KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMACIJOS SISTEMŲ KATEDRA

LAIMONAS ŽOLPYS

OWL ONTOLOGIJŲ ATKŪRIMAS IŠ RELIACINIŲ DUOMENŲ BAZIŲ

Magistro darbas

Darbo vadovas

Prof. dr.: L. Nemuraitė

Kaunas, 2013

KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMACIJOS SISTEMŲ KATEDRA

OWL ONTOLOGIJŲ ATKŪRIMAS IŠ RELIACINIŲ DUOMENŲ BAZIŲ

Magistro darbas

Darbo vadovas
Prof. dr.: L. Nemuraitė

Recenzentas
Doc.: S. Maciulevičius

Studentas
IFM 1/4 L. Žolpys

Kaunas, 2013

ABSTRACT

In recent years the development of ontologines- formal specifications of the terms in the domain and relations among them has been expanding from the Artificial-Intelligence laboratories to the desktops of domain experts. Ontologies have become common on the World-Wide Web. The ontologies on the Web range from large taxonomies categorizing Web sites such as on „*Yahoo"* to categorizations of products for sale and their features such as on „*Amazon.com*". It is a language for encoding knowledge on Web pages to make it understandable to electronic agents searching for information. An ontology defines a common vocabulary for researchers who need to share information in a domain. It includes machine interpretable definitions of basic concepts in the domain and relations among them. Encoded information, vocabulary for researchers, formal specifications of the terms and other are saved in relational databases.

The aim of this research is to improve possibilities of querying ontologies when these are kept in relational databases by creating and realizating the algorithm, which allows to transform ontology from relational databases. Experiments have shown that the method works for relation databases which were created by OWL2RDB algorithm.

Key words: ontology, relational database, OWL, RDB2OWL.

TURINYS

Terminų ir santrumpų žodynėlis

• Ontologija – dalykinės srities konceptų aprašymas;

• OWL Reasoner – ontologijų analizės ir užklausų vykdymo programa;

• RDBOWL2 – ontologijos konvertavimo iš reliacinės duomenų bazės schemos algoritmas;

• RDF (Resource Description Framework) - resursų aprašymo karkasas;

• RDFS (RDF Schema) – RDF išplėtimas;

• URI (Uniform Resource Identifier) – interneto resursu identifikatorius.

• W3C (World Wide Web Consortium) – tarptautinė standartų organizacija.

1. Įvadas

Per pastaruosius metus ontologijų kūrimas- tikslių formalių specifikacijos terminų ir specifikacijų sąryšių dalykinėje srityje, pradėjo plėstis nuo dirbtinio intelekto srities laboratorijų iki dalykinių sričių ekspertų darbalaukių. Ontologijos tapo dažnos pasauliniame žinių tinkle. Ontologijos naudojamos internete nuo didelių sistematikos klasifikavimo puslapių iki tokių kaip *„Yahoo“* ,iki internetinių prekių klasifikavimo ir jų savybių klasifikavimo pardavimui tinklapių, tokių kaip *„Amazon.com“*. Jeigu informacija yra vienodos struktūros, t.y. visi terminai naudojami tie patys, automatinės paieškos sistemos gali sujungti informacija iš skirtingų šaltinių ir pateikti vartotojui kaip visumą. Ontologija apibrėžia dažnai naudojamą žodyną tyrinėtojams, kuriems reikia dalintis informacija dalykinėje srityje. Į tai įeina dalykinėje srityje kompiuterių interpretuojami apibrėžimai apie pagrindines sąvokas, sąryšius tarp jų. Informacija, žodynai tyrinėtojams, kompiuterių interpretuojami apibrėžimai ir kitą yra saugomi reliacinėse duomenų bazėse.

Darbo tikslas: padidinti ontologijų išgavimo iš reliacinių duomenų bazių galimybes sukuriant ir realizuojant tam skirtą algoritmą, leidžiantį atstatyti ontologiją iš duomenų bazės be informacinių nuostolių.

2. Analizė

Ontologijos yra naudojamos fiksuoti žinias apie tam tikrą interesų sritį. Ontologijos aprašo klasių (sąvokų) sritį, o taip pat santykius, esančius tarp šių klasių. Ji turi rinkinį operatorių - pvz. sankirta, sąjunga ir paneigimas. Ji grindžiama loginiu modeliu, kuris leidžia gerai aprašyti klases. Sudėtingos klasės gali būti sukurtos iš paprastesnių klasių. Be to, loginis modelis leidžia naudoti „*Reasoner"*, kuris gali padėti išlaikyti hierarchiją teisingą. Tai ypač naudinga, kai susiduriame su atvejais, kai klasės gali turėti daugiau nei vieną tėvą.

2.1.Tyrimo objektas, sritis ir problema

Darbo tyrimo objektas - ontologijos išgavimo ir transformavimo i OWL ontologiją procesas.

Tyrimo sritis – ontologijų ir duomenų bazių inžinerija ir jos įrankiai.

Problema – ontologijas yra tikslinga saugoti duomenų bazėse. Vienas iš metodų yra sukurtas informacijos sistemų katedroje ir norint šį metodą taikyti praktiškai, reikia turėti galimybę atkurti ontologiją ir jos egzempliorius iš duomenų bazės. Ontologijas saugančių DB įrankių yra daug, tačiau daugelis įrankių išsaugo tik dalį ontologijos savybių arba neišnaudoja reliacinių DB privalumų ir saugojimas yra neefektyvus. ISK kuriamo OWL to RDB metodo tikslas yra išsaugoti ontologijas duomenų bazėse be informacinių nuostolių. Vykdant užklausas, ontologijas reikia atkurti iš duomenų bazės schemos. Išgavimo metu informacija neturi būti prarandama, išgauta ontologija turi atitikti pradinę ontologiją, iš kurios buvo sukurta duomenų bazė

2.2.Tiriamo objekto analizė

2.2.1. OWL2 ontologijos kalba

Dirbtinio intelekto literatūroje yra daug apibrėžimų apie ontologiją, daugelis jų prieštarauja vienas kitam. Ontologija yra formalus tikslus apibūdinimas sąvokų dalykinėje srityje (klasės (kartais vadinamos sąvokomis)), savybės ,kiekvienos sąvokos, apibūdina įvairius požymius ir atributus šių sąvokų.

OWL2 tinklo ontologijos kalba, neoficialiai OWL2 yra ontologijos kalba semantiniam tinklui su formaliai apibrėžta reikšme. OWL2 turi klases, savybes, individus ir duomenų reikšmes joje yra išsaugotos kaip semantinio tinklo dokumentai. OWL2 ontologijos gali būti naudojamos kartu su RDF informacija [14].

OWL2 ontologija yra formalus aprašymas dalykinės srities interesų. OWL2 ontologijos susideda iš trijų skirtingų sintaksės kategorijų:

1. Subjektai tokie kaip klasės, savybės, individai yra identifikuojami pagal IRI. Jie sudaro primityvų ontologijos požiūrį ir yra pagrindiniai ontologijos komponentai. Pavyzdžiui, klasė a:Žmogus gali būti naudojama nusakyti visus absoliučiai žmones. Panašiai objekto savybė a:Tėvas gali būti naudojamas nusakyti tėvo-vaiko santykius. Galiausiai individas a:Petras gali būti naudojamas atstovauti konkrečiam asmeniui vadinam Petru.

2. Išsireiškimai nusako sudėtines sąvokas, aprašant dalykinę sritį. Pavyzdžiui, klasės išraiška tai grupės individų sąvoka su charakteristikos apribojimais.

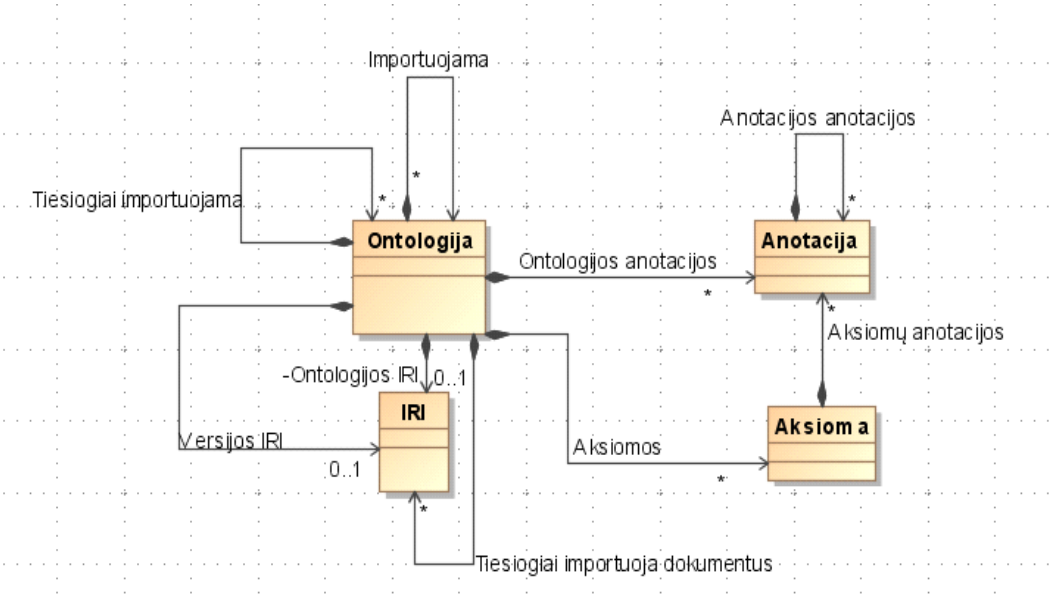
3. Aksiomos yra teiginiai, kurie turi būti teisingi, kada nusakoma dalykinė sritis. Pavyzdžiui, naudojant poklasio aksiomą galima teigti, kad klasė a:Studentas yra poklasis klasės a:Asmuo.

Šios trys sintaksinės kategorijos yra naudojamos nusakyti loginę OWL2 ontologijų dalį, tai yra, jos yra naudojamos tiksliai apibrėžtai semantikai, kuri leidžia naudingai atvaizduoti išvadas. Pavyzdžiui, jeigu individas a:Petras yra atskiras atvejas klasės a:Studentas ir a:Studentas yra poklasis a:Asmuo, tai pagal OWL2 semantiką galima išvesti išvadą, kad a:Petras yra atskiras atvejas iš klasės a:Asmuo.

Be to, subjektai, aksiomos, ontologijos gali būti paaiškinti su OWL2. Pavyzdžiui, klasė gali būti duota žmonėms suprantamu aprašymu, kas suteikia labiau nusakytą klasės reikšmę. Komentarai neturi jokios įtakos loginiams ontologijos aspektams, tai yra OWL2 semantikos tikslams, komentarai nėra nusakyti. Vietoj to komentarai yra palikti programoms, kurios naudoja OWL2. Pavyzdžiui, grafinė vartotojo sąsaja gali vizualizuoti klasę, naudojant vieną jos aprašymų.

OWL2 ontologija yra atskiras atvejas O UML ontologijos klasės iš struktūrinės OWL2 specifikacijos (pav. 1) kuris patenkina konkrečias sąlygas aprašytas žemiau. Pagrindinis OWL2 ontologijos komponentas yra aksiomų rinkinys. Kadangi tarp ontologijos ir jos aksiomų yra

nustatytas ryšys, ontologija negali turėti dviejų aksiomų, kurios yra struktūriškai lygios. Be aksiomų, ontologija gali turėti ontologijos anotacijas (ir jos taip pat gali turėti kitas ontologijas).



1 pav. OWL2 ontologijų struktūra

2.2.2. Reliacinių duomenų bazių sąvoka

DB, kuriose duomenys organizuojami pagal reliacinį modelį, - reliacinės DB. Reliacinė DB yra tokia duomenų visuma, kurioje informacija (duomenys) saugoma vadinamosiose dvimatėse lentelėse.

Lentelę sudaro eilutės ir stulpeliai, taigi lentelės forma yra įprasta ir suprantama vartotojui. Lentelės eilutės vadinamos įrašais, o stulpeliai - laukais. Į lentelės įrašus įtraukiamos duomenų porcijos, kurias sudaro DE reikšmės, dar vadinamos laukų reikšmėmis. Bet kurios eilutės ir bet kurio stulpelio susikirtime turi būti tik viena DE reikšmė, o ne tų reikšmių rinkinys.[16]

Reliacinėse DB kiekviena lentelė pasižymi tokiomis savybėmis:

1. Visi įrašai turi tą pačią struktūrą. Visuose įrašuose yra tiek pat laukų. Kiekvieno lauko reikšmės yra vieno tipo. Tačiau skirtinguose laukuose gali būti įvairių tipų duomenys;
2. Lentelėje negali būti tuščių įrašų, taip pat identiškų įrašų, t. y. įrašų su pilnai pasikartojančiais duomenimis, nors atskiri duomenų elementai gali būti tušti arba pasikartojantys;

3. Įrašų ir laukų išdėstymo tvarka lentelėje nėra svarbi. Atliekant duomenų apdorojimo operacijas lentelės eilutės ir stulpeliai gali būti peržiūrimi bei tvarkomi bet kuria tvarka, nepriklausomai nuo jų informacinio turinio.

Kiekvienai lentelei suteikiamas vardas, kuriuo ji saugoma kompiuterio išorinėje atmintyje (diske) kaip atskiras objektas. Lentelės vardas turėtų atspindėti atitinkamo realaus informacinio objekto pavadinimą, o laukų vardai - to objekto atributų pavadinimus.

Lentelėms nustatomi raktai, t. y. laukai ar laukų grupės, kurių įgyjamos reikšmės yra nepasikartojančios, taigi šios reikšmės vienareikšmiškai identifikuoja tų lentelių įrašus. Lentelės gali turėti po kelis raktus, iš kurių konkrečiu momentu faktiškai naudojamas tik vienas - pirminis raktas.

Į reliacinių DB sudėtį įeinančios lentelės tarpusavyje susiejamos. Ryšį tarp atskirų lentelių nustato bendri, sutampantys tų lentelių laukai, kurie dar vadinami siejančiais laukais. Taip susietų lentelių visuma ir apibrėžia reliacinį modelį. Kitaip negu rodykliniuose duomenų modeliuose, kur ryšiams iliustruoti naudojamos rodyklės, reliaciniame modelyje nėra atskirų elementų vaizduoti ryšiams - jie atspindimi pačiomis lentelėmis (siejančiais laukais). Pertekliniai laukai įvedami dažniausiai dėl dviejų priežasčių: a) daliai lentelių reikalingi laukai, skirti suformuoti pirminius raktus, t. y. vienareikšmius lentelių įrašų identifikatorius; b) kai kurioms lentelėms reikalingi laukai, kurie nėra pirminiai raktai (ar jų dalis), bet naudojami nustatyti ryšiams.

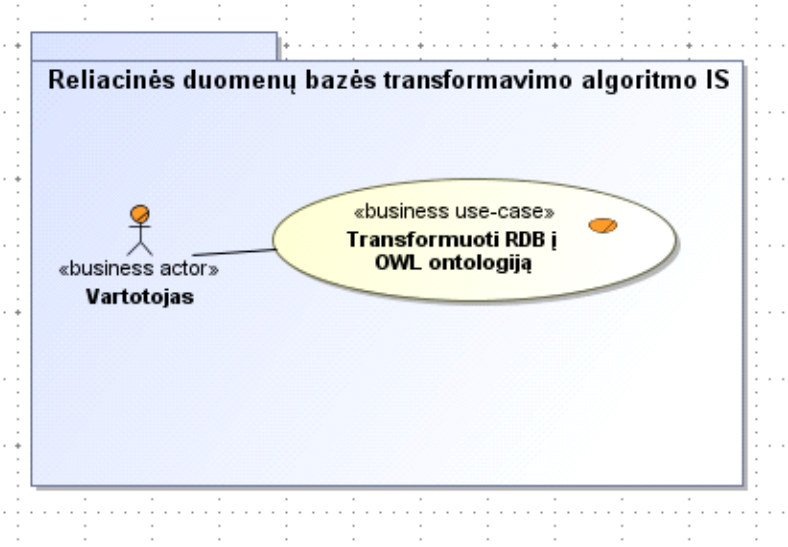
Reliacinė duomenų bazė yra santykių kolekcija. Kiti objektai dažnai laikomi duomenų bazės dalimi, nes jie padeda organizuoti ir susisteminti duomenis be to, verčia duomenų bazę atlikti įvairių reikalavimų rinkinį. Reliacinės duomenų bazės teorija naudoja matematinius terminus, kurie yra apytiksliai lygus SQL duomenų bazės terminologijai. Lentele žemiau atvaizduoja keletą labiausiai svarbių reliacinės duomenų bazės terminų ir jų SQL duomenų bazės atitikmenis.

Lentelė 1: reliacinių bazių ir SQL palyginimas:

Reliaciniai terminai:	SQL atitikmenys:
Santykis (relation), pagrindinis susiejimas (base relvar)	Lentelė (table)
Įgyti santykiai (derived relvar)	Peržiūra (view), užklausų rezultatai (query result), rezultatų rinkinys (result set)
Išrikiuotas elementų sąrašas (tuple)	Eilutė (row)
Atributas (attribute)	Stulpelis (colum)

2.3.Vartotojai ir jų tikslai

Kuriamo algoritmo vartotojai yra semantinio tinklo programų kūrėjai, projektuotojai, galiniai vartotojai ir kiti specialistai. Sukurtą algoritmas jie naudos atkurti ontologiją iš duomenų bazės schemos, sukurtos taikant OWL2- RDB metodą. OWL2RDB metodas leidžia išsaugoti daugiau ontologijos savybių, negu kiti. Dalis ontologijos savybių išsaugoma metaduomenų lentelėse. Dauguma atvejų algoritmas leis atstatyti ontologiją iš duomenų bazės be informacinių nuostolių. Pradinis vartotojų poreikis pavaizduotas *2pav*.



2 pav. Pradinis vartotojų poreikis

Problema – ontologijas saugančių DB įrankių yra daug, tačiau daugelis įrankių išsaugo tik dalį ontologijos savybių arba neišnaudoja reliacinių DB privalumų ir saugojimas yra neefektyvus.

2.4.Esamų sprendimų analizė

Cullot [Cullot et al., 2007] aprašo DB2OWL metodą, kuriame lentelės vaizduojamos klasėmis ir stulpeliai predikatais, tam tikri lentelių ryšiai vaizduojami poklasių ryšiais ir panašiai. Daug ir daug ryšiai bei išoriniai raktai vaizduojami objektų savybėmis.

Kashyap [Kashyap et al., 2007] metodas yra skirtas ontologijoms vaizduoti heterogeniniuose duomenų šaltiniuose naudojant tarpininkus ryšiams vaizduoti tarp ontologijos konceptų ir kitų duomenų šaltinių. Ontologijų vaizdavimas heterogeniniuose duomenų šaltiniuose iliustruojamas gamtos mokslų dalykinės srities scenarijais. Šie šaltiniai apima RDB, tinklo paslaugas ir Excel skaičiuokles (naudojamas MS Office API). SPARQL užklausos automatiškai išverčiamos į atitinkamą užklausų kalbą, panaudojant tarpininkų vaizdavimo klases.

Šukio A. nagrinėtas OWL iš RDB atstatymo algoritmas, aprašo ontologiją kaip aukšto semantinio lygmens dalykinės srities aprašymą [10], kuris apima klases, objektus, jų savybes ir ryšius. Ontologija leidžia daryti semantines išvadas, todėl ontologijos paieškos sistema pateikia vartotojui prasmingus rezultatus. Reliacinėje duomenų bazėje įvykdžius užklausą, sistema pateikia pagal sintaksinius parametrus iš duomenų bazės išrinktus duomenis. Norėdamas suprasti šiuos duomenis, vartotojas pats analizuoja jų prasmę pagal tam tikrus kriterijus, kurių kompiuteris suprasti negali, taigi tam tikrą duomenų paieškos darbo dalį atlieka pats žmogus. Pirmoji duomenų išrinkimo proceso veikla yra vartotojo užklausos pateikimas. Tada pagal vartotojo pateiktus parametrus formuojama *SQL* užklausa, kuri išrenka iš duomenų bazės norimus įrašus. Ontologijai saugoti naudojamas *RDF* resursų aprašymo karkasas. *RDF* pagrindas yra modelis, skirtas vaizduoti savybes ir savybių reikšmes. *RDF* modelis sudarytas iš trijų pagrindinių dalių. Šios trys pagrindinės *RDF* dalys sudaro formuluotę. Tokia duomenų struktūra yra tinkama aprašyti didžiąją dalį informacijos, apdorojamos kompiuterių. *RDF* aprašytos ontologijos saugomos tekstiniuose failuose, tačiau toks sprendimas turi trūkumą – kai ontologija yra didelės apimties, reikia daug kompiuterio darbinės atminties resursų norint saugoti tokią ontologiją

atmintyje ir vykdyti joje užklausas. Siekiant išplėsti semantines *RDF* galimybes, buvo sukurtos dvi papildomos kalbos: *RDF* Schema (*RDFS*) ir *OWL*.

2.4.1. RDB2OWL algoritmo analizė

RDB2OWL yra algoritmas, susiejantis reliacines duomenų bazes su nepriklausomai sukurtomis OWL ontologijomis. Jis yra pagrįstas ontologija atitinkančios RDB schemos kūrimu, užpildant ją informacija iš kurios yra generuojami SQL skriptai, atlikti individų lygmens transformaciją.

Saugojimui gali būti naudojami įvairūs algoritmai, kurie tarpusavyje skiriasi *RDF* saugojimo metodu. Kai kurie jų yra nepriklausomi nuo ontologijos schemos – t. y. duomenų bazės schema pastovi, todėl jie yra lankstesni, lengviau išplečiami. Vienas tokių - *Chebotko* algoritmas, paremtas grafo saugojimu vienoje lentelėje su trimis stulpeliais (*subject, predicate, object*) [10]. naudoja *RDFLib* mechanizmą, kuris saugo ontologiją trijose lentelėse. Norint saugoti sudėtingesnes *OWL* formuluotes, pavyzdžiui, kardinalumo ribojimus, toks saugojimo būdas nėra efektyvus, nes atsiranda daug perteklinių formuluočių [6], todėl pasirinktas algoritmas *OWL2RDB* , kuris transformuoja ontologiją į reliacinės duomenų bazės schemą. Šio algoritmo idėja – kiekvienai klasei sukurti po lentelę, kurios jungiasi ryšiais 1:0..1 (jei aprašoma klasių hierarchija) arba 1:* (jei aprašoma objekto savybė). Klasių objektus atitinka lentelių įrašai. Ontologijos apribojimams sukuriamos papildomos metaduomenų lentelės [11].

Norint įvykdyti *SPARQL* užklausą, kai ontologija saugoma reliacinėje duomenų bazėje pagal *OWL2RDB* algoritmą, užklausą reikia transformuoti. Transformavimui siūlomas metodas, kuris pradinę *SPARQL* užklausą išskaido į *SPARQL* užklausą, skirtą atrinkti tik klases, ir *SQL* užklausas, skirtas atrinkti individualiems konceptams [10]. Šis metodas paremtas *Pellet OWL Reasoner* ontologijų analizės ir užklausų vykdymo biblioteka.

2.4.2. Dalykinės srities specifinių projektų analizė

Green [Green et al., 2008] aprašo erdvinių duomenų integraciją į RDB skirtą sklindančiam vandens užterštumui modeliuoti, tam naudojant OWL-DL ontologijas daugelyje lygių. Pirmo lygio ontologijos (vadinamos duomenų ontologijomis) naudojamos nustatyti kiekvieno duomenų šaltinio konceptus kito lygio ontologijose (vadinamose dalykinės srities ontologijomis). „Duomenų ontologijos“ yra vaizduojamos D2RQ vaizdavimo kalba. „Programų

ontologija“ trečiame lygyje susieja „ dalykinės srities ontologijas“ taip pat prideda specifinę programų informaciją. D2RQ „procesorius“ yra modifikuotas įtraukti erdvinius operatorius ir yra naudojamas t duomenų šaltinių ir ontologijoms bendradarbiauti.

2.4.3. Įrankių ir programų analizė

R2O [Barrasa et al., 2006] yra XML pagrįsta deklaratyvi kalba RDB elementų ir ontologijos susiejimams aprašyti. R2O susiejimai gali būti naudojami surasti susiejimo apibrėžimų nesuderinamumus ir dviprasmybes. *ODEMapster* procesorius naudoja R2O dokumentą vykdyti pateiktos užklausos transformavimą arba paketiniame rėžime sukurti RDF (turinio atminties išklotinę).

RDBToOnto [Cerbah, 2008] yra konfigūruojamas įrankis, kuris palengvina ontologijos išgavimo iš duomenų bazių metodus. Tai taip pat į vartotoją orientuotas įrankis, palaikantis perėjimo procesą nuo prisijungimo prie duomenų bazės iki paskirstytų ontologijų generavimo. Mokymosi parametrų nustatymai ir procesų valdymas atliekamas per specialią – pilnavertę sąsają (*interface*).

Lentelė 2: Esamų alternatyvių sprendimų palyginimas

	Pateikimo kalba	Efektyvumas (ar išnaudoja reliacinių DB galimybes)	Automatizavimo lygis (automatinis ar neautomatinis/pusiau automatinis (Domain Semantics-driven))	Schemos pastovumas (ar schema priklauso nuo ontologijos)	Pritaikomumas
[Cullot et al., 2007]	R2O kalba	+	Automatinis	Dalis schemos nepriklausoma)	Tinka reliacinėms duomenų bazėms
[Kashyap et al., 2007]	Mediator Framework klasės	+	Rankinis/ pusiau automatinis	Dalis schemos nepriklausoma	Tinka reliacinėms duomenų bazėms
[Green et al., 2008]	D2RQ kalba	+	Rankinis/ pusiau automatinis	Dalis schemos nepriklausoma	Tinka reliacinėms duomenų bazėms
[Barrasa et al., 2006]	R2O kalba	+	Abu (vartotojų nustatoma)	Nepriklausoma	Tinka reliacinėms duomenų bazėms

Šukio prototipas	OWL2RDB kalba	+	Automatinis	Dalis schemos nepriklausoma	Pilnai tinka duomenų bazėms, sukurtoms pagal OWL2RDB algoritmą, kitoms - dalinai
[Cerbah, 2008]	Apribojimo taisyklės (Constraint rules)	+	Automatinis	Nepriklausoma	Tinka reliacinėms duomenų bazėms

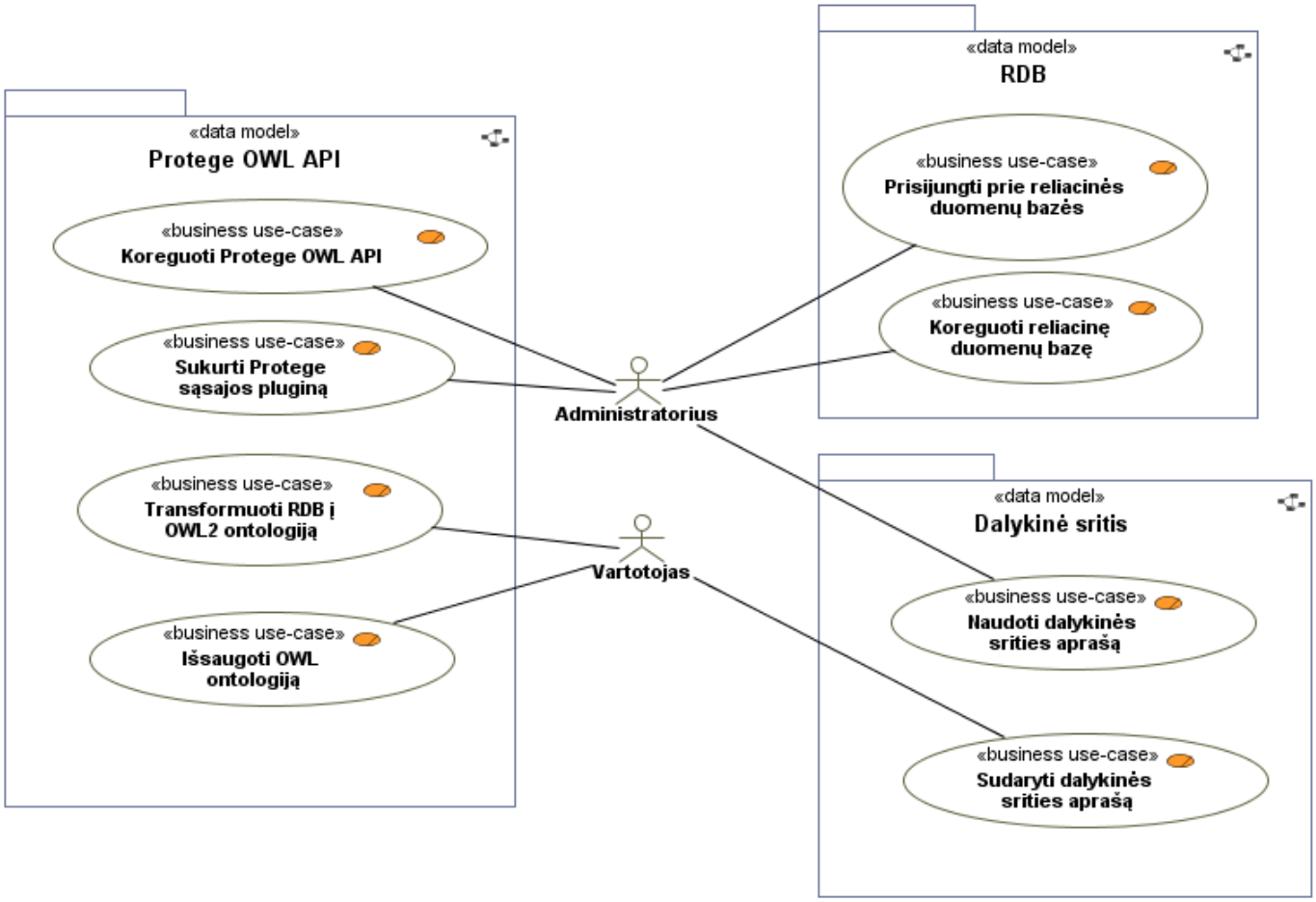
2.5. Siekiamas sprendimas

Siekiant suprasti kuriamą sprendimą bei jo sąvokas, sukurtas veiklos modelis (3pav.) Pradinė veiklos informacija yra dalykinės srities aprašas, kurį sudaro vartotojas. Administratorius naudodamasis aprašu , koreguoja reliacinę duomenų bazę pirma sėkmingai prisijungęs prie jos. Tada administratorius koreguoja *Protege OWL API*, kad per ją būtų galima prisijungti prie SQL serverio. Tuomet sukompiliuojamas algoritmas, kuris vykdo RDB transformaciją į OWL2 ontologiją. Kuriamo algoritmo tinkamumo kriterijus – informacijos nepraradimas transformuojant RDB į ontologiją, kai RDB sukurta iš ontologijos taikant OWL2RDB metodą. Kitaip tariant, pradinės ontologijos atstatymas iš RDB. Toliau administratorius sukuria įskiepį (*plugin*) sąsajai algoritmo su *Protege* įrankiu, kuriame atvaizduojama ontologija.

Vartotojas naudoja sukurtą algoritmą ir transformuoja RDB į OWL2 ontologiją. Sudarytas OWL modelis gali būti papildomas duomenimis, peržiūrimas, patvirtinamas.

Išskirti sudedamieji elementai (3pav.):

- *RDB* – reliacinė duomenų bazė (duomenų modelis);
- *Dalykinė sritis* – sritis, kurioje naudojama sistema.
- *Protege OWL API* – formalus dalykinės srities sąvokų bei jų ryšių aprašymo ir klasifikacijos modelis, sudarytas iš OWL kalbos elementų, besiremiantis *Protege* ontologijų kūrimo ir atvaizdavimo įrankiu.



3 pav. „Transformacijos sistemos veiklos kontekstinė diagrama“

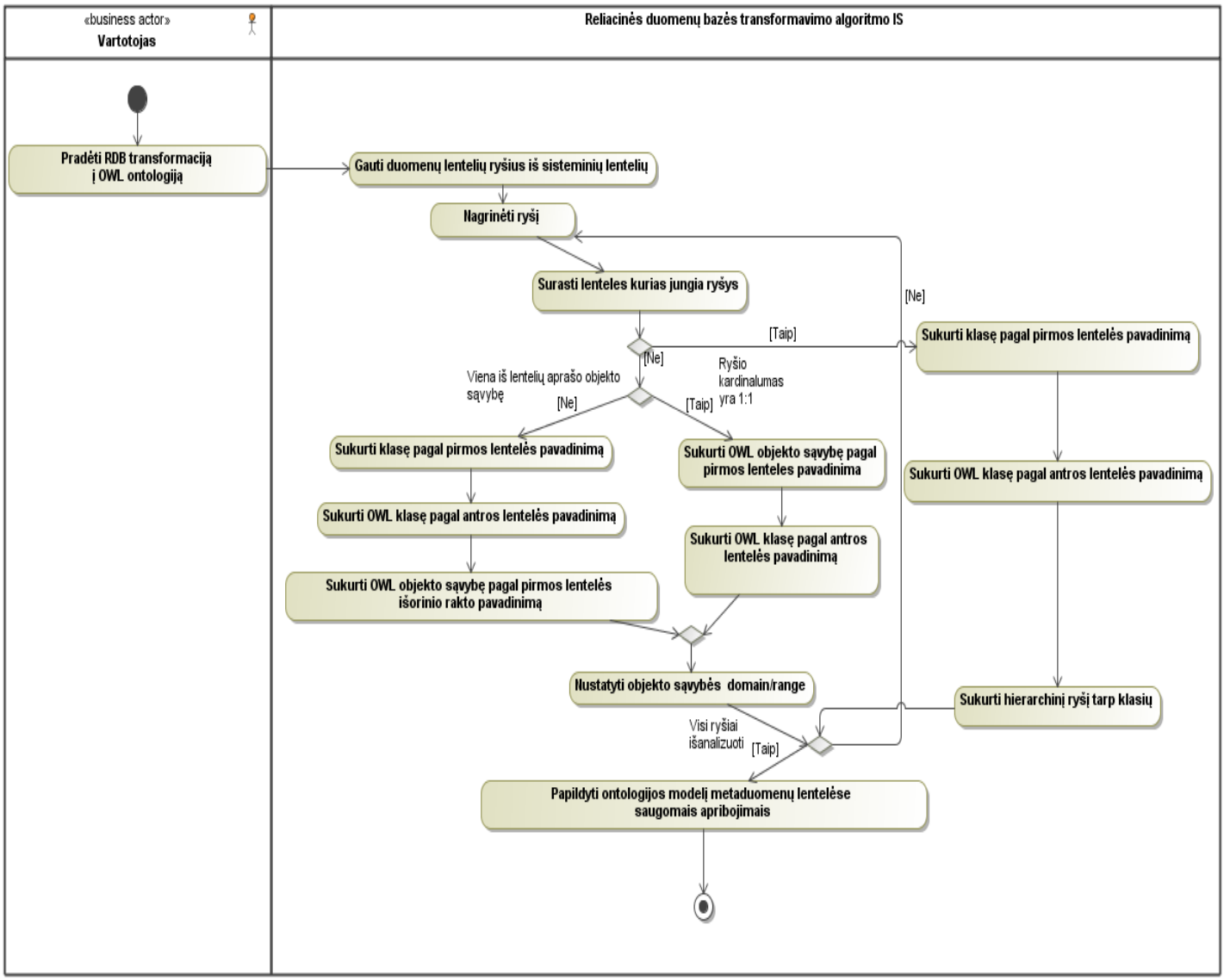
2.6.Darbo tikslas ir uždaviniai

Kuriami algoritmai leis atkurti ontologiją iš duomenų bazės schemos, sukurtos taikant OWL2- RDB metodą. OWL2RDB metodas leidžia išsaugoti daugiau ontologijos savybių, negu kiti. Dalis ontologijos savybių išsaugoma metaduomenų lentelėse. Eksperimentinio tyrimo metu bus patikrintos ontologijos atstatymo galimybės sukuriant ir realizuojant tam skirtą algoritmą, leidžiantį atstatyti ontologiją iš duomenų bazės be informacinių nuostolių.

2.7.Kompiuterizuojamos veiklos procesų modeliai

2.7.1. Ontologijų transformavimo procesas. Vartotojas ir IS.

Ontologijų transformavimo proceso modelis gali būti nusakomas veiklos schema (4pav.)



4 pav. Veiklos procesas. A.Šukio algoritmas atvaizduotas veiklos diagrama.

Vartotojas pateikia RDB transformacijos į ontologiją užklausą reliacinės duomenų bazės transformavimo algoritmo informacinei sistemai. Transformavimo algoritmo informacinė sistema paima duomenų lentelių ryšius iš sisteminių lentelių ir juos nagrinėja. Suradus lenteles

kurias jungia ryšys sukuria klasę pagal pirmos lentelės pavadinimą. Tada sukuria OWL klasę pagal antros lentelės pavadinimą (abiems atvejais kai ryšio kardinalumas yra 1:1 ir ne).

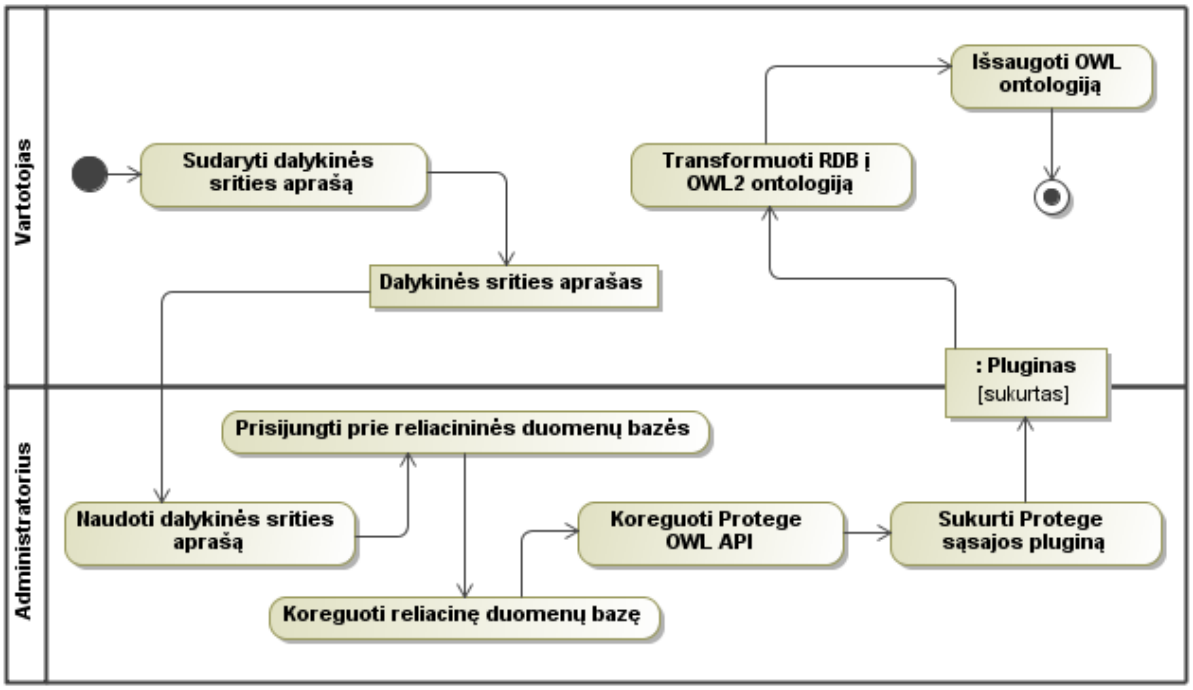
Kai ryšio kardinalumas vienas su daug tada sukuriama OWL objekto savybė pagal pirmos lentelės išorinio rakto pavadinimą. Tada nustatomos objekto savybės domain/range. Jeigu visi ryšiai išanalizuoti tada papildomas ontologijos modelis metaduomenų lentelėse saugomais apribojimais, jeigu ne visi ryšiai tada operacijos vykdomos iš naujo.

Kada ryšio kardinalumas vienas su vienu sukuriamas hierarchinis ryšys tarp klasių. Išanalizavus visus ryšius ontologijos modelis papildomas metaduomenų lentelėse saugomais apribojimais, jeigu neišanalizavus visų ryšių, tada operacijos vykdomos iš naujo.

2.7.1. Ontologijų transformavimo procesas. Vartotojas ir administratorius

Vartotojas pateikia IS administratoriui dalykinės srities specifikacijas, pavyzdžiui, duomenų bazės aprašą, preliminarią duomenų bazę arba panašią ontologiją.(5 pav.) Administratorius, naudodamas reliacinės bazės administravimo įrankį (*SQL Server 2005; SQL Server 2008* ar kt.), sukuria duomenų bazę pagal duomenų bazės aprašą, tinkančią OWL2 ontologijos transformavimo algoritmui. Koreguoja *Eclipse* įrankį, kad šiam pavyktų prisijungti per JDBC jungtį prie *SQL serverio* ir nuskaityti reliacinę duomenų bazę. Sukuria transformacijos algoritmą taikant *edu.stanford.smi.protegex.owl.model* bibliotekas iš *Protege3.5 OWL API*. Sukuria įskiepį *(plugin) Eclipse* programoje sukurto *java* kodo sąsajai su *Protege4.2* programa.

Vartotojas naudojasi programa, kreipiasi į duomenų bazę ir atlieka transformacijos RDB į OWL2 ontologiją algoritmą. *Protege4.2* programoje vartojas gali redaguoti ontologiją bei ją išsaugoti (5 pav.).



5 pav. Vartotojo ir administratoriaus bendradarbiavimas atvaizduotas veiklos diagrama.

2.8.Rizikos faktorių analizė

Atvaizduotu procesų metu (5 pav.) susiduriama su rizikomis

- Vartotojas pateikia IS administratoriui netikslias dalykinės srities specifikacijas.
- Sunkiai transformuojamos reliacinės duomenų bazės.
- Gali būti sunkiai suderinamos DBVS.
- Kuriama algoritmo realizacija priklauso nuo integravimui pasirikto įrankio. Nepakankamai dokumentuoti arba seni *(Protege3.5 OWL API)* įrankio realizacijos komponentai kelia grėsmę transformacijos sistemos integravimui ir projekto įgyvendinimui. Į tai privalo būti atsižvelgta.
- Sistema naudos integracinės terpės komponentą. Todėl algoritmo sudėtingumas bei sprendimo laikas tiesiogiai priklauso nuo šio komponento realizacijos

Išvados

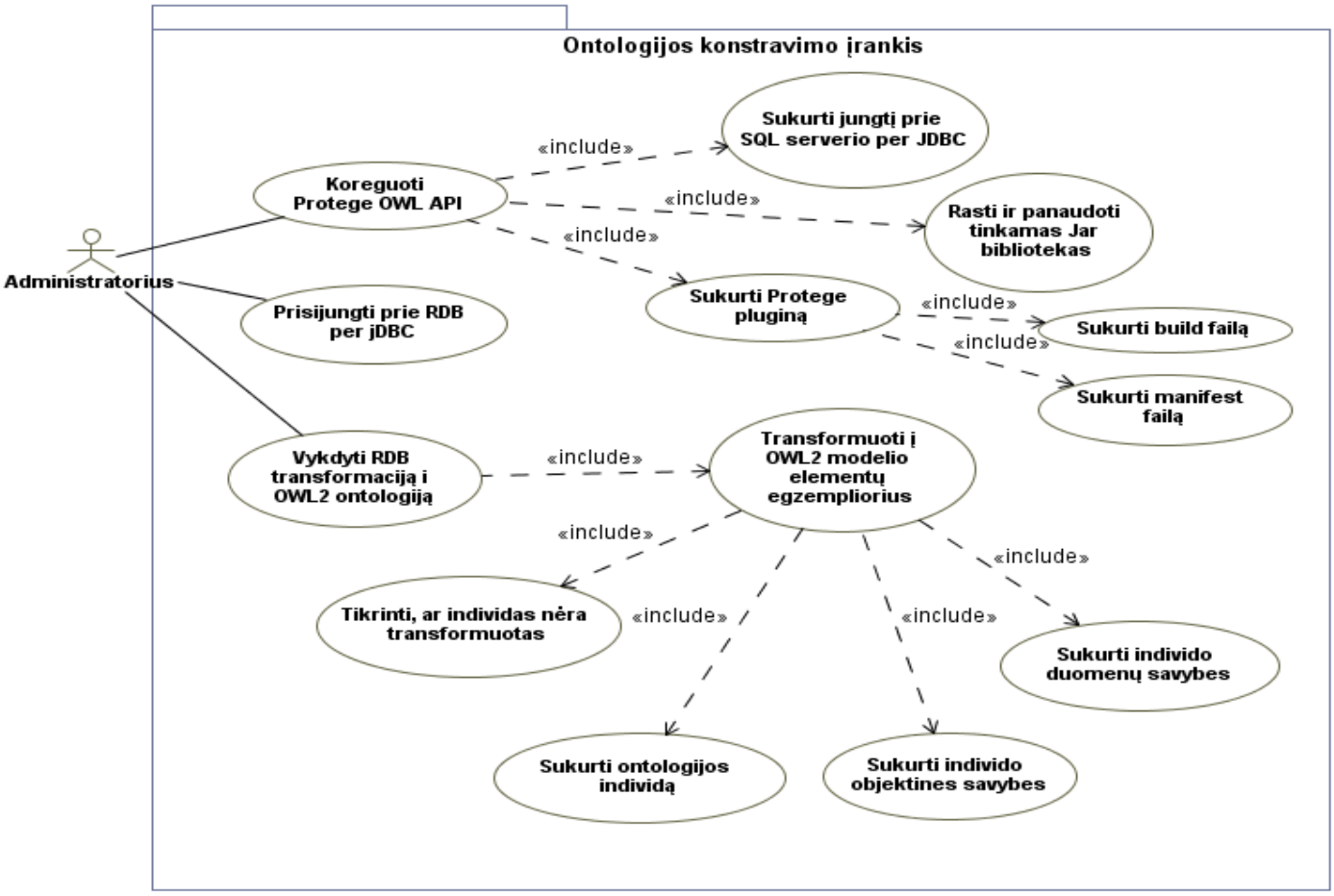
1) Ištirta keletas panašių įrankių kaip: Cullot (2007) RDB vaizdavimo ontologijose metodas (erdvinės informacijos integravimas į RDB); R2O (XML pagrįsta deklaratyvi kalba RDB elementų ontologijos siejimams vaizduoti); RDBToOnto ir kiti. Nustatyta, kad nei vienas neatitinka RDB2-OWL metodo poreikių.
2) Buvo nustatyti šių įrankių vaizdavimo išsamumas, efektyvumas, automatizavimo lygis, schemos pastovumas, pritaikomumas.
3) Nustatytos vartotojo ir administratoriaus bendradarbiavimo proceso metu ir algoritmo kūrimo metu galinčios iškilti rizikos.

Tikimasi, kad kuriamas eksperimentinis algoritmas leis atstatyti ontologiją iš duomenų bazės be informacijos nuostolių.

3. Ontologijų konstravimo reikalavimai

Ontologijos konstravimo sistemos funkciniai reikalavimai pavaizduojami panaudojimo atvejų diagrama ir aprašomi PA specifikacijomis. Įrankis turėtų nuskaityti reliacinių duomenų bazę bei konstruoti ontologiją naudojant OWL2 konstravimo algoritmą. Vartotojas turėtų turėti galimybę sukurti objektus bei redaguoti dalykinės srities ontologiją. Pagrindinė sistemos savybė – reliacinių bazių aprašų eksportavimas į ontologijos aprašus. Įrankio pagalba galima nuskaityti reliacinę duomenų bazę, išsaugotą ontologiją bei atvaizduoti modelio struktūrą OWL. Panaudojimo atvejų specifikacijos pateiktos:

- **Klaida! Nerastas nuorodos šaltinis.**
- **Klaida! Nerastas nuorodos šaltinis.**
- Lentelė 5 PA „Vykdyti RDB transformaciją i OWL2 ontologiją“



6 pav. Ontologijos konstravimo įrankio PA diagrama

Lentelė 3: PA „Koreguoti Protege OWL API“ specifikacija

PA „Koreguoti *Protege* OWL API“	
Tikslas	Koreguoti *Protege* OWL API
Aprašymas	Administratorius pasileidęs *Eclipse* programą, turi surasti ir panaudoti visas reikiamas bibliotekas esančias .jar failuose.
Aktorius	Administratorius
Sistema	OWL2 ontologijos modeliavimo įrankis
Prieš sąlyga	Nėra išsaugota reikiama įrankio konfigūracija. Inicijuojamas *Protege OWL* modelio bibliotekų tvarkymas.
Susiję panaudojimo atvejai	„Sukurti Protege pluginą“ PA, „Prisijungti prie RDB per JDBC“ PA, „Sukurti jungtį prie SQL serverio per JDBC“ PA
Pagrindinis įvykių srautas	**Sistemos reakcija ir sprendimai**
1. Administratorius atsidaręs *Protege* instaliacijos direktoriją suranda reikiamus .jar failus. 2. Administratorius .jar failus išsaugo *Eclipse* projekto lib direktorijoje. 3. Pasirenka *Eclipse* programoje *java build path* į reikiamas bibliotekas. 4. Sukuriama jungtis prie SQL serverio. 5. Sukuriamas build failas 6. Sukuriamas manifest failas 7. Sukuriamas pluginas.	1. Atveriama bibliotekų instaliacijos direktorija. 2. Sistema atlieka bibliotekų išsaugojimą savo aplanko direktorijoje. 3. Sistema naudoja bibliotekas OWL modelio kūrimui. 4. Sistema gauna prisijungimo prie SQL serverio variklį *(driver).* 5.Sistema gauna reikiamą build failą plugino kūrimui. 6. Sistema gauna reikiamą manifest failą plugino kūrimui. 7.Sistema kuria pluginą.
Po sąlyga	Išsaugota aktyvi transformacijos sistemos konfigūracija.
Alternatyvūs scenarijai 1a.Parenkamos netinkamos bibliotekos 1b.Nenurodomi bibliotekų *class path.*	1a.1 Sistemai nepavyksta gauti OWL modelio kūrimo šablonų. 1b.1 Sistemai nepavyksta gauti OWL modelio kūrimo šablonų.
Veiklos taisyklės	1. Vartotojas privalo nurodyti reikiamas bibliotekas. 2. Vartotojas privalo nurodyti teisingus konfigūravimo parametrus.
Kitos sistemos, su kuriomis sąveikauja sistema vykdydama PA	Vartotojo pasirinkta duomenų bazių valdymo sistema.

Lentelė 4: PA „Pasirinkti RDB“ specifikacija

PA „Pasirinkti RDB“	
Tikslas	Prisijungti prie RDB
Aprašymas	Administratorius pasileidęs programą, turi suvesti prisijungimo duomenis, kad galėtų gauti priėjimą prie RDB.

Aktorius	Administratorius
Sistema	OWL2 ontologijos modeliavimo įrankis
Prieš sąlyga	Nėra išsaugota aktyvi įrankio konfigūracija. Inicijuojamas OWL2 transformacijos algoritmas.
Susiję panaudojimo atvejai	„Suvesti prisijungimo duomenis“ PA
Pagrindinis įvykių srautas	**Sistemos reakcija ir sprendimai**
1. Vartotojas naudojasi papildomu modeliavimo įrankio plėtiniu ir pasirenka transformuoti reliacinę duomenų bazę į OWL2 ontologiją. 2. Vartotojas nurodo prisijungimo prie duomenų bazės parametrus. 3. Vartotojas pasirenka konfigūravimo parametrus.	1. Vartotojui parodomas langas, kuriame galima nurodyti prisijungimo prie duomenų bazės konfigūravimo parametrus. 2. Sistema atlieka konfigūracijos išsaugojimą atmintyje.
Po sąlyga	Išsaugota aktyvi transformacijos sistemos konfigūracija.
Alternatyvūs scenarijai 1a.Nurodyti klaidingi konfigūracijos, prisijungimo duomenys. 1b.Vartotojas neprisijungęs	1a.1 Sistemai nepavyksta prisijungti prie reliacinės duomenų bazės. 1b.1 Sistemai nepavyksta prisijungti prie reliacinės duomenų bazės.
Veiklos taisyklės	3. Vartotojas privalo nurodyti teisingą duomenų bazių valdymo sistemos serverio adresą. 4. Vartotojas privalo nurodyti teisingus konfigūravimo parametrus.
Kitos sistemos, su kuriomis sąveikauja sistema vykdydama PA	Vartotojo pasirinkta duomenų bazių valdymo sistema.

Lentelė 5: PA „Vykdyti RDB transformaciją i OWL2 ontologiją“

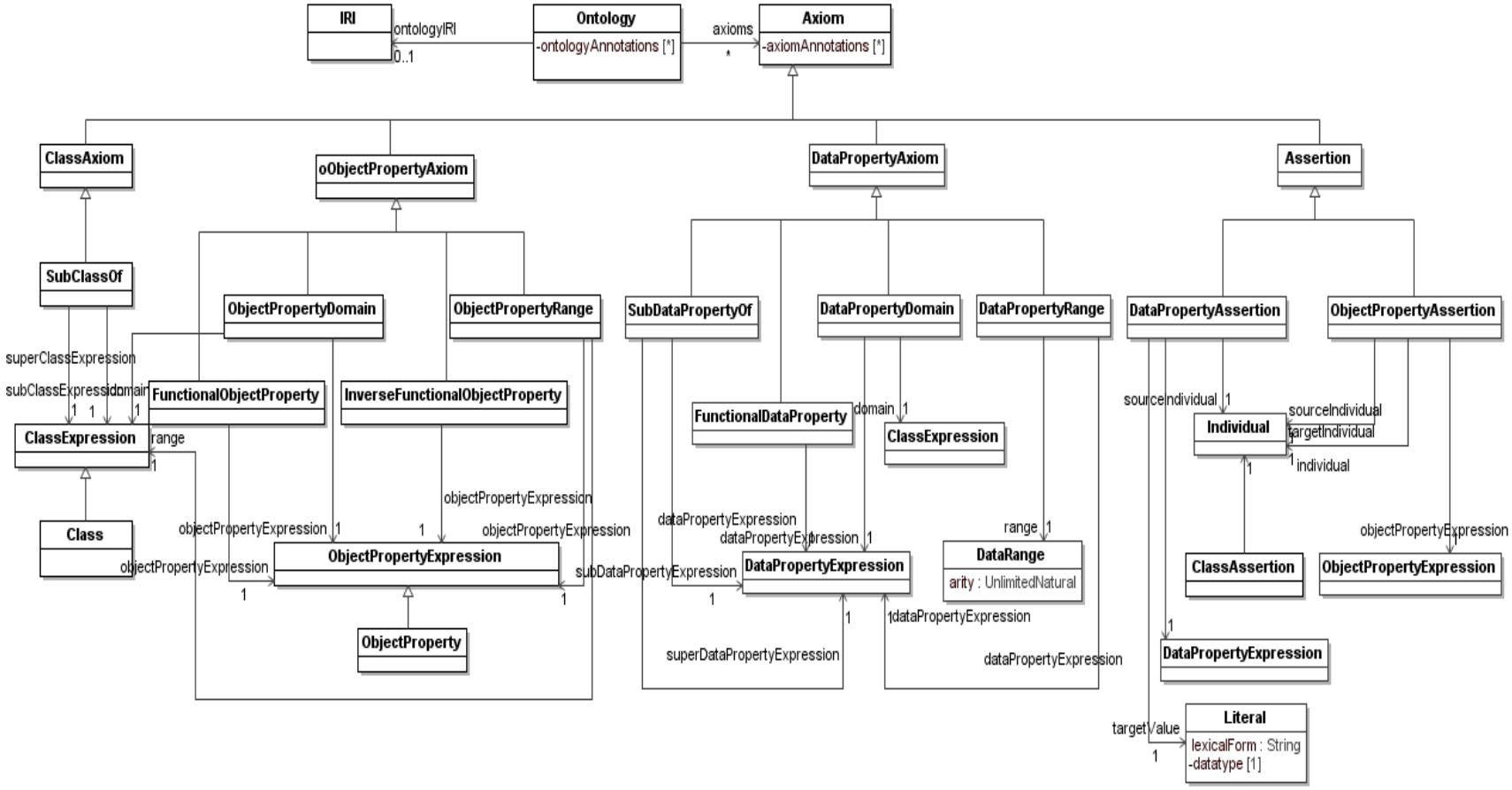
PA „Vykdyti RDB transformaciją i OWL2 ontologiją“	
Tikslas	Transformuoti į OWL2 modelio elementų egzempliorius
Aprašymas	Administratorius prisijungęs prie reliacinės duomenų bazės, peržiūri ją, jei reikia koreguoja, kad būtų galima taikyti OWL2 ontologijos transformavimo algoritmą ir pritaiko jį.
Aktorius	Administratorius
Sistema	OWL ontologijos modeliavimo įrankis
Prieš sąlyga	RDB yra atvaizduojama (aktyvi) modeliavimo įrankyje. Inicijuojamas RDB transformavimas į OWL2 ontologiją.
Susiję panaudojimo atvejai	„Transformuoti į OWL2 modelio elementų egzempliorius” PA
Pagrindinis įvykių srautas	**Sistemos reakcija ir sprendimai**

1. Vartotojas nurodo prisijungimo prie duomenų bazės parametrus. 2. Vartotojas naudojasi modeliavimo įrankiu ir koreguoja aktyvią RDB 3. Vartotojas naudojasi papildomu modeliavimo įrankio plėtiniu ir pasirenka konstruoti ontologiją iš reliacinės duomenų bazės. 4. Vartotojas pasirenka konfigūravimo parametrus.	1. Vartotojui parodomas langas, kuriame galima nurodyti prisijungimo prie duomenų bazės, konfigūravimo parametrus . 2. Sistema prisijungia prie duomenų bazių valdymo sistemos serverio ir duomenų bazės. 3. Sistema atlieka lentelių paiešką. 4. Sistema kuria klases pagal lentelių pavadinimus. 5. Sistema kuria objektų savybes. 6. Sistema atlieka individų, individų savybių, individo duomenų savybių transformavimą į OWL2 ontologiją.
Po sąlyga	Ontologija sukonstruota iš reliacinės duomenų bazės.
Alternatyvos (nesėkmės atvejai) 1. Duomenų bazių valdymo sistemos serveris neveikia. 2. Sistemai nesuteiktos teisės prisijungti prie duomenų bazių valdymo sistemos serverio. 3. Vartotojas nurodo neteisingus prisijungimo duomenis 4. Nepavyko sudaryti bent vieno iš modelių (elementų, egzempliorių, metaduomenų)	1. Sistemai nepavyksta prisijungti prie reliacinės duomenų bazės. 2. Sistemai nepavyksta prisijungti prie reliacinės duomenų bazės. 3.Sistemai nepavyksta prisijungti prie reliacinės duomenų bazės. 4. Sistema nesukuria ontologijos be informacijos nuostolių.
Veiklos taisyklės	1. Vartotojas privalo nurodyti teisingą duomenų bazių valdymo sistemos serverio adresą. 2. Sistemai turi būti suteiktos pilnos teisės dirbti su duomenų baze.
Nefunkciniai reikalavimai	Sinchronizavimas turi būti galimas į įvairias duomenų bazių valdymo sistemas
Kitos sistemos, su kuriomis sąveikauja sistema vykdydama PA	Vartotojo pasirinkta duomenų bazių valdymo sistema. OWL modelio valdiklis, RDB modelio, RDB metaduomenų modelio valdikliai.

4. Ontologijos konstravimo iš RDB algoritmo projektas

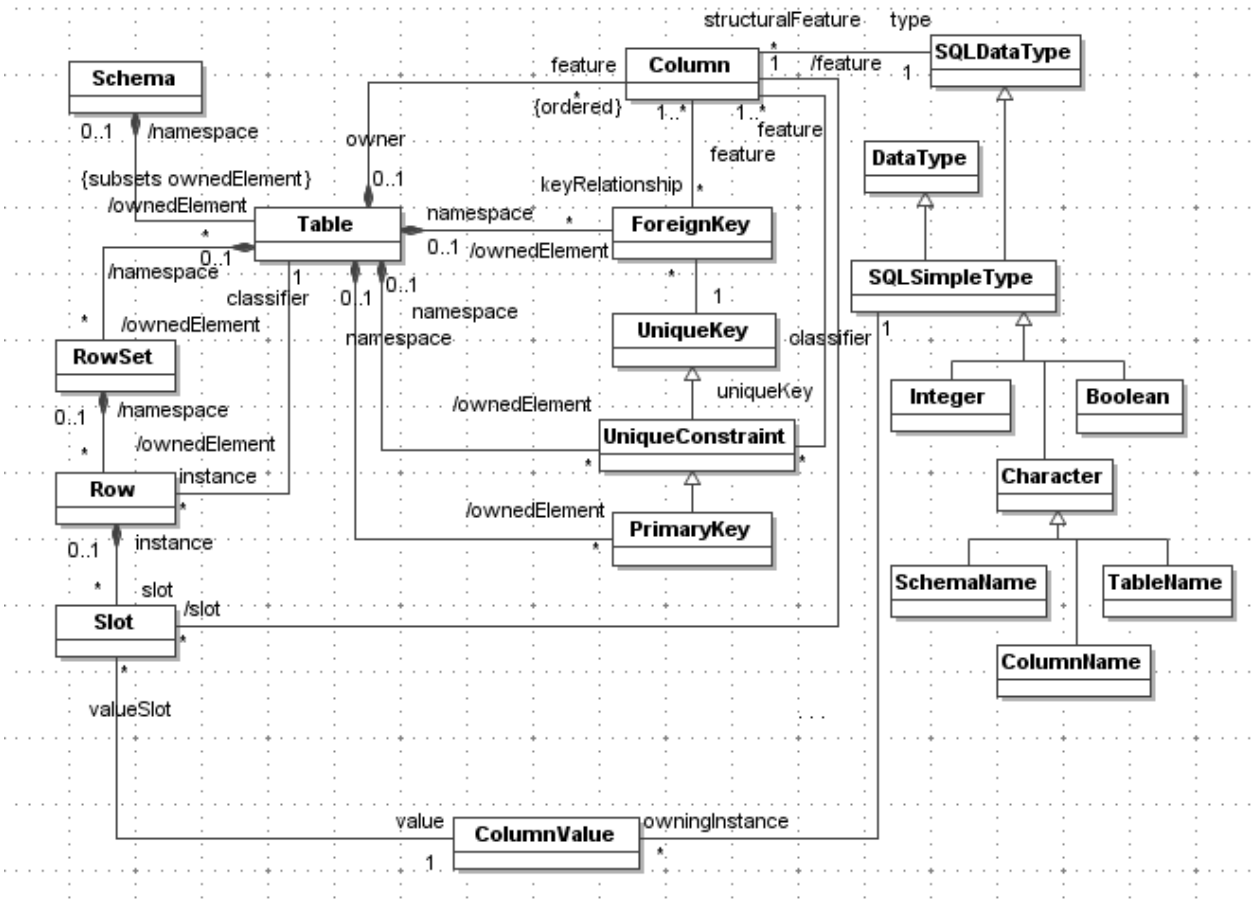
4.1.Dalykinės srities modelis

Transformacijos algoritmo aprašymui naudojami metamodelių fragmentai. Aukščiausio lygio OWL2 metamodelį sudaro elementai : *ontologija*, *aksiomos* 7 pav. *Aksioma* yra pagrindinis elementas apibrėžiant OWL2 semantines konstrukcijas. OWL2 struktūriniai elementai - *Esybės (klasės, objektinės savybės, anotacijų savybės, duomenų savybės, individai, duomenų tipai),* sudaro ontologijos žodyną ir yra identifikuojami pagal IRI.



7 pav. OWL Metamodelis atvaizduotas klasių diagrama

Pagrindiniams reliacinės duomenų bazės konceptams modeliuoti panaudosime metamodelį (8 pav.). Stulpelis (*Column*) reliaciniame metamodelyje atitinka bet kokis duomenis esančius slote (*Slot*). Slotas (Slot) yra eilutėje (Row). Kelios eilutės sudaro eilučių rinkinį (RowSet). Eilučių rinkinys ir stulpelių rinkinys yra fiziniai duomenų bazės lentelės (*Table*) objektai. Lentelių rinkinys laikomas schema. Duomenų išraiška (Slot) taip pat yra ir stulpelyje, kur atitinka stulpelio reikšmę (ColumnValue). Stulpeliams yra priskiriamas susietas duomenų tipas (*SQLDataType*).(8 Pav.) pavaizduotas pagrindinis duomenų tipas -paprastasis (*simpe*). Paprastieji tipai yra apibrėžiami SQL standarto. Paprastieji duomenų tipai nusako duomenų rūšį (*DataType).* Individas (*Character*) sudarytas iš lentelės pavadinimo (*TableName*) ir schemos vardo (*SchemaName*). Individas turi objektines savybes (*Integer, Boolean)* bei duomenų savybes (*DataType*).



8 pav. RDB Metamodelis atvaizduotas klasių diagrama

4.2.Reikalavimų realizacija

Sudarytas transformacijos algoritmo analizės klasių modelis (9 pav.). Šio modelio tikslas padėti išanalizuoti sistemai keliamus reikalavimus. Vartotojo sąsajai keliami reikalavimai atvaizduoti ribinėmis klasėmis (<<*boundary*>> ):

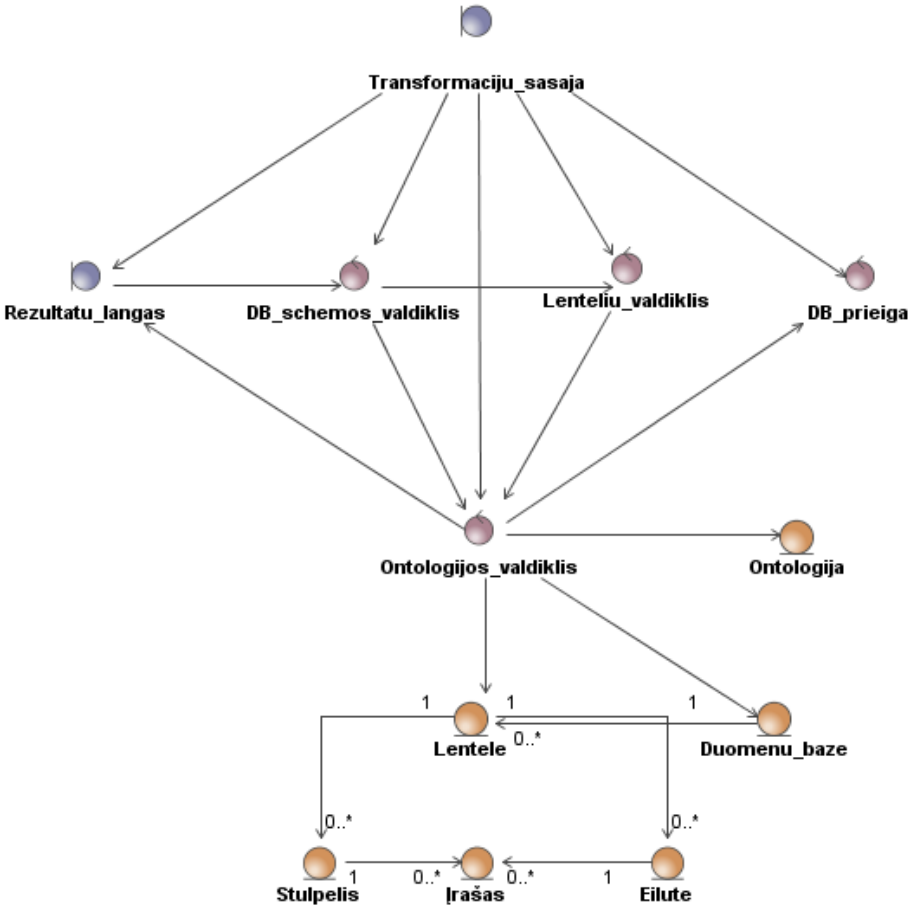
- Transformacijų sąsaja;
- Rezultatų langas;
- Eclipse langas;

Atlikti transformacijos algoritmą reikalingi valdikliai. Sukurtos valdiklių klasės (<<*control*>> ):

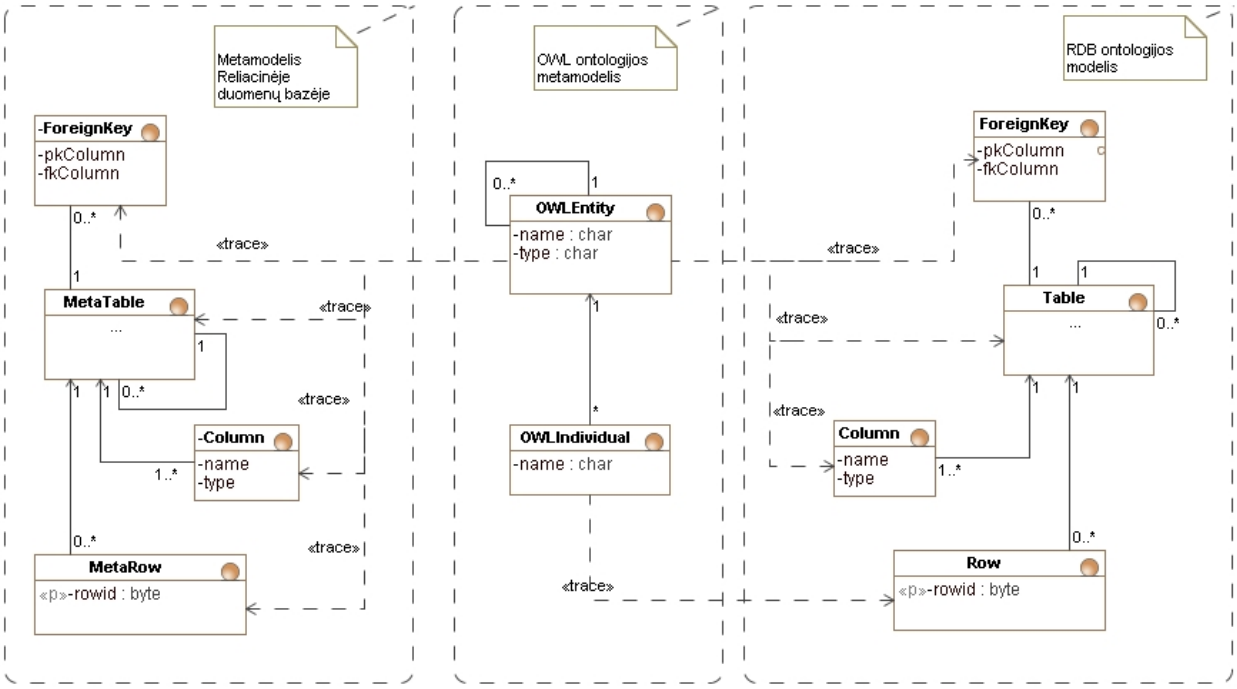
- DB schemos valdiklis;
- Lentelių valdiklis.
- DB prieigos valdiklis;
- Ontologijos valdiklis;
- *Eclipse* valdiklis;
- JDBC valdiklis
- *Protege* plugino valdiklis

Dalykinės srities objektus vaizduoja esybių klasės (<<*entity*>> ).

Panaudojimo atvejis „Redaguoti *Protege OWL API*“(Lentele 3) veiksmai atliekami taip: Administratorius atsiveria programą *Eclipse.* Tuomet parenkama reikiama JDBC biblioteka. JDBC valdiklis sukuria jungtį su SQL serveriu. *Eclipse* lange sukuriamas naujų bibliotekų saugojimo katalogas. Kuriame įdiegiamos visos sistemai reikalingos bibliotekos. Eclipse valdiklis jas naudoja naujo plugino kūrimui. Pirmiausia administratorius per *Eclipse* langą sukuria *build* failą, kuris bus naudojamas kurti pluginą. Šį failą naudoja *Eclipse* valdiklis. Tuomet administratorius sukuria *manifest* failą. Šį failą *Eclipse* taip pat naudoja Eclipse valdiklis. Inicijuojamas plugino sukūrimo procesas.



9 pav. Transformacijos algoritmo analizės modelis

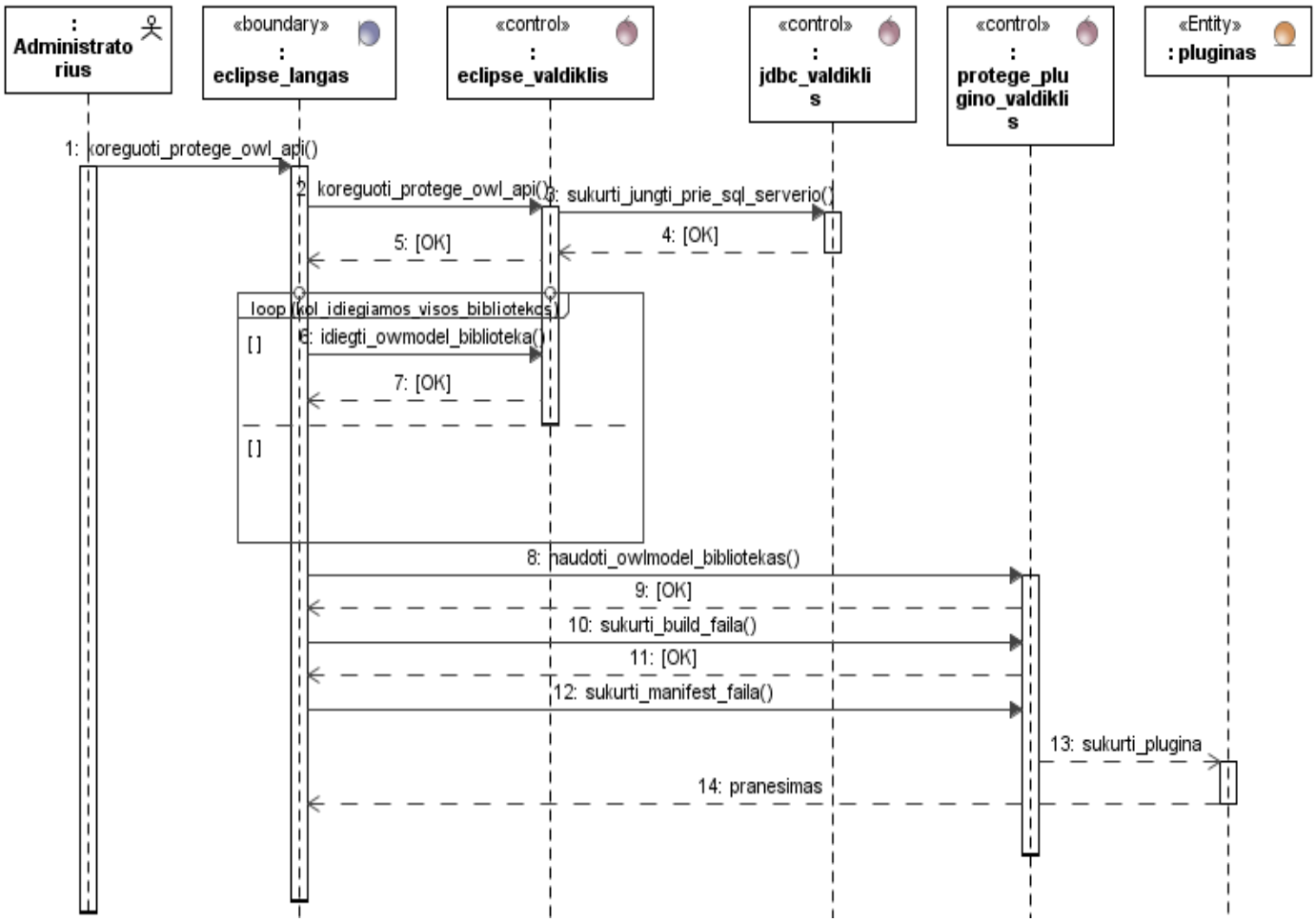


10 pav. *Esybių modelis*

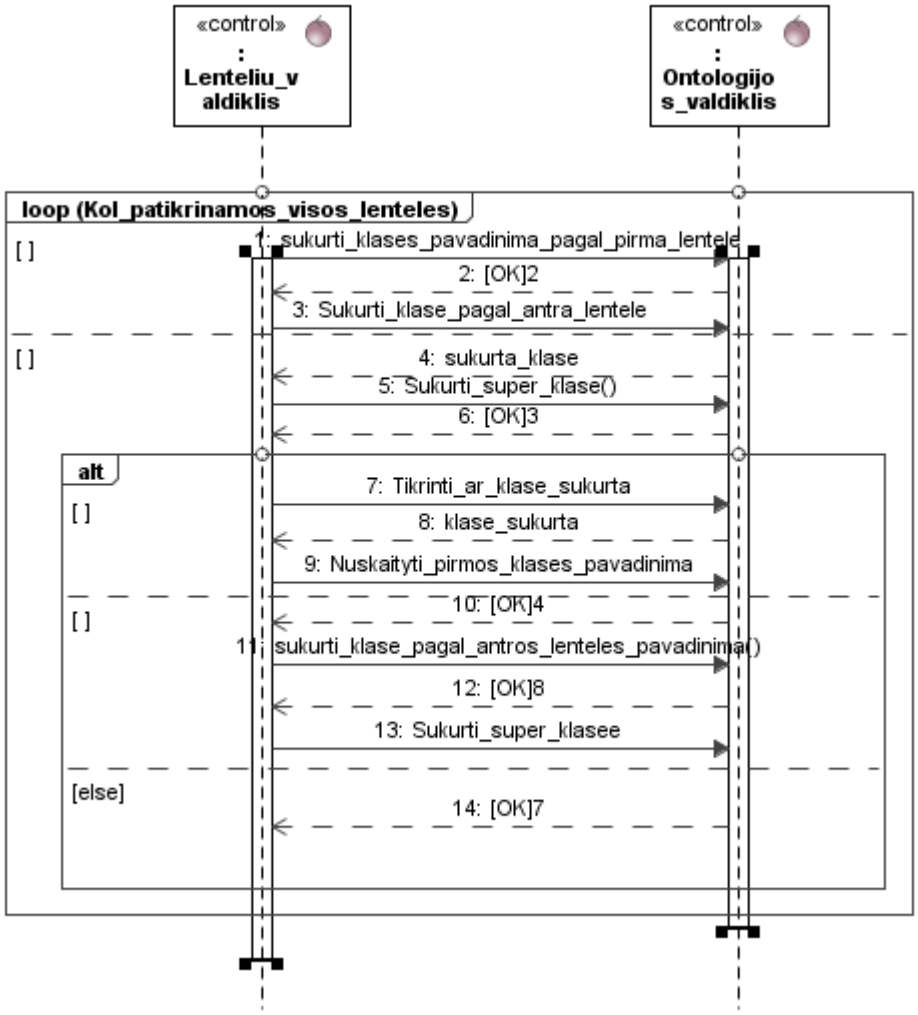
4.3.Panaudojimo atvejų sekų diagramos

Transformacijos sistemos elgsena modeliuojama sekų diagramomis. Analizuojant panaudojimo atvejų modelį(6 pav.) bei panaudojimo atvejų specifikacijose (Lentelė3 –Lentelė5) aprašytus žingsnius, nustatytos analizės klasių operacijos bei detalizuota sąveika tarp valdiklių ir vartotojo sąsajos komponentų.

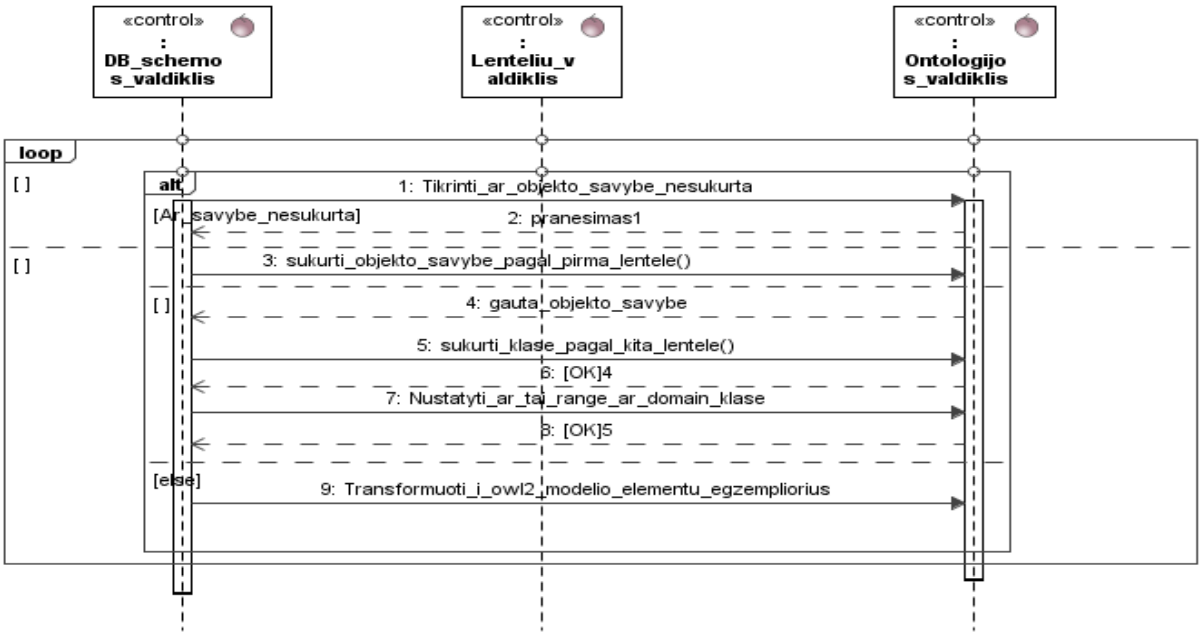
Panaudojimo atvejo „Redaguoti Protege OWL API“ realizacijai pateikiama sekų diagrama (11 pav.).



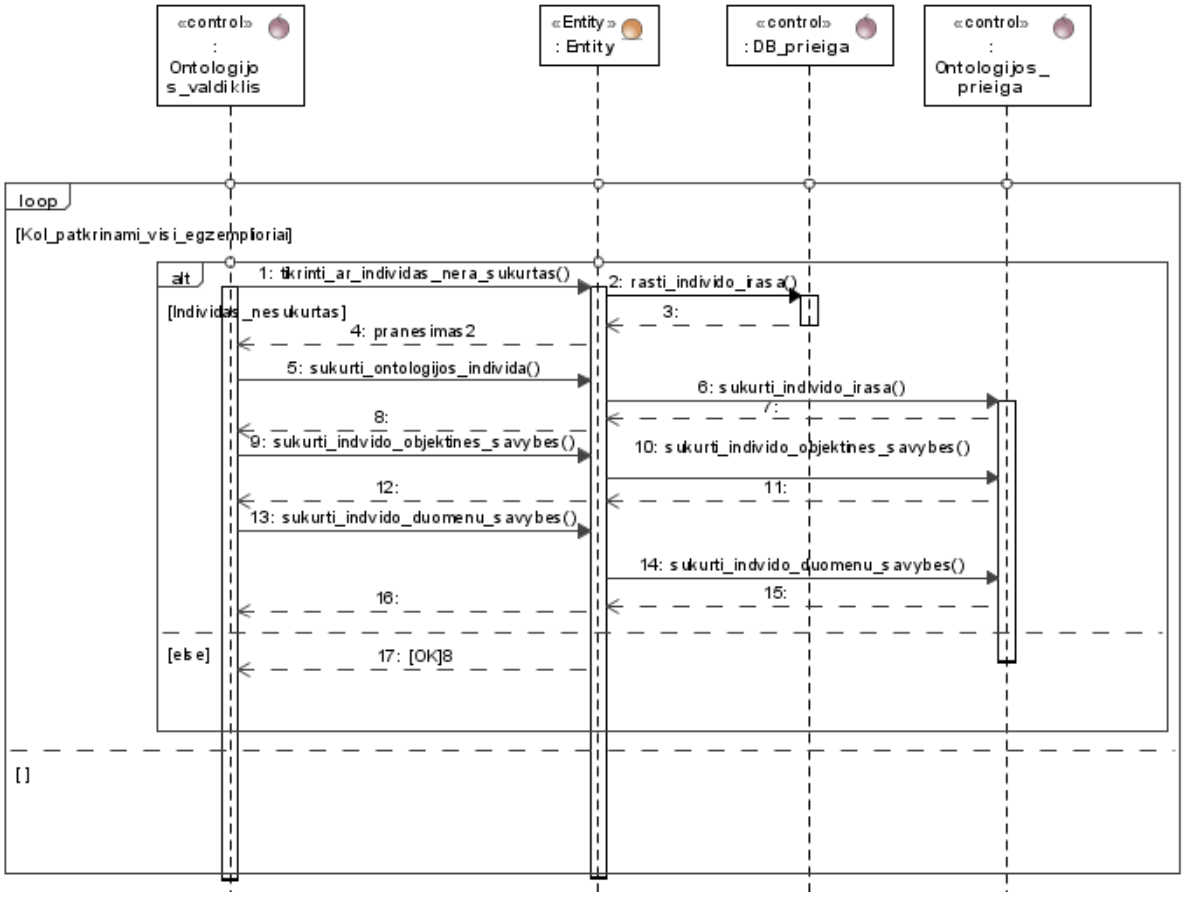
11 pav. „Redaguoti Protege OWL API sekų diagrama“



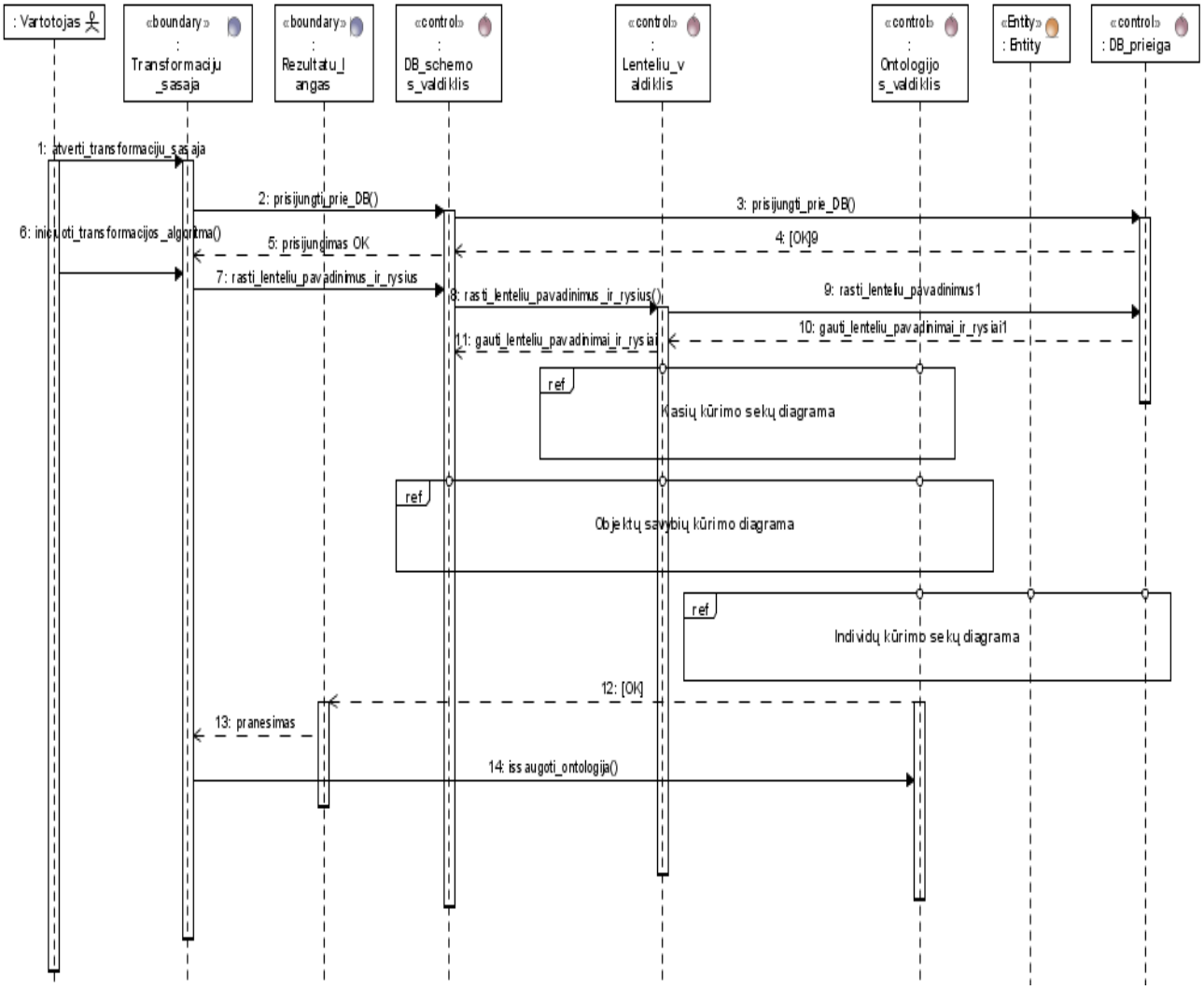
12 pav. „OWL klasių kūrimo sekų diagrama“ naudojama kaip fragmentas



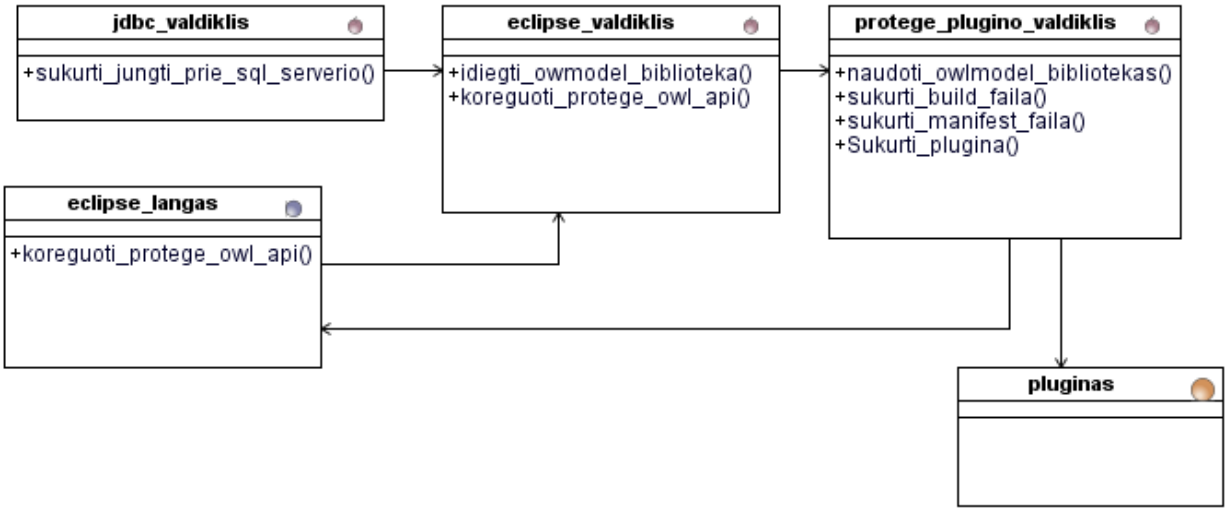
13 **pav.** „OWL objektų savybių kūrimo sekų diagrama“ naudojama kaip fragmentas



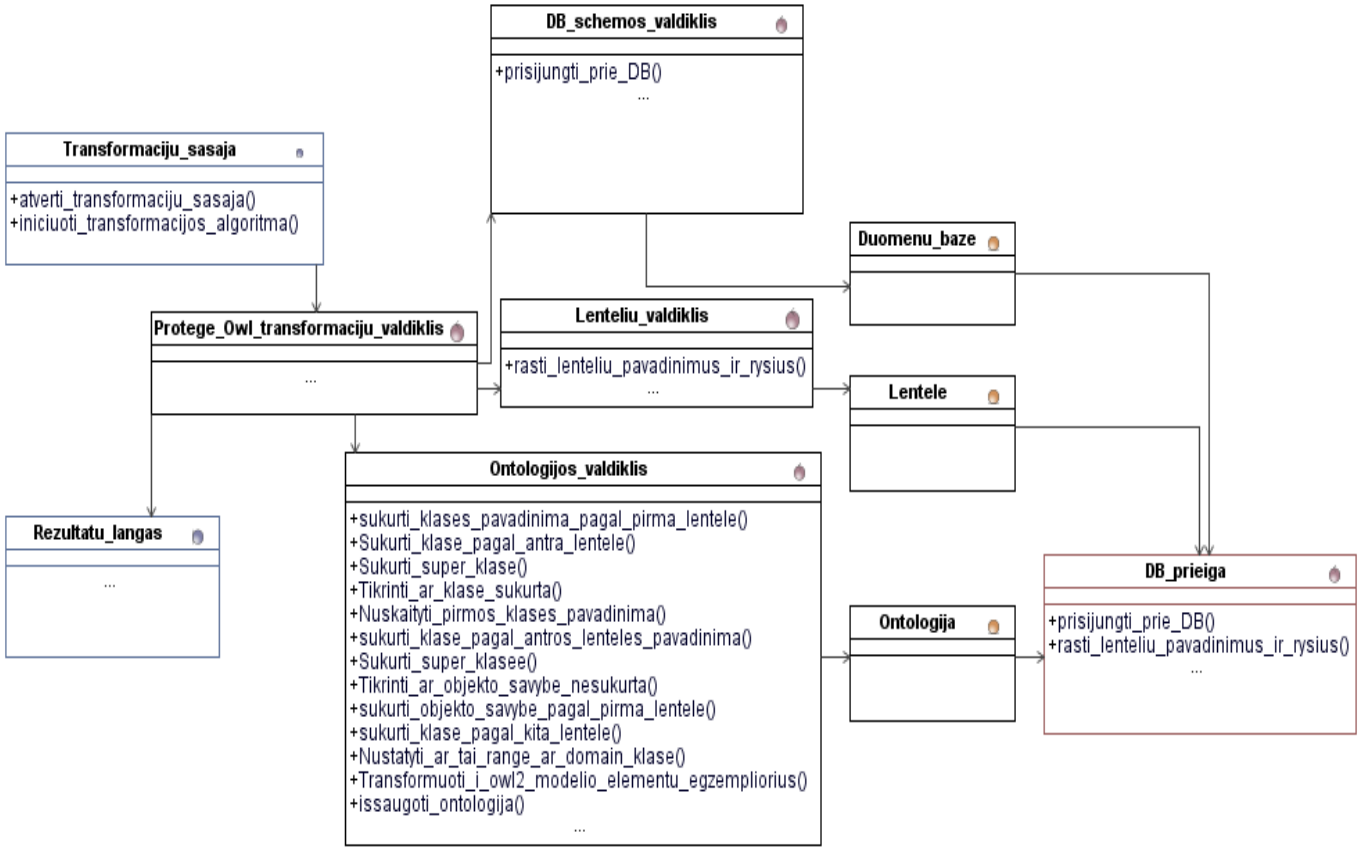
14 pav. „Individo kūrimo sekų diagrama“ naudojama kaip fragmentas



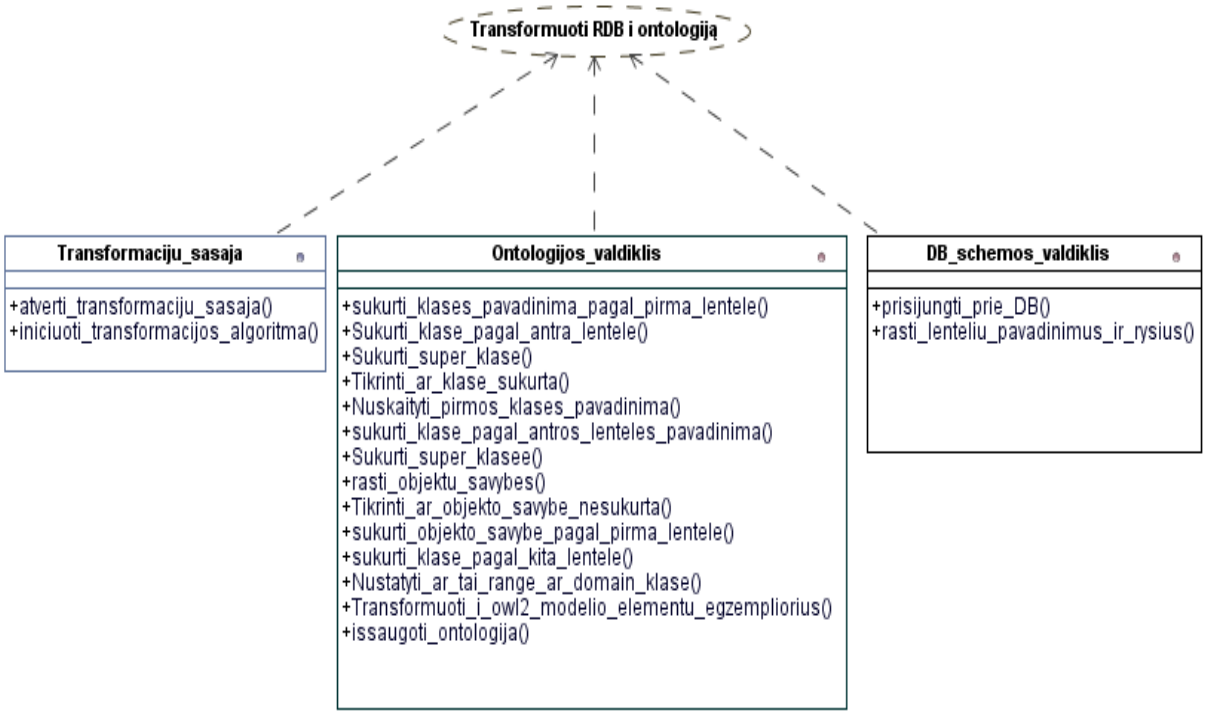
15 pav. Vykdyti RDB transformaciją i OWL2 ontologiją sekų diagrama su nuorodomis į fragmentus



16 pav.„Redaguoti Protege OWL API“ klasių diagrama



17 pav. Transformacijos sistemos veiklos proceso schema



18pav. Vykdyti RDB transformaciją i OWL2 ontologiją klasių diagrama

Sekų diagramose atvaizduotos įvykių sekos realizuojamos programiniame pakete *„Eclipse“ java* kalba. Lentelių valdiklio (12pav.) įvykiai atliekami po to, kai įvykdoma SQL užklausą nustatyti ryšius tarp lentelių (19 pav.).

Connection profile

Type: SQL Server_2005 Name: New SQ Database: ZIiSI_laizolp Status: Connected,

```
USE [ZIiSI_laizolp]
GO
/****** Object:  View [dbo].[table_references]    Script Date: 03/05/2013 21:06:04 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
    CREATE VIEW [dbo].[table_references] AS
    SELECT
        sys.foreign_key_columns.referenced_object_id,
        sysobjects.name,
        parent_column_id,
        referenced_column_id
    FROM
        sys.foreign_key_columns, sysobjects
    WHERE
        sys.foreign_key_columns.parent_object_id = sysobjects.id
```

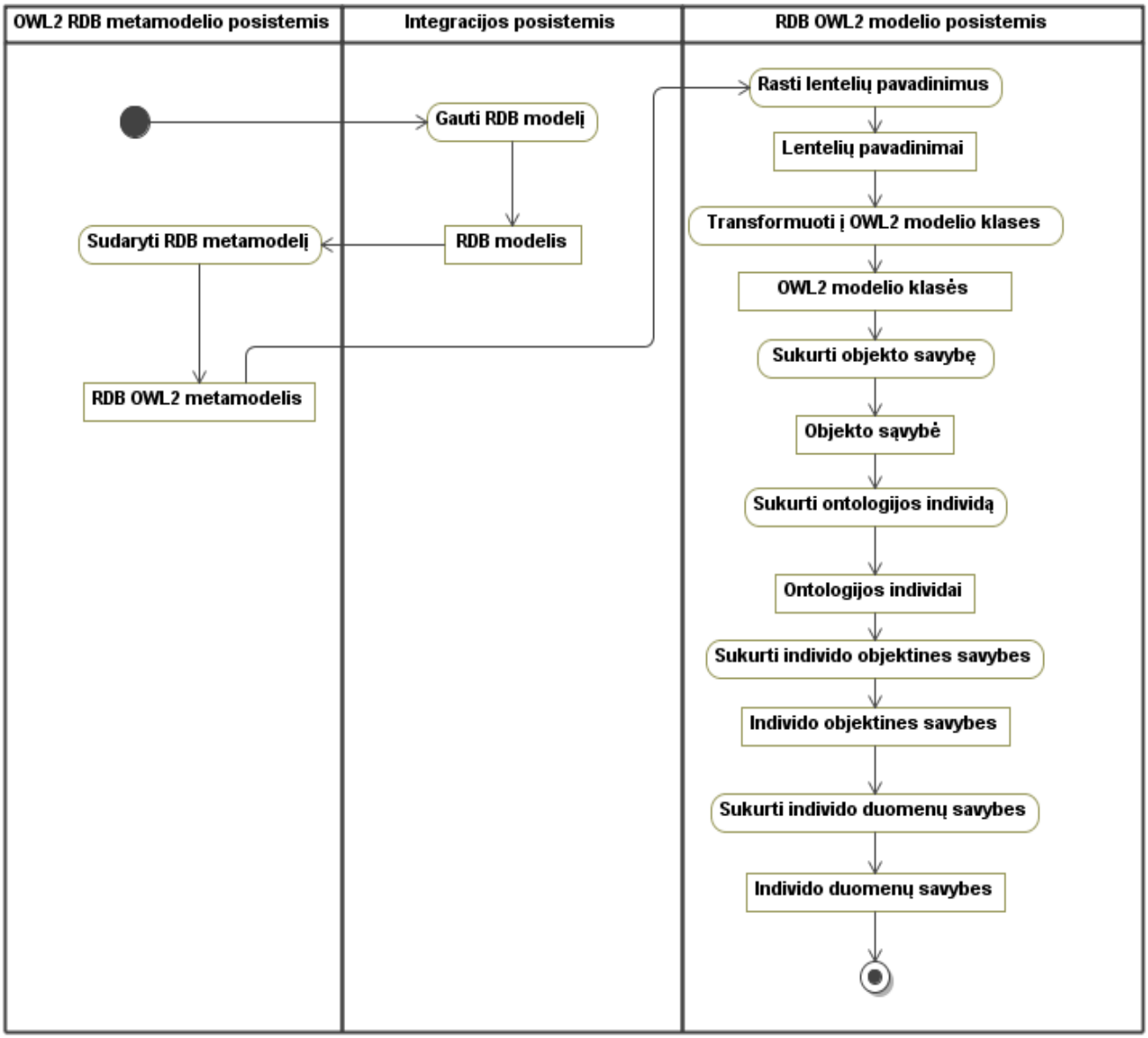
19 pav. RDB vaizdų (view) sukūrimas

PA diagramoje (6 pav.) pavaizduotas procesas prisijungti prie RDB per JDBC jungtį pavaizduotas bendroje sekų diagramoje. Jis neatvaizduotas atskira sekų diagrama, nes yra pakankamai trumpas. Reikia tik suvesti serverio adresą, prisijungimo vardą ir slaptažodį (9 pav.). Tuomet iškviečiame klasė *Class.forName*, kuri panaudoja JDBC variklį. Tada suteikiama prieiga prie DB. Čia naudojamas valdiklis „DB_prieiga“ (15 pav.).

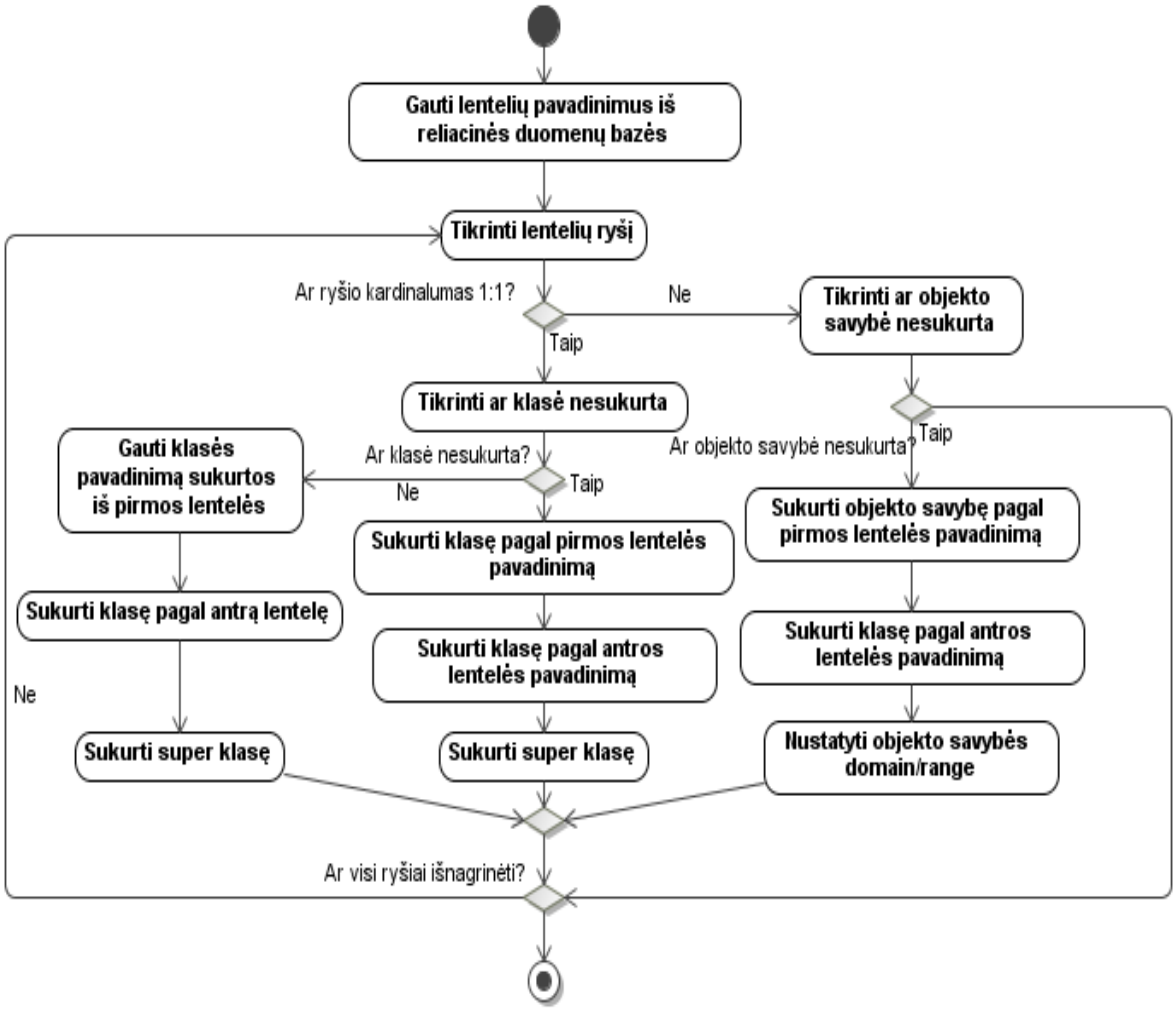
```
public static void main(String[] args) throws OntologyLoadException {

    // prisijungimas prie SQL Server DB su JDBC
     String url = "jdbc:sqlserver://isd.ktu.lt:1433";
     String userName = "ZIiSI_laizolp";
     String password = "b79b6L";

    try {
        Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
        Connection con = DriverManager.getConnection (url, userName, password);
```

20 pav. Prisijungti prie RDB per JDBC

Veiklos procesų modeliais aprašomi RDB OWL2 modelio bei RDB OWL2 metamodelio posistemio komponentai. Integracinio posistemio komponentas *Protege API* yra realizuotas. Bendra transformacijos sistemos veiklos proceso schema pateikta (21 pav.). Analizuojant RDB modelio elementus bei jų ryšius sukuriamas metaduomenų modelis, kuris toliau naudojamas RDB elementų transformacijai. Galutinis proceso rezultatas –išsaugoti ontologijos OWL2 egzemplioriai, individai, individų savybės, individo duomenų savybės.

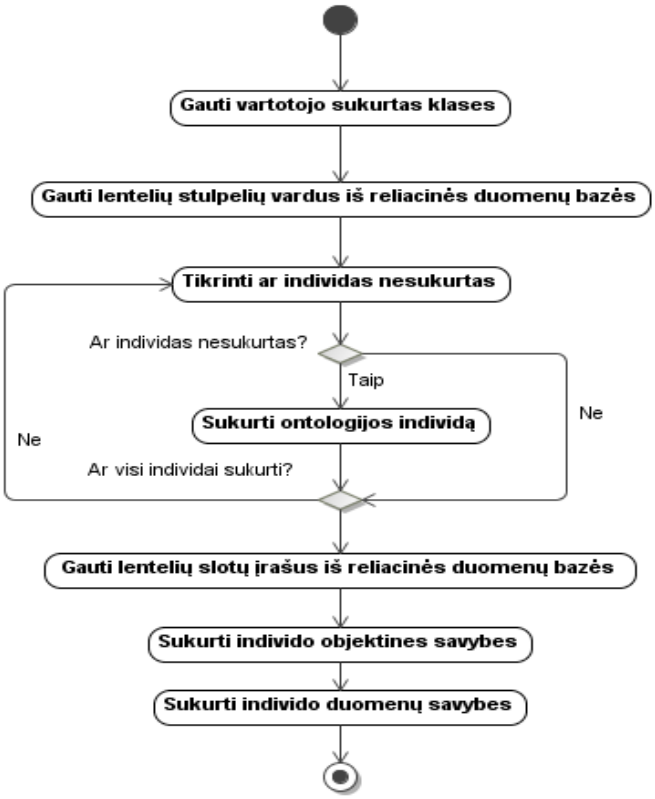


21 pav. Transformacijos sistemos veiklos proceso schema



22 pav. Transformavimo algoritmo pirma dalis. Sukurti klases veiklos diagrama

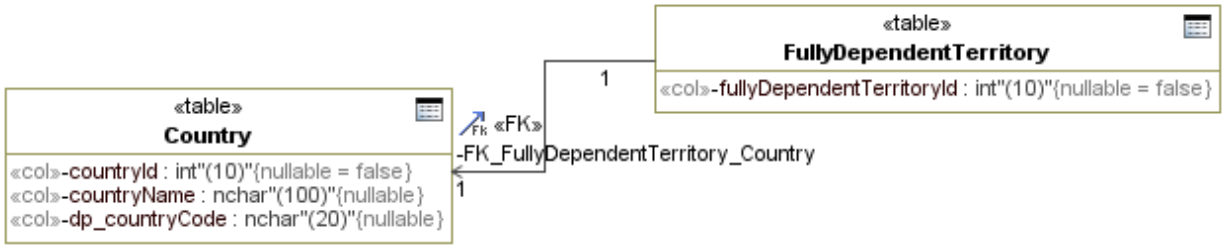
Imamos vartotojo sukurtos klasės iš lentelių pavadinimų ir ryšių (22 pav.) ir klasėms kuriami individai, šiems objektinės bei duomenų savybės.



23 pav. Transformavimo algoritmo antra dalis. Sukurti individus veiklos diagrama**.**

Algoritmas remiasi duomenų bazės sisteminių duomenų analize, t.y. randami ryšiai tarp lentelių, lentelių atributai, pagal juos sukuriamas ontologijos modelis. Žemiau pateikiami algoritmo paaiškinimai:

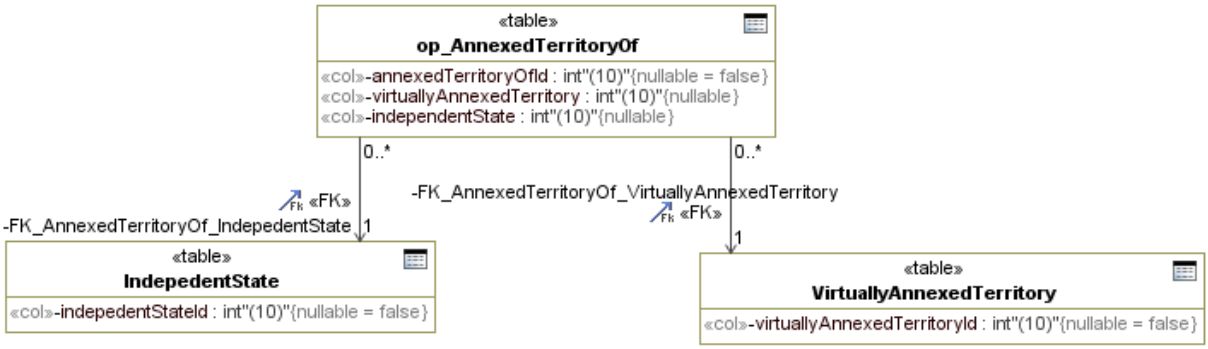
Jeigu ryšio kardinalumas yra 1:1, sukuriamos dvi klasės pagal lentelių pavadinimus bei nustatoma hierarchija tarp jų. (pav.) pateiktame pavyzdyje bus sukurtos klasės FullyDependentTerritory bei Country, Country bus nustatyta kaip tėvinė FullyDependentTerritory klasė.



22 pav. Hierarchinis ryšys tarp duomenų lentelių 1:1

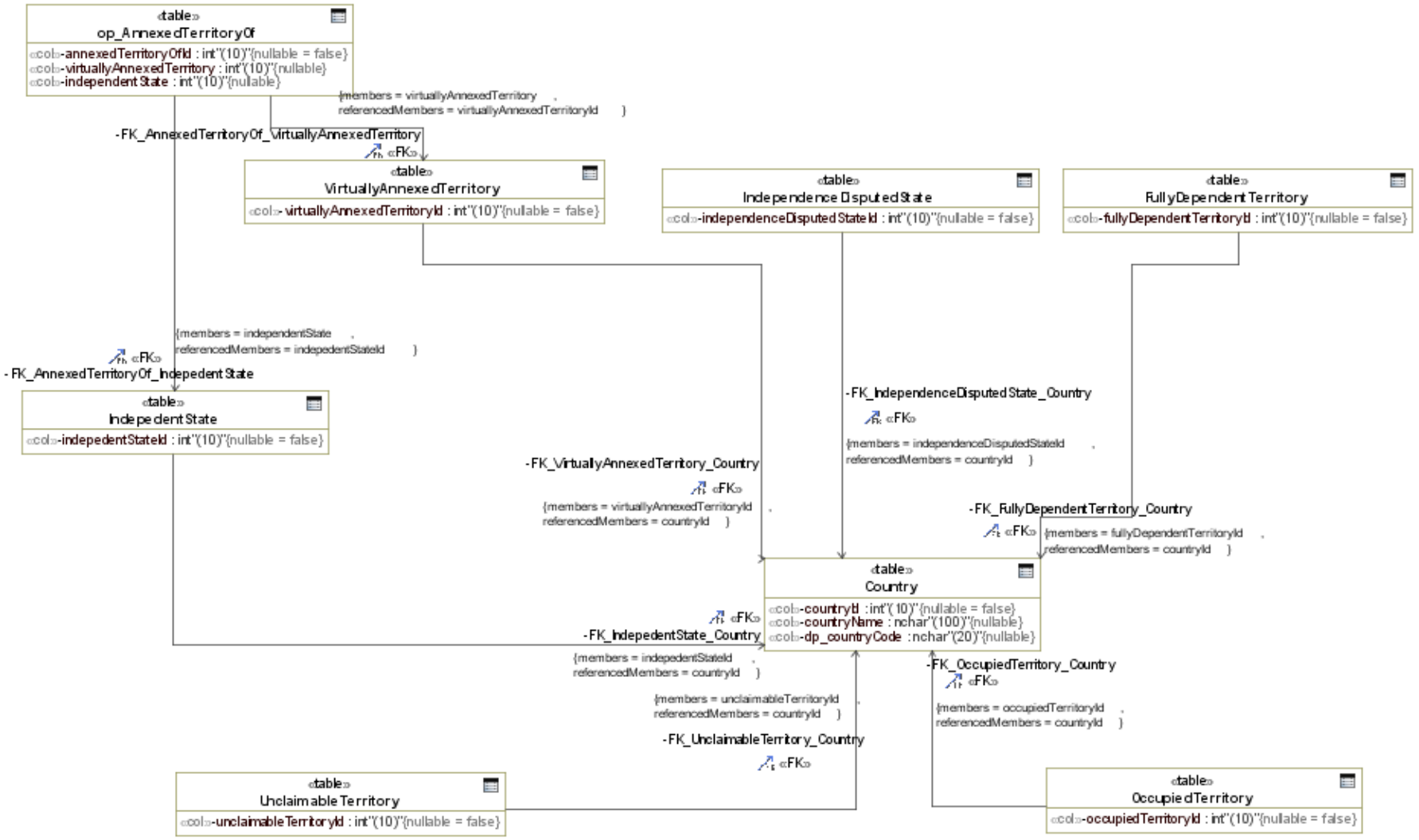
Jeigu ryšio kardinalumas yra 1:*, kuriama objekto savybė, kurios pavadinimas nustatomas pagal antrosios klasės išorinio rakto atributo pavadinimą. Pirmoji klasė nustatoma kaip objekto savybės range, antroji - domain.

Jeigu viena iš ryšio lentelių reiškia objekto savybe (tai nustatoma pagal kardinalumo metaduomenų lenteles), sukuriama objekto savybė pagal jos pavadinimą. Pagal kitą lentelę sukuriama klasė, ji nustatoma kaip objekto savybės domain arba range, priklausomai nuo išorinio rakto pozicijos. 23 pav. pateiktame pavyzdyje bus sukurta objekto savybė op_AnnexedTerritoryOf, jos domain bus klasė IndependetState. Klasė VirtuallyAnnexedTerritory nustatoma objekto savybės op_AnnexedTerritoryOf range klase.



23 pav. Objektų tipo ryšys tarp duomenų lentelių

Ontologijos užklausoms vykdyti naudojama užklausų SQL kalba. Užklausos vykdomos tiek klasių lygmenyje, tiek rasti klasių individams. Klasių kūrimui naudojamas *JenaModel*, kuriame sukuriamas ontologijos modelis pagal reliacinės duomenų bazės schemą. Tačiau šiame modelyje nesaugoma informacija apie individus, jie gaunami iš duomenų bazės lentelių.

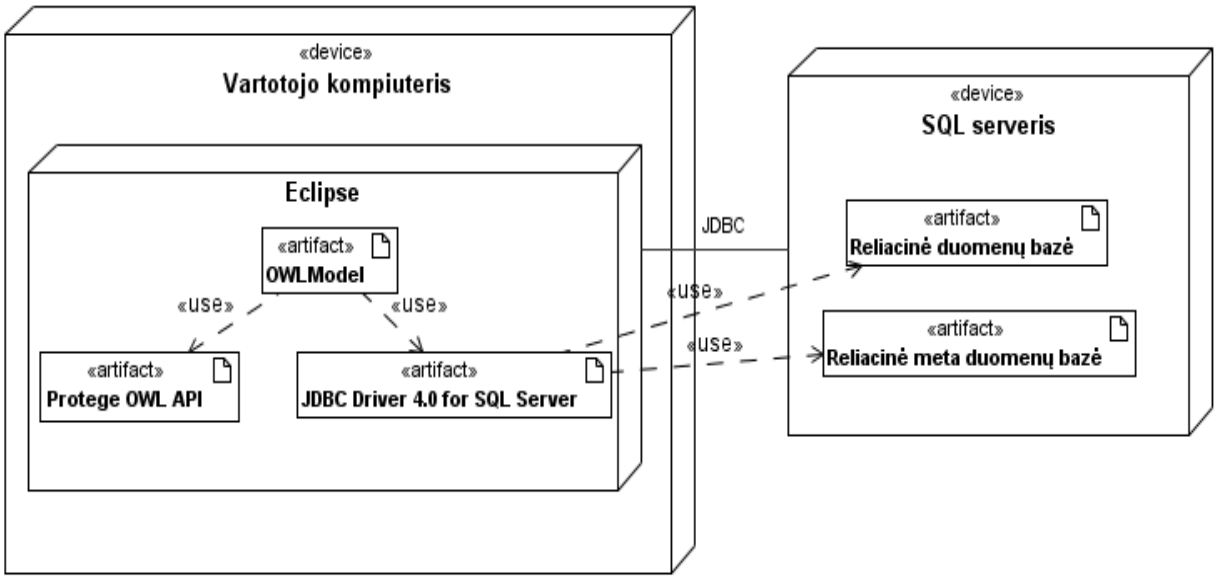


24pav. Duomenų bazės schema

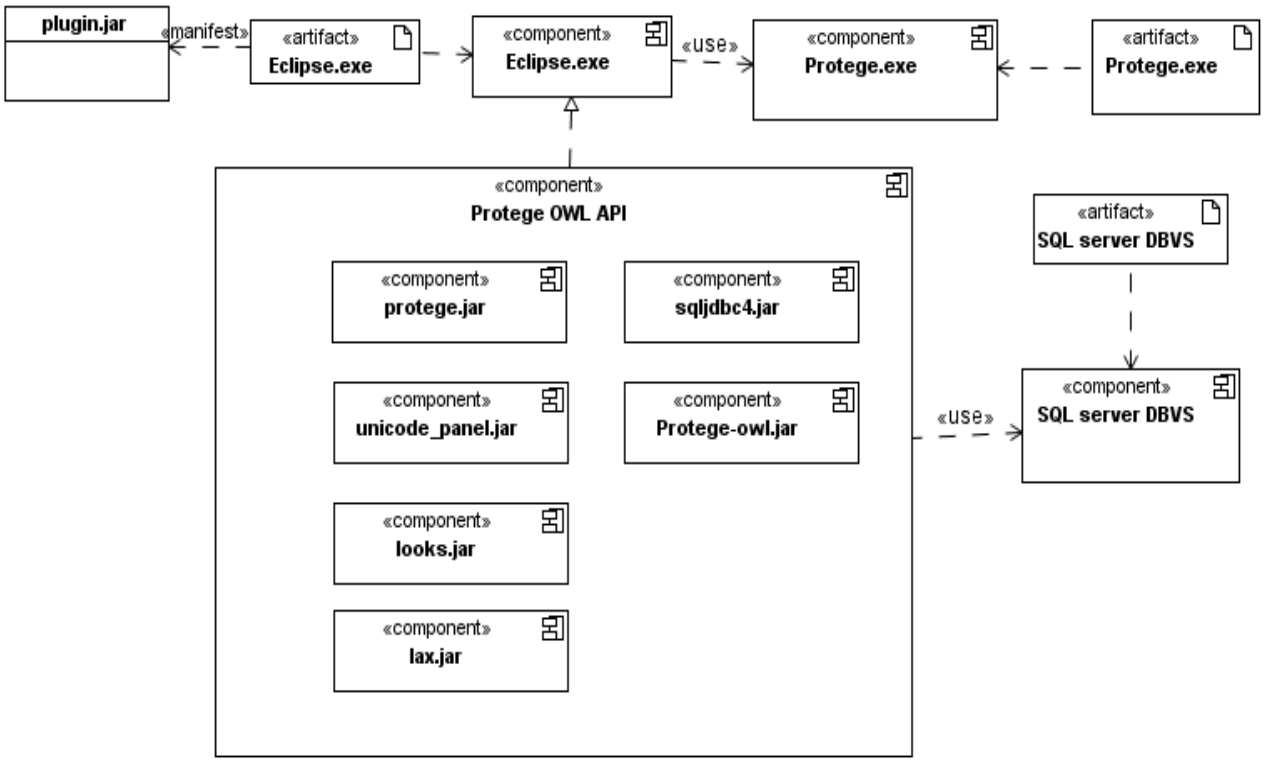
5. Ontologijos atkūrimo iš RDB algoritmo prototipo realizacija

Užklausų vykdymo sistemai kurti buvo pritaikyta *Protege 3.5 OWL* biblioteka programiniam paketui *Eclipse.* Ontologijos kūrimui jame standartiškai naudojamas *java application* operavimo langas, kuris nagrinėja OWL biblioteką ir sukuria *JenaOWLmodel* modelį. Šis variantas nėra tinkamas darbo tikslui pasiekti, nes ontologija turi būti kuriama iš reliacinės duomenų bazės (ne iš URI), tačiau bibliotekoje suteikiama galimybė kurti ontologijas programiškai. Todėl bus realizuotas naujas komponentas *Protege OWL API*, kuris analizuos duomenų bazės struktūrą ir pagal ją sukurs OWL2 ontologiją. Naudojamas pagalbinis komponentas *Microsoft JDBC Driver 4.0 for SQL Server* skirtas prisijungti prie duomenų bazės ir vykdyti joje užklausas.

Sistema realizuota *Eclipse* aplinkoje, ontologijai saugoti pasirinkta *MS SQL Server DBVS*. Naudojamos dvi atskiros duomenų bazės – viena reliacinei duomenų bazei, kita reliacinės duomenų bazės metaduomenims, tam, kad korektiškai veiktų ontologijos kūrimo pagal duomenų bazės schemą algoritmas. (25 pav.) pateikta sistemos kūrimo diagrama.



25 Pav. Sistemos kūrimo diagrama



26 Pav. Naudojamų komponentų modelis

5.1. Užklausų ir komandinių eilučių testavimas

Toliau pateikta užklausa, kuri bus vykdoma testuojant. Ši užklausa reliacinėje duomenų bazėje randa lentelių pavadinimus ir ryšius tarp jų. Iteracija tęsiama tol, kol išrenkami visų lentelių pavadinimai ir ryšiai tarp lentelių, kurie išsaugojami rezultatų rinkinyje *(ResultSet*).

```
ResultSet rs = smt.getResultSet();

OWLModel owlModel = ProtegeOWL.createJenaOWLModel();
while (rs.next()) {
   String column1 = rs.getString("referenced_object_id");
   String column2 = rs.getString("name");
   String column3 = rs.getString("parent_column_id");
   String column4 = rs.getString("referenced_column_id");
```

Toliau vykdoma *java* kodo realizacija reliacinės duomenų bazės lentelių transformavimui į OWL ontologijos klases. Sukūrus klasę pagal pirmos lentelės pavadinimą ir sukūrus klasę pagal antros lentelės pavadinimą nustatoma tėvinė klasė. Prieš sukūrimą atliekamas tikrinimas ar klasė dar nesukurta. Naudojama *OwlModel* biblioteka.

```
  OWLNamedClass clas1 = owlModel.getOWLNamedClass(class1Name);
            if (clas1 == null) {
              clas1 = owlModel.createOWLNamedClass(class1Name);
            }
            OWLNamedClass clas2 = owlModel.getOWLNamedClass(column2);
            if (clas2 == null) {
              clas2 = owlModel.createOWLNamedClass(column2);
            }
            clas2.addSuperclass(clas1);
          }

        }
```

Kada yra sukurta klasė, ji iškviečiama panaudojus *owlModel* biblioteką ir komandą *getUserDefinedOWLNamedClasses.* Atliekama iteracija kol gaunami visi sukurtų klasių pavadinimai.

```
Collection classes = owlModel.getUserDefinedOWLNamedClasses();
        for (Iterator it = classes.iterator(); it.hasNext();) {
            OWLNamedClass cls = (OWLNamedClass) it.next();
            cls.getName();
```

Sukuriamas naujas *statment* objektas *(smt2)* užklausai. Tuomet vykdoma SQL užklausa iš reliacinės duomenų bazės informacinės schemos gauti lentelių stulpelių vardus. Šie įrašomi į naują rezultatų rinkinį *(rs2).* Vykdoma iteracija (**while** (rs2.next())) tol kol duomenų objekte "col" išsaugomi visų lentelių stulpelių pavadinimai. Pasinaudojus *owlModel* bibliotekos galimybėmis gaunamas individo vardas (owlModel.getOWLIndividual(col);). Tikrinama ar individas dar nesukurtas (**if** (col2 == **null**)). Jeigu indvidas dar nesukurtas, kuriamas naujas individas.

```
            int index = cls.getName().lastIndexOf("#");
            String tableName = cls.getName().substring(index+1,
cls.getName().length());
          char c[] = tableName.toCharArray();
          c[0] = Character.toLowerCase(c[0]);
          tableName = new String(c);

            Statement smt2 = con.createStatement();
```

```
          smt2.executeQuery("SELECT * FROM " + tableName);
        ResultSet rs2 = smt2.getResultSet();
        try {
          while (rs2.next()) {
            String col = rs2.getString(tableName + "Name");
            OWLIndividual col2 = owlModel.getOWLIndividual(col.trim());
            if (col2 == null) {
              cls.createInstance(col.trim());
            }
          }
```

Visa kodo realizacija:

```
package snippet;

import java.io.FileOutputStream;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.Collection;
import java.util.Iterator;

import com.hp.hpl.jena.rdf.model.Model;

import edu.stanford.smi.protege.exception.OntologyLoadException;
import edu.stanford.smi.protegex.owl.ProtegeOWL;
import edu.stanford.smi.protegex.owl.model.OWLIndividual;
import edu.stanford.smi.protegex.owl.model.OWLModel;
import edu.stanford.smi.protegex.owl.model.OWLNamedClass;
import edu.stanford.smi.protegex.owl.model.OWLObjectProperty;

public class Snippet {



  public static void main(String[] args) throws OntologyLoadException {
    //-Dprotege.dir=

    // prisijungimas prie SQL Server DB su JDBC
       String url = "jdbc:sqlserver://isd.ktu.lt:1433";
       String userName = "ZIiSI_laizolp";
       String password = "b79b6L";

    try {
       Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
       Connection con = DriverManager.getConnection (url, userName,
password);
       Statement smt = con.createStatement();
       smt.executeQuery("SELECT * from table_references");
       ResultSet rs = smt.getResultSet();


       OWLModel owlModel = ProtegeOWL.createJenaOWLModel();
       while (rs.next()) {
         String column1 = rs.getString("referenced_object_id");
```

```
          String column2 = rs.getString("name");
          String column3 = rs.getString("parent_column_id");
          String column4 = rs.getString("referenced_column_id");



          System.out.println(column1 + " | " + column2 + " | " + column3 + " |
" + column4);

          String sql = "SELECT name from sys.tables WHERE object_id='" +
column1 + "'";
          Statement smt5 = con.createStatement();
          smt5.executeQuery(sql);
          ResultSet rs5 = smt5.getResultSet();
          String class1Name = "";
          while (rs5.next()) {
            class1Name = rs5.getString("name");
          }

          if(column2.startsWith("op_")) {
            // kuriame objeko savybe ir klase bei nurodome, ar tai domain
klase, ar range klase
            OWLObjectProperty property =
owlModel.getOWLObjectProperty(column2);
            if(property == null) {
              property = owlModel.createOWLObjectProperty(column2);
            }
            OWLNamedClass clas = owlModel.getOWLNamedClass(class1Name);
            if(clas == null) {
              clas = owlModel.createOWLNamedClass(class1Name);
            }
            if(column3.equals("2")) {
              property.setDomain(clas);
            } else {
              property.setRange(clas);
            }


            if(column2.equals(2)) {
              OWLObjectProperty property1 =
owlModel.getOWLObjectProperty(column2);
              if(property1 == null) {
                property1 = owlModel.createOWLObjectProperty(column2);
              }

              OWLNamedClass clas2 = owlModel.getOWLNamedClass(class1Name);
              if(clas2 == null) {
                clas2 = owlModel.createOWLNamedClass(class1Name);
              }
              if(column3.equals("2")) {
                property.setDomain(clas2);
              } else {
                property.setRange(clas2);
              }
            }
          } else {
            // kuriame klases
```

```
            OWLNamedClass clas1 = owlModel.getOWLNamedClass(class1Name);
            if (clas1 == null) {
              clas1 = owlModel.createOWLNamedClass(class1Name);
            }


            OWLNamedClass clas2 = owlModel.getOWLNamedClass(column2);
            if (clas2 == null) {
              clas2 = owlModel.createOWLNamedClass(column2);
            }
            clas2.addSuperclass(clas1);
          }

        }


        Collection classes = owlModel.getUserDefinedOWLNamedClasses();
        for (Iterator it = classes.iterator(); it.hasNext();) {
            OWLNamedClass cls = (OWLNamedClass) it.next();

            int index = cls.getName().lastIndexOf("#");
            String tableName = cls.getName().substring(index+1,
cls.getName().length());
          char c[] = tableName.toCharArray();
          c[0] = Character.toLowerCase(c[0]);
          tableName = new String(c);

            Statement smt2 = con.createStatement();
            smt2.executeQuery("SELECT * FROM " + tableName);
          ResultSet rs2 = smt2.getResultSet();
          try {
            while (rs2.next()) {
              String col = rs2.getString(tableName + "Name");
              OWLIndividual col2 = owlModel.getOWLIndividual(col.trim());
              if (col2 == null) {
                cls.createInstance(col.trim());
              }
            }
          } catch(Exception e){e.printStackTrace();}
        }

        // gaunamas Jena OWL modelis
        Model mod = owlModel.getJenaModel();
        try  {
        // sukuriamas isvedimo failas
        FileOutputStream output = new FileOutputStream( "d:/test.owl ");
        // ontologija isvedama i faila
        mod.write(output);
        } catch(Exception e) {}

      }
      catch (ClassNotFoundException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();

      } catch (SQLException e) {
        // TODO Auto-generated catch block
```

```
            e.printStackTrace();
        }
    }
}
```

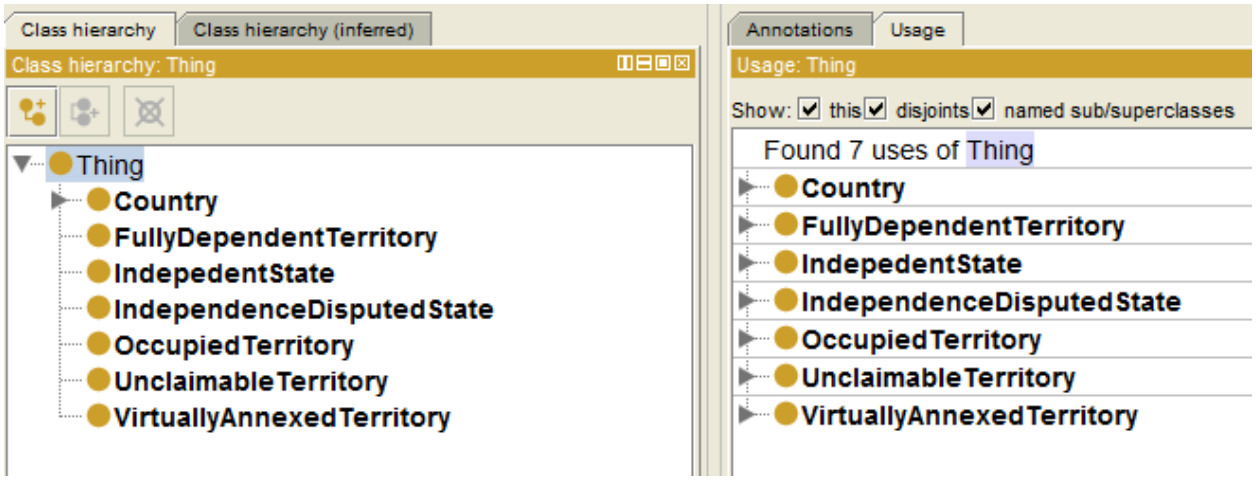
Ištestuotas transformacijos algoritmo kodas pilnai veikia. Individai sukuriami, kada randami lentelių įrašai (27 Pav.).

```
1445580188 | IndependenceDisputedState | 1 | 1
1445580188 | OccupiedTerritory | 1 | 1
1445580188 | UnclaimableTerritory | 1 | 1
1445580188 | VirtuallyAnnexedTerritory | 1 | 1
```

27 Pav. Kodo testavimas

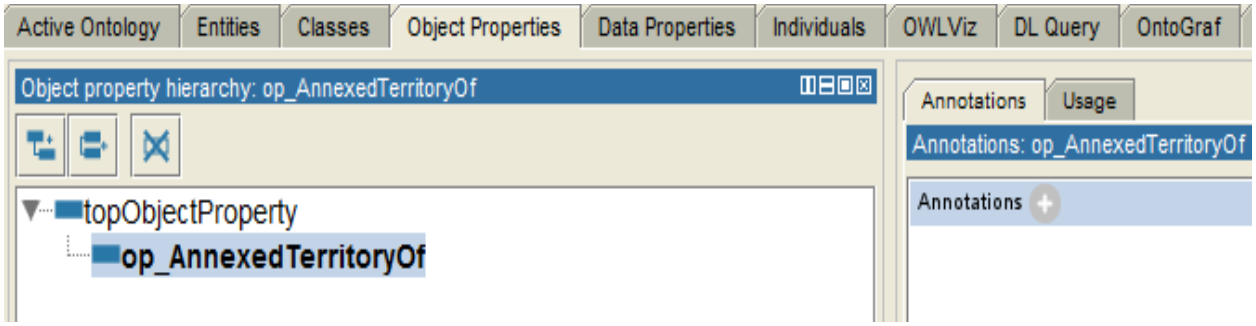
5.2.Eksperimentas

Atliekamas sukonstruotų ontologijos klasių per Protege4.1 programą tyrimas. Vaizduojamos sukonstruotos klasės iš reliacinės duomenų bazės (27 pav.).

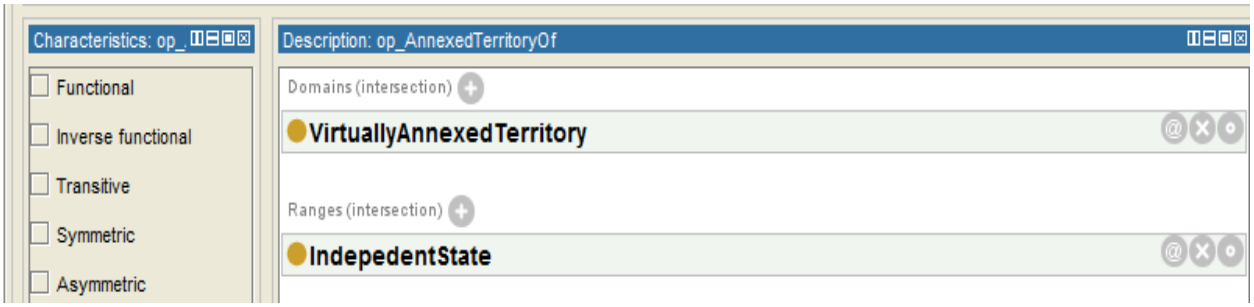


28 Pav. Sukurtos klasės

Tiriama ar sukuriamos objektų savybės:

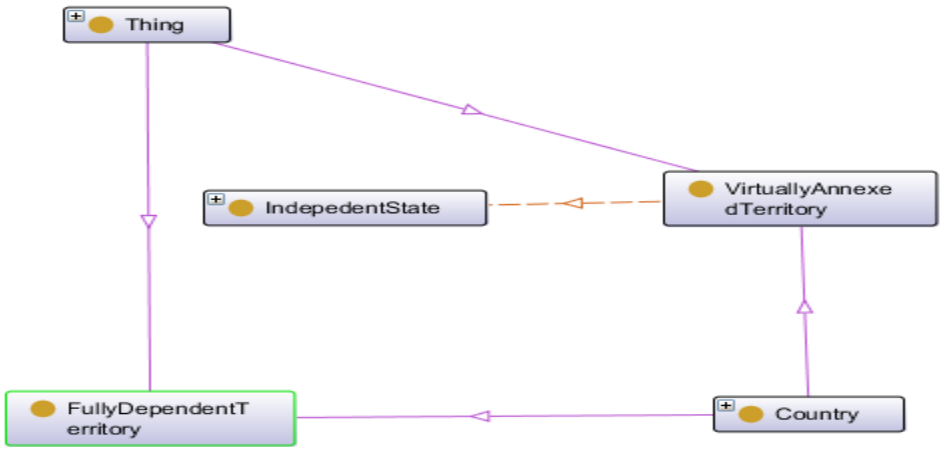


29 Pav. Sukurtos objektų savybės



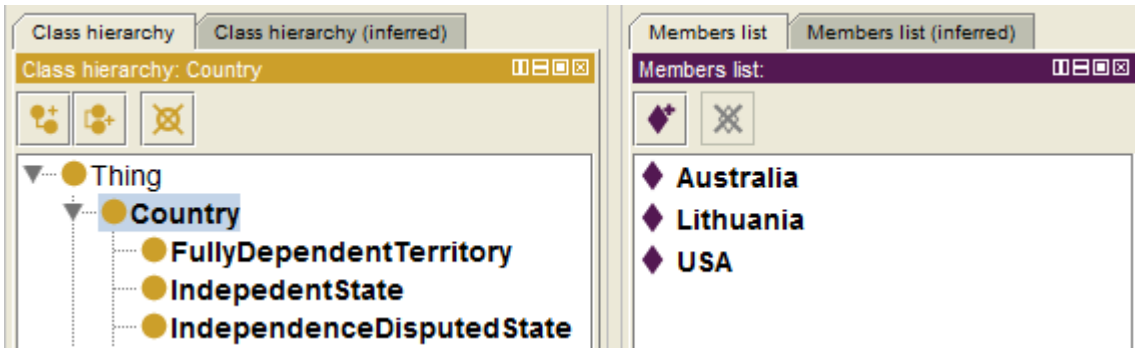
30 Pav. Nustatytos domain ir range klasės

Sukurtų klasių diagrama, kurioje atvaizduotos domain ir range klasės:



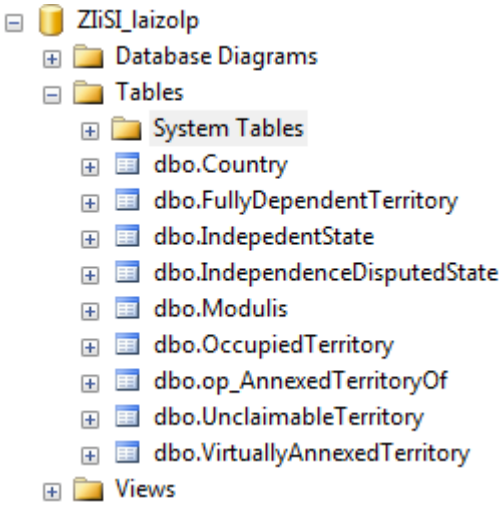
31 Pav. Nustatytos domain ir range klasės

Reliacinės duomenų bazės lentelė „dbo.Country papildyta“ trimis naujais įrašais (Australia, Lithuania, USA). Sukurtos klasės „Country individai atvaizduoti paveikslėlyje (31 pav.)



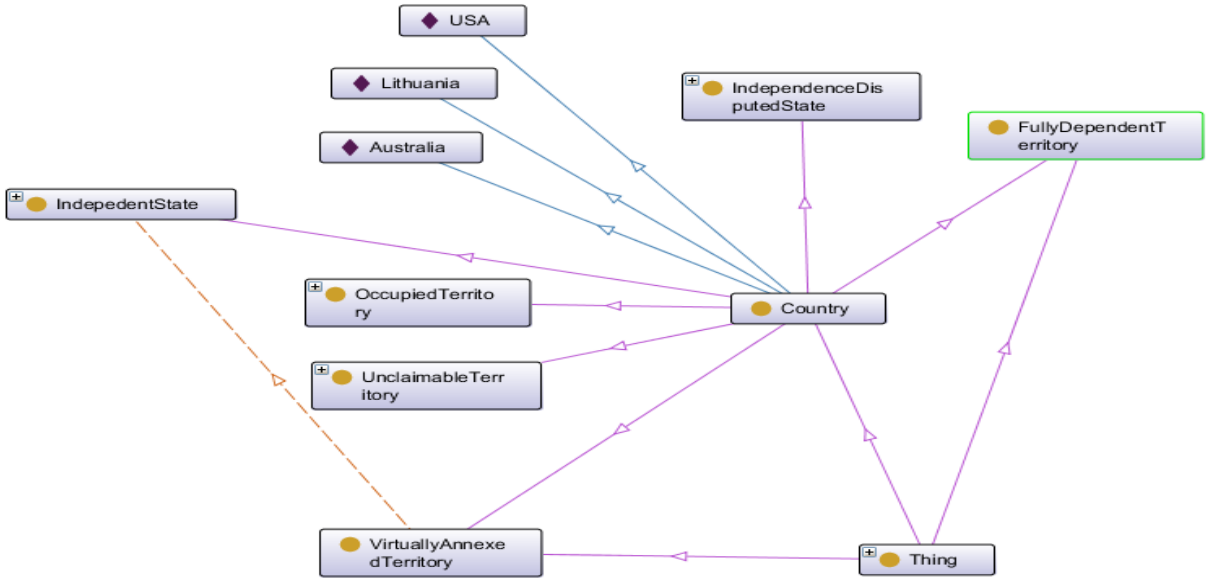
32 Pav. Sukurti individai iš RDB

Naudota reliacinė duomenų bazė yra sukonstruota iš pavyzdinės ontologijos. Užklausa konstruoti RDB pateikta priede (Priedas 1).

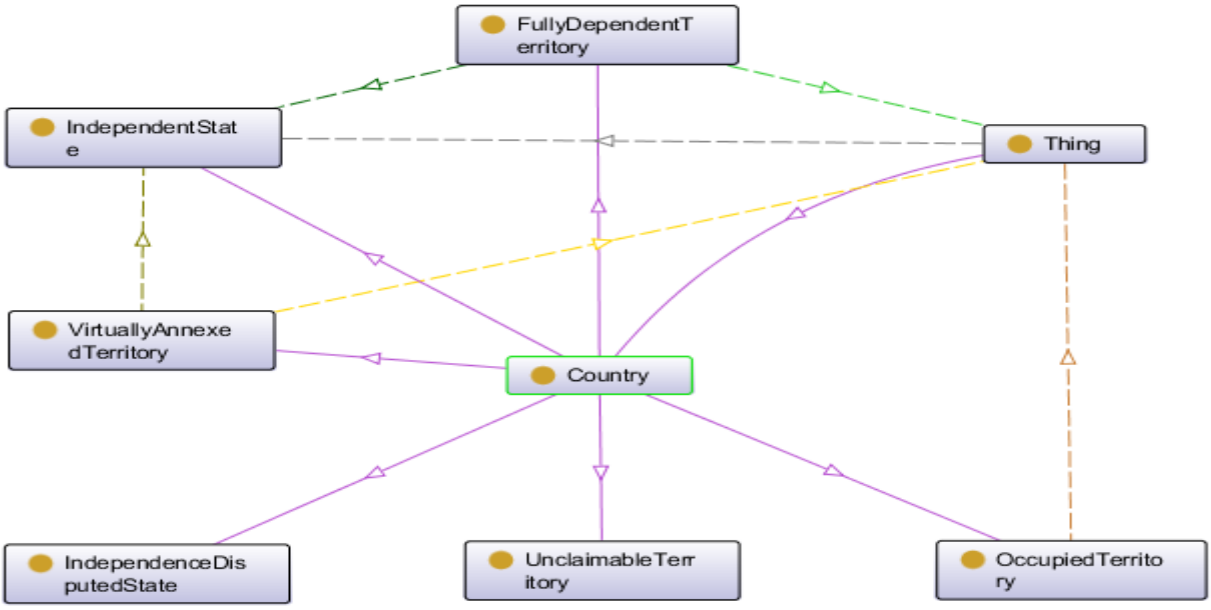


33 Pav. Duomenų bazės lentelės

Nustatyta jog ontologija sukonstruota iš reliacinės duomenų bazės (34 pav.) atitinka pavyzdinę ontologiją (35 pav). Pavyzdine ontologija pateikta priede (Priedas 2) countries.owl.



34 Pav. Sukonstruota ontologija



35 Pav. Pavyzdinės ontologijos schema

Išvados

1. Siekiant efektyviau saugoti didelės apimties ontologijas, kuriami algoritmai, leidžiantys jas išsaugoti reliacinėse duomenų bazėse. Vienas iš tokių algoritmų yra ISK sukurtas OWL to RDB. Vienas iš būdų vykdyti užklausas šiuo algoritmu išsaugotoje ontologijoje - ontologiją atkuriant iš reliacinės duomenų bazės į OWL.
2. Magistrinio darbo metu buvo suprojektuotas RDB to OWL algoritmas, kuris sukonstruoja OWL ontologiją pagal reliacinės duomenų bazės schemą ir duomenis. Algoritmas atstato klasių hierarchiją, klasių individus, objektų savybes, objektų savybėms nustato *domain* bei *range* klases.
3. Sudarytas magistro tiriamojo darbo projektas, sukurtos algoritmą aprašančios sekų diagramos, realizuotas algoritmo programinis prototipas.
4. Naudojant pavyzdinę valstybių ontologiją buvo atliktas eksperimentas. Eksperimento rezultatai parodė, kad algoritmas atkuria klases, klasių hierachiją, individus bei objektų savybes, duomenų tipų savybes.
5. Ateityje ketinama algoritmą praplėsti siekiant atkurti daugiau savybių: ryšius tarp individų, duomenų tipų savybių reikšmių nustatymą bei kitas savybes iš ontologijos metaduomenų bazės. Prototipą planuojama realizuoti kaip *Protege* įskiepį.

Literatūra

1. Barrasa,J., Gómez-Pérez, A.,“Upgrading relational legacy data to the semantic web“. In Proc. of 15th international conference on World Wide Web Conference (WWW 2006), psl.1069-1070, Edinburgh, Anglija, 23-26 gegužis 2006.
2. Cerbah, “Learning Highly Structured Semantic Repositories from Relational Databases - The RDBToOnto Tool.”, Proceedings of the 5th European Semantic Web Conference (ESWC 2008), Tenerife, Ispanija, birželis, 2008
3. Cullot, N., Ghawi, R., Yetongnon, K, “DB2OWL: A Tool for Automatic Database-to-Ontology Mapping”. In Proc. of 15th Italian Symposium On Advanced Database Systems (SEBD 2007),psl. 491-494, Torre Canne, Italija, 17-20 birželis 2007.
4. Green, J., Dolbear, C., Hart, G., Engelbrecht, P., Goodwin, "Creating a semantic integration system using spatial data", J., in International Semantic Web Conference 2008 Karlsruhe, Vokietija.
5. Guntars Būmans. RDB2OWL: a Practical Approach for Transforming RDB Data into RDF/OWL. Latvija. Latvijos universitetas.
6. J. Bock, S. Grimm, J. Henß, J. Kleb. A Database Backend for OWL. In: Proceedings of the 5th International Workshop on OWL: Experiences and Directions OWLED 2009, Chantilly, VA, United States, October 23-24, 2009 , Vol. 529, 2009, 1–8.
7. Jorg Henb, Joachim Kleb, Stephan Grimm. A Protege 4 Backend for Native OWL Persistence. Karslsruhe, Vokietija.Fraunhofer HTB.
8. Kashyap, V., Flanagan, M.,“From Web 1.0 -> 3.0: Is RDF access to RDB enough?”., Position paper for the W3C Workshop on RDF Access to Relational Databases, Cambridge, Jungtinės Amerikos valstijos, 25-26 spalis 2007.
9. Natalya F. Noy ir Deborah L. McGuinness. Ontology Development 101. A Guide For Creating Your First Ontology. Stanford University, Stanford, CA, 94305
10. O. Lassila, R. R. Swick. Resource Description Framework (RDF) Model and Syntax Specification. [žiūrėta 2013 m. sausio 12 d.]. Prieiga per internetą:< http://www.w3.org/TR/PR-rdf-syntax/ > .
11. „Protege Wiki“. ProtegeOWL_API kūrimas [interaktyvus] Stanford 2011 [žiūrėta 2013 m. balandžio 20 d.]. Prieiga per internetą: <http://protegewiki.stanford.edu/wiki/ProtegeOWL_API_Advanced_Topics>.

12. Šukys A., „Semantinių užklausų vykdymas saugant ontologiją reliacinėje duomenų bazėje“. Lietuva. Kauno technologijų universitetas, informacinių sistemų katedra.
13. Vysniauskas E., Nemuraitė L., „Transforming ontology representation from OWL to relational database“. Kaunas. Kauno technologijų universitetas, informacinių sistemų katedra, psl. 333-341, 2006.
14. „W3C Incubator Group“. Tyrimas esamų susiejimų tarp reliacinių duomenų bazių ir RDF [interaktyvus]. Leipzig, 2008 [žiūrėta 2012 m. gruodžio 20 d.]. Prieiga per internetą: <http://www.w3.org/2005/Incubator/rdb2rdf/RDB2RDF_SurveyReport_01082009.pdf>.
15. „W3C Incubator Group“. OWL 2 Web Ontology Language Structural Specification and Functional-Style Syntax http://www.w3.org/TR/2009/REC-owl2-syntax-20091027/
16. Wikipedia. Rational database. [žiūrėta 2011 m. sausio 20 d.]. Prieiga per internetą: < http://en.wikipedia.org/wiki/Relational_database >

1 PRIEDAS

UŽKLAUSOS KODAS KONSTRUOTI RDB

```
USE [Testas2]
GO
/****** Object:  Table [dbo].[Country]    Script Date: 08/30/2010
21:06:40 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Country](
  [countryId] [int] NOT NULL,
  [countryName] [nchar](100) COLLATE Lithuanian_CI_AS NULL,
  [dp_countryCode] [nchar](20) COLLATE Lithuanian_CI_AS NULL,
 CONSTRAINT [PK_Country] PRIMARY KEY CLUSTERED
(
  [countryId] ASC
)WITH (IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]

USE [Testas2]
GO
/****** Object:  Table [dbo].[FullyDependentTerritory]    Script Date:
08/30/2010 21:07:10 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[FullyDependentTerritory](
  [fullyDependentTerritoryId] [int] NOT NULL,
 CONSTRAINT [PK_FullyDependentTerritory] PRIMARY KEY CLUSTERED
(
  [fullyDependentTerritoryId] ASC
)WITH (IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]

GO
USE [Testas2]
GO
ALTER TABLE [dbo].[FullyDependentTerritory]  WITH CHECK ADD
CONSTRAINT [FK_FullyDependentTerritory_Country] FOREIGN
KEY([fullyDependentTerritoryId])
REFERENCES [dbo].[Country] ([countryId])

USE [Testas2]
GO
/****** Object:  Table [dbo].[IndepedentState]    Script Date:
08/30/2010 21:07:25 ******/
```

```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[IndepedentState](
  [indepedentStateId] [int] NOT NULL,
 CONSTRAINT [PK_IndepedentState] PRIMARY KEY CLUSTERED
(
  [indepedentStateId] ASC
)WITH (IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]

GO
USE [Testas2]
GO
ALTER TABLE [dbo].[IndepedentState]  WITH CHECK ADD  CONSTRAINT
[FK_IndepedentState_Country] FOREIGN KEY([indepedentStateId])
REFERENCES [dbo].[Country] ([countryId])

USE [Testas2]
GO
/****** Object:  Table [dbo].[IndependenceDisputedState]    Script
Date: 08/30/2010 21:07:40 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[IndependenceDisputedState](
  [independenceDisputedStateId] [int] NOT NULL,
 CONSTRAINT [PK_IndependenceDisputedState] PRIMARY KEY CLUSTERED
(
  [independenceDisputedStateId] ASC
)WITH (IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]

GO
USE [Testas2]
GO
ALTER TABLE [dbo].[IndependenceDisputedState]  WITH CHECK ADD
CONSTRAINT [FK_IndependenceDisputedState_Country] FOREIGN
KEY([independenceDisputedStateId])
REFERENCES [dbo].[Country] ([countryId])

USE [Testas2]
GO
/****** Object:  Table [dbo].[OccupiedTerritory]    Script Date:
08/30/2010 21:07:50 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[OccupiedTerritory](
```

```
  [occupiedTerritoryId] [int] NOT NULL,
 CONSTRAINT [PK_OccupiedTerritory] PRIMARY KEY CLUSTERED
(
  [occupiedTerritoryId] ASC
)WITH (IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]

GO
USE [Testas2]
GO
ALTER TABLE [dbo].[OccupiedTerritory]  WITH CHECK ADD  CONSTRAINT
[FK_OccupiedTerritory_Country] FOREIGN KEY([occupiedTerritoryId])
REFERENCES [dbo].[Country] ([countryId])

USE [Testas2]
GO
/****** Object:  Table [dbo].[UnclaimableTerritory]    Script Date:
08/30/2010 21:08:00 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[UnclaimableTerritory](
  [unclaimableTerritoryId] [int] NOT NULL,
 CONSTRAINT [PK_UnclaimableTerritory] PRIMARY KEY CLUSTERED
(
  [unclaimableTerritoryId] ASC
)WITH (IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]

GO
USE [Testas2]
GO
ALTER TABLE [dbo].[UnclaimableTerritory]  WITH CHECK ADD  CONSTRAINT
[FK_UnclaimableTerritory_Country] FOREIGN
KEY([unclaimableTerritoryId])
REFERENCES [dbo].[Country] ([countryId])

USE [Testas2]
GO
/****** Object:  Table [dbo].[VirtuallyAnnexedTerritory]    Script
Date: 08/30/2010 21:08:11 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[VirtuallyAnnexedTerritory](
  [virtuallyAnnexedTerritoryId] [int] NOT NULL,
 CONSTRAINT [PK_VirtuallyAnnexedTerritory] PRIMARY KEY CLUSTERED
(
  [virtuallyAnnexedTerritoryId] ASC
)WITH (IGNORE_DUP_KEY = OFF) ON [PRIMARY]
```

```
) ON [PRIMARY]

GO
USE [Testas2]
GO
ALTER TABLE [dbo].[VirtuallyAnnexedTerritory]  WITH CHECK ADD
CONSTRAINT [FK_VirtuallyAnnexedTerritory_Country] FOREIGN
KEY([virtuallyAnnexedTerritoryId])
REFERENCES [dbo].[Country] ([countryId])

USE [Testas2]
GO
/****** Object:  Table [dbo].[op_AnnexedTerritoryOf]    Script Date:
08/30/2010 21:08:20 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[op_AnnexedTerritoryOf](
  [annexedTerritoryOfId] [int] NOT NULL,
  [virtuallyAnnexedTerritory] [int] NULL,
  [independentState] [int] NULL,
 CONSTRAINT [PK_AnnexedTerritoryOf] PRIMARY KEY CLUSTERED
(
  [annexedTerritoryOfId] ASC
)WITH (IGNORE_DUP_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]

GO
USE [Testas2]
GO
ALTER TABLE [dbo].[op_AnnexedTerritoryOf]  WITH CHECK ADD  CONSTRAINT
[FK_AnnexedTerritoryOf_IndepedentState] FOREIGN
KEY([independentState])
REFERENCES [dbo].[IndepedentState] ([indepedentStateId])
GO
ALTER TABLE [dbo].[op_AnnexedTerritoryOf]  WITH CHECK ADD  CONSTRAINT
[FK_AnnexedTerritoryOf_VirtuallyAnnexedTerritory] FOREIGN
KEY([virtuallyAnnexedTerritory])
REFERENCES [dbo].[VirtuallyAnnexedTerritory]
([virtuallyAnnexedTerritoryId])

USE [Testas2];

INSERT INTO Country
           (countryId
           ,countryName
           ,dp_countryCode)
     VALUES(1, 'USA', '1234')

INSERT INTO Country
           (countryId
```

```
            ,countryName
            ,dp_countryCode)
      VALUES(2, 'Lithuania', '5678')

INSERT INTO Country
            (countryId
            ,countryName
            ,dp_countryCode)
      VALUES(3, 'Australia', '9132')


INSERT INTO OccupiedTerritory
            (occupiedTerritoryId)
      VALUES
            (3)


INSERT INTO IndependenceDisputedState
            (independenceDisputedStateId)
      VALUES
            (1)

INSERT INTO IndependenceDisputedState
            (independenceDisputedStateId)
      VALUES
            (2)
```

PAVYZDINĖ ONTOLOGIJA COUNTRIES.OWL

```
<?xml version="1.0"?>
<rdf:RDF

xmlns:Countries="http://www.bpiresearch.com/BPMO/2004/03/03/cdl/Countr
ies#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:protege="http://protege.stanford.edu/plugins/owl/protege#"
    xmlns:xsp="http://www.owl-ontologies.com/2005/08/07/xsp.owl#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:owl2xml="http://www.w3.org/2006/12/owl2-xml#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:swrl="http://www.w3.org/2003/11/swrl#"
    xmlns:swrlb="http://www.w3.org/2003/11/swrlb#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xml:base="http://www.bpiresearch.com/BPMO/2004/03/03/cdl/Countries">
  <owl:Ontology rdf:about="">
    <rdfs:comment
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Countries Ontology

Created by Jenz &amp;amp; Partner GmbH

Version 0.1, 28-FEB-2004</rdfs:comment>
    <owl:versionInfo
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Version 0.1</owl:versionInfo>
  </owl:Ontology>
  <owl:Class rdf:ID="ISO3166DefinedCountry">
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:onProperty>
          <owl:DatatypeProperty rdf:ID="countryCodeISO3166Alpha2"/>
        </owl:onProperty>
        <owl:cardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger"
        >1</owl:cardinality>
      </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:cardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger"
        >1</owl:cardinality>
        <owl:onProperty>
          <owl:DatatypeProperty rdf:ID="countryNameISO3166Short"/>
        </owl:onProperty>
```

```
      </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:onProperty>
          <owl:ObjectProperty rdf:ID="referencesCountry"/>
        </owl:onProperty>
        <owl:cardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger"
        >1</owl:cardinality>
      </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:cardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger"
        >1</owl:cardinality>
        <owl:onProperty>
          <owl:DatatypeProperty rdf:ID="countryCodeUNNumeric3"/>
        </owl:onProperty>
      </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:cardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger"
        >1</owl:cardinality>
        <owl:onProperty>
          <owl:DatatypeProperty rdf:ID="countryCodeISO3166Alpha3"/>
        </owl:onProperty>
      </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:comment
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >The International Standards Organization (ISO) maintains a list
of country codes since 1974. Other standards bodies have adopted the
ISO 3166 code list.</rdfs:comment>
  </owl:Class>
  <owl:Class rdf:ID="VirtualTerritory">
    <rdfs:comment
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >A virtual territory does not exist in reality.

Example: France, Metropolitan, is limited to the European part of
France, excluding overseas territories.</rdfs:comment>
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Country"/>
    </rdfs:subClassOf>
    <owl:disjointWith>
      <owl:Class rdf:ID="VirtuallyAnnexedTerritory"/>
    </owl:disjointWith>
  </owl:Class>
```

```
  <owl:Class rdf:ID="IndependenceDisputedState">
    <rdfs:subClassOf>
      <owl:Class rdf:about="#Country"/>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:minCardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger"
        >1</owl:minCardinality>
        <owl:onProperty>
          <owl:ObjectProperty rdf:ID="sovereigntyChallengedBy"/>
        </owl:onProperty>
      </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:comment
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >A disputed state is regarded as an independent state by its own
population (or at least the majority of them) and has been recognized
as independent by some international body (e.g. the U.N.) at some
point in the past. However, a disputed territory's status of
independence is questioned by some other independent
country/countries.

Example: The Republic of China (Taiwan) and the People's Republic of
China hold differing views on the sovereignty of
Taiwan.</rdfs:comment>
    <owl:disjointWith>
      <owl:Class rdf:ID="OccupiedTerritory"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:ID="UnclaimableTerritory"/>
    </owl:disjointWith>
    <owl:disjointWith rdf:resource="#VirtualTerritory"/>
    <owl:disjointWith>
      <owl:Class rdf:about="#VirtuallyAnnexedTerritory"/>
    </owl:disjointWith>
  </owl:Class>
  <owl:Class rdf:about="http://www.w3.org/2002/07/owl#Thing"/>
  <owl:Class rdf:about="#VirtuallyAnnexedTerritory">
    <rdfs:comment
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >No government body exists, or, if one exists, it is not given
recognition by the annexing country. The territory has been
unilaterally annexed by some independent country, usually as a result
of occupation. The annexing country imposes its legislation on the
annexed territory and considers it an integral part of its own
territory.

Example: The Western Sahara has been virtually annexed by
Morocco.</rdfs:comment>
    <rdfs:subClassOf>
      <owl:Class rdf:about="#Country"/>
```

```
    </rdfs:subClassOf>
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:cardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger"
        >1</owl:cardinality>
        <owl:onProperty>
          <owl:ObjectProperty rdf:ID="annexedTerritoryOf"/>
        </owl:onProperty>
      </owl:Restriction>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:about="#UnclaimableTerritory">
    <rdfs:comment
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >No government body exists. The territory is not or cannot be
claimed by an independent country.

Example: Antarctica is unclaimed resp. unclaimable territory. An
Antarctic Treaty was negotiated that neither denies nor gives
recognition to existing territorial claims.</rdfs:comment>
    <owl:disjointWith rdf:resource="#VirtualTerritory"/>
    <owl:disjointWith rdf:resource="#VirtuallyAnnexedTerritory"/>
    <rdfs:subClassOf>
      <owl:Class rdf:about="#Country"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="FullyDependentTerritory">
    <owl:disjointWith>
      <owl:Class rdf:ID="IndependentState"/>
    </owl:disjointWith>
    <owl:disjointWith rdf:resource="#VirtuallyAnnexedTerritory"/>
    <rdfs:subClassOf>
      <owl:Class rdf:about="#Country"/>
    </rdfs:subClassOf>
    <owl:disjointWith rdf:resource="#VirtualTerritory"/>
    <owl:disjointWith>
      <owl:Class rdf:about="#OccupiedTerritory"/>
    </owl:disjointWith>
    <rdfs:comment
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Self-governing to some limited degree; considered an integral
part of an independent country.

Example: Christmas Island is a territory of Australia; administered by
the Australian Department of Transport and Regional Services.
Christmas Island does not have the right to move to full independence
by unilateral action.</rdfs:comment>
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:onProperty>
          <owl:ObjectProperty rdf:ID="dependentTerritoryOf"/>
```

```
        </owl:onProperty>
        <owl:cardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger"
        >1</owl:cardinality>
      </owl:Restriction>
    </rdfs:subClassOf>
    <owl:disjointWith rdf:resource="#UnclaimableTerritory"/>
    <owl:disjointWith rdf:resource="#IndependenceDisputedState"/>
  </owl:Class>
  <owl:Class rdf:about="#IndependentState">
    <rdfs:subClassOf>
      <owl:Class rdf:about="#Country"/>
    </rdfs:subClassOf>
    <owl:disjointWith>
      <owl:Class rdf:about="#OccupiedTerritory"/>
    </owl:disjointWith>
    <owl:disjointWith rdf:resource="#UnclaimableTerritory"/>
    <owl:disjointWith rdf:resource="#VirtualTerritory"/>
    <owl:disjointWith rdf:resource="#VirtuallyAnnexedTerritory"/>
    <rdfs:comment
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Fully self-governing and sovereign. A portion of legislative
power may be transfered voluntarily to some other political body (e.g.
the European Union) at the government's discretion.

Example: The Federal Republic of Germany enjoys sovereignty and is
self-governing. It is part of the European Union. The European
Parliament is given authority to produce laws in certain fields, which
are considered binding by every member.</rdfs:comment>

  </owl:Class>
  <owl:Class rdf:about="#OccupiedTerritory">
    <rdfs:subClassOf>
      <owl:Class rdf:about="#Country"/>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:cardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger"
        >1</owl:cardinality>
        <owl:onProperty>
          <owl:ObjectProperty rdf:ID="territoryOccupiedBy"/>
        </owl:onProperty>
      </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:comment
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >An existing government body is only given limited recognition by
the occupying country. The territory has been unilaterally
(temporarily) occupied by some independent country, possibly as a
result of war. The occupying country may impose its legislation on the
occupied territory, and allow limited autonomy in the occupied
```

```
territory. The occupied territory is not an integral part of the
occupying country. Occupation may lead to formal annexation.

Example: Dispute over Palestinian territory.</rdfs:comment>
    <owl:disjointWith rdf:resource="#UnclaimableTerritory"/>
    <owl:disjointWith rdf:resource="#VirtualTerritory"/>
    <owl:disjointWith rdf:resource="#VirtuallyAnnexedTerritory"/>
  </owl:Class>
  <owl:Class rdf:about="#Country">
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:onProperty>
          <owl:DatatypeProperty rdf:ID="nameEnglishLong"/>
        </owl:onProperty>
        <owl:maxCardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger"
        >1</owl:maxCardinality>
      </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:onProperty>
          <owl:DatatypeProperty rdf:ID="nameEnglish"/>
        </owl:onProperty>
        <owl:cardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger"
        >1</owl:cardinality>
      </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:onProperty>
          <owl:DatatypeProperty rdf:ID="nameLocalLong"/>
        </owl:onProperty>
        <owl:maxCardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger"
        >1</owl:maxCardinality>
      </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:cardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger"
        >1</owl:cardinality>
        <owl:onProperty>
          <owl:DatatypeProperty rdf:ID="nameLocal"/>
        </owl:onProperty>
      </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:comment
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
```

```
    >A political state or nation or its territory (see Merriam-Webster
Dictionary).

The presentation of the material in this electronic publication does
not imply the expression of any opinion whatsoever on the part of Jenz
&amp;amp; Partner concerning the legal status of any country, territory,
city or area, or of its authorities, or concerning the delimitations
of its frontiers or boundaries.</rdfs:comment>
  </owl:Class>
  <owl:Class rdf:ID="AssociatedState">
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:onProperty>
          <owl:ObjectProperty rdf:ID="associatedTerritoryOf"/>
        </owl:onProperty>
        <owl:cardinality
rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger"
        >1</owl:cardinality>
      </owl:Restriction>
    </rdfs:subClassOf>
    <owl:disjointWith rdf:resource="#IndependentState"/>
    <owl:disjointWith rdf:resource="#VirtuallyAnnexedTerritory"/>
    <owl:disjointWith rdf:resource="#FullyDependentTerritory"/>
    <rdfs:subClassOf rdf:resource="#Country"/>
    <owl:disjointWith rdf:resource="#OccupiedTerritory"/>
    <owl:disjointWith rdf:resource="#IndependenceDisputedState"/>
    <owl:disjointWith rdf:resource="#UnclaimableTerritory"/>
    <rdfs:comment
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Self-governing in free association with some independent country.
An assoiated state is neither occupied nor annexed, nor is its
territory considered an integral part of the country it is associated
with.

Example: Cook Islands is fully responsible for internal affairs; New
Zealand retains responsibility for external affairs and defense, in
consultation with the Cook Islands. Cook Islands has the right at any
time to move to full independence by unilateral action.</rdfs:comment>
    <owl:disjointWith rdf:resource="#VirtualTerritory"/>
  </owl:Class>
  <owl:ObjectProperty rdf:about="#territoryOccupiedBy">
    <rdfs:range>
      <owl:Class>
        <owl:unionOf rdf:parseType="Collection">
          <owl:Class rdf:about="#IndependenceDisputedState"/>
          <owl:Class rdf:about="#IndependentState"/>
        </owl:unionOf>
      </owl:Class>
    </rdfs:range>
    <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
    <rdfs:domain rdf:resource="#OccupiedTerritory"/>
```

```
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:about="#referencesCountry">
    <rdfs:range>
      <owl:Class>
        <owl:unionOf rdf:parseType="Collection">
          <owl:Class rdf:about="#AssociatedState"/>
          <owl:Class rdf:about="#FullyDependentTerritory"/>
          <owl:Class rdf:about="#IndependenceDisputedState"/>
          <owl:Class rdf:about="#IndependentState"/>
          <owl:Class rdf:about="#OccupiedTerritory"/>
          <owl:Class rdf:about="#UnclaimableTerritory"/>
          <owl:Class rdf:about="#VirtualTerritory"/>
          <owl:Class rdf:about="#VirtuallyAnnexedTerritory"/>
        </owl:unionOf>
      </owl:Class>
    </rdfs:range>
    <rdfs:domain rdf:resource="#ISO3166DefinedCountry"/>
    <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="referencedByISO3166">
    <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
    <rdfs:domain rdf:resource="#Country"/>
    <rdfs:range rdf:resource="#ISO3166DefinedCountry"/>
    <owl:inverseOf rdf:resource="#referencesCountry"/>
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:about="#dependentTerritoryOf">
    <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
    <rdfs:range rdf:resource="#IndependentState"/>
    <rdfs:domain rdf:resource="#FullyDependentTerritory"/>
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="hasAnnexedTerritory">
    <rdfs:domain>
      <owl:Class>
        <owl:unionOf rdf:parseType="Collection">
          <owl:Class rdf:about="#IndependenceDisputedState"/>
          <owl:Class rdf:about="#IndependentState"/>
        </owl:unionOf>
      </owl:Class>
    </rdfs:domain>
    <rdfs:range rdf:resource="#VirtuallyAnnexedTerritory"/>
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:about="#annexedTerritoryOf">
    <rdfs:domain rdf:resource="#VirtuallyAnnexedTerritory"/>
    <owl:inverseOf rdf:resource="#hasAnnexedTerritory"/>
    <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
    <rdfs:range rdf:resource="#IndependentState"/>
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="hasAssociatedTerritory">
```

```
    <owl:inverseOf>
      <owl:ObjectProperty rdf:about="#associatedTerritoryOf"/>
    </owl:inverseOf>
    <rdfs:domain>
      <owl:Class>
        <owl:unionOf rdf:parseType="Collection">
          <owl:Class rdf:about="#IndependenceDisputedState"/>
          <owl:Class rdf:about="#IndependentState"/>
        </owl:unionOf>
      </owl:Class>
    </rdfs:domain>
    <rdfs:range rdf:resource="#AssociatedState"/>
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:about="#associatedTerritoryOf">
    <rdfs:domain rdf:resource="#AssociatedState"/>
    <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
    <rdfs:range rdf:resource="#IndependentState"/>
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="challengesSovereigntyOf">
    <rdfs:domain rdf:resource="#IndependentState"/>
    <rdfs:range rdf:resource="#IndependenceDisputedState"/>
    <owl:inverseOf>
      <owl:ObjectProperty rdf:about="#sovereigntyChallengedBy"/>
    </owl:inverseOf>
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="hasOccupiedTerritory">
    <rdfs:range rdf:resource="#OccupiedTerritory"/>
    <owl:inverseOf rdf:resource="#territoryOccupiedBy"/>
    <rdfs:domain>
      <owl:Class>
        <owl:unionOf rdf:parseType="Collection">
          <owl:Class rdf:about="#IndependenceDisputedState"/>
          <owl:Class rdf:about="#IndependentState"/>
        </owl:unionOf>
      </owl:Class>
    </rdfs:domain>
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="hasDependentTerritory">
    <rdfs:domain>
      <owl:Class>
        <owl:unionOf rdf:parseType="Collection">
          <owl:Class rdf:about="#IndependenceDisputedState"/>
          <owl:Class rdf:about="#IndependentState"/>
        </owl:unionOf>
      </owl:Class>
    </rdfs:domain>
    <rdfs:range rdf:resource="#FullyDependentTerritory"/>
    <owl:inverseOf rdf:resource="#dependentTerritoryOf"/>
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:about="#sovereigntyChallengedBy">
    <rdfs:domain rdf:resource="#IndependenceDisputedState"/>
```

```
    <rdfs:range rdf:resource="#IndependentState"/>
  </owl:ObjectProperty>
  <owl:DatatypeProperty rdf:ID="effectiveFrom">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#date"/>
    <rdfs:domain>
      <owl:Class>
        <owl:unionOf rdf:parseType="Collection">
          <owl:Class rdf:about="#AssociatedState"/>
          <owl:Class rdf:about="#FullyDependentTerritory"/>
          <owl:Class rdf:about="#ISO3166DefinedCountry"/>
          <owl:Class rdf:about="#IndependenceDisputedState"/>
          <owl:Class rdf:about="#IndependentState"/>
          <owl:Class rdf:about="#OccupiedTerritory"/>
          <owl:Class rdf:about="#UnclaimableTerritory"/>
          <owl:Class rdf:about="#VirtualTerritory"/>
          <owl:Class rdf:about="#VirtuallyAnnexedTerritory"/>
        </owl:unionOf>
      </owl:Class>
    </rdfs:domain>
    <rdfs:comment
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >The date when an entry becomes effective.</rdfs:comment>
    <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="countryNameISO3166OfficialName">
    <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
    <rdfs:range
rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    <rdfs:domain rdf:resource="#ISO3166DefinedCountry"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:about="#countryNameISO3166Short">
    <rdfs:domain rdf:resource="#ISO3166DefinedCountry"/>
    <rdfs:range
rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:about="#nameEnglishLong">
    <rdfs:range
rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    <rdfs:domain rdf:resource="#Country"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="revocationEffective">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#date"/>
    <rdfs:comment
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >The date when a revocation of an entry becomes
effective.</rdfs:comment>
    <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
```

```
    <rdfs:domain>
      <owl:Class>
        <owl:unionOf rdf:parseType="Collection">
          <owl:Class rdf:about="#AssociatedState"/>
          <owl:Class rdf:about="#FullyDependentTerritory"/>
          <owl:Class rdf:about="#ISO3166DefinedCountry"/>
          <owl:Class rdf:about="#IndependenceDisputedState"/>
          <owl:Class rdf:about="#IndependentState"/>
          <owl:Class rdf:about="#OccupiedTerritory"/>
          <owl:Class rdf:about="#UnclaimableTerritory"/>
          <owl:Class rdf:about="#VirtualTerritory"/>
          <owl:Class rdf:about="#VirtuallyAnnexedTerritory"/>
        </owl:unionOf>
      </owl:Class>
    </rdfs:domain>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:about="#countryCodeISO3166Alpha2">
    <rdfs:comment
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >ISO 3166 Alpha 2</rdfs:comment>
    <rdfs:domain rdf:resource="#ISO3166DefinedCountry"/>
    <rdfs:range
rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:about="#countryCodeUNNumeric3">
    <rdfs:domain rdf:resource="#ISO3166DefinedCountry"/>
    <rdfs:comment
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >UN Numeric Code</rdfs:comment>
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#int"/>
    <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:about="#nameLocalLong">
    <rdfs:domain rdf:resource="#Country"/>
    <rdfs:range
rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:about="#nameEnglish">
    <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
    <rdfs:comment
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >The name of the country in the English language</rdfs:comment>
    <rdfs:domain rdf:resource="#Country"/>
    <rdfs:range
rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:about="#countryCodeISO3166Alpha3">
```

```
    <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
    <rdfs:comment
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >ISO 3166 Alpha 3</rdfs:comment>
    <rdfs:domain rdf:resource="#ISO3166DefinedCountry"/>
    <rdfs:range
rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="modificationTimestamp">
    <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
    <rdfs:range
rdf:resource="http://www.w3.org/2001/XMLSchema#dateTime"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="creationTimestamp">
    <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
    <rdfs:range
rdf:resource="http://www.w3.org/2001/XMLSchema#dateTime"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:about="#nameLocal">
    <rdfs:comment
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >The name of the country in local language, using ISO-8859
characters.</rdfs:comment>
    <rdfs:range
rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    <rdfs:domain rdf:resource="#Country"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="deletionTimestamp">
    <rdfs:range
rdf:resource="http://www.w3.org/2001/XMLSchema#dateTime"/>
    <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
  </owl:DatatypeProperty>
</rdf:RDF>

<!-- Created with Protege (with OWL Plugin 3.4.4, Build 579)
http://protege.stanford.edu -->
```

Sukonstruota ontologija sukonstruota.owl

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:protege="http://protege.stanford.edu/plugins/owl/protege#"
    xmlns:xsp="http://www.owl-ontologies.com/2005/08/07/xsp.owl#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns="http://www.owl-ontologies.com/Ontology1369681744.owl#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:swrl="http://www.w3.org/2003/11/swrl#"
    xmlns:swrlb="http://www.w3.org/2003/11/swrlb#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#" >
  <rdf:Description rdf:about="http://www.owl-
ontologies.com/Ontology1369681744.owl#Studentas">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf
rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
  </rdf:Description>
  <rdf:Description rdf:about="http://www.owl-
ontologies.com/Ontology1369681744.owl#Australia">
    <rdf:type rdf:resource="http://www.owl-
ontologies.com/Ontology1369681744.owl#Country"/>
  </rdf:Description>
  <rdf:Description rdf:about="http://www.owl-
ontologies.com/Ontology1369681744.owl#VirtuallyAnnexedTerritory">
    <rdfs:subClassOf
rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
    <rdfs:subClassOf rdf:resource="http://www.owl-
ontologies.com/Ontology1369681744.owl#Country"/>
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
  </rdf:Description>
  <rdf:Description rdf:about="http://www.owl-
ontologies.com/Ontology1369681744.owl#IndependenceDisputedState">
    <rdfs:subClassOf
rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
    <rdfs:subClassOf rdf:resource="http://www.owl-
ontologies.com/Ontology1369681744.owl#Country"/>
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
  </rdf:Description>
  <rdf:Description rdf:about="http://www.owl-
ontologies.com/Ontology1369681744.owl#op_AnnexedTerritoryOf">
    <rdfs:domain rdf:resource="http://www.owl-
ontologies.com/Ontology1369681744.owl#VirtuallyAnnexedTerritory"/>
    <rdfs:range rdf:resource="http://www.owl-
ontologies.com/Ontology1369681744.owl#IndepedentState"/>
    <rdf:type
rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
  </rdf:Description>
  <rdf:Description rdf:about="http://www.owl-
ontologies.com/Ontology1369681744.owl#IndepedentState">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf
rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
    <rdfs:subClassOf rdf:resource="http://www.owl-
ontologies.com/Ontology1369681744.owl#Country"/>
```

```
  </rdf:Description>
  <rdf:Description rdf:about="http://www.owl-
ontologies.com/Ontology1369681744.owl#FullyDependentTerritory">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf
rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
    <rdfs:subClassOf rdf:resource="http://www.owl-
ontologies.com/Ontology1369681744.owl#Country"/>
  </rdf:Description>
  <rdf:Description rdf:about="http://www.owl-
ontologies.com/Ontology1369681744.owl#StudModul">
    <rdfs:subClassOf
rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
    <rdfs:subClassOf rdf:resource="http://www.owl-
ontologies.com/Ontology1369681744.owl#Studentas"/>
    <rdfs:subClassOf rdf:resource="http://www.owl-
ontologies.com/Ontology1369681744.owl#Modulis"/>
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
  </rdf:Description>
  <rdf:Description rdf:about="http://www.owl-
ontologies.com/Ontology1369681744.owl#Lithuania">
    <rdf:type rdf:resource="http://www.owl-
ontologies.com/Ontology1369681744.owl#Country"/>
  </rdf:Description>
  <rdf:Description rdf:about="http://www.owl-
ontologies.com/Ontology1369681744.owl#Country">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf
rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
  </rdf:Description>
  <rdf:Description rdf:about="http://www.owl-
ontologies.com/Ontology1369681744.owl#OccupiedTerritory">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf
rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
    <rdfs:subClassOf rdf:resource="http://www.owl-
ontologies.com/Ontology1369681744.owl#Country"/>
  </rdf:Description>
  <rdf:Description rdf:about="http://www.owl-
ontologies.com/Ontology1369681744.owl#UnclaimableTerritory">
    <rdfs:subClassOf
rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
    <rdfs:subClassOf rdf:resource="http://www.owl-
ontologies.com/Ontology1369681744.owl#Country"/>
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
  </rdf:Description>
  <rdf:Description rdf:about="http://www.owl-
ontologies.com/Ontology1369681744.owl#USA">
    <rdf:type rdf:resource="http://www.owl-
ontologies.com/Ontology1369681744.owl#Country"/>
  </rdf:Description>
  <rdf:Description rdf:about="http://www.owl-
ontologies.com/Ontology1369681744.owl#Modulis">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
    <rdfs:subClassOf
rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
  </rdf:Description>
```

```
  <rdf:Description rdf:about="http://www.owl-
ontologies.com/Ontology1369681744.owl">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Ontology"/>
  </rdf:Description>
</rdf:RDF>
```