第3卷第4期
1992年11月

软 件 学 报
JOURNAL OF SOFTWARE

Vol. 3, No. 4
Nov., 1992

论面向对象与逻辑系统的结合

朱海滨

(国防科技大学计算机系,长沙 410073)

ON THE COMBINATION OF OBJECT-ORIENTED AND LOGIC SYSTEMS

Zhu Haibin

(*Department of Computer Science, Changsha Institute of Technology, Changsha* 410073)

Abstract Object-oriented programming and logic programming are the two major branches of the programming field in the 1980's. And relevent software systems were built and applied. Smalltalk and PROLOG system are the excellent examples seperately.

Both above programming methodologies have their advantages and disadvantages, and they have the property of complementing each other. In this paper, both above programming styles are discussed and compared, and a possible way of combining the two kinds of systems is proposed, a prototype system which supports both the programming styles is being formed.

摘要 本文着重讨论了面向对象的基本思想和逻辑程序设计思想的关系,比较了各自的优缺点和相互补充的可能,从新的角度论述了将面向对象程序设计与逻辑程序设计相结合的一种可行途径,最后,提出了一系列具体的实现方法.

§1. 引 言

面向对象的程序设计思想被认为是80年代的结构程序设计,从更高更广的角度研究面向对象的方法已成为90年代的热门课题,它已不仅仅局限于程序设计领域,它已逐步渗透到了软件开发、系统模拟、CAD、图形处理、数据库及知识库的组织与管理、专家系统和体系结构等计算机软件的各个方向,许多专家学者已开始从认识方法论的角度研究这一方法,以该思想为基础的程序设计语言也发展起来,并显示出强大的生命力.

虽然目前对于面向对象的研究还缺乏坚实的理论基础和充分的形式化定义,另外个人对于面向对象的看法也不尽相同,但无论如何,面向对象思想已经受到了广泛的重视,丰富而卓有成效的实践也必将推进面向对象的研究,使之向着更深入和更广泛的方向发展.

本文1990年10月8日收到,1990年12月27日定稿.作者朱海滨,国防科大讲师,1988年硕士毕业于国防科技大学,主要研究领域为面向对象系统,设计及方法学,AI系统及应用.

逻辑程序设计以其强有力的知识表达能力,严密的逻辑基础和有效的实用系统为广大计算机工作者所欣然接受.人们在对其研究的基础上取得了许多可喜的成果,逻辑程序设计语言的最杰出代表 LISP 和 PROLOG 已经广泛应用于专家系统、知识库管理系统和模式识别等人工智能的各个领域.

在面向对象和逻辑程序设计得到不断深入研究的基础上,人们发现,面向对象和逻辑两者之间有着自然的联系,两者各有其优缺点,并具有互补特征,从而自然考虑到了两者的结合.

Concurrent-PROLOG 在 PROLOG 语言基础上增加对象定义机制和消息发送谓词并在此基础上实现了并行机制,但是继承机制仍需用户定义.

Orient/84 是一个在类 Smalltalk 的框架下,将对象状态分为结构知识部分和行为知识部分,并将谓词表达能力加入行为知识部分而形成的一种合成语言,它所基于的基本模型仍然是对象,逻辑表达机制集成在对象之中,作为对象的一种特有能力.

Fooplog 是将面向对象、函数式、逻辑式三者相结合而设计的一种原型语言.在逻辑与面向对象的结合上做了很好的尝试.它在面向对象的概念中只强调了计算的局部性.

本文着重讨论了面向对象的基本思想和逻辑程序设计思想的关系,比较了各自的优缺点和相互补充的可能,最后论述了将面向对象程序设计与逻辑程序设计相结合的一种新型途径.

§ 2. 面向对象程序设计与逻辑程序设计的比较

面向对象的程序设计方法起源于信息隐蔽和抽象数据类型概念,被认为是 80 年代的结构程序设计[6].

逻辑程序设计以一阶谓词逻辑为基础,以符号处理为目标.逻辑程序设计的本质在于定理证明,其基本手段是搜索[7],[8].

从对于计算的基本观点上来看.

面向对象的程序设计(OOP)则将计算过程看作为分类过程加状态变换过程,即将系统逐步划分为相互关联的多个对象并建立这些对象的联系,以引发状态变换,最终完成计算.

逻辑程序设计(LP)将计算过程看作为推演过程,即将具有初始状态的输入在一系列条件约束下,采用推理算法和搜索手段进行匹配、演算的过程,有利于描述启发式知识[3].

从基本特征和实际应用来看.

逻辑程序设计具有很强的知识表达能力和严密的数学基础.逻辑程序设计语言具有内在的推理机制和简洁的规则表示能力.比如,对于家族关系规则可以很容易地用 PROLOG 语言描述如下:

```
grandfather(X, Y):-father(X, Z), father(Z, Y).
```

但它所描述的知识是平坦而缺乏层次的,其语言中也缺乏结构化设施和有效的知识库组织手段.

逻辑程序设计语言不支持继承性,从而使得分类性知识的组织在逻辑语言中不易表达和应用,比如对动物分类树的检索,逻辑语言用户须自行建立描述各分支继承关系的工具以及延继承链检索的过程.

而面向对象程序设计则以知识的层次化组织为其固有性质,从而恰好弥补了逻辑程序设计的弱点.在面向对象程序设计语言中,继承性是语言固有的,用户只需对继承性关系进行简

单的描述即可.

比如,动物世界的分类与面向对象语言的继承描述是一致的.对这种分类的查询也是比较容易的,继承机制为此提供了良好的基础,避免了如PROLOG语言中自行定义的树遍历工作.比如,要问"老虎有毛发吗?"这样一个问题,只要查看虎类是否含"具有毛发"这一性质即可,因为系统内已经通过继承性对其性质进行了组织.

在另一方面,面向对象程序设计还缺乏坚实的理论基础,虽然可以用消息和方法来模拟规则以表示启发式知识,但是比较困难.如前述家族关系中的一条简单规则grand_father,要用到某一个类X中的方法grand_father:来表示,并且还要增加某些类变量或全局变量(PersonArray, PersonArrayLimit)才能完成.

```
grand_father: aPerson
|anotherPerson|
1 to: PersonArrayLimit do: [:i|
anotherPerson<-PersonArray at: i.
(self father: anotherPerson)&(anotherPerson father: aPerson)
    ifTrue: [^ true].   ].
^ false.
```

从这一个例子中也可以看出,推理的非确定性知识的用方法是难以表示的,应用时只能将其转化为确定性知识来表示.并且目前的面向对象系统不含推理机制,在这方面无法与逻辑程序设计语言相比拟.

总的来看,面向对象方法模拟了人类认识问题的较高、较广层次的过程,即分类过程,属于战略性方法;而逻辑方法则更适合于模拟人的逻辑思维,处于人类认识问题的较深层次过程,属于战术性方法,它们的结合才是最完美的.

§3. 面向对象与逻辑的结合

当前对面向对象与逻辑程序设计结合的研究主要放在以逻辑为基础加入面向对象机制的方向上,这是因为国内对逻辑程序设计的研究比较深入,以逻辑为基础做进一步的研究是很自然的.但是笔者认为,对两者结合的研究应取其反方向,即以面向对象为框架,才更符合它们的风格特征.本节讨论的两种风格,其语言蓝本是Smalltalk语言和PROLOG语言.

3.1 系统的主界面和框架应当是面向对象的

这与人们的正常思维方式是相符合的,起初,人们对问题的认识是粗浅的,只有通过不断地细化,方能认识到问题的深处,人们总是先进行战略性分析而后才进行战术性研究.而面向对象的界面恰好为用户提供了这一种思维方式的辅助工具,当然可以成为友好的界面.

逻辑程序设计之所以不能作为界面,因为它比较难懂,并且其形式也不易理解,虽然逻辑表示的简洁性和正确性是公认的,但它的深奥也确实令许多用户望而生畏,所以逻辑程序设计作为主界面是不可取的.

3.2 作为知识表达和推理工具逻辑在系统的核心应占有重要地位

从前面的讨论可知,面向对象方法的知识表示能力在表示启发式知识中受到了限制,另外目前的面向对象系统中也缺乏推理能力,使得面向对象系统在知识工程领域中的应用受到了很大限制,作为对面向对象系统功能的补充,理应将逻辑程序设计的核心特性加入到面向对象

的系统中去.

3.3 合成系统的建立

Smalltalk 语言和 PROLOG 语言分别是面向对象和逻辑程序设计语言的杰出代表,并且我们已经有了对 Smalltalk 语言及 PROLOG 语言实现的成功经验.另外我们已经开始在 SUN 工作站上进行了面向对象环境的开发,所以以这两种语言作为合成系统的基础即是合理的又是可行的.

从我们对逻辑程序设计的基本认识入手,在面向对象的系统中加入逻辑程序设计机制是结合两种风格的主要手段.依我们的观点,逻辑的基本要素在于搜索和匹配.下面几个问题的处理是支持逻辑程序设计所必需的.

第一:关于事实存储

(1)在系统存储器中开辟事实库空间,这一点很容易做到,因为对象的状态恰好可以存放关于该对象的事实.另外,为了适于搜索和匹配,我们可以使每一个方法对应设立一个实例变量,如果实例变量中存放了相应的对象(可以有多个同类对象,如一个病人可以有几个症状)表示这是一条(或几条)确定的事实.在搜索时,若实例变量中有确定对象,则可立即回答,否则需要用相应的方法去求解.

(2)加入系统自动增加事实的功能,这需要系统在进行匹配搜索,结果为真时,自动将与该方法相应的实例变量置为求得的对象.

(3)若需要人工增加事实,只要在创建对象时加以描述即可.

(4)在系统核心机制(对象存储管理)上加入搜索机制,由询问功能引发,在面向对象系统中这一搜索功能可以从面向对象系统中的全局名表开始搜索和匹配,就可完成对用户询问的回答.另外,对于由同一名字的规则所涉及的变元不同而产生的不确定性问题,在面向对象系统中,规则都是对应于某类对象的,从而,这一不确定性被消化了.

第二:关于启发式知识的表示

(1)用消息——方法机制描述规则,这种描述方法更为直接,并且保持了逻辑程序中过程调用次序的不确定性.比如:表示某人该吃什么药的谓词 should_take(Person, Drug);可用 Smalltalk 中的类 Person 中的方法 shouldTake:来表示,它带有一个变元 aDrug.

PROLOG 表示:

```
should_take(Person, Drug):-has_complained(Person, Symtom),
                           suppresses(Drug, Symtom),
                           not(is_unsuitable_for(Drug, Person).
```

Smalltalk 表示:

```
shouldTake: aDrug
|aSymtom|
^ (self hasComplained: aSymtom) & (aDrug suppresses: aSymtom)
  & ((aDrug is_Unsuitable_For: self) not).
```

(2)元级知识可以利用面向对象系统的一类对象来表示.对于诸如 how,why,whynot,solve 等规则的描述基本同 PROLOG 一致,只是要求 PROLOG 中不确定的几条规则要分别对应于不同的类.

第三:关于搜索和推理

(1)在系统解释机制上加入变元匹配机制,这一过程在搜索中进行,当解释到一个发送指令(抽象机指令)时,判断变元是否为确定对象,若是,则正常执行;否则,查找实例状态,确定(可以询问用户)相应对象.注意,这里的匹配与PROLOG语言中的匹配有所不同,PROLOG语言中的匹配是变元与表或符号串的结合,在此是对象与变元的结合,保持了逻辑程序中的入/出变元的不确定性.

(2)增加系统的回溯机制,即当系统匹配某一方法返回值为假时需要重新查询知识库,另行匹配其它对象.这一工作包括两个方面,一方面是在解释器中增加重新查询引导机制;另一方面是在实例对象中增加访问标志,以免重复访问同一对象.

第四:关于询问

(1)在系统界面上加入可供用户提问的询问功能,在多窗口界面上相当于增加一条菜单命令.

(2)增加描述系统提问的机制,这一机制用于专家系统设计者说明某一规则是可询问用户的,这一功能是建立专家系统所必不可少的,我们可以为每个对象的每个实例变量配备一个固定的域叫做askable,若askable为真,则该实例对应的方法是可询问的.

有了以上的机制,前面所提到的家族关系规则的表示就很容易了.

```
isGrandFatherOf: aPerson
|anotherPerson|
^ (self isFatherOf: anotherPerson)
  & (anotherPerson isFatherOf: aPerson).
```

§4. 对合成系统的基本评价

基于以上的讨论,合成系统具有以下特性.

1. 保持面向对象的特征:(1)模块性;(2)封装功能;(3)继承性;(4)动态连接;(5)易维护性;(6)增量型设计;(7)分类知识的有效表示.

2. 增加了逻辑程序设计的特性:(1)启发式知识的有效表示;(2)逻辑推理能力.

3. 具备可以进行多种风格程序设计的特点:可以根据应用的具体需要选择不同的程序设计风格.

另一方面,以上的合成系统也有它的不足之处.其一,该合成系统在面向对象系统的基础上建立的,其运行效率可能比纯粹的面向对象系统更低,因此,该系统可能不会立即为用户很快接受.这一问题将随着面向对象系统效率的提高和实现机制的进一步研究而得到逐步的解决.其二,该合成系统在语法和界面上完全类似于面向对象程序设计,从而使逻辑程序设计清晰、简洁的特点受到了一定影响,可能对于熟悉逻辑程序设计的研究人员有不适之感.但是,这一问题对于一般用户影响不大,正如前面讨论的,面向对象作为用户界面是可取的.

总之,该合成系统的提出为人工智能系统的建立提供更好的工具,对于人工智能系统的研究是具有促进意义的.

§5. 结束语

以上重点从方法和应用上比较了面向对象与逻辑程序设计风格,并从新的角度给出了一

个合成系统的雏型，力图在两者的结合方面给出一个较为全面的认识，并想借此文能够起到抛砖引玉的作用，引起研究者更广泛的兴趣.

在本文的形成过程中，得到过高洪奎副教授的指教，与博士生胡子昂进行过有益的讨论，在此谨向他们表示感谢.

参考文献

[1]Adele Goldberg and David Robson, Smalltalk-80: The Language and Its Implementation, Addison-Wesley, Reading, Mass., 1983.

[2]Geoffery A. Pascoe, "The Elements of Object-Oriented Programming," BYTE, Feb., 1986.

[3]P. Wegner, "Learning the Language", BYTE, March, 1989, 245-253.

[4]朱海滨等，"Smalltalk-80 虚拟机在 VAX/VMS 上的实现"，小型微型计算机系统，(10)，1989.

[5]朱海滨，陈火旺，"从 Smalltalk 语言的结构看'软插件'的形成"，计算机科学，(3)，1990.

[6]朱海滨，胡运发，"面向对象方法学的研究"，计算机科学，(5)，1990.

[7]胡运发，高洪奎，《人工智能系统——原理与设计》，国防科技大学出版社，1988.8.

[8]刘风歧，陈跃新，《逻辑程序设计原理和方法》，国防科技大学出版社，1987.10.

第五届全国分布计算系统与固件工程学术会议纪要

本届会议由中国计算机学会分布计算系统与固件工程专业委员会主办，吉林大学计算机系承办，鞠九滨教授(专委会副主任)主持，于 1992 年 7 月 2 日至 5 日在长春市清华宾馆召开.

来自全国 30 个单位的代表 58 人出席了会议. 其中高级职称的代表 25 人，中级职称代表 16 人，研究生 13 人，其他代表 4 人.

经录取、评审后收入会议论文集的论文共 39 篇，论文分为下列几个类型：分布式与网络操作系统，语言/工具/编程，分布/并行结构与并行处理，网络设计与应用，分布式人工智能，固件工程，以及算法等七大类. 这些论文基本上反映了我国近三年来有关分布/并行结构与并行处理、分布式和网络操作系统、网络设计与应用、固件工程几个方面在国家攻关项目，有关科研战线和实践应用的成果与贡献.

五届学术会议在会风上达到了新的水准. 全体代表认真、热情和真诚地做报告，做发言，提问题，进行了热烈的讨论. 代表们心情舒畅、相互交流、共同提高，没有逃会现象.

会议代表和专家对东道主吉林大学计算机系和主持人鞠九滨教授精心、成功地承办五届学术年会表示衷心感谢，对支持五届学术年会和专委会的单位表示衷心感谢.