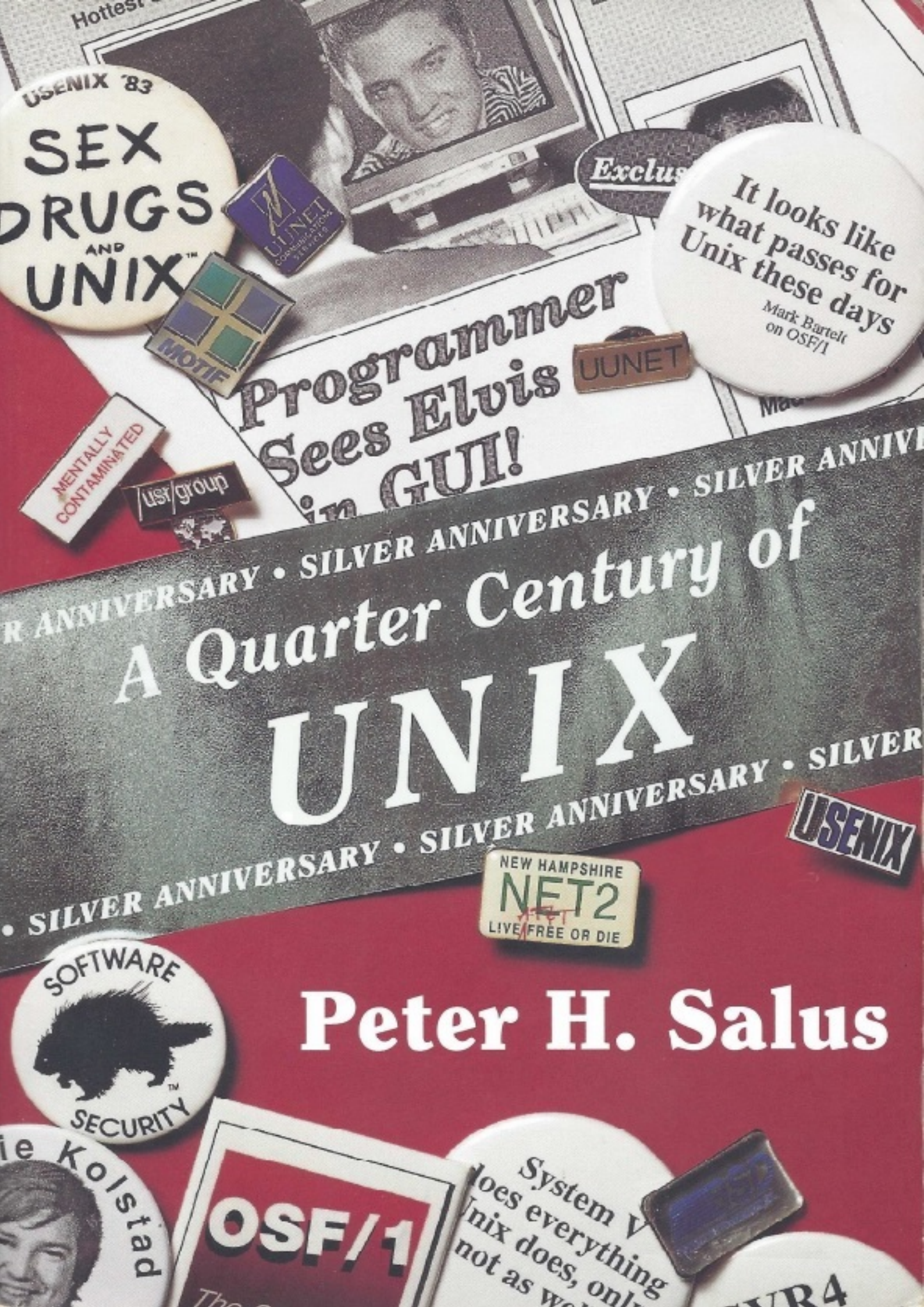

USENIX '83
SEX DRUGS AND UNIX
UUNET
MOTIF
Programmer Sees Elvis in GUI!
UUNET
It looks like what passes for Unix these days
Mark Bartelt on OSF/1
MENTALLY CONTAMINATED
/usr/group
SILVER ANNIVERSARY • SILVER ANNIVERSARY • SILVER ANNIVERSARY
A Quarter Century of
UNIX
SILVER ANNIVERSARY • SILVER ANNIVERSARY • SILVER
USENIX
NEW HAMPSHIRE
NET2
LIVE FREE OR DIE
SOFTWARE SECURITY
Peter H. Salus
OSF/1
System V does everything Unix does, only not as well

目錄

封面 1.1
前言 1.2
感谢 1.3
简介 1.4
起源 1.5
- 太空序曲 1.5.1
- 1969年夏天到1970年秋天 1.5.2
- 机械计算和电子计算 1.5.3
- 操作系统 1.5.4
- MAC项目：CTSS和Multics 1.5.5

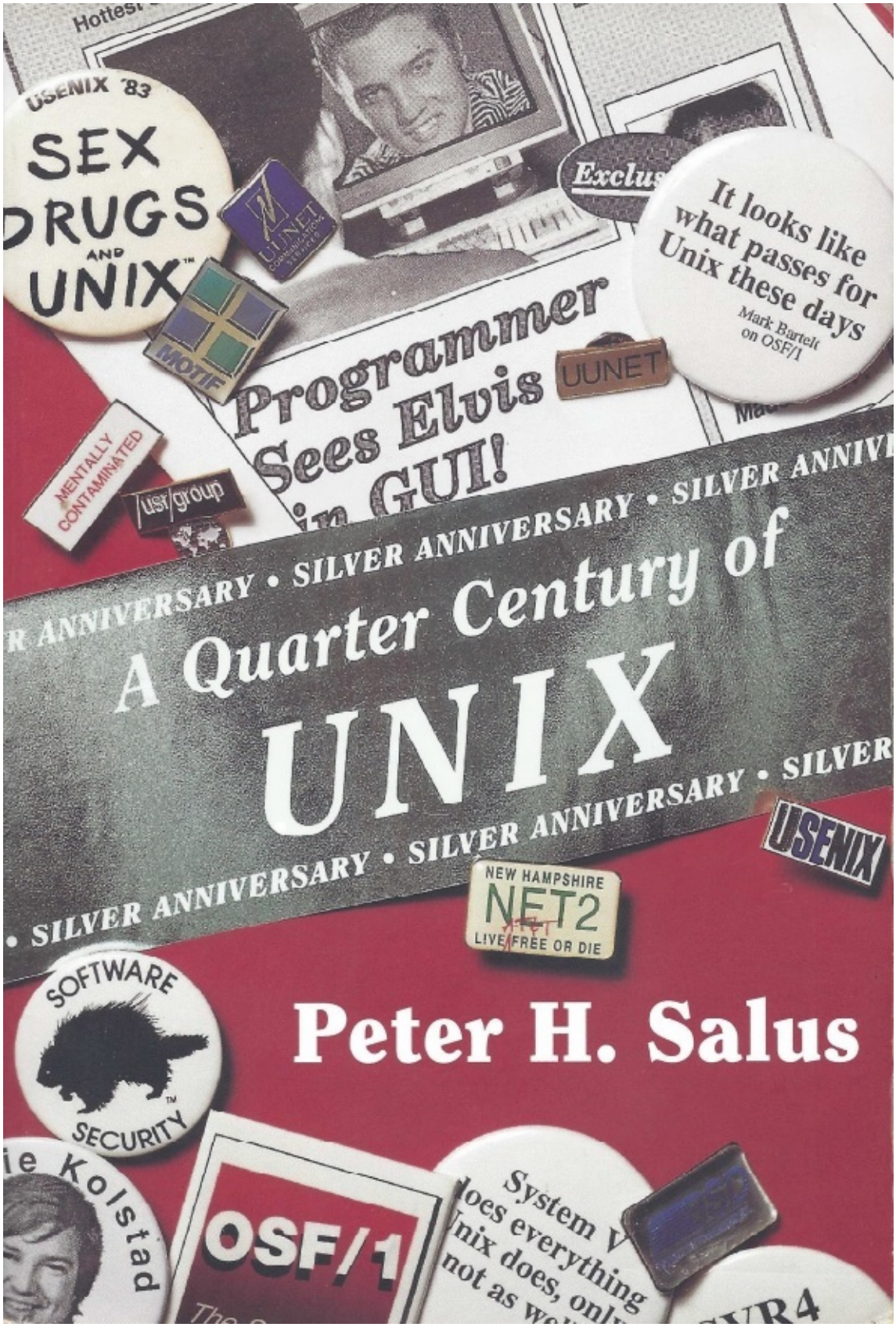
一个系统的诞生 1.6
- PDP-11 1.6.1
- 1971年，第一版 1.6.2
- 1971-1973，C语言和管道 1.6.3
- 1973，第一篇论文 1.6.4
- 法律 第一部分 1.6.5

什么造就了Unix? 1.7
- 用户 1.7.1
- 为什么是Unix 1.7.2
- 风格和工具 1.7.3
- PWB和MERT 1.7.4
- 实用工具 1.7.5

Unix的传播和兴盛 1.8
- 用户 第二部分 1.8.1
- 伯克利Unix 第一部分 1.8.2
- V7 1.8.3
- 伯克利Unix 第二部分 1.8.4
- 商业Unix 1.8.5
- DEC 1.8.6
- 法律 第二部分 1.8.7

Unix产业 1.9
\/usr\/group 1.9.1
Sun和JAWS 1.9.2
标准 1.9.3
现状 1.10
Unix间的竞争 1.10.1
衍生版本 1.10.2
OSF和UI 1.10.3
VAX后的伯克利Unix 1.10.4
法律 第三部分 1.10.5
末章 1.11
它为什么成功 1.11.1
附图 1.11.2

UNIX二十五年 -- A Quarter Century of UNIX 翻译



UNIX是一个简洁，优雅，可移植，以及能力非常强大的软件系统，它没有借助于商业市场就变得非常流行。程序员们使用它，大的商业公司们却在抵制它。十年后，用户们取得了令人难以置信的胜利。 `A Quarter Century of UNIX` 是第一本由亲历者讲述这段历史的书。开发者

和用户们描绘了技术社区是如何战胜了数百万美元的商业公司。准备写这本书时，Peter Salus访问了一百多位关键人物，并为了搜集相关信息从澳大利亚走到了奥地利。本书搜集流传各地的UNIX故事汇编成为详尽的UNIX历史。

特点

> - 提供了文档化的UNIX系统开发历史
> - 包含一百多位UNIX社区关键人物的访谈
> - 收录了珍贵的照片和图片
> - 详细讲解为什么UNIX可以取得成功

关于作者

> Peter H.Salus是一位获得广泛认可的UNIX爱好者，是季刊《计算机系统》的主编。他定期为USENIX的杂志《;login:》撰写专栏《书虫》。他也是一系列图书，文章以及访谈的作者。Salus拥有化学专业学位，德语学位，以及纽约大学的一个语言学博士学位。

译者注

- 本书由Addison-Wesley出版社于1994年出版
- 在线阅读：http://unix25.zhangyangjing.com
- 翻译源码：https://github.com/zhangyangjing/A-Quarter-Century-of-UNIX-zh
- 英文原版：http://wiki.tuhs.org/lib/exe/fetch.php?media=publications:qcu.pdf
- Unix源码：http://minnie.tuhs.org/cgi-bin/utree.pl

前言

我看到的每一本关于操作系统或者UNIX的书都会用两三页讲述系统的历史，1992年我多了一大堆这样的书，没有一本可以把历史讲的完全准确，有些甚至错的可笑。1993年一月在圣地亚哥举办的USENIX协会大会上，Greg Rose有一个大约一小时的关于历史的精彩演讲，但是这个演讲也不是特别完整。在和Greg，John Quarterman还有其他几个人交流后，我决定走访历史中的人物，着手写一个准确完善的历史。

我不能完全表达对Dennis Ritchie，Doug McIlroy，Kirk McKusick，Lou Katz，Peter Collinson的感谢。John Lions耐心详细的回答了我无数刁钻愚蠢的问题。此外我还要感谢Jaap Akkerhuis，Eric Allman，Keith Bostic，Lorinda Cherry，Clem Cole，Mike Cole，George Coulouris，Jim Curry，Marc Donner，Torn Duff，Robert Elz，Stu Feldman，Mel Ferentz，Torn Ferrin，Dan Forsyth，Michael Gschwind，Teus Hagen，Brian Harvey，Paula Hawthorn，Peter Honeyman，Andrew Hurne，Haruhisa Ishida，Steve Johnson，Bill Joy，Mike Karels，Stan Kelly-Bootle，Brian Kernighan，Kouichi Kishida，Dan Klein，Sam Leffler，Mike Lesk，Chris Maltby，Paul Manno，Jim McKie，Mike Muuss，Mike O'Dell，Rob Pike，Erik van der Poe!，Brian Redman，Charlie Roberts，Debbie Scherrer，Bob Schulman，Gene Spafford，Henry Spencer，Armando Stettner，Heidi Stettner，Toru Takahasi，Ken Thompson，David Tilbrook，Chris Torek，and Mike Ube！

我还要特别感谢Mike Mahoney和Peter Collinson，他们允许我引用他们和几个参与者的谈话内容。感谢Dan Appelman帮我评审“法律”章节。感谢Stuart MacRobert，Mike O'Dell和Len Tower Jr.在写我作过程中阅读手稿并提出意见。还要感谢很多匿名的审阅者和Addision-Wesley的Tom Stone和Kathleen Billus。我还有太多太多名字要列在这里，这本书献给所有的人。

最后，我要感谢Mary W. Salus和Emily W. Salus，他们是我最严厉的批评家(这是最贵重的)。这些工作献给Rosa Lustig Kubin，他在50年前带我走进科学。

P.H.S. 波士顿

感谢

标记为“高数Mahoney”的内容来自AT&T贝尔实验室，“口述UNIX历史项目：起始”由Michael S. Mahoney完成(1989年UNIX博览会上打印分发)。和Peter Collinson的面谈非常仓促。感谢Mahoney教授和Collinson博士允许我使用它们的工作成果。

Doug McIlroy告诉我他认为使用大写的UNIX而不用Unix是一个严重的错误。我接受了他的建议，在大多数地方使用Unix。

简介

UNIX操作系统的成长故事是计算机发展的一个主要部分。Unix证明了在个人有能力支付的计算机上运行的小系统也可以运行在其他硬件平台上，它确实是一个机器无关的可移植系统。Sunil Das记录道，“技术上说，Unix是一个简单，一致的的系统，它用一些好的点子突破了限制”。

我认为Unix最大的特点，它是开发方法的结果。相比于公司生产一个硬件产品拿去售卖，Unix的诞生源于部分人对于打造一个简单的，可以支持多人使用的系统的渴望，这些人因此可以有一个舒适的编码环境。并逐步证明这样做的可行性。

结果是现在市场上售卖的所有操作系统都收到了Unix的影响。

Unix的开发和成长是一个激动人心的社区故事。社区显示关系密切的几个人，再是贝尔实验室，然后是电话社区，继而是大学和科研社区。所有这些都是发生在西部电子和AT&T的抵制分发下发生的。抵制越厉害，人们就越互相分享。由于系统不得不同时满足科研群体和个人用户的需求，它从一开就是为这两类人设计。

某种意义上讲，Unix的历史就是三个群体的历史：贝尔实验室的研发部门，伯克利大学的计算机系统研究小组，还有AT&T的UNIX系统小组(还有他的后期分支)。从另一个观点看也可以说有5个群体：贝尔实验室，计算机研究小组，UNIX系统小组，普通用户和外部世界。

可能对UNIX发展最重要的贡献来自于不断增长的计算机网络，几种通讯协议的开发和世界互联网。

自从1970年开始，Unix深刻的影响着其他每一个操作系统：DOS，Apple OS，Windows NT等等，都从Unix借鉴了大量的东西(就像Unix之于CTSS和Multics)。

接下来讲的不是编程的历史，他是一个社会故事。社会里的独立个体和他们之间的相互交互造就了Unix。

太空序曲

> 陷入到火星和环火星小行星带中间的地区，ken打孔一个复杂的策略到他的导航电脑，看了一眼前面的临时屏幕。停止，前进，左转...。几秒钟后，他感觉到几乎无法察觉的活动。
>
> 一段太空领航员的经历，ken知道了奇特的无重力策略。他明白了应该怎样利用重力和加速度围绕一个行星，月亮甚至是一个小行星的曲线轨道飞行。
>
> 该死的，ken咕哝了一句。该死的授权设备使用的人们。该死的所有管理人员。该死的柜员。继而他意识到以这台机器，这上面的软件，自己根本不可能玩星际旅行。他不得不自己来解决这些问题。

时间是1969年夏天刚开始的时候。炎热的天气像一张旧毛毯盖住了整个新泽西，所有生命都快要窒息。

Ken Thompson结束了在伯克利的加利福尼亚大学的学习，在1966年到新泽西默里山成为贝尔电话实验室的一名研究员。他在Multics的一个项目组工作，这是一个由通用电器，贝尔实验室，以及麻省理工学院联合组建的项目，目的是为大型机开发一个可以支持上千用户同时在线的操作系统（Multics是Multiplexed Infomation and Computing Service的缩写）。同时在贝尔实验室为Multics项目工作的还有Dennis Ritchie（之前在Multics项目实习，1968年于哈佛获得应用数学博士学位后加入贝尔实验室），Joseph F. Ossanna，Jr.，Stu Feldman，Doug McIlroy，以及Bob Morris。但是贝尔实验室不久就退出了这个耗费数百万美金却没有得到任何结果的项目。Multics甚至不能有效的支持3个用户同时在线。自此以后贝尔实验室的计算机研究就变得低迷起来。

大多数的研究小组都是在做些肤浅的工作，Ken却在一些朋友的帮助下，即将改变计算机行业的未来。

1969年夏天到1970年秋天

在1987年制作的纪念光盘上，印着Ken Thompson和Dennis Ritchie长满胡须的头像。光盘的标题是“我们的创建者”。一般来说这是对的，却不够精确。Thompson是创始人，Ritche和Rudd Canaday紧随其后，继而是一些贝尔实验室的工作人员。他们是一个关系密切的朋友圈子。由于流传的故事变得无关紧要，这不是戏剧的关键点。

当我问Dennis Ritchie为什么Ken为星际旅行工作，他回答说仅仅因为这很有意思（实际上，星际旅行是一个严格的天文模拟程序，而不仅仅是一个简单的游戏）。

当然在某种程度上，这是在贝尔实验室普遍的工作态度。实验室的基本概念是：

> 曾经是，现在也是雇佣一些有点子的家伙，并让他们实现出来。Ken做了有趣的事情，并让他变得有价值。当一个好的大学雇佣了一个年轻的教授，会期望他们去做什么？好吧，这当然会有很多的原因，但必定是因为他们有一些好点子并且造成一些影响。在学术界和这里的的实验室可能会有些区别，但最终所期望的结果却惊人的相似。人们被授予充分的自由，去做他们自己想做的事情。虽然这样做会有不错的运气，并不是所有人都会做出成果。

当1969年三月贝尔实验室结束参与Multics项目时，Ken Thompson正在为一个GE645型号的机器工作。他选择继续工作，因为“这很好玩”。不管怎样，好玩意味着持续不断地工作，去尝试突破硬件和软件的限制。GE是一个昂贵的机器，而Multics还没有准备好成为任何机器的生产环境。

Dong McIlroy强调，“这是Multics最早开始工作的地方，贝尔实验室的确将645作为运行Multics的机器。3个用户就可以让它满载。”

虽然Thompson继续待在伯克利，他也曾经知道Butler Lampson写的SDS930操作系统。但是在Multics灾难以后，操作系统研究在新泽西就不再流行。

Sandy Fraser在1969年五月来到贝尔实验室。他离开Ferranti是因为他想为Multics工作，然而当他来到贝尔实验室时项目就已经被砍掉了。一天Bill Baker（）和Ed David（）到了默里山贝尔实验室五楼，David读到了一个文件，提到了贝尔正在推出这个项目。Fraser回忆这是一个“痛苦的改变”，所有的人都很泄气。“玩具没了。。。动力也随之而去”，他说。在四月，五月，六月Thompson沉浸于实现一个文件系统。就像他告诉Mahoney的：

> 相比于给你一个地址，你该如何返回具体文件内容及其他类似的事情，这是最底层的事情了，永远不会有更底层的事情。我认为我只有过一或两次讨论。Dennis和Canaday以及我讨论如何顺其自然的把各个文件存放起来，这具体到很多细节：你如何存放文件块的地址，你如何存放...。我们在Canaday的办公室做这些事情，最后Canada会打一个电话。那时贝尔实验室有一个听写记录的新服务，必要时你用录音机录下来给记录员，第二条早上你就会收到打印出来的记录。第二条你收到记录，所有的缩写都没有了，像“inode”会变成“eyen”。然后我们重新来做，解释也会被他们记录下来，这也成为文件系统的工作文档。而在PDP-7（一个数字设备公司的计算机）上实现文件系统只需要一到两天。

Bob Morris于1960年加入 贝尔实验室，他说他觉得Canaday对于UNIX的贡献是早期参与者中最被低估的。“最早是Ken，Dennis以及Rudd”，他说，“其他人一两年或三年后后加入进来，我是第三年，但并没有被列入”。Steve Bourne在1970年加入贝尔实验室，他说的有点正式：

> 那时只有一台带有340显示器的PDP-7可以用，PDP-7只提供了汇编器和加载器。环境非常简陋，同一时间只能有一个用户使用这台计算机，每个用户的使用时间都很宝贵。然而一个单用户的UNIX系统不久就要来临了。星际旅行程序针对PDP-7上重写，一个汇编器和一个初步的操作系统内核正在被创建，并且在GECOS（General Electric Comprehensive Operating System）上针对PDP-7跨平台汇编...
>
> 跨平台汇编的意思是使用两台计算机，每当做出新的变化就把纸带从一个机器放到另一个上...。这个系统支持两个用户同时工作，名字UNICS是一个叫Peter Neumann的喜欢说双关语的家伙在1970年创造的，意思是“UNiplexed Infomation and Computing Service”，和"阉割的Multics"双关。有些人告诉我是Brian Kernighan改变了拼写方式，但Kernighan却告诉我记不起来改成UNIX是谁的主意。

PDP-7属于Joe Condon的组。它被“因为一些图形问题借给了Bill Ninke”。Condon回忆道：“Ninke离开去了Holmdel（贝尔实验室的另一个站点）”。所以当Thompson和Ritchie想使用它的时候它正好是空闲着的。 Ken Thompson回忆道：

> 当时是69年的夏天。实际上，我老婆正在度假，去了加利福尼亚的家里看望我父母。68年我们刚刚有了一个儿子，所以Bonnie带着孩子去了加利福尼亚一个月的时间。我分别花了一星期的时间写了操作系统，命令行，编辑器和汇编器。有了这些，就能图灵完备它自己。在这一个月的时间里，我写了一个看起来像个操作系统的家伙。他有必要的汇编器，编辑器，命令行。如果它还没有自我完备，那他也是即将自我完备的了。提供和GECOS连接...。是的，所有这些只需要一个人做上一个月。

这真的是一个人月。Thompson继续说：

> 它没有保持这样太久，我们给它添加了文件系统，这样你就可以创建文件，删除文件，或者读写文件看看他的能力。要做到这些，你需要一些脚本来执行你想要的文件系统操作，那些脚本在纸带上表示“读文件”，“新建文件”，“写入文件”...。你不得不通过纸带运行脚本，然后它会让磁盘咯吱咯吱的转一会，你不知道正在发生什么。你不能看到它究竟在做什么，你看不到任何东西。所以我们创建了一些文件系统的工具，我们用纸带通过这些工具加载文件系统，然后通过文件系统运行工具。这个就是 `exec` ，顺便说一下，调用这些工具的东西就是 `shell` ，驱动文件系统到我们想要读取的地方，他如何工作并且如何反应。在我们开始开发这些东西之前，它可能会自己运行一天或两天。

当然Multics的一些设计影响到了UNIX。但就像Ritchie说的，也有些东西“非常不一样”。

> 我们在大系统的设计思想上有些压力。Ken想做些简单的事情。大概的，对事物来说，简单实际上就是更小巧。我们能

uniplexed ? never to down? sort of known? maintaining itself?

机械计算和电子计算

当我试着了解计算机和操作系统的历史，我不得不先描绘一些其他的事情，因为我不想错过非专家。

要理解现代计算机的起源，有三个话题不得不提到：

- 机械计算
- 制表机和分类器
- 差分分析机

变化发生于1918年第一次世界大战末期，计算工具从机械计算向电子机械计算，向电子管，向晶体管，向集成电路，向今天的单芯片计算机一路发展。

机械式计算的历史非常久远。算盘可能是最早的计算工具，中国和埃及使用它已经有一千多年的历史。当西班牙人到达时，玛雅人也在使用算盘。当奈培发明对数时，他用象牙做成的骨棒计算乘法，于是发明了计算尺（在伦敦的维多利亚和艾伯特博物馆还保留了很多奈培的骨棒，它们现在归属于查尔斯-巴蓓奇）。

1642年，布莱克-帕斯卡18岁时发明了一个可以实现累加计算的工具帮助他从事税收员的父亲。30年后的1671年，莱布尼兹基于帕斯卡的机器做了更多的改进，使用一个精巧的步进齿轮实现了乘法运算，这个设计一直运用到1940年机械运算的末期。莱布尼兹在1676年把他的计算器展示给了伦敦的皇家学会。第一个取得商业成功的计算器是1820年由查尔斯-泽维尔-托马斯发明的。到1878年，销售数量达到令人震惊的1500台，平均几乎每年卖出30台。直到第一次世界大战后达雷尔仍在巴黎制造。基于1985年Frank Stephen所获专利的 Brunsviga加法器，将莱布尼兹步进齿轮替换为突出可变数字的齿轮，在1892年和1912年期间共卖出了20000台——平均每年1000台。

1850年D.D.Parmalee发明了第一个键盘驱动的计算器，第一个键盘驱动，多类型的高速计算器则是1887年由Dorr Eugene Felt发明的。

1812年，Charles Babbage提出一个新类型的计算器，一个不同以往的引擎。1823年英国政府支持了他的研究，但在1833年停止工作，并在1842年取消了这个项目，因为政府觉得它的花费实在太多了。从1833年起，Babbage把全部精力投入到一种新的计算器，一个可以自动计算任何数学公式的分析引擎。它使用提花织布机的打孔卡片操作分析引擎的几种操作。虽然只有很小一部分被制造出来，但Babbage的笔记，绘图，和部分的引擎都存放在维多利亚和阿尔伯特博物馆里。

提花织布机是自动生产的成功尝试，是几种创新的成果：1725年Bouchon将织布机的弓弦缠线器换成长长的穿孔纸带，1728年Falco又将他们换成由细线串起的卡片。1748年Jacques de Vaucanson组合了穿孔纸带和卡片。织布样式由特殊机器使用模板在卡片上打孔。此时编程机器也就诞生了。

一百年后，哥伦比亚大学的毕业生Herman Hollerith，回忆这些打孔卡片的存在。Hollerith在人口普查局有一份年薪高达600美金的工作，工作内容是调查制造业使用能源和机器的使用情况。在那里他还遇到了负责“重要统计”的John Shaw Billings。一次晚宴上，Billings介绍了在1870年人口普查中使用的由Charles Seaton的发明的统计机器。对此机器他并不看好，因为根据他的预测，按照现在人口增长趋势，1880年的人口普查最少也要7，8年才能完成，而1890年的人口普查到了1900年也做不完。“应该有一个专门的制表机器来做人口普查和类似的统计工作”，Billings说。“我们详细的讨论了”，30年后Hollerith回忆说，“我记得，他设想将个人的信息打孔在卡片的边缘上”。Hollerith设想了一个可以记录和读取这种信息的设备，并且邀请Billings和他一起把这个设备商业化。Billings是一个谨慎的人，他拒绝了Hollerith。

1882年Hollerith作为一个机械工程讲师去了MIT(那年他22岁)，教书生活使他有时间去构建他的机器。首先他考虑将信息记录在长长的纸带上，但实验证明这并不可行。1883年夏天Hollerith乘坐火车去西部旅行时，他见到售票员在车票上打孔一种图案，以此记录游客的个人信息(比如性别，头发和眼睛的颜色等)，来确保整个旅途都是同一个人。

1884年Hollerith得到了他的第一个机器专利，并在1886年管理Baltimore人口普查时的一次实验中再次提高了设计。在他实验报告的基础上，新泽西和纽约订购了机器用于记录人口死亡率。Hollerith和一些商业伙伴投标赢得了1890年人口普查的合约。奥地利政府于1890年订购了一批机器，次年澳大利亚也订购了五台。Hollerith的机器大获成功，1896年组建了制表机器公司销售Hollerith电力制表系统，并在1905年合并。

Hollerith的几个发明在业界一直延续了近80年。首先，他剪去了卡片的右上角，因此操作员们可以准确无误的操作这些卡片。第二，他以合理的价格租借机器(1890年用于人口普查的租金是75万美元，而1889年的普查则花费了500万美元)，却单独销售配套的卡片(1890到1895年期间共销售了一亿多张)。第三，他改造人口普查系统到铁路上，记录货物和旅客的信息。事实上，是Hollerith发明了可重用性。

1907和1908年Hollerith租给电力系统公司一些机器用于记录，每个月他能卖给他们一百万张卡片。卡片大小为3.25英寸宽，7.375英寸长(和旧的美元一样大小)，上面有45行10列的圆孔。

1990年，一个叫Charles Flint的企业家在纽约恩迪科特创办了国际计时公司，制造售卖考勤钟。第二年，合并了澳大利亚计算测量公司，他们生产的磅秤可以直接读到数值，这样就不再需要职员人工计算了。大概1910年，Flint遇到了穷困的Hollerith，合并他的制表公司建立了计算-制表-计时公司(CTR)。不到30个月后，Flint面试了从NCR离职的Thomas Watson，并说服董事会雇佣他。1914年5月，Watson作为总经理接管了CTR，并获得了25000美元的丰厚薪水，期权以及分红。

第一次世界大战使CTR公司商业获得很大增长。1919年Watson的技术投资提前5年收回成本：CTR成功的推出了一台打印制表机，它可以打印出从Hollerith的制表机和分类器获取到的信息。1924年，CTR更名为国际商用机器公司(IBM)。Thomas J. Watson也成为IBM的CEO和COO。

1928年推出了TYPE IV型号制表机和新的80列12行卡片。

1941年到1945年间美国国防部和Thomas Watson的公司资助的许多项目，这些项目催生了现代化的计算机。

尽管Watson在1945年说“我认为整个世界需要有五台计算机”，第一台也是他资助下建立的。来自哈佛的Howard Aiken和一个小团队在1939年开始依照Babbage的理念组建一台名为Harvard Mark I的机器，并在1944年完成。它包含有一个51英寸宽8英寸高的面板，上面有有纸带阅读器，继电器，和旋钮。这台计算器由一个4马力的马达驱动，几乎全部操作都是通过物理开关进行。它可以在23个人的操作下进行10进制的数值计算，每一个基础计算需要耗时0.3秒。这台机器一直服役到1953年，它的一部分现存于剑桥的Harvard's Aiken计算实验室，另外部分则在华盛顿的史密森尼学会。

1944年美国军方和宾夕法尼亚签订了一个合约，请他们建造一个离散变量自动电子计算机(EDVAC)。最初的团队包含Herman H. Goldstine，John von Neumann，以及John Mauchly，其中von Neumann主导了相关设计。

第一台全电子的计算机是电子积分数值积分计算机(ENIAC)，由宾夕法尼亚大学的J.W.Mauchly和J.P.Eckert于1945年着手建造，1946年在亚伯丁弹道研究实验室(BRL)提供的场地组装完成。我认为它是世界上最大的计算机，由18000个电子管(真空管)和1500个继电器组成。ENIAC是电子版本的Mark I,速度是Mark I的几百倍。Mike Muuss告诉我ENIAC的一部分在BRL展示。

1947年Eckert和Mauchly建立他们自己的计算机公司开始建造他们的通用自动计算机(UNIVAC)。他们用光了资金后又找到美国赛马赌金公司为其投资的50万美金(这应该是历史上首次计算机和娱乐相关联的事件，赌金公司需要更快的赔率计算机器，他们认为电子计算机处理这些信息会更有效率)。赌金公司掌控了Eckert-Mauchly公司，但是资金在造出机器前又花光了，Mum兄弟(赌金公司的所有者)将Eckert-Mauchly公司出售给了雷明顿兰德公司。最终UNIVAC机器在一年后的1951年完成并得到了6台订单。

ENIAC也在英国发展起来。1946年Maurice V. Wilkes从普林斯顿反回到剑桥数学实验室后，带领他的小组小组筹划建立一台延迟存储电子自动计算机(EDSAC)。它有ENIAC五十分之一大小，基于von Neumann的一篇论文中提到的方法建造。当1949年完成的时候，它成为第一台可以存储控制程序，含有高速存储以及带有输入输出设备的电子计算机。几年后，EDSAC实验室在Wilkes带领下已经有了150多种控制程序。

几乎就在同一时间，在曼彻斯特一个由M.H.A.Newman领导的小组建造一台和EDVAC相同理念的机器。发明阴极射线管存储装置的F.C. Williams，曾经和图灵一起工作建造巨人解密机器的I.J.Good，以及图灵本人都加入了这个小组。曼彻斯特自动数字机器(MADM)原型于1948年建造，1949年6月运行了它的第一个程序。MADM将分页寄存器和分页用于计算。

在这段时间里，IBM开始研制选择性序列电子计算机(SSEC)。EDSAC被称为首个电子计算机，而SSEC则被称之为首个计算机。它利用一个存储程序集成了计算功能。1948年早期在曼哈顿IBM的总部投入使用时，它被聪明的安放到街边一块大玻璃后面的房子里，这样走过的路人们都可以看到它是怎么运转的。这是一个由13000个电子管和23000个继电器组成的巨大机器，由于所有的算术计算都是由电子管完成，他的速度是MARK ID的100多倍。它包含三个

不同的存储器：一个高速电子管存储，一个继电器组成的大容量存储，还有一个容量更大的8列纸带存储。指令和数据输入时通过打孔纸带，因为有66个并排的打孔纸带，所以控制可以自动一个接一个不间断的传送。“它可能是第一台拥有Babbage和Ada Lady Lovelace所提出的条件转移控制指令的机器”，1953年B.W.Bowden写道。它移植工作到1952年被拆除，特别是其间曾为直为美国原子能委员会工作。

同年六月，John von Neumann和他的同事在普林斯顿狂热的研究者一项超前的技术，他们受雇于F.C. Williams和T. Kilburn，希望可以使用阴极射线管替换电子管存储器。

下一个硬件的进步来自于1944年Jay Forrester在MIT领导的旋风项目。旋风采用一种新型电子存储，采用由16个电子管，每个又包含256个二进制位，实现了每秒2万次16位字长的单地址操作(字是计算机存储的基本单元，包含预先指定的位数。一个8位的字可以存储一个负128到127范围的数字)。旋风项目是第一个尝试以实时持续的计算飞行运动，来确保飞行器高速稳定的飞行。Doug McIlroy告诉我1954年他是一个用户的时候，“旋风拥有核心内存，一秒钟可以执行4万条指令”。旋风项目中Forrester的“阳光男孩”是Kenneth H. Olsen，他在几年后创建了数字设备公司。Olsen参与了磁力开关阵列存储设备，但是在他的学生论文中他却建议使用晶体管替换真空管。这篇论文被Ralph L. Palmer看到，Palmer是1955年SSEC的主要电路设计者，他劝说Bloch在IBM 704机器中使用晶体管存储器。最终这个尝试失败了因为带有充足电力的晶体管驱动铁核内存是不可能的。

基于701A建造的704在1954年发布，701是IBM最长久的胜利(1952，1953)。从701到704的改进由Gene Amdahl。701是由Nathaniel Rochester和Jerrier Haddad领导的35人小组建造的。Rochester在席尔瓦尼亚曾经参与旋风项目的算术单元设计。Haddad是曾经IBM 604电子计算机团队的一员。最终，700系列包含了科研计算机(701，704，709，7040，7044，7090，7094)和商用计算机(707，705，7080)。顺便提一句，IBM 7030拥有64位字长和8位字节。1953年701每月的租金是15000美元，704的租金更高一些。就算租金如此高昂，IBM也售出了18台701和140台704。在商业方面，IBM以160万美金的平均价格售出了175台705。7080是完全晶体管构成的版本，售价220万美金。那是计算机价格都是以百万美金为单位。

与IBM合作中的Forrester接到了国防部旋风二号的合约，一台防空计算机。随着IBM更多的参与，项目1952年改名为SAGE(半自动地面防空系统)。Ken Olsen和他的团队在MIT的林肯实验室制花费了9个月就完成了内存测试计算机的设计和建造。1953年当Forrester和Norman Taylor上尉意识到他们需要一个常驻IBM的联络员，Taylor强制Olsen去了那里。Olsen越来越讨厌IBM的态度，1953年的某个夜晚，他告诉Taylor“我可以做这些人的工作，并超过他们”。这就是数字设备公司(DEC)的由来。

三年半后，Olsen向政府递交了一个商业计划，收到7万美金。他和Harlan向成立自己的公司。一年后他们从Maynard购买了逻辑模块。1959年的东部计算机联合大会在波士顿的斯塔特勒酒店举行，会上DEC公布了PDP-1计算机的原型(PDP，可编程数据处理器)，这个计算机在1960年九月上市，定价12万美金。

PDP-1是一个内存大小为4096到32768字长的18位机器，存储器周期5毫秒，每秒钟可以执行10万次计算。它在Benjamin Gurley的领导下开发，包含有3500个晶体管和4300个二极管。PDP-1拥有编辑器，汇编器，和一个名为DECAL的ALGOL语言编译器。它使用纸带阅读器输入数据，一个IBM字母打印机被作为输出接口。PDP-1是同时代中最具性价比的实时计算机，也是第一个拥有图形显示器的商用计算机。DEC一共售出53台PDP-1。第一台卖给了BBN科技公司，第二台卖给了ITEK公司(Norm Taylor创办的排版公司)。其他的则被劳伦斯利弗莫尔实验室和加拿大原子能(AEC)买走。1962年一月，Olsen向MIT捐赠了一台PDP-1。学生们使用它玩一个叫做mancala的计数游戏。

PDP-2被设计成24位，PDP-3则试图成为一台36位机，但这些都没有真正的制造出来。另一台由Gordon Bell设计的18位机器PDP-4是一个失败项目，它的存储器周期是8毫秒(作为对照，PDP-5是5毫秒)。它只需要PDP-1一半的耗电量(1125瓦对比2160瓦)，运行汇编器，编辑器，和FORTRAN语言。相比于PDP-1的12万美元售价，它只需要6.5万美元。但是Gordon Bell错了:八分之五的能耗(62.5%)，二十四分之13的价格(54%)并不是市场想要的。最终它只卖出了45台。

PDP-6是一台售价高达30万美金的36位计算机，面世没多久就被取消了。一共有23台售出，其中一台卖去了MIT的人工智能实验室。

PDP-7，PDP-9和PDP-15都是和PDP-4一样的18位计算机。PDP-8是另一台由Bell和De Castro设计的12位计算机。1965年当它以1.8万美金的价格上市时引领了小型电脑革命，最终有50000台PDP-8被售出。PDP-10是一台继PDP-6的另一台36位计算机。

1970年PDP-11的发布使DEC成为小型计算机制造商的领头羊。在卡内基梅隆大学进行学术休假的Bell做的概要设计，具体项目是在Andy Knowles的带领下进行的。PDP-11是一台16位机器(拥有种类繁多的配置和外观)，最终售出了25万台。

Gordon Bell的创造和DEC冒险做低价计算机是UNIX得以开发和成功的关键。PDP-11第一次证明了大学和科研站点可以支付部门规模的计算机。此外，Ritchie和Thompson也没有足够的资金购买一台大型计算机，PDP-11 10800美金的起始售价却在贝尔实验室的预算之内。PDP-11 16位的字长也很重要，16位相比于8位可以存储更多的信息。

翻译参考

- http://songshuhui.net/archives/53537
- http://wenku.baidu.com/view/227b730203d8ce2f006623f7.html
- http://ftp.coscoqmc.com.cn:8081/nclass/jsjcourse/kwshiqu/boke.htm
- http://www.flickering.cn/八卦天地/2015/02/计算机前世篇（三，打孔卡片）/
- http://www.flickering.cn/八卦天地/2015/02/计算机前世篇（二，机械计算器）/

操作系统

普度大学的Gene Spafford提醒我说操作系统“应该提供便利的抽象”。我之前一个名为Phil Enslow的教授曾经定义操作系统是有秩序，有规则，有策略的控制计算机共享资源的系统。这是一个很好的观点，这是一个很好的观点，因为它包含有安全策略和调度策略还有其他类似的，这些通常不被认作是人机接口的一部分，或者是应用程序运行环境的一部分。操作系统提高硬件的利用率，为用户提供便利。它可以类比为一个自身不提供什么功能的政府，但是提供环境给其他有具体功能的应用程序。

操作系统是系统资源的分配器。同时它也控制应用程序，阻止对计算机和输入输出设备的不当操作。

最初(可追溯到1954年)计算机只有硬件。(Wilkes的学生Stan Kelly-Bootle告诉我，“我偏向于认为大约在1953年或1954年Swinnerton-Dwyer研制的EDSAC监视器是第一个眞正的操作系统”)。早期的计算机Massive足有房间大小，程序员通过在一个控制台上输入程序，然后手动将程序加载到内存：首先通过前面板上的开关逐条输入指令，然后将这些指令打孔在卡片或者纸带上。程序员通过观察控制台上灯泡的点亮状态查看程序的执行情况，发现错误，或者检查内存或者寄存器的内容。运行结果或者直接打印出来，或者打孔在卡片纸带上。

越来越多的计算机硬件被制造出来，读卡器，线性打印机，磁带设备。还有越来越多通用功能库被开发出来(通用功能库无需修改就可以复制到新的应用中。起初程序打孔在纸带上，按照顺序依次输入机器)。随着硬件软件越来越多的出现，问题也随着而来。

每一个输入输出设备都有独自的特性。每个设备都有独特的缓冲，标志，寄存器，特殊位，都需要为其编写名为设备驱动的特殊子程序。

然后就有了编译器。编译器将高级编程语言翻译成机器语言。1952年Grace Hopper和他的团队在雷明顿设计了第一个编译器。之后FORTRAN，COBOL等各种语言的编译器都被开发出来。这些编译器让程序员的工作变得轻松起来，但是却让软件变得更加复杂。

FORTRAAN提供了一个很哈奥的例子。为了运行一个FORTRAN应用，操作员必须将FORTRAN编译器加载到计算机。这需要将相应的纸带放到纸带驱动器上。应用(在卡片上穿孔的)被放入一个读卡器并且输入到啊另一个纸带上。FORTRAN编译器输出的汇编语言代码需要汇编，这就需要另外的汇编器纸带。汇编器将程序和必要的库链接起来，最后输出的二进制代码就会被执行。它将在控制台上加载和调试，通常并不这样做。

在做上面所有这些工作的时候，CPU是在一旁闲置的。当挂载纸带的时候，CPU是空闲的。为了增加利用率，任务被划分为一组一组的批量工作：这些是FORTRAN，那些事COBOL等等。但仍旧还是有很多问题。

这样浪费浪费掉的计算机时间实在是太昂贵了。

第一个解决方法是离线处理，在挂载到计算机前把任务批量打在纸带上。这样在不同的机器上执行，CPU和IO操作就可以同时进行了。但是为什么不把这些工作在一台机器上同时进行呢？这样就是缓冲(为不同速度的设备临时存储数据)和假脱机(在主进程上分离慢的设备)。假脱机使得CPU和IO操作都能以很快的速度进行。同时它也带来了任务池的可能性，这将带来任务调度的登场。

任务调度带来的最大好处是能够实现多重运程。多重运程实现了我们现在使用计算机的方式，即等待一个应用完成的时候，执行另外一个应用：等待水烧开的同时，做了面包。在50年代后期，这称之为中断。外围设备在中央处理器(CPU)的调度下开始一个任务，然后独立自动工作。当任务完成或者设备需要获得系统资源，设备发送一个中断给CPU，强制把系统调度的焦点放回到设备上。

Doug McIlroy向我指出一个“有趣的退步”。当IBM 704运送到MIT的时候，已将在旋风上实现的自动操作却没有了。

多重运程的实现逻辑是多任务和分时。让我们到MIT去看看这个。

MAC项目：CTSS和Multics

MAC项目是1963年春天由J.C.R. Licklider建议在MIT发起的。他的创始主任是MIT的教授Robert M. Fano。根据1965年Fano为MAC项目进度报告写的前言，MAC代表机器辅助认知(Machine Aided Cognition)和多路访问计算机(Multiple Access Computer)。事实上，它之所以有名是因为MIT计算机科学所在的科技广场五楼。Marvin Minsky的人工智能实验室位于被称之为人与计算机(Man and Computer)的九楼(最初实验室是属于Minsky和MacCarthy的，但当LISP的发明人John MacCarthy去了斯坦福后，他爱的名字被取消了)。

1963年MAC项目主办了一个暑期课程。许多知名的的计算机科学家被带到剑桥，他们使用IBM的使用CTSS(the Compatible Timeshare System，兼容分时系统)并讨论计算的未来。MAC项目这时主要的成就是Fernando J. Corbato的CTSS，Victor H. Yngve的COMIT，还有Minsky的人工智能实验室。

在50年代和60年代早期是批处理的时代。但这既浪费时间又浪费资源。桌椅浪费是因为机房里无所不在的签名表。除非你很幸运，刚排队没多久就就让你运行一个像debug这样预约时间为一小时小任务。你的程序运行了40分钟，或者不幸的，你的程序一小时后还没有完成：它将被直接丢弃(已经完成的部分很可能就完全丢掉了)。然后你又要将来的一个半小时(通常在下午的1点，2点或者3点)。解决这个问题的答案是分时。

兼容分时系统(CTSS)是第一个分时系统，它是由Corbato带领团队在MIT计算中心开发的。CTSS运行经过修改的的IBM 7094，它通过添加两块IBM 720A增加了32K字节存储。另一台7904在1965年早期加入进来。通过一台连接到拨号调制解调器的IBM 7750会话控制器，它最高可以让30哥用户通过电话线远程访问(事实上没有人可以拥有一个小时的CPU时间，因为那已经超出了系统失败的时间上限)。

CTSS作为MAC项目的分时部分，既是一台服务设备，又是系统编程员的实验室。这是这个附加功能(当时7094还在调整)导致了Multics的诞生(Multiplexed Information and Computing Service，多路复用信息和计算服务)。

Multics是第二代分时系统，目的是成为一个“计算机公用程序”的原型。它在1964年作为MAC项目，贝尔实验室，以及通用电气的合作项目开始进行。Corbato是开发计划的主要领导人，项目有九个主要目标：

> - 方便的使用远程终端
> - 像电话系统一样的连续运行
> - 广泛的系统配置
> - 可靠的内部文件系统
> - 支持选择性的信息分享
> - 层次结构的信息分类
> - 支持大量的应用程序
> - 支持多个环境和接口
> - 系统进化的能力

Berkley Tague在1962年加入贝尔实验室，又于1965年作为项目三人组的秘书加入Multics项目组。他说在1967年就可以很明显的看出项目不会成功，因为“三个成员有着互相冲突的目的”。他去新泽西惠帕尼的AT&T工作了三年，在那里他工作于一个防护项目。1970年又重新回到纽约默里山。

MIT的Jack Dennis教授在Multics早期贡献了几个很有影响的架构建议。当需要寻找一个计算机厂商支持新的操作系统时，人们说IBM将要生产叫做360/65的机器。Doug McIlroy告诉我：

> 这是真的，但是Amdahl不会制造虚拟机。最后时刻IBM让Gerry Blaauw主导360/67，但是为时已晚。

IBM的员工对MAC项目组关于分页的主意不感兴趣。那时还是MIT讲师的Joseph Weizenbaum向在通用电气的前同事们介绍了MAC项目组，前同事们对于分段分页机制非常热心，在他们的提议下产生了GE-645(635的升级版本)。就这样通用电气成为Multics项目的一部分。

MIT，贝尔实验室和通用电气一致同意合作。分别来自三个公司的三个人人组成了三人决策组，他们是MIT的Fano，贝尔实验室的E.E.David，和通用电气的C.W.Dix。还有三个人负责具体的实际管理，分别是MIT的Corbato，通用电气的A.L.Dean，贝尔实验室的Peter G.Neumann。值得注意的特性包括分段内存，虚拟内存，高级语言(PL/I)，多处理器共享内存，多语言支持，还有在线配置。

1964年PL/I被选为编程语言。其他候选的还有MAD(the Michigan Algorithm Display)和AED-0(an MIT display)。实现完整的PL/I比预料的还要困难。贝尔实验室和两个外部人员签订了奖励协议。但是一年过去了，两人没有拿出一个PL/I编译器。Bob Morris和Doug McIlroy制定了一个备用计划，在7094上使用McClure的TMG语言，这个语言也被称为EPL(早期PL/I)。McIlroy回忆道：

> 1965年5月3号，我们突然发现协议无法按时完成，马上就就开始自己建造它。我们在年前一段时间就有顾客。Morris的66到67年的学术休假被完整的使用了。我们雇佣的签约人员没有做出任何有价值的东西。所以EPL一直被推迟到1970年通用电气一个精湛的团队做出真正的编译器。
>
> Morris编写了TMG的前端，它输出单路ASCII代码。每个地址都注释着“基址，偏移，模式，精度”这些让人不知所云的东西。我用TMG语言做了代码生成器，它输出汇编语言。Dolores Leagus加入进来做数据结构编译。我们排除了占据PL/I一半语法的输入输出部分，单精度，属性声明，跨段序列，但实现了一个显著的子集，包括IBM早期版本性感的“引用”属性(“引用”使得序列的大小信息可以保留在数据结构的元素中)。
>
> Jim Gimpel加入进来负责数据封装。Multics有着种类繁多的数据结构，我们将一些三路代码加入的中间语言以消除很少使用的临时变量。编译器缺少某些功能，它只能输出“语法错误”和“多重声明”这两种错误提示，当数据越界时提示“非法数据结构”的警告。
>
> McClur将TMG的汇编代码从CDC 1600翻译为7090的编码格式，我在千里之外帮他调试。Clem Pease随后又通过将IBM指令定义为635宏将将它移植到GE 635。

编译器将编译结果送入EPLBSA(EPL引导汇编器)。编译过程非常缓慢。

等待PL/I编译器最终可用的时候，团队编写了Multics系统编程人员手册。它一共有3000页，每一节都经过几轮修订，还有几节重写或者大改了几次。

Tom Van Vleck说：

> 在68年到69年系统已经延期并且资金紧张有被取消的风险。或许这恰好鼓舞了团队精神(相比于表面的士气)。1968年ARPA(the Advanced Research Proect Agency of the Department of Defense)的一次审查是我们最接近被取消的时候，不过他们建议我们继续下去。但为时已晚，贝尔实验室在1969年四月正式退出。

直到1993年仍有Multics机器在运行，它们是稀有物种。Van Vleck像我提供了下面的清单，标记型号的是未能确认的站点：

> - ACTC (加拿大 亚伯达 卡尔加里)
> - Air Force Data Service (华盛顿哥伦比亚特区) *
> - Bull System-M (亚利桑那州凤凰城)
> - Credit Lyonnais (发过巴黎)
> - Department of National Defence (加拿大 新斯科舍省 哈里法克斯)
> - Ford Motor Company (密歇根州 迪尔伯恩)
> - General Motors(密歇根州 沃伦)
> - Ministerie van Sociale Zaken en Werkgelegenheid (荷兰 海牙) *
> - National Security Agency (马里兰州 林西克姆)
> - Societe Europeenne de Propulsion (法国 弗农)

Multics的重要性在于它持久的影响力。霍尼韦尔公司(Honeywell)买下了通用电气的计算机业务，但是它的GCOS 6操作系统，Prime的Primos，Stratus的VOS，还有Apollo的Domain，以及BBN创建的TENEX，TOP-20，都深刻的被Multics影响着(Prime的创建者Bill Poduska曾在通用电气为Multicsul工作，然后去了DEC，Multics曾经的多语言经理Bob Freiburghouse是Stratus公司的创始人)。英特尔386，486系列的内存管理看起来是来自Multics的手册。但最重要影响恐怕是来自贝尔实验室的Dennis Ritchie和Ken Thompson，以及他们的同事Mcllroy，Morris，Neumann，以及接触了CTSS和Multics的Ossanna。Ritchie告诉我：

> 我们从Multics学到了很多重要的事情：
>
> - 树形的文件系统结构
> - 一个命令解释程序，甚至这个程序的名字“shell”也是取自Multics
> - 更重要的是，文件的结构就是没有结构，之看做字节数组。大多数情况下操作系统对此不作解释
> - 文本文件只是有换行字符分隔的字符序列
> - 读写操作的语义都是针对一个文件句柄，一个缓冲区，一个计数。对上隐藏了底层的磁盘簇块

译者注

Multics System Programmers' Manual

PDP-11

Dennis Ritchie，Ken Thompson和Joe Ossanna曾经多次试图说服贝尔实验室为研究小组购买一台计算机，但是在69，70年的时候一台计算机意味着十万美金的价格。尽管是做文件系统和相关工具的研究，但是计算机研究小组还是没有自己的计算机。他们试着让贝尔实验室购买一台PDP-10，或者去租一台，但他们始终没有如愿。Dennis Ritchie对Peter Collinson说：

> PDP-7上的所有软件都是用汇编写的。TMG是一个自顶向下的操作系统编译器，最早是由Bob McClure编写，Doug McIlroy也实现了一个。我们的PDP-7上只有4K的内存供应用程序使用，显然不够用PL/I。所以Doug McIlroy为PDP-7移植了TMG。那时Ken Thompson决定我们不能有一个不提供FORTRAN编译器的操作系统，所以我们都坐下来通过TMG开发一个FORTRAN编译器，这些事情只花了大概一天就搞定了。接下来就出现了B。B是BCPL的一个精简版本。

BCPL(Basic Combined Programming Language，基础组合编程语言)是Martin Richards在1967年“编写编译器和系统的辅助工具”。B是BCPL的精简版本，它的名也同样精简了下来。

Richine解释说：

> 那时候(1968)我加入到实验室，最早是和Rudd Canaday搭档。他曾经将BCPL编译器移植到通用电气的GE-635上的GECOS系统。那时他正在将BCPL移植到Multics，所以我接触到了运行BCPL的Multics。那是非常早的一个移植版本，我们移植后Martin Richards一直在持续改进它。
>
> 既然有了BCPL，我们通过它写了一些简单的程序。在第一个Unix系统上面B是基于BCPL的精简编程语言。它作为一个解释器输出的并不是机器码，而是一种中间码。
>
> 它先是用TMG写成，然后就实现了自举。事实上这个语言的开发过程非常有趣。编译器不断地用尽机器所有内存，Ken不得不修改编译器，但这又使得再添加新特性变得异常困难。之后他使用新架构才使得编译器变得小巧起来，添加特性也变得比较方便了。
>
> B的确是Unix上面的第一个高级编程语言，有一些东西就是用它编写的。它一共有两个具体实现版本，其中一个输出基于栈中间码的普通解释器，另一个被称之为vb(virtual B)的版本可以将机器的4K内存分页映射，解释器因此可以使用大于物理内存的内存。那时我们的一个程序需要比物理内存大的多的内存，我们用vb实现了它。

Andrew Hume告诉我说“B语言一直使用到1989年，控制排字机的程序就是用B写的”。但硬件的种种限制还是让人沮丧。最后Ossanna建议购买一台PDP-11/20用于文书处理系统。Ritchie说：“Lee McMahon认为我们做的很好”。在他的努力之下(和对文书处理的信心)，声学研究总

监Max Mathews资助了种子资金。McIlroy告诉我说“没有外部计算机科学的帮助，Unix可能永远都不会诞生”。实验室的管理员认为文字排版是又用的项目，于是他们预定了PDP-11。Ritchie说：

> 我们在1970年夏天很早就拿到了PDP-11，但只有处理器和内存运到，磁盘还没有过来。它的所有软件都是纸带方式输入，你只能用纸带输入东西，并且还没有操作系统。Unix诞生的第一件事是在PDP-7上用B语言写的PDP-7汇编器编写PDP-11的跨平台汇编器。
>
> 我们拿到PDP-11的时候，只有很少一些特殊基础命令用B语言重写了。第一个完成的应用是PDP-11汇编器。还有一个很早期版本的dc(desk calculator)，是事实上的第一个跑在PDP-11上的应用，在没有操作系统之前就存在了(标准的“DEC操作系统”从来没有安装到PDP-11上)。它是一个很原始的汇编器，其语法和PDP-11的原始指令集几乎没有区别。
>
> Unix的诞生可以分为两个阶段。Ken在还没有磁盘的时候就让他跑了起来，他把内存分为两大部分，一部分用来存放操作系统，另一部分存放一种内存磁盘。要想让这个系统运行，你需要使用纸带加载初始磁盘的代码，然后是加载操作系统代码。所以在有磁盘之前，就已经执行了cp(复制文件)，cat(链接文件)，以及ls(列出文件)命令。
>
> 一旦有了真正的操作系统，就有了退步。B语言版本的汇编器运行速度非常慢，他们只好用汇编语言重写。我猜测还有不少用B语言写的东西。其中较早的一个是展开文件名中星号和其他一些特殊字符的global命令。我不会忘了支持global的原因，他有点不太合理。当shell应用在传入的参数中发现了魔法字符，它就会调用另一个程序展开它们。程序的代码并不在shell程序里面，它是用B语言写成的。

Thompson说当等待PDP-11的磁盘的到时候，它们把PDP-11放在旁边。然后它们把代码移植过去，在内存里运行了文件系统。

但当PDP-11启动起来后，为了可以编程还需要一个文本编辑器。麻省理工的PDP-1有一个名为TECO的编辑器。起初，这是Tap Editor and COrrector的缩写，后来变为Text Editor and COrrector。TECO是EMACS(Editing MACroS)的前身。1967年L.P.Deutsch和B.W.Lampson在SDS-940上面开发了一个名为QED(Quick EDitor)的TECO实现。SDS不久成为Xerox。Thompson在麻省理工曾经在IBM 7094上为CTSS编写过一个QED。他和Ritchie后来又在贝尔实验室为GE-635写了一个。现在Thompson为PDP-11写了一个简单的线性编辑器ed。但PDP-11已经被用作文书处理用，所以这个文本编辑程序应该还要能把文本显示出来。

> J.E.Saltzer曾经写过为CTSS写过runoff。Doug McIlroy回忆说：
>
> 我相信是Morris或者Thompson将runoff移植到了635并且改名为roff。它是一个快速开发的东西，只用了一夜的时间。然后我用BCPL语言重头实现了一个简单却超越runoff的roff。那成为Thompson和Ritchie在Unix上实现(用机器语言)的样本。

McIlroy后来写信给我说：“Ken已经不记得635上面的roff，因此我猜测他其实并没有参与roff的开发。然而他记得他曾大概在Unix之前为PDP-7写过一个叫rf的小程序”。这看起来有些矛盾，后来又“猜测”是Ritchie实现了它。

Ritchie说“我知道有一个骗局正在发生，我们承诺了一个文本处理系统，而不是一个操作系统”。但文本处理的努力付出是成功的：贝尔实验室的法务部门和研究小组共享PDP-11/20，成为第一个Unix用户。更为重要的是，实验室的法务部门随后接管了运行Unix的11/20，并且给计算机研究小组拨款购买了PDP-11/45。

1971年，第一版

伴随着PDP-11/45的出现，系统也在慢慢成长。它慢慢发展出了指令集，子例程和游戏。对你来说了解这些信息的唯一途径是和作者们坐下来提出问题。

Joe Condon在去计算机研究组之前他的小组是PDP-7的拥有者。他回忆起Mike Mahoney，Bob Morris是如何启迪他开始思考Unix的。

> 因为手册写的并不是很清楚，我会过来问“你是怎么知道这些命令到底做了什么”。Morris说“你觉得怎样才是合理的？那就试着去做，joe”。这让我觉得他和其他一些人(应该也包括Dennis)对一个系统命令应该如何工作的设计哲学很是有趣--它的工作方式应该让人非常容易理解，它不应该包含了许多非常复杂的隐藏规则，以至于超出人们的认知领域。
>
> 我觉得Bob Morris告诉我的是黑盒自身应该足够简单，以至于你可以从黑盒的行为上判断黑盒内部是如何工作的。你不应该写一个让人困惑的程序，让人们猜测它到底在做些什么。你应该让他的功能一目了然。

据Ritchie说，是McIlroy抽空写了编程手册。而McIlroy说是Ritchie设计编写了第一个man page。看起来就像是cat--catenate(环状)。Ritchie说他想不起来是谁提出的格式。不管怎么说，起初的设计非常的优秀：我们依然在使用它。Doug McIlroy认为编程手册保持了连续完整。Ritchie做的事情是让风格更加连贯。今天看来man page有些过于简洁，但无论怎么说，他们都比其他操作系统的帮助信息要好很多很多了。

后来贝尔实验室研究部门的领导Sandy (A.G.) Fraser，和Mahoney谈论McIlroy和手册：

> 现在你可能认为那是一个文书工作，但那时错误的。实际上仅仅是他对于手册保持高标准，就意味着他会对手册中提及的任何程序都坚持高标准。接下来他们做的就如他们所说，为了编写下一版本手册重写了一系列的程序，以让它们保持同样高的标准。另外还要说的是，它可能是第一个包含bug章节的手册，你很难找到其他这么诚实的手册。他们不是简单的写下懒得去修正的bug，实际上它们修正了很多的bug。但有一些不是很简单修正就可以的，或者是不确定应该究竟怎么做，因此把它们写在文档里面。我想这种聪明的诚实是非常少有的。

Doug McIlroy也说到过手册：

> 把事情理清楚再去谈论正式Unix的特点。每次编写另一个版本的手册，每次写下丑陋的东西，你都会惊慌的说“我们不能把这打印出来”。为了更加容易讨论，你不得不对文档增增减减，这是做科研的美德。你不必保持旧软件的运行。

实际上，当我仔细翻看最早的四版手册(1971年11月到1973年11月)，我强烈的感受到每次编写之前都有大量的变更：系统不断趋于完善，文档也反映出来。第一版手册的第二段记录着：“变更的频率是如此之快，以至于每当要写一章是前面的章节都要做大量的修改”。不过不

稳定也代表着系统正在不断成长。阅读Thompson写在每一版手册前面的序言尤其能体现这一点。

第一版《UNIX编程手册 K.Thompson D.M.Ritchie著》出版日期是“1971年9月3日”。它以两页半的“简介”开头，然后是只有三页的“目录”。目录分成几节，这些章节包括命令，系统调用，子例程，特殊文件，文件格式，用户程序，以及其他混杂。这些内容在领域内有着广泛的影响。十五年后的1986年，伯克利版本Unix(4.3BSD)的第七卷手册以同样的章节排列。虽然过了这么多年，这七节之外又有所添加，第六节明显成为游戏章节--起初它包含四个游戏：**bj** blackjack, **chess**，**moo**是一个猜谜游戏，还有**ttt** tic-tac-toe。这四个游戏和**sort**(将文件内容按字母排序)，**cal**(输出日历)放在一起。**chess**尤其重要，因为Ken Thompson是一个专业的象棋选手，十年后他编写的Belle象棋程序获得了世界计算机象棋冠军(当在荷兰谈论Belle，Thompson被问及机器是不是一个人造威胁，他回答“只有从飞机上掉下去才是”)。

1973年有一个已经被提出的文本处理特性是构造交替式索引的能力，每一个项目都有一些列的曝光机会。因为在手册的不同章节中有很多相似的命令名字，标记命令章节是为了避免读者困惑。就像**cp(1)**意味着第一节的复制命令，**dc(1)**表示第一节的桌面计算器，而**dc(4)**则表示第四节中用于远程打印机的特殊文件(这个命令在第6版之后取消了)。**cat(1)**意味着“连接文件”，而**cat(4)**则代表“C/A/T 照相排字机接口”。

简介在各方面来说都是很好的文档，因为他给读者一个关键词对照表，就像下面这个：

ken	K. Thompson
dmr	D. M. Ritchie
jfo	J. F. Ossanna
rhm	R. Morris

Unix社区为贡献者保留他们个人信息的传统就是从这里开始的。在很多社区这个传统一直保持着，相比之下很多大公司就从文档，初始化界面，登陆页面中隐藏掉了这些名字(例如DEC，HP，IBM，Sun)。Lorinda Cherry评价说着包含的原理非常简单：“谁最后修改了，就属于谁”。并且所有者会被在登陆页列出来，因为他们有电子邮件或者像**write**这样的交互程序。

第一个版本酒精包含什么？我想最简单的描述就是：非常多。在系统所有的指令中，只有我们后面提到的**pipes**没有在这提及，在第一节一共有60个用户命令：

ar	archive files	chdir	change working directory
as	assembler	check	file system consistency
b	compile B program	chmod	change access mode
bas	BASIC dialect	chown	change owner
bcd	convert ASCII to BCD	cmp	compare file contents
boot	reboot system	cp	copy file
cat	concatenate files	date	get date and time
db	symbolic debugger	od	octal dump of file
dbppt	write binary paper tape	pr	print file with headings
dc	desk calculator	rew	rewind DECtape
df	find free disk space	rkd	dump disk to tape
dsw	delete files interactively	rkf	format RK disk
dtf	format DECtape	rkl	load disk from tape
du	find disk usage	rm	remove [delete] file
ed	text editor	rmdir	remove [delete] directory
find	find file with given name	roff	run off text
for	compile FORTRAN program	sdate	adjust date and time
form	generate form letter	sh	command interpreter
hup	hang up typewriter	stat	get file status
lbppt	read binary paper tape	strip	remove symbols
ld	link editor [loader]	su	become superuser
ln	link to file	sum	sum file
ls	list directory contents	tap	manipulate DECtape
mail	send mail	tm	get time information
mesg	permit or deny messages	tty	find name of terminal
mkdir	create directory	type	print file on IBM 2741
mkfs	create file system	un	find undefined symbols
mount	mount detachable filesystem	wc	get word count
mv	move or rename file	who	who is on the system
nm	print name list	write	write to another user

这60条命令中，有许多我们仍然非常熟，虽然很少还有纸带阅读机，DEC RK05s，或者IBM 2741s存在。其他一些命令已经失去具体作用或者演变成其他的东西：**dsw**(delete using switches)被**rm**(remove)命令包含，**db**在第7版中成为**adb**。**for**在第2版中成为**fc**，第7版中成

为**f77**。在第5版中**lbppt**成为**restor**，**dbppt**成为**dump**。许多命令消失了：**dtf**在第1版后就取消了，同样的还有**hup**，**rkd**，**rkf**，**rkl**，以及**sdate**。

著名的**create**创建新文件命令出现在第二节，系统调用章节。Thompson脸上带着笑容回忆说他最大的错误是保留了**create**的-e选项。

Doug McIlroy告诉我Thompson最大的错误其实是“将Unix全部用大写字母拼写。Unix，Fortran，还有其他的计算机名词都是很好的名字。一张这样的纸比一张全是大写字母的纸更能吸引别人的兴趣”。这段评论导致我在这本书的大部分都用Unix代替了UNIX。就像之前说的，系统一直在持续演变，所以接下来的两年是非常让人兴奋的一段时间。

> 贝尔实验室**Unix**版本列表
>
> 注意：这是AT&T贝尔实验室十个版本的*UNIX*编程手册，第十版在1990年发行。从第一版到第六版的标题页都署名Thompson和Ritchie，第七版虽然有一个由B.W.Kernighan和M.D.McIlroy写的简短前言，标题页却只写了*UNIXF*分时系统而没有署名。第八版第九版的前言是由McIlroy一个人写的，同时在标题页注明了研究版本。第十版的前言署名是“A.G.Hume，M.D.McIlroy，1989年10月”。

第一版	1971，11，03
第二版	1972，06，12
第三版	1973，02
第四版	1973，11
第五版	1974，06
第六版	1975，05
第七版	1979，01
第八版	1985，01
第九版	1986，09
第十版	1989，10

译者注

- Unix Programmer's Manual

1971-1973，C语言和管道

在贝尔实验室的个个部门中Unix无疑是成功的。几个月后，Thompson和Ritchie开始为一个新的手册工作。1972年6月中旬发布了第二版。前言部分写着：

> 在这个手册发行后几个月内，系统自身和它的操作方式都有了巨大的变化。
>
> ...
>
> 有更多的人花费时间参与到UNIX的开发中来。感谢L.L.Cherry，M.D.McIlroy，L.E.McMahon，R.Morris，以及J.F.Ossanna做出的贡献。
>
> 最后，UNIX已经安装到了10台机器上。

第一批安装用户当然是研究部门和法务部门。新泽西中部以外的首位用户是Neil Groundwater，那时他在纽约电话中心工作(如今在Sun公司)。他告诉我说：

> 1972年我在滨州州立大学完成学士学位后加入纽约电话公司。那是计算机专业的毕业生就业是很困难的，我被雇佣的原因是我在大学里有一些使用迷你计算机(ADAGE图形系统)的经验。1972年春天我去新泽西维泊尼的贝尔实验室工作。我在曼哈顿有个房子并且没有汽车，周一早上坐汽车去维泊尼对我来说很方便，然后在那里找个旅馆住下周五再返回纽约。一周都住在公司附近让我很快就沉浸在工作中。我对工作的记忆最多的是一个分享Unix系统如何使用的回忆：这和从烟囱内部向上爬非常想，你不得不从底层的方方面面一点点的开始。系统的很多部分的确是有文档，但是就像我们很多年后所说的“去看源码，卢克”。
>
> 我在维泊尼工作小组的任务是将外部设备的分析工作机械化，其中包括纽约的ESS(电子交换系统)站点。ESS的机器在中央办公室的电传打字机上产生呼叫失败消息(我们称之为“TN08”)。虽然Unix主机并不连接控制“交换”，但可以使用巧妙的线路让一个调制解调器连接电传打字机，另一个调制解调器连接到我们办公室(曼哈顿麦迪逊达到330号14层)，如此就可以发送信号到一个多路复用端口。
>
> 1972年夏天纽约站点的硬件运到了：
>
> ```
> DEC PDP-11/20处理器
> 56Kb核心内存
> 高速纸带读写器
> ASR-33 电传控制台
> DEC tape
> RK11/RK05磁盘 2.4Mb
> RF11固定磁头磁盘(最初由两个，后来又添加了3个)
> DC11(6路)终端
> DM11 16路多路复用器
> ```

> 我的第一个编程任务是为DM11多路复用器重写驱动来处理多个多路复用器(目标是3个)...
>
> Unix的shell程序存放在七页行式打印纸中。没有任何的程序库，直接使用系统调用控制读写。shell中有很多我们今天仍在期盼的特性：输入输出重定向(不使用管道)，元字符扩展(虽然是使用/etc/glob的外部程序执行扩展)，还有少量的shell脚本。
>
> 一个有问题的程序可以很轻松的覆写内核代码。PDP-11的**halt**命令是代码“0”，因此你可以想象使用寄存器指针清空一个内存引用会使CPU立即停止工作。你可以通过编写程序或者使用键盘快捷键转储一个用户程序，这就是故障后调试。
>
> 开发程序通常会在上班时间之外或者在维泊尼。在维泊尼开发机的终端机被放置在一个公共房间，当几个人同时工作时，一个员工在运行**a.out**(链接器的默认输出文件名)之前可以广播一句“即将运行危险程序！”，其他员工看到这个就要赶快保存他们的文件了。
>
> 当某个员工要使用线性打印机时也会呼出一个类似的广播。`pr myfile > /dev/lp` 将会把文件发往打印机，因为打印机没有脱机缓冲或者锁定机制，当两个人同时发送数据，就会将文件内容混淆在一起打印。最先喊出“线性打印机！”的人获得优先使用权。
>
> Unix系统通过向连接到中央办公室的调制解调器发送数据和ESS办公室通信。但在中央办公室接收信号并没有发送给ESS的电传设备，而是发给了一个硬拷贝终端(类似于TI700的热敏打印机)。因为DM11的收发速度是不同的，电传接收的速度是110波特，而发送速度是300波特。
>
> 搜集错误报告后使用FORTRAN编写了一个程序，包含特定设备的特征的报告会被整理到一份“异常报告”。不稳定的电话网络可能会导致错误报告发送给外部设备，而不是当呼叫没完成时交换设备重新试着呼叫并且向终端打印错误报告。实际上有几种不同的把报告类型，但在新系统中TN08总是第一个被排查的设备。
>
> 启动PDP-11/20需要从启动地址(一个磁盘启动地址和一个纸带启动地址)加载到控制台开关，然后按下执行开关。启动ROM从一个DEC的满是二极管的电路板上加载(一个二极管是一个字节位)启动指令。
>
> 1973年在同一站点，我们添加了一个类似PDP-11/20配置的机器。它有一个光学读卡器读取“合适”的卡片，卡片上由面板办公室(一种老式交换设备)的技术人员在发送器死机时用铅笔绘制数据。“面板死机发送器”是他们称呼像TN08这样分析ESS设备的术语。如果你收集到足够多的报告，计算机可以识别重复的内容并且帮助你定位问题的情况。

第三版在8个月后的1973年出现。贡献者名单添加了E.N.Pinson的名字，最重要的是：

> 最终，Unix的安装量达到了16台，超出期望。

目录之前的前言部分有一叠纸之多。在开始部分增加了一些章节，“如何通过终端交流”，“shell”，还有路径名，在“编写程序”和“文本处理”章节还有一些图表。它们以如下文字开头：

> 在Unix，要将程序源码输入一个文件，使用ed(1)。在Unix中有三个主要的编程语言，它们是汇编(参见as(1))，FORTRAN(参见fc(1))，和C(参见cc(1))。

C？C是什么？手册中关于cc的部分(1972年3月15日)告诉我们它是一个C语言编译器，并且引导你去看“C语言参考手册”。五年以后Brian Kernighan和Dennis Ritchie出版了 `The C programming Language` ，尽管手册更早的融合到了文档中。用这种新语言重写Unix(第四版)是极为重要的，而且C语言自身也成为计算机界重要的一部分。

Mike Mahoney向Dennis Ritchie询问关于C语言的设计细节：

> C语言几乎就是Ken的B语言改进版。B语言实际上起源于系统的FORTRAN语言...总之，浪费了一天的时间后他意识到他一点也不想做一个FORTRAN编译器。所以他开发了非常简单的B语言，并且让它运行在PDP-7上，后来又移植到PDP-11。除了操作系统以外，辅助程序中不少都是用B语言写的。因为它是一个解释型语言，所以运行起来相当缓慢。对于B语言有两种看法，一种认为因为它的实现是解释器，所以它会一直很慢。第二种看法是，不同于我们之前用过的所有面向字设计的机器，我们现在用的机器是面向字节设计，而基于BCPL语言改进而来的B语言并不适合这种面向字节的机器。特别是B和BCPL都有名为指针的存储单元...这些对象在不同的机器上大小是不一样的，而B和BCPL都是面向相同对象大小设计的。从语言学上看这是B语言最大的限制。事实上不仅所有的对象都要有同样大小，而且所有指向对象的指针也不合适...因此，大概在同一时间，我尝试着向B语言加入类型，没过多久就又尝试着为它写一个编译器。语言第一次发生了改变。那时我们把更改后的语言叫做新的B语言，简称NB。它也是一种解释型语言，实际上我已经着手开发B的编译器...因为C语言是使用一个和它非常相似的语言写成，每一个步骤都很像...把它们都合并到C语言编译器，然后添加变量，添加类型结构。最后试着把它做成一个编译器。
>
> 编译器的基本结构机器码生成器的构建工作，基于我从印第安纳山贝尔实验室里某人听到的一个主意。尽管我从没有读到相关的论文，但是我从NB语言的代码生成器中得到了相关的技巧，然后成就了C。这一切都因为这位博士的论点。这个技术也运用到了用于交换系统和ESS机器的EPL语言，它支持了ESS编程语言。因此决定C语言成功两步中的第一步就是从B语言改进，加入类型结构，语法上不做太多其他的改变，然做出一个编译器。
>
> 第二部是一个缓慢的过程。它发生在几年之后，看起来真的有点慢。它来源于第一次试图用C语言重写Unix。Ken大概在1972年夏天开始尝试重写，但最后放弃了。这或许是因为他厌倦了或者还有其他什么原因的。但有两件事情是错误的，第一个是他的问题，他不能解决运行基础的协同程序，起始的多道程序——如何在进程间切换控制权，以及内核和不同进程间的关系。第二件事情是他没能解决合适的数据结构，在我看来这更重要一些。起初的C语言没有数据结构，因此创建对象表——比如进程表，文件表，以及各种表——是一件非常痛苦的事情...做起来非常笨重。我猜人们也在用FORTRAN做着同样的事情。
>
> 所有这些事让Ken在那个夏天放弃了。第二年，我添加数据结构，让编译器或许变得更好用一些，代码写起来也更方便。因此，在下一个夏天我们一起努力用C语言重写了整个Unix系统。

第三版中的创新内容还有管道。管道(还有过滤器)是非常简单的概念：它是把一个程序的输出连接到另一个程序的输入的同一机制。在管道出现之前，Dartmouth分时系统就拥有通讯文件，但那是一种无法通用的特殊方式。管道概念是由Doug McIlroy提出，在McIlroy的坚持下由Thompson具体实现(“它是我少有的在Unix管理中实行控制力的地方”)。McIlroy告诉Mike Mahoney：

> 大概在63年，Conway写了一篇关于协同程序的文章发表在CACM(ACM通讯)上。我在59，60年的时候发明了宏(宏是一个指令代指一些列的指令，它是一对多的映射)。如果你仔细思考宏，它主要包含了数据流的切换。我的意思是，你正在输入数据，突然你执行了一个宏调用，那就是说“在这里停止输入，去执行定义的命令”，而在宏的定义中你又会发现调用了另外一个宏。所以在64年之前，我就在某些地方谈论宏是一种“数据流调度器”。也是在64年，一篇文章一直钉在Brian的墙上，当我谈论缠绕的数据流就像花园里的水管时，他想通了一些事情。所以这个点子在我的脑袋里存在很长时间了。
>
> 在同一时间，Thompson和Ritchie正在他们的黑板上描绘一个文件系统，我在这个黑板上画出如何通过将进程连接在一起处理数据，我在寻找一种前缀语言将进程连接在一起。最后我失败了，因为你很容易就会说“cat这个，grep那个”，或者“谁去cat去grep”等等。你很容易就说出这些，从这里可以很容易看出我们习惯于这样说。但是这些命令都有很多的参数，而不仅仅只是简单的输入输出，它们有各种选项，在语法上很难将这些选项附加到前缀语句中的各个命令，cat到grep到who(例如cat(grep(who)))。句法上不知道该怎样描述清楚。所以我在黑板上用这种语言写的漂亮程序没有足够强壮去应对现实中的情况，我们后来的确也没有实现它。
>
> 在1970年到1972年这段时间我一次又一次的说，“我们要怎样才能把事情做好？”，我放弃了一个又一个的提议。突然有一天我想起一种shell的类似水管的语法，Ken说“我这就实现它！”。他已经听过很多诸如此类的提议，那天绝对是让人难以置信的一天，他说“我要把它做出来”。 他并没有完全按照我的提议用系统调用实现管道，而是发明了一种更轻量更好的方式，也就是我们今天一直在用的这种。他采用了我提出的笨拙语法。
>
> 他将管道放进Unix，他将这种概念(这是McIlroy指着黑板，上面写着：f>g>c)放到shell里，这一切只用了一晚的时间。那时候大多数的程序都不能接受标准输入，因为并不真的需要。所以它们都有一个文件参数，gerp有一个文件参数，cat也有一个。Thompson发觉这些程序不能符合管道的使用方式，那个晚上他把这些程序也一起修改了。我不知道他是怎么做到的，第二天早上我们有了一次狂欢。

后来成为PWD主管的Dick Haight对August Mohr说：

> 我很偶然的在他们实现管道的那天见到了实验室成员。显然对于每个成员，系统成功使用管道简直是一个太过美妙的事情。没人会拒绝放弃这个新发明。

在那晚之前，Unix没有“工具箱”的概念。管道的发明为接下来几年对软件的探索探索之路奠定基础，这种理念成为一种独特的哲学。每次提到工具箱，McIlroy都会说：管道创建了它。

Mahoney问他：“管道之后Unix看起来有什么不一样”？McIlroy回答：

> 是的，每个人都开始提到那种哲学，“这是Unix哲学。一个程序只做一件事情，并把事情做好。写不同的程序，让它们一起工作。让程序处理文本流，因为那是普适的接口”。在管道之前这些工具观点还没有形成，但管道出现后它们也都随之而来。

Brian Kernighan参与到工具和工具箱之中，但最初他只是对Thompson实现的管道做了一处修改。他用>替换了^。他对Mahoney说：

> 某种意义上说，管道让Unix成为真正实用的系统。如果没有它你并不是没法做很多这类事情，因为I/O重定向的出现要比管道早不少，虽然也没有特别早，但它确实是比管道要早。我的意思是，它并不是一个新主意。你现在用管道能做的大多数事情之前也是可以做到，只不过没有现在这么方便。我是说，这大概可以类比为用罗马计数法代替阿拉伯计数法。也就是说之前你并不是不能算数，只是会更难，可能在智力上有更多的约束。但现在这一切都被一个中间的箭头符号取代了，甚至在我不知道的情况下就发生了。我记得那个奇怪的>>符号，或者其他人用的什么符号，突然都变成一个竖线，所有事情都开始遵从这条规则。从那时起我可以用这种简洁的方式表达我要做的事情——比如运行**who**命令收集输出内容到一个文件里，通过**wordcount**数一数有多少行就知道有多少用户在线，然后说“看看把**who**连接到**wordcount**是如此的简单”。还可以把**who**连接到**grep**等诸如此类的组合。可以把这些从没有想过的命令放在一起合作，这是如此简单你每次只需要用键盘将它们组合在一起然后就能得到想要的结果。我觉得当我们开始有意识地考虑工具，然后你可以将它们组合在一起，如果你这样做了它们真的就会开始协作。

还有一件事，Doug McIlroy告诉我Thompson将管道符号替换成“|”，“在伦敦的一次谈话中，因为他实在无法忍受我丑陋的语法”。

依靠将程序组合在一起，而不是每个程序自身，Unix从此有了强大的力量。

Unix现在有自己的一套语言。它有自己的气质和哲学。在AT&T贝尔实验室它有一个少部分热心用户组成的小组，但它还没有真正的观众。

> **Unix**哲学
>
> - 一个程序只做一件事，并把事情做好
> - 让程序互相协作
> - 让程序处理文本流，因为它是普适接口

译者注

- The BCPL Cintsys and Cintpos User Guide

1973，第一篇论文

1973年第三版 `UNIX编程手册` 面世了。就像之前说的，大面积的代码都用C重写了。C语言的编译器和调试器也都已经出现。系统被安装到16个工作站上(都在AT&T和西部电子)。Thompson和Ritchie(在Bob Morris，Doug McIlroy，以及Joe Ossanna的“坚定支持”下)向10月将在纽约约克镇高地的IBM Thomas J. Watson研究中心召开的ACM操作系统规范讨论会(SOSP)上提交了摘要报告。

SOSP，就像它的名字所说的，是一个非常大型的会议。Doug McIlroy回忆说他没有想到早上的会议在一个“挤满听众的礼堂”。

Ritchie回忆道，“在一个美丽的秋天开车去科尼克”。宣讲论文的Thompson告诉我：

> 现场有几百名听众，让我非常紧张。人们的反应很普通，只是礼貌性的掌声。我不记得他们提了些什么问题。因为我之前参加过TENEX的讨论(TENEX是DEC-10上流行的操作系统)，正好和我的演讲形成鲜明对比。我花了很多时间讲关于shell和它是如何实现的——试图论证它“只是一个程序”。有一个关于shell的问题但是完全不得要领。可能我讲的不是很好。

Thompson太谦虚了，他的论文获得了巨大的反响。在论文发表后的六个月，安装数量翻了三倍。当论文的修订版本在1974年发表在ACM通讯上后，引起惊人的反响。没人记得TENEX论文(它是7月ACM会议众多演讲中的一个)，ACM通讯的主编评价Thompson和Ritchie的工作成果是“优雅的”。1983年他们因为工作成果获得了业界非常权威的ACM图灵奖。

法律 第一部分

1949年1月14日，杜鲁门政府通过美国司法部反垄断部门对西部电气公司提出申述。位于新泽西地区法院的美国电话电报公司声称对公司的贸易限制违反了谢尔曼反托拉斯法案。经过大量的协商谈判后，在第一届艾森豪威尔政府非常不一样的政治环境里，Thompson F. Meaney法官在1956年1月24日颁布了一个“准许判决”，“任何未经审讯或宣判的现实问题或者相关法律，以及未经这个最终判决任命的任何证明或部门准许...”。

我曾经被建议说，除非政府改变对待商业行为的态度并且考虑“电话公司”，否则案件和判决不会被充分执行。虽然这可能是对的，对于还未出现的Unix来说判决结果以及AT&T和西部电气的律师对它如何解释是非常重要的。

AT&T和西部电气被命令“开始...制造用于销售或租赁的设备(相比于提供电话电报服务)”，还有一些例外，“从事...西部电气或者其附属子公司从事的任何非文字电报业务...”。AT&T被命令“从事除了公共通信服务以外的任何商业活动”，除了一些特殊情况，特殊情况是“出于实验目的的测试或者开发新的公共通信服务”。

大多数的判决涉及到设备制造，购买，零售，核算方式，还有专利授权。1956年早期，还没有计算机和软件的概念。授权许可被提到了，不过没有被写入判决。这说明富有远见是是多么重要。

这里还应当提到的是西部电气是AT&T的全资附属子公司，贝尔实验室是西部电气和AT&T的共同子公司(各占50%)。不管怎样，考虑到Unix接下来几年的命运，读者还是要忍耐着了解一些对于判决对于公司的介绍。

AT&T和西部电气在很长时间里都被政府批准垄断。我们家庭中的电话设备二十年前都还是租赁的，他们属于电话公司所有，作为设备一部分的协议要求不能修改，移除设备。公司和FCC对于设备是什么建立了一个非常宽广的解释——它包含电话簿，和必要的修改(适用于发送信号的塑料设备)，虽然公认法院必要时可以撤销FCC。

根据要求AT&T展示了他们又用搞得专利并且提供信息给(潜在的)竞争对手，判决在某方面来说是一个很大的进步(1956年法务部门成为第一个Unix用户可以证明这一点)。但关键的是这是授权的要求。关于专利最重要的事情他们被授权后就可对全世界公开。但是没有某种介入，尽管全世界都可以读到专利信息，但他们如果没有发明者的授权就不能在17年的保护期内将专利“付诸实践”。在这个案子中，判决条款要求AT&T，西部电气，贝尔实验室以象征性的收费将专利授权给任何人。这样，判决结果在我们的专利法律许可的情况下将技术迅速传播开来。

那个“可以从事除了公共通讯服务外的任何商业活动”也是一件有意思的事情。

Meaney法官，被告，原告都认为公共通信服务是一个公认术语。但是公共通信有着不断变化的法律定义，1934年的通信法案对这个词语的定义是“任何人被雇佣从事公共通信，在洲际或者国际通过有线或无线通信”(47 U.S.C. 153(h))。从这些言辞上来看没有太多的帮助，FCC和法院对此的不同解释更是增加了混乱。

不管什么原因，贝尔实验室的律师非常保守：激怒公正部门这头野兽是没有意义的，除了电话和电报之外没有生意。o

很明显，通用电气和麻省理工学院的“实验研究”是被Meaney法官的判决许可的，计算机研究小组也在很努力的实验。法务部门对于Unix的使用，纽约电话公司，西部电气广泛的政府关系，AT&T和几个贝尔经营的公司也都是合法的。

但Ken Thompson在第四届ACM操作系统规范讨论会(1973年10月15-17日)上做的报告导致站点的一小部分请求完全没有被贝尔实验室，AT&T以及西部电气所理睬。1974年于CACM上发布的改进后的论文引发的巨大反响。

Sandy Fraser是这些感受到Unix的开发需要法律环境做出大量调整的人之一：

> 公司针对这个判决进行了非常大的自我压缩，所以结果就是Unix是一个缺乏吸引力的操作系统。相比计算机，那些公司显然更担心Unix的合法性...
>
> 除了走向大学，Unix没有其他更好的途径。我不得不说，这更像是对不合理的一种回应，而不是从这里赚到钱。

换句话说，贝尔实验室的进一步的活动变成更加危险的公共通信，这被视为违抗判决。Sam Morgan(1967年成为计算研究的总监)补充说：“如果你问我那时候向大学发布Unix是不是个好主意，我会说不是的”。1973年成为贝尔实验室部门主管的Tague需要对支持开发Unix负主要责任！

此外，相对于激怒法庭这头巨龙，AT&T的法务部门决定订阅所需的专利授权，意味着他们不得不也授权Unix。贝尔实验室和学术界有着长久的良好关系，为他们提供暑期实习和学生休假。另外西部电气和AT&T的软件授权给大学所以公司使用了审慎的方式。就像后来成为AT&T软件部经理的Otis Wilson写的：

> 排除任何与判决抵触的争论，AT&T将它的软件在判决令下合法授权，但是必须明确不会将软件用作商业意图。政策在每个渠道一再重申——“这样就没有售后支持，款项预付！”。

1984年冬天Tannenbaum在USENIX/Uniform会议的演讲——更为人知的名字是“第一次华盛顿风暴”——展示了轻微的区别：

> 总是有一个铝簿解释Unix系统的支持政策：
>
> ```
> 没有广告
> 没有支持
> 没有bug修正
> 款项预付
> ```

这个卡片总是会得到热烈掌声和笑声。

从1973年冬天一直到1974年秋天，Unix软件的需求一直在增长。对于创建者以及专利律师，软件一直是以简单的信件协议免费提供(西部电气和加利福尼亚大学管理者之间关于第5版的原始协议有5页，其中两页是标题页和签名页，第5页是“附加定义说明”。它用高中毕业就能看懂的言辞写成，从1973年11月1日开始生效)。后来的授权协议变得更多更复杂。有人告诉我1980年的授权协议排除了许可证的宣传。

就像Otis Wilson所指出的，1970年的专利授权“通常每年有四五次谈判”。职员简直淹没在请求Unix授权中。1974年后当首次有军事和商业方面的授权请求情况变得更糟。

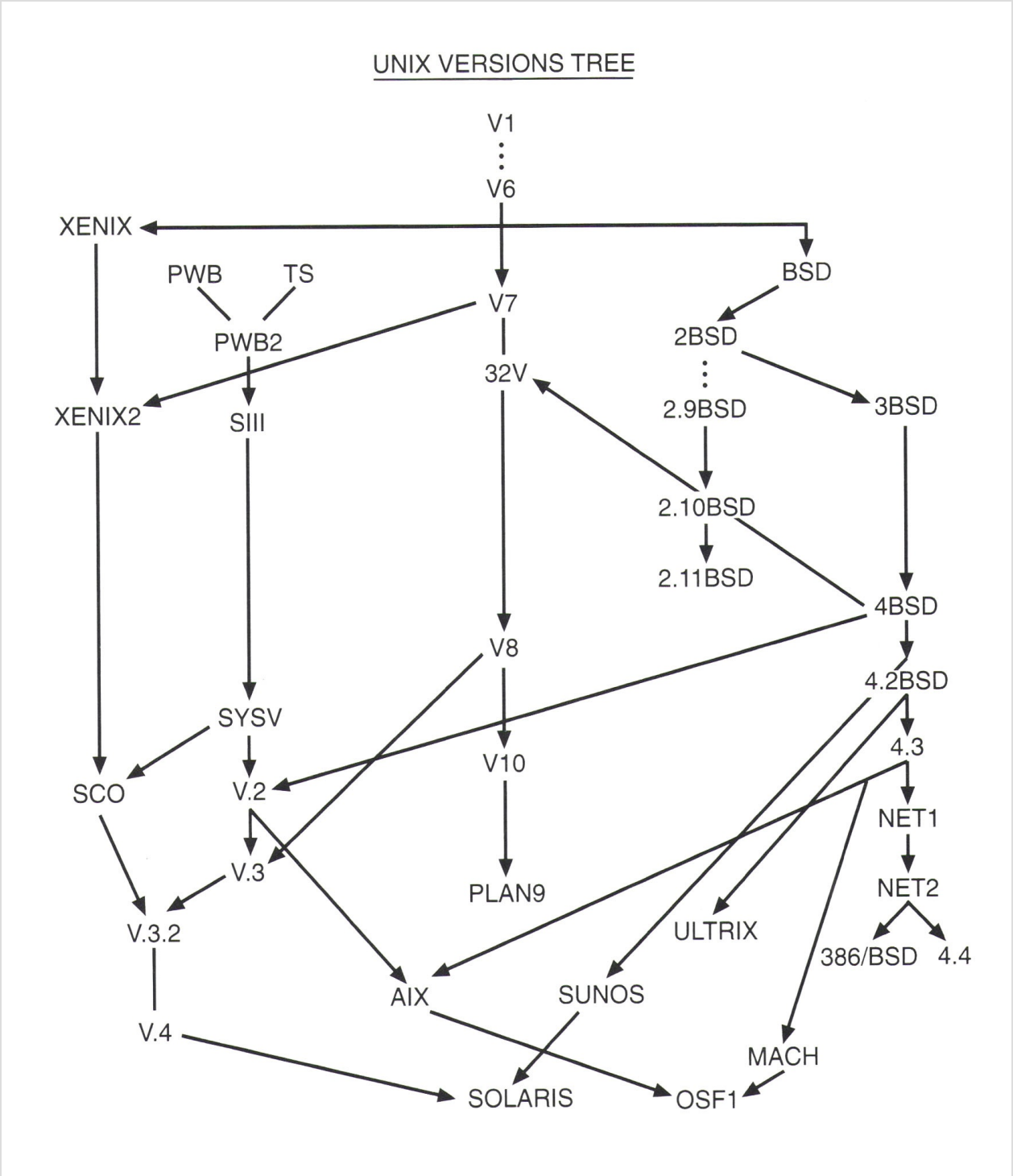
1974年系统的情况

这里看一看1974年Unix提供什么可能会有价值。 通常程序员在一秒钟30个字符的硬拷贝终端上工作()。虽然在今天看来这有些太过缓慢，但是使用像**ed**或**sh**这样的工具相对于提交IBM JCL(任务控制语言)卡片已经是好很多了。

在文本处理被关心的情况下，大多数的安装运行原始的**nroff**，虽然有些小组拥有图形排版机和**troff**。

1974年，很多人开始使用C语言代替汇编进行系统编程。

Unix也提供文件共享和一个支持维护者互相交流的环境，但是PDP-11/45的硬件限制了项目和小组的大小。


UNIX VERSIONS TREE
V1
V6
XENIX
BSD
PWB
TS
V7
PWB2
2BSD
32V
XENIX2
SIII
2.9BSD
3BSD
2.10BSD
2.11BSD
4BSD
V8
4.2BSD
SYSV
4.3
V10
SCO
V.2
NET1
V.3
PLAN9
NET2
V.3.2
ULTRIX
386/BSD
4.4
AIX
SUNOS
V.4
MACH
SOLARIS
OSF1

用户

AT&T的律师决定允许教育机构使用Unix，但是禁止bug修正和后续服务造成一个直接的后果：它强制用户之间互相分享。用户们分享主意，信息，程序，bug修正，以及硬件修理。有些潜在的用户已经听说了Unix，哥伦比亚的生物科学领导者Cyrus Levinthal就是其中一位(Lou Katz对我说“但是他认为Unix是可用的”)，普林斯顿的Richard Langridge是另一位。

但自从1973年发表了第一篇论文后，你可以将程序放在RK05磁盘上。在1974年春天的早些时候，Lou Katz在哥伦比亚大学组织了一次Unix用户会议。哥伦比亚是第一个发行版的使用者——秋天的时候，先是在磁盘上，然后在9轨纸带上(Katz告诉我“Cyrus拿到一个GK05磁盘，但是我们没有驱动器，因此我驾车去莫里山，Ken为我制作了一份9轨纸带”)。会议在1974年5月15日在哥伦比亚大学医学楼3楼的梅里特会议室举办。这就是Reidar Bornholdt提供给我的流程:

UNIX用户会议流程	**1974**年**5**月**15**日 医学楼**3**楼梅里特会议室
10:30	开始
11:30	几个使用者的简短介绍
12:00	午餐
1:00	Ken Thompson演讲
2:00	Unix问题，解决方案，bug修正交流
3:00	DEC问题，解决方案，bug修正交流
4:00	自由讨论

大概有来自十个机构的二十多人参加了会议。Thompson和Ritchie的第一个报告出现在1974年7月的CACM(这次会议的两个月后)。1975年的春天Mel Ferentz(那时在布鲁克林大学)向邮件列表向上的30多个从贝尔实验室收到Unix的用户发送了邀请，会议将在1975年6月18日在纽约大学举办。Thompson向Ferentz提供了近几年收到了RK05磁盘或者纸带的用户列表，一个会议通知发送给这些用户，有20多个Unix新装用户给了回应。6月16日Ferentz向那些Unix新版本用户(第6版)发送了邀请，参会名单修订成37个。那个时候加利福尼亚，布鲁克林，还有纽约大学的家算计中心都安装了Unix。

Ferentz，Katz和Bornholdt在Fuchs的组织下建立了会议。Ferentz自己说(1975年7月30日第1期*UNIX NEWS*，发行量37份)：

> 6月18日在纽约大学举办的会议有来自20多个大学的40多人出席会议。每个大学就简单介绍了它的功能和特点。这里我们不准备再次介绍他们因为接下来有一页详细记录了每一个成员。大家一致同意尽量保持用户组和它通讯邮件为非正式的。

11页的*UNIX NEWS*期刊包含了从波士顿儿童博物馆到多伦多大学的介绍，同时也包含了一篇题为“MUNIX——一个多处理器Unix”的简短文章，另外还有邮件列表。

邮件列表的迷人之处在于它反映了Unix的发展——一个未经宣布，没有支持的操作系统。

第一份邮件列表包含的大学
贝尔实验室
布鲁克林大学
卡尔顿大学
...

Institutions on the First Mailing List

Bell Telephone Laboratories
Brooklyn College
Carleton College
Case Western Reserve University
The Children's Museum
City University of New York
Columbia University
Duke Medical Center
East Brunswick High School
Harvard University
Hebrew University of Jerusalem
Heriot-Watt University
Johns Hopkins University
Knox College
Naval Postgraduate School
Oregon Museum of Science and Industry
Polytechnic Institute of New York
Princeton University
The Rand Corporation
St. Olaf College
Stanford University
The Spence School
Universite Catholique de Louvain
University of Alberta
University of California, Berkeley
University of Manitoba
University of North Carolina
University of Saskatchewan
University of Texas at Dallas
University of Toronto
University of Utah
University of Waterloo
University of Wisconsin

*UNIX NEWS*第二期(1975年10月8日)有60份的发行量，刊载了一条通知:“西部的第一次会议将于10月31日周五在海军研究院举行”。第三期(1976年2月10日)刊载了Robert S. Fabry教授，电子工程和计算机科学部门，加利福尼亚大学，伯克利大学，哈佛科学中心技术服务主管Lewis A. Law的信件，宣告下一次的西部会议在2月27日和28日，东部会议在4月1日和2日。第四期(1976年3月19日)刊载了新南威尔士大学John Lions博士的信件，还有一个有80个成员的邮件列表。到了1976年9月，成员增加到138人。其中13人在加拿大，10人在英国，4人在澳大利亚，以色列和荷兰各有3人，奥地利，比利时，法国，委内瑞拉各有一人，其他的101人则是来自美国。

1977年5月到6月的那期是*UNIX NEWS*的最后一期。从1977年7月开始出版物的名字改成*;login*。AT&T的律师打电话给Mel Ferentz，告诉他说小组(那时还没有名字)不能使用UNIX这个单词，因为他们没有来自西部电气的权限。1978年5月24日到27日在医学大学举办的UNIX用户会议上，在制定一系列组织Unix用户章程的目的下，选举出了一个5人会员会。选出的5个人分别是：

姓名	单位
Mel Ferentz	洛克菲勒大学
Mar Gralia	约翰霍普金斯大学
Lou Katz	哥伦比亚大学
Lew Law	哈佛大学
Peter Weiner	交互系统公司

Law被推选为主席，委员会的名字确定为USENIX委员会(UNIX是贝尔实验室的商标，USENIX不是)

有两行脚注让人满意:他们展示了70年代的一种精神，起源于“我们对抗他们”的态度并结合一种幽默感，Unix在这种精神下团结在一起，并且这种精神今天仍在持续。两次改名使得贝尔实验室的专利代理人Stephen J. Phillips向Ferentz发了封邮件。邮件开头是这样的:

> 1978年6月7月的第六期第三卷让我觉得非常有意思。你选择“USENIX”这个词语从而避免使用UNIX作为委员会的名字是一个非常聪明的想法...

Phillips继续写了一些“淡化UNIX商标”，他希望USENIX在未来不要商业化。Ferentz将这封信发布在*;login:*上。

USENIX这个名字是由Margaret Law创造的，直到最近哈佛和拉德克利夫的一个院系。*;login:*更有意思，Dennis Ritchie解释说:

> 字符“;”是很有意义的。在70年代早期的大多数时候，最流行的终端是37型号的电传打字机。输入序列 `<esc>;` 会让它进入全双工模式，这样终端就不会在本地打印字符，而是让系统显示它们。因此这个序列被放进了问候消息。当然在你使用终端的时候它不会打印出来，但后来出现的其他终端不能识别这个信息所以打印出了“;”。

Bill Shannon和其他几个来自凯斯西储大学的人出席了1978年哥伦比亚的会议。Shannon是Leffler的同事，Leffler参加了下一年在多伦多举办的会议(1979年6月20日到23日)。Leffler说“我们全都去那看Aliens。当我们返回到多伦多的宿舍时，我和Aliens擦肩而过”。早期USENIX会议的休息活动是非常重要的:Leffler和其他几个人回忆他们曾在1980年6月的拉特华州会议时打棒球。这对于Leffler和Unix的未来都是很重要的(我会在讨论伯克利Unix时再说回这个)。

从很早期的Unix用户列表上可以看出来，他们分散在彼此相隔很远的地理位置。虽然如此，下面这些并不让人意外:

> 澳大利亚Unix系统用户组，或者叫AUUG，非正式的存在9年后，在1984年8月27日墨尔本举办的会议上正式成立了。第一个AUUG会议在1975年在新南威尔士大学举办。

新南威尔士大学在1974年成为加利福尼亚西部第一个得到Unix的大学。

英国在1973年收到Unix，之后不久用户就组织了起来。大学之间的计算机会议在1976年9月举办，亚伯丁大学的P.M.D. Gray和他的同事说起Unix在英国不断增长搞得用户量。11月，英国大学*UNIX*通信出版了d第1卷第1册。它包含了一个包含11个安装用户的列表:亚伯丁，达拉谟，埃塞克斯，格拉斯哥，赫尔瓦特(爱丁堡)，肯特(坎特伯雷)，拉夫堡，皇后玛莉学院(伦敦)，威斯特费尔德学院(伦敦)，圣安德鲁，以及苏塞克斯——4个在苏格兰，7个在英格兰。

1977年3月4日，格拉斯哥大学的Alistair C. Kilgour公开了一封邮件，建议成立一个英国Unix用户组作为DECUS特别兴趣小组(SIG)，在4月15日的DECUS英国成员年度会议之后举行相关会议。他提议在1977年5月末尾的时候举办Unix学术报告会。接下来的几期期刊在页首位置印着醒目的*DECUS*英国通讯。DECUS英国支付了副本和邮费。

1977年5月27日在格拉斯哥举办的一场学术报告会议上有40人左右出席。不到一年的时间后，UKUUG联合荷兰北海在阿姆斯特丹自由大学举办了会议。数理中心很早就得到了Unix。Teus Hagen告诉我：

> 我看到CACM上Ken Thompson的文章就立即给他写信了解详细信息，并询问是否有可能得到一份软件。很快我就得收到了他寄来的DEC RK05磁盘，并且没有任何许可限制。那是Unix第5版。我们曾经使用PDP-11/45和RTL或者RSX11为一个图形项目艰苦工作。当我们刚收到Unix就立即扔掉DEC的东西继续使用贝尔实验室的东西。这之后第6版，6.1，6.2，第7版，然后是我们的VAX-11/780(序列号10)陆续到来。我么只订购了VAX硬件，没有购买VMS许可(DEC那时非常惊讶)。我么在VAX上使用了实验室的Unix VAX版本(一定程度上运行在PDP-11模式)，然后换成了伯克利版本(VAX是虚拟地址扩展的缩写)。

NLUUG是在1978年11月成立的。Unix传播到欧洲，另一个国家机构成立了，并且在1981年欧洲Unix系统用户组(EUUG，现在的Eur-Open)有了他们的第一次会议。有趣的是，Unix在日本的出现比澳大利亚晚了两年，因此Unix用户会议要晚于欧洲和澳大利亚。

1976年东京大学的Haruhisa Ishida将Unix介绍到日本后没多久，Jun Murai，Koichi Kishida，Nobuo Saito就成立了日本Unix社团(jus)。最初它是一个Unix学习小组，但它成长的非常快。十年后，Unix真正的被全世界用户使用。

为什么是Unix

让我们在这里先暂停一下，通过一个6年的实验，去考虑为什么Unix如此顺利就能获得全世界学术界的关注。究竟是什么使得它受到这么多人的喜爱。

在最初的CACM论文上，Ritchie和Thompson列出了系统提供的6个特点：

1. 一个可卸载分卷分层级的文件系统
2. 文件，设备和进程间的协同兼容
3. 初始化异步进程的能力
4. 每个用户都可以配置系统命令语言的能力
5. 超过100种的子系统包括十多种语言
6. 高度的可移植性

让我们列一下我喜欢Unix的原因:

- 在很多平台上都可以使用
- 多用户
- 提供目录层级
- 合理的分享计算机资源
- 支持对文件，进程和程序的操作
- 允许进程间和机器间的通信
- 允许访问系统自身的特性

Rudd Canaday将它们简单的总结为:“Unix之所以能走出贝尔实验室是因为大家都喜爱它”。最近，Armando Settner说:“它不区分自己的还是我的方式”。

在1976年和1977年，Tom Lyon使Unix的一些部分可以在普林斯顿大学的IBM 360机器上的VM/360系统上运行。在1977到1978年，Ritchie和Steve Johnson将Unix移植到Interdata 8/32，就在同一时间Richard Miller和他的同事将Unix移植到澳大利亚伍伦贡大学的一台Interdata 7/32上。Ritchie曾经说将Unix移植到Interdata是他最自豪的意见编程工作之一。这说明Unix真正的移植到非DEC制造的机器。我一会还会说到这些。

很容易说Unix有这个或那个的有点，或者它可以运行在一台部门(而不是计算机中心)没有提供的机器上。请允许我，让现在是UUNET科技的副主席的Mike O'Dell，回忆一下1974年的夏天。那时候，O'Dell还是俄克拉荷马州立大学的一名学生。

> 当知名的1974年CACM期刊出版时，我正在为OU计算机中心工作。我们的这台机器叫做ITE，一台断断续续的终端设备，有着世界上最糟糕的BASIC实现，一个伙计写了一些例程让你可以在终端上做I/O操作，这是一个不太重要的功能。所以我们的一个组坐下来试图计算出我们是否能用IBM 2741s作为打孔机做一些像编辑这样有趣的事情。这样你就能编辑任务然后提交它们，或者类似的事情。因此在我最繁忙的时候——我记得我的黑板上画满了数据结构，因为我们将所有事情都放在文件里，因为我们不知道怎样能更好的完成它。
>
> 这时Unix期刊出现了。我记得我走到商场尽头，从邮箱取出它，对自己说:“哦，ACM在操作系统上做了一些事情，或许值得读一读(期刊是‘第4届ACM操作系统规范讨论会论文集’)”。然后我就开始慢慢读它。在期刊里还有其他几个很好的论文。我记得我坐下来然后读到Unix分时系统的论文，这有点像用石头在脑袋上敲了一下。我又重新读了一遍，站起来走出办公是，走到和我做同一工作的George Maybry的位置，把期刊砸在他的桌子上说:“怎么这么多人会错了这么长时间？”。
>
> 然后他说:“你在说什么？”。
>
> 我说:“读这个吧，然后试着告诉我我们做得这些事情不是在犯傻。我么疯了，这才是我们想要的”。就在那时我决定我们无论如何都要得到一份Unix。

实际上O'Dell被Unix如此吸引的原因是俄克拉荷马州电力工程实验室的一个角落里放着一台和PDP-7很相近的PDP-9。O'Dell看到Ritchie和Thompson最初是在PDP-7上完成了Unix的开发，他想在PDP-9上Unix应该也可以跑起来。O'Dell告诉我:

> 我给贝尔实验室打电话，正巧是Dennis Ritchie接的。我说我们没有PDP-11，但是我们有一台PDP-9，我们能不能得到一份Unix的PDP-9版本？Dennis说那并不是一个好主意(他可能在想电话那头的孩子一定是疯掉了)。但他们的态度非常好让我受到了鼓舞。我确信我们我们一定要做这个。
>
> 那时的我是一个无知者无畏的年轻小伙。我打电话给塔尔萨DEC的销售办公室(他们在俄克拉荷马州没有销售办公室)要求找人聊一下PDP-11。最终我和一个叫Stan Bartel的家伙聊上了。
>
> Stan是一个有意思的家伙，他意识到自己永远不能将小型计算机卖到俄克拉荷马州立大学，因为到处都是蓝色的死机屏幕，去计算机中心找一份工作至少会比现在会好一点。所以他并不介意DEC支付的是薪水而不是佣金。他将要我们这附近带上几个星期。不管怎么说，他远道而来，吃了一顿午饭，带来了他所有的手册。处理器手册和外围设备手册。我从没有看过这些文档。那时我是一个不错的程序员，我看着这些手册学习指令集，我说“这些家伙知道这是怎么回事，这太酷了”。故事就这样开始，我被吸引住了。

就如我之前提到的，C语言编程直到1978年才出版。之前在俄克拉荷马发生了什么？O'Dell在1974年到1978年之间的故事和其他很多人都很像，就让他继续说吧:

> 我第实际上是在伊利诺斯州立大学第一次接触到C语言。我和Steve Bunch从俄克拉荷马州立大学去伊利诺斯州立大学访问Mecca，在那里我们见到了Unix。Steve Bunch，Steve Holmgren，还有Mike Mullen这些家伙组成成的小组将Unix发布到了ARPANET。总之，Steve Bunch发给我一份智能终端的内核列表，里面包含了上下文切换，信号量控制，进程间通信，还有一两段代码。因此我们有了五六页论文，他还写了一些无关紧要的笔记，因为我从来没有看到那个编程语言，他解释像**++**这些有趣的操作符是怎么回事。他还说“看看你能不能研究出这是怎么工作的”。我记得上下文切换是用6行汇编实现的，他将相关代码发给我，看看我能不能搞懂。然后我就搞明白了。
>
> 后来我想知道更多相关的资料。于是他给了我**tp**(在**tar**出现很久之前)，老式**tp**格式的帮助信息存储在nroff输出的纸带上。他将纸带寄给我，说:“这上面的东西很有些意思，如果你能解决怎么打印它”。然后他发给我真正的**tp**帮助页。
>
> 我在一个长长的周末坐下来写程序破解这个格式，一点点推敲，最后打印出Unix帮助页。那时我的第一个帮助页。它们现在应该还在我的某个盒子里，因为我当初得到它们是如此困难。
>
> 然后我就不停翻看这些帮助页。虽然我们不能得到系统代码，但我们可以看看它究竟可以做些什么。

靠自己努力解决问题。你认为它会如何工作？Mike并不孤单。RAND编辑器的作者Dave Yost告诉我他是如何在1970年看到代码的:

> 我坐下来仔细看着，看到这些卷曲的括号，**[{**和**}]**，“哇噻”!

在C语言里，**{**和**}**将函数的声明围起来。C没有像PL/I的**DO-END**或者Pascal和Algol的**begin-dend**。C在很多方面看都是一个简单的语言，和自然语言非常相似(比如英语)。它不是一个“高等级语言”。就像Thompson和Ritchie说的:“C语言是一个通用语言，有着简单表达式，现代的流程控制和数据结构，还有一套丰富的操作符...我们认为C语言被证明是一个令人舒适舒适，富有表达力，多功能的编程语言”。根据我的经验，是C语言让编程成为一种优雅的艺术。

1987年，Ritchie在USENIX技术会议上分享了他关于Unix成功的思考。他提到了“简单，一致，高效的计算”，还有“工具箱的概念”，以及“可移植性”。

第一个和最后一个是非常重要的:他能在不同的平台上运行。Unix和C语言的优雅对流行也至关重要。当然还有风格和工具。

译者注

- The UNIX Time-Sharing System

风格和工具

时间回到Ritchie和Thompson为他们为ACM讨论会准备论文的时候。Brian Kernighan和P.J. Plauger正在合作写编程风格的要素(Elements of Programming)。Kernighan说:“我们从1973年开始写，在74年早期结束”。Plauger强调:

> Brian Kernighan和我在莫里山贝尔实验室的隔壁办公室几乎以事故收场...
>
> 我们怜悯计算机编程的糟糕状态，以商业创作结束。我们都是第一次写作，这一本小书花掉我们差不多4个月的时间。据我所知，它让“编程风格”作为成人间合法的讨论话题出现在书上。
>
> 当然，这本书现在已经过时的可怕了...

Kernighan告诉Peter Collinson:

> Plauger和我写那本书，过程很有趣，结果也还很不错。那本书背后的思想是列出很多代码然后评论它们:那样做不对，应该有更好的做法。我们不说应该怎么做，而只是说应该不去做什么。我们的目标语言是那时社区的主要语言FORTRAN和PL/I。

编程风格引起了很大的轰动。但是不管是Plauger还是Kernighan都强调它只涉及了FORTRAN和PL/I。他们两人合作的下一件工作有着更加深远的影响。Collinson问Kernighan他是怎么开始写软件工具(Software Tools)的。他回答说:“这很难说”。

> 几年以后，我们决定是时候去告诉人们一点东西，让他们知道他们应该怎么做事情。那时候，我们清楚的知道一些Unix环境的好处:管道可以让程序在一起协同工作，还可以加上过滤器。还没有一本书清楚了讲解怎么用一个编程语言实现这些。当然，C语言在很多环境里是不存在的。我已经完成了Ratfor(Rational FORTRAN)。Ratfor简单的偷窃了C语言好的一面，但是除此之外并没有添加太多东西。它还算不错，将FORTRAN变成一个编程语言。
>
> 我们决定用Ratfor作为那本书的编程语言实在是违反传统，因为它只有很少的用户。小组里没有一个人，因为那时没人用FORTRAN语言，除了数值分析员。但他们担心可移植性所以他们不想用Ratfor语言写出无法月的的FORTRAN代码。
>
> 最初的版本使用C语言用一些yacc的语法写的(**yacc**是“yet another compiler-compiler”的简写，由Steve Johnson编写)。给出自举，它就能花费一些时间用Ratfor语言写出来。之后它能在一个只运行FROTRAN的机器上运行。为了软件工具包分发的一部分程序是用Ratfor实现的Ratfor，还有一部分是用FORTRAN实现的。纸带上开头就是FORTRAN实现的Ratfor，因此你可以简单的拿出纸带开始运行。

Collinson接着问:“那么你有了Ratfor，你是怎么从它们又做到软件包的”?

> 你坐在哪里，然后问Unix系统中有哪些有意思的事情。事实上有很多的小工具你可以将它们用有意思的方式组合在一起工作。书的前三章内容会让你看到:字符输入输出是最通用途径，文本就是一切，数据就是一行一行的流式文本，也许归档也可以算是。
>
> 然后我们介绍了正则表达式。这些是Unix中非常重要的基本概念。有了这个，接下来的事情就是模式搜索，**grep**家族，还有编辑器。事实上，排序也可以放在这些工具里面，关于算法有很多有趣的算法可以说。
>
> 我们做的另一件事情是文本格式，一次我们做了一个简单的格式化工具。最后，我们讲解了Ratfor，说这是你可以一直用下去的程序。这是用Ratfor实现的Ratfor。

软件工具这本书附带一个Ratfor纸带，Ratfor是为这本书而写。但书实现出版的，Kernighan评论说:

> Ratfor是这本书的基础，但它成了这本书的媒介。书上描述的版本比之带上的版本要简单，因为将它的实现全部解释给你也不会学到更多新的东西。有一些像软件工具小组之类的小组开始出现，主要是在加利福尼亚的劳伦斯伯克利实验室。Debbie Scherrer，Joe Sventek，Dennis Hall还有其他很多人建立起了软件工具小组并且做了很多很棒的工作，比如建立“虚拟操作系统”以及很多其他各种事情。用干净利落的方式在多个平台上使用了优雅的技巧。

再回忆一下Unix的哲学:写一个程序，只做一件事并把事情做好。这就是这些工具背后的思想。并且这句话也激发了劳伦斯实验室的人们。

Debbie Scherrer告诉我:

> 如你所知我最初是在太平洋大学。但是他们只有一个计算机课程，所以我转学到了伯克利。大概在1968年我为我的硕士学位工作的时候，在贝尔实验室有一个工作机会，那时一个很棒的工作。Dennis Hall和Joe Sventek就在楼下，他们在另一个部门工作。他的学位是物理学位，并且做的是像武器测试这样的保密工作...我们给这些研究员维护系统，他们只要计算机工作就好，对计算机本书毫不关心。但是有一天来自阿姆斯特丹自由大学的Andy Tanenbaum在实验室的桌子上留下一本复印的软件工具，他说“你可能会对这个感兴趣”。所以我开始阅读，觉得这本书简直美妙无比。同时Andy Tanenbaum提到这本书，实验室的Dennis Hall发现了它。Dennis将我拉了进来(那时我在为另一个项目工作)，Dennis是一切的起点，是他在我和Joe能接受的范围内指导我们。从那个周末开始我就坐下来开始完成整本书。这太棒了！Dennis，Joe和我都觉得，如果我们支持研究，那么这些工具也支持研究。我们最后完成了所有剩余部分。喔，还没说完。DOE的Jim Poolet一直非常支持我们，给我们资助。没有人曾经这么做过，独自用可怕的FORTRAN程序，扩展的DCL脚本，还有其他可笑的东西。

它就是那么快。

Mike O'Dell告诉我:“记着，Debbie知道Brian，他知道他们在做什么，所以他指出人们在向贝尔实验室询问什么工具。这些事情慢慢形成了用户组”。说起来，软件工具不是一本关于Unix的书，而是关于哲学和风格。Ritchie说:

> 工具的使用简单而强大，但是也需要一些想象力。相比起将更简单，更普通的工具组合在一起，建立一个专用的工具可能需要更加高昂的代价。

有趣的是，Scherrer和他的同事们意识到工具想法是如此强大，他们清楚的知道软件工具的方式会在所有平台上流行起来。所以Hall，Scherrer还有Svented写了一个虚拟操作系统(VOS)，它能在软件工具(Ratfor)和任意的操作系统之间提供一个虚拟的接口。Scherrer谦虚的说:“VOS是Dennis的主意”。

我问Paula Hawthrn，关于软件同居，在贝尔实验室谁有成为O'Dell的主管的声誉。她说:

> 当我加入到贝尔实验室的计算机科学与数学部门，Debbie，Joe还有Dennis Hall已经完成软件工具的几个版本，但还困扰于“但这真的是科研吗”这个问题。
>
> 那个问题是因为贝尔实验室本应该是做科研的，我们能说发布软件工具就是在做研究吗？“但这真的是科研吗”，是真正想做为众人所用的研究的工程师的毒药，因为很多做出成果的事情不需要研究，但是科研如果没有这种基础根本就无从谈起。所以我那时的记忆完全充斥着和“这真的是科研吗”的纠缠，还有“如果你想用Unix，为什么不直接去用Unix”的对抗。大致上，软件工具似乎成了掩盖非Unix系统差异，使他们看起来都和Unix一样。这同样引起很多热烈的讨论...

1978年成立了一个软件工具用户组，用于和一些人讨论工具的问题。1980年9月VOS的文章发布在CACM上，不过一个STUG会议已经于6月16日，在瓦乌布日赫大学先于USENIX会议举办。Wally Wedel在会上报告了LBL主办的非定期通讯杂志软件工具社区(1980年10月4日)。在几个DEC操作系统，数据控制网络，TSO环境下的IBM系统，还有MPX 1.3(MPX是一个早期的操作系统，NASA的Ames研究中心虽然能够让每一个工具在它上面工作，却没能实现一个shell)上的SEL 32/77上软件工具都已经可以使用。在1981年9月的那期杂志上，Phil Scherrer(以前的独角兽系统个，现在在斯坦福大学)写文章说:

> 软件工具现在已经完全移植到一个微型计算机环境。运行在8080，Z80(zilog的商标)，8085处理器上的CP/M(数字研究的商标)操作系统因为可以在众多硬件上运行，被选为目标系统。

那一期的软件工具社区列出了接近30个已经实现工具的硬件架构和移植者的地址。内容从Burroughs到Zilog的制造商(字母排序)，从Z80/8080(64Kb内存)到IBM 730和DEC 20的机器(大小排序)。地理上看移植者来自全球各地:从日本的川崎到美国，到美国，到加拿大，到英国，再到荷兰的埃因霍温。

工具变得重要的站点之一是佐治亚州科技。在那里呆了很长时间(虽然不是在最开始)的Gene Spafford告诉我:

> 1970年的中后期，佐治亚州科技的人们有几个运行Unix的PDP11。这些机器既被用作科研，也是医学数据库研究程序的一部分。
>
> 几个实习生和员工们都被Unix迷住了，他们想把它放在部门的主要科研计算机Prime上。Prime在那时是很好的机器，它有虚拟内存，多用户环境，还有许多很好的特性。不幸的是，Prime的硬件架构使得在上面写一个操作系统并不容易，Unix并没有被移植到这台机器。

作为一个附加诱因，最后一台PDP-11被一个DEC驻场工程师“熔掉了”，他弄颠倒了电源，将110伏的电流接到了背板上。DEC再也没有让它恢复正常，所以我们没了Unix...

实习生们写了一个系统，但是佐治亚州科技不允许这个系统分发出去，所以它只授权给为它付了足够费用的大学和公司。同一时期，它在许多地方都在运行，Prime甚至订购了它。几年以后出现的三个事情杀死了它:

- Prime的一些员工不希望顾客更喜欢软件工具接口，而不是他们自己的Primos，所以他们停止想软件工具小组提供特殊的支持。Prime工程师中的大多数拥护者都因此离开去组建了阿波罗。留下的人们返回到公司的开发团队，将一些软件工具的点子做到Primos里。
- 软件工具小组为系统写了一个非常棒的C语言编译器和运行库。它们想将所有的事情切换到C语言上，并且建立一个独立的公司来提供支持。不幸的是，佐治亚州科技的管理层贪图编译器和相关代码的所有权。结果是整个的团队退出，软件工具事实上没有了后续的支持。
- VAX开始购销售大量的BSD Unix。之前在Preime上使用软件工具的人们现在可以用同样或者更少的花费用上Unix和虚拟内存。需求开始快速的下降。

主要的相关人员有Dan Forsyth，Paul manno，Perry Flinn，Allen Akin，Win Strickland。

在Spafford的建议下我去见了Manno和Forsyth。Forsyth在和前同事们为了准确性挤在一起后，告诉我:

> 佐治亚州信息与计算机科学技术学校的软件工具项目是以1976年几个独立的事件开始的。第一就是软件工具的出版。第二是军队对可移植COBOL的兴趣。最后，也是最贴近生活的，一个资深小组的困境，他们从暑假回来去找他们心爱的Burroughts B5500，但它因为缺少维护资金而损毁，并且所有的PDP-11都因为要保护医学研究数据而隔离起来。在他们原来田园式计算机环境的地方，使用全新的Prime 400建立了大型的军队研究项目。
>
> 移植一个Unix会出于很多原因，包括不能访问PDP-11，最初Prime内部缺乏文档，还有人们缺少兴趣。那时本地编程环境包括一台看起来介于GE-645和Intel 80286之间的机器，一个FORTRAN-66编译器，一个最好可以接收两个6字符文件名和10进制数字命令的分时操作系统。

> Philip H. Enslow博士(军队科研项目的灵魂人物，马上成为许多学生的指导教授)建议的专业设计项目之一是移植Ratfor到Prime。很明显在我们了解之前Job Brian(Kernighan)和Bill(Plauger)就将有挑战的工作完成了。纸带从Addison-Wesley那里可以得到，在我们知道以前Ratfor就已经在运行了。
>
> Allen Akin和Jack Waugh开始意识到工具应该怎么被放在一起使用。1977年早期的某些时候，Primos的命令行接口被认为不能优雅的将工具连接起来，因此一个简单的类Unix风格的shell被匆匆拼凑出来，不久之后Paul Manno和我就这份工作吸引住了。
>
> 虽然我们都强烈渴望着建立一个优雅的计算机环境，但是我们两人都在为其他项目工作。虽然如此，我们经常发现相比于每次都从头开始，我们在K&P的建议下扩展或增加新的工具可以让我们更有效的完成项目。在这种激励下，软件工具子系统开始迅速的成长。由于它是一个非常有效的环境，来自其他学生和员工的贡献也开始成长。我们接收到很多其他研究员的贡献，然后发现为了避免花费很多时间回答问题，需要建立文档标准并且写很多文档。当然，我们也扩展了文本格式化程序和为了完成它所需的其他程序。
>
> 到1978年的夏天，我们有了一个Primos命令行和编程环境的有效替代品。另外对于最初的软件工具，我们有了一个巨大的子例程库，basic命令行，扩展的Ratfor，电子邮件，公告板，全屏编辑器，以及许多其他的新工具。
>
> 通过Prime研究部门的领导Phil Enslow和David Nelson的努力，Prime计算机研究部门也对我们的工作成果有兴趣。在佐治亚州科技和Prime的帮助下，Allen，Perry和我在夏天去Prime的科研中心并且得到一个“高级”命令解释器的灵感。我们将Multics，UNIX V6，RDOS，还有其他系统的特点集成到一个单独的命令语言。软件工具，Unix以及Algol 68这些例子让我们确信我们应该以优雅，正交，可重用为目标。
>
> 1979年初，我们做出了“新命令行”。现在想起来，它看起起来非常像Bash(虽然那时我们从来没有见过它)。它包含几个有趣的概念:使用语法将多个标准输入和输出直接与图形工具连接，任意长度的命令行，通过外部命令控制流程，还有被视为对象的局部变量，它们可以在标准输出上打印他们的值。
>
> Prime科研中心获得了一份这个子系统，就像它被亲切称呼的名字，因为它可以提高文本处理和研究程序，他们在内部使用了它。它可能在这段时间让我们和Prime的工程小组建立了不太令人满意的关系。看起来我们有能力做出和发布这些提高操作系统接口的特性，几乎就在同时Prime工程团队声明这样的任务是不可能做出来的。当然，我们奢侈的拥有一个不大的用户群，并且我们没有保证向后兼容。不管是什么原因，我们的工作间接的影响了Primos后续发布的版本。
>
> 随着获得更多的Prime系统和一种叫做“以太网”的连接中断的方式，软件工具子系统成为ICS学校的行政管理工具。它就像为部门处理文本一样在本地处理电子邮件，它没有宣传就获得了包括我在内众多毕业生的喜爱。相比于将我们的成果免费发布给公众，佐治亚州科技的当权者相信收取一点授权费更为合理。虽然授权费用高达3000美金，但全球仍然有几十个Prime工作站购买了授权，包括英国，法国和澳大利亚。

> 1980年劳伦斯伯克利实验室的Debbie Scherrer邀请我们出席一个软件工具用户组会议，这个会议是多伦多USENIX会议的一部分。佐治亚州科技帮助我们飞到多伦多，在那里我们发现我们被许多新主意包围着。这些会议相当的有趣，让我们看到了在我们孤立的部门环境下无法想象的观点。我们被允许贡献一些我们的软件给用户组，但是因为授权的原因，我们永远不能发布“新命令行”和更多有趣的工具。
>
> 在1981年Allen，Perry和我离开的时候，子系统已经像那几千页重写的文档一样，完全重写了Ratfor预处理器，一个解析生成器，一个语言独立的代码生成器，还有一个C语言编译器(用Ratfor写的!)。项目接下来由Jeanette Myers，Terry Countryman，Peter Man，Scott Lee，还有Arnold Robbins接手，他们做的都非常好。在那期间，AT&T开始提供低价的Unix二进制授权，不到一年，Unix盒子的只需Prime或者PDP-11价格的五分之一。不久，UNix，vi，还有troff就可以被大众使用，商业对子系统的需求开始减弱。
>
> 在佐治亚州科技周围，BSB4.1 Unix VAXen，AT&T 3Bs，还有Sun 3s开始像野草一样萌芽，让计算中心惊讶不已。CSNet，NSFnet，还有USENET让Unix用户们需要参与进不断增长的互联网社区。最后一个子系统在1985年发布，它的代码被发布给公众。不久之后，它和Prime退出市场，被那些弥补了其缺点的软件所代替。

软件工具引发了另一个平民运动。一部分热心用户将它叠加到系统来使用Unix。但是其哲学是可以移植的:写一个程序，只做一件事情，并且把事情做好。

STUG中的许多名字都被Unix用户所熟悉:Neil Groundwater，Mike O'Dell，Scherrer，Joe Svented，Dave Stoffel，还有Wally Wedel。

我上次说到O'Dell时他还是俄赫拉荷马州立大学的一个学生，现在他已经完成了他的硕士学位论文(“不久前我有看了一下它”，他告诉我，“它并不是难的可怕”),并且在找工作。用他的话说:

> 我在莫里山用了两个星期和1127(科研小组的代号)谈工作的事情，接受了面试。然后开始了死亡历程，那里有无穷无尽的书面工作，每天桌子上的文件就像你刚刚去度了两个星期的假期。因此一个需要两周的批准流程...变成两个月。所以在那段时间我去了博尔德举办的“USENIX”一月会议，参加了软件工具会。我被软件工具可以在IBM机器上运行迷住了，那时我们仍然在努力说服而克拉玛州立大学购买一台DEC机器。所以我去软件工具分场，Debbie Scherrer讲了他们杰出的工作，演讲的最后他说:“哦，另外说一句，我们正在在寻找为我们运行Unix系统的人”。我几乎是爬过众人，我清楚记得路上扔过来的每一把椅子，会议不得不中断。那里一共有来自实验室的6个人，Joe Svented，Peter Krebs，Debbie，Dennis Hall，Roland johnson可能也在那。面试程序是和6个人一起吃午饭，他们在你说话的时候吃中国菜。
>
> 不管怎么说，我讲了一些我的论文，他们了给我一个offer。因此我要在LBL和贝尔实验室之间选择一个。贝尔实验室一再的延期，最后LBL说他们必须要有一个结果。因此我打电话给Dennis Ritchie...他说我应该选择另一个工作。

所以O'Dell去了LBL，在那里他成了Unix领袖，ARPANET联络员(11/70是一个早期的ARPANET主机)，和其他一些东西。Neil Groundwater也在那个博尔德会议，同样的他也受到了STUG的影响:

> 1980年的博尔德USENIX会议上我成了STUG的一员。那时看起来LBL(Sventek，Scherrer等)已经完成了90%的工具工作，其他的工具也开始着手。
>
> 我们被卷入美军的几个项目咨询，他们有不同的编程环境。我们建议将工具作为他们在多个平台上控制编译环境的一种形式，事后证明了他们主要将它使用在VAX/VMS上。
>
> 我最大的贡献是一个针对版本控制文本文件执行**create**，**delete**，**get**和**edit**功能的文本控制系统(SCCS)。它是基于贝尔实验室J.W. Hunt和Doug McIlroy发布在贝尔实验室技术备忘录上的的**diff**算法。

STUG是在博尔德会议上正式成立的，Groundwater被选为第一个委员会。O'Dell去了伯克利。

> 我在实验室的三年都是和Debbie分享一个办公室。她是世界上最可爱的人，拥有6个人才能拥有的能力。软件工具达到很高的水平。不好的是Unix社区从来没有真正的学习软件工具的人们给出的课程。软件工具的员工，我坚持认为他们对可移植性的理解无人能及，UNix社区这方面就太弱了。Unix的问题是它太容易去写代码，你找不到理由不去写。这不是救人于水火的方式。许多软件工具工作的更好因为他们真的做对了事情，5700个人不是这去重新发明一些事情。
>
> 工具的概念在Unix意外的领域开始繁荣昌盛。

The picture of Rat-4, drawn by George Kapus, is courtesy of Debbie Scherrer.

译者注

- Plauger
- Ratfor
- Software Tools

- The Elements of Programming Style

PWB和MERT

回忆Thompson和Ritchie为PDP-7写的第一个系统，第一版是在1971年为PDP-11/20写的。第四版是贝尔实验室于1973年末在第三版的基础上用C语言重写的，当它使另一个项目获得成长:程序员的工作台(PWB)。它的第一版帮助Charlie Roberts思考。Bob Morris说Rudd Canaday从来没有得到他提供的贷款。对Roberts来说也是这样。MERT(多环境实时)是Roberts的发明，虽然它的大部分是由Heinz Lycklama和Doug Bayer写的。Roberts告诉我:

> 1970年早期，我知道在PDP-11上有一个Unix，它是用B语言实现的。但在我看来Steve Johnson已经在谈论C语言。在1970年10月我离开莫里山计算机中心回到科研。我作为客户已经意识到专利，并且我回顾思考1127将Unix视为分时系统来为文档做准备。计算机科学研究就像很多的学术部门:它有两部分。Sam Morgan领导了包含有Dennis，Ken，还有Doug McIlroy的计算机科学那部分。Hank McDonald则是系统计算研究部门的总监。我在哪里为Hank工作，他想为80年代建立下一代的ESS。
>
> 我们在15楼，正好在Doug和其他伙计们的下面。我有Heinz Lycklama，Dick House以及Carl Christensen为我工作，还有其他一些程序员。Carl和Heinz那时工作于数字数据处理器DDP516，那是一个小的计算机，它有自己的操作系统和硬件。但我曾经读了Per Brinch Hansen的一些文章。他后来把所有的内容都卸载了操作系统原理 *1973*年这本书里，他让我思考像微内核，操作系统层还有其他很多直到Mach 3和Chorus以及Windows NT才真正成熟的东西。在我看来Unix作为一个分时系统有些太过于聚焦，我想使用一个专注的自定义系统工作——为交换和大数据库，实时和交易处理。
>
> 好了，71年的时候Hank帮我搞到买大容量磁盘11/45机器的钱，接下来几个月我都在为我的系统工作，它的名字叫OSKER——**OS KERnel**。在10月和圣诞节间的某个时间，我引诱Heinz Lycklama离开了DDP。他和Carl看不到我想实现的概念。但Hank孤立Heinz来和我一起工作并且在72年晚期的时候我们得到足够的钱去雇佣一些人，Doug Bayer在73年的时候加入我们，Heinz真的将OSKER开发出来。在那个时候，我和部门的头Doug McIlroy又一次谈话，他建议联合起来合作。但是我看不到可能性并且担心这会让我的员工放弃OSKER去为Unix工作。那时Heinz建议我们“将Unix构建在OSKER上面”，然后那就成了MERT。我们同时有了Unix和实时处理，这很了不起。但是Heinz对内核失去了兴趣，Doug是一个新手。我感到巨大的压力，1973年我决定搬到霍姆德尔将工作更多的放在通信和图像处理上。
>
> 当我离开的时候，Heinz开始开发Unix MERT:那是系统实验室74年的工作内容，他们从来没有进行过内核方面的工作。MERT在Berk Tague组建USG的时候搬到了哥伦布和印第安山。印第安山在78年创建了多路复用MERT——DMERT，它不是一个可移植的微内核，非常依赖硬件。它运行在3B20D上，但它今天仍然运行在每个4ESS和5ESS上。

实用工具

每个Unix机器上都安装着数目繁多的程序，不管是系统原生的还是厂商附加的。这其中包括编辑器，格式器，编译器。实用工具使得用户有一个字母列表跟踪程序或者文档的不同版本，去“伸手戳一下”其他用户。这章将用来讲述它们中的一部分:非常早期的——**troff**，**eqn**，**tbl**，**yacc**，**mail**，**make**，**awk**，还有最近出现的——UUCP，TCP/IP。最后两个又引发了其他的实用工具(Mike Muuss的ping和Earl Cohen的finger就是很好的例子)，并且几乎立即就出现了一个世界范围的由各种不同用户组成的社区。这个社区对Unix来时及其重要。各种问题，解决方案，程序，抱怨，宣告等等都在社区里自由的交换传送。整个社区建立在大约3000万到3500万的网络用户之上——就像John Quarterman所说的，比加拿大的总人口都要多。

mail和roff以及ed一样，在Unix第一版时就存在了。Morris曾经将roff移植到635上，McIlroy用BCPL语言将它重写。Thompson将它移植到了Unix。但Joe Ossanna不久又将roff合并到了troff，一个真正针对电传打字机文档的格式器。Ossanna死于1977年11月28，McIlroy告诉我:

> Joe是军需官，他游走于各个厂商，打探硬件的最新发展，和设备部门做交易，还做了很多很多其他的事。他是一个严厉而有吸引力的人。

eqn在第5版的时候加入进来。Brian Kernighan说这是他对Unix做出的第一个重大贡献。就像他向Collinson说的:

> 那个时候所有的程序都是用汇编语言写成的。troff的最早版本应该是在1972年晚期或者1973年写的，它是一个汇编语言程序并且只针对特定的图形排字机系统。在今天仍然有许多事情依然受到troff语言那古老机器语法的影响。
>
> Joe Ossanna将它转换成C语言，我认为他受到了相当大的影响，他并不真的想将它转为C语言。他死于1977年，之后我接手了它的工作，事实上，虽然我花费了至少一年半的时间试着去编译它。我开始让它独立于图形排版系因为那时它的优势已经很明显了。我开始投入到图形中因为新型的默根瑟勒排版机可以做一些受限的图形操作，你可以非常精细的定位，你可以画直线和曲线。
>
> Mike Lesk大概在1975年做了**tbl**，在Lorinda Cherry和我做出**eqn**不久之后(Cherry告诉我他开始着手项目的时候Kernighan就加入了，但Kernighan因为非常成功的语法而出名)。**eqn**是第一个文档预处理器。它是我们强行实现的一个好主意，需求是发明之母。**troff**几乎用尽了PDP-11/40s的所有地址空间，它用汇编语言写成。我从来没有学过PDP-11s的汇编，我想Honeywell是最后一个我知道汇编语言的机器。
>
> Joe独自保持维护**troff**，他对于修改它使它工作有着难以置信的兴趣。但这是他的程序，他不想让其他人参加进来。这些事情加起来意味着Lorinda和我不得不去做一个分割程序。
>
> 几个事情合作的非常好。C语言变得非常具有可用性，yacc分析生成器刚刚出现，管道也刚被发明出来。所有这些事情一起使商用变得可能。

Kernighan和Cherry实现的“一个数学排版系统”发表在1975年3月的CACM上。1976年9月14日，M.E.Lesk实现的的“Tbl——一个格式化表格”发表在贝尔实验室计算机科学技术报告第49期上。和troff一起，这些预处理程序使得Unix用户可以排版各种各样的复杂文档。精心制作的宏包，增强了的**-mm**(memorandum)宏极大地扩展了功能。Mike Lesk写了**-ms**宏，Eric Allman写了**-me**宏。只有**-ms**出现在Unix的研究版本，**-mm**出现在第三第四版本。**-me**和**-ms**出现在BSD(Lesk还写了**refer**，可以使用引用数据库提供引用和脚注)。

yacc(yet another compiler-compiler，有一个编译器的编译器)，是Steve Johnson的作品。他在Unix出现之前很久就已经在贝尔实验室工作了:

> 我在1963年有了第一份工作。我总是对计算机感兴趣，当我拿到学士学位，我试着找一个研究生学校。只有很少的地方我可以去，这些地方大部分都属于电力工程部门，在那里我不得不学习发电站的建设以及其他那些我完全不感兴趣的课程，所以我决定去学数学。因此我在63年夏天去了贝尔实验室，Tom Growley是我的第一个老板...那里有一个基本的规矩，你不能连续两个夏天都在同一个组工作。因此下一个夏天我去了ALPAC的数学组。再之后的那个夏天，我依照流程去了人类信息处理组，那里有声音，语言，心理方面的人们设计实验和进行数据分析。他们需要做一些多维度的工作，这使得聚类成为我的第一篇论文，它在我所有的出版物种有着最多的引用——一共有700之多。

> 当我完成我的论文，我去贝尔实验室。我可以选择我将要去到哪个组，最后我选了人类信息处理组，因为他们会用一些我很感兴趣的计算机音乐。我在那里做了很多事情。也在那个时候，Multics被认为是繁重计算工作的解决方案，但它是一场灾难。基本上，我们购买了7094并把它运送到印第安山实验室，并且除了645上的GECOS没有其他系统可用。GECOS可以支持1000个用户但是1个就会很勉强(McIlroy向我指出645“运行了大量的批处理”)。
>
> 因此，到了下一年，在极端努力之下这台几百万美元的机器可以运行FORTRAN程序了，我同意帮助他们的工作去介绍分时系统。因此在1969年我实际上在Comp中心和一个小组一起工作。在这些时间里，Comp中心分化出了计算机科学研究实验室。然后有了办事员，计算机操作员，打卡员等等这些向研究部门汇报的员工。我到那里一年之后，他们才姗姗来迟地设立了一个组织运营计算机中心。我决定宁愿留在研究中心，那样有个后门我可以去做Unix的组。
>
> 我最早是和Dennis Ritchie一起工作于另一个叫做Altran的计算机代数系统。Dennis写了Altran语言的编译器。它是一个用FORTRAN写的递归下降编译器，这绝对是一件杰作，就像用牙签建造了一幢摩天大楼。他曾经为B语言写过一个运行在Honeywell系统上的编译器，我们用它写一些系统程序。但当他和Ken开始为Unix工作时，他终止了他的B语言编译器，然后我领取了它。这个语言里面有一些我想要的东西，比如我添加了高级OR操作符。我从和Dennis一起工作中学到了很多编译器的知识，并且这是一个非常体面的工作。我变得相信你可以通过阅读其他人的代码学到非常多的东西。阅读那些杰出的名著。
>
> 当我想添加这个的时候，我向Al Aho请教，因为我听说他对处理表达式的新方法很有兴趣。这是一种非常有趣的情景。Al不停的点头说:“是的，我读了Knuth的这篇论文，这是一个非常好的方法”。所以我们用B语言做出一个非常简单的表达式语法，并且他还不断地说“我会给你做出解析表来”。但是它最后还是延期了一些时间。最后，他去仓库找到他们最大的一张纸，足有两平方英尺那么大。他把它平铺在桌子上，裁成小的方形纸，然后他边小声念叨着咒语一样的东西边在上面写下一些小符号。我看着他做了一会，他说:“你为什么不去做些其他的事情？我做完了会给你说的”。我就走开了，没过几个小时回来看时他仍然在念叨着咒语在纸上写写画画。在傍晚的时候他终于说:“我明天把它做完”。最后到了第二天，他说:“完成了！”，然后把它递给我。我说:“我能拿这个做什么？”。
>
> 他就向我展示怎么做一个解析器，我们一起将表格输入计算机。然后我们就试着解析一些表达式，结果完全正确。接着我们又解析了另一些，但是却出错了，表格里面有一些bug。Al说:“喔，天哪”，然后又花费2，3个小时擦除重写，最后我们把新的表格输入进去，终于解决了那个bug，但是却出现了另一个bug。于是我说:“Al，为什么你不告诉我你在做什么？”。他说:“好吧，这实际上并不难”，然后他交给我怎么制作表格。然后我说:“奥，我可以写个程序来做这个”，“真的？”。于是就诞生了**yacc**。

Johnson同时也和Dennis Ritchie一起在向Interdata 8/32上移植的时候写了第一个可移植的C编译器，lint，spell。lint是一个C程序检测器，可以检测C代码文件是否有bug或者可移植之类的问题。spell是一个拼写检查器，它从文件搜集单词然后在一个拼写列表里面查找它们，单词既没有出现，也不能从拼写列表推导出来，就把它从标准输出打印出来。spell不是第一个拼写检查器，第一个拼写错误检查程是Bob Morris和Lorinda Cherry的typo。它包含在第三版到第七版。spell第一次出现在第五版。typo，用Steve Johnson的话说:

> 从来没有检查出我的拼写错误。我非常偏重通过发音拼写，经常弄混ible和able，以及ance和ence，但是typo从来没有发现这些。它是基于三角统计，我的错误全都出现了。但是typo非常擅长找出缺少e的th。它会将单词按最不可能出现的顺序排序。

Johnson的拼写检查器实际上引用了一个用户可以自行扩展的单词列表。他告诉我:

> 我在1973年的1月到9月去滑铁卢大学学术休假，和Morven Gentleman(现在在加拿大的国家科研委员会)一起做一些工作。Morven曾经在原始的ALPAC系统上工作过，他对计算机代数非常感兴趣。我们一起做成了一些工作，他是一个非常有趣的人。事实证明那时不是一个合适的时间。当我去那里时C语言刚被发明出来，当我离开时，yacc被一些人从B语言翻译到了C语言。同时MIT的合作学生Alan Snyder也开发了第一个可移植的C译言编译器。那是一个非常具有颠覆性的尝试，包含了很多非常好的点子在里面。但因为PDP-11内存太小的原因，它是一个四阶编译器。并且它还非常昂贵——每一阶段都将大部分时间花费在读取上一阶段的结果和为下一阶段输出结果，真正做事情的时间反而非常之少。
>
> 因此它非常的慢，就算老式的B语言编译器也要比它快很多。Al努力让它可以在Honeywell工作，但是仍然有许多的bug。正好，在我从加拿大返回的时候，他也回了MIT，因此我接手了这个项目。我的老板Elliott Pinson强烈建议我我应该接管它。我试着修正代码生成器里面的bug，发现很大部分都是Al做东西的风格。你知道，那时候的代码生成器使用一种我称之为“woopsie”的方式:你确信你有所有需要的寄存器，并且所有的东西都会在正确的位置，当有错误出现时就会发出“woopsie”，然后去找一些东西加载到内存里。大多数时候这都会是一个好的代码，但是当它出了错时那是相当可怕的...
>
> 好吧，我去向Al Aho解释这些，这些谈话导致了我认为我在理论方面最好的技术工作——一个为表达式生成代码的算法，它输出可选择的代码...Al和我，还有Jeff Ullman也做了一些，我们一起做了相当有趣的工作。对我来说最终的结果就是我更好的理解了代码生成。
>
> 后来，我们为PDP-11争取到了更多的内存，这使得我可以将一些Al Snyder的程序放在一起运行。不久之后我重写了所有的软件，保留了一点Al的语法，但是没有任何他的代码留存下来。我将这个可移植的C编译器叫做pcc。我原本是打算做Honeywell的编译器，但后来我们认为将它移植到IBM会非常美好。于是我将Honeywell版本做了一些hack，就成了360版本。再之后有一个叫做3A的交换机，我为他做了一个编译器。在这些过程中

> 我意识到我为这三台机器做了相当多的重复工作，因此我开始认真检查，将这三个编译器放在一起发现它们90%的代码是相同的。于是，我在脑海里为其他的10%建立了一个程序...
>
> lint是一个有趣的故事。当我决定我真的要去写一个可移植编译器，将这三个编译器打造成一个真正可移植的版本，我知道怎样去做前端，但是对于代码生成我依然不太了解...所以我需要将前端代码全部检测，但是却没办法去检测。我想或许我可以在前端做一些有用的事情，如果它真的有用，人们将会用它检测自己的代码，也就是帮我测试了。然后，怎么样才是有用的呢？前端是支持解析函数的...我说好吧，一个经常出现的编程错误就是使用了错误的参数个数调用函数。实际上，指出是哪个函数并不是很容易办到。于是我写了一个程序，它可以读取你的代码文件，最初它会告诉你是哪个函数出错了。然后我意识到它可以在函数调用间也做测试。因此lint真的成为我调试编译器前端代码的一个工具。一言以蔽之，它在各种组织中扮演公共品行守卫者的角色。
>
> 你知道，当1977年我们第一次移植Unix的的时候，我们有一个非常严重的问题，我们有许多代码是人们看着手册写的，他们直接复制了手册上的数据结构声明。当我们介绍头文件的时候，我们设想了机器之间的不同。我们想确认Unix程序都是使用了头文件，特别是操作系统使用的那些头文件。因此lint学着去做的第一件事就是检查头文件的一致性...这让我们可以检查大量的代码——第六版的Unix只用了短短几个星期就解决了一致性问题。它帮助我们将它们合并到第七版的使用工具(第七版是第一个可移植系统)。

克制住想谈论Unix没一件事情的冲动，在转移到通信前我们先专注于**awk**。Peter Collinson向Brian Kernighan请教关于**awk**的事情:

> 我想**awk**是从1977年开始出现的，到目前为止它是我参与过的最大的软件项目。我们中有三个人参与进来，但却完全没法工作。不知为什么，两个人在一起工作要远比三个人容易。工作分割非常困难。大家在想法上有很多分歧，这有时是好的因为意味着更多的创意，但是有时却意味着大家并不团结。换句话说和Al Aho以及Peter Weinberger一起工作非常非常的棒，所以我并没有那些问题。
>
> 一共有三个单独的进度在同时进行。我的那部分是那时还在实验室的Mark Rochkind做的一个工具。这是一个非常基础的数据验证工具，它使用成对的正则表达式和消息，通过扫描输入文件查找是否有符合给出的正则表达式，如果有的话我就会打印出消息。当然，他是用C语言写的这个识别器。这是一个非常优雅的主意。
>
> 某种程度上说，那时**awk**的基本概念。有一大堆的模式，还有许多互相依赖的东西要完成。它不像已有的那些程序，它是他们中非常清晰的一个。我也在可编程编辑器方面混了很长一段时间，我们经历了**qed**非常流行的阶段。**qed**是一个非常糟糕的编程语言，但人们偏偏都在用它。
>
> 有一个行编辑器拥有与多个buffer和多个文件，非常像今天的**emacs**。
>
> 我不停思考一定有更好的方式去操作文本。有些事情会像我想要的那样允许你清晰的同时操作文本和数字。

> Al Aho在正则表达式方面是世界级的专家。Peter Weinberger在数据库相关的方面有着非常强的背景，并且对数据库之类的东西非常感兴趣。现在，**awk**绝对不是一个数据库，但是它有这方面的趋势，这来自Peter的数据库经验。因此，我们将这些混杂主意糅合到一起想做出一个东西。
>
> 它最初的想法是写一两行代码，真的是这样。我想它是非常迷人的因为它自动做了这么多事情。它流畅的处理字符串和数字。它是一个解释器并且没有包袱，没有中间文件。人们开始试着写一两行代码，然后代码越来越多，有一些大的难以置信，据说有人曾写出一万行的代码。
>
> **awk**总是在慢慢变化。有一点可以追溯到我的那个想法，我真的希望有一个编程语言可以让我写程序操作文本就像操作数字一样方便。我想相比于B语言发展到C语言，我会去寻找另外一个方向。
>
> 我发明了一种我称之为A的语言，使用了awk的基本概念并且用编程语言将它们包装起来。它工作的并不太好。它有一些功能可以展示，但是它没有使用任何隐式的输入输出或其他类似的东西。我使用了它但是它始终都不够好。它的实现方式真的非常古怪，因此 我一直没怎么用它，也没有其他人用过它。
>
> 那时有一些awk的其他实现。特别是有一个GNU的版本叫做gawk。它和我们的实现非常接近。当他们做gawk的时候和我们有非常多的联系来试着确保他们的版本和我们的尽可能同步。还有DOS的版本，我最熟悉的是MKS版本，当然，它也和我们的非常类似。最后，还有一个POSIX版本。POSIX版本是标准化的awk，那个时候它还是草稿阶段。
>
> 我们多多少少将这些写到了书里，通常我认为是这样。关于标准过程我认为有些好的争论点是有必要的。有一些是和书上冲突的:比如从数字转换到字符串和从字符串转到数字时变量的状态是什么。这类事情可能他们并没有讲清楚。POSIX和我们在一些观点上有着不同的见解，对其其中的大多数，我想人们会同意主流意见。

Mail存在于Unix非常早期的版本中，man页中记载它的维护者是Ken。它允许Ken，dmr，doug，rhm和其他一些人互相通信。它在其他分时系统上也是可以使用的。但是，就像1940年夏天George Stibitz曾经在达特沃斯州通过远程电传打字机使用了位于新泽西州的计算器，对远程工作的憧憬导致了想和其他人分享数据和程序。

最早的网络是ARPANET(开始于1969年)，它打算实现资源共享:高级研究项目代理(ARPA)认为将他们主办的几个项目连接起来会比给它们每一个都购买昂贵的计算机设备更能节省资金。组织成员可以登录到其他成员的机器并且传送文件。文件传输协议(ftp)和远程登录协议(telnet)实现的非常早。由于之前没有可以网络工作的计算机，所以虽然已经有了机器内通信的mail程序，但却没有机器间通信的mail。在1970年早期，计算机邮件已经比较常用。到1973年它已经足够流行以至于邮件头形成了一种事实上的标准。在1970年晚期，ARPANET标准被重写，包含了SMTP(简单邮件传送协议)。

1973年最初邮件格式的模型是一个运行在流行的DEC-10s上面的TENEX操作系统。更多的成功发生在DEC-20s上面，他们的操作系统是TOPS-20或者TWENEX(TWENEX是由BBN开发的TENEX后续版本)。

但是Unix用户呢？

1976年之前，从一个机器向另一个机器发送文件几乎是不可能的。那年的末尾，AT&T研究室的Mike Lesk写了一个“更好的发行规划”(1977年1月，小型系统通讯)，仅仅一个月之后它就被称之为UUCP(Unix到Unix的复制，一个程序从一个机器向另一个机器复制Unix文件)。它被设计为针对300波特率线路。**uucp**最终在1978年发布，它因广泛的使用而被证明为是Unix最重要的部分之一。Lesk的发明导致了改进的需求，因为原始程序如此流行以至于达到了它处理能力的上限。下个**uucp**版本由Lesk，Dave Nowitz和Greg Chesson重写，并出现在1978年10月的Unix第7版中。这个版本也被证明不足以胜任巨大的用户使用量(更多的系统，更多的软件，更多的需求)，1983年七月Martin Levy请求一个新的**uucp**系统。基于1200波特率异步调制解调器的**uucp**设计上达到巅峰，但也不足以满足不断增长的Unix网路需求。

1983年1月中旬，在AT&T贝尔实验室召开了很多人参加的一个会议，会议的结果是由Peter Honeyman，Dave Nowitz，还有Brian Redman一起开发**uucp**的新版本。最终成果就是著名的HoneyDanBer，或者叫BasicNetworkingUtilities1。

我和Brian Redman聊他的工作:

> 我在亚利桑那州取得硕士学位，在1978年6月去了实验室。我在1977年取得了微生物学的学士学位，我妻子在下一年也去了。但是我为之工作的那个家伙离开了我没有留下任何资金。因此我去另一个生物学的家伙那得到一个维护计算机的工作。我在计算机科学部门学了一些计算机的课程，在亚利桑那那是非常新的知识。然后我意识到我可以在那年拿到计算机科学的学士学位。不管怎么说，我还沉浸在学业的时候，后来成为我引导者的David Hansen从实验室带着一些人来面试。他对学生们说:“如果你有手有脚，你就能在实验室赚大钱”。我猜他假设教室里每个人都有一个脑袋。
>
> 所以我去面试了。最初我面试的是西部电子在普林斯顿的一个工作，并且我会在DEC-10上工作，我印象非常深刻。我在维泊尼和霍姆德尔面试了贝尔实验室并且得到维泊尼的一份工作。因此我妻子刚一毕业我们就搬到了新泽西。
>
> 于是我开始在维泊尼的环路传输部门工作。我不能相信我可以拿钱做这个计算机工作。
>
> 我第一次见到Unix是在我的操作系统课程上，当我读论文的时候。我们在计算机科学中心有一个Unix，但是我一直都是使用DEC-10运行TOPS。因此我真正第一次接触Unix是在我到了贝尔实验室之后。我为Ratfor工作，那本软件工具对我来说真的是一本启蒙书。我非常欣赏简单，模块，重用的概念。
>
> 但我当我到贝尔实验室之后，他们说:“你将使用Unix系统，这有一个程序叫**learn**”。它会向你提供一些信息，并且会问你问题。我记得我输入我的答案，然后它说“不，错了”。但是我确信我是对的，于是我退出程序然后开始查找程序的位置。我找到那个程序，还有

> 它的数据库，它的所有答案。我开始深入了解它，明白它是如何工作的(**learn**最初是由Lesk发起，由Kernighan和Lesk一起开发。它包含在第7版Unix中，并且也包含在BSD中。但是并没有在第3版和第5版中出现)。
>
> 你知道，如果它是一本书，我就会看它的背面。但是他是一台计算机，那么我就看它的文件系统。
>
> 我的老板Doug Cory成为Unix电话网络最早的用户之一。一个名为COSMOS的系统为我们布线，后来它叫做COSMIX。COSMOS一台11/70。实际上，我有这台机器的前面板，序列号5001，并且我被告知它是美国第一个卖出的11/70。无论如何，我们在一台名为wh5ess的11/45上使用mail。Sam Arnold和Lesk发起了新的工具**uucp**。我对它非常有兴趣并且成了管理员，因为我们有了越来越多的用户。我发现我能从莫里山使用**uucp**的用户那接收信息，有时我们会将11/45换成11/70，因此我们有了更强大的机器。
>
> 我们得到的第一台VAX是mothra。那时我们运行300波特率，之后得到了一个新的调制解调器，因此我们可以运行在1200波特率。然后是glass终端，再之后就是这台巨大的机器。于是我问为什么我们不能运行的更快些。他们告诉我你可以在50英尺的距离运行RS-232，但是现在是150英尺。因此我对自己说，好吧，我用不上9600，只能用慢一些的，但仍然要高于1200。于是我扯了一条线，我的到了9600的速度，之后又达到了19200。我的同事说“嘿，也给我来一条”。于是我将它成立了一个项目，我在楼下放了一个接线板，在它上面用上了最先进的技术。这非常有意思，那个时候，我有几百个用户，我花费整个周末让每件事都看起来非常漂亮。我们开始在线路上收到噪音和干扰，因为所有的东西都是并联运行。于是我将他们通通分离，然后把他们再弄成一团，所有事情就都工作的很好。
>
> 我开始和计算机中心一起工作。他们有一个排版机用来制作文档。我通过图像排版机放入man页。我试着让系统变得更有用()。不管怎么说，79年我第一次去了圣莫尼卡举办的USENIX会议。我听到了我所期望的事情并且见到很多人。我希望从维泊尼发送东西到计算机中心。因此我加入了**uucp**。1980年我在特拉华州见到了Peter Honeyman。那时我第一次和他在一起...
>
> 我去参加会议，在会上见到了很多人。我们大家交换电话号码，因为大家没法得到对方的邮件。Clem Cole是我记得的一个人，他有一些东西。我不知道是什么但是是一些我想要的东西。我们有8个调制解调器，这在今天非常普通，但是那时大多数员工只有一两个。他在泰克公司，因此我会给Clem打电话说“我们来发起一个连接”，接着我们就那么做。伯克利发出了第二个发行版。我记得他们说“我们可以向你发送磁带”，我说“我不能等了”，于是我们发起了**uucp**连接。

Clem Cole曾经是卡内基梅隆大学的一名学生，在哪里他和Ted Kowalski将Unix第6版从计算机中心的PDP-11移植到了电子工程部门的11/34(就如Cole和我说的，这件事请“绝非易事，因为卡内基梅隆大学为'e'机器做了非常多的hack，而Unix第6版不能支持34s”)。

Cole后来去了梅隆协会工作，他在那里和Dan Klein培育了Unix，并且当协会购买“第一个RK07's(我记得是#17系列)，Dan和我一起用了一周时间为它写了第一个驱动。RK07给了我们2000万！——多到无穷！”。

Cole从卡内基梅隆拿到学位，在俄勒冈州的泰克公司得到一份工作。

> 我在1979年去了那里。在我到那之前的几个月，Steve Glaser发现了一台运行RSX的11/60，确认泰克会购买一个2万美金的Unix第6版授权，并在它上面运行Unix。由于60并不是一台真正的11/40(类似但不完全相同)，我在俄勒冈的第一个周末就是和Steve一起拆开60把卡内基梅隆大学修改的地方还原回去...
>
> 我在俄勒冈州的第一个圣诞节是和Rick LeFaivre，Steve一起动手安装PDP-11/70。那台机器成为第二台主要的**uucp**工作站，“teklabs”运行V7，我们每天都和“Marxs”通信(Cole谈到了新泽西贝尔实验室的chico和harpo。1981年的USENET地图上标注了teklabs-chico的连接)。
>
> 我们为VMS写了第一个TCP，它在今天依然存在，被称为“CMU-Tek” TCP。我们用70帮助着建立了“Magnolia”，一个68000多处理器Unix工作站，它运行我和Steve基于Unix改写的操作系统Magix。我们得到一场爆炸！
>
> Brian Redman和我每天都会通过**uucp**讨论一些重要的问题，例如用C语言编译器编译“冒险”游戏。事实上，我怀疑我们分享微处理器开发工具要比这些有趣的问题要少。当然我们也分享了11/70的改变和增强。我们有一个共同的敌人——Digital，它不乐意Unix是在他们的帮助下推广的。

Redman继续说着他的故事:

> 我发现我还会定期通过发送电子邮件联系，所以我们有了一个社区，这就是NETNEWS的由来(Usenet起源于1979年杜克大学毕业生Jim Ellis和Tom Truscott的一个主意。Usenet的第一个实现是Steve Bellovin写的，它是北卡罗莱纳州立大学的毕业生。USENET在1979年用于杜克大学(**duke**)和北卡罗莱纳州立大学(**unc**)交换信息。1980年**phs**加入进来。Armando Stettner和Bill Shannon组建了第一条东西部连接线路——**duke**到**ucbvax**，同年也建立了到**decvax**的线路。USENET现在是一个包含5000多个组织和几百万用户的巨D大无中心分散系统)。
>
> 我立即在chico搭建了NETNEWS。我记得花费了相当多的时间表达一个非常聪明的问题，因此我得到一个非常聪明的答案。这全部是出于一种渴望。甚至于好的目的，当一些人说“我会为你制作一盘磁带”和“我会寄一盘磁带给你”...这些都不够好。
>
> 那就是它开始的时候:当NETNEWS开始出现，**uucp**坏掉了。Lesk根本没有考虑**uucp**现在会用作什么用途，也没有考虑到传送的内容。我的问题是NETNEWS上创建了数量巨大的文件，当这些文件开始传送的时候，Unix会执行**uuxqt**命令。但是当磁盘负载很高的时候，连接就会偶尔死掉。并且在恢复之前，另一个**uucp**和几百个更多的文件又开始传送。而当你进到目录里看时系统就会超时，因为有太多东西需要列出来。所以我对**uucp**的第一个重要贡献是我让**uucp**可以为不同的站点使用不同的目录。
>
> 不管怎么说，我做了那些事情。Honeyman发出一封邮件说“让我们一起讨论下UUCP的事情。我们把所有的修改放在一起，然后将它提供给每一个用户，这样我们就都运行着同样的东西”。于是我们发出一个通知，并且在莫里山举办里一个会议。哥伦布的Mark Horton，Honeyman，我，Bellovin，以及实验室的所有运行系统的人都参加了会议。我想Honeyman有一个出席列表。根本上，Honeyman，Dave Nowitz和我一起将所有修改合并在一起。Steve Bellovin也做了很大的贡献。我们在那天晚上开始动手。事实大家分工非常好:我做目录相关的事情，Nowitz对于安全更感兴趣。所以我们一起分工合作并且做的非常棒。

Bellovin改进了他为NETNEWS写的代码，然后Steve Daniel和Truscott都做了改进。最后的结果是**A News**。1981年，毕业于伯克利大学的Mark Horton和高中生Matt Glickman将**A News**重写成为**B News**。1987年多伦多大学的Henry Spencer和Geoff Collyer做出另一个选择**C News**。还有另外很多有用的NETNEWS修改版本，Spencer Thomas，Rick Adams，Ray Essick，Bob Kolstad，以及其他很多人都做出了自己的版本。但是让我们返回到UUCP。

感谢Peter Honeyman，这是他在1983年会议上的部分会议记录(用他独特的文字):

> uucp爱好者的第一次(也是最后一次)会议在1983年4月20号举办。邀请了allegra！honey，vax135！martin，eagle！karn，rabbit！ark，mhb5b！smb，harpo！ber，cbosgd！mark，floyd！trb，research！rtm和eagle！dan。另外还有USG的代表mhtsa！sc和mhtsa！brad，whuxlb！pep和gummo！mmp.下面是未删减的会议记录。

```
From: Honey Wed Apr 20 00:46:26 1983
To: uucplovers
Subject: minutes
Cc: dmr doug
```

原谅我下面的文字编辑，由Levy组织的一个讨论uucp问题的会议的记录。会上的记录人员有:

eagle！dan，vax135！martin，ihnp4！gjm，gummo！ber，mhtsa！lsc，mhb5b！smb，mhtsa！brad，eagle！karn，whuxlb！pep，allegra！honey。

第一个讨论话题是企业电子邮件项目和它的伴生项目网络功能中心。网络中心上有452个成员和774个设备与组织共同管理。Murakani给人的印象是这在实验室是固有设备的起始阶段，其他任何问题都是临时的。L.sys数据库正因为新的数据库和新的的标准不断出现而逐渐走向失败。每个月关于L.sys意义的讨论最后得到了一致意见，工作站应该继续维护一个本地的L.sys，并且那个本地文件应该首先被搜索。参与的人有la koenig，honeyman以及所有人。

关于修改uucp的活跃讨论占据了会议的大部分时间。就像有终端版本的uucp，工作站也有众多版本。其中最主要的竞争者是由Robert T. Morris最初写成后来又经Tom Truscott修改的usg 6.0。Cohen严肃谨慎地对待将优秀功能集成到6.0过程中的各种困难。虽然如此，商业公司也参与进来，我们对他们表示欢迎。并且我们同意以此组织为起点来开发一个满足每个人的版本(特别是我们自己)。

被强烈提议的是spool目录的构成方案。当Truscott为C，D文件使用不同目录(我猜其他文件也一样)，Redman和其他人都使用远程工作站的名字作为子目录。热烈的讨论持续不断，最后获得的一致意见是后者更为强健。Levy的关于目录构成应该包含机器名字简短前缀的提议受到大家的欢迎。当usg中使用setuid进程执行mkdir成为一个问题，Bellovin为了推翻这个特性耐心的解释技巧。经过三或四次跑题，会议再回到正轨，让人感觉cico或者其他的守护进程应该在spoll目录执行rmdir，最终没有得到大家都同意的目录构成方案，我想在信息时代这个问题应该在更合适的态度下讨论。

...

最终我们同意使用最新6.0版本的uucp，并对它进行大量的修改(比如修复bug，让它工作的更好)。Honeyman领取了conn.c，Redman会修改脱机部分，Levy愿意去写一个L.sys编辑器并且帮助Murakami标准化L.sys的行为。Bellovin接受将脱机目录的修改移植到uuxqt。当Nowitz反对对于ifdefs的依赖时，Cohen指出usg不可能欣赏在usg保护下的任何事情。感觉就像我们可以隐藏特性到一个单独文件并且将我们的努力成果掩饰成一个向着Unix第7版兼容性做出的手势 ，一个显而易见但是愚蠢的谎言。总之，这个问题并不重要。

> 议程上的最后一个话题是组织提高安全意识方面的尝试。随后就开始了争吵。在各种和各方面的名誉遭到破坏的过程中，无疑引起疯狂但并不危险的反应，最后名没有得出好的结果。Honeyman不赞成Mcilroy青睐的允许远程机器拉取文件的主张，转而支持在除本地集群意外的机器上禁止这种能力，继而建议可以从非uucppublic目录拉取一个许可文件(这样远程工作站就不能拉取其他工作站发送的文件)。Redman指出L.sys中唯一值得保护的信息是电话号码(当然是数据中最安全的一部分)。Redman请求大家对于即将到来的GEI规章的意见。
>
> 接下来是空空肠胃发出引人注意的巨大声音。人们一致做出一个决定，先暂时休会大家一起去附件的一家中国饭馆，这是那时候各地uucp黑客的传统。这个决定随后修改为去橡树屋(一个不好的前兆)。

橡树屋是莫里山贝尔实验室的咖啡厅，之前几年提供餐桌服务。

在uucp漫长的讨论之前，我提到了ARPANET，TCP，以及ping。由于Mike Muuss参与了TCP早期版本的开发，并且也是ping的作者，我想有必要说一说他的故事。

> 我毕业于霍普金斯大学，我有一个相当独特的职位:运行RSTS(DEC的资源共享分时系统)的PDP-11/45。我到的时候它每天只运行几个小时的Unix。部门说我们除非我们也能运行RSTS，否则我们不能运行Unix...因此我们把它实现了，它成为我们的第一个软件项目——在Unix上运行RSTS。我们真正开始工作是在75年后半年，我们在秋天开始转向第6版。

Muuss在1979年毕业于约翰霍普金斯大学，之后去了弹道研究实验室工作。那年的9月他的工作是于设计实现BRLNET本地高速网络的原型，这个项目是一个和美国军队签订的合约。1980年早期Muuss扩展了BRLNET协议，并且领导一个小组将伊利诺斯州立大学的NCP移植到PDP-11上的Unix。那个团队配备了一个11/34。1981年后期，Muuss定义了实验性的TCP/IP协议，并且着手在PDP-11上的BRL/JHU Unix上实现它，BRLNET则没有继续研究。可能更重要的是Muuss通过ARPANET和USENET向7000多名订阅者发布了TCP/IP摘要。最重要的是，Muuss的工作包含在两个MIL(军事)标准里面:1777和1778。几乎每个现在的TCP/IP实现都包含了Muuss在BRL开发的软件。Muuss告诉我:

> 我们甚至有段时间参与到通讯行业。Ron Natalie是我在那里的主要开发者。我们开发了一个小的操作系统并且将所有东西使用PDP-11通过ARPANET发送了出去。所以我们算是参与了通讯商业。然后，在1983年我们完成了下一个大步骤，争取到了光纤补助。AT&T展出了带状线光纤...我们一箱光纤，一共有145根。我们用他们立即组建了一个全光纤的局域网。

那是10年前的事情了。根据出版物资料，1984年BRL的网络是非并行的。TCP/IP协议到了BBN和伯克利。我会在第18章再次讲述协议的开发。

用户 第二部分

对于我来说，整个Unix故事最让人着迷的部分，是它快速传播到加拿大，欧洲，澳大利亚和日本。因为一一追溯会变得非常乏味，我会选择几个国家让真正的亲历者给我们讲述那时的故事。我之前曾提到过这几位用户，不过现在你可以到到他们自己的声音。

第一位是皇后玛莉学院(QMC，现在是玛莉女王和韦斯特菲尔德学院)的George Coulouris教授。我们他QMC是如何在1973年晚期得到第4版Unix的。

> William Newman曾经在1971年10月来我们实验室做了差不多一年的访问学者(然后他去了Xerox PARC，现在他在剑桥的Rank Xerox EuroPARC)。我们曾在它的指导下发现交互微机系统的价值。他和Mike在QMC合作开发了一个单用户交互操作系统(名字是MIFS)。MIFS运行在一个Interdata计算机上并且在TENEX上完成早期开发。它被用来支持两个研究项目，一个是审查William的设计的图形编程语言，另一个是审查我们一个小组使用微程序仿真设计的的新机器架构。和Unix类似，MIFS包含了很多组件和工具——一个文件系统个，一个系统编程语言，一个编译器，一个行编辑器(可以单字符交互)，一个调试器，等等。几乎所有这些都是由William和Mike开发的。
>
> 当我们开发MIFS的时候我们还没有听说过Unix，但是当William还和我们在一起的时候，他去访问了贝尔实验室。当他回来，他说“他们在莫里山开发了一个和MIFS差不多的东西，但是它多用户并且泡在PDP-11上面”。Mike Cole得到了一些文档，然后开始研究它们，推断它将会给我们带来很大的帮助——它提供了如此多对MIFS有用的工具，有了这些我们会在它的基础上做的更好。
>
> 我们没有9路驱动，只有一个DEC纸带和一个系统工业10M可交换驱动。真正的问题是Ken没有一个SI控制器的驱动(世界上没人有)。显然我们不能在Unix上开发一个DEC纸带的驱动，因为外围设备的数据交换会让系统慢的可怕。Mike Cole使我们系统的老大，最后也是他解决了这个问题。他在73年夏天用几个月访问了Xerox PARC(那时William Newman就在那，Mike去和他一起工作)，在那里他将从Ken那里拿来的第4版纸带放到DEC纸带驱动器上。在PARC的时候他去了Bezerkeley，为SI控制器写了一个简单的驱动，然后将它编译到内核里。但是Bezerkeley没有SI磁盘，因此他在那里不能测试程序！当他回到QMC时将DEC纸挂载上去，直接就运行成功了！Mike是一个非常棒的程序员。
>
> 我们打算运行Unix，却冒失的买了一个磁盘和贝尔实验室的系统不兼容的机器。Mike从来没有写过C程序，没有用过Unix，更没有写过Unix驱动，但是他写的磁盘驱动第一次就运行成功了，这真让人惊奇。

Mike Cole补充说:

> George的描述非常正确，但是有一些观点显然夸张了我的编程技能。我的记忆有些不太清楚，但是我想接下来的故事应该是发生在73年夏天。
>
> 在我去West Cost的路上并没有从贝尔实验室拿到纸带。而是在PARC的时候我在伯克利Mike Stonebraker的小组花了几个下午并且从它们那里得到了一份Unix的拷和文档(也得到了必要的AT&T分配)。然后我开始学习足够的C语言去写SI磁盘的额驱动，并且使用它们的带有LA30 DEC刻盘器的11/45构建了一个系统(一段痛苦的经历)。我记忆里他们那时正在痛苦的开发Ingres。
>
> 因为系统要运行起来的希求非常紧急，我返回QMC的时候决定停止贝尔实验室的工作去英国。10月Ken Thompson检查了我的代码，发现一两个bug(使用C语言的指针几乎总会出错)。我们将整个Unix软件放在7，8个DEC纸带上。然后Unix就转载PanAm袋子里第一次来到了英国。
>
> 建立了UK的第一个工作站，我记得我曾在74年同时被爱丁堡和剑桥邀请去介绍我们做的这个奇怪的叫做Unix的系统。我被消极的回应所打击到了，特别是剑桥的Neil Wiseman和Charles Lang表示不会考虑接受这种没有供应商支持的系统(我记得那时他们在用RSX-11)。这就是早期传教士的失望。

不久之后，Nigel Martin成为QMC的学生，城市大学的Sunil Das得到西部电气的一个授权，不需要再等新泽西购买到英国:Sunil从Nigle那里得到一个第5版Unix。

在那段时间，多伦多大学得到了一份Unix。在详细讲述这段历史之前，很有必要了解加拿大和Unix的紧密联系。Brian Kernighan是一个还在多伦多上学的学生，Al Aho也在那里得到了他的学位。Heinz Lycklama在安大略汉密尔顿的麦克马斯特大学接受教育。Morven Gentleman去麦吉尔(蒙特利尔)并且在滑铁卢(安大略)学院呆了了15年。Tom Duff是滑铁卢的肄业生，去加入了多伦多大学的Rob Pike，David Tilbrook，Howard Trickey，以及Mike Tilson。Tom Cargill在滑铁卢拿到了博士学位。还有许多其他人可以提到(甚至我也作为多伦多大学的职员长达11年)。

1974年中期多伦多大学购买了一台PDP-11/45，运行第5版Unix。11月1日，它收到了第6版的授权。其他大学就没这么多的耐心，Tom Duff告诉我:

> 1974年的滑铁卢大学。Steve Johnson正在学术休假(我就是在那个时候退学的)，他从一个实验室打电话到Unix机器。一些人通过他的肩膀偷看到了号码，然后拨通号码窃取了系统的源代码。

Doug McIlroy告诉我那个传说有些地方是错误的:是他的在实验室的电话号码被偷了。“我在滑铁卢打电话，Unix的电话号码显示在学生计算机俱乐部的账单上。那时学院发现并且绝望的给我打电话，我们和他们的代码被弄得一团糟”。

下一年，Duff成为多伦多的毕业生(Rob Pike是1975年的暑期实习生，Dave Galloway是1976年的实习生)。Duff说:

> 9月我们收到了11/45，12月开始运行第5版Unix。我记得75年早期它开始运行第6版。我在第6版附带的split I&D出现之前开始做我的论文。Mike Tilson，Bill Reeves和我是冷加载第5版纸带的成员。
>
> 我们遇到一些问题会打电话给Ken寻求帮助，他显然接到了非常多这类电话。大概20台第5版工作站远远多于他之前的发行版数量。mkfs命令指出了如何在有4000个存储块的RK05上构建根文件系统，并且留下872个存储快作为交换空间。Bill在我们的第二个RK上输入同样的命令来制作文件系统，他觉得不值得为了浪费的一小点字节而去翻看RK05的文档找出存储空间的准确大小。仅仅一周后我们就遇到了麻烦，我花了几个小时阅读**check**命令-s的代码找出我该如何在超级存储块上打补丁来修正文件系统，然后运行**check -s**更正了这些存储块。

Duff也遇到了问题，他使用部门的Xeror机器来创作**yacc**手册，着这些天里官僚一直索要复制的权利。

> Mike Tilson改进了Duff的版本。
>
> 第一个系统运行在PDP-11/40上面，那是我们等待交付11/45时候的一台过渡机器。我相信我们能验证Unix可以使用仅仅64KB的内存启动。那个纸带我Ken个人制作的，他在贝尔实验室将它给了我(然后，就像现在，AT&T运送的很慢。但不像现在的是，你可以绕过书面工作而自己直接去取纸带)。
>
> 11/40有一个GT-40显示器:它的驱动是多伦多大学的第一批自定义修改。

1977年11月1日那天多伦多收到了一个图像排版机的授权。我将在不久之后返回多伦多。现在，让我们一起去欧洲看看。

Teus Hagen在荷兰关于Unix的故事开始于第10章。Hagen相信“Unix中的技术(字符串表达式，yacc，行处理编辑器，文字处理，awd，字节流思想，管道)在早年将欧洲的计算机科学推进了一大步，在短短几个月的时间内欧洲的技术水平大幅提高”。他同时也对我说:

> 我想我们和伦敦的皇后玛莉学院同时从实验室得到了Unix，这比其他人得到第6版早了一年的时间。我现在仍然保留着最初的手册。
>
> 接下来是奈梅亨大学的Hendrik Jan Thomassen，他在经历无数讨论后安装了Unix。第三个收到Unix的是自由大学的Andy Tanenbaum(Andy那时正在忙着做他自己的操作系统)。
>
> 那时我们的问题之一是计算机设备高昂的价格:一个磁带就要花费4.5万美元，一个驱动器(DEC RM05)的价格是5.5万美元(又一次我们中有一个人试图围着机房飞)。
>
> 我和DEC的维护者有麻烦，他们需要RSX或者VMS才能运行，也不能修复我们的硬件问题。不论如何，我们的错误信息要比Unix上的要好很多，我们最后还是可以说服他们。但你怎么在一个有瑕疵的计算机系统上安装VMS或者RSX?

Hagen不是唯一在DEC方面有麻烦的人，Lou Katz告诉我:“基本上，DEC的维修人员会运行它们自己的怎断程序并且说是我们的软件有问题，因为他们的怎么程序没有找到任何问题。特别是事实证明DEC的怎断程序没有测试PDP-11的监管模式，但是Unix需要这个”。Armando Stettner说:“有许多Unix在PDP-11s上做的事情，不管是DEC的操作系统还是诊断程序都没有测试过”。

> DEC不相信所有的东西都是C语言写的(C是什么鬼?)，并且只有500页的代码。如果你想的话可以随意修改源码。与之形成对比的是DEC堆积的手册和论文堆起来就要和机房差不多高了。
>
> 最初，我们有一台PDP-11/45(带有核心内存，一个很棒的加热器)，但是只有一个1.5Mb大的RK05磁盘驱动器作为数据交换接口。我们不确定公共交通工具会对磁盘上的磁场会有什么影响(在阿姆斯特丹有很多有轨电车)，所以我们我们没有去冒险从数学中心走到自由大学(大约7英里远)。后来我们有了更多的信心，在电车上可以很容易认出Unix的相关人员，他们在车上都会抱着装有RK05的大箱子。

欧洲的第一个Unix网络是Hagen组建的，它后来成为正式的欧洲Unix用户组。在欧洲建立一个网络并不像美国的UUCP和ARPANET那么容易。他说:

> 拨号器和调制解调器在欧洲不被PTTs许可，因此我们在建立Unix网络的时候有非常的问题，PTT的家伙们丝毫想象不到UUCP的功能和作用(我们是说不同语言的两个物种)。于是我们将一个办公用的桌面半自动拨号器放进一个盒子里，盒子是从110/300波特率的调制解调器上拆下来的。一个小公司组装了很多那种盒子，并且将它们作为无线扩音器在边境走私售卖。我在DEC的好朋友Armando Stettner帮我建立了和美国的通讯连接(只有邮件，那时新闻还没有出现)。那时还没有压缩功能，我们因为缓慢的速度花费了不少的钱。这就是为什么如今欧洲有如此严格的网络计费系统。因为国际长途费在晚上会便宜很多，我们改为在夜间和美国建立通讯。一天早上我发现和美国连接的线路挂掉了:满屏的通讯报错和重试信息，电话已经接通了10小时。10个小时的美国长途电话是非常昂贵的，所以我被主管叫到了办公室(因为他是数学家，所以对这些事情不太了解，而且他也不想和计算机的家伙们打交道)。我惹上一堆的麻烦...
>
> 我们得到新闻是几年之后的事情了。我记得我发布在网上的第一篇帖子，因为美国的人们对我感到特别新奇，不断问我地狱是什么样的，我一小时又一小地在他们提问之前回复帖子！他们终于发现了美国以外的另一个世界。
>
> (我向Teus询问关于Kremvax的事情)
>
> 那天是愚人节。我们从数学中心发表了一个帖子宣称克里姆林宫VAX在网络上发布，帖子的地址可以追踪到莫斯科。随着帖子在网络上散播，人们互相喊叫，我听说美国政府安全会议也这个消息惊动了，官方试图立即终止和欧洲的通讯线路。幸运的是接下来发现这个措施并没有执行，网络继续保持着无政府状态。

Kremvax真正的创造者是Piet Beertema，那条消息在1984年的4月1日发出。六年后，当demos.su介入USENET，Demos的资深程序员Vadim Antonov不得不努力说服人们那不是另一个恶作剧。Armando Stettner的故事和Hagen的有些不太一样，他告诉我:

> 我们在很多地方使用UUCP。但是我们没有做出NETNEWS。Teus和我在一个会议上讨论建立NETNEWS的主意。我说“看，做出NETNEWS连接大西洋有什么问题吗？是时候连接起来了。如果在你我的电脑上都安装compression，我就可以向你发送新闻了”。我不知道怎么开始的。但Teus后来给我发邮件说“复制这个东西到电脑上”，然后我们让它运行起来了...不久之后我开始通过Teus的机器和欧洲交换NETNEWS。成功了几个星期后，我通过Robert Elz的机器又将澳大利亚接入到网络。

在那时，Stettner的机器每个月要花掉25万美元的话费。DEC的一名会计师问他为什么仅仅四条电话线就花掉如此之多。“哦，那是我们的高级机器在交换数据”，Stettner说。

Settner向澳大利亚发送消息并没有听起来那么的奇怪。新南威尔士大学的John Lions读到1974年CACM的文章，那时新南威尔士大学正准备去买一台PDP-11/40，并且在年底从西部电子得到了一份授权。下面是Lion的叙述:

> 新南威尔士大学用一台运行KRONOS的CDC Cyber 72-26计算机替换了之前的1965-vintage IBM 360/50。新的网络使可以驱动100台打字机式(300波特率)交互终端和12台PDP-11/40计算机。当7台最小配置(16Kb内存)的PDP-11/40s买来后，5个系统安装了128Kb核心内存，每台机器都配有一个DJ11终端复用器和3个RK05磁盘位。
>
> 我们拷贝的Unix软件出乎意外的好。仅仅两个晚上的实验我们就被这个有趣的系统迷住了，很明显它比RSX-11D更加可靠。我们的问题变成“无法提供远程批处理服务的情况下，如何能像晚上一样全天都运行Unix？”。显然答案是写一个运行在Unix上面的U-200模拟器。一个由Ian Johnstone领导的4人小组接受了这个挑战，他们用了两个星期开发出一个系统原型。4周之后，RSX-11D就被替换下来，之后再也没有用过。

1974年Greg Rose毕业于新南威尔士大学，我和他聊起这段历史:

> 我是毫无经验的毕业生。那时大学处在一种很不稳定的状态，它成了一个IBM商店...360/50已经退役了，实际上它不再用于教学和科研，完全只用于管理——最后，他们也不想重写所有的PL/I程序。有个建议是使用网络，所以学校买了5台11/40s和7台11/10s。计算机网络在1975年投入使用，在1974年中期就有两篇有趣的论文发表出来:一个是关于Unix的，另一个关于pascal。于是John Lion离开并且拥有了Pascal，而Ken Robinson有了Unix。
>
> 教学是它们完成的工作之一。第二年需要汇编编程的可能，而PDP-11上正好有汇编器，因此我们的课程难以置信的可以在PDP系统上使用汇编器。我们不得帮助学生们学习和调试，这样我们就不能在机器上运行Unix了。Ian，Johnstone，Greg James，Ian Hayes，Peter Ivanov，还有John Wainwright决定在这些机器上运行Unix，他们不得不写一个U-200的设备驱动和模拟器，这样部门就可以在运行Unix的情况下继续使用RSX-11。但是必须有人来维护它确保它一直都可以正常工作。Ken Robinson说:“我们有两个不惧挑战的学生，就让他们去做吧”，于是我和Chirs Maltby被选中了。
>
> 我的第一个任务是为一个汇编课程写一个支持批处理程序，那时我还是一个学生！
>
> 在75年到76年Unix在大学里取得了巨大的成功。我们拥有11/70和11/45，管理学院也决定使用Unix。76年成立了伍伦贡大学。最初它是新南威尔士大学的一个分校，但后来分离出去了。那时Interdata几乎放弃机器，为了闯入市场，他们给伍伦贡大学一个难以拒绝的价格。我们常常参加这类信息会议，大概有10个人坐在一起喝咖啡。其中Rich Miller说“你知道我们拿到的这些计算机上装着可拍的操作系统。我们不明白为什么，由于Unix是用高级语言写的，我们不能让它工作”。
>
> 于是我们接下来几天都坐在一起，大多数时候我和Ian Johnstone讨论架构和该如何让编译器产生代码。Rich离开了一段时间(他是一个让人惊奇的家伙，我想他现在在牛津)，等他回来的时候，他说“我想我们有了一个编译器”。因此他和Ian去跨平台编译了一个编译器，然后把它带回伍伦贡。我们带着9路纸带来回跑了4趟100公里的路程。当我们返回伍伦贡，因为他们在Interdata没有磁盘驱动器，我们会在一个Univac 11上读取然后写到可以在Iterdata上使用的磁盘包上。
>
> 但是我们犯过3个错误:一次是编译器，一次是操作系统的剪裁，最后一个是系统升级。因为我们最初安装的是第5版，后来得到礼物第6版，Rich说“好，给我一个最新的源码纸带”。
>
> 就在同时，Steve Johnson和Dennis正在做他们的移植。

这真的是一个不同寻常的工作，Johnson回忆说。Rich Miller在1977年2月澳大利亚的第二届Unix用户组会议上谈论起它。

在1976年3月出版的*UNIX NEWS*上，发表了一封标注日期为1975年10月10日的信件，是John Lions发给Mel Ferentz的。这封信的开头是:

> 我们最近看到你早期的两封UNIX简讯，因此没有更早的做贡献。不管真么说，我们感谢你的工作并希望它能够成功。

Lions在1976年1月21日再次向Ferentz写信宣布第一个“东海岸”Unix用户会议将于1976年8月29日在新南威尔士大学举行，并且告知用户纽卡斯尔大学和悉尼大学也加入到了Unix社区。

9月17日，Lions写信报告会议内容:“有30多个人来参加”。这次会议如此成功，大家确定下一次会议定在1977年2月18日。更重要的是:

> 考虑到Unix非同寻常的售后支持，大家一致同意Unix用户间需要合作。

1977年，John Lion完成了他的《UNIX操作系统注解》，用两卷书再现了第6版的源码。这两本薄书可能是最重要的未出版计算机图书。我问他关于源码和注解，他说:

> 我那时在教操作系统。我要和一个教编译器的同事竞争，他让学生去写一个真正的编译器。所以一个关于代码阅读的练习看起来是个好主意，并且我们的Unix授权没有明确禁止这项活动。为什么是Unix？因为那时没有什么选择，你知道它是非常杰出的，而且它比竞争对手要好很多(我们也得到了Brinch Hansen的SOLO系统)。

1977年《UNIX NEWS》(第2卷第3号)宣布了这本书，附有Ferentz的评论:Ken Thompson曾经看过这本书的第一版，反馈说这是非常好的工作。包含空运邮费的价格是17.7加元(不到20美元)。UKUUG通讯宣布了代码和注解，但是下一期声明应该向贝尔实验室预定，到了1978年这本书不再可用。在整个计算机科学界它们一定是最常被影印的图书。他们加上了恰当的版权声明并且约束授权，但还是没有成功，西部电气阻止了如此一本有价值的书流通。我承认我同时有影印版和复制版，它们有着明亮的橙色和红色封面，还有John给我提词。

> Mike Cole问我是否确认Jim Curry第一次去欧洲带着Unix，将它安装在奥地利的IIASA。据我回忆William来到这里后不久Jim就离开了PARC。72，73年Jim在QMC访问我们，并且让我确信我们彻底疯了，我们要将新的教学系统构建在Unix上面。

Curry是国际应用系统和分析协会的成员，位于维也纳北部的拉克森堡。他是74年到78年计算机服务的负责人，他告诉我:

> 1975年1月我在PDP-11/45上安装了Unix第5版。据我所知，这是欧洲安装的第一个Unix。
>
> 1978年中期，我招聘Jim Kulp接管我计算机科学系主管的工作，从那以后直到1982年我都没有做太多计算机科学部门的事情。因此我对那段时间详细的记忆有些模糊。但我知道Jim在1979年早期购买了VAX/780。我不确定这是不是奥地利的第一台VAX，但是很明显它是第一个运行Unix的。Jim最初运行了一个非常早期的BSD系统——可能是3BSD，或者是4BSD的预发布版。
>
> 网络发展方面，IIASA在历史中并没有特殊的位置。我们早在1975年在几条点对点电信链路上无所事事(主要是和东欧)，但是我们第一次真正使用互联网邮件是通过一条租用的2400波特率线路使用UUCP发给tuvie(维也纳技术大学)。我不记得确切的时间，但是发到tuvie的设备上后，我觉得那没什么意思。

IIASA的Herb Hasler说:“我相信IIASA在奥地利第一个拥有VAX的”。更重要的是，IIASA的Jim Kulp写了第一个版本的Unix任务控制，它包含在4.1BSD(1980年)。

又一次，远程用户的工作包含在一个主要发布版本。

John Lions和Mel Ferentz的来往信件表明，大多数的Unix设备远距离通信，都被计算机中心的支持者所看到——借助于《UNIX NEWS》和它成功的专栏《;login:》。比如，AT&T和西部电气的律师不愿掌握 他们可以或者不可以 公告导致的 哈佛科学中心的Lew Law将着手重新修订和发布Unix手册...1975年发布的第6版《UNIX编程手册》将全部重新修订。大多安装版本将会删除几页大多数用户不需要知道的部分。????????

《UNIX分时系统使用文档》第6版将删除第1章(设置UNIX)和第13章(UNIX安全)。

同一期杂志还包含了Bill Mayhew(来自波士顿儿童博物馆)的一篇文章，题目是“如何使用49美分解决PDP-11/40的静电问题”。下一期杂志(1976年5月至6月)宣布“第一个软件交换邮件”。Mike O'Brien第一个邮寄了软件纸带，哈佛软件的复制品，寄到了芝加哥的伊利诺斯大学。1976年8月《UNIX NEWS》刊载了Alan Nemeth的第一篇文章，讨论的是I&D的浮点数模拟问题。1976年11月出现了O'Brien发表的第二版软件公告:

> 我得到了贝尔实验室第6版的“diff列表”，总之我逐一标记了50多处修改，并将列表发送给了Ken请他验证和注释。这些修改可以通过发送特殊的请求得到...

第二版包含了来自RAND公司，海军研究院，圣地亚哥的加利福尼亚学院，Yale以及UIUC的贡献。《UNIX NEWS》也刊载了来自多伦多大学的一个图形软件公告。

第三版软件发行版在1977年5月至6月的期刊发布。这一期还包含了一条信息，超过150人参加了1977年5月的厄巴纳会议:

> Mini-UNIX已经发布，LSI-UNIX和Mert可能会在稍后的时间发布。

记住:没有通知，没有支持。厄巴纳会议上出现了第一次编程竞赛，是由Steve Holmgren发起的。所有用户都不得不互相帮助。

所有用户中只能依靠自己的是东京大学的Haruhisa Ishida。“我是日本第一个得到Unix授权的人”，他对我说，“在1976年我和AT&T贝尔实验室签订了协议，那个Unix的版本是第6版，只经过很少的修改我就能在LSI-11上运行它”。LSI-11是DEC自己的基于微芯片的PDP-11模拟器，生产于1974年。最早的版本没有内存管理，但可以使用一个单独的芯片实现这个功能。恰巧东京大学购买了这个版本的LSI-11，并且Unix第6版没有内存管理就不能运行。

1980年，一个叫SRA独立软件公司派出Koichi购买了一台VAX，并且带着Bob Schulman去日本在它上面安装Unix。Schulman告诉我:

> 我在1980年的夏天去安装第一个32V，在VAX11/780上面安装BSD3.0。
>
> 他们购买VAX的原因当然是他们想要Unix。他们想要Unix的原因是Kishida，Kishida是一个到处旅行并爱好社交的家伙。他去欧洲和美国参加了很多会议。那个时候，他认识的每个研究人员都说自己正在使用Unix。所以他决定也去搞一个Unix。这样，他就可以和这些有名的研究院拥有同样的环境。我喜欢他的这个理由。
>
> 当我在1980年夏天到那里的时候，他们已经安装了780，但是没有用它做什么，因为他们需要32V纸带。一旦安装好了，如此巨大的投资需要一点神明的帮助，所以他们找来了道教长老，在机房举行了祈福仪式。在仪式上，到处撒了棕榈叶，长老说着古老的日本土语(可能没有人听得懂)。当然也提供了清酒，但是我不认为那台机器喝了它们。
>
> 由于从来都没有告诉过我的安全原因，安装只能在晚上进行。因此你晚上11:45以后开始工作，第二天早上6:00之前不能离开。这是一段令人沮丧的夜班。
>
> 我工作的一部分是向SRA教授“Unix的方式”。我需要试着解释“hacker”和“nerd”的关系。因为语言和文化的障碍(我的日语一直很烂)，让他们最终明白所花费的时间比我之前预计的长了很多很多。“Hacker”对于他们来说是一个很有吸引力的感念。“Nerd”让他们很难理解，我认为他们从来没有感受到我试着描述的nerd的缺点。可能他们觉得一个偏执的呆子并不是一件坏事。

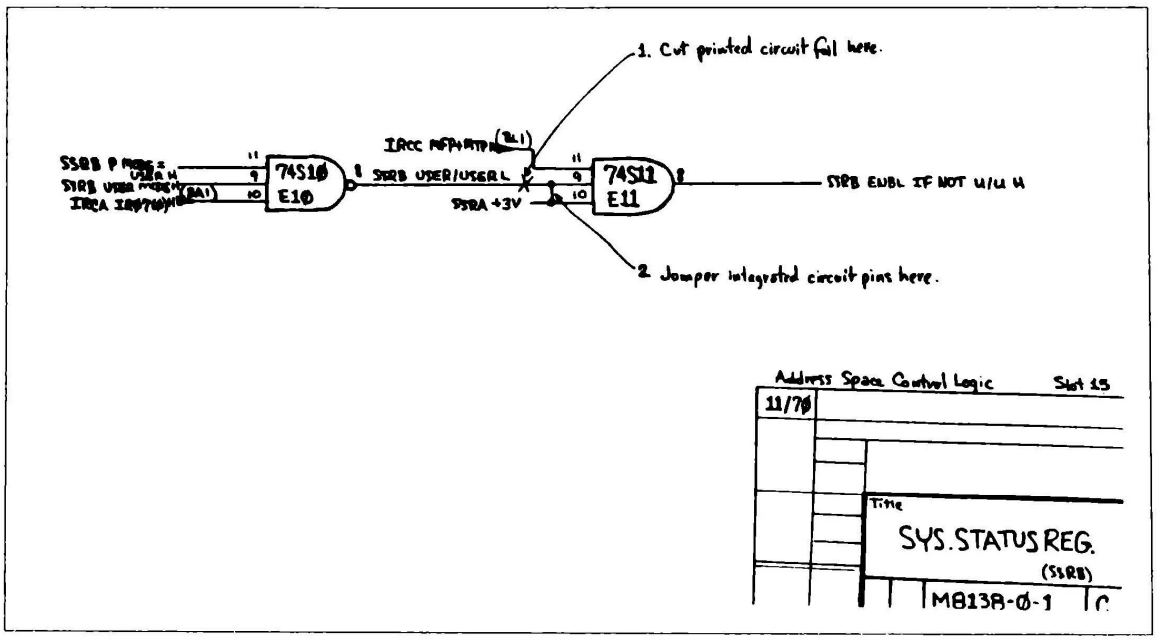
日本Unix社团(jus)是由Kouichi Kishida，Jun Murai，Nobuo Saito建立的。我会再回来继续说这个，但是现在我想回到1977年的USENIX《;login:》。那一期通讯向用户宣告了PWB的发布，用户可以向Ted Dolotta索要相关的论文。下一期(第2卷第9号)宣告“用户组的会员现在超过了250”。1977年11月的《;login:》包含了连个声明，一个是Joseph F. Ossanna的死讯，一个是即将发布的《C语言编程》。

1978年3月，Lew Law宣布4卷本PWD手册已经由哈佛科技中心出版(定价26.5美金)。接下来的8月，Ferentz使用《;login:》的整个首页版面宣布纯Unix信息的《贝尔系统杂志》特刊。

bug修复，硬件增强，新的软件，优雅的hack。这就是USENIX，AUUG，UKUUG所致力于奉献的。我最喜爱的贡献来自Tom Ferrin(来自旧金山的加利福尼亚大学)，他用硬件改动解决了一个软件问题。他两页的信件(1977年9月9日)，一些代码还有一张配图发表在《;login:》上:

> PDP-11/45和PDP-11/70的内存管理单员比其他PDP-11机器提供几种高级的功能。这些强大的特性可以将程序分为指令段和数据段...
>
> PDP-11的4个指令简化了程序在不同寻址模式下指令/数据内存区域的操作，这就是“移动到/从之前指令/数据内存空间”(mtpi，mfpi，mtpd，mfpd)...
>
> 因为DEC想要“保护私有程序的完整，“mfpi”指令在一个状态字等于17 的程序(例如当前和之前的模式都是USER)内执行时会出现问题，这实际上故意阻止了C语言子例程“nargs.s”的功能...
>
> 这个问题有几种解决方案...
>
> 4.修改硬件让它工作的更“正确”
>
> 在和DEC的销售代表打了几通电话和看了几个小时的微码后，我们找到了一个简单的修改方案可以让PDP-11/70的cpu允许“mfpi”指令正常运行。这个方案会花费有经验人员15分钟的时间，你需要剪断一个引脚并添加一个到M8138-YA内存管理电路板的跳线...

Tom Ferrin的图如下所示:



译者注

- Commentatary on the UNIX

伯克利Unix 第一部分

伯克利加利福尼亚大学的Robert Fabry教授在1973年工作于SOSP程序。它对Thompson的报告有着非常深刻的印象。当他返回科瑞教学楼的时候，Fabry的第一份工作是尝试让计算机科学系，统计系，还有数学系合资购买一台PDP-11/45。他向Thompson订购了纸带，在1974年由毕业生Keith Standiford安装了Unix。就像Kirk McKusick所说的，Ken Thompson参与此前所有安装，但是不包含这一个，因为“有几个奇怪的系统崩溃需要他去解决”。Thompson会打电话给在机房里的Keith Standiford，电话随后换成了300波特率声学调制解调器，Thompson通过会通过电话线“在新泽西远程调试bug”。

接下来出现了一个问题，因为数学系和统计系想使用DEC的RSTS系统。因此一天的时间被分成了几部分:RSTS运行16小时后，Unix接着运行剩下的8小时。当时在那里接触到Unix的学生Eric Allman说:

> 我那时学习操作系统课程，他们在6400上使用一个叫做玩具系统的操作系统。但是他们想去使用11/40上面的Unix，我们一天只能使用8小时，而且每天使用的时间都不固定。我努力回忆:我看了手册，我记得没有理解为什么你要使用**echo**命令，非常奇怪。当然我现在知道了。第4版相当古怪，只有研究人员才会喜欢上这个系统。它很慢，只有很少的工具。接下来我们听说了Ingres项目。
>
> Michael Stonebraker和Eugene Wong两位教授的Ingres数据库项目是第一批移植到Unix的软件。由于他们不满意分配的时间，1974年春天他们自己买了PDP-11/40。但自此之后，学生们就没有足够的时间在11/45上面上机。1974年6月Stonebraker和Fabry两位教授计划为计算机科学系再购买两台教学用11/45。他们在1975年早期得到资金，刚好那时DEC宣布了11/79机型，他们觉得这个型号非常合适，于是资金汇集起来。11/70机器在秋天运了过来，正赶上Ken Thompson来这里以访问教授的身份进行一年的学术休假。Thompson，Bob Kridle，还有Jeff Schriebman在新买的11/70上面安装了第6版Unix。

还有一个意外结果是“50 fixes”纸带。Thompson告诉我:

> 我们意识到的第一件事情是外界都在使用Unix发布版(V4，V5，V6，V7)而我们没有使用。我们的观点是连续的。我们在某个时间点拥有过V5，但因为不断地放进模型到处很容易就过期了。
>
> 在第6版以后，我准备回到伯克利教一年书。我将一个系统的纸带放在一起拿着。因为它几乎是一个发布版，我在V6上做了一个**diff**。去伯克利的路上，我在厄本那香槟分校停了下来，照看Greg Chesson完成他的博士学位。我留下diff的纸带在那里并且告诉他我会留意它是否传播开(我想我可能还把它给了Katz)。(O'brien提到diff纸带一定是来时Chesson)。
>
> 我感到它非常重要，因为我发现了非常多的多道程序(异步)bug需要修改。发布的版本没有提供更新。在用户非公开交流的时候对此有很很多咒骂。整个情形看起来已经无法继续，我感到我应该去做些什么。

Lou Katz的说法有些不一样:

> 大量的bug修正被搜集起来，相比于一次发布一个，Ken制作了一个合集纸带(“50 fixes”)。虽然我没有特别记住哪一个，但是这里面有一些修正非常重要。我怀疑这其中有很大一部分是贝尔实验室意外的人制作的。Ken试图将它寄出去，但是律师一次又一次的拖延下去。
>
> 最后，在完全厌烦下，有人“在山岳大道发现了一个存有修复的纸带”(贝尔实验室位于新泽西莫里山600山岳大道)。
>
> 当律师发现这件事情，他们打电话给每一个授权单位试图问出他们是怎么拿到纸带的，并且恐吓他们如果不销毁纸带的话将会有可怕的后果。我猜没有一个人会真的告诉他们纸带是怎么来的(我就没有)。这只是律师证明他们的存在并试图杀死Unix众多行为的第一次。不得不说为Unix工作的律师要远多于技术人员。同时授权条款也开始更改，费用开始迅速的上涨。

也是在那个秋天两个毕业生去了伯克利校园:Chuck Haley和Bill Joy。他们对新系统着了迷，并且和Ken Thompson一起开发Pascal系统。Haley和Joy将Pascal系统提高以至于成为学生编程系统的选择。但是当ADM-3屏幕终端代替了33型号的电传打字终端，Hale和Joy对**ed**行编辑器感到不满。他们使用了一个叫做**em**的编辑器，它是伦敦皇后玛莉学院的Coulouris开发的，然后开发了每次一行编辑器**ex**。1973年Coulouris成为英国第一个接受Unix的人，他告诉我:

> 我在1975年秋天在圣母玛莉学院开发了**em**，它可以更有效的利用我们刚拿到的vdu终端。我们有在图形显示器上开发应用程序的经验，不论是鼠标定位还是触摸板定位(这个经验来自William Newman，他花了一年半的时间和我们一起为Utah和Xerox PARC工作)。我猜测Unix终端输入的的“原始”模式在那个时候完全没有被利用，使用它可以即时看到编辑结果，让人们更方便的编辑文本。 我们将它用于我们的单用户图形系统，这让我开发了**em**。

> 换句话说，**em**的含义是给人用的编辑器，我是在Ken Thompson访问圣母玛莉学院以后给它命名的。Ken看了它然后说“很棒，我看到过类似这样的编辑器，但是我并不觉得我需要它们，我在编辑文件时不想看到它们的状态”。
>
> 因为Ken表现出来的态度，贝尔实验室可能从来没有考虑过开发一个屏幕编辑器，这也使得他们很多年的时间里都没有合适的终端。他们继续使用了TTYs和其他打印终端很长时间，当他们给Unix组里每一个人都买了一台Tektronix 4014s屏幕，一种巨大的贮存管显示器。你不能在贮存管显示器上运行屏幕编辑器，因为它们的图像不能更新！于是这让其他人成了Unix屏幕编辑器的先驱，并且这才只是开始，我们为此持续工作了很多年。
>
> 然后我在1976年夏天作为访问者去了伯克利大学的计算机科学部门。我在一个堆满了Unix电传打字终端的房子里工作。我将**em**存储在DEC纸带上带到那里并且在我使用机器前安装了它(虽然**em**是为vdus设计的，但是它的单字符交互仍然可以在TTYs里使用，在每次交互后都会将当前行重新打印，慢的让人心碎)。
>
> 有一天，我坐在旁边终端机上的人是完全疯狂的黑客/博士/学生Bill Joy，他告诉我他写了一个Pascal编译器。我给他看了**em**，然后他说“这很好，使用这个系统的人可能会对它感兴趣”。他将我介绍给了其他人。他们有几台PDP-11s通过9600波特率电路支持着满满几个房间的vdu终端，这真是一个可以让**em**发挥作用的环境。
>
> 我解释**em**是**ed**的一个扩展，它为单行编辑提供了按键级别的交互，即时在屏幕上显示正在编辑的文本行(一种单行文本屏幕编辑器)。我解释这是通过将终端设置成为“原始”模式来做到输入单个字符的时候就可以在屏幕上看到——在1976年这是一个怪异的程序。
>
> 系统支持人员Jeff Schriebman曾经说过类似这样的话:“那是非常好的东西，但是如果让我们所有用户都用上这个软件，它在原始模式运行(每次有按键被按下都要进行处理)的开销会耗光cpu”。
>
> 我更为这种交互感到苦恼，想着“我想我开发了一个不切实际的编辑器，它的运行成本是如此之高——它可以在我们的小型系统里运行，但是它在伯克利的Unix环境下就无法使用”。
>
> 尽管如此，Bill和系统支持小组的人还是从我这拿走一份源码的拷贝看看他们是不是能用它。我之后去了东海岸大约一周的时间。当我回来的时候，我发现Bill已经将我的代码作为起点开始向着**ex**和之后的**vi**这个漫长的方向开发，并且这个新的编辑器已经安装到了服务器上——当然它仍然需要“原始”模式，但是很明显它带来的好处远大于开销。
>
> 看到可能是**em**提醒了Bill Joy和BSD的人们Unix上屏幕行编辑器的可能性。如果我没有开发**em**或者没有带着它去访问伯克利，早期版本的BSD Unix可能不会包含一个屏幕编辑器。这可以当创意反向跨越大西洋的一个小例子。
>
> 不幸的是，**vi**没有继承到**em**中的一些人机接口理论，在圣母玛莉学院我们已经断定模态交互是一个糟糕的主意，并且我已经做了很多事情来确保在**em**的交互中每一个键都不会有重复的含义。这个理论被**vi**的插入和编辑模式严重的破坏了。

> 因此我们在圣母玛莉学院从来没有用过**vi**。作为替代，一个叫Richard Bornat的同事和我的博士生Harold Thimbleby基于听来的交互理论设计了一个全屏编辑器，这个编辑器叫做**ded**(display editor)。它除了需要几个合理的功能键，几乎就是无模式的。它在70年代末期和80年代早期的欧洲被广泛使用，但是后来被**vi**和**emacs**逐渐取代。

同一时间的另一个地方，康奈尔大学被称为“JCL黑客”的Kirk McKusick，经过在特拉华州立大学学习的同学介绍接触到了Unix。“他像我展示了你该如何在上面玩游戏”，McKusick告诉我

> 在我1976年去伯克利之前我真的没有了解过Unix。我在春天去那里去寻找研究生项目，那时他们正处于春假，所以并没有很多人。但是当我在机房里见到Bill Joy时他正忙着研究一些东西，他说“Hi，我是Bill Joy，这是我们工作的内容——一个运行在Unix上的Pascal系统”。我说“你能用Unix做什么？”。因此他回答“你可以编辑文件，编译文件，你还可以玩象棋。让我帮你登陆进去”。因此我我在那个惬意的下午一直在玩象棋。然后我确认我喜欢伯克利，我来对了地方。

当Thompson在1976年夏末返回贝尔实验室的时候，Haley和Joy的兴趣变成了内核。让Schriebman惊喜的是他们安装了“50 fixes”纸带。学习了如何维护源码后，他们开始建议提高改善。

就在同时，Pascal编译器的新闻开始到处传播(1978年)，Joy开始制作伯克利软件发行版(BSD)。1978年3月它通过一封信提供给Tom Ferrin。“授权”写在信纸的另一面。Tom在3月13日签收了这封邮件。这份磁带包括:

1. Unix Pascal系统
2. Ex文本编辑器

... 作者是

1. W.N.Joy，S.L.Graham，C.B.Haley，K.Thompson
2. W.N.Joy

说明页伴随着授权许可

> 此发行版是标准的“tp”格式，800分辨率磁带。一个1200脚卷轴是最低也是最好的大学。

它的定价是50美金。

这是一个太长的故事，我不得不就此打断，因为我认为Unix的前十年是典型的代表，使它成为一个如此流行的操作系统。

贝尔实验室创建了一些东西，它被以源码方式发布。一个英国的用户基于它创建了一些东西。另一个加利福尼亚的用户同时完善了原始和英国的版本。它按照成本发布给社区。完善版本又包含进了下一个贝尔实验室发布本。

专利和授权根本没法控制着一切，并且系统总是越来越好，使用范围越来越广。

Bill Joy就像是发行版的秘书，在1978年发出了30份“自由”的BSD拷贝。随着一批支持光标定位的ADM-3a型号终端到来，使得Bill可以开发vi(visual editor)。但是这让他去做了其他的事情:优化几个不同终端的代码。Joy决定通过使用一种解释程序重绘屏幕来加强屏幕管理。这个解释程序是被终端的特性驱动的——**termcap**诞生了。

1978年中期已经做完了足够多的事情(Pascal系统更加强健，它可以运行在PDP-11/34s，**vi**和**termcap**也都可以)，第2版BSD发行版被放在磁带上。Bill Joy回复电话，将发新版放在一起，将用户反馈搜集到系统。他也买了接近75份第2版BSD。(PDP-11最新的Unix版本是1.10.1，来自1989年USENIX协会。它大约有80Mb，售价200美元。真实一笔大交易)。

Paula Hawthorn向我回忆这段时间:

> Kashtan刚刚发布了Unix和VMS的对比结果，我已经为我的论文完成了很多工作，所以那应该是发生在77年到78年的冬天。一群EECS的学生在周末去越野滑雪。Bill Joy，Mike UBell和我还有其他一些人就在那里。我们坐在我们租的公寓的厨房地板上(因为要谈论工作，所以从客厅客厅来到厨房)。Bill读了Kashtan更好的新能数据，Bill和Mike读了Unix的源码，我解释了磁盘和缓冲在其他操作系统是如何工作的。我的硕士性能测量在一台运行Kronos的CDC 6600上测试的，我的博士研究是测试Ingres的性能，Ingres的性能非常依赖逻辑上的连续数据是否可以在物理上顺序读取。Unix需要相等的盘区。哦，兄弟，没人想听到这个，但是我测量个性能，Kashtan也测量过。因此我们翻开Unix的代码，找找哪里可以让程序更快一些。这是Unix成为高性能系统的开始...

Mike Ubell(他后来写了历史)说:

> 我记得Bill做其他事情时花了很多时间看代码。他在多底层的cpu优化，减少指令。

Ubell也对我说了shell历史的实现机制:

> Bill在**csh**下实现了一个历史机制。我爱他因为我那臭名昭著的总是拼错单词并且ADM3a上不能复制粘贴。我想他是把历史数据写到了一个文件里，但他并不喜欢这种实现方式，于是j就删除了它。失去一个生产力工具，我猜测它使用shell已经使用了的了内存分析架构。Bill喜欢这种实现方式并且采用了它。我做的那个shell在伯克利持续用了几个月，直到Bill做出新的版本。

C shell的man帮助页(在“历史记录”中)仍然描绘了Ubell的拼写问题:

> 历史记录允许你使用之前在命令行输入的命令。这简化了拼写纠正和重复输入复杂的命令或参数。

Ubell的经历也非常吻合Eric Allman说的:一个通用的软件设计规则是你应该写你想用的程序。

早期版本和**PDP**

Unix版本	PDP型号
1st	PDP-7
3rd	PDP-11; 11/20
•	
•	PDP-11/45
•	
6th	PDP-11/70
7th	PDP-11, Interdata 8/32

译者注

- editor history

V7

Unix第7版在1979年6月由贝尔实验室发布，它有几个主要的提升:它可以使用更大的文件系统，不限制用户的数量，提高了可靠性。Steve Johnson将第7版称之为“第一个可移植的Unix”。这个版本具有极大的影响力因为它增加了很多新的命令。让我列举一部分:

命令	
at	awk
calendar	cb
cd	cpio
cu	deroff
expr	f77
find	lex
lint	m4
make	refer
sed	tail
tar	touch
uucp	uux
系统调用	
ioctl	
子例程	
malloc	stdio
string	
游戏	
backgammon	

awk(Aho Weinberger Kernighan)，**lint**(Johnson)，**make**(Feldman)，以及**uucp**(Lesk)这几个就足够了，但是第7版远比这更多。

第7版的编程手册增加到了400页，还有两卷400页的附加补充。这个Unix中包含了一个完整的Kernighan和Richie C语言编译器，一个更加成熟的shell sh(Bourne Shell)，Dick Haight的find，cpio和expr，还有很多的包含头文件。

这些所有的放在一起，第7版Unix有了一个缺点:它的性能比大部分的第6版要低一些，特别是这些都已经经过“调优”。人们开始为改善这个问题而工作。伯克利的Bill Joy修改了VAX-11/780机器上Unix文件系统的的数据块大小，他的实现被Jeff Schriebman在1980年1月移植到PDP-11/70上面(他离开伯克利去了UniSoft)。1979年12月，那时还在伯克利的Ed Gould将缓冲区移出内核寻址空间。Joy修改了VAX上的*stdio*库，旧金山加利福尼亚大学的Tom Ferrin又将这个改动移植到了PDP-11。霍姆德尔的Tom London优化了用户空间到内核空间的字符移动。Ferrin还写了一个动态单一总线分配方案。新南威尔士大学的John Lions提议一个新的目录路径名命名规则，新南威尔士大学还提供了一个处理表格搜索的新代码。RAND公司的Bruce Borden提供了symorder程序。Ferrin还重写了 `copyseg()` 和 `clearseg` 的部分代码。所有这些来自社区的优化提升被收录到一个PDP-11发行版:2.8.1 BSD。1982年1月Ferrin在加利福尼亚州圣塔莫尼卡举办的USENIX会议上宣布了这个BSD的新版本。用户们因此大幅提高了第7版Unix的性能。

我在这里花费这么多的篇幅是因为第7版是来自工业和学术界，来自美国和澳大利亚。并且所有这些都包含进将来的BSD和AT&T Unix发布版。

第7版同时也在移植方面有所增强:32的XENIX2是第一个可以在8086芯片上运行的Unix(XENIX1是基于第6版)。第7版也可以运行在Z8000和68000芯片上。32V版本(来自Holmdel)则增强了3BSD和它所有的后续版本。

32位Unix的进化故事是Steve Johnson告诉我的:

> 我知道的第一个完整移植的Unix是普林斯顿大学的学生做的。他们移植了很大一部分Unix命令和系统的接口shell到IBM 360上。我想是TSS系统。他们令人惊奇的让Unix在那个系统上运行。他们使用了我们的知识和IBM C语言编译器，这是一件诱人的事情。那个夏天我们雇佣了这些学生中的一位，Tom Lyon，他移植第6版Unix所有的命令到Amdahl。
>
> 我记得当我们决定移植Unix。我们选择了Interdata。这是一段有趣的经历因为IBM将为我们提供2年半的360机器，但是Dennis不喜欢IBM的channel程序，他试着自己重写这些东西。

Marc Donner向我解释说:

> IBM 360是一台多处理器机器。I/O操作不直接在CPU和设备间直接进行，而是在由一个叫做“channels”的附属处理器来控制。channel处理器是一个特定用途的芯片，它知道怎样和不同的处理器通信(在“channel程序”的控制下写入机器码)...它通过向主存写入数据并给CPU一个中断来传送数据，从CPU接受数据也是通过类似的计算...
>
> 在早期channel是很简单的，并且相比channel程序，channel命令更方便一些。但是随着设备越来越复杂，需要更越来越多的处理，channel的编程工具也变得越来越复杂。
>
> 我不认为曾经有更高级的channel编程工具，结果就是它的处理方式变得极端复杂晦涩，只有很少几个人能搞处理好。

回到Johnson:

> IBM意识到我们比Interdata做的更好。还有一个很强大的竞争对手是DEC 20，它是一个26位机器，Dennis对它的评价是如果有人发明了10路纸带驱动器，他就会将Unix移植到DEC 20。基于8位字节写36位数据的问题是这远远超过了Unix文件系统所能处理的极限...
>
> Interdata是我使用的最后一台带有核心的机器——一个绕线的铁圈。Interdata后来被卖给Perkin-Elmer，我想他们又再把它卖给了Concurrent。
>
> 有一天我们收到了一封来自大利亚伍伦贡的信件，这是以我我们从来没有听说过的地方。那里的几个家伙说他们已经移植了Unix并且发来了方法。他们使用一种非常不同的方式让系统工作，一种自上而下的做法。他们在系统之上实现了一个中间层，由这个中间层来实现Unix系统调用，然后逐渐的将中间层向下移动并最终让系统内核看起来像Unix内核。这是一种非常有意思的方式，因为它可以在机器上非常快速的实现。但是总的来说，我们其实用了差不多的总时间。

我指出伍伦贡大学在那时是刚刚建立的。Johnson笑了起来。

> 好吧，你可以想象得到，这当然是我收到过的最让人吃惊的一封信。我们认为自己在科学上有了新的突破，但是有些我们从来没有听过的大学突然跨过半个地球写信过来说“我们觉得你们会对我们做的东西感兴趣”。
>
> 关于Interdata一个有趣的小插曲——有大量的问题，其中有些非常奇怪的问题我从来没有遇到过——关系到硬件分页。在Unix第6版使用返回值为-1的指针代表错误的做法已经非常普遍。一个常见的编程错误是当返回指针的时候忘了检查错误并且继续使用指针。在Interdata上，如果你试图访问位于-1位置的数据，你会引起两个错误:一个是对齐错误，因为-1不知字对齐的，另一个是内存错误，因为你超出了内存的地址范围。而且这又引发了一系列的时钟问题，因为微码引擎需要几个时钟周期来处理错误，当它处理其中一个错误时，另一个错误又发送过来，这时微码引擎就会被被阻塞。这不仅会丢失当前程序执行到的位置，阻塞的引擎还会在它恢复同步以前错过执行接下来的几个指令。
>
> 那时硬件固有的问题。任何用户程序都可以将-1放入寄存器然后让机器陷入无法预料的状态。此时我们只能通过切断电源后再通电让机器恢复正常。所以Dennis和我一起谈了谈Interdata。
>
> 我说:“看，贝尔实验室正用着这么多这个机器(那时有几百台)，你拿到了一个32位机器，我们有我们的Unix系统和应用程序。我么所要做的就是针对硬件的一些修改，之后伙计们就可以开始出售Unix系统了”。然后他们说:“不”。
>
> 所以Dennis和我完成了一个Unix移植，然后决定再也不去做下一个。这在漫长的历史中还是第一次。

时间到了1979年1月，第7版《UNIX编程手册》的封面上既没写Dennis的名字也没写Ken的名字。上面写着:

> UNIX™ TIME-SHARING SYSTEM

Unix已经差不多有10年的历史了。

有趣的是，它已经进入中学校园。1979年1月Brian Harvey去了波士顿郊外的林肯萨德布瑞中学，“去建立一个计算机部门”。它和学校董事会商讨设备合作，并劝说DEC给中学一个客观的折扣。结果是20万美元的设备只收了5万美金。无论如何，安装过程“经过了几个地方的计算机科学家共同的努力...因为我们的PDP-11有不能处理原始的第7版Unix磁盘”。Harvey帮助15岁大的学生么解决了这个以及其他很多的问题，他们开始写一些有用的程序。后来他们成为Unix社区的一部分，开始分享他们的软件。Harvey说:

> 几个月后我们的系统传播了出去。当我们接到来自另一个计算机中心的电话时我和孩子们简直太兴奋了。他们买了Unix系统并且刚好遇到了和我们一样的问题，我们按照他们的配置寄去了一个定制后的启动纸带。从那时起，我们开始通过USENIX贡献学生写的软件...
>
> 中学生开始和成人程序员一样从Unix中获益:它的灵活，可以修改它工作方式的能力，还有支持它的广泛的工具。

第7版包含了比本地宝库更多的“好东西”——无论程序员存咋哪里。第8版移植了vi(Bill Joy)，curses(Ken Arnold)，和来自BSD的termcap(Joy)。Arnold的curses是有一个游戏影响软件开发的例子:curses是Arnold为了玩rogue更容易而写的一个屏幕处理和优化程序。顺便说一句，Arnold还写了forune和其他许多有用并且有意思的程序。但是就像实用的第7版，也非常让人厌烦。这远不止是代码的问题，就像Andy Tanenbaum说的:

> 当AT&T发布了第7版，人们开始意识到Unix是一个非常有价值的商业产品，所以为了避免危及商业机密，第7版的授权阻止在校学生以外的人们获得源码。很多大学简单的取消了UNIX的学习，转而只教理论部分。

Tanenbaum的解决方案是“从头写一个和UNIX兼容的新操作系统”，里面“没有AT&T的一行代码”。Tanenbaum将这个新系统命名为MINIX。

伯克利Unix 第二部分

1978年早期，Richard Fateman教授开始寻找一个有更大内存寻址空间的机器，这样就能在PDP-10上继续做他的Macsyma。Fateman解释说:

> MIT有一个MAC项目，它的程序叫做MACSYMA。为了和VAX只最初只能PDP-10上运行的那个同名项目区分开，我就叫它VAX/Macsyma，有时也会说成vaxima。MIT后来将Macsyma卖了出去，从它的名字也可以看出来，它被卖给了Symbolics公司。接下来所有的后续版本都不被Symbolics所控制，也不被再接下来的所有者马萨诸塞州阿灵顿的Macsyma公司控制，这些版本被称为和Macsyma不一样的名字:
>
> DOE-Macsyma —— (能源部的版本) Paramax —— (Paradigm Associates) Maxims —— (德克萨斯州立大学Bill Schelter移植的通用Lisp版本) Aljabr —— (Jim O'Dell开发的Macintosh版本)
>
> 所有这些分支版本在本质上都和vaxima差不多，虽然它们的支持者都声称自己选择了最好的。

不管怎么说，新的DEC VAX 11/780机器看起来符合Fateman的需求，而且Fateman关于NSF的提议和其他13位教员的建议一起得到了部门基金。最初VAX运行的是DEC的VMS操作系统。

但是部门开始变得习惯使用Unix，并且Fabry得到了一份Unix 32V移植到VAX的拷贝，这是霍姆德尔贝尔实验室的John Reiser和Tom London开发的。移植项目的主管是Charlie Roberts，这些是他回忆的那段故事:

> 77年到78年DEC提前宣布了VAX。Dennis和Ken还有Steve和DEC公司变得相当疏远。他们觉得自己的工作已经安装在很多的机器上，DEC仍然拒绝支持Unix并且继续开发VMS。所以当DEC提供给他们一台VAX的时候，他们拒绝了(Doug McIlroy告诉我他们拒绝的另一个原因是“VAX使用让人非常厌烦的复杂指令集，这于Ken和Dennis的价值观非常不一致”)。之后Dennis和Steve为移植到Interdata上工作，Steve称它为“Intersnial”。
>
> 因此DEC去霍姆德尔找到了我们。显然我们是后备人选。Tom London和John Reiser对此很有兴趣，Ken Swanson也是，我们在78年早期拿到了VAX机器。我不做任何技术工作。实际上，我花费了大量的时间和精力让管理层同意我们做这个。你看，这不是研究。最后管理层给了我们一点时间。大概用了3个月的时间我的小组将第7版Unix一直到了VAX。我们在1月得到机器，然后在4月让它运行起来，8月的时候它就可以真正的工作了。接下来的事情大家都知道了，大约有6个大学给我打了电话——布朗大学，加州大学，伯克利大学，滑铁卢大学，其他的我已经记不起来了。所以我为了专利和授权去见Roy Lipton和Al Arms，经过不断地反复，他们决定我们可以将给大学作为研究途径使用，Al将会和大学签订一份“特别研究协议”。
>
> 我在一些会议上见了Fabry，我那时去伯克利和Ferrari，Emmannuel Blum和Bill Joy做论文和演讲。所以在贝尔实验室11区管理部门的祝福下，我将32V给了伯克利，时间是1978年11月。

批准了Roberts合作的管理层是C.C. Cutler，R.W. Lucky，和W.S. Boyle。1127管理层——Doug McIlroy，Sam Morgan，以及Bob Prim也批准了。

1979年32V成为了3BSD。Kirk McKusick向Peter Collinson描述了故事情节:

> 当Ken离开这里回贝尔实验室去的时候，他留下了一份Unix系统。Bill和Chuck Hale接管了这个系统的开发。那时候他们做的是像升级到第6版和集成实验室的“50 fixes”纸带之类的事情。后来他们又将它升级到第7版。
>
> Bill又在实用工具方面做了很多工作。他厌倦了shll，将John Mashey在PWD中关于shell的一些点子添加到**csh**中。Bill也对**ed**很有些意见，所以决定开发一个扩展的面向行的编辑器**ex**。然后我们将我们的几个终端从没有鼠标定位的ADM-3s升级到具备此功能的ADM-3a。只有Bill和那个房间里的两个人真的喜欢用他们。他开始了**vi**的开发，因此他总是有理由将正在使用新终端的人赶走。最初的**termcp**也被发明出来，这样就可以在不同的ADM终端上运行一样的编辑器。
>
> 关于**vi**为什么没有多窗口模式还有一个寓言故事。Bill编写调试代码终于使它勉强可以工作，但这时候突然有一个磁盘坏掉了。这个有窗口版本的磁盘没有被即时备份，备份好的只是没有窗口的代码，他也不想再去从写开发一遍。
>
> 最初的伯克利软件发行版，叫做BSD而不是1BSD，它包含了Pascal系统和**ex**。Bill除了是一个天才的程序员，还是一个天才的市场销售。他做了很多通俗易懂的演讲告诉人们这个东西是多么的棒，Pascal解释器是如和做到只花费C语言编译器十分之一的时间，并且它还有错误纠正功能，所以它更时候学生因为可以给他们更多的额支持。他卖出了大

概35份BSD纸带，每份定价是“只是成本价”的35美元(原价50美金)。BSD发行版的关键是Bill，他是黑客，是电话客服，是纸带绕带员，他将地址写在标签上，将它们寄出去，他做了所有的事情。

接下来就是2BSD，它在大约一年以后出现，包含了Pascal系统，**ex**，**vi**，和**csh**。他单枪匹马做了这些并且在那年卖出了100份...

2BSD将是基于V7的，因为我们是在1978年拿到了预发布的UNIX/32V。V7由此一直到了VAX。它没有分页和其他什么特点，它只是一个在VAX上运行的基于交换的传统UNIX。我们有一个编号非常小的VAX-11/780，一个个位数的编号。

Bill要做的第一件事就是让32V在VAX上跑起来，这需要你的机器使用和发行版纸带有完全相同的硬件，那时还没有auto-configuration的概念。你不得不做到硬件和他们完全一致，不能是丝毫的差异。正好我们有两盘磁盘，正好它们都是RP06s。

下一件要做的事情是移植使用工具。我曾经为Pascal解释器做过一些事情，添加一些新的功能和修改一些不太对的地方。Bill对我说“你为什么不把解释器移植过来?”。

Pascal解释器是用汇编写的，“移植它非常简单，它只有很少的操作步骤并且汇编语言看起来几乎都一样”。他做了一点小例子向我证明这事做起来很简单。我花了一个月的时间做移植工作，最后我放弃了汇编，用C语言重写了解释器。

还有一个问题，VAX只有2MB的内存...这台机器的硬件有个巨大的进步，你可以在它上面使用虚拟内存。Babaoglu和Domenico Ferrari一起为VAX设计了一个分页系统。Bill在听说了一些他们的工作后就帮他将这个功能运行起来。经过78年12月到79年七月的假期，他们让VAX-11/780实现了独立，这样它就可以启动不同的系统了。你登陆进去后，有一个提示语告诉你“虚拟VAX Unix”，然后系统要卡顿一会，你可以先做些别的工作。经过5分钟的等待，系统会提示它现在是32V。

它起初并不是很稳定，但是最后他们让它变得很健壮，它成了一个基础系统。

Bill决定停止做只有使用工具的发行版，转而去做包完善的系统。他创建了3BSD发行版，这是一个完善的系统，带有基于虚拟内存的内核和移植过来的使用工具。这也是一个完善的可启动系统，在纸带的起始部分是一个启动块，你可以把纸带放在原始的硬件上就可以启动。他还为这个发行版做了其他所有的系统。

那时我开始将越来越多的系统花费在内核问题上。我记得有一件事情让我参与进来。Bill试着调试虚拟内存系统，他将这个巨大的程序打印出来，足足有一英尺半那么厚。他把代码搬过来放在桌上，以他特有的热情开始翻看代码。他一遍又一遍的翻看，“在这页有一个bug导致了竞争状态，我给找出问题的人20美元”。

我从来没有呀看过内核的源代码，但是我上过操作系统课程，所以我知道竞争状态的出现是因为你在某些地方使用了上下文切换。我看着这些写的很标准的代码，在在这页上一共有两个子例程的调用。我指着这两个调用问是不是其中的一个做了上下文切换。“是

> 的！就是它”，他跑出房间，留下一片片飞在空中的纸张。和Bill在一个办公室工作是一件困难的事情，你很难不被他的热情所感染。
>
> 3BSD系统发布出去后受到了很多关注，因为不少大学都购买了VAX机器。你如果想要一个分页系统，那么只有3BSD和VMS两个选择。很多人既想用Unix又想要分页的功能，所以3BSD在VAX早期的购买者中非常流行。

关于这个还有个小故事，McKusick告诉我:

> 债务被拖欠了一年又一年。Bill去了Sun公司，他成了一个百万富翁。四五年前，当UNIX国际组织成立的时候，他们试着做一些宣传，于是去了一个很有影响力的新闻发布会UniForum宣布了他们的存在(1989年1月在旧金山)。他们决定向Unix商业中的杰出人物们颁发一个产业奖项，记者们就可以看到这些产业明星。他们提名了几个人——毫无悬念的包含Dennis和Bill，我也被包含进名单。在他们向我们讨论这件事之前，我们赚的很少。我告诉他们我是如何参与到Unix的故事，他们都大笑起来。但是在有像Dennis Ritchie这样重要人员在场的会议上，他们做了什么？他们直接讲了那个故事。那时Bill正在台上拿着他的奖牌。他们说:“Kirk说他从来没有付钱”。就在我走到Bill Joy后面的时候，他从口袋里掏出50美分，摇晃着他们说‘这就是你想要的二十美元！’。后来我问他“Bill，他们之前告诉过你会发生这些事情吗”，他说“当然没有，如果他们警告过我，那我就会准备20美元”。

Keith Bostic通过NET2的发布将伯克利Unix开发的历史描绘在一起，这是他的大纲:

> *a)* 早期在伯克利运行的系统是Unix第5版，第6版和第7版，这些都是来自贝尔实验室。这些系统在伯克利被计算机中心和EECS部门不停的修改。这些修改包括:
>
> *b)* 伯克利Pascal——大约1977年伯克利为在PDP-11上运行的第6版Unix写的软件
>
> *c)* 2BSD——1978年开始伯克利为第6版和第7版发行版写的软件。新的软件和修改一次又一次的添加进来，包含mail，more，csh，ex，以及vi，Pascal。2BSD使用了第6版和第7版不同的授权。
>
> *d)* 3BSD，1979年晚期——这是第一个针对VAX的伯克利发行版。它基于32V，带有伯克利使用工具和来自2BSD的修改，加上Bill Joy和几个毕业上完成的一个虚拟内存系统。

从1969年开始到第6版Unix这段时间只有一个解释器，就是Shell。73年4月18日出版的入门手册一共只有三页。随着Bournes的Shell集成到第7版Unix和Joy的C Shell集成到2BSD，程序员们有了一个选择。1980年中期AT&T贝尔实验室的David Korn发明了Korn shell，给了程序员所期望的更多选择。1986年它被包含在“实验工具箱”中，1989年正式成为SVR4发行版的一部分(1988年911月16日发布)。通常Bournes shell向上兼容Korn shell。还有很多其他的shell，但是sh，csh和ksh是最常用的。

Unix可以运行在不止一台机器上的特点引起了美国国防高级项目局(DARPA)的主意。DARPA的几个合作单位使用了不同硬件和各种操作系统，而且软件都是互相不兼容的，做不到软件互换。DARPA想要一个统一的环境，所以会有很多的交换。McKusick告诉我“很显然他们会

选择VAX，接下来的问题是他们的操作系统会选择VMS还是Unix”。他告诉Collinson:

> Bill设法说服DARPA Unix才是更好的选择，因为它可以移植到其他系统上去，这样他们就不会被绑定在VAX上面。Bob Fabry在Bill从技术上说服的同时也在政治方面说服他们。
>
> 他们设法和DARPA签订一个大订单，将Unix进行性能增强并添加一些DARPA所需要的基础功能。他们保证这些新功能会马上做好。于是4BSD在DARPA的订单到达伯克利之后的两个月就发布了...这时的时间是1980年晚期。

据Collinson所说，“4BSD之所以能这么流行，是因为VAX-11/750推出后，4BSD是唯一能在它上面运行的Unix系统”。McKusick继续说:

> DEC的Armando Stettner设法安排Bill飞到美纳德去看一下刚刚建造好还没有对外公布的750s，它们被命名为Comet。Bill和Armando一起写代码确保4BSD可以在这些Comet上运行，所以虽然我们没有一台机器，但是4BSD在发布的时候就以可以在750s上运行而声名远播。实际上后来750成了部门工作的主力机型。因为你用大概15万美元的价格就能买到一台。
>
> 4.1BSD是性能巅峰之作，在某种意义上4.0BSD也很好但是4.1针对750做了针对性的优化。4.1BSD系统被贝尔实验室拿回来成为第8版Unix，然后是第9版和第10版。也是它让我们被大家所抱怨，因为Rob Pike加上了**cat -v**。1983年6月在多伦多举办的USENIX会议上，Pike谈论了“像癌症一样扩散”和“cat -v是因为考虑到伤害”，这让他收到大家的抱怨。在过去的10年，内核的膨胀越来越严重:第7版内核的大小是40KB，Ritchie告诉我第10版内核的大小差不多是150KB。Armando Stettner向我指出如果要在4.1BSD和第7版Unix间做选择，没人想回到第7版。
>
> DARPA给的资金到位了，这意味着Fabry开始像给员工一样开始为Bill提供薪水。还有其他一些职位，接听电话，制作纸带，签订协议。Bill终于可以专心做他想做的事情了。项目扩张是的Bill可以雇佣技术人员，于是他找来了Michael Toy和后来的Sam Leffler。
>
> 4.1和4.2BSD之间的工作是DARPA加入进来的真正原因，他们想要一些重要的增强，像可以使用超过磁盘5%带宽的文件系统。他们想要一个基于TCP/IP的网络。
>
> 最初原型是BBN做出来的，他们将它交给了伯克利。它只能在750上运行，速度上限56KB/s，但CPU却被占用了100%。Bill一拿到它马上就开始修改它的代码。56KB的速度对BBN来说够用了，因为他们主干网的速度就是56KB/s，但是在互联网上这就太慢了。Bill删掉代码重新编写使它的速度达到700KB/s。
>
> 这引起了BBN的注意，因为他们给出的是“完成”版本，但是Bill却不接受它。之后几年关于谁的版本应该加入的系统中区有不少的冲突，伯克利版最终胜出。Ritchie告诉我当他在DARPA委员会时，这是一个很重要的问题。“每隔半年我们都会说，使用BBN的版本。这时候Bill就会说不”。
>
> 4.2BSD开始支持第三方组件，这是一个新的虚拟内存系统，允许共享读写段数据和其他一些事情。
>
> 系统中还有一些东西不是因为Bill觉得他们应该在里面，而是因为掌握权力的委员会要求这样。通过messages传送文件描述符就是其中一个例子。

随着VAX的出现伯克利还发生了一些事情:成为sendmail的计算机邮件。这是那时的一个毕业生Eric Allman写的，这是他的故事。

> Eric Schmidt当时正在工作于BerkNet，BerkNet通过一条9600波特率的tty线连接，它是一个批处理系统，就像**uucp**。那段时间贝尔实验室和大学的关系看起来有些问题，所以我们没有**uucp**。当然很快这问题就解决了。当时产业界决定“喔，我们不想让大学拥有那个因为这样我们就会失去它”。然后大学就自己开发了一个，所以产业界觉得“天哪，我们想让自己的产品成为标准，最好还是给它们吧”。BerkNet在那时是很好的网络，但它有一些考虑不周的问题，比如它的主机名是单个字符。我们觉得ARPANET现在只有26台主机，将来是不可能超过256台的！C是cory，E是ernie kovacs，I和J则是Ingres 11/70和VAX。计算机中心得到了A到F，代表着Unix A到Unix F，他们认为自己的取名策略非常有创造性。
>
> 很多办公室里有终端的人们开始使用email和更多的服务。我们让这个网络支持电子邮件和简单的文件传送，甚至还支持了简单的远程命令执行。我对简单的定义是开发时间大概一小时。接下来的压力来自EECS部门的教授们，他们想连接ARPANET。我认为ARPANET之所以仍然在使用11/40是因为他们的链接接口不能连接11/70。最后的结果是我们不想给每个人账号，但最终还是被强迫着做了。我们修复了一些问题，这样ARPANET的邮件不用绕其他路线就可以发送到BerkNet以外。你要了解，那时除了在ARPANET上，人们不在电子邮件的前面放头部信息。虽然RFC822上明确有规定，但是执行的并不好。
>
> 那个时候ernie kovacs的**uucp**，学校的BerkNet，还有ARPANET，每一个都使用不同的邮件标准。如果你想发一封电子邮件到ARAPNET和uucs，你只能将它发送两次。很明显这不是一件好事情。
>
> 我困惑了很长时间，不能找到修复这个问题的好方案。一天我坐在格伦街的家里对自己说“这太愚蠢了，我要自己写代码解决它”。然后我就写了那个程序，那个下午我选定了它配置表的样子，然后它的形式就很清楚了。那个用于**delivermail**的配置表使用字符写的。它说“这里有一个‘@’，它一定是ARPANET的，要用uucp。这个是BerkNet”。它的创意是每个网络都有它特有的标志字符...不管怎么说，**delivermail**包含到了4.0和4.1的纸带。但是显然睡着时间发展它的配置显得不够完善，当我们有了更多的机器的时候它就显得不够通用了。
>
> 因此我开始修订**delivermail**。Bill Joy嘲笑了我，他说“它没有去发送邮件，它只不过是将邮件交给另一个代理”。所以我把它改成**sendmail**。所以**sendmail**应该算是**delivermail**的第2版或者第3版。
>
> 还有一点要指出的是这期间ARPANET正在从NCP过度到TCP。那时一段相当痛苦的时期，短短几年经历了从mit-xx到xx.ARPA再到xx@mit.edu。修改**sendmail**并没有花费我太多的时间，它对我来说还是比较容易的。更幸运的是我可以一直跟进最新的修改...我正在做一个大的修订，因为RFC1123的“Host Requirement”...

ARPANET在1968年作为一个小的研究性实验开始并在1969年并入ARPA。它论证了跨网络长途通信的能力并最终成为美国的骨干网络，国际互联网，抱恨NSFNET和其他网络。它同时也证明了一个通用协议的重要性。每个ARPANET的节点都使用了一个叫做信息接口处理的通

信子网(IMP)。这些最初是Honeywell硬件，后来是BBN硬件。主机间的通信是以BBN1822协议为基础。最初的主机到主机通讯协议是NCP(Network Control Protocol)。

到了1976年已经很明显可以看出将会有大量的局域网络需要互相连接。这导致了TCP/IP协议的开发。顺便说一下，第一个公开的数据传送网络是加拿大在1976年建成的Datapac网络。感谢1974年Vint Cerf和Robert Kahn智慧和远见，以及BBN基金会实现了协议，还有伯克利将协议放入Unix软件发行版，DARPA才能连接了大约90%放入科研大学社区。

这是真正的协同:TCP/IP允许更好的连接，它融入了伯克利Unix。Unix越来越流行，越来越互相连接。我认为完全可以说如果没有Unix，TCP/IP在1994年就不会拥有全世界的3000万。事实上这些用户中的很大一部分使用的是MS-DOS，Apple-OS，或者VMS。

现在，世界互联网提供了非常多的服务，但是气功只有少量是“必不可少的”:电子邮件，文件传输，还有远程登录是最重要的。这些都是来自Unix。

在返回到McKusick的故事前，先让我继续列完Bostic的年表，这将会让读者了解到详细的伯克利版本。

> *e)* 4BSD，1980你那10月——这个发新版包含的性能提升包括使用了虚拟的存的更快的文件系统，任务控制，可靠的信号，自动重启，**delivermail**程序，还要Franz的Lisp系统。
>
> *f)* 4.1BSD，1981年6月——4BSD在性能方面有很多提升，支持一个新的VAX模型和自动配置
>
> *g)* 4.1a——4.1a是一个包含TCP/IP和套接字的测试版本。它卖给了ARPANET的很多需要网络的工作站。
>
> *h)* 4.1b——这是一个测试版本，据我所知只有伯克利内部在使用它。它包含了新的更快的文件系统，使用了聚类获得更高的性能。包含了新的网络代码。
>
> *i)* 4.1c，1982年到1983年——这是4.1BSD和4.2BSD之间的一个测试版，一共发给了大约100个工作站。它包含很多4.2BSD中除了新的信号机制意外的绝大多数特性。
>
> *j)* 4.2BSD，1983年9月——一个主要的系统修订版本，包含网络(TCP/IP和一个通用框架)，一个新特性的更快的文件系统，一个重新设计的更加稳定的系统该接口，还有新的信号机制。
>
> *k)* 4.3BSD，1986年6月——4.3BSD优化和固定了很多4.2的新特性，并且提供了更好的性能。新的特性包括XNS网络，目录名缓存，以及一个互联网名字服务器。
>
> *l)* 4.3-Tahoe，1988年6月——这个中间发布的版本添加了对CCI Power 6的支持，4BSD提供的第一个非VAX的支持。它包含了几个内部内核工具，包括内存分配，内核调试，和磁盘标签的支持。TCP算法被LBL的Van Jacobson大幅提升。

> *m)* BSD Networking Release 1，1988年11月1日——第一个网络发布版是那时伯克利系统的一个子集，和4.3-Tahoe而非常相似。它包内核网络部分移植的源码和文档，C库和实用程序。还包含一个使用了新版本rlogin(网络登陆)的登陆程序。它可以不经任何授权直接使用，并且可以通过匿名FTP重新发布。源码中包含了一份伯克利版权和一个允许重新发布的说明。
>
> *n)* 4.3-Reno，1990年6月——这是一个包含了很多4.4BSD新特性的测试版本。最主要的改动是添加了支持多文件系统的*vnode*框架，一个网络文件系统(NFS)实现，一些网络框架方面的修改，还有对OSI网络的支持。许多的内核代码，库代码，新添加以及重写的代码都包含了一个伯克利版权声明，它和Networking Release 1类似的允许重新发布。对于VAX和Tahoe，4.3-Reno支持基于摩托罗拉680x0处理器的HewlettPackard 9000/300系列机器。
>
> *o)* BSD Networking Release 2，1991年6月——第二个网络发布版包含了更多的额网络代码，但是像第一个网路发布版一样不需要授权。它包含了比4.3-Reno高级很多的当时伯克利系统的一个子集。新的特性包括一个从卡内基梅隆大学的Mach系统得到的新的虚拟内存系统，这个系统是犹他州立大学移植的，并且Bill Jolitz将它移植到了Intel 386/486系统。

另外我还要把4.4加入这个列表:它是在1993年6月发布的。

Kirk向Collinson指出有些东西是最近才做出来的。

> 现在在4.4BSD中内建了Portals，是Jan-Simon Pendry将它放了进去(见28章)。原始的架构文档中也描述了**mmap**系统调用，许多其他厂商抢在伯克利之前实现了它。在Sun要实现它时我们扮演了这样一个角色，我们和他们争论每个标志应该代表什么，每个接口应该是什么样子的。我们可以说我们扮演了一个角色，即便我们在实现方面晚于其他厂家。
>
> Bill做的第一件事情是socket接口和TCP/IP的代码。它日程上的下一件事是文件系统。在春天的时候，他设计了一个初稿，其中从Mike Powell在Cray上开发的DEMOS上借鉴了很多东西。我仍然以研究生的身份在工作，我的导师是Sue Graham。作为一个研究生你在学校的时候只有一半的自由时间，到了暑假你可以全职工作三个月，这样就可以为秋天积攒足够的学费。6月的时候你可以拿到四分之三薪水的支票——两周的半职和两周的全职。不管怎么说，当我在6月月末拿到薪水的时候，只有四分之一的数额。有些家伙搞错了工资单，他们不知道我暑假会在那里，可能因为我去年没在那里。我到处走想看看到底发生了什么。工资单需要我的导师需要给他们一个资助码，我去告诉Sue我需要一个新的资助码，然后我发现她忘了更新资助码。
>
> 很显然应该更新资助码，但是这会花去几个月的时间。同时，我将面对一个没有薪水的暑假。我知道Bill已经拿到了新的资助码，于是我走到楼下问他我在暑假是否依然可以享有DARPA的授权，他能否给我一些事情做让我可以写在纸上。我们都明白我应该为我的论文做些事情，他同意并且建议我可以去实现已经设计好的文件系统的原型。虽然我什么都不知道，但是他完全知道将会发生什么。
>
> 我用整个暑假做出了Bill设计的文件系统原型，让它可以在用户空间访问一个原始的磁盘分区。到了秋天，我有了它运行的一些演示例子。就像你猜到的，Bill建议说把它放进内核应该不会太困难，然后我就做了。Bill说“这非常棒，人们会很喜欢它的，但是它还缺少**dump**，**restore**，**fsck**...”。一件事情引起一连串的事情。18个月后，我们有了更快的文件系统。它很复杂，代码量是原来的三倍...
>
> Sam Leffler在一个网络公司工作过，有不少网络经验。他能在socket接口上做一些重要的修改让它适应高速线缆网络，比如使用Bill做的一系列链接队列。这对于让重度使用下的代码更加强健非常重要。
>
> 我们最后的测试是**ucbvax**，那时邮件中心和ARPANET**uucp**连接的机器。它太慢了以至于挤压了很多要处理的事情。很难看到在这台机器上可以看到50到60个连接。在今天看来这不算是一个大交易但是在那时却是很困难的事情，对文件系统各网络的代码都是很大的考验。
>
> 然后4.2BSD就发行了。
>
> 实际上Bill在4.2BSD发行前不久离开了伯克利。他将当时的系统带到了Sun。Sun的系统在4.2BSD官方发布后可以毫无障碍的升级到这个版本。

目前为止这写还不是伯克利的全部故事。让我们倒回去一点，听听Sam Leffler讲述他的故事。

> 我75年去了克利夫兰的凯斯西储大学。我在那里的主要工作是合并BS/MS程序，按计划应该会在3年后毕业。按计划我会和Bill Shannon一起做这个项目。我们将会建立一个Unix系统，拿到硬件然后移植软件系统。基于Z8000或者其他的什么硬件。但是做出所有这些计划的起者退出了。一切都搞砸了，因此我暂停了一段时间教学工作，直到我可以发现下一个可以做的论题。所以我直到到1980年才毕业，虽然我在79年就做完了所有事情。我在1980年去了伯克利，但是那时Bill去了哥伦比亚的USENIX狐疑，下一年我们一起去了多伦多。那时我们的工作是叠加层。在PDP-11上的Unix第7版做透明叠加层，我们在多伦多遇到了Bill Joy，他当时也在做这方面的工作。我仍然记得我们和Bill关于这些工作的谈话。我们解决了一些他们还没有解决的问题，他带走了这些解决方案。
>
> Shannon和我毕业了，我们以做了很多项目结束了学校生活。凯斯西储大学试图建立一个学校范围的网络，他们尽量提前建成时间，这完全是一场政治活动。委员会和所有的部门一致同意他们需要DEC的设备，并且一些满是官僚主义的人过来说“你可以买Harris”。所以我们以很多的额Harris设备结尾，这是你所能见到的最奇怪的机器。它不能运行Unix以及其他任何系统，这些设备从来没有工作过。Shannon和我不顾一切的寻找一个有经费的项目，“我们想要脱身，所以我们同意去做一个移植系统到这个机器的项目。这是一台奇怪的机器，因此他做了操作系统的工作，而我做了编译器和所有编程语言的工作。我们之间就有了这样诡异的关系，呃，一种共生关系，如果想毕业补习互相依靠对方。我需要操作系统演示我的编译器，而他需要我的编译器让系统运行起来”。
>
> 结果就是，如你所知，Harris是一台非常快速的机器。但是它不能按字节寻址，并且它的字长是24位。我们简单的使用了PDP-11/70的机器，然后将它改成11/43所需的分页。我们联系了Bill Joy，并在他完成之前先做完了工作。
>
> Bill和我终于毕业了，接下来我们开始寻找一份工作。我们有非常相似的资格证书。我只在两个地方看工作机会，我想住在波士顿或者加利福尼亚附近。所以我去了硅谷的一些公司和梅里马克的DEC面试。DEC雇佣了Bill，因此我去了加利福尼亚。

Armando Stettner那时正好在DEC工作，从他的描述角度看看这件事事情:

> Bill Munson和我只有一个招聘名额。我记得我们在特拉华州立大学的USENIX面试，这些家伙正在玩排球。虽然我们只有一个空缺，但是Sam和Bill我们都想要。我们真的很难选择所以我们用排球的比赛结果来做出选择。我们看他们打球，一个是擅长单打独斗光芒四射的健将，一个是普通的团队球员。随意我们决定录用Shannon，那个团队球员。Sam去了Sytek。

Shannon在DEC只待了一年多一点的时间，他后来去硅谷加入了Bill Joy在Sun的团队。Leffler在硅谷的一个公司找了工作，在那里他待了一年时间。但是因为他们都运行VAX并且Leffler可以随时见到Joy，那时他正在为4.0版本做测试。所以当Joy失去了一些员工并且有足够的钱时，Leffler“去了伯克利。先是做4.0，然后是4.1.这就是我去伯克利的过程”。

1980年夏天一个伯克利的研究生John Forderero写了一程序，这个程序可以检测到邮件的到达状态。它会告诉用户你有了一封新的邮件。在那时，Heidi Stettner在开始研究生学业之前在凯文斯教学楼工作。Heidi会带着她的狗去教师和办公室，这是一只非常友好的狗，有很多

学生喜欢在走廊上把球扔出去让它去捡回来。甚至在公告板上也有它和毕业生们的照片:题目是正在工作的狗博士。John决定用这只狗的名字命名他的程序:Biff。根据Heidi所说，John和Bill Joy后来花了很多时间试着去试着解释**biff**这个名字的含义:Be notified if mail arrives。Bill在1993年8月去世，当时年龄是15岁，它在编译课上获得了一个B的成绩。Heidi说Biff咬了邮递员是一个下流的谣言。

Kirk McKusick在1981年夏天加入了这个项目。Leffler从1981年秋天开始在伯克利，大概有6个月的时间Joy很少在东海岸，但是直到1983年4.2发售时，名义上他仍是这个小组的一员。在这段时间里，DARPA的伯克利合约监督人Duane Adams设立了一个“指导委员会”，成员有伯克利的Bob Fabry，Bill Joy，Sam Leffler，Bolt和Beranek以及Newman的Alan Nemeth，Bob Gurwitz，斯坦福的Keith Lantz，麻省理工的Bert Halsted，卡内基梅隆的Rick Rashid，信息科学协会的Dan Lynch，DARPA的Adams，Bob Baker，UCLA的Jerry Popek。从1984年开始，每半年召开一次的委员会会议被一系列的年度研讨会代替。

Gurwitz早期实现的一套TCP/IP协议让BBN非常不满意，于是找Joy来继续修改。1981Joy同时也在做一个进程间通信的项目。Leffler和他一起做一个多网路同时使用的协议。**rcp**，**rsh**，**rlogin**，**rwho**都是作为临时的工具简单的写出来的。他们包含在1982年4月发布的4.1a中。了1982年6月McKusick做出了新的高速文件系统并将它集成到4.1a中。这个系统成为了4.1b，在夏天和秋天用在了操作系统课程上。Mike Karels告诉我:

> Roberts Henry和Bob Kridle离开这里去了mt Xinu，而Kirk和其他所有写出这部分代码的家伙们在上操作系统课程。我不确定那个系统是不是运行过，但是我们谈论了很多关于它的东西。

当Joy离开时，Leffler接手了他的工作。但是他没有被指定为Joy的职位，所有感到有些怠慢。Leffler离开去了卢卡斯影业，最初是以兼职身份在做。4.2之所以可以完成时因为曾经参与了2.9BSD发布的Mike Karels做完了接下来的工作。

4.2BSD是一个巨大的成功，就像McKusick指出的:“4.2卖出的拷贝(前18个月)比之前卖出的伯克利所有软件加起来还要多”。有几个操作系统公司是以4.2为基础开发的:DEC的Ultrix和Sun的post-UniSoft OS是其中最为有名的。虽然如此，还是有很多关于4.2BSD的抱怨，Mike Karels花费了第一年大部分的时间来优化系统。

让他来告诉我们那时的故事:

> 我只去了两个生物研究生学校:伯克利和威斯康星州立大学。作为一个没有毕业的学生，我只是很少量的接触过计算机。我的一个编程课程是使用PL/I。当我到了伯克利，他们有这方面的课程，你不得不在一个实验室花上第一个学期，然后在另一个实验室又花上一个学期，然后你才可以选择一个导师...在我第二个学期我加入一个做细菌遗传的小组。在实验结尾的时候这个小组在一台运行第6版Unix的PDP-11/40上面做数据收集的工作。这看起来很有些意思，我想我在做我的细菌学实验的时候，或许可以写一些程序。好吧，主管计算机的博士后离开了，这个伙计表现出兴趣并且真的写了一个程序，接手了他的工作。我们在计算机中心有一个叫Bill Jolitz的学生，他写了一些软件并且维护这些东西。每当我们得到一个新的终端或者磁盘，我们都要去学习怎么使用这些东西。
>
> 有一次一个磁盘崩溃让我们丢失很很多为第6版Unix定制的东西。那时第7版已经出来了，Bill试着把它从地质调查局带过来，那时他在那里也有一份工作。所以相比起重新构建第6版，我们选择了第7版。如果Bill没在那里，又或者设备驱动没能工作，再假如已经是截止时间，我会代替Bill做这些事情。这样我通过实践学到了很多东西。但当我开始工作于后来的2.8BSD时，是和Bob Kridle以及其他很多我不认识的人一起做的。
>
> 就这样我在自己学业上花费的时间越来越少，越来越多的时间被用在了计算机上。然后在CSRG有了一个工作机会。我在8月1日开始去那里工作。在月末的时候Sam也去了那里做兼职，后来又里来了。我猜我是在1983年4.2发布的前两个月开始工作的。然后Bill离开了——不管怎么说，在最后一年他已经很少出现在那里了。但是，你知道，大概一年前我去了CSRG，我发表了自己的第一篇论文，Sam过来说“为什么你不来为我工作，这样你就可以在一台真正的机器上干活了”，那时VAX正在替换11/40。当我在VAX上工作时这既有趣又让很让人兴奋。在一个新的机器上debug还是有点困难。

VAX是一台真正的机器，但随着时间流逝，出现了一个又一个真正的机器。

商业Unix

关于厂商如何(失败地)拥抱Unix有大量的东西可以写。Doug McIlroy对我评论说:

> IBM和BTL很早就在做TSS/Unix的融合工作，但是当Amdahl进入董事会时这项工作变得毫无影响。DEC试着无视Unix，但是Armando Stettner做出了让步，开始做Unix。HP和DEC，IBM一样有自己的操作系统，也像日本的DEC一样开始满怀热情的使用了Unix。Sun公司是Unix和硬件发展的地方，是它发展了BSD。

考虑需要更多的乐趣，这些公司对于操作系统开发的贡献是如此之小(除Sun以外)，我不会给他们留太多的篇幅。

第一个在商业上支持Unix的公司是1977年的Interactive System。第二年Heinz Lycklama和Interactive System的创始人Peter Weiner在圣塔莫尼卡合作。Lycklama刚刚写了LSX，这是一个针对LSI-11微处理器的Unix版本(LSI-11本质上是一台使用了PDP-11处理器，用于卡内基梅隆实验性的的松散耦合多处理器)。Interactive的产品叫做IS/1，可以在大多数的PDP-11上运行。直到几年之前，当它成为一个芯片公司，Interactive仍然是最著名的移植公司之一。有趣的是，之前Weiner还在RAND公司的时候，他是第一个获得商业Unix授权的人。

第一个商业Unix“克隆”是克隆自Whitesmiths的Idris。Whitesmiths是之前贝尔研究员P.J. Plauger的作品。Plauger说Idris是一个艺术方面的巴比伦人上帝。Whitesmiths是一个软件公司，相比之下Idris可以称为“blacksmith”。Plauger毕业于普林斯顿大学(比Stu Feldman早了一年)，然后1969年又去密歇根州继续攻读了物理学博士学位。他告诉我:

> 在那时研究并不是很有趣。所以当他们给我选择事，我选了霍姆德尔的一个小组，它的工作内容是将电话系统数字化。这是一场不朽的战斗，因为每一个结局方案都意味着通过电话线发送比现在更多的信息，这是远超过电话公司乐意支付的...我在贝尔公司的各个实验室做了几年的销售，其中他们将我借给莫里山。因为空间太狭小，我去了Brian Kernighan隔壁的办公室。在那个大厅里的人有Doug McIlroy和Dennis Ritchie以及Al Aho。Brian和我都非常喜爱喝咖啡，我们每天都会聊上几次。
>
> 我们都很同情那时可怕的编程风格。我们坐下来用了大概三个月的时间写了后来的《编程风格要素》。做完了这些，我们得到了贝尔实验室的一些支持。这激发我们又去写了《软件工具》。但是在参于那个项目几个月后，我最终离开了贝尔实验室。在那里让我感到没有未来，并且我最好在他们让我离开之前就主动离开。我去Yourdon公司找了一份工作...
>
> 《软件工具》是我这一生中做过的最难的事情，比我的博士毕业论文还要难，更加让我不满意。所以我很高兴它最后终于出版了...Brian和我接下来很想Gilbert和Sullivan:我们一起做的事情比我们单独做的要好...
>
> 经过几年到全世界旅行演讲的生活后，我感到我想回到编程生活。Ed Yourdon有一个机会得到了一份商业C语言编译器的合同，我说服他让我来做这个。我昼夜不停的工作了一周的时间，它不是世界上最好的编译器，但它是绝对够用的。过去这段时间，我学到了市场在那里，因为人们会跑过来对我说“我们不想要整个的Unix，我们只想要C语言”。
>
> Ed明白讨论班上的商业，但是不懂软件商业。我决定退出这里开办我自己的公司...当我准备离开的时候，我的妻子建议我到处走走，去参加一些我不敢兴趣的讨论班。因此我在特拉华州建立了一个公司——除我之外还有两个合伙人，我做了多数工作。作为一个三人公司，我们在78年8月1日开工。我们坐下来重头写了一个C语言编译器——我的第3个C语言编译器。我花费了很大力气不去回顾我在贝尔实验室和Yourdon公司的经历。

我主意到对我来说最有趣的事情就是Whitesmiths的C语言编译器是如此的不协调，Plauger笑着说。

> 不不，事后看起来它更要糟糕一些。实际上，我们这几天对于C语言库没有明确的概念，对于在PDP-11意外的C语言没有明确的概念。我诚实的将我最好的技法用在它上面。如果我犯了一些错误，一定是当一些C语言的方言出现时，我应该离开那些人群，使用我自己的C语言库...
>
> 无论如何，我像上瘾一样的写着，到了11月末尾的时候我们有了一个编译器。我们发布了一个简短的公司成立声明——宣告我们将会做一个自定义的C语言编译器。很走运地我们在建立公司一周后接到了来自费城一个公司的电话，他们说“我们看到你们正在做C编译器，我们正想买一个，你们能到费城来一趟吗”。我说“不，我太忙了。你要是想谈谈，可以到这里来”。然后他们真的来了！他们派了3个人来纽约，我们坐在西49街狭小的房间内...我告诉他们我们的计划，然后他们说“我们想要你做出来的东西，我们想要你说的接下来三年所有要做的的东西”。因此我说“好啊，你可以按月支付给我们报酬，那会让生活轻松很多”。然后他们照做了！在公司开张3，4个月后，我们就有了来自Fisher and Porter公司的一个大单...那年末尾我们给了他们一个PDP-11编译器，79年中期我们给了他们一个8080编译器，那年末尾又给了一个VAX编译器。在1980年中期我们给了一个68000编译器。
>
> 同时我们也在做Idris。我们写了相当多的代码。Whitesmiths像野草一样成长起来。我们以我节省下来的2.5万美金起步，6个月后我们有了明确的现金流。那时一段让人陶醉的时间，充满了乐趣。在81年末之前一切事情都很美好。

1982年Whitesmiths经历了一番努力挣扎最后失去了PC上C语言编译器的市场。Plauger坚持Whitesmiths移植到1988年，然后把它卖给了Intermetrics。他说:

> 我花了十年时间学习如何运营一家公司，当我擅长做这件事的时候，我却发现我不想去做了。我过去的四年我给了自己。
>
> 但是我真的想去售卖Unix，我花了一些时间和贝尔的律师讨论。后来我意识到因为和解协议，他们什么都不会去做。市场是一个肮脏的世界，以个危险的肮脏世界。我不得不决定是要继续战斗着去卖Unix还是咬牙停下来。这不是一个简单的决定。我的目标是填补低端市场，但是我不做市场或者让人们去买它...

非常巧合的是，当Greg Rose在1979年离开UNSW，他去的第一个公司得到了Whitesmiths的C编译器授权，当它破产时，Rose和他的一些朋友继续从事Whitesmiths相关的生意。

从某些方面说，Wollongong Group是澳大利亚的第一个Unix公司。它的兴起和衰败是一个非常奇怪的故事，很值得一读。当Rich Miller和他的开发组在伍伦贡大学将Unix移植到Interdata的时候，大学没有纸带驱动器。Rose告诉我:

> 他们非常想要一个纸带驱动器。所以Juris Reinfelds有了一个计划，他去了新泽西。我想他是去参加一个会议。他真的想他们有一些东西——Interdata上面的Unix，Perkin-Elmer的那个阶段，他试着将它卖掉。他真的去和一个公司的人去谈，这个家伙说“你想卖多少价格？”。Juris说他们想要一个纸带驱动器，甚至二手的也没关系。所以那个家伙将Juris交给一下下属，下属听完之后说了“不”。Juris在回澳大利亚的路上非常不开心。他在洛杉矶机场的一个酒吧将他的麻烦讲给了一些 博士。那个叫Dan的家伙说“一个二手的纸带驱动器大概价格是多少？”。Juris说差不多是1.4万美元。Dan说“我会买下你的版权”。然后他们做了一个交易，他写了一个支票并且建立了Wollongong Group。后来他买断了Rich Miller和Juris。Dan开始到处售卖软件版权并且向Perkin-Elmer收取10万美金的维护费。这就是公司成立的经过。

比起其他的故事并不让人更觉得奇怪。但人们奇怪是Interdata/Perkin-Elmer的什么人显示拒绝了Ritchie和Johnson然后是Reinfelds。Greg继续shuo:

> 大概正好是墨尔本大学得到Perkin-Elmer Unix的时候，然后Robert Elz就参与了进来。几年来每一次的AUUG会议上Elz都会发表一篇关于tty驱动器或者其他东西的论文。这其中有些东西是非常重要的，后来又被伯克利Unix吸收回去。我想Elz在79到80年之间应该是在伯克利。

在其他事情之外，Elz还写了BSD磁盘配额(允许管理员控制给每一个用户分配的磁盘空间)和自动配置文件(找出并配置连接到处理器上的外围设备)。Elz的论文在1984年6月盐湖城的USENIX会议上发表。

最后一个早期的Unix企业我想说一下加拿大的一个公司:人类计算资源公司(HCR)。HCR是由来自多伦多大学的人创建的，它的故事也是从那里开始。David Tilbrook是多伦多大学的学生，HCR的创始人，还是几个EUUG和USENIX会议的主席。他向我会议那时的学生组:

> 那是一个非常奇特的用户组，是Ron Baecker的动态图形项目的一部分，唯一做图形的人是Bill Reeves。Duff正在去和Steve Cook一起做可计算。我正在和John Lipson一起做代数。Lipson说为什么我不去用我所有的真实经历来让我的大学生活更加有趣，然后努力为博士生活而工作。所以我决定离开这里去做计算机图形。我参加了一个图形课程。所以我们组成了这个计算机用户组，所有的成员是:我，Bill Reeves，Tom Duff，Martin Tuori，Mike Tilson，Tom Horsley。Rob Pike大概在一年后加入进来。这是一个在很多方面都相当有名气的组。
>
> 当时是75年，CACM的文章已经出来了。我们在370上使用一个叫Zap的软件包做图形工作。Zap是一个用FORTRAN写的软件。你可以用一个库使用PL/I语言编程，然后将它们和FROTRAN子例程链接在一起，然后就可以在数控绘图机上画出图片。我们也使用GT-40来绘图。在毕业课程上我和Baecker一起做项目，我们做了一个动画短片，Tilbrook做了一个排序算法。
>
> 就在同时我还为多伦多公共图书馆的一个合同工作，这需要和UTLAS(多伦多大学自动图书馆系统)接触。因此我可以访问XGP打印机——它的序列号是3。这是我第一次接触可以同时使用不同字体同时使用大小写的打印机。我也看了打印书脊和数目以及卡片文件的问题...
>
> 因此我们是在一个动态图形项目，Tuori和我将要做一个排版工具包——那时什么东西都没有。没有质量产品。所以Martin和我通过在大学和中央图书馆的支持下拼凑了一个使用计算机排版的工作间。我们发出传单，然后收到了大约20个回应...这让我加入了Baecker做动画电影的组。我们三个人是国家研究理事会学者，所以Baecker有足够的资金，于是我们买了PDP-11/45。Reeves做了通用软件包，我继续搞动态图形的东西。就是在那时我提出了动态光标。虽然Reeves，Duff和我都是诵读困难者，有一个风格上的巨大变化。不管怎么说，我喜欢Xerox Sigma 7，并且我用它来写NEWSWHOLE。我每月登陆时间有300小时。
>
> 《The Globe && Mail》的David Slocum来到工作间和我们讨论分页和排版的困难。计算机分页看起来对于这个杂志是一个很好的话题，因此我去《The Globe && Mail》兼职工作，我写了NEWSWHOLE。我带着这些建立了HCR。
>
> 1975年12月，我去见Baecker对他说“我能带来《The Globe && Mail》的合同”。有一个大问题是“这是运行在Three Rivers Graphic Wonder上的Unix系统，我们该怎么发布它?”。(Three Rivers Graphic Wonder现在仍然在多伦多大学Henry Spencer的计算机实验室中)

HCR建立起来了，最初它是一个小规模的组。引用一下它最初的章程:

> 允许针对众多基于计算机的服务和产品进行广泛的研究和实验开发..计算机图形和交互计算，接口设计...一个例子是设计基于计算机的文本处理系统和页面布局系统...

组的成员有Tilbrook，Baecker，Les Mezei，还有Tom Horsley。Mike Tilson在大概两年后加入了他们。HCR也销售软件产品，包括XENIX，微软的第7版衍生产品。但是想要做他们想做的事情，他们需要一个商业Unix授权。Tilbrook告诉我:

> Bill说他曾经给AT&T里扶着这方面事情的Al Arms打电话，Bill告诉他我们想要购买Unix，那时是1776年，在电话上什么都没有做成。Al Arms给出了2万美金的价格，因为他认为这是一个很高的价格，这样几句没人再因为这个打扰他了。

当然，就像RAND公司的Peter Weiner曾经支付的，Tilbrook夸大了数额。虽然如此，HCR成为一个应盈利公司。Tilbrook继续为其他事情努力，Tilson陈威HCR的主席。HCR最后被Santa Cruz公司收购。

DEC

第二章的很大部分内容都是在说数字设备公司(DEC)，这是因为在普林斯顿大学做出成果前，Unix只能运行在DEC生产的机器上:PDP系列(特别是PDP-11)和后来的VAX系列。但是DEC公司自身是不提供官方的Unix支持的，他们有自己的系统:FOCAL-11，VMS，TENEX。本质上DEC在前20年都有一种被Armando Stettner称为NIH的文化——这里就不再细说了。上Unix是贝尔实验室产品的事实足够取消DEC的工程师们。

但是在1970年后期，有一个不断增长的认识是Unix没有离去，硬件工程师们也意识到乐于助人的Unix用户非常重要。Ritchie和Thompson曾经去DEC访问过几次。现在DEC设置了一个Unix支持组——“我们是同一类实验室”。Armando Stettner在1979年后期离开贝尔实验室去了DEC，他是Usenet的早期发起人，Ultrix的架构师(DEC版的4.2BSD)，同时也是Unix车牌的首位用户。他告诉我:

> 我将拥护Unix的氛围带到了DEC，否则DEC就不会有在纳舒厄的Unix工程师组(UEG)。我们在新罕布希尔州的梅里马克工作。我们将Spitbrook路称为“Saliva Creek”。有一次服务和支持部门的人们知道我们在那里，他们给我们打了电话。
>
> 但是当Bill Joy来访问的时候，我仍在做着32V。当我们从波士顿洛根机场到新罕布什尔的时候时间太晚了，唯一还开着的地方是Denny的办公室。我们那天晚上做的事情是描绘出我们将要做什么。我们想维护所有我们身边的软件，修正配置文件，然后把这些发布成一个发行版。我办开玩笑的说“为什么我们不把Unix移植到DEC 20系统上?”。Joy说“看，这需要10天的时间，但是我在这里只能呆7天”。后来证明Bill有10天的时间。接下来我们晚上在VMS工程师组的实验室做VMS组的原型。Bill做VAX 780，Bill Shannon做750，我做730。基本上，我们会一起做出某些东西，然后分头到个自的机器上调试它。我们有一个RP06磁盘，我们将它称为5BSD，因为我们确信下一个发布的将会是5BSD。
>
> 我们复制了一个磁盘，Bill拿着它回了加利福尼亚。Shannon和我将磁盘复制到了decvax——一台780机器，我们的主力系统。Bill Joy接下来几天给我们打电话说“嗨，如果我们发布另一个发行版将会有很多授权方面的麻烦。那么我们为什么不把它称之为4.1BSD?”。

10年后的DEC似乎是在反对Unix。Steve Johnson向我评论说:

> 当然，当VAX机型出现的时候，很明显DEC认为它是一台很棒的机器，至少在硬件方面是这样，Unix可以为他们的销售额做什么？但他们仍然没有赞助。

我问Armando关于事情的情况。

> 我来到以后，UEG做了很多的支持工作——我们写设备驱动和做一些底层工作:Unix的硬件依赖部分。伯克利专注于高层。知道我们的顾客会告诉他们的驻场工程师给我们打电话。我们开发了一个测试和一个语言所以我们可以和驻场的人们有关系。当支持Unix的时候有一个很大的新人问题，当一台机器不能运行DEC的软件时驻场工程师不知道该怎么做。账号管理和客户满意度永远是一个问题。
>
> Spitbrook对Unix总是充满了敌意，坦白说是对整个公司...工程部门和管理部门尤为严重。Dave Cutler是DEC工程部门的中坚力量，他抓住了某些东西作为折中——我想是VMS上的SRI软件包。Cutler把Munson叫来，记得在AT&T和大学，Unix是DEC的一个大问题。因此Cutler被Munson叫来，Munson说他会让我们一些人过去看一看。所以Bill Shannon和我去了那里，大概有20分钟的路程。我们去了那里，Cutler正在他的办公室里。Bill和我坐在一个终端前，任何事情都不能做。Cutler问它有什么问题，我说它不能工作了。Cutler说“好的，谢谢你们”。我们回到梅里马克，Munson把我们叫到办公室(他是资深管理人员)，问我们“你们去那个地狱做了些什么？Cutler给我打电话对我念叨说对你们的工程师深表遗憾并且再也不想在Spitbrook见到你们”。
>
> 我想Cutler的轻蔑影响了他此后的工作。在DEC内部的RISC战争中，Cutler在新的Prism架构上做了另一个操作系统而不是使用Unix。Cutler的系统不能移植，但是从文化上兼容VMS。我想NT中的许多东西可以追溯到Prism(Cutler在1983年去微软工作)。
>
> 我们有一个Unix产品经理，有一天我们在走廊里走着，我说“Bill，你知道我觉得现在是啥时候为VAX做一个真正的Unix产品，一个VAX Unix”。我做了一个计划，Bill把它拿给Munson看，对他说“让我们做出来看看”。基本上，我的计划是基于4.2BSD的，如果伯克利没有发布4.2，那就使用替补计划的4.1。我对于4.2很有信心，因为我们已经拿到了测试版本，并且我们已经在使用它了。但是我们需要一个意外处理方案，也就是4.1。尽管这样，那时Sun已经宣布在伯克利之前宣布了4.2——Bill Joy才刚拿着4.1c离开。我们发现4.1在decvax上非常稳定。我那时是技术经理。

1980年在旧金山的USENIX会议上，Bill Munson宣布UEG的目标是“启发DEC发现Unix和它的机会”。Armando有另外一种启发。

> 1982年1月我在东海湾的圣莫妮卡参加USENIX会议，当Bill Munson冲出来的时候我们正要宣布DEC有一个Unix项目。Slsen在我们的后面。显然他带着我寄给他的Unix车牌，他把车牌挂到一些人的胸口然后对Munson说“去搞一个Unix产品，把它做好！”。所以Munson红着研究去东海岸发出上那个公告。

Armando最为值得记忆的“露面”是在1983年圣地亚哥USENIX会议上，我问他这次会议的情况。

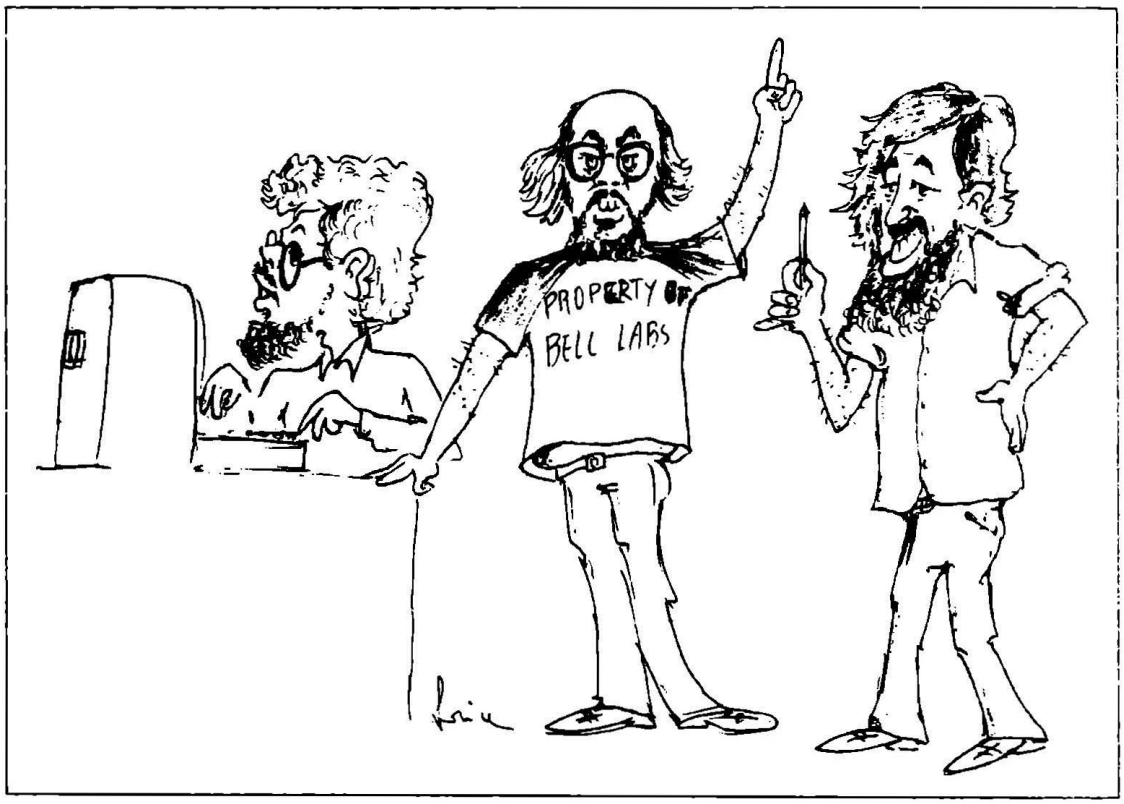
> Bill Shannon刚刚买了一个新的Datsun，一个280ZX，我们开车去汉堡王把它们取回来。Bill有一个“UNIX”的车牌，我们想为USENIX会议做一些事情...每个人都有自己的小伎俩。所以我们说“让我们做一个海报”，又有人说“好，让我们拿一台VAX和一台终端，在木头上将它们建立起来”。我不记得我们为什么要把VAX放到木头里面去。我建议去拿实验室的780，在它上面画一个挡风玻璃，放上去一些VW轮胎并把Shannon的车牌放在挡风玻璃下面，然后用这些制作了一张海报。要想得到做这些事情的权限非常复杂，所以我去和PR的人谈了谈，正好他有所有这些目录。我看着这些目录和已经有的车牌，一个点子就冒出来了。Munson给了我800美元，我制作了3000个绿色UNIX车牌。
>
> 我记得我站起来说我想宣布你现在可以从DEC拿到Unix授权。台下所有这些来自几个Unix OEM厂商的家伙们都震惊了。我扶起车牌，死一样寂静的房间里爆发出大笑和鼓掌的声音。从房间的后面传出来一个声音:“Armando，你能从这件事学到什么?”，是Dave Yost说的。后来我们收到了一封来自贝尔律师的关于商标的信件。但是他们又打电话给我告诉我们他们不得不发送那封邮件来保护商标，但是我们能不能也给他们寄一些车牌。于是我们就寄了。

多年来DEC和Ken Olsen一直忍受着Unix用户大量批评。1983年圣地亚哥的UNICOM上，Bill Munson说Olsen曾对他说“要澄清一下...DEC支持Unix!”。但是Olsen也曾在1988年说过“Unix是蛇油”。无论如何，Armando在DEC花了十多年的时间在DEC扮演“Unix传福音者”，感觉Olsen受到了不公平的对待。

> 事后想起来，我认为Ken有一个坏建议。Ken是那个说“做Unix”的家伙。Ken是那个说“做Pmax(MIPS和DECstation3100)，从别的公司买架构”的家伙。我认为赶走他的别人说的:人们想要它。你知道，那句“Unix是蛇油”的评论是断章取义。当然，这有消极的含义，但是Ken真正的意思是“每个人都在说你需要Unix”。每个人都在推销Unix。我能记得见到那些真正需要Unix的客户，但是他们自己也不知道是为什么。人们到处都在谈论。Ken的意思是“就像人们在一个世纪之前离不开蛇油一样，现在每个企业都需要Unix来做事情”。那句“Unix是蛇油”评论的语境是:做了一个类比。不幸的是，就像一些人说过的，Ken总是不了解他说的那些话会有什么影响。
>
> 但是根据当时在场的一些人的记录，是Ken Olsen向开放软件基金会争取Unix对抗他的副手。DEC的问题是所有运营这个公司的人们都是从工程师成长起来的，没有人有商业经验。就像任何很好的工程师，他们维护自己的产品和他们的合作伙伴。这就是为什么DEC的问题出现在80年代而不是90年代。
>
> DEC有自己一系列的成功:PDP-11和VAX，在它的竞争还比较小的时候。但是到了80年代中期工作站和RISC机器开始和DEC竞争，DEC却没有可以对抗的机器。另外，DEC开始试着离开它的大本营，离开科学-工程师-学术的社区，去争抢IBM的市场。Cutler曾经为这个新机器和新操作系统工作过。我们确信它不会做我们需要它做的事情。如果它成功了，如果可以给IBM的市场定价，它会比Unix市场昂贵太多，DEC将会在和IBM对抗的时候拱手将钱送上。另外，据我们从Sun的SAPRC得到的可靠消息，Prism机器发展的不够快。但是我们想让Cutler为VMS市场做他的系统，我们想在Unix市场之后做自己的系统。我们的系统是基于MIPS R2000 RISC芯片。Kne Olsen对这个相当感兴趣。

> Pmax使得公司两极分化。起初从东面回来的Unix组不想去做这个。我们(在帕洛阿尔托的)觉得DEC这么大已经足够去做和管理两个架构...人们开始讨论一个不能拥有自己架构的公司能否取得成功。
>
> 我们真的非常喜欢MIPS架构和R2000芯片。我们喜欢MIPS的家伙们。它有一个64位的计划。人么问的更多的是什么？和我们谈过的顾客都非常喜欢它！
>
> 如果DEC把它一般的能力花费在MIPS的关系和架构上，他们就会安全的继续做Prism，今天的产业界可能就不是现在这个样子了。
>
> 在Gordon Bell关停DECsystem-20的时候就应该想到这些。我觉得特有系统是叫做Jupiter。我认为是因为他的VAX没有一个强大的竞争对手来争夺所有这些TENEX/TOPS-20的客户。我相信Cutler在70年代的时候不想他的机器有一个竞争对手。他当然不想在80年代为不同的市场提供不同的系统。他真的太担心会杀死Unix。
>
> Ken在董事会的授权下，违抗所有工程管理层的一致劝告，让我们放弃Prism转而去做Pmax。还有一个限制:拒绝我们要求的1年时间，我们不得不在9个月的时间内完成这个奇迹。我们做出来了。《Sam Jose Mercury》说:“DEC震惊了硅谷”。《Upside》公布了一张图片，Ken从一个帽子里拉出一只兔子，图片的标题是:“Scotty，想看我从帽子里拉出另一只兔子吗?”。
>
> 但是DEC继续它的NIH姿态:测试芯片非常像Prism芯片。Ken(或者公司文化，他们有区别吗?)不知道该如何去和别的公司合作。因此他们从来没有为MIPS的未来做任何事情。甚至OSF和X/OPEN也成为问题。只是因为一个人的看法...

有趣的是，一旦Armando和其他人离开DEC，ULTRIX就不再继续:1990年，DEC仍然说“ULTRIX操作系统是一个源于4.2/4.3版本的伯克利软件发行版(BSD)驱动实现，它提供了许多AT&T的System V接口定义所规定的系统和库调用”。知道后来，DEC为了支持OSF/1完全砍掉了ULTRIX。



Bill Joy, Ken Thompson and Greg Chesson tutoring in Italy. Cartoon by an unknown student, courtesy of Ken Thompson.

法律 第二部分

1956年的和解协议对西部电气和AT&T造成了很大冲击。但是1974年11月20日美国政府针对AT&T，西部电气和贝尔实验室单独进行了一项反垄断行动，他们“声称被告在电话通讯服务和设备方面有着大范围的垄断...”(1982年地方法官Harold H. Greene说的。本章接下来说有的引用都来自Greene法官的意见)。

反垄断部门拆分审判将贝尔操作公司从AT&T拆分了出来，西部电气也被拆分。感谢AT&T的律师将“非常小的进度”做了差不多4年的时间，但是在1978年9月11日，“法院提出一个意见解决了所有问题，然后未完成的法律问题安排给未来的法院预审诉讼”。真正的审理开始于两年半以后的1981年1月15日(回忆起来，1978年华盛顿有一个民主审判部门。1981年1月20日Ronald Reagan举行了就职典礼后，民主审判部门的性质发生了戏剧性的改变)。

差不多一年以后，Greene法官面对政府要求的“延期最后审判”，时间推迟到了1982年1月8日。简单的说，这意味着审理部门想要拆分的企图将会导致1956年协议的“延期”，而不是诉讼的决议。1月11日另一个法官Vincent Biunno在简短的了解后，也同意了这个“延期”。第二天Greene法官使用塔尼法案撤回了Biunno法官的意见。

AT&T和共和主义者审判部门维持塔尼法案没有执行，因为这是1956年和解协议的“延期”，不是一个新的事件。Greene法官的回应非常坦率:“这个法官的意见是，如果这个推理是可接受的，这里的当事人...可以仅仅通过一个法令推翻国会”。

这场纠纷又持续了几个月，但是在1982年6月的末尾，Greene法官有了一个“提议判决”，其中的关键是“从剩下的AT&T拆分出贝尔操作公司”。

我不想再去深究乏味的细节，但是Greene法官的意见导致的后果是西部电气被拆分，1984年几个操作公司组成了“小贝尔”，贝尔电话实验室也被拆分出来，重新命名为AT&T贝尔实验室，AT&T现在被允许进入计算机硬件和软件商业。

这就是1983年出现System V的原因，就像来自AT&T一系列全新的授权条款。到了1987年，AT&T的地位收到了很大的冲击。1988年它买下Sun公司20%的股份，一年后AT&T和Sun退出了融合AT&T和BSD特性的SVR4。1990年AT&T建立了UNIX系统实验室来继续开发和销售Unix。

在这期间，AT&T贝尔实验室依然在继续工作，开发实用和创新的软件。更加值得注意的可能是Plan 9，我会在第26章再做详细介绍。

/usr/group

USENIX里来自市场人员的声音从来没有被听到过。John Bass(还有其他人)都曾抱怨过这件事，但是Mel Ferentz，Lou Katz，Lew Law，还有社区的核心人员毫不为之所动。Brian Redman向我回忆1979年1月他去圣莫妮卡参加USENIX会议的经历

> 一些家伙站起来吧他嘘下台，因为他是一个市场顾问或者其他类似角色。Gee，我认为这些家伙非常危险

/usr/group在1980年建立，并在1981年组成公司。它是一个“致力于宣传UNIX操作系统”的交易协会。在1989年8月，协会成员投票将/usr/group的名字改为UniForum，他的大供应商展览会也被这么称呼。

Bass对于USENIX协会的形成和之后/usr/group的创建的看法提供了一个和别人不同的观点，他告诉我:

> 我们在纽约会议上讨论了几个特许试验之后，USENIX官方成立了它。Mel和Lou，还有我们中的很多人在走廊上睡觉，我们通宵讨论，举行了一次全员匿名投票来提名和选举出提名委员会，提名第一届董事会的候选人。我记得在圣莫妮卡的下一次会议上，我们发现他们提名自己并且未经预先讨论就提出一个和全员通过完全不同的许可...并且用一种圓滑的方式掌管了对他们缺乏程序这一问题的反抗。
>
> 他们想去控制USENIX，想通过源码授权重新让大学Unix数据中心管理人员掌控Unix...虽然早期的董事会捏造了3个这样的和2个其他位置，一个商业一个政府。全体成员投票限制了源码授权的持有人，他们成功的将董事会和投票者控制在自己管控范围之内。另外两个被广泛接受的模型是由专业的组织(例如ACM/IEEE)对于成员没有限制，没人可以投一票...每一个Unix站点可以是二进制也可以是源码(我们没有看到限制源码的需要)。Mel觉得这是他的孩子，只是因为他组织了第一次东海岸会议，他拒绝让出控制权。Lou和一个东海岸选的一个人被孤立了。另一个经讨论的简单许可的重要方面是支持本地组织运行学生和其他没有旅行预算的人加来和学习。在Mel第二次意外的投票后，他告诉我们中有强大本地组的人:接受他对于USENIX的愿景或者成为一个“孤立的组”。因此我们让步并试图在他的新框架下工作，它的框架日益变得对非“大学Unix管理俱乐部”不友好。
>
> 一年以后由这些有限制的俱乐部组成的USENIX很明显不能服务剧烈增长的用户...因此Bob Marsh领导的一个组在圣克拉克见面，并且在每一个成员的投票下组成/usr/group。
>
> 如果当初Mel允许像ACM/IEEE这样我们想要的每人一票的专业组织许可，今天的/usr/group就不会存在。我怀疑USENIX和Uniforum的存在引起的分类和不协调也永远不会发生。

1983年的圣地亚哥，1984年的华盛顿，还有1985年的达拉斯的USENIX会议都是由/usr/goup组织的。1987年的华盛顿，1988你那的旧金山，1990年的华盛顿，1991年的达拉斯，还有1992年1月会议在同样的区域举办，但是实在不同的位置(在不同场馆间使用穿梭公交)。通过这几年，/usr/grop是标准(下面会将)的强烈支持者，而USENIX协会更强烈的支持研发。技术会议和厂商展览在时间和地点上的分离是技术和工业市场之间完全不同的一个可悲的例子，尽管很明显的事实是他们之间都离不开对方。

在日本，科研和厂商的分离并没有立即引起了/usr/group的形成。Erik van Poel告诉我:

> 日本的UNIX界是由来自SRA的Kouichi Kishida建立的，一个独立软件工作室。我觉得建立它基本上是因为乐趣，相比于它现在的商业市场，UNIX看起来更像是一个研究区域。因为它是出于乐趣和研究，所以它接受大学和公司的人们一起来做事情(当然日本政府，学术，和工业界的合作一直都有很好的声誉)。
>
> 慢慢的，商业方面的Unix开始出现，我认为它的时间是在连个委员会的诞生:一个是称之为T Kanji Kai(技术委员会)，另一个是B Kanji Kai(行政委员会)。这两个委员会之间，还有管理董事会和一个秘书控制着。
>
> 我在T Kanji呆了一段时间。T和B经常在不同的时间开会，有时也会在一起。他们之间没有竞争和对抗，但是你可以在态度和兴趣上感到明显的不同。我认为这和USENIX于UniForum之间的关系很像。

“分隔”可能马上就来了。就像Toru Takahashi说的:

> 当Jun Murai，Koichi Kishida，还有Nobuo Saito建立了Unix学习组，那时一个由学院和厂商参与的一个很小的组。但是，两年以前，一些厂商和软件公司建立了Unix商业协会(UBA)，UBA成了Unix的商业代表组。因为我是jus执行董事会的成员，我很关心UBA。在jus，商业组织没有真正的权利。但是当jus维护一个面向技术的社团时，UBA成长起来了。
>
> 你知道在jus和JUNIET之间没有商业关系吗？每当我被外面人们问到，我都回答说没有任何关系。Jun Murai是jus的组织者，JUNET和它的WIDE互联网项目，但是他在jus的朋友跟着他做了JUNET。同一个人从jus去组织和开发JUNET。现在，JUNET不再存在。它被分割为商业和地区学术研究网络。

Van der Poel补充说:

> 实际上，整个事情非常的温馨。在这段时间，只有JUNET和一些知名的像SRA，东京大学，东京技术学会，应庆大学，当然还有日本Sun，日本DEC，还有索尼。自从WIDE和后来的IIJ出现，UNET似乎是消失了(导致的结果就是UUCP升级成IP)。
>
> 这样我们有了:
>
> | | |
> |---|---|
> | JUNET | 私有的UUCP，后来的IP(学术，公司和政府) |
> | WIDE | 研究，实验 IP |
> | IIJ | 商业IP(其他还有SPIN和IIKK) |

JUNET是由Jun Murai建立的，我不确定JUNET的官方说法是什么，但是很多人开玩笑说它的名字来自jun这个词，或者它的意思是日本大学网络或者日本Unix网络。

Murai在应庆大学教学。他在1984年和日本的计算机开始产生联系，他声称他最大的目标是“连接世界上的所有计算机”。

Sun和JAWS

第一个真正尝试Unix硬件的是ONYX。它在USENIX协会议上由O'Dell和Neil Groundwater同时宣布。ONYX由Bob Marsh创建，同时他也创建了/usr/group。John Bass那时候在为他工作。他告诉我:

> Z8000 ONYX芯片几乎就是一台VAX。我们拿给Boulder的系统是三片大约15英寸长22英寸宽的板子(我不记得具体的大小了)。它的性能和架构更像是一台PDP-11/45或者70...分段内存，没有分页。紧接着发布的LSI 11/73稍微慢了一点，但是价格便宜了很多(我家里用的就是这个，后来升级到了11/73)。
>
> 另外，ONYX是第一个运行Unix的桌面电脑。它有8个串口(每用户一个)，价格不到2.5万美金，它成为短期内PDP-11 Unix系统的替代品。
>
> 在这个时候上建造一个系统最大的问题是磁盘驱动技术...它非常大，而且价格很高。ONYX真正的创新是使用了8英寸20-60Mb IMI驱动和一个1/4英寸纸带...所有这些东西组成了产品。实际上，ONYX存在的原因是来自Carl Berg的风险投资建立了IMI并设计了驱动器，然后他们需要一个市场销售出去...Carl为了保护他在IMI的投资不得不又创建了3个计算机公司去销售驱动器。ONYX，Corvus，还有一个公司我记不清楚名字了。
>
> 有意思的是Carl的一个朋友拥有一个计算机商场，他不断告诉Carl说Unix没有市场，ONYX应该把精力放在CPM/MPM Z80系统上(C8002的I/O控制器和资金引导机器)。1980年夏天，Carl解散了ONYX团队(所有的Unix员工)，重新关注CPM/MPM的公司...一年以后IBM发布了使用PCDOS/MSDOS的个人计算机，六个月内CPM的市场就死掉了。Carl和他的ONYX不得不重新搭建Unix团队，重新将精力放在Unix上来挽救公司...但是已经太迟了。

就在那个时候，Xerox开发了Alto，这是一个“个人工作站”。它和我们今天所想象的工作站很像，尤其是它有一个位图显示器和一个鼠标。相比而言它也很贵(它从来没有销售，但是估价是在3万美金)，但是当摩托罗拉68000处理器出现的时候，Andreas Bechtolsheim和其他斯坦福的学生就可以很廉价的仿制它。斯坦福授权了一个单板的叫做斯坦福大学网络板，SUN。一些公司得到了它的授权:Codata，Fortune，Dual，Cyb，Lucasfile。另一台工作站——JAWs开始出现，售价不到1万美元，运行4.1BSD或者System 3。大多数公司都在创业的路上。Sun幸运的雇佣了板子原型的设计者Bechtolsheim和4.1BSD的首席设计师Bill Joy。1976年Palo Alto的Boggs和Metcalfe在Xeox项目中发明了以太网，他们在7月的CACM发表了论文。Xerox在1980年10月公布了详细说明书。桌面计算机和网络的概念开始在Palo Alto发芽成长。

1982年6月的《;login:》发表了下面的文字:

> 开发趣闻
>
> **Bill Joy离职UCB去了Sun**
>
> Bill Joy决定成为一个新创业公司的一员并将在接下来的几个月逐步退出伯克利的计算机系统研究小组。他会加入Sun公司，Sun的创始人包括Sun工作站的设计师Andy Bechtolsheim。Sun公司和一些公司一样，计划提供基于微处理器的运行4.2BSD的网络工作站。
>
> 当SMI需要在某些特殊领域开发专用软件时，Bill预计SMI对于4.2BSD程序的修改可以被伯克利一同发行。现在于CSRG和几种产业小组的合作被视为一种关系模型...

这篇文章的签名是“Prof. Bob Fabry”。

1982年7月6日周二在波士顿的USENIX会议上，Joy发表了关于关于4.2BSD和进程间通讯的演讲。Leffler和McKusick也对4.2的其他方面发表了演讲。第二天，Andy Bechtolsheim谈论了“SUN工作站”。

他向1000多位听众介绍工作站:

> 这是一台运行Unix面向图形的个人计算机，被设计为分布式环境下的通用工作站。用户接口包括一个光栅机制的800*1024分辨率位图显示器和一个可选的鼠标。它的处理器是运行在Intel总线上的摩托罗拉68000。将来它会升级成为68010(68010带有全功能的虚拟内存)。你还可以选择添加本地以太网络，几台工作站可以通过联网分享资源，比如可以分享文件系统和像打印机，纸带驱动器之类的外围设备。
>
> 现在的工作站运行的是UniSoft的第7版Unix，它包含了一些伯克利的增强功能。当4.2完成以后它会升级到4.2BSD，这将允许工作站直接在网络上运行而不需要本地磁盘...

在这场会议上，MIT的Jack Test讨论了“NUnix窗口系统”，来自BBN的Rich Fortier和Tony Lake谈论了BBN的智能位图终端BitGraph，贝尔实验室的Rob Pike谈论了“合并Unix与位图”。

1982年Hewlett-Packard宣布了它新的9000系列32位工作站。这之前HP宣布将在基于摩多罗拉68000芯片的9826和9836桌面机上支持Unix，这个Unix系统很大程度上源于System III，但是包含了很多扩展，包括以兼容以太网和伯克利vi屏幕编辑器。

1983年1月在UNICOM(软件工具，USENIX和/usr/group联合举办)会议上，Tom Lyon和Bill Shannon(现在都在Sun)谈论了移植4.2BSD到Sun。它不是真正的4.2，只是4.1c，但是它是BSD并且运行在68000上。在同一场会议上，Eric Shienbrood和他的同时谈论了Apollo计算机上的Unix——这是AUX，一个Unix System III的模拟器。那年的后期，IBM宣布在它的PC上已经可以使用Unix，这是基于16位Intel 8088芯片的机器(移植工作是Interactive做的)。

但是最重要的事情是伴随着Unix操作系统诞生的高速工作站。这直接导致了网络的诞生，如果没有运行Unix的工作站就不会有它的出现。Apollo的领域软件在很多方面要强于互联网协议和Sun的NFS软件。但因为它不是开发的，所以失去了市场。Unix是一个功能性系统，不仅是在大学校园里，更是在科研和设计实验室以及工业领域里。

标准

来自市场的压力和用户的需求驱动着建立一个Unix的标准。在不到10年的时间里市场不断激增。在第10年到来的时候，许多公司诞生了。大多数销声匿迹了，一些幸存下来。不兼容的软件和硬件大量出现是日益增长的厂商们主要的问题。不像1956年到1978年这段时间，当时几乎每一个计算机中心都处在单一的阴影中——通常是蓝色。微型计算机和工作站——苹果，PC，微软——都在改变这些。

这里分为两种:系统标准和语言标准。其中第二种(C语言标准)是完善的:美国国家标准协会成立了X3J11委员会——C编程语言——试图定义ANSI标准X3.159-1989，之后成为国际标准，ISO/IEC 9899:1990。

操作系统标准的建立经历了曲折的过程。Unix标准起始于Heinz Lycklama向/usr/group董事会提议建立了一个标准委员会。1984年在委员会的努力下制定了一份/usr/group标准并被成员接受。虽然它没有官方身份，/usr/group还是非常有影响力的:当ANSI X3J11成立的时候，它使用这份标准作为C语言库部分的基础。总之，随着System 4，4.xBSD，和XENIX的开发分支不断出现，/usr/group标准很快就不再适用。

在1985年早期，/usr/group委员会与新成立的IEEE POSIX(电子和电子工程协会，可移植操作系统)工作组合并，/usr/group标准被接受成为第一版草稿。

到1989年POSIX分化出了10个子委员会。4年以后，子委员会数量超过20个，虽然只有两个标准——不可否认是最重要的——被修订(POSIX.1和POSIX.2)。还有一个POSIX.0文档:POSIX开放系统环境简介，试图解释应用程序的移植。

无论如何，意识到POSIX不是Unix是很重要的。它是一个借口，不是一个实现。Fred Zlotnick曾经将POSIX称为“Unix的一个归纳”。POSIX.1提出了很多Unix的概念并将它们替换为一种抽象，这样将大量其他方面不兼容的系统可以称之为“遵从POSIX的”。因为标准越来越被市场所驱动，相比于技术驱动，它很难预料它的结果。行业工会发布的大量标准对于用户并没有什么帮助。

例如，1985年1月，AT&T发布了它的《System V接口规范》(SVID)，一个“有版权，无专利的开放系统文档”。一个新版本在5年后发布。在这期间，X/Open发布了它的《移植指南》(XPG)，现在它已经有了几个版本:XPG2，XPG3，XPG4。XPG3和XPG4在某些方面超越了SVID。

88open是一个“公司和个人致力于基于摩托罗拉88000 RISC处理器家族创建一个多厂商，开放计算环境”的团体。它建立与1988年。这个联营企业发布了一个资料读物和一个“开放系统参考指南”:《标准世界》。我将在后面的章节再详细介绍开放软件基金会和Unix国际化。

Unix会议的20周年纪念会议上，Ritchie评论C语言标准和POSIX:

> 我们为Unix标准化做了相当小的贡献。最大的合作是为POSIX shell做了一些改良。
>
> 和C语言标准相似，我谨慎的避开了X3J11。当Larry Rosler和Dave Prosser是草稿校订者并且AT&T身处委员会时，我和他们做了一些幕后的合作。但随着著名的无记名例外，我太多嘴，没有参与进去...
>
> 我觉得C语言标准是个很棒的工作。实际它几乎是一个成功的标准化成果的模型。他们将一些本质上没问题但是实际上需要净化和更新的东西在不破坏的前提下净化更新了。当然可以有很多批评...但是，当你对比其他语言的情况，像FROTRAN 8X甚至是Pascal，当有一个标准来规范，C语言的命运更加让人高兴。

Ritchie对于POSIX的成果没有很大的兴趣:

> 这是一个混合的袋子。系统调用和程序库甚至是命令看起来相当合理，争议和搞砸都还不坏。着这个结果上产业界看起来很不错。换句话说，这部分缺乏一些很显而易见的东西，就像网络的连贯一致。Presotto和我几年前谈论USENIX的一种方法...无论如何，我们的方法看起并没有接管世界。
>
> 同样，IEEE POSIX的成果看起来是一种糟糕的方式，它吓到了大多数的人们。特别是，实时组看起来聚集了所有未被充分理解的点子，他们看上去和已有的系统不想匹配。我现在担心程序任务的想法，几个小的进程在同一个内存镜像里执行，因为Unix和它的软件并不能很好的适应——C语言不支持，程序库也不支持。多处理器在你设计软件时有很多暗示，我害怕仅仅列出来会导致混乱。
>
> 说我自我矛盾是保守的!标准化产生的不安在很多方面是非常巨大的。
>
> 经常提到的一个是创新和标准化的矛盾，我觉得这不是一个真正的问题。如果有了标准，你可以精心的选择是遵从它们还是做些其他的。这是一场赌博，但是是最合理的一个。真正的问题是还没有决定特定范围是否应该标准化。当一个应该制定的唯一标准(VHS和XBeta)还没有做出来才是真正可怕的。
>
> 人们应该意识到的是和我们相关的许多标准是由非常小的小组制定的，如果你足够关心并且以正确的方式工作，他们会受到影响。在起草阶段，一个恰当的主张，一个具体的建议都大有帮助。这是悖论的一部分，当然，它意味着一些很棒的人可以挽救大局，但是也意味着一些愚蠢的人可以把事情搞砸好几年。

自从Ritchie的评论发表后，POSIX子委员会的数量渐渐变多，他们之中只有少部分有提议并继续投票。后来，随着更多工会的成立，非正式标准的数量随着几个IEEE委员会所做的事情一起变多。

Unix间的竞争

直到1978年，Unix都意味着AT&T的操作系统。随着PWB 2.0，V7，32V，和3BSD的出现，它加入了更多东西。但是Unix不只是AT&T的产品，它是一个“电话通讯支持工具”。因为和解协AT&T议被迫离开计算机商业，不允许继续在计算机市场参与竞争。没有了AT&T的支持，Unix的未来命运从科研界转手到了Unix支持小组(USG)。USG在70年代大部分时间掌握着Unix的命运，然后到了1979年，微软和圣克鲁斯操作公司(SCO)推出了XENIX2，伯克利推出了4BSD，这两个系统都是V7的衍生版本。XENIX是Unix针对Intel 8086和其他很多硬件架构的第一个移植版本。虽然它变得和其他版本不再兼容，但它也是一个非常流行的版本。自由软件基金会的Len Tower向我评论说“SCO Unix用起来就像是在向前穿越时间”。

正如Andy Tannenbaum指出的，“贝尔实验室有一个Unix支持小组，但是你从没听说过Unix开发小组。这是因为贝尔公司没有权限开发计算机操作系统...”。1981年Jeff Schriebman创建的UniSoft推出了一个名为UniPlus+的移植版本，它与System III和现在的System V保持兼容。1983年伯克利的一个小组成立了mt Xinu来支持和商业化BSD，其中的成员有Bob Kridle，Alan Tobey，Ed Gould，还有Vance Vaughan。Debbie Scherrer不久后也加入了他们。他们先是销售了更多的4.2BSD，然后是4.3BSD(mt Xinu擅长于客户服务:1986年Debbie自己安装了我的4.3纸带)。

5年的时间中，Apollo，DEC，Eakins，Gould，Integrated Solutions，Masscomp，mt Xinu，NSC，和Wollongong是众多Unix销售公司中的一部分。这些销售AT&T System III或者System V衍生版本的公司有:AT&T，Altos，Apollo，Compaq，Convergent，HP，Honeywell，IBM，ITT，Intel，Interactive，Masscomp，Microport，Microsoft，Motorola，NCR，NUXI，Opus，SCO，Silicon Graphics，Sperry，Sun，Tandy，UniSoft，Wollongong。此外，Amdahl，Apollo，Apple，Cray，DEC，Data General，HP，IBM，Inter，Motorola，Unisys和许多其他的公司提供私有的Unix版本，其中一部分是基于4.2BSD的。

所有的这些版本，不管是AT&T还是BSD的衍生版本，都需要AT&T的授权。最近几个Unix的版本可以不需要这种授权。虽然没有一个商业产品是眞正强健的，但BSDI，386/BSD，还有NetBSD都已经可以在任何386/486机器上运行。这些“自由授权”版本都衍生自自由的CSRG发行版，当我写这些的时候它仍然在诉讼中。详细信息可以参见29章。Linus Torvalds写的Linux也运行在386/486机器上，不过没有使用CSRG的代码。这四个系统都使用了大量自由软件基金会的软件。Ritchard M. Stallman创建了自由软件基金会，并为他几近完成的GNU系统写了那些软件。GNU软件包括大多数Unix软件可以自由分发的重制版本(它们在GNU项目中一个叫做‘copyleft’的授权下发行，这意味着原始和改进后的版本必须可以让其他人自由修改和分发源码。如果发布了二进制版本，源码也必须一起发布出来)。**gcc**是GNU的C语言编译器，可能是Steve Johnson的pcc最好的继任者。

最终，这些衍生和克隆版本可以运行在一系列的处理器上:DEC，Intel，MIPS，Motorola，NSC，更多的就不再一一列出。

这些长长的名字列表让用户和准用户们感到疑惑:这些都是做什么的？当然，尽管名字们看起来很混乱，但没有“开放系统”就没有任何标准化，就会导致无法协同工作。换句话说，如果你理解了BSD或者SVR4，又或者AIX，HP/UX，Solaris，Ultrix，那么你对其他的变种就已经理解了90%。程序开发的编程语言是ISO/IEC C。

Doug McIlroy问我:

> 你有没有想过，为什么Unix在刚开始的时候那么小巧，又有着比MS-DOS强大的多的能力，但它却没有进入个人计算机的世界？还有为什么现在的Unix这么臃肿？

Armando Stettner给我的答案是Unix从来没有进入“微软的市场”，并“人们工作中真正的需求要比V6和V7提供的多很多”。我想Unix没能进入个人计算机市场的一个原因是它没有一个让普通人可以很容易使用的界面。虽然我很多编程的朋友都声明命令行比MS-DOS更加友好更加灵活。当IBM退出了运行MS-DOS的PC，当乔布斯和沃兹尼亚克推出了苹果，从来没有坐到计算机前面的人们感到他们可以和这些机器交互。回忆一下1983年苹果的Lisa，他们也没有特别成功。这些从来没有在大学校园和科研协会使用过大型机器的人们，他们不知道自己错过了什么。虽然MS-DOS比任何一个Unix命令行都要差劲，还有着差强人意的错误消息。

我不知道微软投资XENIX是不是想要阻止Unix在市场上的成功，但是它显然有这样的效果。

关于Unix为什么变得臃肿了，我觉得这很简单。如果我们想象Ken和Dennis，Bill Joy和Kirk以及Sam Leffler，Mike Karels去做“品位裁决”，就不会有现在这么多的Unix厂商。当System V采用了大量的伯克利增强修改，将它们整合进系统，AT&T的Unix系统实验室(1993年被Novell收购)仅仅将将代码收录在一起。这是一种满足用户所有要求的态度。想要一个编辑环境？这有Bill Joy的vi和Richard Stallman的emacs。想要一个窗口系统？这有X Window系统和Motif以及OpenLook两种用户接口。想要一个命令行？这有sh，csh，还有ksh。在那时，USG被告知为了证明开放性，要讲大量BSD的特性添加到System V中去。我不能确认这一点。这些特性每一个都增加了空间占用，分配了更多的内存。每一个特性都是几千上万行的代码。一个苗条的系统应该有更少的选择和更少的实用工具。

衍生版本

就像早期的Unix版本衍生出了MERT，PWB，LSX和Idris，V7和4.xBSdD是一大票系统的先辈。字母表上可以从Amoeba到V，地理上可以从西方的耶路撒冷到东方的台湾，这里可以列出来的实在太多了。不管怎么说，在过去十年间开发出了各种类型的版本，在这一章我将选出其中5个来介绍:MOSIC(Israel)，Chorus(France)，Amoeba(Netherlands)，Mach和Plan 9(US)。

MOSIX(Unix的多计算机操作系统)是一个可以将多个计算机连接成为一个虚拟单机Unix环境的系统。1981年Amnon Barak在耶路撒冷的希伯来大学开发了MOSIX。最初它的名字是MOS，被设计成基于第7版Unix连接多台PDP-11/23s。83年到84年期间开发出来并在M68000处理器上运行。到了1993年一共有连续4个版本发布，包括一个32V版本(1987年，叫做NSMOS，因为它在基于National Semiconductor NS32332的计算机上运行)，一个VAX版本(1988年夏天基于SVR2)，还有一个VME523版本——这个最流行的版本运行在多处理器工作站集群上。最初引入MOSIX的特点就是通过动态迁移实现的负载均衡，或者也可以称之为缩放算法。

1979年Chorus最初是Hubert Zimmerman在INRIA开始做的一个研究程序。直到1986年，一个很大的团队为发行版系统开发了3个面向通讯的内核，这些早期版本都有一个自定义的接口。1986年的Chorus-V2兼容于Unix System V。大约在这个时候这个项目被作为一个商业公司剥离出来:Chorus系统公司。

Unix系统实验室在1993年宣布成立，在它被Novell收购之前它使用Chorus的技术。这里一定要指出的是，作为一个Unix研究社区开放性的直接结果——就像我在之前章节不断提到的——在本章这些系统和项目之间有大量的“分享”。Chorus-V3发布时有着和Mach类似的虚拟内存和“线程”，网络寻址看起来和Amoeba很像，统一文件命名又和第9版Unix差不多。在1970年后期，只是一个优点而非缺点。

Chorus包含了一个核心和几个服务系统，服务系统在一个子系统环境中互相协作。其中一个子系统的实现是用Unix实现的。Chorus系统公司卖出了大量既有用又有趣的核心和Unix子系统。

1980年中期，阿姆斯特丹自由大学的Andy Tannenbaum(MINIX的作者)预言了未来世界的计算，包括分布式的物理硬件和逻辑上集中的软件。在这个想法下，Amoeba项目在自由大学开始创建。随后，Sape Mullender和其他几个CWI(之前的数学中心)的人组成团队和自由大学的人一起开发Amoeba。Amoeba包括:

- 窗口系统的工作站
- 针对计算的处理器池
- 定制的服务(文件和目录)
- 和其他系统的连接

Amoeba的软件是面向对象的，有一个模拟Unix软件包，还有其他各种各样的东西。Amoeba的设计理念是，一个多台机器组成的本地局域网集群，应该可以通过远程网络连接远方一个类似的机器集群。远程网络通畅速度很慢并且不太可靠，Amoeba在网络方面的主要目标是在不牺牲性能的前提下透明压缩传送数据。

作为1978年Tannenbaum和AT&T授权不愉快的结果，Amoeba是从头开始写起的，内核和实用工具都不包含任何AT&T或者伯克利的代码，并且不需要任何授权。

Unix进入多处理器世界的另一个副产品是卡内基梅隆大学创造的Mach。最初的开发人员是Avadis Tevanian，Rick Rashid，还有他们的一些同事。Mach的目标是大量的计算机架构，包括但处理器，多处理器，和分布式系统。它想要提供一个紧凑高效的内核，然后在此之上为各种操作系统提供接口层。Mach使用的文件系统来自1982年卡内基梅隆大学维护的4.1BSD的一部分。内核来自Rashid1981年在Accent的工作(一个‘面向通讯的网络操作系统内核’)。Mach完全的向后兼容4.3BSD。Mach是开放软件基金会在89年冬天为它未来的操作系统指定的内核。

1987年Rob Pike和Ken Thompson开始设计一个Unix的代替者。贝尔实验室的Dave Presotto，Tom Duff，Howard Trickey很快也都加入进来。它们把这个系统称为Plan 9(名字来自史上最烂的科幻电影《来自外太空的Plan 9》)，系统分为三层，一端是公共服务器，一端是终端。服务器是多处理器的机器，终端本质上是无盘工作站。第三层是广域网，连接终端和服务器的网络。Greg Chesson，Sandy Fraser和Dennis Ritchie的Datakit网络是远程网络。Plan 9违背了现代计算环境的趋势:从1986年Sun 3的冒险开始，“多数主义”观点变成大容量磁盘工作站组成的本地网络。Plan 9是一个融合单独服务器和终端的分布式计算环境。每台机器都用单一协议和本地命名空间操作连接起来。

Plan 9曾经安装在基于i386处理器的笔记本电脑上作为服务器用。在1993年后期被一个商业厂商接手了。Dave Presotto告诉我“我们开始了一个新的系统——但使用了一个烂电影的名字”。

OSF和UI

1987年后期AT&T宣布收购了Sun的大部分股份，Sun将会和AT&T/USL一样开发新的软件。Sun宣布它的下一个操作系统不会是将基于伯克利Unix的SunOS继续扩展，而是将会源于System V第4版。Unix世界受到很大的震动:科研社区感觉Sun正在背离他们，其他的制造商担心“特殊关系”会让Sun抢在他们前面。Armando Stettner告诉我:

> 当Sun和AT&T宣布结盟，我们DEC的人都担心AT&T会变得不再友好，不再是那个友善的Unix先驱...
>
> Sun是每个人的强大竞争对手。我们看到Sun的系统可以直接替换VAX，就想:这次结盟联合了我们最大最有创新力的竞争对手和系统软件的鼻祖，市场平衡被改变了。

直接的后果是1988年1月7日在DEC位于加利福尼亚州帕洛阿尔托的西部办公室举办的会议。参会者来自Apollo，DEC，Gould Electronics，Hewlett-Packard，Honeywell-Bull，InfoCorp，MIPS，NCR，Silicon Graphics，UniSoft，以及Unisys。因为DEC的办公室是在汉密尔顿林荫道，出席的人被称为汉密尔顿小组。他们在1月15日发给ATT&T CEO James E. Olson的一封电报:

> 作为AT&T软件的授权方和开放Unix标准的支持者，我们很关心近期AT&T和Sun发布的公告。这份公告让我们的公司和顾客对Unix作为一个开放标准的未来产生忧虑。我们感觉在即将到来的“uniform”会议之前很有必要针对这个问题得到更好的理解，我们请求我们公司负责Unix战略的管理者和Cassoni先生在1988年1月25日举办一次会议...

Vittorio Cassoni是AT&T分管数据系统的资深总监。作为一个良好活动的结果，在电报上签名的代表分别来自Apollo，Data General，DEC，Gould，HP，Intergraph，Integrated Solutions，MIPS，Motorola，NCR，Prime，SGI，Tandem，UniSoft，以及Wyse。

1月27日晚上HP的Larry Lytle在肯尼迪万豪酒店召开了一个预备会议。这次和Cassoni的会议发生在1月28日。2月9日在达拉斯还有一个汉密尔顿小组的跟进会议。和Cassoni的会议没有得出小组关心的明确结果。到了3月，小组决定邀请IBM加入。

Apollo，DEC(在Stettner对Ken Olsen的催促下)，HP，Bull(法国)，Nixdorf，以及Siemens(德国)举行了一次半半开放的会议，并且在1988年宣布了开放软件基金会(OSF)的成立。这个基金会致力于操作系系统的开发，用户接口，发行版环境，和自由分享。最后，这个Unix分支会成为免AT&T授权。1988年5月18日的《Wall Street Journal》上记录着出席OSF午餐的人们没人记得Ken Olsen的分享“一个IBM领导的舞台”。

那时的Thompson正在澳大利亚。当Ritchie告诉他发生的事情时，他说“只要想一想，IBM和DEC在一个房间里，这是我们做的!”。

没过多久，AT&T就和Sun以及他们的合作伙伴建立了一个反对者联盟:Unix国际(UI)，致力于销售SVR4。

1989年末OSF推出了一个用户接口Motif，被广泛接受。OSF也将Phillips和Hitachi加入他们的赞助商。总之，HP收购了Apollo，Siemens收购了Nixdorf。OSF和UI都努力增加会员数量，大量赠送铅笔，徽章和贴纸。最终他们都有大概200个成员。

91年和92年因为经济恶化，Bull，DEC，IBM和Siemens的计算机部门都在亏损。并且随着AT&T出售它所持有的Sun股份，这场斗争看起来视乎结束了。早在1989年11月就有人谈论为了Unix的未来OSF和UI应该合并。1992年末时这看起来已经无关紧要:已经没人再谈论OSF的免AT&T授权操作系统，Sun采用了Motif。到1993年末时，UI退出了商业，OSF取消了它之前宣布的几个产品——压缩软件和分布式管理环境。Bull和Siemens(资金原因)以及Phillips(因为出售了计算机业务)终止了对OSF的赞助。Armando Stettner评论说:“不清楚OSF还会有什么作用”。

VAX后的伯克利Unix

在第18章我，我引用了Kirk提到到他们将“portals”放到了4.4中。作为British对Unix开发的又一个贡献，这是Jan Simon对我说的:

> 我将一个叫做“portals”的东西放到了4.4版本BSD里。它们和4.3版BSD架构手册中说的“portals”不是同一种东西，但是你可以用他们做很多相同的事情。可能我应该另外选择一个名字。
>
> 它具体的功能是一个守护进程创建了一个Unix域套接字，然后把它附加到系统命名空间(按惯例路径是‘/p’)。它在套接字上等待来自内核的消息...
>
> 正真的想法是将所有网络相关的代码从应用程序里拿出来，放到一个或两个守护进程。它的实现的V9的**ipcopen()**非常相似。当然，在4.4里面没有应用程序被修改过。你无需一堆**perl**代码，直接用**awk**脚本就可以创建网络连接，。

UC伯克利Unix的故事结局就像Kirk说的:

> 我们对Computer Console公司的Tahoe感兴趣。对我们来说很明显VAX不够快，也不够便宜。当然，DEC这次有Ultrix产品并且他们讲4.3BSD视为竞争对手(Armando Stettner已经不在DECUnix小组)。我们被停止反馈:新硬件的增强信息，每个东西的详细规格，设备驱动等等。很明显DEC会把我们的生活弄得更加困难，还不清楚从长远看VAX会成为什么样的架构。
>
> 在这段时间Sam为LucasFilm工作，他做了一个机器搜索。他将市场上所有的机器都分析了一遍。他决定CCI Tahoe系列的机器在当前是最有性价比的架构。并且它们也在绘图板上为未来的版本计划了很多很棒的东西。
>
> 它们有一台和今天的台式电脑一样大小的6 MIP机器。它们的绘图板上还有一台35 MIP机器，它会在大多数厂商都向年轻的学生销售MIP机器时开始出产。我们准备得到一些他们的机器。他们给了我们一个BSD的移植。他们觉得他们不得不向其他厂商做的那样为了市场销售去做一个基于System V的系统。
>
> Tahoe的4.3版本被设想成移植到Tahoe的4.3版BSD。实际上，就在同时有另外一些事情已经完成(比如磁盘标签)并且已经包含在系统里面，所以有些事情已经变得不一样。他们想做的第一件事就是不区分系统，这样它就可以同时支持两种硬件架构。
>
> 不幸的是，CCI取消了他们接下来的项目，Tahoe成为一个死掉的架构。他们让我们连第二版都没推出。
>
> Reno的发布时我们做的上一个系统，它是一个“直接在空白硬件上启动”的发行版。很明显我们需要移植到一个不是工作站的主机上。我们得到一个连接HP工作站的通讯线，68Kb的速度。部分地，我们之所以这么做是因为伯克利的一些人已经开始移植，犹他州立大学的人们已经完成了。HP乐意给我们硬件，所以它工作的很好。

1993年6月，伯克利发布了4.4版本，同时结束了CSRG。Both Keith Bostic和Kirk McKusick离开了加利福尼亚大学。Mike Karels在前一年离开。贝尔实验室的Unix研究在1980年晚期终止。现在最重要的非AT&T社团也“远离了Unix开发”。Unix仍然在开发，但历史上最重要的两个团体已经不再参与开发。

法律 第三部分

从1986年开始，Keith Bostic开始每半年举行一次USENIX会议，并且宣布他CSRG项目的进度:大概33%的代码剔除了AT&T授权，大概55%，大概77%...每一次进度的更新，都会迎来热烈的鼓掌和欢呼。AT&T的律师在1970年开始踏上错误的道路。但是这并不是欢呼的全部原因。人们要创造无需授权Unix的动机来自AT&T的收费。当UCSF收到V7，花了大约7000美金的授权费，商业授权远比这个还要高很多。Clem Cole回忆说:

> 自从梅隆协会开始使用Unix的这些天，为了公司利益，Dan Klein和我最后开始“罢工”直到CMU购买了一个真正的“产业”Unix授权。CMU成为第一个花费2万美元购买授权的大学。在其他一些地方也是这种策略，像凯斯西储大学几年后也购买了授权。

那是1978年。十年以后，程序员和专业用户仍然感觉授权使得他们和最喜爱的操作系统变得疏远:源码授权的收费已经增长到了10万美金。1993年到了20万。这远远超出了个人和小公司的承受能力。

John Gilmore和其他几个人催促CSRG创建他们的免授权版本。很明显AT&T反对其他的派生版本:Linux，MINIX等等。1988年秋天，在BSD的工作场地，Keith，Mike Karels和Kirk McKusick宣布已经完成了BSD Networking Release 1。它在11月上市。

NET1在那时是伯克利系统的一个子集，和4.3 Tahoe非常相似。它的内核网络部分，C语言库和使用工具都提供源码和文档。它不需要任何授权(AT&T或者伯克利)就可以使用，也可以通过匿名FTP在网络上重新发布。源码中包含了一份伯克利版权声明和说明，允许所有者重新发布。

两年半以后的1991年6月，在纳什维尔的USENIX会议上发布了redistribution with attribution 2。

NET2除了网络部分还包含更多的代码，但是像NET1一样无需授权就可以使用。新的特性包括源于卡内基梅隆大学Mach系统的虚拟内存系统，它被犹他州立大学移植到了386/486系统。

NET2是美国与俄罗斯合作的产物，做出贡献的有Bill Joliz，Donn Seeley，Trent Hein，Vadim Antonov，Mike Karels，Igor Belchinskiy，Jeff Polk，以及Tony Sanders，在他们的努力下成为一个商业产品。最后的结果是BSDI(伯克利软件设计公司)在1991年末组成，并在1993年4月10日向公众宣布。延迟了这么久的最大原因是Unix系统实验室(USL)，它为了阻止BSDI销售产品向他们提起诉讼。USL说“BSD/386及其后续实验，侵犯了USL在32V软件的版权并且侵犯USL的商业安全”，他们请求禁止这些发行版。1993年3月3日法官驳回了USL对于初步禁令的请求，裁定USL“没能列举版权声明和商业安全声明”。

1993年3月30日，新泽西州的Dickinson Debevoise法官重申他驳回了USL对BSDI初步禁令的请求。法院发现32V的源码发布时并没有附带版权声明。法院拒绝了USL关于32V有足够限制以避免丧失产权，这样USL没能论证它能保护自己的版权。法院还裁定USL不能因为商业机密而对BSD/386做任何约束。

USL然后提出复议申请，请求地方法院针对USL是否在发布32V时没有附带版权声明举行一次听审会。USL争辩说法院之前的裁定是基于一份错误的32V发行版(USL的复议请求没有改变法院的裁定，USL没能证实商业机密被挪用)。

法院禁止了USL的复议申请。虽然法院重新验证了发行版，但是发现并不影响之前关于没有附带版权声明的裁决。

自从Ken在SOSP上发布论文已经过去了20年。距离《UNIX NES》更名《;login:》和Unix用户组变成USENIX也已经有15年。但是AT&T，西部电气，和现在的USL都没有真正了解用户社区。

BSDI和其他人尝试去做的是确保Unix操作系统的开发，成长和使用。USL的诉讼是尝试去保护一个太晚才被发现有价值的东西。可能Ritchie和Thompson曾经在70年代中期不小心的处理了系统，可能是贝尔实验室的雇员有意将Unix不加限制的送给公众。

BSDI曾经发布了一个预览版BSD/386，成为现在国际发行的BSD/386。全部源码的价格大概是1000美元(在1994年1月到2月的《;login:》里Lou Katz写道‘它起作用了！起作用了！’)。

因为已经授权给加利福尼亚大学，大学的管理者也被USL诉讼。1993年6月管理层开始反击，诉讼USL没能遵守和DSRG签订的协议。就在同时，Novell收购了USL。

在那时，没法预见诉讼对这个市场的影响。不幸的是，一些像微软这样的公司在Unix战争和Unix厂商间互相制造的障碍中获取了利益。那时如果Novell想消除分歧来防御Windows NT的攻击，可能就会决定撤销接手的诉讼。

最后说一下，1994年2月4日，Novell和加利福尼亚大学同意撤销所有相关的诉讼和纷争。BSDI立即发布了4.4BSD-lite。“我们很高兴解决和Novell的冲突，这样我们就可以将全部精力都放到创造产品和服务何物上”，BSDI主席Rob Kolstad对我说。

它为什么成功

从CTSS到Multics再到PDP-11/20上面的Unix，我们看到了一个很长的故事，一个跨越30年时间的故事。但是我相信大多数(如果不是全部)计算机产业公司都没有看到其中的教训。

可能这其中最重要是在贝尔实验室盛行的态度。Dennis Ritchie在第1章里关于乐趣和实用的言论是这种态度的典型代表。Brian Redman向我评论他在惠帕尼所做的工作“充满了乐趣”。Plauger评论说“太奇妙了，如果你很有乐趣并且热爱你所做的东西，这就是AT&T和商业有关的东西”。Brian Kernighan说当他被雇佣时他问Doug McIlroy他该去做什么，他得到的答案是做任何你想做的事情。所以人们得到了eqn，ratfor，pic这类软件工具。

Kernighan告诉Peter Collinson关于awk:

> 到目前为止它是我参与过的最大的软件项目。我们有三个人参与其中，原本这是根本不可能完成的。然而，和两个人配合工作远比和三个人配合更容易...因为事情很难分割开。有太多的意见分歧，有时这是好事因为这以为这考虑了更多的事情，但是有时这也意味着可能没有那么团结。

人员太多了？如果你注意看新闻，你回发现微软有超过600位程序员在开发Windows NT。但在霍姆德尔有3个人正在创建32V(London，Reiser，和Swanson)。3个人在写HoneyDanBer，Hall，Sventek以及Scherrer写VOS，Lorinda Cherry和Bob Morris写bc和dc。伯克利的CSRG偶尔会有不少的全职人员:Haley和Joy。Haley，Joy和Kridle。Kridle和Leffler。McKusick的业余时间和他得到博士学位前的夏天。McKusick，Jolitz以及Karels。McKusick，Karels以及Bostic。Larry Wall自己写了一些补丁per，rn。看看这些离职的人们:Ken Thompson，George Coulouris，几个澳大利亚人，Jan Simon Pendy。再看看其他的贡献者:Mike Muuss，Doug Kingston，Jim Curry，Rick Adams，James Cosling，Rob Pike，Armando Stettner，Bill Shannon等等。

Steve Johnson告诉我:

> 如果你在大学里，我会认为你要么是个做软件的，哟啊么就是一个做理论的。将会有大把的金钱支付我的薪水。如果我是一个软件工程师，并且和理论家交谈，付我薪水的人将会认为收到了欺骗。相反的，如果我是一个理论家，并且试着去做一些有用的事情...我总是可以成为一个通才。我总是在不同的训导间操作，我发现了大量多产的地方。因此我将写一个程序，这将会发现一些我可以验证的命题将原理放回到程序中。这发生在我完成yacc和pcc以及其他一些东西之后。它只是不能很好的适应一个结构划分。

还有另外一个观点:避免区分研究和开发。

Eric Allman对我说:

> 我觉得一个软件设计的通用规则是你应该去写你想用的程序。Ken和Dennis想用Unix。他们为了去用它而去开发它。我们想用**sendmail**，它不是那种我们说的“好，让我们写一个邮件程序，然后把它发给...”。Bill Joy没有跑来对我说“Eric，我们需要的就是这个”，实际上，我是在他的催促下将它发布出去的。我们有一个问题需要解决。Ken有一系列的问题:他没有一个足够的系统可以让他去玩太空游戏，所以他就写了一个。与此相比的X.400，我被说服人们从来不像在纸上写一样去用电子邮件。其他公司的软件没成功是因为你指派人们去做那些工作。

当我和Armando Stettner聊天时，他对我评论说:

> 你知道，关于Unix和今天的Plan 9最美妙的事情是这些是大量优秀的想法随着时间通过一种非常有趣而高效的方式融合在一起...贝尔实验室集合了好主意并将它们实现出来。伯克利Unix将许多好想法做成了一个程序员接口——高质量的多种实现。集合了来自全世界人们的代码形成这个系统。并且裁决者有着很好的品位:Bill Joy和Sam Leffler是正真的艺人，Kirk McKusick，Mike Karels因为缺乏一个好的头衔，我们可以称他们为架构师。但是他们真的拥有良好的品位。他们讲人们的工作集合起来，他们确保事情清晰合适。

综合来说就是创造性。但是关于Unix的演变最有意思的观点就是它大部分都发生在它的“拥有者”AT&T和西部电气。Greg Rose向我指出:

> 在Unix如此多的演变阶段中，AT&T采取的扼杀行动实际上引起了相反的效果。

Brian Kernighan说:

> 你知道，个人计算机的商业环境需要持续不断的改变来保持占有市场。你不得不一直有些新的东西，或者你的竞争对手有了新的东西然后你被从市场中赶了出来。其他事情是个人计算机商业环境本质上是同类的，它们全都运行在相同他的处理器芯片上。因为环境没什么差异并且PC市场要比Unix市场大很多，因此你可以分期偿还你的花费。它是一个非常大的市场，如果一些人有了一个好主意他们会参与到PC市场而不是去Unix市场。我觉得市场将会发生改变，Unix的好处在长期过程中会超过DOS:可移植和通用的好处。

Unix影响了90年代计算机的每一个方面。甚至Kernighan声称“PC市场”带有Unix的印记。另外，微软的Windows NT很明显是一个类似Unix的系统，而且他们的产品宣称是“遵从POSIX”(虽然我不清楚比尔盖茨如何理解这一点)。

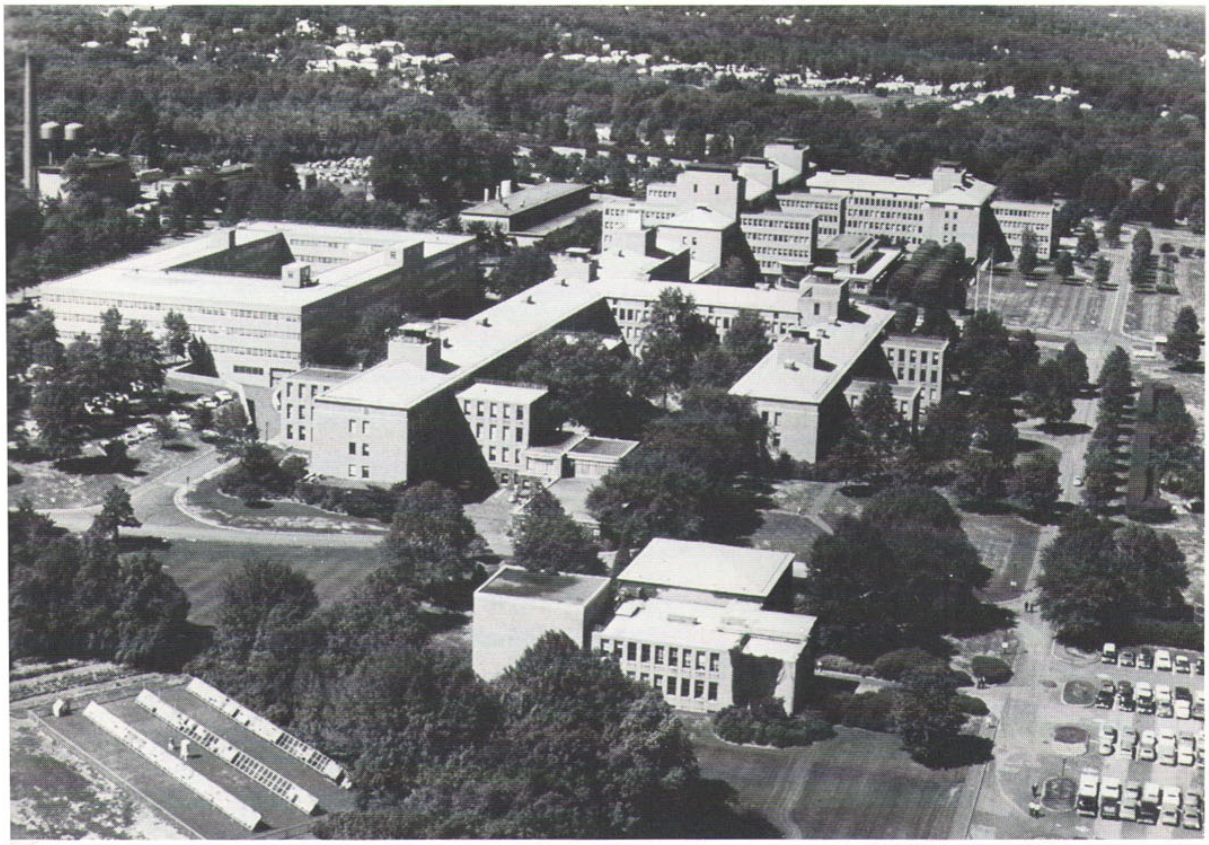
1984年Dennis Ritchie退出后曾被问到，贝尔实验室导致Unix出现的氛围是否可以重现。他的答案出现在1984年的CACM，几乎正好是发表《Unix分时系统》之后的第10年。在文章里Ritchie写道:

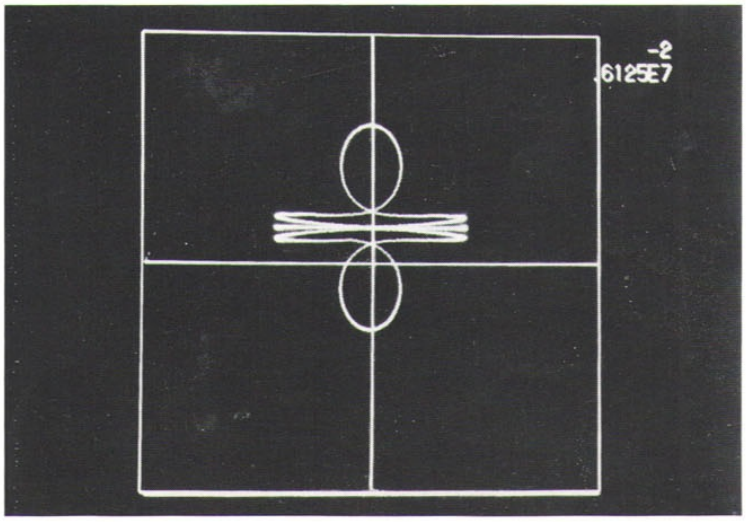
> 一些人感觉最初的Unix工作是一种非法项目，一个“臭鼬工作”。事实上并不是这样。研究工作者们被希望发现或者发明一些新的东西...我们总是管理激励...我们的意图是为我们自己创建一个舒适的计算环境，并且人们也都喜欢我们所希望的..
>
> 今天对于计算机科学研究最危险的事情是过度关联...另一个危险是商业压力会将决策者的注意力从真正的创新转向利用当前人们的爱好，从探索转向开采。
>
> 如果我们对创新保持足够的宽容，对通讯有着足够的自由，对成功有着足够的耐心，我们就有可能让未来的Ken Thompson发现一台CRAY/1计算机并且创作一个系统，继而产生影响，就像Unix。

Thompson和Ritchie想去创建一些自己喜欢使用的东西，他们成功了，其他人也想要分享这些令人愉悦的东西。Sunil Was写道“没有Ken Thompson的愿景，Unix就不会成为现实。没有Dennis的眼光，它就不会成为一个优美的存在。没有Michael Lesk的想象以及Brian Kernighan的推广，它就不会取得如此多人的广泛钟爱”。这份钟爱激发了伦敦的Coulouris，细腻的Lion，伯克利的Joy以及本书中提到的无数其他人。

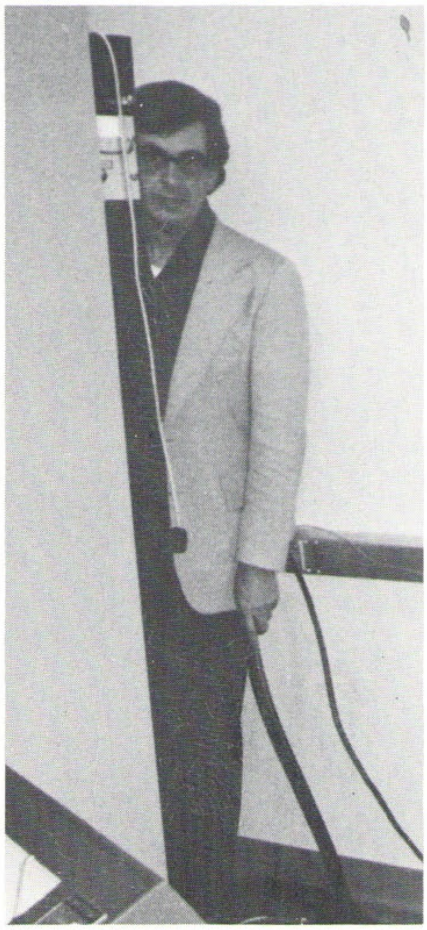
David Tilbrook总结说:“关于Unix已经明确的是它不是计算机历史上一次伟大的进步，如果一定要评价的话，它是一次杰出的简化，它将用户带入了之前无法想象的领域”。

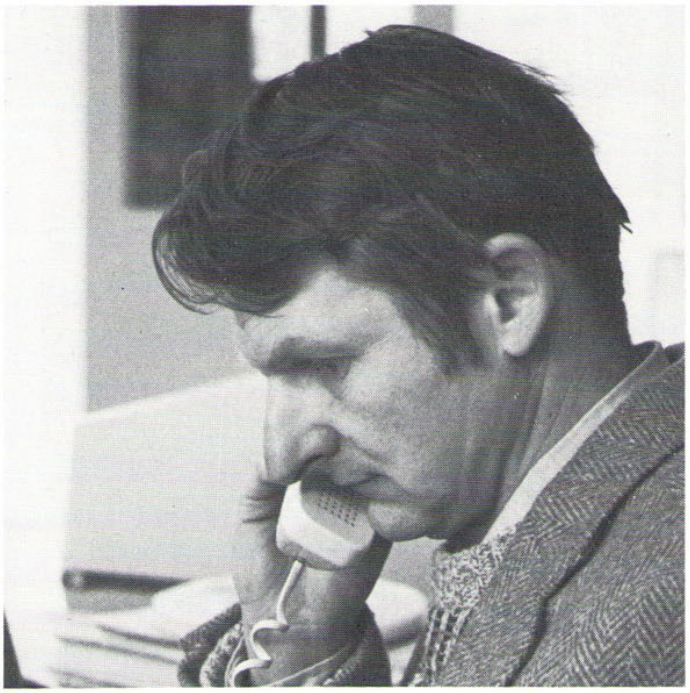
附图



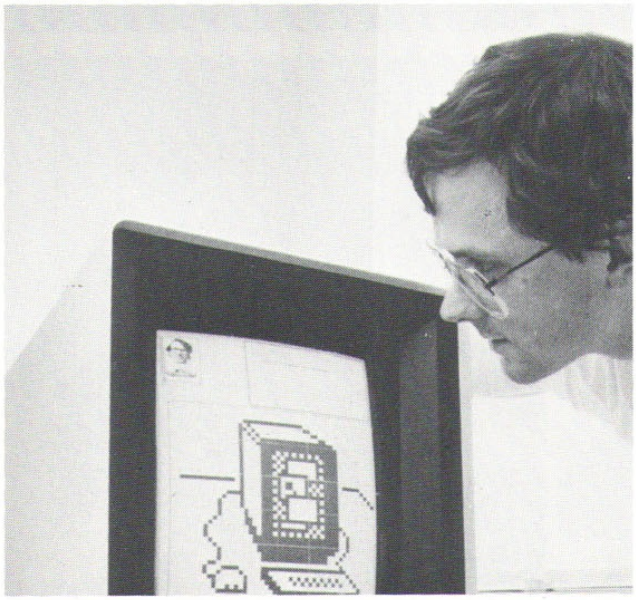


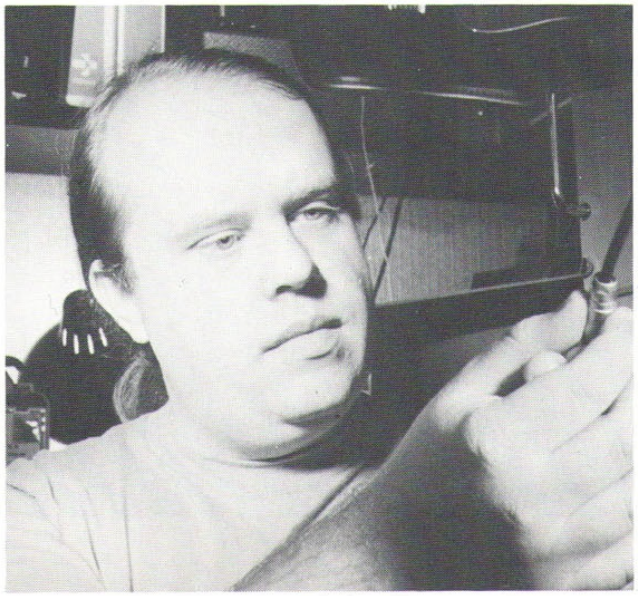
Top: AT&T Bell Labs in 1968 *(AT&T Archives)*
Center: Ken Thompson and Dennis Ritchie *(Kirk McKusick)*
Bottom: A Space Travel Orbit, photo from the PDP-7 *(Ken Thompson)*





Above: Joseph Ossanna, Jr. *(Ken Thompson)* **Top right:** Doug McIlroy *(AT&T Archives)* **Center right:** Dennis Ritchie and Bill Joy *(Kirk McKusick)* **Bottom right:** Ken Thompson and Belle *(Ken Thompson)*

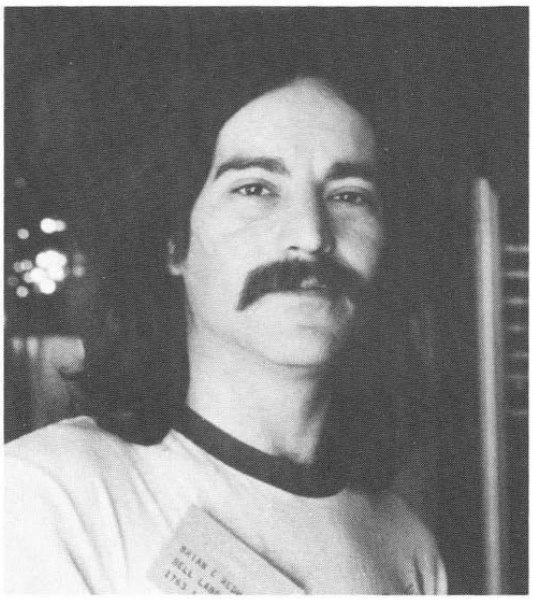


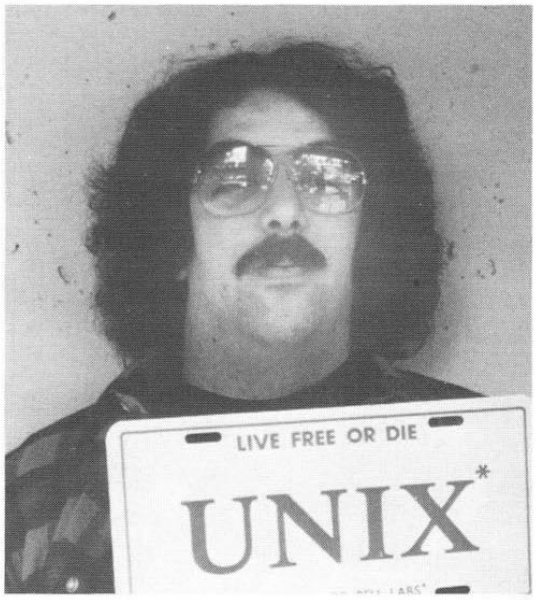


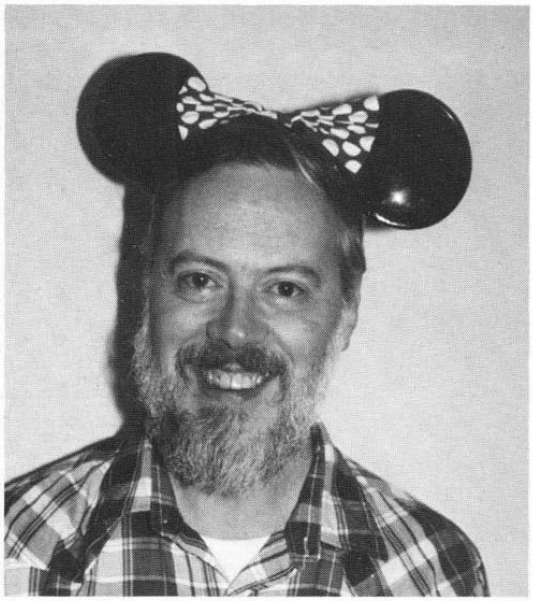
Above: Heidi Stettner and Biff in 1981 *(Carolyn Carr)* **Top left:** Robert Morris and Lorinda Cherry in 1974 *(AT&T Archives)* **Center left:** Rob Pike *(AT&T Archives)* **Bottom left:** Tom Duff *(AT&T Archives)*

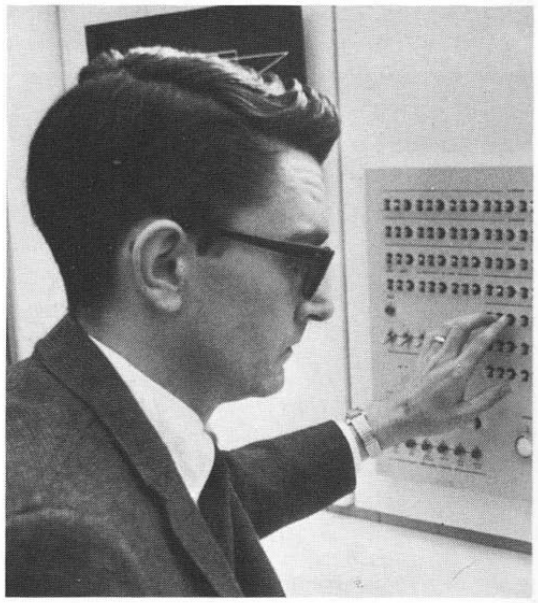
This page top: Debbie Scherrer and Teus Hagen in Dublin, 1987 *(Debbie Scherrer)* **Bottom left:** Evi Nemeth *(Tom Ferrin)* **Bottom right:** Dennis Ritchie *(Kirk McKusick)*

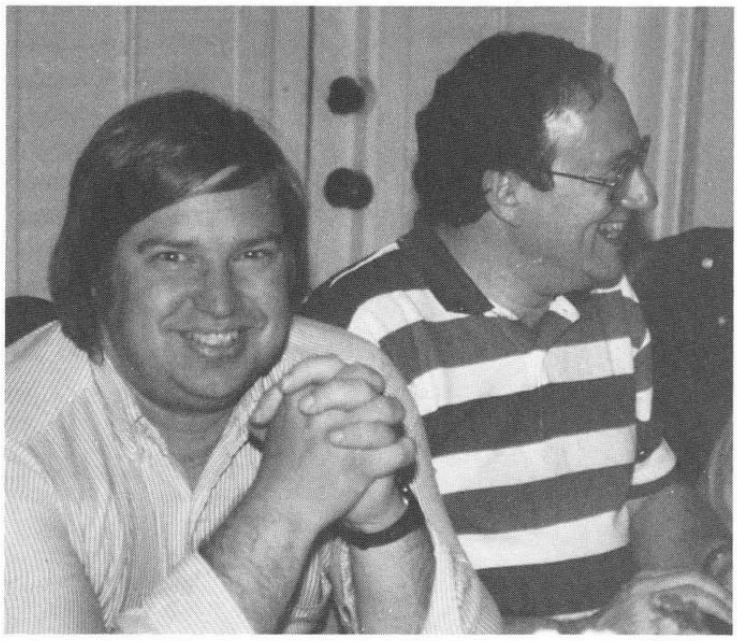
Facing page top: Steve Johnson, Lou Katz and Dennis Ritchie, Atlanta, 1986 *(Kirk McKusick)* **Center left:** Mike O'Dell and Kirk McKusick *(Tom Ferrin)* **Center right:** Tom Ferrin and Mike Karels, Snowbird, Utah, 1984 *(Tom Ferrin)* **Bottom:** Bill Joy in his Ferrari *(Kirk McKusick)*



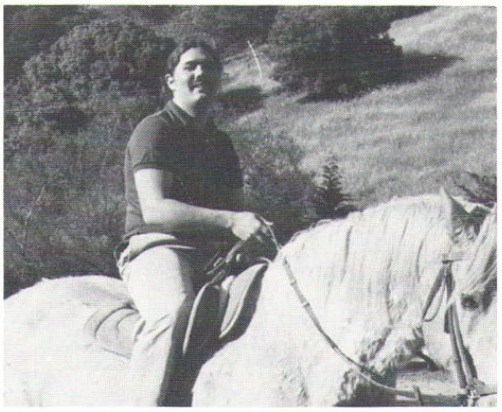




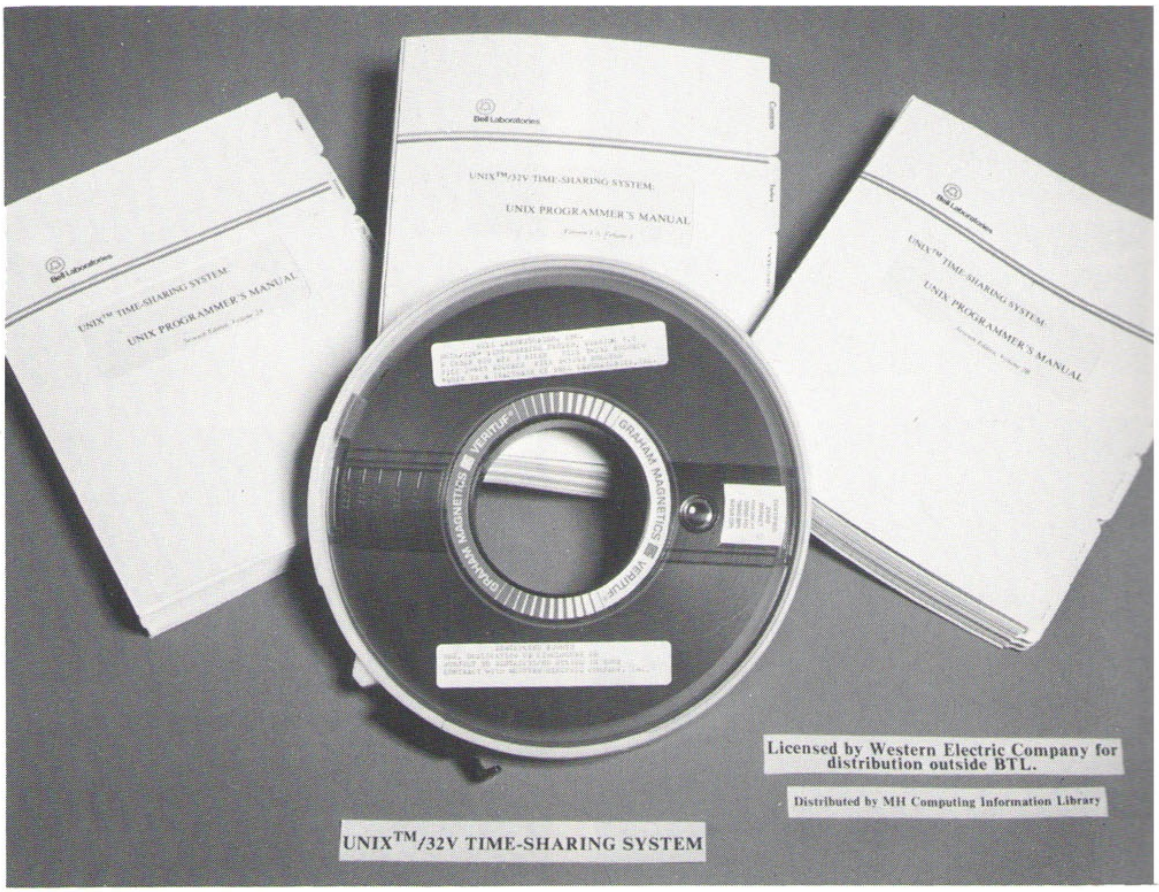




Top right: Armando Stettner *(Lou Katz)* **Top left:** Brian Redman *(Lou Katz)* **Center left:** Dennis Ritchie with Minnie Mouse ears, Anaheim Wizard's Party, 1990 *(Debbie Scherrer)* **Center right:** Lee McMahon *(AT&T Archives)* **Bottom:** Rob Kolstad and Alan Nemeth *(Tom Ferrin)*



The UNIX Equestrian Team **First row left:** Debbie Scherrer **Right:** Ed Gronke **Second row left:** David Tilbrook **Right:** Peter Collinson **Third row left:** JR Oldroyd **Center:** Jaap Akkerhuis **Right:** Dan Klein **Fourth row:** Melinda Stone *(Debbie Scherrer)*



Top: The UNIX/32V Distribution *(Charlie Roberts)* **Bottom (left to right):** Rob Kolstad, Steve Johnson, Kirk McKusick, Debbie Scherrer, Tom Ferrin, Lou Katz, Dave Yost, Peter H. Salus, Mike O'Dell and Lew Law, Atlanta, 1986 *(Kirk McKusick)*