STUDIA INFORMATICA 2013
Volume 34 Number 2B (112)

Wojciech KIJAS
Politechnika Śląska, Instytut Informatyki

ONTOLOGIA OWL JAKO SKŁADNICA DANYCH DLA APLIKACJI .NET

Streszczenie. W artykule podjęta została próba stworzenia biznesowej aplikacji .NET, w której składnicę danych stanowi ontologia OWL. Przeprowadzona została analiza możliwych rozwiązań takiego problemu, wraz z porównaniem funkcjonalności każdego z nich. Dla porównania przeprowadzono próbę zastąpienia relacyjnej bazy danych ontologią OWL w istniejącej aplikacji biznesowej .NET, z użyciem znalezionych rozwiązań.

Słowa kluczowe: Semantic Web, ontologia OWL, aplikacja .NET, relacyjna baza danych

OWL ONTOLOGY AS A DATA REPOSITORY FOR .NET APPLICATION

Summary. In this paper we made an attempt to create .NET business application with OWL ontology as data repository. We analyzed possible solutions of this problem with comparing the functionality of each. For comparison purposes, we tried to replace relational database with OWL ontology in an existing business application using previously found solutions.

Keywords: Semantic Web, OWL ontology, .NET application, relational database

1. Wprowadzenie

W ostatniej dekadzie można zaobserwować stosunkowo powolny, aczkolwiek ciągły i konsekwentny, rozwój Sieci Semantycznej (Semantic Web). Sieć Semantyczna to wspólny ruch kierowany przez organizację World Wide Web Consortium (W3C), która zajmuje się ustanawianiem standardów tworzenia treści internetowych [1]. Poprzez zachęcanie do włą-

czenia treści znaczeniowej w kodzie stron internetowych, Semantic Web ma na celu przekształcenie obecnego Internetu zdominowanego przez niestrukturalne i półstrukturalne dokumenty w „sieci danych".

Jednak aktualnie Semantic Web to pojęcie dotyczące nie tylko stron internetowych. Idea dołączania treści znaczeniowych znajduje również zastosowanie dla zbiorów danych, które niekoniecznie są upublicznione w Internecie. Kluczem do prawidłowego funkcjonowania przedsiębiorstwa jest odpowiednie zarządzanie danymi, które są przez nie gromadzone, i tworzenie z tych danych użytecznej wiedzy, z której korzystać można przy podejmowaniu decyzji biznesowych. Relacyjne bazy danych nie zawsze odpowiadają swoimi możliwościami na wymagania użytkowników, szczególnie w przypadku systemów informatycznych:

- w których funkcjonowanie niejako wpisane są częste zmiany struktur zapisywanych danych,
- w których obok ustrukturyzowanych danych często używane są również niestrukturalne i półstrukturalne dane, pochodzące między innymi wprost z Internetu,
- w których jednym z wymagań jest udostępnianie danych w Internecie i integrowanie ich z już istniejącymi zasobami.

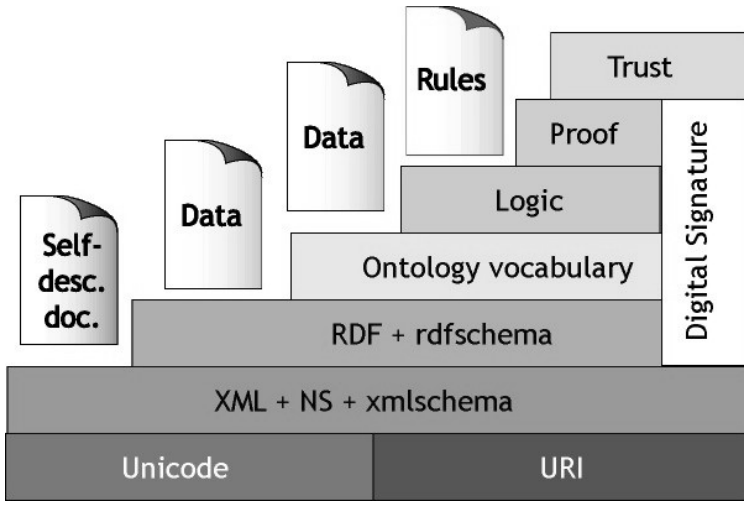
Właśnie tak spracyzowane wymagania systemu informatycznego stały się motywacją do prób mających na celu użycie składnicy danych RDF/OWL w systemach, w których powszechnie stosuje się relacyjne bazy danych.

2. Technologie Semantic Web

W 2001 roku Tim Berners-Lee opublikował artykuł [2], w którym przedstawił wizję utworzenia standardów opisywania treści, gdzie dane będą istniały wraz ze swoją semantyką. W ten sposób dostępne dane będą mogły być przetwarzane nie tylko przez człowieka (jak to miało miejsce w przypadku treści internetowych przed badaniami nad Semantic Web), ale również przez specjalne programy (np. tzw. agenty), które będą potrafiły przetwarzać dane odpowiednio do ich znaczenia. Przedstawiona wizja wymagała opracowania wielu standardów, które pozwalałyby na uniwersalny i elastyczny zapis informacji, definiowanie nowych pojęć, ich atrybutów, a także nowych pojęć i atrybutów na podstawie już istniejących.

2.1. Logiczna struktura technologii Semantic Web

Logiczną strukturę technologii Semantic Web najczęściej przedstawia się za pomocą tzw. torcika Sieci Semantycznej (rys. 1). Pełny opis poszczególnych warstw wykracza poza zakres niniejszego artykułu. Poszczególne warstwy przedstawionej struktury zostały szczegółowo opisane w ramach opracowania [3].



Rys. 1. Torcik Sieci Semantycznej
Fig. 1. Semantic Web Cake

2.2. Języki RDF i OWL

Modelem danych używanym przez Semantic Web jest język RDF (*Resource Description Framework*) [4]. W języku tym dane zapisywane są jako zbiór wyrażeń zbudowanych z trzech części: podmiotu (*subject*), orzeczenia (*predicate*) i dopełnienia (*object/value*). Ze względu na tę trójczęściową budowę, wyrażenia te często nazywane są trójkami (*triples*). Trójki RDF służą do opisu rzeczy (*things*) poprzez tworzenie wyrażeń opisujących ich właściwości. Podmiot to zasób, jaki chcemy opisać, natomiast orzeczenie określa związek, jaki istnieje między podmiotem a dopełnieniem, gdzie dopełnieniem może być zarówno obiekt, jak i wartość typu prostego. Opracowany został również język zwany RDF Schema (RDFS) [5], który zapewnia słownictwo do opisu zasobów, ich właściwości, klas i podklas. Wyrażenia w języku RDFS również zapisywane są w postaci trójek RDF.

Językiem opartym na założeniach RDF i RDFS jest OWL (*Web Ontology Language*) [6]. OWL rozwija i rozszerza RDF i RDFS, dodając kolejne hasła słownikowe, opisujące klasy, fakty na temat tych klas, relacje między klasami lub obiektami i właściwości tych relacji. W dużym uproszczeniu można powiedzieć, że RDF stanowi abstrakcyjny model danych, zaś OWL dodaje nowe słownictwo i możliwości modelowania zaawansowanych zależności wiedzy dziedzinowej, tworzenia ontologii, a na ich podstawie baz wiedzy.

Językiem, dzięki któremu można tworzyć zapytania do danych RDF, jest zdefiniowany przez rekomendację W3C język SPARQL [7].

2.3. Porównanie trójek RDF i relacyjnej bazy danych

Z punktu widzenia znacznej większości systemów informatycznych, relacyjna baza danych jest ich najważniejszym elementem. To w bazach danych najczęściej przechowywane są

wszystkie informacje składowane przez system informatyczny oraz metadane i dane konfiguracyjne. Warto zastanowić się nad tym, jakie korzyści może przynieść zamiana tradycyjnej relacyjnej bazy danych w systemie informatycznym na składnicę danych bazującą na zapisie trójek RDF.

Tabela 1 przedstawia porównanie podstawowych cech relacyjnej bazy danych i ich odpowiedników dla danych zapisanych w postaci trójek RDF.

Tabela 1

Porównanie cech relacyjnej bazy danych i bazy trójek RDF

Cecha	**Relacyjna baza danych**	**Baza trójek RDF/OWL**
Struktura danych	Schemat bazy danych	Wyrażenia RDF/OWL
Dane	Wiersze w tabelach zbudowanych z kolumn	Wyrażenia RDF/OWL
Zależności	Zależności między tabelami, definiowane za pomocą kluczy zbudowanych z kolumn tabeli.	Zależności między klasami, właściwościami i obiektami, definiowane za pomocą wyrażeń RDF/OWL
Unikalność	Klucze tabeli, ograniczenia	URI
Język zapytań	SQL	SPARQL

Sposób zapisu w relacyjnej bazie danych zależy od wcześniej zaprojektowanej struktury bazy danych, zwanej schematem bazy danych. W przypadku danych RDF, sposób zapisu zależy od zdefiniowanej, za pomocą trójek RDF/OWL, struktury bazy wiedzy. Warto tutaj zaznaczyć, że w przypadku relacyjnych baz danych dodanie nowej tabeli czy kolumny jest zupełnie inną operacją niż dodanie wiersza danych. Schemat bazy danych powinien być ostatecznie określony przed rozpoczęciem dodawania danych do bazy. Dla baz wiedzy używany jest ten sam język do opisu struktury bazy wiedzy oraz do zapisu obiektów, a więc struktura może zmieniać się w dowolnym momencie, bez większego wpływu na architekturę systemu informatycznego.

W relacyjnej bazie danych zależności między danymi realizowane są za pomocą kluczy zbudowanych z jednej lub kilku kolumn tabeli. Struktury relacyjnych baz danych znakomicie sprawdzają się w zastosowaniach wykorzystujących proste, niezagnieżdżone dane, jednak do wielu zastosowań okazują się zbyt sztywne. Odwzorowanie złożonych obiektów na struktury relacyjne wiąże się z jednej strony z częściową utratą semantyki danych, a z drugiej ze wzrostem złożoności zapytań odpytujących takie struktury. Zapis danych z wykorzystaniem technologii Semantic Web oferuje możliwość zamodelowania złożonych obiektów bez utraty semantyki, która pozostaje „zaszyta” w definicji klas, właściwości i obiektów. Możliwe jest tutaj definiowanie zależności między klasami (np. rdfs:subClassOf, owl:equivalentClass, owl:disjointWith), zależności między właściwościami (np. rdfs:subPropertyOf, owl:equivalentProperty, owl:inverseOf) oraz zależności między obiektami (np. owl:sameAs, owl:differentFrom, owl:AllDifferent) [8]. Zależności między obiektami mogą być również niezależne od definicji klas. Przykładowo, w danym systemie można określić pojazd jako zasób

o określonej liczbie kół, masie i długości, nie określając takiej właściwości jak kolor. Mimo to, dla konkretnego pojazdu, oprócz liczby kół, masy czy długości, można określić kolor jako czerwony, używając odwołania do słownika określonego na zewnątrz danego systemu [9]. Warto w tym miejscu zauważyć, że tworzenie zależności nie jest ograniczone tylko do jednej ontologii. Definicje nowych klas, właściwości czy obiektów mogą zawierać odwołania do definicji zawartych w innych udostępnionych ontologiach.

3. Baza wiedzy jako alternatywa dla relacyjnej bazy danych

Technologie Semantic Web są wystarczająco elastyczne i uniwersalne, by opisać za ich pomocą dane i ich relacje zaimplementowane w wielu innych językach, bez straty żadnych informacji. Praktycznie każda relacyjna baza danych, model UML czy dokument XML może być w pełni opisany za pomocą języka RDF/ OWL. W każdym przypadku, gdy rozważane jest wykorzystanie jednej z tych technologii, powinno się rozważyć również wykorzystanie RDF lub OWL jako korzystniejszej alternatywy.

Języki RDF i OWL są szczególnie użyteczne przy transportowaniu danych, gdyż łącznie z przenoszonymi danymi można zachować również semantykę relacji. Z tej przyczyny pionierami w stosowaniu tych technologii są społeczności, w których występuje częsta potrzeba wymiany zgromadzonych danych, których format często się zmienia i bardzo trudne do utrzymania są jednakowe standardy słownictwa i nazewnictwa. Społeczności, o których mowa, najczęściej związane są z branżami, takimi jak: służba zdrowia, farmacja, biotechnologia, agencje ochrony środowiska, i to właśnie na ich potrzeby odpowiadają swoimi możliwościami technologie Semantic Web.

Należy pamiętać, że technologie Semantic Web są na etapie rozwoju, na którym relacyjne bazy danych znajdowały się w latach osiemdziesiątych dwudziestego wieku, a więc relacyjne bazy danych mają przewagę wielu lat optymalizacji. Języki, takie jak RDF czy OWL, mają tylko niewielką część użytkowników, narzędzi, forum czy grup dyskusyjnych, jakie mają bazy danych, takie jak Oracle czy SQL Server. Z tego względu większość składnic danych RDF/OWL nie ma wielu cech i możliwości, których dostępność w świecie relacyjnych baz danych jest czymś oczywistym, w szczególności można wskazać na:

- narzędzia do zarządzania wysoką dostępnością danych i kopią zapasową,
- wielopoziomowa kontrola dostępu,
- zaawansowane narzędzia partycjonowania,
- wbudowane procedury i funkcje,
- indeksowanie i optymalizacja zapytań.

Podobnych cech i funkcjonalności można by z pewnością wymienić więcej. Ze względu na zupełnie inny sposób zarządzania danymi, nie jest proste przeniesienie tych funkcjonalności wprost na składnice danych RDF/OWL, lecz niezbędne są projekt i implementacja podobnych rozwiązań od nowa. Żaden z tych problemów nie powstrzymuje jednak programistów od używania danych RDF/OWL w swoich projektach. Od czasu opublikowania specyfikacji języków RDF i OWL przez organizację W3C, powstało wiele zaawansowanych projektów, z powodzeniem wykorzystujących te technologie. Niektóre z nich opisane zostały w ramach opracowania [10].

4. Współpraca Semantic Web i platformy .NET

Znaczna większość aplikacji utworzonych z użyciem platformy .NET, jako składnicy danych, używa relacyjnej bazy danych. Wynika to z faktu, że platforma .NET oferuje, w ramach zestawu podstawowych bibliotek, zbiór komponentów ADO.NET [11], używanych przez programistów celem dostępu do danych. ADO.NET dostarcza programistom odpowiednie biblioteki, umożliwiające dostęp do większości relacyjnych baz danych dostępnych na rynku. Obok standardowej komunikacji z bazą danych za pomocą zapytań SQL, powstało również wiele technik i rozszerzeń, umożliwiających mapowanie obiektowej architektury systemu informatycznego na tabele relacyjnej bazy danych (mapowanie obiektowo-relacyjne) [12], przez co operowanie na danych staje się dużo bardziej intuicyjne, a po utworzeniu odpowiedniego mapowania, wymaga znacznie mniej nakładów programistycznych. Takie udogodnienia dla komunikacji z relacyjnymi bazami danych w platformie .NET powodują, że programiści podejmują decyzję o użyciu relacyjnej bazy danych zbyt pochopnie, bez rozważenia innych technologii (w tym Semantic Web), które mogłyby się okazać znacznie lepiej dostosowane do wielu zastosowań. Bardziej otwarta jest platforma Java, z którą związanych jest wiele więcej społeczności. W konsekwencji to właśnie na platformę Java powstaje większość nowych frameworków i wzorców projektowych, które często są na bardzo wysokim stadium rozwoju i dopracowania, przez co są praktycznie gotowe do użycia.

W te prawidłowości wpisują się również nowe projekty, związane z rozwojem technologii Semantic Web. Znaczna większość tych projektów to implementacje na platformę Java. Na stronie internetowej [13] prowadzony jest wykaz narzędzi wspomagających tworzenie aplikacji Semantic Web. Z listy wszystkich narzędzi przefiltrowano te, które zawierają biblioteki (API) do operowania na danych RDF na platformie .NET, oraz te, które udostępniają trwałą bazę trójek RDF (*triple store*), zaimplementowaną na platformie .NET, którą można odpytywać zapytaniami w języku SPARQL. Wyniki zostały przedstawione w tabeli 2. W przypadku dwóch ze znalezionych projektów (Open Anzo i SemWeb) rozwój został z ja-

kichś przyczyn wstrzymany i zarówno ich strony internetowe, jak i informacje znalezione na forach internetowych sugerują, że platformy te nie są rozwijane. Projekt RDFSharp jest w początkowej fazie rozwoju i mimo bardzo obiecujących planów, zaimplementowane dotychczas funkcjonalności nie są wystarczające.

Tabela 2

Dostępne biblioteki Semantic Web dla platformy .NET

Nazwa biblioteki	API	Baza trójek RDF	Status
dotNetRDF (http://www.dotnetrdf.org/)	x		Stabilna.
IntelliDimension (http://www.intellidimension.com/)	x	x	Stabilna
Open Anzo (http://www.openanzo.org/)	x	x	Nierozwijana. Strona internetowa projektu nie jest dostępna od ponad 2 lat.
RDFSharp (http://rdfsharp.codeplex.com/)	x	x	Projekt przewiduje zarówno API jak i bazę RDF, lecz aktualnie jest w bardzo wczesnym stadium rozwoju. Zaimplementowane są tylko podstawowe biblioteki do operowania na trójkach RDF, brak wsparcia SPARQL.
SemWeb (http://razor.occams.info/code/semweb/)	x	x	Rozwój i wsparcie wstrzymane w 2010 roku.

Biblioteki IntelliDimmension i dotNetRDF to jedyne biblioteki, które są zaprojektowane dla platformy .NET i ich poziom rozwoju jest wystarczający dla użycia w projektach.

4.1. Platforma IntelliDimension

Platforma IntelliDimension oferuje całą gamę narzędzi przeznaczonych do tworzenia aplikacji bazujących na Semantic Web z użyciem platformy .NET. Oferta obejmuje biblioteki do składowania danych semantycznych w postaci trójek RDF (z wykorzystaniem Microsoft SQL Server i integracji z .NET CLR [14]), biblioteki do operowania na danych semantycznych, umożliwiające zadawanie zapytań w języku SPARQL, oraz komponenty warstwy prezentacji dla aplikacji tworzonych w technologiach ASP.NET i Silverlight. Tym, co wyróżnia rodzinę produktów IntelliDimension, jest spójna platforma oparta w całości na technologiach Microsoft (platformę .NET i SQL Server). Jedną z podstawowych wad jest natomiast fakt, że jest to platforma komercyjna. Można otrzymać jedynie darmową licencję na 60-dniowy okres próbny.

4.2. Projekt dotNetRdf

Twórcy projektu dotNetRdf oferują w ramach licencji typu *Open Source* bibliotekę działającą na platformie .NET, zawierającą uniwersalne interfejsy, pozwalające na łączenie z różnymi dostawcami baz trójek RDF, zadawanie zapytań w języku SPARQL i operowanie na danych RDF. Aktualnie biblioteka oferuje klasy do połączenia z siedmioma serwerami danych RDF (AllegroGraph, Dydra, 4store, Fuseki, Stardog, Virtuoso, Sesame) i ich liczba w dalszym ciągu wzrasta. Największą zaletą tej biblioteki jest jej uniwersalność. Klasy obsługujące dostęp do różnych składnic danych RDF implementują wspólne interfejsy, przez co ewentualne zmiany dostawcy w czasie trwania projektu nie sprawiają żadnych problemów. Przedstawiona podstawowa zaleta może okazać się również wadą, gdyż w ramach projektu dotNetRdf nie jest rozwijana żadna autorska składnica danych RDF, do której byłyby zoptymalizowane tworzone biblioteki. Dla celów testowych, jako składnicę danych RDF, z którą współpracują biblioteki dotNetRdf, wykorzystano serwer Sesame [15], czyli jeden z najbardziej popularnych serwerów RDF. Jako repozytorium na serwerze Sesame użyto w podstawowej konfiguracji repozytorium Native Sesame, a także OWLIM-SE (we wcześniejszych edycjach określane nazwą BigOWLIM) [16].

4.3. Użycie klas Java w środowisku .NET

Alternatywnym rozwiązaniem, umożliwiającym oprogramowanie w aplikacji .NET warstwy dostępu do składnicy danych RDF, udostępniającej autorskie API jedynie dla platformy Java, jest próba użycia klas Java w środowisku .NET. Takie rozwiązania stały się możliwe dzięki twórcom projektu IKVM.NET [17], którzy zaimplementowali maszynę wirtualną Javy (*Java Virtual Machine*) dla platformy .NET. Projekt ten udostępnia biblioteki pozwalające na dynamiczne uruchamianie klas napisanych w języku Java, a także na konwertowanie bibliotek Java (jar) na biblioteki .NET (dll). Takie podejście wykorzystali twórcy projektu dotSesame [18], którzy przekonwertowali do bibliotek .NET biblioteki napisane w języku Java przez producenta serwera Sesame.

W ramach pracy nad niniejszym artykułem, celem analizy tego rodzaju podejścia, wykorzystano projekt Sesame Windows Client (SWC) [19], który do swojego działania wykorzystuje technologie opracowane w ramach projektów [17] i [18]. Projekt SWC to aplikacja napisana w technologii .NET, będąca w swym pierwotnym założeniu klientem do serwera Sesame. Z wykorzystaniem aplikacji SWC możliwa jest z jednej strony administracja serwerem Sesame (na przykład tworzenie i usuwanie repozytoriów danych, import i eksport danych), a z drugiej strony aplikacja SWC umożliwia zadawanie zapytań w języku SPARQL i analizę danych RDF zgromadzonych w repozytoriach. W najnowszych wersjach projektu Sesame Windows Client,

obok serwera Sesame dodano również możliwość pracy z innymi serwerami RDF (na przykład Virtuoso [20]), jednak do dostępu do serwerów innych niż Sesame wykorzystywana jest biblioteka dotNetRdf, która jest również jedną z bibliotek testowanych w ramach niniejszej pracy.

5. Implementacja i porównanie znalezionych rozwiązań pod kątem czasu wykonania operacji

Ważnym wyznacznikiem, decydującym o tym, czy dana biblioteka będzie mogła być wykorzystana do zastąpienia relacyjnej bazy danych w aplikacji biznesowej, jest jej szybkość działania przy typowych operacjach na danych, wykonywanych przez użytkownika. Kryterium czasu wykonania operacji przyjęto jako podstawowe w niniejszej pracy. Jako punktu odniesienia użyto relacyjnej bazy danych (MS SQL Server 2008 R2 Express) i działającego w środowisku Windows systemu do zarządzania informacjami o klientach i sprawach prowadzonych w niewielkiej kancelarii prawnej.

Zaimplementowanie warstwy dostępu do danych z wykorzystaniem dostarczonego API, zarówno w ramach projektu dotNetRdf, jak i projektu IntelliDimension, nie sprawiało większych trudności. Obydwie biblioteki udostępniają bardzo przydatne przykładowe aplikacje realizujące podstawowe zadania, których kod można bardzo łatwo przystosować dla własnych potrzeb. W przypadku projektu Sesame Windows Client implementacja dla celów testowych nie była konieczna, gdyż wraz z projektem dostarczona była gotowa, uniwersalna aplikacja, za pomocą której można wykonywać operacje na danych po wcześniejszym połączeniu z serwerem Sesame (dla aplikacji Sesame Windows Client i dotNetRdf wykorzystano ten sam serwer Sesame). Niewielka zmiana w kodzie aplikacji Sesame Windows Client była konieczna jedynie po to, by dodać dokładne wyliczanie czasu wykonania zapytań.

Dla przetestowania szybkości działania poszczególnych API z rzeczywistymi danymi wykonano następujące kroki:

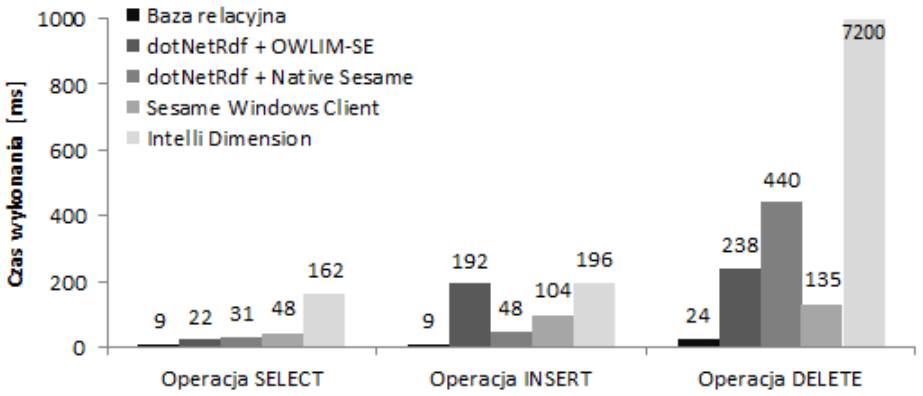
- Przygotowano cztery relacyjne bazy danych, zawierające informację o różnej liczbie klientów (odpowiednio 100, 1 000, 10 000 i 100 000 klientów).
- Przy użyciu narzędzi D2RQ[21] i TopBraid Composer[22] zaprojektowano ontologię OWL na podstawie istniejącego schematu relacyjnej bazy danych.
- Wykorzystując zaprojektowaną ontologię i wcześniej przygotowane dane, utworzono cztery testowe bazy wiedzy, o wielkości odpowiednio około 2 300, 23 000, 230 000 i 2 300 000 trójek RDF.
- Przygotowano zapytania SPARQL i fragmenty kodu, wykorzystujące wymienione wcześniej biblioteki w celu sprawdzenia czasu, jaki poszczególne biblioteki potrzebują na wy-

konanie trzech podstawowych operacji na składnicy danych RDF, a więc odczytu, zapisu i usunięcia danych pojedynczego klienta.

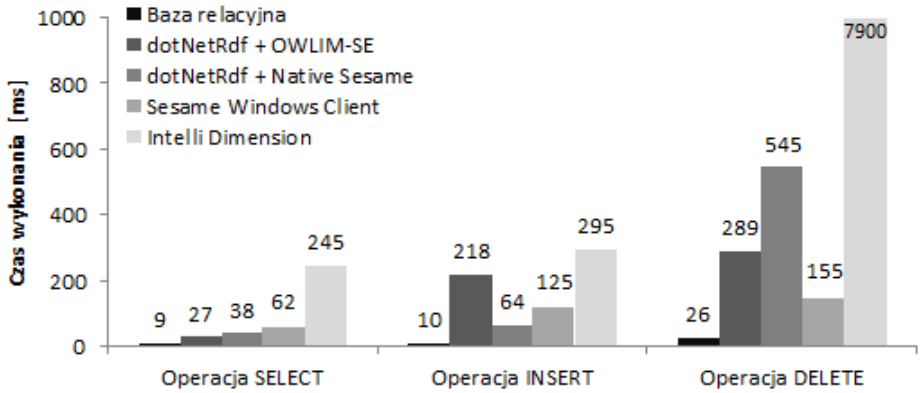
Przeprowadzone zostały pomiary w następujących warunkach testowych:

- wykorzystano maszynę z procesorem Pentium 4 HT 3.0 GHz, dostępną pamięcią operacyjną 1,5 GB, pracującą pod kontrolą systemu operacyjnego Windows 7,
- na jednej maszynie pracowały zarówno testowane biblioteki, jak i składnice danych RDF i relacyjna baza danych MS SQL Server 2008 R2 Express,
- w każdym przypadku w testowanym środowisku pracował tylko jeden użytkownik,

Otrzymane wyniki pomiarów przedstawione zostały na rysunkach 2, 3, 4, 5.



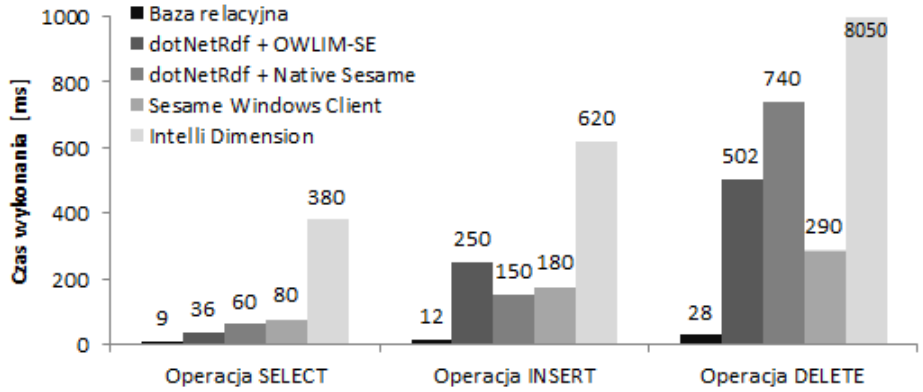
Rys. 2. Czas wykonania operacji dla 100 klientów (2300 trójek RDF)
Fig. 2. Execution time for 100 customers (2300 RDF triples)



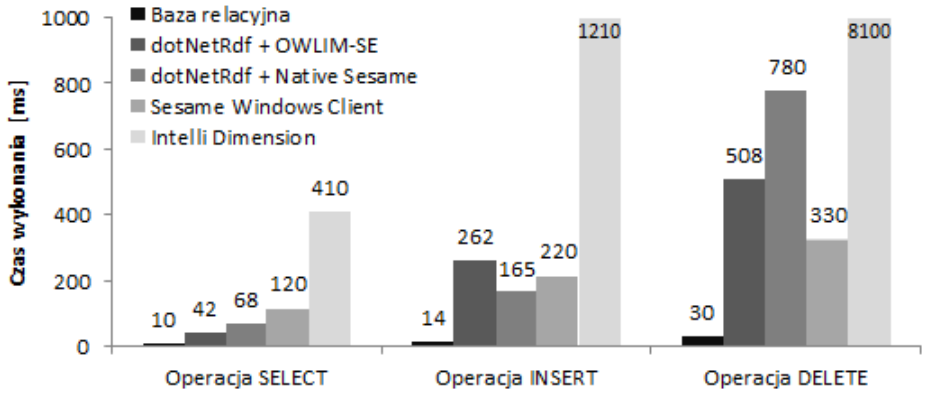
Rys. 3. Czas wykonania operacji dla 1000 klientów (23 000 trójek RDF)
Fig. 3. Execution time for 1000 customers (23 000 RDF triples)

Z przygotowanych wykresów wynika, że biblioteka IntelliDimension wyraźnie odbiega od pozostałych. Czas operacji zmierzony dla tej biblioteki, zarówno dla pobierania, jak i dodawania danych, odbiegał od czasów zmierzonych w przypadku pozostałych platform dla wszystkich testowanych zestawów danych. Zupełnie nieakceptowane, z punktu widzenia użytkownika aplikacji biznesowej, są również czasy osiągnięte dla tej biblioteki przy usuwaniu danych. Kilka sekund oczekiwania na wykonanie operacji na prostym obiekcie to czas, który z pewnością nie zaspokoi wymagań przeciętnego użytkownika aplikacji biznesowej. Na

rysunkach 2, 3, 4 i 5 czasy, które wykraczają poza przyjętą maksymalną wartość skali dla osi y (1000 ms), zostały wpisane wewnątrz słupków.



Rys. 4. Czas wykonania operacji dla 10 000 klientów (230 000 trójek RDF)
Fig. 4. Execution time for 10 000 customers (230 000 RDF triples)



Rys. 5. Czas wykonania operacji dla 100 000 klientów (2 300 000 trójek RDF)
Fig. 5. Execution time for 100 000 customers (2 300 000 RDF triples)

Warto zwrócić uwagę na czasy trwania operacji przeprowadzanych na relacyjnej bazie danych i porównać je z czasami dla pozostałych bibliotek w przypadku rosnącej ilości danych. Dla relacyjnej bazy danych, wraz ze wzrostem ilości danych czas trwania poszczególnych operacji (szczególnie operacji SELECT) zmienia się tylko nieznacznie. W przypadku wszystkich bibliotek operujących na danych RDF, czas operacji wyraźnie wzrasta wraz z ilością danych. Zachowanie stałego czasu odpowiedzi relacyjnej bazy danych gwarantuje odpowiednie indeksowanie danych w tabelach. Indeksowanie to tylko jedna z wielu cech i funkcjonalności, które w świecie relacyjnych baz danych są czymś powszechnym, zaś w świecie składnic RDF są w bardzo początkowych fazach rozwoju. Wiele z takich cech zostało wymienionych w rozdziale 3 niniejszego opracowania.

Repozytorium OWLIM-SE, obsługiwane przez bibliotekę dotNetRdf, wykazało w testach najlepszy czas odpowiedzi na zapytania. Wynika to z zaimplementowanych wielu technik optymalizacji wykonywania zapytań, bazujących głównie na buforowaniu danych i metadanych w pamięci operacyjnej [23]. Jednocześnie jednak repozytorium OWLIM-SE osiąga

znacznie słabsze wyniki od większości pozostałych testowanych repozytoriów pod względem czasu wykonywania operacji usuwania i dodawania danych. Na potrzeby aplikacji biznesowej, w której operacje modyfikacji danych są sporadyczne, najlepiej odpowiada repozytorium OWLIM-SE, obsługiwane przez biblioteki dotNetRdf. Jednak dla aplikacji, w której operacje modyfikacji danych wykonywane są równie często jak operacje wyszukiwania, znacznie lepszym rozwiązaniem będzie użycie natywnego repozytorium Sesame i bibliotek utworzonych w ramach projektu Sesame Windows Client, gdyż to połączenie technologii wykazuje najlepszy średni czas odpowiedzi na wszystkie operacje.

6. Podsumowanie

W niniejszej pracy jako najistotniejsze kryterium porównania badanych bibliotek przyjęto czas wykonania najważniejszych operacji. Wynika to z faktu, że użytkownicy aplikacji biznesowych, korzystających najczęściej z relacyjnych baz danych, szybką odpowiedź serwera przy typowych operacjach uznają za coś naturalnego. Jak się okazało po przeprowadzeniu testów, szybka odpowiedź serwera w przypadku podstawowych operacji wykonywanych na składnicach danych RDF nie jest już tak oczywista.

Technologie Semantic Web są w stosunkowo wczesnym stadium rozwoju, dlatego też poprawienie ich wydajności jest tylko kwestią czasu. Z punktu widzenia czasu wykonania podstawowych operacji, można rozważyć użycie testowanych bibliotek dotNetRdf i Sesame Windows Client, wraz z repozytorium Natvie Sesame lub OWLIM-SE, w niedużych aplikacjach biznesowych, składujących stosunkowo niewielką ilość danych. Większy czas wykonania operacji, w porównaniu z relacyjną bazą danych, jest rekompensowany przez zalety reprezentacji wiedzy w postaci ontologii OWL. W ramach kolejnych prac planowane jest przestawienie konkretnego zastosowania przedstawionych rozwiązań.

Projektując warstwę dostępu do danych RDF z użyciem testowanych bibliotek, przyjęto, że z testowaną aplikacją pracuje tylko jeden użytkownik. Takie założenie to znaczne uproszczenie, głównie ze względu na brak potrzeby zarządzania transakcjami i poziomami izolacji. W ramach prowadzonych prac założono również, że testowane składnice danych (w tym relacyjna baza danych) przechowują wszystkie dane na trwałych nośnikach dyskowych, podczas gdy zarówno w przypadku relacyjnych baz danych, jak również w przypadku repozytoriów RDF coraz częściej wykorzystywane są repozytoria pamięciowe przechowujące całość lub część danych w pamięci operacyjnej komputera [24]. Jak pokazuje przykład repozytorium OWLIM-SE, efektywne korzystanie z pamięci operacyjnej do buforowania danych może w dużym stopniu poprawić czas odpowiedzi na zapytania. Porównanie wydajności pamięciowych relacyjnych baz danych i pamięciowych repozytoriów RDF oraz próba dodania do testowanych aplikacji możli-

wości pracy równoległej przez więcej niż jednego użytkownika na tym samym repozytorium danych RDF, będą kolejnym krokiem prowadzonych badań.

BIBLIOGRAFIA

1. Strona internetowa konsorcjum W3C, http://www.w3.org/.
2. Berners-Lee T.: The Semantic Web – A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities. Scientific American Magazine, 2001.
3. Goczyła K., Zawadzka T.: Ontologie w Sieci Semantycznej. Studia Informatica, Vol. 27, No. 2 (67), Gliwice 2006, s. 65÷92.
4. Rekomendacja W3C, RDF, http://www.w3.org/TR/2004/REC-rdf-primer-20040210/.
5. Rekomendacja W3C, RDF Schema, http://www.w3.org/TR/rdf-schema/.
6. Rekomendacja W3C, OWL 2, http://www.w3.org/TR/owl2-overview/.
7. Rekomendacja W3C, SPARQL, http://www.w3.org/TR/rdf-sparql-query/.
8. Rekomendacja W3C, Wyrażenia języka OWL, http://www.w3.org/TR/owl-ref/.
9. Relational Databases on the Semantic Web, Tim Berners-Lee Design Issues, http://www.w3.org/DesignIssues/RDB-RDF.html.
10. Bąk J., Jędrzejek C.: Semantic Web – technologie, zastosowania, rozwój. XV Konferencja PLOUG, 2009.
11. Przegląd architektury ADO.NET, http://msdn.microsoft.com/en-us/library/h43ks021.aspx.
12. Przegląd technologii Entity Framework, http://msdn.microsoft.com/en-us/library/bb399567.aspx
13. Lista narzędzi Semantic Web, http://www.w3.org/2001/sw/wiki/Tools.
14. Wprowadzenie do użycia rozszerzeń .NET CLR na platformie SQL Server, http://msdn.microsoft.com/en-us/library/ms254498(v=vs.80).aspx.
15. Strona internetowa projektu Sesame, http://www.openrdf.org/.
16. Strona internetowa produktów OWLIM, http://www.ontotext.com/owlim/.
17. Strona internetowa projektu IKVM.NET, http://sourceforge.net/projects/ikvm/.
18. Strona internetowa projektu dotSesame, http://sourceforge.net/projects/dotsesame/.
19. Strona internetowa projektu Sesame Windows Client, http://sourceforge.net/projects/sesamewinclient/.
20. Strona internetowa producenta serwera Virtuoso, http://virtuoso.openlinksw.com/.
21. Strona internetowa projektu D2RQ, http://d2rq.org/.

22. Strona internetowa producenta narzędzia TopBraid Composer, http://www.topquadrant .com/products/TB_Composer.html.
23. Kiryakov A., Ognyanov D., Manov D.: OWLIM – A Pragmatic Semantic Repository for OWL. Lecture Notes in Computer Science, Vol. 3807, s. 182÷192.
24. Bach M., Duszeńko A., Werner A.: Koncepcja pamięciowych baz danych oraz weryfikacja podstawowych założeń tych struktur. Studia Informatica, No. 2B (90), Gliwice 2010, s. 63÷76.

Wpłynęło do Redakcji 8 stycznia 2013 r.

Abstract

In the last decade we can be observe relatively slow, but continuous and consequent development of Semantic Web technologies. The technologies are visible mostly in internet and open source projects. In this paper we try to find some ways to use Semantic Web technologies also for .NET business applications which use mostly relational databases as storage engines so far. The idea is to replace relational database in .NET project with Semantic Web knowledge base.

We tested execution time of select, insert and delete data operations for IntelliDimension, dotNetRdf and Sesame Windows Client libraries and compared them with results obtained for sample relational database. For testing purposes we used 4 relational databases and 4 sets of data with different amount of data. All the results are presented on figures 2, 3, 4 and 5.

The result figures show that tested libraries are still less mature than relational databases, but they may be sufficient for .NET applications with small amount of data. Only operations execution times for IntelliDimension libraries are not acceptable (especially for delete operations). However, as Semantic Web technologies are still in the early stage of development and don't have the advantage of many years of optimization, they have a great potential for the future.

Adres

Wojciech KIJAS: Politechnika Śląska, ul. Akademicka 16, 44-100 Gliwice, Polska, wojciech.kijas@gmail.com.