国外计算机科学教材系列

UNIX系统编程

（第二版）

UNIX System Programming

Second Edition

[英] Keith Haviland
Dina Gray
Ben Salama 著

舒明 熊战波 等译

Addison-Wesley

PHEI

電子工業出版社
Publishing House of Electronics Industry
www.phei.com.cn

国外计算机科学教材系列

UNIX 系统编程

（第二版）

UNIX System Programming
Second Edititon

Keith Haviland
[英] Dina Gray 著
Ben Salama

舒 明 熊战波 等译

電子工業出版社
Publishing House of Electronics Industry
北京 · BEIJING

内 容 简 介

本书以清晰而层次分明的方式，给出了UNIX平台下C语言的编程方法和技巧，主要针对当前日趋流行的面向分布环境的IT解决方案。本书反映了UNIX操作系统的标准和本质，重点讲解了操作系统内核（UNIX中真正的操作系统部分）、运行于UNIX环境系统中程序之间的系统调用接口以及UNIX系统所带的一些重要的库函数。本书通过对系统调用和库函数的剖析，使读者亲身实践UNIX下的编程过程，并且对UNIX系统内核有更进一步的了解。

本书可作为理工类大专院校计算机相关专业高年级本科生和研究生教材，对于从事UNIX程序开发工作的软件工程师也有很好的参考价值。

版权贸易合同登记号：图字：01-2001-1458

图书在版编目（CIP）数据

UNIX系统编程（第二版）/（英）哈维兰（Haviland, K.）著；舒明等译. -北京：电子工业出版社，2003.1
（国外计算机科学教材系列）
书名原文：UNIX System Programming, Second Edition
ISBN 7-5053-7644-6

Ⅰ.U... Ⅱ.①哈... ②舒... Ⅲ.①UNIX操作系统-程序设计 ②C语言-程序设计 Ⅳ.TP316.81

中国版本图书馆CIP数据核字（2002）第102349号

责任编辑：马 岚 杜闽燕
印 刷 者：北京天竺颖华印刷厂
出版发行：电子工业出版社 http://www.phei.com.cn
北京市海淀区万寿路173信箱 邮编：100036
经 销：各地新华书店
开 本：787 x 1092 1/16 印张：16.75 字数：418千字
版 次：2003年1月第1版 2003年1月第1次印刷
定 价：24.00元

凡购买电子工业出版社的图书，如有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系。
联系电话：（010）68279077

出 版 说 明

21世纪初的5至10年是我国国民经济和社会发展的重要时期，也是信息产业快速发展的关键时期。在我国加入WTO后的今天，培养一支适应国际化竞争的一流IT人才队伍是我国高等教育的重要任务之一。信息科学和技术方面人才的优劣与多寡，是我国面对国际竞争时成败的关键因素。

当前，正值我国高等教育特别是信息科学领域的教育调整、变革的重大时期，为使我国教育体制与国际化接轨，有条件的高等院校正在为某些信息学科和技术课程使用国外优秀教材和优秀原版教材，以使我国在计算机教学上尽快赶上国际先进水平。

电子工业出版社秉承多年来引进国外优秀图书的经验，翻译出版了“国外计算机科学教材系列”丛书，这套教材覆盖学科范围广、领域宽、层次多，既有本科专业课程教材，也有研究生课程教材，以适应不同院系、不同专业、不同层次的师生对教材的需求，广大师生可自由选择和自由组合使用。这些教材涉及的学科方向包括网络与通信、操作系统、计算机组织与结构、算法与数据结构、数据库与信息处理、编程语言、图形图像与多媒体、软件工程等。同时，我们也适当引进了一些优秀英文原版教材，本着翻译版本和英文原版并重的原则，对重点图书既提供英文原版又提供相应的翻译版本。

在图书选题上，我们大都选择国外著名出版公司出版的高校教材，如Pearson Education培生教育出版集团、麦格劳－希尔教育出版集团、麻省理工学院出版社、剑桥大学出版社等。撰写教材的许多作者都是蜚声世界的教授、学者，如道格拉斯·科默（Douglas E. Comer）、威廉·斯托林斯（William Stallings）、哈维·戴特尔（Harvey M. Deitel）、尤利斯·布莱克（Uyless Black）等。

为确保教材的选题质量和翻译质量，我们约请了清华大学、北京大学、北京航空航天大学、复旦大学、上海交通大学、南京大学、浙江大学、哈尔滨工业大学、华中科技大学、西安交通大学、国防科学技术大学、解放军理工大学等著名高校的教授和骨干教师参与了本系列教材的选题、翻译和审校工作。他们中既有讲授同类教材的骨干教师、博士，也有积累了几十年教学经验的老教授和博士生导师。

在该系列教材的选题、翻译和编辑加工过程中，为提高教材质量，我们做了大量细致的工作，包括对所选教材进行全面论证；选择编辑时力求达到专业对口；对排版、印制质量进行严格把关。对于英文教材中出现的错误，我们通过与作者联络和网上下载勘误表等方式，逐一进行了修订。

此外，我们还将与国外著名出版公司合作，提供一些教材的教学支持资料，希望能为授课老师提供帮助。今后，我们将继续加强与各高校教师的密切联系，为广大师生引进更多的国外优秀教材和参考书，为我国计算机科学教学体系与国际教学体系的接轨做出努力。

电子工业出版社

教材出版委员会

译 者 序

无论是 UNIX 编程初学者还是高级程序员，都需要掌握两方面的内容：操作系统提供的系统调用和其他的一些函数库，这些函数库最终都是通过系统调用实现的。可以说系统调用是 UNIX 下所有程序的基石。目前市面上流行的一些关于 UNIX 编程的书，有些是着重于 UNIX 原理的介绍，另外一些基本上是对系统调用接口的语法描述，前者过于理论化，而后者又过于肤浅，缺乏相关的技术分析。本书详尽地描述了 UNIX 编程中常用的系统调用与应用程序的接口，并且在如何应用这些系统调用、如何提高程序的运行效率方面也有精辟的论述。在阐述某些系统调用接口的同时，更是深入到操作系统内核，分析了内核中相关的数据结构以及它们对系统调用接口本身的影响。更难能可贵的是，本书还介绍了不同 UNIX 系统标准下（如 XSI、POSIX 等）相近系统调用之间的细微区别，对编程者提高程序的可移植性有极大的帮助。在介绍系统调用的同时，本书还穿插了大量的实例与习题，这些实例和习题都非常有针对性，有些就是如何实现 UNIX 系统下的一个实际工具，不但加深了读者对系统调用的理解，并且更加全面和深刻地阐述了整个 UNIX 系统的概貌。

本书第 1 章到第 4 章以及第 6 章由舒明负责翻译，第 8 章到第 12 章由熊战波负责翻译，第 5 章和第 7 章由孙勇负责翻译，最后舒明做了全书的统稿工作。参加本书翻译工作的还有刘亚君、余义军、杜波、张衍圣、艾望峰等，此外在本书的成稿过程中得到了魏永明的大力帮助，在此表示衷心感谢。

前　言

本书宗旨

自 1969 年在贝尔试验室诞生以来，UNIX 操作系统已经得到了极大的发展。UNIX 系统最早只是用于一些研究机构，然后才在 20 世纪 80 年代发展成为新一代小型机和微机的标准操作系统。在撰写本书时，UNIX 系统还在不断地发展中。

本书的第一版最早完成于 1987 年。至今十几年过去了，UNIX 一直都没有停止其前进的步伐，它已经成为 21 世纪技术领域一道亮丽的风景。除了保持在科学界和工程界的强劲势头之外，UNIX 平台又新增了许多大型数据处理功能和事务处理功能，可以说是因特网的脊梁，因为很多服务器都是建立在 UNIX 系统上的。

《UNIX 系统编程》一书像 UNIX 系统一样也取得了很大的成功，自第一版出版以来，每年都要重印一到两次。本书非常有针对性，它着重反映了 UNIX 系统的标准和本质。但是，自第一版以来，UNIX 系统又有了很大的改变，读者迫切期盼本书（第二版）的面世。本书将涉及到 20 世纪 90 年代末非常流行的更加分布式的 IT 解决方案。

在第二版中我们将一如既往地坚持本书所制定的目标。与第一版一样，我们会着重介绍操作系统内核（UNIX 中真正的操作系统部分）和运行于 UNIX 环境系统中程序之间的编程接口，该接口通常称为系统调用接口（尽管现在系统调用接口与通常的库函数之间的区分并不是非常明显）。我们会看到，这些系统调用正是构建 UNIX 程序（不管是系统附带的程序还是第三方所开发的程序）的基石。本书面向的读者是具有基本 UNIX 经验并且将来有志于用 C 语言开发 UNIX 程序的程序员。所有立志开发 UNIX 程序的人，无论开发的是系统级软件还是应用程序，或是商业软件，都能从本书获益匪浅。

除了系统调用之外，本书还将重点介绍 UNIX 系统所带的一些重要的库函数。这些库函数都是用系统调用写出来的，并完成与系统调用类似的功能，但是相比之下这些函数更高级，也更易于使用。

X/OPEN 规范

UNIX 系统已经有很多年的历史了，存在着许多正式的和事实上的 UNIX 标准，并且还有许多商业和学术界的变种。但是 UNIX 系统的内核部分却保持着相对的稳定性，本书涉及到的就是这些相对稳定的部分。

《UNIX系统编程》的第一版是以 AT&T 的System V Interface Definition（系统 V 接口定义，SVID）第二期为基础的，系统 V 是 UNIX 系统实现中流行最为广泛的一个版本。《UNIX 系统编程》的第二版以 1994 年的 X/OPEN 的第四期第二版的内容和实例为基础，包括系统接口定义、系统接口和头文件，还有一些与网络服务有关的文档。为了方便起见，这里把 X/OPEN 第四期第二版简记为 XSI。我们会在本书必要的地方讨论一些与实际实现相关的问题。

在这里，我们需要提到一些背景知识。X/OPEN 最早只不过是一些关注开放系统和 UNIX 平台

的硬件提供商组织，后来才逐渐扩大为现在这种规模。该组织的一个主要任务就是 UNIX 系统的标准化，另外还提供一个可移植 UNIX 系统的建议（XPG），许多主要的 UNIX 提供商[包括 Sun、IBM、Hewlett-Packard、Novell 和开放软件组织（OSF）] 都以 XPG 为自己 UNIX 系统的起点。我们通常又把它称为 1170 初始规范（1170 指的是可移植的 1170 个系统调用、头文件、命令和工具集）。该规范的最终目标是实现一个包括系统调用在内的单一完整的 UNIX 系统服务规范，系统调用也是本书的中心内容。这个规范综合了一些相互冲突的 UNIX 标准而形成了一套通用的、可实现的规范，它包括上面所提到的系统接口定义文档、系统接口文档和头文件文档等一些关键部分。其他文档包括 UNIX 命令和屏幕处理。

从系统编程的角度来讲，XSI 只不过是实现 UNIX 的起点，所以本书涉及到的绝大多数例子应该可以运行于当前大部分 UNIX 系统上。这个 X/OPEN标准，综合了一系列相关和互补的标准以及 UNIX 的实现部分。包括 ANSI/ISO C 标准、非常重要的基本 POSIX 标准（IEEE Std 1003.1-1990）、SVID、开放软件组织规范以及影响很大的 Berkeley UNIX（伯克利 UNIX）。当然标准化是一个不断发展的过程，X/OPEN 和 OSF 于 1996 年合并为开放组织（Open Group）。Open Group 把 POSIX 的最新规范也包括进来了，我们可以在 Open Group 的单一 UNIX 规范第二版中找到最新的 POSIX 规范。Open Group 的这个规范不但包括 POSIX 规范，也包括第五期规范的系统接口定义、系统接口、头文件和网络服务部分。本书还包括了一些非常重要但在一般情况下很少涉及的 UNIX 扩展规范，比如线程、实时和动态链接等规范。

最后应该注意的两点是：

1. 任何规范的涉及面都是相当广的，它包括许多功能相同的不同子规范，而且有些功能很少用到，但它们又非常重要。我们对 UNIX 编程规范做出了自己的取舍，并不准备讲解所有的 UNIX 标准。
2. 如果读者在编程中需要考虑标准的兼容性，可以检测一些系统设置和相关的标志，比如 _XOPEN_SOURCE 或 _POSIX_SOURCE。

本书的结构

本书按以下结构组织内容：

- 第 1 章简要介绍 UNIX 中的一些基本概念和术语，其中着重讨论了文件和进程。作者希望大多数读者在阅读本书之前对我们在下一节列出的知识有所了解。
- 第 2 章讨论的是有关文件操作的系统调用，包括打开、创建、读、写及随机访问一个文件。最后还讲解了系统调用的出错处理。
- 第 3 章将学习文件权限和属性，以及如何在 UNIX 下操作这些权限和属性。
- 第 4 章主要以程序员的角度看待 UNIX 的目录。还要了解 UNIX 的底层文件系统结构以及 UNIX 如何以特殊文件来表示设备。
- 第 5 章将研究 UNIX 进程控制。有关进程控制最重要的系统调用有 fork 和 exec，也是本章需要重点讲解的地方。将会有一个小小的 shell（命令解释程序）用来解释 UNIX 中的进程控制。
- 第 6 章介绍的是三种进程间通信手段中的第一种。包括信号和信号的处理，它们可用来捕获和通知进程的一些非正常状态。

- 第 7 章讨论 UNIX 中最有用的进程间通信技术——管道。管道可用来将一个进程的输出连接到另一个进程的输入。我们会讲解如何创建、读和写管道，以及如何在多个管道中选取一个管道。
- 第 8 章讨论 UNIX 系统 V 中的进程间通信手段，包括记录锁、消息传递、信号量和共享内存。
- 第 9 章在系统调用级上讨论终端，并举出一个关于如何使用伪终端的例子。
- 第 10 章简要讨论 UNIX 中的网络部分，以及如何在 UNIX 中使用套接字把消息从一台机器发送到另外一台机器。
- 第 11 章开始从系统调用转移到 UNIX 上的主要函数库。在这一章中我们会系统介绍非常重要的标准 I/O 函数库。标准 I/O 函数库比起系统调用来说提供了更多的文件处理机制。
- 最后，第12章将简要说明其他的系统调用和库函数，这些系统调用和库函数是非常重要的，本章也将讨论字符串处理、时间函数和内存管理。

读者需要掌握的预备知识

本书的主要目的是介绍 UNIX 系统调用接口，而不是介绍 UNIX 系统或者 C 语言。所以在阅读本书之前读者应该熟悉以下几点：

- 如何登录到 UNIX 系统；
- 如何在 UNIX 系统上用编辑器创建一个文件；
- UNIX 的目录树结构；
- 操作文件和目录的 UNIX 命令；
- 如何创建和编译 C 程序（包括程序分为几个源程序的情况）；
- 如何在 C 程序中用 I/O 函数 printf 和 getchar 输入输出；
- 怎样在 C 程序中使用命令行参数，也就是主函数中的 argc 和 argv 参数；
- 如何使用系统参考手册（由于一些软件商重新组织了自己的参考手册，所以本书无法给出一个比较统一详尽的介绍。传统的参考手册共分为八部分，每部分都按字母顺序排序，其中前三部分为：命令、系统调用和库函数）。

如果读者对上面几点还不熟悉，请先做下面的习题。

此外，编程不光是一个看的过程，本书自始至终非常重视习题和例子。在开始学习之前，应该确保拥有一台可以使用的 UNIX 机器。

习题 P.1 解释下面 UNIX 命令的作用：

```
ls cat rm cp mv mkdir cc
```

习题 P.2 使用自己喜欢的编辑器创建一个小的文本文件，然后使用 cat 命令创建另外一个文件，该文件内容为前面的小文件重复 5 遍。
使用 wc 计算两个文件的字符数和单词数。创建一个子目录，并把这两个文件移动到该子目录下。

习题 P.3 创建一个文件，此文件内容是 /bin 目录和你的 home 目录清单。

习题 P.4 写一个简单的命令显示当前系统登录用户总数。

习题 P.5 写一个 C 程序打印一条指定的欢迎信息，并编译和执行该程序。

习题 P.6 写一个程序打印出自己的参数，并编译和执行该程序。

习题 P.7 使用 getchar()和 printf()，写一个程序计算输入内容的单词总数、行数、字符数。

习题 P.8 创建一个文件，文件中有一个 C 函数打印信息“hello, world”。再创建一个包含主程序的单独文件。编译并执行这个程序（该程序命名为 hw）。

习题 P.9 在系统参考手册中查询下面的内容：cat 命令、printf 函数以及 write 系统调用。

目　录

第1章　基本概念和术语

本章将对本书可能用到的一些基本概念和术语进行简要的介绍。首先，我们来看看UNIX文件的含义。

1.1　文件

在UNIX系统中，信息是以文件形式存储起来的。UNIX中典型的文件操作命令有：

```
$ vi my_test.c
```

这个命令通过调用编辑器vi来创建和编辑文件my_test.c，接下来有：

```
$ cat my_test.c
```

cat可以把my_test.c文件的内容打印出来，输出到终端上，还有：

```
$ cc -o  my_test  my_test.c
```

cc通过调用C编译器编译源文件my_test.c，最终生成可执行文件my_test，当然，我们要假设my_test.c没有语法错误。

大多数文件都被创建它们的用户赋予了各种各样的逻辑结构。比如，一篇文档可能由单词、行、段落或页组成。对于系统来说，所有UNIX文件看起来都是由许多简单、无序的字节或字符组成的，用户可以通过调用系统所提供的一些原始接口来随机或有序地存储文件中的特定字节。从逻辑上看，UNIX文件中不存在记录和文件的结束字符，也没有各种各样的记录之分。

“简单”是UNIX设计思想的最典型诠释。UNIX文件是一个简单、基本的概念，由此可派生出更复杂、意义更为清晰的结构（像带索引的文件组织），UNIX忽略一些不成熟和特殊的情况，使其尽可能简单。举个例子来说，在我们通常用的文本文件中，一般都需要通过系统工具和用户程序来处理新行字符（就是ASCII码中的换行符），而对UNIX来说，它只是一个字符而已，与其他字符并无两样，除非你希望你的文件是由行组成的，那样才需要处理换行符。

UNIX也不区分各种各样的文件，一个文件可以包含可读字符（比如商品清单和你正在阅读的此段文字），也可以包含二进制数据（比如一个编译后的程序）。我们可以以相同的原语操作和工具集来读取它们。基于这一点，你在UNIX文件中找不到其他操作系统常有的各种文件命名规则（但是像cc这样的程序仍遵循简单的命名规则）。UNIX的文件名几乎是随意的，在SVR4（System V Release 4）中，文件名可以长达255个字符。然而，XSI指出：要真正做到可移植，文件名不能超过14个字符，因为一些早期的UNIX上存在这种限制。

1.1.1　目录和路径

与文件相关的一个重要概念是目录。目录是一些文件的集合，目录的存在使得我们可以对系统中的信息做一个逻辑上的划分。比方说，每个用户一般有自己的home目录，然而命令、系统库文件和管理程序通常位于一些特定的目录下。除了文件之外，目录也可以包含一些子目录，子目录又

可以有子目录，依此类推，形成一个巨大的目录结构。事实上，目录可以有任意想要的深度，这样，UNIX 的文件被组织成一个多层次的树状结构，每个非叶子节点都对应着一个目录。这棵树的顶点只有一个目录，我们通常把它称为根目录。

第4章将详细介绍目录结构，然而，值得注意的是，因为本书通篇都使用UNIX文件这种称法，有必要说明一下，UNIX 文件的全称是文件路径名，它更能反映这个树状结构。路径名指出了到达一个文件先后所经过的所有目录，举个例子说：

```
/usr/keith/book/chap1
```

可以分解如下：第一个“/”表示此路径是从根目录开始的，即路径名给出了文件存储器中文件的绝对位置。接下来是 usr，它是根目录的一个子目录，keith 目录则是 /usr 的子目录，同理，book 是 /usr/keith 的子目录。最后一个元素 chap1 可以说是目录，也可以说是一个常规文件，因为它们采用相同的命名法则。在图 1.1 的例子中，目录树包含了这个路径名。

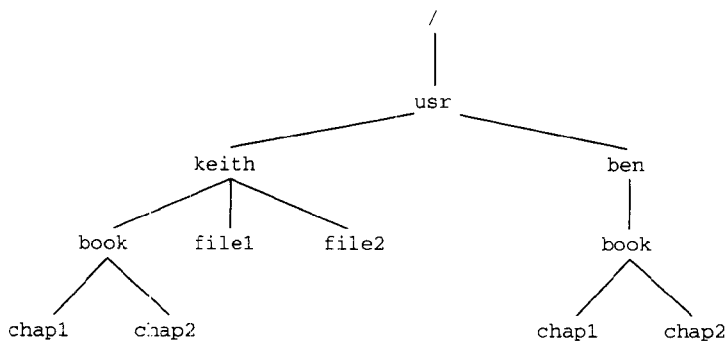

图 1.1 目录树的一个例子

以非“/”开始的路径称为相对路径，它是相对于当前工作的目录来说的。打个比方说，路径名：

```
chap1/intro.txt
```

描述了文件 intro.txt，它包含在当前目录的 chapl 子目录下。在一个极端的例子中，例如：

```
intro.txt
```

这样的名字仅仅是指当前目录下的 intro.txt 文件。再次提醒大家，要真正使你的程序可移植，路径名中每一个单独的元素都不能超过 14 个字符长度。

1.1.2 所有权和权限

UNIX 文件并不仅仅由它所包含的数据来标识，它还有一些其他的原始属性。比如，每个文件都有一个特定的所有者。所有权给予用户一定的权利，其中一个就是可以改变文件的权限。我们将在第3章介绍，权限用以决定哪些用户可以读、写、执行一个文件，当然，要执行一个文件的前提是该文件是可执行的。

1.1.3 文件概念的推广

UNIX 的文件概念不仅包括通常的文件（用 UNIX 系统术语来表述就是正规文件），还包括外

围设备和进程间通信的管道，这就意味着同一个原语操作可用于可读写文件文本、二进制文件、终端、磁带单元甚至是主存。该机制使得程序可以被当做一个普遍的工具，运用于各种各样的设备。例如：

```
$ cat file > /dev/rmt0
```

可以将一个文件写到磁带上（路径名 /dev/rmt0 是磁带驱动器的一个常用记法）。

1.2 进程

UNIX 的术语中，进程仅仅是一个可以执行程序的实例。创建进程的最简单方法是在 UNIX 命令处理器或 shell 中键入一个命令。比如用户可以键入：

```
$ ls
```

接受这个命令的 shell 进程会特意创建另一个进程来执行查看目录内容的程序 ls，因为 UNIX 是一个多任务系统，可以同时有多个进程在 UNIX 中运行。事实上，对于 UNIX 系统用户来说，至少会有一个，通常是多于一个的进程。

1.2.1 进程间的通信

UNIX 允许并发的进程采用进程间通信（IPC）方法来传递信息。管道是其中的一种方法。管道通常用来将一个程序的输出连接到另一个程序的输入上，而不需要任何中间文件。同样，用户可以通过 shell 来使用管道机制。例如下面这个命令行：

```
$ ls | wc -l
```

使得 shell 同时创建两个进程，一个运行 ls，一个运行计数程序 wc，shell 还把 ls 的输出连接到 wc 的输入上。该命令的结果是计算当前目录的文件数。

除管道之外，信号也是 UNIX 的一种进程通信机制。信号是一种基于中断的通信模型，UNIX 中更高级的通信机制包括共享内存和信号量。除此之外，套接字常用来进行网络间进程的通信，也可用做同一台计算机上进程间的简单 IPC 机制。

1.3 系统调用和库函数

在前言中曾经说过，本书的着重点是 UNIX 的系统调用接口。对于一些读者来说，可能还需要进一步了解系统调用。

系统调用实际上可以视为软件开发者到 UNIX 内核的通行证。我们在前言中遇到过内核这个概念，它实际上是一串常驻内存程序，负责调度 UNIX 进程和 I/O 操作。简而言之，内核就是 UNIX 系统中的特权者。所有用户程序和所有文件系统的访问，都通过内核处理、监视和控制。

系统调用与程序员调用一个普通的 C 语言函数或者库函数没什么区别。举一个例子，我们可以用 C 库函数中的 fread 来从一个文件中读入数据：

```
nread = fread (inputbuf, OBJSIZE, numberobjs, fileptr );
```

或者可以用一个比较低级的系统调用 read 来实现读取：

```
nread = read(filedes, inputbuf, BUFSIZE );
```

库函数和系统调用的主要区别是：一个程序调用库函数，库函数本身是程序的一部分，虽然它可能是从程序之外的库中链接进来的；对于一个系统调用来说，它实际上是内核的一部分，而不是程序的一部分。换句话说，调用程序直接使用内核提供的方法。用户进程和内核间的切换通常采用软件中断的方法来实现。

你并不应该感到奇怪，大多数系统调用都是关于文件和进程的。事实上，系统调用的基本原语操作包括文件和进程两部分。

文件方面的操作包括写数据到一个文件或从一个文件获取数据、随机访问一个文件或者改变文件的访问权限。

进程方面的操作包括创建一个新的进程、终止一个已经存在的进程、获得关于进程状态的信息或者建立一两个进程间的通信管道。

还有一小部分系统调用与文件和进程都没有关系，通常来说，这些系统调用都是关于系统信息和控制信息的。例如，一个系统调用允许一个进程查询内核，以获得当前时间和日期；一个系统调用允许一个程序重新设置它们。

除了系统调用接口，UNIX 系统也提供大量的标准库函数。一个非常重要的例子就是 I/O 标准库。I/O 库里包含一些库函数，实现了格式转换和自动输入、输出缓冲；如果直接采用系统调用访问文件，是没有这些机制的。虽然标准I/O库提高了输入输出效率，但它们最终还要依赖系统调用。它们应该看做是构筑在系统调用上的额外一层文件访问机制，而不是单独的子系统。可以这样说，只要一个进程与其所处的系统环境存在交互关系，那么，无论它有多小，都会在某一点上依赖系统调用。

图 1.2 表明了程序代码与库函数之间的关系以及库函数与系统调用之间的关系。它说明了库函数 fread 最终是作为底层系统调用 read 的一个接口来使用的。

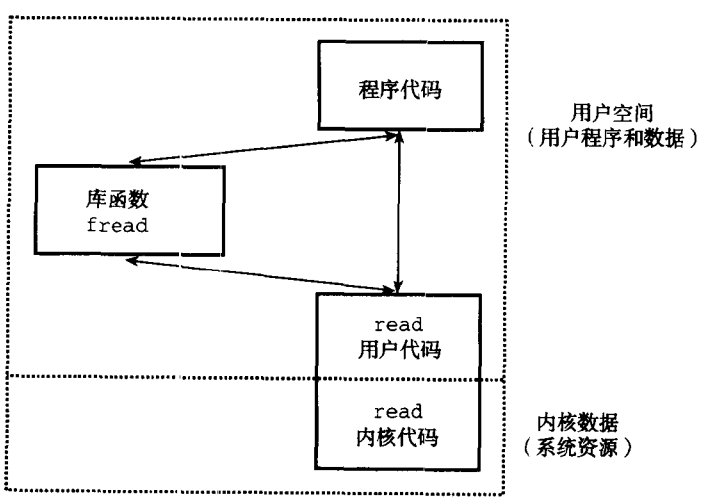

图 1.2 程序代码、系统调用和库函数之间的关系

习题 1.1 解释下面术语的含义：

内核 系统调用 C 库函数 进程 目录 路径名