



20317

UNIX 系统 V 第 4 版
程序员指南: X11/NeWS
图形窗口系统 NeWS

*UNIX[®] SYSTEM V
RELEASE 4*

*Programmer's Guide: X11/NeWS[®]
Graphical Windowing System
NeWS*



UNIX Software Operation

電子工業出版社

目 录

前言

0.1 引言	(1)
0.2 记号约定	(1)

第一章 导论..... (3)

1.1 引言	(3)
1.2 NeWS 程序设计概况	(3)
1.2.1 POSTSCRIPT 语言	(3)
1.2.2 NeWS 数据类型	(3)
1.2.3 NeWS 操作命令	(4)
1.2.4 X11/NeWS 服务器	(4)
1.2.5 客户-服务器通信	(4)
1.2.6 C 客户接口	(5)
1.2.7 画布	(5)
1.2.8 NeWS 成像模型	(6)
1.2.9 事件	(6)
1.2.10 存储器管理	(6)
1.2.11 颜色支撑	(7)
1.2.12 字型支撑	(7)
1.3 用于服务器的 POSTSCRIPT 语言文件	(7)
1.3.1 窗口管理程序	(8)
1.3.2 类	(8)
1.3.3 调试	(8)

第二章 画布..... (9)

2.1 引言	(9)
2.1.1 扩展的类型 canvatype	(9)
2.1.2 画布操作命令	(10)
2.2 坐标系统综述	(10)

2.3 创建和显示画布	(11)
2.3.1 创建画布	(11)
2.3.2 形成画布	(12)
2.3.3 设置当前画布	(12)
2.3.4 映射画布至屏幕	(13)
2.4 操纵画布	(14)
2.4.1 移动画布	(14)
2.4.2 透明与非透明画布	(16)
2.4.3 可保留和非保留画布	(19)
2.4.3.1 关于破拟的进一步信息	(20)
2.4.4 关键字 SaveBehind	(21)
2.5 父辈、子辈以及同辈画布	(21)
2.5.1 同辈表	(21)
2.5.2 建立新辈	(24)
2.6 覆盖式画布	(24)
2.6.1 在覆盖上绘图	(25)
2.7 画布剪切操作命令	(25)
2.8 光标	(26)
2.9 画布、文件和成像过程	(27)
2.9.1 把画布写入文件	(27)
2.9.2 从文件中读出画布	(27)
2.9.3 其它与文件相关的操作命令	(28)
2.9.4 成像	(28)
2.10 其它的字典关键字	(29)
2.10.1 事件	(29)
2.10.2 颜色	(30)
2.10.3 X 专有特征	(30)
2.10.4 抓取状态	(30)
2.10.5 文件共享	(30)

第三章 事件

3.1 引言	(31)
3.1.1 扩展的类型 eventtype	(31)
3.1.2 事件操作命令	(32)
3.2 事件发布概况	(32)

3.3 创建事件	(33)
3.4 表示兴趣	(34)
3.4.1 表示兴趣前拷贝事件	(34)
3.4.2 更改和重用兴趣点	(34)
3.5 事件与兴趣点的匹配规则	(35)
3.5.1 关键字 Name 和 Action 的匹配规则	(35)
3.5.2 关键字 Process 的匹配规则	(35)
3.5.3 关键字 Serial 的匹配规则	(36)
3.6 事件送入发布机构	(36)
3.7 等待事件	(36)
3.8 关键字 Name、Action 和 Canvas 表示为字典	(37)
3.8.1 不可执行的字典值	(37)
3.8.2 可执行的字典值	(39)
3.9 使用关键字 Canvas 匹配多个兴趣点	(40)
3.9.1 两种兴趣表	(40)
3.9.2 兴趣点的匹配顺序	(41)
3.9.2.1 关键字 Canvas 为单个画布	(41)
3.9.2.2 关键字 Canvas 为数组或字典	(41)
3.9.2.3 关键字 Canvas 为空值	(41)
3.9.3 使用关键字 EventsConsumed	(42)
3.9.4 多个 Post-child 兴趣点匹配的实例	(43)
3.9.5 多个 Pre-child 兴趣点匹配的实例	(46)
3.10 系统生成事件	(47)
3.10.1 鼠标事件	(47)
3.10.2 进入和退出事件	(50)
3.10.2.1 关键字 Name 的值	(50)
3.10.2.2 关键字 Action 的值	(53)
3.10.2.3 使用操作命令 Postcrossings	(53)
3.10.3 使用关键字 XLocation 和 YLocation	(53)
3.10.4 使用关键字 Coordinates	(55)
3.10.5 焦点事件	(55)
3.10.6 键盘事件	(55)
3.10.6.1 重复键字典	(56)
3.10.7 破损事件	(56)
3.10.8 废弃事件	(56)
3.10.9 ProcessDied 事件	(57)

3.11 使用关键字 ClientData	(57)
3.12 使用关键字 Priority	(58)
3.13 使用关键字 Exclusivity	(59)
3.13.1 使用操作命令 redistributevent	(60)
3.14 使用关键字 TimeStamp	(61)
3.14.1 使用操作命令 recallevnt	(62)
3.15 使用关键字 Process	(63)
3.16 多进程的输入同步	(64)
3.16.1 使用 blockinputqueue	(65)
3.17 事件登录	(65)

第四章 类

(67)

4.1 引言	(67)
4.2 基本术语和概念	(67)
4.2.1 类与实例	(67)
4.2.2 继承性与类树	(68)
4.2.2.1 超类与子类	(68)
4.2.2.2 直接超类	(69)
4.2.2.3 继承性	(69)
4.2.2.4 单继承性与多继承性	(69)
4.2.2.5 继承性数组	(70)
4.2.3 单继承性实例	(71)
4.2.4 术语摘要	(72)
4.3 创建新类	(73)
4.3.1 定义类	(73)
4.3.2 操作命令 classbegin	(74)
4.3.3 操作命令 classend	(74)
4.3.4 操作命令 redef	(74)
4.3.5 初始化新类	(74)
4.4 用操作命令 send 发送消息	(75)
4.4.1 send 的通常形式	(75)
4.4.1.1 send 中执行步骤	(75)
4.4.2 用 send 调用方法	(76)
4.4.3 嵌套式 send	(76)
4.4.4 用 send 创建新实例	(78)

4.4.5	send 的另一种形式	(78)
4.4.6	用 send 改变实例变量的值	(78)
4.4.7	用 send 改变类变量的值	(79)
4.5	伪变量 self 和 super	(79)
4.5.1	伪变量 self	(81)
4.5.2	super 伪变量	(82)
4.5.3	super 将消息沿着超类链上传	(83)
4.5.4	self 和 super 使用限制	(84)
4.6	方法编译	(84)
4.6.1	编译 self send	(84)
4.6.2	编译 super send	(85)
4.6.3	本地字典	(85)
4.6.4	控制方法编译	(86)
4.6.5	/methodcompile	(86)
4.6.6	/installmethod	(87)
4.6.7	/doit	(87)
4.6.8	SetLocalDicts	(88)
4.7	创建新实例	(90)
4.7.1	/new	(90)
4.7.2	/newobject	(91)
4.7.3	/newinit	(92)
4.7.4	/newmagic	(93)
4.8	内部类	(94)
4.8.1	/newdefault	(95)
4.8.2	/defaultclass	(95)
4.8.3	/SubClassResponsibility	(96)
4.9	用 UserProfile 取代类变量	(96)
4.9.1	取代 DefaultClass	(97)
4.10	将类变量提升为实例变量	(97)
4.10.1	promote	(97)
4.10.2	unpromote	(98)
4.10.3	promoted?	(98)
4.10.4	避免意外的提升	(98)
4.11	销毁类和实例	(99)
4.11.1	/destroy	(99)
4.11.2	classdestroy	(99)

4.11.3	/cleanoutclass	(99)
4.12	类系统中的废弃对象	(99)
4.12.1	/obsolete	(100)
4.13	多重继承性	(100)
4.13.1	简单的多重继承性例子: 一个公用类	(101)
4.13.2	复杂多重继承性的例子	(103)
4.13.2.1	有效继承性数组次序的规则	(104)
4.13.2.2	例中可能的继承性数组	(104)
4.13.2.3	次序选择	(106)
4.13.2.4	限制继承性数组的次序	(107)
4.13.2.5	super 与多重继承性	(107)
4.14	设置和找取类的 Name 和 Classname 的公用程序	(108)
4.14.1	/name	(108)
4.14.2	/setname	(108)
4.14.3	/classname	(109)
4.15	询问对象状态的公用程序	(109)
4.15.1	isobject?	(109)
4.15.2	isclass?	(109)
4.15.3	isinstance?	(109)
4.16	询问对象继承者的公用程序	(109)
4.16.1	/superclasses	(109)
4.16.2	/subclasses	(110)
4.16.3	/instanceof?	(110)
4.16.4	/descendantof?	(110)
4.16.5	/understand?	(110)
4.16.6	/class	(110)
4.17	send 堆栈上查找对象的公用程序	(110)
4.17.1	/topmostinstance	(111)
4.17.2	/topmostdescendant	(111)
4.17.3	/sendtopmost	(111)
4.18	类操作命令	(111)
4.19	类方法	(112)

第五章 客户-服务器界面 (113)

5.1	引言	(113)
-----	----	-------

5.2	CPS 设施	(113)
5.3	创建.cps 文件	(114)
5.3.1	变元类型	(115)
5.3.2	发送 POSTSCRIPT 语言代码而无返回值	(116)
5.3.3	接受同步回答	(117)
5.3.4	接收异步回答	(119)
5.4	创建.h 文件	(120)
5.4.1	CPS 公用程序	(120)
5.5	创建.c 文件	(121)
5.5.1	POSTSCRIPT 语言通信文件	(122)
5.5.2	读客户的输入队列	(122)
5.6	记号与记号表	(123)
5.6.1	编译.c 文件	(124)
5.6.2	注释	(124)
5.7	调试 CPS	(124)
5.8	用其它语言支持 NeWS	(125)
5.9	字节流格式	(126)
5.9.1	压缩记号的编码	(126)
5.9.2	对象表	(127)
5.9.3	对照值	(128)
5.9.4	实例	(128)

第六章	调试	(131)
6.1	引言	(131)
6.2	装载调试器	(131)
6.3	启动调试器	(131)
6.4	使用调试器	(131)
6.4.1	多进程调试	(132)
6.5	客户命令	(132)
6.6	用户命令	(133)
6.7	杂项功能	(137)
6.7.1	别名	(137)
6.7.2	多个调试连接	(138)

第七章 存储管理	(139)
7.1 引言	(139)
7.2 引用计数	(139)
7.2.1 对象	(139)
7.2.2 被计数对象的引用	(139)
7.2.2.1 被计数的引用	(140)
7.2.2.2 非计数引用	(140)
7.2.2.3 软引用	(140)
7.2.2.4 废弃事件	(141)
7.2.2.5 引用记分	(141)
7.2.3 对象类型	(141)
7.3 存储管理操作命令	(142)
7.4 存储管理的调试操作命令	(143)
7.4.1 使用 debugdict	(143)
7.4.2 调试操作命令	(143)
7.5 未用的字型高速缓存	(146)
7.5.1 指定高速缓存的容量	(146)
7.5.2 清除高速缓存	(147)
7.5.3 应用程序	(147)
第八章 NeWS 类型的扩展	(149)
8.1 引言	(149)
8.2 作为字典的 NeWS 类型	(149)
8.3 NeWS 类型表	(150)
8.3.1 POSTSCRIPT 语言类型	(150)
8.3.2 NeWS 的扩展类型	(150)
8.4 类型 colortype	(151)
8.5 类型 graphicsstatetype	(151)
8.6 类型 monitortype	(151)
8.7 类型 packedarraytype	(152)
8.8 类型 pathtype	(152)
8.9 类型 canvastype	(152)
8.10 类型 colormaptype	(157)

8.11 类型 colormapentrytype	(158)
8.12 类型 cursortype	(158)
8.13 类型 enviromenttype	(159)
8.14 类型 eventtype	(161)
8.15 类型 fonttype	(165)
8.16 类型 processtype	(165)
8.17 类型 visualtype	(169)

第九章 NeWS 操作命令的扩展	(171)
-------------------------------	--------------

第十章 POSTSCRIPT 语言文件的扩展	(205)
-------------------------------------	--------------

10.1 引言	(205)
10.2 初始化文件	(205)
10.3 用户创建的扩展文件	(206)
10.3.1 其它扩展文件	(206)
10.3.2 扩展文件内容	(207)
10.4 杂类	(207)
10.5 数组操作	(211)
10.6 条件操作命令	(213)
10.7 输入操作命令	(213)
10.8 矩形公用程序	(216)
10.9 类操作命令	(217)
10.10 图形公用程序	(217)
10.11 文件访问实用程序	(219)
10.12 CID 实用程序	(220)
10.13 常量	(221)
10.14 键映射公用程序	(221)
10.15 重复键	(222)
10.16 标准颜色	(223)
10.17 登录事件	(223)
10.17.1 未登录的事件	(224)

附录 A NeWS 操作命令	(225)
A.1 引言	(225)
A.2 按字母顺序排列的 NeWS 操作命令	(225)
A.3 按功能排列的 NeWS 操作命令	(230)
A.3.1 画布操作命令	(230)
A.3.2 事件操作命令	(231)
A.3.3 数学操作符	(231)
A.3.4 进程操作命令	(232)
A.3.5 轨道操作命令	(232)
A.3.6 文件操作命令	(233)
A.3.7 颜色操作命令	(233)
A.3.8 键盘与鼠标操作命令	(234)
A.3.9 光标操作命令	(234)
A.3.10 字型操作命令	(234)
A.3.11 杂项操作命令	(234)
附录 B 扩展的输入系统	(237)
B.1 基于 NeWS 输入设施	(237)
B.2 Lite 用户界面	(237)
B.3 键盘输入	(238)
B.3.1 键盘输入: 简单的 ASCII 字符	(238)
B.3.1.1 撤消对键盘事件的兴趣	(239)
B.3.2 键盘输入: 功能键	(239)
B.3.3 指派功能键	(240)
B.3.4 键盘输入: 编辑与光标控制	(240)
B.4 选择	(241)
B.4.1 选择数据结构	(241)
B.4.2 选择进程	(243)
B.4.3 选择事件	(245)
B.5 输入焦点	(248)

附录 C 删节和实现限制	(251)
C.1 操作符删节和限制	(251)
C.2 映像删节	(252)
C.3 Statusdict 字典	(252)
C.4 实现限制	(252)
C.5 与 POSTSCRIPT 语言的其它区别	(253)

图与表

图 2-1: 在 0.0 处映射的画布	(14)
图 2-2: 在 25,25 处映射的画布	(15)
图 2-3: 一个被映射的子画布	(17)
图 2-4: 透明的父画布	(18)
图 2-5: 非透明并重涂色的父画布	(18)
图 2-6: 非保留画布上的破损	(20)
图 2-7: 年幼的同辈遮掩了年长者	(23)
图 2-8: 年长的同辈遮掩了年幼者	(23)
图 2-9: 画布间辈份的改变	(24)
图 2-10: 画布剪切操作的结果	(26)
图 2-11: 映射的画布	(28)
图 2-12: 用 buildimage 操作命令映射的画布	(29)
图 3-1: 画布的初始外观	(45)
图 3-2: Pre-Child 兴趣点匹配后的结果	(46)
图 3-3: 多个 Pre-Child 兴趣点匹配后的结果	(47)
图 3-4: FirstCanvas 和 SecondCanvas 的初始外观	(51)
图 3-5: 第一个进入事件, 被 FirstCanvas 匹配	(52)
图 3-6: 第二个进入事件, 被 SecondCanvas 匹配	(52)
图 3-7: 第二个进入事件, 被 FirstCanvas 匹配	(52)
图 3-8: 鼠标生成事件的结果	(54)
图 4-1: 一个简单的类树	(69)
图 4-2: 一个带有多继承性的类树	(70)
图 4-3: 一个单继承性的例子	(71)
图 4-4: 一个嵌套式 send 执行之前和其间的字典堆栈	(77)
图 4-5: 多继承性例子中基本的类层次结构	(101)

图 4-6: 具有 utility 类的层次结构	(102)
图 4-7: LabeledDialog 例子中的类树	(104)
图 4-8: LabeledDialog 的继承性数组中的广度优先次序	(105)
图 4-9: LabeledDialog 的继承性数组中的深度优先次序	(106)
表 3-1: 边界跨越事件	(53)
表 3-2: 输入焦点	(55)
表 4-1: 术语摘要	(73)
表 5-1: CPS 变元类型	(116)
表 5-2: CPS 所提供的 C 公用子程序	(121)
表 5-3: 记号值	(128)
表 5-4: 编码实例中字节的意义	(129)
表 7-1: 非计数对象的类型	(141)
表 7-2: 计数对象的类型	(142)
表 8-1: POSTSCRIPT 语言中的标准对象类型	(150)
表 8-2: 附加的 NeWS 对象类型	(151)
表 9-1: 发送给 incanvas 及其父画布的事件	(191)
表 9-2: 发送给 outcanvas 及其父画布的事件	(192)
表 9-3: 光栅操作代码值	(200)
表 10-1: 标准 NeWS 光标	(216)
表 B-1: 选择字典关键字	(242)
表 B-2: 系统定义的选择属性	(242)
表 B-3: 请求字典项	(243)
表 B-4: 高级的与选择有关的事件	(243)
表 B-5: 输入焦点	(248)
表 C-1: 实现限制	(252)
表 C-2: 各种 POSTSCRIPT 语言操作命令的 NeWS 版本	(253)

前言

0.1 引言

本手册是 NeWSTM 语言的编程指南。该语言是作为 X11/NeWSTM 服务器的一部分来提供的，而 X11/NeWSTM 服务器本身又是分布式窗口系统 OpenWindowsTM 的一个组成部份。

解释程序设计语言 NeWS 是以 POSTSCRIPT[®] 语言为基础的。

POSTSCRIPT 语言是由 Adobe 系统公司所开发，它是一种通用型程序设计语言，主要用来指明打印文本的视觉外观。NeWS 语言使用 POSTSCRIPT 语言的操作命令，来实现图形监视器上正文和图像的显示。不仅如此，重要的是 NeWS 语言还提供了扩充 POSTSCRIPT 语言的操作命令和数据类型；这些扩充的多数用于处理窗口管理的交互问题，后者 POSTSCRIPT 并未加以考虑。

本手册描述 NeWS 程序设计的所有基本概念。它也将提供每一 NeWS 操作命令的语法分析，包括一些编码实例用以说明 NeWS 扩充的操作命令和数据类型的用法。在此，假设读者已经熟悉了 POSTSCRIPT 语言。

关于使用 X11/NeWS 服务器的信息，请参阅：

- X11/NeWS 服务器指南
- X11/NeWS 版本注释

关于 OpenWindows 的信息，参阅：

- OpenWindows 用户指南
- OpenWindows 安装和运行指南

关于 POSTSCRIPT 语言的信息，参阅：

- POSTSCRIPT 语言教程和入门

(Adobe Systems 公司著，Adelison-Wesley 公司 1985 年 7 月出版)

- POSTSCRIPT 语言参考手册 (Adobe Systems 公司著，Addison-Wesley 公司 1985 年 7 月出版)

0.2 记号约定

本手册使用以下记号约定^①：

- 等宽字型，例如：`cdef`，该字型表示由键盘输入的正文或代码；并表示由计算

^①由于排版的原因，中文版与英文版记号约定略有不同。

机显示的信息；也用在代码举例和正文段中，以表示所用的是 C 程序设计语言。

- 普通印刷体字型，例如 `/result3`，该字型用于代码举例中，以表示所使用的 POSTSCRIPT 语言或 NeWS 扩充。
- 黑体字型，例如 **bold font**，该字型用于正文段中，以表示系统定义的 POSTSCRIPT 的或 NeWS 的对象名。
- 斜体字型，例如 *italic font*，该字型用于代码举例或正文段中，以表示加入程序或命令行的由用户所指明的参数。在 NeWS 的正文描述中，它也用来表示专用术语或短语。

第一章 导 论

1.1 引言

X11/NeWS 服务器既可用于单机系统中，也可用于由通信网络联接的多机系统中；因此，X11/NeWS 服务器是一个分布式窗口系统。当 X11/NeWS 服务器用于多机系统中时，在其中一台机器上执行的应用程序可以动用由另一台机器所显示出的窗口。

解释型程序设计语言 NeWS 是以 POSTSCRIPT 语言为基础的。POSTSCRIPT 语言是由 Adobe 系统公司所开发，它主要用来指明打印文本的视觉外观。一个 POSTSCRIPT 程序是由若干操作组成，后者被送给常驻在打印机内的 POSTSCRIPT 语言解释器；当它们被解释时，便定义正文、图形和页坐标。

NeWS 语言使用 POSTSCRIPT 语言的操作命令来在一台图形监视器上显示正文和图像。NeWS 程序由 X11/NeWS 服务器加以解释并执行；而后者则是常驻存在与图形监视器机相连的主机上的。值得一提的是，NeWS 语言还提供了扩充 POSTSCRIPT 语言的操作命令和数据类型，在这些扩充中，许多是用来处理窗口系统方面涉及交互式和多任务的部分，因为 POSTSCRIPT 语言本身不能对付这些。

1.2 NeWS 程序设计概况

本节介绍 NeWS 程序设计的概况。详细情况留待今后各章节来提供。

1.2.1 POSTSCRIPT 语言

POSTSCRIPT 语言是一种高级语言，它被设计用来向打印机描述页面的外观。它拥有大量的图形操作命令。然而该语言中专门用于图形操作的命令只占大约三分之一；其余的命令则用于一般的程序设计。

POSTSCRIPT 语言是可扩充的，因而允许程序员使用已提供的操作命令定义他们自己的进程，这种可扩充性便于产生模块化代码，促进形成结构良好且易于理解的程序设计，并且有助于保持程序的精练。

1.2.2 NeWS 数据类型

NeWS 语言实现了由 POSTSCRIPT 语言所提供的全部标准数据类型，此外，NeWS 语言还提供了 POSTSCRIPT 语言所不具备的其它的数据类型，作为扩充。

在这些 NeWS 扩充的数据类型中，有些可以当作正在 POSTSCRIPT 语言字典中那样地被动用。这些对象被称为魔术字典对象。这些魔术字典中具有着预先定义了名字的关

键字。许多关键字的值可以由程序员来改变；其它的则只能对其读。程序员可以为魔术字典增加新的关键字。

另一部分 NeWS 扩充的数据类型是非透明的，因而不能被当作字典来动用。NeWS 所扩充的全部数据类型的完整描述，将在第八章“NeWS 类型的扩展”中提供。

1.2.3 NeWS 操作命令

NeWS 语言实现了大部分 POSTSCRIPT 语言所提供的标准操作命令；那些未加以实现的操作命令有许多是涉及到对打印机页面的描述，与窗口系统无关。相反，NeWS 语言提供了许多 POSTSCRIPT 语言所不具备的扩充的操作命令；这些操作命令中许多与交互作用的要求有关，还有很多是用来产生和处理 NeWS 扩充数据类型的。

NeWS 所扩充的全部操作命令的完整描述，将在第九章“NeWS 操作命令的扩展”中提供。

1.2.4 X11/NeWS 服务器

X11/NeWS 服务器并不是一台用来提供文件服务的机器；而是一个进程。一个可以存在于网络中任何一台具有图形功能的机器上的进程。该进程的功能是：解释并执行用 POSTSCRIPT 语言编写的程序，并且在屏幕上显示图形结果。

X11/NeWS 服务器包含着多个微进程，其中一部分用来与客户进程通信。微进程并非 UNIX 进程；而是生存在服务器地址空间中并由服务器进行调度来执行的进程。每个微进程都可以在显示器上执行操作，并且可以从键盘、鼠标或别的微进程接收信息。一个微进程可以与另一个微进程共享数据。许多微进程只需较少的开销便可创建。微进程也被称为 NeWS 进程。

应注意到，X11/NeWS 服务器既不是一个工具箱，也不是一个用户界面；它既不提供标准的、亦不提供缺省的创建和显示窗口的方法。X11/NeWS 服务器只是简单地解释并执行由程序员编写的程序中所指定的 POSTSCRIPT 语言的操作和 NeWS 的扩充。程序员可以设计出各种完全不同的用户界面，而所有这些用 POSTSCRIPT 语言编写的用户界面都可以在 X11/NeWS 服务器上运行。

1.2.5 客户-服务器通信

X11/NeWS 服务器可以与本地执行的或进程执行的客户程序进行通信。客户向服务器发送 POSTSCRIPT 语言代码，服务器则为客户运行发来的代码。

一般来讲，客户程序主要由两个部分所组成。一部分可以用 C、FORTRAN 或其它高级语言编写，用来完成应用程序的基本运算任务，该部分就在客户进程中执行。另一部分则必须用 POSTSCRIPT 语言编写，用来提供相应的窗口和图形，该部分则由服务器进程来作解释。在客户程序中可以把 POSTSCRIPT 语言编写的那部分独立出来，发送到服务器，并通过函数调用来实现远程执行。