计算机科学2004Vol. 31№. 3

# FORTRAN 2000与C++面向对象机制比较*)

黄文伟　徐宝文

（东南大学计算机科学与工程系　南京210096）（江苏省软件质量研究所　南京210096）

**摘　要**　本文在简单讨论面向对象方法的基本机制的基础上，比较分析了FORTRAN 2000与C++语言对面向对象机制的支持，涉及对象、继承、多态性、操作以及可访问性控制等多个方面。

**关键词**　面向对象程序设计方法，FORTRAN 2000，C++

## Comparison of FORTRAN 2000 and C++ in Object Oriented Facilities

HUANG Wen-Wei　XU Bao-Wen

(Dept. of Computer Science &Engineering, Southeast University, Nanjing Jiangsu 210096)

(Jiangsu Institute of Software Quality, Nanjing 210096, China)

**Abstract**　Based on discussing the object-oriented programming techniques, this paper compares in detail the object-oriented facilities supported in FORTRAN 2000 and C++, including object, inheritance, polymorphism, operation, and control of access, etc.

**Keywords**　Object oriented programming technique, FORTRAN 2000, C++

## 1　引言

面向对象方法是一种运用对象、类、继承、封装、消息、多态性等概念来构造系统的软件开发方法。它不仅是一些具体的软件开发技术与策略，而且是一整套关于如何看待软件系统与现实世界的关系、用什么观点来研究问题并进行问题求解以及如何进行系统构造的软件方法学。它在支持软件重用，提高软件质量、软件生产率等方面起着重要的作用。

FORTRAN语言是世界上广泛流行的、最适用于数值计算的一种计算机语言，也是世界上最早出现的高级程序设计语言之一。随着面向对象方法的普及，FORTRAN 90中增加了面向对象的机制，而FORTRAN 2000则在FORTRAN 90与95的基础上又对其进行了改进并增加了一些新的特性。C++是出现较早且现在最为流行的面向对象程序设计语言之一。

本文主要对这两种语言的面向对象机制进行探讨。

## 2　FORTRAN 2000与C++的面向对象机制比较

面向对象程序设计语言的三个基本特征是封装性、继承性和多态性。C++和FORTRAN 2000中都提供了相应的语法成分来保证和支持这三个基本特征，但它们在对面向对象程序设计方法的实现方式上有许多不同。

### 2.1　类和对象

对象是反映客观事物属性及行为特征的描述。类描述了一组有相同特征（数据成员）和相同行为（成员函数）的对象。类本质上是一种数据类型，它是程序员为了解决具体问题而创建的。C++与FORTRAN 2000在类型的定义和类对象之间的共享机制上都有很大的不同。

在C++中，类是通过关键字class来定义的。如图1(a)所示，由class定义的类，不仅表示一种新的类型的产生，也表示了数据与操作的封装。它实现了类的外部特征与类的内部实现的隐藏和隔离，符合面向对象机制中的封装特性。在FORTRAN 2000中，新类型通过关键字type定义。如图1(b)所示，由type定义的类偏向于数据的封装。虽然我们也可以在type定义中增加过程指针或类型绑定过程来表示操作，但这些过程的实现通常在模块中进行。模块(module)是一种程序单元但不能被直接执行，它可以包含导出类型和数据以及操作这些数据的过程和函数。FORTRAN 2000中可通过关键字type和module结合使用来实现面向对象中的类的概念。FORTRAN 2000共享数据是存储在模块中的，以关键字save来表示该数据是共享的。一个具有save属性的实体在一个模块中，保持它的结合状态、分配状态、定义状态和值。当在模块中显式初始化一个实体时，该实体就具有了save属性。而在派生类型定义时，由于类型内部不能使用关键字save，因此FORTRAN 2000的同类型对象之间不能共享数据。在C++中，类对象之间共享数据是通过静态数据成员来实现的。不管类创建了多少个对象，其静态数据成员只有一个副本，这个副本被所有(属于这个类的)对象共享。

```
class c_example {
   private:
int a;
   public:
void operate();
}
c_example example;
```

(a)C++中类的定义及使用

```
module f_example
```

*)本文部分得到国家自然科学基金项目、国防"九五"及"十五"重点预研项目、国防武器装备预研基金项目、江苏省自然科学基金项目、江苏省科技攻关项目、武汉大学软件工程国家重点实验室开放基金项目、江苏省计算机信息处理技术重点实验室开放基金项目（苏州大学）的资助。**黄文伟**　硕士生，主要研究领域为软件工程、程序设计语言；**徐宝文**　教授，博士生导师，主要从事程序设计语言、软件工程、并行与网络软件技术等方面的教学与科研工作。

```
type f_type
    integer,private:: a
    procedure::operate=>module_op
end f_type
contains
    subroutine module_op
    !与(a)中 c_example::operate 的实现相同
    end subroutine module_op
end module f_example
use f_example
type(f_type)example
```

(b)FORTRAN 2000中类的定义及使用

图1 C++与FORTRAN 2000中类的定义及使用的比较

### 2.2 类型扩展机制

继承是一种联结类的层次模型，允许和鼓励类的重用，它提供了一种明确表述共性的方法，大大增强了程序的重用性和易维护性。

在C++中，由class定义的类结构具有可继承性，所创建的新类可直接从该类中派生，并继承除构造函数和析构函数以外的所有成员。在FORTRAN 2000中，当由type定义一个新类型时，在缺省的情况下该类型不具有继承性。若想让其成为基类则必须在定义该类型时增加extensible属性，从该基类派生的所有类型也具有可继承性。在FORTRAN 2000中还可通过关键字non_overridable来防止基类中的过程在子类中被重定义。在C++中，并未提供相应的机制来阻止重定义的发生。FORTRAN 2000与C++在继承机制上的最大区别是，C++支持多重继承，而FORTRAN 2000只支持单继承。在FORTRAN 2000中也未提供其它机制(例如Java中的接口)来实现多重继承的功能。

### 2.3 多态性的实现

多态性是指允许不同类的对象对同一消息作出响应。多态性可以用"一个对外接口，多个内在实现的形式"来表示。

在C++中，实现多态性是通过晚捆绑(late binding)来实现的。当一个成员函数以关键字virtual修饰时表示希望该函数具有晚捆绑的灵活性。要实现多态性，对象还要通过指针或引用的方式来声明。在FORTRAN 2000中，通过关键字class来实现多态性。当声明一个对象时，用class关键字替代type关键字，本质上是声明了一个类域对象。这样就可以把应用于某个给定类域的代码应用于该类域中的每个类型对象。

这两种语言在多态性实现上主要有以下两点区别：

(1)C++以对象的引用或指针的方式实现动态绑定，而FORTRAN 2000中却通过类域类型来实现。在FORTRAN 2000中，当看到关键字class出现时，我们便知道程序为了实现多态性而声明了一个类域类型，这样的程序可读性强，便于其他程序员理解。

(2)在FORTRAN 2000中，对类域对象的操作都是动态绑定的。而在C++中，除了要对对象进行动态绑定外，还要把函数声明为虚函数，这样就给实现多态性增加了复杂度。因为声明虚函数时，不仅要占用虚拟表的空间，而且晚捆绑使程序的运行效率也降低了。为了提高程序的运行效率和节省运行空间，在设计类时我们并没有把一些不需要晚绑定的成员函数设为虚拟的，以后当要增加新的动态绑定的需求时，我们要重新修改基类。

### 2.4 类中成员方法的定义及使用

C++符合面向对象程序设计语言的封装机制，它把操作和数据绑定在一起组成了类，通过特殊的形式调用对象的方法。在C++中，每个类结构只定义一种类型，在类中的每个非静态方法中都有一个隐含的参数。当通过访问操作符"."访问该方法时，把与该方法绑定的对象作为第一个参数。如图2(a)所示，声明了一个对象example，当通过example.operate()访问example中的方法时，编译程序就自动把对象example作为operate()方法的第一个参数来使用。由于静态方法是被所有同一类型的对象共享的，因此静态方法不需要对象作为隐含的参数。当调用静态方法时，也就不需要再通过对象名来调用，只要通过类名就可以调用。

在FORTRAN 2000中可以通过过程指针成员或命名类型绑定过程来实现操作与类型的绑定。在缺省的情况下，以上两个类型成员过程在定义时有一个pass属性，该属性指明当调用该成员过程时把被调用的对象作为隐含的第一个参数传递给该过程。该机制与C++中调用非静态成员方法的实现机制相同。我们还可以显式地定义一个pass属性，并在定义该pass属性时指定一个参数名，该参数名是用来在成员过程调用时与被调用对象绑定的。在定义成员过程时通过使用关键字nopass来指明调用成员过程时不需要把对象绑定到该过程参数中，这与调用C++中的静态成员机制相似，都是为了实现成员过程在同类型对象间共享。但C++中的静态成员方法可以通过类名来调用，我们可直观地看出该方法是与类型相关的，这就提高了可读性。而在FORTRAN 2000中，我们要调用类型中的成员过程必须先声明一个类型的实例，通过对象来调用成员过程。

在FORTRAN 2000中，过程是定义在模块中的，在导入模块后，我们可直接调用模块中的过程而不用再通过类型中的过程指针成员或类型绑定的过程来调用。因此在FORTRAN 2000中类型和操作可以分开，操作作用于对象，调用对象的操作和调用其它操作形式是相同的。如图2(b)所示，FORTRAN 2000提供了两种访问对象操作的方法。一种是通过对象访问操作符"."进行操作，这种方式与C++的访问方式相同，体现了操作与类型的紧密结合，符合面向对象的机制。另一种是直接调用模块中提供的过程，把对象作为实参传进模块过程中(与Ada的调用机制相似)。

### 2.5 可访问性控制

C++通过访问控制符private、protected和public将类成员划分为：该类成员不能被外界访问、只允许子类访问该类成员以及该类成员可以被外界访问三种访问控制形式。C++的类成员在默认情况下为私有的(private)，只有当类构建者希望对外提供类的某个特性时才用public来显式声明。这体现了面向对象机制中的封装性。

FORTRAN 2000中只有public和private两种访问控制符。public表示该模块成员在该模块外的其它程序块中也可以被访问；private表示该模块成员在该模块外的其它程序块中是不能被访问的。FORTRAN 2000中也有关键字protected，但含义与C++中的不同。在FORTRAN 2000中，protected属性应用于模块中的变量或指针，它表示变量的值或指针的状态只可以在模块中被改变。如果对象具有protected属性，那么它所有子类型对象也是被保护的。使用protected属性可防止模块中的实体在模块外被改变其状态。FORTRAN 2000的类成员在缺省的情况下为公开的(public)，这并不符合面向对象的封装特性。C++中有"友元"的机制，它提供了一种途径使得类以外的对象或函数能够访问类中的私有成员。在FORTRAN 2000中并没有提供相应的机制，若某个过程或对象需要访问类中的私有成员，则可以把

该函数和对象定义在该类所在的模块中，因为 FORTRAN 2000的访问机制是对应于模块而言的。

```
class c_example {
  public:
    void operate();
}
...
c_example example;
example.operate();
```

(a)C++对成员方法的调用

```
module f_example
  type f_type
       procedure,pass(are)::operate=>op
  end type f_type
  interface operate
       module procedure op
  end interface
  contains
     subroutine op(arg)
        type(f_type),intent(in)::arg
     end subroutine op
  end module f_example
  ...
  use f_example
  type(f_type)example
  !第一种调用成员方法的方式
  call operate(example)
  !第二种调用成员方法的方式
  call example.operate
```

(b)FORTRAN 2000对成员方法的调用

图2 C++与 FORTRAN 2000类操作调用的比较

**结束语** FORTRAN 语言是一门历史悠久的语言，有很强的数值计算能力，它曾经在科学计算、工程问题和企事业管理中发挥巨大的作用。但是今年来随着面向对象技术的流行，越来越多的人认为现在最为流行的面向对象语言 C++有可能替代 FORTRAN。本文在此环境下向人们介绍了 FORTRAN 2000的面向对象机制，并把它与 C++语言的面向对象机制作了比较。从中我们可以看出，FORTRAN 2000已经具有了一定的面向对象特征，基本上可以从面向对象的角度来对问题进行分析并编程实现。由于 FORTRAN 语言在面向对象机制上起步较晚，FORTRAN 2000与 C++语言相比其面向对象机制还不完善。但 FORTRAN 语言凭借其在数值计算上的高效性和稳定性，它在工程应用中的地位还是无可替代的。如何完善 FORTRAN 语言的面向对象机制同时又保持其在数值计算上的优势是 FORTRAN 语言今后发展的方向。

## 参 考 文 献


1 Reid J. The New Feature of FORTRAN 2000. ISO/IEC JTC1/SC22/WG5 N1495

2 ISO/IEC 1539. WORKING DRAFT,J3/02-007R3,Sep. 2002

3 Liang Xianzhong, Wang Zhenyu. Ada-based Support for Abstraction, Encapsulationg and Unit Hierarchy. In: Proc. of TRI-Ada'91,San Jose. New York: ACM Press ,Oct. 1991

4 Booch G. Object-Oriented Development. IEEE Trans,1986,SE-12(2)

5 Sheidewitz E. Object-Oriented Programming in Smalltalk and Ada. ACM OOPSLA'86 Proc. ,1988

6 Liang Xianzhong. GRASIS: Graphical Ada-based Specification and Implementation Support for Object-oriented Development. CSRDA (China Ship Research and Development Academy). Thesis of MSc (in Chinese),Mar. 1988


(上接第133页)


14 Ma Q, Wang J T L, Shasha D, Wu C H. New techniques for extracting features form protein sequences. IBM Systems Journal (Special Issue on Deep Computing for the Life Sciences),2001, 40(2):426～441

15 Reese M G,Eeckman F H. Novel Neural Network Algorithms for Improved Eukaryotic Promoter Site Recognition. In: The 7th Int. Genome Sequencing and Analysis Conf. 1995

16 Neural Network Promoter Prediction: Input. http://www.fruitfly.org/seq_tools/promoter.html

17 Knudsen S. Promoter2.0: for the recognition of PolII promoter sequences. Bioinformatics, 1999,15(5): 356～361

18 Ma Q,Wang J T L,Gattiker J R. Mining biomolecular data using background knowledge and artificial neural networks. Handbook of Massive Data Sets, Kluwer Academic Publishers, 2002

19 Ma Q, Wang J T L, Shasha D, Wu C H. DNA sequence classification via an expectation maximization algorithm and neural networks: a case study. IEEE Transactions on Systems, Man, and Cybernetics, Special Issue on Knowledge Management, 2001

20 Noordewier M O,Towell G G,Shavlik J W. Training Knowledge-Based Neural Networks to Recognize Genes in DNA Sequences. Advances in Neural Information Processing Systems, 1993. 530～536

21 Wu C H. Gene classification artificial neural system. Neural and Biological Systems, May 1995. 102

22 黄小兵，陈锋，等. 1%人类基因组数据库系统. 计算机科学，2002，29(8. 增A):273～276

23 Misra M. Prarllel environment for implementing nerual networks. Neural Computing Surveys, 1997,1:48～60

24 Rogers R O,Skillicorn D B. Using the BSP cost model for optimal parallel neural network training. Technical report of department computing and information science, Queens University, Dec. 1997

25 Färber P,Asanovic K. Parallel neural network training of Multi-SPERT. http://cag-www.lcs.mit.edu/～krste/papers/MStrain.ps

26 Färber P. Quicknet on MultiSpert: fast parallel neural network training:[Technical Report TR-97-047]. International Computer Science Institute, Dec. 1997

27 Coetzee L,Botha E C. Parallel neural net traininig on the KSR1. Concurrency - Practice and Experience, 1996,8(8): 617～638

28 DeWitt D J,Gray J. Parallel database systems: the future of high performance database processing. ACM, 1992,36(6)

29 Hagan M T,Demuth H B,Beale M H. 戴葵等 译. 神经网络设计. 北京:机械工业出版社,2002

30 Rogers R O, Skillicorn D B. Batch size and training times in supervised and unsupervised networks, Dec. 1997

31 Färber P. Parallel computing on MultiSpert:[Technical Report TR-97-046]. International Computer Science Institute, Dec. 1997

32 Ahmad A S, Zulianto A,Sanjaya E. Design and implementation of parallel batch-mode neural network on parallel virtual Machine. Proceedings, Industrial Electronic Seminar, 1999

33 Opitz D, Maclin R. Popular Ensemble Mehtods: An Empirical Study. Journal of Artificial Intelligenc Research,1999,11:169～198

34 Riis S K,Krogh A. Improving Prediction of Protein Secondary Structure using Structured Neural Networks and Multiple Sequence Alignments. Journal of Computational Biology,1996,3:163～183