Analiza porównawcza technologii dostępu do baz danych w środowisku Borland Delphi

Andrzej Barczak[1], **Dariusz Zacharczuk**[1]

Streszczenie: Różne języki programowania oferują liczne technologie dostępu dedykowane określonej bazie danych lub pozwalające obsługiwać kilka serwerów baz danych. W rozdziale przedstawiona jest analiza i ocena technologii dostępu do baz danych Oracle, DB2, SQL Server, Informix, InterBase, MySQL oraz Access z poziomu aplikacji napisanych w języku Delphi i Java. Praca jest próbą opracowania metody mierzenia efektywności technologii dostępu do baz danych oraz wykorzystania tej metody do przeprowadzenia badań mających na celu stwierdzenie, która z technologii dostępu jest najbardziej wydajna przy wykonywaniu różnego rodzaju zapytań SQL.

Słowa kluczowe: Databases, ODBC, ADO, BDE.

1. Ewolucja technologii dostępu do baz danych

Po tym, jak na początku lat dziewięćdziesiątych systemy baz danych, takie jak dBase, Paradox, Clipper i FoxPro stały się popularne, zaistniała konieczność stworzenia mechanizmów dostępu do danych przechowywanych w różnych bazach z poziomu różnych języków. To podstawowa przyczyna powstania Query By Example (QBE) – pierwszej technologii dostępu do baz danych. Query By Example dokonuje konwersji zapytania użytkownika do postaci formalnego zapytania bazy danych. Dzięki temu użytkownik może realizować nawet złożone zapytania do bazy danych bez znajomości metod formalnych.

Jednym z pierwszych mechanizmów dostępu do danych stanowiących konkurencję dla Query By Example była technologia DAO. Jest to technologia, która jako pierwsza posłużyła się obiektami dostępu do danych.

Kolejną technologią było ODBC (Open DataBase Connectivity – otwarte łącze baz danych). ODBC jest interfejsem umożliwiającym programom łączenie się z systemami zarządzania bazami danych. Jest to interfejs API (Application Programming Interface) niezależny od języka programowania, systemu operacyjnego i bazy danych.

ODBC definiuje niskopoziomowy zbiór funkcji umożliwiających aplikacjom klienta i serwera wymianę danych i przekazywanie instrukcji bez konieczności posiadania dokładnych informacji o implementacji zarówno klienta, jak i serwera, niezależnie od tego, czy klient i serwer działają na tym samym, czy na różnych komputerach, a nawet na różnych platformach programowych lub/i sprzętowych.

Na architekturę interfejsu ODBC składają się cztery elementy (Rys. 1):

[1] Akademia Podlaska, Instytut Informatyki, ul. Sienkiewicza 51, 08-110 Siedlce, Polska
e-mail: abarczak@ap.siedlce.pl, dzariusz@dzariusz.pl

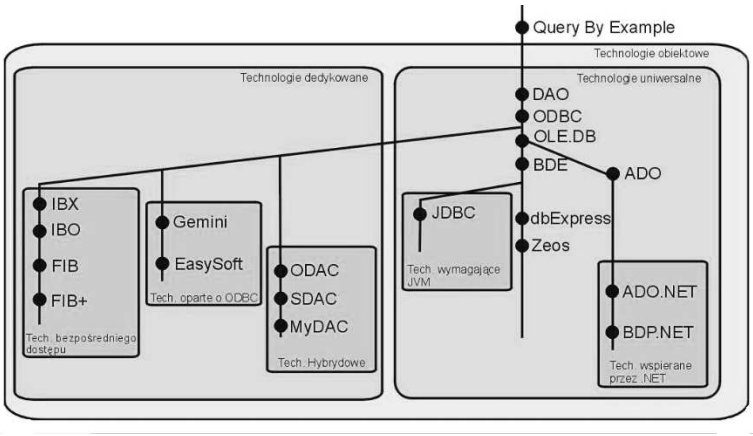
Rysunek 1. Architektura ODBC

- aplikacja – wykonująca specyficzne czynności przetwarzania posługując się zapytaniami SQL w celu uzyskania i składowania danych niezbędnych dla procesu;
- zarządca sterowników (Driver Manager) – w postaci biblioteki DLL (Dynamic Link Library), którego zadaniem jest udostępnianie aplikacji odpowiedniego sterownika bazy danych;
- sterownik – zazwyczaj w postaci biblioteki DLL, element wykonujący funkcje interfejsu ODBC wywoływany przez zarządcę sterowników. Przekazuje on również do źródła danych żądania SQL, a do aplikacji – uzyskane wyniki. Jeśli jest to wymagane – sterownik może modyfikować wykonywane zapytanie SQL w celu dostosowania go do specyfiki docelowego źródła danych (np. dopasowanie dialektu SQL konkretnego systemu zarządzania bazą danych)
- źródło danych – najczęściej system zarządzania bazą danych.

W czasie, gdy powstawał interfejs ODBC w firmie Microsoft powstała koncepcja modelu obiektów COM (Component Object Model), które mogą być wykorzystywane w każdym środowisku programistycznym Win32: OLE.DB. Aplikacje mogą wykorzystywać OLE DB do bezpośredniego dostępu do danych, albo poprzez OLE.DB mogą wywoływać ODBC, aby uzyskać dostęp do bazy danych poprzez ODBC.

OLE.DB jest bardzo trudne do implementacji, dlatego później stworzono technologie wyższego poziomu, które wprawdzie wykorzystują OLE.DB do łączenia się z bazami danych, ale ich implementacja jest zdecydowanie uproszczona (np. ADO - ActiveX Data Objects).

Po sukcesie OLE.DB i ODBC zaczęły powstawać inne technologie dostępu do baz danych. Jednak największy rozkwit technologii dostępu rozpoczął się wraz z pojawieniem się na rynku środowisk Borland Delphi i Borland C++ Builder. Dla tych środowisk stworzono wiele specjalistycznych technologii dostępu do baz danych: IBX (InterBase Express), IBO (InterBase Objects), FIB (Free InterBase), ODAC (Oracle Data Access Components), SDAC (SQL Server Data Access Components), MyDAC (MySQL Data Access Components), Gemini, EasySoft, BDP:NET



Rysunek 2. Ewolucja technologii dostępu do baz danych

(Borland Data Provider for .NET), BDE (Borland Data Engine). Technologie dostępu dla swoich narzędzi programistycznych stworzyły również Firmy Microsoft i Sun. Tak powstały ADO i JDBC (Java DataBase Connectivity). Technologia ADO została zaadoptowana także przez inne firmy, dzięki czemu można aktualnie korzystać z ADO np. w Borland Delphi.

Ewolucję dostępu do baz danych z uwzględnieniem charakterystycznych cech poszczególnych technologii przedstawiono na Rys. 2. Należy zauważyć, że wraz z rozwojem technologii dostępu do baz danych następuje ich podział i specjalizacja. Na drodze ewolucji wykształciły się technologie, które muszą być uruchamiane na maszynie wirtualnej, albo w środowisku uruchomieniowym (.NET Framework). Dzięki temu ich implementacja i możliwość przenoszenia na inne komputery jest dużo łatwiejsza.

Wraz z pojawieniem się na rynku środowiska Borland Delphi zaczęły powstawać technologie dedykowane jednemu serwerowi baz danych. Wśród nich są technologie:

- bezpośredniego dostępu do baz danych,
- technologie oparte na mechanizmach ODBC,
- technologie hybrydowe, które oprócz specyficznych mechanizmów dostępu do baz danych potrafią również wykorzystywać źródła ODBC.

Po pewnym czasie zauważono, że ODBC i OLE.DB nie są wystarczająco efektywne i stworzono nowy mechanizm dostępu - BDE (Borland DataBase Enigme). Była to – wkrótce po jej pojawieniu się – podstawowa technologia dostępu do baz danych, obecnie jest ona wypierana przez inne, takie jak ADO czy dbExpress. Zaletą BDE jest większa w porównaniu z ODBC szybkość działania oraz prostota obsługi, wadą zaś – ograniczona zdolność przenoszenia. Wynika to z faktu, że BDE nie obsługuje systemu zarządzania bazą danych bezpośrednio, lecz wykorzystuje program pośredniczący SQL Links.

Rysunek 3. Zależności poszczególnych poziomów w BDE

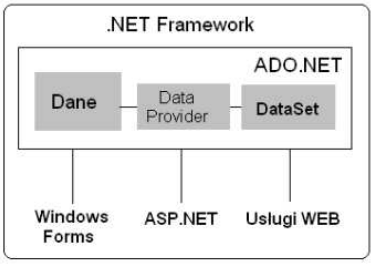
W BDE po raz pierwszy pojawiło się pojęcie silnika bazodanowego (engine). Każdy system zarządzania bazami danych posiada własny interfejs API (Application Programming Interface) komunikujący się z bazą danych. Rzadko, albo nigdy nie zdarza się tak, aby dwa systemy miały taki sam interfejs. Natomiast dzięki BDE nie potrzeba znać szczegółów implementacji takich baz danych – połączenie np. z Oracle czy InterBase przebiega tak samo. Specjalne sterowniki w postaci programów SQL Links dokonują translacji poleceń BDE na polecenia specyficzne dla danego systemu zarządzania bazą danych. Dzięki takiemu rozwiązaniu aplikacja może się łączyć z dowolną bazą danych nie znając szczegółów jej implementacji.

Również w BDE po raz pierwszy pojawił się własny bufor rekordów w postaci zbioru danych (dataset). Definicja zbioru danych została sformułowana jako struktura kolumn (grupujących dane tego samego typu) i wierszy będących warstwą pośredniczącą pomiędzy komponentami obsługi danych (w Delphi jest to np. komponent klasy TTable), a komponentami wizualnymi, za pomocą których prezentowana jest zawartość tabeli (np. DBGrid). Ilustrację poszczególnych poziomów BDE przedstawiono na Rys. 3.

BDE nie jest, niestety, rozwiązaniem idealnym. Zaprojektowany został w czasach, gdy dominowały proste systemy bazodanowe oparte na płaskich plikach, takie jak Paradox czy dBase. Obecnie najczęściej stosowanym rozwiązaniem są relacyjne bazy danych, obsługiwane za pomocą języka SQL, współpracujące z oprogramowaniem w architekturze klient - serwer. Instalowanie BDE na wynajmowanych serwerach internetowych bardzo często nie jest możliwe. Obecnie technologia BDE nadaje się do realizacji małych projektów o charakterze lokalnym, gdzie bezpieczeństwo i spójność danych są stosunkowo mało istotne.

Sukcesy ODBC i BDE skłoniły firmę Microsoft do opracowania nowego mechanizmu dostępu do danych, który jest łatwiejszy w implementacji niż OLE.DB. Tak powstała technologia dostępu do baz danych o nazwie ADO (ActiveX Data Object). ADO, podobnie jak OLE.DB, bazuje na obiektach COM (Component Object Model).

Idea, która przyczyniła się do powstania ADO, zakładała dostęp do bazy danych bez znajomości jej wewnętrznej struktury. Taką samą ideą kierowali się twórcy BDE. Jednak różnica pomiędzy BDE i ADO jest znacząca. BDE to zwykłe komponenty wykorzystujące dodatkowe biblioteki, natomiast ADO jest warstwą pośrednią



Rysunek 4. Architektura ADO.NET

i technologią samą w sobie. Oprócz dostępu do baz danych, za pomocą ADO można uzyskać dostęp do plików Excela, plików tekstowych, plików Lotusa, HTML i wielu innych źródeł danych, co jest pomysłem i nowatorskim i bardzo użytecznym.

Wraz z pojawieniem się platformy .NET firma Microsoft wprowadziła w ADO pewne zmiany. Obecnie dostępne jest „nowe" ADO.NET, które można podzielić na: mechanizmy dostępu do danych oraz system przechowywania danych.

Uproszoną architekturę ADO.NET, której głównymi komponentami są: zbiór danych (DataSet) oraz dostarczyciel danych (Data Provider), przedstawiono na Rys. 4. Źródło danych może znajdować się w fizycznej bazie danych lub pliku XML. Data Provider wykonuje połączenia i wysyła polecenia, a DataSet reprezentuje dane w pamięci.

Dostarczyciel danych (Data Provider) jest odpowiednikiem sterownika, którego zadaniem jest połączenie się z bazą danych lub odczytanie odpowiedniego pliku, ukrywając przy tym szczegóły implementacyjne. ADO.NET zapewnia obsługę baz danych MS SQL Server, Oracle oraz technologii OLE.DB. Jednak firma Borland udostępnia dodatkowe mechanizmy – Borland Data Provider (BDP), które umożliwiają proste połączenia z dodatkowymi źródłami danych, np. z bazami InterBase lub DB2.

W ADO dane pomiędzy źródłem danych i aplikacją transportowane były w formie zbioru rekordów (RecordSet) bez względu na to, czy pochodziły z jednej, czy z wielu tabel. W ADO.NET zbiór rekordów został zastąpiony przez zbiór danych. Dane znajdujące się w zbiorze danych zawarte są w tabelach (DataTable). Jeśli aplikacja żąda od ADO.NET zwrócenia danych z dwóch tabel, to dane zwracane są w formie dwóch tabel, które znajdują się w zbiorze danych. Relacje pomiędzy tabelami znajdującymi się w zbiorze danych reprezentowane są przez instancję klasy DataRelation.

Sukces technologii ADO przyczynił się do opracowania przez firmę Borland nowszej i dużo lepszej niż BDE technologii dostępu do baz danych. Tak powstała technologia dbExpress, która dostępna jest zarówno w Delphi, jak i w Kylix'ie. Jest to szybka i elastyczna technologia, dużo bardziej efektywna niż BDE lub ODBC. Technologia ta posługuje się efektywnymi sterownikami i dzięki temu zapewnia jeden z najszybszych dostępów do informacji przechowywanych w bazach danych. Specjalnie stworzone komponenty tej grupy mają charakter uniwersalny, dostępne

są także dla platformy Linux. Jednak uniwersalność ta przesądza również o stosunkowo ubogim repertuarze możliwości w zakresie manipulowania danymi.

Używając BDE, ADO lub dbExpress korzysta się z silnika niezależnego od serwera. Można przełączać serwer używany przez aplikację, chociaż nie jest to proste. Jeśli potrzeba, aby aplikacja niezmiennie używała jednego serwera baz danych, należy zadbać, aby silnik bazy danych obsługiwał konkretne API serwera baz danych, który będzie używany. Spowoduje to, że programy nie będą przenoszalne na inne serwery baz danych, ale omija się warstwę pośrednią, zyskując w ten sposób na funkcjonalności i przede wszystkim – na szybkości działania.

Używając InterBase Express (IBX) pomija się warstwę pośrednią w postaci np. BDE i komunikuje się bezpośrednio z oprogramowaniem klienckim InterBase lub Firebird. Komponenty IBX dostępne na dwóch paletach Delphi w całości zostały stworzone właśnie w Delphi i są specjalizowane dla serwera InterBase i Firebird. Dzięki temu, że usunięto warstwę pośrednią, jaką jest np. BDE, pomiędzy aplikacją i serwerem InterBase, zmniejszono do minimum narzut na wykonanie zapytań, a czas komunikacji z serwerem stał się rewelacyjnie krótki.

Ponieważ komponenty i klasy IBX są napisane w całości w Delphi, więc „wkompilowują" się w pliki exe aplikacji. Nie potrzeba dystrybuować z aplikacją wielu plików DLL, tak jak jest to w przypadku BDE. Ponieważ aplikacja składa się tylko z pliku wykonywalnego nie trzeba także budować dla niej wyrafinowanych programów instalacyjnych. Nie trzeba również dbać o to, by ktoś nie zepsuł konfiguracji (tak jak jest to w przypadku BDE) – wszystko jest pod kontrolą IBX.

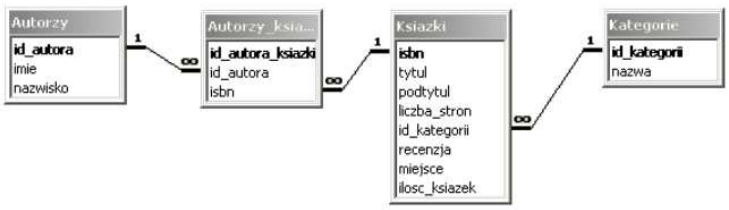
2. Technologie dostępu do baz danych XXI wieku

Na początku XXI wieku narodziły się dwie technologie: FIB i IBO pozwalające na dostęp tylko do bazy danych InterBase. Technologie IBX, FIB i IBO są do siebie bardzo podobne. Różnią się nazwami komponentów, właściwości i klas, jakie trzeba używać, aby połączyć się z bazą danych oraz efektywnością. Poza tym różnic między tymi technologiami nie ma zbyt wiele. W technologiach FIB i IBO nie występują klasy, które nie mają swoich odpowiedników w technologii IBX. Z tego względu technologie FIB i IBO nie są rewolucyjne, a są po prostu kolejnymi technologiami, które służą do pracy z InterBase.

Drugą grupą technologii dedykowanych określonemu serwerowi baz danych są technologie, które wywodzą się z ODBC. Doskonałym przykładem takich technologii są Gemini i EasySoft. Jednak ta gałąź ewolucji technologii dostępu do baz danych ma coraz mniejsze znaczenie.

Po sukcesie IBX firma Core Lab Software Development postanowiła stworzyć technologie dostępu do baz danych dedykowane innym niż InterBase serwerom baz danych. Tak narodziły się technologie ODAC, SDAC i MyDAC. Technologie te są o tyle rewolucyjne, że są technologiami hybrydowymi. Oznacza to, że mają własne mechanizmy dostępu do baz danych, ale potrafią również korzystać ze źródeł ODBC zdefiniowanych w systemie operacyjnym.

Wraz z powstaniem nowego, przenośnego, niezależnego od platformy i architektury sprzętowej języka programowania, jakim jest Java, pojawiła się idea skonstruowania nowego intefejsu bazodanowego, który spełniłby podobne wymagania



Rysunek 5. Schemat relacji pomiędzy tabelami bazy danych

jak sam język. Naturalną konsekwencją takiego podejścia było oderwanie się od już istniejących mechanizmów API, a w szczególności – ODBC i opracowanie nowej technologii, wykonanej w pełni w Javie. Nie oznacza to wcale, że programiści używający dotąd ODBC musieli przestawić się na zupełnie inny tok myślenia przy tworzeniu aplikacji bazodanowych. Przeciwnie, JDBC funkcjonalnie przypomina technologię ODBC. Zwiększono jedynie możliwości interfejsu oraz przystosowano go do odmiennej specyfiki języka Java.

JDBC określa pewien poziom zgodności z istniejącymi standardami SQL. Jednak główne założenie mówi, iż każdy sterownik JDBC musi odpowiadać co najmniej wersji ANSI SQL-92 standardu SQL. Ponadto twórcy i pomysłodawcy interfejsu stworzyli inne założenia, które spełniają kolejne wersje standardu JDBC (omówione dalej). Według tych założeń zapytania SQL przesyłane są do odpowiedniego SZBD bez względu na możliwość ich realizacji. W przypadku, gdy dany SZBD nie potrafi obsłużyć zlecenia przekazanego przez JDBC, aplikacja generuje określony wyjątek. Ponadto JDBC udostępnia informacje o SZBD i jego cechach szczególnych (ustawieniach, możliwościach itd.). Każdy sterownik JDBC służy do komunikacji z konkretną bazą danych i nie nadaje się do wykorzystywania w przypadku baz innych producentów.

3. Przedmiot i zakres badań

Na potrzeby badań efektywności technologii stworzona została prosta baza danych, której struktura przedstawia Rys. 5.

W celu zbadania efektywności technologii opracowano w środowisku Borland Delphi programową metodę mierzenia czasu wykonania zapytań SQL. Należy jednak zauważyć, że badania przeprowadzone były w systemie MS Windows, który nie jest systemem czasu rzeczywistego, dlatego czas wykonania zapytania na różnych komputerach może być różny i zależny od wielu czynników: zajętości procesora, szybkości taktowania procesora, ilości pamięci operacyjnej, obciążenia systemu i wielu innych (Tabela 1 prezentuje platformę programową i sprzętową, na jakiej przeprowadzane były badania). Wpływ na czas wykonania zapytań ma również liczba procesów uruchomionych w systemie, dlatego podczas testowania programów wszystkie zbędne procesy zostały wyłączone. Dzięki temu różnice czasu wykonania takich samych zapytań były bardzo małe i maksymalnie osiągały wartość do 0,01 sekundy.

Tabela 1. Parametry sprzętu wykorzystywanego w testach

Parametry sprzętowe	Parametry systemowe
procesor: AMD Duron 800 MHz, pamięć RAM: 384 MB (DIMM), dysk: Samsung 80 GB, 4 MB cache Nagłówek 4 poziomu	system operacyjny: Windows XP Professional + SP2

Programista tworzący aplikacje bazodanowe, któremu zależy na ich efektywności, musi wziąć pod uwagę zarówno efektywność w obrębie technologii, jak i efektywność w obrębie samej bazy danych.

Tabela 2 przedstawia wyniki pomiarów efektywności badanych technologii współpracujących z poszczególnymi bazami danych.

Kolejny rodzaj testu, został przeprowadzony przy 100% użyciu procesora. Zdjęcie o rozmiarze 3554x2629 poddano obróbce w Photoshop'ie za pomocą wymagającej wtyczki Thredgeholder Pro z pakietu Little Ink Pot – dzięki temu przez jakieś 40 sekund pracuje tylko procesor a nie pracuje dysk. W takim przypadku efektywność technologii spada od 2 do 5 razy, jednak w dalszym ciągu najbardziej efektywne są technologie dedykowane (IBX, MyDAC). Tabela 3 przestawia wyniki testu.

Kolejnym krokiem w kierunku testowania efektywności jest zajęcie dysku. Podczas wykonywania kolejnego testu równolegle uruchomiono archiwizowanie duzego pliku wideo. Porównując test przy 100% użyciu procesora z testem przeprowadzanym przy 100% użyciu procesora i dodatkowo obciążonym dysku można stwierdzić, że efektywność nieznacznie spada, ale głównie przy operacjach wymagających zapisu (Insert, Update), praktycznie nie zmienia się czas wykonania instrukcji select, instrukcja delete jest tylko nieznacznie wolniej wykonywana. Ogólnie efektywność spadła 2-6 razy w stosunku do testu przeprowadzonego na nieobciążonym komputerze. Wyniki prezentuje tabela 4.

Ostatnią próbą, jakiej zostały poddane testowane technologie było zmierzenie czasu wykonania zapytania zawierającego podzapytanie.

Przy małej ilość rekordów (prawie pustej bazie) – czasy różnią się o parę tysięcznych do paru setnych sekundy. Przy 1000 rekordów książek, 500 autorach i 100 kategoriach wyniki można obserwowa w tabeli 5.

Czas wykonania zapytania rośnie znacząco przy dużo większej liczbie rekordów w poszczególnych tabelach. Przypuszczalnie, dla 2 milionów książek i 100 tysięcy autorów oraz kategorii to poszczególne zapytania select trwały by po kilkanaście-/kilkadziesiąt minut. W każdym bądź razie na testowanej liczbie rekordów czas wykonania poszczególnych zapytań jest kilkukrotnie dłuższy niż przy pustej bazie danych. Między poszczególnymi technologiami są też wyraźne różnice – tak dużych różnic nie byłoby, gdyby w bazie było miliony zapytań – w takich przypadkach decyduje szybkość wykonania zapytania przez serwer, natomiast sama technologia dostępu ma zdecydowanie mniejsze znaczenie.

Tabela 2. Efektywność w obrębie technologii dostępu i systemu zarządzania bazą danych

System baz danych	Technologia	Średni czas wykonania prostego zapytania (bez podzapytań) w sekundach (miejsce w klasyfikacji)			
		SELECT	INSERT	UPDATE	DELETE
Oracle	ADO	0,0148 (20)	0,0103 (20)	0,0096 (18)	0,0102 (24)
	dbExpress	0,0051 (5)	0,0093 (19)	0,0076 (15)	0,0079 (21)
	ODAC	0,0063 (11)	0,0072 (16)	0,0072 (14)	0,0072 (18)
	BDE	0,0096 (17)	0,0069 (15)	0,0082 (16)	0,0079 (21)
	ODBC	0,0094 (16)	0,0241 (25)	0,0249 (25)	0,0130 (25)
	EasySoft	0,0249 (23)	0,0317 (26)	0,0285 (26)	0,0196 (26)
InterBase	IBX	0,0051 (5)	0,0015 (2)	0,0012 (1)	0,0008 (1)
	ADO	0,0342 (25)	0,0029 (8)	0,0039 (9)	0,0024 (10)
	dbExpress	0,0020 (2)	0,0017 (4)	0,0019 (3)	0,0015 (3)
	BDE	0,0076 (14)	0,0023 (6)	0,0022 (5)	0,0019 (6)
	FIB	0,0058 (8)	0,0011 (1)	0,0013 (2)	0,0008 (1)
	IBO	0,0125 (18)	0,0035 (10)	0,0040 (10)	0,0020 (7)
	Gemini	0,0081 (15)	0,0028 (7)	0,0024 (6)	0,0020 (7)
SQL serwer 2000	ADO	0,0163 (21)	0,0092 (18)	0,0132 (22)	0,0086 (23)
	SDAC	0,0059 (10)	0,0033 (9)	0,0029 (8)	0,0023 (9)
	dbExpress	0,0022 (3)	0,0052 (13)	0,0083 (17)	0,0060 (17)
	ODBC	0,0058 (8)	0,0051 (12)	0,0048 (12)	0,0050 (15)
DB2	ADO	0,0273 (24)	0,0087 (17)	0,0118 (21)	0,0074 (19)
	dbExpress	0,0016 (1)	0,0064 (14)	0,0098 (19)	0,0051 (16)
	ODBC	0,0072 (12)	0,0105 (21)	0,0114 (20)	0,0075 (20)
MySQL	MyDAC	0,0044 (4)	0,0016 (3)	0,0019 (3)	0,0018 (4)
	ODBC	0,0072 (12)	0,0019 (5)	0,0024 (6)	0,0018 (4)
Access	ADO	0,0208 (22)	0,0039 (11)	0,0060 (13)	0,0047 (14)
	ODBC	0,0054 (7)	0,0141 (23)	0,0045 (11)	0,0035 (11)
Informix	ADO	0,0374 (26)	0,0144 (24)	0,0167 (24)	0,0041 (13)
	ODBC	0,0136 (19)	0,0115 (22)	0,0138 (23)	0,0035 (11)

4. Podsumowanie

Analizując powyższe wyniki badań można odpowiedzieć na pytanie, jaki system bazy danych i jaką technologię dostępu do danych wybrać, żeby aplikacja była najbardziej efektywna. Pięć najbardziej efektywnych technologii służących do wykonywania różnego typu zapytań SQL wyróżniono w tabeli pogrubieniem. Wyniki doświadczeń wskazują, że najlepszym rozwiązaniem dla osób, którym zależy na efektywności aplikacji, jest wybranie serwera InterBase i technologii IBX, FIB, dbExpress lub serwera MySQL i technologii MyDAC. Tabela potwierdza również opinię, że najbardziej efektywne są technologie dedykowane określonemu serwerowi (IBX, FIB i MyDAC obsługują tylko jeden serwer baz danych). Potwierdza się również fakt, że najbardziej efektywne serwery baz danych to InterBase i MySQL. Należy pamiętać, że InterBase to serwer baz danych napisany w Delphi. FIB

Tabela 3. Efektywność w obrębie technologii dostępu i systemu zarządzania bazami danych przy 100% użyciu procesora przez inny proces.

System baz danych	Technologia	Średni czas wykonania zapytania w sekundach			
		SELECT	INSERT	UPDATE	DELETE
Oracle	ADO	0,0624	0,0559	0,0629	0,0501
	dbExpress	0,0231	0,0560	0,0606	0,0479
	ODAC	0,0603	0,0501	0,0586	0,0474
	BDE	0,0639	0,0514	0,0580	0,0505
	ODBC	0,0621	0,0612	0,0633	0,0600
InterBase	IBX	0,0461	0,0159	0,0190	0,0191
	ADO	0,0683	0,0269	0,0518	0,0591
	dbExpress	0,0229	0,0220	0,0200	0,0205
	BDE	0,0538	0,0271	0,0204	0,0284
	FIB	0,0488	0,0162	0,0200	0,0187
	IBO	0,0613	0,0294	0,0299	0,0315
SQL serwer 2000	ADO	0,0610	0,0512	0,0599	0,0501
	SDAC	0,0471	0,0224	0,0272	0,0218
	dbExpress	0,0216	0,0487	0,0506	0,0442
	ODBC	0,0483	0,0476	0,0500	0,0434
DB2	ADO	0,0618	0,0521	0,0609	0,0499
	dbExpress	0,0216	0,0497	0,0601	0,0420
	ODBC	0,0543	0,0499	0,0524	0,0490
MySQL	MyDAC	0,0249	0,0203	0,0203	0,0198
	ODBC	0,0483	0,0288	0,0296	0,0227
Access	ADO	0,0583	0,0467	0,0598	0,0507
	ODBC	0,0489	0,0422	0,0501	0,0447
Informix	ADO	0,0702	0,0591	0,0683	0,0516
	ODBC	0,0662	0,0552	0,0626	0,0504

i IBX to technologie również napisane w Delphi (dzięki temu pomijane są wszelkie warstwy dostępu do danych np. ODBC), a dbExpress zostało stworzone przez firmę Borland (producenta Delphi). Dlatego nasuwa się kolejny wniosek: największą efektywność uzyskuje się korzystając z produktów i technologii jednej firmy (tu – firmy Borland).

Pozostałe przetestowane technologie zostały stworzone w innych językach programowania, wykorzystują warstwy pośrednie lub są przeznaczone dla mniej efektywnych systemów baz danych. Stąd wynika przewaga w szybkości wykonywania instrukcji SQL technologii IBX i FIB. Zdaniem autorów, w przyszłości trudno będzie w znaczący sposób poprawić efektywność technologii IBX i FIB. Pod względem efektywności są to najlepsze technologie współpracujące z najlepszym pod tym względem sęrwerem – InterBase.

Wspomniana już technologia dbExpress jest najlepszym rozwiązaniem pod względem efektywności, jeśli chodzi o technologię, która pozwala na dostęp do wielu różnych serwerów. dbExpress jest rekomendowaną przez firmę Borland technologią

Tabela 4. Efektywność w obrębie technologii dostępu i systemu zarządzania bazami danych przy 100% użyciu procesora przez inny proces i dużym obciążeniu dysku (odczyt i zapis do kilkuset procent).

System baz danych	Technologia	Średni czas wykonania zapytania w sekundach			
		SELECT	INSERT	UPDATE	DELETE
Oracle	ADO	0,0628	0,0594	0,0643	0,0515
	dbExpress	0,0245	0,0599	0,0646	0,0493
	ODAC	0,0612	0,0540	0,0622	0,0490
	BDE	0,0630	0,0539	0,0622	0,0523
	ODBC	0,0629	0,0630	0,0672	0,0616
InterBase	IBX	0,0452	0,0183	0,0202	0,0200
	ADO	0,0699	0,0298	0,0550	0,0609
	dbExpress	0,0230	0,0262	0,0241	0,0220
	BDE	0,0541	0,0300	0,0243	0,0300
	FIB	0,0489	0,0199	0,0248	0,0201
	IBO	0,0619	0,0334	0,0339	0,0332
SQL serwer 2000	ADO	0,0621	0,0542	0,0638	0,0520
	SDAC	0,0480	0,0252	0,0313	0,0230
	dbExpress	0,0219	0,0513	0,0586	0,0465
	ODBC	0,0480	0,0513	0,0551	0,0452
DB2	ADO	0,0617	0,0561	0,0649	0,0514
	dbExpress	0,0210	0,0537	0,0643	0,0430
	ODBC	0,0539	0,0539	0,0566	0,0499
MySQL	MyDAC	0,0250	0,0244	0,0245	0,0205
	ODBC	0,0484	0,0324	0,0345	0,0240
Access	ADO	0,0584	0,0498	0,0633	0,0519
	ODBC	0,0492	0,0463	0,0540	0,0461
Informix	ADO	0,0710	0,0622	0,0703	0,0530
	ODBC	0,0679	0,0592	0,0666	0,0521

dostępu do baz danych. Czasy uzyskiwane przez tę technologię podczas wykonywania zapytań SQL są naprawdę bardzo dobre, a czasy wykonywania zapytań typu SELECT najlepsze spośród wszystkich testowanych technologii.

Pod względem efektywności zawodzi technologia ADO. W przypadku ADO nowoczesne mechanizmy stworzone przez firmę Microsoft pozwalają w łatwy sposób wykonywać instrukcje SQL z poziomu kilkunastu języków programowania m.in. z poziomu Delphi. Jednak czasy wykonywania zapytań SQL są niejednokrotnie gorsze od czasów wykonywania zapytań SQL za pomocą technologii ODBC.

Programista tworząc oprogramowanie bazodanowe bardzo często staje przed dylematem, jaką technologię dostępu do danych ma wybrać. Czy ma wykorzystać technologię wieloplatformową, czy jednoplatformową, obsługującą wiele systemów baz danych, czy dedykowaną konkretnemu serwerowi?

Nie ma jednoznacznej odpowiedzi na te pytania. Jeśli jednak zależy nam na efektywności i szybkości działania oraz aplikacja ma być przeznaczona do sys-

Tabela 5. Efektywność w obrębie technologii dostępu i systemu zarządzania bazami danych przy wykonaniu skomplikowanych zapytań select.

System baz danych	Technologia (Średni czas wykonania zapytania w sek.) SELECT		
Oracle	ADO (0,0448) BDE (0,0383)	dbExpress (0,0158) ODBC (0,0382)	ODAC (0,0299)
InterBase	IBX (0,0209) BDE (0,0235)	ADO (0,0839) FIB (0,0229)	dbExpress (0,0083) IBO (0,0378)
SQL serwer 2000	ADO (0,0483) ODBC (0,0288)	SDAC (0,0228)	dbExpress (0,0079)
DB2	ADO (0,0810)	dbExpress (0,0077)	ODBC (0,0217)
MySQL	MyDAC (0,0237)	ODBC (0,0316)	
Access	ADO (0,0600)	ODBC (0,0288)	
Informix	ADO (0,0989)	ODBC (0,0408)	

temu operacyjnego MS Windows, dobrym rozwiązaniem jest technologia dbExpress oraz technologie IBX, FIB, ODAC, SDAC, MyDAC, które dedykowane są określonemu serwerowi baz danych i dzięki temu są bardzo efektywne.

Literatura

Boduch A. (2003) *Delphi 7. Ćwiczenia zaawansowane.* Helion, Gliwice.

Boduch A. (2004) Optimal regulation of nonlinear dynamical systems. *Delphi 8 .NET. Kompendium programisty.* Helion, Gliwice.

Cantu M. (2002) *Delphi 6.* MIKOM, Warszawa.

Mościcki A., Kruk I. (2006) *Oracle 10g i Delphi. Programowanie baz danych.* Helion, Gliwice.

www.ai.komisauto.pl, www.openlinksw.com

Comparative analysis of Technology of Access to Databases in Borland Delphi Environment

In this paper the evolution of technologies of access to databases is presented. For definite sysytems of databases management (databases servers) and technologies of access, there was made an experiment, enabling to estimate the avarage of time of answering, definite types of questions - what means eficiency of each technologies.