Latvijas Universitāte
Datorikas fakultāte

JŪLIJA OVČIŅŅIKOVA

VIZUĀLI SEMANTISKO TEHNOLOĢIJU RĪKI

Promocijas darba kopsavilkums

Zinātnes doktora grāda Zinātnes Doktors (Ph. D.) Dabaszinātnēs
iegūšanai Datorzinātnes un informātikas nozarē
Apakšnozare: Programmēšanas valodas un sistēmas

Zinātniskais vadītājs:
Profesors, Dr. sc. comp.,
Kārlis Čerāns

Rīga, 2023

Promocijas darbs izstrādāts Latvijas Universitātes
Datorikas fakultātē un Latvijas Universitātes Matemātikas un informātikas institūtā, Sistēmu modelēšanas un programmatūras tehnoloģiju laboratorijā
laika posmā no 2014. gada līdz 2023. gadam

Eiropas Sociālā fonda projekts „Atbalsts doktora studijām Latvijas Universitātē ” Nr.2009/0138/ 1DP/1.1.2.1.2./ 09/IPIA/ VIAA/004.

Eiropas Sociālā fonda projekts “LU doktorantūras kapacitātes stiprināšana jaunā doktorantūras modeļa ietvarā” (Nr. 8.2.2.0/20/I/006)

Darbs sastāv no ievada, 2 nodaļām, nobeiguma, literatūras saraksta.

Darba forma: publikāciju kopa datorzinātņu nozarē, programmēšanas valodas un sistēmas apakšnozarē

Darba zinātniskais vadītājs: Profesors, Dr. sc. comp., Kārlis Čerāns

Darba recenzenti:

1) *Profesore, Dr.sc.comp., Laila Niedrīte, Latvijas Universitāte*

2) *Profesors, Dr.habil.sc.ing., Jānis Grundspeņķis, RTU; LZA īstenais loceklis*

3) *Tehniskās komandas vadītājs, Dr.sc.comp., Uģis Sarkans, Eiropas Bioinformātikas Institūts (EBI, Lielbritānija)*

Promocijas darba aizstāvēšana notiks 2023. gada 8. septembrī plkst. 15:00
Latvijas Universitātes Datorzinātnes un informātikas nozares promocijas padomes atklātā sēdē Latvijas Universitātes Matemātikas un Informātikas institūtā, Raiņa bulvārī 29, 413. telpā.

Ar promocijas darbu un tā kopsavilkumu var iepazīties Latvijas Universitātes Bibliotēkā Rīgā, Kalpaka bulvārī 4.

LU Datorzinātnes un informātikas zinātņu nozares promocijas

padomes priekšsēdētājs Guntis Bārzdiņš

promocijas padomes sekretāre Ruta Ikauniece

Anotācija

Darbā izstrādātas oriģinālas metodes, kas ļauj vizuālus uz paplašinātām UML veida grafu diagrammām balstītus rīkus izmantot praktisku ontoloģiju un semantisko datu vaicājumu veidošanai un attēlošanai.

OWL ontoloģiju vizuālas modelēšanas jomā darbā izveidoti līdzekļi konkrētam lietojumam vai lietojumu grupai specifiskas modelēšanas notācijas uzdošanai un izmantošanai. Šim nolūkam OWLGrEd rīka arhitektūras ietvarā tika izveidots mehānisms lietotāja definētu notāciju uzdošanai un attēlošanai, ontoloģiju vizualizācijas parametru ietvars, modulārs ontoloģiju eksporta modulis un uz gramatikām balstīta tekstuālo konstrukciju priekšāteikšanas metode. Izstrādātās metodes tika praktiski izmantotas Latvijas Uzņēmumu reģistra datu modelēšanā.

Darbā piedāvāts risinājums vizuālai bagātīgu datu vaicājumu (kas var ietvert apakšvaicājumus un agregācijas, kā arī atribūtu un nosacījumu izteiksmes) veidošanai pār RDF datubāzēm, un to translēšanai uz tekstuālu SPARQL valodu, kurā pierakstītie vaicājumi var tikt tieši izpildīti pār RDF datu bāzēm. Risinājumā tiek ietverts arī gramatikas atbalsts tekstuālu izteiksmju izmantošanai vizuālajā vaicājumu rīkā un izteiksmju teksta automātiska turpināšana, kas ir balstīta uz konkrēta datu avota shēmu un uz šo brīdi izveidotu vaicājuma fragmentu. Izstrādātās metodes ir izmantojamas vizuālu vaicājumu veidošanā pār nozīmīgām saistīto datu kopām, kā DBPedia, Wikidata un Europeana.

Praktiskās realizācijas veidotas LU MII izstrādātajās grafisko rīku platformās TDA un ajoo, redaktorā OWLGrEd un tā paplašinājumos, kā arī vizuālo vaicājumu rīkā ViziQuer.

Atslēgvārdi: OWL, OWLGrEd, teksta priekšāteicējs, domēnspecifiska ontoloģiju attēlošana, SPARQL, vizuāli vaicājumi, ViziQuer.

Izmantotie saīsinājumi

Saīsinājums	Termins angliski	Termins latviski
LU MII	The Institute of Mathematics and Computer Science, University of Latvia	Latvijas Universitātes Matemātikas un Informātikas institūts
W3C	World Wide Web Consortium	Vispasaules Tīmekļa konsorcijs
SQL	Structured Query Language	Strukturētu vaicājumu valoda
RDF	Resource Description Framework	Resursu aprakstīšanas ietvars
RDFS	Resource Description Framework Schema	Resursu aprakstīšanas ietvara shēma
IRI	Internationalized Resource Identifier	Starptautisks resursu identifikators
URI	Uniform Resource Identifier	Vienotais resursu identifikators
SHACL	Shapes Constraint Language	Formu ierobežojumu valoda
ShEx	Shape Expressions	Formu izteiksmes
OWL	Web Ontology Language	Tīmekļa ontoloģiju valoda
UML	Unified Modeling Language	Vienotā modelēšanas valoda
SPARQL	SPARQL Protocol and RDF Query Language	SPARQL protokols un RDF vaicājumu valoda
TDA	Transformation-Driven Architecture	Transformāciju vadīta arhitektūra
GrTP	Transformation Based Graphical Tool Building Platform	Uz transformācijām balstīta grafiskā rīku būves platforma
DSML	Domain-Specific Modeling Language	Domēnspecifiska modelēšanas valoda
OBIS	Ontology-Based Information System	Uz ontoloģijām balstīta informācijas sistēma
CNL	Controlled Natural Language	Kontrolēta dabiskā valoda
ACE	Attempto Controlled English	Attempto kontrolēta angļu valoda
JSON	JavaScript Object Notation	JavaScript objektu notācija

Saturs

Vispārīgs darba raksturojums

Darbā tika izstrādātas metodes, kas ļauj vizuālus uz grafiskām diagrammām balstītus rīkus izmantot praktisku ontoloģiju un semantisko datu vaicājumu veidošanai un attēlošanai.

Darbs tika izstrādāts, veicot pētījumus LU MII 11 gadu garumā, tostarp ERAF projektā Nr. 2011/0009/2DP/2.1.1.1.0/10/APIA/VIAA/112, ERAF projektā Nr. 2014/0020/2DP/2.1.1.1.0/14/APIA/VIAA/072, Valsts Pētījumu Programmas (2014-2017) NexIT projektā Nr.1 "Ontoloģiju, semantiskā tīmekļa un drošības tehnoloģijas", LZP projektā "Vizuāli uz ontoloģijām balstīti vaicājumi" (Nr. lzp-2020/2-0188), LZP projektā "Vizuāli vaicājumi dalītos zināšanu grafos" (Nr. lzp-2021/1-0389), projektā "LU doktorantūras kapacitātes stiprināšana jaunā doktorantūras modeļa ietvarā" (Nr. 8.2.2.0/20/I/006).

Promocijas darbs veidots kā 24 zinātnisku publikāciju kopums.

Pētījuma aktualitāte un novitāte

Mūsdienās semantiskās tehnoloģijas ir plaši izmantojamas daudzos lietojumos un rīkos. Semantiskais Tīmeklis (Berners-Lee et al., 2001) balstās uz atvērtiem saistītajiem datiem (WEB, h).

Saskaņā ar World Wide Web Consortium (W3C) (WEB, aa) semantiskais tīmeklis ir "vienots ietvars, kas ļauj koplietot un atkārtoti izmantot datus lietojumprogrammās, uzņēmumos un kopienas ietvaros". Semantiskās tehnoloģijas ļauj apstrādāt datus, gan tīmeklī esošus, gan organizācijā lokālus, izmantojot semantiskā tīmekļa standartus un principus.

Semantiskās tehnoloģijas piedāvā augstāka līmeņa skatu uz datiem nekā klasiskās relāciju datu bāzes ar SQL vaicājumu valodu. Datu modelis balstās uz jēdzieniem un attiecībām starp šiem jēdzieniem. Atsevišķu jēdzienu jēga tiek izteikta ar attiecībām, kas šim jēdzienam ir ar citiem jēdzieniem. Datu modelēšanas formālisms ļauj no vieniem faktiem izsecināt citus. Semantiskās tehnoloģijas ļauj veidot attiecības starp datiem, kas ir iegūti no dažādiem avotiem un atbilst dažādiem sākotnējiem formātiem, tādā veidā semantiskās tehnoloģijas ir labi piemērotas, lai vienviet integrētu no dažādiem avotiem nākošas zināšanas.

Semantiskās tehnoloģijas balstās uz datu attēlošanu RDF (Hayes and Patel-Schneider, 2014) trijnieku formātā. Informācija par datu struktūru parasti tiek attēlota RDF shēmas (RDF Schema (Brickley and Guha, 2014)) vai valodā OWL (Motik et al., 2012) uzdotas ontoloģijas veidā. Šīs tehnoloģijas ļauj pierakstīt dažāda veida (heterogēnas) zināšanas mašīnām apstrādājamā veidā.

Pēdējos gados popularitāti iegūst arī valodas SHACL (Shapes Constraint Language) (Knublauch and Kontokostas, 2017) un ShEx (Shape Expressions) (Prud'hommeaux et al., 2014) – RDF grafu struktūras aprakstīšanai un validācijai.

SPARQL (WEB, t) ir standarta vaicājumu valoda pār RDF datiem.

Attēlot ontoloģijas struktūru cilvēkam saprotamā formā ir svarīgi gan ontoloģiju izstrādātājiem, gan arī to lietotājiem. Gan vienā, gan otrā lomā ir svarīgi iesaistīt ne tikai datu modelēšanas un ontoloģiju speciālistus, bet arī konkrētās nozares speciālistus, kas var nebūt eksperti informācijas tehnoloģiju jomā, bet kuriem ir daudz vairāk informācijas par doto sfēru un nepieciešamo izmaiņu un uzlabojumu koncepciju.

Vizuāla prezentācija ir viens no ontoloģijas struktūras attēlošanas veidiem, kurā var būt atvieglota ontoloģijas uztvere, jo tajā loģiski saitīti jēdzieni (piemēram, klase un šīs klases instancēm tipiski piemītošās īpašības) tiek arī grafiski attēloti kopā. Prezentējot informāciju vizuāli, var tikt paplašināts arī to lietotāju loks, kas ar šo informāciju var strādāt tieši.

Plaši izplatīta modelēšanas notācija, kas tiek atbalstīta arī dažādos modelēšanas rīkos, ir UML (WEB, y), tomēr tās piedāvātais iespēju klāsts tipiski nav pietiekams visu ontoloģiju modelēšanas un attēlošanas uzdevumu risināšanai.

Eksistē virkne risinājumu OWL ontoloģiju vizualizācijai un/vai vizuālai pārvaldībai: OWLViz (WEB, p), VOWL (Lohmann et al., 2016), ODM (WEB, j), TopBraid Composer (WEB, x) un citi. To starpā ir arī OWLGrEd (Barzdins et al., 2010a) redaktors, kas attēlo OWL ontoloģijas UML klašu diagrammu veidā. OWLGrEd rīkā UML notācija ir bagātināta ar specifiskiem laukiem dažādu ontoloģiju

uzdošanas konstrukciju attēlošanai, piemēram, ekvivalentas klases, virsklases, nepārklājošas klases, un citi. OWLGrEd sniedz iespēju vizuāli attēlot visas būtiskākās OWL 2 ontoloģiju konstrukcijas.

Tomēr praktiskos ontoloģiju modelēšanas risinājumos, lai tie būtu ērti, ir nepieciešami ontoloģijas attēlošanas pamata vizuālos līdzekļus bagātināt ar lietojumam vai lietojumu grupai specifiskiem līdzekļiem (piemēram, noteiktu veidu anotāciju attēlošana specifiskā grafiskā veidā, ka arī dažāda veida rīka funkcionalitātes paplašinājumi) (Cerans et al., 2013, 2019d).

Lai atbalstītu lietojumiem specifiskas notācijas izveidi, UML valodā ir pieejams t.s. stereotipu mehānisms (WEB, y), kā arī ir pieejamas dažādas vizuālu domēnspecifisku modelēšanas rīku būves platformas, t.sk. Microsoft DSL Tools (Cook et al., 2007), Eclipse GMF (Almendros-Jiménez et al., 2009), Sirius (Viyovic et al., 2014), GrTP/TDA platforma (Barzdins et al., 2007), ajoo (Sprogis, 2016).

Darbā izveidotas lietojumiem specifiskas modeļu veidošanas iespējas, kas ir piemērojamas tieši OWL ontoloģiju vizuālas modelēšanas kontekstā, iespējami balstoties uz jau esošajām OWL ontoloģiju UML veida modelēšanas iespējām, kādas ir attīstītas, piemēram, redaktorā OWLGrEd.

Darbā piedāvātais OWLGrEd redaktora paplašināšanas mehānisms nodrošina lietotājam iespējas pievienot ontoloģijas attēlojumam gan jaunus strukturālus elementus, gan arī tos apkalpojošu funkcionalitāti (tostarp ontoloģijas konstrukciju importa un eksporta informāciju). Tādējādi tiek iegūts mehānisms, kas ir spēcīgāks nekā, piemēram, UML stereotipi, un šis mehānisms ir vispārināms arī uz citiem redaktoriem, kas veidoti GrTP/TDA vai līdzīgās rīku būves platformās (piemēram, ajoo).

Līdzās vizuālai ontoloģiju datu modelēšanai otra būtiska joma, kurā vizuālas UML veida diagrammas var paplašināt ar datiem tieši strādājošu personu loku, ir vaicājumu veidošana. Veidojot vaicājumus pār RDF formā strukturētiem datiem, var tikt iegūtas iespējas izgūt datus no RDF datubāzēm. Uz ontoloģijām balstīta datu piekļuve (OBDA) un tās atbalsta rīki, kā Ontop (Xiao et al., 2020), piedāvā iespējas semantiskos vaicājumus attiecināt arī pār datiem relāciju datubāzēs (šim nolūkam vēl nepieciešams datu atbilstības definējums).

Tomēr, līdzīgi kā SQL valoda relāciju datubāzēm, arī standarta tekstuālā SPARQL valoda RDF datubāzēm nav piemērota galalietotājiem bez specifiskām IT zināšanām (Soylu et al., 2017).

Pastāv dažādi rīki, kas paredzēti, lai galalietotājam palīdzētu vaicājumu veidošanā pār semantiskajām (RDF formāta) datubāzēm. Šādi rīki var balstīties uz datu atlases formām (piemēram, PepeSearch (Vega-Gorgojo et al., 2016), WYSIWYQ (Khalili and Merono-Penuela, 2017)), dabiskās valodas teksta fragmentiem (piemēram, SPARKLIS (Ferré, 2017)), kā arī uz vizuāli prezentētiem un veidotiem vaicājumiem. Pieejamie vizuālo vaicājumu rīki pār RDF datubāzēm ietver OptiqteVQS (Soylu et al., 2018), QueryVOWL (Haag et al., 2015) RDF Explorer (Vargas et al., 2019) un ViziQuer (agrīnās šī rīka versijas) (Barzdins et al., 2009).

Pieejamie vizuālo vaicājumu rīki atbalsta vienkāršu konjunktīvu vaicājumu uzdošanu, ietverot vienkāršus nosacījumus uz datu vērtībām. Šī darba ietvaros ViziQuer/web rīkā (Cerans et al., 2017, 2018b) ir izstrādātas iespējas būtiski paplašināt veidojamo vizuālo vaicājumu kopumu, ietverot tajā arī vaicājumus ar agregācijām, apakšvaicājumiem un datu vērtību izteiksmēm. Šāda notācija nodrošina iespēju, ka lietotājs, sākot darbu vizuālo vaicājumu vidē vispirms ar vienkāršiem vaicājumiem, varēs darbu tajā turpināt arī tad, kad vaicājumi kļūs sarežģītāki.

ViziQuer/web rīks balstās uz vaicājumu attēlošanu paplašinātas UML klašu diagrammas veidā, pievienojot klašu diagrammu pamata notācijai specifisku notāciju vaicājumu attēlošanai.

Vizuālo vaicājumu notācija aprobēta lietojamības testos ar dalībniekiem, kas nav semantisko tehnoloģiju speciālisti, kā arī veikta tās piemērošana vaicāšanai pār populāriem saistīto datu avotiem, piemēram, Europeana (WEB, c), kā arī Wikidata (Vrandečić and Krötzsch, 2014) un DBPedia (Auer et al., 2007).

Mērķi un uzdevumi

Pētījuma mērķis ir parādīt, ka:

paplašināta UML klašu diagrammu notācija un universālas rīku būves platformas ir piemērotas, lai veidotu rīkus darbam ar bagātīgiem semantisko datu modeļiem (ontoloģijām) un vaicājumiem pār strukturētiem semantiskajiem datiem.

Darba uzdevumi:

1. Ontoloģiju vizuālās modelēšanas jomā:
   a. Izpētīt literatūru par vizuālo rīku izmantošanu ontoloģiju vizuālās modelēšanas jomā.
   b. Izstrādāt risinājumu OWLGrEd ontoloģiju redaktora papildināšanai ar lietotāja definētu lauku un skatījumu mehānismu, kas ļauj īpašā vizuālā veidā attēlot un specifiski apstrādāt noteiktu aksiomu (t.sk. anotāciju) informāciju.
   c. Izstrādāt paplašināmu ontoloģiju importa un vizualizācijas parametru ietvaru.
   d. Izstrādāt uz šabloniem balstītu modulāru eksporta risinājumu, kas izmantojams OWLGrEd rīkā un tā paplašinājumos.
   e. Izstrādāt uz gramatikām balstītu metodi teksta turpinājumu priekšāteikšanai, kas piemērota izmantošanai GrTP/TDA un ajoo platformās.
   f. Izstrādāt un aprobēt konkrētus OWLGrEd redaktora paplašinājumus.

2. Vizuālo vaicājumu jomā:
   a. Izpētīt literatūru par vizuālo rīku izmantošanu vizuālo vaicājumu jomā.
   b. Izstrādāt risinājumus vizuālu vaicājumu pār RDF datubāzēm tulkošanai uz izpildāmu SPARQL notāciju.
   c. Izveidot gramatiku atbalstu tekstuālo izteiksmju izmantošanai vizuālajā vaicājumu rīkā.
   d. Izstrādāt risinājumus izteiksmju teksta automātiskai turpināšanai, balstoties uz izmantoto datu avota shēmu un uz šo brīdi izveidotu vaicājuma fragmentu.
   e. Veikt izstrādāto risinājumu pielāgošanu tīmeklī pieejamiem saistīto datu avotiem un šo risinājumu lietojamības novērtēšanu.

Tēzes

Darba ietvaros praktiski pārbaudītas šādas tēzes:

1. Paplašināta UML klašu diagrammu struktūra ir piemērota bagātīgu semantisko datu modeļu veidošanas un vaicājumu rīku izveidei.
2. Interpretējamas modelēšanas rīku būves platformas, kas uztur konfigurācijas informāciju vienā konceptuālā līmenī ar izpildes laika instanču informāciju (piemēram, platformas GrTP/TDA un ajoo), var tikt veiksmīgi lietotas rīku ar sarežģītu biznesa loģiku veidošanai.
3. Uz gramatikām balstīti servisi nodrošina iespējas atbalstīt strukturētu teksta fragmentu izmantošanu rīkos veidoto datu modeļu kontekstā:
   a. Eksperimentāla priekšāteikšana OWLGrEd rīkā
   b. OWLGrEd un tā paplašinājumu eksporta gramatika
   c. ViziQuer/web izteiksmju analīze
   d. ViziQuer/web atribūtu un nosacījumu izteiksmju priekšāteikšana
4. Konfigurējams paplašinājumu mehānisms ir noderīgs vizuālu modelēšanas risinājumu izveidei uz esošu rīku bāzes.

Promocijas darba zinātniskā un praktiskā nozīmība

1. tēze. Paplašināta UML klašu diagrammu struktūra ir piemērota bagātīgu semantisko datu modeļu veidošanas un vaicājumu rīku izveidei.

UML klašu diagrammas ir plaši izplatīta modelēšanas notācija, tomēr tās piedāvāto iespēju klāsts tipiski ir nepietiekams bagātīgu ontoloģiju un vaicājumu uzdošanai.

Ontoloģiju definēšanā ODM rīks (WEB, j) piedāvā UML profilu OWL ontoloģiju uzdošanai, kas, paliekot UML specifikācijas ietvaros, nespēj piedāvāt notāciju, kas būtu līdzīgi kompakta un pārskatāma, kā pašas UML klašu diagrammas. TopBraid Composer (WEB, y) piedāvā vizuālu ontoloģiju attēlošanas un rediģēšanas vidi, tostarp izmantojot UML modelēšanas principus, tas ir komerciāls rīks, kas nav pieejams daļai potenciālo lietotāju. OWLGrEd redaktora versijas (pirms darba

autores ieguldījuma) (Barzdins et al., 2010a) paplašina UML notāciju, ieviešot virkni lauku specifisku ontoloģiju konstrukciju modelēšanai.

Minētie rīki, kaut arī pārliecinoši iezīmē nepieciešamību un iespējamību izmantot paplašinātas UML klašu diagrammas OWL ontoloģiju uzdošanai, galalietotājam neatstāj nekādas vai atstāj minimālas iespējas veidot savas lietojuma vai datu konteksta noteiktas ontoloģiju vizuālās prezentācijas iespējas.

Šis darbs paplašina ontoloģiju vizuālajā apstrādē izmantojamu vizuālo primitīvu kopumu, tajā skaitā atļaujot arī galalietotāja definētus vizuālās prezentācijas likumus.

Semantisko vaicājumu uzdošanā OptiqueVQS rīks (Soylu et al., 2018) piedāvā vizuālu diagrammu vaicājuma struktūras uzdošanai (tipiski tiek atbalstīti vienkārši konjunktīvi vaicājumi, vai konjunktīvi vaicājumi ar ārēju agregāciju), vaicājuma diagrammai pakārtotās daļas darot pieejamas lietotājam tikai interaktīvajā saskarnē (diagrammā tās nav redzamas). Agrīnās ViziQuer rīka versijas (Barzdins et al., 2009) arī piedāvā vaicājumu veidošanas iespējas UML klašu diagrammu veidā, tomēr, aprobežojoties ar vienkāršiem vaicājumiem, kas neietver agregācijas, apakšvaicājumu un sarežģītu datu izteiksmju veidošanas iespējas.

Šajā darbā ir demonstrēta paplašinātu UML klašu diagrammu izmantošanas iespēja bagātīgu vizuālu vaicājumu uzdošanā, kas ietver agregācijas un apakšvaicājumus, kā arī bagātīgu tekstuālu valodu atribūtu vērtību un nosacījumu uzdošanai.

2. tēze. Interpretējamas modelēšanas rīku būves platformas, kas uztur konfigurācijas informāciju vienā konceptuālā līmenī ar izpildes laika instanču informāciju (piemēram, platformas GrTP/TDA un ajoo), var tikt sekmīgi lietotas rīku ar sarežģītu biznesa loģiku veidošanai.

Uz metamodeļiem balstītas rīku būves platformas ir vienkārši izmantojamas modelēšanas rīku konfigurēšanā līdz brīdim, kamēr platforma tieši atbalsta visas rīkā paredzētās iespējas, tajā skaitā, rīks ir iegūstams tikai ar konfigurēšanas palīdzību. Rīku būves platformas tipiski piedāvā arī iespēju pievienot rīkam specifisku funkcionalitāti, izmantojot paplašināšanas punktu mehānismu, tomēr šādas funkcionalitātes sekmīgas izmantošanas iespēja ir mazāk skaidra.

Darba ietvaros izstrādātā sistēma jaunu lauku pievienošanai un simbolu vizuālo efektu pārvaldībai ontoloģiju redaktorā OWLGrEd, kā arī teksta turpinājumu priekšāteikšanas funkcionalitātes izstrāde ViziQuer rīkā, parāda, ka GrTP/TDA (Barzdins et al., 2007) un ajoo (Sprogis, 2016) platformas var sekmīgi pievienot bagātīgu/sarežģītu darbības loģiku.

3. tēze. Uz gramatikām balstīti servisi nodrošina iespējas atbalstīt strukturētu teksta fragmentu izmantošanu rīkos veidoto datu modeļu kontekstā.

Lai gan vizuālās modelēšanas metodes atvieglo informācijas uztveramību, grafisko attēlošanas veidu nevar pielietot pilnīgi visām konstrukcijām. Balstoties uz datu apjomu un to struktūru, dažās situācijās labāk būtu pielietot tekstuālo attēlošanas veidu.

Svarīga ir arī iespēja eksportēt semantiskos datus no vizuālas notācijas uz tekstuālu formātu. Rīkiem, kas veidoti uz metamodeļiem balstītas rīku būves platformās, ir noteikta iekšējā struktūra un tekstuālo izteiksmju notācija. Tas dod iespēju izmantot uz gramatikām balstītos servisus eksportējamo tekstuālo izteiksmju veidošanai un apstrādei.

Darba ietvaros izveidoti uz gramatikām balstīti servisi tekstuālo izteiksmju analīzes un priekšāteikšanas funkcionalitātei ViziQuer un OWLGrEd rīkos, kā arī izveidota uz gramatikas balstīta eksporta šablonu valoda OWLGrEd rīka pamata funkcionalitātes un paplašinājumu notāciju saglabāšanai tekstuālā formātā.

4. tēze. Konfigurējams paplašinājumu mehānisms ir noderīgs dažādu vizuālu modelēšanas risinājumu izveidei uz esošu rīku bāzes.

Semantiskajās tehnoloģijās ir tipiska situācija, ka ir parādījies jauns lietojums, kuru pilnībā nepārklāj esoša rīka notācija un rodas vajadzība paplašināt rīku ar lietojumam specifisku funkcionalitāti. Tipiski šajā situācijā tiek veidots jauns rīks, par pamatu ņemot esošo un papildinot to. Līdz ar to pieaug rīku skaits ar līdzīgu pamata funkcionalitāti, kur katrs rīks ir pielāgots vienam specifiskam lietojumam. Lai novērstu šādu rīku skaita pieaugumu, ir noderīga rīka paplašinājumu veidošanas iespēja.

Darba ietvaros izstrādāti OWLGrEd rīka paplašinājumi dažādu vizuālu modelēšanas risinājumu izveidei parāda, ka paplašinājumu mehānisms ir noderīgs rīka domēnspecifiskas notācijas veidošanai un papildināšanai, vienlaikus saglabājot rīka pamata konfigurāciju un ļaujot savstarpēji kombinēt dažādus paplašinājumus.

Pētījuma metodes

Promocijas darbā tika izmantotas šādas pētījumu metodes:

- Zinātniskās literatūras un radniecīgu tehnoloģisko risinājumu izpēte esošās situācijas, pastāvošo problēmu un jau esošo risināšanas pieeju apzināšanai.
- Pētniecisko hipotēžu izvirzīšana saistībā ar iespējamiem pastāvošo problēmu risināšanas virzieniem.
- Pētniecisko hipotēžu pierādīšanai nepieciešamo programmatūras prototipu izstrāde, ietverot programmēšanas valodas izvēli (Lua, Javascript, Java), datu struktūras definējumu, algoritmu un programmas koda izstrādi un testēšanu, risinājuma sākotnējo lietojamības novērtēšanu un moduļu funkcionalitātes demonstrāciju.
- Prototipu analīze, salīdzinot tos ar eksistējošiem risinājumiem un lietotāju vajadzībām, izmantojot lietojamības pētījumus.
- Zinātnisku publikāciju gatavošana pētniecisko hipotēžu pierādījumu un izstrādāto programmēšanas prototipu demonstrēšanai.

Darba rezultāti

Praktiskai lietojamībai darbam ar ontoloģijas redaktoru, tostarp dažādiem ontoloģijas veidošanas uzdevumiem, šajā darbā ir izveidotas šādas svarīgas atbalsta struktūras:

- Lietotāja definēto lauku un skatījumu mehānisms – domēnspecifisku ontoloģiju attēlošanas notāciju uzdošanai.
- Ontoloģiju vizualizācijas parametru ietvars un eksporta modeļa risinājumi – domēnspecifisko paplašinājumu ielasīšanai/attēlošanai ontoloģijā un to saglabāšanai kādā no OWL formātiem.
- Uz gramatikām balstīta priekšāteikšanas metode.
- Konkrēti uz lietojumiem orientēti papildinājumi, t.sk. paplašinājums darbam ar Latvijas Uzņēmumu reģistra datu modeli.

Vizuālo vaicājumu jomā šajā darbā tiek:

- Piedāvāts risinājums vizuālai bagātīgu datu vaicājumu formulēšanai pār RDF datubāzēm, un to tulkošanai uz SPARQL (WEB, t) standarta valodu.
- Izveidots gramatiku atbalsts tekstuālo izteiksmju izmantošanai gan no izteiksmju analīzes, gan teksta turpinājumu priekšāteikšanas viedokļa.
- Veikti lietojamības pētījumi, kas parāda, ka noteiktās situācijās piedāvātajai notācijai ir priekšrocības vaicājumu pār RDF datiem veidošanā salīdzinājumā ar SPARQL valodu.

Aprobācija/publikācijas

Darba pētījumu rezultāti ir publicēti 24 zinātniskos rakstos, no kuriem 21 ir iekļauts SCOPUS datubāzē (katrai publikācijai tālāk ir norādītas atbilstošās tēzes un darba uzdevumi, uz ko pierādīšanu vai risināšanu publikācija attiecas, kā arī darba autores ieguldījuma apraksts):

Publikācija	Tēzes Nr.	Uzdevuma Nr.	Autores ieguldījums	Ieguldījuma apraksts
K. Cerans, R. Liepins, A. Sprogis, J. Ovcinnikova, G. Barzdins. Domain-Specific OWL Ontology Visualization with OWLGrEd, Lecture Notes in Computer Science (subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). 2015. Vol. 7540, pp. 419-424 (2015): The Semantic Web: ESWC 2012 Satellite Events (Heraklion, Greece; 27-31 May 2012) **[SCOPUS]**	1, 4	1a	40%	OWLGrEd paplašinājuma izveide lietotāja definēto lauku atbalstam. Līdzdalība publikācijas teksta veidošanā.
K. Cerans, G. Barzdins, R. Liepins, J. Ovcinnikova, S. Rikacovs and A. Sprogis, Graphical Schema Editing for Stardog OWL/RDF Databases using OWLGrEd/S // Proc. of OWLED 2012, Heraklion, Greece, 27-28 May 2012. CEUR-WS Vol. 849, p. 8 **[SCOPUS]**	4	1a	20%	OWLGrEd lietotāja definēto lauku mehānisma paplašināšana. Konkrēto notāciju atbalstoša lietotāja definēto lauku profila izveide. Līdzdalība publikācijas teksta veidošanā.
K. Čerāns, R. Liepiņš, J. Ovčiņņikova, and A. Sproģis. Advanced OWL 2.0 Ontology Visualization in OWLGrEd. In: A. Caplinskas et al. (Eds.). Frontiers of AI and Applications, Vol. 249, pp. 41-54, Databases and Information Systems VII, IOS Press, 2013. **[SCOPUS]**	1, 2, 4	1a, 1e	50%	OWLGrEd paplašinājuma lietotāja definēto lauku atbalstam un konkrēto lauku profilu izveide un validācija. Līdzdalība publikācijas teksta veidošanā.
K. Čerāns, G. Bārzdiņš, G. Būmans, J. Ovčiņņikova, S. Rikačovs, A. Romāne. On Semantic Re-Engineering of Relational Databases. H. Haav et al. (Eds.) Databases and Information Systems. Proceeding of the 11th International Baltic Conference on DB and IS, 8-11 June 2014, Tallinn, ESTONIA	1, 4	1e	20%	OWLGrEd Relāciju datu bāzes piesaistes un Informatīvā sistēmas bez programmēšanas lietotāju definēto lauku profilu izveide. Līdzdalība publikācijas teksta veidošanā.
K. Čerāns, G. Bārzdinš, G. Būmans, J. Ovčiņņikova, S. Rikačovs, A. Romāne. A Relational Database Semantic Re-Engineering Technology and Tools. Baltic J. Modern Computing, Vol. 2 (2014), No. 3, pp. 183-198	1, 4	1e	20%	OWLGrEd Relāciju datu bāzes piesaistes un Informatīvā sistēmas bez programmēšanas lietotāju definēto lauku profilu izveide. Līdzdalība publikācijas teksta veidošanā.
J. Ovčiņņikova, K. Čerāns, R. Liepiņš. Grammar based content completion method using Lua LPeg.re module.//Proc. Of 2014 IEEE 2nd Workshop on Advances in Information,	3	1d, 2c	70%	Priekšā-teikšanas metodes izstrāde un realizācija. Vadošā loma publikācijas teksta veidošanā.

Electronic and Electrical Engineering (AIEEE), Vilnius, Lithuania, 28-29 November, IEEE Xplore Digital Library, 2014. **[SCOPUS]**				
K. Cerans, J. Ovcinnikova, M. Zviedris. Towards Graphical Query Notation for Semantic Databases. In: Matulevičius, R., Dumas, M. (eds) Perspectives in Business Informatics Research. BIR 2015. Lecture Notes in Business Information Processing, Vol. 229. Springer, Cham. pp. 273-281. **[SCOPUS]**	1	2a	40%	Agregāciju un apakšvaicājumu sākotnējā ieviešana ViziQuer Desktop rīkā. Līdzdalība publikācijas teksta veidošanā
K. Cerans, J. Ovcinnikova, M. Zviedris. SPARQL Aggregate Queries made easy with Diagrammatic Query Language ViziQuer. In ISWC 2015 Posters & Demonstrations Track, CEUR Workshop Proceedings, Vol. 1486 **[SCOPUS]**	1	2a	40%	Agregāciju un stereotipu sākotnējā ieviešana ViziQuer Desktop rīkā. Līdzdalība publikācijas teksta veidošanā.
K. Cerans, J. Ovcinnikova. ViziQuer: Notation and Tool for Data Analysis SPARQL Queries. In Voila 2016, CEUR Workshop Proceedings, Vol. 1704, pp. 151-159 **[SCOPUS]**	1	2a	50%	Pamata notācijas un stereotipu izveide ViziQuer Desktop rīkā. Līdzdalība publikācijas teksta veidošanā.
J. Ovčiņņikova and K. Čerāns, Advanced UML Style Visualization of OWL Ontologies, in Proc. of VOILA 2016. CEUR, Vol. 1704, CEUR-WS.org, 2016, pp. 136-142. **[SCOPUS]**	4	1b, 1d, 1e	70%	OWLGrEd Ontoloģiju vizualizācijas parametru izstrāde, RefactoringServices paplašinājuma izstrāde. Līdzdalība publikācijas teksta veidošanā.
R. Liepiņš, N. Grūzītis, K. Čerāns, J. Ovčiņņikova, U. Bojārs, E. Celms. Adding Verbalization to Graphical Ontology Editor OWLGrEd. In: G. Arnicans et al. (Eds.). Frontiers in Artificial Intelligence and Applications, Vol. 291, Databases and Information Systems IX, IOS Press, pp. 17-30, 2016 **[SCOPUS]**	4	1e	20%	OWLGrEd kontrolētas dabiskās valodas lauku paplašinājuma vizuālās saskarnes komponenšu izstrāde. Līdzdalība publikācijas teksta veidošanā.
K. Cerans, J. Barzdins, A. Sostaks, J. Ovcinnikova, L. Lace, M. Grasmanis and A. Sprogis. Extended UML Class Diagram Constructs for Visual SPARQL Queries in ViziQuer/web In Voila 2017, CEUR Workshop Proceedings, Vol. 1947, pp. 87-98 **[SCOPUS]**	1	2a	20%	Pamata notācijas izveidošana ViziQuer/web rīkā. Līdzdalība lietojamības pētījuma rezultātu analīzē un publikācijas teksta veidošanā.
K. Cerans, A. Sostaks, U. Bojars, J. Ovcinnikova, L. Lace, M. Grasmanis, A. Romane, A. Sprogis and J. Barzdins.	1, 2, 3	2a, 2d	30%	ViziQuer grafisko vaicājumu translācijas metode uz izpildāmu

ViziQuer: a Visual Notation for RDF Data Analysis Queries. In Garoufallou, E., Sartori, F., Siatri, R., Zervas, M. (eds) Metadata and Semantic Research. MTSR 2018. Communications in Computer and Information Science, Vol. 846. Springer, Cham, pp. 50-62. **[SCOPUS]**				SPARQL tekstu. Līdzdalība lietojamības pētījuma rezultātu analīzē un publikācijas teksta veidošanā.
K. Cerans, A. Sostaks, U. Bojars, J. Ovcinnikova, L. Lace, M. Grasmanis, A. Romane, A. Sprogis and J. Barzdins. ViziQuer: a Web-based Tool for Visual Diagrammatic Queries over RDF Data In The Semantic Web: ESWC 2018 Satellite Events, Springer Verlag Lecture Notes in Computer Science, Vol. 11155, pp. 158-163. Springer, Cham. **[SCOPUS]**	1	2a, 2b, 2c. 2d	30%	ViziQuer grafisko vaicājumu translācijas metode uz izpildāmu SPARQL tekstu. Līdzdalība publikācijas teksta veidošanā.
K. Čerāns, J. Ovcinnikova, R. Liepiņš and M. Grasmanis. Extensible Visualizations of Ontologies in OWLGrEd. In The Semantic Web: ESWC 2019 Satellite Events, ESWC 2019. Lecture Notes in Computer Science, Vol. 11762, pp. 191-196. Springer, Cham. 2-6 June, Portorož, Slovenia, 2019 **[SCOPUS]**	2, 4	1a, 1e	40%	OWLGrEd paplašinājumu izstrāde. Līdzdalība publikācijas teksta veidošanā.
K. Čerāns, J. Ovčiņņikova, L. Lāce, J. Hodakovska, A. Romāne, M. Grasmanis, E. Kalniņa, A. Sproģis, A. Šostaks. Visual Query Environment over RDF Data. In Posters and Demos at SEMANTiCS 2019. CEUR-WS volume 2451. 9-12 September, Karlsruhe, Germany, 2019 **[SCOPUS]**	1	2a, 2d	30%	ViziQuer grafisko vaicājumu translācijas metode uz izpildāmu SPARQL tekstu. Priekšā-teikšanas metodes ieviešana ViziQuer rīkā. Līdzdalība publikācijas teksta veidošanā.
K. Čerāns, L. Lace, A. Romane, J. Ovcinnikova, S. Kozlovics, M. Grasmanis, J. Hodakovska, A. Sprogis and A. Sostaks. Visual Queries over Scholarly Data and other Linked Data Endpoints. In ISWC Satellites 2019. CEUR-WS volume 2456, pp. 297-300. 26-30 October, New Zealand, 2019 **[SCOPUS]**	1	2d	20%	ViziQuer grafisko vaicājumu translācijas metode uz izpildāmu SPARQL tekstu. Līdzdalība publikācijas teksta veidošanā.
K. Čerāns, L. Lāce, A. Romāne, J. Ovčiņņikova, M. Grasmanis, A. Sproģis, A. Šostaks. Schema-Based Visual Queries over Linked Data Endpoints. In: Garoufallou E., Fallucchi F., William De Luca E. (eds) Metadata and Semantic Research. MTSR 2019.	1	2a	30%	ViziQuer grafisko vaicājumu translācijas metodes uz izpildāmu SPARQL tekstu izstrāde. Līdzdalība publikācijas teksta veidošanā.

Communications in Computer and Information Science, Vol. 1057, pp. 200-206. Springer, Cham. 28-31 October, Rome, Italy, 2019 **[SCOPUS]**				
J. Ovčiņņikova. Ontology Export Patterns in OWLGrEd Editor. Baltic J. Modern Computing, Vol. 8(2020), No. 3, 444-460, **[SCOPUS]**	3	1c	100%	OWLGrEd rīka uz šabloniem balstīta modulāra eksporta risinājuma izstrāde. Publikācijas teksta veidošana.
K. Čerāns, J. Ovčiņņikova, U. Bojārs, M. Grasmanis, L. Lāce, A. Romāne. Schema-Backed Visual Queries over Europeana and other Linked Data Resources. The Semantic Web: ESWC 2021 Satellite Events. ESWC 2021. Lecture Notes in Computer Science, Vol. 12739, pp. 82–87. Springer, Cham. **[SCOPUS]**	1	2a, 2d	20%	ViziQuer grafisku vaicājumu translācijas metode uz izpildāmu SPARQL tekstu. Līdzdalība publikācijas teksta veidošanā.
K. Čerāns, L. Lāce, M. Grasmanis, J. Ovčiņņikova. A UML-Style Visual Query Environment Over DBPedia. Metadata and Semantic Research : 15th International Conference, MTSR 2021, Virtual Event, 29 November – 3 December 2021: Proceedings. Communications in Computer and Information Science; Vol. 1537 CCIS. Springer: Cham, 2022 pp. 16-27. **[SCOPUS]**	1	2a, 2c, 2d	30%	Priekšā-teikšanas metodes izveide ViziQuer rīkā. Lietotājiem draudzīgu formu izveide vaicājuma elementu pievienošanai. DBPedia vaicājumu testa kopas sagatavošana. Līdzdalība publikācijas teksta veidošanā.
K. Čerāns, J. Ovčiņņikova, M. Grasmanis, L. Lāce. Towards UML-Style Visual Queries over Wikidata (Posters and Demos) // The Semantic Web: ESWC 2022 Satellite Events, Hersonissos, Crete, Greece, 29 May – 2 June 2022: Proceedings / P. Groth, et al. (Eds.). Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics); Vol. 13384. Cham: Springer, 2022 pp. 11-15.. **[SCOPUS]**	1	2a, 2d	40%	ViziQuer grafisko vaicājumu translācijas metode uz izpildāmu SPARQL tekstu. Wikidata vaicājumu testa kopas sagatavošana. Līdzdalība publikācijas teksta veidošanā.
K. Čerāns, J. Ovčiņņikova, M. Grasmanis, L. Lāce. Experience with Visual SPARQL Queries over DBPedia. In Voila 2022, CEUR Workshop Proceedings, Vol. 3253, pp. 59-65. Visualization and Interaction for Ontologies and Linked Data (Voila 2022) 23 October 2022, Virtual Workshop **[SCOPUS]**	1	2d	40%	ViziQuer grafisko vaicājumu translācijas metode uz izpildāmu SPARQL tekstu. DBPedia vaicājumu testa kopas sagatavošana. Līdzdalība publikācijas teksta veidošanā.

				Lietojamības pētījuma rezultātu analīze.
J. Ovčiņņikova, A. Šostaks, K. Čerāns. Visual Diagrammatic Queries in ViziQuer: Overview and Implementation. Baltic J. Modern Computing, Vol. 11 (2023), No. 2, 317-350 Pieejams arī arXiv arXiv:2304.14825 [cs.DB].	2, 3	2a, 2b, 2c	50%	ViziQuer grafisko vaicājumu translācijas metode uz izpildāmu SPARQL tekstu. Līdzdalība publikācijas teksta veidošanā.

Pētījumu rezultāti tika prezentēti 20 starptautiskās konferencēs, sešās konferencēs pētījumu rezultātus prezentēja autore.

- 9th OWL: Experiences and Directions Workshop (OWLED 2012) Heraklion, Crete, 27-28 May 2012
- **9th Extended Semantic Web Conference (ESWC), Heraklion, Crete, 27-31 May 2012**
- **10th International Baltic Conference on Databases and Information Systems, DB&IS2012, Vilnius, Lithuania, 8-11 July 2012**
- **11th International Baltic Conference on Databases and Information Systems, DB&IS2014, Tallinn, ESTONIA, 8-11 July 2014**
- **2014 IEEE 2nd Workshop on Advances in Information, Electronic and Electrical Engineering (AIEEE), Vilnius, Lithuania, 28-29 November 2014**
- 4th International Conference on Perspectives in Business Informatics Research, BIR 2015, Tartu, Estonia, 26-28 August 2015
- The 14th International Semantic Web Conference, Bethlehem, Pennsylvania, 11-15 October 2015
- 12th International Baltic Conference on Databases and Information Systems, DB&IS2016, Riga, Latvia, 4-6 July 2016
- **Visualization and Interaction for Ontologies and Linked Data. 2nd International Workshop co-located with ISWC 2016, Kobe, Japan, 17 October 2016**
- Visualization and Interaction for Ontologies and Linked Data. 3rd International Workshop co-located with ISWC 2017, Vienna, Austria, 22 October 2017
- 15th Extended Semantic Web Conference (ESWC 2018), Heraklion, Crete, 3-7 June 2018
- 12th International Conference on Metadata and Semantics Research Limassol, Cyprus, 23-26 October 2018
- 16th Extended Semantic Web Conference (ESWC 2019), Portorož, Slovenia, 2-6 June 2019
- 15th International Conference, SEMANTiCS 2019, Karlsruhe, Germany, 9-12 September 2019
- 18th International Semantic Web Conference (ISWC 2019) Auckland, New Zealand, 26-30 October 2019
- **13th International Conference on Metadata and Semantics Research Rome (MTSR 2019), Italy, 28-31 October 2019**
- Extended Semantic Web Conference (ESWC 2021), Heraklion, Crete, Greece, 6-10 June 2021
- 15th International Conference on Metadata and Semantics Research Rome (MTSR 2021), Italy, 29 November – 3 December 2021
- Extended Semantic Web Conference (ESWC 2022), Hersonissos, Crete, Greece, 29 May – 2 June 2022
- Visualization and Interaction for Ontologies and Linked Data. 7th International Workshop co-located with ISWC 2022, 23 October, Virtual Workshop

Pētījumu rezultāti tika prezentēti sešās vietēja līmeņa konferencēs, divās konferencēs pētījumu rezultātus prezentēja autore.

- **70. LU konference, "Datorzinātņu un informācijas tehnoloģiju apvienotā sekcija", 2012. Referāts "Papildu lauku mehānisms TDA rīku būves platformā un tā lietojumi"**
- 72. LU konference, "Datorzinātņu un informācijas tehnoloģiju sekcija", 2014. Referāts "Dabiskās valodas saskarne OWL ontoloģiju redaktorā".
- 73. LU konference, "Datorzinātņu un informācijas tehnoloģiju sekcija", 2015. Referāts "Semantisko datubāzu platforma: iespējas un izaicinājumi"
- 74. LU konference, "Veselības organizācijas un ekonomika", medicīnas sekcija, 2016. Referāts "Dinamiskas analīzes rīki medicīnas datiem".
- 75. LU konference, Plenārsēde "Inovatīvas informācijas tehnoloģijas", 2017. Referāts "Vizuālie vaicājumi semantiskajās datubāzēs: problēmas un risinājumi".
- **77. LU konference, "Datorzinātņu un informācijas tehnoloģiju sekcija", 2019. Referāts "Vizuālo vaicājumu translācija".**

1 Esošās situācijas apskats

Šajā nodaļā tiek aprakstīti jēdzieni, metodes un rīki, kas raksturo esošo situāciju, kurā promocijas darbs sniedz savu ieguldījumu, tostarp minot jēdzienus, metodes un rīkus, kas izmantoti šajā darbā.

1.1 Semantisko tehnoloģiju pamata valodas

Šajā nodaļā minētas pamata valodas, ko izmanto semantiskajās tehnoloģijās, ar īsu skaidrojumu un atsauci uz valodas definīciju, jo tālākais darbs būtiski izmanto šīs valodas.

RDF un RDF Shēma

RDF (Resource Description Framework) ir semantisko datu standarts, resursu un to īpašību aprakstīšanai (Hayes and Patel-Schneider, 2014). Katrs resurss tiek identificēts ar unikālu identifikatoru – IRI. RDF glabā informāciju (izteikumus) par resursiem vienkāršu trijnieku veidā, kas sastāv no subjekta, predikāta un objekta (WEB, v). RDF shēma nodrošina datu modelēšanas vārdnīcu RDF datiem (Brickley and Guha, 2014). RDF shēma apraksta datu struktūru, izmantojot klases, īpašības un vērtības (Brickley and Guha, 2014).

Klases ir resursu grupas. Klasē ietilpstošie resursi tiek saukti par klases instancēm (Brickley and Guha, 2014). Objekta piederību klasei var aprakstīt, izmantojot RDF īpašību *rdf:type* (Brickley and Guha, 2014).

RDF īpašība ir attiecība starp subjekta un objekta resursiem (Brickley and Guha, 2014).

Shēmas līmenī īpašība *rdfs:subClassOf* tiek izmantota, lai norādītu, ka viena klase ir citas klases apakšklase (Brickley and Guha, 2014). Īpašību *rdfs:subPropertyOf* var izmantot, lai norādītu, ka viena īpašība ir citas apakšīpašība (Brickley and Guha, 2014). *rdfs:range* īpašība tiek izmantota, lai norādītu, ka īpašības vērtība ir vienas vai vairāku klašu instance (Brickley and Guha, 2014). *rdfs:domain* īpašība tiek izmantota, lai norādītu, ka jebkurš resurss, kuram ir dota īpašība, ir vienas vai vairāku klašu instance (Brickley and Guha, 2014).

OWL

OWL (*Web Ontology Language*) (McGuinness and Harmelen, 2004) ir semantiskā tīmekļa ontoloģiju valoda, kas paredzēta bagātīgu un sarežģītu zināšanu atspoguļošanai par lietām, lietu grupām un attiecībām starp lietām (Hitzler et al., 2012). Šobrīd aktuālā OWL valodas standarta versija ir OWL 2 (Motik et al., 2012).

OWL valoda ir veidota kā RDF un RDFS valodu paplašinājums (Motik et al., 2012). OWL valodā ontoloģijas aprakstā, līdzīgi kā RDFS, tiek izmantots atvērtās pasaules pieņēmums (Smith et al., 2004).

OWL valodā izšķir objektu un datu īpašības. Objektu īpašība apraksta attiecības starp diviem indivīdiem (Motik et al., 2012). Datu īpašība apraksta attiecības starp entītijas instanci un literāļa datu tipa vērtību (Motik et al., 2012). OWL ir arī īpašību aksiomas, lai definētu īpašību papildu raksturojumus, piemēram, *owl:inverseOf* vai *owl:TransitiveProperty* (Bechhofer et al., 2004). OWL piedāvā arī iespējas īpašību vērtībām noteikt papildus ierobežojumus. Ir divu veidu īpašību ierobežojumi: vērtības ierobežojums un kardinalitātes ierobežojums (Bechhofer et al., 2004). Vērtības ierobežojums (piemēram, *owl:allValuesFrom*, *owl:someValuesFrom*) kontrolē īpašības vērtību (*range*) (Bechhofer et al., 2004). Kardinalitātes ierobežojums (*owl:minCardinality*, *owl:maxCardinality*) nosaka īpašības vērtību skaitu (Bechhofer et al., 2004). Klases īpašība *owl:oneOf* nosaka, ka klase ir uzskaitījuma klase (McGuinness and Harmelen, 2004).

OWL 2 ontoloģiju valoda, līdzās bagātākam ontoloģiju uzdošanas līdzekļu klāstam (piemēram, kvalificētās kardinalitātes vai īpašību virknes) ietver arī plašas anotāciju uzdošanas iespējas (Motik et al., 2012), ko var izmantot, lai attēlotu ar ontoloģiju saistītu informāciju, kas nav izsakāma pamata ontoloģijas notācijā.

SPARQL

SPARQL (WEB, t) ir W3C standarta vaicājumu valoda pār RDF datiem. SPARQL vaicājumu pamata konstrukcija ir trijnieku šablonu kopa, ko sauc par pamata grafa šablonu (WEB, t). Trijnieku šabloni ir līdzīgi RDF trijniekiem, kuros kādi no subjekta, predikāta vai objekta var būt mainīgie. Trijnieku šabloni ir ietverti SPARQL WHERE klauzulā (WEB, t). FILTER šablons tiek izmantots, lai sašaurinātu rezultātus, pamatojoties uz nosacījumiem (WEB, t). Lai šablonus WHERE klauzulā iestatītu kā neobligātus, tiek izmantota klauzula OPTIONAL (WEB, t). NOT EXISTS un MINUS klauzulas tiek izmantotas, lai norādītu šablonus, kuri ir izslēdzami no rezultātu kopas (WEB, t).

SELECT klauzula tiek izmantota, lai norādītu, kuri vaicājuma mainīgie jāiekļauj vaicājuma rezultātā (WEB, t). Lai iegūtu rezultātus apkopotā veidā, tiek izmantotas tādas agregācijas funkcijas kā SUM, MIN, MAX, AVG un GROUP_CONCAT (WEB, t). SPARQL piedāvā arī iespēju vaicājumiem veidot apakšvaicājumus.

SPARQL var izmantot, lai izteiktu vaicājumus pār dažādos datu avotos esošiem datiem neatkarīgi no tā, vai dati tiek saglabāti sākotnēji RDF veidā, vai arī tiek attēloti RDF formātā caur starpprogrammatūru (WEB, t).

1.2 Esošo rīku un pieeju pārskats

Šajā nodaļā minētie rīki un pieejas, kas izmanto un ir balstīti uz semantiskajām tehnoloģijām, un ir domāti ontoloģiju vizualizācijai un vizuālo vaicājumu uzdošanai pār semantiskajiem datiem. Darbā piedāvātie risinājumi bagātina pieejamo iespēju klāstu gan ontoloģiju vizualizācijas un vizuālas apstrādes, gan arī vizuālo vaicājumu veidošanas jomā.

1.2.1 Ontoloģiju vizualizācija

OWL ontoloģiju attēlošana saprotamā formā ir būtiska gan ontoloģijas izstrādātājiem, gan to lietotājiem. Ir izstrādātas vairākas pieejas un rīki ontoloģijas grafiskai attēlošanai tā, lai ontoloģiju saistītas konstrukcijas prezentācijā ir savienotās savā starpā. Rakstā (Dudas et al., 2018) ir veikts plašs pētījums par ontoloģiju attēlošanas metodēm un rīkiem.

Šajā sadaļā ir dots īss pārskats par populārākajām grafiskajām ontoloģiju attēlošanas pieejām un rīkiem.

OWLViz

OWLViz (WEB, p) ir plaši izmantotā ontoloģiju redaktora Protégé (WEB, r) spraudnis, kas grafiski attēlo OWL ontoloģijā esošo klašu hierarhiju. OWLViz ļauj aplūkot un inkrementāli caurstaigāt OWL ontoloģiju klašu hierarhijas, ļaujot salīdzināt tieši ieviesto un secināto klašu hierarhijas (WEB, p). Abi skati var tikt saglabāti vairākos grafiskos formātos. Diagrammā klases tiek attēlotas kā virsotnes, savukārt īpašības tiek parādītas kā virsotnes savienojošas šķautnes (WEB, p). Attēlo tikai klases (kā virsotnes) un *is-a* attiecības (šķautnes) starp virsotnēm. Var izvēlēties attēlot klases apakšklases, virsklases vai visās ontoloģijas klasēs (WEB, p).

No OWL konstrukciju kopuma OWLViz atbalsta tikai klases un apakš klases attiecības, citas OWL konstrukcijas nav pieejamas. Rīks arī neatbalsta ontoloģijas grafisko rediģēšanu.

VOWL

VOWL (*Visual Notation for OWL Ontologies*) (Lohmann et al., 2016) ir grafiska, uz lietotāju orientēta valoda ontoloģiju attēlošanai. Tā nodrošina OWL valodas elementu grafisko attēlošanu, apvienotu ar uz spēku vērstu grafu izkārtojumu, kas vizualizē ontoloģiju (Lohmann et al., 2016).

Ontoloģijas klases tiek attēlotas kā apļa veida virsotnes, kas ir savienotas ar līnijām, kas savukārt apraksta īpašības attiecības starp domēna un vērtību klasi; īpašības nosaukums tiek attēlots taisnstūrī, kas novietots uz līnijas (Lohmann et al., 2016).

Ir divas VOWL realizācijas: ProtégéVOWL (WEB, s) ir realizēts kā Java spraudnis ontoloģijas redaktora Protégé Desktop versijai, un WebVOWL (WEB, ab) ir atsevišķa lietojumprogramma. ProtégéVOWL koncentrējas uz ontoloģijas shēmas vizualizāciju, attēlojot klases, īpašības un datu tipus, bet tajā netiek ņemti vērā indivīdi un datu vērtības (Lohmann et al., 2014). WebVOWL informācija par indivīdiem un datu vērtībām tiek parādīta vai nu pašā vizualizācijā (klases virsotnes izmērs atspoguļo instanču skaitu), vai arī citā lietotāja saskarnes daļā (Lohmann et al., 2016). WebVOWL atbalsta lielākās daļas OWL valodas konstrukciju vizualizācijas iespējas. WebVOWL nodrošina vairākus filtrus, kas palīdz samazināt VOWL diagrammas lielumu, tādus kā rādīt/slēpt datu tipa īpašības, objektu īpašības, apakšklases, nepārklājošas klases (Lohmann et al., 2016).

WebVOWL Editor (Wiens et al., 2018), kas pašlaik ir daļa no WebVOWL, nodrošina grafiskas ontoloģijas veidošanas un rediģēšanas iespējas.

Ontoloģiju vizualizācijā VOWL saimes rīki neizmanto UML veida notāciju, līdz ar to nevar iegūt UML formai atbilstošu kompaktu reprezentāciju. Nav arī pieejama iespēja vizuāli attēlot indivīdus un definēt konkrētiem lietojumiem specifiskas ontoloģiju vizualizācijas konstrukcijas.

TopBraid Composer

TopBraid Composer (WEB, x) ir biznesa klases modelēšanas vide semantiskā tīmekļa ontoloģiju un semantisko lietojumprogrammu veidošanai, kas veidota uz Eclipse ietvara bāzes (WEB, x).

TopBraid Composer ir divi atsevišķi režīmi ontoloģiju vizuālai prezentēšanai: RDF grafu veidā un kā zināšanu grafu shēmu diagrammas, kurās tiek izmantota UML tipu notācija, attēlojot klases kā virsotnes un īpašības kā orientētas šķautnes ar predikāta vārdu (WEB, x).

TopBraid Composer vide piedāvā ontoloģijas rediģēšanas iespējas, izmantojot formu saskarni, grafiskās ontoloģijas rediģēšanas iespējas nav. TopBraid Composer atbalsta lielu ontoloģiju ielādi, bet tajā nav iespējas attēlot ontoloģiju pilnībā, tajā var attēlot tikai izvēlētas ontoloģijas daļas (WEB, x). RDF grafu veida prezentācija atbalsta pilnu OWL konstrukciju vizualizāciju, tomēr šeit netiek nodrošināta kompakta ontoloģiju vizuālā attēla veidošana. Attēlojot ontoloģiju UML klašu diagrammu formā, var tikt attēlotas tikai vienkāršākās OWL konstrukcijas (klases, īpašības, to sasaiste, datu tipi). TopBraid Composer rīkā arī nav paredzēti līdzekļi lietojumam specifisku ontoloģiju attēlošanas notāciju veidošanai.

OntoDia

OntoDia (Mouromtsev et al., 2015) ir atvērtā pirmkoda tīmekļa rīks RDF datu kopu un OWL ontoloģiju vizualizēšanai un vizualizēto grafu koplietošanai (Mouromtsev et al., 2015). OntoDia vizualizācija ir balstīta uz RDF trijnieku struktūru. Tiek atbalstītas ontoloģiju filtrēšanas iespējas, var izvēlēties kāda veida šķautnes rādīt/slēpt un attēlot virsotnes saistītos objektus ar noteiktiem šķautņu tipiem (Mouromtsev et al., 2015).

OntoDia atbalsta lielu ontoloģiju ielādi, bet nav iespējas attēlot ontoloģiju pilnībā, var attēlot tikai izvelētas ontoloģijas daļas. Kardinalitātes un īpašību raksturojumi netiek atbalstīti. OntoDia neizmanto UML klašu diagrammu veida notāciju, līdz ar to nevar iegūt UML formai atbilstošu kompaktu reprezentāciju. Rīks neatbalsta ontoloģijas grafisko rediģēšanu, kā arī tajā nav paredzēti līdzekļi lietojumam specifisku ontoloģiju attēlošanas notāciju veidošanai.

OntoGraf

OntoGraf (WEB, k) ir Protégé rīka spraudnis, kas sniedz atbalstu interaktīvai navigācijai OWL ontoloģiju attiecībās. OntoGraf parāda klases kā pakārtotas virsotnes, kas apzīmētas ar to nosaukumiem. Hierarhiskās attiecības, īpašības un ierobežojumi tiek parādīti kā saites. OntoGraf atbalsta dažādas attiecības: apakšklases, indivīdu, objektu īpašību domain/range attiecības un ekvivalences attiecības (WEB, l). Attiecības un virsotņu tipus var filtrēt (WEB, k), var izvēlēties kāda veida šķautnes rādīt/slēpt, paslēpt virsotnes, kas nav saistītas ar pārējo grafu, attēlot tikai virsotnes tiešos kaimiņus (visus, vai ar noteiktiem saites veidiem). Ontoloģijas struktūras automātiskai organizēšanai tiek atbalstīti dažādi izkārtojumi (WEB, k).

Kardinalitātes, īpašību raksturojumi un anotāciju aksiomas netiek atbalstītas. OntoGraf neizmanto UML klašu diagrammu veida notāciju, līdz ar to nevar iegūt UML formai atbilstošu kompaktu reprezentāciju. Rīks neatbalsta ontoloģijas grafisko rediģēšanu, kā arī nav pieejamas informācijas par lietojumam specifisku ontoloģiju attēlošanas notāciju veidošanas iespējām tajā.

OntoSphere3D

OntoSpere3D (Bosca et al., 2005) ir ontoloģiju vizualizācijas rīks, kas izmanto trīsdimensiju telpu, kur informāciju bagātina vizuālas norādes (piemēram, vizualizēto entītiju krāsa vai izmērs) (Bosca et al., 2005). Jēdzieni tiek parādīti kā sfēras, instances tiek attēlotas kā kubi, literāļi tiek renderēti kā cilindri, un attiecības starp entītijām attēlo ar bultiņu līnijām (Bosca et al., 2005). Rīks ir ieviests kā spraudnis diviem dažādiem ietvariem: Protégé un Eclipse (WEB, m). Rīkā ir pieejami vairāki skati uz ontoloģiju: jēdziena tiešas attiecības, jēdziena apakškoks, pilna informācija par vienu jēdzienu un instanču skats (WEB, m).

Kardinalitātes un anotāciju aksiomas netiek atbalstītas. OntoSpere3D neizmanto UML tipu notāciju, līdz ar to nevar iegūt UML formai atbilstošu kompaktu reprezentāciju. Rīks neatbalsta ontoloģijas rediģēšanu, kā arī nav pieejamas informācijas par lietojumam specifisku ontoloģiju attēlošanas notāciju veidošanas iespējām tajā.

OWLGrEd

OWLGrEd (Barzdins et al., 2010a) ir UML tipa grafiskais redaktors OWL 2, kas attēlo ontoloģijas, izmantojot paplašinātu UML klašu diagrammu (Scott, 2009) notāciju un piedāvā ontoloģijas rediģēšanas iespējas (Barzdins et al., 2010a).

OWL klases vizuālajā notācijā tipiski tiek attēlotās kā UML klases, datu īpašības kā klases atribūti, objektu īpašības kā asociāciju lomu vārdi, indivīdi kā objekti, asociāciju domēna klases kardinalitāšu ierobežojumi kā UML kardinalitātes (Barzdins et al., 2010a). OWLGrEd redaktors nodrošina iespēju specificēt klašu izteiksmes kompaktā tekstuālā formā (Barzdins et al., 2010b). OWLGrEd atbalsta pilnu OWL konstrukciju vizualizāciju, kā arī OWL ontoloģiju vizuālas rediģēšanas iespējas. Tomēr arī OWLGrEd rīks pirms promocijas darba autores ieguldījuma neatbalstīja iespēju veidot lietojumiem specifiskas ontoloģiju attēlošanas iespējas.

OWLGrEd rīks kā tehnoloģija, uz kuru balstās šajā promocijas darbā veiktā izstrāde, ir plašāk apskatīts nodaļā 1.4.

Secinājumi

Tabulā 1.1 apkopoti aprakstīto rīku OWL pamata konstrukciju attēlošanas veidi un rīku raksturojošas īpašības.

Ontoloģiju attēlošanai grafiskā veidā ir izstrādāti dažādi rīki un metodes. Aprakstītās pieejas ļauj vizualizēt un pārskatīt ontoloģijas. Lielāka daļa no apskatītajiem rīkiem nenodrošina vai nodrošina tikai minimālas ontoloģijas grafiskas veidošanas un rediģēšanas iespējas, kas ir svarīgs aspekts ontoloģiju veidotājiem. Īpašību attēlošana šķautnes veidā, kas tiek atbalstīta vairumā apskatīto rīku, vidēju un lielu ontoloģiju gadījumā apgrūtina ontoloģijas uztveri un darbu ar to. Daļā rīku (piemēram, OntoGraf (WEB, k), OntoDia (Mouromtsev et al., 2015), WebVOWL (Lohmann et al., 2016)) lietotājam ir iespēja norādīt, kādas ontoloģijas konstrukcijas iekļaut un kādas neiekļaut vizualizācijā. Tomēr izvēle starp dažādiem vienas konstrukcijas attēlošanas veidiem, kas var būt svarīga, lai noteiktās situācijās ontoloģijas attēlotu vēl kompaktāk, parasti ir neesoša (OWLGrEd redaktorā šāda iespēja tiek piedāvāta grafiskās ontoloģiju rediģēšanas ietvaros, sk. nodaļu 1.4).

Apskatītie rīki nenodrošina notācijas paplašināšanu ar konkrētiem lietojumam specifiskiem papildinājumiem, kas ietver vizuālo izskatu, semantikas definēšanu un tās saglabāšanu. Rakstā (Dudas et al., 2018) šīs iespējas trūkums ontoloģiju vizualizācijas un rediģēšanas rīkos ir minēts kā viens no izaicinājumiem dotajā jomā. Šajā darbā tiek piedāvāts iespējams risinājums šai problēmai OWLGrEd ontoloģiju vizualizācijas un vizuālas rediģēšanas rīka kontekstā.

1.1. tabula. Aprakstīto rīku OWL pamata konstrukcijas un iespējas

	Klases	Īpašības	Indivīdi	Apakš klases attiecības	Kardinalitātes	Īpašību vērtības ierobežojumi	Īpašību raksturojumi	Anotācijas	Kompakts attēlojums	Vizuāla rediģēšana
OWLViz	+	–	–	+	–	–	–	–	–	–
VOWL	+	+	–	+	+	+	+	+	–	+
TopBraid Composer (UML)	+	+	+	+	+	–	–	–	+	–
TopBraid Composer (RDF grafs)	+	+ (1)	+	+	+	+	+	+	–	–
OntoDia	+	+	+	+	–	+	–	+	–	–
OntoGraf	+	+	+	+	–	+	–	–	–	–
OntoSphere3D	+	+	+	+	–	+	+	–	–	–
OWLGrEd	+	+	+	+	+	+	+	+	+	+
(1) Īpašības tiek attēlotas kā virsotnes, pārējos rīkos īpašības tiek attēlotas kā šķautnes										

1.2.2 Vizuālo vaicājumu rīki

SPARQL ir W3C standarta vaicājumu valoda pār RDF datiem. Kaut arī semantiskās RDF/SPARQL tehnoloģijas piedāvā augstāka līmeņa skatu uz datiem nekā klasiskās relāciju datu bāzes ar SQL vaicājumu valodu, SPARQL vaicājumu formāla tekstuāla notācija joprojām sarežģī tās izmantošanu IT profesionāļiem un nozaru ekspertiem (Soylu et al., 2016). Eksistē vairākas grafiskas pieejas, lai atvieglotu SPARQL vaicājumu veidošanu, tādas kā vaicājumu attēlošanā izmantojot UML stila notāciju vai RDF virsotņu un šķautņu vizuālos grafus.

Šajā sadaļā ir dots īss pārskats par pieejām un rīkiem vizuālo vaicājumu uzdošanu pār semantiskiem datiem.

OptiqueVQS

OptiqueVQS (Soylu et al., 2018) ir vizuālo vaicājumu formulēšanas rīks, kas ļauj vaicāt pār ontoloģiju, izmantojot UML stila notāciju. OptiqueVQS sastāv no vizuālās saskarnes un navigācijas grafa, kas izvilkts no ontoloģijām (Kharlamov et al., 2017). Saskarnes komponentes tiek aizpildītas saskaņā ar informāciju no navigācijas grafa.

OptiqueVQS ir paredzēts galalietotājiem, kuriem nav vai ir ļoti ierobežotas tehniskās prasmes un zināšanas (Soylu et al., 2016). OptiqueVQS var konfigurēt, lai pieslēgtos SPARQL piekļuves punktam, vai var izmantot OBDA ietvaru, kas pārveido relāciju datus RDF formātā (Soylu et al., 2018).

OptiqueVQS nav atbalsta tādām SPARQL konstrukcijām kā neobligāta vaicājuma daļa, negācija, apakš vaicājums un sarežģītāku izteiksmju pieraksts. Agregēšana iespējama tikai no rezultātu tabulas, nav atbalsta agregēšanai no grafiskas saskarnes.

ViziQuer (agrīnās versijas)

ViziQuer (Barzdins et al., 2009; Zviedris, 2014) ir rīks SPARQL vaicājumu ģenerēšanai, kas vaicājumu attēlošanai izmanto UML stila diagrammas. Vaicājuma valodā ir divas galvenās komponentes – klase (jēdziens) un saite (relācija, īpašība), kas apzīmē asociāciju starp divām klasēm.

Vaicājuma veidošanas rīka galvenā ideja ir tāda, ka lietotājs vispirms izvēlas tos jēdzienus (klases) no ontoloģijas, kas ir iesaistīti viņa vaicājumā, un pēc tam specificē, kā šie jēdzieni ir jāsavieno (Zviedris, 2014). Lietotājs var sašaurināt vaicājumu, dažām atlasītajām klasēm pievienojot atribūtu nosacījumus, un var izvēlēties, kuras klases atribūtus viņš vēlas redzēt rezultātu tabulā (Zviedris, 2014).

ViziQuer atbalsta tādās SPARQL konstrukcijas kā neobligāta vaicājuma daļa un negācija (caur "not exists" vai "!bound" klauzulām). Rīkā nav atbalsta tādām konstrukcijām kā apakš vaicājums un sarežģītāku izteiksmju pieraksts. Atbalsts agregāciju veidošanai arī ir ierobežots (agregācijas argumentam ir jābūt mainīgajam, tas nevar būt izteiksme).

QueryVOWL

QueryVOWL (Haag et al., 2015) ir vizuāla valoda, ko var izmantot, lai veidotu vaicājumus pār saistītiem datiem. Vizuāli tā ir balstīta uz VOWL, semantiski tā ir definēta, saistot to ar SPARQL fragmentiem (Haag et al., 2015). QueryVOWL notācijas elementi ir semantiski definēti, izmantojot SPARQL vaicājuma fragmentus vai instrukcijas, kā ģenerēt šādus fragmentus, pamatojoties uz QueryVOWL elementu grafu kontekstu. Vaicājuma virsotnes tiek attēlotas kā apļi, kas ir savienoti ar līnijām, kas reprezentē datu un objektu īpašības.

QueryVOWL vaicājums var tikt izpildīts pār RDF grafiem. Vaicājuma rezultātu kopa ir visu RDF grafa apakšgrafu kopa, kas atbilst šablonam, kuru izsaka QueryVOWL vaicājums (Haag et al., 2015).

QueryVOWL nav atbalsta tādām SPARQL konstrukcijām kā neobligāta vaicājuma daļa, negācija, apakš vaicājums, agregācija un sarežģītāku izteiksmju pieraksts. QueryVOWL rīks vizualizācijai neizmanto UML veida notāciju, līdz ar to nevar iegūt vaicājumu kompaktā veidā.

RDF Explorer

RDF Explorer (Vargas et al., 2019) sistēmas mērķis ir dot iespēju nespeciālistiem veikt vaicājumus un izpētīt RDF grafus. Vizuālā vaicājuma valoda balstās uz vizuālā vaicājuma grafa jēdzienu, kas tiek tulkots par SPARQL (Vargas et al., 2019). Vizuālā vaicājuma grafs sastāv no četriem algebriskiem operatoriem, kuri atbilst lietotāju darbībām: pievienot mainīgā virsotni, pievienot konstantu virsotni, pievienot šķautni starp divām esošām virsotnēm ar mainīgā iezīmi un pievienot šķautni starp divām esošām virsotnēm ar IRI iezīmi (Vargas et al., 2019).

RDF Explorer saskarne sastāv no sešām galvenajām komponentēm, kas tiek parādītas trīs kolonnās: meklēšanas panelis, vizuālo vaicājumu redaktors, virsotnes detalizēts skats, virsotņu redaktors (ļauj pievienot ierobežojumus izceltajai virsotnei), SPARQL vaicājumu redaktors (parāda pašreizējo vaicājumu) un palīdzības panelis (Vargas et al., 2019). Vizuāla vaicājuma skatījumā virsotnes var tikt ievilktas vaicājuma vizuālajā panelī, kur tās var tikt savienotas ar līnijām, kas reprezentē īpašības (Vargas et al., 2019). Lai izvairītos no pārblīvēšanas, datu tipa īpašības var attēlot iekšā virsotnē (Vargas et al., 2019).

RDF Explorer nav atbalsta tādām SPARQL konstrukcijām kā neobligāta vaicājuma daļa, negācija, apakš vaicājums, agregācija un sarežģītāku izteiksmju pieraksts. RDF Explorer rīks vizualizācijai neizmanto UML veida notāciju, līdz ar to nevar iegūt vaicājumu kompaktā veidā.

LinDA Query Designer

LinDA Query Designer var izmantot, lai izveidotu saistīto datu vaicājumus vilkt-un-nomest (drag-and-drop) veidā (WEB, g). Shēma tiek automātiski nolasīta no pieslēgta piekļuves punkta. Vienā vaicājumā var būt elementi no vairākiem piekļuves punktiem, lai atļautu to sasaisti (WEB, g). Ir meklēšanas funkcionalitāte klasēs un attiecībās (WEB, g). Dažādu klašu instances ar izvēlētām īpašībām var filtrēt, grupēt vai kārtot pēc šīm īpašībām (WEB, g). LinDA Query Designer vizualizācijai izmanto UML veida notāciju, attēlojot īpašības kā klases atribūtus.

LinDA Query Designer atbalsta SPARQL neobligāto klauzulu bet tikai uz trijnieka šablona, neatbalsta neobligāto klauzulu uz Group Graph Pattern. Rīkā nav atbalsta tādām konstrukcijām kā negācija, apakš vaicājums un sarežģītāku izteiksmju pieraksts. Agregēšana iespējama tikai no rezultātu tabulas, nav atbalsta agregēšanai no grafiskās saskarnes.

GRUFF

GRUFF ir grafiska vizualizācijas programmatūra datu saistību izpētei (Aasman, 2017). GRUFF sniedz iespēju vizuāli veidot vaicājumus un izpētīt, kā tie attīstījās laika gaitā (Aasman, 2017). Tā

darbojas ar AllegroGraph (WEB, b) un ar SPARQL piekļuves punktiem (Aasman, 2017). Informāciju var pārlūkot kā virsotņu un šķautņu vizuālos grafus, kas tiek izkārtoti automātiski, kā arī kā virsotņu īpašību tabulas (Aasman, 2017). Vaicājumus var rakstīt tekstuāli kā SPARQL, Prolog kodu vai noformēt vizuāli kā grafus (Aasman, 2017).

GRUFF atbalsta tādās SPARQL konstrukcijas kā neobligāta vaicājuma daļa un negācija (caur "not exists" vai "minus" klauzulām). Rīkā nav atbalsta tādām konstrukcijām kā apakš vaicājums, agregācija un sarežģītāku izteiksmju pieraksts. GRUFF vizualizācijai neizmanto UML veida notāciju, līdz ar to nevar iegūt vaicājumu kompaktā veidā.

SPARQLGraph

SPARQLGraph ir tīmekļa platforma, kas lietotājiem ļauj grafiskā veidā izveidot semantisko tīmekļa datu bāžu vaicājumus (Schweiger et al., 2014). Platformas galvenā saskarne sastāv no lielas rasēšanas tāfeles, ko izmanto jauno vaicājumu grafu zīmēšanai (Schweiger et al., 2014). Lietotāji vaicājumam var pievienot jaunus elementus un to atribūtus, vienkārši velkot un nometot tos uz tāfeles (Schweiger et al., 2014). SPARQLGraph ir spējīga izveidot apvienotus vaicājumus, kur vienlaikus tiek meklēts dažādās datu bāzēs, bez datu pārveidošanas vai manuālas rezultātu filtrēšanas (Schweiger et al., 2014).

SPARQLGraph nav atbalsta tādām SPARQL konstrukcijām kā neobligāta vaicājuma daļa, negācija, apakš vaicājums, agregācija un sarežģītāku izteiksmju pieraksts. SPARQLGraph vizualizācijai neizmanto UML veida notāciju, līdz ar to nevar iegūt vaicājumu kompaktā veidā.

Secinājumi

Vizuālo vaicājumu par semantiskiem datiem veidošanai ir izstrādāti dažādi rīki un metodes. Tabulā 1.2 ir apkopotas apskatīto rīku atbalstītas funkcionalitātes. Šī darba ietvaros izstrādātais rīks ViziQuer/web tika pievienots tabulai salīdzināšanas nolūkā.

1.2. tabula. Vizuālo vaicājumu rīku atbalstītas funkcionalitātes

	Neobligāta izteiksme	**Negācija**	**Agregācija**	**Apakš vaicājumi**	**Tekstuālas izteiksmes**	**Kompakts attēlojums**
OptiqueVQS	–	–	+ / – (1)	–	–	+
QueryVOWL	–	–	–	–	–	–
RDF Explorer	–	–	–	–	–	–
LinDA Query Designer	+ / – (3)	–	+ / – (1)	–	–	+
GRUFF	+	+	–	–	–	–
SPARQLGraph	–	–	–	–	–	–
ViziQuer (2014)	+	+	+ (2)	–	–	+
ViziQuer/web	+	+	+	+	+	+

(1) Agregēšana iespējama tikai no rezultātu tabulas, nav atbalsta agregēšanai no grafikas saskarnes
(2) Agregēšana iespējama tikai pār mainīgā vārdu, nav atbalsta agregēšanai pār izteiksmēm
(3) Neobligāta klauzula iespējama tikai attiecībā uz trijnieka šablona, neatbalsta neobligāto klauzulu attiecībā uz Group Graph Pattern

Aprakstītās pieejas ļauj strādāt tikai ar vienkāršiem vaicājumiem, kur agregācijas izteiksmju izveides iespējas ir ļoti ierobežotas (OptiqueVQS un LinDA nodrošina tikai ārējā līmeņa agregācijas iespēju), trūkst apakšvaicājumu konstrukcijas un iespējas rakstīt sarežģītākas izteiksmes. Daļa no rīkiem nepiedāvā neobligātas klauzulas un negācijas iespējas (tādi kā OptiqueVQS, QueryVOWL, RDF Explorer, SPARQLGraph). Kompakts vaicājuma pieraksta veids, ko nodrošina UML veida notācija, ir atbalstīts OptiqueVQS, LinDA Query Designer un ViziQuer rīkos. Lai galalietotāji varētu pilnvērtīgi strādāt ar vizuālo vaicājumu rīkiem, tiem, līdzi ar plaši lietotām konstrukcijām, kā OPTIONAL un NOT

EXISTS klauzulas, ir jānodrošina plašāks iespēju klāsts, tādu kā sarežģītāko izteiksmju un agregāciju atbalsts. Šādas iespējas tiks piedāvātas darbā izstrādātajā ViziQuer/web rīkā.

1.3 Rīku būves platformas

Promocijas darbs, tostarp tajā attīstītie rīki, balstās uz LU MII izstrādātām vizuālo rīku būves platformām GRTP/TDA un ajoo. Abas platformas uztur konfigurācijas informāciju vienā konceptuālā līmenī ar izpildes laika instanču informāciju. Tālāk dots īss šo platformu apraksts, kas skaidro kontekstu darbā veidotajiem risinājumiem.

1.3.1 GrTP/TDA

GrTP/TDA (*Graphical Tool Building Platform*) (*Transformation-Driven Architecture*) (Barzdins et al., 2007) ir uz meta modeļiem un modeļu transformācijām balstīta rīku modelēšanas platforma, kas atbalsta vairākus informācijas prezentēšanas veidus, izmantojot prezentācijas dziņus ar to datu struktūrām, ko apraksta prezentācijas meta modeļi (Barzdins et al., 2007). Svarīgs prezentācijas dzinis ir dzinis grafiskajām diagrammām (ar atbilstošo grafiskās diagrammas meta modeli) (Barzdins et al., 2010c). Lietotāju dialogu logiem arī ir dzinis un meta modelis (Kozlovics, 2010). GrTP/TDA transformācijas tiek izmantotas, lai saistītu dažādas prezentācijas, kā arī citus datu domēnus, ja tādi ir izveidoti (Barzdins et al., 2007). Katrs prezentācijas meta modelis var tikt apstrādāts neatkarīgi, izmantojot tam atbilstošu dzini.

OWLGrEd redaktors ir izstrādāts uz GrTP/TDA platformas bāzes.

Konfigurators (Sprogis, 2010) GrTP/TDA platformā ir rīks, kas paredzēts, lai ātri un vienkārši veidotu un uzturētu vizuālās rīku definīcijas, kas ir kodētas kā rīku definēšanas meta modeļa instances (kas sastāv no diagrammu un to elementu "tipiem", kā arī to vizuālā stila definēšanas iespējām) (Sprogis, 2010). Ar konfiguratora palīdzību jaunie tipi tiek veidoti grafiski, un to atribūti tiek pievienoti ar dialogu logu palīdzību (Sprogis, 2010).

Ar konfiguratora palīdzību var veidot ne tikai jaunus tipus un to atribūtus, bet arī noteikt tipa uzvedību, definēt dialoga loga parametrus, pievienot tipam un atribūtiem specifiskus paplašinājuma punktus, kā arī noteikt dotā tipa elementa un atribūtu attēlošanas stilus (Sprogis, 2010).

1.3.2 ajoo

ajoo (Sprogis, 2016) ir DSML (*Domain-Specific Modeling Languages*) ietvars vizuālu modelēšanas rīku veidošanai tīmekļa vidē. Ietvars sastāv no DSML rīku veidošanas platformas un DSML konfigurācijas rīka. Platforma ļauj veidot dažādus DSML rīkus, atkārtoti izmantojot to komponentes, neprogrammējot tos katru reizi no jauna (Sprogis, 2016). Vizuālā rīka definīcija ajoo platformā ir JSON fails, kas apraksta konkrētā rīka struktūru.

ajoo platformā ir pieejams **konfigurators**, kas ļauj specificēt DSML rīkus, izmantojot grafisko lietotāja saskarni (Sprogis, 2016). Tas vienkāršo un paātrina DSML ieviešanas procesu, tādējādi ietvars ļauj implementēt plašu DSML rīku apgabalu bez programmēšanas (Sprogis, 2016).

Konfiguratora saskarne ir daļēji grafiska, tas ir, tā ļauj izveidot kastīšu un līniju specifikācijas kā grafiskos elementus diagrammā, un mērķa redaktorā attēlot to noklusētos stilus pēc to izskata (Sprogis, 2016). Elementa īpašību vērtības tiek ievadītas, izmantojot dialogu logus.

Platformas ajoo ietvaros promocijas darbā ir izstrādāta ViziQuer rīka tīmekļa versija.

1.4 OWLGrEd

OWLGrEd (Barzdins et al., 2010a) ir UML tipa grafiskais redaktors OWL 2, kas attēlo ontoloģijas, izmantojot paplašinātu UML klašu diagrammas (Scott, 2009) notāciju un piedāvā ontoloģijas rediģēšanas iespējas.

OWLGrEd redaktors (WEB, n) ir būvēts uz GrTP/TDA platformas bāzes.

OWL klases vizuālajā notācijā tipiski tiek attēlotās kā UML klases, datu īpašības kā klases atribūti, objektu īpašības kā asociāciju lomu vārdi, indivīdi kā objekti, asociāciju domēna klases kardinalitāšu ierobežojumi kā UML kardinalitātes. Lai atbalstītu vizuālo OWL ontoloģijas notāciju UML klašu diagrammas OWLGrEd redaktorā ir bagātinātas ar jauniem paplašinājumiem un notācijām (Barzdins et al., 2010a; Cerans et al., 2013):

- Jaunie lauki klasēm atbilstošajās virsotnēs – ekvivalentas klases (*equivalent class*), virsklases (*super class*), nepārklājošas klases (*disjoint class*); katrs no šiem laukiem izmanto izteiksmes OWL Mančestras sintaksē (Horridge and Patel-Schneider, 2012), kas padara notāciju kompaktāku un saprotamāku;
- Asociācijām un atribūtiem lauki ekvivalentam, virs-īpašībām un nepārklājošam īpašībām, kā arī lauki detalizētai īpašību aprakstīšanai (*functional*, *transitive*, u,c);
- Iespēja veidot anonīmas klases, kas nesatur klases nosaukumu, bet satur ekvivalento klašu izteiksmi;
- Savienotāji (connectors) *equivalent* un *disjoint* aksiomu vizualizācijai.;
- Saites, kas atspoguļo objektu īpašību ierobežojumus, tādus kā *some*, *only*, *exactly*.

Attēlā 1.1 ir parādīts OWLGrEd redaktorā veidotas diagrammas piemērs.

Tā vietā, lai lietotu atsevišķu grafisko elementu katram loģiskam vienumam klases izteiksmē, OWLGrEd redaktors nodrošina iespēju specificēt klašu izteiksmes kompaktā tekstuālā formā (Barzdins et al., 2010b). Ja izteiksme tiek lietota vairākās vietās, to var attēlot kā anonīmu klasi. Anonīmas klases tiek izmantotas arī kā pamats īpašību *domain* un *range* aksiomu specificēšanai, ja tās nav vārdā nosauktās klases (Barzdins et al., 2010b).

OWLGrEd redaktors atbalsta vairākus alternatīvus veidus, kā attēlot vienas un tās pašas konstrukcijas īpašībām (*properties*), ierobežojumiem (*restrictions*), ekvivalentām klasēm (*equivalent class*), apakšklasēm (*subclass*), nepārklājošām klasēm (*disjoint class*) (Barzdins et al., 2010b). Līnija, kas grafiski savieno divas klases, parāda to attiecības. Tekstuāla forma var būt piemērotāka, lai samazinātu grafa pārslodzi ar līnijām vai atļautu lielu grafu sadalīšanu atsevišķos fragmentos.

Tā kā OWLGrEd rīks piedāvā vienu un to pašu OWL aksiomu attēlošanu dažādos veidos, tās paver iespējas ieviest mehānismu, kas ļautu lietotājiem izvēlēties piemērotāko veidu izvēlēto OWL aksiomu attēlošanai gan ontoloģijas vizualizācijas laikā, gan turpmākajā darbā.

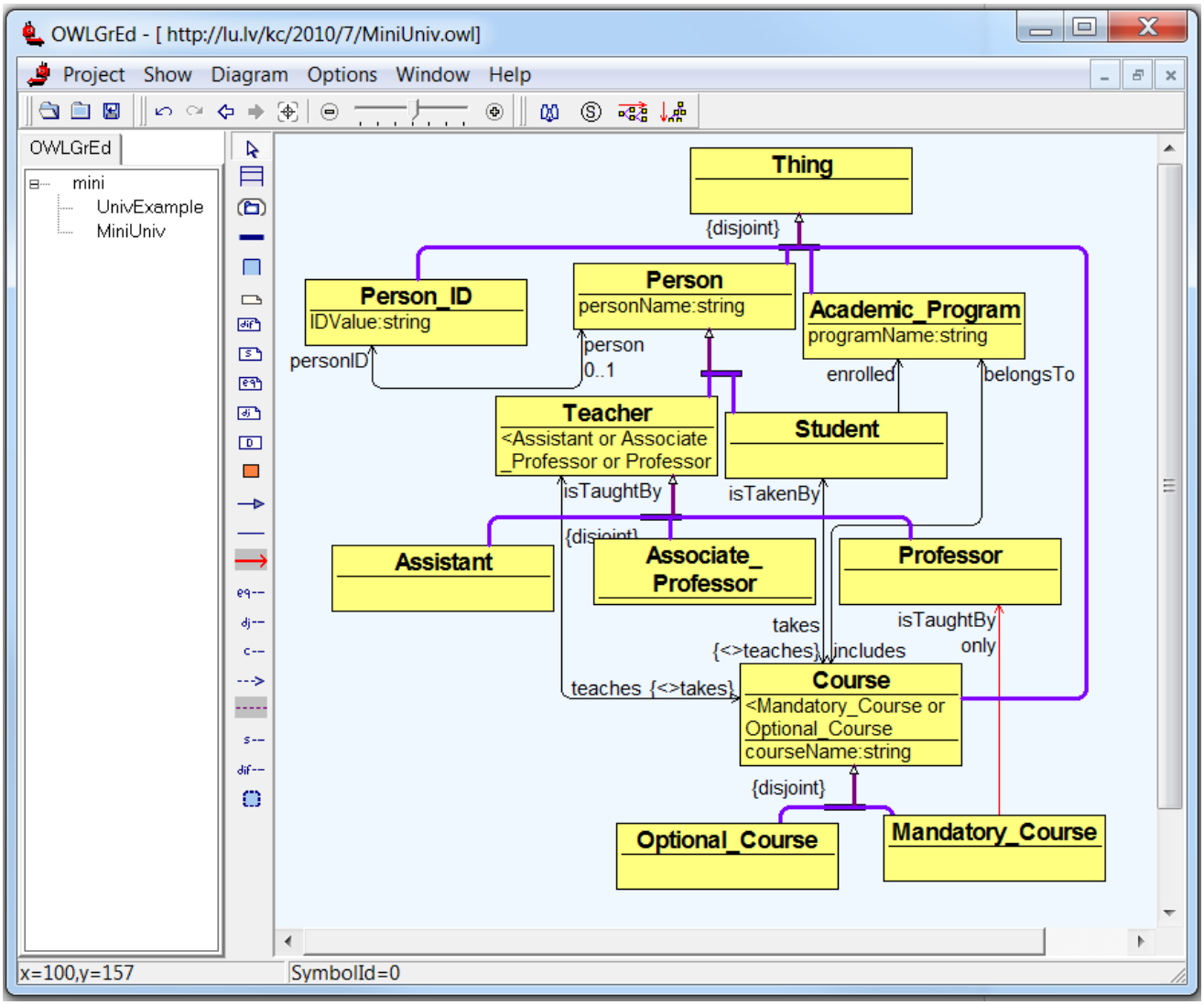
Tiek atbalstītas arī tādas funkcionalitātes kā vairāku diagrammu izkārtojuma veidi un meklēšanas mehānisms.

OWLGrEd redaktors izmanto divus principus (Cerans et al., 2013):

- Ontoloģijas autorēšana (ontoloģija tiek veidota un rediģēta OWLGrEd redaktorā un saglabāta vienā no standarta OWL formātiem);
- Ontoloģijas vizualizācija, kas ietver automātisko ontoloģijas veidošanu un izkārtošanu, kam seko manuāla precizēšana.

OWLGrEd redaktoru var ērti izmantot UML stila ontoloģiju vizualizēšanai un ontoloģiju veidošanai vizuālajā vidē. Tomēr tas nodrošina tikai pamata notāciju vizuālai OWL ontoloģijas prezentācijai un izstrādei, bez tiešas iespējas nodrošināt nozarei specifiskus pielāgojumus, piemēram, papildus sintaksi noteiktu anotāciju vizuālai prezentācijai vai pielāgotu elementu vizuālo izskatu (piemēram, virsotņu formu vai krāsu), ja ontoloģijā ir noteiktas aksiomas.

Darbā veiktā OWLGrEd paplašinājumu izstrāde balstās uz OWLGrEd rīka kodējumu GrTP/TDA rīku būves platformā (Barzdins et al., 2007), kas nodrošina rīka definīcijas un izpildes laika datu uzglabāšanu noteiktas struktūras modeļu repozitorija veidā, kā arī piedāvā vispārēju mehānismu rīka papildinājumu iestrādāšanai (izmantojot noteikta veida rīka darbības paplašinājuma punktus un to apstrādes funkcionalitāti).



1.1. attēls. OWLGrEd redaktorā veidotas diagrammas piemērs

2 Galvenie pētījuma rezultāti

2.1 OWLGrEd paplašinājumi un attīstība

Pētījumā (Dudas et al., 2018) par grafiskiem ontoloģijas attēlošanas rīkiem tika atzīts, ka ontoloģiju vizualizācijas risinājumu pielāgošana lietotāju vajadzībām ir viena no problēmām, kas pastāv šobrīd eksistējošiem rīkiem. Šī problēma tiek risināta OWLGrEd rīka kontekstā, paplašinot to ar šādām iespējām:

- Ontoloģiju vizualizācijas parametri dažādu ontoloģijas attēlošanas veidu atbalstam.
- Modulāra eksporta konfigurācija – universāls, dažādām platformām pielāgojams eksporta modulis ontoloģijas saglabāšanai kādā no OWL tekstuālās formas standartiem.
- Lietotāja definētu lauku mehānisms, kas piedāvā ērtu un vienkāršu veidu, kā papildināt OWLGrEd redaktora notācijas iespējas un vizualizēt domēnspecifiskus anotāciju apgalvojumus.
- Izteiksmju turpinājumu priekšāteikšana – universāls mehānisms, kas sniedz uz gramatikām balstītu priekšāteicēja funkcionalitāti.
- OWLGrEd paplašinājumi ar lietotājiem nepieciešamajiem paplašinājumiem un funkcionalitāti.

Šajā nodaļā tiek aprakstīti promocijas darbā izstrādātie OWLGrEd rīka papildinājumi, kas atspoguļoti (Cerans et al., 2012, 2013, 2014a, 2014b, 2015a, 2019d), (Ovcinnikova and Cerans, 2016), (Liepins et al., 2016), (Ovcinnikova et al., 2014), (Ovcinnikova 2020) publikācijās.

2.1.1 Ontoloģiju vizualizācijas parametri

Svarīgs grafisko ontoloģiju rīka aspekts ir iespēja kvalitatīvi vizualizēt ontoloģiju. OWLGrEd redaktorā ontoloģijas vizualizācija notiek 2 posmos:

- Ontoloģija tiek ielādēta redaktorā un automātiski vizualizēta, izmantojot uz diagrammām un semantiku balstītu automātisko izvietotāju;
- Lai iegūtu labāku ontoloģijas vizualizācijas kvalitāti, ir iespēja veikt ontoloģijas attēla manuālu precizēšanu.

OWLGrEd notācija nodrošina dažādu, viena un tā paša semantiskā elementa prezentāciju. OWLGrEd rīkā ir gan grafiskas, gan tekstuālas notācijas tādiem semantiskiem elementiem kā apakšklases attiecības, objektu īpašību attiecības, anotācijas un objektu īpašības kā asociācijas lomas vai kā atribūti. OWLGrEd redaktorā noklusēts princips ontoloģiju vizualizācijai ir izmantot grafisko notāciju, ja nav acīmredzamu iemeslu pārslēgties uz tekstuālo reprezentāciju.

Dažādām ontoloģijām optimālākie attēlošanas veidi var atšķirties. Piemēram, ja ontoloģijā nav daudz objektu īpašību ierobežojumu (*object property restrictions*), tad labāks veids ir attēlot tos grafiski. Savukārt, ja ontoloģijā ir daudz ierobežojumu vai ir ierobežojumi, kuru mērķa klase ir owl:Thing, tekstuāls ierobežojumu attēlošanas veids būtu piemērotāks. Bieži vien ir arī svarīgi vizualizēt datu modeli ar vai bez instancēm.

Lai atbalstītu dažādus ontoloģijas attēlošanas veidus, dažādām ontoloģijām un vajadzībām, promocijas darba ietvaros tika izstrādāts paplašināms ontoloģiju vizualizācijas parametru ietvars (Ovcinnikova and Cerans, 2016).

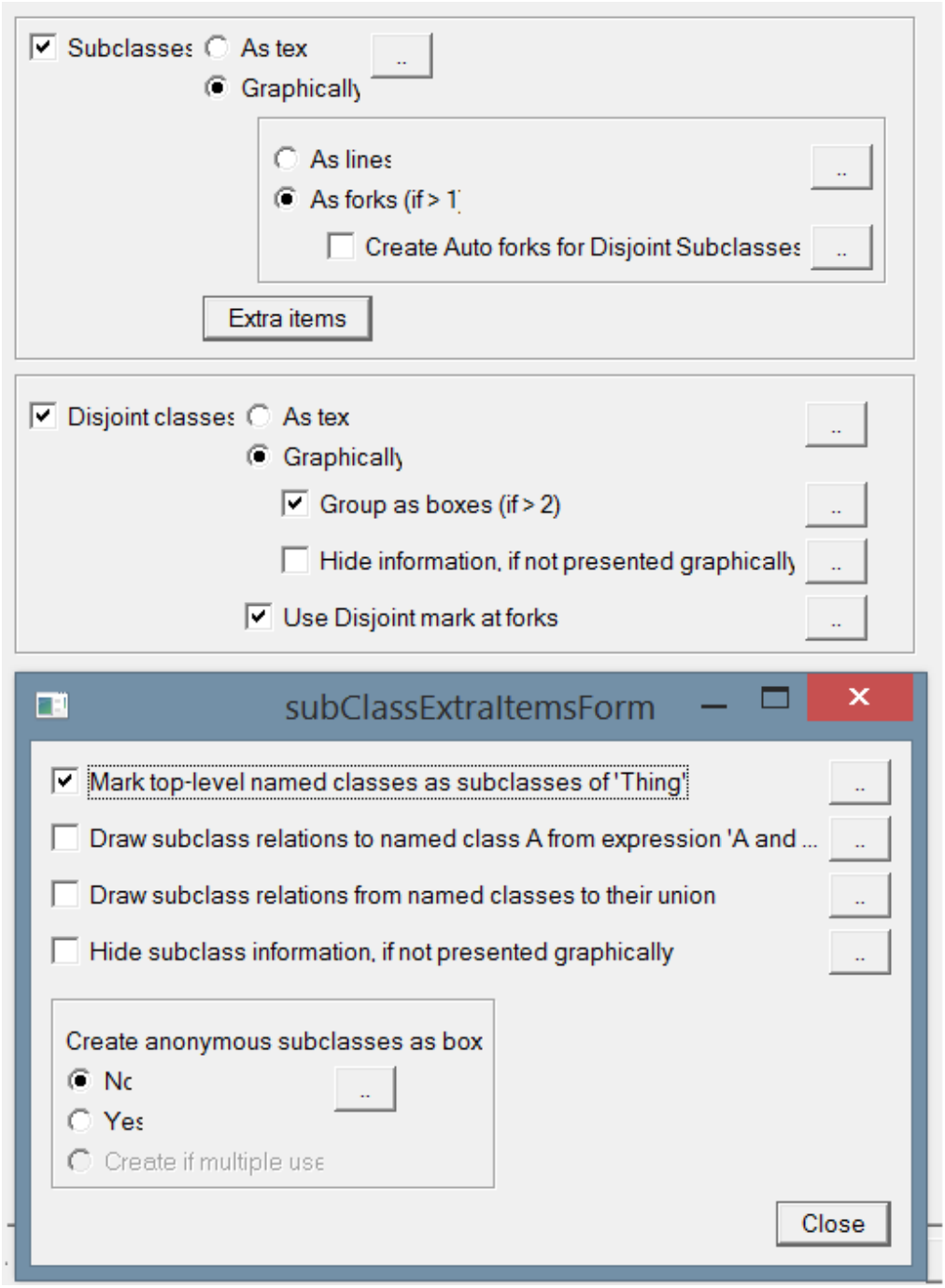
Ietvars nodrošina:

- Dažādu objektu tipu iekļaušanu vai neiekļaušanu ontoloģijas vizualizācijā (piemēram, ir iespēja vizualizēt ontoloģiju, iekļaujot vai neiekļaujot indivīdu informāciju).
- Dažādu entītiju un aksiomu tipu dažādo veidu attēlošanu (piemēram, grafiska vai tekstuāla reprezentācija).

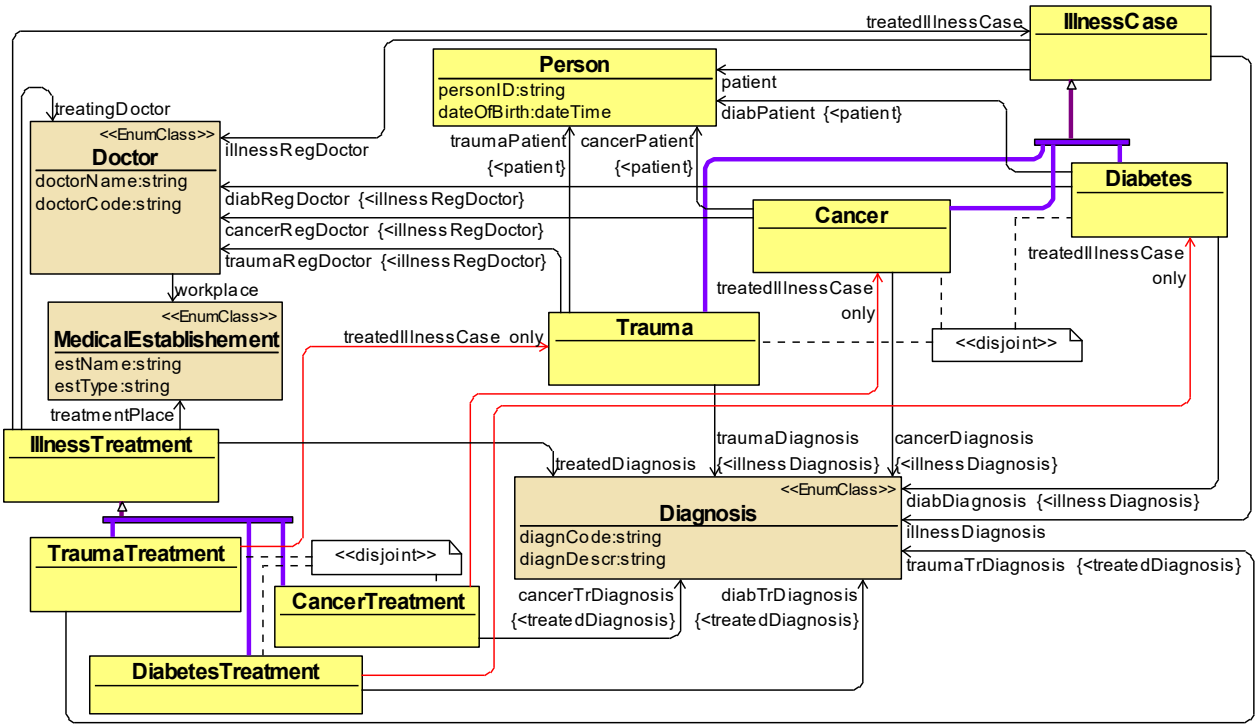
Attēlā 2.1 ir parādīts ontoloģijas vizualizācijas parametru dialoga loga piemērs ar iespēju attēlot apakšklašu attiecības tekstuāli (kā OWL Mančestras sintakses izteiksmi pie klases virsklasi aprakstošā kompartmenta) vai grafiski, kā vispārināšanu ar papildu iespēju apvienot vienas klases apakšklases zem vispārināšanas elementa.

Piemērā parādīta iespēja nepārklājošas klases (*Disjoint classes*) attēlot tekstuāli vai grafiski, kur grafiskajā notācijā var izvēlēties veidot nepārklājošo elementu specifikācijas, izmantojot speciālas grafa virsotnes. Lai atvieglotu ontoloģijas diagrammas izskatu var lietot iezīmi "Use Disjoint mark at forks", kas pievieno iezīmi *{disjoint}* pie vispārināšanas elementa, ja visas klases apakšklases ir nepārklājošas.

Piemērā tiek parādīti arī apakšklašu papildu vizualizācijas parametri, tādi kā "Mark top-level named classes as subclasses of 'Thing'", "Draw subclass relations to named class A from expression 'A and ...'", "Draw subclass relations from named classes to their union", "Hide subclass information, if not presented graphically" un iespēja noteikt, kādās situācijās izteiksmēm jāveido anonīmas klases. Lietotāju izvēlētie parametri parametru dialogā ietekmē vizuālo elementu (piemērā – apakšklases līniju) daudzumu ontoloģijas vizualizācijā.

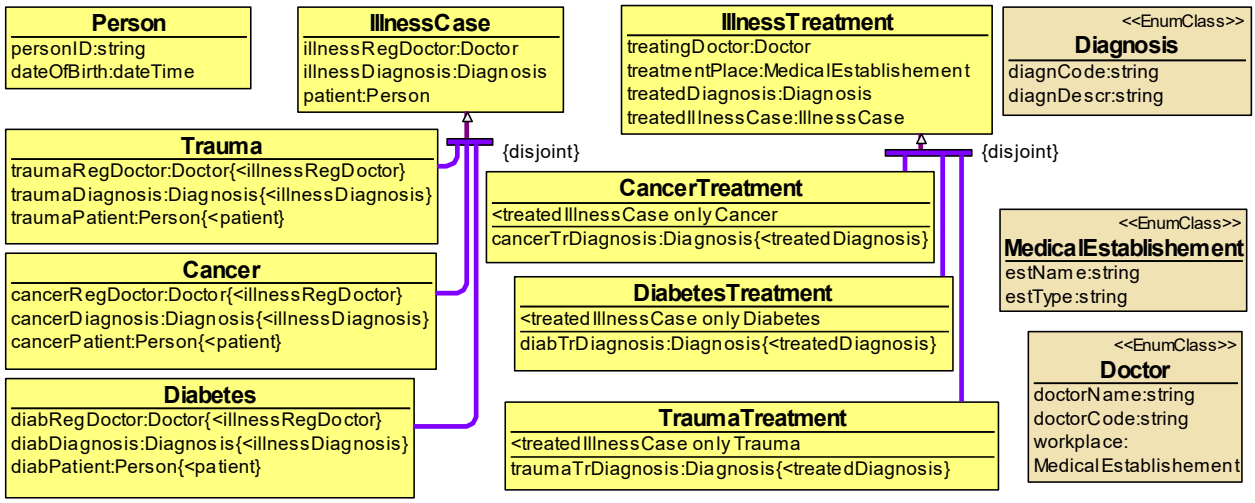


2.1. attēls. Ontoloģiju vizualizācijas parametru piemērs (Ovcinnikova and Cerans, 2016)



2.2. attēls. Piemērs no Medicīnas reģistra datu ontoloģijas (Ovcinnikova and Cerans, 2016)

Attēls 2.2 parāda piemēru no Medicīnas reģistra datu ontoloģijas, kur objekta īpašības un objektu tipa īpašību ierobežojumi ir attēloti grafiski un nepārklāšanās aksiomas ir attēlotas grafiski izmantojot nepārklājošas aksiomu virsotnes.



2.3. attēls. Piemērs no Medicīnas reģistra datu ontoloģijas, tekstuāla reprezentācija (Ovcinnikova and Cerans, 2016)

Attēlā 2.3 ir attēlots tas pats Medicīnas reģistra datu ontoloģijas piemērs kā attēlā 2.2, kur apakšklašu attiecības attēlotas grafiski, izmantojot atribūtu {disjoint} vispārināšanu, izmantojot iezīmi {disjoint} pie vispārināšanas elementa, un ar objektu īpašībām un objektu īpašību ierobežojumiem, kas attēloti tekstuāli. Iegūtā ontoloģijas reprezentācija satur mazāku elementu skaitu, kas noteiktās situācijās var būt ieguvums, bet objektu īpašības un objektu īpašību ierobežojumi tekstuālā veidā paslēpj ontoloģijas struktūru.

2.1.2 Modulāra eksporta konfigurācija

Lai pilnvērtīgi izmantotu grafisko ontoloģijas rīku, ne mazāk svarīga ir iespēja saglabāt ontoloģiju kādā no OWL tekstuāliem standartiem. Līdzās pamata ontoloģijas elementiem ir svarīgi saglabāt arī domēnspecifiskas aksiomas.

Promocijas darba ietvaros tika izstrādāta universāla uz eksporta šabloniem balstīta pieeja, kas pielāgojama dažādām platformām (Ovcinnikova, 2020).

OWLGrEd ontoloģijas eksports ir realizēts, izmantojot šablonu definīcijas, kas pievienotas grafisko elementu tipiem ontoloģijas diagrammas struktūras definīcijas ietvaros (OWLGrEd rīkā jau eksistēja specifiska ontoloģijas eksporta implementācija, bez modulāras struktūras un ērtām paplašināšanas iespējām).

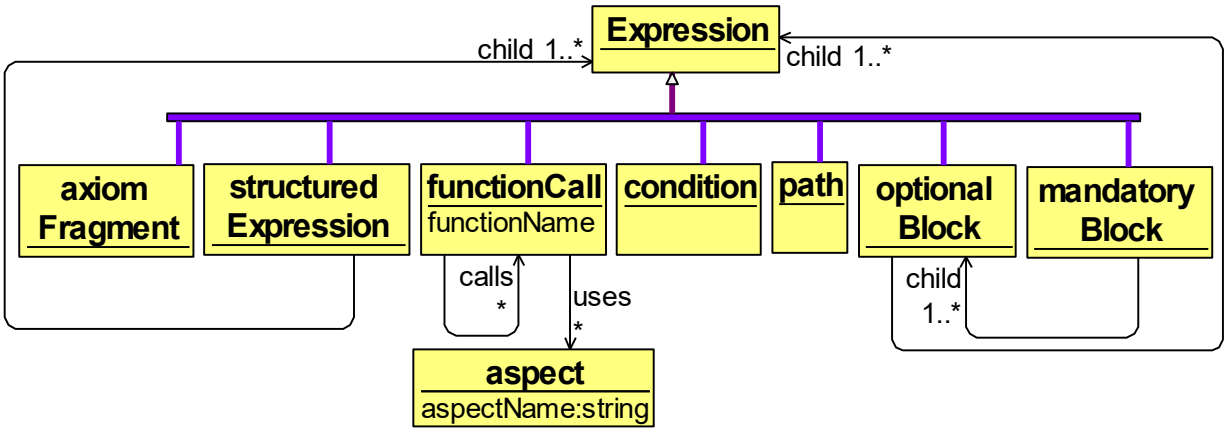
OWLGrEd rīka grafiskā abstraktā sintakse, kuras pamatā ir virsotnes, šķautnes un lauki, un OWL ontoloģijas aksiomas (kā loģiski apgalvojumi) OWL funkcionālajā sintaksē, ir ļoti dažādas struktūras. Uz šabloniem balstīta valoda ir viena no metodēm, kā sasaistīt šīs dažādās struktūras.

Uz šabloniem balstītā eksporta programmā katra ontoloģijas aksioma tiek saistīta ar vienu pamata konstrukciju grafiskajā diagrammā, uz kuru šī aksioma tieši attiecas.

OWLGrEd ontoloģiju eksporta programma ir tipisks modeļa par tekstu pārveidošanas risinājums. Eksistē valodas un rīki, kas atbalsta šo principu, piemēram, Acceleo Query Language (AQL) (WEB, a), Epsilon Generation Language (Rose et al., 2008), Xpand (WEB, ac), JET (WEB, e), MOFScript (Oldevik, 2006). Mūsu no modeļa uz tekstu veidotā transformāciju metode darbojas tieši vidē, kur tiek ieviests OWLGrEd redaktors.

Katram elementa vai kompartmenta tipam OWLGrEd ontoloģijas diagrammā ir pievienots tekstuāls šablons (vai vairāki šabloni) OWL funkcionālās sintakses aksiomu ģenerēšanai no attiecīgā tipa elementiem vai kompartmentiem. Informācija par eksporta šabloniem tiek glabāta Tag tipa instancē. OWLGrEd eksporta programma apstrādā visus diagrammas elementu un kompartmentu tipus, kuriem ir pievienota eksporta atzīme, parsē eksporta šablonu un, izmantojot parsēšanas rezultātu un ontoloģijas diagrammas datus, ģenerē OWL aksiomu funkcionālajā sintaksē.

Attēlā 2.4 parādīta OWLGrEd eksporta šablonu augšējā līmeņa struktūra. Izveidoto OWL funkcionālās sintakses prezentāciju vēlāk var pārveidot par jebkuru oficiālu tekstuālo OWL 2 sintaksi, izmantojot OWL API (Horridge and Bechhofer, 2011).



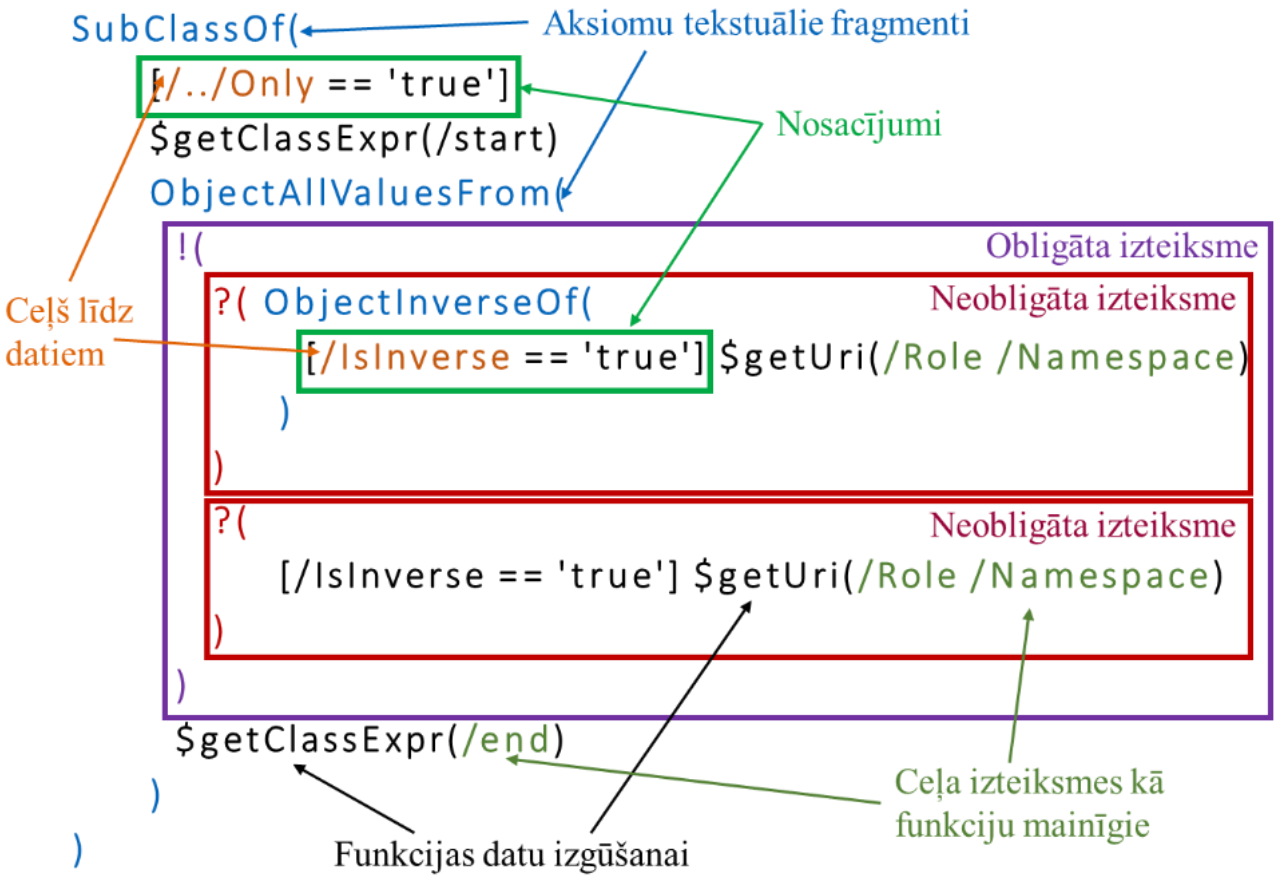
2.4. attēls. Eksporta šablonu augšējā līmeņa struktūra (Ovcinnikova 2020)

Valodas galvenās komponentes ir:

- **Aksiomu tekstuālā konstrukcija** – OWL funkcionālās sintakses aksiomas teksta daļa, piemēram, aksiomas nosaukums.
- **Strukturētas izteiksmes konstrukcija** – sākas ar teksta fragmentu (aksiomas nosaukumu), kam seko izteiksmju saraksts.
- **Ceļa izteiksmes konstrukcija** – nosaka ceļu no pašreizējās atrašanās vietas tipu struktūrā līdz vajadzīgajiem datiem.

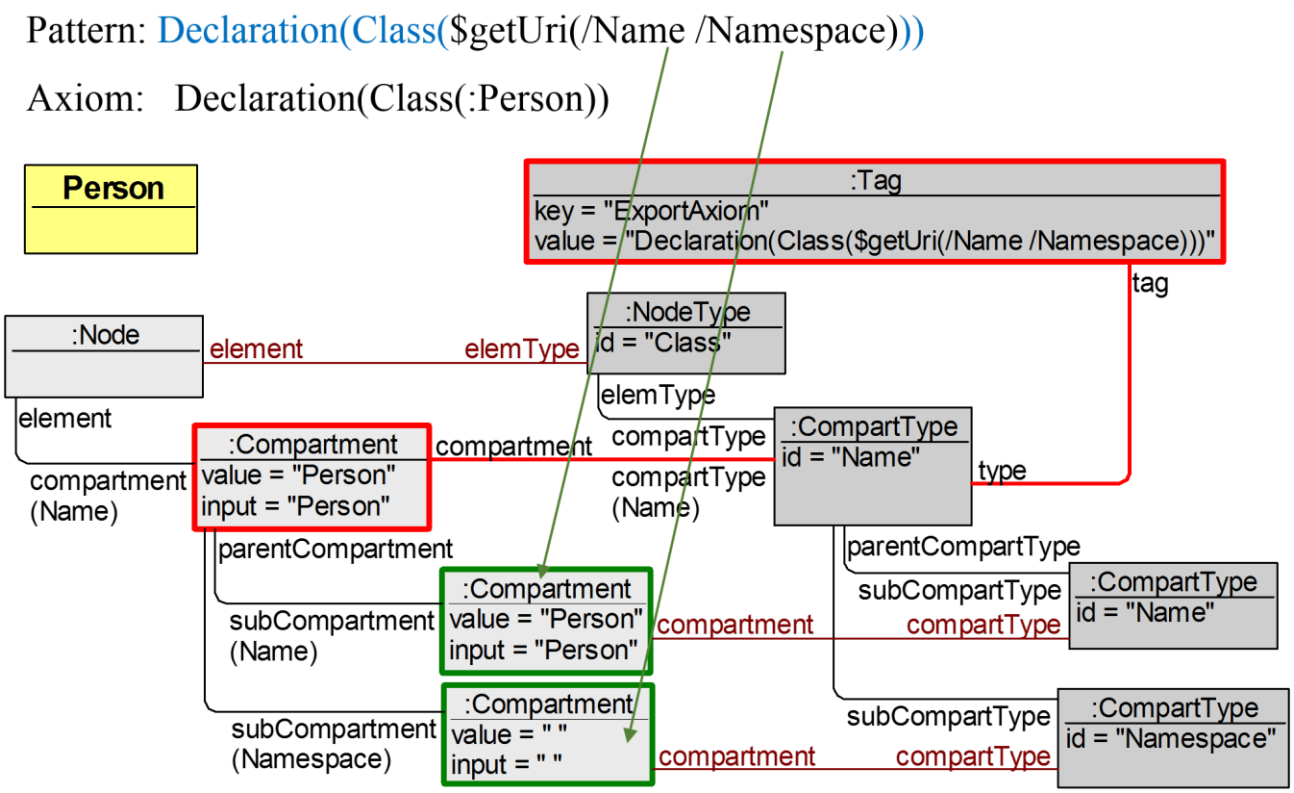
- **Funkciju konstrukcija** – izgūst datus no repozitorija, aprēķina un atdod aksiomas fragmentu vai tekstuālo vērtību.
- **Nosacījumu konstrukcija** – pārbauda, vai, pamatojoties uz dotajiem datiem, var pielietot aksiomu vai aksiomas fragmentu. Vienā nosacījumā var tikt ietverti vairāki OR tipa nosacījumi.
- **Neobligātas konstrukcijas bloks** – definē aksiomas fragmentu, kas ir neobligāts aksiomā. Neobligātas konstrukcijas bloks sastāv no vienas vai vairākām izteiksmēm. Izteiksme var būt jebkura eksporta šablonu valodas augstākā līmeņa struktūra.
- **Obligātas konstrukcijas bloks** – definē aksiomas fragmentu, kas ir obligāts aksiomā. Obligātas konstrukcijas bloks sastāv no vairākām neobligātām izteiksmēm ar nosacījumu, ka vienai no neobligātajām izteiksmēm ir jāizpildās.

Svarīga OWLGrEd eksporta šablonu valodas daļa ir funkcijas, ko izmanto, lai izgūtu un pārveidotu OWLGrEd diagrammas datus OWL aksiomas fragmentā. Funkcija var meklēt saistīto strukturālo elementu, tādu kā klases vārdu virsotnes atribūtam (kas izmantojams atribūta īpašības domēna aksiomā) vai meklēt informāciju visas diagrammas kontekstā (piemēram, pilnos URI, kas atbilst noradītiem īsiem vārdiem (funkcija $getUri), vai visu klašu vārdu sarakstu diagrammā). Ir arī funkcijas sintakses konvertēšanai, informācijas apkopošanai no vairākiem strukturālajiem elementiem un atbalsta informācijas nodrošināšanai (Ovcinnikova 2020). OWLGrEd eksporta šablonu piemērs ar atzīmētām šablonu konstrukcijām ir parādīts attēlā 2.5.



2.5. attēls. Eksporta šablonu struktūras piemērs (Ovcinnikova 2020)

Attēlā 2.6 parādīta klases deklarācijas aksiomas ģenerēšana. Klases deklarācijas aksiomas ģenerēšana sākas no kompartmenta, kas savienots ar kompartmenta tipu (:compartType), kura ID ir vienāds ar "Name". Šim kompartmenta tipam ir eksporta šablons, kas tiek glabāts saistītajā Tag instancē. Klases deklarācijas eksporta šablons ir definēts kā šāda virkne: ***"Declaration (Class ($getUri (/ Name / Namespace)))"***.



2.6. attēls. Eksporta šablona piemērs klases deklarācijai (Ovcinnikova 2020)

Šajā nodaļā aprakstītā uz šabloniem balstīta pieeja tiek izmantota OWLGrEd rīka paplašinājumu definīcijā, kas aprakstīta nodaļā 2.1.3. Izstrādāto pieeju var izmantot arī modeļu, kas veidoti citā grafiskā sintaksē GrTP/TDA modelēšanas rīku būves platformā (Barzdins et al., 2007) vai konceptuāli līdzīgā platformā kā ajoo (Sprogis, 2016), tulkošanai tekstuālajā attēlojumā. Piemēram, varētu implementēt vizuālo notāciju SHACL valodai (Knublauch and Kontokostas, 2017) RDF grafu validēšanai vai nu GrTP/TDA, vai ajoo platformā un izmantot tur uz šabloniem balstītu eksporta pieeju.

Paredzams, ka izvēlētā uz eksportiera šablonu balstītā arhitektūra ļaus to pārcelt no šobrīd implementētās Lua programmēšanas valodas (WEB, w) ar lQuery bibliotēku (Liepins, 2012) citā programmēšanas vidē (piemēram, JavaScript (WEB, d) ar jQuery (WEB, f).

2.1.3 Lietotāja definētu lauku un skatījumu mehānisms

Līdz ar vienotu grafisko ontoloģiju attēlošanas notāciju, kas tiek piedāvāta OWLGrEd rīkā, atsevišķās situācijās ontoloģijas uztveramību var veicināt to papildinoša, specifiska – tieši konkrētai ontoloģijai vai kādam ontoloģiju kopumam piemērota – notācija.

Promocijas darba ietvaros tika izstrādāts lietotāja definētu lauku un skatījumu mehānisms (Cerans et al., 2012, 2013, 2015a), (Ovcinnikova and Cerans, 2016), kas piedāvā iespēju, kā ērti un vienkārši papildināt TDA platformā veidoto rīku, tostarp arī OWLGrEd redaktora notāciju klāstu, un vizualizēt domēnspecifiskus anotāciju apgalvojumus.

Tika izveidots mehānisms, kas ļauj ērti:

- papildināt OWLGrEd redaktora notāciju ar jauniem laukiem un tiem atbilstošu semantiku;
- pielietot lietotāja definētus stila uzstādījumus laukiem, izvēles vienumiem un simboliem, kas ir atkarīgi no konkrēta teksta vai izvēles lauka vērtības;
- definēt skatījumus, kas visai diagrammai piemēro noteiktu vizuālo izskatu, izceļ svarīgus elementus, paslēpj nevajadzīgās elementu daļas un atvieglo diagrammu lasīšanu.

Lietotāja definētu lauku mehānisms ir realizēts kā redaktora paplašinājums un pašlaik ir iekļauts OWLGrEd redaktora pamata konfigurācijā.

Lietotājiem tiek piedāvāts ērts veids, kā papildināt OWLGrEd redaktoru, neietekmējot redaktora pamata struktūru, un noņemt pievienoto funkcionalitāti, kad tā vairs nav vajadzīga, neradot nekādus

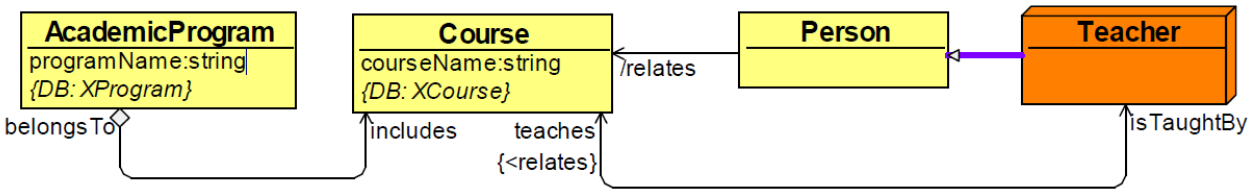
informācijas zudumus. Tiek atbalstīta arī jauno lauku pārnesamība no viena projekta uz citu, saglabājot jauno lauku informāciju teksta failā un ielādējot to jaunā projektā.

OWLGrEd redaktora notācijas paplašināšana ar lietotāja definētu lauku kopumu notiek, definējot ontoloģijas vizualizācijas profilus.

Katrs ontoloģijas vizualizācijas profils sastāv no grafisko vienumu (lauku) kopas specifikācijas, kur katrs lauks ietver:

- lauka tipu (piemēram, ievada lauks, izvēles rūtiņa);
- lauka izskatu (piemēram, redzamību un burtu izmēru);
- diagrammas simbolu un citu lauku vizuālos efektus (piemēram, simbola krāsa un forma);
- lauka semantiku (kādām OWL aksiomām vai anotāciju aksiomām atbilst lauka vērtība).

Laukiem var uzstādīt lietotāja definētu stilu – veidu, kā lauka informācija tiek attēlota diagrammā. Stilus var uzstādīt gan lietotāja definētiem laukiem, gan eksistējošiem laukiem (stilu uzstādīšana būs atkarīga no lietotāja definēto lauku izvēles vienumiem).



2.7. attēls. Lietotāja definēto notāciju piemērs (Cerans et al., 2013)

Attēlā 2.7 ir parādīts piemērs ar lietotāja definētām notācijām. Piemērā ir izmantotas šādas jaunas notācijas:

- *A:DBExpr* – datu ievades lauks klases virsotnē, ar prefiksu "{DB:" un sufiksu "}", kas grafiski ir izcelts ar slīprakstu (piemērā "*{DB: XProgram}*" ieraksts).
- *A:isImportant* – izvēles rūtiņa klases virsotnē, kas vērtības "true" gadījumā, nomaina klases kastītes vizuālo izskatu (piemērā klase "Teacher").
- *A:isComposition* – asociācijas lomu vārda izvēles rūtiņa, kas vērtības "true" gadījumā nomaina līnijas gala izskatu (rombs pie "*belongsTo*" īpašības).
- *A:isDerivedUnion* – asociācijas lomu vārda izvēles rūtiņa, kas vērtības "true" gadījumā pievieno asociācijas lomu vārdam prefiksu "/" ("*/relates*" īpašība).

Paplašinātas OWLGrEd notācijas ontoloģijā var uzskatīt par jaunām anotāciju īpašībām (*annotation property*). Anotācijas īpašību var uztvert, kā anotācijas tipu, kas piešķir vērtību ontoloģijas entītijai (piemēram, klasei, objektu vai datu īpašībai). Saglabājot ontoloģiju kādā no OWL tekstuāliem formātiem, notāciju papildinājumi tiek saglabāti kā *AnnotationAssertion* aksiomas. Piemērā izmantotās notācijas tiktu saglabātas ar tālāk norādītajām aksiomām.

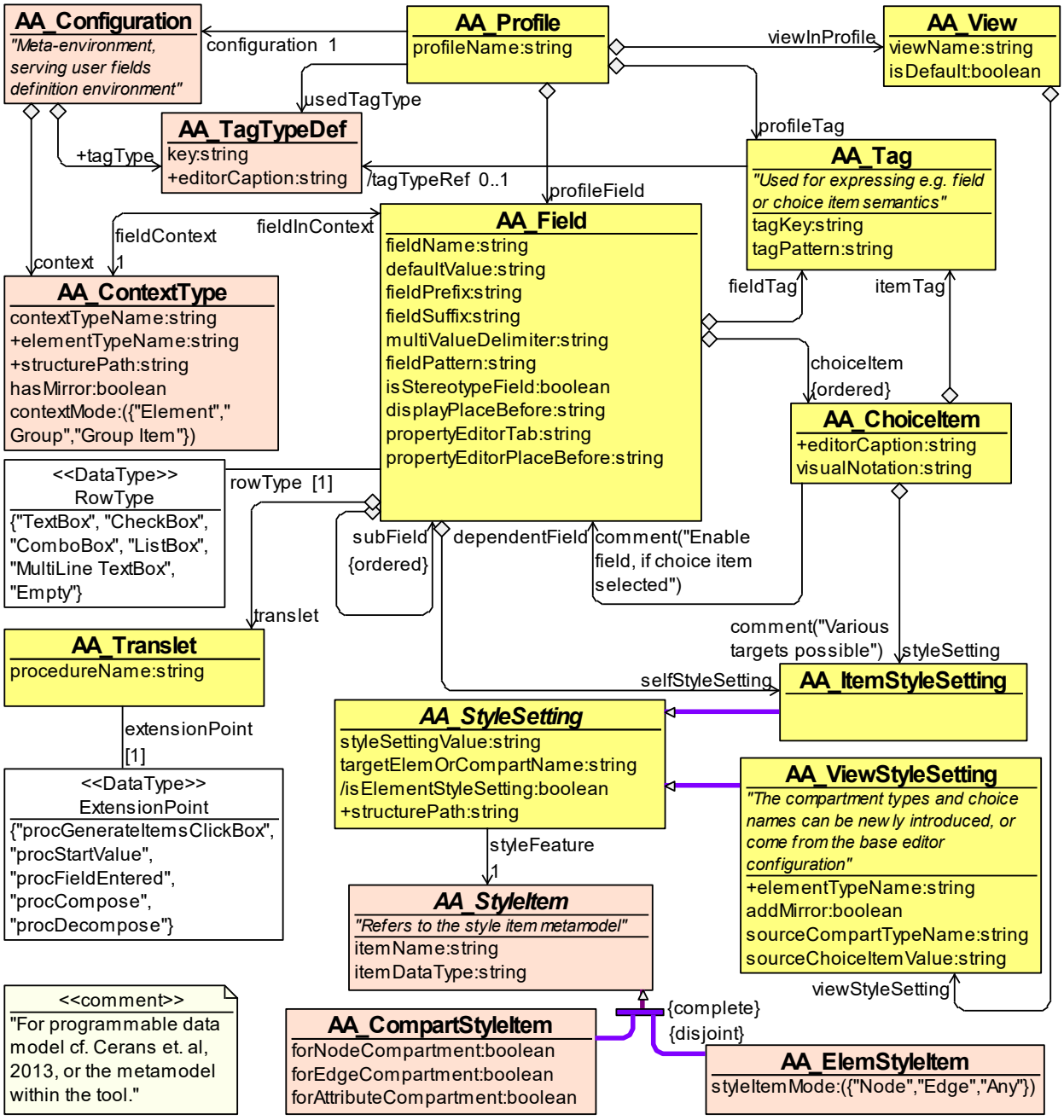
AnnotationAssertion(A:DBExpr A:AcademicProgram "XProgram")
AnnotationAssertion(A:DBExpr A:Course "XCourse")
AnnotationAssertion(A:isImportant A:Teacher "true")
AnnotationAssertion(A:isComposition A:includes "true")
AnnotationAssertion(A:isDerivedUnion A:relates "true")

Profilu specifikācija

Profilu pamata struktūra un pieejamā funkcionalitāte ir atspoguļota meta modelī attēlā 2.8. Galvenās meta modeļa klases ir:

- *AA#Profile* – paplašinājuma profils (satur jaunu lauku un skatījumu definīcijas).
- *AA#Field* – grafisks diagrammas lauks (jauns paplašinājuma lauks).
- *AA#StyleSetting* – stila efekts jaunizveidotam laukam (*selfStyleSetting*) vai citam diagrammas vienumam (elementam, laukam).

- *AA#ChoiceItem* – profilā definēts lauka izvēles vienums (piemēram, kā true/false izvēles vienumi checkbox lauka tipam, vai kā izvēles vienums combo-box sarakstā). Izvēles vienumam var tikt piekārtoti no šī vienuma atkarīgie elementa vai cita lauka stila uzstādījumi. Šajā situācijā atkarīgajam elementam vai laukam tiks uzstādīts noteikts stils, ja lietotāja definēta lauka vērtība būs vienāda ar izvēles vienumu.
- *AA#View* – stilu uzstādījumu kolekcija, kas tiek pievienota konkrētam profilam un ko var lietot katrai diagrammai atsevišķi. Pastāv iespēja uzstādīt stilus gan elementiem, gan laukiem. Skatījumi var tikt veidoti kā noklusētie – skats nav jāpielieto diagrammām, un stili tiek uzstādīti automātiski.
- *AA#Tag* – iezīmēs lauku un izvēles vienumu apstrādei. Tag elementi glabā nodefinētu semantiku lietotāja definētiem laukiem, kas tiek izmantota ontoloģijas importēšanas laikā.



2.8. attēls. Profilu struktūras meta modelis (Cerans et al., 2013)

2.1.4 Izteiksmju turpinājumu priekšāteikšana

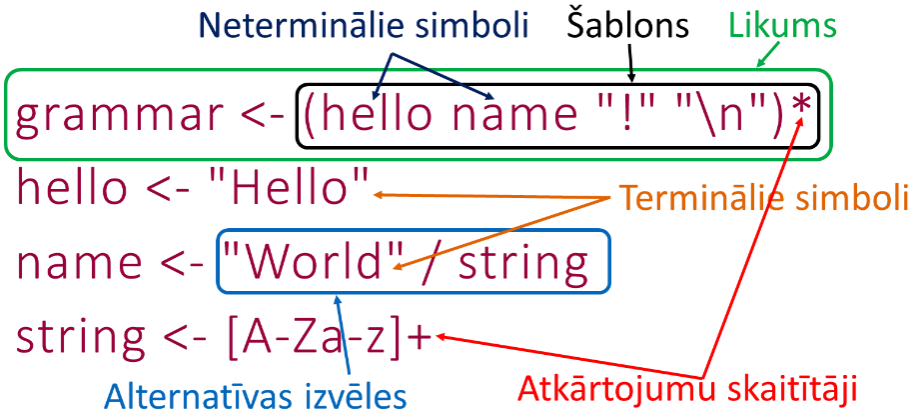
Strādājot ar redaktoru, kurā tiek lietoti teksta lauki ar izteiksmēm, kuru daļas var atsaukties uz izveidoto diagrammu, vai modeļa elementiem, vai citiem iepriekš definētiem vienumiem, būtiska ir izteiksmju turpinājumu priekšāteikšanas funkcionalitāte.

Lai atbalstītu priekšāteikšanas funkcionalitāti OWLGrEd redaktorā, tika izveidots universāls mehānisms, kas sniedz uz gramatikām balstītu priekšāteicēja funkcionalitāti un darbojas TDA vidē (Ovcinnikova et al., 2014).

Eksistē vairāki ietvari, tostarp Spoofax (Kats and Visser, 2010) un Xtext (Voelter, 2006), kas paredzēti domēnspecifisko valodu izstrādei, tostarp priekšāteikšanas iespējas definēto valodu kontekstā. Šie ietvari iekļauj gramatikas kompilācijas soli, pirms lietotājs var definēt priekšāteikšanas likumus. Tiek piedāvāta tiešāka, skaidrāka un mazāk resursietilpīga pieeja, kas ļauj definēt priekšāteikšanas likumus tādā pašā līmenī kā gramatikas definīcija.

Šīs pieejas priekšrocība ir tāda, ka priekšāteikšanas funkcionalitāti ir viegli pievienot jebkurai jaunizveidotai vai esošai gramatikai.

Šis risinājums sākotnēji tika rakstīts Lua programmēšanas valodā, izmantojot LPeg.re (WEB, i), kas ir LPEG regulāro izteiksmju sintakse. Izstrādāto pieeju var pielietot jebkurai regulāru izteiksmju veidā aprakstītai valodai. Attēlā 2.9 var redzēt LPeg.re galvenās komponentes.



2.9. attēls. LPeg.re galveno komponenšu ilustrācija (Ovcinnikova et al., 2014)

Izveidotais risinājums dod iespēju atlasīt ne tikai gramatikas simbolus, bet arī vērtības no esošās vizuāli attēlotās ontoloģijas, vai kāda cita pieejama datu avota. OWLGrEd redaktora ietvaros tika izstrādāti priekšāteicēji RDB2OWL izteiksmēm un Mančestras OWL sintaksei.

Zemāk ir dota vienkārša aritmētisko izteiksmju gramatika, kas uzrakstīta LPeg.re notācijā (Ovcinnikova et al., 2014).

```
local grammar = re.compile([[
        gMain <- Exp !.
        Exp <- (Factor (FactorOp Factor)*)
        Factor <- (Term (TermOp Term)*)
        Term <- (Number / Variable / ("(" Exp ")"))
        FactorOp <- "-" / "+"
        TermOp <- "*" / "/"
        Number <- [0-9]+
        Variable <- (([A-Za-z] / "_") ([A-Za-z] / "_" / [0-9])*)
    ]])
```

Lielākā daļa regulāro izteiksmju analizatoru apstrādā izteiksmi līdz pirmajam atrastajam ceļam, kas atbilst izteiksmei. Lai atrastu visus priekšāteikšanas turpinājumus, vispirms ir jāpanāk, lai dotās izteiksmes sintakses analīzes laikā, tiktu izstaigāti visi gramatikas ceļi, kas atbilst dotajai izteiksmei.

Vienkāršākais veids, kā to nodrošināt, ir gramatikas beigās pievienot iepriekš definētu šablonu, kas visos gadījumos atgriezīs *false* vērtību.

Ejot cauri gramatikai, visus iespējamos terminālus (aritmētisko izteiksmju gramatikā, tie būtu simboli „-", „+", „*", „/", „(", „)"), kas var kalpot kā turpinājumi, ir jāpiefiksē. Turpinājumi tiks glabāti kopā ar pozīciju izteiksmē, kurā šo turpinājumu varētu lietot. Tas tiek panākts, pievienojot katram terminālim, kas var kalpot par turpinājumu, iepriekšdefinētu šablonu, kas piefiksē turpinājuma informāciju un ļauj analizatoram virzīties tālāk pie paša simbola. Līdzīgā vaidā tiek piefiksēti arī turpinājumi no ontoloģijas vai repozitorija, kas izteiksmju gramatikā ir definēti ar netermināliem simboliem. Lai to izdarītu, gramatikai pirms neterminālā ir jāpieliek iepriekšdefinētais šablons, kas kā turpinājumus pievienos ārpus gramatikas iegūtos datus. Aritmētisko izteiksmju gramatikā varētu vēlēties kā šādus turpinājumus piedāvāt mainīgo nosaukumus.

Aritmētisko izteiksmju gramatika pēc paplašinājumu ieviešanas izskatīsies šādi (Ovcinnikova et al., 2014):

```
local grammar = re.compile([[
     gMain <- (Exp !. %parse_fail)
     Exp <- (Factor (FactorOp Factor)*)
     Factor <- (Term (TermOp Term)*)
     Term <- (Number / %parse_variable Variable /
("%parse_bracketOpen"("(" Exp %parse_bracketClose")"))
     FactorOp <- %parse_minus "-" / %parse_plus "+"
     TermOp <- %parse_multiplication "*" / %parse_devision "/"
     Number <- [0-9]+
     Variable <- (([A-Za-z] / "_") ([A-Za-z] / "_" / [0-9])*)
    ]]
 , {
        parse_fail = function (message, current_pos) return false
end,
        parse_minus = function (message, current_pos)
            add_message(messages, current_pos, "-", 90) return
current_pos end,
        parse_plus = function (message, current_pos)
              add_message(messages, current_pos, "+", 90) return
current_pos end,
        parse_multiplication = function (message, current_pos)
              add_message(messages, current_pos, "*", 90) return
current_pos end,
        parse_devision = function (message, current_pos)
              add_message(messages, current_pos, "/", 90) return
current_pos end,
        parse_bracketOpen = function (message, current_pos)
              add_message(messages, current_pos, "(", 90) return
current_pos end,
        parse_bracketClose = function (message, current_pos)
              add_message(messages, current_pos, ")", 90) return
current_pos end,
        parse_variable = function (message, current_pos)
            local variables = {"a", "b", "c", "d"}
            for i, v in pairs(variables) do
                add_message (messages, current_pos, v, 90)
            end
            return current_pos
        end
      })
```

Izstrādātā metode ļauj ne tikai atlasīt priekšāteikšanas turpinājumus, bet arī noteikt vietu izteiksmē, kur pieļauta kļūda, un piedāvāt variantus tās labošanai.

Priekšāteikšanas mehānismu, kas izveidots OWLGrEd ontoloģijas redaktora kontekstā (ieviests Lua programmēšanas valodā ar LPeg.re bibliotēku TDA platformā) ir bijis iespējams atkārtoti izmantot arī ViziQuer rīka kontekstā (Sadaļa 2.2.6) realizējot to JavaScript programmēšanas valodā ajoo platformā.

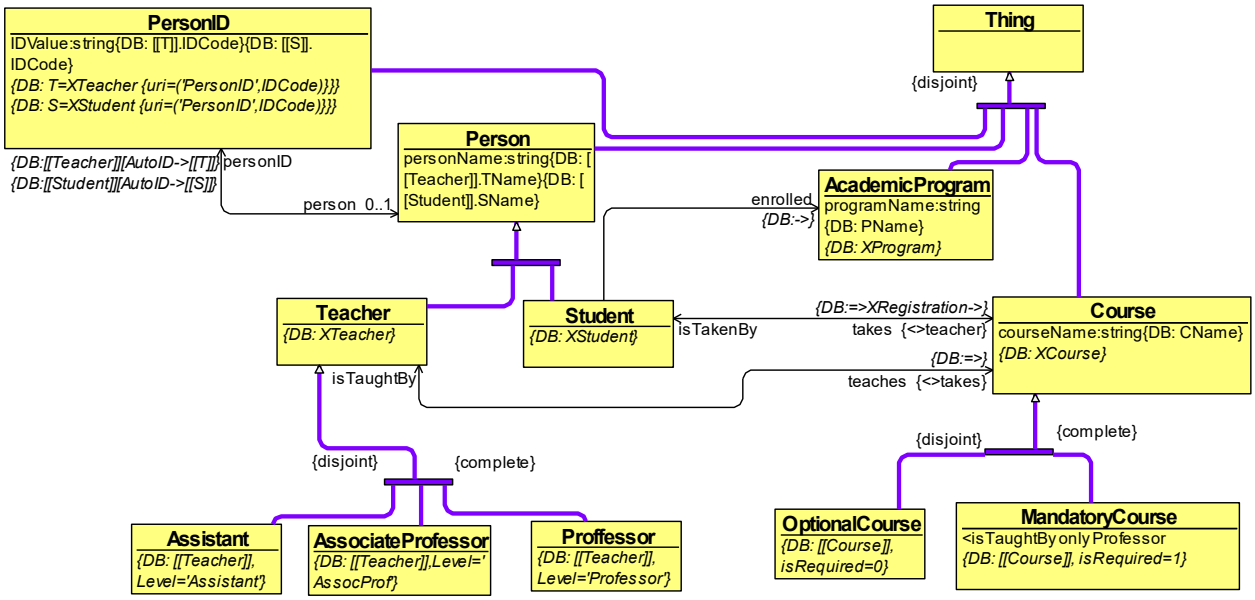
2.1.5 OWLGrEd paplašinājumi

Balstoties uz lietotāja definēto lauku un skatījumu mehānismu (Sadaļa 2.1.3) un līdzautoru izveidoto OWLGrEd redaktora paplašināšanas mehānismu, darba ietvaros ir izstrādāti vairāki konkrētiem lietojumiem specifiski paplašinājumi (Ovcinnikova and Cerans, 2016; Cerans et al., 2014a, 2014b, 2019d; Liepins et al., 2016), kas tiks tālāk aprakstīti šajā sadaļā.

Relāciju datu bāzes piesaistes paplašinājums

Semantiskas informatīvas infrastruktūras izveide ietver datu aizpildīšanu no eksistējošiem datiem, kas tipiski tiek glabāti relāciju datubāzē. Relāciju datubāzes datu pārnesē uz semantiskajām tehnoloģijām ietver attēlojumu definēšanu no relāciju datubāzes uz RDF/OWL formātu. Latvijas Universitātes Matemātikas un Informātikas institūtā tika izstrādāta RDB2OWL (Cerans et al., 2014a, 2014b; Cerans and Bumans, 2011; Bumans and Cerans, 2011) izteiksmju valoda relāciju datubāzes shēmas un ontoloģijas piesaistei. RDB2OWL ir augsta līmeņa deklaratīvā valoda, kas izmanto ontoloģijas struktūru kā pamatu attēlošanas struktūrai, glabājot datubāzes piesaistes informāciju ontoloģijas klašu, atribūtu un asociāciju lomu anotācijās, kā arī pašas ontoloģijas anotācijās.

Lai nodrošinātu vienkāršu RDB2OWL definēšanu, OWLGrEd redaktorā tika izveidots relāciju datubāzes piesaistes paplašinājums, kas ietver lietotāja definētu lauku mehānisma profilu ar laukiem, kas satur RDB2OWL specifiskās anotācijas, nodrošinot šo aksiomu saglabāšanu un ielasīšanu no ontoloģijas un RDB2OWL sintakses priekšāteikšanas funkcionalitāti.



2.10. attēls. Mini Universitātes piemērs paplašināts ar DBExpr laukiem (Cerans et al., 2013)

Attēlā 2.10 parādīts Mini Universitātes piemērs, kas paplašināts ar DBExpr laukiem RDB2OWL piesaistes anotāciju veidošanai.

Lai nodrošinātu priekšāteikšanu RDB2OWL izteiksmēs, izmantota relācijas datubāzes shēmas informācija, kas ir iepriekš ielādēta OWLGrEd redaktora repozitorijā.

Informatīvā sistēma bez programmēšanas

OBIS (*Ontology-Based Information System*), (Cerans et al., 2014a, 2014b; Zviedris et al., 2013) ir rīks, kas automātiski ģenerē pilnībā funkcionālu datu glabāšanas un izguves informācijas sistēmu, balstoties uz dotās ontoloģijas RDF shēmu. Tādējādi tiek iegūta uz tīmekļa bāzēta platforma ontoloģijas zināšanu bāzes rediģēšanai un caurskatei.

Lai informācijas sistēma darbotos korekti un būtu ērti lietojama, ar pašreizējo ontoloģijas struktūru, ko var definēt, izmantojot OWL loģiskās aksiomas, nepietiek. Informācija, kas ir svarīga uz ontoloģiju balstītas informācijas sistēmas funkcionēšanai, tika saglabāta speciālās anotācijās.. Lai atbalstītu ērtu jauno anotāciju definēšanu, ir izveidots OWLGrEd redaktora paplašinājums, kas pievieno laukus nepieciešamo anotāciju specifikācijai.

Tika ieviestās šādas anotācijas un atbilstošie OWLGrEd lauki:

- **obis:isAbstract** – norāde, ka klase ir abstrakta; izmanto, lai aizliegtu iespēju tieši ievadīt šai klasei atbilstošās instances.
- **obis:isDerived** – norāde, ka ontoloģijas īpašība ir atvasināta no citām klasēm un īpašībām; izmanto, lai aizliegtu iespēju tieši ievadīt šai īpašībai atbilstošos RDF trijniekus.
- **obis:isEnumerated** – norāde, ka klase ir klasifikators ar relatīvi stabilu instanču kopu, kas ir maināma sistēmas uzturēšanas darbību veikšanas ietvaros, bet ne ikdienas darbībās ar sistēmu.
- **obis:defaultOrder** – noklusētā datu un objektu īpašību secība klasei.
- **obis:view** – klasei definēts skatījums, satur neobligātu vārdu, paša skatījuma atribūtus, kā arī secībā sakārtotu skatījuma lauku sarakstu, ar iespējamiem atribūtiem, kas norādīti konkrētiem skatījuma laukiem.
- **obis:report** – klasei definēta atskaite, tikai datu aplūkošanas režīmā skatāmas tabulas izveidei.
- **obis:rule** – klasei, datu vai objektu īpašībai piekārtots attēlojuma veida noteikšanas vai validācijas likums.
- **obis:textPattern** – reprezentējošas teksta vērtības šablons ontoloģijas klasēm.

OBIS paplašinājums vēlāk tika sadalīts vairākās neatkarīgās komponentēs, kas var būt noderīgas atsevišķi. Šādi tika izveidoti paplašinājumi:

DefaultOrder – pievieno anotācijas, kas satur datu un objektu īpašību secību klases ietvaros. Paredz iespēju sakārtot klases datu īpašības alfabētiskā secībā.

UML_Plus – pievieno anotācijas *isAbstract*, *isDerived*, *isEnumerated* un *isComposition* laukiem. Paplašinājums arī pievieno *maxCardinality* aksiomu datu un objektu īpašībām, kuras tiek attēlotas tekstuālā notācijā un kuru kardinalitāte nav atklāti norādīta.

Valodu lauku paplašinājums

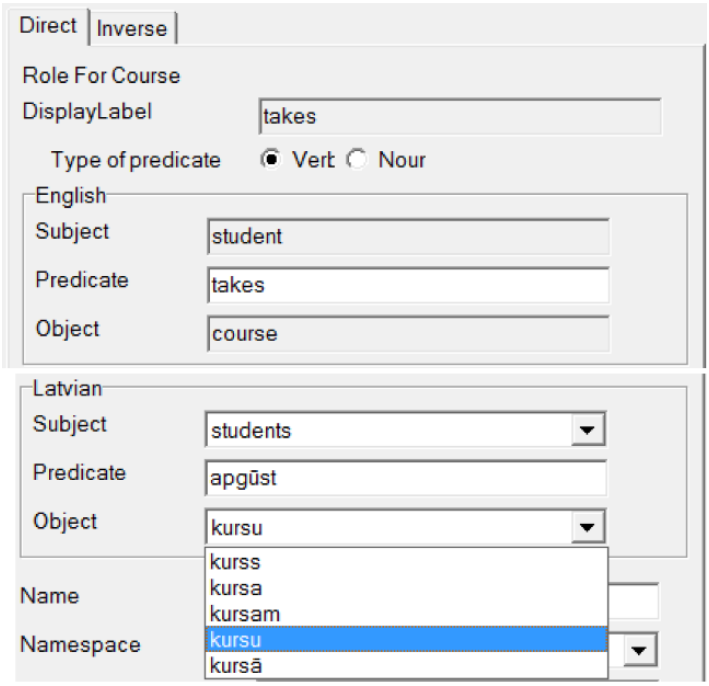
CNL (*controlled natural language*) (Kuhn, 2013; Wyner et al., 2010) jeb kontrolētas dabiskas valodas ir veidotas kā dabisko valodu apakškopa ar precīzi formulētiem gramatikas, leksikas un stila ierobežojumiem. Šie ierobežojumi parasti tiek pierakstīti kā likumi un palīdz samazināt dabiskās valodas sarežģītību un neskaidrības.

LU MII tika izveidots risinājums, kas ontoloģijai, kura ir attēlota ar UML klašu diagrammu, uzliktu CNL verbalizācijas slāni. Šis risinājums dod iespēju nozaru ekspertiem izveidot OWL ontoloģiju bez nepieciešamības būt ekspertiem OWL ontoloģiju notācijās un semantikā (Liepins et al., 2016).

Šī risinājuma ieviešanai tika izmantots OWLGrEd grafiskais redaktors, lai kontrolētu dabisko valodu ACE (Kaarel and Norbert 2007) ontoloģiju verbalizācijai un lai aprakstītu leksikalizācijas un daudzvalodības jautājumus, kas tiek risināti, izmantojot gramatisku ietvaru (*Grammatical Framework*) (Angelov and Ranta, 2009).

Kā daļa no šī uzdevuma tika izveidots valodu lauku paplašinājums, kas pievieno jaunus laukus OWLGrEd redaktoram, kas piedāvā iespējas klasei, asociācijas lomu vārdam, atribūtam un indivīdam piekārtot atbilstošos nosaukumus redaktorā implementētajās valodās. Šie lauki ir ieviesti angļu un latviešu valodai.

Ontoloģijas diagrammas aizmetņa elementam ir pievienoti lauki, kas ļauj norādīt, vai ontoloģijā esošie objekti jāparāda atbilstoši tehniskajam vārdam vai arī iezīmes (*label*) tipa anotācijām noteiktā valodā. Ievadot šo informāciju atbilstošajos laukos, ir iespējams attēlot ontoloģijas prezentāciju atbilstoši leksiskajai informācijai angļu vai latviešu valodā. Attēls 2.11 parāda objekta īpašības leksiskās informācijas ievades dialoga fragmentus.



2.11. attēls. Objekta īpašības leksiskās informācijas ievades dialoga fragmenti (Liepins et al., 2016)

RefactoringServices

RefactoringServices paplašinājums piedāvā grafisko pārstrukturēšanu, kas ļauj rediģēt grafiskas notācijas, nemainot semantiku. RefactoringServices paplašinājums ietver transformācijas, kas palīdz pielāgot ontoloģijas izskatu pēc tās sākotnējās ielādēšanas redaktorā. Paplašinājums nodrošina iespēju pārslēgties starp objektu īpašību un objektu īpašību ierobežojumu grafisko un tekstuālo izskatu diagrammas ietvaros. Turklāt paplašinājums nodrošina nepārklājošo klašu (*Disjoint classes*) transformācijas, piemēram, atsevišķu nepārklājošo klašu iezīmju pārveidošanu par {disjoint} atzīmi pie apakšklašu vispārināšanas simbola.

2.1.6 OWLGrEd rīka paplašinājumu lietojums

Lielākais paplašinājumu lietojums ir Uzņēmumu reģistra vēsturiskās datu struktūras modelēšana (Cerans et al., 2019d), kur ar OWLGrEd rīka paplašinājumiem izdevās sagatavot paplašinātu vizuālo notāciju, kas izmantota ārpus LU MII, kurā bija iespēja definēt nepieciešamo ontoloģijas struktūru. Modelis ietvēra vairāk nekā 300 klases un vairāk nekā 800 īpašības. Tika izmantoti pieci OWLGrEd paplašinājumi:

- OWLGrEd_UserFields: pamatmehānisms lietotāju lauku un prezentācijas skatu izveidei;
- RefactoringServices: transformācijas starp ontoloģijas elementu vizuālās prezentācijas alternatīvām (piemēram, objekta īpašību prezentāciju var pārslēgt starp grafisko un tekstuālo formu);
- UML_Plus: UML elementi: kompozīcija, uzskaitījuma klase, abstrakta klase un atvasināta īpašība;
- DefaultOrder: atribūta secības informācija klasē;
- OWLGrEd_Schema: apgalvojumi par īpašības pielietojamību noteiktā klases kontekstā.

Uzņēmumu reģistra vizuālas prezentācijas piemērs ir pieejams OWLGrEd redaktora mājaslapas sadaļā “Veiksmes stāsti” (WEB, o).

2.2 Vizuālo vaicājumu rīki

Semantiskas tehnoloģijas piedāvā augstāka līmeņa skatu uz datiem, salīdzinot ar relāciju datubāzēm un SQL valodu, tas paver iespēju datu analīzē vairāk iesaistīt nozaru ekspertus. Tekstuālā SPARQL (WEB, t) valodas sintakse joprojām ir pietiekami sarežģīta, lai iesaistītu semantisko datu atlasē lietotājus no ārpus informācijas tehnoloģiju nozarēm. Eksistē vairāki rīki, kas lietotājiem ļauj veidot vaicājumus par semantiskiem datiem, tādi ka OptiqueVQS (Soylu et al., 2018), QueryVOWL (Haag et al., 2015) RDF Explorer (Vargas et al., 2019) un citi (plašāk par pieejamiem rīkiem un tehnoloģijām sadaļā 1.2.2.), bet lielākā daļa no tiem ļauj atlasīt tikai vienkāršākos vaicājumus, un tiem trūkst iespējas veidot agregācijas vaicājumus, vaicājumus ar apakšvaicājumiem un sarežģītākām izteiksmēm.

Promocijas darbā tiek piedāvāts grafiskais SPARQL vaicājuma rīks ViziQuer (WEB, z), kas ļauj lietotājiem veidot vizuālus vaicājumus pār RDF datiem, tajā skaitā vaicājumus ar agregācijām, apakšvaicājumiem un bagātīgu izteiksmju notāciju. Piedāvātā notācija ļauj rīku piedāvāt darbam gan IT profesionāļiem, gan arī dažādu nozaru ekspertiem.

Šajā nodaļā tiek aprakstīta promocijas darbā izstrādātās komponentes ViziQuer/web rīka implementācijai, kas aprakstīta (Cerans et al., 2015b, 2015c, 2017, 2018a, 2018b, 2019a, 2019b, 2019c, 2021a, 2021b, 2022a, 2022b) un (Cerans and Ovcinnikova, 2016), (Ovcinnikova et al., 2023) publikācijās. Kopējā ViziQuer rīka implementācijas shēma ir pieejama (Ovcinnikova et al., 2023) publikācijā.

Sākotnējās ViziQuer rīka versijas tika veidotas TDA platformas ietvaros (Cerans et al., 2015b, 2015c), (Cerans and Ovcinnikova, 2016). Ņemot vērā vispārējo tehnoloģiju attīstību, kā arī ajoo (Sproģis, 2016) rīku veidošanas platformas pieejamību, tika pieņemts lēmums izveidot jaunu tīmekļa platformu vizuālo vaicājumu veidošanai. Darba autore galvenokārt nodarbojās ar ViziQuer/web rīka grafisko vaicājumu translāciju uz SPARQL tekstuālo formu (Cerans et al., 2017, 2018a, 2018b, 2019a, 2019b, 2019c, 2021a, 2021b, 2022a, 2022b; Ovcinnikova et al., 2023). Autore pētījuma veikšanas gaitā izstrādāja arī priekšāteikšanas funkcionalitāti SPARQL tekstuālām izteiksmēm.

Sadaļā 2.2.1. ir dots pārskats par vizuālo vaicājumu notāciju ar vaicājumu piemēriem kopā ar to tulkošanas rezultātiem uz SPARQL, kam seko tulkošanas implementācija no sadaļās 2.2.2. līdz sadaļai 2.2.5., teksta priekšāteikšanas funkcionalitātes apraksts sadaļā 2.2.6 un lietojamības novērtējumi sadaļā 2.2.7.

2.2.1 Grafiska vaicājuma veidošana

Grafisks vaicājums ViziQuer/web rīkā ir UML klašu diagrammas tipa grafs ar virsotnēm, kas apraksta datu instances šķautnēm, kuras attēlo instanču attiecības, un laukiem, kuri veido vaicājuma atlases sarakstu.

Katram vaicājumam ir viena galvenā virsotne (grafiski attēlota kā oranža kastīte) un atkarīgās virsotnes (violetas kastītes), kas saistītas savā starpā ar šķautnēm (saitēm). Grafa ietvaros strukturālas šķautnes veido koku, kurā koka sakne ir vaicājuma galvenā virsotne. Šķautnēm var tikt piešķirti īpašību vārdi no shēmas modeļa. Izšķir obligātas, neobligātas un negāciju saites. Vaicājuma daļa zem neobligātas vai negāciju saites tiek ietverta attiecīgi neobligātā vai negāciju vaicājuma blokā.

Katrai vaicājuma virsotnei var norādīt lauku – īpašību vai izteiksmi, kas tiks atlasīta vaicājumā, – un filtra nosacījumu, pēc kura tiks šķiroti atlasāmie dati. Pēc noklusējuma visi vaicājuma atlases lauki ir neobligāti.

Vaicājuma apakšvaicājumi ir attēloti ar melnu apli pie saites, kas saista pamata vaicājumu ar apakšvaicājumu. Vaicājuma grafiskā daļa zem šīs saites kopā ar saiti tiek uzskatīta par apakšvaicājumu.

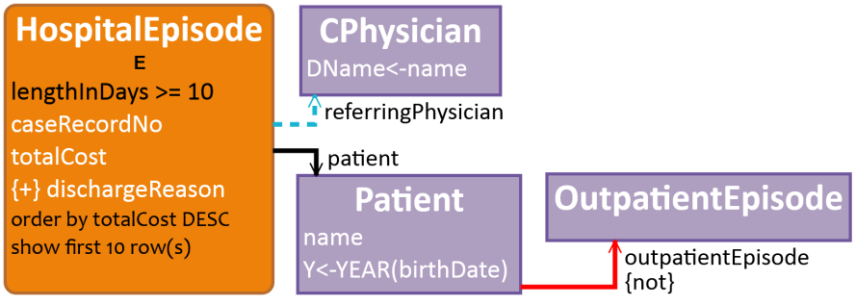
Vaicājuma vai apakšvaicājuma galvenajā virsotnē var atlasīt agregācijas atribūtus statistikas vaicājumu veidošanai. Papildus vaicājuma vai apakšvaicājuma galvenajā virsotnē var norādīt, pēc kādiem laukiem grupēt un kārtot vaicājuma rezultātus.

Vizuāla notācija ļauj izveidot vaicājumus, kuru struktūra ir bagātāka nekā koka veida struktūra. Šim nolūkam ir ieviestas atsauces saites (apzīmētas kā saites ar rombiem abos galos), kas var pievienot papildu savienojumus starp vaicājuma koka virsotnēm.

Lai paplašinātu vizuālās notācijas iespējas, ir ieviestas brīvās malas saites (apzīmētas ar ++) un vienādu datu malas saites (apzīmētas ar ==). Brīvās malas saite savieno vaicājuma virsotnes, starp tām neveidojot datu savienojumu. Vienādu datu malas saite nodrošina veidu, kā ieviest vienu un to pašu datu instanci, ko attēlo vairāk nekā viena virsotne.

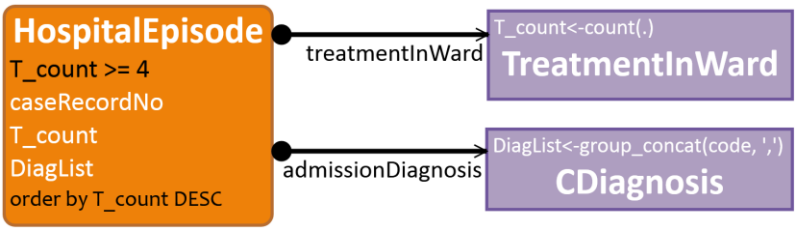
Ir arī vadības virsotnes (virsotnes bez datiem): vienības virsotnes (unit) [ ] un apvienības virsotnes (union) [+], kuras var izmantot tālākai vaicājumu strukturēšanai. Šīs virsotnes pašas neapraksta nevienu datu instanci, taču tajās var definēs laukus, nosacījumus, agregācijas un to secību, izmantojot izteiksmes, kas attiecas uz citās virsotnēs aprēķinātajiem datiem.

Attēlos 2.12 – 2.16 ir parādīti grafisko vaicājumu notācijas piemēri ViziQuer/web rīkā ar SPARQL vaicājuma tekstiem, kas ģenerēti no tiem, kur SPARQL ģenerēšana notiek, izmantojot darbā aprakstīto metodi.



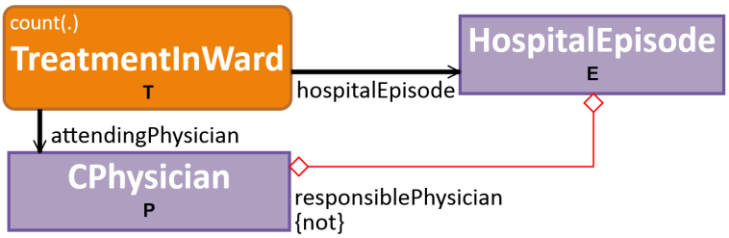

```
PREFIX : <http://lumii.lv/ontologies/2016/mini-bkus-en#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
SELECT ?caseRecordNo ?totalCost ?dischargeReason ?DName ?name ?Y WHERE{
  ?E a :HospitalEpisode.
  ?E :patient ?Patient.
  ?Patient a :Patient.
  OPTIONAL{?E :caseRecordNo ?caseRecordNo.}
  OPTIONAL{?E :totalCost ?totalCost.}
  ?E :dischargeReason ?dischargeReason.
  ?E :lengthInDays ?lengthInDays.
  OPTIONAL{?Patient :name ?name.}
  OPTIONAL{?Patient :birthDate ?birthDate.}
  OPTIONAL{
    ?E :referringPhysician ?CPhysician.
    ?CPhysician a :CPhysician.
    OPTIONAL{?CPhysician :name ?DName.}}
  FILTER NOT EXISTS{
    ?Patient :outpatientEpisode ?OutpatientEpisode.
    ?OutpatientEpisode a :OutpatientEpisode.}
  BIND(YEAR(xsd:dateTime(?birthDate)) AS ?Y)
  FILTER(?lengthInDays >= 10)
} ORDER BY DESC(?totalCost) LIMIT 10
```

2.12. attēls. ViziQuer/web vaicājumu piemērs: Atlasīt 10 visdārgākās slimnīcas epizodes, kas ilgst vismaz četras dienas, kurām ir norādīts izrakstīšanas iemesls un pacientam nav nevienas ambulatorās epizodes; uzskaitīt epizodes ieraksta kārtas numuru, kopējās izmaksas un izrakstīšanas iemeslu, pacienta vārdu un dzimšanas gadu, kā arī nosūtošā ārsta vārdu, ja norādīts (Ovcinnikova et al., 2023; Cerans et al., 2017, 2018a)



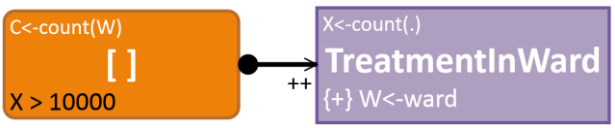

```
PREFIX : <http://lumii.lv/ontologies/2016/mini-bkus-en#>
SELECT ?caseRecordNo ?T_count ?DiagList WHERE{
  ?HospitalEpisode a :HospitalEpisode.
  {SELECT ?HospitalEpisode (COUNT(?TreatmentInWard) AS ?T_count) WHERE{
    ?HospitalEpisode :treatmentInWard ?TreatmentInWard.
    ?TreatmentInWard a :TreatmentInWard.}
    GROUP BY ?HospitalEpisode
  }
  {SELECT ?HospitalEpisode
              (GROUP_CONCAT(?code; SEPARATOR=',') AS ?DiagList) WHERE{
    ?HospitalEpisode :admissionDiagnosis ?CDiagnosis.
    ?CDiagnosis a :CDiagnosis.
    OPTIONAL{?CDiagnosis :code ?code.}}
    GROUP BY ?HospitalEpisode
  }
  OPTIONAL{?HospitalEpisode :caseRecordNo ?caseRecordNo.}

  FILTER(?T_count >= 4)
}
ORDER BY DESC(?T_count)
```

2.13. attēls. ViziQuer/web vaicājumu piemērs: Atlasīt visas slimnīcas epizodes ar vismaz 4 ārstniecības nodaļām, parādīt epizožu ierakstu skaitu un ārstniecības nodaļu skaitu; atlasīt dilstošā secībā pēc ārstniecības nodaļu skaita (Ovcinnikova et al., 2023; Cerans et al., 2017, 2018a)



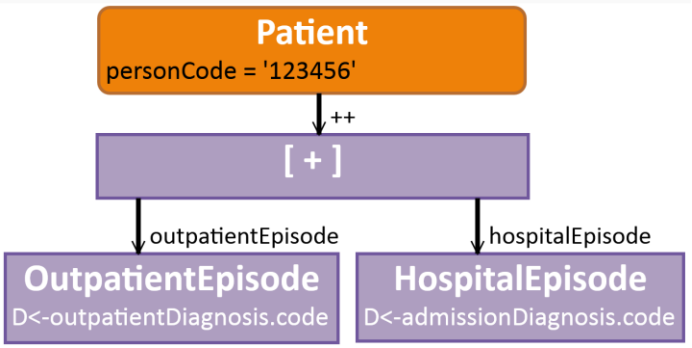

```
PREFIX : <http://lumii.lv/ontologies/2016/mini-bkus-en#>
SELECT (COUNT(?T) AS ?T_COUNT) WHERE{
  ?T a :TreatmentInWard.
  ?T :attendingPhysician ?P.
  ?P a :CPhysician.
  ?T :hospitalEpisode ?E.
  ?E a :HospitalEpisode.
  FILTER NOT EXISTS{?E :responsiblePhysician ?P.}
}
```

2.14. attēls. ViziQuer/web vaicājumu piemērs: Uzskaitīt ārstniecības nodaļas, kurās ārstējošais ārsts nav atbildīgais par nodaļas slimnīcas epizodi (Ovcinnikova et al., 2023; Cerans et al., 2017, 2018a)




```
PREFIX : <http://lumii.lv/ontologies/2016/mini-bkus-en#>
SELECT (COUNT(?W) AS ?C) WHERE{
  {SELECT ?W (COUNT(?TreatmentInWard) AS ?X) WHERE{
    ?TreatmentInWard a :TreatmentInWard.
    ?TreatmentInWard :ward ?W.}
    GROUP BY ?W
  }
  FILTER(?X > 10000)
}
```

2.15. attēls. ViziQuer/web vaicājumu piemērs: Saskaitīt visas nodaļas (atribūta nodaļa vērtības), kurās ir vairāk nekā 1000 ārstniecības gadījumu (Ovcinnikova et al., 2023; Cerans et al., 2017, 2018a)




```
PREFIX : <http://lumii.lv/ontologies/2016/mini-bkus-en#>
SELECT ?D WHERE{
  ?Patient a :Patient.
  {
    ?Patient a :Patient.
    ?Patient :hospitalEpisode ?HospitalEpisode.
    ?HospitalEpisode a :HospitalEpisode.
    OPTIONAL{?HospitalEpisode :admissionDiagnosis/:code ?D.}
  }
  UNION
  {
    ?Patient a :Patient.
    ?Patient :outpatientEpisode ?OutpatientEpisode.
    ?OutpatientEpisode a :OutpatientEpisode.
    OPTIONAL{?OutpatientEpisode :outpatientDiagnosis/:code ?D.}
  }

  ?Patient :personCode '123456' .
}
```

2.16. attēls. ViziQuer/web vaicājumu piemērs: Atrast diagnožu kodus, kas noteiktam pacientam tiek noteiktas kā slimnīcas epizodes diagnozes vai kā ambulatorās epizodes diagnozes (Ovcinnikova et al., 2023; Cerans et al., 2017, 2018a)

Vizuāls vaicājums tiek izveidots un palaists konkrētas datu shēmas (datu modeļa) kontekstā, kas nodrošina entītiju vārdnīcu, sasaistot entītijas lokālo nosaukumu un prefiksu ar pilnu entītijas IRI. Datu shēma nodrošina vaicājuma priekšāteikšanas iespējas vizuālā rīka lietotājiem, kā arī ļauj vizuālajā notācijā izmantot īsos apzīmējumus, nevis to pilnos IRI.

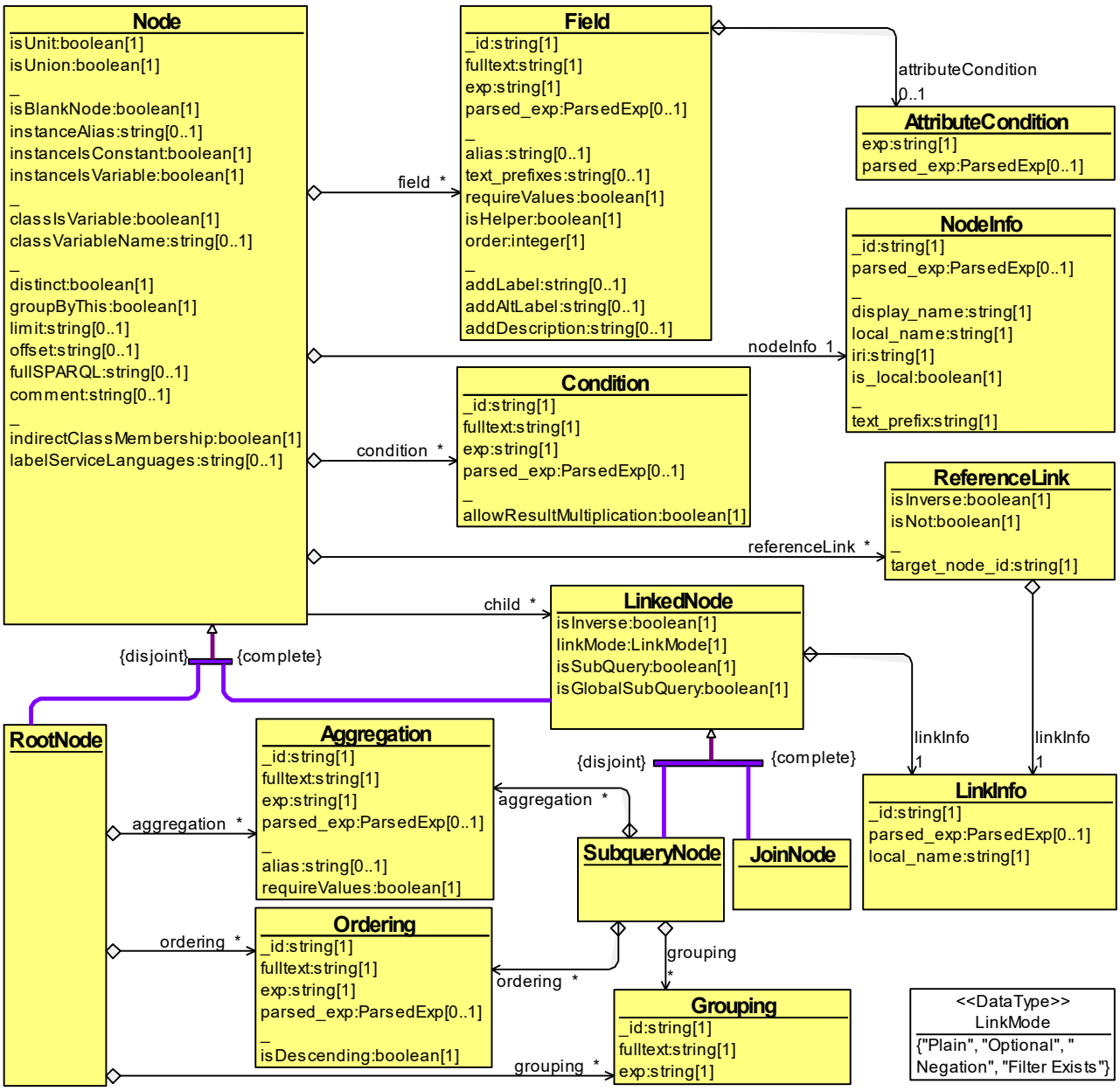
2.2.2 Vaicājuma struktūras attēlojums: abstraktās sintakses koks un simbolu tabula

Kad grafiskais vaicājums ir izveidots, pirmais solis vaicājuma vizuālās formas tulkošanā uz SPARQL ir izveidot tam abstrakto sintakses koku un atbilstošo simbolu tabulu.

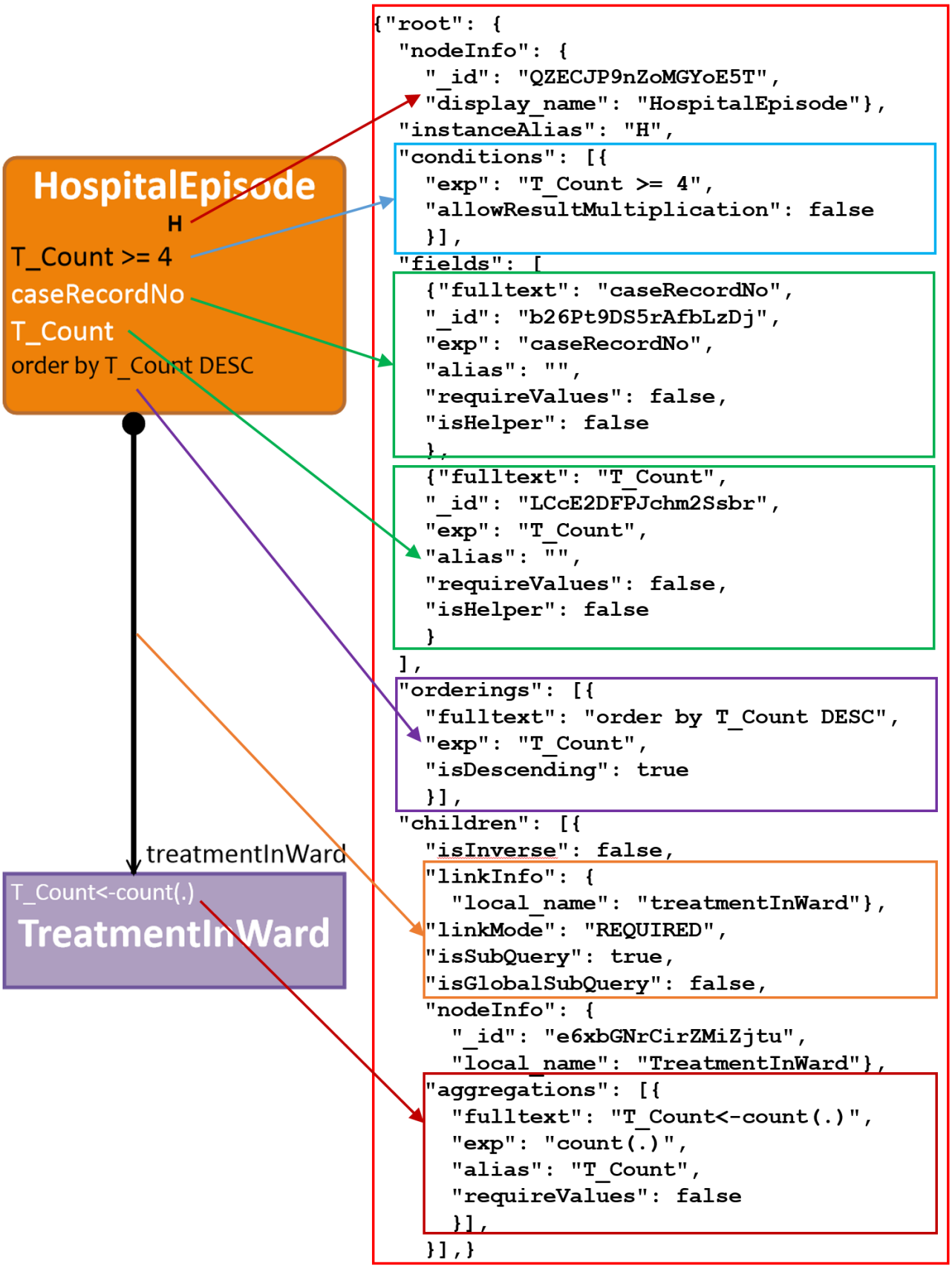
Attēlā 2.17. parādīta ViziQuer/web vaicājuma abstraktā sintakses koka struktūra. Struktūras centrālais elements ir Virsotne (*Node*). Katra virsotne, izņemot vaicājuma saknes virsotni, pieder *LinkedNode* klasei, kur ir specificētas gan virsotnes īpašības, gan tās piesaiste vecāka virsotnei vaicājuma struktūrā. Atbilstoši ienākošās saites veidam LinkedNode virsotnes tiek sadalītas *JoinNode* un *SubqueryNode* klasēs, kur apakšvaicājuma virsotnei var būt papildu informācijas vienumi (piemēram, agregācijas, grupējumi un kārtojumi).

Katram grafiskajam elementam (virsotnēm, saitē m un virsotņu/saišu kompartmentiem: laukiem, nosacījumiem, agregācijām, grupējumiem, kārtojumiem) ir atribūts _id, kas tehniski identificē elementu; virsotnēm atribūts _id tiek saglabāts *NodeInfo* instancē, bet saitēm – *LinkInfo* instancē.

Visiem diagrammas elementiem ir atribūts *parsed_exp*, kas satur elementā ievietotās teksta izteiksmes izparsētu formu, un no tā tiek nodrošinātas atsauces uz datu modeli.



2.17. attēls. ViziQuer/web grafiskā vaicājuma abstraktās sintakses implementācijas modelis (Ovcinnikova et al., 2023)



2.18. attēls. ViziQuer/web grafiskais vaicājums un tā abstraktais sintakses koks (Ovcinnikova et al., 2023)

Attēlā 2.18. ir parādīts ViziQuer grafiskā vaicājuma piemērs abstraktā sintakses koka notācijā.

Vizuālā vaicājuma notācija ļauj atsaukties gan uz datu modeļa elementiem (klasēm un īpašībām), gan uz nosaukumiem, kas ir ieviesti dažādās vaicājuma daļās (piemēram, virsotnes, lauka un agregācijas vārdi). Simbolu tabula palīdz atšķirt dažādās lomas, kuras var ieņemt vārds vaicājumā, kā arī, ja

nosaukums ir no datu modeļa, tiek apkopota ar to saistītā informācija, kas var būt svarīga SPARQL vaicājuma ģenerēšanai.

Simbolu tabula ir strukturēta kontekstos, kas atbilst vaicājuma virsotnēm. Katrai virsotnei tiek uzturēta no tās redzamā vārdu kopa, katram vārdam var būt viens vai vairāki apraksti, kas apzīmē iespējamās dažādās vārda nozīmes.

2.2.3 Izteiksmes sintakse un parsēšana

Lai realizētu ViziQuer/web rīka izteiksmju translāciju uz korektu SPARQL, promocijas darba ietvaros tika izveidota izteiksmju gramatika un parsētājs. Par pamatu ViziQuer/web izteiksmju gramatikai tika paņemta SPARQL 1.1 gramatikas apakšdaļa, kas sākas no *Expression* izveduma (WEB, u). Gramatika tika pielāgota un paplašināta, lai atbalstītu ViziQuer/web notāciju.

ViziQuer/web izteiksmēs nav skaidra SPARQL mainīgo apzīmējuma, kas veidots, izmantojot apzīmējumu "?" (izņemot atsevišķus īpašību mainīgos, kas nav iekļauti bagātīgākās izteiksmes struktūrās). Vietās, kur SPARQL gramatika pieļauj mainīgo, ViziQuer/web gramatika atļauj izmantot:

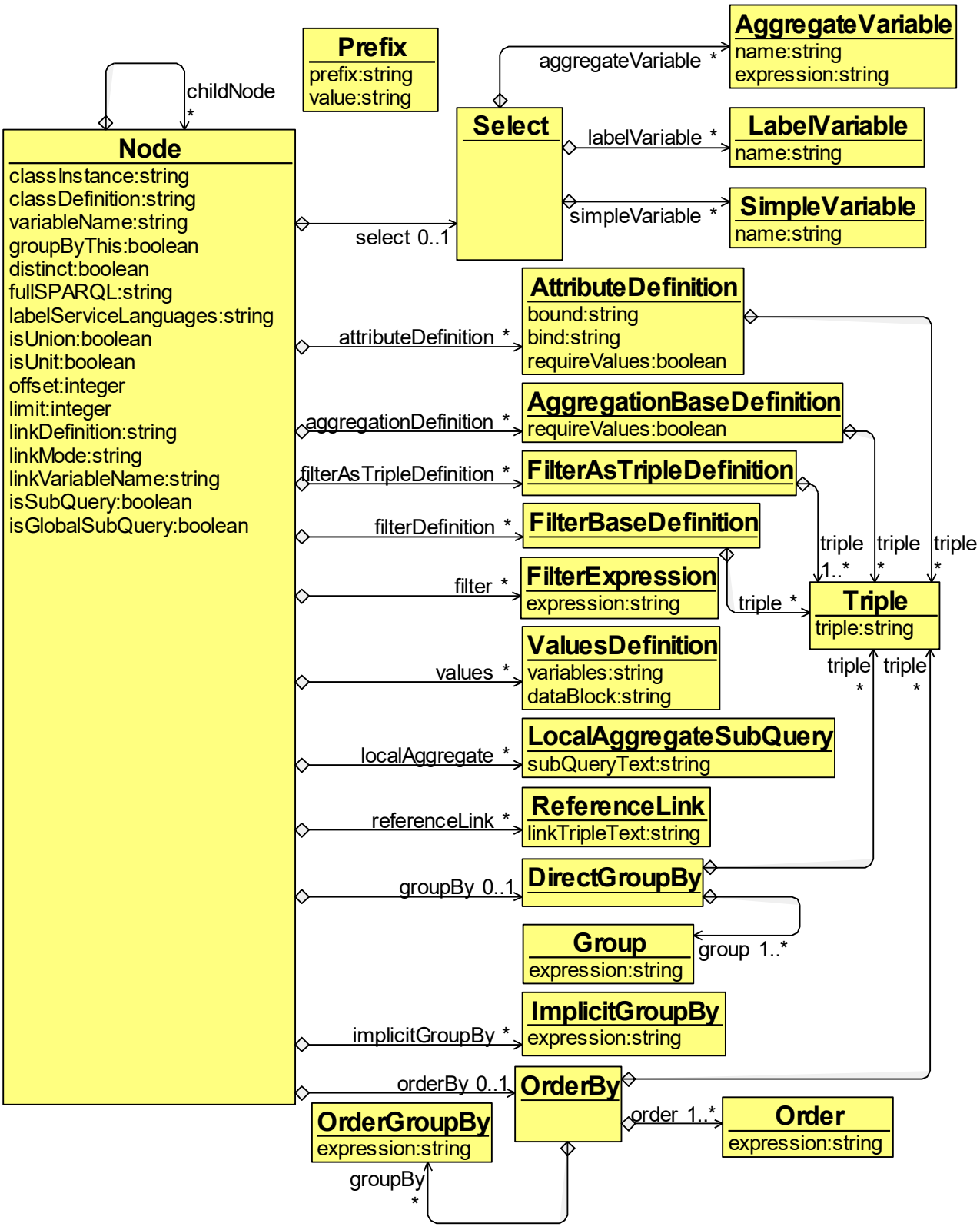
- atsauci uz virsotni, lauku vai vaicājuma mainīgo vai
- ceļa izteiksmi (parasti īpašības/atribūta vārdu), kas atbilst pārim:
  - SPARQL trijnieks, kas saista izteiksmes atsauces mainīgo (parasti mainīgo, kas atbilst virsotnei), izmantojot ceļa izteiksmi, ar jaunu SPARQL mainīgo;
  - izteiksmei, kas ietver izveidoto SPARQL mainīgo.

ViziQuer izteiksmes notācijā ir ieviesti arī sintakses saīsinājumi (Ovicinnilkova et.al. 2023) piemēram, alternatīvas virknes salīdzināšanai ar šablonu, kas atļauj SPARQL REGEX izteiksmi rakstīt infiksa formā, izmantojot SQL stila notāciju ~, ~* (piemēram, *REGEX(?ward, "1-2$"), kur ward ~ '1-2$'* apzīmē līdzvērtīgas izteiksmes). Apakšvirkni *SUBSTRING(ward,1,3)* var norādīt, izmantojot vienkārši tās garumu iekavās (piemēram, *ward[3]*).

Paplašinātā gramatika tika izmantota, lai izveidotu JSON kodētu parsētāja koku noteiktai izteiksmei. Parsētāja funkcionalitāti ViziQuer rīkā veic PEG.js bibliotēka (WEB, q). Gramatikas termiem ir pievienoti likumi, kas identificē nosauktos elementus (atsauces uz datu modeļa elementiem vai vaicājumā ieviestiem vārdiem) izteiksmju tekstā. Pēc izteiksmes parsētāja koka izveidošanas, nosauktajiem elementiem tiek veikta izmantotās nozīmes piemeklēšana, to veida un datu tipa piešķiršana.

2.2.4 SPARQL vaicājumu ģenerēšanas modelis

Nākamajā solī ViziQuer/web grafiskā vaicājuma abstraktās sintakses koks tiek pārtulkots par SPARQL vaicājumu ģenerēšanas koku, ņemot vērā abstraktās sintakses koka struktūru un simbolu tabulu. Attēlā 2.19. parādīts SPARQL vaicājumu ģenerēšanas modelis.



2.19. attēls. SPARQL vaicājumu ģenerēšanas modelis (Ovcinnikova et al., 2023)

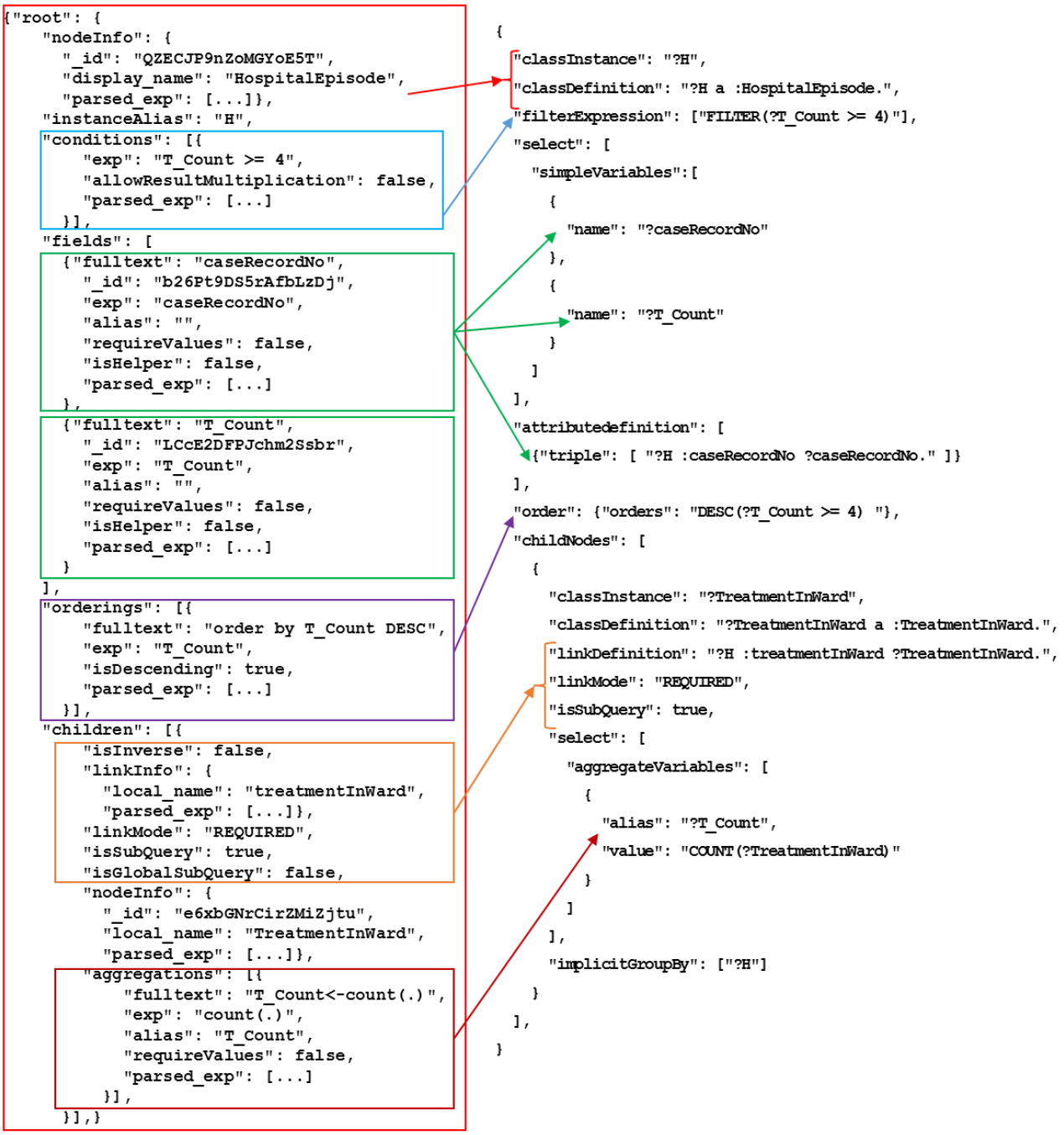
Pamata klase ir *Node* klase, kas attēlo ViziQuer grafiskā vaicājuma virsotni. Tā kā ViziQuer vaicājums veido koka grafu (ko veido nereferenču saites), virsotnei var būt piekārtotas apakšvirsotnes. Modeļa Node klasei ir virkne pakārtotu klašu, kas atspoguļo dažādus SPARQL vaicājuma fragmenta aspektus, kas saistīti ar katru virsotni.

Select – ietver mainīgos, kas nāk no virsotnes un ir iekļauti vaicājuma (vai apakšvaicājuma) atlases klauzulā. Sarakstā ir vienkārši mainīgie, iezīmju mainīgie un agregācijas mainīgie.

AttributeDefinition satur SPARQL klauzulas, kas apraksta virsotnes atribūta lauku. *AggregationBaseDefinition* satur SPARQL klauzulas, kas apraksta virsotnes agregācijas lauku. *FilterAsTripleDefinition* ietver SPARQL klauzulu, kas apraksta virsotnes nosacījuma lauku, kas attēlots kā trijnieka šablons. *FilterBaseDefinition* un *FilterExpression* satur SPARQL klauzulas, kas apraksta virsotnes nosacījumu lauku. *FilterExpression* sastāv no filtra izteiksmes un trijnieka šabloniem, kas apraksta visas nosacījuma izteiksmē iekļautās īpašības. *FilterBaseDefinition* sastāv no trijnieku šabloniem, kas apraksta visas nosacījumu izteiksmē iekļautās īpašības, kas jāpievieno SPARQL tekstam ārpus FILTER klauzulas. *ReferenceLink* satur trijnieku šablonu klauzulu, kas apraksta īpašības starp divām virsotnēm.

DirectGroupBy satur sadaļu, kas apraksta lietotājam pievienotās grupēšanas izteiksmes. *ImplicitGroupBy* satur sadaļu, kas satur grupēšanu izteiksmēs, kas tiek savāktas automātiski no vaicājuma laukiem. *OrderBy* satur sadaļu, kas apraksta secības izteiksmi.

Attēlā 2.20. ir parādīta SPARQL vaicājumu ģenerēšanas modeļa izveide.



2.20. attēls. ViziQuer vaicājuma abstraktās sintakses koks un SPARQL vaicājumu ģenerēšanas modeļa koks (Ovcinnikova et al., 2023)

2.2.5 SPARQL vaicājumu teksta ģenerēšana

Lai pabeigtu vizuālā vaicājuma tulkošanu uz SPARQL, pēdējais solis ir uzģenerēt SPARQL teksta vaicājumu no SPARQL vaicājumu ģenerēšanas koka.

SPARQL teksts no SPARQL vaicājumu ģenerēšanas koka tiek ģenerēts noteiktā secībā. Pirmkārt, visu vaicājumā izmantoto prefiksu definīcijas tiek izgūtas no datu modeļa un ievietotas SPARQL tekstā. Pēc tam tiek ģenerēta SPARQL vaicājuma SELECT klauzula, kas iekļauj sevī vienkāršos mainīgos, agregācijas mainīgos un iezīmju mainīgos no visām virsotnēm, kas nav zem negācijas vai apakšvaicājuma saitēm.

Vaicājuma WHERE klauzula tiek veidota, savācot tās tekstuālo formu, apstaigājot SPARQL vaicājuma abstrakto sintakses koku, sākot no saknes virsotnes, katram vaicājuma fragmentam savācot SPARQL kodu secībā, kas ir sadalīta trīs daļās:

- 1. posms. Sākotnējā struktūra. Fragmentu virsotņu ienākošās saites, virsotņu klases apgalvojumi, pozitīvas atsauces saites un filtri kā trijnieki starp fragmentu virsotnēm un virsotnēm virs fragmenta.
- 2. posms. Obligātie un neobligātie apakšvaicājumi pie fragmentu virsotnēm (ar rekursīvi izsauktu SPARQL vaicājumu ģenerēšanu).
- 3. posms. Galvenā apstaigāšana: Katrai fragmenta virsotnei, sākot no fragmenta galvenās virsotnes, veicot meklēšanu dziļumā:
    - vispirms tiek apstrādātas tālākas bērnu virsotnes fragmentā (SPARQL fragmenti, kas atbilst dziļākām virsotnēm vaicājuma tekstā, tiks iekļauti vispirms);
    - tiek izveidoti SPARQL teksti atbilstoši neobligātajām (*OPTIONAL*) saitēm un virsotnei piesaistītajiem UNION blokiem;
    - virsotnē ierakstīto lauku (atribūtu) informācijai atbilstošie SPARQL fragmenti;
    - filtru informācijas SPARQL fragmenti;
    - agregāciju SPARQL bāzes trijnieku definīcijas, kas apraksta agregācijas izteiksmē iesaistītās īpašības/atribūtus.

Galvenā vaicājuma vai apakšvaicājuma apstrādes beigās SPARQL tekstam tiek pievienotas trijnieku modeļa definīcijas no izteiksmēm Order by un Group by un tiek pievienotas pašas klauzulas GROUP BY, ORDER BY, OFFSET un LIMIT.

SPARQL koda fragmentu savākšanas secība ir svarīga SPARQL konstrukciju pareizai izvietošanai (piemēram, MINUS vai BIND), turklāt saprātīga modeļu secība vaicājumā var uzlabot izveidoto SPARQL vaicājumu lasāmību, kā arī palīdzēt vaicājumu izpildes programmām izveidot saprātīgu vaicājuma izpildes plānu.

Attēlā 2.21. parādīts teksta SPARQL vaicājuma piemērs, kas ģenerēts no izmantotās SPARQL abstraktās sintakses struktūras.

```
PREFIX : <http://lumii.lv/ontologies/2016/mini-bkus-en#>
SELECT ?caseRecordNo ?T_count WHERE{
  ?H a :HospitalEpisode.
  {SELECT ?H (COUNT(?TreatmentInWard) AS ?T_count) WHERE{
    ?H :treatmentInWard ?TreatmentInWard.
    ?TreatmentInWard a :TreatmentInWard.}
    GROUP BY ?H }
  OPTIONAL{?H :caseRecordNo ?caseRecordNo.}
  FILTER(?T_count >= 4)
} ORDER BY DESC(?T_count)
```

2.21. attēls. Tekstuālais SPARQL vaicājums, kas atbilst attēla 2.20 abstraktajai struktūrai (Ovcinnikova et al., 2023)

2.2.6 Teksta turpināšana

Tā kā ViziQuer/web rīks savā notācijā izmanto teksta izteiksmes (piemēram, atribūtu vai nosacījumu laukos), un šīs izteiksmes parasti balstās uz vārdiem, kas jau ir definēti izveidotajā vaicājumā (piemēram, vai nu kā īpašību vārdi datu modelī, vai kā vaicājumā ieviesti vārdi/nosaukumi), ir svarīga izteiksmju turpinājumu funkcionalitāte, kas darba ietvaros tika izstrādāta un ieviesta ViziQuer rīkā.

Teksta turpinājumu piedāvāšanai tika jaunā vidē realizēta uz LPeg.re bāzes izveidotā gramatiku balstītā metode, kas aprakstīta publikācijā (Ovcinnikova et al., 2014) un iepriekš tika aprakstīta sadaļā 2.1.4. Piedāvāto risinājumu varēja pilnībā pārnest uz JavaScript valodu.

Teksta turpinājumu priekšā-teikšanai par pamatu tika ņemta ViziQuer/web izteiksmju gramatika, kas ir realizēta kā JavaScript PEG (WEB, q) gramatika.

Gramatikā tika iebūvēti izvedumi, kas ļauj piedāvāt kā turpinājumus attiecīgās klases īpašības un ceļa izteiksmes no projektā ielādētās shēmas, turklāt priekšāteicējs saņem informāciju no simbolu tabulas, kas satur vaicājumā izveidoto lauku nosaukumus, kas ļauj papildināt turpinājumu sarakstu ar konkrētā vaicājumā izmantotajiem mainīgajiem.

2.2.7 Lietojamības novērtēšana

Lai novērtētu rīka un tā notācijas lietojamību, rīka izstrādes laikā tika veikti vairāki lietojamības pētījumi.

Pirmie notācijas lietojamības pētījumi tika veikti ar Latvijas Universitātes Medicīnas fakultātes maģistra programmas studentiem un to pasniedzēju, kas tika aprakstīti (Cerans et al., 2017). Kopā pētījumā piedalījās 7 dalībnieki bez iepriekšējās IT apmācības. Dalībniekiem tika iedota vienkārša ontoloģija un datu instanču diagramma, tika izskaidrota rīka notācija. Dalībniekiem lūdza interpretēt 10 vienkāršus vizuālos vaicājumus. Rezultāti parādīja, ka 6 no 7 dalībniekiem bija spējīgi interpretēt vismaz 70% no vaicājumiem, kas ietvēra parastos vaicājumus, agregācijas vaicājumus un vaicājumus ar apakšvaicājumiem.

Tālāki lietojamības pētījumi bija saistīti ar ViziQuer/web rīka vizuālo vaicājumu veidošanu, salīdzinot to ar tekstuālo SPARQL vaicājumu rakstīšanu. Pētījums tika aprakstīts (Cerans et al., 2018a). Pētījumā piedalījās Latvijas Universitātes Datorikas fakultātes maģistra programmas studenti. Dalībniekiem tika izstāstīti SPARQL valodas pamati, un viņi tika iepazīstināti ar ViziQuer/web notāciju un rīku. 30 dalībnieki tika sadalīti divās grupās, viena grupa rakstīja tekstuālos SPARQL vaicājumus, otra veidoja šos vaicājumus ar ViziQuer/web palīdzību. Kopumā tika iedoti 10 uzdevumi. Pētījuma rezultāti ir šādi: vidēji 2,92 pilnībā veiksmīgi vaicājumi vienam SPARQL vaicājumu rakstīšanas dalībniekam un 5,27 vaicājumi vizuālā pieraksta dalībniekam. Pētījums parādīja, ka vizuālo vaicājumu veidošana ir vieglāka par tekstuālo SPARQL rakstīšanu, tomēr ir nepieciešams uzlabot rīka notāciju.

Lai pārbaudītu vizuālo vaicājumu notācijas uzlabojumus, pēc gada atkārtojām iepriekšējā lietojamības pētījuma vizuālo daļu, uzaicinot 28 līdzīgi sagatavotus dalībniekus (Cerans et al., 2019c). Vidēji pareizi izpildīto vaicājumu rezultāts palielinājās līdz 7,5 ar līdzīgiem nosacījumiem kā iepriekšējos pētījumos. Rezultāts parādīja pareizo rezultātu pieaugumu, salīdzinot ar iepriekšējo gadu.

ViziQuer rīks piedāvā veidot vaicājumus arī pār liela izmēra heterogēnām datu kopām, kā DBPedia SPARQL piekļuves punkts. Lai izpētītu vizuālās vaicājumu vides piemērotību vaicājumu veidošanai pār DBPedia, tika veikts lietojamības pētījums (Cerans et al., 2022a). Pētījums veikts ar tehniski izglītotiem galalietotājiem (IT maģistrantūras studentiem) ar ierobežotu iepriekšējo ViziQuer pieredzi. Pētījums parāda, ka vidi var izmantot attiecīgā galalietotāju grupa, ja ir skaidrs informācijas kodējums datu piekļuves punktā. Pētījuma rezultāti liecina, ka DBPedia datu kopas shēmas lielums nekavē vaicājuma izveidi aplūkotajai lietotāju grupai, jo vaicājumi parasti ir veiksmīgi izveidoti.

Lietojamības pētījumu rezultāti ir apstiprinājuši iespēju izmantot ViziQuer rīku vaicājumu veidošanā, izmantojot RDF datus. Pētījumos ir identificētas atsevišķas problēmas, kas atklājās vaicājuma veidošanas laikā, un uz kurām ir reaģēts tālākajā rīka izstrādē (piemēram, apzīmējums {exists}, kas aprakstīts (Ovcinnikova et.al. 2023)). ViziQuer/web rīka jaunākā versija ļauj to piedāvāt vizuālu vaicājumu veidošanai, pār praktiska lieluma un nozīmes datu avotiem.

3 Secinājumi

Promocijas darba pētījuma mērķis bija parādīt, ka paplašināta UML klašu diagrammu notācija un universālas rīku būves platformas ir piemērotas, lai veidotu rīkus darbam ar bagātīgiem semantisko datu modeļiem (ontoloģijām) un vaicājumiem pār strukturētiem semantiskajiem datiem.

Darba gaitā tika atrisināti visi izvirzītie uzdevumi.

Darba ietvaros ir izveidotas praktiski izmantojamas atbalsta struktūras dažādiem ontoloģiju veidošanas uzdevumiem:

- Ir izveidots lietotāja definēto lauku un skatījumu mehānisms, kas ir izmantojams lietotāja definētas notācijas uzdošanai un attēlošanai, ļaujot pielikt jaunus laukus, elementus, vizuālos izskatus (t.sk. atkarīgus no citām konstrukcijām) un semantiku jauno konstrukciju saglabāšanai ontoloģijā. Profilus ar šī mehānisma ietvaros veidotiem lietotāja definētiem laukiem var labi izmantot arī kā pamatu citiem paplašinājumiem. Jaunās notācijas ir viegli pieliekamas, nemainot rīka pamata funkcionalitāti.
- Ir izstrādāts ontoloģiju vizualizācijas parametru ietvars – ontoloģiju ielasīšanai lietotājam vēlamā veidā. Ietvars ļauj izvēlēties, kādas aksiomās vizualizēt ontoloģijā un kādas ne. Līdzās tam var izvēlēties, no plaša OWLGrEd redaktora aksiomu attēlošanas klāsta piemērotāko veidu, kā attēlot izvelētās aksiomas diagrammā, kas ļauj katram konkrētam lietojumam un ontoloģijai izveidot labāko veidu kā to attēlot.
- Ir izstrādāti ontoloģiju eksporta modeļa risinājumi – valoda ontoloģijas aksiomu šablonu definēšanai vizuāli attēlotu ontoloģiju saglabāšanai tekstuālā OWL standarta funkcionālās sintakses formātā. Izveidotājs risinājums ir viegli pielāgojams citām platformām un ļauj viegli modificēt nosacījumus aksiomu ģenerēšanai no ontoloģijas vizuālās formas.
- Ir izstrādāta uz gramatikām balstīta priekšāteikšanas metode, kas ļauj piedāvāt turpinājumus, pamatojoties uz diagrammā jau izmantotajiem datiem un informāciju no repozitorija. OWLGrEd rīks atbalsta Mančestras sintakses izteiksmes, kas ļauj tekstuāli pierakstīt sarežģītas OWL aksiomas, un aksiomas, kas intuitīvi nevar attēlot vizuāli. Uz gramatikām balstīta priekšāteikšanas metode ļauj strādāt ar Mančestras sintakses izteiksmēm un veidot aksiomas arī lietotājiem, kas nav pazīstams ar Mančestras sintakses notāciju. Priekšāteikšanas metode ir viegli modificējama un pārnesama uz citām vidēm, tajā skaitā uz JavaScript vidi ViziQuer rīka realizācijas ietvaros.
- Ir izveidoti konkrēti uz lietojumiem orientēti papildinājumi, t.sk. paplašinājums Uzņēmumu reģistra datu modelēšanai un datu struktūras aprakstam, kā arī paplašinājumi relāciju datu bāzu un ontoloģiju atbilstību definēšanai un uz ontoloģijām balstītu datu pārvaldības modeļu aprakstam. OWLGrEd rīka paplašinājumi kopā ar lietotāja definēto lauku un skatījumu mehānismu, ļauj redaktoram pievienot konkrētām lietojumam specifiskus laukus, elementus, transformācijas un kodu, nemainot rīka pamata funkcionalitāti.

Vizuālo vaicājumu atbalsta jomā ir piedāvāti virkne risinājumu ViziQuer valodas realizācijas pamata funkcionalitātes un atbalsta servisu nodrošināšanai.

- Ir izstrādāts risinājums bagātīgu datu vaicājumu vizuālai formulēšanai pār RDF datubāzēm un to tulkošanai uz izpildāmu tekstuālu SPARQL valodu, ietverot tādas konstrukcijas kā: negācijas, agregācijas, apakš vaicājumi, apvienojumi, tekstuālas izteiksmes, ko kopumā neatbalsta neviens no apskatītajiem vizuālajiem vaicājumu rīkiem, un kas ļauj veidot sarežģītus SPARQL vaicājumus.
- ViziQuer rīkam ir pievienots uz gramatikām balstīts risinājums tekstuālu izteiksmju lietošanai vizuālo vaicājumu rīkā. SPARQL valodā piedāvā plašu klāstu ar konstrukcijām un izteiksmēm, kam grafisks attēlošanas veids nav pats piemērotākais, vai kam ir nepieciešams kompaktāks attēlošanas veids. ViziQuer rīks atbalsta plašu SPARQL izteiksmju lietošanu un

tulkošanu. Izvelētais uz gramatikām balstīts risinājums ļāva viegli modificēt un papildināt SPARQL izteiksmju gramatiku ar ViziQuer specifiskiem pierakstiem un notācijām.

- Ontoloģiju veidošanas kontekstā izstrādātā uz gramatikām balstītā priekšāteikšanas metode tika iestrādāta vizuālajā vaicājumu rīkā, citas programmēšanas valodas kontekstā, tādējādi parādot, ka to var pielāgot dažādām valodām, kas atbalsta izteiksmju parsēšanas funkcionalitāti. ViziQuer rīkā uz gramatikām balstītā priekšāteikšanas metode ļāva atvieglot lietotājiem darbu ar plašu un sarežģītu SPARQL valodas izteiksmju notāciju, piedāvājot turpinājumus no valodas konstrukcijām, datu shēmas un paša vizuālā vaicājuma.
- Veikti lietojamības pētījumi, kas parādīja piedāvātas notācijas izmantojamību un ļāva konstatēt ViziQuer notācijas priekšrocības un novērst tās trūkumus un nepilnības.

Darba ietvaros piedāvātie risinājumi realizē komplicētu lietojumu loģiku, lai atbalstītu darbu ar ontoloģiju vizualizāciju UML klašu diagrammu formā un UML veida vizuālajiem vaicājumiem. Šie risinājumi nodrošina vienotu bagātināta OWLGrEd ontoloģiju redaktora funkcionalitāti, kā arī piedāvā iespēju veidot UML veida vizuālus vaicājumus pār RDF datubāzēm (ViziQuer rīka realizāciju). Gan OWLGrEd rīka paplašinājumu, gan arī ViziQuer rīka realizācija ir veidotas universālās rīku būves platformās, līdz ar to pamatojot promocijas darba mērķa sasniegšanu, ka paplašināta UML klašu diagrammu notācija un universālas rīku būves platformas ir piemērotas, lai veidotu rīkus darbam ar bagātīgiem semantisko datu modeļiem (ontoloģijām) un vaicājumiem pār strukturētiem semantiskajiem datiem.

Pateicības

Izsaku pateicību visiem, kas palīdzēja darba tapšanā:

- Zinātniskajam vadītājam profesoram Kārlim Čerānam.
- OWLGrEd komandas locekļiem: profesoram Kārlim Čerānam, profesoram Jānim Visvaldim Bārzdiņam, Renāram Liepiņam, Artūram Sproģim, Sergejam Kozlovičam, Leldei Lācei.
- ViziQuer komandas locekļiem: profesoram Kārlim Čerānam, Leldei Lācei, Mikum Grasmanim, Agrim Šostakam, Artūram Sproģim, Uldim Bojāram.
- Studiju metodiķēm Ellai Aršai un Anitai Ermušai (kura vairs nav mūsu vidū).
- LU MII Sistēmu modelēšanas un programmatūras tehnoloģiju laboratorijas vadītājam Edgaram Celmam.
- LU MII Sistēmu modelēšanas un programmatūras tehnoloģiju laboratorijas kolēģiem.

Šo darbu daļēji atbalstījis Eiropas Sociālais fonds projekta "LU doktorantūras kapacitātes stiprināšana jaunā doktorantūras modeļa ietvarā" (Nr. 8.2.2.0/20/I/006) ietvaros.

Literatūra

Publikāciju saraksts

Cerans, K., Barzdins, G., Būmans, G., Ovcinnikova, J., Rikacovs, S., Romane, A. (2014a) On Semantic Re-Engineering of Relational Databases. H. Haav et al. (Eds.) Databases and Information Systems. Proceeding of the 11th International Baltic Conference on DB and IS, 8-11 June 2014, Tallinn, ESTONIA.

Cerans, K., Barzdins, G., Būmans, G., Ovcinnikova, J., Rikacovs, S., Romane, A. (2014b). A Relational Database Semantic Re-Engineering Technology and Tools. Baltic J. Modern Computing, Vol. 2 (2014), No. 3,183-198.

Cerans, K., Barzdins, G., Liepins, R., Ovcinnikova, J., Rikacovs, S., Sprogis, A. (2012). Graphical Schema Editing for Stardog OWL/RDF Databases using OWLGrEd/S // Proc. of OWLED 2012, Heraklion, Greece, May 27-28, 2012. CEUR-WS Vol. 849, p. 8 (http://ceur-ws.org/Vol-849/paper_19.pdf), DBLP Bibliography Server.

Cerans, K., Barzdins, J., Sostaks, A., Ovcinnikova, J., Lace, L., Grasmanis, M., Sprogis, A. (2017). Extended UML Class Diagram Constructs for Visual SPARQL Queries in ViziQuer/web In Voila 2017, CEUR Workshop Proceedings, Vol. 1947, pp. 87-98.

Cerans, K., Lace, L., Grasmanis, M., Ovčiņņikova, J. (2021a) A UML-Style Visual Query Environment Over DBPedia. Metadata and Semantic Research: 15th International Conference, MTSR 2021, Virtual Event, 29 November – 3 December 2021: Proceedings. Communications in Computer and Information Science ; Vol. 1537 CCIS. Springer: Cham, 2022, pp. 16-27. DOI: 10.1007/978-3-030-98876-0_2.

Cerans, K., Lace, L., Romane, A., Ovcinnikova, J., Grasmanis, M., Sprogis, A., Sostaks, A. (2019a). Schema-Based Visual Queries over Linked Data Endpoints. In: Garoufallou E., Fallucchi F., William De Luca E. (eds) Metadata and Semantic Research. MTSR 2019. Communications in Computer and Information Science, Vol. 1057, pp. 200-206. Springer, Cham. 28-31 October, Rome, Italy, 2019.

Cerans, K., Lace, L., Romane, A., Ovcinnikova, J., Kozlovics, S., Grasmanis, M., Hodakovska, J., Sprogis, A., Sostaks, A. (2019b). Visual Queries over Scholarly Data and other Linked Data Endpoints. In ISWC Satellites 2019. CEUR-WS volume 2456, pp. 297-300. 26-30 October, The University of Auckland, New Zealand, 2019.

Cerans, K., Liepins, R., Sprogis, A., Ovcinnikova, J., Barzdins, G. (2015a) Domain-Specific OWL Ontology Visualization with OWLGrEd, Lecture Notes in Computer Science ((subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)). 2015. Vol. 7540, pp. 419-424 (2015): The Semantic Web: ESWC 2012 Satellite Events (Heraklion, Greece; 27-31 May 2012)

Cerans, K., Ovcinnikova, J. (2016). ViziQuer: Notation and Tool for Data Analysis SPARQL Queries. In Voila 2016, CEUR Workshop Proceedings, Vol. 1704, pp. 151-159.

Cerans, K., Ovcinnikova, J., Bojars, U., Grasmanis, M., Lace, L., Romane, A. (2021b). Schema-Backed Visual Queries over Europeana and other Linked Data Resources. The Semantic Web: ESWC 2021 Satellite Events. ESWC 2021. Lecture Notes in Computer Science, Vol. 12739. pp 82-87. Springer, Cham. https://doi.org/10.1007/978-3-030-80418-3_15.

Cerans, K., Ovcinnikova, J., Grasmanis, M., Lace, L. (2022a) Experience with Visual SPARQL Queries over DBPedia. In Voila 2022, CEUR Workshop Proceedings, Vol. 3253, pp. 59-65. Visualization and Interaction for Ontologies and Linked Data (Voila 2022) 23 October 2022, Virtual Workshop.

Cerans, K., Ovcinnikova, J., Grasmanis, M., Lace, L. (2022b) Towards UML-Style Visual Queries over Wikidata (Posters and Demos) // The Semantic Web: ESWC 2022 Satellite Events, Hersonissos, Crete, Greece, 29 May – 2 June 2022: Proceedings / P. Groth, et al. (Eds.). Lecture Notes in Computer Science

(including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) ; Vol. 13384. Cham: Springer, 2022, pp. 11-15. https://doi.org/10.1007/978-3-031-11609-4_2.

Cerans, K., Ovcinnikova, J., Lace, L., Hodakovska, J., Romane, A., Grasmanis, M., Kalnina, E., Sprogis A., Sostaks, A. (2019c). Visual Query Environment over RDF Data. In Posters and Demos at SEMANTiCS 2019. CEUR-WS volume 2451. 9-12 September, Karlsruhe, Germany, 2019.

Cerans, K., Ovcinnikova, J., Liepins, R., Grasmanis, M. (2019d). Extensible Visualizations of Ontologies in OWLGrEd. In The Semantic Web: ESWC 2019 Satellite Events, ESWC 2019. Lecture Notes in Computer Science, Vol. 11762 pp. 191-196. Springer, Cham. 2-6 June, Portorož, Slovenia, 2019.

Cerans, K., Ovcinnikova, J., Liepins, R., Sprogis, A. (2013). Advanced OWL 2.0 Ontology Visualization in OWLGrEd // Caplinskas, A., Dzemyda, G., Lupeikiene, A., Vasilecas, O. (eds.) Databases and Information Systems VII, IOS Press, Frontiers in Artificial Intelligence and Applications, Vol. 249, pp. 41-54, 2013.

Cerans, K., Ovcinnikova, J., Zviedris, M. (2015b). SPARQL Aggregate Queries made easy with Diagrammatic Query Language ViziQuer. In ISWC 2015 Posters & Demonstrations Track, CEUR Workshop Proceedings, Vol. 1486.

Cerans, K., Ovcinnikova, J., Zviedris, M. (2015c). Towards Graphical Query Notation for Semantic Databases. In: Matulevičius, R., Dumas, M. (eds) Perspectives in Business Informatics Research. BIR 2015. Lecture Notes in Business Information Processing, Vol. 229. Springer, Cham. https://doi.org/10.1007/978-3-319-21915-8_19 pp. 273-281.

Cerans, K., Sostaks, A., Bojars, U., Barzdins, J., Ovcinnikova, J., Lace, L., Grasmanis, M., Sprogis, A. (2018a). ViziQuer: A Visual Notation for RDF Data Analysis Queries. Garoufallou, E., Sartori, F., Siatri, R., Zervas, M. (eds) Metadata and Semantic Research. MTSR 2018. Communications in Computer and Information Science, vol. 846. Springer, Cham, pp. 50-62. https://doi.org/10.1007/978-3-030-14401-2_5.

Cerans, K., Sostaks, A., Bojars, U., Ovcinnikova, J., Lace, L., Grasmanis, M., Romane, A., Sprogis, A., Barzdins, J. (2018b). ViziQuer: a Web-based Tool for Visual Diagrammatic Queries over RDF Data In The Semantic Web: ESWC 2018 Satellite Events, Springer Verlag Lecture Notes in Computer Science, Vol. 11155, pp. 158-163. Springer, Cham. https://doi.org/10.1007/978-3-319-98192-5_30.

Liepins, R., Gruzitis, N., Cerans, K., Ovcinnikova, J., Bojars, U., Celms, E. (2016). Adding Verbalization to Graphical Ontology Editor OWLGrEd. In: G. Arnicans et al. (Eds.). Frontiers in Artificial Intelligence and Applications, Vol. 291, Databases and Information Systems IX, IOS Press, pp. 17-30, 2016.

Ovcinnikova, J. (2020). Ontology Export Patterns in OWLGrEd Editor. Baltic J. Modern Computing, Vol. 8 (2020), No. 3, 444-460, https://doi.org/10.22364/bjmc.2020.8.3.04.

Ovcinnikova, J., Cerans, K. (2016). Advanced UML Style Visualization of OWL Ontologies. Proc. of VOILA 2016. CEUR, vol. 1704, CEUR-WS.org, 2016, pp. 136-142.

Ovcinnikova, J., Cerans, K. Liepins, R. (2014) Grammar based content completion method using Lua LPeg.re module.//Proc. Of 2014 IEEE 2nd Workshop on Advances in Information, Electronic and Electrical Engineering (AIEEE), Vilnius, Lithuania, 28-29 November, IEEE Xplore Digital Library, 2014.

Ovcinnikova, J., Sostaks, A., Cerans, K. (2023) Visual Diagrammatic Queries in ViziQuer: Overview and Implementation. In arXiv arXiv:2304.14825 [cs.DB]. Baltic J. Modern Computing, Vol. 11 (2023), No. 2, 317-350 https://doi.org/10.22364/bjmc.2023.11.2.07

Izmantota literatūra

Aasman, J. (2017). Graph visualization with a time machine, In Dataconomy 29 August 2017, http://dataconomy.com/2017/08/graph-visualization-time-machine/.

Almendros-Jiménez, J., Iribarne, L., Asensio, J.A., Padilla, N., & Vicente-Chicote, C. (2009). An Eclipse GMF Tool for Modelling User Interaction. WSKS 2009, Chania, Crete, Greece, 16-18 September 2009.

Angelov, K., Ranta, A. (2009). Implementing Controlled Languages in GF. Fuchs, N.E. (ed.) CNL 2009. LNCS, Vol. 5972, pp. 82-101. Springer, Heidelberg (2010).

Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z. (2007). DBpedia: A Nucleus for a Web of Open Data, in Aberer. K., et al. (ed.), Proceedings of The Semantic Web, ISWC 2007. Lecture Notes in Computer Science, Vol. 4825, Springer, Berlin, Heidelberg, pp. 722-735. https://doi.org/10.1007/978-3-540-76298-0_52.

Barzdins, G., Liepins, E., Veilande, M., Zviedris, M. (2009) Ontology Enabled Graphical Database Query Tool for End-Users. Selected papers from DBIS'2008, Hele-Mai Haav (Eds.), Frontiers in Artificial Intelligence and Applications series, IOS Press, 2009. 187:105-116.

Barzdins, J., Barzdins, G., Cerans, K., Liepins, R., Sprogis, A. (2010a). OWLGrEd: a UML Style Graphical Notation and Editor for OWL 2. In: Proc. of OWLED 2010.

Barzdins, J., Barzdins, G., Cerans, K., Liepins, R., Sprogis, A. (2010b). UML Style Graphical Notation and Editor for OWL 2. In Perspectives in Business Informatics Research, Lecture Notes in Business Information Processing, Volume 64, Part 2, pages 102-114, 2010

Barzdins, J., Cerans, K., Kozlovics, S., Rencis, E., Zarins A. (2010c). A Graph Diagram Engine for the Transformation-Driven Architecture. Scientific Papers, University of Latvia, 2010. Vol. 756 Computer Science and Information Technologies 139–149 P.

Barzdins, J., Zarins, A., Cerans, K., Kalnins, A., Rencis, E., Lace, L., Liepins R., Sprogis, A. (2007). GrTP: Transformation Based Graphical Tool Building framework, Proc. of MoDELS 2007 Workshop on Model Driven Development of Advanced User Interfaces, MDDAUI 2007; Nashville, TN; USA.

Bechhofer S., Harmelen, F., Hendler, J., Horrocks I., McGuinness D.L., Patel-Schneider P.F., Stein L.A. (2004). OWL Web Ontology Language Reference. W3C Recommendation 10 February 2004. Available at https://www.w3.org/TR/owl-ref/

Berners-Lee, T., Hendler, J., Lassila, O. (2001). The Semantic Web, Scientific American, May 2001, pp. 29-37.

Bosca, A., Bonino, D., Pellegrino, P. (2005). OntoSphere: more than a 3D ontology visualization tool. In SWAP, the 2nd Italian Semantic Web Workshop.

Brickley, D., Guha, R.V,. (2014). RDF Schema 1.1. Pieejams: https://www.w3.org/TR/rdf11-schema/.

Bumans, G., Cerans, K. (2011). Advanced RDB-to-RDF/OWL mapping facilities in RDB2OW BIR, volume 90 of Lecture Notes in Business Information Processing, pp. 142-157. Springer, (2011).

Cerans, K., Bumans, G. (2011). RDB2OWL: a RDB-to-RDF/OWL Mapping Specification Language, Proceedings of the 2011 conference on Databases and Information Systems VI, DB&IS 2010.

Cook, S., Jones, G., Kent, S., Wills, A. C. (2007) Domain-Specific Development with Visual Studio DSL Tools. Addison-Wesley, 2007.

Dudas, M., Lohmann, S., Svatek, V., Pavlov, D. (2018). Ontology visualization methods and tools: a survey of the state of the art. The Knowledge Engineering Review, 33.

Ferré, S. (2017). Sparklis: An Expressive Query Builder for SPARQL Endpoints with Guidance in Natural Language. Semantic Web 8(3), 405-418. IOS Press.

Haag, F., Lohmann, S., Siek, S., Ertl, T. (2015). QueryVOWL: Visual Composition of SPARQL Queries. In: The Semantic Web: ESWC 2015 Satellite Events. LNCS, Vol. 9341, pp. 62-66. Springer, (2015), http://vowl.visualdataWEB ().org/queryvowl/.

Hayes, P. J., Patel-Schneider, P. F. (2014). RDF 1.1 Semantics. Pieejams: https://www.w3.org/TR/rdf11-mt/.

Hitzler, P., Krötzsch, M., Parsia,B., Patel-Schneider, P.F., Rudolph, S. (2012). OWL 2 Web Ontology Language Primer (Second Edition). Pieejams: https://www.w3.org/TR/owl2-primer/.

Horridge, M., Bechhofer, S. (2011). The OWL API: A Java API for OWL ontologies. Semantic Web, 2, 11-21.

Horridge, M., Patel-Schneide, P. F. (2012). W3C Working Group Note, 11 December 2012. Pieejams: https://www.w3.org/TR/owl2-manchester-syntax/.

Kaarel K., Norbert E. F. (2007). Verbalizing OWL in Attempto Controlled English. Proceedings of OWL: Experiences and Directions (OWLED 2007).

Kats, L.C., Visser, E. (2010). The spoofax language workbench: rules for declarative specification of languages and IDEs. // Rinard, M. (eds.), Proceedings of the 25th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications, ACM.

Khalili, A., Merono-Penuela, A. (2017). WYSIWYQ–What You See Is What You Query, Proceedings of Voila 2017, CEUR Workshop, 1947, 123-130.

Kharlamov, E., Hovland, D., Skjaeveland, M. G., Bilidas, D., Jimenez-Ruiz, E., Xiao, G., Soylu, A., Lanti, D., Rezk, M., Zheleznyakov, D., et al (2017). Ontology Based Data Access in Statoil. Journal of Web Semantics, 44, pp. 3-36. doi: 10.1016/j.websem.2017.05.005

Knublauch, H., Kontokostas, D. (2017). Shapes Constraint Language (SHACL). Pieejams: https://www.w3.org/TR/shacl/.

Kozlovics, S. (2010). A Dialog Engine Metamodel for the Transformation-Driven Architecture. Scientific Papers, University of Latvia, 2010. Vol. 756. Computer Science and Information Technologies 151–170 P.

Kuhn, T. (2013). The Understandability of OWL Statements in Controlled English. Semantic Web journal, Volume 4, pp. 101-115. IOS Press, 2013.

Liepins, R. (2012). Library for model querying: IQuery. Proc. of 12th Workshop on OCL and Textual Modeling, OCL 2012 – Being Part of the ACM/IEEE 15th International Conference on Model Driven Engineering Languages and Systems, MODELS 2012; Innsbruck; Austria.

Lohmann, S., Negru, S., Bold, D. (2014). The ProtégéVOWL Plugin: Ontology Visualization for Everyone. In: Presutti, V., Blomqvist, E., Troncy, R., Sack, H., Papadakis, I., Tordai, A. (eds) The Semantic Web: ESWC 2014 Satellite Events. ESWC 2014. Lecture Notes in Computer Science(), vol 8798. Springer, Cham.

Lohmann, S., Negru, S., Haag F., Ertl, T. (2016) Visualizing Ontologies with VOWL. Semantic Web 7(4), 399-419.

McGuinness, D. L., Harmelen, F. (2004). OWL Web Ontology Language. https://www.w3.org/TR/owl-features/.

Motik, B., Patel-Schneider, P. F., Parsia, B. (2012). OWL 2 Web Ontology Language Structural Specification and Functional-Style Syntax. Pieejams: https://www.w3.org/TR/owl2-syntax/.

Mouromtsev, D., Pavlov, D., Emelyanov, Y., Morozov, A., Razdyakonov, D. & Galkin, M. (2015). The simple, web-based tool for visualization and sharing of semantic data and ontologies. In ISWC 2015 Posters & Demonstrations Track. CEUR (2015).

Oldevik, J. (2006). MOFScript User Guide. 9 February 2006, https://umt-qvt.sourceforge.net/mofscript/docs/MOFScript-User-Guide.pdf.

Prud'hommeaux, E., Labra Gayo, J.E., & Solbrig, H.R. (2014). Shape expressions: an RDF validation and transformation language. Joint Conference on Lexical and Computational Semantics.

Rose, L.M., Paige, R.F., Kolovos, D.S., Polack, F.A.C. (2008). The Epsilon Generation Language. In: Schieferdecker, I., Hartman, A. (eds) Model Driven Architecture – Foundations and Applications. ECMDA-FA 2008. Lecture Notes in Computer Science, Vol. 5095. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-69100-6_1.

Schweiger, D., Trajanoski, Z., Pabinger, S. (2014). SPARQLGraph: a web-based platform for graphically querying biological Semantic Web databases, In BMC Bioinformatics 2014 15:279.

Scott W. A. (2009). UML 2 Class Diagrams. Webdoc 2003-2009 Pieejams: http://www.agilemodeling.com/artifacts/classDiagram.html.

Smith, K.M., Welty, C., McGuinness D.L. (2004). OWL Web Ontology Language Guide. W3C Recommendation 10 February 2004. Available at https://www.w3.org/TR/owl-guide/

Soylu, A., Giese, M., Kharlamov, E., Jimenez-Ruiz, E., Zheleznyakov, D., Horrocks, I. (2017). Ontology-based End-user Visual Query Formulation: Why, What, Who, How, and Which?. Universal Access in the Information Society. 16, 435–467. https://doi.org/10.1007/s10209-016-0465-0.

Soylu, A., Giese, M., Jimenez-Ruiz, E. et al. (2016). Experiencing OptiqueVQS: a multi-paradigm and ontology-based visual query system for end users. Univ Access Inf Soc 15, 129–152 (2016). https://doi.org/10.1007/s10209-015-0404-5

Soylu, A., Kharlamov, E., Zheleznyakov, D., Jimenez-Ruiz, E., Giese, M., Skjaeveland, M. G., Hovland, D., Schlatte, R., Brandt, S., Lie, H. and Horrocks, I. (2018). OptiqueVQS: A visual query system over ontologies for industry. Semantic Web, 9(5), doi: 10.3233/SW-180293.

Sprogis, A. (2010). The Configurator in DSL Tool Building, Scientific Papers, University of Latvia, Vol. 756. Computer Science and Information Technologies. Riga, Latvia, 2010, pp. 173–192.

Sprogis, A. (2016). ajoo: WEB Based Framework for Domain Specific Modeling Tools. // In: Frontiers of AI and Applications, Vol. 291, Databases and Information Systems IX, IOS Press, pp. 115-126, 2016, http://ebooks.iospress.com/volumearticle/45704.

Vargas H., Buil-Aranda C., Hogan A., López C. (2019). RDF Explorer: A Visual SPARQL Query Builder. In: Ghidini C. et al. (eds) The Semantic Web – ISWC 2019. ISWC 2019. Lecture Notes in Computer Science, Vol. 11778. Springer, Cham. https://doi.org/10.1007/978-3-030-30793-6_37.

Vega-Gorgojo, G., Giese, M., Heggestøyl, S., Soylu, A., Waaler, A. (2016). PepeSearch: semantic data for the masses. PloS one, 11(3), e0151573. https://doi.org/10.1371/journal.pone.0151573 0151573

Viyovic, V., Maksimovic, M., Perisic, B. (2014). Sirius: A rapid development of DSM graphical editor. IEEE 18th International Conference on Intelligent Engineering Systems INES 2014, 233-238.

Voelter, M. (2006). oAW xText: A framework for textual DSLs. Modeling Symposium at Eclipse Summit 2006.

Vrandečić, D., Krötzsch, M. (2014). Wikidata: a Free Collaborative Knowledge Base. Communications of the ACM 57, 10, 78-85. https://doi.org/10.1145/2629489.

Wiens, V., Lohmann, S., Auer, S. (2018). WebVOWL Editor: Device-Independent Visual Ontology Modeling. Proceedings of the ISWC 2018 Posters & Demonstrations, Industry and Blue Sky Ideas Tracks. CEUR Workshop Proceedings, vol. 2180, CEUR-WS.org, 2018.

Wyner, A., Angelov, K., Barzdins, G., Damljanovic, D., Davis, B., Fuchs, N., Hoefler, S., Jones, K., Kaljurand, K., Kuhn, T., Luts, M., Pool, J., Rosner, M., Schwitter, R., Sowa, J. (2010). On Controlled Natural Languages: Properties and Prospects Workshop on Controlled Natural Languages, CNL 2009, LNCS/LNAI 5972, Springer, 2010.

Xiao, G., Lanti, D., Kontchakov, R., Komla-Ebri, s., Güzel-Kalayci, E., Ding, L., Corman, J., Cogrel, B., Calvanese, D., Botoeva, E. (2020). The Virtual Knowledge Graph System Ontop. In: Pan J.Z. et al. (eds) The Semantic Web – ISWC 2020. ISWC 2020. Lecture Notes in Computer Science, Vol. 12507. Springer, Cham. https://doi.org/10.1007/978-3-030-62466-8_17.

Zviedris, M. (2014). Dati kā ontoloģija - glabāšana, vaicāšana, vizualizācija. Promocijas darbs

Zviedris, M., Romane, A., Barzdins, G., Cerans, K. (2013). Ontology-Based Information System JIST 2013 Seoul, Korea, 2013.

WEB (a). Acceleo Query Language (AQL), https://www.eclipse.org/acceleo/documentation/

WEB (b). AllegroGraph – New FedShard Feature, https://allegrograph.com/products/allegrograph/

WEB (c). Europeana SPARQL endpoint: live and under a new address, https://pro.europeana.eu/post/europeana-sparql-endpoint

WEB (d). JavaScript, https://www.javascript.com/

WEB (e). JET, https://www.eclipse.org/articles/Article-JET/jet_tutorial1.html

WEB (f). Jquery, https://jquery.com/

WEB (g). LinDA Query Designer, https://2017.semantics.cc/sites/2017.semantics.cc/files/files/LinDAQueryDesigner.pdf

WEB(h). Linked Open Data, https://www.w3.org/egov/wiki/Linked_Open_Data

WEB(i) http://www.inf.puc-rio.br/~roberto/lpeg/re.html

WEB (j). ODM UML profile for OWL, https://www.omg.org/spec/ODM/1.1/PDF

WEB (k). OntoGraf Protege plugin, http://protegewiki.stanford.edu/wiki/OntoGraf

WEB (l). OntoGraf Tab, https://protegewiki.stanford.edu/wiki/Pr4_UG_rt_OntoGraf

WEB (m). OntoSpere3D "More than a 3D ontology visualization tool", https://ontosphere3d.sourceforge.net/index.html

WEB (n). OWLGrEd, http://owlgred.lumii.lv/

WEB (o). OWLGrEd success stories, http://owlgred.lumii.lv/success_stories

WEB (p). OWLViz, https://protegewiki.stanford.edu/wiki/OWLViz

WEB (q). Parser Generator for JavaScript, https://pegjs.org/

WEB (r). Protégé, https://protege.stanford.edu/

WEB (s). ProtégéVOWL: VOWL Plugin for Protégé, http://vowl.visualdataweb.org/protegevowl.html

WEB (t). SPARQL 1.1 Query Language. W3C Recommendation 21 March 2013, https://www.w3.org/TR/sparql11-query/

WEB (u). SPARQL 1.1 Query Language, Expression, 21 March 2013, https://www.w3.org/TR/sparql11-query/#rExpression

WEB (v). SubjectPredicateObject, https://www.w3.org/wiki/SubjectPredicateObject

WEB (w). The Programming Language Lua, https://www.lua.org/

WEB (x). TopBraid Composer, https://www.topquadrant.com/

WEB (y). Unified Modeling Language (UML), https://www.omg.org/spec/UML/2.5.1/PDF

WEB (z). ViziQuer/web, https://viziquer.lumii.lv/

WEB (aa). World Wide Web Consortium (W3C), https://www.w3.org/2001/sw/

WEB (ab). WebVOWL: Web-based Visualization of Ontologies, http://vowl.visualdataweb.org/webvowl.html

WEB (ac). Xpand, http://wiki.eclipse.org/Xpand