计算机科学1994 Vol. 21 №. 4

面向对象系统开发方法 ZOOM*)

李文军　李师贤
（中山大学软件研究所　广州510275）

摘　要　本文提出的一种面向对象系统开发方法——ZOOM 方法，以软件系统所作用的现实世界为系统主题，首先构造系统模型，然后在其上建立系统功能。文中介绍了该方法的思想、开发步骤及支持环境的设计。

1. 引言

在面向对象技术研究中，面向对象需求分析方法是比较薄弱的环节，现有面向对象技术与 JSD 方法[3]相结合可望开拓一条新的途径。

ZOOM 方法的基本观点是把软件系统看作是现实世界信息系统的仿真，以获得更有效的问题解[1]。这种方法的基本思想和 JSD 方法一样，也是首先建立现实世界的模型，在此模型的基础上再引入系统功能。但不同的是 JSD 方法进行模型化的工具是顺序进程；在 JSD 方法的初始模型步，根据现实世界的抽象描述使用顺序进程来模拟实体的动作序列；而在 ZOOM 方法中的模型化工具是对象；现实世界中的实体用对象来模拟，实体相应的动作由对象中的操作模拟。

之所以将系统功能建立在系统模型基础上，是因为模型比功能更加稳定。系统功能是以用户的概念和术语来描述的，而这些概念和术语即是系统模型提供的概念和术语，因此当现实发生变化时，模型与功能都将发生变化；但用户的功能需求发生变化时，不一定导致模型变化。

系统模型隐含地定义了系统所能提供的全部功能的集合。系统开发者以模型作为定义功能的上下文，可使功能的定义无二义性，并且模型使用的是用户的术语，有利于沟通开发者与用户。

ZOOM 方法主要开发步骤包括：(1)系统模型步构造现实世界抽象模型，用对象和行为刻划现实世界中实体、动作及动作时序，为提供系统功能构造框架；(2)系统功能步将系统要提供的功能引进系统模型；(3)实现步以面向对象程序设计方法学为指导，选择目标语言编写类文本和例程文本。

2. 系统模型步

使用 ZOOM 方法开发系统的出发点是原始需求说明，据此约束系统模型边界。ZOOM 方法认为系统的主题应该是系统所关心的现实世界，所以首先把注意力放在对现实世界的研究，而不是系统本身，并在此基础上构造现实世界的仿真模型。模型化工作包括：

2.1　抽取对象和行为

对象是现实世界中的人、组织或物体等实体，行为则是现实世界中的一个动作。对象和行为紧密相关：行为总是由一个或多个对象执行或承受的，对象则通过行为以及行为时序表现其特性。对象必须存在系统主题之内，必须执行或承受若干行为，这些行为应具有某种时间顺序；行为则必须是原子的（不可以再划分为子行为），必须发生在某一时刻而不是在某段时间（这样才能区别对象行为和对象状态）。

我们首先构造候选对象清单和候选行为清单，然后根据经验和一定原则筛选真正需要的对象和行为。最后得到的对象和行为分别列成对象清单和行为清单。每个行为要说明其执行或承受的对象、非形式化描述、涉及的数据等。对象则不必说明，因为对象通过它所涉及的行为以及行为的时间顺序来反映其特性。被筛去的对象或行为都应说明被筛去的原因，并且这些对象或行为都应保存在文档之中，因为系统开发文档记录的不仅是系统开发的结果，而且要记录系统开发的历史。

*)南京大学软件研究所国家重点实验室基金资助项目

绘制对象行为关系图作为抽取对象和行为的总结。对象行为关系图可看作是一种扩充实体关系图：方框表示对象，棱形表示行为，每一行为上的箭头由行为的执行者指向行为的承受者。行为涉及的数据以及对象固有的数据都可以作为属性加到对象行为关系图中，分别以椭圆挂在方框或棱形之下。对于小系统可以这样做，但大型系统开发则不适合，因为加进属性会破坏对象行为关系图简明易读的特点，况且这时属性还是不全面的。

对象行为关系图是抽取对象和行为的检查点。如果图中某一对象是分离的，即无行为与另一对象相联系，则说明此对象缺少行为——要么补充此对象的行为，要么删去此对象。如果图中某一行为的执行者或承受者不完备，反映在行为的箭头不能标出，我们应当重新考虑这一行为。

如何从现实世界中抽取我们所关心的对象和行为是一个重要问题。虽然人们不断提出一些启发式的方法，但在这一步很大程度上不得不依赖于开发者的经验，因而保存这些需求说明知识，提供给开发人员重新使用是非常有意义的。ZOOM 方法的系统模型步比较适宜以计算机管理领域知识，建立不同应用领域的可重用软件构件库，构件库中以对象与行为作为重用单位，可支持需求说明一级的软件重用。

2.2 描述行为的时序

我们所关心的现实世界是动态的，每一对象有关的行为都满足一定时间顺序，这在实时控制系统特别突出。因而有必要显式描述行为的时序。

对象行为关系图中每一个对象的行为的时序可使用结构图[3]、状态转换图或正则表达式来描述。这些图或表达式的表达能力只相当于正则语言，对于正则语言不能描述的时间顺序可引入非形式化说明进行补充。

结构图或状态转换图也是一个检查点，我们通过研究一个行为能否很自然地挂在一个对象下，观察一个对象还欠缺什么合理的行为，以及结构图或状态转换图的中间结点能否简单地命名，来检查对象清单、行为清单以及对象行为关系图的合理性。

由结构图或状态转换图规定的对象行为时序将从系统分析一直保持到系统实现，如果系统实现采用 Eiffel 这类支持断言的语言时，这些时序将约束对象之间的客户关系，并反映在类不变式及例程的前、后置断言上。

2.3 定义对象模型

通过对象行为关系图和结构图，我们对所关心的现实世界作了抽象描述。在这一步我们将行为及其所作用的对象聚集为类，并将对象行为关系图转换为对象模型图，对象行为关系图中的联系则反映为对象模型图中的客户关系。对象模型由两方面定义：类文本描述了详细的对象模型；对象模型图给出了对象模型简明而直观的理解，使得系统模型易阅读和交流。

这一步实际上是对现实世界进行抽象与划分。ZOOM 方法采用面向对象的模块划分原则，将现实世界划分为若干对象。对象行为关系图描述了每一行为的主动对象和被动对象，主动对象执行行为，行为作用在被动对象上。所谓行为作用在被动对象上，实质上是行为的执行是在被动对象的内部数据上进行操作。内部数据类似 JSD 方法中状态向量的概念[3]，由于这时不考虑数据的具体表示，也称为数据抽象。根据面向对象的模块划分原则，对象行为关系图中的每一个行为都应聚集于其被动对象之上。

在 ZOOM 方法中，系统模型是通过继承关系和客户关系[6]组织起来的。在系统模型步使用继承机制是为了更有效地组织和分析所获取的关于系统主题的知识。

2.3.1 *类文本* 在一个系统中通常有一些对象具有相同的特征和活动方式，我们把这些对象的描述称为类，而每一个对象则作为该类的一个实例。类的描述是通过对具体实例的研究得到的，类文本包括类名字、简要说明、属性表、继承关系表、客户关系表、关键属性、行为描述等内容。

我们还可以对类中所封装的行为的属性进行分析，从而得到该类的数据抽象。这时数据抽象还是不完备的，在系统功能步将得到进一步完善。

2.3.2 *对象模型图* 对象模型图反映了系统模型和系统结构，它是一个系统的核心，贯穿于系统开发过程。对象模型图将类文本中的重要信息以图形的方式表达出来，起着提纲作用。对象模型图的基本图符包括：类、功能例程、驱动模块、子系统、继承关系、客户关系等。

为开发大型应用系统，ZOOM 方法引入了子系统的概念。一个子系统是类、功能例程、驱动模块和子系统的集合。子系统作为一种视域，可使得一个大的对象系统由实质上的拓扑结构呈现出层次结构，从而帮助开发者更好地理解系统的结构。

2.3.3 *定义驱动模块* 对象模型是系统主题-现实世界的抽象模型，为保证对象模型准确地描述动态的现实世界，对象模型必须接受现实世界的输入信息，使对象模型与现实世界保持同步。

对于每一个行为，我们总是让行为的主动对象提供输入信息。完成这种输入工作的称为驱动模块。一个系统可以只定义一个驱动模块，它负责输入所有有关信息；也可以对每一可能接受输入信息的对象定义一个驱动模块。当有多个驱动模块时，表示现实世界的输入信息是并发的。

定义驱动模块实际上是设计一个输入子系统。现实世界中各个实体动作常常是并发的，对象模型定义的对象间也有一定并发性，我们可通过设计驱动模块决定我们实现的系统是否是并发的。不过这样的设计策略应放在实现步考虑，我们这时只关心哪些对象需从对象模型之外接受输入信息。

驱动模块还要保证输入信息的合理性，所谓合理是相对于前面所作的需求说明而言的，特别是结构图规定的行为时间顺序。我们将在实现步考虑如何保证驱动模块输入信息对于对象模型的合理性，这对于一个强健的系统来说是很重要的。但在这一步，我们可以认为驱动模块传给对象模型的信息是合理的。

至此，我们定义了现实世界的一个仿真模型，这个模型将作为我们提供系统功能的支持框架。当我们在选择现实世界中哪些实体作为系统模型的对象时，就已经隐含地定义了系统模型所支持的全部系统功能的集合；在此集合中的功能可以比较容易地加入到系统模型上，否则将要重新定义系统模型。

3. 系统功能步

在实际应用中，系统模型与系统功能的界限有时是很模糊的。一个区别方法是看它是否产生输出：系统功能总是表述为在一定条件下产生满足某一要求的输出。不同应用领域有不同的技巧说明系统功能，但至少有一个原则必须遵循：系统功能的引进不应该影响系统模型的运行！

开发系统功能时，各功能之间是相对独立的。当有些复杂功能必须基于其他简单功能时，我们仍是独立地说明各个功能，在实现步才考虑其优化。对于每个独立说明的系统功能，我们通常要完成以下工作：

3.1 非形式化说明系统功能

对于每一系统功能，我们首先给出非形式化说明，以便和用户交流。对系统功能的说明应该无二义性、易于理解，最好能使用结构化语言，诸如结构英语、问题说明语言 PSL 和自然语言都可用来说明系统功能。

最典型的系统功能可表述为：

if C happens then output P

这表示在条件 C 发生的情况下应输出结果 P。为使非形式化说明的系统功能可自然地引进到系统模型中，P 应该是能够被系统模型所表达的，而 C 则要么是能够被系统模型所表达的，要么是外界调用系统功能时提供的输入。

所谓能被系统模型所表达是指能用系统模型提供的术语直接或间接地进行描述。这些术语包括系统模型提供的类、对象、数据抽象、行为等。对于复杂系统还可以将常用的模型间接表达的术语汇编成字典。

3.2 将系统功能引进系统模型中

对于一些简单的系统功能，我们只要稍微修改系统模型中对象类的某一操作即可，而不影响对象模型图，这种功能称为嵌入式功能。但在大多情况下，对于每一个系统功能我们引入一个功能例程到系统模型中，即在对象模型图中增加一个功能例程，最后将所有对象模型图合并为一，成为系统分析的结果。功能例程和驱动模块一样，都用例程文本描述。

3.3 回溯

描述系统功能时，我们需要完善在系统模型步没有确定的对象类的属性。有时为了提供一个系统功能，我们需要重新回到系统模型步来考虑是否要扩大系统模型范围或考虑是否增加一个对象类中的内部数据。

所定义的系统模型必须足够丰富才能支持用户所需的全部系统功能。一旦发现某系统功能不能以系统模型提供的概念来表达它，我们应当复审系统模型的定义是否合理，并作相应的修正。

4. 系统实现步

ZOOM 方法将系统开发过程划分为系统分析和系统设计两部分。系统分析过程包括系统模型步和功能步，系统设计过程是 ZOOM 方法的实现步，相应地系统开发人员组织为面向用户的分析人员和面向计算机的技术人员。

ZOOM 方法混淆了设计和实现的界面，因为设计和实现本质上是同样的活动：构造满足某一需求说明的软件。其区别在于抽象层次不同：设计阶段我们忽略某些细节，而实现阶段所有的东西都要完全描述出来。实现步的主要任务有：

4.1 设计与编码的重用

系统模型步和系统功能步是面向用户的，所以将重用设计与编码这一技术问题放在实现步来考虑。ZOOM 方法从两个方面来考虑这个问题：如何重用现有软件构件和如何从系统开发过程中提取可重用软

件构件、这两方面相辅相成。面向对象技术中的继承机制是支持软件重用的有力工具，它将构件的构造技术、修改技术和合成技术融为一体。通过重命名我们可以在语法上修改所继承的特征；通过重定义我们可以在语义上修改所继承的特征。

重用软件在很大程度上需要可重用软件构件库和相应库管理工具的支持。我们已经开发了以代码作为构件的可重用构件库以及库管理系统。

4.2 设计和实现驱动模块

驱动模块的设计实质上是设计了整个软件系统的调度策略，即决定了系统中各对象由外界接受输入并作相应行为的顺序。在实现步，我们将根据系统要运行的目标环境决定系统的运行是联机的、批处理的、并发的还是以上混合的。

驱动模块要保证输入相对于系统需求说明是合理的。使用前、后置断言是一种途径，它可以减少类中例外处理的说明，使类文本更易读。此外 Eiffel 语言中提供的例外恢复机制和 Ada 语言提供的例外处理机制都可以用来方便地实现驱动模块。

4.3 面向对象数据库组织

在开发应用软件系统，特别是数据处理系统过程中，通常需要数据库管理系统的支持。使用 ZOOM 方法最好能有面向对象数据库系统的支持，但是由于条件所限，当前我们采用的做法仍是与关系数据库接口。

4.4 编码工作

最后将以具体程序设计语言实现对象类文本和例程文本，成为一个真正可运行的系统。

5. ZOOM 支撑环境的设计

ZOOM 支撑环境的核心是环境信息库，它负责存放在系统开发过程中产生的各种软件产品和半成品，由 ZOOM 支撑环境提供的软件工具进行查询、加工和存取等管理。支撑环境的设计目标是最大程度地支持 ZOOM 方法，管理和维护系统开发过程中产生的文档，辅助开发阶段之间的平滑过渡，检测文档之间的一致性与完整性等。ZOOM 支撑环境提供了以下软件工具：

5.1 项目规划工具

开发不同的软件系统由不同的项目规划文件对开发过程进行管理。项目规划文件中包含了本项目开发和管理的有关信息。

5.2 对象清单和行为清单编辑器

对象清单和行为清单用于收集、整理和筛选对象以及行为，它们可视为一个规范的二维表，每行表示一个对象或行为。每个对象和行为都包括了名字、别名、是否选中、删除原因等信息。

如果以前曾涉及此领域的开发并已建立此领域的可重用软件构件库，则编辑器可通过库管理工具重用此领域中已经抽取的对象和行为。编辑器还可自动检测行为清单中对象与对象清单的一致性。

5.3 对象行为关系图编辑器

对象行为关系图用于描述对象和行为间关系。编辑器可自动检测图中有无重名、有无孤立对象以及与对象清单、行为清单的一致性与完整性。

5.4 结构图编辑器和状态转换图编辑器

结构图和状态转换图都用于描述作用在某对象上的行为之间的时序关系。结构图编辑器实际上是一个结构树的编辑器。编辑器可自动检测叶结点与行为清单的一致性。

5.5 对象模型图编辑器

对象模型图用于描述系统的结构，即对象之间的关系。编辑器支持多层次子系统之间的切换以及不同层次子系统之间内容的重新划分。用户可在对象模型图上直接浏览各种相应的文本。

编辑器可自动检测对象模型图中子系统划分与对象类文本、功能例程文本和驱动模块文本的一致性和完整性。所谓一致性是指对象模型图中每一个继承和客户关系是否符合相应文本的描述；所谓完整性是指所有文本中描述的每一个继承和客户关系是否在对象模型图中表现出来。在结构化分析方法中也有类似问题，如数据流图分解要保证数据流的平衡。

5.6 类文本与例程文本结构编辑器

结构编辑器主要用于半形式化地编辑对象类文本、例程文本和驱动模块文本。半形式化是指文本必须满足一定描述形式，同时又允许某些自然语言描述。

6. 结束语

面向对象技术侧重于知识的结构方面，结构化方法则侧重于知识的功能方面，ZOOM 方法较好地结合了这两种思维方式；系统模型步考虑的主要是系统主题、现实世界的结构，所以采用面向对象的观点，将现实世界中的实体和动作模型化为系统模型中的对象和行为；而系统功能步考虑的主要是系统所要提供的功能，所以采用结构化方法的观点，将系统功能描述为功能例程。

（下转第24页）

定理证明程序 MGTP(KL1 实现)是法律推理系统的一个基于规则的推理器。它使系统采用一阶逻辑的知识表示，负载平衡优化成功，并行处理能力同 PE 数成正比。

(3)自然语言处理

开发软件工具和语言数据库是为了实现自然语言接口。语言工具箱(LTB)包括自然语言分析器，句子生成器，语言数据库和句法规则词典。

6. 基准的和实验性的并行应用系统。为评估并行推理系统、各种工具和方法，为发展并行程序设计方法、知识表示技术和较高级推理机制，ICOT 努力开拓了知识处理的新应用领域；如 VLSI CAD 系统，遗传信息处理和法律专家系统，博奕系统等。

5. 并行推理系统的评估

1. 对 KL1 的评估　事实证明：工作划分和负载平衡方法是成功的，KL1 语言的规格说明是切实可行的，由于采用自动存储管理机制和自动数据流同步机制，所以 KL1 的并行软件生产率比传统语言的软件生产率高得多。

2. 对 PIMOS 的评估　PIMOS 的功能(其中一些作为 KL1 的函数)对硬件上运行和调试用户程序非常有效。某些特殊工作上的资源管理和执行机制也正如人们的期望，如允许程序员使用 4000 个进程优先级，方便编制各种搜索算法和纯理论计算。

3. 对硬件支持的评估　比较 PIM PE 和一般微处理机的执行速度，得出 1LIPS 近似等价 100IPS，即 500KLIPS 的 PE 相当于 50MIPS 的微处理机。但是，KL1 程序执行特点不同于一般微处理机的基准程序执行特点。ICOT 认为，未来处理机设计应把特征体系结构做为一般用途微处理机标准功能的一部分。

4. 对高级程序设计开销的评估　尽管 KL1 生产率高，但与传统语言程序执行速度相比，KL1 开销不小。有两种直接补偿开销的办法：一是采用简单的子程序调用机制，把 C 语言程序连接到 KL1 程序上。二是改进编译器的优化技术。方法二比方法一更优美。

6. 结论

EGCS 计划是日本的第一个国家计划，其研究开发目标(特别是并行推理系统)已经达到。ICOT 不断公布、推广研究成果，传播 KL1 LP 和 PIMOS，希望它们能被移植到 MIMD 机上，为未来知识处理技术提供一个研究平台。FGCS 计划亦得到世界同仁的宝贵建议和帮助，因此，FGCS 取得的成就是世界范围内从事逻辑程序设计并行处理和相关领域的科研人员辛苦协作的结果。

主要参考文献：Shunichi Uchida，Summary of the Parallel Inference Machine and its Basic Software，Proceedings of the International Conference on Fifth Generation Computer Systems 1992，ICOT

(上接第 64 页)

使用 ZOOM 方法还有利于构造快速原型，通常有以下方法：(1)完成系统模型步工作后，最简单地实现驱动模块，使系统模型可以运行起来。我们可通过支撑工具观察其运行，抽查对象的内部数据，并测试系统模型是否正确地反映了现实世界。(2)对每一系统功能构造原型，使得系统功能在形式地说明后可立即得到演示和测试。(3)最快、最经济地实现系统需求说明，而不必考虑实现的效率问题。

主要参考文献

[1]S. Bailin，An Object-Oriented Requirements Specification Method，CACM.，1989，5，32(5)，pp. 608-623

[2]G. Booch，Object-Oriented Development，IEEE Tran. Softw. Eng.，1986，2，SE12(2)，pp. 211-221

[3]M. Jackson，System Development，Prentice Hall，1982

[4]J. Knudsen，Teaching Object-Oriented Programming is More Than Teaching Object-Oriented Programming Languages，ECOOP'88，LNCS 322，1988，pp. 21-40

[5]B. Meyer，Object-Oriented Software Construction，Prentice Hall，1988

[6]B.Sanden，An Entity-Life Modelling Approach to the Design of Concurrent Software，CACM.，1989，3，32(3)，pp. 330-343

[7]S. Shlaer，An Object-Oriented Approach to Domain Analysis，Softw. Eng. Notes.，1989，7，14(5)，pp. 66-77

[8]仲萃豪等，"应用软件的开发方法"，《计算机科学》，1991，1，pp. 5-15