STUDIA INFORMATICA 2012
Volume 33 Number 2A (105)

Ewa MAGIERA
Uniwersytet Śląski w Katowicach, Instytut Informatyki

INŻYNIERIA TEMPORALNYCH WYMAGAŃ W ŚWIETLE PARADYGMATU DW 2.0

Streszczenie. W celu pełnego opisu temporalnych zależności w procesie gromadzenie wiedzy dla hurtowni danych został zaproponowany model czasu. W pierwszej części omówiono zagadnienia związane z czasem w kontekście hurtowni danych, następnie zaprezentowano założenia paradygmatu DW 2.0. Na podstawie liniowej dyskretnej temporalnej logiki został utworzony model czasu, którego wybrane elementy opisano.

Słowa kluczowe: hurtownia DW 2.0, temporalna wiedza, model czasu, gromadzenie wymagań

TEMPORAL REQUIREMENTS ENGINEERING AND DW 2.0

Summary. Issues concerning gathering, representation and modeling temporal relations are discussed. The temporal aspects are taken with accordance to the data warehousing systems and paradigm DW 2.0. The assumptions of the temporal linear logic and the selected elements of time model are described.

Keywords: data warehouse 2.0, temporal knowledge, time model, gathering requirements

1. Wprowadzenie

Na wczesnym etapie analizy istotnym problemem jest prawidłowe i pełne zgromadzenie wymagań dla systemów transakcyjnych i hurtowni danych. Niepełne opisanie wymagań, szczególnie o zależnościach temporalnych, skutkuje wadliwym modelem transakcyjnej bazy danych i właściwie niemożnością zbudowania hurtowni danych zgodnej z wymaganiami biznesowymi. Przykładem może być tutaj wdrożenie zintegrowanego informatycznego systemu zarządzania w jednej z organizacji sektora publicznego, które oparte było na oprogramowa-

niu firmy SAP. Z powodu niepoprawnie zgromadzonych wymagań źle zaprojektowano infotypy modułów kadrowego i płacowego. Dotyczyły one struktury organizacyjnej i nie uwzględniały zmienności w czasie tej struktury. W efekcie tego poprawne wdrożenie hurtowni danych na bazie tych modułów wymagało wprowadzenia zmian w strukturach danych (infotypach) wspomnianych, działających modułów. Tego typu modyfikacje na tzw. systemach produkcyjnych są zawsze obarczone znaczącym ryzykiem i zaburzają płynność działania organizacji. Zagadnienia związane z czasem w informatyce są przedmiotem rozważań od wielu lat, czego efektem są liczne prace dotyczące tworzenia i wykorzystania temporalnych logik [5, 12, 14]. Poruszają one temporalne zagadnienia w aspekcie baz danych i hurtowni [3, 6, 11, 20, 22]. Nowe podejście do struktury hurtowni danych, paradygmat DW 2.0 [10], wprowadza szczególne podejście do przechowywania informacji zależnej i zmiennej w czasie. Tworząc model bazy danych zwraca się uwagę na odzwierciedlenie rzeczywistego świata zewnętrznego, zgodnego z wymaganiami biznesowymi w danym momencie. Zwykle model bazy pozwala na przechowywanie aktualnego (bieżącego) stanu rzeczywistości biznesowej i jeśli stan ten ulegnie zmianie to dane w bazie są aktualizowane, a poprzednie w większości wypadków tracone. Systemy transakcyjne często nie dają możliwości gromadzenia danych historycznych i rejestracji zmiany ich wartości w czasie; w okresie do kilku lat od startu produktywnego systemów biznesowo-produkcyjnych wiele organizacji decyduje się na wdrożenie hurtowni danych jako głównego źródła informacji na potrzeby prognoz, raportów i analiz, w tym statystycznych. Często analitycy baz danych zapominają jak duże znaczenie dla zaprojektowania hurtowni danych mają tzw. dane zmienne w czasie i odpowiednia rejestracja ich wartości. Bazy danych zamodelowane na gromadzenie informacji wrażliwej na zmienność w czasie zwane są temporalnymi bazami danych. W literaturze [13, 20, 22] w przypadku baz danych rozróżnia się dwa rodzaje czasu: tzw. czas ważności (ang. *valid time*), określający przedział czasu, w którym dana jest prawdziwa w świecie biznesowym, oraz czas transakcji (ang. *transaction time*) odpowiadający czasowi wykonania transakcji realizującej pojawienie się danej w bazie danych. O ile ten ostatni czas jest automatycznie generowany i zapisywany przez system, o tyle pierwszy, jako że dotyczy świata zewnętrznego, wymaga wprowadzenia do systemu. Informacje temporalne są sprowadzane jedynie do przedstawienia w każdej krotce wartości czasu jednego ze wskazanych typów lub obu. W drugim przypadku relacja zwana jest bitemporalną.

1.1. Czas w hurtowniach danych

Problem czasu w hurtowniach danych poruszył, jako jeden z pierwszych, Kimball w [11], proponując trzy typy (sposoby) zapamiętania zmian wartości atrybutu, istotnych z punku widzenia użytkownika. Typ pierwszy odpowiada zapisowi danej w przypadku zmiany wartości z jednoczesna utratą danych wcześniejszych, drugi typ zakłada utworzenie nowej krotki

z nowym indeksem i nową wartością, która uległa zmianie. Trzeci typ to wprowadzenie dwóch kolumn dla wartości atrybutu ulegającego zmianie w czasie; w jednej jest pamiętana wartość poprzednia, w drugiej aktualna, nadal jednak niepamiętana jest pełna historia wartości atrybutów.

O ile zamodelowanie w opisany sposób temporalnych informacji w zwykłych bazach transakcyjnych i hurtowniach pierwszej generacji wydawało się być w większości przypadków wystarczające, o tyle obecne biznesowe wymagania wykorzystania zgromadzonych danych są dużo wyższe. Powodem jest szeroko obecnie dostępne oprogramowanie do analiz statystycznych, eksploracji i raportowania danych, bo to właśnie informacje przechowywane w hurtowniach danych stanowią dane źródłowe dla analiz biznesowych.

Dużo lepsze rozwiązanie, zapewniające zarządzanie zmianami danych i struktur w czasie, to wielowersyjne hurtownie danych, które szczegółowo, w różnych aspektach opisano w pracach [1, 17, 23]. W [19] przedstawiono model hurtowni umożliwiający retrospektywną (ang. *late registration*) rejestrację danych, natomiast w [4] – temporalny metamodel hurtowni danych, nazwany przez autorów COMET, dający możliwość rejestracji zmian w schemacie i strukturze hurtowni. Przykład implementacji temporalnej telemetrycznej hurtowni danych można znaleźć w [9]. Wymienione prace dotyczą projektu i implementacji hurtowni danych przy założeniu, że etap analizy dostarczył pełnych wymagań. Niestety w przypadku braków w wymaganiach, szczególnie o temporalnych zależnościach, zaprojektowanie poprawnego modelu hurtowni nie jest możliwe [18]. Model czasu, zaprezentowany w części 3, może zapewnić poprawność gromadzenia temporalnych wymagań.

1.2. Hurtownia danych 2.0 – DW 2.0

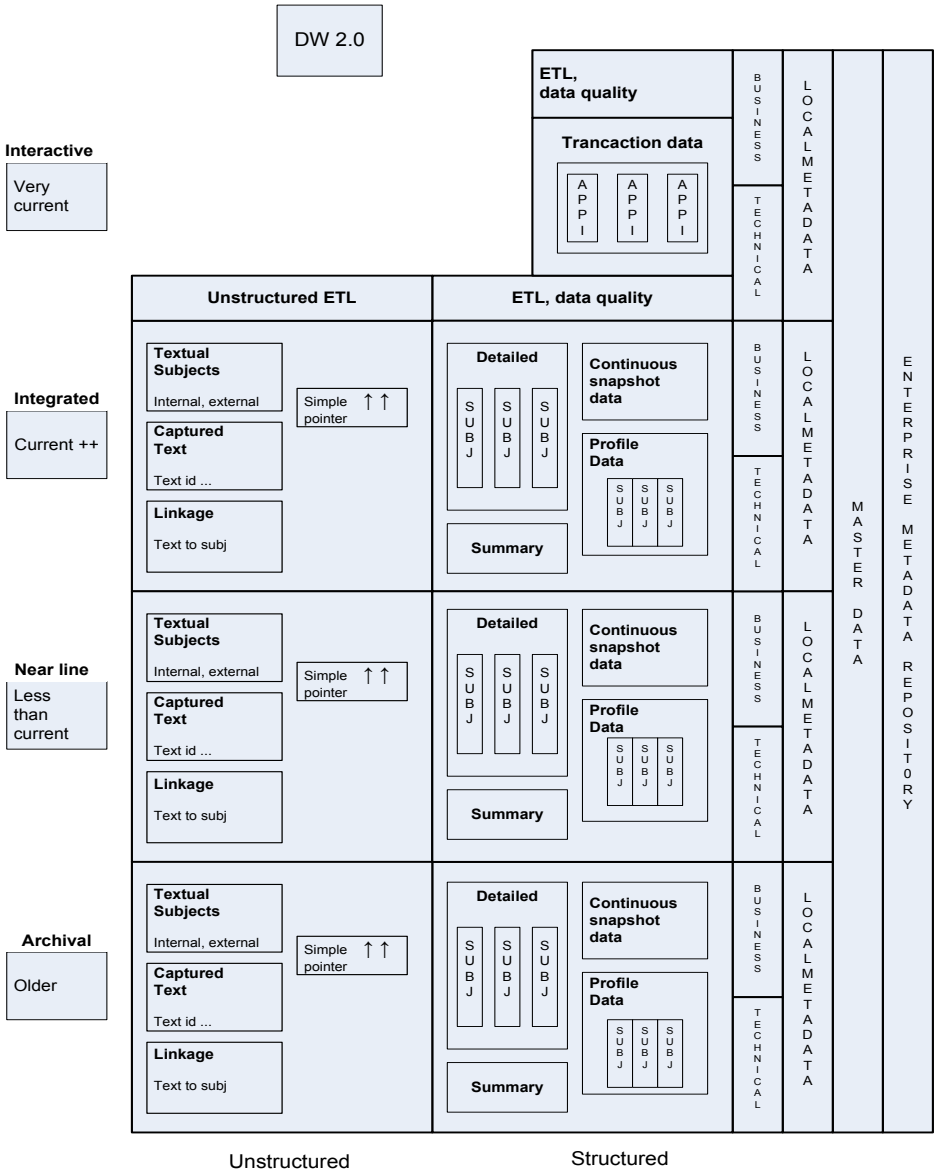
DW 2.0 to zaproponowana przez Williama Harveya Inmona, zwanego „ojcem hurtowni danych", kolejna architektura hurtowni danych [10]. Przedstawia ją rys. 1.

Najważniejsze nowe elementy, wprowadzone w architekturze hurtowni danych 2.0, to:

1. Cykl życia danej (informacji), związany z jej przechodzeniem przez sektory hurtowni 2.0.
2. Metadane.
3. Dwa rodzaje danych: ustrukturyzowane (ang. *data structured*) i nieustrukturyzowane (ang. *data unstructured*).
4. Wprowadzenie wymogu rejestracji zmienności w czasie danych.

Projekty i implementacje hurtowni danych pierwszej generacji zakładały, że wraz z zapisem danej w hurtowni rozpoczyna się jej „życie". Zupełnie inne podejście do „życia" danej prezentuje DW 2.0. Wprowadza się cztery etapy cyklu życia danej, odpowiadające czterem sektorom hurtowni. Pierwszy to sektor interaktywny (ang. *interactive sector*), gdzie dana podlega sprawdzeniu, modyfikacji i integracji, aby następnie przejść do drugiego etapu swojego cyklu, czyli sektora zintegrowanego (ang. *integrated sector*). W sektorze tym dana jest

przechowywana przez cały okres jej biznesowej przydatności. Następnie, kiedy jest wykorzystywana rzadko lub wcale (np. dane urzędów skarbowych tracą ważność po 5 latach), zostaje przeniesiona do jednego z dwóch kolejnych (trzeciego lub czwartego) sektorów.



Rys. 1. Struktura danych hurtowni DW 2.0 [10]
Fig. 1. The structure of data within DW 2.0 [10]

Trzeci sektor (ang. *near line sektor*) tworzony jest w przypadku bardzo dużych hurtowni danych i stanowi swojego rodzaju rozszerzenie sektora zintegrowanego; przechowuje się w nim dane, do których dostęp musi być zapewniony, a dane te już nie są często wykorzystywane. Czwarty, ostatni to sektor archiwalny (ang. *archival sector*), gdzie przechowuje się dane, których prawdopodobieństwo wykorzystania jest bardzo niskie. W sektorze tym kończy się cykl życia danej.

Czas przechowywania danych w poszczególnych sektorach uzależniony jest przede wszystkim od wymagań użytkownika i potrzeb biznesowych, często wynikających z przepi-

sów prawa. Powiązanie z czasem wszystkich danych w hurtowni 2.0 jest wymagane, a sposób podejścia do określenia czasu zależy od sektora, w którym dana się znajduje. W sektorze interaktywnym, który jest swojego rodzaju przedłużeniem systemu transakcyjnego, ważny jest jedynie czas transakcji i udostępniania danej, atrybut związany z czasem nie jest elementem rekordu. Zupełnie inna sytuacja zachodzi w przypadku pozostałych trzech sektorów. Czas jest rozpatrywany w różnych aspektach i wymiarach, może być określony jako moment (punkt w czasie) lub przedział, i stanowi jeden z atrybutów budujących klucz. W przypadku zmiany jakiejkolwiek wrażliwej temporalnie wartości atrybutu, musi zostać wpisany kolejny rekord z nowymi wartościami. W ten sposób tworzy się swojego rodzaju łańcuch zależnych od czasu (temporalnych) informacji (stanów), którego elementami są kolejno zapisywane rekordy. Początek pierwszego elementu tego łańcucha może mieć miejsce w czasie nieskończenie przeszłym, a ostatni kończyć się w czasie nieskończenie przyszłym, ale równie dobrze początki i końce przedziałów mogą być dokładnie określone, a miara czasu wynika tylko i wyłącznie z wymagań biznesowych. Oczywiste jest, że w rzeczywistych systemach rzadko kiedy wartością jest nieskończoność, jednak może to być wartość nieokreślona lub umowna.

2. Prezentacja temporalnej wiedzy

Jak wspomniano wcześniej, niewystarczające zamodelowanie w fazie projektowania danych i zdarzeń wrażliwych na zmiany w czasie skutkuje w okresie produkcyjnym niewiarygodnymi wynikami analiz czy raportami wygenerowanymi z baz i hurtowni [18]. Narzędzia takie jak UML [21] nie dostarczają odpowiednich mechanizmów. W [13] można znaleźć metamodel temporalnej bazy danych, jednak będzie on trudny do zrozumienia dla biznesowego użytkownika i do zapisu na etapie analizy. Rodzi się więc pytanie: jakie metody zapisu zastosować, aby pomogły one zapisać pełną temporalną wiedzę? Już w połowie lat 80. XX w. badacze zaproponowali wykorzystanie temporalnej logiki w informatyce. W [5, 8, 15] autorzy wskazują na następujące dziedziny jej wykorzystania:

- bazy danych – w zakresie zarządzania, modyfikacji i wnioskowania zależnego od czasu,
- specyfikacja, kodowanie, weryfikacja oprogramowania współbieżnego,
- analiza tekstów w językach naturalnych,
- w zakresie sprzętowym, np. VLSI,
- systemy rozproszone,
- planowanie,
- systemy czasu rzeczywistego,
- jako część programowania w logice.

2.1. Przykład temporalnej logiki

Jedną z pierwszych dziedzin zastosowania temporalnych logik była sztuczna inteligencja, gdzie zostały one wykorzystane jako jeden ze sposobów reprezentacji wiedzy [8]. Logiki te mogą być podstawą do stworzenia modelu czasu wspierającego gromadzenie temporalnych wymagań na etapie analizy.

Temporalna logika [7] stanowi rozszerzenie logiki klasycznej przez dodanie temporalnych operatorów. Przykładem są trzy podstawowe temporalne operatory: „○"– w kolejnym momencie, „□" – w każdym przyszłym momencie, „◊" – w jakimś przyszłym momencie. Oznacza to, że również wykorzystywane są klasyczne operatory logiczne.

I tak, liniową temporalną logikę przedstawia się jako typową strukturę Kripkego:

$$M = \langle S, R, \pi \rangle, \tag{1}$$

gdzie: S – zbiór dostępnych momentów (stanów lub zdarzeń); R – temporalne relacje; $\pi : S \mapsto \mathbb{P}$ – odwzorowuje każdy moment bądź zdarzenie lub stan dla danego momentu do zbioru zdań prawdziwych. $\mathbb{P}$ to operator, który dla danego zbioru X generuje zbiór wszystkich jego podzbiorów, np. $\mathbb{P}(\{a,b\}) = \{\emptyset,\{a\},\{b\},\{a,b\}\}$.

Ponieważ temporalna logika liniowa jest dyskretna można założyć, że jest izomorficzna ze zbiorem liczb naturalnych i model przyjmuje uproszczoną postać:

$$M = \langle \mathbb{N}, \pi \rangle, \tag{2}$$

gdzie $\pi : \mathbb{N} \mapsto \mathbb{P}$ odwzorowuje każdą liczbę naturalną, przedstawiającą jakiś moment na zbiór zdań prawdziwych w danym momencie.

Szczegółowy opis semantyki logiki temporalnej można znaleźć w [7, 8].

W tabeli 1 przedstawiono dopuszczalne formuły temporalnej logiki liniowej; ψ i φ odpowiadają zdaniom logiki klasycznej.

Tabela 1

Temporalne operatory i formuły

Formuła	Znaczenie
○φ	Φ jest prawdą w kolejnym momencie
□φ	Φ jest prawdą we wszystkich momentach w przyszłości
◊φ	Φ jest prawdą w jakimś momencie
ΦUψ	Φ trwa i jest prawdziwe (w teraźniejszości i przyszłości), dopóki ψ jest prawdziwe
φWψ	Φ trwa i jest prawdziwe, dopóki ψ nie stanie się prawdziwe

Przykłady formuł:

Φ – student studiuje na semestrze S,

Ψ – student zaliczył wymogi semestru S.

Formuła (student studiuje na semestrze S) W (student zaliczył wymogi semestru S) jest poprawna, czyli WFF (ang. *well-formed formula*), i prawdziwa.

Kolejny, bardziej złożony przykład formuły typu WFF to:

(student zalicza semestr S) ⇔ ○((student studiuje na semestrze S) W (student zaliczył wymogi semestru S)).

W latach 80. przez Moszkowskiego [15, 16] została zaproponowana temporalna logika interwałów ITL (*ang. interval temporal logic*), gdzie zostały przyjęte podobne temporalne operatory jak w liniowej temporalnej logice, przedstawionej wcześniej.

3. Model czasu

Prace [2, 7, 15, 16] stały się inspiracją dla opracowania modelu czasu, wspierającego gromadzenie informacji temporalnych dla baz danych i hurtowni zgodnych z paradygmatem 2.0. Model tenprzewiduje dwa podstawowe, elementarne temporalne typy: punkt (ang. *point*) i interwał (przedział, ang. *interval*), które mogą tworzyć bardziej złożone struktury, np. łańcuchy. Przyjęta konwencja oparta została na rachunku predykatów pierwszego rzędu i klauzuli Horna z wykorzystaniem notacji języka Prolog, co pozwala na pewną prostotę i zrozumienie zapisanych temporalnych informacji i zależności na etapie gromadzenia wymagań również przez przedstawicieli biznesu. Dalej przedstawiono wybrane z modelu czasu podstawowe temporalne typy, które w późniejszych, kolejnych etapach modelowania i projektowania mogą być odpowiednikami zdarzeń, encji czy atrybutów oraz powiązań między nimi.

punkt= określenie;
dtd(dzien_tygodnia,data);
dtg(dzień_tygodnia,godzina);
dg(data,godzina);
d(data);
g(godz);
dt(dzień_tygodnia);
miesiąc(m);
rok(r)

Punkt może być również zdarzeniem opisanym słownie, i właśnie „określenie” odpowiada temu przypadkowi.

data= data(integer,
integer;string,
integer)

określenie, dzień_tygodnia,m = string

godz= godz(integer,
integer)

Przykłady predykatów dla punktów:

p(string) /*p(dzienna_wartość_kursów_walut) oznacza, że publikacja kursów walut jest punktem, czyli w bazie lub hurtowni powinien zostać uwzględniony tylko czas transakcji.

poprzedza(punkt,punkt) /*poprzedza(wypłata_zaliczki,wypłata_wynagrodzenie) obydwie wypłaty są typu punkt i wypłata_zaliczki powinna nastąpić wcześniej niż wypłata_wynagrodzenia)

poprzedza=(punkt,punkt) /*podobnie jak wyżej, ale oba punkty mogą zaistnieć jednocześnie)

równe(punkt,punkt)

Przykłady klauzul dla punktów:

poprzedza(X,Z):-poprzedza(X,Y),poprzedza(Y,Z)

Kolejny typ temporalnej informacji to przedział lub inaczej interwał.

interval= interval (punkt, /*początek
punkt, /*koniec
integer, /*dlugość
string) /*jednostka

i(string) /*i(umowa_cywilno_prawna) oznacza, że każda umowa cywilnoprawna trwa pewien określony czas i musi mieć swój początek i koniec, i z punktu widzenia temporalnej informacji jest interwałem.

rowne(interval,interval) /*równe(umowa_cywilno_prawna,ubezpieczenie_zdrowotne) oznacza to, że czas ubezpieczenia zdrowotnego musi pokrywać się z okresem trwania umowy_cywilno_prawnej, a oba te zdarzenia są typu „interwal", czyli należy zarejestrować ich początek i koniec.

Model czasu zawiera wiele predykatów i klauzul opisujących zależności pomiędzy interwałami oraz pomiędzy punktami a interwałami; przykładami są:

p1p2(X,Y):-poprzedza(początek(X),początek(Y))

/*w prezentowanym przypadku X i Y to interwały, a predykat „początek" określa punkty będące początkami X i Y.

p_wewnatrz_i(punkt, interval)

/*zdarzenie typu punkt musi zajść po rozpoczęciu i przed zakończeniem zdarzenia typu interval, np. p_wewnatrz_i(pobranie_środków_czystości, umowa_o_prace), co odpowiada biznesowej informacji, iż środki czystości pracownik może pobrać tylko i wyłącznie w trakcie umowy o pracę.

Bardziej złożonymi strukturami temporalnymi w modelu są tzw. łańcuchy regularne punktów lub interwałów:

lan_punkt_1=lan_punkt_1(punkt,
integer, /*liczba elementów
integer, /*odległość
string) /*jednostka

lan_punkt_2=lan_punkt_2(punkt,
integer, /*odległość
string) /*jednostka

lan__int_1= lan__int_1(punkt, /*początek łańcucha
punkt, /*koniec łańcucha
interval /*element
integer /*liczba elementów
integer /*odległość między elementami
string) /*jednostka

W dalszej części przedstawiono przykłady zapisu informacji uzyskanej przez konsultanta od przedstawiciela biznesu w trakcie gromadzenia wymagań, która brzmi: „publikacja_wyników_sprzedaży następuje raz w tygodniu o godz. 15:20".
Zapis został wykonany przy użyciu modelu czasu i jest następujący:

p(dzienna_publikacja_wyników_sprzedaży)
dzienna_publikacja_wyników_sprzedaży(godz(15:20))
lan_punkt_2(publikacja_wyników_sprzedaży)
publikacja_wyników_sprzedaży (dzienna_publikacja_wyników_sprzedaży,24,godz).

Zaproponowane zapisy mogą stanowić uzupełnienie opisu wymagań, zgromadzonych i zamodelowanych np. w UML. Model czasu, którego fragment przedstawiono, wymaga dalszych opracowań, szczególnie w zakresie takich struktur temporalnych, jak nieregularne łańcuchy punktów czy interwałów.

4. Podsumowanie

Potrzeba ujednolicenia zapisu wiedzy biznesowej na wczesnym etapie gromadzenia wymagań wynika z licznych obserwacji realizowanych projektów informatycznych, gdzie zauważa się pojawiające się, nieoczekiwanie w procesie projektowania hurtowni danych, problemy, które wynikają z niedostatków zgromadzonych na etapie analizy temporalnych wymagań. W większości przypadków hurtownie realizowane są w drugim lub trzecim etapie projektu informatycznego, systemy produkcyjne z bazami transakcyjnymi zaś w etapie pierwszym. Rejestracja zmian wartości danych w czasie jest kluczowa dla kompletności i poprawności danych w hurtowni. Gromadzenie wiedzy przy użyciu zaprezentowanego modelu czasu może zapewnić poprawność i kompletność zapisów oraz wymusić na konsultantach prawidłowe zbieranie temporalnych wymagań. Dalsze prace będą dotyczyć rozbudowania modelu czasu i zaimplementowania go w języku Prolog, tak aby jednocześnie mógł zostać wykorzystany do sprawdzania kompletności i poprawności zgromadzonych wymagań.

BIBLIOGRAFIA

1. Chmiel J., Stabno M., Wrembel R.: Struktury fizyczne dla wielowersyjnych hurtowni danych, [w:] Kozielski S., Małysiak B., Kasprowski P., Mrozek D. (red.): Bazy danych: Rozwój metod i technologii, WKŁ 2008, s. 37÷49.
2. Cotofrei P., Stoffel K.: First-Order Logic Based Formalism for Temporal Data Mining. Studies in Computational Intelligence (SCI) 6, Springer, 2005, s. 185÷210.
3. Eder J., Koncilia Ch., Morzy T.: A Model for a Temporal Data Warehouse. Proceedings of OES-SEO 2001 Workshop, s 48÷54.
4. Eder J., Koncilia Ch., Morzy T.: The Comet Metamodel for Temporal Data Warehouses. CAISE 2002, LNCS, Vol. 2348, Springer, 2002, s. 83÷99.
5. Gabbay D. M., Hodkinson I. M.: Temporal Logic in Context of Databases, [in:] Copeland J. (ed.): In Logic Reality. Oxford University Press, 1996, s. 69÷87.
6. Golfarelli M., Rizzi S.: A Survey on Temporal Data Warehousing. International Journal of Data Warehousing & Mining, Vol. 5(1), 2009.
7. Fisher M.: An introduction to Practical Formal Methods using Temporal Logic. Wiley, 2011.
8. Fisher M.: Temporal Representation and Reasoning. Handbook of Knowledge Representation. Foundation of Arificiall Intelligence, Elsevier, 2008, s. 513÷550.

9. Gorawski M., Bortlik Ł.: Temporalna telemetryczna hurtownia danych, [w:] Kozielski S., Małysiak B., Kasprowski P., Mrozek D. (red.): Bazy danych: Modele, technologie, narzędzia. WKŁ, 2005, s. 51÷57.
10. Inmon W. H, Strauss D., Neushloss G.: DW 2.0: The Architecture for the Next Generation of Data Warehousing. Morgan Kaufman Series in Data Management Systems, 2008.
11. Kimmball R., Rose M.: The data warehouse Toolkit: The Complete Guide to Dimensional Modeling, second edition. Wiley, 2002.
12. Lori J. B.: Towards a Spatial-Temporal Processing Model, Innovations in Computing Sciences and Software Engineering. Springer Science+Business Media, 2010.
13. Malinowski E., Zimanyi E.: Advanced Data Warehouse Design. Springer, 2009.
14. Mazur Z., Macyna W.: Weryfikacja dynamicznych więzów integralności w temporalnych bazach danych opartych na czasie rzeczywistym, [w:] Kozielski S., Małysiak B., Kasprowski P., Mrozek D. (red.): Bazy danych: Modele, technologie, narzędzia. WKŁ, 2005, s. 213÷221.
15. Moszkowski B.: Executing Temporal Logic Programs. Camridge University Press, 1986.
16. Moszkowski B.: Compositional Reasoning Using Interval Temporal Logic and Tempura. In Compositionality: The Significant Difference. LNCS, Vol. 1536, Springer, 1998, s. 439÷464.
17. Morzy T., Wrembel R.: On Querying of Multiversion Data Warehouse. ACM Seventh International Workshop on Data Warehousing and OLAP, Washington DC, s. 92÷101.
18. Prakash N., Gosain A.: An approach to engineering the requirements of data warehouses. Requirements Eng., Vol. 13, Springer, 2008, s. 49÷72.
19. Rizzi S., Golfarelli M.: What Time is it in the Data Warehouse? Proceedings of 8th International Conference on Data Warehousing and Knowledge Discovery (DaWaK 2006), LNCS, Vol. 4081, s. 134÷144.
20. Silberschatz A., Korth H. F., Sudarshan S.: Database System Concepts, 6th edition. McGraw-Hill Higher Education, 2010.
21. Soden M., Eichler H.:Temporal Extensions of OCL Revisited. ECMDA-FA 2009, LNCS Vol. 5562, Springer, 2009, s. 190÷205.
22. Todman Ch.: Projektowanie hurtowni danych. Helion, 2011.
23. Wrembel R., Bębel B.: Metadata Management in Multiversion Data Warehouse. Journal on Data Semantics VIII, LNCS, Vol. 4380, Springer, 2007, s. 118÷157.

Wpłynęło do Redakcji 15 stycznia 2012 r.

Abstract

Time is a key feature in the data warehousing systems. The registration of all changing values at data's life cycle requires a special approach in modeling and implementation of data warehouse processes. There are many research works discussing special data warehouse model and structures. They discuss a registration of historical and time-changing data problems, the examples are the model for retrospective (late) registration or multiversion data warehouses. The new paradigm DW 2.0 (figure 1) also takes into consideration temporal relations at data warehouse. The effective and correct data warehousing systems depend on gathering all temporal relations and requirements from the business at the very beginning of the creation of data warehouse. The time model is presented as a tool for gathering temporal requirements. The base of this model are temporal logics often used in computer science. The assumptions of the temporal linear logic were described (formulas 1,2). The time model contains such fundamental structures as point, interval, regular and irregulars chains of points or intervals. Furthermore some temporal predicates and clauses are presented as well as the example of using the time model in gathering requirement process.

Adres

Ewa MAGIERA: Uniwersytet Śląski w Katowicach, Instytut Informatyki, ul. Będzińska 39, 41-200 Sosnowiec, Polska, ewa.magiera@us.edu.pl.