Kauno technologijos universitetas

Informatikos fakultetas

„Data Vault“ metrikų saugyklos sudarymo metodika

Baigiamasis magistro projektas

Karolis Butkus

Projekto autorius

Doc. Lina Čeponienė

Vadovė

Kaunas, 2024

Kauno technologijos universitetas

Informatikos fakultetas

„Data Vault“ metrikų saugyklos sudarymo metodika

Baigiamasis magistro projektas

Veiklos skaitmeninimas ir sistemų architektūros (6211BX009)

Karolis Butkus

Projekto autorius

Doc. Lina Čeponienė

Vadovė

Doc. Linas Ablonskis

Recenzentas

Kaunas, 2024

Kauno technologijos universitetas

Informatikos fakultetas

Karolis Butkus

„Data Vault“ metrikų saugyklos sudarymo metodika

Akademinio sąžiningumo deklaracija

Patvirtinu, kad:

1. baigiamąjį projektą parengiau savarankiškai ir sąžiningai, nepažeisdama(s) kitų asmenų autoriaus ar kitų teisių, laikydamasi(s) Lietuvos Respublikos autorių teisių ir gretutinių teisių įstatymo nuostatų, Kauno technologijos universiteto (toliau – Universitetas) intelektinės nuosavybės valdymo ir perdavimo nuostatų bei Universiteto akademinės etikos kodekse nustatytų etikos reikalavimų;

2. baigiamajame projekte visi pateikti duomenys ir tyrimų rezultatai yra teisingi ir gauti teisėtai, nei viena šio projekto dalis nėra plagijuota nuo jokių spausdintinių ar elektroninių šaltinių, visos baigiamojo projekto tekste pateiktos citatos ir nuorodos yra nurodytos literatūros sąraše;

3. įstatymų nenumatytų piniginių sumų už baigiamąjį projektą ar jo dalis niekam nesu mokėjęs (-usi);

4. suprantu, kad išaiškėjus nesąžiningumo ar kitų asmenų teisių pažeidimo faktui, man bus taikomos akademinės nuobaudos pagal Universitete galiojančią tvarką ir būsiu pašalinta(s) iš Universiteto, o baigiamasis projektas gali būti pateiktas Akademinės etikos ir procedūrų kontrolieriaus tarnybai nagrinėjant galimą akademinės etikos pažeidimą.

Karolis Butkus

Patvirtinta elektroniniu būdu

Butkus, Karolis. „Data Vault“ metrikų saugyklos sudarymo metodika. Magistro projektas / vadovė doc. dr. Lina Čeponienė; Kauno technologijos universitetas, Informatikos fakultetas.

Studijų kryptis ir sritis (studijų krypčių grupė): Informacijos sistemos, Informatikos mokslai.

Reikšminiai žodžiai: metrikų saugykla, *Data Vault 2.0*, duomenų saugykla, kompiuterinių resursų metrikos, *ETL*.

Kaunas, 2024. 99 p.

Santrauka

Šiame darbe aprašoma metrikų saugyklos sudarymo metodika, skirta *Data Vault 2.0* duomenų saugykloms. Šiuo metu *Data Vault 2.0* duomenų saugyklose kompiuterinių resursų metrikų informacija nėra kaupiama, tačiau ši informacija yra labai aktuali, siekiant spręsti duomenų saugyklos neefektyvaus veikimo situacijas. Ši informacija galėtų būti panaudojama duomenų saugyklos kontrolei, siekiant sistemą pagreitinti, sumažinti sunaudojamų resursų kiekį ar sumažinti sistemos veikimo kaštus. Šiai informacijai saugoti *Data Vault 2.0* architektūra apibrėžia metrikų saugyklos komponentą, tačiau nėra detalizuojama, kaip šis komponentas turėtų būti realizuotas ir kokie metodai turėtų būti taikomi. Taigi, šio darbo tikslas yra palengvinti kompiuterinių resursų metrikų stebėjimą ir kontrolę *Data Vault 2.0* duomenų saugykloje, pasiūlant metrikų saugyklos sudarymo metodiką.

Šiame darbe pateikiama metrikų saugyklos sudarymo metodika, aprašanti visą metrikų saugyklos sudarymo procesą nuo metrikų duomenų šaltinių pridėjimo iki metrikų duomenų atvaizdavimo. Metodika sudaryta atsižvelgiant į dabartinių *Data Vault* duomenų saugyklų sudarymo eigą, siekiant šį procesą kuo labiau integruoti į jau egzistuojančius procesus. Taip pat pateikiamas realizuotas įrankis, kuris automatizuoja metrikų saugyklos sudarymo žingsnius. Įrankis sukurtas naudojant *SpringBoot* ir *React* internetinius karkasus. Realizuoto įrankio pagalba naudotojai gali sumodeliuoti metrikų saugyklą, priskirti kategorijas pagal pateiktą metrikų taksonomiją, taip pat gali sugeneruoti *DDL* ir *ETL* kodą, skirtą sukurti duomenų saugyklą ir ją užpildyti metrikų duomenimis. Siekiant įvertinti pasiūlytos metodikos ir realizuoto įrankio pritaikomumą, buvo įvykdytas eksperimentas, kurio metu sukurta metrikų saugykla, skirta saugoti įmonės serverių resursų duomenis, ir įvykdyta apklausa, kurios metu apklausti su *Data Vault* duomenų saugyklomis dirbantys specialistai. Sukurtoje metrikų saugykloje išsaugoti apie 13 mln. įrašų ir sukurtos vizualizacijos 7 skirtingoms metrikoms. Apklausoje surinkti respondentų atsakymai daugiausiai teigia, kad pasiūlyta metodika ir realizuotas įrankis gali paskatinti naudotojus kurti *Data Vault* metrikų saugyklas metrikų stebėjimo uždaviniams spręsti.

Butkus, Karolis. Metrics Vault Development Methodology for Data Vault. Master's Final Degree Project / supervisor assoc. prof. dr. Lina Čeponienė; Faculty of Informatics, Kaunas University of Technology.

Study field and area (study field group): Information Systems, Computing.

Keywords: Metrics Vault, Data Vault 2.0, data warehouse, computer resource metrics, ETL.

Kaunas, 2024. 99 p.

Summary

This thesis proposes a Metric Vault development methodology for Data Vault 2.0 data warehouses. Currently, Data Vault 2.0 data warehouses do not collect computer resource metrics, however, this information is very relevant when trying to solve data warehouse performance problems. This information could be used for data warehouse management to speed up the system, reduce resource usage, or reduce operational costs. To store such information Data Vault 2.0 architecture proposes a new component called Metric Vault, however, it does not provide guidelines on how it should be implemented. Thus, the main goal of this thesis is to facilitate the monitoring and analysis of computer resource metrics by proposing a Metric Vault development methodology.

This thesis presents the Metric Vault development methodology, which details the whole Metric Vault development process from adding metric data sources to creating metric visualizations. The methodology has been developed by analyzing current Data Vault formation practices so that it could be integrated into current processes. A new Metric Vault development tool has also been created to automate the Metric Vault creation flow. The tool allows users to create their Metric Vault model, categorize the metrics, and generate DDL and ETL code, which creates a data warehouse structure and loads the metric data. An experiment has been conducted to evaluate the applicability of the proposed methodology and implemented tool. During the experiment a new Metric Vault was created to store the company’s server data, also a survey was published to collect the responses from Data Vault data warehouse specialists. The newly created Metric Vault stored about 13 million records and had 7 visualizations to illustrate different metrics. The majority of responses from the survey stated that the proposed methodology and the implemented tool can facilitate the creation of Metric Vaults for Data Vault data warehouses to help with metric monitoring tasks.

Turinys

Lentelių sąrašas

Paveikslų sąrašas

Santrumpų ir terminų sąrašas

Santrumpos:

ETL – duomenų iškėlimas, apdorojimas ir išsaugojimas (angl. *Extract Transform Load*);

SQL – reliacinių duomenų bazių užklausų ir modifikacijų kalba (angl. *Structured Query Language*);

DDL – duomenų struktūrų aprašymo kalba (angl. *Data Definition Language*);

Terminai:

Duomenų saugykla (angl. *data warehouse*) – tai įvairių didelių duomenų apdorojimo ir saugojimo sistema skirta duomenų analizei. Duomenų saugyklos daugiausiai naudojamos dideliems verslo duomenims apdoroti, pagal kuriuos būtų priimami svarbūs verslo valdymo sprendimai.

Veiklos operacinio lygio sistema (angl. *operational system*) – programų sistema, skirta veiklą vykdančios organizacijos kasdienėms veiklos operacijoms vykdyti.

Verslo raktai (angl. *business key*) – *Data Vault 2.0* duomenų modelyje naudojamas atributų stereotipas. Šis stereotipas skirtas pažymėti atributus ar atributų kombinacijas naudojamas veikloje identifikuoti specifinį objektą.

Centras (angl. *hub*) – *Data Vault 2.0* duomenų modelio komponentas. Šis komponentas skirtas aprašyti duomenų esybes, kurios saugo tiktai objektą identifikuojančius verslo raktus.

Sąsaja (angl. *link*) – *Data Vault 2.0* duomenų modelio komponentas. Šis komponentas skirtas aprašyti ryšius tarp skirtingų centrų. Šie komponentai savyje saugo tiktai ryšį tarp dviejų ar daugiau centrų, apibūdinančius verslo raktus.

Palydovas (angl. *satellite*) – *Data Vault 2.0* duomenų modelio komponentas. Šis komponentas skirtas aprašyti visą likusį kontekstą, kurio neapima centrai ir sąsajos. Šie komponentai yra labiausiai besikeičiantys, kadangi jie aprašo visą papildomą informaciją, kuri nėra skirta identifikavimui ir dėl to yra daug labiau linkusi keistis. Palydovai gali turėti ryšius tiek su centrais, tiek ir su sąsajomis. Kiekvienas centras ar sąsaja gali turėti ryšius su keletą palydovų, siekiant informaciją struktūrizuoti pagal funkcionalumą, pokyčių dažnumą ar duomenų šaltinį.

Duomenų paruošimo sritis (angl. *staging area*) – pirmoji *Data Vault 2.0* duomenų saugyklos architektūros sritis. Ši sritis skirta neapdorotų duomenų įkėlimui į duomenų saugyklą. Į šią sritį duomenys patenka iš įvairiausių duomenų šaltinių, duomenys pateikiami skirtingais formatais.

Duomenų saugojimo sritis (angl. *enterprise data warehouse*) – antroji *Data Vault 2.0* duomenų saugyklos architektūros sritis. Ši sritis skirta įkeltų duomenų apdorojimui ir struktūrizavimui. Tai svarbiausia *Data Vault 2.0* architektūros sritis. Šioje srityje visi pirmojoje srityje esantys duomenys yra apjungiami į duomenų struktūras pagal *Data Vault 2.0* modeliavimo metodiką ir išsaugojami *Data Vault* sluoksnyje.

Informacijos perdavimo sritis (angl. *information delivery*) – trečioji *Data Vault 2.0* duomenų saugyklos architektūros sritis. Ši sritis skirta duomenų atvaizdavimui ir analizei įvairiais pjūviais. Šie pjūviai yra skirti galutiniam duomenų saugyklos naudotojui, kuriais remdamasis naudotojas gali priimti sprendimus, analizuoti sistemos klaidas, metaduomenis, duomenų saugyklos metrikas.

Įvadas

Šis darbas priklauso Kauno technologijos universiteto Veiklos skaitmeninimo ir sistemų architektūrų studijų programai.

Darbo problematika ir aktualumas

Duomenų saugykla – įvairių didelių duomenų apdorojimo ir saugojimo sistema, skirta duomenų analizei. Duomenų saugyklos daugiausiai naudojamos dideliems veiklos duomenims apdoroti, kad jie būtų paversti naudinga informacija, pagal kurią būtų priimami svarbūs veiklos valdymo sprendimai [1]. Duomenys į saugyklą patenka iš įvairiausių duomenų šaltinių ir gali būti tiek struktūrizuoti (saugojami reliacinėse duomenų bazėse), tiek pusiau struktūrizuoti (saugojami *JSON* formato struktūrose, *HTML* kalbos atributuose ir pan.) arba iš viso nestruktūrizuoti (saugojami tekstiniuose dokumentuose, nuotraukose ir pan.) [2]. Šie duomenys yra apdorojami pagal iš anksto aprašytus procesus, struktūrizuojami ir išsaugojami duomenų saugykloje, o vėliau pernaudojami įvairioms reikšmingoms ataskaitoms generuoti. Visas šis duomenų apdorojimo procesas vadinamas iškėlimu-apdorojimu-išsaugojimu – *ETL* (angl. *Extract-Transform-Load*) [3].

2015 metais Daniel Linstedt su Micheal Olschimke pasiūlė naują duomenų saugyklų architektūrą *Data Vault 2.0* bei pateikė metodiką kaip šią architektūrą realizuoti [3]. Pirmiausia, metodika pasiūlo naują duomenų modelį, kitokį, negu naudoja tradicinės duomenų saugyklos. Naujasis duomenų modelis, skirtingai nei pirmtakai, orientuotas į sistemos plečiamumą bei duomenų struktūros keičiamumą. *Data Vault 2.0* ne tik suteikia galimybę sistemą lengvai išskirstyti per keletą aparatinės įrangos mašinų, bet taip pat užtikrina paprastesnį naujų duomenų struktūrų pridėjimą [4]. Sistema taip pat palaiko automatizaciją, dėl to valdyti ir keisti duomenų saugyklą yra paprasčiau [4].

Nors *Data Vault 2.0* technologija ir išsprendžia keletą tradicinių duomenų saugyklų architektūrų problemų, ši metodika vis dar yra nauja. Dėl to vis dar egzistuoja nedaug automatizacijos įrankių, įgyvendinančių šį sprendimą [5], [6]. Nors šie įrankiai automatizuoja duomenų modelio sukūrimo procesą (*Data Vault* sukūrimą), jie nesuteikia galimybės įvertinti duomenų saugyklos kompiuterinių resursų metrikų.

Šiuo metu *Data Vault 2.0* kompiuterinių resursų metrikų informacija yra teikiama tik debesijos paslaugų tiekėjų ir ne visada apima visą reikiamą informaciją našumo problemoms spręsti. Dėl to yra poreikis turėti vadinamąją metrikų saugyklą (angl. *Metrics Vault*), kurioje būtų kaupiama kompiuterinių resursų metrikų informacija. Ši informacija vėliau būtų panaudojama duomenų saugyklos kontrolei, pavyzdžiui, siekiant sistemą pagreitinti, sumažinti naudojamų serverių kiekį ar paprasčiausiai sumažinti įmonės išlaidas, tenkančias debesijos paslaugoms.

Darbo tikslas ir uždaviniai

Šio darbo tikslas – palengvinti kompiuterinių resursų metrikų stebėjimą ir kontrolę *Data Vault 2.0* duomenų saugykloje, pasiūlant metrikų saugyklos sudarymo metodiką.

Tikslui pasiekti iškelti šie uždaviniai:
1. atlikti *Data Vault 2.0* metodikos analizę;
2. atlikti kompiuterinių resursų metrikų rinkimo įrankių analizę;
3. išanalizuoti duomenų saugyklų sudarymo metodikas;
4. pasiūlyti metrikų saugyklos metodiką, grindžiamą *Data Vault 2.0* architektūros principais;

5. suprojektuoti ir realizuoti metrikų saugyklos sudarymo įrankį;
6. eksperimentiškai ištirti metrikų saugyklos sudarymo metodikos ir tam skirto įrankio taikymą, sudarant metrikų saugyklą;
7. apibendrinti eksperimentinio tyrimo rezultatus.

Darbo rezultatai ir jų svarba

Šiame darbe pateikiama metrikų saugyklos sudarymo metodika *Data Vault 2.0* duomenų saugyklai. Metodika aprašo visą metrikų saugyklos sudarymo procesą nuo metrikų duomenų šaltinių pridėjimo iki metrikų duomenų atvaizdavimo. Pateikiama sudaryta metrikų taksonomija, kuri skirta palengvinti kompiuterinių resursų metrikų analizę, metrikoms priskiriant kategorijas. Pagal šią metodiką galima realizuoti metrikų saugyklą ir pademonstruoti, kad ji palengvina kompiuterinių resursų metrikų stebėjimą ir kontrolę.

Darbe taip pat pateikiamas realizuotas metrikų saugyklos sudarymo įrankis, kuris automatizuoja metrikų saugyklos sudarymo žingsnius. Realizuoto įrankio pagalba naudotojai gali sumodeliuoti metrikų saugyklą, priskirti kategorijas pagal pateiktą metrikų taksonomiją, taip pat gali sugeneruoti *DDL* ir *ETL* kodą, skirtą sukurti *Data Vault* duomenų saugyklą ir ją užpildyti metrikų duomenimis.

Siekiant įvertinti pasiūlytos metodikos ir realizuoto įrankio pritaikomumą buvo įvykdytas ir aprašytas eksperimentas. Eksperimento metu sukurta metrikų saugykla, skirta saugoti įmonės serverių resursų duomenis, bei įvykdyta apklausa, kurios metu apklausti su *Data Vault* duomenų saugyklomis dirbantys specialistai. Sukurtoje metrikų saugykloje išsaugoti apie 13 mln. įrašų ir sukurtos vizualizacijos 7 skirtingoms metrikoms. Apklausoje surinkti respondentų atsakymai, patvirtinantys, kad pasiūlyta metodika ir realizuotas įrankis gali paskatinti naudotojus kurti *Data Vault* metrikų saugyklas kompiuterinių resursų metrikų stebėjimo uždaviniams spręsti.

Darbo struktūra

Visą darbą sudaro penki skyriai. Pirmajame skyriuje pateikiama *Data Vault 2.0* duomenų saugyklų analizė, analizuojamos dažniausiai renkamos kompiuterinių resursų metrikos, palyginamos skirtingos duomenų saugyklų architektūros ir *Data Vault* sudarymo metodikos. Antrajame skyriuje aprašomi keliami reikalavimai *Data Vault* metrikų saugyklos sudarymo metodikai, pateikiama pagal juos sudaryta metodika ir formalizuotas sprendimo aprašas, aprašantis svarbiausius metrikų saugyklos sudarymo etapus. Trečiajame skyriuje aprašomi realizuoto metrikų saugyklos sudarymo įrankio panaudojimo atvejai, nefunkciniai reikalavimai, duomenų modelis ir įrankio loginė architektūra. Ketvirtajame skyriuje aprašomas realizuotas metrikų saugyklos sudarymo įrankis, įrankio testavimo planas ir diegimo modelis. Penktajame skyriuje aprašomas įvykdytas eksperimentas, skirtas įvertinti pasiūlytos metodikos ir realizuoto įrankio pritaikomumą. Pateikiamas eksperimento vykdymo planas bei gauti rezultatai, kurie yra apibendrinami pateikiant eksperimento išvadas.

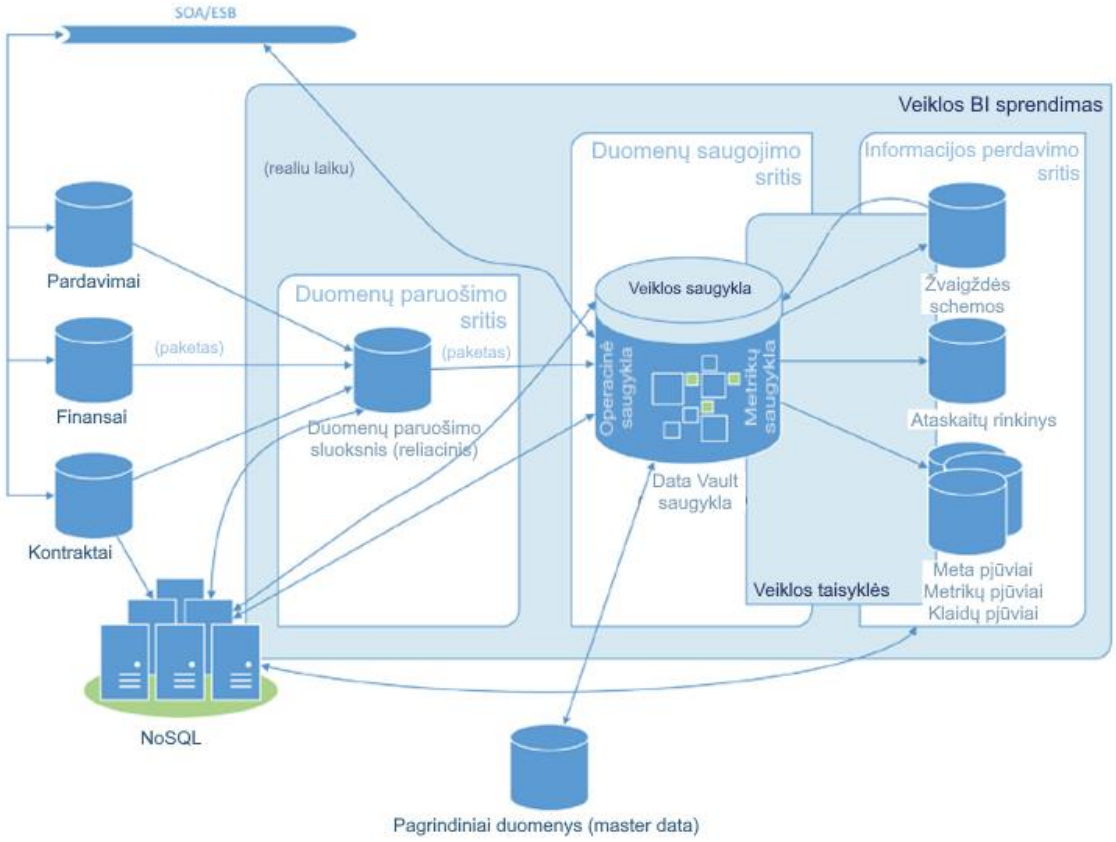
1. Probleminės srities analizė

1.1. Tyrimo objektas, sritis ir problema

Šio darbo tyrimo objektas yra duomenų saugykla, parengta pagal *Data Vault 2.0* metodiką. Duomenų saugykla yra tiriama kompiuterinių resursų metrikų ir jų valdymo kontekste. Tai atliekama todėl, nes dabartiniu metu sudėtinga identifikuoti ir išspręsti *Data Vault 2.0* duomenų saugyklos lėto, neefektyvaus ar finansiškai brangaus veikimo situacijas. Šios situacijos daugiausiai kyla dėl prasto aparatinės įrangos resursų išnaudojimo. Siekiant efektyviau išspręsti šias problemas, reikia kaupti metrikas apie duomenų saugyklos darbą, kurios indikuotų neefektyviausiai veikiančias sistemos vietas. Šiam tikslui pasiekti *Data Vault* architektūra užsimena apie metrikų saugyklos komponentą, tačiau kaip jį realizuoti, kas ir kaip jame turėtų būtų saugoma, nėra detaliai aprašyta.

1.2. *Data Vault 2.0* duomenų saugyklos analizė

Data Vault 2.0 duomenų saugyklos naudoja trijų sričių architektūrą duomenų įkėlimui, saugojimui ir perdavimui (žr. 1.1 pav. [3]). Ši architektūra grįsta *Inmon* architektūra (angl. *Inmon data warehouse architecture*) ją daugiausiai patobulinant duomenų saugojimo srityje [3].



1.1 pav. *Data Vault 2.0* duomenų saugyklos architektūra [3]

Pirmoji architektūros sritis [3] skirta neapdorotų duomenų įkėlimui į duomenų saugyklą. Ši sritis vadinama duomenų paruošimo sritimi (angl. *staging area*). Į šią sritį duomenys patenka iš įvairiausių duomenų šaltinių. Duomenys gali būti pateikiami skirtingais formatais. Duomenys gali būti reliacinės duomenų bazės įrašai, nereliacinės (angl. *NoSQL*) duomenų bazės duomenų struktūros, pusiau struktūrizuoti duomenys (*JSON*, *HTML* failai) ar nestruktūrizuoti duomenys (tekstiniai dokumentai,

paveikslėliai) [2]. Šioje *Data Vault 2.0* architektūros srityje jokia istorinė duomenų struktūrų informacija nėra saugojama. Taip pat nėra atliekamos jokios duomenų apdorojimo operacijos, tik užtikrinama, kad buvo įkelti numatyti duomenų tipai. Taip yra todėl, nes šios srities pagrindinė paskirtis – teikti duomenys sekančiai architektūros sričiai duomenų apdorojimo operacijoms. Taip sumažinamas duomenų perdavimo operacijų kiekis tarp duomenų šaltinių ir duomenų saugyklos.

Antroji architektūros sritis [3] skirta apdoroti įkeltus duomenis ir juos struktūrizuoti pagal *Data Vault 2.0* duomenų modelį. Ši sritis vadinama duomenų saugojimo sritimi (angl. *enterprise data warehouse*). Tai svarbiausia *Data Vault 2.0* architektūros sritis, kuri labiausiai išsiskiria nuo tradicinių *Inmon* (angl. *Inmon data warehouse architecture*) ir *Kimball* (angl. *Kimball data warehouse architecture*) duomenų saugyklų architektūrų. Šioje srityje visi duomenų paruošimo srityje esantys duomenys yra sujungiami į duomenų struktūras pagal *Data Vault 2.0* duomenų modelio reikalavimus ir išsaugojami *Data Vault* saugykloje. Šioje srityje pasirinktinai gali būti sukuriamos ir dar kelios papildomos saugyklos. *Business Vault* – saugykla skirta apdoroti duomenis nustatant įvairias veiklos taisykles. *Operational Vault* – saugykla skirta veiklos operacinio lygio sistemoms (angl. *operational systems*), kurios atnaujina metaduomenis ar užtikrina duomenų ir transakcijų vientisumą. *Metrics vault* – saugykla skirta kaupti duomenų saugyklos kompiuterinių resursų metrikas. Taigi, pagrindinė duomenų saugojimo srities paskirtis – sustruktūrizuoti duomenis ir juos išsaugoti. Šioje srityje saugojami duomenys toliau yra panaudojami trečioje architektūros srityje.

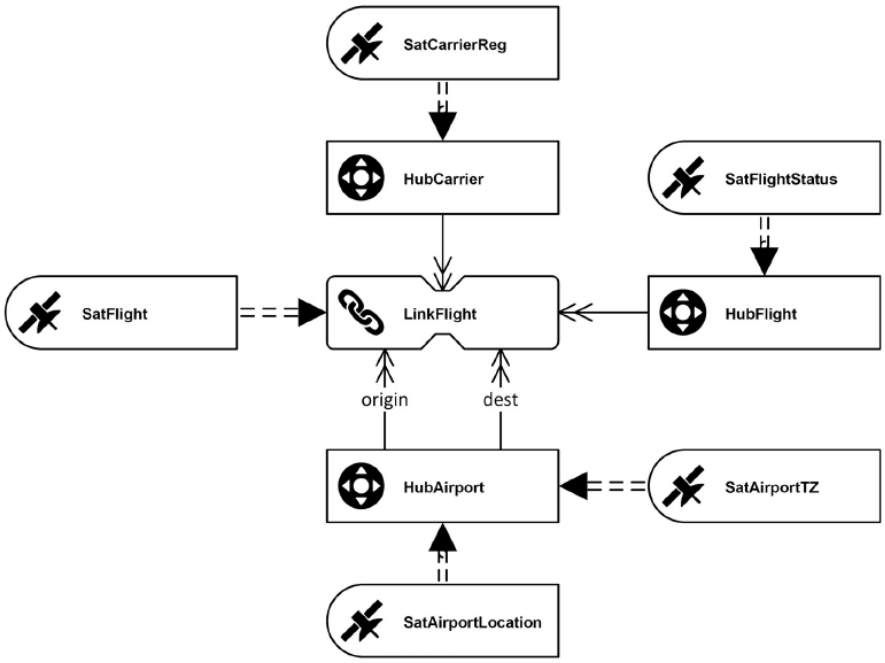
Trečioji architektūros sritis [3] skirta duomenų atvaizdavimui ir analizei įvairiais pjūviais. Ši sritis vadinama informacijos perdavimo (angl. *information delivery*) sritimi. Sritis skirta duomenų ir verslo procesų analitikams. Informacijos perdavimo srityje dažniausiai pateikiamas tik tam tikras duomenų pjūvis (angl. *information mart*), kuris gali būti pateikiamas normalizuotu pavidalu (pvz. trečiąja normaline forma), „žvaigždės“ schema (angl. *star schema*) ir pan. Šie pjūviai yra skirti galutiniam duomenų saugyklos naudotojui. Naudotojas gali panaudoti šiuos duomenų pjūvius sprendimu priėmimui, sistemos klaidų analizei, duomenų saugyklos metrikų analizei. Taigi, šios srities pagrindinė paskirtis – perduoti struktūrizuotus ir agreguotus duomenis naudotojui pagal jo išreikštus pageidavimus.

Data Vault 2.0 duomenų saugykloje duomenys yra saugojami pagal *Data Vault* duomenų modelį, kurį aprašo trys pagrindiniai komponentai: centrai (angl. *hubs*), sąsajos (angl. *links*), ir palydovai (angl. *satellites*) [3]. Kiekvienas komponentas turi specifinę paskirtį ir apribojimus.

Centrai [3] – komponentai skirti aprašyti duomenų esybes, kurios saugo tik objektą identifikuojančius verslo raktus (angl. *business keys*). Verslo raktai – tai atributai ar atributų kombinacija naudojama identifikuoti objektą veiklos kontekste. Verslo raktais gali būti asmens kodas, sąskaitos numeris, oro uosto identifikacinis kodas, automobilio markės ir modelio kombinacija. *Data Vault 2.0* metodika akcentuoja, kad šie raktai turėtų būti naudojami veikloje, o ne būti išvestiniai atributai tokie kaip dirbtiniai pirminiai raktai sutinkami duomenų bazėse. Centrų pavyzdžiai pateikiami 1.2 paveiksle parodytame pavyzdiniame skrydžių duomenų modelyje. Šiame modelyje centrai yra oro uostas (*HubAirport*), skrydis (*HubFlight*) bei oro linijos *(HubCarrier*).

Sąsajos [3] – komponentai skirti aprašyti ryšius tarp skirtingų centrų. Šie komponentai savyje saugo tik ryšį tarp dviejų ar daugiau centrų apibūdinančius verslo raktus. Sąsajos pavyzdys 1.2 paveiksle yra skrydžio sąsaja (*LinkFlight*).

Palydovai [3] – komponentai skirti aprašyti visą likusį kontekstą, kurio neapima centrai ir sąsajos. Šie komponentai yra labiausiai besikeičiantys, kadangi jie aprašo visą papildomą informaciją, kuri nėra skirta identifikavimui ir dėl to yra labiau linkus keistis. Palydovai gali turėti ryšius tiek su centrais, tiek ir su sąsajomis. Taip pat kiekvienas centras ar sąsaja gali turėti ryšius su keliais palydovais. Tokiu būdu informacija struktūrizuojama pagal funkcionalumą, pokyčių dažnumą ar duomenų šaltinį. Palydovuose gali būti saugoma tokia informacija kaip asmens amžius, lytis, automobilio spalva, apmokėjimo data ir pan. Palydovų pavyzdžiai 1.2 paveiksle yra oro linijų informacija (*SatCarrierReg*), skrydžio informacija (*SatFlight*), skrydžio būsenos informacija (*SatFlightStatus*), oro uosto laiko zonos informacija (*SatAirportTZ*), oro uosto vietos informacija (*SatAirportLocation*).



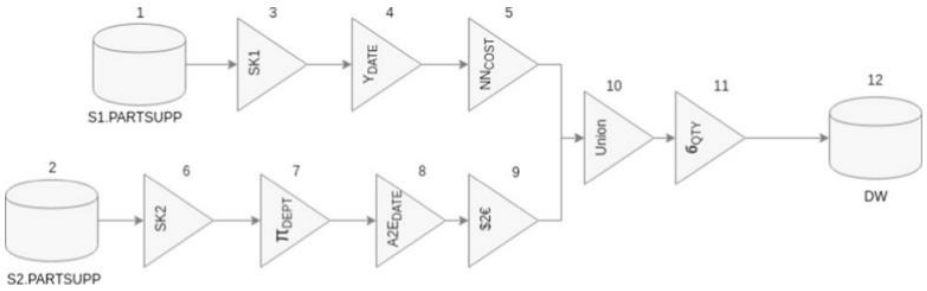
1.2 pav. *Data Vault 2.0* duomenų modelio pavyzdys skrydžių duomenų sistemai [3]

Taigi, atlikta *Data Vault 2.0* analizė parodė, kad *Data Vault 2.0* duomenų saugyklos remiasi trijų sričių architektūra, kad gaunami duomenys būtų efektyviai išnaudojami ir apdorojami. *Data Vault 2.0* daugiausiai dėmesio skiria duomenų modelio modernizavimui. Tokiu būdu siekiama šį modelį labiau priartinti prie veiklos procesų ir jų objektų. Todėl yra atsisakoma dirbtinių sistemos raktų naudojimo, o objektų identifikavimui pasitelkiami verslo raktai. Šis modernizavimas tokiu būdu palengvina duomenų saugyklos plečiamumą.

1.3. *ETL* procesų analizė

Pagrindinis metrikų duomenų šaltinis duomenų saugyklose yra *ETL* procesai. Kadangi šie procesai įkelia, apdoroja ir išsaugoja duomenis duomenų saugyklose, jie gali būti analizuojami įvairiais našumo aspektais (greitaveikos, atminties išnaudojimo, apkrautumo ir pan.). Tačiau, dėl šios plačios įvairovės yra svarbu nustatyti, kurios metrikos iš tikrųjų yra aktualios ir naudingos neefektyvaus duomenų saugyklos veikimo situacijų identifikavimui ir sprendimui.

ETL procesų eigai (angl. *ETL workflow*) atvaizduoti dažnai yra pasitelkiami kryptiniai becikliai grafai (angl. *directed acyclic graph*) [7, 8]. Šie grafai leidžia susidaryti geresnį ir išsamesnį įspūdį apie visą *ETL* procesą. Šie grafai labai tinka atvaizduoti *ETL* procesus, kadangi gerai atspindi jų struktūrą, parodo iš kokių smulkesnių užduočių (angl. *task*) jie susideda, iš kokių duomenų šaltinių renkami duomenys, kokia eiga vykdomi darbai ir kaip jie yra išlygiagretinti (žr. 1.3 pav. [7]).



1.3 pav. *ETL* procesas pavaizduotas kryptiniu becikliu grafu [7]

Siekiant optimizuoti *ETL* procesus galima pasitelkti įvairias metodikas. Kai kurios metodikos optimizuoja *ETL* procesus keisdamos darbų eigą pagal atliktą darbų selektyvumo [7, 9, 10], jų darbo spartos [10, 11] ir tarpusavio priklausomybių analizę [9], [12]. Tačiau, taip pat egzistuoja *ETL* procesų optimizavimo metodikos [13, 14] grindžiamos *ETL* proceso vykdymo istorijos parametrų apskaičiavimu ir panaudojimu tolimesnei optimizacijai. Surinkta informacija vėliau gali būti perduota mašininio mokymosi algoritmui. Algoritmas pagal pateiktus parametrus bando surasti optimalią serverių ir juose vykdomų darbų išlygiagretinimo kombinaciją, kuri tenkintų naudotojo nustatytus greitaveikos ar finansinių išteklių reikalavimus [14].

Apie *ETL* procesus galima kaupti įvairiausią informaciją [9, 15–17] kurią galima suskaidyti į kelias pagrindines grupes: greitaveikos, atminties efektyvumo, duomenų naujumo ir patikimumo, lygiagretumo bei bendrinės informacijos metrikas.

Greitaveikos metrikų pavyzdžiai:

- *ETL* proceso vykdymo laikas;
- apdorotų duomenų eilučių kiekis per sekundę;
- nuskaitytų duomenų kiekis per sekundę;
- apdorotų duomenų kiekis per sekundę;
- procesoriaus panaudojamumo procentas (angl. *CPU usage*);
- internetinio ryšio vėlavimas (angl. *network latency*) tarp duomenų atsisiuntimo ir apdorojimo procesų.

Atminties efektyvumo metrikų pavyzdžiai:

- disko skaitymo/įrašymo operacijų kiekis;
- operatyvios atminties panaudojamumo procentas;
- operatyvios atminties dydis.

Duomenų naujumo ir patikimumo metrikų pavyzdžiai:

- veiklos taisykles pažeidžiančių duomenų kiekis;
- trūkstamų duomenų procentas.

Lygiagretumo metrikų pavyzdžiai:

- procesoriaus branduolių kiekis;
- serverių kiekis;
- vienu metu lygiagrečiai vykdomų darbų kiekis;
- apdorojamų duomenų kiekis kiekviename lygiagrečiai vykdome darbe.

Bendrinės informacijos metrikų pavyzdžiai:

- debesijos paslaugų tiekėjas;
- finansinė operacijų kaina;
- operacinė sistema.

Visa ši sukaupta informacija gali būti panaudota skirtingų *ETL* procesų palyginimui. Tokiu būdu randant efektyviausią, ekonomiškiausią ar labiausiai probleminei sričiai spręsti tinkamą *ETL* proceso eigą.

Taigi, atlikta *ETL* procesų analizė parodė, kad *ETL* procesų optimizavimui pasitelkiamos įvairios našumo metrikos. Šios metrikos daugiausiai apima greitaveikos ir atminties efektyvumo parametrus, duomenų naujumo ir patikimumo duomenis, proceso išlygiagretinimo informaciją.

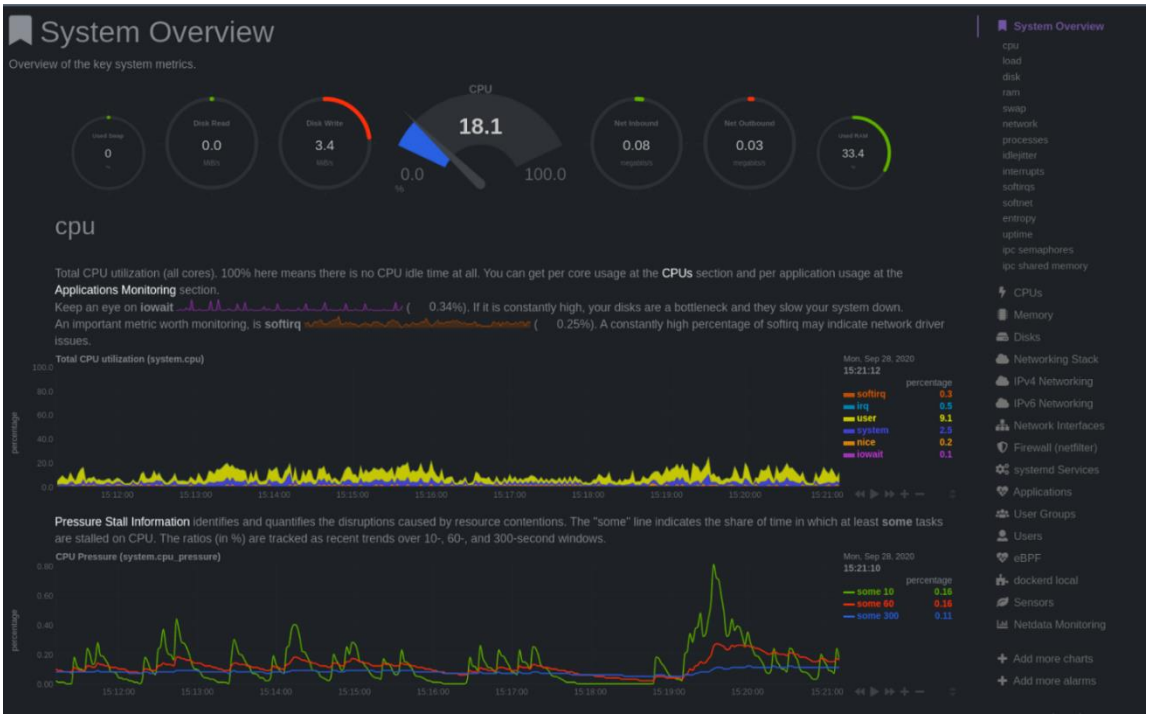
1.4. Kompiuterinių resursų metrikų rinkimo įrankių analizė

Duomenys apie metrikas gali būti surenkami pasitelkiant įvairiausius programinius įrankius. Šie įrankiai tai serveryje veikiančios programos, renkančios duomenis apie serverio darbą. Populiarios naudojamos sistemos: *Prometheus Node Exporter* [15], *Netdata* [16], *Telegraf* [17]. Šios sistemos realiu laiku nurodytu intervalu surenka įvairias serverio metrikas ir jas persiunčia į duomenų saugojimo platformą (duomenų bazę, duomenų „ežerą“, duomenų saugyklą).

Prometheus Node Exporter yra *Prometheus* įrankio papildinys skirtas eksportuoti įvairias aparatinės įrangos ir operacinės sistemos metrikas. Šis įrankis parašytas *Go* programavimo kalba ir skirtas *UNIX* tipo operacinėms sistemoms (*Linux, FreeBSD, macOS X*). Įrankis suteikia prieigą prie plataus metrikų rinkinio, kurias galima pasiekti nurodant norimus metrikų surinkėjus (angl. *collectors*). Galimi *Prometheus Node Exporter* metrikų rinkiniai: *boottime* (sistemos užsikrovimo laikas), *CPU* (procesoriaus veikimo metrikos), *cpufreq* (procesoriaus dažnio metrikos), *diskstats* (įvesties/išvesties metrikos), *filesystem* (failų sistemos metrikos), *loadavg* (duomenų įkėlimo laikas), *meminfo* (atminties naudojimo metrikos), *netdev* (internetinio ryšio naudojimo metrikos), *OS* (operacinės sistemos informacija), *schedstat* (užduočių planavimo informacija), *thermal* (procesoriaus temperatūros informacija), *time* (sistemos laikas) ir pan. [15]. Atrinktos metrikos vėliau gali būti perduotos vizualizavimo įrankiui (pvz. *Grafana*) ir vizualiai pateiktos kaip 1.4 paveiksle [18].

1.4 pav. *Prometheus Node Exporter* metrikos atvaizduotos *Grafana* įrankyje [18]

Netdata yra nepriklausomas atviro kodo įrankis skirtas ne tik eksportuoti metrikas, bet ir automatizuotai (naudojant mašininio mokymosi algoritmus) aptikti problemas ir jas spręsti. Šis įrankis parašytas *C*, *Python* ir *Javascript* programavimo kalbomis ir skirtas *Linux* tipo operacinėms sistemoms, tačiau gali rinkti ir dalį *Windows* operacinės sistemos metrikų. Įrankis teikia labai platų spektrą metrikų rinkinių, kuriuos galima pasiekti nurodant įvairius surinkėjus. Galimi surinkėjai: *disk space* (išorinės atminties naudojimo metrikos), *NVIDIA GPU* (*NVIDIA* vaizdo plokštės naudojimo metrikos), *RAM* (operatyviosios atminties naudojimo metrikos), *CPU utilization* (procesoriaus panaudojamumo metrikos), *CPU performance* (procesoriaus našumo metrikos), *CPU frequency* (procesoriaus dažnio metrikos) ir pan. Atrinktos metrikos gali būti vizualizuojamos ir pačiame *Netdata* įrankyje (žr. 1.5 pav.) [16].



1.5 pav. *Netdata* metrikų vizualizacija [16]

Telegraf yra *InfluxDB* sukurtas atviro kodo įrankis skirtas metrikų surinkimui. *Telegraf* parašytas *Go* programavimo kalba ir skirtas tiek *Linux*, tiek *Windows*, tiek *macOS* operacinėms sistemos. Įrankis taip pat siūlo platų metrikų pasirinkimą. Metrikos yra surenkamos nurodant atitinkamus įskiepius. Galimi *Telegraf* įskiepiai: *CPU* (procesoriaus naudojimo metrikos), *Disk* (išorinės atminties naudojimo metrikos), *DiskIO* (įvesties/išvesties operacijų metrikos), *Filestat* (failų sistemos informacija), *Mem* (operatyviosios atminties naudojimo metrikos), *Net* (internetinio ryšio naudojimo metrikos), *System* (sistemos informacija) ir pan. Atrinktos *Telegraf* metrikos gali būti vizualiai atvaizduojamos ir *InfluxDB* grafinės sąsajos papildinyje (žr. 1.6 pav.) [17].

1.6 pav. *Telegraf* metrikų vizualizacija *InfluxDB* grafinėje sąsajoje [17]

Kaip galime matyti, metrikų rinkinys priklauso nuo kiekvieno įrankio, bet jos gali būti sujungtos į kelias pagrindines metrikų grupes.

Procesoriaus darbo metrikos:

- procesoriaus branduolių kiekis;
- procesoriaus branduolių apkrautumas;
- procesoriaus branduolio įvesties/išvesties operacijų laukimo laikas;
- procesoriaus branduolio pertraukčių laukimo laikas.

Atminties naudojimo metrikos:

- naudojamos atminties dydis;
- laisvos atminties dydis;
- naudojamos spartinančios atminties dydis.

Disko operacijų metrikos:

- skaitymo operacijų greitis;
- rašymo operacijų greitis;
- skaitymo operacijų kiekis;
- rašymo operacijų kiekis;

Internetinio ryšio naudojimo metrikos:

- išsiųstų paketų kiekis;
- gautų paketų kiekis;
- išsiuntimo greitis;
- atsisiuntimo greitis;
- klaidų kiekis.

Temperatūros metrikos:

- procesoriaus branduolių temperatūra;
- procesoriaus branduolio temperatūros kritinė zona.

Taigi, metrikų kaupimo įrankiai dažniausiai renka informaciją apie šias pagrindines metrikų kategorijas: procesoriaus darbo metrikas, atminties naudojimo metrikas, disko operacijų metrikas, internetinio ryšio naudojimo metrikas, temperatūros metrikas.

1.5. Metrikų taksonomijos analizė

Įvairių procesų analizei atlikti neretai yra pasitelkiamos metrikų grupės padedančios įvertinti proceso kontekstą. Šis metrikų klasifikavimas neretai sutinkamas debesijos paslaugų tiekėjų sprendžiamuose uždaviniuose. Uždavinių tikslas – užtikrinti pastovų, patikimą ir naudotojo poreikius atitinkantį kompiuterinių resursų prieinamumą. Metrikų taksonomijos apibrėžimas padeda geriau nustatyti aktualius duomenų rinkinius ir jų sąsajas, tokiu būdu palengvindamas resursų prieinamumo užtikrinimą.

Vieno bendrinio priimto standarto debesijos paslaugų metrikų klasifikacijai nėra. Šios metrikų grupės dažniausiai priklauso nuo debesijos paslaugų tiekėjo. Atliktose mokslinėse paslaugų kokybės, našumo analizėse metrikų grupės tai pat neretai išsiskiria ir priklauso nuo autorių apibrėžimo (žr. 1 lentelė). Tačiau, siekiant standartizuoti šią metrikų įvairovę [19], autoriai išskiria tris pagrindines metrikų kategorijas, klasifikuojančias įvairias metrikas naudojamas komunikacinėse sistemose. Šios trys pagrindinės kategorijos yra: energijos metrikos (angl. *power-related metrics*), internetinio ryšio metrikos (angl. *network traffic metrics*) ir našumo metrikos (angl. *performance metrics*). Energijos metrikos – tai metrikos, skirtos įvertinti, kaip efektyviai išnaudojami elektros energijos ištekliai. Internetinio ryšio metrikos – tai metrikos, skirtos įvertinti internetiniu srautu perduodamų duomenų spartą, kiekį, kokybę, delsimą (angl. *latency*). Našumo metrikos – plačiausia metrikų kategorija, skirta įvertinti, kaip efektyviai yra panaudojami pagrindiniai kompiuteriniai resursai: procesorius, operatyvioji atmintis, išorinė atmintis. Ši, [19] autorių siūloma, metrikų taksonomija padeda suklasifikuoti didelį kiekį debesijos paslaugų naudojamų kompiuterinių resursų metrikų. Tačiau analizuojant [20–23] šaltinius galima pastebėti, kad didžioji dalis metrikų patenka po našumo metrikų kategorija (žr. 2 lentelė). Tai labiausiai kenkia tikslui turėti aiškią, pagal paskirtį apibrėžtą metrikų taksonomiją. Kita problema – skirtingas angliškos *performance* sąvokos interpretavimas. [20] šaltinio autoriai *performance* sąvoka apibūdina visą debesijos sistemos darbą, ne tik jos našumą. Po sąvoka patenka ir sistemos prieinamumo, saugumo ir kitos panašios metrikos. [20–23] darbuose galima matyti, kad sistemos našumas daugiau įvardinamas kaip kompiuterinių resursų panaudojamumas (angl. *resource utilization*). Dėl to kyla poreikis [19] autorių pateikiamą *performance* kategoriją praplėsti ir labiau detalizuoti.

Atlikta [20–25] šaltinių analizė parodė, kad dažniausiai yra nagrinėjamos trys kompiuterinių resursų panaudojamumo kategorijos: procesoriaus darbo metrikos, operatyviosios atminties darbo metrikos

ir išorinės atminties darbo metrikos. Procesoriaus darbui įvertinti dažniausiai pasitelkiamos procesoriaus apkrautumo/panaudojamumo metrikos. Šios metrikos apibūdina kokią procentinę dalį laiko procesorius praleidžia atlikdamas skaičiavimus, kokią dalį laukdamas išorinių įrenginių, kokią dalį budėdamas. Operatyviosios atminties darbui įvertinti taip pat pasitelkiama atminties apkrautumo/panaudojamumo metrika. Taip pat matuojamas atliktų skaitymo ir rašymo operacijų kiekis per apibrėžtą laiko tarpą. Išorinės atminties darbui įvertinti pasitelkiamos skaitymo ir rašymo operacijų kiekio metrikos, duomenų perdavimo laikas.

1 lentelė. Metrikų kategorijos, minimos skirtinguose šaltiniuose

Šaltinis	**Metrikų kategorijos įvardinamos šaltinyje**	
[19]	Energijos metrikos	
	Kompiuterių tinklų metrikos	
	Sistemos darbo metrikos	
[20]	Sistemos darbo metrikos	Resursų panaudojamumas
[21]	Resursų panaudojamumas	
[22]	*CPU*	
	RAM	
	I/O	
[23]	*CPU*	
	RAM+I/O	
	Kompiuterių tinklai	
[24]	*CPU*	
	RAM	
	I/O	
[25]	-	

2 lentelė. Atliktoje metrikų analizėje dažniausiai minimos metrikos ir jų kategorijos

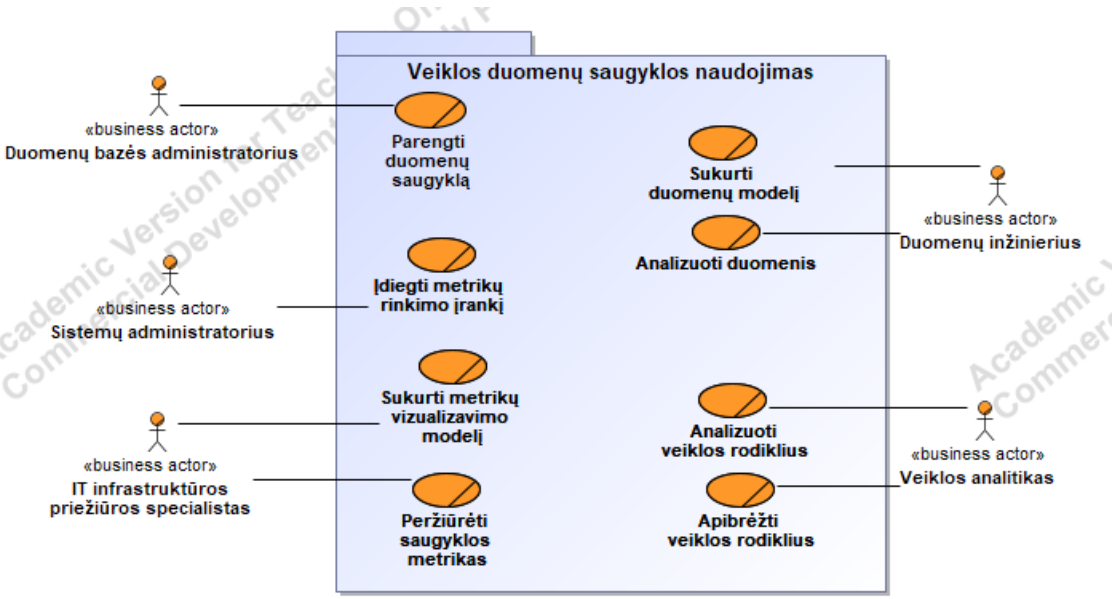
Metrika	**Šaltiniai**	**Kategorijos**							
		Energijos naudojimas	**Kompiuterių tinklai**	**Sistemos darbas**	**Resursų panaudojamumas**	***CPU***	***RAM***	***I/O***	**Kita**
CPU panaudojamumas	[20–25]			X	X	X			
Atminties panaudojamumas	[20–25]			X	X		X		
I/O skaitymo / rašymo operacijų kiekis	[20–25]			X	X			X	
Internetinio tinklo apkrautumas	[19–23, 25]		X						
Pralaidumas (angl. *throughput*)	[20, 22–24]		X						
Delsa (angl. *latency*)	[19, 20, 22, 24]		X						
Energijos sunaudojimas	[19, 20, 23]	X							
Temperatūra	[20, 23]	X							
Klaidų dažnis (angl. *error rate*)	[19]			X					X
Pasiekiamumas (angl. *availability*)	[20]			X					X
Kaštai / Pelnas	[20–24]			X					X

Taigi, atlikta metrikų kategorijų analizė parodė, kad skirtinguose šaltiniuose yra įvardijamos skirtingos metrikų kategorijos. Tam tikros iš jų persidengia, tam tikros apima platesnį kontekstą nei kitos. 1 lentelėje yra pateikiamos analizuotose šaltiniuose minimos kategorijos. Labiausiai struktūrizuotą ir plačiausią kontekstą aprašo [19] šaltinyje pateikta taksonomija. Jos kategorijos apima energijos metrikas, kompiuterių tinklų metrikas ir sistemos darbo metrikas. Kiti šaltiniai daugiausiai aprašo tiktai sistemos darbo, ypač resursų panaudojamumo (*CPU, RAM, I/O*) tipo metrikas.

Tačiau resursų panaudojamumo metrikos taip pat dažnai yra išskiriamos ne be reikalo. Kadangi, kaip galima matyti iš 2 lentelės, jos yra svarbiausios ir dažniausiai analizuojamos metrikos siekiant įvertinti sistemų būklę, sudarant metrikų taksonomiją į tai būtina atsižvelgti ir labiau detalizuoti kompiuterinių resursų panaudojamumo kategoriją. Taip pat buvo pastebėta, kad kai kurios sistemos darbą aprašančios metrikos neapibūdina konkretaus resurso panaudojamumo, o yra bendrinės sistemos darbo metrikos. Todėl būtina išskirti ir atskirą kategoriją pavadinimu *Kita*, kuri apibūdintų visas kitas sistemos darbo metrikas, kurios neapibūdina individualaus resurso panaudojamumo ar būklės.

1.6. Tyrimo objekto naudotojų analizė

Data Vault 2.0 duomenų saugykloje saugomus duomenis naudoja daug skirtingų naudotojų. Veiklos analitikai analizuoja pagrindinius veiklos rezultatų rodiklius, duomenų inžinieriai analizuoja veiklos procesų duomenis pasitelkdami įvairius statistinius skaičiavimus, programuotojai naudoja agreguotą informaciją įvairių ataskaitų ir grafikų atvaizdavimui. IT infrastruktūros specialistai verslo duomenų neanalizuoja, bet stebi šių duomenų išgavimo ir apdorojimo procesų metrikas, prižiūri aparatinės įrangos būklę, stebi jos apkrautumą. Tam pasiekti jiems reikia turėti metrikų surinkimo įrankius, metrikų duomenų saugyklas ir vizualizacijas, todėl šiam procesui realizuoti reikalingi panaudojimo atvejai aprašyti 1.7 paveiksle.

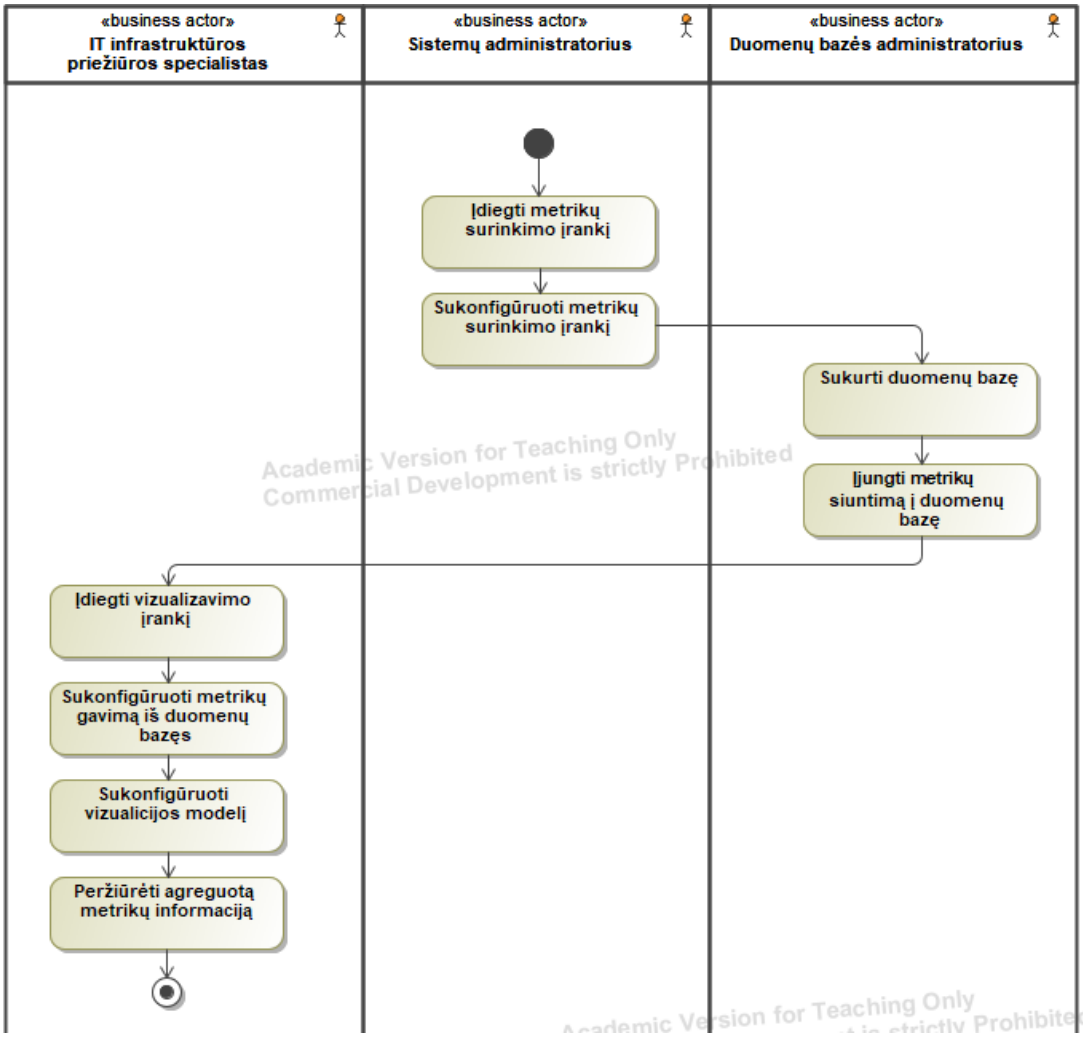


1.7 pav. Metrikų surinkimo ir atvaizdavimo panaudojimo atvejai

Tačiau dabartiniu metu IT infrastruktūros priežiūros specialistai turi pasitelkti papildomas išorines sistemas infrastruktūros būsenos stebėjimui ir metrikų surinkimui. Ši infrastruktūra dažniausiai susideda iš trijų pagrindinių komponentų:

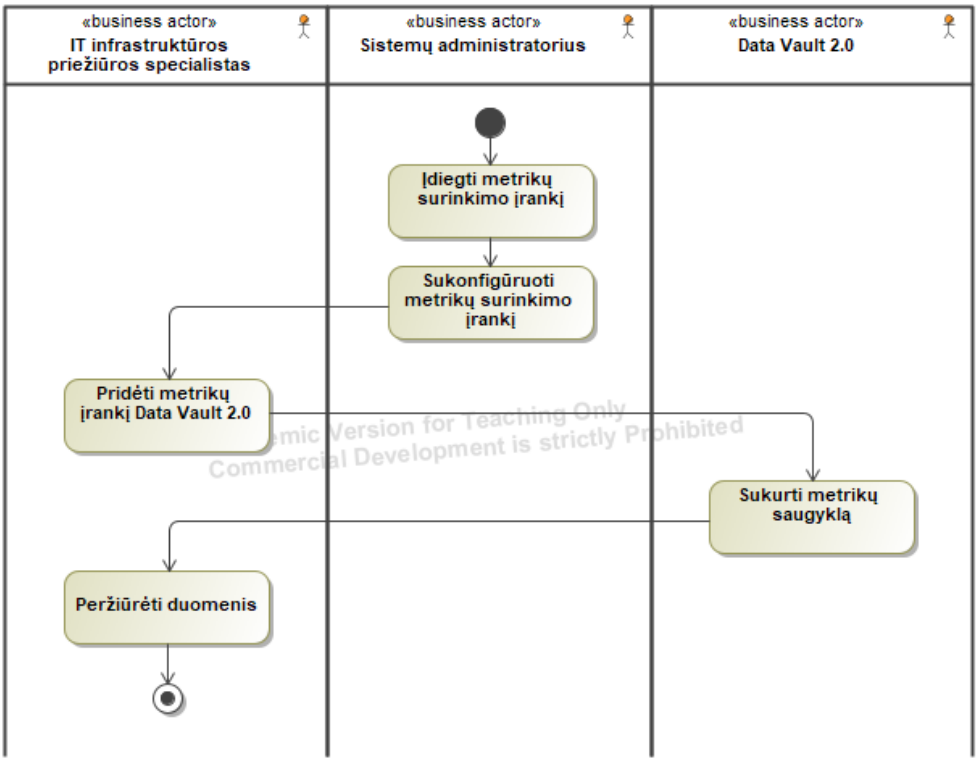
1. metrikų šaltinio surinkimo įrankio (*Prometheus Node Exporter, Telegraf, Netdata* ir pan.);
2. metrikų duomenų bazės (*InfluxDB, Prometheus DB, Timescale DB* ir pan.);
3. vizualizavimo įrankio (*Grafana, Kibana, Datadog* ir pan.).

Dabartinį metrikų surinkimo ir peržiūros procesą taip pat atvaizduoja ir 1.8 paveiksle pavaizduota veiklos diagrama.



1.8 pav. Esama metrikų rinkimo ir stebėsenos situacija

Kaip galima matyti, dabartiniu metu IT infrastruktūros priežiūros specialistai turi pasikliauti dviem papildomais įrankiais ir taip sudėtingoje IT infrastruktūroje. Taip pat šie įrankiai reikalauja papildomos konfigūracijos, saugaus ir pastovaus duomenų persiuntimo užtikrinimo ir papildomų darbuotojų įsikišimo. Todėl yra siūloma metrikų saugojimo ir agregavimo procesus perkelti į *Data Vault 2.0* duomenų saugyklą. Metrikos būtų saugomos šalia verslo procesų duomenų specializuotoje metrikų saugykloje, o agreguoti metrikų duomenys būtų atvaizduojami informacijos perdavimo srityje. Tokiu būdu būtų pasiekiama situacija vaizduojama 1.9 veiklos diagramoje.



1.9 pav. Siekiama metrikų rinkimo ir stebėsenos situacija

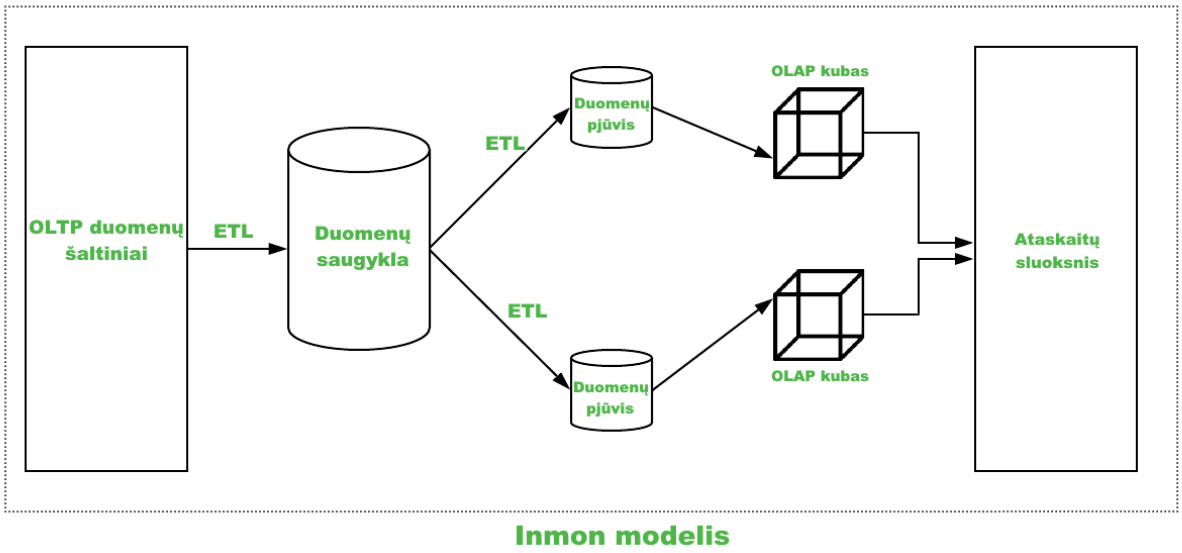
Kaip galima matyti, siekiamu būdu atliekant metrikų rinkimą ir stebėjimą, sumažėtų konfigūruojamų įrankių skaičius, procesas taptų paprastesnis, būtų mažiau duomenų perdavimo srautų ir metrikų duomenys būtų susieti su *ETL* procesais.

1.7. Duomenų saugyklų architektūrų ir *Data Vault 2.0* sudarymo metodų analizė

1.7.1. Duomenų saugyklų architektūrų palyginimas

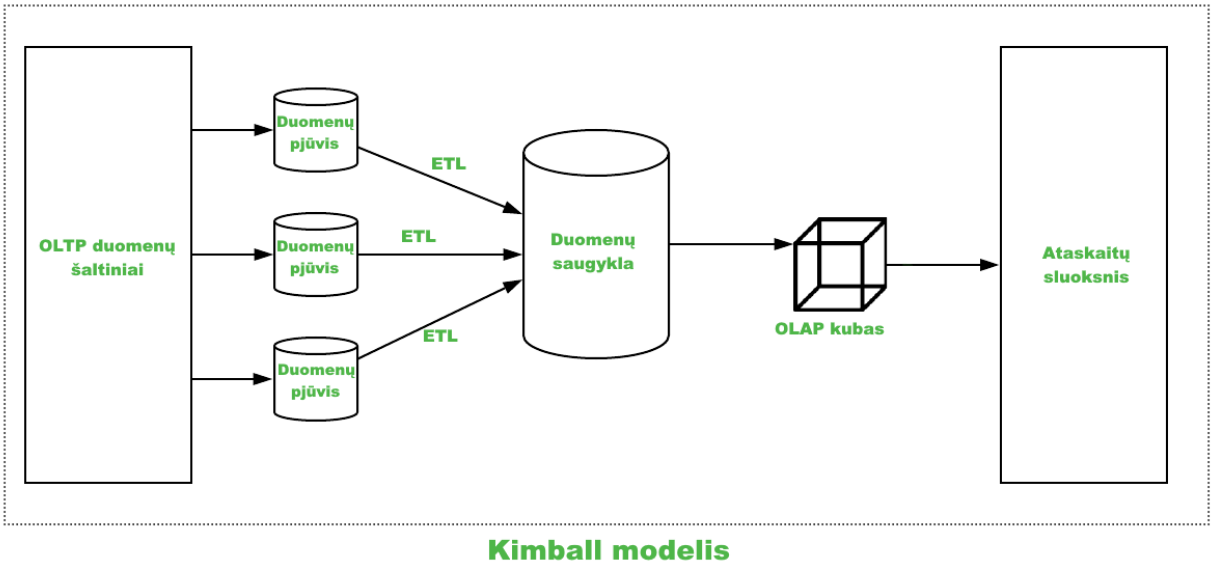
Duomenų saugyklos architektūros kūrimui gali būti pasitelktos ir kitos metodikos negu *Data Vault 2.0*. *Inmon* ir *Kimball* metodikos yra vienos populiariausių duomenų saugyklos sudarymo metodikų.

Inmon architektūra formuojama iš viršaus į apačią (angl. *top-down approach*) [3, 4]. Pagal *Inmon* architektūrą pirmiausia turi būti parengiamas duomenų saugyklos duomenų modelis. Šis modelis aprašo kaip duomenys bus saugojami duomenų saugykloje. *Inmon* architektūros atveju, duomenų modelis yra normalizuota duomenų schema (3 normalinė forma). Suformavus duomenų modelį yra parengiami duomenų pjūviai. Šie pjūviai sudaromi naudojant dimensijų modelius tokius kaip „žvaigždės“ ar „snaigės“ schemos. Ši duomenų saugyklos architektūra pavaizduota 1.10 paveiksle [26].



1.10 pav. *Inmon* duomenų saugyklos modelis [26]

Kimball architektūra formuojama iš apačios į viršų (angl. *bottom-up approach*) [3, 4]. Pagal *Kimball* architektūrą pirmiausia suformuojami duomenų pjūviai pasitelkiant „žvaigždės“ schemą. Tada šie duomenų pjūvių dimensijų modeliai yra apjungiami į vieną visumą. Ši visuma suformuoja duomenų saugyklos duomenų modelį. *Kimball* duomenų saugyklos architektūra pavaizduota 1.11 paveiksle [26].



1.11 pav. *Kimball* duomenų saugyklos modelis [26]

Šios metodikos yra senesnės už *Data Vault 2.0* metodiką ir nepateikia metrikų saugyklos ar panašių alternatyvų duomenų saugyklos kompiuterinių resursų metrikų rinkimui. Tačiau, pagrindiniai šių saugyklų pranašumai prieš *Data Vault* pagrindu sudarytą duomenų saugyklą yra duomenų išrinkimo greitis ir paprastumas (*Kimball* architektūros atžvilgiu) ir normalizuoto modelio kompaktiškumas (*Inmon* architektūros atžvilgiu).

Kadangi *Data Vault* taip pat kaip ir *Inmon* architektūra stengiasi aprėpti ir išsaugoti visą veiklos sritį apimantį duomenų modelį, abi architektūros orientuotos į kiek galima didesnį informacijos dubliavimo sumažinimą. Architektūros tam pasitelkia panašius, tačiau skirtingus būdus: *Inmon* – 3 normalinę formą, o *Data Vault* – savo duomenų modelį sudarytą iš centrų, sąsajų ir palydovų. Tačiau, didelis duomenų išskaidymas tarp didelio kiekio lentelių turi neigiamą pasekmę. Duomenų išrinkimo užklausos iš duomenų saugyklos tampa sudėtingesnės, nes reikia sujungti daug skirtingų lentelių, ir lėtesnės, nes lentelių sujungimai yra lėtesni, negu tiesioginis duomenų išrinkimas iš lentelės. Šiai problemai spręsti *Data Vault* siūlo pasitelkti *Business Vault* saugyklą, kurioje būtų atrenkami ir susiejami dažnai kartu nagrinėjami duomenys tam pasitelkiant *PIT* (angl. *Point In Time*) ir *Bridge* lenteles [3]. *PIT* lentelės skirtos kaupti konkretaus momento informaciją apie norimo centro ir jo palydovų informaciją, kuri gali būti greitai išrenkama pasitelkiant duomenų filtravimą pagal išorinį raktą užfiksuotą *PIT* lentelėje tam tikram laiko momentui. *Bridge* lentelės skirtos palengvinti dažnai kartu naudojamų centrų sujungimą išsaugant konkrečių centrų įrašų ir juos jungiančių sąsajų raktus vienoje lentelėje, tokiu būdu supaprastinant ir pagreitinant duomenų išrinkimo procesą.

Kitas *Data Vault* trūkumas – didelis duomenų modelis [27]. Ši problema kyla dėl to, kad architektūra stengiasi užtikrinti didelį ir lengvą duomenų saugyklos plečiamumą bei nesudėtingą automatizaciją. Nors *Inmon* architektūra ir sukuria didelį duomenų modelį atlikdama normalizaciją 3 normaline forma, šis duomenų modelis stipriai nesiplečia ir atitinkamai yra mažesnis nei *Data Vault* duomenų modelis. Tačiau, šį didelį duomenų modelį *Data Vault* išnaudoja tuo, kad norint modelį praplėsti ar atnaujinti pasikeitus veiklos duomenų struktūrai, dažniausiai tereikia pridėti naujus palydovus prie jau egzistuojančių centrų, o *Inmon* ar *Kimball* architektūrai šie pokyčiai yra daug sudėtingesni [28]. *Inmon* duomenų modelis, kadangi yra 3 normalinės formos, turi daug tarpusavio priklausomybių, kurias atnaujinti išsaugant istorinius duomenis ir jų struktūrą yra sudėtinga ir gali pareikalauti visiškai naujos schemos sudarymo. *Kimball* duomenų modelis reikalauja naujos schemos sudarymo, kadangi yra orientuotas į galutinės informacijos išvedimą ir nesaugo visos veiklos struktūros, kiekvienas didesnis informacijos pokytis reikalauja naujos žvaigždės schemos sudarymo. Galiausiai, didelio *Data Vault* duomenų modelio kompleksiškumo problemą siūloma spręsti pasitelkiant automatizacijos įrankius, kadangi architektūra yra palanki duomenų saugyklos sudarymo procesų automatizavimui [4] ir rankiniu būdu kuriamas sudėtingesnis veiklos duomenų modelis gali greitai peraugti į sudėtingai suvaldomą procesą.

Visų nagrinėtų architektūrų trūkumai ir pranašumai pateikiami lyginamojoje analizėje (žr. 3 lentelė [3, 4, 28]).

3 lentelė. Duomenų saugyklų modeliavimo metodikų palyginimas [3, 4, 28]

	Inmon	***Kimball***	***Data Vault 2.0***
Paskirtis	IT profesionalams	Galutiniams sistemos naudotojams	Galutiniams sistemos naudotojams
Duomenų modelio dydis	Didelis – saugomi visi veiklos duomenys normalizuoti 3 normaline forma	Mažiausias – saugomos nedidelės kompaktiškos schemos reikalingos tik duomenų išrinkimui	Didžiausias – duomenys stipriai skaidomi, dėl ko atsiranda daug skirtingų duomenų lentelių
Daugelio duomenų šaltinių integracija	Duomenų transformacijos taisyklės turi būti įdiegtos į *ETL* procesus	Duomenų transformacijos taisyklės turi būti įdiegtos į *ETL* procesus	Duomenų išskirstymas pagal duomenų pobūdį sumažina sudėtingumą

	Inmon	***Kimball***	***Data Vault 2.0***
Duomenų modelio plečiamumas	Tam tikri pokyčiai gali reikalauti istorinių duomenų migracijos	Tam tikrais atvejais gali tekti modifikuoti duomenų lenteles	Tereikia pridėti naujus palydovus
Duomenų pjūvių reikalavimų pokyčiai	Nereikalingi duomenų modelio pokyčiai	Duomenų modelis turi būti atnaujintas	Nereikalingi duomenų modelio pokyčiai
Užklausų greitis	Užklausos lėtos dėl 3 normalinės formos	Užklausos greitos, nes architektūra sudaryta duomenų pjūvių pagrindu	Tiesioginės užklausos lėtos. Reikalingi duomenų pjūviai suformuoti naudojant dimensijų modelius
***ETL* procesų aprašymo sudėtingumas**	Nesudėtingas, jei duomenų šaltinių ir duomenų saugyklos duomenų modeliai panašūs	Sudėtingos transformacijos iš operacinių duomenų modelių į dimensijų modelius	Paprastos transformavimo taisyklės į centrus, sąsajas ir palydovus
Automatizacija	Nepritaikyta duomenų saugyklos sudarymo automatizacijai	Nepritaikyta duomenų saugyklos sudarymo automatizacijai	Tinkama automatizacijai, sunkus saugyklos kūrimas rankiniu būdu (reikalauja automatizacijos įrankių)
Aprašo metrikų saugyklą	Neminima	Neminima	Užsimena apie galimybė turėti metrikų saugyklą
Aprašo metrikų saugyklos sudarymo metodiką	Ne	Ne	Ne

Taigi, kaip galima matyti iš atliktos lyginamosios analizės *Data Vault 2.0* architektūra yra palankiausia duomenų saugyklos plečiamumo atžvilgiu ir nuolat kintančių veiklos pokyčių atžvilgiu. Taip pat *Data Vault 2.0* architektūra aprašo saugyklos metrikų rinkimo ir saugojimo idėją, kurią reikėtų išplėsti ir detalizuoti, pasiūlant tam skirtą metodiką.

1.7.2. *Data Vault 2.0* sudarymo metodikų palyginimas

Data Vault 2.0 metodika aprašo bendrinius duomenų saugyklos duomenų modelio sudarymo principus skirtingoms duomenų saugyklos architektūros sritims. Tačiau, pačios saugyklos sukūrimas, duomenų šaltinių struktūrų sujungimas ir duomenų atvaizdavimas stipriai priklauso nuo dalykinės srities. Todėl siekiant palyginti kuriamą metodiką buvo išanalizuoti kiti metodai detaliai nagrinėjantys šiuos aspektus.

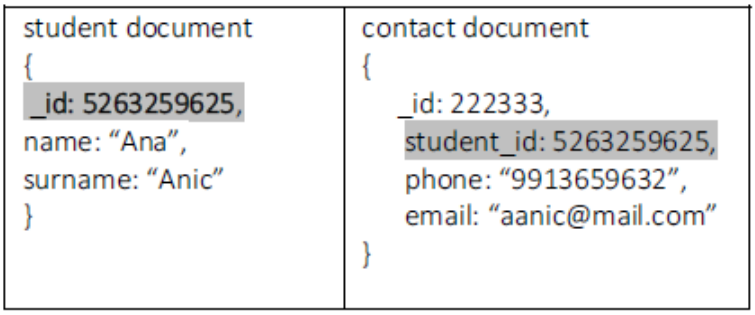
Pirmiausia, duomenys į *Data Vault 2.0* duomenų saugyklą gali būti įkeliami iš įvairiausių duomenų šaltinių ir būti pateikiami įvairiais formatais. Tai gali būti struktūrizuoti, pusiau struktūrizuoti ar iš viso nestruktūrizuoti duomenys. Šių duomenų sujungimas ir kategorizavimas pagal vieną duomenų modelį yra sudėtingas procesas. Šiam procesui palengvinti yra parengtos metodikos aprašančios pagal kokias taisykles turi būti identifikuojami verslo raktai, išskiriami centrai (angl. *hubs*), sąsajos (angl. *links*) ir palydovai (angl. *satellites*). Pavyzdžiui, [29] šaltinio autoriai pateikia metodiką kaip išskaidyti ir transformuoti *JSON* struktūros dokumentus saugomus nereliacinėje *MongoDB* duomenų bazėje, kad jie atitiktų *Data Vault 2.0* modeliui keliamus reikalavimus.

[29] autoriai pateikia 5 pagrindines taisykles duomenų transformavimui:

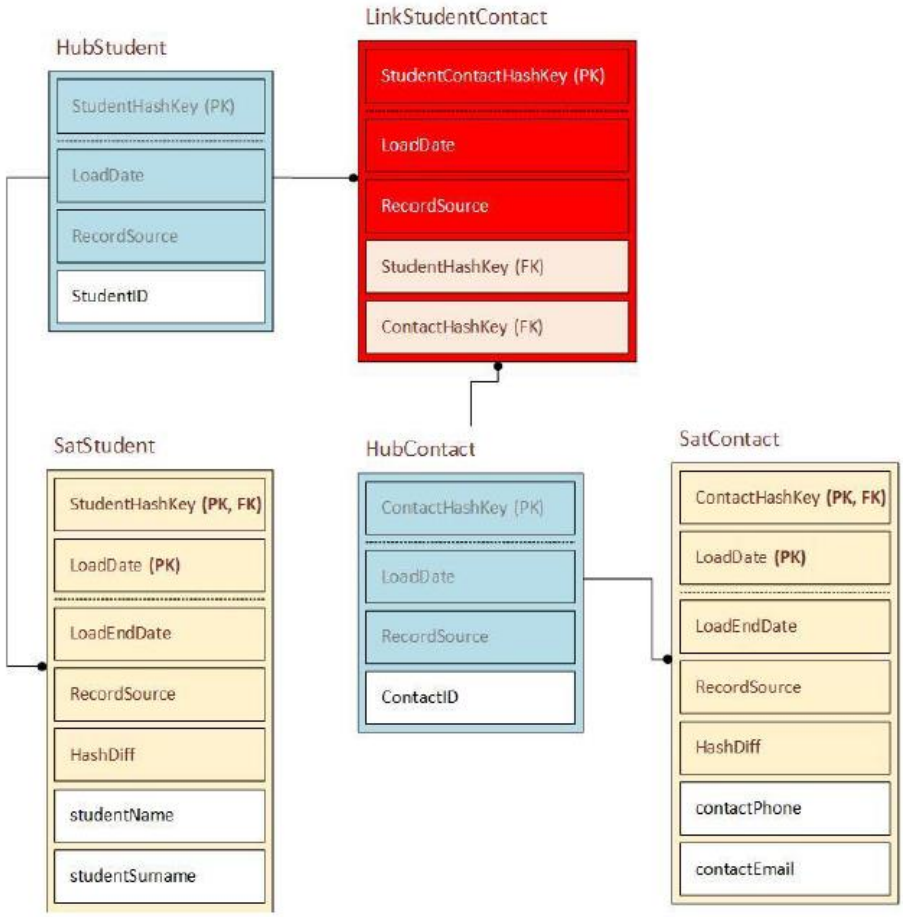
1. visi dokumentai saugojami *MongoDB* yra atitinkamai išskaidomi į centrus (angl. *hubs*), sąsajas (angl. *links*) ir palydovus (angl. *satellites*);

2. kiekvienas dokumento identifikatorius (*id*) yra paverčiamas verslo raktu saugojamu centre;
3. neidentifikuojantys laukai yra paverčiami į palydovo atributus, kur palydovas yra susijęs su atitinkamu centru;
4. dokumentų nuorodos į kitus dokumentus yra paverčiamos į sąsajas, kurios susieja atitinkamus centrus, kurie buvo sukurti transformuojant dokumentų identifikacinius laukus;
5. vidiniai dokumentai (dokumentai, kurie yra kito dokumento viduje) yra paverčiami į palydovus, kurie yra prijungiami prie tėvinio elemento, taip pat yra įvykdomos 2–4 taisyklės.

Vadovaujantis nurodytomis taisyklėmis pateikti studento ir jo kontaktų duomenys 1.12 pav. yra išskaidomi iki *Data Vault* duomenų modelio matomo 1.13 pav. [29].

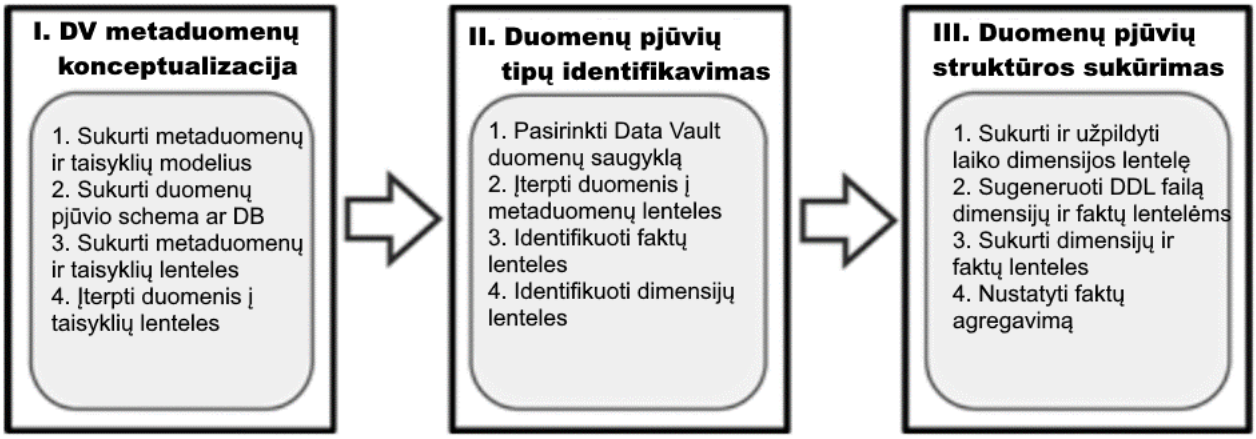


1.12 pav. Studento ir jo kontaktų *JSON* dokumentų pavyzdys [29]

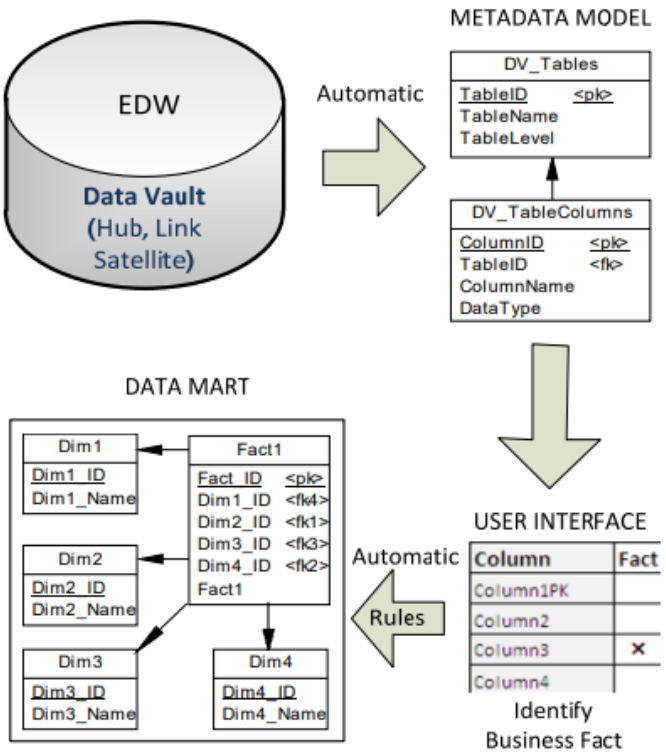


1.13 pav. Studento ir jo kontaktų palydovai, centrai ir sąsaja po įvykdytų transformavimo taisyklių [29]

Kitas svarbus duomenų saugyklos aspektas – duomenų pateikimas reikiamu pjūviu naudotojui. Duomenų pjūvis (angl. *data mart*) – tai naudotojui aktualūs, agreguoti duomenys, kurių analizė, įvairių užklausų vykdymas, filtravimas ir rikiavimas yra patogesnis ir vykdomas daug greičiau negu šias operacijas vykdant tiesiogiai su visais duomenų saugyklos duomenimis. Šios srities parengimas taip pat stipriai priklauso nuo dalykinės srities ir jo sudarymui egzistuoja įvairios metodikos. [30] autoriai pateikia trijų etapų metodiką šiai sričiai sudaryti (žr. 1.14 pav. ir 1.15 pav.).



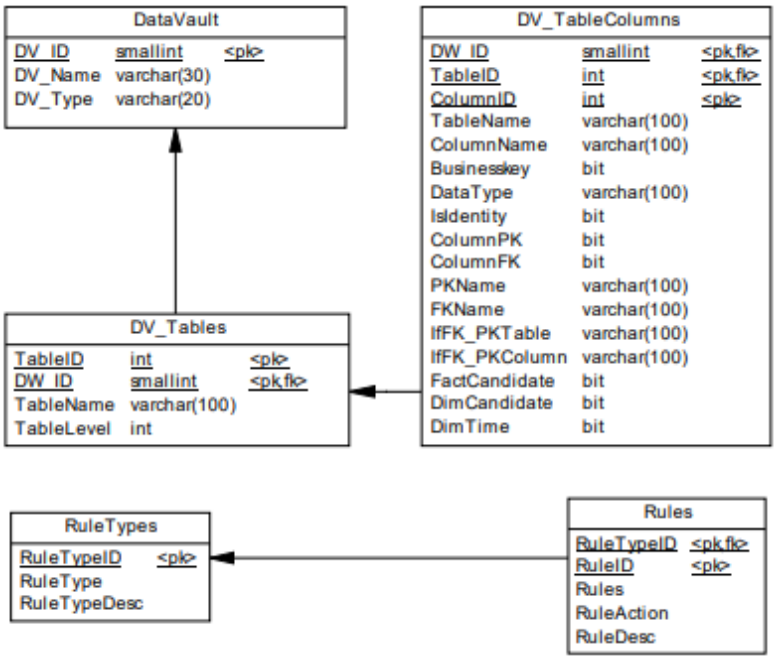
1.14 pav. Duomenų pjūvio sudarymo metodikos schema [30]



1.15 pav. Duomenų pjūvio metodikos vizualizacija [30]

Pirmas etapas – *Data Vault* duomenų saugyklos metaduomenų apsibrėžimas. Šio etapo metu autoriai teigia, kad pirmiausia reikia susidaryti *Data Vault* naudojamų lentelių metaduomenų modelį (žr. 1.16 pav.). Šis modelis aprašo duomenų saugykloje esančias lenteles ir jos atributus: pavadinimą,

pirminius ir išorinius raktus, verslo raktus ir pan. Taip pat yra sudaromas ir taisyklių modelis pagal kurį bus nusprendžiama ar lentelė bus transformuojama duomenų pjūvyje į faktų ar dimensijų lentelę.



1.16 pav. *Data Vault* duomenų saugyklos metaduomenų ir taisyklių modelis [30]

Pirmo etapo antras žingsnis – duomenų pjūvio schemos (ar duomenų bazės) sukūrimas. Ši schema saugos duomenis apie duomenų pjūvio struktūrą. Sekantis žingsnis – sumodeliuotų metaduomenų ir taisyklių lentelių sukūrimas. Paskutiniu pirmo etapo žingsniu yra įrašomos taisyklės į sukurtas lenteles.

Antras etapas – duomenų pjūvio tipų nustatymas. Šio etapo metu pirmiausia yra pasirenkama atitinkama duomenų saugykla, tada į sukurtą metaduomenų modelį yra įkeliami saugyklos metaduomenys. Trečiu žingsniu yra identifikuojamos faktų lentelės. Autorių teigimu šios lentelės gali būti identifikuojamos rankiniu būdu arba automatiniu būdu panaudojant sukurtą taisyklių lentelę. Paskutiniu šio etapo žingsniu yra nurodomos dimensijų lentelės, kurios taip pat gali būti nurodomas rankiniu ar automatiniu būdu.

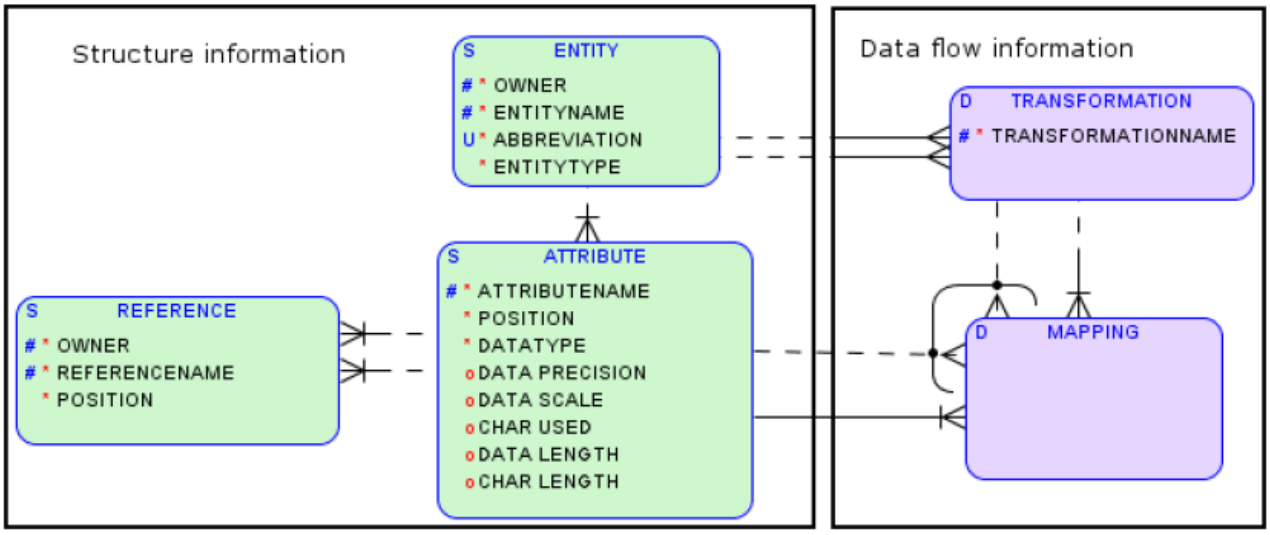
Paskutinis trečias etapas – duomenų pjūvio struktūros sukūrimas. Pirmiausia, sukuriama ir duomenimis užpildoma laiko dimensijos lentelė. Tada yra parengiami *DDL* failai faktų ir dimensijų lentelėms ir šie failai įvykdomi sukuriant pirmiausia dimensijų lenteles, o po to faktų lenteles. Galiausiai, yra aprašomos duomenų agregavimo funkcijos faktų lentelėms.

Ši trijų etapų metodika leidžia parengti reikalingą duomenų pjūvių struktūrą, kuri po to yra užpildoma *Data Vault* duomenimis ir gali būti naudojama suinteresuotų naudotojų.

Dar vienas svarbus aspektas – *Data Vault* saugyklos sudarymo procesų automatizavimas. *Data Vault* sukūrimas reikalauja parengti duomenų modelius, pridėti ir sukonfigūruoti duomenų šaltinius, parengti duomenų pjūvius ir pan. Tai yra sudėtingi ir daug laiko reikalaujantys procesai, todėl yra aktualu kaip įmanoma labiau juos automatizuoti ir paspartinti. [31] šaltinio autoriai pateikia metodiką kaip būtų galima automatizuoti *Data Vault* saugyklos duomenų modelio sudarymo ir atnaujinimo procesą.

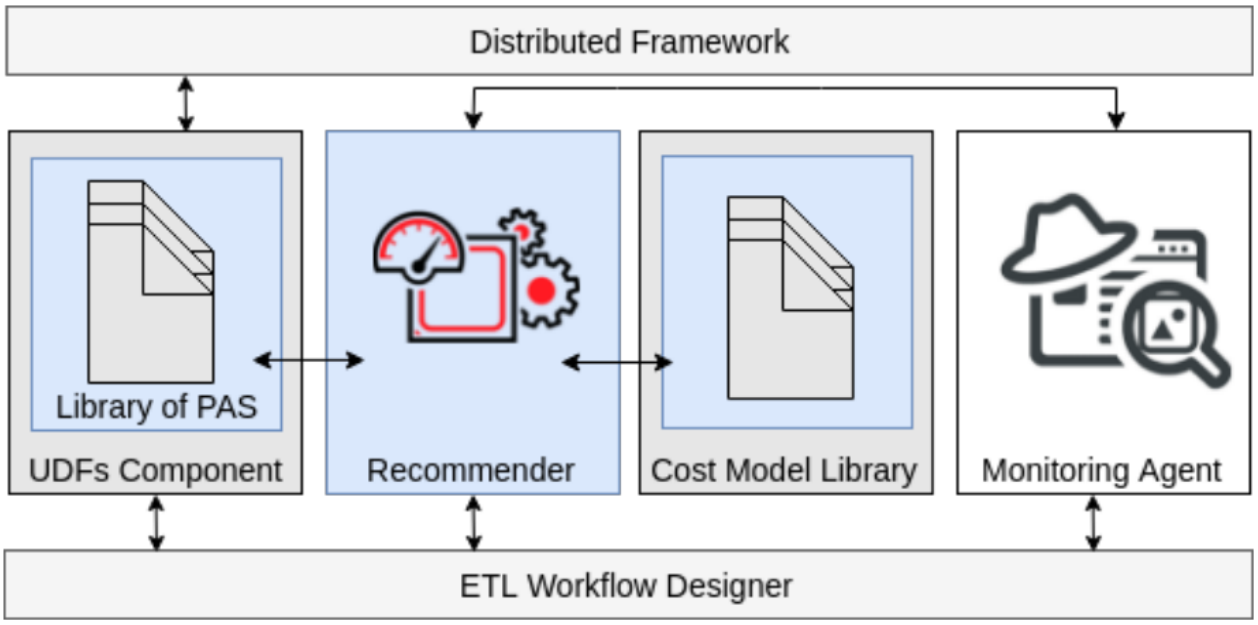
Automatizavimo procesui realizuoti autoriai pateikia kelis etapus kaip atitinkamas duomenų struktūras (šaltinio pateikiamu atveju – reliacinės duomenų bazės lenteles) paversti *Data Vault* duomenų modelį atitinkančiomis lentelėmis.

Pirmiausia, autoriai nurodo, kad reikia susidaryti transformacijų duomenų modelį (žr. 1.17 pav.). Šios transformacijos turi parodyti, transformavimo ryšius iš vienų struktūrų/esybių į kitas struktūras/esybes. Šis modelis yra parengiamas rankiniu būdu naudojant modeliavimo įrankį ir po to užpildomas reikiama informacija. Duomenų šaltinio struktūrų informacija gali būti surenkama iš šaltinio (duomenų bazės) metaduomenų, o *Data Vault* duomenų modelio struktūros sudaromos pritaikant *Data Vault* struktūrų formavimo taisykles. Sudarius struktūros informaciją yra sudaroma transformacijų informacija. Ši informacija yra parengiama susiejant atitinkamas šaltinio (angl. *source*) ir tikslo (angl. *target*) struktūras ryšiais bei sudarant jų atributų sąryšius. Suformavus duomenų transformacijų modelį, norimos duomenų lentelės gali būti sugeneruojamos suformuojant atitinkamas *DDL* lentelių sukūrimo ir stulpelių įterpimo užklausas. Tačiau, autoriai visą šią metodiką daugiau pateikia tik kaip galimo būsimo sprendimo pagrindimą, o ne kaip realų įgyvendintą automatizavimo procesą. Deja, bet didžioji dalis *Data Vault 2.0* automatizacijos sprendimų yra komerciniai produktai, kurie neatskleidžia savo automatizacijos proceso eigos ir metodų.



1.17 pav. Duomenų struktūrų transformacijų modelis [31]

Atliekant duomenų transformacijas taip pat svarbu yra parengti ne tik teisingus, bet ir efektyvius *ETL* duomenų įkėlimo ir transformavimo procesus. Šiam procesui automatizuoti [13, 14] autoriai siūlo naudoti *ETL* procesų įvertinimo karkasą. Šis karkasas pagal naudotojo nurodytas našumo metrikas pasitelkdamas mašininio mokymosi algoritmą įvertiną *ETL* procesą. Apmokimui ir įvertinimui algoritmas naudoja istorinius duomenis ir naudotojo nurodytas metrikas. Karkaso architektūra susideda iš 4 pagrindinių komponentų (žr. 1.18 pav.): *UDFs*, *Recommender*, *Cost Model Library* ir *Monitoring Agent*. *UDFs* (angl. *user defined functions)* komponentas skirtas aprašyti nestandartines funkcijas, kurios nėra pateikiamos *ETL* variklyje. *Recommender* komponentas atsakingas už *ETL* proceso optimizaciją pasitelkiant mašininio mokymo algoritmą naudojantį naudotojo nurodytus rodiklius ir istorinę informaciją. *Cost Model Library* komponentas skirtas aprašyti naudotojo siekiamas našumo metrikas, o *Monitoring Agent* komponentas skirtas surinkti istorinę informaciją apie *ETL* procesų vykdymo istoriją, jų sunaudotus resursus ir pan.



1.18 pav. *ETL* procesų įvertinimo karkaso architektūra [13, 14]

Taigi, kaip galima pastebėti, egzistuoja keletas metodikų ir sprendimų, detaliau analizuojančių vieną ar kitą duomenų saugyklos sudarymo proceso aspektą. Todėl, siekiant įvertinti kiekvieno iš įvardintų sprendimų pranašumus ir trūkumus, sudaryta lyginamoji analizė, pateikiama 4 lentelėje.

4 lentelė. *Data Vault 2.0* sudarymo metodikų palyginimas

	***NoSQL document translation to Data Vault* [29]**	***Data mart design from a Data Vault* [30]**	***Automating Transformations in Data Vault* [31]**	***ETL Framework* [13, 14]**	**Metrikų saugyklos metodika**
Paskirtis	Duomenų įkėlimui iš duomenų šaltinių	Duomenų pjūvių sudarymui	Transformacijai iš neapdorotų duomenų į duomenų saugyklos modelį	Duomenų saugyklos optimizacijai	Metrikų saugyklos sudarymui
Aprašo duomenų įkėlimo metodiką	X	-	-	-	X
Aprašo naudojamą duomenų modelį	X	X	X	-	X
Aprašo duomenų pjūvių sudarymo metodiką	-	X	-	-	X
Naudoja automatizuotus procesus	-	-	X	X	X
Aprašo metrikų rinkimo procesą	-	-	-	X	X
Naudoja mašininio mokymosi algoritmus	-	-	-	X	-

	***NoSQL document translation to Data Vault* [29]**	***Data mart design from a Data Vault* [30]**	***Automating Transformations in Data Vault* [31]**	***ETL Framework* [13, 14]**	**Metrikų saugyklos metodika**
Naudoja *Data Vault 2.0* duomenų saugyklą	X	X	X	-	X

Kaip galima pastebėti, šiame darbe kuriama metodika vienintelė aprašo visą metrikų saugyklos sudarymo procesą – nuo metrikų surinkimo ir įkėlimo, iki duomenų modelio sudarymo ir duomenų pjūvių pateikimo galutiniams naudotojams.

1.8. Technologijų lyginamoji analizė

Darbo uždaviniams įgyvendinti, reikia sukurti ir ištestuoti duomenų saugyklą. Duomenų saugyklos sukūrimui gali būti pasitelkiamos kelios skirtingos sistemos, turinčios savų privalumų ir trūkumų.

Pirmasis ir lengviausiai prieinamas variantas būtų duomenų saugykla realizuota *PostgreSQL* sistemoje [32]. *PostgreSQL* yra atviro kodo, nemokama sistema, dėl ko ji yra labai tinkama nedideliems, eksperimentiniams projektams. *PostgreSQL* palaiko ne tik struktūrizuotus duomenis, bet taip pat ir pusiau struktūrizuotus, ir visiškai nestruktūrizuotus duomenis. Tačiau, pritaikyti sistemą dideliems projektams yra gana sudėtinga, reikia atlikti daug rankinio konfigūravimo. Norint pasinaudoti debesijos paslaugomis, reiktų sistemą įsidiegti ir susikonfigūruoti vienoje iš debesijos paslaugų platformų – iš anksto paruoštos galimybės nėra.

Kitas variantas būtų *Snowflake* platforma [33]. Ši platforma išsiskiria savo dideliu greičiu. Skirtingai, nei *PostgreSQL*, ji iš karto teikia galimybę kurti duomenų saugyklą debesijos platformoje bei gali būti stipriai plečiama. Tačiau, ji reikalauja patyrusio eksperto žinių, siekiant tinkamai sukonfigūruoti duomenų architektūrą, bei taip pat yra apmokestinama už sunaudotus saugojimo ir skaičiavimo resursus.

Trečiasis variantas būtų *BigQuery* platforma [34]. Tai vienas iš *Google* įmonės produktų, dėl ko labai gerai tinkamas, jei yra integruojamas į kitų jau naudojamų *Google* produktų ekosistemą. Teikia debesijos paslaugas, automatinį srauto paskirstymą. Labai tinkama mašininio mokymo sistemų duomenims kaupti ir analizuoti. Tačiau, taip pat yra apmokestinama už sunaudotus saugojimo resursus ir įvykdytas duomenų užklausas.

Visų trijų sistemų apibendrintas palyginimas pateikiamas 5 lentelėje.

5 lentelė. Duomenų saugyklų sistemų lyginamoji analizė [32]–[34]

Kriterijai	***PostgreSQL***	***Snowflake***	***BigQuery***
Galimybė realizuoti *Data Vault* duomenų saugyklą	+	+	+
Atviro kodo	+	-	-
Palaikomi duomenų formatai	Struktūrizuoti, pusiau struktūrizuoti ir nestruktūrizuoti	Struktūrizuoti, pusiau struktūrizuoti ir nestruktūrizuoti	Struktūrizuoti, pusiau struktūrizuoti ir nestruktūrizuoti
Debesijos paslauga	-	+	+

Kriterijai	***PostgreSQL***	***Snowflake***	***BigQuery***
Srauto paskirstymas (angl. *load balancing*)	Komplikuotas didesnėms sistemoms	Automatinis	Automatinis
Plečiamumas	Sudėtingas	Didelis	Didelis
Greitis	Lėčiausias	Greičiausias	Vidutinis
Palaikymas	Reikalaujantis daug rankinio konfigūravimo	Sudėtingas, reikalaujantis patyrusių ekspertų	Didžiąja dalimi automatizuotas, supaprastintas
Kaina	Nemokama	Mokestis už sunaudotus resursus ir skaičiavimo resursus	Mokestis už sunaudotus resursus ir įvykdytas užklausas

Kaip galima matyti iš atlikto duomenų saugyklų palyginimo, *Snowflake* ir *BigQuery* platformos yra pranašesnės ir tinkamesnės realizuoti *Data Vault* duomenų saugyklą vykdomos veiklos tikslams pasiekti. Tačiau, *PostgreSQL* yra nemokama, lengvai prieinama ir patogi sistema mažos apimties eksperimentiniams projektams.

Duomenų saugyklos kūrimui automatizuoti gali būti pasitelkiami įvairūs sprendimai, kurie daugiausiai yra realizuoti internetinių puslapių pagrindu [5, 6]. Todėl svarbu palyginti kokiomis technologijomis jie gali būti realizuoti.

SpringBoot [35] ir *React* [36] karkasų rinkinys yra vienas iš galimų sprendimų tokioms sistemoms realizuoti. *SpringBoot* yra *Java* programavimo kalba grįstas karkasas, skirtas serverinei daliai realizuoti. *React* yra *JavaScript* programavimo kalba grįstas karkasas, skirtas klientinei daliai realizuoti. Abu karkasai turi didelį populiarumą ir naudotojų skaičių, todėl teikia plataus spektro funkcijas reikalingas sistemai realizuoti. *SpringBoot* karkasas teikia autentifikacijos, autorizacijos, sesijos valdymo, duomenų validacijos funkcionalumą. Abiejų karkasų rinkinys leidžia pritaikyti *MVC* (angl. *model-view-controller*) architektūrą. Abu karkasai yra atviro kodo ir dėl to lengvai naudojami.

ASP .NET Core [37] yra kitas alternatyvus karkasas leidžiantis kurti internetines sistemas. Šis karkasas yra grįstas *C#* programavimo kalba ir leidžia realizuoti tiek serverinės, tiek klientinės dalies funkcionalumą. Šis karkasas yra greičiausias iš lyginamų. *ASP .NET Core* taip pat teikia iš anksto parengta autentifikacijos, autorizacijos, sesijos valdymo, duomenų validacijos funkcionalumą. Karkaso struktūra organizuojama pagal *MVC* architektūrą. Kadangi karkasas yra kuriamas *Microsoft* kompanijos, jis labai gerai integruojasi į šios įmonės produktų ekosistemą. Tačiau, karkasas iš visų lyginamų variantų yra mažiausiai populiarus.

Laravel [38] karkasas yra trečia alternatyva. Šis karkasas yra grįstas *PHP* programavimo kalba ir leidžia realizuoti tiek serverinę, tiek klientinę dalis. Karkasas organizuojamas pagal *MVC* architektūros principus. Teikia autentifikacijos, autorizacijos, sesijos valdymo, duomenų validacijos funkcijas. Tačiau, lyginant su kitais karkasais yra pats lėčiausias.

Apibendrintas internetinių karkasų palyginimas pateikiamas 6 lentelėje.

6 lentelė. Internetinių karkasų lyginamoji analizė [35]–[51]

Kriterijai	***SpringBoot* ir *React***	***ASP .NET Core***	***Laravel***
Programavimo kalba	*Java* (*SpringBoot*) ir *JavaScript* (*React*)	*C#*	*PHP*

Kriterijai	***SpringBoot* ir *React***	***ASP .NET Core***	***Laravel***
Karkaso tipas	*SpringBoot* – serverinė dalis, *React* – klientinė dalis	Serverinę ir klientinę dalis apimantis karkasas	Serverinę ir klientinę dalis apimantis karkasas
Autentifikacija	+	+	+
Autorizacija	+	+	+
Sesija	+	+	+
Palaikomi duomenų bazių tipai	Reliacinės / Nereliacinės	Reliacinės / Nereliacinės	Reliacinės / Nereliacinės
MVC	+	+	+
Duomenų validacija	+	+	+
Greitis [39]	Vidutinis	Greičiausias	Lėčiausias
Atviro kodo	+	+	+
Populiarumas *Github* platformoje (suinteresuotų naudotojų kiekis) [40–43]	72 tūkst. *SpringBoot* ir 221 tūkst. *React*	47 tūkst.	76 tūkst.
Populiarumas *Stack Overflow* platformoje (pagal procentą visų užduodamų klausimų platformoje) [44]	*React* ~ 5%, *SpringBoot* ~ 1,8% visų platformos klausimų	~0.5% visų platformos klausimų	~1,2% visų platformos klausimų

Atlikta lyginamoji internetinių karkasų analizė parodė, kad visos trys alternatyvos yra tinkamos realizuoti duomenų saugyklos automatizacijos įrankius. Visi karkasai suteikia reikiamas priemonės šiam tikslui pasiekti, tačiau vieni įrankiai yra labiau populiaresni. *SpringBoot* ir *React* karkasų rinkinys turi didžiausias ir aktyviausias bendruomenės dėl to iškilusių klausimų ir problemų sprendimas gali būti greitesnis ir lengvesnis.

Internetinio sprendimo duomenims saugoti reikalinga duomenų bazė. Pasirinkta apžvelgti reliacines duomenų bazių valdymo sistemas, kadangi automatizuotų sprendimų transformacijoms saugoti yra svarbesnė duomenų struktūra negu duomenų išrinkimo greitis, kuris būtų didesnis nereliacinėse duomenų bazėse.

Trys palygintos duomenų bazių valdymo sistemos yra: *PostgreSQL* [32], *MariaDB* [45] ir *Oracle* [46]. Visos šios sistemos teikia pagrindines svarbiausias reliacinių duomenų bazių valdymo funkcijas (žr. 7 lentelė), kaip indeksavimą, trigerius, funkcijas, duomenų replikavimą, užtikrina *ACID* principus. Turi didelį skirtingų tvarkyklių palaikymą.

7 lentelė. Duomenų bazių valdymo sistemų lyginamoji analizė [32, 45, 46]

Kriterijai	***PostgreSQL***	***MariaDB***	***Oracle***
Reliacinė duomenų bazių valdymo sistema	+	+	+
Atviro kodo	+	+	-
Realizavimo kalba	*C*	*C* ir *C++*	*C* ir *C++*
Indeksavimas	+	+	+
SQL palaikymas	+	+	+

Kriterijai	***PostgreSQL***	***MariaDB***	***Oracle***
Tvarkyklės	*JDBC, ODBC, npgsql, HDBC*	*ADO.NET, JDBC, ODBC*	*JDBC, ODBC, ODP.NET, Oracle Call Interface*
Trigeriai	+	+	+
Transakcijos	+	+	+
Funkcijos	+	+	+
ACID	+	+	+
Replikavimas	+	+	+

Visos palygintos duomenų bazių valdymo sistemos yra tinkamai pritaikytos išsaugoti informaciją reikalingą automatizacijos įrankio veikimui. *PostgreSQL* ir *MariaDB* pranašumas yra tai, kad jos abi yra atviro kodo ir nemokamos. *PostgreSQL* sistema tai pat yra patogus pasirinkimas, kai duomenų saugyklai yra naudojama *PostgreSQL* sistema, kadangi tai sumažina naudojamų technologijų aibę. Taigi, galutiniam sprendimui realizuoti nuspręsta pasitelkti *PostgreSQL* sistemą duomenų saugyklai ir duomenų bazei, o automatizacijos įrankį sukurti pasitelkiant *SpringBoot* ir *React* internetinius karkasus.

1.9. Siekiamo sprendimo apibrėžimas

Šiame darbe siūloma metodika, kuri pateikia pilną ir išsamų metrikų saugyklos sudarymo procesą *Data Vault 2.0* pagrindu sudarytai duomenų saugyklai. Metodika aprašo metrikų šaltinių pasirinkimo ir pridėjimo į duomenų saugyklą eigą, metrikų saugyklos sukūrimą *Data Vault* saugykloje bei metrikų duomenų atvaizdavimą, sudarant duomenų pjūvius. Taip pat metodikos veiksmingumo pagrindimui atliekamas eksperimentas, kurio metu sukuriama pavyzdinė metrikų saugykla, naudojant realios įmonės serverių veiklos duomenis.

1.10. Analizės išvados

1. Atlikta *Data Vault 2.0* analizė parodė, kad *Data Vault 2.0* duomenų saugykla yra grindžiama trijų sričių architektūra, siekiant efektyviai išnaudoti ir apdoroti gaunamus duomenis, naudojant specifinį *Data Vault* duomenų modelį. Šis duomenų modelis daugiausiai orientuotas į veiklos procesus, dėl to yra lengvai keičiamas, plečiamas ir nesunkiai prisitaikantis prie duomenų šaltinių pokyčių.
2. Atlikta metrikų rinkimo įrankių analizė parodė, kad egzistuoja daug sprendimų, skirtų stebėti serverio ir duomenų saugyklų darbą. Tačiau, visi įrankiai turi skirtingus metrikų rinkinius, kurie dažniausiai apima šias pagrindines metrikų kategorijas: procesoriaus darbo, atminties naudojimo, disko operacijų, internetinio ryšio naudojimo ir temperatūros metrikas.
3. Atlikta metrikų taksonomijos analizė parodė, kad dabartiniu metu egzistuoja daug skirtingų metrikų kategorijų, tačiau jų klasifikavimas nėra standartizuotas ir stipriai kinta nuo tiriamos dalykinės srities. Analizė parodė, kad dažniausiai yra nagrinėjamos trys kompiuterinių resursų metrikų kategorijos: energijos metrikos, internetinio ryšio metrikos ir sistemos darbo metrikos. Sistemos darbo metrikos taip pat dažnai yra skaidomos pagal skirtingų resursų panaudojamumą į procesoriaus darbo metrikas, operatyviosios atminties darbo metrikas ir išorinės atminties darbo metrikas.
4. Atlikta duomenų saugyklų sudarymo metodikų lyginamoji analizė parodė, kad daugelis metodikų specializuojasi vieno duomenų saugyklos aspekto kūrime ir nė viena metodika neaprašo metrikų saugyklos sudarymo *Data Vault 2.0*. Nustatyta, kad, siekiant sukurti metrikų saugyklą, reikės

aprašyti metrikų surinkimo procesą, duomenų įkėlimo procesą, duomenų modelį ir duomenų pjūvių sudarymo eigą.

5. Atlikta technologijų lyginamoji analizė parodė, kad egzistuoja keletas įrankių ir sistemų tinkamų metrikų saugyklos sudarymo įrankiui įgyvendinti. Šiame darbe duomenų saugyklai ir duomenų bazei kurti ir valdyti buvo pasirinkta *PostgreSQL* sistema dėl savo lengvo prieinamumo, tinkamumo ir paprastumo realizuoti duomenų saugyklas. Įrankiui kurti buvo nuspręsta panaudoti *SpringBoot* ir *React* karkasus dėl jų tinkamumo ir didžiausio populiarumo lyginant su kitais karkasais. Tačiau, tiek duomenų saugyklai naudoti, tiek įrankiui kurti galima būtų pasitelkti ir alternatyvias technologijas, kurios užtikrintų *Data Vault* technologijos pritaikymą.

2. *Data Vault* metrikų saugyklos sudarymo metodikos reikalavimų specifikacija, projektas ir formalus aprašas

2.1. *Data Vault* metrikų saugyklos sudarymo metodikos reikalavimai

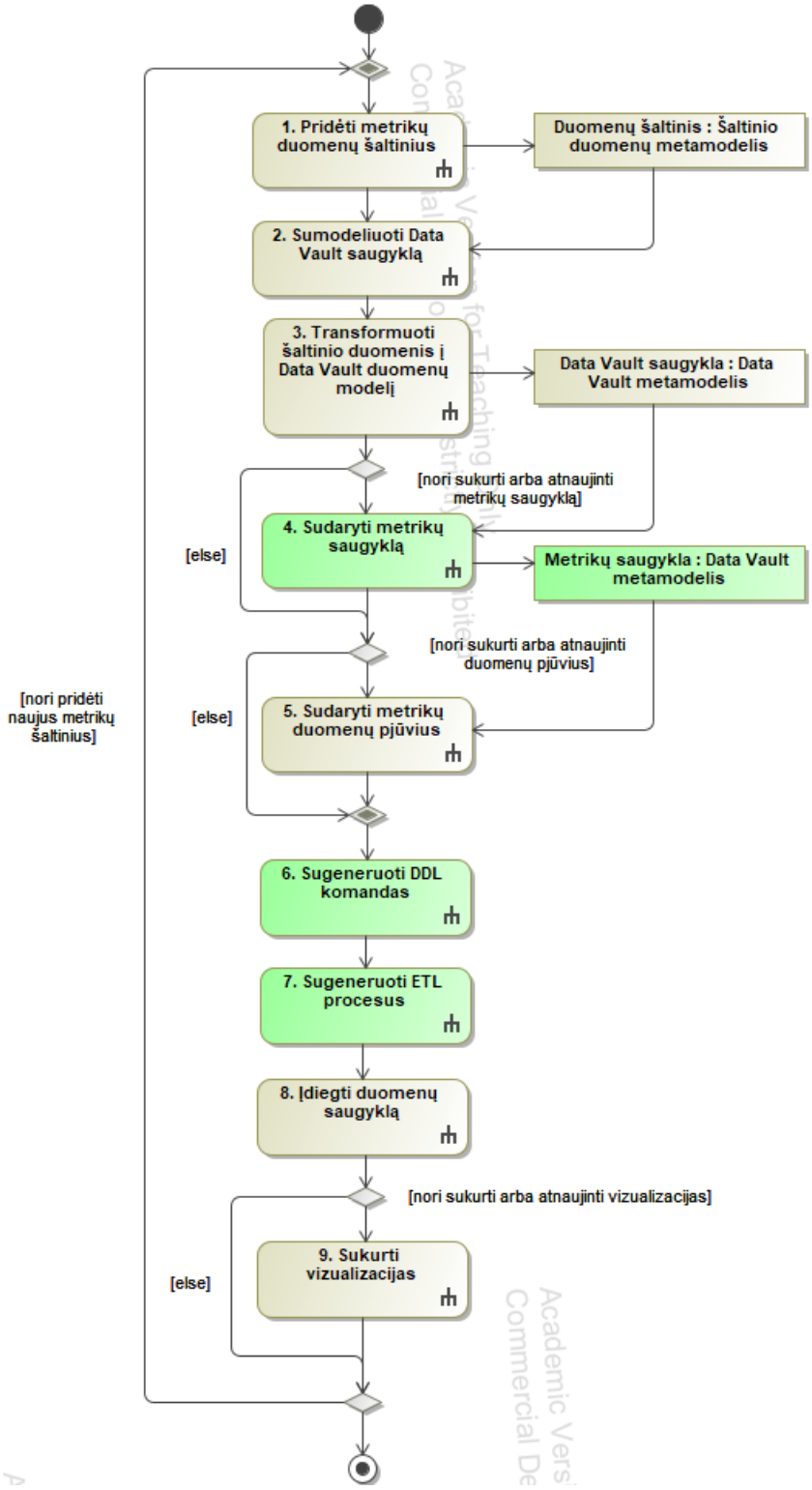
Data Vault metrikų saugyklos pagrindinė panaudojimo paskirtis palengvinti kompiuterinių resursų metrikų stebėjimą ir kontrolę. Šiam tikslui pasiekti svarbu aprašyti kriterijus, kuriuos turi tenkinti metrikų saugykla, jos metodika ir kokios savybės būdingos jai. Pagrindinės metrikos saugyklos charakteristikos ir iš jų kylantys metodikos reikalavimai yra pateikiami 8 lentelėje. Pagal šiuos reikalavimus yra parengta metrikų saugyklos sudarymo eiga, kuri yra aprašyta 2.1 paveiksle pateikta veiklos diagrama.

Procesas susideda iš kelių svarbių žingsnių. Viskas prasideda nuo metrikų duomenų šaltinių pridėjimo. Šiuo žingsniu naudotojas nurodo, iš kokių duomenų šaltinių turi būti surenkami duomenys. Kitas žingsnis – neapdorotų duomenų sumodeliavimas ir transformavimas į *Data Vault* duomenų modelį. Šiuo žingsniu surenkami neapdorotų duomenų struktūrų metaduomenys ir transformuojami į *Data Vault* modelio duomenų elementus. Naujas ar atnaujintas modelis išsaugomas antrojoje duomenų saugyklos srityje – *Data Vault* sluoksnyje. Trečias žingsnis – metrikų saugyklos sudarymas. Šis žingsnis susideda iš metrikų duomenų struktūrų (centrų, sąsajų ir palydovų) saugojamų metrikų saugykloje nurodymo, agregavimo funkcijų ir išskaičiuojamų funkcijų aprašymo. Ketvirtas žingsnis – duomenų pjūvio (angl. *data mart*) sudarymas. Šiame žingsnyje yra parengiamas duomenų pjūvis skirtas atvaizduoti norimą duomenų informaciją. Paskutiniais žingsniais yra sugeneruojamas *DDL* ir *ETL* kodas. *DDL* kodas yra skirtas sukurti norimas duomenų struktūras pasirinktoje duomenų saugyklos platformoje, įvykdant šį kodą yra sukuriamos naujos duomenų lentelės, atnaujinamos esančios lentelės pridedant naujus stulpelius, sukuriami ir pridedami ryšiai. Sugeneruoti *ETL* procesai yra skirti užpildyti duomenų saugyklą reikiamais duomenimis iš duomenų šaltinių tam tikrais laiko intervalais. Taip pat *ETL* procesai aprašo ir duomenų įkėlimo bei transformavimo veiksmus tarp skirtingų duomenų saugyklos sričių. Tokiu būdu užtikrinama, kad į saugyklą patenkantys nauji duomenys bus įkelti, išsaugoti ir apdoroti pagal anksčiau apibrėžtus reikalavimus ir pasieks naudotoją norima duomenų atvaizdavimo forma.

8 lentelė. Metrikų saugyklos charakteristikos

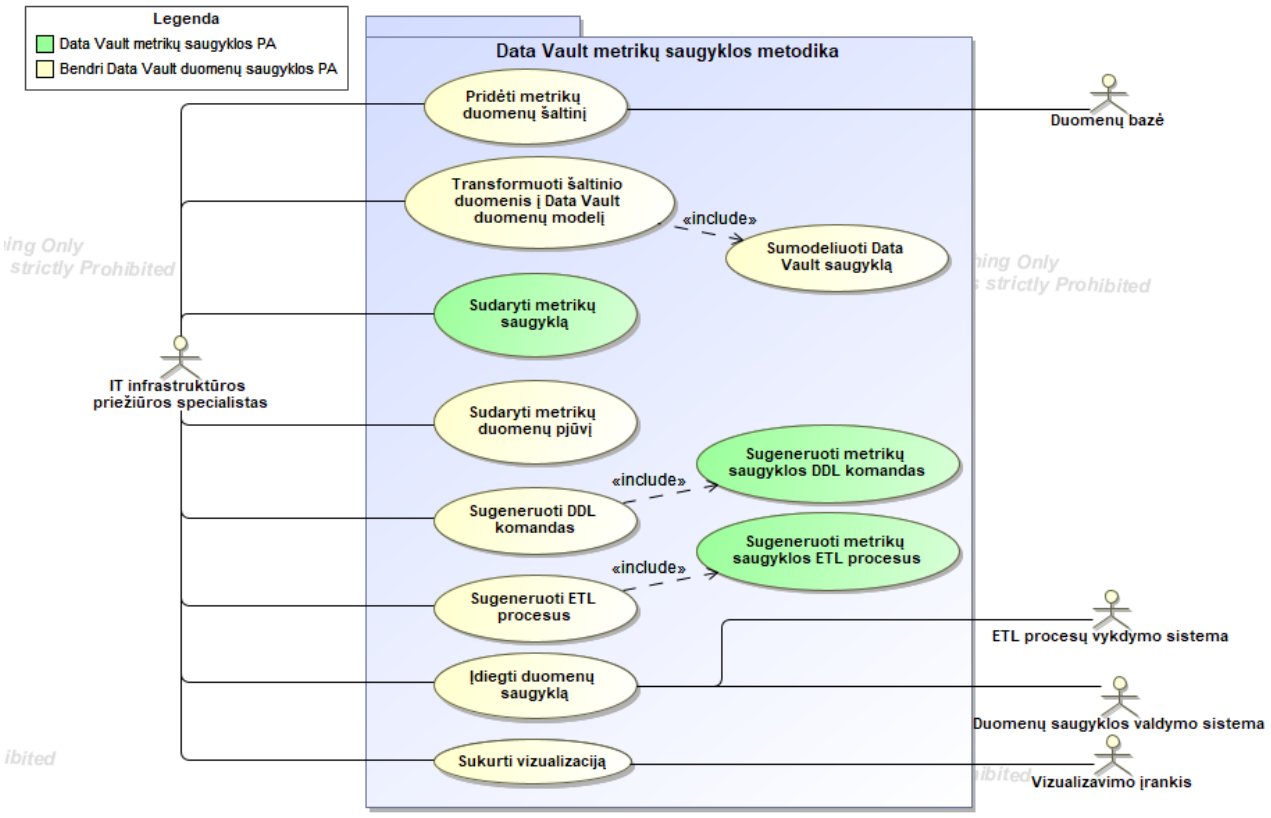
Charakteristika	**Reikalavimas (metodikos žingsnis)**	**Šaltiniai**
Metrikų duomenys surenkami iš kelių skirtingų duomenų šaltinių	*Data Vault* duomenų saugykla turi suteikti galimybę pridėti skirtingus metrikų duomenų šaltinius (1)	[1–3, 27]
Metrikų saugykloje metrikų duomenys saugomi *Data Vault* duomenų metamodelio pavidalu	Pridėti metrikų duomenys turi būti transformuoti iki *Data Vault* duomenų metamodelio struktūros (2-3)	[3, 4, 27–29]
Metrikos yra laiko eilučių duomenys	Metrikų duomenys turi fiksavimo laiko atributą (4)	[19–21, 23–25]
Naudojamos įvairios skaitinės reikšmės: absoliučios reikšmės, santykiai, procentai, skirtumai	Metrikų saugykloje saugomi duomenys gali būti įvairių tipų, daugiausiai skaitiniai duomenų tipai (4)	[3, 14, 19–25]
Metrikai gauti pritaikomos įvairios matematinės funkcijos	Formuojant metrikų saugyklą turi būti galimybė pritaikyti standartines *SQL* kalbos matematines funkcijas (4)	[3, 7, 14, 19–25, 47–49]

Charakteristika	**Reikalavimas (metodikos žingsnis)**	**Šaltiniai**
Fiksuojamos metrikos turi sau būdingas kategorijas	Metrikų saugykloje saugomos metrikos turi būti kategorizuojamos (4)	[19–25]
Metrikoms sudaromi duomenų pjūviai	Metrikų saugyklos duomenims pasiekti turi būti sudaromi duomenų pjūviai (5)	[3, 21, 24, 25, 30]
Data Vault saugyklos pasižymi didele automatizacija dėl kitu atveju intensyvaus rankinio darbo	Metrikų saugyklos struktūros *DDL* kodas turi būti automatiškai sugeneruojamas (6)	[3–6, 31]
Metrikos renkamos nustatytu laiko dažniu	Metrikų duomenų įkėlimui turi būti naudojamos iš anksto suplanuotos automatinės *ETL* funkcijos (7)	[3, 7–13, 15, 17, 31, 50–52]
Metrikos analizuojamos įvairių skirtingų naudotojų iš skirtingų sistemų	Metrikų saugykla turi būti įdiegta *Data Vault* duomenų saugykloje (8)	[1–3, 27]
Metrikoms sudaromos vizualizacijos skirtos duomenų analizei	Šie pjūviai vėliau gali būti panaudojami vizualizacijų sudarymui. (9)	[3, 16–18, 21, 24]



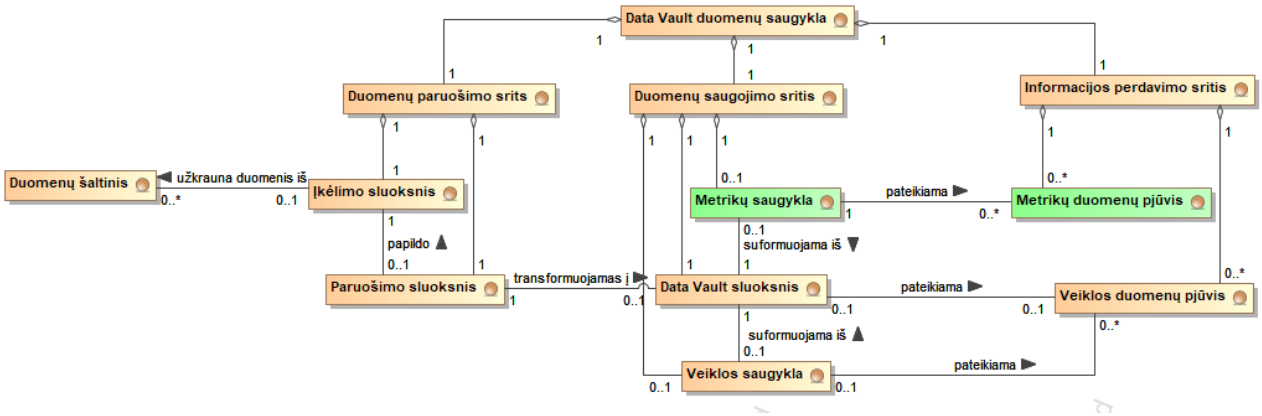
2.1 pav. *Data Vault* metrikų saugyklos sudarymo ir duomenų atnaujinimo procesas

Pagrindiniai aprašyto proceso žingsniai yra *Data Vault* metrikų saugyklos metodikos panaudojimo atvejai, kuriuos atlieka IT infrastruktūros priežiūros specialistas (žr. 2.2 pav.). Kiekvieną kartą specialistas, norėdamas atnaujinti infrastruktūros priežiūros sistemą, turi galėti pridėti naujus duomenų šaltinius, šių šaltinių duomenys turi būti transformuojami, turi būti atnaujinama metrikų saugykla, sudaromi nauji arba atnaujinami esami duomenų pjūviai ir visam tam pasiekti yra sugeneruojamos naujos *DDL* komandos bei *ETL* procesai.



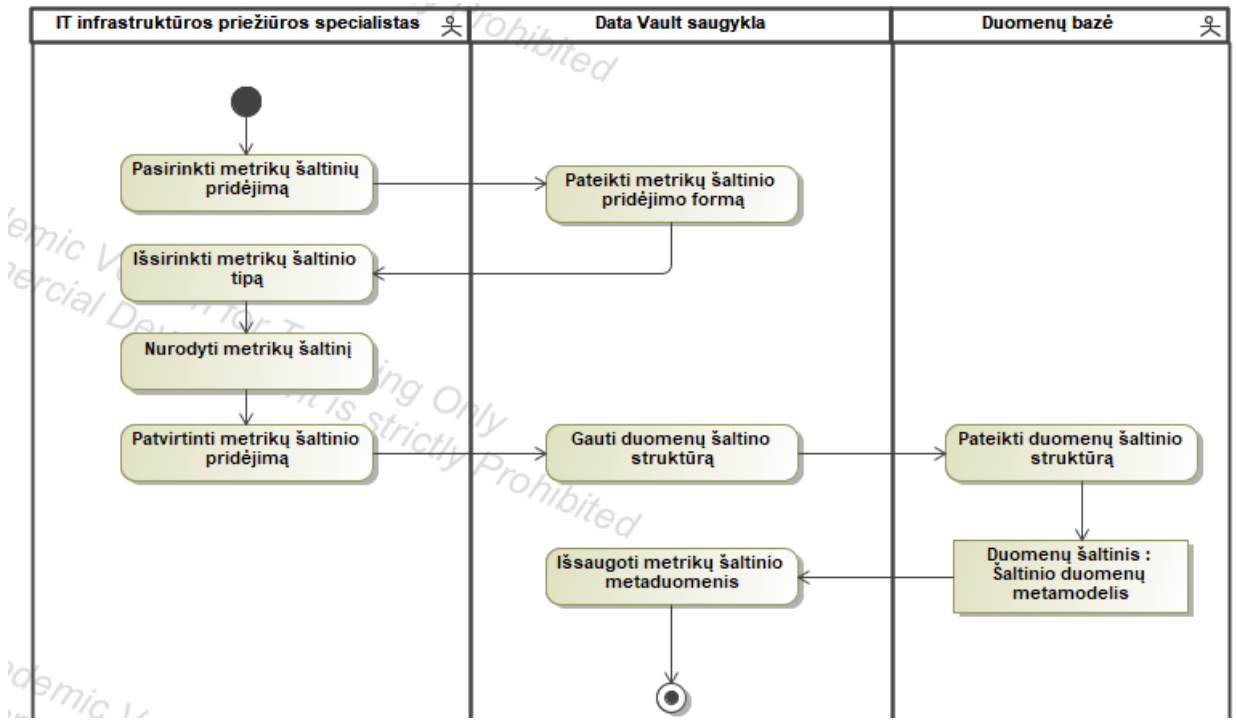
2.2 pav. *Data Vault* metrikų saugyklos metodikos panaudojimo atvejų diagrama

Geriau įsivaizduoti *Data Vault* metrikų saugyklos koncepciją padeda 2.3 paveiksle pateiktas visos *Data Vault* duomenų saugyklos su metrikų saugykla koncepcinis modelis. Į saugyklą duomenys patenka iš duomenų šaltinio, šie duomenys laikinai patenka į įkėlimo sluoksnį, yra papildomi būtina informacija paruošimo sluoksnyje. Tada pagal *Data Vault* duomenų modelio taisykles paruošti duomenys transformuojami ir pastoviam saugojimui išsaugojami *Data Vault* sluoksnyje. Visi šie etapai jau ir dabar yra vykdomi egzistuojančiuose *Data Vault* duomenų saugyklose. Dabartinėse sistemose iš *Data Vault* duomenų sluoksnio yra suformuojama veiklos saugykla, pritaikant įvairiais veiklos taisykles, tada apsibrėžiami naudotoją dominantys informacijos aspektai ir suformuojami veiklos duomenų pjūviai skirti veiklos analitikams. Tačiau, pagal metrikų saugyklos koncepciją metrikoms saugoti yra siūloma sudaryti atskirą metrikų saugyklos komponentą, kuris būtų formuojamas *Data Vault* sluoksnio pagrindu. Šiame komponente būtų saugojama tik su sistemos darbu susijusi metrikų informacija. Galiausiai, naudotojas metrikoms atvaizduoti sudarytų metrikų duomenų pjūvius iš metrikų saugyklos duomenų. Tokiu būdu šie duomenys būtų pasiekiami įvairių vizualizacijų sudarymui, kurios būtų pasiekiamos IT infrastruktūros priežiūros specialistams analizuoti.



2.3 pav. *Data Vault* metrikų saugyklos koncepcinis modelis

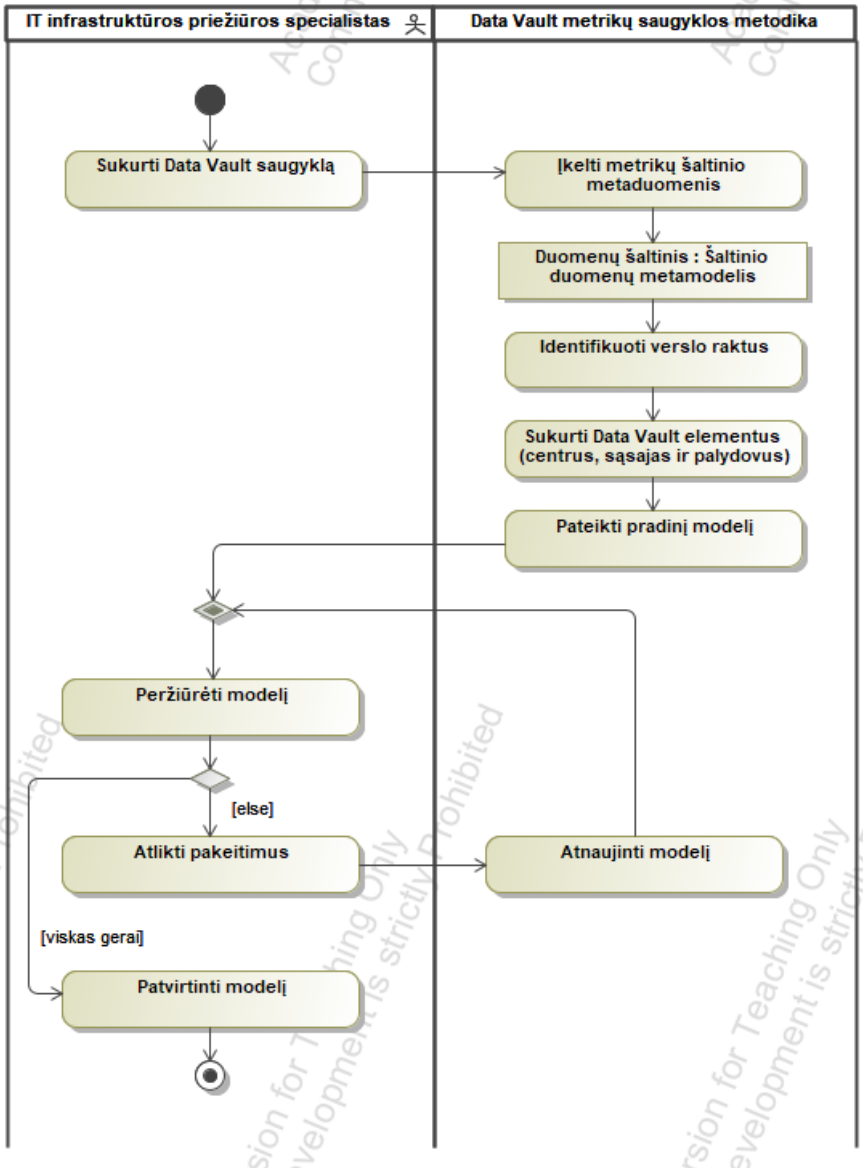
Taigi, kaip galima matyti ir iš koncepcinio modelio ir panaudojimo atvejų diagramos, pirmasis žingsnis metrikų saugyklos kūrime yra duomenų šaltinio pridėjimas. Šis procesas skirtas duomenų saugyklai nurodyti iš kur turėtų būti surenkami duomenys. Pirmiausia turi būti nurodomas prieigos adresas, autentifikacijos ir autorizacijos parametrai, duomenų nuskaitymo dažnumas. Tada turi būti patvirtinami metrikų šaltinio duomenų tipai (tipai gali būti identifikuojami automatiniu būdu iš duomenų šaltinio schemos arba nurodami naudotojo). Įsitikinus, kad duomenų šaltinio parametrai yra sukonfigūruoti teisingai, naudotojas patvirtina šaltinio pridėjimą. Po patvirtinimo sistema išsaugo duomenų šaltinio konfigūraciją ir įkelia metrikų šaltinio duomenų struktūrų metaduomenis. Visą šį procesą aprašo 2.4 paveiksle pateikta metrikų pridėjimo panaudojimo atvejo veiklos diagrama.



2.4 pav. „Pridėti metrikų duomenų šaltinį“ veiklos diagrama

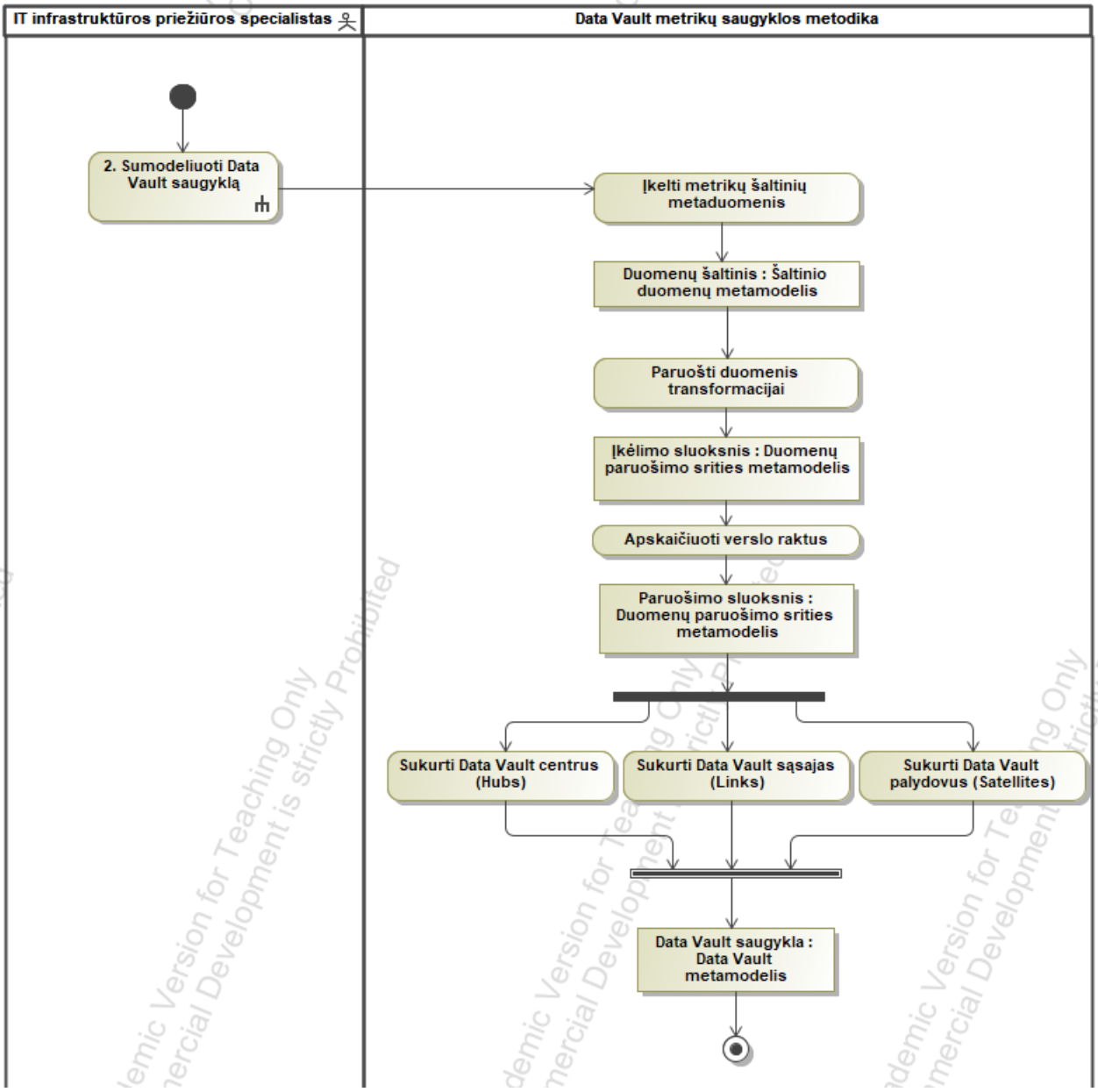
Sekantis žingsnis duomenų saugyklos kūrime yra pridėtų metrikų šaltinių duomenų struktūrų transformavimas į naują modelį sudarytą pagal *Data Vault* duomenų modeliavimo taisykles. Šis žingsnis yra didžiąja dalimi automatizuotas, tačiau reikalauja naudotojo įsikišimo siekiant užtikrinti

veiklos logiką atitinkančią modelio struktūrą. Transformavimui atlikti pirmiausia turi būti parengiamas *Data Vault* saugyklos duomenų modelis. Šio modelio formavimas pradedamas nuo visų metrikų šaltinių duomenų struktūrų metaduomenų įkėlimo. Pirmiausia yra identifikuojami struktūrų verslo raktai pagal kuriuos gali būti identifikuojami atskiri esybių egzemplioriai. Identifikavus verslo raktus yra sukuriami *Data Vault* centrai, kuriuose ir bus saugomi atskiri verslo raktai. Sukūrus centrus yra sukuriamos *Data Vault* sąsajos tarp centrų. Šios sąsajos aprašo kaip yra susijusios atskiros esybės, o sąsaja yra šių dviejų esybių verslo raktų kombinacija. Galiausiai, yra sukuriami *Data Vault* palydovai, kurie gali būti prijungiami tiek prie centrų, tiek prie sąsajų ir skirti saugoti kitus atributus apibūdinančius centrus ir sąsajas, tačiau neskirti jų unikaliam identifikavimui. Kadangi visas šis procesas yra įvykdomas automatiniu būdu reikalinga įsitikinti, kad neįvyko jokie netikslūs duomenų struktūrų išskirstymai ir sujungimai. Tam prieš išsaugant transformuotą modelį naudotojas gali peržiūrėti ir atnaujinti modelį rankiniu būdu pašalindamas netikslumus. Išsprendus visas problemas naudotojas patvirtina modelio transformaciją ir išsaugomas naujas ar atnaujintas duomenų modelis naudojamas antrojoje duomenų saugyklos srityje – *Data Vault* sluoksnyje. Visą šį procesą aprašo 2.5 paveiksle pavaizduota transformacijos panaudojimo atvejo veiklos diagrama.



2.5 pav. „Sumodeliuoti Data Vault saugyklą“ veiklos diagrama

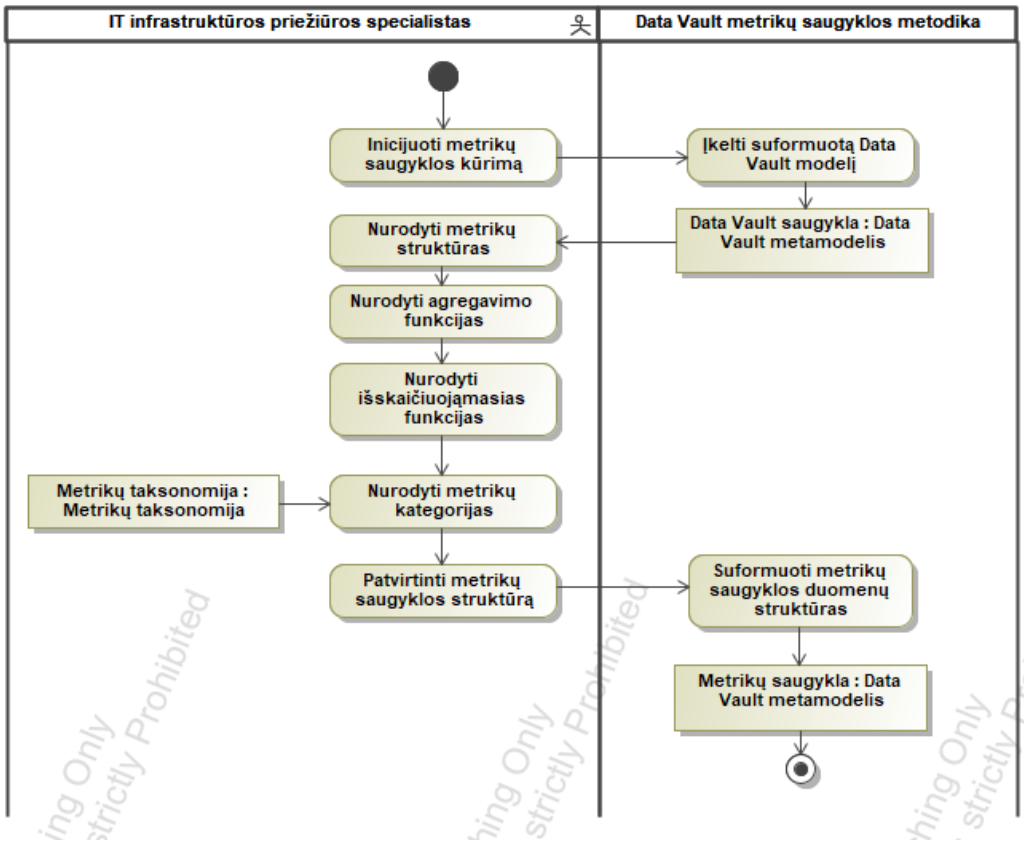
Patvirtinus duomenų modelį pradedamas transformavimo automatizacijos procesas aprašytas 2.6 veiklos diagramoje. Pirmiausia yra įkeliami metrikų šaltinio metaduomenys. Šie metaduomenys tada yra panaudojami suformuojant duomenų įkėlimo sluoksnį, kuriame yra nukopijuojama metrikų šaltinio duomenų struktūra ir papildoma informaciniais atributais (detalesnis aprašymas 2.2 skyriuje). Tada iš duomenų įkėlimo sluoksnio yra suformuojamas duomenų paruošimo sluoksnis, kuris nukopijuoja duomenų įkėlimo struktūrą ir ją papildo verslo raktų apskaičiavimais. Galiausiai, yra suformuojama *Data Vault* saugykla, kurioje iš paruošimo sluoksnyje pateiktų lentelių yra išskiriami *Data Vault* centrai, sąsajos ir palydovai. Visi šie elementai yra įkeliami lygiagrečiai.



2.6 pav. „Transformuoti metrikas į *Data Vault* duomenų modelį“ veiklos diagrama

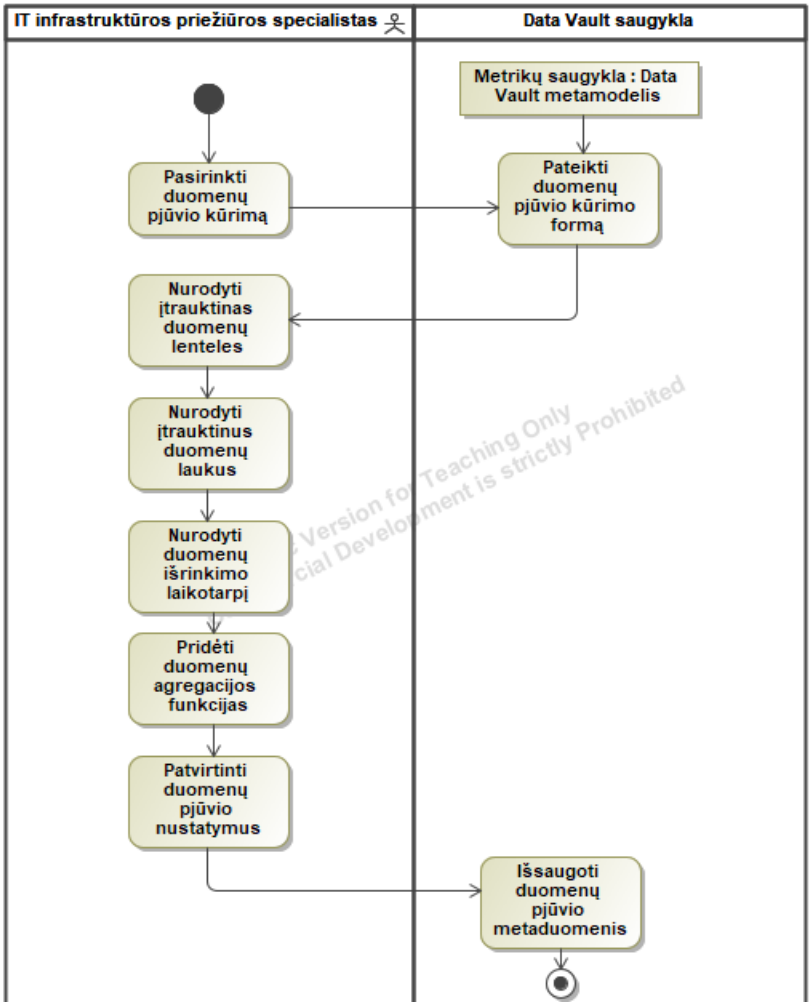
Atlikus duomenų transformaciją ir suformavus Data Vault saugyklą, galima sukurti metrikų saugyklą. Metrikų saugyklos kūrimui pirmiausia reikia nurodyti duomenų struktūras, įtrauktinas į šią saugyklą. Kadangi Data Vault saugykloje saugomi visi duomenys (tiek veiklos duomenys, tiek metrikų duomenys), reikia išsirinkti mus dominančius metrikų struktūrų duomenis. Be pradinių duomenų reikšmių metrikų saugykloje taip pat galimai bus norima saugoti tam tikras iš anksto agreguotas ar

išskaičiuotas reikšmes. Dėl to kuriant metrikų saugyklą galima nurodyti agregavimo ir išskaičiavimo funkcijas. Taip pat sudarant saugyklą reikia nurodyti metrikų kategorijas, kurios padeda nustatyti problemas analizuojant metrikas skirtingomis kategorijomis. Šios kategorijos parenkamos iš pateiktos metrikų taksonomijos ir padeda greičiau identifikuoti problemos šaltinį. Baigus nurodinėti metrikų saugyklos konfigūracijos parametrus patvirtinama metrikų saugyklos struktūra, tada yra suformuojamos reikalingos Data Vault duomenų struktūros bei jos išsaugojamos. Visą šį procesą aprašo 2.7 paveiksle pateikta metrikų saugyklos sudarymo panaudojimo atvejo veiklos diagrama.



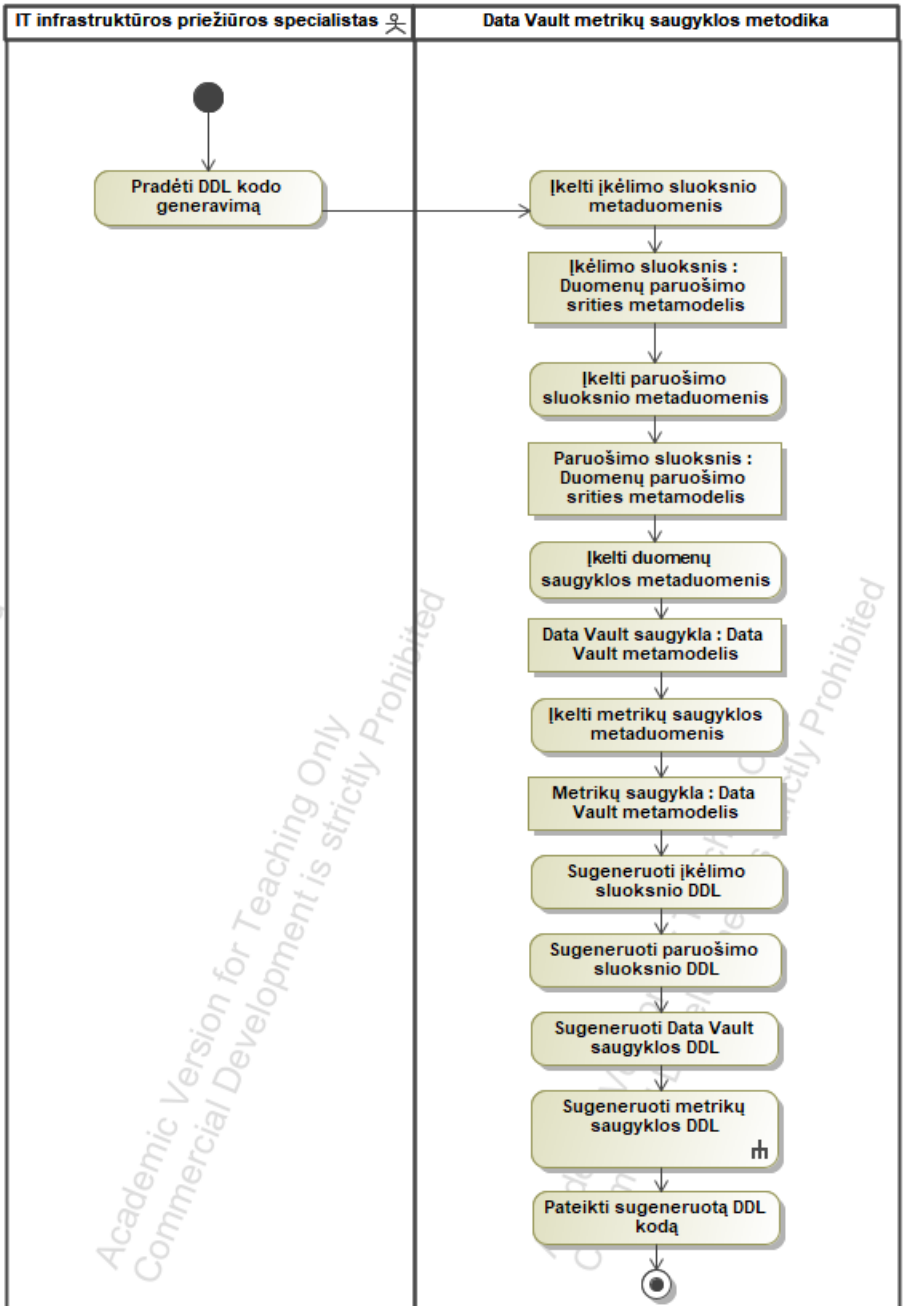
2.7 pav. „Sudaryti metrikų saugyklą“ veiklos diagrama

Suformavus metrikų saugyklą galima pradėti kurti duomenų pjūvius šios saugyklos pagrindu. Duomenų pjūvio kūrimui naudotojas turi nurodyti įtrauktinas metrikų duomenų lenteles, jų atributus. Taip pat kadangi duomenų pjūviai skirti greitam ir informatyviam duomenų pateikimui norimu pjūviu, naudotojui reikia nurodyti duomenų surinkimo laikotarpį (periodą). Pavyzdžiui, naudotoją domina paskutinio mėnesio duomenys, tokiu būdu senesni duomenys nebūtų įkeliami, tai pagreitina duomenų išrinkimą. Naudotojas taip pat gali nurodyti agregavimo funkcijas skirtas sujungti tam tikrus duomenis. Užbaigus visą duomenų pjūvio konfigūraciją naudotojas ją patvirtina ir yra sukuriama bei išsaugoma šio duomenų pjūvio struktūra. Visą šį procesą aprašo 2.8 paveiksle pateikta veiklos diagrama.



2.8 pav. „Sudaryti metrikų duomenų pjūvį“ veiklos diagrama

Visi prieš tai aprašyti žingsniai skirti tiktai surinkti informacija apie esamų duomenų struktūrą, transformuoti ir suformuoti naujas duomenų struktūras – jų informaciją, tačiau ne sukurti fizines lenteles ir jų ryšius duomenų saugykloje. Tam pasiekti yra pasitelkiamas automatinis *DDL* komandų generavimas, kuris įvykdomas surenkant visą sukauptą informaciją apie duomenų saugyklos sluoksnius ir sugeneruojant *DDL* komandas *Data Vault* metrikų saugyklos sukūrimui pasirinktoje duomenų saugyklos platformoje. Visą šį procesą aprašo 2.9 paveiksle pateikta *DDL* komandų generavimo panaudojimo atvejo veiklos diagrama.



2.9 pav. „Sugeneruoti *DDL* komandas“ veiklos diagrama

Lentelių sukūrimas duomenų saugykloje nėra paskutinis žingsnis, kadangi reikia lenteles užpildyti duomenimis. Tam pasiekti turi būti sugeneruoti *ETL* procesai. Šie procesai apima visus duomenų saugyklos sluoksnius ir turi specifinę paskirtį. Pirmiausia turi būti parengiami *ETL* procesai skirti duomenų įkėlimui į duomenų saugyklą. Turi būti parengiama po atskirą procesą įkelti duomenis iš konkretaus šaltinio. Įkėlus duomenis jie turi būti transformuoti į *Data Vault* sluoksnį esantį duomenų saugojimo srityje. Tam yra parengiami atskiri *ETL* procesai, kurie transformuoja duomenis iš kiekvieno metrikų šaltinio sukuriant centrus, sąsajas ir palydovus. Užpildžius *Data Vault* sluoksnį, turi būti užpildoma metrikų saugykla. Tam pasiekti yra suformuojami *ETL* procesai, kurie iš *Data Vault* saugyklos išrenka duomenis, juos agreguoja ir apskaičiuoja naujas reikšmes pagal naudotojo nurodytas funkcijas ir jas išsaugoja metrikų saugykloje. Visą šį procesą aprašo 2.10 paveiksle pavaizduota *ETL* procesų generavimo panaudojimo veiklos diagrama.

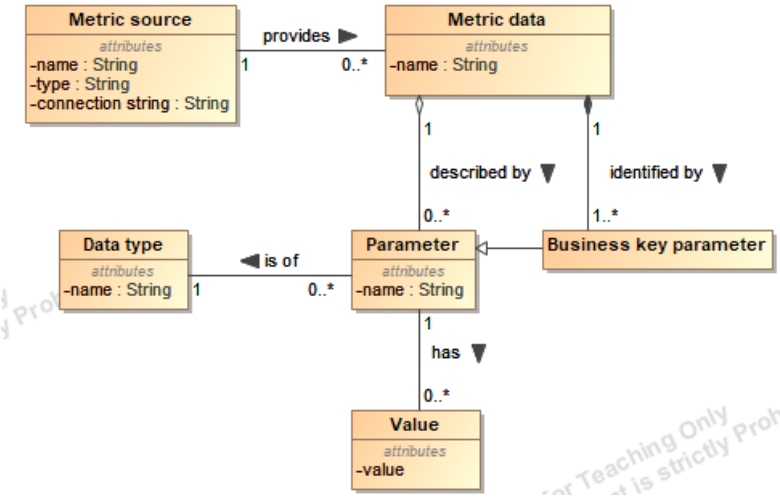
2.10 pav. „Sugeneruoti *ETL* procesus“ veiklos diagrama

Taigi, kaip galima pastebėti, visi panaudojimo atvejai aprašo pilną metrikų saugyklos sudarymo procesą nuo duomenų įkėlimo iki jų duomenų pjūvių sudarymo skirtų kompiuterinių resursų metrikų stebėsenai ir analizei.

2.2. *Data Vault* metrikų saugyklos sudarymo metodikos duomenų modelis

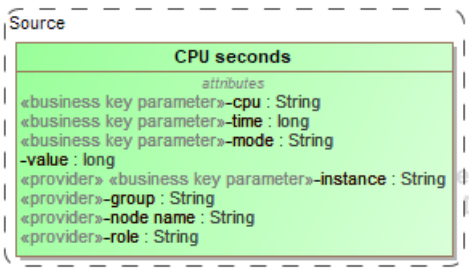
Data Vault 2.0 duomenų saugyklos struktūrai aprašyti yra pasitelkiami trys pagrindiniai metamodeliai: duomenų šaltinio metamodelis, duomenų paruošimo srities metamodelis ir *Data Vault*

saugyklos metamodelis. Metrikų duomenų šaltinio metamodelis (žr. 2.11 pav.) naudojamas aprašyti naudojamų metrikų duomenų šaltinių struktūrą. Pagrindiniai šio metamodelio elementai: metrikų šaltinis ir metrikos duomenų struktūra. Metrikų šaltinis skirtas aprašyti specifinę konkretaus metrikų šaltinio informaciją (pavadinimą, tipą, prisijungimo būdą). Metrikos duomenų struktūra aprašoma keliomis esybėmis. Pirmiausia, kiekviena metrika turi savo pavadinimą ir su ja susijusius parametrus. Kai kurie iš šių parametrų yra verslo raktai ir yra naudojami identifikuoti unikalius metrikos duomenų įrašus. Kiekvienas metrikos parametras turi pastovų nustatytą duomenų tipą ir per laiką sukaupiamą reikšmių rinkinį.



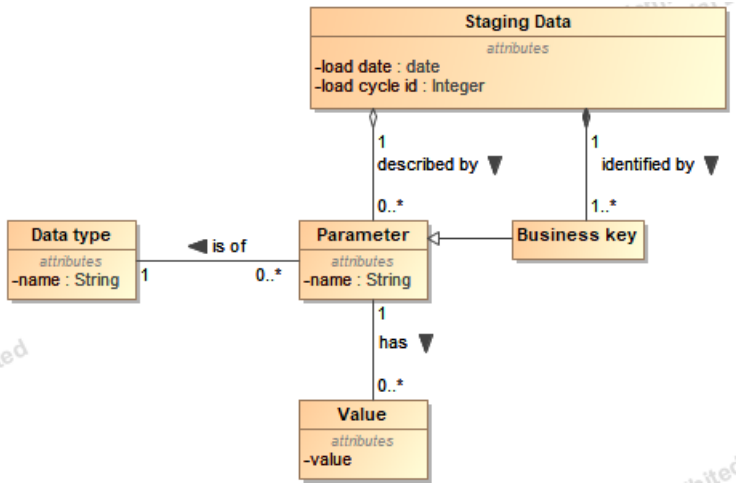
2.11 pav. Metrikų duomenų šaltinio metamodelis

Pagal šį metrikų duomenų šaltinio metamodelį sudarytą metrikų duomenų šaltinio pavyzdį galima matyti 2.12 paveiksle. Šis pavyzdys aprašo procesoriaus darbo metrikos duomenis. Duomenų šaltinyje apie metriką kaupiama tokia informacija, kaip procesoriaus pavadinimas, laikas kada buvo užfiksuota metrikos reikšmė, procesoriaus darbo režimas (budėjimas, įvedimo/išvedimo operacijų laukimas, pertrauktis ir pan.), laikas praleistas nurodytame darbo režime, įrenginio IP adresas, grupė, kuriai priklauso įrenginys, įrenginio rolė. Tai kaip galima matyti, metriką „Procesoriaus sekundės“ (angl. *CPU* seconds), kuri saugoma duomenų šaltinyje, aprašo įvairūs parametrai turintys savo tipus ir reikšmes. Kai kurie parametrai (procesoriaus pavadinimas, metrikos užfiksavimo laikas, režimas, įrenginio IP adresas) yra identifikaciniai ir yra pažymėti verslo rakto stereotipu „business key parameter“. Taip pat galima matyti, kad atributai aprašantys dvi skirtingas esybes (vieni aprašo procesoriaus darbo informaciją, kiti aprašo įrenginio informaciją – išskirti „provider“ stereotipu) yra saugomi vienoje ir toje pačioje struktūroje.



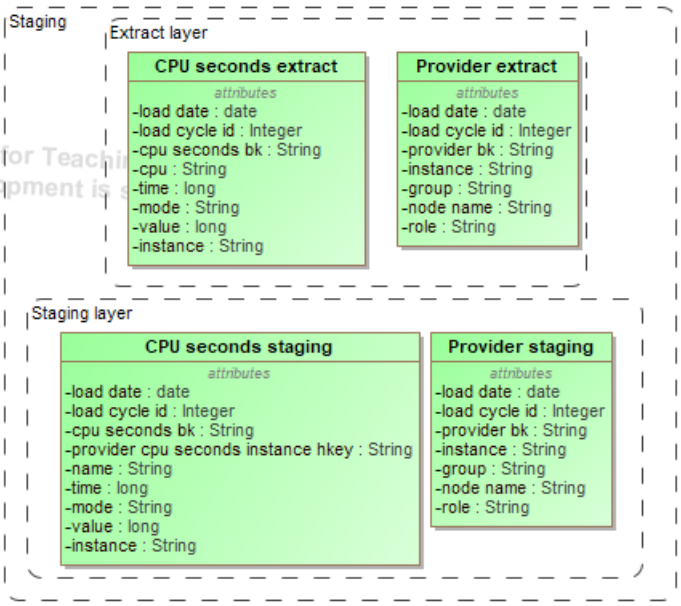
2.12 pav. Metrikos pavyzdys saugojamas duomenų šaltinyje

Duomenų paruošimo srities metamodelis (žr. 2.13 pav.) skirtingai negu metrikų šaltinio metamodelis jau naudojamas duomenų saugyklos viduje saugojamų duomenų struktūrai aprašyti. Šis metamodelis naudojamas suformuoti duomenų įkėlimo (angl. *extraction layer*) ir duomenų paruošimo (angl. *staging layer*) sluoksnius. Duomenų įkėlimo sluoksnis naudojamas duomenų šaltinio struktūros nukopijavimui pridedant pagalbinius elementus skirtus *Data Vault* saugyklos formavimui ir jei reikia išskaidant duomenų šaltinio struktūras pagal paskirtį. Duomenų paruošimo srities metamodelis yra labai panašus į metrikų duomenų šaltinio metamodelį, tik neturi metrikų šaltinio elemento ir metrikų duomenų struktūra yra papildoma duomenų įkėlimo ciklo ir datos informacija. Taip pat duomenų paruošimo srities metamodeliu suformuojamas duomenų paruošimo sluoksnis turi daugiau verslo raktų elementų negu duomenų šaltinis. Taip yra todėl, nes iš anksto yra apskaičiuojami visi sujungtų verslo raktų maišos funkcijos rezultatai. Šie maišos funkcijų rezultatai yra naudojami identifikuoti duomenų įrašams išskaidytiems iš vieno struktūros į centrus, palydovus ir sąsajas. Šis išankstinis maišos funkcijų apskaičiavimas leidžia užpildyti *Data Vault* centrus, palydovus ir sąsajas lygiagrečiai vienu metu.



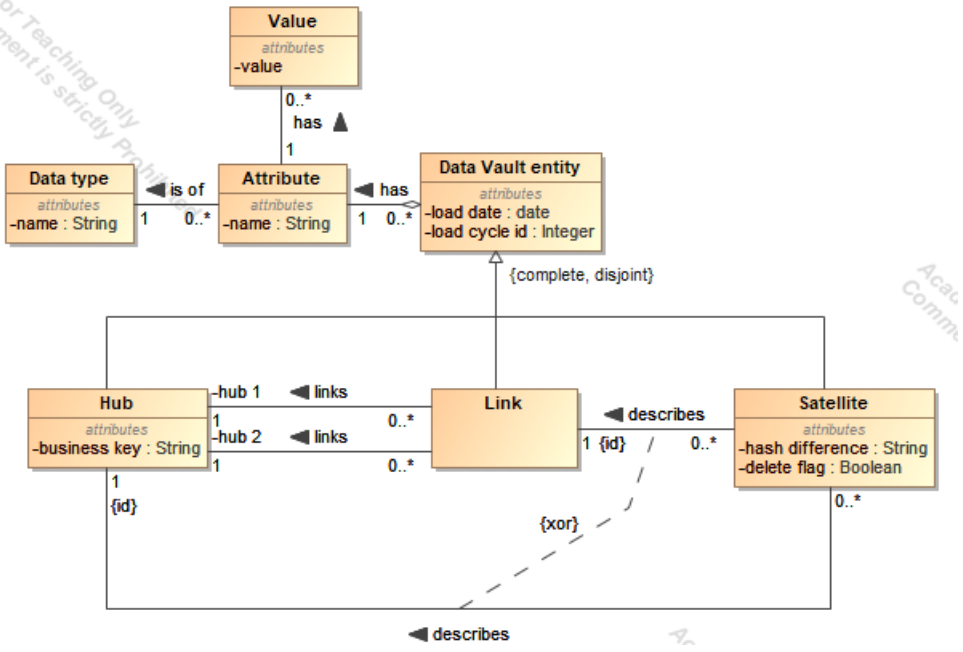
2.13 pav. Duomenų paruošimo srities metamodelis

Pagal šį duomenų paruošimo srities metamodelį sudarytą duomenų paruošimo srities pavyzdį, skirtą anksčiau aprašytai procesoriaus darbo metrikai, galima matyti 2.14 paveiksle. Kaip galima pastebėti duomenų paruošimo srityje pradinė duomenų šaltinio struktūra buvo išskaidyta į dvi: procesoriaus darbo sekundžių struktūrą ir metrikų įrenginio/tiekėjo struktūrą. Pirmiausia, buvo suformuotas duomenų įkėlimo sluoksnis (angl. *extract layer*), kuriame abi struktūros buvo papildytos duomenų įkėlimo laiko (angl. *load date*) ir ciklo (angl. *load cycle id*) atributais, taip pat buvo pridėtas sujungtų verslo raktų maišos atributas (*cpu seconds bk* – procesoriaus darbo struktūroje ir *provider bk* – įrenginio struktūroje). Baigus formuoti duomenų įkėlimo sluoksnį buvo sudarytas duomenų paruošimo sluoksnis (angl. *staging layer*). Sluoksnis buvo užpildytas duomenų įkėlimo sluoksnio duomenimis ir papildytas maišos rezultatu identifikuojančiu metrikų įrenginio ir procesoriaus darbo struktūrų ryšį. Ši maiša apskaičiuojama sujungiant abiejų struktūrų verslo raktų maišos rezultatus ir suskaičiuojant naują maišos rezultatą. Šis rezultatas vėliau naudojamas formuojant *Data Vault* sąsajų elementus *Data Vault* saugykloje.



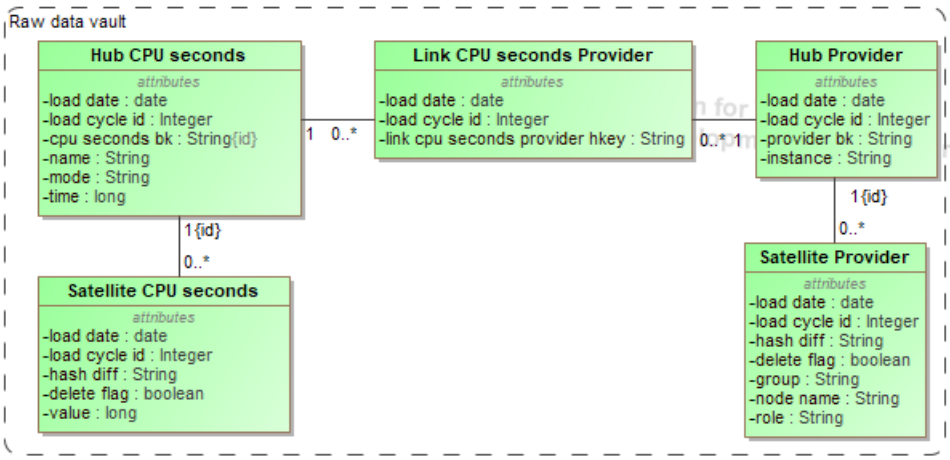
2.14 pav. Išskaidytos metrikos pavyzdys duomenų paruošimo srityje

Data Vault sluoksnio metamodelis skirtas aprašyti *Data Vault* saugykloje saugojamų duomenų struktūrą. Šio sluoksnio pagrindiniai trys elementai yra centrai, sąsajos ir palydovai. Kiekvienas iš šių elementų turi savo įkėlimo ciklą ir datą, turi jį aprašančius atributus, kurie turi savo duomenų tipus ir reikšmes. Centrai išsiskiria tuo, kad juose yra saugojami duomenų įrašus identifikuojantys verslo raktai. Sąsajos išskirtinės tuo, kad sujungia kelis duomenų centrus. Palydovai savyje saugo aprašomąją informaciją dėl to jie gali būti susieti tiek su centru, tiek su sąsaja. Palydovai taip pat turi pagalbinius elementus identifikuoti ar įrašas buvo ištrintas (ištrynimo indikatorius) ir ar jis buvo atnaujintas (aprašomųjų atributų maišos funkcija, skirta palyginti su kita maišos funkcija ar kas nors pakito).



2.15 pav. *Data Vault* sluoksnio metamodelis

Pagal šį metamodelį sudarytą *Data Vault* saugyklos pavyzdį skirta anksčiau aprašytai procesoriaus darbo metrikai galima matyti 2.16 paveiksle. Kaip galima pastebėti tiek procesoriaus darbo, tiek metrikų įrenginio/tiekėjo struktūros buvo išskaidytos pagal *Data Vault* modelį į centrus ir palydovus bei taip pat buvo sukurtas sąsajos elementas aprašantis sąryšį tarp procesoriaus darbo metrikos ir įrenginio, kuriame ji buvo užfiksuota. Taip pat galima matyti, kad palydovuose buvo įtraukti duomenų pašalinimo (angl. *delete flag*) bei maišos skirtumui (angl. *hash diff*) skaičiuoti skirti atributai. Taip pat, kadangi *Data Vault* duomenų saugyklos sluoksnis yra pastovus jame tai pat yra sukuriami ir ryšio apribojimai, skirtingai nuo duomenų paruošimo srities, kuri su kiekvienu nauju duomenų įkėlimu yra išvaloma ir dėl lygiagretaus duomenų įkėlimo nenaudoja ryšio apribojimų.

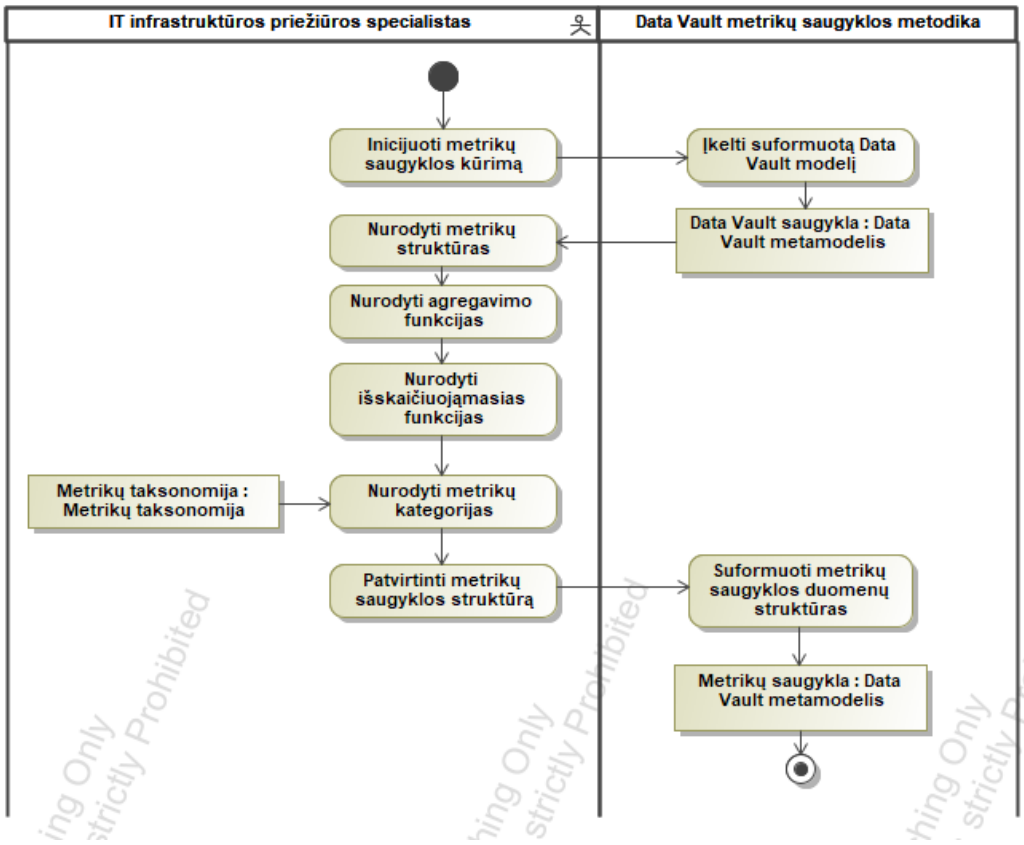


2.16 pav. Metrikos pavyzdys saugojamas *Data Vault* saugykloje

2.3. *Data Vault* metrikų saugyklos sudarymo formalizuotas sprendimo aprašas

Keliamiems reikalavimams spręsti yra siūloma vadovautis *Data Vault* metrikos saugyklos sudarymo metodika (žr. 2.1 pav.) ir sukurti metrikų saugyklos sudarymo įrankį. Ši metodika aprašo visą procesą nuo duomenų šaltinio pridėjimo iki metrikų atvaizdavimo vizualizavimo įrankyje. Nemaža dalis metrikų saugyklos metodikos žingsnių jau dabar yra realizuoti egzistuojančiais *Data Vault* duomenų saugyklos sudarymo automatizacijos įrankiais, tačiau pats metrikų saugyklos sudarymo procesas nėra realizuotas ar plačiau detalizuotas. Dėl to šio darbo siūlomas sprendimas daugiausiai apima metrikų saugyklos sudarymo procesą ir formalizuotai aprašo pagal metodiką sumodeliuotos metrikų saugyklos struktūros (*DDL* komandų) ir duomenų įkėlimo (*ETL* komandų) sugeneravimo procesus.

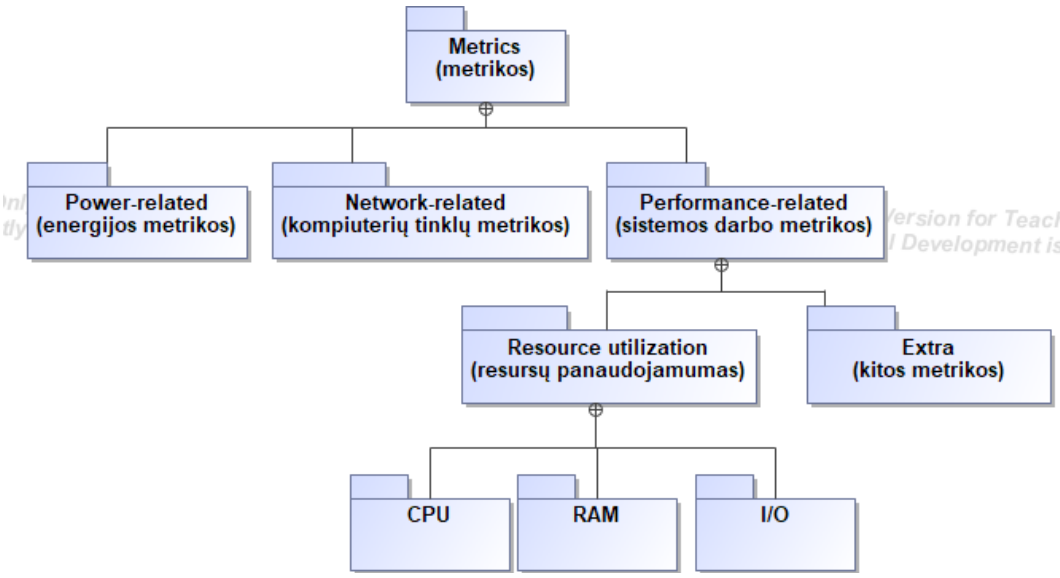
Prieš generuojant *DDL* ir *ETL* komandas, metrikų saugykla turi būti sumodeliuota naudotojo pagal metrikų saugyklos sudarymo procesą pateikiamą metodikoje (žr. 2.17 pav. pateiktą veiklos diagramą).



2.17 pav. „Sudaryti metrikų saugyklą“ veiklos diagrama

Naudotojui inicijuojant metrikų saugyklos kūrimą pirmiausia yra įkeliama informacija saugoma *Data Vault* sluoksnyje – suformuoti centrai (angl. *hubs*), sąsajos (angl. *links*) ir palydovai (angl. *satellites*). Įkėlus šią informaciją naudotojui yra pateikiama metrikų saugyklos sudarymo forma, kurioje naudotojas gali nurodyti įtrauktinas metrikas – t. y. jų metrikų duomenų centrus ir su jais susijusius palydovus, sąsajos esančios tarp įtraukiamų centrų yra įtraukiamos automatiškai. Įtraukus metriką naudotojas gali ją transformuoti nurodydamas agregavimo sąlyga, tokiu būdu visi grupavimo laukai patampa verslo raktais, o agregavimo funkcijos gali būti nurodomos išskaičiuojamųjų funkcijų pagalba. Išskaičiuojamosios funkcijos taip pat gali būti sukuriamos ir ne agreguojamoms metrikos, naudotojui tereikia nurodyti funkcijos išraišką, kurioje gali būti nurodomi palydovuose egzistuojantys duomenų stulpeliai, konstantos ar standartinės *SQL* funkcijos. Galiausiai, pridėtai metrikai būtina nurodyti metrikų kategoriją iš pateikiamos metrikų taksonomijos.

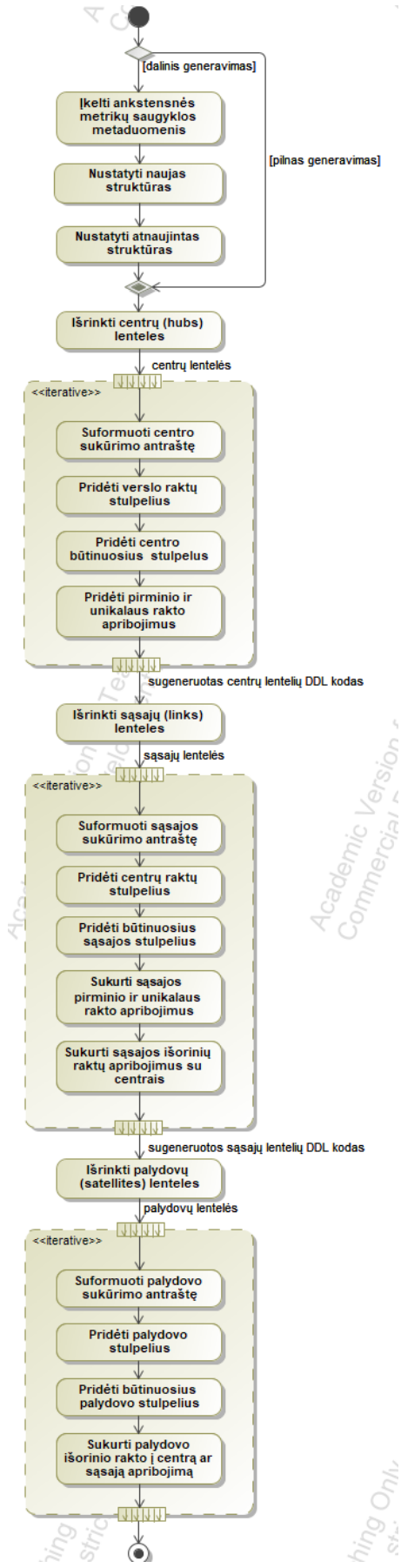
Atlikta metrikų taksonomijos analizė (žr. 1.5 poskyrį) parodė, kad siūloma taksonomija turi apimti tris pagrindines kategorijas: energijos metrikas, kompiuterinių tinklų metrikas ir sistemos darbo metrikas (1 lentelė). Tačiau, ši taksonomija turi taip pat labiau detalizuoti sistemos darbo metrikų kategoriją, kadangi ji yra pati aktualiausia (2 lentelė). Taigi, atsižvelgiant į šias metrikų taksonomijos analizės išvadas, buvo parengta nauja metrikų taksonomija, kuri suskirsto metrikas į skirtingas grupes pagal jų paskirtį ir pateikiama 2.18 paveiksle.



2.18 pav. Metrikų taksonomija

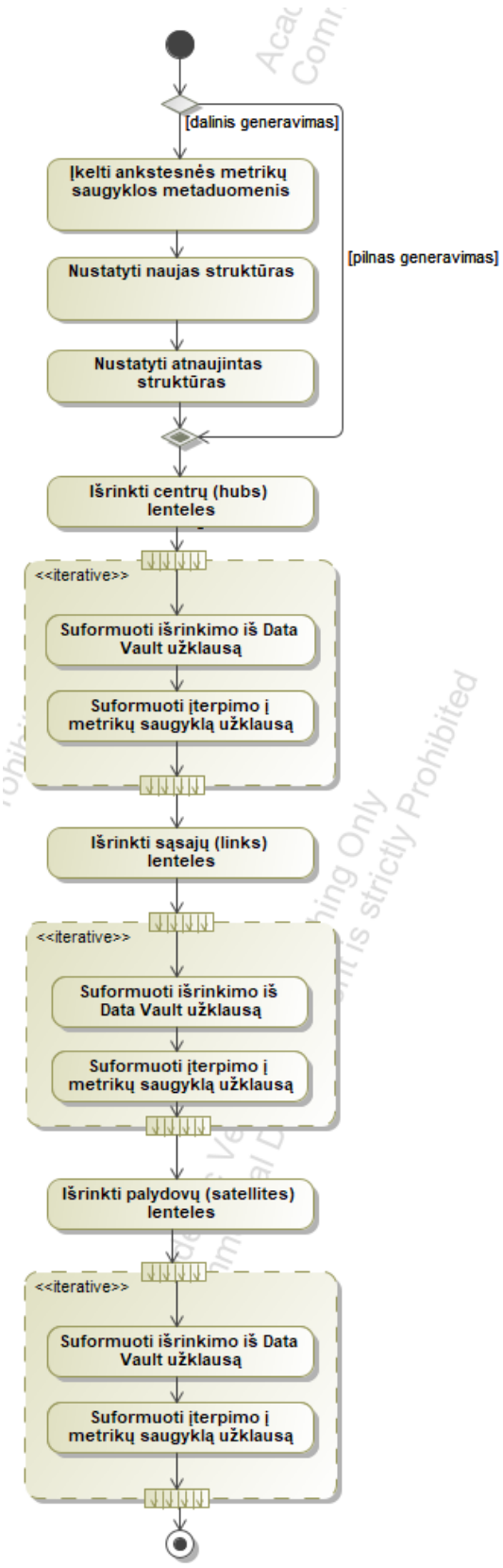
Kaip galima matyti, sudaryta metrikų taksonomija sujungia 1.5 poskyryje nagrinėtų šaltinių dažniausiai naudojamas metrikų grupes, išskiriant tris aukščiausio lygio grupes: energijos našumo, internetinio ryšio ir sistemos darbo metrikų kategorijas. Taip pat išskaido sistemos darbo metrikas į resursų panaudojamumo bei kitas metrikas. Tokiu būdu, pasitelkiant šią taksonomiją, didžiąją dalį šaltiniuose analizuotų metrikų galima tiksliai suskirstyti pagal jų paskirtį.

Baigęs formuoti metrikų saugyklą, naudotojas patvirtina modelio išsaugojimą ir ji yra išsaugojama. Turint šią informaciją apie metrikų saugyklą, įrankis gali sugeneruoti metrikų saugyklos struktūrą ir duomenų įkėlimo procesus. Metrikų saugyklos struktūra sudaroma pasitelkiant *DDL* komandas, kurios gali būti sugeneruotos pasitelkiant 2.19 pav. pateiktą algoritmą. Yra du galimi metrikų saugyklos struktūros generavimo būdai: pilnas generavimas ir dalinis generavimas. Pilnas generavimas apima visos metrikų saugyklos sudarymą nuo nulio. Dalinis generavimas surenka informaciją apie prieš tai sukurtos metrikos saugyklos struktūrą ir sugeneruoja *DDL* komandas tik naujoms ar atsinaujinusioms lentelėms. Taigi, generavimo įrankis pirma turi įvertinti kokį būdą nurodė naudotojas ir jei buvo pasirinktas dalinis generavimas, surinkti informaciją apie metrikų saugyklos struktūros pokyčius. Tada pereinama prie metrikų saugyklos centrų struktūros generavimo. Pirmiausia išrenkamos visos centrų lentelės. Tada sukamas ciklas per kiekvieną centro lentelę ir jai atliekami šie veiksmai: suformuojama centro sukūrimo antraštė (pvz. *CREATE TABLE schema.hub_metric*), pridedami verslo rakto stulpeliai ir taip pat pridedami būtinieji centro stulpeliai pagal *Data Vault* duomenų modelį. Galiausiai, suformuojami pirminio ir unikalaus rakto apribojimai. Sukūrus visus centrus pereinama prie sąsajų kūrimo. Vėlgi sukamas ciklas per kiekvieną sąsajos lentelę ir jai atliekami šie veiksmai: suformuojama sąsajos sukūrimo antraštė, pridedami susijusių centrų raktų stulpeliai, pridedami būtinieji stulpeliai pagal *Data Vault* modelį ir sukuriami pirminio, unikalaus ir išorinio rakto apribojimai tarp sąsajos ir susijusių centrų. Užbaigus kurti sąsajas pereinama prie palydovų sudarymo. Sukamas ciklas per palydovų lenteles ir joms atliekami šie veiksmai: suformuojama palydovo sukūrimo antraštė, pridedami palydovo aprašomieji stulpeliai, pridedami būtinieji palydovo stulpeliai pagal *Data Vault* metodiką ir sukuriami išorinio rakto apribojimai į susijusią centro ar sąsajos lentelę. Baigus generuoti šias *DDL* komandas metrikų saugyklos struktūros generavimo procesas yra užbaigtas.



2.19 pav. Metrikų saugyklos *DDL* komandų sugeneravimo algoritmas

Metrikų saugyklos lentelėse duomenys įkeliami pasitelkiant *ETL* procesus. Šie procesai gali būti sugeneruoti pasitelkiant 2.20 pav. pateiktą *ETL* komandų generavimo algoritmą. Pirmiausiai, taip pat kaip ir su *DDL* struktūros generavimu, turi būti atsižvelgta į tai ar generuojami visų metrikos saugyklos lentelių procesai atliekant pilną generavimą, ar tik pokyčiai atliekant dalinį generavimą. Jei atliekamas dalinis generavimas, turi būti surenkama informacija apie metrikų saugyklos pasikeitimus. Tada pereinama prie centrų, sąsajų ir palydovų informacijos įkėlimo. Ši informacija įkeliama į visas lenteles panašiai. Pirmiausia yra suformuojama duomenų išrinkimo iš *Data Vault* saugyklos duomenų užklausa, kuri pritaiko aprašytas išskaičiuojamąsias funkcijas ir jei reikia agregavimą. Tada yra suformuojama duomenų įterpimo į nurodytą lentelę užklausa, kuri išrinktą informaciją jau įrašo į metrikų saugyklos lenteles. Sugeneravus visus *ETL* procesus centrams, sąsajoms ir palydovams, generavimo procesas yra baigiamas.



2.20 pav. Metrikų saugyklos *ETL* komandų sugeneravimo algoritmas

Taigi, tokiu būdu yra pasiekiamas metrikų saugyklos sudarymo įgyvendinimas, pasitelkiant metrikų saugyklos sudarymo įrankį ir jame realizuojamus algoritmus.

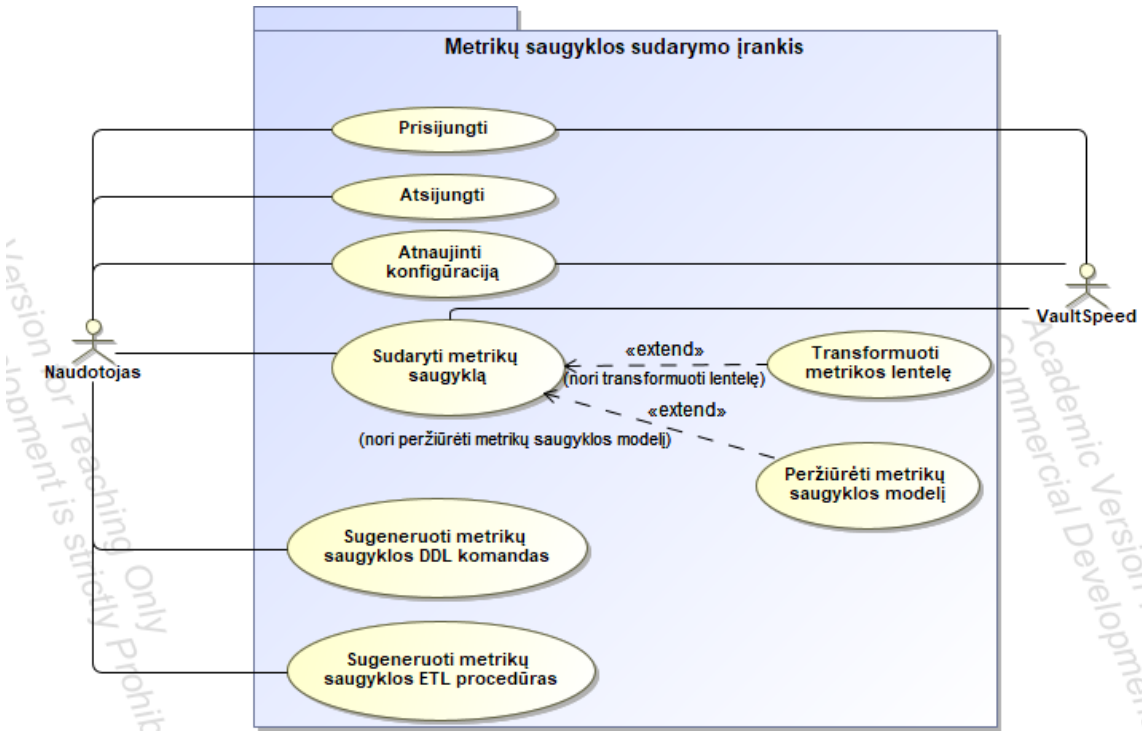
2.4. *Data Vault* metrikų saugyklos sudarymo metodikos reikalavimų apibendrinimas

Norint parengti *Data Vault 2.0* duomenų saugyklos metrikų saugyklą, reikia sudaryti metodiką nuo duomenų šaltinio pridėjimo, metrikų saugyklos sudarymo iki duomenų atvaizdavimo. Didžioji dalis metodikos procesų, kaip duomenų šaltinių pridėjimas, duomenų įkėlimo ir saugojimo sričių sudarymas, gali būti automatizuojami jau egzistuojančių įrankių kaip *VaultSpeed*, dėl to šie žingsniai gali būti įgyvendinami, pasitelkiant jau egzistuojančius sprendimus. Tačiau, metrikų saugyklos sudarymas, jos metrikų kategorizavimas, metrikų saugyklos *DDL* struktūros ir *ETL* procesų sugeneravimas nėra realizuotas. Dėl to nustatyta, kad reikia realizuoti įrankį, leidžiantį naudotojui sudaryti metrikų saugyklą, naudojantį *Data Vault* duomenų modelį ir sugeneruoti jos *DDL* ir *ETL* kodą.

3. *Data Vault* metrikų saugyklos sudarymo įrankio realizacijos projektas

3.1. *Data Vault* metrikų saugyklos panaudojimo atvejai

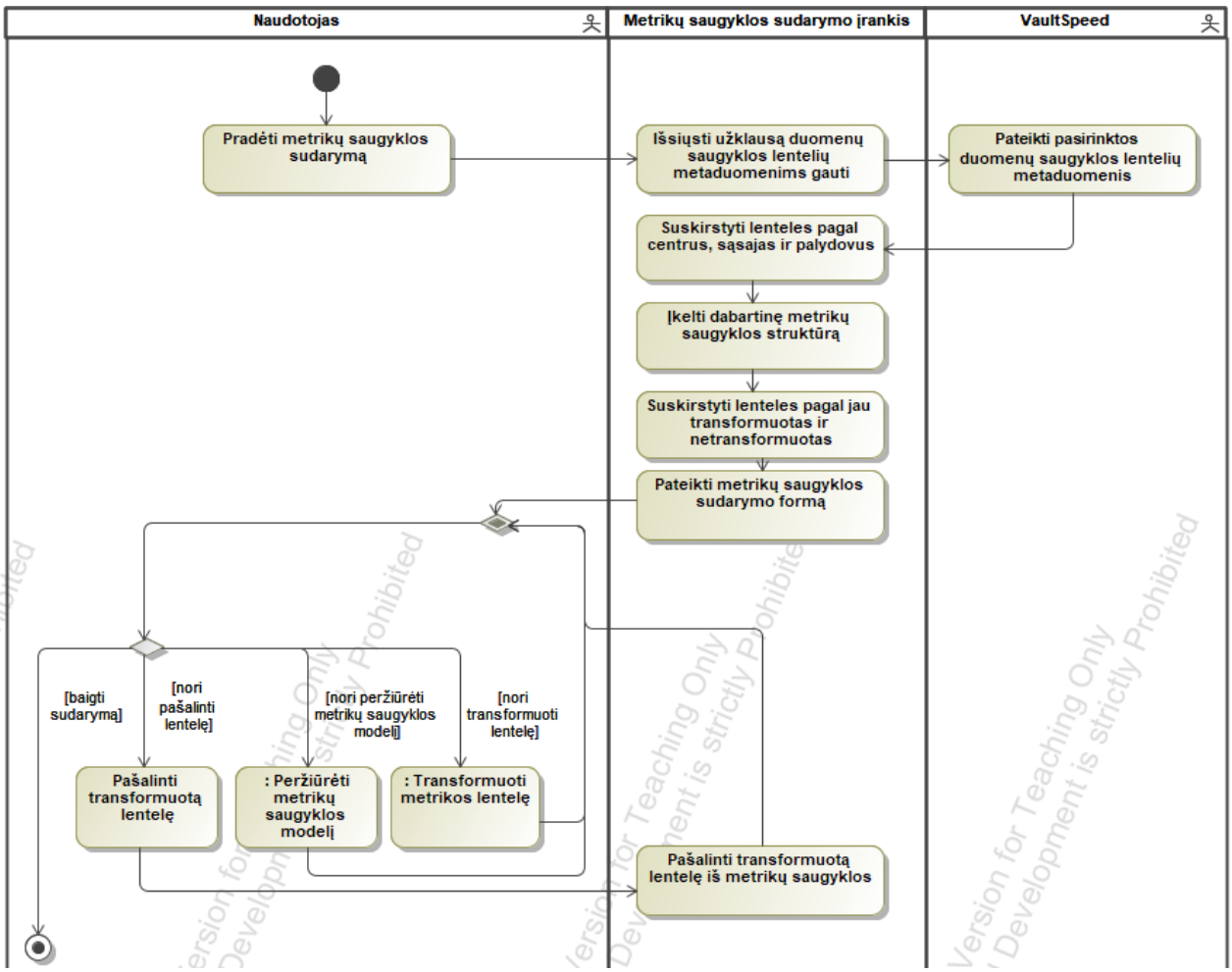
Metrikų saugyklos sudarymo įrankiui įgyvendinti buvo apibrėžti pagrindiniai panaudojimo atvejai, kuriuos šis įrankis turi leisti naudotojui įvykdyti. Panaudojimo atvejų diagrama pateikiama 3.1 paveiksle.



3.1 pav. Metrikų saugyklos sudarymo įrankio panaudojimo atvejų diagrama

Kadangi naudotojas metrikų saugyklos sudarymui naudojasi *VaultSpeed* įrankyje suformuotos *Data Vault 2.0* duomenų saugyklos lentelių struktūra, reikalingas prisijungimas prie *VaultSpeed* įrankio. Tam pasiekti skirti prisijungimo, atsijungimo ir konfigūracijos atnaujinimo panaudojimo atvejai. Norint prisijungti prie metrikų saugyklos sudarymo įrankio reikia prisijungti su egzistuojančia *VaultSpeed* paskyra, tokiu būdu įrankis gali pasiekti jam reikalingus duomenų saugyklos metaduomenis. Baigęs darbą, naudotojas gali atsijungti nuo įrankio pasinaudodamas atsijungimo panaudojimo atveju. Jei naudotojas turi kelis skirtingus projektus ir duomenų saugyklas susietas su ta pačia *VaultSpeed* paskyra, jis gali atnaujinti savo metrikų saugyklos sudarymo įrankio konfigūraciją, pasirenkant specifinį jį dominantį projektą ir duomenų saugyklą.

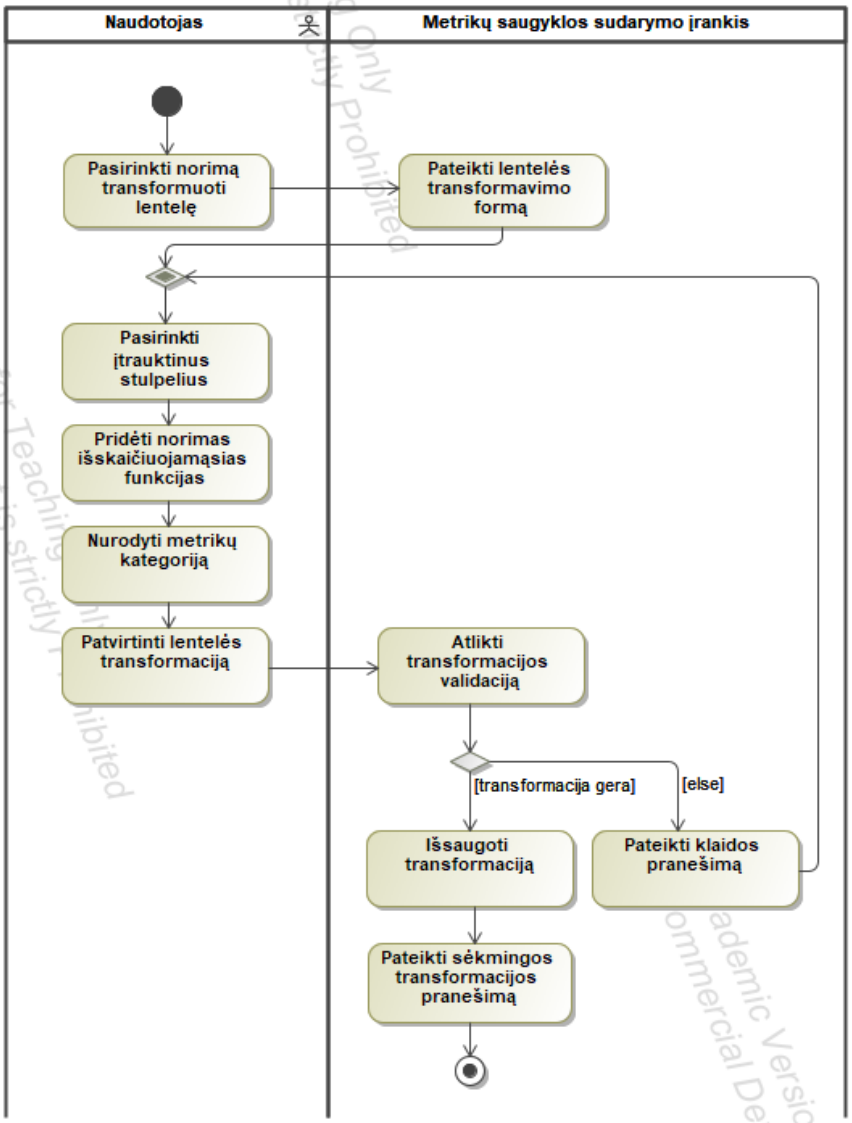
Pagrindinė metrikų saugyklos sudarymo įrankio paskirtis – metrikų saugyklos sudarymas. Tam pasiekti skirtas metrikų saugyklos sudarymo panaudojimo atvejis. Šio panaudojimo atvejo kontekste yra suprojektuojama metrikų saugyklos struktūra, apibrėžiamos įtraukiamos lentelės, jų stulpeliai, pridedamos išskaičiuojamosios funkcijos. Visą šio panaudojimo atvejo eiga aprašo 3.2 paveiksle pateikta diagrama.



3.2 pav. Metrikų saugyklos sudarymo panaudojimo atvejo veiklos diagrama

Metrikų saugyklos sudarymo procesas prasideda nuo duomenų saugyklos sluoksnio lentelių metaduomenų įkėlimo iš *VaultSpeed* įrankio. Šie metaduomenys apima informaciją apie duomenų saugyklos sluoksnyje sukurtus centrus, sąsajas ir palydovus. Apie kiekvieną lentelę žinomas jos pavadinimas, tipas, stulpeliai, stulpelių duomenų tipai, jų tikslumas, bei ryšiai tarp lentelių. Gavus šiuos duomenis lentelės yra suskirstomos pagal centrus, sąsajas ir palydovus, kad vėliau naudotojui lentelių transformavimo pasirinkime būtų rodomos tik centrų lentelės. Tik pasirinkus centro lentelę būtų užpildomi susiję palydovai, o sąsajos įtraukiamos tik tuo atveju, jei jos egzistuoja tarp jau transformuotų centrų. Įkėlus *VaultSpeed* metaduomenis, taip pat įkeliama ir egzistuojanti metrikų saugyklos informacija. Tai yra įkeliamos jau atliktos transformacijos, jei tokios buvo atliktos. Įkėlus šią informaciją, lentelės suskirstomos pagal jau transformuotas ir netransformuotas, ir naudotojui pateikiama metrikų saugyklos sudarymo forma.

Naudotojas gali rinktis iš kelių galimų veiksmų. Pašalinti jau transformuotą lentelę, tokiu būdu lentelė pašalinama iš metrikų saugyklos transformacijų, naudotojas taip pat gali transformuoti metrikos lentelę. Šis procesas detalizuojamas dar viena veiklos diagrama pateikiama 3.3 paveiksle.

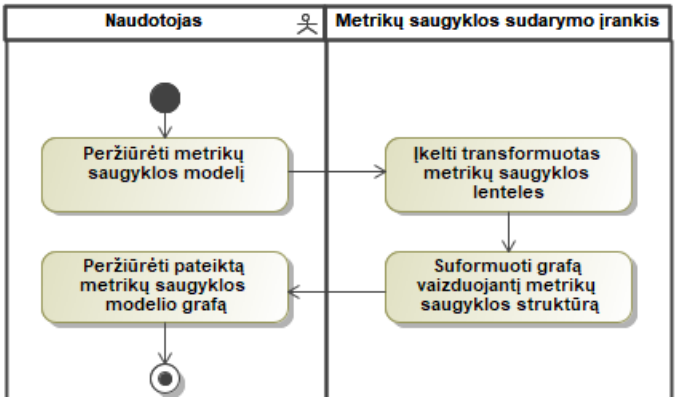


3.3 pav. Metrikos lentelės transformavimo panaudojimo atvejo veiklos diagrama

Transformavimo procesas pradedamas lentelės išsirinkimu. Naudotojui išsirinkus lentelę pateikiama transformavimo forma. Šioje formoje naudotojas pažymi norimus įtraukti stulpelius, kurių informacija įkeliama iš *Data Vault* duomenų saugyklos sluoksnio. Naudotojas taip pat gali pridėti naujus išskaičiuojamuosius stulpelius, kurie apskaičiuojami pagal naudotojo aprašytą išraišką galinčią naudoti paprastas matematines operacijas (suma, atimtis, daugyba, dalyba), lentelės pasirinkto palydovo stulpelių reikšmes ir tam tikras pateikiamas standartines *SQL* kalbos funkcijas. Galiausiai, naudotojas privalo nurodyti metrikų kategoriją pagal 2.18 paveiksle pateiktą metrikų taksonomiją. Baigus nurodinėti transformavimo parametrus naudotojas transformaciją išsaugo. Sistema atlieka transformacijos validaciją ir jei kažkas sukonfigūruota neteisingai pateikiamas klaidos pranešimas. Jei transformacija gera, ji yra išsaugojama ir pateikiamas sėkmingos transformacijos pranešimas.

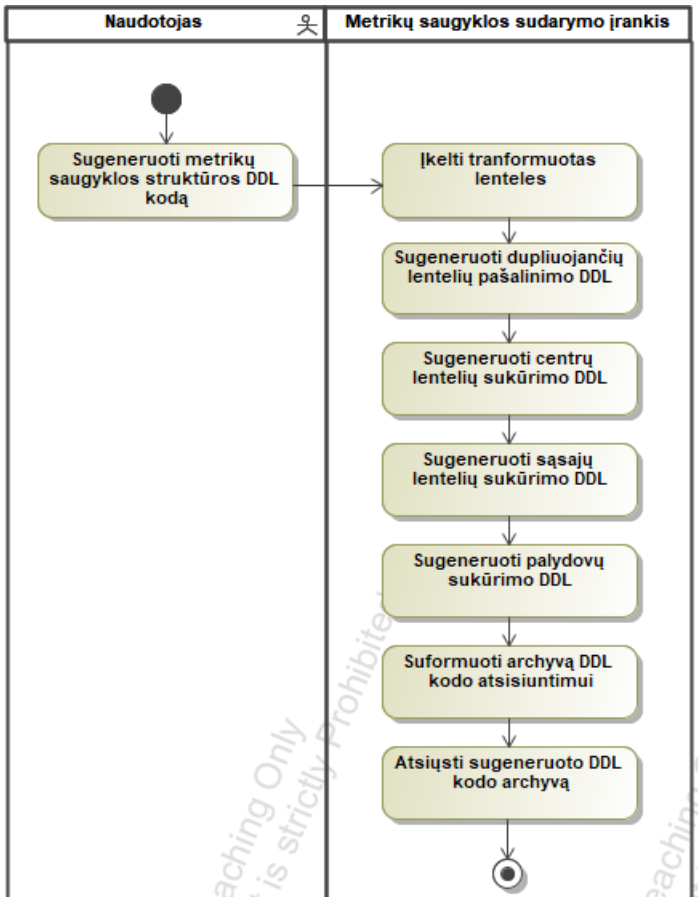
Grįžtant prie metrikų saugyklos sudarymo naudotojas taip pat gali peržiūrėti metrikų saugyklos modelį, kuris atvaizduoja jau transformuotas lenteles ir ryšius tarp jų. Šiam procesui atvaizduoti buvo sudaryta veiklos diagrama pateikiama 3.4 paveiksle. Naudotojui pasirinkus peržiūrėti metrikų

saugyklos modelį yra įkeliama transformuotų lentelių informacija ir suformuojamas grafas atvaizduojantis transformuotas lenteles ir ryšius tarp jų. Galiausiai, jei metrikų saugyklos modelis tenkina, metrikų saugyklos sudarymo procesas yra užbaigiamas.



3.4 pav. Metrikų saugyklos modelio peržiūros panaudojimo atvejo veiklos diagrama

Baigus sudarinėti metrikų saugyklą įrankyje lieka du paskutiniai žingsniai: *DDL* ir *ETL* komandų sugeneravimas. Metrikų saugyklos struktūros *DDL* kodo generavimą aprašo 3.5 paveiksle pavaizduota veiklos diagrama.

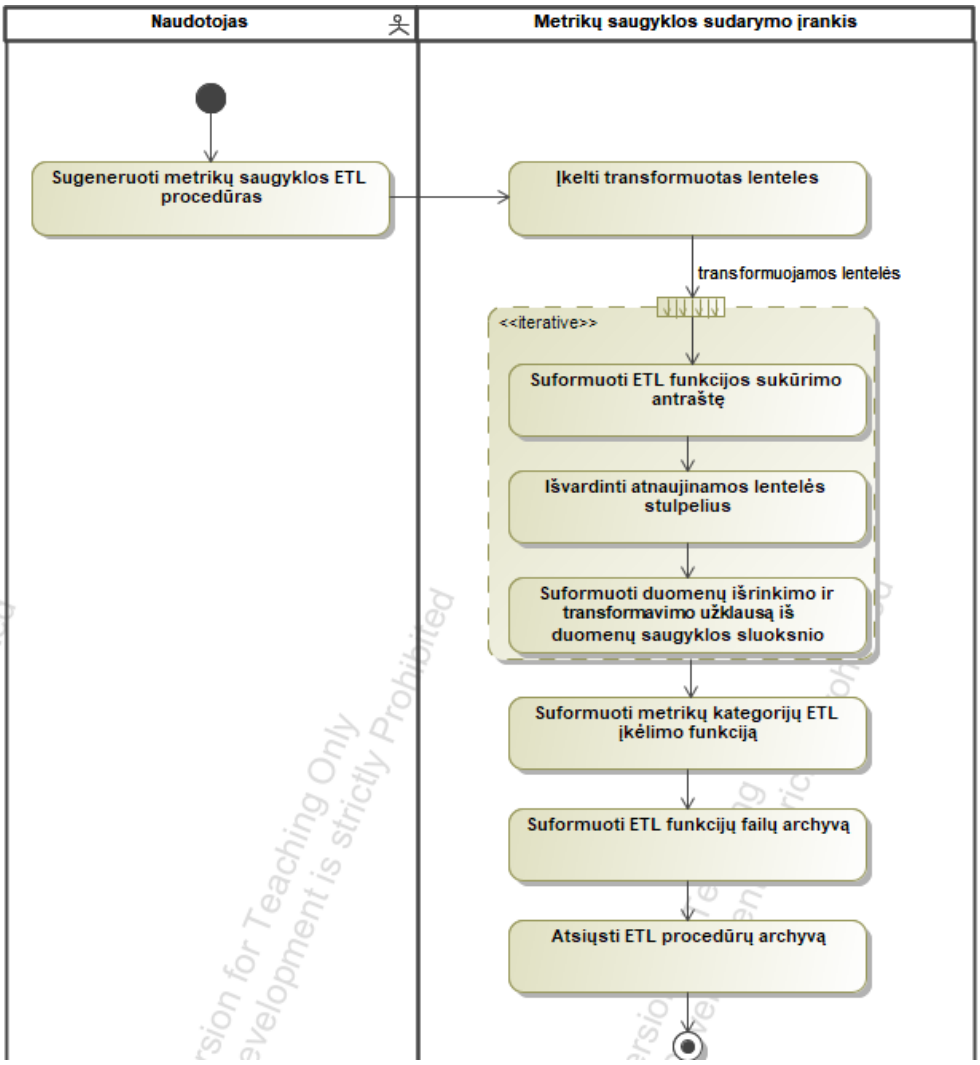


3.5 pav. Metrikų saugyklos struktūros *DDL* kodo sugeneravimo panaudojimo atvejo veiklos diagrama

DDL kodo generavimas pradedamas transformuotų lentelių informacijos įkėlimu. Tada yra sugeneruojamos komandos skirtos pašalinti metrikų saugykloje dubliuojančias lenteles. Po to yra

sugeneruojamos centrų lentelių sukūrimo funkcijos, po centrų sukuriamos sąsajos ir galiausiai sukuriami centrai. Aprašius visas šias *DDL* funkcijas jos yra išsaugojamos atskirame faile ir suformuojamas archyvas, kuris atsiunčiamas naudotojui.

ETL procedūrų generavimo procesą aprašyti buvo sudaryta veiklos diagrama pateikiama 3.6 paveiksle. Generavimas taip pat pradedamas transformuotų lentelių informacijos įkėlimu. Tada kiekvienai lentelei yra suformuojama *ETL* funkcija, kuri įkels duomenis į ją. Funkcijos kūrimas pradedamas jos antraštės suformavimu, tada išvardinama lentelės struktūrą (jos stulpeliai) ir aprašoma duomenų išrinkimo iš *Data Vault* duomenų saugyklos sluoksnio užklausa, kuri jei tai paprastas stulpelis tiesiog išrenka reikšmę, jei išskaičiuojamasis stulpelis pritaiko naudotojo aprašytą funkciją. Kiekviena suformuota *ETL* procedūra išsaugojama atskirame faile. Taip pat kadangi metrikų kategorijų sąrašas yra pastovus ir nepriklausomas nuo *Data Vault* duomenų saugyklos sluoksnio, yra sukuriama iš anksto aprašyta metrikų kategorijų įkėlimo *ETL* procedūra. Visi *ETL* procedūrų failai yra sujungiami į vieną archyvą ir atsiunčiami naudotojui.



3.6 pav. Metrikų saugyklos *ETL* procedūrų generavimo panaudojimo atvejo veiklos diagrama

Tokiu būdu metrikų saugyklos sudarymo įrankis įgyvendina visus metrikų saugyklos sudarymo metodikos panaudojimo atvejus, skirtus metrikų saugyklos sluoksnio sudarymui.

3.2. *Data Vault* metrikų saugyklos sudarymo įrankio nefunkciniai reikalavimai

Data Vault metrikų saugyklos sukūrimui be jau aprašytų panaudojimo atvejų funkcinių reikalavimų turėtų būti įgyvendinti ir tam tikri nefunkciniai reikalavimai. Šie reikalavimai aprašyti žemiau (žr. 9 lentelė - 11 lentelė) ir apima reikalavimus keliamus *Data Vault* architektūrai, procesų automatizacijai ir metrikų kategorizavimui.

9 lentelė. *Data Vault 2.0* architektūros naudojimo nefunkcinis reikalavimas

Reikalavimas	Naudojama *Data Vault 2.0* architektūra ir duomenų modelis
Aprašymas	Realizacija turi būti parengta naudojant *Data Vault 2.0* architektūrą ir *Data Vault* duomenų modelį
Pagrindimas	Metrikų saugykla skirta *Data Vault 2.0* duomenų saugyklai, dėl to turi būti naudojami šiai saugyklai sudaryti skirta architektūra ir standartinis duomenų modelis
Prioritetas	Aukštas

10 lentelė. *Data Vault 2.0* architektūros naudojimo nefunkcinis reikalavimas

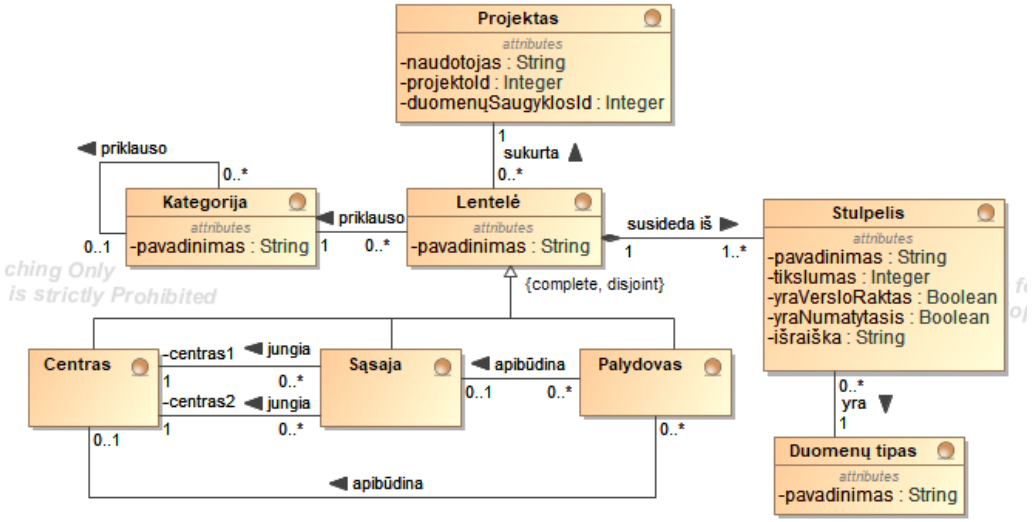
Reikalavimas	*Data Vault 2.0* duomenų paruošimo ir duomenų saugojimo sričių automatizacija *VaultSpeed*
Aprašymas	Duomenų paruošimo srities duomenų įkėlimo ir paruošimo sluoksniai, bei duomenų saugojimo srities *Data Vault* sluoksnis turi būti sudaryti pasitelkiant *VaultSpeed* duomenų saugyklos sudarymo automatizacijos įrankį
Pagrindimas	*Data Vault* duomenų saugykloms sudaryti jau egzistuoja automatizacijos įrankiai, kurie stipriai palengvina ir pagreitina šios saugyklos sudarymo ir įdiegimo procesą. *VaultSpeed* siūloma naudoti dėl to, nes turima prieiga prie šio įrankio ir galima turėti konsultacijas su šio įrankio specialistais
Prioritetas	Vidutinis

11 lentelė. Metrikų taksonomijos naudojimo nefunkcinis reikalavimas

Reikalavimas	Metrikų taksonomija
Aprašymas	Įrankis metrikų kategorijoms parinkti turi naudoti metrikų saugyklos metodikoje pateiktą metrikų taksonomiją.
Pagrindimas	Naudotojui norint nustatyti problemą yra naudinga metrikas analizuoti įvairiomis skirtingomis kategorijomis. Šios kategorijos padeda greičiau identifikuoti problemos šaltinį.
Prioritetas	Aukštas

3.3. *Data Vault* metrikų saugyklos sudarymo įrankio dalykinės srities modelis

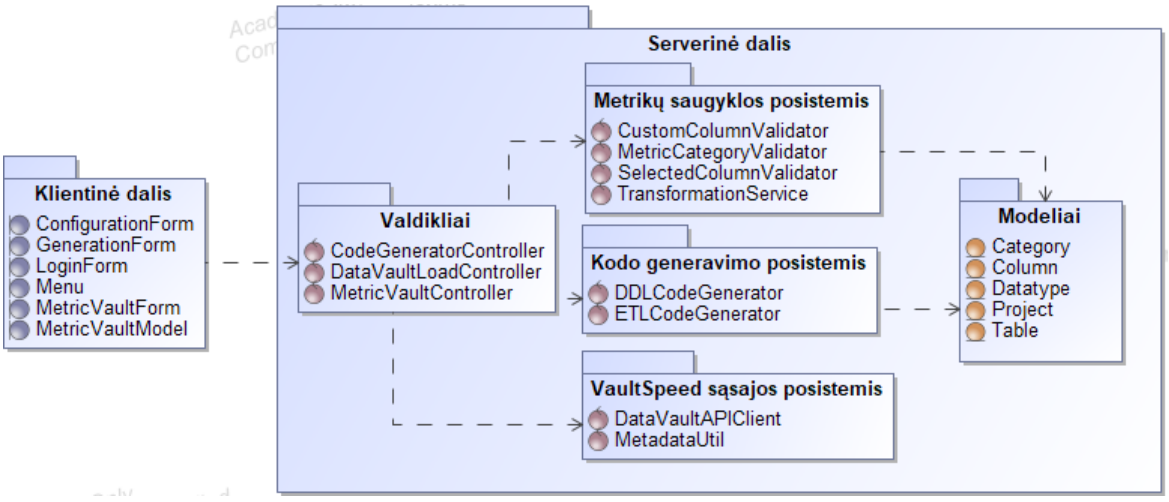
Metrikų saugyklos sudarymo įrankio duomenis apibūdina 3.7 pav. pateiktas dalykinės srities modelis. Šis modelis aprašo tai kokios lentelės (t. y. centrai (angl. *hubs*), sąsajos (angl. *links*) ir palydovai (angl. *satellites*) turi būti sukurti, kokiai kategorijai jie priskirti, su kokiu *VaultSpeed* duomenų saugyklos projektu yra susijusios šios lentelės, kokie stulpeliai sudaro šias lenteles, kokie jų tipai ir ar jie yra apskaičiuojami pagal kažkokią išskaičiuojamąją funkciją.



3.7 pav. Metrikų saugyklos sudarymo įrankio duomenų modelis

3.4. *Data Vault* metrikų saugyklos sudarymo įrankio loginė architektūra

Metrikų saugyklos sudarymo įrankio loginė architektūra išskaidoma į dvi pagrindines dalis: klientinę dalį atsakingą už grafinę sąsają ir serverinę dalį atsakingą už veiklos logiką. Klientinės dalies struktūra yra gana paprasta ir apima tik naudotojui pateikiamų formų komponentus. Serverinė dalis sudėtingesnė ir susideda iš kelių posistemių. Realizuojamas metrikų saugyklos sudarymo įrankis apims tris pagrindinius procesus – metrikų saugyklos sudarymą, *DDL* struktūros ir *ETL* procesų komandų generavimą. Todėl realizuojamo sprendimo loginę architektūrą sudarys trys pagrindiniai posistemiai: metrikų saugyklos posistemis, kodo generavimo posistemis, ir pagalbinis posistemis skirtas komunikacijai su *VaultSpeed*. Pradiniai *Data Vault* saugyklos sudarymo veiksmai bus atliekami *VaultSpeed* duomenų saugyklos automatizacijos įrankiu, todėl suformuotų duomenų išgavimui bus panaudojama šio įrankio sąsaja. Visą realizuojamo įrankio loginę architektūrą aprašo 3.8 paveiksle pavaizduota diagrama. Diagramoje galima matyti, kad sistema bus realizuojama pasitelkiant *MVC* (modelių-vaizdų-valdiklių) architektūrą. Valdikliai ir vaizdai bus išskaidomi pagal išvardintus posistemius, o modeliai bus saugojami bendriniame pakete, kadangi tarp posistemių naudojamos struktūros išlieka vienodos.



3.8 pav. Metrikų saugyklos sudarymo įrankio loginė architektūra

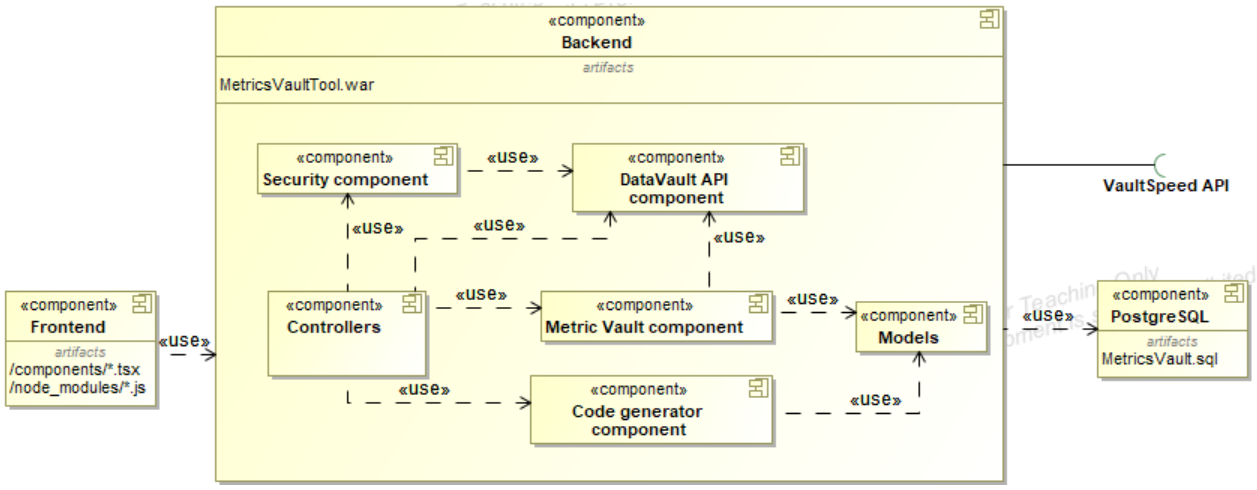
4. *Data Vault* metrikų saugyklos sudarymo įrankio realizacija ir testavimas

4.1. *Data Vault* metrikų saugyklos sudarymo įrankio komponentų modelis

Metrikų saugyklos sudarymo įrankio struktūrą aprašo komponentų diagrama pateikta 4.1 paveiksle. Įrankis susideda iš dviejų pagrindinių realizuojamų dalių: klientinės dalies (angl. *frontend*) ir serverinės dalies (angl. *backend*). Klientinė dalis realizuojama *React* karkaso pagalba naudojant *Typescript* programavimo kalbą. Šis komponentas atsakingas už naudotojui pateikiamą programos grafinę sąsają ir sąveiką su naudotojo vykdomais veiksmais.

Už sistemos veiklos logiką atsakinga serverinė dalis. Šis komponentas susideda iš kelių smulkesnių komponentų. Sistemos valdiklių (angl. *controllers*) komponentas skirtas priimti užklausas ateinančias iš klientinės dalies reaguojant į naudotojo veiksmus. Komponentas priklausomai nuo kviečiamos funkcijos iškviečia kitų komponentų funkcijas, kurios įvykdo numatytas funkcijas, grąžina rezultatą, kuris vėliau valdiklio yra persiunčiamas klientinei daliai. Saugumo komponentas (angl. *security component*) atsakingas už siunčiamų užklausų patikrinimą siekiant užtikrinti sistemos saugumą. Kiekviena užklausa siunčiama į serverinę dalį praeina pro šį komponentą ir yra patikrinama ar yra autorizuota pasiekti kviečiamą valdiklio funkciją. Komunikacijai su *VaultSpeed* įrankio sąsaja egzistuoja atskiras komponentas (diagramoje *DataVault API component*), kuris vykdo visas užklausas į *VaultSpeed*. Šis komponentas naudojamas naudotojo prisijungimui prie *VaultSpeed* paskyros, *VaultSpeed* įrankyje sukurtų projektų ir duomenų saugyklų informacijai gauti ir pasirinktos *Data Vault* duomenų saugyklos metaduomenų atsisiuntimui. Už metrikų saugyklos sudarymą/projektavimą yra atsakingas metrikų saugyklos (angl. *metric vault component*) komponentas. Šis komponentas išsaugo naudotojo atliekamas metrikų saugyklos lentelių transformacijas, atlieka iš klientinės dalies atsiunčiamų transformacijų validaciją, suformuoja metrikų kategorijų lenteles. Paskutinis už veiklos logiką atsakingas komponentas yra kodo generavimo komponentas (angl. *code generator component*). Šis komponentas skirtas sugeneruoti *DDL* komandas ir *ETL* procedūras, kurios naudojamos metrikų saugyklos sukūrimui ir užpildymui duomenų saugykloje. Galiausiai, egzistuoja modelių komponentas (angl. *models*), kuris skirtas aprašyti duomenų bazėje saugojamų duomenų struktūras, kurios naudojamos metrikų saugyklos sudarymo įrankyje duomenis skaitant, išsaugant ir atliekant kitas įvairias duomenų manipuliacijas.

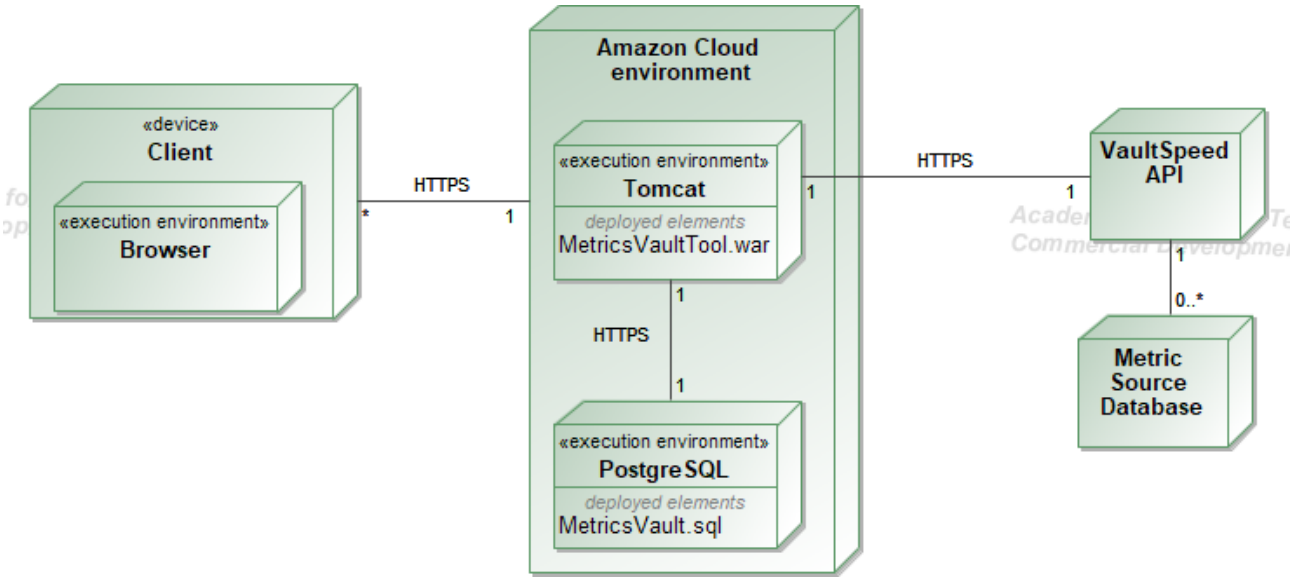
Neskaitant metrikų sudarymo įrankio komponentų taip pat matomi ir kiti du svarbūs komponentai. *PostgreSQL* duomenų bazių valdymo sistemos komponentas, kurioje saugoma metrikų saugyklos sudarymo įrankio duomenų bazė. Ir taip pat pateikiamas *VaultSpeed API* komponentas per kurį yra vykdoma komunikacija su *VaultSpeed* įrankiu siekiant gauti metrikų saugyklos sudarymo įrankiui reikalingus duomenis.



4.1 pav. Metrikų saugyklos sudarymo įrankio komponentų diagrama

4.2. *Data Vault* metrikų saugyklos sudarymo įrankio diegimo modelis

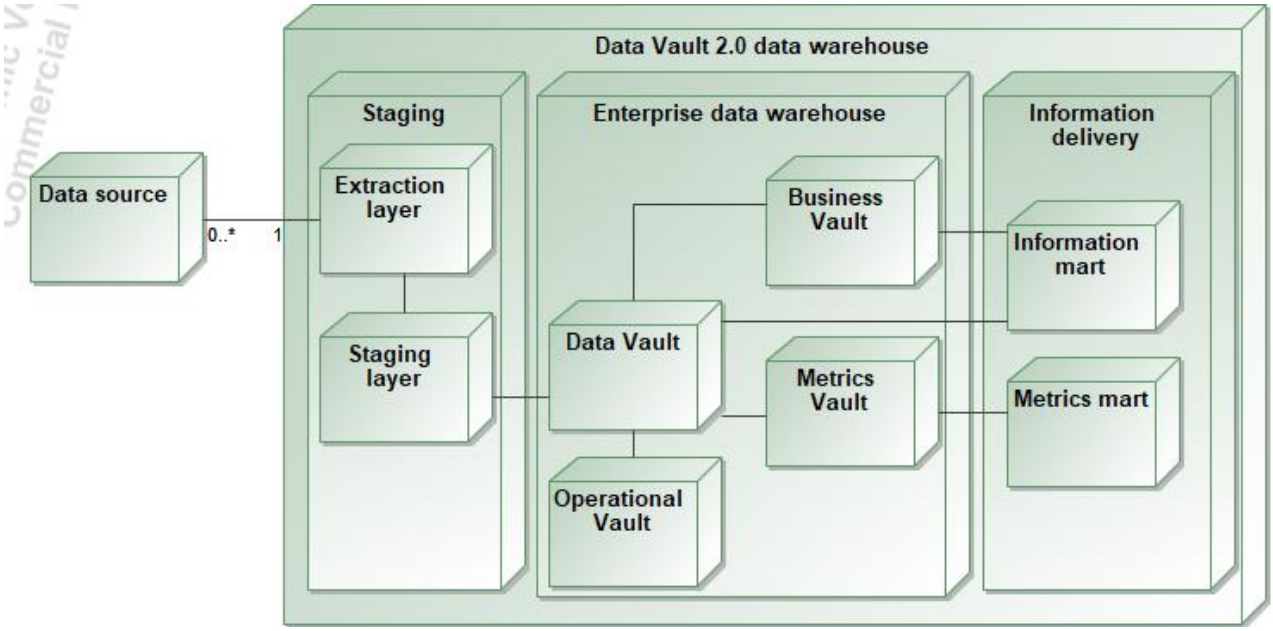
Metrikų saugyklos sudarymo įrankio diegimą aprašo 4.2 paveiksle pateikta diegimo diagrama. Įrankis yra diegiamas į *Amazon* debesijos paslaugų aplinką. Pats įrankis yra įdiegiamas į *Tomcat* serverį, o duomenų bazė į *PostgreSQL* duomenų bazių valdymo sistemą, kuri taip pat paleidžiama *Amazon* debesijoje. Kadangi įrankis yra internetinis puslapis, jis gali būti pasiekiamas kliento per naršyklę naudojantis *HTTPS* protokolu. Taip pat, kadangi įrankis naudojasi *VaultSpeed* automatizacijos įrankio duomenimis, šis įrankis taip pat turi ryšį su *VaultSpeed API*.



4.2 pav. Metrikų saugyklos įrankio diegimo diagrama

Įrankio pagalba realizuota duomenų saugykla yra įdiegiama duomenų saugyklos valdymo aplinkoje. Duomenų saugykla kaip ir minėta ankstesniuose skyriuose sudaro trys pagrindinės sritys: duomenų paruošimo sritis, duomenų saugojimo sritis ir informacijos perdavimo sritis. Duomenų paruošimo srityje yra sukuriami du sluoksniai: duomenų įkėlimo (angl. *extraction layer*) ir duomenų paruošimo sluoksnis (angl. *staging layer*). Šie sluoksniai yra tarpiniai ir saugo pradinę duomenų šaltinio

informaciją, kuri su kiekvienu nauju duomenų įkėlimo ciklu yra išvaloma. Duomenų saugojimo srityje yra sukuriamas *Data Vault* sluoksnis, skirtas pastoviam duomenų saugojimui *Data Vault* duomenų modelio pavidalu, į kurį duomenys įkeliami iš duomenų paruošimo sluoksnio pritaikius *Data Vault* modelio transformavimo taisykles. Duomenų saugojimo srityje taip pat gali egzistuoti trys papildomos saugyklos: veiklos duomenų saugykla (angl. *business vault*), operacinio lygio duomenų saugykla (angl. *operational vault*) ir metrikų saugykla (angl. *metrics vault*). Tiek veiklos, tiek metrikų saugykla formuojama iš *Data Vault* sluoksnio. O operacinio lygio duomenų saugykla yra skirta operacinio lygio sistemoms, kurios įterpia duomenis iš karto į *Data Vault* sluoksnį. Galiausiai, paruošiama informacijos perdavimo sritis. Šioje srityje daugiausiai saugojami veiklos duomenų pjūviai (angl. *information mart*), kurie yra formuojami veiklos duomenų saugyklos ir *Data Vault* sluoksnio pagrindu. Tačiau, metrikų saugyklos kontekste, saugojami ir įvairių sistemos metrikų pjūviai (angl. *metrics mart*), skirti pateikti sistemos metrikų informaciją aktualiais aspektais.



4.3 pav. *Data Vault* duomenų saugyklos diegimo architektūra su metrikų saugykla

4.3. *Data Vault* metrikų saugyklos sudarymo įrankio realizacijos ir veikimo aprašymas

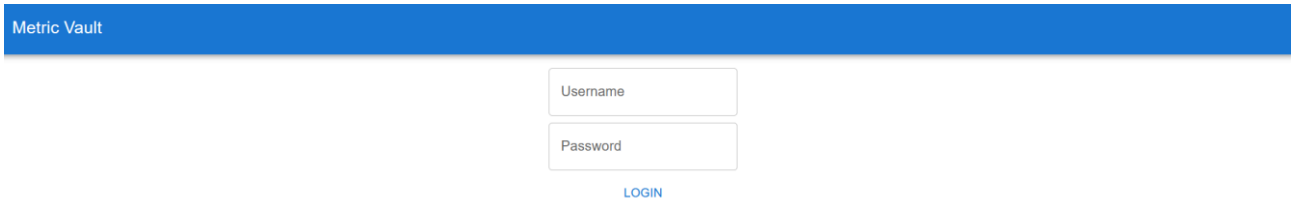
4.3.1. Paskirtis

Metrikų saugyklos sudarymo įrankio pagrindinė paskirtis automatizuoti anksčiau metrikų saugyklos sudarymo metodikoje aprašytą metrikų saugyklos sudarymo procesą. Todėl šio įrankio pagrindinės funkcijos apima *Data Vault* sluoksnio duomenų modelio įkėlimą, metrikų saugyklos suprojektavimą ir suprojektuotos metrikų saugyklos *DDL* ir *ETL* kodo sugeneravimą. Visos kitos pradinės *Data Vault* duomenų saugyklos automatizacijos gali būti atliekamos egzistuojančių įrankių pagalba kaip *VaultSpeed* (kuriuo ir yra naudojamasi šio įrankio realizacijos procese).

4.3.2. Prisijungimas ir sistemos navigavimas

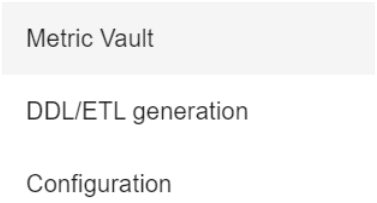
Kadangi metrikų saugyklos sudarymo įrankis kuriamas kaip naujas įrankis, tačiau naudojantis *VaultSpeed* įrankio duomenis, prisijungimas prie metrikų saugyklos sudarymo įrankio realizuojamas pasitelkiant *VaultSpeed* programinę sąsają (angl. *application programming interface – API*). Todėl norint prisijungti prie metrikų saugyklos sudarymo įrankio reikia turėti aktyvią *VaultSpeed* paskyrą.

Taip pat visas naudotojo autentifikacijos ir autorizacijos procesas atliekamas *VaultSpeed* įrankio pusėje, o metrikų saugyklos sudarymo įrankis veikia kaip tarpininkas. Taip yra daroma todėl, nes metrikų saugyklos sudarymui yra reikalingas *Data Vault* sluoksnis, kuris yra sudarytas *VaultSpeed* platformoje. Todėl *API* pagalba šis modelis yra atsiunčiamas iš *VaultSpeed* sistemos. Sistemos prisijungimo langas pateikiamas 4.4 paveiksle.



4.4 pav. Metrikų saugyklos sudarymo įrankio prisijungimo langas

Prisijungus prie įrankio pirmiausia naudotojui pateikiamas metrikų saugyklos projektavimo langas, tačiau naudotojas gali naviguoti tarp skirtingų sistemos langų atsidaręs meniu pasiekiamą viršutiniame kairiajame įrankio lango kampe. Egzistuoja trys meniu parinktys: metrikų saugyklos projektavimo langas (pavadinimu *Metric Vault*), automatinio *DDL* ir *ETL* kodo generavimo langas (pavadinimu *DDL/ETL generation*) ir sistemos/projekto konfigūracijos langas (pavadinimu *Configuration*). Meniu parinktys matomos 4.5 paveiksle.

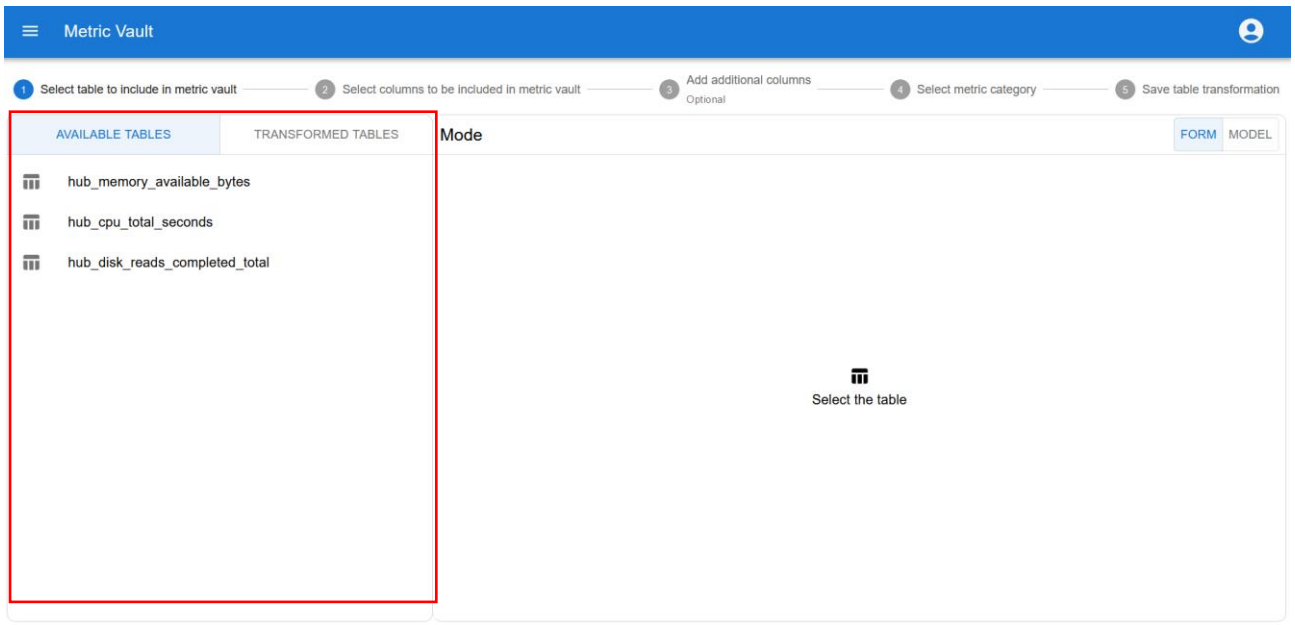


4.5 pav. Metrikų saugyklos sudarymo įrankio grafinės sąsajos meniu parinktys

Atsijungimas nuo sistemos pasiekiamas paspaudus ant naudotojo ikonos ir pasirinkus atsijungimo (pavadinimu *Logout*) parinktį.

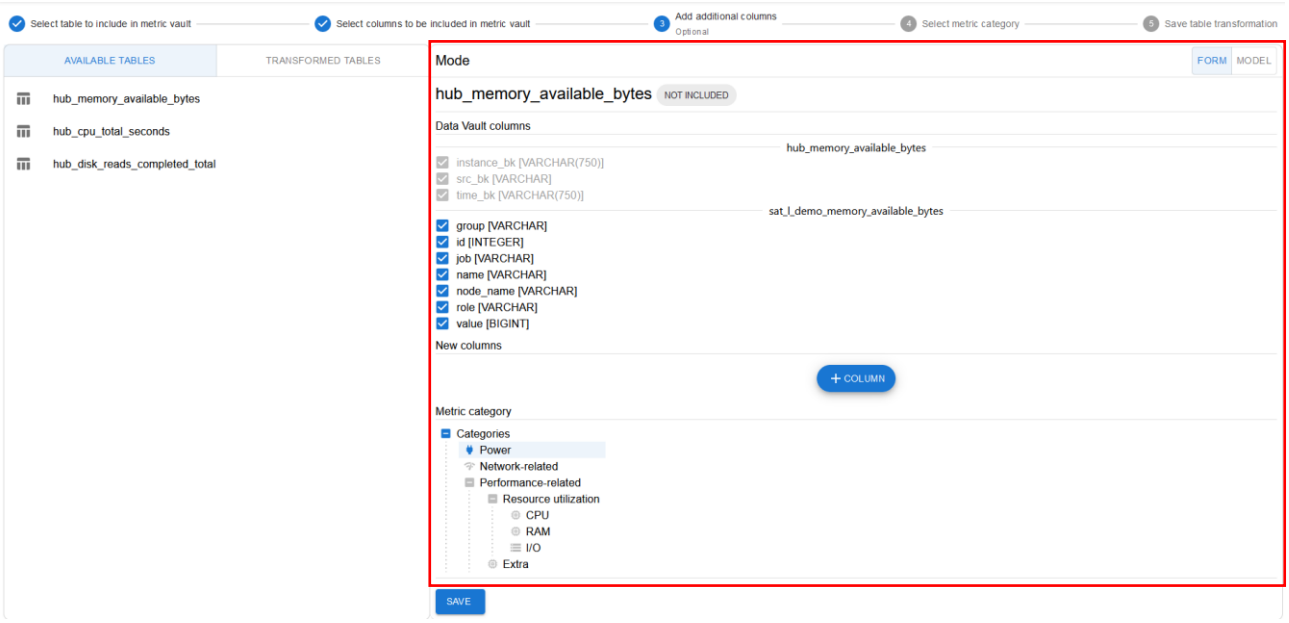
4.3.3. Metrikų saugyklos modeliavimas

Pagrindinė įrankio paskirtis sudaryti metrikų saugyklą, kurios *DDL* ir *ETL* kodas vėliau yra automatiškai sugeneruojamas ir įdiegiamas duomenų saugykloje. Norint naudotojui pradėti sudarinėti metrikų saugyklą yra reikalingas jau sudarytas *Data Vault* sluoksnis *VaultSpeed* platformoje. Kai šis sluoksnis yra sudarytas, metrikų sudarymo įrankis naudodamasis *VaultSpeed API* atsisiunčia visą šio sluoksnio struktūrą (t. y. centrus, sąsajas ir jų palydovus) ir pateikia visus egzistuojančius centrus kairiajame krašte esančioje srityje (sąsajos ir palydovai nepateikiami).



4.6 pav. Metrikų saugyklos sudarymo langas (prieinami centrai apibraukti raudonai)

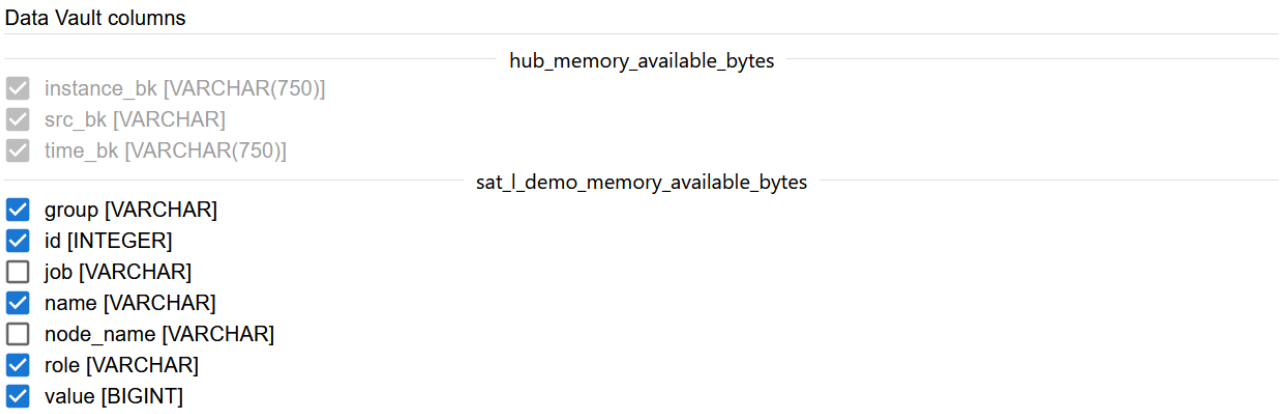
Įkėlus visus *Data Vault* sluoksnyje esančius centrus, galima pradėti metrikų saugyklos projektavimą. Pirmiausia, iš visų prieinamų centrų išsirenkamas centras, kurį norima įtraukti į metrikų saugyklą. Pasirinkus centrą, atsinaujina dešinėje srityje vaizduojama forma (žr. 4.7 pav.). Ši forma pateikia tokius duomenis: pasirinkto centro pavadinimą, jam priklausančius stulpelius, prie centro prijungtų palydovų stulpelius ir parinktis pridėti papildomus išskaičiuojamuosius stulpelius bei nurodyti metrikos kategoriją.



4.7 pav. Metrikų saugyklos sudarymo langas (forma apibraukta raudonai)

Kiekvienas egzistuojantis stulpelis turi parinktį būti įtrauktas arba pašalintas iš metrikų saugyklos. Tai nurodama naudotojui uždedant arba nuimant varnelę prie nurodyto stulpelio (žr. 4.8 pav.). Verslo rakto stulpeliai yra visada įtraukiami ir neturi galimybės būti atžymėti, todėl visi pasirinkto centro

stulpeliai yra visada įtraukiami. Numatytieji stulpeliai kaip įkėlimo laikas (angl. *load time*), įkėlimo ciklo identifikatorius (angl. *load cycle id*), verslo rakto stulpelis (stulpeliai pasibaigiantys *_hkey*) yra visada automatiškai sugeneruojami ir pridedami kiekvienam duomenų centrui, sąsajai ar palydovui, todėl formoje jie atskirai nėra pateikiami.



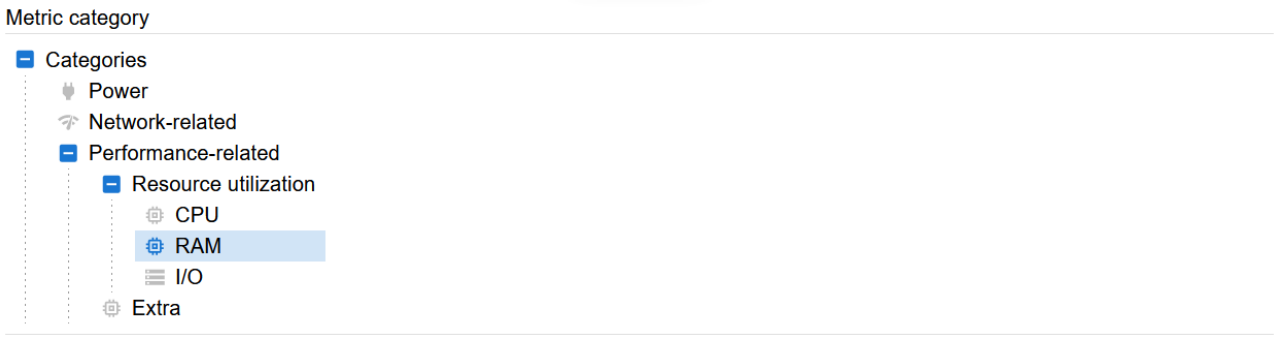
4.8 pav. Pasirinktini stulpeliai metrikų saugyklos sudarymo formoje

Formoje, neskaitant jau egzistuojančių stulpelių, taip pat galima pridėti ir naujus išskaičiuojamuosius stulpelius, kurių reikšmė priklauso nuo kitų stulpelių jiems pritaikant kažkokią transformaciją (pavyzdžiui padauginant, padalinant, pritaikant funkciją). Norint pridėti išskaičiuojamąjį stulpelį, reikia nurodyti stulpelio pavadinimą, palydovą, kuriame bus saugomas šis stulpelis, stulpelio duomenų tipą ir išraišką kaip yra gaunama šio stulpelio reikšmė (žr. 4.9 pav.). Išraiška aprašoma naudojantis *SQL* standarto funkcijomis (sudėtis (+), atimtis (-), daugyba (*), dalyba (/) bei papildomos pagalbinės funkcijos: *LENGTH, LOWER, UPPER, SUBSTRING, ABS, SQRT, FLOOR, CEILING*, sąlygos operacija *CASE*), o nuorodos į stulpelių reikšmes aprašomos nurodant stulpelio pavadinimą apskliaustą laužtiniais skliaustais (pavyzdys: {*stulpelis*}). Taigi, pavyzdinis išskaičiuojamasis stulpelis kito stulpelio reikšmę padalinantis iš 1000 būtų: {*stulpelis*}/1000.

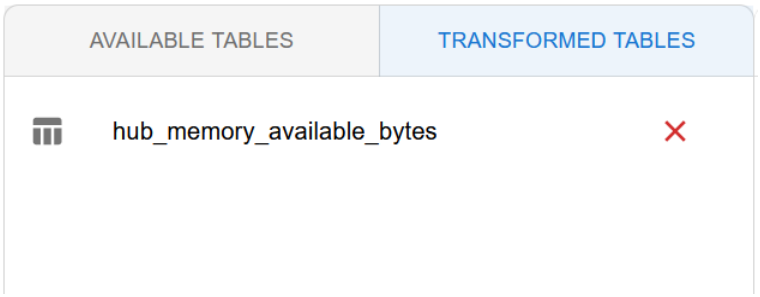
4.9 pav. Išskaičiuojamo stulpelio forma

Galiausiai, formoje pasirinktam centrui turi būti nurodoma metrikos kategorija (žr. 4.10 pav.), kuriai jis priklauso (daugiau apie metrikų kategorijų hierarchiją pateikta 2.3 poskyryje). Parinkus metrikos kategoriją, nurodžius įtrauktinus egzistuojančius stulpelius bei pridėjus naujus išskaičiuojamuosius stulpelius, forma yra išsaugoma pasirenkant išsaugojimo mygtuką (pavadinimu *Save*). Transformacija yra išsaugojama ir transformuotas centras perkeliamas iš prieinamų centrų skyriaus į transformuotų centrų skyrių. Transformuotą centrą bet kada galima atnaujinti jį tiesiog vėl pasirenkant iš sąrašo, atliekant transformavimo veiksmus ir išsaugant atnaujintą struktūrą. Jei

transformacija nebereikalinga ar panašiai, ji gali būti pašalinta paspaudžiant ant *X* mygtuko šalia centro (žr. 4.11 pav.).



4.10 pav. Metrikos kategorijos forma

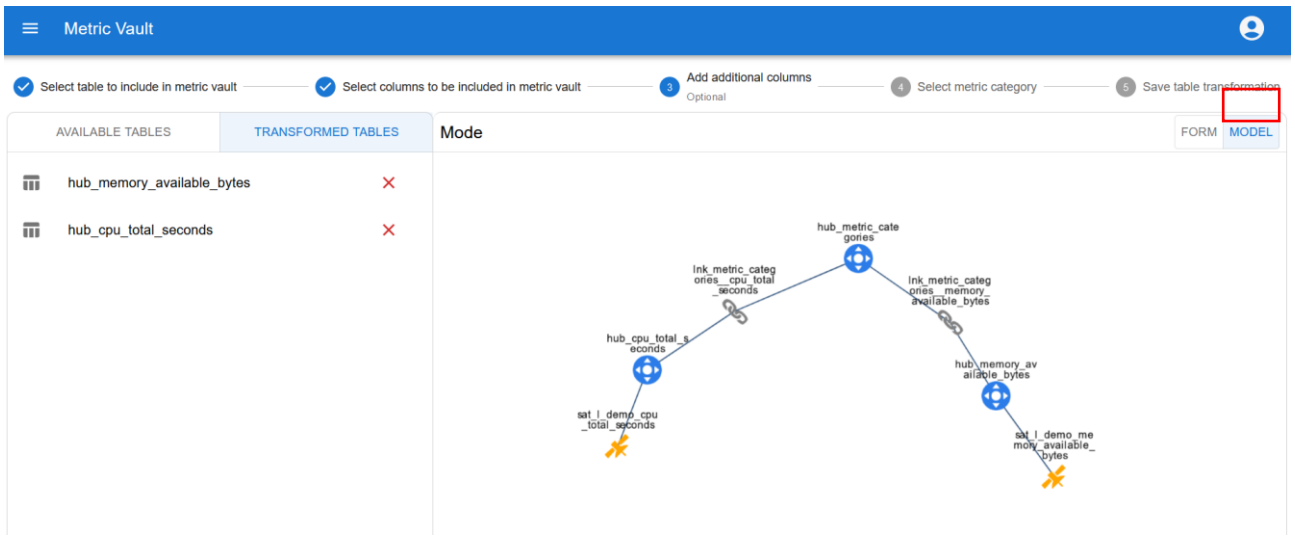


4.11 pav. Transformuotas centras

Jeigu metrikų saugyklos sudarymo etapai užsimiršta, visada viršuje yra pateikiamas gidas, kaip atlikti transformaciją (žr. 4.12 pav.).

4.12 pav. Metrikų saugyklos sudarymo gidas

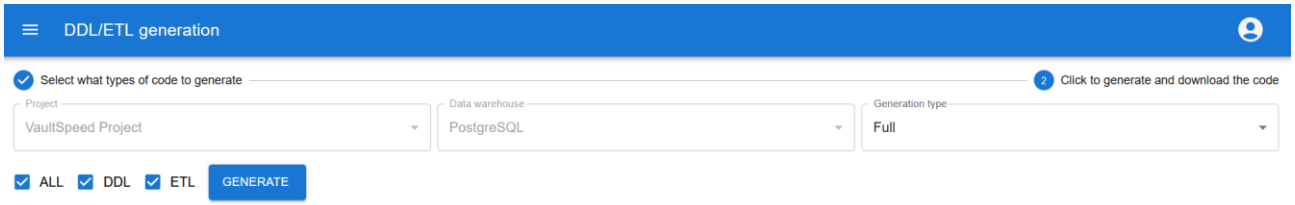
Norint peržiūrėti tarpinį metrikos saugyklos vaizdą – transformuotus centrus, sąsajas, jų palydovus ir ryšius tarp jų, galima įsijungti metrikų saugyklos modelio peržiūros langą. Lange pateikiamas tuo metu esantis jau suprojektuotas metrikų saugyklos modelis, kuris būtų gaunamas įvykdžius automatinį kodo generavimą (žr. 4.13 pav.).



4.13 pav. Sumodeliuotos metrikų saugyklos modelis (modelio peržiūros režimas apibrauktas raudonai)

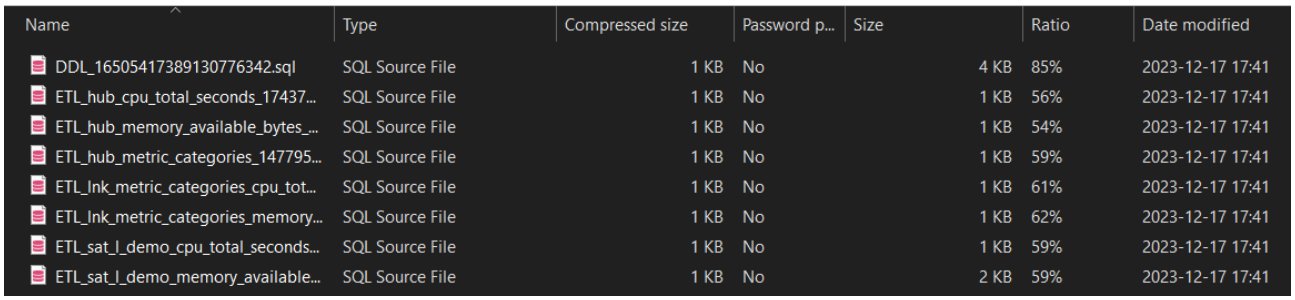
4.3.4. Automatinis *DDL* ir *ETL* kodo generavimas

Baigus modeliuoti metrikų saugyklą galima automatiškai sugeneruoti *DDL* ir *ETL* kodą skirtą sukurti metrikų saugyklą ir įkelti duomenis į ją. Tai vykdoma iš kodo generavimo lango pavaizduoto 4.14 pav. Pasirenkama ar bus norima generuoti tik *DDL* kodą, tik *ETL* kodą ar abu ir spaudžiamas generavimo mygtukas (pavadinimu *Generate*).



4.14 pav. *DDL* ir *ETL* kodo generavimo langas

Pasirinkus generuoti yra sugeneruojamas kodas ir atsiunčiamas *.zip* failų archyvas, kuriame yra sugeneruoti *DDL* ir *ETL* kodų failai. Sugeneruoto archyvo pavyzdys pateiktas 4.15 pav.

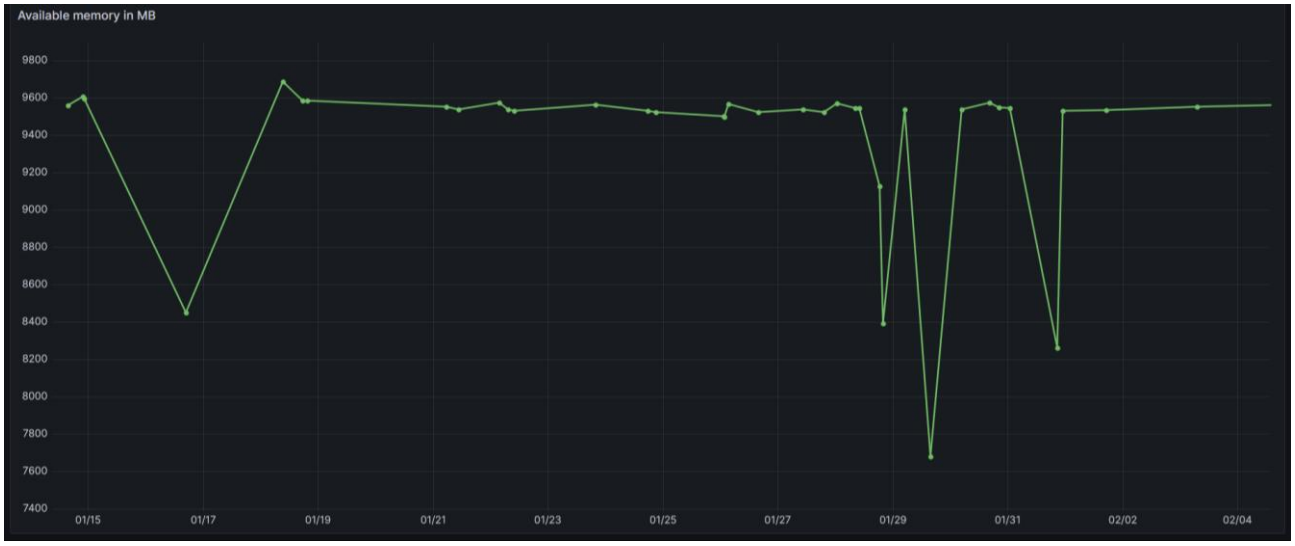


Name	Type	Compressed size	Password p...	Size	Ratio	Date modified
DDL_1650541738913077634 2.sql	SQL Source File	1 KB	No	4 KB	85%	2023-12-17 17:41
ETL_hub_cpu_total_seconds_17437...	SQL Source File	1 KB	No	1 KB	56%	2023-12-17 17:41
ETL_hub_memory_available_bytes_...	SQL Source File	1 KB	No	1 KB	54%	2023-12-17 17:41
ETL_hub_metric_categories_147795...	SQL Source File	1 KB	No	1 KB	59%	2023-12-17 17:41
ETL_lnk_metric_categories_cpu_tot...	SQL Source File	1 KB	No	1 KB	61%	2023-12-17 17:41
ETL_lnk_metric_categories_memory...	SQL Source File	1 KB	No	1 KB	62%	2023-12-17 17:41
ETL_sat_l_demo_cpu_total_seconds...	SQL Source File	1 KB	No	1 KB	59%	2023-12-17 17:41
ETL_sat_l_demo_memory_available...	SQL Source File	1 KB	No	2 KB	59%	2023-12-17 17:41

4.15 pav. Sugeneruoto *DDL* ir *ETL* kodo failų archyvas

Šis kodas turi būti įvykdomas duomenų saugykloje sukurti metrikų saugyklai ir įkelti duomenis. Tai atlikus yra suformuojamas metrikų duomenų pjūvis ir iš jo duomenys gali būti panaudojami įvairių vizualizacijų sudarymui. Vienas panaudojimo pavyzdys pateikiamas 4.16 pav. Šiuo atveju

vizualizacija buvo sudaryta *Grafana* įrankyje. Ji skirta atvaizduoti laisvos atminties resursus baitais laike.



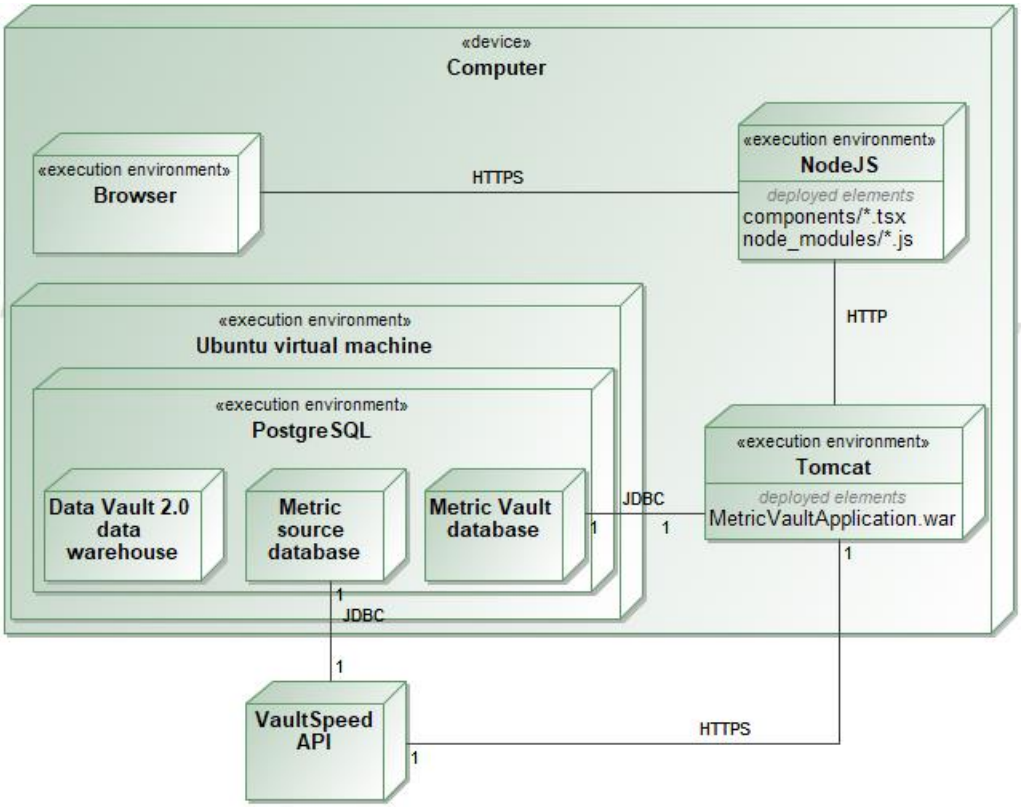
4.16 pav. *Grafana* įrankyje sudaryta vizualizacija laisvos atminties kiekiui laike atvaizduoti

4.4. *Data Vault* metrikų saugyklos sudarymo įrankio testavimo modelis

Siekiant patikrinti, ar realizuotas metrikų saugyklos sudarymo įrankis veikia tinkamai, buvo sudarytas testavimo planas. Testavimo tikslas – patikrinti, ar metrikų saugyklos sudarymo įrankio pagalba naudotojas gali suprojektuoti metrikų saugyklos struktūrą, nurodyti įtrauktinas metrikas, jų duomenis, pridėti išskaičiuojamuosius stulpelius ir sugeneruoti *DDL* ir *ETL* komandas.

Testavimui vykdyti buvo pasirinkta naudoti tiek automatinį, tiek rankinį testavimą. Automatiniam testavimui vykdyti pasitelkta *JUnit5* biblioteka automatinių testų vykdymui, o automatiniai vienetų testai buvo sudaryti naudotojo atliekamų transformavimo veiksmų suformavimo ir validacijos tikslumui patikrinti. Aprašyti vienetų testai pateikiami prieduose (žr. 1 priedas). Rankinis testavimas buvo pasirinktas patikrinti bendrai sistemos integracijai ir panaudojimo atvejų scenarijams. Testavimu buvo vykdomi žingsniai pagal panaudojimo atvejų scenarijus ir lyginama ar gaunami rezultatai sutapo su numatytais. Taip pat buvo patikrinami atvejai kuomet naudotojas įveda klaidingus duomenis ir ar dėl to buvo pateikiami klaidos pranešimai. Taip pat sugeneruotos *DDL* ir *ETL* komandos buvo įvykdomos testavimui skirtoje *Data Vault 2.0* duomenų saugykloje ir tikrinama ar įvykdytos komandos suformuodavo tinkamas duomenų lenteles ir į jas įkeldavo tinkamus metrikų duomenis. Testavimui buvo pasitelkti pavyzdiniai realaus serverio darbo duomenys apimantys serverio procesoriaus darbo informaciją, laisvos operatyviosios atminties kiekio kitimo laike informaciją ir išorinės atminties skaitymo ir rašymo operacijų informaciją.

Testavimui vykdyti buvo pasitelkta lokali aplinka, kurios diegimo modelis pateikiamas 4.17 pav. Ši aplinka imituoja numatomo veikimo diegimo aplinką, tik yra vykdoma lokaliame kompiuteryje, o ne debesijos paslaugų aplinkoje. Taip pat klientinė ir serverinė dalys yra išskirstytos į du mazgus, bet sistemos veikimo logikos tai nekeičia. Metrikų šaltinio duomenys yra saugojami *Ubuntu* virtualioje mašinoje *PostgreSQL* duomenų bazių valdymo sistemoje metrikų šaltinio duomenų bazėje. Testavimo *Data Vault 2.0* duomenų saugykla sukurta taip pat *PostgreSQL* duomenų bazių valdymo sistemoje atskiroje duomenų bazėje.



4.17 pav. Testavimo aplinkos diegimo modelis

4.5. *Data Vault* metrikų saugyklos sudarymo įrankio diegimo gidas

Metrikų saugyklos sudarymo įrankio diegimas susideda iš 3 pagrindinių žingsnių:
1. metrikų saugyklos įrankio duomenų bazės paruošimo;
2. metrikų saugyklos įrankio serverinės dalies įdiegimo;
3. metrikų saugyklos įrankio klientinės dalies įdiegimo.

Metrikų saugyklos sudarymo įrankio duomenų bazei naudojama *PostgreSQL* (naudota 14 versija) duomenų bazių valdymo sistema. Ši sistema turi būti įdiegta į numatomą metrikų saugyklos sudarymo įrankio vykdymo aplinką. Tada turi būti sukuriama duomenų bazė *metricvault*, kurioje ir bus saugomi visi metrikų saugyklos sudarymo įrankio duomenys. Duomenų bazės lentelės ir ryšiai yra sukuriami įvykdant pateiktą *.sql* formato failą su *SQL* kalbos kodu. Atlikus šiuos veiksmus bus sukurta ir paruošta darbui duomenų bazė.

Metrikų saugyklos sudarymo įrankis yra internetinė sistema sukurta su *SpringBoot* karkasu (*Java* programavimo kalba) ir *React* biblioteka (*Typescript* programavimo kalba). Todėl šiai sistemai vykdyti nuspręsta naudoti *Tomcat* serverį. Taigi, diegimo aplinkoje iš pradžių turi būti įdiegiamas *Tomcat* (naudota 10 versija) serveris. Įdiegus šį serverį, į *Tomcat* serverio failų *webapps* katalogą turi būti perkelti serverio ir klientinės dalies failai. Serverinei daliai turi būti perkeltas *.war* tipo failas, o klientinei daliai turi būti perkeltas *metrics* katalogas su statiniais *Javascript*, *HTML* ir *CSS* kalbų failais. Perkėlus failus reikia iš naujo paleistį *Tomcat* serverį ir sistema turėtų būti prieinama *Tomcat* serverio numatytuoju adresu pridėjus */metrics* adresą naršyklės lauke. Viskam sėkmingai pasileidus sistemos diegimas yra baigtas.

5. *Data Vault* metrikų saugyklos sudarymo metodikos eksperimentas

Norint įvertinti pasiūlytą metrikų saugyklos sudarymo metodiką ir realizuotą įrankį, reikia ištirti kelis aspektus. Pirmiausia, reikia patikrinti, ar pagal pateiktą metodiką galima realizuoti metrikų saugyklą, kurioje būtų saugomi dabartiniai realūs metrikų duomenys. Ir taip pat reikia patikrinti, ar pasiūlyta metodika yra aiški ir skatina metrikų saugyklų kūrimą. Tai pasiekti buvo nuspręsta atlikti du skirtingus eksperimentus. Pirmasis eksperimentas skirtas patikrinti metodikos išbaigtumą ir pritaikomumą sukuriant metrikų saugyklą, kurioje būtų saugojami realios įmonės metrikų duomenys. O antrasis eksperimentas – apklausa, kurios tikslas – surinkti nuomonę apie pasiūlytą metodiką ir realizuotą įrankį iš *Data Vault* duomenų saugyklų sudarymo specialistų.

5.1. *Data Vault* metrikų saugyklos sudarymas eksperimentiniams metrikų duomenims

5.1.1. Eksperimento planas

Patikrinti metrikų saugyklos sudarymo metodikos išbaigtumą ir pritaikomumą, buvo sudaryta metrikų saugykla, pasitelkiant realios įmonės serverių metrikų duomenis.

Eksperimentui įvykdyti buvo sudarytas planas, apimantis šiuos žingsnius:
1. Gauti eksperimentinius duomenis iš realios įmonės serverių ir juose veikiančių virtualių mašinų, juos išsisaugoti eksperimentinėje duomenų bazėje *PostgreSQL* duomenų bazių valdymo sistemoje.
2. Sukurti eksperimento *Data Vault* duomenų saugyklos projektą *VaultSpeed* automatizacijos įrankyje ir suprojektuoti duomenų paruošimo ir duomenų saugojimo sritis.
3. Suprojektuoti metrikų saugyklą pasitelkiant metrikų saugyklos sudarymo įrankį.
4. Sugeneruoti *DDL* ir *ETL* kodą reikalingą *Data Vault* duomenų saugyklos sukūrimui.
5. Įvykdyti *DDL* kodą skirtą sugeneruoti duomenų saugyklos struktūrą bei *ETL* funkcijas skirtas įkelti duomenis į duomenų saugyklą.
6. Sudaryti metrikų duomenų pjūvius ir vizualizacijas skirtas atvaizduoti metrikų duomenis.

Eksperimentui vykdyti nuspręsta pasitelkti lokalų kompiuterį su jame veikiančiu metrikų saugyklos sudarymo įrankiu ir virtualia mašina, kurioje bus sukurta eksperimentinė *Data Vault* duomenų saugykla. Techninės eksperimento vykdymo aplinkos charakteristikos pateiktos 12 lentelėje.

12 lentelė. Eksperimento vykdymo aplinkos techninės charakteristikos

Kompiuteris	*Lenovo Legion 5*
Operacinė sistema	*Windows 10 (10.0.19045)*
Procesorius	*AMD Ryzen 5 5600H*
Vaizdo plokštė	*NVIDIA Geforce 3600 RTX*
Operatyvioji atmintis	*32 GB*
Virtualios mašinos operacinė sistema	*Ubuntu 20.04 LTS*
Duomenų bazių valdymo sistema	*PostgreSQL 14*

5.1.2. Eksperimento rezultatai

Eksperimentui naudojami duomenys buvo gauti iš *VaultSpeed* įmonės. Metrikų duomenys buvo surinkti iš įmonės serveriuose veikiančių virtualių mašinų, skirtų naujų darbuotojų ir potencialių

klientų apmokymui. Iš viso buvo panaudoti 7 duomenų rinkiniai. 5 duomenų rinkiniai skirti aprašyti serverio apkrautumo metrikas ir 2 rinkiniai apibūdinti veiklos metrikas, susijusiąs su sistemos darbu. Detalus duomenų rinkinių aprašymas pateikiamas žemiau.

Pirmas duomenų rinkinys skirtas aprašyti serverio procesoriaus praleidžiamą laiką skirtinguose darbo režimuose atliekant įvairias užduotis (13 lentelė).

13 lentelė. Procesoriaus darbo laiko skirtinguose darbo režimuose metrikos duomenys

Duomenys	Procesoriaus darbo laikas skirtinguose režimuose
Aprašymas	Metrika aprašanti procesoriaus praleistą darbo laiką skirtinguose darbo režimuose. Duomenys kaupti apie 8 skirtingų procesorių darbą. Metrikos duomenys yra kaupiamojo tipo – t. y. saugomi ne pokyčiai tarp matavimų, o nuolat didėjanti reikšmė, kuri gaunama prie paskutinio matavimo pridėjus naujausią pokytį.
Pavyzdinės reikšmės	2023-12-04 00:00:13 (GMT+0) matavimo duomenimis, procesorius nr. “1” iki matavimo momento praleido 305513,31 sekundžių “idle” režime, 1790,21 sekundės “iowait” darbo režime, 0 sekundžių “irq” režime, 0,16 sekundžių “nice” režime, 214 sekundžių “softirq” režime, 0 sekundžių “steal” režime, 3293,6 sekundes “system” režime, 6448,24 sekundes “user” režime.
Metrikos tipas	Procesoriaus (*CPU*) darbo duomenys
Duomenų matavimo periodas	2023-12-04 00:00:13 - 2024-01-03 13:42:58 (GMT+0)
Duomenų matavimo dažnis	Kas 15 sekundžių
Duomenų kiekis	11269888 įrašai
Stulpelių kiekis	12

Antras duomenų rinkinys skirtas aprašyti serverio skaitymo/rašymo (*I/O*) įrenginių įvykdytą skaitymo operacijų kiekį (14 lentelė).

14 lentelė. Disko skaitymo operacijų kiekio metrikos duomenys

Duomenys	Disko skaitymo operacijų kiekis
Aprašymas	Metrika aprašanti skaitymo/rašymo (*I/O*) įrenginių skaitymo operacijų kiekį. Metrikos duomenys yra kaupiamojo tipo.
Pavyzdinės reikšmės	2023-12-04 00:00:13 (GMT+0) matavimo duomenimis, įrenginys “sda” iki matavimo momento atliko 156419 skaitymo operacijų, įrenginys “sdb” atliko 244 operacijas, įrenginys “dm-0” atliko 34 operacijas
Metrikos tipas	Skaitymo/rašymo (*I/O*) operacijos
Duomenų matavimo periodas	2023-12-04 00:00:13 - 2024-01-03 13:42:43 (GMT+0)
Duomenų matavimo dažnis	Kas 15 sekundžių
Duomenų kiekis	528273 įrašai
Stulpelių kiekis	11

Trečias duomenų rinkinys skirtas aprašyti serverio skaitymo/rašymo įrenginių įvykdytą rašymo operacijų kiekį (15 lentelė).

15 lentelė. Disko rašymo operacijų kiekio metrikos duomenys

Duomenys	Disko rašymo operacijų kiekis
Aprašymas	Metrika aprašanti skaitymo/rašymo (*I/O*) įrenginių rašymo operacijų kiekį. Metrikos duomenys yra kaupiamojo tipo.
Pavyzdinės reikšmės	2023-12-04 00:00:13 (GMT+0) matavimo duomenimis, įrenginys “sda” iki matavimo momento atliko 3780307 rašymo operacijas, įrenginys “sdb” atliko 8 operacijas, įrenginys “dm-0” atliko 0 operacijų
Metrikos tipas	Skaitymo/rašymo (*I/O*) operacijos
Duomenų matavimo periodas	2023-12-04 00:00:13 - 2024-01-03 13:42:28 (GMT+0)
Duomenų matavimo dažnis	Kas 15 sekundžių
Duomenų kiekis	528270 įrašai
Stulpelių kiekis	11

Ketvirtas duomenų rinkinys skirtas aprašyti serverio laisvos operatyvios atminties (*RAM*) kiekį matavimo momentu (16 lentelė).

16 lentelė. Laisvos operatyvios atminties kiekio metrikos duomenys

Duomenys	Laisvos operatyvios atminties kiekis
Aprašymas	Metrika aprašanti laisvą operatyviosios atminties (*RAM*) kiekį baitais matavimo momentu.
Pavyzdinės reikšmės	2023-12-04 00:00:13 (GMT+0) matavimo duomenimis, laisvas operatyvios atminties (*RAM*) kiekis buvo 28420968448 baitai.
Metrikos tipas	Operatyviosios atminties (*RAM*) apkrautumas
Duomenų matavimo periodas	2023-12-04 00:00:13 - 2024-01-03 13:54:43 (GMT+0)
Duomenų matavimo dažnis	Kas 15 sekundžių
Duomenų kiekis	176139 įrašai
Stulpelių kiekis	10

Penktas duomenų rinkinys skirtas aprašyti serverio visą operatyvios atminties (*RAM*) kiekį matavimo momentu (17 lentelė).

17 lentelė. Visos operatyvios atminties kiekio metrikos duomenys

Duomenys	Visas operatyvios atminties kiekis
Aprašymas	Metrika aprašanti visą operatyviosios atminties (*RAM*) kiekį baitais matavimo momentu.
Pavyzdinės reikšmės	2023-12-04 00:00:13 (GMT+0) matavimo duomenimis, visas operatyvios atminties (*RAM*) kiekis buvo 33736380416 baitai.
Metrikos tipas	Operatyvioji atmintis (*RAM*)

Duomenų matavimo periodas	2023-12-04 00:00:13 - 2024-01-03 13:54:43 (GMT+0)
Duomenų matavimo dažnis	Kas 15 sekundžių
Duomenų kiekis	176139 įrašai
Stulpelių kiekis	10

Šeštas duomenų rinkinys skirtas aprašyti įmonės matuojamą vidaus metriką, apibūdinančią įmonės produkto funkcijos (naujų versijų sudarymo) generavimo spartą (18 lentelė).

18 lentelė. Naujos versijos generavimo trukmės metrikos duomenys

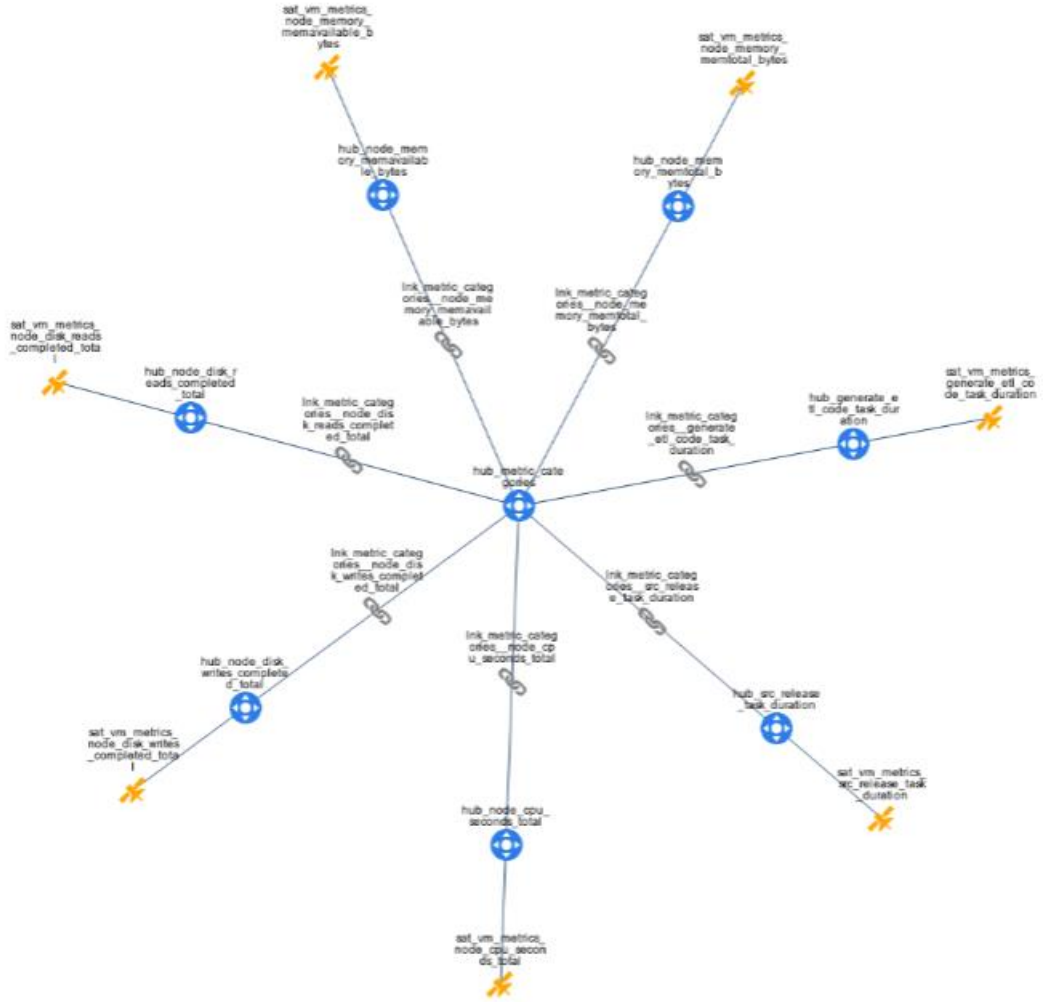
Duomenys	Naujos versijos (angl. r*elease*) generavimo trukmė
Aprašymas	Metrika aprašanti kiek laiko užtruko sugeneruoti naują versiją po atliktų naudotojo pakeitimų.
Pavyzdinės reikšmės	2023-11-09 10:10:05 matavimo duomenimis, naujos versijos sugeneravimo užduotis užtruko 0,0445 sekundės duomenų nuskaitymui iš duomenų bazės ir 4 sekundes naujos versijos sugeneravimui
Metrikos tipas	Veiklos metrika
Duomenų matavimo periodas	2023-11-09 10:10:05 - 2023-11-30 21:53:07 (GMT+0)
Duomenų matavimo dažnis	Kiekvieną kartą kai įvykdoma naujos versijos generavimo užduotis
Duomenų kiekis	49 įrašai
Stulpelių kiekis	4

Septintas duomenų rinkinys skirtas aprašyti įmonės matuojamą vidaus metriką, apibūdinančią įmonės produkto funkcijos (*ETL* kodo) generavimo spartą (18 lentelė).

19 lentelė. *ETL* kodo generavimo trukmės metrikos duomenys

Duomenys	*ETL* kodo generavimo trukmė
Aprašymas	Metrika aprašanti kiek laiko užtruko sugeneruoti *ETL* kodą
Pavyzdinės reikšmės	2021-07-07 11:11:27 matavimo duomenimis, *ETL* kodo generavimo užduotis užtruko 4,73 sekundes informacijos nuskaitymui iš duomenų bazės ir 9 sekundes generuojant *ETL* kodą.
Metrikos tipas	Veiklos metrika
Duomenų matavimo periodas	2021-07-07 11:11:27 - 2023-12-01 10:33:05 (GMT+0)
Duomenų matavimo dažnis	Kiekvieną kartą kai įvykdoma *ETL* kodo generavimo užduotis
Duomenų kiekis	3552 įrašai
Stulpelių kiekis	4

Visi šie duomenų rinkiniai iš pradžių buvo įkelti, suformuoti ir parengti metrikų saugyklos kūrimui *VaultSpeed* duomenų saugyklos sudarymo automatizacijos įrankyje. Tada, pasitelkiant 4 skyriuje aprašytą metrikų saugyklos sudarymo įrankį, buvo sudaryta metrikų saugykla. Sudarytas metrikų saugyklos modelis matomas 5.1 pav.



5.1 pav. Sudarytos metrikų saugyklos modelis

Sudarius metrikų saugyklą, buvo sugeneruotas *DDL* ir *ETL* kodas, kuris kartu su *VaultSpeed* įrankyje sugeneruotu kodu (skirtu duomenų paruošimo srities ir *Data Vault* sluoksnio sudarymui) buvo įvykdytas eksperimentinėje duomenų saugykloje.

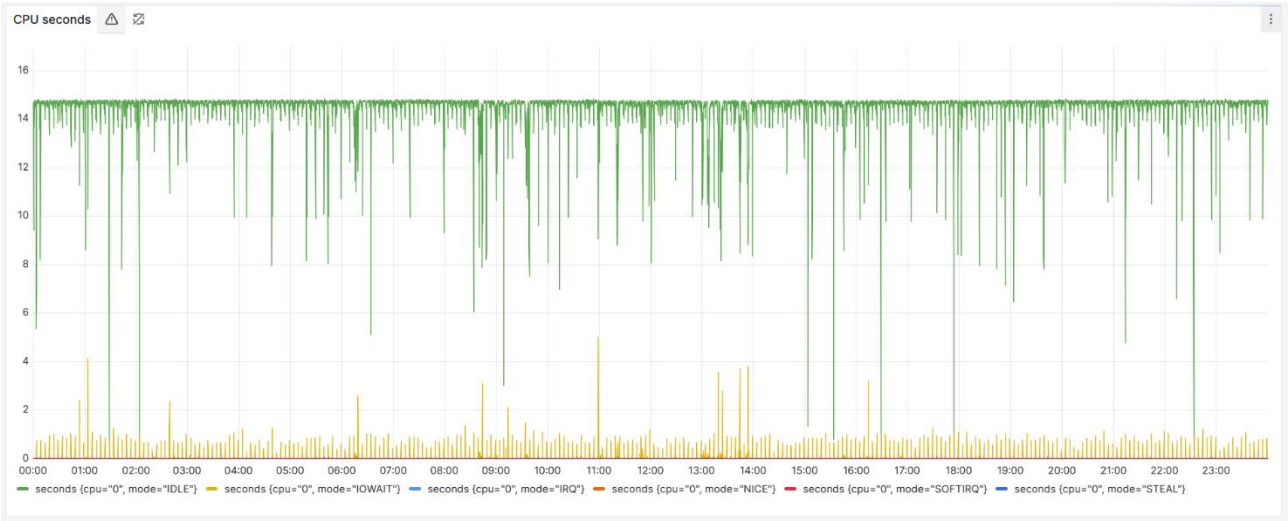
Įvykdžius visas *DDL* ir *ETL* funkcijas, buvo sudaryti metrikų duomenų pjūviai, iš kurių vėliau buvo nuskaitomi duomenys ir parengtos metrikų vizualizacijos. Visi gauti rezultatai aprašomi žemiau.

Pirmas duomenų rinkinys – procesoriaus darbo laiko skirtinguose darbo režimuose. Šio rinkinio metrikų saugyklos sudarymo rezultatai pateikiami 20 lentelėje ir sudaryta vizualizacija 5.2 paveiksle.

20 lentelė. Procesoriaus darbo laiko skirtinguose režimuose metrikos sudarymo rezultatai

Rezultatai	Procesoriaus darbo laikas skirtinguose režimuose
Sukurtų lentelių kiekis	1 centras (angl. *hub*), 1 palydovas (angl. *satellite*), 1 sąsaja (angl. *link*)
Įterptų duomenų kiekis	Centras – 11269890 įrašų Palydovas – 11269890 įrašų Sąsaja - 11269890 įrašų
Priskirta metrikų kategorija	*CPU*

Sugeneruotas *DDL* kodo kiekis	1 *DDL* failas su 3 lentelių sukūrimo (*CREATE*) komandomis
Sugeneruotas *ETL* kodo kiekis	3 *ETL* failai po 1 funkciją skirtą įkelti duomenis į centrą, palydovą ir sąsają
Sukurti pjūviai	1 pjūvis
Sudarytų išskaičiuojamųjų stulpelių kiekis	0 stulpelių



5.2 pav. Procesoriaus darbo laiko skirtinguose režimuose metrikos vizualizacija

Antras duomenų rinkinys – disko skaitymo operacijų kiekis. Šio rinkinio metrikų saugyklos sudarymo rezultatai pateikiami 21 lentelėje ir sudaryta vizualizacija 5.3 paveiksle.

21 lentelė. Disko skaitymo operacijų kiekio metrikos sudarymo rezultatai

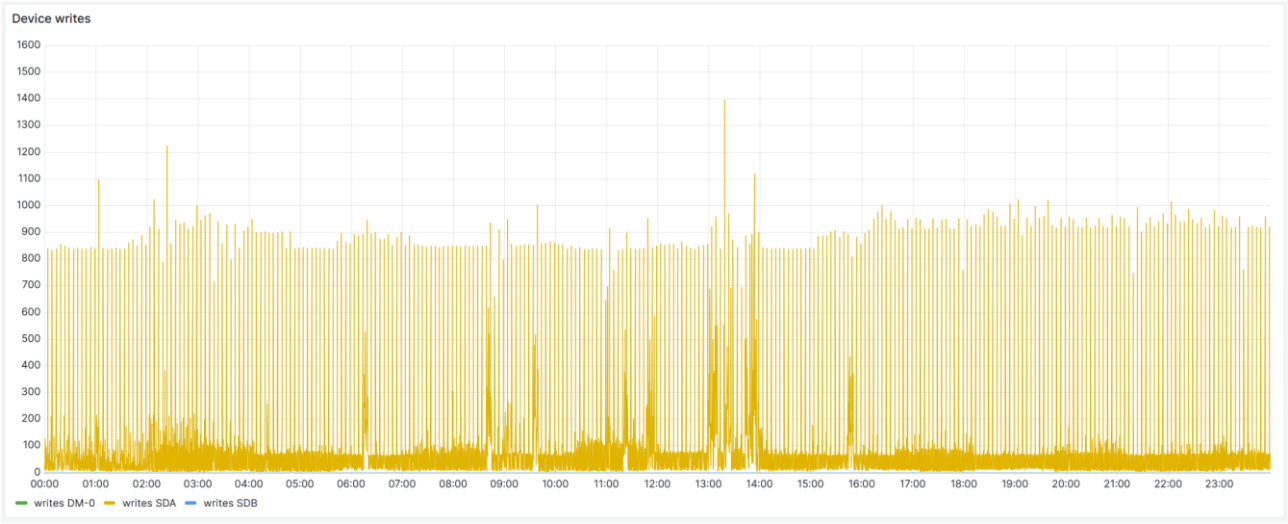
Rezultatai	Disko skaitymo operacijų kiekis
Sukurtų lentelių kiekis	1 centras (angl. *hub*), 1 palydovas (angl. *satellite*), 1 sąsaja (angl. *link*)
Įterptų duomenų kiekis	Centras – 528275 įrašai Palydovas – 528275 įrašai Sąsaja - 528275 įrašai
Priskirta metrikų kategorija	*I/O*
Sugeneruotas *DDL* kodo kiekis	1 *DDL* failas su 3 lentelių sukūrimo (*CREATE*) komandomis
Sugeneruotas *ETL* kodo kiekis	3 *ETL* failai po 1 funkciją skirtą įkelti duomenis į centrą, palydovą ir sąsają
Sukurti pjūviai	1 pjūvis
Sudarytų išskaičiuojamųjų stulpelių kiekis	0 stulpelių

5.3 pav. Disko skaitymo operacijų kiekio metrikos vizualizacija

Trečias duomenų rinkinys – disko rašymo operacijų kiekis. Šio rinkinio metrikų saugyklos sudarymo rezultatai pateikiami 22 lentelėje ir sudaryta vizualizacija 5.4 paveiksle.

22 lentelė. Disko rašymo operacijų kiekio metrikos sudarymo rezultatai

Rezultatai	Disko rašymo operacijų kiekis
Sukurtų lentelių kiekis	1 centras (angl. *hub*), 1 palydovas (angl. *satellite*), 1 sąsaja (angl. *link*)
Įterptų duomenų kiekis	Centras – 528272 įrašai Palydovas – 528272 įrašai Sąsaja - 528272 įrašai
Priskirta metrikų kategorija	*I/O*
Sugeneruotas *DDL* kodo kiekis	1 *DDL* failas su 3 lentelių sukūrimo (*CREATE*) komandomis
Sugeneruotas *ETL* kodo kiekis	3 *ETL* failai po 1 funkciją skirtą įkelti duomenis į centrą, palydovą ir sąsają
Sukurti pjūviai	1 pjūvis
Sudarytų išskaičiuojamųjų stulpelių kiekis	0 stulpelių



5.4 pav. Disko rašymo operacijų kiekio metrikos vizualizacija

Ketvirtas duomenų rinkinys – laisvos operatyvios atminties kiekis. Šio rinkinio metrikų saugyklos sudarymo rezultatai pateikiami 23 lentelėje ir sudaryta vizualizacija 5.5 paveiksle.

23 lentelė. Laisvos operatyvios atminties kiekio metrikos sudarymo rezultatai

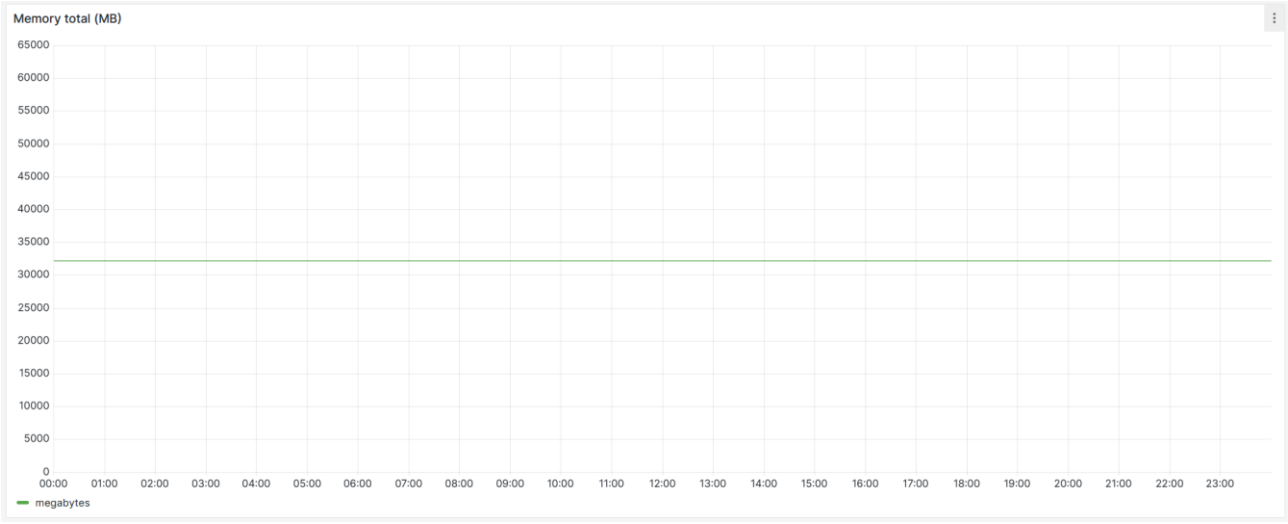
Rezultatai	Laisvos operatyvios atminties kiekis
Sukurtų lentelių kiekis	1 centras (angl. *hub*), 1 palydovas (angl. *satellite*), 1 sąsaja (angl. *link*)
Įterptų duomenų kiekis	Centras – 176141 įrašas Palydovas – 176141 įrašas Sąsaja - 176141 įrašas
Priskirta metrikų kategorija	*RAM*
Sugeneruotas *DDL* kodo kiekis	1 *DDL* failas su 3 lentelių sukūrimo (*CREATE*) komandomis
Sugeneruotas *ETL* kodo kiekis	3 *ETL* failai po 1 funkciją skirtą įkelti duomenis į centrą, palydovą ir sąsają
Sukurti pjūviai	1 pjūvis
Sudarytų išskaičiuojamųjų stulpelių kiekis	1 stulpelis

5.5 pav. Laisvos operatyvios atminties kiekio metrikos vizualizacija

Penktas duomenų rinkinys – laisvos operatyvios atminties kiekis. Šio rinkinio metrikų saugyklos sudarymo rezultatai pateikiami 24 lentelėje ir sudaryta vizualizacija 5.6 paveiksle.

24 lentelė. Viso operatyvios atminties kiekio metrikos sudarymo rezultatai

Rezultatai	Visas operatyvios atminties kiekis
Sukurtų lentelių kiekis	1 centras (angl. *hub*), 1 palydovas (angl. *satellite*), 1 sąsaja (angl. *link*)
Įterptų duomenų kiekis	Centras – 176141 įrašas Palydovas – 176141 įrašas Sąsaja - 176141 įrašas
Priskirta metrikų kategorija	*RAM*
Sugeneruotas *DDL* kodo kiekis	1 *DDL* failas su 3 lentelių sukūrimo (*CREATE*) komandomis
Sugeneruotas *ETL* kodo kiekis	3 *ETL* failai po 1 funkciją skirtą įkelti duomenis į centrą, palydovą ir sąsają
Sukurti pjūviai	1 pjūvis
Sudarytų išskaičiuojamųjų stulpelių kiekis	1 stulpelis

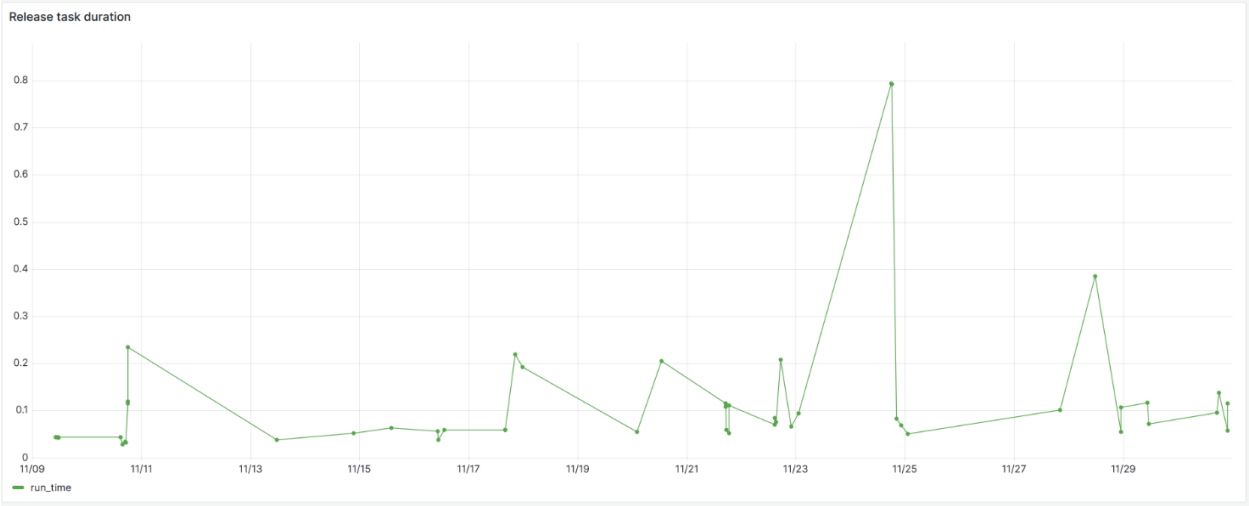


5.6 pav. Viso operatyviosios atminties kiekio metrikos vizualizacija

Šeštas duomenų rinkinys – laisvos operatyvios atminties kiekis. Šio rinkinio metrikų saugyklos sudarymo rezultatai pateikiami 25 lentelėje ir sudaryta vizualizacija 5.7 paveiksle.

25 lentelė. Naujos versijos generavimo trukmės metrikos sudarymo rezultatai

Rezultatai	Naujos versijos (angl. r*elease*) generavimo trukmė
Sukurtų lentelių kiekis	1 centras (angl. *hub*), 1 palydovas (angl. *satellite*), 1 sąsaja (angl. *link*)
Įterptų duomenų kiekis	Centras – 51 įrašas Palydovas – 51 įrašas Sąsaja - 51 įrašas
Priskirta metrikų kategorija	Ekstra
Sugeneruotas *DDL* kodo kiekis	1 *DDL* failas su 3 lentelių sukūrimo (*CREATE*) komandomis
Sugeneruotas *ETL* kodo kiekis	3 *ETL* failai po 1 funkciją skirtą įkelti duomenis į centrą, palydovą ir sąsają
Sukurti pjūviai	1 pjūvis
Sudarytų išskaičiuojamųjų stulpelių kiekis	2 stulpeliai

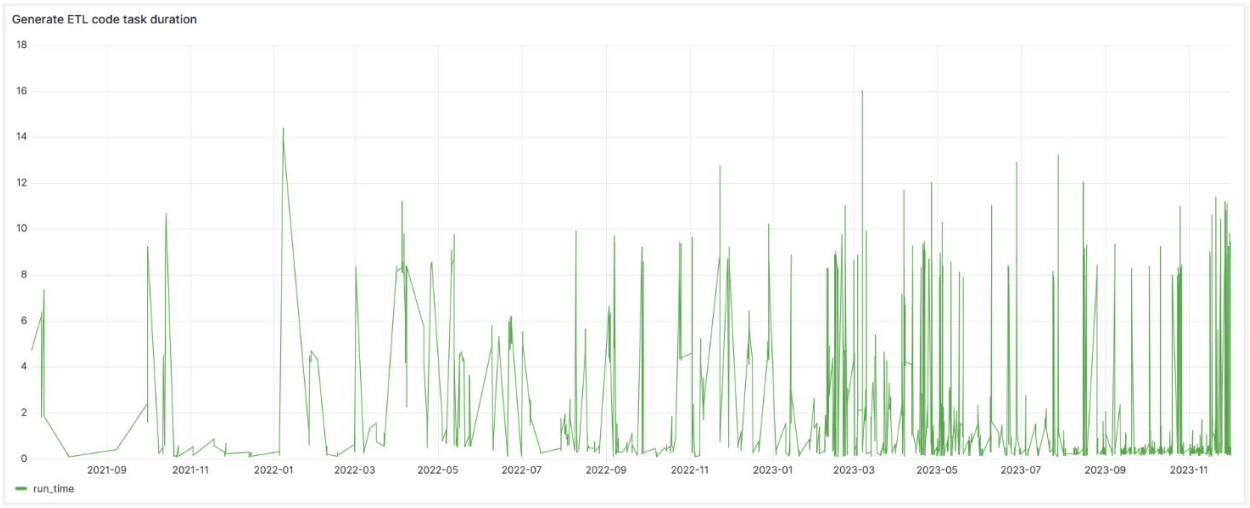


5.7 pav. Naujos versijos generavimo trukmės metrikos vizualizacija

Septintas duomenų rinkinys – laisvos operatyvios atminties kiekis. Šio rinkinio metrikų saugyklos sudarymo rezultatai pateikiami 26 lentelėje ir sudaryta vizualizacija 5.8 paveiksle.

26 lentelė. *ETL* kodo generavimo trukmės metrikos sudarymo rezultatai

Rezultatai	*ETL* kodo generavimo trukmė
Sukurtų lentelių kiekis	1 centras (angl. *hub*), 1 palydovas (angl. *satellite*), 1 sąsaja (angl. *link*)
Įterptų duomenų kiekis	Centras – 3547 įrašaį Palydovas – 3547 įrašai Sąsaja - 3547 įrašai
Priskirta metrikų kategorija	Ekstra
Sugeneruotas *DDL* kodo kiekis	1 *DDL* failas su 3 lentelių sukūrimo (*CREATE*) komandomis
Sugeneruotas *ETL* kodo kiekis	3 *ETL* failai po 1 funkciją skirtą įkelti duomenis į centrą, palydovą ir sąsają
Sukurti pjūviai	1 pjūvis
Sudarytų išskaičiuojamųjų stulpelių kiekis	2 stulpeliai



5.8 pav. *ETL* kodo generavimo trukmės metrikos vizualizacija

Taigi, pagal gautus atlikto eksperimento rezultatus galima teigti, kad metrikų saugykla buvo sukurta sėkmingai pateiktiems realios įmonės kompiuterinių resursų metrikų duomenims saugoti.

Visi pradiniai metrikų duomenys (apie 13 mln. įrašų) buvo išsaugoti atitinkamos *Data Vault* metrikų saugyklos struktūrose, kaip tai galima matyti iš įrašų skaičių. Visi įrašai buvo padalinti tarp 3 pagrindinių struktūrų:

- 8 centrų, kuriose buvo išsaugoti identifikaciniai (verslo raktų) stulpeliai;
- 7 palydovų, kuriose buvo išsaugoti aprašomieji stulpeliai;
- 7 sąsajų, kurios sujungia centro reikšmes su atitinkama metrikų kategorija.

Visas įterptų duomenų kiekis (apie 13 mln. įrašų) sutampa su pradiniuose duomenyse esančių duomenų kiekiu, tik yra 2 elementais didesnis kiekvienoje lentelėje, bet tai 2 papildomos numatytosios *Data Vault* architektūros reikšmės, skirtos identifikuoti tuščias (*NULL*) reikšmes ir nerastas reikšmes (angl. *unknown*), reikalingas formuojant maišos funkcijas, kai centras susietas su tuščiomis arba nerastomis reikšmėmis.

Visoms metrikoms buvo sėkmingai priskirtos metrikų kategorijos, kurios padengė šias taksonomijos kategorijas: *CPU*, *RAM*, *I/O* ir *Kita*.

Įrankio sugeneruotas *DDL* ir *ETL* kodas buvo sėkmingai įvykdytas ir sėkmingai sudarė metrikų saugyklos struktūrą bei įkėlė visus pradinius metrikos duomenis bei praplėtė jas 6 skaičiuojamųjų stulpelių pagalba.

Metrikų saugyklai buvo sėkmingai sudaryti 7 pjūviai ir sukurtos 7 vizualizacijos, atvaizduojančios metrikų duomenis.

5.2. *Data Vault* metrikų saugyklos sudarymo metodikos apklausa

5.2.1. Apklausos planas

Patikrinti metrikų saugyklos metodikos suprantamumą ir pagalbą kuriant metrikų saugyklą buvo atlikta apklausa. Apklausa skirta įvertinti metodikos suprantamumą ir pagalbą kuriant metrikų saugyklas. Todėl, norint įvertinti šį aspektą, buvo pasirinkta apklausti specialistus, turinčius patirties

su *Data Vault* duomenų saugyklomis ir metrikų rinkimų bei analize. Dalis apklaustųjų buvo iš *VaultSpeed* įmonės, kuriai buvo pateikta demonstracinė sistemos versija (2 priedas).

Apklausą nuspręsta sudaryti iš dviejų dalių: metodikos pristatymo ir klausimyno.

Metodikos pristatymo dalis skirta supažindinti tyrimo dalyvius su metrikų saugyklos sudarymo metodika. Tam pasiekti buvo parengtas 12 minučių pristatomasis vaizdo įrašas. Įraše pristatomas metodikos tikslas, metrikų saugyklos vieta dabartinėje *Data Vault* duomenų saugyklos architektūroje ir pateikiama pavyzdinė metodikos pritaikymo situacija naudojant sukurtą metrikų saugyklos sudarymo įrankį. Vaizdo įraše pristatomi visi žingsniai nuo pradinio projekto parengimo ir metrikų šaltinio pridėjimo *VaultSpeed* įrankyje iki vizualizacijų sukūrimo *Grafana* įrankyje.

Metodikos pristatymą peržiūrėjęs dalyvis yra nukreipiamas į klausimyną. Klausimyną sudaro iš viso 8 klausimai – 7 privalomi uždaro tipo ir 1 neprivalomas atviro tipo. Klausimynas buvo parengtas anglų kalba ir pateikiamas prieduose (3 priedas).

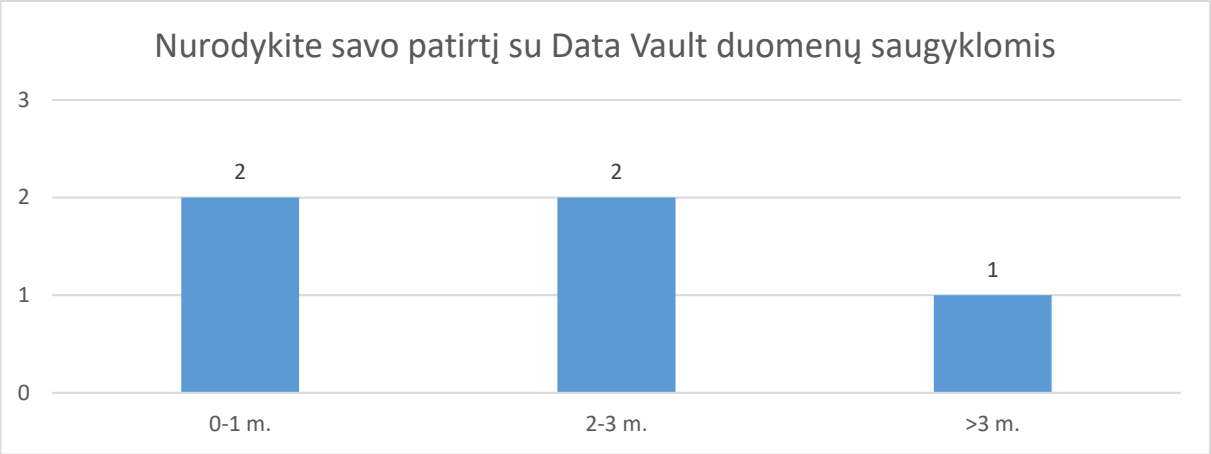
Pirmi du klausimai skirti surinkti informaciją apie respondentų patirtį.
1. Klausimas: *Nurodykite savo patirtį su Data Vault duomenų saugyklomis*.
2. Klausimas: *Nurodykite savo patirtį su metrikų rinkimu ir analize.*

Sekantys penki klausimai skirti surinkti respondentų nuomonę apie metodikos suprantamumą, pritaikomumą ir pagalbą kuriant metrikų saugyklas. Šie klausimai pateikiami kaip teiginiai, juos įvertinant penkiabalėje Likerto skalėje [53].
3. Klausimas: *Pateikta metrikų saugyklos metodika yra aiški ir suprantama*.
4. Klausimas: *Pateikta metrikų saugyklos metodika ir realizuotas įrankis skatina metrikų saugyklos kūrimą.*
5. Klausimas: *Pateikta metrikų taksonomija (kategorijos) padeda metrikų analizėje ir stebėjime.*
6. Klausimas: *Jūs naudotumėte pateiktą metrikų saugyklos metodiką savo darbe.*
7. Klausimas: *Jūs rekomenduotumėte pateiktą metrikų saugyklos metodiką ir realizuotą įrankį kitiems.*

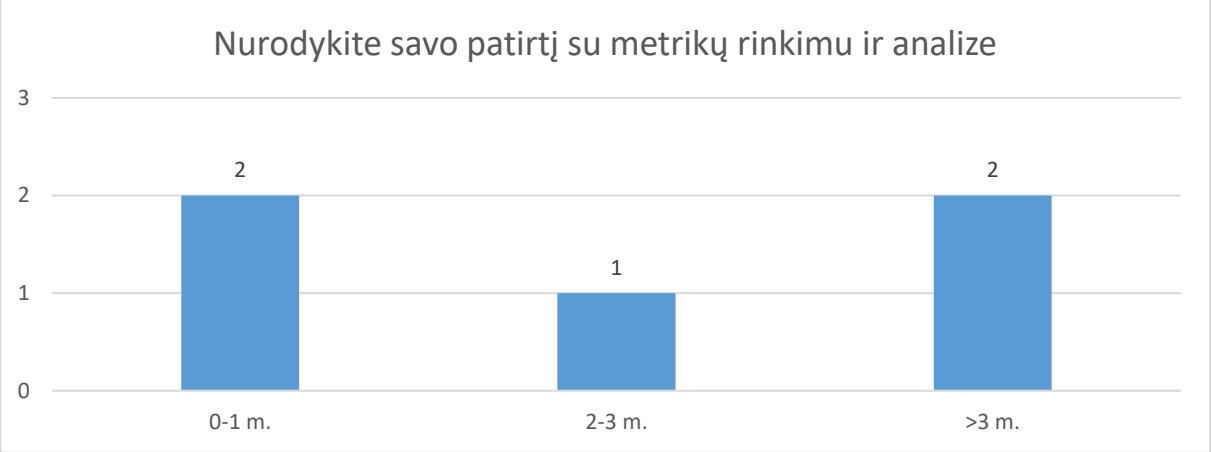
Paskutinis klausimas yra atviro tipo ir neprivalomas. Jis skirtas surinkti atsiliepimus ir pasiūlymus apie metrikų saugyklos metodiką ir realizuotą įrankį.
8. Klausimas: *(Neprivalomas) Jūsų papildomos remarkos ir pastabos apie pateiktą metrikų saugyklos metodiką ir realizuotą automatizacijos įrankį.*

5.2.2. Apklausos rezultatai

Iš viso buvo gauti 5 respondentų atsakymai. Kaip ir buvo minėta apklausos plane, apklausti buvo specialistai, turintys patirtį tiek su *Data Vault* duomenų saugyklomis, tiek su metrikų kaupimu ir analize. Tai galima matyti ir iš atsakymų į pirmus du apklausos klausimus (žr. 5.9 ir 5.10 pav.)



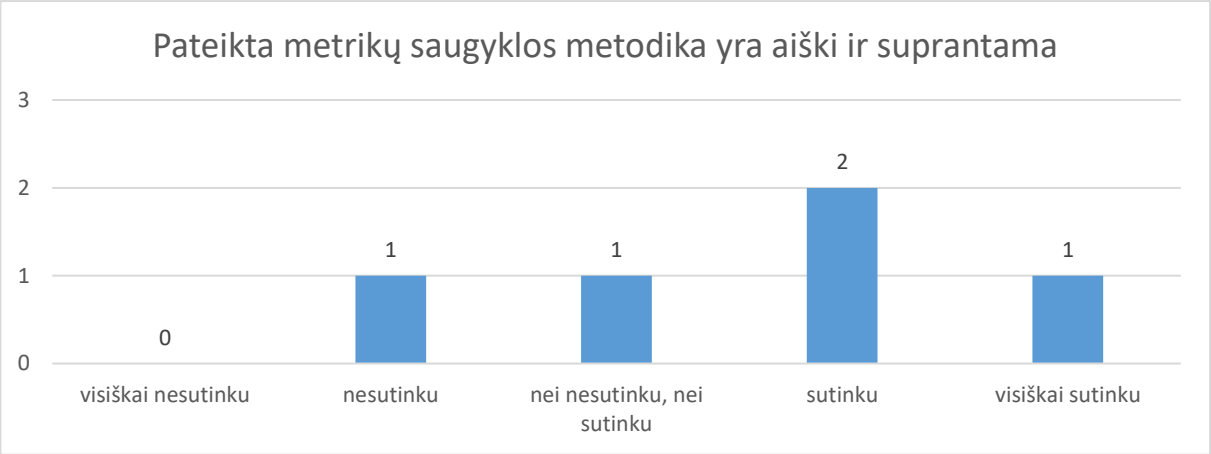
5.9 pav. Respondentų patirtis su *Data Vault* duomenų saugyklomis



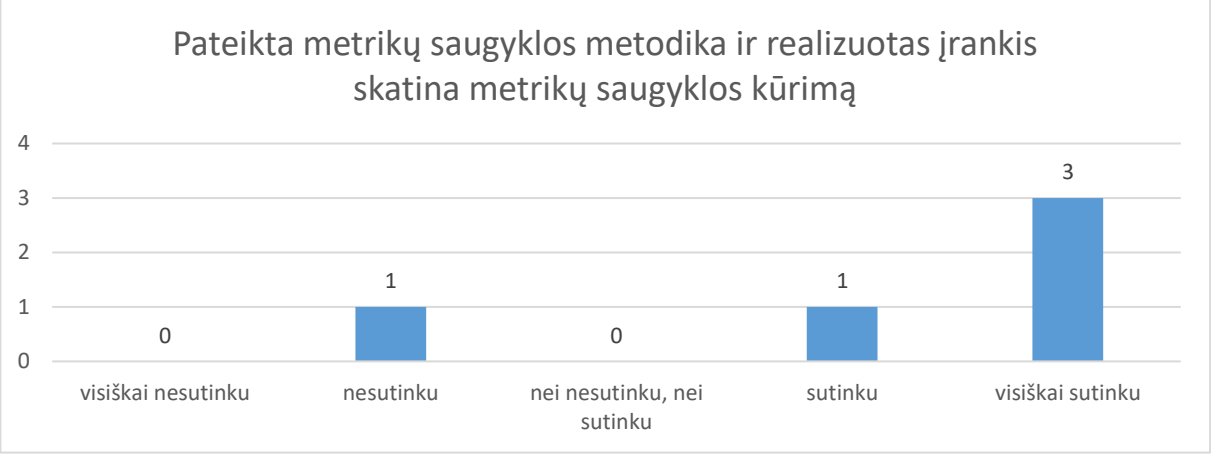
5.10 pav. Respondentų patirtis su metrikų analize ir stebėjimu

Iš pirmų dviejų klausimų atsakymų galima pastebėti, kad buvo apklausti specialistai tiek tik pradedantys darbą su *Data Vault* duomenų saugyklomis, tiek ir sukaupę daugiau nei dvejus metus patirties. Panaši situacija ir su patirtimi metrikų kaupimo ir analizės sferoje. Du respondentai atsakė, kad turi iki dviejų metų patirties, o likę trys turi sukaupę bent dviejų metų patirtį.

Sekantys du apklausos klausimai buvo skirti įvertinti, kiek aiški ir suprantama pateikta metodika bei ar realizuotas metrikų saugyklos įrankis skatina metrikų saugyklų kūrimą (žr. 5.11 ir 5.12 pav.).



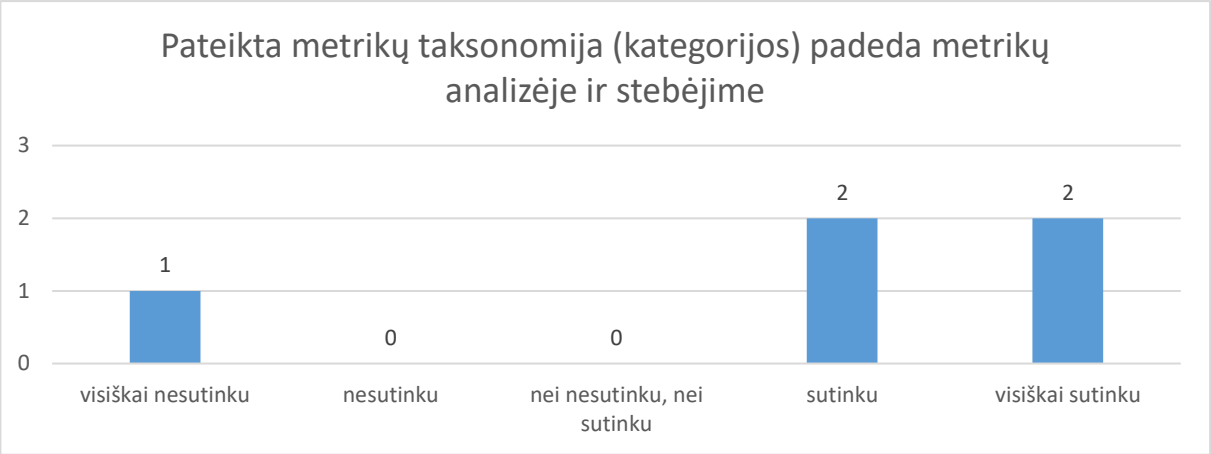
5.11 pav. Klausimo apie metodikos aiškumą atsakymų pasiskirstymas



5.12 pav. Klausimo apie metodikos ir realizuoto įrankio skatinimą kurti metrikų saugyklas atsakymų pasiskirstymas

Metrikų saugyklos metodikos aiškumo ir suprantamumo klausimu respondentų nuomonės išsiskyrė, tačiau dauguma linkus teigti, kad metodika yra aiški ir suprantama. Taip pat 4 iš 5 respondentų sutiko (3 iš 5 stipriai sutiko), kad realizuotas įrankis skatina metrikų saugyklos kūrimą. Tiktai 1 respondentas turėjo priešingą nuomonę. Tačiau, svarbu atkreipti dėmesį, kad būtent šis respondentas taip pat pažymėjo, kad turi nedaug patirties (nuo 0 iki 1 metų) metrikų rinkimo ir analizės srityje.

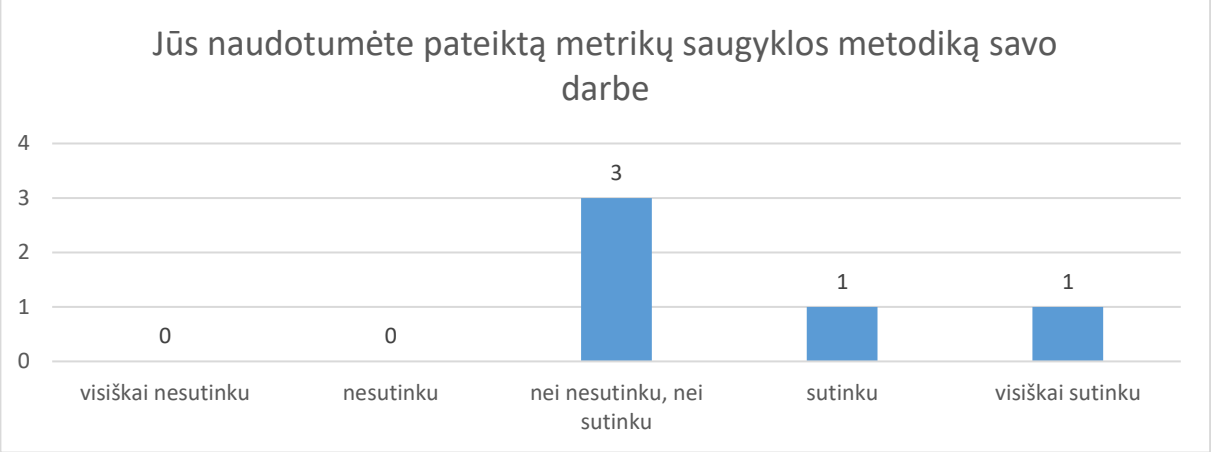
5 klausimu buvo siekiama išsiaiškinti ar pasiūlyta metrikų taksonomija yra naudinga metrikų saugyklos kūrime ir padeda metrikų analizėje (žr. 5.13 pav.).



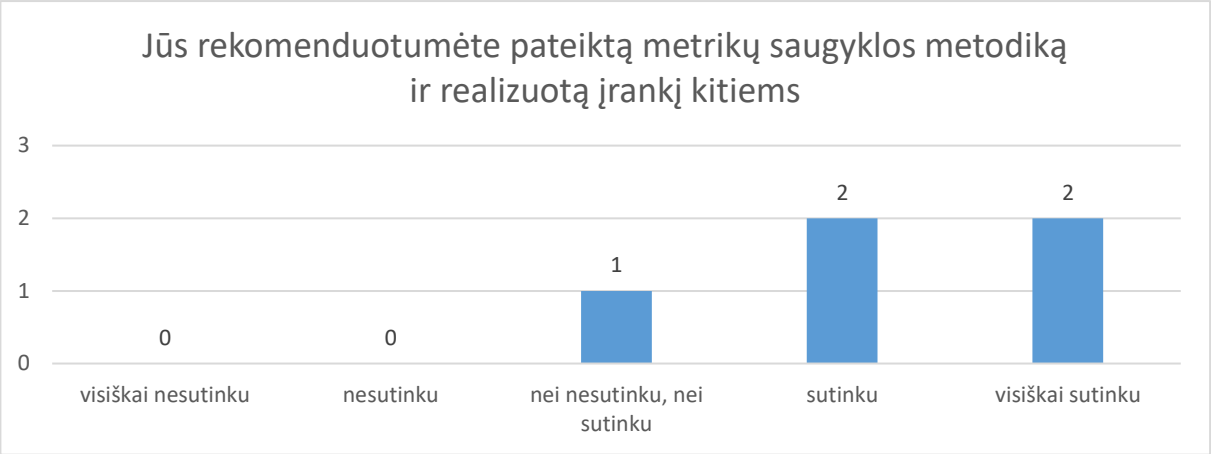
5.13 pav. Klausimo apie metrikų taksonomijos pagalbą atsakymų pasiskirstymas

Šiuo klausimu 4 iš 5 respondentų taip pat sutiko, kad metrikų taksonomija yra naudinga metrikų analizei ir stebėjimui, išskyrus 1 tą patį respondentą turintį stiprią priešingą nuomonę.

6 ir 7 klausimai buvo skirti įvertinti kiek pasiūlyta metodika ir realizuotas įrankis yra išbaigti, suprantami ir pritaikomi darbinėje sferoje. Dėl to buvo paprašyta apklausos dalyvių įvertinti kiek jie linkę naudoti pasiūlytą metrikų saugyklos metodiką savo darbe ir ar rekomenduotų realizuotą įrankį kitiems (žr. 5.14 ir 5.15 pav.).



5.14 pav. Klausimo apie metodikos pritaikomumą darbe atsakymų pasiskirstymas



5.15 pav. Klausimo apie metodikos ir įrankio rekomendaciją kitiems atsakymų pasiskirstymas

Didžioji dauguma respondentų neišreiškė stipraus įsitikimo, kad naudotų ar nenaudotų pasiūlytos metodikos savo darbe, tačiau net 4 iš 5 patvirtino, kad rekomenduotų metodiką ir realizuotą įrankį kitiems. Tai gali būti susiję su tuo, kad dalis apklaustųjų buvo iš *VaultSpeed* įmonės ir dirba prie *Data Vault* duomenų saugyklų sudarymo automatizacijos įrankio, o ne patys naudojasi duomenų saugykla. Dėl to jie galimai labiau linkę rekomenduoti metrikų saugyklą savo klientams, o ne naudoti ją savo darbe.

Taigi, iš apklausos rezultatų galima teigti, kad pateikta metodika ir realizuotas įrankis galėtų prisidėti prie *Data Vault* metrikų saugyklos sudarymo. Realizuotas įrankis gali palengvinti ir paskatinti *Data Vault* duomenų saugyklos naudotojus kurti metrikų saugyklas, o pasiūlyta metrikų taksonomija gali būti naudinga metrikų analizei ir stebėjimui. Ateityje metodiką galima būtų papildyti duomenų pjūvių sudarymo gairėmis, dažniausių standartizuotų metrikų formulių ir laukų pridėjimu, tokiu būdu dar labiau palengvinant metodikos pritaikomumą.

Išvados

1. Atlikta *Data Vault 2.0* architektūros analizė parodė, kad *Data Vault 2.0* duomenų saugykla yra grindžiama trijų sričių architektūra, siekiant efektyviai išnaudoti ir apdoroti gaunamus duomenis, naudojant specifinį *Data Vault* duomenų modelį. Šis duomenų modelis daugiausiai orientuotas į veiklos procesus, dėl to yra lengvai plečiamas prisitaikant prie duomenų šaltinių pokyčių.
2. Atlikta metrikų rinkimo įrankių analizė parodė, kad egzistuoja daug sprendimų skirtų stebėti serverio ir duomenų saugyklų darbą. Tačiau, visi įrankiai turi skirtingus metrikų rinkinius, kurie dažniausiai apima šias pagrindines metrikų kategorijas: procesoriaus darbo, atminties naudojimo, disko operacijų, internetinio ryšio naudojimo ir temperatūros metrikas.
3. Atlikta duomenų saugyklų sudarymo metodikų lyginamoji analizė parodė, kad daugelis metodikų specializuojasi vieno duomenų saugyklos aspekto kūrime ir nė viena metodika neaprašo metrikų saugyklos sudarymo *Data Vault 2.0* duomenų saugyklai. Nustatyta, kad, siekiant sukurti metrikų saugyklą, reikės aprašyti metrikų surinkimo procesą, duomenų įkėlimo procesą, duomenų modelį ir duomenų pjūvių sudarymo eigą.
4. Sudarant metrikų saugyklos kūrimo metodiką, buvo aprašytas visas metrikų saugyklos sudarymo procesas – nuo metrikų duomenų įkėlimo iki jų atvaizdavimo. Procese buvo identifikuoti trys pagrindiniai duomenų metamodeliai: duomenų šaltinio metamodelis, konstravimo sluoksnio metamodelis ir *Data Vault* duomenų metamodelis. Viso proceso metu metrikų duomenys tolygiai transformuojami iki *Data Vault* duomenų metamodelio, kuris išsaugojamas metrikų saugykloje.
5. Projektavimo metu taip pat buvo sudaryta metrikų taksonomija, susidedanti iš trijų pagrindinių kategorijų: energijos metrikų, internetinio ryšio metrikų ir našumo metrikų. Ši taksonomija skirta palengvinti metrikų kategorizavimą ir analizavimą skirtingais pjūviais. Tokiu būdu ši taksonomija prisideda prie pagrindinio metodikos tikslo – palengvina *Data Vault* duomenų saugyklos metrikų stebėjimą ir kontrolę.
6. Pasiūlytos metodikos veikimui ir taikymui dabartiniuose *Data Vault 2.0* duomenų saugyklos sudarymo procesuose pademonstruoti buvo realizuotas įrankis, kuris leidžia sudaryti metrikų saugyklą ir automatiškai sugeneruoti jos *DDL* struktūros sukūrimo komandas ir *ETL* duomenų įkėlimo procesus. Įrankis palengvina metrikų saugyklos sudarymą, automatizuodamas jos sudarymo procesus bei pateikdamas saugyklos sudarymo gaires.
7. Išbandant pasiūlytos metodikos ir realizuoto įrankio veikimą, buvo sukurta metrikų saugykla realios įmonės serverių metrikų duomenims saugoti. Eksperimentas parodė, kad pasiūlytą metodiką galima pritaikyti kuriant metrikų saugyklas *Data Vault* duomenų saugyklai, o realizuotas įrankis palengvina metrikų saugyklos sudarymo procesą jį automatizuodamas.
8. Taip pat buvo įvertintas metodikos aiškumas ir įrankio pritaikomumas, apklausiant *Data Vault* duomenų saugyklų srities ekspertus. Apklausos rezultatai parodė, kad, daugumos apklaustų ekspertų nuomone, pasiūlyta metodika ir realizuotas įrankis gali paskatinti naudotojus kurti *Data Vault* metrikų saugyklas metrikų stebėjimo uždaviniams spręsti, o pasiūlyta metrikų taksonomija gali pagelbėti metrikų duomenų analizėje.

Literatūros sąrašas

1. What is a data warehouse? [interaktyvus]. 23 spalio 2022. [žiūrėta 17 gegužės 2024]. Prieiga per: https://www.oracle.com/database/what-is-a-data-warehouse/
2. OLSCHIMKE, Michael. *Data lake vs data warehouse. Is there a middle ground?*. 2021.
3. LINSTEDT, Dan ir OLSCHIMKE, Michael. *Building a Scalable Data Warehouse with Data Vault 2.0*. . 1st. San Francisco, CA, USA : Morgan Kaufmann Publishers Inc., 2015. ISBN 0128025107.
4. OLSCHIMKE, Michael. *Why Data Vault is the best model for data warehouse automation*. 2022.
5. VaultSpeed. [interaktyvus]. 23 spalio 2022. [žiūrėta 17 gegužės 2024]. Prieiga per: https://vaultspeed.com/
6. WhereScape. [interaktyvus]. 23 spalio 2022. [žiūrėta 17 gegužės 2024]. Prieiga per: https://www.wherescape.com/solutions/project-types/data-vault-automation/
7. ALI, Syed Muhammad Fawad ir WREMBEL, Robert. From conceptual design to performance optimization of ETL workflows: current state of research and open problems. *The VLDB Journal*. [interaktyvus]. 2017. Vol. 26, no. 6, p. 777–801. Prieiga per: doi: 10.1007/s00778-017-0477-2.
8. EL AKKAOUI, Zineb, VAISMAN, Alejandro A ir ZIMÁNYI, Esteban. A Quality-based ETL Design Evaluation Framework. Iš : *ICEIS (1)*. 2019. p. 249–257.
9. KUMAR, Nitin ir KUMAR, P Sreenivasa. An Efficient Heuristic for Logical Optimization of ETL Workflows. Iš : CASTELLANOS, Malu, DAYAL, Umeshwar ir MARKL, Volker (sud.), *Enabling Real-Time Business Intelligence*. Berlin, Heidelberg : Springer Berlin Heidelberg, 2011. p. 68–83. ISBN 978-3-642-22970-1.
10. SIMITSIS, A, VASSILIADIS, P ir SELLIS, T. State-space optimization of ETL workflows. *IEEE Transactions on Knowledge and Data Engineering*. 2005. Vol. 17, no. 10, p. 1404–1419. Prieiga per: doi: 10.1109/TKDE.2005.169.
11. TZIOVARA, Vasiliki, VASSILIADIS, Panos ir SIMITSIS, Alkis. Deciding the Physical Implementation of ETL Workflows. Iš : *Proceedings of the ACM Tenth International Workshop on Data Warehousing and OLAP*. [interaktyvus]. New York, NY, USA : Association for Computing Machinery, 2007. p. 49–56. DOLAP '07. ISBN 9781595938275. Prieiga per: doi: 10.1145/1317331.1317341.
12. HAHN, Sarah Myriam Lydia. Analysis of Existing Concepts of Optimization of ETL-Processes. Iš : SILHAVY, Radek (sud.), *Software Engineering Methods in Intelligent Algorithms*. Cham : Springer International Publishing, 2019. p. 62–76. ISBN 978-3-030-19807-7.
13. ALI, Syed Muhammad Fawad. Next-generation ETL Framework to Address the Challenges Posed by Big Data. Iš : *DOLAP*. 2018.
14. ALI, Syed Muhammad Fawad ir WREMBEL, Robert. Framework to Optimize Data Processing Pipelines Using Performance Metrics. Iš : *International Conference on Big Data Analytics and Knowledge Discovery*. 2020. p. 131–140.
15. Prometheus Node Exporter. [interaktyvus]. [žiūrėta 18 gruodžio 2022]. Prieiga per: https://github.com/prometheus/node_exporter
16. Netdata. [interaktyvus]. [žiūrėta 18 gruodžio 2022]. Prieiga per: https://learn.netdata.cloud/
17. Telegraf. [interaktyvus]. [žiūrėta 18 gruodžio 2022]. Prieiga per: https://www.influxdata.com/time-series-platform/telegraf/
18. Node Exporter for Prometheus Dashboard. [interaktyvus]. [žiūrėta 18 gruodžio 2022]. Prieiga per: https://grafana.com/grafana/dashboards/11074-node-exporter-for-prometheus-dashboard-en-v20201010/

19. FIANDRINO, C, KLIAZOVICH, D, BOUVRY, P ir ZOMAYA, A Y. Performance and Energy Efficiency Metrics for Communication Systems of Cloud Computing Data Centers. *IEEE Transactions on Cloud Computing*. 2017. Vol. 5, no. 4, p. 738–750. Prieiga per: doi: 10.1109/TCC.2015.2424892.
20. ASLANPOUR, Mohammad S, GILL, Sukhpal Singh ir TOOSI, Adel N. Performance evaluation metrics for cloud, fog and edge computing: A review, taxonomy, benchmarks and standards for future research. *Internet of Things*. [interaktyvus]. 2020. Vol. 12, p. 100273. Prieiga per: doi: https://doi.org/10.1016/j.iot.2020.100273.
21. ANWAR, Ali, SAILER, Anca, KOCHUT, Andrzej ir BUTT, Ali R. Anatomy of Cloud Monitoring and Metering: A Case Study and Open Problems. Iš : *Proceedings of the 6th Asia-Pacific Workshop on Systems*. [interaktyvus]. New York, NY, USA : Association for Computing Machinery, 2015. APSys '15. ISBN 9781450335546. Prieiga per: doi: 10.1145/2797022.2797039.
22. LORIDO-BOTRAN, Tania, MIGUEL-ALONSO, Jose ir LOZANO, Jose A. A Review of Auto-scaling Techniques for Elastic Applications in Cloud Environments. *Journal of Grid Computing*. [interaktyvus]. 2014. Vol. 12, no. 4, p. 559–592. Prieiga per: doi: 10.1007/s10723-014-9314-7.
23. MADNI, Syed Hamid Hussain, LATIFF, Muhammad Shafie Abd, COULIBALY, Yahaya ir ABDULHAMID, Shafi'i Muhammad. Recent advancements in resource allocation techniques for cloud computing environment: a systematic review. *Cluster Computing*. [interaktyvus]. 2017. Vol. 20, no. 3, p. 2489–2533. Prieiga per: doi: 10.1007/s10586-016-0684-4.
24. CURINO, Carlo, JONES, Evan P C, MADDEN, Samuel ir BALAKRISHNAN, Hari. Workload-Aware Database Monitoring and Consolidation. Iš : *Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data*. [interaktyvus]. New York, NY, USA : Association for Computing Machinery, 2011. p. 313–324. SIGMOD '11. ISBN 9781450306614. Prieiga per: doi: 10.1145/1989323.1989357.
25. MEERA, A ir SWAMYNATHAN, S. Agent based Resource Monitoring System in IaaS Cloud Environment. *Procedia Technology*. [interaktyvus]. 2013. Vol. 10, p. 200–207. Prieiga per: doi: https://doi.org/10.1016/j.protcy.2013.12.353.
26. Difference between Kimball and Inmon. [interaktyvus]. [žiūrėta 15 sausio 2023]. Prieiga per: https://www.geeksforgeeks.org/difference-between-kimball-and-inmon/
27. GIEBLER, Corinna, GRÖGER, Christoph, HOOS, Eva, SCHWARZ, Holger ir MITSCHANG, Bernhard. Modeling Data Lakes with Data Vault: Practical Experiences, Assessment, and Lessons Learned. Iš : LAENDER, Alberto H F, PERNICI, Barbara, LIM, Ee-Peng ir DE OLIVEIRA, José Palazzo M (sud.), *Conceptual Modeling*. Cham : Springer International Publishing, 2019. p. 63–77. ISBN 978-3-030-33223-5.
28. YESSAD, Lamia ir LABIOD, Aissa. *Comparative study of data warehouses modeling approaches: Inmon, Kimball and Data Vault*. . 2016.
29. ČERNJEKA, K, JAKŠIĆ, D ir JOVANOVIC, V. NoSQL document store translation to data vault based EDW. Iš : *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*. 2018. p. 1197–1202. Prieiga per: doi: 10.23919/MIPRO.2018.8400217.
30. KRNETA, Dragoljub, JOVANOVIC, Vladan ir MARJANOVIC, Zoran. An approach to data mart design from a data vault. *INFOTEH-Jahorina BiH*. 2016. Vol. 15.
31. PUONTI, Mikko, RAITALAAKSO, Timo, AHO, Timo ir MIKKONEN, Tommi. Automating Transformations in Data Vault Data Warehouse Loads. *Frontiers in Artificial Intelligence and Applications*. 1 sausio 2017. Vol. 292, p. 215–230. Prieiga per: doi: 10.3233/978-1-61499-720-7-215.

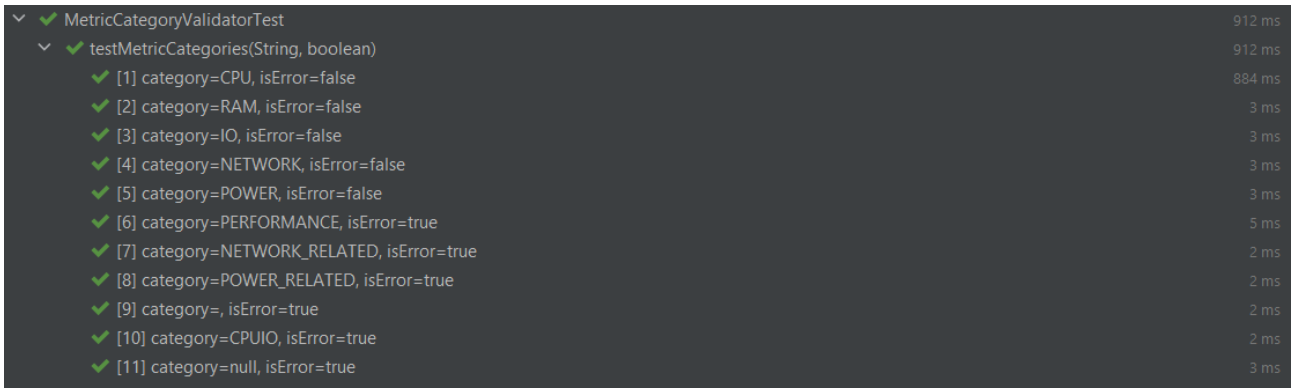
32. PostgreSQL. [interaktyvus]. 13 balandžio 2024. [žiūrėta 13 balandžio 2024]. Prieiga per: https://www.postgresql.org/
33. Snowflake. [interaktyvus]. 13 balandžio 2024. [žiūrėta 13 balandžio 2024]. Prieiga per: https://www.snowflake.com/en/
34. BigQuery. [interaktyvus]. 13 balandžio 2024. [žiūrėta 13 balandžio 2024]. Prieiga per: https://cloud.google.com/bigquery
35. SpringBoot. [interaktyvus]. 13 balandžio 2024. [žiūrėta 13 balandžio 2024]. Prieiga per: https://spring.io/projects/spring-boot
36. React. [interaktyvus]. 13 balandžio 2024. [žiūrėta 13 balandžio 2024]. Prieiga per: https://react.dev/
37. ASP .NET Core. [interaktyvus]. 13 balandžio 2024. [žiūrėta 13 balandžio 2024]. Prieiga per: https://dotnet.microsoft.com/en-us/apps/aspnet
38. Laravel. [interaktyvus]. 13 balandžio 2024. [žiūrėta 13 balandžio 2024]. Prieiga per: https://laravel.com/
39. Web Framework Benchmarks. [interaktyvus]. 6 balandžio 2024. [žiūrėta 6 balandžio 2024]. Prieiga per: https://www.techempower.com/benchmarks/#hw=ph&test=composite&section=data-r22
40. Spring Boot (Github). [interaktyvus]. 6 balandžio 2024. [žiūrėta 6 balandžio 2024]. Prieiga per: https://github.com/spring-projects/spring-boot
41. React (Github). [interaktyvus]. 6 balandžio 2024. [žiūrėta 6 balandžio 2024]. Prieiga per: https://github.com/facebook/react
42. ASP .NET (Github). [interaktyvus]. 6 balandžio 2024. [žiūrėta 6 balandžio 2024]. Prieiga per: https://github.com/topics/aspnet
43. Laravel (Github). [interaktyvus]. 6 balandžio 2024. [žiūrėta 6 balandžio 2024]. Prieiga per: https://github.com/laravel/laravel
44. Stack Overflow Trends. [interaktyvus]. 6 balandžio 2024. [žiūrėta 6 balandžio 2024]. Prieiga per: https://insights.stackoverflow.com/trends?tags=.net-core%2Cspring-boot%2Claravel%2Creactjs
45. MariaDB. [interaktyvus]. 13 balandžio 2024. [žiūrėta 13 balandžio 2024]. Prieiga per: https://mariadb.org/
46. Oracle. [interaktyvus]. 13 balandžio 2024. [žiūrėta 13 balandžio 2024]. Prieiga per: https://www.oracle.com/database/
47. ET. AL., Dr. J. Preetha,. An Improved Framework for Bitmap Indexes and their Use in Data Warehouse Optimization. *Turkish Journal of Computer and Mathematics Education (TURCOMAT)*. 10 balandžio 2021. Vol. 12, no. 2, p. 1513–1520. Prieiga per: doi: 10.17762/turcomat.v12i2.1389.
48. RAZA, Basit, ASLAM, Adeel, SHER, Asma, MALIK, Ahmad Kamran ir FAHEEM, Muhammad. Autonomic performance prediction framework for data warehouse queries using lazy learning approach. *Applied Soft Computing*. [interaktyvus]. 2020. Vol. 91, p. 106216. Prieiga per: doi: https://doi.org/10.1016/j.asoc.2020.106216.
49. RAZA, Basit, SHER, Asma, AFZAL, Sana, MALIK, Ahmad Kamran, ANJUM, Adeel, KUMAR, Yogan Jaya ir FAHEEM, Muhammad. Autonomic workload performance tuning in large-scale data repositories. *Knowledge and Information Systems*. [interaktyvus]. 2019. Vol. 61, no. 1, p. 27–63. Prieiga per: doi: 10.1007/s10115-018-1272-0.
50. TAYADE, Dipti. Comparative Study of ETL and E-LT in Data Warehousing. *International Research Journal of Engineering and Technology (IRJET)*. 2019. Vol. 6, no. 6, p. 2803–2807.

51. SIMITSIS, Alkis, VASSILIADIS, Panos, DAYAL, Umeshwar, KARAGIANNIS, Anastasios ir TZIOVARA, Vasiliki. Benchmarking ETL workflows. Iš : *Technology Conference on Performance Evaluation and Benchmarking*. 2009. p. 199–220.
52. MAJCHRZAK, Tim A, JANSEN, Tobias ir KUCHEN, Herbert. Efficiency Evaluation of Open Source ETL Tools. Iš : *Proceedings of the 2011 ACM Symposium on Applied Computing*. [interaktyvus]. New York, NY, USA : Association for Computing Machinery, 2011. p. 287–294. SAC '11. ISBN 9781450301138. Prieiga per: doi: 10.1145/1982185.1982251.
53. JOSHI, Ankur, KALE, Saket, CHANDEL, Satish ir PAL, D Kumar. Likert scale: Explored and explained. *British journal of applied science & technology*. 2015. Vol. 7, no. 4, p. 396–403.

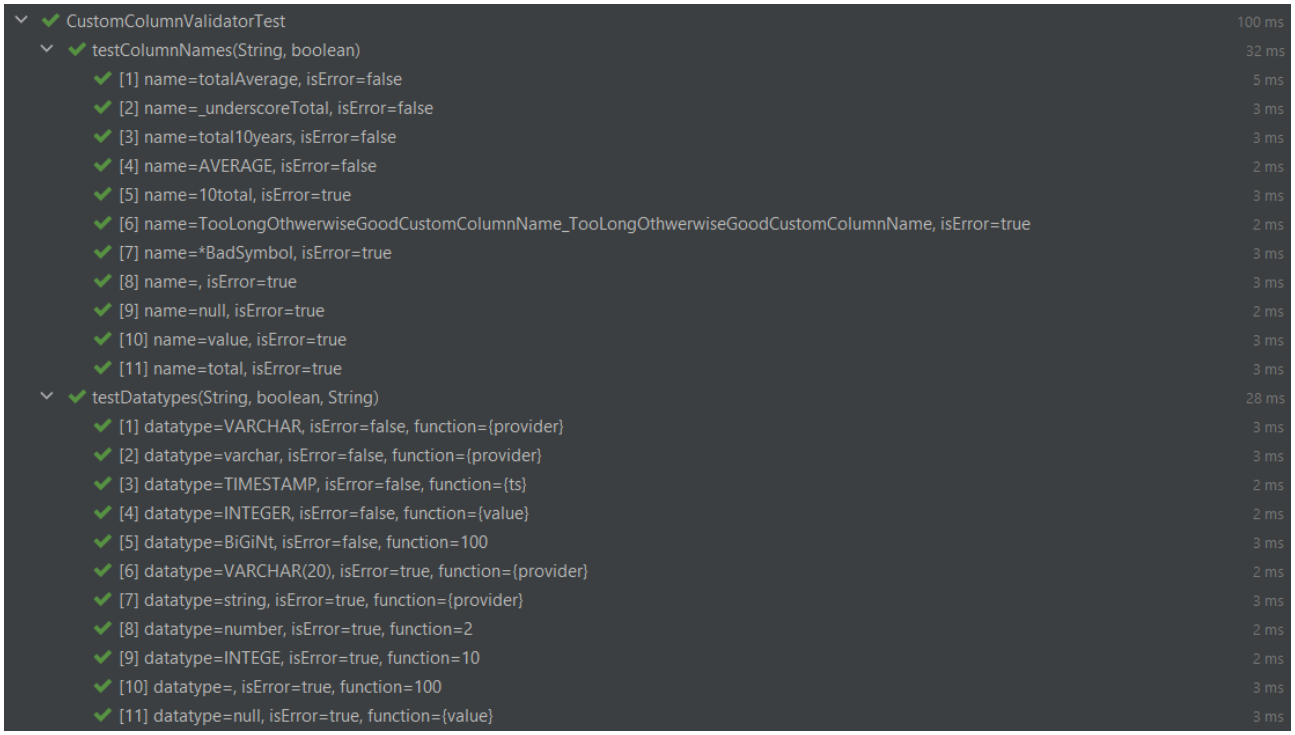
Priedai

1 priedas. Automatiniai testai

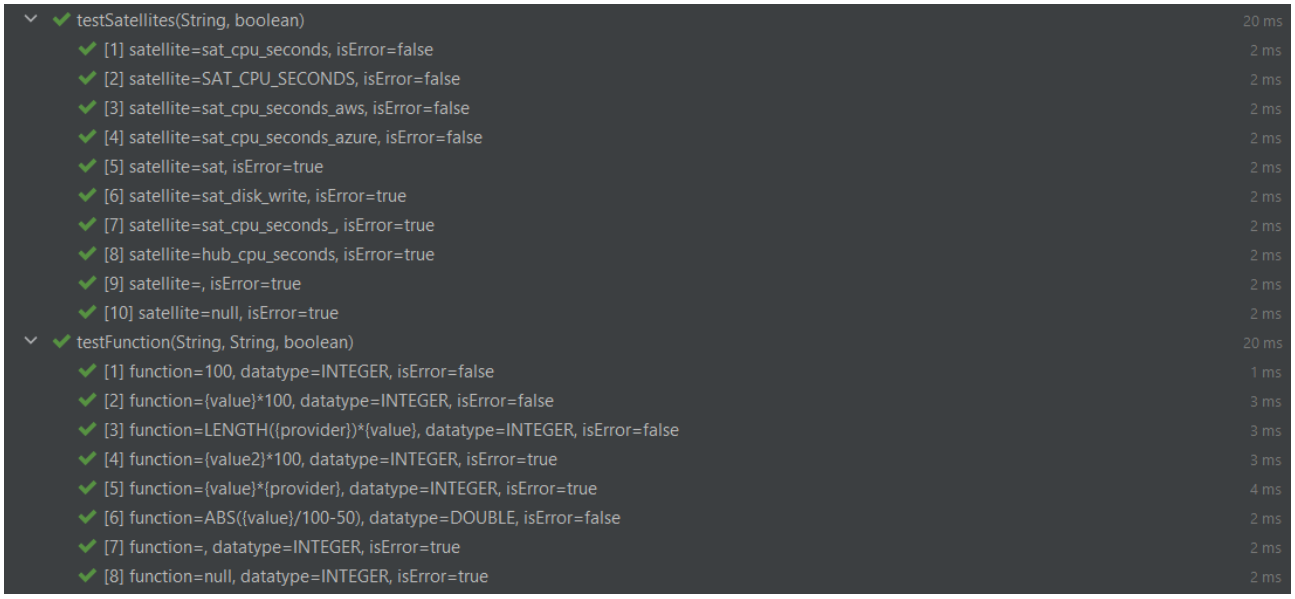
Metrikų saugyklos įrankio transformuojamų struktūrų korektiškumui užtikrinti buvo parengti automatiniai testai. Testai pateikti žemiau esančiuose paveiksluose (žr. 1 priedo 1 – 1 priedo 6 pav.).



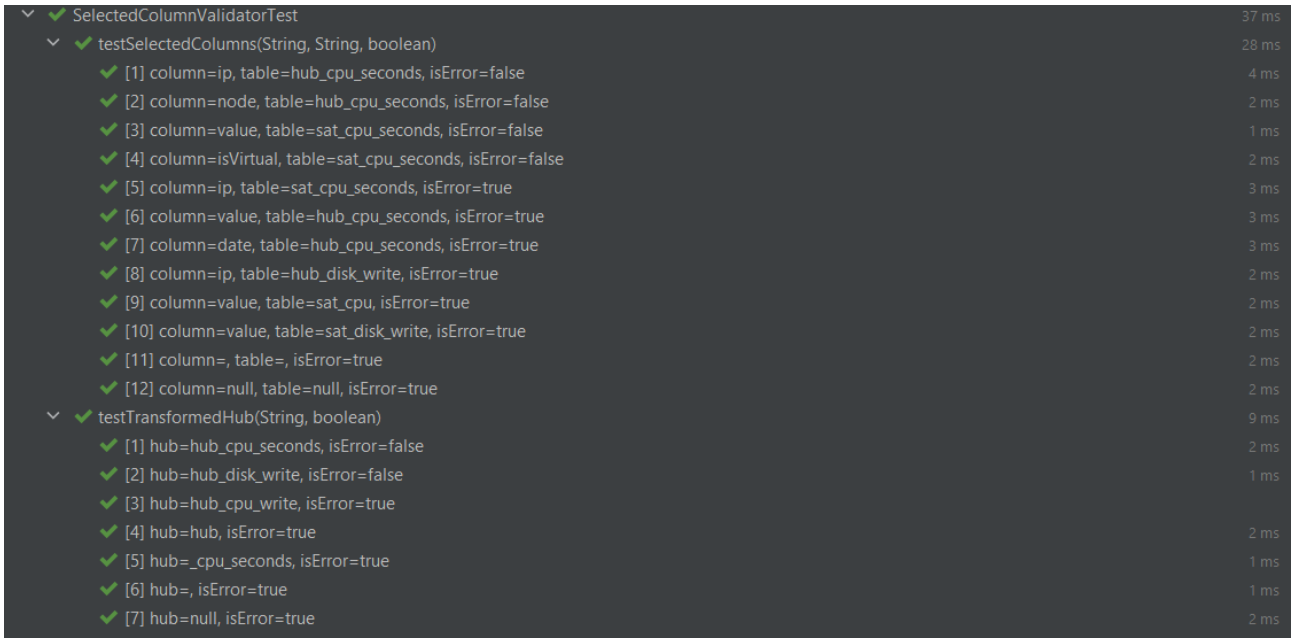
1 priedo 1 pav. Metrikų kategorijos testai



1 priedo 2 pav. Skaičiuojamųjų stulpelių testai (1)



1 priedo 3 pav. Skaičiuojamųjų stulpelių testai (2)



1 priedo 4 pav. Pasirenkamųjų stulpelių testai

1 priedo 5 pav. Kodo analizatoriaus testai (1)

1 priedo 6 pav. Kodo analizatoriaus testai (2)

2 priedas. Diegimo aktas

VaultSpeed

Kauno Technologijos Universitetui

DIEGIMO AKTAS

2024-03-14

Patvirtiname, kad studento Karolio Butkaus realizuotas baigiamojo magistro projekto „Data Vault metrikų saugyklos sudarymo metodika“ prototipas yra įdiegtas ir naudojamas įmonės reikmėms.

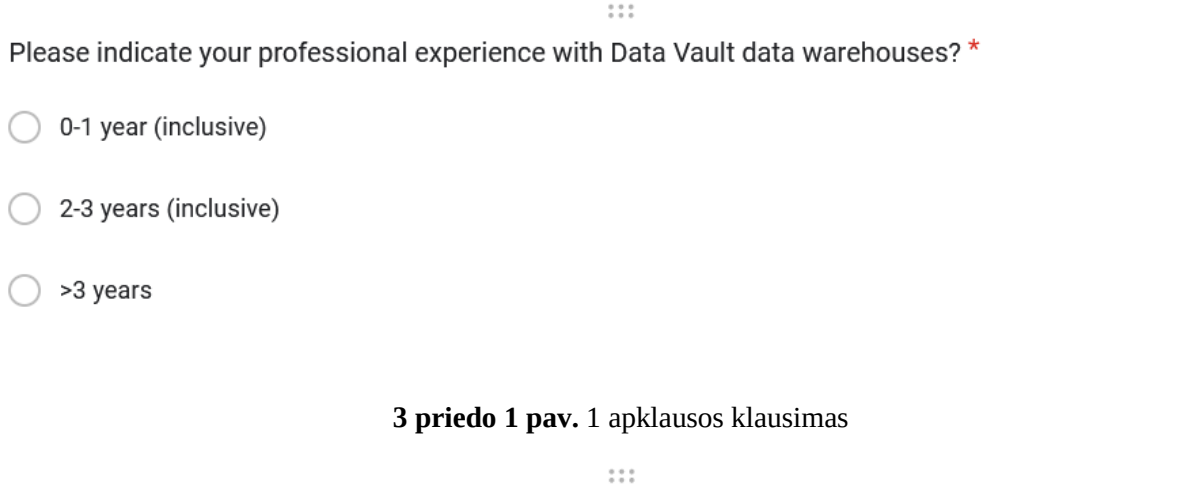
Šiuo metu įdiegtas prototipas yra testuojamas siekiant įvertinti prototipo galimybes ir reikalingus funkcionalumo patobulinimus. Sėkmingai pabaigus testavimo etapą ir išplėtus prototipo funkcionalumą sukurtą sprendimą numatoma integruoti į įmonės kuriamą produktą taip išplečiant klientams teikiamą funkcionalumą ir suteikiant galimybę kurti metrikų saugyklą pagal parengtą metodiką.

Gytis Repečka
Sprendimų architektas

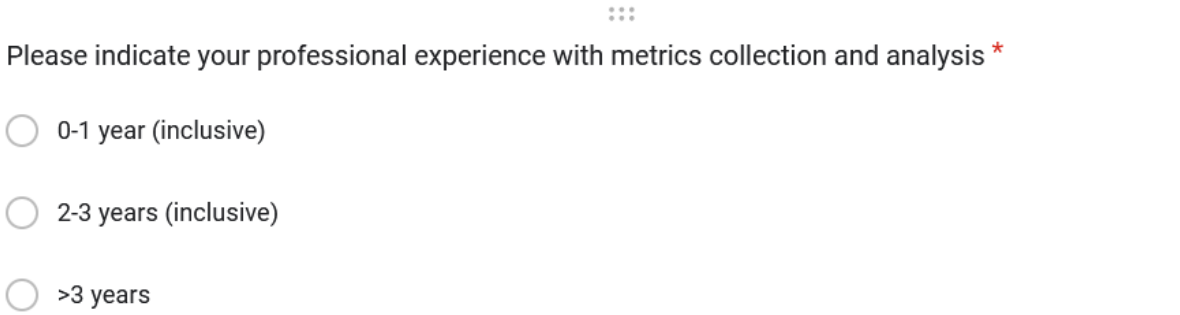
G. Repečka

3 priedas. Apklausos klausimynas

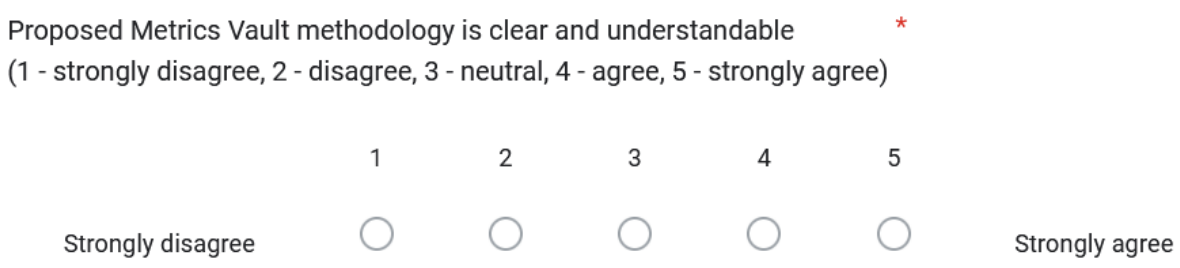
Eksperimentui įvykdyti buvo parengta apklausa, kurioje buvo pateiktas klausimynas sudarytas iš klausimų anglų kalba pavaizduotų 3 priedo 1 – 3 priedo 8 paveiksluose.



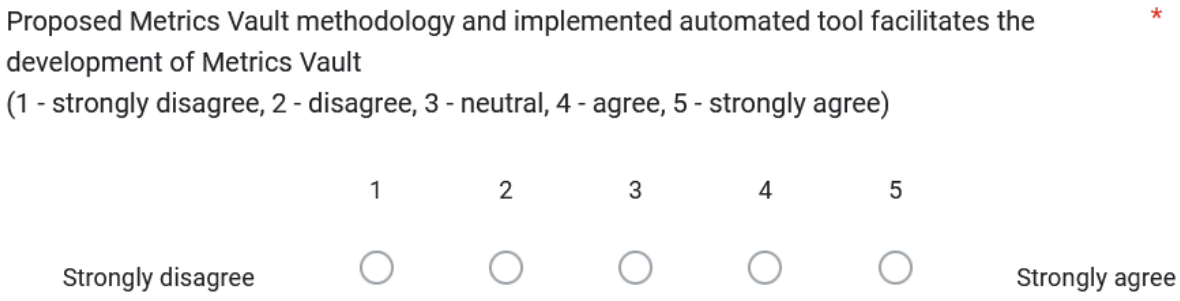
3 priedo 1 pav. 1 apklausos klausimas



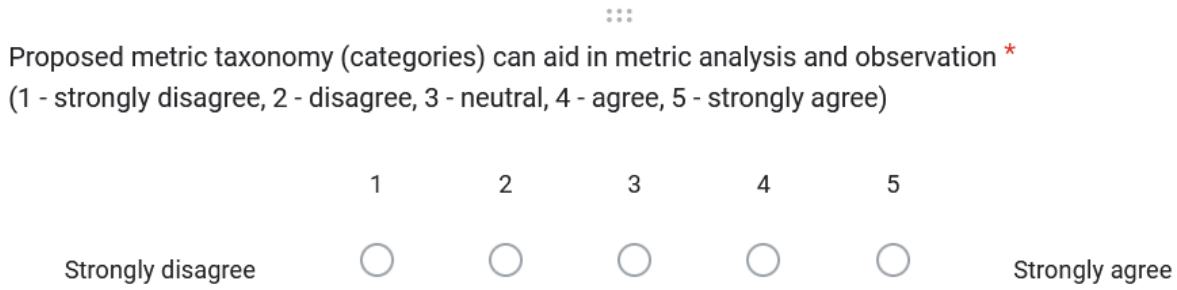
3 priedo 2 pav. 2 apklausos klausimas



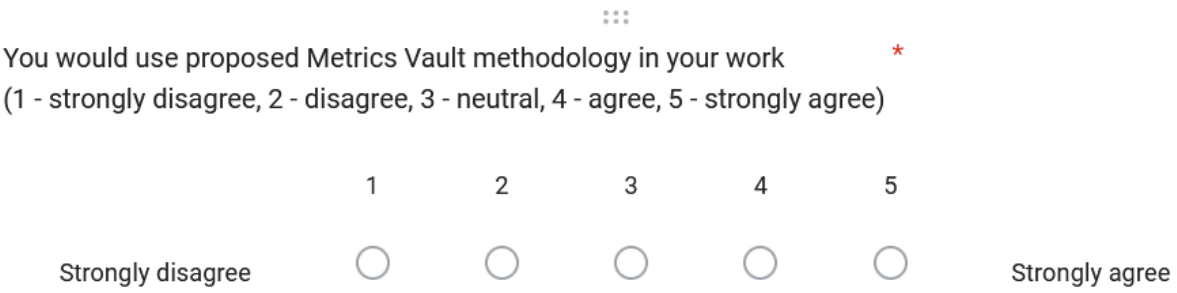
3 priedo 3 pav. 3 apklausos klausimas



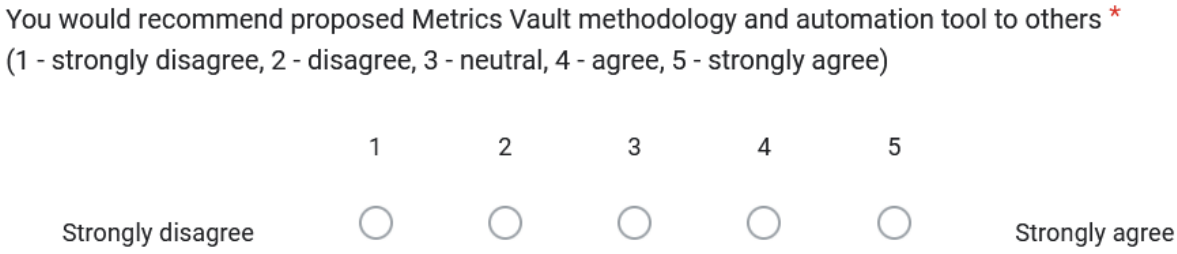
3 priedo 4 pav. 4 apklausos klausimas



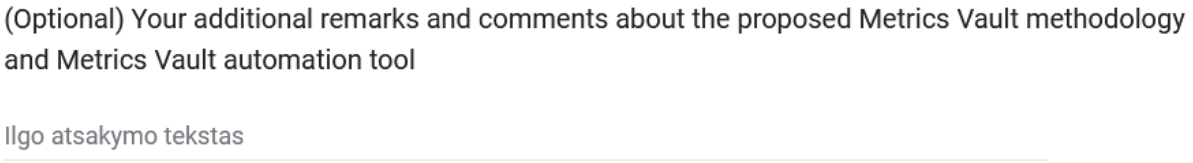
3 priedo 5 pav. 5 apklausos klausimas



3 priedo 6 pav. 6 apklausos klausimas



3 priedo 7 pav. 7 apklausos klausimas



3 priedo 8 pav. 8 apklausos klausimas