第9卷 第4期 计 算 物 理 Vol. 9, No. 4
1992年12月 CHINESE JOURNAL OF COMPUTATIONAL PHYSICS Dec., 1992

面向对象的程序设计

张淑誉 郝柏林
(中国科学院物理研究所) (中国科学院理论物理研究所，北京 100080)

摘 要 面向对象的程序设计已经变得同七十年代的结构程序设计一样重要。本文以多窗口管理系统，地震活动性分析系统和常微方程定性分析系统为例，说明OOP与计算物理的关系，并简单介绍了OOP的主要工具C++语言。

关键词 对象 程序设计 C++

一、引 言

当前计算技术的迅速进步，使软件发展远远落后于硬件。有人估计，在整个九十年代还缺少一千万行软件。这主要是指系统软件。在科学计算方面，也面临着从"程序块加科学库"的模式，转向创造人与计算机交互关系更为密切的计算和研究环境。从程序设计思想上讲，人们正在从面向过程的程序设计(Procedure-Oriented Programming，简称POP)，转变到面向对象的程序设计(Object-Oriented Programming，简称OOP)。这个转变过程在八十年代中期已经开始，现在到了计算物理工作者也应当了解和适应的时候。我们愿在本文中介绍一些自己的学习心得。

二、OOP的基本思想

为了掌握OOP的基本思想，应当先回顾一下POP的概念。我们大家从五十年代后期以来所熟悉的使用各种语言的程序设计，不论是ALGOL、FORTRAN、BASIC还是C，都属于POP的范畴。一个程序由许多过程(函数，子程序)组成，常用的过程进入许多用户分享的科学计算库，每个过程的实现，通常基于结构程序设计思想，采用顺序结构，选择结构(分枝)和重复结构(循环)来完成。在POP中有各种数据结构，但数据结构与过程的关系相当自由。从一段程序最前面的数组说明

```
INTEGER  A(100)
```

看不出来对数组A要作什么事情。运行一个POP程序，基本上就是按事先设计好的顺序，依次调用种种过程，过程的调用顺序也可以在运行中作适当改动，但这都要事先在程序中设计好，采用判断或"中断"的方式实现。程序员不能灵机一动，就在运行过程中随意改变各个过程的组织方式。

在OOP中，数据结构都是具体的，形式上没有规定过程与数据结构的关系，这种"自由"度，反而使程序的运行方式变得相当不自由。

OOP的基本思想是数据结构抽象化。在定义数据结构时，就要作好两个方面的规定。第一，数据结构的性质、状态或属性，这在POP中已经有一些；第二，对此数据结构可能实行的操作，凡是没有规定的操作以后都不许使用，这是OOP所特有的。这样的数据结构称为一个对象(Object)，对象是带有操作的数据结构，或者说由操作刻划的数据结构。从形式上看，数据结构与操作的关系不自由了，然而这些对象却可以提供相当自由的程序运行环境。程序设计就是分别规定各种对象，把对象在调度程序中进行登记，而程序的运行就是动态地激活和调用各种对象。对象的调用顺序和同时激活的对象数目都相当自由。

三、三个实例

为了具体说明对象这个概念，我们考察三个实例。

第一个实例是多窗口管理系统。现在很多操作系统的用户接口，都使用多窗口结构。它们实际上是基于OOP概念。这里的对象是窗口。它的性质可能包括：窗口尺寸，是文字还是图形窗口，是否要上下左右活动，窗口内容是否要保留和自动恢复等等。允许在窗口这个对象上实行的操作可能有：建立或取消窗口、移运位置、改变大小、打开、关闭、退出等等。

事实上，窗口是一大类对象。它规定了各种窗口所共有的性质和操作。然后可以在这类对象之下再说明子类，规定具有特定的附加性质和操作的窗口。然而，在用户实际上建立和打开之前，窗口只是一种抽象的

1992年6月20日收到来稿。

数据结构。通常用户在工作过程中，按不能事先规定的顺序，自由地建立、打开、关闭、移动或退出若干窗口，改变它们的尺寸，在各个窗口中运行不同的作业。简言之，多窗口结构提供了形象的多道并行环境。

作为第二个实例，我们设想一个用于研究的中国地震震中分布图。它有两类基本对象。第一类对象是用于实现国界、省界、河流、邻国边界等线条的地理对象。其性质有线型、线宽、颜色、数据文件名称等，而它们的共同操作是画出和删除。使用 OOP 概念编写的地图程序，可以在工作过程中用不同的颜色和线型绘出、删除和再绘出各种地理成份。这些地理对象又总是在窗口之内运行，因此很容易放大、缩小、以及实现左右上下滚动等等。

第二类对象是地震震中，它们的性质可以包括震级范围、时间跨度、空间坐标等等，而除了画出和删除这两个基本操作外，还可以包括对所选出的地震数据进行某种处理的操作，以及把处理结果另开窗口加以显示的操作等等。

上述地震震中分布程序在运行时不是单纯显示已成定局的计算结果，而是提供了分析地震活动性的研究环境。用人机对话的方式在这一环境中反复尝试，才能逐步得到研究结论。

第三个实例和计算物理关系更为密切，是借助高精度数值计算对常微分方程进行定性研究的一个工作环境[1]。它主要由两类对象组成：

第一类对象是积分曲线族。作为性质，人们可以规定曲线的数目，例如 100；曲线的颜色、点数、用点线还是实线绘出，以及积分步长等等。当然，一条重要性质是说明微分方程组右端函数的文件名字，可以用于此类对象的操作包括：给定和改变参数，以步进或连续方式开始计算，停止计算、保存和提取已算出的数据等等。

第二类对象是一些作为积分初值流形的几何标志，例如直线段、圆或圆弧。它们的性质有尺寸、位置、颜色等等，而作用于这类对象的操作包括建立、取消、移动、放大、缩小等等。

建立了以上两类抽象的数据结构后，就有了一个对微分方程进行定性研究的环境。用户开始工作时，可以给定一个大圆作初值流型，然后激活一个积分曲线族对象，要求从分布在大圆上的 100 个初值点开始积分，并以点线形式绘出积分曲线。这样很容易发现相空间中多个解共存以及奇异点等特殊情形。用户可以令第一个对象暂停计算，再在相空间有兴趣的部位激活新的几何标志和积分曲线族，进行更细致的局部研究。如果所用的计算机足够快，可以同时计算出多组积分曲线，把它们用不同的颜色显示出来。使用这样的系统，很容易发现吸引子共存，寻找奇异点，在奇异点附近构造稳定和不稳定流形，以及研究从一个吸引域逃逸到另一个吸引子的过程等等。

上面这三个实例，都与图形显示有关。这并非偶然，却又不是一种限制。例如，我们前几年实现的一个群论知识工程系统[2]，从原则上讲，就很适宜用OOP 概念改写，而它可能根本不用图形。

四、历史概述

面向对象的程序设计，实际上已经有大约三十年的历史。可以说，它始自 1964 年一位瑞典程序员设计的 Simula 语言，其中已经用到对象这个概念。到了七十年代，在施乐公司的 Palo Alto 研究中心（PARC），人们设计了一种 OOP 语言 Smolltalk 72。不过，它在很长时间里没有商品化。八十年代，Smalltalk的设计人之一另立公司，发展出Smalltalk80。它从 1988 年起成为商品。

OOP 真正热起来是在八十年代后期。1982 年就有人写文章推测，八十年代的 OOP 会像七十年代的结构程序设计一样"走红"。事实上，目前在计算机行业中已经出现了一大批以大写 OO 开始的缩写字。例如 OOP、OOM（管理）、OOD（设计）、OODBMS（数据库管理系统）。另外，还建立了名为 OMG 的组织，着手对 OOP 进行标准化。现在已经有了期刊 JOOP 和 OOT、The C++Report 等快讯。据说，还要出版一种杂志 Object Magazine。

五、OOP 用的语言

前面提到的 Smalltalk 是专门用于OOP 的语言，它在概念上完备，但执行效率较低。另外还有 Common Lisp 语言的 OOP 扩充，称为 CLOS。它在概念上更完整，但也失之效率过低。我们没有使用 Smalltalk 和 CLOS 的经验，在此就不多讲。

对 OOP 的发展起了很大促进作用的，是 C 语言的 OOP 扩充，即C++（它的 GNU 版本称为 g++）。这是一个简短、高效、实用的语言，但是并没有实现 OOP 的全部理想。目前 OOP 和 C++经常被人们相提并论。

同 C 语言一样，C++也诞生在贝尔实验室。1980 年出现了"带类的 C 语言"（C with Class）。1985 年它开始提供给公众使用。该语言的设计者 B. Strouptrup 在 1986 年出版的《C++ 程序设计》一书很维

阅读。好在后来有人帮助他加了注释和实例[3]。目前手册[3]已被 ANSI 选为 C++ 标准化的基础。

我们只在这里介绍 C++ 中的 Class 概念。它在形式上很像 C 语言中的 Struct，但是增加了对操作的说明。例如：

```
Class Point {
        int px , py;
      Public:
        Void Set_pt (int x, int y) {px = x; py = y; }
        Void Offset_pt (int x, int y) {px+ = x; py+ = y; }
        Viod Draw_pt (int, int);
        Void Erase_pt (int, int);
            }
```

这样就说明了一个带操作的数据结构，其中有些操作已在同一行之内定义，有些操作只说明了名子和参数类型，需要另外单独给出定义，如

```
Void Point::Draw_pt (int x, int y);
    {.....              }
```

今后，Point 就可以当作类型说明，用来定义一个或多个同类的具体对象。例如，用语句

```
Point pt, pta, ptb;
```

说明了三个对象。此后可在程序中写

```
pt. Set_pt (10, 10);
pta. Set_pt (-100, -100);
ptb. Set_pt (0, 50);
```

等等。

应当特别指出，对于熟悉 C 语言的程序员，C++ 是很容易掌握的[4]。

最后，我们提一下使用 OOP 概念的程序总结构：

1. 说明一批对象（性质加操作），并向调度程序登记；
2. 说明一批调度程序知道的输入输出事件（键盘、游标、鼠标等等），以便激活或取消对象，向对象提供信息；
3. 在调度程序控制下运行。

今后在建立各种计算物理环境，特别是图形显示起关键作用的环境时，OOP 和 C++ 一定会发挥更大的作用。我们自己也是刚刚起步，希望以后与有志于此道者分享经验。

参　考　文　献

[1] H. B. Stewart, CHAOS－Visual Dynamics Software for IRIS Workstations, vet. 2.2, Brookhaven National Laboratory, 1989.

[2] Gao Junming (高俊铭), Cui Zhan (崔湛), Hao Bai Lin (郝柏林), The Konwledge－based System GRAPE and Its Application, *Computer physics Reports*, **12** (1990), 289－381.

[3] M. Ellis, B. Stroupstrup, The Agnotated C++ Reference Manual, Addison－Wesley, 1990.

[4] S. Hekmatpour, C++, A Guide for C Programmers, prentice Hall, 1990.

OBJECT－ORIENTED PROGRAMMING

ZHANG Shu－yu

The Institute of physics, Academia Sinica, 100080

HAO Bai－lin

The Institute of Theoretical Physics, Academia Sinica, 100080

ABSTRACT OOP is becoming as important as structured programming in the 1970s. Relavance of OOP to computational physics is discussed on the examples of multiwindow systems, a seismicity－analyzing system, and a system for qualitative study of ODE's. The C++ language as a main tool for OOP is also briefly touched.

KEY WORDS object programming C++.