

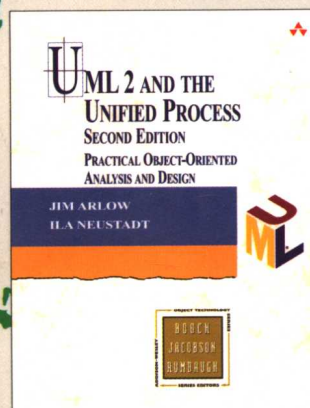
UML 2 and the Unified Process :

Practical Object-Oriented Analysis and Design, Second Edition

[美] Jim Arlow Ila Neustadt 著

UML 2 和统一过程：

实用面向对象的分析与设计
(第 2 版) (英文版)



典藏原版书苑

UML 2 和统一过程： 实用面向对象的分析与设计 (第2版)(英文版)

UML 2 and the Unified Process: Practical Object-Oriented
Analysis and Design (Second Edition)

江苏工业学院图书馆
藏书章

[美]

Jim Arlow
Ila Neustadt

著

人民邮电出版社

图书在版编目 (CIP) 数据

UML 2 和统一过程: 实用面向对象的分析与设计: 第 2 版/ (美) 阿洛 (Arlow, J.), (美) 诺伊施塔特 (Neustadt, I.) 著. —北京: 人民邮电出版社, 2006.7
(典藏原版书苑)

ISBN 7-115-14957-7

I. U... II. ①阿... ②诺... III. 面向对象语言, UML—程序设计—英文 IV. TP312
中国版本图书馆 CIP 数据核字 (2006) 第 071837 号

版 权 声 明

Original edition, entitled UML 2 AND THE UNIFIED PROCESS: PRACTICAL OBJECT-ORIENTED ANALYSIS AND DESIGN, 2nd Edition, 0321321278 by ARLOW, JIM; NEUSTADT, ILA, published by Pearson Education, Inc, publishing as Addison Wesley Professional, Copyright © 2005 Pearson Education, Inc. All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from Pearson Education, Inc.

China edition published by PEARSON EDUCATION ASIA LTD., and POSTS & TELECOMMUNICATIONS PRESS Copyright © 2006.

This edition is manufactured in the People's Republic of China, and is authorized for sale only in People's Republic of China excluding Hong Kong, Macau and Taiwan.

仅限于中华人民共和国境内 (不包括中国香港、澳门特别行政区和中国台湾地区) 销售。

本书封面贴有 Pearson Education (培生教育出版集团) 激光防伪标签。无标签者不得销售。

典藏原版书苑

UML 2 和统一过程: 实用面向对象的分析与设计 (第 2 版) (英文版)

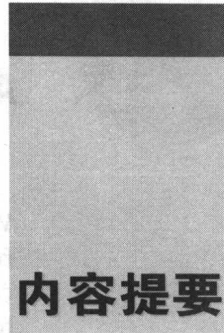
-
- ◆ 著 [美] Jim Arlow Ila Neustadt
责任编辑 李 际
 - ◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街 14 号
邮编 100061 电子函件 315@ptpress.com.cn
网址 <http://www.ptpress.com.cn>
北京顺义振华印刷厂印刷
新华书店总店北京发行所经销
 - ◆ 开本: 800×1000 1/16
印张: 38
字数: 817 千字 2006 年 7 月第 1 版
印数: 1—3 000 册 2006 年 7 月北京第 1 次印刷

著作权合同登记号 图字: 01-2006-3198 号

ISBN 7-115-14957-7/TP · 5529

定价: 65.00 元

读者服务热线: (010)67132705 印装质量热线: (010)67129223



内容提要

本书是使用 UML（统一建模语言）进行 OO（面向对象）分析与设计的绝佳指南。

本书在第 1 版的基础上针对 UML 2.0 进行了全面更新。本书重点在蓝图 UML，即使用正式、精确的 UML 模型详细规定软件系统。本书不仅详细描述了如何应用 UML 和统一过程进行面向对象分析与设计，还结合实例讨论了大量可以立即应用的实践技术。

本书内容丰富，结构合理，适合面向对象技术分析人员、设计人员、软件工程技术人員阅读，同时，也非常适合作为 UML 课程的教材。



Acknowledgments

We would like to thank Fabrizio Ferrandina, Wolfgang Emmerich, and our friends at Zühlke Engineering for encouraging us to create the UML training course that led to this book. Special thanks to Roland Leibundgut of Zühlke for his comments on the use case chapters and to Jos Warmer and Tom Van-Court for their insightful comments on the OCL chapter. Thanks are also due to our other technical reviewers, Glen Ford, Birger Møller-Pedersen, Rob Pettit, Gary Pollice, Ishan De Silva, and Fred Waskiewicz. Thanks to Sue and David Epstein for essential nontechnical support throughout the course of the project. Thanks to Andy Pols for sharing his thoughts on use cases and software engineering with us. Thanks to Lara Wyson, Mary Lou Nohr, and Kim Arney Mulcahy at Addison-Wesley for their great work on the text, and to our editor, Mary O'Brien. Thanks to the Neustadt family for their patience and to Al Toms for light relief. And thanks to our cats, Homer, Paddy, and Meg, for the many hours they spent sleeping on the various drafts of the manuscript, thereby imbuing it with that "quality without a name".

Finally, we must acknowledge the "Three Amigos"—Grady Booch, Jim Rumbaugh, and Ivar Jacobson—for their fine work on UML and UP that this book is all about.

关于本书

本书的目标是帮助读者使用统一建模语言 (Unified Modeling Language, UML) 和统一过程 (Unified Process, UP) 学习面向对象 (Object-Oriented, OO) 分析与设计。

UML 为 OO 建模提供了可视化建模语言; UP 提供了软件工程过程框架, 告诉我们如何执行 OO 分析与设计。

UP 的内容丰富, 本书仅涉及与 OO 分析人员和设计人员工作直接相关的内容, 有关 UP 其他方面的细节, 请参考[Rumbaugh 1]以及本书参考文献中的其他 UP 书籍。

本书充分介绍 UML 知识以及相关的分析与设计技术, 以便读者能够将这些建模知识有效地应用于实际项目。根据 Stephen J Mellor 的观点[Mellor 1], 目前主要有 3 种 UML 建模方式:

- 草图 UML——这是 UML 非正式用法。在这种用法中, 会勾画出一些草图来形象地描述一个软件系统, 这有点像在餐巾纸背面勾画某种构思。这些草图除了最初的作用之外, 几乎没有其他价值, 它不会被维护, 最终会被抛弃。通常, 可以采用白板或者诸如 Visio 和 PowerPoint 这样的画图工具来创建非正式草图。

- 蓝图 UML——这是 UML 更加正式和精确的用法。在这种用法中, UML 被适时维护, 并且成为项目的一个重要交付物。这种用法需要使用诸如 Rational Rose 或者 MagicDraw UML 这样真正的建模工具。

- 可执行 UML——使用模型驱动构架 (Model Driven Architecture, MDA), UML 模型可被用作编程语言。为 UML 模型添加足够的细节, 以便从模型中编译生成系统, 这是 UML 最正式和精确的用法。我们认为这是软件开发的未来。这种用法需要诸如 ArcStyler 这样支持 MDA 的 UML 工具。MDA 不在本书详细讨论范围之内, 但是我们会在 1.4 节中简要讨论。

本书的重点在于蓝图 UML。本书涉及的技术同样可以运用于可执行 UML。在学习了蓝图 UML 后，读者将能够自然而然地使用 UML 勾画所需要的草图。我们会尽可能以直接且易于接受的方式来展示 UML 和 UP。

约定

为了帮助浏览全书，我们在每章都以 UML 活动图的形式提供了该章的概览，这些图表示阅读活动以及阅读顺序。我们将在第 14 章详细讨论活动图。图 1 将帮助读者理解章节概览图的作用。

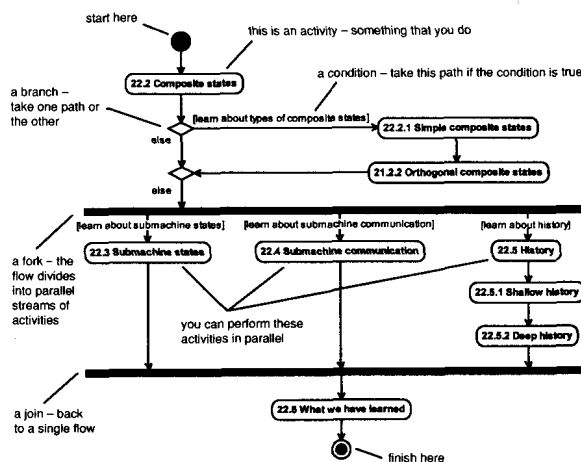


图 1

本书中大多数图是 UML 图，有些灰色的注释性文字并不是 UML 语法的组成部分。

Notes indicate
important
information.

我们在页边提供了注释，指出重要的信息。我们使用的是 UML 注视图标，如页边所示。

下面内容在书中使用不同的字体：

This font is for UML modeling elements.
This font is for code.

这种字体表示 UML 建模元素。
这种字体用于代码。

本书读者对象

本书针对以下读者：

- 需要学习 OO 分析与设计的分析人员/设计人员；

- 需要学习 UP 框架内的 OO 分析与设计技术的分析人员/设计人员；
- 学习 UML 课程的学生；
- 需要 UML 参考书的软件工程师；
- 正在参加 UML 培训课程的软件工程师，本书就是培训教材。

如何阅读本书

可读的书太多，而可用来读书的时间又太少，考虑到这一矛盾，我们将本书设计为可以根据需要以几种不同的方式阅读。

快速浏览

如果想概览整本书或只想阅读某一章，请选择快速浏览方式。

- 选择一章。
- 阅读章节概览图以便了解该阅读哪里。
- 通过看图和阅读页边注释浏览该章。
- 阅读每章“**What we have learned**”部分。
- 回溯到任何你感兴趣的部分并阅读它。

快速浏览是阅读本书快速和高效的方法。如果读者一开始就明确地知道自己想要获得什么信息，快速阅读方式效果最好。

参考

如果需要了解 UML 的特定部分或者想学习特定的技术，我们提供了详细的索引和目录，这应该能够帮助你快速高效地找到所需的信息。本书提供了充分的索引功能。

复习

复习本书有两种策略：

- 如果想尽可能快速、高效地更新自己的 UML 知识，阅读每章的“**What we have learned**”部分，当你不了解某些内容时，回溯并阅读相应章节；
- 如果有更多时间，可以浏览每章，看图并阅读页边注释。

翻阅

如果你有几分钟空余时间，可以拿起本书，随机翻阅。我们尽量保证每一页都有让你感兴趣的东西。即便你已经非常了解 UML，你仍可以发现值得学习的新东西。

章节概览图

图 2 是本书的章节概览图，指出了可以采用的阅读顺序，以及在第一遍阅读时可以选择跳过的高级技术。

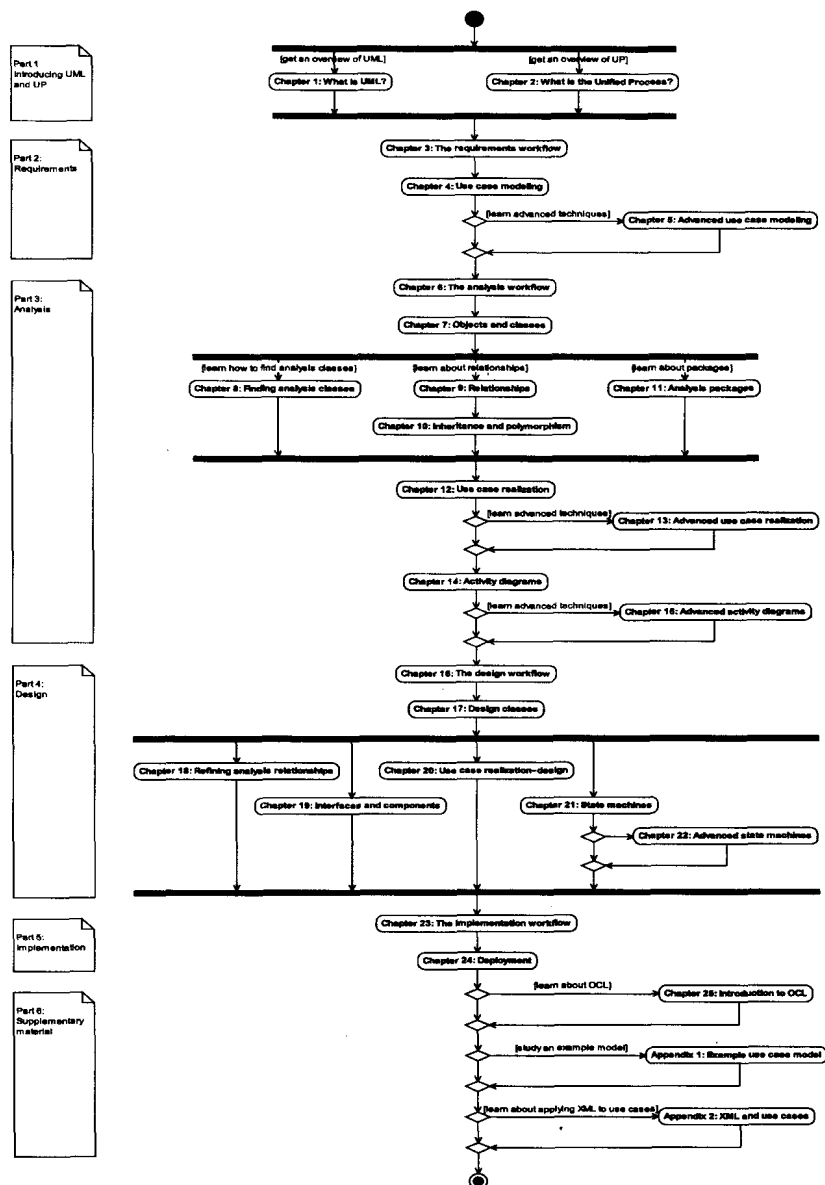
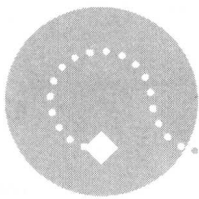


Figure 2

图 2



Contents

Part 1

Introducing UML and UP

1 What is UML?

1.1	Chapter roadmap	3
1.2	What is UML?	5
1.3	The birth of UML	5
1.4	MDA – the future of UML	7
1.5	Why “unified”?	9
1.6	Objects and UML	10
1.7	UML structure	10
1.8	UML building blocks	11
1.9	UML common mechanisms	15
1.10	Architecture	23
1.11	What we have learned	24

2 What is the Unified Process?

2.1	Chapter roadmap	27
2.2	What is UP?	28
2.3	The birth of UP	29
2.4	UP and the Rational Unified Process	32
2.5	Instantiating UP for your project	34
2.6	UP axioms	34
2.7	UP is an iterative and incremental process	35
2.8	UP structure	37
2.9	UP phases	39
2.10	What we have learned	44

Part 2**Requirements**

47

3 The requirements workflow

49

- 3.1 Chapter roadmap 49
- 3.2 The requirements workflow 51
- 3.3 Software requirements – metamodel 52
- 3.4 Requirements workflow detail 53
- 3.5 The importance of requirements 55
- 3.6 Defining requirements 55
- 3.7 Finding requirements 61
- 3.8 What we have learned 65

4 Use case modeling

67

- 4.1 Chapter roadmap 67
- 4.2 Use case modeling 69
- 4.3 UP activity: Find actors and use cases 69
- 4.4 UP activity: Detail a use case 77
- 4.5 Use case specification 78
- 4.6 Requirements tracing 90
- 4.7 When to apply use case modeling 91
- 4.8 What we have learned 92

5 Advanced use case modeling

95

- 5.1 Chapter roadmap 95
- 5.2 Actor generalization 97
- 5.3 Use case generalization 99
- 5.4 «include» 102
- 5.5 «extend» 105
- 5.6 When to use advanced features 110
- 5.7 Hints and tips for writing use cases 111
- 5.8 What we have learned 113

Part 3**Analysis**

117

6 The analysis workflow

119

- 6.1 Chapter roadmap 119
- 6.2 The analysis workflow 120

6.3	Analysis artifacts – metamodel	121
6.4	Analysis workflow detail	122
6.5	Analysis model – rules of thumb	122
6.6	What we have learned	124
7	Objects and classes	125
7.1	Chapter roadmap	125
7.2	What are objects?	127
7.3	UML object notation	131
7.4	What are classes?	132
7.5	UML class notation	136
7.6	Scope	147
7.7	Object construction and destruction	148
7.8	What we have learned	151
8	Finding analysis classes	155
8.1	Chapter roadmap	155
8.2	UP activity: Analyze a use case	157
8.3	What are analysis classes?	158
8.4	Finding classes	163
8.5	Creating a first-cut analysis model	171
8.6	What we have learned	172
9	Relationships	175
9.1	Chapter roadmap	175
9.2	What is a relationship?	177
9.3	What is a link?	177
9.4	What is an association?	180
9.5	What is a dependency?	195
9.6	What we have learned	201
10	Inheritance and polymorphism	205
10.1	Chapter roadmap	205
10.2	Generalization	206
10.3	Class inheritance	208
10.4	Polymorphism	211
10.5	Advanced generalization	215
10.6	What we have learned	221

11	Analysis packages	223
11.1	Chapter roadmap	223
11.2	What is a package?	224
11.3	Packages and namespaces	226
11.4	Nested packages	227
11.5	Package dependencies	228
11.6	Package generalization	231
11.7	Architectural analysis	231
11.8	What we have learned	235
12	Use case realization	239
12.1	Chapter roadmap	239
12.2	UP activity: Analyze a use case	241
12.3	What are use case realizations?	242
12.4	Use case realization – elements	243
12.5	Interactions	244
12.6	Lifelines	244
12.7	Messages	246
12.8	Interaction diagrams	248
12.9	Sequence diagrams	249
12.10	Combined fragments and operators	256
12.11	Communication diagrams	264
12.12	What we have learned	268
13	Advanced use case realization	273
13.1	Chapter roadmap	273
13.2	Interaction occurrences	274
13.3	Continuations	279
13.4	What we have learned	281
14	Activity diagrams	283
14.1	Chapter roadmap	283
14.2	What are activity diagrams?	284
14.3	Activity diagrams and the UP	285
14.4	Activities	286
14.5	Activity semantics	288
14.6	Activity partitions	290
14.7	Action nodes	293

14.8	Control nodes	297
14.9	Object nodes	301
14.10	Pins	305
14.11	What we have learned	307

15 Advanced activity diagrams 309

15.1	Chapter roadmap	309
15.2	Connectors	311
15.3	Interruptible activity regions	311
15.4	Exception handling	312
15.5	Expansion nodes	313
15.6	Sending signals and accepting events	314
15.7	Streaming	317
15.8	Advanced object flow features	318
15.9	Multicast and multireceive	320
15.10	Parameter sets	321
15.11	«centralBuffer» node	322
15.12	Interaction overview diagrams	323
15.13	What we have learned	325

Part 4

Design 329

16 The design workflow 331

16.1	Chapter roadmap	331
16.2	The design workflow	332
16.3	Design artifacts – metamodel	333
16.4	Design workflow detail	337
16.5	UP activity: Architectural design	338
16.6	What we have learned	339

17 Design classes 341

17.1	Chapter roadmap	341
17.2	UP activity: Design a class	342
17.3	What are design classes?	344
17.4	Anatomy of a design class	345
17.5	Well-formed design classes	347
17.6	Inheritance	350
17.7	Templates	354

17.8	Nested classes	357
17.9	What we have learned	358
18	Refining analysis relationships	361
18.1	Chapter roadmap	361
18.2	Design relationships	363
18.3	Aggregation and composition	363
18.4	Aggregation semantics	364
18.5	Composition semantics	367
18.6	How to refine analysis relationships	368
18.7	One-to-one associations	369
18.8	Many-to-one associations	370
18.9	One-to-many associations	371
18.10	Collections	371
18.11	Reified relationships	375
18.12	Exploring composition with structured classes	378
18.13	What we have learned	382
19	Interfaces and components	387
19.1	Chapter roadmap	387
19.2	UP activity: Design a subsystem	389
19.3	What is an interface?	389
19.4	Provided and required interfaces	391
19.5	Interface realization vs. inheritance	394
19.6	Ports	398
19.7	Interfaces and component-based development	399
19.8	What is a component?	399
19.9	Component stereotypes	402
19.10	Subsystems	403
19.11	Finding interfaces	404
19.12	Designing with interfaces	404
19.13	Advantages and disadvantages of interfaces	408
19.14	What we have learned	408
20	Use case realization–design	413
20.1	Chapter roadmap	413
20.2	UP activity: Design a use case	415
20.3	Use case realization–design	416
20.4	Interaction diagrams in design	417

20.5	Modeling concurrency	419
20.6	Subsystem interactions	426
20.7	Timing diagrams	427
20.8	Example of use case realization–design	430
20.9	What we have learned	436

21 State machines 437

21.1	Chapter roadmap	437
21.2	State machines	439
21.3	State machines and the UP	441
21.4	State machine diagrams	442
21.5	States	443
21.6	Transitions	445
21.7	Events	448
21.8	What we have learned	453

22 Advanced state machines 457

22.1	Chapter roadmap	457
22.2	Composite states	458
22.3	Submachine states	465
22.4	Submachine communication	466
22.5	History	468
22.6	What we have learned	470

Part 5 **Implementation** 473

23 The implementation workflow 475

23.1	Chapter roadmap	475
23.2	The implementation workflow	476
23.3	Implementation artifacts – metamodel	477
23.4	Implementation workflow detail	478
23.5	Artifacts	479
23.6	What we have learned	480

24 Deployment 481

24.1	Chapter roadmap	481
24.2	UP activity: Architectural implementation	482
24.3	The deployment diagram	483

24.4	Nodes	484
24.5	Artifacts	486
24.6	Deployment	490
24.7	What we have learned	491

Part 6

Supplementary material

493

25 Introduction to OCL

25.1	Chapter roadmap	495
25.2	What is the Object Constraint Language (OCL)?	497
25.3	Why use OCL?	497
25.4	OCL expression syntax	498
25.5	Package context and pathnames	500
25.6	The expression context	501
25.7	Types of OCL expressions	502
25.8	The expression body	504
25.9	OCL navigation	522
25.10	Types of OCL expression in detail	526
25.11	OCL in other types of diagrams	535
25.12	Advanced topics	540
25.13	What we have learned	546

<i>Appendix 1: Example use case model</i>	551
---	-----

<i>Appendix 2: XML and use cases</i>	559
--------------------------------------	-----

<i>Bibliography</i>	567
---------------------	-----

<i>Index</i>	569
--------------	-----