Broadview
WWW.BROADVIEW.COM.CN

来自开发第一线的项目全过程实例分析！

道法自然

面向对象实践指南

王咏武
王咏刚
著

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

道法自然——面向对象实践指南

王咏武　王咏刚　著

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

内 容 简 介

本书是一本试图用实战案例阐释面向对象技术体系的指南。本书共分 19 章，以实际的开发案例——FishGUI 项目为主线依次介绍了需求和用例分析、面向对象分析、架构分析、面向对象设计、设计模式、编码技巧等几个主要的技术领域，并基本按照时间顺序，描述了 FishGUI 系统设计和实现的全过程。

如果把面向对象大师们偏重理论建构的经典读物称为面向对象盛宴中的主菜，那么，这本指南若能充当佐餐的凉菜或是饭后的甜点，作者就心满意足了。本书适合所有在软件开发领域辛勤工作的开发人员、管理人员、系统分析人员、测试人员、技术支持人员以及广大在校学生阅读。

图书在版编目（CIP）数据

道法自然：面向对象实践指南 / 王咏武，王咏刚著. —北京：电子工业出版社，2004.10
ISBN 7-121-00226-4

Ⅰ.道… Ⅱ.①王… ②王… Ⅲ.面向对象语言－软件开发 Ⅳ.TP311.52

中国版本图书馆 CIP 数据核字（2004）第 080840 号

责任编辑：毕 宁 bn@phei.com.cn
印 刷：北京智力达印刷有限公司
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：787×980 1/16 印张：29 字数：556 千字
印 次：2004 年 10 月第 1 次印刷
印 数：5000 册 定价：45.00 元

凡购买电子工业出版社的图书，如有缺损问题，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。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。

序　言

自从2000年面向对象设计领域内的里程碑著作《设计模式》被译成中文出版后，面向对象和软件工程领域的经典名著就陆续被各出版社大批引进。模式、框架、UML、用例、MDA、AOP这些新潮名词开始不断出现在专业报刊杂志中，程序员的求职简历中也都纷纷加上了精通面向对象的字句，但一阵热潮过去后，开发者的实际设计水平提高了多少呢？很遗憾，从业界的实际反馈来看，一般开发者的面向对象设计水平并没有实质性的提高。

问题出在哪里呢？我们认为，关键在于经验。国外译著的理论水平很高，但是理论如何落实到实践，这是一个经验的问题。我们出版的技术图书当中，能够向读者传授实践经验的不多。图书市场上充斥了不少销售得还不错的软件开发案例丛书，它们本应当承担起教授读者实践经验的责任。但是它们的表现如何呢？大部分案例完全没有OO的设计思想，没有设计过程，没有重构，没有复用，重复代码到处出现，甚至还有不少错误的设计和代码。就是这样的图书，却能影响成千上万的学习者，这样学习下来的结果，当然不会令人乐观。

在软件开发的第一线人员中，OO 的应用到底是什么样的景象呢？按实际应用的水平可以把软件开发人员分成五个层次：

第一层是不管什么OO，直接编码，把功能实现了项目就是成功；

第二层是理解了几个面向对象的典型概念——封装，继承，模式。囫囵吞枣，不管实际效率如何就开始应用，为OO而OO；

第三层是在设计阶段摆出一副OOAD的架子，进入编码实现就逐步把OO放到了一边，OO与实践分离；

第四层是基本掌握OO理论，在具体应用中能大量使用OO技术来进行系统分析和程序设计；

第五层是把OO思想融会贯通，不管用不用OO技术，都能选择最佳设计和实现方法。

我喜欢把设计模式类比成围棋定式，OO 技术同样也可以类比成围棋技术。在围棋中定式的定义："在局部战斗中，用最稳妥的顺序，而且能经得住以后的检验，从而被固定下来的就是定式"。

在软件开发中模式的定义："将一类不断重复发生的，类似的问题以及该类问题的解决

之道总结出此类问题的共同点并抽象成一定的描述及规范，以便遇到此类问题无须再做过多的考虑，直接使用已经总结好的解决之道。我们称之为模式”。

我们可以看到这两者的类似，围棋入门容易，精通难。抛开入门级别，围棋手按水平从业务初段到职业9段还可以划分16个等级，其差距就在于棋手们在实践中对技术的综合使用能力。

上面这几个层次的差别也在于此，要学习OO理论和方法并不难，难的是在实践中如何用好面向对象技术。OO大师们提出的理论和方法都有着深刻背景，剖析了软件开发过程，经过了提炼和抽象，大师们的名著是学习的基础，也有很多来源于实践的例子，但每个具体项目有着千差万别的情况，如果不能结合具体实践去灵活应用，可能很难达到较高的水平。

如果能跟踪学习一个完整项目的进展过程，对于在实践中全面掌握OO会有很大益处。

长久以来，我一直希望在《程序员》杂志上开设这样的栏目，针对真正的第一线实战项目，记录一个项目从分析到设计，并实现的思考全过程；类似于MBA教学中的案例分析，类似于围棋中的名局细解，从实践中去领悟体会OO技术的内涵。可是很困难，大部分的实践项目都不愿意完整地公开，另外，真正的专家高手又很难花时间去做这样的工作。

这本书弥补了这个遗憾，本书的作者有着多年丰富的软件开发经验，以一个真实的项目——FishGUI为基础，全面记录了项目的实施，并对每一个技术过程做了分析总结。凝聚了他们对OO技术的解读与实践，提炼出了经过咀嚼的精华。

读者可以跟着这个项目实施的步伐，从需求分析，用例建模，OO分析，OO系统设计，模式应用一步步去真实体验。书中精彩的部分包括对各种理论的提炼和解读，既讲清了这些纷繁技术的来龙去脉和关系，又点出了OO技术的关键点。华罗庚先生说过，读书要先把书读“厚”，再把书读“薄”。这些总结就是作者把OO技术读薄的成果。

“一花独放不是春”，只有不断交流，知识才能分享，技术才能进步，我衷心希望看到更多的第一线专家分享自己的成果，看到更多这样来源于实践的作品。

《程序员》杂志社社长

自　序

程序员的生活天天都在改变，不变的只有求知的乐趣和创新的快感；面向对象的世界时时都在进步，永恒的只有技术的纯洁和科学的谨严。

面向对象技术让许多软件项目由难变易，也让不少程序员的学习过程由易变难。不是我们迟钝，也不是我们偷懒，谁让面向对象的海洋如此浩瀚？

计算机科学在根本上是实践的科学，面向对象的技术也不例外。记熟了三千佛法，却不懂得定慧双修的出家人不见得就能大彻大悟、见性成佛；背诵了“封装、继承和多态，模型、模式加模板”，却不知道联系实际的程序员也不见得能开发出优秀的软件。

从实践中来，到实践中去，这是学习面向对象技术的不二法门，也是我们在写作本书时努力追求却未必能窥其堂奥的理想境界。但愿本书能为读者步入面向对象的殿堂提供些许指引和帮助；除此以外，我们别无他求。

《老子》中说：“人法地，地法天，天法道，道法自然。”如果可以妄解先贤的名句，我们倒宁愿把这里的“自然”理解为——实践。

王咏武　王咏刚

2004 年 7 月

前　言

——一份关于本书的 FAQ

我讨厌长篇大论的“前言”——为什么每一本严肃的图书都要用正文以外的篇幅浪费读者的大量时间？我也讨厌喋喋不休的“致谢”——IT 书籍又不是奥斯卡颁奖典礼上的那些除了“感谢”二字就再也不知道如何谦虚的大牌明星。所以，大家在此处看到的前言只是一份简单的 FAQ（常见问答，最有助于节省程序员时间的文档形式之一），读者完全可以像使用其他 FAQ 那样在这里快速查找自己需要的信息。

为什么需要一本面向实践的指南

每 1000 个学习面向对象技术的程序员中，有 997 个都要将学到的知识付诸实践；在另外的 3 个人中，一个是只靠做研究和写论文就可以衣食无忧的理论家，另一个是试图记住 800 种以上的技术名词以便在人前炫耀的牛皮家，最后一个是根本不知道面向对象为何物却非要摆出头悬梁、锥刺股的架势博取老板欢心的钻营家。

显然，99.7%的人都需要在学习面向对象技术时，更多、更好地掌握理论和方法在实际项目中的应用秘诀。在此过程中，面向对象大师们的经典著作当然都是必读的教材。但实际工作中有太多太多的案例告诉我们，像赵括那样只会纸上谈兵的程序员迟早会在冷酷的现实中一败涂地。也许，一本面向实践的指南读物可以帮我们更快地适应面向对象的思维方式，更容易地将书本知识应用到实践中去——我们有什么理由拒绝这样一本实用的书籍呢？

谁需要这样一本指南

绝大多数听说了也见过了面向对象的神奇魅力，却始终无法真正迈进面向对象大门的程序员需要这样一本指南；绝大多数在面向对象领域里刻苦攻读、努力实践，却迟迟不能看到美好回报的程序员需要这样一本指南；绝大多数在面向对象的开发过程中屡战屡败、灰心丧气，恨不得明天就去国足寻找同病相怜者的程序员需要这样一本指南；绝大多数对面向对象大师们的鸿篇巨著佩服得五体投地，却总也猜不出大师们在字里行间究竟隐藏了多少玄机的程序员需要这样一本指南；当然，绝大多数好奇心强烈，想了解本书到底有多大价值，想知道本书作者究竟是何许人也的评论者也不妨纡尊降贵，先读过本书再发表意见。

补充说明一：本书的读者最好能掌握一门面向对象的程序设计语言，C++、Java、C#、

Delphi 等语言均可。尽管本书案例使用的是 C++语言，但大多数面向对象的语言都有着类似的精神实质，都能很好地体现本书的主旨。如果你对面向对象的程序设计语言还比较陌生，那不妨在阅读本书的同时学一学 C++的语法和编码技巧——有这样一举两得的机会，你又何乐而不为呢？

补充说明二：如果你了解 UML 语言的基础知识，阅读本书时，你就不会对着一幅幅抽象的图形发呆；如果还没有接触过 UML 语言，那不妨找一本《UML 用户指南》之类的书籍，在阅读本书之余，顺便补习一下相关知识。

补充说明三：本书所说的“程序员”是指广义的程序员，包括所有在软件开发领域辛勤工作的开发人员、管理人员、系统分析人员、测试人员和技术支持人员，等等。是否选择本书是读者的自由，但我们不想因为使用更亲切、更顺口的“程序员”一词而将许多潜在的读者拒之门外。

为什么要结合一个实际的案例

这个问题纯属多余：一本自称面向实践的指南读物，如果不紧扣一个实际的开发案例，不就和《笑傲江湖》中林震南所使的辟邪剑法一样徒有其表了吗？

惟一需要说明的是，贯穿本书始终的案例——FishGUI 项目（一个有趣的名字）是本书作者王咏武亲身参与的一个软件开发项目。无论项目的最终效果如何，我们从 FishGUI 的开发过程里获得的许多经验、教训都值得和大家分享。FishGUI 产品本身属于通用的 GUI（图形用户界面）框架系统，其用途类似于 Java 的 Swing，但功能远比后者简单。因为强调通用性和灵活性，这一类系统中往往包含了最多的面向对象知识和最丰富的开发技巧，用这样的例子来论证面向对象领域的基本理论和实践准则，再合适不过了。即便 FishGUI 的设计和实现还存在着诸多不足，但在书中把这样的例子详详细细展现出来，供方家指正，让读者品评，这不也是一种最好的交流方式吗？

为什么每章的标题都包含了一组对立统一的概念

的确，本书正文中每一章的标题都包含了一组对立统一的概念。采用这种写法的主要原因是，我们对面向对象技术有一个最根本的认识：

> 面向对象领域充满了辩证关系，开发者不得不在每一个设计和实现环节中仔细权衡、择善从之。

许多急功近利的程序员都喜欢在开发时把某种技术、架构或设计方案的作用绝对化。

在他们眼中，一种好的技术就理应放之四海而皆准，一种成功的设计方案也绝不存在什么适用场合和环境的问题。套用软件工程界广为流传的一个说法，这些人其实都是“银弹万能论”的忠实拥趸。

实际上，软件开发和所有人类活动一样，都是充满了矛盾的复杂过程。概括起来，绝大多数软件开发都是由一次又一次的选择所组成的：我们必须根据实际情况，谨慎地选择模型、架构、模式、语言、接口、流程、算法……在每一次选择时，我们几乎都会面对两难的境地：对同一需求，往往有两种以上的方案可以奏效，但每一种方案都各有其长处、缺陷和适用范围。没有哪种技术完美无缺，也没有哪个负责任的程序员能够用掷硬币的方式做出取舍。

在面向对象的实践中，我们只有承认这种矛盾关系，才能更好地认识和把握它，并在对立统一的思辨中找到最适合当前情况的解决方案。从这个意义上讲，面向对象领域里那些游刃有余的高手们，都天生有着黑格尔般的敏锐头脑和缜密思路，都是IT世界里的业余哲学家。

如何阅读本书

本书共分19章，依次介绍了需求和用例分析、面向对象分析、架构分析、面向对象设计、设计模式、编码技巧等几个主要的技术领域，并基本按照时间顺序，描述了FishGUI系统设计和实现的全过程。对于大多数以学习为目的的读者，我们推荐从头至尾、循序渐进的阅读方式；如果你对其中的某个或某几个领域早已了如指掌，当然可以采用跳跃式的阅读办法，只翻检你最关心的章节。

既然是一本面向实践的指南读物，我们在此强烈建议大家在阅读时多多动手：遇到不明白的UML图示，不妨在Rational Rose等工具中勾画几下；碰见繁难的代码问题，打开你的IDE调试本书的示例代码自然是最好的选择；假如在阅读过程中，你的脑子里突然跳出了某个新鲜的念头，那你应该干净利落地把书抛在地上，然后飞快地跑到计算机前，在面向对象的开发工具中记录或验证你的灵感！

什么是“笨笨点评”

本书大多数章节里都会随机出现一两段“笨笨点评”。这里的“笨笨”就是王咏刚，也就是那个慵懒闲散、不务正业、还时不时在报刊上发两篇笨拙文章的家伙了。所谓“点评”，就是王咏刚在整理和删改本书的过程中（本书的主要技术思想来自第一作者王咏武），因为大脑短路或其他类似的原因，突然想到并随手记下的各种稀奇古怪的文句，其中既有技术的随想，也有历史的回眸，既有肆意的评论，也有大胆的臆测。总之，大家千万别把“笨

笨点评”当正文来读，高兴的时候尽管当它是下酒的凉菜，不高兴时对它视而不见也未尝不可。

为什么使用 C++语言

为什么不呢？C++语言是普及面最广、适用性最强、开发工具最多、开发成果最丰富的程序设计语言之一。既然本书的案例终归要由某种特定的程序设计语言来实现，那为什么不选择著名的 C++呢？而且，现实中的 FishGUI 项目使用的就是 C++语言，FishGUI 的最终产品在执行效率、可扩展性、可移植性等方面的表现也都令人非常满意，我们有什么理由放弃 C++语言呢？

我们知道，有人一定会说出 C++太复杂、C++不如 C#时髦等冠冕堂皇的理由。我们不想为此多费口舌。但需要强调的是，在面向对象的领域里，只要你真正掌握了面向对象的基本理论和基本方法，使用何种语言对你来说并不重要。举例而言，如果你是不折不扣的 Java 迷，那你完全可以在阅读本书的同时，随着 FishGUI 的设计和开发的进程，用 Java 语言把 FishGUI 重写一遍，这可是一种最好的实践方式呀！到那时，如果你能把你的大作寄给我们学习和参考，我们将不胜荣幸。

FishGUI 到底有什么用

FishGUI 系统是本书中最重要的开发案例，也是一个真实存在过的软件产品。很多在 Windows 或 Linux 系统中开发过 GUI 应用的读者可能大惑不解：既然已经有 MFC、.NET Framework、AWT、Swing、SWT、Tcl/Tk、Qt 等许多 GUI 开发环境可用，为什么还要自己开发 FishGUI 这样的 GUI 框架呢？我们觉得，除了为本书提供案例支持外，FishGUI 的存在价值至少还包括以下几个方面。

- 我们可以很容易地将 FishGUI 移植到各种嵌入式或实时操作系统（RTOS）中。对于在那些缺乏 GUI 支持的操作系统中开发应用软件，或者没有实力购买商用 GUI 开发包的用户来说，FishGUI 显然可以助他们一臂之力。
- 同样的道理，基于 FishGUI 开发的应用软件也可以快速地移植到不同的嵌入式或实时操作系统中，并保持相同的外观和使用体验（Look and Feel）。
- 嵌入式软件开发者可以利用 FishGUI 在 Windows 或 Linux 系统下建立一个模拟开发环境，并在该模拟环境中利用 Visual Studio 等通用 IDE 快速完成图形用户界面的开发、调试工作，然后将应用程序不加修改地移植到嵌入式或实时操作系统中。
- 通过研习 FishGUI 系统的架构，我们可以更好地理解 MFC、.NET Framework、AWT、Swing、SWT、Tcl/Tk、Qt 等复杂 GUI 开发包的实现机理，这显然有助于提高我们

的 GUI 开发水平。

怎样下载和使用本书的源代码

本书各章节的源代码，FishGUI 系统的所有 UML 模型文件、文档、源代码和相关说明文件都可以从网上直接下载。网址是：

http://www.contextfree.net/wangyw/ooguide/

你可以使用 Rational Rose 2002 或更高版本打开 FishGUI 系统的 UML 模型文件；在 Windows 系统下可使用 Visual Studio .NET 2003 或 Visual C++ 6.0 编译所有 C++工程和源代码（Visual Studio .NET 2003 也用于编译少量 C#语言示例代码）；在 UNIX/Linux 系统下可使用 gcc 3.2.2 或更高版本编译 C++工程和源代码。

有关 FishGUI 系统本身更详细的生成、安装和使用说明请参见本书附录及 FishGUI 源代码中所附的说明文档。

如何反馈

本书作者都是普通的程序员，水平有限，书中的错误也在所难免。我们欢迎所有肯牺牲自己的宝贵时间，就本书的内容、文字及体例不吝赐教的读者直抒己见，我们也欢迎所有熟识的、陌生的、有缘相逢的抑或未曾谋面的良师益友随时提供建设性的意见。谢谢！

邮箱地址：wangyw@contextfree.net

王咏武　王咏刚

2004 年 7 月

目　录