抽象工厂设计模式在 MIS 中的应用①

贾延明，张永涛

(商丘科技职业学院 计算机科学系，商丘 476000)

摘 要: 本文从设计模式出发，分析了抽象工厂模式(Abstract Factory Pattern)的优缺点，研究了抽象工厂设计模式在分层分布式系统中的应用，并将抽象工厂设计模式应用于管理信息系统(MIS)中。抽象工厂模式为系统结构提供了非常灵活强大的动态扩展机制，能够降低模块间的耦合性，更好的实现软件复用。

关键词: 设计模式；抽象工厂；MIS

Abstract Factory Design Pattern in the Application of MIS

JIA Yan-Ming, ZHANG Yong-Tao

(Shangqiu Science and Technology Vocational College, Shangqiu 476000, China)

Abstract: This article from the design pattern, analyzed the abstract factory pattern (Abstract Factory Pattern), the abstract factory design pattern in a hierarchy application of distributed systems, and abstract factory design pattern to management information system (MIS). Abstract factory pattern for the system architecture provides a very flexible and powerful dynamic extension mechanism, to reduce the coupling between modules, better implementation of software reuse.

Keywords: design pattern; the abstract factory; MIS

1 引言

设计面向对象软件比较困难，而设计可复用的面向对象软件就更加困难。必须找到相关的对象，以适当的粒度将它们归类，再定义类的接口和继承层次，建立对象之间的基本关系。你的设计应该对要解决的问题有针对性，同时对将来的问题和需求也有足够的通用性。

设计模式使人们可以更加简单方便地复用成功的设计和体系结构，将已证实的技术表述成设计模式也会使新系统开发者更加容易理解其设计思路，设计模式帮助你做出有利于系统利用的选择，避免设计损害了系统利用性。通过提供一个显式类和对象作用关系以及它们之间潜在联系的说明规范，设计模式甚至能够提高已有系统的文档管理系统维护的有效性，简而言之，设计模式可以帮助设计者更好更快的完成系统设计。

Christopher Alexander 说过：“每一个模式描述了一个在我们周围不断重复发生的问题，以及该问题的解决方案的核心。这样，你就能一次又一次地使用该方案而不必做重复劳动”[1]。尽管 Alexander 所指的是城市和建筑设计模式，但他的思想也同事样适用于面向对象设计模式，只是在面向对象的解决方案里，我们用对象和接口代替了墙壁和门窗。两类模式的核心都在于提供了相关问题的解决方案。

一般而言，一个设计模式有四个基本要素：

(1) 模式名称(pattern name)一个助记名，它用一两个词来描述模式的问题、解决方案和效果。

(2) 问题(problem)描述了应该在何时使用模式。

(3) 解决方案(solution)描述了设计的组成部分，它们之间的相互关系及各自的职责和协作方式。

(4) 效果(consequences)描述了模式应用的效果及使用模式应权衡的问题[2]。

2 抽象工厂设计模式概述

定义：提供一个创建一系列相关或相互依赖对象的接口，而无需指定它们具体的类。抽象工厂(Abstract

① 收稿时间:2010-03-25;收到修改稿时间:2010-06-21

Factory)模式又称为 Kit 模式，属于对象创建型模式[3]。此模式结构如图 1 所示。

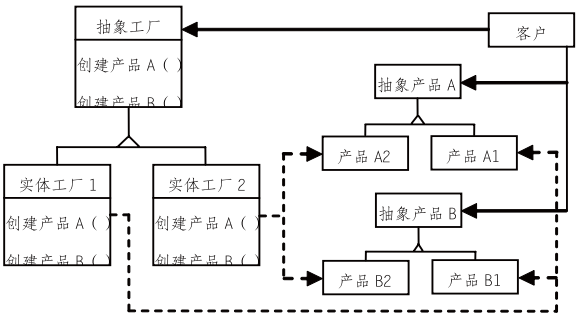

图 1 抽象工厂设计模式结构图

(1) 模式名

抽象工厂设计模式(Abstract Factory Design Pattern)。

(2) 问题

在以下情况可使用抽象工厂(Abstract Factory)模式

1) 一个系统要独立于它的产品的创建、组合和表示时。

2) 一个系统要由多个产品系列中的一个来配置时。

3) 当你要强调一系列相关的产品对象的设计以便进行联合使用时。

4) 当你提供一个产品类库，而只想显示它们的接口而不是实现时。

(3) 解决方案

抽象工厂

－声明一个创建抽象产品对象的操作接口。

实体工厂

－实现创建具体产品对象的操作。

抽象产品

－为一类产品对象声明一个接口。

实体产品

－定义一个将被相应的具体工厂创建的产品对象。

客户

－仅使用由抽象工厂和抽象产品类声明的接口。

(4) 效果

抽象工厂模式有下面的一些优点和缺点：

1) 它分离了具体的类。抽象工厂模式帮助你控制一个应用创建的对象的类。因为一个工厂封装创建产品对象的责任和过程，它将客户与类的实现分离。客户通过它们的抽象接口操纵实例。产品的类名也在具体工厂的实现中被分离。

2) 它使得交换产品系列变的简单。一个具体工厂类在一个应用中仅出现一次——即在它初始化的时候。这使得改变一个应用程序的具体工厂创建了一个完整的产品系列。它只需要改变具体的工厂即可使用不同的产品配置，这是因为一个抽象工厂创建了一个完整的产品系列，所以整个产品系列会立刻改变。

3) 它有利于产品的一致性。当一个系列中的产品对象被设计成一起工作时，一个应用一次只能使用同一个系列中的对象，这一点很重要，而抽象工厂很容易实现这一点。

4) 难以支持新种类的产品。难以扩展抽象工厂以生产新种类的产品。这是因为抽象工厂接口确定了可以被创建的产品集合。支持新种类的产品就需要扩展该工厂接口，这将涉及抽象工厂类及其所有子类的改变。

3 抽象工厂设计模式在MIS中的应用

3.1 应用背景

在软件设计过程中，尤其是在设计管理信息系统(MIS)时，经常会遇到针对不同子系统进行一系列相同或相似的复杂操作，而且对于所涉及的子系统的规模不能精确的评估。这就是说系统必须要有良好的可扩展性，以满足用户不断扩展的需求。同时，用户针对不同的子系统的操作必须要屏蔽系统内部数据通信的差异性，也就是说，让用户感觉不到不同子系统的区别，这就是抽象工厂设计模式的应用背景。在这个设计模式中，用户可以提出不同的需求，软件开发者通过给客户提供一个接口，客户不必指定具体产品的类型即可以创建一个系列中所有相关的对象。

3.2 具体应用

我们以一个具体管理信息系统(MIS)实例——商丘科技职业学院试题库建设与管理软件(Exam Paper System)入手，分析抽象工厂设计模式在该 MIS 中的应用。在该系统的具体结构和访问方式为同时支持 SQL Server 和 Oracle 数据库。在这种情况下，使用抽象工厂设计模式将能很好的解决这个问题。即实体工厂有两种：SQL Server 实体工厂和 Oracle 实体工厂，每一种实体工厂都可以创建出分属于不同产品等级结构的一个产品族中的所有对象。

试题库建设与管理软件通过在软件的三层结构中使用抽象工厂设计模式，使系统实现多数据库支持成为可能。该系统使用抽象工厂后的模式结构如图 2 所示：

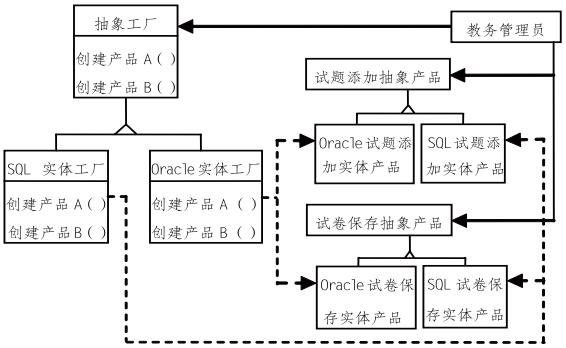

图 2 试题库建设与管理软件使用抽象工厂设计模式后的模式结构图

在具体项目中，添加 3 个类：抽象工厂(AbstractDALFactory)类，用以提供数据访问对象创建功能；SQL Server 具体工厂(SqlDALFactory)类，用以封装 SQL Server 访问对象的创建；Oracle 具体工厂(OracleDALFactory)类，用以封装 Oracle 访问对象的创建。实现抽象工厂类：

```
public abstract class AbstractDALFactory
{
    //创建工厂的选择应该用反射实现
        public static AbstractDALFactory ChooseFactory()
    {
        switch (dbType)
        {
            case "Sql":
                //创建 Sql 具体工厂
                    factory = new SqlDALFactory();
                break;
            case "Oracle":
                //创建 Oracle 具体工厂
                factory = new OracleDALFactory();
                break;
        }
        return factory;     //返回创建的实体工厂
    }
 //数据访问对象创建接口（抽象工厂提供抽象产品）
public abstract IAddPaperService CreateAddPaperService();
public abstract ISavePaperService CreateSavePaperService();
```

实现 SQL Server 具体工厂类：

```
public class SqlDALFactory : AbstractDALFactory
{
public override IAddPaperService CreateAddPaperService ()
    {
        //创建 Sql 方式的试题添加具体产品
        return new AddPaperService();
    }
public override ISavePaperService CreateSavePaperService ()
    {
        //创建 Sql 方式的试卷保存具体产品
        return new SavePaperService();
    }
}
```

实现 Oracle 具体工厂类：

```
public class OracleDALFactory : AbstractDALFactory
{
public override IAddPaperService CreateAddPaperService ()
    {
        //创建 Oracle 方式的试题添加具体产品
        return new AddPaperService();
    }
public override ISavePaperService CreateSavePaperService ()
    {
        //创建 Oracle 方式的试题添加具体产品
        return new SavePaperService();
    }
```

4 总结

抽象工厂设计模式是在面向对象对象编程中大量使用的一种设计模式，在系统设计中，复杂的设计模式并不是系统设计必须的选择。但是对于业务逻辑复杂，业务变动频繁的系统，在三层结构下采用抽象工厂设计模式可以使用系统层次更加分明，便于以后系统的维护与升级。同时可降低程序模块间的耦合性，最大限度的实现软件复用。当然它难以扩展抽象工厂种类以生产新种类的产品，因此，在设计初期应做好调研并做好软件规模的控制。

参考文献

1 阿博泰克北大青鸟信息技术有限公司.在.NET 框架下开发三层结构数据库应用程序.北京:科学技术文献出版社, 2008.103－105.

2 莫勇腾.深入浅出设计模式(C#/Java 版).北京:清华大学出版社, 2006.45－46.

3 Gamma E, Helm R, Johnson R, Vissides J. Design Pattern: Elements of Resuable Object-Oriented software. Beijing: China Machine Press, 2007.87－91.