Разработка многоуровневой инфраструктуры веб-сайта с использованием ASP.NET

А. С. Шкуркин
Филиал Кемеровского государственного университета
в г. Анжеро-Судженске, Россия
e-mail: shkurkin@asf.ru

Layered infrastructure for a web site is developed using ASP.Net. It consists of the 4 layers: Data Storage Layer, Data Access Layer; Business Logic Layer; Presentation Layer (User Interface).

1. Постановка задачи

При проектировании распределенных приложений необходимо решить, в каком виде представить бизнес-данные, с которыми работает приложение, как связаны элементы управления с уровнем бизнес-логики и каким образом осуществляется доступ к данным. Типичная структура распределенного приложения при доступе к бизнес-данным с помощью веб-страниц с использованием ASP.Net состоит из четырех уровней (рис. 1):

— уровень данных;

— слой доступа данных;

— слой бизнес-логики;

— интерфейс пользователя.

Для отделения логики доступа к данным от самих данных используют компоненты слоя доступа данных (Data Access Logic), которые считывают информацию из базы данных и сохраняют в базе данных информацию бизнес-объекта. DAL-компоненты содержат методы, позволяющие по усмотрению вызывающего решать при работе с базой данных следующие задачи:

— создавать записи в базе данных;

— читать записи из базы данных и возвращать данные бизнес-объектов вызвавшему компоненту;

— обновлять записи базы данных измененными данными бизнес-объектов, переданными вызвавшим компонентом;

— удалять записи базы данных.

Каждый DAL-компонент работает с определенным типом бизнес-объектов, которые обладают следующими характеристиками.

• Бизнес-объекты предоставляют программный доступ с поддержкой состояний к бизнес-данным и (в некоторых случаях) к функциональности, связанной с этими данными.

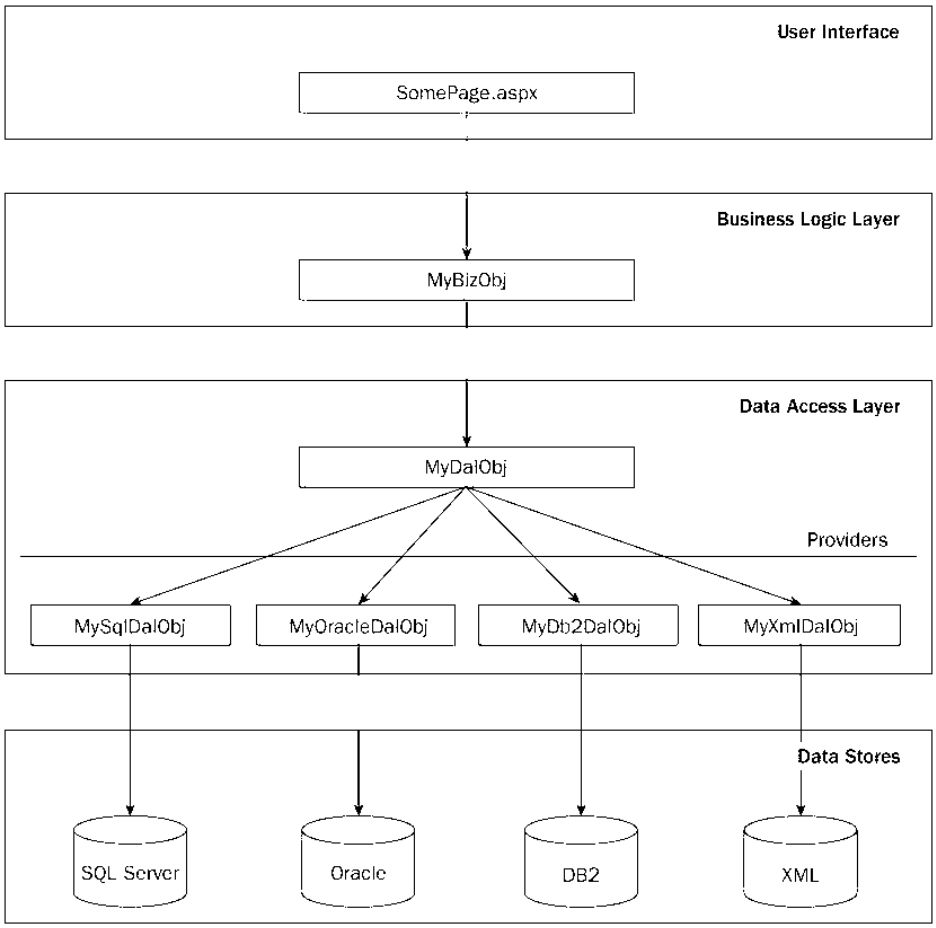

Рис. 1

• Бизнес-объекты могут формироваться по данным со сложными схемами. Данные обычно берутся из нескольких связанных таблиц базы данных.

• Данные бизнес-объектов могут передаваться в составе параметров ввода-вывода бизнес-процессов.

• Бизнес-объекты можно сериализовать, чтобы сохранять текущее состояние объектов.

• Бизнес-объекты не обращаются к базе данных напрямую. Весь доступ к базе данных осуществляется через связанный с бизнес-объектом DAL-компонент.

• Бизнес-объекты не инициируют никакие виды транзакций. Транзакции инициируются приложением или бизнес-процессом, работающим с бизнес-объектами.

Способ представления данных в приложении и способ передачи данных между уровнями не обязательно должны совпадать. Однако использование целостного и ограниченного набора форматов повышает производительность и обеспечивает удобство поддержки, так как сокращается количество дополнительных уровней преобразования.

2. Базовый класс для всех классов слоя доступа данных

Абстрактный базовый класс — это класс в объектно-ориентрированном программировании, который обеспечивает общее функциональное назначение, которое нужно дру-

гим классам. Поскольку он абстрактный, вам не нужно обеспечивать реализацию для каждого метода, но вы можете обеспечить некоторые реализации метода, если делаете их виртуальными. Каждый модуль сайта будет иметь собственный абстрактный базовый класс DAL и один или более классов поставщика, которые обеспечивают специфическую реализацию RDBMS. Тем не менее все абстракции базовых поставщиков наследуются из базового класса. Это класс DataAccess, обеспечивающий несколько свойств, которые считываются из файла web.config, и основные методы объекта DbCommand (SqlCommand, OleDbCommand, OracleCommand и т. п.): ExecuteNonQuery, ExecuteReader и ExecuteScalar.

На рис. 2 представлено отношение наследования между этим базовым классом DataAccess, абстрактным классом поставщика и конкретной реализацией.

Первый абстрактный класс DataAccess — общая база всех классов DAL, и он содержит вспомогательные методы, которые относятся к различным реализациям DAL. Основной интерфейс DAL определен другим абстрактным классом — MyDalObj, который определяет функциональное назначение (свойства и методы), требующиеся всем специфическим реализациям DAL.

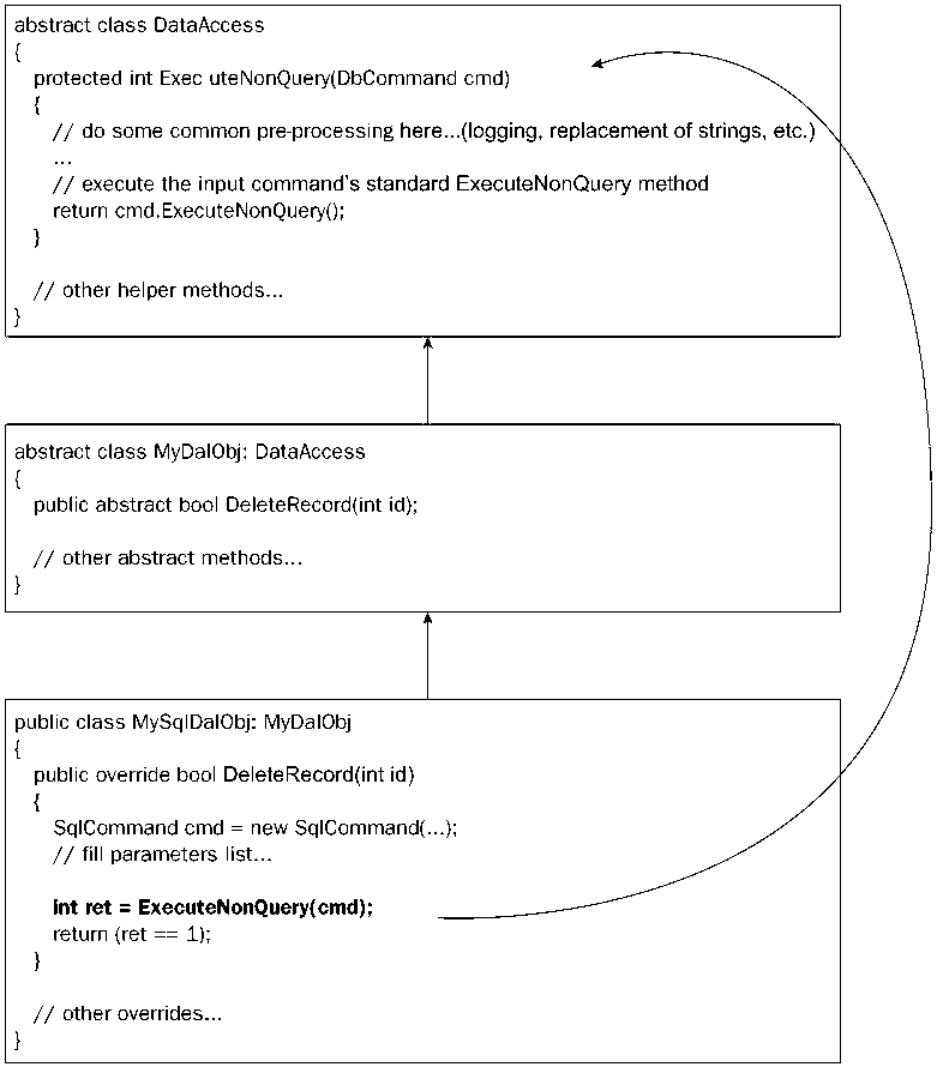

Рис. 2

В работе рассматривается одна реализация DAL для SQL Server — MySqlDalObj (рис. 2). Функция ExecuteNonQuery в базовом классе DataAccess — специальный вспомогательный метод, который может быть использован для перевода имени загруженной процедуры или имени таблицы в текстовом запросе SQL в соответствующие обозначения для конкретной базы данных, используемой веб-сайтом.

3. Проектирование слоя бизнес-логики

Данные, возвращаемые DAL, — все еще исходные данные, слой бизнес-логики (BLL) преобразует эти данные и предоставляет их слою интерфейса пользователя UI, а также добавляет логику подтверждения, вычисляемые свойства и статические методы, чтобы удалять, редактировать, добавлять и извлекать данные. Это объектно-ориентированное представление любых данных обеспечивает чрезвычайно прочную абстракцию базы

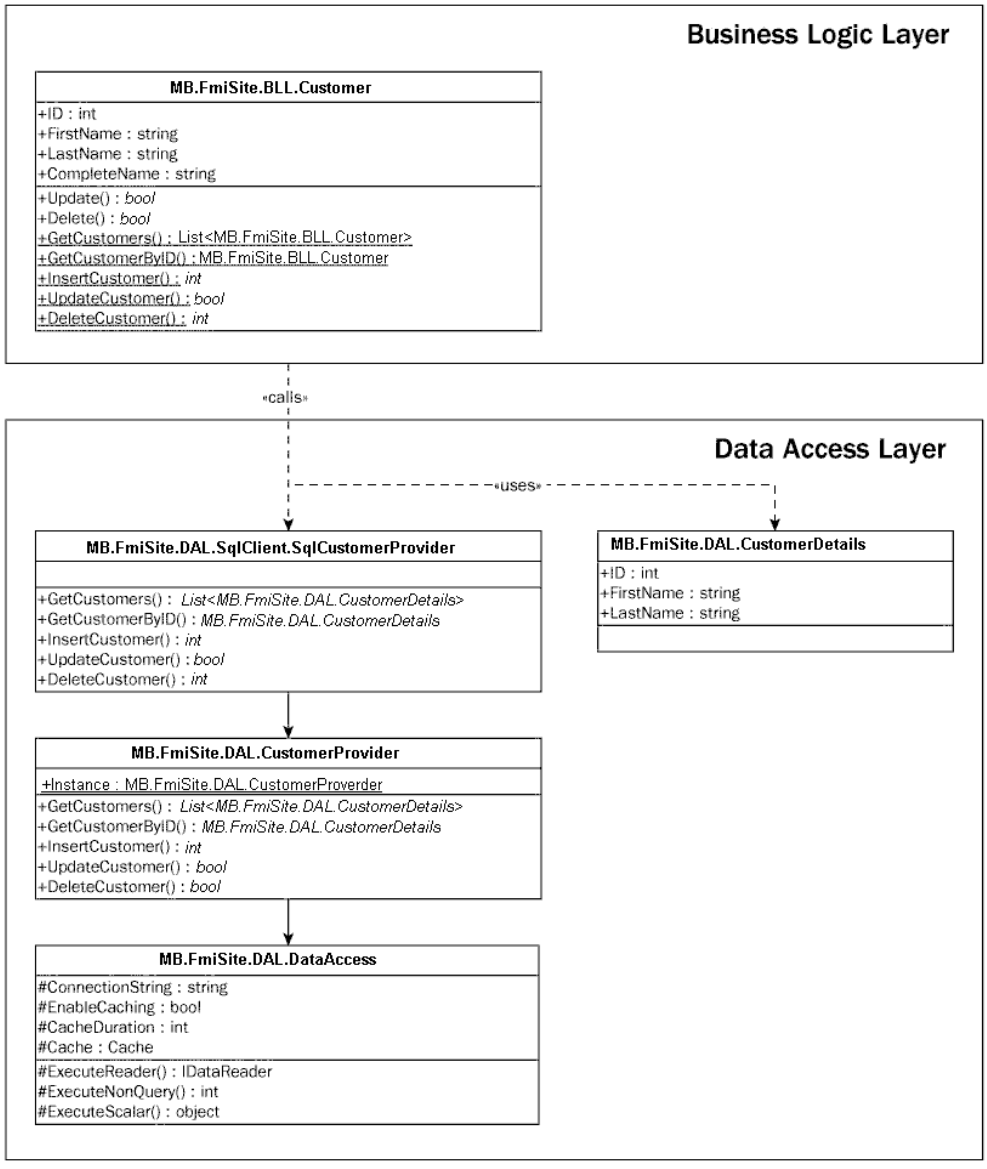

Рис. 3

данных, а также простую и мощную установку классов для разработчика UI без необходимости знать любые детали о том, как и где исходные данные будут загружены, сколько таблиц в базе данных или какая связь существует между ними. На рис. 3 представлено отношение между поставщиками DAL's и классом сущности, а также доменными объектами BLL's. Как видно, бизнес-класс Customer имеет те же свойства класса сущности DAL's CustomerDetails, а также вычисленное свойство только для чтения CompleteName, возвращающее конкатенацию FirstName и LastName. К тому же у него есть пара методов, чтобы удалять и корректировать данные, представленные специфическим объектом, и множество статических методов, чтобы извлекать список объектов Customer. На основе рассмотренной реализована структура сайта факультета информатики, экономики и математики (`http://fmi.asf.ru`).

Список литературы

[1] Crocker A., Olsen A., Jezierski E. Designing Data Tier Components and Passing Data Through Tiers. 2002.
`http://www.microsoft.com/downloads/release.asp?ReleaseID=44269`

Поступила в редакцию 28 марта 2008 г.