motif 图形用户界面编程接口的可移植性封装

刘　曙，　汤伟华
（空军工程大学 导弹学院，陕西 三原　713800）

摘　要：跨平台软件开发是工程实践中经常遇到的难题。采取了封装图形界面接口的方法，建立一层接口，称之为界面抽象层，当移植整个程序时，只需修改此层接口，使得工作量最少，从而较好的解决了程序移植性问题。以 Alpha 工作站上利用 Motif 开发可移植性的图形界面封装接口为例，从消息循环、界面基本组件以及事件回调等方面实现了 Alpha－Unix 平台到 PC－Window 平台跨平台软件开发，实际效果良好。

关键词：软件开发；跨平台；Alpha 工作站；Motif；图形界面接口
中图分类号：TP393.08　　**文献标识码**：A　　**文章编号**：1009－3516(2003)02－0078－03

程序的可移植性一直是一个困扰大家的问题，得不到完全的解决。跨平台开发出来的程序具有平台无关性，目前平台无关性主要有两种：目标代码级和源代码级[1]。以 Java 为代表的虚拟机跨平台方法，其特点是可以做到“一次开发，到处运行”，在目标代码级上实现了平台无关性。虽然 Java 具有良好平台无关性，但在某些场合下要求具有很强的实时性，这时采用 Java 就不太合适了。

在源代码级实现了平台无关性，就是利用大多数平台所共同支持的同一种计算机语言，用同一个源程序在不同平台上编译链接，生成目标代码取得相同的运行效果，从而实现跨平台的目的。C 语言是一种几乎所有操作系统支持的，能运用于广阔应用领域的编程语言[2]，适合开发系统一级的大规模软件。

但是，图形用户界面(GUI)不能直接跨平台，因为各种操作系统有不同的 API 调用函数生成各自的图形界面，而像 UNIX 这样的操作系统甚至不止一种界面标准，如：Motif、OpenLook 等[1]。界面程序在整个程序总量中占有很大的比例，而且移植起来较为困难，这是采用 C 语言进行跨平台开发面临的一个难题，我们通过研究，采取了封装图形界面接口的方法，只需修改此层接口，就能较好的解决程序移植性问题。

1　设计思想与实现

欲使一套接口适用于所有标准的图形用户界面，必须分析它们的共同特征，除了界面外观及函数原型不同外，它们的设计思想是相同的，从而主体结构也是相似的[3]。

2.1　作为程序核心的消息循环

所有的图形用户界面都是以事件驱动作为核心的运行模式，这就决定了各种图形用户界面编程时都需要一个消息循环作为程序的驱动中心，为此我们设计了如下的接口函数：

intLgGuiInit(structLgGuiInfo ＊, …); (1)

char LgEventDeal(charbEventFirst, …); (2)

函数(1)是图形界面初始化程序，它调用有关 Motif 函数，初始化编程接口，获得当前机器的各图形特征，如屏幕数、各屏幕颜色深度、各屏幕尺寸等。函数(2)为主循环应当调用的事件处理函数，举例如下：

structLgGuiInfolgGuiInfo;

charbQuit;

收稿日期：2002－12－02
基金项目：军队科研基金资助项目(W9960)
作者简介：刘　曙(1970－)，男，湖南益阳人，讲师，硕士，主要从事指挥自动化系统开发研究.

```
LgGuiInit( &lgGuiInfo );
do
{
bQuit = LgEventDeal( True );
……./*其它处理*/
}while( bQuit );
…
```

上面这段程序是一种最典型的调用结构,在"其它处理"部分我们可以执行网络处理,或者诸如时钟处理、空闲处理之类的其它任务,函数(2)内参数的含义是控制事件是否优先,如果是事件优先,将优先处理完所有累计事件后再退出,否则处理完一个事件即退出。

2.2 界面基本组件

任何图形用户界面都是由一系列的窗口、对话框、按钮…等基本组件构成,不同的GUI,它们以不同的数据类型如整数、指针等来标识这些组件,用一个结构来唯一标识一个组件如下:

```
struct LgBox
{
struct LgBox *pNext;
Widget w;
Struct LgCbSeal *pHeadCbSeal;
}
```

定义一系列的函数接口以封装那些创建、摧毁和处理组件的不同种类函数,

2.3 事件回调

事件驱动的核心机构是消息循环[4-5],这里有一个关键性的机制——事件回调,不同的GUI有不同的定义回调函数及加载回调函数的方式,这使得设计回调函数接口成为一个棘手的问题。

首先使得回调函数的定义有统一的一个标准如下:

```
char CallbackFun(struct LgBox *pBox,char *pAny);(3)
```

用户定义的任意回调函数必须符合函数(3)的形式,通过函数(3)的两个参数,回调函数将得到本组件的标示指针以及传给它的一个char指针,可以将它转成任意需要的数据类型,这样就传递了用户的数据。

然后必须定义一套自己的标准回调事件标示,例如:

```
#define LgEVT_WIN_POP     0
#define LgEVT_DLG_OK      1
……
```

给组件加载回调函数则通过如下函数·

```
struct LgCbSeal *LgAddCbFun(struct LgBox *pBox,int iEvent,void (*CbFun)(),char *pAny);(4)
```

函数(4)给pBox所标示的组件加上回调函数CbFun,由于不同的GUI,回调函数形式不同,我们定义的函数(3)的形式不能直接加载到组件上,而是要先将一个Motif要求的回调函数,通过参数传给其真正所需的用户回调函数及需要传递的参数,在这个Motif的标准回调函数内再调用真正的用户回调函数,如下所示。

```
struct LgCbSeal
{
struct LgCbSeal *pNext;
LgBox *pBox;
Void(*CbFun)(struct LgBox *pBox,char *pAny);
Char *pAny;
}
void MotifCbFun(Widget w,struct LgCbSeal *pCbSeal)(9)
{pCbSeal->CbFun(pCbSeal->pBox,pCbSeal->pAny);(10)
}struct LgCbSeal *LgAddCbFun(struct LgBox *pBox,
```

```
int iEvent,void ( * CbFun)( ),char * pAny)
{
struct LgCbSeal * pCbSeal;
pCbSeal = ( struct LgCbSeal * ) malloc( sizeof( struct LgCbSeal) ) ;
if( pCbSeal = = NULL )
return( NULL) ;
pCbSeal - > pBox = pBox;
pCbSeal - > CbFun = CbFun;
pCbSeal - > pAny = pAny;
switch( iEvent)
{
case LgEVT_WIN_POP:
XtAddCallback( pBox - > w,popupCallback,MotifCbFun,pCbSeal) ;( 11)
Break;
case LgEVT_DLG_OK:
.....
}
.....
}
```

实际效果证明,建立界面抽象层,对 Motif 图形用户接口进行封装,再移植到其它平台上的方法是切实可行的。

参考文献:

[1] 张 倪,莫 斌. Motif 与图形用户界面开发工具[M]. 北京:清华大学出版社,1995.

[2] 徐士良. C 常用算法程序集[M]. 北京:清华大学出版社,1999.

[3] John Shapley Gray. UNIX 进程间通信[M]. 北京:电子工业出版社,2001.

[4] Dijstra E W. A Discipline of Programming[M]. Upper Saddle River,NJ:Prentice - Hall,1976.

[5] Martin J. System Design from Provably Correct Constructs[M]. Upper Saddle River,NJ:Prentice - Hall,1985.

(编辑:田新华)

The Transplantable Encapsulation of OSF/Motif GUI

LIU Shu, TANG Wei - hua

(The Missile Institute, Air Force Engineering University, Sanyuan 713800, Shaanxi, China)

Abstract: Multi - platform software development is a difficult problem frequently met in engineering practice. In practice, we adopt the method of GUI encapsulation in establishing one layer of interface called the interface abstracts layer. In transporting the whole program, the only thing to do is to revise this layer of interface, which only needs the least work load, thus better solving the problem of program transportability. On the Alpha work station, by taking the exploitation of GUI interface with Motif for example, the development in message loop, base widget of GUI interface and Event Callback is realized from Alpha - Unix's platform to PC - window's platform, and a good practical effect is obtained.

Key words: software development; multi - platform; Alpha work station; Motif; GUI