AUTOMATYKA • 2009 • Tom 13 • Zeszyt 3

Kamil Kuliberda*, Tomasz M. Kowalski*, Jacek Wiślicki*, ***,
Radosław Adamus*, ***, Michał Meina**

Integracja oraz indeksowanie rozproszonych zasobów danych w technologii „data grid”****

1. Wprowadzenie

Grid to nowatorska technologia szeroko badana przez akademickie oraz komercyjne instytucje. Przez ostatnich pięć lat corocznie, na świecie odbywa się ponad dziesięć konferencji poświeconych tematyce gridowej. Rozwiązania gridowe są nie tylko ważne w sferze badawczej, obecnie ich znaczenie jest niemal takie samo w sferze biznesowej. Do tej pory powstało kilka różnych rozwiązań systemów gridowych współpracujących z różnymi źródłami danych jak i wspierających różne procesy biznesowe jednak nadal są one niekompatybilne ze sobą nawzajem. Artykuł ma na celu przybliżyć czytelnikowi badania, których celem jest opracowanie ogólnej architektury gridowej wspierającej potrzeby biznesowe w przetwarzaniu ogromnych ilości danych przechowywanych w heterogenicznych bazach danych.

Niniejszy artykuł prezentuje architekturę *data grid* (DG), w której przetwarzane są dane strukturalne. Architektura DG opisuje aspekty techniczne przetwarzania danych w *wirtualnym repozytorium* (WR) (*virtual repository*), które jest kompletną technologią dla przeźroczystego przetwarzania danych dostępnych również w różnych formach – postaciach biznesowych dla użytkowników. Wirtualne repozytorium zostało stworzone na podstawie *podejścia stosowego do baz danych* (*Stack Based Approach to databases*) w skrócie SBA oraz języka do baz danych SBQL – zaimplementowanym w prototypowej obiektowej

* Katedra Informatyki Stosowanej, Politechnika Łódzka
** Wydział Matematyki i Informatyki Uniwersytetu Mikołaja Kopernika w Toruniu
*** Stypendysta projektu „Innowacyjna dydaktyka bez ograniczeń – zintegrowany rozwój Politechniki Łódzkiej – zarządzanie uczelnią, nowoczesna oferta edukacyjna i wzmacnianie zdolności do zatrudniania, także osób niepełnosprawnych” współfinansowany przez Unię Europejską w ramach Europejskiego Funduszu Społecznego
**** Praca naukowa finansowana ze środków na naukę w latach 2008/2009 jako projekt badawczy nr N516 383334

bazie danych ODRA. Wirtualne repozytorium posiada również wbudowaną platformę transportową peer-to-peer (TP) działającą jako tzw. warstwa pośrednia (*middle-layer*) wspomagająca wymianę danych (poprzez wewnętrzne indeksowanie) między poszczególnymi węzło-zasobami gridu. Omawiane rozwiązanie dedykowane jest dla rozproszonych, heterogenicznych zasobów danych, które w nieskomplikowany sposób mogą być wirtualnie powiązane w jeden scentralizowany, jednorodny byt – grid bazodanowy [1]. Do szybkiego dostępu do dużej ilości danych dostępnych z tak skonstruowanego gridu wykorzystywany jest w WR *indeks rozproszony* (*distributed index*).

Reszta artykułu jest zorganizowana w następujący sposób. W sekcji 2 prezentujemy podstawowe funkcjonalne elementy WR i DG, dalej omówione są szczegóły SBA oraz implementacji ODRA oraz mechanizm ulotnego indeksowania.

2. Wirtualne repozytorium oraz data grid

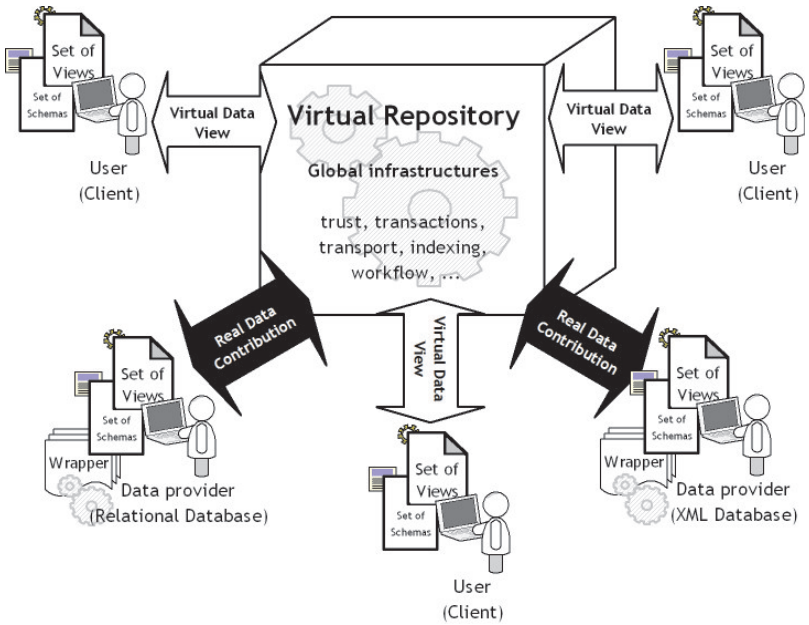
Przetwarzanie danych w WR obejmuje kilka złożonych problemów. Jednym z nich jest aktualizacja wirtualnych danych widocznych poprzez WR – stan badań z tej dziedziny nie jest jeszcze kompletny i praktycznie niezbadany. Podobna problematyka tyczy się budowy globalnej infrastruktury bezpieczeństwa zbudowanej w obrębie WR, konsolidującej mechanizmy spadkowe oraz globalny mechanizm transakcji. Kolejny problem związany jest z wydajnością przetwarzania danych a w szczególności z optymalizacją zapytań języka zorientowanego obiektowo jako interfejsu WR.

Problemy uwidocznione powyżej inspirują badania nad stworzeniem ogólnych metodologii, środowisk, narzędzi i języków, które mają wspomagać tworzenie sieci grid (biznesowo zorganizowanej jako WR), ukierunkowanej na stworzenie konkretnego scenariusza integracji. Proponowanym przez nas rozwiązaniem jest wykorzystanie architektury przedstawionej na rysunku 1 oraz techniczne rozwiązanie poszczególnych jej następujących składników:

- budowa na podstawie kontraktu biznesowego modelu kanonicznego i schematu odpowiadającego wymaganiom globalnego użytkownika WR (uwzględniając przy tym regulacje prawne, standardy oraz działanie poszczególnych użytkowników); model i schemat powinien zostać wyrażony w języku łatwo interpretowalnym maszynowo jak i przez człowieka;
- budowa lokalnych modeli i schematów poszczególnych dostawców danych i usług, które w ten sposób wyrażają swoje zasoby w terminologii globalnego modelu;
- budowa scenariusza integracji na podstawie lokalnych schematów i globalnego schematu, uwidaczniającego zależności między poszczególnymi użytkownikami (m.in. redundancję, replikacje, zależność biznesową itp.);
- stworzenie pośredniej warstwy transportowej ponad różnymi protokołami transportowymi oraz architekturą sieci fizycznej (NAT, firewalle itd.); ten moduł powinien również umożliwiać w sposób przezroczysty utrzymanie aktualnego stanu WR – szczególnie jego zawartości danych;

– realizacja narzędzi do pracy ze zdalnymi obiektami, w szczególności udostępniane w sposób przezroczysty lokalnie dostępnych danych jako dane globalne dostępne dla wszystkich współużytkowników gridu;
– opracowanie założeń oraz realizacja osłon (*wrappers*) – zewnętrznych mechanizmów pozwalających na udostępnianie danych przez ich poszczególnych dostawców.

Pomimo że literatura zawiera wiele prac poruszających omawianą tematykę, dziedzina ta jest jeszcze we wczesnym etapie badań, a rozwiązania dotychczas wprowadzane w praktyce dalekie są od uniwersalnych i łatwo dostępnych. Nasze badania nad generyczną bazą danych zorientowaną obiektowo skupiły się nad wykorzystaniem języka zapytań oraz mechanizmów *aktualizowalnych perspektyw* [9] w celu zaprojektowania mechanizmu gridowego zawierającego zarówno rozważania teoretyczne jak i praktyczne implementacje wykraczające poza aktualny stan sztuki.



Rys. 1. Idea wirtualnego repozytorium. Użytkownicy mają dostęp do danych w najbardziej odpowiadającej im formie bez wiedzy o rzeczywistej formie danych oraz dostarczających je usługach

Rzeczywista realizacja wskazanej koncepcji zakłada współpracę już istniejących i niezależnych technologii. Podstawowym aspektem jest stworzenie tutaj dobrego projektu platformy dla utworzenia WR, zakładającego użycie mechanizmów utrzymywania heterogenicznych zasobów (np. poprzez użycie mechanizmów osłon lub mapowania schematów), platformy wymiany danych, bezpieczeństwa sieci grid oraz zarządzania WR (w ramach

Platformy Transportowej). Takie rozwiązanie może być fizycznie zrealizowane poprzez współistnienie oraz współpracę oddzielnych mechanizmów (modułów lub komponentów) biorących udział w przetwarzaniu tych samych danych. Na bazie takich narzędzi oraz przy znajomości architektury WR każdy programista będzie w stanie samodzielnie zrealizować odpowiedni moduł dostępu do danych dla istniejącego już gridu.

Nasze podejście zakłada użycie następujących rozwiązań w ramach poszczególnych komponentów:

- *wirtualne repozytorium* (WR) – fizycznie dostępne przez aplikacje opierające się na opisanej architekturze; poszczególne części repozytorium to aplikacje klienta i dostawcy oraz aplikacja zarządzająca gridem;
- *platforma transportowa* (PT) – udostępniająca niezależne środowisko programistyczne, odpowiedzialna za transfer danych oraz przetwarzanie transakcyjne na poziomie komunikacji. Platforma powinna umożliwić nieograniczony fizyczny dostęp do sieci grid klientom i dostawcom danych oraz zapewnić protokoły wymiany danych. PT zorganizowana jest w postaci centralnie zarządzanej sieci peer-to-peer, umożliwiającej m. in. identyfikację kooperujących węzłów, konwencje nazw, połączenia między węzłami, bezpieczeństwo transmisji itd. Ponadto istotnym elementem jest zapewnienie przezroczystego dostępu do zasobów oraz skalowalność platformy, a także uniezależnienie się od sieci fizycznej TCP/IP. Wszystkie te założenia zostały zaimplementowane przy użyciu multiprotokołowej, programowalnej platformy p2p biblioteki JXTA [6].

3. Implementacja SBA i SBQL w silniku bazy danych ODRA

Integracja rozproszonych zasobów wymaga zastosowania elastycznej platformy spełniającej następujące założenia:

- generyczny model danych, zdolny do wyrażenia szerokiej gamy możliwych modeli źródeł danych;
- pełnoprawny język programowania ogólnego przeznaczenia zintegrowany z wysokopoziomowym językiem zapytań;
- wirtualne perspektywy z możliwością wyrażenia metody aktualizacji, umożliwiające przede wszystkim wyrażenie lokalnych zasobów w ramach globalnego modelu, budowanie globalnej aplikacji poprzez integracje lokalnych zasobów i dostosowywanie globalnych zasobów dla potrzeb konkretnych klientów.

Wyżej wymienione funkcjonalności są trudne do osiągnięcia przy użyciu aktualnych rozwiązań, języków programowania oraz wspólnych modeli danych. Niemniej jednak podejście stosowe (*Stack Based Approach*) do języków zapytań/programowania [7, 8] umożliwia stworzenie platformy z pełnym zestawem narzędzi koniecznych do realizacji opisywanego rozwiązania. SBA umożliwia wyrażenie wszystkich popularnych koncepcji pochodzących z rozważań nad obiektowymi bazami danych i wprowadza kilka nowych elementów, wcześniej nie znanych: np. dynamiczne role obiektów czy wirtualne aktualizowalne perspektywy.

Język zapytań oparty o podejście stosowe nazywany jest SBQL (*Stack Based Query Language*) [7, 8], w którym nie występuje rozróżnienie na wyrażenia języka programowania oraz na konstrukcje deklaratywne (zapytania) przetwarzające dane. SBQL jest zdefiniowany na bardzo ogólnym modelu danych z zachowaniem zasady wewnętrznej identyfikacji obiektów.

SBQL ponadto udostępnia szereg imperatywnych konstrukcji i mechanizmów, tj.: instrukcje sterujące, procedury, klasy, interfejsy i moduły. Jednym z najważniejszych konstruktorów SBQL jest *aktualizowalna perspektywa obiektowa* [9], przy czym umożliwia ona wyrażenie dowolnego sposobu aktualizacji. Programista/projektant perspektywy ma pełną kontrolę nad procesem aktualizacji, który jest dokonywany na wirtualnym obiekcie, zdefiniowanym przez perspektywę. Nie istnieją żadne restrykcje związane z operacjami dozwolonymi. Definicja perspektywy może być zagnieżdżana oraz zawierać procedury oraz zmienne.

Język SBQL został zaimplementowany w ramach prototypowego silnika bazodanowego ODRA (Object Database for Rapid Application Development) [10]. Celem tego projektu jest stworzenie zorientowanej obiektowo platformy do tworzenia aplikacji.

4. Rozproszony indeks w wirtualnym repozytorium

Indeks jest sztuczną (redundantną) strukurą bazodanową utrzymywaną po stronie serwera, pozwalającą na szybszy dostęp do obiektów (rekordów) pasujących do zadanego kryterium. Administrator bazy danych zarządza pulą indeksów generując nowe bądź usuwając stare zależnie od potrzeb. Niewątpliwą zaletą indeksów jest ich relatywnie mały rozmiar (w porównaniu do całej bazy danych) i jednoaspektowe wyszukiwanie, które czyni organizację indeksu niezwykle prostą.

Implementacja indeksowania w WR zorientowanego obiektowo wymaga zastosowania standardowych technik indeksowania dostępnych lokalnie i technik dedykowanych środowisku rozproszonemu. To podejście nie jest jeszcze dokładnie zbadane i wymaga wprowadzenia nowych rozwiązań. Autorzy skupili się na rozwiązaniu *Scalable Distributed Data Structure* (SDDS) [4] jako punkcie wyjściowym do organizacji rozproszonego indeksu, po to, aby optymalnie wykorzystać zasoby WR (opis technik optymalizacja zapytań dla lokalnych i rozproszonych indeksów z uwzględnieniem ich przezroczystości zostaje w tekście celowo pominięty z uwagi na jego złożoność).

Istotną właściwością systemu rozproszonego upraszczającą projektowanie oraz zarządzanie danymi jest właściwość przezroczystości dostępu. Identycznie sytuacja przedstawia sie z indeksowaniem. Przykładowo używając relacyjnych SZBD (Systemów Zarządzania Bazą Danych) programista nie musi być świadom istnienia indeksów z racji tego, iż są one używane automatycznie podczas ewaluacji zapytania. Dzięki temu administrator bazy danych może dowolnie generować nowe indeksy i usuwać istniejące bez konieczności dokonywania zmian w aplikacji.

Informacja o indeksach dostępnych w lokalnych składach danych nie musi być dostępna na poziomie globalnego schematu. Zapytanie użytkownika gridu podczas procesu przepisywania w wielu przypadkach może być dekomponowane na podzapytania i wysłane do poszczególnych udziałowców gridu. Takie podzapytanie odnosi się jedynie do lokalnych danych i podczas jego ewaluacji może być optymalizowane poprzez przepisywanie z wykorzystaniem lokalnego metaschematu i lokalnej puli indeksów.

Istnieje wiele zalet stosowania strategii lokalnych indeksów w środowisku rozproszonym, m.in.:

- dane i indeksy są składowane w jednym miejscu więc wymagane są tylko lokalne metody zachowania spójności między nimi;
- globalna optymalizacja zapytania jest podzielona między użytkowników gridu na poziomie globalnym i lokalnym; globalny optymalizator nie musi brać pod uwagę lokalnej optymalizacji, a z drugiej strony lokalna indeksacja dokonuje się niezależnie od globalnej infrastruktury.

Z racji tego, że lokalne indeksowanie nie zawsze jest w stanie zapewnić najefektywniejszej mocy obliczeniowej gridu stosujemy dodatkowo globalną strukturę indeksującą. SDDS jest skalowanym rozproszonym indeksem zaprezentowanym w [4]. SDDS składuje pozycje indeksu w pliku rozproszonym w sieci. Właściwości SDDS czynią z niego dobrą technikę do indeksowania lokalnych bądź globalnych zasobów w systemie gridowym. SDDS wykorzystuje LH*, które jest uogólnieniem techniki liniowego haszowania do wykorzystania z pamięcią rozproszoną bądź plikami na dysku. Zastosowanie SDDS wprowadza następujące właściwości do rozproszonego indeksu:

- decentralizację;
- umożliwia równoległą i rozproszoną ewaluację zapytań;
- udostępnia przezroczystość współbieżności;
- skalowalność – nie zakłada żadnych ograniczeń na swój rozmiar bądź pojemność;
- plik SDDS rozprzestrzenia się na nowe serwery, jeśli optymalne obciążenie aktualnego serwera zostało przekroczone;
- aktualizacja indeksu nie wymaga odświeżenia danych na wszystkich serwerach i klientach;
- nieduży nakład transferu w sieci – w ogólności niewielką ilość wiadomości wymienianymi między serwerami (jedną dla losowej operacji dodania i dwie dla wyszukiwania po kluczu)

Wszystkie te cechy wskazują, że SDDS przewyższa efektywnością indeks scentralizowany lub jakąkolwiek inną statyczną strukturę – stąd planowana jest jego implementacja w aktualnie projektowanym prototypie systemu gridowego [1, 2, 3].

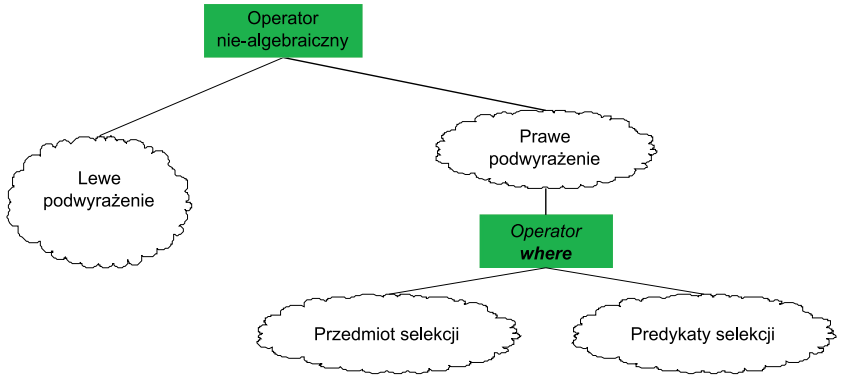
Integracja rozproszonych zasobów w postać gridu, w sposób przedstawiony w następnym rozdziale wymaga optymalizacji z racji dużej ilości danych poddawanych procesowaniu. Tak jak zostało pokazane w [5], efektywne metody optymalizacyjne mogą zostać łatwo

wprowadzone dzięki podejściu stosowemu. Indeksowanie odgrywa bardzo istotną rolę w optymalizacji zapytań ponad globalnym schematem, dzięki czemu może z łatwością zmniejszyć obciążenie całego systemu gridowego.

5. Techniki optymalizacji przetwarzania zdalnych obiektów w ramach platformy transportowej oparte o indeksowanie

W tym rozdziale omówiona jest opracowana przez autorów *technika ulotnego indeksowania*. Polega ona na podstawowych elementach architektury systemu indeksowania, tj. optymalizatora stosującego indeksy i modułu zarządzania indeksami, z wyłączeniem automatycznej aktualizacji indeksu, której osiągniecie w heterogenicznym, rozproszonym środowisku wirtualnego repozytorium jest bardzo trudnym zadaniem, a nawet w praktyce często niemożliwym. Pominięcie mechanizmów automatycznej aktualizacji indeksu jest możliwe, gdyż w odróżnieniu od normalnych indeksów ulotny indeks jest materializowany podczas wykonywania zapytania. Dzięki takiemu podejściu można tę technikę stosować do danych wirtualnie udostępnionych przez perspektywy SBQL, także do zasobów wirtualnego repozytorium.

Opracowana technika jest skuteczna w przetwarzaniu złożonych zapytań, w których indeks jest wywoływany więcej niż jeden raz. Taka sytuacja jest przedstawiona na rysunku 2. Optymalizowane wyrażenie **where** powinno znajdować się po prawej stronie operatora nie-algebraicznego, którego lewe podwyrażenie zwraca kolekcję. Przedmiot selekcji powinien być niezależnym zapytaniem, a wartości predykatów selekcji zależne od wspomnianego operatora. Reguły te stanowią metodykę warunkującą stosowanie ulotnego indeksu.



Rys. 2. Drzewo zapytania podatnego na zastosowanie ulotnego indeksu

Najważniejsza cecha odróżniająca ulotny indeks od normalnego indeksu dotyczy ograniczeń na definicję wartości niekluczowych. Regularne indeksy mogą być zakładane tylko na kolekcjach obiektów określonych przez proste wyrażenia ścieżkowe. Takie ograniczenie jest spowodowane możliwościami mechanizmu automatycznej aktualizacji indek-

su, więc nie dotyczy ono techniki ulotnego indeksowania. Definicja wartości niekluczowej ulotnego indeksu może być dowolnym wyrażeniem zwracającym:

- referencje zdalnych obiektów,
- wirtualne obiekty (określone przez perspektywy),
- zdalne obiekty w ramach platformy transportowej,
- bindery i literały.

Podstawowym założeniem dotyczącym definicji klucza i wartości niekluczowych jest determinizm, tj. że wartości niekluczowe i kluczowe pozostają niezmienne dopóki dane użyte do ich wyznaczenia się nie zmienią.

Znaczącą zaletą techniki ulotnego indeksowania jest jej praktyczność z punktu widzenia integracji, rozproszonych i heterogenicznych zasobów w ramach wirtualnego repozytorium.

6. Wnioski

W niniejszym artykule zostały zaprezentowane wyniki pracy dotyczącej integracji oraz indeksowania rozproszonych obiektów w gridowej obiektowej bazie danych w ramach wirtualnego repozytorium i architektury data grid. W artykule przedstawiono model gridu baz danych w jakim wykorzystano prezentowane podejście do indeksowania danych, szczegóły podejść do indeksowania rozproszonego oraz nowatorską technikę ulotnego indeksowania danych rozproszonych, która została zaimplementowana w całości, a jej działanie potwierdzone testami.

Literatura

[1] Habela P., Kaczmarski K., Kozankiewicz H., Lentner M., Stencel K., Subieta K., *Data-Intensive Grid Computing Based on Updateable Views*. ICS PAS Report 974, June 2004.

[2] Kozankiewicz H., Stencel K., Subieta K., *Implementation of Federated Databases through Updateable Views*. Proc. EGC 2005 – European Grid Conference, Springer LNCS, 2005.

[3] Kozankiewicz H., Stencel K., Subieta K., *Integration of Heterogeneous Resources through Updatable Views*. ETNGRID2004 WETICE2004, Proceedings published by IEEE.

[4] Litwin W., Nejmat M.A., Schneider D.A., *LH*: Scalable, Distributed Database System*. ACM Trans. Database Syst., 21(4), 1996, 480–525

[5] Plodzien J., *Optimization Methods in Object Query Languages*. IPIPAN, Warszawa, 2000 (Ph.D. Thesis).

[6] Project JXTA Community: http://www.jxta.org *(home-page)*.

[7] Subieta K., *Stack-Based Approach (SBA) and Stack-Based Query Language (SBQL)*. http://www.ipipan.waw.pl/~subieta, Description of SBA and SBQL, 2006.

[8] Subieta K., *Theory and Construction of Object-Oriented Query Languages*. Editors of the Polish-Japanese Institute of Information Technology, 2004 (in Polish).

[9] Kozankiewicz H., *Updateable Object Views*. PhD Thesis, 2005, http://www.ipipan.waw.pl/~subieta/ -> Finished PhD-s -> Hanna Kozankiewicz.

[10] Lentner M., Subieta K., *ODRA: A Next Generation Object-Oriented Environment for Rapid Database Application Development*. http://www.ipipan.waw.pl/~subieta/artykuly/ODRA%20paperpl.pdf, 2006.