# Fortran新标准的主要特征及面向过程语言的生命力问题

郭浩志 （国防科技大学计算机系，长沙410073）

摘 要

In this paper, main characteristics of the new standard Fortran language are outlined and the life-force problem of procedure-oriented language is further discussed.

从FORTRAN77问世起（1978年），至八十年代末，为产生一个FORTRAN的新标准，曾做了大量工作。1987年秋新标准的初稿被发往世界各地广泛征求意见。在认真研究了500多份答复之后，对初稿进行了大范围的修改。在再次收集各方意见的基础上，1990年1月在美国德克萨斯州召开了FORTRAN工作会议，统一了新标准的技术工作，并非正式地取名为Fortran90。同年3月该标准的最终结构得到国际标准化组织Fortran工作组ISO-IEC JTC1/SC22/WG5和美国标准协会Fortran分委员会ANSI X3J3的双重同意。

评判者是用户。对用户不友善的软件，用户当然不会乐于使用。

3.4.2 **自然语言** 为使软件对用户友善，人与系统的接口界面应尽可能采用自然语言，这一点颇为重要。

3.4.3 **接口界面结构** 采用自然语言只是接口界面的表示问题，更重要的还是接口界面的结构。结构宜简明、力戒烦琐。

## 4. 结束语

软件技术过去困难，现在困难，将来仍然困难。当然，困难的性质与程度会有所不同。只有迎着困难上，战略上藐视，战术上重视，理论联系实际，双管齐下，科学态度，所面临的困难定将逐一克服，软件的本质、软件开发过程的本质定将逐步摸清。届时软件领域将出现前所未有的欣欣向荣的局面，计算机科学技术的发展将别开生面，柳暗花明又一村。

## 参 考 文 献

[1] Freeman, P., Strategic Directions in Software Engineering: Past, Present and Future. Information Processing 89

[2] 徐家福，软件笔谈.计算机科学.1990.3

[3] Liskov Barbara, Structure of Distributed Programs. 12th International Conference on Software Engineering. March 1990

[4] Xu Jiafu et al., Some Research on Inductive Program Synthesis. University Computing (U. K.) 1990 №.3

[5] 徐家福、戴 敏、吕 建，从软件功能规格说明到设计规格说明的转换.计算机学报.第14卷第2期. 1991.2

[6] Xu Jiafu, Dai Min, Lu Jian, From *What to How*. *Advances in Chinese* Computer Science Vol.3.1991

从新标准历时12年的开发，可以看出其指导思想是，继承Fortran的优良传统，扩大在数值计算方面的优势；吸收现代程序设计语言中业已被证实成熟的新概念和新机制，提高软件产品的可靠性、可维性和可移植性；增加民族字符处理能力，在世界范围进一步扩大影响，力求使国际上第一个被广泛使用的FORTRAN的地位与作用，以稳健的姿态延续到21世纪中去。

## 一、Fortran新标准的主要特征

从整体上看，Fortran新标准(以下简称为Fortran NS)的主要特征可概括为：

1. 与FORTRAN77标准的语法和语义完全兼容。因此，所谓对FORTRAN77的变动，实际上指的是语言扩充；

2. 吸收并进一步精化现代程序设计语言中较成熟的概念与机制，如模块机制、自定义类型等等；

3. Fortran成为第一个全面增加民族字符处理能力的程序设计语言。对非英语国家来说，最感兴趣的也许是该语言支持多字符集和多字节字符集；

4. 具有可演变机制；

5. 不给出子集；

6. 为适应当今源程序的直接输入方式和提高可读性，除保留面向穿孔卡片的固定分区格式外，也允许源程序采用自由格式，因而允许一行写多个语句，一语句占多行，每个语句可有拖后的注解。

## 二、Fortran NS与FORTRAN77的主要区别

FORTRAN66与FORTRAN77先是作为美国标准开发的，然后再由国际标准化组织认可。而Fortran NS是美国Fortran工作组在国际Fortran工作组的监督指导下开发的。从技术上看，Fortran NS语言对77版本的主要扩充为：

1. 引进了数据和过程定义的模块结构，从而提供了一个强有力的、安全的数据和过程封装机制；

2. 改善数值计算设施，其中包括引入参数化的内部数值类型；

3. 增设参数化的内部非数值类型，允许Fortran处理器同时支持多个字符集；

4. 提供对整个数组或子数组施行操作的设施；

5. 增加指针数据类型，进而允许建立和管理动态数据结构；

6. 允许用户自定义数据类型；

7. 语言演变机制。

Fortran NS其它重要的变动还有过程递归、动态数组，进一步的控制结构(如CASE…END CASE，DO…END DO，DO WHILE条件，…)，输入输出设施和更多的内部过程。

下面就Fortran NS的主要扩充作一介绍。

## 三、数值计算

虽然非数值计算近十多年来在科学应用中显著增长，但数值计算仍是Fortran的主要应用领域。新标准所有技术工作的重要目标是强化Fortran作为科学软件实现工具的作用。因此，不仅将早期Fortran系统中适于数值计算的设施（例如类型REAL，DOUBLE PRECISION，COMPLEX和INTEGER）全部保留下来，并且还引入了诸如参数化的内部数值类型，具有可移植性的数值精度控制说明以及数值程序性能改进的控制等等。例如：

```
REAL(KIND=4)A
```

指出A是一个含Kind类型参数4的实型变量，而

```
COMPLEX(KIND=8)B
```

指出B是一个含Kind类型参数8的复型变量，即该变量每一成份都是一个含Kind类型参数8的实型变量。

新的内部函数SELECTED-REAL-KIND允许程序员通过自变元指出所需的精度，并返回满足所指精度的Kind类型参数的最小值。例如，下列说明将确保编译程序使用至少含12位(十进制)数字精度和幂范围从$10^{-4}$

到$10^{48}$的数据类型：

```
INTEGER self-real
PARAMETER ( self-real = SELECTED-
    REAL-KIND(12, 48))
REAL(self-real)x, y, z
COMPLEX(self-real) u, v, w
```

**四、参数化的非数值数据类型**

参数化特性也可用于逻辑类型和字符类型。对于逻辑类型，其主要目的是允许处理器为逻辑变量提供一种可任选的存贮结构，即该变量实际占有的空间可不足一个存贮单元，甚至只占有1比特。这对于用于掩码的逻辑数组的操作尤为重要。

字符数据类型的参数化用于另一个十分重要的方面，即允许字符数据也可用于非英语的其它自然语言，尤其是因字符太多而无法用一个字节表示的汉语、日语和其它东方语言。

**五、用户自定义类型**

当今大多数现代程设计语言都允许程序员自定义数据类型。Fortran NS的派生类型提供了一种有效的数据抽象机制。除包含与FORTRAN77一样的6个内部数据类型外，Fortran N还允许程序员将已知类型的任意组合定义为新的数据类型。例如，一个涉及二维几何图形的程序可以包含所谓Point，Line和Circle等新数据类型的定义如下：

```
TYPE Point
  REAL x, y     !x和y都是点的座标
END TYPE Point
TYPE Line
  REAL a, b, c  ! a, b和c是下列直线方程的
                ! 系数：ax+by+c=0
END TYPE Line
TYPE Circle
  TYPE(Point)c  ! c 是圆心
REAL r          ! r 是圆的半径
ENDTYPE Circle
```

Fortran NS允许直接访问派生类型的整个对象，或其单个元素。

```
TYPE (Point)p1; p2
p1=Point (4.5, 5.5) !p1是点 (4.5, 5.5)
p2%x=4.5            !p2的x座标为4.5
p2%y=5.5            !p2的y座标为5.5
        !p1和p2都表示具有同样座标的点
```

实际上，单纯的自定义数据类型是很少使用的，一般还定义这些类型的操作符，并使该类型定义是全程可用的，以使派生类型的对象可用作过程的变元、赋值和输入输出。Fortran NS既允许定义新的操作符，又允许重载内部操作符。过程定义可用来定义（内部的或派生的）类型的操作和非内部赋值。

**六、数组操作**

Fortran NS之所以引入对整个数组和子数组(数组片)的处理操作，主要出于两个原因：

(1) 大数组计算是工程和科学计算问题的一个重要部分，然而FORTRAN77只能逐个元素地处理数组。Fortran NS引入数组操作，不仅使程序语言更为简洁和高级，而且使程序员能更快更可靠地开发和维护科学工程应用；

(2) 大大地简化了施加于很多计算机结构之上的数组操作的优化。

新标准将FORTRAN77的算术、逻辑和字符等操作以及内部函数全都扩充为可对值数组的操作对象施行。新的实施包括对整个数组、子数组和屏蔽数组等的赋值，值数组的常量、表达式和函数(内部的和外部的)。还提供若干新的内部过程，以利用和构造数组，执行聚集/分散操作以及支持可扩充的数组计算能力。

例如，设有数组说明：

```
REAL A(10, 20), B(10, 20), C(10, 20)
```

下列FORTRAN77代码：

```
  DO 10, J=1, 20
    DO 10, I=1, 10
      A(I, J)=B(I, J)+C(I, J)
10 CONTINUE
```

在Fortran NS中可用语句：

A=B+C

替代。为执行基本的向量操作和矩阵操作，新标准提供了不少有关子程序，如：

·标量积DOTPRD(A, B, AB, N)：构造两个长为N的向量A和B的标量积，返回结果AB；

·矩阵乘MATMLT (X, Y, Z, P, N, Q)：构造两个矩阵X和Y的矩阵乘，其中X的体积为P*N，Y的体积为N*Q，返回结果是体积为P*Q的矩阵Z。

在*Fortran* NS中，这些操作都是内部定义的。下列程序Matrix-demo说明了其用法。

```
PROGRAM Matrix-demo
  REAL, DIMENSION(5)::a, b
  REAL, DIMENSION(5, 10)::d, e, f, g
  REAL              ::ab, x(2, 5), y(5, 9),
                      z(2,9)
  INTEGER           ::I
  a=(/1.0, 2.0, 3.0, 4.0, 5.0/)
    !利用数组构子对整个向量a赋值
  b=(/6.0, 7.0, 8.0, 9.0, 10.0/)
  ab=DOTPRODUCT(a, b)
    !利用内部函数DOTPRODUCT 将向量a与b
    !之标量积赋给ab
  DO I=1, 10, 2  !建立数组d, 其奇数行为向
                 !量a, 其偶数行为向量b
    d(:, I)=a
    d(:, I+1)=b
  END DO
  e=SPREAD(a, 1, 10)  !根据a的10 行建立矩
                      !阵a
  f=d+e               !矩阵加
  g=d-e               !矩阵减
  x(1, :)=a           !将a赋给x的第一列
  x(2, :)=b           !将b赋给x的第二列
  y=f(:, 1:9)         !将f的适当大小的数组
                      !片赋给y
  z=MATMUL(x, y)
    !利用内部函数MATMUL 将x与y 的矩阵乘
    !赋给z
  STOP
END PROGRAM Matrix-demo
```

在FORTRAN77中，对于大小需求不明的数组，常常先说明一个比通常需要大很多的数组，然后再输入，或者先确定该程序此次运行的数组最大可能长度，然后往往只使用数组的一部分。这两种方法浪费内存，甚至仍然解决不了问题。为此，Fortran NS提供了动态分配数组能力。例如：

```
SUBROUTINE Work(a, b, c)
  ! a, b, c都是大小仅在运行时刻才确定的数
  !组
  REAL, ALLOCATABLE, DIMENSION(:,:)
    ::a, b, c
  INTEGER  ::n, m
        ⋮
  !子程序运行到此处，必须建立所要求大小的数
  !组a, b, c
  READ *, n, m
  ALLOCATE(a(n, m), b(n, m), c(n, 2*m))
  ! a, b, c现在可以随意使用
        ⋮
  DEALLOCATE(a, b, c)  ! a, b, c现在不再
                       !存在
        ⋮
END SUBROUTINE Work
```

## 七、指针

指针允许动态地测定数组大小，排列数组，链接诸结构，并建立链表、树和图。任一内部类型或派生类型的对象均可被说明具有指针属性。

指针不是数据实体，而是某种数据实体的属性，它必须借助一种形式新颖的说明来定义。例如，下列语句

```
TYPE(Node), POINTER::head, cur-
    rent, tail
```

说明了三个指针对象：head，current和tail，它们均指向派生类型Node的对象。又如：

```
REAL, DIMENSION(:, :), POINT-
    ER::in, out
```

说明了两个指针对象in和out，它们都指向两维实型数组。

## 八、模块机制

FORTRAN77为全程量的访问只提供了公用区一种方式。在程序设计语言和软件思想日趋新颖的今天，这种方式就显得很落后了。首先，公用区的用法很麻烦，它要求在所涉及的所有程序块逐一给出同一公用区的说明；另外，公用区的使用很不安全，它要求程序员谨慎考虑其中元素的类型匹配和名字冲突问题。这都要求程序员在安排内存数据方面有较强的能力和十分严谨的作风；再者，倘若在一些新型计算机体系中实现公用区，那是要大大降低效率的。

从程序块的多入口功能来看，ENTRY语句对于实现一组相关的过程（也许还涉及公共数据对象）是很别扭的和受限制的。此外，在FORTRAN77中无法表示局部于程序单元的过程定义，尤其是接口信息。

为此，Fortran NS总结了自Simula的Class到ADA的Package的各种典型数据抽象的利弊，提出了新的模块机制。

模块是一种新颖的程序单元。一般情况下，它能提供完善的封装结构和信息隐蔽机制，以利将很多新的方法引入Fortran程序设计。而FORTRAN77中的说明仅与数据对象有关。除了必须具有一全程名外，Fortran NS的模块还可包括数据对象、派生类型、过程定义和过程接口信息等各种说明的任意组合。

在任一模块中，允许将若干项指定为private，以导致任何使用该模块的过程均无法访问到这些项。这种所谓信息隐蔽思想在开发可靠的、可移植的软件库时十分重要。

下面给出的几何图形模块，定义了若干新的数据类型和操作。

```
MODULE Geometry
  TYPE Point
    PEAL x, y      !x和y均为点的座标
  END TYPE Point
  TYPE Line
    REAL a, b, c
    !a, b和c均为下列直线方程的系数：
    !ax+by+c=0
  END TYPE Line
  TYPE Circle
    TYPE(Point)c   !c是圆心
    REAL r         !r是圆的半径
  END TYPE Circle
  INTERFACE OPERATOR (.TO.)
    TYPE (Line) FUNCTION Join (point1
          point2)
      TYPE (Point), INTENT (IN)::point1,
            point2
    END FUNCTION Join
  END INTERFACE
  INTERFACE OPERATOR(*)
    TYPE(Point)FUNCTION Intersect(line1,
          line2)
      TYPE (Line) , INTENT(IN) ::line1,
            line2
    END FUNCTION Intersect
  END INTERFACE
  PRIVATE Join, Intersect
  !禁止从模块外访问Join和Intersect两函数名
    ⋮
CONTAINS
  TYPE (Line) FUNCTION Join (point1,
        point2)
  !此函数求解两点之间的直线方程
    TYPE(Point), INTENT(IN)::point1, po-
          int2 !哑元
  !连接点(x1, y1)和(x2, y2)的直线之方程为：
  ! (y1-y2)x-(x1-x2)y+x1*y2-x2*y1=0
    Join%a=point1%y-point2%y
    Join%b=point2%x-point1%x
    Join%c=point1%x*point2%y-point2%x*
      point1%y
    RETURN
  END FUNCTION Join
  TYPE(Point)FUNCTION Intersect (line1,
        line2)
  !此函数计算两线交点之座标
  !若两线不相交，则返回在无穷远处之点
```

```
    TYPE(Line), INTENT(IN)::line1, line2
      !哑元
    REAL small, denom              !局部变量
    PARAMETER (small=1E-10)
    denom=line1%a * line2 * b-line2%a*
          line1%b
    IF (denom<small) THEN
        !测试平行线
      Intersect%x=HUGE(denom)
        !置x和y座标为最大可能的值
      Intersect%y=HUGE (denom)
    ELSE
      Intersect%x= (line1%b * line2%c-li-
        ne2%b*line1%c) /denom
      Intersect%y= (line1%c * line2%a -li-
        ne2%c*line1%a) /denom
    END IF
    RETURN
  END FUNCTION Intersect
        :
END MODULE Geometry
```

设过程Geometry-demo需要使用上述模块的几何图形类型和操作。下面是它的框架：

```
SUBROUTINE Geometry-demo
  USE Geometry
  TYPE(Point)pt1, pt2, pt3
  TYPE(Line)ln1, ln2, ln3
          :
  ln3=pt1.TO.pt2    !利用用户定义操作符.TO.
                    !计算两点连成的直线
  pt3=ln1*ln2       !利用重载运算符*
                    !计算两线之交点
        :
END SUBROUTINE Geometry-demo
```

在上述模块中，若略去PRIVATE语句，则两个函数名Join和Intersect 也是公用的，我们可以写出如下语句：

```
  ln3=Join(pt1, pt2)
  pt3=Intersect(ln1, ln2)
```

在特殊情况下，模块可用作一种只提供全程变量的更可靠、更有效的方法，例如：

```
MODULE Common-Block
  REAL x, y(10), table(5, 20)
  INTEGER i, j
  CHARACTER string*30
END MODULE Common-Block
```

为达到与上列Fortran NS模块同样的效果，FORTRAN77就不得不在每一个使用有关公用区的程序块中写出下列5个语句：

```
  REAL X, Y(10), TABLE (5, 20)
  INTEGER I, J
  COMMON/BLK1/X,Y(10),TABLE (5,10),
    I, J
  CHARACTER STRING*30
  COMMON/BLK2/STRING
```

## 九、面向过程语言为何经久不衰？

近十多年来，第四代语言因软硬件的迅猛发展而越来越受到人们的青睐。函数型、逻辑型和面向对象型语言的相继问世，在计算机科学研究领域内掀起了向强制式语言猛烈冲击的热浪。专家们纷纷批评面向过程语言有这样或那样的缺点，但是从当前计算机实际使用情况与发展特征来看，没有足够的证据可以表明一些最流行的面向过程语言（例如，FORTRAN, COBOL, PASCAL, C, BASIC）会在近期内退出计算机的历史舞台。虽然它们多数诞生在六十年代，有的更早，但其生命力却特别旺盛。几次国际范围的统计结果都说明，在数以千计的程序设计语言中，它们始终保持为使用最多的几个（当然它们之间的排列次序因调查范围不同而有所不同）。面向过程语言之所以经久不衰，我们认为主要有以下几条原因：

1. 已有一大批长期使用面向过程语言的用户，他们使用得心应手，并积累了十分丰富的实践经验。他们不会轻易换用风格迥然不同的新语言；

2. 流行的面向过程语言大多编程简易，使用方便，而且世上已出版了大量有关教材和参考书籍，各类学校和培训班大都乐意以它们为工具，讲授程序设计，推广语言；

3. 围绕面向过程语言已逐渐产生一套

内容丰富、功能健全、形式多样、互相配套的成熟工具（如各种标准程序、标准子程序、接口程序与其它实用程序），有的甚至已形成有机的环境。如ADA语言的APSE环境，C语言的UNIX环境；

4. 主要流行语言出于以下两个目的，都及时、多次推出新的标准版本：

（1）利于编译程序研制人员统一遵循一种独立于具体计算机的语言规定，以提高源程序的通用性和可移植性；

（2）建立一种便于人与人之间交流和表示算法的工具。

国际上很多组织，诸如ISO（国际标准化组织）、IEC（国际电工委员会）、ANSI（美国国家标准化协会）、IEEE(电气与电子工程师学会)、ACM SIGPLAM（计算机协会的程序设计语言专门组织）、JIS（日本标准协会）和我国标准化组织等等都经常从事程序设计语言的标准化工作。当前已建立标准的几乎都是面向过程语言。例如，FORTRAN，ALGOL60，COBOL，PASCAL，PL/1，Minimal BASIC和ADA，而且这些语言大多在不同发展阶段都还通过吸收新概念和新机制，及时推出新的标准。如FORTRAN先后推出的主要标准有Ⅱ，Ⅳ，77和90；

5. 为降低软件成本，提高它的性能，并增强其商品的竞争能力，面向过程语言不时被渗入软硬件的新技术和新方法，以崭新面目推出市场。例如，目前PC机的三种典型BASIC产品(Borland公司的Turbo BASIC，Microsoft公司的Quick BASIC和BASIC原创始人推出的True BASIC)先后都采用了高速处理器，高分辨率的彩色显示设备，鼠标器等硬件以及窗口、菜单技术等软件；

6. 流行的面向过程语言无例外地都得到了政府或权威机构的强有力的支持和赞助。例如，FORTRAN，PL/1和APL得到了国际上最大的计算机公司IBM的支持，COBOL、ADA是在世界上最大的计算机用户——美国国防部直接领导下完成的，PASCAL为欧美许多著名大学和研究机构所乐意采用，而C、SNOBOL是由国际著名的贝尔实验室研制成功的；

7. 其它非过程语言，尽管具有很多吸引人的优点，但毕竟不成熟，未商品化，在市场上尚未对流行的过程语言构成具有威胁的竞争力。况且，出于效率等原因，很多第四代语言实际上也包含过程性功能。

## 十、展望

Fortran新标准是面向过程语言的一个典型代表。它的出现反映了面向过程语言的发展特征。我们估计，进入九十年代后，面向过程语言总的发展将趋于成熟和平稳；使用人数在比例上因第四代语言的崛起而逐步下降；但在个别概念、思想、技术与机制上仍会取得若干成果；在数值计算和实时控制等应用中仍占优势；ADA在军用软件中将占主导地位。进入二十一世纪后，面向过程语言将与第四代语言有机地结合到软件生产自动化过程之中。

### 主要参考文献


[1] ISO, IEC, ANSI, Information technology——Programming languages——FORTRAN, 1990 June.

[2] T. M. R. ELLIS, FORTRAN77 Programming with an introduction to the Fortran 90 standard, Addison-Wesley, 1990.

[3] 郭浩志主编，程序设计语言概论，国防科技大学出版社，1989.7.

[4] 郭浩志，计算机语言的特征、分类、应用与发展，国防科技大学，1990.10.