Projektowanie obiektowe oprogramowania Wykład 12 – Wzorzec repozytorium (2) Wiktor Zychla 2012

Repository – strategia w warstwie dostępu do danych.

Zalety wprowadzenia repozytorium jako warstwy izolującej dostęp do danych:

- Uniezależnienie warstwy przetwarzania danych (logika biznesowa) od implementacji warstwy dostępu do danych
- Umożliwienie łatwego zastępowania implementacji repozytorium – testy jednostkowe warstw przyległych!

Wady wprowadzenia repozytorium:

- Dodatkowa warstwa w architekturze aplikacji
- Dobra implementacja generycznego repozytorium może być trudna

Przykładowy generyczny interfejs repozytorium:

```
public interface GenericRepository<T>
{
    IEnumerable<T> RetrieveAll();
    T RetrieveSingle( int Id );
    void Insert( T item );
    void Update( T item );
    void Delete( T item );

    IQueryable<T> Query { get; }
}
```

Podczas wykładu zobaczymy przykład na żywo budowania warstwy repozytorium dla trzech przykładowych technologii mapowania obiektowo-relacyjnego : Linq2SQL, Entity Framework i NHibernate. Użyjemy kontenera IoC do konfiguracji wybranej implementacji i zobaczymy, że kod klienta (warstwy logiki biznesowej) może w ogóle nie zmieniać się przy wymianie warstwy dostępu do danych.

Podstawowa trudność jaka pojawia się przy takiej implementacji polega na tym, że niektóre technologie mapowania OR tworzą własne klasy modelu dziedzinowego. Z kolei wymienne repozytorium nie może więc zależeć od żadnej konkretnej implementacji klas modelu dziedzinowego.

Jak rozwiązać ten problem? Literatura sugeruje żeby posłużyć się wzorcem ViewModel (patrz Mark Seemann, „Dependency Injection in .NET”), jednak to nie jest dobry pomysł z uwagi na brak możliwości implementacji leniwych zależności typu parent-child w klasach modelu.

Na wykładzie pokażemy, że właściwe rozwiązanie polega na opisaniu modelu przez interfejsy klas, a następnie implementacji repozytoriów względem interfejsów klas modelu dziedzinowego.