结合实例探讨创建型设计模式

Research of Creational patterns in context

王方苏 张 毅 （重庆大学软件学院 重庆 400030）

摘要：设计模式是以一种让别人容易接受的方式，发现那些重复出现问题的解决方案。本文根据设计模式的使用经验，将设计模式在思想的体现上，结合一个实际的实例，借创建型设计模式为例进行讨论。达到对设计模式的理解不再孤立、片面的目的。

关键词：设计模式 创建型 面向对象

1 引言

目前，随着软件规模的日益增大，复杂性迅速增加，传统方法的软件开发设计出来的系统会存在过于僵硬、过于脆弱、复用率低、黏度过高这些方面的问题。于是当设计模式的出现时，许多开发者就开始了设计模式的学习。但是，有些人员在学习方面又遇到了孤立、片面的对待设计模式的问题，所以在系统中的模式应用方面，没有达到设计模式应有的威力。在此，我想用实例与设计模式场景结合的方式，以自己的观点，特以创建型模式为例，进行一定的归纳阐述。

2 概念

2.1 设计模式的概念

Christopher Alexander 提出："每一个模式描述了一个在我们周围不断重复发生的问题，以及该问题解决方案的核心，这样可以重复使用该方案而不必重复劳动。"这一思想同样适用于面向对象的设计模式。于是 GOF 在此基础上就进一步在软件领域中将设计模式分为了创建型、结构型、行为型三大类。如图 1 所示。

此处的特定情景十分重要性，可以说设计模式的应用直接依赖于特定情景，也就是我们通常所说的场景。这是相当多的设计模式者学习过程中容易忽略的地方，是导致看待设计模式孤立片面的根源所在。

2.2 创建型设计模式的概念

创建型模式－－－－－－就是用来创建对象的模式。它抽象了实例化对象的过程，帮助一个系统独立于如何创建对象、如何组合对象和表示它的那些对象。同时创建型设计模式有两个不变的本质特征：

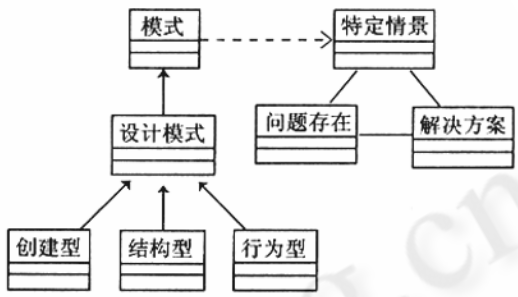

图 1

（1）它们都将关于系统使用哪些具体的类的信息封装起来。

（2）它隐藏了类的实例是如何被创建和放在一起的。

具体的说，创建型模式根据 GOF 的分类包括这样五种工厂模式（Factory Method）、抽象工厂模式（Abstract Factory）、单例模式（Singleton）、原型模式（Prototype）、构造器模式（Builder）它们的描述如表 1 所示。

3 创建型设计模式之间的异同点的阐述

抽象工厂模式和工厂模式的区别是初学使用设计模式时候的一个容易引起困惑的地方。实际上，抽象工厂模式是为创建一组相关或依赖的对象提供创建接口，而工厂模式是为一类对象提供创建接口或延迟对

象的创建到子类中实现。并且可以看到 Factory 使用的是继承机制，而抽象工厂使用的是组合的机制。

表 1

名称	描 述
工厂方法	Factory 模式不单是提供了创建对象的接口，其最重要的是延迟了一类子类的实例化。
单例	提供一种受约束的对象创建机制，确保指定类只有一个实例。
抽象工厂	允许为一组相关类中的类创建实例，不必让客户对象指定实际类。
原型	从现有的对象中复制对象来创建复杂的对象。
构造器	允许相同的构造过程生成不同的对象表示。

构造器模式要解决的问题：当我们要创建的对象很复杂的时候－－－－－通常是由很多其他的对象组合而成，我们需要将复杂对象的创建过程和这个对象的表示分离开来，这样做的好处就是，通过一步步的进行复杂对象的构建，使得经过相同的步骤创建最后得到的对象的展示不一样。

单例和原型模式意图较清晰不易混淆。

4 从现实生活中的实例来理解创建型设计模式

Kent Beck 在第一次程序设计模式语言大会上说过：最好的学习方法，就是通过实例来学习，这就是范例或动机在模式中如此重要的原因。如果把模式与实例联系起来，那么就会很容易地记住这些模式。下面以一个机械厂的发展史以及业务过程作为背景，再结合创建型设计模式体现的思想与运用场景，以我的观点来一步一步的对这五个创建型模式进行阐述。

例如，几年前，xxx 机械厂成立了，该厂的成立就标志着该厂的存在，并且是唯一的存在。

从这里的描述，再结合设计模式里的模式的符合场景，能够判断出这正是单例模式所符合的。如图 2 所示。

单例模式其实是一种职责型模式。因为我们创建了一个对象，这个对象扮演了独一无二的角色，在这个单独的对象实例中，它集中了它所属类的所有权力，同时它也肩负了行使这种权力的职责。如何绕过常规的构造器，提供一种机制来保证一个类只有一个实例这应该是类设计者的责任，而不是类使用者的责任。

随着时间的慢慢推移，xxx 机械厂开始使用初始的资金，进行车间的建设。由于该厂的初步目标是进行摩托车的生产，当然生产出的摩托车也是有等级的，有高级的中级的和一般的，于是需要的设备也不同。这样就建出了高中低档次都能生产的车间，以便进行生产。其中建车间过程图 3 所示。

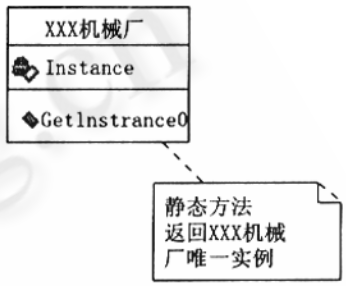

图 2

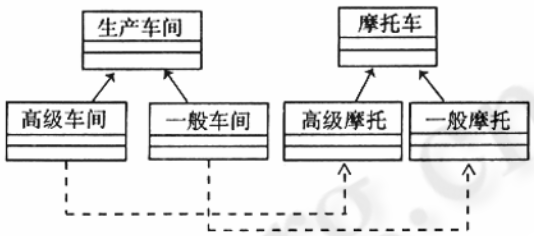

图 3

现在结合设计模式里的模式的符合场景，这就是工厂方法模式，它提供了一种封装机制来隔离出“易变对象”的变化，从而保持系统中“其它依赖该对象的对象”不随着需求的改变而改变。就象图中的“高级摩托车”这一对象如果由于技术的更新，“高级摩托车”的概念上会越来越高级。但是我们将它的变化封装到“高级车间”这一对象中，外界仍然使用“高级车间”生产的摩托就行。

接下来，该厂在全体员工和领导的共同努力下，资金迅速膨胀，为了能够更好的发展，扩充厂的规模的呼声越来越强烈，于是厂领导们经过慎重考虑－－－－－－－扩充厂的规模，于是领导们想到了生产小轿车。现在的该厂的生产规模如图 4 所示。

在软件系统中，经常面临着“一系列相互依赖的对象”的创建工作；同时由于需求的变化，往往存在着更多系列对象的创建工作。如何应对这种变化？如何绕

过常规的对象的创建方法(new),提供一种"封装机制"来避免客户程序和这种"多系列具体对象创建工作"的紧耦合,这就是抽象工厂模式所体现的思想。

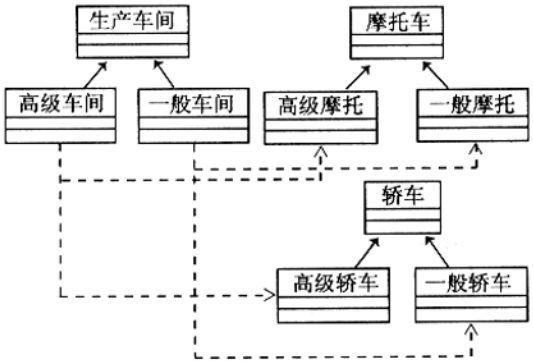

图 4

因为建厂的最终目的是要将产品销售出去,接下来就是销售问题,现在有客户 A 需要订购高级摩托车,和一般的轿车。有客户 B 需要订购一般的摩托车,和一般的轿车。有客户 C 需要订购高级摩托车,和高级的轿车。他们会给厂的销售部门打电话说明自己的情况,销售部门再将这些转交个产品发送中心。这种需求,其实有一个相对稳定的步骤就是订摩托和订轿车。这一过程如图 5 所示。

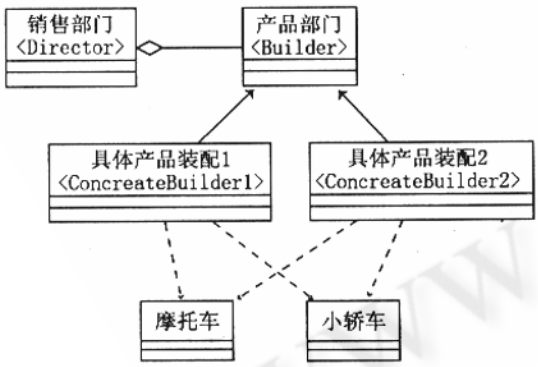

图 5

这就是构造器模式,提供一种"封装机制"来隔离出"复杂对象的各个部分"的变化,从而保持系统中的"稳定构建算法"不随需求改变而改变。

现在该厂生产的产品由于质量好,价格适当,在消费者中产生了很好的口碑,于是有客户现在开始大规模的购买该厂的产品了。

由于产品本身的生产过程就是一个复制的过程,该过程如图 6 所示。

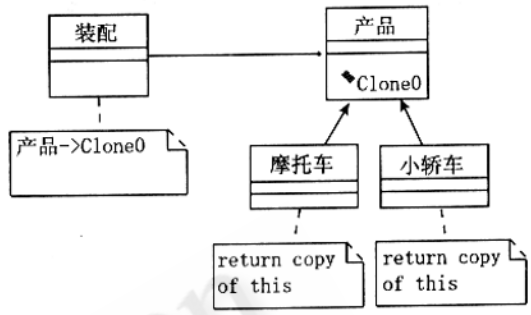

图 6

到此,无论客户有多少数量的需求都能生产自如了。这就是原型模式。用原型实例指定创建对象的种类,并且通过拷贝这些原型创建新的对象。

现在,在见证了该厂的发展壮大以及运营的过程中,创建型设计模式的 5 个模式也随之讨论完毕。这就是我对设计模式关于它所适合的场景的分析观点,同时我们可以用这种思维方式,结合具体场景与各种设计模式体现的思想,来分析和掌握更多的设计模式,以达到对设计模式体现的思想的合理性,不再孤立片面的理解设计模式。

5 结束语

本文通过将创建型设计模式与具体的实例结合,通过提取它们的本质特征,使我们对设计模式的理解不再孤立、片面。当这种方式融入到了你的思想中后,你就会不自觉地使用这种思想去进行你的设计和开发,这才是最重要的。

参考文献

1 Erich Gamma、Rocjard Helm、Ralph Johnson、John Vlissides 设计模式:可复用面向对象软件的基础[M],北京:机械工业出版社,2000.
2 Brandon Goldfedder 模式的乐趣[M],北京:清华大学出版社,2003.
3 Alan Shalloway & James R. Trott 设计模式精解[M],北京:清华大学出版社,2004.
4 K_Eckel 设计模式精解 http://www.Mscenter.edu.cn/blog/K_Eckel