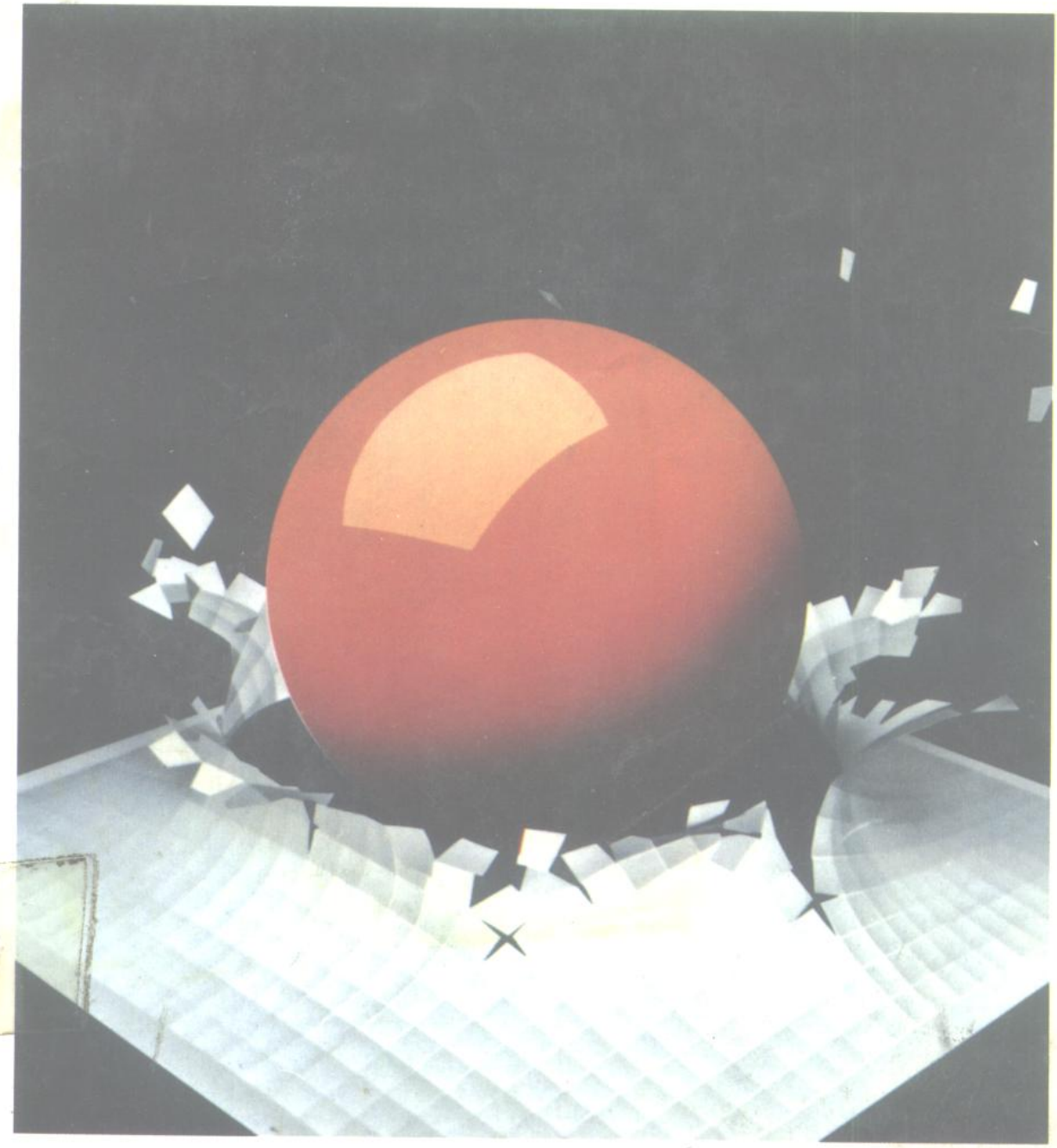
UNIX SYSTEM V

内核剖析

●杨学良　刘杰　张晓东　王威炜　王吉阳　徐晰　编著



電子工業出版社

·研究报告·

UNIX SYSTEM V内核剖析

杨学良 刘 杰 张晓东
王威炜 王吉阳 徐 晰 编著

电 子 工 业 出 版 社
1990.1.

内容提要

本书为中国科大研究生院计算机系及航天部二院 204 所共同合作的有关 UNIX SYSTEM V的研究报告.

全书共分十一章. 主要内容有: UNIX 系统内核导引; 文件系统的数据结构和表示; 文件系统的系统调用; 进程结构和控制; 中断, 陷入和软中断; 内存管理; 进程通信; 字符设备管理; 块设备管理; 系统初启等.

本书适合于高等院校、科研、计算中心等科研开发及 UNIX 广大用户使用.

UNIX SYSTEM V 内核剖析

杨学良 刘 杰 张晓东 编著

王威炜 王吉阳 徐 晖

责任编辑: 王惠民

电子工业出版社出版(北京市万寿路)

电子工业出版社发行, 各地新华书店经售

中国科技情报所印刷厂印刷

*

开本: 787×1092 毫米 1/16 印张: 18.75 字数: 464 千字

1990 年 1 月第 1 版 1990 年 1 月第 1 次印刷

印数: 0～2,000 册 定价: 35.00 元

书号: ISBN7-5053-0674-X/TP·111

(技术资料)

前　言

UNIX 系统是美国 Bell 实验室 Ken Thompson 和 Dennis Ritchie 于 1969–1971 年共同设计和实现的一个通用的，交互式的分时系统．目前在国内外正广泛地应用于教学、科研、工业和商业等多个领域，成为计算机发展史上的佼佼者．其运行范围垮越不同类型的计算机(大、中、小、微甚至巨型机也不例外)，不同厂商的产品，在全世界大约有二百万台计算机运行 UNIX 操作系统，应用之广在计算机发展史上也是罕见的．一九八八年世界信息产业十大要闻(IDC 公布)"UNIX 风云遍全球"被名列榜首．几乎全世界所有主要的计算机公司、研究单位、和大学都卷入 UNIX 的开发和应用之中．UNIX 已成为工业标准这是不言而喻的．

美国 ACM 协会为 UNIX 系统设计师颁发图灵奖(Turning)时指出"UNIX 操作系统的成功在于它对少数关键思想作恰如其分的选择并加以精美的实现．"众所周知，UNIX 以结构紧凑，功能强，效率高，使用方便和可移植性好等优点而著称．因此把 UNIX 系统移植到各类计算机上比研制新的操作系统更容易、更快、更好、更省时，可取得更高的性能价格比．

目前 UNIX 系统已引起国内人们普遍的关注，很多计算机的研制和开发工作，大学教育和培训工作都是基于 UNIX 操作系统．越来越多的科研人员、大学生、研究生不仅需要学习和使用 UNIX，而且迫切要求深入了解 UNIX 系统的内部构造．由于科研工作的需要，我们在学习和使用的基础上，对 UNIX System V 的核心程序的功能、设计、算法及重要的数据结构和主要程序进行了深入的剖析，画出了程序流程图，写出了剖析报告．在此基础上，加工，整理，出版，奉献给读者．

全书共分十一章，由杨学良（第一、二章)，张晓东（第三、四章)，刘杰（第五、六和十章)，王吉阳（第七章)，王威炜（第八章)，张晓东和刘新焕（第九章)，徐晰（第十章)．共同完成，最后由杨学良修改定稿．

该项剖析工作是在航空航天部二院 204 所三室的支持下完成的，俞忠东同志给予了很多帮助．在剖析过程中曾参考了国内一些同志对 UNIX 操作系统早期版本的分析，在此一并致以谢忱．

由于我们水平有限，时间非常匆忙，文中不当和错误之处，敬请读者批评、指正．

杨学良　89 年春于北京

中国科大研究生院

目　录

1 对系统的看法

UNIX 系统是美国 Bell 实验室 Ken Thompson 和 Dennis Ritchie 于 1969～1971 年设计和实现的。1974 年 7 月在美国 ACM 通讯杂志上正式发表公布于世。虽然 UNIX 原来是为程序开发而研制的，但实践业已证明，对于软件应用来说，它同样是一个理想的、通用的交互式的分时系统。UNIX 系统开始是在小型计算机上取得成功，后来向微型机和大、中型机系统甚至巨型机系统领域渗透。到目前为止，跨过不同的厂商和产品，全世界已有二百多万台计算机运行 UNIX 系统，这在计算机发展史上也是罕见的。它不仅广泛的用于科学研究、高等院校和政府机关，而且在企业界也获得了重要地位，正逐步形成为工业标准。

随着 UNIX 系统应用的迅猛拓广，曾进行了多次修改，派生出很多版本。但它的根本思想没有发生变化。当今最流行的有两个版本，其一是美国 AT & T 制定的 UNIX SYSTEM V，正在研制的是 System V 4.0 版。其二是美国加州大学伯克力分校研制的 UNIX 4.2 BSD，更新的版本是 UNIX 4.3 BSD。本书集中剖析和描述的是 UNIX SYSTEM V 的内核。

由于科研工作的需要，在学习和使用 UNIX 系统的基础上我们剖析了 AT & T Bell 实验室的 UNIX SYSTEM V 核心源程序(也称 UNIX 操作系统或内核)。在理解的基础上，阐明其工作原理，探讨其设计思想和具体的实现。希望我们的工作对想了解 UNIX 操作系统内部构造和从事改造和开发应用的读者有些帮助。除此之外，我们对源程序也作了一些注释以供参考。本剖析主要描述了 UNIX 操作系统的数据结构和算法。对内核程序进行了仔细的剖析，分别给出有关程序块的功能，输入和输出参数，进一步还阐明了具体实现过程，并给出调用关系和框图。最后，提供给用户的是一个标准用户接口。我们深信该"剖析"对 UNIX 开发和拓广应用会是有益的。本书第一章是对整个系统总的看法。首先回顾 UNIX 系统发展的历史，然后给出 UNIX 系统的总体结构和主要特点。最后给出关于硬件的设想。第二章是 UNIX 内核的导引。首先给出 UNIX 内核的结构，然后讨论主要的基本概念，如文件系统、进程结构和控制等，最后是数据结构和系统管理。其目的是为学习以下各章打下基础。第三、四两章我们首先阐述文件系统，它是 UNIX 成功的关键。给出文件系统的数据结构和内部算法，i 节点管理，打开文件表的管理。然后详细的分析了文件系统的系统调用等。第五、六两章集中讨论进程结构和控制，然后剖析了中断、陷入和软中断及进程调度等。第七章是内存管理。第八章是 UNIX SYSTEM V 关于进程通信的描述和实现。第九、十两章讨论字符设备和块设备的管理。第十一章叙述了系统的初启过程。

1.1 发展历史

UNIX 操作系统是美国 Bell 实验室 Ken Thompson 和 Dennis Ritchie 共同设计和实现的一个通用的，交互式的分时系统。在设计时除了吸取以往操作系统设计和实践中的各种成功的经验和失败的教训外，特别是从 Bell lab. 和 MIT 联合研制的 CTSS 和 MULTICS 系统中吸取了极为丰富的养料。最早的实现是在 PDP7 小型计算机上完成，

后来到 1971 年正式的移植到 PDP11 计算机上，在 Bell 实验室内部流传．1973 年 D.Ritchie 研制成 C 语言并把他原来用汇编语言写的 UNIX 源程序用 C 语言改写完毕，这就是最早的正式文本叫 UNIX 第五版．同年他们俩人合写的论文"UNIX 分时系统"正式发表公布于众．由于它的简洁、有效和可移植性好开始在大学和科研单位广泛流传取得意外的成功．主要配备在 PDP11 小型计算机上．到 1978 年发表了对后来有重大影响的 UNIX 第七版，主要在 PDP11／70 和 Interdata 8／32 计算机上运行．这就是当代 UNIX 操作系统的前辈．不久它被移植到当今还流行的 DEC 公司的 VAX 系列计算机上．在 VAX 机上也叫 V32．以后 AT & T Bell 实验室成立一个专门组织 USG(UNIX Support Group)来管理、控制和研究 UNIX．进一步改进，1981 年对外提供 UNIX SYSTEM Ⅲ．到 1982 年 SYSTEM Ⅲ又吸收了第七版，V32 和另外的版本的特点，发表了实时 UNIX(UNIX／RT)．1983 年 Bell lab.的 USG 宣布了当今最流行的版本 UNIX SYSTEM Ⅴ．UNIX 开始为工业界承认，成了许多厂商和行业共同采用和推广应用的对象，这个趋势至今还经久不衰的发展．本书所剖析的正是 Bell 实验室 1984 年正式公布的 UNIX SYSTEM Ⅴ的内核，它是国内外比较流行的版本．到 1986 年大约有近二百万台计算机上安装了 UNIX 系统，这在计算机发展上也是罕见的．另外值得一提的是美国加州大学 Berkeley 分校 Bill Joy 和 Ozalp Babaoglu 开发的 3BSD 后来发展成 UNIX 4.2 BSD．87 年更新的版本叫 UNIX 4.3 BSD．在美国大学中也有不少应用．

图 1-1 给出 UNIX 系统发展历史，对各种版本的关系也给予说明．

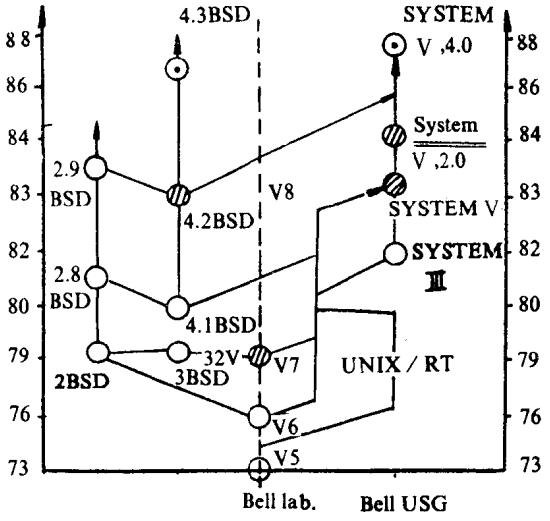


图 1-1　UNIX 系统发展史

1.2　设计思想

UNIX 系统取得如此成功并为国内外的学者和用户倍加推崇的原因，除了外界因素即时代的需要之外，起决定作用的是其内在因素即它本身的特点和性能．我们首先探讨一下 UNIX 系统的设计思想也许会受到有益的启发．

美国著名的计算机专家 K. Thompson 和 D. Ritchie 在设计 UNIX 系统时虽然没有过多的标新立异，但对前人的工作做了有选择地继承，除分析了以往操作系统成败的经验教训外，在总体设计思想上有所发展．他们俩人都参加过 Multics 的设计，而开发该系统所使用的工具是 CTSS (Compatible Time-Sharing System)．他们在设计 UNIX 系统时从上述两个系统中吸取了不少成功的经验．例如文件系统，用户界面等．过去人们设计操作系统总是希望包罗万象，使系统非常庞大和复杂．人们为理解、修改和维护操作系统大伤脑筋，常常感到无能为力．而 UNIX 系统的设计师则对系统的功能进行了仔细斟酌，着眼于向用户提供开发工具，为应用建立开发环境．希望系统尽量小，便于学习和使用．大多数算法选择原则是尽量简单，而不是在速度和复杂性方面．系统在实现中并不把很多问题都加以阐述，而是把灵活性留给用户，后来成了发展推广应用 UNIX 系统的关键因素．在上述设计思想指导下，UNIX 系统的结构设计采用了内核和实用程序分开的方案．内核(Kernel)一般也称 UNIX 操作系统，它包括文件系统、进程结构和控制、存储管理和设备管理等，对内核的功能进行了仔细考虑，对内核采用的主要算法经过反复推敲，对其中采用的数据结构和程序进行了精心设计．使其简洁精干、确保了系统的高效运行．凡是能从内核分离出来的部分都以实用程序在用户环境运行，内核对实用程序提供强有力的支持，实用程序则都以内核为基础，通过系统调用把两者结合成一个有机的整体，为用户提供优良的友善的服务．

在 UNIX 系统中，采用树形结构的文件系统，把文件和设备统一作为文件处理(文件是无结构的字符序列)．用户可按需任意组织文件．文件既可顺序读写，也可随机存取．设备和文件都统一作为文件处理，它们在用户面前有相同语法语义，使用相同的保护机制．这样不但简化了系统设计又方便了用户的使用．

UNIX 系统采用灵活的命令程序设计语言 shell，它既是与终端用户交互式发生作用的命令语言，又是在命令文件执行时的程序设计语言．用户通过 shell 语言可以方便的使用 UNIX 系统中各种程序设计的工具．

UNIX 系统内核和外层的实用程序，绝大部分都用 C 语言书写，使系统易于理解、修改和扩充，特别是可移植性好．这一特点为 UNIX 移植到不同的厂商生产的不同的机型、及广泛的应用到各行各业提供了前提，奠定了基础．

1.3 系统结构

图 1-2 给出 UNIX 系统的结构示意图．图中最核心是计算机硬件，它提供对 UNIX 软件的支持．靠近硬件的内层是 UNIX 内核程序也称操作系统．内核直接和硬件打交道是程序和硬件之间的接口或界面．它对一切外层程序提供公共服务，通过内核把一切外部程序和硬件隔离起来．也就是说，程序的执行不再依赖具体的硬设备．因而为程序在不同机器之间提供很好的可移植性．内核程序是 UNIX 系统中唯一不能由用户任意变化的部分．它大致分成文件系统管理、进程管理、内存管理等几部分．进程管理又分为低级进程管理和高级进程管理．低级进程管理主要包括：进程调度分配、控制占用处理机的程序和基本的进程通信．高级进程管理主要包括：进程的创建、终止、进程间通信、进程在内存和外存之间转储、信号机构和进程间跟踪控制等．内核内部的层次结构并不很清晰．内核程序的外层是实用程序，内核提供对实用程序的支持，两层之间的界面是系统调用．

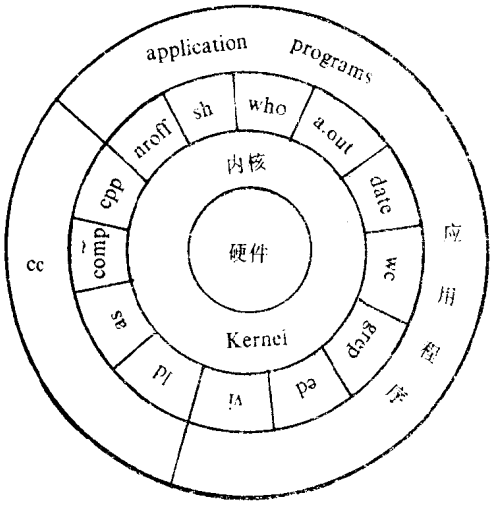


图 1-2 UNIX 系统结构

内核外的实用程序通过系统调用来和内核打交道．实现的过程是核外程序通过一种特殊指令(陷入指令)进入内核，通过陷入处理程序再转入相应的系统调用处理程序．

实用程序例如 shell 和编辑程序 ed 和 vi 它们和内核互相作用都是通过系统调用．系统调用通知内核为调用程序作各种操作和为调用程序和内核之间交换数据(在这一层中还有若干命令)．但是，个别用户程序也可放在这一层，例如(a.out)这个标准名字靠 C 编译来生成可执行文件．另外，各种应用程序都放在图 1-2 的最外层．例如 C 编译程序(cc)．图中应用程序分成两层(实际上用户可以设计成若干层)．UNIX 系统的特点之一就是能巧妙的把上述现存的各种程序组合起来，为了完成某项任务而设计出更复杂的大型程序．

很多应用子系统(程序)和实用程序例如 shell,ed, vi 和 sccs(源码控制系统)一起逐渐形成了 UNIX 系统的同义词．所有的应用低层都通过内核，利用系统调用来服务．我们剖析的这个版本 UNIX SYSTEM V 中有 56 种系统调用(其中常用的不超过 32 种)．关于这些系统调用的剖析将在第四章及其他有关章节中给出更详细的描述．最后，我们再强调一下本书描述的只是 UNIX SYSTEM V 的内核，关于 UNIX 系统的其他部分请参阅有关文献资料．

1.4 主要特点

美国 ACM 协会于 1983 年给 UNIX 系统设计师 K. Thompson 和 D. Ritchie 颁发图灵奖(Turning)时指出："UNIX 操作系统的成功在于它对少数关键思想作恰如其分的选择并加以精美的实现．"

UNIX 系统在计算机发展史上取得如此的成功，安装的系统和用户还在不断扩大，应用范围继续增长．随着应用的深化，系统更加日趋完善．我们认为任何事物的发展决不是偶然的，UNIX 也不例外．应用是 UNIX 发展的出发点和旧宿，应用是推动事物发展的强大动力．下面我们简要的讨论一下 UNIX 系统的主要特点：

(1) 使用方便的界面

UNIX 提供两种界面(或接口)．其一是用户界面，用户通过终端打入相应命令和系统进行交互式作用．UNIX 系统在用户界面提供了一种命令程序设计语言 shell，它不但具有命令语言的功能(Command Language)，也具有应用程序设计语言的特点，功能强、灵活，使用它能提高系统运行效率．UNIX 系统提供的另一个界面是程序界面，也称系统调用(SYSTEM Call)，它提供用户编制程序时和内核相互作用的接口．一般操作系统是在汇编语言级提供给用户，而 UNIX 操作系统是在 C 语言级提供这些功能．用户程序和内核之间的互相作用可以看成是 C 语言的部分应用，使系统更灵活，使用更方便．UNIX

系统提供这两种界面，使功能更齐全、易于修改和扩充，用户使用非常方便。

(2) 树形结构文件系统

UNIX 系统中的文件系统是树形层次结构，系统中的文件是无结构的字符流式文件。整个文件系统分成基本文件系统和可装卸的文件卷两部分。它既能扩大文件的存储空间，又有利于安全和保密。在 UNIX 系统中文件系统的另一个特点是把文件和设备统一处理，为用户提供了一个简单而又统一的接口。文件、目录和外部设备都作为文件统一处理，文件无结构、无类型的概念，所有文件均作为无格式的字符流序列。用户可任意组织格式，对文件既可进行顺序读写也可随机存取。这样的文件系统简化了系统设计又方便了用户。人们都认为 UNIX 文件系统的设计是很成功的。

(3) 可移植性好

UNIX 操作系统和核外实用程序基本上是用 C 语言书写。使它易于理解、修改和扩充。和其他操作系统相比它的可移植性特别好。移植性好有两层含义：其一是 UNIX 系统的开发与硬件无关，从单用户的微型计算机到多用户的大型机甚至巨型计算机，象 CRAY-2 和 ETA-10 都能运行 UNIX 系统。其二是在 UNIX 系统下开发的各种应用软件可以比较容易的移植到运行 UNIX 系统的其他计算机上去。因此，用户购卖和使用 UNIX 系统最明显的好处是，随着时间的推移，用户可以选择更适合的硬件配置来对系统进行升级，而且无需考虑所采用硬件的制造商，因此也不必担心系统升级会受制于任何制造厂商。

(4) 提供丰富的实用程序、语言和开发工具

UNIX 系统中提供了大量的实用程序，用户可通过 shell 命令使用它们，对用户编程给予强有力的支持。很多问题的求解可以不必按传统方法编程，而仅需要把现有的资源组合起来即可。UNIX 系统许多命令看起来简单而普通，当组织在一起就表现出强有力的功能和广泛的用途。

UNIX 提供十几种常用程序设计语言，如 C、FORTRAN 77、Pascal、SNOBAL、BASIC 语言等等。还提供开发工具，如 YACC (Yet Another Compiler-Compiler)、LEX (Generator of Lexical Analyzers)等。所有这些都放入文件系统，通过 shell 命令调用。正是由于 UNIX 系统为用户提供大量系统软件，为开发和应用创建了一个相当完备的开发环境。

1.5 用户评述

这一节主要从用户的观点评述 UNIX 系统高层的特点例如文件系统、处理环境和组建块原语(管道 pipes)等，以后的章节将进一步讨论内核对它们的支持。

1.5.1 文件系统综述

用户看 UNIX 文件系统有如下特点:

- 层次结构，
- 文件、数据采用一致的处理，
- 能创建和删除文件，
- 文件动态增长，
- 文件数据保护，

·外部设备(例如终端、磁盘、磁带)象文件一样的处理。

UNIX 的文件系统组织成象是一个具有根节点(ROOT)的树，其根节点写成"／"，每一个文件系统无叶节点的结构是文件的目录，文件在树的叶节点上可以是普遍文件、目录或是特殊设备文件。一个文件的名字靠路径名给出，路径名描述如何判定一个文件系统的层次结构中的位置。路径名是一系列靠"／"分开的元素名字。一个元素名是一个顺序字符串，它规定要求任一个文件名必须在前面的目录中是唯一的。一个用斜线符号"／"为首的全路径名，规定了一个文件，沿着路径即可找到一个确定的文件。例如：路径名象是"／etc／passwd,""／bin／who,"和"／usr／src／cmd／who.c"等等如图 1-3 所示。

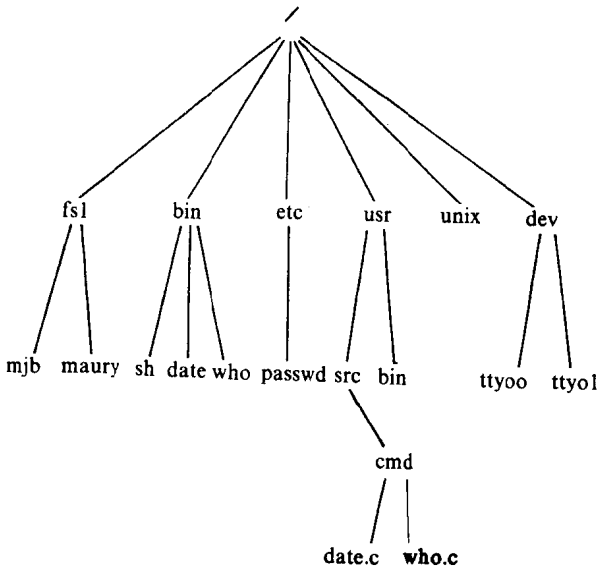


图 1-3　树形结构文件系统

为了进一步理解文件系统，下面给出一个 C 语言的程序如图 1-4 所示。目的是对一个现存的文件建立一个副本(copy)，假定这个可执行的文件是叫 copy。用户在终端上打入如下的命令即可

copy oldfile newfile

文件系统的讨论可以有两种观点：其一是从用户的观点，目的是研究用户思维中的抽象文件，它的侧重点在于为用户提供一个逻辑结构清晰，使用简便的逻辑文件形式。用户将按照这种形式去存储、检索和加工有关文件的信息。

另一种是从实现的角度来研究，其目的是研究驻留在存储介质上的实际文件即物理文件，它是侧重于选择一些工作性能良好、设备利用率高的物理文件结构，系统将按这种形式和文件存储设备发生联系并控制信息的传输。文件系统的重要作用之一就是在用户逻辑文件和相应存储设备上的物理文件之间建立映射关系。关于两者之间的具体映射关系以后的章节将给出更详细的讨论。

```
# include <fcntl.h>
char buffer[2048];
int version = 1;          / * Chapter 2 explains this * /
main(argc,argv)
    int argc;
char * argv [ ];
{
    int fdold,fdnew;
    if (argc! = 3)
    {
        printf("need 2 arguments for copy progam\n");
        exit(1);
    }
    fdold = open(argv[1],O__RDONLY);   / * open source file read only * /
    if (fdold = = -1)
    {
        printf("cannot open file %s\n",argv[1]);
        exit(1);
    }
    fdnew = creat(argv[2],0666);      / * create target file rw for all * /
    if (fdnew = = -1)
    {
        printf("cannot create file %s\n",argv[2]);
        exit(1);
    }
    copy (fdold,fdnew);
    exit(0);
}
copy (old,new)
    int old,new;
{
    int count;
    while((count = read(old,buffer,sizeof(buffer))) > 0)
        write(new,buffer,count);
}
```

图 1-4　拷贝文件程序

1.5.2　处理环境

一个程序是一个可执行的文件，一个进程是程序的执行．在 UNIX 系统中很多进程可以同时执行(也称多道程序)．在逻辑上没有数量限制．一个程序(例如 copy)的多个进程可在系统中同时存在．各种系统调用允许进程去创建新的进程，终止活动的进程，同步进程执行状态，控制对发生的事件做出反应．在用系统调用的条件下，各进程是互相独立的执行．

如图 1-5 所示，创建一个新进程拷贝文件程序．它执行 fork 系统调用创建一个新进程叫 child（子进程），从 fork 得到返回值 0 后调用 execl 执行 copy 程序，则这个 execl

用文件"copy"覆盖 的子进程的地址空间假定是在当前目录下运行用户提供参数的程序，如果这个 execl 调用成功，它决不返回，因为这进程在一个新的地址空间执行，正象将在进程控制中看到的那样．在这同时，请求 fork 的进程(即父进程)接收到一个非零的返回值后，调用 wait 暂停它的执行直到该 copy

```
main(argc,argv)
    int argc;
    char * argv[ ];
{
    / * assume 2 args: source file and target file * /
    if (fork ( ) = = 0)
      execl("copy","copy",argv[1],argv[2],0);
    wait((int * )0);
    printf("copy done\n");
}
```

图 1-5　创建一个新进程拷贝文件程序

完成后，打印出消息"copy done"并执行 exit 进程结束．例如，如果这个可执行程序的名字是 run，用户请求调用这个程序靠打入如下命令:

run oldfile newfile

进程拷贝"oldfile"到"newfile"并打印出结果．为了进程控制，用了四种调用:

- fork
- exec
- wait
- exit(使用要谨慎)

很多操作系统中的内核功能，在 UNIX 系统中被放入实用程序，通过 shell 执行命令解释程序来实现．shell 是一个用户程序而不是内核的一部分．很容易修改和剪裁它成为一个专门的环境．shell 使用户具有同时执行多个进程的能力，进程能动态的创建新的进程，并能同步进程的执行．总之，用户感到 UNIX 系统为用户提供了一个强有力的执行环境．

1.5.3　组建块原语

正如上述，UNIX 系统还提供了若干操作系统原语，使用户只写很小、模块化的程序就能利用组建块原语组建复杂的大程序．组建块原语有:

(1) 转向输入、输出原语

进程一般对三类文件进行存取

- 从标准输入文件读，
- 写到它的标准输出文件，
- 写错误消息到它的标准错误文件．

为了便于用户和有关进程进行交互式作用，通常与该进程相关的终端机输入、输出部