图形用户接口

郭　江　(北京航空航天大学)　廖越虹　(中国科学院计算中心)

摘要:本文主要介绍了图形用户接口(*GUI*)的功能、标准和输出特征、并简要讨论了目前流行的*GUI*产品及兼容工具的概念。

由于图形用户接口主要是用于和用户进行交互,因而其功能和实现就要针对这些交互来完成。图形用户接口必须要具有交互功能,如高分辨率的图形和声音,而且在性能水平下降时也不会降低。但是,完整的应用系统需求通常超出了图形用户接口的交互功能,而只能由各个系统部分通过合作来满足。

一、图形用户接口的角色和功能

图形用户接口系统在客户、服务器环境中的主要功能是表示和某些事务逻辑。终端用户和应用系统的交互是通过表示逻辑完成的。表示逻辑是一个应用层,一方面和应用系统的事务逻辑进行交互,另一方面和终端用户进行交互。和用户的交互包括所有和物理设计(终端)的交互以及实际终端用户进行输入、输出处理(屏幕I/O、键盘I/O、鼠标等)。

传统的表示功能处理是基于字符显示的,也就是处理器顺序显示从应用系统接收到的字符。由于显示功能的不断发展,高性能工作站提供了图形显示能力。这些显示器通过建立特殊的"位图"允许处理器控制屏幕上的单个象素。图上的每一位相应于屏幕上的一个象素(象形元素),这样处理器就可以画出任意图形,如字符、符号以及图画。屏幕的分辨率是由有效的象素数目来确定的。因此,颜色、大小以及屏幕上对象的位置并不受特定字符形式以及行和列的限制。这种显示能力使得软件开发人员可以利用丰富的图形、图象、以及视听输入、输出设备创建有效、直观的应用系统接口。很明显,有这种显示能力的图形用户接口系统必须以图形表示的方式来显示。这样应用系统的应用逻辑就可以使用电子邮件、服务器资源访问控制软件等等,从而可以将应用逻辑的开发和表示逻辑分开来。

二、表示管理

图形用户接口系统的表示管理说明必须满足开放分布式环境中的客户/服务器体系结构的目标。表示管理要满足一个标准,这样系统就能和用户以一致的风格进行交互,而且不仅能在各种硬件平台上移植,还能和其他开放式的系统应用进行互操作。

1.标准的GUI

用户和应用系统之间的一致性接口是开放系统的一个关键。这个需求非常重要,因为用户接口既影响到了开发人员,也影响到了终端用户。实际上,某些程序和数据库的接口对用户是隐含的,而应用表示则对开发人员和系统用户都是可见的。用户和应用系统间的表示接口称为图形用户接口(GUI),主要用于以图形方式向用户表示信息,这里的图形并不一定是图画。图形表示还用于常见多文本、多风格的字处理和桌面排版系统。

目前,GUI接口有许多种类(包括基于字符方式的)。每次出现新的接口都需要用户和开发人员去学习和修改应用系统。一个新的图形用户接口可能引起整个应用系统的重写以支持所需的GUI。通常是为一个GUI写的应用系统不能移植到其它GUI环境之中。这样,开发人员的产品开发就限制到了特定的接口上,使他们很难进行选择。

这样的例子很多,不兼容的GUI接口包括:Macintosh,Microsoft Windows以及OS/2的Presentation Manager。因此,具有共同应用程序设计接口(API)的标准GUI对开发人员和用户的生产率有重大影响。

2.标准GUI的总需求

从工业角度出发,由硬件和软件厂家所支持的、用户组织所选择的特定 GUI 就可能成为 GUI 的标准。但是,开放系统的标准图形接口应用满足下列需求:

(1) 可移植性: 应用系统和用户技能应可在各种开发系统平台上移植。一个标准的 GUI 应为各种平台支持特定的 API,这样就能迅速从一个平台移植到另一个平台。一个标准 GUI 应在所有平台上保持"观感"的一致性,因而减少了学习的需求。

(2)标准协调:这个需求是开放系统的关键。也是 GUI 吸引大量公司和政府用户的必要条件。一个所熟知的开发系统 GUI 的标准是 MIT 的 X Window 系统。X Window 系统成了 NIST、ANSI 和 IEEE 的标准。MIT 的 X 协会推出了一套说明以便图形用户接口应用系统能一起通讯和工作。这些说明列在了 ICCCM 之中,是应用系统间互操作的关键。

(3)开发工具:一个标准的 GUI 必须有一套应用广泛的开发工具。这些工具可以加速应用系统的开发,允许开发人员自定义扩充,为各种平台建立 GUI 应用系统。

(4)灵活性:标准的 GUI 必须有足够的灵活性和可扩充性,以适应新型的显示器和其他输入、输出设备。

(5)国际性:在全球市场中,国际性是达到应用系统可移植性的另一途径。包括: 其他国家的语言、计量单位、日期和时间格式、特殊符号和它国文化的消息特征。

(6)平以独立性:要达到真正的开放标准,GUI 必须独立于操作系统和硬件平台。同样,在网络环境中,标准 GUI 应独立于底层的网络协议。

三、GUI 特征

一般而言,在屏幕上 GUI 表示信息的矩形区域称为窗口。窗口相互间可以重叠。用户可以在窗口上执行多种操作,如改变大小和位置。窗口可以包含对象, 每个对象的图形表示称为图符。整个窗口可以缩小为一个图标,用户也可以将图符恢复成窗口。

高级的 GUI 几乎完全取消了键入命令的需求,用户只需使用鼠标和功能键进行选择就可以了。窗口还可以包含其他图形实体(如滚动条、滑动杆和按钮)以便用户控制窗口的内容,给应用系统提供额外输入。

和传统的程序设计相比,表示逻辑的最大差别是表示总使用户处于逻辑控制之中。因此传统的结构化程序设计所包括的输入、输出部分和处理部分要进行修改。GUI 程序设计必须能随时接受和处理用户或系统的异步事件。

1.事件类型

用户生成的输入事件和系统生成的事件随 GUI 实现的不同而有很大差异。共同的事件类型有:

(1)鼠标事件:用户将鼠标移进或移出一个实体、在实体内外按键、释放一个键。

(2)键盘事件:用户按下或释放一个键。

(3)菜单事件:用户从菜单中选择一个命令。

(4)窗口更改事件:应用系统窗口的某个部分被重叠时必须重画。

(5)窗口大小的更改:用户更改窗口的大小。

(6)激活、挂起事件:由 GUI 生成以便用户改变当前活动的窗口。

(7)启动、终止事件:创建、消灭 GUI 实体以便应用系统能进行必要的创建、删除逻辑。

2.事件分布

这些事件必须由表示逻辑和应用逻辑来协同处理,将必要的处理分布到了 GUI、应用逻辑以及特定的 GUI 应用程序设计接口(API)之中。一般而言,API 是特定的 GUI 库例程,执行下列功能:创建窗口和显示各种图形。因而有下列几种事件分布处理模型:

(1)事件循环模型:这种模型而言,应用子系统必须包括一个事件循环,由事件循环调用一个特定的库例程检查是否有事件发生。每个产生的事件引起应用系统在控制返回给事件循环之前发出一个事件处理例程。为了使用户感到总是在控制之中,应用系统必须迅速返回事件控制甚至在事件处理还没有完成的情况下就返回。

(2)事件回调模型:这模型需要应用系统为每个 GUI 创建的实体注册一个事件处理函数,因而就从前面重要的事件循环中释放了应用系统。在 GUI 检查到实体的一个事件时(如菜单命令或击键),它就调用合适的应用系统事件例程。应用系统仅在实体初始化或调用其事件处理例程时取得控制。

(3) 混合模型:这个模型将事件循环模型和事件回调模型组合了起来。Microsoft Windows 就利用了一个混合模型(其中应用系统必须包含一个事件循环)来调用例程取得下一个事件。一个应用系统也可以调用另一个

API 例程,API 例程反过来可调用系统的事件处理器。

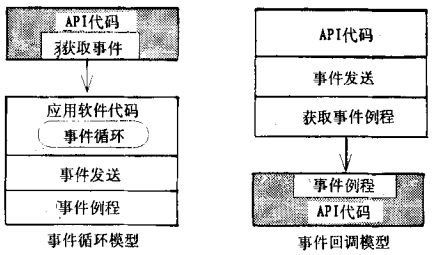

图 1　事件循环及事件回调模型

3.GUI 输出特征

GUI 的输出特征随 GUI 的不同而有很大差异:

(1)坐标空间描述了两维坐标系统以便 GUI 通过定义图空间的开始点和分辨率来确定屏幕上单个象素的位置。大多数 GUI 将开始点放在显示器的左上角,而坐标向右向下增加,在某些接口中,如 OS/2 的 Presdntation Manager,开始点放在左下角,而坐标向右向上增加。不同 GUI 的坐标系统在分辨率上也有差异,分辨率以每英寸点的个数来度量(DPI),一般有 75DPI、100DPI 等。对字符终端而言,点就是一个字符。应用系统必须要考虑到坐标空间和分辨率,这样才能在屏幕上画出正确的图形。

(2)画图算法描述了特定 GUI 画线、中心线以及联接线的方式。差别有时是比较显著的。例如两个点间的三个象素宽的一条线,既可以以端点为中心,也可以在端点上或端点下。当然,在低分辨的平台上,线看起来都一样,但在高分辨率的显示器上,线就可能中断或重叠。

(3)色彩影响着 GUI 画的图的"观感"。各种显示设备的色彩有很大差异。GUI 本身对色彩的处理也不同,而且使用不同数目的位来表示色彩,这样就决定了色彩的精确程序。

(4)文本表示在图形方式下和字符方式下有很大差异。在 GUI 环境中,文本以图形方式来处理,而且有很大的选择范围。这些参数包括色彩、字符大小、字体和风格。

四、目前 GUI 的产品

OSF.Motif 和 Open Look 均基于 X Windows 系统,是目前在工作站上用得最广泛的 GUI 系统。除此之外,还有不少其他 GUI 产品,比较有名的有:Macintosh 是最早、最成熟的公用 GUI。人们花了很大代价不断改进 Macintosh 接口。Macintosh 目前在图形和出版系统中非常流行。Microsoft 的 Windows 也是一个很流行和成熟的 GUI。尽管 Microsoft Windows 限制在了 MS DOS 平台上,但 3.0 版扩充了 DOS 的能力,实际上消除了 DOS 的大部分缺陷。OS/2 的 Presentation Manager 是由 Microsoft 和 IBM 联合开发的,基本上是基于 MS Windows,在 OS/2 下运行。提供了较强的 GUI 功能,避免了 DOS 的限制。DECWindows 是 DEC 的 UNIX(Ultrix)和 VMS 系统的 GUI。它建立在 X Windows 系统之上,是 OSF/Motif 的基础。DECWindows 增加了许多有用的 GUI 特征,如文本域、按钮、滚动杆和标准的对话框。NeXTGUI 提供了非常好的灰度显示,并引入了光盘的使用,扩大了 GUI 的能力。但是由于缺少软件和价格太高而没有得到推广。尽管字符和块方式的终端显示器不如高分辨率的 GUI,但它们仍能提供某些标准的 GUI 机制,如窗口、菜单、正文域、检查框,特别是目前仍有大量的这类设备在使用。

客户、服务器体系结构并没有限制在高分辨率终端的特定实现。因此,能在这些平台上让用户开发和运行有同样表示"观感"的应用软件是很有益的,另一个好处是将应用软件从一个平台上移植到另一个平台上而勿需改变 GUI 的观感。

一种方法是使用工具使不同 GUI 之间的差别对应用软件是透明的。

五、兼容工具

兼容工具的目的是提供所有 GUI 的共同基础。有了这样的工具,开发人员就能在给定的平台上生成一个想要的图形用户接口,而不用学习几种不同的接口。这意味着根据所使用的特定 GUI,应用软件代码执行该 GUI 的 API 功能集,这个集合随 GUI 的不同而差别很大。例如,当一个使用 OSF/Motif GUI 的应用软件要移植到支持 MS Windows GUI 的 DOS 平台上时,应用软件不需修改。在这种情况下所有 GUI 交互都通过一个公共的工具 API 来处理。沟通各种 GUI 实现的工具必须考虑 API(Application Programming Interfaces)之间的差别,包括各种 GUI 所用的事件处理逻辑、坐标空

(下转第 30 页)

(上接第 58 页)

间、画图算法、色彩、正文以及字体。同时,这些工具应尽可能保留 GUI 的观感。这些工具的另一个需求是抛掉"最小共同基础"的方法,它简单消除了一个特定 GUI 的所有优势,因此这样的工具必须至少提供一个和最先进的 GUI 一样丰富的功能集。

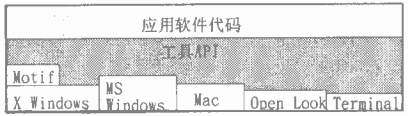

图 2

对给定的各种层次的 GUI 接口而言,要创建一个兼容的工具应支持其他几个需求:

(1)使用工具编写的应用软件至少要能在某个 GUI API 上运行。

(2)工具应支持将来的 GUI、新的输入/输出和显示设备以及多媒体,工具应具有灵活性和可扩充性。目前,已经有若干厂商在开发这样的兼容产品了。如 Oracle 公司正致力于其兼容工具 Oracle Toolkit。目前,Oracle Toolkit 为下列 GUI 提供一个公共 API: Macintosh,Microsoft Windows,X Window, OSF/ Motif 以及字符和块终端。这样,工具的另一个例子是 XVT 公司的 XVT(Extensible Virtual Toolkit)。XVT 包含了一个库集和"Include"文件,它在 GUI 顶部又构造了一层。为了将开发者和每个 GUI 处理的结构、字体、窗口、进程间的通讯隔离开,XVT 在一个实际工具集中实现了自己的接口,并带有一个资源编译器。兼容工具如 Oracle Tookit 和 XVT 均有助于开发人员跟上 GUI 技术的进步,并有利于应用软件的移植和表示的一致性。

参考文献:

[1]Alex Berson,"Client/ Server Architecture", Mc-Graw-Hill Inc.,1992.

[2] 郭江、张荣肖,"人机接口屏幕格式设计分析",《计算机系统应用》1993.9