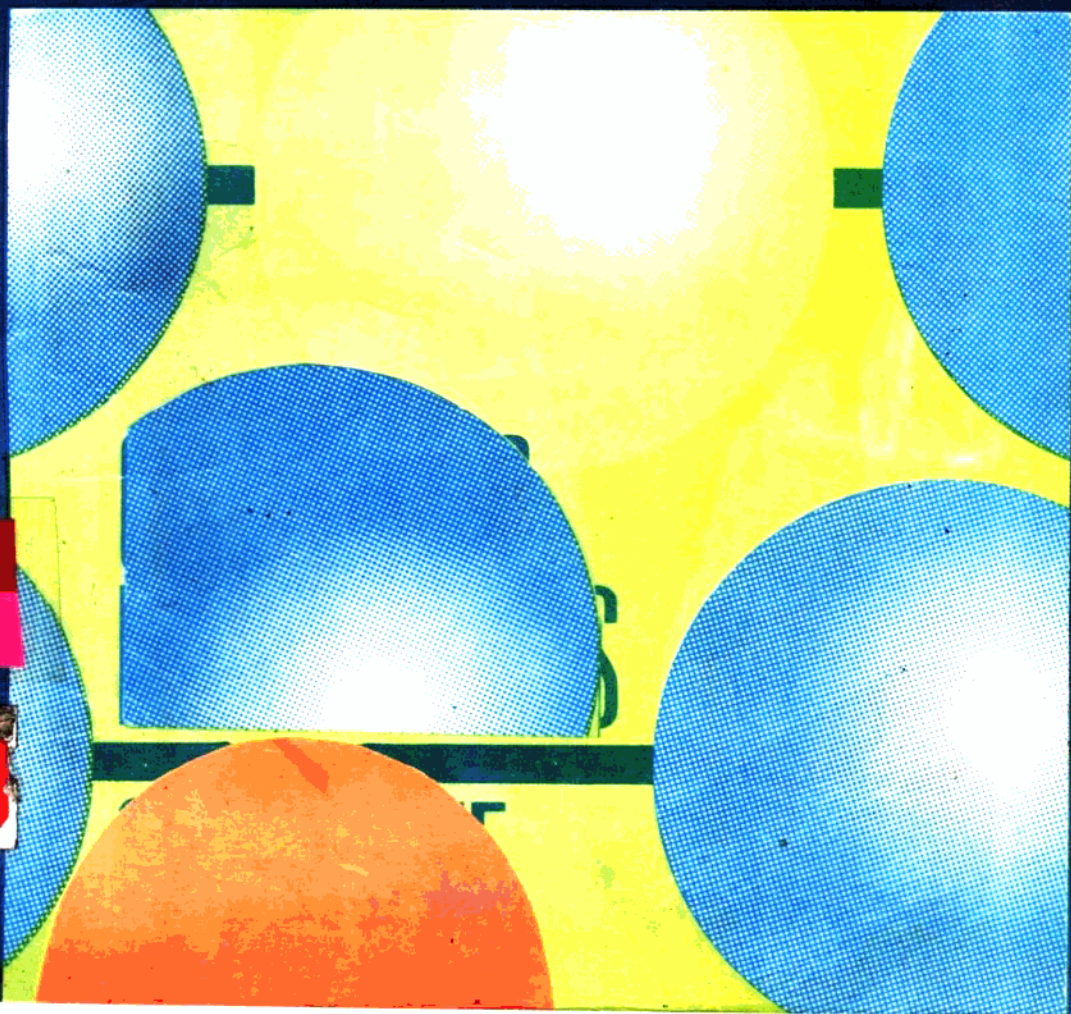




HOPE

如何编写 X—Windows 系统下的应用程序



北京希望电脑公司

如何编写X-Windows系统下的应用程序

荣 堂

北 京 希 望 电 脑 公 司

一九九一年九月

版权所有
翻印必究

■ 北京市新闻出版局

准印证号：3562—91562

■ 订购单位：北京8721信箱资料部

■ 邮 码：100080

■ 电 话：2562329

■ 传 真：01—2561057

■ 乘 车：320、332、302路

车至海淀黄庄下车

■ 办公地点：希望公司大楼一楼

往里走101房间

译者序

本书是学习在X-Windows环境下编写应用程序的一本难得的参考书。作者根据其多年在X-Windows环境下的编程经验,从X-Windows的基本概念出发,深入浅出地介绍了X-Windows系统应用编程的基本方法及技术。它既可作为X-Windows程序员的入门教材,也可作为X-Windows系统及应用开发人员的日常参考书,还可供大专院校的师生参考。

目前,开发X-Windows系统下的应用程序已成为一种国际潮流,X-Windows将成为九十年代主要的软件开发环境,相信本书的出版,对加速我国的X-Windows系统下的应用开发,培养大批X-Windows下的应用程序开发人员,会起到积极的推动作用。这也是译者翻译本书的最大心愿。

译者

目 录

引言

1. X-Windows的历史.....	(2)
2. 为什么现在流行 X-Windows.....	(2)
3. 什么是X-Windows	(4)
4. X-Windows的基本构成	(5)
5. 运行X-Windows 需要什么.....	(6)
6. 本书对X-Windows的处理方法	(6)

第一部分 学习X-Windows编程

第一章 构造第一个 X-Windows 程序	(7)
1.1 例1: 一个最小的X-Windows程序.....	(8)
1.1.1 例1的源码.....	(9)
1.2 例2: 开一个最简单的窗口.....	(13)
1.2.1 创建一个窗口	(14)
1.2.2 向窗口管理程序发“暗示”(Hints).....	(17)
1.2.3 卸(Flushing)显示器	(18)
1.2.4 关闭窗口	(18)
1.2.5 例2的源码.....	(19)
1.3 例3: 增添开窗函数的内容.....	(23)
1.3.1 Icons(肖像)	(23)
1.3.2 例3的源码.....	(25)
1.3.3 Icon 的位象	(28)
1.4 小结	(29)
第二章 用X-windows画图	(30)
2.1 画线和矩形	(30)
2.2 图形上下文	(31)
2.2.1 为窗口创建一个图形上下文	(31)
2.3 例1: 画线和矩形.....	(32)
2.3.1 例1的源码.....	(32)
2.4 通过弧画椭圆	(43)
2.5 填充椭圆和矩形	(43)
2.6 例2: 画并填充椭圆和矩形.....	(44)
2.6.1 例2的源码.....	(44)
2.7 画多条线、多个矩形及圆弧	(48)
2.8 小结	(51)

第三章 颜色的使用	(51)
3.1 彩色应用的移植性及有用性	(51)
3.2 X-Windows中的颜色映象	(52)
3.3 例1: 用红色画一条线	(53)
3.3.1 例1的源码	(54)
3.4 例2: 使用更多的颜色, 且重复使用	(60)
3.4.1 例2的源码	(60)
3.5 小结	(67)
第四章 使用文本	(67)
4.1 字型	(67)
4.1.1 装入字型	(68)
4.1.2 建立图形上下文	(68)
4.1.3 写文本	(69)
4.1.4 文本写在何处	(70)
4.1.5 找到字型的大小	(70)
4.1.6 释放字型	(71)
4.1.7 使用字型时的小麻烦	(71)
4.1.8 找到可用的字型	(72)
4.1.9 有限的X服务器资源	(75)
4.2 范例程序	(75)
4.2.1 范例程序的源码	(75)
4.3 小结	(79)
第五章 事件	(80)
5.1 事件驱动型编程	(80)
5.2 X-Windows事件的模型	(80)
5.3 要求事件	(81)
5.4 事件Mask	(81)
5.5 从X-Windows服务器接收事件	(82)
5.6 轮询事件	(83)
5.7 事件类型	(84)
5.8 鼠标按钮事件	(84)
5.9 键盘事件	(87)
5.10 进入/离开事件	(88)
5.11 暴露(exposure)事件	(89)
5.12 例1: 一个检查事件的程序	(90)
5.12.1 例1的源码	(91)
5.13 例2: 一个基于鼠标的画图程序	(97)
5.13.1 例2的源码	(98)
5.14 用xev程序获取有关事件模型的更多信息	(104)

5.15 小结	(104)
第六章 键盘事件	(105)
6.1 Meta (Alt) 键	(109)
6.2 鼠标按钮	(110)
6.3 范例程序	(112)
6.3.1 范例程序的源码	(112)
6.4 键盘映象什么时候变化	(119)
6.5 小结	(120)
第七章 Rubber-Band线	(120)
7.1 光栅操作函数	(121)
7.2 GC图形函数小结	(126)
7.3 例1: Rubber-Band线	(126)
7.3.1 例1的源代码	(128)
7.4 例2: Rubber-Band 椭圆	(136)
7.4.1 例2的源代码	(136)
7.5 例3: 显示光栅操作函数	(141)
7.5.1 例3的源代码	(146)
7.6 小结	(152)
第八章 X-Windows 标准及与窗口管理程序的相互作用	(153)
8.1 X-Windows命令行参数	(153)
8.2 显示器名	(154)
8.3 X-Windows 几何	(154)
8.4 为窗口创建光标	(155)
8.4.1 作为文本的光标	(156)
8.4.2 创建你自己的光标	(159)
8.4.3 释放光标资源	(160)
8.5 例1: 接收命令行参数及建立光标	(160)
8.5.1 例1的源码	(164)
8.6 例2: 向窗口管理程序发送更多信息	(180)
8.6.1 类型性质 (Class Property)	(181)
8.6.2 改变窗口尺寸	(181)
8.6.3 Icon名	(183)
8.6.4 例2的源码	(183)
8.7 小结	(189)

第二部分 构造一个X-Windows系统应用程序

第九章 介绍一个绘图应用程序Draw: 用户指南	(190)
9.1 浮动式选择查找表(Palette)	(191)

9.1.1	点	(192)
9.1.2	文本	(192)
9.1.3	装进新文件	(192)
9.1.4	恢复(Undo)绘图操作	(193)
9.2	命令行参数	(193)
9.3	熟练用户Meta键的妙用(shortcut)	(194)
9.4	关键错误的处理	(194)
9.5	处理致命性I/O错	(196)
9.6	小结	(197)
第十章	多窗口应用程序	(168)
10.1	浮动的查找表(Palette)	(198)
10.1.1	加亮一个查找表选择	(207)
10.1.2	刷新查找表窗口	(208)
10.1.3	查找表光标	(211)
10.2	绘图窗口	(216)
10.3	清除X-Windows及GC资源	(218)
10.4	Draw应用程序的文件格式	(219)
10.5	Pixmap及窗口	(221)
10.5.1	XCopyPlane	(221)
10.5.2	用Pixmap恢复	(222)
10.6	小结	(224)
第十一章	Draw应用程序中的事件处理	(224)
11.1	事件mask及选择事件	(226)
11.2	绘图事件	(227)
11.3	查找表选择事件	(236)
11.4	文本输入事件	(239)
11.5	键盘妙用事件	(244)
11.6	X-Windows服务器生成事件	(247)
11.7	小结	(249)
第十二章	弹出窗口及对话框	(250)
12.1	在X-Windows中建立一个弹出窗口	(251)
12.2	从查找表中调用弹出窗口	(253)
12.3	确定鼠标定位光标的位置	(254)
12.4	一个弹出文件名对话框	(256)
12.5	实现对话框	(257)
12.6	建立对话框窗口	(261)
12.7	显示对话框窗口的内容	(263)
12.8	对话事件	(264)
12.9	清除	(267)

12.10 有关临时窗口的暗示	(268)
12.11 小结	(268)
第十三章 Draw应用程序的源码	(269)
Drawapp应用程序的源文件	(270)
第十四章 改进Draw应用程序	(349)
14.1 处理暴露事件	(349)
14.2 放大	(350)
14.3 上卷 (scrolling)	(351)
14.4 光标编辑	(351)
14.5 裁和贴/移动及拷贝区域	(351)
14.6 修剪 (cropping)	(351)
14.7 翻转 (flip)、反向 (reversing) 及旋转	(351)
14.8 小结	(351)

第三部分 X-Windows工具箱 (X-Windows Toolkit)

第十五章 X-Windows工具箱引论	(352)
15.1 Xt工具箱	(353)
15.2 一个样板Xt程序	(354)
15.2.1 初始化Xt库	(354)
15.2.2 创建子Widget	(355)
15.2.3 管理Widget	(356)
15.2.4 实现Widget	(356)
15.2.5 Widget事件循环	(356)
15.2.6 样板Widget程序的源码	(356)
15.2.7 编译Xt程序	(359)
15.3 小结	(359)

附录A X-Windows 颜色数据库	(360)
----------------------------------	---------

附录B X-Windows事件的类型及结构	(362)
1.ButtonPress, ButtonRelease	(363)
2.CirculateNotify	(364)
3.CirculateRequest	(364)
4.ClientMessage	(364)
5.ColormapNotify	(365)
6.ConfigureNotify	(365)
7.ConfigureRequest	(366)
8.CreateNotify	(366)
9.DestroyNotify	(366)

10.EnterNotify, LeaveNotify	(367)
11.Expose.....	(367)
12.FocusIn, FocusOut	(368)
13.GraphicsExpose, NoExpose	(368)
14.GravityNotify	(369)
15.KeyMapNotify	(369)
16.KeyPress, KeyRelease	(370)
17.MapNotify, UnmapNotify	(370)
18.PropertyNotify	(371)
19.ReparentNotify	(371)
20.ResizeRequest	(372)
21.SelectionClear	(372)
22.SelectionNotify	(372)
23.SelectionRequest	(373)
24.VisibilityNotify	(373)
附录C 图形上下文	(373)

引 言

这是一本学习X-Windows系统的书。该书是在我们为X-Windows系统写应用程序时特别需要,但又很难找到的那种书,它告诉我们如何简单易行地在X-Windows下编程。该书吸取了作者早年在X-Windows下移植应用软件的经验。当年作者在X-Windows环境下作软件移植时,翻阅了大量的有关X-Windows的书籍及文章。当时绝大部分材料只提供有关X-Windows库的调用方法方面的情况,尽管这些资料对高级X-Windows编程相当有价值,但它们既不能为X-Windows入门者提供恰如其分的引导,又不提供X-Windows编程的现实例子。

本引言将概述X-Windows的一些基本概念及结构,以及X-Windows系统的历史。在你阅读完该引言后,即使你还不能马上进入X-Windows编程实践也没关系,因为它仅仅是让你熟悉X-Windows的术语及环境。该引言中提到过的每件事情,都将会在以后的章节中作详细的叙述。

该书不是一本包含X-Windows的各个方面的参考手册。相反,它的目的,在于为你在X-Windows下创建实用程序提供一个开端。我们觉得,其它书籍中所缺乏的,正好就是如何将不同的X-Windows函数调用组织到一块:如怎样画rubber-band线(第七章),或者如何在显示器上用红色画一根线(第三章)。简言之,将这些复杂应用程序的简单构造物了解透对我们来说是十分艰难的。

X-Windows系统是一个大型复杂系统,其复杂性主要是由于X-Windows试图处理实际可能的各种不同类型的图形显示器。X-Windows还试图提供一个完整的图形系统——即足以创建Macintosh型的开窗界面,以及足以处理图形密集型的页式设计或计算机辅助设计软件包(CAD)。

本书将集中告诉你如何完成事情,以及如何马上将事情完成。X-Windows系统十分庞大且复杂,因而不可能在一本书中面面俱到。该书不是一本纯粹的参考书,它是一本描述你开始写X-Windows系统应用程序时所需要的关键成份的指导书。各章所介绍的关键内容,都将在该章的小结中列举出来,这样,你就可以在需要的时候查阅。

X-Windows系统在解决看似难以解决的问题方面显示了光明的前景。即如何在很多不同的,运行多种操作系统及带各种不同显示器的计算机之间提供一个共同的接口。X-Windows所提供的图形接口,可以在从IBM PCs至各种大型机及巨型机的各种机器上运行。Cray巨型机可以在DEC或Apollo工作站显示器上显示其输出。几乎每一个主要计算机厂商,从Apple Computer至Xerox,从Hewlett-Packard至Sun Microsystems,都在使用X-Windows系统。

该书主要面向商业及学术界具有C语言编程的工作经验,且想在下述几个方面做工作的入门者:

- 论证转向X-Windows系统的可行性
- 将公司软件移植到X-Windows下运行
- 为应用程序提供一个多厂商外观及感觉
- 学习有关最新的连网图形开窗系统
- 开始在X-Windows下作应用开发

1. X-Windows的历史

为了对X-Windows系统有一个概括的了解,你可能会发现了解有关X-Windows的历史,以及X-Windows系统设计者的初始目标是有益的。

1984年,麻省理工学院(MIT)的官员们面临一个对商业及学术计算界共同的问题:即他们通过捐赠或购买,获得了一大批互不兼容的工作站,而他们的目的,是要构造一个可用于教学目的的图形工作站网。面对这一大堆互不相同的操作系统及硬件厂商,MIT的官员们决定组织一个“雅典娜计划”(Project Athena),由MIT的一个开发小组与DEC及IBM联合完成。

雅典娜计划的目标,就是要设计一个网络,它既可以运行本地应用程序,又可以调用远程资源,该网络最初以斯坦福大学的W软件环境为基础构成。设计者们通过一个图形连网环境将IBMs, DECs及其它不同的工作站连接起来,创建了第一个真正硬件及厂商独立的操作环境,即X-Windows系统。

正如Jim Gettys, Bob Schieffler, 以及Ron Newman在他们合写的“X-Window System: C Library and Protocol Reference”(DEC Press, 1988)中所写的那样,开发小组在设计X-Windows时具有以下目的:

- 除非实现代码在完成一个实际应用程序时绝对需要,否则不增加任何新的功能。
- 确定系统不是什么与确定它是什么一样重要。不要满足世界上的一切需要,但要使系统可扩充,以便能以向上兼容的方式满足新的需要。
- 比从一个例子一般化更糟的唯一一件事就是随意捏造。
- 如果没有完全了解问题,最好是不要对它提供任何解答。
- 如果你只花了实现全部功能所需要努力的百分之十就获得了百分之九十的功能,那么还是使用简单的方法。
- 尽量将复杂性离散。
- 提供机制而不是办法。特别是,要将用户接口办法交给客户。

注意,在所有上述指南中,最后这一条是最适合于本书的;在论述X-Windows系统的原则时将会进一步讨论这一点。

X-Windows系统是一次成功。到1986年,X-Windows的消息已经传到了要求访问权的外部世界。作为响应,磁带介质上的X10.4发行,只收象征性费用(与UNIX的早期十分类似)。但是,对X10的外部开发放慢。很明显,版本11必然与版本10不兼容。

1988年3月,MIT正式发行版本11,发行2,即X11.2,尽管早在1987年9月发行过一个不完整的X11.1版本。目前的X-Windows系统的开发是由1988年1月形成的X联合会来主持的。

由于X联合会成员迅速介绍了几个基于X的产品,X-Windows系统已经在计算机新闻界十分流行。此外,1989年1月,开放软件基金会(OSF)采纳DEC, HP及Microsoft公司的建议,将X-Windows作为一个主要成份,融入它的图形用户界面—使UNIX更容易使用图形界面,从而使X-Windows获得一次飞跃。

2. 为什么现在流行X-Windows

在很大程度上讲,X-Windows是当今最恰当的窗口系统。在过去,小型机及其以下级别的硬件,不能以合理的价格提供功能强大的图形连网系统。随着强有力的工作站及微

机的出现，窗口系统已经成了一个最有生命力的产品。

此外，需要X-Windows这个开发工具还有一些其它原因。

(1) 灵活的开窗系统

当用户不断地习惯于窗口型界面时，窗口系统提供了一个平滑及灵活的用户界面。纵观Apple Macintosh的巨大成功以及对微机Microsoft-Windows的日益接受，加上Sun Microsystems' NeWs和DEC的DECWindows的流行，多年来研究表明，开窗界面比字符型界面更容易学习和使用。尽管X-Windows仅仅是窗口系统流行趋势的一部分，但它比上面提到的任何系统都具有更大的灵活性及潜力。

在很大程度上讲，那些系统是固连于一个硬件环境及一个厂商的。对软件设计及开发人员来说，现在的年代，是连接各种不同类型的计算机作开发的年代，因此，这样的限制显然会阻碍这种跨机开发的努力，而对X-Windows系统来说则不存在这个问题。X-Windows是一个操作系统独立及网络透明系统，通过将窗口管理程序及窗口服务器分开，可以不用昂贵的仿真卡和各种联网方案，就可以将不同的硬件连接起来。由于用户界面只作X-Windows调用，因而不存在对任何操作系统的依赖。

这就是X-Windows系统的设计者只提供机制，而不是办法的初始信条所导致的结果。软件设计者，在很大程度上可以说是用户，对界面的最终外观有确切的了解，通过编程，应用程序自身定义窗口界面及应用程序的“外观及感觉”。此外，你还可以建立任何你所喜欢的界面，要找Macintosh的肖像(icon)及窗口吗？没问题，喜欢NeWs的外观吗？更没问题。就X-Windows而言，界面的外观及感觉相对来说是不重要的。X-Windows提供机制，通过该机制，你可以构造很多不同风格的用户界面。

在讨论X-Windows系统及UNIX操作系统时，这点变得特别重要。在过去，UNIX被指责为是用户界面相当不友好和神秘的。在用X-Windows系统作为一个界面后，UNIX应用程序的使用变得更容易，实际上，对MS-DOS, VMS这样的操作系统，也有有关用户界面不友好的指责。

通过操作系统独立，X-Windows大大提高了软件的可移植性，称作XLib的标准XC库程序中的例程，在运行X-Windows的每台机器上是相同的，这就意味着，你的界面代码可以从一台机器上直接移植到另一台机器上。由于用户界面通常要占整个代码的30—60%，因此，你的应用程序就具有相当的可移植性。

(2) 共享资源

X-Windows系统允许几个程序同时共享鼠标，键盘及图形显示器这样的设备资源。实际上，就X-Windows而言，你的整个工作站就是一个显示器—显示器由一个键盘，一个定标设备（通常为鼠标），以及一个或多个监视器构成。多屏幕可通过键盘及定标设备连在一起工作。

(3) 产品系列的一致性

正如前面所提到的，X-Windows系统适合于几个不同的操作系统，包括UNIX, MS-DOS, A/UX, 以及VMS。X-Windows系统是一个标准的应用执行环境，为一台机器写的应用程序不必为另一台机器重写。

X-Windows系统已经被几家组成X联合会的大公司采纳为图形开窗环境。联合会的成员包括全成员Apollo Computer, Apple Computer, AT&T, Bell, Digital Equipment

Corp, Fujitsu, Hewlett-Packard, IBM, NCR, Sony, Sun Microsystems, Wang以及Xerox; 以及象Adobe Systems, Carnegie-Mellon University和Software Productivity Consortium这样的副成员。从本质上讲, X-Windows已经被主要的硬件制造商采纳为它们的一个开窗系统,

3. 什么是X-Windows

由于你已经知道了为什么X-Windows系统是一个如此惊人的操作环境, 因此, 我们现在要给你一些X-Windows系统的核心概念。

客户/服务器型体系结构

X-Windows系统的体系结构以一个简单的客户/服务器关系为基础。在这个关系中, 显示服务器是这样—一个程序, 它控制显示监视器, 并向它画所有输出, 跟踪客户输入, 以及相应地更新你的窗口, 而客户则是一些完成具体任务的应用程序。因为X-Windows是一个联网的环境, 因此, 客户和服务器不必在相同的系统上计算(尽管在许多情况下它们是在同一个系统上计算)。相反, X-Windows系统允许分布式处理。比如, 一个Apollo工作站可以运行一个X服务器, 并在网络内调用Cray巨型机的处理能力, 然后在Apollo监视器上显示Cray的计算结果。

这是你第一次——但不是最后一次遇到与现有计算机学术语稍有不同的X-Windows术语。在微机和小机网络领域, 服务器是一个在网络中心运行的硬件设备, 它向连网工作站和终端分布数据及处理能力, 但这个概念在X-Windows领域就不完全正确。在X-Windows范畴里, 服务器是一个控制显示器的本地软件程序。由于网络中的其它系统可以访问你的显示器, 因此, X-Windows服务器不能看成是局域网(LAN)中的文件服务器。

在X-Windows中, 显示器包括一个键盘, 定标设备(通常为鼠标), 以及一个或多个监视器, 它通常与一个计算机工作站相关。正如你在图0.1中所能看到的, 显示服务器跟踪多个输入, 允许用户运行几个不同的客户(象数据库管理程序、字处理程序, 以及图形应用程序等)。一个显示器可以运行由键盘和鼠标连接在一起的多个显示器, 但只要一个用户只局限于一个工作站, 多个屏幕也就构成一个显示器。

对本地或远程系统上运行的程序(通常叫做客户或应用程序)来说, 服务器就象一个交通中转站, 它要完成以下工作:

- 允许其它客户访问一个显示器
- 发送网络信息
- 从其它客户截取别的网络信息
- 实现二维画图, 使客户摆脱对大量图形操作的处理
- 跟踪客户间共享的资源(象窗口, 光标, 字型以及图形上下文)
- 允许分布式处理
- 如果X-Windows同多任务操作系统一块儿使用, 则允许多任务。(例如, 与UNIX一起使用时, X-Windows就允许你调用UNIX的多任务功能)。

在上面这些任务中, 最重要的, 也就是服务器从显示器获取输入, 然后将其通知给客户。在X-Windows中, 这种输入称作事件。当你按下一个键时, 那就是一个事件; 当你放开该键时, 那又是另外一个事件。类似地, 当你用鼠标操作光标时, 那又是另一个事件。这些事件都通过事件队列送给应用程序。事件将在第五和第六章中详细讨论。

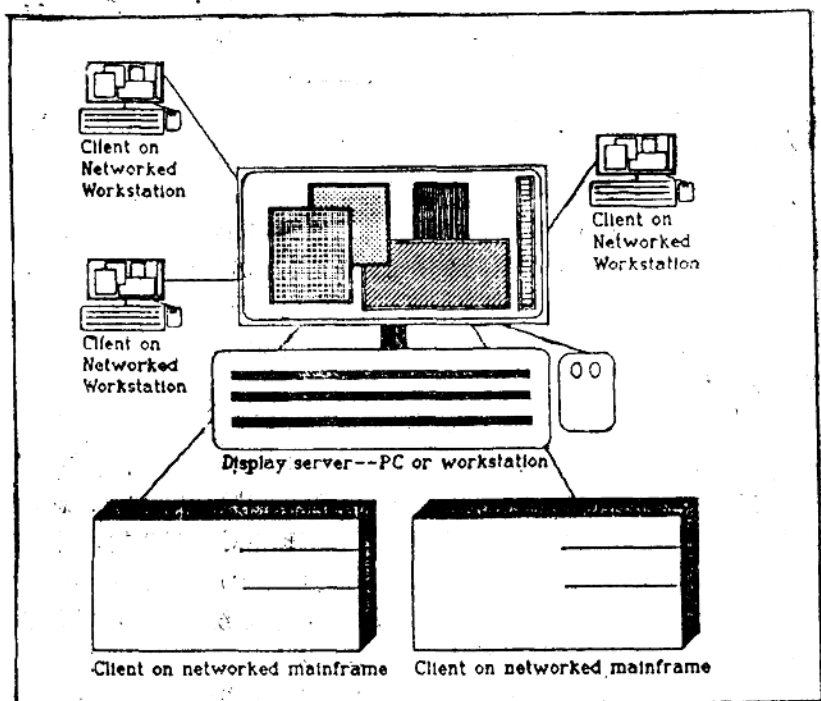


图 0.1 联网环境中的显示服务器

正如前面所提到的，客户是一些可以同时运行的应用程序。在本书中，“客户”和“应用程序”指的是同一个东西。依据网络配置的不同，一个客户可以同时使用几个服务器。有几种类型的客户；该引言中将介绍两种。

与Macintosh os这样的很多系统不同，窗口管理程序是这样一个客户，它监视你的显示器上窗口的大小和位置。所有X-Windows系统都带uwm，即通用或ULtrix窗口管理程序；uwm允许你改变窗口大小，安排你的窗口顺序，创建多个窗口，以及其它功能。在第八章中，我们将更深入地解释窗口管理程序。

由于X-Windows系统生成位象（bit-mapped）图，因此，它可以在多用户系统上安装终端仿真程序，而不需要购买工作站或个人计算机。例如，xterm就仿真DEC VT102或Tektronix 4014终端，并允许你用“dumb”终端运行多个应用程序。这样做，主要是为大量的ASCII终端程序提供兼容性。

4. X-Windows的基本构成

在很多方面，X-Windows是它的部件集合。先看看X-Windows可以分解成多少构件，然后你就可以懂得这些构件是如何在一起工作的。

由MIT发行的一般X-Windows系统包括XLib图形子程序库，X-Windows网络协议，一个X-Windows工具箱（Toolkit），以及几个窗口管理程序。应用程序员通过图形和窗口函数库XLib来连接客户程序。

（1）XLib

XLib大约包含300个例程,它们分别对应X-Windows协议请求或提供实用函数。XLib实际所做的事情,就是将C语言函数调用转换成实现给定功能的X-Windows协议请求,象XDrawLine画一根线。这些函数包括创建、毁坏、移动窗口或改变窗口大小;画线及多边形;设置背景图案;以及跟踪鼠标。XLib还允许你按不同方式访问窗口,包括向多个窗口的重叠或同时输出。它支持多种字型、通用的光栅操作、画线、以及彩色及单色应用等。

(2) X-Windows工具箱(Toolkit)

X-Windows工具箱是简化编程的程序子程序库。它们来自Hp, AT&T/Sun, IBM/Project Athena, DEC, 以及Sony等,各有一些差别,且都在不断修改当中,但这些不同的工具箱实现之间有许多类似之处。例如,工具箱都包含Scroll bars, buttons, 弹出菜单,窗口边界,对话框等。工具箱将在第十五章中更详细地介绍。

(3) X网络协议

X-Windows网络协议定义用于客户与服务器之间传输请求的数据结构。从技术上讲,X-Windows网络协议是一个异步流型进程间通信,它不以过程调用或核心调用接口为基础,应用程序在这里不做工作;协议是XLib的一个函数,这种结构可加速信息交换。

如果你有兴趣(不过坦率地说,在大多数情况下没有理由对这感兴趣。)协议说明可由MIT X-Windows联合会通过带提供,且在Robert Scheiffler所著的“X-Window System Protocol, Version 11”中定义。如果你在UNIX系统下工作,那么你可能已经具有该连机文档;它可在目录doc/Protocol/Spec中找到。

目前,X-Windows网络协议只在DEC net和TCP/IP上实现,尽管正计划将来在其它网络系统上实现。

5. 运行X-Windows需要什么

实际上,任何计算机,象微机、小型机、大型机等,都可以运行X-Windows。当你开始研究时,你就会发现市场上还没有X-Windows产品的机器已相当少。

这是厂商独立系统的一个很大优点。如果你不能在Sun工作站或DEC VAX上使用X-Windows,那还有在MS-DOS上运行的多种X版本。一般地说,在微机级上,你至少需要一个AT级带图形板及增强功能的机器,才能很好地运行X-Windows。此外,White Pine Systems为Macintosh操作系统实现了X。为了跟踪微机级X-Windows,你需要投入一定的精力,因为这个领域是不断变化的。

还有一个冲击市场的产品就是X-Windows终端。许多厂商,象Acer, Visual Technologies, Tektronix, NCD, 以及Graphon, 都发布了自己的X-Windows终端。这些终端通常采用Motorola 680X0,带1—2M内存。由于X-Windows是一个图形密集型系统,因此,你将需要X-Windows终端所提供的一切功能。X-Windows终端的价格通常在\$2000—\$4000之间。

6. 本书对X-Windows的处理方法

本书按15章组织,目的在于向你介绍X-Windows系统编程的关键概念。由于学习编程的最好方法是动手,因此,我们每章都包含一些源程序例子。在第一部分,各章所描述的是实现该章描述概念的独立程序。而第二部分构造的则是一个完整的X-Windows系统应用程序。第三部分则介绍X-Windows工具箱集。

第一部分 学习X-Windows编程

这部分从解释X-Windows系统开始,接着介绍在你创建你自己的X-Windows应用程序时需要知道的最基本内容。这些内容是:

- 如何建立与服务器的互联,接着是一个X-Windows范例程序,向你介绍X-Windows的理论及过程。
- 如何用X画图。X-Windows毕竟是一个图形界面,而大多数重要的图形应用程序都包含某种画图。
- 如何在你的应用程序中加入颜色,这其中包括对颜色数据库的描述。
- 如何画文本。如何通过X-Windows调用不同的字型也在讨论之列。
- 如何进入事件,事件是应用程序了解显示器所发生情况的手段。没有完全掌握事件,你就不能在X-Windows下编程。
- 如何编程画rubber-band线,这是任何鼠标型画图程序的基石。
- 如何与X窗口管理程序相互作用,同时还要讨论你如何确认这种相互作用确实发生。

第一章 构造第一个X-Windows程序

这一章要讨论如何构造你的第一个X-Windows程序,即打开与X-Windows服务器的互联,并报告某些与服务器有关的信息。该程序的代码由三个C文件构成: initX.c, quitX.c 和 test1.c, 然后从第一个程序再构造两个程序。第二个程序是开一个最简单的窗口,第三个程序则是卸走(Flushing)OpenWindow函数,并增加一个icon肖象。

本书将全部遵循这种类似的结构,任何程序的不同部分都将分成模块,目的在于以后可重复使用一些模块。每一章给出的源代码,都是较前一章有改变的。随着更多新概念的引入,某些章节可能会向这些文件增加一些新代码。

X-Windows是一个面向网络的窗口系统,它包括服务器和客户两部分。服务器控制显示屏,而客户则是一些象文本编辑、电子表格、时钟显示这样的应用程序。客户请求服务器的服务,象开窗口,在窗口内画线或文本等,拥有屏幕的是服务器。客户通过一个互联,通常是网络型互联,与服务器通信。因此,在任何X-Windows程序中,第一步是必须先建立一个与服务器的互联。

XOpenDisplay 调用建立一个与服务器的网络型互联。根据你所用来运行X服务器的机器的不同,这种互联可能是一个TCP/IP socket(网套),DECnet互联,或者是本地机器上的一个共享内存连接。不过你不用担心,因为XopenDisplay会根据情况建立正确的互联。XopenDisplay以你要与之通话的显示器名为一个参数,典型的缺省显示器就是本地机器及显示器。下面就是打开与缺省显示服务器互联的标准代码。

```
# include <stdio.h>
# include <X11/Xlib.h>
:
Display = the Display, /*...A Pointer to a display structure */
theDisplay = XOpenDisplay (NULL);
```