KLAIPĖDOS UNIVERSITETAS
GAMTOS IR MATEMATIKOS MOKSLŲ FAKULTETAS
INFORMATIKOS KATEDRA

AUDRIUS RIMA
Informatikos studijų studentas

VERSLO TAISYKLIŲ PANAUDOJIMAS DUOMENŲ ANALIZEI METAMODELIŲ TRANSFORMACIJŲ PAGRINDU

Baigiamasis magistro darbas

Mokslinis vadovas: prof. dr. Olegas Vasilecas
Konsultantas: lekt. Aidas Šmaižys

KLAIPĖDA, 2007

Klaipėdos universitetas
Gamtos ir matematikos mokslų fakultetas
Informatikos katedra

Baigiamasis magistro darbas
Verslo taisyklių panaudojimas duomenų analizei metamodelių transformacijų pagrindu
Audrius Rima

Anotacija

Didėjantis duomenų kiekis šiuolaikinėse informacinėse sistemose verčia ieškoti geresnių ir patogesnių priemonių ir metodų šių duomenų analizei. Esant dideliam duomenų kiekiui žmogus nebegali aprėpti informacijos įvairovės, atrasti logines sąsajas tampa sudėtinga, todėl reikalingos priemonės, kurios palengvintų, automatizuotų ir intelektualizuotų duomenų analizę.

Šiame darbe nagrinėjamas verslo taisyklių pritaikymas intelektualizuotai duomenų analizei. Darbe pasiūlomas metodas leidžiantis verslo taisykles, užrašytas XML kalba, transformuoti iki daugiamatės duomenų analizės instrukcijų programų sistemoje. Pasiūlytas metodas grindžiamas metamodelių transformacijomis. Darbe siūlomas metodas patikrintas eksperimentu, be to jis realizuotas programų sistemos prototipe.

Reikšminiai žodžiai: verslo taisyklės, verslo taisyklių transformavimas, verslo taisyklių vaizdavimas, XML, MDX, OLAP.

Klaipėda University
Faculty of Natural Sciences and Mathematics
Computer Science Department

MSc thesis
Metamodel based transformation method for business rule driven data analysis
Audrius Rima

Annotation

Rising amount of data in information system require to search better and usable tools and methods for this data analysis. When is large data amount, then people can't see diverseness of information, there is complicated to find logical links, therefore required tools, which can make data analysis usual, automation and intelligent.

The paper describes business rule using for intelligent data analysis and offers a method for transformation of a business rule, described in XML language, into the multidimensional data analysis rules in the program system. The method based on metamodel transformations. There is offered method, which is validated by experiment and implemented in prototype of software system.

Keywords: Business rules, business rules representation, transformation of business rules, XML, MDX, OLAP

Informacija yra vertingiausia iš visko, kas sukurta žemėje.

Dėkoju magistrinio darbo vadovui prof. dr. Olegui Vasilecui, magistrinio darbo konsultantui Aidui Šmaižiui, Informatikos katedros vedėjui doc. dr. Vitalijui Denisovui ir visam Informatikos katedros dėstytojų kolektyvui už studijų metais suteiktas žinias, už skatinimą tobulėti.

TURINYS

PRIEDAI:

1. Straipsnis paskelbtas leidinyje „Informacinės technologijos 2007“.
2. Straipsnis paskelbtas 10-ojoje jaunųjų mokslininkų konferencijoje.
3. Verslo taisyklų vaizdavimo kalbos SRML XSD schema.
4. MDX kalbos metamodelio XSD schema.
5. XSLT transformacijos schema, VT transformavimui į IS lygmens taisyklę.

6. XSLT transformacijos schema, taisyklės iš IS lygmens transformavimui į MDX konstrukcija.
7. XSLT transformacijos schema, VT transformavimui į SQL sakinį
8. Kompaktinis diskas su programų sistemos prototipu ir eksperimentų metu medžiaga.

PAVEIKSLŲ RODYKLĖ

LENTELIŲ RODYKLĖ

ĮVADAS

Temos aktualumas

Verslo taisyklių panaudojimo informacinių sistemų kūrimui tema šiuo metu yra labai populiari ir plačiai nagrinėjama moksliniuose straipsniuose. Labai svarbus uždavinys šiuolaikiniame informacinės sistemos kūrimo etape yra tiksliai ir aiškiai nustatyti vartotojo poreikius, surinktas žinias kaupti, saugoti ir panaudoti. Šiuolaikinės įmonės didžiąją dalį duomenų kaupia duomenų bazėse, o versle naudojamos informacinės sistemos yra skirtos ne tik tų duomenų įvedimui ir apdorojimui, bet ir analizei. Analizės rezultatas dažniausia yra tam tikra informacija, kuri pateikiama įvairių ataskaitų pavidalu. Tokia informacija yra naudojama sekti, ar esama padėtis versle atitinka verslo strategiją ir taktiką, ar laikomasi nustatytos verslo politikos ir taisyklių. Duomenų analizė leidžia gauti informacijos, reikalingos verslo sistemos pokyčių prognozei ir įtakos analizei bei rizikos ir galimos naudos įvertinimui, be to, ji reikalinga priimant sprendimus.

Paskutiniu metu populiarėja daugiamatė duomenų analizė, tačiau sukurta programinė įranga reikalauja specifinių žinių, kurių su programa dirbantis analitikas ar verslo atstovas gali neturėti. Todėl reikalingos priemonės, kur daugiamatės duomenų analizės instrukcijos pateikiamos verslo atstovui suprantama kalba, be to suteiktų galimybę automatizuoti duomenų analizę.

Šiame darbe nagrinėjama žiniomis pagrįsta duomenų analizė, kur žinios užrašytos panaudojant verslo taisykles. Darbe pasiūlytas metodas, pagrįstas metamodelių transformacijomis, leidžiantis verslo taisykles transformuoti iki daugiamatės duomenų analizės instrukcijų. Pasiūlytas metodas leidžia intelektualizuoti ir automatizuoti duomenų analizę.

Tyrimo objektas

Tyrimo objektas yra verslo taisyklėmis grindžiamų informacinių sistemų kūrimo metodika, akcentuojant verslo taisyklių panaudojimą intelektualiai duomenų analizei.

Problematika

Verslo taisyklių taikymas intelektualiai duomenų analizei ir sprendimų paramos sistemose.

Darbo tikslas ir uždaviniai

Darbo tikslas – išplėtoti metodą, leidžiantį XML kalba užrašytas verslo taisykles transformuoti į duomenų analizės instrukcijas SQL arba MDX kalbomis, kurios gali būti įvykdomos programų sistemoje.

Darbo uždaviniai:

1. Išanalizuoti su šio darbo problematika susijusią literatūrą.

2. Ištirti verslo taisyklių vaizdavimo XML kalba metodus ir naudojamus metamodelius ir kokiu tikslu juos galėtų panaudoti.
3. Išanalizuoti verslo taisyklių transformavimą į informacijos apdorojimo taisykles ir IS taisyklių realizavimo metodus programų sistemoje.
4. Atlikti sprendimų paramos sistemų tyrimą ir informacijos analizės taisyklių galimybes panaudojant: OLAP technologijas, duomenų saugyklose, daugiamatės duomenų analizės užklausas - MDX kalba
5. Suformuluoti ir aprašyti metodą, leidžiantį verslo taisykles transformuoti iki daugiamatės duomenų analizės instrukcijų.
6. Realizuoti pasiūlytą metodą ir sukurti IS prototipą bei atlikti sukurto prototipo bandymus.

Temos naujumas

Tema yra nauja. Verslo taisyklės sąvoka atsirado gana neseniai. Dažniausia verslo taisyklė siejama su aktyviomis duomenų bazių sistemomis. Programų sistemoje dažniausia verslo taisyklės realizuojamos darnos ribojimais ir trigeriais, tačiau literatūroje taip pat nagrinėjama ir verslo taisyklių ne vien aktyviose duomenų bazių sistemose.

Programinės įrangos ir duomenų bazių sistemų gamintojai, paskutiniu metu pakankamai ištobulino duomenų saugyklų ir OLAP programinę įrangą. Todėl straipsniuose šį technologija vis labiau tyrinėjama, siūlomi nauji metodai ir priemonės duomenų analizės intelektualizavimui ir automatizavimui. Žiniomis grindžiama duomenų analizė, naudojant OLAP technologiją, kur žinios būtų užrašytos verslo taisyklėmis, išnagrinėta nepakankamai.

Tyrimo metodai

Tyrimas atliktas naudojant literatūros apžvalgos, lyginamosios analizės, sisteminės analizės ir eksperimentinės analizės metodus.

Mokslinė darbo vertė

Darbe pasiūlytas metodas leidžiantis panaudoti verslo taisykles, pavaizduotas XML kalba, transformuoti iki daugiamatės duomenų analizės instrukcijų, transformacijai naudojant XSLT schemas.

Darbo rezultatai

- Pasiūlytas ir aprašytas metodas verslo taisyklių, užrašytų XML kalba, transformavimui iki daugiamatės duomenų analizės instrukcijų. Metodas pagrįstas metamodelių transformacijomis.
- Eksperimentais patikrintas pasiūlytas metodas.

- Sukurtas siūlomą metodą realizuojantis programų sistemos prototipas. Eksperimentu parodyta, kaip iš verslo taisyklės, atvaizduotos XML kalba, gauti SQL ar MDX instrukciją.

Viešas darbo rezultatų pristatymas

Svarbiausi darbo rezultatai pateikti straipsnyje leidinyje „Informacinės technologijos 2007. Konferencijos pranešimų medžiaga“ [20].

Taip pat darbo rezultatai pateikti pranešime, skaitytame 2007 m. balandžio 13 d., Vilniaus Gedimino technikos universitete vykusioje 10-ojoje jaunųjų mokslininkų konferencijoje “Lietuva be mokslo – Lietuva be ateities”. Pagal šį pranešimą parengtas straipsnis [21].

Darbo sandara

Pirmame skyriuje apžvelgiama verslo taisyklių problematiką nagrinėjanti literatūra, bei pateikiama verslo taisyklių analizė ir jos rezultatai. Antrame skyriuje nagrinėjama verslo taisyklių poaibio, skirto informacijos apdorojimui, panaudojimas duomenų analizei ir sprendimų paramos sistemose. Trečiame skyriuje aprašomas metodas leidžiantis verslo taisykles, pavaizduotas XML kalba, transformuoti iki MDX instrukcijos, transformacijai naudojant XSLT schemas. Ketvirtame skyriuje pateikti aprašyto metodo eksperimentų rezultatai.

1 INFORMACINIŲ SISTEMŲ KŪRIMO VERSLO TAISYKLIŲ PAGRINDU ANALIZĖ

Verslo taisyklės (VT) yra neabejotinai svarbios organizacijoms, jos apibrėžia kaip organizacijos vykdo verslą. VT reikšmė taip pat buvo pripažinta informacinių sistemų kūrime, daugiausia dėl jų galėjimo kurti lanksčias ir leidžiančias keistis aplikacijas.

Verslo taisyklių šaknys siekia dirbtinį intelektą (angl. *Artificial Intelligence*), kur jos sėkmingai pritaikytos žinių vaizdavimui. Žiniomis grindžiamose sistemose žmogaus eksperto žinios ir samprotavimai gali būti renkamos ir kaupiamos kompleksiniuose tinkluose, sudarytuose iš taisyklių. Taisyklės paprastai aprašomos deklaratyviomis kalbomis, kurios neduoda suprasti sekos ar srauto kontrolės.

Taisyklės yra saugomos taisyklių bazėje ir apdorojamos specialiu komponentu, vadinamu išvedimo mašina (angl. *Inference engine*). Išvedimo mašina įvertina taisyklių sąlygas ir bet kokiu laiko momentu nustato, kuri taisyklė yra tinkama vykdyti.

Platus verslo taisyklių tyrinėjimas susijęs su duomenų bazių tyrinėjimu. Kaip rezultatas šių pastangų pasirodė aktyvios duomenų bazės, kurios, lyginant su pasyviomis duomenų bazėmis, pavartojo aktyvius komponentus, tokius kaip tirgerius ar duomenų bazės procedūras savo duomenų vientisumo įvykdymui. Kitas su duomenų bazėmis susijęs tyrinėjimas apima deduktyvias (angl. *deductive*) duomenų bazes. Tuo tarpu tradicinės duomenų bazių sistemos moka naudotis tik papildomomis žiniomis, įtvirtintomis faktuose ir reikalavimuose (angl. *instances*), deduktyvios duomenų bazės prideda iš anksto apgalvotas žinias (angl. *intentional knowledge*), kurios yra virš DB faktinio turinio. Šis žinių požymis gali būti pilnai specifikuotas taisyklėmis ir yra saugomas taisyklių bazėje, prieš tai kai duomenų bazė yra žinoma.

1.1 Verslo taisyklės samprata

Verslo taisyklių pagrindas atėjo iš dirbtinio intelekto (angl. *Artificial intelligence*) srities, kur taisyklės sėkmingai taikomos žinių vaizdavimui. Daugelis autorių bando aiškiai ir trumpai apibrėžti verslo taisyklės sąvoka. Verslo taisyklės sąvoka apibrėžiama iš daugelio perspektyvų (pvz. verslo, informacinių sistemų).

„Verslo taisyklių grupė" (angl. *Business Rules Group*) šią sąvoką iš informacinių sistemų perspektyvos formaliai apibrėžia taip:

> *From the information system perspective,*
> *...a business rule is a statement that defines or constrains some aspect of the business. It is intended to assert business structure, or to control or influence the behavior of the business* [5].

... verslo taisyklė yra nuostata, kuri apibrėžia arba apriboja tam tikrą verslo požiūrį. Ji skirta apginti verslo struktūrą arba įtakoti verslo aplinką.

Pagal R.G. Ross [23] verslo taisyklė – tai nuoroda, įtakojanti arba vadovaujanti verslo elgsenai, funkcionavimui (angl. *behavior*).

Verslo procesai gali būti specifikuojami verslo taisyklėmis, bet kita vertus ir verslo taisyklės gali būti detalizuojamos procesais, o įvedus subprocesus tuo pačiu ir įgyti tam tikrą hierarchinę struktūrą.

1.2 Verslo taisyklių klasifikacija

Šiuo metu nėra nusistovėjusio vieningo verslo taisyklių klasifikavimo, stinga išbaigtos, suderintos ir visuotinai priimtinos taisyklių tipų sistematikos. Nesant bendro susitarimo skirtingos organizacijos taisykles klasifikuoja savaip.

Verslo taisyklės dažniausia skirstomos į tris klases [37]:

- Vientisumo apribojimai (angl. *integrity constraints*), taip pat dar vadinamos suvaržymo taisyklėmis (angl. *constraint rules*) arba apribojimo taisyklėmis (angl. *integrity rules*).
- Kilmės taisyklės (angl. *derivation rules*).
- Veiksmo arba reagavimo taisyklės (angl. *reaction rules*).

Kitas populiarus VT klasifikavimo būdas, R. Ross sukurta klasifikavimo sistema:

- Faktai (PVM yra 18%)
- Sąlygos (Geras pelnas > 30%)
- Taisyklės (j*ei pelnas < 30% analizuoti pardavimus)*
- Apribojimai (*Įsipareigojimai < turtas/2)*
- Išvedimai (*Prekės kaina = kaina + PVM)*
- Išvados (*Jei pirkėjas perka už > 100Lt taikyti nuolaidas)*
- Laiko paskirstymo taisyklės (s*utartis turi būti įforminta per 9 min.)*
- Sekos (*Atspausdinti sutartį, suderinti su ...)*
- Euristikos (*13d. Penktadienį verslas nesiseka)*

Savo prigimtimi VT skirtos verslo ypatumams formaliai aprašyti. Kaip tai yra padaroma ir kokių rezultatų tikimasi iš tokių formalių taisyklių aprašų dažniausia priklauso nuo pačių inžinierių požiūrio ir naudojamų metodų. Ką galima aprašyti taisyklėmis – taisyklėmis apibrėžiami įvykiai, sąlygos ir galimi veiksmai.

Reikia pastebėti, kad verslo taisyklių klasifikacija priklauso nuo tikslo, kuriuo VT reikia suklasifikuoti.

1.3 Verslo taisyklių įsisavinimas

Surikiuoti informacinę sistemą ir verslo operacijas yra viena pagrindinė problema visose organizacijose. Ši problema dar nėra efektyviai išspręsta.

Verslo taisyklių perpratimas (angl. *acquisition*) yra nelengva užduotis. Iš daug taisyklių yra sunku identifikuoti verslo taisykles. Ypatingai tai taikoma taisyklėms, kurios neturi aiškaus vaizdavimo [2].

Dauguma taisyklių, kurios yra naudojamos versle neturi formalaus vaizdavimo. Jos yra grindžiamos žodžiais neišreikštomis žiniomis (angl. *tacit knowledge*) ir yra dinamiškos aplinkoje ir sunkiai išreiškiamos. Jos yra labai asmeniškos ir subjektyvios, grindžiamos patirtimi, idealais, emocijomis ir intuicija [2].

Sutinku su M. Bajec ir M. Krisper nuomone, kad perprasti verslo taisykles IS kūrėjui nėra lengva, nes verslo atstovas kartais negali VT išreikšti žodžiais, t.y. neturi VT formalaus vaizdavimo.

Kadangi verslui reikalingas lankstumas sandėriuose realiame pasaulyje D. Rosca ir C. Wild straipsnyje [22] siūlo architektūrą skirta, verslo taisyklių išgavimui (angl. *elicitation*), išskleidimui (angl. *deployment*) ir evoliucijai (angl. *evoliution*). Ši architektūra prižiūri nesuderinamumo (angl. *inconsistency*), neužbaigtumo (angl. *incompleteness*) ir dviprasmiškumo (angl. *ambiguity*) problemas, kurios yra neišvengiamos keičiantis aplinkai.

1.4 Verslo taisyklių ir procesų modeliavimas

Verslo taisyklės turi būti suprantamos ir informacinės sistemos kūrėjams, ir verslo atstovams, taip pat kitiems žmonėms, dalyvaujantiems IS kūrime. Dažniausia verslo žmonės nėra susipažinę su formaliomis kalbomis. O IS kūrėjams reikia vienareikšmiškų verslo taisyklių deklaracijų, kad sukurti programos kodą, verslo taisyklių vykdymui. Iš čia seka, kad verslo taisyklėms reikia skirtingo formalumo lygio specifikacijų, kurios būtu suprantamos visiems su verslo taisyklėmis dirbantiems žmonės.

Formali specifikavimo kalba nė problema. Be tradicinių metodų (pvz. UML diagramos) ir technikų yra ir kitų sprendimų verslo taisyklių modeliavimui: išorinių taisyklų kalba (angl. *External rule language*), objektų apribojimų kalba (angl. *Object Constraint Language*), Ross metodas (angl. *RossMethod*) [2].

Problema yra rasti neformalią kalbą, kuri būtų suprantama verslo atstovams ir būtų vis dar aiški ir vienareikšmiška.

Norint, kad verslo modeliavimo, projektavimo, analizavimo procesas būtų suprantamas tiek informacinių technologijų specialistams, tiek verslo žmonėms, vis dažniau naudojamos grafinio

modeliavimo kalbos. Grafinio modeliavimo privalumai – standartiniai žymėjimai, vieninga terminologija, intuityvus supratimas. Pagrindinės verslo procesų modeliavimo kalbos: UML 2.0, BPMN (*Business process modelling notation*) bei BPEL4WS (*Business Process Execution Language for Web Services*).

1.5 Verslo taisyklių įgyvendinimas

Vienas svarbiausių veiklų verslo taisyklėmis grindžiamų informacinių sistemų kūrime yra verslo taisyklių įgyvendinimas (angl. *implementation*). Yra daug skirtingų technologijų ir priemonių verslo taisyklių įgyvendinimui (realizacijai) ir palaikymui. Tony Morgan pateikia pagrindinės technologijas verslo taisyklių įgyvendinimui [15]:

- Programos kodas, skriptai arba kaip specializuotas taisyklių komponentas ;
- Taisyklių varikliai, kurie paprastai turi loginio serverio formą su specifine taisyklių kalba ir veiksmų stiliumi;
- Duomenų bazėse, kuriose siūloma keletas kelių taisyklių įgyvendinimui;
- *Workflow* sistemos su vidiniu varikliu ir įprastais stiliais taisyklių kodavimui;
- *Look-up* lentelės, kurios turi iš anksto sukompiliuotas žinias formoje, kuri suteikia greitą ir patogų priėjimą.

Viena iš pagrindinių priežasčių kurti priemonės verslo taisyklių įgyvendinimui yra programos kodo generavimas iš verslo taisyklių specifikacijų. Perspektyvi idėja išspręsti šią problemą yra *Model Driven Architecture* (MDA). MDA apibrėžia du tipus modelių nuo platformos nepriklausomas modelis (angl. *Platform Independent Model,* PIM) ir specifinis platformos modelis (angl. *Platform Specific Model,* PSM). PIM aprašo sistemą apie be jokių detalių apie platformą, kurioje bus įgyvendinama planuojama (projektuojama) sistema. PSM charakterizuoja planuojamą sistemą konkrečioje platformoje (pvz. NET, J2EE).

MDA koncepcija gali būti panaudota verslo taisyklėmis grindžiamame kūrime, kur verslo taisyklės yra transformuojamos iš specifikacijų į kodą. Keletas sąlygų kaip tai gali būti įgyvendinta [23]:

- Reikalingos atitinkamos kalbos PIM ir PSM. Kadangi abu PIM ir PSM modeliai reikalauja formalios kalbos, papildomas lygis abstrakcijos (PIM abstrakcija) turi turėti apibrėžtą palaikymą verslo taisyklių specifikavimui formoje, kuri būtu suprantama verslo atstovams. Tai reiškia, kad formalios kalbos tokios kaip UML ar OCL, nėra pasirinkimas.
- Įskiepiai (angl. *plug-ins*) turi būti parašyti taip, kad palaikytų PIM transformacijas skirtingose technologijose, kurios yra naudojamos verslo taisyklių apdorojimui ir vykdymui. Be to, turi būti palaikomas automatiška verslo kalbos (verslo taisyklės

abstrakcijos aukščiausias lygis) transformacija į PIM. Čia taip pat susiduriama su problemomis.

- Kadangi MDA yra tik koncepcija, todėl reikalingas atitinkamas įrankis MDA palaikymui.

Pritariu M. Bajec ir M. Krisper teiginiui, kad MDA koncepcija galima panaudoti verslo taisyklėmis grindžiamam kūrimui, jei būtų išspręstos aukščiau išvardintos problemos. Pakankamai svarbi problema yra verslo taisyklių transformacija iš vieno lygmens į kitą, iš verslo sistemoje naudojamo VT aprašo į IS objektus ir pan.

1.6 Verslo taisyklių valdymas

Jei verslo taisyklės nėra tvarkomos tradiciniais keliais, netiesiogiai per kitus informacinės sistemos aspektus (pvz. duomenys, procesai, funkcijos, objektai ir t.t.), tačiau kreipiamasi tiksliai, tada joms reikalingas atitinkamas tvarkymo (angl. *management*) lygis. Tai apima verslo taisyklų perpratimą, specifikaciją, modeliavimą, įgyvendinimą ir taip pat priežiūrą, tokią kaip versijų kontrolė, pasikeitimų kontrolė, taisyklių efektyvumo monitoringas.

Bajec M. pateikia verslo taisyklių valdymo problemą:

Aplinkos nustatymui, kurioje gali būti tvarkomos visos organizacijos verslo taisyklės, reikalingas ekstensyvus darbas, kuris ne visoms organizacijoms gali būti pakankamai motyvuojamas. Todėl būtina rasti kelius išlaidų minimizavimui, kurios atsiranda su organizacijos taisyklių vadyba, galbūt integracija su procesais, kurie užsiima verslo analize, tokia kaip strateginis planavimas, verslo procesų reinžinerija, informacinės sistemos atnaujinimas [2].

M. Bajec ir M. Krisper straipsniuose [1, 3] nagrinėja VT valdymo įmonėje problemą. Jie apibrėžia pagrindines veiklas, kurias apima VT valdymo procesas, t.y. nuo VT surinkimo-įsisavinimo iki realizacijos IS lygmenyje.

Jie išskiria VT valdymo įrankių grupes pagal, suteikiamas tam tikras, reikalingas galimybes [3]:

1. VT įrankiai skirti taisyklėmis grindžiamų IS kūrimui.
2. VT įrankiai skirti kurti žiniomis grindžiamas aplikacijas.
3. VT įrankiai skirti visos įmonės verslo taisyklių valdymui

Bet šie egzistuojantys įrankiai nepalaiko pilnai visų verslo taisyklių valdymo įmonėje, nes egzistuojantys VT įrankiai siūlo tik ribotą paramą įmonės modeliavime, VT valdymo proceso tikslas yra tvarkyti verslo taisykles visoje informacinėje sistemoje ir ne tik IS lygmenyje.

Mano manymu, jei rinkoje būtų toks įrankis, kuris leistų atlikti VT valdymą įmonėje, nuo taisyklių surinkimo iki jų įgyvendinimo IS, būtų VT versijų kontrolė, VT keičiant VT būtų automatiškai pakeistos VT ir IS lygmenyje, turėtų būti išspręstos visos problemos, su kuriomis

susiduriama nagrinėjant, IS kūrimą VT požiūriu. Sukūrus tokį įrankį IS galėtų iškart prisitaikyti prie naujos įmonės politikos, verslo taisyklių, t.y. būtų išvengiama šiuo metu brangaus IS modifikavimo ar pakeitimo, naudojant tradicines metodikas.

Kadangi verslo taisyklės gali nuolat keistis dėl vidinių ir išoriniu veiksnių (pvz. įstatymų pasikeitimas), todėl organizacijoje turi būti taisyklių pakeitimas pakankamai motyvuojamas.

Verslo taisyklių valdyme išskiriamas svarbus komponentas verslo taisyklių repozitorius. Tai yra viena svarbesnė dalis, be kurios negalima realizuoti VT grindžiamos informacinės sistemos.

1.7 Verslo taisyklių vaizdavimas

1.7.1 Verslo taisyklių vaizdavimas būdai

Pats paprasčiausias būdas užrašyti taisyklę, be abejo yra paprastas tekstas natūralia kalba. Dažniausia versle naudojamos taisyklės yra neformalios, todėl jas reikia formalizuoti.

Galima panaudoti taisyklių vaizdavimo šablonus ir VT užrašyti natūralia kalba. Tačiau taip vaizduoti objektinės sandaros taisykles ne visada yra patogu. Ypač naudojant įvairias taikomąsias programų sistemas, skirtas informacinių sistemų projektavimui, žymiai efektyviau tokios taisyklės galėtų būti vaizduojamos UML diagramomis.

Žinių, o tuo pačiu ir taisyklių vaizdavimo kalbos kūrimu susidomėjo The Knowledge Sharing Effort (http://ksl.stanford.edu/knowledge-sharing). The Knowledge Sharing Effort iniciatyva buvo sukurtos KQML, Ontolingua ir KIF.

1.7.2 Verslo taisyklių vaizdavimas XML

Šiuo metu yra sukurta daug verslo taisyklių ir žinių vaizdavimo kalbų XML pagrindu. Kiekvieną tokią kalbą aprašo specialus DTD (*Document Type Definition*) dokumentas, kuriame išvardintos visos galimos struktūros, jų atributai ir apribojimai. Populiariausios iš jų yra RuleML, BRML ir SRML, kurios panaudotos eilėje komercinių produktų, skirtų verslo taisyklėmis pagrįstų informacinių sistemų kūrimui.

Verslo taisyklės gali būti įgyvendintos programoje jei-tai (angl. *if-then*) sakiniu. Tai pats paprasčiausias būdas, bet jei taisyklės pasikeistų, tai būtu brangu atnaujinti.

CommonRules turi klasikinę išvados darymo (angl. inference) struktūrą. Taisyklės pateikiamos sakiniu jei–tai (angl. *if-then*). Duomenys (pvz. faktai) sistemai suteikiami kaip įėjimas arba per duomenų bazės, failų interfeisą. Išvados darymo variklis sąlygoja sąryšį tarp taisyklių ir duomenų, ir pasiekia sprendimus arba išvadas [24].

CommonRules specialiai sukurtos e-verslo (angl. *eBusiness*) aplinkai. Žinios vaizduojamos yra XML kalbos forma, kuri vadinasi verslo taisyklių žymėjimo kalba (angl. *Business Rules*

Markup Language, BRML). BRML leidžia verslo partneriams keistis taisyklėmis (pvz. kainų nustatymas, užsakymo vykdymo laikas ir pan.) ir dinamiškai modifikuoti elgesį, kuris yra taikomas kai tariamasi dėl individualių partnerių specifinių poreikių. BRML grindžiama deklaratyvia logine programa ir tam tikra išplėsta forma, kuri vadinama Courteous Logic Programs.

Prieinamos verslo taisyklės (angl. *Accessible Business Rules*, ABR) yra karkasas, kuris leidžia verslo įmonei kurti paskirstytas verslo aplikacijas, kurios sistemingai išreiškia laiko- ir situacijos kintamojo dalis jų verslo logiką kaip išoriškai taikomas esybes vadinamas „verslo taisyklėmis“.

Straipsnyje [24] I. Rouvellou ir kt. taip pat nagrinėjama kaip sinergetiškai sujungti du produktus: „CommonRules“ ir Prieinamos verslo taisyklės. Teigiama, kad apjungus šiuos du produktus pašalinami jų trūkumai. Tačiau jų siūloma kalba turi dar trukumų.

Verslo taisyklų vaizdavimas (užrašymas) XML yra pakankamai daug nagrinėjamas, sukurta pakankamai daug užrašymo kalbų. Vaizdavimo kalba reiktų pasirinkti nuo tikslo, kur bus naudojamos taisyklės, kokie programiniai įrankiai bus naudojami taisyklių valdymui ir, taip pat priklauso nuo inžinieriaus projektuojančio IS.

1.7.2.1 Business Rules Markup Language

Verslo taisyklių žymėjimo kalba (angl. *Business Rules Markup Language, BRML*) yra XML taisyklių kalba skirta agentų komunikavimui, grindžiama įprastomis loginėmis programomis [6]. Ji naudojama su „CommonRules“ ir plėtojama kartu su IBM Business Rules E-Commerce projektu. Susijęs siūlymas yra duotas agentų bendravimo žymių kalba (angl. *Agent Communication Markup Language*) naujas XML versijos FIPA (*The Foundation for Intelligent Physical Agents*) juodraštinis standartas agentų bendravimo kalbos (angl. *Agent Communication Language*).

BRML leidžia verslo partneriams keistis taisyklėmis (pvz. kainos nustatymo, grąžinimo kriterijų, užsakymo aptarnavimo laiko ir pan.) ir dinamiškai modifikuoti elgseną, kuri yra naudojama, kai dalijamasi specifiniais poreikiais individualių partnerių. BRML grindžiama deklaratyviomis loginėmis programomis (angl. *declarative logic programs*) ir tam tikra išplėsta forma, taip vadinama pagarbios logikos programos (angl. *Courteous Logic Programs*)

1.7.2.2 Simple Rule Markup Language

Paprasta taisyklų žymėjimo kalba (angl. *Simple Rule Markup Language*, SRML) aprašo tam tikrą taisyklių kalbą, susidedančią iš kalbų pogrupio konstruktų paplitusių, tiesioginio išvedimo (angl. *forward-chaining*) taisyklių varikliuose. Kadangi SRML nenaudoja konkrečių firminių konkrečios kalbos konstruktų (angl. *constructs*) taisyklės specifikuotas naudojant XML DTD dokumentą galima lengvai suprasti ir įvykdyti kiek nors pritaikytoje taisyklių mašinoje

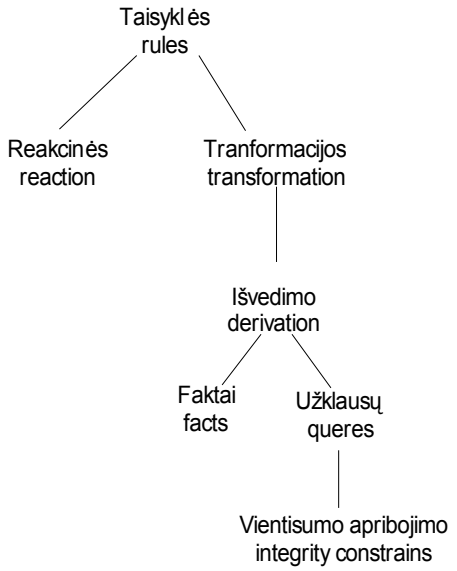
(angl. *rule engine*). Kaip apibrėžta DTD schemoje – taisyklių aibė (angl. *ruleset*) yra pagrindinis elementas SRML XML dokumento ir sudaro sąrašą taisyklių (angl. *rules*). Taisyklės turi sąlygos dalį [`conditionPart`] ir veiksmų dalį [`actionPart`]. Sąlygos dalis turi turėti bent vieną sąlygą. Sąlygos parašomos kaip testinės išraiškos ir gali būti paprastos sąlygos arba be sąlygos. Paprastos sąlygos gali būti suvaržytos kintamaisiais, kai nesąlygos negali. Taisyklės veiksmų dalis susideda iš veiksmų, kuriuose gali būti reguliuojamos deklaracijos (angl. *declaration*s) ir paskyrimai (angl. *assignments*), taip pat tradicinis tvirtinimas (angl. *assert*), įtraukimas (angl. *retract*) ir modifikavimas (angl. *modify*) taisyklių kalbos sakiniai. Tvirtinimo veiksmas prideda objektą į veikiančią atmintį. Įtraukimo veiksmas pašalina objektą iš veikiančios atminties (angl. *working memory*). Modifikavimo (angl. *modify*) veiksmas modifikuoja objektą veikiančioje atmintyje. Išraiškos pasirodo visoje kalboje ir gali būti priskiriamos (kintamieji ar laukai), konstantos (literalai tokie kaip strings, ints, floats, booleans ir pan.), aritmetinės ar loginės išraiškos. [25]

SRML kalbą labai lengva perprasti ir užrašyti verslo taisykles. SRML galima sėkmingai pritaikyti verslo taisyklių užrašymui, tai įrodyta A. Šmaižio darbe [26]. Taip pat šiame darbe aprašoma kaip taisyklės transformuojamos pasinaudojus XSLT schemomis į įvairius IS objektus.

1.7.2.3 Rule Markup Language (RuleML)

Rule Markup Initiative apsiėmė apibrėžti Rule Markup Language (RuleML), kurie leidžia abu teisiogines (angl. *forward*) (bottom-up) ir atbulines (angl. *backward*) (top-down) taisykles XML dedukcijai (angl. deduction), perrašymui (angl. *rewriting*) ir tolimesnėms išvadų darymo it transformacijos (angl. *inferential-transformational*) užduotims.

Rule Markup Initiative tikslas sukurti RuleML kaip kanoninę interneto (angl. *web*) kalba taisyklių, naudojant XML žymėjimui, formaliai semantikai ir efektyviam įgyvendinimui.



1 pav. ***RuleML taisyklių tipų klasifikacija***

RuleML iniciatyvos autoriai išskiria šešių tipų taisykles, kurių tarpusavio priklausomybės pavaizduotos grafiškai 1 pav. [31]. Tai vėl parodo, kad nėra nusistovėjusios taisyklių klasifikacijos.

RuleML kalba yra nuolatos vystoma, naudojama daugelyje projektų. RuleML kalbai yra sukurta DTD ir XSD schemos. RuleML kalba užrašyti taisyklę yra sudėtingiau negu SRML. Nors ir nėra sukurta SRML XSD schemos, tačiau DTD dokumentą galima transformuoti į XSD schemą ir ją naudoti taisyklių validavimui.

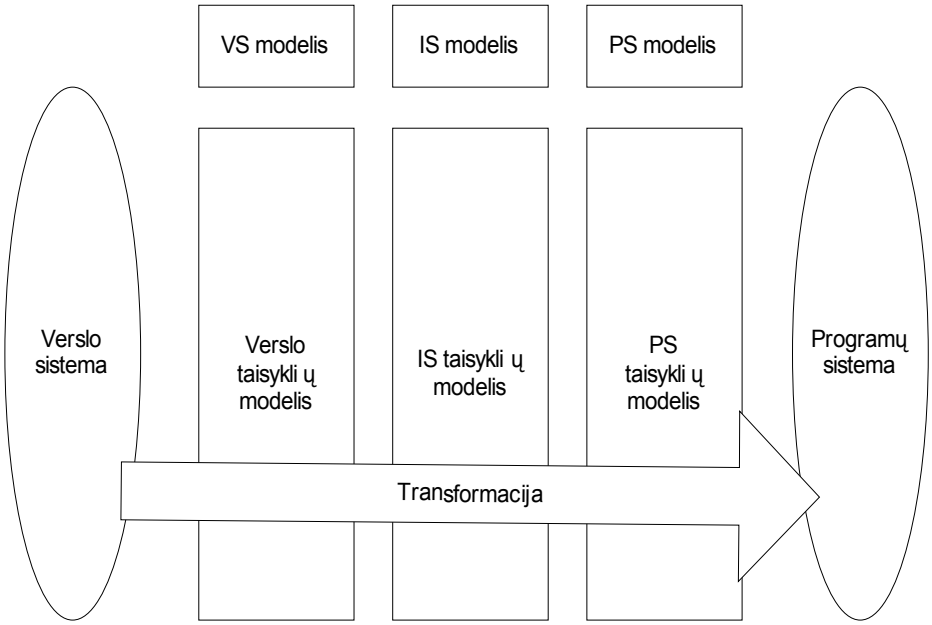
1.8 Informacijos apdorojimo taisyklės

Verslo taisyklės samprata skiriasi verslo ir informacinės sistemos lygmenyse. Verslo modelyje specifikuotos verslo taisyklės, kurios yra susijusios su informacijos apdorojimu ir realizuojamos IS, yra transformuojamos į informacijos apdorojimo taisykles.

Verslo taisyklės labiau tinkamos verslo lygmenyje, kalbant apie verslą, verslo procesus, verslo modeliavimą. IS lygmenyje verslo taisyklės apibrėžimas nebetinkamas, kadangi IS naudojamos taisyklės tiesiogiai susijusios su informacijos apdorojimu, o ne taisykles aprašančias verslo procesų apribojimus. Tokias taisykles derėtų vadinti informacijos apdorojimo taisyklėmis (angl. *information processing rule*).

1.9 Taisyklių transformacija

Inžinerijos procese, einant nuosekliai iš verslo sistemos modelio į informacinės sistemos modelį ir programų sistemos modelį, verslo taisyklės yra nuolatos modifikuojamos atliekant įvairias transformacijas, arba panaudojamos tam tikriems modelio artefaktams sukurti ir modifikuoti.



2 pav. ***Verslo taisyklių transformacija per sistemų lygmenis***

Verslo taisyklės gali būti transformuojamos į įvairius darinius (artefaktus) skirtinguose modeliuose ir gali būti panaudojamos:

- VT transformuojamos į kitas verslo taisyklių vaizdavimo kalbas;
- VT transformuojamos į "žmonių" kalbą;
- VT transformuojamos į predikatų logikos sakinais užrašytą formatą;
- VT transformuojamos į informacinės sistemos taisykles/informacijos apdorojimo taisykles.
- VT rinkiniai transformuojami į objektų modelio darinius.
- VT rinkiniai transformuojami į duomenų modelio darinius

Taip pat reikia pastebėti, kad verslo lygmenyje užfiksuota verslo taisyklė IS modelyje gali būti įgyvendinta viena ar keliomis IS taisyklėmis, kurios vadinamos informacijos apdorojimo taisyklės.

1.10 Verslo taisyklių repozitorius

Dažnai kyla klausimas - kur padėti verslo taisykles, kad jas būtų galima bet kada nesunkiai surasti, o susietą programinį kodą bet kada panaudoti verslo procesams vykdyti ar aprašyti?

Verslo taisyklių saugykla yra labai svarbi kuriant IS grindžiama VT požiūriu. Todėl svarbu suformuluoti reikiamus VT saugyklos reikalavimus, tai nagrinėjama [7, 11, 13] straipsniuose.

Verslo taisyklių repozitoriaus paskirtis yra palaikyti verslo taisykles visų suinteresuotų asmenų (angl. *stakeholders*) informacinius poreikius verslo taisyklėmis grindžiamame IS kūrimo požiūryje.

Straipsnyje [11] išskiriami specialūs verslo taisyklių repozitoriaus tikslai:

- palaikyti su viso verslo ir suinteresuotų asmenų su taisyklėmis susijusiu reikalavimus
- Sugebėjimas užklausti ir raportuoti poveikio analizei, atsekamumui ir verslo taisyklės pakartotiniam panaudojimui apimant ir internetines technologijas
- Verslo taisyklės ir su taisykle susijusios informacijos patikimumas ir integralumas

Svarbūs yra verslo taisyklių repozitoriaus reikalavimai. Reikia, kad VT saugykloje būtų išsaugomos verslo taisyklės, su VT susijusios informacijos apdorojimo taisyklės, programų sistemoms taisyklės, taip pat reikia stebėti taisyklų pasikeitimą, taisyklės versiją.

K. Kapočius ir M. Butleris straipsnyje [13] suformuluoja verslo taisyklių repozitoriaus subsistemas:

1. Duomenų šaltinių subsistema (angl. *Data sources subsystem*).
2. Funkcijų ir verslo sprendimų subsistema (angl. Fu*nctions and business decisions susbsytem*).
3. Konceptuali duomenų modelio subsistema (angl. *Conceptual data model subsystem*).
4. Nestruktūrinių VT šablonų subsistema (angl. *Non-structural BR templates subsystem*).
5. Nestruktūrinių VT subsistema (angl. *Non-structural BR subsystem*).

Autoriai išskiria verslo taisyklių šablonų subsistemą, kurioje saugomos kiekvienos kategorijos taisyklės (termas, faktas ir pan.) raktiniai žodžiai ar struktūrinis šablonas, kuris yra išskirstytas į dalis ir saugomas repozitoriuje. Toks repozitorius tiktų verslo taisyklių, užrašytų XML kalba, saugojimui, kadangi galima saugoti taisyklės šablona kaip DTD dokumentą arba XSD schemą, o verslo taisyklių subsitemos būtų saugomos pačios verslo taisyklės užrašytos XML kalba.

VT Saugykla turi leisti:

- aprašyti (vaizduoti) verslo taisykles, panaudojant šablonus;
- kurti šablonus taisyklių vaizdavimui, apibrėžiant taisyklių struktūrą;
- tikrinti (testuoti) pavaizduotas taisykles;
- susieti pavaizduotą taisyklę su joje panaudotais kitų modelių objektais;
- pateikti taisyklę verslo vartotojui suprantama kalba (arba grafiniu pavidalu).

VT saugykloje išsaugotos verslo taisyklės turi būti kaip nors įvykdomos. Vienas iš būdų apie kurį užsimenama [13] straipsnyje. Saugomos VT vykdomos verslo taisyklių variklio (angl. *business rules engine*), kuris susideda iš trijų pagrindinių komponentų: įvykių užfiksavimas, taisyklės tipo interpretavimas, specialių simbolių interpretavimas. VT vykdomos atsiradus įvykiui, kuris gali būti generuojamas vartotojo ar duomenų bazės. Specifinis taisyklės užklausimas yra tada generuojamas taisyklių variklio. Turi būti nurodyti, kad daug duomenų

bazės ir vartotojo įvykių sukelia paieška tinkamos taisyklės, net jeigu neįmanoma surasti tinkamos taisyklės.

Kiekvienai verslo taisyklėmis grindžiamai informacinei sistemai reikalingas verslo taisyklių repozitorius. Tačiau verslo taisyklių saugyklos reikalavimų sukūrimas atskira problema, kuri darbe plačiau bus nenagrinėjama.

2 SPRENDIMŲ PARAMOS SISTEMŲ IR INFORMACIJOS ANALIZĖS TAISYKLIŲ PANAUDOJIMO ANALIZĖ

Vienas iš VT grindžiamo kūrimo galimas panaudojimas būtų, VT panaudojimas duomenų analizei, kuri nagrinėjamas straipsniuose [17, 34, 35]. Straipsnyje [33] O. Vasilecas ir A. Šmaižys aprašo VT panaudojimą informacijos pateikimui ir duomenų analizei ir suformuluoja duomenų analizės uždavinius:

- Duomenų atitikimas verslo reikalavimams (verslo politikai),
- Duomenų apdorojimas ir analizė informacijai gauti,
- Iš apdorotų duomenų gautos informacijos atitikimas verslo reikalavimams (verslo politikai),
- Informacijos gavimas siekiant verslo sistemos objekto kontrolės,
- Informacijos gavimas sprendimų priėmimui,
- Informacijos gavimas ir apdorojimas žinių generavimui, siekiant pagerinti verslo procesus.

Šiuolaikinis verslas naudoja didžiulį duomenų kiekį, tačiau įmonėje šie duomenys gali likti nereikalinga našta, jeigu duomenys nebus analizuojami arba nesugebama jų išanalizuoti ir tinkamai interpretuoti. Todėl vis daugiau įmonių naudoja sprendimo paramos arba rengimo sistemas, o didėjantis duomenų kiekis verčia naudoti OLAP duomenų kubus. Šiame skyriuje panagrinėsiu sprendimų paramos sistemas, duomenų saugyklas ir OLAP sistemas ir galima verslo taisyklių panaudojimą juose.

2.1 Sprendimų paramos sistemos

Sprendimų paramos sistemos (angl. *Decision Support Systems*) yra kompiuterinės sistemos, kurios yra naudojamos padėti sprendimų priėmėjui spręsti pusiau struktūrizuotas ir nestruktūrizuotas problemas.

Sprendimų paramos sistemos (SPS) iš kitų IS išsiskiria lanksčiomis situacijos modeliavimo galimybėmis, būtinų duomenų išrinkimu, duomenų, žinių ir modelių integravimu, svarbiausių rezultatų sprendimų priėmimui išskyrimo savybėmis.

1Pagrindiniai SPS tikslai [14]:

1. Pagerinti sprendimo kokybę: SPS pagalba pagerinamas nestruktūrizuotų, nesuprogramuojamų sprendimų efektyvumas, taip padidinant pozityvų sprendimo rezultatą. Koncentruotos ir išskirtos analitinės informacijos dėka sprendimas atliekamas objektyviau, logiškiau ir greičiau.

2. Įžvalgumas ir mokymasis: SPS leidžia sprendimų priėmėjui eksperimentuoti su įvairiomis strategijomis ir alternatyvomis, modeliuojant ir projektuojant įvairias sprendimų pasekmes prognozuojančias aplinkybes (jautrumo analizė: „kas būtų, jei“, tikslų paieškos metodas). Intelektualių metodų taikymas ir praėjusių sprendimų bei jų pasekmių sekimas „apmoko“ SPS tolesnių sprendimų priėmimui.

M. Demarest pateikia sprendimo priėmimo prototipinį modelį [10]:

- sprendėjas (angl. *decision-maker*)
- aibė įvedamos informacijos sprendimo priėmimo procesui (duomenys, skaitiniai arba kokybiniai modeliai interpretuoti duomenims, istorinė patirtis su panašiomis duomenų aibėmis arba panašios sprendimo priėmimo situacijos, ir įvairūs dalykai kultūrinės ar psichologinės normos ir konstrukcijos asocijuotos su sprendimų priėmimu).
- sprendimo priėmimo procesas
- aibė išvedamos informacijos iš sprendimo priėmimo proceso (apima sprendimus ir aibę kriterijų sprendimo evoliucionavimui sukurtų iki aibės poreikių, problemos ar tikslai, kurie būdingi sprendimo priėmimui)

Tame pat straipsnyje pateikiamas komercinis sprendimo priėmimo procesas [10]:

- sprendimo parengimas (angl. *decision preparation*), kuris yra inicijuotas reakcijos į verslo būsenos pasikeitimą ir per kurią pavieniai ar kartu sprendėjai surenka reikalingą informaciją sprendimo struktūrizavimui, evoliucionavimui ir priėmimui.
- sprendimo struktūrizavimas (angl. *decision structuring*), kurio metu sprendimas yra suformuluojamas (angl. *framed*), modeliuojamas ir nagrinėjamas, kad, nustatyti reikšmę, priklausomybę ir sąsają su kitais sprendimais.
- konteksto parengimas ir ištyrimas (angl. *context development and exploration*), kurio metu scenarijai, kiekvienas, kuris galbūt suteikia atsakymą į tam tikrą klausimą, yra plėtojami ir išplečiami.
- sprendimo priėmimas (angl. *decision making*), kurio metu vienas ar keli scenarijai yra parenkami kaip problemos optimalūs sprendimai.
- sprendimo propagavimas (angl. *decision propagation*), kurio metu pasirinkti scenarijai yra paskelbiami ar išplatinami visose organizacijos srityse.

- sprendimo valdymas (angl. *decision management*), kurio metu konkretaus sprendimo įgyvendinimas yra stebimas ir sprendimo efektas verslo būsenai yra susijęs su sprendimo tikslais.

Pagal orientaciją į objektą išskiriamos į [18]:

1• Komunikacijos valdymo (grupės valdymo) SPS (angl. *comunication-driven, group-driven*) skirtos daugiau nei vieno žmogaus tikslams, užduotims suderinti, komunikuoti tarpusavyje, koordinuoti atskirų grupių veiklą.

2• Duomenų SPS (angl. *data-driven*) realizuoja priėjimą ir panaudojimą prie istorinių arba laiko eilučių duomenų organizacijos vidiniuose arba išoriniuose resursuose.

3• Žinių valdymo SPS (angl. *knowledge-driven*) - jos pateikia žinias apie tam tikrą sritį, padeda suvokti tos srities problemas. Čia integruojamos ekspertinės sistemos.

4• Dokumentų SPS (angl. *document-driven*) orientuotos į dokumentų panaudojimą veiklos sprendimų priėmime.

5• Modelių valdymo SPS (angl. *model-driven*) akcentuoja priėjimą prie modelio ir jo valdymą. Paprastos statistinės ir analitinės priemonės pateikia pačio paprasčiausio lygio funkcionalumą. Apskritai modelio valdymo SPS sprendimų priėmimui naudoja kompleksinius- finansinius, modeliavimo ir/arba optimizavimo- modelius. Modelio valdymo SPS naudoja sprendimų priėmėjų pateiktus duomenis ir parametrus. Pagal juos analizuoja situaciją ir padeda priimti sprendimus.

Sprendimų paramos sistemos tipinė struktūra [14]:

Duomenų valdymo subsistema. Duomenys gali būti asmeninėje sprendimų priėmėjo duomenų bazėje, duomenų bazėje, kuri sukurta specialiai SPS arba pasiekiami tiesiogine sąsaja su kita organizacijos ar išorine duomenų baze. Duomenų valdymo sistema – tai programinė įranga arba teisės, kurių pagalba pasiekiamas duomenų bazės turinys. Duomenų bazė turi užklausų atlikimo galimybę.

Modelių valdymo subsistema. Modelių bazė – problemos sprendimo modelių bazė, kurioje realizuotas programinis funkcijų, apletų, pagalbinių priemonių rinkinys. Modelių kataloge matomi visų modelių pritaikymas, tikslai, reikalavimai, versijos ir kt. Modelių bazės valdymo sistema padeda vartotojui pasirinkti ir pritaikyti norimą modelį, pasiima svarbiausius arba reikalingiausius duomenis iš duomenų bazės pasirinkto modelio vykdymui.

Žinių bazės subsistema - dirbtiniu intelektu paremtų taikomųjų paketų integravimas SPS architektūroje: padeda vartotojui pasirinkti adekvatų modelį/ius problemai spręsti, į

matematinius modelius integruoja neapibrėžtumo ir euristikos principus, padeda vartotojui interpretuoti gautus rezultatus bei išvadas.

Vartotojo sąsaja ir dialogo valdymo subsistema realizuoja sprendimo priėmėjo ir SPS sąveiką ir pateikia rezultatus bei išvadas įvairių formatų (tekstiniu, skaitiniu, grafiniu, animuotu ir kt.) pavidalu.

Sprendėjas – pagrindinis sprendimo priėmimo proceso komponentas, pagal SPS suteiktas žinias ir informaciją priimantis sprendimą.

2.2 Duomenų saugyklos

Duomenų saugykla (angl. *Data warehouse*) yra koncepcija, kurios populiarumas didėja, kadangi yra poreikis kokybiškiems duomenims. Duomenų saugojimas (angl. *warehousing*) yra kūrinys iš specializuotos duomenų bazės, kuri padeda palaikyti sprendimų priėmimą. Kadangi problemos asocijuojamos su reliacinėmis duomenų bazėmis, dabartinis sprendimas yra sukurti specializuotą duomenų bazę, kuri ištraukia duomenis iš įprastos duomenų bazės ir saugo duomenis vienoje didelėje duomenų bazėje arba serveryje. Ši duomenų saugykla paprastai naudojama organizacijos (angl. *enterprise*) duomenims iš skirtingų gamybos sistemų ir yra aproksimacija (priartėjimas) visos organizacijos duomenų.

Duomenų saugyklos suteikia galimybę daug paprasčiau rasti reikiamą informaciją duomenų struktūroje, skirtoje žinioms išgauti, pagerina sprendimų paramą, sumažina informacijos gavimo sąnaudas, leidžia tiksliau identifikuoti įmonės tikslus ir pan. DS paskirtis iš esmės ir yra šių galimybių įgyvendinimas per strateginius įmonės tikslus.

Duomenų saugykla yra paprastai periodiškai atnaujinama paketiniu apdirbimu (angl. *batch processing*) ir todėl yra labiau statinis arba istorinis modelis organizacijos duomenų. Vieną kartą sukurti duomenų saugykloje duomenys yra iš esmės statiniai ir nesikeičia iki kito duomenų saugyklos atnaujinimo. DS yra sukuriama patenkinti poreikius sprendimų priėmėjams ir turi skirtingą struktūrą ir vaizdavimą, kuris yra labiau intuityvus ir reaguojantis į valdymo užklausas.

DS duomenys dažniausia yra modeliuojami kaip daugiamačiai (angl. *multidimensional*). Labiausiai atitinkantis daugiamačių duomenų modelis yra suteikiamas OLAP aplikacijų. OLAP įrankiai suteikia aplinka sprendimų sudarymui ir verslo veiklos modeliavimui per ad-hoc užklausų palaikymą. Yra du keliai įgyvendinti daugiamati duomenų modelį [16]:

- Naudojant reliacinę architektūrą pseudo-daugiamačiam modeliui projektuoti.
- Naudojant tikras daugiamates duomenų struktūras, tokius kaip masyvai.

DS paprastai kuriama kaip centrinė duomenų saugykla, kuri suteikia duomenims formatą, kuris yra suprantamas vartotojams. DS suteikia tradicinį, gerai valdoma duomenų centrą

sprendimų paramos sistemoms, tai sąlygoja, kad duomenys yra greitai prieinami vartotojams. DS yra ne tik kopija kitų sistemų duomenų, bet unikali, turtinga duomenų aibė, kuri yra optimizuota sprendimų priėmimui.

Sprendimų paramai ir verslo analizei reikalingas platus ir nuodugnus supratimas verslo esybių, užduočių, taisyklių ir aplinkos. Verslo metaduomenų tikslas suteikti šį supratimą.

Straipsnyje [16] N.L. Sarda "Structuring Business Metadata in Data Warehouse Systems for Effective Business Support" nagrinėja problemas susijusias su verslo metaduomenų struktūrizavimu, tai gali suteikti kontekstą verslo valdymui ir sprendimų paramai, kai integruojama į duomenų saugyklą, Straipsnyje aprašomas objektinis verslo metaduomenų modelis ir nustatomas ryšis su techniniais metaduomenimis ir duomenų saugykla.

Šiandien didelėms organizacijoms reikalingas lankstus priėjimas prie įvairios rūšies informacijos, kuri dabar yra naudojama sistemoje. Duomenų saugyklos technologija palengvina sukūrimą integruotų ir dalykinių istorinių duomenų, ir suteikia lankstų priėjimą, kaupimą ir vizualizavimą informacijos duomenų saugykloje. Kad vartotojas žinotų prasmę ir struktūrą pasiekiamų duomenų, reikia sukurti „metaduomenų saugyklas“ (angl. *metadata repositories*), kurios turi išsamų duomenų aprašymą. Metaduomenys turi būti plačiai suklasifikuoti į verslo ir techninius metaduomenis. Verslo metaduomenys suteikia atitinkamą verslo kontekstą supratimui, duomenų analizei ir sprendimų palaikymui [14].

2.3 Aktyvios duomenų saugyklos

Duomenų bazių sistemos pagrindinis komponentas yra OLTP (*On-Line Transaction Processing*) sistema, kuri palaiko naudojamus verslo procesus. Paprastai, duomenys skirti vienam verslo procesui yra saugomi skirtingose specialios paskirties OLTP duomenų bazėse (pvz. pardavimai, sandėliavimas).

Pagrindinis komponentas OLAP sistemos yra duomenų saugykla (angl. *Data Warehouse*), kuri yra sprendimų palaikymo duomenų bazė, kuri periodiškai atsinaujina dėl ištraukimo, transformavimo ir pakrovimo naudojamų duomenų iš keleto OLTP duomenų bazių.

Duomenų saugyklos ištraukia (angl. *extracts*), transformuoja (angl. *transforms*) ir pakrauna (angl. *loads*) duomenis iš OLTP sistemos automatiškai be vartotojo įsikišimo.

T. Thalhammer ir kt. straipsnyje [30] "Active Data Warehouses: Complementing OLAP with Active Rules" aprašo naują architektūrą – aktyvios duomenų saugyklos (angl. *active data warehouse*). Aktyvios duomenų saugyklos siūlo galimybę automatizuoti sprendimų priėmimą šabloniškoms sprendimų užduotims ir pusiau-šabloniškoms sprendimo užduotims. Aktyvi duomenų saugykla gali tiesiogiai arba po vartotojo patvirtinimo eksportuoti sprendimus atgal į

OLTP sistemą. Tokios sistemos realizuoja uždaros-kilpos (angl. *closed-loop*) sprendimo priėjimą, kurį pavadino aktyvios duomenų saugyklos ciklu (angl. *active data warehouse cycle*).

Analizės ir sprendimų priėmimo automatizavimui panaudojama įvykis/sąlyga/veiksmas (angl. *event/condition/action*, ECA) taisyklių idėja iš aktyvių duomenų bazių sistemų. Įvykis/sąlyga/veiksmas taisyklės sėkmingai įdiegtos OLTP duomenų bazėse automatizuoti periodinėms užduotims, kurios naudojasi verslo procesais ar verslo taisyklėmis ir padaro tokias sistemas labiau autonomiškas, o verslo taisykles yra lengviau aptarnauti. Kartą sukurta įvykis/sąlyga/veiksmas taisyklė gali būti vykdoma automatiškai be vartotojo įsikišimo ar aplikacijos užklausos.

2.4 Tiesioginis analitinis apdorojimas

Analitinio apdorojimo tinkle arba tiesioginis analitinis apdorojimas (angl. *On-Line Analytical Processing*, OLAP) – tai programinės įrangos technologija, leidžianti greitai ir efektyviai peržiūrėti ir analizuoti didelius informacijos kiekius, atlikti prognozavimą. Duomenų atvaizdavimui naudojamas daugiamatis duomenų modelis, atspindintis realų kompanijos vaizdą, lengviau suprantamą vartotojui [28].

Daugiamačiam duomenų modelio atvaizdavimui, OLAP sistemose naudojami daugiamačiai duomenų kubai. Daugiamatės sistemos koordinačių ašimis yra pagrindiniai analizuojamo verslo proceso atributai. Pavyzdžiui kalbant apie prekybą, šiomis koordinačių ašimis gali būti prekė, regionas, laikas. Dimensija skaidoma į hierarchijas, taip atvaizduojamas detalumo lygis. Pavyzdžiui laiko dimensija suskyla į metų, ketvirčio, mėnesio ir dienos dimensijas. Dimensijos reikšmės vadinamos dimensijos nariais. Dimensijos nariai nurodo duomenų elemento padėtį dimensijoje. Dimensijų narių susikirtime saugoma informacija, vadinama kintamaisiais (angl. *variables*).

OLAP sistemos leidžia duomenis agreguoti ir detalizuoti (angl. *roll-up* ir *drill-down*).

ROLAP architektūra – tai daugiamatė reliacinių duomenų bazių lentelių sąsaja. Ši architektūra turi 2 privalumus: ji gali būti lengvai integruota į kitas egzistuojančias reliacines informacines sistemas ir reliaciniai duomenys gali būti saugomi efektyviau nei daugiamačiai duomenys. ROLAP architektūroje duomenys organizuojami dviem duomenų modeliais: žvaigždės (angl. star) ir snaigės (angl. *snowflake*).

Naudojant MOLAP architektūrą, duomenys saugomi daugiamačiame masyve. Kiekviena masyvo dimensija atitinka konkrečią OLAP kubo dimensiją. Fizinis vaizdas santykinai atitinka loginį OLAP kubo vaizdą, nes daugiamačiame masyve saugomos kintamųjų reikšmės.

Naudojant MOLAP architektūrą daugiamačiame masyve saugomi ne patys detaliausi duomenys, o iki tam tikro lygio agreguoti duomenys.

HOLAP architektūra apjungia ROLAP ir MOLAP savybes. Šioje architektūroje naudojamos ir reliacinės duomenų bazės ir daugiamačiai masyvai. Jei reikalingi ypatingai detalūs duomenys, tuomet HOLPA sistema formuoja užklausą reliacinei duomenų bazei, kai tuo tarpu pastovūs ir agreguoti duomenys saugomi daugiamačiuose masyvuose.

Duomenų kubas yra loginis modelis. Šį modelį fiziškai skirtingai organizuojant galima gauti du pagrindinius OLAP kubo saugojimo fizinius modelius: ROLAP architektūra (Relational OLAP) ir MOLAP architektūra (Multidimensional OLAP). Sujungus ROLAP ir MOLAP architektūrų savybes gauta HOLAP architektūra. [12]

G. Bukhbinder ir kiti straipsnyje [4] pateikia keturis standartus sėkmingai OLAP sistemai:

- Savalaikiškumas (angl. *timeliness*): užklausos turi būti apdorojamos greitai – sekundėmis arba minutėmis, ne valandomis ar dienomis
- Suprantamumas (angl. *understandability*): ataskaitos turi būti aiškios.
- Lengvai naudojama: analitikas turi būti pajėgus lengvai sukurti ataskaitas (angl. *reports*), be programavimo.
- Priėjimas (angl. *access*): sistema turi būti lengvai pasiekiama iš skirtingų vietų.

Sutinku su G. Bukhbinder ir kt. nuomone, kad sistema turi būti savalaikiška, suprantama, lengvai naudojama. Šios sąlygos tinka ne tik OLAP, bet ir bet kokia kitai sistemai.

2.5 Daugiamatės duomenų analizės kalbos

2.5.1 Egzistuojančios daugiamatės analizės kalbos

Nors OLAP yra puiki technologija, bet pats OLAP API neturi universalaus standarto. Reliacinėse duomenų bazės SQL (*Structured Query Language*) ir ODBC (*Open Database Connectivity*) yra universalios pripažintos ir plėtojamos kalbos. Šiuo metu rinkoje nėra dominuojančio OLAP produkto, kad galėtų laimėti rinkoje labiausia paplitęs produktas. Todėl yra sukurta daug aplikacijos programavimo interfeisų (angl. *application programming interfaces,* API), tokių kaip duomenų apibrėžimo kalbos (angl. *data definition languages*, DDL) arba manipuliavimo duomenimis kalbos (angl. *data manipulation languages*, DML).

Pagrindiniai OLAP programavimo interfeisai:

- Hyperion Essbase Report Writer API

Hyperion Report Writer API yra tekstu pagrįsta ataskaitų aprašymo API, kuri suteikia geriausią finansinių ataskaitų ir konsolidacijos funkcionalumą. Ši API yra viena iš

dominuojančių API OLAP erdvėje. Tai leidžia vartotojams išrinkti duomenis, modifikuoti schemą ir formuoti rezultatų aibę.

- Microsoft MultiDimensional Expression kalba

Microsoft'o MultiDimensional Expression (MDX) kalba yra OLAP užklausų kalba sukurta vietoj Microsoft OLEDB MD, multidimensinių duomenų bazių praplėtimas iki Mircrosft naujo OLEDB priėjimo sluoksnio. MDX formatas ir struktūra panaši į SQL. Naudojama FROM sakinys duomenų šaltinio specifikavimui, WHERE sakinys duomenų filtravimui ir SELECT sakinys specifikuoti ar apskaičiuoti members rezultatų aibėje.

Palyginus su SQL reliacinėse lentelėse, naudoti MDX multidimensinėse duomenų bazėse yra sudėtingiau ir sunkiau, negu kurti užklausas su SQL. SQL užklausos išrenka duomenis dviejų dimensijų formate, kai MDX atrenka daugiamačius duomenis, kurie leidžia verslo analitikui vykdyti užduotis, tokias kaip „roll-up“ ir „drill down“, taip pat „slice“ ir „dice“ duomenis pagal verslo reikalavimus. Išskiriami trys lygiai MDX panaudojimo: bazinių duomenų atrinkimas, 1-2 dimensijų rezultatas, ir sudėtingas duomenų atrinkimas 2 dimensinėje kubo struktūroje

- XML/A

XML for Analysis yra kildinama iš Microsoft’o OLE DB OLAP'ui, dėl ko pasirodo daug panašumų. Pagrindinis skirtumas yra tas, kad XMLA skirtas Web servisam ir nepriklauso nuo platformos. Taip pat nėra tvirtai susietas su kokiu nors klientu ar serveriu.

- MDAPI

OLAP Council siūlo MDAPI, kuris yra suprojektuotas palaikyti įvairiarūšių duomenų apdorojimo aplinką, kuri išsiskiria nuo Microsot'o OLAP API. Tai suteikia galimybę klintui ir serveriui komunikuoti dėka standartinių technologijų tokių kaip DCOM, CORA ir Java. API suteikia Java bibliotekas ir COM objektus ir gali būti integruotos su programavimo kalbomis, tokiomis kaip C++ ir Java. Be to, dalykinių programų įrankį, tokį kaip Visual Basic ir Java Script galinti palaikyti API. Deja nėra realaus palaikymo šio API.

- JOLAP

JOLAP – Java grindžiama kelių organizacijų OLAP API iniciatyva ir yra grindžiam Java® API, skirta J2EE platformai. Tai įgalina manipuliuoti OLAP duomenimis ir metaduomenimis nepriklausomai nuo programinės įrangos gamintojo metodo. Tai liečia dvi svarbias misijas: universalios OLAP užklausos ir OLAP duomenų ir metaduomenų valdymas. Tikslas pakeisti Microsoft’o OLE DB skirto OLAP ir XML for Analysis, taip pat OLAP Council’o sukurto ir pasenusio MDAPI ir vietoj jų sukurti universalią OLAP užklausų API. JOLAP išplečia bazines funkcijas, su planuotu palaikymu- kūrimo, saugojimo, priėjimo ir priežiūros duomenų ir metaduomenų OLAP serveriuose ir daugiadimensinėse duomenų bazėse.

2.5.2 MultiDimensional Expression (MDX)

MDX kalba išreiškianti atrinkimus, skaičiavimus ir šiek tiek metaduomenų apibrėžimus iš OLAP duomenų bazės ir suteikia sugebėjimą specifikuoti, kai užklausos rezultatai bus vaizduojami. Skirtingai nuo kitų OLAP kalbų tai, nėra pilnai ataskaitų formavimo kalba. MDX užklausos rezultatai grąžinami kliento programai kaip duomenų struktūros, kurios turi būti apdorojamos kokiu nors būdu kaip lentelė, grafikas arba kita išvedimo forma. Tai panašu į SQL veikimą reliacinėse duomenų bazėse ir kaip funkcionuoja dalykinės programos programavimo interfeisas (angl. *application programming interface,* API). API, kuris palaiko MDX, apimantis Object Linking and Embedding Data Base for Online Analytical Processing (OLE DB for OLAP) ADO MD, ADOMD.Net, XMLA (*XML for Analysis*), Hyperion Essbase C ir Java APIs, ir Hyperion ADM API.

MDX konstrukcijos gali būti panaudotos realizuoti VT skirtas duomenų analizei. Tai detaliau nagrinėjama kitame skyriuje.

2.5.2.1 MDX užklausos struktūra

Apžvelgsime paprastas MDX užklausas. Įsivaizduokite labai paprasta kubą, su tokiomis dimensijomis: laiko, pardavimų geografija ir matavimais. Kubą pavadinsime Sales (pardavimai). Norime peržiūrėti Masačiutseso pardavimus ir kainas pirmų dviejų 2005 metų ketvirčių. MDX užklausos rezultatas lentelėje, kurios ląstelėse turi duomenų reikšmes. Tinklelis (angl. *grid*) gali turėti dvi dimensijas, kaip ir skaičiuoklė arba lentelė, bet taip pat gali turėti viena, tris ar daugiau. Žemiau pateiktas pavyzdys ir rezultatas:

```
SELECT
{ [Measures].[Dollar Sales], [Measures].[Unit Sales] }on columns,
{ [Time].[Q1, 2005], [Time].[Q2, 2005] } on rows
FROM [Sales]
WHERE ([Customer].[MA])
```

Užklausoje SELECT, FROM, ir WHERE yra ranktiniai žodžiai, kurie nurodo skirtingas užklausos dalis. Rezultatas MDX užklausos yra pats tinklelis, iš esmės kitas kubas.

Ši užklausa grąžina eilutes ir stulpelius. Pagal MDX terminija, axis yra padėtis arba dimensija (angl. dimensijon) užklausos rezultato. Axis naudoti geriau už dimensiją, taip pasidaro paprasčiau atskirti nuo kubo dimensijų nuo rezultato dimensijų. Be to kiekviena axis gali turėti daug kubo dimensijų. Užklausą padaliname į dalis:

1. SELECT raktažodis pradeda sakinį, kuris apibrėžia ką jūs norite išrinkti.
2. ON raktažodis yra naudojamas su ašies (angl. *axis*) pavadinimu specifikavimui, kurios dimensijos iš duomenų bazės yra vaizduojamos. Šiame pavyzdyje užklausa atvaizduoja matavimus ant stulpelio ašies ir laiko periodus ant eilutės ašies.

3. MDX naudoja raitytus skliaustelius { ir }. Jie skirti apsupti aibę elementų iš skirtingų dimensijų arba dimensijų aibių. Pateikta paprasta užklausa turi tik vieną dimensiją kiekvienai ašiai (matavimų dimensija (angl. *measures dimension*) ir laiko dimensija. Skirtingus elementus galime atskirti kableliais (,). Elementų vardus galima atskirti laužtiniais skliaustais ([ ]) ir galima daugialypes dalis atskirti taškais (.).
4. MDX užklausoje, galima specifikuoti kaip dimensijos iš duomenų bazės išsidėstys ant ašių norimame gauti rezultate. Pateikta užklausa išdėsto matavimus ant stulpelio ašies ir laiko periodus ant eilutės. Kiekviena užklausa gali turėti skirtingą skaičių rezultatinių ašių. Pirmosios trys ašys turi tokius vardus stulpeliai (angl. *columns*), eilutės (angl. *rows*) ir puslapiai (angl. pages). Šie pavadinimai ir spausdinami raportuose.
5. FROM sakinys MDX užklausoje įvardina kubą iš kurio duomenys yra užklausiami. FROM sakinys yra panašus į FROM sakinį SQL kalboje.
6. WHERE sakinys suteikia prieigą specifikuoti narius kitoms kubo dimensijoms, kurios neatsiranda stulpeliuose arba eilutėse (arba kitose ašyse). Jei nėra specifikuojamas narys kažkokiai dimensijai, tada MDX naudos narį pagal nutylėjimą. Naudojamas WHERE yra neprivalomas.

Kai tik duomenų bazė nustato užklausos grąžinamą rezultatą, tada yra užpildomos lentelės celės duomenimis iš užklausto kubo. Dalis MDX raktinių žodžių SELECT, FROM, ir WHERE yra paimti iš SQL kalbos. Reikšmės ir semantika yra ganėtinai skirtinga ir bandant naudoti SQL konceptus MDX, labai dažnai galima susidurti su painiava. MDX yra labiau sukoncentruotas prie kiekvienos dimensijos, prie matavimų dimensijų. Rezultato, kuris parodytų pardavimus trim ketvirčiam (pavaizduota stulpeliuose) iš dviejų valstijų (eilutės), gavimui sukuriame tokią užklausą:

```
SELECT
{ [Time].[Q1, 2005], [Time].[Q2, 2005], [Time].[Q3, 2005] } on columns,
{ [Customer].[MA], [Customer].[CT] }on rows
FROM Sales
    WHERE ( [Measures].[Dollar Sales] )
```

2.6 MS SQL Server 2005™ Analysis Services

Microsoft™ SQL Server 2005 Analysis Services (SSAS) suteikia OLAP ir duomenų gavybos funkcionalumą „verslo intelektualumo“ (angl. *business intelligence*) dalykinėm programom. Analysis Services leidžia naudotis OLAP projektavimui, kūrimui ir pasinaudojimui multidimensinėmis struktūromis, kurios turi duomenis, susidedančius iš kitų duomenų šaltinių, tokių kaip reliacinės duomenų bazės. OLAP kubui sukurti patogiausia pasinaudoti vedliu (angl.

Wizard), bet vedlys nesuteikia galimybės manipuliuoti kodu, todėl leidžiama tiesiogiai redaguoti kubo XML failą. XML failo redagavimas leistų, kubo kūrimui panaudoti taisyklės.

Analysis Services veiksmai gali susieti bet kokius struktūrinius ar nenuoseklius (angl. *unstructured*) duomenis arba komandas su beveik bet kokia OLAP kubo dalimi. Jei norite savo organizacijoje padidinti sprendimų kokybę ir savalaikiškumą, naudokite veiksmus susijusios informacijos gavimui intuityviu keliu gauti.

Veiksmus galima susieti su kubo celėm (angl. *cells*), aibėm (angl. *sets*), dimensijų lygiais (angl. *dimension levels*), dimensijomis arba su visu kubu.

Veiksmų (angl. *action*) tipai:

- *Command line:* įvykdo komandinę eilutę. Su tuo, galima paleisti iš esmės bet kokią programą su komandinės eilutės parametrais, programos incializavimui su tiesiogiai susijusia informacija. Pavyzdžiui komandinė eilutė veiksmo produkto vardo gali iškviesti inventoriaus programą, kuri parodo esamą inventorių ir rūšiuoja esamos padėties situacija.
- *Statement:* Įvykdo OLE DB komandą, kuri taip pat gali pavykti arba nepavykti, bet negrąžina rezultato. Pavyzdys duomenų apibrėžimo komandos, tokios kaip CREATE MEMBER ar DROP MEMBER.
- *HTML:* Apibrėžia HTML atidarymą Interneto naršyklėje. Kliento dalykinė programa išsaugo HTML laikiname faile ir paleidžia pagal nutylėjimą nustatytą interneto naršyklę, HTML failo parodymui .
- *URL:* Panašus į HTML veiksmą, išskyrus tai, kad aprašo URL adresą vietoj HTML kodo. URL veiksmas yra naudojamas susieti kubo metaduomenis su internetoar intraneto puslapiais.
- *Dataset:* Turi MDX konstrukcija, kuri grąžina duomenų aibę. Duomenų aibė gali būti iš kito tos pačios dumenų bazės kubo. Šis veiksmo tipas yra naudojamas susieti vartotojus su skirtingais, bet susijusiais kubais.
- *Rowset:* Turi OLE DB komandą kaip sakinys ir duomenų aibės tipas, bet ši komandą grąžina eilučių aibę (angl. *rowset*). Eilučių aibės veiksmas gali apimti „drillthrough“ komandą arba standartinę SQL užklausą.
- *Proprietary:* Gali apimti bet ką, kas reikalinga. Jei naudojamas įprastam veiksmui, kurį analizės dalykinė programa gali atpažinti dėl specialios elgsenos, kuri netinka vienai iš daugelio kategorijų.

Veiksmų aprašymas:

- Vardas (angl. *Name*) – veiksmo pavadinimas
- Veiksom taiknys (angl. *Action Target*) – objektas, kuriam prisikiriamas veiksmas

- Sąlyga. MDX išraiška, kuri grąžina loginę reikšmę. Jei true veiksmas vykdomas.
- Veiksmo tipas.
- Papildomos savybės.

Galimos savybės:

- Invocation – apibrėžia, kada veiksmas yra vykdomas.
  - Batch
  - Interactive
  - On Open
- Application – apibūdina, veiksmo aplikacija
- Description – apibūdina veiksmą
- Caption – antraštė, kuri yra rodoma veiksmui.
- Caption is MDX – suteikti true jei reikia MDX antraštės

Iš išvardintų veiksmų aprašymo ir savybių, mano manymu, būtų galima SSAS veiksmus pritaikyti VT realizavimui programų sistemos lygmenyje. Veiksmai turi pagrindines dalis sąlygos ir veiksmo, reikalingos CA (condition-action) taisyklės realizavimui. Pavyzdžiui, jei pardavimai paskutinio mėnesio mažesni už kažkokią sumą, formuojamas HTML failas su reikiama informacija (kokia reikalinga informacija įvesti taip pat galima apibrėžti taisyklėmis), kad vartotojas jį persiųstų kitiems suinteresuotiems asmenims. Detaliau veiksmai ir su jais susijusios verslo taisyklės analizuojamos kitame skyriuje.

Ataskaitos veiksmas (angl. *Report Action*)

Report Server reguoja į URL užklausas skirtas ataskaitoms. Pasirinkimai apibrėžiami ataskaitų veiksmui:

- Report Server.

  Savybės:

Savybė	Aprašymas
Server name	Kompiuterio vardas, kuriame veikia Report Server
Server path	Kelias
Report format	HTML5, HTML3, Excel, DF.

- Parameters (Optional) . Parametrai, kurie yra siunčiami serveriui kaip dalis URL eilutės. Parametrai apima Parameter Name ir Parameter Value, kuri yra MDX išraiška.

Report server URL yra konstruojama:

```
http://host/virtualdirectory/Path&parametername1=parametervalue1& ...
Pvz:
http://localhost/ReportServer/Sales/YearlySalesByCategory?rs:Command=Render&R
egion=West
```

Drillthrough veiksmas

Drillthrough veiksmas yra apibrėžimas kaip rowset veiksmas, kuris yra grąžinamas kliento aplikacijai kaip drillthrough sakinys.

Opcijos:

Drillthrough Columns – viena ar daugiau dimensijų, ir kiekvienai dimensijai grąžinami kliento aplikacijai drillthrough stulpeliai.

Galima teigti, kad veiksmus siūloma naudoti daugiausia papildomos informacijos gavimui. Bet, kadangi galima aprašyti veiksmui sąlygą (angl. *condition*), t.y. veiksmas įvyks tik prie tam tikslių sąlygų, todėl, mano manymu, galima realizuoti sąlyga-veiksmas taisykles. Analysis Service veiksmai, mano manymu, būtų vienas iš būdų realizuoti verslo taisykles OLAP kube, duomenų analizei.

2.7 XML kalba ir jos panaudojimas dokumentų transformavimui

XSL (eXtensible Stylesheet Language) kalba buvo sukurta W3C (Wide Web Consortium) konsorciumo, kaip formatavimo šablonų (angl. *stylesheet*) kalba XML pagrindu.

CSS – HTML formatavimo šablonuose yra naudojamos iš anksto apibrėžtos žymės, kurių reikšmė yra iš anksto žinoma, todėl naudojant CSS stilius naršyklei yra visai paprasta pasakyti kokį šriftą ar jo dydį naudoti išvedant tam tikrus elementus (pvz. lentelės).

XSL – XML kalba užrašytuose formatavimo šablonuose žymės nėra iš anksto apibrėžtos, todėl ir naršyklei jos nėra iš anksto suprantamos. Tam kad žymės taptų suprantamos naudojama XSL.

Tačiau XSL yra daugiau nei formatavimo šablonų kalba, kurią sudaro sekančios dalys:

XSLT kalba, skirta XML dokumentų transformacijai,

Xpath kalba, skirta XML dokumento dalių apibrėžimui,

XSL-FO kalba, skirta XML dokumentų formatavimui.

XSL – yra kalbų aibė kurią galima transformuoti XML į XHTML, filtruoti (išrinkti) ir rūšiuoti XML duomenis, apibrėžti XML dokumento dalis, formatuoti duomenis pavaizduotus XML ir išvesti juos į skirtingus šaltinius – ekraną, popierių, balsą, …

XSLT nuo 1999 Lapkričio 16 tapo standartu (W3C Rekomendacija). Šią kalbą transformacijos procese naudoja Xpath, kuri apibrėžia šaltinio dokumento dalis, kurios atitinka vieną ar daugiau iš anksto apibrėžtų šablonų (Templates). Kai tinkama dalis yra surandama – ji yra transformuojama į rezultato dokumentą. Šaltinio dokumento dalys, kurios nėra apibrėžtos nėra pavaizduojamos rezultatų dokumente.

XML dokumentai ar duomenys gali būti transformuojami naudojant XSL arba XSLT schemas tiek serverio pusėje ir perduodami klientui (pvz.: HTML formatu) tiek ir kliento pusėje

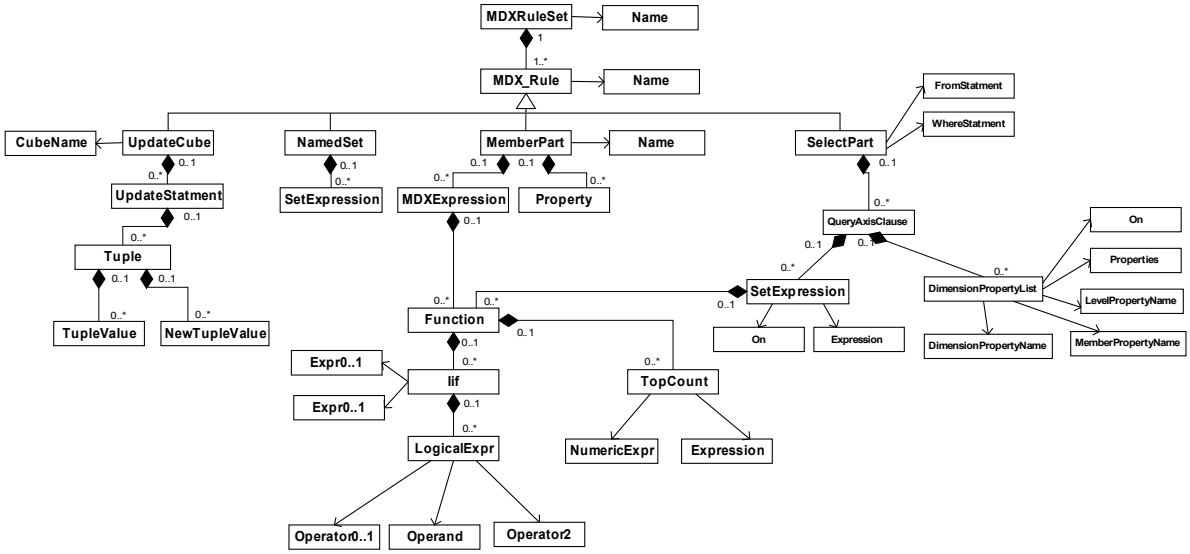
siunčiant jam patį XML dokumentą, kuriame yra nuoroda į atitinkamą XSL schemą. Pirmasis variantas yra labiau priimtinas, nes HTML supranta visos naršyklės, o su XML pavaizdavimu gali kilti problemų, ypač senesnėse naršyklių versijose.

3 METODO VERSLO TAISYKLŲ TRANSFORMAVIMUI Į DAUGIAMATĖS DUOMENŲ ANALIZĖS INSTRUKCIJAS FORMULAVIMAS

Šiame skyriuje suformuluojamas metodas, pagrįstas SRML ir MDX metamodelių transformacijomis, skirtas verslo taisyklės transformavimui į MDX instrukciją. 3.1 skyriuje pateikiamas SRML metamodelis ir 3.2 skyriuje pateikiamas siūlomas MDX kalbos metamodelis. Šie metamodeliai reikalingi tolimesniam metodo suformulavimui. Metodas aprašytas 3.3 skyriuje.

3.1 Daugiamatės duomenų analizės kalbos metamodelis

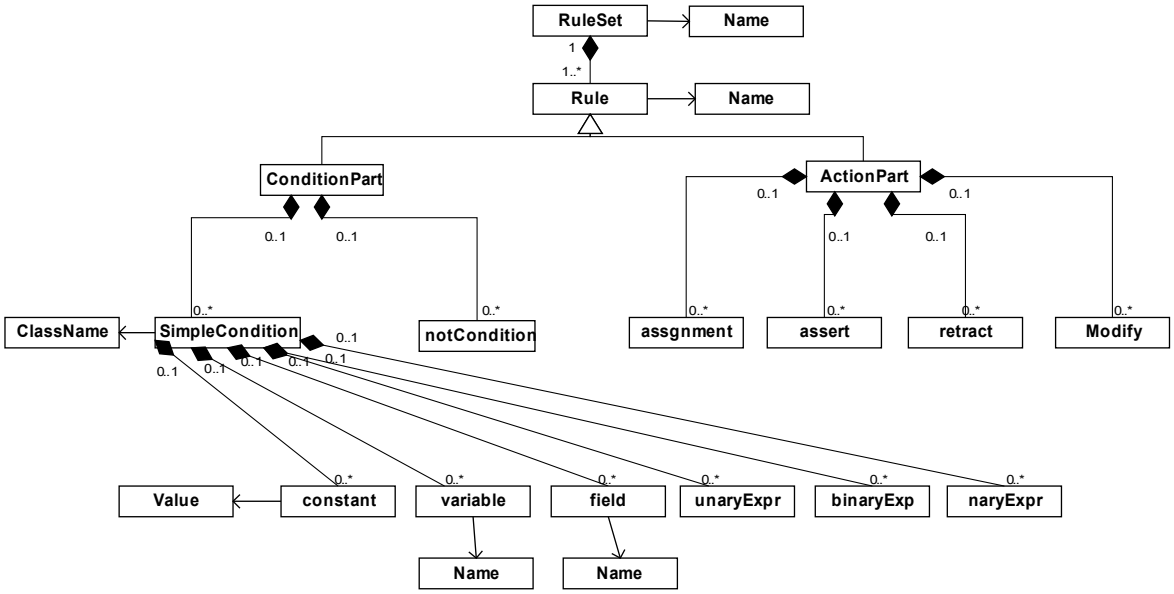
Verslo taisyklių transformacijoms atlikti reikalingas MDX metamodelis. Sukurtas MDX kalbos metamodelis pavaizduotas 3 pav. Šiame metamodelyje pavaizduotos tik tos dalys, kurios tinka verslo taisyklių realizavimui. Metamodelis realizuojamas XSD schemoje (4 priedas).



3 pav. ***MDX kalbos metamodelis***

3.2 SRML kalbos metamodelis

Iš SRML DTD dokumento sukuriama XSD schema. Šios SRML metamodelio dalis pavaizduotas 4 paveikslėlyje. Paveikslėlyje pavaizduota tik metamodelio dalis, elementus notcondition, assignement, assert, retract, modify taip pat sudaro tokie pat elementai, kaip ir SimpleCondition elementą, t.y. assignment elementas gali turėti constant, variable, field, unaryExpr, binaryExpr, naryExpr dalis.



4 pav. ***SRML kalbos metamodelis***

3.3 Metodo aprašymas

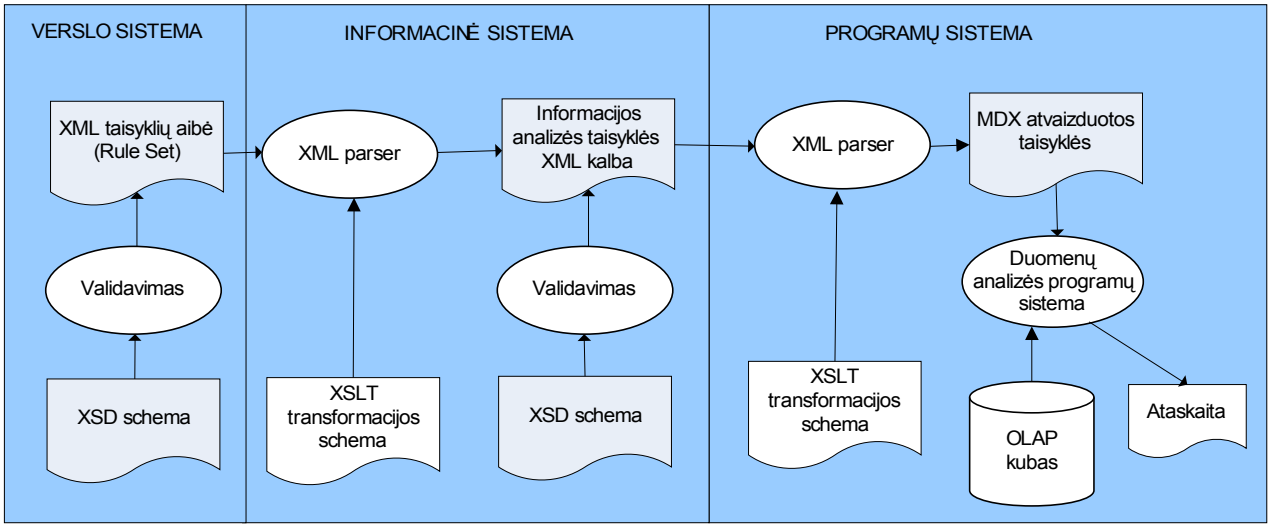
Inžinerijos procese, einant nuosekliai iš verslo sistemos modelio į IS modelį ir programų sistemos (PS) modelį, VT yra nuolatos modifikuojamos atliekant įvairias transformacijas arba panaudojamos tam tikriems modelio artefaktams sukurti ir modifikuoti. Taip pat reikia pastebėti, kad verslo lygmenyje užfiksuota verslo taisyklė, IS modelyje gali būti įgyvendinta viena ar keliomis IS taisyklėmis, kurios vadinamos informacijos apdorojimo taisyklėmis. Atitinkamai ir viena informacijos apdorojimo taisyklė IS modelyje gali būti įgyvendinta viena ar keliomis taisyklėmis (PS) modelyje.

Informacinės sistemos kūrimo metu verslo taisyklės užrašytos verslo sistemoje yra transformuojamos į informacijos apdorojimo taisykles informacinėje sistemoje, o šios dar kartą transformuojamos į programinį kodą ar programų sistemos objektus programų sistemoje. XML kalba yra labai lanksti transformacijų požiūriu. Transformacijoms aprašyti yra sukurta XSLT kalba, kuri leidžia XML dokumentą transformuoti praktiškai į bet kokį pavidalą.

A. Šmaižio darbe [7,25] parodyta, kad VT, užrašius XML ir panaudojus XSLT šablonus, galima transformuoti į daugelį informacinės sistemos ar net programų sistemos struktūrinių darinių ar pavidalų.

Metodas skirtas VT užrašytos XML pavidalu transformavimui į MDX konstrukcijas. VT laikome taisyklė kuri skirta informacijos analizei.

Pirmiausia reikia taisykles suklasifikuoti. Taisyklės, užrašomos XML kalba ir validuojamos, pagal DTD dokumentą arba XSD schemą. Kiekvienai taisyklių klasei sukuriama XSLT transformaciją.



5 pav. ***VT transformavimo schema***

Paveikslėlyje (5 pav. VT transformavimo schema) pateiktas VT transformavimo metodas per tris sistemos lygmenis (VS, IS, PS) iki MDX instrukcijos. Pateiktame metode siūloma suklasifikuoti VT ir jas užrašyti XML kalba. Kiekvienai taisyklių grupei turi būti sukuriama XSD schema arba DTD dokumentas. XSD schema arba DTD dokumentas skirti validuoti taisyklėms, t.y. turi būti patikrinama, ar taisyklė atitinka tai taisyklių klasei skirtą DTD dokumentą ar XSD schemą. Kitame žingsnyje VT turi būti transformuojamos į informacijos analizės taisyklės (IS lygmuo). IS lygmenyje taisyklės turi būti susijusios su informacijos apdorojimu. Šiame lygmenyje informacijos analizės taisyklė taip pat užrašoma XML kalba. Užrašyta taisyklė taip pat turi būti validuojama XSD schema arba DTD dokumentu. Verslo taisyklės iš verslo sistemos lygmens į informacijos analizės taisyklę IS lygmenyje transformavimui sukuriama XSLT transformavimo schema.[20]

Kitame žingsnyje informacijos analizės taisyklė (IS lygmuo) transformuojama į MDX instrukciją (PS lygmuo). Transformacijai tai pat sukuriama XSLT transformavimo schema, kuri skirta transformuoti IS lygmens informacijos analizės taisyklę į MDX instrukciją. Gauta MDX instrukcija įvykdomą duomenų analizės programų sistemoje, kuri kreipiasi į MS SQL Server 2005 Analysis Services ir įvykdo užklausą ir grąžiną rezultatą. Kadangi MDX kalboje naudojami verslo objektų pavadinimai, pavyzdžiui dimensijos (angl. *dimensions*) - laikas, pardavimai, produktas, ir pan., taip pat dimensijų lygiai (laiko dimensijos lygiai – sausis, vasaris, kovas ir t.t.) yra verslo objektai, tai supaprastina verslo taisyklės transformaciją į MDX instrukciją. MDX taip pat turi didelį rinkinį funkcijų, kurios taip pat gali būti panaudotos verslo taisyklėse (pvz. vidutiniai produkto pardavimai – MDX avg() funkcija). [20]

Toks taisyklių įgyvendinimas leidžia atskirti taisykles nuo programos kodo ir lengviau jas pakeisti, pasikeitus verslo sistemos būsenai. Metodas leidžia surinkti ir užrašyti XML kalba VT susijusias su duomenų analizę. VT rinkinį galima taikyti pagal konkrečią verslo sistemos būsena

(pvz. kai pardavimų rodikliai geri reikalingi vienokie duomenys, kai pardavimų rodikliai blogi, reikalingi detalesni duomenys). Toks sugrupavimas leidžia sudaryti taisyklių rinkinį ir panaudoti jį intelektualiai duomenų analizei, pagal verslo sistemos būseną bus sukuriamos skirtingos ataskaitos.

Svarbu yra suformuoti tokį verslo taisyklių rinkinį, kurį galima būtų pritaikyti konkrečiai verslo sistemos būsenai. Bet reikia pastebėti, kad yra beveik neįmanoma surinkti visų VT susijusių su duomenų analize, nes verslo taisyklės nuolat kinta veikiamos tiek vidinių, tiek išorinių veiksnių, galiausia tų pačių duomenų analizės rezultatų, todėl svarbu yra įgyvendinti tik pagrindines VT.

4 METODO EKSPERIMENTINIS PRITAIKYMAS

šiame skyriuje atliksiu eksperimentus, kuriais parodysiu, kad VT galima transformuoti į SQL sakinius ir MDX kalbos instrukcijas. Šiame skyriuje aprašomų eksperimentų metu siekiama patikrinti aukščiau aprašytą metodą.

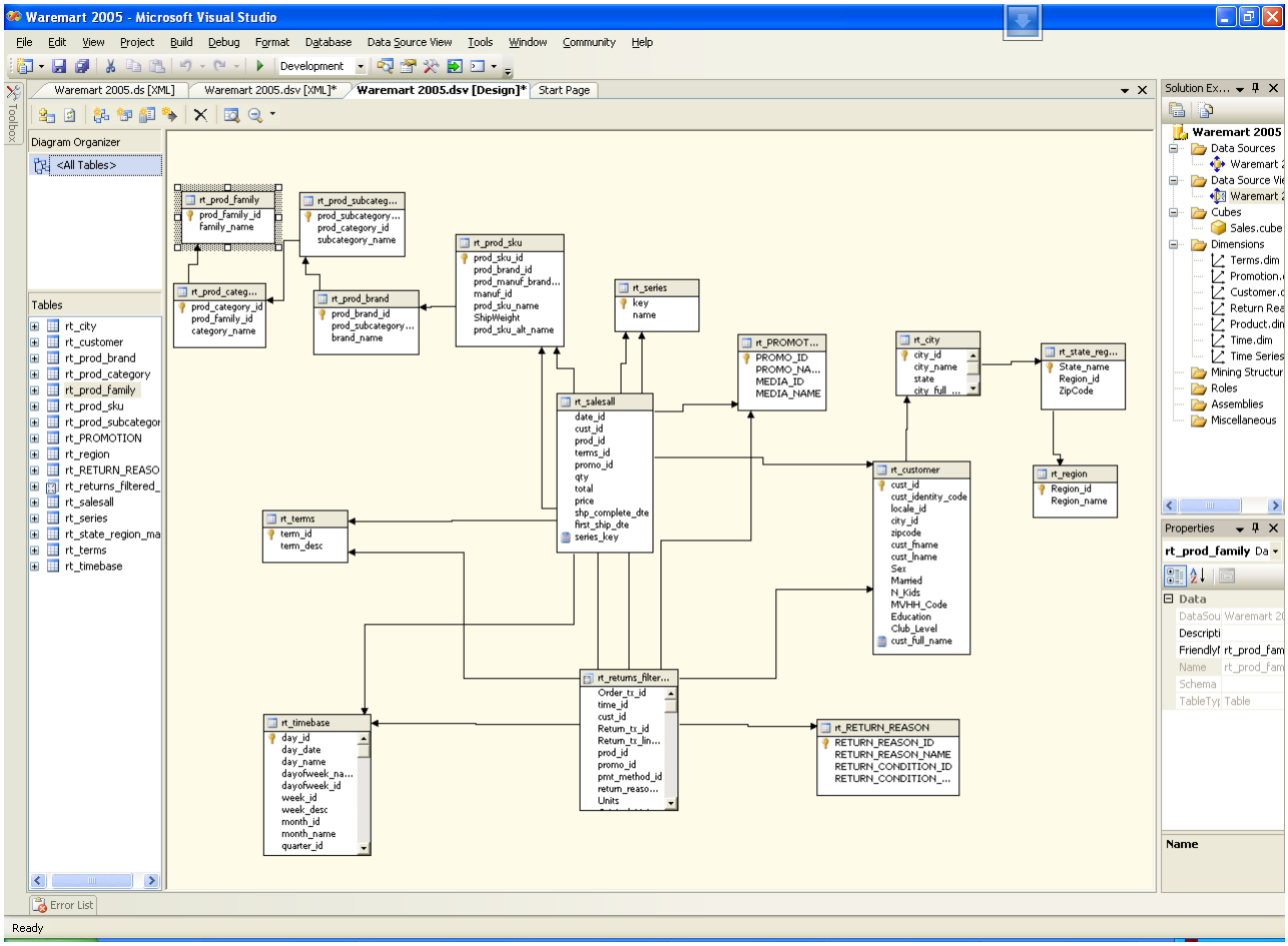
4.1 Eksperimentams naudoti įrankiai ir priemonės

Eksperimentams atlikti pasinaudosiu Waremart 2005 duomenų baze, kuri naudojama G. Spofford ir kitų „MDX Solutions: with Microsoft SQL Server Analysis Services 2005 and HyperionEssbase“ knygoje [22].

Lentelė pavadinimas	**Atributų sk.**	**Įrašų skaičius lentelėje**
rt_city	3	168
rt_customer	13	16497
rt_manufacturer	2	696
rt_prod_brand	3	1200
rt_prod_category	3	25
rt_prod_family	2	4
rt_prod_sku	6	7250
rt_prod_subcategory	3	150
rt_PROMOTION	4	40
rt_region	2	8
rt_RETURN_REASON	3	6
rt_return1998	12	6355
rt_return1999	12	7604
rt_return2000	12	707
rt_returnsall	12	14032
rt_salesall	10	165949
rt_series	2	1
rt_state_region_map	3	48
rt_terms	2	7
rt_timebase	17	731
rt_tx1998	15	74906
rt_tx1999	15	91499
rt_tx2000	15	5354
rt_utility	6	6
rt_WAREHOUSE	3	12

*1 Lentelė **Waremart2005 duomenų bazės įrašų skaičius***

Norint atlikti teisingus eksperimentus su OLAP technologija reikalingas pakankamai didelis duomenų kiekis. 1 lentelėje pateiktos duomenų bazę sudarančios lentelės ir duomenų (įrašų) kiekis lentelėse, iš šios lentelės matome, kad duomenų bazėje yra pakankamas duomenų kiekis atlikti eksperimentus su OLAP kubu.



6 pav. ***MS SQL Server Business Intelligence Development Studio. Kubo "Sales" projektavimo vaizdas***

Parinktame OLAP kube yra pakankamas kiekis duomenų, kad eksperimentai būtu tikslingi. Taip pat OLAP kubo struktūra yra pakankamai sudėtinga, tai leidžia aprašyti sudėtingas taisykles, pvz. subkubų arba paprastesnių lokalių kubų sukūrimui.

SQL užklausas vykdysiu Northwind duomenų bazėje, kuria testavimui siūlo Microsoft

Altova XMLSpy programa bus naudojama verslo taisyklės užrašymui XML kalba ir validavimui pagal atitinkama XSD schemą ar DTD dokumentą.

Altova MapForce programa naudojama XSLT transformavimo schemos, kuri skirta XML transformavimui į kitokios struktūros XML failą, sukūrimui.

Altova StyleVision programa naudojama XSLT transformacijos schemos, kuri skirta XML failo į tekstinį dokumentą, sukūrimui.

SQL ir MDX užklausų įvykdymui naudojama Microsoft SQL Server 2005 Enterprise Evaluation Edition, kurį galima parsisiųsti iš Microsoft svetainės ir naudotis 180 dienų.

Visus eksperimentų metus sukurtus XML ir XSLT failus galima naudoti nepriklausomai nuo programinio produkto, nes sukurti artefaktai parašyti standartine XML kalba.

4.2 Eksperimentas

Šiame skyriuje patikrinsiu ar hipotezė, kad XML kalba pavaizduotą taisyklę galima transformuoti į SQL konstrukciją, panaudojant XSLT transformacijos schemas yra teisinga.

4.2.1 Eksperimento aprašas

Eksperimento uždavinys: Taisyklę iš VS lygmens transformuoti į SQL instrukciją.

Eksperimento planas:

1. Užrašyti verslo taisyklę XML kalba ir validuoti pagal SRML XSD dokumentą.
2. Sukurti XSLT transformacijos schemą, taisyklės transformavimui į SQL sakinį.
3. Pagal sukurta XSLT schemą generuojamas SQL kalbos sakinys.

Verslo taisyklė VS lygmenyje apibrėžiame taip:

Jei pagal 1998.05.06 užsakymo datą suma, gauta produkto vieneto kainą padauginus iš parduoto produkto kiekio buvo mažesne už 5000000, tada reikia suformuoti atskaitą, kuria sudarytų produkto pavadinimas, vieneto kaina, kiekis ir užsakymo data.

If sum, by order date 1998.05.06, of product unit price multiply by Quantity is < 5000000, then give report, which contains Product Name, product Unit price, Quantity and Order Date.

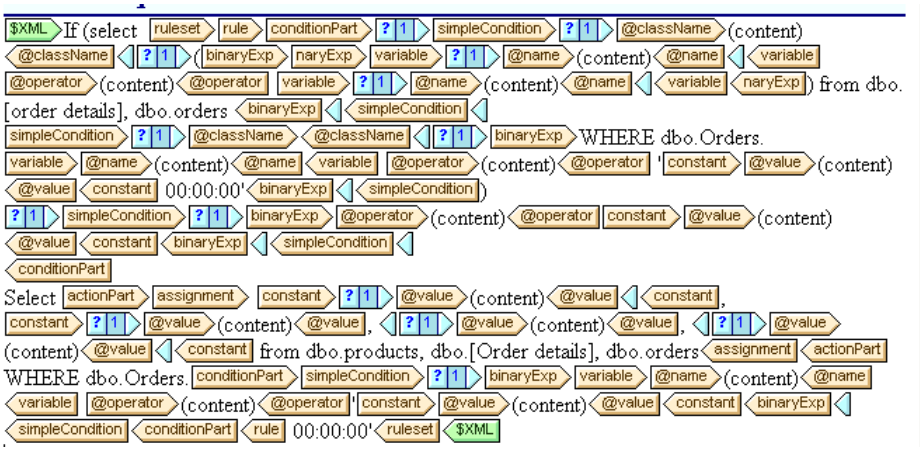
4.2.2 Eksperimento eiga

Programos Altova XMLSpy pagalba užrašome pateikta taisyklę XML kalba ir validuojama pagal SRML XSD schemą. Žemiau pateiktas XML kodas.

```
<?xml version="1.0" encoding="UTF-8"?>
<ruleset xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="srml.xsd">
<rule name="SalesReport">
<conditionPart>
<simpleCondition className="Sum">
<binaryExp operator="&lt;">
<naryExp operator="*">
<variable name="Quantity"/>
<variable name="UnitPrice"/>
</naryExp>
<constant type="double" value="5000000" />
</binaryExp>
</simpleCondition>
<simpleCondition className="Date">
<binaryExp operator="=">
<variable name="OrderDate"/>
<constant type="string" value="1998.05.06" />
</binaryExp></simpleCondition></conditionPart>
<actionPart>
```

```
<assignment>
<constant type="string" value="productname"/>
<constant type="string" value="Unitprice"/>
<constant type="string" value="Quantity"/>
<constant type="string" value="orderdate"/>
</assignment></actionPart></rule></ruleset>
```

Programos Altova StyleVision pagalba XML dokumentui sukuriama XSLT transformacijos schema (7 priedas) skirta transformuoti XML kalba užrašyta taiyklę į SQL konstrukciją.



7 pav. ***Altova StyleVision XSLT transformacijos kūrimas***

```
Sukurūs XSLT transformacijos schemą transformuoju XML kalba užrašyta VT į SQL
konstrukciją.
If (select  Sum(Quantity * UnitPrice) from dbo.[order details], dbo.orders
WHERE dbo.Orders.OrderDate = '1998.05.06 00:00:00')<5000000
Select  productname, Unitprice, Quantity, orderdate from dbo.products,
dbo.[Order details], dbo.orders
WHERE dbo.Orders.OrderDate ='1998.05.06 00:00:00'
```

Gauta SQL užklausa įvykdau MS SQL Server 2005 aplinkoje. Užklausa vykdoma Northwind duomenų bazei ir gaunu rezultatą, kuris pavaizduotas žemiau.

	ProductName	UnitPrice	Quantity	OrderDate
1	Chai	14,00	12	1998-05-06 00:00:00
2	Chai	9,80	10	1998-05-06 00:00:00
3	Chai	34,80	5	1998-05-06 00:00:00
4	Chai	18,60	9	1998-05-06 00:00:00
5	Chai	42,40	40	1998-05-06 00:00:00
6	Chai	7,70	10	1998-05-06 00:00:00
7	Chai	42,40	35	1998-05-06 00:00:00
8	Chai	16,80	15	1998-05-06 00:00:00

8 pav. ***Užklausos grąžinamas rezultatas***

4.2.3 Eksperimento rezultatai

- XML kalba užrašyta taisyklę galime transformuoti į SQL konstrukciją.
- Sugeneruota XSLT schema, pateiktos taisyklės transformavimui į SQL instrukciją.

- Nors eksperimente ir naudojama Altova® programinė įranga, tačiau sukuriami artefaktai yra parašyta standartine XML kalba, todėl rezultatai nėra priklausomi nuo programinio produkto.

4.3 Eksperimentas

Atliksiu eksperimentą pagal metodą aprašyta aukščiau. Šiame metode VT transformuojama per tris sistemos lygmenis (VS, IS, PS).

4.3.1 Eksperimento aprašas

Eksperimento uždavinys: Verslo taisyklę iš VS lygmens transformuoti į MDX instrukciją, panaudojant aukščiau aprašytą metodą.

Eksperimento planas:

1. Užrašyti verslo taisyklę XML kalba ir validuoti pagal SRML XSD dokumentą
2. Sukurti XSD dokumentą taisyklei užrašyti IS lygmenyje ir transformuoti ją iš VS lygmens į IS lygmens taisyklę.
3. Sukurti XSLT transformacijos schemą, VT iš VS lygmens transformavimui informacijos apdorojimo taisyklę IS lygmenyje.
4. Sukurti XSLT schemą taisyklės transformavimui į MDX instrukciją.
5. Sugeneruoti pasinaudojus sukurta XSLT schema MDX instrukcija ir įvykdyti.

Verslo taisyklių šaltinis

Kainų reguliavimas

Verslo taisyklė VS lygmenyje apibrėžiame taip:

If Product Category Unit sales less than 5000 units on 2005 quarter 3, then these sales is Bad sales.

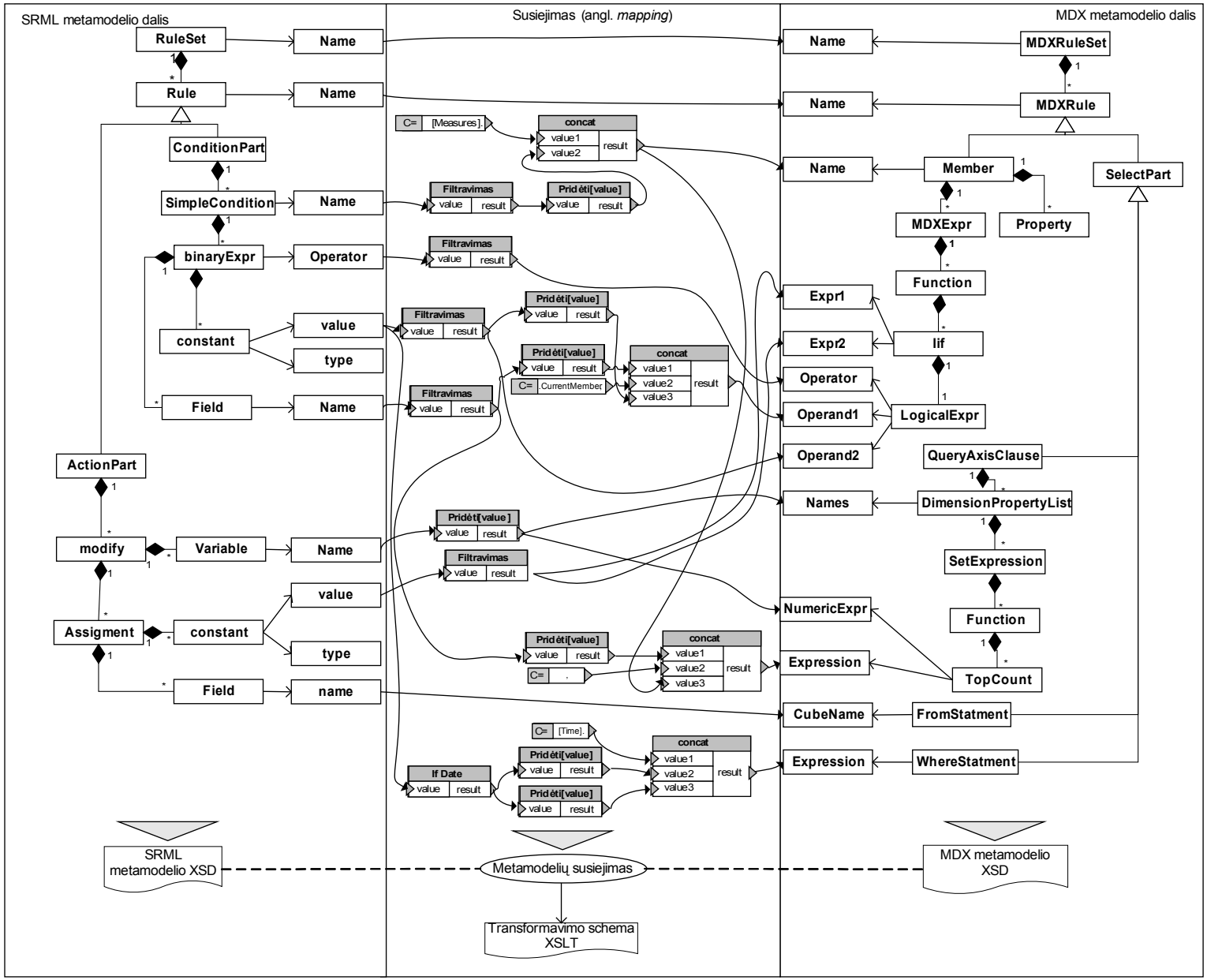
Jei produkto vieneto pardavimai 2005 metų trečiame ketvirtyje yra mažesni už 5000 vienetų, tada šie pardavimai yra blogi.

4.3.2 Eksperimento eiga

Su Altova™ XMLSpy užrašome taisyklę XML kalba ir validuojame ją pasinaudodami SRML kalbos XSD dokumentu (3 priedas).

```
<ruleset><rule name="SalesType">
<conditionPart>
<simpleCondition className="LowSales">
<binaryExp operator="lt"><constant type="string" value="Unit Sales" />
<constant type="int" value="5000"/>
</binaryExp>
</simpleCondition>
<notCondition className="productstype">
<field name="Product"/></notCondition>
<simpleCondition className="Date">
<binaryExp operator="eq">
<field name="year" >
<constant type="int" value="2005"/>
</binaryExp>
<binaryExp operator="eq">
<field name="quater" />
<constant type="int" value="3"/>
</binaryExp>
</simpleCondition>
</conditionPart>
<actionPart>
<modify>
<variable name="UnitSales"/><assignment>
<field name="sales"/>
<constant type="string" value="Bad sales"></constant>
</assignment>
</modify>
</actionPart>
</rule></ruleset>
```

Pateikta, XML kalba užrašyta taisyklė turi būti transformuojama į informacinės sistemos lygmens verslo taisyklę. Verslo taisyklės transformavimui sukuriame XSLT transformacijos schemą (5 priedas). Šį XSLT transformacijos schemą kuriama pasinaudojant metamodelių transformacijomis, kuri pateikta 4 paveikslėlyje. Transformacijoje naudojamos funkcijos: filtravimas, concat ir pan. Filtravimo funkcija naudojama norint atfiltruoti tam tikrus VT užrašytos XML kalba elementus. Filtravimo funkcija palygina einamą XML elemento poziciją su parinkta ir reikšmes sulygina ir grąžina reikiamą rezultatą. Funkcija prideti[value]: prie reikšmės (verslo objekto prideda laužtinius skliaustus (reikšmės pradžioje ir pabaigoje). Funkcija concat apjungia simbolių eilutės (angl. *string*) reikšmes. Funkcija „if date“ palygina ar nagrinėjamas XML elementas turi atributą date, jei toks atributas aptinkamas, tuomet atfiltruojamas atitinkamas XML elementas. Bendras šių funkcijų derinys leidžia atlikti verslo taisyklės transformaciją į IS lygmens taisyklę.



9 pav. ***VT transformavimas***

Kadangi OLAP kube objektų pavadinimai yra verslo objektai, tai kai kuriuos objektus galime transformuoti tiesiogiai: Product, Unit Sales, 2005, Bad sales. Kuriame XSLT transformacijos schema su Altova MapForce programa.

Panaudojant XSLT transformacijos schemą, aukščiau XML kalba užrašytą verslo taisyklę transformuojame į IS lygmens XML kalba užrašytą taisyklę, kuri validuojama pagal sukurtą XSD schemą. Pateikiama XML kalba užrašytos taisyklės ištrauka:

```
<rule name="SalesType">
<MemberPart name="[Measures].[LowSales]">
<MDXExpression>
<Function>
<iif Expression1="Bad sales" Expression2=" ">
<Logical_Expression operand1="[Product].[Category].CurrentMember, [Unit
Sales])" operator="&lt;" operand2="5000"/>
</iif>
</Function>
</MDXExpression>
</MemberPart>
<SelectPart>
<query_axis_clause>
<dimension_property_list_clause DimensionPropertyName="[Measures].[LowSales]"
LevelPropertyName="[Unit Sales]" ON="on columns"/>
<SetExpression Onx="on rows">
<Function>
<TopCount Set_Expression="[Product].[Category].Members" Count="25"
NumericExpression="[Unit Sales]"/>
</Function>
</SetExpression>
</query_axis_clause>
<FromStatment>[sales]</FromStatment>
<WhereStatment Expression="[Time].[quarter]" Param1="Q3" Param2="2005"/>
</SelectPart>
</rule>
</ruleset>
```

Pateikto XML dokumentas vaizduoja kaip pasikeitė taisyklės struktūra. Transformuojant prie objektų pavadinimų pridėti laužtinius skliaustus ([]), šiais skliaustais MDX kalboje apskliaudžiami verslo objektai. Taip pat Nario dalyje (member part) prie nario pavadinimo pridedamas [Measures] eilutė, kadangi nario (angl. *member*) pagalba sukuriamas naujas matavimas (angl. *measures*).

Informacinės sistemos lygmens XML kalba užrašytą taisyklę turime transformuoti į MDX instrukciją. Transformavimo schemai XSLT sukurti pasinaudojame Altova StyleVision™ programą. Altova StyleVision™ naudojama taisyklėms transformuoti iš XML į kitus failų tipus: tekstinius, HTML, RTF ir kitus. Šiuo atveju XML failą transformuosime į paprasta tekstinį failą. Sukurta XSLT schema pateikta 6 priede. MDX instrukcija atrodytų taip:

```
With Member [Measures].[LowSales] as
'iif (([Product].[Category].CurrentMember, [Unit Sales])<5000, "Bad sales",
"")'
Select [unit sales], [LowSales]} on columns,
topcount ( [Product].[Category].members, 50, [unit sales]) on rows
```

```
from [sales] where [Time].[Quarter].[Q3, 2005]
```

Žemiau pateiktas MDX isntrukcijos įvykdymo rezultatas programų sistemoje.

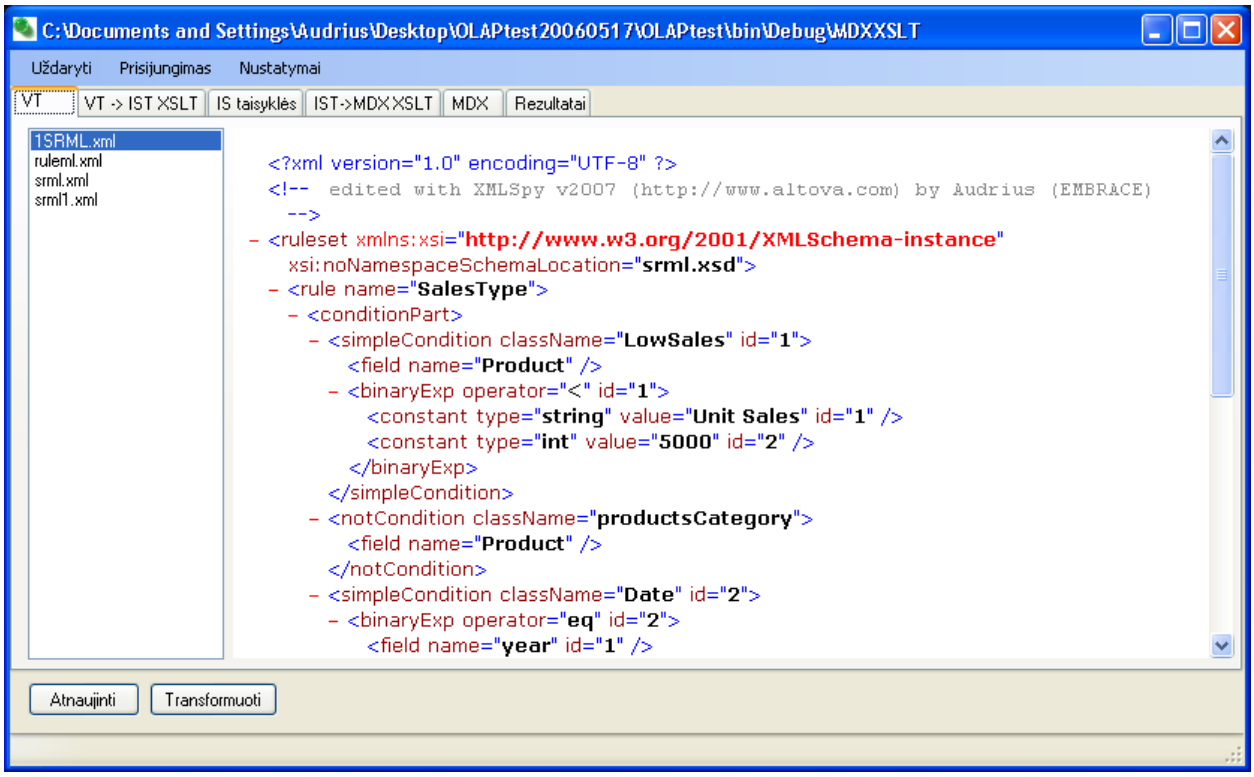
	Unit Sales	LowSales
Outdoor Gear	5,349	
Sports Equipment	5,054	
Exercise, Fitness	4,955	Bad sales
TV, DVD, Video	4,771	Bad sales
Bikes, Scooters	4,470	Bad sales
Indoor, Outdoor Games	4,310	Bad sales
Home Audio	3,996	Bad sales
Personal Care	3,869	Bad sales
Golf	3,605	Bad sales
Oral Care	3,561	Bad sales
Massagers, Spas	3,187	Bad sales

10 pav. ***Eksperimento metu gautos užklausos grąžinamas rezultatas***

4.3.3 Eksperimento rezultatai

- Eksperimento metu užrašyta VT verslo sistemos lygmenyje XML kalba, sukurta XML transformavimo schema, taisyklės transformavimui į IS lygmens taisyklės. IS lygmens taisyklės transformavimui į MDX instrukcija, taip pat sukurta XSLT schema. Eksperimentas patvirtina aprašytą metodą, kad verslo taisyklių poaibį skirtą informacijos apdorojimui ir užrašyta XML kalba galima transformuoti iki MDX instrukcijos.
- Verslo taisyklė transformuojama per tris sistemos lygmenis VS, IS ir PS.
- Eksperimento metu sukurtus artefaktus, kurie parašyti XML kalba, galima realizuoti programų sistemoje, palaikančioje XML, taip pat automatizuoti duomenų analizę.

4.4 Prototipo aprašas



11 pav. ***XML ir VT atvaizdavimas***

3 skyriuje aprašytam metodui realizuoti buvo sukurtas programos sistemos prototipas. Prototipas skirtas verslo taisyklių transformavimui iki MDX instrukcijos.

Prototipas parašytas C# kalba, naudojantis MS .NET Framewok 2 bibliotekomis. Prototipo veikimui reikalinga: MS SQL Server 2005 su Analysis Service serveriu, .NET Framewok 2 ir Microsoft Core XML Services (MSXML) 6.0.

Prototipui turi būti pateikiami tokie duomenys:

- XML failas, kuriame būtų XML kalba užrašytos verslo taisyklė arba verslo taisyklų aibė.
- XSLT transformacijos schemą, verslo taisyklės transformavimui iki IS lygmens taisyklės.
- XSLT failas, IS lygmens taisyklės transformavimui iki MDX instrukcijos.

Pateikus prototipui reikalingus duomenis (failus) ir paspaudus mygtuką transformuoti – VT bus transformuota į IS lygmens taisyklę ir MDX instrukciją. Šią MDX instrukciją galima įvykdyti, jei prieš tai buvo prisijungta prie MS SQL Server 2005 Analysis Services serverio (tai galima padaryt per Meniu punktą Prisijungti->prisijungti, kuriame nurodomas serverio adresas ir prisijungimo vardas ir slaptažodis) ir prisijungus prie serverio reikia parinkti norimą kubą (meniu punktas Prisijungti-> Kubo parinkimas, kuriame parenkama duomenų saugykla ir kubas). Įvykdytos MDX užklausos rezultatai parodomi lange „Rezultatai“. Programų sistemos prototipas sėkmingai išbandytas su antrame eksperimente sukurtais artefaktais, t.y. failas verslo taisykle

užrašyta XML kalba, failas su XSLT transformacija į IS lygmens taisyklę ir failas su XSLT transformacija į MDX instrukciją.

IŠVADOS IR SIŪLYMAI

Darbe išnagrinėtas verslo taisyklių taikymas intelektualiai duomenų analizei, kur verslo taisyklės užrašomos XML kalba ir transformuojamos iki programų sistemos objektų (SQL ir MDX konstrukcijų).

1. Atlikus analizę, paaiškėjo, kad daugelis verslo taisyklių yra neidentifikuojamos verslo taisykles. Tai ypatingai taikoma taisyklėms, kurios neturi aiškaus vaizdavimo.
2. Atlikus analizę, nustatyta, kad verslo taisyklės gali būti pavaizduotos XML kalba. Panaudojant XSLT transformavimo schemas VT gali būti transformuojamos į įvarius informacinės sistemos ir programų sistemos objektus.
3. Atlikus analizę, nustatyta, kad verslo taisyklės gali būti panaudotos intelektualiai duomenų analizei ir sprendimų priėmimui. Taisyklių panaudojimas galimas naudojant OLAP technologiją, duomenų saugyklas.
4. Atlikus analizę, paaiškėjo, kad duomenų saugyklos ir OLAP technologija suteikia galimybę daug paprasčiau rasti reikiamą informaciją duomenų struktūroje, skirtoje žinioms išgauti, pagerina sprendimų paramą, duomenų analizę, sumažina informacijos gavimo sąnaudas, leidžia tiksliau identifikuoti įmonės tikslus ir pan.
5. Suformuluotas taisyklės transformavimui į MDX instrukciją metodas, kuris pagrįstas metamodelių transformacijomis. Verslo taisyklės, užrašytos XML kalba, transformuojamos į MDX instrukcijas, naudojant XSLT transformavimo schemą.
6. Pasiūlytas metodas, leidžia intelektualizuoti ir automatizuoti duomenų analizę naudojant OLAP technologiją. Užrašius VT XML kalba ir transformuojant jas iki MDX instrukcijų automatizuojamas duomenų analizės procesas ir sprendimų priėmimas.

Tolimesni darbai: metodo tobulinimas, būtų pilno verslo taisyklių rinkinio suformavimas ir transformacijų schemų sukūrimas, pasiūlyto metodo realizavimui informacinėje sistemoje reikalingo VT repozitoriaus reikalavimų suformavimas ir jo sukūrimas.

TERMINŲ IR SANTRUMPŲ ŽODYNAS

ABR (Accessible Business Rules)	–	karkasas, kuris leidžia verslo įmonei kurti paskirstytas verslo aplikacijas
API (application programming interface)	–	dalykinės programos programavimo sąsaja
DB	–	duomenų bazė
DBVS	–	duomenų bazių valdymo sistema
DML (data manipulation language)	–	Duomenų manipuliavimo kalba
DS	–	Duomenų saugykla
DTD (Document Type Definition)	–	Dokumento tipo apibrėžimas.
IS	–	Informacinė sistema.
KIF (Knowledge Interchange format)	–	Apsikeitimo žiniomis formatas.
MDA	–	Model Driven Architecture
MDX (MultiDimensional Expression)	–	Multidimensinių išraiškų kalba
ODBC (Open Database Connectivity)	–	atvira API priėjimui prie duomenų bazės
OLAP (On-Line Analytical Processin)	-	programinės įrangos technologija, leidžianti greitai ir efektyviai peržiūrėti ir analizuoti didelius informacijos kiekius, atlikti prognozavimą. Duomenų atvaizdavimui naudojamas daugiamatis duomenų modelis, atspindintis realų kompanijos vaizdą, kuri yra lengviau suprantamas vartotojui. [32]
OLTP (On-Line Transaction Processing)	–	transakcijų realiuoju laiku apdorojimas
PIM (Platform Independent Model)	-	platformos nepriklausomas modelis
PS	–	Programų sistema
PSM (Platform Specific Model)	-	specifinis platformos modelis
RuleML (Rule Markup Language)	–	Taisyklių vaizdavimo žymėmis kalba.
SPS	–	Sprendimų paramos sistema
SQL (Structured Query Language)	–	struktūrinė užklausų kalba.
SRML (Simple Rule	–	Paprasta žymių kalba taisyklėms vaizduoti.

Markup Language)		
SSAS	–	SQL server analysis services
UML (Unified Modeling Language)	–	unifikuota modeliavimo kalba.
Verslo taisyklės (business rules)	–	tai visuma visų taisyklių, kurios reglamentuoja veiklos vykdymą (įstatymai, politika, vidaus tvarka ir t.t.). Šis taisyklių rinkinys apima statinius (struktūrinius) ir dinaminius (funkcinius, elgsenos) organizacijos veiklos aspektus.
VS	–	verslo sistema
VT	–	verslo taisyklė
XML (eXtensible Markup Language)	–	Išplečiama žymių kalba skirta dokumentams su struktūrizuota informacija.
XSD (W3C XML Schema)	–	XML schemos apibūdinimo kalba
XSLT (eXtensible Stylesheet Language Transformations)	–	Išplečiamos šablonų kalbos transformacija.

LITERATŪRA

1. Bajec M., Krisper M. Managing business rule in enterprises. *Elektroteniški vestnik*, *Electronical Review*, No. 68, Ljubljana, Slovenija, 2001, p. 236-241.
2. Bajec M., Krisper M. Issues and challanges in business rule-based Information systems development. *Proc. of 13th European Conference on Information Systems (ECIS 2005), Information systems in a rapidly changing economy*, Regensburg: Institute for Management of Information Systems, 2005, p. 1-12.
3. Bajec M., Krisper M. A Methodology and Tool Support for Managing Business Rules in Organisations. *Information Systems,* vol. 30, issue: 6, 2004, p. 423-443.
4. Bukhbinder G., Krumenaker M., Phillips A. Insurance Industry Decision Support: Data Marts. *Proc. of CAS 2005 Winter Forum*, *OLAP and Predictive Analytics,* 2005, p.171-195.
5. Business Rules Group, *Defining Business Rules: What are they Really?* 2002, URL: http://www.businessrulesgroup.org/brgdefn.htm [žiūrėta: 2006-09-21].
6. Business Rules Markup Language (BRML). Technology Reports. 2002.URL: http://xml.coverpages.org/brml.html [žiūrėta: 2006-04-20].
7. Butleris R., Kapočius K. The Business Rules Repository for Information Systems Design. *Proc. of 6th East-European Conference, Advances in Databases and Information Systems (ADBIS'2002), Research Communications,* Vol.2, Bratislava: STU, 2002, p. 64 – 77.
8. Butleris R.; Motiejūnas L. Meta duomenys veiklos taisyklėms su duomenų baze integruoti. *Informacinės technologijos 2005, Konferencijos pranešimų medžiaga*. Kaunas: Kauno technologijos universitetas, 2005, p. 534 –539.
9. Cawsey A. Knowledge Representation and Inference. 1994, URL: http://www.cee.hw.ac.uk/~alison/ai3notes/chapter2_4.html. 1994. [Žiūrėta: 2005-12-29].
10. Demarest M. Technology and Policy in Decision Support Systems. *DPA White Paper,* 1998, URL: http://www.noumenal.com/marc/techpoly.pdf [Žiūrėta: 2006-05-18].
11. Haggerty N., Wall J. Grace Van Etten. Defining The Requirements For a Business Rule Repository. *The Data Administration Newsletter*, 2001, URL: http://www.tdan.com/i016ht01.htm. [Žiūrėta: 2006-05-15].
12. Kalinauskienė L. OLAP sistemų projektavimas žinių modelio pagrindu. *Informacinės technologijos 2002: konferencijos pranešimų medžiaga*, Kaunas: Kauno technologijos universitetas, 2002, p. 146-150.

13. Kapočius K., Butleris R. Repository for Business Rules Based IS Requirements. *Informatica*, Vol. 17, No. 4, Institute of Mathematics and Informatics, Vilnius, p. 503-518.
14. Merkevičius E., Garšva G., Cepkovataja O. Intelektualios sprendimų paramos sistemos struktūra kredito rizikos vertinimui. *Informacinės technologijos 2005, Konferencijos pranešimų medžiaga*. Kaunas: Kauno technologijos universitetas, 2005, p. 725-732.
15. Morgan T. *Business Rules and Information Systems: Aligning IT with Business Goals*. Addison Wesley, 2002.
16. Nandlal L. Sarda. Structuring Business Metadata in Data Warehouse Systems for Effective Business Support, 2001, URL: http://arxiv.org/abs/cs.DB/0110020 [žiūrėta: 2006-10-20].
17. Pašilskytė I., Nemuraitė L. Verslo procesų modeliavimo kalbų analizė ir specifikavimo priemonių sukūrimas. *Informacinės technologijos 2005, Konferencijos pranešimų medžiaga*. Kaunas: Kauno technologijos universitetas, 2005, p. 525 – 533.
18. Paulraj Ponniah. *Data Warehousing Fundamentals: A Comprehensive Guide for IT Professionals,* John Wiley & Sons, 2001.
19. Power D. J. What is a DSS?. *The On-Line Executive Journal for Data-Intensive Decision Support,* Vol. 1, 1997, URL: http://dssresources.com, [žiūrėta: 2006-11-20].
20. Rima A., Šmaižys A., Vasilecas O. Intelektualizuota duomenų analizė verslo taisyklių transformacijų pagrindu. *Informacinės technologijos 2007*, *Konferencijos pranešimo medžiaga*. Kaunas: Kauno technologijos universitetas, 2007, p. 202-206.
21. Rima A., Šmaižys A., Vasilecas O. Verslo taisyklių panaudojimas duomenų analizės metamodelių transformacijų pagrindu. *10-osios Lietuvos jaunųjų mokslininkų konferencijos "Mokslas – Lietuvos ateitis" pranešimų rinkinys, Vilnius: Technika*, 2007, p. 2-8.
22. Rosca D., Wild C. Business rules in the real world: a decision support approach. *Software Engineering and Knowledge Engineering,* 1996, p. 121-128.
23. Ross R. G. *Principles of the Business Rule Approach.* Addison-Wesley. 2003.
24. Rouvellou I., DeGanaro L., Chan H. ir kt. Combining Different Business Rules Technologies: A Rationalization. *Proc. of the OOPSLA 2000 Workshop on Best-practices in Business Rule Design and Implementation*, 2000, p. 10-14.
25. Simple Rule Markup Language (SRML). Technology Reports. 2001 URL: http://xml.coverpages.org/srml.html [žiūrėta: 2006-03-20].

26. Šmaižys A. *Verslo taisyklių taikymas informacinių sistemų kūrimui.* Baigiamasis magistro darbas, Klaipėdos Universitetas, darbo vadovas: prof. dr. O. Vasilecas, 2004, p.1-83.
27. Šmaižys A**.** XML-based representation of business rules, *VII-osios jaunųjų mokslininkų konferencijos medžiaga*. Vilnius: VGTU. Technika. 2004, p.1-7.
28. Spofford G., Harinath S., Webb C., Hai Huang D., Civardi F. *MDX Solutions Second Edition With Microsoft SQLServer™ Analysis Services 2005 and Hyperion Essbase.* Wiley Publishing, 2006.
29. Staab S., Schnurr H.P. Knowledge and Business Processes: Approaching an Integration. *Knowledge Management and Organizational Memories*. Kluwer. URL: www.aifb.uni-karlsruhe.de/~sst/Research/Publications/staabschnurr-om99.pdf [žiūrėta: 2006-11-20].
30. Thalhammer T.; Schrefl M.; Mohania M. Active Data Warehouses: Complementing OLAP with Active Rules. *Data & Knowledge Engineering (DKE),* Vol. 39, No. 3, Elsevier Science, 2001, p. 241-269.
31. RuleML. The Rule Markup Initiative. URL: http://www.ruleml.org/ [žiūrėta: 2006-03-23].
32. Thomsen E. OLAP *Solutions Building Multidimensional Information Systems*. Second Edition, John Wiley & Sons, 2002.
33. Vasilecas O., Šmaižys A. Verslo taisyklių panaudojimas duomenų analizei ir informacijos pateikimui. *Informacinės technologijos 2005, Konferencijos pranešimų medžiaga*. Kaunas: Kauno technologijos universitetas, 2005, p.655-663.
34. Vasilecas O., Šmaižys A. Business rule based data analysis for decision support and automation. *Proc. of the International Conference on Computer Systems and Technologies "CompSysTech'06"*, Varna, Bulgaria, session II, no. 9, 2006, p. 1–6.
35. Vasilecas O., Šmaižys A. The Framework: an Approach to Support Business Rule Based Data Analysis. *Proc. of 7th International IEEE Baltic Conference on Databases and Information Systems*, July 3, 2006, p. 141 – 147.
36. Velinov G., Kon-Popovska M. Physical database design for data warehouses. *Proc. of the Second International Conference on Informatics and Information Technology*. Macedonia, 2001, p. 318-334.
37. Wagner G., Taveter K. Agent-Oriented Enterprise Modeling Based on Business Rules. *Proc. of 20th Int. Conf. on Conceptual Modeling (ER2001)*, Yokohama, Japan, Springer-Verlag, 2001, p.1-14.