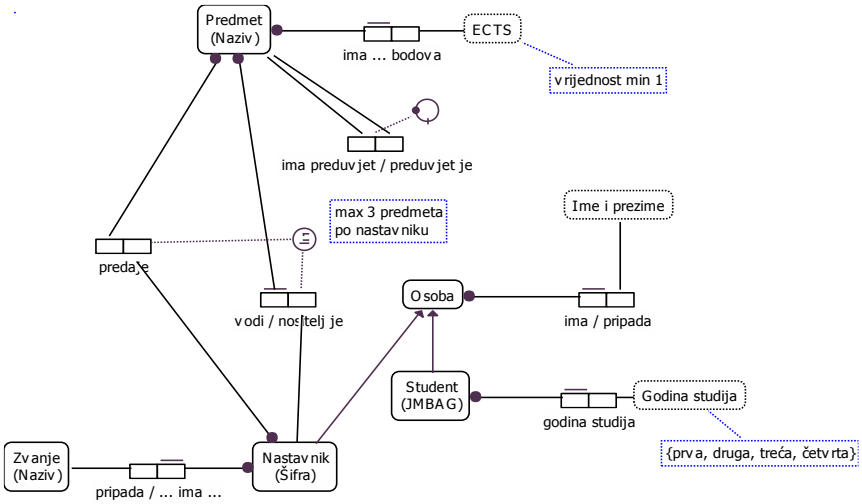
Naziv metode dolazi iz pogleda kojim se ista koristi, pri čemu je aplikacija skup objekata (entiteta ili vrijednosti) koji imaju određene uloge (u međusobnim odnosima). U ORM-u sve činjenice, sve izjave, iskazane su kao unarni, binarni ili veće arnosti[8], odnosi između entiteta. Zanimljivo je da ORM nudi grafički i tekstualni jezik za modeliranje informacija, no podržava i procedure za dizajn konceptualnih modela, transformacije između različitih konceptualnih prikaza itd. Objekti ORM modela nemaju atribute, čime se dobiva nekoliko prednosti u odnosu na ostale metode modeliranja (Halpin, 2009a):

- semantička stabilnost (smanjuje obuhvat potencijalne promjene pri potrebi zapisa o nekom atributu),
- prirodno izražavanje (sve činjenice i pravila mogu se ispisati na jednostavan način, korištenjem široko poznatih pojmova),
- populativnost (za svaki je odnos moguće zapisati primjere koji zadovoljavaju dana ograničenja),
- izbjegavanje praznih vrijednosti.

Kao što će biti prikazano kroz nekoliko primjera, ORM grafički jezik i notacija značajno su izražajniji u svrhu modeliranja od usporedivih UML ili ER dijagrama.

Konačni ORM dijagram za jednostavan primjer, sličan onom u prethodnom poglavlju izražen korištenjem UML-a, izgleda kako slijedi na Slika 3.6:

[7] http://www.uml.org/, pristupljeno 10.6.2013.
[8] Odnosi veće arnosti su odnosi s više povezanih entiteta (unarni s jednim, binarni s dva, ternarni s tri itd.)



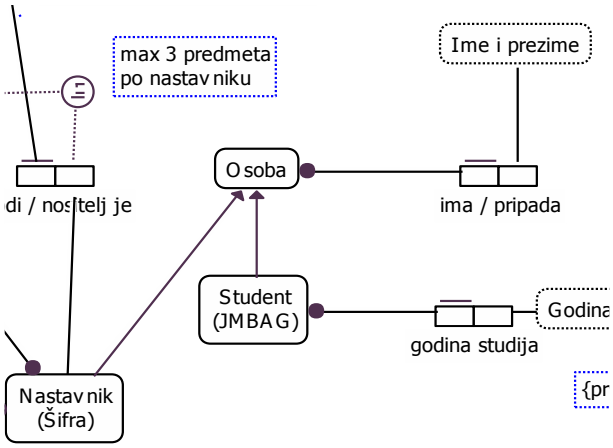
Slika 3.6: Konačni ORM dijagram primjera. (izvor: izrada autora)

U modelu na Slika 3.6 objekti su prikazani sa zaobljenim kutovima, dok su između njih uloge i odnosi između tih objekata. Pravila PP 3.15, PP 3.16 i PP 3.17 prikazana su u dijelu dijagrama kao na Slika 3.7.

PP 3.15: Osoba mora imati ime i prezime.

PP 3.16: Student mora biti osoba.

PP 3.17: Nastavnik mora biti osoba.



Slika 3.7: Osoba mora imati ime i prezime, a *Student* i *Nastavnik* su podklase klase *Osoba.* (izvor: izrada autora)

Na Slika 3.7 vidi se zapisana činjenica da svaka individua klase `Osoba` mora imati točno jednu vrijednost `Ime i prezime`. Nužnost se u ORM dijagramu označava točkom na kraju veze, dok je kardinalnost, tj. oblik veze N:1 izražena vodoravnom crtom nad lijevim dijelom veze između objekta `Osoba` i `Ime i prezime`. Nasljeđivanje, tj. odnos podklase, prikazan je vezom sa strelicom na kraju, pa se tako vidi da je `Nastavnik` podklasa klase `Osoba`, kao što je to i klasa `Student`. Time su zadovoljena pravila PP 3.15, PP 3.16 i PP 3.17.

Sljedeća ograničenja, koja se uvode poslovnim pravilima PP 3.18 i PP 3.19, služe jednoznačnom određivanju individua klase `Student`. Ograničenja uvode atribute koji obavezno

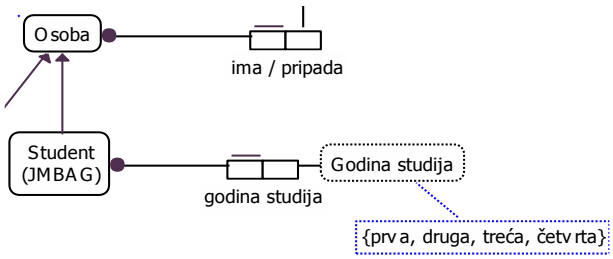
imaju vrijednost te su jedinstveni među svim individuama klase. ORM dijagram takvih ograničenja prikazan je na Slika 3.8.

PP 3.18: Student mora imati JMBAG.

PP 3.19: JMBAG mora jednoznačno identificirati akademskog građanina.

PP 3.20: Student mora biti na jednoj od sljedećih godina studija:

- *prva,*
- *druga,*
- *treća,*
- *četvrta.*



Slika 3.8: *JMBAG* jednoznačno određuje elemente klase *Student.* (izvor: izrada autora)

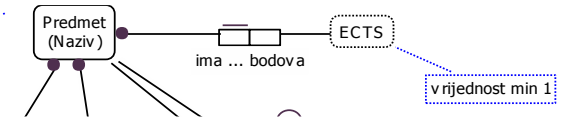
ORM maksimalno pojednostavnjuje određivanje identifikacijskog atributa, što je vidljivo upravo na klasi `Student` i atributu `JMBAG`: naziv atributa dovoljno je zapisati uz naziv klase i podrazumijeva se da dani atribut jednoznačno određuje individue klase te, kao takav, sam po sebi mora imati unesenu vrijednost.

Ograničenje postavljeno poslovnim pravilom PP 3.20 u trenutnoj je inačici alata ORMLite (0.13b u trenutku pisanja), nemoguće prikazati izravno glavnim elementima dijagrama, već za to treba poslužiti element tipa bilješka, u koji je, u ovom slučaju, upisano da vrijednost atributa `Godina studija` mora biti element skupa sljedećih vrijednosti: `prva`, `druga`, `treća`, `četvrta`.

Poslovna pravila PP 3.21 i PP 3.22 uvode osnovna ograničenja klase `Predmet`. Takva ograničenja prikazana su na Slika 3.9.

PP 3.21: Predmet mora imati naziv.

PP 3.22: Predmet mora imati broj ECTS bodova veći od 0.



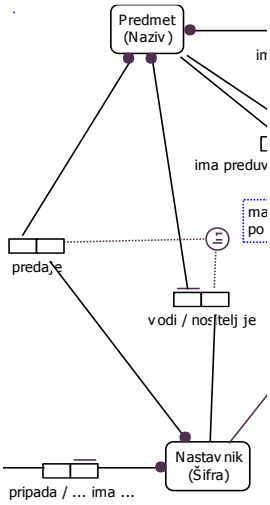
Slika 3.9: Predmet ima identificirajući *Naziv* i broj ECTS bodova veći od 0. (izvor: izrada autora)

Ponovno identificirajući atribut svoje mjesto nalazi odmah uz naziv klase, dok je atribut `ECTS`, koji označava broj ECTS bodova određenog predmeta, ograničen na tri načina: obavezno ima vrijednost, svaki predmet ima samo jednu `ECTS` vrijednost, i `ECTS` ne može imati vrijednost

manju od 1. Treće ograničenje uvedeno je korištenjem bilješke iz već navedenog razloga. Primjetan je na Slika 3.9 i način zapisivanja definicije nekog odnosa, u ovom slučaju „ima … bodova“, što omogućava programu automatsko sastavljanje značenja promatrane veze, gdje automatski na početak rečenice značenja ubacuje lijevi objekt, dok na mjesto trotočja ubacuje naziv drugog objekta, čime se dobiva zapis veze primjereniji krajnjem korisniku.

Jednostavna ograničenja nad vezama uvode poslovna pravila PP 3.23 i PP 3.24, koja su iskazana u ORM dijagramu na Slika 3.10.

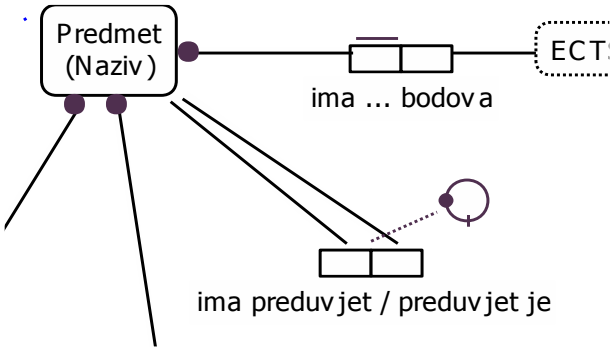
> *PP 3.23: Predmet mora voditi jedan nastavnik nositelj tog predmeta.*
>
> *PP 3.24: Predmet mora predavati jedan ili više nastavnika.*



Slika 3.10: Veze između klasa *Nastavnik* i *Predmet.* (izvor: izrada autora)

Prvo pravilo, PP 3.23, prikazano je zdesna i koristi već opisane elemente: točka na početku veze, koja označava nužnost i vodoravnu crtu iznad jednog dijela odnosa, što znači da svaki predmet ima točno jednog nastavnika koji ga vodi. Nastavnik ne mora biti nositelj niti jednog predmeta, a može i više njih. S lijeve strane prikazano je ograničenje drugog pravila, PP 3.24, koje uspostavlja krajnje jednostavan odnos, bez posebnih ograničenja, osim onog o nužnosti, gdje svaki element klase `Predmet` svakako mora biti povezan ovim odnosom sa barem jednim elementom klase `Nastavnik`, a vrijedi i obrnuto, da svaki nastavnik mora predavati barem jedan predmet. Novost u ovom dijagramu je simbol podskupa koji povezuje dva upravo opisana odnosa, a koji označava da par `Nastavnik`-`Predmet` u vezi koja opisuje nositeljstvo predmeta, mora biti podskup iz skupa parova `Nastavnik`-`Predmet` u odnosu koji opisuje predavane predmete. Osim podskupa, uloge mogu biti povezane još nekim odnosima (Halpin, 2009a).

Sljedeće poslovno pravilo, PP 3.25, uvodi opet novi element u ORM dijagram, a to je oznaka za ograničavanje prstenastog odnosa, tj. binarne uloge koja spaja objekt sam sa sobom, kao što je to prikazano na Slika 3.11.



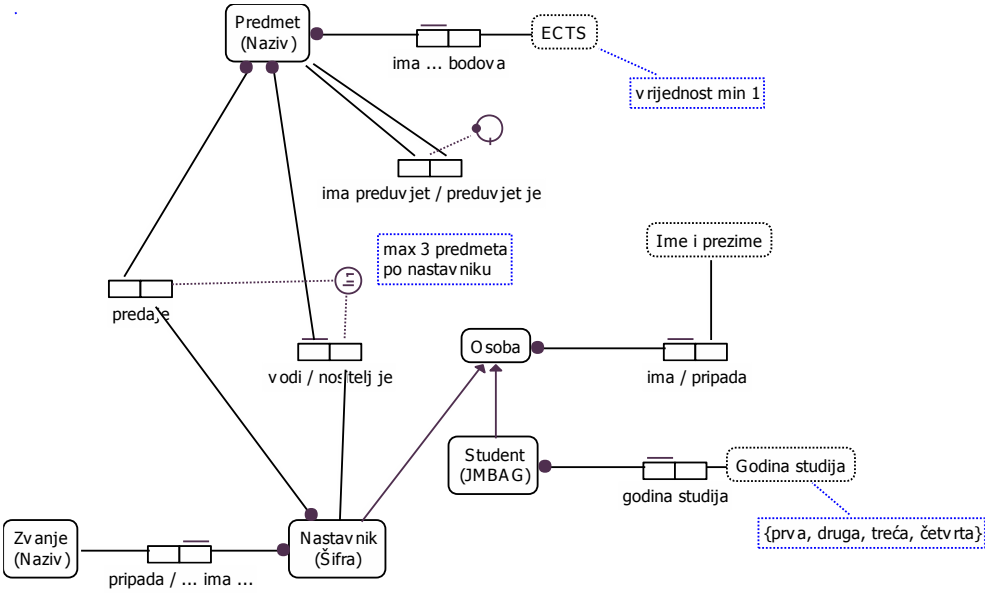
Slika 3.11: Predmet može imati drugi predmet kao preduvjet ili biti preduvjet drugom predmetu. (izvor: izrada autora)

PP 3.25: Upis predmeta može biti uvjetovan drugim predmetima.

Sama uloga označena je kao i ostale do sad promatrane u ORM dijagramu, no dodana je oznaka za prstenaste uloge. Ova oznaka opisuje prstenastu ulogu kao irefleksivnu, zbog činjenice da predmet ne može biti preduvjet sam sebi, niti može za preduvjet imati samog sebe, a sve u skladu sa poslovnim pravilom PP 3.26. Jasno, ne bi bilo pogrešno ulogu označiti i kao tranzitivnu, s obzirom da, ukoliko je jedan predmet preduvjet drugom, a taj drugi, pak, preduvjet trećem, moguće je zaključiti da nije moguće upisati treći predmet bez upisanog prvog predmeta (pritom se ovdje ne ulazi u sam postupak upisa predmeta s obzirom na njegove preduvjete).

PP 3.26: Upis predmeta ne smije biti uvjetovan istim predmetom.

Ograničenje poslovnog pravila izjavljenog u PP 3.27 u ORM dijagramu prikazano je kao i ograničenje pravila PP 3.22, korištenjem bilješke, kao što je vidljivo na Slika 3.12, koja ujedno prikazuje i konačni ORM model.



Slika 3.12: Konačni ORM model, opisan navedenim poslovnim pravilima. (izvor: izrada autora)

PP 3.27: Nastavnik ne smije biti nositelj više od 3 predmeta.

ORM model koristan je zbog jednostavnog korištenja i detaljnog prikaza odnosa objekata kroz uloge koje mogu biti binarne, ternarne, ali i više arnosti. Uloge je moguće opisati jednostavnim riječima te automatski izraditi njihove opise, kao i opise ostalih objekata.

Korisnost ORM dijagrama očituje se i u mogućnosti automatskog generiranja drugih, poznatijih, dijagrama, ali i koda koji može poslužiti za jednostavnije baratanje bazama podataka, s obzirom da je sve potrebne podatke moguće samo modelirati dijagramom, dok alat izradi kod potreban za prenašanje dijagrama u „stvarni“ svijet (Demey, et al., 2002).

3.3 Usporedba

Uspoređujući dva opisana načina modeliranja poslovnih pravila, autor smatra da je jasno vidljivo da je ORM više prilagođen modeliranju poslovnih pravila od UML-a. Zahvaljujući općenito većem području korištenja, UML je učestaliji odabir, zato što olakšava integraciju u ostala područja modeliranja. ORM omogućava detaljno određivanje ograničenja nad objektima domene koja se modelira. Osim navedenog, ORM omogućava generiranje ER (eng. *Entity Relationship*) dijagrama koji se uvelike koriste za prikazivanje modela baza podataka, kao i automatsko generiranje SQL koda koji odražava ORM model. Glavna razlika je i da UML za kvalitetni prikaz poslovnih pravila treba pomoć OCL-a, dok je ORM samodostatan.

4 Semantičko modeliranje

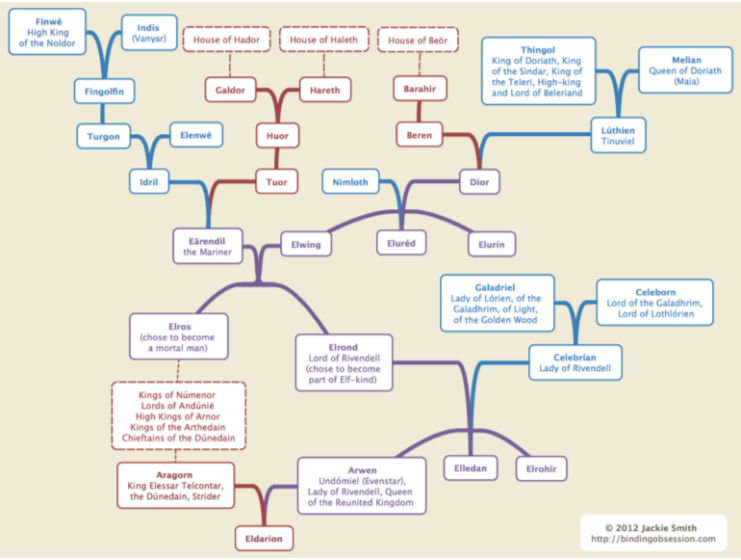
Uzimajući u obzir temu ovog rada te ranije obrađena poslovna pravila, slijedi obrada koncepta semantičkog modeliranja, kako bi konačno bilo moguće objediniti te dvije teme u semantičko modeliranje poslovnih pravila, temeljnu temu ovog rada.

Semantičko modeliranje jedan je od oblika modeliranja znanja kojim se dobiva semantički model ili model znanja koji opisuje značenje i vezu između različitih apstraktnih podataka i informacija te pomaže u razumijevanju međusobne povezanosti pojedinih elemenata modela.

4.1 Ontologija

Jedan od modela znanja je i ontologija, semantički model koji se sastoji od skupa koncepata i njihovih međusobnih veza. U širem smislu te riječi, ontologija pokriva istraživanja vezana uz postojanje ili stvarnost, a prvi put su je koristili 1613. godine filozofi Rudolf Göckel i Jakob Lorhard, kao pojam sastavljen od grčkih riječi *ontos* i *logos*, označavajući filozofsku granu metafizike koja traži odgovore na temeljna pitanja prirode bivanja (Halpin, 2009b). Pa ipak, pod pojmom ontologija ovdje će biti proučavan nešto drugačiji koncept. U informacijskim znanostima ontologija najčešće označava konceptualni model poslovne domene, pri čemu je model izražen na način koji olakšava dijeljenje informacija o danoj domeni, pritom koristeći određen skup standardiziranih elemenata. Jednostavan primjer ontologije, u obliku prikaza objekata u određenoj vezi, prikazan je na Slika 4.1, koja prikazuje rodoslovno stablo loze poluvilenjaka svijeta J.R.R. Tolkiena.

Potpuna ontologija sadrži deklaraciju odnosa među danim objektima, pravila koja te objekte određuju, ograničenja koja postavljaju mogućnosti te pravila koja omogućavaju zaključivanje novog o istim tim objektima. Takvu ontologiju nije moguće prikazati jednostavno kao povezane objekte, poput Slika 4.1, već bi njen potpun prikaz bio nešto kompleksniji. Ontologije uobičajeno sadrže objekte iz samo jedne poslovne domene te je interpretacija nekog objekta, tj. njegovo značenje, jasna svim korisnicima.



Slika 4.1: Poluvilenjaci svijeta J.R.R. Tolkiena i njihov rodbinski odnos; plavo su označeni čistokrvni vilenjaci, ljubičasto poluvilenjaci, a crveno ljudi. (izvor: preuzeto s http://bindingobsession.com/misc/)

Ontologija u cjelokupnosti svojih entiteta, veza, pravila i ograničenja sadrži zapisano znanje o određenoj realnoj ili poslovnoj domeni koju opisuje. Znanje sadržano u tom obliku čovjeku je teško čitljivo zbog načina zapisa, koji je za jednostavne primjere koji obuhvaćaju dvoznamenkast broj jednostavno opisanih entiteta donekle i shvatljiv, no postaje teško shvatljiv pri većem broju, naročito detaljno, opisanih entiteta. Nadalje, znanje koje ontologija sadrži najbolje je iskoristivo i najveću vrijednost ima, kao i znanje uopće, kada je isto moguće podijeliti sa ostalima. Upravo zbog te dvije činjenice ontologiju je najbolje prikazati uz pomoć danas sveprisutne informacijsko-komunikacijske tehnologije, koja omogućava iznimno jednostavno dijeljenje zapisa i znanja te rezoniranje nad tim znanjem. Takav smjer razvoja informacijskih znanosti doveo je do pojave semantičkog weba koji predstavlja unapređenje popularnog *World Wide Weba*.

4.2 Semantički web

Skup dokumenata dostupnih diljem Interneta, koje dijele razni korisnici, stvoreni su s ciljem prikazivanja samih tih dokumenata. Stvaranje zajedničkog smisla tih dijeljenih podataka te njihovo smisleno dijeljenje ili kombiniranje u smislene cjeline iznimno je otežano takvom neorganizacijom. Stoga su 2001. godine Sir Tim Berners-Lee, Jim Hendler i Ora Lassila uveli pojam semantičkog weba, kao tehnologije koja će promijeniti navedene probleme (Halpin, 2009b). Ideja semantičkog weba je obogaćivanje dokumenata dostupnih na webu globalnim označivačima (eng. *identifier*) i strukturom, primjerice, korištenjem stalnog označivača elementa[9] (URI, eng. *Uniform Resource Identifier*), ugrađenim oznakama (eng. *tag*), s ciljem omogućavanja ukazivanja prirode dokumenta i semantike njegovog dijeljenja automatiziranim agentima. Semantički web temelji se na nekoliko međusobno povezanih slojeva, kao što je to prikazano na Slika 4.2, gdje se jasno vidi najniži sloj te međuslojevi do konačne, korisniku dostupne, aplikacije semantičkog weba (Halpin, 2009b).

Gornji slojevi semantičkog weba, kao što se vidi na Slika 4.2 ovise o nižim slojevima, gdje je Unicode sa stalnim označivačima elemenata najdonji sloj, na koji se nadograđuje sloj XML zapisa i tipova podataka, koji je nadalje nadograđen RDF i RDFS slojem, koji je, konačno, obogaćen OWL slojem.

Slika 4.2: Temeljni slojevi semantičkog weba, Unicode i URIrefs na dnu te OWL koji se nadograđuje na RDF, RDFS te XML na vrhu. (prema: (Halpin, 2009))

Unicode je međunarodni standard za kodiranje pisanih znakova i teksta koji dodjeljuje jedinstven broj svakom znaku, neovisno o razvojnoj ili korisničkoj platformi, programu ili korištenom jeziku.[10] XML opisuje klasu podatkovnih objekata zvanih XML dokumenti i

[9] Iako podržavam obogaćivanje hrvatskog jezika i uvođenje riječi koje mogu kvalitetno zamijeniti tuđice, engleske izraze pokušat ću prevesti na hrvatski, no najčešće dalje koristiti engleske originale ili kratice, tamo gdje je to moguće.

[10] http://www.unicode.org/standard/WhatIsUnicode.html, pristupljeno 5.1.2013.

djelomično opisuje ponašanje računalnih programa koji ih opisuju.[11] RDF, RDFS i OWL će biti detaljnije opisani kako slijedi.

4.3 RDF

Sustav za opis elemenata, RDF[12] (eng. *Resource Description Framework*), predstavlja standard koji nadograđuje proširiv jezik za označavanje, XML (eng. *eXtensible Markup Language*), a služi označavanju metapodataka dokumenata dostupnih online.

RDF model moguće je zamisliti kao usmjeren graf koji se sastoji od imenovanih čvorova te imenovanih usmjerenih lukova, gdje svaki luk povezuje točno dva čvora (Halpin, 2009b). Svaki RDF čvor moguće je jedinstveno identificirati korištenjem stalnog označivača elementa (URI) koji može biti ili stalni pokazivač lokacije elementa, URL (eng. *Uniform Resource Locator*) ili stalno ime elementa, URN (eng. *Uniform Resource Name*). URL sadrži ime i lokaciju danog elementa na kojoj je isti moguće pronaći kao podatak Interneta, dok URN sadrži samo imenički prostor i naziv elementa: uri:ISBN:9536166011 pronalazi knjigu Hobit autora J.R.R. Tolkiena prevedenu na hrvatski jezik. Točnije, neki objekt se uz pomoć RDF-a identificira korištenjem URI reference (URIref) koja sadrži URI i lokalni identifikator. Prema tome, URIref http://www.w3.org/TR/rdf-mt/#urisandlit se sastoji od URI-ja http://www.w3.org/TR/rdf-mt/ i lokalnog identifikatora „urisandlit“ povezanog URI znakom „#“. Takav URIref prenesen u Internet-preglednik dovodi do dokumenta *RDF Semantics*, poglavlja *1.2 URI references, Resources and Literals*.

Kao što je već navedeno, RDF zapisi mogu se prikazati jednostavnim usmjerenim grafom imenovanih čvorova i imenovanih usmjerenih lukova. Slika 4.3 prikazuje takav usmjereni graf koji objašnjava da nakladnik knjige ISBN: 978-0547928227 ima naziv „Mariner Books“.

Slika 4.3: RDF model prikazan kao jednostavan imenovan usmjeren graf. (prema: (Halpin, 2009b))

RDF model na Slika 4.3 pokazuje čvor imenovan `urn:isbn:978-0547928227` kao neku određenu knjigu, dok čvor imenovan `http://www.pr.net/Nakladnici/info#MarBks` sadrži adresu koja identificira nekog točno određenog nakladnika, iz čega je moguće zaključiti da ova dva čvora predstavljaju instance svojih tipova, ili članove ekstenzije nekih koncepata. Čvorovi imenovani `http://www.pr.net/koncept#knjiga` i `http://www.pr.net/koncept#nakladnik`

[11] http://www.w3.org/TR/2008/REC-xml-20081126, pristupljeno 5.1.2013.
[12] Potpune i najdetaljnije informacije: http://www.w3.org/RDF/

predstavljaju tipove `Knjiga` i `Nakladnik`, tj. koncepte čijih su ekstenzija članovi prethodno promatrani elementi. Bitno je ovdje primijetiti da se u RDF modelu prikazom, niti posebnim ophođenjem, ne razlikuju tipovi objekata (npr. `Knjiga`) i instance tih tipova (npr. `urn:isbn:978-0547928227`).

Korisno je ovdje napomenuti da se RDF zapisi, pa tako i zapisi RDF modela, uvijek sastoje od tri elementa: subjekta, predikata i objekta (Halpin, 2009b). Slika 4.3 prikazuje točno četiri takve RDF trojke, gdje čvorovi predstavljaju subjekte i objekte, dok su lukovi predikati. Predikat je u RDF trojki obično iskazan glagolom, no ne sadrži svojstvo predikata iz gramatike, već dodjeljuje neku vrijednost prikazanu objektom svojstvu subjekta. Sva su tri elementa ovakve veze identificirana pomoću URIrefa. Objekt u RDF trojki može biti neki element ili literal, vrijednost određenog tipa poput cjelobrojnog, niza znakova ili nedefiniranog.

Prema tome, luk imenovan URIrefom `http://www.pr.net/koncept#imaNakladnika` čita se kao predikat značenja „ima nakladnika“. Analogno tome, luk `http://www.pr.net/koncept#naziv`, se čita kao predikat „naziva se“ ili „ima naziv“. Zanimljivo je napomenuti da u ontologiji, pa tako niti u RDF modelu, ne postoji jednoznačnost imena. Drugim riječima, nije nužno istinito da jedan naziv uvijek ukazuje na identičan objekt, pa je moguće da se dogodi situacija gdje isti niz znakova, npr. „vodi“, ne označava uvijek isti predikat: u jednom slučaju moguće je sastaviti trojku koja znači da `Voditelj vodi` neki `Projekt`, dok je korištenjem drugačijeg značenja moguće postaviti trojku koja znači da u utrci na 100 metara `vodi` Usain Bolt. Upravo s ciljem izbjegavanja takvih dvosmislenosti, u RDF modelu se predikati umjesto samo svojim skraćenim nazivom (npr. `imaNakladnika`) označavaju duljom verzijom naziva (npr. `http://www.pr.net/koncept#imaNakladnika`).

Do sad ponuđenim objašnjenjima elemenata na Slika 4.3 primjetno je da postoji nekoliko vrsta veza između elemenata RDF modela (Halpin, 2009b): veza između instanci, veza između instance i tipa te do sad nenavedena, veza između tipova. Veza između instanci najuobičajenija je veza koja prikazuje odnos između dviju instanci ili instance i literala te pridružuje određenu vrijednost danom svojstvu subjekta RDF trojke, npr. `http://www.pr.net/Nakladnici/info#MarBks http://www.pr.net/koncept#naziv „Mariner Books“` u značenju „Nakladnik šifre MarBks ima naziv Mariner Books.“ Veza između instance i tipa ukazuje da je subjekt instanca objekta, tj. da predstavlja element ekstenzije danog koncepta, npr. `urn:isbn:978-0547928227 http://www.w3.org/1999/02/22-rdf-syntax-ns#type http://www.pr.net/koncept#knjiga` u značenju „Element šifre urn:isbn:978-0547928227 je tipa knjiga.“ Treća vrsta veza je veza između tipova, odnosno elemenata koji označavaju koncepte, a najčešće se koristi za uspostavljanje odnosa između tipova, poput određivanja nadtipova i podtipova. Sve tri vrste veza RDF tretira na jednak način.

U proteklim je primjerima lako primijetiti da je zapis RDF trojki, prvenstveno zbog duljine URIrefa, nepraktičan i teško čitljiv. Ovaj problem rješava se pomoću određivanja imeničkog prostora. Ranije je spomenuto da se s ciljem izbjegavanja sinonima svi entiteti RDF modela referenciraju kroz svoje cjelokupno ime, tj. potpun URIref. Imenički prostor predstavlja skup koncepata, određenu domenu, kojoj entiteti pripadaju. Prema tome, predikate korištene u Slika 4.3 skraćeno je moguće zapisati korištenjem imeničkog prostora, koji se definira pri početku koda. S obzirom da je XML zapis, koji predstavlja formalnu strukturu RDF zapisa i OWL ontologija, prilično nečitljiv u obliku koji treba za prikaz korištenih primjera, ovdje će biti korištena jedna od alternativnih sintaksi, Manchester sintaksa, koja je, prema (Halpin, 2010b), uz Turtle sintaksu, iznimno lagana za korištenje i čitanje, čak i za osobe koje nisu dovoljno upoznate s tematikom.

Imenički prostor (eng. *namespace*) definira se korištenjem ključne riječi `Prefix` (Halpin, 2010b), kao što je prikazano ovdje:

```
Prefix: koncepti: <http://www.pr.net/koncept#>
```

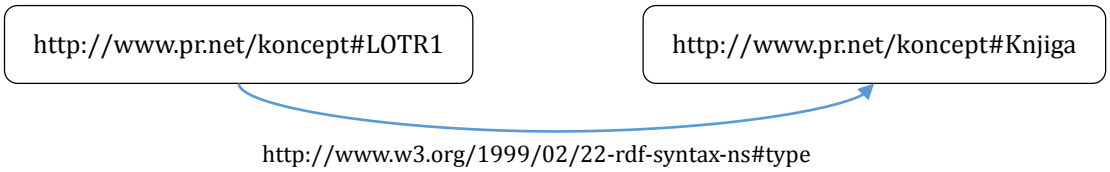
Kôd 4.1: Manchester zapis koji definira imenički prostor *koncepti*:

U Kôd 4.1 naveden je korisnički imenički prostor, no neki imenički prostori prethodno su definirani, pa ih nije potrebno posebno navoditi. Primjeri takvih imeničkih prostora navedeni su u Kôd 4.2 kako slijedi.

```
Prefix: rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
Prefix: owl: <http://www.w3.org/2002/07/owl#>
```

Kôd 4.2: Manchester zapis definiranja imeničkih prostora *rdf*: i *owl*:

Korištenjem navedenog o imeničkim prostorima, RDF trojku prikazanu u Slika 4.4 moguće je zapisati korištenjem imeničkog prostora kako je to učinjeno u Kôd 4.3.



Slika 4.4: RDF trojka značenja „LOTR1 je instanca tipa knjiga.“

```
Prefix: pr: <http://www.pr.net/koncept#>
Individual: LOTR1
    Types: Knjiga
```

Kôd 4.3: Manchester zapis RDF trojke iz Slika 4.4.

RDF trojka opisana u Kôd 4.3 opisuje činjenicu da je objekt identificiran kao `LOTR1` član ekstenzije koncepta `Knjiga`, tj. da je instanca tipa `Knjiga`. U zapisu u Kôd 4.3 izostavljeno je izravno navođenje predikata, sastavnog dijela RDF trojke, uz subjekt i objekt. Neizravno je moguće predikat pročitati iz ključne riječi `Types:` koja ima značenje koje se skriva u ranije korištenom konceptu imenovanom URIrefom `http://www.w3.org/1999/02/22-rdf-syntax-ns#type`.

4.4 RDFS

RDFS (eng. *RDF Schema*) izravno se nadovezuje na RDF te ga proširuje podrškom za klase i svojstvo podklase, čime se olakšava i dodatno formalizira određivanje odnosa između tipova, odnosno klasa (Halpin, 2009c).

Nezaobilazno je primijetiti da je ranije korišteno svojstvo instanciranja i pripadanja nekom tipu (jasno je to prikazano na Slika 4.4 i Kôd 4.3), iako neproširen RDF ne podržava u potpunosti to svojstvo. RDF omogućava korisniku određivanje tipa određenog objekta, tj. kojem tipu određeni entitet pripada. Svojstvo određivanja podklasa i pripadanja klasama, koje uvodi RDFS, snažnije definira odnos instance i klase te podklase i nadklase.

Novouvedenim svojstvom lakše je odrediti vezu dvaju koncepata te je dobivena mogućnost određivanja nadklase neke klase, tj. koncepta koji je obuhvaćen nekim drugim konceptom, kao što je to vidljivo u Kôd 4.4 koji slijedi.

```
Prefix: pr: http://www.pr.net/koncept#
Class: Tiskovina
Class: Knjiga
    SubClassOf: Tiskovina
```

Kôd 4.4: Manchester zapis dviju klasa i njihovog međusobnog odnosa.

U Kôd 4.4 jasno je vidljivo da je klasa `Knjiga` podklasa klase `Tiskovina`, što znači da nasljeđuje njena osnovna svojstva te da svaka instanca klase `Knjiga` sadrži svojstva kojima pripada i klasi `Tiskovina`. Veza tipiziranja, koja je poznata RDF zapisima, kao što je to prikazano u Kôd 4.3, ne uključuje upravo navedena svojstva, već ukazuje isključivo na povezanost objekta i tipa. Imenički je prostor ovdje ponovno definiran, no s obzirom na prirodu Manchester sintakse, nigdje nije prikazan ili izravno korišten. Standardno je ponašanje programa za obradu OWL ili RDFS zapisa da, u slučaju nenavođenja imeničkog prostora, podrazumijeva se da je korišten imenički prostor upravo obrađivanog dokumenta, čime se većinom uklanjaju eventualne sintaktičke pogreške. U daljnjim primjerima stoga neće biti naveden imenički prostor, već se on podrazumijeva.

Svojstvo podklase u RDFS-u osim općenito uvođenja mogućnosti određivanja podklase i nadklase, subjekt i objekt automatski označava klasama, što nije slučaj u svojstvu tipiziranja kod RDF-a, gdje je automatski moguće zaključiti samo da je objekt klasa. Odnos klase i nadklase, kao i klase i podklase, tranzitivno je svojstvo, što omogućava zaključivanje na razini višoj od one prilikom korištenja RDF-a te je tako moguće iz zapisa prikazanog u Kôd 4.4 zaključiti sadržaj zapisan u Kôd 4.5.

```
Class: Knjiga
    SubClassOf: Tiskovina
Class: Tiskovina
    SubClassOf: Medij
```

Kôd 4.5: Manchester zapis koji opisuje klasu *Knjiga* kao podklasu klase *Tiskovina* koja je, pak, podklasa klase *Medij*.

```
Class: Knjiga
    SubClassOf: Medij
```

Kôd 4.6: Manchester zapis klase *Knjiga* kao podklase klase *Medij*.

Tranzitivnost omogućava zaključivanje kao što je to prikazano kroz Kôd 4.5 i Kôd 4.6, tj. da iz odnosa gdje je klasa `Knjiga` podklasa klase `Tiskovina`, a klasa `Tiskovina` podklasa klase `Medij`, bude izvučen zaključak da je klasa `Knjiga` podklasa klase `Medij`. Osim navedenog, svojstvo podklase iz RDFS-a je i refleksivno, time omogućavajući da bilo koja klasa bude podklasa same sebe.

RDFS nadograđuje RDF i mogućnošću točnijeg određivanja svojstava, tj. predikata u RDF trojkama (Halpin, 2009c). Svojstvu je moguće odrediti domenu i doseg koji opisuju koji entiteti mogu biti subjekt, a koji objekt danog svojstva, time ograničavajući svojstvo, kao što je zapisano u Kôd 4.7.

```
ObjectProperty: autorJe
    Domain: Knjiga
    Range: Autor
```

Kôd 4.7: Manchester zapis ograničenja svojstva *autorJe* na domenu *Knjiga* i doseg *Autor.*

Kao što se vidi u Kôd 4.7, svojstvo naziva `autorJe` može u trojki sudjelovati kao predikat isključivo u kombinaciji u kojoj je subjekt element klase `Knjiga`, a objekt element klase `Autor`, čime je dano svojstvo ograničeno opsegom svog mogućeg korištenja. Domena nekog svojstva ne mora biti nužno ograničena na isključivo jednu klasu, kao niti doseg tog istog svojstva, već je moguće kao ograničenje uvesti nekoliko klasa, pri čemu subjekt svojstva domene ograničene većim brojem klasa (najmanje dvije), mora biti element presjeka tih klasa; isto vrijedi i za objekt svojstva čiji je doseg ograničen većim brojem klasa.

Slično kao i prilikom rada s klasama, svojstva mogu biti u hijerarhijskom poretku, čime nastaju nadsvojstva i podsvojstva (Halpin, 2010a). Analogno nadklasama i podklasama, dano svojstvo nasljeđuje karakteristike svog nadsvojstva te je moguće zaključiti da, ukoliko je jedan element svojstvom povezan s drugim elementom, tada su ta dva elementa povezana i nadsvojstvom danog svojstva. Podsvojstvo se definira kao u Kôd 4.8.

```
ObjectProperty: jeSastavniDioZa
    SubPropertyOf: jeDioZa
```

Kôd 4.8: Manchester zapis svojstva *jeSastavniDioZa* kao podsvojstva od *jeDioZa.*

Kôd 4.8 prikazuje kako odrediti nadsvojstvo danog svojstva korištenjem ključne riječi `SubProperyOf`. Definiranjem nadsvojstva moguće je od izraza kao u Kôd 4.9 zaključiti izraz prikazan u Kôd 4.10, što bez zapisa u Kôd 4.8 ne bi bilo moguće.

```
Class: Procesor
    SubClassOf: jeSastavniDioZa some Računalo
```

Kôd 4.9: Manchester zapis značenja „Neki Procesor sastavni je dio nekog Računala.“

```
Class: Procesor
    SubClassOf: jeDioZa some Računalo
```

Kôd 4.10: Manchester zapis značenja „Neki Procesor dio je nekog Računala.“

Posljednja dva primjera koriste ključnu riječ `some`, koja označava korištenje egzistencijalnog kvantifikatora, a uobičajeno je da se čita kao „neki“. Prema tome, formalno pročitano značenje posljednjeg primjera (Kôd 4.10) formulirano je kako slijedi: „Postoji neki objekt koji je Procesor i koji je dio nekog Računala.“

RDFS proširio je RDF mogućnostima određivanja nad- i podklasa, kao i nad- i podsvojstava, no moguće je primijetiti da samo ta svojstva ne pružaju dovoljnu izražajnost potrebnu za kvalitetan opis ontologije. Upravo iz tog razloga nastao je jezik za web-ontologije, OWL.

4.5 OWL

OWL, jezik za web-ontologije (eng. *Web Ontology Language*), koristi se kada je potrebno da informacije sadržane u danom dokumentu budu prilagođene za računalnu obradu, a ne služe isključivo za prikaz ljudskim korisnicima (W3C, 2004). Ontologija, koncept koji je ranije u radu uveden, sastoji se od značenja pojmova i prikaza njihovih međusobnih veza, što su mogućnosti koje uvodi OWL kao nadogradnju na RDFS. Baš kao što je to prikazano na

slojevima semantičkog weba, OWL je sljedeća stepenica prema semantičkim aplikacijama, koja sadrži XML kao standard zapisa, RDF i RDFS sa određenim jednostavnim mogućnostima, te sve navedeno proširuje. Upravo zbog svih značajki koje donosi, organizacija W3C (eng. *World Wide Web Consortium*), potpuno je podržala OWL kao jezik za razvoj semantičkog weba i ontologija.

OWL se u svojoj suštini sastoji od tri podjezika koji se međusobno razlikuju u razini izražajnosti te mogućnostima zaključivanja, kako slijedi (W3C, 2004; Halpin, 2010a):

- OWL Lite predstavlja pojednostavnjenu inačicu jezika, namijenjenu prvenstveno korisnicima kojima je potrebna klasifikacijska hijerarhija te jednostavna ograničenja.
- OWL DL (DL dolazi od deskriptivnih logika) koristan je korisnicima koji žele iznimnu izražajnost, uz što im je važna i potpunost (sva zaključivanja će se provesti) te odlučivost (sve računanje završit će u konačnom vremenu).
- OWL Full (potpuni) koji je namijenjen korisnicima kojima je važna najveća moguća izražajnost te sintaktička sloboda na razini RDF-a, no bez osiguravanja izračuna.

Navedeni podjezici OWL-a povezani su na sljedeći način (W3C, 2004):

- Svaka pravilna[13] OWL Lite ontologija je pravilna OWL DL ontologija.
- Svaka pravilna OWL DL ontologija je pravilna OWL Full ontologija.
- Svaki ispravan[14] OWL Lite zaključak ispravan je OWL DL zaključak.
- Svaki ispravan OWL DL zaključak ispravan je OWL Full zaključak.

Uz navedene poveznice između podjezika OWL-a, korisno je napomenuti da je isključivo OWL Full potpuno proširenje RDF-a, dok su OWL Lite ili OWL DL proširenja jednog ograničenog dijela RDF-a, pa je svaki OWL zapis RDF zapis, no RDF zapis je isključivo OWL Full zapis, zato što može sadržavati elemente koje OWL Lite ili OWL DL ne podržavaju. (W3C, 2004)

Kao dodatno proširenje OWL jezika osmišljen je OWL2, koji pruža nove mogućnosti izražavanja i finog podešavanja, poput ključeva, niza svojstava, asimetričnih, refleksivnih i odvojenih svojstava i još neka. (W3C, 2012.)

OWL u svrhu rezoniranja koristi pretpostavku otvorenog svijeta (eng. *Open World Assumption*, OWA) što onemogućava zaključivanje o nezapisanim činjenicama (Halpin, 2010a). Drugim riječima, sve zapisano može se ocijeniti kao istina ili laž, no nije moguće prosuditi o informacijama i znanju koje nije izričito zapisano.

Kontrast pretpostavci otvorenog svijeta je pretpostavka zatvorenog svijeta, gdje se rezoniranje odvija isključivo temeljem zapisanih podataka, što znači da je u danom svijetu, tj. u danoj domeni, poznato samo ono što je zapisano i samo se zapisano koristi u rezoniranju.

4.5.1 Kardinalnost

Prvo od zanimljivih proširenja OWL-a u odnosu na RDFS je mogućnost određivanja nužnosti svojstava te određivanje kardinalnosti svojstava (n:1, 1:1, 1:n), čime se olakšava uvođenje dodatnih ograničenja na svojstva te povećana izražajnost (Halpin, 2010a).

Standardna kardinalnost svojstva je 0, što znači da takva veza, dano svojstvo, nije nužno, tj. da je opcionalno i da ne postoji ograničenje koje zahtijeva da dani predikat bude povezan sa subjektom ili objektom. OWL omogućava postavljanje najvećeg ili najmanjeg broja sudionika

[13] eng. *legal*
[14] eng. *valid*

u svojstvu korištenjem ključnih riječi `maxCardinality` i `minCardinality`, no ukoliko su ta dva broja jednaka, dakle ukoliko se očekuje točan broj, umjesto navedenog, koristi se ključna riječ `cardinality`. Postavljanjem `minCardinality` na neki prirodni broj, kao što je to u Kôd 4.11, određeno je da dani predikat povezuje najmanje 1 objekt sa subjektom tog predikata[15].

```
Class: Knjiga
    SubClassOf: imaAutora min 1
```

Kôd 4.11: Manchester zapis ograničenja najmanje kardinalnosti svojstva *imaAutora* na 1.

Drugim riječima, svojstvo `imaAutora` nužno jedan subjekt povezuje sa najmanje jednim objektom. Zadana vrijednost svojstva `maxCardinality` je nepostojeća, tj. moguće je predikatom povezati jedan subjekt sa neograničenim brojem objekata. Uvođenjem ograničenja `maxCardinality` ograničava se gornja granica, najveći broj takvih objekata. Predikat u Kôd 4.12 ograničen je na povezivanje najviše 15 i najmanje 0 objekata sa danim subjektom.

```
Class: Igrač
    SubClassOf: brojPokušaja max 15, brojPokušaja min 0
```

Kôd 4.12: Manchester zapis ograničenja najveće i najmanje kardinalnosti svojstva *brojPokušaja* na 15, odnosno 1.

Kôd 4.12 opisuje predikat koji dani subjekt ne povezuje nužno sa objektima, no ukoliko veze postoje, tada ih može biti najviše 15. U slučaju kada predikat povezuje subjekt sa točno određenim brojem objekata, koristi se ključna riječ `cardinality`, kao što je to prikazano u Kôd 4.13.

```
Class: Osoba
    SubClassOf: imaSpol exactly 1 and imaSpol only {žensko, muško}
    SubClassOf: imaOIB exactly 1
```

Kôd 4.13: Manchester zapis kojim je određeno da svaka individua klase *Osoba* ima točno jedan spol pridružen svojstvom *imaSpol* i točno jednu vrijednost pridruženu svojstvom *imaOIB*.

Svojstvo opisano u Kôd 4.13 subjekt povezuje s točno jednim objektom, u ovom slučaju literalom. Dakle, svakom subjektu ovim je predikatom pridružen točno jedan objekt.

4.5.2 Sinonimi

Suprotno ranije navedenom, OWL ne koristi URI-je, već su entiteti identificirani internacionaliziranim označivačem elemenata, IRI-jem (eng. *Internationalized Resource Identifier*), što omogućava da jednom entitetu budu dodijeljeni različiti IRI-ji (Halpin, 2010b). Pa ipak, OWL prepoznaje kao iste individue samo one koje su eksplicitno navedene kao takve ključnom riječi `SameAs`, kao što je prikazano u Kôd 4.14.

[15] Ovdje valja napomenuti da se OWL svojstvo `maxCardinality` u Manchester sintaksi označava ključnom riječi `max`, svojstvo `minCardinality` ključnom riječi `min`, a svojstvo `cardinality` ključnom riječi `exactly`. (Drummond, 2009.)

```
Individual: Gandalf
    Types: Istari
    SameAs: Mithrandir, Olórin
    DifferentFrom: Curunír
```

Kôd 4.14: Manchester zapis značenja da je individua *Gandalf* isto što i individua *Mihtrandir* i individua *Olórin.*

Kôd 4.14 govori da je jedna individua imena Gandalf isto što i individua imena Mithrandir, no različita je od individue imena Curunír, što je opisano ključnom riječi `DifferentFrom`. Slično navedenom vrijedi i za klase, no koriste se drugačije ključne riječi: `EquivalentTo` i `DisjointWith`, kao u Kôd 4.15 i Kôd 4.16.

```
Class: Auto
    EquivalentTo: Automobil
```

Kôd 4.15: Manchester zapis klase *Auto* koja je jednaka klasi *Automobil.*

```
Class: Muškarac
    DisjointWith: Žena
```

Kôd 4.16: Manchester zapis klase *Muškarac* koja je odvojena od klase *Žena.*

Kôd 4.15 prikazuje klasu i klasu kojoj je dana klasa ekvivalentna, što znači da su individue jedne klase individue i druge klase, tj. da obje klase opisuju isti koncept. Suprotno tome, Kôd 4.16 prikazuje klasu koja je disjunktna klasi `Žena`, što znači da te dvije klase nemaju zajedničkih individua, no ne ograničava da individua bude element neke treće klase. Disjunktnost klasa može se definirati i izvan određivanja klase, ključnom riječi `DisjointClasses`, kao u Kôd 4.17.

```
Class: Muškarac
Class: Žena
DisjointClasses: Muškarac, Žena
```

Kôd 4.17: Manchester zapis dvije klase i odnosa disjunktnosti među njima.

4.5.3 Svojstva

Objektna svojstva predikati su koji entitete povezuju s entitetima (individue, klase, svojstva ili tip podataka). Podatkovna svojstva predikati su koji entitetu pridružuju literal (podatkovna vrijednost). OWL sadrži nekoliko unaprijed definiranih svojstava (Halpin, 2010c): `owl:topObjectProperty` koje je nadsvojstvo svih objektnih svojstava, ili klasa svih objektnih svojstava, `owl:topDataProperty` koje je klasa svih podatkovnih svojstava, `owl:bottomObjectProperty` i `owl:bottomDataProperty` koja predstavljaju prazno objektno, tj. podatkovno svojstvo. Posebnu vrstu svojstava OWL dodaje u vidu zabilješki.

Poznato je da su svojstva predikati koji, usmjereno, povezuju subjekt sa objektima. No, za neka svojstva postoje svojstva koja idu u suprotnom smjeru, kao što je prikazano u Kôd 4.18, gdje su definirana dva objektna svojstva: `jeZavršio` i `bivšiJeStudent`.

```
ObjectProperty: jeZavršio
    Domain: Osoba
    Range: Fakultet
ObjectProperty: bivšiJeStudent
```

```
    Domain: Fakultet
    Range: Osoba
```

Kôd 4.18: Manchester zapis dvaju svojstava koja imaju međusobno suprotne domenu i doseg.

Svojstva kao u Kôd 4.18 jasno vidljivo imaju inverzne domenu i doseg, tj. svojstvo `jeZavršio` subjektu koji je individua klase `Osoba` pridružuje individuu klase `Fakultet`, dok svojstvo `bivšiJeStudent` čini upravo suprotno, individui klase `Fakultet` pridružuje individue klase `Osoba`. Svojstva ovakvog oblika su inverzna, pa je poželjno navedeni zapis proširiti kako slijedi u Kôd 4.19.

```
ObjectProperty: jeZavršio
    InverseOf: bivšiJeStudent
```

Kôd 4.19: Manchester zapis kojim je određeno da je dano svojstvo inverzno nekom drugom svojstvu.

Kao što je vidljivo u Kôd 4.19, svojstvo inverzno trenutno promatranom označava se ključnom riječi `InverseOf`. Jasno, inverz ne može biti podatkovno svojstvo, zato što OWL ne dozvoljava da literal bude subjekt (Halpin, 2010c).

Postoje predikati koji za subjekte i objekte mogu imati isti skup elemenata. Takva svojstva, koja u svojstvu domene i dosega imaju isti skup, nazivaju se prstenastim svojstvima, a mogu zadovoljavati karakteristike refleksivnosti, irefleksivnosti, simetričnosti, asimetričnosti i tranzitivnosti.

Svojstva imaju određene karakteristike, od kojih je samo prva od sljedećih primjenjiva na podatkovna svojstva, dok su sve primjenjive na objektna svojstva (W3C, 2004; Halpin, 2011b):

- Funcionalnost (`Functional`): predikat oblika N:1 ima karakteristiku funkcionalnog svojstva, što znači da svakom subjektu pridružuje najviše jedan objekt, tj. da je objekt funkcija subjekta te da se dobiva ograničenje jedinstvenosti
- Inverzna funkcionalnost (`InverseFunctional`): funkcionalnost u suprotnom smjeru određuje da je objektu danim predikatom pridružen najviše jedan subjekt, tj. osigurava ograničenje jedinstvenosti objektu;
- Refleksivnost (`Reflexive`): formalno izraženo, svojstvo je refleksivno ako i samo ako je svaki element domene danim predikatom povezan sam sa sobom;
- Irefleksivnost (`Irreflexive`): prstenasto svojstvo refleksivno je isključivo kad vrijedi da svaki element domene nije danim svojstvom povezan sam sa sobom;
- Simetričnost (`Symmetric`): simetrično prstenasto svojstvo ograničeno je na način da za svaka dva elementa postoji simetrično svojstvo, takvo da spaja dane elemente u suprotnom smjeru, pri čemu je korisno napomenuti da simetrično svojstvo ima jednake domenu i doseg;
- Asimetričnost (`Asymmetric`): asimetrično svojstvo ograničava da za svaka dva povezana elementa, ne postoji simetrično svojstvo, tj. da ista dva elementa istim svojstvom nisu povezana u suprotnom smjeru;
- Tranzitivnost (`Transitive`): svojstvo je tranzitivno ako i samo ako za tri individue, ako je prva povezana s drugom, i druga s trećom, moguće je zaključiti da je istim predikatom prva individua povezana s trećom.

Najčešći primjer refleksivnog svojstva je predikat `poznaje`, određen kao u Kôd 4.20:

```
ObjectProperty: poznaje
```

```
    Domain: Osoba
    Range: Osoba
    Characteristics: Reflexive
```

Kôd 4.20: Manchester zapis refleksivnog svojstva poznaje.

Temeljem svojstva kakvo je opisano u Kôd 4.20, za bilo koju individuu klase `Osoba` može se zaključiti da je ta individua danim svojstvom povezana sama sa sobom.

Irefleksivnost je korisno naznačiti u slučaju kad su domena i doseg danog svojstva ista klasa, no nije moguće da svojstvom budu povezane iste individue. Drugim riječima, svojstvo je potrebno naznačiti kao irefleksivno kad istim tim svojstvom mogu biti povezane samo različite individue neke klase.

Tranzitivno svojstvo na neki način određuje nasljednost. Najčešće je objašnjeno predikatom `jePredakOd`, koji bude definiran kao u Kôd 4.21.

```
ObjectProperty: jePredakOd
    Domain: Osoba
    Range: Osoba
    Characteristics: Transitive, Asymmetric
```

Kôd 4.21: Manchester zapis tranzitivnog asimetričnog svojstva *jePredakOd*.

Ovako ograničenim svojstvom, kao u Kôd 4.21, iz činjenica zapisanih u Kôd 4.22 može se zaključiti činjenica zapisana u Kôd 4.23.

```
Individual: Ozano
   Types: Osoba
   Facts: jePredakOd Zrinka
```

```
Individual: Zrinka
   Types: Osoba
   Facts: jePredakOd Zvonko
```

Kôd 4.22: Manchester zapis individue *Ozano* i *Zrinka*, gdje je individua *Ozano* predak od individue *Zrinka*, a *Zrinka* predak od individue *Zvonko*.

```
Individual: Ozano
    Facts: jePredakOd Zvonko
```

Kôd 4.23: Manchester zapis individue *Ozano*, koja je predak od individue *Zvonko*.

Svojstva je korištenjem OWL-a moguće ulančavati u složene nizove svojstava i tako određivati nova svojstva, koja ovise isključivo o povezanim svojstvima (Halpin, 2012a). Niz svojstava označava se ključnom riječi `SubPropertyChain` i veznikom `o`, a stvara se kao što je to navedeno u Kôd 4.24.

```
ObjectProperty: slušaPredmet
    Domain: Student
    Range: Predmet
ObjectProperty: predaje
    Domain: Predmet
    Range: Nastavnik
ObjectProperty: SlušaNastavnika
    SubPropertyChain: slušaPredmet o predaje
```

Kôd 4.24: Manchester zapis svojstva *SlušaNastavnika* kao niza od svojstava *slušaPredmet* i predaje.

Iz Kôd 4.24 lako je zaključiti da, ukoliko student sluša neki određeni predmet, a taj predmet predaje neki određeni nastavnik, dani student sluša tog određenog nastavnika, što proizlazi iz zadanog niza svojstava.

Niz svojstava može se sastojati od većeg broja svojstava, koja mogu i ne moraju biti jednaka. Primjer gdje su svojstva u nizu jednaka prikazan je u Kôd 4.25, dok je primjer sa više svojstava zapisan u Kôd 4.26.

```
ObjectProperty: jeUnukOd
    SubPropertyChain: jeDijeteOd o jeDijeteOd
```

Kôd 4.25: Manchester zapis svojstva *jeUnukOd* kao niza od dva *jeDijeteOd* svojstva.

```
ObjectProperty: studiraUDržavi
    SubPropertyChain: podađaFakultet o jeUGradu o jeUDržavi
```

Kôd 4.26: Manchester zapis svojstva *studiraUDržavi* kao niza od tri svojstva.

Niz svojstava kojima je ograničeno svojstvo iz Kôd 4.26, čita se kako slijedi: „Student pohađa fakultet koji je u gradu koji je u državi…“. Primjetno je korištenje ključne riječi „koji“ za povezivanje značenja svojstava u nizu.

4.5.4 Skupni operatori

OWL dopušta korištenje operatora unije, presjeka i komplementa, koji imaju već standardizirano značenje i način korištenja, kako slijedi ukratko.

Neka su A i B skupovi elemenata. Tada se unija, unutar teorije skupova, određuje kao skup elemenata koji pripadaju skupu A ili skupu B ili pripadaju i skupu A i skupu B. Unutar OWL-a uniju je moguće koristiti na klasama ili literalima ključnom riječi `or`, kao što je to prikazano u Kôd 4.27 (Halpin, 2011a).

```
Class: StudentDiplomskog
    EquivalentTo: StudentOPS or StudentIuO or StudentBPBZ or
StudentIPI or StudentDSEP
```

Kôd 4.27: Manchester zapis klase *StudentDiplomskog* kao unije klasa.

Particiju vrijednosti moguće je u OWL-u izraziti pomoću ključne riječi `DisjointClasses`, gdje klase navedene uz tu ključnu riječ predstavljaju međusobno isključive klase, tj. klase koje nemaju zajedničkih individua (Halpin, 2011a). Kôd 4.28 određuje klasu koja je unija takvih klasa. Isto je moguće postići korištenjem ključne riječi `DisjointUnionOf`, kao u Kôd 4.29, čime se ograničava neku promatrana klasa kao unija klasa koje nemaju zajedničkih individua.

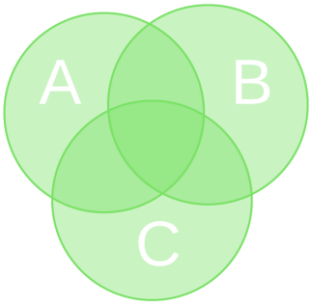
```
DisjointClasses: Čaj, Kava
Class: TopliNapitak
    EquivalentTo: Čaj or Kava
```

Kôd 4.28: Manchester zapis dviju međusobno isključivih klasa i klase koja potpuno obuhvaća obje te klase.

```
Class: TopliNapitak
    DisjointUnionOf: Čaj, Kava
```

Kôd 4.29: Manchester zapis kojim se izravno određuje da klasa *TopliNapitak* obuhvaća klase *Čaj* i *Kava*, dok su one međusobno isključive.

Presjek neka dva skupa sadrži elemente koji se nalaze u oba ta skupa (Halpin, 2011a). OWL presjek označava ključnom riječi `and`. Analogno, presjek većeg broja skupova sadrži isključivo elemente koji pripadaju svim tim skupovima, kao što je to prikazano na Slika 4.5. Kôd 4.30 prikazuje primjer presjeka izražen Manchester zapisom.



Slika 4.5: Presjek triju skupova, prikazan Vennovim dijagramom, gdje presjek predstavlja područje gdje se sva tri skupa preklapaju, označeno najtamnije zeleno. (izvor: izrada autora)

```
Class: Student
    SubClassOf: Osoba
Class: Zaposlen
    SubClassOf: Osoba
Class: IzvanredniStudent
    EquivalentTo: Student and Zaposlen
```

Kôd 4.30: Manchester zapis klase *IzvanredniStudent* definirane kao presjek klasa *Student* i *Zaposlen*.

U pravilu svaki izvanredni student mora biti zaposlen, dok je ista ta osoba svakako student, pa je moguće zaključiti da je `IzvanredniStudent` presjek klasa `Student` i `Zaposlen`, kao što je prikazano u Kôd 4.30.

Komplement nekog skupa su svi elementi koji se ne nalaze u tom skupu, a unutar OWL-a se označava ključnom riječi `not` (Halpin, 2011a). Općenito unutar OWL-a, svi elementi pripadaju klasi `Thing`, pa je moguće reći da komplement nekog skupa obuhvaća sve elemente klase `Thing` koji ne pripadaju promatranoj klasi. Primjer komplementa je izražen u Kôd 4.31.

```
Class: PoložiliIspit
    SubClassOf: IzašliNaIspit
    EquivalentTo: not PaliIspit
```

Kôd 4.31: Manchester zapis klase studenata koji su položili ispit kao komplementa klase studenata koji su pali ispit, pri čemu su promatrani samo studenti koji su uopće izašli na ispit.

Navedene logičke operatore moguće je međusobno kombinirati, kao u Kôd 4.32:

```
Class: Vozač
    EquivalentTo: Osoba and vozi min 1 Auto
Class: NeVozač
    EquivalentTo: Osoba and not Vozač
```

Kôd 4.32: Manchester zapis klase *Vozač* pomoću presjeka te klase *NeVozač* pomoću presjeka i komplementa.

4.5.5 Ekstenzijsko određivanje i ograničavanje vrijednošću

Klase i literale moguće je, osim svojom definicijom, kao što je ranije navedeno, odrediti i ograničavanjem skupa vrijednosti koje mogu poprimiti. Takvo određivanje klasa i podatkovnih tipova provodi se ograničavanjem elemenata ekstenzije, pa odatle naziv ekstenzijsko određivanje. Klasu `RadniDan` moguće je ograničiti na svega nekoliko vrijednosti, kao što je to prikazano u Kôd 4.33 (Halpin, 2011c).

```
Class: RadniDan
    EquivalentTo: {Ponedjeljak, Utorak, Srijeda, Četvrtak, Petak}
```

Kôd 4.33: Manchester zapis klase *RadniDan*, koja ima konačan skup elemenata.

Klase je pomoću OWL-a moguće ograničavati i izričitim navođenjem određene vrijednosti, koja može biti literal, element ekstenzije klase ili pak ograničenje sastavljeno od svojstava i vrijednosti koja ona mogu poprimiti. Tako je studenta diplomskog studija moguće odrediti kako je prikazano u Kôd 4.34:

```
Class: RazinaStudiranja
    EquivalentTo: {preddiplomska, diplomska, poslijediplomska}
Class: StudentDiplomskog
    EquivalentTo: Student and studiraNaRazini value diplomska
```

Kôd 4.34: Manchester zapis klase *StudentDiplomskog* koja je ograničena klasom *Student* i vrijednošću svojstva *studiraNaRazini*.

Ograničavanje vrijednošću proširuje mogućnosti korištenjem ključnih riječi `some` ili `only`, kao u primjeru u Kôd 4.35:

```
Class: VoćnaSalata
    EquivalentTo: Salata and imaSastojke only Voće
Class: SalataSaSirom
    EquivalentTo: Salata and imaSastojke some Povrće and
imaSastojke some Sir
```

Kôd 4.35: Manchester zapis klase *VoćnaSalata* koja ima sastojke koji pripadaju isključivo klasi *Voće*, dok se instanca klase *SalataSaSirom* sastoji od povrća i od sira.

U Kôd 4.35 prikazana je klasa `VoćnaSalata` koja ima određene sastojke, no svi sastojci koje korisnik navede nužno i isključivo moraju biti članovi klase `Voće`. S druge strane, klasa `SalataSaSirom` ima neograničen broj (i neobavezan) sastojaka koji mogu biti članovi klasa `Povrće` ili `Sir`.

4.5.6 Negacija

OWL u svoj zaključivanju koristi pretpostavku otvorenog svijeta, što znači da je moguće da je zapisano znanje o svijetu nepotpuno, što bilo koju izjavu koja nije zapisana ili zaključiva iz zapisanog, čini potencijalno istinitom, zato što nije moguće, temeljem dostupnih informacija, zaključiti da je ona neistinita. Ovakav se pristup uvelike razlikuje od pretpostavke zatvorenog svijeta, koja se obično koristi kod baza podataka, gdje je istina isključivo ono što je izričito zapisano. Negacija se u OWL-u označava ključnom riječi `not` (Halpin, 2012b).

5 Semantičko modeliranje poslovnih pravila

Glavna tema ovog diplomskog rada, funkcionalni spoj semantičkog modeliranja i poslovnih pravila bit će prikazani kroz korištenje OWL-a i jezika za izražavanje pravila u okruženju semantičkog weba (eng. *Semantic Web Rule Language*, SWRL). OWL je detaljno opisan ranije u radu, dok je SWRL ovdje novospomenuti koncept.

5.1 SWRL (O'Connor, 2012)

Jezik pravila za semantički web (eng. *Semantic Web Rule Language*, SWRL) razvio se iz RuleML jezika koji je prvenstveno namijenjen istraživanju, a koji objedinjuje porodicu XML-serijaliziranih jezika pravila koja se proteže kroz sve industrijski standardizirane oblike web pravila[16].

SWRL objedinjuje moć konceptualizacije i razvoja modela OWL-a i izražajnost koju nudi RuleML u suradnji sa OWL-om. Prvotno prijavljen 2004. godine, ovaj jezik sadrži potpunu snagu OWL DL jezika, na uštrb odlučivosti i praktičnih implementacija. Kao jezik za izražavanje pravila temeljenih na konceptima OWL-a, SWRL omogućava u svojim izrazima korištenje OWL koncepata kako bi osigurao mogućnosti zaključivanja naprednije od onih koje ima sam OWL. Semantički je SWRL izgrađen korištenjem deskriptivne logike, iste osnove kakvu ima OWL. (O'Connor, 2012)

SWRL pravila sastoje se od dva dijela: premisa (antecedent) i zaključak (konsekvent). Premisa predstavlja tijelo SWRL pravila, dok zaključak predstavlja tzv. glavu SWRL pravila. Svako pravilo u svojoj suštini sastoji se od atoma, pa tako i premisa i zaključak, kako slijedi:

$$atom \wedge atom \wedge atom \rightarrow atom \wedge atom$$
$$premisa \rightarrow zaključak$$

pri čemu su atomi unutar SWRL pravila povezani isključivo konjunkcijom, gdje se zaključak ispunjava u slučaju kad su svi atomi premise zadovoljeni. Ukoliko bi za povezivanje atoma bila korištena disjunkcija, dolazi do problema pri korištenju varijabli, kako će biti prikazano nešto kasnije.

Svaki atom je sljedećeg oblika:

$$p(arg1, arg2, \ldots, argn)$$

gdje je p simbol predikata iz OWL-a, dok su $arg1, \ldots, argn$ argumenti tog predikata. Pri tome simbol predikata može predstavljati OWL klasu, svojstvo ili tip podatka. Argumenti mogu biti OWL individue, podatkovne vrijednosti ili varijable koje se na njih odnose.

5.1.1 SWRL atomi

Atome je moguće koristiti na razne načine, kombiniranjem OWL klasa, svojstava, tipova podataka, individua, podatkovnih vrijednosti i varijabla, kao što će ovdje biti prikazano.

Najjednostavnije korištenje atoma je za prikaz OWL individue, kao što je to prikazano u SWRL 5.1.

[16] http://ruleml.org, pristupljeno 6.3.2013.

Osoba(?o)

SWRL 5.1: Varijabla *?o* postaje individua OWL klase *Osoba.*

Izraženo u SWRL 5.1 `Osoba` je OWL klasa, dok je `?o` varijabla koja predstavlja individuu klase `Osoba`. Varijablu je moguće zamijeniti i konkretnom OWL individuom, kao u SWRL 5.2. U slučaju zapisa konkretne OWL individue provjerava se da li je dana individua član ekstenzije dane klase.

Osoba(Maja)

SWRL 5.2: Provjerava se je li OWL individua naziva *Maja* član ekstenzije OWL klase *Osoba.*

SWRL pravilo korištenjem isključivo naziva klasa kao simbola predikata može biti kao ono zapisano u SWRL 5.3:

Osoba(?o), PrivatniKlijent(?o) -> PrivatnaOsoba(?o)

SWRL 5.3: Osoba koja je *PrivatniKlijent* je *PravnaOsoba*.

Navedeno pravilo moguće je detaljno pročitati kako slijedi: ako je varijabla naziva `?o` individua klase `Osoba` i ako je ta ista varijabla individua klase `PrivatniKlijent`, onda je ta varijabla individua klase `PrivatnaOsoba`. Drugim riječima izraženo, varijabla koja je individua klase `Osoba` i koja je individua klase `PrivatniKlijent`, individua je klase `PrivatnaOsoba`. Konačno, isto je pravilo moguće shvatiti kako slijedi: ako je varijabla `?o` individua klase `Osoba` i individua klase `PrivatniKlijent`, onda je ta varijabla individua klase `PrivatnaOsoba`.

Dakako, za potrebe izražavanja takve ovisnosti nije nužno koristiti SWRL, već je lako moguće istu zavisnost, isto pravilo, izraziti korištenjem isključivo OWL elemenata, bez potrebe za SWRL pravilom. Tako izraženo ograničenje zapisuje se uz korištenje ključne riječi `EquivalentTo` u sklopu definiranja klase `PrivatnaOsoba`, kao što je zapisano u Kôd 5.1:

```
Class: PrivatnaOsoba
    EquivalentTo: Osoba and PrivatniKlijent
```

Kôd 5.1: Manchester zapis klase *PrivatnaOsoba* kao presjeka klasa *Osoba* i *PrivatniKlijent.*

Svaka OWL individua može imati dodijeljena joj nekakva svojstva. Takvim se svojstvima bave atomi svojstava individua, kojima se svojstva izražavaju kao u SWRL 5.4, pri čemu u atomu svojstva individua uvijek postoji naziv danog svojstva i dvije varijable koje označavaju elemente koji sudjeluju u promatranom svojstvu.

imaMentora(?z, ?m)

SWRL 5.4: Istinito u slučaju kad je individua *?z* povezana s individuom *?m* svojstvom *imaMentora.*

Jasno, atome klasa i atome svojstava individua moguće je kombinirati, kao i ostale vrste atoma koje će biti spomenute kasnije. Za potrebe klasifikacije neke individue kao člana klase `Voditelj` moguće je iskoristiti sljedeće pravilo SWRL 5.5:

Zaposlenik(?z), vodiDioOrganizacije(?z, ?o), DijeloviOrganizacije(?o) -> Voditelj(?z)

SWRL 5.5: *Zaposlenik* koji *vodiDioOrganizacije* je *Voditelj*.

Pravilo SWRL 5.5 sve varijable `?z` koje zadovoljavaju ograničenja postavljena u premisi klasificira kao individue klase `Voditelj`. Svaka individua `?z` koja je individua klase `Zaposlenik` i koja sudjeluje u svojstvu `vodiDioOrganizacije` sa individuom `?o` koja je individua klase `DijeloviOrganizacije`, individua je klase `Voditelj`.

Nadalje, zaključak nije nužno atom klase, već može biti i neki drugi oblik atoma, pa pravilo može biti izraženo kao ono u SWRL 5.6:

Odjel(?o), jeDioIspostave(?o, ?i), Ispostava(?i), jeDioPodružnice(?i, ?p), Podružnica(?p) -> jeOdjelPodružnice(?o, ?p)

SWRL 5.6: Ako je *Odjel jeDioIspostave* koja *jeDioPodružnice*, onda je *Odjel jeOdjelPodružnice*.

Kao što je vidljivo iz pravila SWRL 5.6, premisa se može sastojati od većeg broja atoma, no bitno je da su svi atomi povezani konjunkcijom. Također, moguće je koristiti uvelike izmiješane atome klasa i atome svojstava. Navedeno pravilo iskazuje da, ukoliko je individua `?o` individua klase `Odjel` i sudjeluje u svojstvu `jeDioIspostave` sa individuom `?i` koja je individua klase Ispostava i koja sudjeluje u svojstvu `jeDioPodružnice` sa individuom `?p` koja je individua klase Podružnica, onda individua `?o` sudjeluje u svojstvu `jeOdjelPodružnice` sa individuom `?p`.

Za razliku od atoma svojstva individua, atomi podatkovnih svojstava sadrže i varijable koje se ne odnose isključivo na OWL individue, već i na podatkovne vrijednosti. Korištenje takvih varijabli analogno je korištenju varijabli koje predstavljaju OWL individue, kao što je prikazano u SWRL 5.7:

imaStanovnika(?m, ?broj)

SWRL 5.7: Svojstvo *imaStanovnika* pridružuje nekom mjestu cijeli broj koji predstavlja broj stanovnika tog mjesta.

Svojstvo `imaStanovnika` u SWRL 5.7 povezuje individuu klase `Mjesto` sa prirodnim brojem koji označava broj stanovnika danog mjesta. Varijable koje poprimaju podatkovne vrijednosti mogu sudjelovati u nekoliko svojstava koja su predefinirana u SWRL jeziku, a koja služe za osnovne operacije uspoređivanja, rada s brojevima, nizovima znakova, vremenskim podacima te RDF listama. Primjer usporedbe vrijednosti izražen je u pravilu SWRL 5.8:

Mjesto(?m), imaStanovnika(?m, ?broj), greaterThan(?broj, 10000) -> Grad(?m)

SWRL 5.8: Mjesto koje ima broj stanovnika veći od 10 000 je grad.

Pravilo SWRL 5.8 moguće je pročitati kako slijedi: ako varijabla `?m` predstavlja individuu klase `Mjesto` koja ima broj stanovnika `?broj` i taj `?broj` je veći od 10 000, onda varijabla `?m` predstavlja individuu koja je i individua klase `Grad`.

5.1.2 Ugrađena svojstva

Jedno od predefiniranih SWRL svojstava je i svojstvo koje omogućuje označavanje individua kao različitih, ukoliko one nisu takvima označene ranije. SWRL izraz poput pravila SWRL 5.9, opisuje svojstvo koje ima domenu jednaku dosegu tog svojstva.

Zaposlenik(?z), jeMentorZaposleniku(?z, ?z2), Zaposlenik(?z2) -> Mentor(?z)

SWRL 5.9: Ako je zaposlenik mentor zaposleniku, onda je on mentor.

Svojstvo uključeno u pravilo SWRL 5.9 ima domenu i doseg koji se u potpunosti preklapaju, tj. sve individue uključene u svojstvo `jeMentorZaposleniku` moraju biti individue klase `Zaposlenik`. Uz ovo ograničenje, postoji i ograničenje irefleksivnosti, zato što ne može zaposlenik biti mentor sam sebi. Upravo u ovakvom pravilu, gdje je potrebno dodatno ograničiti korištenje individua i njihovo sudjelovanje u predikatima, dodaje se atom koji označava da su dvije individue istoznačne. Pravilo SWRL 5.9 tako je potrebno dodatno ograničiti jednim od SWRL predefiniranih svojstava, kao što je prikazano u pravilu SWRL 5.10:

Zaposlenik(?z), jeMentorZaposleniku(?z, ?z2), Zaposlenik(?z2), differentFrom(?z, ?z2) -> Mentor(?z)

SWRL 5.10: Ako je zaposlenik mentor zaposleniku, ne samome sebi, onda je on mentor.

U pravilu SWRL 5.10 ključnom riječi `differentFrom` uvedeno je ograničenje da varijable `?z` i `?z2` predstavljaju individue koje su međusobno različite. Analogno ovome postoji atom koji određuje individue kao međusobno jednake, a označava se ključnom riječi `sameAs`.

Predefinirana SWRL svojstva, tzv. ugrađeni atomi, u pravilima sudjeluju ravnopravno sa atomima koji se odnose na OWL entitete, pri čemu je varijable iz ugrađenih atoma moguće sasvim uobičajeno koristiti u ostalim atomima. Korištenje jednog takvog atoma prikazano je u pravilu SWRL 5.11, gdje je varijabla poprimila izlaznu vrijednost jednog ugrađenog svojstva, nakon čega je dalje korištena u pravilu.

Račun(?r), imaKupca(?r, ?k), Klijent(?k), imaIznos(?r, ?i), double(?i), imaPopust(?k, ?p), double(?p), multiply(?ukupno, ?i, ?p) -> imaIznosUkupno(?r, ?ukupno)

SWRL 5.11: Račun koji ima kupca koji je klijent i koji ima određeni iznos, a kupac ima određeni popust, ukupan iznos računa se umanjuje za iznos popusta.

Pravilo navedeno u SWRL 5.11 koristi cijeli niz svojstava, individua i koncepata za zaključivanje konačnog ukupnog iznosa koji mora biti naveden na računu. U ovome je primjeru odmah vidljivo koliko je jednostavno povezati svojstva i individue te doći do željenog rješenja. Objašnjenje ograničenja pravila ovdje će biti prikazano u malo proširenom izdanju. Glavni subjekt premise pravila je individua klase `Račun`, koja je svojstvom `imaKupca` vezana sa varijablom `?k` koja predstavlja individuu klase `Klijent`. Dana individua klase `Račun` sudjeluje u svojstvu `imaIznos` sa nekom `double` vrijednošću, dok je individua klase `Klijent` sa `double` vrijednošću povezana svojstvom `imaPopust`. U konačnici, potrebno je popust klijenta pomnožiti sa iznosom računa, kako bi se dobio ukupni iznos, u koji je uključen popust ovisno o statusu klijenta. Množenje iznosa i popusta u ovom je pravilu riješeno korištenjem ključne riječi `multiply`, koja predstavlja ugrađeno SWRL svojstvo za množenje brojeva, a koristi najmanje tri argumenta: prvi koji predstavlja rezultat množenja, ovdje ga koristi varijabla `?ukupno` i ostali koji sudjeluju u množenju, ovdje varijable `?i` i `?p`.

SWRL jezik razvijen je na temeljima OWL-a i RuleML-a te stoga može u svojim izrazima koristiti izraze koji se u OWL-u mogu koristiti za naprednije ograničavanje klasa.

Nastavno na upravo navedeno, SWRL baš kao i OWL djeluje uz pretpostavku otvorenog svijeta, u kojoj se ne može zaključiti da nešto jest ili nije istina ukoliko isto nije izričito zapisano. Tako je ranije već u ovom radu navedeno da standardna baza podataka djeluje uz pretpostavku zatvorenog svijeta, gdje sve što je zapisano predstavlja sve moguće znanje i ne postoji ništa izvan tog zapisanog svijeta. Primjerice, ukoliko je zapisana činjenica da „Vedran ima crveni auto.“, moguće je zaključiti da nije istina da „Vedran ima zeleni auto.“ Suprotno tome, uz pretpostavku otvorenog svijeta, drugi izraz nije moguće ocijeniti kao neistinitog, zato što ta činjenica nigdje nije izričito navedena. Osim navedenog, korištenje pretpostavke otvorenog svijeta uzrok je nužnog korištenja atoma jedinstvenosti prilikom rada sa individuama, tj. pretpostavka otvorenog svijeta koju SWRL koristi ne zaključuje da su individue međusobno različite samo zato što one imaju nazive koji se razlikuju. Takvo zaključivanje rješava se kao što je to objašnjeno uz pravilo SWRL 5.10.

Nadalje, upravo kao i OWL, SWRL ne dopušta nemonotono zaključivanje, što dovodi do činjenice da korištenjem SWRL-a nije moguće mijenjati ili uklanjati informacije iz ontologije. Moguće je dodavati, no uklanjanje ili mijenjanje već zapisanog nije omogućeno.

5.1.3 Disjunkcija?

Kao što je ranije napomenuto, SWRL izrazi uvijek se sastoje od premise i zaključka, koji se pak sastoje od atoma povezanih konjunkcijom. Spajanje atoma disjunkcijom nije moguće zbog potencijalnih komplikacija na koje je moguće naići. U slučaju prijeke potrebe definiranja disjunkcije, ipak je moguće iskoristiti način djelovanja SWRL-a. Apstrahirano pravilo SWRL 5.12, koje u takvom obliku nije podržano, moguće je provesti dijeljenjem na dva pravila koja su navedena u SWRL 5.13. Takvim dijeljenjem uvedena su dva nova pravila, od kojih će se jedno ili drugo provesti te će dovesti do istinitosti zaključka.

$$A(?b) \vee B(?b) \rightarrow C(?b)$$

SWRL 5.12: Atomi premise povezani konjunkcijom.

$$A(?b) \rightarrow C(?b)$$

$$B(?b) \rightarrow C(?b)$$

SWRL 5.13: Konjunkcija iz prethodnog primjera razbijena je na dva pravila.

Ukoliko je prvo pravilo u SWRL 5.13 nezadovoljivo, provodi se evaluacija drugog pravila te bi se takav postupak nastavio do zadnjeg pravila u nizu. Činjenica je da su ova pravila povezana zaključkom te uzimajući u obzir da se radi o identičnom zaključku, moguće je skup pravila SWRL 5.13 shvatiti tako da prvo pravilo vodi do zaključka, ili drugo pravilo vodi do zaključka itd. Pa ipak, sukladno ranije navedenom, ukoliko su oba pravila u SWRL 5.13 zadovoljiva, drugim pravilom činjenica koju uvodi prvo pravilo neće biti izmijenjena, već samo nadodana, što može zakomplicirati zapis i kasnije zaključivanje. Jasno, ovo je vidljivo prvenstveno u slučaju da zaključak uvodi funkcionalno svojstvo.

5.1.4 OWL/XML zapis

Sva do sad navedena SWRL pravila izražena su korištenjem sintakse koja je čitljiva čovjeku, služeći se riječima koje imaju značenje i stvaraju lako shvatljive cjeline. Standardna sintaksa za pohranjivanje ontologija je OWL/XML, pa su i pravila, sastavni dijelovi ontologija, standardno zapisana korištenjem XML zapisa. Pravilo izraženo ljudski čitljivom sintaksom u SWRL 5.14 u XML zapisu izgleda kao Kôd 5.2.

Klijent(?k), imaKategorijuKlijenta(?k, KategorijaBronca)
-> imaPopust(?k, 0.97)

SWRL 5.14: Klijent brončane kategorije ima popust 3%.

```
<DLSafeRule>
    <Body>
        <ClassAtom>
            <Class IRI="#Klijent"/>
            <Variable IRI="urn:swrl#k"/>
        </ClassAtom>
        <ObjectPropertyAtom>
            <ObjectProperty IRI="#imaKategorijuKlijenta"/>
            <Variable IRI="urn:swrl#k"/>
            <NamedIndividual IRI="#KategorijaBronca"/>
        </ObjectPropertyAtom>
    </Body>
    <Head>
        <DataPropertyAtom>
            <DataProperty IRI="#imaPopust"/>
            <Variable IRI="urn:swrl#k"/>
            <Literal datatypeIRI="&xsd;double">0.97</Literal>
        </DataPropertyAtom>
    </Head>
</DLSafeRule>
```

Kôd 5.2: Pravilo SWRL 5.14 zapisano u OWL/XML obliku.

OWL/XML zapis SWRL pravila prati određeni predložak:

- cijelo je pravilo, opcionalno, smješteno u `<DLSafeRule>` oznaku koja ukazuje na to da se radi o pravilu koje je sigurno za zaključivanje u okruženju deskriptivnih logika, što je način zaključivanja koji se koristi u OWL-u;
- premisa je smještena u `<Body>` oznaku;
- zaključak se nalazi unutar `<Head>` oznake;
- svaki atom pravila smješten je unutar svoje oznake, koja ovisi o prirodi atoma: u primjeru Kôd 5.2 korištene su različite oznake:
    - `<ClassAtom>` za atom klase,
    - `<ObjectPropertyAtom>` za atom koji se odnosi na objektno svojstvo, tj. atom svojstva individue,
    - `<DataPropertyAtom>` za atom podatkovnog svojstva;
- unutar oznake svakog atoma nalaze se dodatne oznake sa pripadajućim vrijednostima koja uređuju elemente koji sudjeluju u danom atomu, poput:
    - Naziva klase i varijable koja označava individuu te klase, za prvi atom u SWRL 5.14,
    - Naziva objektnog svojstva koje čini drugi atom u SWRL 5.14 te argumenata koji u njemu sudjeluju, u ovom slučaju jedna varijabla i jedna imenovana individua.

SWRL pravila uobičajeno se koriste za dodatnu izražajnost kad OWL izrazi za to nisu dovoljni. Upravo zbog činjenice da za izražavanje uobičajenih činjenica, kao da je neki entitet individua neke klase ili da su dvije individue povezane određenim svojstvom ili, konačno, da

je dana individua određenim svojstvom povezana nekim literalom, nema potrebe za korištenjem SWRL izraza, već je dostatno jednostavno korištenje OWL izraza, poslovna su pravila ovdje izražena u jednostavnom obliku korištenjem OWL izraza, dok su SWRL pravila korištena za zaključivanje na nešto višoj i kompleksnijoj razini.

5.2 RIF

U ovome radu valja spomenuti i format za razmjenu pravila (RIF, eng. *Rule Interchange Format*) koji je s razvojem počeo 2005. godine, a zadnja formalizirana verzija objavljena je 5. veljače 2013. godine, što ga čini budućim standardom u razvoju. Glavna ideja RIF-a je stvaranje standarda za razmjenu pravila između sustava, kao kontrast dosadašnjim nastojanjima izrade standardiziranog jezika (W3C, 2013.). U raznolikosti jezika i dijalekata, familija RIF dijalekata trebala bi biti uniformirana i proširiva, što znači da se očekuje da RIF dijalekti koriste što je više moguće standardiziranih i uobičajenih sintaktičkih i semantičkih elemenata i da bude moguće razvijati dodatne RIF dijalekte koji služe kao proširenje već postojećih. Dijalekti u ovom kontekstu podrazumijevaju svojevrsne grupe jezika unutar RIF standarda. Idejno je zamišljeno da postoje dva glavna dijalekta (W3C, 2013.): dijalekt temeljen na logici (RIF-BLD) i dijalekt za pravila s akcijama (RIF-PRD). Dijalekt temeljen na logici koristi neku vrstu logike, poput logike prvog reda. Dijalekt za pravila s akcijama bavi se sustavima produkcijskih pravila i reaktivnim pravilima.

Za potrebe ovog rada najzanimljiviji je upravo dijalekt temeljen na logici, koji je najviše namijenjen web okruženju, iako nije ograničen na isto. Zajedno sa RIF OWL i RIF RDF čini integrirane RIF-BLD/RDF i RIF-BLD/OWL jezike. Bitno je napomenuti ovdje da RIF nema za cilj stvoriti okruženje koje služi za rješavanje problema razmjene pravila u najširem kontekstu, već se očekuje da će već postojeći jezici za iskazivanje pravila biti u mogućnosti u RIF prevesti samo pravila određene vrste, dok se od osoba koje modeliraju pravila očekuje da će s vremenom određena, pretpostavka da su to najbitnija, pravila modelirati po načelu da su prilagođena RIF standardu, pa samim time i razmjenjiva među sustavima koji podržavaju neki od RIF dijalekata.

RIF, dakako, nije napravljen ni iz čega, već neke karakteristike dijeli sa ISO standardiziranom logikom (CL, eng. *ISO Common Logic*[17]), pa upravo poput CL, RIF glavnu strukturu nalazi u XML sintaksi, koristi internacionalizirane označivače elemenata (IRI), proširuje se RIF-BLD/RDF i RIF-BLD/OWL jezicima za rad sa semantičkim webom te ima bogatu lepezu podržanih tipova podataka i ostalih ugrađenih elemenata (W3C, 2013.).

Ovdje je iznimno bitno napomenuti da je RIF prvenstveno namijenjen razmjeni pravila, dok SWRL u prvom redu jezik za izražavanje pravila stvoren kao dodatak na OWL. (RIF Working Group, 2013) S obzirom na njegovu jednostavnost, većina mogućnosti SWRL jezika obuhvaćene su RIF izražajnošću. Štoviše, RIF nudi mogućnosti koje SWRL ne obuhvaća. U korist SWRL-a ide ugrađeno svojstvo koje opisuje dvije individue različitima, dok RIF, kao dokaz šireg spektra mogućnosti, dozvoljava korištenje predikata više arnosti, funkcija, većeg skupa ugrađenih tipova podataka i svojstava te korištenje disjunkcije u pravilima.

5.2.1 Odnos RIF i SWRL

Prema (Ma & Wang, 2012) gdje se predlaže arhitektura za razmjenu pravila temeljena na XML sintaksi (eng. RIAxml, *Rule Interchange Architecture based on XML syntax*), SWRL pravila prevode se u RIF pravila, i obrnuto, korištenjem postupaka prevođenja i mapiranja. Štoviše, autori predlažu rješenje u obliku RIAxml-a koje prevodi trenutno najpopularnije jezike

[17] Više na http://common-logic.org/

za izražavanje pravila (SWRL, RuleML, R2ML i F-logika) u RIF te iz RIF-a u jedan od navedenih jezika. Posebno ističu da je najveći problem prevođenja ograničena izražajnost RIF-a, što dovodi do gubitka informacija i ograničene prevodivosti.

Ma i Wang (2012, p. 399) navode da, uzrokovano činjenicom da je RIF stvoren kao jednostavan dijalekt ograničene i slabe izražajnosti, ne postoji element kojim se može izraziti SWRL element koji označava atom klase koji određuje komplement entiteta. Također, RIF ne poznaje definicije za neke od atributa koje SWRL nasljeđuje od OWL-a, poput simetričnosti i tranzitivnosti.

Prilikom prevođenja RIF izraza u SWRL izraze, Ma i Wang (2012, p. 398) navode da je najveći problem što SWRL dopušta da su premisa i zaključak sastavljeni isključivo od konjunkcije atoma ili samo jednog atoma, dok RIF nema takvo ograničenje. U slučaju da je RIF izraz sastavljen od premise disjunktnih atoma, a zaključak od konjunkcije atoma, izravni prijevod nije moguć. Pa ipak, određenim transformacijama moguće je neizravno RIF izraze prevesti u SWRL pravila.

5.2.2 RIF sintaksa

Sintaksa RIF izraza u svojoj je osnovi slična sintaksi SWRL pravila, no u ljudski čitljivom obliku zapisa postoje razlike. RIF se izrazi, s ciljem postavljene razmjenjivosti, za potrebe računalne obrade i pohrane, pohranjuju u obliku XML zapisa, no postoje dva oblika sintakse predviđene za korištenje ljudi, pri čemu se one razlikuju minimalno i to stilski (W3C, 2012.). Pravilo oblika `ako A onda B` u prezentacijskoj sintaksi izražava se u obliku Kôd 5.3, a u apstraktnoj sintaksi u obliku Kôd 5.4.

```
B :- A
```

Kôd 5.3: B je logička posljedica od A.

```
If A Then B
```

Kôd 5.4: Ako A onda B.

RIF, za razliku od SWRL-a, podržava u svojim izrazima i konjunkciju i disjunkciju. Disjunkcija se označava ključnom riječi `Or`, dok je za konjunkciju ključna riječ `And`, kao što je to prikazano u RIF 1:

```
atomZaključka :- And (prviAtomPremise drugiAtomPremise)
```

RIF 1: *atomZaključka* logička je posljedica konjunkcije atoma premise.

Analogno izrazu RIF 1 slaže se izraz za disjunkciju korištenih atoma.

Korištenje individua u RIF izrazima identično je onom u SWRL sintaksi. Prilikom korištenja u nekom od svojstava, dovoljno je zapisati naziv tražene individue, kao u izrazu RIF 2:

```
Mentor(Ozano) :- And (
    Zaposlenik(Ozano)
    jeMentorZaposleniku(Ozano Iva)
    Zaposlenik(Iva)
)
```

RIF 2: *Ozano* je individua klase *Mentor* ako je individua klase *Zaposlenik* i sudjeluje u vezi *jeMentorZaposleniku* s individuom *Iva*, klase *Zaposlenik*.

RIF izrazi obavezno koriste univerzalni ili egzistencijalni kvantifikator. Pravilo poput sljedećeg:

Postoji klijent koji ima status Zlatnog klijenta.

moguće je u RIF sintaksi izraziti kao što slijedi u RIF 3:

```
Exists ?ZlatniKlijent (imaStatusKlijenta(?ZlatniKlijent
StatusKlijentaZlatni))
```

RIF 3: Postoji barem jedan klijent koji ima status *StatusKlijentaZlatni*.

Za razliku od SWRL pravila, koja su izravno povezana sa ontologijom koje su dio, u sklopu cjelovitih RIF dokumenata potrebno je označiti imenički prostor, s ciljem jedinstvene identifikacije korištenih ključnih riječi i naziva entiteta (W3C, 2012.). Imenički prostor u RIF dokumentu označava se na samom njegovom početku, kao i u RDF/OWL dokumentima, korištenjem ključne riječi `Prefix`, kako je prikazano u RIF 4:

```
Prefix(dipl <http://example.com/diplomskiPrimjer#>)
```

RIF 4: Prefiks *dipl* bit će zamjena za navedeni IRI.

Za valjani RIF dokument nužno su potrebna još dva izraza (W3C, 2012.): `Document` i `Group`. S obzirom da RIF dokument omogućava uvoz drugih dokumenata, potrebno je, iz praktičnih razloga, trenutni dokument oviti `Document` ključnom riječi. Slijedi izražavanje prefiksa te zatim pravila grupirana prema volji korisnika korištenjem ključne riječi `Group`. Grupiranje pravila istovjetno je izrazu `And (Pravilo1 Pravilo2 …)`. Stoga jedan dokument može imati veći broj grupa pravila koje mogu sadržavati po jedno ili više izraženih pravila. Dakako, grupiranje na ovaj način namijenjeno je prvenstveno ljudima koji čitaju sintaksu.

Uzevši u obzir sve navedeno, valjani RIF dokument koji sadrži pravilo RIF 2, izgledao bi kao što je to prikazano u izrazu RIF 5.

```
Document(
    Prefix (dipl <http://example.com/diplomskiPrimjer#>)
    Group(
       Forall ?Z1 ?Z2 (
          dipl:Mentor(?Z1) :- And (
                                  dipl:Zaposlenik(?Z1)
                                  dipl:jeMentorZaposleniku(?Z1 ?Z2)
                                  dipl:Zaposlenik(?Z2)
                                  )
        )
    )
)
```

RIF 5: Valjani RIF dokument s pravilom RIF 2.

Dakako, koncept određivanja prefiksa postaje puno korisniji u slučaju korištenja entiteta iz većeg broja imeničkih prostora.

Svojom je strukturom i osnovnom sintaksom zapis RIF 5 po pretpostavci dobar. No izražavanje pripadanja klasi RIF ne podržava na način na koji isto podržava SWRL, tj. na način na koji je to pokušano u navedenom RIF izrazu. Naime, za izražavanje pripadanja ekstenziji određene klase, RIF koristi izraz iz RDF imeničkog prostora, izražen ključnom riječi `type`, kao što je spomenuto ranije u radu. Prema tome, u navedeni RIF dokument potrebno je uvesti

nekoliko izmjena: dodatni imenički prostor te izmjenu u izražavanju pripadanju klasi. S obzirom na učestalost korištenja izraza koji označava pripadnost ekstenziji neke klase, RIF podržava skraćen zapis istog, korištenjem znaka `#` (W3C, 2012.). Pripadnost klasi korisno je izražavati u pravilima zbog činjenice da se time uvodi dodatno sigurnosno ograničenje u dano pravilo. Prema svemu ovome, u potpunosti valjan navedeni RIF dokument izgledao bi kako je prikazano u RIF 6:

```
Document(
    Prefix(rdf <http://www.w3.org/1999/02/22-rdf-syntax-ns#>)
    Prefix(rdfs <http://www.w3.org/2000/01/rdf-schema#>)
    Prefix (dipl <http://example.com/diplomskiPrimjer#>)
    Group(
        Forall ?Z1 ?Z2 (
            ?Z1 # dipl:Mentor :- And (
                ?Z1 # dipl:Zaposlenik
                dipl:jeMentorZaposleniku(?Z1 ?Z2)
                ?Z2 # dipl:Zaposlenik
                )
        )
    )
)
```

RIF 6: U potpunosti valjan RIF dokument.

Iznimno je korisna činjenica da je RIF u nekim područjima izražajniji od SWRL-a, pa je tako u RIF izrazu moguće odrediti da vrijednost nekog svojstva bude izmijenjena, dok SWRL omogućava isključivo dodavanje informacija na već postojeće. Izmjena činjenica zapisanih u bazi znanja označava se ključnom riječi `Modify`.

5.3 Postupak prevođenja

Jasno je da su poslovna pravila u svom izvornom obliku, stvorena za korištenje u poslovnom sustavu, izražena korištenjem riječi ljudima razumljivog jezika. Standardna sintaksa nekog od ljudskih jezika korištena je za izražavanje pravila u njihovom najjednostavnijem obliku, kao pravilo PP 5.1. Za uspješno prevođenje poslovnih pravila iz ovog prirodnog jezika u sustav koji koristi ontologiju i prateći jezik za izražavanje pravila, potrebno je proći određene korake prevađanja. U nastavku je u osnovnim crtama opisan postupak prevađanja.

> *PP 5.1: Student koji kroz kontinuirano praćenje, iz kolokvija, zadaća, seminara i projekta, sakupi više od 90 bodova, dobit će ocjenu izvrstan.*

Kao što je navedeno ranije u ovom radu, pravila izražena na upravo opisan način iznimno je korisno standardizirati korištenjem RuleSpeak smjernica, kako bi se ona svela na oblik koji zadržava izražajnost pravila i njegovo značenje, no uklanja mogućnost pogrešnog tumačenja tog značenja ili nekog dijela danog pravila. Standardizacija pravila na takav način predstavlja prvi od nekoliko koraka prevođenja pravila u oblik koji je glavna tema ovog rada. Zanimljivo je da već i takva standardizacija predstavlja određen oblik formalnog izraza pravila. Prema tome, pravilo PP 5.1 moguće je standardizirati uklanjanjem suvišnih podataka iz pravila, a ipak zadržavajući osnovno značenje:

> *PP 5.2: Više od 90 ukupno bodova studentu nosi ocjenu izvrstan.*

PP 5.3: Student sa više od 90 ukupno bodova ima ocjenu izvrstan.

Pravila PP 5.2 i PP 5.3 sadržajem su istovjetna, no razlikuje im se izraz. Iz ovog primjera jasno je vidljivo da se određeno pravilo može izraziti na više različitih načina, no pritom je iznimno važno paziti na sadržaj tog pravila.

U pravilima je zatim potrebno identificirati osnovne koncepte koji se koriste: subjekte, predikate i objekte. Subjekti i objekti najčešće su, kao i u standardnim rečenicama, imenice; dok su predikati najčešće glagoli. Pravilo PP 5.3 sadrži imenice `student`, `ukupno bodova` i `ocjena` te jedan ključni predikat `ima`. Drugi predikat dobiva se iz sklopa riječi „student sa […] ukupno bodova“, iz čega je moguće zaključiti da postoji još jedan predikat `ima`.

Imenice i predikati izvađeni iz pravila predstavljaju koncepte i entitete konačnog sustava te ih je kao takve potrebno točno odrediti u modelu činjenica. Definiranjem svakog entiteta u modelu činjenica osigurava se jednoznačnost svakog pravila i pravilno korištenje tih entiteta kroz sva pravila određene domene.

Uobičajeno je da imenice (subjekti i objekti) modela činjenica budu prevedeni u klase ontologije konačnog sustava pravila. Pa ipak, već pravilo PP 5.3 pokazuje suprotno, zato što će biti postignut bolji sustav ukoliko činjenica da `student ima određen broj bodova` bude pretvorena u podatkovno svojstvo koje individuu klase `Student` povezuje sa običnim cjelobrojnim brojem, nego da broj bodova bude određen kao klasa te sa klasom `Student` povezan objektnim svojstvom `ima`. Glagoli modela činjenica prevode se u svojstva unutar ontologije koja podržava sustav izražavanja poslovnih pravila.

U konačnici, kao u do sad prikazanim primjerima, kao i primjeru koji slijedi, određena pravila izražena su korištenjem OWL izraza, dok su neka složenija prevedena u SWRL izraze.

6 Praktični primjer

6.1 Uvod u primjer

Ranije u ovom radu navedeni su određeni koncepti koji se povezuju sa prevađanjem poslovnih pravila iz specifičnih primijećenih aktivnosti poslovnog sustava u riječi i, na kraju, u svojevrsni krajnje formalizirani oblik čitljiv računalu. Ovdje valja spomenuti korake tog prevađanja.

Prevođenje poslovnih pravila u formalizirane izraze iskazano je u (Ceravalo, et al., 2007) kroz četiri koraka:

1. Prepoznavanje simbola rječnika i modela činjenica, gdje se prepoznaju riječi unutar poslovnog pravila i njihovo značenje;
2. Analiza temeljem jezičnih pravila, kojima se utvrđuju imena, osnovni koncepti, glagoli i ključne riječi te njihov međusobni odnos;
3. Izražavanje korištenjem logičkih činjenica;
4. Iskaz logičkih činjenica poput objekata;
5. Zapis objekata u XML obliku.

Navedeni koraci osiguravaju prevođenje poslovnog pravila u XML format kojim se gotovo u potpunosti uvodi mogućnost izražaja koji je neovisan o sustavu u kojem je izražen ili drugim faktorima.

Za sljedeće primjere stoji pretpostavka da su (poslovna) pravila, u poslovnom sustavu, izražena gotovo razgovornim jezikom. Neovisno o izražaju, činjenica je da pravila postoje i nema potrebe provoditi postupke njihovog dobivanja iz analize poslovnih procesa ili drugih izvora. Uz takvu pretpostavku, moguće je postupak formaliziranja pravila i njihovo prevođenje u SWRL pravila te dalje korištenje istih, opisati sljedećim koracima, koji će biti korišteni u primjerima:

1. **Analiza sadržaja izraza pravila**, koji obuhvaća analizu simbola korištenih u izrazu pravila te njihovo kategoriziranje u imenice, glagole, imena, koncepte, ključne riječi i sl.;
2. **Prepoznavanje simbola i njihovih odnosa**, što uključuje analizu simbola i njihovog značenja te pronalaženje logičkih cjelina izraza pravila;
3. **Izražavanje poslovnih pravila u formaliziranom obliku**, nužno je za uniformirane jednoznačne izraze, a neizostavno uključuje:
    - Kombiniranje simbola u standardizirane izraze (koncepte modela činjenica), čime se stvaraju temelji izraza za korištenje u kasnijim koracima;
    - Ažuriranje modela činjenica, ukoliko neki od pronađenih simbola ili logičkih izraza nisu uvedeni u postojeći model činjenica, čime nije poznato njihovo standardizirano značenje;
4. **Izražavanje i zapis pravila u odabranom računalu shvatljivom jeziku**, u daljnjim primjerima bit će korišten SWRL, kao jezik odabran za ovaj rad.

6.2 Primjer

Proizvodi jedne organizacije prate najnovije trendove na području tehnologije dostupne u svijetu, a usklađeni su sa sveprisutnim ekološkim načelima te vrhunskim dizajnom koji odiše kvalitetom, jednostavnošću i ljepotom prirode. Ista Organizacija svoje proizvode prodaje određenim klijentima. Prodaja je krenula tek nedavno, no rezultati su zadovoljavajući. Kako bi pokazali da brinu o svojim klijentima, Organizacija je odlučila uvesti najjednostavniji oblik

praćenja lojalnosti klijenata, pa će prema ukupno potrošenom iznosu pojedinog klijenta dodijeliti mu jednu od tri kategorije, koje sa sobom donose i određen popust pri svakoj sljedećoj kupnji.

Poslovna pravila u ovom slučaju određivat će uvjete koje kupac mora zadovoljiti da bi ušao u neku od kategorija predviđenih za klijente, kao i pravila kojima je određen način računanja određenih iznosa te još neke sitnice.

U Organizaciji određena su sljedeća pravila za kategorizaciju klijenata:

> *PP 6.1: Klijent koji kod nas potroši do 500 kn nema pravo ni na kakav popust pri kupovini.*
>
> *PP 6.2: Klijent koji kod nas potroši do 2000 kn ima odobren popust do 10%, ostvariv pri svakoj budućoj kupovini.*
>
> *PP 6.3: Klijentu s potrošenih 2000 kn odobrava se popust u iznosu od 20%.*
>
> *PP 6.4: Klijent koji kod nas potroši ukupno 3000 kn ima pravo na popust u iznosu od 30%.*

Sva četiri navedena pravila u osnovnom svom značenju, iako različitog izraza, imaju poruku da klijent koji u povijesti svoje kupovine kod Organizacije potroši određen iznos, ima pravo na kupovinu uz određen popust. Takva je osnovna poruka jasna iz svakog od navedena četiri pravila, pa ipak se njihovi izrazi uvelike razlikuju. Problem interpretacije i različitih mogućih značenja pravila proizlazi iz nestandardiziranosti i korištenja nejednoznačnih izraza. Pa ipak, razgovorno izražena pravila očekivano nisu standardizirana, zbog same prirode izražaja, što pogoduje ovom primjeru zbog koraka koje je potrebno provesti, a koji su navedeni ranije.

Na izraz poslovnog pravila izravno utječu prva tri od spomenuta četiri pravila. Njihovom primjenom pravilo PP 6.1 bit će prevedeno u standardizirani, formalizirani, oblik.

Pravilo PP 6.1

Za lakše praćenje pravila i njegove preobrazbe korisno je opet ga imati zapisanog ovdje:

> *PP 6.1: Klijent koji kod nas potroši do 500 kn nema pravo ni na kakav popust pri kupovini.*

Iz ovog konkretnog pravila moguće je zaključiti sljedeće, gdje podcrtane riječi označavaju cjeline koje su najbitnije za daljnje razmatranje:

- Klijentima se <u>odobrava popust</u>;
- Odobreni popust ovisi o <u>kategoriji klijenta</u>;
- Uvjet za jednu kategoriju je potrošenih <u>do 500 kn</u>.

Riječi ili sintagme moguće je iz pravila izdvojiti kako slijedi u Tablica 6.1:

RIJEČ/SINTAGMA	ULOGA	POJAŠNJENJE
klijent	*subjekt pravila*	*fizička ili pravna osoba koja ulazi u poslovni odnos s Organizacijom*
potrošiti do 500 kn	*pobliže označuje subjekt*	*potrošenim se smatra iznos koji je klijent platio Organizaciji, a u sklopu svojih prošlih financijskih obveza prema Organizaciji, proizašlih iz ispostavljenih računa*

imati pravo	*predikat pravila*	*ima pravo na korištenje popusta koji je dodatno određen ovim pravilom*
popust pri kupovini	*objekt pravila*	*popust se odnosi na dani račun u dano vrijeme te je određen isključivo kategorijom klijenta*
nikakav	*pobliže označuje objekt*	*jednak nuli, nema ga*

Tablica 6.1: Analiza simbola pravila PP 6.1. (izvor: izrada autora)

Analizom pravila utvrđeni su njegovi bitni elementi i njihove uloge unutar samog pravila, što omogućuje stvaranje standardiziranog pravila, korištenjem elemenata modela činjenica, a koji odgovaraju pojašnjenju svakog pojma.

Pretpostavka je da se u modelu činjenica nalaze, između ostalog, zapisani koncepti kao što je navedeno u Tablica 6.2:

KONCEPT	POJAŠNJENJE
Klijent	*Fizička ili pravna osoba koja u bilo kojem vremenskom trenutku ulazi u poslovni odnos s Organizacijom*
Popust	*Postotna vrijednost koja umanjuje konačan iznos računa prema klijentu u trenutku njegovog ispostavljanja*
klijent ima iznos	*Numerička vrijednost koja označava ukupan iznos koji je Organizacija primila od klijenta temeljem računa ispostavljenih danom klijentu do danog trenutka*

Tablica 6.2: Isječak modela činjenica promatrane organizacije. (izvor: izrada autora)

Korištenjem Tablica 6.2, kao izvora simbola (koncepata) koje je dozvoljeno koristiti u standardiziranom izražaju pravila, zatim Tablica 6.1, kao rezultata analize početnog izražaja promatranog pravila, i, u konačnici, samog promatranog pravila (PP 6.1), moguće je sastaviti standardizirani oblik danog pravila, koji je svojim sadržajem jednak izvornom pravilu, no izrazom odgovara standardima poslovnog pravila i modela činjenica. Standardizirano pravilo PP 6.1 izraženo je pravilom PP 6.5 kako slijedi:

PP 6.5: Klijent koji ima iznos manji od 500 kn, ima odobren popust koji iznosi 0%.

Uzimajući u obzir pravilo PP 6.5, koje je svojim izrazom u zadovoljavajućoj mjeri standardizirano, a svojim se sadržajem ne razlikuje od pravila PP 6.1, obavljeno je prevođenje pravila u standardiziran, formaliziran, oblik. Simboli izraza pravila PP 6.5 jasni su i jednoznačni, čemu doprinosi ranije spomenut model činjenica. S ciljem dodatnog osiguravanja jednoznačnosti pravila, u model činjenica korisno je dodati jedan koncept, kako slijedi u Tablica 6.3:

KONCEPT	POJAŠNJENJE
klijent ima odobren popust	*Činjenica da će klijentu u sklopu svakog budućeg ispostavljenog računa biti obračunat popust određene vrijednosti*

Tablica 6.3: Dodatak modelu činjenica promatrane organizacije. (izvor: izrada autora)

S ciljem utvrđivanja svega navedenog u postupku prevađanja pravila, slijedi primjer prijevoda pravila PP 6.3.

Pravilo PP 6.3

PP 6.3: Klijentu s potrošenih 2000 kn odobrava se popust u iznosu od 20%.

Iz navedenog je pravila moguće zaključiti sljedeće, gdje podcrtane riječi ponovno označavaju svojevrsne logičke cjeline:

- Klijentima se odobrava popust;
- Odobreni popust ovisi o kategoriji klijenta;
- Uvjet za jednu kategoriju je potrošenih 2000 kn.

Riječi ili sintagme moguće je iz pravila izdvojiti kako slijedi u Tablica 6.4:

RIJEČ/SINTAGMA	ULOGA	POJAŠNJENJE
klijent	*subjekt pravila*	*fizička ili pravna osoba koja ulazi u poslovni odnos s Organizacijom*
potrošenih 2000 kn	*pobliže označuje subjekt*	*potrošenim se smatra iznos koji je klijent platio Organizaciji, a u sklopu svojih prošlih financijskih obveza prema Organizaciji, proizašlih iz ispostavljenih računa*
odobriti	*predikat pravila*	*odobrava se popust na računu pri njegovom ispostavljanju*
popust	*objekt pravila*	*popust se odnosi na dani račun u dano vrijeme te je određen isključivo kategorijom klijenta*
iznos od 20%	*pobliže označuje objekt*	*iznos je jednak točno 20%*

Tablica 6.4: Analiza simbola pravila PP 6.3. (izvor: izrada autora)

Baš kao i u prethodnom prevođenju pravila, analizom ovog pravila određeni su njegovi bitni elementi, njihove uloge i njihov odnos, čime je prijevod, tj. standardizirani oblik promatranog pravila, korak bliže.

Pretpostavka u prijevodu ovog pravila je da je poznat model činjenica spomenut u postupku prevođenja iznad, a koji sadrži elemente navedene u Tablica 6.5:

KONCEPT	POJAŠNJENJE
Klijent	*Fizička ili pravna osoba koja u bilo kojem vremenskom trenutku ulazi u poslovni odnos s Organizacijom*
Popust	*Postotna vrijednost koja umanjuje konačan iznos računa prema klijentu u trenutku njegovog ispostavljanja*
klijent ima iznos	*Numerička vrijednost koja označava ukupan iznos koji je Organizacija primila od klijenta temeljem računa ispostavljenih danom klijentu do danog trenutka*
klijent ima odobren popust	*Činjenica da će klijentu u sklopu svakog budućeg ispostavljenog računa biti obračunat popust određene vrijednosti*

Tablica 6.5: Isječak modela činjenica promatrane organizacije. (izvor: izrada autora)

U konačnici, korištenjem modela činjenica i početnog oblika izraza pravila PP 6.3, prijevod završava standardiziranim izrazom pravila:

> *PP 6.6: Klijent koji ima iznos veći od ili jednak 2000 kn i manji od 3000 kn, ima odobren popust koji iznosi 20%.*

Prilikom prevođenja promatranog pravila korištena je pretpostavka da su poznate granice postavljene ostalim pravilima. Korištenjem takve pretpostavke uvedeno je dodatno ograničenje, koje nije postojalo u početnom izrazu pravila, no proizlazi iz sva četiri početno postavljena pravila.

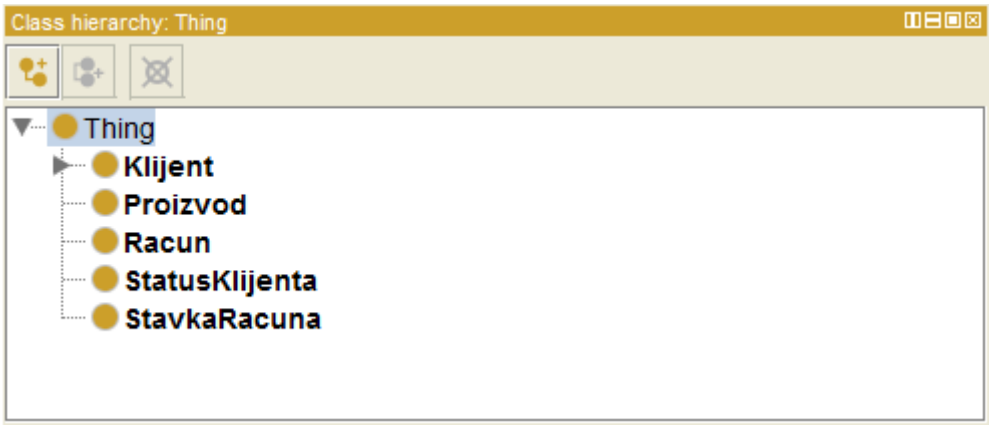
Za potrebe daljnjeg razmatranja primjera, dodaje se proširenje cijelog sustava dodjeljivanja popusta klijentima tako što Organizacija svoje klijente, s ciljem lakše dodatne manipulacije, razdjeljuje u četiri kategorije – zlatnu, srebrnu, brončanu i bez kategorije. Navedene kategorije, tzv. statusi klijenata, kreću se u skladu s pravilima PP 6.1 - PP 6.4, tako što se zlatni status dodjeljuje klijentima s najvećim potrošenim iznosom, a brončani status klijentima s najmanjim potrošenim iznosom, od iznosa spomenutih u navedenim pravilima.

Računalni oblik

Standardizirani izraz pravila, iako konačan cilj u kontekstu izražavanja pravila, međukorak je u prevođenju pravila iz neformalnog izraza u SWRL izraz koji računalo može koristiti za zaključivanje nad ontologijom. Ontologija u ovom smislu predstavlja svojevrsni prošireni računalni zapis modela činjenica. Stoga bi moglo biti korisno u modelu činjenica, uz pojašnjenje određenog koncepta, iskazati i entitet kojim je isti prikazan u povezanoj ontologiji.

Ovaj primjer izrađen je korištenjem ontologije stvorene alatom Protege 4.2.0, alatom za zaključivanje nad ontologijom Pellet 2.3.0 te razvojnim okruženjem Django.

Za početak, imenice, odnosno koncepti koji predstavljaju imenice koje se mogu shvatiti kao klase koje u stvarnom korištenju imaju individue koje im pripadaju, definirane su kao klase promatrane ontologije. Kao što se vidi na Slika 6.1, postoji klasa za koncept klijenta, proizvoda, računa, statusa klijenta i stavke računa. Sukladno navedenom savjetu, model činjenica koji se bavi navedenim konceptima izgledao bi kao u Tablica 6.6. Nadalje neće biti navođeni izvodi modela činjenica koji bi pobliže pojašnjavali korištene koncepte.



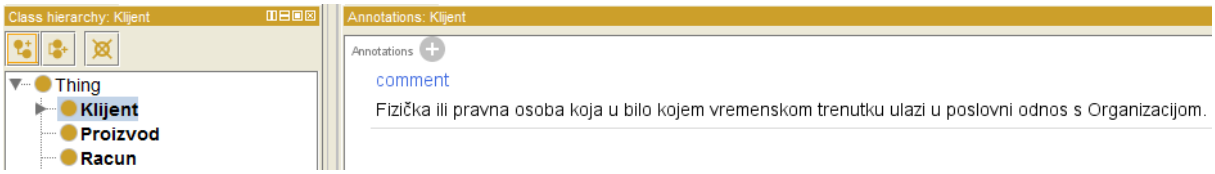
Slika 6.1: Klase promatrane ontologije. (izvor: izrada autora)

KONCEPT	POJAŠNJENJE	ENTITET
Klijent	...	Klijent

Proizvod	...	Proizvod
Račun	...	Racun
Status klijenta	...	StatusKlijenta
Stavka računa	...	StavkaRacuna

Tablica 6.6: Predloženo proširenje modela činjenica, bez navođenja pojašnjenja koncepata s ciljem smanjenja tablice. (izvor: izrada autora)

Kao što je napomenuto ranije, entiteti gdje je to moguće, potrebno ili korisno, mogu unutar ontologije biti dodatno opisani komentarom. Pritom je prijedlog da komentar, ukoliko je to moguće, sadrži pojašnjenje navedeno u modelu činjenica, kao što je prikazano na Slika 6.2:



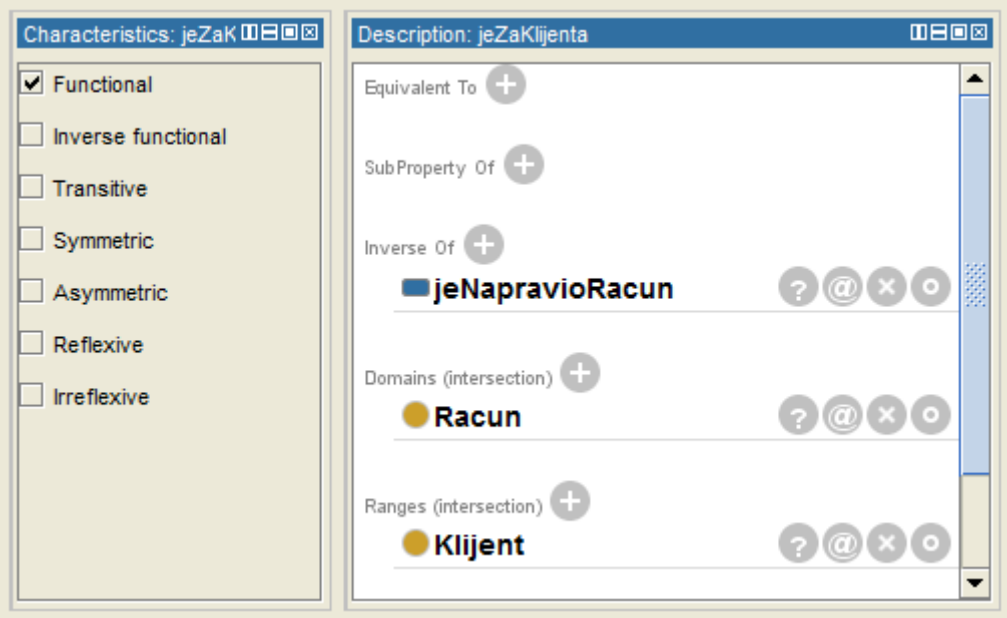
Slika 6.2: Komentar uz klasu *Klijent* promatrane ontologije. (izvor: izrada autora)

Ovisno o mogućem načinu korištenja, koncepti iskazani glagolima i sintagmama s glagolima, u svoje se računalne oblike prevađaju kao podatkovna ili objektna svojstva. Pritom je potrebno ponoviti da podatkovna svojstva zahtijevaju određeni zapis u obliku niza znakova, broja ili nekog trećeg tipa podataka (generalno literala), dok objektna svojstva povezuju dva entiteta ontologije. Objektna svojstva ove ontologije bave se statusom klijenta, dodjeljuju stavku računu, ukazuju na račune pojedinog klijenta te povezuju stavku računa s određenim proizvodom. Sva navedena svojstva, uz nekoliko dodatnih, prikazana su na Slika 6.3:

Slika 6.3: Objektna svojstva promatrane ontologije. (izvor: izrada autora)

Svojstva koja su dodana u Slika 6.3, a nisu upravo navedena su inverzna nekima od navedenih svojstava. Tako je svojstvo `jeZaKlijenta` inverzno svojstu `jeNapravioRacun`, pa oba svojstva služe identifikaciji računa koji glase na nekog klijenta, odnosno identifikaciji klijenta na kojeg glasi neki račun. Dodatnim ograničenjima svojstva `jeZaKlijenta` određeno je da je domena tog svojstva sastavljena od individua klase `Racun`, a doseg od individua klase `Klijent`.

Osim toga, ovo je svojstvo funkcionalno, što se jasno vidi u definiciji svojstva prikazanoj na Slika 6.4.



Slika 6.4: Ograničenja i karakteristike svojstva *jeZaKlijenta*. (izvor: izrada autora)

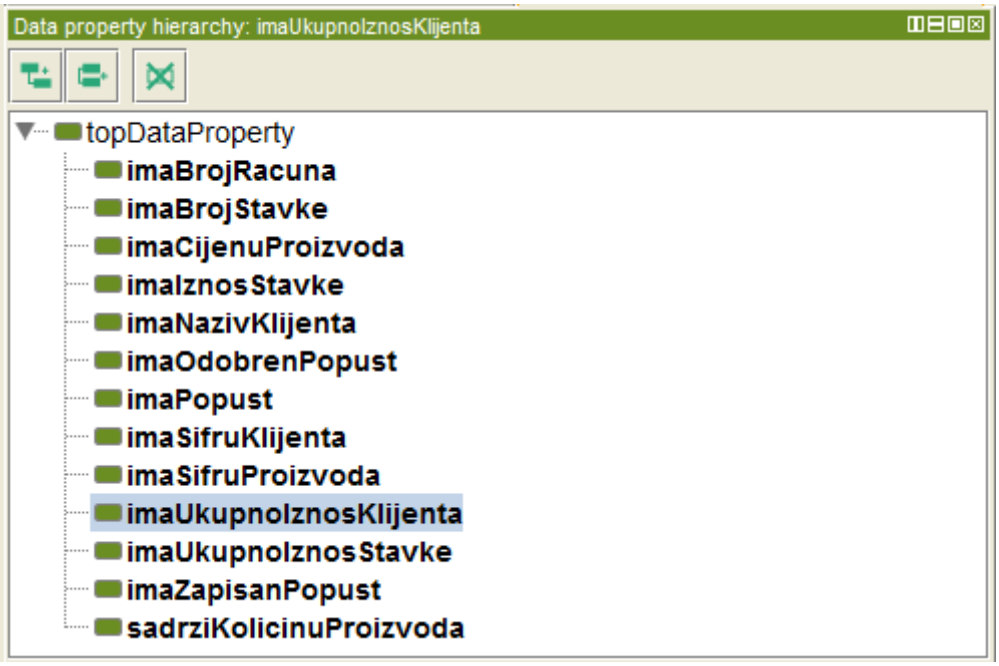
Svojstvo `jeZaKlijenta` i `jeNapravioRacun` u OWL/RDF datoteci izražena su kao u Kôd 6.1, gdje se jasno opet vide domena i doseg svojstva, funkcionalnost i inverzno svojstvo. Bitno je ovdje napomenuti korištenje imeničkog prostora koji će cijelim ovim primjerom biti „http://www.dipl.edu/primjer2“.

```
<owl:ObjectProperty
rdf:about="http://www.dipl.edu/primjer2#jeNapravioRacun">
    <owl:inverseOf
rdf:resource="http://www.dipl.edu/primjer2#jeZaKlijenta"/>
</owl:ObjectProperty>

<owl:ObjectProperty
rdf:about="http://www.dipl.edu/primjer2#jeZaKlijenta">
    <rdf:type rdf:resource="&owl;FunctionalProperty"/>
    <rdfs:range
rdf:resource="http://www.dipl.edu/primjer2#Klijent"/>
    <rdfs:domain
rdf:resource="http://www.dipl.edu/primjer2#Racun"/>
</owl:ObjectProperty>
```

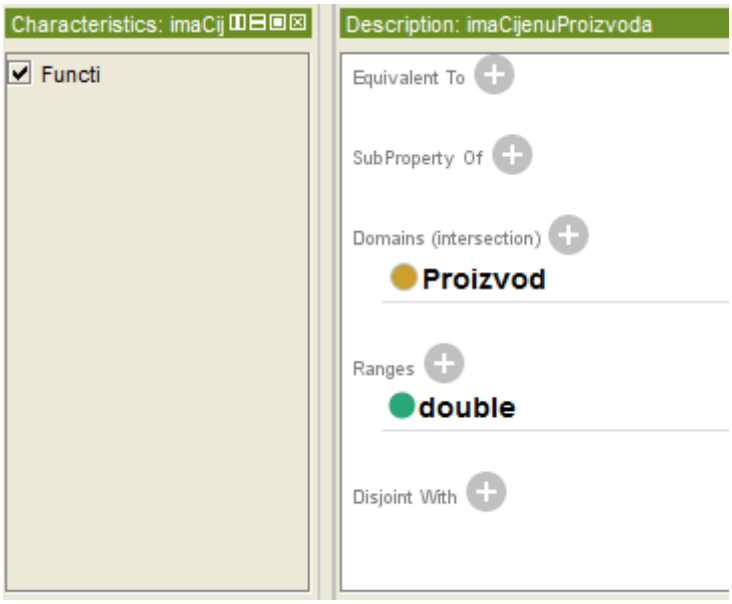
Kôd 6.1: XML zapis objektnih svojstava *jeNapravioRacun* i *jeZaKlijenta*.

Podatkovna su svojstva u ovoj ontologiji i ovoj konkretnoj interpretaciji promatrane domene nešto brojnija, a sva su prikazana na Slika 6.5 koja slijedi:



Slika 6.5: Podatkovna svojstva promatrane ontologije. (izvor: izrada autora)

Sva su podatkovna svojstva ograničena sadržajem domene, dok su neka ograničena i tipom podataka kojem mora pripadati vrijednost koju dano svojstvo pridružuje svom subjektu. Tako je svojstvo `imaCijenuProizvoda` ograničeno na domenu sastavljenu od individua klase `Proizvod`, a domena na vrijednosti tipa `double`. Štoviše, i ovo je funkcionalno svojstvo, što se jasno vidi u Slika 6.6 ispod.



Slika 6.6: Ograničenja i karakteristika svojstva *imaCijenuProizvoda*. (izvor: izrada autora)

U dokumentu ontologije, ovo je svojstvo izraženo kao u Kôd 6.2:

```
<owl:DatatypeProperty
rdf:about="http://www.dipl.edu/primjer2#imaCijenuProizvoda">
```

```
    <rdf:type rdf:resource="&owl;FunctionalProperty"/>
    <rdfs:comment>Cijena proizvoda izražena prirodnim
brojem</rdfs:comment>
    <rdfs:domain
rdf:resource="http://www.dipl.edu/primjer2#Proizvod"/>
    <rdfs:range rdf:resource="&xsd;double"/>
</owl:DatatypeProperty>
```

Kôd 6.2: XML zapis objektnih svojstva *imaCijenuProizvoda*.

Konkretne instance navedenih klasa u ontologiji su izražene individuama pripadajuće klase, pa postoje individue klase `Klijent` koje predstavljaju određene klijente, zajedno s njihovim nazivom, šifrom i iznosom koji pojedini klijent ima. Svaki je klijent u svojoj osnovi izražen korištenjem triju svojstava: `imaUkupnoIznosKlijenta`, `imaNazivKlijenta` i `imaSifruKlijenta`. Svojstvo `imaSifruKlijenta` korišteno je kao ključno svojstvo klase `Klijent`, što u procesu zaključivanja osigurava jednoznačnost svake individue, tj. implikaciju da su dvije individue iste (jednake, predstavljaju istu instancu u promatranom svijetu), ukoliko su im iste vrijednosti pridružene navedenim svojstvom. Definicija jedne individue klase `Klijent` prikazana je u Kôd 6.3.

```
<owl:NamedIndividual
rdf:about="http://www.dipl.edu/primjer2#klnt1">
    <rdf:type rdf:resource="http://www.dipl.edu/primjer2#Klijent"/>
    <imaUkupnoIznosKlijenta
rdf:datatype="&xsd;double">1464.2</imaUkupnoIznosKlijenta>
    <imaNazivKlijenta>Count von Count</imaNazivKlijenta>
    <imaSifruKlijenta>K0001</imaSifruKlijenta>
</owl:NamedIndividual>
```

Kôd 6.3: XML zapis individue klase *Klijent*.

Postavljanjem individua klase `Klijent` omogućeno je zaključivanje temeljem pravila koja su prevedena ranije u primjeru. Ta pravila, korištenjem entiteta ontologije, koji se koriste prema uputama iz modela činjenica, a u skladu sa standardiziranim izrazima pravila, moguće je u SWRL prevesti kao što će biti prikazano u nastavku, za zlatni status klijenta.

Poslovno pravilo

> *PP 6.7: Klijentu koji ima iznos veći od ili jednak 3000 kn, pripada status zlatnog klijenta.*

moguće je u SWRL prevesti korištenjem sljedećih elemenata:

- klase `Klijent`,
- svojstva `imaUkupnoIznosKlijenta` (koje se koristi za koncept modela činjenica „klijent ima iznos“),
- SWRL elementa za uspoređivanje brojeva,
- klase `'zlatni klijent'`.

Klase statusa klijenta nisu do sad bile spomenute, a služe za dodatno klasificiranje klijenata prema vrijednosti svojstva `imaUkupnoIznosKlijenta`. Ovaj naizgled nepotreban korak u određivanju statusa i, konačno, odobrenog popusta klijenta, može biti koristan u dodatnoj nadogradnji sustava praćenja lojalnosti klijenata.

Izraženo u SWRL obliku, pravilo PP 6.7 izgleda kao u SWRL 6.1:

Klijent(?k), imaUkupnoIznosKlijenta(?k, ?i), greaterThanOrEqual(?i, 3000) -> 'zlatni klijent'(?k)

SWRL 6.1: Klijent koji ima iznos veći od ili jednak 3000, pripada klasi *'zlatni klijent'*.

Komentiranje nije jedini način neobaveznog dodatnog opisivanja entiteta ontologije, već postoji i mogućnost postavljanja oznaka za određene entitete. Oznaka[18] je postavljena i za klasu koja objedinjuje individue klase `Klijent`, a kojima pripada status zlatnog klijenta, pa je stoga, i zbog činjenice da je za oznaku korišteno više odvojenih riječi, naziv klase naveden unutar znakova navodnika.

Svaki klijent Organizacije koji uspije postići status zlatnog klijenta ima odobren popust u iznosu od 30%. Ovakvo pravilo nadopunjuje ono upravo prevedeno te zajedno na kraju tvore jedno od pravila s početka razmatranja ovog primjera.

Iznimno slično prevođenju pravila PP 6.7, pravilo PP 6.8:

PP 6.8: Klijent statusa zlatnog klijenta ima odobren popust koji iznosi 30%.

izraženo u SWRL obliku glasi:

'zlatni klijent'(?k) -> imaOdobrenPopust(?k, 0.3)

SWRL 6.2: Zlatnom klijentu odobren je popust od 30%.

U SWRL 6.2 moguće je primijetiti da je iznos popusta izražen u decimalnom obliku, a ne u postotnom obliku, no udaljavanje od pojašnjenja iz modela činjenica proizlazi iz lakše manipulacije podacima kasnije u kodu.

Postupak dodavanja novog računa najbolje prikazuje navedena pravila u primjeni, kao i primjenu dodatnih pravila korištenih za zaključivanje neunesenih podataka, a koji su svi potrebni za konačno uspješno korištenje ontologije.

Za svaki novi račun Organizacije korisnik nužno mora odabrati klijenta na kojeg se odnosi taj isti račun, kao što je prikazano na Slika 6.7 dolje.

Slika 6.7: Prikaz osnovnih podataka klijenta za račun. (izvor: izrada autora)

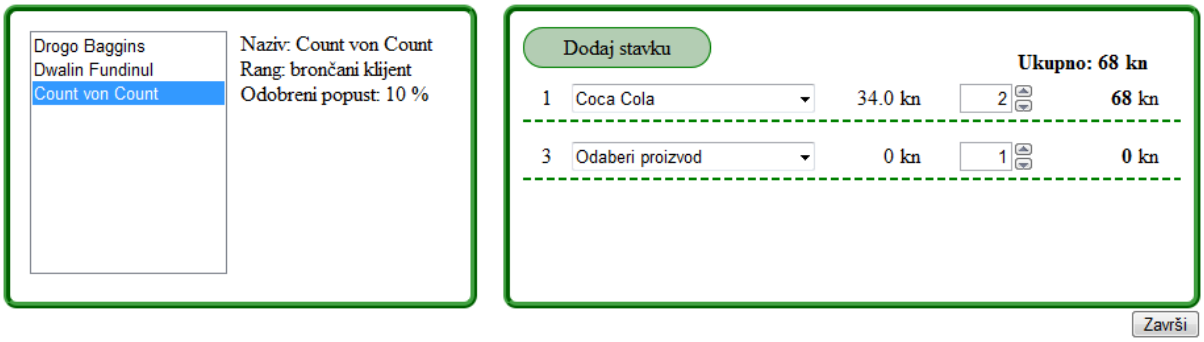
[18] eng. *label*

Korisnik dodavanjem stavke dobiva mogućnost odabira proizvoda iz palete dostupnih proizvoda Organizacije, pregled njegove cijene[19], unos tražene količine tog proizvoda te uvid u ukupni iznos stavke, kao što je prikazano na Slika 6.8.

Slika 6.8: Dodana stavka računa. (izvor: izrada autora)

Stavke računa moguće je brisati korištenjem „skrivenog“ gumba koji se nalazi na području rednog broja stavke. Tako su, prije brisanja, trenutnom računu pridružene tri stavke (Slika 6.9), dok su brisanjem stavke rednog broja 2 ostale samo dvije stavke, ona rednog broja 1 i ona rednog broja 3 (Slika 6.10).

Slika 6.9: Stavke trenutnog računa prije brisanja stavke redni broj 2. (izvor: izrada autora)



Slika 6.10: Stavke trenutnog računa nakon brisanja stavke redni broj 2. (izvor: izrada autora)

Završetkom obrade računa započinje proces zapisivanja cijelog računa, koji se sastoji od nekoliko koraka:

[19] Svaka sličnost sa stvarnim vrijednostima je slučajna.

1. zapis stavki računa,
2. zapis samog računa,
3. ažuriranje iznosa klijenta.

Stavke računa, sa proizvodom i količinom, prenose se u jednom python rječnik tipu podataka, a u ontologiju se zapisuju korištenjem python `rdflib` biblioteke koja olakšava rad s RDF datotekama, kao u Kôd 6.4:

```
for k in stavke.keys():
    stavka = rdflib.URIRef(stavke[k][2])
    g.add((stavka, RDF['type'], PR['StavkaRacuna']))
    g.add((stavka, RDF['type'], OWL['NamedIndividual']))
    g.add((stavka, PR['sadrziProizvod'],
rdflib.URIRef(stavke[k][0])))
    g.add((stavka, PR['sadrziKolicinuProizvoda'],
rdflib.Literal(int(stavke[k][1]), datatype=XSD.int)))
```

Kôd 6.4: Zapisivanje stavki računa u ontologiju.

Nešto pojašnjenja Kôd 6.4 je potrebno. `rdflib` biblioteka omogućava čitanje i pisanje RDF dokumenata korištenjem python jezika. Varijabla `stavka` predstavlja entitet koji će biti dodan u RDF dokument. `g.add` ovdje omogućava dodavanje RDF trojki u već postojeći niz RDF zapisa koji su dobiveni ranijim učitavanjem trenutno promatrane ontologije. Prema tome, izraz `g.add((stavka, RDF['type'], PR['StavkaRacuna']))` umeće u postojeći RDF dokument RDF trojku gdje je novi entitet stavka subjekt trojke, predikat je entitet naziva `type`, koji je dio `RDF` imenskog prostora, a objekt je entitet (u ovom slučaju klasa) naziva `StavkaRacuna`, koji je dio imenskog prostora ovog primjera, označenog prefiksom `PR`. Prilikom umetanja RDF trojke u kojoj je objekt nekog točno određenog tipa podataka, taj je tip potrebno posebno naznačiti, kao u

```
g.add((stavka, PR['sadrziKolicinuProizvoda'],
rdflib.Literal(int(stavke[k][1]), datatype=XSD.int)))
```

gdje je tip podatka (cjelobrojni broj) označen korištenjem `datatype=XSD.int`.

Analogno umetanju individua stavki računa, provodi se umetanje individua klase `Racun`, dok se za ažuriranje vrijednosti koristi Kôd 6.5:

```
g.set((rdflib.URIRef('http://www.dipl.edu/primjer2#' + klijent),
PR['imaUkupnoIznosKlijenta'],
rdflib.Literal(ansJSON['results']['bindings'][0]['ukupno']['value']
, datatype=XSD.double)))
```

Kôd 6.5: Ažuriranje vrijednosti iznosa koju ima klijent.

Korištenjem naredbe `set` umjesto `add` postojeća se RDF trojka iz RDF dokumenta briše te se unosi nova trojka kako je to određeno u kodu. Takva je praksa pozitivna iz očitog razloga potencijalnog sukoba i nekonzistentne ontologije u slučaju pokušaja dodavanja još jednog svojstva koje je u svojoj osnovnoj karakteristici funkcionalno.

Zanimljivo je ovdje primijetiti da se prilikom zapisivanja u sklopu ovog primjera za stavke računa zapisuje samo proizvod na koji se stavka odnosi i njegova količina. Nadalje, prilikom unosa računa zapisuje se samo njegov broj (identifikacijsko svojstvo), pripadajuće stavke, klijent kojem će dani račun biti ispostavljen te iznos popusta koji je odobren uzimajući u obzir status klijenta. U konačnici, klijentu se samo ažurira ukupan iznos koji isti ima. Dakako, u oba se zapisa u RDF dokument zapisuje i klasa kojoj dana individua pripada.

Dodatni podaci, poput ukupnog iznosa stavki, pa samim time i računa te broj stavke (identifikacijsko svojstvo), zaključuju se postupkom zaključivanja nad promatranom ontologijom, a na temelju zapisanih SWRL pravila. Tako se broj stavke računa određuje kao spoj šifri računa i proizvoda koji je uključen u danu stavku, izraženo sljedećim SWRL pravilom:

Racun(?r), StavkaRacuna(?s), jeDioRacuna(?s, ?r), sadrziProizvod(?s, ?p), imaBrojRacuna(?r, ?brr), imaSifruProizvoda(?p, ?sf), stringConcat(?brs, "R", ?brr, "P", ?sf) -> imaBrojStavke(?s, ?brs)

SWRL 6.3: Pravilo kojim je određen način stvaranja šifre stavke.

Slično navedenom, ukupni iznos stavke, dobiven umanjenjem iznosa stavke računa za odobreni popust klijenta, računa se sljedećim SWRL izrazom. Zanimljivo je ovdje primijetiti da se popust primjenjiv na stavku povlači iz vrijednosti popusta pridruženoj računu kojem dana stavka pripada.

StavkaRacuna(?s), jeDioRacuna(?s, ?r), imaIznosStavke(?s, ?i), imaPopust(?r, ?pp), multiply(?uk, ?p, ?i), subtract(?p, 1, ?pp) -> imaUkupnoIznosStavke(?s, ?uk)

SWRL 6.4: Pravilo kojim je određen način izračuna ukupnog iznosa stavke.

Navedenim relativno jednostavnim primjerom prikazana je kompleksnost prevođenja pravila korištenih u poslovnom sustavu, izraženih u nestandardiziranom obliku, do pravila shvatljivih računalnom sustavu koji ih može koristiti za kompleksne zadaće zaključivanja temeljem pripadajuće ontologije i pravila. Usprkos činjenici da je opisani primjer jednostavan i iznimno ograničenog opsega u kojem pokriva situacije poslovnog sustava, zorno prikazuje mogućnosti koje predstavljaju poslovna pravila izražena u obliku koji je pogodan za računalno zaključivanje. Osim toga, primjerom su prikazane neke od opisanih mogućnosti OWL zapisa, kao i SWRL zapisa.

7 Zaključak

U sadržaj ovog rada uključeni su koncepti iz poslovnog svijeta i koncepti iz svijeta informacijskih tehnologija te je prikazan način na koji se te dvije naizgled različite sfere mogu preklopiti u području gdje, isključivo zajedno, mogu postići kvalitetniju suradnju. Poslovna pravila čine temelje poslovnih procesa, pa time i poslovanja neke organizacije, zbog čega je važna njihova jednoznačnost i laka shvatljivost. Ontologije su modeli znanja zapisani tako da ih računala mogu razumjeti i zaključivati temeljem zapisa koji u njima postoje. Zajedno, ontologije dodatno određene poslovnim pravilima, predstavljaju sklop koji uvodi računalnu moć zaključivanja u područje poslovanja poslovnog sustava.

Takav spoj svoju svrhu može pronaći u velikim organizacijama gdje standardizirano poslovanje omogućava brže donošenje odluka, provođenje poslovnih procesa i poslovanje općenito. Sustavi gdje se temeljem određenih ulaznih podataka korisniku vraćaju informacije koje ovise isključivo o dobro utemeljenim poslovnim pravilima, koja proizlaze iz temeljnih poslovnih dokumenata organizacije, omogućavaju objektivno poslovanje, uz pravodobnu podršku odlučivanju i jasne smjernice djelovanja.

Sadržajem ovog rada, a naročito kratkom analizom praktičnog primjera, prikazano je da ovakav spoj ontologija, kao oblika pohrane podataka i znanja, i poslovnih pravila, nije u potpunosti neizvediv ili bespredmetan. Ontologije, u usporedbi sa relacijskim bazama podataka, omogućavaju jednostavnije postavljanje kompleksnijih upita, a i zaključivanje, što olakšava unos podataka. Unatoč zamijećenom nedostatku proučavanja ovog područja, prikazano je da ima smisla dodatno proučiti ovaj spoj koji ima potencijala biti koristan poslovnom svijetu, ali i informatičkom svijetu, mijenjajući odnos prema podacima, znanju, ali i poslovanju, tj. principima poslovanja zrcaljenim u poslovnim pravilima.

Pa ipak, područje ovog rada nije bila usporedba trenutno standardnog i iznimno raširenog načina zapisa podataka korištenjem relacijskih baza podataka, već istraživanje mogućnosti suradnje ontologija sa poslovnim pravilima. Usprkos tome, bilo bi iznimno korisno istražiti razliku između pohranjivanja podataka korištenjem ontologija i relacijskih baza podataka. Kvantitativna spoznaja takve razlike ili ideja o međusobnoj suradnji ontologija i „uobičajenih“ baza podataka, ukazala bi o kvaliteti i korisnosti daljnje razrade u ovom radu proučavanog spoja.

Semantički web, a time i ontologije, relativno su novi koncepti u području informacijskih znanosti, naročito u svom široko primjenjivom obliku. Upravo stoga je za očekivati da se njihov veliki razvoj tek očekuje, a da će njihova primjena rasti u budućnosti. Ponovna iskoristivost, međusobna povezivost, velike količine podataka, rast računalne moći i integracija računalnih sustava u svakodnevni život razlozi su zbog kojih autor ovog rada smatra da se može očekivati daljnji razvoj semantičkog modeliranja. Poslovna pravila, a i pravila uopće, u tom okruženju samo dodatno olakšavaju integraciju informacijskih znanosti u poslovne sustave.

U konačnici, proučavanje daljnjih mogućnosti integracije poslovnih pravila i ontologija, pa i usporedbe s nekim drugim sustavima, na većoj ili realnoj poslovnoj domeni, iznimno bi doprinio ovom radu i kvaliteti zaključaka istog. U svakom slučaju, temelji su postavljeni.

Literatura

[1] Business Rule Solutions, 2009. *Basic RuleSpeak Guidelines.* s.l.:Business Rule Solutions, LLC.

[2] Business Rule Solutions, 2009. *RuleSpeak Sentence Forms.* s.l.:Business Rule Solutions, LLC.

[3] Business Rules Group, 2000. *Defining Business Rules ~ What Are They Really?,* s.l.: an.

[4] Business Rules Group, 2003. *The Business Rules Manifesto.* [Mrežno]
Dostupno na: http://www.businessrulesgroup.org/brmanifesto.htm

[5] Ceravalo, P., Fugazza, C. & Leida, M., 2007. *Modeling Semantics of Business Rules.* Cairns, an., pp. 171 - 176.

[6] Demey, J., Jarrar, M. & Meersman, R., 2002. *A Markup Language for ORM Business Rules.* Brussels, an., pp. 107-128.

[7] Drummond, N., 2009.. *The Manchester OWL Syntax.* [Mrežno]
Dostupno na: http://www.co-ode.org/resources/reference/manchester_syntax/
[Pokušaj pristupa 30 ožujka 2013.].

[8] Gerrits, R., 2012. Putting Business Rules in the Hands of the Business. *Business Rules Journal,* travanj.13(4).

[9] Halpin, T., 1996. Business Rules and Object Role Modeling. *Database Programming & Design*, October.

[10] Halpin, T., 2009. Object-Role modeling. U: *Encyclopedia of database systems.* New York: Springer.

[11] Halpin, T., 2009. Ontological Modeling (Part 1). *Business Rules Journal,* rujan.10(9).

[12] Halpin, T., 2009. Ontological Modeling (Part 2). *Business Rules Journal,* prosinac.10(12).

[13] Halpin, T., 2010. Ontological Modeling (Part 3). *Business Rules Journal,* ožujak.11(3).

[14] Halpin, T., 2010. Ontological Modeling (Part 4). *Business Rules Journal,* lipanj.11(6).

[15] Halpin, T., 2010. Ontological Modeling (Part 5). *Business Rules Journal,* prosinac.11(12).

[16] Halpin, T., 2011. Ontological Modeling (Part 7). *Business Rules Journal,* lipanj.12(6).

[17] Halpin, T., 2011. Ontological Modeling (Part 8). *Business Rules Journal,* rujan.12(9).

[18] Halpin, T., 2011. Ontological Modeling (Part 9). *Business Rules Journal,* prosinac.12(12).

[19] Halpin, T., 2012. Ontological Modeling (Part 10). *Business Rules Journal,* ožujak.13(3).

[20] Halpin, T., 2012. Ontological Modeling (Part 12). *Business Rules Journal,* studeni.13(11).

[21] Ma, Z. M. & Wang, X., 2012. Rule Interchange in the Semantic Web. *Journal of Intormation Science and Engineering,* Issue 28, pp. 393-406.

[22] Object Management Group, 2012.. *Introduction To OMG's Unified Modeling Language™ (UML®).* [Mrežno]
Dostupno na: http://www.omg.org/gettingstarted/what_is_uml.htm

[23] O'Connor, M., 2012. *SWRLLanguageFAQ.* [Mrežno]
Dostupno na: http://protege.cim3.net/cgi-bin/wiki.pl?SWRLLanguageFAQ
[Pokušaj pristupa 6 ožujka 2013].

[24] RIF Working Group, 2013. *RIF FAQ.* [Mrežno]
Dostupno na: http://www.w3.org/2005/rules/wiki/RIF_FAQ
[Pokušaj pristupa 18 ožujka 2013].

[25] Ross, R. G., 2000. *What is a "Business Rule"?.* [Mrežno]
Dostupno na: http://www.brcommunity.com/b005.php
[Pokušaj pristupa 2 listopada 2012].

[26] Ross, R. G., 2010. What is a Business Rule?. *Business Rules Journal,* ožujak.11(3).

[27] Ross, R. G., 2012. Business Processes: Better with Business Rules. *Business Rules Journal,* svibanj.13(5).

[28] Ross, R. G. & Lam, G. S. W., 2011. Business Rules: Basic Principles. *Business Rules Journal,* prosinac.12(12).

[29] W3C, 2004. *OWL Web Ontology Language Overview.* [Mrežno]
Dostupno na: http://www.w3.org/TR/owl-features/

[30] W3C, 2012.. *OWL 2 Web Ontology Language Document Overview (Second Edition).* [Mrežno]
Dostupno na: http://www.w3.org/TR/owl2-overview/

[31] W3C, 2012.. *RIF Primer (Second Edition).* [Mrežno]
Dostupno na: http://www.w3.org/TR/2013/NOTE-rif-primer-20130205/

[32] W3C, 2013.. *RIF Basic Logic Dialect (Second Edition).* [Mrežno]
Dostupno na: http://www.w3.org/TR/2013/REC-rif-bld-20130205/

[33] W3C, 2013.. *RIF Overview (Second Edition).* [Mrežno]
Dostupno na: http://www.w3.org/TR/2013/NOTE-rif-overview-20130205/

Dodatni elementi

Popis slika

Slika 2.1: Dijagram dijela modela činjenica sustava visokog školstva Hrvatske. (izvor: izrada autora) 5
Slika 2.2: Kategorizacija u primjeru modela činjenica. (izvor: izrada autora) 6
Slika 2.3: Svojstva koncepta *Student* u primjeru modela činjenica. (izvor: izrada autora) 6
Slika 2.4: Klasifikacija koncepta *Visoko učilište* u primjeru modela činjenica. (izvor: izrada autora) 6
Slika 2.5: Objektifikacija veze *Student – Visoko učilište – ECTS bod* u primjeru modela činjenica. (izvor: izrada autora) 7
Slika 2.6: Uloge u primjeru modela činjenica. (izvor: izrada autora) 7
Slika 2.7: Kompozicija koncepta *Sveučilište* u primjeru modela činjenica. (izvor: izrada autora) 8
Slika 3.1: Konačni UML dijagram klasa primjera modeliranja poslovnih pravila pomoću UML-a. (izvor: izrada autora) 16
Slika 3.2: Apstraktna klasa *Osoba* s atributom *prezIme*. 16
Slika 3.3: Klasa *Student* je podklasa klase *Osoba* i ima atribute *godinaStudija* i *jmbag.* (izvor: izrada autora) 17
Slika 3.4: Klasa *Nastavnik* je podklasa klase *Osoba* i ima atribut *titula.* (izvor: izrada autora) 18
Slika 3.5: Konačni UML dijagram opisanog primjera. (izvor: izrada autora) 19
Slika 3.6: Konačni ORM dijagram primjera. (izvor: izrada autora) 21
Slika 3.7: Osoba mora imati ime i prezime, a *Student* i *Nastavnik* su podklase klase *Osoba.* (izvor: izrada autora) 21
Slika 3.8: *JMBAG* jednoznačno određuje elemente klase *Student.* (izvor: izrada autora) 22
Slika 3.9: Predmet ima identificirajući *Naziv* i broj ECTS bodova veći od 0. (izvor: izrada autora) 22
Slika 3.10: Veze između klasa *Nastavnik* i *Predmet.* (izvor: izrada autora) 23
Slika 3.11: Predmet može imati drugi predmet kao preduvjet ili biti preduvjet drugom predmetu. (izvor: izrada autora) 24
Slika 3.12: Konačni ORM model, opisan navedenim poslovnim pravilima. (izvor: izrada autora) 24
Slika 4.1: Poluvilenjaci svijeta J.R.R. Tolkiena i njihov rodbinski odnos; plavo su označeni čistokrvni vilenjaci, ljubičasto poluvilenjaci, a crveno ljudi. (izvor: preuzeto s http://bindingobsession.com/misc/) 26
Slika 4.2: Temeljni slojevi semantičkog weba, Unicode i URIrefs na dnu te OWL koji se nadograđuje na RDF, RDFS te XML na vrhu. (prema: (Halpin, 2009)) 27
Slika 4.3: RDF model prikazan kao jednostavan imenovan usmjeren graf. (prema: (Halpin, 2009b)) 28
Slika 4.4: RDF trojka značenja „LOTR1 je instanca tipa knjiga.“ 30
Slika 4.5: Presjek triju skupova, prikazan Vennovim dijagramom, gdje presjek predstavlja područje gdje se sva tri skupa preklapaju, označeno najtamnije zeleno. (izvor: izrada autora) 39
Slika 6.1: Klase promatrane ontologije. (izvor: izrada autora) 57
Slika 6.2: Komentar uz klasu *Klijent* promatrane ontologije. (izvor: izrada autora) 58
Slika 6.3: Objektna svojstva promatrane ontologije. (izvor: izrada autora) 58
Slika 6.4: Ograničenja i karakteristike svojstva *jeZaKlijenta*. (izvor: izrada autora) 59
Slika 6.5: Podatkovna svojstva promatrane ontologije. (izvor: izrada autora) 60
Slika 6.6: Ograničenja i karakteristika svojstva *imaCijenuProizvoda*. (izvor: izrada autora) 60
Slika 6.7: Prikaz osnovnih podataka klijenta za račun. (izvor: izrada autora) 62
Slika 6.8: Dodana stavka računa. (izvor: izrada autora) 63

Slika 6.9: Stavke trenutnog računa prije brisanja stavke redni broj 2. (izvor: izrada autora) 63
Slika 6.10: Stavke trenutnog računa nakon brisanja stavke redni broj 2. (izvor: izrada autora) ... 63

Popis tablica

Tablica 6.1: Analiza simbola pravila PP 6.1. (izvor: izrada autora) 55
Tablica 6.2: Isječak modela činjenica promatrane organizacije. (izvor: izrada autora) 55
Tablica 6.3: Dodatak modelu činjenica promatrane organizacije. (izvor: izrada autora) 55
Tablica 6.4: Analiza simbola pravila PP 6.3. (izvor: izrada autora) 56
Tablica 6.5: Isječak modela činjenica promatrane organizacije. (izvor: izrada autora) 56
Tablica 6.6: Predloženo proširenje modela činjenica, bez navođenja pojašnjenja koncepata s ciljem smanjenja tablice. (izvor: izrada autora) 58

Popis poslovnih pravila

PP 2.1: Redoviti student pri upisu više godine studija, sa, uključivo, više od 55 ECTS bodova položenih na prethodnoj godini studija, ne plaća školarinu. 4
PP 2.2: Srebrni kupac kupuje često. 5
PP 2.3: Za vrijeme zime, kaput se mora ostaviti u garderobi. 9
PP 2.4: Na predavanje student može nositi bilježnicu. 9
PP 2.5: Pozvani predavač mora dobiti pratnju do dvorane za predavanje. 9
PP 2.6: Nakon upisnog roka, student bi pri upisu trebao platiti zakasninu 300,00 kn. 9
PP 2.7: Astronautsko odijelo mora biti nošeno prilikom šetnje svemirom. 11
PP 2.8: Indeks upisanog studenta trebao bi biti potpisan. 11
PP 2.9: Indeks upisanog studenta mora biti potpisan. 11
PP 2.10: Student može dobiti potvrdu o upisu u studentskoj referadi. 11
PP 2.11: Student može dobiti potvrdu o upisu u studentskoj referadi samo za vrijeme rada sa strankama. 11
PP 2.12: Student mora potpisati ugovor prilikom upisa, bez iznimaka. 11
PP 2.13: Student mora potpisati ugovor prilikom upisa. 11
PP 2.14: Student polaganjem predmeta osvaja samo ECTS bodove koji pripadaju tom predmetu. 12
PP 2.15: Projekt mora podržati Studentski zbor. 12
PP 2.16: Projekt mora voditi Studentski zbor. 12
PP 2.17: Ako su studenti počinili stegovni prekršaj, onda moraju pristupiti stegovnom sudu. 12
PP 2.18: (Svaki) Student koji je počinio stegovni prekršaj mora pristupiti stegovnom sudu. 12
PP 2.19: Laboratorijske vježbe moraju imati više od 15+1 računalnih mjesta. 12
PP 2.20: Broj računalnih mjesta na laboratorijskim vježbama mora biti više od 15+1. 12
PP 2.21: Student za prijavu na Natječaj mora ispuniti online prijavu i predati motivacijsko pismo. 13
PP 2.22: Student za prijavu na Natječaj mora obaviti sve navedeno: 13
PP 2.23: Redovni student mora upisati više od 50 ECTS bodova godišnje. 13
PP 2.24: Redovni broj ECTS bodova mora biti više od 50 ECTS bodova godišnje. 13
PP 2.25: Redovni student mora upisati redovni broj ECTS bodova. 13
PP 2.26: Predmet nije moguće položiti ako je neocijenjen. 13
PP 2.27: Položen predmet mora biti ocijenjen. 13
PP 3.1: Osoba mora imati ime i prezime. 16
PP 3.2: Student mora biti osoba. 16
PP 3.3: Nastavnik mora biti osoba. 16
PP 3.4: Student mora imati JMBAG. 17
PP 3.5: JMBAG mora jednoznačno identificirati akademskog građanina. 17

PP 3.6: Student mora biti na jednoj od sljedećih godina studija: 17
PP 3.7: Nastavnik mora imati jednu od sljedećih titula: 17
PP 3.8: Predmet mora imati naziv. 18
PP 3.9: Predmet mora imati broj ECTS bodova veći od 0. 18
PP 3.10: Predmet mora voditi jedan nastavnik nositelj predmeta. 18
PP 3.11: Predmet mora predavati jedan ili više nastavnika. 18
PP 3.12: Upis predmeta može biti uvjetovan drugim predmetima. 18
PP 3.13: Nastavnik ne smije biti nositelj više od 3 predmeta. 19
PP 3.14: Upis predmeta ne smije biti uvjetovan istim predmetom. 19
PP 3.15: Osoba mora imati ime i prezime. 21
PP 3.16: Student mora biti osoba. 21
PP 3.17: Nastavnik mora biti osoba. 21
PP 3.18: Student mora imati JMBAG. 22
PP 3.19: JMBAG mora jednoznačno identificirati akademskog građanina. 22
PP 3.20: Student mora biti na jednoj od sljedećih godina studija: 22
PP 3.21: Predmet mora imati naziv. 22
PP 3.22: Predmet mora imati broj ECTS bodova veći od 0 22
PP 3.23: Predmet mora voditi jedan nastavnik nositelj tog predmeta. 23
PP 3.24: Predmet mora predavati jedan ili više nastavnika. 23
PP 3.25: Upis predmeta može biti uvjetovan drugim predmetima. 24
PP 3.26: Upis predmeta ne smije biti uvjetovan istim predmetom. 24
PP 3.27: Nastavnik ne smije biti nositelj više od 3 predmeta. 25
PP 5.1: Student koji kroz kontinuirano praćenje, iz kolokvija, zadaća, seminara i projekta, sakupi više od 90 bodova, dobit će ocjenu izvrstan. 51
PP 5.2: Više od 90 ukupno bodova studentu nosi ocjenu izvrstan. 51
PP 5.3: Student sa više od 90 ukupno bodova ima ocjenu izvrstan. 52
PP 6.1: Klijent koji kod nas potroši do 500 kn nema pravo ni na kakav popust pri kupovini. 54
PP 6.2: Klijent koji kod nas potroši do 2000 kn ima odobren popust do 10%, ostvariv pri svakoj budućoj kupovini. 54
PP 6.3: Klijentu s potrošenih 2000 kn odobrava se popust u iznosu od 20%. 54
PP 6.4: Klijent koji kod nas potroši ukupno 3000 kn ima pravo na popust u iznosu od 30%. 54
PP 6.5: Klijent koji ima iznos manji od 500 kn, ima odobren popust koji iznosi 0%. 55
PP 6.6: Klijent koji ima iznos veći od ili jednak 2000 kn i manji od 3000 kn, ima odobren popust koji iznosi 20%. 57
PP 6.7: Klijentu koji ima iznos veći od ili jednak 3000 kn, pripada status zlatnog klijenta 61
PP 6.8: Klijent statusa zlatnog klijenta ima odobren popust koji iznosi 30%. 62

Popis isječaka koda

Kôd 4.1: Manchester zapis koji definira imenički prostor *koncepti*: 30
Kôd 4.2: Manchester zapis definiranja imeničkih prostora *rdf*: i *owl*: 30
Kôd 4.3: Manchester zapis RDF trojke iz Slika 4.4. 30
Kôd 4.4: Manchester zapis dviju klasa i njihovog međusobnog odnosa. 31
Kôd 4.5: Manchester zapis koji opisuje klasu *Knjiga* kao podklasu klase *Tiskovina* koja je, pak, podklasa klase *Medij*. 31
Kôd 4.6: Manchester zapis klase *Knjiga* kao podklase klase *Medij*. 31
Kôd 4.7: Manchester zapis ograničenja svojstva *autorJe* na domenu *Knjiga* i doseg *Autor*. 32
Kôd 4.8: Manchester zapis svojstva *jeSastavniDioZa* kao podsvojstva od *jeDioZa*. 32
Kôd 4.9: Manchester zapis značenja „Neki Procesor sastavni je dio nekog Računala.“ 32
Kôd 4.10: Manchester zapis značenja „Neki Procesor dio je nekog Računala.“ 32

Kôd 4.11: Manchester zapis ograničenja najmanje kardinalnosti svojstva *imaAutora* na 1. 34
Kôd 4.12: Manchester zapis ograničenja najveće i najmanje kardinalnosti svojstva *brojPokušaja* na 15, odnosno 1. 34
Kôd 4.13: Manchester zapis kojim je određeno da svaka individua klase *Osoba* ima točno jedan spol pridružen svojstvom *imaSpol* i točno jednu vrijednost pridruženu svojstvom *imaOIB.* 34
Kôd 4.14: Manchester zapis značenja da je individua *Gandalf* isto što i individua *Mihtrandir* i individua *Olórin.* 35
Kôd 4.15: Manchester zapis klase *Auto* koja je jednaka klasi *Automobil.* 35
Kôd 4.16: Manchester zapis klase *Muškarac* koja je odvojena od klase *Žena.* 35
Kôd 4.17: Manchester zapis dvije klase i odnosa disjunktnosti među njima. 35
Kôd 4.18: Manchester zapis dvaju svojstava koja imaju međusobno suprotne domenu i doseg. 36
Kôd 4.19: Manchester zapis kojim je određeno da je dano svojstvo inverzno nekom drugom svojstvu. 36
Kôd 4.20: Manchester zapis refleksivnog svojstva poznaje. 37
Kôd 4.21: Manchester zapis tranzitivnog asimetričnog svojstva *jePredakOd*. 37
Kôd 4.22: Manchester zapis individue *Ozano* i *Zrinka*, gdje je individua *Ozano* predak od individue *Zrinka*, a *Zrinka* predak od individue *Zvonko* 37
Kôd 4.23: Manchester zapis individue *Ozano*, koja je predak od individue *Zvonko* 37
Kôd 4.24: Manchester zapis svojstva *SlušaNastavnika* kao niza od svojstava *slušaPredmet* i predaje. 38
Kôd 4.25: Manchester zapis svojstva *jeUnukOd* kao niza od dva *jeDijeteOd* svojstva. 38
Kôd 4.26: Manchester zapis svojstva *studiraUDržavi* kao niza od tri svojstva. 38
Kôd 4.27: Manchester zapis klase *StudentDiplomskog* kao unije klasa. 38
Kôd 4.28: Manchester zapis dviju međusobno isključivih klasa i klase koja potpuno obuhvaća obje te klase. 38
Kôd 4.29: Manchester zapis kojim se izravno određuje da klasa *TopliNapitak* obuhvaća klase *Čaj* i *Kava*, dok su one međusobno isključive. 39
Kôd 4.30: Manchester zapis klase *IzvanredniStudent* definirane kao presjek klasa *Student* i *Zaposlen.* 39
Kôd 4.31: Manchester zapis klase studenata koji su položili ispit kao komplementa klase studenata koji su pali ispit, pri čemu su promatrani samo studenti koji su uopće izašli na ispit. 39
Kôd 4.32: Manchester zapis klase *Vozač* pomoću presjeka te klase *NeVozač* pomoću presjeka i komplementa. 40
Kôd 4.33: Manchester zapis klase *RadniDan*, koja ima konačan skup elemenata. 40
Kôd 4.34: Manchester zapis klase *StudentDiplomskog* koja je ograničena klasom *Student* i vrijednošću svojstva *studiraNaRazini.* 40
Kôd 4.35: Manchester zapis klase *VoćnaSalata* koja ima sastojke koji pripadaju isključivo klasi *Voće*, dok se instanca klase *SalataSaSirom* sastoji od povrća i od sira. 40
Kôd 5.1: Manchester zapis klase *PrivatnaOsoba* kao presjeka klasa *Osoba* i *PrivatniKlijent.* 43
Kôd 5.2: Pravilo SWRL 5.14 zapisano u OWL/XML obliku. 47
Kôd 5.3: B je logička posljedica od A. 49
Kôd 5.4: Ako A onda B. 49
Kôd 6.1: XML zapis objektnih svojstava *jeNapravioRacun* i *jeZaKlijenta*. 59
Kôd 6.2: XML zapis objektnih svojstva *imaCijenuProizvoda*. 61
Kôd 6.3: XML zapis individue klase *Klijent.* 61
Kôd 6.4: Zapisivanje stavki računa u ontologiju. 64
Kôd 6.5: Ažuriranje vrijednosti iznosa koju ima klijent. 64

Popis SWRL pravila

SWRL 5.1: Varijabla *?o* postaje individua OWL klase *Osoba.* 43
SWRL 5.2: Provjerava se je li OWL individua naziva *Maja* član ekstenzije OWL klase *Osoba.* 43
SWRL 5.3: Osoba koja je *PrivatniKlijent* je *PravnaOsoba*. 43
SWRL 5.4: Istinito u slučaju kad je individua *?z* povezana s individuom *?m* svojstvom *imaMentora.* 43
SWRL 5.5: *Zaposlenik* koji *vodiDioOrganizacije* je *Voditelj.* 43
SWRL 5.6: Ako je *Odjel jeDioIspostave* koja *jeDioPodružnice*, onda je *Odjel jeOdjelPodružnice.* 44
SWRL 5.7: Svojstvo *imaStanovnika* pridružuje nekom mjestu cijeli broj koji predstavlja broj stanovnika tog mjesta. 44
SWRL 5.8: Mjesto koje ima broj stanovnika veći od 10 000 je grad. 44
SWRL 5.9: Ako je zaposlenik mentor zaposleniku, onda je on mentor. 45
SWRL 5.10: Ako je zaposlenik mentor zaposleniku, ne samome sebi, onda je on mentor. 45
SWRL 5.11: Račun koji ima kupca koji je klijent i koji ima određeni iznos, a kupac ima određeni popust, ukupan iznos računa se umanjuje za iznos popusta. 45
SWRL 5.12: Atomi premise povezani konjunkcijom. 46
SWRL 5.13: Konjunkcija iz prethodnog primjera razbijena je na dva pravila. 46
SWRL 5.14: Klijent brončane kategorije ima popust 3%. 47
SWRL 6.1: Klijent koji ima iznos veći od ili jednak 3000, pripada klasi *'zlatni klijent'* 62
SWRL 6.2: Zlatnom klijentu odobren je popust od 30%. 62
SWRL 6.3: Pravilo kojim je određen način stvaranja šifre stavke. 65
SWRL 6.4: Pravilo kojim je određen način izračuna ukupnog iznosa stavke. 65

Popis RIF pravila

RIF 1: *atomZaključka* logička je posljedica konjunkcije atoma premise. 49
RIF 2: *Ozano* je individua klase *Mentor* ako je individua klase *Zaposlenik* i sudjeluje u vezi *jeMentorZaposleniku* s individuom *Iva*, klase *Zaposlenik*. 49
RIF 3: Postoji barem jedan klijent koji ima status *StatusKlijentaZlatni*. 50
RIF 4: Prefiks *dipl* bit će zamjena za navedeni IRI. 50
RIF 5: Valjani RIF dokument s pravilom RIF 2. 50
RIF 6: U potpunosti valjan RIF dokument. 51

Popis OCL izraza

OCL 1: Atribut *jmbag* ne smije biti prazan. 17
OCL 2: Nastavnik može biti nositelj najviše 3 predmeta. 19
OCL 3: Predmet ne može za uvjet imati samog sebe. 19