Duwamish 和 Petshop 数据访问层设计方案比较

Comparison of Duwamish and Petshop Data Access Layer Design

乔　明　胡荷芬　刘凡君　（上海师范大学计算机应用技术 上海 200233）

摘　要： 数据访问层的好坏直接影响到应用程序性能的发挥，本文从研究基于 ADO. NET 的数据访问层设计出发，首先分别分析了 Duwamish 和 PetShop 数据访问层的设计和实现，并在此基础上比较 Duwamish 和 PetShop 两个应用程序的数据访问层数据访问模式和数据访问策略，得出它们各自的优缺点和应用环境。

关键词： N 层架构 数据访问层 ADO. NET PetShop Duwamish

1　引言

N 层架构（N – Tier Architecture）最初是为了解决与传统的客户端/服务器应用程序的相关问题而出现的。随着软件技术的迅速发展，这一体系结构逐渐成为设计主流。N 层架构的核心思想是，将整个应用程序划分为表示层、业务层、数据访问层、数据库，明确地将客户端的表示层、业务逻辑访问、和数据访问及数据库访问划分出来。

分层式设计可以达到分散关注、松散耦合、逻辑复用、标准定义的目的。其中，数据访问层的作用将所有对数据库操作的有关过程业务分离出来，将物理数据的逻辑视图提交给业务层。使业务层只需实现业务逻辑即可，无需考虑数据的存储细节。对一个好的数据访问层，当数据库的结构等发生改变时，只需要对数据访问层的代码进行修改就可以了，不需要再修改其他的地方，这样会方便和不同的数据库进行打交道。

2　ADO. NET 体系架构

在. NET 下的开发的应用程序，数据访问层几乎都用到了 ADO. NET 数据访问技术。ADO. NET 是一种处理数据的新方法，它提供了对 Microsoft SQL Server、OLE DB 和 XML 等公开数据源的一致访问，并进行检索和更新数据库操作。ADO. NET 建立在. NET Framework 提供的平台之上，在很大程度上封装了数据源访问和数据操作的动作。

ADO. NET 包含两个核心组件[1]：数据提供程序和数据集。它的体系结构如图 1 所示。

（1）数据提供程序（Data Provider）是专门为数据处理以及快速地只进、只读访问数据而设计的组件。主要功能是在数据源应用程序中用与数据源的活动连接访问数据和通过数据适配器（DataAdapter）给独立的数据集（DataSet）发送和接受数据。

（2）数据集（DataSet）是支持数据以关联的方式，在断开连接的情况下缓存数据，并且可以更新数据。DataSet 是数据的内存驻留表示形式，无论数据源是什么，它都会提供一致的关系编程模型。它是专门为独立于任何数据源的数据访问而设计的。

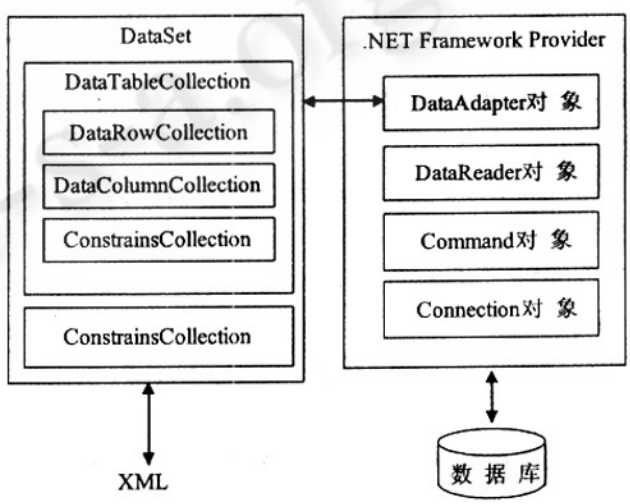

图 1 ADO. NET 的体系结构

3　Duwamish

Duwamish 是一家在网上销售图书的虚拟公司。其模型是典型的网上购物实践中最为普遍的电子商务企业对客户（B2C）模式。Duwamish 则采用的是一个

四层应用结构,并使用不同的项目分隔开,分别为表示层(web 项目),业务外观层(BussinessFacade 项目),业务规则层(BussinessRule 项目)和数据访问层(DataAccess 项目)。

3.1 业务实体

在介绍数据访问层之前,必须现了解一下作为业务实体。业务实体是业务对象模型、领域模型中和业务相关的实体,作为数据的的载体在数据访问层和业务外观层之间,以及数据访问层和业务规则层之间传输的对象。业务实体放在 Common 项目中。在 Common 项目中有 4 个业务实体,BookData、CategoriesData、OrderItems、CustomerData,下面是 BookData 类的示例代码[2]。

```
public class BookData : DataSet //继承自 DataSet
{
    ... ...
    //通过构造函数,调用 BuildDataTabe( )
    public BookData( )
    {
        BuildDataTables( );
    }

    //BookData 具有 DataSet 的架构,因此可以在其中封装一个或多个数据表
    private void BuildDataTables( )
    {
        DataTable table = new DataTable(BOOKS_TABLE);
        DataColumnCollection columns = table.Columns;
        columns.Add(PKID_FIELD, typeof(System.Int32));
        columns.Add(TYPE_ID_FIELD, typeof(System.Int32));
        ... ...
        this.Tables.Add(table);
    }
}
```

3.2 数据访问层

数据访问层放在 DataAccess 项目中,在 DataAccess 项目中包含 4 个类,Books、Categories、Customers、Orders,下面是 Books 类的示例代码。

```
public class Books : IDisposable
{
    //在构造函数中初始化 datasetcommand 对象
public Books( )
{
        dsCommand = new SqlDataAdapter( );
        dsCommand.SelectCommand = new SqlCommand( );
        dsCommand.SelectCommand.Connection = new SqlConnection
(DuwamishConfiguration.ConnectionString);
        //数据库中的列和数据集建立映射
        dsCommand.TableMappings.Add("Table", BookData.BOOKS_TABLE);
    }
    //用指定书号检索所有书籍
public BookData GetBooksByISBN(String searchText)
    {
        //向 FillBookData 方法传入三个参数:"GetBookByISBN"是存储过程的名称,"@ISBN"
        //是参数名称,"searchText"是参数值
        return FillBookData("GetBooksByISBN", "@ISBN", searchText);
    }
}
```

通过上述代码,我们可以清楚的看到:数据访问层与业务外观层和业务规则层是通过类型华的 DataSet 来通信的。每一个业务实体对应一个数据访问组件,保证了良好的封装性和可维护性。另外,程序中大量使用了存储过程,一方面提高了应用程序的性能,另一方面也可在其中加入业务规则,为编写程序提供方便。

4 Petshop

PetShop 是一家网上的宠物商店。它采用的是一个三层应用结构,分别是表示层,业务逻辑层,数据访问层。

4.1 数据实体

PetShop 中的数据实体,对应数据库中相应的数据表,它们作为数据的载体,便于业务逻辑对相应数据表

进行读/写操作。

图 2 OrderInfo 数据实体

数据实体没有行为，仅用于表现对象的数据。这些实体类都被放到 Model 程序集中，例如数据表 Order 对应的实体类 OrderInfo，其类图如图 2 所示。

由于数据访问层和业务逻辑层都将对这些数据实体进行操作，因此程序集 Model 会被这两层的模块所引用。

4.2 数据访问层

的数据访问层有 4 个项目：DALFactory 是数据抽象工厂，IDAL 是据访问层接口定义，SQLServerDAL 是 SQLServer 据访问层，OracleDAL 是 racle 数据访问层，BUtility 是据库访问组件基础类。数据访问层的模块结构如图 3 所示[4]。下面是 SQlSeverDAL 类的 GetOrder 方法的示例代码。

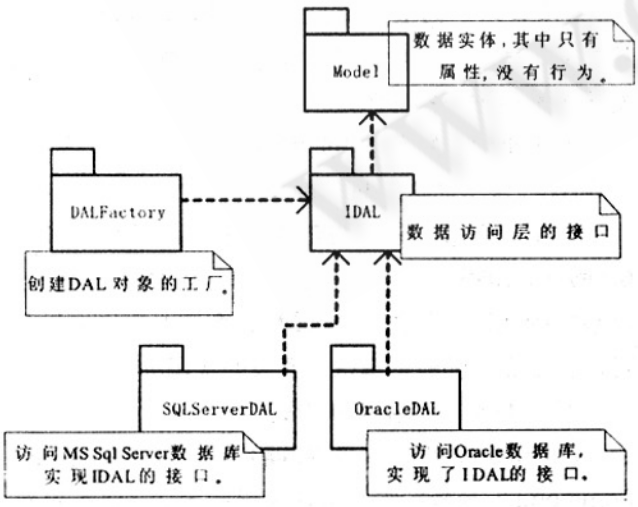

图 3 数据访问层模块结构图

```
public OrderInfo GetOrder(int orderId)
{
OrderInfo order = new OrderInfo();
  //创建一个参数并赋值
  SqlParameter parm = new SqlParameter(PARM_
ORDER_ID, SqlDbType.Int);
  parm.Value = orderId;
  //使用 Helper 类执行一个查询来读取 order
   using (SqlDataReader rdr = SqlHelper.Exe-
cuteReader
SqlHelper.ConnectionStringOrderDistributedTransaction,
CommandType.Text, SQL_SELECT_ORDER, parm))
  {
   if (rdr.Read())
   {
      //从第一行生成一个 order header
      ... ...
      order = new OrderInfo(orderId, rdr.GetDate-
Time(0), rdr.GetString(1), null, BilingAddress,
shippingAddress, rdr.GetDecimal(21), null, null);
      //创建列表
      IList<LineItemInfo> lineItems = new List<
LineItemInfo>();
      LineItemInfo item = null;
      //从第一行以及其后行创造 lineItems
      do
      {
         item = new LineItemInfo(rdr.GetString
(22), string.Empty, rdr.GetInt32(23), rdr.GetInt32
(24), rdr.GetDecimal(25));
         lineItems.Add(item);
      } while (rdr.Read());
  order.LineItems = new LineItemInfo[lineItems.
Count];
  lineItems.CopyTo(order.LineItems, 0);
  }
 }
  return order;
}
```

通过上述代码，我们来看一下数据访问层对数据

实体进哪些行操作。数据访问层首先用 DataReader 获取数据，接着创建数据实体类，根据字段类型填充数据实体类，然后将数据实体添加到列表类中（这一步仅针对返回超过一条数据的情况）。

这个过程与用 DataAdapter. Fill（）来填充 DataSet 中的 DataTable 所产生的效果大同小异，只不过，在 Fill（）中 DataAdpater 自动创建 DataReader 去获取数据，之后创建 DataTable（相当于数据实体类），并根据字段类型填充 DataTable，当然，如果可能返回多条记录，DataTable 完全可以处理，就没必要去实现列表操作了。

此时，我们可能有觉得既然用 DataAdapte. fill（）来填充 DataTable 的方法如此简洁，PetShop 中为何还需要数据实体类呢？

这是因为数据实体类是轻量级的 structure，一般仅包含数据字段，没有什么操作方法，这比 DataTable 或者 DataRow 有性能上的优势。

5 Duwamish 与 PetShop 数据访问层设计比较

5.1 数据访问模式

选择不同的数据访问模式必定会影响到应用程序性能的发挥。性能问题主要是为了解决由于软件的发展速度远远超过了硬件的发展速度所引发的瓶颈问题，它普遍存在于计算机系统架构和程序开发的方方面面。

Duwamish 采用的是 DataAdapter 和类型化的 DataSet 配合的数据访问模式，而 PetShop 采用的是 DataReader 和数据实体类配合的数据访问模式。

影响数据访问性能主要有以下 2 个因素：

① 构造和填充数据访问对象的所表现出的开销不同。例如，DataSet 对象实例化和填充的开销要比 DataReader 对象实例化的开销大。

② 应用程序与数据库服务器之间的连接时间和连接次数。

在 Duwamishp 中采用 DataAdapter 和类型化 DataSet 配合的数据访问模式。数据集是支持数据以关联的方式，在断开连接的情况下缓存数据。数据库连接是一项非常宝贵的服务器资源，数据集仅在被填充（Fill（））时才需要占用数据库连接，一旦填充完成，就可关闭数据连接。以后数据可在数据集（即在缓存）中读取，因为数据已被填充到缓存中，而不需要重新连接。当更改数据时，Dawamish 采用的 DataAdapter 的 Update 方法，将 DataSet 的改变一次性的提交到数据库中，可以一次性更新大批量数据，减少了数据库的连接次数。

因此，当并发访问数目超过一定数量而开始发生争用数据库连接时，采用 DateSet 能获得根好的性能。

在 PetShop 中采用而 PetShop 采用的是 DataReader 和数据实体类配合的数据访问模式。因为 DataReader 对于需要只读和只前移的数据访问来说是更好的选择，而且相较 Duwamish 而言，程序显得简洁而轻灵。

5.2 数据访问策略

数据访问策略是应用程序用来存储、检索和管理数据的方式。性能、部署和可伸缩性是计划和实现数据访问策略时应考虑的因素，它们可以最终影响结构模型。

(1) Duwamish 的数据访问策略

① 将处理转移到数据

为了将处理转移到数据，Duwamish 7.0 数据访问层对所有数据处理使用存储过程。好处是应用程序的数据访问层对数据库逻辑的更改更具灵活性[3]。

② 将数据库资源保留最短的时间

数据库资源稀有且昂贵。Duwamish 7.0 数据访问层尽可能推迟数据库资源分配并且尽可能快地释放数据库资源。相比之下，PetShop 在打开数据连接之后，并没有马上读取数据，而是将 DataReader 传递给另外的对象来执行数据读的操作，然后才关闭连接，浪费了宝贵数据连接资源，

(2) PetShop 的数据访问策略

① 提供了抽象的 Helper 类

Help 类包装了一些常用的操作[4]，如 ExecuteNonQuery、ExecuteReader 等方法。使用 Helper 类是一个比较好的策略，利用它来完成数据库基本操作的封装，可以减少很多和数据库操作有关的代码。

② 在线即时数据更新模式

PetShop 在数据中更新操作时，采用的是使用 Command 对象执行单个存储过程的方式来进行更新

（下转第 116 页）

(上接第 106 页)

操作,是属于一种在线即时数据更新模式。这种模式虽然较长时间的占用了数据库资源,但是它适合于在数据库改动非常频繁的情况下需要实时的跟踪数据变化。

6 小结

应用程序开发和应用都离不开数据访问。数据访问层的设计将数据访问完全封装提高了软件的开发效率,增强应用系统数据访问性能,加强数据库安全,提高软件的可重用性和维护性。duwamish 和 petshop 数据访问层的设计各有优缺点,在我们设计该层时需根据实际的应用环境审时度势,综合采用。

参考文献

1 James Huddleston 等著,杨浩译. C#数据库入门经典(第 2 版). 清华大学出版社,2006.

2 Visual Studio 示例:Duwamish 7.0[EB/OL]. http://msdn.microsoft.Com/library/chs/default.asp? url=/library/CHS/dwamish7/html/vtoriDuwamish70Overview.asp,2004.

3 郝刚. Duwamish 7 大剖析之数据访问. CSDN 开发高手,2003.

4 annel. petshop4.0 详解之二(数据访问层之数据库访问设计), http://blog.csdn.net/tannel/archive/2007/08/27/1760408.aspx,2007.