Liet. matem. rink., **44**, *spec. nr.*, 2004, 383–387

Verslo taisyklių sistemų modeliavimas UML ir jų realizacijos reliacinėse DBVS analizė

Olegas VASILECAS, Vladimir AVDEJENKOV (VGTU)
el. paštas: olegas@fm.vtu.lt, vladimir@isl.vtu.lt

1. Įvadas

Pastaruoju metu, kalbant apie informacines sistemas, vis dažniau yra atkreipiamas dėmesys į verslo taisykles informacinėse sistemose ir į informacines sistemas, orientuotas į taisykles [1], [2].

Verslo taisyklės yra neatskiriama verslo dalis, jos aprašo verslo funkcionavimą, apibrėžia verslo procesus, vykstančius organizacijos viduje ir išorėje. Verslo taisykles formuluoja verslo vartotojai, ir dažniausiai formuluoja paprasta neformalizuota kalba (anglų, lietuvių ir t.t.) [3]. Galima paminėti, kad egzistuoja daugelis taisyklių tipų: tai yra ribojimo, skaičiavimo ir t.t. [4]. Skiriami taisyklių atsiradimo šaltiniai, naudotojai.

Jeigu kalbėsime apie įprastas informacines sistemas, tai tokiais atvejais suformuluotas verslo taisykles programuotojai rankiniu būdu įdiegia į informacines sistemas.

Antra vertus, yra verslo taisyklių valdymo sistemos (*business rules management systems*), kurių esmė – atskirti verslo taisykles nuo taikomųjų programų. Tai yra daroma tam, kad būtu lengviau centralizuotai jas projektuoti, redaguoti, valdyti. Toks metodas sutrumpina informacinės sistemos kūrimo laiką, žymiai sumažina palaikymo ir patobulinimo kaštus.

Aktyvios duomenų bazių valdymo sistemos leidžia informacinių sistemų kūrėjams efektyviai valdyti ir realizuoti dalykinėje srityje naudojamas žinias. Tai galima pasiekti naudojant aktyvias taisykles, arba dar kitaip vadinamus trigerius [5]. Problema yra ta, kad jeigu trigerių yra pakankamai daug, jie apima visą duomenų bazę, tai kontroliuoti ir vystyti tokią sistemą yra sudėtinga ir reikalauja daug darbo sąnaudų. Trūksta priemonių, kurios leistų modeliuoti trigerius vieningu būdu, užtikrintų aktyvių taisyklių vertinimą, konfliktų nustatymą, rekursijos fiksavimą.

Darbe [6] nagrinėjama galimybė koncepcinius grafus naudoti vieningam dalykinės srities duomenų struktūroms, dalykinės srities ribojimams bei dalykinės srities objektų elgsenos modeliavimui. Darbe siūlomas būdas susieti tradicinį ER modeliavimą su koncepcinių grafų modeliu naudojantis transformavimo taisyklėmis. Darbe pristatomas koncepcinių grafų formalizmas ir parodoma, kad visų tipų dalykinės srities žinias galima modeliuoti koncepciniais grafais. Taip pat pristatomas metodas transformuoti žinias iš ER modelio į modelį koncepciniais grafais. Kitame, susietame darbe [7] siūlomas SQL trigerių gavimo metodas iš UML/OCL specifikacijos. Darbe apsiribojama tik darnos palaikymo trigeriais. Pagal autorių tvirtinimą kitų trigerių tipų gavimas, šio metodo pagalba neįmanomas. Yra analizuojama OCL specifikacija, ir jos

pagrindu apribojimai transliuojami į SQL trigerio specifikaciją. Analizuojama OCL specifikacija, ir jos pagrindu apribojimai transliuojami į SQL trigerio specifikacija. Pagal autorių tvirtinimą kitų trigerių tipų gavimas, šio metodo pagalba neįmanomas. Yra analizuojama OCL specifikacija, ir jos pagrindu apribojimai transliuojami į SQL trigerio specifikaciją.

Atlikus nagrinėjamos srities literatūros analizė, paaiškėjo, kad iki šiol buvo bandoma UML/OCL pagalba projektuoti ir transformuoti į SQL trigerius tik atskiras verslo taisyklės. Dar nėra metodo, kaip UML pagalba projektuoti verslo taisyklių sistemą bei transformuoti ją į SQL trigerių sistemą. Šiame straipsnyje yra nagrinėjama galimybė UML priemonėmis modeliuoti verslo taisyklių sistemą bei tolimesnį jos realizavimą duomenų bazių valdymo sistemoje. Tokiu būdu taisyklės centralizuotai apjungiamos ir valdomos UML modelyje. Aprašytos taisyklės yra transformuojamos į duomenų bazių trigerius, kurie dirba atskirai nuo taikomosios programos.

2. Verslo taisyklės ir UML

UML – yra trumpinys *Unified Modeling Language* (Unifikuota Modeliavimo Kalba). UML pagalba galima modeliuoti įvairaus sudėtingumo sistemas [8]. Mūsų uždavinys yra UML pagalba sumodeliuoti verslo taisyklių sistema, ir transformuoti ją į aktyvią RDBVS. Klasių diagrama (*class diagram*) naudojama modelio statinės struktūros aprašymui. Ji puikiai tinka duomenų bazės loginio modelio modeliavimui.

Trigeris yra aktyvus DBVS elementas, kuris bendru atveju aprašo sistemos elgsena. Tokį elementą galima aprašyti tokiu UML diagramų pagalba, kurios aprašo sistemos elgseną. UML kalboje sistemos elgsenai aprašyti naudojamos procesų (*activity*) ir būsenų (*statechart*) diagramos. Tokiu atveju ECA taisyklę sumodeliuoti galima šių diagramų pagalba.

Mūsų uždavinys – sukurti taisyklių sistemą. Reikia aprašyti taisyklių prioritetus, nustatyti konfliktus, rekursiją ir kt. Tam tikslui papildomai yra įvedama bendradarbiavimo diagrama (*collaboration diagram*).

Taigi, išanalizavus UML galima padaryti išvadą, kad duomenų bazės modelis aprašomas naudojant klasių diagramą, o verslo taisyklės aprašomos procesų ir būsenų diagramų pagalba, o tai, kaip vienos taisyklės surištos su kitomis taisyklėmis, aprašoma bendradarbiavimo diagramos pagalba. Tokio diagramų rinkinio užtenka verslo taisyklių sistemos modeliavimui.

3. Verslo taisyklės ir SQL trigeriai

RDBVS trigeriai (*triggers*) yra aktyvūs DBVS elementai, kurie aktyvuojasi susidarius tam tikroms sąlygoms ir užtikrina platų funkcionalumo spektrą nuo darnos palaikymo iki sudėtingų verslo taisyklių realizavimo.

Trigerį galima nagrinėti kaip ECA taisyklę: įvykis (*Event*), sąlyga (*Condition*) ir veiksmas (*Action*). Tai yra trigeris reaguoja į kažkokį įvykį (*Event*), patikrinama sąlyga (*Condition*) ir priklausomai nuo sąlygos tikrinimų rezultatų atliekamas veiksmas (*Action*). Galimi trys įvykio variantai: eilutės įterpimas (*INSERT*), eilutės trynimas (*DELETE*) arba eilutės atnaujinimas (*UPDATE*). Jeigu įvyko nustatytas įvykis, yra tikrinama sąlyga – įrašytų, ištrintų duomenų patikrinimas, duomenų iš kitų lentelių

patikrinimas ir t.t. Priklausomai nuo sąlygos patikrinimo rezultatų atliekamas veiksmas – praktiškai bet kokia SQL kalbos komanda (transakcijos atšaukimas, duomenų pakeitimas kaip bazinėje lentelėje, taip ir bet kurioje lentelėje arba pranešimų siuntimas).

Jeigu egzistuoja keli trigeriai, yra galimybė naudojant raktinius žodžius nustatyti jų prioritetus. Taip pat galima leisti arba uždrausti trigerių rekursiją. Šios papildomos komandos leidžia valdyti trigerius.

Bendrai trigerio sintaksė užrašoma taip (MS SQL Server 2000):

```
CREATE TRIGGER trigger_name
ON { table | view }
[ WITH ENCRYPTION ]
  { { FOR | AFTER | INSTEAD OF } { [ INSERT ] [ , ] [ UPDATE ] [,] [DELETE] }
    [ WITH APPEND ]
    [ NOT FOR REPLICATION ]
    AS
    [ { IF UPDATE ( column )
      [ { AND | OR } UPDATE ( column ) ]
        [ ...n ]
    | IF ( COLUMNS_UPDATED ( ) { bitwise_operator } updated_bitmask )
        { comparison_operator } column_bitmask [ ...n ]
    } ]
    sql_statement [ ...n ]
}
```

Trigerių prioritetai nustatomi papildomos procedūros pagalba:

```
sp_settriggerorder [@triggername = ] 'triggername'
, [@order = ] 'value'
, [@stmttype = ] 'statement_type'
```

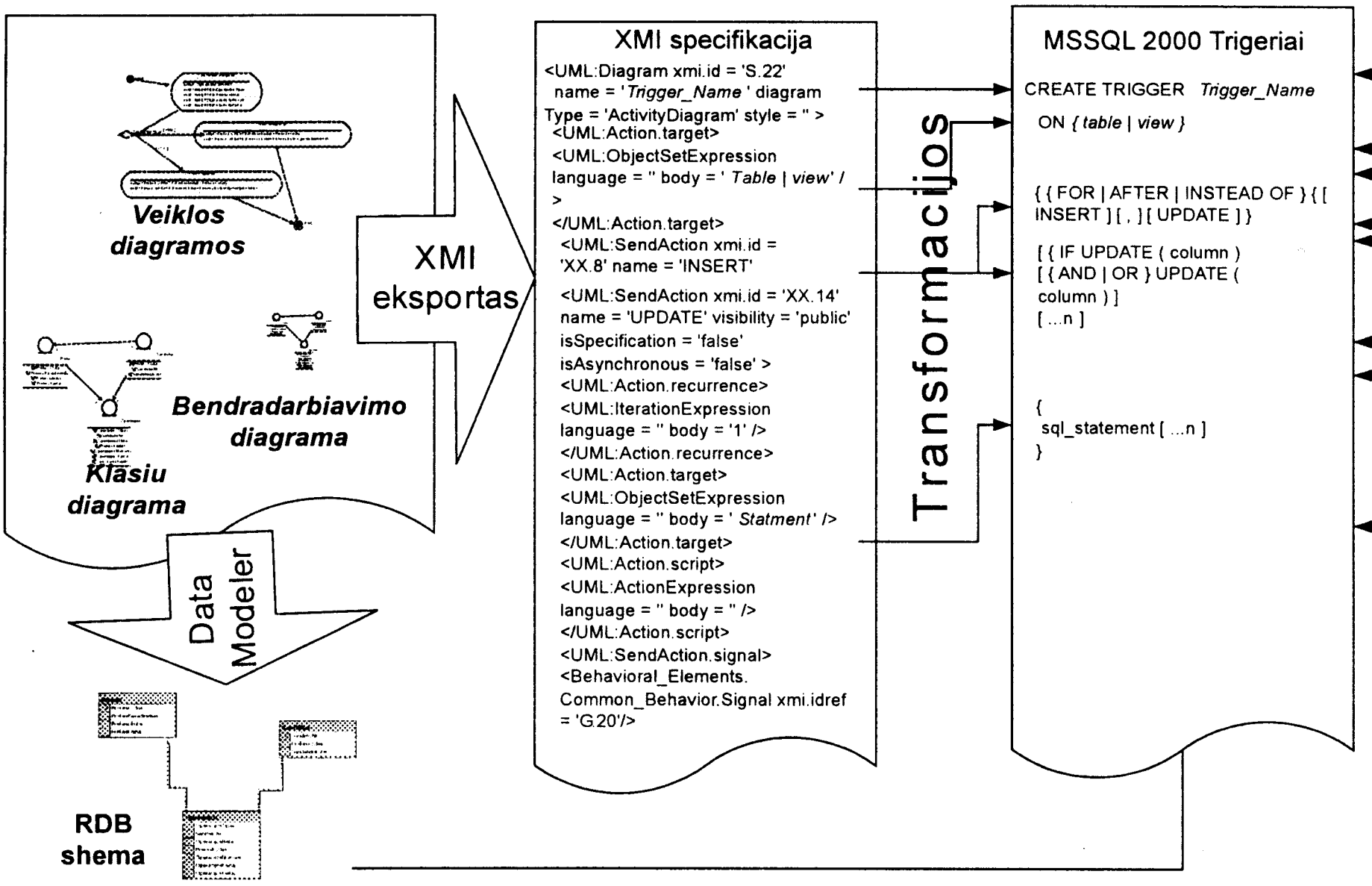
trigger_name – aprašo trigerio vardą, kuriuo jis identifikuojamas duomenų bazėje;
table | view – aprašo lentelės pavadinimą, kuriai suveikia trigeris;
[INSERT] [,] [UPDATE] [,] [DELETE] – įvykiai, dėl kurių suveikia trigeris; galima užduoti nuo vieno iki trijų įvykių;
IF UPDATE (column) – patikrinama sąlyga, ar pakeistas atitinkamas stulpelis;
sql_statement [...n] – veiksmas (SQL kalbos komandos), kuris vykdomas atitikus sąlygai.

Į tokį pavidalą ir bus transformuojamos verslo taisyklės. Kartu trigerių aibė sudarys verslo taisyklių sistemą.

4. Verslo taisyklių gavimo metodas

Atlikus UML analizę, DBVS aktyvių taisyklių analizę, o taip pat ir atsižvelgiant į jau padarytus tyrimus šioje srityje, galime pasiūlyti SQL trigerių iš UML specifikacijos gavimo metodą. Bendru atveju šis procesas pavaizduotas 1 pav.

Procese dalyvauja trijų tipų diagramos: klasių, veiklos ir bendradarbiavimo. Klasių diagrama naudojama duomenų struktūros modeliavimui, o veiklos diagrama naudojama trigerių modeliavimui. Klasių diagrama *Data Modeler* pagalba transformuojama



1 pav. SQL trigerių iš UML diagramų gavimo procesas.

į reliacinės domenų bazes schemą, o veiklos ir bendradarbiavimo diagrama išsaugoma XML formate. XML specifikacija yra analizuojama, ir atliekant transformacijas gaunamas SQL trigeriai. XML formatas vis dažniau naudojamas kaip standartas specifikuojant verslo taisykles [9].

Taip yra pasiekiamas uždavinio tikslas: verslo taisyklių sistema modeliuojama UML diagramų pagalba, o taisyklių modelis transformuojamas į DB trigerius, kurie palaiko verslo procesus nepriklausomai nuo taikomųjų programų. Su UML diagramomis gali dirbti ne tik programuotojai bet ir verslo žmonės, o tai gali palengvinti verslo taisyklių kūrimą ir palaikymą.

5. Išvados

Straipsnyje nagrinėjamos verslo taisyklės, verslo taisyklių valdymo sistemos bei galimybės UML diagramų pagalba modeliuoti verslo taisyklių sistemas ir šias taisyklių sistemas, trigerių pagalba, realizuoti aktyviose RDBVS. Duomenų modelio aprašymui siūloma naudoti klasių diagramą, verslo taisyklių specifikavimui – veiklos diagramas, o verslo taisyklių sąveikos nustatymui – bendradarbiavimo diagramas. UML pagalba užrašytos taisykles (XML dokumentas) analizuojamos ir transformuojamos į DML (Data Manipulating Language) komandas. Atlikta analizė parodė, kad UML pagalba galima modeliuoti taisyklių sistemas ir, naudojant išsamų taisyklių modelį, generuoti aktyvias DBVS taisykles (trigerius).

Literatūra

1. V. Avdejenkov, I. Valatkaitė, O. Vasilecas, Verslo taisyklių modeliavimas UML ir jų realizavimas reliacinėje DBVS, *Informacinės technologijos 2003*, KTU, ISBN 9955-09-335-8.
2. B. von Halle, *Business Rules Applied – Business Better Systems Using the Business Rules Approach*, ISBN 0-471-41293-7.
3. R.G. Ross, *Principles of the Business Rule Approach*, *Addison Wesley*, January 30 (2003), ISBN 0-201-78893-4.
4. R.G. Ross, *The Business Rule Book*: *Classifying, Defining and Modeling Rules*, 2nd edition, Database Research Group, Boston, MA (1997).
5. O. Vasilecas, ECA taisyklių realizacija aktyviose duomenų bazių valdymo sistemose, *Lietuvos mokslas ir pramonė, Konferencijos „Informacinės technologijos 2002" paranešimų medžiaga*, Kaunas, Technologija (2002), pp. 71–77.
6. I. Valatkaitė, O. Vasilecas, Verslo taisyklių modeliavimas koncepciniais grafais ir jų realizavimas naudojant aktyvių duomenų bazių trigerius, *Liet. matem. rink.*, **42** (spec. nr.), 289–293 (2002).
7. M. Badwy, K. Richta, Deriving triggers from UML/OCL specification, in: *Proceedings of Information Systems Development*, Ryga, Latvija (2002).
8. G. Booch, J. Rumbaugh, I. Jacobson, *The Unified Modeling Language User Guide*, Addison-Wesley (2000).
9. N. Benjamin, K. Grosof, Standardizing XML rules, *Preliminary Outline of Invited Talk International Joint Conference on Artificial Intelligence* (IJCAI-01).

SUMMARY

O. Vasilecas, V. Avdejenkov. Business rules systems modelling with UML and rules enforcing in active relational databases

Business rules, control system of business rules, modelling of business rules by using UML diagrams and possibility to apply these systems of rules in the RDBVS are analysed in the article. Analysis showed that systems of rules could be modelled by UML and active rules of DBVS could be generated using model of rules. Possibility to describe priorities for rules, to determine conflicts between rules and recursions using UML diagrams was researched.

Keywords: business rules, business rules management systems, unified modeling language, ECA rules, bussiness rules engine.