KAUNO TECHNOLOGIJOS UNIVERSITETAS

INFORMATIKOS FAKULTETAS

Aurelijus Saldauskas

ONTOLOGIJŲ VAIZDINIO PATEIKIMO MODELIS IR JO REALIZACIJA SEMANTINIAME TINKLE

Baigiamasis magistro projektas

Vadovas
prof. L. Nemuraitė

KAUNAS, 2015

KAUNO TECHNOLOGIJOS UNIVERSITETAS

INFORMATIKOS FAKULTETAS

ONTOLOGIJŲ VAIZDINIO PATEIKIMO MODELIS IR JO REALIZACIJA SEMANTINIAME TINKLE

Baigiamasis magistro projektas

Informacinių sistemų inžinerijos studijų programa (kodas 621E15001)

Vadovas

prof. L. Nemuraitė

2015-05-25

Recenzentas

doc. dr. Eimutis Karčiauskas

2015-05-25

Projektą atliko

Aurelijus Saldauskas

2015-05-25

KAUNAS, 2015

KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS

(Fakultetas)

(Studento vardas, pavardė)

Informacinių sistemų inžinerijos studijų programa, 621E15001

(Studijų programos pavadinimas, kodas)

Baigiamojo projekto „ONTOLOGIJŲ VAIZDINIO PATEIKIMO MODELIS IR JO REALIZACIJA SEMANTINIAME TINKLE“

AKADEMINIO SĄŽININGUMO DEKLARACIJA

20 ___ m. _____ ____ d.

Kaunas

Patvirtinu, kad mano, **Aurelijaus Saldausko**, baigiamasis projektas tema „.....“ yra parašytas visiškai savarankiškai ir visi pateikti duomenys ar tyrimų rezultatai yra teisingi ir gauti sąžiningai. Šiame darbe nei viena dalis nėra plagijuota nuo jokių spausdintinių ar internetinių šaltinių, visos kitų šaltinių tiesioginės ir netiesioginės citatos nurodytos literatūros nuorodose. Įstatymų nenumatytų piniginių sumų už šį darbą niekam nesu mokėjęs.

Aš suprantu, kad išaiškėjus nesąžiningumo faktui, man bus taikomos nuobaudos, remiantis Kauno technologijos universitete galiojančia tvarka.

_____ _____

(vardą ir pavardę įrašyti ranka) *(parašas)*

Saldauskas, A. Ontology Representation Model and its Implementation in Semantic Network. *Final Degree Project of Master of Information Systems Egineering* / Supervisor Prof. Lina Nemuraitė; Kaunas University of Technology, Faculty of Informatics.

Kaunas, 2015. 77 p.

SUMMARY

Modern semantic search engines is developed based on ontologies, which allows complex and large-scale search and information analysis. These systems users and developers not always clearly understand what they can look for or what results they can expect from semantic search engines and the best what they can do is to look in the graphical representation model of ontology.

Realized solution is ontology representation model which suits semantic analysis and search engines, also including Lithuanian systems. The prototype is realized to be universal, to suit any subject area and allows users to view ontology representation model and to have better understanding of what search can be performed. This solution stands out from other existing online solutions that this solution can process and display various subject areas of ontologies and is designed to work as a component in semantic web that can be used in other systems or software.

Keywords: Ontology, Visual representation, OWL 2, RDF, Semantic network, Jena framework.

TURINYS

LENTELIŲ SĄRAŠAS

PAVEIKSLŲ SĄRAŠAS

TERMINŲ IR SANTRUMPŲ ŽODYNAS

Santrumpa, terminas	Paaiškinimas
Ontologija	− dalykinės srities sąvokų visumos specifikacija
UML (angl. *Unified Modeling Language*)	− unifikuota modeliavimo kalba
OWL (angl. *Web Ontology Language*)	− semantinio tinklo kalba
RDF (angl. *The Resource Description Framework*)	− semantinio tinklo duomenų modelis
SPARQL (angl. *Protocol and RDF Query Language*)	− semantinio tinklo užklausų protokolas ir kalba
HTTP (angl. *Hypertext Transfer Protocol*)	− metodas pasiekti informaciją pasauliniame tinkle
PHP (angl. *Hypertext Preprocessor*)	− dinaminė interpretuojama programavimo kalba, specialiai pritaikyta interneto svetainių kūrimui
Java	− objektiškai orientuota programavimo kalba

ĮVADAS

Pastaruoju metu vis plačiau taikomos semantinės technologijos, kurios leidžia atlikti sudėtingos ir gausios informacijos analizę ir paiešką. Modernios semantinės paieškos sistemos kuriamos remiantis ontologijomis. Šių sistemų naudotojams ir kūrėjams ne visada aiškiai suprantama, ko galima ieškoti arba kokio rezultato tikėtis, kol jie neperpranta ontologijos struktūros, o tą geriausiai galima padaryti, turint vaizdų grafinį ontologijos modelį. Deja, šiuo metu esami ontologijų vaizdinio pateikimo įrankiai neatitinka naudotojų poreikių – dažniausiai jie yra nevaizdūs, parodo tik dalį reikiamų savybių, arba yra pritaikyti konkrečiai dalykinei sričiai.

Darbe sprendžiama problema - universalių įrankių, leidžiančių peržiūrėti ontologijų informaciją internete, trūkumas. Esami vaizdavimo būdai yra painūs, nepateikia visos informacijos, todėl vartotojui sunku perprasti ontologijų turinį.

Klausimai, į kuriuos turi atsakyti šis darbas:

- Ar galima sukurti universalų ontologijų vaizdinio pateikimo įrankį, kuris būtų nepriklausomas nuo dalykinės srities?
- Ar galima sukurti universalų ontologijų vaizdinio pateikimo įrankį, kuris leistų pavaizduoti visas ontologijų savybes?

Todėl šio darbo **tyrimo sritis** yra ontologijų grafinio vaizdavimo internete metodai ir technologijos, o **tyrimo objektas** yra ontologijų vaizdinio pateikimo vartotojui, siekiančiam atlikti paiešką, procesas.

Darbo tikslas - suteikti galimybes patogiai peržiūrėti išsamią ontologijos informaciją, sudarant ontologijų vaizdinio pateikimo modelį, kuris apimtų ontologijos peržiūrėjimo procesą ir pateiktų išsamią informaciją apie ontologijos turinį, bei sukuriant šį modelį realizuojančios programinės įrangos prototipą.

Darbo uždaviniai:

1. Išanalizuoti:
   1.1. Ontologijų kalbos *OWL 2* ir užklausų kalbos *SPARQL* sąvokas;
   1.2. Esamus ontologijų vaizdinio pateikimo metodus, technologijas, interneto sistemas;
2. Sudaryti ontologijų vaizdinio pateikimo modelį;
3. Suprojektuoti ontologijų vaizdinio pateikimo modelį realizuojančia informacinę sistemą;
4. Realizuoti programinį prototipą;
5. Atlikti eksperimentą modelio ir sistemos tinkamumui įvertinti.

Siekiamas sprendimas yra ontologijų vaizdavimo modelis ir jį realizuojantis prototipas, kuris tiktų semantinės analizės ir paieškos sistemoms, įskaitant ir lietuviškas sistemas. Prototipą siekiama sukurti universalų, kuris tiktų bet kokiai sričiai ir leistų vartotojams, norintiems atlikti paiešką, peržiūrėti ontologiją ir geriau suprasti, kokią paiešką galima atlikti. Šis sprendimas išsiskiria iš kitų šiuo metu internete veikiančių sprendimų tuo, kad gali apdoroti ir atvaizduoti įvairių sričių ontologijas ir sukurtas kaip semantinio tinklo komponentas, kurį galima panaudoti kitose sistemose arba programinėje įrangoje.

Tiriant ontologijų vaizdavimo metodus ir technologijas, buvo remtasi literatūros šaltiniais [1], [2], [3] ir [4], kuriuose aprašoma semantinio tinklo koncepcija. Analizuojant ontologijų kalbas, ontologijų užklausų kalbą ir jų sintakses semantiniame tinkle buvo remtasi [2], [5], [6], [7], [8], [9] šaltiniais. Semantinio tinklo technologijos *Jena*, skirtos darbui su *Java* programavimo kalba, analizei buvo remtasi [5], [6], [10] šaltiniais, o *PHP* klasių bibliotekos *RAP* analizei - [11], [12] šaltiniais. Ontologijų vaizdavimo metodų analizė atlikta remiantis [13] ir vaizdavimo būdų analizė atlikta remiantis [4] ir [14].šaltiniais. Palyginant šiuo metu veikiančias interneto sistemas, remtasi [15], [16], [17] šaltiniais.

Darbo rezultatai ir jų svarba:

Ontologijų vaizdinio pateikimo modelis ir jo realizacija sukurta taip, kad veiktų kaip sistemos komponentas, kurį gali būtų įtraukti į kitas sistemas arba programinę įrangą. Šis semantinio tinklo komponentas gali nuskaityti įvairių sričių ontologijų duomenis ir juos atvaizduoti grafiniu modeliu. Ontologijų vaizdinis modelis išskiria skirtingų tipų elementus, juos sujungia tarpusavyje ryšiais ir

pateikia ryšių pavadinimus. Skirtingų tipų elementus lengva atskirti, jie pateikiami skirtingomis spalvomis, sujungti kryptingais ryšiais, kad naudotojui būtų aiškiai suprantama ontologijos specifikacija. Šiam sprendimui realizuoti pasirinkta naudoti *Jena* karkasą, kuris yra plačiausiai naudojamas darbui su ontologijomis ir turi geriausiai išvystytas ontologijų apdorojimo galimybes, ir *Arbor* biblioteka, gebanti atvaizduoti įvairius grafinius modelius, atskirti modelio elementus, sujungti elementus ryšiais ir suprantamai pateikti modelyje norimą atvaizduoti informaciją.

Ontologijų vaizdinio pateikimo modeliui tinkamumui įvertinti buvo atliktas eksperimentas, kurio metu modelis buvo patikrintas su 10 skirtingų ontologijų. Komponento nuskaitomų elementų skaičiaus ir faktinio ontologijoje esančių elementų skaičiaus palyginimas parodė, kad vaizdinio pateikimo modelis sugeba atvaizduoti visus teisingai aprašytus ontologijos elementus.

Realizuotą komponentą galima naudoti kitose sistemose ar programinėje įrangoje, kurios veikia nutolusiuose serveriuose. Kadangi ontologijų vaizdinio pateikimo modelis veikia kaip semantinio tinklo komponentas, todėl jį galima įtraukti tiesiai į kitų sistemų langus arba nukreipti naudotojus į viešai prieinamo ontologijos vaizdinio pateikimo langą.

Darbo struktūra:

- Skyriuje „Ontologijų vaizdavimo metodų ir technologijų analizė“ pateikiamos ontologijų vaizdavimo problemos, pateikta semantinio pasaulinio tinklo koncepcija, ontologijų kalbų analizė, semantinio tinklo technologijų, skirtų apdoroti ontologijas, analizė, išnagrinėti literatūroje siūlomi ontologijų vaizdavimo metodai ir palyginti literatūroje aprašyti modelių vaizdavimo būdai ir šiuo metu internete veikiantys sprendimai.
- Skyriuje „Grafinio ontologijų vaizdavimo įrankio modelis ir specifikacija“ pateikiami sistemai keliami reikalavimai, ontologijų vaizdinio pateikimo modelio veiklos procesas, ontologijos elementų metamodelis, navigavimo planas ir vartotojo sąsajos projektiniai langai.
- Skyriuje „Sistemos projektas“ pateikiama sistemos architektūra ir klasių modeliai, architektūros elementų sąveika, sistemos elgsenos modelis, loginė ontologijos duomenų schema, programinių komponentų architektūra ir diegimo modelis.
- Skyriuje „Sprendimo realizacija ir testavimas“ pateikiamas sprendimo realizacijos ir veikimo aprašas, pateikiami testavimo scenarijai ir testavimo rezultatai.
- Skyriuje „Eksperimentinis modelio tyrimas“ pateikiamas eksperimento planas, pagal kurį buvo vertinamas ontologijų vaizdinio pateikimo modelio efektyvumas, atlikto eksperimento rezultatai ir pateikiama rezultatų analizė, pateiktos sprendimo taikymo rekomendacijos.
- Išvadose pateikiami apibendrinti darbo rezultatai.

1. ONTOLOGIJŲ VAIZDAVIMO METODŲ IR TECHNOLOGIJŲ ANALIZĖ

1.1. Analizės tikslas

Šio darbo analizės tikslas yra išsiaiškinti ontologijoms naudojamų kalbų sąvokas ir sintaksę, ontologijų apdorojimo technologijas semantiniam tinklui ir grafinio vaizdavimo metodus ir technologijas bei nustatyti galimas panaudoti technologijas realizuojant modelio prototipą.

1.2. Tyrimo objektas, sritis ir problema

Tyrimo objektas yra ontologijų vaizdinio pateikimo vartotojui, siekiančiam atlikti paiešką, procesas.

Tyrimo sritis yra ontologijų grafinio vaizdavimo internete metodai ir technologijos.

Problema, kad nėra universalių įrankių, leidžiančių peržiūrėti ontologijų informaciją internete. Esami vaizdavimo būdai yra nelankstūs, neuniversalūs, nepateikia išsamios informacijos apie ontologijos elementų savybes.

1.3. Ontologijų kalbų analizė

Darbo tyrimo analizei atlikta literatūros, susijusios su ontologijų kalbomis, analizė. Siekiant įgyvendinti šio darbo tikslą, analizuojama literatūra apie semantinio pasaulinio tinklo koncepciją, ontologijų kalbas *RDF*, *OWL*, duomenų modelio užklausų kalbą *SPARQL*, semantinio tinklo technologijas *Jena*, *RAP*.

1.3.1. Semantinio pasaulinio tinklo koncepcija

Pasak Tim Berners-Lee, Semantinis pasaulinis tinklas yra dabartinio žiniatinklio išplėtimas, kuriame informacija turės aiškiai apibrėžtą prasmę ir leis kompiuteriams bei naudotojams geriau veikti kartu [1]. Semantinis pasaulinis tinklas leis prasmingai integruoti duomenis taikant semantines technologijas [2]. Nors formalus semantinis informacijos kodavimas semantiniame pasauliniame tinkle leidžia programoms protingai apdoroti informaciją, paprastam vartotojui aiškiai pateikti šiuos duomenis nėra paprasta. Pagrindinė vartotojo sąsajos problema – kaip suteikti vartotojams daugiau galimybių naudotis Semantiniu pasauliniu tinklu, paslepiant jo sudėtingumą.

Semantinis pasaulinis tinklas yra pagrįstas informacijos kodavimu formaliu semantinio lygmens būdu. Semantinis pasaulinis tinklas turi bendrą duomenų modelį.

Papildomai prie klasikinių interneto žiniatinklių, *W3C* padeda sukurti technologiją, kuri palaikytų „duomenų žiniatinklius“. Galutinis duomenų žiniatinklių tikslas yra, kad kompiuteriai galėtų padaryti daugiau naudingo darbo ir padėtų kurti sistemas, kurios galėtų palaikyti patikimamas sąveikas tinkle. Sąvoka „Semantinis tinklas“ yra *W3C* vizija, kuri apibūdina duomenų žiniatinklius. Semantinio tinklo technologijos suteikia galimybę žmonėms kurti duomenų saugyklas žiniatinklyje, kurti žodynus bei aprašyti įvairias taisykles šių duomenų apdorojimui. Semantinio tinklo duomenims tvarkyti panaudojamos *RDF*, *SPARQL*, *OWL* ir *SKOS* technologijos [3].

Ontologija – tam tikros dalykinės srities sąvokų visumos specifikacija. Ontologijų mechanizmas leidžia formuoti prasmingus hierarchinius ryšius tarp objektų. Ontologijos apibrėžia nagrinėjamos srities:

- Sąvokas, esybių (reiškinių, daiktų) tipus;
- Sąvokų hierarchijas, esybių tipų tarpusavio sąryšius, priklausomybes;
- Aksiomas, taisykles, dėsningumus apie esybių tipus ir sąryšius [4].

1.3.2. RDF analizė

RDF (*The Resource Description Framework*) yra duomenų modelis, skirtas atvaizduoti informacijai apie pasaulinio tinklo resursus [2]. Grindžiant viskuo, *RDF* duomenų modelis nurodo kaip

informacija bus atvaizduojama semantiniame pasauliniame tinkle. Modelis yra pagrįstas paprastų tripletų rinkiniais, kuriuos sudaro:

Subjektas (angl. *Subject*);

Savybė / Ryšys (angl. *Property / Relationship*);

Objektas (angl. *Object*);

RDF duomenų modelyje kiekvienas subjektas ir ryšiai, naudojami sąsajoje, turi unikalų identifikatorių, o objektas yra kaip kitas subjekto identifikatorius arba žodinė reikšmė. Naudojant tuos pačius identifikatorius keliuose tripletuose, suformuojamas mazgų ir lankų tinklas, jungiantis tripletus kartu į grafikus, taip sukuriant daugiau kompleksiškų informacijos formų modelių.

Nors *RDF* duomenų modelis suteikia paprastą būdą pateikti beveik visą informaciją, paprastai jis nenurodo ką naudojamas subjektas ir ryšiai reiškia.

RDF sintaksė yra labai panaši į *XML* sintaksę [5]. *RDF* sintaksės pavyzdys pateiktas 1.1 lentelėje.

1.1 lentelė. RDF sintaksės pavyzdys

```
<owl:Class rdf:ID="Daiktas"/>
<owl:Class rdf:ID="Regionas"/>
<owl:Class rdf:ID="Vartojamas dalykas"/>
```

1.3.3. OWL analizė

OWL (*Web Ontology Language*) yra ontologijų kalba, skirta atpažinti ontologijas. *OWL* paremtas *RDF/XML* struktūra [6]. Standartinė ontologijų kalba semantiniame pasauliniame tinkle yra *OWL*. Šiuo metu plačiai naudojama naujausia *OWL 2* ontologijų kalba, kuri pakeičia *OWL 1* ontologijų kalbą. Literatūros šaltiniuose didžiausias dėmesys kreipiamas *OWL 2* ontologijų kalbai.

OWL 2 ontologijos pateikia klases, savybes, individus ir duomenų reikšmes, kurie aprašyti semantinio tinklo dokumentuose. *OWL 2* ontologijos gali būti naudojamos su informacija, aprašyta *RDF* ir *OWL 2* ontologijose, apibūdinant juos kaip *RDF* dokumentus. *OWL 2* teikia prasmingus darinius, kurie apibrėžiami pagal jų struktūrą ontologijoje. Taip pat prasmingiems dariniams yra apibrėžiama funkcinio stiliaus sintaksė su pavyzdžiais ir neformaliais aprašymais [7].

OWL sintaksės pavyzdys pateiktas 1.2 lentelėje.

1.2 lentelė. OWL sintaksės pavyzdys

```
<owlx:Class owlx:name="Daiktas" owlx:complete="false" />
<owlx:Class owlx:name="Regionas" owlx:complete="false" />
<owlx:Class owlx:name="Vartojamas dalykas" owlx:complete="false" />
```

1.3.4. SPARQL analizė

SPARQL (*SPARQL Protocol and RDF Query Language*) – tai *RDF* užklausų kalba. Tai yra užklausų kalba duomenų bazėms, galinčioms gauti ir manipuliuoti saugomais duomenimis *RDF* formate. 2013 metais *SPARQL 1.1* tapo oficialia *W3C* rekomendacija. *SPARQL* leidžia užklausas, kurios susideda iš trigubų modulių, jungtukų, atjungtukų ir neprivalomų modulių [8].

Atsižvelgiant į užklausas, kad skaityti duomenis iš duomenų bazės, *SPARQL* kalba nurodo keturis skirtingus užklausos variantus skirtingiems tikslams:

- SELECT užklausa – naudojama išgauti eilučių vertes iš *SPARQL*, rezultatai yra grąžinami lentelės formatu.
- CONSTRUCT užklausa – naudojama išgauti informaciją iš *SPARQL* ir transformuoti rezultatus į galiojantį *RDF*.
- ASK užklausa – naudojama pateikti paprastą *true* / *false* rezultatą *SPARQL* užklausai.
- DESCRIBE užklausa – naudojama išgauti *RDF* grafą su *SPARQL* užklausa, kai leidžiama turinį nustatyti pagal tai, kas manoma yra naudinga informacija.

Kiekviena iš šių užklausų naudoja WHERE bloką, kad apriboti užklausą. Tačiau DESCRIBE užklausai WHERE blokas yra nebūtinas [8].

SPARQL sintaksės pavyzdžiai pateikti 1.3 lentelėje.

1.3 lentelė. SPARQL sintaksės pavyzdys [9]

Nr.	Užklausa	Rezultatas

1.	SELECT ?title
	WHERE
	{
	<http://example.org/book/book1>
	<http://purl.org/dc/elements/1.1/title>
	?title .
	}
		title
		"SPARQL Tutorial"

2.	PREFIX foaf:
	<http://xmlns.com/foaf/0.1/>
	SELECT ?name ?mbox
	WHERE
	{
	?x foaf:name ?name .
	?x foaf:mbox ?mbox
	}
		name: "Johnny Lee Outlaw" — mbox: <mailto:jlow@example.com>
		name: "Peter Goodguy" — mbox: <mailto:peter@example.org>

1.3.5. Semantinio tinklo technologijos

Semantinio tinklo technologijos taikomosioms programoms suteikia programavimo sąsają (*API*) dirbti su *RDF* (*Resource Description Framework*) grafais. Dauguma sistemų palaiko ontologijų kalbą *RDF*, *RDFS* ir *OWL*. Be to, *SPARQL* užklausų kalba suteikia galimybes atlikti pagrindines funkcijas, tokias kaip „*select*“, „*insert*“, „*update*“, ir „*delete*“ [10]. Semantinio tinklo technologijų analizei pasirinktos dvi technologijų sistemos – *Jena* ir *RAP*.

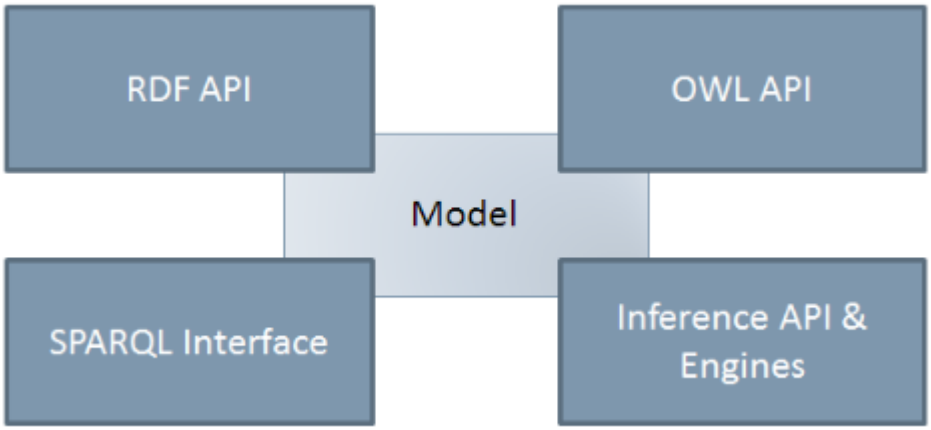
1.3.5.1. Jena analizė

Jena – semantinio pasaulinio tinklo sistema, skirta dirbti su *Java* programavimo kalba. *Jena* buvo sukurta ir palaikoma „HP Labs“ iki 2009m.. Tačiau nuo to laiko ši sistema yra palaikoma atviro kodo bendruomenės. *Jena* palaiko *RDF*, *RDFS*, *OWL* ir *SPARQL* [10]. Ši sistema palaiko daugumą *RDF* standartų ir šiuo metu plačiausiai naudojama technologijų sistema darbui su ontologijomis. Viena ypatybė, kuo pasižymi *Jena*, tai įvairių mechanizmų palaikymas, kurie padeda gauti papildomos informacijos iš ontologijos [10]. Be to, ji suteikia galimybę nuskaityti ir parašyti bendrus *RDF* žymėjimus:

- *RDF / XML*;
- *N3*;
- N-tripletus [6];

Taip pat *Jena* plėtoja *RDF* serverį *Joseki* (ir jo protėvį *Fuseki*), kuris suteikia *HTTP* sąsają *RDF* ir palaiko *SPARQL*.

Semantinio tinklo sistema *Jena* sudaryta iš 5 pagrindinių komponentų. Komponentai *RDF API* ir *OWL API* suteikia prieigą prie ontologijos ir *RDF* grafų. *SPARQL Interface* realizuoja visą *SPARQL* protokolo funkcionalumą ir *Inference API & Engines* aprūpina integruotais ir išoriniais mechanizmais ontologijos apdorojimui. Visi šie komponentai yra apjungti į modelio sąsają (*Model*) [10]. *Jena* sistemos pagrindinių komponentų išdėstymas pateiktas 1.1 paveiksle.

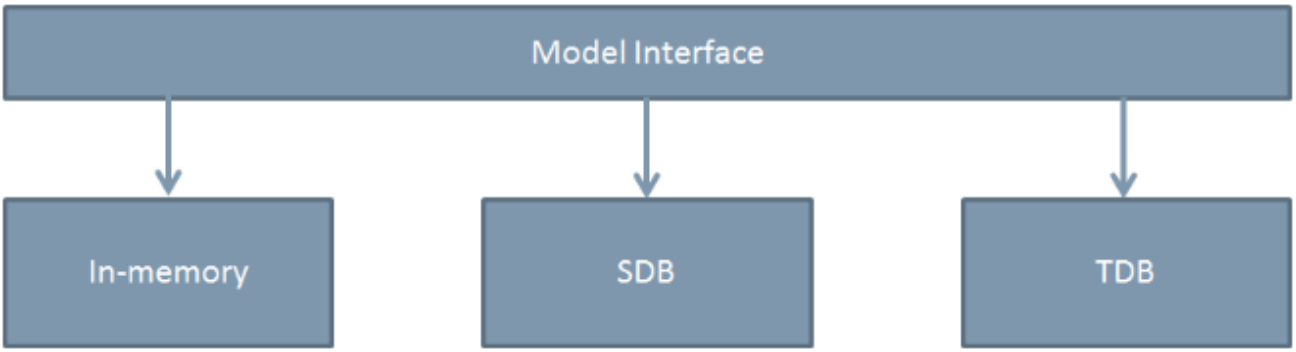


1.1 pav. Jena sistemos pagrindiniai komponentai (šaltinis [10])

Jena pateikia *RDF* grafą kaip abstraktų modelį. Modelis iš esmės yra daugelio formuluočių rinkinys ir suteikia metodus, kad atlikti pridėjimo, pašalinimo ir vykdymo užklausas. Tai reiškia, kad yra įmanoma sukurti *RDF* grafą naudojant *Java API*, bet papildomai jis gali būti sudarytas iš *RDF* failų [10].

Modelio duomenų išsaugojimui *Jena* sistema palaiko tris būdus, pateiktus 1.2 paveiksle, duomenų saugojimui:

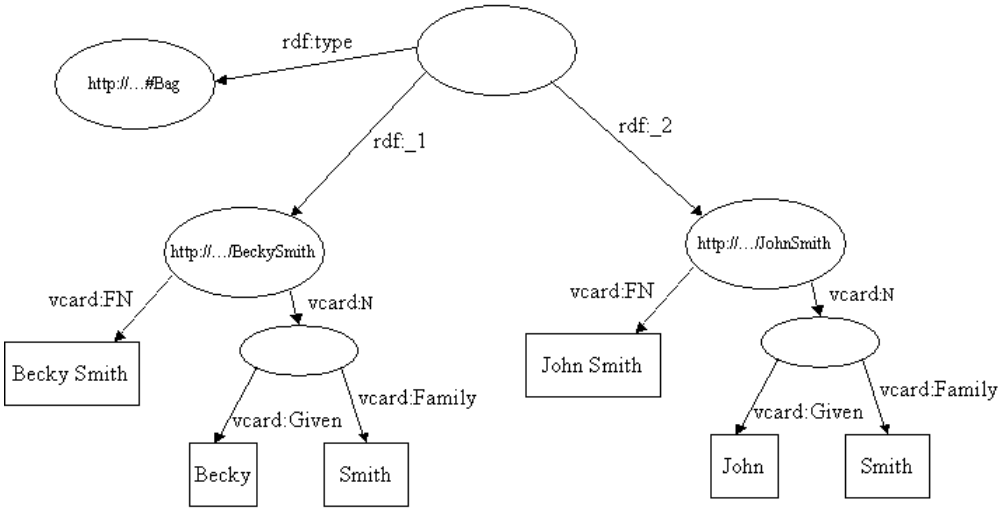
- Atmintyje;
- Reliacinėje duomenų bazėje (*SDB*);
- Specializuotoje duomenų bazėje saugoti tripletams (*TDB*);



1.2 pav. Jena sistemos pagrindiniai komponentai (šaltinis [10])

Modelio duomenų nuskaitymą galima atlikti vienu iš dviejų būdų. Vienas būdas juos pasiekti per *Java API*, kuris skirtas apdoroto *RDF* ir *OWL* ontologijų kalbas. Kitas būdas duomenis pasiekti yra per užklausų variklį *ARQ*, kuris platinamas su *Jena*. *ARQ* užklausų variklis palaiko standartinį *SPARQL* ir *SPARQL/Update* (*SPARQL 1.1*) užklausų kalbas. Taip pat jis palaiko išplėstines *SPARQL* užklausas ir skirtingus plėtinius, pavyzdžiui apibendrinimus. Kitas plėtinys yra laisva teksto paieška naudojant teksto paieškos variklį *Lucene*. Su šiomis sąsajomis *Jena* užtikrina gerą prieigą prie aplikacijos duomenų [10].

Semantiniam tinklui palaikyti, *OWL* teikia trijų lygių standartus: *OWL Lite*, *OWL DL* ir *OWL Full*. *OWL Full* yra labiausiai ekspresyvi. *OWL Lite* suteikia galimybę lengviausiu būdu dirbti su pilnu paslaugų mechanizmų paketu, kuris neįmanomas *OWL Full*. Šie du lygiai yra atskirti sintaksiškai. Dvi suderinamos semantikos gali būti pritaikytos *OWL Full* arba *OWL DL* standartui. *OWL Full* yra semantikos praplėtimas, visiškai paremtas *RDF*. OWL DL abstrakti sintaksė yra surišama su *RDF* kaip kintama kalba [6].



1.3 pav. Jena RDF modelis (šaltinis [5])

1.3.5.2. RAP analizė

RAP (*RDF API for PHP*) – yra semantinio pasaulinio tinklo sistema, skirta *PHP* programavimo kalbai. Ji siūlo funkcijas, skirtas apdoroti, manipuliuoti, saugoti, vykdyti užklausoms ir aptarnauti *RDF* grafus. *RAP* buvo pradėtas plėtoti kaip atviro kodo projektas Freie Universität Berlyne 2002 metais ir buvo praplėstas semantinio pasaulinio tinklo bendruomenės kodu [11]. *RAP* sistemos branduolys palaiko du būdus saugoti RDF grafas:

- Atmintyje;
- Reliacinėje duomenų bazėje (*SDB*) [12];

Dėka šių saugyklų, *RAP* suteikia plačias programavimo sąsajas manipuliuoti *RDF* grafais skirtinguose abstrakcijų sluoksniuose. Taip pat *RAP* palaiko *RDFS* sąsają, taip pat ir *OWL* apribojimus, leidžiančius programuotojams dirbti su nuspėjamais teiginiais. *RAP* sudaro įvairūs įrankiai, papildantys sistemą:

- *RDF / XML* analizatorius;
- Integruotas *RDF* serveris;
- Grafinė vartotojo sąsaja, skirta valdyti *RDF* modelius duomenų bazėje, panaudojant *RDQL* užklausų kalbą [11];

RAP sistema gali pasiūlyti paprastus, primityvius budus RDF grafų nuskaitymui ir įrašymui pagal pasirinktą semantinio tinklo kalbą [12]. Šios sistemos palaikomos kalbos:

- *RDF/XML*;
- *N3* (kitaip dar vadinama *Notation3*);
- N-tripletai;

Dirbant su atmintyje išsaugotais duomenimis, naudojant *RAP* sistemą, kyla viena esminė problema, kad įvykdžius *PHP* kodo veiklą, visi sukurti ir panaudoti modeliai dingsta, nebent būtų išsaugoti į failą. Nors ir išsaugoma į failą, vis tiek tą failą nuskaitant tektų visus *RDF* duomenis dar kartą nagrinėti, kai *PHP* kodas būtų vėl paleistas. Šis procesas reikalauja daug laiko, ypač su dideliais duomenų kiekiais. Norint išspręsti šią problemą, *RDF API* palaiko *RDF* duomenų saugojimą reliacinėje duomenų bazėje. Duomenų saugojimas duomenų bazėje ne tik išsaugo daug darbinės stoties operatyviosios atminties, bet suteikia greitą *RDF* duomenų nuskaitymą iš duomenų bazės [12].

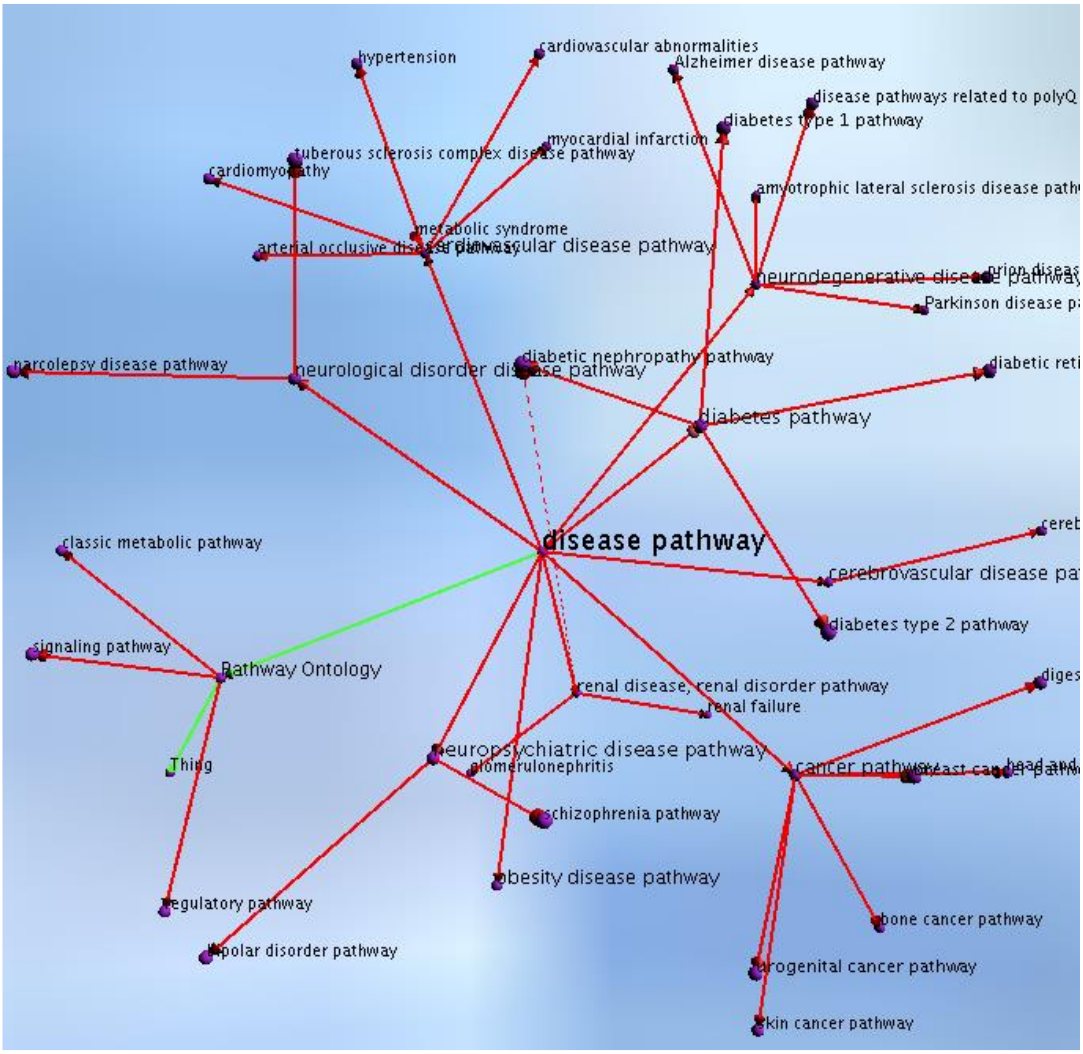
1.4. Modelių vaizdavimo metodų analizė

Pasak Julia Dmitrieva ir Fons J. Verbeek, ontologijų kalba gali būti labai ekspresyvi. Atliktoje literatūros analizėje autoriai apibūdina dvi skirtingas vaizdavimo technikas: viršūnė-ryšys ir izoliavimo technika. Abi vaizdavimo technikos atvaizduoja ontologijos struktūrą skirtingai.

Vaizdavimo metodas „viršūnė-ryšys“ atvaizduoja ontologiją kaip grafų struktūrą. Ši grafų struktūra, remiantis ontologijos hierarchija ir savybėmis, gali būti atvaizduojama kaip skirtingos geometrijos: Euklido, hiperbolės ir sferos. Vaizdavimo metodas „izoliavimas“ atvaizduoja tik hierarchinę struktūrą. Vietoje tradicinės dviejų dimensijų technikos, autoriai panaudoja sferos modelį ir sumodeliuoja trijų dimensijų vaizdavimo metodą [13].

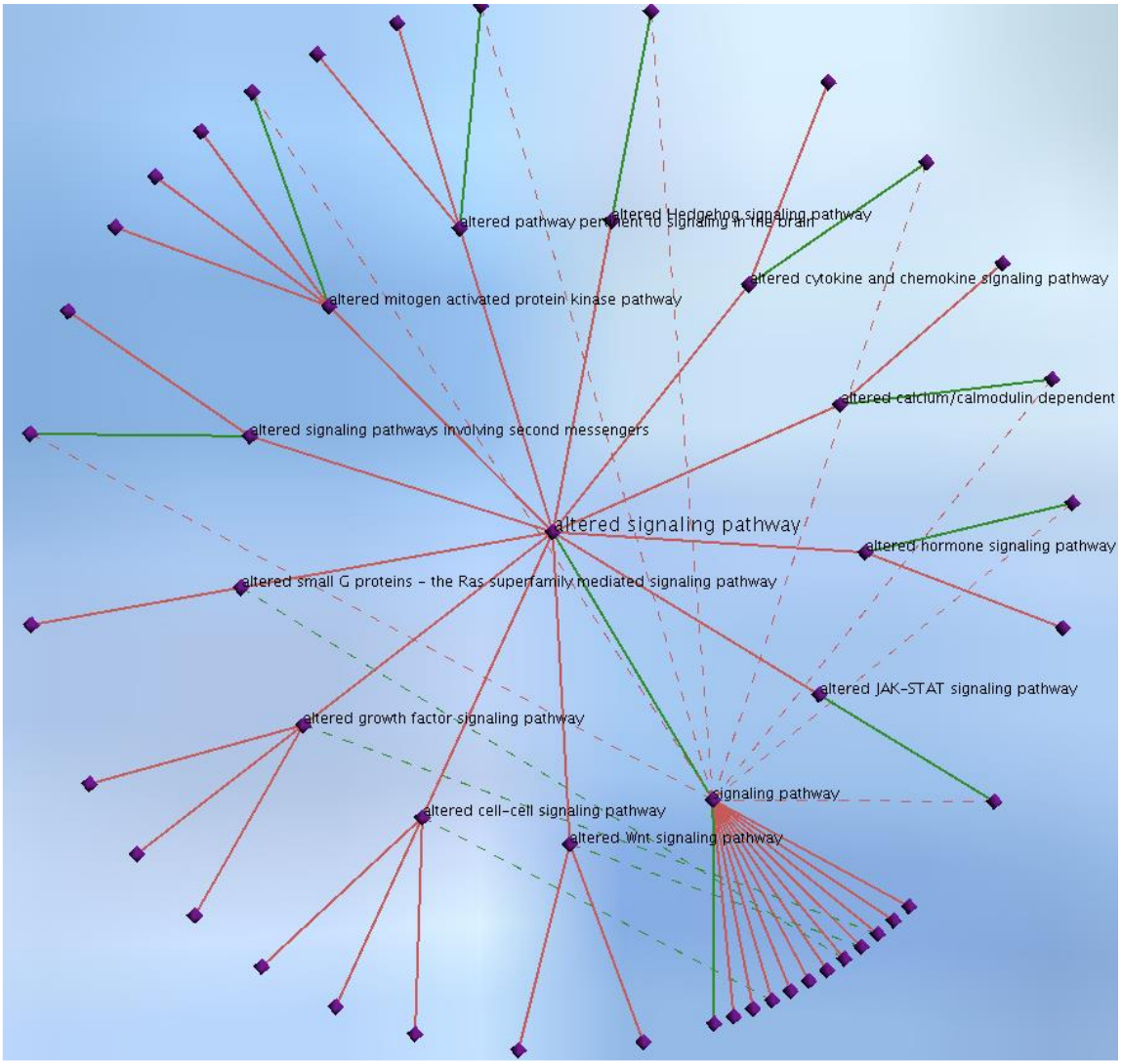
1.4.1. Vaizdavimo metodas geometriniais modeliais

Ontologijos vaizdavimui semantiniame tinkle galima panaudoti įvairius geometrijos modelius. Šiam metodui galima buvo pasitelktos dvi Euklido geometrijos formos: sfera ir diskas. Sferos vaizdavimo metode pradinės viršūnės atvaizduojamos viduryje modelio. Susijusios viršūnės supa pradinę viršūnę ir yra atitolusios vienodais atstumiąs ant įsivaizduojamo sferos paviršiaus [13]. Ontologijos vaizdavimo pavyzdys sferos paviršiuje pateiktas 1.4 paveiksle.



1.4 pav. Ontologijos vaizdinis modelis sferos paviršiuje (šaltinis [13])

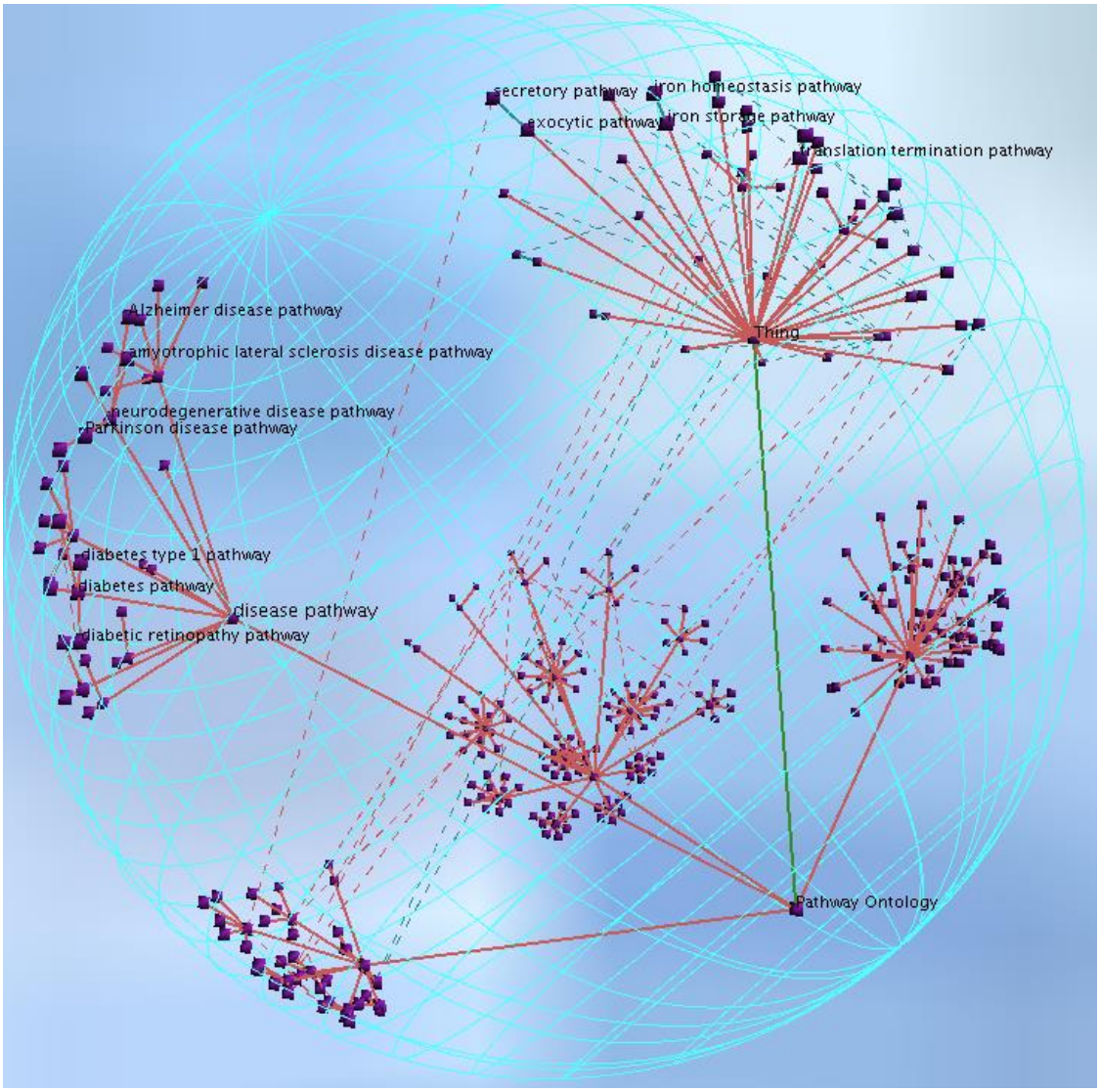
Disko modelis, pagal Klein modelį ir Poincar‘e disko modelį, yra paremtas hiperboline geometrija [13]. Ontologijos vaizdavimo pavyzdys disku pateiktas 1.5 paveiksle.



1.5 pav. Ontologijos vaizdinis modelis disku (šaltinis [13])

1.4.2. Vaizdavimo metodas izoliavimo požiūriu

Šis vaizdavimo metodas yra sudarytas panaudojant vaizdavimą geometriniais modeliais. Viršūnės išdėstytos sferos paviršiuje pagal hierarchijos sluoksnius. Pirmasis sluoksnis suformuojamas iš šakninių hierarchijos viršūnių. Antrasis sluoksnis suformuojamas iš vaikinių viršūnių, kurios priklauso pirmojo sluoksnio viršūnėms. Kiekviena viršūnė atvaizduojama kaip atskira sfera, kurios dydis priklauso nuo atšakų skaičius. Vaikinės sferos įtraukiamos į tėvines sferas ir yra įtraukiamos į bendrą sferos paviršių, kurio spindulys yra didesnis, kad suteikti vizualizacijai reljefo įvaizdį [13]. Ontologijos vaizdavimo pavyzdys sferų izoliavimo metodu pateiktas 1.6 paveiksle.



1.6 pav. Ontologijos vaizdinis modelis izoliavimo požiūriu (šaltinis [13])

1.5. Modelių vaizdavimo būdų analizė

Pasak Simon Suigen Guo ir Christine W. Chan, programiniai įrankiai gali padėti sumažinti pastangas, reikalingas sukurti ontologijai ir jos žinių bazei. Ontologijos vaizdavimo įrankiai gali padėti pagerinti žmonių supratimą apie konceptualius modelius, iliustruojant vizualiniame grafiniame modelyje. Vizualizacija iškart suteikia visą vaizdą apie žinių bazę ir jos struktūrą dalykinės srities ekspertui ir leidžia išvengti varginančio, išsamaus ryšių tarp sąvokų nagrinėjimo, išreikštų ontologijų kalba. Šie įrankiai pagerina žinių įsisavinimą ir pagreitina ontologijos kūrimo procesą. Kitaip tariant, naudotojas greitai gali nustatyti sąvokas ir jų santykius su kitomis sąvokomis ir pavaizduoto grafinio modelio vietoje to, kad bandytų tai suprasti iš ontologijos failo [4].

1.5.1. Esami modelių vaizdavimo būdai

Ontologijų modelių vaizdavimui šiuo metu jau yra sukurti modelių vaizdavimo įrankiai: OWLViz, Jambalaya, OntoSphere, Onto3Dviz [4]. Pateikiamas šių modelių vaizdavimo palyginimas 1.4 lentelėje.

1.4 lentelė. Modelių vizualizacijos palyginimas [4]

	OWLViz	Jambalaya	OntoSphere	Onto3Dviz
Konceptų hierarchijos atvaizdavimas	Hierarchinis medis	Palaiko įvairius išdėstymus: hierarchinis medis, žemėlapis ir t.t.	Konceptų hierarchija neatvaizduojama	Hierarchinis 3D medis
Peržiūros perspektyvų skaičius	1	Pateikia 5 grafines perspektyvas	Pateikia keturias 3D scenas	1 dinaminė 3D peržiūra
Žemiausias peržiūros lygis	Maksimalus peržiūrų lygis 10	Neribojama	Neribojama	Neribojama
Išdėstymas	Ribotas, netelpa kompleksiniai modeliai į ekraną	Optimizuotas, bet pavadinimai užsirašo ant viršaus	Optimizuotas	Ribotas, netelpa kompleksiniai modeliai į ekraną
Priartinimas	Nepalaikomas	Ribotas priartinimas	Lankstus priartinimas	Lankstus priartinimas
Koncepto paieška	Palaikoma	Palaikoma	Palaikoma	Nepalaikoma
Koncepto filtravimas	Palaikoma vartotojui nurodant konceptų lygmenį	Palaikoma suskleidžiant vizualinius mazgus	Palaikoma vartotojui keičiant 3D perspektyvą	Tik palaikoma paslepiant statines arba dinamines žinias
Koncepto redagavimas	Palaikoma per Protege redaktorių	Palaikoma per Protege redaktorių	Palaikoma per Protege redaktorių	Nepalaikoma
Fokusavimas	Nepalaikoma	Palaikoma keičiant grafinį sluoksnį	Palaikoma vartotojui keičiant 3D perspektyvą	Palaikoma vartotojui manipuliuojant 3D modeliu
Statinės žinių bazės palaikymas	Palaikoma	Palaikoma	Palaikoma	Palaikoma
Dinaminės žinių bazės palaikymas	Nepalaikoma	Nepalaikoma	Nepalaikoma	Palaikoma
Papildomos vizualinės detalės	Ribota	Ribota	Optimizuota	Optimizuota
2D ar 3D	2D	2D	3D	3D
Sistemos reikalavimai	Protege ir GraphViz įskiepis	Protege	Protege ir Java 3D	Java 3D

Išvardinti modelių vaizdavimo įrankiai netinkami dėl keliamų sistemos reikalavimų ir neturi pritaikymo galimybių semantiniam tinklui, todėl realizavimui galimas sprendimo būdas panaudoti nemokamą modelių vaizdavimo biblioteką *Arbor*.

1.5.2. Arbor analizė

Arbor – nemokama atviro kodo grafų vizualizacijos biblioteka, sukurta interneto naudotojų ir *jQuery*. *Arbor* ne bando apimti viską, bet suteikia veiksmingą ir į tikslą orientuotą išdėstymo algoritmą. Taip pat abstrakcijas grafo organizavimui ir ekrano atnaujinimo palaikymui [14].

Jis suteikia tikrąjį ekrano modeliavimą patiems kūrėjams. Tai reiškia, kad galima ją naudoti su paveikslėliais, *SVG* ar net išdėstomais *HTML* elementais, tinkamas bet kokios rezoliucijos naudotojams ir veiklos poreikiams.

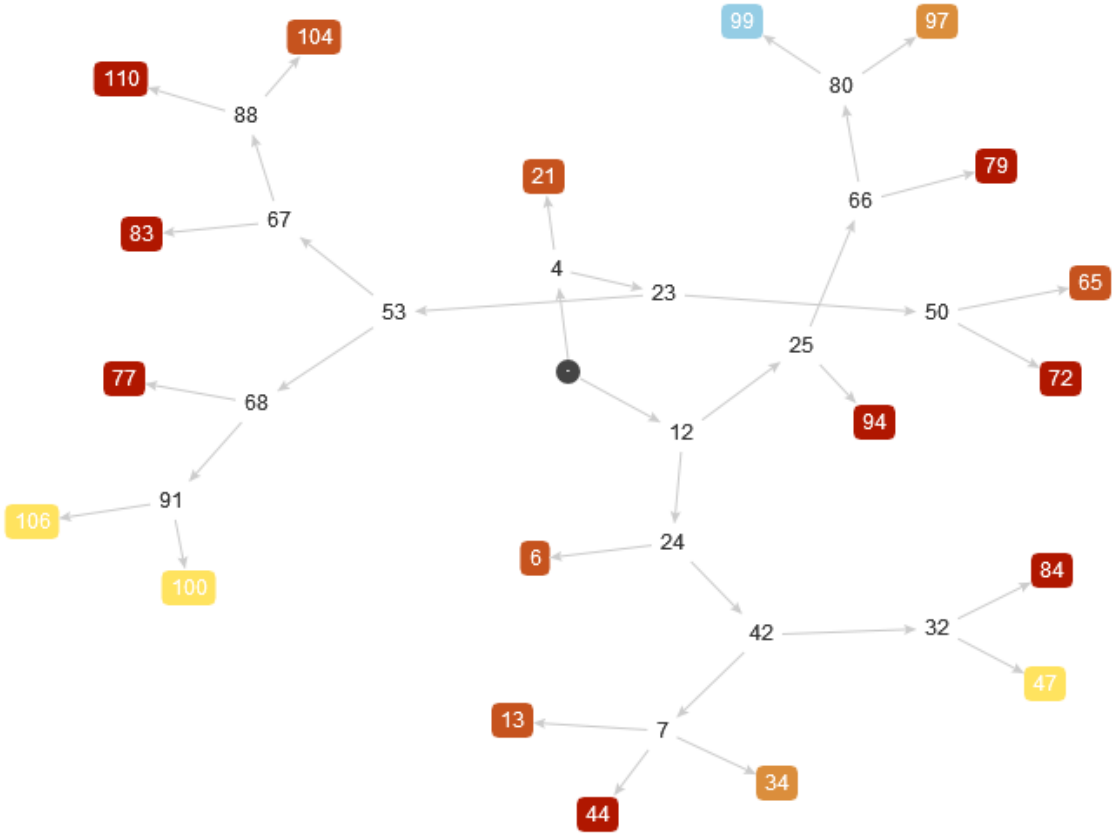
Todėl, kodui, kuris rašomas, gali tekti skirti daugiau laiko tokiems dalykams, kad projektas būtų unikalus – grafų duomenims ir vizualizacijos stiliui, o ne eikvoti laiką fizikai ir matematikai, kad sukurti išdėstymą.

Kiekvienas grafiko susikirtimo taškas su sujungtomis linijomis tarpusavyje atvaizduoja ryšius tarp taškų.

Grafike galima išskirtinai atvaizduoti susikirtimo taškus, kurie parodo grafiko atšakos pabaigos tašką.

Grafiko detalių sistema saugo taškų ir briaunų koordinates. Modeliavimui progresuojant sugeba atnaujinti taškų ir briaunų koordinates. Grafiko objektams pritaikytos šiuolaikinės fizikos būsenos, o taip pat suteikia galimybę pritaikyti metaduomenis.

Arbor nekelia jokių sistemos reikalavimų vartotojui – pakanka interneto naršyklės. Taip pat suteikia galimybes pritaikyti biblioteką pagal poreikius, išplėsti jos galimybes ir vizualizacijos stilių, todėl tokią biblioteką galima pritaikyti ontologijų vaizdinio modelio atvaizdavimui semantiniame pasauliniame tinkle.



1.7 pav. *Arbor* modelis (šaltinis [14])

1.6. Vartotojų analizė

1.6.1. Vartotojų aibė, tipai ir savybės

Ontologijų vaizdinio pateikimo modelis naudingas interneto naudotojams - paieškos kūrėjams.

1.6.2. Vartotojų tikslai ir problemos

Vartotojams, norintiems atlikti paiešką, yra poreikis išanalizuoti ontologiją, kuria remiasi paieška. Patogių įrankių, kurie pateiktų ontologiją vaizdiniu modeliu ir remtųsi paieškai naudojama ontologija, trūksta.

1.5 lentelė. Vartotojų problemos

Problema	Kaip viskas vyksta dabar
Ryšiai tarp ontologijų neįvardinti	Modeliuose nepateikiamas ryšio tarp ontologijų pavadinimas
Nėra universalaus modelio	Negalima peržiūrėti modeliuose universalių ontologijų. Modeliuose naudojami specifinių ontologijų duomenys.

1.7. Esamų sprendimų analizė

Norint geriau suprasti ontologijų vaizdavimo metodus semantiniame tinkle, buvo analizuojamos interneto sistemos, kuriose ontologijų duomenys pateikiami vaizdiniais būdais. Esamų sprendimų analizės metu buvo bandoma patikrinti ontologijų vaizdinio pateikimo įrankius internete, bet šiuo metu nei vienas toks įrankis nebeveikia, todėl esamų sprendimų analizei pasirinktos tam tikroms dalykinėms sritims pritaikytos semantinio tinklo sistemos:

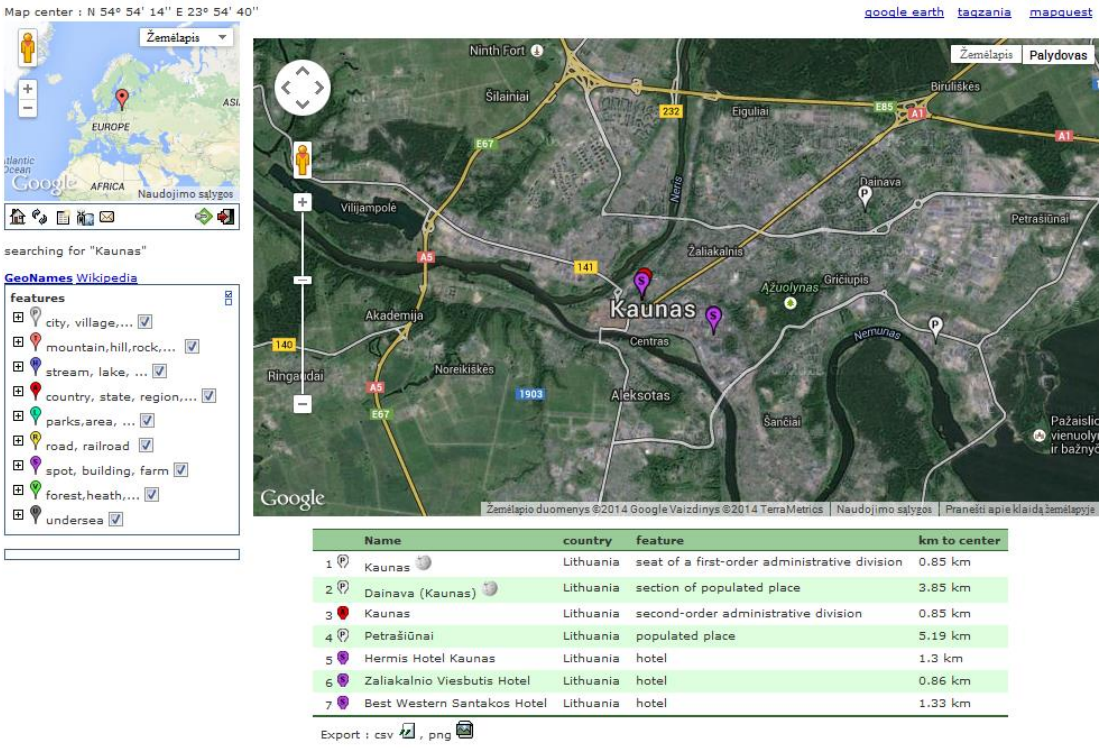
- GeoNames – www.geonames.org
- Epicurious.com – emm.newsexplorer.eu
- GeneOntology – www.geneontology.org

Kiekvienas analizuotas, šiuo metu veikiantis sprendimas, ontologijų elementus ir savybes pateikia skirtingais būdais.

1.7.1. GeoNames

GeoNames interneto sistema ontologijų vaizdavimui naudoja žemėlapį (Google Maps). GeoNames pateikia šalių, miestų, vietovių, lankytinų vietų ontologijų žemėlapį. Objektai žemėlapyje susiejami su jų savybėmis ir pateikiami naršant žemėlapį. Taip pat ontologijas galima peržiūrėti naršant ontologijų medį – nuo šalių, miestų iki lankytinų vietų. Peržiūrint ontologijos informaciją pateikiami duomenys su kuo ontologija turi ryšius – šalis turi kokius miestus, miestas kokius rajonus ir lankytinas vietas. Interneto sistema suteikia galimybę gauti visus ontologijų duomenis parsisiųsti *RDF/OWL* formatu.

Pavyzdinis GeoNames modelis pateikiamas 1.8 paveiksle.



	Name	country	feature	km to center
1	Kaunas	Lithuania	seat of a first-order administrative division	0.85 km
2	Dainava (Kaunas)	Lithuania	section of populated place	3.85 km
3	Kaunas	Lithuania	second-order administrative division	0.85 km
4	Petrašiūnai	Lithuania	populated place	5.19 km
5	Hermis Hotel Kaunas	Lithuania	hotel	1.3 km
6	Zaliakalnio Viesbutis Hotel	Lithuania	hotel	0.86 km
7	Best Western Santakos Hotel	Lithuania	hotel	1.33 km

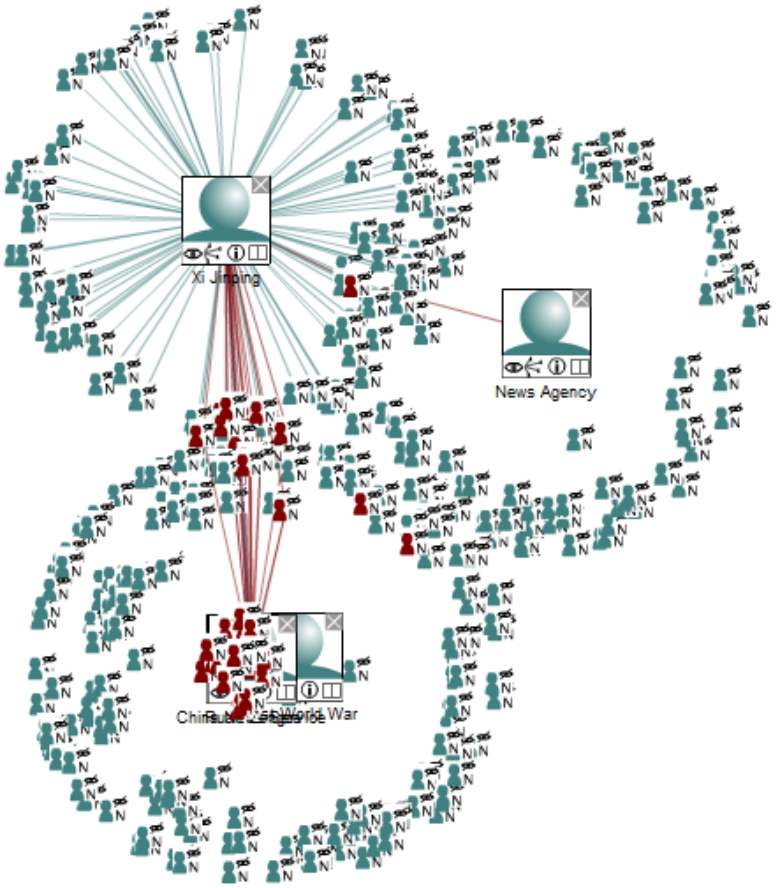
Export : csv , png

1.8 pav. GeoNames modelis (šaltinis [15])

1.7.2. EMM News Explorer

EMM News Explorer interneto sistema ontologijų vaizdavimui naudoja Flash įrankį. EMM News Explorer gali atvaizduoti asmenų ryšius su kitais asmenimis ir ryšius tarp jų, susijusių asmenų ratus. Tokiu būdu matomas ontologijų medis, kai viena ontologija susiejama su kita. Interneto sistemoje yra pateikiamas pilnas ontologijos savybių aprašymas.

Pavyzdinis EMM News Explorer modelis pateikiamas 1.9 paveiksle.

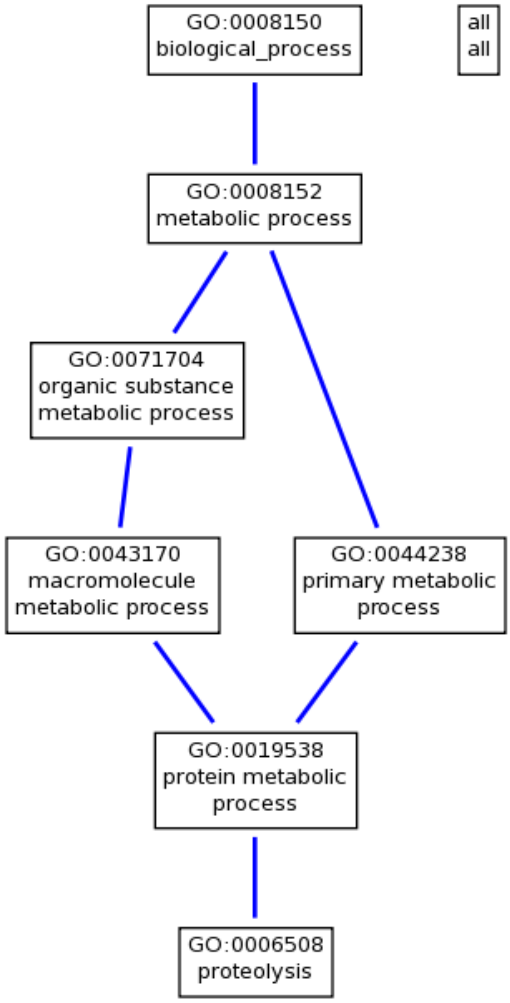


1.9 pav. EMM News Explorer modelis (šaltinis [16])

1.7.3. The Gene Ontology

The Gene Ontology interneto sistema ontologijų vaizdavimui naudoja grafinį atvaizdavimą klasėmis ir ryšiais. The Gene Ontology pateikia genų, proteinų ir terminų ontologijas.

Pavyzdinis The Gene Ontology modelis pateikiamas 1.10 paveiksle.



1.10 pav. The Gene Ontology modelis (šaltinis [17])

Esamų sprendimų analizės palyginimui pateikta 1.6 lentelė, kurioje atrinkti esminiai sprendimų kriterijai ir pažymėtas kiekvieno esamo sprendimo atitikimas arba neatitikimas kriterijui. Esamų sprendimų trūkumai – tinkamumas tik tam tikrai sričiai ir negebėjimas pavaizduoti ryšių pavadinimų, kurie gali būti labai skirtingi.

1.6 lentelė. Esamų sprendimų palyginimas

Kriterijus	**GeoNames**	**EMM News Explorer**	**The Gene Ontology**
Paieška	+	+	+
Grafinis vaizdavimas	Žemėlapis	Ontologijos modelis	Ontologijos modelis
Ontologijų hierachija	+	+	+
Ontologijų elementų ryšiai	+	+	+
Ontologijų elementų ryšių pavadinimai	-	-	-
Galimybė parsisiųsti duomenis (OWL/RDF)	+	-	+
Ontologijos savybių peržiūra	+	+	+
Universalumas	-	-	-

1.8. Tyrimo tikslas ir uždaviniai

Darbo tikslas - suteikti galimybes patogiai peržiūrėti išsamią ontologijos informaciją, sudarant ontologijų vaizdinio pateikimo modelį, kuris apimtų ontologijos peržiūrėjimo procesą ir pateiktų išsamią informaciją apie ontologijos turinį, bei sukuriant šį modelį realizuojančios programinės įrangos prototipą.

Uždaviniai:

1. Išanalizuoti:
   1.1. Ontologijų kalbos *OWL 2* ir užklausų kalbos *SPARQL* sąvokas.
   1.2. Esamus ontologijų vaizdinio pateikimo metodus, technologijas, interneto sistemas.
2. Sudaryti ontologijų vaizdinio pateikimo modelį.
3. Suprojektuoti ontologijų vaizdinio pateikimo modelį realizuojančią informacinę sistemą.
4. Realizuoti programinį prototipą.
5. Atlikti eksperimentą modelio ir sistemos tinkamumui įvertinti.

1.9. Siekiamas sprendimas

Siekiamas sprendimas yra ontologijų vaizdavimo modelis ir jį realizuojantis prototipas, kuris tiktų semantinės analizės ir paieškos sistemoms, įskaitant ir lietuviškas sistemas.

Sukurtas prototipas leistų vartotojams, norintiems atlikti paiešką, peržiūrėti ontologiją ir geriau suprasti, kokią paiešką galima atlikti.

Prototipą bus siekiama sukurti universalų, kuris tiktų bet kokiai sričiai, o eksperimentas bus atliekamas su pasirinkta dalykine sritimi.

1.10. Analizės išvados

1. Ontologijų vaizdinio pateikimo sprendimų analizė parodė, kad veikiantys sprendimai yra pritaikyti specialių sričių ontologijoms, o sprendimų, pritaikytų įvairių sričių ontologijoms, šiuo metu nėra.
2. Aprašyti informacijos šaltiniuose arba lokalūs sprendimai yra riboti, nepatogūs, pateikiamos ne visos ontologijų savybės.
3. Išanalizavus informacijos šaltiniuose siūlomus ontologijų vaizdavimo metodus, nustatyta, kad tinkamiausias metodas pateikti ontologijos duomenis semantiniame tinkle yra geometriniais modeliais.
4. Tinkamiausia semantinio tinklo technologija yra *Jena*, nes ši sistema yra populiariausia ir turi labiausiai ištobulintas ontologijos apdorojimo priemones.
5. Siekiamas sprendimas yra ontologijų vaizdavimo modelis ir jį realizuojantis prototipas, kuris tiktų semantinės analizės ir paieškos sistemoms, įskaitant ir lietuviškas sistemas.
6. Sukurtas prototipas leistų vartotojams, norintiems atlikti paiešką, peržiūrėti ontologiją ir geriau suprasti, kokią paiešką galima atlikti.
7. Prototipą bus siekiama sukurti universalų, kuris tiktų bet kokiai sričiai, o eksperimentas bus atliekamas su pasirinkta dalykine sritimi.

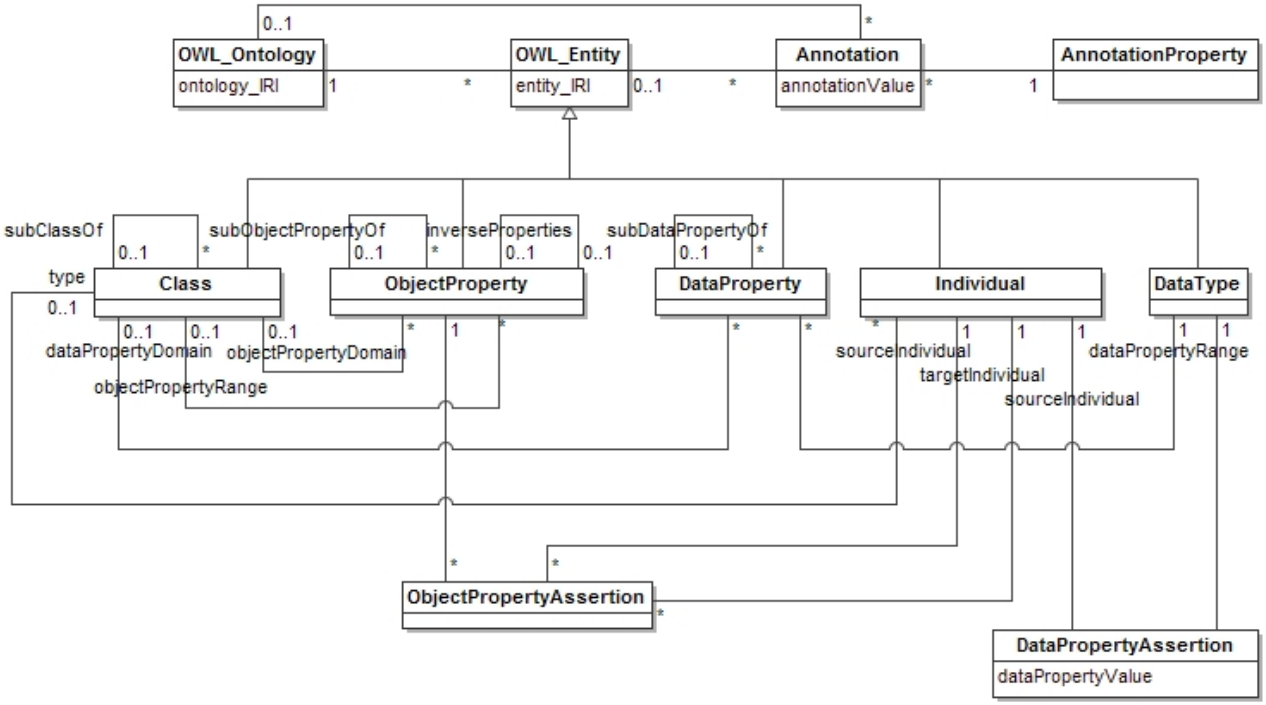
2. GRAFINIO ONTOLOGIJŲ VAIZDAVIMO ĮRANKIO MODELIS IR SPECIFIKACIJA

2.1. Formalus ontologijų vaizdinio pateikimo aprašas

Formalų ontologijų vaizdinio pateikimo aprašą sudaro ontologijos elementų modelis, vaizdinio šių elementų pateikimo modelis ir modelio peržiūros procesas.

2.1.1. Ontologijos elementų modelis

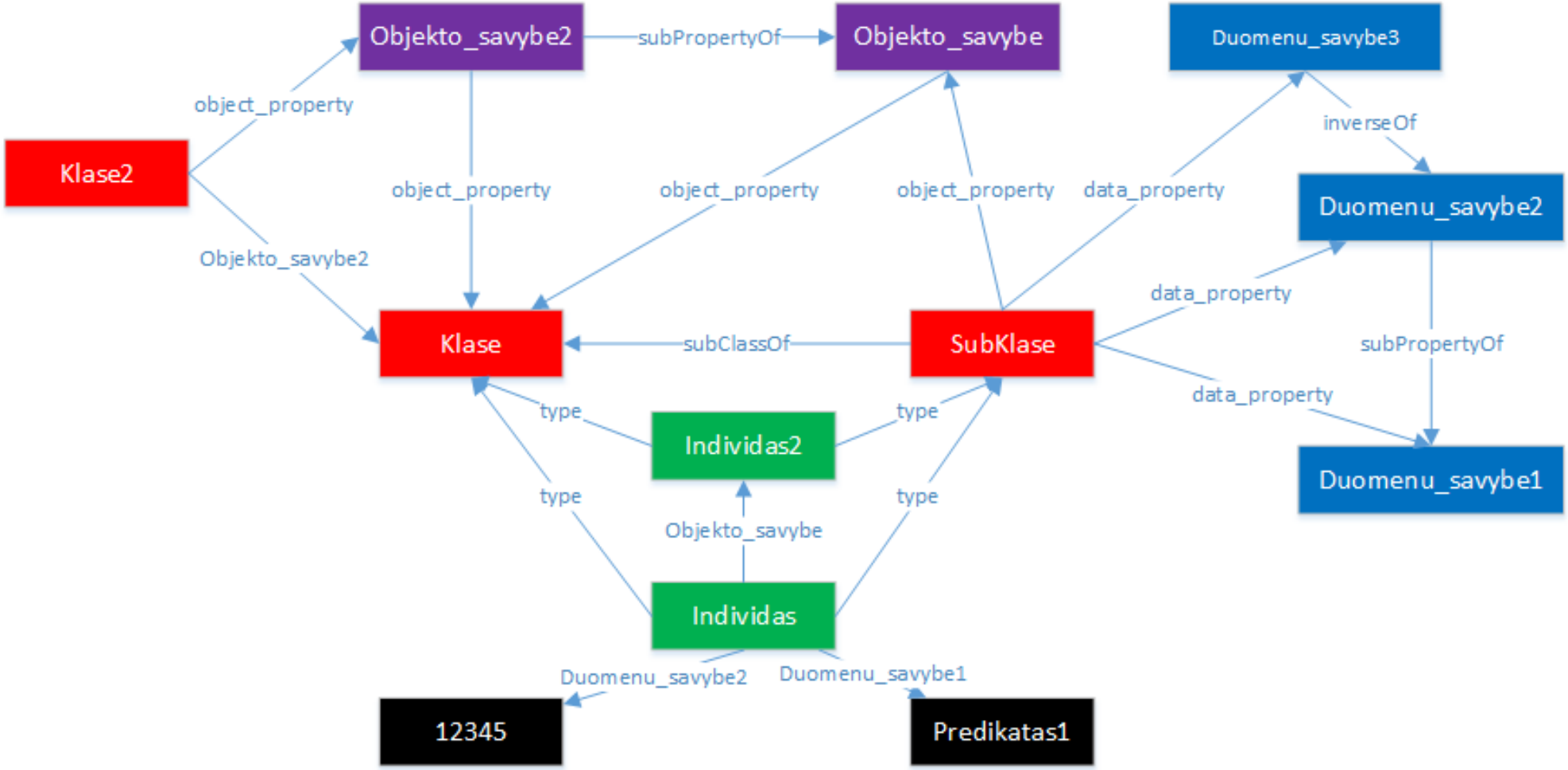
Ontologijos elementų ir jų ryšių modelis remiasi *OWL 2* ontologijos metamodeliu ir pateikiamas 2.1 paveiksle.



2.1 pav. Ontologijos elementų metamodelis (sudarytas remiantis [7])

2.1.2. Ontologijos elementų vaizdinio pateikimo modelis

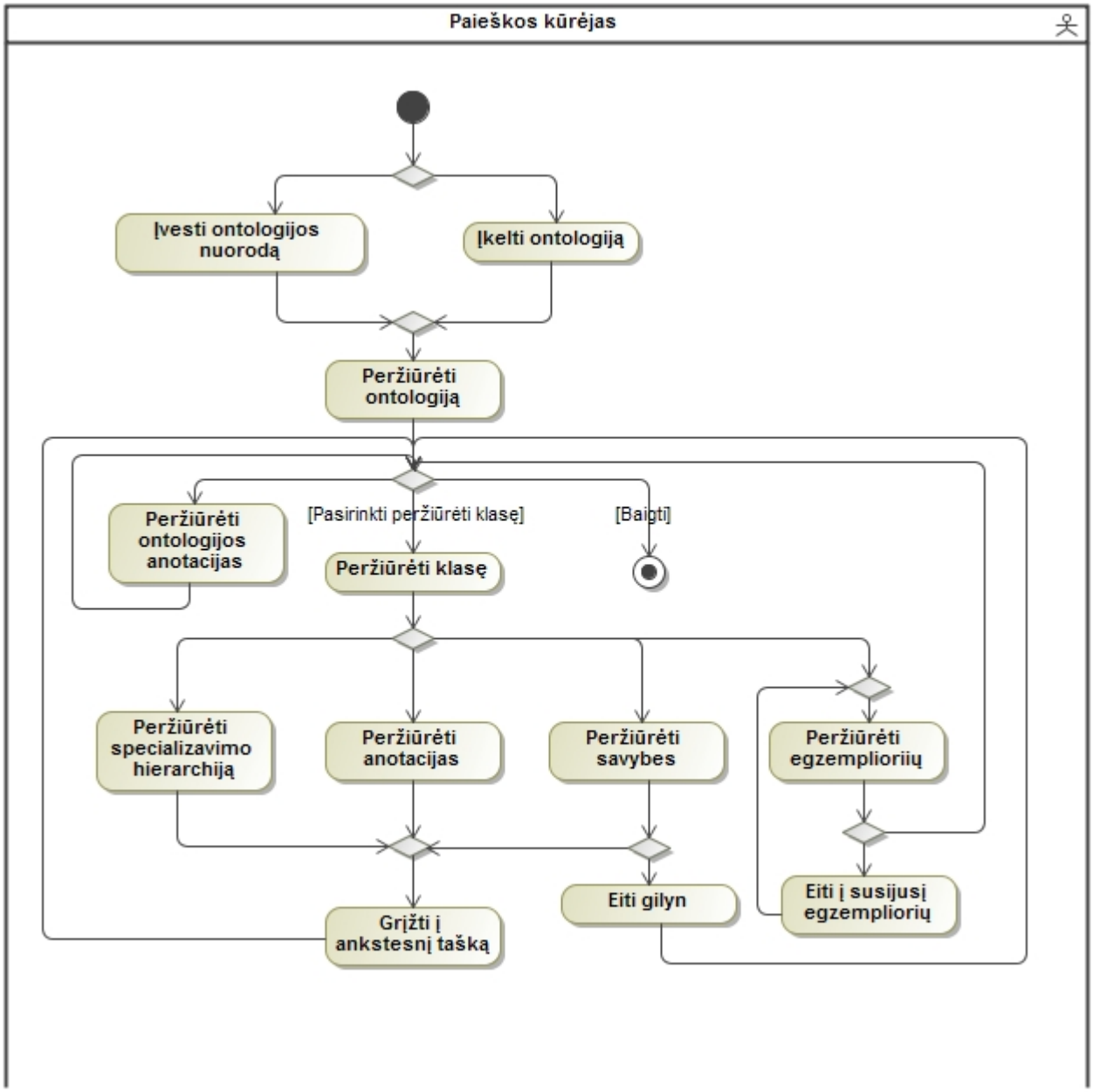
Ontologijos elementų ir jų ryšių vaizdinio pateikimo modelis, remiantis *OWL 2* ontologijos metamodeliu 2.1 paveiksle, pateikiamas 2.2 paveiksle.



2.2 pav. Ontologijos elementų vaizdinio pateikimo modelis

2.1.3. Ontologijų vaizdinio modelio peržiūros procesas

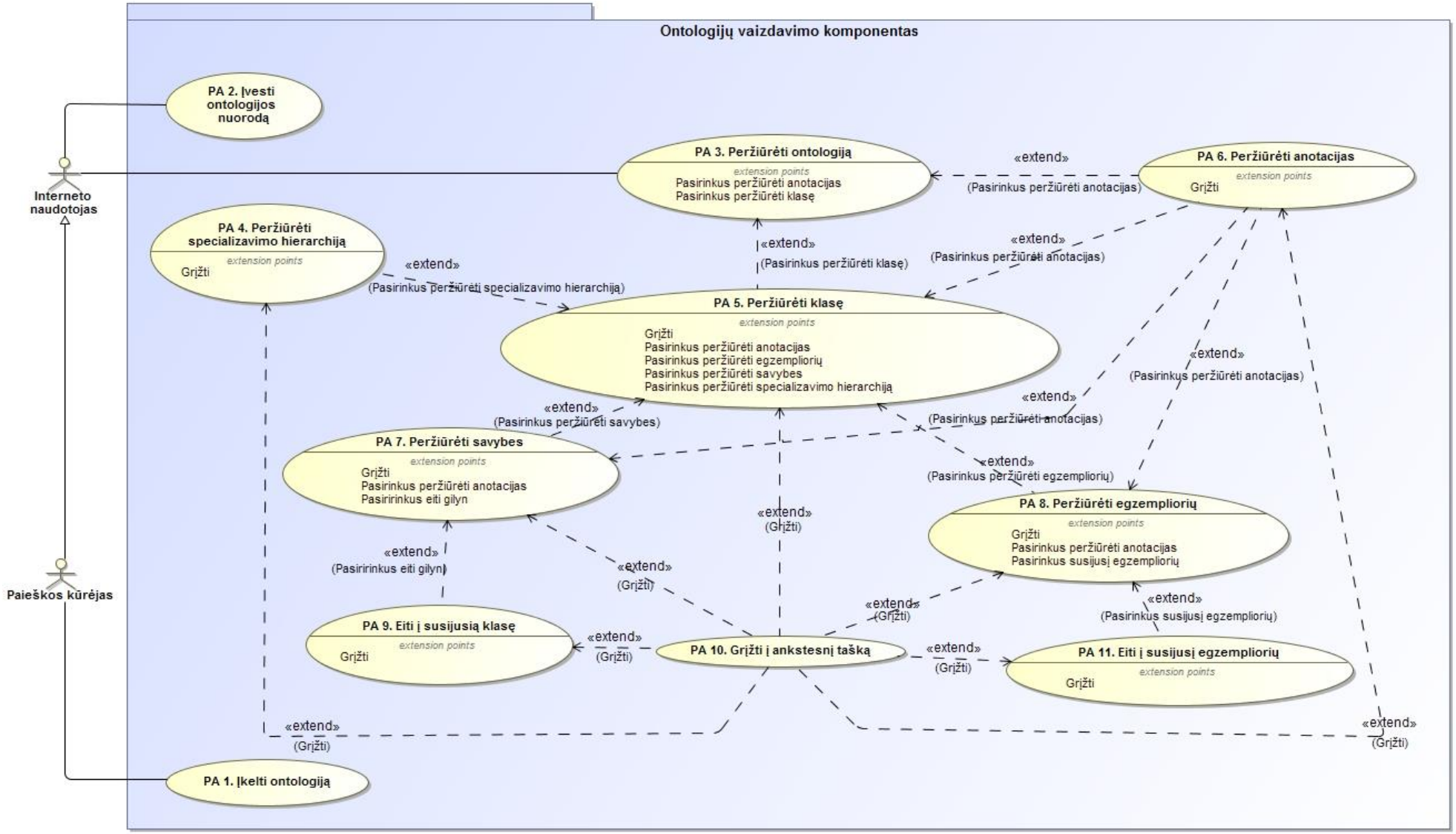
Ontologijos vaizdinio modelio peržiūros procesas apibūdina kokias funkcijas naudotojas gali atlikti projektuojamoje sistemoje. Vaizdinio modelio peržiūros procesas, pavaizduotas *UML* veiklos diagrama, pateiktas 2.3 paveiksle.



2.3 pav. Ontologijų vaizdinio modelio peržiūros procesas, pavaizduotas *UML* veiklos diagrama

2.2. Grafinio ontologijų vaizdavimo įrankio reikalavimų specifikacija

2.2.1. Panaudojimo atvejų diagrama



2.4 pav. Grafinio ontologijų vaizdavimo įrankio panaudojimo atvejų modelis

2.2.2. Panaudojimo atvejų specifikacijos

Grafinio ontologijų vaizdinio įrankio panaudojimo atvejams, kurie pateikiami 2.1 paveiksle, aprašytos panaudojimo atvejų specifikacijos, pateikiamos 2.1 – 2.11 lentelėse.

2.1 lentelė. Panaudojimo atvejo „Įkelti ontologiją“ specifikacija

PA 1. Įkelti ontologiją į serverį		
Aktorius		Paieškos kūrėjas
Prieš sąlygą		
Sužadinimo sąlyga		Naudotojas atidaro tinklapį
Susiję panaudojimo atvejai	**Išplečia PA**	-
	Apima PA	-
	Specializuoja PA	-
Pagrindinis įvykių srautas		**Sistemos reakcija ir sprendimai**
1. Pažymimas laukas ontologijos įkėlimui		Atidaromas pasirinkimo langas, kad įkelti ontologijos failą
2. Įkeliama ontologija		Į sistemą užkraunama ontologija
Po sąlyga:		Atidaromas ontologijos grafinis modelis

2.2 lentelė. Panaudojimo atvejo „Įvesti ontologijos nuorodą“ specifikacija

PA 2. Pasirinkti ontologiją		
Aktorius		Interneto naudotojas
Prieš sąlygą		
Sužadinimo sąlyga		Naudotojas atidaro tinklapį
Susiję panaudojimo atvejai	**Išplečia PA**	-
	Apima PA	-
	Specializuoja PA	-
Pagrindinis įvykių srautas		**Sistemos reakcija ir sprendimai**
1. Įvedama ontologijos nuoroda		Į sistemą užkraunama ontologija
Po sąlyga:		Atidaromas ontologijos grafinis modelis

2.3 lentelė. Panaudojimo atvejo „Peržiūrėti ontologiją“ specifikacija

PA 3. Peržiūrėti ontologiją		
Aktorius		Interneto naudotojas
Prieš sąlygą		Sistemoje turi būti įkelta ontologija
Sužadinimo sąlyga		Įkeliama nauja arba pasirenkama jau įkelta ontologija
Susiję panaudojimo atvejai	**Išplečia PA**	-
	Apima PA	-
	Specializuoja PA	-
Pagrindinis įvykių srautas		**Sistemos reakcija ir sprendimai**
1. Pažymimas laukas ontologijos įkėlimui		Atidaromas pasirinkimo langas, kad įkelti ontologijos failą
2. Užkraunamas ontologijos failas		Į sistemą užkraunama ontologija
Po sąlyga:		Pateikiamas vaizdinis modelis

2.4 lentelė. Panaudojimo atvejo „Peržiūrėti specializavimo hierarchiją“ specifikacija

PA 4. Peržiūrėti specializavimo hierarchiją

Aktorius		Interneto naudotojas
Prieš sąlygą		Sistemoje turi būti įkelta ontologija
Sužadinimo sąlyga		Peržiūrima įkeltos ontologijos klasė
Susiję panaudojimo atvejai	**Išplečia PA**	PA 5. Peržiūrėti klasę
	Apima PA	-
	Specializuoja PA	-
Pagrindinis įvykių srautas		**Sistemos reakcija ir sprendimai**
1. Ontologijos vaizdiniame modelyje pasirenkama peržiūrėti klasę		Sugeneruojama ir pateikiama specializuota hierarchija
Po sąlyga:		Pateikiama specializuota hierarchija

2.5 lentelė. Panaudojimo atvejo „Peržiūrėti klasę“ specifikacija

PA 5. Peržiūrėti klasę		
Aktorius		Interneto naudotojas
Prieš sąlygą		Sistemoje turi būti įkelta ontologija
Sužadinimo sąlyga		Naudotojas įkelia arba pasirenka peržiūrai ontologiją
Susiję panaudojimo atvejai	**Išplečia PA**	PA 3. Peržiūrėti ontologiją
	Apima PA	-
	Specializuoja PA	-
Pagrindinis įvykių srautas		**Sistemos reakcija ir sprendimai**
1. Ontologijos vaizdiniame modelyje pasirenkama peržiūrėti klasę		Sugeneruojamas ir pateikiamas vaizdinis modelis
Po sąlyga:		Pateikiamas vaizdinis modelis

2.6 lentelė. Panaudojimo atvejo „Peržiūrėti anotacijas“ specifikacija

PA 6. Peržiūrėti anotacijas		
Aktorius		Interneto naudotojas
Prieš sąlygą		Sistemoje turi būti įkelta ontologija
Sužadinimo sąlyga		Naudotojas atidaro tinklapį
Susiję panaudojimo atvejai	**Išplečia PA**	PA 3. Peržiūrėti ontologiją PA 5. Peržiūrėti klasę PA 7. Peržiūrėti savybes PA 8. Peržiūrėti egzempliorių
	Apima PA	-
	Specializuoja PA	-
Pagrindinis įvykių srautas		**Sistemos reakcija ir sprendimai**
1. Ontologijos vaizdiniame modelyje pasirenkama anotacija		Sugeneruojamas ir pateikiamas vaizdinis modelis
Po sąlyga:		Pateikiamas vaizdinis modelis

2.7 lentelė. Panaudojimo atvejo „Peržiūrėti savybes“ specifikacija

PA 7. Peržiūrėti savybes		
Aktorius		Interneto naudotojas
Prieš sąlygą		Sistemoje turi būti įkelta ontologija
Sužadinimo sąlyga		Naudotojas atidaro tinklapį
	Išplečia PA	PA 5. Peržiūrėti klasę

Susiję panaudojimo atvejai	**Apima PA**	-
	Specializuoja PA	-
Pagrindinis įvykių srautas		**Sistemos reakcija ir sprendimai**
1. Ontologijos vaizdiniame modelyje pasirenkama peržiūrėti klasės savybes		Sugeneruojamas ir pateikiamas savybių sąrašas
Po sąlyga:		Pateikiamas vaizdinis modelis

2.8 lentelė. Panaudojimo atvejo „Peržiūrėti egzempliorių“ specifikacija

PA 8. Peržiūrėti egzempliorių		
Aktorius		Interneto naudotojas
Prieš sąlygą		Sistemoje turi būti įkelta ontologija
Sužadinimo sąlyga		Naudotojas atidaro tinklapį
Susiję panaudojimo atvejai	**Išplečia PA**	PA 5. Peržiūrėti klasę
	Apima PA	-
	Specializuoja PA	-
Pagrindinis įvykių srautas		**Sistemos reakcija ir sprendimai**
1. Ontologijos vaizdiniame modelyje pasirenkama peržiūrėti egzempliorių		Sugeneruojamas ir pateikiamas vaizdinis modelis
Po sąlyga:		Pateikiamas vaizdinis modelis

2.9 lentelė. Panaudojimo atvejo „Eiti į susijusią klasę“ specifikacija

PA 9. Eiti į susijusią klasę		
Aktorius		Interneto naudotojas
Prieš sąlygą		Sistemoje turi būti įkelta ontologija
Sužadinimo sąlyga		Naudotojas atidaro tinklapį
Susiję panaudojimo atvejai	**Išplečia PA**	PA 7. Peržiūrėti savybes
	Apima PA	-
	Specializuoja PA	-
Pagrindinis įvykių srautas		**Sistemos reakcija ir sprendimai**
1. Ontologijos vaizdiniame modelyje pasirenkama iš savybės eiti į susijusią klasę		Sugeneruojamas ir pateikiamas vaizdinis modelis
Po sąlyga:		Pateikiamas vaizdinis modelis

2.10 lentelė. Panaudojimo atvejo „Grįžti į ankstesnį tašką“ specifikacija

PA 10. Grįžti į ankstesnį tašką		
Aktorius		Interneto naudotojas
Prieš sąlygą		Sistemoje turi būti įkelta ontologija
Sužadinimo sąlyga		Naudotojas atidaro tinklapį
Susiję panaudojimo atvejai	**Išplečia PA**	PA 4. Peržiūrėti specializavimo hierarchiją PA 5. Peržiūrėti klasę PA 6. Peržiūrėti anotacijas PA 7. Peržiūrėti savybes PA 8. Peržiūrėti egzempliorių PA 9. Eiti į susijusią klasę PA 11. Eiti į susijusį egzempliorių
	Apima PA	-

	Specializuoja PA	-
Pagrindinis įvykių srautas		**Sistemos reakcija ir sprendimai**
1. Ontologijos vaizdiniame modelyje iš peržiūrėtų klasių pasirenkama grįžti atgal.		Sugeneruojamas ir pateikiamas vaizdinis modelis
Po sąlyga:		Pateikiamas vaizdinis modelis
Alternatyvūs scenarijai		
1.a Ontologijos vaizdiniame modelyje iš peržiūrėtų anotacijų pasirenkama grįžti atgal.		Sugeneruojamas ir pateikiamas vaizdinis modelis
1.b Ontologijos vaizdiniame modelyje iš peržiūrėtų savybių pasirenkama grįžti atgal.		Sugeneruojamas ir pateikiamas vaizdinis modelis
1.c Ontologijos vaizdiniame modelyje iš peržiūrėtų egzempliorių pasirenkama grįžti atgal.		Sugeneruojamas ir pateikiamas vaizdinis modelis

2.11 lentelė. Panaudojimo atvejo „Eiti į susijusį egzempliorių“ specifikacija

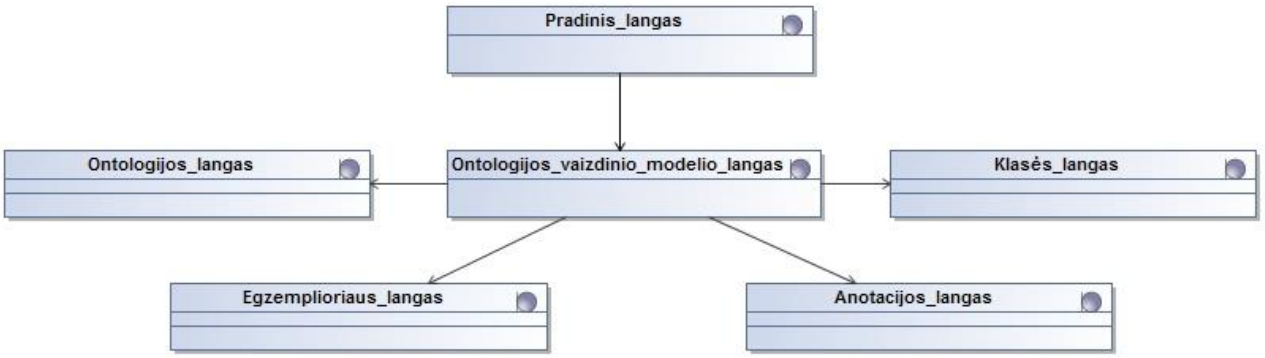
PA 11. Eiti į susijusį egzempliorių		
Aktorius		Interneto naudotojas
Prieš sąlygą		Sistemoje turi būti įkelta ontologija
Sužadinimo sąlyga		Naudotojas atidaro tinklapį
Susiję panaudojimo atvejai	**Išplečia PA**	PA 8. Peržiūrėti egzempliorių
	Apima PA	-
	Specializuoja PA	-
Pagrindinis įvykių srautas		**Sistemos reakcija ir sprendimai**
1. Ontologijos vaizdiniame modelyje pasirenkama iš egzemplioriaus eiti į susijusį egzempliorių		Sugeneruojamas ir pateikiamas vaizdinis modelis
Po sąlyga:		Pateikiamas vaizdinis modelis

2.3. Vartotojo sąsajos modelis

Vaizdinio ontologijų pateikimo modelis projektuojamas atsižvelgiant į vartotojo sąsajos modelius ir navigavimo planą.

2.3.1. Navigavimo planas

Ontologijos vaizdinio pateikimo modelio vartotojo sąsajos navigavimo planas pateikiamas 2.5 paveiksle.



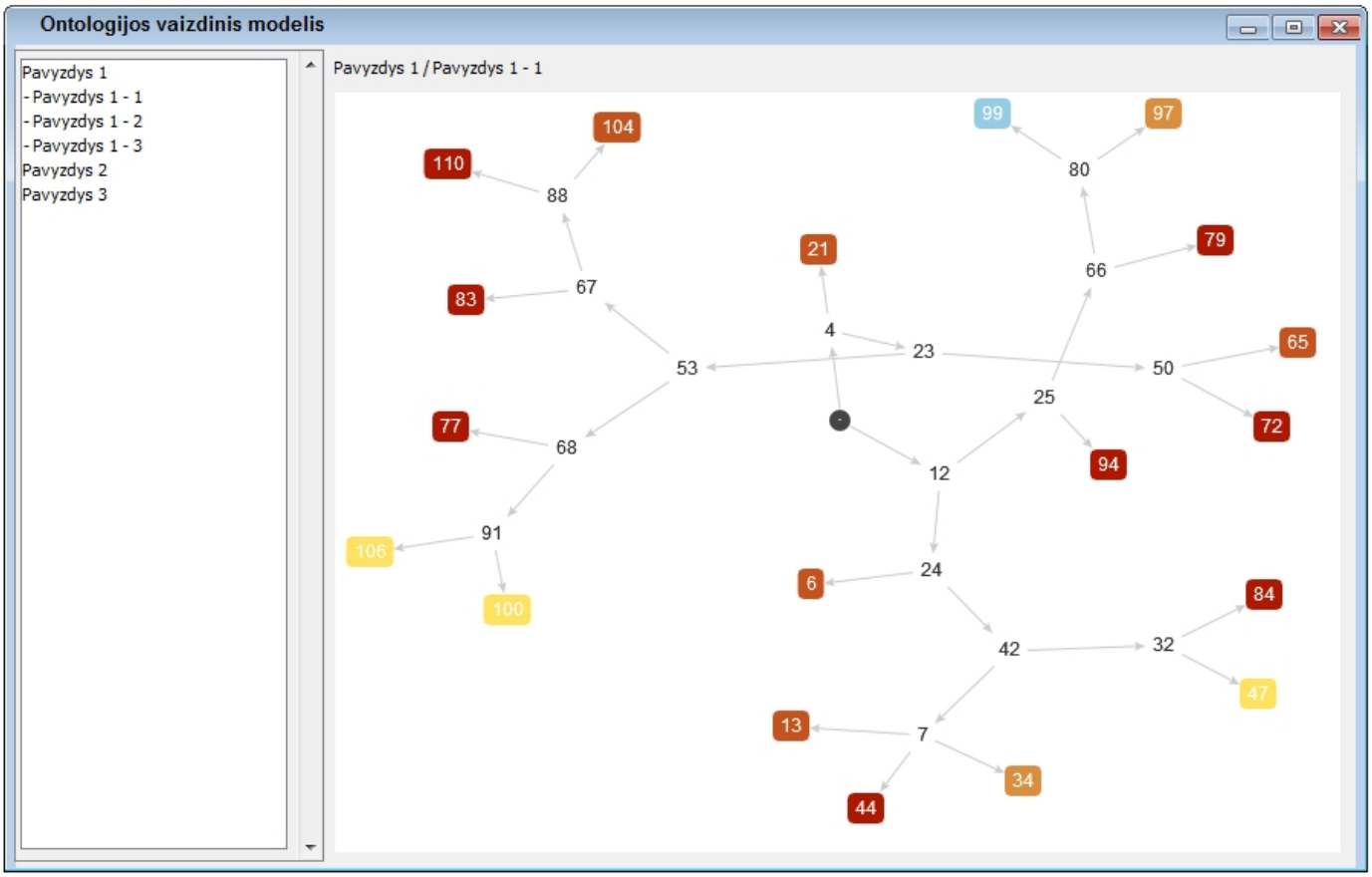
2.5 pav. Vartotojo sąsajos navigavimo planas

2.3.2. Vartotojo sąsaja

Ontologijos vaizdinio pateikimo modelio vartotojo sąsajos pradinio lango projektas, skirtas įvesti ontologijos nuorodai iš nutolusio serverio arba įkelti ontologiją, pateikiamas 2.6 paveiksle.

2.6 pav. Pradinis langas

Ontologijos vaizdinio pateikimo modelio lango projektas, kuriame pateikiamas ontologijos vaizdinis modelis ir hierarchinė struktūra, pateikiamas 2.7 paveiksle.



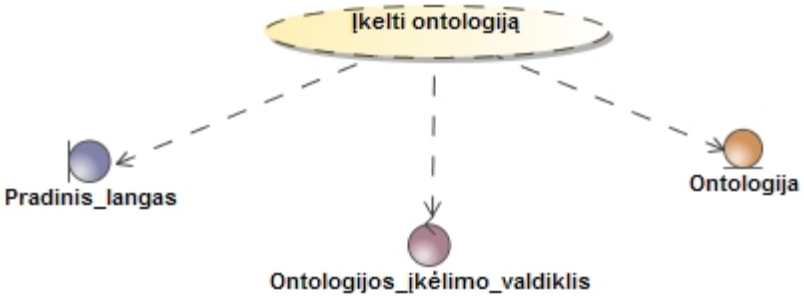
2.7 pav. Ontologijos vaizdavimo langas

3. SISTEMOS PROJEKTAS

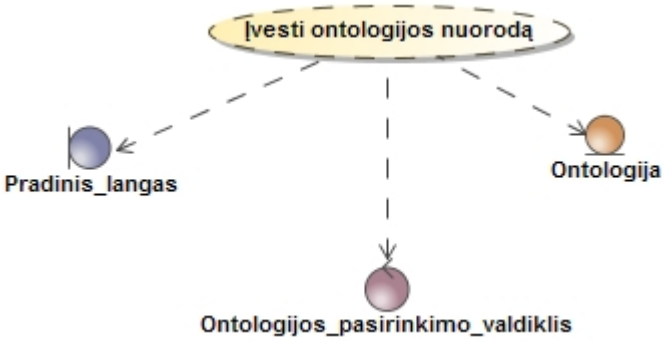
3.1. Sistemos architektūra

3.1.1. Reikalavimų analizė

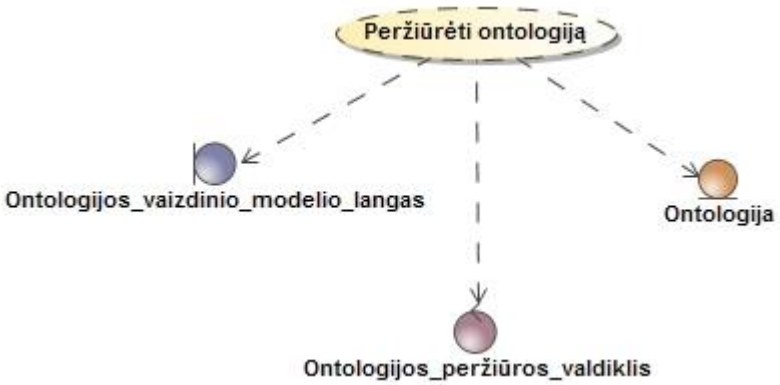
Sistemos projektui, kiekvienam panaudojimo atvejui sudaromos analizės klasių diagramos, pateiktos 3.1 – 3.11 paveiksluose.



3.1 pav. PA 1. „Įkelti ar ištrinti ontologiją“ analizės klasės

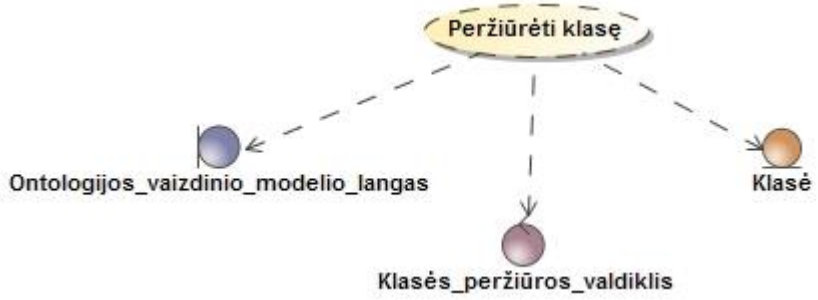


3.2 pav. PA 2. „Įvesti ontologijos nuorodą“ analizės klasės

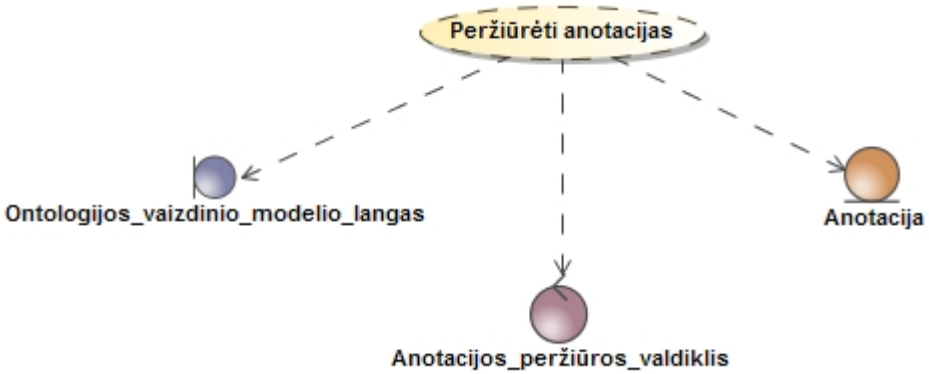


3.3 pav. PA 3. „Peržiūrėti ontologiją“ analizės klasės

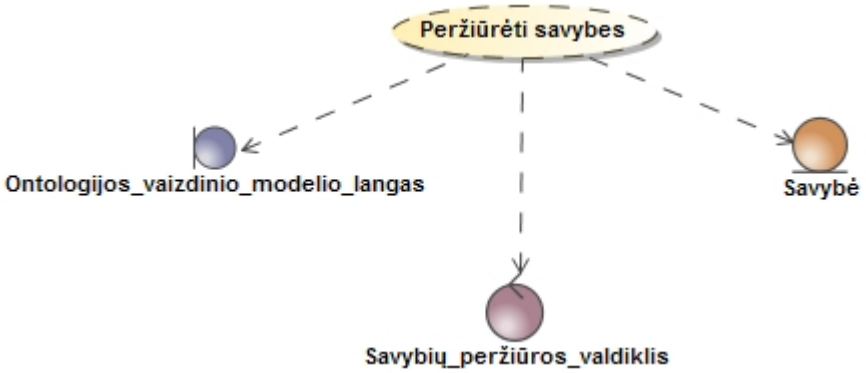
3.4 pav. PA 4. „Peržiūrėti specializavimo hierarchiją“ analizės klasės



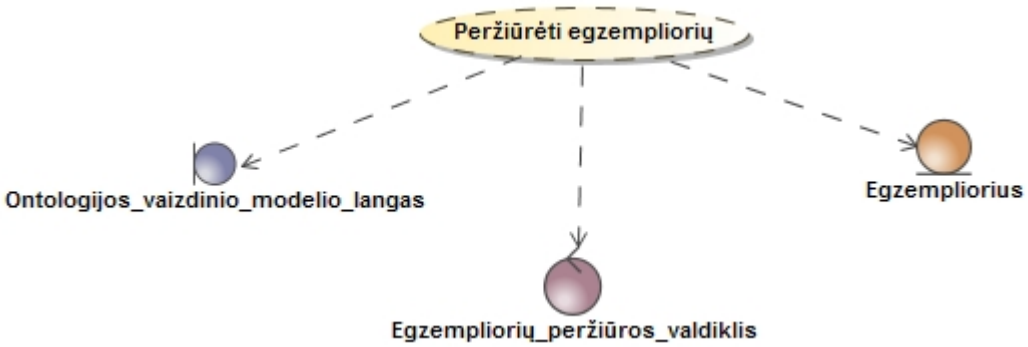
3.5 pav. PA 5. „Peržiūrėti klasę“ analizės klasės



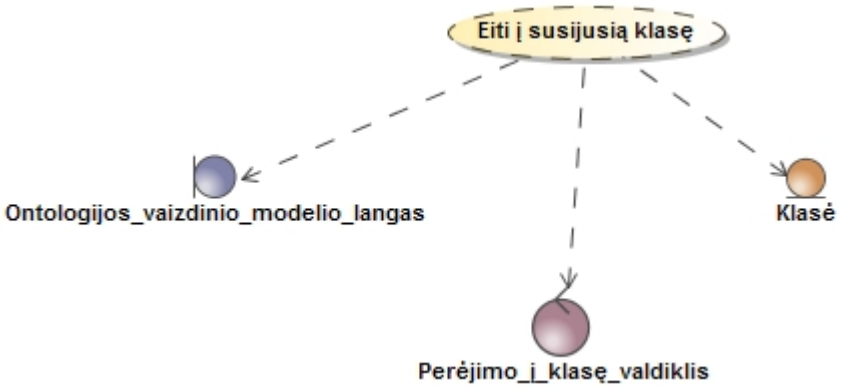
3.6 pav. PA 6. „Peržiūrėti anotacijas“ analizės klasės



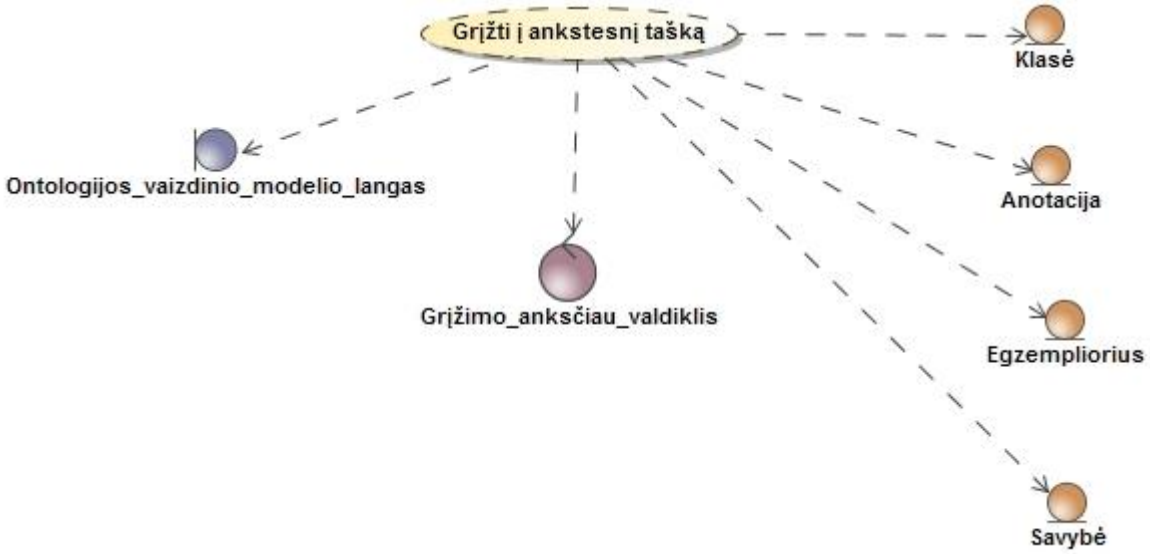
3.7 pav. PA 7. „Peržiūrėti savybes“ analizės klasės



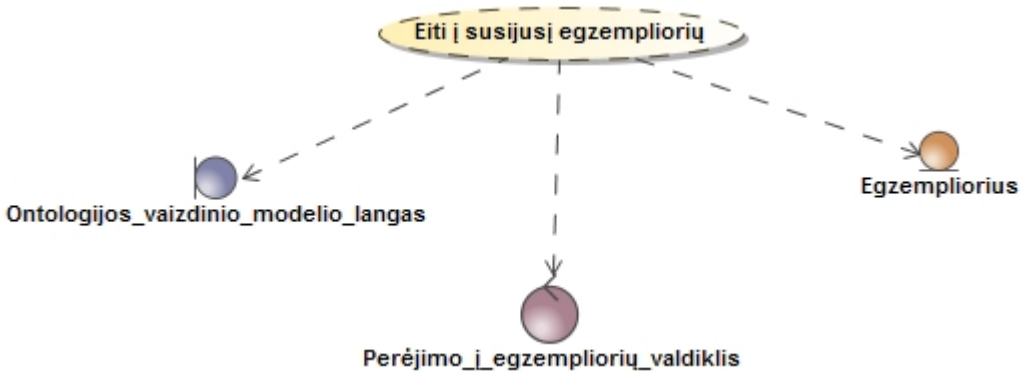
3.8 pav. PA 8. „Peržiūrėti egzempliorių“ analizės klasės



3.9 pav. PA 9. „Eiti į susijusią klasę“ analizės klasės



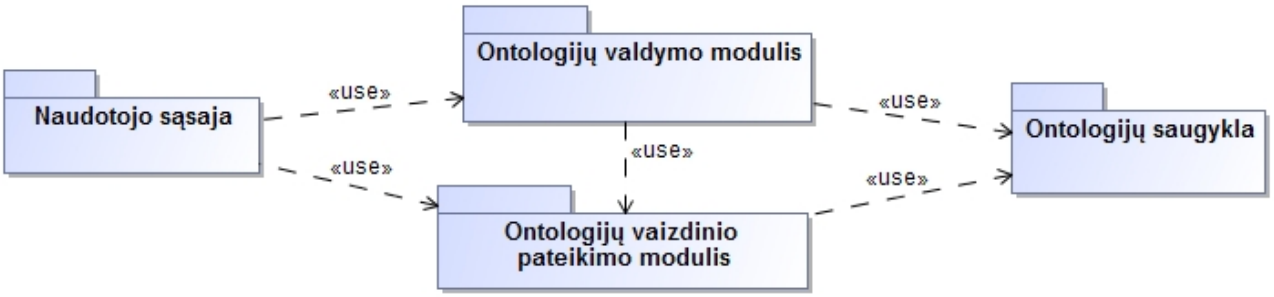
3.10 pav. PA 10. „Grįžti į ankstesnį žingsnį“ analizės klasės



3.11 pav. PA 11 „Eiti į susijusį egzempliorių“ analizės klasės

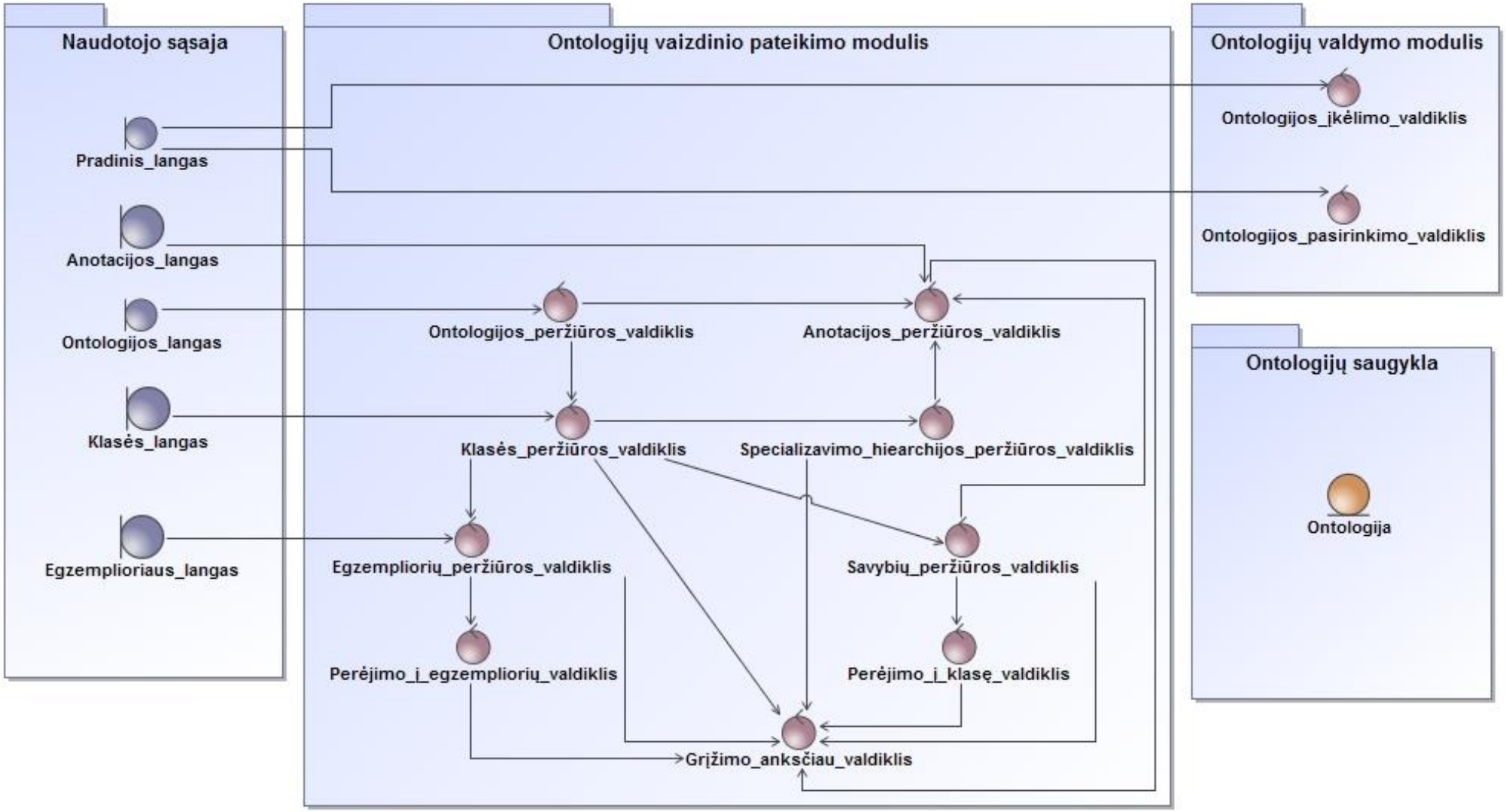
3.1.2. Loginė visos sistemos architektūra

Ontologijos vaizdinio pateikimo modelio loginei architektūrai atvaizduoti sumodeliuota visos sistemos loginė architektūra pateikta 3.12 paveiksle.



3.12 pav. Loginė visos sistemos architektūra

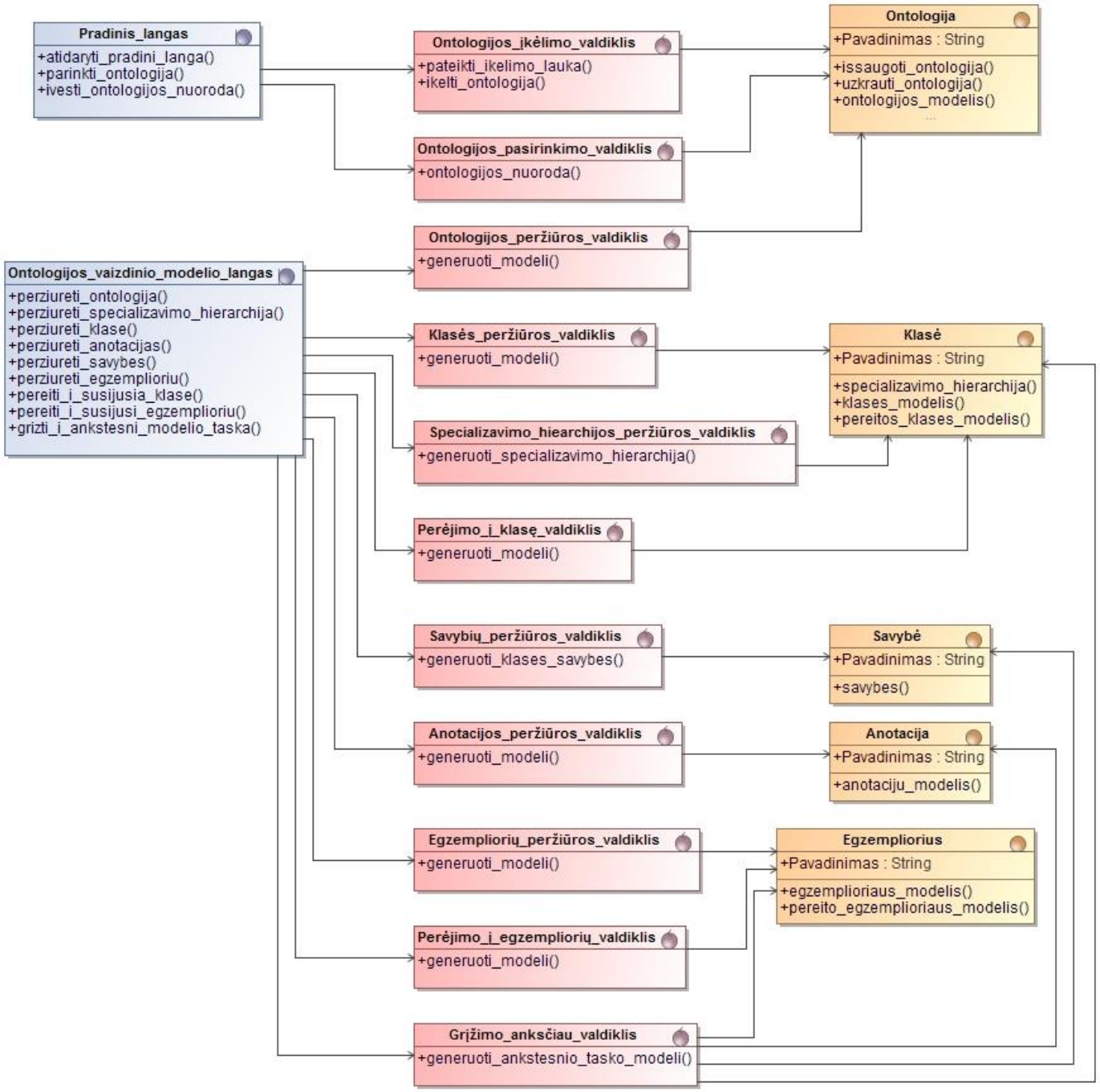
Detalizuota sistemos loginė architektūra pateikta 3.13 paveiksle.



3.13 pav. Detalizuota sistemos loginė architektūra

3.1.3. Vartotojo sąsajos klasių modelis

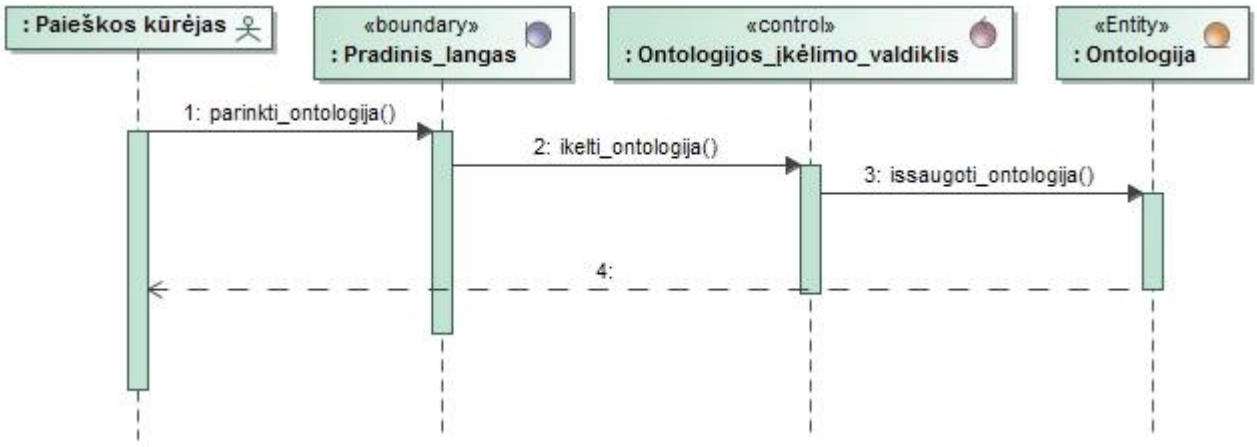
Atsižvelgiant į sistemos projektavimo loginę architektūrą, vartotojo sąsajai sudaroma projektavimo klasių diagrama pateikta 3.14 paveiksle.



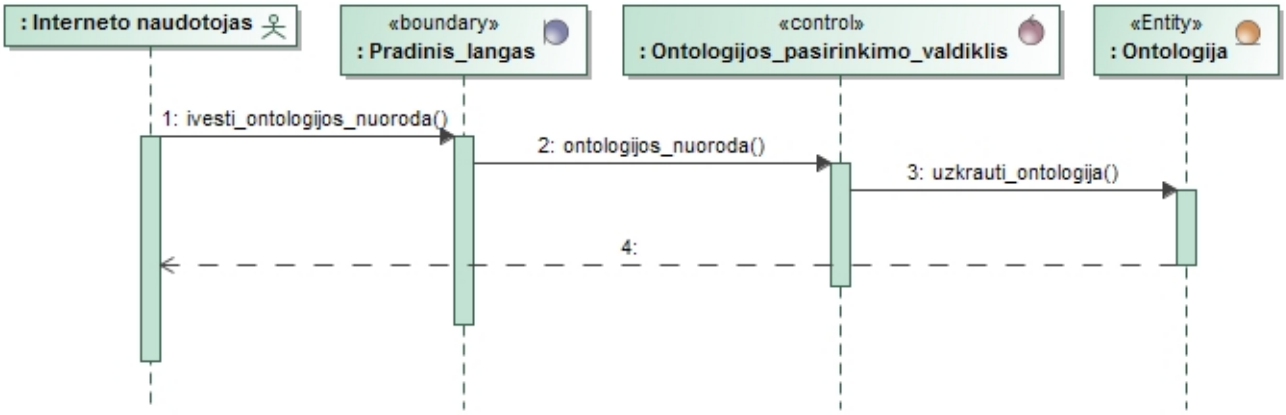
3.14 pav. Ontologijų vaizdinio pateikimo įrankio klasių modelis

3.2. Sistemos elgsenos modelis

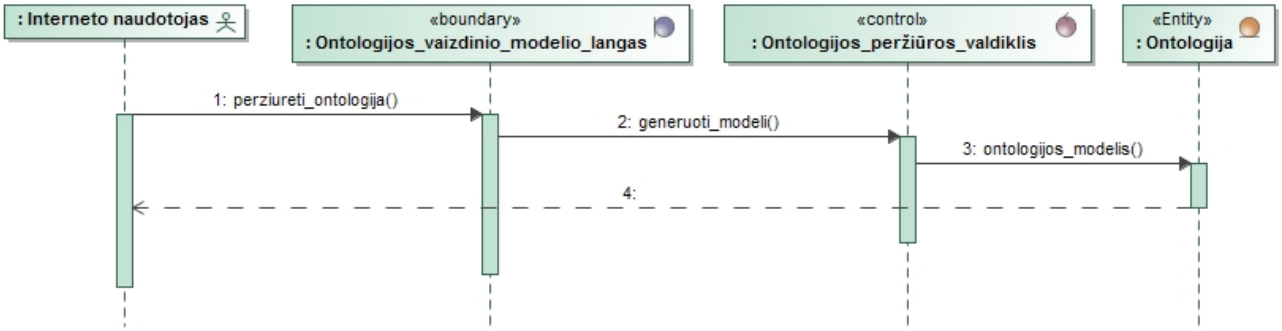
Panaudojimo atvejams sudaromos sekų diagramos, pateiktos 3.15 – 3.25 paveiksluose.



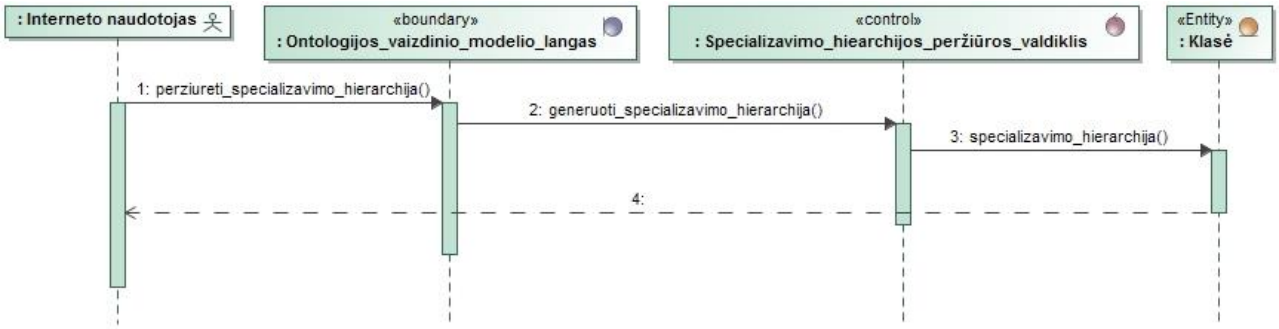
3.15 pav. PA 1. „Įkelti ontologiją“ sekų diagrama



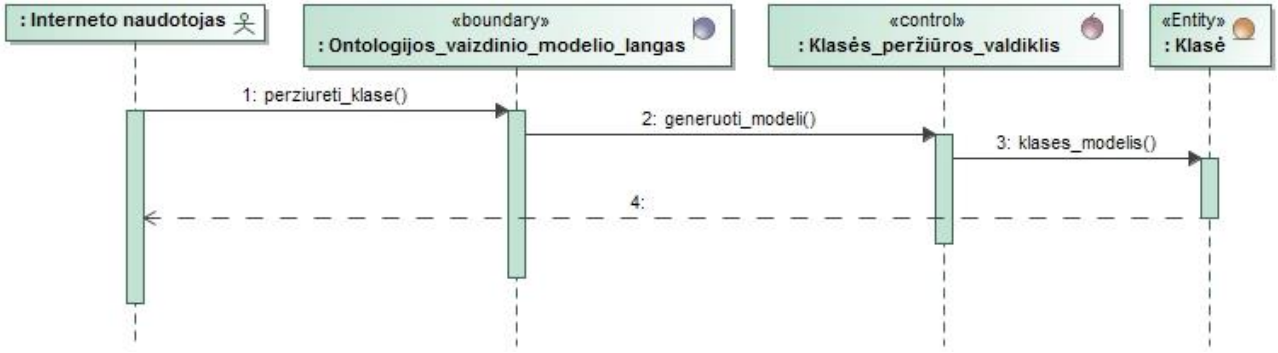
3.16 pav. PA 2. „Įvesti ontologijos nuorodą“ sekų diagrama



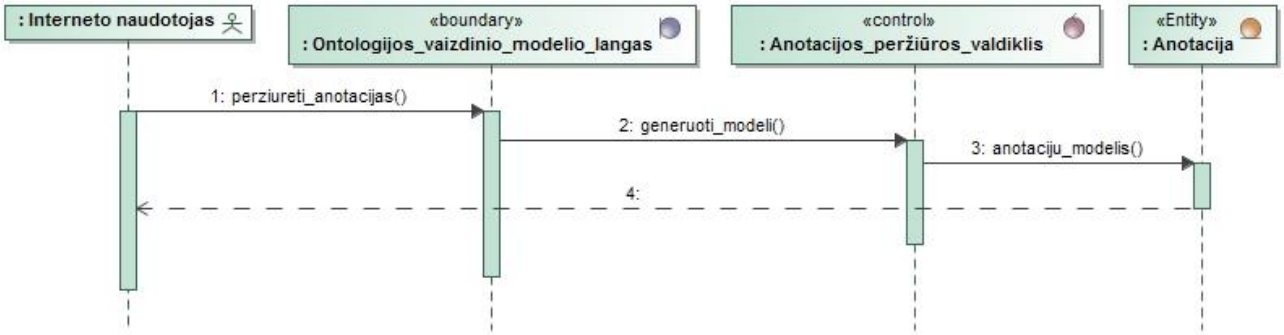
3.17 pav. PA 3. „Peržiūrėti ontologiją“ sekų diagrama



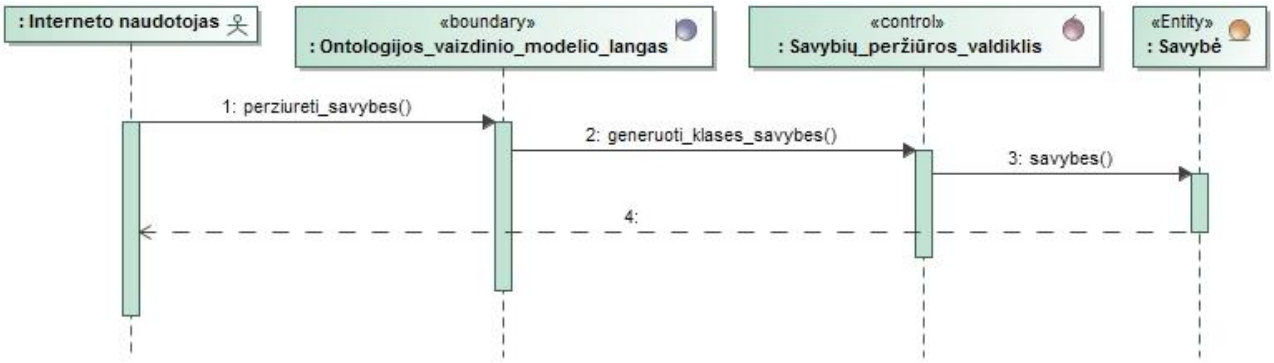
3.18 pav. PA 4. „Peržiūrėti specializavimo hierarchiją“ sekų diagrama



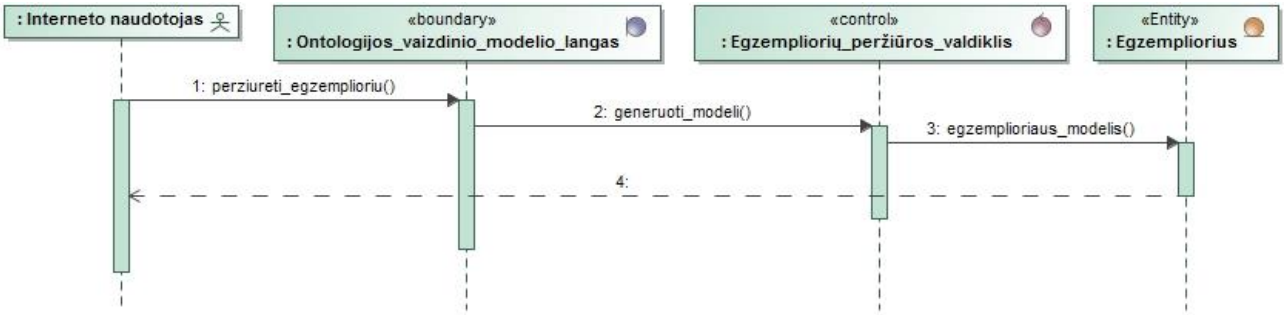
3.19 pav. PA 5. „Peržiūrėti klasę“ sekų diagrama



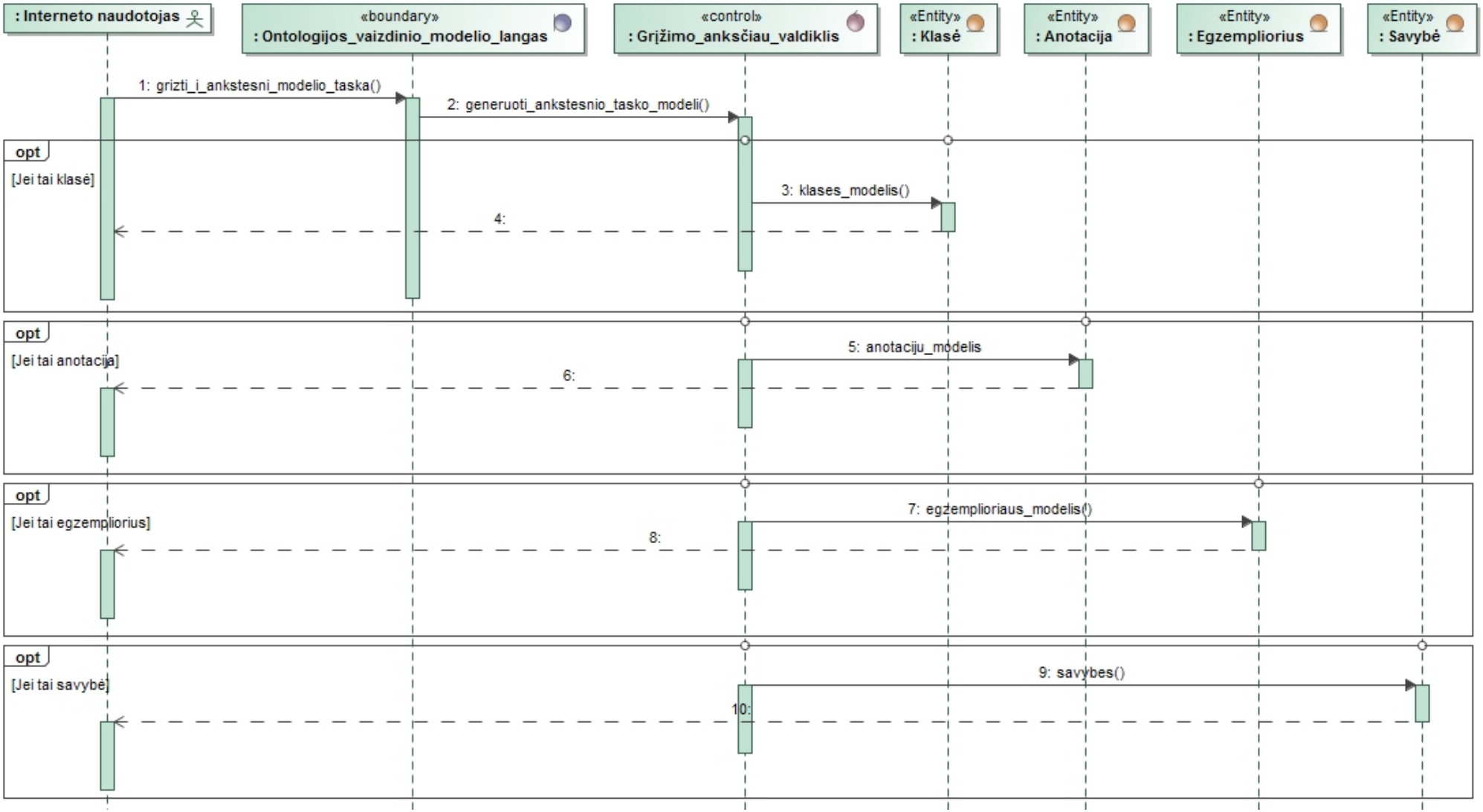
3.20 pav. PA 6. „Peržiūrėti anotacijas“ sekų diagrama



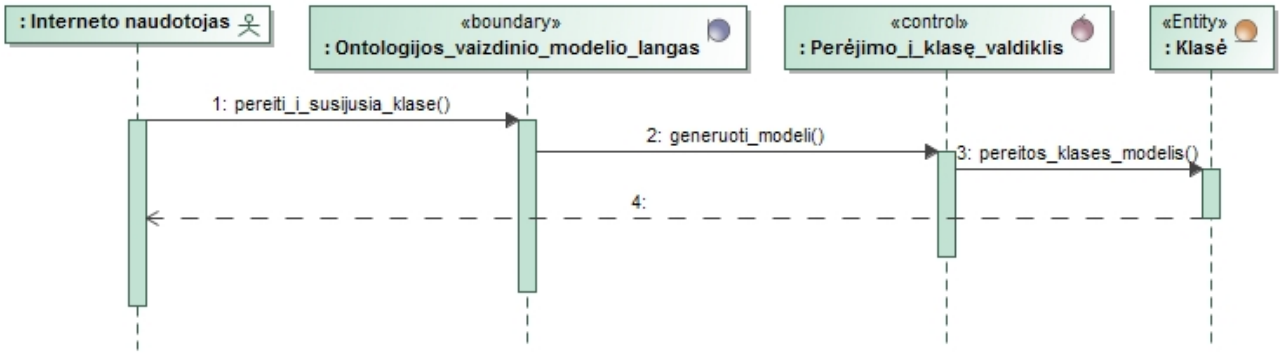
3.21 pav. PA 7. „Peržiūrėti savybes“ sekų diagrama



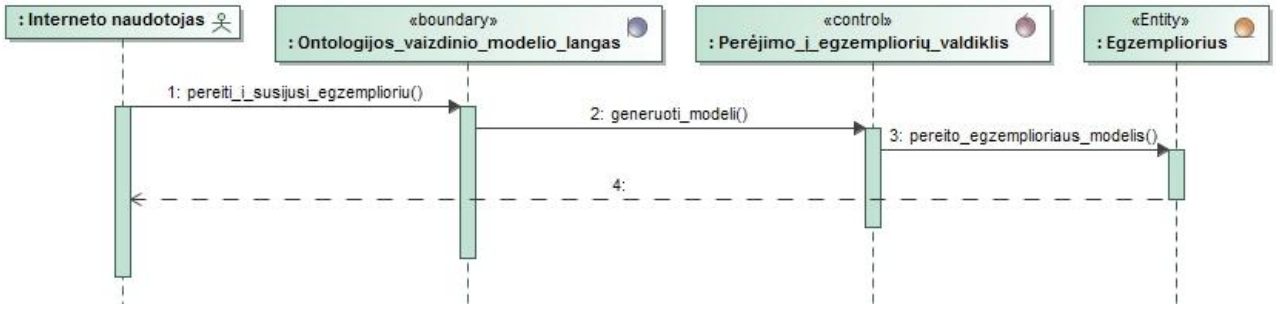
3.22 pav. PA 8. „Peržiūrėti egzempliorių“ sekų diagrama



3.23 pav. PA 10. „Grįžti į ankstesnį žingsnį“ sekų diagrama



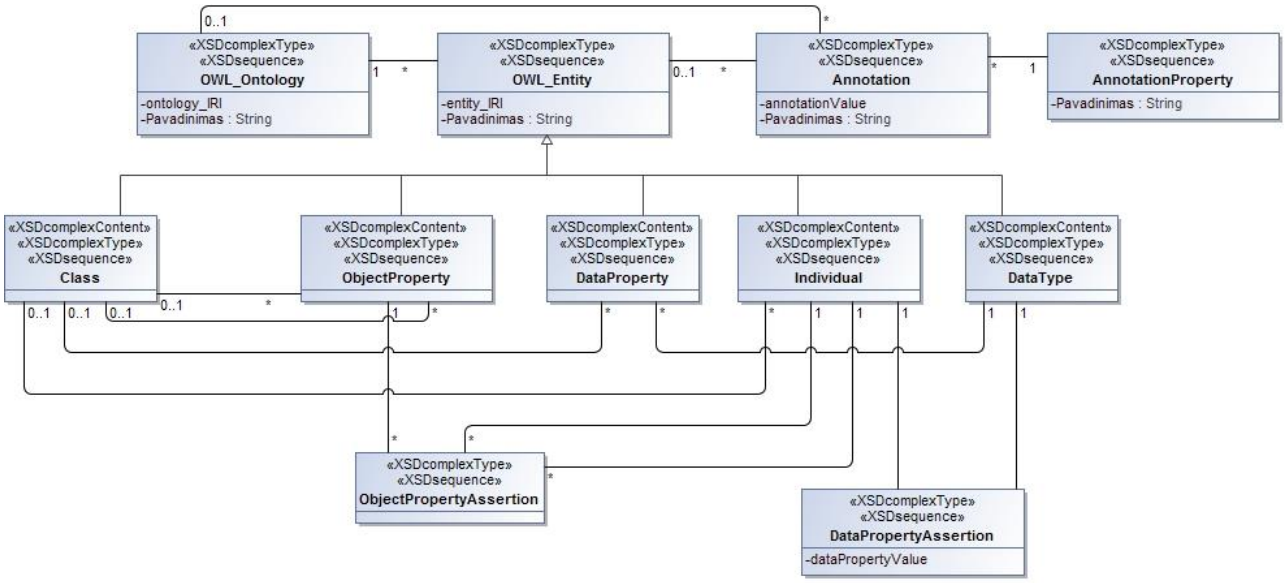
3.24 pav. PA 9. „Eiti į susijusią klasę“ sekų diagrama



3.25 pav. PA 11. „Eiti į susijusį egzempliorių“ sekų diagrama

3.3. Loginė ontologijos duomenų schema

Sistemos realizacijai duomenų bazė nebus naudojama ir duomenys bus panaudojami tiesiogiai iš ontologijos failo. Loginė ontologijos duomenų schema pateikta 3.26 paveiksle.

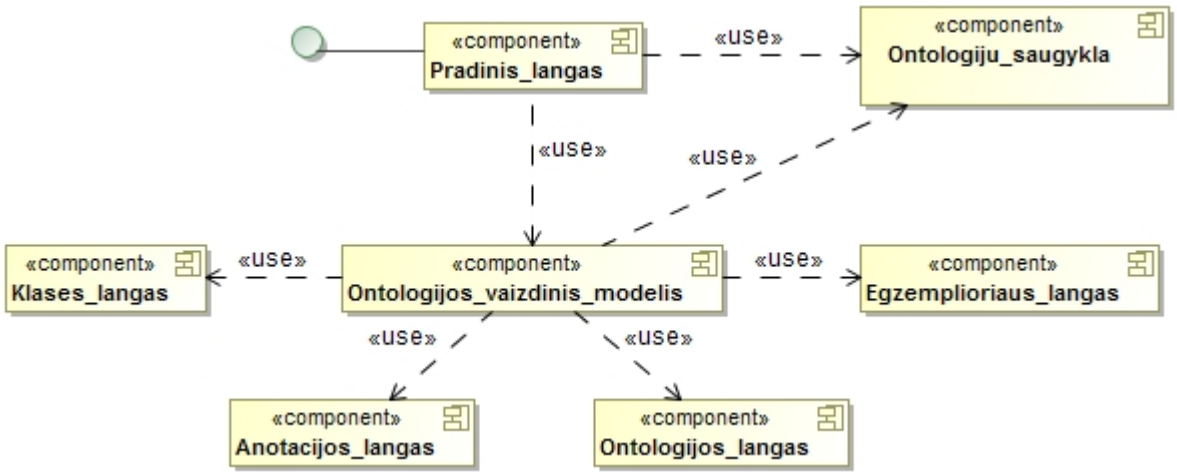


3.26 pav. Loginė ontologijos duomenų schema

3.4. Realizacijos modelis

3.4.1. Programinių komponentų architektūra

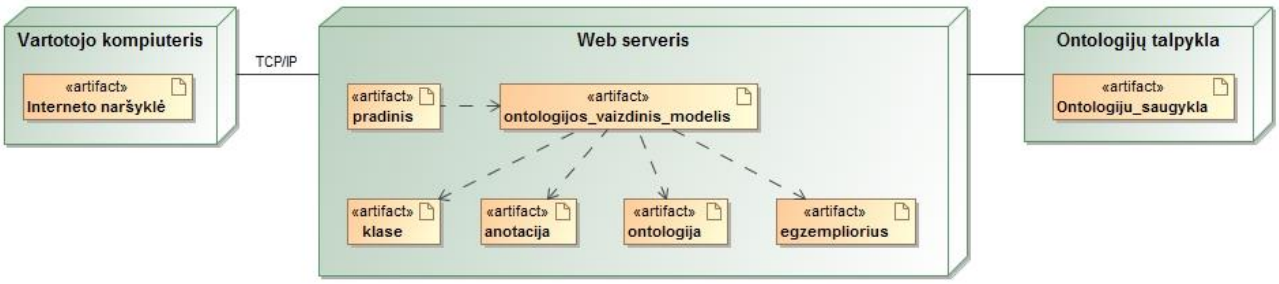
Sistemos programinių komponentų architektūra pateikta 3.27 paveiksle.



3.27 pav. Programinių komponentų architektūra

3.4.2. Diegimo modelis

Sistemos diegimo modelis pateiktas 3.28 paveiksle.



3.28 pav. Diegimo modelis

4. SPRENDIMO REALIZACIJA IR TESTAVIMAS

Tam, kad pademonstruoti ontologijų vaizdinio pateikimo modelio veikimą, šis modelis realizuotas Java programavimo kalba, panaudojant *Jena* sistemą ir *Arbor* biblioteką. Realizuotas ontologijų vaizdinio pateikimo modelis turi veikti semantinio tinklo komponento principu tam, kad suformuotą modelį būtų galima naudoti įvairiose išorinėse sistemose ar programinėje įrangoje. Kreipiantis per interneto protokolo užklausą suformuojamas modelis pagal gautus parametrus.

4.1. Sprendimo realizacija

Sprendimo realizacijos metu sukurta komponentė, skirta suformuoti ontologijų vaizdinio pateikimo modelį ir pradiniam ontologijos iškvietimui skirtas langas.

Ontologijų vaizdinio pateikimo modelio sprendimas realizuotas semantiniam tinklui sukūrus komponentę, kuria naudojantis galima peržiūrėti ontologijos duomenis grafiniame modelyje. Sukurtoje komponentėje grafinis modelis suformuojamas panaudojant *Arbor* biblioteką. Norint formuoti grafinį modelį prieigai prie komponentės nereikalinga autorizacija. Komponentė ontologijos duomenų saugojimui nenaudoja duomenų bazės, o ji duomenis tiesiogiai nuskaito iš nurodyto ontologijos failo, esančio ontologijų saugykloje. Nuskaitytą ontologijos failą suformuoja rezultatą, kurį pateikia grafiniu modeliu.

Realizacijai ir duomenų apdorojimui naudotos technologijos:

- *RDF / OWL / XML*, nes tai plačiausiai naudojamos standartinės kalbos;
- *JavaScript*, kad praplėsti ir pritaikyti *Arbor* bibliotekos funkcionalumą pagal reikalavimus;
- *Jena*, nes šis karkasas turi geriausiai išvystytas ontologijų apdorojimo galimybes;
- *JSON* duomenų perdavimui ir apdorojimui *JavaScript* funkcijose;
- *Arbor* biblioteka, nes yra sukurta atvaizduoti grafinius modelius ir suteikia plačias galimybes pritaikyti grafinį modelį pagal poreikius.

Ontologijos vaizdinio pateikimo modelio komponentės realizacijos veikla vykdoma tokiais etapais:

- Kreipiantis per interneto protokolą į komponentę, komponentė apdoroja GET arba POST metodais perduodamus parametrus, pateiktus 4.1 lentelėje, kurie yra privalomi norint sugeneruoti ontologijos vaizdinio pateikimo modelį.
- Naudojant *Jena* karkaso funkcijas apdorojamas ontologijos failas, kurio saugojimo vieta perduota su parametro „url“ reikšme. Atitinkamai pagal nurodyto parametro „type“ reikšmę suformuojamas grafinis modelis – ontologijos modelis su savybėmis arba ontologijos schema.
- Gauti duomenys per *Jena* karkasą, kurie yra neprieštaringi ir teisingai specifikuoti, sugrupuojami į klases, individus, duomenų savybes, objekto savybes, anotacijas ir pagal aprašytus ryšius tarp jų suformuojami elementų ryšiai su ryšių pavadinimais.
- Sugrupuoti duomenys konvertuojami į JSON formatą, pritaikytą *Arbor* bibliotekai. Pagal JSON formato duomenis *Arbor* biblioteka užpildoma tarpiniais ontologijos duomenimis.
- Pirminis ontologijos vaizdinio pateikimo modelis užpildomas šakniniais hierarchijos elementais, kurie nustatomi panaudojant *Jena* funkciją.
- Naviguojant po grafinio modelio elementus – spaudžiant ant elementų – iš tarpinės duomenų struktūros nuskaitomi pasirinkto elemento duomenys ir su juo susiję ryšiai. Nuskaityti tarpiniai duomenys perduodami atvaizdavimui ir pagal papildytus elementus ir ryšius tarp elementų, atnaujinamas grafinis modelis naudotojo peržiūrai.

4.1 lentelė. Komponentės parametrai

Parametras	Tipas	Galimos reikšmės	Aprašymas
url	Tekstas		Ontologijos bylos adresas.

Type	Tekstas	model | schema	Ontologijos vaizdinio pateikimo modelio peržiūros tipas.

4.2. Veikimo aprašas

4.2.1. Pradžia norint naudotis modeliu

Ontologijos vaizdinio pateikimo modelio komponentę galima pasiekti naudojant interneto naršyklę (*Internet Explorer, Firefox, Chrome* ar kita), nes komponentė veikia patalpinta serveryje.

4.2.2. Ontologijos vaizdinio pateikimo modelio iškvietimas

Norint sugeneruoti ontologijos vaizdinio pateikimo modelį, sukurtas pradinis ontologijos iškvietimo langas su įvedamais privalomais komponentei parametrais:

- Lauke „Ontologija“ įvedamas pilna nuoroda iki patalpintos ontologijos failo;
- Lauke „Peržiūros tipas“ pasirenkama reikšmė „modelis“ arba „schema“.

Paspaudus mygtuką „Peržiūrėti modelį“ nukreipiamas naudotojas į ontologijos vaizdinio pateikimo modelio komponentę su įvestais parametrais.

Pradinis ontologijos vaizdinio pateikimo modelio iškvietimo langas pateiktas 4.1 paveiksle.

4.1 pav. Ontologijos vaizdinio pateikimo iškvietimo langas

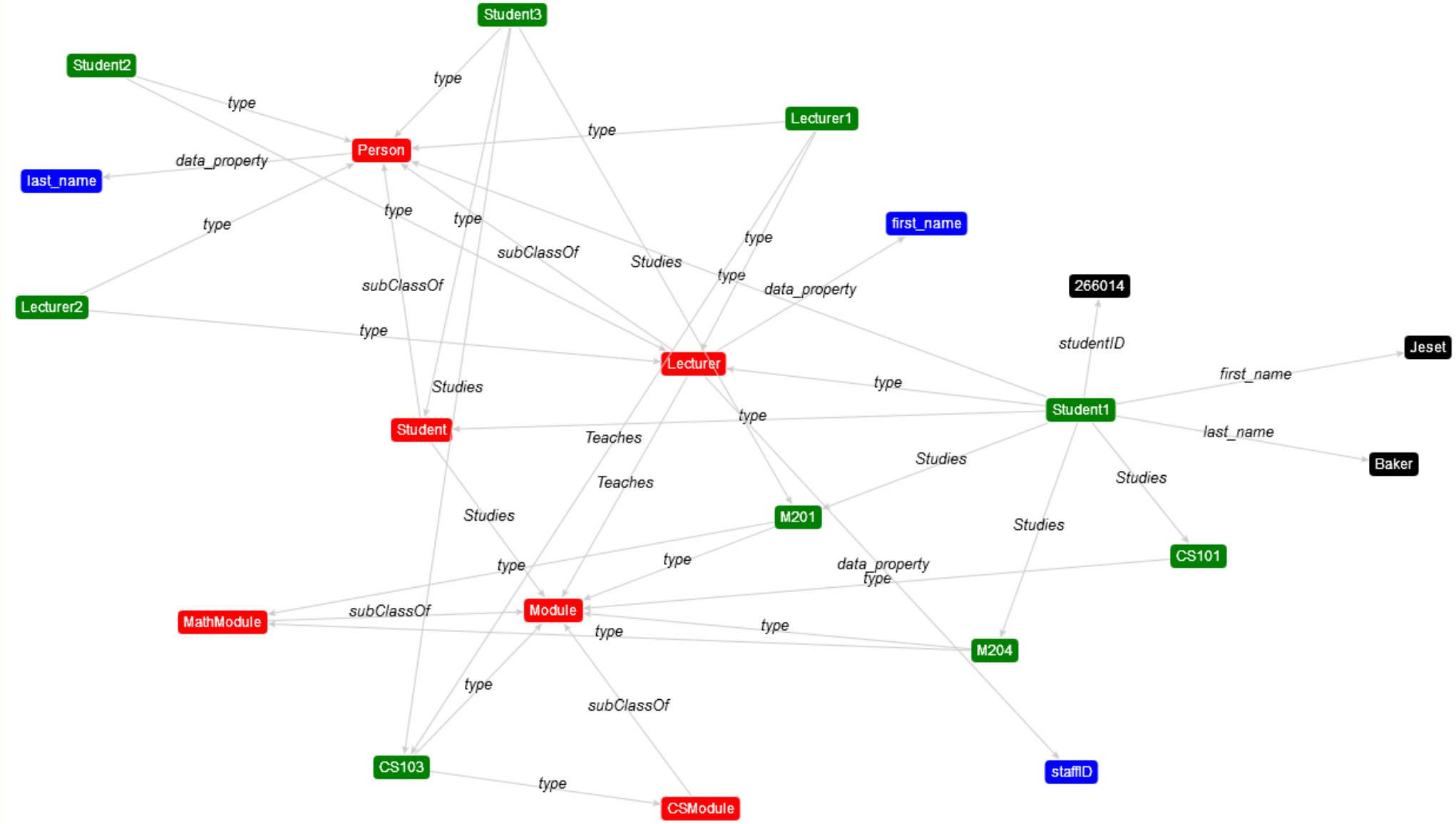
4.2.3. Suformuotas ontologijos vaizdinio pateikimo modelis

Su privalomais parametrais ir teisingomis reikšmėmis iškviesta ontologijos vaizdinio pateikimo modelio komponentė suformuoja grafinį modelį iš ontologijos šakninių hierarchijos elementų, nuo kurių galima pradėti vykdyti navigavimą po modelį. Paspaudus ant pasirinkto elemento pridedami susiję ontologijos elementai ir sujungiami tarpusavyje ryšiais tarp jų.

Ontologijos vaizdinio pateikimo modelyje, pateiktame 4.2 paveiksle, atvaizduojami ontologijos elementai:

- Klasė (*Class*) – raudonos spalvos modelio elementas.
- Individai (*NamedIndividual*) – žalios spalvos modelio elementas.
- Duomenų savybė (*DataProperty*) – mėlynos spalvos modelio elementas. Taip pat duomenų savybė nurodoma kaip ryšio pavadinimas tarp elementų.
- Objekto savybė (*ObjectProperty*) – violetinės spalvos modelio elementas. Taip pat objekto savybė nurodoma kaip ryšio pavadinimas tarp elementų.
- Predikatas (*Predicate*) – juodos spalvos modelio elementas.
- Anotacija (*Anotation*) – geltonos spalvos modelio elementas.
- Ryšiai tarp elementų – ryšiai tarp susietų elementų turi atvaizduojamą kryptį. Ryšiai tarp elementų turi ryšio pavadinimą, kuris atvaizduojamas juodu tekstu baltame fone.

Suformuotas ontologijos vaizdinio pateikimo modelis pateiktas 4.2 paveiksle.



4.2 pav. Ontologijos vaizdinio pateikimo modelio langas

4.3. Testavimo modelis, duomenys, rezultatai

Tam, kad patikrinti ar realizuotas ontologijų vaizdinio pateikimo modelis veikia tinkamai, reikalinga atlikti testavimą. Ontologijų vaizdinio pateikimo modelio komponentei testuoti numatyti testavimo scenarijai pateikti 4.2 lentelėje.

4.2 lentelė. Testavimo scenarijai

Nr.	Kas tikrinama	Laukiamas rezultatas	Ar gautas rezultatas?
1.	Komponentės lango atvėrimas su ontologijos duomenimis	Atidarytas ontologijų vaizdinio pateikimo modelio langas	Taip
2.	Navigavimas ontologijos elementais	Pridedama daugiau susijusių elementų ir ryšių prie šakninės hierarchijos elementų	Taip

5. EKSPERIMENTINIS MODELIO TYRIMAS

5.1. Eksperimento planas

Ontologijų vaizdinio pateikimo modelio tyrimui atlikti pasirinktas eksperimentinis tipas. Eksperimentui atlikti įvykdyti 10 bandymų su atsitiktinai pasirinktomis ontologijomis, kurių duomenys pateikti *RDF* formatu. Eksperimentu siekiama įvertinti kiek tiksliai gali atrinkti ir atvaizduoti ontologijos klasių, individų, duomenų savybių, objekto savybių, anotacijų ontologijų vaizdinio pateikimo modelio komponentė. Rezultatai fiksuojami ir statistiniu būdu palyginami su faktiškai suskaičiuotomis klasėmis, individais, duomenų savybėmis, objektų savybėmis ir anotacijomis, kurios aprašytos ontologijos faile.

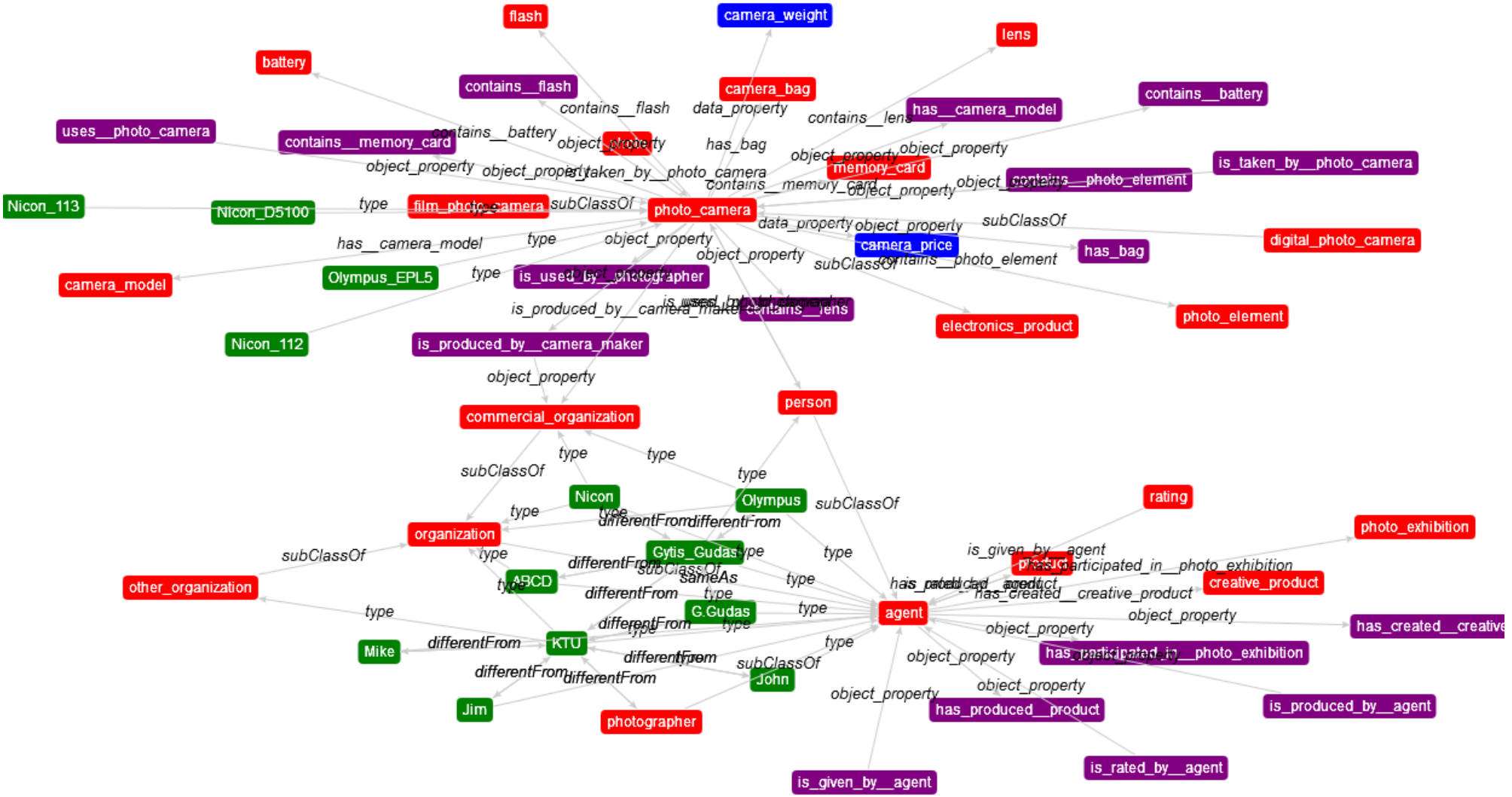
Eksperimento tikslas - išanalizuoti modelį siekiant įvertinti ontologijos vaizdinio pateikimo modelio komponentės efektyvumą.

Eksperimento planas:

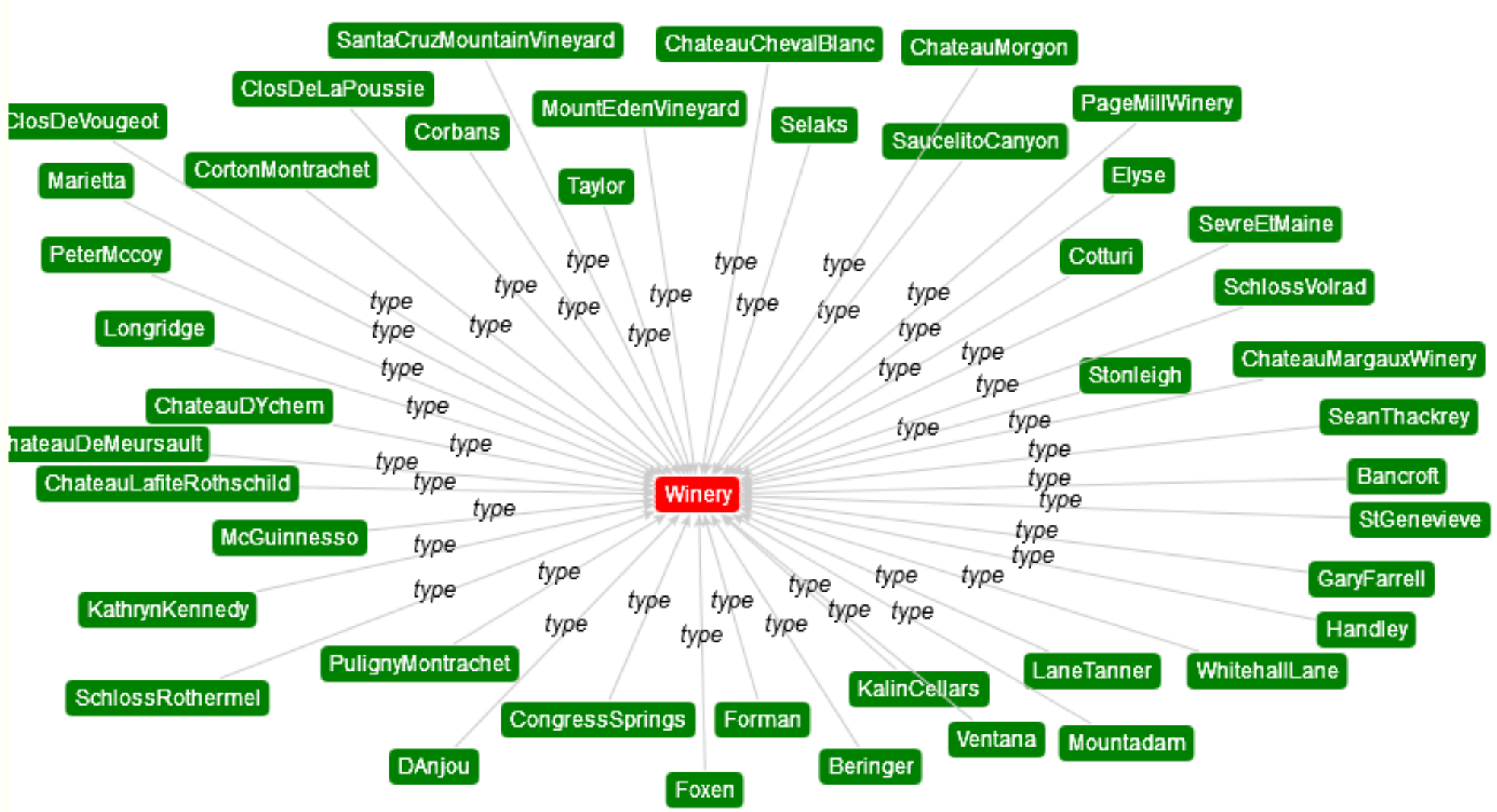
- Atlikti 10 bandymų su skirtingomis, atsitiktinai pasirinktomis ontologijomis, kurių duomenys pateikti *RDF* formatu.
- Bandymo metu ontologijų vaizdinio pateikimo modelio komponentė suskaičiuoja, kiek ontologijos elementų perduodama į komponentę.
- Eksperimento vykdytojas suskaičiuoja, kiek ontologijos faile faktiškai yra elementų, panaudodamas teksto redaktorių *Notepad++*.
- Statistiškai palyginami komponentės pateikti ir faktiniai elementų skaičiai.

5.2. Eksperimento rezultatai

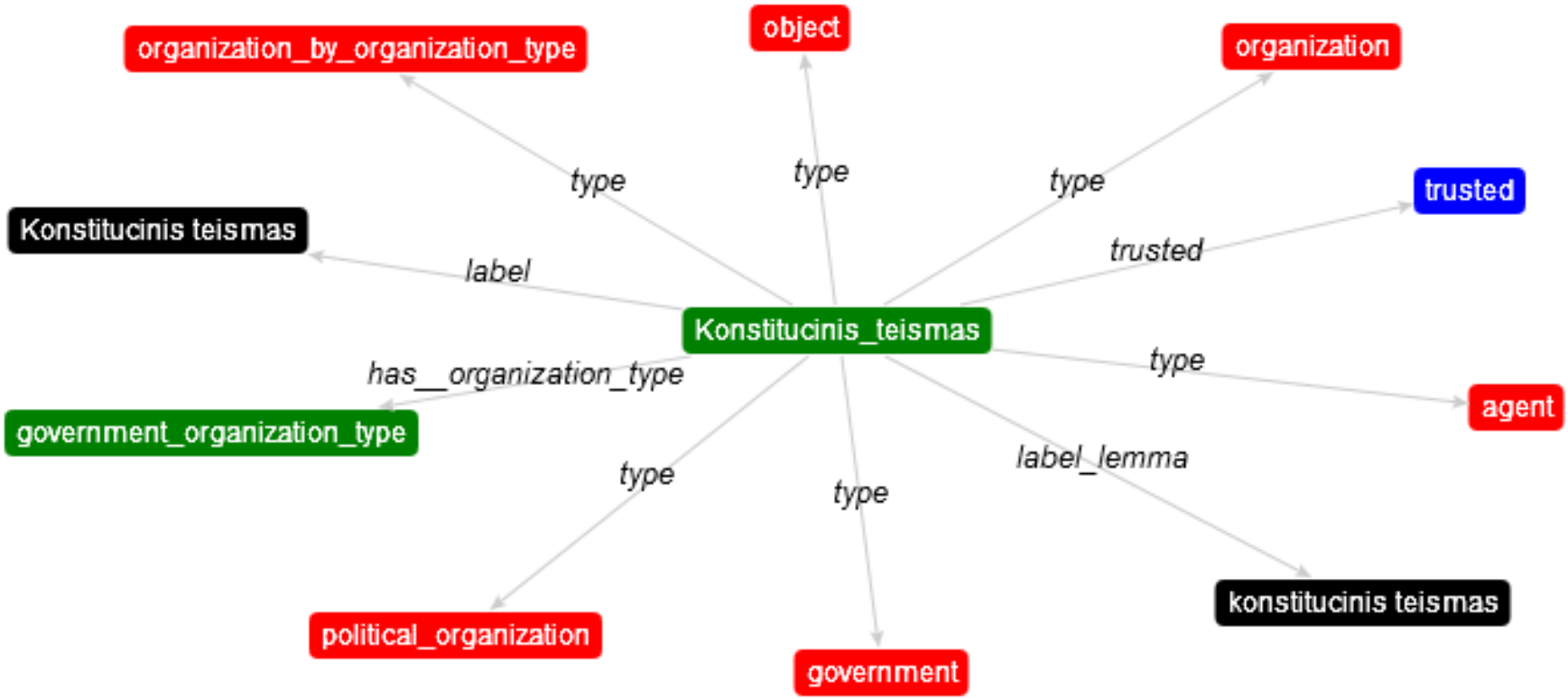
Atlikti 10 eksperimentų su skirtingomis ontologijomis, kurio metu kiekviena ontologija patikrinama su ontologijų vaizdinio pateikimo modelio komponente ir eksperimento vykdytojo, panaudojant teksto redaktorių *Notepad++*, paskaičiuojamas faktinis klasių, individų, duomenų savybių, objekto savybių ir anotacijų skaičius. Eksperimentų ontologijų vaizdiniai pateikimo modeliai, kuriuose pateikiami tam tikri pasirinkti ontologijos elementai su jų ryšiais ir ryšių pavadinimais, pateikti 5.1 – 5.5 paveiksluose. Skaičiavimų rezultatai su ontologijos vaizdinio pateikimo modelio komponente ir *Notepad++* teksto redaktoriumi pateikti 5.1 lentelėje.



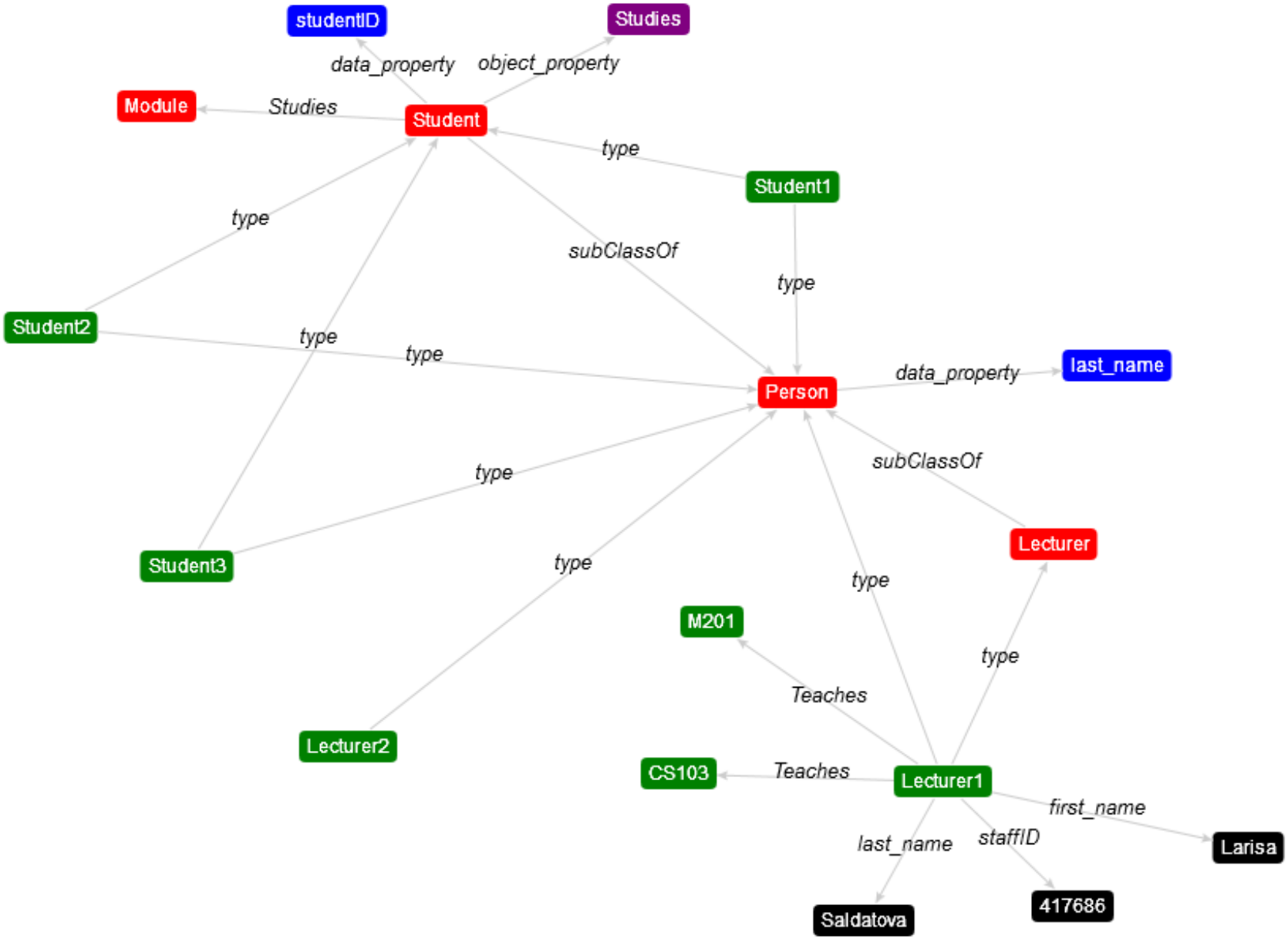
5.1 pav. Ontologijos „Photocameras.owl“ pasirinkti elementai ir jų ryšiai su pavadinimais



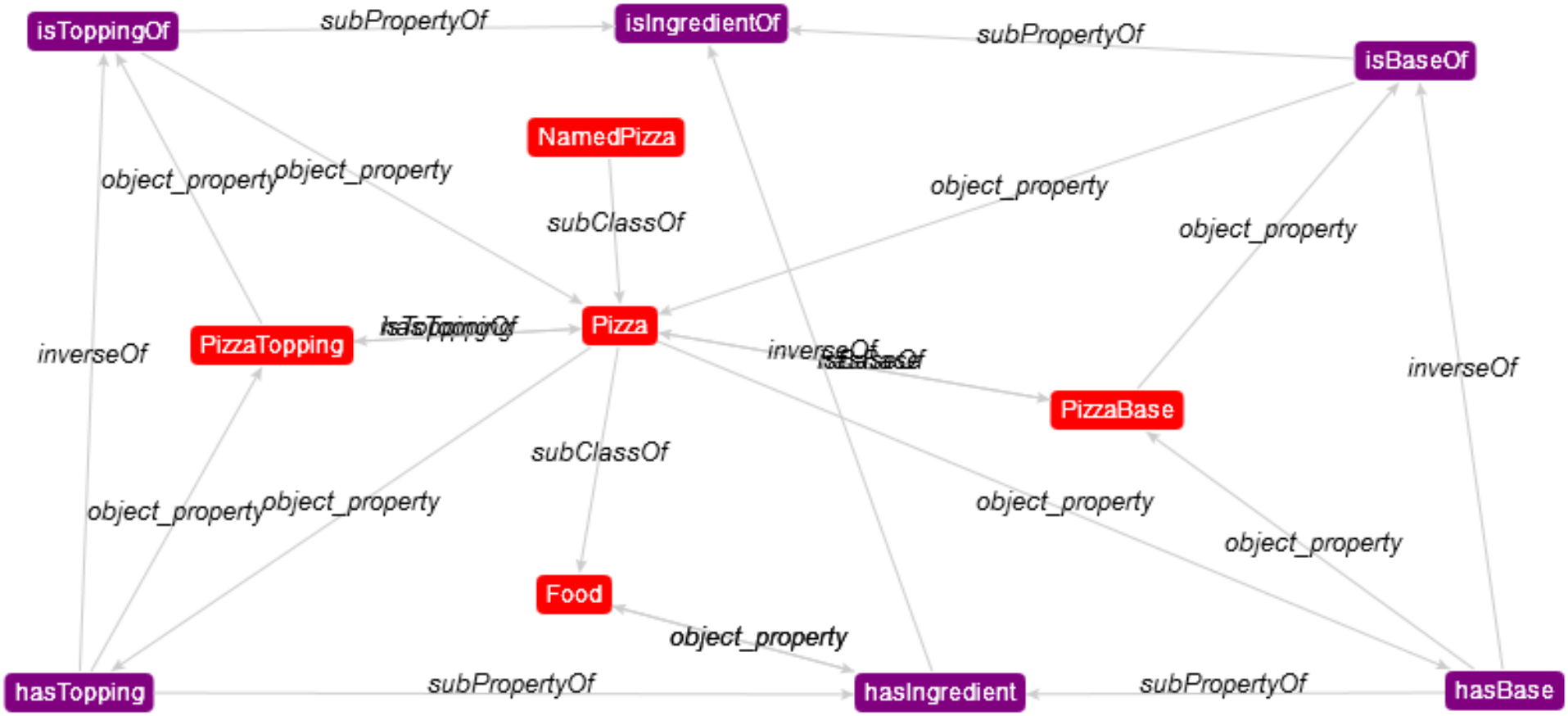
5.2 pav. Ontologijos „OntologyWineMerged.owl“ pasirinkta klasė su jai priklausančiais individais ir jų ryšiai su pavadinimais



5.3 pav. Ontologijos „AgentuOntologija.owl“ pasirinkto individo susiję elemantai ir jų ryšiai su pavadinimais



5.4 pav. Ontologijos „University.owl“ pasirinktos kelios klasės, susiję elementai ir jų ryšiai su pavadinimais



5.5 pav. Ontologijos „pizza.owl“ pasirinkta klasė ir jos savybės, susijusios klasės bei jų ryšiai su pavadinimais

5.1 lentelė. Komponentės nuskaitomų elementų skaičiaus ir faktiškai suskaičiuojamų elementų skaičių palyginimas

	Su ontologijų vaizdinio pateikimo modelio komponente					**Su *Notepad*++ teksto redaktoriumi**				
Ontologijos failo pavadinimas	**Klasės**	**Individai**	**Duomenų savybės**	**Objekto savybės**	**Anotacijos**	**Klasės**	**Individai**	**Duomenų savybės**	**Objekto savybės**	**Anotacijos**
Photocameras.owl	45	**40**	12	39	1	45	**41**	12	39	1
OntologyWineMerged.owl	74	161	1	12	0	74	161	1	12	0
AgentuOntologija.owl	57	102	8	9	19	57	102	8	9	19
University.owl	6	9	4	2	0	6	9	4	2	0
pizza.owl	100	5	0	8	0	100	5	0	8	0
PoliticsMerged.rdf	128	289	11	79	28	128	289	11	79	28
EconomyAndBusinessMerged.rdf	135	51	11	96	19	135	51	11	96	19
EventsMerged.owl	113	230	11	62	23	113	230	11	62	23
PublicAdministrativeMerged.rdf	119	238	11	78	23	119	238	11	78	23
Plants.owl	1691	0	0	10	47	1691	0	0	10	47

5.3. Sprendimo veikimo ir savybių analizė, kokybės kriterijų įvertinimas

Išanalizavus eksperimento rezultatus, pateiktus 5.1 lentelėje, nustatyta, kad atsitiktinai pasirinktos ontologijos *Photocameras.owl* nuskaitomų individų skaičius su ontologijų vaizdinio pateikimo modelio komponente nesutampa su individų skaičiumi, rastų su *Notepad++* teksto redaktoriumi. Ontologijų vaizdinio pateikimo modelio komponentė iš ontologijos failo nuskaito 40 individų, o teksto redaktoriaus pagalba galima rasti 41 individą. Atlikus paiešką tarp nuskaitytų elementų nustatytas individas, kurio ontologijos vaizdinio pateikimo modelio komponentė nenuskaitė iš ontologijos failo. Išanalizavus individo sintaksės aprašymą nustatyta, kad individas *Nicon_D5100* ontologijos faile *Photocameras.owl* yra aprašytas netaisyklinga sintakse. Ontologijos vaizdinio pateikimo modelio komponentė nenuskaito netaisyklingai aprašytų ontologijos elementų. Netaisyklingai aprašyto individo sintaksė pateikta 5.6 paveiksle.

```
2188    <!-- http://semantika.lt/ns/photocameras#Nicon_D5100 -->
2189
2190    <NamedIndividual rdfsyn:about="http://semantika.lt/ns/photocameras#Nicon_D5100">
2191        <rdfsyn:type rdfsyn:resource="http://semantika.lt/ns/photocameras#digital_photo_camera"/>
2192        <photo:camera_weight rdfsyn:datatype="http://www.w3.org/2001/XMLSchema#int">150</photo:camera_weight>
2193        <photo:has__rating rdfsyn:resource="http://semantika.lt/ns/photocameras#Average_rating"/>
2194        <photo:has__camera_model rdfsyn:resource="http://semantika.lt/ns/photocameras#COOLPIX_P600"/>
2195        <photo:is_produced_by__camera_maker rdfsyn:resource="http://semantika.lt/ns/photocameras#Nicon"/>
2196        <photo:has__accessory rdfsyn:resource="http://semantika.lt/ns/photocameras#Nicon_bag"/>
2197    </NamedIndividual>
2198    <rdfsyn:Description>
2199        <rdfsyn:type rdfsyn:resource="http://www.w3.org/2002/07/owl#NegativePropertyAssertion"/>
2200        <sourceIndividual rdfsyn:resource="http://semantika.lt/ns/photocameras#Nicon_D5100"/>
2201        <assertionProperty rdfsyn:resource="http://semantika.lt/ns/photocameras#contains__photo_element"/>
2202        <targetIndividual rdfsyn:resource="http://semantika.lt/ns/photocameras#sensor2"/>
2203    </rdfsyn:Description>
2204    <rdfsyn:Description>
2205        <rdfsyn:type rdfsyn:resource="http://www.w3.org/2002/07/owl#NegativePropertyAssertion"/>
2206        <targetValue rdfsyn:datatype="http://www.w3.org/2001/XMLSchema#string">151</targetValue>
2207        <sourceIndividual rdfsyn:resource="http://semantika.lt/ns/photocameras#Nicon_D5100"/>
2208        <assertionProperty rdfsyn:resource="http://semantika.lt/ns/photocameras#product_number"/>
2209    </rdfsyn:Description>
```

5.6 pav. Individo Nicon_D5100 sintaksė

Palyginus kitus rezultatus nesutampančių duomenų nenustatyta, todėl galima vertinti, kad ontologijų vaizdinio pateikimo modelis sugeba nuskaityti visus ontologijos failo elementus, jeigu elementai aprašyti be sintaksės klaidų.

Atlikus eksperimentą nustatytos savybės, kurias atlieka ontologijų vaizdinio pateikimo modelio komponentė ir šio sprendimo savybių palyginimas pateiktas 5.2 lentelėje.

5.2 lentelė. Sprendimų savybių palyginimas

Kriterijus	**GeoNames**	**EMM News Explorer**	**The Gene Ontology**	**Ontologijos vaizdinio pateikimo modelio komponentė**
Paieška	+	+	+	+
Grafinis vaizdavimas	Žemėlapis	Ontologijos modelis	Ontologijos modelis	Ontologijos modelis
Ontologijų hierachija	+	+	+	+
Ontologijų elementų ryšiai	+	+	+	+
Ontologijų elementų ryšių pavadinimai	-	-	-	+
Ontologijos savybių peržiūra	+	+	+	+

Universalumas	-	-	-	+

5.4. Sprendimo taikymo rekomendacijos

Ontologijos vaizdinio pateikimo įrankį galima naudoti kaip komponentą įvairiose sistemose arba pasinaudoti įdiegtu komponentu, pasiekiamu per interneto naršyklę. Ontologijos vaizdinio pateikimo įrankį įvairiose sistemos kaip komponentą rekomenduojama naudoti vienu iš dviejų būdų: modelio įterpimo į sistemos langą arba tiesioginio naudotojo nukreipimo į modelį būdu.

5.4.1. Modelio įterpimas į sistemos langą

Komponentą su atvaizduojamu ontologijos vaizdinio pateikimo modeliu galima įterpti į sistemą arba programinę įrangą, kuri veikia nutolusiame serveryje. Įterpiant komponento sugeneruotą ontologijos vaizdinio pateikimo modelį reikia perduoti parametrus *GET* metodu su nurodytomis reikšmėmis pagal 5.3 lentelėje pateiktus nurodymus. Iškviesti komponentą į savo sistemą arba programinę įrangą ir įterpti sugeneruotą ontologijos vaizdinio pateikimo modelį galima naudojant *HTML* žymą *<iframe>*.

5.3 lentelė. Komponento parametrai modelio įterpimui į programinę įrangą

Parametras	**Tipas**	**Galimos reikšmės**	**Aprašymas**
url	Tekstas		Ontologijos adresas.
type	Tekstas	model | schema	Ontologijos vaizdinio pateikimo modelio peržiūros tipas.

5.4.2. Tiesioginis naudotojo nukreipimas į modelį

Ontologijos vaizdinio pateikimo modelį galima naudoti vykdant tiesioginį kreipinį *GET* arba POST metodu per interneto protokolą, nukreipiant naudotoją naujame arba esamame naršyklės lange į ontologijos vaizdinio pateikimo modelį. Tiesiogiai į komponentą kreiptis galima iš nutolusiame serveryje veikiančios sistemos arba programinės įrangos. Kreipiantis į komponentą tiesiogiai *GET* arba *POST* metodu reikia perduoti parametrus su nurodytomis reikšmėmis pagal 5.4 lentelėje pateiktus nurodymus. Perduoti *GET* arba *POST* metodu parametrus į komponentą galima naudojant *HTML* žymą *<form>*.

5.4 lentelė. Komponento parametrai tiesioginiam naudotojo nukreipimui į modelį

Parametras	**Tipas**	**Galimos reikšmės**	**Aprašymas**
url	Tekstas		Ontologijos bylos adresas.
Type	Tekstas	model | schema	Ontologijos vaizdinio pateikimo modelio peržiūros tipas.

6. REZULTATŲ APIBENDRINIMAS IR IŠVADOS

1. Informacijos paieškai ir analizei vis dažniau taikant semantines technologijas ir ontologijų kalbas, jų vartotojų poreikių analizė rodo, kad prieš atliekant paiešką tikslinga išanalizuoti ontologiją, kuria remiasi paieška, tačiau trūksta patogių įrankių, kurie leistų tai atlikti.
2. Esamų ontologijų vaizdavimo įrankių analizė parodė, kad dažniausiai pasitaikančios problemos yra ontologijos vaizdavimo įrankių universalumo trūkumas (pritaikymas vienai sričiai) ir negalėjimas parodyti visų savybių (dažniausiai – ryšių pavadinimų), todėl buvo nuspręsta sukurti universalų įrankį, kuris leistų pavaizduoti bet kokių sričių ontologijas su visomis savybėmis.
3. Kad sprendimas būtų prieinamas daugeliui naudotojų, jis buvo kuriamas kaip savarankiškas komponentas, kuriuo galima naudotis pasauliniame tinkle arba integruoti į norimą sistemą.
4. Norit užtikrinti šias galimybes, ontologijos vaizdinio pateikimo modelis buvo sudarytas remiantis *W3C* konsorciumo ontologijos metamodeliu; realizacijai pasirinktas *Jena* karkasas, turintis geriausiai išvystytas ontologijų apdorojimo galimybes, vaizdiniam pateikimui – *Arbor* biblioteka, kuri geba atvaizduoti įvairius grafinius modelius, atskirti modelio elementus, sujungti elementus ryšiais ir pateikti ryšių pavadinimus.
5. Eksperimentinis tyrimas, atliktas su 10 skirtingų sričių ontologijomis, parodė, kad įrankis leidžia pavaizduoti bet kokių sričių ontologijas su visų tipų savybėmis.
6. Detali vaizduojamų elementų analizė parodė, kad ontologijos vaizdinio pateikimo įrankis sugeba nuskaityti visus teisingai specifikuotus ontologijos elementus ir tik klaidingi elementai nepateikiami.

7. LITERATŪRA

[1] T. Berners-Lee, J. Hendler ir O. Lassila, „The Semantic Web," *Scientific American*, May 2001.

[2] E. Mäkelä, View-Based User Interfaces for the Semantic Web, 2010-10-26.

[3] „Semantic Web," W3C, 2013. [Tinkle]. Available: http://www.w3.org/standards/semanticweb/. [Kreiptasi 19 05 2015].

[4] S. S. Guo ir C. W. Chan, A Comparison and Analysis of Some Ontology Visualization Tools.

[5] „Jena," [Tinkle]. Available: http://jena.apache.org/tutorials/rdf_api.html . [Kreiptasi 17 05 2015].

[6] J. J. Carrol, I. Dickinson, C. Dollin, D. Reynolds, A. Seaborne ir K. Wilkinson, Jena: Implementing the Semantic Web Recommendations, 2003-12-24.

[7] B. Motik, P. F. Patel-Schneider, B. Parsia, C. Bock, A. Fokoue, P. Haase, R. Hoekstra, I. Horrocks, A. Ruttenberg, U. Sattler ir M. Smith, OWL 2 Web Ontology Language Structural Specification and Functional-Style Syntax (Second Edition), 2012.

[8] E. Sirin ir B. Parsia, SPARQL-DL: SPARQL Query for OWL-DL.

[9] „SPARQL Query Tests," W3C, [Tinkle]. Available: http://www.w3.org/2001/sw/DataAccess/rq23/examples.html. [Kreiptasi 20 05 2015].

[10] F. Lindörfer, Semantic Web Frameworks, 2010.

[11] F. Ghaleb, S. Daoud, A. Hasna, J. M. ALJa'am, S. A. El-Seoud ir H. El-Sofany, E-Learning Model Based On Semantic Web Technology, 2006.

[12] R. Oldakowski, C. Bizer ir D. Westphal, RAP: RDF API for PHP.

[13] J. Dmitrieva ir F. J. Verbeek, Node-Link and Containment Methods in Ontology Visualization.

[14] „Arbor," 2011. [Tinkle]. Available: http://arborjs.org/introduction. [Kreiptasi 15 05 2015].

[15] „GeoNames," [Tinkle]. Available: http://www.geonames.org/maps/showOnMap?q=Kaunas. [Kreiptasi 19 05 2015].

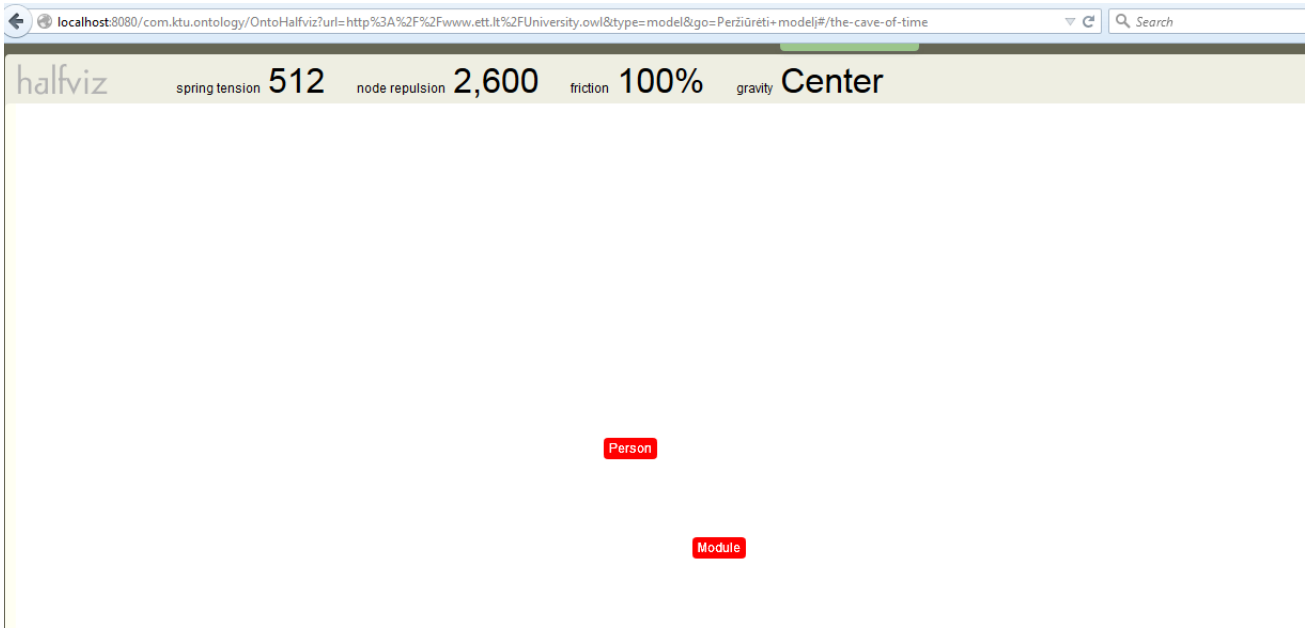
[16] „EMM News Explorer," [Tinkle]. Available: http://emm.newsexplorer.eu/NewsExplorer/flash/visual.jsp?id=759391. [Kreiptasi 19 05 2015].

[17] „The Gene Ontology," [Tinkle]. Available: http://amigo.geneontology.org/amigo/term/NCBITaxon:9567. [Kreiptasi 19 05 2015].

8. PRIEDAI

8.1.priedas. Testavimo rezultatai Nr. 1

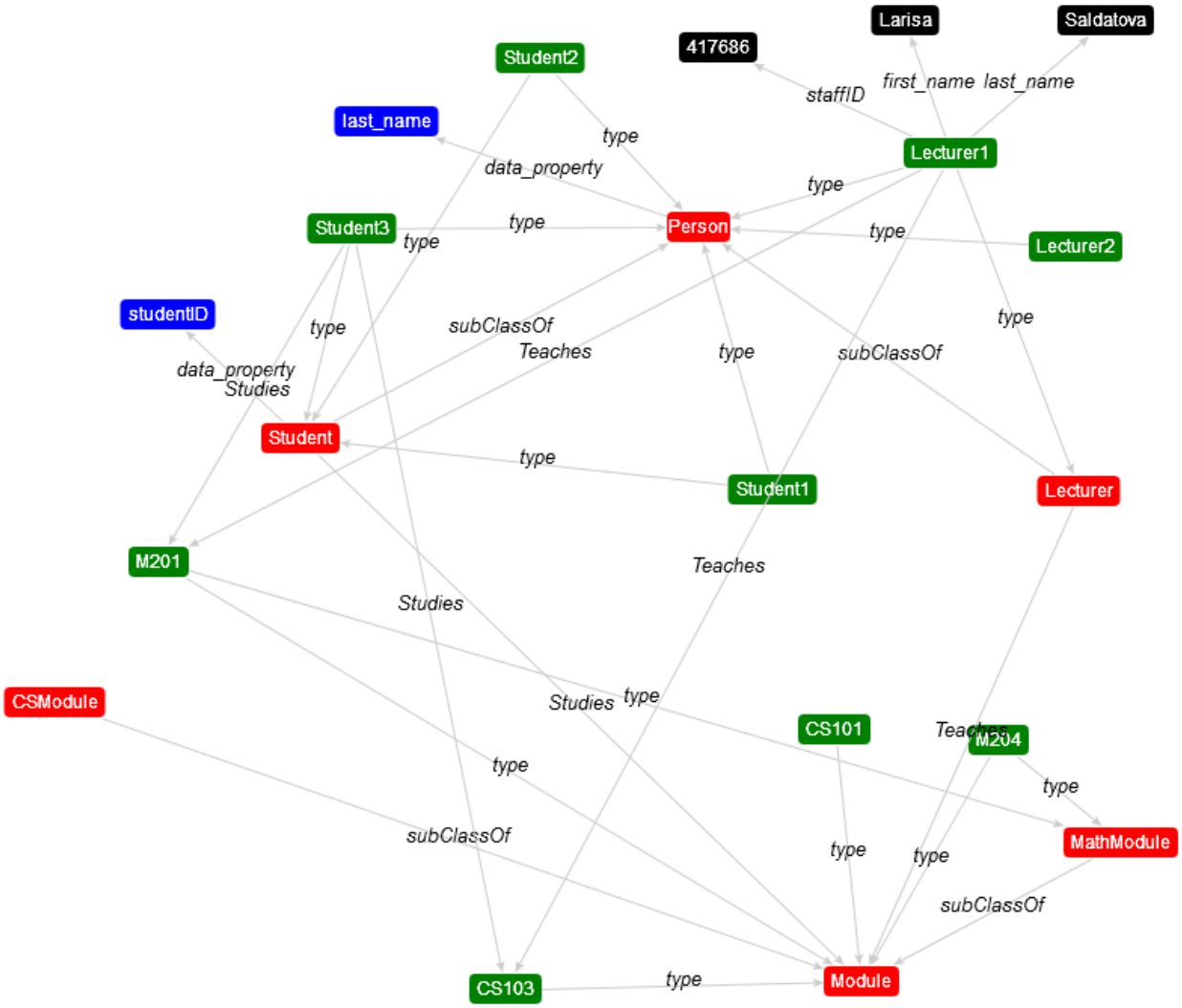
Testavimo scenarijaus Nr. 1 testavimo rezultatų ekrano vaizdas pateiktas 8.1 paveiksle.



8.1 pav. Komponentės atvertas langas

8.2.priedas. Testavimo rezultatai Nr. 2

Testavimo scenarijaus Nr. 2 testavimo rezultatų ekrano vaizdas pateiktas 8.2 paveiksle.



8.2 pav. Ontologijos elementai su ryšiais

8.3.priedas. Eksperimento duomenys

Bandymo su *Photocameras.owl* ontologijos failu, naudojant ontologijų vaizdinio pateikimo modelio komponentę, rezultatai pateikti 8.3 paveiksle.

```
--------------------------------------
Ontologijos URL:http://www.ett.lt/Photocameras.owl
Klasiu skaicius: 45
Duomenu savybiu skaicius: 12
Objekto savybiu skaicius: 39
Individu skaicius: 40
Anotacijos skaicius: 1
--------------------------------------
```

8.3 pav. Eksperimento rezultatai su Photocameras.owl

Bandymo su *OntologyWineMerged.owl* ontologijos failu, naudojant ontologijų vaizdinio pateikimo modelio komponentę, rezultatai pateikti 8.4 paveiksle.

```
-------------------------------------
Ontologijos URL:http://www.ett.lt/OntologyWineMerged.owl
Klasiu skaicius: 74
Duomenu savybiu skaicius: 1
Objekto savybiu skaicius: 12
Individu skaicius: 161
Anotacijos skaicius: 0
-------------------------------------
```

8.4 pav. Eksperimento rezultatai su OntologyWineMerged.owl

Bandymo su *AgentuOntologija.owl* ontologijos failu, naudojant ontologijų vaizdinio pateikimo modelio komponentę, rezultatai pateikti 8.5 paveiksle.

```
-------------------------------------
Ontologijos URL:http://www.ett.lt/AgentuOntologija.owl
Klasiu skaicius: 57
Duomenu savybiu skaicius: 8
Objekto savybiu skaicius: 9
Individu skaicius: 102
Anotacijos skaicius: 19
-------------------------------------
```

8.5 pav. Eksperimento rezultatai su AgentuOntologija.owl

Bandymo su *University.owl* ontologijos failu, naudojant ontologijų vaizdinio pateikimo modelio komponentę, rezultatai pateikti 8.6 paveiksle.

```
-------------------------------------
Ontologijos URL:http://www.ett.lt/University.owl
Klasiu skaicius: 6
Duomenu savybiu skaicius: 4
Objekto savybiu skaicius: 2
Individu skaicius: 9
Anotacijos skaicius: 0
-------------------------------------
```

8.6 pav. Eksperimento rezultatai su University.owl

Bandymo su *pizza.owl* ontologijos failu, naudojant ontologijų vaizdinio pateikimo modelio komponentę, rezultatai pateikti 8.7 paveiksle.

```
-------------------------------------
Ontologijos URL:http://www.dcs.bbk.ac.uk/~michael/sw/slides/pizza.owl
Klasiu skaicius: 100
Duomenu savybiu skaicius: 0
Objekto savybiu skaicius: 8
Individu skaicius: 5
Anotacijos skaicius: 0
-------------------------------------
```

8.7 pav. Eksperimento rezultatai su pizza.owl

Bandymo su *PoliticsMerged.rdf* ontologijos failu, naudojant ontologijų vaizdinio pateikimo modelio komponentę, rezultatai pateikti 8.8 paveiksle.

```
-------------------------------------
Ontologijos URL:http://www.ett.lt/PoliticsMerged.rdf
Klasiu skaicius: 128
Duomenu savybiu skaicius: 11
Objekto savybiu skaicius: 79
Individu skaicius: 289
Anotacijos skaicius: 28
-------------------------------------
```

8.8 pav. Eksperimento rezultatai su PoliticsMerged.rdf

Bandymo su *EconomyAndBusinessMerged.rdf* ontologijos failu, naudojant ontologijų vaizdinio pateikimo modelio komponentę, rezultatai pateikti 8.9 paveiksle.

```
-------------------------------------
Ontologijos URL:http://www.ett.lt/EconomyAndBusinessMerged.rdf
Klasiu skaicius: 135
Duomenu savybiu skaicius: 11
Objekto savybiu skaicius: 96
Individu skaicius: 51
Anotacijos skaicius: 19
-------------------------------------
```

8.9 pav. Eksperimento rezultatai su EconomyAndBusinessMerged.rdf

Bandymo su *EventsMerged.owl* ontologijos failu, naudojant ontologijų vaizdinio pateikimo modelio komponentę, rezultatai pateikti 8.10 paveiksle.

```
-------------------------------------
Ontologijos URL:http://www.ett.lt/EventsMerged.owl
Klasiu skaicius: 113
Duomenu savybiu skaicius: 11
Objekto savybiu skaicius: 62
Individu skaicius: 230
Anotacijos skaicius: 23
-------------------------------------
```

8.10 pav. Eksperimento rezultatai su EventsMerged.owl

Bandymo su *PublicAdministrativeMerged.rdf* ontologijos failu, naudojant ontologijų vaizdinio pateikimo modelio komponentę, rezultatai pateikti 8.11 paveiksle.

```
-------------------------------------
Ontologijos URL:http://www.ett.lt/PublicAdministrativeMerged.rdf
Klasiu skaicius: 119
Duomenu savybiu skaicius: 11
Objekto savybiu skaicius: 78
Individu skaicius: 238
Anotacijos skaicius: 23
-------------------------------------
```

8.11 pav. Eksperimento rezultatai su PublicAdministrativeMerged.rdf

Bandymo su *Plants.owl* ontologijos failu, naudojant ontologijų vaizdinio pateikimo modelio komponentę, rezultatai pateikti 8.12 paveiksle.

```
-------------------------------------
Ontologijos URL:http://www.ett.lt/Plants.owl
Klasiu skaicius: 1691
Duomenu savybiu skaicius: 0
Objekto savybiu skaicius: 10
Individu skaicius: 0
Anotacijos skaicius: 47
-------------------------------------
```

8.12 pav. Eksperimento rezultatai su Plants.owl

8.4.priedas. Eksperimentinė ontologija OWL formatu

Prefix(owl:=<http://www.w3.org/2002/07/owl#>)
Prefix(rdf:=<http://www.w3.org/1999/02/22-rdf-syntax-ns#>)
Prefix(urn:=<http://semantika.lt/ns/str_05_140110#>)
Prefix(xml:=<http://www.w3.org/XML/1998/namespace>)
Prefix(xsd:=<http://www.w3.org/2001/XMLSchema#>)

```
Prefix(rdfs:=<http://www.w3.org/2000/01/rdf-schema#>)
Prefix(photo:=<http://semantika.lt/ns/photocameras#>)


Ontology(<http://semantika.lt/ns/photocameras#>

Declaration(Class(photo:accessory))
EquivalentClasses(photo:accessory
ObjectIntersectionOf(ObjectSomeValuesFrom(photo:is_accessory_of__product photo:product)
photo:product))
SubClassOf(photo:accessory photo:role)
Declaration(Class(photo:agent))
EquivalentClasses(photo:agent ObjectUnionOf(photo:person photo:organization))
DisjointUnion(photo:agent photo:person photo:organization)
Declaration(Class(photo:average_rated_product))
EquivalentClasses(photo:average_rated_product
ObjectIntersectionOf(ObjectHasValue(photo:has__rating photo:Average_rating) photo:product_by_rating))
SubClassOf(photo:average_rated_product photo:product_by_rating)
Declaration(Class(photo:average_weight_camera))
EquivalentClasses(photo:average_weight_camera
ObjectIntersectionOf(DataSomeValuesFrom(photo:camera_weight DatatypeRestriction(xsd:int
xsd:minExclusive "200"^^xsd:integer xsd:maxInclusive "1000"^^xsd:integer)) photo:camera_by_weight))
SubClassOf(photo:average_weight_camera photo:camera_by_weight)
Declaration(Class(photo:award))
SubClassOf(photo:award ObjectMinCardinality(1 photo:received_for__photo photo:photo))
Declaration(Class(photo:battery))
SubClassOf(photo:battery photo:electronics_product)
Declaration(Class(photo:camera_bag))
SubClassOf(photo:camera_bag photo:other_product)
Declaration(Class(photo:camera_by_weight))
AnnotationAssertion(rdfs:label photo:camera_by_weight "camera by weight "@en)
AnnotationAssertion(photo:label_sbvr photo:camera_by_weight "camera_by_weight"@en)
EquivalentClasses(photo:camera_by_weight photo:photo_camera)
SubClassOf(photo:camera_by_weight photo:electronics_product)
DisjointUnion(photo:camera_by_weight photo:average_weight_camera photo:heavy_camera
photo:lightweight_camera)
Declaration(Class(photo:camera_maker))
EquivalentClasses(photo:camera_maker
ObjectIntersectionOf(ObjectSomeValuesFrom(photo:has_produced__product photo:photo_camera)
photo:commercial_organization))
SubClassOf(photo:camera_maker photo:role)
Declaration(Class(photo:camera_model))
SubClassOf(photo:camera_model DataExactCardinality(1 photo:is_fast xsd:boolean))
SubClassOf(photo:camera_model DataExactCardinality(1 photo:is_professional xsd:boolean))
SubClassOf(photo:camera_model DataExactCardinality(1 photo:is_smart xsd:boolean))
Declaration(Class(photo:commercial_organization))
SubClassOf(photo:commercial_organization photo:organization)
Declaration(Class(photo:component))
EquivalentClasses(photo:component
ObjectIntersectionOf(ObjectSomeValuesFrom(photo:is_contained_in__product photo:product)
photo:product))
SubClassOf(photo:component photo:role)
Declaration(Class(photo:creative_product))
SubClassOf(photo:creative_product photo:product)
Declaration(Class(photo:digital_photo_camera))
EquivalentClasses(photo:digital_photo_camera
ObjectIntersectionOf(ObjectSomeValuesFrom(photo:contains__photo_element photo:digital_sensor)
photo:photo_camera))
```

SubClassOf(photo:digital_photo_camera photo:photo_camera)
SubClassOf(photo:digital_photo_camera ObjectAllValuesFrom(photo:contains__photo_element photo:digital_sensor))
DisjointClasses(photo:digital_photo_camera photo:film_photo_camera)
Declaration(Class(photo:digital_sensor))
SubClassOf(photo:digital_sensor photo:photo_element)
DisjointClasses(photo:digital_sensor photo:film_sensor)
Declaration(Class(photo:electronics_product))
SubClassOf(photo:electronics_product photo:product)
Declaration(Class(photo:film_photo_camera))
SubClassOf(photo:film_photo_camera photo:photo_camera)
SubClassOf(photo:film_photo_camera ObjectAllValuesFrom(photo:contains__photo_element photo:film_sensor))
DisjointClasses(photo:film_photo_camera photo:digital_photo_camera)
Declaration(Class(photo:film_sensor))
EquivalentClasses(photo:film_sensor ObjectIntersectionOf(ObjectComplementOf(photo:digital_sensor) photo:photo_element))
SubClassOf(photo:film_sensor photo:photo_element)
DisjointClasses(photo:film_sensor photo:digital_sensor)
Declaration(Class(photo:flash))
SubClassOf(photo:flash photo:electronics_product)
Declaration(Class(photo:heavy_camera))
EquivalentClasses(photo:heavy_camera ObjectIntersectionOf(DataSomeValuesFrom(photo:camera_weight DatatypeRestriction(xsd:int xsd:minExclusive "1000"^^xsd:integer)) photo:camera_by_weight))
SubClassOf(photo:heavy_camera photo:camera_by_weight)
Declaration(Class(photo:lens))
SubClassOf(photo:lens photo:electronics_product)
Declaration(Class(photo:lightweight_camera))
EquivalentClasses(photo:lightweight_camera ObjectIntersectionOf(DataSomeValuesFrom(photo:camera_weight DatatypeRestriction(xsd:int xsd:minInclusive "0"^^xsd:integer xsd:maxInclusive "200"^^xsd:integer)) photo:camera_by_weight))
SubClassOf(photo:lightweight_camera photo:camera_by_weight)
Declaration(Class(photo:low_rated_product))
SubClassOf(photo:low_rated_product photo:product_by_rating)
Declaration(Class(photo:memory_card))
SubClassOf(photo:memory_card photo:electronics_product)
Declaration(Class(photo:organization))
EquivalentClasses(photo:organization ObjectIntersectionOf(ObjectComplementOf(photo:person) photo:agent))
SubClassOf(photo:organization photo:agent)
DisjointClasses(photo:organization photo:person)
Declaration(Class(photo:other_organization))
SubClassOf(photo:other_organization photo:organization)
Declaration(Class(photo:other_product))
SubClassOf(photo:other_product photo:product)
Declaration(Class(photo:person))
SubClassOf(photo:person photo:agent)
DisjointClasses(photo:person photo:organization)
Declaration(Class(photo:photo))
EquivalentClasses(photo:photo photo:picture)
SubClassOf(photo:photo photo:creative_product)
Declaration(Class(photo:photo_artist))
EquivalentClasses(photo:photo_artist ObjectIntersectionOf(ObjectSomeValuesFrom(photo:has_participated_in__photo_exhibition photo:photo_exhibition) photo:photographer))
SubClassOf(photo:photo_artist photo:photographer)
Declaration(Class(photo:photo_camera))

AnnotationAssertion(photo:label_sbvr photo:photo_camera "photo_camera"@en)
AnnotationAssertion(rdfs:label photo:photo_camera "photo camera"@en)
EquivalentClasses(photo:photo_camera photo:camera_by_weight)
SubClassOf(photo:photo_camera photo:electronics_product)
SubClassOf(photo:photo_camera ObjectAllValuesFrom(photo:is_produced_by__camera_maker photo:commercial_organization))
SubClassOf(photo:photo_camera ObjectMinCardinality(1 photo:contains__memory_card photo:memory_card))
SubClassOf(photo:photo_camera ObjectExactCardinality(2 photo:contains__battery photo:battery))
SubClassOf(photo:photo_camera ObjectExactCardinality(1 photo:contains__flash photo:flash))
SubClassOf(photo:photo_camera ObjectExactCardinality(1 photo:contains__photo_element photo:photo_element))
SubClassOf(photo:photo_camera ObjectMaxCardinality(2 photo:contains__memory_card photo:memory_card))
Declaration(Class(photo:photo_element))
EquivalentClasses(photo:photo_element ObjectUnionOf(photo:film_sensor photo:digital_sensor))
SubClassOf(photo:photo_element photo:electronics_product)
Declaration(Class(photo:photo_exhibition))
Declaration(Class(photo:photographer))
EquivalentClasses(photo:photographer ObjectIntersectionOf(ObjectSomeValuesFrom(photo:captured__photo photo:photo) photo:person))
SubClassOf(photo:photographer photo:role)
Declaration(Class(photo:picture))
EquivalentClasses(photo:picture photo:photo)
SubClassOf(photo:picture photo:creative_product)
Declaration(Class(photo:product))
EquivalentClasses(photo:product photo:product_by_rating)
SubClassOf(photo:product ObjectHasSelf(photo:is_replaced_by__replacing_product))
Declaration(Class(photo:product_by_rating))
EquivalentClasses(photo:product_by_rating photo:product)
Declaration(Class(photo:product_part))
EquivalentClasses(photo:product_part ObjectIntersectionOf(ObjectSomeValuesFrom(photo:is_part_of__product photo:product) photo:product))
SubClassOf(photo:product_part photo:role)
Declaration(Class(photo:production_date))
EquivalentClasses(photo:production_date ObjectIntersectionOf(ObjectSomeValuesFrom(photo:is_production_date_of__product photo:product) photo:time))
SubClassOf(photo:production_date photo:role)
Declaration(Class(photo:rating))
EquivalentClasses(photo:rating ObjectOneOf(photo:Low_rating photo:Top_rating photo:Average_rating))
SubClassOf(photo:rating ObjectExactCardinality(1 photo:is_given_by__agent photo:agent))
SubClassOf(photo:rating ObjectExactCardinality(1 photo:is_rating_of__product photo:product))
Declaration(Class(photo:replaceable_product))
EquivalentClasses(photo:replaceable_product ObjectIntersectionOf(ObjectSomeValuesFrom(photo:is_replaced_by__replacing_product photo:product) photo:product))
SubClassOf(photo:replaceable_product photo:role)
Declaration(Class(photo:replacing_product))
EquivalentClasses(photo:replacing_product ObjectIntersectionOf(ObjectSomeValuesFrom(photo:replaces__replaceable_product photo:product) photo:product))
SubClassOf(photo:replacing_product photo:role)
Declaration(Class(photo:role))
Declaration(Class(photo:time))
Declaration(Class(photo:top_rated_product))
SubClassOf(photo:top_rated_product photo:product_by_rating)

Declaration(ObjectProperty(photo:captured__photo))
SubObjectPropertyOf(photo:captured__photo photo:has_created__creative_product)
InverseObjectProperties(photo:is_captured_by__photographer photo:captured__photo)
ObjectPropertyDomain(photo:captured__photo photo:person)
ObjectPropertyRange(photo:captured__photo photo:photo)
Declaration(ObjectProperty(photo:consists_of__product_part))
EquivalentObjectProperties(photo:consists_of__product_part photo:contains__component)
InverseObjectProperties(photo:is_part_of__product photo:consists_of__product_part)
TransitiveObjectProperty(photo:consists_of__product_part)
ObjectPropertyDomain(photo:consists_of__product_part photo:product)
ObjectPropertyRange(photo:consists_of__product_part photo:product)
Declaration(ObjectProperty(photo:contains__battery))
SubObjectPropertyOf(photo:contains__battery photo:contains__component)
ObjectPropertyDomain(photo:contains__battery photo:photo_camera)
ObjectPropertyRange(photo:contains__battery photo:battery)
DisjointObjectProperties(photo:contains__battery photo:contains__flash photo:contains__lens photo:contains__memory_card photo:contains__photo_element)
Declaration(ObjectProperty(photo:contains__component))
EquivalentObjectProperties(photo:contains__component photo:consists_of__product_part)
SubObjectPropertyOf(photo:contains__component photo:partitive_object_property)
InverseObjectProperties(photo:is_contained_in__product photo:contains__component)
ObjectPropertyDomain(photo:contains__component photo:product)
ObjectPropertyRange(photo:contains__component photo:product)
AnnotationAssertion(photo:label_sbvr photo:contains__flash "photo_camera contains flash"@en)
SubObjectPropertyOf(photo:contains__flash photo:contains__component)
ObjectPropertyDomain(photo:contains__flash photo:photo_camera)
ObjectPropertyRange(photo:contains__flash photo:flash)
DisjointObjectProperties(photo:contains__battery photo:contains__flash photo:contains__lens photo:contains__memory_card photo:contains__photo_element)
Declaration(ObjectProperty(photo:contains__lens))
SubObjectPropertyOf(photo:contains__lens photo:contains__component)
ObjectPropertyDomain(photo:contains__lens photo:photo_camera)
ObjectPropertyRange(photo:contains__lens photo:lens)
DisjointObjectProperties(photo:contains__battery photo:contains__flash photo:contains__lens photo:contains__memory_card photo:contains__photo_element)
SubObjectPropertyOf(photo:contains__memory_card photo:contains__component)
ObjectPropertyDomain(photo:contains__memory_card photo:photo_camera)
ObjectPropertyRange(photo:contains__memory_card photo:memory_card)
DisjointObjectProperties(photo:contains__battery photo:contains__flash photo:contains__lens photo:contains__memory_card photo:contains__photo_element)
Declaration(ObjectProperty(photo:contains__photo_element))
SubObjectPropertyOf(photo:contains__photo_element photo:contains__component)
ObjectPropertyDomain(photo:contains__photo_element photo:photo_camera)
ObjectPropertyRange(photo:contains__photo_element photo:photo_element)
DisjointObjectProperties(photo:contains__battery photo:contains__flash photo:contains__lens photo:contains__memory_card photo:contains__photo_element)
Declaration(ObjectProperty(photo:has__accessory))
InverseObjectProperties(photo:is_accessory_of__product photo:has__accessory)
ObjectPropertyDomain(photo:has__accessory photo:product)
ObjectPropertyRange(photo:has__accessory photo:product)
Declaration(ObjectProperty(photo:has__camera_model))
FunctionalObjectProperty(photo:has__camera_model)
ObjectPropertyDomain(photo:has__camera_model photo:photo_camera)
ObjectPropertyRange(photo:has__camera_model photo:camera_model)
Declaration(ObjectProperty(photo:has__photographer))
ObjectPropertyDomain(photo:has__photographer ObjectUnionOf(photo:photo_camera photo:photo))
ObjectPropertyRange(photo:has__photographer photo:person)
Declaration(ObjectProperty(photo:has__production_date))

InverseObjectProperties(photo:has__production_date photo:is_production_date_of__product)
ObjectPropertyDomain(photo:has__production_date photo:product)
ObjectPropertyRange(photo:has__production_date photo:time)
Declaration(ObjectProperty(photo:has__rating))
InverseObjectProperties(photo:is_rating_of__product photo:has__rating)
ObjectPropertyDomain(photo:has__rating photo:product)
ObjectPropertyRange(photo:has__rating photo:rating)
Declaration(ObjectProperty(photo:has_bag))
SubObjectPropertyOf(photo:has_bag photo:has__accessory)
ObjectPropertyDomain(photo:has_bag photo:photo_camera)
ObjectPropertyRange(photo:has_bag photo:camera_bag)
Declaration(ObjectProperty(photo:has_created__creative_product))
SubObjectPropertyOf(photo:has_created__creative_product photo:has_produced__product)
ObjectPropertyDomain(photo:has_created__creative_product photo:agent)
ObjectPropertyRange(photo:has_created__creative_product photo:creative_product)
Declaration(ObjectProperty(photo:has_participated_in__photo_exhibition))
ObjectPropertyDomain(photo:has_participated_in__photo_exhibition photo:agent)
ObjectPropertyRange(photo:has_participated_in__photo_exhibition photo:photo_exhibition)
Declaration(ObjectProperty(photo:has_produced__product))
InverseObjectProperties(photo:is_produced_by__agent photo:has_produced__product)
InverseFunctionalObjectProperty(photo:has_produced__product)
ObjectPropertyDomain(photo:has_produced__product photo:agent)
ObjectPropertyRange(photo:has_produced__product photo:product)
Declaration(ObjectProperty(photo:is_accessory_of__product))
InverseObjectProperties(photo:is_accessory_of__product photo:has__accessory)
ObjectPropertyDomain(photo:is_accessory_of__product photo:product)
ObjectPropertyRange(photo:is_accessory_of__product photo:product)
Declaration(ObjectProperty(photo:is_captured_by__photographer))
InverseObjectProperties(photo:is_captured_by__photographer photo:captured__photo)
ObjectPropertyDomain(photo:is_captured_by__photographer photo:photo)
ObjectPropertyRange(photo:is_captured_by__photographer photo:person)
Declaration(ObjectProperty(photo:is_contained_in__product))
InverseObjectProperties(photo:is_contained_in__product photo:contains__component)
FunctionalObjectProperty(photo:is_contained_in__product)
ObjectPropertyDomain(photo:is_contained_in__product photo:product)
ObjectPropertyRange(photo:is_contained_in__product photo:product)
Declaration(ObjectProperty(photo:is_given_by__agent))
ObjectPropertyDomain(photo:is_given_by__agent photo:rating)
ObjectPropertyRange(photo:is_given_by__agent photo:agent)
Declaration(ObjectProperty(photo:is_part_of__product))
InverseObjectProperties(photo:is_part_of__product photo:consists_of__product_part)
TransitiveObjectProperty(photo:is_part_of__product)
ObjectPropertyDomain(photo:is_part_of__product photo:product)
ObjectPropertyRange(photo:is_part_of__product photo:product)
Declaration(ObjectProperty(photo:is_produced_by__agent))
InverseObjectProperties(photo:is_produced_by__agent photo:has_produced__product)
FunctionalObjectProperty(photo:is_produced_by__agent)
ObjectPropertyDomain(photo:is_produced_by__agent photo:product)
ObjectPropertyRange(photo:is_produced_by__agent photo:agent)
AnnotationAssertion(rdfs:label photo:is_produced_by__camera_maker "photo camera is produced by camera maker"@en)
AnnotationAssertion(photo:label_sbvr photo:is_produced_by__camera_maker "photo_camera is_produced_by camera_maker"@en)
SubObjectPropertyOf(photo:is_produced_by__camera_maker photo:is_produced_by__agent)
FunctionalObjectProperty(photo:is_produced_by__camera_maker)
ObjectPropertyDomain(photo:is_produced_by__camera_maker photo:photo_camera)
ObjectPropertyRange(photo:is_produced_by__camera_maker photo:commercial_organization)
Declaration(ObjectProperty(photo:is_production_date_of__product))

InverseObjectProperties(photo:has__production_date photo:is_production_date_of__product)
FunctionalObjectProperty(photo:is_production_date_of__product)
ObjectPropertyDomain(photo:is_production_date_of__product photo:time)
ObjectPropertyRange(photo:is_production_date_of__product photo:product)
Declaration(ObjectProperty(photo:is_rated_by__agent))
ObjectPropertyDomain(photo:is_rated_by__agent photo:product)
ObjectPropertyRange(photo:is_rated_by__agent photo:agent)
Declaration(ObjectProperty(photo:is_rated_in__rating_period))
ObjectPropertyDomain(photo:is_rated_in__rating_period photo:rating)
ObjectPropertyRange(photo:is_rated_in__rating_period photo:time)
Declaration(ObjectProperty(photo:is_rating_of__product))
InverseObjectProperties(photo:is_rating_of__product photo:has__rating)
ObjectPropertyDomain(photo:is_rating_of__product photo:rating)
ObjectPropertyRange(photo:is_rating_of__product photo:product)
Declaration(ObjectProperty(photo:is_replaced_by__replacing_product))
InverseObjectProperties(photo:replaces__replaceable_product photo:is_replaced_by__replacing_product)
ObjectPropertyDomain(photo:is_replaced_by__replacing_product photo:product)
ObjectPropertyRange(photo:is_replaced_by__replacing_product photo:product)
Declaration(ObjectProperty(photo:is_role_of))
InverseObjectProperties(photo:plays__role photo:is_role_of)
AnnotationAssertion(photo:label_sbvr photo:is_taken_by__photo_camera "photo is_taken_by photo_camera"@en)
AnnotationAssertion(rdfs:label photo:is_taken_by__photo_camera "photo is taken by photo camera"@en)
FunctionalObjectProperty(photo:is_taken_by__photo_camera)
ObjectPropertyDomain(photo:is_taken_by__photo_camera photo:photo)
ObjectPropertyRange(photo:is_taken_by__photo_camera photo:photo_camera)
Declaration(ObjectProperty(photo:is_used_by__photographer))
InverseObjectProperties(photo:is_used_by__photographer photo:uses__photo_camera)
ObjectPropertyDomain(photo:is_used_by__photographer photo:photo_camera)
ObjectPropertyRange(photo:is_used_by__photographer photo:person)
Declaration(ObjectProperty(photo:partitive_object_property))
IrreflexiveObjectProperty(photo:partitive_object_property)
ObjectPropertyDomain(photo:partitive_object_property owl:Thing)
ObjectPropertyRange(photo:partitive_object_property owl:Thing)
Declaration(ObjectProperty(photo:plays__role))
InverseObjectProperties(photo:plays__role photo:is_role_of)
Declaration(ObjectProperty(photo:received_by__photo_artist))
FunctionalObjectProperty(photo:received_by__photo_artist)
ObjectPropertyDomain(photo:received_by__photo_artist photo:award)
ObjectPropertyRange(photo:received_by__photo_artist photo:person)
Declaration(ObjectProperty(photo:received_for__photo))
ObjectPropertyDomain(photo:received_for__photo photo:award)
ObjectPropertyRange(photo:received_for__photo photo:photo)
Declaration(ObjectProperty(photo:received_in__photo_exhibition))
FunctionalObjectProperty(photo:received_in__photo_exhibition)
ObjectPropertyDomain(photo:received_in__photo_exhibition photo:award)
ObjectPropertyRange(photo:received_in__photo_exhibition photo:photo_exhibition)
Declaration(ObjectProperty(photo:replaces__replaceable_product))
InverseObjectProperties(photo:replaces__replaceable_product photo:is_replaced_by__replacing_product)
ObjectPropertyDomain(photo:replaces__replaceable_product photo:product)
ObjectPropertyRange(photo:replaces__replaceable_product photo:product)
InverseObjectProperties(photo:is_used_by__photographer photo:uses__photo_camera)
ObjectPropertyDomain(photo:uses__photo_camera photo:person)
ObjectPropertyRange(photo:uses__photo_camera photo:photo_camera)
Declaration(DataProperty(photo:bag_weight))

SubDataPropertyOf(photo:bag_weight photo:product_weight)
FunctionalDataProperty(photo:bag_weight)
DataPropertyDomain(photo:bag_weight photo:camera_bag)
DataPropertyRange(photo:bag_weight xsd:int)
Declaration(DataProperty(photo:camera_price))
SubDataPropertyOf(photo:camera_price photo:product_price)
FunctionalDataProperty(photo:camera_price)
DataPropertyDomain(photo:camera_price photo:photo_camera)
DataPropertyRange(photo:camera_price xsd:decimal)
Declaration(DataProperty(photo:camera_weight))
SubDataPropertyOf(photo:camera_weight photo:product_weight)
FunctionalDataProperty(photo:camera_weight)
DataPropertyDomain(photo:camera_weight photo:photo_camera)
DataPropertyRange(photo:camera_weight xsd:int)
Declaration(DataProperty(photo:feature))
DataPropertyDomain(photo:feature photo:product)
Declaration(DataProperty(photo:is_fast))
DataPropertyDomain(photo:is_fast photo:camera_model)
DataPropertyRange(photo:is_fast xsd:boolean)
Declaration(DataProperty(photo:is_professional))
DataPropertyDomain(photo:is_professional photo:camera_model)
DataPropertyRange(photo:is_professional xsd:boolean)
Declaration(DataProperty(photo:is_smart))
DataPropertyDomain(photo:is_smart photo:camera_model)
DataPropertyRange(photo:is_smart xsd:boolean)
Declaration(DataProperty(photo:prize_value))
FunctionalDataProperty(photo:prize_value)
DataPropertyDomain(photo:prize_value photo:award)
DataPropertyRange(photo:prize_value xsd:string)
Declaration(DataProperty(photo:product_number))
FunctionalDataProperty(photo:product_number)
DataPropertyDomain(photo:product_number photo:product)
DataPropertyRange(photo:product_number xsd:string)
Declaration(DataProperty(photo:product_price))
SubDataPropertyOf(photo:product_price photo:feature)
FunctionalDataProperty(photo:product_price)
DataPropertyDomain(photo:product_price photo:product)
DataPropertyRange(photo:product_price xsd:decimal)
Declaration(DataProperty(photo:product_weight))
SubDataPropertyOf(photo:product_weight photo:feature)
FunctionalDataProperty(photo:product_weight)
DataPropertyDomain(photo:product_weight photo:product)
DataPropertyRange(photo:product_weight xsd:int)
Declaration(DataProperty(photo:time_value))
FunctionalDataProperty(photo:time_value)
DataPropertyDomain(photo:time_value photo:time)
DataPropertyRange(photo:time_value xsd:dateTime)
Declaration(NamedIndividual(<http://semantika.lt/ns/photocameras#2013>))
ClassAssertion(photo:time <http://semantika.lt/ns/photocameras#2013>)
Declaration(NamedIndividual(<http://semantika.lt/ns/photocameras#2014_01_15>))
ClassAssertion(photo:time <http://semantika.lt/ns/photocameras#2014_01_15>)
Declaration(NamedIndividual(photo:ABCD))
ClassAssertion(photo:organization photo:ABCD)
Declaration(NamedIndividual(photo:Average_rating))
ClassAssertion(photo:rating photo:Average_rating)
Declaration(NamedIndividual(photo:Award1))
ClassAssertion(photo:award photo:Award1)
ObjectPropertyAssertion(photo:received_by__photo_artist photo:Award1 photo:John)

ObjectPropertyAssertion(photo:received_for__photo photo:Award1 photo:Photo_111)
ObjectPropertyAssertion(photo:received_in__photo_exhibition photo:Award1 photo:Photo_exhibition1)
Declaration(NamedIndividual(photo:Battery_1))
ClassAssertion(photo:battery photo:Battery_1)
Declaration(NamedIndividual(photo:COOLPIX_P340))
ClassAssertion(photo:camera_model photo:COOLPIX_P340)
Declaration(NamedIndividual(photo:COOLPIX_P600))
ClassAssertion(photo:camera_model photo:COOLPIX_P600)
Declaration(NamedIndividual(photo:Flash_1))
ClassAssertion(photo:flash photo:Flash_1)
Declaration(NamedIndividual(photo:G.Gudas))
ClassAssertion(photo:person photo:G.Gudas)
SameIndividual(photo:G.Gudas photo:Gytis_Gudas)
Declaration(NamedIndividual(photo:Gytis_Gudas))
ClassAssertion(photo:person photo:Gytis_Gudas)
SameIndividual(photo:Gytis_Gudas photo:G.Gudas)
Declaration(NamedIndividual(photo:Jim))
ClassAssertion(photo:person photo:Jim)
ObjectPropertyAssertion(photo:uses__photo_camera photo:Jim photo:Nicon_112)
Declaration(NamedIndividual(photo:John))
ClassAssertion(photo:person photo:John)
ObjectPropertyAssertion(photo:captured__photo photo:John photo:Photo_121)
ObjectPropertyAssertion(photo:captured__photo photo:John photo:Photo_131)
ObjectPropertyAssertion(photo:captured__photo photo:John photo:Photo_111)
ObjectPropertyAssertion(photo:has_participated_in__photo_exhibition photo:John photo:Photo_exhibition1)
ObjectPropertyAssertion(photo:uses__photo_camera photo:John photo:Nicon_D5100)
Declaration(NamedIndividual(photo:KTU))
ClassAssertion(photo:other_organization photo:KTU)
Declaration(NamedIndividual(photo:Lens_1))
ClassAssertion(photo:lens photo:Lens_1)
Declaration(NamedIndividual(photo:Low_rating))
ClassAssertion(photo:rating photo:Low_rating)
Declaration(NamedIndividual(photo:Memory_card_1))
ClassAssertion(photo:memory_card photo:Memory_card_1)
Declaration(NamedIndividual(photo:Mike))
ClassAssertion(photo:person photo:Mike)
ObjectPropertyAssertion(photo:captured__photo photo:Mike photo:Photo_211)
ObjectPropertyAssertion(photo:captured__photo photo:Mike photo:Photo_212)
ObjectPropertyAssertion(photo:uses__photo_camera photo:Mike photo:Nicon_113)
Declaration(NamedIndividual(photo:Nicon))
ClassAssertion(photo:commercial_organization photo:Nicon)
Declaration(NamedIndividual(photo:Nicon_112))
ClassAssertion(photo:digital_photo_camera photo:Nicon_112)
ObjectPropertyAssertion(photo:has__accessory photo:Nicon_112 photo:Nicon_bag)
ObjectPropertyAssertion(photo:has__camera_model photo:Nicon_112 photo:COOLPIX_P600)
ObjectPropertyAssertion(photo:has__production_date photo:Nicon_112 <http://semantika.lt/ns/photocameras#2014_01_15>)
ObjectPropertyAssertion(photo:is_produced_by__camera_maker photo:Nicon_112 photo:Nicon)
DataPropertyAssertion(photo:product_number photo:Nicon_112 "112"^^xsd:string)
Declaration(NamedIndividual(photo:Nicon_113))
ClassAssertion(photo:digital_photo_camera photo:Nicon_113)
ObjectPropertyAssertion(photo:has__accessory photo:Nicon_113 photo:Nicon_bag)
ObjectPropertyAssertion(photo:has__camera_model photo:Nicon_113 photo:COOLPIX_P600)
ObjectPropertyAssertion(photo:is_produced_by__camera_maker photo:Nicon_113 photo:Nicon)
DataPropertyAssertion(photo:camera_weight photo:Nicon_113 "300"^^xsd:int)
DataPropertyAssertion(photo:product_number photo:Nicon_113 "112"^^xsd:string)
Declaration(NamedIndividual(photo:Nicon_D5100))

ClassAssertion(photo:digital_photo_camera photo:Nicon_D5100)
ObjectPropertyAssertion(photo:has__accessory photo:Nicon_D5100 photo:Nicon_bag)
ObjectPropertyAssertion(photo:has__camera_model photo:Nicon_D5100 photo:COOLPIX_P600)
ObjectPropertyAssertion(photo:has__rating photo:Nicon_D5100 photo:Average_rating)
ObjectPropertyAssertion(photo:is_produced_by__camera_maker photo:Nicon_D5100 photo:Nicon)
NegativeObjectPropertyAssertion(photo:contains__photo_element photo:Nicon_D5100 photo:sensor2)
DataPropertyAssertion(photo:camera_weight photo:Nicon_D5100 "150"^^xsd:int)
NegativeDataPropertyAssertion(photo:product_number photo:Nicon_D5100 "151"^^xsd:string)
Declaration(NamedIndividual(photo:Nicon_bag))
ClassAssertion(photo:camera_bag photo:Nicon_bag)
Declaration(NamedIndividual(photo:Olympus))
ClassAssertion(photo:commercial_organization photo:Olympus)
Declaration(NamedIndividual(photo:Olympus_EPL5))
ClassAssertion(photo:film_photo_camera photo:Olympus_EPL5)
ObjectPropertyAssertion(photo:is_produced_by__camera_maker photo:Olympus_EPL5 photo:Olympus)
DataPropertyAssertion(photo:camera_weight photo:Olympus_EPL5 "1100"^^xsd:int)
Declaration(NamedIndividual(photo:Photo_111))
ClassAssertion(photo:photo photo:Photo_111)
Declaration(NamedIndividual(photo:Photo_121))
ClassAssertion(photo:photo photo:Photo_121)
ObjectPropertyAssertion(photo:has__photographer photo:Photo_121 photo:photographer)
Declaration(NamedIndividual(photo:Photo_131))
ClassAssertion(photo:photo photo:Photo_131)
Declaration(NamedIndividual(photo:Photo_211))
ClassAssertion(photo:photo photo:Photo_211)
Declaration(NamedIndividual(photo:Photo_212))
ClassAssertion(photo:photo photo:Photo_212)
Declaration(NamedIndividual(photo:Photo_exhibition1))
ClassAssertion(photo:photo_exhibition photo:Photo_exhibition1)
Declaration(NamedIndividual(photo:Price1))
Declaration(NamedIndividual(photo:Product1))
ClassAssertion(photo:product photo:Product1)
ObjectPropertyAssertion(photo:consists_of__product_part photo:Product1 photo:Product2)
Declaration(NamedIndividual(photo:Product2))
ClassAssertion(photo:product photo:Product2)
ObjectPropertyAssertion(photo:consists_of__product_part photo:Product2 photo:Product3)
Declaration(NamedIndividual(photo:Product3))
ClassAssertion(photo:product photo:Product3)
Declaration(NamedIndividual(photo:Sony))
ClassAssertion(photo:camera_model photo:Sony)
Declaration(NamedIndividual(photo:Top_rating))
ClassAssertion(photo:rating photo:Top_rating)
Declaration(NamedIndividual(photo:photographer))
ClassAssertion(photo:photographer photo:photographer)
Declaration(NamedIndividual(photo:sensor1))
ClassAssertion(photo:digital_sensor photo:sensor1)
Declaration(NamedIndividual(photo:sensor2))
ClassAssertion(photo:film_sensor photo:sensor2)
Declaration(NamedIndividual(photo:sensor3))
ClassAssertion(photo:digital_sensor photo:sensor3)
Declaration(AnnotationProperty(photo:label_sbvr))
DifferentIndividuals(photo:John photo:ABCD)
DifferentIndividuals(photo:Gytis_Gudas photo:ABCD)
HasKey(photo:photo_camera (photo:has__camera_model photo:is_produced_by__camera_maker) (photo:product_number))
SubObjectPropertyOf(ObjectPropertyChain(photo:captured__photo photo:has__photographer) photo:plays__role)

DifferentIndividuals(photo:COOLPIX_P600 photo:COOLPIX_P340)
SubObjectPropertyOf(ObjectPropertyChain(photo:uses__photo_camera photo:has__photographer) photo:plays__role)
DifferentIndividuals(photo:Olympus_EPL5 photo:Nicon_D5100)
DisjointClasses(photo:battery photo:flash photo:lens photo:memory_card photo:photo_camera photo:photo_element)
DifferentIndividuals(photo:John photo:Jim)
DifferentIndividuals(photo:G.Gudas photo:ABCD)
SubObjectPropertyOf(ObjectPropertyChain(photo:has__rating photo:is_given_by__agent) photo:is_rated_by__agent)
DisjointClasses(photo:creative_product photo:electronics_product photo:other_product)
)