计算机软件新技术

——面向对象的系统分析

李芳芸　柴跃廷　编

清华大学出版社

计算机软件新技术

——面向对象的系统分析

李芳芸　柴跃廷　编著

清华大学出版社

内 容 简 介

本书比较详细全面地介绍了面向对象的系统分析的基本概念和基本方法。对面向对象的系统分析的关键——面向对象的信息模型的概念、建立方法、描述方法以及信息模型在系统设计中的作用作了深入的讨论。

本书在论述内容和风格上有自己的显著特色，它从实用的观点出发，通过具体实例阐述有关概念和方法。语言通俗易懂，论述清楚、直观，内容易于理解和掌握。

本书可作为系统工程专业、管理工程专业和计算机工程专业的系统分析与设计课程的教学参考书。同时也可作为有关专业的工程技术人员的技术参考书。

（京）新登字158号

计 算 机 软 件 新 技 术

——面向对象的系统分析

李芳芸　柴跃廷　编著

责任编辑：魏荣桥　蔡鸿程

☆

清华大学出版社出版

（北京 清华园）

北京通县向阳印刷厂印刷

新华书店总店科技发行所发行

☆

开本：787×1092 1/32　印张：5.5　字数：122千字

1992年1月第1版　1992年1月第1次印刷

印数：0001—8000

ISBN 7-302-00942-2/TP·345

定价：2.90 元

编 者 序

面向对象的技术是80年代以来国外关注的问题，特别是近几年，面向对象的研究遍及计算机软、硬件系统的各个领域，如面向对象的程序设计语言、面向对象的程序设计方法、面向对象的操作系统、面向对象的数据库管理系统以及面向对象的系统分析方法等。

在大型软件开发项目中，无论是实时过程控制系统、数据库系统、专家系统、人工智能系统，还是具体到自动代码生成，或扩展到一个工厂的计算机集成制造系统，都存在着一个共同的需求，这就是在系统分析阶段建立系统的信息模型。

建立信息模型的核心问题是：

· 如何规范化或观测一个模型？

· 如何用图表表示或描述一个大型模型？

· 如何为一个信息模型提供文本说明？

书中讨论了为什么要建立信息模型以及信息模型在系统开发中的作用，并叙述了面向对象的系统分析方法和利用这种方法来解决上述三个核心问题。

书中主要内容参阅了Sally Shlaer、Stephen J. Mellor 所著"Object-Oriented Systems Analysis——Modeling the World in Data"一书，并参阅了"An Object-Oriented Methodology for Systems Analysis and

Specification"(Barry D. Kurtz 等, hewlek-packard Jounal appil, 1989)等大量的国内外有关资料。由于我们的水平有限，书中难免有错误，欢迎读者批评指正。

编 者

目　录

第一章 概 述

在本章里，我们主要介绍了面向对象的基本概念、面向对象的程序设计语言、面向对象的数据库技术、面向对象的系统分析与设计方法的发展过程。强调了在系统分析中建立信息模型的必要性。了解这些概念，弄清信息模型在系统分析中的地位和作用，对于深入学习以后各章颇有益处。

§ 1-1 历史的回顾

随着计算机科学的发展，应用领域的不断扩大，对计算机技术本身的要求越来越高。特别是当计算机硬件有了飞速发展之后，各种应用领域对软件提出了更高的要求。例如，软件的质量、软件开发的生命周期、软件的可靠性和重用性等。计算机软件技术的发展滞后于硬件，加之60年代后期的软件危机，使得软件的结构和简明性，而不是表达能力，成为语言设计的基础。软件工程学应运而生，提出了一些结构化程序设计语言和结构化分析与设计方法作为应用程序设计的基础。对于软件，人们认识的重点从组成程序的语句序列转到了构成软件的模块序列。美国国防部为解决软件危机而开发的Ada语言支持了包括函数、过程、程序包、任务与类属程序单元在内的各种模块。60年代中期挪威开发的仿真程序语言Simula67首次出现了对象的概念。同时Simula67 是

第一种面向对象的程序设计语言。程序设计语言支持对象的概念是十分重要的。可以说，面向对象的概念的出现是程序设计方法学和软件工程方法学的一个新的里程碑。

客观世界的问题都是由客观世界中的实体及其相互之间的关系构成的。我们将客观世界中的实体抽象为问题空间中的对象。因问题具有特殊性，因此对象是不固定的。一本书可以是一个对象，一家图书馆也可以是一个对象。

从本质上讲，我们用计算机解题，是借助于某种语言的规定对计算机实体施加某种动作，以此动作的结果去映射解。我们把计算机实体称为解空间对象。显然，这个对象因语言而异。例如：汇编语言提供的对象是存贮单元；过程语言是各种内部定义的变量、数组、记录和文件；一旦提供了某种解空间的对象，就隐含着该对象允许的操作。例如，布尔变量只能进行布尔操作，存贮单元只能作存取操作。在一个程序中，这种操作是共享的。如，不同的布尔变量都可以作布尔加法。

从动态的观点看，对象及其操作就是对象的行为。问题空间中对象的行为是极其丰富多彩的，而解空间中的对象的行为是极其死板而又无特殊性的。因此，只有借助于极其复杂的方法才能操纵解空间中的对象从而得到解。这就是常说的"语义断层"，也是程序设计始终是一门学问的原因。人们只能在程序设计语言提供的对象和操作的概念的基础上作程序设计。愈是高级的语言，提供的概念愈多，愈是远离机器和过程实现的概念，则愈宜人。

面向对象的程序设计语言提供的概念——对象，是让程序员自己去定义解空间对象，首先描述对象及其行为，然后

用传统的方法实现。例如，数据堆栈可以作为一个对象，它必须有一个栈体存放数据，可按照后进先出的次序压入或弹出数据。栈满了不能压入，栈空了不能弹出。至于栈体中存放什么类型的数据，并不是堆栈行为的主要特征。面向对象提供一种机制，使得私有数据有其私有操作，即描述这个对象(数据)的独特行为。这样就向着减少语义断层的方向迈进了一步，在某些问题上可以直接模拟问题空间的对象，因而，写出的程序易于理解和维护。

从存贮的角度看，对象是一片私有存贮，其中有数据及其操作。其它对象的操作不能直接操作该对象的私有数据，只有对象的私有操作才可以操纵它。

从对象实现的机制看，对象是一台自动机，其中私有数据表示了对象的状态，该状态只能由私有操作来改变。每当需要改变对象的状态时，只能由其它对象向该对象发送消息，对象响应消息后按照消息模式找出匹配的方法(操作)，并执行相应的操作。发送消息和过程调用是不同的，发送消息只能触发自动机，同样的输入参数可因自动机的状态不同从而得出不同的终态。而调用过程时，只要输入相同，输出总是恒定的。

至此，我们对对象的初步概念可理解为：对象是一个封装数据和操作的实体。数据描述了对象的状态，操作可操纵私有数据，改变对象的状态。每当其它对象向本对象发出消息时，本对象响应时，其操作才得以实现。

自从80年代 Smalltalk 及其环境向计算机界推行以来，面向对象的技术引起了计算机界的极大关注。因为面向对象的技术对于软件工程学濒临的困境和人工智能所遇到的障碍

都是一个很有希望的突破口。所以，近几年来，面向对象的研究遍及计算机软硬件的各个领域，如面向对象的程序设计语言，面向对象的程序设计方法，面向对象的操作系统，面向对象的数据库，面向对象的软件开发环境及硬件支持等，目前已取得丰硕成果。虽然有些软件、硬件产品已投入使用，但研究仍在继续，特别是在面向对象的理论方面。

概括起来，面向对象的技术可分为：面向对象的程序设计语言，面向对象的数据库技术，面向对象的系统分析与设计方法。

一、面向对象的程序设计语言

自从60年代以来，相继出现了各种面向对象的程序设计语言，其大致的发展过程如图1-1所示。

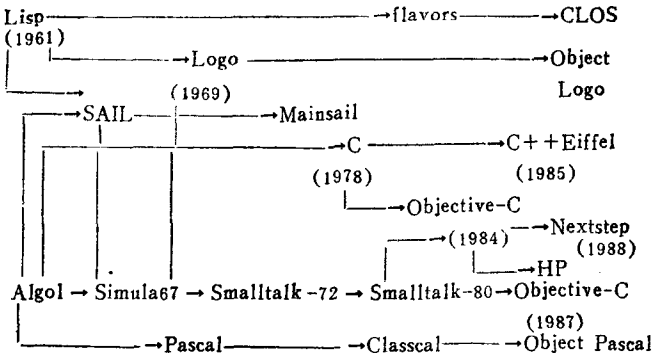

图 1-1　面向对象的语言的发展过程

可以说，除了Eiffel以外，其它各种语言都起源于某种非面向对象的语言。由于面向对象的技术仍处于发展之中，因此，存在着多种面向对象的语言，而没有一种是标准的。

Simula67 是 60 年代挪威开发的一种仿真程序语言，Smalltalk-70 是70年代末由 Xerok，Corporation's palo Alto 研究中心研制出来的，用对象的概念表述系统的各个组成部分乃至整个系统。1981年，HP实验室将Smalltalk应用于个人计算机，但由于其操作性能差，而未得到普遍使用。尽管如此，Smalltalk 的思想形成了今天的面向对象系统的雏形。

到了1983年，包括 HP 在内的各个研究中心都致力于实现面向对象的语言，这些语言都利用预处理器将面向对象的语言翻译成传统的程序设计语言，如 C，Pascal 等。这种思想为实现面向对象的语言提供了捷径。而且和传统的语言是兼容的。有利于充分地利用已有的方法和技术。

在人工智能领域，由 Flavors 语言产生了普通的 Lisp 对象系统(Common Lisp Object System)，简称 CLOS. Pascal 作为面向对象的语言的基础，除了在 Pascal 基础上实现了面向对象的概念-类之外，还产生了 Clascal。最近又形成了Object Pascal。

由 Stepstone 公司开发的 Objective-C 语言是一种预处理器。它可以将标准C代码与Smalltalk 混合起来。由AT&T贝尔实验室开发的C++语言是另一种产生C代码的预处理器。最近从C派生出来的面向对象的语言是Next Step。

Eiffel 是一种相对新的语言，它可以更有效地支持软件工程。

面向对象的语言种类繁多，尤其是80年代以来，发展迅速。可以想象，随着研究的深入，面向对象的语言会更趋完善，最终形成相对标准的程序设计语言。

二、面向对象的数据库技术

随着对面向对象技术的研究的不断深入，近年来，面向对象的数据库技术也有了长足的发展。数据库是信息系统的核心，由于信息系统的迅速发展，研究并开发更适合于应用要求的数据库系统已成为一个重要的研究领域。70年代初提出来的关系数据模型，由于它是面向终端用户的，把数据的逻辑模型和物理实现分开，设计了标准的非过程性查询语言等，已成为目前应用最广泛的数据库系统。但关系数据库也有许多明显的弱点。如关系模型缺乏对实体的结构进行描述，不能反映实体之间的层次关系和组成关系，而对于大而复杂的系统，描述实体之间的层次关系和组成关系是必不可少的。如一些工程数据库，计算机辅助工程设计和制造用数据库，图像数据库、地理数据库、计算机集成制造系统的信息管理与决策支持系统数据库等。这些领域的应用环境比传统的应用环境复杂得多。为了满足这些新的应用环境的要求，人们提出了许多新的数据模型，面向对象的数据模型就是在这种背景下产生的。

面向对象的数据模型是一种语义关联模型。其基本组成单元是数据对象。对象是现实问题中的一个实体，类似于关系模型中的实体。用属性描述对象的特征。并指定某个属性(或多个属性组合)作为对象的标识符。如实体类：人、作为一个对象，其属性有姓名、年龄、性别、……。对象之间的关系概括为归纳关系、组合关系、关联关系。归纳关系反映一个对象类与若干个互不相容的子类之间的分类关系。高层类说明一般属性，低层类说明特殊属性。低层类对象继承高层类对象的属性。如对象类设备可分为机床类、电机类、吊车

类等子类，子类机床类、电机类、吊车类可以继承设备类属性。组合关系反映对象之间的构成关系。如产品的BOM(Bill of Material)可利用组合关系加以描述和抽象。关联关系反映对象之间的相互依赖、相互作用的关系。分为1：1、1：M、M：N三种。关联关系与关系模型中的关系是同义的。在某种意义上讲，面向对象的语义关联模型可以看成是关系模型的发展。是在关系模型、网状模型和层次模型的基础上发展起来的。但一个成功的面向对象的数据库管理系统可以将数据和应用程序结合起来，显示出它的独到之处，应用于复杂环境。

面向对象的语义关联模型可以转化实体的结构，为所研究的问题领域提供适当的抽象层次，减少了数据模型和数据库设计语言的差别以及数据模型和应用程序之间的语义差别，为数据库技术的发展提供了美好的前景。近年来，国内外对面向对象的数据库的研究都取得了很大的进展。在国外，已有商品化的面向对象的数据库管理系统投入使用。

三、面向对象的系统分析与设计方法

对象和操作并不是程序设计的新概念，但是，面向对象的系统分析与设计方法却是一种新的方法。在计算机技术发展的早期时候，系统语言使得程序员能够使用机器指令操作数据，这种方法解决问题的抽象层次是很低的。随着高级语言(例如FORTRAN，ALGOL，COBOL)的出现，现实问题中的对象和操作可以用高级语言中预先定义的数据和控制结构进行描述。一般软件设计都着眼于用所选择的程序设计语言对处理过程的细节进行描述，从而引入了诸如功能的逐步求精，过程模拟块以及结构化程序设计等软件设计的概念。

在70年代，引入了数据抽象和信息隐藏的概念，出现了数据驱动的设计方法，但是软件设计者仍将精力集中在处理过程及其描述上，同时现代的高级语言(例如Pascal、C)引入了更加丰富的数据结构和类型。

所谓面向对象的设计方法是在过去十几年当中发展起来的，它强调了系统设计之前的系统分析，强调以系统中的数据或信息为主线，全面、系统、详尽地描述系统的信息，建立系统的信息模型，指导系统的设计。从某种程度上讲，它是一种数据驱动的系统分析与设计方法。传统的面向数据结构的设计方法(DSSD)和Jackson系统开发方法(SSD)都可看作是面向对象的设计方法。

在80年代，随着计算机技术的普及，其应用领域不断扩大，诸如数据库系统、专家系统、实时过程控制系统、系统软件、自动代码生成、人工智能、计算机集成制造系统等。软件系统也变得愈来愈大，愈来愈复杂，表现在下列几个方面：

(1) 大型软件系统。对于这样的系统，通常有不同层次不同领域的技术人员参与开发，如何协调与控制开发人员的工作，才能保证开发项目的顺利进行，而不致引起混乱从而导致系统开发的失败？

(2) 需求变化的系统。有些软件系统，在开发初期，由于用户是不固定的，因而其需求是不明确的或者是动态变化的。那么如何组织系统的信息，进行系统设计，使得最终系统能够适应需求变化的要求，而不会作大的改动？

(3) 需求模糊的系统。有些系统在开发初期，其需求是不明确的，不完全的，或不一致的。如何识别和确定系统的

需求，进而进行系统设计？

(4) 新的应用领域。对于系统开发人员来讲。面对新的软件开发系统，如何能深入调查与理解系统，确定系统的结构？

(5) 软件系统涉及不同的专业。一个软件系统通常会涉及多种不同的领域和专业知识和各种不同的专业术语。如何正确有效地组织系统的信息，适合开发的需求？

面对这些新的复杂的问题，传统的系统分析与设计方法，由于它们不能很好地正确捕捉和组织系统中的所有信息，从而不能满足分析与设计的要求。自然人们会提出这样的问题：如何能够正确、有效、全面地描述和组织整个系统的信息，以此来指导系统的开发？基于面向对象和数据驱动的思想，人们对面向对象的系统分析与设计方法产生了极大的兴趣，从而展开了对面向对象的系统分析与设计方法的研究，并把它应用于各个领域。

§ 1-2 为什么要建立信息模型

一、软件开发中的困难

建立一个大型信息处理系统为什么如此困难？因为在软件开发的过程中，经过许多开发步骤以后，才能得到一个满足用户需求和期望的系统(当然，是在时间和资金允许的范围内)，而且是一个易于维护、修改和理解的系统。那么，在开发过程中，我们究竟要做些什么工作呢？准确地弄清这些事情是非常困难的。

根据经验，我们知道有些问题是基本的：这些问题反复

出现，它们危害着大大小小的开发项目。也就是说，它们是开发项目失败的主要原因。这些问题是以一种或另一种方式来源于信息，或者说是来源于错误信息。对于下述问题，我们已经实践过：

1．开发项目超出了专业范围

当一个计算机专家接纳一项新的工作，或者被安排到一个新的项目，他通常面临一个新的应用领域，过去的经验和正规培训常常与新的工作完全无关，而是涉入一个新的专业知识领域，诸如会计、通讯、财政、化学处理、铁路客流控制等等。

问题在于突然有许多东西要学，而且需要消耗时间和精力。然而，专门用于学习的时间和精力是有限的，而且初次学习没有现成的框架和方法。

由于最终系统的规划是以专业知识为根据的，因此，我们必须找到一种方法，用来识别生疏领域的知识，并将它应用到软件开发过程中。

2．多种术语/学科交叉的研究

大的软件系统通常需要多种专业知识的集成：财务管理人员、审计员、工程师、操作技术人员都必须加入到需求分析过程中去。为了设法做到这一点，虽然每个部分都具有相对独立性，但是有时具有相互矛盾的词汇或术语，应着重发现相互重迭的专业知识范围。例如，对于冷却发动机的炉缸，是根据热交换和水流来命名的；对于真空发动机的炉缸，是一个必须抽空的容器。同一个词汇——炉缸，却用于两个明显不同的物理实体，尽管它们是密切相联的。同样，订单这一词汇，在发货部门和销售部门也是完全不同的。