软件开发 面向对象模型 软件工程

计算机科学1996 Vol. 23№. 1

62-65

软件开发与面向对象模型*)

王斌君 郝克刚

(西北大学计算机科学系 西安 710069)

TP311.52

摘 要 From software development views, this paper discusses some important object-oriented models in current, and give an ideal 3-DOOM model by myself.

关键词 Model, Object-oriented model, Three-dimensional object-oriented model.

一、引 言

70、80年代,以结构化方法为代表的软件工程方法使得人们开发软件可按步骤、有章可循地组织完成,但它并没有从根本上彻底解决60年代末出现的软件危机所带来的问题,其原因是计算机可接受的软件模型(程序设计语言的结构)与其要解决的现实世界中的问题结构大相径庭。面向对象的程序设计语言(OOPL)可直接地表达问题域中的对象(如人、马达、栈等)和结构(如分类结构、各种关系等),使得软件的模型与问题空间模型非常地相近和相似,从而保证了软件易于构造。

近几年,人们为了充分地发挥OOPL的优势,将研究的精力集中在面向对象的设计(OOD)[1,2]和面向对象的分析(OOA)[3~6],试图从面向对象方法学的高度,探索出一条面向对象的软件开发(Object-Oriented Software Development,简称OOSD)方法,即通过OOA、OOD和OOP将现实世界的空间模型平滑而自然地过渡到面向对象的软件模型,使得软件开发过程与人们认识问题空间的过程保持最大限

OORTST的使用比较简单。尽管OORTST的实用性已经得到了证明,但只是试验性的,作为产品其考虑仍不完全。例如,采用一个面向对象的数据库系统将有利于复杂信息的查询与存储,可以提高OORTST的支持能力。就OORTST的用户而言,对使用实例的面向对象方法和需求可跟踪性问题必须有一定的了解。另外,在系统开发中由专人负责工具的使用也是比较有益的。

参考文献

[1]A. M. Davis, Software requirements: Analysis and specification, Prentice-Hall Inc., 1990

[2]Richard Jordan 等, Streamling the project cycle with object-oriented requirements, OOPSLA'94

[3]J. D. Palmer 等, An integrated environment for requirements engineering, IEEE Software, May 1992

[4]Kenneth S. Rubin 等, Object behavior analysis, CACM, Vol. 35, No. 9, 1992

[5]Klaus Pohl, The three dimension of requirements engineering: A framework and its application, Information Systems, Vol. 19, No. 3, 1994

[6]R. G. Mays 等., PDM: A requirements methodology for software system enhancements, IBM System J. Vol. 24, No. 2

[7]B. Ramesh 等, Issues in the development of a requirements traceability model, Proc. of the IEEE Intl Symposium on Requirements Engineering, San Diego, California, Jan. 4—6, 1993

[8]J. M. Drake 等, Approach and case study of requirement analysis where end users take an active role, IEEE Software, 1993

[9]C3ISE-XD-93计划,西安电子科技大学软件工程研究所

*)本文受863/CIMS高技术项目支持。王斌君 讲师;郝克刚,教授,主要研究方向为软件工程、面向对象等

度的一致，用 OOSD 开发出的软件正确、合理、易理解、易维护且重用性强。用 OOSD 开发软件其关键就是要建立一个全面、合理的面向对象模型，它既能反映问题空间的本质、易于构造，且能被软件空间所接受。本文汇集了这几年来关于面向对象模型的研究现状和各种模型之优、缺点，并给出了作者自己提出的面向对象的三维模型，以飨读者。

二、面向对象模型之现状

建立模型就是对我们要研究的问题空间给出一个完美的抽象。形式地讲，就是"假如 M 可用于回答所有关于 A 的各种问题，则称 M 是 A 的模型。有很多建立模型的手段，如图、文字、表格及严格的数学描述等。建立模型，可使我们对问题进行有效的抽象、建立系统前的测试等，从而降低了求解问题的复杂度。那么，什么是一个好模型呢？我们给出了如下的标准：①能正确地反映问题域的本质，②简单、易理解、易构造。③便于开发小组中各类人员之间的通讯和协作。④允许和易于修改、维护性好。⑤一致的风范。即有简单的一面，也有复杂的一面；既能反映系统的整体性质，也能反映其细节；既能反映问题空间的实质，也能被软件空间所接受。

软件工程在这二、三十年的发展过程中，人们从不同的角度和不同的观点对这一问题进行了探讨。在此领域中，形成了一些能较好反映问题空间的模型，如 DFD、STD、ERD 等。其中，DFD 主要用于描述系统的功能，可用于软件系统的建模；ERD 主要描述系统的静态结构，适于数据库系统的建模；而 STD 能说明系统的动态特性，可用于实时系统的建模。它们分别主要描述了系统的功能、数据和状态。

表1　三种传统模型的比较

	①	②	③	④	⑤
DFD	√	√	√	×	×
STD	√	√	√	×	×
ERD	√	√	√	√	√

由表1可见，这三个模型都不能全面地抽象出问题域的本质，它们都反映了问题空间的某一方面，而且它们都不能保持从问题领域抽象出来的模型与软件空间的一致性，DFD 和 STD 不利于软件的维护。

近几年，特别是88年 Shlaer & Mellor 发表了文[3]以来，出现了很多面向对象的软件开发方法和相应的面向对象模型[1-5]，面向对象的各种方法其目的就是从问题域中抽象出对象、类、对象与类之间的各种关系，如继承关系、组成关系、协作关系等，以及对象与类的动态特性等来描述问题域。但把问题域中这些面向对象的成份都找出来并不构成问题域的面向对象的系统模型，如果应用这些概念和机制使其有机地组合在一起反映系统就构成了各种各样的面向对象的模型。目前，这一领域的研究工作正处在百花齐放、百家争鸣的大发展阶段，已形成一套完整方法的有20多种。为了使大家对这一领域有一个全面的了解，下面以最有代表性的五种方法为例，分析它们各自的优、缺点和存在的问题：

1. Shlaer & Mellor 方法[3]：首先建立信息模型（对象及它们之间的关系），然后，为每个对象建立状态转换图和数据流图。该方法充分反映了 OO 方法从 ER 方法演变的痕迹。其模型简单，但缺少很多信息，如：组成关系等，并且动态和功能描述缺乏系统一级的抽象。该方法是第一个采用 OO 思想建立起来的面向对象模型，目前已很少采用。

2. Booch 方法[1]：通过建立类图（类、类之间的各种静态关系），状态变换图（对象的状态变化可嵌套），对象图（对象、对象之间的关系及可视性），时间图（对象间可能的消息通讯序列），模块图（子系统），处理图（分配不同的处理器），从而构成了一个如下的模型：

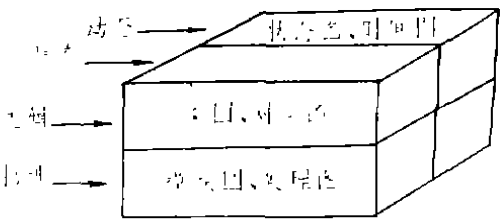

图1　Booch 模型

该方法反映的信息全面，但各种图太多，图之间的关系很复杂，层次不清，不利于应用。

3. Coad & Yourdon 方法[4]：通过标识类 与对象、结构（继承和组成）、属性、服务以及主题（紧密联系的类与对象构成的子系统）构成了如下的面向对象模型：

——主题层
——类与对象层
——结构层
——属性层
——服务层

图2　Coad & Yourdon 模型

该方法实用，相对简单、易构造，但它对系统的动态特征不充分，而且各分层模型放在一个维度上

描述不同的抽象级别是不合理的。

4. Rumbaugh 方法[6]：通过建立对象模型（对象与类，以及它们之间的联系，包括继承和聚合关系），动态模型（事件迹图、状态转换图）和功能模型（DFD图）从三个不同的侧面反映同一个系统的不同特征，各视角模型责任和分工明确，描述的信息全面。与 Shlaer & Mellor 方法相比，它的对象模型更加成熟，反映的信息全面，动态和功能方面也有了系统级的描述，但该方法使用了传统的 DFD 描述系统的功能模型，由于 DFD 不能对系统进行有效的抽象，且它采用的技术（结构化）与其它两个模型采用的技术（面向对象）完全不同，它们之间存在着不确定的转换，该方法也不理想。事实上，很多人用 OMT 开发软件，并不用其功能模型。

5. Wirfs-Brock 方法[2]：通过标识类（类和类的继承层次），以系统的责任为龙头，并分配这些负责到类中，再通过类之间的协约（利用其它类的责任，实现自身的责任，以及为其它类的实现提供责任）构成整个系统。用这种责任驱动的方法得到了一组称之为类卡的文档。它对类与对象之间除继承关系外的其它关系缺乏表达的手段，系统的动态行为也描述得不够，但它用责任和协作很好地描述了系统的功能。

如果我们建立如下评价标准：①模型是否易于开发系统？②模型中反映系统的结构特征是否完整（整体和细节）？③模型中反映系统的动态特征是否完整（整体和细节）？④模型中反映系统的功能特征是否完整（整体和细节）？⑤模型之间是否一致。那么，以上五种模型可总结为表2：

表2 五种典型的 OO 模型的比较

	①	②		③		④		⑤
		整体	细节	整体	细节	整体	细节	
Shlaer & Mellor	√	×	×	×	√	×	√	√
Booch	×	√	√	√	√	√	√	√
Coad & Yourdon	√	√	√	×	√	×	√	√
Rumbaugh	√	√	√	√	√	√	√	×
Wirfs-Brock	√	×	√	×	×	√	√	√

三、3-DOOM 简介

从软件工程多年的研究成果中，我们认为系统的特征可从三个方面加以完整的描述，即数据、功能和状态。在面向对象技术出现以前，人们已在这三个不同的方面进行了成功的探索，并有成熟的模型对此可以描述，如图3所示：

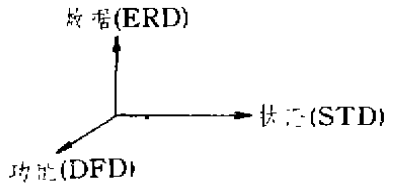

图3 系统的三个维度

它是三种互不相容的技术对同一个问题空间在不同的维度上的建模。通过上一节的分析，我们可以看出，面向对象技术有望用同一种观点，对现实世界从这三个维度上进行建模。这就是我们提出的面向对象的三维模型（Three-Dimensional Object-Oriented Model，简称3-DOOM）[7,8]的基本出发点，即用对象模型、动态模型和协作模型分别描述系统的数据、状态和功能，并在不同的侧面有不同的抽象层次，从而构成了面向对象的三维模型，如图4。

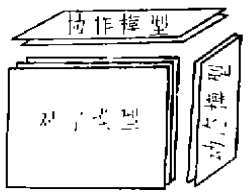

图4 面向对象的三维模型

·对象模型：从系统描述的数据角度看，系统是由类与对象，以及它们之间的各种依赖关系构成的整体（系统层），并通过标识每个对象所具有的属性和操作反映对象的性质（对象层），以及用 PDL 描述操作等（细节层）。

·动态模型：从系统所具有的状态角度看，系统是由许多具有动态行为的对象构成的整体。用事件迹图构成系统层的描述，而每个对象的状态转换图构成对象层的描述。

·协作模型：从系统反映的功能角度看，系统是由相互发送消息的对象相互协作、共同完成了系统的功能。系统层用对象之间的协约关系描述，对象层用接口服务来描述。

该方法主要用责任驱动的 Wirfs-Brock 方法以及文[9,10]的思想，代替 Rumbaugh 方法中的功能模型，并在各模型中引入了抽象层次等演化而来。我们认为，现存的面向对象模型大部分都有一个共同的弱点，那就是主要强调从系统中抽象类与对象，并

标识它们之间的各种关系，即用 bottom-up 的方法构造系统，但没有充分地描述这些类与对象如何协作完成系统的功能。当然，对象模型反映了系统主要的方面，用它构造的系统稳定性好，便于维护，也是进一步领域分析的基础。但我们构造一个软件系统，其首要的目的，就是用它能模拟问题空间功能，并得到一个正确的答案。这两方面是相辅相成的，例如：在对象模型中有一个公司类和雇员类，它们之间有一个雇用关系，而在协作模型中有一个查询该公司有多少雇员的功能，则在公司类中必须有一个相应的操作。显然，对象模型中的雇用关系是这一操作实现的基础（有了这个关系，我们还可以查询其它的功能，如男雇员的人数，中年雇员的人数等）；而这个功能则是我们在对象模型中标识雇用关系之目的的表现，并且，以分解系统的责任（top-down）为线索，容易找到系统中的对象，确定系统的范围（boundary）。先确定系统中的对象和对象所具有的操作，再确定其属性结构，这也反映了封装性和信息隐蔽原则。

另外，我们认为在 OO 中有两种最基本的成份：对象和消息，缺一都不能构成一个完整的系统。一个现实世界就是由发送消息的对象构成的整体。在3-DOOM 中，这两个方面都给了较好的描述。用3-DOOM 开发软件，即分析（构造各视角模型的第一，第二层），设计（构造各视角模型的第二，第三层）和实现，是一个逐步细化和形成化的过程。

最近的一些研究成果[5,11]与我们的观点类似。Eembley 方法[5]是由对象关系模型、对象行为模型、对象交互模型以及一些抽象层次的控制来完成系统的建模，Coleman 等人提出的 Fusion 方法[11]是通过分析阶段的对象模型和接口模型（其中操作模型反映系统的单个功能；生命周期模型反映系统动态行为），设计阶段的对象交互模型、继承图、类描述等描述整个系统。它们的共同点就是用对象的交互模型反映系统的功能，在 Booch 方法的第二版[12]中，也将原时间图变成了对象的交互图。

四、结论

面向对象的软件开发，其关键是建立一个合理的、全面的反映问题域且内部一致的面向对象的模型。本文分析了几个典型的已有面向对象模型，并在此基础上提出了一个理想的面向对象三维模型。进一步的工作，我们要对3-DOOM 进行形成化的探索[13,14]，以确保各视角模型及各开发阶段的一致性、完整性和正确性。

参考文献

[1]G. Booch, Object-Oriented Design with Applications, Redwood City, CA: the Benjamin/cummings publishing company. Inc. ,1991

[2]R. Wirfs-Brock 等. ,Designing Object--Oriented Software, Englewood Cliffs, NJ: Prentice-Hall Inc. ,1990

[3]Shlaer, S. j. Mellor, Object-Oriented System Analysis: Modeling the Word in Data, 同[2], 1988

[4]P. Coad, E. Yourdon, Object-Oriented Analysis, 同[2], 1991

[5] D. W. Embley 等, Object-Oriented System Analysis: A Model-Driven Approach, 同[2], 1992

[6]J. Rumbaugh 等, Object-Oriented Modeling and Design, 同[2], 1991

[7]王斌君、卢安国，面向对象的方法学与 C++语言，陕西师范大学出版社，1995

[8]王斌君、郝克刚、葛玮，面向对象的三维模型，西北大学学报，1995，4

[9]K. S. Rubin, A. Goldberg, Object Behavior Analysis. CACM, Vol. 35, No. 9, 1992

[10]R. Wirfs-Brock 等, Object-Oriented Design: A Responsibility-Driven Approach, OOPSLA'89

[11]D. Coleman 等, Object-Oriented Development: the Fusion Method, 同[2], 1994

[12]G. Booch, Object-Oriented Analysis and Design with Applications, second edition, 同[1], 1994

[13]葛玮、王斌君、郝克刚，一种基于 petri 网的半形式化面向对象的开发方法，西北大学学报，1995. 5(待发)

[14]周继鹏、王斌君、葛玮，面向对象模型的若干代数性质，西北大学学报. 1995. 6(待发)