

“十一五”国家重点图书 计算机科学与技术学科前沿丛书
计算机科学与技术学科研究生系列教材（中文版）

面向对象的系统设计

（第2版）

邵维忠 杨芙清 著



清华大学出版社



“十一五”国家重点图书 计算机科学与技术学科

计算机科学与技术学科研究生系列教材

面向对象的系统设计

(第2版)

邵维忠 杨芙清 著

清华大学出版社

北京

内 容 简 介

本书是一本论述面向对象设计方法的专著,其第1版于2003年由清华大学出版社出版,被国内许多大学用作研究生或高年级本科生教材,并被许多软件开发单位作为工程技术用书。本次再版,根据国内外面向对象领域理论与技术的最新发展做了不少修改。

本书是作者的另一本著作《面向对象的系统分析》(第2版)的姊妹篇,二者构成完整的OOA&D方法体系。

本书的主要内容是论述如何在面向对象的分析(OOA)基础上进行面向对象的设计(OOD)。全书分为7章,第1章介绍OOD的发展历史、现状和几种典型的OOA&D方法,论述OOA和OOD的关系。第2章介绍本书提出的OOD方法概貌。第3~6章分别介绍OOD模型各个组成部分的设计方法。第7章介绍统一建模语言(UML),并分析和讨论其优点与缺点。

读者对象:计算机软件专业的教师、研究生和本科生,软件技术培训教师与学员,计算机软件领域的研究人员和工程技术人员。

关键词:软件工程,面向对象的系统设计,UML

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13501256678 13801310933

图书在版编目(CIP)数据

面向对象的系统设计/邵维忠,杨芙清著.—2版.—北京:清华大学出版社,2007.5

(计算机科学与技术学科研究生系列教材)

ISBN 978-7-302-14798-5

I. 面… II. ①邵… ②杨… III. 面向对象语言—程序设计—研究生—教材 IV. TP312

中国版本图书馆CIP数据核字(2007)第029513号

责任编辑:焦虹 顾冰

责任校对:时翠兰

责任印制:李红英

出版发行:清华大学出版社

<http://www.tup.com.cn>

c-service@tup.tsinghua.edu.cn

社总机:010-62770175

投稿咨询:010-62772015

地址:北京清华大学学研大厦A座

邮编:100084

邮购热线:010-62786544

客户服务:010-62776969

印刷者:北京市昌平环球印刷厂

装订者:三河市李旗庄少明装订厂

经 销:全国新华书店

开 本:185×260 印 张:14.25

字 数:340千字

版 次:2007年5月第2版

印 次:2007年5月第1次印刷

印 数:1~4000

定 价:29.00元

本书如存在文字不清、漏印、缺页、倒页、脱页等印装质量问题,请与清华大学出版社出版部联系调换。
联系电话:010-62770177 转 3103 产品编号:024412-01

“十一五”国家重点图书 计算机科学与技术学科前沿丛书

计算机科学与技术学科研究生系列教材

编
委
会

■ 名誉主任：陈火旺

■ 主 任：王志英

■ 副 主 任：钱德沛 周立柱

■ 编委委员：(按姓氏笔画为序)

马殿富 李晓明 李仲麟 吴朝晖

何炎祥 陈道蓄 周兴社 钱乐秋

蒋宗礼 廖明宏

■ 责任编辑：马瑛琚

本书责任编辑：钱乐秋

Abstract

This is a book on object-oriented design (OOD). Its first edition published in 2003 by our press had been widely used as a textbook for graduate or undergraduate students and as a technology handbook for software engineer in Chinese universities and software companies. In this edition, a lot of improvement have been made according as the development of the object-oriented technologies. This book is a companion volume to *Object-Oriented Systems analysis*, which is another composition of the same authors published by our press in 2006, and together they constitute an integrated method of Object-Oriented Analysis and Design.

There are seven chapters in this book. The first chapter makes a brief introduction to the OOD, and discusses the relationship between OOA and OOD. The second chapter summarizes the main characteristics of the OOD method suggested in this book. Chapter 3 to 6 expatiates upon the whole process of OOD, presents the modeling activities for every major components with detailed process guide and engineering strategies. Chapter 7 makes an introduction and comment to UML.

Audience: teachers, graduates and undergraduates majoring in computer software; teachers and students in software technology training schools; researcher and engineers in the field of computer software.

Keywords: software engineering, object-oriented design, UML

序

未来的社会是信息化的社会,计算机科学与技术在其中占据了最重要的地位,这对高素质创新型计算机人才的培养提出了迫切的要求。计算机科学与技术已经成为一门基础技术学科,理论性和技术性都很强。与传统的数学、物理和化学等基础学科相比,该学科的教育工作者既要培养学科理论研究和基本系统的开发人才,还要培养应用系统开发人才,甚至是应用人才。从层次上来讲,则需要培养系统的设计、实现、使用与维护等各个层次的人才。这就要求我国的计算机教育按照定位的需要,从知识、能力、素质三个方面进行人才培养。

硕士研究生的教育须突出“研究”,要加强理论基础的教育和科研能力的训练,使学生能够站在一定的高度去分析研究问题、解决问题。硕士研究生要通过课程的学习,进一步提高理论水平,为今后的研究和发展打下坚实的基础;通过相应的研究及学位论文撰写工作来接受全面的科研训练,了解科学研究的艰辛和科研工作者的奉献精神,培养良好的科研作风,锻炼攻关能力,养成协作精神。

高素质创新型计算机人才应具有较强的实践能力,教学与科研相结合是培养实践能力的有效途径。高水平人才的培养是通过被培养者的高水平学术成果来反映的,而高水平的学术成果主要来源于大量高水平的科研。高水平的科研还为教学活动提供了最先进的高新技术平台和创造性的工作环境,使学生得以接触最先进的计算机理论、技术和环境。高水平的科研也为高水平人才的素质教育提供了良好的物质基础。

为提高高等院校的教学质量,教育部最近实施了精品课程建设工程。由于教材是提高教学质量的关键,必须加快教材建设的步伐。为适应学科的快速发展和培养方案的需要,要采取多种措施鼓励从事前沿研究的学者参与教材的编写和更新,在教材中反映学科前沿的研究成果与发展趋势,以高水平的科研促进教材建设。同时应适当引进国外先进的原版教材,确保所有教学环节充分反映计算机学科与产业的前沿研究水平,并与未来的发展趋势相协调。

中国计算机学会教育专业委员会在清华大学出版社的大力支持下,进行了计算机科学与技术学科硕士研究生培养的系统研究。在此基础上组织来自多所全国重点大学的计算机专家和教授们编写和出版了本系列教材。作者们以自己多年来丰富的教学和科研经验为基础,认真研究和结合我国计算机科学与技术学科硕士研究生教育的特点,力图使本系列教材对我国计算机科学与技术学科硕士研究生的教学方法和教学内容的改革起引导作用。本系列教材的系统性和理论性强,学术水平高,反映科技新发展,具有合适的深度和广度。同时本系列教材两种语种(中文、英文)并存,三种版权(本版、外版、合作出版)形式并存,这在系列教材的出版上走出了一条新路。

相信本系列教材的出版,能够对提高我国计算机硕士研究生教材的整体水平,进而对我国大学的计算机科学与技术硕士研究生教育以及培养高素质创新型计算机人才产生积极的促进作用。

陈火旺

第 2 版 前 言

本书是一本介绍面向对象设计方法的著作,是我们的前一本著作《面向对象的系统分析》^[53]的姊妹篇,二者构成一个完整的面向对象的分析与设计方法体系。

面向对象的设计(OOD)是在面向对象的分析(OOA)基础上继续运用面向对象方法解决软件生命周期中设计阶段的问题,产生一个满足用户需求,并且完全可实现的系统模型,即 OOD 模型。在面向对象的软件开发中,系统分析建立的 OOA 模型离实现的要求还有很大的距离,因为还有很多设计问题尚未解决,需要在设计阶段运用 OOD 方法去解决这些问题,并且把设计结果在 OOD 模型中表达出来,使模型成为真正可实现的,这就是 OOD 所要解决的问题。

尽管 OOD 的出现早于 OOA,但是长期以来关于 OOD 的理论与技术却远不如 OOA 成熟,其内容也不如 OOA 充实,主要存在以下问题:

- 对什么是 OOD,各种著作和论文缺乏统一认识。主要原因是,早期的(即 OOA 出现之前的)OOD 和现今的(即基于 OOA 的)OOD 在内容上有很大的不同,而迄今大部分文献在讨论 OOD 时没有清晰地对二者加以区别,所以对 OOD 的概念、过程以及它应该包含哪些内容没有形成一致的见解。
- 以往大部分关于面向对象分析与设计(OOA&D)的著作都是以论述 OOA 为主,对 OOD 的论述则过于简略。对软件生命周期的设计阶段需要解决的大量实际问题缺乏全面、深入的讨论和切实可行的面向对象设计策略。有些方法对面向对象概念在设计阶段的运用基本上局限于对 OOA 结果进行细化,对许多全局性的设计问题没有给出与面向对象方法密切相关的设计策略,和在结构化设计中所采用的策略没有什么不同。
- 对 OOA 和 OOD 的关系,特别是对它们之间的界限和分工缺乏统一认识。一种基本一致的意见是,OOA 和 OOD 属于软件生命周期的两个不同阶段,它们之间的界限不像结构化分析和结构化设计之间那样严格和清晰;但是除了这种原则性的共同见解之外,没有更进一步的一致意见。特别是对 OOA 和 OOD 之间既存在界限,同时又允许一定程度的模糊这个问题,大部分著作没有深入地加以讨论,并具体指出哪些建模问题应该明确地属于 OOA 或者 OOD,哪些建模问题允许模糊二者之间的界限。
- 在以往的面向对象分析与设计方法中,Coad-Yourdon 方法^[8,9]是对 OOD 讨论最多的一种,对 OOA 和 OOD 之间关系的处理也较为得当。但是由于历史条件的限制,一些在 20 世纪 90 年代出现的理论与技术没有在有关的方法和著作中得以体现。对有些设计问题,例如一个用面向对象方法开发的系统如何用关系数据库或文件系

统存储其对象的问题,虽然提出了正确的解决方案,但是理论上的论述和技术策略的介绍都不够详细,使一般读者在学习之后仍然不知道如何实施。

本书作为一本在当前科学技术背景下出版的 OOD 著作,在学习和借鉴前人研究成果的基础上,力求在以下方面取得进步:

1. 系统地阐述 OOA 与 OOD 的理论体系

本书从区别早期的 OOD 和基于 OOA 的 OOD 入手,通过讨论二者在内容和特点上的不同,在概念上澄清了关于什么是 OOD 的问题。把面向对象的观点运用于整个软件生命周期,在此前提下对什么是 OOD 给出更确切的定义。详细地论述了 OOA 和 OOD 之间的关系。

2. 充实和完善 OOD 的内容

在面向对象的软件开发中,OOD 是软件生命周期中的一个阶段。在这个阶段中有大量的技术问题需要解决,需要建立一个可实现的系统模型。从工作量和难度来看,OOD 的分量决不比 OOA 小。许多运用 OO 方法进行系统开发的读者所遇到的一个问题是,做完 OOA 之后不知道 OOD 到底该做些什么,因此对设计阶段的许多问题不得不继续采用非 OO 的设计技术去解决。本书的目标是向读者提供一种内容比较完善、策略具体、可操作性强的 OOD 方法,其中包含了普通应用系统的设计阶段需要解决的大部分问题,包括全局性设计决策和局部的模型细化两个方面的问题。

3. 充分运用 OO 基本概念解决设计问题

我们在《面向对象的系统分析》中提出,在 OOA 中应该充分运用面向对象的基本概念(即目前大部分面向对象编程语言能够直接支持的概念)解决各种复杂的建模问题,限制扩充概念的引入。本书依然坚持这一宗旨,没有采用比 OOA 更多的面向对象建模概念,更没有采用诸如“模块”、“块”等非 OO 的建模元素。此外,本书对所有的设计问题都是运用面向对象的观点给出设计策略,使读者能够在软件开发中完全采用面向对象的概念和表示法来建立系统的设计模型。这意味着本书运用了尽可能少的建模概念解决了较多的设计问题。建模概念的简练使本书提出的方法更容易学习、掌握和使用,并使得 OOA、OOD 和 OOP 在概念上保持高度一致,使模型与实现后的程序具有良好的映射关系。当然,为了做到这一点,本书给出了更强的过程指导,告诉读者如何运用一个精练的 OO 基本概念集合去解决各种复杂的建模问题。

4. 适应当前计算机科学技术的新发展

迄今在国际上影响较大的大多数 OOA&D 方法都是在 20 世纪 80 年代末和 90 年代初问世的。在此之后的近十年中,计算机科学技术在很多方面取得新的进步。例如软件复用技术的新发展、可视化编程环境的流行、软件体系结构的研究、网络与分布式系统的普及等。大量的新技术已被广泛地运用到当前的系统开发中。在这种形势下,要求 OOD 方法做出相应的发展。我们的目标是力求在 OOD 方法中体现计算机科学技术的新发展。本书的 OOD 方法是根据当前的技术背景提出的。针对在当前软件开发中被广泛采用的几种主要

技术,分别给出相应的设计策略,并且用 OO 概念表达其设计决策。

5. 解决工程实践中提出的问题

在本书的写作过程中,作者曾经以技术培训、工程指导、项目合作等多种方式与软件企业界的人士进行接触和交流。他们在运用面向对象方法进行软件开发时,常常在 OOD 阶段遇到这样那样的问题。其中有些问题在以往的著作中找不到现成的答案。这些来自工程实践的问题对于 OOD 方法的研究和发展具有很强的促进作用。本书的许多内容是针对这些问题开展研究并进行总结提炼的结果。

本书的第1版^[51]于2003年2月出版,数年来已多次重印。这次再版主要有以下变化:

- 按照《面向对象的系统分析》从第1版^[48]到第2版^[53]的内容变化,对本书的内容做了相应的修改与调整。
- 根据 UML2.0 的最新版本和我国的“信息技术 软件工程术语”(GB/T 11457—2006)国家标准^[45]等标准化文件,进一步规范了全书的术语和模型元素的表示法。
- 在第7章对 UML1.x 的介绍和评论中,增加了引导读者阅读文献^[53]关于 UML2.0 相关内容的线索。
- 对全书的内容进行了全面的审核和订正,在文字表述方面比第1版更为准确和精练。

全书共分7章,其内容安排如下:

第1章讨论什么是 OOD。通过区分早期的 OOD 和基于 OOA 的 OOD,论述了 OOD 在不同历史时期的不同特点及内容。简要地介绍了几种典型的 OOA&D 方法。对历史上关于 OOA 与 OOD 之间关系的一些传统观点进行了分析和讨论,并从模型驱动的体系结构(MDA)的观点出发论述了应如何处理二者之间的关系。

第2章概括地介绍本书提出的 OOD 方法概貌,包括它所采用的概念、表示法、原则、模型及过程。这一章的内容是整个 OOD 方法的总纲。

第3章介绍 OOD 模型的核心部分,即问题域部分的设计。内容包括在 OOA 模型基础上针对编程语言、复用支持、硬件性能等实现条件对模型进行修改、补充和调整,完善对象的细节,定义对象实例,建立从 OOA 模型到 OOD 模型之间的映射。

第4~6章分别介绍 OOD 模型的三个外围组成部分。每个外围部分是针对一个方面的实现条件来设计的,其作用是处理问题域部分与这些实现机制的接口,并且隔离它们的变化对整个系统的影响。第4章实际上既包括人机交互部分的需求分析,也包括该部分的设计。在分析方面给出的主要策略是:从用况提取人机交互需求,然后对交互过程进行细化,进而确定命令的组织结构;在设计方面讨论的主要内容是:根据给定的实现条件选择实现人机交互的界面元素,并且在模型中用 OO 概念表示所有的界面元素以及它们之间的关系。第5章主要解决并发系统和分布式系统中控制流的设计问题。首先讨论了系统总体方案、软件体系结构、系统的并发性等相关技术问题,然后讨论如何按选定的软件体系结构风格确定系统的分布方案,进而识别系统中的控制流。最后给出用面向对象概念表示系统中进程和线程的设计策略。第6章讨论如何解决对象的永久存储问题。首先介绍文件系统、关系数据库管理系统和面向对象数据库管理系统这三种最主要的数据管理系统的原理、功能和技术特点,然后针对不同的数据管理系统给出相应的设计策略。对使用文件系统和关系数

数据库管理系统的情况,首先论述了用非OO的数据管理系统实现对象存储的理论依据,进而给出把应用系统中的对象映射到文件或者关系数据库的存储方案,并详细地介绍数据接口部分的设计策略。

第7章介绍统一建模语言UML,并进行分析和评论。包括UML的背景及演化历史,语言体系结构及定义方式,UML规范中的主要文献,各种图、建模元素及表示法。着重解释那些使读者和用户感到困惑和难以理解的概念,目的是排除读者在学习和使用UML时的主要障碍。第2版保留了这一章中针对UML1.x中的介绍和评论,同时介绍了从UML1.x到UML2.0的演化背景,并给出引导读者阅读文献[53]中关于UML2.0相关内容的线索。

本书的研究与写作得到国家自然科学基金项目“元模型理论与建模方法研究”(60473064)和国家重点基础研究发展计划(973计划)课题“虚拟计算环境的程序设计方法学”(2005CB321805)的支持。在写作过程中,我们曾就书中的许多学术思想与南京大学徐家福教授,北京航空航天大学金茂忠教授,北京大学王立福、梅宏、孙家骕、麻志毅、谢冰、王千祥、张世琨等同事,以及其他许多专家和应用领域的朋友进行了讨论和印证,从中受益良多。北京大学软件研究所的OO建模工具课题组所研制的建模工具集JBOO4.0为本书和文献[53]中的OOA与OOD方法提供了技术支持。我们的许多博士生和硕士生为本书的写作查阅、收集了大量文献资料,并绘制了其中的一部分插图。在此,我们谨向上述单位和个人致以衷心的感谢!最后,我们恳切希望各位读者对本书可能存在的错误与疏漏给予批评指正。

作者联系信息

联系人:邵维忠

通信地址:北京大学信息科学技术学院; 邮政编码:100871

电话:010-62751790; 传真:010-62751792

电子邮件:wzshao@pku.edu.cn

作者

2006年12月于北京大学

目 录

第 1 章 什么是 OOD	1
1.1 早期的 OOD	1
1.2 基于 OOA 的 OOD	2
1.2.1 Booch 方法	4
1.2.2 Coad/Yourdon 方法	7
1.2.3 Jacobson 方法	9
1.2.4 Rumbaugh 方法	14
1.3 OOD 在软件生命周期的位置	18
1.4 OOA 与 OOD 的关系	19
1.4.1 “做什么”和“怎么做”	19
1.4.2 分析——需求分析和系统分析	20
1.4.3 “问题空间”和“解空间”	21
1.4.4 一致的概念与表示法	22
1.4.5 不同的目标、内容和抽象层次	23
1.4.6 从模型驱动的体系结构看 OOA 与 OOD 的关系	24
第 2 章 本书的 OOD 方法概貌	26
2.1 概念与表示法	26
2.1.1 建模元素	27
2.1.2 表示法	28
2.1.3 建模原则	29
2.2 OOD 模型框架	33
2.3 OOD 过程	35
第 3 章 问题域部分的设计	36
3.1 什么是问题域部分	36
3.2 实现条件对问题域部分的影响	36
3.3 设计过程	37
3.3.1 设计准备	37
3.3.2 设计内容及策略	38

3.3.2.1	针对编程语言支持能力的调整	38
3.3.2.2	增加一般类,提供共同协议	43
3.3.2.3	为实现复用采取的设计策略	44
3.3.2.4	提高性能	46
3.3.2.5	为实现对象永久存储所做的修改	52
3.3.2.6	完善对象的细节	52
3.3.2.7	定义对象实例	57
3.3.2.8	修改或补充辅助模型及模型规约	58
3.3.3	建立与 OOA 文档的映射	59
第4章	人机交互部分的设计	62
4.1	什么是人机交互部分	62
4.2	人机交互部分的需求分析	63
4.2.1	分析参与者——与系统交互的人	63
4.2.2	从用况分析人机交互	64
4.2.3	分析处理异常事件的人机交互	68
4.2.4	命令的组织	69
4.2.5	输出信息的组织结构	73
4.2.6	总结与讨论	74
4.3	人机界面的设计准则	75
4.4	人机界面的 OO 设计	77
4.4.1	界面支持系统	77
4.4.2	界面元素	78
4.4.3	设计过程与策略	79
4.5	可视化编程环境下的人机界面设计	86
4.5.1	问题的提出	86
4.5.2	所见即所得的界面开发	87
4.5.3	设计的必要性	90
4.5.4	基于可视化编程环境的设计策略	91
第5章	控制驱动部分的设计	97
5.1	什么是控制驱动部分	97
5.2	相关技术问题	97
5.2.1	系统总体方案	98
5.2.2	软件体系结构	99
5.2.3	分布式系统的体系结构风格	100
5.2.4	系统的并发性	103
5.3	如何设计控制驱动部分	111
5.3.1	选择软件体系结构风格	111

5.3.2 确定系统分布方案	111
5.3.3 识别控制流	116
5.3.4 用主动对象表示控制流	119
5.3.5 把控制驱动部分看成一个包	121
第6章 数据接口部分的设计	122
6.1 什么是数据接口部分	122
6.2 数据管理系统及其选择	123
6.2.1 文件系统	123
6.2.2 数据库管理系统	124
6.2.2.1 关系数据库和数据库管理系统	125
6.2.2.2 面向对象数据库和数据库管理系统	127
6.2.3 数据管理系统的选择	129
6.3 对象存储方案和数据接口的设计策略	131
6.3.1 针对文件系统的设计	131
6.3.1.1 对象在内存空间和文件空间的映像	131
6.3.1.2 对象存放策略	132
6.3.1.3 设计数据接口部分的对象类	134
6.3.1.4 问题域部分的修改	135
6.3.2 针对 RDBMS 的设计	136
6.3.2.1 对象及其对数据库的使用	136
6.3.2.2 对象在数据库中的存放策略	139
6.3.2.3 数据接口部分的对象设计和问题域部分的修改	148
6.3.3 使用 OODBMS	150
6.4 本章小结	151
第7章 UML 介绍与评论	152
7.1 UML 的背景与演化历史	152
7.2 UML 内容简介	154
7.2.1 UML 是什么,不是什么	154
7.2.2 UML 规范的主要文献	155
7.2.3 UML 语言体系结构及定义方式	157
7.2.3.1 四层元模型体系结构	157
7.2.3.2 包结构	159
7.2.3.3 语法及语义定义方式	160
7.2.3.4 关于若干术语译法问题的讨论	162
7.2.4 UML 的各种图、建模元素及表示法	164
7.3 评论与问题研究	178
7.3.1 UML 的成就及其原因	178

7.3.2	UML 能够起到的积极作用	179
7.3.3	UML 不能起到的作用	180
7.3.4	UML 的缺点与问题	181
7.3.4.1	与四层元模型体系结构有关的问题	182
7.3.4.2	形式化方面的问题	184
7.3.4.3	类图和对象图并存问题	186
7.3.4.4	协作图的问题	188
7.3.4.5	无道理的复杂性	193
7.4	从 UML1.x 到 UML2.0	196
7.4.1	UML2.0 提案需求	196
7.4.2	被采纳的 UML2.0 提案	197
7.4.3	UML2.0 概况	197
7.4.4	进步与不足	201
索引		203
参考文献		206

第 1 章

什么是OOD

顾名思义,面向对象的设计(OOD)就是运用面向对象方法进行系统设计。然而这只是一种很粗略的解释。在不同的历史背景和不同的场合下,OOD 的目标、内容和方法有许多不同,确切地解释其含义须从历史说起。

1.1 早期的 OOD

在软件生命周期的各个阶段全面地运用面向对象方法进行系统开发,是人们长期以来努力追求的目标。特别是,从分析与设计阶段就开始运用面向对象方法,比仅仅用面向对象编程语言来编程更为重要,并且更能从根本上发挥面向对象方法与技术的优势。对此我们曾在《面向对象的设计》^[47]的译者序和《面向对象的系统分析》^[48,53]的第 1 章中进行了论述。在面向对象的软件开发中,通常是首先进行面向对象的分析(OOA),其次进行面向对象的设计(OOD),然后进行面向对象的编程(OOP)和测试(OOT)。但是面向对象方法的发展历程,却是首先开始于 OOP,然后发展到 OOD、OOA 和 OOT,即 OOD 的出现早于 OOA。

20 世纪 80 年代初发布的 Smalltalk-80 是第一个比较完善的面向对象编程语言(OOPL)。随后涌现的一大批 OOPL 标志着 OO 方法走向了实用。此时人们认识到面向对象的软件开发不仅仅是编程问题,在采用 OOPL 所提供的类、对象等元素和封装、继承等机制来编写程序之前,必须先用面向对象的观点进行构思和设计。这样才能明确程序中到底应该定义哪些类和对象,并明确它们的内部特征和相互关系。

这一认识促使人们将面向对象的思想向前推进了一步——从单纯地解决编程问题推进到从设计阶段就面向对象运用这一思想。G. Booch 于 1982 年发表了题为“Object-Oriented Design”的论文^[2],首次使用了“面向对象的设计”的术语;1986 年,他又发表了题为“Object-Oriented Development”的论文^[3],较完整地阐述了他的 OOD 思想。当时,“面向对象的设计”和“面向对象的开发”都用 OOD 作为缩写,并且在内容上也没有根本区别——称“开发”时,也是主要讨论设计问题;称“设计”时,也不是单纯地针对软件生命周期的设计阶段,还涉及一些本应属于分析的工作。

R. J. Abbott 在 1983 年提出一种通过正文分析来发现对象的方法,即用规范的英语对问题进行陈述,然后从正文中提取对象及其特征^[1]。例如,通过名词识别对象,通过动词识别对象的操作以及通过定语成分识别对象的属性等。这种正文分析方法被后来的许多 OOD 方法采用。

Booch 方法^[3]被视为最早的 OOD 方法。在此之后,从 1986 年到 20 世纪 90 年代初,相继出现了一批 OOD 方法,其中有:

- 通用面向对象的开发(general object-oriented development,GOOD)^[31];
- 层次式面向对象的设计(hierarchical object-oriented design,HOOD)^[16];
- 面向对象的结构设计(object-oriented structured design,OOSD)^[39]。

由于面向对象的分析方法在 20 世纪 80 年代末尚未出现,所以当时提出的 OOD 方法都受到了这种历史背景的限制,其影响延续到 20 世纪 90 年代初期。在这种背景下出现的 OOD 方法具有以下特点:

(1) 不是在面向对象的分析基础上进行面向对象的设计,而是基于结构化分析。例如早期的 Booch 方法和 GOOD 都是从结构化分析所产生的数据流图(DFD)开始进行面向对象的设计,HOOD 是自顶向下地进行对象分解,其中要用到一些结构化分析和结构化设计的图形表示(如 DFD),并将所得到的结构化概念映射为对象及其外部接口。OOSD 也部分地基于结构化方法,是一种结构化和面向对象相结合的方法,为熟悉结构化设计的开发者提供了逐渐过渡到 OO 方法的措施。

(2) 大部分方法是针对特定编程语言的。例如早期的 Booch 方法是针对 Ada 和 Modula-2 的,GOOD 和 HOOD 都是针对 Ada 的。只有少数方法(例如 OOSD)是独立于语言的。针对特定编程语言的 OOD 方法在很大程度上受到语言的影响,例如将包或模块的概念反映到 OOD 方法中来。

(3) 从现今的观点看,早期出现的 OOD 方法大部分不是纯 OO 的。一方面对一些重要的 OO 概念缺少支持,例如 Booch 方法、GOOD 和 HOOD 都不能表示类之间的继承;另一方面却很注重对某些非 OO 概念(例如数据流、模块和异常处理等)的表示。

(4) 名曰“面向对象的设计”,实际上不只是针对设计问题。早期的许多 OOD 方法往往在不同程度上包括了 OOA 阶段的工作,例如识别问题域中的对象。但是其识别对象的策略很不完善,特别是没有哪种方法强调从考察问题域中的事物入手来确定系统中的对象。因此,早期的 OOD 方法可以看成是现今的 OOA&D 方法的雏形,而不是只对应其中的 OOD 部分。原因是 OOA 方法在当时尚未形成,编程之前所有的建模问题(包括分析与设计)都属于新生的 OOD 方法所考虑的范围。对分析问题表达能力和处理策略的不足使之在很大程度上借助和依赖结构化分析。

1.2 基于 OOA 的 OOD

OOD 的出现是面向对象方法与技术的进步——在编程之前,首先运用面向对象方法建立一个系统设计模型,比单纯地用 OOPL 编程更能保证系统的 OO 风格。然而,在 OOD 中用来确定系统设立哪些对象类的依据是什么?这一根本问题尚未真正解决。从结构化分析的结果 DFD 来发现对象,显然不是运用 OO 原则的最好途径。面向对象方法的基本思想之一就是从问题域中客观存在的事物出发来识别对象并建立由这些对象所构成的系统,而不是先把问题域抽象为某种已经很难辨认事物本来面貌的非 OO 的软件概念(如 DFD),然后再通过它们来识别对象。识别对象最准确、最可靠的途径是以面向对象的观点直接地对问题域进行分析与研究,以系统责任作为当前目标对问题域中的事物进行抽象,从而确定系统