

Verslo taisyklių modeliavimas koncepciniais grafais ir jų realizavimas naudojant aktyvių duomenų bazių trigerius

Irma VALATKAITĖ, Olegas VASILECAS (VGTU)

el. paštas: irma@isl.vtu.lt, olegas@isl.vtu.lt

1. Įvadas

Dalykinės srities žinios gali būti modeliuojamos naudojant jau tapusias tradicinėmis modeliavimo kalbas – esybių ryšių (ER) ar vieningą modeliavimo (UML) kalbas. Tačiau atskirais atvejais šioms kalboms trūksta išraiškos galimybių visiems dalykinės srities aspektams modeliuoti vieningu būdu, kaip pastebėjo Mineau ir kiti (2000); kas žymiai sumažintų sistemos projektuotojo darbą ir leistų turėti vieną dalykinės srities modelį, kuriame atsispindėtų ir dalykinės srities struktūra, ir elgsenos taisyklės, ir struktūros bei elgsenos ribojimai.

Šiame straipsnyje koncepciniai grafai (toliau – CG) yra pristatomi kaip tokia modeliavimo kalba (Sowa, 1984). CG yra formali koncepcinio modeliavimo kalba, turinti grafinę notaciją bei keletą alternatyvių standartinių vaizdavimo formų – CGIF (koncepcinių grafų apsikaitimo formatas), KIF (žinių apsikaitimo formatas) ir LF (tiesinis formatas) (Sowa, 2000).

Šiuo metu UML (Booch ir kiti, 2000) yra turbūt plačiausiai naudojama modeliavimo kalba. Tačiau sudėtingesnei dalykinės srities objektų elgsenai modeliuoti turi būti naudojama OCL kalba (Object Constraint Language). OCL yra loginė kalba, neturinti jokios grafinės notacijos. Tokiu būdu yra gaunami keli tos pačios dalykinės srities modeliai. Tuo tarpu CG turi priemonių bei išraiškos galimybių modeliuoti visus dalykinės srities aspektus vieningu būdu (Valatkaitė ir kiti, 2002).

Šiame straipsnyje yra siūloma naudoti CG kaip modeliavimo kalbą informacinėms sistemoms modeliuoti ir siūlomas žinių transformavimo metodas. Parodoma, kad galima sugeneruoti duomenų bazės trigerius iš dalykinės srities modelio, užrašyto CG.

2. Dalykinės srities žinios

Dalykinės srities žinios gali būti analizuojamos trimis aspektais: nagrinėjama dalykinės srities duomenų struktūra, objektų elgsena ir struktūros bei elgsenos ribojimai. Jeigu dalykinės srities modelis apima šiuos tris aspektus, galima sakyti, kad dalykinė sritis informacinės sistemos požiūriu yra pilnai sumodeliuota (Mineau, 2000).

Objektų elgseną paprastai nusako taip vadinamos verslo taisyklės. Verslo taisyklė suprantama kaip „teiginys, kuris apibrėžia ar apriboja kuri nors verslo aspektą; jos paskirtis yra nusakyti verslo struktūrą, taip pat kontroliuoti ar įtakoti verslo elgseną“ (Gottesdiener, 1997). Šiame straipsnyje remiamasi verslo taisyklių klasifikacijos schema, pasiūlyta Herbst ir kitų (1994). Verslo taisyklės skirstomos į darnos ir veikos. Darnos taisyklės nurodo duomenų darnos reikalavimus, o veikos taisyklės nusako veiksmus ar operacijas, kurios turi būti įvykdytos.

Aktyvių duomenų bazių (turinčių įvykių palaikymo ir būsenų stebėjimo mechanizmą) tyrimuose siūloma verslo taisyklės struktūrizuoti naudojant ECA taisyklių mechanizmą (Event, Condition, Action – įvykis, sąlyga, veiksmas) (Dayal ir kiti, 1988). ECA taisyklės yra tokios formos: įvykus *įvykiui*, jei tenkinama *sąlyga*, vykdomas *veiksmas*.

Jeigu verslo taisyklės, palaikomos informacinėje sistemoje, yra „išmėtytos“ visos sistemos ribose, tai reiškia didesnes darbo sąnaudas tokios sistemos vystymui bei palaikymui. Reikalavimas, kad visos verslo taisyklės būtų modeliuojamos vieningu būdu, yra akivaizdus. Naudojant aktyvias duomenų bazes ir trigerių mechanizmą galima visas verslo taisykles turėti viename repozitoriume, tuo būdu yra pasiekiamas vieningas realizavimo būdas.

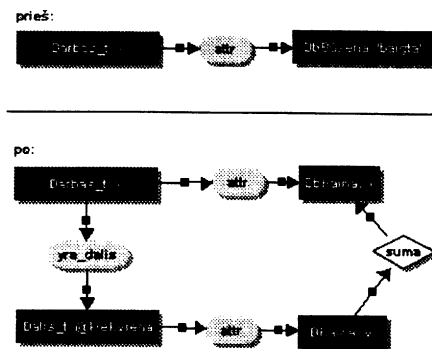
Trigeriai verslo taisyklės taip pat atvaizduoja pagal ECA schemą: įvykis yra duomenų bazės keitimo operacija; sąlyga yra predikatas arba užklausa į duomenų bazę; veiksmas nurodo, kokios operacijos turi būti atliktos aktyvavus taisyklę.

Šiame straipsnyje bus nagrinėjamos tik veikos taisyklės (toliau vadinamos tiesiog verslo taisyklėmis), kadangi būtent jos yra sudėtingiausias modeliavimo bei realizavimo požiūriais – jos apibrėžia visus versle naudojamus veiklos bei skaičiavimų algoritmus ir netrivialius duomenų ryšius.

3. Verslo taisyklės kaip procesai CG modelyje

Verslo taisyklės gali būti struktūrizuojamos naudojant ECA mechanizmą. Tai reiškia, kad kiekvienai taisyklei turi būti nustatyti jos įvykis, sąlyga ir veiksmas.

ECA taisyklės struktūra yra atvaizduojama į CG procesą, kurį pasiūlė ir aprašė Mineau (2000). Modeliuojant procesais, CG yra grupuojami į taisykles, kurios susideda iš



I pav. Verslo taisyklė Nr. 4 kaip CG procesas.

1 lentelė. Dalykinės srities verslo taisyklės

Nr.	Verslo taisyklė natūraliaja kalba	Taisyklės tipas
1.	Kiekvienas darbas turi turėti bent vieną jam priskirtą operaciją	Darnos
2.	Kiekviena darbo operacija – darbo dalis – turi turėti jai priskirtą bent vieną darbuotoją	Darnos
3.	Darbo dalies kaina apskaičiuojama padauginus darbuotojo sugaištų valandų skaičių iš to darbuotojo darbo valandos kainos	Veikos
4.	Darbui sugaištas laikas apskaičiuojamas kaip atskiroms darbo dalims sugaištų darbo valandų suma	Veikos

„prieš“ ir „po“ sąlygų. Kai „prieš“ sąlygos žinių bazėje yra tenkinamos, ta žinių bazė yra papildoma koncepciniais grafais, nurodytais „po“ sąlygose.

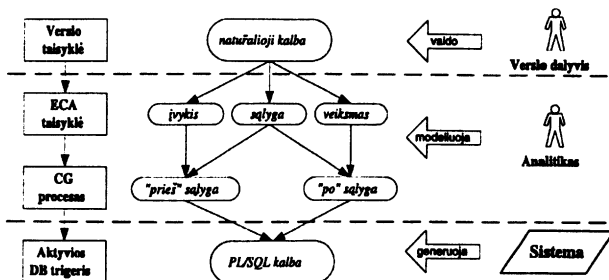
Panagrinėsime pavyzdį iš verslo dalykinės srities:

Įmonė gamina pakavimo dėžutes. Kiekvienas užsakymas yra atliekamas kaip atskiras darbas, jam yra priskiriama keletas operacijų iš tipinio rinkinio. Kiekvienai darbo operacijai yra paskiriamas darbuotojas ir žymima, kiek laiko jis sugaišo kiekvienai operacijai atlikti. Kai darbas yra baigtas, suskaičiuojama darbo kaina (pagal darbuotojų valandos darbo kainą) bei sugaištas valandų kiekis.

Šioje dalykinėje srityje galima identifikuoti keletą verslo taisyklių, kurios pateiktos 1 lentelėje. Verslo taisyklės Nr. 3 ir 4 yra veikos taisyklės ir negali būti realizuojamos duomenų bazėje kaip darnos ribojimai. Toliau nagrinėjama tik verslo taisyklė Nr. 4. 2 lentelėje yra parodyta, kaip ši taisyklė struktūrizuojama pagal ECA mechanizmą, o jos CG modelis yra parodytas 2 pav.

2 lentelė. Verslo taisyklės pagal ECA schema

Nr.	Įvykis	Sąlyga	Veiksmas
4	keičiama darbo būseną	darbo būseną yra „baigtas“	suskaičiuojamas darbo valandų kiekis kaip visų darbo dalių valandų suma



2 pav. Metodo schema.

4. Trigerių generavimo metodas

Sukurtas trigerių generavimo metodas nusako, kaip verslo taisyklių modelis, užrašytas CG, naudojamas verslo taisyklių realizacijai aktyvių duomenų bazių trigeriais gauti. Metodo įeities duomenys yra verslo taisyklės CGIF formatu, o rezultatas – aktyvios duomenų bazės trigeris (trigeriai). Metodo schema parodyta 3 pav.

Pagal apibrėžimą (ISO&ANSI, 1999) trigeris nurodo veiksmų seka, kuri aktyvuojama ir įvykdoma tada, kai įvyksta nurodytas įvykis. Trigeris susideda iš kelių dalių, kurios nustatomos pagal verslo taisyklę:

- Vardas: unikalus trigerio vardas;
- Įvykis: trigerio suaktyvinimo įvykis (duomenų keitimo operacija);
- Aktyvavimo laikas: nusako, ar trigeris aktyvuojamas prieš ar po įvykio;
- Vykdomo sritis: apibrėžia, ar trigeris vykdomas operacijai ar įrašui;
- Sąlyga: trigerio veiksmo vykdymo sąlyga;
- Veiksmas: konkrečios duomenų atnaujinimo operacijos, kurios turi būti įvykdytos.

Straipsnyje pateikiami metodo taikymo verslo taisyklei Nr. 4 (2 pav.) rezultatai: gautas trigeris pateiktas 3 lentelėje.

3 lentelė. Verslo taisyklė Nr. 4: CGIF ir trigeris

CGIF	Trigeris
[prieš:" [DbBusena:*a'baigta'] [Darbas_t:*b'x'] (attr?b?a)] [po:" [Dalis_t:*c'@kiekviena'] [DbKaina:*d'y'] [Darbas_t:*e'x'] [DKaina:*f'v'] (attr?e?d) (attr?c?f) (yra_dalis?e?c) <suma?f ?d>]	CREATE OR REPLACE TRIGGER trigeris_3 before update on Darbas_t for each row WHEN (new.DbBusena = 'baigta') declare a number; begin select sum(DKaina) into a from Dalis_t where d_dbid=:new.dbid; :new.DbKaina := a; end;

5. Išvados ir tolimesni tyrimai

Šiame straipsnyje išnagrinėtas koncepcinių grafų naudojimas dalykinės srities žinioms modeliuoti. Atlikta analizė parodė:

1. Koncepcinių grafų teorija, sukurta Sowa ir išplėsta kitų autorių, turi išsamų ir vieningą modeliavimo konstrukcijų rinkinį, tinkamą modeliuoti visus dalykinės srities aspektus vieningu būdu.

2. Dabartiniai aktyvių duomenų bazių tyrimai parodė, kad galima generuoti trigerius naudojantis UML dalykinės srities modeliu, tačiau tik ta jo dalimi, kuri yra aprašyta formalia OCL kalba ir vaizduoja darnos taisykles.
3. Pateiktas trigerių generavimo metodas, pagal kurį dalykinės srities verslo taisyklės realizuojamos aktyvių duomenų bazių trigeriais. Eksperimento metu buvo apibrėžtos taisyklės, kurios, sumodeliuotos CG procesais, pagal pateiktą metodą buvo realizuotos trigeriais.

Tolimesnių tyrimų kryptys yra kelios: išplėsti ir realizuoti pasiūlytą metodą; palyginti UML, UML su OCL ir CG išraiškos priemonės verslo taisyklėms modeliuoti; apibrėžti nepriklausomą nuo DBVS trigerių generavimo metodą.

Literatūra

- [1] G. Booch, J. Rumbaugh, I. Jacobson, *The Unified Modeling Language User Guide*, Addison-Wesley (2000).
- [2] U. Dayal, A.P. Buchmann, D.R. McCarthy, Rules are objects too: a knowledge model for active, object-oriented database management system, in: *Advances in Object-Oriented Database Management Systems*, Dittrich, K.R. (Ed.), Springer, Berlin (metai ???), pp. 129–143.
- [3] H.S. Delugach, Dynamic assertion and retraction of conceptual graphs, in: *Proceedings of Sixth Annual Workshop on Conceptual Graphs*, Way, E.C. (Ed.), SUNY Binghamton, Binghamton, New York (1991), pp. 15–26.
- [4] E. Gottesdiener, Business rules show power and promise, *Application Programming Trends*, 4(3) (1997).
- [5] H. Herbst, G. Knolmayer, T. Myrach, M. Schlesinger, The specification of business rules: a comparison of selected methodologies, *Methods and Associated Tools for the Information System Life Cycle*, Verijn–Stuart, A.A., Olle T.-W., Elsevier, Amsterdam, 29–46 (1994).
- [6] International Organization for Standardization (ISO) & American National Standards Institute (ANSI), ANSI/ISO/IEC 9075–2:99. *ISO International Standard: Database Language SQL*, Part 2: Foundation (SQL/Foundation) (1999).
- [7] G.W. Mineau, R. Missaoui, R. Godin, Conceptual modeling using conceptual graphs, in: *Proceedings of the 7th International Workshop on KRDB 2000*, M. Bouzeghoub, M. Klusch, W. Nutt, U. Sattler (Eds), Berlin (2000), pp. 73–86. <http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-29/>,
- [8] J.F. Sowa, *Conceptual Structures: Information Processing in Mind and Machine*, Addison-Wesley (1984).
- [9] J.F. Sowa et al., *Conceptual Graph Standard, American National Standard NCITS*, T2/98-003 (2000), <http://www.jfsowa.com/cg/cgstand.htm>.
- [10] I. Valatkaitė, O. Vasilecas, Žinių transformacija naudojant koncepcinius grafus, in: *Konferencijos "IT'2002" pranešimų medžiaga*, KTU, Kaunas, Technologija (2002), pp. 63–70.

Deriving active database triggers from business rules model with conceptual graphs

I. Valatkaitė, O. Vasilecas

ECA rules implemented into database management systems (DBMS) as triggers have proven to be powerful and effective mechanisms to enforce business rules in applications. Collection of all business rules implemented in united repository of specific DBMS reduces efforts of implementation and maintenance, so does uniform domain conceptual model. Several conceptual modelling languages might be used but we propose to use conceptual graphs (CGs) because CGs offer constructs for modelling all static and dynamic aspects of application domain in a uniform way, and have both, a standard representation, and a graphical notation. In this paper we propose a generic algorithm to generate database triggers from domain model represented by CGs. The algorithm uses Conceptual Graphs Interchange Format (CGIF) representation. Such usage of CGs for business rules modelling is advocated.