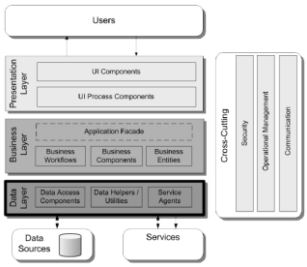
Object-relational mapping

Mapowanie obiektowo-relacyjne.

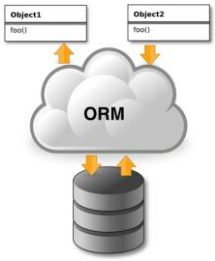
Architektura wielowarstwowa



Źródło: MS Application Architecture Guide v 2.0a

Czym jest technika ORM

Mapowanie obiektowo-relacyjne (ORM, O/RM, oraz O/R mapping) w ujęciu inżynierii oprogramowania jest techniką pozwalającą na konwersje danych pomiędzy heterogenicznymi systemami: relacyjną lub obiektowo-relacyjną bazą danych oraz obiektowo zorientowanym językiem programowania i na odwrót.



Relacyjna baza danych a programowanie obiektowe

Relacyjna baza danych

- tabele (zbiory krotek)
- relacje (więzy integralności odwołań)
- typy danych (varchar, timestamp, itp.)
- transakcje
- strukturalny język zapytań

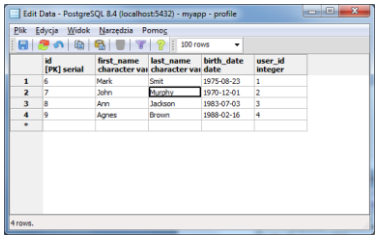
Obiektowy język programowania

- klasy
- obiekty
- uogólnienia
- typy

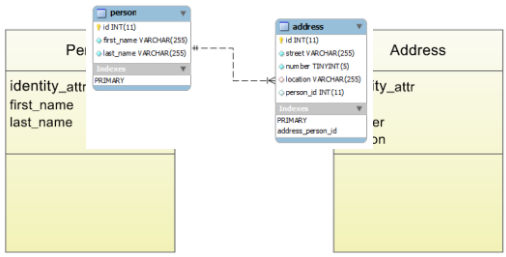
- Cel - uzyskanie:
  - abstrakcji danych
  - enkapsulacji
  - polimorfizmu

Obiekty

```
Class Person(object):
    def display(self):
        return '%s, %s' % (self.last_name, self.first_name)
>>> e = Person()
>>> e.first_name
= 'John'
>>> e.last_name
= 'Murphy'
>>> print(e.display())
Murphy, John
```



Co chcemy uzyskać



Jak to osiągnąć

- Dane:
  - odczyt, zapis, usuwanie, uaktualnianie
  - mapowanie i walidacja typów
- Relacje
  - Mapowanie relacji:
    - jeden do jednego
    - jeden do wielu
    - wiele do wielu
- Praca współbieżna
- Inne..

Jak to osiągnąć

- Relacyjna baza danych:

created_date timestamp with time zone
2010-04-29 22:14:32+02
2010-04-29 22:15:16+02
2010-04-29 22:16:07+02
2010-04-29 22:16:37+02

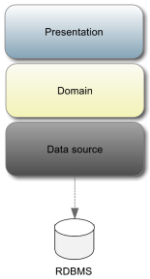
- Model obiektowy

```
>>> type(thing.created_date)
<class 'datetime.datetime'>
>>> print(thing.created_date.year)
2010
>>> type(thing.created_date.year)
<class 'int'>
```

Sposób implementacji

- Modelowanie w oparciu o bramy dostępu
- Aktywne rekordy
- Wykorzystanie wzorca *Mapper*

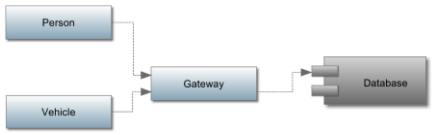
Wielowarstwowość aplikacji



Modelowanie w opraciu o bramy

Głównym zadaniem bramy (ang. *Gateway*) jest hermetyzacja dostępu do zewnętrznego zasobu oraz jednoczesne udostępnienie interfejsu programistycznego.

Modelowanie w opraciu o bramy



Modelowanie w opraciu o bramy

Dwa kluczowe podejścia:

- *Table Data Gateway*
- *Row Data Gateway*

Table Data Gateway

jeden obiekt

cała tabela (kolekcja rekordów)

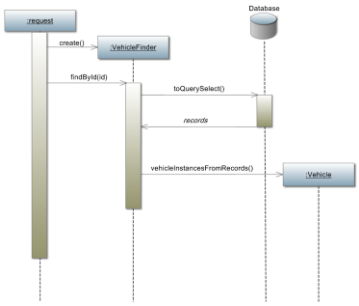
Row Data Gateway

jeden obiekt

jedna krotka

- obecność tzw. wyszukiwaczy

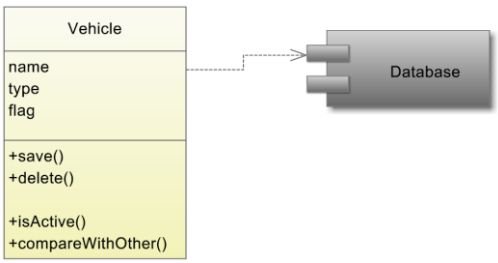
Row Data Gateway - demonstracja



Aktywne rekordy

Rolą schematu Active Record jest zapewnienie osłony dla wiersza tabeli, hermetyzacja operacji dostępu do zgromadzonych tam danych oraz realizacja elmentów logiki dziedziny.

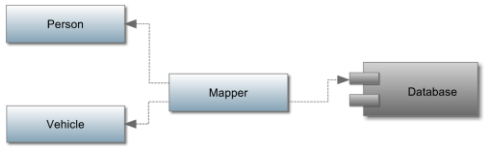
Aktywne rekordy



Odwzorowanie danych (*Mapper*)

Traktowany jest on w systemie jako dodatkowa warstwa pośrednicząca w wymianie danych pomiędzy dwoma obiektami reprezentującymi niezależne podsystemy. Jego zadaniem jest całkowita kontrola nad przepływem danych i ich adaptacją dla poszczególnych środowisk, przy czym żaden z podsystemów nie powinien wiedzieć o jego istnieniu.

Odwzorowanie danych (*Mapper*)



Modelowanie zachowania

- Identity field
- Foreign key mapping
- Association Tabel Mapping
- Metadata Mapping
- Query Object
- Lazy Load
- Optimistic/Pessimistic Offline Lock

Przykładowe narzędzia *O/R Mapping*

- (Java) Hibernate
- (Java) ObJectRelationalBridge
- (Python) PyDO
- (Perl) Tangram
- (C++) SourcePro DB
- (.NET) GENOME O/R mapper
- (Ruby) ActiveRecord
- (PHP) Doctrine

- i inne..

Technologie dostępu do danych w .NET

Microsoft:
- ADO.NET Core
- ADO.NET Data Services Framework
- ADO.NET Entity Framework
- ADO.NET Sync Services
- LINQ to SQL

Non-MS:
- Nhibernate (FluentNH)
- LLBLGen
- inne

Scenariusze użycia

- Mapowanie obiektowo-relacyjne
  - *ADO.NET EF, LINQ to SQL*
- SOA
  - *ADO.NET Core, ADO.NET Data Services*
- Architektury wielowarstwowe
  - *ADO.NET Core, ADO.NET Data Services, ADO.NET EF, LINQ to SQL*

ADO.NET Core

- tradycyjny dostęp do danych via SQL
- niskopoziomowe API
- maksymalna kontrola
- największa wydajność

ADO.NET Core API

Provider	Przedrostek
ODBC Data Provider	Odbc
OleDB Data Provider	OleDB
Oracle Data Provider	Oracle
MS SQL Data Provider	Sql
Borland Data Provider	Bdp

- SqlConnection
- SqlCommand
- SqlDataReader
- SqlDataAdapter
- DataSet

LINQ to SQL – Czym jest LINQ?

LINQ – Language Integrated Query

Przykład:

```
List<Node> boundaryNodes = (from n in _nodes
                            where n.Coordinates.FirstCoord == 0 ||
                            Math.Abs(n.Coordinates.FirstCoord - _width) < 0.0001 ||
                            n.Coordinates.SecondCoord == 0 ||
                            Math.Abs(n.Coordinates.SecondCoord - _height) < 0.0001
                            select n).ToList<Node>();
```

LINQ to SQL

LINQ to SQL – O/RM dostępny od wersji 3.5 frameworka .NET. Cechy:

- operuje na kontekście danych
- silnie typowany dostęp do danych
- proste mapowania 1:1
- tylko MS SQL Server

```
NorthwindDataContext db = new NorthwindDataContext();

var products = from p in db.Products
               where p.Category.CategoryName == "Beverages"
               select p;
```

LINQ to SQL – zalety i wady

Zalety:

- prostota
- szybki setup
- zapytania LINQ

Wady:

- mapowanie 1:1
- niewygodne obiekty
- konieczność regenerowania kontekstu

ADO.NET Entity Framework

ADO.NET EF – O/RM dostępny w pełnej funkcjonalności od wersji 4.0 frameworka. Cechy:

- operuje na kontekście danych
- silnie typowany dostęp do danych
- docelowo wszystkie funkcjonalności L2S
- model dostępu do danych oparty o providery
- Obsługa POCO
- Złożone scenariusze mapowania

ADO.NET Entity Framework – zalety i wady

Zalety:
- możliwość wykorzystania POCO
- obsługa dziedziczenia
- decoupling struktury bazy i logiki
- oparty o providery

Wady:
- konieczność regenerowania kontekstu

ADO.NET Entity Framework vs. LINQ to SQL

LINQ to SQL:
- proste scenariusze, mapowanie 1:1

ADO.NET EF:
- aplikacje typu enterprise
- duże wymagania co do mapowania
- architektury n-tier