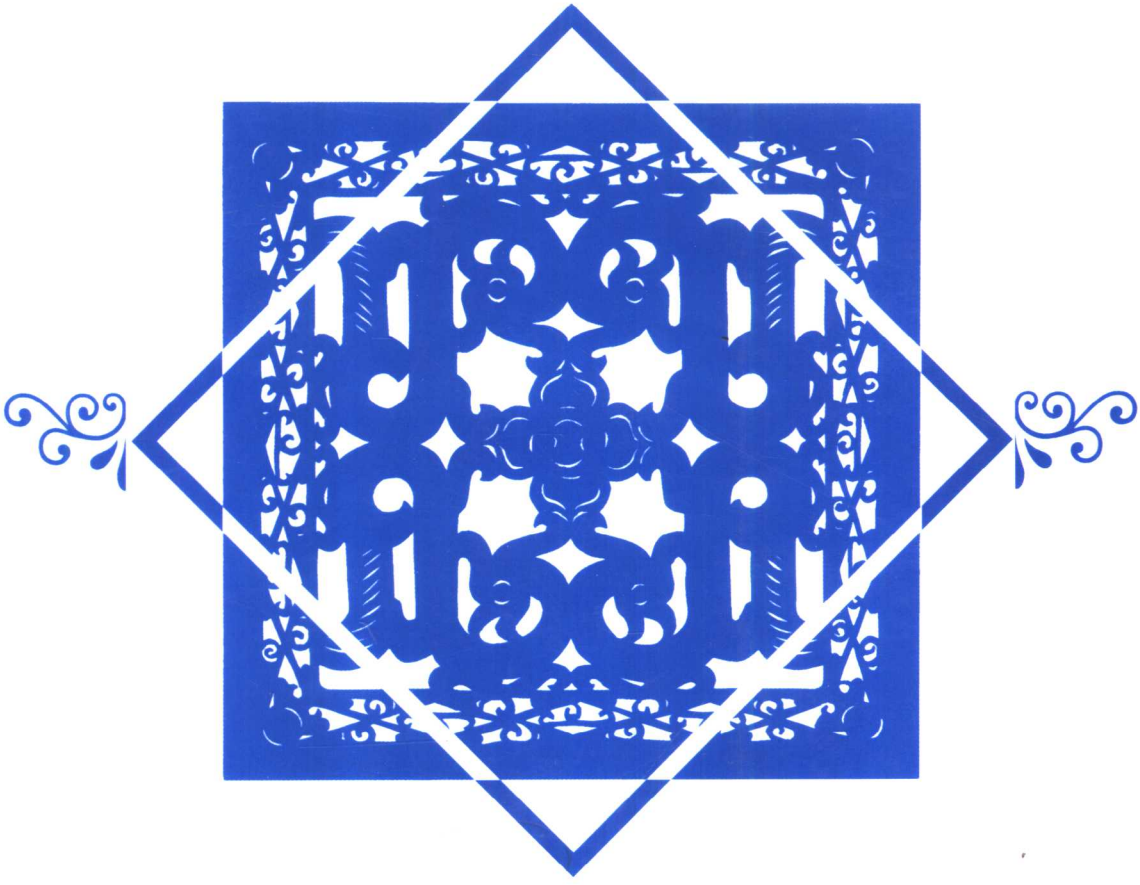
Broadview
WWW.BROADVIEW.COM.CN

博文视点原创精品大系
微软技术专家



Framework的设计与应用
——基于Windows Forms的应用开发实践

黄忠成 著

CD-ROM
電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

Framework 的设计与应用

——基于 Windows Forms 的应用开发实践

黄忠成 著

電子工業出版社
Publishing House of Electronics Industry
北京 • BEIJING

内 容 简 介

本书着眼于三类 Framework 中的 Domain Application Framework 与 Application Framework，先建立一个可套用于多数应用程序的 Application Framework，再以此为基础，建立出趋近实际需求的 Domain Application Framework。本书分成四部分，第一部分讲解.NET Framework、ADO.NET、Windows Forms、Remoting 等基本概念及操作，第二部分讲解如何撰写 Application Framework，第三部分讲解如何撰写 Domain Application Framework，第四部分以 Domain Application Framework 撰写一个小型进销存系统。从概念、设计、强化到实践，一应俱全。

图书在版编目（CIP）数据

Framework 的设计与应用：基于 Windows Forms 的应用开发实践 / 黄忠成著. —北京：电子工业出版社，2006.10

ISBN 7-121-03138-8

Ⅰ.F... Ⅱ.黄... Ⅲ.计算机网络—程序设计 Ⅳ.TP393

中国版本图书馆 CIP 数据核字（2006）第 103712 号

责任编辑：周筠 梁晶 特约策划：刘铁锋

印 刷：北京智力达印刷有限公司

装 订：北京中新伟业印刷有限公司

出版发行：电子工业出版社

北京市海淀区万寿路 173 信箱 邮编 100036

经 销：各地新华书店

开 本：787×1092 1/16 印张：40.75 字数：840 千字

印 次：2006 年 10 月第 1 次印刷

印 数：6 000 册 定价：69.00 元（含光盘 1 张）

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系电话：（010）68279077；邮购电话：（010）88254888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线：（010）88258888。

献给我最爱的小乖，谢谢你的耐心与宽容，
你的笑容与鼓励，是我最大的动力。

黄忠成先生谈 Framework

针对黄忠成先生的新书《Framework 的设计与应用——基于 Windows Forms 的应用开发实践》的出版，博文视点和《程序员》杂志携手对黄忠成先生进行了专访，现将本次专访整理成文，以飨读者。

采访人：博文视点首席策划编辑刘铁锋（Joylite）和《程序员》杂志技术编辑欧阳璟（欧阳）
受访人：黄忠成（Code6421）
采访方式：电子邮件
访谈主题：纯软件应用的书是如何撰写出来的

书籍创作的背景以及具体思路

♠ Joylite:

黄先生，您好。非常高兴看到自《深入剖析 ASP.NET 组件设计》之后，又有一部力作上市。我想对于读过您的作品的读者来说，应该非常熟悉您写作的风格——我个人感觉为“技术狂人”的写作风格。在我的印象中，《深入剖析 ASP.NET 组件设计》一书是想人所不敢想，为人所不能为。庖丁解牛般愣是冲开了微软 ASP.NET 厚厚的“牛皮”，一下子为读者展现了组件设计的本质和实用技巧，让读者能够不费吹灰之力，利用您的源代码进行有效的开发，并且有效地扩展了开发 ASP.NET 程序的思路。不知道您的这本书，是否也延续了这种风格呢？

♣ Code6421:

目的不同，风格上自然会有所差异，ASP.NET 组件设计面对的是一个已经存在的 Framework，目的是拆开这个 Framework 的包装纸，将内部的零件一一呈现在读者面前，忠实地告诉读者，各个零件的功能及设计初衷，了解之后方能活用，活用后方能幻化，是技术研究的不二法门。框架设计则是一个完全相反的角度，因为不存在，所以必须创造，由零件开始，一个个地建立，将零件建立前的动机、建立间的细节、建立后与主体的结合，一一呈现在读者眼前，若要用一句话来点出两本书的不同，那就是“《深入剖析 ASP.NET 组件设计》是解构，此书是建构”。

♠ *Joylite:*

对于您这个题目来说，读者可能会很诧异。因为在读者印象中，框架是个非常大的概念。比如.NET Framework。因此，您是如何决定写作这本书的？

♣ Code6421:

对于一个长期处于商用数据库应用程序世界中的工程师来说，框架是一直存在于他的生活中的，它可能是一群未经过整合的函数库，也可能是一群未经过整合的组件，没有经过严谨的整合及说明，它们只是资深工程师的私家法宝，代代相传给新进工程师，但一旦通过架构的规划，并加以整合之后，将变成一个可以撑起一家公司的框架。框架不见得是非常大的，但它一定是非常有效率的，在提供快速的开发方式外，也可以快速地引导新进人员进入生产线，发挥生产力，更可以控制产品的质量。

举例子来说，Delphi 工程师常撰写 Base Form 及组件，利用继承及组件组装，来快速地生产应用程序，将这些整合起来后，就是一个框架，即使它不像 VCL 和.NET Framework 那么庞大，但它是一个框架这点是毋庸置疑的。从 Delphi 时代以来，我一直都扮演着建立这类框架的角色，到今日的.NET Framework，我仍然站在这个位置上，写这类书籍一直是我所想做的事，算是了一个心愿吧！

♠ Joylite:

对于绝大部分的程序员来说，写代码是件很爽的事情，但是写文档就比较痛苦。甚至因为对细节的理解程度不太一致，也很难通过一种有效的方式给新手把代码讲解清楚，尤其是对于您这本书的内容来说，介绍一个框架，不仅需要介绍如何设计框架，还需要解释框架里面的代码内容。那么，从整本书的撰写来说，您是通过何种方式来组织内容，针对怎样的人群来讲解本书的核心内容的呢？

♣ Code6421:

呵，大部分的程序员也包括了我，我很讨厌写文档，这点可以从我所写的程序代码惜字如金的批注中看出，面对一个庞大的 Framework，如何将意念传达给读者，是件很难的工作，我秉持一个原则，先解释该框架的背景，也就是诞生的动机，然后从设计面切入，告诉读者该如何设计这个框架，又可能会遇到哪些困难，又该如何解决，最后由实现面切出，将程序代码呈现在读者面前，不可否认，对于新手来说，大量的程序代码是吓退他们最好的武器，这也是为何本书会先由设计面切入的原因，从动机看结果，比从结果中找寻动机来得容易。针对初学者，书中的前五章从基础逐步讲解，给予读者阅读随后章节所需的基本功，第 6 章从设计面角度切入，一一点出开发应用程序会遭遇到的困难，再一一找出解决方法，并融入 Framework 之中。

书籍创作中最大的问题：对框架的选择

♠ Joylite:

正如您前言中所提及，在本书的写作过程中，对于框架的选择和设计，您也有着很多的方案。并且在写作的过程中，也是几易其稿，那么，您是如何做出决定的呢？或者说，您是如何来评估这个设计方案的好坏的呢？对于读者如何设计其他的框架，您有没有什么好的意见和建议，尤其是在设计方案评价上有没有具体的技术指标？

♣ Code6421:

决定一个框架是否有用，只要用该框架来写一个应用程序即可，决定一个框架是否足以应付日后的变化，也只要假想客户会提出何种需求即可。这两点对我来说都不难，我所任职的公司正以.NET Framework 2.0 开发 Windows Forms 应用程序，这个框架中的许多手法，都是为了解决他们的问题，而他们的客户所提出的问题，则间接证明了这个框架有着一定的扩展性，事实上，这个框架中的许多设计是为了解决客户的需求，而做出变动的。

设计一个 Domain Application Framework，最大的难点是，你必须找出产品线中通用的部分，架构、使用接口和常见需求，然后将其萃取出来，画出设计图，再一一实践之。以本书中范例为例，一个进销存系统，我必须找出一致的操作接口、一致的错误处理流程和通用的功能，如用户管理、Plug-In 系统等等，然后看看我能做什么，例如将操作接口预先写成一个操作类别，或是写成 Base Form，将错误处理流程包成一个简单易用的类别，将通用功能写成组件或控件。越懒惰的人，相反越能写出易用的 Framework，当然，前提是他得勤劳到写一个 Framework。

对 Enterprise Library 的选择以及比较

♠ Joylite:

我想您对 Enterprise Library 不会陌生，在国内也有很多的朋友对 Enterprise Library 进行了研究。在您的书籍中，框架中涉及到的内容，基本上 Enterprise Library 中都有涉及，您如何对比 Enterprise Library 和您创作的框架呢？同时，您觉得在快速开发中，应该如何选择和利用现有的框架呢？（学习有学习曲线，但学会之后是可以提高效率的，但给维护带来了问题。）

♣ Code6421:

理论与实践间的差异，Enterprise Library 大部分是多个理论的结果，Orphean WinForm

Framework 大部分是一个实践的结果，OWF 着眼于面对问题时的解决方案，以高度的假设性及简洁性来换取开发效率，在理论及完善度上，都不见得是最好的做法。Enterprise Library 则着眼于理论的实现，以高度的扩展性及理论性为基础，若详细审视 Enterprise Library，你会发现它是如此的美丽且完善，许多情况都已经考虑周全。只是理论有时会变成一种负担，令整个 Framework 变得庞大、难以学习，接着使开发效率降低，实践有时则会变成一种限制，让产品缺乏扩展性，两者各有优缺点。学会之后可以提高效率，这是一个理想性的问题，因为前提是需要花上一段时间先学会，面对人员的流动，能学会常常变成最难达到的目标。在建筑理想软件之前，得先考虑软件是否真的能做出来。

对快速数据库开发的建议

♠ Joylite:

使用.NET 开发的开发者，或多或少会和数据库打交道，并且更多的是需要快速地开发数据库应用程序。您是从 Delphi 开发出生的，也经历了种种技术的变迁，您是否可以对比一下近年来的数据库相关技术（如 ODBC、ADO、DAO 和 ADO.NET），您认为 ADO.NET 最大的优势在哪里？怎么才能最有效地发挥 ADO.NET 的功效？您怎么评价 ADO.NET 中提出的离线的 dataset 功能？您对使用 dataset 的建议是什么呢？

♣ Code6421:

ODBC 到 ADO，是程序设计模式的变化，从 OOP 到 CBD 的转变，而 ADO 到 ADO.NET 则是环境所引发的变化，为了让一个数据库能服务更多的用户，ADO.NET 采用了离线式的数据库，这并非是一个新想法，早在 ODBC、ADO、BDE 时代就已萌芽，只是 ADO.NET 将其披露在我们面前而已。ADO.NET 最大的优势在于其简单且强大的扩展性，设计一个 ADO.NET Data Provider 并不是件很难的事，尤其在你已经拥有 Database Native Driver 时，例如我们可以很轻易地运用 P/Invoke 通过 BDE 写出一个 for Paradox 数据库的 ADO.NET Data Provider，这在 ODBC、ADO 和 BDE 时代是很难想象的。使用一个离线式的数据库，你必须在 Local 与 Server 间做选择，这个工作是交给 Local 做来得快，还是交给 Server 做来得快。以查询来说，虽然在某些架构下，Local 端已经有需要的数据，只要过滤后即可显现，但 Local 端的过滤是速度较慢的动作，此时交给 Server 来做就会快许多，另外如何处理拥有大量数据的数据表也是个问题，对用户来说，一次传回比起只传主键，当需显示时再传回整条数据来得慢，但对 Server 来说，后者的负担远比前者来得大，折中的方式应该是批次传回，也就是以每页固定条数为单位传回，这样一来，用户不会等待过久，Server 也不会负担过重。使用离线式数据库，设计师必须视项目的情况来调整 Local 与 Server 的负担平衡，就像是操作天平般。

♠ Joylite:

从您书里描述的框架中，我看到了一个每位程序员都曾经试图想做的、能够有效地提高自己工作效率的方案。记得以前和您聊及，您甚至直接撰写了一个基于.NET 的 O/R 框架。那么，您如何对比使用 O/R Mapping 以及传统的 dataset 技术呢？

♣ Code6421:

O/R Mapping 技术是一种理论，一种纯化 OO 概念后的数据库新概念，将对象对应成一个数据行，将一个 Collection 对应成一个数据表，或是一个结果集，运用 O/R Mapping 技术来撰写程序，对于 OOP 设计师来说，是再直接不过的工作，但在实践中，设计师得面对许多问题。首先，你得确定所选的 O/R Mapping 实现体够稳定，且有效率，再者，建立 Mapping Class 的机制必须够直观，日后的变动也必须够简单，若要快速开发应用程序，那么与 Data Bindings 系统的整合也是不可缺的要素。

传统的 dataset 技术虽然看起来有点过时，但是由于这个技术已经演化了多年，那个功能该如何实践，又该处理那些问题，都已经有了较制式的解法，因此稳定性及效率都具有一定的质量，加上它是.NET Framework 内建的数据库处理机制，你不需花费额外的费用就能使用它，比起质量不可预知的 Open Source 及昂贵的 3rd Party 产品，自然更适用于立即性的产品开发。未来的.NET Framework 3.0 会拥有一个 O/R Mapping 系统，届时 dataset 技术将正式走入历史。

♠ Joylite:

在您的框架中，您提供了一个混合的 N-tier 的框架，可以提供 Web Services 以及.NET Remoting 的支持，并且通过一个简单的程序提供了技术演示。那么，您如何看待 Smart Client 这样的技术？您认为这种技术会成为.NET 开发的主流吗？

♣ Code6421:

与其从技术角度来看 Smart Client 是否会成为主流，倒不如从人的角度来看，如你所知，我是 Delphi 的爱好者，事实上，Smart Client 能做的事，Delphi 早就能做到，那么为何要使用新的技术，甚至是新的语言呢？在台湾，将一个项目以 Smart Client 开发，而不以 Delphi 开发的理由很简单，就是资源，找一个.NET 设计师比找一个 Delphi 设计师简单，或许你会好奇，Delphi 不是比较成熟吗？是的，不过教这个语言的学校并不多，甚至少到可以用 10 根手指数出来，那这代表什么？这代表着，找一个 Delphi 设计师不容易，未来你甚至找不到人来维护既有的程序，这些风险促进了原来使用 Delphi 的公司往.NET 走，而适合他们的技术就是 Smart Client。

♠ Joylite:

.NET 2.0 方兴未艾，.NET 3.0 又即将推出，微软推出新技术的速度实在让我等目不暇接。那

么您如何看待.NET 3.0 中的新功能呢？对于技术跟进者又有什么好的建议呢？

♣ Code6421:

.NET 3.0 中披露了许多的新技术，每一个都会改变未来我们写程序的方式，WPF 会彻底改变 Windows 应用程序的写作方式，LINQ 会改变数据库访问的方式，WWF 会改变应用程序组成的方式，WCF 会改变目前 N-Tier 的写法，面对这些会对我们造成极大冲击的新技术，我持谨慎及保留的态度，以目前的状态而言，我将研究重点放在 LINQ 及 WCF 上，因为我认为这两者较有可能优先采用于未来将开发的程序中，而 WPF、WWF 则是排在这两者告一段落后才会开始研究，原因是 WPF 的变动太大，且目前所看到的开发及执行效率仍然欠佳，WWF 则是一个架构问题，要采用此技术，也代表着架构必须作大幅度的改变，否则就只是将新技术用于旧架构上而已。

对.NET 技术学习的建议

♥ 欧阳:

Joylite 提到的一些技术问题很不错，是中高级.NET 程序员比较关心的话题，比如框架、组件、逻辑分层、对象关系映射和程序库等，大多数读者以及所有的.NET 开发人员可能也非常关心.NET 技术的学习。

请您谈谈您在学习.NET 技术的时候感觉有本质飞跃的那个时期或者某个经历，是什么让您一下子对.NET 有所顿悟，或者是积累到了什么程度才能对.NET 有本质的了解？

♣ Code6421:

.NET Framework 是一个相当庞大的 Framework，任何个人都很难窥其全貌，我也不例外。

第一次接触.NET Framework 时，我就像是面对一个原始森林，到处都是难见其顶的大树，茫然不知所从。当时我做了一个抉择，既然.NET Framework 是由众多组件所组合而成，那就从研究这些组件的设计及实现面下手吧。对我而言，这个方向是正确的，从这里，我学到了 GoF 所提出的设计模式的实务板，充分理解这些设计模式的成功之处，及.NET Framework 为了实现某个目的所做出的变形设计模式，这些信息，成为我在深耕 ASP.NET 内部运作模式的基础功法，不至于迷失于纠葛成结的 ASP.NET 核心中。现在，这些知识也同样成为我研究 Windows Forms 的后援，我确信，它们日后也会是我研究 WPF、WCF 和 WWF 时的最佳后援。

♥ 欧阳:

很多人觉得.NET 是一个可以快速入门的技术领域，一个程序员花 3 个月的时间就可以非常有

效、快速地编写 ASP.NET 代码，请您简单评价一下这种说法。

♣ Code6421:

看起来越简单的事物，幕后往往隐藏着一大片难以走出的迷宫森林，我常常把 Java 与.NET 拿来作比较，Java 将程序员当成训练有素的人，从不隐藏任何会让程序员感到困惑及麻烦的事，相反的，.NET 则把程序员当成是一个稍加受过程序逻辑训练的人，极力简化其所必须付出的时间成本，协助他们快速完成所需完成的工作。但随着时间的流逝，这些速成的.NET 的程序员会开始感到无力，这在程序交付给客户时更为明显，对程序所遭遇的 Bug 和客户的额外要求，这些人无力招架，最后还是得由资深的主导人来解决。反观 Java 就不同，由于先前的付出，这些人已经拥有处理问题的能力，能走到将程序交付给客户这一步，通常也意味着项目接近成功。我时常跟同事提及，我们是程序员，方便的工具是可以节省时间，但这些省下来的时间，应该用来深耕幕后的知识，这一点决定着你可以在这一行业中生存多久。

♥ 欧阳:

请您给.NET 程序员一些建议。

♣ Code6421:

对于.NET 程序员来说，目前可说是最混沌的时期，.NET 2.0 方尘埃落定，.NET 3.0 又急袭而来，面对如此快速的变动，相信许多人都有种无力感。该如何在这么短的时间内，学会应用这些新技术来开发应用软件，也一定是许多人的共同疑惑。在我自身的规划中，其实并不急于使用.NET 3.0 的技术来开发软件，而是将研究重点放在这些新技术背后的观念及理论，Atlas 为何会被实现出来，又能够解决何种问题，带来何种效应。WPF 有何迷人之处，可以取代 Windows Forms，其设计基础为何等等。了解这些，或许不能让我在短时间内使用它们来开发软件，但我确信，当我开始使用它们时，这些知识会给我强大的后援，让开发工作进行得更顺利。

致谢

一本书的完成，是需要许多人的帮助与努力的，没有这些人，这本书不可能呈现在读者眼前，于此要特别感谢博文视点的周筠、梁晶、铁锋，没有你们的帮忙，这本书不可能完成，也要感谢所有的排版人员，因为你们的努力，使这本书更具质量。最后要感谢鼓励我踏入写书一职的李维老师、匡正，没有你们，我不会有勇气出第一本书，当然也不会有第二本书。

导读

本书的主轴

这是一本蛮特别的书，书中除了教导读者如何使用 Windows Forms 来撰写数据库应用程序外，还引导读者进入快速生产应用程序的世界，在这个世界中，Framework 扮演着举足轻重的角色，它提供了一个数据库应用程序所需的基本功能，让设计师可以快速建构应用程序，不需再重新撰写这些常用的功能。除此之外，Framework 也提供了一致的 UI 接口、一致的操作流程及一致的错误处理流程，让设计师可以将研发重点放在实现客户的需求上，而非这些既琐碎且耗时的事情上。书中同时也将此 Framework 的设计概念展现在读者眼前，让读者了解如何设计与开发一个 Framework。为了让读者了解，这个 Framework 是由实务经验所萃取出来的，本书最末以此 Framework 开发了一个小型的进销存系统，以此证明此 Framework 的能力及优点。

读者所需具备的基本知识

要阅读这本书，读者至少必须熟悉 C#程序语言。

专有名词的使用

对于计算机技术书籍的作者而言，使用原文术语还是使用中文术语，一向是难以抉择的，大量使用原文术语会造成读者阅读上的困扰，但使用一些较少见的中文术语却又会让读者摸不着头脑，弄不清楚意思到底为何，造成阅读上的障碍。麻烦的是，少见或是常见是很主观的，作者所认知的常见术语对读者来说却不见得如此。在本书中，笔者尽量在该术语第一次出现时，同时列出原文及中文，之后则一律使用中文术语。

本书的结构

本书分成三个部分，第 1~5 章讨论的是关于 Windows Forms 开发所需的基础知识，其中包含了.NET Framework 概念、ADO.NET、Windows Forms 和.NET Remoting 四种技术。第 6~13 章则探讨 Framework，包含了 Framework 的概论，如何设计、实现及测试等知识。第 14~17 章则以一个简单的进销存系统为实例，展示如何使用此 Framework 来开发一个可用于真实世界的应用程序。

建议的阅读顺序

以一个作者而言，我当然希望读者们可以逐章地阅读本书，毕竟那是我所喜欢的编排，不过这只是笔者的主观认知，对于不同层次的读者，笔者有几个阅读本书的建议。以初学者而言，笔者建议逐章阅读，这可以让读者以较平滑的方式提取各章节中的知识。对于有一定基础者，笔者建议以较轻松的心态快速浏览第 1~5 章，这些章节中所讨论的虽然是基础知识，但其中仍然穿插许多实践上可能遭遇的难题及解法，跳过这些，读者可能就无法理解第 6～13 章中一些 Framework 为何会设计成那样子。不管读者是初学者还是有一定基础者，笔者都建议读者一定要将附书光盘中的范例安装到计算机中，本书所讨论的程序代码相当庞大，无法在书中一一列出，只能挑出其中的重点列出，其他的细节程序代码得从附书光盘中取得。

开发工具

本书使用 Visual Studio 2005 作为开发工具，搭配 SQL Server 2005 数据库系统，

范例在哪里

本书随附的光盘中包含了书中所有的范例及组件，提醒读者，书中的 Framework 范例列表以书末小型进销存系统所使用的那一版为主，如有变动，光盘中仍会保留原文件，新文件将以.new 文件名结尾，这代表着该.new 文件是修改但未经测试的版本。

技术支持网站

我拥有一个个人网站，在本书出版后，网站中会维护一份勘误表及范例更新记事，网站中也会不定期地发表相关的文章及范例。

网址：*http://www.dreams.idv.tw/~code6421*

前言

在写第一本书之前，我从来不觉得写一本书有多难，在认知上，这应该只是将自身的经验，对技术的理解以文字来阐述，最难的部分应该只在于打打字、抓抓图和写写例子而已。但等到我开始着手工作时，发现问题并非这么简单，一个技术可以有多种观点，多种用法，每一种都能够解释该技术，每一种用法都能够达到目的，要从这里面萃取出正确、可以用来引导读者的一种是相当困难的事，当然，我曾经告诉自己，别想太多，只要忠实地写出自己所理解的部分，忠实地告诉读者，我是使用哪种技术来达到目的的，就算是对读者尽到责任了，但是我仍然无法以此说服自己，我依然不免会想，我对这个技术的观点是否正确，这个用法是否是该技术正确的用法，这使得每写一章，我就会一直回想，该章中的内容是否全部正确，有没有可能有别的、更好的用法还是观点可以取代，最末润稿时，这个动作将重复一次，电子稿上的校稿又再重复一次，纸版上的校稿又重复一次，周而复始的结果是，一本书写了半年，纸版校稿用了一个月，直到出版社下达最后通牒，不准我再加上一行字、一小节，只能改错字后，我这才收起字字计较的心，乖乖地做完最后一次的错字校对，停止对排版人员的虐待，让出版社能顺利出版。说真的，当出版社下达最后通牒时，我心中似乎有块大石落下，终于，我的责任尽完了，可以安心休息了，这是言语无法形容的舒畅。在这本书中，同样的感觉如影随形，而且更加强烈，这是因为这本书的题材所致，书里除了阐述 Windows Forms、ADO.NET 和.NET Remoting 等技术的概念及用法外，还阐述如何以它们为基础，设计一个更高级的 Framework，协助设计师快速地开发应用程序，是的，这就是压力的来源，该如何说服自己，开发的 Framework 真的能达到此目的？Framework 中所使用的技巧，是否是正确的、有效率的呢？为了这个，我应用此 Framework 写了一个小型进销存程序，目的就是为了证明，这个 Framework 是真的可以达到预期的目标。但事情并非到此为止，当我做到这点时，我又开始怀疑，Framework 中的技术是否有更好的解法，更有效率的手法，为此，我又花了几个月的时间重新审视 Framework 的设计面及实现面。时间永远不够用，这是我写这本书的感想，在写这篇前言前，我重新思考，事何以至此？最终获得一个答案，技术阐述者与技术

创作者的角色重叠在一起了，也就是作者与程序员的角色重叠在一起了，作为一个程序员，我只求程序能正常运作，从不考虑完成该需求的手法是否有效率，又是否有其他更好的做法，除非，该功能明显很没效率或是出现致命的错误，否则，只要程序能动，那就是正确的。但作为作者，我无法说服自己，将一个会动的程序，但做法可能不是最好的、最合宜的构想传达给读者，文字业！就是怕造文字业啊！不管过程如何艰辛，内心爬过了多少挣扎及妥协，当你看到这篇前言时，也代表了这本书已经出版了，一切皆已尘埃落定，至少，在有限的时间内，我做了最大的努力，将最好的、通过数次检验的内容呈现在了读者眼前。

黄忠成

2006/5/23 于台北

目录