



UNIX 系统 V 第 4 版
程序员指南: 系统服务和应用软件打包工具

*UNIX® SYSTEM V
RELEASE 4*

*Programmer's Guide:
System Services and
Application Packaging Tools*



UNIX Software Operation

電子工業出版社

目 录

前言	(1)
0.1 目的	(1)
0.2 读者和所需的预备知识	(1)
0.3 组织	(1)
0.4 与 C 的关系	(2)
0.5 硬件/软件依赖性	(2)
0.6 记号约定	(2)
0.7 命令索引	(3)
0.8 示例信息	(3)

第一章 应用编程介绍	(1)
1.1 引言	(1)
1.2 应用编程	(1)
1.3 UNIX 系统工具以及有关资料	(2)
1.3.1 本指南中涉及的工具和未涉及的工具	(2)
1.4 UNIX 系统环境支持的语言及其有关资料	(2)
1.4.1 C 语言	(3)
1.4.2 shell	(3)
1.4.3 awk	(3)
1.4.4 lex	(4)
1.4.5 yacc	(4)
1.4.6 m4	(4)
1.4.7 bc 和 dc	(5)
1.4.8 curses	(5)
1.4.9 FMLI	(5)
1.4.10 ETI	(5)

1.4.11 XWIN 图形窗口系统	(7)
1.4.12 OPEN LOOK 图形用户界面	(7)

第二章 UNIX 系统环境中的应用编程

(9)

2.1 引言	(9)
2.2 系统调用	(9)
2.2.1 出错处理	(10)
2.2.2 基本的文件 I/O	(10)
2.2.3 高级的文件 I/O	(10)
2.2.4 终端 I/O	(11)
2.2.5 进程	(11)
2.2.5.1 进程概述	(12)
2.2.5.2 exec(2)	(13)
2.2.5.3 fork(2)	(14)
2.2.6 基本的进程间通信	(15)
2.2.7 高级的进程间通信	(15)
2.2.8 存储管理	(15)
2.2.9 文件系统控制	(16)
2.2.10 信号	(16)
2.2.10.1 信号概述	(17)
2.2.10.2 信号类型	(17)
2.2.10.3 信号处理程序	(18)
2.2.10.4 发送信号	(19)
2.2.10.5 保护临界区	(19)
2.2.10.6 信号栈	(20)
2.2.11 杂类系统调用	(20)
2.3 应用软件的开发	(21)
2.3.1 文件和记录加锁	(21)
2.3.1.1 何处能找到详细信息	(22)
2.3.2 进程间通信	(22)
2.3.2.1 何处能找到详细信息	(23)
2.3.3 进程调度	(23)
2.3.3.1 何处能找到详细信息	(23)

2.3.4 符号连结	(24)
2.3.4.1 何处能找到详细信息	(25)
2.3.5 存储管理	(25)
2.3.5.1 存储映象界面	(25)
2.3.5.2 何处能找到详细信息	(25)
2.3.6 数据合法性确认工具	(25)
2.3.6.1 何处能找到详细信息	(26)
2.4 软件包的开发和安装	(26)
2.4.1 打包应用软件	(26)
2.4.1.1 何处能找到详细信息	(27)
2.4.2 修改 sysadm 界面	(27)
2.4.2.1 何处能找到详细信息	(28)

第三章 文件和记录的加锁 (29)

3.1 引言	(29)
3.2 术语	(29)
3.3 文件保护	(30)
3.3.1 为记录加锁打开一文件	(30)
3.3.2 设置文件锁	(31)
3.3.3 设置和取消记录锁	(33)
3.3.4 获取加锁信息	(37)
3.3.5 死锁处理	(39)
3.4 选择咨询加锁或强制加锁	(39)
3.4.1 强制加锁使用注意事项	(40)
3.4.2 记录加锁和未来的 UNIX 系统版本	(40)

第四章 进程间通信 (41)

4.1 引言	(41)
4.2 消息	(42)
4.2.1 使用消息	(42)
4.2.2 得到消息队列	(45)

4.2.2.1 使用 msgget	(45)
4.2.2.2 实例程序	(47)
4.2.3 控制消息队列	(49)
4.2.3.1 使用 msgctl	(49)
4.2.3.2 实例程序	(50)
4.2.4 对消息的操作	(54)
4.2.4.1 使用 msgop	(54)
4.2.4.2 实例程序	(56)
4.3 信号量	(62)
4.3.1 使用信号量	(64)
4.3.2 得到信号量	(66)
4.3.2.1 使用 semget	(66)
4.3.2.2 实例程序	(68)
4.3.3 控制信号量	(71)
4.3.3.1 使用 semctl	(71)
4.3.3.2 实例程序	(72)
4.3.4 信号量上的操作	(79)
4.3.4.1 使用 semop	(80)
4.3.4.2 实例程序	(81)
4.4 共享内存	(84)
4.4.1 使用共享内存	(85)
4.4.2 得到共享内存段	(87)
4.4.2.1 使用 shmget	(87)
4.4.2.2 实例程序	(89)
4.4.3 控制共享内存	(91)
4.4.3.1 使用 shmctl	(92)
4.4.3.2 实例程序	(92)
4.4.4 对共享内存的操作	(97)
4.4.4.1 使用 shmop	(97)
4.4.4.2 实例程序	(98)

第五章 进程调度程序

(103)

5.1 引言

(103)

5.2 进程调度程序概述	(104)
5.2.1 分时类调度	(105)
5.2.2 系统类调度	(105)
5.2.3 实时类调度	(106)
5.3 命令与函数调用	(106)
5.3.1 priocntl 命令	(108)
5.3.2 priocntl 系统调用	(111)
5.3.2.1 PC-GETCID PC-GETCLINFO	(112)
5.3.2.2 PC-GETPARMS PC-SETPARMS	(116)
5.3.3 priocntlset 系统调用	(120)
5.4 与其他函数的相互作用	(123)
5.4.1 核心进程	(123)
5.4.2 fork, exec	(123)
5.4.3 nice	(123)
5.4.4 init	(123)
5.5 性能	(123)
5.5.1 进程状态转换	(124)
5.5.2 软件延时	(125)
5.5.3 用于实时 U 块的内存	(126)

第六章 符号连结

6.1 引言	(127)
6.2 符号连结的使用	(128)
6.2.1 符号连结的特征	(128)
6.2.2 符号连结的创建	(129)
6.2.2.1 语义规则和语法规则	(129)
6.2.2.2 例子	(130)
6.2.3 符号连结的删除	(131)
6.2.4 符号连结的存取	(131)
6.2.5 符号连结的拷贝	(131)
6.2.6 符号连结的连结	(131)
6.2.7 符号连结的改名	(132)
6.2.8 归档命令	(132)

6.2.9 文件属主和存取权限	(133)
6.2.10 在 RFS 中使用符号连结	(133)

第七章 存贮管理

(137)

7.1 虚拟存贮系统概述	(137)
7.1.1 虚存、地址空间和映射	(137)
7.1.2 网络、异机种和一致性	(138)
7.2 存贮管理界面	(138)
7.2.1 创建和使用映射	(139)
7.2.2 删除映射	(142)
7.2.3 高速缓存控制	(142)
7.2.4 其他映射函数	(145)
7.3 地址空间的布局	(145)

第八章 打包应用软件

(147)

8.1 软件打包概述	(147)
8.2 软件包的内容	(147)
8.2.1 必要的成分	(147)
8.2.2 可选的软件包信息文件	(148)
8.2.3 可选的安装脚本	(149)
8.3 软件包的结构生命周期	(149)
8.4 打包工具	(150)
8.4.1 pkgmk命令	(150)
8.4.2 pkgtrans命令	(151)
8.4.3 pkgproto命令	(151)
8.5 安装工具	(151)
8.6 软件包信息文件	(152)
8.6.1 pkginfo文件	(152)
8.6.2 prototype文件	(153)
8.6.2.1 描述行	(153)
8.6.2.2 命令行	(154)

8.6.3 compver文件	(155)
8.6.4 copyright文件	(156)
8.6.5 depend文件	(156)
8.6.6 space文件	(157)
8.6.7 pkgmap文件	(158)
8.7 安装脚本	(158)
8.7.1 脚本处理	(158)
8.7.2 安装参数	(159)
8.7.3 为脚本获取软件包信息	(160)
8.7.4 脚本的退出码	(160)
8.7.5 请求脚本	(160)
8.7.5.1 请求脚本的命名约定	(161)
8.7.5.2 请求脚本的使用规则	(161)
8.7.6 类动作脚本	(161)
8.7.6.1 类动作脚本的命名约定	(162)
8.7.6.2 类动作脚本的使用规则	(162)
8.7.6.3 类的安装	(162)
8.7.6.4 类的删除	(163)
8.7.7 特别系统类	(164)
8.7.7.1 sed 类脚本	(164)
8.7.7.2 awk 类脚本	(165)
8.7.7.3 build 类脚本	(165)
8.7.8 过程脚本	(165)
8.7.8.1 过程脚本的命名约定	(166)
8.7.8.2 过程脚本的使用规则	(166)
8.8 打包的基本步骤	(166)
8.9 指定软件包的缩写名	(168)
8.10 定义软件包实例	(168)
8.10.1 标识一个软件包实例	(168)
8.10.2 在你的脚本中存取实例标识符	(169)
8.11 编写安装脚本	(169)
8.12 使软件包对象可重定位	(169)
8.12.1 定义群体可重定位对象	(170)
8.12.2 定义个别可重定位对象	(170)
8.13 对象分类	(170)
8.14 定义软件包的依赖性	(171)

8.15	编写版权信息	(171)
8.16	在安装机器上保留附加空间	(172)
8.17	创建 pkginfo 文件	(172)
8.18	创建 prototype 文件	(172)
8.18.1	手工创建prototype文件	(173)
8.18.1.1	创建连接	(173)
8.18.1.2	映射开发路径名到安装路径名上	(173)
8.18.1.3	定义 pkgadd 创建的对象	(173)
8.18.1.4	使用命令行	(174)
8.18.2	用pkgproto创建文件.....	(174)
8.18.2.1	创建一个基本的 prototype 文件	(174)
8.18.2.2	为对象指定类	(175)
8.18.2.3	用 pkgproto 重新命名路径名	(175)
8.18.2.4	pkgproto 和 links	(176)
8.19	把软件包分布在多个卷上	(176)
8.20	用 pkgmk 创建软件包	(176)
8.20.1	创建软件包实例.....	(177)
8.20.2	帮助pkgmk确定软件包内容.....	(177)
8.21	用 pkgtrans 创建软件包	(177)
8.21.1	创建数据流软件包.....	(178)
8.21.2	转换软件包实例.....	(178)
8.22	打包过程的快速引用	(179)

第九章 修改 sysadm 界面

9.1	sysadm 修改的概述	(181)
9.1.1	工具介绍	(181)
9.1.1.1	edsysadm 命令	(181)
9.1.1.2	delsysadm 命令	(182)
9.1.1.3	数据合法性确认工具	(182)
9.1.2	打包修改文件的介绍	(183)
9.1.3	界面修改过程的概述	(183)
9.2	设计你的界面修改	(184)
9.2.1	决定是否你应该修改界面	(184)

9.2.2 设计你的修改位置	(184)
9.2.2.1 界面结构概貌	(184)
9.2.2.2 设计你的管理结构	(185)
9.2.3 命名你的界面修改	(186)
9.2.3.1 如何命名你的修改	(186)
9.2.3.2 界面命名的要求	(186)
9.2.3.3 系统如何处理命名冲突	(186)
9.3 编写管理操作	(187)
9.4 编写求助消息	(187)
9.4.1 条目求助文件	(187)
9.4.1.1 菜单条目求助的消息格式	(188)
9.4.1.2 默认标题格式	(188)
9.4.1.3 字段条目求助的消息格式	(189)
9.4.2 标题层次	(190)
9.4.3 在 FACE 对象中建立条目求助	(190)
9.4.4 条目求助文件的举例	(191)
9.5 打包你的界面修改	(193)
9.5.1 打包你的修改的基本步骤	(193)
9.5.2 为一个菜单项建立或改变打包	(193)
9.5.2.1 为一个菜单项建立打包	(193)
9.5.2.2 为一个菜单项改变打包	(194)
9.5.2.3 联机测试你的菜单改变	(195)
9.5.2.4 菜单定义表格	(195)
9.5.3 为一个任务项建立或改变打包	(196)
9.5.3.1 为一个任务项建立打包	(196)
9.5.3.2 为一个任务项改变打包	(197)
9.5.3.3 任务定义表格	(198)
9.6 准备你的软件包	(199)
9.7 删除界面修改	(200)

第十章 数据合法性确认工具

(201)

10.1 工具的介绍	(201)
10.2 工具的类型	(201)

10.3 工具的特性 (201)

10.3.1 数据合法性确认工具提示符..... (202)

10.3.2 数据合法性确认工具求助消息..... (202)

10.3.3 数据合法性确认工具的出错消息..... (203)

10.3.4 消息格式化..... (203)

10.3.5 shell 命令 (203)

10.3.6 直观工具..... (205)

附录 A liber，一个图书馆系统 (207)

附录 B 命令手册页 (219)

附录 C 软件包安装实例分析 (273)

C.1 引言 (273)

C.2 实例#1 (273)

C.2.1 技术 (273)

C.2.2 方法 (273)

C.2.3 样本文件 (274)

C.3 实例#2 (277)

C.3.1 技术 (277)

C.3.2 方法 (277)

C.3.3 样本文件 (278)

C.4 实例#3 (281)

C.4.1 技术 (281)

C.4.2 方法 (281)

C.4.3 样本文件 (282)

C.5 实例#4 (284)

C.5.1 技术 (284)

C.5.2 方法 (284)

C.5.3 样本文件 (285)

C.6 实例#5a	(286)
C.6.1 技术	(286)
C.6.2 方法	(287)
C.6.3 样本文件	(288)
C.7 实例#5b	(289)
C.7.1 技术	(289)
C.7.2 方法	(289)
C.7.3 样本文件	(290)
C.8 实例#5c	(292)
C.8.1 技术	(292)
C.8.2 方法	(292)
C.8.3 样本文件	(293)
C.9 实例#6	(294)
C.9.1 技术	(294)
C.9.2 方法	(294)
C.9.3 样本文件	(295)

图和表

图 1-1 一个简单的 ETI 程序	(7)
图 2-1 基本的文件 I/O 的系统调用	(10)
图 2-2 高级文件 I/O 的系统调用	(11)
图 2-3 终端 I/O 的系统调用	(11)
图 2-4 进程的系统调用	(12)
图 2-5 进程状态	(13)
图 2-6 fork 的例子	(14)
图 2-7 高级进程间通信的系统调用	(15)
图 2-8 存储管理的系统调用	(16)
图 2-9 文件系统控制的系统调用	(16)
图 2-10 信号的系统调用	(16)
图 2-11 杂类系统调用	(21)
图 4-1 ipc_perm 数据结构	(44)
图 4-2 操作权限代码	(46)
图 4-3 msgget 系统调用实例	(48)

图 4-4	msgctl 系统调用实例	(52)
图 4-5	msgop 系统调用实例	(59)
图 4-6	操作权限代码	(67)
图 4-7	semget 系统调用实例	(69)
图 4-8	semctl 系统调用实例	(75)
图 4-9	semop 系统调用实例	(82)
图 4-10	操作权限代码	(88)
图 4-11	shmget 系统调用实例	(90)
图 4-12	shmctl 系统调用实例	(94)
图 4-13	shmop 系统调用实例	(100)
图 5-1	系统 V 4.0 版的进程调度程序	(104)
图 5-2	进程优先级 (程序员角度)	(106)
图 5-3	PC_CETPARMS 的返回值	(117)
图 5-4	进程状态转换图	(124)
图 6-1	符号连结文件树	(129)
图 6-2	RFS 中的符号连结例 1	(134)
图 6-3	RFS 中的符号连结例 2	(135)
图 8-1	软件包的内容	(148)
图 8-2	样本 pkginfo 文件	(152)
图 8-3	样本 #1 prototype 文件	(154)
图 8-4	样本 #2 prototype 文件	(155)
图 8-5	样本 compver 文件	(155)
图 8-6	样本 copyright 文件	(156)
图 8-7	样本 depend 文件	(157)
图 8-8	样本 space 文件	(157)
图 8-9	把参数存放到安装环境中	(161)
图 8-10	sed 脚本格式	(164)
图 8-11	awk 脚本格式	(165)
图 9-1	一个表格的项目求助文件	(191)
图 9-2	多个表格的条目求助文件	(192)
图 10-1	shell 命令	(204)
图 10-2	直观工具	(205)
图 C-1	实例 #1 pkginfo 文件	(274)
图 C-2	实例 #1 prototype 文件	(275)
图 C-3	实例分析 #1 请求脚本	(276)
图 C-4	实例 #2 prototype 文件	(278)

图 C-5	实例#2 pkginfo 文件	(278)
图 C-6	实例#2 请求脚本	(279)
图 C-7	实例#2 postinstall 脚本	(280)
图 C-8	实例#3 pkginfo 文件	(282)
图 C-9	实例#3 prototype 文件	(282)
图 C-10	实例#3 space 文件	(283)
图 C-11	实例#3 安装类动作脚本 (i.admin)	(283)
图 C-12	实例#删除类动作脚本 (r.cfgdata)	(284)
图 C-13	实例#4 pkginfo 文件	(285)
图 C-14	实例#4 copyright 文件	(285)
图 C-15	实例#4 compver 文件	(286)
图 C-16	实例#4 depend 文件	(286)
图 C-17	实例#5a pkginfo 文件	(288)
图 C-18	实例#5a prototype 文件	(288)
图 C-19	实例#5a sed 脚本 (/sbin/inittab)	(288)
图 C-20	实例#5a postinstall 脚本	(289)
图 C-21	实例#5b pkginfo 文件	(290)
图 C-22	实例#5b prototype 文件	(290)
图 C-23	实例#5b 安装类动作脚本 (i.inittab)	(291)
图 C-24	实例#5b 删除类动作脚本 (r.inittanb)	(291)
图 C-25	实例#5c inittab 文件	(292)
图 C-26	实例#5c pkginfo 文件	(293)
图 C-27	实例#5c prototype 文件	(293)
图 C-28	实例#5c balld 脚本 (/sbin/init)	(294)
图 C-29	实例#6 pkginfo 文件	(295)
图 C-30	实例#6 prototype 文件	(296)
图 C-31	实例#6 安装类别动作脚本 (i.cron)	(296)
图 C-32	实例#6 删除类别动作脚本 (r.cron)	(297)
图 C-33	实例#6 根 crontab 文件 (与软件包一起传送)	(297)
图 C-34	实例#6 syscrontab 文件 (与软件包一起传送)	(297)

第一章 应用编程介绍

1.1 引言

本章介绍 UNIX 系统环境中的应用编程。

首先简单描述了什么是应用编程，然后讨论 UNIX 系统工具，并指出在哪些资料中介绍它们。然后再讨论 UNIX 系统环境所支持的语言以及应参考哪些资料。

在本章及本指南的其余各部分中，读者将发现要参考其他指南和手册，以查到更详细说明资料。特别是在很多地方都要参考《程序员指南：ANSI C 和编程支持工具》。《程序员指南：ANSI C 和编程支持工具》与《程序员指南：系统服务和应用软件打包工具》有密切的关系。这两本指南中许多信息与 UNIX 3.2 版中一本《程序员指南》中的相同。而 UNIX 4.0 版则分成了两本指南。

本指南集中于从应用程序员观点来看如何在 UNIX 系统 V 下利用核心提供的系统服务来开发和打包应用软件。

《程序员指南：ANSI C 和编程支持工具》描述 C 编程环境、库、编译程序、连接编辑和文件格式。它还描述了 UNIX 系统/C 环境所提供的建立、分析、调试和维护程序的工具集。

如果你不能肯定应参考哪本书，请查阅《产品概览和总索引》。该书说明了整套资料是如何组织的以及从那里可找到所需的信息。

1.2 应用编程

本指南所讨论的编程目的是为了产生在 UNIX 系统计算机上运行的应用程序。

在这个环境中工作的程序员所开发的应用程序是为其他非程序员用户服务的。大多数大型商业计算机应用需要一组应用程序开发人员。他们可以是最终用户机构的雇员，或者他们为某个软件开发公司工作。在这个环境中工作的某些人，更多的是从事项目管理，而不是从事编程人。

应用编程有下述特点：

- 应用常常是大型的，需要一组人员来编写需求说明、设计、测试和最终用户使用资料。这意味着要使用项目管理方法，包括版本控制（在《程序员指南：ANSI C 和编程支持工具》中描述），变更请求，跟踪等等。
- 应用必须开发得更健全些。

- 应用程序必须容易使用, 这意味着使用字符的或图形的用户界面。
- 必须检查所有输入数据的合法性(例如, 使用第十章所述的数据合法性工具)。
- 应能处理大量的数据。

- 应用必须易于安装和管理(参见第八章“打包应用软件”和第九章“修改sysadm界面”)。

1.3 UNIX 系统工具以及有关资料

让我们首先弄清“UNIX 系统工具”这个术语的含义。从最狭窄的含义来说, 它是指一些已有的软件, 它可用来作为一个新任务中的一部分。从较广的含义来看, 这个术语常用来指 UNIX 系统中那些基本成分, 可称为特性、实用程序、程序、过滤器、命令、语言、函数等。事情常会混淆不清, 由于用上述名字调用的任何程序常常是在狭义下使用, 作为解决编程问题的一部分。

1.3.1 本指南中涉及的工具和未涉及的工具

《程序员指南: 系统服务和应用软件打包工具》是关于在 UNIX 系统环境建立程序过程中所用到的工具的介绍, 所以我们首先要谈一下我们所指的工具是哪些, 哪些工具在本书中并不涉及, 以及在何处可找到本书未涉及的工具的有关信息。实际上, 本书未涉及的工具可能对你更为重要。我们在这一本书中不可能把你需要了解的 UNIX 系统工具的所有信息都包括进来。

本书没有涉及的工具具有:

- login过程
- UNIX系统编辑程序和如何使用
- 文件系统如何组织及如何使用
- shell程序设计

关于这些题目的材料可在《用户指南》和许多市场上买得到的课本中找到。

本书所涉及的工具用于应用程序。尽管每个应用程序执行不同的功能, 但经历同样的基本步骤: 输入, 加工和输出。这些工具帮助你完成这些步骤。

本书还包括打包应用软件和裁剪用户界面用的工具。

1.4 UNIX 系统环境支持的语言及其有关资料

本节描述 UNIX 系统环境支持的各种语言。

所谓“语言”是指由 AT&T 提供的在运行 UNIX 系统 V 当前版本的计算机上使用的语言。因为语言系统是单独购买的, 所以你的机器上不一定安装了所有的语言系统。另外,

你的机器上可能使用从其他公司购买来的语言系统，这些语言在本节中并不提及。

1.4.1 C 语言

C 和 UNIX 系统是紧密相连的，因为 C 原是为编写 UNIX 系统核心而研制的。如果你需要使用一些 UNIX 系统功能调用来实现低级 I/O，存储管理或设备管理，或进程间通信时，C 是最适合用的语言。但是大多数程序并不需要与操作系统直接打交道，之所以决定选择 C 语言是基于 C 的下述各种特性：

- 各种数据类型：字符，各种长度的整数，以及浮点数。
- 低级结构(大多数UNIX系统核心是用C编写的)。
- 导出数据类型，如有数组，函数，指针，结构和联合。
- 多维数组。
- 定位指针(scaled pointer)，并且指针可作算术运算。
- 逐位运算。
- 各种流程控制语句：if, if-else, switch, while, do-while, 以及for 语句。
- 高度的可移植性。

使用 C 的困难在于需要集中使用几个月以上才能达到自如运用而成为 C 程序员。如果你不是专职程序员的话，你最好还是选用对用户要求较低的语言比较省劲些。

关于 C 语言的详细说明请参阅《程序员指南：ANSI C 和编程支持工具》

1.4.2 shell

你可以使用 shell 来建立程序(新的命令)。这种程序称为 shell 过程。《用户指南》中介绍了如何利用命令、变量、位置参数、返回码和基本的程序控制结构来建立和执行 shell 程序的方法。

1.4.3 awk

awk(这个名字是由开发者名字的首字母构成的缩略字)根据一个说明文件中描述的模式扫描输入文件找出相匹配的行。找到一个相匹配的行时对它执行什么动作也是在说明文件中描述的。常常是由几行组成的 **awk** 程序所完成的工作，用 **FORTRAN** 或 **C** 这类编程语言来描述的话需要几页。例如，考虑下述情况：有一组记录，每一记录由一个键字段和一个表示数量的第二个字段组成。任务是要输出与每个键相对应的数量的总和。这个程序的伪代码如下：