bazy danych, XML,
Java, Internet

Lech Madeyski [*]

XML W BAZACH DANYCH

Java i XML to dwa bardzo modne i często używane ostatnio słowa – klucze. Trudno sobie jednak wyobrazić poważne aplikacje korporacyjne bez użycia baz danych. Są one integralną częścią zarówno aplikacji zastanych (*legacy applications*), jak i nowych – tworzonych w architekturze wielowarstwowej – aplikacji internetowych biznesu elektronicznego. Podobnie jak Java zapewnia przenośność tworzonych aplikacji, czyli ich niezależność od platformy sprzętowo-systemowej, tak XML zapewnia przenośność danych. Nic więc dziwnego, że Java – a w szczególności XML – wywiera duży wpływ również na stosunkowo hermetyczną do tej pory dziedzinę baz danych. Można śmiało stwierdzić, że wraz z popularyzacją XML także w bazach danych nastąpiła swego rodzaju rewolucja. Pojawiło się całe spektrum nowych rozwiązań, które zostaną przedstawione w niniejszym artykule. Użyta terminologia i klasyfikacja bazuje na stosowanej przez Ronalda Bourreta [1].

1. WSTĘP

Jest wiele czynników decydujących o wyborze i architekturze środowiska bazodanowego. Część z nich ma charakter pomocniczy, inne odgrywają w tym procesie szczególnie istotną role. Jeżeli środowisko bazodanowe ma wspierać XML (eXtensible Markup Language), to najważniejszym czynnikiem mającym wpływ na wybór bazy danych jest jej przeznaczenie. Dokumenty XML można bowiem podzielić na dwie główne kategorie:

1. Dano-centryczne (*data-centric*).
2. Dokumento-centryczne (*document-centric*).

Dano-centryczne dokumenty XML to takie dokumenty, w których XML jest używany do transportu danych np. zamówienia, notowania giełdowe, dynamicznie generowane katalogi online czy listy adresowe na podstawie znanych, regularnych zbiorów danych. Ich fizyczna struktura – np. kolejność elementów siostrzanych lub sposób przechowywania danych (w atrybutach, elementach) – jest często nieistotny.

[*] Wydziałowy Zakład Informatyki, Wydział Informatyki i Zarządzania Politechniki Wrocławskiej, 50-370 Wrocław, Wybrzeże Wyspiańskiego 27, madeyski@ci.pwr.wroc.pl.

Dokumento-centryczne dokumenty XML to np. książki, listy elektroniczne, broszury. Charakteryzują się one nieregularną lub mało regularną strukturą, a ich struktura fizyczna (np. kolejność elementów) jest istotna.

Zaprezentowany podział dokumentów XML na dwie kategorie jest bardzo pomocny przy doborze odpowiedniej bazy danych, chociaż w praktyce podział ten nie zawsze jest tak klarowny. Może bowiem zdarzyć się, że dokumenty XML dokumento-centryczne zawierają regularnie strukturyzowane dane (np. metadane dokumentu) i na odwrót dokumenty XML dano-centryczne mogą np. zawierać dane o nieregularnej strukturze.

2. XML W BAZACH DANYCH

Rozwiązania bazodanowe dobiera się w zależności od kategorii dokumentów XML, które mają być obsługiwane.

Do pobierania i składowania danych w dokumentach dano-centrycznych, potrzebna jest baza danych dostosowana do przechowywania tych danych (np. relacyjna lub obiektowa) i oprogramowanie odpowiedzialne za transfer (import i eksport) danych w postaci dokumentów XML. Oprogramowanie to może być wbudowane w bazę i wtedy mówimy, że taka baza oferuje mechanizm translacji BD-XML (*XML-enabled database*) lub może być zewnętrznym translatorem BD-XML, czyli produktem klasy *Middleware*.

Do pobierania i składowania danych w dokumentach dokumento-centrycznych, potrzebna jest Natywna Baza XML (*Native XML Database*), czyli baza danych specjalnie zaprojektowana do przechowywania dokumentów XML lub System Zarządzania Zawartością (*Content Management System*), będący swego rodzaju aplikacją zaprojektowaną do zarządzania dokumentami zbudowaną z wykorzystaniem natywnej bazy danych. Natywne Bazy Danych jak i Systemy Zarządzania Zawartością są zaprojektowane, by przechowywać fragmenty zawartości np. rozdziały, elementy słownika, metadane dokumentu. Systemy Zarządzania Zawartością, choć zwykle używają Natywnej Bazy XML, mają dodatkową funkcjonalność (edycja zawartości, kontrola wersji, zarządzanie przepływem dokumentów).

Powyższe ogólne zasady są pomocne, nie są jednak ścisłe. Wiąże się to z tym, że oferowane rozwiązania gwałtownie ewoluują, a granice pomiędzy poszczególnymi grupami produktów mają charakter arbitralny. Na przykład granice pomiędzy tradycyjnymi bazami danych a natywnymi bazami danych XML ulegają zatarciu. Tradycyjne bazy danych powszechnie już oferują wsparcie dla XML, a natywne bazy XML zaczynają wspierać składowanie dokumentów w tradycyjnych (relacyjnych) bazach danych.

3. GRUPY PRODUKTÓW

W związku z pojawieniem się szerokiego spektrum produktów bazodanowych wspierających XML konieczne stało się ich pogrupowanie i określenie jednolitej terminologii. W związku z tym wyodrębniono następujące grupy produktów [1]:

- Translatory DB-XML (*Middleware*).
- Bazy danych wspierające transfer dokumentów XML (*XED – XML Enabled Databases*).
- Natywne bazy XML (*NXD – Native XML Databases*).
- Serwery XML (*XS – XML Servers*) i serwery aplikacji XML (*XAS – XML Application Servers*).
- Systemy zarządzania zawartością (*CMS – Content Management Systems*).
- Motory kwerend XML (*XQE – XML Query Engines*).
- Translatory XML-Obiekty (*XDB – XML Data Binding*).

W dalszej części tego rozdziału powyższe grupy produktów zostaną pokrótce omówione.

3.1. TRANSLATORY DB-XML

Translatory DB-XML (*Middleware*) zapewniają translację pomiędzy dokumentami XML a bazami danych. Są one zwykle wykorzystywane w przypadku aplikacji dano-centrycznych. Gdy są połączone z komercyjną bazą danych jako jej integralna część, to o takiej bazie mówi się, że wspiera transfer dokumentów XML (*XML Enabled Database*).

Znane są dwa podejścia wykorzystywane w produktach tej klasy:

- Mapowanie bazujące na szablonie (*template based mapping*) – zwykle ograniczające się do transformacji BD->XML.
- Mapowanie bazujące na modelu (*model based mapping*) – bardziej nadające się do realizacji dwukierunkowej transformacji BD<->XML.

Przykładowymi przedstawicielami tej grupy produktów są dwa komercyjne produkty firmy Oracle: XML SQL Utility for Java i XSQL Servlet oraz XML-DBMS, będący produktem Open Source.

3.1.1. XML SQL UTILITY FOR JAVA I XSQL SERVLET FIRMY ORACLE

Produkt XML SQL Utility for Java wykorzystuje mapowanie bazujące na modelu zapewniając dwukierunkowe mapowanie BD<->XML. XSQL Servlet jest serwletem używającym XML SQL Utility for Java do transferu danych z BD do XML.

W przypadku, gdy dane są transferowane z BD do XML, użytkownik dostarcza wyrażenie SELECT lub JDBC ResultSet zaś rezultaty są zwracane w postaci dokumentu XML lub DOM (*Document Object Model*). Gdy dane są transferowane z XML do BD, użytkownik dostarcza dokument XML lub DOM.

3.1.2. XML-DBMS

XML-DBMS [6] jest produktem klasy Open Source, wykorzystującym mapowanie bazujące na modelu, zapewniając dwukierunkowe mapowanie BD<->XML. Używa obiektowo-relacyjnego mapowania, które jest opisane poprzez język mapowania bazujący na XML. Użytkownicy mogą definiować, w jaki sposób elementy, atrybuty, PCDATA są mapowane w BD. Język jest dość elastyczny i zapewnia np. zdolność do przechowywania własności (atrybutów) w tabeli klasy lub w oddzielnej tabeli własności czy możliwość wyboru formatu parsowanej daty, czasu itd.

XML-DBMS może także generować relacyjny schemat (w postaci wyrażeń CREATE TABLE) na podstawie dokumentu DTD.

Struktura typowej aplikacji xml-dbms:

1. Stworzenie obiektu map.
2. Przekazanie obiektu map do obiektu domToDBMS lub DBMSToDom.
3. Odbiór i manipulacja rezultatami translacji.

3.2. BAZY WSPIERAJĄCE TRANSFER DOKUMENTÓW XML

Bazy danych wspierające transfer dokumentów XML (*XML Enabled Databases*), to bazy danych (zwykle relacyjne) zawierające rozszerzenie (wykorzystujące mapowanie bazujące na szablonie lub modelu) służące do transferu danych pomiędzy dokumentami XML, a samą bazą danych. Są one zwykle wykorzystywane w przypadku aplikacji dano-centrycznych.

Przykładami produktów, które można zaliczyć do tej grupy są m.in. najpopularniejsze relacyjne bazy danych:

- Oracle 8i, 9i ;
- Informix;
- DB2 XML Extender and DB2 Text Extender;
- Microsoft Access 2002;
- Microsoft SQL Server 2000.

Bazy te oferują bezpośredni dostęp do danych poprzez ODBC/JDBC.

Przykładowo Oracle 9i wprowadza obiektowy typ danych XMLType [4, 5], który przechowuje dokumenty XML jako CLOB. Umożliwia m.in. wykonywanie SQL-

owych operacji na zawartości XML-owej. Na dokumentach XML mogą być realizowane kwerendy (z użyciem XPath), co pozwala użytkownikom nie tylko pobierać dane z tych dokumentów w aplikacjach, ale także budować indeksy poszczególnych elementów i atrybutów w dokumentach.

3.3. NATYWNE BAZY XML

Natywne bazy XML (*Native XML Databases*) definiują logiczny model dokumentu XML zawierający przynajmniej elementy, atrybuty, PCDATA i porządek dokumentów (np. XPath, XML Infoset, modele implikowane przez DOM i SAX). Natywne bazy XML są wykorzystywane zarówno w przypadku aplikacji dano-centrycznych jak i dokumento-centrycznych.

W przypadku natywnych baz XML dokument XML jest fundamentalną jednostką (logicznego) składowania, tak jak w przypadku relacyjnych baz danych, jest nią wiersz tabeli. Nie jest natomiast wymagany żaden konkretny model fizycznego składowania. Natywna baza XML może bazować na relacyjnej, hierarchicznej lub obiektowej bazie danych albo też wykorzystywać własny format składowania np. indeksowane, kompresowane pliki.

Można wyróżnić dwie kategorie natywnych baz XML:

- Ze składowaniem tekstowym. Dokumenty są składowane w formie tekstowej (np. jako BLOB w relacyjnej bazie danych, plik w systemie plików z indeksami dokumentów w XML, własny, zoptymalizowany system składowania z indeksami, wsparciem transakcji itp.).
- Ze składowaniem binarnym. W tym przypadku składowany jest binarny model dokumentu (np. DOM).

W natywnych bazach XML jedynym interfejsem do danych jest interfejs bazujący na XML: XPath, DOM lub API bazujące na XML np. XML:DB API. To ostatnie jest interesującą koncepcją, która znalazła już praktyczne zastosowanie w przypadku natywnej bazy XML Apache Xindice.

XML:DB API zostało stworzone przez XML:DB Initiative [2] w celu ułatwienia tworzenia aplikacji wykorzystujących XML w bazach danych. Funkcjonalnie jest ono tym dla natywnych baz XML, czym ODBC lub JDBC dla relacyjnych baz danych.

XML:DB API bazuje na koncepcji kolekcji, które przechowują zasoby. Zasobem może być dokument XML, binarny BLOB lub jakiś inny typ jeszcze nie zdefiniowany. Kolekcje mogą być organizowane w sposób hierarchiczny, ich architektura przypomina system plików. Kolekcje oferują serwisy pozwalające na realizację kwerend dokumentów XML z wykorzystaniem XPath lub uaktualnianie zasobów w transakcyjnie bezpieczny sposób. Kwerendy mogą dotyczyć nie tylko pojedynczych dokumentów XML, ale także kolekcji.

Nie wszystkie natywne bazy XML wymagają, by z kolekcją był związany schemat. Oznacza to, że w takiej bazie, określanej mianem niezależnej od schematu (*schema-independent*), można przechowywać dowolne dokumenty XML w kolekcjach bez względu na schemat. Właśnie takie rozwiązanie przyjęto w przypadku bazy Apache Xindice [3]. Jest ono bardzo elastyczne i ułatwia życie programistom, ale należy pamiętać o konsekwencjach (integralność danych). Do niektórych zastosowań należy więc wybierać natywną bazę XML wspierającą schematy lub szukać innego rozwiązania problemu przechowywania danych XML.

Xindice zawiera wielowątkowy motor bazodanowy optymalizowany dla danych XML. Dokumenty są przechowywane w formie skompresowanej. Xindice zawiera motor kwerend bazujący na XPath. Kolekcje są indeksowane w celu zwiększenia efektywności realizowanych kwerend. W celu aktualizacji danych Xindice implementuje XML:DB XUpdate. Zawiera również kompletny zestaw narzędzi linii komend i trójwarstwowe API:

- XML:DB API to główne API do tworzenia aplikacji dbXML w Javie. Jest zbudowane na bazie dbXML Corba API.
- dbXML Corba API zapewnia dostęp do dbXML za pomocą innych języków niż Java. Jest zbudowane na bazie Core Server API. W przyszłości będzie ono zastępione nowym sieciowym API.
- Core Server API jest wewnętrznym API motoru bazodanowego.

3.4. SERWERY XML I SERWERY APLIKACJI XML

Serwery XML (*XML Servers*) są nie tyle bazami danych, co kompletnymi platformami do tworzenia aplikacji eBusiness czy aplikacji internetowych. Wykorzystują dane w postaci dokumentów XML w procesie natywnej komunikacji (wysyłania i odbierania danych) i składowania (persystencji). Zapewniają dostęp do różnych danych np. zastanych baz danych, systemów plików itd. wspierając procesy integracji istniejących systemów. Serwery XML są zwykle wykorzystywane w przypadku aplikacji dano-centrycznych.

Serwery aplikacji XML (*XML Application Server*) zwykle bazują na szablonach i używają języków skryptowych lub własnych języków znaczników, w które wbudowane są wyrażenia SQL do odczytywania danych i dynamicznego budowania dokumentów XML.

Produkty z tej grupy wykorzystują zwykle serwery WWW (*Web servers*), które obsługują komunikację poprzez Web i serwują statyczne dokumenty. Serwery aplikacji XML są wykorzystywane zarówno w przypadku aplikacji dano-centrycznych, jak i dokumento-centrycznych. W tej grupie produktów umieszcza się na przykład Apache Cocoon [7] czy ColdFusion firmy Macromedia (dawniej Allaire) [8].

Linia podziału pomiędzy serwerami XML a serwerami aplikacji XML nie jest klarowna. Zaproponowany przez Bourreta podział może się wydawać dyskusyjny. Wykorzystywanie języków skryptowych lub własnych języków znaczników nie wydaje się być jedynym i głównym wyznacznikiem przynależności produktu do grupy serwerów aplikacji, choć ich popularność jest bezdyskusyjna. Przykładowo na platformie Java serwery zawierające jedynie kontenery EJB nie wykorzystujące języków skryptowych ani własnych języków znaczników są zaliczane do serwerów aplikacji.

3.5. SYSTEMY ZARZĄDZANIA ZAWARTOŚCIĄ

Systemy zarządzania zawartością (*Content Management Systems*) spełniają zwykle rolę fasady natywnych baz XML, które są niejako ukryte przed użytkownikiem.

Systemy zarządzania zawartością są dostosowane do zarządzania dokumentami (*document-centric documents*), a także oferują dodatkową funkcjonalność (np. edycję zawartości, kontrolę wersji, dostęp dla wielu użytkowników). W rezultacie zapewniają składowanie, pobieranie i tworzenie nowych dokumentów z istniejących fragmentów (zawartości).

Systemy zarządzania zawartością różnią się od natywnych baz XML tym, że oferują więcej funkcjonalności związanej z zarządzaniem dokumentami (np. kontrolę wersji).

Większość takich systemów może publikować XML w sieci WWW, ale w odróżnieniu od serwerów aplikacji XML, systemy zarządzania zawartością są głównie zaprojektowane do zarządzania dokumentami (*document-centric documents*) podczas gdy XML'owe serwery aplikacji są zaprojektowane do serwowania zarówno dano-centrycznych jak i dokumento-centrycznych dokumentów w sieci WWW.

Do tej grupy produktów można zakwalifikować komercyjne rozwiązania oferowane przez firmę Documentum [9], jak również projekt Prowler [10] rozwijany na zasadach Open Source. Ten ostatni jest jednym z elementów większej całości – portalu informacyjnego przedsiębiorstwa (*Enterprise Information Portal*).

3.6. MOTORY KWEREND XML

Motory kwerend XML (*XML Query Engines*) są samodzielnymi programami/motorami zapewniającymi tylko i wyłącznie realizację kwerend dokumentów XML. Mogą implementować różne języki – oczywiście najlepiej te, które są standaryzowane przez W3C, a więc XPath i XQuery.

3.7. TRANSLATORY XML-OBIEKTY

Translatory XML-Obiekty (*XML Data Binding*) wiążą dane zawarte w dokumentach XML z obiektami zaprojektowanymi specjalnie dla tych danych. Zaletą wykorzystania w aplikacjach (zwykle dano-centrycznych) rozwiązań z tej grupy jest prosty i bardziej naturalny niż w przypadku DOM, SAX czy JAXP sposób operowania na danych. Operacje na dokumentach XML są bowiem realizowane na związanych z nimi obiektami np. ziarnami Javy (*JavaBeans*). Obiecującymi rozwiązaniami wykorzystującymi wiązanie danych zawartych w dokumentach XML z obiektami Javy są JAXB (*Java Architecture for XML Binding*) [11] firmy Sun i Castor XML [12] organizacji ExoLab Group wspierającej tworzenie oprogramowania na zasadach Open Source.

Produkty należące do tej grupy mogą niekiedy przechowywać i odtwarzać obiekty z bazy danych.

4. PODSUMOWANIE

W artykule skrótowo zaprezentowano szeroki wachlarz produktów i zagadnień związanych ze wsparciem XML w systemach bazodanowych. Pewne jest, że powszechna akceptacja dla XML spowodowała duże zmiany w segmencie rynku baz danych. Natywne bazy XML i inne produkty bazodanowe wspierające XML nie są oczywiście panaceum na wszelkie bolączki. Rozszerzają one paletę dostępnych narzędzi, a jeśli zostaną odpowiednio użyte, mogą przynieść twórcom systemów wiele korzyści. Autor ma nadzieję, że treści zaprezentowane w tym artykule będą pomocne w dokonaniu świadomego wyboru odpowiednich narzędzi.

Trudno przesądzać, jaki kształt przyjmą systemy bazodanowe w przyszłości. Możliwe są różne scenariusze począwszy od rozwiązań uniwersalnych, które będą ewoluować z obecnych systemów relacyjnych baz danych po rozwiązania kombinowane łączące różne typy repozytoriów. Wciąż również pojawiają się nowe koncepcje (jak chociażby *XML Data Binding*) mogące wywrzeć istotny wpływ na kierunek dalszego rozwoju aplikacji internetowych. a w konsekwencji również systemów bazodanowych wspierających XML.

Literatura

[1] Strona domowa Ronalda Bourreta, http://www.rbourret.com.

[2] XML:DB Initiative, http://www.xmldb.org.

[3] Projekt Apache Xindice, http://www.dbxml.org.

[4] Oracle Technet, http://technet.oracle.com/tech/xml/content.html.

[5] Oracle, Database Support for XML, http://download-east.oracle.com/otndoc/oracle9i/901_doc/appdev.901/a88894/adx05xml.htm#1012692.

[6] Ronald Burret, XML-DBMS, http://www.rpbourret.com/xmldbms/index.htm.

[7] Projekt Apache Cocoon, http://xml.apache.org/cocoon/index.html.

[8] Macromedia ColdFusion Server, http://www.macromedia.com/software/coldfusion/.

[9] Documentum, http://www.documentum.com/products/content/index.html.

[10] Prowler, http://www.infozone-group.org/projects_main.html.

[11] Sun JAXB, http://java.sun.com/xml/jaxb/index.html.

[12] ExoLab Group, http://castor.exolab.org.

XML IN DATABASES

Java and XML are hot topics today. However it is difficult to realize serious corporate portals or Internet applications without databases. Databases are important parts of legacy applications as well as multitiered eBusiness applications. Java supports applications portability through platform independence and XML supports data portability. It is no wonder that Java and especially XML exert an influence on database field. It appeared a wide range of new products which are presented in this article. The terminology is based on the proposed by Ronald Bourret [1].