

面向对象

分析

OOA

和设计

宛延闾 定海 编著

OOD



清华大学出版社

<http://www.tup.tsinghua.edu.cn>



面向对象分析和设计

宛延阁 定海 编著

清华大学出版社

(京)新登字 158 号

内 容 简 介

本书详尽地描述了面向对象概念集和与编程无关的图形表示法——对象模型技术(OMT)。OMT表示法适用于从分析问题需求,到设计这个问题的解,然后用编程语言或数据库实现这个解的全过程。

本书共分4个部分21章。第一部分介绍基本的与编程无关的面向对象高层概念和OMT技术表示法,并将OMT贯穿于全书所有例子之中;第二部分循序渐进地描述软件开发的面向对象方法学;第三部分描述在不同开发环境中面向对象设计的实现,包括面向对象语言、非面向对象语言和关系数据库;第四部分介绍了几个典型的实例和实践技巧。这些实例对面向对象方法在各个领域中的应用和开发具有现实意义。书中每章都有练习并在书后对较难的练习作了提示性解答。

本书可作为计算机专业本科生和研究生的软件工程和面向对象技术课程的教材,或作为数据库、程序设计语言的补充教材,也可作为系统分析和设计人员、软件开发人员以及面向对象程序设计人员必备的参考书。

版权所有,翻印必究。

本书封面贴有清华大学出版社激光防伪标签,无标签者不得销售。

书 名: 面向对象分析和设计

作 者: 宛延闾 定海 编著

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.tsinghua.edu.cn>

印刷者: 北京市密云胶印厂

发行者: 新华书店总店北京发行所

开 本: 787×1092 1/16 印张: 29.5 字数: 678 千字

版 次: 2001 年 2 月第 1 版 2001 年 2 月第 1 次印刷

书 号: ISBN 7-302-01271-7/TP·479

印 数: 0001~8000

定 价: 39.00 元

序 言

本书所介绍的面向对象方法,是对现实世界的对象模型的一种软件开发。面向对象方法能够把现实世界中的对象,经过有效合理地组织,创建成一种与编程语言无关的面向对象模型。这种模型无论是对问题的透彻理解,还是对设计的清晰性,以及更好的维护系统都有极大地促进作用。

本书描述了面向对象概念集以及与编程无关的图形表示法——对象模型技术(Object Modeling Technique,简称 OMT)表示法。OMT 能够适用于从分析问题的需求,到设计这个问题的解,然后用编程语言或数据库实现这个解的全过程。该方法可以应用于整个软件开发过程,软件开发者在每个开发阶段上不需要再采用其他方法,更不需要在软件开发阶段中转换成另一种新的表示法。

在介绍对象模型技术的同时,本书也展示了如何将面向对象概念贯穿于整个软件开发生命周期——从分析到设计再到实现。我们不仅仅描述基本面向对象语言或代码,而且更着重强调了代码是经过包括问题叙述开始的需求理解,构造问题解和用特定语言实现一种程序开发过程的最后产物。一个好的设计技术必定将实现细节推迟,直到设计的最后阶段,以确保系统的灵活性。

面向对象技术是大大超越程序设计方式的一种技术,是把一组相互作用对象中的数据和行为紧密结合在一起的组织系统的一种策略。该技术是对现实世界概念的一种抽象思维方式,而不是计算机的概念,因此大多数用惯计算机概念的人的思想需要有不同程度的转变。过去的程序设计语言强迫人们根据计算机概念,而不是根据实际应用来思考问题。本书自始至终强调面向对象程序设计能够帮助程序员学习抽象思维,而不是用程序设计结构来思维。书中展示的对象模型技术的图形表示法可以帮助软件开发人员提高软件产品化开发能力,与最终的实现编程语言无关。

我们希望通过生动的面向对象技术的刻画,以重实效的方法来解决绝大多数应用领域的问题,并提供生动的、优秀的表示法和方法学,同时也提供一些技巧、好的和坏的设计,使软件开发人员吸取经验和教训,避免再次出现错误。同时,我们用面向对象 OMT 技术描述了几个真实的应用开发实例,以飨读者,并从中来全面理解面向对象概念的本质。

本书读者将学习到如何把面向对象概念应用于软件开发生命周期的所有阶段。现在这样的书很少,即使有的话,面向对象书籍也仅覆盖软件生命周期,而程序设计或分析却是单独撰写的。事实上,很少教科书在面向对象技术上是这种类型的。虽然面向对象技术是当前的“热门”话题,但大多数读者在这方面经验很少,所以我们不假定读者有面向对象方面的知识,我们只假定读者有基本的计算机概念。每个试图采用面向对象技术的程

序员,将从学习如何系统设计程序中获益。他们或许会惊奇地在这一过程中发现或总结出某些共通的面向对象设计代码实践经验,以及好的设计原则。

面向对象哲理创建了一种把抽象、封装、模块紧密结合在一起的强有力的协同机制,并贯穿于软件开发的整个生命周期。对数据库设计者来说,面向对象模型技术将会导致更有趣的发现,虽然目前对面向对象程序设计语言给予了足够的关注,但数据库的面向对象设计更有其丰富实践的挑战性和及时性。本书将完整地描述面向对象设计如何实现现存的关系数据库系统以及它们的应用。

本书能够作为大学本科生和研究生的软件工程以及面向对象技术的教材。它可以作为数据库和程序设计语言的补充。学习本书必备的条件包括模型结构程序设计语言和基本计算机科学方面的知识和概念,诸如语法、语义、递归、集合、过程、图和状态,而严格正规的知识背景并不需要。

许多面向对象书籍在讨论程序设计问题时,通常从一个单一语言观点出发,似乎有点狭隘或者片面。正确的观点最好从设计问题出发,然而市面上的这些面向对象书籍还是仅仅讨论程序设计语言。至今,很少书籍撰写面向对象分析和设计。本书所展示的面向对象概念能够或应该适用于整个软件生命周期。近年来,已有一些面向对象方法学的书籍陆续出现,但本书是适用于面向对象方面的分析和设计的,并且在内容上更能很好地理解面向对象分析和设计。

在软件方法学上,过去已出版的一些书籍均以过程观点来讨论软件整个生命周期,虽然近年来也用面向对象概念加以修正,但本质上却没有太大的变化,表面上似乎基于对象,但实际上快速地转换和变换,又回到过程问题中去了。

面向对象程序设计某些方面是与建模和设计方法学一致的,我们强调现实世界建模的面向对象结构,而不是程序设计的技术。把内部对象关系推向一类相同的语义层,而不是作为指针隐含在对象的内部。我们还详细说明了继承机制,特别强调类型、类、模型和高层抽象机制,尽可能使用大家都能接受的术语以及其他所能选择的最好术语。这里,不存在所谓公共能够接受的面向对象构造的图形表示法,所以尽管集中几节作了“其他表示法”的介绍,但本书使用 OMT 表示法,因为这是一种成功的表示法。在这种情况下,面向对象概念本身是最重要的事情,而不是拘泥于表示它们的符号形状,书中也展示了如何把面向对象概念应用于有穷自动状态机的情况。

本书包含四个部分:第一部分介绍与编程语言无关的面向对象的高层概念,这些概念是基本的。OMT 技术表示法在第一部分介绍,并贯穿于书中全部例子。第二部分循序渐进描述软件开发的面向对象方法学,从问题的叙述出发,经过分析、系统设计和对象设计各个阶段。所有方法学的最后阶段都是与编程语言无关,甚至对象设计大多数都与具体的编程语言无关。第三部分描述在不同开发环境中面向对象设计的实现,包括面向对象语言、非面向对象语言和关系数据库,描述了在不同环境的可应用性。虽然对面向对象程序设计作了详尽的描述,但本书并非替代面向对象程序设计。第四部分介绍了典型的应用实例开发,这些实例既覆盖了应用领域和实现目标,又将面向对象概念的本质展示无疑。读者将会发现与图形表示法和开发方法学一起使用面向对象概念,可以极大地提高软件的质量、灵活性和可理解性。

为了加深对每章的概念以及对象模型、动态模型和功能模型的理解,每章后面附有一定量的练习。本书最后对一些较难的习题作了提示性解答。

本书编写过程,自始至终得到了国内有关专家的关心和支持,也得到同行的热忱鼓励,在此一并表示感谢!

鉴于作者水平有限,不妥之处在所难免,欢迎同行和读者批评指正。

我们希望本书能够对 21 世纪我国软件产品广泛采用面向对象技术开发和应用,起抛砖引玉的作用。

宛延阁 定海

二〇〇〇年九月于北京



目 录

序言

第 1 章 引言	(1)
1.1 面向对象的概念	(1)
1.1.1 对象的特性	(2)
1.2 面向对象开发的概念	(4)
1.2.1 面向对象的思维	(4)
1.2.2 面向对象方法学	(4)
1.2.3 三种模型	(6)
1.2.4 功能方法学上的差异	(6)
1.3 面向对象技术要点	(6)
1.3.1 抽象	(7)
1.3.2 封装	(7)
1.3.3 数据和行为的联合	(8)
1.3.4 共享	(8)
1.3.5 重点在对象结构,不是在过程结构	(8)
1.3.6 协同作用	(9)
1.4 面向对象开发的可用性例证	(9)
1.5 面向对象方法的要点和主要优点	(10)
1.6 本书的组织	(11)
练习	(12)

第一部分 建模概念

第 2 章 一种设计技术的建模	(17)
2.1 建模	(17)
2.1.1 使用模型的目的	(17)
2.1.2 抽象	(18)
2.2 对象模型技术	(18)

2.2.1 对象模型	(19)
2.2.2 动态模型	(19)
2.2.3 功能模型	(19)
2.2.4 三种模型的联系	(20)
2.3 小结	(20)
练习	(20)
第3章 对象模型	(23)
3.1 对象和类	(23)
3.1.1 对象	(23)
3.1.2 类	(24)
3.1.3 对象图	(24)
3.1.4 属性	(25)
3.1.5 操作和方法	(26)
3.1.6 对象类的表示小结	(27)
3.1.7 对象和类的 OMT 表示的改进	(27)
3.1.8 改进的对象和类的表示小结	(29)
3.2 链接和关联	(30)
3.2.1 一般概念	(30)
3.2.2 重数	(32)
3.2.3 关联的重要性	(33)
3.3 高级链接和关联概念	(33)
3.3.1 链接属性	(33)
3.3.2 用关联模型化为类	(35)
3.3.3 角色名	(35)
3.3.4 排序	(36)
3.3.5 资格符	(36)
3.3.6 聚合	(37)
3.4 概括和继承	(38)
3.4.1 一般概念	(38)
3.4.2 概括的使用	(40)
3.4.3 重写特征	(40)
3.5 构造分组	(40)
3.5.1 模块	(40)
3.5.2 表	(41)
3.6 对象模型的一个实例	(41)
3.7 OMT 对象模型在链接和关联方面的改进	(43)
3.8 实践的技巧	(45)

3.9 小结	(46)
练习	(47)
第4章 高级对象模型	(55)
4.1 对象和类的概念	(55)
4.1.1 例化	(55)
4.1.2 类属性和操作	(55)
4.1.3 属性的重数	(56)
4.1.4 类的候选关键字	(56)
4.1.5 域	(57)
4.1.6 数据的辅助特征	(58)
4.2 链接和关联的概念	(59)
4.2.1 重数	(59)
4.2.2 三元关联	(60)
4.2.3 关联的候选关键字	(60)
4.2.4 异或关联	(61)
4.2.5 资格关联	(61)
4.3 聚合	(62)
4.3.1 聚合与关联	(62)
4.3.2 聚合与概括	(63)
4.3.3 递归聚合	(63)
4.3.4 操作的传播	(64)
4.3.5 物理聚合与分类聚合	(65)
4.3.6 物理聚合的语义扩展	(66)
4.3.7 分类聚合的语义扩展	(66)
4.4 概括	(67)
4.4.1 抽象类和具体类	(67)
4.4.2 概括与其他对象建模结构	(67)
4.5 多重继承	(69)
4.5.1 有不同鉴别器的多重继承	(69)
4.5.2 无公共祖先的多重继承	(69)
4.5.3 多重继承的工作环境	(70)
4.6 包	(72)
4.6.1 水平逻辑	(73)
4.6.2 包的实例	(74)
4.7 导出数据和约束	(76)
4.8 对象元模型	(78)
4.8.1 元数据和元模型	(78)

4.8.2	框架	(82)
4.8.3	模式	(83)
4.9	高级实践技巧	(87)
4.10	小结	(88)
	练习	(89)
第5章	动态模型	(95)
5.1	事件和状态	(95)
5.1.1	事件	(96)
5.1.2	脚本和事件轨迹	(97)
5.1.3	状态	(98)
5.1.4	状态图	(99)
5.1.5	条件	(100)
5.2	操作	(101)
5.2.1	控制操作	(101)
5.2.2	操作的状态图小结	(102)
5.3	嵌套状态图	(103)
5.3.1	平状态图问题	(103)
5.3.2	嵌套状态图	(103)
5.3.3	状态概括	(104)
5.3.4	事件概括	(105)
5.4	并发性	(106)
5.4.1	聚合并发性	(106)
5.4.2	对象内部并发性	(107)
5.5	高级动态模型概念	(107)
5.5.1	进入和退出动作	(107)
5.5.2	内部动作	(108)
5.5.3	自动变迁	(109)
5.5.4	发送事件	(109)
5.5.5	并发活动的同步	(110)
5.6	动态模型的实例	(111)
5.7	对象模型和动态模型的关系	(115)
5.8	实践技巧	(116)
5.9	小结	(116)
	练习	(117)
第6章	功能模型	(123)
6.1	功能模型	(123)



6.2 数据流图	(124)
6.2.1 处理	(125)
6.2.2 数据流	(125)
6.2.3 施动者	(126)
6.2.4 数据存储	(126)
6.2.5 嵌套数据流图	(128)
6.2.6 控制流	(128)
6.3 指定的操作	(129)
6.4 约束	(131)
6.5 数据库应用中的功能模型	(131)
6.5.1 伪码	(132)
6.5.2 ONN 的伪码	(133)
6.5.3 ONN 的构造	(133)
6.5.4 组合 ONN 构造	(140)
6.5.5 添加 ONN 特性	(141)
6.5.6 其他范型	(143)
6.5.7 实践技巧	(145)
6.6 功能模型的实例(飞行模拟机装置)	(146)
6.7 功能模型与对象模型和动态模型的关系	(149)
6.8 小结	(150)
练习	(151)

第二部分 设计方法学

第7章 方法学简介	(157)
7.1 OMT 是一种软件工程方法学	(157)
7.2 OMT 方法学	(158)
7.3 面向对象方法的深远影响	(159)
7.4 小结	(159)
练习	(160)
第8章 分 析	(161)
8.1 分析综述	(161)
8.2 需求陈述	(162)
8.3 自动取款机例子	(163)
8.4 建立对象模型	(164)
8.4.1 找出对象类	(165)

8.4.2	筛选出正确的对象类	(166)
8.4.3	准备数据字典	(168)
8.4.4	确定关联	(169)
8.4.5	划分主题	(172)
8.4.6	确定属性	(173)
8.4.7	用继承性改进对象模型	(175)
8.4.8	反复修改对象模型	(176)
8.5	动态模型	(177)
8.5.1	编写脚本	(178)
8.5.2	用户界面	(179)
8.5.3	画事件轨迹图	(179)
8.5.4	画状态图	(180)
8.5.5	审查动态模型	(183)
8.6	功能模型	(184)
8.6.1	找出输入和输出值	(184)
8.6.2	建立数据流图	(184)
8.6.3	描述功能	(186)
8.6.4	找出对象之间的约束	(186)
8.6.5	指定优化规则	(187)
8.7	定义服务	(187)
8.7.1	常规行为	(187)
8.7.2	从事件导出的操作	(187)
8.7.3	与数据流图中处理框对应的操作	(187)
8.7.4	利用继承减少冗余操作	(188)
8.8	小结	(188)
	练习	(189)
第9章	系统设计	(197)
9.1	系统设计综述	(197)
9.2	将系统划分为子系统	(198)
9.2.1	分层	(199)
9.2.2	分块	(199)
9.2.3	系统拓扑	(200)
9.3	识别并发性	(200)
9.3.1	识别固有并发性	(200)
9.3.2	定义并发任务	(201)
9.4	给子系统分配处理器和任务	(201)
9.4.1	估计硬件资源需求	(201)

9.4.2 硬件与软件之间的折衷	(201)
9.4.3 为任务指定处理器	(202)
9.4.4 确定物理连接	(202)
9.5 数据存储管理	(203)
9.5.1 使用数据库的优点	(203)
9.5.2 使用数据库的缺点	(204)
9.6 处理全局资源	(204)
9.7 选择软件控制实现	(205)
9.7.1 过程驱动系统	(205)
9.7.2 事件驱动系统	(205)
9.7.3 并发系统	(206)
9.7.4 内部控制	(206)
9.7.5 其他范型	(206)
9.8 处理边界条件	(207)
9.9 设置折衷的优先权	(207)
9.10 公共的体系结构框架	(208)
9.10.1 批处理变换	(208)
9.10.2 连续变换	(209)
9.10.3 交互式接口	(210)
9.10.4 动态模拟	(211)
9.10.5 实时系统	(211)
9.10.6 事务处理管理	(211)
9.11 ATM 系统的结构	(212)
9.12 小结	(213)
练习	(214)
第 10 章 对象设计	(220)
10.1 对象设计综述	(220)
10.1.1 从分析和系统结构着手	(220)
10.1.2 对象设计的步骤	(221)
10.1.3 对象模型工具	(221)
10.2 组合三种模型	(222)
10.3 设计算法	(223)
10.3.1 选择算法	(223)
10.3.2 选择数据结构	(225)
10.3.3 定义内部类和操作	(225)
10.3.4 指定操作的职责	(225)
10.4 设计优化	(226)

10.4.1	添加冗余关联获取有效访问	(226)
10.4.2	重新安排执行次序以获得效率	(228)
10.4.3	保存导出属性避免重复计算	(228)
10.5	控制实现	(229)
10.5.1	在程序内进行状态设置	(229)
10.5.2	状态机器引擎	(231)
10.5.3	控制作为并发任务	(231)
10.6	继承的调整	(231)
10.6.1	重新安排类和操作	(231)
10.6.2	抽象出公共的行为	(232)
10.6.3	使用授权共享实现	(233)
10.7	关联设计	(234)
10.7.1	分析关联遍历	(234)
10.7.2	单向关联	(234)
10.7.3	双向关联	(235)
10.7.4	链接属性	(235)
10.8	对象的表示	(236)
10.9	物理打包	(236)
10.9.1	信息隐藏	(237)
10.9.2	实体的相关性	(237)
10.9.3	构造模块	(238)
10.10	设计决策文档	(238)
10.11	小结	(239)
	练习	(240)
第 11 章	方法学总结	(245)
11.1	分析	(245)
11.2	系统设计	(246)
11.3	对象设计	(247)
11.4	小结	(248)
	练习	(248)
第 12 章	方法学比较	(250)
12.1	结构化分析/结构化设计(SA/SD)	(250)
12.1.1	SA/SD 方法概述	(250)
12.1.2	与 OMT 方法学的比较	(251)
12.2	Jackson 结构化开发方法	(252)
12.2.1	JSD 方法概述	(252)

12.2.2 与 OMT 方法学的比较	(253)
12.3 信息建模的表示法	(254)
12.4 小结	(256)
练习	(256)

第三部分 实 现

第 13 章 从设计到实现	(261)
13.1 用程序设计语言实现	(261)
13.2 用数据库系统实现	(262)
13.3 计算机的外部实现	(263)
13.4 第三部分概述	(263)
第 14 章 程序设计风格	(264)
14.1 面向对象程序设计风格	(264)
14.2 可重用性	(264)
14.2.1 可重用的类型	(265)
14.2.2 可重用准则	(265)
14.2.3 使用继承机制	(266)
14.3 提高可扩充性	(267)
14.4 提高健壮性	(268)
14.5 小结	(268)
练习	(269)
第 15 章 面向对象语言	(273)
15.1 把设计转换成实现	(273)
15.2 类定义	(275)
15.2.1 C++ 中的类定义	(275)
15.2.2 Eiffel 中的类定义	(276)
15.2.3 Smalltalk 中的类定义	(277)
15.3 创建对象	(278)
15.3.1 C++ 中创建对象	(278)
15.3.2 Eiffel 中创建对象	(280)
15.3.3 Smalltalk 中创建对象	(280)
15.4 调用操作	(281)
15.4.1 C++ 中的调用操作	(281)
15.4.2 Eiffel 中的调用操作	(282)

面向对象分析和设计	
15.4.3	Smalltalk 中的调用操作 (283)
15.5	使用继承机制..... (284)
15.5.1	C++中使用继承..... (284)
15.5.2	Eiffel 中使用继承 (286)
15.5.3	Smalltalk 中使用继承 (287)
15.6	实现关联..... (288)
15.6.1	C++中实现的关联..... (289)
15.6.2	Eiffel 中实现的关联 (292)
15.6.3	Smalltalk 中实现的关联 (293)
15.7	面向对象语言的特性..... (294)
15.7.1	多重继承..... (295)
15.7.2	类库..... (295)
15.7.3	效率问题..... (295)
15.7.4	强类型与弱类型..... (296)
15.7.5	内存管理..... (296)
15.7.6	封装..... (297)
15.7.7	打包..... (298)
15.7.8	开发环境..... (298)
15.7.9	元数据..... (298)
15.7.10	参数化类 (299)
15.7.11	断言和约束 (299)
15.7.12	数据的持久性 (299)
15.8	面向对象语言的综述..... (300)
15.8.1	Smalltalk (300)
15.8.2	C++..... (301)
15.8.3	Eiffel (302)
15.8.4	CLOS (302)
15.8.5	面向对象数据库程序设计语言..... (303)
15.8.6	面向对象语言比较..... (304)
15.9	小结..... (305)
	练习..... (306)
第 16 章	非面向对象语言 (309)
16.1	映像面向对象的概念..... (309)
16.1.1	图形编辑器的例子..... (310)
16.1.2	在 C 语言中的实现 (310)
16.1.3	在 Ada 语言中的实现 (310)
16.1.4	在 Fortran 语言中的实现 (310)

16.1.5 其他语言	(310)
16.2 把类转换成数据结构	(311)
16.2.1 把类转换成 C 结构说明	(311)
16.2.2 把类转换成 Ada 记录定义	(311)
16.2.3 把类转换成 Fortran 的数组	(312)
16.3 传送参数到方法	(313)
16.3.1 在 C 中传送参数	(313)
16.3.2 在 Ada 中传送参数	(313)
16.3.3 在 Fortran 中传送参数	(313)
16.4 分配对象	(314)
16.4.1 在 C 中分配对象	(314)
16.4.2 在 Ada 中分配对象	(315)
16.4.3 在 Fortran 中分配对象	(315)
16.5 实现继承	(316)
16.5.1 在 C 中实现继承	(316)
16.5.2 在 Ada 中实现继承	(317)
16.5.3 在 Fortran 中实现继承	(318)
16.6 实现方法分解	(319)
16.6.1 在 C 中的方法分解	(320)
16.6.2 在 Ada 中的方法分解	(323)
16.6.3 在 Fortran 中的方法分解	(323)
16.7 实现关联	(324)
16.7.1 在 C 中实现关联	(325)
16.7.2 在 Ada 中实现关联	(326)
16.7.3 在 Fortran 中实现关联	(326)
16.8 处理并发	(327)
16.9 封装	(328)
16.9.1 在 C 中的封装	(328)
16.9.2 在 Ada 中的封装	(328)
16.9.3 在 Fortran 中的封装	(330)
16.10 你得到了什么	(330)
16.11 小结	(330)
练习	(331)
第 17 章 关系数据库	(333)
17.1 一般的 DBMS 概念	(333)
17.2 关系 DBMS 概念	(335)
17.2.1 RDBMS 逻辑数据结构	(335)