İSTANBUL TEKNİK ÜNİVERSİTESİ
ELEKTRİK-ELEKTRONİK FAKÜLTESİ

.NET VE SSCLI' DA VERİTABANI KATMANLARININ TÜMLEŞİK OLUŞTURULMASI

Bitirme Ödevi

İlker NACAKLI
040020358

Bengü DEREN
040020320

Bölüm : Bilgisayar Mühendisliği
Anabilim Dalı : Bilgisayar Bilimleri

Danışman : Yrd. Doç. Feza Buzluca

Mayıs 2007

Özgünlük Bildirisi

1. Bu çalışmada, başka kaynaklardan yapılan tüm alıntıların, ilgili kaynaklar referans gösterilerek açıkça belirtildiğini,
2. Alıntılar dışındaki bölümlerin, özellikle projenin ana konusunu oluşturan teorik çalışmaların ve yazılım/donanımın benim tarafımdan yapıldığını bildiririm.

İstanbul, 2007

İlker NACAKLI Bengü DEREN

.NET VE SSCLI' DA VERİTABANI KATMANLARININ TÜMLEŞİK OLUŞTURULMASI

(ÖZET)

Günümüzde geliştirilen yazılım projelerinin çok büyük bir kısmı veritabanı uygulamaları içermektedir. Programcılar uygulamalarını geliştirdikleri süreç içerisinde veritabanı işlemlerini gerçekleştirmek ve proje kodunu yazmak için farklı uygulamalar kullanmaktadırlar. Bu süreçte, veritabanı ve kod arasında bir iletişim katmanına ihtiyaç duymaktadırlar. Her uygulamada aynı anda farklı iki platform kullanmak ve iletişimi sağlayacak katmanın, her projede benzer özellikler taşımasına rağmen baştan yazılması yazılımcılar için yorucu ve zaman kaybettirici olmaktadır. Bu şekilde tanımlanabilen problemin çözümü için üretilmiş olan ürünler ise kullanım zorluğu ve platform bağımlılığı gibi sebeplerden dolayı soruna tam bir çözüm getirememektedir.

Geliştirilen proje, yazılım projelerinde kullanılan veritabanıyla ilgili katmanların temel işlevlerinin, yazılımın geliştirildiği platformdan otomatik olarak oluşturulmasını sağlayarak, yazılımcıya zaman kazandırmaktır. Hazırlanan yazılım; veritabanı tablolarını, tablolarla ilgili temel prosedürleri ve prosedürlere erişmek için kullanılacak veriye erişim katmanını otomatik olarak oluşturmaktadır. Proje, hem .NET hem de SSCLI (Shared Source Common Language Infrastructure-ROTOR) platformlarına entegre edilerek, yazılımcının yazılım geliştirdiği platformdan hem veritabanı sorgularını kontrol etmesi, hem de veriye erişim katmanını oluşturması sağlanmıştır.

Sonuç olarak her uygulamada tekrarlanmak zorunda kalınan işlemler kısaltılarak yazılımcıların daha hızlı program geliştirmeleri sağlanmıştır. Bunun yanı sıra oluşturulan veri tabanı elemanları ve veriye erişim katmanı belli bir standarda göre isimlendirildiği için birden çok yazılımcının çalıştığı projelerde yaşanabilecek olan uyum sorununun da önüne geçilmiştir. Son olarak, proje SSCLI' da da geliştirildiği için SSCLI projelerinde de kullanılabilecektir. Ayrıca proje içeriğinde yer alan SSCLI ile ilgili teknik bilgilerle ve SSCLI araçlarının örnek kullanımlarıyla SSCLI' da geliştirilecek bundan sonraki çalışmalara yardımcı olabilecek niteliktedir.

INTEDRATED AUTO-GENERATE DATABASE LAYERS IN .NET AND SSCLI

(SUMMARY)

Most of the software projects use database systems to store data that collected in the solution phase. That's why, software projects are developed in two fundamental layers. These fundamental layers generally have been generated from separate platforms. Layers that generated from different programs have conceptually different mechanism. This issue produces a necessity to use communication layer between distinct mechanisms. Software developer has to use these separate platforms in every phase of project and consume time by duplicate the similar process.

The main goal of this project is to generate the basic processes in database layers automatically from the software platform and save time for developer by atomize the similar jobs. This project generates the database tables, basic store procedures and data access layers that will be used for accessing these store procedures, automatically. Project is developed for both MS .NET and SSCLI (Shared Source Common Language Infrastructure-ROTOR) platforms so developer can manage database and data access layer processes from the platform that use to develop software.

Besides the main goal, this project generates database members and data access layer with a suitable naming convention. Because of this, developers that use this project are forced to fit these naming conventions and remove the conformity problems in team members.

In addition, project can be used for projects that are developed by SSCLI. Because of this property of project, it facilitates the project development in SSCLI by the help of auto-generated database layers.

At the beginning, projects that are results of the previous studies in this area, developed as a third-party tool (separate from the software platform). For example Alachisoft Company develops the *Tier Developer* program for auto-generate the database layers. Tier Developer, generates the database queries for store procedures belongs to database tables

that is taken from the database platforms. In addition to this process, it generates the data access layer in MS .NET with selected programming language.

Otherwise, Microsoft implements a component to the .NET Framework 2.0 that can auto-generate and manage the data access layer in the same platform.

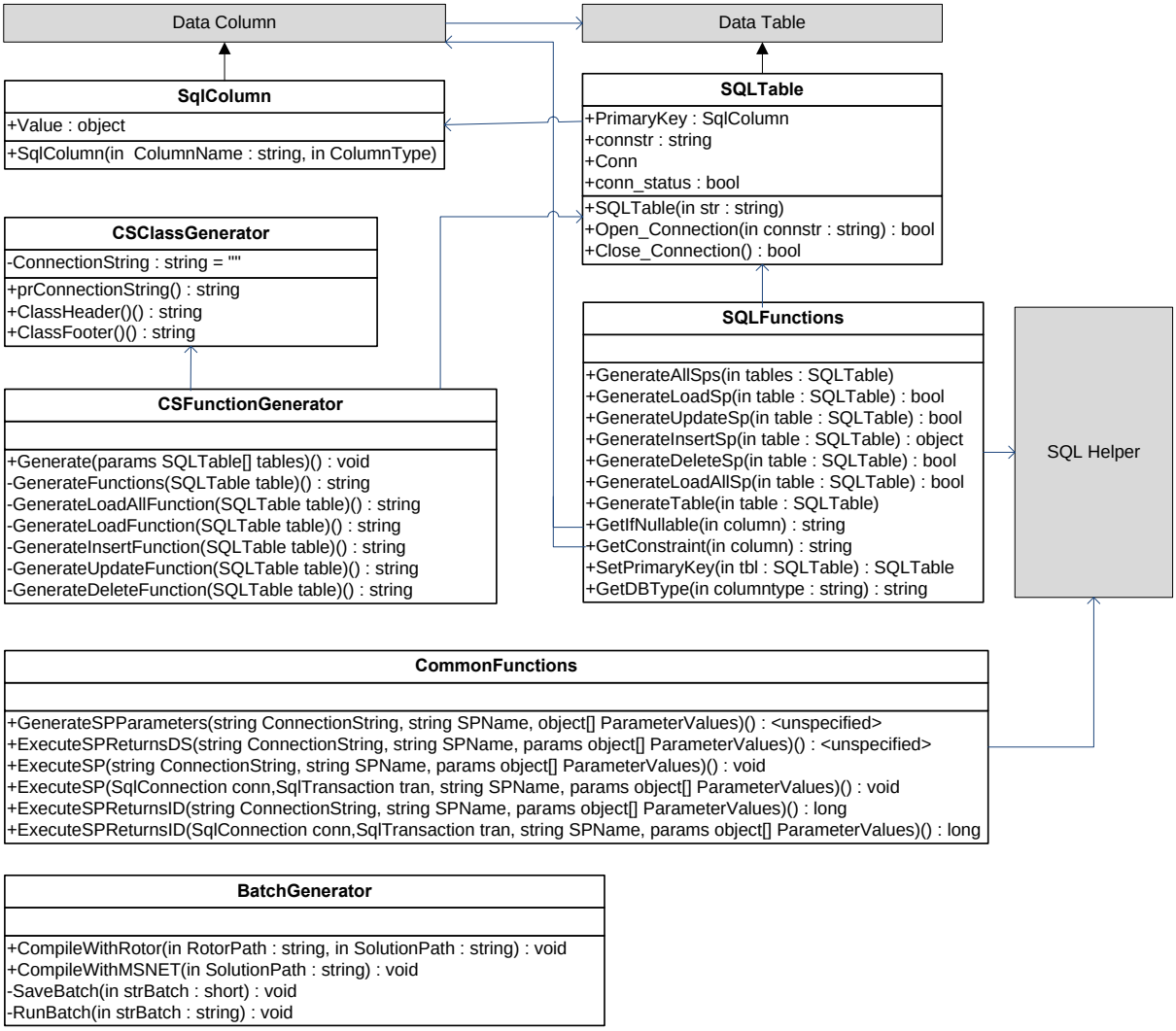
Through the aim of this projects, project is developed in six phase. These phases are education, analysis, architectural design, development, and documentation.

In the analysis phase problem is specified as a first step. As mentioned before, nowadays software projects, independent from the type of project (Web Application, Windows Application etc.) needs database systems. That's why these projects are generally developed in three layers. These layers are database layer that stores data, data access layer that provides access to stored data, and presentation layer that collects, and use this data. The aim of this stratified architecture is protect the whole system from specific damages by separates these layer and shortened the code by prevent to duplicate similar issues. Although these stratified architecture nearly the same for projects, programmer duplicate the similar job in every project by developing these architecture in the beginning.

In second step, project is modeled as a two main module to solve the problems that analysis.

First module is the Database Module. This module focuses on the database. It automatically generates the tables and the main stored procedures in the way the user chooses. Also this module creates a naming convention so that the programmers will not have high risk for the naming problem. The second module is the data access layer module which is designed for using the first module for the projects easily by the user. With the second the data access layer is automatically created by the project in C# language.

The class UML of the project can be seen below.



The Class UML diagrams

In the presentation layer of the project there will be an object created by giving the *SQLTable* class a connection string in sting format. After this the table's columns will be generated by giving the *SQLColumn* class the necessary parameters and these columns will be added to the table object. SQLFunctions class's GenerateTable function will be called with SQLTable classes objects in order to create the tables in database. The same table objects will be sent to GenerateAllSps function of *SQLFunctions* class in order to cerate the basic stored procedures. After the database actions completed, the table objects will be sent to *CSFunctionGenerator* class's *Generate* function. In this way the data access methods will be created by accessing the *CSClassGenerator* class's methods .The data access class will be created in C# language and the class will be saved to user's C: disc. After this process by using *BatchGenerator class* the user can choose which platform to

use in order to compile the data access class.The *CompileWithRotor* and *CompileWithMSNET* methods of BatchGenerator class can be used for this task. These methods will create and call the suitable batch files and save the data access class's dll to user's specified path. Finally the *CommonFunction* class is designed for the user to call data access class methods.

The project is tested with two different test programs. One of them is created with Microsoft .NET 2.0 the other one is created by using SSCLI 2.0. With both of the test projects two phases are passed. In the first phase the framework set up to different platforms and checked if it works correctly or not. In the second phase the data access layer which is created by the project in C# is tried on different platforms and two different test programs.

As a result of these test programs' results it is observed that with a table of three columns the framework creates almost a hundred lines of code automatically. It can be said that if a software project with more database tables is used by the developers the gain will increase. It is also an advantage that the framework's database objects can be used with under the same transaction. Besides the framework supports SSCLI 2.0 so that it can be used in this platform just like .Net 2.0 platforms.

There are some incomplete parts in the developed framework. One of the missing parts is that the framework does not support relational database. But the project can be developed in time and it is possible to add relation property to the project. With this volume of the project if the user creates the database tables the basic stored procedures can be automatically generated by the project despite not supporting the relational database.

The project does not support special stored procedures which are dependent to processes. But after creating the database tables, basic procedures and the data access layer by using the project the user can add the special procedures by using the database platform. Also the user can add the necessary methods to the data access layer.

The project supports MS SQL 2000 and MS SQL 2005. In the project the SQLHelper class is used in as lower data access layer. Because of this with adding some features for Oracle the framework can be used wih Oracle database platform,too.

The project does not give infotmation messages to the user.This feature can be added to the project in a very short time because the infrastructure is desined in early parts of the poject.

Despite using SSCLI in the project, the framework has only been tested on Windows operating system.SSCLI is dependent from operating system type so that with more effort the project can become pearting system independent.

Finally the project contains a lot of information about SSCLI .With this property it can be used as a guide for the projects which will use SSCLI in development.

İÇİNDEKİLER

1.GİRİŞ

Günümüzde geliştirilen yazılım projelerinin büyük çoğunluğu topladığı verileri depolamak için veritabanı kullanmaktadır. Bu nedenle yazılım projeleri iki ana katmandan oluşturulur. Bu katmanlar da genellikle iki farklı program yardımıyla oluşturulmaktadır. Katmanların farklı programlarda olması ve farklı işleyişlere sahip olması, katmanlar arasında bir iletişim katmanı ihtiyacını doğurmuştur. Programcı, projenin her aşamasında bu farklı katmanları farklı programlarda kullanmak durumunda kalmakta ve benzer işlemleri tekrarlayarak zaman kaybetmektedir.

Projenin temel hedefi, yazılım projelerinde kullanılan veritabanıyla ilgili katmanların temel işlevlerinin, yazılımın geliştirildiği platformdan otomatik olarak oluşturulmasını sağlayarak, yazılımcıya zaman kazandırmaktır. Hazırlanan yazılım; veritabanı tablolarını, tablolarla ilgili temel prosedürleri ve prosedürlere erişmek için kullanılacak veriye erişim katmanını otomatik olarak oluşturmaktadır. Proje, hem .NET hem de SSCLI (Shared Source Common Language Infrastructure-ROTOR) platformlarına entegre edilerek, yazılımcının yazılım geliştirdiği platformdan hem veritabanı sorgularını kontrol etmesi, hem de veriye erişim katmanını oluşturması sağlanmıştır.

Geliştirilen proje temel hedefinin yanında veritabanı elemanlarını ve veriye erişim katmanını belirli bir isimlendirme standardına göre oluşturmaktadır. Böylece yazılımcıların, bu standarda bağlı kalarak yazılım geliştirmeleri sağlanarak grup projelerinde yaşanan uyum probleminin önüne geçilmesi amaçlanmıştır.

Ayrıca proje SSCLI' da da geliştirildiğinden bundan sonra yapılacak SSCLI projelerinde kullanılabilecektir. Proje bu özelliğiyle, geliştirme zorluğu ve kaynak yetersizliği nedeniyle sınırlı sayıda geliştirilmiş olan SSCLI projelerinin, veri katmanının otomatik olarak oluşturulması sayesinde kısmen kolaylaştırmaktadır.

Bu alanda yapılmış olan çalışmalar incelendiğinde ilk olarak veri katmanını otomatik oluşturan programlar yazılım platformundan ayrı özel paket programlar olarak geliştirilmiştir. Bunlara *Alachisoft* şirketi tarafından geliştirilen *Tier Developer* örnek olarak gösterilebilir. *Tier Developer* programı, farklı veri tabanı sistemlerinden veri tabanı nesnelerini alarak bu nesnelere ait temel prosedürleri oluşturacak sorguları hazırlamaktadır. Bunun yanı sıra Microsoft .NET' te veriye erişim katmanını çeşitli dillerde oluşturmaktadır. *Alachisoft* şirketinin ticari web sitesinde *Tier Developer*'ın geliştirme süresini yarıya indirdiğini iddia etmektedir.[1]

Ayrıca Microsoft .NET Framework 2.0' da getirilen bir özellikle veriye erişim katmanını oluşturan ve bu katmanı yöneten bir yapı Framework yapısına eklenmiştir. Microsoft eğitim dokümanında veri tabanı erişimi olan projelerde genel olarak çok katmanlı yapı kullanıldığı ve sunum, veri erişimi, veritabanı olmak üzere kullanılan yapının katmanlandırılarak birbirinden daha bağımsız çalışmasının sağlanması, bu eklentinin amacını olarak belirtilmiştir.[2]