Podpora OLAP na platformě .NET

Semestrální práce na předmět IT_380 – Vývoj klient/server aplikací

Vypracoval Borek Bernard, duben 2004

Abstrakt

Tato práce se zabývá problematikou podpory OLAP databází na platformě Microsoft .NET. Po úvodu do OLAP zpracování dat se věnuje vícedimenzionálnímu dotazovacímu jazyku MDX a specifikaci XML for Analysis (XMLA). Poslední kapitola popisuje knihovnu ADOMD.Net – z čeho vychází, k čemu slouží a jaký je její základní objektový model. Čtenář by si měl odnést základní představu o tom, jak lze na platformě .NET přistupovat k analytickým službám MS SQL Serveru 2000.

Úvod do OLAP

Aby byl celý následující text pochopitelný, stručně popíšeme základní pojmy jako např. OLAP, datová kostka, dimenze apod.

OLTP (On-line Transactional Processing)

Relační databáze pomocí transakcí umožňují mnoha klientům současně provádět určité operace na databázovém serveru tak, aby byla zachována konzistence dat. Tyto výkonné databáze jsou základem často rozsáhlých informačních systémů, které s postupem času nashromáždí ohromné množství dat.

Nevhodnost OLTP pro analytické zpracování dat

Jelikož jsou transakční databáze postaveny na „2D" relačním modelu, nejsou úplně vhodné pro následné zpracování nashromážděných dat. Je sice pravda, že i nad OLTP databází lze postavit analytickou aplikaci, ale má to několik zásadních vad. Zaprvé by analytické práce musely být prováděny souběžně se zpracováváním dalších a dalších transakcí, zadruhé se doporučuje data ukládat do normalizovaných tabulek. To znamená hodně atomických relačně svázaných tabulek a tudíž značnou neefektivnost a pomalost při pokládání dotazů. Zatřetí, charakter analytických dat je v reálném světě z podstaty vícedimenzionální, proto už intuice dává tušit, že relační systémy nebudou to pravé.

Co je to OLAP

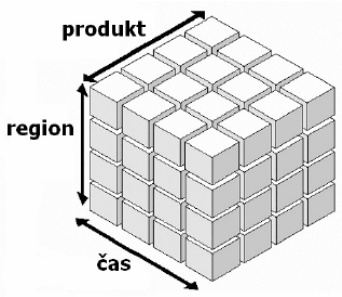
Proto byly k analytickým účelům vyvinuty OLAP systémy. Zkratka OLAP znamená On-line Analytical Processing a jedna z možných definic jejího významu je tato: *OLAP je volně definovaná sada principů, které poskytují dimenzionální rámec pro podporu rozhodování.*
V této definici je dobré povšimnout si tří věcí:

- OLAP neznamená žádnou konkrétní implementaci
- pohled na data je multidimenzionální (na rozdíl od 2D pohledu v OLTP databázích)
- cílem OLAP je poskytnout pomoc manažerům v rozhodovacích situacích

Datová kostka

Stejně jako jsou v relačních databázích data ukládána v tabulkách, v OLAP databázích se k tomuto účelu používají tzv. datové kostky. Jedná se o zavedený termín, který nejlépe odpovídá lidské prostorové představivosti, a rozhodně nevyjadřuje omezení počtu dimenzí. Těch může být teoreticky nekonečně mnoho, vždy samozřejmě záleží na konkrétní implementaci (jazyk MDX zabudovaný do analytických služeb MS SQL Serveru 2000 jich podporuje až 64).

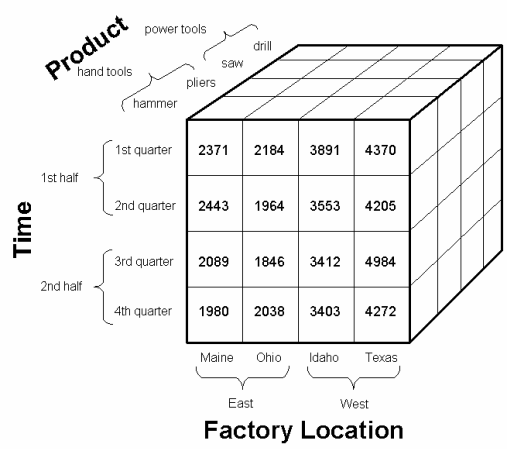
Základní představu o tom, co to kostka vůbec je, si lze udělat z prvního obrázku (zdroj: [Lacko]). Tato kostka obsahuje 3 dimenze – produkt, region a čas. Jednotlivé malé kostičky, které tvoří obsah kostky, buďto nesou určité business informace (např. náklady, zisky, objemy prodeje apod.), nebo jsou samy dalšími datovými kostkami, ovšem s vyšší rozlišovací schopností.



Obrázek 1 – Schematické zobrazení kostky

Dimenze, úrovně, členové a měrné jednotky

Bližší pohled na kostku ukazuje obrázek 2 (zdroj: [SQLServer]). Obsahuje tři **dimenze (dimensions)** – produkt, čas a umístění továrny. Každá z dimenzí obsahuje určité **úrovně (levels)** – např. dimenze čas obsahuje úrovně pololetí a kvartál, dimenze produkt obsahuje úrovně výrobek a skupina výrobků (tato pojmenování jsem si vymyslel; úroveň můžeme chápat jako míru podrobnosti pohledu, více viz níže). Každá úroveň obsahuje určité **členy (members)** – např. úroveň kvartály obsahuje členy „1st quarter", „2nd quarter" atd. To, co se nachází v průsečíku dimenzí, se nazývá **měrné jednotky (measures)**[1] – např. ve druhém čtvrtletí bylo v továrně v Ohiu vyrobeno 1964 kladiv.

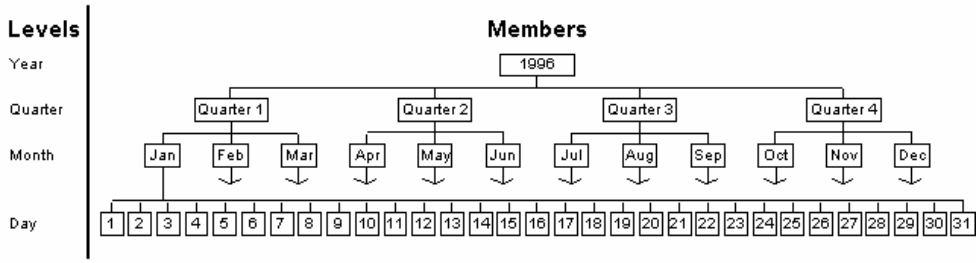


Obrázek 2 – Podrobnější pohled na kostku

Struktura dimenze

Pro lepší pochopení struktury dimenze je uveden třetí obrázek (zdroj: [SQLServer]). Ten krásně ilustruje, jak se jednotliví členové hierarchicky vyšších dimenzí rozpadají na podrobnější a podrobnější.

[1] Ačkoliv zní překlad „měrné jednotky" nezvykle, skutečně se používá. Výraz „measures" by se totiž mezi jinak celkem kompletní českou terminologií příliš nevyjímal.



Obrázek 3 – Struktura dimenze

Agregace

Když jsme se nyní letmo seznámili s datovými kostkami, můžeme konečně ukázat, co je na nich tak výjimečného. Obrázek 2 je sice skoro ideální pro první seznámení se strukturou kostek, zároveň však vyvolává dojem, že jednotlivé business informace jsou k dispozici pouze pro nejdetailnější úroveň (např. pro jednotlivé dny). Hlavní zbraň datových kostek je ale v tom, že konkrétní čísla jsou přístupná pro všechny členy všech úrovní – stejně jako dostanu odpověď na to, kolik bylo v Ohiu vyrobeno kladiv 12. května 2002, dostanu odpověď také na dotaz, kolik jich bylo vyrobeno za měsíc květen, za druhý kvartál, za první pololetí nebo za celý rok 2002. Oficiální terminologií bychom řekli, že měrné jednotky jsou agregovány pro jednotlivé členy. Nezní to jako podstatná vlastnost pro podporu manažerského rozhodování?

Měrné jednotky jsou vektory

Posledním zjednodušením, pro které se autoři druhého obrázku v zájmu jednoduchosti rozhodli, je zobrazení jednoho konkrétního čísla na průsečíku dimenzí. Kostky se ale neomezují na skalární měrné jednotky a umožňují současně nasčítávat celý vektor business informací – současně můžeme uchovávat data např. o počtu vyrobených kusů, o množství prodaných kusů, o nákladech, o příjmech, o ziscích atd. atd.

MDX

Na závěr zůstává otázka, jakým způsobem se lze na kostku dotazoval. V prostředí MS SQL Serveru 2000 (respektive jeho analytických služeb) máme k dispozici jazyk MDX (Multidimensional Expressions), který se snaží zachovat si podobnost s SQL (SELECT … FROM … WHERE), přesto stojí na úplně jiných základech. Jeho popis by vystačil na samostatnou práci, v krátkosti se na něj podíváme v následující kapitole.

Shrnutí

OLAP je rozsáhlá disciplína a terminologie oficiální dokumentace analytických služeb MS SQL Serveru 2000 zavádí ještě mnoho dalších pojmů. Ačkoliv jsou některé z nich velmi důležité (např. tabulka faktů, tabulky dimenzí, schémata dimenzí apod.), věřím, že více souvisí se serverovou stranou mince a že se bez nich čtenář této práce zaměřené především na klientský přístup docela dobře obejde.

MDX – Multidimensional Expressions

MDX, dotazovací jazyk pro vícedimenzionální struktury, je základním stavebním kamenem všech klientských aplikací pracujících s OLAP databázemi. Letmo se podíváme na jeho klíčové syntaktické prvky.

Základní syntaxe

Nejjednodušší možný MDX dotaz vypadá následovně:

```
SELECT <sada členů> ON <osa1>, <sada členů> ON <osa2>, ...
FROM <název kostky>
```

Typicky se používá dotaz rozšířený o klauzuli WHERE, například[2]

```
SELECT
   {Time.[1997].Children} on columns,
   {[Measures].[Store Sales]} on rows
FROM Sales
WHERE Store.[All Stores].USA
```

Výsledkem bude následující tabulka, která říká, kolik bylo v jednotlivých obchodech v USA prodáno zboží za jednotlivá čtvrtletí roku 1997:

	Q1	Q2	Q3	Q4
Store Sales	139 628,35	132 666,27	140 271,89	152 671,62

Pro pochopení MDX výrazů je potřeba zavést několik dalších pojmů. Bohužel k některým z nich neexistují rozumně znějící české překlady, proto se budeme držet anglické terminologie.

Set

Set je posloupnost členů z jedné nebo více dimenzí, která je ohraničena složenými závorkami. Příkladem setu je `{Time.[1997].Children}`, což vyjadřuje posloupnost všech členů na nejbližší nižší úrovni, které jsou dětmi člena `1997` (v našem případě to jsou jednotlivá čtvrtletí). Za povšimnutí stojí, že jednoslovné názvy začínající písmenem není nutné dávat do hranatých závorek (`Time`), zatímco víceslovné názvy nebo názvy začínající číslem je nutno uzavřít (`[1997]`).

Všimněte si, že v syntaxi MDX jsou měrné jednotky rovněž zcela rovnoprávnou dimenzí, kterou lze dotazovat pomocí klíčového slova `Measures`.

Axis dimension

Sada členů (set) uvedená v klauzuli `SELECT` se nazývá axis dimension. Osová dimenze je tedy dimenze, pro kterou se data zobrazují pro více členů (pro set členů).

Slicer dimension

Slicer dimension je dimenze uvedená v klauzuli `WHERE` a data se budou pro tuto dimenzi získávat pouze pro jednoho jejího člena – vyjadřuje tedy omezení dotazu.

MDX funkce

Dalším významným prvkem jazyka MDX jsou funkce – v našem příkladu například používáme funkci `Children`, která vrací set přímých potomků daného člena. Další často používanou funkcí je `Members`, která vrací sadu členů dané úrovně (levelu) – např. `[Time].[Year].Members` vrátí set

[2] Zde použitý příklad využívá analytickou databázi FoodMart 2000, která se instaluje spolu s analytickými službami MS SQL Serveru 2000. Ideálním nástrojem na testování MDX dotazů je jednoduchá aplikace napsaná ve Visual Basicu, kterou lze najít v nabídce Start na cestě Microsoft SQL Server → Analysis Services → MDX Sample Application

`{[1997], [1998]}`.

Tuple

Pojem tuple vyjadřuje nějakou podkostku (výřez). Uzavírá se do kulatých závorek, např. `([Time].[1998], [Product].[All Products].[Drink])`. Tento tuple definuje podkostku obsahující hodnoty pro rok 1998 a současně pro produkty z rodiny nápojů.

Calculated members

Posledním důležitým konceptem je možnost definovat kalkulované členy. Tito členové fyzicky v kostce nejsou a nasčítávají se až za běhu. Zde je jednoduchý příklad:

```
WITH MEMBER [Measures].[Profit] AS
   '[measures].[Sales] - [measures].[Cost]',
SELECT
   {[Measures].[Profit]} on columns,
   ...
```

Definujeme člena Profit a jak se má nasčítat a potom ho můžeme známým způsobem používat. Pro úplnost ještě jeden „detail" – kalkulovaného člena lze definovat buďto uvedeným způsobem, po čemž má platnost pouze jednoho dotazu, nebo pomocí syntaxe CREATE MEMBER, po čemž má platnost jedné relace.

Další vlastnosti MDX

Mezi další užitečné vlastnosti MDX patří např. možnost definovat pojmenované sety (named sets) a potom se na ně odkazovat pomocí jména.

Bohatství MDX spočívá především v množství jeho funkcí (dobrý přehled si lze udělat buďto z oficiální dokumentace nebo z MDX Sample Application). Mezi podstatné patří funkce pro drill-down (zanořování, zvyšování podrobnosti pohledu) a pro roll-up (opak). Těch je značné množství, jako příklad můžeme uvést funkci Descendants. Důležité je, že pomáhají jednoduchým způsobem provádět „navigační" posuny po kostce a výrazně tak ulehčují implementaci.

Shrnutí

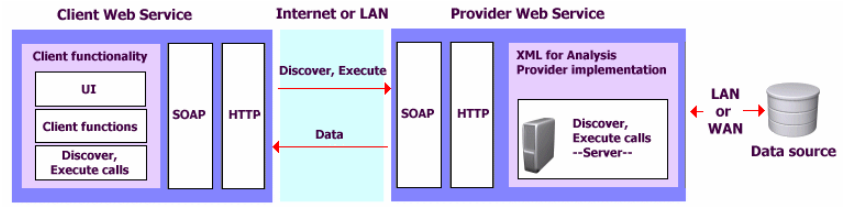
MDX je mocný jazyk, který znamená pro vícedimenzionální struktury totéž, co SQL pro relační databáze. Pro podrobnější informace doporučuji nahlédnout do dokumentace analytických služeb SQL Serveru 2000.

XML for Analysis (XMLA)

Seznámili jsme se se základy OLAP technologií a s dotazovacím jazykem MDX. Nyní je čas podívat se na to, jak jsou MDX dotazy kostce vůbec předávány.

Architektura XMLA

XMLA je iniciativa Microsoftu a několika dalších firem. Jedná se o komunikační API založené na XML (konkrétně na protokolu SOAP) určené pro přístup ke standardním datovým zdrojům umístěným na webu. Architekturu ozřejmuje obrázek 4 (zdroj [XMLA]):



Obrázek 4 – Architektura XMLA

Tato architektura eliminuje nutnost používat OLE DB rozhraní a komunikuje přes velmi obecné a všeobecně rozšířené protokoly jako HTTP nebo XML.

Discover a Execute

Klíčové jsou dvě funkce – Discover a Execute. Discover je určena pro získávání metadat o daném datovém zdroji, přičemž metadata jsou např. názvy tabulek, dimenzí, kostek apod. Metoda Execute slouží k vykonání konkrétního dotazu.

Tyto základní informace budou pro naše účely stačit.

ADOMD.Net

„Zlatým hřebem večera" je technologie ADOMD.Net, pomocí které lze na platformě .NET snadno přistupovat k OLAP databázím. Nyní, na konci dubna 2004, je knihovna ADOMD.Net pro veřejnost k dispozici v beta verzi, od 7. dubna je na požádání k dispozici první Release Candidate a podle příslibu podpory Microsoftu bude finální verze vydána co nevidět.

Předchůdce se jmenoval ADOMD

Jak už to u technologií Microsoftu bývá, když má něco koncovku .NET, někdy v minulosti ji to pravděpodobně nemělo (viz ASP.NET nebo ADO.NET). ADOMD.Net není výjimkou a předchůdce se jmenoval ADOMD, což je zkratka pro ActiveX Data Objects – Multidimensional.

ADO – ActiveX Data Objects

ADO je jedna z mnoha technologií, které Microsoft vyvinul pro přístup k datům. Konkrétně používá OLE DB poskytovatele (alternativně lze použít ODBC). ADOMD je rozšíření, které se poprvé objevilo v MS SQL Serveru 7.0 a které rozšiřuje možnosti OLE DB poskytovatelů o přístup k OLAP službám.

ADOMD.Net

ADOMD.Net je datový poskytovatel, který má sice podobnou architekturu jako ADOMD, ale na rozdíl od něj je standardním .Net datovým poskytovatelem, což znamená, že je celý napsaný v řízeném kódu[3]. Microsoft se tedy snaží jako u svých ostatních „.NET" technologií vývojářům maximálně usnadnit přechod tím, že vnitřně všechno kompletně přepíše, ale API zachová podobné.

K čemu ADOMD.Net

Když už víme, co se pod pojmem ADOMD.Net skrývá, bylo by dobré ujasnit, k čemu vlastně slouží. Z předchozího textu vyplynulo, že pro přístup k OLAP databázím potřebujeme v podstatě pouze dvě věci: zaprvé dota-

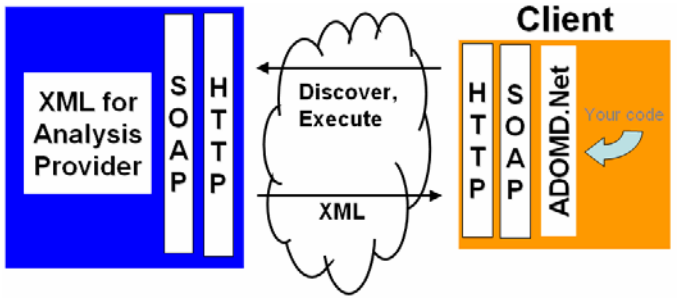
[3] Řízený kód (managed code) je kód napsaný pro běh v rámci CLR (Common Language Runtime). Více informací o CLR lze získat např. z oficiální dokumentace SDK .NET Frameworku.

zovací jazyk MDX a zadruhé XMLA, tedy něco, co sedí na serveru, přijímá SOAP požadavky, dekóduje z nich MDX dotazy, které vykoná, výsledky opět zabaluje do SOAP a odesílá zpět klientovi. ADOMD.Net tedy není ničím jiným, než klientskou technologií, jejímž hlavním cílem je poskytnout jednoduché API odstiňující programátora od implementačních detailů XMLA. Konkrétně, největší část XMLA specifikace je věnována datové struktuře MDDataSet, která vyjadřuje multidimenzionální data. ADOMD.Net umí (mimo jiné) tuto strukturu přečíst a inicializovat objekty svého objektového modelu, které potom zpřístupní klientské aplikaci.

Čemu se ale vývojář nevyhne, je MDX. Ačkoliv je jazyk MDX poměrně složitý, není to dáno jeho syntaxí, ale komplexitou vícedimenzionálních struktur. Těžko by šlo vymyslet nějaké další zjednodušující API. Toto tvrzení je asi zřejmé, nicméně jsem to chtěl pro jistotu zdůraznit, aby nedošlo k mýlce.

Vztah ADOMD.Net a XMLA

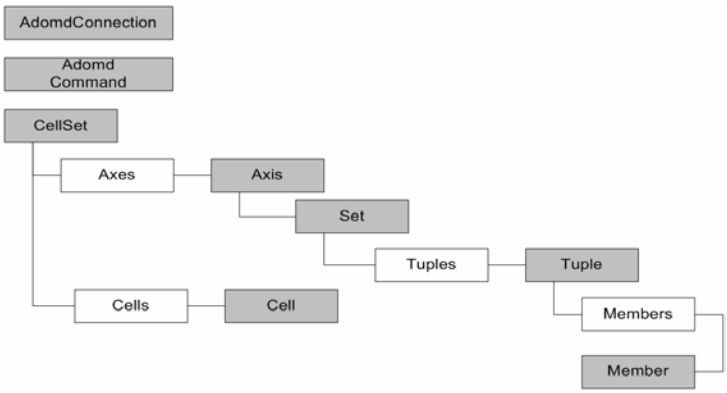
Jak už bylo řečeno, ADOMD.Net není ničím jiným než objektovým modelem zprostředkujícím komunikaci s XMLA – ADOMD.Net tedy nelze používat, pokud na straně serveru není XMLA nainstalováno[4]. Základní podstatu zobrazuje obrázek 5 (zdroj [TechEd]):



Obrázek 5 – Vztah XMLA a ADOMD.Net

Objektový model

Šestý obrázek (zdroj [TechEd]) zobrazuje nejdůležitější elementy objektového modelu ADOMD.Net:



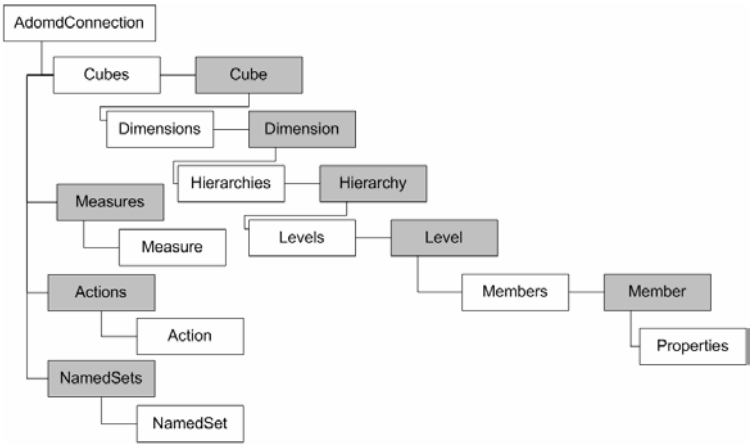
Obrázek 6 – Objektový model ADOMD.Net

[4] Perličkou, která podle mnoha příspěvků v diskusních skupinách potrápila nejednoho vývojáře, je skutečnost, že ADOMD.Net 1.0 Beta ke svému běhu potřebuje XMLA 1.1 **Beta**, zatímco s verzí 1.0 Stable vůbec nefunguje. Příští verze SQL Serveru (codename Yukon) už bude XMLA nativně podporovat, takže se přístup k OLAP databázím o něco zjednoduší.

`AdomdConnection` a `AdomdCommand` mají nápadně podobné názvy objektům `Connection` a `Command` z ADO.NET. Není to náhoda, ADOMD třídy jsou od nich skutečně odvozeny.

Adomd-Connection

`AdomdConnection` není jen objektem představujícím připojení k analytické databázi. Kromě toho totiž zpřístupňuje všechna důležitá metadata, jak lze nejsnáze nahlédnout z následujícího obrázku (zdroj [TechEd]):



Obrázek 7 – Struktura objektu AdomdConnetion

Tato struktura je velmi podobná té v COMovském ADOMD, kde byly metainformace zveřejněny prostřednictvím objektu `CubeDef`. ADOMD.Net přidává další objekty, konkrétně `Measure`, `Action` a `NamedSet`, v podstatě se ale hodně věcí nezměnilo.

Adomd-Command

Druhým top-level objektem je `AdomdCommand`, jehož hlavním úkolem je nést MDX dotaz. Jeho vykonáním získáváme třetí a poslední nejdůležitější objekt, `CellSet`.

CellSet

`CellSet` je objekt nesoucí (respektive zprostředkující) samotná data výsledku dotazu na kostku. Jeho strukturu lze vidět na dříve uvedeném obrázku (Obrázek 6 – Objektový model ADOMD.Net). První větev obsahuje data z dimenzí – např. hodnoty jednotlivých členů apod. Druhá větev potom obsahuje hodnoty výsledku jako takové v objektu Cell.

Alternativní objekty zpřístupňující výsledek

ADOMD.Net kromě objektu CellSet nabízí ještě dva další, které zpřístupňují výsledek dotazu:

- AdomdDataReader, což je forward-only read-only objekt zpřístupňující výsledek dotazu v podobě 2D tabulky. Pro mnoho případů je úspornější náhradou CellSetu, který se vždy musí nejdřív načíst celý; na druhou stranu se s objektem AdomdDataReader hůře zachází, protože jsou vícedimenzionální data zakódována do 2D tabulky.
- XmlReader, což je standardní třída .NET Frameworku (ze jmenného prostoru System.Xml). S výsledky vůbec nic nedělá a pouze zpřístupňuje získaný SOAP dokument.

Shrnutí ADOMD.Net je knihovna umožňující jednoduchý přístup k OLAP databázím tím, že místo složité práce s XMLA poskytovatelem vystavuje elegantní objektový model, který je navíc implementován v řízeném kódu. Hlavními objekty tohoto modelu jsou AdomdConnection, AdomdCommand a CellSet.

Závěr

Zvláště v oblasti „velkého businessu" jsou systémy pro podporu rozhodování důležitou součástí každodenní práce managementu, OLAP má proto mezi databázemi důležité místo. Tato práce se věnovala letmému úvodu do problematiky s popsáním základních komponent architektury klientského přístupu pomocí ADOMD.Net. Poznali jsme jazyk MDX, technologii XMLA a objektový model knihovny samotné.

Obsah

Zdroje

[Lacko] Ľuboslav Lacko: Analytické možnosti produktu Microsoft SQL Server 2000, elektronická brožura, MSDN CZ

[Youness] Sakhr Youness: USING MDX and ADOMD to Access Microsoft OLAP Data, elektronická brožura

[SQLServer] Dokumentace k MS SQL Serveru 2000

[ADOMDNet] Dokumentace ADOMD.Net 1.0 Beta

[XMLA] Dokumentace XML for Analysis 1.1 Beta

[TechEd] Tom Conlon, Dave Wickert: Developing Client Applications with ADOMD.Net, prezentace z konference Microsoft TechEd 2003