郑 州 工 学 院 学 报

第9卷 第4期 JOURNAL OF ZHENGZHOU INSTITUTE Vol.9, No.4

1988年12月 OF TECHNOLOGY December 1988

共享程序包在UNIX系统中的实现方法

王文义

（计算机及自动化系）

提 要：本文通过对unlx系统支持的各种语言特点的研究、开发和总结了一组不同语言间接口的方法及注意事项，从而为实现扩充系统软件资源，建立一个为用户共享的程序包提供了条件和可行性。

关键词：UNIX操作系统，接口。

1 前言

每一种计算机系统，通常可以支持多种语言工具，从机器语言，汇编语言到各种高级语言等等。对于MC68000微机系统，一般都配备有硅谷软件（简称SVS）BASIC—PLUS、PASCAL／300、FORTRAN77（后简称为PASCAL、FORTRAN）、C以及ASM68K等语言。这些语言，从设计的角度来看，可以说是风格各异，当然用途也不尽相同。我们知道，并不是每一个计算机用户都精通所有上述语言的，这个不足或多或少地限制了他们编程的技巧与质量。因为在研制大型的工程应用软件时就只能用某种单一的语言逐条进行编制与调试，而不能充分利用其它语言在某一方面的突出优点。所以说，拿程序研制周期、程序员调度、内存占有量、机时占有量、程序风度及可读性等诸因素来衡量，该程序就未必是最好的。

针对上述问题，本文开发与总结了在UNIX系统支持下各种不同语言间的级别顺序和调用关系、注意事项及处理方法，从而为在系统中建立一个用各种语言书写（由精通该语言的高级程序员或应用人员编制）的程序包（目标为中间代码形式）——扩充系统资源提供了途径。程序包可因不同的专业而异，也可以是通用的。它的建立，使得用户只要熟悉一种高级语言，就可以得心应手的去利用相应的程序包高效的实现自己的目的，根本勿需再临时去学习其它语言，也不需要去了解自己调用（或链接与装配）的程序包子结构是用何种程序编写的。程序包可视为系统资源，它为所有用户所共享。由于用户在自己的程序中使用了程序包，因此用户的源程序一定显得短小精干，可读性强且结构性好，这对于推广微机应用与提高微机应用水平是具有一定实际意义的。

2 数据存贮与参数传递

计算机支持的不同语言对于数据存贮方式与参数传递方式的规定一般也是不一样的，这就给我们使用各种语言程序的调用带来了不便，它们必需通过某种改变以后才能建立调用的关系。应说明的是，MC68000上的BASIC—PLUS由于采用了解释方式而不是编译方式，因此用它编写的BASIC程序的执行方式与其它语言如PASCAL、FORTRAN、C等有着根

本的不同，因此本文中所说的调用，是对于不涉及BASIC—PLUS而言的。

FORTRAN对于所有的数据类型如整型、实型和逻结型数据都开辟有相同的存贮空间，即32位（4个字节或2个字长），其参数传递方式为传地址方式，传递过程为

• 对实参为简变时，变量地址⇒形式单元；

• 对实参为下标变量时，数组元素地址⇒形式单元；

• 对实参为常数、表达式时，先计算出实参的值⇒临时存贮单元T，再把T的地址⇒形式单元。（符号“⇒”意即送入）

• 当被调用段执行时，对形式参数的访问，执行为对形式单元的间接访问（相当于指针）。

PASCAL的数据存贮方式不同于FORTRAN语言。以整型数据为例，当发生调用时，若FORTRAN中数据为隐含整型，那么PASCAL中的整型变量只能说明为长整型，即Longint形式，否则将出现数据类型不一致的错误。PASCAL中参数传递方式有两种：变量参数和数值参数，其中前者是传地址方式（含义与上面对FORTAAN的解释相同），后者是传值方式。传值方式的含义定义如下：

• 先计算实参的值，并⇒形式单元；

• 对形式参数的访问就是对上述形参的值的访问。

不同的传递方式，有着不同的结果。如有一类PASCAL程序如

```
begin     I: =1;
          A[1]: =1; A[2]: =2; A[3]: =3; A[4]: =4;
          ABC(I, I, A[I]);
          write(A[I], I)
end;
Procedure    ABC(X, y, Z);
  begin      X: =X+1;
             y: =y+1;
             Z: =Z+1
       end.
```

若采用传地址方式，程序运行结果为：

```
3        3
```

若采用传值方式，则结果为

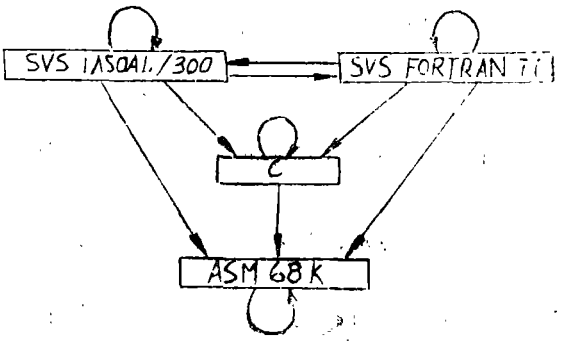
```
1        1
```

可见，不同程序的调用，为了兼容，必需搞清其数据存贮方式与参数传递方式是否一致。

3　各语言程序间的调用模式

UNIX系统中各语言间存在着不同的调用级别，即存在着同级可调（不一定是递归调用），级别高者可调用级别低者规则。其中PASCAL、FORTRAN属高级语言，级别最高；ASM68K是汇编语言，级别最低；C语言是介于高级语言与低级语言之间的语言，故

级别居中。各语言的调用关系如下图所示（调用关系以箭头示之）：



下面分别说明各语言调用的接口规则及注意事项：

3.1 PASCAL和FORTRAN程序的互相调用。

PASCAL和FORTRAN的参数传递是通过堆栈实现的，参数依它们被说明的先后顺序压入栈中。若被调对象为函数，则在任何参数压栈之前，予先在栈中为函数的结果分配空间，当控制从函数（或过程）返回时，应弹出栈中的所有参数，仅保留函数结果于栈中（过程则不存在该项）。

由于FORTRAN是模块化结构，其函数子程序或子例行子程序本身就是可以进行单独编译的程序单位，于是当PASCAL作为主程序对其进行调用时，只需注明该子程序属于外部过程，加上关键字EXTERNAL即可，其参数的处理方法类似于PASCAL中向前调用FORWARD。反之，对于主程序是FORTRAN语言，子程序是PASCAL语言，那么对该子程序的处理就稍为复杂些，因为PASCAL的子程序不构成完整的编译单位，必需通过一定的方法把它变成一个等价的可编译单位，然后才能进行链接与装配，处理方法如下：

```
UNIT<文件名>;
    INTERFACE
    全 ┌ 常量（定义）
    局 │ 类型（定义）
    有 │ 变量（定义）
    效 └ 过程或函数（定义）/* 有形参，但无执行体 */
    IMPLEMENTATION
    局 ┌ 标号（定义）
    部 │ 常量（定义）
    有 │ 类型（定义）
    效 │ 变量（定义）
       └ 过程或函数 /* 无参数，只有执行体 */
END
```

简例：

① FORTRAN 主程序

```
      DO  100  i=1, 10
         CALL ABC(i, j)
  100  WRITE(*, *) i, j
      STOP
      END
```

PASCAL 子程序

```
UNIT GG;
 INTERFACE
PROCEDURE ABC(VAR i, j: LONGINT);
    IMPLEMENTATION
       PROCEDUER  ABC;
          BEGIN
            j: =i*i
          END;
END
```

② PASCAL 主程序

```
PROGRAM  B52;
  VAR
    a, b: LONGINT;
PROCEDURE ABC(VAR i, j: LONGINT);
    EXTERNAL;
  BEGIN
    FOR a: =1 TO 10 DO
      BEGIN
        ABC(a, b);
        WRITELN(a, b)
      END
END.
```

FORTRAN 子程序

```
SUBROUTINE  ABC(i, )
   j=i*i
   RETURN
   END
```

当被调用段予先已被变换成某种形式的中间代码时，那么用主程序源文件和该中间代码文件一起完成编译、链接和装配工作，最终的可执行码或放于a.out中，或放于用户指定的文件中，然后就可运行执行码。

当参数传递不兼容，如上例PASCAL程序的参数没被说明为变量参数，将指示出如下信息：

```
Ld: undefined-
    %-TFWR
    %-IXFWR
```

从而导致系统无法实现存贮管理而跳出。

3.2 FORTRAN、PASCAL与C程序的调用

C语言函数可以独立进行编译，勿需特殊处理。但由于C语言的调用级别低于FORTRAN和PASCAL语言，因此在接口与调用惯例上就有所差别，FORTRAN语言与PASCAL语言不能直接调用C语言子程序，必需通过一个用ASM68K汇编语言书写的缓冲接口程序“wrapper”搭桥，才能使之兼容。该接口程序很容易编写。FORTRAN调用

C与PASCAL调用C的情况基本相同，仅举后者为例。

PASCAL程序调用C语言函数是通过把该函数作为一个外部函数来实现的。PASCAL与C语言之间应考虑下述差别：

①C语言的所有参数，不论是什么变元，都在栈内为每个变元分配4个字节的存贮空间，故对于整型参数，PASCAL中应定义为“长整型”（LONGINT）、且应是变量参数。

②C语言参数压栈的顺序与调用它的PASCAL程序中参数的排列顺序恰恰相反。如：当C函数参数是ABC（a，p，m）时，则PASCAL中的调用语句应是ABC（m，p，a）。

③C语言函数值放在寄存器D_0中，而PASCAL程序的函数值却是放在栈内。

④C语言与PASCAL语言在对浮点数的存贮处理上根本不同，故应慎用。

掌握了上述区别，即可正确的写出PASCAL与C间的“wrapper”，具体实现过程为：

①wrapper需定义为全局量（·globl说明）。

②传送给系统链接程序 ulinker 的外部访问名应用大写字母（产生PASCAL），而wrapper中调用的则应是对应的小写名（产生C），注意在C例行程序名（小写）之前有一横线符，该符号在PASCAL语言中是非法的。

③在所有情况下，PASCAL把它的返回地址放入栈内。wrapper应实现把堆栈的入口内容托进地址寄存器A_3，A_3保证不受C程序的干扰。

④控制从C函数程序返回时，wrapper对于从PASCAL传递的每一个参数保证清除4个字节的位置。

⑤若PASCAL调用的是一个C函数过程，则wrapper就必需把返回值从寄存器D_0推入栈内。

简例：

PASCAL程序

```
PROGRAM B52;
  VAR i, j, k: INTEGER;
    FUNCTION abc(VAR a, b, c: LONGINT): LONGINT;
      EXTERNAL;
  BEGIN
    j:=1; k:=1;
    FOR i:=1 TO 5 DO
      BEGIN
        j:=abc(i, j, k);
        WRITELN(j)
      END
  END
```

C程序

```
abc(c, b, a)
  INT*a, *b, *c;
```

```
{
  RETURN ( *a + *b + *c )
}
```

接口程序wrapper

```
 .globl    ABC, -abc
ABC:    movl    sp@+, a3
        jsr     -abc
        addl    #12, SP
        movl    do, SP@
        jmp     a3@
```

PASCAL调用C程序中对PASCAL、C与wrapper的具体处理步骤为：首先把C 程 序和wrapper各自变换成UNIX系统的中间代码文件，然后用PC对PASCAL主程序 及 上述两个代码文件进行编译、链接和装配。

3.3 FORTRAN、PASCAL、C对ASM68K（汇编）程序的调用。

该三种调用，原理上是相同的，仅举FORTRAN调用ASM68K为例说明之。

FORTRAN主程序是通过堆栈向汇编子程序传递参数并实现调用的， 其 步 骤可描 述为：

对字符参数，将有一对应指向字符目标的32位指示字推入栈内并后随16位长度的字符。除此之外，其它任何类型的参数都是依其说明的位置上以一个指向目标的32位指示字压入栈中。若调用的是函数，则在任何参数压栈之前予先为函数结果在栈中分配空间。当控制从汇编子程序返回时，清除栈内的所有参数，只把函数结果留在栈中。汇编子程序由三部 分 组成：入口代码、汇编程序体和结束返回代码。如果仅有两个传递参数时，该汇编子程序具有如下形式：

```
          .globl    ROUTINE          } 入口代码
ROUTINE:  movl      sp@+, a3         }
              ⋮                      } 汇编子程序体
          addl      #8,   SP         } 结束与返回代码
          jmp       a3@              }
```

PASCAL、C调用汇编子程序的结构与上同，但在PASCAL中必需对所调过程（或函数）加上外部标识，而在C调用汇编子程序时，应把入口代码中的标识改为小写。

4 建立共享程序包的方法

在认清了各种语言间的调用关系及调用规则之后，我们会很自然的想到如何利用这些关系去更有效的为用户服务。

可以设想在UNIX系统中建立一个程序包，它的内容包括通常的函数、文件管理、非线

性规划、仿真、优化及一些高度专业化的应用程序，如建筑类的有限元，电力类的系统规划等等。它们均以子程序的形式出现，并且请专家们选用能体现高效率的语言编制，然后把它们编译成彼此独立的中间代码文件或系统目标代码文件存于介质之上。为了便于管理与查询，还需在系统中辟一目录，把上述文件置于该目录之下，这样做还有一个好处，就是可以随时扩充该程序包的容量。

程序包初举规模之后，当用户需要使用程序包时，他只需用自己熟悉的语言编一主程序，通过使用PC，F77，CC等编译命令把主程序与所需程序包中的子程序链接装配起来，成为一个可执行文件。

程序包的建立，对扩充UNIX系统的软件资源，避免程序的重复编制、减低程序冗余度、缩短大型应用程序的研制周期，提高程序的质量及计算机的应用水平，将会起到一个非常积极的推动作用。

参考文献

〔1〕 《计算机研究与发展》编辑部 UNIX分时系统程序员手册 一、二卷

〔2〕 美国硅谷软件公司 SVS FORTRAN77、SVS PASCAL/300、C、ASM68K参考手册（1981）

〔3〕 北京工业大学微型计算机研究开发应用中心 MC68000十六位微处理器（1983.8）

The Realization Method of the Shared Package in Unix System

Wang Wenyi

(Computer and automation engineering department)

Abstract: By studying the characteristics of all kinds of languages supported by Unix System, the author of this article has developed and summarized a set of interface methods used between different Languages and the points which attention should be paid to. Therefore the condition and the feasibility have been created to realize the expansion of the system's software resources and build up a program package shared with users.

Keywords: Unix operating system, Interface.