KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS
INFORMACIJOS SISTEMŲ KATEDRA

Kristina Skalskytė

Oracle Business Rules įrankio galimybių veiklos taisyklėms automatizuoti tyrimas

Magistro darbas

Darbo vadovas
prof. dr. L. Nemuraitė

Kaunas, 2009

KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS
INFORMACIJOS SISTEMŲ KATEDRA

Kristina Skalskytė

Oracle Business Rules įrankio galimybių veiklos taisyklėms automatizuoti tyrimas

Magistro darbas

Recenzentas
doc. dr. E. Karčiauskas
2009-01-12

Vadovas
prof. dr. L. Nemuraitė
2009-01-12

Atliko
IFM-3/4 gr. stud.
Kristina Skalskytė
2009-01-12

Kaunas, 2009

TURINYS

Investigation of Potentiality of Oracle Business Rules Tool for Automation of Business Rules

SUMMARY

Business rules approach is spreading among enterprises due to business agility and requirements of transparency and controllability. The theory of business rules have been researched widely, however, practical usage of business rules is still more intuitive than based on any research findings. The object of this study is Oracle Business Rules tool which is a part of Oracle Middleware and could be attractive for enterprise developers and managers, if they use or plan to use Oracle production. In this study the author investigated Oracle Business Rules tool environment, the principles of user interface and work organization, the behavior of rule engine and the potential to automate different kinds of business rules. The competitiveness of rule-based systems was proven by demonstrating that the efficiency of the tool equals the efficiency of solutions based on standard technologies. Furthermore, a very attractive feature of the tool is its potential to integrate it. Different application areas were investigated and discussed in separate chapters in order to better emphasize the features of the tool. The study conclusions are given at the end of the paper.

1. ĮVADAS

Vienas iš požiūrių, plintančių informacijos sistemų (IS) projektavime ir realizacijoje – veiklos taisyklėmis paremtos IS. Tai sprendimas veiklos politiką (veiklos taisykles) atskirti į atskirą architektūros lygmenį, kas palengvintų ir pagreitintų nuolat kintančių reikalavimų valdymą. Yra nemaža skirtingų požiūrių į veiklos taisykles, jų taikymo metodiką ir realizavimą.

Veiklos taisyklių apibrėžimas informacinių technologijų požiūriu teigia, kad „veiklos taisyklės – tai formuluotė, apibrėžianti ar apribojanti veiklos aspektus. Ji skirta apibrėžti veiklos struktūrą ir valdyti arba daryti įtaką jos veikimui“[1] [11]. Veiklos taisyklėmis paremtos IS projektuojamos tose dalykinėse srityse, kur dažna veiklos politikos (strategijos) kaita, reikia greito atsako pasikeitimų valdymui, be to, pageidautinas aukšto lygio pritaikymas vartotojo reikalavimams. Tai ypač būdinga šioms veiklos sritims:

- draudimas, sveikatos apsauga;
- finansai ir rizikos analizė;
- operacijų valdymas;
- skambučių centrai, ir kt.

Bet kuri veikla koncepciniame lygmenyje turi savo taisykles. Veiklos mastu aktualu ne tai, kaip atrasti ar išskirti, o kaip valdyti veiklos taisykles. Šiandien veiklos taisyklės tampa veiklos informaciniu turtu, informacinių technologijų priemonėmis užrašyta veiklos strategija. Veiklos taisykles naudojančiose sistemose propaguojamas kelių veiklos taisyklių rinkinių sudarymas, pakartotinis taisyklių panaudojimas, bendradarbiaujančių veiklos objektų mainai (paslaugų architektūra), taisyklių rinkinių valdymo perdavimas dalykinės srities specialistams.

Temos aktualumas

Verslas (veikla) ir jo reikalavimai kinta greičiau nei IS spėja prisitaikyti. Tai pagrindinė priežastis, kodėl veiklos taisyklių aktualumas auga. Kai didžioji dalis linkusios keistis logikos iškeliama į atskirą architektūrinį lygmenį, esantį tarp duomenų bazės ir vartotojo sąsajos, pakeitimų valdymas tampa greitesnis ir lengvesnis. Veiklos taisyklių žodynų ir taisyklių rinkinių (strategijų) sukoncentravimas vienoje vietoje veiklos atžvilgiu duoda aiškumą ir skaidrumą, tuo pačiu lengvina pačios veiklos logikos valdymą (loginiu, ne vien techniniu požiūriu), šablonų ar grafinės notacijos

[1] Business Rule is a statement, that defines or constraints an aspect of the business. It is intended to assert business structure, or to control or influence the behavior of the business.

naudojimas leidžia taisykles užrašyti verslo atstovams suprantama kalba, taip jų valdymą atiduodant specialistams.

Veiklos taisyklių požiūrio plitimą rodo ir tai, kad beveik visi šiuo metu didžiausi verslo programinės įrango kūrėjai (Microsoft, Oracle, IBM, SAP) savo produktuose vysto ir diegia veiklos taisyklių įrankius. Tuo tarpu pats veiklos taisyklių požiūris gimė jau 1989 m. su projektu GUIDE (suformuotu IBM vartotojų grupės). Išskiriami 2 pagrindiniai įvykiai, turėję įtakos veiklos taisyklių raidai. 1995m. pasirodęs dokumentas „Veiklos taisyklių apibrėžimas. Kas jos yra iš tikro? [2]“ (GUIDE, 1995) ir „Veiklos taisyklių knyga[3]“ (1994, R. Ross). Šiuo metu veikia ne viena organizacija, užsiimanti veiklos taisyklių populiarinimu ir moksliniais tyrimais. Tarp jų žymiausios BRC (Business Rules Community), BRG (Business Rules Group) ir kitos. Šis darbas taip pat remiasi informacija, pateikiama šių organizacijų interneto prieigose [12].

Tyrimo objektas ir tikslai

Praktinis veiklos taisyklių panaudojimas ir įrankių kūrimas taisyklėms specifikuoti, saugoti ir apdoroti ėmė vystytis ne taip jau senai. Dalį techninių realizacijos galimybių veiklos taisyklės perėmė iš ekspertinių sistemų. Techninis šių realizacijų veikimas gana panašus, tačiau skiriasi panaudojimo sritys, atvejai ir tikslai.

Darbe bus tiriamas vienos didžiausių verslo programinės įrangos gamintojų, Oracle veiklos taisyklių įrankis – Oracle Business Rules (OBR).

Tyrimo tikslas - ištirti Oracle veiklos taisyklių įrankio tinkamumą kurti informacines sistemas.

Darbo uždaviniai:

- Išanalizuoti Oracle veiklos taisyklių įrankio veikimo principus;
- Nustatyti įrankio pranašumus ir trūkumus;
- Praktiškai patikrinti veiklos taisyklių automatizavimo galimybes;
- Pateikti rekomendacijas IS kūrėjams.

Įvykdžius iškeltus uždavinius, bus pagrindžiamas įrankio tinkamumas ir įvertintos galimybės kurti veiklos taisyklėmis paremtas sistemas.

Darbo struktūra

Darbas struktūra atitinka iškeltus uždavinius. Pirmuose skyriuose bus apžvelgiami teoriniai veikos taisyklių klausimai: klasifikavimas ir šablonai. Toliau nagrinėjama Oracle veiklos taisyklių įrankio architektūra, vartotojo aplinka ir veikimo principas. Siekiant atlikti kuo išsamesnę įrankio

[2] Defining Business Rules - What Are They Really?
[3] The Business Rule Book

analizę, kiekvienas tyrimui parenkama kita dalykinė sritis. Tyrimą aprašantys skyriai pradedami trumpu dalykinės srities pristatymu ir tyrimo uždavinio formuluote, toliau pateikiama tyrimo eiga ir apibendrinami rezultatai. Tyrime naudojamos šios dalykinės sritys:

- Automobilių nuoma – standartinė veiklos taisyklių užrašymui naudojama dalykinė sritis, aprašyta dar GUIDE [1] projekte. Tyrimo tikslas – praktiškai išsiaiškinti įrankio taisyklių procesoriaus veikimo principus.
- Nurodymų įforminimas – dalykinė sritis su dideliu ribojimų skaičiumi. Tyrimo tikslas – ištirti kuo įvairesnių veiklos taisyklių automatizavimo galimybes ir metodiką.
- Telekomunikacinių paketų apdorojimas – dalykinė sritis su dideliu pradinių duomenų skaičiumi. Tyrimas skirtas įrankio našumui įvertinti. Realizavus tą patį uždavinį OBR ir Java technologijomis, atliktas laiko tyrimas duomenų apdorojimo galimybėms įvertinti.
- Valiutos kursai – ši dalykinė sritis panaudota įrankio universalumo ir integravimo galimybių demonstravimui.

Darbo pabaigoje pateikiamos išvados.

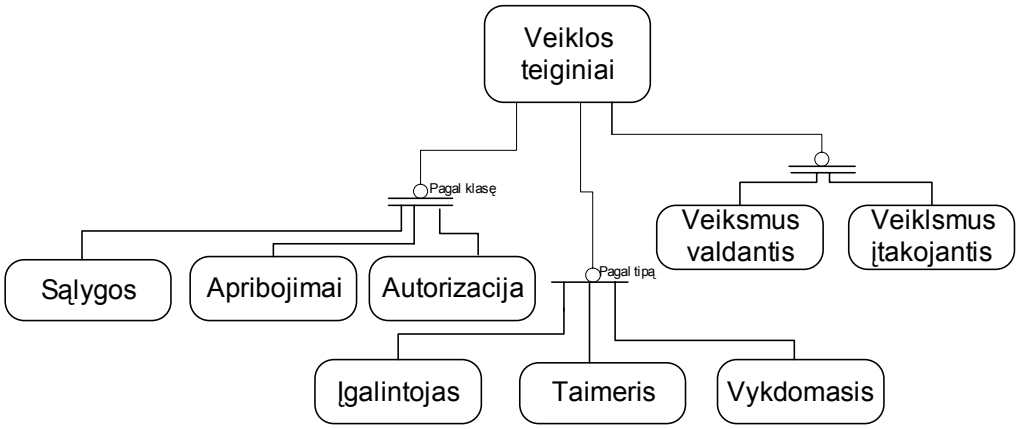
2. VEIKLOS TAISYKLIŲ TYRIMO SRITIES APŽVALGA

Šio darbo tyrimo sritis yra veiklos taisyklių automatizavimo požiūris ir jį realizuojančių įrankių taikymo metodika. Tyrimo objektas – Oracle veiklos taisyklių įrankis ir jo taikymo informacinėms sistemoms kurti procesas.

2.1. Veiklos taisyklių klasifikacija

Atliekant darbą buvo susipažinta su skirtingais požiūriais į veiklos taisyklių klasifikavimą, identifikavimą, specifikavimą, skirtingus informacijos sistemų, paremtų veiklos taisyklėmis, realizavimo metodus. Gana daug veiklos taisykles aprašančių metodų yra teoriniai. Šiame darbe pagrindu (kuriuo buvo remiamasi tiriant įrankį) pasirinktas vienas pirmųjų veiklos taisyklių teorinių modelių - Business Rule Group Guide projektas. Projekto rezultatas - sudarytas išsamus veiklos taisyklių metamodelis, pateikiamas priede (1 Priedas). Juo remiasi ir daugelis kitų, vėliau sudarytų modelių, pasižyminčių platesne ir detalesne taisyklių klasifikacija. GUIDE projektas [1] veiklos taisykles suskirstė į:

- struktūrinius teiginius – apima terminus (objektai, jų atributai) bei faktus (sąryšiai tarp objektų);
- veiklos teiginius – kurie dar skirstomi pagal 1 pav. Veiklos teiginių klasifikacija pagal GUIDE projektą [1] parodytą struktūrą ir apibrėžia dinaminę veiklos pusę;
- išvestis – kurios skirstomos į matematinius skaičiavimus ir išvadas.



1 pav. Veiklos teiginių klasifikacija pagal GUIDE projektą [1]

GUIDE projekto nariai veiklos taisykles nagrinėja tik nuo architektūros nepriklausomame IS projektavimo lygmenyje, duomenų bei motyvacijos aspektais (2 Priedas), tai atitiktų Zachman

matricos trečią eilutę, pirmą ir šeštą stulpelius. Šiame darbe aktualios IS modeliavimo, detalaus projektavimo ir realizavimo fazės.

Kitą, gana išsamų veiklos taisyklių modelį, pateikia Ronaldas G. Rosas. Jis taisykles skirsto į atomines ir išvestines. Atominėms taisyklėms yra priskyręs 7 taisyklių šeimas (apima 32 tipus), išvestinėms – 12 šeimų (apima 58 taisyklių tipus). Taigi Roso klasifikavimas labai smulkus, be to, jis pasiūlė ir taisyklių modeliavimo grafinę notaciją.

Dar vienas metodų – Barbaros von Halle taisyklų klasifikavimas į 7 tipus. Ji taip pat pasiūlė šablonus šių taisyklių specifikavimui.

1. lentelė

Barbaros von Halle veiklos taisyklių klasifikavimo modelis

Taisyklės tipas	**Šablonas**
Terminas	<term> IS DEFINED AS <text>
Faktas	<term1> IS A <term2> <term1> <verb> <term2> <term1> IS COMPOSED OF <term2> <term1> IS A ROLE PLAYED BY<term2> <term1> HAS A PROPERTY OF <term2>
Skaičiavimas	<term1> IS COMPUTED AS <formula>
Būtinumo ribojimas	<term1> MUST HAVE <at least, at most, exactly n of><term2> <term1> MUST BE <comparison> <term2>, <value>, <value list> <term1> MUST BE (MUST NOT BE) IN LIST <a, b, c> IF <rule phrase> THEN <constraint of any of the above types>
Rekomendacija	<term1> SHOULD HAVE <at least, at most, exactly n of><term2> <term1> SHOULD BE <comparison> <term2>,<value>,<value list> <term1> SHOULD BE (SHOULD NOT BE) IN LIST <a, b, c> IF <rule phrase> THEN <constraint of any of the above guideline types>
Išvestis	IF <term1> <operator> <term2, value, value list> AND <again> THEN <term 3> <operator> <term4> Operator : • =, not =, =<, =>, <> • <at least n, at most n, exactly n> of
Įgalintojas	IF <term1> <operator> <term2> THEN <action>

Be skirtingų metodikų, yra ir vieningai pripažintos veiklos taisyklių taisyklės, pavyzdžiui, Business Rules Group sudarytas veiklos taisyklių manifestas apibrėžiantis jų nepriklausomumą [2] arba dvylika C. J. Date išskirtų meta taisyklių [6].

2.2. Problemos sprendimo analizė

Panaudojant veiklos taisykles IS kūrimui stengiamasi atskirti trečią sluoksnį, kuris įsiterpia tarp duomenų vaizdavimo ir duomenų logikos ir realizuoja dalį IS dinamikos - veiklos politiką (business policy). Pagrindiniai tokios sistemos komponentai, reikalingi veiklos taisyklių realizavimui, pateikiami 2 lentelėje

2 lentelė

Pagrindiniai veiklos taisyklių realizacijos IS komponentai

Įrankis	Paskirtis	Privalumai lyginant su standartine IS
Taisyklių specifikavimo įrankis	Naudojamas taisyklių aprašymui, redagavimui. Dažniausiai turi grafinę vartotojo sąsają, supaprastintą sintaksę arba grafinę notaciją veiklos taisyklėms modeliuoti ir yra skirtas veiklos analitikams.	• Garantuoja greitą veiklos politikos keitimą (nereikia perprogramuoti sistemų, pakeitimai vykdomi per specifikavimo įrankį). • Paprastesnis darbo principas (veiklos politikos nustatymą, pakeitimus atlieka veiklos analitikai, be programuotojų pagalbos).
Taisyklių saugykla	Saugo veiklos taisykles (veiklos politiką) ir kitą susijusią informaciją (veiklos duomenų žodynus ir pan.)	• Užtikrina sistemos skaidrumą. Kai veiklos logika atskirta nuo programinio kodo ir saugoma saugykloje, ją lengviau nagrinėti, be to, tai atlieka veiklos analitikai. • Leidžia pakartotinį veiklos taisyklių panaudojimą
Taisyklių interpretavimo procesorius	Įgalina veiklos taisyklių mechanizmo realizaciją.	• Atskirta logika gali būti naudojama kaip nutolusi, pavyzdžiui, kaip tinklo paslaugos (web servisai).

Trijų lygių architektūros realizacija labai įvairi. Skiriasi realizavimo priemonės, techninės detalės, pati veikimo logika. Pavyzdžiui, taisyklių saugyklas vieni gamintojai linkę talpinti į duomenų bazę, kiti duomenų saugojimą realizuoja failuose. Taip pat išskiriami 2 pagrindiniai taisyklių interpretavimo procesorių tipai, kurie nusako sprendinių išvedimo mechanizmus.

Tiesioginio išvedimo mechanizmas eina nuo gautų faktų link išvadų. Atgalinio išvedimo mechanizmas analizuoja reikiamas įrodyti išvadas, ir nustato, kokie faktai gali tas išvadas patvirtinti. Didžiausia įvairovė sutinkama taisyklių specifikavimo įrankių pasirinkime. Pagrindiniai taisyklių redaktorių tipai būtų du: palaikantys grafinių elementų modeliavimą, ir paprastesni, naudojantys taisyklių užrašymo šablonus.

Kertinis viso mechanizmo akmuo yra pati veiklos taisyklė – būtinybė pateikti veiklos logiką fiksuota forma, poreikis vienodai interpretuoti tai, kas užrašyta.

Vartotojų grupės, kurioms aktualus atliekamas įrankio tyrimas:

- IS projektuotojai (turi numatyti 3 lygių architektūros sudarymą, atskirti nuolat kintančią veiklos logiką nuo pastovios logikos. Oracle Business Rules taikymo atveju jie turėtų išskirti taisyklėms pateikiamus faktus, taisyklių nulemiamus veiksmus, nustatyti konfigūracijos specifikaciją „Rule Author“ įrankyje: objektų ir jų savybių matymo ribas, pradines sąlygas);
- programuotojai (turi atlikti faktų perdavimą ir pirminės aplinkos paruošimą “ Rule Author“ įrankyje, sukurti taisyklėmis paremtos sistemos realizaciją Java aplinkoje);
- veiklos analitikai: (atlieka veiklos logikos konstravimą: nustato, keičia, testuoja taisykles).

2.3. Panašių sistemų analizė

Šiandien yra nemaža komercinių veiklos taisyklių saugyklų ir procesorių. Dažniausiai jų veikimo principas paremtas vienokia ar kitokia Rete algoritmo atmaina (Rete, JESS, CLIPS) taisyklių procesoriaus darbo pagreitinimui, tuo tarpų realizavimo priemonės, architektūra ir teikiamų paslaugų išvystymas gana skirtingi. 3 priede pateikiama komercinės organizacijos Bizrules.com sudaryta diagrama, atspindinti veiklos taisyklių procesorių vystymosi istoriją ir tendencijas. Iš diagramos matoma, kad OBR vienas jauniausių įrankių, vertinamas kaip rinkos pretendentas, kaip ir kito, vieno didžiausių programinės įrangos gamintojų, Microsoft įrankis Business Rules Framework. Antra vertus, būtent didieji „platformų žaidėjai“ [9] yra linkę perimti, įsisavinti ar perpirkti potencialą turinčius technologinius sprendimus, o galimybė juos integruoti įmonių tarpinėje įrangoje (Middleware) skatina šių įrankių naudojimą ir paplitimą.

Analizuojamo įrankio įvertinimui rinkoje buvo atliktas jo palyginimas su Microsoft įrankiu Business Rules Framework, bei vienu lyderiaujančių šiuo metu veiklos taisyklių srityje, įrankiu Fair Isaac Blaze Advisor. Įrankių palyginimas pateiktas 3 lentelėje. Reikia pabrėžti, kad Business Rules Framework įrankio palyginimas atliktas remiantis dokumentacija.

3 lentelė

Oracle Business Rules ir MS Business Rules Framework ir Fair Isaac Blaze Advisor įrankių palyginimas

Kriterijai	Oracle Oracle Business Rules	Microsoft Business Rules Framework	Fair Isaac Blaze Advisor
Lyginamos versijos	Serverio 10g (10.1.3.1.0) versija.	BizTalk Server 2004.	6.5 versija.
Aplinka	Kaip įrankis Oracle programų serveryje.	Kaip savarankiška programa MS BizTalk serveryje.	Savarankiška programa įvairioms OS.
Diegimas	Būtinas Oracle programų serverio diegimas su SOA (tam būtinas ir duomenų bazės diegimas).	Būtinas MS BizTalk serverio diegimas.	Lengvai diegiamas, savarankiškas produktas. Pagrindiniai reikalavimai keliami JVM.
Dokumentacija, vartotojo vadovas.	Ribota, netiksli net ir įrankio viduje.	Nebuvo atliktas produkto diegimas.	Plati, išsami. Įrankio vartotojo vadovas demonstratyvus.
Taisyklių specifikavimo įrankis	Rules Author.	Business Rules Composer.	Pati programa.
Vartotojo sąsaja	GUI taisyklių specifikavimo įrankiui. Ribota ir gana nepatogi.	GUI.	Patogi, gerai organizuota, platus grafikos elementų (modeliavimui) naudojimas.
Autorizacija, įrankio pasiekiamumas	Autorizacija būtina. Rolėmis ribojamas vartotojo teisės veiksmams (kurti žodyną, keisti kintamuosius) bet ne konkrečioms žodyno versijoms.	Rolėmis gali būti ribojamos tiek teisės į žodynus, tiek į taisyklių rinkinius. Skirtingo lygio ribojimai veiksmams (kūrimui, keitimui).	Autorizacija nenaudojama. Programa diegiama lokaliame personaliniame kompiuteryje.
Žodynų valdymas	Žodynų versijų palaikymas.	Žodynų versijų palaikymas.	Žodynų versijų palaikymas su komandinio darbo galimybe (pakeitimų valdymu)
Priimami faktai	Faktai Java klasės arba XML dokumentai. (RL klasės vidiniam naudojimui).	Faktai XML, DB elementai, .Net klasės komponentai.	DB elementai, XML dokumentai, Java objektai, .NET/COM objects, COBOL copybooks.
Faktų kontrolė	Faktų (bei jų argumentų) matomumo ribos.	Neaptikta.	Nėra.
Žodyno objektai	Faktai, kintamieji, išvardinimai, rėžiai,	Faktai, kintamieji, apribojimai.	Faktai, faktų objektai, kintamieji, išvardinimai,

	reguliarios išraiškos, funkcijos.		įverčių medžiai, taškų modeliai, sprendimų lentelės, funkcijos.
Taisyklių organizavimas	Taisyklių grupavimas taisyklių rinkinių pagalba.	Taisyklių grupavimas kaip veiklos politika (Policy).	Taisyklių grupavimas taisyklių rinkinių pagalba, taisyklių eigos sudarymas. Išvystytas vykdymo kelio modeliavimas.
Žodynų publikavimas	Eksportuojama tik į saugyklos tipo failus.	Importuoti (XML -> SQL rule) ir eksportuoti (SQL ->XML) strategijas (policy)	Eksportuojama kaip archyvų failai, ar projekto dalys.
Taisyklių panaudojimo apribojimai	Nėra.	Rakinti taisykles, kad jų nebūtų galima panaudoti iš taikomųjų programų.	Neaptikta.
Testavimo galimybės	Elementarus taisyklių rinkinių testavimas įrankio viduje. Sunkiai naudojamas.	Numatytas taisyklių testavimas įrankio viduje.	Plačiai apimančios, intuityviuos testavimo galimybės. Testavimo atvejų kūrimas, testavimo šablonų kūrimas, testavimo automatizavimas.
Pakeitimų priėmimas	Realaus laiko pakeitimai.	Nepilnai realaus laiko pakeitimai.	Nėra bandyta su klientinėmis programomis.
Taisyklių saugojimo būdas	Neaptiktas vidinis veiklos taisyklių saugyklos architektūros aprašymas.	Saugyklos duomenys saugomi SQL DB arba XML failuose.	Neaptiktas vidinis saugyklos aprašas.
Taisyklių procesoriaus veikimo optimizavimas	Tiesioginio sprendimų išvedimo procesorius, naudoja Rete algoritmą taisyklių optimizavimui ir konfliktų sprendimui. Konfliktų sprendimas remiasi prioritetais.	Tiesioginio sprendimų išvedimo procesorius, naudoja Rete algoritmą taisyklių optimizavimui ir konfliktų sprendimui. Konfliktų sprendimas remiasi prioritetais. Taip pat naudojamas Faktų skirstymas į ilgalaikius ir trumpalaikius našumo pagerinimui. Faktų kešavimas.	Rete III naudojimas. Išsamus mechanizmas neaprašomas.
Monitoringas	Taisyklių aktyvacijos, faktų judėjimo, taisyklių rinkinių aktyvacijos monitoringas.	Veiklos taisyklių procesų stebėjimas (monitoringas).	
Individualus pritaikymas	Taisyklių individualaus pritaikymo galimybės (Dalykinės srities specialistui).	Taisyklių individualaus pritaikymo galimybės (Dalykinės srities specialistui).	Paprasto (tiesioginio) būdo nerasta. Bet yra galimybės programuoti naudojant kintamųjų

			reikšmes. Toks būdas daug plačiau apimantis, nei kituose įrankiuose.
Taisyklių kalbos konverteriai.	Nėra galimybės konvertuoti.	Galimybė konvertuoti į BRL (XML grįsta kalba).	Neaptikta.

Po atlikto preliminaraus įrankių palyginimo galima teigti, kad labiausia išvystytas, daugiausia galimybių bei draugiškiausią vartotojo sąsają turi Blaze Advisor. Tačiau vienas iš pliusų, skatinančių OBR ir MS BRF naudojimą – jų integravimas į programų serverius, tad organizacijoms, turinčioms šių gamintojų aplinkas, nereikia papildomai rūpintis naujų įrankių įsigijimu bei diegimu.

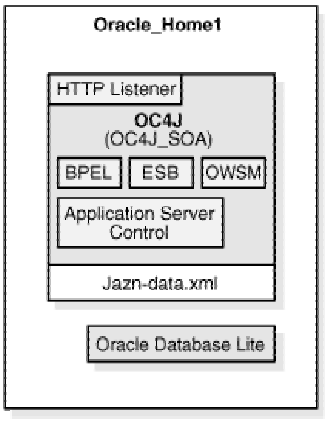
2.4. Tyrimo aplinkos paruošimas

Kaip buvo apžvelgta panašių sistemų palyginimo metu - Oracle Business Rules yra Oracle programų serverio komponentas. Todėl šis tyrimas buvo atliekamas naudojant tokius resursus:

4. lentelė

Darbo metu naudoti techniniai ir programiniai resursai

Parametras	Reikšmė	Pastabos
Kompiuterio parametrai	2,4GHz Intel(R) Celeron(R), 1,25GB RAM personalinis kompiuteris.	Tyrimas atliktas personaliniu kompiuteriu, kurio parametrai atitiko reikalavimus, tačiau būtų per žemi realios sistemos darbo metu.
Operacinė sistema	Microsoft Windows Server 2003EE su SP2	OS, reikalingas Oracle programų serverio palaikymui.
OBR aplinka	Oracle Aplication Server 10.1.3.1, pagal įprastinę (basic) J2EE Server + SOA Suite topologiją.	OBR yra tinklo paslaugų architektūros SOA dalis. SOA diegiama kartu tu programų serveriu, pagal vieną iš 8 topologijų (nurodytai programų serverio versijai). Pasirinkta įprastinė topologija, rekomenduojama SOA kūrėjams [10].
Klientinė dalis	Java aplinkai naudota Eclipse platforma, 3.3.2 versija.	



2 pav. Oracle programų serverio išdėstymo topologija, naudota tyrimo metu.

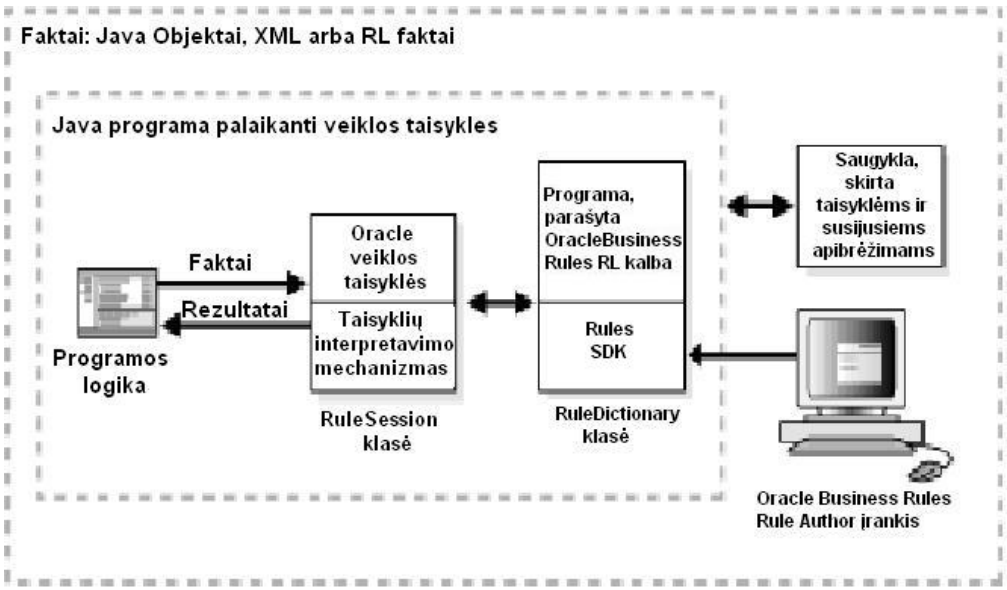
Išbandžius įrankį susidaro įspūdis, kad sprendimo esmė daugiau ne techninėse galimybėse, o veikimo logikoje (pati veiklos taisyklių koncepcija). Jei darbo tikslas tėra išbandyti OBR veikimą, arba naudoti šį įrankį, kaip vieną iš galimų veiklos taisyklių procesorių, kuriant Java architektūros programinę įrangą– tai tampa sudėtinga dėl dviejų pagrindinių priežasčių: Oracle serveris brangus produktas, kuris reikalauja įgūdžių jį diegiant ir valdant. Naujausios Oracle produktų versijos ir dokumentacija (MetaLink konsultacijų centras) prieinamos tik turint komercines licenzijas. Reikia pripažinti, kad OBR įrankis skirtas ribotai auditorijai, jau naudojančiai Oracle produkciją.

3. ORACLE VEIKLOS TAISYKLIŲ ĮRANKIO ARCHITEKTŪROS IR SAVYBIŲ ANALIZĖ

Šiame skyriuje bus aptarta Oracle veiklos taisyklių įrankio architektūra ir veikimo principai. Architektūrinius ir techninius įrankio aprašus galima rasti įrankio dokumentacijoje [7], [8]. Taisyklių specifikavimo (Rule Author) įrankio vartotojo sąsajos apžvalga pateikta, siekiant vaizdžiau išaiškinti įrankio darbo organizavimą. Taip pat buvo sudarytas sąrašas trūkumų, kurie kliudė ar sunkino vartotojo darbą, bei pasiūlymų, kaip galbūt įmanoma vartotojo sąsają organizuoti patogiau.

3.1. Oracle veiklos taisyklių architektūra

Oracle Business Rules sudarytas iš 3 esminių dalių: taisyklių saugyklos, taisyklių specifikavimo ir taisyklių interpretavimo įrankių, realizuotų Java kalbos klasėmis, atliekančių taisyklių vykdymą ir suteikiančių priėjimą prie taisyklių saugyklos. Tipiniu atveju OBR klientas yra Java kalba parašyta programa, galinti naudotis taisyklėmis. Antra vertus, esant poreikiui Javą klasę, naudojančią OBR taisykles galima kviesti kaip Web servisą. Paveiksle pateikiama įrankio veikimo schema.



3 pav. Oracle Business Rules architektūra

Sudėtinių dalių aprašas:

- Veiklos taisyklių saugykla, savarankiška OBR įrankio dalis. Tai failas, kuriame saugomi veiklos žodynai ir taisyklės. Gali būti pasiekiamas per tinklinę sistemą (failas, matomas Oracle programų serveriui, arba kaip failas, pasiekiamas per WebDAV (Web-based Distributed Authoring and Versioning) priemones.
- OBR Rule Author skirtas – taisyklių specifikavimo įrankis, skirtas taisyklių apibrėžimui, modifikavimui ir valdymui. Taip pat yra programų serveryje, bet veiklos analitikui ar programuotojui pasiekiamas per interneto sąsają.
- Taisyklėmis paremtos Java programos pusėje veikia taisyklių interpretavimo mechanizmas (taisyklių procesorius). Jis realizuotas Java kalbos RuleSession klase, kuri priima iš programos ateinančius faktus ir naudodama saugykloje esančias taisykles vykdo jų apdorojimą ir pateikia rezultatus.
- Taisyklių valdymui saugykloje (programiniam kūrimui, pasiekimui) skirti įrankiai yra Rules SDK pakete (Java kalbos klasė RuleDictionary). Ši klasė apima ir Oracle veiklos taisyklių kalbą RL (Rule Language). RL – aukšto lygio į Java panaši kalba, kuri atlieka taisyklių specifikavimą. Per ją Java programos savo objektus pateikia kaip faktus, o veiklos taisyklės gali kreiptis į objektų atributus, inicijuoti metodus.

Reikėtų pabrėžti, kad faktai fiziškai nėra pernešami į saugyklą. Java kalba rašytas kodas nuosekliai vykdo logiką, kol prisireikia veiklos taisyklių paslaugų. Tada konkretus objektas perduodamas apdoroti taisyklių mechanizmui (RuleSession Java bibliotekai), šis naudoja RuleDictionary biblioteką, iš veiklos taisyklių saugyklos perima veiklos taisykles, kaip elgtis su turimu objektu ir priima sprendimą. Kliento programa, kuri perdavė valdymą OBR klasėms, tęsia darbą, sulaukusi rezultatų.

Kaip teigia OBR dokumentacija [7], sudarant taisyklių žodyną ir atliekant Java faktų importą, klasių aprašai nėra kopijuojami į taisyklių saugyklą. Šią procedūrą galima laikyti analogiška bibliotekų importavimui programose (nuorodos sukūrimą į reikiamą failą, turintį visą informaciją). Detaliau žodyno kūrimas aptariamas sekančiame skyriuje.

3.2. Vartotojo sąsajos ypatybės

OBR matomoji dalis (grafinė sąsaja) – Rule Author, skirtas veiklos taisyklių ir žodynų valdymui. Įrankis realizuotas kaip interneto (web) prieiga, jo privalumai ir trūkumai apžvelgiami 3.3

poskyryje. Vartotojo sąsajos aprašas gana gerai atspindi visą darbo su OBR principą ir eigą. Dėl gana siauros įrankio dokumentacijos čia pateikiama informacija buvo surinkta bandant įrankio aplinką. Beveik visą Rules Author funkcionalumą, kaip kuriose srityse plačiau, galima pasiekti naudojant Rules SDK klases ir dirbant komandinėje eilutėje. Tyrimo metu naudotasi Rules Author galimybėmis, o sudėtingesnės situacijos aprašytos tiesiog RL (Rule Language) kalbos blokais, palaikomais įrankio viduje.

Įrankio darbo principai vyksta atitinkamai pagal kortelių seką: „Saugykla“, „Apibrėžimai“, „Taisyklių rinkiniai“, „Pritaikymas“ ir „Rule Language“ aplinka.

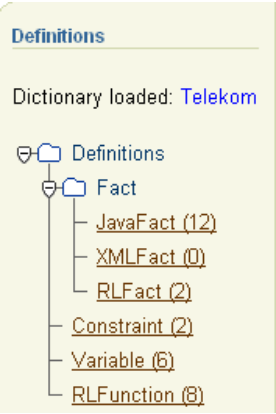
4 pav. Kortelių meniu Rule Author aplinkoje

Sritis „Saugykla“

Srityje „Saugykla“ vyksta saugyklos ir žodynų valdymas. Viena saugykla savyje talpina neribotą skaičių veiklos žodynų, kiekvienas žodynas gali turėti keletą versijų (nebuvo rasta apribojimų šiems parametrams). Jei saugyklos tipas failas, vienu metu su vienu žodynu dirba vienas asmuo. Veiklos taisyklių keitimai netrukdo darbui su veiklos taisyklėmis (nereikia stabdyti programų veikimo).

Sritis „Apibrėžimai“

Ji skirta taisyklių žodyno sudarymui. Nuo dalykinės srities žodyno paruošimo priklauso taisyklių naudojimo sėkmė: tiek sąlyginės veiklos taisyklių dalies galimybės (IF apribojimai), tiek procedūrinė dalis, atliekama vykdant taisyklę. Plačiau apie žodyno apribojimai bus nagrinėjami tyrimuose. Toliau pateikiamas žodyno apibrėžimų tipai, jų paskirtis ir naudojimas.



5 pav. Apibrėžčių tipai Rule Author įrankyje.

5. lentelė

Java apibrėžčių tipai ir jų aprašymas

Java faktai	Importuojami iš išorės Java klasių aprašai. Kaip faktai naudojami taisyklių sužadinimui. Reikalavimai klasei: ji turi būti .jar archyve, įdėta savo vardo srities (namespace) aplanke. Jei bus naudojamos kitos standartinės Java klasių funkcijos (datų apdorojimas, išvedimas ir t.t.) čia taip pat turi būti atliekamas reikiamų (sisteminių) Java klasių importas iš java, javax ir org vardo sričių. Taisyklių saugykla nedaro importuojamų Java klasių kopijų, o tiesiog kaupia nuorodas į jas [7]. Taigi OBR visiškai priklausomas nuo programų serverio OC4J konteinerio.
XML faktai	Importuojami iš išorės XML objektai. Kaip faktai naudojami taisyklių sužadinimui Šio tyrimo metu nebuvo nagrinėti.
RL faktai	Įrankio viduje galimos kurti Rules Language varotojo sudarytos klasės (greičiau struktūros). Savyje turi tik savybes, bet kurio iš anksčiau importuotų ar sisteminių objektų tipų. Gali būti naudojami kaip faktai naudojami taisyklių sužadinimui
Apribojimai	3 tipų apribojamai: diapazonas (skaičių), išvardijimas ir reguliarioji išraiška. Globalūs sesijos mastu.
Kintamieji	Bet kurio iš anksčiau importuotų (ar sisteminių) tipų kintamasis su konkrečia reikšme. Čia apibrėžtas kintamasis globalus sesijos mastu.
RL Function	Rule Language kalba rašomos funkcijos. Stipriai praplečia įrankio galimybes. Gali būti naudojamos kaip: 1) pakartotinio panaudojimo funkcijos, kada naudojamos keliose taisyklėse; 2) Darbo organizavimui. Taisyklių suaktyvinimui iš klientinių programų, kada reikia atlikti veiksmus prieš atliekant faktų apdorojimą ar po jo(pavyzdžiui nustatyti globalaus kintamojo reikšmę prieš faktų sąrašo apdorojimą, baigus pasiimti ir atspausdinti jo reikšmę). Taip pat patogu nustatyti naudojamus taisyklių rinkinius ir jų eiliškumą (jei nenorima to daryti kliento pusėje), monitoringo galimybes ir t.t. Plačiau aptariama 6 skyriuje.

Didžiausias dėmesys turėtų būti skiriamas Java klasių aprašų importui. Išbandžius skirtingus variantus, suformuoti Java klasei keliami reikalavimai.

6 lentelė

Java apibrėžčių tipai ir jų aprašymas

Ypatybė Java kalboje	Kaip supranta OBR	Pastabos
Privataus tipo atributai su reikšmės nustatymo ir gavimo savybėmis (get, set).	Virsta savybėmis. Matoma tiek public, tiek private dalis. Taisyklių užrašyme galima naudoti abi	Savybių generavimas privatiems laukams būtinas, kitaip jie nėra matomi. Pageidautina, kad savybės atitiktų Java kalbos principus (kitaip nėra matoma).
Public tipo atributai	Virsta laukais	Patogu naudoti klasių ryšių modeliavimui
Metodai ir funkcijos	Virsta metodais, funkcijomis	
Konstruktoriai	Nėra matomi aprašuose	Yra naudojami kuriant naujus faktus, todėl svarbus apibrėžti. Palaiko keletą konstruktorių. Naudojant GUI, šablonu kuriant naują faktą pateikiamas vienas, tačiau RL bloke galima naudoti bet kurį.

Fakto klasės atributai gali būti paslėpti nuo tolesnio publikavimo žodyne, naudojant atributą „nematomas“, taip pat pervadinti veiklos sritį o ne programavimo aplinką atitinkančiu pseudonimu. Tokie darbai įeina į veiklos žodyno paruošimo eigą.

Sritis „Veiklos taisyklės“

Užrašomos ir modifikuojamos taisyklės, organizuotos į veiklos taisyklių rinkinius. Rekomenduojama turėti porą rinkinių tai pačiai veiklos sričiai (tarkime, agresyvi ir nuosaiki verslo politika). Su kuriuo rinkiniu dirbama nurodoma arba kliento programoje arba darbą organizuojančioje RL funkcijoje. Kliento programa naudoja RuleSession klasės metodus, Prisijungimo parametrais nurodomi skirtingi lygiai: saugykla, žodynas, žodyno versija, naudotini taisyklių rinkiniai. Taisyklės OBR turi pavadinimą, aprašą ir prioritetą („1“ atitinka minimumą, „99“ maksimumą, „0“ – „nenurodyta“). Taisyklės organizuotos IF <struktūra> THEN <veiksmas> principu. Kadangi šiose dalyse galima naudoti viską iš paruošto žodyno (žodyno faktų objektus su matomais atributais ir metodais, RL funkcijas) daroma prielaida, kad tokia struktūra gali įgyvendinti bet kokią logiką. Tyrimų tikslas tai įrodyti.

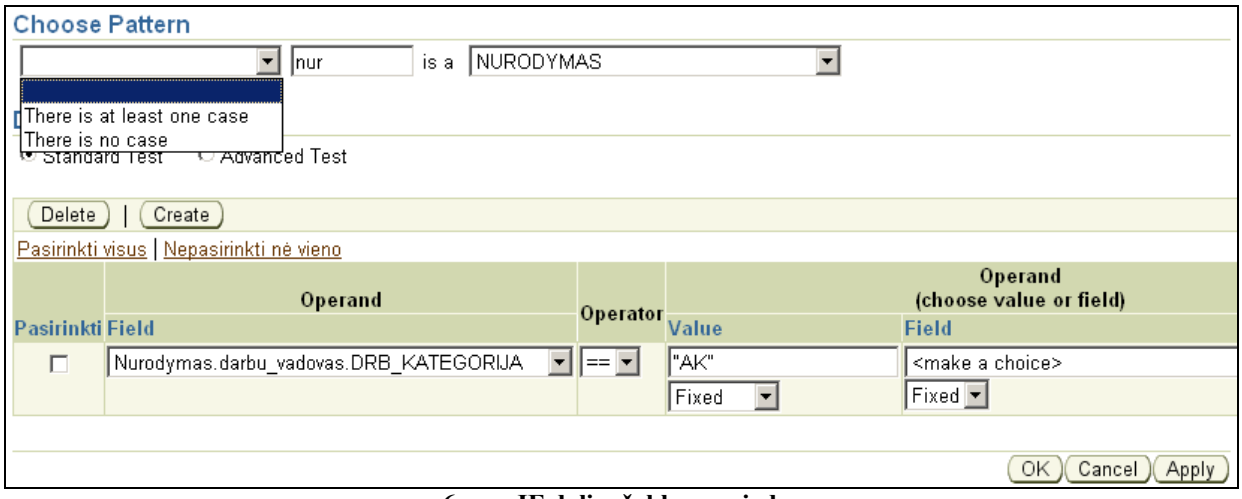
Taisyklių užrašymui Rule Author pateikia šablonus, taip sumažėja veiksmų variacijų galimybės: savybių matomumas, metodų pasiekiamumas, ypač, kada atsiranda poreikis pasiekti

giliau, nei pirmo lygio hierarchine priklausomybe susijusius objektus, pavyzdžiui, dirbama su objektais, tarpusavyje susijusiais ryšiu ir bandoma kviesti *vaiko* objekto metodą ar gauti savybę, taip pat, kada analizuojami sąrašai. Tokia sugriežtinta kontrolė didina veiklos valdymą, bet kartais apsunkina galimybę užrašyti pačią veiklos taisyklę. Tokiu atveju norimą logiką pravartu formuoti naudojant RL kalbos blokus.

Taisyklės sąlyginės dalies šablonas (IF dalis) pavaizduotas 6 paveiksle. Nurodoma nagrinėjamo objekto klasė (kokio fakto laukiama), pseudonimas, laukiamas santykis su faktais ir norima sąlyga. Šablonas atitinka standartinę veiklos taisyklės išraišką su tam tikais patobulinimais. Įdomesnė dalis – taisyklės santykis su faktu. Taisyklė aktyvuojama:

- jei nieko nenurodyta - kiekvienam objektui;
- „yra nors vienas toks atvejis“ – galioja faktų blokui, ieškoma, ar tarp tokio tipo faktų yra tenkinančių sąlygas;
- „nėra tokio atvejo“ - nei vieno nurodytos klasės fakto, su iškelta sąlyga.

Pastarieji du veiksmai dirba su objektų grupe. Toks santykio su faktais išskyrimas artimas duomenų bazės veikimo principui, kada lentelių trigeriai, priklausomai nuo rūšies, suveikia kiekvienam įrašui arba visai grupei. Šablono sąlygų kiekis nėra ribotas, jos jungiamas AND operatoriumi. Bet tokiu šablonu užrašyti įmanoma nesudėtiną logiką. Tarkime šablonas negali patikrinti nurodymo brigados dydžio, išreškiamo taisykle Nurodymas.vykdo.size() != 0, nes 3 hierarchijos lygyje esančio metodo pasiekti neišeina. Pažangus sąlygų rašymo režimas leidžia naudoti RL blokus.



6 pav. IF dalies šablono vaizdas

Then dalis neturi konkretaus šablono, bet yra dar stipriau apribota. Panašu, kad Rule Author įrankio paskirtis – struktūrizavimas. Then dalyje privalu nurodyti, kokio tipo veiksmai bus atliekami

ir pagal tai siūloma tam tikra šablono aplinka, kurios negalima apeiti su pažangiu režimu. Pagrindiniai tipai ir jų savybės aptariamos lentelėje.

7. lentelė

Java apibrėžčių tipai ir jų aprašymas

Veiksmo tipas	Reikšmė	Siūlomas šablonas
Pareikšti (Assert)	Nurodytas objektas registruojamas apdorojimui Agendoje. Plačiau skyriuje „**Error! Reference source not found.**“	Pagal šabloną dažniausiai galima rinktis tik tą patį objektą, kuris sąlygojo taisyklės iškvietimą.
Pareikšti naują (Assert New)	Į taisyklių apdorojimo mechanizmą sukuria ir paduoda naują objektą.	Objektas gali būti naujas nepriklausomas (neturintis konstruktoriuje kitų tipų objektų) Java objektas, arba RL faktas.
Priskirti (Assign)	Keičia objektą ar jo atributus.	Galimybė keisti apdorojamo fakto matymo ribose esančius objektus ir jų savybes.
Iškviesti (Call)	Kviečia funkcijas.	Gali būti esamojo objekto metodai, žodyne apibrėžtos RL funkcijos, arba Main klasės metodai.
Atitraukti (Retract)	Išmeta objektą iš taisyklių interpretavimo mechanizmo. (Tolesnės taisyklės nebebus taikomos)	Analogiškai assert atvejui leidžiama taikyti tik tam pačiam objektui, sąlygojusiam taisyklės iškvietimą.
Rule language išraiška (RL)	Bet koks teisingas kodo fragmentas, parašytas RL kalba.	Stipriai praplečia procedūrines taisyklės galimybes. Galimi visi aukščiau išvardinti atvejai bei jų kompozicijos. Tačiau kodas rašomas „pliku“ teksto redaktoriumi, todėl būtinas geras paruošto žodyno bei RL žinojimas.

Gana elementarūs RuleAuthor sąsajos trūkumų taisyklių rinkinių sudaryme, kad nėra galimybės išjungti į taisyklę įtrauktą veiksmą, negalima sukeisti jų vietomis. Blogai apsirašius vykdomų veiksmų seką – teks taisyklę perrašyti iš naujo.

Sritis „Pritaikymas“

Skirta greitam veiklos taisyklių pakeitimui. Greitai pakeičiamos gali būti tik tos taisyklės, kurios užrašytos specialiu būdų naudojantis šablonu, o kadangi šablonai gana apriboti savo galimybėmis, visuose tyrimuose ši įrankio savybė buvo sunkiai realizuojama.

Sritis „Rule Language“

Sritis turi tris esmines funkcijas. Yra būtina vartotojui, nes įgyvendina tą funkcionalumą. kurio trūksta kitose įrankio dalyse. (Pavyzdžiui, sintaksės tikrinimas RL kode gali būti atliekamas tik kreipiantis į šį sritį. Vienas iš didelių įrankio minusų.).

- RL kodo generavimas konkrečiam taisyklių rinkiniui. Patogu, kad reikia pamatyti kaip atrodo programinis kodas.
- RL kodo tikrinimas. Sintaksės tikrinimo įrankis. Ribotos klaidos atpažinimo galimybės. Pagrindinis veikimo principas – nurodoma, kurioje generuoto kodo eilutėje galima rasti klaidą.
- Taisyklių testavimas. Galimybė naudojantis RL funkcijomis ištestuotai pasirinktą taisyklių rinkinį neturint išorinės Java programos. RL funkcija turi sukurti ir paduoti reikiamo tipo faktus ir inicijuoti taisyklių rinkinio kvietimą.

3.3. Įrankio sąsajos trūkumai

Atlikus tyrimus, buvo išskirta nemaža įrankio trūkumų, tiek funkcionalumo tiek ergonomiškumo atžvilgiu. Toliau pateikiamas sąrašas su komentarais.

- Internetinė taisyklių specifikavimo įrankio sąsaja: lėtas veikimas, nestabilumas dėl JavaScript naudojimo, „nedraugiška“ sesija (neišsaugojus pakeitimų viename polapyje, negalima eiti į kitą).
- Nepatogus vartotojo sąsajos organizavimas. Pavyzdžiui:
    - patikrinti parašyto RL kodo sintaksę minimaliai kainuoja 3 žingsnius;
    - užmiršus fakto savybes ar metodus, teks išsaugoti neužbaigtą kodą ir ieškoti žodyno.
- Automatinio sintaksės tikrinimo, nuspėjimo nebuvimas. RL kodo rašymui pateikiamas plikas teksto redaktorius.
- Ribota pagalba.
- Netikslūs (abstraktūs) klaidų pranešimai.

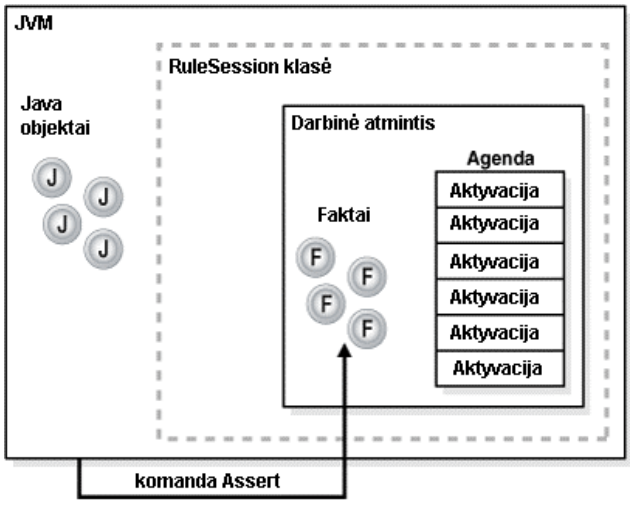
- Riboti šablonai. Dažniau veikia ne kaip pagalbos, o kaip galimybių apribojimų priemonė, todėl būtini RL (Rule Language) kalbos pagrindai.
- Importuotų faktų vaizdavimas Rule Author įrankyje nepateikia konstruktoriaus. Problema, jei faktas palaiko porą konstruktorių, be programinių klasių aprašo negalima išnaudoti visų klasės galimybių.
- Nėra galimybės sudaryti importuotų faktų hierarchijos. Pageidautina savybė – Java dalykinės srities faktų atskyrimas nuo sisteminių klasių.
- Nėra galimybės išjungti taisyklių ar taisyklės procedūrinės dalies procesų. Jei rinkinyje atsirado šiuo metu nereikalinga taisyklė, ją tenka šalinti.
- Nėra galimybės sukeisti taisyklės vykdomų procesų vietomis.
- Pagal Business Rules Community pageidautina, kad taisyklė neštų savyje informaciją apie šaltinį ir autorių. Galbūt tai galima užrašyti taisyklės aprašo dalyje, bet specialiai išskirto informacijos lauko nėra.
- Sudėtingas Java paketų (.jar archyvų) padavimas įrankio testavimui.
- Pasitaikantis neapibrėžtas naujų žodyno objektų interpretavimas.

4. TAISYKLIŲ PROCESORIAUS VEIKIMO TYRIMAS

Šis tyrimas skirtas praktiškai išsiaiškinti OBR įrankio taisyklių procesoriaus veikimo principus. Pirmoje dalyje pateikiamas sutrumpintas taisyklių procesoriaus aprašas, dokumentuojamas RL kalbos specifikacijoje [8] ir esminės sąvokos ir komandos, į kurias reikėtų atkreipti dėmesį dirbant su veiklos procesoriumi. Jos išskirtos analizuojant dokumentaciją ir atliekant 4.2 poskyryje aprašytą tyrimą.

4.1. Taisyklių procesoriaus veikimo principai

OBR taisyklių procesorius naudoja tiesioginio išvedimo (Forward chaining) mechanizmą, faktų optimizavimui - JESS Rete atmainą. Procesoriaus veikimo principas pateikiamas 7 paveiksle. Veikimo esmė: Java kalbos objektai perduodami taisyklių procesoriui komanda Assert (čia jie tampa Java faktais, aktyvuojančiais taisykles), toliau nuosekliai, pagal nutylėjimą LIFO principu, kiekvieno fakto informacija sutapatinama su taisyklės sąlygine dalimi, Agendoje fiksuojama fakto ir taisyklės aktyvacija, ir, priklausomai nuo taisyklių prioritetų, atliekamas agendoje registruotų įvykių vykdymas. Jei faktai pasikeičia (toliau nebeatitinka taisyklės sąlyginės dalies), jų aktyvacija iš Agendos pašalinama. Faktų registracija Agendoje vyksta tik jų padavimo ar išmetimo momentu (Assert, Retract), todėl atlikus pakeitimą su faktu, ir norint, kad jis vėl būtų apdorojamas taisyklių rinkinio, būtina jį vėl užregistruoti (panaudoti assert komandą). Tyrimas OBR veiklos taisyklių procesoriaus darbo principams išsiaiškinti atliktas 3 skyriuje.



7 pav. Oracle Business Rules taisyklių procesoriaus veikimo schema

Vienareikšmiškai galima teigti, kad veiklos taisyklių atsakymas (darbo rezultatas) priklauso ne tik nuo taisyklių užrašymo ir paduodamų faktų, bet ir taisyklių procesoriaus darbo organizavimo.

Toliau pateikiami Rule Language kalbos raktiniai žodžiai, padedantys organizuoti taisyklių procesoriaus veikimą.

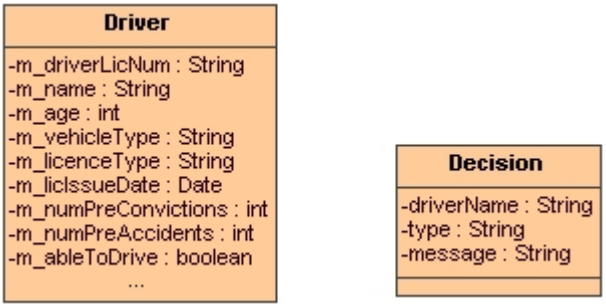
8. lentelė

Taisyklių procesoriaus veikimą veikiančios sąvokos ir komandos

Raktinis žodis	Reikšmė ir aktualumas
RuleSet	Taisyklių rinkinys. Vienu metu gali būti vykdomas tik vienas. Tik tada, kai einamajam taisyklių rinkiniui Agenda ištuštėja, pereinama prie kito taisyklių rinkinio vykdymo (jei toks užregistruotas sesijoje).
RuleSetStack	Taisyklių rinkinių stekas. Visi sesijai užregistruoti rinkiniai, vykdomi paeiliui. Vienas rinkinys į steką gali būti sudėtas kiek norima kartų. Egzistuoja RL komandos taisyklių rinkinių įtraukimui, išėmimui, taip pat postūmiui į steko viršų (Į einamojo rinkinio vietą). Taisyklių rinkinio steko sudarymas atitinka taisyklių vykdymo scenarijaus sukūrimą (RuleFlow).
Assert (AssertXPath)	Fakto padavimo į taisyklių procesorių ir registravimo Agendoje komanda. AssertXPath registruoja tarpusavyje susijusių faktų medį.
Retract	Fakto registracijos šalinimas iš Agendos. Realiai faktas ir toliau išlieka matomas taisyklių procesoriui, bet nėra registruojamas Agendoje.
Priority	Taisyklės gali turėti prioritetus, nusakančius taisyklių vykdymo tvarką. Taisyklių vykdymo tvarka nustatoma jau sudarytai Agendai. Pirmiausia vykdomos taisyklės su didesniu prioritetu, jei prioritetas vienodas, pirmenybė teikiama vėliausiai registruotiems faktams. Prioritetų galiojimas apsiriboja vieno taisyklių rinkinio matomu ribose.
Strategy	Įprastine tvarka taisyklių rinkiniui galioja steko LIFO sistema. Šis parametras yra keičiamas, gali būti nustatyta steko FIFO strategija.
Watch (Rules, Activations, Facts, Focus)	Procesoriaus procesų stebėjimui skirtos komandos. Galima stebėti taisyklių vykdymą, Agendos aktyvaciją, faktų registravimą, išregistravimą, modifikavimą, taisyklių rinkinio postūmį steke.

4.2. Procesoriaus veikimo tyrimas jį taikant „Automobilių nuomos“ dalykinei sričiai

Šis bandymas buvo atliktas su įprastine automobilių nuomos dalykine sritimi, kuri apibrėžta GUIDE projekte ir iki šiol dažnai taikoma veikos taisyklių tyrimui. Java programa turi vieną klasę Driver (vairuotojas). Žodyne sukuriama dar vienas RL tipo faktas Dessision (sprendimas). Jų struktūra atvaizduota 8 paveiksle. Nagrinėsime dalykinės srities veiklos taisyklę: „Vairuotojas gali išsinuomoti automobilį, jei jis vyresnis nei 15 metų“.



8 pav. Java fakto Driver ir žodyne sukurto RL fakto Decision klasės

4.3. Tyrimo eiga ir rezultatai

Veiklos taisyklių žodyne apsirašome 4 pagrindines taisykles, padėsiančias realizuoti koncepciniame modelyje apibrėžtą veiklos taisyklę bei stebėti jos vykdymo sąlygas. Taisyklių procedūrinės dalies veiksmus sunumeruojame, kad vėliau patogiau stebėtume taisyklių vykdymo eiliškumą.

9. lentelė

Taisyklės, padedančios įgyvendinti „nuoma tik vyresniems nei 15 metų“, taisyklę.

Eil. Nr.	Taisyklės pavadinimas	Sąlyginė dalis	Procedūrinė dalis
1.	Under_Age (nepilnametis)	Jei turime vairuotoją ir vairuotojas jaunesnis nei 15 metų	1.1 Spausdinti komandinėje eilutėje „Nuoma atmesta. Vardenio Pavardenio amžius netinkamas.“ 1.2 Sukurti naujo tipo kintamąjį Sprendimas su pranešimu apie atmetimą. 1.3 Šalinti vairuotoją.
2.	Show_decision (rodyti sprendimą)	Jei turime sprendimą	2.1 Parodyti sprendimą.
3.	Correct_age (tinkamas amžius)	Jei turime vairuotoją ir vairuotojas vyresnis	3.1 Sukurti naujo tipo kintamąjį Sprendimas su pranešimu, kad priimta.

		daugiau nei 15 metų	
4.	Show_driver_details (paviešinti vairuotojo detales)	Jei turime vairuotoją	4.1 Spausdinti komandinėje eilutėje „Vairuotojo duomenys Vardenis Pavardenis, amžius.“

Toliau pateikiamas sutrumpintas RL kodas šių taisyklių užrašymui. Kodas generuotas OBR įrankiu RL kortelėje automatiškai. Taip pat pradiniai duomenys (tik aktualių atributų informacija) pateikiami į taisyklių procesorių:

10. lentelė

Į taisyklių procesorių paduodami tokie duomenų objektai

Eil. Nr.	Paduodamas faktas Driver(vardas, amžius)
1.	Driver(Dave, 50)
2.	Driver(Qun, 15)
3.	Driver(Lance, 44)

```
1. ruleset vehicleRent
3.  {
5.    rule UnderAge
6.    { priority = 5;
8.      if ( ( fact carrental.Driver v0_Driver && ( (v0_Driver.age < 15))))
15.       {
16.        DM.println(“Nuoma atmesta: "+v0_Driver.name+" Netinkamas amžius, amžius yra:"+v0_Driver.age);
17.        assert (new DM.Decision(driverName: v0_Driver.name,type:"atmesti",message: "Netinkamas amžius "));
18.         retract( v0_Driver );
20.       }
21.   } //end rule UnderAge;
22.
23.   rule Show_decision
24.   {
25.     priority = 99;
26.     if ( (fact DM.Decision v0_Decision))
32.     {
33.       DM.showDecision(v0_Decision);
34.     }
35.   } //end rule Show_des,
36.
37.   rule Correct_age
38.   {priority = 5;
40.     if ( (fact carrental.Driver v0_Driver && ( (v0_Driver.age > 15))))
47.     {
48.       assert (new DM.Decision(driverName: v0_Driver.name, type: "priimta", message: "Vairuotojo amžius tinkamas"));
49.     }
50.   } //end rule Correct_age;
51.
52.   rule Show_driver_details
53.   { priority = 1;
55.     if ( (fact carrental.Driver v0_Driver))
61.     {
62.       DM.println("Vairuotojo duomenys"+v0_Driver.name+", amžius"+v0_Driver.age+".");
```

```
63.      }
64.    } //end rule Show_driver_details;
65.  }// end ruleset vehicleRent
```

Toliau lentelės forma pateikiami testavimo rezultatai. Čia pateikti 5 bandymai, demonstruojantys, kad skirtingų prioritetų nustatymas taisyklėms, o taip pat darbinių faktų pakartotinis patvirtinimas (assert) arba pašalinimas (retract) gali visiškai pakeisti gaunamą rezultatą, arba bent vykdymo sekos grandinės ilgį.

11. lentelė

Taisyklių procesoriaus veikimo principų tyrimas. Pradinės sąlygos ir rezultatai

Eil nr.	Keičiami parametrai	Taisyklių vykdymo eiliškumas (priekyje taisyklės numeris)	Gautas rezultatas	Komentarai
1.	Visų taisyklių prioritetas lygus 0	3. Correct_age (Lance) 2. Show_decision(Lance) 1. UnderAge(Qun) 2. Show_decision(Qun) 3. Correct_age (Dave) 2. Show_decision(Dave) 4. Show_driver_details(Lance) 4. Show_driver_details(Dave)	2.1Sprendimas: priimta, vairuotojas Lance, priežastis: Vairuotojo amžius tinkamas 1.1Nuoma atmesta:Qun, netinkamas amžius, amžius:15 2.1Sprendimas: atmesta, vairuotojas Qun, priežastis: Netinkamas amžius. 2.1Sprendimas: priimta, vairuotojas Dave, priežastis: Vairuotojo amžius tinkamas 4.1 Vairuotojo duomenys Lance, amžius 44 4.1 Vairuotojo duomenys Dave, amžius 50	Duomenys apdorojami tvarka LIFO, taisykles skaitant iš eilės. Turime tik 2 vairuotojų detales, nes po 1 eilutes Qun objektas buvo “išmestas”. Jo duomenys pasiekiami, bet taisyklės jam nebetaikomos
2.	Prioritetai: Show_decision =1 Show_driver_ details = 99 UnderAge = 5 Correct_age = 5	4.Show_driver_details(Lance) 4.Show_driver_details(Qun) 4.Show_driver_details(Dave) 1. UnderAge(Qun) 3. Correct_age (Dave) 3. Correct_age (Lance) 2. Show_decision(Qun) 2. Show_decision(Dave) 2. Show_decision(Lance)	4.1Vairuotojo duomenys Lance, amžius 44 4.1Vairuotojo duomenys Qun, amžius 45 4.1Vairuotojo duomenys Dave, amžius 50 1.1Nuoma atmesta: Qun, netinkamas amžius ,amžius:45 2.1Sprendimas: priimta, vairuotojas Dave, priežastis: Vairuotojo amžius tinkamas 2.1Sprendimas: atmesta, vairuotojas Qun, priežastis: Netinkamas amžius. 2.1Sprendimas: priimta, vairuotojas Lance, priežastis: Vairuotojo amžius tinkamas	Pakeitus prioritetus pirmiausia rodomi visų vairuotojų informacija, vėliau apdorojami duomenys. Dėl prioriteto, matome ir Qun informaciją (taisyklė suveikė prieš išmetant faktą iš Agendos), tuo tarpu Correct_age, Under_age ir Show_decision vykdymo eiliškumas nebeatitinka iškvietimo eiliškumo, užrašyto taisyklėse.
3.	Visų taisyklių prioritetus keičiame į	3. Correct_age (Lance) 2. Show_decision(Lance) 4.Show_driver_details(Qun)	2.1Sprendimas: priimta, vairuotojas Lance, priežastis: Vairuotojo amžius tinkamas 4.1Vairuotojo duomenys Qun, amžius 45	Eiliškumas pirmiausia sudedamas faktams (Lance, Qun, Dave), po to taisyklių eigai.

	0.Prie taisyklės Correct_age pridedam funkciją retract.	1. UnderAge(Qun) 2. Show_decision(Qun) 3. Correct_age (Dave) 2. Show_decision(Dave)	1.1Nuoma atmesta:Qun, netinkamas amžius, amžius:45 2.1Sprendimas: atmesta, vairuotojas Qun, priežastis: Netinkamas amžius. 2.1Sprendimas: priimta, vairuotojas Dave, priežastis: Vairuotojo amžius tinkamas	Kadangi prioritetai vienodi, pirma vykdoma vėliausiai redaguota taisyklė (Correct_age). Ji atliko fakto išmetimą iš procesoriaus, todėl nebepamatysime Show_desision(Lance)
4.	Pridėta nauja taisyklė Older. Kuri sutikusi 45 metų vairuotoją praneša "Sendinam" ir pakeičia amžių į 30. Taisyklės prioritetas 5. Correct_age taisyklės prioritetas = 4	Older(Qun) 1. UnderAge(Qun) 3. Correct_age (Dave) 3. Correct_age (Lance) 2. Show_decision(Dave) 2. Show_decision(Lance) 4.Show_driver_details(Qun) 2. Show_decision(Qun)	Sendinam 1.1Nuoma atmesta: Qun, netinkamas amžius ,amžius:30 2.1Sprendimas: priimta, vairuotojas Dave, priežastis: Vairuotojo amžius tinkamas 2.1Sprendimas: atmesta, vairuotojas Qun, priežastis: Netinkamas amžius. 2.1Sprendimas: priimta, vairuotojas Lance, priežastis: Vairuotojo amžius tinkamas	Apgaulinga OBR savybė, nepaisant to, kad taisyklių procesorius pakeitė fakto reikšmę (amžių = 30), yra išvedamas rezultatas, kad nuoma negalima. T.y. taisyklė dėl netinkamo amžiaus vis dar lieka galioti, nors jos išvedama informacija prieštarauja taisyklės sąlygos logikai.
5.	Taisyklėje Older pridedama dalis Assert	Older(Qun) 3. Correct_age (Qun) 2. Show_decision(Qun) 3. Correct_age (Lance) 2. Show_decision(Lance) 3. Correct_age (Dave) 2. Show_decision(Dave)	Sendinam 2.1Sprendimas: priimta, vairuotojas Dave, priežastis: Vairuotojo amžius tinkamas 2.1Sprendimas: priimta, vairuotojas Lance, priežastis: Vairuotojo amžius tinkamas 2.1Sprendimas: priimta, vairuotojas Qun, priežastis: Vairuotojo amžius tinkamas	Tik perregistravus faktus Agendoje gaunama korektiška informacija. Taisyklių suvykdymo eiliškumas pagal paskutinį užregistruotą Agendoje.

Taisyklių vykdymo eiliškumas priklauso nuo nustatytos strategijos. Tarkime, turime LIFO strategiją, tokiu atveju, jei taisyklės neturi prioriteto, vykdomos visos vienam faktui (paskutiniajam paduotam į darbinę sritį) priskirtos taisyklės. Jų eiliškumas priklauso nuo registravimo taisyklių žodyne. Jei taisyklėms nustatomi prioritetai, jie pakoreguoja jau sudarytos Agendos vykdymo seką. Ypač svarbus tampa prioritetų ir faktų padavimo ar šalinimo iš taisyklių rinkinio naudojimas, kada turimos tarpusavyje susijusios taisyklės. Tokiu atveju galima gauti visiškai skirtingus sprendimus, priklausomai nuo taisyklės prioriteto reikšmės.

5. SKIRTINGŲ VEIKLOS TAISYKLIŲ AUTOMATIZAVIMO GALIMYBIŲ TYRIMAS

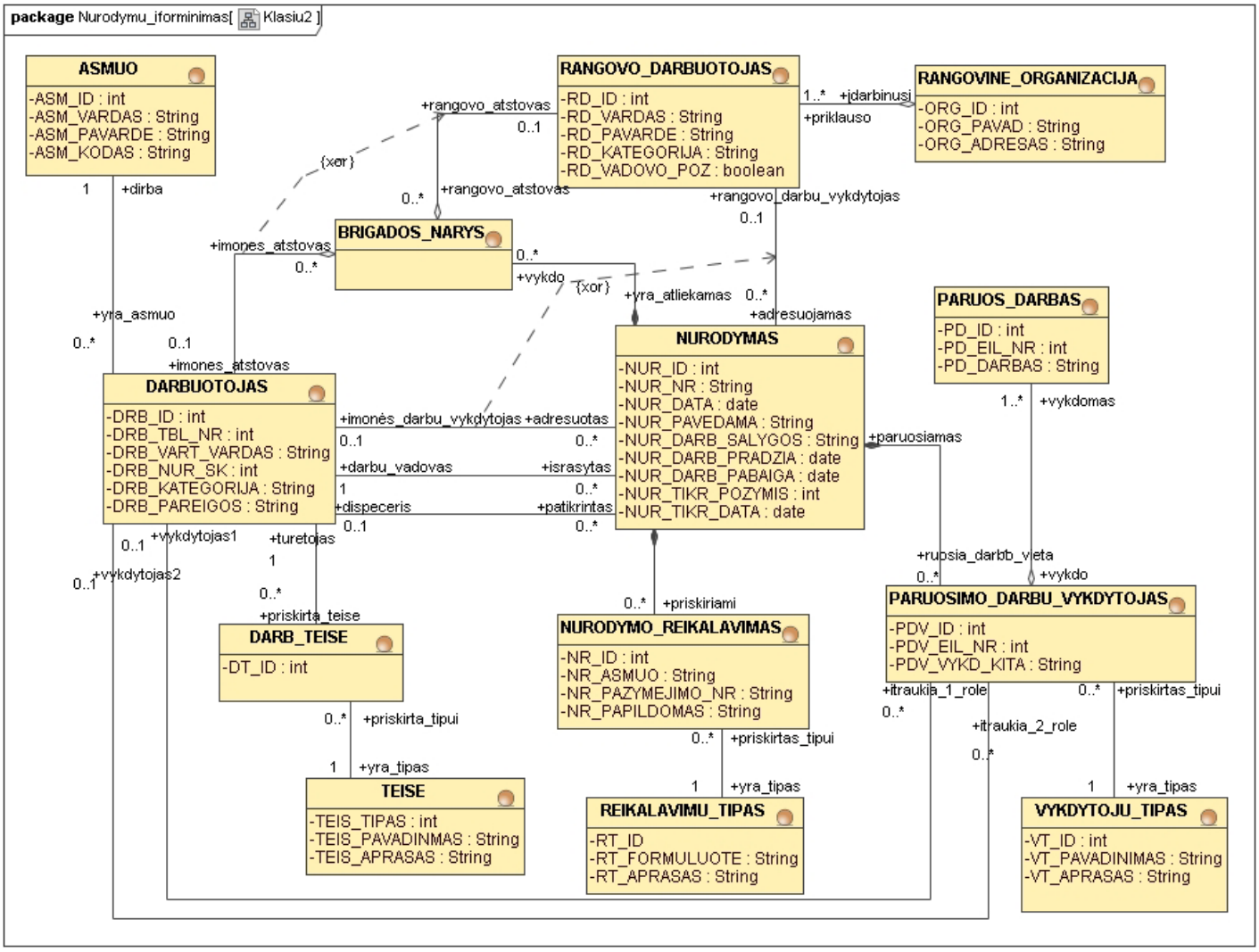
Praeituose skyriuose išsiaiškinta įrankio architektūra, veikimo principas, vartotojo aplinka. Šio tyrimo metu bus analizuojamos įvairių veiklos taisyklių užrašymo galimybės, bandoma sudaryti metodiką veiklos taisyklių automatizavimui.

5.1. Dalykinė sritis „Nurodymų įforminimas“

Šiam tyrimui pasirinkta dalykinė sritis – nurodymų įforminimas darbams elektros įrenginiuose. Dėl savo specifikos (gana griežtai apibrėžtos nurodymo formos ir jo rekvizitų, be to pačios veiklos srities – darbo su gyvybei pavojingais įrengimais) modeliuojama veikla turi nemažą ribojimų skaičių. Kadangi Oracle Business Rules įrankis dirba kaip faktus priimdamas Java objektus, o veiklos taisyklių žodynas sudaromas naudojant klasių aprašus (laikantis objektinio požiūrio į veiklos modeliavimą), dalykinės srities modelis 9 paveiksle pateikiamas klasių diagrama.

Veiklos rezultatas – pagal darbuotojų saugos reikalavimus suformuotas dokumentas (nurodymas), apibrėžiantis, kas, kur ir kada turi būti atliekama, kieno (išvardinant visus brigados narius), kas atsakingas už darbų atlikimą, kokiomis sąlygomis darbai atliekami, kokios darbo vietos paruošimo priemonės – išvardijant darbų eiliškumą ir vykdytojus, kokie papildomi reikalavimai keliami nurodymui. Pagal įmonės darbo taisykles, toks suformuotas nurodymas prieš vykdymą yra suderinamas (patvirtinamas) dispečerinės tarnybos.

Nurodymas su brigados nariais, paruošimo darbų vykdytojais ir reikalavimais sudaro kompozicijos ryšį, nes tai suprantama kaip viena visuma (vienas išbaigtas nurodymas), dalys viena be kitos netenka prasmės (programiniu požiūriu nurodymo klasės konstruktorius kuria visas su kompozicija susijusias dalis). Klasės teisė, reikalavimų tipas ir vykdytojų tipas naudojamos kaip klasifikatoriai ir yra kita sritis, lemianti apribojimus. Šiuo metu nusistovėjusios 7 skirtingos darbuotojų teisės, 20 reikalavimų tipų ir 7 skirtingi vykdytojų tipai. Paruošimo darbų vykdytojas suprantamas kaip vienetas (asmuo, komanda, tarnyba) atliekantis savo konkrečius darbo vietų paruošimo darbus. Pagal schemoje pateikiamą modelį vykdytojas arba susideda iš 2 žmonių komandos arba laisvu tekstu nurodomas skiltyje pdv_vykd_kita.



9 pav. Nurodymų įforminimo klasių diagrama

Ši sistema yra realizuota asp.NET + Oracle duomenų bazės architektūra, stengiantis kuo daugiau veiklos logikos perkleti į PL/SQL duomenų bazės blokus (neišvengiant funkcinių reikalavimų užrašymo, programinio kodo lygmenyje, nors dažnai dalis funkcinių reikalavimų yra linkę virsti veiklos taisyklėmis). Tokia architektūra buvo taikyta siekiant greitesnio ir lengvesnio pakeitimų valdymo. Tuo tarpu kuriant prototipą veiklos taisyklių išbandymui, uždavinys buvo supaprastintas (siekiant nekurti naujos klientinės programos dalies) iki jau suformuotų nurodymų audito. Sistemos testavimui naudojama Oracle duomenų bazėje išsaugota (korektiška) informacija, kuri nuskaitoma Java klientinės programos ir faktų pavidalu pateikiama į OBR.

5.2. Automatizuojamų veiklos taisyklių išskyrimas

Toliau bus išvardinti neformalia struktūra pateikti funkciniai reikalavimai dalykinei sričiai bei jų įvertinimas.

12 lentelė

Nurodymų įforminimo dalykinei sričiai keliami reikalavimai

Nr.	Reikalavimas	Komentaras
1.	Nurodymuose dalyvaujantys darbuotojai privalo turėti kategoriją (atestaciją darbui elektros įrenginiuose).	Veiklos taisyklė. Turi būti nurodytos, kokios kategorijos pripažįstamos įmonėje.
2.	Nurodymus išrašo tik darbų vadovo teises turintys darbuotojai.	Veiklos taisyklė. Kad nurodymas yra išrašomas – architektūrinė dalis, konkrečių darbuotojo teisių nurodymas, veiklos taisyklė (galimybė teisių kitimui).
3.	Nurodymo numeris susideda iš darbuotojo tabelio numerio + darbuotojo išrašytų nurodymų skaičiaus.	Veiklos taisyklė.
4.	Darbuotojų nurodymų skaičius tampa nuliu kiekvienų metų sausio 01 dieną.	Funkcinis reikalavimas.
5.	Kiekvienas darbų vadovas mato tik savo (asmeninius) nurodymus.	Funkcinis reikalavimas.
6.	Darbų vadovas nurodymus gali išrašyti tik savo miesto skyriaus darbų vykdytojams.	Funkcinis reikalavimas programinėje pusėje, gali būti veiklos taisyklė.
7.	Dispečeriai tvirtina savo miesto nurodymus bei regiono elektrotechnikos skyrių vykdytojus. (Šiuo metu įmonės organizacinių vienetų hierarchija suskirstyta į 3 regionus bei 20 miesto skyrių).	Funkcinis reikalavimas programinėje pusėje, gali būti veiklos taisyklė.
8.	Nurodymų adresatas (darbų vykdytojas) gali būti įmonės atstovas arba rangovinė organizacija.	Veiklos taisyklė dėl būtinumo kontroliuoti ryšį. (Įmonės politika atlikti darbus be rangovinių organizacijų arba naujo adresato tipo atsiradimo tikimybė.)
9.	Jei adresatas įmonės atstovas – jis turi turėti darbų vykdytojo teises.	Veiklos taisyklė teisių atžvilgiu.

10.	Jei rangovinė organizacija – būtina užfiksuoti jos pavadinimą, darbų vadovo vardą, pavardę ir kvalifikacinę kategoriją.	Funkcinis reikalavimas. Veiklos taisykle gali būti tik dėl to, kad šioje vietoje dažnos klaidos, o programos tikslas logiškai ir struktūriškai teisingas dokumentas. (Sintaksės veiklos taisyklė)
11.	Nurodyme turi būti nurodytas (skaičiais) brigados narių skaičius, po to išvardinami ir brigados nariai.	Sintaksės veiklos taisyklė.
12.	Jei nurodymas išrašomas įmonės viduje, į brigadą gali būti įtraukti tik įmonės darbuotojai, turintys brigados nario teises.	Veiklos taisyklė.
13.	Jei nurodymas išrašomas rangovui, išvardinami rangovo brigados nariai: vardas, pavardė, kvalifikacinė kategorija arba pareigos.	Sintaksinė veiklos taisyklė.
14.	Nurodymo skiltis „pavedama“ negali likti tuščia. Nurodymas turi nurodyti atliekamus darbus.	Sintaksinė veiklos taisyklė.
15.	Nurodymo darbo pradžios laikas turi būti ankstesnis nei darbo pabaigos laikas.	Ribojimas.
16.	Nurodymo darbų trukmė negali būti didesnė nei 14 dienų. Kitaip turi formuojamas antras nurodymas sekančiam laiko etapui.	Veiklos taisyklė.
17.	Praėjus 30 min. nuo nurodyme užfiksuotos darbų pradžios nurodymas rakinamas (nebeleidžiama jo redaguoti.)	Veiklos taisyklė. Nebus analizuojama apibrėžtos architektūros atveju, kada atliekamas jau suformuoto nurodymo tikrinimas, bet būtų aktuali, naudojant dialogo režimu su veiklos taisyklių saugykla dirbančioje programoje.
18.	Nurodymas, pasirašytas dispečerinės tarnybos, yra užrakintas. Nebent buvo atmestas.	
19.	Egzistuoja 7 darbo vietų paruošimo vykdytojų tipai. Į nurodymą gali įeiti 1 – 6 vykdytojai, arba tik 7. Jei yra keletas, jų eiliškumo tvarka laisvai keičiama.	Veiklos taisyklė, dėl polinkio keistis paruošimo darbų vykdytojų tipams (kartu su jiems galiojančiais apribojimais).

20.	1 tipo nurodymų vykdytojas (Perjungimų lapelis) turi eiti tik pirmoje vietoje.	Veiklos taisyklė.
21.	Antro tipo vykdytoją pasirinkti galima tik tuo atveju jei abu nurodymo asmenys (darbų vadovas ir prižiūrintysis) turi operatyvines teises.	Veiklos taisyklė.
22.	Pasirinki 3-čio tipo vykdytojus galima tik tada, kai darbų vykdytojas ir nors vienas iš brigados narių turi operatyvines teises.	Veiklos taisyklė.
23.	4 tipo vykdytojas, skyriaus dispečerinė, pasirinkimas leidžiamas tik iš fiksuotų įmonės skyrių.	Veiklos taisyklė.
24.	Tiek vykdytojų tiek darbų eiliškumas yra svarbus.	Nefunkcinis reikalavimas.
25.	Esant 1 tipo reikalavimui dėl kranų darbų vadovo, būtina nurodyti pažymėjimo numerį.	Sintaksinė veiklos taisyklė.
26.	Esant 2 tipo reikalavimui, komandos stropuotojas turi būti parinktas iš jau įtrauktų brigados narių.	Veiklos taisyklė.
	...	

Diskutuotinas klausimas, kurios veiklos taisyklės turi būti modeliuojamos atskirame lygmenyje (kurios apskritai pripažįstamos, kaip veiklos taisyklės). Kurias taisykles galima priskirti prie architektūros klausimų, realizuojamų klasių ar duomenų bazės schema, o ką prie linkusios keistis logikos. Visgi naudojantis GUIDE veiklos taisyklių modeliu [1], galima sakyti, kad žodyno struktūra (faktai, atributai ir ryšiai tarp jų) taip pat yra veiklos taisyklės. OBR objektiškumas taip pat reikalauja išbaigto veiklos modelio, vadinasi, norint gauti sprendimą dėl vieno ar kito esybės atributo reikia apibrėžti ir visą esybę.

Lentelėje naudota sąvoka „sintaksinė veiklos taisyklė“ turėtų būti suprantama, kaip taisyklė konkrečios nurodymų įforminimo dalykinės srities atveju, dėl vieno iš tikslų atlikti ne tik loginį bet ir struktūrinį dokumento patikrinimą. Vargu ar reikalavimą užpildyti vieną ar kitą lauką išskirtume

prie veiklos taisyklių kitose dalykinės srityse. Bet kokiu atveju pakartosime teiginį, kad veiklos taisyklių išskyrimas – intuityvus procesas.

5.3. Dalykinės srities sąsaja su GUIDE veiklos taisyklių modeliu

Esminis veiklos taisyklėmis grindžiamų informacinių sistemų darbas – identifikuoti dalykinės srities taisykles ir jas užrašyti. Siekiant geriau suprasti taisyklių skirstymą į tipus, aptartus 2.1 poskyryje, atlikta nurodymų įforminimo dalykinės srities apžvalga taikant GUIDE [1] veiklos taisyklių metamodelį ir jo aprašą.

13. lentelė

Pagrindiniai taisyklių tipai pagal GUIDE[4] ir jų atitikmuo nurodymų įforminimo dalykinėje srityje

Taisyklių rūšis / tipas	**Atitikmuo dalykinėje srityje „Nurodymų įforminimas“.**	**Kaip realizuojama Oracle veiklos taisyklių įrankyje.**
Struktūriniai teiginiai / Sąvokos	Klasės, išskirtos 9 pav. pavaizduotoje klasių diagramoje. Pavyzdžiui, Nurodymas, Darbuotojas. **Bendroji sąvoka** galėtų būti Asmuo (įprasta, kad asmuo turi vardą, pavardę ir asmens kodą). **Veiklos sąvokos** – visos likusios klasės, nes būdingos tik šiai dalykinei veiklos sričiai taip pat yra naudojamos nors vieno fakto apraše.	Sąvokos atsispindės OBR apibrėžimų kortelėje (5 pav.) kaip Java, XML, RL faktai, apribojimai ir kintamieji.
Struktūriniai teiginiai / Faktai	Faktai skirstomi į pirminius ir išvestinius. **Pirminis faktas** - pavyzdžiui nurodymo ID (nurodymo raktas). Nurodymo numeris tampa **išvestinu faktu**, nes nors yra fiksuojamas pačiame nurodyme kaip atributas – jis išskaičiuojamas pagal tam tikrą taisyklę (išvestį). Kitas skirstymas	Veiklos taisyklių faktai OBR modeliuojami kaip sukurtų žodyno faktų atributai. Jei turimas faktas atributas, dažniausiai realizuojamas žodyno fakto savybe (property), turinčia reikšmę nustatančius ir nuskaitančius laukus. Jei faktas asociacija – naudojami žodyno faktų laukai (field). Daugianarė asociacija modeliujama sąrašu. Norint tikrinti

	Faktai atributai (nurodymo data, brigados narių skaičius) atitinka klasių diagramos atributų laukus. **Faktai dalyviai/asociacijos** – atitinka asociacijos ryšius tarp klasių. Pavyzdžiui nurodymą tikrina dispečeris. **Faktai /agregatai** – atitinka agregacijos ryšį tarp klasių. Nurodymas agreguojamas iš brigados narių, paruošimo darbų vykdytojų ir reikalavimų. **Faktas / apibendrinimas –** asmuo apibendrina darbuotoją, tai reiškia, darbuotojas turi tas pačias savybes (atributus), kaip ir asmuo.	asociacijos fakto buvimą, tikriname atitinkamo lauko nustatymą (!= null). Jei turimas asociacijos faktas, per jį galima pasiekti kitą sąvoką (kitą asociacijos pusę). Tiek faktai agregatai tiek apibendrinimas OBR realizuojami objektinių klasių pagalba ir sukuriami sudarant žodyną. (Agregatais laikome tuos faktus, kurie sukuriami sąvokos (klasės) konstruktoriuje.) Norėdami užtikrinti kompoziciją tikrinsime ar ryšį neša abi pusės. Apibendrinimui realizuoti naudojamas klasių paveldėjimas (paruošiant importuojamas java klases).
Veiklos teiginiai / sąlygos, apribojimai leidimai	• Sąlygos veiklos teiginys – Jei nurodymo darbų vadovas yra įmonės darbuotojas, brigadą formuojama iš įmonės brigados narių – įgalinus pirmąją sąlygą, įgalinamas antrosios veikimas ir veiklos taisyklės tikrinamos toliau. • Ribojimas – jei nurodymas patvirtintas dispečerių – jis užrakintas (su juo nebegalimi jokie veiksmai). Sąlyga turi būti griežtai išpildyta. Taisyklių interpretavimo mechanizmui sutikus tokią sąlygą belieka daryti analizuojamo objekto išmetimą iš agendos (atidavimą) nes nėra ką daugiau analizuoti (nebent galimybę atrakinti).	Atitinka veiklos taisyklių sritį OBR įrankyje. Užrašomi IF – THEN šablonu, naudoja žodyno faktus ir jų atributus sąlyginėje dalyje, bei faktų metodus, RL kalbos funkcijas, priskyrimus procedūrinėje. Skirtingi veiklos teiginių potipiai pasiekiami skirtingai organizuojant darbą.

	• Leidimo pavyzdys – tik darbų vadovas gali išrašyti nurodymus.	
Išvestys	• Matematinės išvesties pavyzdys - nurodymo numeris sudaromas pagal tokią struktūrą: <darbų vadovo tabelio numeris> „ - “ <darbų vadovo nurodymų skaičius nuo metų pradžios>. • Loginė išvesti – jei nurodymas turi tikrinimo požymį, jis užrakintas redagavimui.	OBR gali būti realizuotos arba atskirais fakto metodais (jei tai Java faktas), arba RL funkcijomis, gražinančiomis tam tikras reikšmes, arba reguliariosiomis išraiškomis. Bet kokiu atveju išvesties sudarymo metu dažniausiai po THEN dalies naudojamas taisyklės tipas priskirti (assign) (Taisyklių tipai aprašyti 0. skyriuje.)

Pagal GUIDE[1] sąlygos teiginiai išreiškia galimybę (dalinai nulemtą architektūrinių, nekintamos logikos dalių). Tai atspindėta OBR sprendime, kada pagrindiniais faktais (veiklos sąvokomis) tampa importuojamos Java klasės. Jei klasė apibrėžta konkretiems atributams ar ryšiams, daugiau jų atsirasti negali. Tuo tarpu veiklos teiginiai realizuoja apribojimus. Pirmieji apribojimai, kurie turėtų būti fiksuojami veiklos žodyne – atributų privalomumas ir ryšių kardinalumas. Atributų privalomumas lengvai tikrinamas veiklos teiginiais ir yra lengvai keičiama savybė (sudarius taisykles), o ryšių kardinalumas tikriausiai turėtų būti modeliuojamas pertekliškai. Ryšys daug tarp klasių kuriamas sąrašu (List), ryšys vienas – paprastu lauku. Todėl siekiant turėti galimybę valdyti kardinalumą, verta Java klasių santykį apsibrėžti kardinalumu daug su daug, ir reikiamus apribojimus nurodyti tik OBR įrankyje. Kita galimybė XPath klasės naudojimas objektų tarpusavio santykiams modeliuoti. Tačiau ji šiame darbe nebuvo nagrinėjama.

Nors pavyksta atrasti visus teorinio GUIDE modelio taisyklių tipus OBR įrankyje, jis nėra modeliuojamas pagal GUIDE ar kurią kitą, analizėje aptartą metodiką. Reikėtų pastebėti, kad OBR taisyklių šablonai artimesni Barbaros von Halle metodikai, trumpai aptartai 2.1 poskyryje. Oracle Business Rules įrankyje egzistuoja faktai ir labai lanksčios IF - THEN struktūros, kurių galimybes praplečia objektinės Java kalbos (arba RL) naudojimas sistemos pagrindui.

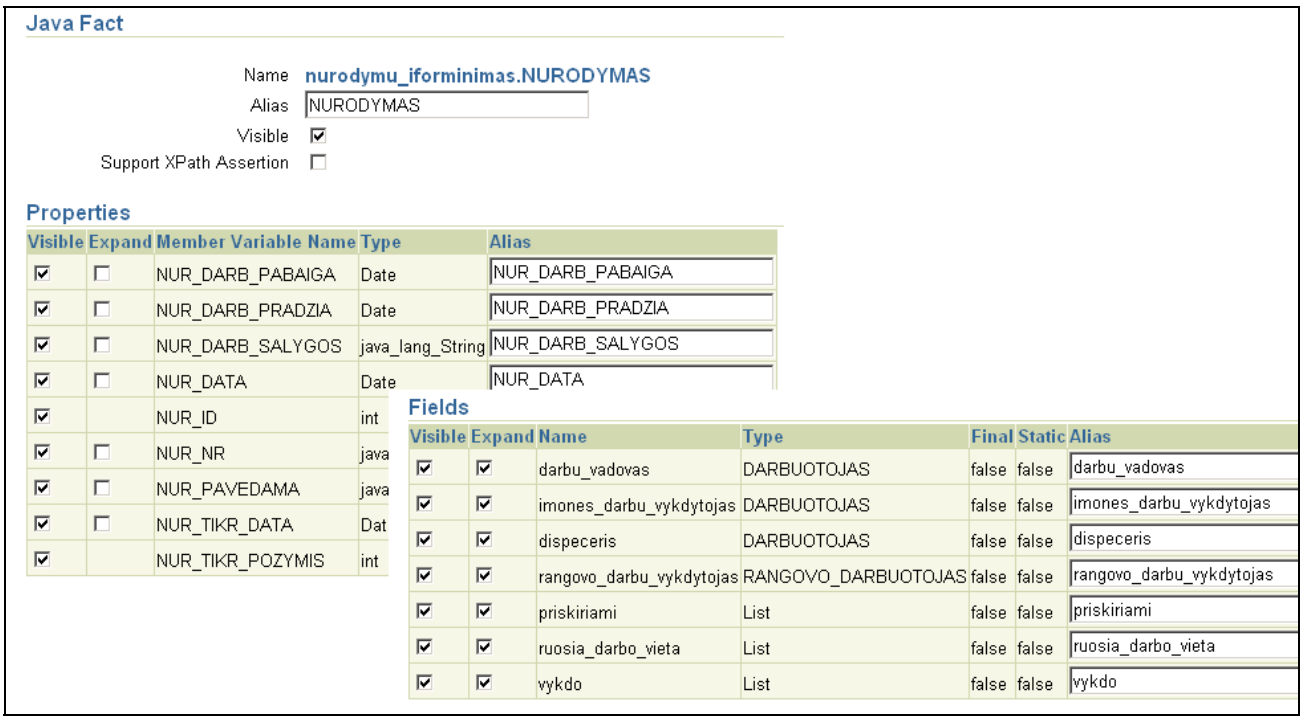
5.4. Taisyklių automatizavimas. Realizacijos veikimo patikrinimas

5.4.1. Žodyno paruošimas

Dalykinės srities Java faktai buvo sukurti importuojant .jar paketą, sugeneruotą iš klasių diagramos, su tam tikrais pakeitimais. Laikytasi principo, kad konceptų klasės neturi metodų, tik konstruktorius ir savybes (Java Beans), ir ryšius su kitomis klasėmis, modeliuojamus public tipo laukais. Ryšių modeliavimui OBR taip pat galima naudoti XPath java klasę, palaikančia objektų medžio struktūros padavimą taisyklių rinkiniui. Šio darbo metu tokia galimybė nebuvo tyrinėta. Toliau pateikiamas importuotas Java faktų sąrašas, bei pagrindinio fakto „Nurodymas“ savybių ir ryšių vaizdavimas Rule Author žodyne.

☐	java.util.List	List
☐	java.util.Arrays	Arrays
☐	java.util.ArrayList	ArrayList
☐	nurodymu_iforminimas.RANGOVINE_ORGANIZACIJA	RANGOVINE_ORGANIZACIJA
☐	nurodymu_iforminimas.RANGOVO_DARBUOTOJAS	RANGOVO_DARBUOTOJAS
☐	nurodymu_iforminimas.NURODYMAS	NURODYMAS
☐	nurodymu_iforminimas.DARBUOTOJAS	DARBUOTOJAS
☐	nurodymu_iforminimas.BRIGADOS_NARYS	BRIGADOS_NARYS
☐	nurodymu_iforminimas.ASMUO	ASMUO
☐	nurodymu_iforminimas.PARUOSIMO_DARBU_VYKDYTOJAS	PARUOSIMO_DARBU_VYKDYTOJAS
☐	nurodymu_iforminimas.VYKDYTOJU_TIPAS	VYKDYTOJU_TIPAS
☐	nurodymu_iforminimas.PARUOS_DARBAS	PARUOS_DARBAS
☐	nurodymu_iforminimas.NURODYMO_REIKALAVIMAS	NURODYMO_REIKALAVIMAS
☐	nurodymu_iforminimas.REIKALAVIMU_TIPAS	REIKALAVIMU_TIPAS
☐	nurodymu_iforminimas.TEISE	TEISE
☐	nurodymu_iforminimas.Main	Main
☐	nurodymu_iforminimas.DARB_TEISE	DARB_TEISE

10 pav. Nurodymo įforminimų Java faktų sąrašas



11 pav. Klasės „Nurodymas“ fakto savybės ir laukai (ryšiai su kitomis klasėmis)

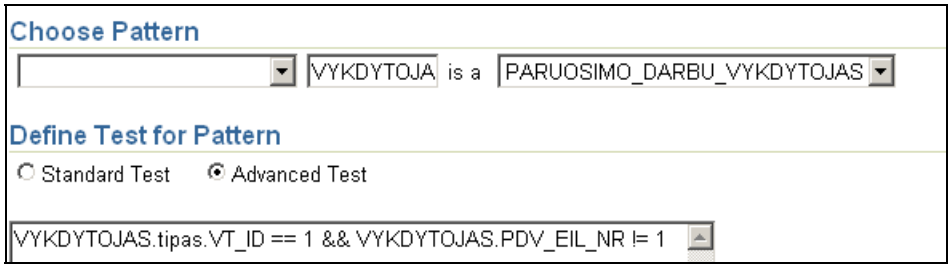
5.4.2. Taisyklių automatizavimas

Sudarant taisykles laikytasi principų, kad jos atominės, nepriklausomos. Nesvarbi jų užrašymo tvarka [2]. Todėl kurtas vienas taisyklių rinkinys. Darbas organizuojamas sekančiu principu:

- Kaip faktas paduodamas vienas objektas (klasės nurodymas), bet per jo viešus atributus (ryšius) galime pasiekti visą kitą informaciją;
- Taisyklės suformuotos ieškoti veiklos taisyklių pažeidimų. Radus klaidą, išvedamas pranešimas, bet nurodymas tikrinamas iki galo.
- Dalis taisyklių tikrina informaciją, pasiekiamą ryšių pagalba. Kita dalis sukuria ir registruoja naują objektą Agendoje (taip fakto tikrinimo organizavimas atiduodamas taisyklių procesoriui).

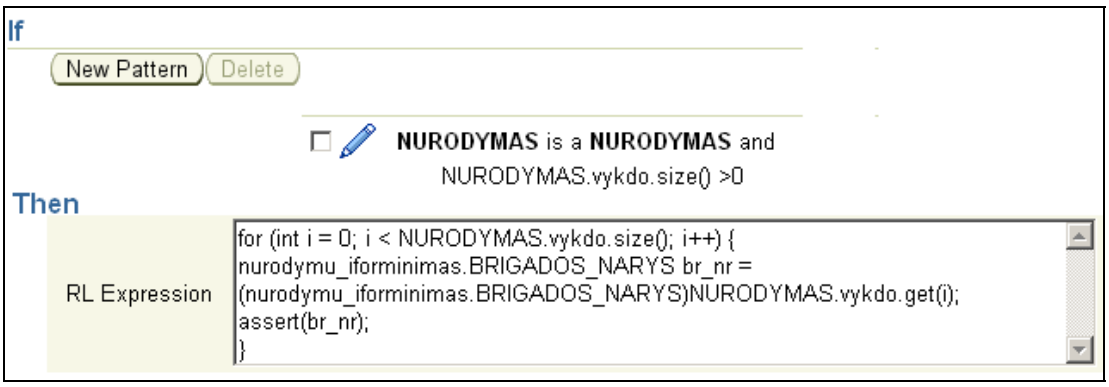
Toliau pateikiami sudėtingesnių taisyklių pavyzdžiai.

12 pav. Nurodymų įforminimo taisyklių rinkinys bei taisyklė pirmajam reikalavimui, darbuotojai privalo turėti kategorijas.

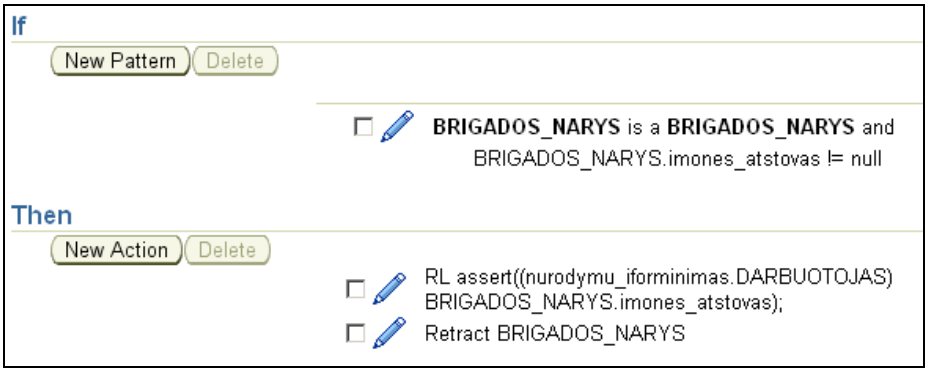


13 pav. Tikrinamas 9 reikalavimas – perjungimų lapelis pirmoje vietoje. Objektas vykdytojo_tipas pasiekiamas per ryšį.

Tarkime reikalavimas, kad visi nurodyme dalyvaujantys įmonės nariai turėtų kategoriją. Norint, patikrinti kategoriją visiems brigados nariams, kurie yra įmonės darbuotojai vykdoma trijų taisyklių grandinėlė:



14 pav. Taisyklė Brigada. Jei nurodymas turi nurodytą brigadą (ryšys vykdo), kiekvieną per sąrašą pasiekiamą brigados narį kuriame kaip objektą ir paduodame (Assert) taisyklių procesoriaus tolimesniam apdorojimui.

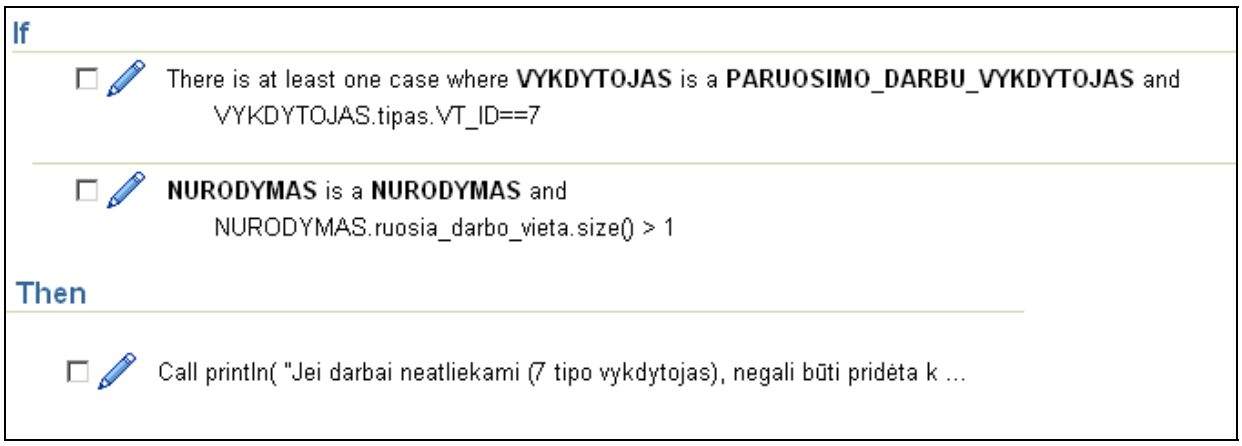


15 pav. Taisyklė BrigadosNariaiImones. Aptikus brigados narį, jei jis įmonės atstovas, taisyklių procesoriui paduodamas klasės darbuotojas objektas, priskirtas kaip įmonės brigados narys.

Toliau suveikia taisyklė Kategorijos, pavaizduota 12 paveiksle , kuri patikrina, ar reikiamas brigados narys turi reikiamą kategoriją. Toks naujų objektų registravimas į Agendą taisyklių vykdymui naudingas tada, kai:

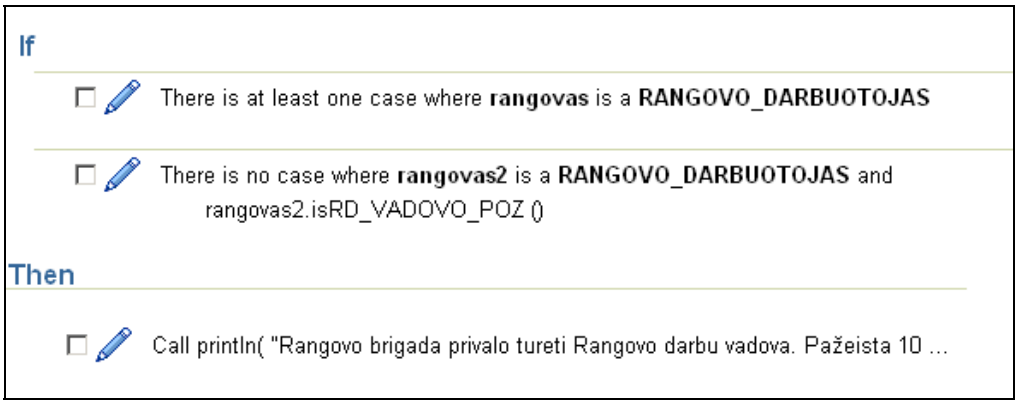
- Tikrinamo tipo klasei priskirta bent pora taisyklių;
- Tikrinamo tipo klasė plačiai naudojama skirtingose vietose (mūsų atveju klasė darbuotojas siejasi su nurodymu kaip brigados narys, darbų vadovas, darbų vykdytojas ar dispečeris);
- Susidaro situacijos, kada kitas būdais negalima pasiekti norimos objekto (sąrašų apdorojimas). Nors tokiu atveju būtinas naujo objekto sukūrimas, bet nėra būtinas jo registravimas agendoje;
- Norima daugiau skaidrumo taisyklių užrašyme. (Kiekvienam objektui po taisyklę, o ne topologijų nagrinėjimas);
- Siekiama paprastesnės taisyklių struktūros, mažiau RL kodo blokų, logikos formavimo šablonais.

Kita pavyzdinė situacija, kada naudojamas ne konkretaus fakto tikrinimas, o nagrinėjama, ar toks faktas apskritai registruotas Agendoje. Tokio tipo šablonai patogūs tikrinant faktų tarpusavio sąryšius – ar faktas kažką (kitą faktą, atributą) turi arba neturi.



16 pav.Taisyklė Vykdytojas7. Jei nurodyme nurodyta, kad paruošimo darbų vykdytojas yra 7 tipo (Darbai neatliekami), bet pačiam nurodymui yra priskirta daugiau paruošimo darbų vykdytojų – generuojama klaida.

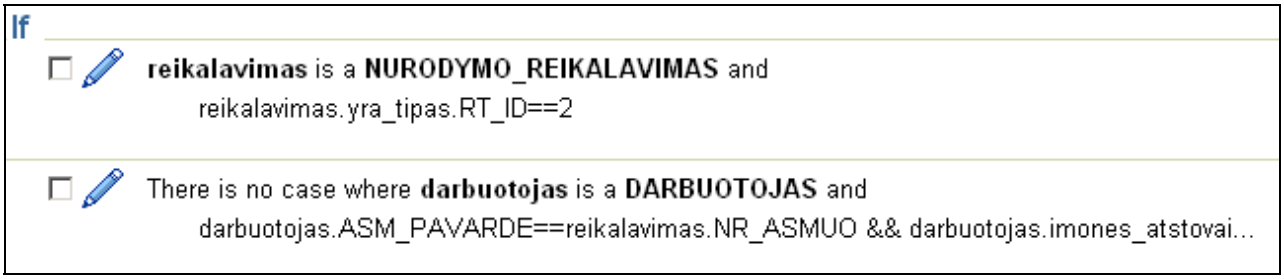
Dar vienas darbo su objektų grupe pavyzdys, automatizuojame 10 taisyklę „Jei rangovinė organizacija – būtina užfiksuoti jos pavadinimą, darbų vadovo vardą, pavardę ir kvalifikacinę kategoriją“. Tiksliname taisyklę – jei darbus atlieka rangovo brigada, turi būti užfiksuotas rangovo darbų vadovas (atsakingas asmuo). Analogiškai įmonės brigados narių tikrinimui bus įvykdoma per 3 taisyklių grandinėlę: randami nurodymui priskirti brigados nariai, jei brigados nariai rangovo, į taisyklių procesorių registruojami nauji faktai, trečiąja taisykle tikriname, ar tarp narių yra darbų vadovas (17 pav.).



17 pav. Tikrinama taisyklė, kad rangovo brigada privalo turėti rangovo darbų vadovą

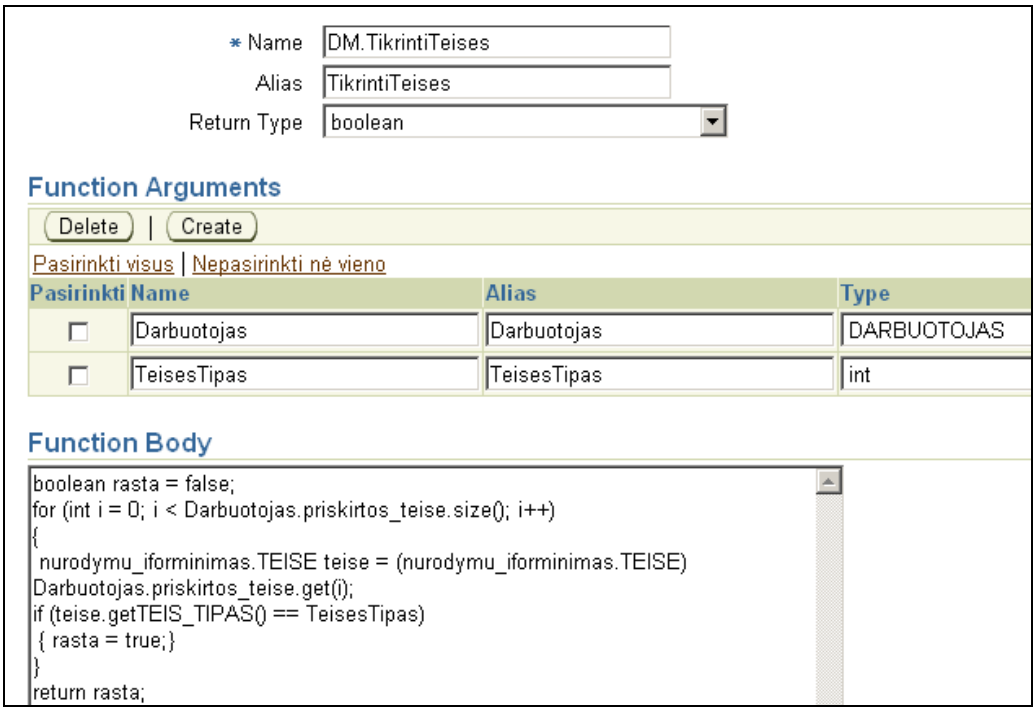
Kaip buvo minėta 3.2 skyriuje, viena iš žodyne aprašomų RL funkcijų paskirčių yra pakartotinis panaudojimas. Sekanti situacija pateikia pavyzdį, kada patogu naudoti RL funkcijas. Probleminė sritis – klasių „darbuotojas“ – „teisė“ ryšys daug su daug (9 pav.). Tarkime, norint, kad darbuotojas būtų nurodymų darbų vadovu, jis privalo turėti darbų vadovo teisę, bet gali turėti ir daug kitų. Faktų tarpusavio ryšiai ir atvejai turi / neturi šiek tiek aukščiau buvo sprendžiami kuriant naujo tipo objektus, paduodant juos į Agendą ir analizuojant darbinėje atmintyje esančių objektų savybes

su santykiais „Yra nors vienas toks atvejis“, „Nėra tokio atvejo“. Ten aprašomi atvejai, kada analizuojami objektai turėjo tiesioginį ryšį su nurodymu (vienas nurodymas vienoje sesijoje, nėra pašalinių faktų). Tuo tarpu ir darbuotojų ir teisių registravimas į Agendą rezultate duotų daug darbuotojų, kiekvienas jų gali turėti daug teisių. Tokioje situacijoje būtina susieti taisyklės sąlygoje kartu naudojamus objektus.



18 pav. Darbinėje atmintyje yra daug darbuotojų ir daug reikalavimų. Norint atrinkti konkretaus reikalavimo darbuotoją, siejami abiejų faktų atributai. Taisyklės Reikalavimas2 sąlyginė dalis

Tą patį uždavinį galima spręsti kitu metodu – faktų sąrašų apdorojimą perduoti RL kodo blokams. Duotuoju atveju su teisėmis tai geresnis sprendimas (darbinė atmintis nėra perkraunama dideliu skaičiumi toliau nenagrinėjamų faktų).



19 pav. RL funkcija, naudojama konkretaus fakto sąraše paieškai (Ar darbuotojas turi ieškomą teisę)

5.5. Tyrimo išvados

Atlikus tyrimą, galima apibendrinti naudotus metodus standartinių veiklos taisyklių situacijų sprendimui.

Jei taisyklių įrankyje naudojami Java faktai, faktų savybes patogu modeliuoti Java Beans tipo savybėmis (privatus laukas, turintis reikšmę nuskaitantį ir nustatantį metodus), ryšius – viešais atributais, sąrašų pavidalu (kardinalumo ribojimus galima uždėti veiklos taisyklių įrankyje).

OBR įrankis nemato fakto atributų (metodų) giliau nei per 2 – 3 lygius. Tokiu atveju galima išsiversti naudojant pažangų sąlygos formavimo režimą ir RL tipo blokus procedūrinėje taisyklės dalyje. Jei norimas pasiekti atributas yra kitoje ryšio pusėje esantis faktas, naudinga tokį objektą „ištraukti“, t. y. objektą registruoti Agendoje savarankiškam tikrinimui veiklos taisyklių atžvilgiu.

Atvejai „nėra tokio atvejo“ arba „yra nors vienas tokio tipo atvejis“ naudojami darbui su faktų grupėmis ir patogūs tikrinant faktų tarpusavio sąryšius, ar faktas kažką (kitą faktą, atributą) turi arba neturi.

Veiklos žodyne apibrėžiami kintamieji, globalūs sesijos metu, naudingi suminiams rezultatams kaupti arba požymiams laikyti. Funkcijos patogios pakartotinio panaudojimo metu, taip pat dažnai sprendžia uždaviniu, kurių neišeina realizuoti kitose dalyse.

Agendoje susidarantys konfliktiniai rinkiniai sprendžiami naudojant taisyklių prioritetus, jei jų nėra, naudojama vėliausiai į Agendą pakliuvusi aktyvacija.

Prie šio tyrimo reikėtų paminėti, kad įrankis nesprendžia ciklų susidarymo ar resursų paskirstymo problemų. Jei sudarytos dvi taisyklės, taip modifikuojančios faktą, kad paeiliui kviečia viena kitos vykdymą, įrankis užstringa (begalinio ciklo susidarymas). Jei susidaro situacija, kad taisyklės perpildo darbinę atmintį, taip pat generuojama nepalaikoma klaida.

6. ĮRANKIO NAŠUMO TYRIMAS

Tyrimas atliekamas norint palyginti taisyklėmis grindžiamų IS ir paprastų IS našumą. Norint supaprastinti atliekamo tyrimo vertinimą, visą apibrėžto uždavinio realizaciją atliksime OBR, o po to Java technologijomis. Tai ne visiškai atitinka veikos taisyklių paskirtį ir veikimo architektūrą. Paprastai veiklos taisyklės suprantamos kaip servisai, padedantys priimti sprendimus, o šiuo atveju bus naudojamos ne tik veiklos (strategijos) palaikymui ir valdymui, bet ir realizacijai. Antra vertus, tai puiki proga išanalizuoti veiklos taisyklių technines galimybes.

6.1. Dalykinė sritis „Telekomunikacinių paketų apdorojimas“

Šiame tyrime iškeltas tikslas realizuoti tarpinės sistemos (telekomunikacinių paketų apdorojimo) prototipą. Tarkime, turime telekomunikacinių paslaugų tiekėją, siūlantį fiksuoto ryšio paslaugas (tiek duomenų, tiek balso perdavimas vykdomas telefono linijomis). Tiekėjas apskaičiuoja suteiktų paslaugų kainą, apdorodamas techninių įrenginių (maršrutizatorių) saugomas duomenų ataskaitas. Failuose fiksuojami techniniai duomenys, iš kurių tiekėjas išgauna sau reikalingą informaciją, susijusią su klientu.

14. lentelė

Techniniame faile saugomų duomenų aprašas

Atributo pavadinimas	Privalomas	Aprašas
CDR_ID	taip	Unikalus įrašo ID, unikaliai identifikuojantis maršrutizatorių.
SUBSCRIBER_ID	taip	Unikalus vartotojo ID. Turi griežtai nustatyta formą.
SERVICE	taip	Teikiama paslauga, balsas „voice“ arba duomenys „data“
BILLABLE	taip	Identifikuoja, ar paslauga apmokestinama. (Neapmokestinamos techninės, derinimo paslaugos)
SUBSCRIBER_IP	ne	Tiekėjo IP, turi atitikti IPv4 standartą.
NAS_IP_ADDRESS	ne	Vartotojo IP, turi atitikti IPv4 standartą
SESSION_START_TIME	taip	Sesijos starto laikas sekančiu formatu ‘YYYYMMDDHH24MISS’
SESSION_END_TIME	ne	Sesijos pabaigos laikas. Jei nėra – ryšys nutrūko arba buvo perduotas kitam įrenginiui.
START_TIME_PARTIA	taip	Duomenų perdavimo startas laikas.

L		
STOP_TIME_PARTIAL	taip	Duomenų perdavimo pabaigos laikas.
DURATION	taip	Sesijos trukmė sekundėmis.
UPLINK_VOLUME	taip	Išsiųstas duomenų kiekis.
DOWNLINK_VOLUME	taip	Parsiųstų duomenų kiekis.
TERMINATION_CAUSE	ne	Sesijos nutraukimo kodas.
SESSION_ID	taip	Sesijos ID
UL_MAX	ne	Maksimalus išsiunčiamų duomenų greitis Kbits/s.
DL_MAX	ne	Maksimalus parsiųstų duomenų greitis Kbits/s.
INTERIM	taip	Nusako, ar tai paskutinis seanso paketas. (Ilga sesija skaidoma į porą paketų, ir po to gali būti klijuojama).

Dalykinė sritis atitinka reikalavimus kurti veiklos taisyklėmis paremtą IS, kadangi:

- dažna veiklos politikos kaita (didelis duomenų šaltinių skaičius, skirtingais parametrais pateikiamos informacijos apdorojimas, su naujomis technologijomis nuolat kinta ir veiklos logikos parametrai);
- greito atsako į pasikeitimus poreikis (tarpinės sistemos duomenų apdorojimą atlieka iškarto, todėl labai greitai aptinkami linijų perkrovimo ar gedimo atvejai, į kuriuos turi būti reaguojama);
- yra poreikis palaikyti keletą veiklos politikos rinkinių vienu metu (apmokestinimo planai, duomenų kiekių skaičiavimo variantai skirtingų tipų vartotojams).

Realizuojamą uždavinį galima apibrėžti sekančia seka:

- Maršrutizatorius generuoja duomenų failą.
- Failas apdorojamas, išskiriant paketus.
- Paketai turi tenkinti veiklos reikalavimus, patvirtinti, kad rasti visi 14 lentelėje privalomi laukai, ir jie užrašyti tinkamu formatu. Jei paketas turi blogo formato duomenų, arba pažymėtas kaip sugadintas, tokie paketai nufiltruojami į blogų sąrašą.
- Paketai perskirstomi vartotojams. Vartoto balso ir duomenų paketai saugomi atskirai.
- Vartotojai apmokestinami pagal galiojančias veiklos taisykles ir paketų duomenis.
- Generuojamos ataskaitos.
- Apdorojama statistika.

Kaip pradiniai duomenys naudojamas maršrutizatoriaus generuojamų išeities duomenų failas. Faile pateikiamų duomenų formatas pavaizduotas 20 paveiksle, apimtis 14 000 eilučių. Taip pat žinoma duomenų paketo eilutės gramatinio nagrinėjimo (parse) schema.

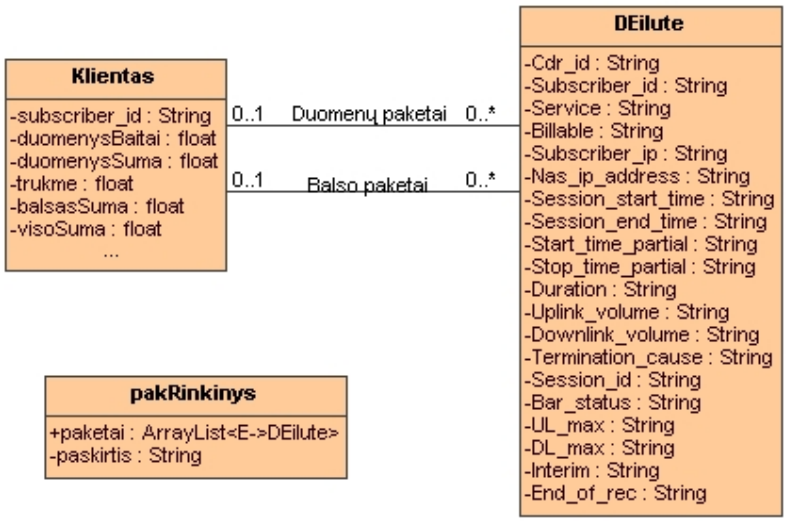
```
023403662245419341 98RETCS0391207632datay192.22.9.119236.155.248.19620060601
100000200606011057202006060110000020060601105720000034400000000000005469938
200000000000045931408001602e5f2aadbcbe990000003440000 20480

023403662245419341 99RETCS0551706894datay14.229.0.16556.58.216.1692006060110
000020060601100000200606011049290000296900000000000011602777000000000000787
99439000907c6069d571f5950000003440000 20481
```

20 pav. Pradinių duomenų fragmentas. Dvi vartotojo duomenų paketų persiuntimo eilutės

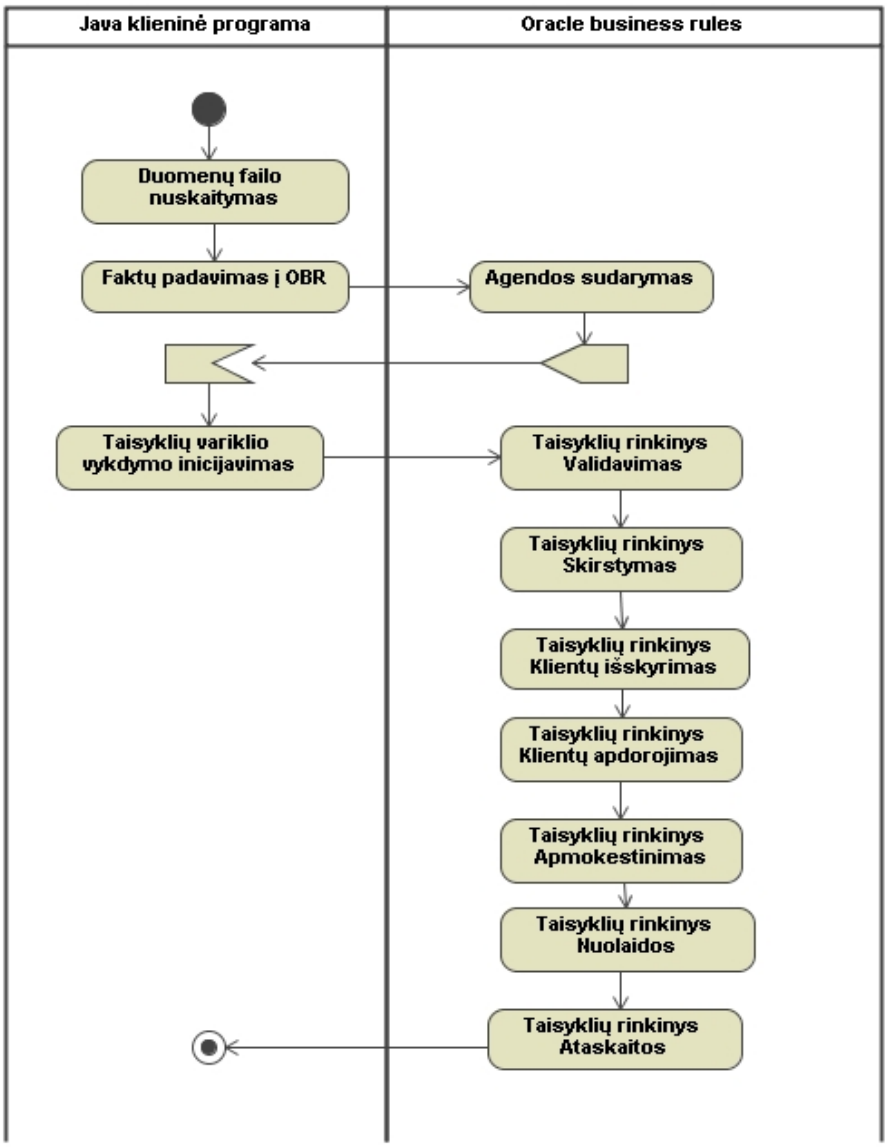
Siekiant išbandyti įrankio savybes nuo paketų nuskaitymo iš failo, visa kita logika buvo realizuota OBR įrankyje. Ne visi etapai atitinka veiklos taisykles. Pavyzdžiui, ataskaitų generavimas neturi nieko bendro su IF - THEN struktūros naudojimu, tačiau įtrauktas į realizaciją siekiant išbaigto iškelto uždavinio realizavimo vienoje technologijoje.

6.2. Dalykinės srities realizavimas

Kliento programa nuskaito tekstinį duomenų failą ir visus 14 000 eilučių paduoda į taisyklių procesorių. Tik tada pradedamas faktų apdorojimas. Kiekvienam iš duomenų apdorojimo etapų buvo kuriamas atskiras taisyklių rinkinys. Norint garantuoti logišką duomenų apdorojimą, jie paeiliui sudėti į taisyklių rinkinių steką. Taisyklių steko eiliškumas pavaizduotas veiklos diagramoje.



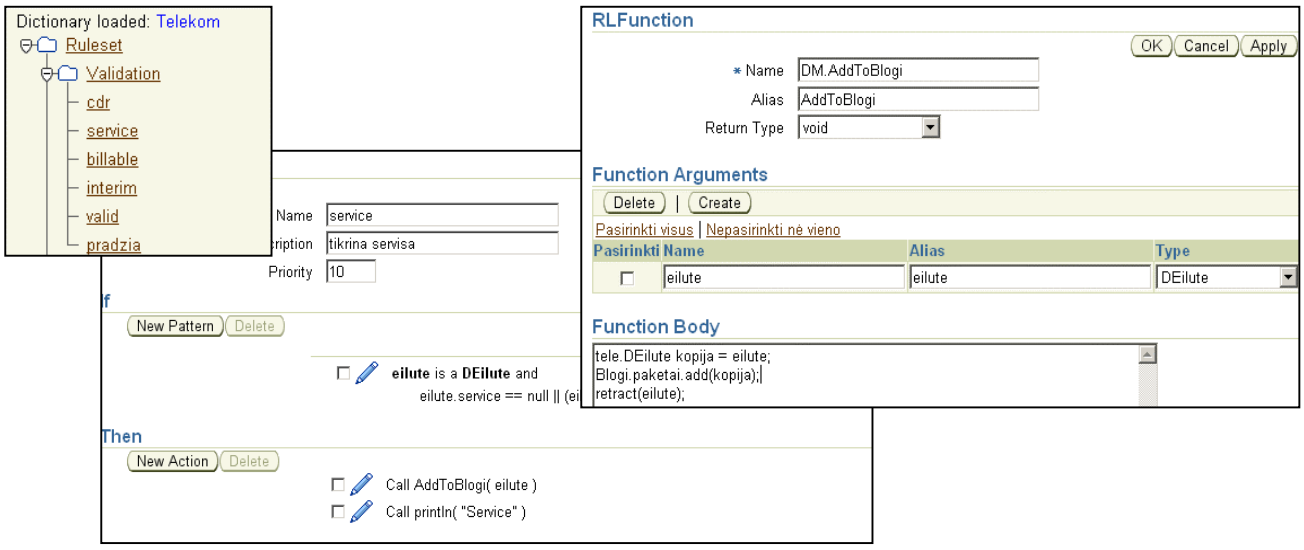
21 pav. Dalykinės srities klasių diagrama. Ryšiai klasėse modeliuojami sąrašais.



22 pav. Telekomunikacinių paketų apdorojimo veiklos diagrama.

Veiklos taisyklių vykdymas nepriklauso nuo jų užrašymo eiliškumo (viename taisyklių rinkinyje). Tačiau organizuojant taisyklių rinkinių eiliškumą galima kontroliuoti vykdymo logiką. Toliau paeiliui pateikiamas trumpas kiekvieno rinkinio aprašas.

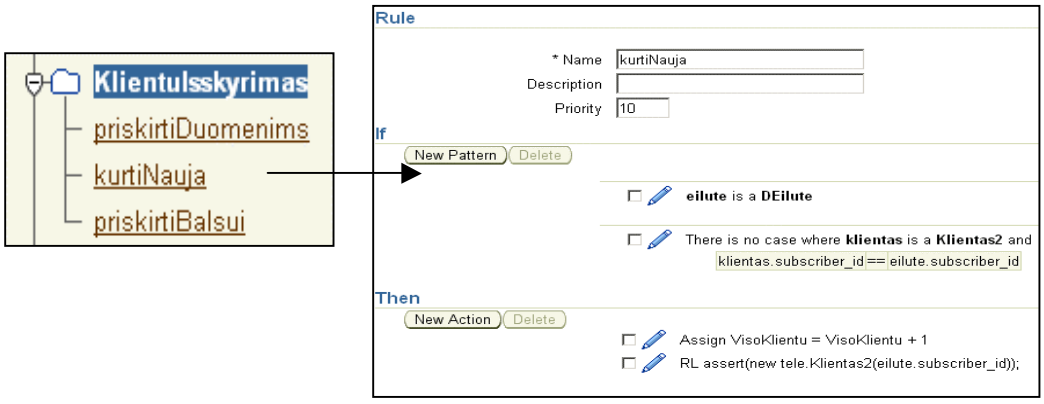
Taisyklių rinkinys "Validavimas" tikrina, ar paduotas faktas atitinka paketo struktūrą. Paketo struktūra aprašyta 14 lentelėje. Tarkime, tam tikro lauko duomenys turi atitikti IPv4 adreso struktūrą, arba būtina nustatyti, kokio tipo, balso ar duomenų, paketai yra apdorojami (atributo service reikšmė gali būti lygi „data“ arba „voice“) Jei ne, atidedama į blogų paketų sąrašą ir jo aktyvacija šalinama iš Agendos.



23 pav. Taisyklių rinkinys Validavimas. Tikrinamas serviso tipas, jei taisyklė tenkinama – paketas įdedamas į blogų sąrašą ir šalinamas iš tolesnio apdorojimo.

Taisyklių rinkinys Skirstymas − paketas statistikai. Atskirai sumuojamas bendras balso ir bendras duomenų paketų skaičių, perduotas maršrutizatoriumi.

Taisyklių rinkinys „KlientuIsskyrimas“ - sukuriamas faktas Klientas ir jam priskiriami atitinkamai jo balso ir duomenų paketai. Esami 14 000 faktų persiskirsto atitinkamai į skirtingų klientų skaičių. Žemiau pavaizduota taisyklė, skirta klientų išskyrimui. Jei sutiktas paketas, kurio kliento ID dar nėra priskirta nei vienam klientui – kuriamas naujas faktas. Jei ne – surandamas reikiamas Kliento faktas ir paketas priskiriamas į jo duomenų ar balso sąrašus.



24 pav. Taisyklių rinkinys Klientų išskyrimas, taisyklė kurtiNauja sukuria naują Klientas2 tipo faktą.

Taisyklių rinkinys KlientuApdorojimas – apdoroja ir apmokestina kliento duomenis. Prioritetai leidžia viename taisyklių rinkinyje pirma atlikti paketų sumavimą ir perskaičiavimą, po to apmokestinimą. Pagal veiklos taisykles, apmokestinimo taisyklės turėtų naudoti arba pritaikymo

vartotojui reikšmes, arba atskiras funkcijas arba taisykles greitam finansinės informacijos pakeitimo galimybių valdymui.

25 pav. Taisyklių rinkinį Klientų apdorojimas sudaro 4 taisyklės.

Taisyklių rinkinys „Nuolaidos“ - veiklos logika. Naudinga turėti keletą tokio tipo rinkinių skirtingoms situacijoms.

26 pav. Taisyklių rinkinio Nuolaidos taisyklė, taikanti 5 procentų nuolaidą balso paketams, jei apmokestinama suma viršija 10 (Lt).

Taisyklių rinkinys „Ataskaitos“ nevisiškai atspindi veiklos taisykles (proceso realizacija), bet leidžia pademonstruoti įrankio galimybes. Naudojant RL funkciją realizuotas rezultatų failo generavimas. Rezultatas gali būti bendras arba kiekvienam klientui atskiras. Rezultato pavyzdys:

Klientas: RETCS0498705482

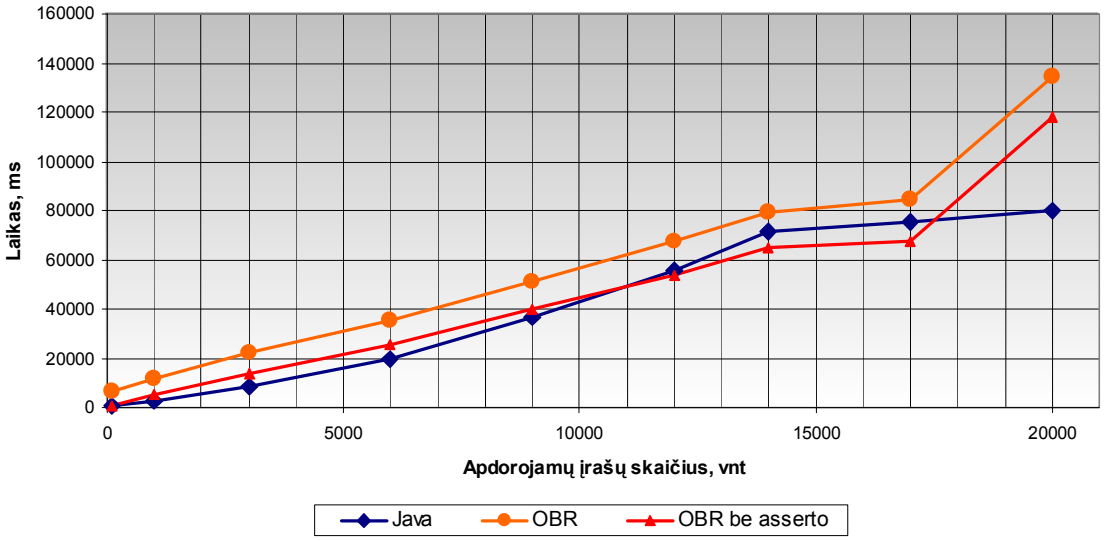
Perduota duomenų: 43.247673 MB | Apmokestinta: 4.0574438 Lt.

Pokalbiu trukmė: 0.0 s. | Pokalbių kaina: 0.0Lt.

Tokiu būdu buvo sukurta pilna uždavinio realizacija nuo pradinių duomenų (maršrutizatoriaus duomenų eilučių) iki reikiamo tipo informacijos (klientų apmokestinimo ataskaitų). Tarpiniame lygmenyje sukuriant duomenų faktą Klientas, kurį galima naudoti tolesniam apdorojimui arba saugojimui duomenų bazėje.

6.3. Realizacijų taikant taisyklių procesorių OBR ir Java programavimo kalbą našumo palyginimas

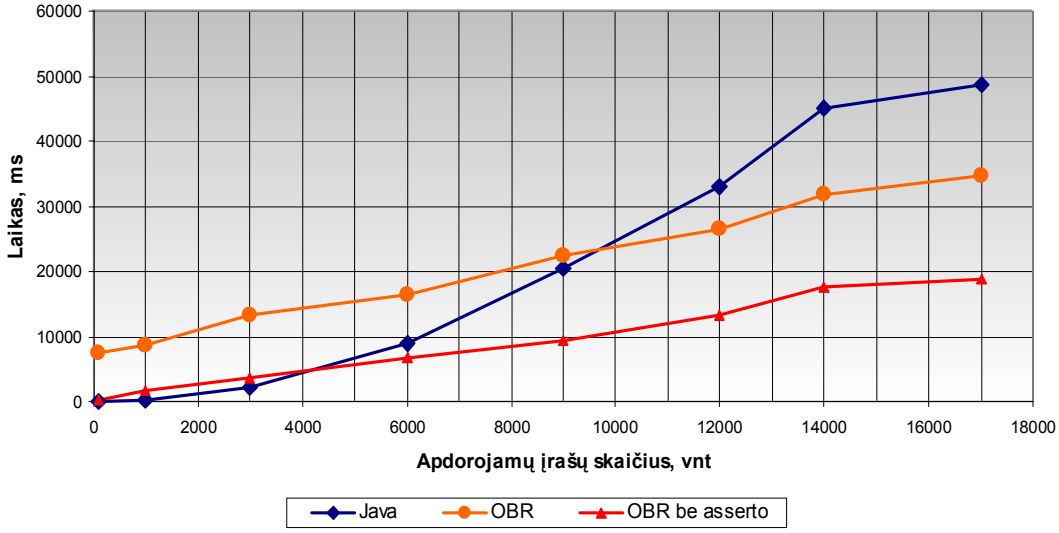
Stengiantis laikytis tų pačių duomenų apdorojimo principų, šis uždavinys buvo realizuotas Java programavimo kalba. Toliau buvo atliekamas abiejų realizacijų vykdymo laiko priklausomybės nuo apdorojamų duomenų kiekio tyrimas, kuris pateikiamas grafiškai.



27 pav. Viso mediavimo uždavinio sprendimo laiko priklausomybės nuo apdorojamų duomenų kiekio grafikas.

Fiksuojant laiko skaičiavimus buvo atliekama po 5 matavimus kiekvienam duomenų kiekiui. Oracle veiklos taisyklių įrankio tyrimui išskirtos dvi sąvokos. Bendras OBR laikas – tai laikas, kurį sugaišo įrankis nuo sesijos pradžios iki pabaigos. OBR be „assert“ būtų laikas, kuris skaičiuojamas vien faktams apdoroti, neįskaičiuojant, kiek laiko užtrunka faktų padavimas taisyklių procesoriui (į tą laiką įeina Agendos sudarymas). Sunku pasakyti, kuris grafikas tiksliau apibrėžia įrankio veikimą.

Kaip matyti grafike, vykdymo laikas tolygiai tiesiškai auga iki 14 000 duomenų paketų. Nuo 14 000 iki 17 000 turime santykinai mažesnes laiko sąnaudas (nes turimas duomenų failas yra 14000 eilučių dydžio, todėl yra paduodamas apdoroti antrą kartą, ir tam tikri duomenų apdorojimo etapai iškrenta). Tačiau nuo 17 000 Java toliau dirba stabiliai, tuo tarpu OBR sąnaudos stipriai išauga. Įvyksta darbinės atminties užpildymas, mat įrankio pagreitinimui naudojamas Rete algoritmas yra labai imlus atminties resursams. Su turima eksperimento įrangos konfigūracija (4 lentelė) nepavyko atlikti skaičiavimų su daugiau nei 22 000 eilučių.



28 pav. Dalies tarpinio uždavinio, kada naudojami paketų tikrinimo, statistikos skaičiavimo ir klientų išskyrimo taisyklių rinkiniai, sprendimo laiko priklausomybės nuo apdorojamų duomenų kiekio grafikas

Atsisakius dalies taisyklių rinkinių (ypač tokių, kurie nebūdingi veiklos taisyklių sričiai, kaip kad ataskaitų formavimas) tyrimo rezultatai tampa naudingesni OBR įrankiui. Čia galima įžvelgti Rete algoritmo veikimą, kurį panagrinėsime sekančiame skyriuje.

6.4. Rete algoritmo analizė.

Kaip buvo pastebėta 6.3 poskyryje, su dideliu duomenų kiekiu OBR susidoroti nepajėgia (įrankis ir nėra tam skirtas). Tyrimo metu tokia situacija naudojama išryškinti Rete algoritmo įtaką.

Rete algoritmas sukurtas daktaro Charles L. Forgy ir šiuo metu naudojamas beveik visuose veiklos taisyklių procesoriuose (viena ar kita atmaina). Jis skirtas pagreitinti taisyklių procesorių, naudojančių tiesioginio išvedimo mechanizmą, veikimą, stengiantis sumažinti skaičių pakeitimų, reikalingų perskaičiuoti konfliktinius rinkinius po kiekvienos taisyklės vykdymo. 4.1 poskyryje buvo aptarta OBR taisyklių procesoriaus veikimo principas. Rete naudojamas atliekant Agendoje esančių taisyklių-faktų aktyvacijų peržiūrą, po kiekvieno taisyklės suvykdymo. Esminis Rete algoritmo minusas - reikalauja didelio kiekio darbinės atminties. Toliau bus išdėstytas klasikinio Rete veikimo principas [12].

Rete remiasi 2 esminiais pastebėjimais:

- Taisyklės įvykdymas dažniausiai pakeičia tik pora faktų. Ir tik keletas taisyklių yra susijusios (turi reaguoti) į šių faktų pasikeitimą;
- Struktūriniai panašumai – ta pati sąlyga dažnai naudojama daugelio taisyklių sąlyginėje dalyje.

Algoritmo realizacija naudoja šakninį orientuotąjį grafą be ciklų. Grafo viršūnės, išskyrus šaknį, atspindi sąlygas, kelias, nuo šaknies lapų link - taisyklės sąlygines dalis (kairioji taisyklės pusė). Grafą sudaro 3 tipų viršūnės:

- vieno įėjimo šakninė viršūnė – į ją paduodami faktai jų registravimo ar šalinimo metu;
- vieno įėjimo šablonų viršūnės – aprašo faktų tipą (kiekviena veiklos taisyklė nurodo, kokiam faktui yra taikoma), 2 lygmuo po šaknies.
- keletos įėjimų sąjungos viršūnių – atitinka taisyklės sąlygos užrašymą.

Kiekviena viršūnė saugo informaciją apie faktus, tenkinančius kelio, nuo šaknies iki einamosios viršūnės, sąlygas. Taigi kiekvienai viršūnei išskiriama atminties struktūra, sauganti informaciją, apie taisyklę tenkinančius faktus. OBR tai atitinka aktyvaciją Agendoje.

Nuo taisyklių ir faktų skaičiaus priklauso atminties, reikalingos Rete grafo sudarymui, kiekis. Klasikiniame Rete variante teigiama, kad tam įtakos turi ir sąlyginės taisyklės dalies užrašymo būdo. Tačiau darbe eksperimentu šito patvirtinti nepavyko. Atliktame tyrime, kada naudojami 5 taisyklių rinkiniai (20 taisyklių) Rete atminties trūkumas atsiranda dirbant su 17 000 pradinių faktų (taisyklių vykdymo eigoje buvo sukurta dar 10 000 kito tipo faktų). Atminties resursai yra techninis parametras, priklausantis tiek nuo uždavinio sudėtingumo, tiek nuo darbo techninės aplinkos, tačiau galvojant apie praktinį uždavinių sprendimą reikėtų į jį atkreipti dėmesį.

7. ĮRANKIO INTEGRAVIMO GALIMYBĖS.

Šiame skyriuje bus aptariamos įrankio integravimo galimybės. Klientinė programa Oracle taisyklių procesorių pasiekia naudodama JSR -94 „Java Rules Engine API“, programa turi turėti nuorodas į oracle.rules.sdk ir orcalce.rules.rl.ruleSession bibliotekas. Bet kokiu atveju, jei nėra galimybės klientinę dalį realizuoti Java programa, galima naudoti tinklo paslaugas (web servisus) su WSDL.

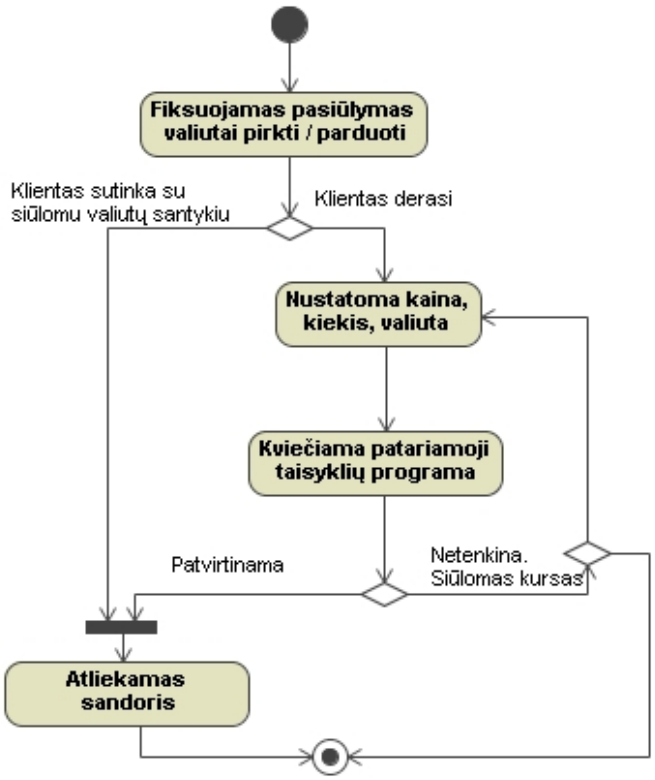
Antra vertus, jei veiklos taisyklių procesorius kaip faktus naudoja Java klases, ir nedaro apribojimų metodų naudojimui ir matomumui (išskyrus matomumą RuleAuthor viduje), kodėl nepasinaudoti Java galimybėmis pačio įrankio viduje. 6 skyriaus tyrime, vykdant taisyklių rinkinį „Ataskaitos“, buvo naudojamas duomenų rašymas į failus tiesiogiai iš OBR. Sekantis pavyzdys realizuoja tinklo paslaugų iškvietimą iš taisyklių procesoriaus.

7.1. Dalykinė sritis „Valiutos kursai.“

Organizacija „Valiutos keitykla“ turi valiutų keityklų tinklą visoje Lietuvoje. Įmonės politika – turėti centralizuotą visų padalinių veiklos valdymą, vieną valiutų dilerį, nustatantį valiutų kursą ir apibrėžiantį veiklos politiką visuose padaliniuose. Problemos:

- klientams, kurių sumos viršija nustatytą ribą, valiutos kursas nustatomas individualiai. Kuo didesnė suma, tuo palankesnis kursas;
- individualus valiutos kursas gali svyruoti priklausomai nuo įmonės apyvartos, laukiamo valiutų santykio ar paklausos pokyčio artimiausiu metu, turimų atsargų ir panašiai.

Nustatyti valiutos kursą kiekvienai dienai ir kiekvienam padaliniui nėra sudėtinga. Tačiau individualūs atvejai reikalauja konsultacijos su profesionaliu valiutos dileriu. Aprašyta situacija galbūt nėra verta ekspertinės sistemos, tačiau pakankama veiklos taisyklių panaudojimui. Toliau pateikiama siūloma veiklos diagrama konceptualiame lygmenyje diagrama.



29 pav. Valiutos keityklų darbo principas, naudojantis centralizuota sandorių valdymo sistema.

7.2. Integravimo realizacija

Pagrindinis uždavinio tikslas - tinklo paslaugų pasiekiamumas iš veiklos taisyklių aplinkos. Buvo panaudoti Lietuvos bankas oficialūs užsienio valiutų santykiai. Sukurta tinko paslaugas kviečianti biblioteka, ji importuojama į taisyklių rinkinį. Nors grafinėje įrankio sąsajoje neatvaizduojamos visos pagalbinės bibliotekos, įrankyje turi būti registruojami visi nuorodomis susiję komponentai. Gauname tinklo paslaugos klientą, pasiekiamą iš veiklos taisyklių srities.

☐	lt.lb.webservices.ExchangeRates.ExchangeRates
☑	lt.lb.webservices.ExchangeRates.ExchangeRatesSoap
☐	lt.lb.webservices.ExchangeRates.GetListOfCurrenciesResponseGetListOfCurrenciesResult
☐	lt.lb.webservices.ExchangeRates.GetExchangeRatesByDateResponseGetExchangeRatesByDateResult
☐	lt.lb.webservices.ExchangeRates.GetExchangeRatesByCurrencyResponseGetExchangeRatesByCurrencyResult
☐	lt.lb.webservices.ExchangeRates.ExchangeRatesLocator

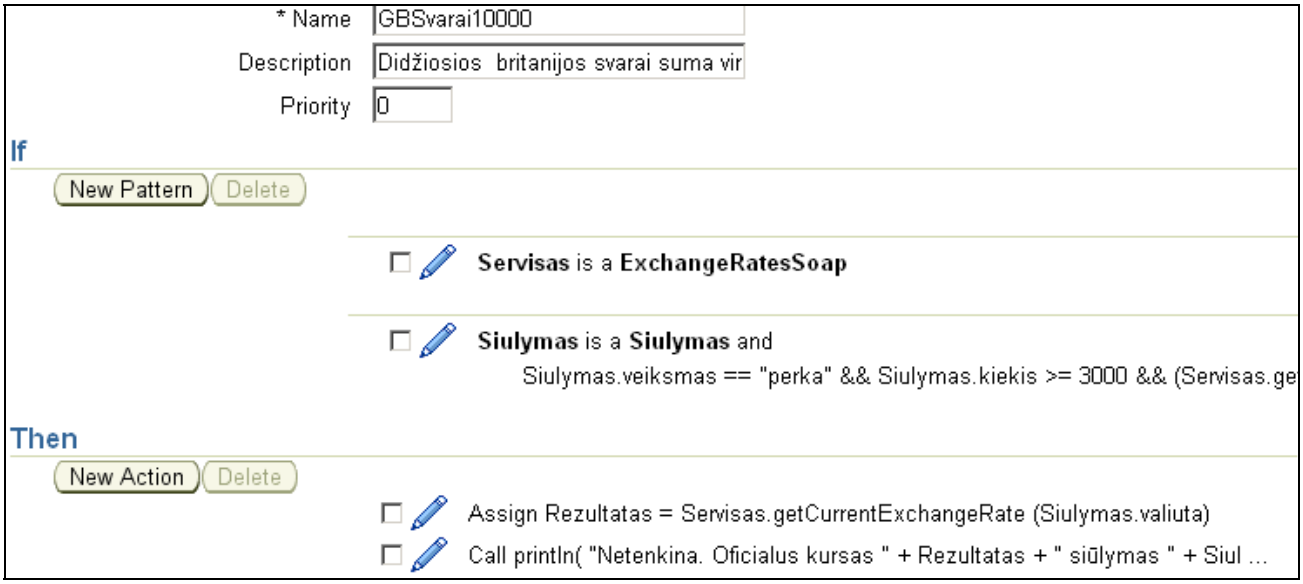
30 pav. Lietuvos banko oficialiai teikiamas tinklo paslaugas naudojančio kliento klasės (dalis), integruotos veiklos taisyklių uždavinyje. Pažymėtas faktas naudojamas tinklo paslaugai pasiekti.

Methods

Visible	Expand	Method Name	Argument Type	Return Type
☑	☐	getListOfCurrencies		GetListOfCurrenciesResponseGetListOfCurrenciesR
☑	☐	getCurrentExchangeRate	java_lang_String	BigDecimal
☑	☐	getExchangeRate	java_lang_String, java_lang_String	BigDecimal
☑	☐	getExchangeRatesByDate_XmlString	java_lang_String	java_lang_String
☑	☐	getExchangeRatesByCurrency_XmlString	java_lang_String, java_lang_String, java_lang_String	java_lang_String
☑	☐	getExchangeRatesByDate	java_lang_String	GetExchangeRatesByDateResponseGetExchangeR
☑	☐	getExchangeRatesXmlSchema		java_lang_String
☑	☐	getExchangeRatesByCurrency	java_lang_String, java_lang_String, java_lang_String	GetExchangeRatesByCurrencyResponseGetExchar

31 pav. Metodai valiutos kursų santykiams gauti.

Viena iš pavyzdinių taisyklių vaizduojama paveiksle. Atsakymas būtų neigiamas, jei perkama valiuta - Didžiosios Britanijos Svarai, kiekis > nei 3000, bet siūloma kliento kaina tik 0,02(Lt) didesnė nei oficialus valiutos kursas. Siūlymo duomenys gaunami iš kliento, išoriniai duomenys, valiutų kursai gaunami kviečiant tinklo paslaugas, veiklos taisyklių įrankis naudojasi informacija ir priima sprendimą. (Pastaba: paprastumo dėlei į įrankį jau paduodamas tinklo paslaugas kviečiantis objektas. Įrankyje tik panaudojami jo metodai.)



32 pav. Taisyklė, atmetanti pirkimo pasiūlymą, jeigu skirtumas tarp siūlomo kurso oficialaus LB patvirtinto kurso mažesnis nei 0,02 Lt.

Kadangi pavyko gana nesunkiai panaudoti Java technologijų funkcionalumą įrankio viduje, galima teigti, kad tokios galimybės nėra ribojamos. O ir pačio įrankio pasiekiamumas tinklo servisų pagalba, tiek Javos, tiek XML standarto faktų palaikymas užtikrina neblogą įrankio integralumą su kitomis sistemomis.

IŠVADOS

1. Oracle veiklos taisyklių įrankis buvo pasirinktas tyrimui dėl veiklos taisyklių požiūrio taikymo aktualumo Oracle programinės įrangos vartotojams, kuriems perspektyva taikyti tokio tipo įrankį yra patraukli, tačiau įrankio galimybės neaiškios dėl dokumentacijos trūkumo.

2. Nustačius vėliau pasirodžiusiais informaciniais šaltiniais paremtą Oracle veiklos taisyklių įrankio vertinimą panašių sistemų atžvilgiu ir atlikus vartotojo sąsajos analizę, galima teigti, kad kol kas platus įrankio taikymas didelių įmonių veiklos taisyklių valdymui dėl savo silpno išvystymo yra abejotinas, o mažose įmonėse šio įrankio taikymas netenka prasmės.

3. Atlikta veiklos taisyklių procesoriaus veikimo analizė automobilių nuomos dalykinės srities pagrindu rodo, kad laukiamas sprendimas priklauso ne vien nuo taisyklių logikos ir apdorojamų faktų, bet ir nuo taisyklių vykdymo organizavimo (eilės, prioritetų ir t. t.).

4. Atlikta įrankio galimybių įvairioms taisyklėms apdoroti analizė dalykinės srities „Nurodymų įforminimas“ pagrindu parodė, kad Oracle veiklos taisyklių įrankis nesiremia jokia fundamentalia veiklos taisyklių metodologija (Guide, Ross, von Halle), bet greičiau ribotais objektinio programavimo principais.

5. Atliktas įrankio našumo tyrimas telekomunikacinių paketų apdorojimo dalykinės srities pagrindu, lyginant su to paties uždavinio realizacija Java technologija, parodė, kad didelių našumo skirtumų nėra. Veiklos taisyklių įrankis praranda dalį našumo apibrėžiant taisyklių sąrašą, taip pat netinka dideliems duomenų (ne žinių) kiekiams apdoroti.

6. Java technologijų taikymas stipriai padidina įrankio galimybes. Telekomunikacinių paketų apdorojimo tyrimas parodė, kad veiklos taisyklių įrankiui galima patikėti visą uždavinio vykdymą, taip pat kviesti išorines tinklo paslaugas. Šios galimybės prasilenkia su veiklos taisyklių paskirtimi, bet yra patrauklios programuotojams.

7. Kadangi pastaruoju metu visi didieji programinės įrangos gamintojai papildo savo technologijas veiklos taisyklių procesoriais, galima tikėtis, kad Oracle veiklos taisyklių įrankis bus patobulintas arba pakeistas kitu.

8. LITERATŪRA

1. Business Rules Project Guide (Final Raport), 1997 Spalis – 62 p.
2. The Business Rules Group. Business Rules Manifest. 2003 Lapkritis – 2 p. Prieiga per interneta: <http://www.businessrulesgroup.org/brmanifesto/BRManifestLithuanian%28v4.0%29.pdf>
3. BizTalk Server Business Rules Framework. Iš *MicrosoftTechNet* [interaktyvus]. 2005 gegužė [žiūrėta 2007-01-11]. Prieiga per internetą: <http://www.microsoft.com/technet/prodtechnol/biztalk/biztalk2004/planning/business-rules-framework-overview.mspx>
4. Butleris R., Kapočius K. "Struktūrizuotų veiklos taisyklių saugyklos architektūra" Prieiga per internetą: <http://www.leidykla.vu.lt/inetleid/inf-mok/47/str6.html>
5. Kapočius K. „Veiklos taisyklių struktūrizavimo modeliai ir jų taikymas kuriant informacijos sistemas": Daktaro disertacijos santrauka: technologijos mokslai, informatikos inžinerija (07T) / Kęstutis Kapočius; Kauno technologijos universitetas. Kaunas, 2006. 34p.
6. Date C. J. Twelve Rules for Business Rules 2000 sausis [žiūrėta 2007-01-11]- 8 p. Prieiga per interneta: <http://www.alphora.com/42Rules-Date.pdf>
7. Van Raalte T., Yu Hwang K. Oracle® Business Rules User's Guide, 10g(10.1.3.1.0) B28965-04, 2006 Rugsėjis. [Žiūrėta 2006-10-23] Prieiga: Oracle Application Server List of Books
8. Van Raalte T. Oracle® Business Rules Language Reference 10g(10.1.3.1.0) B28964-01, 2006 Rugsėjis – 128 p. [Žiūrėta 2006-10-23] Prieiga: Oracle Application Server List of Books
9. Gartner blogas. [Žiūrėta 2008-12-11]. Prieiga per interneta: <http://blogs.gartner.com/jim_sinur/2008/11/18/ibm-microsoft-oracle-and-sap-have-bought-business-rule-technology-whats-up-with-that/>
10. Ginter M., Oracle® Application Server Installation Guide10g Release 3 (10.1.3.1.0) for Microsoft Windows B34044-01, 2006 Spalis - . [Žiūrėta 2006-12-10] Prieiga: Oracle Application Server List of Books.
11. Business Rules Community organizacijos interneto svetainė. [Žiūrėta 2008.10.10]. Prieiga per internetą: <http://www.brcommunity.com/ >
12. Pateikėjas ingargiola@cis.temple.edu, IS587: The RETE Algotithm. *Temple University College of science and Technology* pedagoginė medžiaga. [Žiūrėta 2008.10.10] Prieiga per internetą: <http://www.cis.temple.edu/~ingargio/cis587/readings/rete.html>

9. SANTRUMPŲ ŽODYNAS

IS – Informacinė sistema.

OBR – Oracle Business Rules. Oracle įrankis veiklos taisyklių palaikymui.

RL – Rule Language, OBR viduje naudojama aukšto lygio į Java panaši programavimo kalba.

SOA – servine Oriented Architecture (į servisus orientuota architektūra).

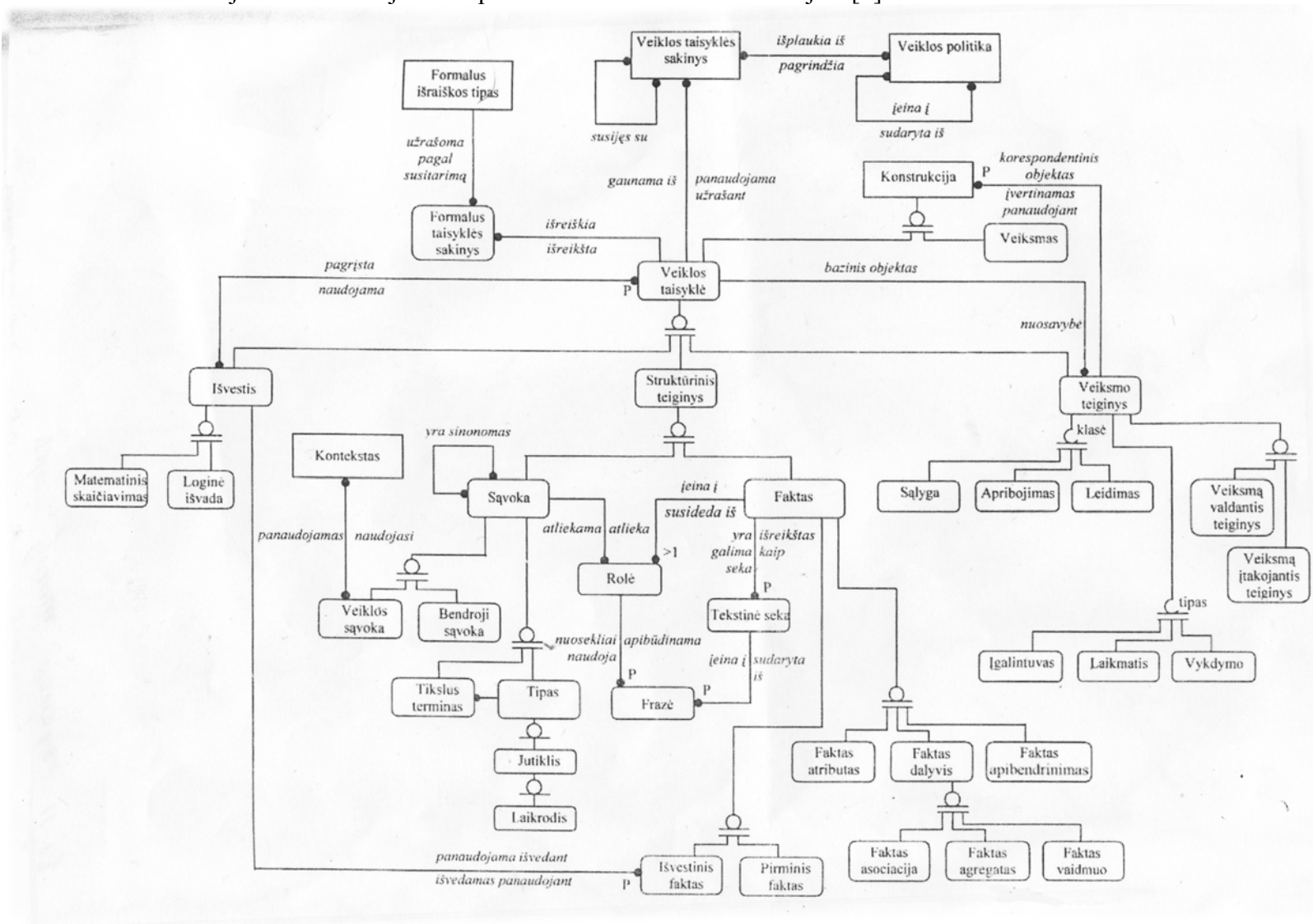
Vt – veiklos taisyklė.

WSDL – web servine definition language.

10.PRIEDAI

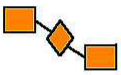
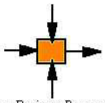
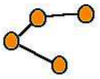
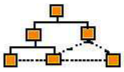
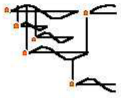
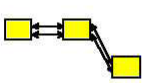
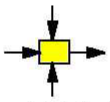
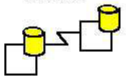
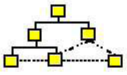
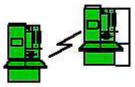
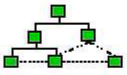
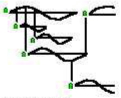
10.1. Priedas nr. 1. GUIDE veiklos taisyklių modelis lietuvių kalba.

Naudojama informacija K. Kapočiaus daktaro darbo disertacijos. [5]



10.2. Priedas nr. 2. Zachman Framework

ENTERPRISE ARCHITECTURE - A FRAMEWORK ™

	DATA *What*	FUNCTION *How*	NETWORK *Where*	PEOPLE *Who*	TIME *When*	MOTIVATION *Why*	
SCOPE (CONTEXTUAL) *Planner*	List of Things Important to the Business  ENTITY = Class of Business Thing	List of Processes the Business Performs  Function = Class of Business Process	List of Locations in which the Business Operates  Node = Major Business Location	List of Organizations Important to the Business  People = Major Organizations	List of Events Significant to the Business  Time = Major Business Event	List of Business Goals/Strat  Ends/Means=Major Bus. Goal/ Critical Success Factor	SCOPE (CONTEXTUAL) *Planner*
ENTERPRISE MODEL (CONCEPTUAL) *Owner*	e.g. Semantic Model  Ent = Business Entity Reln = Business Relationship	e.g. Business Process Model  Proc. = Business Process I/O = Business Resources	e.g. Logistics Network  Node = Business Location Link = Business Linkage	e.g. Work Flow Model  People = Organization Unit Work = Work Product	e.g. Master Schedule  Time = Business Event Cycle = Business Cycle	e.g. Business Plan  End = Business Objective Means = Business Strategy	ENTERPRISE MODEL (CONCEPTUAL) *Owner*
SYSTEM MODEL (LOGICAL) *Designer*	e.g. Logical Data Model  Ent = Data Entity Reln = Data Relationship	e.g. "Application Architecture"  Proc. = Application Function I/O = User Views	e.g. "Distributed System Architecture"  Node = I/S Function (Processor, Storage, etc) Link = Line Characteristics	e.g. Human Interface Architecture  People = Role Work = Deliverable	e.g. Processing Structure  Time = System Event Cycle = Processing Cycle	e.g., Business Rule Model  End = Structural Assertion Means = Action Assertion	SYSTEM MODEL (LOGICAL) *Designer*
TECHNOLOGY MODEL (PHYSICAL) *Builder*	e.g. Physical Data Model  Ent = Segment/Table/etc. Reln = Pointer/Key/etc.	e.g. "System Design"  Proc.= Computer Function I/O = Screen/Device Formats	e.g. "System Architecture"  Node = Hardware/System Software Link = Line Specifications	e.g. Presentation Architecture  People = User Work = Screen Format	e.g. Control Structure  Time = Execute Cycle = Component Cycle	e.g. Rule Design  End = Condition Means = Action	TECHNOLOGY CONSTRAINED MODEL (PHYSICAL) *Builder*
DETAILED REPRESEN-TATIONS (OUT-OF-CONTEXT) *Sub-Contractor*	e.g. Data Definition  Ent = Field Reln = Address	e.g. "Program"  Proc = Language Stmt I/O = Control Block	e.g. "Network Architecture"  Node = Addresses Link = Protocols	e.g. Security Architecture  People = Identity Work = Job	e.g. Timing Definition  Time = Interrupt Cycle = Machine Cycle	e.g. Rule Specification  End = Sub-condition Means = Step	DETAILED REPRESEN-TATIONS (OUT-OF-CONTEXT) *Sub-Contractor*
FUNCTIONING ENTERPRISE	e.g. DATA	e.g. FUNCTION	e.g. NETWORK	e.g. ORGANIZATION	e.g. SCHEDULE	e.g. STRATEGY	FUNCTIONING ENTERPRISE

Zachman Institute for Framework Advancement - (810) 231-0531

10.3. Priedas nr. 3. Veiklos taisyklių procesorių vystymosi istorija