PROGRAMMER TO PROGRAMMER

Professional Design Patterns in VB.NET:
Building Adaptable Applications

VB.NET 设计模式高级编程

——构建强适应性的应用程序

Tom Fischer
John Slater 等著
刘雷 康珍梅 等译

清华大学出版社

VB.NET 设计模式高级编程

—— 构建强适应性的应用程序

Tom Fischer
John Slater
刘雷 康珍梅 等译

等著

清华大学出版社
北 京

北京市版权局著作权合同登记号：01-2002-3187

内 容 简 介

在当今的面向对象编程中，软件编程人员更加注重代码的重用性和可维护性。设计模式使人们可以更加简单、方便地重用成功的设计和体系结构。本书不仅向读者介绍了设计模式是什么、如何实现设计模式，更通过一些精选的实例帮助读者深刻理解设计模式的真正含义，其内容包括设计模式的含义，设计模式在数据层、中间层和表示层的应用，使用.NET Remoting 技术建立设计模式，以及有关设计模式的一些相关主题。

本书用例经典，高效实用，非常适合于面向对象开发人员和设计人员阅读，对于项目管理人员和系统架构师也颇具参考价值。

Tom Fischer　John Slater et al : Professional Design Patterns in VB.NET:
Building Adaptable Applications

EISBN：1-86100-698-5

图书在版编目(CIP)数据

VB.NET 设计模式高级编程——构建强适应性的应用程序/(美)费舍尔等著；刘雷等译.
—北京：清华大学出版社，2003
书名原文：Professional Design Patterns in VB.NET: Building Adaptable Applications
ISBN 7-302-06574-8

Ⅰ.V… Ⅱ.①费… ②刘… Ⅲ.BASIC 语言—程序设计 Ⅳ.TP312

中国版本图书馆 CIP 数据核字(2003)第 029189 号

出 版 者：清华大学出版社(北京清华大学学研大厦，邮编 100084)
http：www.tup.com.cn
责任编辑：陈宗斌
封面设计：康博
版式设计：康博
印 刷 者：北京通州大中印刷厂
发 行 者：新华书店总店北京发行所
开　　本：787×1092 1/16　印张：23.75　字数：492 千字
版　　次：2003 年 6 月第 1 版　2003 年 6 月第 1 次印刷
书　　号：ISBN 7-302-06574-8/TP·4926
印　　数：0001—4000
定　　价：50.00 元

出版者的话

近年来，国内计算机类图书出版业得到了空前的发展，面向初级用户的应用类软件图书铺天盖地，但是真正有深度和内涵的高端图书不多。已经掌握计算机和网络基础知识的人们，尤其是 IT 专业人士迫切需要“阳春白雪”。IT 图书市场呼唤精品！

为了满足这种市场需求，清华大学出版社从世界出版业知名品牌 Wrox 出版公司引进了受到无数 IT 专业人士青睐，被奉为 IT 出版界经典之作的 Professional 系列丛书。这套讲述最新编程技术与开发环境的高级编程丛书，从头到尾都贯穿了 Wrox 出版公司“由程序员为程序员而著(Programmer to Programmer)”的出版理念，每一本书无不是出自软件大师之手。实际上，Wrox 公司的图书作者都是世界顶级 IT 公司(如 Microsoft，IBM，Oracle 以及 HP 等)的资深程序员，他们的作品既深入研究编程机理，传授最新编程技术，又站在程序员的角度，指导程序员拓展编程思路，学习实用开发技巧，从而风靡世界各地，被 IT 专业人士和程序员视为职业生涯中的必读之作。

为了保证该系列丛书的质量，清华大学出版社迅速组织了一批位于 IT 开发领域前沿的专家学者进行翻译，经过编辑人员的进一步加工整理后，现陆续奉献给广大读者。

读者可以从 www.wrox.com 网站下载所需的源代码并获得相关的技术支持。同时，也欢迎广大读者参与 p2p.wrox.com 网站上的在线讨论，与世界各地的编程人员交流读书感受和编程体验。

前　言

大家可能会很容易把一个设计模式当作一个解决方案的模板。一旦提出一个问题，首要的工作就是要标识该问题所定义的特性。然后，检查我们的设计模式库，以确定是否具有一个一般的解决方案来解决已经描绘其特征的问题。如果事实如此，那么就应该应用该解决方案模板来处理相关的问题。

设计模式本身描述了问题的特性，以及解决方案的特性。解决方案的模板已经被多次使用并且被证明运行良好，也就是说一旦我们正确地标识了将要使用的设计模式，就可以应用它了，而无需再进行调查研究和概念的证明以及测试。

这一过程可以普遍适用于生活中的各个方面——例如：建筑学、医学、家具回收等等。虽然并不是所有的规律都使用“设计模式”这个术语，但是无论如何，这个过程是基本相同的。本书介绍了 OOP(object-oriented programming，面向对象程序设计)中的设计模式。

因此，OOP 中的设计模式是一个解决方案的模板——它描述了问题的特性以及解决方案本身的特性，但是它需要您(开发人员)来实现解决方案中的细节问题。设计模式不是关于编程“技巧”的，但是它非常容易渗透到问题的核心部分，而且它允许把事物分解为它们的组成部分，以便开发人员分别处理。设计模式可以帮助您全面地了解解决方案的方方面面，而无需指定某一个解决方案。

本书利用了 Gang of Four (或 GoF)(Erich Gamma、Richard Helm、 Ralph Johnson 和 John Vlissides)的著作，他们具有创新性的一本著作是 Design Patterns：Elements of Reussble Object-Oriented Software(Addison-Wesley, ISBN 0-201-63361-2)，此书描述了 OOP 设计模式的基础，并且详细地记载了 23 种设计模式。

设计模式依赖于产品语言中的面向对象的功能。和以前典型的 Visual Basic 语言不同，VB.NET 是一种真正的面向对象的语言，并且要求一种完全不同的编程习惯。在本书中，我们将重点介绍 VB.NET 中的设计模式的使用，以及如何在上下文中使用 GoF 模式(和其他的模式)。

本书读者对象

本书主要适用于那些希望了解设计模式(使用 VB.NET 编写)的概念和功能性示例的开发人员和设计人员。支持设计模式的思想是不依赖于编程语言的；因此，本书中所

描述的思想可以应用于任何一种真正的面向对象的语言。不过，本书中的实例都是使用 VB.NET 编写的。

本书假定您对设计模式没有任何的认识，且不要求通晓 Visual Basic .NET。本书还利用了一部分篇幅介绍关于 OOP 和 UML(统一建模语言)的知识，因为大多数的设计模式都以面向对象的原理为基础，并且都是利用 UML 图来描述的。

本书的附录 A 介绍的就是 UML 入门的知识，这正是为那些对 OOP 和 UML 了解不多的读者准备的。对于这部分读者来说，本书使用 UML 图描述了功能强大和通俗易懂的 OOP 应用程序，并且它也可以说是一本学习有关 OOP 和 UML“基础原理”的好伴侣。本书并没有在各个章节中介绍 OOP 或 UML 的基础知识，我们希望读者已经有一些有关的经验，或者非常愿意自己去补充有关的知识。

本书适用的读者还有：

- 通晓 VB.NET，并且希望使用像设计模式之类的面向对象的技术从编程语言和应用程序的体系结构中获取更多知识的读者。
- 阅读过 GoF 的 Design Patterns 一书，并且希望学习一些利用 VB.NET 来演示的 GoF 模式和其他模式的读者。
- 使用 Visual Basic 来实现设计模式，并且希望学习真正的面向对象的 VB.NET 的技术是如何让整个过程变得更加简单的读者。

本书偏重于实用性，对于 VB.NET 开发人员来说，它是学习偏重于理论性的 GoF 的 Design patterns 一书的最佳伴侣。

本书主要内容

第 1 章介绍了设计模式的定义，并且讨论了对 VB 和 VB.NET 编程人员有意义的设计模式。其中还包括了许多有关 GoF 设计模式的运行演示程序，它们都带有注释和对比。

第 2～4 章介绍了三个案例分析，分别对应标准的三层应用程序中的每一层。第 2 章介绍了数据层，第 3 章介绍了中间(业务逻辑)层，第 4 章介绍了表示层。这些案例分析为我们提供了大量的训练。例如：

- 讨论识别(假定问题的)设计模式的特性的过程，然后设计一个基于该模式的解决方案的体系结构。
- 测试 GoF 模式和其他模式的实现(在隔离体和联合体中)。
- 介绍如何利用代码来实现这些设计模式。
- 重点介绍将设计模式应用于特定问题的优点。

虽然这些案例分析是相似的，但是它们每一个都是一个独立的个体，都是为每一层

上下文中的设计模式的关联和实现而设计的。

为了配合三个主要的案例分析，第 5 章介绍了.NET 中的远程登录测试——一种将.NET 应用程序的分布式的层次绑定到一起的一种候选技术。第 5 章没有直接介绍.NET 的远程登录，而是从设计模式的角度介绍了.NET 的远程登录的主题。其中，首先介绍了如何依据.NET 的远程登录原理建立设计模式，然后介绍了.NET 的远程登录是如何支持利用设计模式实现远程登录的开发人员的。

本书的第 6 章介绍了与设计模式的分析相关的一些领域和主题。这一章也包括了一些有关资源、引用和进一步学习的内容。

本书没有涉及的内容

本书并没有包括设计模式的所有内容。确切地说，它的主题是实用的设计模式应用程序。在本书中，我们将会讨论大多数的(尽管并不全是)GoF 设计模式。我们还将利用一部分其他非 GoF 设计模式(例如：Store 、 Forward、Model/View/Controller 和 Asynchronous 编程等设计模式)。

本书的正文中没有涉及 VB.NET 语言、OOP 和 UML 的知识；我们只是假定读者对这些主题已经有所了解，或者在必要时他们愿意自己去参考其他的资料。但是，正如我们所提到的，本书后面的附录 A 是一个 UML 入门——它是专门为初学者提供 UML 的快速指导和参考入门而准备的。

使用本书所需的条件

本书中的示例是用 VB.NET 编写而成的。如果您希望建立自己的示例，那么您将需要以下列出的部分或者全部：

- 合适的操作系统。Windows 2000 Professional、Server、Advanced Server 版 (编写本书时，最新的服务包是 SP2)、或者 Windows XP 专业版。
- .NET Framework SDK。
- 在第 1～4 章中，我们使用 Visual Studio .NET IDE 构建和编译一些应用程序。实际上，VB.NET 中装载了.NET Framework SDK 和 Visual Studio .NET 两种运行环境；也可以使用.NET Framework SDK 中的命令行工具构建和编译这些应用程序，而不使用 VS.NET。本书中我们只在第 5 章使用编译器。
- 第 2 章和第 3 章利用了 SQL Server 2000 的数据库服务。如果无法成功地安装 SQL Server，也可以使用 SQL Server Desktop Engine(也称为 MSDE)来运行有关示例。

注意：

.NET Framework SDK 装载了 MSDE，VS.NET 也具有相同的功能。为了安装 MSDE，首先要执行 InstMSDE.exe 文件(该文件可以在\Program Files\Microsoft.NET\FrameworkSDK\Samples\Setup\MSDE 文件夹中找到)，MSDE 并不提供与 SQL Server 相同的管理工具，但是它提供一些非常有用的命令行工具，在\Program Files\Microsoft.NET\FrameworkSDK\Samples\Setup\html\ConfigDetails.htm 文件中有关于这些工具的详细介绍。除此之外要注意的是，VS.NET 专门为 SQL Server 数据库提供了一种可以说是相当优秀的 UI。

- 第 3 章利用了一种“传统”的类库应用程序。我们为这种应用程序提供源代码以及经过编译的 DLL(动态链接库)。您无需自己构建 DLL。但是，如果希望自己编译 DLL，那么将需要利用 Visual Basic 6.0 来完成这一工作。
- 第 5 章利用了 MSMQ，而第 2 章和第 4 章利用了 IIS 5.0。这些 Windows 组件装载到了 Windows 2000 所有的版本，以及 Windows XP 专业版上。有些版本并没有把 MSMQ 和 IIS 作为默认的安装操作系统的一部分——您可以从 Control Panel | Add/Remove Programs 对话框中选择用手动来安装它们。

目　　录

第1章　设计模式入门

自从开发人员首次开始应用面向对象的技术以来，已经创造了许多奇迹。随着对实现 OOD(面向对象设计)技术越来越全面地理解和认识，开发团体也越来越多地发现某些类型的问题一再地出现——那么自然会想到，可以利用类似的解决方案来解决类似的问题。

在全世界范围内，经验丰富的软件工程师在项目设计的过程中，都可以独立地识别特定结构的(和与对象相关的)需求。除此之外，每识别一个特定的需求，他们都可以应用一个特定的对象模式来完成所需的解决方案。

一段时间之后，OO 编程已经很自然地生成了一系列通用的可识别的问题及其解决方案。因此，当开发人员开始设计一个应用程序，并发现需要解决的特定问题和结构时，他们通常会发现那种类型的问题或结构已经在别处被标识出来了，也就是说一个合适的、测试良好的解决方案已经存在了。

目前这种现象已经被授予了特有的名称，它被称为——设计模式。在这一章中，我们将从三个不同的角度分别来处理该主题，介绍设计模式：

- 首先，介绍设计模式的概念，以及它对于 Visual Basic 开发人员的重大意义。
- 其次，将以一个虚构的薪水账册应用程序为基础，对两种不同的设计进行比较，并且进一步介绍如何使用设计模式来改进一个 VB.NET 的应用程序。
- 最后，介绍一些基本的设计模式——特别是一些非常通用的设计模式，以及与本书的知识密切相关的设计模式。

1.1　设计模式的概念

从本质上说，设计模式是两件事物的组合：对一个问题的描述及其解决方案的描述。设计模式从它定义的特性来描述一个问题，并且从该问题所包括的元素这一角度来描述它的解决方案，同时设计模式还描述了两者是如何共同运作，以及它们所实现的功能。

设计模式的概念可以应用于生活的各个领域。例如：在都市规划和建筑学中，设计模式常常用来描述城市设计和建筑设计中的普通问题的特性，以及一些为解决某些问题而准备的标准的解决方案的模式。对城市的设计者和建筑师来说，工作的艰巨性在于他们需要从真实世界的问题中辨别出它们属于哪一类的标准模式，并找出一种标准

设计模式的理论与之相匹配，然后应用规定的解决方案来解决实际问题。

注意：

解决方案并没有描述一个实现的细节问题。取而代之的是，它就像一个模板：用通用的语言描述了所有必需的元素和解决方案的特性，并且形成了许多应用程序得以实现的主要部分。

在这本书中,我们准备利用设计模式的概念来进行面向对象的编程(OOP)。在各种OOP环境中，设计模式的原理都是相似的。每一种设计模式中都包括：

- 一个关于通用编程问题的描述，使用通用术语来声明。
- 一个关于问题的测试良好的解决方案的描述。这个解决方案存在于模板中，它描述了每一个预期的类、角色、该类具有的职责以及类之间的相关性。

对于从事面向对象分析和设计的编程人员来说，真正的挑战在于能够识别存在于工程之中的每一个通用问题的模式，将它与已经建立的设计模式相匹配，并且在设计中将有关的解决方案的模板应用于他们的项目。

1.1.1 模式分类

在这一节中，我们将会告诉大家设计模式从何而来。是否应该存在一个确定的设计模式的列表，每一个面向对象的编程人员所采用的模式都不能超出列表的范围？

创建一个列表来包含所有的设计模式几乎是不可能实现的事情，但是随着时间的推移，发现一些全新的和基础的设计模式已经变得越来越困难了。虽然这是一个非常有趣的挑战，但是提出设计模式概念的真正目的是帮助我们创建一些应用程序，在最近的几年中，人们已经识别了大多数的最通用的设计模式，并且将它们存入了档案。

在这一领域中，存在一个特别重要和著名的著作，我们将在后面介绍这个著作：

1. Gang of Four

在九十年代早期，这种新的设计习惯对软件制造业产生了很深远的影响。技术资料、讨论以及一些团体的不断涌现，为编程人员进行最新的思想和模式的交流提供了巨大的舞台。但是刚开始的时候，由于面对的是全新的科学领域，而且还没有对相关的信息进行很好的分类归纳，因此使用起来也相当不方便。1994年，有四个人(Erich Gamma, Richard Helm, Ralph Johnson 和 John Vlissides)解决了这个问题。他们创作了一部可以被称为是设计模式领域中的“圣经”的著作——《设计模式：可重用的面向对象的软件元素》(Addison-Wesley, ISBN 0-201-63361-2)。这本书的成功为他们赢得了一个亲切的称号：四人组(或者 GoF)——在这本书中所描述的模式也常被称为 GoF 模式。

GoF 的《设计模式手册》中详细地记载了二十三种设计模式。它们不仅能够激发作

者的兴趣和灵感，此外，这些模式描述了一种设计思想的集合，其中的思想都是经常出现在许多强大的面向对象的应用程序之中的。例如：他们著作中的 Adapter 设计模式可以完成的工作是：花费最小的精力将一个现有的类接口转换为另一个接口。任何一个理解 Adapter 设计模式的软件工程师都可以直接套用既定的模式来完成相关的任务，而不必重复创建和调试的过程。

每一个 GoF 设计模式都具有一个固定的形式，它由两部分组成：

- 一个用通用语言描述问题的语句，其中包括设计模式的用途、行为方式及其应用性。
- 一个对解决方案的描述语句，其中包括设计模式的结构、参与合作成员的列表、应用该模式的结果、实现的注意事项、一些样本代码、已知的用途以及相关模式的列表。

2. 其他的记载和资源

以上提及的二十三种 GoF 设计模式并不是现在仅有的面向对象的设计模式。但是，本书并没有介绍其他的设计模式。在 Web 站点上有许多其他的引用，如果需要获取更多有用的信息，可以登录以下的 Web 站点(以下并没有包括所有的相关站点)：

- http://www.cetus-links.org/oo_patterns.html—— 一个出色的链接列表。
- http://hillside.net/patterns/—— 模式社区的非官方站点。
- http://c2.com/cgi-bin/wiki—— 模式社区的另一个非官方站点。
- http://www.xmlpatterns.com/——XML 模式(无疑不是 GoF 模式)
- http://www.dofactory.com/patterns/patterns.asp—— 该站点是针对 C#程序员的，但所有的模式原理都可以应用到 VB.NET 中。

1.1.2　设计模式与 Visual Basic

尽管设计模式的适用范围很广泛，但底层目标语言的性能仍然非常重要。在 GoF 的设计模式著作中，他们使用 C++和 Smalltalk 来演示其思想——而这两种语言都同产品语言一样是面向对象的语言。

实际上，这恰恰显示了把设计模式应用到面向对象环境中所得到的一个重要的结论：即产品语言必须是一种真正意义上的面向对象语言。其中的原因在于：面向对象的设计模式的思想是建立在面向对象的理论基础之上的——即具有抽象性、封装性 、多态性和(实现中的)继承性等特性。如果您预期的目标语言不具备这些关键特性(与真正意义上的面向对象的程序开发有关的)，那么可能会导致的结果是：您根本不能使用该语言来实现一个设计模式。

同时，这也为那些希望在 Visual Basic 应用程序中应用设计模式的开发人员提出了

一个新的挑战。问题在于到目前为止，没有一个版本的 Visual Basic(包括 6.0 版本)是真正意义上的面向对象语言，VB 不能够直接支持继承性的实现。在 VB 中使用设计模式的任务是非常麻烦的，任何用 VB6 开发的应用程序都需要一个相当大的工作区，因为这些应用程序依赖于一种非常复杂的继承模式。

VB.NET 的出现

考虑到以上的情况，我们就很容易理解大多数的 Visual Basic 开发人员想要停止使用设计模式的心情了。虽然设计模式的思想为程序设计带来一定的益处，但是在 Visual Basic 应用程序中生硬地应用现有的设计模式所带来的复杂性和风险却让那些益处显得微不足道了。经验丰富的 Visual Basic 开发人员进行了一定的成本/收益分析，得出了一个结论——设计模式不适用于 Visual Basic。

.NET Framework 和 VB.NET 的出现已经明显地改变了以上的分析结果。VB.NET 是一种真正意义上的面向对象的语言，它也因此成为了设计面向对象应用程序的产品语言的最佳选择，而设计这些应用程序都是以设计模式为基础的。

突然之间，Visual Basic 的开发人员获取了一种强有力的面向对象的编程语言。但是，同时他们也发现了想要从 VB6 迁移到 VB.NET 比迁移到 VB 任何的早期版本都要更加困难。虽然他们精通陌生的 API 调用和非正式的技巧，但是这并不能为创建强大的面向对象的应用程序提供更多的帮助。在 VB.NET 中编程需要的是一种观念上的转变，从而更加适应面向对象的特性。

幸运的是，大量针对其他面向对象语言的组织，也在为 VB.NET 编程团体提供一些解决办法。从一些使用设计模式的应用程序(用 C++ 或 Smalltalk 语言编写)完成的工作看来，存在一个巨大的理论体系和最佳的实践案例可供我们学习，这些理论和案例适用于所有 OO 语言的设计模式。

1.1.3 使用设计模式的时机

有的时候，在一个特定的情况下很难判断使用设计模式是否有意义——也就是说，需要判断使用设计模式的应用程序何时能够获取绝对的利益。正如 VB6 的开发人员，他们放弃了对设计模式的使用，因为将结果生硬地套用到他们所使用的语言中非常困难，这就是一个不适用设计模式的情况。

列出一些开发人员不希望使用设计模式的情况很有意义。毕竟，除非能够真正地提供一些显而易见的利益，否则您应该不会使用一种增添程序复杂性的工具。如果在当前的情况下出现以下的问题，那么您可能会发现试图在那个项目中应用设计模式是不适合的：

- 如果现在创建的应用程序无需任何修改，其中的代码已经完全满足了所有的需

求；而且也没有计划升级或增加其中的功能，那么您创建的这个应用程序将会成为第一个，也是最后一个版本。(的确如此，将无需使用设计模式)

- 如果您的应用程序的需求是独一无二的，将无需使用设计模式。迄今为止，还没有任何一个软件工程师能够创建这样一种应用程序。因为程序不能处理所有的常规问题，例如：创建对象和事件通知。(那么，它到底做了些什么？)
- 如果有充裕时间来规范所有新的设计思想，将无需使用设计模式。当然，前提是您对新的思想能够成功实现具有足够的信心，它们无需任何概念验证(proof-of-concept)测试。
- 如果团队中所有的成员在一起工作的时间都已经超过二十年或者更长的时间，将无需使用设计模式。如果要求 Bill 尝试“thing-a-ma-jig”，他会非常清楚地知道应该怎么做，因为他的团队已经习惯于一个通用的设计词典。如果您也具有类似的情况，就无需使用其他的设计模式了。

如果您的情况与以上所描述的任一情况相符，请将本书赠与您的朋友；否则，请继续学习本书。

当您在项目中判断是否需要使用设计模式时，应该注意两个问题。第一：考虑到额外的复杂性和潜在的执行故障之后，是否仍然可以证明存在适用的设计模式。第二：应该权衡应用设计模式的收益以及较低的维护费用是否多于前期的投资。

1.1.4　设计模式对设计的改进

开始学习一些与 GoF 模式相关的描述和示例之前，首先介绍一个虚构的工资单应用程序的设计示例。我们将分别介绍两种不同的设计方案：其中一个方案使用一个设计模式，而另外一个方案不使用设计模式。最终，可以看到使用设计模式的方案如何为应用程序提供一种显著的可扩充性和灵活性设计。

1. 虚拟的 TakeHomePay 应用程序

首先为 WantaBeBig 公司新的“Take Home Pay Calculator”应用程序(此后称为 TakeHomePay)收集最基本的需求。WantaBeBig 公司已经邀请了两家软件公司(N 公司和 P 公司)为应用程序的设计提供各自的建议，并且每一家公司都希望能够赢得该合同。

WantaBeBig 公司已经指明应用程序的目标语言使用 VB.NET，但是并没有指定任何有关应用程序的设计要求。

程序的需求文档内容如下：

- 概括 TakeHomePay 应用程序的设计目的：WantaBeBig 公司雇员在提交时间表后能够立即看到下一次的工资数额
- 描述了计算工资总额应该包括的一些复杂的计算：它们基于不同的国税、地税