

UML

面向对象设计基础

[美] Meilir Page-Jones 著
包晓露 赵晓玲 叶天军 唐亚东 译



UML 面向对象设计基础

[美] Meilir Page-Jones 著

包晓露 赵晓玲 叶天军 唐亚东 译

人民邮电出版社

UML 面向对象设计基础

- ◆ 著 [美] Meilir Page - Jones
译 包晓露 赵晓玲 叶天军 唐亚东
责任编辑 陈冀康
- ◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街 14 号
邮编 100061 电子函件 315@pptph.com.cn
网址 <http://www.pptph.com.cn>
读者热线:010-67129212 010-67129211(传真)
北京汉魂图文设计有限公司制作
北京鸿佳印刷厂印刷
新华书店总店北京发行所经销
- ◆ 开本:787×1092 1/16
印张:21
字数:512 千字 2001 年 4 月第 1 版
印数:1-5 000 册 2001 年 4 月北京第 1 次印刷

著作权合同登记 图字:01-2000-0658 号

ISBN 7-115-09154-4/TP·2109

定价:36.00 元

内容提要

本书是关于面向对象软件设计的基本概念、符号表示、术语、准则以及原理的书籍。

本书第一部分（第 1 章和第 2 章）介绍了面向对象的基本概念以及面向对象的发展史。第二部分（第 3 章至第 7 章）较全面地介绍了统一建模语言（Unified Modeling Language, UML），UML 已成为描述面向对象设计符号的事实上的标准。第三部分（第 8 章至第 14 章）较深入地介绍面向对象设计的原理。本书最后一章（第 15 章）分析软件构件的特点，包括优点和缺点。

本书针对采用面向对象技术的程序员、设计人员、系统工程师或技术经理，无论您是面向对象技术的初学者，还是具有一定面向对象技术经验的高手，都将从本书中受益匪浅。

参加本书翻译工作的有：包晓露、赵晓玲、叶天军、唐亚东、刘东红、黄飞、张捷、雷英、魏威、焦跃、肖鸿飞、朱国平。

版权声明

Meilir Page-Jones: Fundamentals of Object-Oriented
Design in UML

Copyright © 2000 by Dorset House Publishing.

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise without the prior consent of Addison-Wesley.

Published by arrangement with Addison Wesley Longman, Inc. All Rights Reserved.

版权所有。未经出版者书面许可，对本书任何部分不得以任何方式或任何手段复制和传播。

人民邮电出版社经 Addison Wesley Longman 公司授权出版。版权所有，侵权必究。

人们评论本书时向我提出了几个问题，你可能也关心这些问题。下面我回答其中的一些问题。

我是一名程序员。为什么要关心设计？

每个人在编写代码的同时，也在设计（design）代码，不管是好是坏，还是有意或无意。我写本书的目标是鼓励 O.O. 专业人士（这类人士越来越多）有意识地在编程之前建立良好的面向对象设计。为此，我介绍符号表示法、面向对象的原则以及术语，使你及你的同事可以对设计的系统进行评估和讨论。

本书将教会我们 O.O. 的编程语言吗？

回答是否定的。尽管我偶尔会提到程序，但本书不是关于面向对象编程的。

我正在学习面向对象语言，本书会有帮助吗？

回答是肯定的。如果你目前还不了解一种面向对象语言，可以从第 1 章中获得面向对象的知识。了解面向对象的主要概念可以加快你对一种面向对象语言的学习，并且能增强你跨入陌生领域的信心。本书后面有关如何设计的章节也将有助于使你以前的程序运行得更好。

另一方面，如果你是一位有经验的面向对象程序员，可以通过本书的第二和第三部分提高设计水平，这点对于成为全面的、专业的软件设计者或编程者是至关重要的。

为什么不用 C++ 编写本书的程序例子？

本书中的程序采用我自己发明的语言编写，该语言融合了 4 种流行的语言：C++，Eiffel，Java 及 Smalltalk。我这样做是因为有两类程序员：一类是熟悉 C++ 的，一类是不熟悉 C++ 的。如果你是 C++ 的爱好者，会发现本书的程序可以很轻松地转成 C++ 程序。如果你不熟悉 C++，可能会厌烦其神秘的语法。有些例子是用 Java 写的，这是因为非 Java 程序员接受 Java 比非 C++ 程序员接受 C++ 要容易些。我希望无论你的编程语言是什么，本书都能

适合你。

为什么本书不是针对窗口、图标及菜单设计的？

原因有两个：其一，我不认为面向对象只适用于图形用户界面的设计；其二，在市场上有许多专门针对面向对象窗口设计的书。我希望本书能包括一些其他面向对象的图书没有涉及到的内容。在第 7 章提供了窗口导航设计的内容。

本书是有关方法学的吗？

回答是否定的。我们知道，开发方法学的内容比设计要多得多。例如，方法学还包括需求分析、库管理等等。而且，真正的方法学应该解释各种开发活动是如何有机地结合在一起。内容很多！

因此，我不打算将本书写成和其他许多有关面向对象的书一样内容松散，而集中于一个主题：面向对象设计。

你说了本书许多不涉及的内容，但本书涉及哪些内容呢？

本书主要涉及面向对象软件设计的基本概念、符号表示、术语、准则以及原理。面向对象软件是由对象以及其所属的类构成的软件。一个对象是一个软件构件，其操作（类似函数或过程）与一系列变量（类似数据）有关。类实现了一种类型，它定义属于该类的一组对象。

上述朴素的语句蕴藏对于软件设计者和编程者非常有用的内涵，由此引出了继承、多态性及二次设计等的设计概念。但由于你问了一个特定的问题，所以我给你一个专门的回答。

本书第一部分（第 1 和第 2 章）介绍面向对象。第 1 章综述了主要概念，并揭开了“多态性”、“一般性”及其他所有 O.O.行话的神秘面纱。第 2 章将面向对象置于以前的软件开发框架中。如果你已经熟悉面向对象（也许用面向对象语言编过程序），那么可以跳过第一部分。

第二部分（第 3 至第 7 章）介绍有关 Unified Modeling Language（统一建模语言，UML）的内容，UML 已成为描述面向对象设计符号的事实上的标准。此外，第二部分还说明了许多可以在面向对象系统中找到的结构。第 3 章介绍用 UML 描述类及其属性和操作。第 4 章介绍用 UML 表示关联、聚合、组合对象，以及子类 and 超类的层次。第 5 章说明如何用 UML 表示消息（顺序的和异步的），第 6 章介绍用 UML 描绘状态图。第 7 章总结 UML 用于系统结构和人机交互界面的窗口的内容。

第三部分（第 8 至第 14 章）较深入地介绍面向对象设计的原理。第 8 章讲述共生性的关键概念及 2 级封装。第 9 章探讨各种类所属的域并描述不同的类的内聚程度。第 10、11 章是第三部分的主要内容，将状态空间和行为的概念应用到既合理又可扩展的类层次结构。

第 12 章提供了一些轻松的调节剂，从现实世界中检验设计，既有精巧的设计也有可笑的设计（第 12 章确实列举了设计者有关继承和多态性的可笑设计）。第 13 章给出一些如何组织给定类操作的方法，并说明提高可重用性和可维护性设计的技术，如混合类和操作环。

第 14 章 触及一个令人头痛的问题：“如何设计一个好的类？”为回答这个问题，第 14 章描述了从可怕的到美妙的各种类接口。具有典型接口的类易于进行抽象数据类型的实现。如果类的设计遵守前面几章介绍的基本原理，这样的类更具有健壮性、可靠性、可扩展性、可重用性和可维护性。

作为本书结束的第 15 章分析软件构件的特点，包括优点和缺点。在商用应用软件的面向对象开发过程中，回顾了在前面几章中介绍的一些面向对象原理。

尽管我在主要的论述中增加了丰富的例子和习题，但必须承认第三部分中有些内容比较难。但不管怎样我不想弱化重要问题。面向对象设计的某些方面内容是比较难的，值得一提以免引起误解。

本书包括了面向对象设计的所有内容吗？

我对此深表怀疑。每天，我都能学到许多有关面向对象的内容，相信你也一样。实际上，一本书可以告诉我们面向对象的一切而不需要我们再学习，是十分荒谬的。本书中的内容一定有不当之处！在我写完几本书之后，随着年龄的增长和阅历的增加会改变我的一些观点。

因此，尽管我在本书中包括了许多重要的设计原理，如果你对面向对象十分重视，应尽可能广泛阅读并挑战你所学的知识。

你提供面向对象设计的课程吗？

是的。我所在的公司 Wayland System 提供一些面向对象专题的课程。我们的课程经常有变动，有关最新信息请查阅www.waysys.com。

这本书适合我吗？

这是什么问题？你希望我说“不？”。如果你已经是或是将要成为采用面向对象技术项目的程序员、设计者、系统工程师或技术经理，那么本书适合你。如果你是面向对象的初学者，将从本书的第一部分学到许多知识，然后做些面向对象编程的练习，再回到本书的第二和第三部分。

如果你是已经掌握了标准的结构化编程的大学生或职业程序员，希望进一步扩大知识范围，本书也适合你。

无论你属于何种角色，希望你能喜欢本书并从中受益。

祝好运！

Meilir Page-Jones

1999 年 9 月

华盛顿 Bellevue

E-mail 地址: [meilir @ Waysys.com](mailto:meilir@waysys.com)

对象已成为当今软件中普遍存在的构件，面向对象也成为软件工程实践中普遍使用的技术。种种有关面向对象的书早已面市，但当本书的第一版《What Every Programmer Should Know About Object-Oriented Design (面向对象设计程序员必读)》出版时，该书立即被公认为出自一位当今软件开发中最明智的思想家和通俗作家之手的、新颖而有独创意义的著作。

在全新的重新修订并重新命名的第二版中，扩充了基础内容，增加了素材，修改了符号表示以符合最新的潮流。本书融入了面向对象开发的新思想，从对于继承使用的剖析，到如何针对对象类之间疑难数据关系建模。Page-Jones 不愧为最新智慧的代表。

作为咨询专家和设计者，作者几十年来一直工作在第一线，本书的每一页倾注了作者的经验和教训。我曾与他在同一“战线”中工作，特别是最近在一个大型项目中一起合作，初始的用例 (use-case) 模型中的用例就超过 340 个。随着读者对本书的学习，您将看到作者还是一个实用主义者，对基础和细节的注重反映在他的分析和设计研究及其作品中。

事实上 Meilir 还是一位有天赋的教师，他擅长澄清复杂、难以理解的概念，形成一个概念上的模式使其变得清晰。他可以列举一大堆问题并将其包揽在一个原形例子中，使得一切都变得十分明朗，让其他人无不为之感叹：为什么我们没想到。以面向对象的方法考虑农场牛奶问题，挤奶的时间到了你会做什么呢？给 Cow 对象发送一个消息让其挤奶还是给 Milk 发送一个消息让它自己脱离 Cow？瞬间的反映及事件管理的需求使协调挤奶的过程变得十分清晰。他的许多阐述例证已成为面向对象中广为流传的一部分内容，例如来自会议讨论组记录的例子，以及“Person Owns Dog”的难题。

本书恰当地说明了如何从长远利益出发建立好的设计原则，这些原则已在实际中广泛使用，并在最先进的语言和挑战环境中开发面向对象系统被采纳。建立在这种基础上，本书基于实际经验解决复杂的实际问题，通过浓缩的例子所介绍的经验，为新老开发者设

计更好的面向对象解决方案提供了指导。

Meilir 作为一位咨询专家、一位教师或一位方法学家，在面向对象开发方面积累了丰富的经验。他是 Synthesis 方法的创建者之一，这种方法是早期的面向对象分析和设计的语义方法之一，并且，在建立流行的 Uniform Object Notation(统一对象符号表示法)中我们是合作者，这种方法可以在当今众多的面向对象方法和表示法中找到。我们的研究成果甚至得到了 Unified Modeling Language (统一建模语言, UML)的认可，UML 已经成为事实上的工业标准，本书采用该标准对样例进行说明。

在这里可以找到掌握面向对象设计基础需要的所有内容。不仅包括清晰易懂的设计和建立对象的基本技术，而且包括丰富的范例，详细地讨论一个好的面向对象系统应该怎样而不应该怎样。

Larry Constantine

1999 年 9 月

Massachusetts, Rowley

致 谢

如果专门致谢所有应该感谢的人，那么致谢词可能要延伸到 94 页。因此，我对多年来鼓励、帮助过我的众多同事一并感谢。但对于一些朋友，我要给予特别致谢：

- 对本书的第一或第二版提出宝贵意见的专家有：Jim Beaver, Bob Binder, Ken Boyer, Tom Bragg, Tom Bruce, Martin Fowler, Alan Hecht, Mike Frankel, Cris Kobryn, Jim Odell, Tim Ottinger, Ben Sano, Mike Silves, Jess Thompson, Lynwood Wilson 及 Becky Winant。他们的建议非常有用，有些建议令人沉思，而有些建议则令人鼓舞。这些建议使我避免了更多的愚蠢错误并进行了许多重大改进。感谢他们的鼓励，使我不断地改进本书。
- Kendall Scott 对本书手稿所作的特别的工作。
- Steve Weiss（与 Walter Back 一起）早在面向对象开始流行以前就劝说我转到面向对象领域。没有 Steve，我不可能从事本书的工作。
- Larry Constantine 是实施软件设计原则的鼻祖。
- 健谈的、多才多艺的 Stan Kelly-Bootle 给予我很大的支持（不久我将回报你的支持）。
- Bert Weedon，他惊人的音乐艺术天赋和在计算天体物理学中的科学天赋一样出色。

- Ed Yourdon 及 Tim Lister 长久以来的信任。
- Bertrand Meyer 使我不断获得灵感的源泉。
- Paul Becker 多年的鼓励。
- Roland Racko 与我分享了其许多的观点。
- 我阅读过的有关面向对象的书的 97 位作者。
- Miller Freeman 公司的《软件开发杂志》(Software Development magazine) 的 Jolt Panel, 他授予我在 Dorset House 出版的本书第一版 1996 年度产品奖。
- Nuno Andrade, Michael Richter, Michael Lumelsky, David McClintock, Wendy Eakin 及 Dorset House Publishing 出版公司的其它人员, 这些高水平的专家们工作勤勉努力和掌握的最新技术。他们克服了相当大的困难, 彻底消除了原稿中的具有二义性的模糊词语、以动名词开头难以理解的长句子、指代不清 (比如 this one)、牵强附会 (甚至词不达意)、不一致的时态, 以及那些华丽而无助于产生一个专业论著的辞藻。再次感谢朋友们的出色工作!
- Cater Shanklin 及 Addison Wesley Longman 的职员, 由于他们的努力才将本书的这一版本推向市场。
- 与我一起分享他们的经验的遍及世界各地的咨询客户和研究会的学生们。
- 我的家人在此书撰写时的隐居式生活给予了极大的耐心。
- 最后, 特别感谢下面两位业界资深人士: Sid Dijkstra 教授对软件过程做出了不可估量的贡献; Kedney Dogstar, 传奇的编程牛仔 (Legendary Coding Cowboy), 他是 Extreme Debugging 的发明者, 他在 Bramston Capra Consulting 咨询公司时利用午间编写的程序比我们大多数人一生编写的程序还要多。

第一部分 引言	1
第 1 章 面向对象的含义	3
1.1 封装	7
1.2 信息/实现隐藏	9
1.3 状态保持	11
1.4 对象标识	11
1.5 消息	14
1.5.1 消息结构	15
1.5.2 消息参数	15
1.5.3 消息中的对象角色	17
1.5.4 消息的类型	18
1.6 类	19
1.7 继承	23
1.8 多态性	27
1.9 一般性	30
1.10 本章小结	34
1.11 习题	35
1.12 答案	36
第 2 章 面向对象简史	40
2.1 面向对象的起源	40
2.1.1 Larry Constantine	40
2.1.2 O.-J. Dahl 和 K.Nygaard	40
2.1.3 Alan Kay, Adele Goldberg 等人	41
2.1.4 Edsger Dijkstra	41
2.1.5 Barbara Liskov	41
2.1.6 David Parnas	41
2.1.7 Jean Ichbiah 等人	41
2.1.8 Bjarne Stroustrup	41

2.1.9 Bertrand Meyer	42
2.1.10 Grady Booch, Ivar Jacobson 及 Jim Rumbaugh	42
2.2 面向对象的成熟期	42
2.3 类似工程学的面向对象	43
2.4 面向对象的益处	44
2.4.1 用户需求分析	45
2.4.2 软件设计	45
2.4.3 软件构造	46
2.4.4 软件维护	47
2.4.5 软件使用	47
2.4.6 软件项目管理	48
2.5 本章小结	49
2.6 习题	49
2.7 答案	50
第二部分 统一建模语言 (UML)	51
第 3 章 类、属性和操作的基本表示法	56
3.1 类	56
3.2 属性	57
3.3 操作	58
3.4 重载操作	61
3.5 属性和操作的可视性	62
3.6 类属性和类操作	63
3.7 抽象操作和类	63
3.8 实用程序	65
3.9 参数化类	66
3.10 本章小结	67
3.11 习题	68
3.12 答案	68
第 4 章 类图	72
4.1 一般化结构	72
4.1.1 单一继承	72
4.1.2 多重继承	74
4.1.3 子类的划分	74
4.1.4 分类识别名称	76
4.2 关联结构	77
4.2.1 关联的基本 UML 表示法	78

4.2.2 关联类	80
4.2.3 多向关联	81
4.2.4 关联的导航	82
4.3 整体/部分关联	83
4.3.1 组成	83
4.3.2 聚集	85
4.4 本章小结	88
4.5 习题	89
4.6 答案	90
第 5 章 对象交互图	94
5.1 协同图	94
5.1.1 描述一个消息	95
5.1.2 协同图中的多态性	97
5.1.3 重复的消息	98
5.1.4 在消息中使用 self	98
5.2 时序图	100
5.3 异步消息及并发执行	102
5.3.1 描述一个异步消息	103
5.3.2 回调机制	104
5.3.3 异步消息的优先权	107
5.3.4 描述一个广播（无目标）消息	108
5.4 本章小结	109
5.5 习题	110
5.6 答案	110
第 6 章 状态图	112
6.1 基本状态图	112
6.2 嵌套状态	114
6.3 并发状态和同步	117
6.4 消息结果参数的瞬时状态	121
6.5 连续的、可变的属性	122
6.6 本章小结	124
6.7 习题	125
6.8 答案	126
第 7 章 体系结构和接口图	130
7.1 描述系统体系结构	130
7.1.1 包	130

7.1.2 硬件环境的配置图	132
7.1.3 软件结构的配置图	133
7.2 人机界面的描述	135
7.2.1 窗口布局图	136
7.2.2 窗口导航图	137
7.2.3 题外话：什么是面向对象的 GUI	139
7.3 本章小结	139
7.4 习题	140
7.5 答案	140

第三部分 面向对象设计的原则 143

第 8 章 封装与共生性（connascence） 144

8.1 封装结构	144
8.1.1 封装的级别	144
8.1.2 制定标准管理各个封装级别之间的相互作用	146
8.2 共生性（connascence）	147
8.2.1 共生性的种类	147
8.2.2 差异共生性（contranascence）	150
8.2.3 共生性和封装边界	151
8.2.4 共生性和可维护性	152
8.2.5 面向对象系统中共生性的滥用	154
8.2.6 共生性（connascence）的术语	156
8.3 本章小结	156
8.4 习题	157
8.5 答案	158

第 9 章 领域、依附集和内聚 160

9.1 对象类的领域	160
9.1.1 基础领域	161
9.1.2 结构领域	161
9.1.3 商业领域	162
9.1.4 应用领域	163
9.1.5 每一领域类的来源	163
9.2 依附集（encumbrance）	165
9.2.1 什么是依附集（encumbrance）	165
9.2.2 依附集的使用	167
9.2.3 Demeter 规则	168

9.3 类的内聚 (Class Cohesion): 一个类和它的特性	169
9.3.1 混合事例型内聚	170
9.3.2 混合领域型内聚	171
9.3.3 混合角色型内聚	172
9.4 本章小结	173
9.5 习题	174
9.6 答案	175
第 10 章 状态空间和行为	178
10.1 一个类的状态空间和行为	178
10.2 子类的状态空间	181
10.3 子类的行为	183
10.4 状态空间的一个约束条件: 类的不变式	184
10.5 前置条件和后置条件	186
10.6 本章小结	188
10.7 习题	189
10.8 答案	189
第 11 章 类型一致性与闭合行为	192
11.1 类与类型	192
11.2 类型一致性原则	194
11.2.1 抗变性与协变性原则	195
11.2.2 抗变性与协变性实例	195
11.2.3 图解抗变性与协变性	199
11.2.4 类型一致性必要条件综述	200
11.3 闭合行为 (Closed Behavior) 原则	201
11.4 本章小结	202
11.5 习题	203
11.6 答案	204
第 12 章 继承与多态性的危险性	207
12.1 继承的滥用	207
12.1.1 错误的聚集	207
12.1.2 倒置的层次结构	208
12.1.3 混淆类及其实例	209
12.1.4 误用	212
12.2 多态性的危险性	215
12.2.1 操作的多态性	215
12.2.2 变量的多态性	217