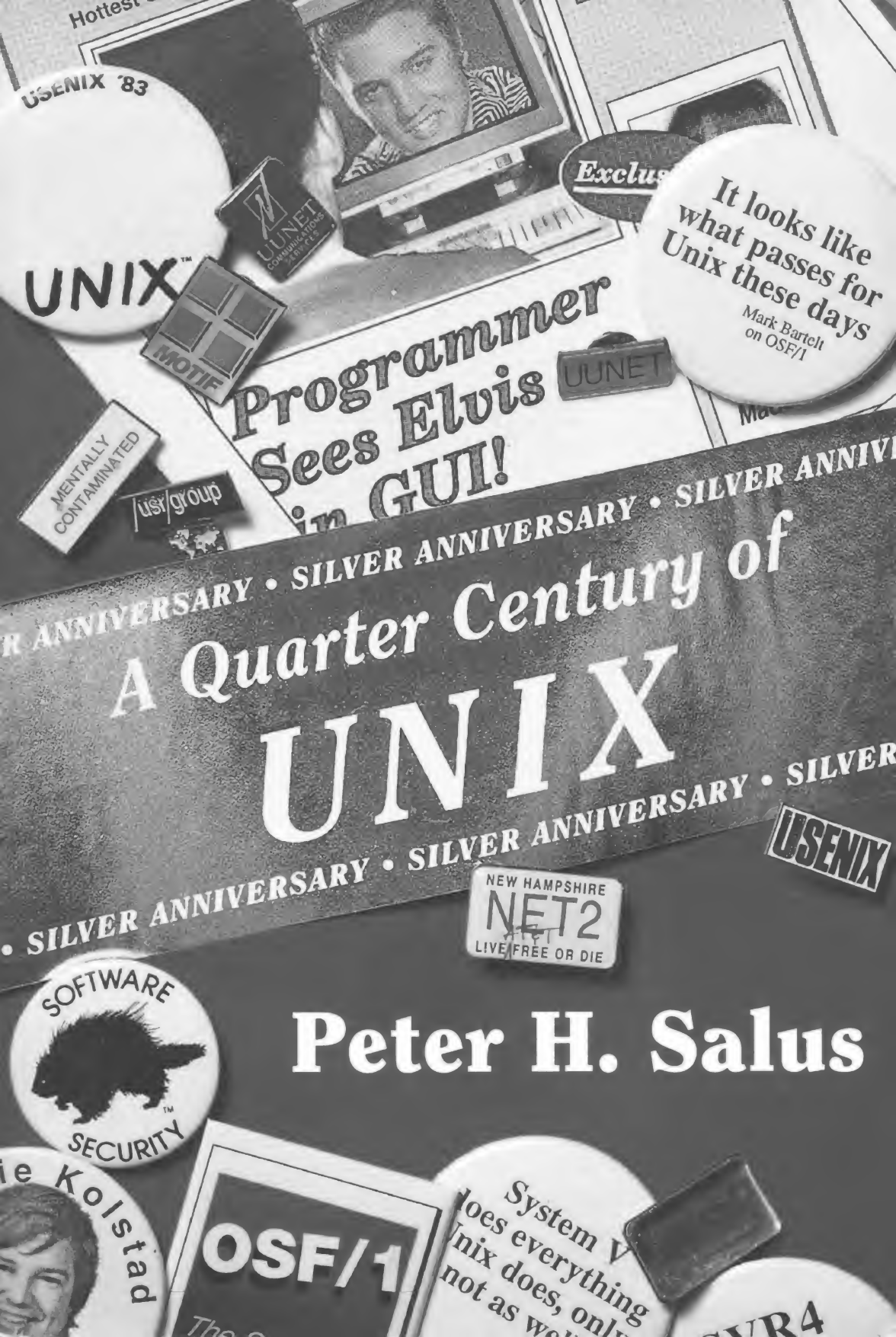

USENIX '83
UNIX
UUNET
MOTIF
Programmer Sees Elvis in GUI!
It looks like what passes for Unix these days
Mark Bartelt on OSF/1
MENTALLY CONTAMINATED
/usr/group
A Quarter Century of UNIX
SILVER ANNIVERSARY • SILVER ANNIVERSARY • SILVER ANNIVERSARY
NEW HAMPSHIRE
NET2
LIVE FREE OR DIE
USENIX
SOFTWARE SECURITY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

/* You are not expected to understand this */

```
if (rp -> p_flag & SSWAP) {
        rp -> p_flag =& ~SSWAP;
        aretu (u.u_ssav);
}
```

lines 2240-2243; Sixth Edition© Western Electric Co.;
the comment is line 2238.

Peter H. Salus

Addison-Wesley Publishing Company

Reading, Massachusetts • Menlo Park, California • New York
Don Mills, Ontario • Wokingham, England • Amsterdam • Bonn
Sydney • Singapore • Tokyo • Madrid • San Juan • Milan • Paris

Several of the photos in the insert are courtesy of AT&T Archives.

Library of Congress Cataloging-in-Publication Data
Salus, Peter H.
A quarter century of UNIX / by Peter H. Salus.
p. cm.
Includes index.
ISBN 0-201-54777-5
1. UNIX (Computer file) I. Title.
QA76.76.063S342 1994
005.4'3--dc20 93-38281
CIP

1 2 3 4 5 6 7 8 9 10-MA-969594

Series Foreword

Marshall Kirk McKusick
John S. Quarterman

Addison-Wesley is proud to publish the **UNIX and Open Systems Series.** The primary audience for the Series will be system designers, implementors, administrators, and their managers. The core of the series will consist of books detailing operating systems, standards, networking, and programming languages. The titles will interest specialists in these fields, as well as appeal more broadly to computer scientists and engineers who must deal with open-systems environments in their work. The Series comprises professional reference books and instructional texts.

Open systems allow users to move their applications between systems easily; thus, purchasing decisions can be made on the basis of cost-performance ratio and vendor support, rather than on which systems will run a user's application suite. Decreasing computer hardware prices have facilitated the widespread adoption of capable multiprocess, multiuser operating systems, UNIX being a prime example. Newer operating systems, such as Mach and Chorus, support additional services, such as lightweight processes. The Series illuminates the design and implementation of all such open systems. It teaches readers how to write applications programs to run on these systems, and gives advice on administration and use.

The Series treats as a unified whole the previously distinct fields of networking and operating systems. Networks permit open systems to share hardware and software resources, and allow people to communicate efficiently. The exponential growth of networks such as the Internet and the adoption of protocols such as TCP/IP in industry,

government, and academia have made network and system administration critically important to many organizations.

This Series will examine many aspects of network protocols, emphasizing the interaction with operating systems. It will focus on the evolution in computer environments and will assist professionals in the development and use of practical networking technologies.

Standards for programming interfaces, protocols, and languages are a key concern as networks of open systems expand within organizations and across the globe. Standards can be useful for system engineering, application programming, marketing, and procurement; but standards that are released too late, cover too little, or are too narrowly defined can be counterproductive. This series will encourage its readers to participate in the standards process by presenting material that details the use of specific standards to write application programs, and to build modern multiprocess, multiuser computing environments.

Newer operating systems are implemented in object-oriented languages, and network protocols use specialized languages to specify data formats and to compile protocol descriptions. As user interfaces become increasingly sophisticated, the level at which they are programmed continues to evolve upward, from system calls to remote procedure call compilers and generic description environments for graphical user interfaces. The effects of new languages on systems, programs, and users are explored in this series.

Series Editors
John S. Quarterman
Marshall Kirk McKusick

UNIX AND OPEN SYSTEMS SERIES

Network Management: A Practical Perspective	Allan Leinwand, Karen Fang
UNIX, POSIX, and Open Systems: The Open Standards Puzzle	John S. Quarterman, Susanne Wilhelm
Practical Internetworking with TCP/IP and UNIX	Smoot Carl-Mitchell, John S. Quarterman
Programming under Mach	Joseph Boykin, David Kirschen, Alan Langerman, Susan LoVerso
The Internet Connection: System Connectivity and Configuration	John S. Quarterman, Smoot Carl-Mitchell
A Quarter Century of UNIX	Peter Salus

Preface

Every book I have seen on modern operating systems or on Unix has an obligatory two or three pages on the history of the system. By 1992, I had read about two dozen of these. None was completely accurate. Some were hilariously in error. At the January 1993 USENIX Association conference in San Diego, Greg Rose gave an excellent one-hour talk on history. But it, too, was incomplete. After talking to Greg, John Quarterman, and several others, I decided to embark on a true history if I could get the cooperation of the actual players.

I cannot begin to express my thanks to Dennis Ritchie, Doug McIlroy, Kirk McKusick, Lou Katz, Peter Collinson, and John Lions, who have responded to my pettiest and inanest queries with patience and grace. My thanks also go to Jaap Akkerhuis, Eric Allman, Keith Bostic, Lorinda Cherry, Clem Cole, Mike Cole, George Coulouris, Jim Curry, Marc Donner, Tom Duff, Robert Elz, Stu Feldman, Mel Ferentz, Tom Ferrin, Dan Forsyth, Michael Gschwind, Teus Hagen, Brian Harvey, Paula Hawthorn, Peter Honeyman, Andrew Hume, 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 Poel, 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 Ubell.

My special thanks go to Mike Mahoney and Peter Collinson for permitting me to quote from their interviews with several of the participants; to Dan Appelman for reviewing the "legal" chapters; and to Stuart McRobert, Mike O'Dell and Len Tower Jr. for reading and commenting on the manuscript as it developed. My thanks, too, to the several anonymous reviewers and to Tom Stone and Kathleen Billus at Addison-Wesley. There are too many others for me to list here. This book is my thank you note to all of them.

Finally, my gratitude to Mary W. Salus and Emily W. Salus, who have always been my severest (and therefore most valuable) critics. This work is dedicated to Rosa Lustig Kubin (Ph.D., Vienna, 1931) who, nearly 50 years ago, introduced me to the idea of science.

P.H.S., Boston

Acknowledgments

The citations noted as "told to Mahoney" are from AT&T Bell Laboratories, "The Unix Oral History Project: Release.0, The Beginning," ed. and transcr. by Michael S. Mahoney (Typescript distributed by AT&T at UNIX Expo, 1989); the interviews with Peter Collinson previously appeared in *.EXE*. I am indebted to both Professor Mahoney and Dr. Collinson for their kind permission to reprint their work here.

Doug McIlroy told me that he thought spelling UNIX with capital letters, rather than Unix, had been a grave error. I have adopted his suggestion and used Unix in most places.

Contents

Introduction

The story of the growth and development of the UNIX operating system is the tale of one of the major advances in computing. Unix demonstrated that a (relatively) small system running on an affordable computer could be employed on other hardware platforms—that it was portable and effectively machine-independent. Sunil Das has noted that, "Technically, Unix is a simple, coherent system which pushes a few good ideas to the limit."

The greatest virtues of Unix, in my opinion, are those that emerged as a result of the way that it developed. Rather than being the product of a manufacturer with hardware to sell, Unix grew from the desire of a few individuals to build a system that was simple, that would support more than one user, and that would serve as a comfortable environment within which those users could program. This soon grew to supporting document preparation as well.

The result is that Unix has influenced every operating system on sale today.

But the growth and development of Unix is an exciting sociological tale. The community was closely-knit, first within Bell Labs Research, then within the telephone community, then the university and research community. All of this took place as the corporate entities of Western Electric and AT&T resisted wide distribution with might and main. That very resistance welded the user community into an ever more participatory whole. As the system had to satisfy both

the research community and the users, it incorporated utilities for both groups from the start.

In some sense, the history of Unix is the history of three groups: Bell Telephone Laboratories' Research Division, the Computer Systems Research Group of the University of California at Berkeley, and the UNIX Systems Group (and its descendants) at AT&T. From a different view, there are five groups: BTL Research, CSRG, USG, the users, and the outside world. I have tried to give appropriate weight to each in the following.

Perhaps the most important contribution to the proliferation of Unix was the growth of networking, the development of various protocols, and the chaotic nature of Usenet and the Internet.

Since the late 1970s, Unix has had a profound influence on every other operating system: DOS, Apple OS, Windows NT, etc., have all drawn heavily from Unix (as Unix did from CTSS and Multics). Windowing, multitasking, networking would not be what they are without Unix.

The following is not a history of programming. It is a sociological narrative. The nature of the individuals and their interactions is what made Unix vital.

Genesis

CHAPTER

Prelude to Space

Trapped in the plane of the ecliptic, between Mars and the asteroid belt, Ken punched a complicated maneuver into his navigational computer and glanced at the makeshift screen in front of him. <n> [stop], <f> [front], <l> [left], <n>. Nothing. A few seconds later, he discerned barely noticeable activity.

An experienced space pilot, Ken knew the vagaries of gravity-free maneuvers. He understood how one could "slingshot" around a planet or a moon or even a planetoid using gravity and acceleration to yield a hyperbolic orbit. He could even cope with three-body problems.

Dammit, Ken muttered. Damn the folks who authorized equipment. Damn everyone in an administrative job. Damn the bean counters. It was then he realized that this was no way to play Space Travel. The machine wasn't up to it. The software wasn't up to it. And he was going to solve the problem.

It was early summer 1969. The heat was hovering in central New Jersey like a blanket ready to smother everything.

After studying at the University of California at Berkeley, Ken Thompson had joined the technical staff at Bell Telephone Laboratories (BTL) in Murray Hill, NJ, in 1966. He'd been working as part of a team on Multics, a joint attempt by General Electric, BTL, and the Massachusetts Institute of Technology to create an operating system for a large computer that could accommodate up to a thousand simultaneous users. (Multics stood for **Multiplexed Information and Com-**

puting Service.) Also involved with the Multics project at BTL was Dennis Ritchie (who had joined BTL in 1968 after completing his doctorate in applied mathematics at Harvard and who had worked part-time on Multics while in graduate school), Joseph F. Ossanna, Jr., Stu Feldman, Doug McIlroy, and Bob Morris. But BTL had just withdrawn from the project, having spent several million dollars with no visible result. Multics couldn't even accommodate three users efficiently. Computer research at BTL was in the doldrums.

Most of the small group was just treading water. Ken, with a little help from his friends, was about to change the future of computing.

CHAPTER

1

Summer 1969–Fall 1970

From the commemorative plate that Mike Tilson, then of Human Computing Resources in Toronto, had made in 1987, the bearded faces of Ken Thompson and Dennis Ritchie look forth. The plate is labelled "Our Founders." In any general sense, this is correct, but to be precise, Thompson was the initiator, with Ritchie and Rudd Canaday very close by and with a small band of BTL staff making up the cohort. They were a compact, friendly group. While the opening tale is merely fiction, it is not dramatically off the mark.

When I asked Dennis Ritchie why Ken was working on Space Travel, he responded: "Just for fun." (Actually, Space Travel was a serious astronomical simulation program, not *merely* a game.)

To a certain extent, that was the attitude that prevailed at BTL. The basic notion at the Labs (in Ritchie's words),

> was and is to hire people who generate their own good ideas and carry them out.... Ken was doing something interesting that would turn out to be valuable.... When a good university hires young professors, what do they expect them to do? Well, a certain amount of grot and service of various kinds, but really to have good ideas that somehow make an impact. The details and flavor of the grot vary between academia and here, but the ultimate expectation is surprisingly similar. People are given a lot of freedom to blaze their own path. Not all of them find the path, of course, though we've been fortunate in our selections.

Ken Thompson had been working on the Multics machine, a GE-645, when Bell's involvement in the project ceased in March 1969. He continued to work on it, "just for fun." However, having fun meant trying to work, and that meant coming up against the limits of both the hardware and the software. The GE was an expensive machine, and Multics wasn't ready to be used as a production environment for anything.

Doug McIlroy remarked, "When Multics began to work, the very first place it worked was here.... BTL actually used the 645 as a Multics machine. Three people could overload it."

While he was still at Berkeley, Thompson had been exposed to the SDS930 operating system that Butler Lampson was writing. But research on operating systems (OS) wasn't popular in New Jersey after the Multics disaster.

Sandy Fraser came to Bell in May 1969. He had left Ferranti specifically because he wanted to work on Multics, but arrived to learn that BTL had chopped it. One day Bill Baker (the Vice President for Research) and Ed David (the liaison to the Multics project) had come to the fifth floor of BTL in Murray Hill where David read a letter informing the staff that Bell was withdrawing from the project. Fraser recalled that this was a "traumatic change" and that people were quite down. "The toy was gone ... there was a clear lack of momentum," he said. During April, May, and June, Thompson was interested in writing a file system. As he told Mike Mahoney:

> It was never down to a design to the point of where you put the addresses and how you expand files and things like that; it was never down to that level. I think it was just one or two meetings. Dennis and [Rudd] Canaday and I were discussing these ideas of the general nature of keeping the files out of each other's hair and the nitty-gritty of expanding, of the real implementation: where you put the block addresses, where you put... We did it in Canaday's office, and at the end of this discussion, Canaday picked up the phone; there was a new service at Bell Laboratories that took dictation. You call up essentially a tape recorder and you give notes, and then the next morning the notes are typed and sent to you. The next

> day these notes came back, and all the acronyms were butchered, like 'inode' was 'eyen...'. So we got back these...descriptions and they were copied, and we each had copies of them and they became the working document for the file system—which was just built in a day or two on the PDP-7 [a Digital Equipment Corporation computer].
>
> At first...we'd used it for other things, you know, the famous Space Travel game, and it was the natural candidate as the place to put the file system. When we hacked out this design, this rough design of the file system on the dictation [machine] that day in Canaday's office, I went off and implemented it on the PDP-7.

Bob Morris, who joined Bell Labs in 1960, has said that he considers Canaday's contribution to UNIX "the most underrated" of the original participants: at the beginning, it was "Ken, Dennis, and Rudd," he said, "The others came one, two, three years later.... I was there, but not involved." Steve Bourne, who was at Bell Labs in the 1970s, put it a bit more formally:

> A cast-off PDP-7 with a 340 display was available but the PDP-7 provided only an assembler and a loader. One user at a time could use the computer, each user having exclusive use of the machine. This environment was crude and parts of a single user UNIX system were soon forthcoming. The space travel program was rewritten for the PDP-7 and an assembler and a rudimentary operating system kernel were written and cross assembled for the PDP-7 on GECOS [General Electric Comprehensive Operating System] ...
>
> Cross assembling meant using two computer systems and carrying paper tapes from one to the other each time a change was made. ... The system supported two people working at the same time and the term UNICS was apparently coined by Peter Neumann, an inveterate punster, in 1970. [UNiplexed Information and Computing Service, was a pun on 'emasculated Multics'; several people have told me that Brian Kernighan changed the spelling, but Kernighan said that no one recalls whose idea the change to UNIX was].

The PDP-7 belonged to Joe Condon's group. It had been "bought for a graphics something-or-other that Bill Ninke wanted to do," Condon recalled. "But Ninke went off to Holmdel [another BTL site]," and so it wasn't in use when Thompson and Ritchie wanted to use it.

Ken Thompson recalled:

> It was the summer of '69. In fact, my wife went on vacation to my family's place in California to visit my parents; we just had a new son, born in August '68, and they hadn't seen the kid, and so Bonnie took the kid to visit my family, and she was gone a month in California. I allocated a week each to the operating system, the shell, the editor, and the assembler, to reproduce itself, and during the month she was gone, it was totally rewritten in a form that looked like an operating system, with tools that were sort of known, you know, assembler, editor, and shell—if not maintaining itself, right on the verge of maintaining itself, to totally sever the GECOS connection.... Yeh, essentially one person for a month.

That was an actual man-month. Thompson continued:

> It didn't exist by itself very long. What we did was—to run the file system you had to create files and delete files and read and write files and see how well it performed. To do that, you needed a script of what kind of traffic you wanted on the file system, and the script we had was paper tapes that said "read a file," "create a file," "write a file,"... And you'd run the script through the paper tape and it would rattle the disk a little bit, and you wouldn't know what happened. You just couldn't look at it, you couldn't see it, you couldn't do anything. So we built a couple of tools on the file system to—we used the paper tape to load the file system with these tools, and then we would run the tools out of the file system; that's called *exec,* by the way,—and type at these tools, and that was called the *shell*—to drive the file system into the contortions that we wanted to measure; how it worked and reacted. It only lasted

by itself for maybe a day or two before we started developing the things we needed to load it.

While there were things in Multics that influenced UNIX, there were also, as Ritchie put it, "profound differences":

> We were a bit oppressed by the big system mentality. Ken wanted to do something simple. Presumably, as important as anything was the simple fact that our means were much smaller—we could get only small machines with none of the fancy Multics hardware.
>
> So UNIX wasn't quite a reaction against Multics, it was more a combination of these things. Multics wasn't there for us any more, but we liked the feel of interactive computing that it offered; Ken had some ideas about how to do a system that he had to work out; and the hardware available as well as our inclinations tended to trying to build neat small things, instead of grandiose ones. Multics colored the UNIX approach, but didn't dominate it one way or the other, toward an anti-Multics system, or a copy on the cheap.

Thompson had "scarfed up this PDP-7 and he (mostly) did this neat stuff with it," Ritchie said. But Ritchie is over-modest. Thompson and he had created a new toy. They had begun something in the summer and early autumn of 1969 that would give direction to computer research at BTL for a decade and an experimental platform for another one. The toy would initiate work on a new system all over the world. But, at the outset, they had only the PDP-7 to work with: "the hardware was borrowed, and badly obsolete, to boot," Ritchie told me.

CHAPTER

2

Calculating and Computing

While I have no intention of attempting a history of the computer, or even a history of operating systems, it is necessary for me to sketch something here, because I don't want to lose the non-specialist.

Three different threads must be followed to understand the source of the modern computer:

- Mechanical calculation
- Tabulation and sorting
- Difference and analytical engines

Once these threads were interwoven, around the end of the First World War in 1918, technology moved from the mechanical to the electro-mechanical (Aiken), to the electronic (Eckert and Mauchly), to the transistorized (IBM 7090), to the micro-electronic computer-on-the-board (Altair 8008), to the computer-on-a-chip of today.

Mechanical methods for calculating are very old. The abacus is probably the oldest of such constructions. The Chinese and Egyptians had this device nearly four millennia ago. The Mayans possessed it when the Spanish arrived. It was only a few years after Napier's discovery of logarithms (1614), and his "bones" (marked ivory rods) for multiplication, that the slide rule was invented. (There is a set of Napier's

bones in the Victoria and Albert Museum in London that belonged to Charles Babbage.)

In 1642, at the age of 18, Blaise Pascal invented a calculator that could add and "carry" to aid his father, a tax collector. Almost 30 years later, in 1671, Leibniz took Pascal's machine a step further and built a prototype machine that could multiply using an ingenious device called the stepped wheel, which was still in use in the last mechanical calculators manufactured in the late 1940s. Leibniz demonstrated his calculator to the Royal Society in London in 1676. The first commercially successful calculator was invented by Charles Xavier Thomas in 1820. By 1878, an astounding 1,500 had been sold—nearly 30 per year. They were still being manufactured by Darras in Paris after the First World War. The Brunsviga adding machine, based on an 1875 patent by Frank Stephen Baldwin, which substituted a wheel with a variable number of protruding teeth for the Leibniz stepped wheel, sold an incredible 20,000 machines between 1892 and 1912—1000 per year.

The first keyboard-driven calculator was patented in 1850 by D.D. Parmalee and the Comptometer of Dorr Eugene Felt—the first successful key-driven, multiple-order calculating machine—was patented in 1887.

In 1812, Charles Babbage came up with a notion for a different type of calculator, which he termed a difference engine. He was granted support by the British government in 1823. Work stopped in 1833, and the project was abandoned in 1842, the government having decided the cost was too great. From 1833 on, though, Babbage devoted himself to a different sort of machine, an analytical engine, that would automatically evaluate any mathematical formula. The various operations of the analytical engine were to be controlled by punched cards of the type used in the Jacquard loom. Though only a fraction of the construction appears to have been effected, Babbage's notes, drawings, and portions of the engine are in the Victoria and Albert Museum.

The Jacquard loom, a successful attempt at increasing production through automation, was itself the result of several prior innovations: in 1725 Bouchon substituted an endless paper tape with perforations

for the bunches of looped string. In 1728 Falcon substituted perforated cards, but attached them to strings, and in 1748, Jacques de Vaucanson combined the bands of perforated paper and the cards. The patterns on the cards were perforated by special machines that cut on designs painted on by stencils. The programmed machine was born.

Over a hundred years later, Herman Hollerith, a graduate of Columbia College, recalled the existence of those perforated cards. Hollerith had just started work at the Census Bureau at a generous salary of $600 per year. There he was put to work on a survey of power and machinery used in manufacturing. But he also met John Shaw Billings, who was in charge of "vital statistics." One night at dinner, Billings complained about the recently invented but inadequate tabulating device of Charles Seaton, which had been used for the census of 1870. Billings felt that given the increased population, the 1880 census might well not be completed in less than seven or eight years, and the the 1890 census would still be incomplete in 1900. "There ought to be a machine for doing the purely mechanical work of tabulating population and similar statistics," Billings said. "We talked it over," Hollerith recalled 30 years later, "and I remember ... he thought of using cards with the description of the individual shown by notches punched in the edge of the card." Hollerith thought about the construction of a device to record and read such information and asked Billings to go into business with him. Billings was a cautious man and said no.

In 1882 Hollerith went to MIT as an instructor in mechanical engineering (he was then 22). Teaching at MIT gave him the time to work on his machine. He first considered putting the information on long strips of paper, but this proved impractical. In the summer of 1883, Hollerith took a train trip West. On the train he saw the "punch photograph," a way for conductors to punch passengers' descriptions onto tickets, so they could check that the same individual was using the ticket throughout the trip (things like gender, hair and eye color, etc., were used).

Hollerith patented his first machine in 1884 and an improved design in 1886, when he performed a trial by conducting the Baltimore census. On the basis of reports of the trial, New Jersey and New York placed orders for machines (to tally mortality rates). Hollerith and

some business colleagues bid for the contract for the 1890 census and won it. The government of Austria ordered machines in 1890. Canada ordered five the next year. Italy and Norway followed, and then Russia. The machines were a clear success. Hollerith incorporated his Hollerith Electric Tabulating System as the Tabulating Machine Company in 1896; reincorporating it in 1905.

Nearly 80 years passed before the computer industry moved beyond several of Hollerith's insights. First, so that operators would have no problem orienting the cards, he cut a corner from the upper right. Second, he *rented* the machines at a reasonable rate (the rental fees for the 1890 census were $750,000; the labor cost in 1880 had been $5 million), but *sold* the patented cards (over 100 million between 1890 and 1895). Third, he adapted the census-counting to tally freight and passenger data for railroads. Effectively, Hollerith invented reusability.

In 1907 and 1908 Hollerith rented some of his machines to electric utility companies for billing purposes: he sold a million cards a month to them. The cards were a standard size: 3.250 by 7.375 inches (the same size as the "old" dollar bill), they had 45 rows and 10 columns of round holes.

In the meantime, an entrepreneur named Charles Flint started the International Time Recording Company in Endicott, NY, in 1900. It manufactured time clocks. The next year, he cobbled together the Computing Scale Company of America, producing scales that "read out" costs as well as weights, thus doing away with the need for the clerk to be able to calculate. In 1910—just how or when is not recorded—Flint met a cash-poor Hollerith. The merger yielded the Computing-Tabulating-Recording (CTR) Company. Less than 30 months later, Thomas Watson, recently fired from NCR, was interviewed by Flint and then by Flint's tame board of directors. In May 1914, Watson took over as General Manager of CTR, at a munificent salary of $25,000, stock options, and a share of the profits.

While the First World War gave rise to business that was good to CTR and Watson, it was in 1919 that Watson's investment in development five years earlier paid off: CTR successfully unveiled a printer-lister, which printed out the information gathered from Hollerith's

tabulators and sorters. CTR became the International Business Machines Company with Thomas J. Watson as both chief executive officer and chief operating officer in February 1924.

In 1928, IBM introduced its Type IV Tabulator and its new 80-column, 12-row card.

From 1941 through 1945, the US Department of Defense funded many of the projects that brought modern computing into being. Thomas Watson's company funded most of the others.

Despite the fact that Watson said (in 1945), "I think there is a world market for about five computers," the first completed was one he had funded. Howard Aiken of Harvard and a small team began in 1939 to put together a machine that exploited Babbage's principles. It consisted, when completed in 1944, of a 51-foot by 8-foot panel on which tape readers, relays, and rotary switches were mounted. Nearly all of the operations of the Harvard Mark I Calculator were controlled by mechanical switches, driven by a four-horsepower motor. It could handle numbers in decimal form to 23 significant figures, running on 24-hole punched tape that moved at 200 steps per minute. The base operation time for addition was 0.3 seconds. It was still in use in 1953, and a portion of the Mark I is on display at Harvard's Aiken Computation Laboratory in Cambridge, MA. Another piece is in the Smithsonian Institution in Washington, DC.

In October 1944 the US Army offered a contract to the University of Pennsylvania to build an Electronic Discrete Variable Computer (EDVAC). The original team consisted of Herman H. Goldstine, John von Neumann, and John Mauchly. Its design was thoroughly von Neumann's.

The first all-electronic computer was the Electronic Numerical Integrator and Calculator (ENIAC). Completed by J.W. Mauchly and J.P. Eckert of the University of Pennsylvania in late 1945 and installed in 1946, it was commissioned by the Ballistics Research Laboratory (BRL) at the Aberdeen (Maryland) Proving Ground. It was—and will remain, I expect—the largest computing machine ever built: it was made up of 18,000 tubes (valves) and 1,500 relays. ENIAC was the electronic analogue of the Mark I, but ran several hundred times faster. (Mike Muuss tells me that parts of ENIAC are on display at the BRL.)

Eckert and Mauchly formed their own computer company in 1947 and began building their UNIVersal Automatic Computer (UNIVAC). They ran out of funds and succeeded in obtaining a $500,000 investment from the American Totalizator Company (in some way, this must have been the earliest connection between computing and games, for Totalizator provided parimutuel betting machines, and thought that computers might be useful in information processing). American Totalizator took control of Eckert-Mauchly, but funds ran out again before the first machine was completed, and the Mun Brothers (who owned Totalizator) sold Eckert-Mauchly to James Rand of Remington Rand. The initial UNIVAC was delivered (a year late) in 1951, with six more on order.

ENIAC had offspring in England, too. Maurice V. Wilkes and his group began planning their Electronic Delay Storage Automatic Calculator (EDSAC) in late 1946, on Wilkes' return from Pennsylvania, and began work at the University Mathematical Laboratory in Cambridge early in the new year. It was one-fifth the size of ENIAC and based on ideas that von Neumann had presented in a paper. When it performed its first fully automatic calculation in May 1949, EDSAC became the first electronic machine to be put into operation that had a high-speed memory (store) and with I/O (input/output) devices. Within a few years, EDSAC's library contained over 150 subroutines, according to Wilkes.

At virtually the same time, in Manchester, a team under M.H.A. Newman began work on a machine that was to embody the EDVAC concepts. F.C. Williams, who invented cathode ray tube storage, I.J. Good, who had worked on the Colossus code-breaking machine with Alan M. Turing, and Turing himself, joined the team. The Manchester Automatic Digital Machine prototype was built in 1948 and the definitive machine ran its first program in June 1949. MADM introduced both the index register and pagination to computing.

In the meantime, IBM had begun work on its Selective-Sequence Electronic Calculator (SSEC). It is important to remember that while EDSAC was the first electronic computer, the SSEC was the first *computer*—it combined computation with a stored program. It was put into operation at IBM headquarters in Manhattan early in 1948, clev-

erly placed behind plate glass windows at street level, so that pedestrians could see it operate. It was a large machine with 13,000 tubes and 23,000 relays. As all the arithmetic calculations were carried out by the tubes, it was more than 100 times as fast as the Mark I. It also had three different types of memory: a high-speed tube store, a larger capacity in relays, and a vastly larger store on eighty-column paper tape. Instructions and input were punched on tape and there were 66 heads arranged so that control was transferred automatically from one to the other. "It was probably the first machine to have a conditional transfer of control instruction in the sense that Babbage and [Ada] Lady Lovelace recommended," wrote B.W. Bowden in 1953. It did work, among other things, for the Atomic Energy Commission, before being dismantled in August 1952.

That very June, John von Neumann and his colleagues completed Maniac at the Institute for Advanced Studies in Princeton. It employed the electrostatic memory invented by F.C. Williams and T. Kilburn, which required a single cathode ray tube, instead of special storage tubes.

The next advance in hardware came at MIT's Whirlwind project begun by Jay Forrester in 1944. Whirlwind performed 20,000 single-address operations per second on 16-digit words, employing a new type of electrostatic store in which 16 tubes each contained 256 binary digits. (The *word* is the basic unit of data in computer memory. Units consist of predetermined numbers of bits. An 8-bit word can hold numbers in the range of -2^7 to $+2^7 -1$ [−128 to +127].) The Whirlwind was the first attempt at real-time computing, continuously calculating aircraft motion to solve problems of both stability and high-speed flight. Doug McIlroy told me that in 1954, when he was a Whirlwind user, "it had core memory and could execute 40,000 instructions per second." One of Forrester's "bright boys" on the Whirlwind project was Kenneth H. Olsen, who was to found the Digital Equipment Company a few years later. Olsen was experimenting with magnetic switch-core matrices for memory devices, but in a student paper he suggested using transistors instead of vacuum tubes. This paper was read by Ralph L. Palmer, the circuit designer of much of the SSEC, who, early in 1955, urged Erich Bloch to use transistors in the memory for the IBM 704. The attempt failed because transis-

tors with sufficient power to drive the ferrite-core memories just weren't available.

The 704, originally the 701A, was released in 1954. It was the logical successor to the IBM 701 (1952, 1953). The evolution of the 701 into the 704 was headed up by Gene Amdahl. The 701 was built by a team of 35, led by Nathaniel Rochester and Jerrier Haddad. Rochester had built the arithmetic unit for the Whirlwind while at Sylvania; Haddad was part of the team that had designed the IBM 604 electronic calculator. Eventually, the 700-series contained both scientific computers (the 701, 704, 709, 7040, 7044, 7090, and 7094) and business computers (the 707, 705, and 7080). (The IBM 7030, incidentally, had 64-bit words and 8-bit bytes.) The 701 rented for $15,000 per month in 1953; the 704 incremental cost was minor. But even at such a high rental, IBM shipped 18 701s and later sold 140 704s. On the business side, 175 IBM 705s were sold at an average cost of $1.6 million. The 7080 was a fully transistorized version. It sold at $2.2 million: computing was still in the multi-million dollar cost bracket.

Forrester, in collaboration with IBM, received a contract from the Defense Department for Whirlwind II, an air-defense computer. With IBM's increasing participation, the project was renamed SAGE (Semi-Automatic Ground Environment) in 1952. Ken Olsen and his team had designed and built the Memory Test Computer at MIT's Lincoln Labs in just nine months. When Forrester and his lieutenant, Norman Taylor, realized in early 1953 that they needed a full-time liaison person on site at IBM, Taylor twisted Olsen's arm. Olsen went. And he grew to hate the IBM attitudes. One night in Poughkeepsie, late in 1953, he told Taylor: "Norm, I can beat these guys at their own game." It was the beginning of the Digital Equipment Corporation (DEC).

Three-and-a-half years later, Olsen presented a business plan to American Research & Development, and received the $70,000 he and Harlan Anderson wanted to start their company. A year later they were shipping logic modules from Maynard, MA. In December 1959, at the Eastern Joint Computer Conference at the Statler Hotel in Boston, DEC unveiled the prototype of its PDP (Programmed Data Processor) 1. It was priced at $120,000 and deliveries began in November 1960.

The PDP-1 was an 18-bit machine with a memory capacity of between 4096 and 32,768 words. The PDP-1 had a memory cycle of 5

microseconds and a computing speed of 100,000 computations per second. It was the result of a project led by Benjamin Gurley and was composed of 3,500 transistors and 4,300 diodes. It had an editor, macroassembler, and DECAL, an ALGOL compiler. It employed a paper tape reader for input and an IBM typewriter for output. The PDP-1 had the best cost/performance of any real-time computer of its generation. It was also the first commercial computer to come with a graphical display screen. DEC sold 53 of them. The first went to Bolt, Beranek and Newman, the second to Itek (Norm Taylor's start-up typesetting house). Others went to Lawrence Livermore Labs and Atomic Energy of Canada (AEC). In January 1962, Olsen donated a PDP-1 to MIT. The students programmed it to play mancala, a counting game.

The PDP-2 was supposed to be a 24-bit machine; the PDP-3 was intended to be 36-bit. Neither was ever built. The PDP-4, another 18-bit machine, designed by Gordon Bell, was a failure. Its memory cycle was 8 microseconds (as opposed to 5 for the PDP-1); it required only half the wattage of the PDP-1 (1,125 vs. 2,160); it ran assembler, editor, and FORTRAN; and it cost only $65,000, as opposed to $120,000. But Gordon Bell had been wrong: 5/8 the power (62.5%) at 13/24 the price (54%) wasn't what the market wanted. Only 45 were sold.

The PDP-5 was another Gordon Bell creation, designed by Edson de Castro to act as the front-end of a PDP-4 that was in use at the AEC site in Chalk River, Ontario. It was a small, general-purpose 12-bit machine. DEC was going to build ten machines, which was calculated to earn back the cost of engineering. But at $27,000, the PDP-5 was a hit and DEC eventually sold nearly 1,000.

The PDP-6 was a large 36-bit machine that cost $300,000 and was killed soon after release. Twenty-three were shipped, one to MIT's AI Lab.

The PDP-7, the PDP-9, and the PDP-15 were 18-bit followers of the PDP-4. The PDP-8 was another Bell/de Castro 12-bit computer. It initiated the minicomputer revolution when it was introduced in 1965 at a mere $18,000. Over 50,000 PDP-8s were eventually sold. The PDP-10 was a large 36-bit follow-on to the PDP-6.

But it was the PDP-11, conceptualized by Bell while he was on sabbatical at Carnegie-Mellon University and with Andy Knowles as project leader, that made DEC the leading minicomputer manufac-

turer when it was released in 1970. It was a 16-bit machine and (in a vast variety of configurations and models) eventually sold 250,000 units.

Gordon Bell's creations and DEC's venture into low-cost computing were the keys to the development and success of UNIX, for the PDP-11 proved to be the first department-sized machine that universities and research sites could afford. Moreover, whereas Ritchie and Thompson couldn't get funds for a large machine, the $10,800 starting-price of the PDP-11 was within BTL's budget. The PDP-11's 16-bit words were important, too. 16-bit words hold more bits than 8-bit words.

CHAPTER

3

Operating Systems

Gene Spafford of Purdue University remarked to me that an operating system "is there to provide more convenient abstractions. One of my former professors, Phil Enslow, used to define an OS as the mechanisms, policies, and procedures that supported the controlled sharing of computing resources." This is a good view, as it includes security policy, scheduling policy, and the like, which are not usually seen as part of the interface between the user and the hardware or as part of the environment within which application programs can be executed. The OS offers convenience to the user and efficiency where the hardware is concerned. It has been compared to a government in that while it serves no useful function itself, it supplies the environment in which other programs can function.

The operating system is a resource allocator; it is also the control program, preventing errors and improper use as well as regulating the operation of the computer and the input/output (I/O) devices.

In the beginning (up to about 1954) there was only hardware. (Stan Kelly-Bootle, a student of Wilkes, told me "I'm biased, but 'tis oft claimed that Swinnerton-Dwyer's EDSAC Monitor circa 1953/4 was the first true OS.") Massive—room-sized—early computers were operated from a console where the programmer would write the program and then manually load it into memory: first by flicking switches on the front panel one instruction at a time, then by punching those instructions on cards or paper tape. The programmer would observe the

flashing of display lights on the console to monitor execution, discover bugs, or examine the contents of the memory or registers. Output was either printed directly or punched onto cards or paper tape for later printing. This was truly "hands-on" and "interactive."

The problems arose as more and more hardware was developed—card readers, line printers, magnetic tape devices—and more and more software—libraries of commonly used functions were deveoloped. (Common functions could then be copied into new programs without the necessity of rewriting. Originally, programs were punched onto paper tape and loaded into the machines in the proper order, one at a time, or actually spliced together in order.)

Every I/O device had its own vagaries. Special subroutines, called *device drivers,* had to be written for each device, which had particular buffers, flags, registers, and special bits.

And then there were compilers. Compilers interpret between higher-level languages and machine language. The first complete compiler was designed by Grace Hopper and her group at Remington Rand in 1952. Later, compilers for FORTRAN, COBOL, etc., were developed. This made the programmer's job easier, but it made the software more complex.

FORTRAN supplies a good example. To prepare a FORTRAN program, the operator would load the FORTRAN compiler into the computer. This entailed mounting the appropriate tape on a tape drive. The program (which had been typed onto punched cards) would then be read by a card reader and written to a (different) tape. The output of the FORTRAN compiler—which was in assembly-language—would then be assembled, which required mounting the assembler tape. The assembler would be linked to the appropriate library routines, and finally the binary output of the program would be executable. It could be loaded and debugged from the console—usually, though, it wasn't.

During all of the job set-up time the CPU was idle. During tape mounting, the CPU was idle. In order to increase efficiency, jobs were *batched* together in groups: those in FORTRAN, those in COBOL, etc. But there were still many problems.

Computer time was too expensive to waste.

Off-line processing, in which jobs were batched on tape prior to mounting the tape on the computer, was the first answer. The CPU and

I/O operations could be overlapped in time by executing them on different machines. But why not execute this overlap on the same machine? And so buffering (temporarily storing data to compensate for the various speeds of different devices) and spooling (decoupling slow devices from the main process) came into being. Spooling enabled the CPU and the I/O to operate at much higher speeds. It also made the formation of a job pool possible, and this brought about job scheduling.

The most important element of job scheduling was the ability to *multiprogram*. Multiprogramming was for the computer what most of us do in reality: execute a chore while waiting for some element of another job to complete: while waiting for the water to boil, I made toast. In the late 1950s, this meant "interrupts." Peripheral devices were started by the central processing unit (CPU), and continued executing their chores automatically. When the chore was completed or the device needed attention, the device sent an interrupt to the CPU, forcing attention to be turned to that device.

Doug McIlroy pointed out to me that there had been "an interesting backward step. When the IBM 704 went into MIT, automatic operation, which had been the rule on Whirlwind, went out."

The logical extension of multiprogramming is multitasking, or *timesharing*. And for that, let's go to MIT.

CHAPTER

4

Project MAC: CTSS and Multics

Project MAC was organized at MIT in the spring of 1963 at the suggestion of J. C. R. Licklider. Its founding director was MIT Professor Robert M. Fano. According to Fano's "Preface" to Project MAC's *Progress Report II* (1965), MAC stood for Machine-Aided Cognition and Multiple-Access Computers. It was, in fact, so known on the fifth floor of Tech Square, the MIT building where computer science was housed. It was known as Man and Computer on the ninth floor, where Marvin Minsky's Artificial Intelligence Lab was located (the Lab was Minsky's and McCarthy's at the outset, but when John McCarthy, the inventor of LISP, moved to Stanford, his name was dropped from the usage).

In 1963 Project MAC hosted a summer study, which brought many well-known computer scientists to Cambridge to use CTSS (the Compatible Timesharing System) on the IBM and to discuss the future of computing. At this time, the major efforts of Project MAC were Fernando J. Corbato's CTSS, Victor H. Yngve's COMIT, and Minsky's AI Lab.

The 1950s and the early 1960s were the era of batch processing. But this was wasteful of both time and resources. The waste came because of the fateful sign-up sheet, omnipresent in computer rooms. Unless you were lucky and at a site that let you do small jobs, like

debugging, by getting in to the queue for a few seconds, you signed up for an hour. Your program ran in 40 minutes. Or, worse, your program wasn't done at the end of the hour: it was then dropped (and, in most cases, the partially completed work was lost), so you signed up for a future hour-and-a-half (frequently at 1 or 2 or 3 a.m.). The answer to this problem was timesharing.

The Compatible Timesharing System (CTSS) was one of the first timesharing systems. Developed at the MIT Computation Center by a team led by Corbato, CTSS ran on a modified IBM 7094 with a second 32K-word bank of memory, using two IBM 7320As. A second 7094 was connected in early 1965. Remote access was provided to up to 30 users via an IBM 7750 communications controller connected to dial-up modems for access over phone lines. (Actually, no one ever had an hour of CPU time, as that would stretch beyond the maximum time between failures of the system!)

CTSS, as the timesharing portion of Project MAC, was both a service facility and a laboratory for system programmers. It was this second function that led—while the 7094 was still being tuned—to Multics (**Multi**plexed Information and Computing Service).

Multics was a second generation timesharing system, intended to be a prototype of the "computer utility." It was started in 1964 as a joint project of Project MAC, Bell Telephone Laboratories, and General Electric. Corbato was the principal leader of the development effort, which had nine major goals:

- Convenient remote terminal use
- Continuous operation like telephone service
- A wide range of system configurations
- A high reliability internal file system
- Support for selective information sharing
- Hierarchical structures of information
- Support for a wide range of applications
- Support for multiple environments and interfaces
- The ability to evolve the system

Berkley Tague, who had joined BTL in 1962 and the Multics group in 1965 (as a "secretary for the project triumvirate") has said that by 1967 it was clear that the project couldn't succeed because the "three parties had incompatible goals." He went off to AT&T in Whippany, NJ, for three years, where he worked on a project called Safeguard, only to return to Murray Hill in 1970.

Professor Jack Dennis of MIT contributed some influential architectural ideas to the beginning of Multics. When it was time to select a vendor for the computer that would support the new operating system, the folklore says that IBM pitched the machine that would become the 360/65. Doug McIlroy told me:

> This is true. But [Gene] Amdahl wouldn't make VM [virtual machine]. At the last instant IBM let Gerry Blaauw out of the back room to pitch the 360/67, but it was too late.

IBM's staff weren't interested in the MAC team's ideas about paging and Joseph Weizenbaum, who was then a lecturer at MIT, introduced the MAC team to former colleagues of his from GE in Schenectady, colleagues who were receptive and enthusiastic about paging and segmentation, and who proposed what became the GE-645 (the upgrade of the 635). Thus GE became part of the Multics effort.

MIT, BTL, and GE agreed on a structure for cooperation. A trinity made major policy decisions, with one person from each organization: Fano (MIT), E. E. David (BTL), and C. W. Dix (GE). A triumvirate was in charge of actual management of the implementation: Corbato (MIT), A. L. Dean (GE), Peter G. Neumann (BTL). Notable features included segmented memory, virtual memory, high-level language implementation (PL/I), shared memory multiprocessor, multi-language support, and on-line reconfiguration.

PL/I was chosen as the programming language in 1964. Other possibilities were a port of MAD (the Michigan Algorithm Display) or a port of AED-0 (an MIT display). The full PL/I language was harder to implement than expected. A contract was awarded to an outside firm to produce a PL/I compiler, and BTL administered the contract. The contractor assigned two people and had produced no compiler a year later. Bob Morris and Doug McIlroy (at BTL) created a back-up plan for

a PL/I compiler, using McClure's TMG language on the 7094. This language was called EPL (Early PL/I). McIlroy recalls:

> We began building it on May 3, 1965, just as soon as it was clear that the contractor was getting nowhere. We had customers well before end of the year. It was in full use by the time Morris left on sabbatical for the '66-'67 academic year.
>
> The contractor we engaged to make a production compiler never produced anything worthwhile, so EPL remained in use until around 1970 when a skilled team at GE made a real compiler.
>
> Morris wrote the front end in TMG; it put out 1-address code in ASCII. Each address was annotated with <base, scale, mode, precision>, which made quite a mouthful. I did the code generator, also in TMG; it put out assembly language. Dolores Leagus joined in to compile structure declarations. We excluded I/O, which accounts for about half the syntax of PL/I, scaled fixed point (divide yielded a float), the "defined" attribute, and array cross-sections, but handled a remarkably big subset, including the nasty "refer" attribute, which IBM left out of their early releases. ("Refer" allowed the dimensions of an array contained in a structure to be given by earlier elements in the structure.)
>
> Jim Gimpel joined in to work on packed data, which Multics had aplenty, and we retrofitted some 3-address code into the intermediate language to help eliminate useless temporaries. The compiler lacked certain production niceties. There were only two error messages, "syntax error" and "redeclaration," and one warning, "idiotic structure," when a short datum spanned a word boundary.
>
> TMG had been ported from the CDC 1600 by McClure transliterating the assembly language to 7090 coding forms and me debugging it 1,000 miles away. Clem Pease moved it next to the GE 635 by defining IBM instructions as 635 macros and assembling afresh.

The compiler produced output in EPLBSA (EPL Bootstrap Assembler). Compilation was very slow.

While waiting for the PL/I compiler to become available, the team wrote the *Multics System Programmer's Manual.* It ran about 3,000 pages; every section went through serious review and many sections were rewritten or deeply revised several times.

Tom Van Vleck says:

> In 1968-69 the system was late and under significant financial pressure and threat of cancellation. Maybe this helped esprit de corps (as opposed to surface morale). A review by a select ARPA [*the Advanced Research Projects Agency of the Department of Defense*] committee in 1968 was one time we came close to cancellation; they recommended that we continue. But it was too late. [Because of tardiness,] Bell Labs withdrew in April 1969.

While there were still Multics machines running in 1993, they were a rare species. Van Vleck supplied me with the following list. The two sites marked with asterisks were unconfirmed.

ACTC (Calgary, Alberta, Canada)

Air Force Data Services Center (Washington, DC) *

Bull System-M (Phoenix, AZ)

Crédit Lyonnais (Paris, France)

Department of National Defence (Halifax, NS, Canada)

Ford Motor Company (Dearborn, MI)

General Motors (Warren, MI)

Ministerie van Sociale Zaken en Werkgelegenheid (Den Haag, The Netherlands) *

National Security Agency [DOCKMASTER] (Linthicum, MD)

Société Européenne de Propulsion [SEP] (Vernon, France)

Multics' importance was in its lasting effect. Honeywell purchased GE's computer business, but its GCOS 6 operating system, Prime's Primos, Stratus' VOS, and Apollo's Domain were strongly influenced by the Multics system, as were TENEX, created at BBN, and TOPS-20. (Bill Poduska, Prime's founder, had worked on Multics at GE,

and then gone to DEC; Bob Freiburghouse, one-time Multics languages manager, was a founder of Stratus.) The memory management in the Intel 386/486 looks as though it came out of the Multics manual. But the most important effect may well have been on Dennis Ritchie and Ken Thompson of Bell Labs, and their colleagues—McIlroy, Morris, Neumann, and Ossanna—who had been exposed to CTSS and to Multics. Ritchie told me:

> There were lots of fundamental things we learned from Multics:
>
> - a tree-structured file system
> - a separate, identifiable program to do command interpretation; even the name for this program, the "shell," was taken from Multics
> - more fundamentally, the structure of files, that is, no structure, except as byte arrays, in most cases not interpreted by the operating system
> - text files are just sequences of characters separated by newline characters
> - the semantics of I/O operations (read and write) as referring to a file handle, a buffer, and a count—concealing the underlying disk blocks.

Birth of a System

CHAPTER

The PDP-11

Dennis Ritchie, Ken Thompson and Joe Ossanna had tried several times to convince BTL to purchase a computer for the research group. But most computers in 1969 and 1970 meant an expenditure of well over $100,000. And despite the work on file systems and tools, the computing research group still had no computer of its own. They tried to get BTL to purchase a PDP-10, or to part lease/part purchase a machine, but they were totally unsuccessful. Dennis Ritchie told Peter Collinson:

> On the PDP-7 Unix system everything was written in assembly language. Doug McIlroy did an implementation of TMG, which is a compiler writing system done originally by Bob McClure. It's a top down parser ... Since we only had 4K words of memory for user programs on the PDP-7, PL/I was clearly out of reach. So Doug did TMG for the PDP-7. That was the year that Ken Thompson decided that we couldn't have a serious system without a FORTRAN compiler, so we sat down to write a FORTRAN compiler using TMG. This lasted about a day. Instead what happened was B. B was essentially a cut-down version of BCPL. ...

BCPL (Basic Combined Programming Language) was Martin Richards' "tool for compiler writing and systems programming" in

1967. As B was a "cut-down version of BCPL," its name was an abbreviation, too.

Ritchie explained:

> At the time I came into the Labs [in 1968], almost the very first thing that I did was to get hooked up with Rudd Canaday who was the person who had ported the BCPL compiler to the GECOS machine, the General Electric [later Honeywell] GE-635. At the time he was moving the compiler to Multics. So I got access into Multics using a BCPL compiler. This was an early dialect of BCPL, Martin Richards himself continued to change it after we grabbed it.
>
> So BCPL was around, and we had written some noticeable programs in it. On the first Unix system, B was a new, simpler language based on BCPL. It was an interpreter and didn't produce machine code; it produced an intermediate code.
>
> It was first written in TMG, and later it was bootstrapped. In fact, the development of the language was amusing. The compiler was always pushing against the size limit in the machine. Ken would add something to the compiler and there would be a painful period when it was very hard to add another feature. But then he would include the new construct in the compiler—and that would make it smaller again, allowing more new things to be added.
>
> B was really the first high-level language that was used on Unix. A few things were written in it. There were two implementations. One was this ordinary interpreter generating some intermediate stack-based language. There was also a thing called **vb**, a Virtual B, that was a software paged version of the same thing allowing 4K word segments to be paged in and out by the interpreter. Whenever we had a program that got too big to fit into memory, **vb** was used.

Andrew Hume told me that "B was still being used up to 1989; the program that drove our typesetter was a B program." But the hardware limitations were really frustrating. Finally, Ossanna suggested the purchase of a PDP-11/20 for a text preparation project. Lee McMahon,

Ritchie said, "thought what we had done was good." As a result of his efforts (and confidence in the word-processing system), Max Mathews, the director of acoustics research, chipped in seed money. Doug McIlroy told me that "Without that helping hand from outside computer science, Unix might never have gotten beyond the fetal stage." The administration at the Lab understood that text processing was something useful. The PDP-11 was ordered. Ritchie recounts:

> We got the PDP-11 very early. It came during the summer of 1970. Only the processor and memory arrived, there was no disc. It was all paper tape software, you loaded things with paper tape, there was no operating system as such. The first Unix for it was written cross-assembling from the PDP-7 using the PDP-7 assembler that was written in B.
>
> By the time we got the PDP-11, only a few things had been re-written in B; none of the basic commands just a few extra ones. One of the first programs was the PDP-11 assembler. There was also a version of **dc**, desk calculator, a very very early program. That was actually the first program that ran on the PDP-11. It ran standalone before there was an operating system. [The 'Standard DEC OS' was never installed on the PDP-11.] It was a very raw assembler. It was really syntactically almost the same as the original instruction set on the PDP-11.
>
> Unix came up in two stages. Ken got it going before there was a disk, he divided the memory up into two chunks and got the operating system going in one piece and used the other piece for a sort of RAM disc. To try it out, you'd first load this paper tape that initialized the disk and then load the operating system. So there was a **cp** [*copy file*], a **cat** [*catenate files*], and an **ls** [*list files*] actually running before there was a disc.
>
> Once the system was really there, there was a regression. The B version of the assembler was pretty slow and that was re-written in assembly language. I guess there were a few new things that were written in B. One of the earlier ones was the thing that did the expansion of stars and whatnot in file-

> names: the **glob** command. This stood for **global** for reasons that escape me; it's not very sensible. Expansion was done by a separate program, when the shell recognized it had a magic character in an argument, it would call this command to do the expansion. The code wasn't built into the shell. That program was written in B.

Thompson says that while waiting for the disk for the PDP-11, they had put the PDP-7 by its side. They then ported the code by tape and ran the file system in memory.

But once the PDP-11 was up, an editor was needed in order to program. The PDP-1 at MIT had had an editor called TECO. Originally, this was an acronym for [paper] Tape Editor and COrrector, later for Text Editor and COrrector. TECO is a linear ancestor of EMACS (Editing MACroS). In 1967, L.P. Deutsch and B.W. Lampson implemented TECO on the SDS-940 as QED (Quick EDitor). The SDS later became a Xerox. Thompson had written a version of QED for CTSS on the IBM 7094 at MIT. He and Ritchie then wrote a version for the GE-635 at BTL. Now Thompson wrote the simplified line editor **ed** for the PDP-11. But the PDP-11 had been acquired for text processing, so a program that would render text was necessary.

J.E. Saltzer had written **runoff** for CTSS. Doug McIlroy recalls:

> I believe it was Morris and maybe Thompson who somehow moved runoff to the 635 and called it roff. It was a quick hack, done literally overnight. I then [1969] wrote a roff from scratch in BCPL, both simplifying and extending it beyond runoff. That became the model for the roff that Thompson and/or Ritchie brought up (in machine language) on Unix.

McIlroy later wrote me: "Ken does not remember the first 635 roff, so I guess he had nothing to do with it. He does remember, though, another one that I don't—a minimal program called **rf** that he wrote for the PDP-7, probably before Unix itself. Apparently it was an evolutionary dead end." Ritchie then "guessed" that he had done it.

Ritchie said, "we knew there was a scam going on—we'd promised a word processing system, not an operating system." But the text processing effort was a success: the Patent Department of BTL

became the first Unix user, sharing the PDP-11/20 with the research group. Even more important: Bell's Patent Department subsequently took over the 11/20 running Unix and turned funds over to Computing Research, with which a PDP-11/45 was purchased.

CHAPTER

6

First Edition, 1971

With the advent of the PDP-11/45, the system began to grow. It put on weight in terms of instructions, subroutines, and games. But the only way you could learn it was to sit down with one of the authors and ask questions.

Joe Condon, whose group "owned" the original PDP-7, later moved to Computing Research. He recalled for Mike Mahoney just how Bob Morris initiated him to the Unix way of thinking.

> I would come around and say, "How do you understand what these commands do?", because the manuals are, the manual pages aren't all that clear. And he [Morris] says, "What do you think is the reasonable thing to do? Try some experiments with it and find out, Joe." And that was a very interesting clue to at least his philosophy and some of the other people's philosophy (I think Dennis' also) of how a system command or how a thing should work—it should work in a way which is easy to understand; it shouldn't be a complex function, which is all hidden in a bunch of rules and verbiage and what not, that there's a field of cognitive engineering.
>
> I think that what Bob Morris was telling me is that the black box itself should be simple enough such that when you form the model of what's going on in the black box, that's in fact what is going on in the black box. You shouldn't write a program to try and outwit the person and to try to double

> guess what they're going to want to do. You should make it such that it's clear what it does.

According to Ritchie, it was McIlroy who bent arms to get a programmer's manual. McIlroy says it was Ritchie who designed and wrote the first man page. It appears to have been *cat*—catenate. Ritchie told me he doesn't recall who thought up the format. However, the design-on-the-fly was brilliant: we still use it. Doug McIlroy saw the programmer's manual as a way of maintaining integrity and coherence. What Ritchie had done was to set the style for that integrity. Many today find the man pages too terse and uninformative, but whatever the complaints, they are infinitely better than the information available with other operating systems.

Sandy (A.G.) Fraser, later head of the Computer Systems Research Department at BTL, talked to Mahoney about McIlroy and the manual:

> Now you may think of that as a clerical job, but don't think of it that way. The fact that there was a manual, that he insisted on a high standard in the manual, meant that he insisted on a high standard of every one of the programs that was documented. Then they say, as they have just done, to produce the next edition of the manual that the work that went into producing the manual involved rewriting all sorts of programs in order that they should meet the same high standard. And then add to all of that, it's probably the first manual that ever had a section with bugs in it. That's a level of honesty you don't find. It wasn't that they simply documented the bugs that they were too lazy to fix; they fixed a lot of bugs. But some of them weren't so easy to fix, or it was uncertain what one ought to do. So they documented that. I think a level of intellectual honesty was present in that activity that is rare.

Doug McIlroy spoke about the manual, too:

> Cleaning something up so you can talk about it is really quite typical of Unix. Every time another edition of the manual would be made, there would be a flurry of activity and, when you wrote down the uglies, you'd say, "We can't put this in

> print," and you'd take features out and put features in in order to make them easier to talk about. It's the virtue of being in a research center. You don't have to keep any old software running.

In fact, when I looked closely at the first four manuals (November 1971 through November 1973), I got the strong feeling that each had been changed a great deal prior to being typeset: the systems were the essence of flux, and the documentation reflects that. The second paragraph of the First Edition manual notes: "The rate of change of the system is so great that a dismayingly large number of early sections had to be modified while the rest were being written." But the flux was present because the system was a living, growing entity. Reading the brief remarks that Thompson wrote at the beginning of each edition illuminates this.

The first edition of the "*UNIX PROGRAMMER'S MANUAL* [by] K. Thompson [and] D. M. Ritchie" is dated "November 3, 1971." It begins with a two-and-a-half page "INTRODUCTION," followed by a "TABLE OF CONTENTS" that is just over three pages long. The table of contents is divided into sections. These sections, containing *every* **command, system call, subroutine, special file, file format, user maintained programs,** and "miscellaneous," have had a profound influence on the field. Fifteen years later, in 1986, the seven volumes of manuals for Berkeley's version of Unix (4.3BSD) were arranged under the same subheadings (though, over the years, there have been additions to the numbering beyond these seven topics and six has become "Games" overtly—originally it contained four games anyway—**bj** blackjack, **chess**, **moo** a guessing game, and **ttt** tic-tac-toe—as well as programs like **sort** (sort a file into alphabetical order) and **cal** (print the calendar). **chess** was particularly important, for Ken Thompson was a serious chess player and the creator of Belle, which a decade later became the World Computer Chess champion. (When taking Belle to the Netherlands, Thompson was asked whether the machine was "a customs threat." "Only when dropped from the airplane," he is said to have responded.)

One of the text processing features already present in 1973 was the ability to construct a "permuted" index, one in which every item

in a series occurred. Because there were similar command names in various sections of the manual, the notation **command(section)** was introduced to help avoid confusion. Thus, **cp(1)** meant that the copy command was in Section 1, **dc(1)** referred to desk calculator in Section 1, and **dc(4)** referred to a special file in Section 4 that worked a remote typewriter (this command disappeared after 6th Edition). **cat(1)** referred to "catenate files," but **cat(4)** to "C/A/T phototypesetter interface."

The introduction is a fascinating document in many ways, not least because it gives the reader/user a key to just who is responsible for what. It thus contains the list:

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

as the "owners" of the various programs. The tradition of giving credit where credit is due in the Unix community begins here. It continues in many parts of that community, though large corporate users have eliminated the names, initials or logins from their documentation (e.g., DEC, HP, IBM, Sun). Lorinda Cherry remarked that the principle involved was a simple one: "He who touched it last, owned it." And the owners were listed by their logins, because they already had email and interactive programs like **write**.

What was in First Edition? I think the easiest way to put it is: lots. In terms of structures of the system, only **pipes**, which we'll look at later, weren't there. There were only 60 user commands in Section 1.

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
dbppt	write binary paper tape
dc	desk calculator
df	find free disk space
dsw	delete files interactively
dtf	format DECtape
du	find disk usage
ed	text editor
find	find file with given name
for	compile FORTRAN program
form	generate form letter
hup	hang up typewriter
lbppt	read binary paper tape
ld	link editor [loader]
ln	link to file
ls	list directory contents
mail	send mail
mesg	permit or deny messages
mkdir	create directory
mkfs	create file system
mount	mount detachable filesystem
mv	move or rename file
nm	print name list
od	octal dump of file
pr	print file with headings
rew	rewind DECtape
rkd	dump disk to tape
rkf	format RK disk
rkl	load disk from tape
rm	remove [delete] file
rmdir	remove [delete] directory
roff	run off text
sdate	adjust date and time
sh	command interpreter
stat	get file status
strip	remove symbols
su	become superuser
sum	sum file
tap	manipulate DECtape
tm	get time information
tty	find name of terminal
type	print file on IBM 2741
un	find undefined symbols
wc	get word count
who	who is on the system
write	write to another user

Of those 60 commands, many are still familiar to us, though few paper tape readers, DEC RK05s, or IBM 2741s—if any—are still around. Other commands have lost their utility or evolved into something else: **dsw** (delete using switches) was absorbed into **rm** (remove) and **db** became **adb** in Seventh Edition; **for** became **fc** in Second Edition, which became **f77** in Seventh Edition; **lbppt** became **restor** in Fifth

Edition; **dbppt** became **dump**. Many commands vanished away: **dtf** vanished after First Edition, as did **hup, rkd, rkf, rkl,** and **sdate.**

The (in)famous **creat** "create a new file" was in Section 2, *System Calls*. It raises a smile among old-timers who recall that Thompson said that his biggest mistake was leaving the **-e** off **creat.**

Doug McIlroy told me that Thompson had been wrong, the biggest mistake: "was spelling Unix with all caps. Unix, Fortran, and other good computer words are proper names. A page that treats them as such is much more inviting than a page marred by all-caps works." This remark has led me to use Unix rather than UNIX in most of this history. As I have said, the system was in constant flux. And so the next two years were very exciting ones.

BTL Unix Editions

NOTE: This list is of the ten AT&T Bell Laboratories editions of the *UNIX PROGRAMMER'S MANUAL*. The Tenth Edition was published commercially in 1990. The First through Sixth editions bear the names of Thompson and Ritchie on the title page; the Seventh Edition is headed, for the first time, *UNIX™ TIME-SHARING SYSTEM*, with no names, although there is a brief "PREFACE" by "B. W. Kernighan [and] M. D. McIlroy." The Eighth and Ninth Editions carry brief prefaces by McIlroy alone; they also carry the rubric *Research Version*. The preface to the Tenth Edition is signed "A.G. Hume, M.D. McIlroy, October, 1989."

First Edition	November 3, 1971
Second Edition	June 12, 1972
Third Edition	February 1973
Fourth Edition	November 1973
Fifth Edition	June 1974
Sixth Edition	May 1975
Seventh Edition	January 1979
Eighth Edition	February 1985
Ninth Edition	September 1986
Tenth Edition	October 1989

CHAPTER 7

C and pipes: 1971–1973

Inside certain parts of Bell Telephone Laboratories, Unix was a success. Within a few months, Thompson and Ritchie were at work on a new manual. The Second Edition appeared in mid-June 1972. The "PREFACE" reports:

> In the months since this manual first appeared, many changes have occurred both in the system itself and in the way it is used.
>
> . . .
>
> [T]he number of people spending an appreciable amount of time writing UNIX software has increased. Credit is due to L. L. Cherry, M. D. McIlroy, L. E. McMahon, R. Morris, and J. F. Ossanna for their contributions.
>
> Finally, the number of UNIX installations has grown to 10.

The first installations, of course, were those of Research and the Patent Department. The first user outside of central New Jersey was Neil Groundwater, then at New York Telephone (he is now with Sun Microsystems). He told me:

> I joined New York Telephone in February of 1972 after completing a B.S. in Computer Science at Penn State.
>
> The job market for computer grads was tough at the time but the reason I was hired was that I'd had some experience with a minicomputer at Penn State (an ADAGE graphics

system). In the spring of 1972 I began traveling to Bell Labs in Whippany, New Jersey. Since I had an apartment in Manhattan and no car it was convenient for me to take a bus to Whippany on Monday mornings, stay at a hotel during the week, and return to New York on Fridays. Being near the Labs for a week at a time got me immersed in the work pretty quickly. One of my first recollections about the job there is a description they shared about learning the Unix system: it's a lot like climbing a chimney from the inside, you have to work your way up all the sides a bit at a time and at the same time. Documentation did exist on many parts of the system but as we would come to say years later, "Use the Source, Luke."

The group I worked with at Whippany was "mechanizing" the analysis of some of the "outside plant" tasks involving ESS [Electronic Switching System] offices in New York City. The ESS machines generated call-failure messages (our first were a type called "TN08") on teletype printers in the central offices. Although the Unix host didn't communicate directly back to control the "switch," some artful wiring allowed a modem into the current-loop of the teletype printer and another modem at our office (on the 14th floor of 330 Madison Avenue in Manhattan) sent the signal into a multiplexer port.

In the summer of 1972 the hardware for the New York site arrived:

DEC PDP-11/20 processor

56 Kbytes of core memory

High-speed paper tape reader/punch

ASR-33 Teletype - console

DECtape - twin drive

RK11/RK05 disk (2) - 2.4 Mbytes

RF11 fixed head disk (2 at first, 3 more added later)

DC11 (6 lines) for local terminals

DM11 16-line multiplexers (3)

My first programming assignment was to rewrite a driver written for a single DM11 multiplexer to handle multiple multiplexers (in this case, three)....

The Unix "shell" program fit onto seven pages of line-printer paper. There were no library routines, just direct system calls for reading or writing. Many of the features one expects today were in the shell already: input and output redirection (but no pipes), meta-character expansion (although an external program **/etc/glob** was invoked to perform the expansion), and shell scripts are a few.

An incorrect program could easily overwrite part of the kernel. A **halt** instruction on the PDP-11 was a "zero" so you can imagine that clearing a location referenced by a register-pointer could stop the CPU pretty quickly. There was a way to dump core of a user program, either programmatically or by a keyboard sequence, this afforded post-mortem debugging.

Program development generally occurred out-of-hours or at Whippany. At Whippany, the terminals on the development machine were in a common room and when several people were at work, one would call out "dangerous program!" before executing a new **a.out** file (the default output file from the linking editor). This allowed others to save their editor-files quickly (and often).

A similar call cried out when one wanted to use the line-printer. There was no spooling or lockout. **pr myfile > /dev/lp** was how you sent your listing to the printer. If two users sent output to the printer at the same time, their outputs were interspersed. Whoever shouted, "line printer!" first owned the queue.

The Unix system communicated back out to the ESS offices by sending back out to the modems connected to the central offices; but at the offices, the receiving signal was not routed to the ESS' teletype, but to an Execuport hardcopy terminal (similar to a TI Silent 700 printing on heat-sensitive paper). Because the DM11 was capable of different incoming and outgoing speeds, the incoming (teletype) speed was 110

baud while the outgoing (Execuport) line was sending at 300 baud.

After collecting trouble reports, FORTRAN programs sorted and formed "exception reports" when multiple reports contained the identifiers of a particular piece of equipment. The operation of the telephone network is such that an individual piece of equipment may report a trouble condition external to that device; rather than the call not completing (i.e. connecting the two phones) the switching equipment retries the call and prints a trouble report on the console. Actually there were a variety of report types, but the TN08 was identified as the first they would try to troubleshoot with the new system.

Booting the PDP-11/20 consisted of loading a starting address (one address for a disk-boot and another for tape-boot) into the console switches and hitting the Execute switch. The boot ROM was literally a board full of diodes that came from DEC with all the bits (one bit, one diode) full and they were snipped out to create the instructions for booting.

In 1973, we added a second, similarly configured PDP-11/20 at the same site. It had an optical card reader that read "mark sense" cards that had been stroked with a pencil by panel-office (an older type of switching equipment) technicians when a "sender" got stuck. "Panel stuck sender" was their term for the condition and, like the TN08 analysis of an ESS, if you collect enough reports, the computer can identify repeating items and help locate trouble conditions.

The Third Edition appeared eight months later in February 1973. E.N. Pinson's name was added to the list of contributors. And, most importantly,

> Finally, the number of Unix installations has grown to 16, with more expected.

The front matter of Third Edition ran to nearly a dozen pages before the "Table of Contents." In addition to sections on getting started,

"How to communicate through your terminal," "The shell," and path names, there were a few paragraphs on "Writing a program" and "Text processing." The first of these began:

> To enter the text of a source program into a Unix file, use ed(1). The three principal languages in Unix are assembly language (see as(1)), FORTRAN (see fc(1)), and C (see cc(1))....

C? What was C? The manual page for cc (dated 3/15/72) told you that it was a C compiler and referred you to the 'C reference manual.' It would be five years before Brian Kernighan and Dennis Ritchie published *The C Programming Language,* though the manual was incorporated into the documentation earlier. But the rewriting of Unix in this new language (which was effected in Version 4) was tremendously important. And the language itself became important to all of computing.

Mike Mahoney asked Dennis Ritchie about designing C:

> It was an adaptation of B that was pretty much Ken's. B actually started out as system FORTRAN.... Anyway, it took him about a day to realize that he didn't want to do a FORTRAN compiler at all. So he did this very simple language called B and got it going on the PDP-7. B was actually moved to the PDP-11. A few system programs were written in it, not the operating system itself, but the utilities. It was fairly slow, because it was an interpreter. And there were sort of two realizations about the problems of B. One was that, because the implementation was interpreted it was always going to be slow. And the second was that, unlike all the machines we had used before, which were word-oriented, we now had a machine that was byte-oriented and that the basic notions that were built into B, which was in turn based on BCPL, were just not really right for this byte-oriented machine. In particular, B and BCPL had notions of pointers, which were names of storage cells.... There were all these different sizes of objects, and B and BCPL were really only oriented toward a single size of object. From a linguistic point of view that was the biggest limitation of B; not only the fact that all objects were the

same size but also that just the whole notion of pointer to object didn't fit well.... So, more or less simultaneously, I started trying to add types to the language B, and fairly soon afterwards tried to write a compiler for it. Language changes came first. For a while it was called NB for New B; it was also an interpreter, and I actually started with the B compiler... because C was written in a language very much like itself, at every stage of the game... and sort of merged it into the C compiler and added the various, the type structure. And then tried to convert that into a compiler.

The basic construction of the compiler—of the code generator for the compiler—was based on an idea that I'd heard about; someone at the [Bell] Labs at Indian Hill. I never actually did find and read the thesis, but I had the ideas in it explained to me, and some of the code generator for NB, which became C, was based on this Ph.D. thesis. It was also the technique used in the language called EPL, which was used for switching systems and ESS machines; it stood for ESS Programming Language. So that the first phase of C was really these two phases in short succession of, first, some language changes from B, really, adding the type structure without too much change in the syntax and doing the compiler.

[The] second phase was slower. It all took place within a very few years, but it was a bit slower, or so it seemed. It stemmed from the first attempt to rewrite Unix in C. Ken started trying it in the summer of probably 1972 and gave up. And it may be because he got tired of it, or whatnot. But there were sort of two things that went wrong. And one was his problem, in that he couldn't figure out how to run the basic coroutine, multiprogramming primitives—how to switch control from one process to another, the relationship inside the kernel of different processes. The second thing that he couldn't easily handle was, from my point of view, the more important, and that was the difficulty of getting the proper data structure. The original version of C did not have structures. So to make tables of objects—process tables and file tables, and that tables and this tables—was really fairly

> painful.... It was clumsy; I guess people still do the same sort of thing in FORTRAN.
>
> The combination of the things caused Ken to give up that summer. Over the year, I added structures and probably made the compiler somewhat better—better code—and so over the next summer, we made the concerted effort and actually did redo the whole operating system in C.

The other innovation present in Third Edition was the **pipe.** Pipes (and filters) are very simple concepts: they are a uniform mechanism for connecting the output of one program to the input of another. The Dartmouth Time-sharing System had communication files, which anticipate pipes, but in a far more specific (less general) way. The notion was Doug McIlroy's. The implementation was Thompson's at McIlroy's insistence ("It was one of the only places where I very nearly exerted managerial control over Unix," he said). McIlroy told Mike Mahoney:

> In the early '60s Conway wrote an article about coroutines—'63 perhaps, in the *CACM* [*Communications of the ACM*]. I had been doing macros, starting back in '59, '60. [A macro is an instruction that refers to set of instructions; basically, it is a one-to-many mapping.] And if you think about macros, they mainly involve switching data streams. I mean, you're taking input and you suddenly come to a macro call, and that says, "Stop taking input from here. Go take it from the definition," and in the middle of the definition you'll find another macro call. So macros even as early as '64 —somewhere I talked of a macro processor as a "switchyard for data streams." Also in '64, there's a paper that's hanging on Brian's wall still, [which] he dredged out somewhere, where I talked about screwing together streams like garden hoses. So this idea had been banging around in my head for a long time.
>
> At the same time that Thompson and Ritchie were on their blackboard, sketching out a file system, I was sketching out how to do data processing on this blackboard by connecting together cascades of processes and looking for a kind of prefix notation language for connecting processes together, and failing because it's very easy to say "*cat* into *grep* into ...,"

or "*who* into *cat* into *grep*," and so on; it's very easy to say that, and it was clear from the start that that was something you'd like to say. But there are all these side parameters that these commands have; they don't just have input and output arguments, but they have the options, and syntactically it was not clear how to stick the options into this chain of things written in prefix notation, *cat* of *grep* of *who* [i.e., *cat*(*grep*(*who* ...))]. Syntactic blinders: didn't see how to do it. So I had these very pretty programs written on the blackboard in a language that wasn't strong enough to cope with reality. So we didn't actually do it.

And over a period from 1970 to 1972, I'd from time to time say, "How about making something like this?" and I'd put up another proposal, another proposal, another proposal. And one day I came up with a syntax for the shell that went along with the piping, and Ken said, "I'm going to do it!" He was tired of hearing all this stuff, and that was—you've read about it several times, I'm sure—that was absolutely a fabulous day the next day. He said, "I'm going to do it." He didn't do exactly what I had proposed for the pipe system call; he invented a slightly better one that finally got changed once more to what we have today. He did use my clumsy syntax.

He put pipes into Unix, he put this notation [*Here McIlroy pointed to the board, where he had written: f > g > c*] into shell, all in one night.... Most of the programs up to that time couldn't take standard input, because there wasn't the real need. So they all had file arguments; *grep* had a file argument, and *cat* had a file argument, and Thompson saw that that wasn't going to fit with this scheme of things and he went in and changed all those programs in the same night. I don't know how.... And the next morning we had this orgy of one-liners.

Dick Haight, later to become manager of PWB, told August Mohr:

I happened to have been visiting the research crew the day they implemented pipes. It was clear to everyone, practically minutes after the system came up with pipes working, that it was a wonderful thing. Nobody would ever go back and give that up if they could.

Prior to that night, Unix had no "toolbox" concept. The invention of pipes laid the groundwork for a fresh way of thinking about software that was explored over years to come. That way of thinking would lead to a unique philosophy. Where the toolbox was concerned, McIlroy states: **Pipes created it.**

Mahoney asked him: "Unix looked different after pipes?" And McIlroy said:

> Yes, the philosophy that everybody started putting forth, "This is the Unix philosophy. Write programs that do one thing and do it well. Write programs to work together. Write programs that handle text streams, because that is a universal interface." All of those ideas, which add up to the tool approach, might have been there in some unformed way prior to pipes, but they really they came in afterwards.

Tools and toolboxes were something Brian Kernighan was to get involved with. But initially he merely made one change to Thompson's implementation of pipes. He substituted ^ for >. He told Mahoney pipes were

> the thing that makes it all work, in some sense. It's not that you couldn't do those kinds of things, because I/O redirection predates pipes by a noticeable amount—not a tremendous amount, but it definitely predates it; I mean, that's an oldish idea. And that's enough to do most of the things that you currently do with pipes; it's just not notationally anywhere near so convenient. I mean, it's sort of loosely analogous to working with Roman numerals instead of Arabic numerals. I mean, it's not that you can't do arithmetic; it's just a bitch. Much more difficult, perhaps, and therefore mentally not more constraining. But all of that stuff is now squashed into such a narrow interval that I don't even know when it happened. I remember that the preposterous syntax, the > >, or whatever syntax that somebody came up with, and then all of a sudden there was the vertical bar and just everything clicked at that point. And that was the time then I could start to make up these really neat examples that would show things like

> doing—you know, running *who* and collecting the output on a file and then word-counting the file to say how many users there were and then saying, "Look how much easier it is with the *who* into the wordcount," and then running *who* into *grep,* and starting to show combinations that were things that were never thought of and yet that were so easy you could just compose them at the keyboard and get 'em right every time. And that's I think when we started to think consciously about tools, because then you could compose the things together, if you had made them so that they actually worked together.

There was only one more step, Doug McIlroy told me that Thompson had made the substitution of "|" as the pipe symbol "for a talk in London, because he couldn't bear to reveal my ugly syntax."

The power of Unix originated here, from the relationships generated among programs, not from individual programs themselves.

Unix now had a language all its own. It had a philosophy, an ethos. It had a relatively small group of devoted users on a handful of AT&T Bell sites. But it had no real audience.

The Unix Philosophy

- Write programs that do one thing and do it well.
- Write programs to work together.
- Write programs that handle text streams, because that is a universal interface.

CHAPTER

8

The First Paper, 1973

In February 1973 the Third Edition of the *UNIX PROGRAMMER'S MANUAL* appeared. As has been noted, the code was now largely rewritten in C. The C compiler and C debugger were in place. The system had been installed on 16 sites (all within AT&T/Western Electric). Thompson and Ritchie (with the "constant support" of Bob Morris, Doug McIlroy, and Joe Ossanna) submitted an abstract for a presentation to the ACM Symposium on Operating Systems Principles, to be held that October at the IBM Thomas J. Watson Research Center in Yorktown Heights, NY.

SOSP, as it has come to be known, was not a very large meeting. Doug McIlroy remarked to me that he did not open the morning session "to a well-filled auditorium."

Ritchie remarked that, "It was a beautiful fall day to drive up the Taconic." Thompson, who gave the paper, told me:

> The audience was several hundred. I was pretty nervous. The response was the normal, polite applause. I don't remember questions. As I recall I had seen the TENEX talk earlier [TENEX was an OS popular with users of the DEC-10] and adjusted my talk to show up some contrasts. I spent a lot of time on the shell and its implementation—trying to demonstrate how it was "just a program." There was a question about the shell that completely missed the point. Perhaps I didn't do a good job.

Thompson is too modest. The response to the paper was tremendous. Within six months of its delivery, the number of installations had trebled. When the revised version of the paper was published in the July 1974, *Communications of the ACM*, the response was amazing. No one remembers that TENEX paper (it wasn't among the half-dozen in the July *CACM*), and the *CACM* editorial referred to Thompson and Ritchie's work as "elegant." In 1983 they were to share the ACM's prestigious Turing Award for their work.

CHAPTER

9

The Law—Part I

On January 14, 1949, the Truman administration, through the Antitrust Division of the Department of Justice of the United States of America filed a complaint against the Western Electric Company, Inc., and the American Telephone and Telegraph Company in the District Court for the District of New Jersey, claiming that the companies were acting in restraint of trade in violation of the Sherman Antitrust Act. After extensive negotiations, and in the very different political environment of the first Eisenhower administration, Judge Thomas F. Meaney entered a "consent decree" on January 24, 1956, "without trial or adjudication of any issues of fact or law herein and without this Final Judgment constituting any evidence or admission by any party..."

It has been suggested to me that the suit and the decree aren't well understood unless the changing government attitudes toward business behavior and the "phone company" are considered. While this may be true, what was important to not-yet-nascent Unix was what the decree actually said and how the AT&T and Western Electric lawyers interpreted it.

AT&T and Western Electric were enjoined "from commencing ... manufacture for sale or lease any equipment [other than that used in providing telephone or telegraph services]," with a few exceptions; "from engaging ... in any business not of a character or type engaged in by Western or its subsidiaries ..."; AT&T was enjoined "from engaging in any business other than the furnishing of common carrier com-

munications services...," with a few exceptions. Exception (b) was, "experiments for the purpose of testing or developing new common carrier communications services."

Most of the decree concerns manufacture of equipment, purchase and resale, accounting methods, and the licensing of patents. As might be expected in early 1956, there is no mention of computers or software. Licensing—which was mentioned, but not featured—was to prove to be of far greater importance than had been foreseen.

It might be appropriate to mention here that Western Electric was a wholly owned subsidiary of AT&T and that Bell Telephone Laboratories was a jointly owned subsidiary of both Western Electric and AT&T (50% each). However, it is important that the reader bear the corporate interpretation of the decree in mind when considering the fate of Unix over the years to come.

AT&T and Western Electric had been government-sanctioned monopolies for a long time. The very telephone equipment in our homes was until less than two decades ago merely leased. It was owned by The Phone Company. Part of receiving telephone service was an agreement not to modify, remove, or tamper with the equipment. The companies and the FCC had established a very wide interpretation of just what equipment was —it included the telephone directories—and what entailed modification (fitting a Bakelite voicecone was altering equipment), although it must be admitted that the courts reversed the FCC on occasion.

By requiring AT&T to reveal what patents it held and supply information about them to (potential) competitors, the judgment was a great step forward in one respect. (The Patent Department that was the first Unix "customer" had been enlarged in 1956 to satisfy this clause.) But the key was the requirement to license. The most important thing about patents is that after they are granted they are available to the world. But, without some kind of intervention, even though the whole world can read the patents, they can't "practice the inventions" recited by the patents (in other words, no one else can make, use, or sell) during the 17-year life of the patent without a license from the inventor. In this case, the terms of the decree required AT&T/Western Electric/Bell Laboratories to license to anyone at nominal fees. Thus, the decree resulted in a much more rapid dissemina-

tion of technology than would otherwise have been possible under our patent law.

The question of "engaging in any business other than the furnishing of common carrier communications services," was an interesting one, too.

Judge Meaney, the defendants and the plaintiff all thought that *common carrier services* was a known term. But common carrier has its own evolving legal definitions, starting with the words of the 1934 Communications Act which states, "a common carrier is any person engaged as a common carrier for hire, in interstate or foreign communication by wire or radio" (47 U.S.C. 153(h)). By design it seems, this isn't much help, and various interpretations by the FCC and the courts have added to the confusion.

Whatever the reason, the lawyers at BTL were conservative: there was no sense in aggravating the beast that was the Justice Department. No business but phones and telegrams.

Clearly, the collaboration with GE and MIT was a "research experiment," something Judge Meaney's decree permitted, and the internal efforts of the computing research group were experimental as well. The use of Unix by the Patent Department, NY Telephone, and far-flung offices of Western Electric, AT&T and the various Bell Operating Companies were legitimate, too.

But Ken Thompson's presentation at the Fourth ACM Symposium on Operating Systems Principles (October 15-17, 1973) resulted in a trickle of requests from sites that had nothing at all to do with BTL, AT&T or Western Electric. The publication of the revised version of the paper in *CACM* in July 1974 started a flood.

Sandy Fraser was one of those who felt the legal environment had a lot to do with the development of Unix.

> The company took a very self-constricting view of the decree, so Unix as an operating system held little appeal as outcome. The company worried about the legitimacy of several things much less obviously computing than Unix. ...
>
> No great strategy [was] associated with it [the way Unix went to the universities]. [It was] more a way of responding to what seemed irrelevancies, I would say, or something we couldn't make money from.

In other words, the further that Bell Labs' activities took them from common carrier activities, the more in danger it became of being viewed as violating the decree. Sam Morgan (who became Director of Computing Research in 1967) added: "AT&T management didn't understand what we had in Unix." In fact, Berkley Tague told Mohr (in 1985): "If you'd asked me at the time if releasing Unix to the universities was a good idea, I would have said no." And Tague (from 1973 on) was manager of the BTL department responsible for central support and development of Unix!

Furthermore, rather than irritating the great judicial dragon, the AT&T Legal Department decided that the order requiring the licensing of patents, meant that they had to license Unix, too. BTL had a long history of good relationships with academia as a result of their summer internship and sabbatical programs. Other Western Electric and AT&T software had been licensed to universities and so the company took the most prudent approach. As Otis Wilson, who was later to become manager of AT&T's software sales, put it:

> To preclude any conflict with the Consent Decree, AT&T would license its software under the Consent Decree's legally established procedures but would make it clear that it had no intention of pursuing software as a business. The policy was restated over and over again at every gathering of the faithful—"As is, no support, payment in advance!"

Andy Tannenbaum, in his talk at the Winter 1984 USENIX/UniForum meeting—otherwise known as the "first Washington snowstorm"—put it slightly differently:

> There was always a foil which described Bell System Unix support policy:
>
> | |
> |---|
> | no advertising |
> | no support |
> | no bug fixes |
> | payment in advance |

This slide was always greeted by wild applause and laughter.

From the winter of 1973 through the autumn of 1974, there were an increasing number of requests for Unix software, directed to the creators, and, through the patent attorneys, software was conveyed royalty-free under simple letter agreements. (The original agreement between Western Electric and The Regents of the University of California for Version 5 runs to six pages, two of which are the title page (unnumbered) and the signature page. Page 5 is the "Definitions Appendix." It is written in language any high school graduate would understand. It was "Effective as of December 1, 1973.") Later licenses got longer and more intricate. I have been told that there were licenses in the 1980s that precluded the dissemination of the terms of the license.

As Otis Wilson pointed out, Patent Licensing in 1970 "would typically be involved in only four or five negotiations per year." The office staff were, simply, overwhelmed by the number of requests for Unix licenses. It got worse after 1974, when first the military and then commercial enterprises asked for licenses.

System Status, 1974

It may be valuable here to look at just what Unix provided in 1974.

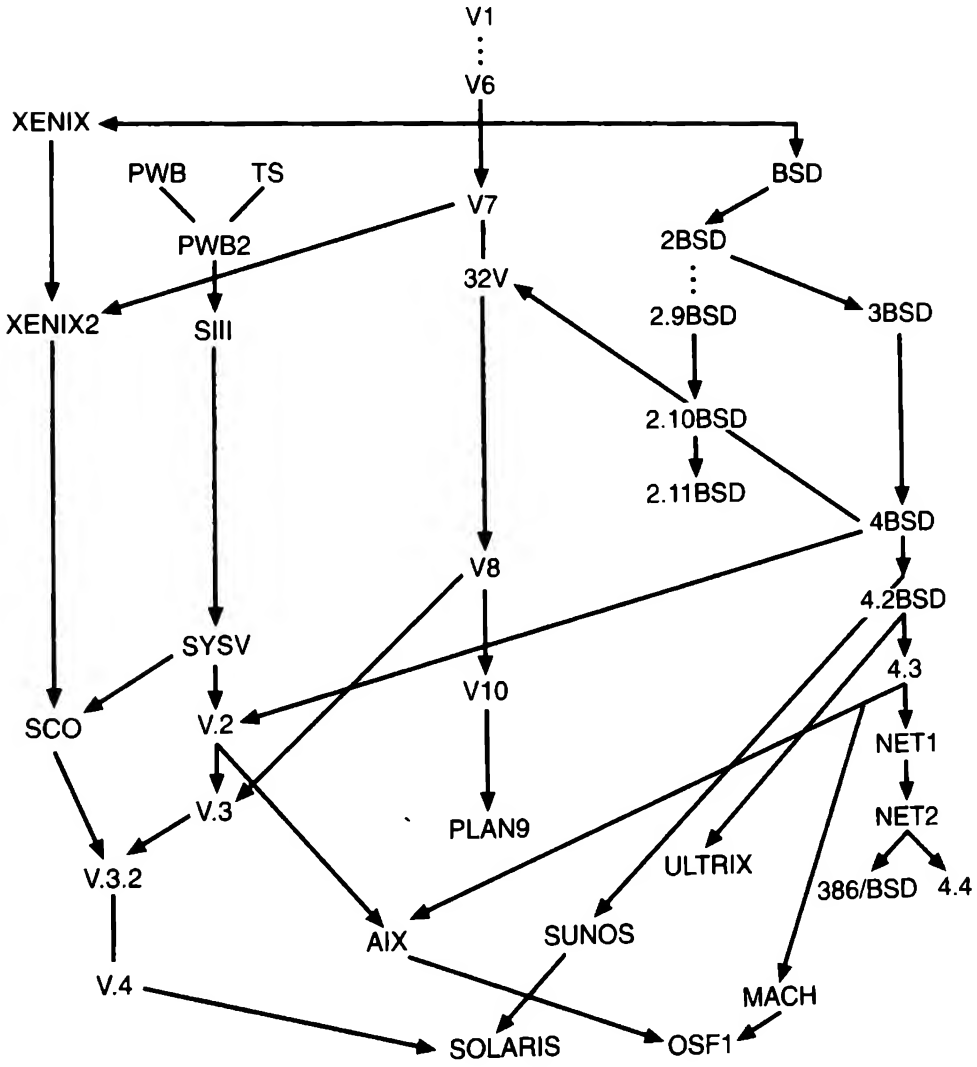
The typical programmer worked at a 30-character-per-second hardcopy terminal (the DECwriter II [LA36], which came out in 1975 could transmit at either 110 or 300 baud). While this would be considered excruciatingly slow today, using tools like *ed* or *sh* was wonderful when compared to submitting IBM JCL (Job Control Language) cards.

Insofar as text processing was concerned, most installations ran raw *nroff*, although some groups had phototypesetters and *troff*.

In 1974, many people were beginning to use C instead of assembler for their systems programming.

Unix also offered the opportunity to share files and supplied an environment in which maintenance and communication tools functioned. But the size of a project or group was limited by the hardware to the number who could fit on the PDP-11/45.

UNIX VERSIONS TREE



What Makes UNIX Unix?

CHAPTER 10

The Users

The decision on the part of the AT&T lawyers to allow educational institutions to receive Unix, but to deny support or bug fixes had an immediate effect: it forced the users to share with one another. They shared ideas, information, programs, bug fixes, and hardware fixes. Some of the potential users had already heard about Unix: Cyrus Levinthal, head of Biological Sciences at Columbia, was one ("but he didn't think it was available," Lou Katz told me) and Richard Langridge at Princeton was another.

But after the first paper was delivered in October 1973, you could have the bits put on your RK05 disks. In the early spring of 1974, Lou Katz (then at Columbia University) organized a meeting of Unix users. Columbia had been the recipient of the first distribution—first on disk, then on 9-track tape—in the autumn ("Cy got RK05s for the department," Katz told me, "but we didn't have a drive, so I drove down to Murray Hill and Ken cut me a 9-track tape.") The meeting was held on May 15, 1974, in the Merritt Conference Room on the third floor of Columbia's College of Physicians and Surgeons (P&S), to which Katz had moved from Biology. Here is the program, supplied to me by Reidar Bornholdt:

NN

UNIX USERS MEETING AGENDA
MAY 15, 1974
MERRITT CONFERENCE ROOM - 3RD FLOOR P&S

10:30	START
11:30	BRIEF DESCRIPTION OF THE SEVERAL INSTALLATIONS AND THEIR USE OF UNIX
12:00	LUNCH
1:00	KEN THOMPSON SPEAKS
2:00	INTERCHANGE OF UNIX HINTS, PROBLEMS SOLUTIONS, BUGS
3:00	INTERCHANGE OF DEC HINTS, PROBLEMS SOLUTIONS, BUGS
4:00	FREE-FOR-ALL DISCUSSION ?

About two dozen people from nearly a dozen institutions showed up. Recall, the first article by Thompson and Ritchie was to appear in *CACM* in July 1974—two months after this first meeting. By the next spring (1975) Mel Ferentz (then at Brooklyn College) sent out an "invitation" to be placed on a Unix mailing list to three dozen sites that had received software from BTL. (Thompson supplied Ferentz with the list of those who had received RK05s or tapes from him several times in these early years.) A notice of a meeting of Unix users, to be held on June 18, 1975, at City University of New York (CUNY), was then sent to those who responded *plus* "20 new installations." A third notice, to Ferentz' revised list of 37 respondees, of the "new edition of Unix" (Sixth Edition) was sent on June 16. By then, Columbia, Brooklyn, and CUNY's Computer Center all had Unix installations.

Ferentz, Katz and Bornholdt set up the meeting, with Ira Fuchs making the arrangements. In Ferentz' words (from *UNIX NEWS*, Number 1, July 30, 1975; *Circulation 37*):

> The meeting on June 18 at the City University of New York was attended by over 40 people from 20 institutions. Each institution described briefly its function and idiosyncrasies. We will not try to reproduce them here since we expect one page write-ups for subsequent inclusion from each installation. (Several such are in-

cluded in this issue.) There was unanimous sentiment for keeping the user's group and its newsletter as informal as possible.

That 11-page issue of *UNIX NEWS* included descriptions from the Boston Children's Museum and the University of Toronto. It also included a brief article titled "MUNIX—A Multiprocessor Unix" and a mailing list.

The mailing list is fascinating as it reflects the spread of Unix—an unannounced, unsupported operating system.

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

(continued)

Institutions on the First Mailing List (continued)

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 Number 2 (October 8, 1975), Circulation 60, carried the notice: "The first meeting in the West will be held on Friday, October 31, at the Naval Post Graduate School." Issue 3 (February 10, 1976), reproduced letters from Professor Robert S. Fabry, Department of Electrical Engineering and Computer Science, University of California, Berkeley, and Lewis A. Law, Director of Technical Services, Harvard Science Center, announcing the next meetings in the West (February 27 and 28) and the East (April 1 and 2). Number 4 (March 19, 1976), carried a first letter on the installation at the University of New South Wales from Dr. John Lions and a new mailing list, now with 80 names. By September 1976, there were 138 entries. Thirteen of these were in Canada, ten in Great Britain, four in Australia, three each in Israel and the Netherlands, and one each in Austria, Belgium, Germany, and Venezuela. The other 101 were in the US.

The May-June 1977 issue of *UNIX NEWS* was its last. Beginning in July 1977, the publication was called *;login:*. Mel Ferentz had been phoned by an AT&T lawyer and told that the group (it still had no name) could not use the term UNIX, as they had no permission to do so from Western Electric. At the meeting of UNIX USERS at the College of Physicians and Surgeons, May 24-27, 1978,

> a committee of five people was elected ... with the purpose of proposing a set of bylaws for an organization of users of UNIX* installations. The people elected were:
>
> | | |
> |---|---|
> | Mel Ferentz | Rockefeller University |
> | Mars Gralia | Johns Hopkins University |
> | Lou Katz | Columbia University |
> | Lew Law | Harvard University |
> | Peter Weiner | Interactive Systems Corp. |
>
> Law was elected chairman.... The name of the committee shall be the USENIX** committee ...

It is gratifying to see the two footnotes: they reveal that the spirit that bonded Unix users together in the 1970s and which continues today had its roots in an "us-against-them" attitude combined with a sense of humor. The two name changes gave rise to a letter to Ferentz from Stephen J. Phillips, a patent attorney, at Bell Labs. Phillips' letter began:

> It was with interest that I read Vol. 3, No. 6, June/July 1978. Your choice of the name "USENIX" was rather an inspired way of avoiding the use of the UNIX* trademark in the committee name.

Phillips went on at some length about the "dilution of the UNIX mark," which he hoped USENIX "would not incorporate" in any future names. Ferentz, wryly, published the letter in *;login:*.

The name USENIX was coined by Margaret Law, until recently a faculty member at Harvard and Radcliffe; *;login:* is more interesting. Dennis Ritchie explained:

> The ; was utilitarian. During most of the early '70s the most popular terminal was the Teletype model 37. The sequence <esc>; put it in full-duplex mode so the terminal didn't print characters locally, but let the system echo them. So this se-

*UNIX is a trademark of Bell Laboratories, Inc.

**USENIX is not a trademark of Bell Laboratories, Inc.

> quence was put into the greeting message. Of course it didn't print when you used that terminal, but other terminals that appeared later didn't understand the message and so printed the ; .

The May 1978 meeting at Columbia was attended by Bill Shannon and several others from Case-Western Reserve University in Cleveland. Shannon was Sam Leffler's colleague, and Leffler went along the next year to the Toronto meeting (June 20-23, 1979). Leffler recalls that "we all went to see *Alien* there; and while walking back to the Toronto dorms, we were looking over our shoulders—you know—for aliens." The extracurricular activities were quite salient at the early USENIX meetings: Leffler and several others recalled "we played volleyball" at the June 1980, Delaware meeting. This was very important to Leffler's future—and the future of Unix. (I will return to this later, when I discuss Berkeley Unix.)

As can be seen from the very first lists of Unix users, they were far from geographically constrained. Thus, it comes as no surprise to learn that

> The Australian Unix systems User Group, or AUUG, was formally constituted on the 27th August 1984 at a meeting of Unix users held on the campus of Melbourne University, after nine years of informal existence.... The first AUUG meeting was held in 1975 at the University of New South Wales. (Peter Barnes, Secretary, AUUG)

for the University of New South Wales was the first site west of California to get Unix—in 1974.

The United Kingdom, which had received Unix in 1973, was not far behind in getting the users organized. At the Inter-University Computer Council meeting in September 1976, P. M. D. Gray of the University of Aberdeen spoke with a number of his colleagues about the growing number of Unix installations in the UK. In December, Volume 1, Number 1 of the *UK UNIVERSITIES UNIX NEWSLETTER* appeared. It contains a list of 11 installations: Aberdeen, Durham, Essex, Glasgow, Heriot-Watt (Edinburgh), Kent (Canterbury), Loughborough, Queen

Mary College (London), Westfield College (London), St. Andrews, and Sussex—four in Scotland and seven in England.

On March 4, 1977, Alistair C. Kilgour of the University of Glasgow circulated a letter suggesting the formation of a UK Unix Users Group as a DECUS Special Interest Group (SIG), with interested parties meeting after the DECUS UK Annual General Meeting, Friday 15th April. He proposed further the holding of a Unix Colloquium in Glasgow at the end of May 1977. The next few issues of the newsletter carried the rubric *DECUS U.K.* at the top of the page, and were entitled *UK UNIX USERS GROUP NEWSLETTER*. DECUS UK paid for duplication and postage.

There was, indeed, a colloquium in Glasgow on 27 May 1977, "attended by about 40 people." Within a year, the UKUUG had joined hands across the North Sea with the Netherlands, and was holding a meeting at the Vrije Universiteit in Amsterdam. The Mathematische Centrum had obtained Unix quite early. Teus Hagen told me:

> I saw the article by Ken Thompson in *CACM* and immediately wrote a letter to him asking for more details and the possibility of receiving the software, which I received immediately from him (on DEC RK05 cartridges) without any license limitation whatsoever. That was Unix Version 5. We had been struggling with a PDP-11/45 and RTL or RSX11 for a graphics project. On receipt of Unix we immediately threw the DEC stuff away and went on with the BTL stuff. This was later followed by Version 6, 6.1, 6.2, Version 7, and then our DEC VAX-11/780 (serial number 10) arrived. We had ordered it without a VMS license (DEC was astonished at that time) and used first the Labs' Unix VAX version (partly running in PDP-11 mode), and later the Berkeley stuff. [*VAX stands for Virtual Address Extension.*]

The NLUUG had come into being by November 1978. As Unix spread across Europe, other national organizations were formed, and in April 1981 the European Unix systems User Group (EUUG, now EurOpen) had its first meeting (also in Amsterdam). Interestingly, as the advent of Unix in Japan was less than two years behind Australia, so

the meetings of Unix users began soon after those in Europe and Australia.

After Haruhisa Ishida at the University of Tokyo introduced Unix to Japan in 1976, it was not long before the Japan Unix Society (jus) was formed by Jun Murai, Koichi Kishida, and Nobuo Saito. Originally, it was a Unix study group, but it grew rapidly. At ten years of age, Unix was genuinely being used worldwide.

CHAPTER

11

Why Unix?

Let's pause here, with a six-year-old experiment, to consider just why Unix had hit the international academic audience so favorably. What, indeed, was it about Unix that so many folks liked?

At the beginning of the *CACM* paper, Ritchie and Thompson list six features offered by the system:

(i) A hierarchical file system incorporating demountable volumes,

(ii) Compatible file, device, and inter-process I/O,

(iii) The ability to initiate asynchronous processes,

(iv) System command language selectable on a per-user basis,

(v) Over 100 subsystems including a dozen languages,

(vi) High degree of portability.

Let me list a few things that I like about Unix:

- Available on a number of platforms
- Multiuser
- Provides a directory hierarchy
- Shares computer resources sensibly
- Supports manipulation of files, processes and programs
- Allows interprocess and inter-machine communication
- Permits access to its operating features

Rudd Canaday has put it simply: "Unix spread throughout Bell Laboratories because people loved to use it." More recently, Armando Stettner said, "It doesn't get in its own way or mine."

In 1976 and 1977, Tom Lyon had enabled some parts of Unix to run under VM/360 on an IBM 360 at Princeton University. During 1977 and 1978, Ritchie and Steve Johnson had ported Unix to the Interdata 8/32, at the same time that Richard Miller and his colleagues were porting Unix to an Interdata 7/32 at the University of Wollongong in Australia. Ritchie has said that the port to the Interdata was one of the pieces of programming he was proudest of. It demonstrated that Unix was indeed portable to a machine that had not been manufactured by DEC. I will return to this later.

It is facile to say that Unix featured this or that, or that it was available on a machine that a department (rather than a computer center) could afford. Permit me, instead, to let Mike O'Dell, now a vice-president at UUNET Technologies, recall the summer of 1974. At that time, O'Dell was an undergraduate at the University of Oklahoma.

> When the famous 1974 *CACM* issue appeared, I was working at the OU Computer Center. We had this thing called ITF, the Intermittent Terminal Facility, which had the world's worst implementation of BASIC, and one of the guys had written some routines which let you do I/O on terminals—and this was a non-trivial feat. So a group of us sat down and tried to figure out whether we could do something interesting, like edit, using the [IBM] 2741s as keypunches. So you could edit jobs and submit them, that sort of stuff. So I was in the thick of that—I remember my blackboard being filled with data structures, because we were building everything into the files, because we didn't know better.
>
> The Unix issue came. I remember going down the hall and getting it out of my mailslot and saying to myself, 'Oh, ACM's got something on operating systems, maybe it's worth reading' [the issue was "Papers from the fourth ACM Symposium on Operating Systems Principles"]. And I started reading through it. And there were some other good papers in that

issue. But I remember sitting down and reading through this paper on the Unix timesharing system. It was sort of like being hit in the head with a rock. And I re-read it. And I got up and went out of my office, around the corner to George Maybry who was one of the other guys involved with this. And I threw the issue down on his desk and said: "How could this many people have been so wrong for so long?"

And he said: "What are you talking about?"

And I said: "Read this, and then try to tell me that what we've been doing is not just nuts. We've been crazy. This is what we want." And at that point I decided that we had to get one of these one way or another.

In fact, one of the reasons O'Dell was intrigued by this was that in one of the corners of the Oklahoma electrical engineering lab was a PDP-9, a close relative of the PDP-7. O'Dell's reading that Ritchie and Thompson had originally done their work on a PDP-7 caused him to think that he might run this wonder on the PDP-9. O'Dell told me:

I phoned Bell Labs and actually spoke to Dennis Ritchie, and said that we didn't have a PDP-11, but we had a 9 and could we get the PDP-7 version and port it? And Dennis said that he didn't think that was a good idea. (At this point they probably thought they had this cuckoo kid on the phone.) But they were very nice, so I was encouraged. And I made up my mind that we had to get us one of these.

So there I was, between naivete and chutzpah, and I called up the DEC sales office in Tulsa—there wasn't a sales office in Oklahoma City—and asked to talk to someone about a PDP-11. I got to talk to this guy, Stan Bartel, who was the salesman.

Stan was one of these salt-of-the-earth guys who realized that he'd never been able to get a mini-computer into OU—the blue death was all-controlling—and that he could do worse than have some rabid minion working in the Computer Center. And back then DEC salesmen were paid salary, not commission, so it didn't matter [*they still are not on commission*]. He was going to be down in our area in a couple of

> weeks, anyway, and he came over at lunch and brought with him the entire stack of manuals. The processor manual and the peripheral manual. I'd never seen this kind of documentation. So I read through the manuals—I was a pretty good programmer at that time—and I read through the instruction set and I said: "These guys know what's going on. This is really cool." That started the process. I was hooked.

As I mentioned earlier, *The C Programming Language* wasn't published until 1978. What were the "unconnected" in Oklahoma to do? Again, I'm going to let O'Dell speak, as his story is very like that of many others in the period 1974 to 1978.

> My first exposure to C actually came through the University of Illinois. Steve Bunch went from OU to UIUC [University of Illinois in Urbana/Champaign]. And we went up to visit Mecca. And we got to see Unix ... Steve Bunch and Steve Holmgren and Mike Mullen ... the same group of guys who put Unix up on the ARPANET. Anyway, Steve [Bunch] sent me a listing for the kernel for the Intelligent Terminal, which consisted of the context switcher, the stuff to do semaphore control keys, the IPC mechanism, and one or two other snippets. So it was five or six sheets of paper. And he had written some marginal notes, because I had never seen the programming language, and he explained what these funny operators like ++ would do. And he said, "See if you can figure out how this works." I recall the context switcher was about six lines of assembler and he mailed it to me, basically, to see if I could figure it out. And I figured it out.
>
> Later, I wanted to know more about this stuff, so he made me a **tp**—this is long before **tar**—the old **tp** format—tape of the man pages in nroff output form. And he sent me this tape, and said, "There's something interesting on here, if you can figure out how to print it." And he sent me the actual **tp** man page.
>
> So I sat down and, basically over a long weekend, wrote a ... program to crack the format, get the underlining right, and print the Unix man pages. And that's where my first man

> pages came from. I probably still have them somewhere in a box, because I never worked so hard to get something.
>
> Then we started passing around the man pages: we couldn't get the system, but we could see what it did. That's when the bug really hit.

Teach yourself. Figure it out. How do you think it should work? Mike wasn't alone. Dave Yost, one of the creators of the RAND editor, told me of how he had looked at some code (in the mid-1970s):

> So I sat down and looked, and I saw those curly brackets [{ and }] and I said, "wow!"

In C, { and } enclose the statements that make up a function. C doesn't have the DO-END of PL/I or the **begin-end** of Algol or Pascal. C is in many ways a simpler language, a language more like a natural language (like English). It is not a "very high level language." As Thompson and Ritchie put it: "C is a general-purpose programming language which features economy of expression, modern control flow and data structures, and a rich set of operators.... In our experience, C has proven to be a pleasant, expressive, and versatile language...." In my experience, C also lends itself to programs with style and elegance.

In 1987, Ritchie offered his thoughts about the success of Unix at the USENIX Technical Conference. He talked about the "simple, coherent, and powerful model of computation"; the "metaphor of the toolbox"; and portability.

The first and last of these were important: its availability on several platforms and the elegance of Unix and C were a good part of the system's popularity. Style and tools were the other parts.

CHAPTER

12

Style and Tools

Just about the time that Ritchie and Thompson were working on their paper for the ACM symposium, Brian Kernighan and P.J. Plauger were collaborating on *Elements of Programming Style*. Kernighan has said, "We were writing it in 1973, and we finished it early in '74." Plauger remarked:

> Brian Kernighan and I ended up with adjacent offices at Bell Labs in Murray Hill almost by accident....
>
> We began by commiserating over the sad state of computer programming and ended up in the authoring business, both for the first time, by writing this little book over a period of about four months. As far as I know, it marks the first time that "programming style" was identified in print as a legitimate topic of discussion by adults.
>
> The book is terribly dated now, of course....

Kernighan told Peter Collinson:

> Plauger and I wrote the style book and that worked out pretty well and was fun. The idea behind the style book was to take a large number of programs and criticize them: that isn't right, that could have been better. We weren't saying *how* to do things, rather how *not* to do them. It was FORTRAN and PL/I, at the time they were the dominant languages in the community we were aiming at.

Programming Style made quite a splash. But, as both Plauger and Kernighan remark, it concerned PL/I and FORTRAN. Their next collaboration was far more general, and far more influential. Collinson asked Kernighan how he became involved with the *Software Tools* book, he responded, "It's hard to say."

> A couple of years later, we decided that the time had come to tell people a little about how they ought to do things. By then, we had a clearer picture of some of the benefits of the Unix environment: the advantage that you can get by piping programs together and building things that were going to be filters. It wasn't clear what to use as a language for the book. C, of course, didn't exist in very many environments. I had already done Ratfor [Rational FORTRAN]. Ratfor was around, it had simply stolen the good appearance of C, but didn't add much beyond it. It was fine, it converted FORTRAN into a programming language.
>
> We decided to use Ratfor as the programming language for the book which was really unconventional, [because] there were very few users. There was no-one in this group here, because nobody wrote in FORTRAN at that time except the numerical analysts who were worried about portability and they did not want to produce unreadable FORTRAN which is what Ratfor did.
>
> The original version was written in C with a small amount of yacc grammar. [yacc is "yet another compiler-compiler," written by Steve Johnson.] Given that as a bootstrap, it took a very short time to write it in Ratfor. It can then be bootstrapped on a machine only running FORTRAN. Part of the program distribution for the Software Tools package was Ratfor in Ratfor, and [part] Ratfor in FORTRAN. The first thing on the tape was Ratfor in FORTRAN so you could just peel it off and start running.

Collinson then asked, "So you have Ratfor, how do you get from there to Software Tools?"

> Then you sit down and ask what are the things that are interesting in a Unix system. It's the fact that there is this large number of very small tools that you can glue together in interesting ways. That takes you through the first three chapters of the book: the ideas of character input and character output as the most common denominator, text is everything, data is just streams of lines of text, and maybe up as far as archiving.
>
> Then we do regular expressions. These are a very important fundamental notion in Unix. Using this, the next couple of things are the pattern searching, the **grep** family, and then the editor. Actually, sorting is one of those tools that you put in as well. There are interesting algorithmic things that you can say about sorting.
>
> The only other thing that we do is text formatting, so we do a little simple formatter. Finally, we do Ratfor, saying this is the program you have been using all along. This is Ratfor in Ratfor.

Software Tools came out with a Ratfor tape, which had been written for the book. But the book came first. Kernighan remarked,

> Ratfor was independent of the book, but it became the vehicle for the book. The version that was described in the book was simpler than the version that was on the tape because you didn't learn anything new by having the full implementation described to you. There were groups like the Software Tools group that sprang up; which was primarily done at the Lawrence Berkeley Lab in California. These were people like Debbie Scherrer, Joe Sventek, and Dennis Hall. They set up the Software Tools group and they did some really nice stuff with it to create the "Virtual Operating System" and all kinds of stuff. Neat tricks across multiple machines. It was all done in a very clean way.

Recall the Unix Philosophy: **Write programs that do one thing and do it well**. That was the idea behind the tools. And it was the idea that fired up those folks at LBL.

Debbie Scherrer told me:

> You know, I started at UOP [University of the Pacific], but they had only one computer course. So I transferred to Berkeley. And when I was working on my master's, around 1968, there was a job at LBL. It was great. There was Dennis Hall. And Joe Sventek was working downstairs, in an altogether different department. His degree was in physics, and he was doing something secret that had to do with weapons testing.... We were maintaining systems for all these researchers who didn't care at all about computing, just that it worked. But one day Andy Tanenbaum [of the Vrije Universiteit in Amsterdam] left a copy of this book, *Software Tools*, on the table in the Lab. He said, "You might be interested in this." So I read it. And I thought it was wonderful. At the same time that Andy Tanenbaum mentioned the tools book, Dennis Hall at the Lab discovered it. It was Dennis who dragged me into it (I had been working on another project at the time), and it was Dennis who was the PI, got the funding, directed us, to the extent that either Joe or I could be directed. So over the weekend I sat down and I started to implement all the tools. It was great! And Joe and Dennis and I thought, well, if we're supposed to be supporting research, then these tools support research, and we implemented all the rest. Oh, and some other stuff, too. Jim Poole at DOE was always extremely supportive and gave us funding. Nobody else ever got the point, and went on along with their ghastly FORTRAN programs, extended DCL scripts, and other sillinesses.

It was that fast.

Mike O'Dell told me, "Remember, Debbie knew Brian, and he knew what they were doing, so he pointed people who asked about the tools to LBL. And that started the user group." Recall, *Software Tools* wasn't about Unix, it was about philosophy and style. Ritchie said:

> The tool-using approach is powerful and intellectually economical, but it takes imagination to use. It may also be more

> costly to combine simpler, more general tools than to build a more specialized one.

Interestingly, Scherrer and her colleagues realized just how powerful the tool concept was. It was clear to them that the software tools could be used on just about any available architecture. So Hall, Scherrer and Sventek wrote a Virtual Operating System (VOS) that would serve as a pseudo-interface between the software tools (in Ratfor) and whatever OS was running. Scherrer said, modestly, "VOS was Dennis' idea."

I asked Paula Hawthorn, who had the distinction of being Mike O'Dell's manager at LBL, about the tools. She told me:

> When I joined the Computer Science and Math Department at LBL, Debbie and Joe and Dennis Hall already had completed several releases of the software tools, and were in the middle of the "but is it really research?" issues.
>
> The problem was that LBL was supposed to be doing research, and how could we say that making another release of the Software Tools really was research? "But is it really research?" is the bane of the engineer who wants to make the results of the research into something that people actually use, because many of the things you do to deliver a working system are not necessarily research, but the research is totally invalid if it has had no field trial. So my memories of that time are colored with the "but is it really research?" fight, and the "if you need to use Unix, why not just use Unix?" battle. To a first approximation, the Software Tools appeared to be just papering over non-Unix systems so that they were more Unix-like. This also caused heated discussions...

Communicating with a few others about the tools, a Software Tools User Group was founded in 1978. The article on VOS appeared in *CACM* in September 1980, but a STUG meeting had been held on June 16, preceding the USENIX meeting at the University of Delaware. Wally Wedel reported on the meeting in *Software Tools Communications* (#4, October 1980)—an aperiodic newsletter initially produced through LBL. The software tools were already available for several DEC

operating systems, for the Control Data Cyber, for IBM systems in a TSO environment, and on an SEL 32/77 under MPX 1.3 (MPX was a primitive operating system; NASA's Ames Research Center was unable to implement a shell under it, though they got every tool to work, etc.). By the November 1981 issue, Phil Scherrer (formerly of Unicorn Systems, now at Stanford University) was able to report that:

> The software tools have now been completely ported to a microcomputer environment. The CP/M (trademark of Digital Research) operating system which runs on 8080, Z80 (trademark of Zilog), and 8085 processors, was chosen because of its wide availability on systems with (barely) sufficient hardware. ...
>
> All the tools from the STUG distribution tape, as well as well as many of the extensions specified in the CACM article, have been brought up and run quite well.

The same issue of *Software Tools Communications* listed nearly three dozen architectures on which the tools had been implemented, together with the names and addresses of the implementors. The manufacturers ranged (alphabetically) from Burroughs to Zilog; the machines (in size) from the Z80/8080 (64k bytes of program memory) to the IBM 370 and the DEC 20. Geographically, the implementors spanned the globe: from Kawasaki, Japan, across the US and Canada, to the UK and Eindhoven in the Netherlands.

One of the sites where the Tools became important was Georgia Tech. Gene Spafford, who was there for much of the time (though not for the very beginning), told me:

> In the mid-to-late 1970s, the folks at Georgia Tech had several PDP-11 machines running Unix. These were used both for research and as part of a medical database research program.
>
> Several of the students and faculty got hooked on Unix and wanted to bring it up on the main research computers in the department, which were Prime machines. The Primes, at the time, were really nice machines. They had virtual memory, good multi-user capacity, and many other nice features. Unfortunately, the Prime architecture was not an easy machine to write an OS for. Unix was not going to map onto the machines.

As an added inducement, the last remaining PDP-11 got "melted down" by a DEC field engineer who jammed the power supply in upside down and sent 110 VAC into the backplane. DEC never made the situation right, so we were without Unix....

[The students then wrote a system, but] Georgia Tech wouldn't allow this system to be given away, and so it was licensed to universities and companies who wanted it enough to pay for it. At one time, many score places were running it. Prime even marketed it. After several years, three things came along to kill it:

- Some folks at Prime weren't pleased that customers liked the Software Tools interface better than Primos, so they stopped providing special assistance to the Software Tools team. Many of the "champions" inside Prime engineering left to form Apollo. The ones who remained built some Software Tools ideas into Primos and turned their backs on the development team at Tech.
- The Software Tools group wrote a very good C compiler and library for the system. They wanted to switch everything over to C and form an independent company to provide support. Unfortunately, the administration at Georgia Tech got petty over ownership rights to the compiler and associated code. The result was that the team, as a whole, quit and Software Tools was effectively left without further support.
- VAXes began to be shipped in quantity with BSD Unix available. People who had previously used Software Tools could now get real Unix and virtual memory for the same (or less) cost than their Primes with Software Tools. Demand rapidly fell off.

The principals involved were Dan Forsyth, Paul Manno, Perry Flinn, Allen Akin, and Win Strickland.

Spafford's tip led me to Manno and Forsyth. Forsyth, after huddling with his former colleagues for the sake of accuracy, told me:

> The Software Tools project at the Georgia Tech School of Information and Computer Science began as the result of several independent events in 1976. First was the publication of *Software Tools*. Second was the Army's interest in portable COBOL. Finally, and closer to home, was the plight of a group of seniors who returned from summer break to find that their beloved Burroughs B5500 had met its end for lack of money for maintenance and that all the PDP-11s had been walled off to protect confidential medical research data. In the place of their formerly idyllic computing environment, funded largely by the Army's research project, was a shiny new Prime 400.
>
> A UNIX port was out of the question for many reasons, including the lack of access to a PDP-11 for porting, the initial lack of documentation on the internals of the Prime, and a lack of faculty interest. Meanwhile the local programming environment consisted of a machine architecture that looked like a cross between a GE-645 and what was to become an Intel 80286, a FORTRAN-66 compiler, and a timeshared operating system that, at its best, could accept commands consisting of two six-character file names and ten octal numbers!
>
> One of the senior design projects suggested by Dr. Philip H. Enslow (PI for the Army's research project and soon-to-become faculty advisor to many of the students) was the porting of Ratfor to the Prime. This was clearly before any of us knew what an excellent job Brian [Kernighan] and Bill [Plauger] had done. The tape was obtained from Addison-Wesley and Ratfor was running before we knew it.
>
> Allen Akin, Perry Flinn, and Jack Waugh began to recognize how the tools could be put to use. At some point in early 1977, the Primos command line interface was deemed entirely unsuitable for the graceful connection of tools, so a simple UNIX-like shell was cobbled together. Paul Manno and I were drawn into the effort shortly thereafter.

Although the desire to create an elegant computing environment was strong, each of us was primarily commited to other projects. Nonetheless, we often found that by following K&P's advice to extend or add new tools, we could accomplish our project work more effectively than starting from scratch each time. With this incentive, the Software Tools Subsystem began to grow rapidly. Since it was a much more effective environment, other student and faculty use began to grow as well. We received many contributions from other researchers and in return found it necessary to establish documentation standards and write bundles of documentation to avoid losing all of our time to questions. Of course, we had to extend the text formatter and other tools to accomplish it.

By the summer of 1978, we had an effective replacement for both the Primos command and programming environments. In addition to the original Software Tools, we had a large subroutine library, basic shell, extended Ratfor, electronic mail, bulletin board, full screen editor, and many other new tools.

Through the efforts of Phil Enslow and David Nelson, head of research at Prime, the research division of Prime Computer also became interested in our efforts. With the assistance of Georgia Tech and Prime, Allen, Perry, and I made a visit to Prime Research that summer and came away with the inspiration to produce an "advanced" command interpreter. We set about combining features from Multics, V6 UNIX, RDOS, and other systems into a single command language. The examples set for us by Software Tools, Unix, and Algol 68 convinced us to strive for the goals of elegance, orthogonality, and reusability.

By the beginning of 1979, we had produced the "new shell" that, in retrospect, looked quite a bit like the externals of the Bourne shell (although we had never seen it). It included several interesting concepts: multiple standard inputs and outputs along with syntax to connect directed graphs of tools, arbitrarily long command lines, control structures managed by external commands, and scoped variables that were

treated as objects whose execution yielded their value on standard output.

Prime Research acquired a copy of the Subsystem, as it was fondly called, and made it available internally, since it was superior for text processing and research programming. It was perhaps during this time that we established a less-than-desirable relationship with the engineering group at Prime. It seems we had a knack for implementing and releasing features, especially those dealing with improving the operating system interface, at nearly the same time that Prime engineering declared the task to be impossible or impractical. Of course, we had the luxury of a small installed base and no commitment to backward compatibility. For whatever reason, our work had only an indirect influence on later releases of Primos.

With the acquisition of more Prime systems and a new-fangled way of connecting terminals called "ethernet," the Software Tools Subsystem became the administrative link for the School of ICS. It handled local electronic mail as well as text processing for the department, and it provided many graduate students, including me, with submission-quality theses without paying a typist. Rather than release our efforts to the public, the powers-that-be at Georgia Tech decided that licensing for a fee was more appropriate. Even at the then-astronomical fee of $3,000, many dozens of copies were licensed to Prime sites around the world, including England, Germany, and Australia.

In 1980, Debbie Scherrer from Lawrence Berkeley Labs invited us to attend a Software Tools User's Group meeting as part of the USENIX meeting in Toronto. Georgia Tech assisted us in flying to Toronto and we found ourselves surrounded by many new ideas. These were greatly interesting meetings that exposed us to new perspectives that were unimagined in our insular, departmental environment. We were allowed to contribute some of our software to the user's group, but because of the licensing considerations, we could never release the "new shell" and the more interesting tools.

By the time Allen, Perry, and I left in 1981, the Subsystem had a completely rewritten Ratfor preprocessor, a parser generator, a language-independent code-generator and a C compiler (in Ratfor!), as well as thousands of pages of documentation. The project was taken on by Jeanette Myers, Terry Countryman, Peter Wan, Scott Lee, and Arnold Robbins who managed it well. During that period, AT&T began offering low-cost Unix binary licenses. Within a year, Unix boxes could be purchased for one-fifth the price of a Prime or PDP-11. Before long, Unix, vi, and troff were available to the masses, and the commercial need for the Subsystem began to wane.

Around Georgia Tech, 4.1BSD Unix VAXen, AT&T 3Bs, and Sun 3s had been sprouting like weeds, much to the consternation of the central computing czars. CSNet, NSFnet, and USENET made Unix cycles a necessity for participation in the growing Internet community. A final release of the Subsystem was made in 1985 and the code was put into the public domain. Shortly thereafter, it and the Primes were retired from local use, fittingly replaced by the software whose lack had spurred its creation.

The Software Tools had given rise to another grassroots movement. Another set of avid users, only partially overlapping those using Unix. But the philosophy was portable: Write programs that do one thing and do it well.

Many of the names involved in STUG are familiar to Unix users, too: Neil Groundwater, Mike O'Dell, the Scherrers, Joe Sventek, Dave Stoffel, and Wally Wedel.

I last mentioned O'Dell when he was still an undergraduate at the University of Oklahoma. He had now completed his master's thesis ("I looked at it again a while ago," he told me. "It's not terribly embarrassing.") and was job-hunting. In his words:

> I was within two weeks of taking a job with 1127 [the Research Group] at Murray Hill. I went off and interviewed. And then the death march started, where the paperwork went from desk to desk to desk, arriving on a person's desk just as

> they were leaving on a two-week vacation. So what should have been a two-week approval process ... turned into two months. So I went —in the meantime—to the [USENIX] January meeting in Boulder and went to the Software Tools session. I had been interested in the Software Tools stuff as a way to get interesting stuff onto the IBM machine, while we were still fighting [trying to get Oklahoma to purchase a DEC machine]. So I went to the Software Tools thing and Debbie Scherrer spoke and she talked about all the wonderful stuff they were doing and at the end of her talk she said, "Oh, and by the way, we're looking for someone to run our Unix systems for us." And I all but climbed over people, I remember literally throwing chairs out of the way as the meeting was breaking up, to get up there. There were six people from the Lab there—Joe Sventek, and Peter Krebs, Debbie, Dennis Hall, Roland Johnson was probably there. The interviewing procedure was go to lunch with six people and they get to eat Chinese while you talk—tag-team interview.
>
> Anyway, I went home and they flew me out and I gave a talk on my thesis and they made me an offer. So I had this awful decision to make between LBL and BTL. And BTL kept delaying and delaying and delaying. And finally LBL said they had to have a decision. So I called up Dennis [Ritchie] ... and he said I should take the other job.

So O'Dell went to LBL, where he became the Unix guru, the ARPANET liaison (the 11/70 was an early ARPANET host), and several other things. Neil Groundwater was also at the Boulder meeting. He, too, got involved with STUG:

> I became familiar with the STUG at the Boulder USENIX Conference in 1980. At about that time it seemed that LBL (Sventek, Scherrer, et al.) had done 90% of the tools and work, but others were beginning to "use the source" (tools).
>
> We were involved in consulting to the US Navy on several projects that had differing programming environments.

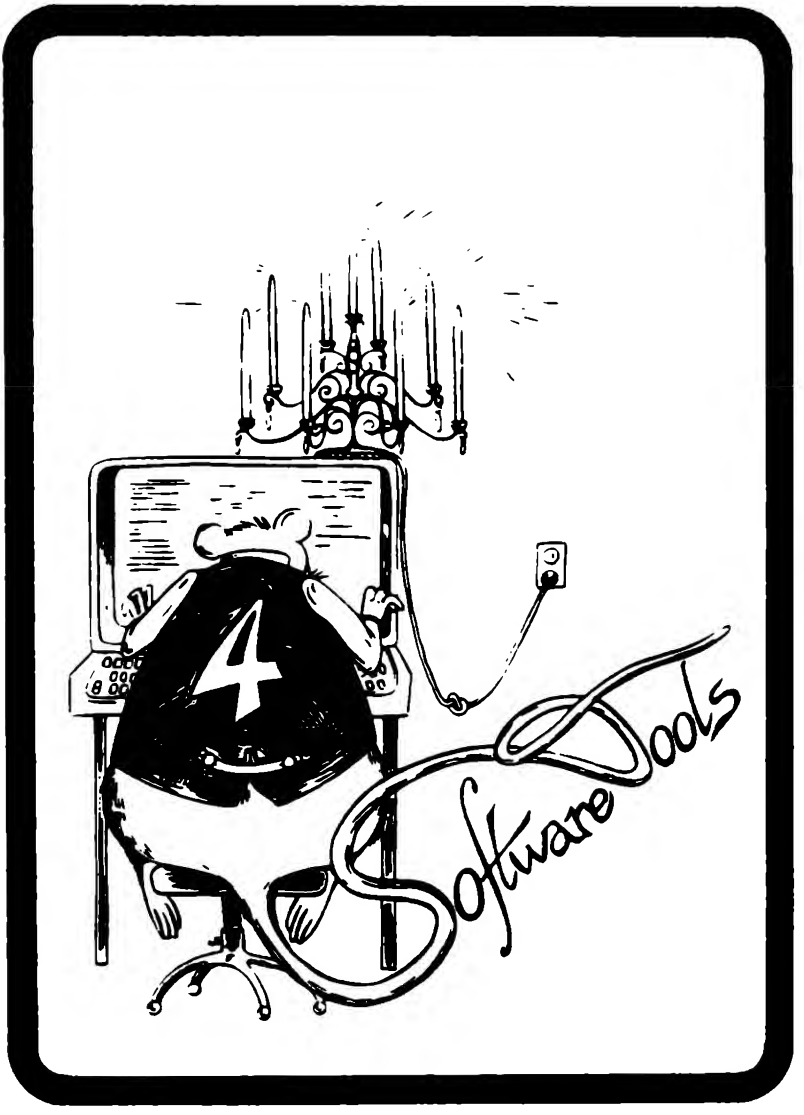
We suggested the Tools as a means for them to control their build environment on multiple platforms, but it turned out that they primarily used it on VAX/VMS.

My biggest contribution was a text control system (SCCS to Unix fans) that performed **create**, **delta**, **get**, and **edit** functions for version control of text files. It was based on the **diff** algorithm published as a Bell Labs Technical Memorandum by J. W. Hunt and Doug McIlroy.

STUG was formally set up at that Boulder meeting. Groundwater was elected to the first Board. And O'Dell moved to Berkeley.

Debbie and I shared an office for the three years I was at the Lab. She's the dearest person in the world, with more energy than any six people should be allowed to have. The Software Tools stuff sort of peaked, leveled off. And the sad part is that the Unix community has never really learned the lessons that the Software Tools people were teaching. The Software Tools folks, I would argue, understand portability to a degree that no one does. And the Unix community is much poorer for it. The problem with Unix is that it was so easy to write code, you saw no reason not to. This is not a version of salvation through suffering. A lot of things in the Software Tools stuff worked better because they really got it right, and 5,700 people didn't try to reinvent something.

The tools concept was flourishing beyond the realms of Unix.



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

CHAPTER 13

PWB and MERT

Recall that the first system Thompson and Ritchie wrote was for the PDP-7. First Edition was for the PDP-11/20 (1971). Fourth Edition, which was basically Third Edition rewritten in C, made it out of Bell Labs towards the end of 1973, while it gave rise to another project: Programmer's Workbench (PWB). But First Edition had gotten Charlie Roberts thinking. Bob Morris said that Rudd Canaday never got the credit he deserved. The same is true of Roberts. MERT (Multi-Environment Real Time) was Roberts' brainchild, although it was written largely by Heinz Lycklama and Doug Bayer. Roberts told me:

> By early 1970 I knew there was a Unix on the PDP-11, implemented in B. But Steve Johnson was already talking about a C, it seems to me. In October 1970 I left the Murray Hill Computer Center and returned to Research. I was aware of Patents as customers, and I recall thinking about 1127 pitching Unix as a timesharing system for document preparation. Computer Science research was like a lot of academic departments: it had two parts. Sam Morgan headed the part where Dennis and Ken and Doug McIlroy were—the CS part; and Hank McDonald was the Director of Computing Research for Systems—the EE part. I went to work for Hank, who wanted to build the next-generation ESS for the next decade (the '80s).
>
> We were on the fifth floor, right below Doug and the other guys. I had Heinz Lycklama and Dick House and Carl

Christensen working for me, along with some programmers. Carl and Heinz were working with this DDP516—Digital Data Processor, a small computer with its own OS and hardware. But I'd been reading some articles by Per Brinch Hansen; he later put it all together in a book [*Operating System Principles*, 1973], and he got me thinking about stuff like microkernels and a layered OS and things that didn't really mature 'til Mach 3 and Chorus and Windows NT. It seemed to me that Unix was focussed too much as a timesharing system, and I wanted to work with a dedicated custom system—for switching and large databases, real time and transaction processing.

Well, Hank helped me to get the money to buy an 11/45 in '71 with big disks, and for some months I worked on my system, which we called OSKER—**OS KERnel**. Somewhere between October and Christmas, I lured Heinz Lycklama off the DDP. He and Carl just didn't see the conceptual steps I wanted to implement. But Hank encouraged Heinz to work with me and we got money to hire someone late in '72—Doug Bayer came in '73—and Heinz really got OSKER going. Around that time, I had a talk with my fellow department head, Doug McIlroy, in which he suggested that we link up. But I just didn't see the possibilities and I was afraid that it was just a way to get my staff to give up OSKER and work on Unix. Then Heinz suggested that we "put Unix on top" of OSKER, and that's what became MERT. We had Unix and real-time processing. It was fantastic. But Heinz lost interest in the kernel and Doug was very much the junior person. I felt very overstressed and in 1973 I made the decision to move to Holmdel to work more on communications and picture processing.

When I left, Heinz began to develop the Unix MERT: by '74 that was the thing the Systems Lab worked on—they never progressed with the work on the kernel. MERT got shipped to Columbus and to Indian Hill around the time Berk Tague formed the USG. Indian Hill created Duplex MERT—DMERT around '78. It wasn't a portable microkernel. It was very hardware dependent. It ran on the 3B20D. But it's still there today in every 4ESS and 5ESS.

I'll return to Roberts' narrative later; but first let me proceed to Programmer's Workbench. PWB was produced by a group at Bell Labs that was unrelated to 1127. Initiated in mid-1973 by Evan Ivie, the group was led by Rudd Canaday and originally supported a version of Unix designed for large software development projects. Though it began as an offshoot of Third Edition, PWB was really a Sixth Edition offshoot. As will be seen, Sixth Edition (1976) had a large number of descendants. By June 1977, when T. C. Dolotta and R. C. Haight wrote *PWB/UNIX—Overview and Synopsis of Facilities,* PWB supported "in excess of 1,000 users." A year later, that number had grown to 1,100—nearly all internal to AT&T.

The system ran on DEC PDP-11/45s and 11/70s. "A typical PWB/UNIX system costs about $120,000 and can support 24 simultaneous users with ease. Larger systems can support twice that number." In addition to the now-standard Unix facilities, PWB made two genuine contributions: RJE (Remote Job Entry), which "provides for the submission and retrieval of jobs from an IBM host system...," and SCCS (the Source Code Control System), an "integrated set of commands designed to help software development projects control changes to source code and to files of text (e.g., manuals)."

It may be worth pointing out here that Ted Dolotta, who was supervisor of the PWB/Unix group until 1978, became supervisor of the support group, and was responsible for the **-mm** (memorandum) macro package for **troff** written largely by John Mashey, who also wrote the shell that was an influence on the Bourne shell, but which long battled with it. Dick Haight, among other things, wrote **find**, **cpio**, and **expr**.

CHAPTER

14

Utilities

A vast array of programs comes with every Unix installation, no matter the derivation or vendor. These include editors, formatters, compilers; utilities that enable the user to alphabetize lists, to keep track of different versions of programs or documents, to "reach out and touch" other users. This chapter will be devoted to a very few of these: some very early—troff, eqn, tbl, yacc, mail, make, awk—a few more recent—UUCP, TCP/IP. The last two have given rise to other utilities (Mike Muuss' ping and Earl Cohen's finger are good examples) and to a worldwide community of users who "converse" with one another virtually instantaneously. This community is vitally important to Unix. Problems, solutions, programs, complaints, announcements, etc., were and still are freely exchanged. An entire community has been built up of network users, estimated at 30 to 35 million people — a country larger than Canada, as John Quarterman has pointed out.

Mail was already in Version 1, as were roff and ed. Once Morris had moved roff to the 635 and McIlroy had rewritten it in BCPL, Thompson moved it into Unix. But Joe Ossanna then pushed roff into troff, a true formatter for typesetting documents. Ossanna died on November 28, 1977, but McIlroy told me:

> Joe was the quartermaster. He travelled the vendors, spotted what was new in hardware, dealt with the plant department and on and on. He was a ramrod and a peach.

eqn came in V5. Brian Kernighan has said that it was his first major contribution to Unix. As he told Collinson:

> The programs were all in assembly language at that stage. The original version of **troff**, which was probably written about 1973 or late 1972, was an assembly language program and was very specifically intended for the Graphics Systems typesetter. To this day, there are things wired into the syntax of the **troff** language that are relics of that ancient machine.
>
> Joe Ossanna converted it to C, I think under considerable pressure, he didn't really want to convert it into C. He died in late 1977 and I inherited it, kind of de facto, although it was at least a year and a half before I tried to compile it. I started to work on making it independent of the Graphics Systems typesetter because at that time that was clearly on its last legs. I started putting in graphics things because the new typesetter, a Mergenthaler machine, had some limited capability to do graphics. You could position dots so finely that you could draw lines and curves.
>
> Mike Lesk did **tbl** probably in 1975. Quite quickly after Lorinda Cherry and I did **eqn**. [Cherry told me that she had started the project and that Kernighan had "joined in"; but Kernighan gets the credit for the spectacularly successful syntax.] **eqn** was the first document preparation pre-processor. It was one these things where we were forced into a good idea, necessity is the mother of invention—or something. **troff** was already fairly close to the limit of the address space of the PDP-11/40s. It was written in assembly language. I never learned the assembly language for the PDP-11s, I think that Honeywell machines were the last ones where I knew the assembly language.
>
> Joe kept **troff** pretty close to his chest anyway, he was just incredibly willing to help modify it to make it do something, but it was his program and he wasn't going to let somebody else go and do their own thing with it. The combination of all these things meant that Lorinda and I were pretty well forced into making a separate program.

> The confluence of several things made that work out pretty well. One was that C was just becoming fairly usable. The yacc parser generator had just appeared and pipes had just been invented. All these things taken together made this a feasible way to do business.

"A System for Typesetting Mathematics" by Kernighan and Cherry appeared in the March 1975 *CACM*. "Tbl — A Program to Format Tables" by M. E. Lesk was Bell Labs' Computing Science Technical Report #49, September 14, 1976. Together with troff, these pre-processors enabled the Unix user to typeset a vast variety of complex documents. The elaboration of macro packages, supplementing the -mm "memorandum" macros, extended the facilities yet further. Mike Lesk wrote -ms and Eric Allman wrote the -me macros; only -ms appeared in the Research versions of Unix, -mm appeared in Systems III and V; -me and -ms in BSD. (Lesk also wrote refer, which provides references and footnotes using a bibliographic database.)

yacc—yet another compiler-compiler—was Steve Johnson's creation. He started at BTL long before there was a Unix:

> I had my first summer job there in 1963. I had always been interested in computers, and when I got my bachelor's, I looked at the possibility of going to graduate school. There were only a couple of places I could have gone, and most of those were in EE departments, where I would have had to take courses in the construction of power stations and other things that I had absolutely no interest in. So I decided to go on in mathematics. So I went to Bell Labs in the summer of '63. Tom Crowley was my first boss there. ... And they had a rule that, basically, you couldn't work for the same group two summers in a row. So the next summer I worked with the Math group, on ALPAC. And the summer after that, sort of rounding out the program, I was in the Human Information Processing group, which had a wonderful collection of acousticians and linguists and psychologists — people doing design of experiments and data analysis. They had just produced some work on multidimensional scaling, which led me to my first paper, on clusterings, which has more citations

than everything else I've ever published — 700 and something citations.

When I finished my thesis, I came to Bell Labs and I had my choice as to which group I wanted to be in. And I picked the Human Information Processing group because they were doing some work with computer music, which I was interested in. I did a variety of things there. At that time Multics was supposedly the solution to heavy and intense computing, but it was a catastrophe. Basically, we had taken the 7094 and shipped it to the Indian Hill Laboratory, and there was no computer system available, except the GECOS on the 645 that was supposed to handle 1,000 users and could barely cope with one. [McIlroy pointed out to me that the 645 "did plenty of batch processing."]

So, after a year, in a desperate effort to get this multimillion dollar hardware to actually run some FORTRAN programs, I agreed to help this effort to introduce timesharing. So, in 1969, I actually went over to work with the Comp. Center and a small group. In those days, the Comp. Center was run out of the Computer Science Research Lab. And then there was this collection of clerks and computer operators, key-punchers and so on, who reported in to the Research Department. After I'd been there for about a year, they actually set up an organization to run the Computer Center, which was long overdue. I decided I'd rather stay in Research, and that's the backdoor way I got into the group that did Unix.

I first worked with Dennis Ritchie on another computer algebra system, called Altran; Dennis wrote the compiler for the Altran language. It was a recursive descent compiler that he wrote in FORTRAN, which was an absolute *tour de force*—like building a skyscraper out of toothpicks. He had written a compiler for a simple language called B that ran on the Honeywell system—we used it to write some system programs. But when he and Ken began working on Unix, he abandoned his B compiler, and I sort of adopted it. There were a few things that I wanted in this language, for example, I added the exclusive OR operator. I learned a ton of what I know about compilers from working with Dennis and looking at a really decent

piece of work. I also became a believer in the idea that you can learn a great deal from reading other people's code. Reading a masterpiece. Great books.

Well, when I wanted to add this, I talked to Al Aho, because I had heard that he had an interest in new methods for handling expressions. This was quite a funny scene. Al kept on nodding and saying "Yes, I've read this paper by Knuth and it's a much better method." So we worked out a simple grammar for expressions in B, and he kept on saying "I'll make up the parsing table for you." But it got postponed a couple of times. Finally, he went up to the stockroom and got the biggest piece of paper they had, about two feet square, spread it out on the table, and divided it up into little, tiny squares. And then he started making little incantations over the thing and muttering and writing these little symbols in. And I watched him for a while, and he said "Why don't you go do something else, and I'll let you know when I'm done?" So I went away and came back every couple of hours and he was still muttering over this thing, as his pencil moved across. And eventually, at the end of the first day, he said, "I'll finish this up tomorrow." Finally, the next day, he said, "It's finished" and handed it to me. And I said, "What do I do with this?"

So he showed me how to make a parser, and we typed the table in. We parsed a couple of expressions correctly, and then we parsed another expression and it was wrong—there was a bug in the table. And Al said, "Oh, no!" and then spent another two or three hours erasing and we typed the new table in and got rid of that bug, and then there was another bug. So I said, "Al, why don't you tell me what you're doing?" And he said, "Well, OK, it's really not very hard." And he told me how to make the table. And I said, "Oh, well I can write a program to do that." He said, "Really?" So, that's how yacc was born.

Johnson also wrote the first portable C compiler, lint, spell, and worked with Dennis Ritchie on the port to the Interdata 8/32 (1977/78). lint is a C program checker. It is a handy program that de-

tects features of C program files that are likely to be bugs, nonportable, or wasteful. spell is a spelling checker. It collects words from named files and looks them up in a spelling list. Words that neither occur among nor are derivable from words in the spelling list are printed on the standard output. spell wasn't the first spelling checker. The prize for that goes to Bob Morris and Lorinda Cherry for typo. It was included in Versions 3 through 7. spell first appeared in V5. typo, in Steve Johnson's words,

> just never caught my spelling errors. I'm a very phonetic speller and mistake *-ible* and *-able* and *-ance* and *-ence* and typo never caught those. It was based on the statistics of trigrams, and my errors all occurred. But typo was very good at finding *th* with the *e* missing. It would sort the things by how unlikely they were to appear.

Johnson's spell actually referred to a word list, which could be expanded by the user.

He told me,

> I went away on sabbatical in 1973, from January to September, to the University of Waterloo, to do some work with Morven Gentleman [now at the National Research Council of Canada]. Morven had worked on the original ALPAC system and was interested in computer algebra—we had done some work together—and he was a genuinely interesting person to do research with. It turned out to be an awkward time to be gone. The C language was invented while I was gone. And when I returned, yacc had been translated from B to C by somebody else. And there was also the first portable C compiler—so-called—that the same MIT co-op student, Alan Snyder, had done. And it was a very provocative attempt. It had a number of very good ideas in it, but because of small memory capacity of the PDP-11, it was a four-pass compiler. And it was extremely expensive—each pass spent most of its time reading the results of the previous pass and writing results for the next pass with hardly any time for doing work.
>
> So it was extremely slow, as well as being a great contrast to the old B compiler, which compiled very quickly. And Al

had sort of gotten it to work on the Honeywell, but there were still a lot of bugs in it. Well, he went back to MIT around the time I returned from Canada, so I inherited this thing. And my boss [Elliott Pinson] gave me a very strong suggestion that I should take it over. So I set about trying to fix the bugs in the code generator, and discovered that a lot of them were inherent in the way Al had done things. You know, at that time code generators worked on what I called the "woopsie" approach: you set out believing that you have all the registers you'll need, and that things will end up in the right place, and when something goes wrong you say "woopsie," and start looking around for something to store in memory. Most of the time this would give you good code, but when it was bad it was horrid...

Well, I was complaining to Al Aho about this, and those conversations led to what I think was my very best technical work on the theoretical side—an algorithm for generating code for expressions. It guarantees optimal code.... Al and I—some of the work was done with Jeff Ullman—did some really interesting work. The net result for me was a much better understanding of code generation.

Meanwhile, we had gotten more memory for the PDP-11 and this enabled me to take some of Al Snyder's program and put it back together and before long I had rewritten the whole thing. There's a bit of Al's grammar but basically none of his code left. And that became what I called portable C compiler—**pcc**. Intended to be a compiler for the Honeywell. And a little bit later on we thought it would be nice to move it to the IBM, so I did a 360 version, just by hacking the Honeywell compiler. And a bit later still there was a switching machine called a 3A, and I did a compiler for that. In this process, I realized that there was an awful lot of what I was doing for these three machines that was the same; so I went through, over time, and took these three compilers and brought them together and saw that 90% of the code was the same. So, in my head, I built a process for that other 10% ...

lint is an interesting story. When I decided that I was really going to write a portable compiler, to take these three

things I had and make them truly portable, I knew how to do that for the front end, but the code generator was still very much a mystery to me. ... So here I was with a front end that needed to be tested, and no way to test it. So I thought maybe I can do something useful with the front end, and if it does something useful, maybe people will run it on their code and debug it for me. Well, what would be useful? The front end was supposed to be parsing functions ... and I said, well, one common coding error is writing function calls with the wrong number of arguments; in fact, it's not always easy to figure out where functions are. So I wrote a program that would read your files and, first of all, just tell you where the functions were. Then I realized that it could do this checking between calls. So **lint** really was a way for me to debug the front end of the compiler. That explains a lot about it, because it took on the role in a number of organizations of, well, the guardian of public morality.

You know, when we ported Unix for the first time, in 1977, we had a very serious problem in that there was a lot of code that people had written where they'd looked in the manual and copied the structure declaration out of the manual. It was at that point that we introduced the header files, which we conceived of as being different from machine to machine. We wanted to make sure Unix programs all used header files, in particular the same header files the operating system used. So one of the things that **lint** learned to do early on was to check the consistency of header files. ... And this allowed us to go through a large body of code—6th Edition Unix—in a few weeks and get rid of the inconsistencies. And it helped us to form them into the V7 utilities (V7 was the first portable system).

Resisting the impulse to talk about everything in Unix, let me merely go to awk, before turning to communication. Peter Collinson asked Brian Kernighan about awk:

awk dates from, I think, 1977. It's by far the biggest software project that I have ever been involved with. There were three

of us in that, and that's completely unworkable. Somehow, it's much easier working with two rather than three. It's harder to split things. There's more divergence of opinion, sometimes that's good because it means that more things there but sometimes it means that it's not as cohesive as it might be. On the other hand it was very very nice to work with Al Aho and Peter Weinberger so I had no problem with that.

There are three separate threads that came together. Mine is that there was a tool that was done by Mark Rochkind who was at the Labs at that time. This was basically a data validation tool, it took pairs of regular expressions and messages. It scanned through an input file saying if I see something that matches one of these regular expressions I will print these messages. Of course, he wrote this in C that was compiled and made this recognizer. It was *such* a neat idea.

In some sense, that's the basic notion of **awk**. There is a bunch of patterns and there is something to be done with each one of them. It's not that there weren't already programs like that, but this was a crystal clear example of one of them. Also, I had fooled around with programmable editors for a long time. We went through a phase when **qed** was popular. **qed** was an appalling programming language but people used it.

This was a line editor having multiple buffers and multiple files, much like **emacs** today.

I kept thinking that there had to be a better way of manipulating text. Something that would allow you to manipulate both text and numbers was clearly what I wanted.

Al Aho was the world expert on regular expressions. Peter Weinberger had some kind of strong background in things related to databases and was interested in things that felt like databases. Now, *awk* was in no sense a database tool but there is that flavor to it, that comes from Peter's database experience. So, we took this mishmash of ideas and glued them together and built something.

It was meant originally for writing these one and two line programs. It really was. I think it's very seductive because

it does so many things automatically. It handles strings and numbers smoothly. It is an interpreter and there's no baggage, no derived object files. People start to write a one and two line program that just grows and grows; some of them grow unbelievably large: tens of thousands of lines—which is nonsense.

There was always a slow background set of changes to *awk*. At one point I went back to my notion that I really wanted a programming language where I could write programs that would manipulate text as easily as most programming languages can manipulate numbers. I thought that rather than going in the direction that C had gone from B, I would go in the other direction.

I invented a language that I called A, a very simple thing that took the fundamental notions of *awk* and wrapped them up in a programming language. It didn't work very well. It did have functions for example, but it didn't have any of the implicit input or output or any of that kind of stuff. I played around with that but it was never very good. The implementation was really flakey, so I never used it much and nobody else ever used it.

At the moment there are a couple of other implementations of *awk* around. In particular there is the GNU version called *gawk*. It's quite close to our implementation. There was a lot of contact back and forth while they were doing that to try and make sure that the versions stayed in sync as much as possible. There are also versions for DOS, the MKS version is the one I'm most familiar with; again that's quite close to what we have. Finally, there's POSIX. POSIX is standardizing *awk* and that's in draft stage at this point.

That's more or less what we put in the book. Mostly I think that it is. There are some fine points of contention that I suppose are necessary about the standardizing process. There are some things that conflict with the book: things like what state variables are in as they go through conversion from number to string and from string to number. These sort of things are perhaps not spelled out as well as they ought to be.

> POSIX takes one view and we take another on at least a couple of points. For the most part I think everybody agrees on the mainstream stuff.

Mail existed in the very earliest version of Unix, the man page lists the "owner" as **ken**. It enabled ken, dmr, doug, rhm, and a few others to communicate with one another. It was available on other timesharing systems, too. But, as early as the summer of 1940 George Stibitz had used a calculator via remote teletype from Dartmouth in New Hampshire to New Jersey. Wanting to work remotely led logically to wanting to share information with others—both data and programs.

The earliest network was the ARPANET (begun in 1969), intended for resource-sharing: the Advanced Research Projects Agency (ARPA) thought that linking its various sponsored projects would be cheaper than buying expensive computing facilities for all of them. Organizations were to log in to each others' machines and transfer files. File Transfer Protocol (ftp) and remote login (telnet) were implemented early on. And though there had been intra-machine mail, there had been no inter-machine mail previously, as there had been no networked computers. By the early 1970s, computer mail was in relatively common use; by September 1973 it was sufficiently popular that mail headers were standardized into a de facto standard. In the late 1970s, the ARPANET specifications were rewritten to include SMTP (the Simple Mail Transfer Protocol).

The original (1973) model for the message format was the one run on the TENEX operating system, popular on DEC-10s. Many of the successor hosts were DEC-20s, running TOPS-20 or TWENEX as the OS. (TWENEX was developed by BBN as their follow-up to TENEX, which they had developed.)

But what of the Unix user?

Prior to 1976, it was nearly impossible to send files from one machine to another. Toward the end of that year, Mike Lesk of AT&T Research wrote a "scheme for better distribution" (*Mini-System Newsletter*, January 1977), which was referred to as UUCP only a month later. (UNIX-to-UNIX copy —a program that copies Unix files from one machine or system to another.) It was designed for use over 300 baud

lines. uucp was finally published in February 1978. It proved to be one of the most important parts of Unix, because of its wide use. Lesk's invention led to a need for improvements, as the original program couldn't handle the burden placed upon it by its popularity. The next version of uucp was written by Lesk and Dave Nowitz (with contributions by Greg Chesson) and appears in Seventh Edition (October 1978). This version, too, proved inadequate to the tremendous use to which uucp was being put (more systems, more software, more demands), and in April 1983 Martin Levy sent out a plea for a new uucp system. It had become clear that a uucp designed when asynchronous 1200-baud modems were state of the art, was inadequate to meet the needs of a growing Unix network.

In mid-April 1983 there was a meeting of a number of people at AT&T Bell Labs and the result was the coding of a new version of uucp by Peter Honeyman (login honey), Dave (dan) Nowitz, and Brian Redman (ber). The result is widely known as HoneyDanBer, or (to users of System V) BasicNetworkingUtilities 1.

I spoke to Brian Redman about his work:

> I graduated with a masters from the University of Arizona and moved to the Labs in June, 1978. I had gotten my bachelors in 1977 in microbiology, and my wife had another year to go. But the guy I was supposed to be working for left me without any funding. So I went to another guy in biochemistry and got a job minding his computers. I'd taken some computer courses in the Computer Science Department, which was pretty new at Arizona. And I realized that I could take a master's in Computer Science during that year. Anyway, I was in my coursework and David Hansen, who'd been sort of my mentor, brought in some people from the Labs to conduct interviews. He told the class: "If you've got two arms and two legs you can make big bucks at the Labs." I guess he assumed everybody in the class had a brain.
>
> So I interviewed—originally I interviewed for a job at Western Electric in Princeton, and I would have been working on a DEC-10—I was very impressed. I interviewed with Bell Labs at Whippany and at Holmdel and I got a job in

Whippany. So as soon as my wife graduated we moved to New Jersey.

So I was working in this department of loop transmission at Whippany.... I couldn't believe that I could get paid to do this computer stuff.

My first exposure to Unix was in my operating systems class, where I read the paper. We had Unix in the Computer Science Department, but I was using the DEC-10 and running TOPS. So my first real exposure was when I came to Bell Labs. I did work in Ratfor. The *Software Tools* book was a real enlightenment to me. I really appreciated that the concepts were simplicity and modularity. Reuseability.

But I came to Bell Labs. And they said "You're going to be using this Unix system and here's this program called **learn**." And it would provide you with some information and ask you some questions. I can remember typing in my responses and it would say "no, wrong." But I was sure I was right, so I left the program and started searching for where the program was. I found the program and its database and the answers, and I started to get into it and understand how things work. [**learn** *was initiated by Lesk and written by Kernighan and Lesk. It appeared in V7 and was incorporated into the various BSD releases, but not into System III nor System V.*]

You know, if it was a book, I'd look at the back of the book; it was a computer, so I looked into its file system.

Well, my boss, Doug Cory, had been one of the earliest users of Unix for telephony, a system called COSMOS, for laying out wiring, later called COSMIX. They had an 11/70 in COSMOS; in fact, I have the front panel of this machine—serial number 5001—and I'm told it was the first 11/70 delivered in the US. Anyway, we were running on an 11/45 named wh5ess, and we used mail, and Sam Arnold was putting up this new tool from Lesk, **uucp**. I was interested in that and became the administrator because we had more and more users. And I discovered I could get stuff from Murray Hill using it. And at some point we swapped the 11/45 for an 11/70 so we'd have a more powerful machine.

The first VAX we got was mothra. Well, we were running 300 baud and we got new modems so we could run at 1200 and then glass terminals and there was this big machine, so I asked why we couldn't run faster. And they told me that you could only run RS-232 for 50 feet, and it was 150. So I said to myself, well, so I don't get 9600, I get less, it's still more than 1200. So I ran a wire. And I got 9600; and I got 19,200. And my officemate said, "Hey. Run me a wire." So then I made it a project: I put in a patch panel downstairs and made it state-of-the-art technology. It was fun. By this time, I had hundreds of users, and I took a whole weekend and made everything look beautiful. And we started getting noise on the lines and interference—because everything was running in parallel. So I took it all apart and made it a rat's nest again and everything worked fine.

I started working with the Comp. Center, they had a typesetter, and got into documentation. I put the man pages through the photo typesetter. I was trying to make the system more usable. [*Redman's work led directly to the USENIX Association's publication of the manuals, several years later.*] Anyway, I went to my first USENIX Conference in Santa Monica in '79. And I heard about stuff I wanted to get and met people. And I wanted to get stuff from Whippany over to the Comp. Center. So I got involved with **uucp**. I met Peter [Honeyman] in Delaware [June 1980]. That's the first time I spent any time with him....

Well, I'd go to conferences and meet people and we'd exchange telephone numbers, because we had no way of getting mail to each other. Clem Cole was one person I remember. He had something. I don't know what it was, but it was something I wanted. And we had eight modems—that wasn't common in those days— most folks had one or two. So I would call Clem, he was at Tektronix, and say "Let's set up a connection." And we'd do that. Or Berkeley, they were sending out the second distribution. I recall they said, "We can send you a tape." And I said, "I can't wait." So we set up a **uucp** link.

Clem Cole had been a student at Carnegie Mellon University, where he and Ted Kowalski moved Unix V6 over from the computer science PDP-11 to the Electrical Engineering department's 11/34. (As Cole remarked to me, this was "no mean feat, since the CMU system was heavily hacked for the 'e' machines, and 'stock' Version 6 could not support the 34s.")

Cole then went to work at the Mellon Institute, where he and Dan Klein brought up Unix and, when the Institute bought "one of the first RK07's (serial #17 as I recall), Dan and I wrote its first driver together one week. The RK07 gave us 20 Megs!— practically an infinity!"

Cole got his degree from CMU and a job at Tektronix in Oregon.

> I got there in May 1979. A few months before I got there, Steve Glaser had found a 11/60 running RSX, convinced Tek to purchase a $20K V6 license and started to bring Unix up on it. Since the 60 was not really an 11/40 (close but not quite), my first weekend in Oregon was spent working with Steve tearing down the 60 and putting it back together with a bunch of CMU mods. ...
>
> My first Christmas in Oregon was spent installing a PDP 11/70 Rick LeFaivre, Steve and I got our hands on. That machine became the second major uucp site—"teklabs" running V7—we talked to the "Marxs" machines daily. [Cole is referring to chico and harpo at BTL in New Jersey; the June 1, 1981 USENET map shows the teklabs - chico link.]
>
> We wrote the first TCP for VMS. This still lives today as the "CMU-Tek" TCP. We used the 70 to help build "Magnolia," a 68000, multiprocessor, Unix workstation, which ran Steve's and my Unix-based OS—Magix. And we had a blast!
>
> Brian Redman and I would communicate daily over uucp on important matters—like getting "adventure" to compile (in C). Actually, I suspect we were sharing microprocessor development tools and other less interesting matters. Certainly we were sharing 11/70 changes, and supercharging. We had a common enemy—Digital—which did not want to believe Unix was something they should help happen.

Redman continued with his story:

> I found myself remaining in contact and sending electronic mail on a regular basis. So we had a community. And that's how NETNEWS came along. [*Usenet originated as an idea of Jim Ellis and Tom Truscott when they were graduate students at Duke University in 1979. The first implementation of Usenet was written by Steve Bellovin, a graduate student at the University of North Carolina. Thus USENET began in 1979 as an exchange of information between the University of North Carolina (**unc**) and Duke University (**duke**). A third host **phs** was added in 1980. Armando Stettner and Bill Shannon arranged for the first east-west connection (**duke** to **ucbvax**—with a connection to **decvax** in the same year. USENET is now a huge, decentralized association of systems with over 5,000 groups and millions of users.*]
>
> I put up NETNEWS immediately on chico. I remember spending a lot of time phrasing an intelligent question so I'd get an intelligent answer. It all came about because of impatience. Even with good intentions, when someone says "I'll make a tape for you" and "I'll mail you a tape"... It's not good enough.
>
> And that's when it got started: when NETNEWS began, **uucp** broke. Lesk just hadn't considered that what **uucp** was now used for. Nor for that kind of load. So my problem was that NETNEWS created a lot of files. And NETNEWS would come over and when all the files were transferred Unix would issue the **uuxqt** command but when the volume got high, the connection would die at some point. And before it got back, there would be another **uucp** connection and hundreds more files would come over. And when you went to look in the directory, it would time out because there was so much to look through. So the first significant contribution I made to **uucp** was to have a different directory for each site I talked to.
>
> Anyway, I did that. And Honeyman sent out mail and said, "Let's get together and talk about this UUCP thing. Let's put all the changes together and supply them to everybody so we'll all be running the same thing." So we sent out a note,

> and we had a meeting at Murray Hill. [Mark] Horton was there—he was at Columbus by then, Honeyman, myself, Bellovin, everyone at the Labs who was running systems. Honeyman's got the attendance list, I think. So basically, Honeyman and I and Dave Nowitz we'd do the work of putting the changes together. Steve Bellovin, it turned out, made significant contributions, too. And we started hacking that night. It turned out that the work was partitioned quite well: I did the directory stuff; Honeyman was interested in the dialer stuff; and Nowitz was interested in security issues. So we worked separately and we worked together and we did it.

Bellovin revised his code for *NETNEWS*. It was revised by Steve Daniel and then Truscott. The result was **A News**. In 1981, Mark Horton (a graduate student at UC Berkeley) and Matt Glickman (a high school student) rewrote **A News** into **B news**. In 1987, Henry Spencer and Geoff Collyer of the University of Toronto produced an alternative, **C News**. There are a number of other useful modifications to *NETNEWS*, by Spencer Thomas, Rick Adams, Ray Essick, Rob Kolstad, and many more. But let me return to *UUCP*.

Thanks to Peter Honeyman, here's part of the minutes for the 1983 meeting (in Peter's inimitable orthography):

> The first (and last) meeting of the uucp-lovers interest group took place on 20 April 1983. Invited were allegra!honey, vax135!martin, eagle!karn, rabbit!ark, mhb5b!smb, harpo!ber, cbosgd!mark, floyd!trb, research!rtm, and eagle!dan. Others who showed up were representatives from USG, mhtsa!lsc and mhtsa!brad, whuxlb!pep and gummo!mmp. The unexpurgated minutes follow.
>
> From: honey Wed Apr 20 00:46:26 1983
> To: uucplovers
> Subject: minutes
> Cc: dmr doug
>
> pardon my editorial asides in what follows, the minutes of a meeting called by levy to discuss uucp concerns. attendants at today's gathering of the clan were

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

the first topic of discussion was the corporate electronic mail project and its sidekick, network action central. action appears to be jointly administered through 452 (staff) and 774 (equipment and organizational support). murakami lent the impression that this was the beginning of a permanent fixture w/in the labs and that any startup problems are temporary. the L.sys database is gradually falling together as new data are collected and standards are promulgated. discussion about the means for handling the monthly L.sys produced a consensus that sites should continue to maintain a local L.sys and that the local file should be searched first, a la koenig, honeyman, and apparently everyone.

a lively discussion on uucp hacking consumed the major part of the meeting. there are as many versions of uucp as there are sites, with the major contenders being usg 6.0, the new code by morris [Robert T. Morris], and tom truscott's hacks. cohen grimly cautioned on the difficulty of getting good stuff into 6.0, nonetheless, the company flag was raised, all saluted, and we agreed to use the organizational heavy as a starting point for producing a version that would satisfy all (and ourselves, in particular).

the major enhancement proposed was a hashing scheme for the spool directory. while truscott uses separate directories for the C and D files (as well as other files, i imagine), redman and others use the names of the remote sites as the subdirectories. lively discussion ensued, during which a consensus was reached that the latter scheme was more robust. a proposal by levy that the hash function be based on a short prefix of the machine name was met with interest. while there appears to be a problem with mkdir by setuid processes in usg, bellovin patiently explained techniques to subvert this feature; after the third or fourth go 'round, the meeting got back on track. the feeling is that cico or some daemon should do occasional rmdir's in the spooling directory; ultimately there was not total agreement on all of the mechanics of the hashing scheme, and i imagine the

issue will be discussed further in a manner fit for the information age.

...

finally, we agreed to take the most recent version of 6.0 uucp and hack it into an unrecognizable state (i.e. remove the bugs, and make it work well). honeyman commandeered conn.c, redman got the spooling hacks, levy was volunteered to write an L.sys editor and help murakami standardize the action L.sys, and bellovin accepted the responsibility for putting the spool directory mods into uuxqt. cohen pointed out that usg is unlikely to appreciate anything that falls under the ucb umbrella, while nowitz argued against reliance on ifdefs. the feeling was that we can hide the idiosyncrasies in a single file and mask our efforts as a gesture toward version 7 compatibility, a transparent but innocent falsehood. in any case, this issue is peripheral.

the final topic on the agenda was the corporate attempt at security consciousness raising; a shouting match ensued, in the course of which several and various reputations were sullied, certain paranoid reactions were taken less than seriously, and no great meeting of the minds was met. honeyman argued in favor of mcilroy's position against allowing remote machines to pull files, falling back on a position that denied this capacity to all but a local cluster, falling further back on a suggestion that permits files to be pulled from a directory other than uucppublic (so that one remote site can't pull a file that another remote site has sent). redman pointed out that the only information worth protecting in L.sys is the phone number (certainly the least secure piece of data). redman solicits opinions on the forthcoming GEI regulations.

upon notable growlings of empty stomachs, a decision was made to adjourn to a local chinese restaurant, in the time honored tradition of uucp hackers everywhere. this decision was subsequently rescinded in favor of the oak room (constituting an unpropitious omen).

The Oak Room is the cafeteria at Bell Labs in Murray Hill; prior to a few years ago, it had actual table service.

Prior to this lengthy discussion of uucp, I mentioned the ARPANET and TCP, as well as ping. As Mike Muuss was involved with the earliest versions of TCP and is the author of ping, I consider his tale worthwhile.

> I was an undergradate at Hopkins. We had a fairly unique situation: the PDP11/45 was running RSTS [DEC's Resource Sharing Timesharing System] when I showed up, it was only running Unix a couple of hours a day. And the department said we couldn't run Unix unless we could also run RSTS. ... So we did that, and it became our first software product—running RSTS under Unix. It was late '75 when that really started working. We went over to Version 6 in the fall. That had consequences, too. I sent a copy to the guy in charge of the Cray-2 Unix effort...

Muuss graduated from Johns Hopkins in 1979 and went to work for the Ballistics Research Lab. In September of that year he worked on implementing the prototype BRLNET high-speed local network that he had designed earlier that year under a US Army contract. In early 1980, Muuss extended the BRLNET protocols, and led a team to port the University of Illinois NCP capability to PDP-11 Unix. The team installed an 11/34. In late 1981, Muuss identified the experimental TCP/IP suite and began implementing it for BRL/JHU Unix on the PDP-11, rather than continuing the extension of BRLNET. Perhaps more important, Muuss began an electronic publication called *The TCP/IP Digest*, with a circulation of over 700 subscribers via ARPANET and USENET. And most important, much of Muuss' work was incorporated into the two MIL (military) standards: 1777 and 1778. Nearly every current TCP/IP implementation includes protocol software developed by Muuss at BRL. Muuss told me:

> We even got into the gateway business for a while. Ron Natalie was my main developer there. We built a little operating system and gatewayed all sorts of things to the ARPANET from the PDP-11. So we were in the gateway business for a while. Then, in 1983, we took the next big step and pushed for a big fiber-optic grant. AT&T had just come out with rib-

> bon-cable style fiber optics.... We bought main trunk cable with 145 fibers in it. We set up a local area net immediately and went for true fiber later.

That was a decade ago. In 1984 the BRL network was unparalleled, according to press reports. The TCP/IP protocols went to both BBN and Berkeley. I'll come back to development of the protocols in Chapter 18.

Unix Spreads and Blossoms

CHAPTER

15

The Users—Part II

Among the most fascinating aspects of the Unix story, to me at least, is the speed with which it proliferated into Canada and Europe, Australia and Japan. As a true trace would ultimately prove tedious, I have selected a few countries and will allow the actual participants to tell the tale. I have mentioned some of these users earlier, but now you can listen—as it were—to their own voices.

First, here's Professor George Coulouris of Queen Mary College (now Queen Mary and Westfield College). I asked him how QMC had gotten V4 in late 1973.

> William Newman was a Visiting Research Fellow in our Lab for a little over a year from about October 1971 (he then moved to Xerox PARC, and is now at Rank Xerox EuroPARC in Cambridge). We had discovered the value of interactive minicomputer systems with his guidance. He and Mike Cole developed a single-user interactive operating system (called MIFS) at QMC. MIFS ran on an Interdata computer and was loosely modelled on TENEX. It was developed to support two research projects, one by William to investigate the design of a graphical programming language and one by a group of us to investigate the design of a new machine architecture, using microprogrammed emulation. Like Unix, MIFS included a wide range of components and tools—a filing system, a systems programming language, a compiler-compiler, a line edi-

tor (with single-character interaction), a debugger, etc. Nearly all of these were developed by William and Mike.

We hadn't heard of Unix while MIFS was under development, but while William was still with us he made a visit to Bell Labs. When he came back he said "they seem to have developed something like MIFS at Murray Hill, but it supports multiple users and runs on PDP-11s." Mike Cole obtained some documentation, studied it, and concluded that it could indeed bring us great benefits—it offered most of the facilities that we had found so useful in MIFS and went well beyond it.

We didn't have a 9-track drive, just DECtape and a System Industries 10 Meg exchangeable drive. The real problem was that Ken didn't have a driver for the SI controller (nor did anyone else in the world). Obviously, we couldn't develop a new driver using a Unix system with DECtape as the only peripheral—the swaps would have rendered the process horrendously slow! Mike Cole was our system guru, and he solved the problem. He picked up the V4 release from Ken on DECtape in the Summer of '73, on his way to spend a couple of months visiting Xerox PARC (William Newman was there by that time, and Mike went there to work with him). While at PARC he went to Bezerkeley, wrote and compiled a simple driver for the SI controller and built a kernel that included it. But Bezerkeley didn't have any SI disks, so he couldn't test it there! When he got back to QMC the DECtape was duly mounted, and hey presto!—fortunately Mike was a very good programmer.

It's surprising that (a) we had the temerity to buy a machine with disks that weren't compatible with the Bell Labs system when we were intending to run Unix, and (b) Mike managed to write a disk driver that worked first time, having never written a Unix driver, never programmed in C, and never used Unix.

Mike Cole added:

George's account is largely correct, but there are a few points that unfortunately diminish my apparently legendary pro-

gramming skills. My memory is a little hazy in places, but I think the following was the tale of summer '73.

I didn't pick up the tape from Bell Labs on my way to the West Coast. Rather, while at PARC I spent a few afternoons in Mike Stonebraker's group at Berkeley and obtained a copy of Unix and the documentation from them (having obtained the requisite AT&T dispensations) in the process. I then set about learning sufficient C to write the SI disk driver and build a system using their 11/45 with LA30 DECwriters (a painful experience). As I recall they were in the throes of developing Ingres at that time.

Since it was so critical that the system work first time when I got back to QMC I decided to stop over at Bell Labs on my way back to the UK in October Ken Thompson checked over my code, found a bug or two (almost certainly to do with using pointers in C incorrectly) and dumped the entire Unix software onto 7 or 8 DECtapes. Unix then entered the UK for the first time in a PanAm bag!

Being the first site in the UK I remember being invited to both Edinburgh and Cambridge in '74 to present what we were doing with this odd-ball operating system called Unix. I was struck by the very negative response, particularly from Neil Wiseman's and Charles Lang's people at Cambridge who just wouldn't contemplate adopting a system for which there was no support from the supplier (as I recall they were using RSX-11 at the time). Such are the disappointments of the early evangelists!

Just a bit later, Nigel Martin was a student at QMC, and when Sunil Das at City University obtained a Western Electric license, there was no need to wait for New Jersey to work out what to do about shipping to England: Sunil got V5+ from Nigel.

In the meantime, the University of Toronto had obtained Unix. Before detailing any of the history, it is of importance to recognize the strong Canadian connection of Unix. Brian Kernighan was an undergraduate at Toronto; Al Aho received his degrees there, too; Heinz Lycklama received his education at McMaster in Hamilton, Ontario;

Morven Gentleman attended McGill (Montreal) and was on the faculty at Waterloo (Ontario) for 15 years; Tom Duff, an undergraduate at Waterloo, went on to join Rob Pike, David Tilbrook, Howard Trickey, and Mike Tilson at the University of Toronto; Tom Cargill earned his doctorate at Waterloo. Many others could be named. (Even I was a faculty member at the University of Toronto for 11 years.)

The University of Toronto purchased a PDP 11/45 in mid-1974. It ran Version 5. On October 1, it received a license for V6. Other universities weren't so patient. Tom Duff told me:

> It was 1974 at the University of Waterloo. Steve Johnson was on sabbatical (I was an undergraduate there at the same time) and phoned in to the Unix machine from a lab. Somebody was watching over his shoulder, stole the number and called up later to grab the system source code.

Doug McIlroy told me that the mythology has it wrong: it was he from whom the Labs' phone number was filched. "I gave a talk at Waterloo, and the Unix phone number showed up on the bill at the student computer club. By the time the faculty found out and called me in despair, our code and theirs had been inextricably confounded."

The next year, Duff was a graduate student at Toronto. (Rob Pike was the summer apprentice in 1975; Dave Galloway was the summer 1976 apprentice.) Duff said:

> The 45 arrived in September and started running 5th Edition in December. I remember it being some time before 6th Edition was running ... early '75 could be right. I was doing my thesis before the advent of split I&D, which came with 6th Edition. [Mike] Tilson, [Bill] Reeves and I were the crew that cold-loaded the 5th Edition tape.
>
> We had some sort of problem and phoned up Ken to get help, although he'd obviously had more calls than he'd have liked. The 20-or-so 5th Edition sites was a much larger distribution than he'd had to deal with before. The **mkfs** instructions indicated how to make a root filesystem on an RK05 with 4000 blocks, leaving 872 blocks (almost half a megabyte!) for swap space. Bill typed the same formula to make a filesystem on our second RK, thinking that the few

> wasted kilobytes weren't worth the effort of sorting through the RK05 documentation to find out exactly how big the thing was. It was only a week later that we hit the wall and I spent a couple of hours reading the -s code in **check** to see that I could in fact patch the filesystem size in the superblock and run **check** -s to reclaim those blocks.

Duff also got into trouble for using the departmental Xerox machine to produce the yacc manual—in those days the bureaucracy ruled the power to copy.

Mike Tilson amended Duff's version.

> The first system was a PDP-11/40, an interim machine while we awaited delivery of the 11/45. I believe that we were able to test the claim that Unix would boot with only 64KB of memory! The tape was personally produced by Ken, who gave it to me at Bell Labs. (Then, as now, AT&T was slow to ship. But unlike now, you could bypass the paperwork and get the tape yourself.)
>
> The 11/40 had a GT-40 display: the driver for it was one of the first "custom hacks" at U of T.

On Friday, February 11, 1977, a fire broke out in the northeast corner of the Sanford Fleming Engineering Building on the Toronto campus. Despite the best efforts of the Toronto Fire Department, the building was nearly gutted, and the remainder (except the facade, which forms the front of the new building) was razed. The 11/45 ran through the night, according to Tilson, until

> around 9:00 am, which was when the fire department finally figured out the maze of wiring in the building and managed to cut the power. Another note: one week prior to the fire, there was a discussion with the lab manager about off-site backups. It was concluded that this was an alarmist notion, probably not needed. When we got back after the fire, one of the first orders of business was the institution of an off-site backup program.

On November 1, 1977 Toronto received a license for Phototypesetter V7. I'll return to Toronto later. Right now, let's go to Europe.

Teus Hagen's narration of Unix in the Netherlands began in Chapter 10. Hagen believes that "the technology in Unix (string expressions, yacc, a line editor, text processing, awk, byte streams philosophy, pipes) of those early days pushed computer science in Europe forward a lot. Within a very few months results here in Europe jumped ahead." He also told me:

> I think we (Mathematical Center, Amsterdam) received Unix from the Lab about the same time that Queen Mary College London got it. It took about a year before others received Version 6. I've still the original manuals in my possession.
>
> The University of Nijmegen (Hendrik Jan Thomassen) installed Unix next and, after a lot of discussion, Andy Tanenbaum of the Free University obtained it (Andy was busy with his own OS at the time).
>
> One of the problems for us was that the price of computer equipment was so high: a magnetic tape unit cost about $45K, a diskdrive (DEC RM05) cost $55K (one of ours tried to fly around the computer room on one occasion).
>
> We had trouble with the DEC maintenance people as they required RSX or VMS to be running, or they could not fix our hardware problems. However, as our error messages were far better under Unix, we were able to convince them eventually. But how can you install VMS or RSX on a defective computer system?

Hagen wasn't the only one having trouble with DEC, Lou Katz told me: "Basically, DEC servicemen would run their diagnostic programs and then say it was our software, since their diagnostics worked. In particular, it turned out that no DEC diagnostic tested the PDP-11 Supervisor mode, but Unix used it." Armando Stettner said: "There were many things that Unix did on PDP-11s that neither DEC OS nor diagnostics did or tested."

> DEC could not believe that everything was written in C (what was C?) and available in a listing of about 500 pages. And that you could change source if you wanted to. This compared to a pile of manuals and papers from DEC about the height of the computer room.

> Initially, we had a PDP-11/45 (with core memory; a good heating device!) but with only one RK05 1.5 Mb diskdrive as exchange medium. We were unsure what the effect of the public transport vehicles on the magnetic field of the platters would be (we have a lot of trams in Amsterdam). So we did not take the risk and would walk from the Math Center to the Free University (about 7 miles). Later we had more faith, and Unix people could be recognised by the fact that they would be carrying huge boxes containing RK05 cartridges on the trams.

It was Hagen who started the first Unix network in Europe and began the formalization of the European Unix User Group. Starting a network in Europe was more complex than UUCP and ARPANET had been in the US. In his words:

> The use of dialers and modems was not permitted by the European PTTs, so we had a lot of trouble setting up the Unix network. The PTT guys could not imagine what UUCP functionality was at all (we spoke a different language). So we used an office desktop semi-automatic dialer and built that into a box (separate from the 110/300 baud modem). A small firm built more of them and smuggled them as radio amplifiers across the borders. My good friend at DEC, Armando Stettner, helped me to set up the link to the US (only mail, as news did not exist then). Compression was unknown, so we spent quite some money because of the slow speed. This is the reason why there is such strict accounting system on the net in Europe still today. Because tariffs are much lower at night we shifted the US communication time to the night period. One morning I discovered the modem line to the US hanging: exchanging error and restart messages. It had been going on for about 10 hours. A 10-hour phone call to the US was real expensive, so I was called to the director's office (who did not know much about it as he was mathematician and did not want to deal with computer stuff at all). I got into a lot of trouble. ...
>
> It was several years before we began getting news. I recall my first articles on the net, because people in the US would

> yell at me, asking how the hell it could be that I was replying to articles hours and hours before they had written them! They had just discovered that there was a world outside the US.
>
> [I asked Teus about Kremvax]
>
> It was April Fool's Day. We sent an article with a path trace to Moscow from the Math Center that Kremlin VAX was announcing its appearence on the net. Besides the to-do it caused on the net, with a lot of people yelling at each other, I was told that a US administration security meeting was agitated by this. Officials tried to stop the communications lines to Europe immediately. But luckily, it was soon discovered that such a step was futile and that the network anarchy would go on.

The actual originator of Kremvax was Piet Beertema; the message was posted April 1, 1984. Six years later, when demos.su joined USENET, Vadim Antonov, the senior programmer at Demos, had to convince folks that it wasn't just another prank. Armando Stettner's story is only slightly different from Hagen's. He told me:

> We [DEC] were doing UUCP with a lot of places. But we weren't doing **NETNEWS**. And Teus and I were at a conference, talking about the idea of doing **NETNEWS**. And I said "Look, what's the issue about doing **NETNEWS** across the Atlantic? It's connect time. If you put compression into your end and mine, I'll send you news." And I didn't know what would happen. But Teus sent me mail later and said "Copy this stuff off our machine." And we got it up and running.... Shortly thereafter I started exchanging **NETNEWS** with Europe, through Teus' machine. After a successful few weeks, I brought Australia into the net through Robert Elz' machine.

At one point, decvax, Stettner's machine, was running a phone bill of nearly $250,000 per month. One of the DEC accountants asked him how, with only four lines, they could be running such a bill. "Oh," responded Stettner, "that's our high-end machines exchanging information."

Stettner's sending news to Australia wasn't really as strange as it sounds. John Lions, at the University of New South Wales, read the article in *CACM* in the summer of 1974, when UNSW was about to get a PDP-11/40, and the university negotiated a license with Western Electric before the end of the year. Here's Lions' narrative:

> In 1974 the UNSW ... replaced its 1965-vintage IBM 360/50 computer with a CDC Cyber 72-26 running under KRONOS. The Cyber ... served about 100 teletype-like (300 baud) interactive terminals and twelve PDP-11/40 computers ... While seven of the PDP 11/40s were purchased with minimum configuration (16Kbytes of memory), five systems were configured with 128Kbytes of core memory, a DJ11 terminal multiplexor and three RK05 disk units each...
>
> The stage was thus set for the arrival of our copy of the Unix software. Just two evenings of experimentation were sufficient to show that we had stumbled on something rather interesting and that the Unix system would have substantial advantages over RSX-11D ... Our problem became "How to arrange to run Unix during the day as well as the evening when the remote-batch service was not required?" The answer obviously was to write a U-200 emulator that would run under the Unix system. A group of four enthusiasts led by Ian Johnstone accepted the challenge and were able to develop a prototype system in two weeks. Four weeks later RSX-11D was displaced, never to return.

Greg Rose was an undergraduate a UNSW in 1974. I chatted with him about the history.

> I was the rawest of undergraduates. And the university was in a state of flux. It had been an IBM shop ... the 360/50 was being retired. Actually, it was being relieved of teaching and research and turned over completely to the administration—after all, they didn't want to rewrite all their PL/I programs. The idea was to go with the Cyber... So the university bought five 11/40s and seven 11/10s. The Cyber was to come into use in January 1975 and halfway through 1974 these two interest-

ing papers were published: the one on Unix, the other on Pascal. So John Lions ran off and obtained Pascal, and Ken Robinson got Unix....

Teaching was something that had to be done. Well, one of the requirements in the second year was assembler programming and the assembler had to run on the PDP-11. So there was this incredible timetable so that we could have a batch assembler for the PDP system. And we had to support student assignments and debugging or we weren't going to be able to run Unix on the machine. So the bunch—Ian Johnstone, Greg James, Ian Hayes, Peter Ivanov, and John Wainwright—decided to run Unix on this machine and so they had to write all these U-200 device drivers and emulators so that the department could continue with RSX-11 while the machine ran Unix.... But someone had to maintain it and make sure it worked. So Ken Robinson said "We've got these two students who aren't being challenged, let's give it to them." And I was one and Chris Maltby was the other.

My first assignment was to write the support for batch processing for the course in assembler in which I was a student!

In '75-'76 Unix was a big success on campus. We had an 11/70 and another 11/45 and the new School of Management decided they needed Unix. Then in '76, the University of Wollongong was created. Originally it had been a college of UNSW, but it was separated. And at that time Interdata was virtually giving away machines, in order to break into the market, and they made Wollongong an offer they couldn't refuse. Well, we used to have these informal meetings, where about ten of us would sit about and drink coffee. And at one of them Rich Miller said "You know we've got this computer with a simply awful operating system. But we don't see why, as Unix is written in a relatively high-level language, we couldn't make it work."

So we sat down for a couple of days, mostly me and Ian Johnstone, talking about the architecture and what we had to do with the compiler to produce code. And Rich went away

> for a while—he was an amazing guy—I think he's now at Oxford—and he came back and said "I think I've got a compiler." So he and Ian went away and cross-compiled the compiler and took it back to Wollongong. We would drive back and forth—a hundred kilometers each way—with 9-track tapes. And when we got to Wollongong, because they didn't have a tape-drive on the Interdata, we would read them on a Univac 11-something and write disk-packs which were compatible with the Interdata.
>
> But it was really only three trips: once was for the compiler; one was the first cut at the operating system; and the last one was for the upgrade, because we'd begun with Version 5, and when we got Version 6, Rich said "Well, now give me an up-to-date source tape."
>
> At the same time Steve Johnson and Dennis were doing their port.

It was a truly remarkable effort, as Johnson recalled. Rich Miller talked about it at the Second Australian Unix Users Group in February 1977.

In the March 1976 issue of *UNIX NEWS*, there appeared a long letter, dated 10 October 1975, from John Lions to Mel Ferentz. The letter began:

> We have only recently seen copies of your first two UNIX Newsletters and hence have not been in a position to contribute earlier. However, we do appreciate your effort in starting the Newsletter and hope it will be successful.

Lions wrote Ferentz again on 21 June 1976 to announce the first "East Coast" Unix Users Meeting at UNSW on August 29, 1976, and to let the users know that both the University of Newcastle and the University of Sydney had become part of the Unix community.

On 17 September, Lions wrote to report on the meeting: "more than 30 persons gathered." The meeting was so successful that another was scheduled for 18 February, 1977. More importantly,

> It was agreed that there is a real need for cooperation between Unix users in view of the unconventional nature of Unix support.

In 1977, John Lions completed his *Commentary on the UNIX Operating System*, companion volume to his reproduction of the V6 source code. These two slim books may be the most important computer items never to be published. I asked him about the *Source Code* and the *Commentary*. He said:

> I was teaching Operating Systems; I was competing with a colleague who was teaching Compilers by getting students to write real compilers; so a code **reading** exercise seemed a good idea. Also **our** Unix license wasn't explicit enough to forbid the activity. Why Unix? There wasn't much choice. It was highly competent as you know, and it was much better than the competition (we had also acquired Brinch Hansen's SOLO system).

The March 1977 *UNIX NEWS* (vol. 2, no. 3) announced the availability of the book, to licencees, together with a note by Ferentz: Ken Thompson has seen the first version of the book and reports that it is a good job. The price, including airmail, was $A17.70 (under $20 US, at that time). The UKUUG newsletter announced the availability of the code and commentary, but the next issue stated that future orders should be placed with Bell Laboratories and by 1978 the volumes were no longer available. They must be the most frequently photocopied books in the entire area of computer science. They carry the appropriate copyright notices and the restriction to licensees, but once again, there was no way that Western Electric could stem the circulation of something of such value. I confess to possessing both a many-generation photocopy and a copy, in the bright orange and red covers, inscribed to me by John Lions. Mike Cole asked me whether I had

> established that Jim Curry brought Unix into Europe for the first time, installing it at IIASA in Austria. As I recall, Jim left PARC shortly after William went there. Jim visited us at QMC in the winter of '72/'73 and helped to convince me that we weren't entirely crazy to base our new teaching system on Unix.

Curry was (and still is) at the International Institute for Applied Systems and Analysis in Laxenburg, just south of Vienna. He was Head of Computer Services there from 1974–78; he told me:

> I installed UNIX (Version 5) on our PDP-11/45 in January 1975. As far as I know, this was the first UNIX installation in Europe.
>
> In mid-1978, I recruited Jim Kulp to take over from me as Head of CS, and didn't have much to do directly with the CS Department from then until 1982, so I'm somewhat hazy on details over that period. But I do know that Jim bought our VAX/780 in early 1979. I'm not sure whether this was Austria's first VAX or not, but it was certainly the first one running UNIX; Jim was initially running a very early BSD system—probably 3BSD, or perhaps a prerelease of 4BSD (before it became 4.0BSD).
>
> As far as net access goes, IIASA has no special place in history. We were fooling around with various ad-hoc telecommunications linkups (mainly to Eastern Europe) as early as 1975, but our first real access to international mail was provided by a 2400Baud leased UUCP line to tuvie [the Technical University in Vienna]. I don't recall exactly when that was, but since it post-dates tuvie's facilities, I imagine that it's not of interest.

Herb Hasler (of IIASA) said, "I believe IIASA was the first to have a VAX in Austria." More importantly, Jim Kulp at IIASA wrote the first version of Unix job control, which was included in 4.1BSD (1980).

Yet again, the work of a remote user was incorporated into a major release.

As is clear from the correspondence of John Lions with Mel Ferentz, much of the communication among the far-flung Unix fanatics—for so they were seen by the advocates of computer centers—was by means of *UNIX NEWS* and its successor *;login:*. For instance, the reluctance of AT&T/Western Electric's lawyers to come to grips with what they might or might not do resulted in the announcement (30 April 1976) that Lew Law at the Harvard Science Center

> is willing to undertake the task of reproducing and distributing the manuals for UNIX. ... The "UNIX PROGRAMMER'S MANUAL" Sixth Edition dated May 1975 will be reproduced in its entirety. Most installations will want to remove several pages which most users need not know about.
>
> "DOCUMENTS FOR USE WITH THE UNIX TIME-SHARING SYSTEM" Sixth Edition will be reproduced omitting sections 1 (Setting Up UNIX) and 13 (On the Security of UNIX)....

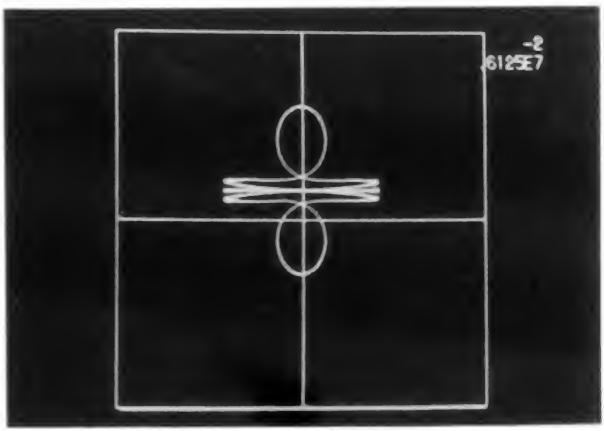
The same issue contained an article by Bill Mayhew (of the Children's Museum in Boston) on "How to fix your PDP-11/40's Static Electricity Problems for 49 cents (Plus Tax)." The next issue (May-June 1976) announced "the first mailing from the Software Exchange." The first software tape contained Harvard software; the duplication and mailing was done by Mike O'Brien, then at the University of Illinois at Chicago Circle. The August 1976 *UNIX NEWS* contained the first contribution from Alan Nemeth, concerning problems with the floating point simulator in separated I&D space. November 1976 saw the announcement of the Second Software Distribution as well as the following note from O'Brien:

> I got the "diff" listing of all changes to Bell UNIX system proper from "standard" version 6 ... Anyway, I've itemized some 50 changes, and sent the list to Ken for verification and comments. The changes will be available through the center by special request....

The second distribution contained contributions from the RAND Corporation, the Naval Postgraduate School, the University of California at San Diego, Yale, and UIUC. *UNIX NEWS* also carried an announcement of the availability of graphical software from the University of Toronto.

A Third Software Distribution was announced in the May-June 1977 issue, which also contained the information that "over 150 people" had attended the May 1977, Urbana meeting and the notice

> Mini-UNIX has been released and LSI-UNIX and Mert will probably follow along at some later date.



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)*

Remember: **No advertising, no support**. The Urbana meeting also featured the first programming contest: put together by Steve Holmgren. All the users had to help themselves.

Among the users who had helped himself was Haruhisa Ishida at the University of Tokyo. "I was the first user and licensee of UNIX in Japan," he told me. "It was in 1976 that I signed the agreement with AT&T Bell Labs. The Unix was version 6, which I implemented on an LSI-11 with only slight modification." The LSI-11 was DEC's own emulation of the PDP-11 on the microchip. It came out in 1974. The earliest version didn't have memory management, which was available on a separate chip. Most likely, it was this version of the LSI-11 that Tokyo purchased, as V6 would not have run without memory management.

In 1980, Koichi Kishida of SRA, an independent software house, bought a VAX and brought Bob Schulman to Japan to install Unix on it. Schulman told me:

> I went over during the summer of 1980 to install first 32V, then BSD 3.0 on a VAX 11/780.
>
> The reason they got the VAX was, of course, that they wanted Unix. The reason they wanted Unix was that Kishida, at the time, was traveling and a sociable kind of guy. He went to lots of conferences in Europe and the US. Around that time, just about every researcher he ran into told him they were using Unix. So he decided that he had to have Unix. Basically, so he could have the same environment as all these hot researchers. I liked his reasoning.
>
> When I got there during the summer of 1980, they had just had the 780 installed, but weren't doing much with it because they needed the 32V tape. Once installed, such a vast investment needed to have a little cosmic help, so they brought in a Shinto priest, and had some kind of blessing ceremony in the machine room. During the ceremony palm fronds were waved around, and the priest spoke in some ancient Japanese dialect (which probably no one understood). Sake was of course offered, though I don't think the 780 drank its share.

> The building was locked up at night for security reasons that were never clear to me. Therefore, if you wanted to work past 11:45 pm or so, you couldn't leave until after 6 am or so. This sort of discouraged late night hacking.
>
> One part of my job description (such as it was) was to teach SRA the "Ways of Unix." I figured that if there were a social aspect to "the Ways of Unix," I needed to try to describe what a "hacker" and a "nerd" were as well. Because of the language and cultural barriers (my Japanese never became very good), it took a long long time before I thought they finally got the idea. Hackers were a very appealing concept to them. Nerds, though, were difficult to understand, and I don't think they ever quite got the same negative reaction to nerd as I was trying to elicit. Probably because they thought the monomania of a nerd wasn't such a bad thing.

The Japan UNIX Society (jus) was set up by Kouichi Kishida, Jun Murai, and Nobuo Saito. I will return to jus later, but now I want to get back to USENIX. *;login:* for September 1977 informed the members that PWB had been released and that the paper about it was available on request from Ted Dolotta. The next issue (volume 2, number 9) stated "The membership in the Users' Group now exceeds 250." The November 1977 *;login:* contained two announcements: one of the death of Joseph F. Ossanna; the other of the impending publication of *The C Programming Language*—

> Need we
> say more?

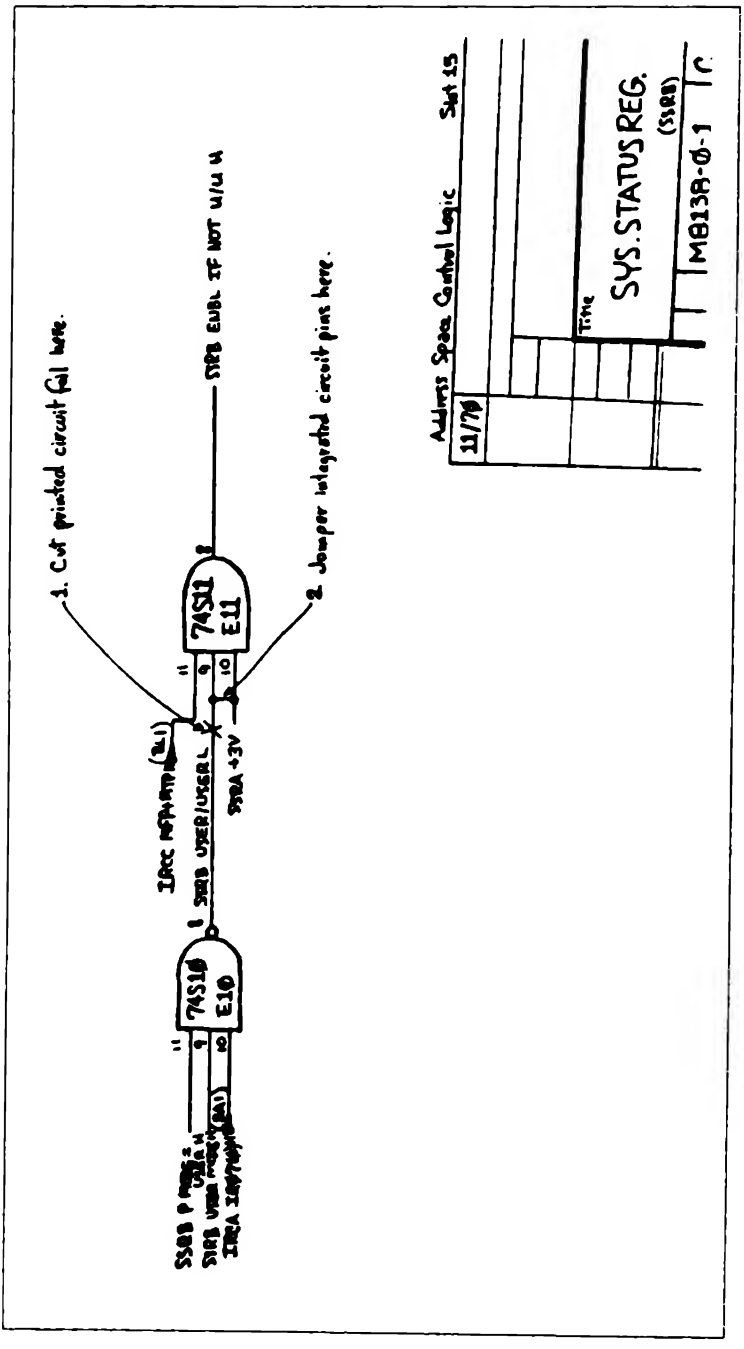
In March 1978, Lew Law announced the availability of the PWB manuals in four volumes (total cost $26.50) from the Harvard Science Center. The following August, Ferentz dedicated the entire front page of *;login:* to the announcement of the special issue of *The Bell System Technical Journal* (Vol. 57, No. 6, Part 2) devoted to Unix.

Bug fixes; hardware fixes; new software; neat hacks. That's what USENIX, the AUUG, and the UKUUG were devoted to. My favorite contribution was one from Tom Ferrin, then as now at the University of California, San Francisco: a hardware solution to a software prob-

lem. His two-page letter (9 December 1977), some code and a diagram appeared in *;login:*. The letter, in part, ran:

> The memory management unit in the PDP-11/45 and 11/70 computers offer several advantages over those found in the other PDP-11 family computers. Among the more powerful features is the ability to separate programs into instruction segments and data segments...
>
> Four PDP-11 instructions facilitate program communication between different addressing modes and instruction/data areas in memory. These are "move to/from previous instruction/data memory space" (mtpi, mfpi, mtpd, mfpd)...
>
> Because of DEC's desire to "...preserve the integrity of proprietary programs", the 'mfpi' instruction does not work correctly when executed with a process status word equal to 17xxxx (i.e. both current and previous modes are USER). This fact prevents the C subroutine 'nargs.s' from operating as intended...
>
> There are several solutions to this deficiency ...
>
> 4. Modify the hardware to work more "correctly".
>
> After several telephone calls to DEC representatives and a few hours of looking at microcode ..., we have arrived at a simple modification to ... the PDP-11/70 cpu to allow the 'mfpi' instruction to function properly. The modification takes about 15 minutes for an experienced person to implement and involves cutting one foil etch and adding one jumper wire to the M8138-YA memory management board...

Tom Ferrin's diagram appears on the next page.


1. Cut printed circuit foil here.
2. Jumper integrated circuit pins here.
74S11
E11
74S10
E10
Address Space Control Logic
Slot 15
11/70
SYS.STATUS REG.
M8138-0-1

CHAPTER

16

Berkeley Unix: Part I

Professor Robert Fabry of the University of California at Berkeley was on the SOSP program in October 1973. He was one of those who was very impressed by Thompson's presentation. When he got back to Cory Hall, Fabry's first job was to try to put together a joint purchase by Computer Science, Statistics, and Mathematics of a PDP-11/45. He then ordered a tape from Thompson, and in January 1974 Keith Standiford (a graduate student) installed Unix. As Kirk McKusick tells the story, Ken Thompson had been involved in all the actual installations up to that time, but not in this one, though "his expertise was soon needed to determine the cause of several strange system crashes." Thompson would call Keith Standiford in the machine room, the phone would be inserted in the 300 baud acoustic modem, and Thompson would "remotely debug crash dumps from New Jersey."

The next problem arose because Math and Statistics wanted to run DEC's RSTS. So slices of each day were allocated: Unix would be run for eight hours, followed by 16 hours of RSTS. One of the undergraduates introduced to Unix at that time was Eric Allman.

> I was taking the introductory OS course and they had been using something called the Toy Operating System on the 6400. But they wanted to get off it onto Unix on the 11/40, where we could only work 8 hours a day, and a different 8 hours each day. I recall having difficulties: I was reading the manual, and I remember not understanding why you would

> ever want the echo command. Totally bizarre. Now, of course, I know. Of course, 4th Edition was pretty flaky. It was a system that only a researcher could love. It was slow. It didn't have a lot of tools. Then I got hired by the Ingres project.

The Ingres database of Professors Michael Stonebraker and Eugene Wong was one of the first projects moved to Unix. As they were dissatisfied with the time allocated, they bought their own PDP-11/40 in spring 1974. But even thereafter, there just wasn't enough time available for students on the 11/45. In June 1974, Professors Stonebraker and Fabry set out to get two instructional 11/45s for Computer Science. The money was obtained early in 1975, just about the time that DEC announced the 11/70, which seemed more suitable. So the money was pooled. The 11/70 arrived in the fall, just when Ken Thompson arrived for a one-year sabbatical as visiting professor. Thompson, Bob Kridle, and Jeff Schriebman brought up Sixth Edition on the newly installed 11/70.

An interesting side-effect of this was the "50 bugs" tape. Thompson told me:

> The first thing to realize is that the outside world ran on releases of Unix (V4, V5, V6, V7) but we did not. Our view was a continuum. V5 was what we had at some point in time and was probably out of date simply by the activity required to put it in shape to export.
>
> After V6, I was preparing to go to Berkeley to teach for a year. I was putting together a system to take. Since it was almost a release, I made a **diff** with V6. On the way to Berkeley, I stopped by Urbana-Champaign to keep an eye on Greg Chesson who was finishing up his Ph.D. (subtle recruiting). I left the diff tape there and told him that I wouldnt mind if it got around. (I think I gave it others too, perhaps Katz.) [The **diff** tape that O'Brien mentioned must have come to him from Chesson.]
>
> I felt that it was important because I had discovered a whole set of very bad multi-programming (asynchronous) bugs that really should be fixed. The versions that were distributed were "as is" with no provisions for updates. There was also some very strong language in the non-disclosure

about the users exchanging such information. The whole situation was unworkable and I felt that I had to do something.

Lou Katz's version is a bit different:

A large number of bug fixes was collected, and rather than issue them one at a time, a collection tape ("The 50 fixes") was put together by Ken. Some of the fixes were quite important, though I don't remember any in particular. I suspect that a significant fraction of the fixes were actually done by non-Bell people. Ken tried to send it out, but the lawyers kept stalling and stalling and stalling.

Finally, in complete disgust, someone "found a tape on Mountain Avenue" which had the fixes. [The address of Bell Labs is 600 Mountain Avenue, Murray Hill, NJ.]

When the lawyers found out about it, they called every licensee and threatened them with dire consequences if they didn't destroy the tape, after trying to find out how they got the tape. I would guess that no one would actually tell them how they came by the tape (I didn't). It was the first of many attempts by the lawyers to justify their existence and to kill Unix. By this point there were many more lawyers working on Unix than technical people. Also, the license terms started to change and the fees went up very rapidly.

That same autumn two new graduate students arrived on the Berkeley campus: Chuck Haley and Bill Joy. They were fascinated by the new system and began working on the Pascal system that Ken Thompson had hacked together. Haley and Joy improved the Pascal system to the point that it became the programming system of choice for the students. But when the Model 33 Teletype terminals were replaced by ADM-3 screen terminals, Haley and Joy felt frustrated by the **ed** line editor. They took an editor called **em** that had been developed by Coulouris at Queen Mary College, London, and developed the line-at-a-time editor **ex**. Coulouris had been the first recipient of Unix (Version 4) in the UK, in late 1973. He told me:

I developed **em** at QMC in the autumn of 1975, to enable us to exploit more effectively some vdu terminals that we had re-

cently acquired. We had a background of experience in developing applications on graphical displays with both cursor-key and tablet-driven cursor positioning (this experience came to us with William Newman, who spent a year and half working with us between working at Utah and Xerox PARC). I concluded that Unix's "raw" mode of terminal input, which was at that time almost totally unexploited, could be used to give some of the convenience and immediacy of feedback for text editing that we were used to in our single-user graphical systems, and this led me to develop **em**.

By the way, **em** stands for **editor for mortals**—I christened it that after Ken Thompson visited our lab at QMC while I was developing it and said something like: "Yeah, I've seen editors like that, but I don't feel a need for them, I don't want to see the state of the file when I'm editing."

The development of screen editing for Unix was probably never considered at Bell Labs partly because of the attitude expressed by Ken, but also because for many years they had no suitable terminals. They carried on with TTYs and other printing terminals for a long time, and when they did buy screens for everyone in the Unix group they got Tektronix 4014s. These were large storage-tube displays. You can't run a screen editor on a storage-tube display as the picture can't be updated! Thus it had to fall to someone else to pioneer screen editing for Unix, and that was us initially, and we continued to do so for many years.

Then I spent the summer of 1976 as a visitor to the CS Dept. at Berkeley. I worked in a room full of teletype terminals using the departmental Unix. I had brought **em** with me on DECtape and installed it there for my own use. (Although **em** was designed for vdus, its single-character interaction could be used—painfully slowly—in TTYs—the current line being reprinted after each interaction!)

One day, sitting at the next terminal was this fairly frenzied hacker/Ph.D. student (Bill Joy) who told me that he was writing a Pascal compiler. I showed him **em**, and he said, "that's nice, the system support people might be interested in

that." He took me and introduced me to them. They had a couple of PDP-11s (I'm not sure which models) supporting several rooms full of vdu terminals connected at 9600 baud, an environment in which **em** could really shine.

I explained that **em** was an extension of **ed** that gave key-stroke level interaction for editing within a single line, displaying the up-to-date line on the screen (a sort of single-line screen editor). I explained that this was achieved by setting the terminal mode to "raw" so that single characters could be read as they were typed—an eccentric thing for a program to do in 1976.

The system support person [Jeff Schriebman] said something like: "That's very nice, but if we made it available to all of our users the overheads associated with running in raw mode (a process swap on each key depression) would swamp the cpu."

I was rather depressed by this reaction, thinking "I guess I have been unrealistic in developing an editor that is so expensive to run—it's ok in our small system, but it's no use in the big Unix environment at Berkeley."

Nevertheless, Bill and the system support people took a copy of my source to see if they would use it. I then went to the East Coast for a week or so. When I returned, I found that Bill had taken my code as a starting point and had got a long way towards what was to become **ex** and subsequently **vi**, and that the editor was installed on the service machines—of course it still required "raw" mode, but clearly, the benefits had outweighed the objection.

Seeing **em** was probably what alerted Bill Joy and the BSD people to the possibility of a screen editor for Unix. If I hadn't developed **em** and visited Berkeley with it, early versions of BSD Unix probably wouldn't have contained a screen editor. This can be chalked up as a small example of a reverse flow of ideas across the Atlantic in the Unix story.

Unfortunately, **vi** didn't pick up some of the human interface principles that were embedded in **em**. At QMC we had already concluded that modal interaction was a bad idea, and

> I had gone to some lengths to ensure that the interaction in **em** didn't involve more than one meaning for any key. This principle was badly violated in **vi** with its insert and edit modes.
>
> Consequently, we have never used **vi** at QMC. Instead, a colleague, Richard Bornat, designed a full-screen editor based on sound human interface principles, in collaboaration with one of my Ph.D. students, Harold Thimbleby. The QMC editor was **ded** (display editor). It was almost entirely mode-free, but required terminals with a reasonable number of function keys. It was widely distributed and adopted in Europe in the late 70s and early 80s, but was gradually superseded by **vi** and **emacs**.

Meanwhile (in another part of the universe), Kirk McKusick, at that time a "JCL hacker" at Cornell, was exposed to Unix by a friend who was studying at the University of Delaware. "He showed me how you could play games on it," McKusick told me,

> so I really didn't have any exposure to Unix till I got out to Berkeley in 1976. I came out in the spring because I was looking at graduate programs to go to, and they were on spring break, so there weren't a lot of people around. But I came across Bill Joy in the computer room where he was busily hacking away and he said "Hi, I'm Bill Joy, this is what we're working on here—a Pascal system that runs under Unix." And I said, "What can you do with Unix?" So he said "You can edit files and compile files and you can play chess. Let me log you in." So I proceeded to play chess the rest of the afternoon. And I decided I liked Berkeley and that's where I'd go.

When Thompson went back to BTL at the end of the summer of 1976, Haley and Joy turned their interests to the kernel. With Schriebman supervising, they installed the fixes on the "50 Bugs" tape. After learning how to manipulate source, they began to suggest improvements.

At the same time, news of the Pascal compiler had gotten around and (early in 1978) Joy began producing the Berkeley Software Distribution (BSD). It was offered to Tom Ferrin in a letter of 9 March 1978.

The "license" was one side of a sheet of paper. Tom signed it on 13 March. The tape consisted of

> a) Unix Pascal System
> b) Ex Text Editor
> ... created by
> a) W. N. Joy, S. L. Graham, C. B. Haley, K. Thompson
> b) W. N. Joy

The descriptive sheet that accompanied the license stated that

> The distribution is a standard "tp" format, 800 bpi magnetic tape. A 1200 foot reel is the minimum and preferred size.

It cost $50.

I have gone at such length about this, because I feel that it exemplifies what was best about Unix in its first decade, what made it such a popular OS.

Something was created at BTL. It was distributed in source form. A user in the UK created something from it. Another user in California improved on both the original and the UK version. It was distributed to the community at cost. The improved version was incorporated into the next BTL release.

There was no way that Patent and Licensing could control this. And the system got better and more widely used all the time.

Bill Joy, acting as distribution secretary, sent out about 30 "free" copies of BSD in 1978. But the arrival of a few ADM-3a terminals with addressable cursors made it possible for him to create **vi** (visual editor). But this led him to something else: optimizing code for several different types of terminals. Joy decided to consolidate screen management by using an interpreter to redraw the screen. The interpreter was driven by the terminal's characteristics—**termcap** was born.

By mid-1978, enough had been done (the Pascal system was more robust, it could be run on PDP-11/34s, as well as **vi** and **termcap**), that a Second Berkeley Software Distribution was put on tape. Bill Joy answered the phone, put together the distributions, incorporated user feedback into the system. He also shipped nearly 75 tapes of 2BSD. (The most recent version of Unix for the PDP-11 was 2.10.1, available from the USENIX Association in 1989. It was about 80Mb and cost $200. Still a real bargain.)

Paula Hawthorn recalled those days for me:

> Kashtan had just come out with his comparison of Unix vs. VMS and I had already finished a lot of my work for my thesis, so it must have been in the winter of 77/78. A bunch of EECS students went cross-country skiing for a weekend. Bill Joy was there and Mike Ubell and me and others. We sat on the floor in the kitchen of the condo we had rented (in the kitchen because we were talking about work, thus banned from the living room). As Bill read Kashtan's better performance numbers, Bill and Mike read the Unix source code, and I explained how disks and buffering work in other OSs—my masters' performance measurements had been on a CDC 6600 running Kronos—what had happened with my Ph.D. research on the performance of Ingres was that Ingres performance was clearly very dependent on being able to physically sequentially read logically sequential data. Unix needed the equivalent of extents. Oh brother, no one wanted to hear that, but I had performance measurements and Kashtan had performance measurements. So we went through the Unix code, deciding where to make things go faster. It was the beginning of making Unix a high-performance system...

Mike Ubell (who later wrote history) said:

> I remember Bill spending lots of time looking at the code while doing other things. He was doing low level cpu optimizations, getting the instruction counts down.

Ubell also told me about the shell history mechanism:

> Bill had implemented a history mechanism in **csh**. I loved it because of my notorious ability to misspell even the most common word and the ADM3a did not have cut and paste. I think he was writing things to a file and did not like the way it worked so he took it out. Having lost a productivity tool, I reimplemeted it using the in-memory parse structure that the shell already used. Bill liked the way that worked and away he went. My version of the shell lived for several months around Berkeley, till the next version of Bill's appeared.

The man page for the C shell (under "History Substitution") still illustrates Ubell's spelling problems:

> History substitution allows you to use words from previous command lines in the command line you are typing. This simplifies spelling corrections and the repetition of complicated commands or arguments.

Ubell's experience also fits perfectly with what Eric Allman said: One general rule of software design is that you should be writing a program that you want to use.

Early Editions and the PDP

UNIX Edition	PDP Model
1st	PDP-7
3rd	PDP-11; 11/20
•	
•	PDP-11/45
•	
6th	PDP-11/70
7th	PDP-11, Interdata 8/32

CHAPTER 17

Version 7

Version 7 Unix offered several major improvements as it came from the Labs in June 1979: it accommodated large filesystems; it did not restrict the number of user accounts; it had improved reliability. Steve Johnson referred to Version 7 as "the first portable Unix." It had tremendous influence because of this and because of the number of new commands it contained. Let me list a selection of these.

Commands

at
awk
calendar
cb
cd
cpio
cu
deroff
expr
f77
find
lex
lint
m4
make
refer
sed
tail
tar
touch
uucp
uux

System calls

ioctl

Subroutines

malloc
stdio
string

Games

backgammon

awk (Aho-Weinberger-Kernighan), lint (Johnson), make (Feldman), and uucp (Lesk) would have been enough. But V7 had lots more.

The Seventh Edition programmer's manual had grown to nearly 400 pages, with two 400-page supplementary volumes accompanying it. This Unix contained a full Kernighan and Ritchie C compiler; a far more sophisticated shell sh (the Bourne shell); Dick Haight's find, cpio, and expr, and a large number of include files.

Together with all of this, Version 7 Unix came with a major drawback: its performance was poorer than most V6 systems, especially those that had been "tuned." The users went to work to better this situation. Bill Joy (at Berkeley) changed the size of the data blocks in the filesystem on the VAX-11/780. His implementation was then ported to the PDP-11/70 by Jeff Schriebman in April 1980 (he had gone to UniSoft from Berkeley). In December 1979, Ed Gould (then at Berkeley) moved the buffers out of kernel address space. Joy changed the *stdio* library on the VAX, and Tom Ferrin (at UC San Francisco) ported the changes to the PDP-11. Tom London in Holmdel improved the movement of output characters from user space to kernel space. Ferrin also wrote a dynamic unibus allocation scheme. John Lions at the University of New South Wales proposed a new procedure for directory pathnames. UNSW also provided new code for process table searches. Bruce Borden (at the RAND Corporation) provided the symorder program. Ferrin also rewrote parts of copyseg() and clearseg(). The entire set of improvements was made available to the community on a PDP-11 distribution: 2.8.1BSD; it was announced by Ferrin at the USENIX Conference in January 1982, in Santa Monica, CA. The users had enhanced V7's performance tremendously.

I have gone to such lengths here because these improvements to V7 came from industry and academia, from the US and Australia. And all of them were incorporated into future releases of both BSD and AT&T Unix.

V7 also gave rise to several Unix ports: the 32-bit implementations as well as XENIX2, a Microsoft-Santa Cruz Operation collaboration, which was the first Unix implementation for the Intel 8086 chip (XENIX1 was based on V6). V7 also gave rise to Unix for the Z8000 and 68000 chips. And 32V (the Holmdel version) gave rise to 3BSD and all its descendants.

The evolution of 32-bit Unix was narrated to me by Steve Johnson.

The first port of Unix that I was aware of was done by a couple of Princeton students. They ported a very significant bunch of Unix commands and the OS interface—the shell—to an IBM 360. I think a TSS system. They had a surprising amount of Unix going on the system. They made use of our IBM C compiler, with our knowledge. It was a kind of beguiling thing to do. We hired one of the students, Tom Lyon, for the summer, and he ported all the V6 commands to the Center, to the Amdahl.

I remember when we decided to port Unix. We chose the Interdata; which was an interesting experience because IBM would have given us a 360 for two-and-a-half years, but Dennis didn't like the IBM channel programs, he'd tried to code those things.

Marc Donner explained to me:

The IBM 360 was a multiprocessor. I/O didn't go between the CPU and devices directly, it was under the control of subsidiary processors called "channels." A channel processor was a special purpose engine that knew how to talk directly to various devices (under the control of "channel programs" written in what amounted to machine code)... it delivered data to the processor by writing into the main memory and giving the CPU an interrupt. It received data from the CPU by means of a similar technique....

In the early days, channels were much dumber and had channel commands, rather than channel programs, but as devices got more complex and required more and more sophisticated handling, the facilities for channel programming became more and more complex.

I don't think there ever was a proper high-level programming tool for channel programming and as a result it remained an extremely arcane process performed well by only a very small number of people.

Returning to Johnson:

> IBM recognized what we were proposing better than Interdata did. The other big contender was a DEC 20, which was a 36-bit machine, and Dennis's comment on that was that he'd port to a DEC 20 when someone invented 10-track tape drives. The problem of writing 36 bits on 8-bit-wide bytes was just sufficiently beyond what the Unix file system handled....
>
> The Interdata was the last machine I ever used that had real core—steel donuts with wires running through them. Interdata was then sold to Perkin-Elmer and they, in turn, sold it to Concurrent, I think.
>
> Well, one day we just got a letter from these guys in a place we'd never heard of—Wollongong, Australia— saying we've ported Unix and here's what we've done. They used a rather different approach to doing systems work. Kind of a more top-down approach. They implemented a layer on the system they had there that was the Unix system calls, and slowly infiltrated that layer down and replaced pieces of the existing kernel until it looked like the Unix kernel. It was an interesting approach, because it got them up on the machine very quickly. But on the whole, we got finished in a very similar amount of time.

I pointed out that the University of Wollongong was brand new at that time. Johnson laughed.

> Well, it was certainly the most astonishing letter I'd ever received. You can't imagine. We thought we were breaking new ground in science and some university we'd never heard of halfway around the world writes and says "we thought you'd be interested that we're doing this."
>
> One interesting vignette about the Interdata—there were a number of problems, some of the strangest problems I've ever encountered—concerned the paging hardware. In 6th Edition Unix it was very common to represent error by returning –1 as the pointer value. And a common programming

error was to forget to check for error when the pointer was returned and then just go ahead and use the pointer. Well, in the Interdata, if you attempted to access a word that was at size -1, you actually got two faults that were generated: one of which was an alignment fault, because -1 wasn't word-aligned; and the other was a memory fault, because you were outside of the memory bound. And it turned out that these faults happened a couple of ticks apart, so the microcode engine was several ticks into processing one fault when it received another fault, which caused the microcode engine to become completely curdled in its state. Not only did it totally lose track of where you had been, but it curdled the engine so that it mis-executed the next several instructions before it managed to get itself back into synchrony.

That was inherent in the hardware. Any user program could have put -1 into a register and blow it into a psychotic state. We also had editions of that machine that could only be cleared by turning the power off and turning it back on. So Dennis and I went down to have a talk to Interdata.

And I said, "Look, here's this whole Bell System with umpty-ump machines we're using—which at that time was in the hundreds—and you've got a 32-bit machine and we've got our Unix system and the following applications, and the only thing we need are a couple of changes in the hardware. And then you guys can clean up selling us Unix systems." And they said, "No."

So, Dennis and I became the first in the long history of people who, having done one Unix port, decided they never wanted to do another one.

Dated January 1979, the title page of the Seventh Edition **UNIX PROGRAMMER'S MANUAL**, bore neither Dennis' nor Ken's names. It was headed:

UNIX™ TIME-SHARING SYSTEM

Unix was nearly ten years old.

Interestingly, it was already in high school. In January 1979 Brian Harvey went to Lincoln-Sudbury Regional High School, in the suburbs of Boston, "to set up a Computer Department." He talked the school board into a bond-issue for equipment and persuaded DEC to give the high school a massive discount. The result was about $200,000 in equipment for $50,000. However, installation "took the cooperative efforts of computer scientists at several places ... because our PDP-11 had a type of disk drive that the original version 7 ... couldn't handle." Harvey's 15-year-olds solved that problem and several others; they began writing useful programs. And they became part of the Unix community, sharing their software. Harvey said:

> Several months after our system was finally on the air, it was a real thrill for the kids and for me when we got a phone call from another computer center that had just bought a Unix system and had the same disk problem we'd had. We were able to send them a boot tape tailored to their configuration. Since then, we have contributed student-written software through USENIX...
>
> High school students benefit from Unix for the same reasons that adult programmers do: its flexibility, the capability to modify the way it works, and the wide variety of tools available to support it.

V7 led to more "goodies" than the local candy store—no matter where the programmer's store was. V8 ported vi (by Bill Joy), curses (Ken Arnold), and termcap (Joy) from BSD. Arnold's curses was yet another example of the influence of games on software development: curses is a screen handler and optimization program that Arnold wrote to make the playing of rogue easier. Arnold, incidentally, also wrote fortune and many other useful and enjoyable programs. But as useful as V7 was, it also was irksome. Not because of the code, far from it. No, as Andy Tanenbaum put it:

> When AT&T released Version 7, it began to realize that UNIX was a valuable commercial product, so it issued Version 7 with a license that prohibited the source code from being

> studied in courses, in order to avoid endangering its status as a trade secret. Many universities complied by simply dropping the study of UNIX, and teaching only theory. [*Operating Systems* (1987) p. 13]

Tanenbaum's solution was "to write a new operating system from scratch that would be compatible with UNIX," but without "even one line of AT&T code." Tanenbaum called it MINIX.

CHAPTER

18

Berkeley Unix: Part II

Early in 1978, Professor Richard Fateman began looking for a machine with a larger address space, so that he could continue his work on Macsyma begun on a PDP-10. Fateman explained:

> The program at MIT's Project MAC was called MACSYMA. In order to distinguish the VAX version from the original one which ran only on the PDP-10 (and to some extent on the Multics system), I called it VAX/Macsyma, then sometimes shortened it to vaxima. MIT subsequently sold the program Macsyma, as well as its name, to Symbolics, Inc. All subsequent versions not under the direct control of Symbolics, or its successor in ownership, "Macsyma Inc." of Arlington, MA, have been called something other than Macsyma.
>
> DOE-Macsyma—(DOE= Dept of Energy)
> Paramax— (Paradigm Associates)
> Maxima—(Common Lisp version ported by Bill Schelter of the University of Texas)
> Aljabr—(a Macintosh version by Jim O'Dell).
>
> All of these off-shoots are essentially similar to each other and to vaxima, although each proponent will claim superiority.

Anyway, the new DEC VAX 11/780 seemed to fit Fateman's requirements and—together with 13 other faculty members—Fateman put to-

gether an NSF proposal to be combined with departmental funds. Initially, the VAX ran DEC's VMS OS.

But the department had gotten used to Unix and Fabry obtained a copy of the 32V port of Unix to the VAX that John Reiser and Tom London had done at BTL in Holmdel. The manager of the porting project was Charlie Roberts, so here's the story in his words:

> In '77-'78 DEC pre-announced the VAX. Dennis and Ken and Steve had gotten fairly alienated by DEC. They felt that their work had sold a lot of machines, and DEC was still refusing to support Unix and was going ahead with that VMS stuff. So when DEC offered them a VAX, they just said no. [Doug McIlroy told me that another reason was "that the VAX has an offensively fat instruction set. It's full of irrelevancies that go against the values of Ken and Dennis."] Dennis and Steve were working on the Interdata port, anyway. Steve called it the "Intersnail."
>
> So DEC came to us at Holmdel. We were clearly the second string. Tom London and John Reiser were interested and so was Ken Swanson, and we got the VAX in early '78. I didn't do any of the technical work. In fact, I devoted a lot of energy and time to getting the management to let us do it. It wasn't research, you see. However, they let us take the time. And in about three months my small group ported Version 7 to the VAX. We got the machine in January, they had it running in April, and by August it really worked. By then folks knew what we were doing. I had had calls from about six universities—Brown, UCLA, Berkeley, Waterloo, I can't recall the others. So I went to Roy Lipton and Al Arms in Patents and Licensing about getting it out. After a lot of back-and-forth, they decided that we could give it to one university for research purposes and that Al would set up a "special research agreement" with that institution.
>
> I had met Fabry at a couple of conferences and I had been out in Berkeley and given a paper and talked to Ferrari as well as Emmanuel Blum and Bill Joy. So, with the blessings of BTL Area 11 management, we sent 32V to Berkeley. It was in October or November, 1978.

Roberts' management who approved were C.C. Cutler, R.W. Lucky, and W.S. Boyle. The 1127 management—Doug McIlroy, Sam Morgan, and Bob Prim—also approved.

It was 32V that became 3BSD in 1979. Kirk McKusick told the tale to Peter Collinson:

> When Ken left to go back to Bell Labs, he left behind a Unix system. Bill and Chuck Haley took over the care and feeding of that system. At that time, they did things like upgrading to Version 6 and incorporating the "50 Bugs" tape from the Labs. Later they upgraded it to Version 7.
>
> Bill continued to work more in the area of the utilities. He got fed up with the shell, took some of the ideas from John Mashey's shell [in PWB] and came up with **csh**. Bill also got rather tired of **ed**, and decided to work on **ex** that was an expanded line-oriented editor. Then we had the upgrade of several of our terminals from ADM-3s that were not cursor addressable to ADM-3a terminals that were. There were only two of these in the room and Bill really liked to use them. He started to work on **vi** so he always had an excuse to kick off whoever was on the new terminals. The original **termcap** was invented so that the same editor could run both types of ADM terminals.
>
> There is a fabled story about why **vi** doesn't have multiple windows. Bill hacked the code and had it half working. Then there was a disk crash. The disk hadn't been backed up so the version that he got off backup tapes didn't have the windows in it. He just never got back around to doing that.
>
> The original Berkeley Software Distribution, called BSD not 1BSD, consisted of the Pascal System and **ex**. Bill, in addition to being a very talented programmer, was also a very talented marketing person. He would go out and give real rabble rousing speeches about how great this stuff was. How the Pascal interpreter took a tenth of the time than the C compiler *and* it had error correction so it was much better for students because you could support more of them. He sold about 35 of these BSD tapes, sold in the sense of "made them available"

for something like $35. [It was $50.] The BSD distribution at that point was Bill, he was the hacker, the phone answerer, the tape spinner, he wrote addresses on the labels, sent the tapes out, everything.

2BSD was a follow-on to that, it came about a year later and was the Pascal system, ex, vi and csh. He did that single-handedly also and shipped around 100 copies of that over the space of a year....

That would have been V7 based, because 1978 was when we got the pre-release of UNIX/32V. This was V7 moved to the VAX. It had no paging or anything, it was just a traditional swap-based Unix system running on the VAX. We had a very low numbered VAX-11/780, a single digit serial number.

Bill's first job was to get 32V running on the VAX. This was fairly straightforward provided that you had exactly the same hardware as the release tape. There was no notion of auto-configuration. You had to have exactly what they had, you couldn't have anything extra, and you couldn't have anything missing. There had to be precisely two disks and they had to be RP06s.

The next thing was to bring the utilities across. I had been doing things with the Pascal interpreter, adding some bells and whistles and filling out some of the parts that weren't quite right. Bill said to me "why don't you bring the interpreter across?"

It had been written in assembly language and "it would be really easy, there are just these little operations and the assembly language looks almost the same." He did a couple to show me how easy it was going to be. I went off and spent the next month doing that. I eventually backed off from assembler and wrote it in C.

One of the other problems was that the VAX only had 2MB of memory.... There was a great push to actually use some of the virtual memory hardware on the machine. Özalp Babaoğlu working with Domenico Ferrari had designed a paging system for the VAX. He enlisted help from Bill in bringing it up and getting it running since Bill knew a bit about the

system by then. Over the vacation period of December '78 to January '79, they could get the VAX11/780 stand-alone so they could boot different systems. You would login and have a banner saying "Virtual VAX Unix," inevitably you would be working for a while and the system would freeze up. There would be a 5 minute wait and it would come up announcing itself as 32V.

It was flipping back and forth but at the end of the vacation they had it running in a stable manner and it became the base system.

Bill then decided to stop doing distributions that were just utilities and switch to packaging complete systems. He created the 3BSD release. This was a complete system with the virtual memory based kernel and the utilities that had been ported across. It was a complete bootable system, it had a boot block at the start of the tape and you could roll it in onto raw hardware. He still did all the work to make the release happen.

Meanwhile, I had started to get more seriously drawn into the kernel stuff. I remember the event that caused me to get involved. Bill was trying to debug part of the virtual memory system. He would carry around these big program listings of the kernel that would be about an one and a half inches thick. He came in, and in his very enthusiastic style, he threw the listing down on the desk open to some page. He circled the code around and around and around. "Somewhere on this page is a bug due to a race condition, and I'll give $20 to the person who finds it."

I had never looked at the kernel source code, but I had taken an operating systems course, and so I knew that the way you got race conditions was that you would have something where you would get a context switch. I looked through the code and it was standard stuff, there were only two subroutine calls showing on the page. I pointed to these subroutine calls and asked if either of them could do a context switch. "That's it! That's it" and he rushed out of the room with paper trailing behind him. It was hard to work in an office with Bill and not get infected with his enthusiasm.

> The 3BSD system went out and was noticed because a number of universities started getting the VAXes. You either had to run VMS or 3BSD if you wanted a paging system. Many people wanted Unix but they also wanted to take advantage of the paging. So, 3BSD was quite popular among early buyers of VAX machines.

There is a footnote to this. McKusick told me: "The debt went unpaid for years and years. Bill went off to Sun. Bill became a multi-millionaire. Four or five years ago, when UNIX International was being formed, they were trying to get some publicity, and they were going to have a big press conference to announce their existence at UniForum [San Francisco, January 1989]. And they decided they would give Industry Awards to various luminaries in the Unix biz, and the press would come to see the luminaries. They nominated the folks—Dennis and Bill Joy—you know 'Round up the usual suspects.' Anyhow, I got added to this list. Well, before this thing they were talking to us, getting tid-bits. And I told them the story of how I got to Unix and everyone went ha-ha. But we get up there and they started with the important people like Dennis Ritchie, and they finally get to me and what do they do? They read this story. And there's Bill on the stage clutching his plaque. And they say: 'Kirk reports that he's never been paid.' And as I walk behind Bill Joy, he hauls out a $50 bill from his pocket, flashes it, and says, 'Here's your $20 with interest!' So I sat down. And later I asked him, 'Bill, did they warn you this was going to happen?' And he said 'Heck no. If they had warned me, I would have had a twenty.' "

In 1980 it was Bill Joy who flew to New Jersey and, together with Steve Bourne, put 4BSD up in Bell Labs. "We did it overnight," Joy told me. "We decided to do it about 9 p.m.; people showed up in the morning and a new os was running. It was pretty amazing." It sure was.

Keith Bostic put together a sketch of the history of the development of Berkeley Unix up through the release of NET 2. Here is the beginning of his outline.

> *a) Early systems running at Berkeley were Fifth Edition (Version 5, V5), Sixth Edition (V6, about 1976) and Seventh Edition (V7, about 1978), all from Bell Labs. All ran on versions of the DEC PDP-11 computer. These systems were continually modified at*

Berkeley in the Computer Center and in the EECS Department. These modifications included:

b) Berkeley Pascal tape, BSD— Software written for use with V6 on PDP-11 distributed by Berkeley about 1977.

c) 2BSD—Software written for use with V6 and/or V7 distributed by Berkeley starting about 1978. New software and modifications were added from time to time, including Mail, more, csh, ex and vi, Pascal software, etc. 2BSD was shipped in different versions to sites with V6 and/or V7 licenses.

d) 3BSD, late 1979—This is the first Berkeley release for the VAX. It was based on 32V, with Berkeley utilities and modifications from 2BSD, plus a virtual memory system done by several graduate students including Bill Joy.

From 1969 to V6 there was only one interpreter, a program called the Shell. The manual entry in Fourth Edition (dated 4/18/73) was three pages in length. With the inclusion of Bourne's shell in V7 and Joy's C shell in 2BSD, the programmer had a choice (though not if that programmer had System III). This, as might be expected, gave rise to conflicts as to "which?" In the mid-1980s, David Korn of AT&T Bell Labs invented the Korn shell. It was incorporated into "Experimental Toolchest" in 1986 and became a part of USL's SVR4 distribution in 1989 (the version is dated 16 November 1988). In general, the Bourne shell is upwardly compatible with the Korn shell. There are a number of other flavors of shell, but sh, csh and ksh are the most common.

The fact that Unix would now run on more than one machine was what brought it to the Defense Advanced Research Projects Agency's (DARPA's) attention. DARPA's notice meant that the agency had become concerned that their various contractors were using different hardware and a range of operating systems. Moreover, there was no possibility of software exchange—the software was simply incompatible. DARPA wanted a common base, so that there would be more interchange. McKusick told me that "It was pretty clear they were going to choose the VAX, the question came down to whether VMS or Unix should be the operating system of choice." He told Collinson:

> Bill managed to convince DARPA that Unix was a better base because it had the ability to move to other systems as well, so they would not be tied to the VAX. Bob Fabry was doing the

> political side of the convincing while Bill was doing the technical side.
>
> They managed to land a big DARPA contract to get these performance enhancements put in and add some other basic things that were needed. The promise was that this would come out right away. So 4BSD was put out within two months of the DARPA grant arriving at Berkeley.... This was late in 1980.

According to Collinson, "The reason that 4BSD became popular was that the VAX-11/750 came out and 4BSD was the the only Unix system that would run on it." McKusick continued:

> Armando Stettner at DEC managed to arrange for Bill to fly out to Maynard to see one of these new 750s that were to be announced but were not yet available. It was code-named *Comet*. Bill and Armando hacked together something to ensure that 4BSD would run on these Comets, so when 4BSD was released it was known to run on the 750s even though we didn't have one at Berkeley. In fact, later the 750 became the work-horse of the Department because you could afford to buy a six-pack of them for around $150,000. [*It was Bill Joy, Bill Shannon and Armando. Armando's story is in a later chapter.*]
>
> [4.1BSD] was the high point of performance, in the sense that it was really 4.0BSD but tuned to a fine hone especially for the 750. The 4.1BSD system was taken back into Bell Labs to become the 8th Edition Unix, then the 9th and 10th. It also was the one where we were raked over the coals by Rob Pike for having **cat -v**. [*At the Toronto USENIX Conference in June 1983, Pike railed at the lack of style in Berkeley Unix, talking about 'cancerous growth' and remarking that 'cat-v [is] considered harmful.' Over the past decade, the bloat has increased even more severely: the V7 kernel was 40KB; Ritchie told me that the 10th Edition "kernel" was about "150KB." Armando Stettner pointed out to me that given the choice between 4.1BSD and V7, no one ever wanted to go back to V7.*]
>
> The DARPA money came in. This meant that Fabry offered Bill a real salary as a staff person. There were support staff, someone else to answer the phone, spin tapes, and deal

with licenses. Bill was able to concentrate on things that he wanted to concentrate on. The project started to expand in the sense that Bill was able to hire a technical employee, this was initially Michael Toy and later Sam Leffler.

The work between 4.1BSD and 4.2BSD was the real reason why DARPA had come in. They wanted some serious enhancements, like a file system that could use more than 5% of the bandwidth of the disk. They wanted networking that was to be based on TCP/IP.

An initial prototype was done by BBN and given to Berkeley. Bill immediately started hacking on it because it would only run an ethernet at about 56KB/sec utilising 100% of the CPU on a 750. This was fine for BBN because 56KB/sec was the speed of the main trunk links but it just didn't cut it for ethernet. Bill lobotomized the code and increased its performance to on the order of 700KB/sec.

This caused some consternation with BBN when they came in with their "finished" version, and Bill wouldn't accept it. There were battles for years after, about which version would be in the system. The Berkeley version ultimately won. [*Ritchie told me that this was one of the real issues when he was on the DARPA committee. "Every six months we'd say, use the BBN version, and Bill would say no."*]

The third component that was supposed to be in 4.2BSD was a new virtual memory system that would allow shared read/write segments and so on.

There are some things in the system that are there not because Bill believed that they should be there but because the steering committee required that they should be. Passing file descriptors in messages is one example of something that was imposed, but Bill didn't believe in.

Something else had happened at Berkeley with the advent of the VAX: computer mail, what became sendmail. As that was written by Eric Allman, a graduate student at the time, I'll let him narrate.

Eric Schmidt had been working on BerkNet and BerkNet was connected over 9600 baud tty lines—it was a batch system, like **uucp**— and that was one of the periods when the rela-

tions between Bell and the university seemed to be iffy, so we couldn't get **uucp**. Of course we did, fairly quickly. What happens is that industry decides "Oh, we wouldn't want the university to have that because we might lose it"; then the university does it themselves, so industry goes "Oh my God. We wanted ours to be standard, we'd better give it to them." BerkNet was a fine network for its time, but it had little things like the hostnames were single character. We thought the ARPANET was bad for having a limit of 255 hosts this was 26 hosts! C was cory; E was ernie kovacs; I and J were the Ingres 11/70 and VAX, respectively; the Computer Center got A through F, which was Unix-A through Unix-F, they thought they were very creative at their naming policies.

So there were lots of people with terminals in their offices and starting to use email and services more. We had this network that supported mail and primitive file copy and even-more primitive remote execution. I really mean primitive, the numbers were about an hour. The pressure that came up was from professors in the department [EECS] who wanted to be on the ARPANET. I think the ARPANET was still on the 11/40 because the interface link wouldn't plug into the 11/70. The result was that we didn't want to give accounts to everyone, but we were forced to. We had to fix things so that ARPANET mail would go out along BerkNet, but not the other way round. You've got to understand, at that time people didn't put headers on their mail, except on the ARPANET, which had RFC822 [Requests for Comment are the equivalent of standards in the networking community], which wasn't very well enforced.

At that time you had **uucp** coming in to ernie kovacs, BerkNet mail within the UC campus, and ARPANET mail going out, each of which used different mail standards. If you wanted to send mail to the ARPANET and to **uucp** your only choice was to send it twice. This was, pretty clearly, not a good idea.

I thrashed for a long time, not being able to see what was the right way to fix this. One day I sat down in my living room, I was living on Glen Street, and I said, "OK, this is stupid, I'm

going to write down the *ad hoc* code." And as I wrote the *ad hoc* code, it became clear that there were patterns and later that afternoon I figured out what the configuration table should look like. The configuration table for **delivermail** was compiled in—it looked at characters. It said "Oh, there's an @, it must be the ARPANET; ! it must be **uucp**; : it must be BerkNet." The idea was that every network had its own magic character.... Anyway, **delivermail** was shipped on the 4.0 or the 4.1 tape. But it started to become clear that that configuration was inadequate. It was just unwieldy as we got more machines.

So I began working on a revision of **delivermail** and Bill Joy ragged on me. He said "It's not delivering mail, it just hands it off to another agent." So I changed it to **sendmail.** So **sendmail** is really just **delivermail** version 2 or 3.

It's worth pointing out that during this the ARPANET was undergoing the transition from NCP [Network Control Protocol] to TCP [Transmission Control Protocol]. And that was an extremely painful period, going from **mit-xx** to **mit-xx.ARPA** to **xx@mit.edu** over a couple of years. It didn't take me long to figure out that the easier it was for me to change **sendmail**, the more likely it would be that I could keep up to date.... I'm doing a major revision right now, because of RFC 1123 "Host Requirements"....

The ARPANET has its beginnings in 1968 as a small research experiment and was delivered to ARPA in 1969. It demonstrated the viability of long-haul packet switched networks and eventually grew into the US backbone network, the Internet, including NSFNET and others. It also demonstrated the need for a common protocol. Each of the nodes of the ARPANET uses a communications subnet, called an Interface Message Processor (IMP). These were originally Honeywell hardware, then BBN hardware, communicating with hosts by means of the BBN 1822 Protocol (the number of the Report that described it). The original host-host protocol was NCP (Network Control Protocol).

In 1976 it became clear that there would be a proliferation of local networks that needed to be interconnected. This led to the development of the TCP/IP protocols. The first public data network, incidentally, was Datapac, a Canadian network begun in 1976. Thanks to

the ability and foresight of Vint Cerf and Robert Kahn in 1974, and the funding of BBN to implement the protocols and of Berkeley to integrate the protocols into the Unix software distribution, DARPA was able to connect about 90% of the research university community.

In this, there was genuine synergy: TCP/IP, which enabled greater connectivity, was integrated into Berkeley Unix; Unix became increasingly popular, increasing connectivity. I consider it quite fair to say that the worldwide Matrix of over 30 million users we see at the beginning of 1994 could not have come about without Unix. The fact that a very large percentage of these users are running MS-DOS, Apple-OS, or VMS is irrelevant.

Right now, the Internet and the rest of the Matrix provide many services, but only a few "vital" ones: mail, file transfer, and remote login are the most significant. All grew out of Unix.

Let me insert the remainder of Bostic's chronology before returning to McKusick's narrative, as this will give the reader a less-discursive view of the various Berkeley versions.

> *e) 4BSD, Oct. 1980—This release contained performance improvements including a faster file system for use with virtual memory, job control, reliable signals, automatic reboot, the delivermail program, and the Franz Lisp system.*
>
> *f) 4.1BSD, June 1981—4BSD was upgraded to include many performance improvements, support for a new VAX model and autoconfiguration.*
>
> *g) 4.1a—4.1a was a test release including TCP/IP and the socket interface. It was shipped to a fair number of ARPANET sites that needed it to use the network.*
>
> *h) 4.1b—This was a test release used only at Berkeley to my knowledge. It included the new 'Fast File System,' using clustering for much improved performance, and contained the new networking code. [*P4.1b was used in a graduate OS class, see below.*]*
>
> *i) 4.1c, 1982/1983—The third test release between 4.1BSD and 4.2BSD was sent to about 100 sites. It included most of the new features of 4.2BSD except for the new signal facility.*
>
> *j) 4.2BSD, September 1983—A major system revision, including networking (TCP/IP and a general framework), a faster filesystem with new*

features, a substantially redesigned system interface, and a new signal facility.

k) 4.3BSD, June 1986—The 4.3BSD release finished and tuned many of the features in 4.2BSD, and offered substantially better performance. New features included XNS networking, the directory name cache, and an Internet name server.

l) 4.3-Tahoe, June 1988—This intermediate release added support for the CCI Power 6 (Tahoe), the first non-VAX supported by 4BSD. It included several internal kernel facilities, including a memory allocator, a kernel debugger, and disklabel support. The TCP algorithms had been greatly improved by Van Jacobson at LBL.

m) BSD Networking Release 1, Nov. 1988—NET 1—The first Networking Release was a subset of the current Berkeley system at the time, which was quite similar to 4.3-Tahoe. It included source code and documentation for networking portions of the kernel, C library and utility programs. It included a login program that worked with the newer version of rlogin (network login). It was available without evidence of any prior license (AT&T or Berkeley), and was redistributed via anonymous FTP. The source files contained a Berkeley copyright and a legend that allowed redistribution with attribution.

n) 4.3-Reno, June 1990—The 4.3-Reno release was a test release containing a number of new features done for 4.4BSD. The major changes were the addition of a vnode framework for multiple file system implementations, a Network File System (NFS) implementation, a number of changes in the network framework, and OSI networking support. Many of the kernel files, library sources, and newly-written (or re-written) utilities contained a Berkeley copyright notice with a legend allowing redistribution with attribution similar to that used in the first Networking release. In addition to the VAX and Tahoe, 4.3-Reno supported Hewlett-Packard 9000/300-series machines based on the Motorola 680x0 processors.

o) BSD Networking Release 2, June 1991—NET 2—The Second Networking Release contained more than just networking code, but like the first Networking release was available with no prior license. It contained a subset of the then-current Berkeley system, which had advanced beyond 4.3-Reno. The new features included a new virtual memory system derived from Carnegie-Mellon University's Mach system, ported at the

University of Utah, and a port to the Intel 386/486 system from Bill Jolitz.

Let me add 4.4 to this list: it was released in June 1993.

Kirk pointed out to Collinson that some things had been implemented quite recently.

> We now have portals implemented in 4.4BSD, Jan-Simon Pendry has just put them in [see Chapter 28]. Also, the **mmap** system call was described in the original architecture document, many other manufacturers beat Berkeley to the punch in implementing it. We did play a role when Sun was implementing it, we argued with them what the flags ought to be and what the interface should look like. We can say that we had a role in that, even though it was after the fact in terms of the implementation....
>
> The first thing that Bill attacked was the socket interface and the TCP/IP code. The next thing on his agenda was the file system. In the spring, he had sketched out a design picking up a lot of stuff from the system that Mike Powell did on the Cray called DEMOS.
>
> I was still working as a graduate student and my advisor was Sue Graham. As a graduate student, you are supposed to get half time during the time you are in school and then in the summer you are allowed to work full time for three months so you can amass enough money to pay your tuition in the Fall. In the month of June, you should get a paycheck that is three-quarter salary—part-time for two weeks and full time for two weeks. However, at the end of the month I got my paycheck and it was only one quarter. I figured that someone had screwed up in payroll, they didn't know that I was going to be there for the Summer, as I hadn't been there in previous summers. I went round and asked what was happening. Payroll blamed my advisor who needed to supply them with a grant number. I went and told Sue that I needed a new grant number and discovered that she had forgotten to apply for her grant renewal.

It was clear that it would get renewed but that would take several months. In the meantime, I was facing a summer with no salary. I went downstairs because I knew that Bill had gotten the new grant and asked him whether I could go on the DARPA grant to the summer. Could he give me some little thing and I would write up a paper. Really we both understood that I would be working on my thesis. He agreed and suggested that I could perhaps prototype the design for the file system. He knew full well what was going to happen, although I didn't.

I spent the Summer prototyping the file system that Bill had designed. I did it in user space accessing a raw partition on the disk. In the Fall, I had a little demo of it running. As you might guess, Bill suggested that it would be easy to put it into the kernel. This I did. Bill said: "That's really good, people would really like to use that—but it's useless without **dump**, **restore**, **fsck**..." One thing leads to another and 18 months later, we had the Fast File System. It was complicated for its time. It tripled the size of the file system code....

Sam [Leffler] actually had networking experience having worked for a networking company. He was able to make some important modifications to the socket interface to bring it into line with high volume networking, things like having connection queuing, which Bill had made serial. This was very important to make the code robust in the face of heavy usage.

One of our ultimate tests was **ucbvax**, the mail hub and ARPANET **uucp** connection machine at that time. It was slow enough that you had a lot of processing going on there. It wasn't unusual to see 50 or 60 network connections on the machine. This does not seem such a big deal today but at that time really pounded away at things, both at the file system and at the networking code.

[4.2BSD was then released.]

Actually, Bill left Berkeley shortly before 4.2BSD came out. He took the current system with him to Sun. Sun's sys-

tem was ultimately upgraded to 4.2BSD after the official release. Sam took over at the helm, finished up the last bits and pieces of 4.2BSD and pushed that out the door.

This isn't all the Berkeley story, by far. Let me go back a bit, and let Sam Leffler tell his tale.

I was at Case Western Reserve in Cleveland. I entered in '75. I was in this combined BS/MS program and was supposed to graduate in three years. I was supposed to do this project with Bill Shannon. We were going to build a Unix system: get the hardware and port the system. Based on the Z8000 or something. But the funders, who were supplying all the support, backed out. That screwed me up, so I ended up teaching for a while—until I could find another topic to do. So I didn't graduate until '80. Though I really finished in '79. I got here [to Berkeley] in '80. But Bill went off to the USENIX meeting at Columbia [in 1978], and we both went to Toronto the next year. At that time we were working on overlays. Transparent overlay stuff for the PDP-11—this is Version 7—and we met Bill Joy in Toronto, who was simultaneously working on overlays, too. I can still remember the conversations we had with him about this stuff. We had solved some problems he hadn't solved, and he took away these things.

Well Shannon and I graduated. We ended up going through a bunch of projects. Case Western was trying to put in a campus-wide network. They were well ahead of their time. It was total politics. The committee and all the departments had agreed that they wanted DEC equipment, and some regent or chancellor or something came in and said: "You will buy Harris." So we ended up with a lot of Harris equipment. Strangest equipment you'd seen in your life. Didn't run Unix or anything. The equipment never worked. Well, Shannon and I were desperate looking for a project that had funding, 'cause we needed to get out. So we agreed to do a port to this machine. It was a very strange machine. So he did the operating system and I did the compiler work and all the language work. So we had this incestuous, uh, symbiotic

> relationship. One couldn't graduate without the other, 'cause I needed the system to demonstrate my compiler, and he needed the compiler to get the system running.
>
> In the end, you know, the Harris is a pretty fast machine. But it wasn't byte-addressable and it had 24-bit words. We basically took a PDP-11/70-class machine and changed it into an 11/34 with demand paging. We were in contact with Bill [Joy] and we had our system running before he did.
>
> Bill and I graduated and the two of us were out looking for jobs. We had very similar credentials. And I looked at only two places—I wanted to live near Boston or in California. So I interviewed with places in Silicon Valley and with DEC in Merrimack. And DEC hired Bill, so I came out to California.

Armando Stettner, who was at DEC at that time, lent a different perspective to the decision:

> Bill Munson and I had only one requisition. I remember we were recruiting at the University of Delaware at USENIX and these guys were playing volleyball. And though we had only one req, we wanted both Sam and Bill. We really had a hard time so we based our decision on the volleyball game. We watched and here were these guys who wanted jobs and one was flashy and the other was this regular team-player. So we decided to take Shannon, the team-player. Sam went to Sytek.

Shannon only stayed at DEC for a bit over a year: he moved to Silicon Valley to join Bill Joy at Sun Microsystems. Leffler took a job with a company in the Valley, where he stayed for a year. But as they were running VAXes and Leffler had stayed in touch with Joy, he was a beta-test site for 4.0. So when Joy lost a couple of people and there was money available, Leffler "went up to Berkeley. And I was doing 4.0 and then 4.1 stuff. And that's how I got to Berkeley."

John Foderero, a graduate student at Berkeley in the summer of 1980, wrote a program that checked whether new mail had arrived. It would tell a user You have new mail. At the time, Heidi Stettner worked in Evans Hall, prior to beginning graduate study. Heidi would bring her dog with her to class and to her office. He was a very friendly dog,

and a lot of the students enjoyed throwing a ball down the corridor for him to fetch. He even had his picture on the bulletin board with the graduate students: the legend read that he was working on his Ph.Dog. John decided to name the program after the dog: Biff. According to Heidi, John and Bill Joy then spent a lot of time trying to compose an explanation for **biff**—they came up with "Be notified if mail arrives." Biff, who died in August 1993, at 15, once got a B in a compiler class. According to Heidi, the story of Biff barking at the mailman is a scurrilous canard.

Kirk McKusick had joined the project in Summer 1981. Leffler was at Berkeley from Fall 1981, through a period of six months or so when Joy was rarely in the East Bay, but nominally still part of the Group (in 1982), to 1983, when 4.2 shipped. During this period, Duane Adams, the DARPA contract monitor for Berkeley, set up a "steering committee," comprised of Bob Fabry, Bill Joy, and Sam Leffler at Berkeley; Alan Nemeth and Bob Gurwitz at Bolt, Beranek and Newman; Dennis Ritchie; Keith Lantz at Stanford; Bert Halsted at MIT; Rick Rashid at Carnegie-Mellon; Dan Lynch at the Information Sciences Institute; Adams and Bob Baker from DARPA; and Jerry Popek at UCLA. Beginning in 1984, the semi-annual committee meetings were supplanted by a series of annual workshops (at Berkeley till 1988; at the University of Colorado in Boulder in 1990 and 1992).

It was Gurwitz's early implementation of TCP/IP that Joy worked on, much to BBN's dissatisfaction. Joy also worked on an implementation of interprocess communication in 1981. Leffler worked with him on the accommodation of the simultaneous use of multiple network protocols. **rcp**, **rsh**, **rlogin**, and **rwho** were written purely as temporary tools. They were in 4.1a in April 1982. By June 1982, McKusick had implemented his new fast file system and integrated it into 4.1a. This system became 4.1b, which was used in the operating systems graduate course that summer and autumn. Mike Karels told me:

> Robert Henry and Bob Kridle, who went off to mt Xinu, and Kirk, and all these guys who had implemented parts of the code, were in that operating systems course. I'm not sure the system ever ran, but we sure talked about it a lot.

When Joy left, Leffler took over responsibility. But he was not appointed to Joy's post and felt slighted by this. Leffler left for Lucasfilm—at first only part-time, so that 4.2 could be completed—and Mike Karels, who had been involved with the 2.9BSD release, took over the job.

4.2BSD was a great success, as McKusick has pointed out: "More copies of 4.2 had been shipped [in the first 18 months] than all of the previous Berkeley software distributions combined." Several commercial OSes were based upon 4.2: DEC's Ultrix and Sun's post-UniSoft OS were the most notable. Nonetheless, there were a lot of complaints about 4.2BSD, and Mike Karels spent most of his first year tuning and polishing.

I'll let him tell his story, too:

> I only applied to two graduate schools [in Biology]: Berkeley and the University of Wisconsin. I had played a little with computers as an undergraduate, but not much. My one class in programming was in PL/I. And when I arrived at Berkeley, they had this custom where you'd spend the first quarter in one lab and the second quarter in another, and then you'd pick an advisor. ... And in my second rotation I got involved with this group doing bacterial genetics. And at the end of the lab was this group doing data collection on a PDP-11/40 running V6 Unix. So this looked interesting and I thought that while I was doing my bacteriology project, I might write a little program. Well, the post-doc who was in charge of the computer left; and here was this guy who had expressed interest and actually written a program. So I was put in charge. And we had this undergraduate in Computer Science named Bill Jolitz who wrote software and took care of things. And every once in a while we'd get a new terminal or disks and I'd have to learn how to do something.
>
> Well, we had a disk crash and lost a bunch of the customized V6 stuff. And V7 was coming out at this time and Bill was trying to bring it up over at the Geological Survey, where he was working at the same time. So rather than reconstruct

> the V6 stuff, we got V7. And if Bill wasn't around or the device driver didn't work or it was finals time, I'd do it instead of Bill. So I got to learn mostly through necessity. But then I began working on what became 2.8BSD, along with Bob Kridle and lots of people I didn't know.
>
> So I gradually spent less and less time working on my degree and more and more on the computer. And then there was this job at the CSRG. I began on August 1 and Sam went part-time the end of the month, and then left. I guess I started two months before 4.2 came out [in 1983]. And Bill had gone—anyway, in his last year he was barely there. But, you know, about a year before I went to the CSRG, I gave my first paper [it was on **vfork** and delivered at the San Diego USENIX meeting, January 26, 1983], and Sam came up and said "Why don't you come work for me and you can work on a real machine," meaning a VAX instead of an 11/40. But it was both fun and exciting when I started on the VAX. It was a bit difficult doing debugging on a new machine.

The VAX was a real machine, but as time passed, there were other real machines, too.

CHAPTER

19

Commercial Unix

A great deal might be written about how vendors embraced (and failed to embrace) Unix. For example, Doug McIlroy commented to me:

> IBM and BTL managed the TSS/Unix marriage quite early on, but that had no effect on IBM while Amdahl promptly came on board. DEC tried to ignore Unix, and still does, Armando Stettner notwithstanding. HP, who like DEC and IBM had a proprietary operating system, took up Unix enthusiastically, as did the Japanese. And there's Sun, where Unix and hardware grew symbiotically, and which propagated BSD.

While I consider much of this interesting, as few (if any, except Sun) of these companies contributed to the development of the operating system, I will not devote much space to them.

The first company to support Unix commercially was Interactive Systems in 1977. Heinz Lycklama joined Peter Weiner, Interactive's founder, in Santa Monica the next year. Lycklama had just written LSX, a version of Unix for the LSI-11 microprocessor. (Recall, the LSI-11 was essentially a PDP-11 processor that was used in Carnegie-Mellon's Cm*, an experimental, loosely-coupled multiprocessor.) Interactive's product was called IS/1 and ran on most PDP-11s. Up to a few years ago, when it became a trading chip, Interactive was still one of the foremost porting companies. Interestingly, earlier, while at the

RAND Corporation, Weiner had been the first person to acquire a commercial Unix license.

The first commercial Unix "clone" was Idris from Whitesmiths, Ltd. Whitesmiths was the creation of P.J. Plauger, another former Bell researcher. (Plauger says that Idris was a Babylonian god of the arts; Whitesmiths was a software company, and thus the opposite of a "blacksmith.") Plauger had graduated from Princeton (where he was a year ahead of Stu Feldman) and gone on to earn a Ph.D. in Physics from Michigan State in 1969. He told me:

> Research didn't look very interesting at the time. So when they gave me my choice, I had picked a group in Holmdel. Its charter was to digitize the phone system. It was an eternal battle, because every solution meant sending more current through the wires, which was more than the phone company was willing to pay for ... I was shipped all around Bell Labs for a couple of years. And among other things they loaned me to Murray Hill. And just because of tight space, I ended up in an office right next door to Brian Kernighan. Down the hall from people like Doug McIlroy and Dennis Ritchie and Al Aho. Brian and I were really very good at drinking coffee and shooting the bull several times a day.
>
> We were commiserating about the horrible state of style in programming. And we just sat down and in about three months we wrote what became *The Elements of Programming Style* [1974]. Having done that, we got some support within Bell Labs. And that inspired us to write *Software Tools* [1976]. But several months into that project, I ended up leaving Bell Labs. I felt that I didn't have a future there and that I'd better move on before they asked me to move on. And I was able to get a job at Yourdon, Inc....
>
> *Software Tools* was the hardest thing I've ever done in my life. Harder than my doctoral dissertation. And less satisfying. So I'm pleased that it came out as it did.... Brian and I get along like Gilbert and Sullivan: what we do together is better than what we do separately....

> Well, after a few years of travelling all over the world lecturing, I felt that I wanted to get back to programming. Ed [Yourdon] had an opportunity to get a contract to write a commercial C compiler and I talked him into doing it. I worked around the clock for a week. It wasn't the greatest compiler in the world, but it was adequate. And along the way, I learned where the market was, because people would come to me and say "We don't want all that Unix stuff, we just want C."
>
> Ed understood the seminar business, but not the software business. And I decided to quit and start my own company.... When I was ready to leave, my wife suggested that going around and giving seminars wasn't all I was interested in. And these two guys who were working for me started hanging around. So I formed a company—incorporated in Delaware—and these two guys were partners with me, with me holding the majority. A three person company. I think we started on August 1st, '78. We were going to sit down and write a C compiler from scratch—my third C compiler, I guess. I paid a lot of attention to not having any notes from my Lab days or my Yourdon days.

I noted that to me the most interesting thing about the Whitesmiths C compiler had been how meticulously it had been "incompatibled." Plauger laughed.

> No, no. In hindsight that looks worse than it ever was. In actuality, in those days there was no clear notion of what went into C libraries, nor a clear notion of the C language outside of the PDP-11. I gave it my best shot, to be honest. If I made any mistake it was that when the dialects emerged around C, I should have gone with the crowd, and given away my C libraries....
>
> Anyway, I wrote like a fiend, and by the end of November we had a compiler. But we really lucked out in that we had sent out a sort of birth announcement about the company—that we were going to do custom C compilers—and we got this phone call like a week after we were incorporated,

> from a company in Philadelphia. They said, "We see you're doing C compilers. We're in the market for one. Can you come down to Philadelphia?" But I said, "No, I'm too busy. You wanna talk, you come here." And they did! They sent three people up [to New York] and we sat in this little apartment on West 79th Street ... and we told them what we were planning, and they said "We want what you've got. We want everything you say you're going to do for the next three years." So I said, "Well, you could make us regular monthly payments, that would make life much easier." And they did! And within three or four weeks of starting the company, we had a major contract, from Fisher and Porter.... We gave them a PDP-11 compiler by the end of the year; we gave them an 8080 compiler by the middle of '79; a VAX compiler by the end of that year; and we gave them a 68000 compiler in the middle of '80.
>
> And we were doing Idris at the same time. We wrote a sizable chunk of code. Whitesmiths grew like a weed. We started with $25,000 of my savings and within six months we had a positive cash flow. It was a heady time. It was fun. Everything was fun until the end of '81.

In 1982, Whitesmiths went through a struggle for power and lost ground to the PC C compilers that began to arrive. Plauger stuck with Whitesmiths until 1988, when he sold it to Intermetrics. He said:

> I spent ten years learning how to run a business; and when I got good at it, I realized I didn't want to do it. And I've been on my own for the past four years.
>
> But I really wanted to sell Unix. I spent some time talking to the Bell lawyers before I realized that because of the consent decree, they just weren't going to do it. Marketing was a dirty word, a dangerous dirty word. And I had to decide whether I wanted to fight to sell Unix or whether I'd just bite the bullet and knock it off. It wasn't an easy decision. My goal was to fill the lower niche. But I didn't do the marketing or line up people to buy into it....

Strangely enough, when Greg Rose left UNSW in 1979, his first employer was a company that had licensed the Whitesmiths C compiler, and when it went belly-up, Rose and some of his friends continued to pursue the Whitesmiths ideas.

In some way, the Wollongong Group was the first Unix company in Australia. The tale of its founding (and loss) is so strange that it is worth reading. Recall, when Rich Miller and his group ported Unix to the Interdata at the University of Wollongong, the university had no tape drive. Rose told me:

> They really wanted a tape drive. So, Juris Reinfelds got on a plane and went to New Jersey. I think he was going to a conference. He really thought they had something—Unix on the Interdata—Perkin-Elmer by that stage—and he tried to sell it. He actually got to talk to someone in the company. And this guy said, "How much do you want for it?" And Juris said they wanted a tape drive, even a used tape drive. So the guy handed Juris over to an underling who listened and said "No." Well, Juris was on his way back to Australia. Really unhappy. And he was unloading his troubles in a bar in the Los Angeles airport to some doctor. And the chap, Dan something, said, "Well, how much does a used tape drive cost?" And Juris said about $14,000. And Dan said, "I'll buy the rights off you." And they made a deal and he wrote out a check and founded The Wollongong Group. Later he bought out Rich Miller and Juris. And Dan went off and sold software rights and ongoing maintenance to Perkin-Elmer for $100,000. And that's how the company got started.

Most likely no stranger than other tales. But one wonders who those folks at Interdata/Perkin-Elmer were who rejected both first Ritchie and Johnson and then Reinfelds. Greg went on:

> It was just about that time that the University of Melbourne got Perkin-Elmer Unix. And that's how Robert Elz got involved. For years every AUUG meeting would have a paper by Elz on tty drivers or something. And some of the stuff was

> really important and made it back into Berkeley Unix. Elz spent time at Berkeley. 1979-80, I think.

Among other things, Elz wrote BSD disk quotas (which allow administrators to control the number of blocks allocated to any user) and the autoconfiguration files (which find and configure peripherals that are attached to a processor). Elz' paper was given at the USENIX Conference in Salt Lake City in June 1984.

The final early commercial Unix venture I want to look at is the Canadian entry: Human Computing Resources Corporation. HCR was founded by people from the University of Toronto, and its story begins there. David Tilbrook, student at Toronto, founder of HCR, and program chair of several EUUG and USENIX conferences, reminisced about the group of students with me:

> It was a peculiar group, part of the Dynamic Graphics Project of Ron Baecker, the only person who had started out to do graphics was Bill Reeves. Duff was going to do computability with Steve Cook. I was going to do algebra with John Lipson. Lipson said why didn't I reap the benefits of all the practical experience I had, have fun for my master's year and work hard on my Ph.D. So I decided to go off and do computer graphics. I had taken a graphics course, because U of T required breadth. So we ended up with this computing group, all working for Baecker: me, Bill Reeves, Tom Duff, Martin Tuori, Mike Tilson, Tom Horsley. Rob Pike came along a year or so later. A fairly illustrious group in many ways.
>
> This was '75 and the *CACM* article had come out. We were doing graphics on the 370, using a package called Zap. It was in FORTRAN. There was a library of sorts and you used to write PL/I programs that linked in to these FORTRAN subroutines that produced paper plots on a Calcom plotter. We also produced paper plots which could be taken to the GT-40. In the graduate course that I had done with Baecker, we had to produced an animated short. [Tilbrook did a sorting algorithm.]
>
> At the same time I was working on a contract for the Toronto Public Library and had come in contact with UTLAS [the University of Toronto Library Automation System]. So I

had access to the XGP printer—serial number 3. And this was the first time I had contact with a printer with variable fonts as well as upper and lower case. I was also looking at the problem of producing spine backs and book catalogs and card files ...

So here we were at the Dynamic Graphics Project and Tuori and I were going to do a typesetting package—at that time there was nothing. No quality product. So Martin and I through the auspices of the University and the Central Library put together a workshop on using computers to do typesetting. We sent out a flyer and we got about 20 responses.... Well, that got me into Baecker's group. That and the animated films. Well, three of us were National Research Council scholars, so Baecker had all this money to play with, and that's how he bought the PDP-11/45. Reeves did the general purpose package, and I kept on screwing things up by demanding dynamic graphics. That was when I introduced the dynamic cursor. There was a tremendous variation of styles, though Reeves, Duff and I were all dyslexic. Anyway, I liked the Xerox Sigma 7, and I used that to write **NEWSWHOLE**. I logged in 300 hours in a month.

David Slocum of *The Globe & Mail* had come to the workshop and talked about the difficulty of pagination and layout. So computerized pagination seem like a good topic for a thesis. So I went to work at *The Globe & Mail* ... and I developed **NEWSWHOLE**. Out of that work came the *Globe & Mail* connection. And I took that with me to form HCR.

In December 1975, I went to Baecker with "I can bring the *Globe & Mail* contract in." And the big question was "Here's this Unix system which runs on a Three Rivers Graphic Wonder, how the hell do we distribute it?" [*The Three Rivers Graphic Wonder is still in place in Henry Spencer's computer lab in the Ramsay Wright Zoology Building at the University of Toronto.*]

HCR was made up, originally, of a small group (to quote from the original prospectus):

> of specialists with extensive research and development experience in a wide range of computer-based services and products ...

> *computer graphics* and *interactive computing, interface design* ... One example is in the design of computer-based *text processing systems* and *page layout systems*...

The group was Tilbrook, Baecker, Les Mezei, and Tom Horsley. Mike Tilson joined them about two years later. HCR also marketed software products, including XENIX, Microsoft's Version 7 derivative. But to do what they wanted to do, HCR needed a commercial Unix license. Tilbrook told me:

> Bill said that he had called Al Arms, who was in charge of this thing at AT&T that had nothing to do with telephony, and told him that we wanted to buy Unix, this was 1976. And Arms came up with the price of $20,000, because he thought that was so high that no one would bother him about it.

Of course, as Peter Weiner at The RAND Corporation had paid, Tilbrook is exaggerating. Nonetheless, HCR became a profitable corporation. Tilbrook went on to another endeavor and Tilson became president of HCR, which was eventually bought up by the Santa Cruz Operation.

CHAPTER

20

DEC

Much of Chapter 2 was concerned with the Digital Equipment Corporation. This was because up to the Princeton effort, Unix ran solely on machines manufactured by DEC: the PDP line (especially the PDP-11) and then the VAX. But DEC itself didn't officially support Unix. It had its own OSes: FOCAL-11, VMS, TENEX. Internally, DEC had a culture for its first 20 years that Armando Stettner referred to as NIH—not invented here. The fact that Unix was a product of Bell Labs was enough to turn off DEC's engineers.

But in the late 1970s, there was a growing realization that Unix was not going away, and hardware engineers realized that accommodating Unix users was important. Ritchie and Thompson had been visitors at DEC several times. Now DEC set up a Unix Support Group—"We were a sort of skunkworks." Armando Stettner, who had gone to DEC from Bell Labs in late 1979, was an early sponsor of Usenet, an architect of Ultrix (the DEC version of 4.2BSD), and was the originator of the Unix license plate, told me:

> I was brought in as a sort of Unix champion at DEC. They wouldn't have us—the UEG, the DEC Unix Engineering Group—in Nashua. We were put in Merrimack, NH. We referred to Spitbrook Road [DEC's main site in New Hampshire] as "Saliva Creek." Once the service and support people knew we were there, they called us.

> But when Bill [Joy] came to visit, I was still working 32V. And when we got to New Hampshire [from Logan Airport in Boston] it was pretty late. And the only place open was the Denny's. What we did that night was sketch out what we were going to do. We wanted to roll up all the software things that were around, fix the configuration stuff, and put together a release. And I half-jokingly said "Why not port Unix to the DEC System 20?" And Joy said "Look, that's going to take 10 days and I'm only out here for seven." It turns out that Bill was out for 10 days. What we did was work at night on the VMS group's prototypes in the VMS Engineering Group's lab. We had Bill on the [VAX] 780, Bill Shannon was on the 750, and I was on the 730. Basically, we would get together at various points and start merging the stuff and then go back to the separate machines to do the debugging. We got an RP06 disk pack and we started referring to this stuff as 5BSD, because we were sure that the next release would be 5BSD.
>
> We made a dump tape and Bill packed up and went back to California. Shannon and I took the disk pack and copied it and brought it up on decvax—a 780, our main system. Bill Joy called a couple of days later and said, "Hey, there's going to be a lot of hassle with the license if we do another release. So why don't we call it 4.1BSD?"

For a decade it had seemed as though DEC was actually fighting Unix. Steve Johnson remarked to me:

> Of course, when the VAX came along, which clearly was a very good machine, DEC did realize, at least on the hardware side, what Unix could do for them in terms of sales. But they still weren't encouraging.

I asked Armando about the state of affairs.

> Well, after I came, the UEG did a lot of support work—we wrote device drivers and worked at the low-level: the machine-dependent part of Unix. Berkeley concentrated on the higher level. Customers who knew about us would tell their field service guys to call us. We developed an experience and a

language so that we could relate to the field-service folks. 'Cause there was a big confidence problem when it came to supporting Unix. The service guys just didn't know what to do if a machine wasn't running DEC software. And there was always the question of account management or customer satisfaction.

There was a lot of animosity towards Unix from Spitbrook, and, frankly, up and down the company ... engineering and engineering management in a big way. Cutler [Dave Cutler, one of DEC's engineering elite] had gotten hold of something—I think it was the SRI package on top of VMS—as a compromise. Cutler called Munson up, remember Unix was a real problem for DEC. At AT&T and at the universities. So Cutler called up and Munson said he'd send a couple of us down to look. So Bill Shannon and I drove down there—about a 20 minute drive. We got there and Cutler was in his office. So Bill and I sat down at a terminal, and it just didn't do things you wanted. Cutler asked us how it was, and I said it didn't work. And Cutler said "Well, thank you very much." We drove back up to Merrimack and Munson called us into his office— he was Senior Group Manager—and asked "What the hell did you guys do down there? Cutler called and chewed me out and said you were sorry excuses for engineers and he never wants to see you in Spitbrook again."

And I think Cutler's disdain has been reflected in his work ever since. Cutler was doing yet another OS based on a new architecture called Prism, not Unix, during Digital's internal RISC wars. Initially, Cutler's OS wasn't portable, but was culturally compatible with VMS. There is a lot of stuff in NT that I think can be traced to Prism. [Cutler went to work for Microsoft around 1983.]

We had a Unix products manager and one day we were walking down the hallway and I said, "Bill, you know I think it's time we did a real UNIX product for the VAX—a native VAX UNIX." We did a plan and Bill took it to Munson and said, "Well, let's see if it flies." Basically, my plan was to base our product on 4.2BSD. The fallback was to base on 4.1, if

> Berkeley didn't release 4.2. I was confident about 4.2, because we had been getting the betas and we were on top of things. But we did need a contingency, and 4.1 was it. Even though, by then, Sun had announced 4.2 before Berkeley did—though Bill Joy had only taken 4.1c with him when he left. We had found 4.1 pretty stable on decvax. And I was the technical manager.

At the January 1980 USENIX in San Francisco, Bill Munson had stated that the goal of the UEG was "to enlighten DEC about Unix and its opportunities." Armando had another form of enlightenment in mind.

> I was on the west coast for the USENIX conference in Santa Monica [January 1982] and we were going to announce that DEC had a Unix product, when Bill Munson flew out. Olsen was behind us. Apparently he took a Unix license plate that I had sent him and slapped it on someone's chest and told Munson "go out there and make a Unix product and make it right!" So Munson took the red-eye to the west coast to make the announcement.

One of Armando's most memorable "appearances" was at the January 1983 USENIX Conference in San Diego. I asked him about it.

> Bill Shannon had just bought a new Datsun, a 280ZX, and we drove to the Burger King to check it out. Bill had "UNIX" as the license plate and we wanted to do something for the USENIX conference ... everyone had had their gimmicks. So we said "Let's make a poster" and someone said "Well, let's take a VAX and a terminal and set them up in the woods." I don't remember why we couldn't take the VAX into the woods. So I suggested that we take a 780 in the lab, paint a windshield on the side of it, lean some VW tires and Shannon's license plate under the windshield and make a poster of this. Getting the permissions for this proved too complicated. So I was talking to one of our PR people and she had all these catalogs. And I was looking at these logo catalogs and I came to license plates. And

the idea hit me. Munson gave me $800 and I made an initial run of 3000 of the green UNIX license plates.

I recall standing up and saying that I wanted to announce that you could now get a Unix license from DEC. And there were all these guys from various Unix OEMs and they just went pale. And I held up the license plate. And there was laughter and applause and when it died down, there was a voice from the back of the room, "Armando, where'd you learn to do this?" It was Dave Yost. Later we got a letter from a Bell lawyer about the trademark. But they then called up and told us that they had to send the letter to protect the trademark, and could we send them a few plates. So we did.

Over the years, DEC and Ken Olsen have borne the brunt of criticism from Unix users. In 1983 at the San Diego UNICOM, Bill Munson said that Olsen had told him to "Make it clear ... DEC supports UNIX!" But Olsen is also the person who, in 1988, said "Unix is snakeoil." Armando, however, who spent over a decade at DEC as "Unix evangelist," feels that Olsen has been treated unfairly.

In hindsight, I think Ken had bad advice. Ken is the guy who said "Do Unix." Ken is the guy who said "Do the Pmax [the MIPS deal and the DECstation 3100]. Buy architecture from another company." And I think that what drove him was the data that said: people want it. You know, that "Unix is snakeoil" comment was taken out of context. Of course, it has negative connotations, but what Ken meant was "Everybody and their mother says that what you need is Unix." Everybody was peddling Unix. I can remember meeting customers who were really adamant about needing Unix, but they didn't know why. There was a lot of hype around. Ken meant, "Much the way people were peddling snake oil a century ago, now every vendor is hyping Unix as a cure for everything." That "Unix is snake oil" comment was just out of context: an analogy was being made. Unfortunately, as someone once said, Ken didn't always realize the impact of what he says.

But for the record, from someone who was there, it was Ken Olsen who pushed for Unix against his lieutenants; for the Open Software Foundation. The problem with DEC was that all the people who were running the company were engineers who had grown up; not people with business experience. And like any good engineer, they had allegiances to their product and their associates. That's what was the cause of DEC's problems in the late 80s and now in the 90s.

DEC had this series of successes: the PDP-11 and the VAXes, in a time when they had little competition. But in the mid-80s, the workstations and RISC machines competed with DEC and DEC had no competitive machine. In addition, DEC was starting to try and move outside of its traditional stronghold, the scientific-engineering-academic community, and into the IBM markets. Cutler had been working on this new machine and new operating system. We were convinced it wouldn't do the job we needed it to do. If it was successful, if priced for the IBM markets, it would be too expensive for the Unix market; if priced for the Unix market, DEC would leave money on the table when going up against IBM. And besides, from the reliable info we were getting on Sun's SPARC, the Prism machine wasn't shaping up to be fast enough. But we wanted Cutler to do his system for the VMS market and we wanted to do our system to go after the Unix market. Our system was based on the MIPS R2000 RISC chips. Ken [Olsen] was really interested!

The Pmax polarized the company. Initially even the Unix group back east didn't want to do it. We (in Palo Alto) thought that DEC was big enough to do and manage two architectures.... Some of the discussion centered about whether a computer company could be successful and not own its architecture.

We really liked the MIPS architecture and the R2000 chipset; we liked the MIPS guys; and it had a 64-bit future. What more could anybody ask for? The customers we talked to liked it!

If DEC had spent half the energy on the MIPS relationship and architecture that they did on secretly continuing the

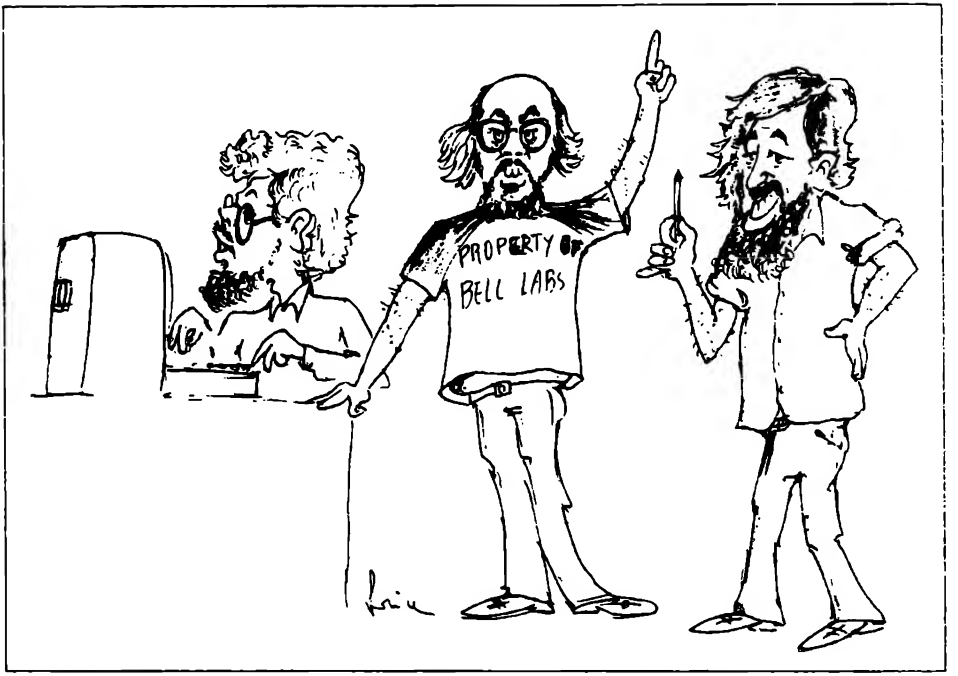
Prism stuff (which begat Alpha), who knows what the industry would have been like today?

Some of this sort of reasoning may have been exercised when Gordon Bell killed off the DECsystem-20. I think that particular system was called the Jupiter. I think it was because he didn't want a viable competitor to the VAX, getting all those TENEX/TOPS-20 customers. I'm sure that Cutler didn't want a competitor to his machines back in the late 70s; he sure wasn't interested in serving differing markets with different systems in the 80s. He was really hung up on killing Unix.

Against almost unanimous advice from the engineering management, Ken, with the Board's approval, gave us the go-ahead for the Pmax and cancelled the Prism. There was a condition: instead of the year we had asked for, we were to perform the miracle within 9 months. We did it. The *San Jose Mercury* said it all: "Digital Stuns the Valley." *Upside* mag showed a picture of Ken pulling a rabbit out of a hat with a caption reading: "Scotty, wanna see me pull another rabbit out of my hat?"

But DEC continued its NIH attitude: the Alpha chip is really a lot like the Prism chip. Ken (or the corporate culture; are they different?) just didn't know how to cooperate with another company; so they never put any ergs into making a future with MIPS. Even OSF and X/Open are problematic. Just one person's opinion...

Interestingly, once Armando and a number of others had left DEC, ULTRIX didn't progress: in 1990, DEC was still stating that "The ULTRIX operating system is a 4.2/4.3 Berkeley Software Distribution (BSD)-derived implementation. It provides many of the system and library calls that are common to AT&T's System V as defined in the System V Interface Definition...." And still later, DEC dropped ULTRIX completely in favor of OSF/1.



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

CHAPTER 21

The Law—Part II

The Consent Decree of 1956 had delivered a shock to Western Electric and AT&T. But on 20 November 1974, the US government filed a separate antitrust action against AT&T, Western Electric, and Bell Telephone Labs. This complaint "alleged monopolization by the defendants with respect to a broad variety of telecommunications services and equipment...." (Harold H. Greene, District Judge, in 552 F.Supp. 131 (1982). All following citations in this chapter are from Judge Greene's *Opinion*.)

The Antitrust Division of the Department of Justice sought the divestiture from AT&T of the Bell Operating Companies and the dissolution of Western Electric. Thanks to the AT&T lawyers "very little progress [was] made" for nearly four years, but on 11 September 1978, "the Court issued an opinion which disposed of all then outstanding legal issues and laid out the future course of the pretrial proceedings." The actual trial began two-and-a-half years later, on January 15, 1981. (Recall, in 1978 there was a Democratic Justice Department in Washington; Ronald Reagan was inaugurated on 20 January 1981, and the nature of the Justice Department changed dramatically.)

Nearly a year later, Judge Greene was confronted by a "Modification of Final Judgment" entered into by the parties to the suit on 8 January 1982. Briefly, this meant that the Justice Department's attempts at dissolution would result in a "modification" of the 1956 agreement, not a resolution of the suit. This "Modification" had been agreed to, follow-

ing a brief hearing, by a different judge, Vincent Biunno, on 11 January. The next day Judge Greene vacated Judge Biunno's order under the Tunney Act [the "Tunney Act provides that a proposal for a consent judgment submitted by the United States in an action brought under the antitrust laws may not be entered by the Court without prior compliance with certain procedures. These procedures include a sixty-day comment period...," publication of a competitive impact statement, a period for public comment, and determination by the Court that the agreement is "in the public interest."].

AT&T and the Republican Justice Department maintained that the Tunney Act didn't apply, as this was a "Modification" to the 1956 Consent Decree, not a new action. Judge Greene's response was quite blunt: "In the opinion of this Court, that reasoning may most charitably be described as disingenuous. If that reasoning were deemed acceptable, the parties here ... could subvert the clearly expressed will of Congress by a mere act of labelling."

The jockeying went on for several more months, but by the end of June 1982, Judge Greene had a "proposed decree," the key feature of which was "the divestiture of the Operating Companies from the remainder of AT&T."

I do not want to go into tedious detail here, but the result of Judge Greene's Opinion was that Western Electric was dissolved, that the various operating companies formed the "Baby Bells" (in 1984), that Bell Telephone Labs was split off, renamed AT&T Bell Laboratories (and Bellcore was formed), and that AT&T was now permitted to enter the hardware and software computer business.

This is what brought about System V in 1983, as well as a totally new set of licensing terms from AT&T. By 1987, however, AT&T faced a backlash over its terms. In 1988 it purchased 20% of Sun Microsystems and the next year AT&T and Sun brought forth SVR4, which combined AT&T and BSD features. In 1990, AT&T formed the UNIX System Labs as a spinoff to develop and market Unix.

Throughout this, AT&T Bell Laboratories was working away, developing useful and innovative software that seemed to have little to do with USL's marketing efforts. Most notable, perhaps, was Plan 9, to which I will return in Chapter 26.

The Unix Industry

CHAPTER

22

/usr/group

The voices of marketeers had never been heard within USENIX. John Bass (and others) had complained about this, but Mel Ferentz, Lou Katz, Lew Law, and other pillars of the community were steadfast. Brian Redman recalled to me that he had gone to the Santa Monica USENIX Conference (January 1979) where

> some guy got up and they booed him off the stage because he was a marketing consultant or something. Gee, I thought, these guys are serious.

/usr/group was founded in 1980 and incorporated in 1981. It is a trade association "dedicated to the promotion of the UNIX operating system." In August 1989, the membership voted to change /usr/group's name to UniForum, which its large spring vendor exhibition had been called.

Bass's view of the formation of the USENIX Association and the subsequent creation of /usr/group offers a different slant from those of others. He told me:

> USENIX officially got its start at the New York meeting after we were floating several trial charters about for discussion. Mel and Lou, while a number of us were sleeping off a near all night hallway discussion, held an unannounced floor vote to nominate and elect themselves the nominating committee to find canidates for the first board. At the next meeting in Santa

Monica as I remember, we discovered they nominated themselves and presented an entirely different charter for floor approval without any prior discussion,... and with a slick railroad approach took charge over a number of protests regarding their lack of accepted procedure in this matter.

They wanted to have control of USENIX rest securely in the hands of University Unix Data Center Managers with source licenses ... and thus the original board was made up of three such with two other positions, one commercial and one government. With the voting membership restricted to holders of source licenses, they successfully stacked both the board and eligible voters in their control. The other two models that were widely accepted were that of a professional org (i.e. ACM/IEEE) where there was no restriction to membership and one person/one vote ... and per Unix site either binary or source (since we did not see any need to limit the scope to source sites only). Mel felt this was his baby, just because he had held the first East coast meeting, and refused to release control. Lou and a select clique from the East coast just tagged along. Another important aspect of the sample charters being discussed was support for local chapters to allow more students and others without a travel budget to join in and learn. After Mel's second surprise vote, he told those of us who had a strong local group "Tough ...": accept his vision of USENIX or just become a "splinter group."So we backed off and attempted to work within this new framework which became increasingly hostile to any non-"University Unix Managers club" input.

A year later it became clear that USENIX formed from this restricted club would not service the rapidly growing number of binary users ... so a group of us led by Bob Marsh (founder of ONYX and PLEXUS) met in Santa Clara and formed /usr/group under the principle of one member one vote.

If Mel had allowed a one-person one-vote professional org charter like the ACM/IEEE model we wanted, /usr/group would not exist today. I suspect that the polarization and

disharmony caused by the existence of both USENIX and UniForum would have never occured either.

In January 1983 (San Diego), January 1984 (Washington, DC), and January 1985 (Dallas) the USENIX Conferences were held jointly with /usr/group. January 1987 (Washington), January 1988 (San Francisco), January 1990 (Washington), January 1991 (Dallas), and January 1992 conferences were held in the same areas, but different locations (with shuttle buses between the venues). Over the years, /usr/group was a strong supporter of standards (see below) and the USENIX Association the even stronger supporter of research and development. The total separation in time and place of the technical conferences and vendor shows is a lamentable example of the totally different outlook of the technical and manufacturing-marketing groups, despite the obvious fact that neither can exist without the other.

The research/vendor split that was the immediate cause of the formation of /usr/group wasn't there in Japan. Erik van der Poel told me

> The Japan UNIX Society (jus [purists write the abbreviation in lower case]) was set up by Kouichi Kishida, who is from SRA, an independent software house. I think it was basically set up for fun, and UNIX was seen more as a research area than the commercial thing it has now become. Since it was for fun/research, it was OK for university and corporate people to do it together. (Of course, Japan has always had the reputation of enjoying good cooperation between government, academia and industry anyway.)
>
> Gradually, the commercial aspects of UNIX came to light, and I think that's when the two committees were born: the so-called T Kanji Kai (Technical Committee) and B Kanji Kai (Administrative Committee [though the B originally came from "Business"]). Apart from these two committees, there is also the governing Board and a Secretariat that hangs off that.
>
> I was on the T Kanji for some time. The T and B Kanjis often met at separate times, and sometimes together. There was no antagonism between the two, but you could sense a clear difference in attitudes and interests, in my view quite similar to the USENIX/UniForum relationship.

The "separation"may be coming about now. As Toru Takahashi put it:

> When Jun Murai, Koichi Kishida and Nobuo Saito established the Unix study group, it was a small group of both academics and vendors. But, two years ago, some vendors and software companies set up the UNIX Business Association (UBA), and UBA is becoming the business representative group of UNIX. As I was a member of the jus board of directors, I was concerned about UBA. In jus, the business concerns have not been powerful. But UBA is growing while jus is remaining a technology-oriented society.
>
> Did you know there was no official relationship between jus and JUNET? Whenever I have been asked by outsiders, I have replied there was no relationship. Jun Murai is the originator of jus, JUNET and the WIDE internet project, but his friends in jus followed to him to make JUNET. The same persons in jus moved to organize and develope JUNET. Right now, JUNET no longer exists. It split into commercial and regional academic/research networks.

Van der Poel added:

> Actually, the whole thing was and is really very cosy. In those days, it was jus and JUNET, and some of the big names were SRA, Tokyo University, Tokyo Institute of Technology, Keio University and of course Nihon Sun, Nihon DEC, and Sony. These days, JUNET seems to be fading away, since WIDE and more recently IIJ appeared on the scene (as a result of the shift from UUCP to IP).
>
> So we have:
>
> | | |
> |---|---|
> | JUNET | private UUCP, later IP (academic, corporate and gov.) |
> | WIDE | experimental/research IP |
> | IIJ | commercial IP (others are SPIN and IIKK) |

JUNET was set up by Jun Murai. I'm not sure what JUNET "officially"stands for, but many people jokingly say

that it was named after Jun, or that it could mean Japan University Net or Japan UNIX Net.

Murai teaches at Keio University. He began to connect Japan's computers in 1984, he claims his ultimate goal is "to connect all the computers in the world."

CHAPTER

23

Sun and JAWS

The first attempt at genuinely Unix hardware was the ONYX. It was announced at the USENIX Association conference that both Mike O'Dell and Neil Groundwater mentioned. ONYX had been founded by Bob Marsh, who would go on to start /usr/group. John Bass was working for him at the time. He told me:

> The Z8000 ONYX was hardly a VAX or on a chip. The system we took to Boulder was on three boards approx 15" x 22" (I don't remember the exact foot print size). Its performance and architecture was more like a PDP-11/45 or 70 ... segmented memory, no paging. The LSI 11/73 that was released slightly later was probably faster and much cheaper. The LSI 11/23 was a little slower and a lot cheaper (my home system at the time, which was later upgraded to an 11/73).
>
> That aside, the ONYX was the first table top system designed to run Unix. With 8 serial ports (users) and at under $25k it made a short term alternative to PDP-11 Unix systems.
>
> The biggest problem with building systems at this point in time was disk drive technology ... it was large and expensive. The real innovation of ONYX was use of the 8" 20-60mb IMI drive and 1/4" streaming tape ... this combined package made the product. In fact, the reason ONYX existed was that the venture capitalist (Carl Berg) had funded IMI to design the drive, then needed a market to sell them ... Carl had to then

fund three computer companies to use the drive to protect his investment in IMI. ONYX, Corvus, and one other company I don't remember.

The funny part is that a friend of Carl's that owned a computer store kept telling Carl that there was no market for Unix, and that ONYX should focus on CPM/MPM Z80 systems (the I/O controller for the C8002 and the bootstrap cash flow machine). In the summer of 1980, Carl removed the founding ONYX team (all the Unix people) and refocused the Company on CPM/MPM ... a year later when IBM released the PC with PCDOS/MSDOS the CPM market died in less than six months. Carl/ONYX were forced to rebuild its Unix team and refocus on Unix to save the company ... but it was too late.

In the meantime, Xerox had developed the Alto, a "personal workstation" which actually resembled what we think of as a workstation, today. *Inter alia*, it had a bitmapped screen and a mouse. It was relatively expensive (it was never on sale, but estimates were in the $30,000 range), but when the Motorola 68000 CPU came out, Andreas Bechtolsheim and other students at Stanford were able to emulate it cheaply. Stanford licensed a single-board called the Stanford University Network board, SUN. A number of companies licensed it: Codata, Fortune, Dual, Cyb, Lucasfilm, among them. Machines—Just Another Workstation, JAWs—began appearing, some costing as little as $10,000; all running either 4.1BSD or System III. Most of the companies went the way of most startups. Sun Microsystems, which hired the architect of the original board (Bechtolsheim) and the chief architect of 4.1BSD (Joy) survived. Boggs and Metcalfe had invented the Ethernet at Xerox in Palo Alto in 1976; their article appeared in the *CACM* that July; Xerox published the specifications in October 1980. The notions of desktop machines and networking were alive in Palo Alto.

The June 1982 issue of *;login:* carried the following:

Interesting Developments

Bill Joy of UCB Moving to Sun Microsystems

Bill Joy has decided to become involved with a new startup company and will be phasing out of the Computer Systems Research

> Group at Berkeley over the next few months. He will be joining Sun Microsystems, Inc., a company whose founders include Andy Bechtolsheim, the designer of the Sun workstation. SMI is one of a small number of companies which plan to offer microprocessor-based networked workstations running 4.2BSD software ...
>
> While SMI may need to develop proprietary software in certain specialized areas, Bill expects fixes to the shared base of 4.2BSD programs which are made at SMI can be distributed by Berkeley. The current cooperative efforts between CSRG and various industrial groups are seen as a model for the relationship....

The article was signed "Prof. Bob Fabry."

On Tuesday, July 6, 1982, at the USENIX Conference in Boston, Joy gave talks on both 4.2BSD and on Interprocess Communication. (Leffler and McKusick gave talks on other aspects of 4.2.) The next day, Andy Bechtolsheim spoke about "The SUN Workstation."

He told an audience of over 1,000 that the workstation was

> a graphics-oriented personal computer running Unix. It was designed to be used as a general purpose work station in a distributed computing environment. The user interface consists of a 800x1024 pixel bitmap display with a 'RasterOP' mechanism for high-speed display updates and an optional mouse. The processor is a Motorola 68000 running on the Intel Multibus. It will be upgraded to a 68010 when it becomes available. (The 68010 will allow full virtual memory capability.) An optional local network using Ethernet is available. A group of Workstations may be networked to provide sharing of, for example, the file system and/or peripherals such as printers, tapes, etc.
>
> The Workstation currently runs UniSoft's Unix V7, which includes some of the Berkeley enhancements. It will be upgraded to 4.2BSD when that system is available. This will allow Workstations on a network to run without a local disk ...

At that same meeting, Jack Test of MIT discussed the "NUnix Window System," Rich Fortier and Tony Lake of BBN spoke about the BitGraph—BBN's intelligent bitmap terminal, and Rob Pike (of Bell Labs) talked about "Merging Bitmap Graphics and Unix."

In December 1982, Hewlett-Packard announced its new 9000 series of 32-bit workstations. This was followed by the announcement that HP would support Unix on the 9826 and 9836 desktop models, which were based on the Motorola 68000 chip. The operating system (called HP-UX) was largely derived from System III, but contained a number of extensions, including Ethernet compatibility and the Berkeley vi screen editor.

In January 1983, at UNICOM (the joint Software Tools, USENIX, and /usr/group conference), Tom Lyon and Bill Shannon (both now at Sun Microsystems) spoke about porting 4.2BSD to the Sun. It wasn't really 4.2, it was 4.1c, but it was BSD and it did run on the 68000. At the same meeting, Eric Shienbrood (who had written more) and his colleagues talked about Unix on Apollo computers—this was AUX, an *emulation* of Unix System III. Later that year, IBM announced the availability of Unix on its PC, which was built around the 16-bit Intel 8088 chip (the port had been done by Interactive).

But the most important things were the birth of the high-speed workstation coupled with the Unix operating system. This led directly to networking, which could not have come about in its present form without workstations running Unix. Apollo's Domain software, for example, was in many ways better than the Internet Protocol or Sun's NFS software. But it was not open, so it lost. Unix was a functioning system not only on university campuses, but in the scientific and design labs and in commercial institutions.

CHAPTER

24

Standards

The pressures of the market and the requirements of the user drove the impulse to establish a standard for Unix. In less than a decade the marketplace had proliferated. In the decade to come, many companies were born. Most vanished, but some survived. The proliferation of incompatible hardware and software was a major drawback in a marketplace that was increasingly multi-vendor, unlike the period from 1956 to 1978, when nearly every computer center came in a single shade—usually blue. The microcomputer and the workstation—Apple, the PC, and Microsoft—were influential in changing this.

There are two streams here: system standardization and language standardization. The second of these (the standardization of C) is complete: The American National Standards Institute formed committee X3J11—The C Programming Language—and managed to issue ANSI Standard X3.159-1989 which, in turn, has become an international standard, ISO/IEC 9899:1990.

The operating system has travelled a more tortured path. Unix standardization began with a proposal from Heinz Lycklama to the Board of /usr/group that a Standards Committee be formed. By 1984 the committee's efforts had produced a /usr/group Standard that was adopted by the membership. Though it had no official status, the /usr/group Standard was very influential: when ANSI X3J11 was formed, it used this Standard as the basis for its C Library section. However, because of the divergent developments of System V, 4.nBSD, and XENIX, the /usr/group Standard was soon inadequate.

In early 1985, the /usr/group committee was merged with the newly formed IEEE POSIX Working Group (Institute of Electrical and Electronics Engineers, Portable Operating Systems), and the /usr/group Standard was adopted as a first draft.

By 1989 POSIX had spawned 10 subcommittees; four years later, there were over 20, though only two standards—admittedly the most important ones—had been produced (POSIX.1 and POSIX.2). There is also a POSIX.0 document: *Guide to POSIX Open Systems Environment*, which attempts to explain applications portability.

It is, however, important to realize that POSIX is not Unix. It is an interface, not an implementation. Fred Zlotnick has referred to POSIX as "a generalization of Unix." POSIX.1 has taken a number of Unix concepts and replaced them with abstractions, thus rendering a number of otherwise incompatible systems "POSIX-compliant." As standardization has become more and more a marketing-driven, rather than technology-driven process, it is difficult to forecast its result. The raft of industry consortia publishing standards does nothing to encourage the user.

For example, in January 1985, AT&T published its *System V Interface Definition* (SVID), a "copyrighted, non-proprietary open systems document." A new edition was published five years later. In the interim, X/Open (which had been formed in late 1984), had published its *Portability Guide* (XPG), which has now gone through several editions: XPG2, XPG3, XPG4. XPG3 and XPG4 go beyond the SVID in several respects.

88open is "a community of companies and individuals dedicated to creating a multivendor, open computing environment based on the Motorola 88000 RISC processor family." It was formed in 1988. The consortium publishes a *Sourcebook* and an "Open Systems Reference Guide," *The World of Standards*. I will look at both the Open Software Foundation and UNIX International in a later chapter.

At the 20th Anniversary of Unix conference, Ritchie remarked about both the C standard and POSIX:

> We have had rather little to do with the Unix standardization efforts. The biggest interaction was to try to make some improvement in (or carp on) the POSIX shell.

Similarly with the C standard. I deliberately left X3J11 alone. I did have some behind-the-scenes interaction with Larry Rosler and Dave Prosser when they were the redactors of the draft and the AT&T representatives to the committee, but with the notable exception of the noalias affair, in which I was very vocal, I didn't take part...

I think [the C Standard] is a good job. In fact, almost a model for a successful standardization effort. They took something that was basically not broken, but in serious need of clarification and updating, and clarified it and updated it without breaking it. There are certainly many criticisms that can be made... But still, when you compare what happened with other languages, like FORTRAN 8X and even Pascal when standards bodies try to define languages, the fate of C was highly favorable.

Ritchie was far less enthusiastic about the POSIX effort:

This is a more mixed bag. The system call interface and the libraries and even the commands seem reasonably solid, and not badly controversial or messed up. The industry seems to be pretty well behind the result. On the other hand, these parts lack some fairly obvious things, like a coherent approach to networking. Presotto and I talked about [an approach] at USENIX a few years ago... However, our approach doesn't seem to have taken over the world yet.

Also, the IEEE POSIX effort seems to be sort of freaking out in ways that are dismaying to a lot of people. In particular, the real-time group seems to be gathering all sorts of undigested ideas that don't seem to fit together with the existing system very well. I'm worried about the thought of tasking, in which several small processes execute in the same memory image, because the model on which Unix and its software are built don't really accommodate it very well—C doesn't support it, the libraries don't. Multiprocessing has lots of implications about the way you design software, and I fear that just shoving it in will lead to a mess.

To say I am ambivalent is an understatement! The tension (psychological, political) generated over standardization is tremendous across many dimensions.

The usual one mentioned is the struggle of innovation vs. standardization. I don't think this is the real problem. If there are standards, you can make a calculated decision whether to live with them or to do something else. This is a gamble, but a reasonable one. The real tension is deciding whether a particular area should be standardized yet. It really is quite awful when a single standard should have been made but wasn't (VHS vs XBeta).

One thing that people should realize is that lots of standards in areas relevant to us are made by quite small groups, and if you care enough and work in the right way, they can be influenced. In the formative stages, a properly posed, concrete suggestion can go a long way. This is part of the paradox, of course. It means that a few good people can really save the day, but it also means that a few idiots can mess things up for years to come.

Since the time of Ritchie's comments, the number of POSIX subcommittees has waxed and only a few of the then-extant proposals have proceeded to ballot. Further, as more consortia are formed, the number of informal standards that fail to mesh with what the various IEEE committees are doing has waxed, too.

The Currents of Change

CHAPTER 25

Duelling Unixes

Up to 1978, Unix meant AT&T's operating system. It was only with the advent of PWB 2.0, V7, 32V, and 3BSD that confusion set in. But Unix was not an AT&T OS product. It was a "telecommunications support tool." AT&T, recall, felt that because of the consent decree it was constrained to stay out of the computer business. It wasn't permitted to compete in the computer marketplace. The future of Unix within AT&T was handed off from Research to USG—the UNIX Support Group, which controlled its future for most of the 1970s. Then, in 1979, Microsoft and the Santa Cruz Operation came out with XENIX2 and Berkeley with 4BSD, each a V7 derivative. XENIX was the first implementation of Unix for the Intel 8086 and many other architectures. It remains a very popular implementation, though it has become increasingly incompatible with others. Len Tower of the Free Software Foundation remarked to me, "Using SCO Unix is like travelling back in time."

As Andy Tannenbaum pointed out, "Within BTL there was a UNIX SUPPORT GROUP, but you never heard about the UNIX DEVELOPMENT GROUP. This is because no Bell System organization had the charter to develop computer operating systems..." In 1981, UniSoft (founded by Jeff Schriebman) brought out a port called UniPlus+, which has remained compatible with System III and now System V. In 1983 a group of Berkeley-ites formed mt Xinu (*Unix tm* backwards) to commercialize and support BSD. Among these were Bob Kridle, Alan

Tobey, Ed Gould, and Vance Vaughan. Debbie Scherrer soon joined them. They marketed first more/4.2BSD and then more/4.3BSD. (mt Xinu excelled in customer service: Debbie installed my 4.3 tape herself in 1986.) Also in 1983, USG and the PWB group were merged into the Unix System Development Lab.

Within five years, Apollo, DEC, Eakins, Gould, Integrated Solutions, Masscomp, mt Xinu, NSC, and Wollongong were among the companies marketing Berkeley Unix. Among those marketing AT&T System III or System V derivatives were: 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, and Wollongong. Furthermore, Amdahl, Apollo, Apple, Cray, DEC, Data General, HP, IBM, Intel, Motorola, Unisys and a host of others offer proprietary versions of Unix, several of which are 4.2BSD-based.

All of these, whether AT&T or BSD-derived, require licenses from AT&T. Recently, several versions of Unix that do not require such licensing have become available. Though none is a truly robust, commercial product, BSDI, 386/BSD, and NetBSD run on any 386/486 machine. [These 'license-free' versions, all derived from the free CSRG releases, were still under litigation when I wrote; this changed on 4 February 1994; see Chapter 29.] Linux, written by Linus Torvalds, also runs on 386/486 machines and uses no CSRG code. All four systems employ many of the programs that the Free Software Foundation (FSF), founded by Richard M. Stallman, has written for their near-finished GNU ("GNU's Not Unix") system. GNU programs include original, freely redistributable recreations of most Unix software (they are distributed under a scheme called "copyleft" by those involved with the GNU Project. This means that both the original and any improved versions must remain distributable and modifiable by all and that source code must be distributed, if binaries are). gcc, GNU's C compiler, is probably the most important successor to Steve Johnson's pcc.

Finally, various of these derivatives and clones run on a catalog of computer chips: DEC, Intel, MIPS, Motorola, NSC, to name a few.

The result of this thick sludge of alphabet soup has been confusion on the part of the users and prospective purchasers: what will work with what? Certainly, despite all the to-do, there are no "open

systems" nor has the standards process resulted in any sort of interoperability. On the other hand, when you understand BSD or SVR4, AIX or HP/UX, Solaris or Ultrix, you are 90% of the way to understanding any other variant. And the language of program development is ISO/IEC C.

Turning tables, Doug McIlroy asked me:

> Do you have any insights into why Unix, which began tinier and far more powerful than MS-DOS, never made it in the PC world? And why is Unix so fat today?

Armando Stettner's response (to me) was that Unix had never had "Microsoft's marketing" and that "people with real work need many more things than V6 or V7 had to offer." I think that one of the reasons that Unix didn't make it into the PC world was that until very recently it had no user interface that the non-professional could use with ease. (Though many of my programmer friends will aver that the shell is friendlier and more flexible than MS-DOS.) When IBM came out with the PC running MS-DOS and when Jobs and Wozniak brought out the Apple (and then the Macintosh), people who had never sat down in front of a computer felt they could interact with these machines. (They weren't all successful, recall the Apple Lisa in 1983.) And as these weren't people who had been working with big machines on university campuses or at research institutions, they didn't know what they were missing, though the MS-DOS interface is in some ways worse than any Unix shell, with genuinely inferior error messages.

I don't know whether Microsoft's investment in XENIX was a ploy to *prevent* Unix from becoming successful in the marketplace, but it certainly did have that effect.

As to why Unix has waxed the way it has, I think that's easy. If we think of Ken and Dennis, Bill Joy and Kirk and Sam Leffler and Mike Karels as "arbiters of taste," [Stettner] then there is none now among the Unix vendors. When System V adopted a number of Berkeley-isms, instead of integrating them into the system, AT&T's Unix Systems Labs (which was bought up by Novell in 1993) merely taped the code together. There was also an attitude of doing everything the user requested. Want an editing environment? Here's vi (Bill Joy) and

emacs (Richard Stallman). Want a windowing system? Here are the X Window System and both the Motif and the OpenLook user interfaces. Need a shell? Here's sh and csh and ksh. At the time, there was a rumor that USG had been told to include BSD features in System V in order to demonstrate openness. I have been unable to confirm this. Each of these features takes up space; each requires memory; each is written in thousands or tens of thousands of lines of code. A slim, lightweight system has fewer alternatives and fewer utilities.

CHAPTER

26

Offspring Systems

Just as early versions of Unix gave rise to MERT, PWB, LSX, and Idris, V7 and 4.xBSD were progenitors of a large number of systems. Running the alphabetic range from Amoeba to V, and the geographic from Jerusalem westward to Taiwan, they are too numerous to list here. However, as instances of the sorts of development of the past decade, I will conduct a whirlwind tour of five in this chapter: MOSIX (Israel), Chorus (France), Amoeba (Netherlands), and Mach and Plan 9 (US).

MOSIX (Multicomputer Operating System for UNIX) is a distributed operating system that integrates a cluster of loosely connected computers into a virtual single-machine Unix environment. MOSIX was begun in 1981 by Amnon Barak at the Hebrew University in Jerusalem. Originally, called MOS, it was designed for a cluster of PDP-11/23s and was based on Version 7. During 1983-84 a version was developed that ran on the M68000 chip. As of 1993 there have been four successive versions of MOSIX, including a 32V version (1987, called NSMOS, because it ran on the National Semiconductor NS32332-based computers), a VAX version (summer 1988, based on SVR2), and a VME532 version—this most recent one running on a cluster of multiprocessor workstations. The unique properties that were initially introduced in MOSIX were load balancing by dynamic process migration and the use of probabilistic algorithms to allow scaling.

Chorus began as a research program at INRIA in 1979, initiated by Hubert Zimmerman. Up to 1986, three communications-oriented

kernels for distributed systems had been developed by a rather large team. These early versions all had a custom interface. Chorus-V2 (1986) was compatible with Unix System V. Around this time the project was spun off as a commercial enterprise: Chorus Systèmes.

UNIX System Labs announced in 1993, prior to its purchase by Novell, that it would be using Chorus' technology. It must be pointed out that, as a direct result of the openness of the Unix research community—as I have repeatedly mentioned in previous chapters—there is a good deal of "sharing" among the various systems and projects itemized in this chapter. Chorus-V3 thus has distributed virtual memory and "threads" similar to Mach, network addressing like that of Amoeba, and uniform file naming as in Ninth Edition Unix. As in the late 1970s, this is a strength, not a weakness.

Chorus consists of a nucleus and several system servers, which cooperate in the context of subsystems. One subsystem implementation is Unix. Chorus Systèmes has marketed a number of useful and interesting variants of both the nucleus and the Unix subsystem.

In the mid-1980s, Andy Tanenbaum at the Vrije Universiteit in Amsterdam (who had written MINIX) theorized that the characteristics of computing in the future would involve the need for physically distributed hardware and the need for logically centralized software. With this in mind, the Amoeba project was begun at VU. Subsequently, Sape Mullender and others at the CWI (formerly the MC—Mathematische Centrum) teamed up with VU to develop Amoeba jointly. The Amoeba architecture consists of:

- Workstations with a window system
- Pool processors for computing
- Specialized servers (file and directory)
- Gateways to other systems

The Amoeba software is object-based; there is a Unix emulation package, among a variety of other things. Amoeba was designed with the notion that a collection of machines on a local network should be able to communicate with a similar collection of machines remotely over a wide-area network. As wide-area networks are typically slower and less reliable, Amoeba's primary goal in networking was to achieve transparency without sacrificing performance.

As a result of Tanenbaum's unhappiness with AT&T's licensing in 1978, Amoeba was written from scratch and neither the kernel nor the utilities contain any AT&T or Berkeley code, obviating the need for either license.

Another outgrowth of Unix into the multiprocessor world is Mach, invented at Carnegie-Mellon University. Originating from work by Avadis Tevanian, Rick Rashid, and a group of their colleagues, Mach targeted a broad range of computer architectures, including uniprocessor, multiprocessor and distributed systems. The intention was to provide a compact, efficient kernel on top of which interfaces for a variety of operating systems could be layered. The file system used by Mach grew out of the one maintained at CMU as part of 4.1BSD in 1982. The kernel grew out of Rashid's work on Accent (a "communication oriented network operating system kernel") in 1981. By 1990, Mach provided full backward compatibility with 4.3BSD. Mach was designated the kernel upon which OSF would build its future OS in the winter of 1989-90.

In 1987, Rob Pike and Ken Thompson started designing a replacement for Unix. Dave Presotto, Tom Duff, and Howard Trickey at Bell Labs joined in rather quickly. Called Plan 9 (after *Plan 9 from Outer Space*, arguably the worst science fiction film ever made), the system is three-tiered, with communal servers at one end and terminals at the other. The servers are multiprocessor machines; the terminals are basically diskless workstations. The third tier is a wide-area network, connecting the terminal and server networks. The Datakit network of Greg Chesson, Sandy Fraser and Dennis Ritchie is the long-haul network used. Plan 9 runs counter to the current trend in computing environments: from the advent of the Sun 3 in 1986, the "majoritarian" view has been of workstations with ever-larger disks connected in local networks. Plan 9 is a distributed computing environment assembled from separate machines acting as servers, terminals, etc. The various pieces are connected by a single protocol and local name space operations.

Plan 9 has been implemented on notebook computers running the i386 as well as on large servers. In late 1993 it was handed over to a commercial vendor, Dave Presotto told me, adding, "We're starting on a new system —got any bad movie titles for us to use?"

CHAPTER

27

OSF and UI

In late 1987 AT&T announced that it had purchased a sizable percentage of Sun Microsystems and that Sun would receive preferential treatment as AT&T/USL developed new software. Sun announced that its next operating system would not be a further extension of SunOS, which was derived from Berkeley Unix, but would be derived from System V, Revision 4. A frisson of horror ran though a good part of the Unix world: the scientific community felt that Sun was turning its back on them, and the other manufacturers feared that the "special relationship" would mean that Sun would get the jump on them. Armando Stettner told me:

> When Sun and AT&T announced the alliance, we at Digital were concerned that AT&T was no longer the benign, benevolent progenitor of Unix....
>
> Sun was everyone's most aggressive competitor. We saw Sun's systems were direct replacements for the VAX. Just think: the alliance combined our most aggressive and innovative competitor with the sole source of the system software—the balance shifted.

The direct consequence was that a meeting was held at DEC's Western Offices in Palo Alto, CA, on 7 January 1988. There were participants from Apollo, DEC, Gould Electronics, Hewlett-Packard, Honeywell-Bull, InfoCorp, MIPS, NCR, Silicon Graphics, UniSoft, and

Unisys. Because DEC's building is at 100 Hamilton Avenue, the attendees were referred to as the Hamilton Group. The immediate result was a telegram sent to James E. Olson, CEO of AT&T on January 15. It read:

> AS LICENSEES OF AT&T SOFTWARE AND SUPPORTERS OF AN OPEN UNIX STANDARD, WE ARE CONCERNED ABOUT RECENT ANNOUNCEMENTS BETWEEN AT&T AND SUN MICROSYSTEMS. THESE ANNOUNCEMENTS HAVE CREATED CONCERN WITHIN OUR COMPANYS [sic] AND OUR CUSTOMERS REGARDING THE FUTURE OF UNIX AS AN OPEN STANDARD. WE FEEL IT IS IMPORTANT TO GET A BETTER UNDERSTANDING OF THESE ISSUES BEFORE THE UPCOMING "UNIFORUM" CONFERENCE AND WE REQUEST A MEETING BETWEEN OUR CORPORATE MANAGER RESPONSIBLE FOR UNIX STRATEGY AND MR. CASSONI DURING THE WEEK OF JANUARY 25 1988....

Vittorio Cassoni was the Senior Vice President of AT&T's Data Systems Division. As a result of a good deal of activity, the telegram was signed by representatives of Apollo, Data General, DEC, Gould, HP, Intergraph, Integrated Solutions, MIPS, Motorola, NCR, Prime, SGI, Tandem, UniSoft, and Wyse.

Larry Lytle of HP called a preliminary meeting at the JFK Marriott for the evening of 27 January. The meeting with Cassoni took place on 28 January. There was a followup meeting of the Hamilton group in Dallas on 9 February. The meeting with Cassoni had no positive result where the Group was concerned. By March, the Group had decided to invite IBM to join.

Apollo, DEC (with Stettner urging Ken Olsen on), HP, IBM, Bull (France), and Nixdorf and Siemens (in Germany) held semi-secret meetings and, in May 1988, announced the formation of the Open Software Foundation, to be dedicated to the production of an operating system, a user interface, a distributed environment, and free cotton candy. Eventually, this Unix off-shoot would be AT&T license-free. The *Wall Street Journal* of 18 May 1988, noted that no one present at the launch of OSF could recall Ken Olsen sharing "a stage with an IBM chief executive."

Thompson was in Australia at the time. When Ritchie told him what had transpired, he said: "Just think, IBM and DEC in one room and we did it!"

It did not take long for AT&T, Sun, and their coterie to form a counter-consortium: UNIX International, dedicated to the marketing of SVR4.

By the end of 1989, OSF had come out with a user interface, Motif, which was well-received. OSF had also increased its number of sponsors by adding Philips and Hitachi. However, HP swallowed up Apollo and Siemens bought up Nixdorf. Both OSF and UI ran membership drives and gave out pens and badges and stickers. Each ended up with about 200 members.

In 1991 and 1992, as the economy worsened, with Bull, DEC, IBM and the computer side of Siemens all losing money, and with AT&T selling its portion of Sun, the fierce fighting and mudslinging appeared to be over. As early as November 1989 there had been talk of an OSF/UI merger, for the good of Unix. By the end of 1992, it hardly seemed to matter: there was no more talk of an AT&T license-free OS from OSF, and Sun had adopted Motif. By the end of 1993, UNIX International had gone out of business and OSF had abandoned several of its previously announced products— shrink-wrapped software and the distributed management environment. Bull and Siemens (for financial reasons) and Philips (because it sold its computer business) ceased their sponsorship of OSF. Armando Stettner remarked to me: "It's not clear whether there's any purpose to OSF any more."

CHAPTER 28

Berkeley Unix: After the VAX

In Chapter 18, I quoted Kirk mentioning that they had put "portals" into 4.4. As an instance of yet another British contribution to the development of Unix, here's what Jan-Simon Pendry told me:

> I put a thing called "portals" into 4.4BSD. They aren't quite the same thing as the <<portals>> defined in the 4.3 bsd architecture reference, but you can do a lot of the same things with them. Perhaps I should have chosen a different name.
>
> What happens is that a daemon creates a Unix domain socket, and attaches it to the system namespace (conventionally '/p'). It waits for messages from the kernel on that socket. ...
>
> The real idea is to get all of the junk network-specific code out of the applications, and into a daemon, or two. It makes implementing the V9 **ipcopen()** function very simple too. Of course, none of the applications have been modified in 4.4. You can also create network connections from your **awk** script, without needing all the socket junk that has infested **perl**.

The end of the UC Berkeley story is Kirk's:

> We got interested in the Computer Console Inc's Tahoe. It was clear to us that VAXes were not getting faster or cheaper.

Also, DEC had the Ultrix product by this time and they viewed 4.3BSD as a competitor. [*Armando Stettner was no longer with the DEC Unix group.*] We had stopped having the feedback: advance information on the new hardware, detailed specs on what everything did, device drivers and so on. It was clear that DEC was going to make life more difficult for us, and it wasn't clear that the VAX was an architecture that was going to make it in the long haul.

Sam did a machine search, he was working for LucasFilm by this time. He did an analysis of all the machines on the market. He decided that the CCI Tahoe series of machines had the best architecture and good current price/performance. Also they had really good things on the drawing board for future versions.

They had a 6 MIP machine that was the size of a desk side machine today. They had a 35 MIP machine on the drawing board that would have been out at a time when most of the other vendors would have been shipping machines with MIPs in the low to mid teens. We arranged to get some of their machines. They had a BSD port that they contributed to us. They felt that they had to go over to a System V based system for marketing reasons as many of the other vendors were doing.

The 4.3 Tahoe release was supposed to be vanilla 4.3BSD but ported to the Tahoe. In fact, some other things had been done in the meantime (like disk labels) and were included in the system, so some things were different. The idea was to make the first cut at dividing the system so that it was supporting two architectures.

Unfortunately, CCI cancelled their follow-on project and Tahoe turned into a dead architecture. They had at least had gotten us as far as doing the n=2 case.

The Reno release was the last system that we did that was a "boot it onto bare hardware" release. It became clear that we needed a port to something that was not really a mainframe but was a workstation. We got a line onto the HP workstations, the 68K based ones. In part, we did this because

> some people at Berkeley had begun that port and this was finished by the folks at the University of Utah. HP was willing to give us hardware and so it all worked well.

In June 1993, Berkeley released 4.4, terminating the CSRG at the same time. Both Keith Bostic and Kirk McKusick left the University of California; Mike Karels had departed the previous year. Research at Bell Labs had stopped working on Unix in the late 1980s. Now the most important non-AT&T site was "out of Unix development," too. Unix continues to develop, but the two most historically important sites no longer participate in that development.

CHAPTER

29

The Law—Part III

From about 1986 on, Keith Bostic would get up at the semi-annual USENIX conferences and announce the progress of his—the CSRG's—project: about 35% of the programs are AT&T code free; about 55% ...; about 77%.... Whatever the progress—and there always was some progress —there were cheers and loud applause. AT&T's lawyers had started off on the wrong foot in the mid-1970s. But that wasn't the whole reason for the cheers. The main motivation for the creation for a license-free Unix lay in AT&T's fees. When UCSF obtained V7, the license cost about $7,000. The "commercial" license cost far more. Clem Cole recalled:

> Since what Mellon Institue was using Unix for in those days, was for commerial gain, Dan [Klein] and I eventually went on "strike" until CMU purchased a real "industrial" Unix license. CMU became the first University to purchase the $20K license. This was a policy that a few other places, like Case Western Reserve followed a few years later.

That was 1978. A decade later, the programmers and professional users still felt alienated by the licensors of their favorite operating system: and the cost had climbed to over $100,000 for a source license. In 1993, it was around $200,000. This was far beyond the reach of individuals and small companies.

John Gilmore and several others nudged the CSRG to produce their license-free version. It was clear that AT&T hadn't objected to other derivatives: Linux, MINIX, etc. In the autumn of 1988, at the BSD Workshop, Keith, Mike Karels, and Kirk McKusick announced the completion of the BSD Networking Release 1. It appeared that November.

NET 1 was a subset of the current Berkeley system at the time, which was quite similar to 4.3-Tahoe. It included source code and documentation for networking portions of the kernel, C library and utility programs. It was available without evidence of any prior license (AT&T or Berkeley), and was redistributed via anonymous FTP. The source files contained a Berkeley copyright notice and a legend that allowed redistribution with attribution.

Two-and-a-half years later, in June, 1991, at the USENIX conference in Nashville, BSD Networking Release 2 was available.

NET 2 contained more than just networking code, but like NET 1 it was available with no prior license. The new features included a new virtual memory system derived from Carnegie-Mellon University's Mach system, which had been ported at the University of Utah, and a port to the Intel 386/486 system.

NET 2, a US-Russia collaboration, with contributions by Bill Jolitz, Donn Seeley, Trent Hein, Vadim Antonov, Mike Karels, Igor Belchinskiy, Pace Willisson, Jeff Polk, and Tony Sanders, was turned into a commercial product. The result, known as BSDI (Berkeley Software Design, Inc.) was complete by the end of 1991 and released to the public on April 10, 1993, largely as the result of delays caused by UNIX Systems Laboratories, which filed suit to prevent BSDI from shipping its product. USL requested an injunction barring distribution of "BSD/386, pending trial, arguing that BSD/386 infringed USL's copyright in its 32V software and misappropriated USL trade secrets." The Court denied USL's request for a preliminary injunction on 3 March 1993, ruling that USL had "failed to show a likelihood of success on either its copyright claim or its trade secret claim."

On 30 March 1993, Judge Dickinson Debevoise of the United States District Court of New Jersey reaffirmed his denial of USL's motion for a preliminary injunction against BSDI. The Court found that

the 32V source code had been distributed without a copyright notice. The Court rejected USL's argument that the publication of 32V was sufficiently limited to avoid a forfeiture, and thus found that USL had not demonstrated a likelihood that it could defend its copyright. The Court further ruled that USL had failed to establish that BSD/386 contained any USL trade secrets.

USL then filed a motion for reconsideration, asking the District Court to hold a new hearing on the issue of whether USL had published 32V without a copyright notice. USL argued that the Court's prior ruling was based on an incorrect finding as to the number of copies of 32V distributed. (USL's motion for reconsideration did not challenge the Court's ruling that USL had failed to establish trade secret misappropriation.)

The Court denied USL's motion for reconsideration. Although the Court amended its prior factual finding as to the number of copies distributed, it found the number was not critical to its ruling on the issue of publication without notice.

It was just under 20 years since Ken had delivered that paper at SOSP. It was 15 years since *UNIX NEWS* had become *;login:* and the UNIX Users Group had turned into USENIX. But through all of this AT&T, Western Electric, and now USL had learned nothing about the user community.

What BSDI and others were trying to do was ensure the continued development, growth, and use of the Unix operating system. The suit by USL was an attempt to protect something that had been discovered to be of value too late. It may be that Ritchie and Thompson had handled the system carelessly in the mid-seventies; it may be that BTL employees intentionally gave Unix to the public without any significant restriction.

BSDI had distributed pre-production releases of BSD/386 (Beta version). It now began international distribution of BSD/386. Full source was priced around $1000. (In the January/February 1994 *;login:*, Lou Katz wrote: "It works! It works!")

Because of the way the licenses had been granted to the University of California, the Regents of the University had been included in USL's suit. In June 1993, the Regents struck back, filing suit against

USL for not complying with agreements made with the CSRG. In the meantime, Novell had acquired USL.

At this time, there is no way of foreseeing the outcome of this marketing by litigation. What is clear, unfortunately, is that companies like Microsoft have taken advantage of both the Unix wars and the various marketing impediments that Unix vendors have thrown up. By the time this is printed, Novell, defending diversity against the onslaught of Windows NT, may have decided to quash the lawsuit they inherited.

[Last minute insert: on Friday, February 4, 1994, Novell and the University of California agreed to drop all relevant suits and countersuits. BSDI immediately announced the availability of "4.4BSD-Lite." "We are delighted to settle with USL so that we can devote our full efforts to creating products and serving our customers," Rob Kolstad, president of BSDI, said to me.]

Finale

Finale: What Made It work?

From CTSS to Multics to Unix on the PDP-11/20 to where we are now has been a long tale—a 30-year story. But there are lessons to be learned that I believe many (if not all) of the commercial entities of the computer industry have not seen.

Perhaps foremost among these was the attitude that prevailed at Bell Labs. Dennis Ritchie's words about fun and utility in Chapter 1 exemplify this attitude. Brian Redman remarked to me that the work he did at Whippany was "a lot of fun." Plauger remarked, "It's amazing, but if you're having fun and you're doing anything at all sensible, you can relate it to the business of AT&T." Brian Kernighan says that when he was hired he asked Doug McIlroy what he should be doing; Do what you want, he was told. So we—the world of programmers and users—got eqn, ratfor, and pic as well as *Software Tools*.

Recall, Kernighan told Peter Collinson that awk was

> by far the biggest software project that I have ever been involved with. There were three of us in that, and that's completely unworkable. Somehow, it's much easier working with two rather than three. ... It's harder to split things. There's more divergence of opinion, sometimes that's good because it means that more things are there but sometimes it means that it's not as cohesive as it might be.

Three too many? Microsoft, if one can believe the trade press, had 600 programmers working on Windows NT. But there were three working at Holmdel, creating 32V (London, Reiser, and Swanson); three who wrote HoneyDanBer; Hall, Sventek and Scherrer wrote VOS; Lorinda Cherry and Bob Morris wrote bc and dc; and the CSRG at Berkeley rarely had more than a handful of full-time workers: Haley and Joy; Haley, Joy and Kridle; Joy, Kridle and Leffler; McKusick part-time and in the summers until he earned his Ph.D.; McKusick, Jolitz and Karels; McKusick, Karels and Bostic. Larry Wall wrote patch, perl, rn by himself. And look at those who just "dropped by": Ken Thompson, George Coulouris, various Australians, Jan-Simon Pendry. And look at the other contributors: Mike Muuss, Doug Kingston, Jim Curry, Rick Adams, James Gosling, Rob Pike, Armando Stettner, Bill Shannon, etc.

Steve Johnson told me:

> If I had been at a university, I would have been considered either a software person or a theory person. There would have been a pot of money that paid my salary. And if I were a software person and talked to theoreticians, the software people who were paying my salary would have felt cheated. And conversely, if I had been a theory person, and tried to do something useful.... I've really always been a generalist. I've operated typically in the cracks between different disciplines, where I found a lot of very fertile ground. So I will write a program and this will suggest some problems that I can prove theorems out of and turn about and put the theorems back in the programs. That happened with yacc and pcc and some things I've done since then. It just doesn't fit well into a compartmentalized structure.

Here's another point: avoid compartmentalizing researchers and developers.

Eric Allman remarked to me that:

> I think one general rule of software design is that you should be writing a program that you want to use. Ken and Dennis wanted to use Unix. They did what they needed in order to make it work. We wanted to use **sendmail**, it wasn't something where we said "Oh, let's write a mailer and send it

> out..." Bill Joy didn't come to me and say "Oh, Eric, what we need is this," in fact, I had to lean on him to get it released. We had a problem that needed to be solved. Ken had a problem of sorts: he didn't have an adequate system to do space games—so he wrote one. Compare this to X.400, where I'm convinced that people who never actually use mail simply write papers about it. Other proprietary OSes, too, because you assign people to do the jobs.

While I was chatting with Armando Stettner, he remarked:

> You know, the wonderful thing about Unix—and about Plan 9, today, as I understand it—is that both of those are collections of a lot of great ideas that were around at the time and some original ideas, but put together in a very interesting, very powerful way.... Bell Labs brought together lots of ideas in a very good implementation. There were also new ideas, original ideas. Berkeley Unix brought together lots of good ideas into a programmers interface—the implementation varied in quality. But it brought together actual work—people from all over the world put code in and got this system and participated in it. And there were arbiters of good taste: Bill Joy and Sam Leffler really were artistes. Kirk McKusick, Mike Karels. For lack of a better title, we call them architects. But they were really arbiters of good taste. They put the stuff in, they made sure things fit cleanly.

Synthesis *is* creativity. But one of the most interesting points about the evolution of Unix is that much of it occurred despite the activities of its "owners," of AT&T and Western Electric. Greg Rose pointed this out to me:

> It seems that in so many stages of Unix's evolution, an action that AT&T took in order to stifle something actually caused the opposite to happen.

Brian Kernighan said:

> You know it's the PC environment that requires continuous change to keep the market alive as it were. You have to have

> something new, or your competitor will have something new and he will drive you out of business. The other thing is that the PC environment is essentially homogeneous, they are all basically running the same chip—the same processor; because the environment is in that sense homogeneous and it is much larger than the Unix market so you can amortize your costs over a very much larger base. It's such big market that if somebody has a good idea they are going to go into the PC market and not into the Unix market. I think the market will change and that the advantages of Unix over the long haul will outweigh the advantages of DOS: The advantages of portability and universality.

Unix has influenced every aspect of computing in the '90s. Even what Kernighan called "the PC environments" have taken on Unix traits and characteristics. What's more, Microsoft's Windows NT is clearly a Unix-like environment and the product has been announced as "POSIX-compliant." (Though it is unclear to me just what Bill Gates meant by this.)

In 1984, after divestiture, Dennis Ritchie was asked whether the ambience of the Labs which had given rise to Unix could be duplicated. His response appeared in the August 1984 *CACM*, almost exactly a decade after the "UNIX Time-Sharing System" had been published. Among other things, Ritchie wrote:

> Some people have the impression that the original Unix work was a bootleg project, a "skunk works." This is not so. Research workers are supposed to discover or invent new things.... We always had management encouragement.... Our intent was to create a pleasant computing environment for ourselves, and our hope was that others liked it....
>
> The greatest danger to good computer science research today may be excessive relevance.... Another danger is that commercial pressure of one sort or another will divert the attention of the best thinkers from real innovation to exploitation of the current fad, from prospecting to mining a known lode....

> If we can keep alive enough openness to new ideas, enough freedom of communication, enough patience to allow the novel to prosper, it will remain possible for a future Ken Thompson to find a little-used CRAY/1 computer and fashion a system as creative, and as influential, as Unix.

Thompson and Ritchie wanted to build something they would enjoy using. They succeeded, and others wanted to share this enjoyment. Sunil Das wrote: "Without the vision of Ken Thompson, UNIX would not have come into existence; without the insight of Dennis Ritchie, it would not have evolved into a polished presence; without the imagination of Michael Lesk, and the popularising touch of Brian Kernighan, it would not have acquired the extroverted personality that commands such widespread loyalty." This loyalty inspired Coulouris in London, Lions in Sydney, and Joy in Berkeley, as well as the myriad others mentioned in these pages.

David Tilbrook concluded, "The one thing has to be stated about Unix is that it wasn't a great advance in computing; if anything it was a great simplification, it put into the realm of the user things that were just inconceivable prior to that."

Further Reading

Where the prehistory of computing is concerned, the articles on calculating and the various individuals (e.g. Babbage, Hollerith, Leibniz, Napier, Newton, Pascal), in the *Encyclopedia Britannica* are excellent. For the period up to 1960, Robert Sobel's *IBM Colossus in Transition* (Times Books, 1981) and B.V. Bowden, ed., *Faster than Thought* (Pitman, 1953) are excellent. MIT Press has also published several fascinating volumes of memoirs: Pugh's *Memories That Shaped an Industry* (1984), Maurice Wilkes' *Memoirs of a Computer Pioneer* (1985), Bashe et al.'s *IBM's Early Computers* (1986), and Lundstrom's *A Few Good Men from Univac* (1987), are the most notable. In French, Rene Moreau's *Ainsi naquit l'informatique* (Bordas, 1981) is exceptionally good for the period up to 1963. Where the immediate predecessors of Unix are concerned, F. J. Corbato's brief book on CTSS (MIT Press, 1963) and Organick's *The Multics System* (MIT Press, 1972) are recommended. *A History of Computing in the Twentieth Century*, ed. by Metropolis, Howlett, and Rota (Academic Press, 1985) [from a 1976 conference], unfortunately promises more than it delivers.

For a brief overview of hardware architecture, see Baron and Higbie, *Computer Architecture* (Addison-Wesley, 1992); the companion volume of *Case Studies* is exceptionally useful.

As has been related, the first article about Unix was the revision of the paper by Ritchie and Thompson delivered in 1973 and published in *CACM* in July 1974. In 1977, John Lions published *UNIX Operating System Source Code Level Six* and its companion, *A Commentary on the UNIX Operating System*. His "Experiences with the UNIX Time-Sharing System" appeared in *Software—Practice and Experience* in 1979. In the meantime, *The Bell System Technical Journal* had published its long-awaited Unix issue: 21 articles of genuine value and importance (vol. 57, no. 6, part 2; the entire issue was reprinted by Prentice Hall in 1987).

Commercial Unix publishing began with Richard L. Gautier's truly inadequate *Using the Unix System* (Reston Publishing Company, 1981), which was based on Version 7. The next year, Rebecca Thomas and Jean Yates' *User Guide to the UNIX System* (Osborne McGraw-Hill, 1982) appeared. It was a vast improvement over Gautier, though it contained a large number of errors. Most of these were corrected in the second edition (1985). In 1983, McGilton and Morgan's *Introducing the UNIX System* (McGraw-Hill) was published, as was S.R. Bourne's superb *The UNIX System* (Addison-Wesley). But the Unix eruption in publishing came in 1984—the lava flow has not yet abated.

Kernighan and Pike's *The UNIX Programming Environment* (Prentice-Hall), Kochan and Wood's *Exploring the UNIX System* (Hayden), and Sobell's *Practical Guide to the UNIX System* (Benjamin/Cummings) were just the beginning. The first volumes were also appearing in Britain: Chris Miller and Roger Boyle's *Unix for Users* (Blackwell), A.N. Walker's *The UNIX Environment* (Wiley) and P.J. Brown's truly excellent *Starting with UNIX* (Addison-Wesley) were the starters. And in October 1984 the newly-named *AT&T Bell Laboratories Technical Journal* (nee *BSTJ*), published a "second" Unix issue (vol. 63, no. 8). Thinking back to the various announcements made by Ken Thompson as to the number of Unix systems in use in the "early" years, the remark of Robert L. Martin (then Executive Director of the Computer Systems Software division) in the "Preface" that "there are on the order of 100,000 *UNIX* systems now in operation" is the key to just why Unix books were appearing. With several hundred thousand potential buyers, publishers were rushing to fill the perceived vacuum.

The past decade has seen such a surge in publishing that any attempt at enumeration would be absurd. However, several books deserve to be singled out: Maurice J. Bach's *The Design of the UNIX Operating System* (Prentice-Hall, 1986) on SVR2; Sam Leffler, Kirk McKusick, Michael Karels, and John Quarterman's *The Design and Implementation of the 4.3BSD UNIX Operating System* (Addison-Wesley, 1989), and Andy Tanenbaum, *Operating Systems: Design and Implementation* (Prentice-Hall, 1987), on the technical level; Harley Hahn's *A Student's Guide to UNIX* (McGraw-Hill, 1993), for the learner.

The early history of programming languages is limned in Jean Sammet's *Programming Languages: History and Fundamentals* (Prentice-Hall, 1969) and Richard L. Wexelblat, ed., *History of Programming Languages* (Academic Press, 1981) [the proceedings of the ACM's History of Programming Languages meeting, June 1-3, 1978]. The proceedings of the 1993 HOPL meeting in Cambridge, MA, were not yet available in January 1994.

Kernighan and Ritchie's *The C Programming Language* (Prentice-Hall, 1978) remains a lucid and informative work; the revised second edition (1988) complies with the ANSI C standard. Aho, Weinberger and Kernighan, *The awk Programming Language* (Addison-Wesley, 1988) presents the language superlatively.

Software Tools by Kernighan and Plauger (Addison-Wesley, 1976) is unsurpassed where learning how to write good programs is concerned. There is no history of networking, yet. John Quarterman's *The Matrix* (Digital Press, 1990) remains the best source, though the information is scattered through the volume.

The many articles in *;login:*, the UKUUG and AUUG newsletters, and the USENIX, AUUG, UKUUG, and EUUG (EurOpen) proceedings are too inaccessible for me to list. There have been a number of interesting articles on Unix and Unix-like systems in the quarterly *Computing Systems*, including Amoeba, Chorus, Dune, Sprite, and Clouds. Finally, *UNIX Review* for January 1985 (vol. 3, no. 1) contains a number of entertaining and valuable articles on the history of Unix.

Who's Who and What's What

ADM-3a an early screen terminal

Adams, Rick wrote SLIP; principal of UUNET and BSDI

Aho, Al the A in awk; taught Steve Johnson about compilers; at BTL since 1966.

AIX IBM's version of Unix

Allman, Eric inventor of sendmail, tset, the -me macros, and a major contributor to Ingres

Alpak the name of two different computer algebra systems

Antonov, Vadim major contributor to Unix in the USSR; now with UUNET in the US

Arnold, Ken wrote curses and lots of other stuff

ARPANET the first computer network, sponsored by the Advanced Research Projects Agency of the US Department of Defense

AUUG The Australian Unix User's Group

AUX Apollo's version of System III

Babaoglu, Ozalp with Bill Joy, wrote virtual memory for BSD (1978-79); designed paging

Baker, W.O. VP for Research at BTL when Multics was axed; rejected request for DEC-10

Baecker, Ron responsible for the Toronto CSRG and for Pike, Duff, Tilbrook, and Tilson

Barak, Amnon originator of MOSIX

BBN Bolt, Baranek & Newman, important in the development of editors, the ARPANET, etc.

Bechtolsheim, Andreas designer of original SUN board

Beertema, Piet perpetrator of kremvax

Bellcore Bell Communications Research

Bellovin, Steve author of first USENET implementation

Biff Heidi's dog

Boggs, David co-inventor of Ethernet (with Metcalfe)

Bostic, Keith member of CSRG; worked on 2.10, 4.4, etc.

BRL US Army's Ballistics Research Lab

BSD Berkeley Software Distribution

BSDI Berkeley Software Design, Inc.

BTL Bell Telephone Laboratories

Canaday, Rudd original collaborator on the Unix file system; creator of PWB group

Cassoni, Vittorio Senior VP of Data Systems at AT&T, tragically killed in an air crash in 1988

CMU Carnegie Mellon University

Cole, Clem early Unix enthusiast; influential where uucp protocols and early OSF were concerned

Collinson, Peter originator of UKnet

Corbato, Frederico leader of CTSS project; MIT's principal of the Multics effort

Coulouris, George originator of em, ancestor of ex and vi; first user of Unix outside North America

CSRG the Computer Systems Research Group at the University of California, Berkeley; also (unrelated) group at University of Toronto

CTSS Compatible Timesharing System

CWI the Centrum voor Wijskunde en Informatica, Amsterdam (formerly the MC)

Das, Sunil long-time chair of the UKUUG

DARPA [US] Defense Advanced Research Projects Agency

Debevoise, Dickinson judge of the US District Court in New Jersey

DeCastro, Edson DEC engineer who implemented PDP-5 and PDP-8; left DEC to form Data General in 1968

DEC Digital Equipment Corporation

DECUS DEC Users Society

Dix, C.W. GE policy representative for Multics

Dolotta, Ted first director of USG; wrote -mm macros

EDSAC Electronic Delay Automatic Storage Calculator

EDVAC Electronic Discrete Variable Computer

Elz, Robert early Australian user who wrote BSD disk quotas and autoconfiguration files

EMACS Editing Macros

ENIAC Electronic Numerical Integrator and Calculator

Ethernet a local area network invented by Metcalfe and Boggs

EurOpen European Unix organization; formerly EUUG

EUUG European Unix Systems User Group (now EurOpen)

Fabry, Robert faculty advisor of CSRG who obtained DARPA funding for BSD

Fano, Robert M. founding director of MIT's Project MAC

Fateman, Richard Berkeley professor who developed Macsyma, invented Ingres, and purchased the VAX 11/780

Feldman, Stu inventor of make, architectural historian

Ferentz, Mel founder of *UNIX NEWS* and one of the founders of USENIX

Ferrin, Tom computer graphics developer, Unix hacker, deviser of hardware fix to software problem

Foderero, John wrote biff and Franz Lisp

Forrester, Jay director of Whirlwind, Whirlwind II, and SAGE

Forsyth, Dan one of the principals of the Georgia Tech software tools development

FSF Free Software Foundation

FTP File Transfer Protocol

GECOS General Electric Comprehensive Operating System

GCOS Honeywell's renaming of GECOS

Gilmore, John major force at Sun early on; principal of Cygnus

GNU product of the FSF, a recursive acronym for Gnu's Not Unix

Groundwater, Neil first user of Unix outside of New Jersey; first user of INGRES outside of Berkeley; early supporter of STUG

Gurwitz, Bob BBN member of DARPA's steering committee; author of TCP/IP code

Hagen, Teus established first trans-Atlantic uucp connection

Haight, Dick major contributor to PWB; wrote find, cpio, etc.

Haley, Chuck collaborator with Joy on ex and Pascal shell; wrote tar

Hall, Dennis co-implementor of VOS

Hawthorn, Paula database activist; manager of O'Dell and Allman at different times

HCR Human Computing Resources, the first Canadian Unix company

Henry, Robert creator of error

Holmgren, Steve coauthor of first ARPANET code for Unix

HoneyDanBer most common upgrade of UUCP

Honeyman, Peter the Honey of HoneyDanBer

Hume, Andrew one of the editors of 10th Edition

Idris Plauger's Unix-like system

IEC International Electrotechnical Commission

IIASA International Institute for Applied Systems and Analysis, Laxenburg, Austria

IIJ commercial Japanese IP network

INGRES Interactive Graphics and Retrieval System; first Unix-based relational database

IP Internet Protocol

IPC interprocess communication

Ishida, Haruhisa first user of Unix in Japan

JAWs Just Another Workstation

Johnson, Steve wrote lint, yacc, spell, pcc; worked with Ritchie on Interdata port; fifth president of USENIX

Joy, Bill Unix enthusiast; created much of BSD, 2BSD, 3BSD, 4BSD; co-founder of Sun Microsystems; designed NFS

JUNET Japan Universities' Network

jus Japan Unix Society

Karels, Mike motive force at the CSRG

Kashtan, David author of Eunice

Katz, Lou first president of USENIX

Kernighan, Brian the k in awk; wrote ratfor, ditroff, eqn, pic; co-author with Ritchie (*The C Programming Language*) and Plauger (*Software Tools*); and much, much more

Kilburn, Tx coinventor of electrostatic memory

Kishida, Koichi one of the founders of jus

Kolstad, Rob first humor editor of *;login:*; now editor

Korn, David inventor of Korn shell

Kridle, Bob systems programmer at Berkeley; later one of the founders of mt Xinu

Law, Lew director of technical services at Harvard Science Center; member of first USENIX Board; publisher of Unix manuals

LBL Lawrence Berkeley Laboratories

Leffler, Sam major force behind 4.2BSD and TCP; wrote tip with Shannon

Lesk, Mike wrote lex, tbl, refer, -ms macros, first uucp

Lions, John first trans-Pacific Unix user; author of first Unix book

London, Tom one of the creators of 32V

LSX version of Unix for the LSI-11 processor

Lycklama, Heinz wrote MERT with Bayer; wrote LSX; chaired first Unix standards committee

Lyon, Tom while at Princeton ported much of Unix to IBM VM/370

Lytle, Larry one of the forces behind OSF

Maltby, Chris Australian Unix pioneer

Manno, Paul Georgia Tech tools guru

Marsh, Bob founder of Onyx; founder of /usr/group

Mashey, John wrote much of PWB; wrote Mashey shell

MC Mathematische Centrum, Amsterdam (now CWI)

McIlroy, Doug suggested pipes; wrote tmg; wrote diff; herded cats

McKusick, Marshall Kirk wrote Berkeley Fast File System; major force behind 4.3, Net-1, Net-2, and 4.4; fourth president of USENIX

McMahon, Lee wrote sed

Meaney, Thomas F. Judge; author of 1956 Consent Decree

MERT Roberts' Multi-environment Real Time

Metcalfe, Robert co-inventor of the Ethernet

Mezei, Les one of the founders of HCR

MINIX Unix-like system invented by Andy Tanenbaum

MOSIX distributed system invented by Amnon Barak

MSDOS Microsoft disk operating system

Mullender, Sape one of the principals of the Amoeba project

Multics Multiplexed Information and Computing Service

MUNIX the first Multiprocessor Unix

Murai, Jun founder of JUnet, co-founder of jus

Muuss, Mike responsible for JHU/BRL Unix, for early TCP code; wrote ping

NCP Network Control Protocol, first ARPANET protocols, by Steve Holmgren, Steve Bunch, and others; ported by Muuss

Nemeth, Alan BBN member on DARPA committee; second president of USENIX; responsible for BBN's C machine

NFS Network File System

NLUUG Netherlands Unix Systems User Group

Nowitz, Dave The Dan in HoneyDanBer

O'Dell, Michael D. inventor of bsmtp protocol; colleague of Rick Adams

Olsen, Kenneth founder of DEC

OSF Open Software Foundation

OSKER Roberts' Operating System Kernel

Ossanna, Joseph responsible for troff

PDP DEC's Programmed Data Processors

PDP-1 18-bit machine; 1960; $120,000

PDP-4 18-bit predecessor of the PDP-7; 1962

PDP-7 the first Unix machine; 18-bits; 1965; ~$60,000

PDP-11 DEC's first and only 16-bit computer; 1970; $10,800 in minimum configuration; there were many models—an 11/94 was produced in 1990

Pendry, Jan-Simon implementor of portals in 4.4BSD

Pike, Rob co-developer of the Blit terminal; involved with Plan 9

Plauger, P.J. wrote first commercial C compiler; founded Whitesmiths; created Idris

Pmax DEC workstation

POSIX set of computer standards committees

Presotto, Dave wrote vgrind with Bill Joy; involved with Plan 9

PTT Post, Telephone, Telegraph

PWB Programmer's Workbench

QMC Queen Mary College, University of London; now Queen Mary and Wakefield College

Quarterman, John S. author of *The Matrix* and editor of *Matrix News*

Rashid, Rick responsible for Mach

Redman, Brian the Ber of HoneyDanBer

Reiser, John coauthor of 32V

RFC Request for Comment

RISC Reduced Instruction Set Computer

Ritchie, Dennis M. one of the originators of Unix; principal author of C

RJE Remote Job Entry

RK a family of DEC drives

Roberts, Charlie creator of MERT; director of 32V project

Saito, Nobuo one of the founders of jus

SCCS Source Code Control System

Scherrer, Debbie one of the implementors of VOS; founder of STUG; president of mt Xinu; equestrian extraordinaire; third president of USENIX

Scherrer, Phil founder of Unicorn Systems; early STUG booster

Schriebman, Jeff founder of UniSoft

Schulman, Bob installer of Unix on Japan's first VAX

SCO Santa Cruz Operation

Seeley, Donn worked on f77 and pcc as well as Net-2

SGI Silicon Graphics, Inc.

Shienbrood, Eric wrote more

SIG Special Interest Group

SMTP Simple Mail Transfer Protocol

SOSP Symposium on Operating System Principles

Spafford, Gene involved with Georgia Tech tools effort

SSEC Selective Sequence Electronic Calculator

Stallman, Richard M. chief GNUsance; responsible for emacs, GNU and FSF

Standiford, Keith installer of Unix at Berkeley in January 1974

Stettner, Armando got DEC to acknowledge Unix; instigator of OSF

Stettner, Heidi owner of Biff

STUG Software Tools User Group

Sventek, Joe one of the implementors of VOS; co-founder of STUG

SVID System V Interface Definition

Tanenbaum, Andy creator of MINIX; originator of Amoeba

Tague, Berkely secretary to the Multics triumvirate; founder of USG

TCP Transmission Control Protocol

TECO early MIT editor

TENEX BBN OS for the DEC-10

Tevanian, Avadis co-originator of Mach

Thompson, Ken originator of Unix; implementor of many things; creator of Belle, sometime computer chess champion

Tilbrook, David originator of NEWSWHOLE, founder of HCR; program chair of first EUUG conference

Tilson, Michael president of HCR; now VP of SCO

Tobey, Alan co-founder of mt Xinu

Torvalds, Linus creator of Linux

Tower, Len Associate GNUsance; finder of vegetarian restaurants

Trickey, Howard part of Plan 9 team

Truscott, Tom co-originator of USENET

TWENEX BBN follow-up OS for the DEC-20

Ubell, Mike wrote history prototype

UEG DEC's Unix Engineering Group

UKUUG United Kingdom Unix Systems User Group

Ultrix DEC's version of 4.2BSD

UNICOM the 1983 joint STUG, USENIX and /usr/group conference

UNICS original name of Unix

UniForum current name of /usr/group

UniPlus UniSoft's port of Unix

USENET over 6000 examples of chaos theory

USENIX oldest and largest of the Unix users' groups

USG AT&T's Unix Systems Group

USL Unix Systems Laboratories

UUCP Unix-to-Unix Copy

UUNET online communications supplier conceived by Rick Adams

Van Vleck, Tom toiler in the Multics area

VAX DECs 32-bit machine family; Virtual Address Extension

VM virtual machine

VMS DEC's OS

VOS Virtual Operating System, created by Dennis Hall, Debbie Scherrer and Joe Sventek

VU Vrije Universiteit, Amsterdam

Wall, Larry author of patch, perl, and rn

Wedel, Wally STUG and USENIX activist

Weinberger, Peter the w in awk; the face on the water tower

Weiner, Peter obtained first commercial Unix license; founded Interactive Systems

Whitesmiths software house founded by P.J. Plauger

Wollongong University that ported Unix to the Interdata 7/32 as Ritchie and Johnson were porting to the 8/32

X3J11 C language standards committee

XENIX Microsoft-SCO Unix collaboration

XENIX1 based on V6

XENIX2 based on V7

XPG X/Open Portability Guide

Zimmerman, Hubert founder and president of Chorus Systèmes

Zlotnick, Fred POSIX activist

Index

UNIX/Operating Systems

On June 12, 1972, Ken Thompson and Dennis Ritchie wrote, "the number of UNIX installations has grown to 10, with more expected." Two years later the number was 50. It is estimated that there are over 3 million UNIX systems in operation today...

A Quarter Century of UNIX

Peter H. Salus

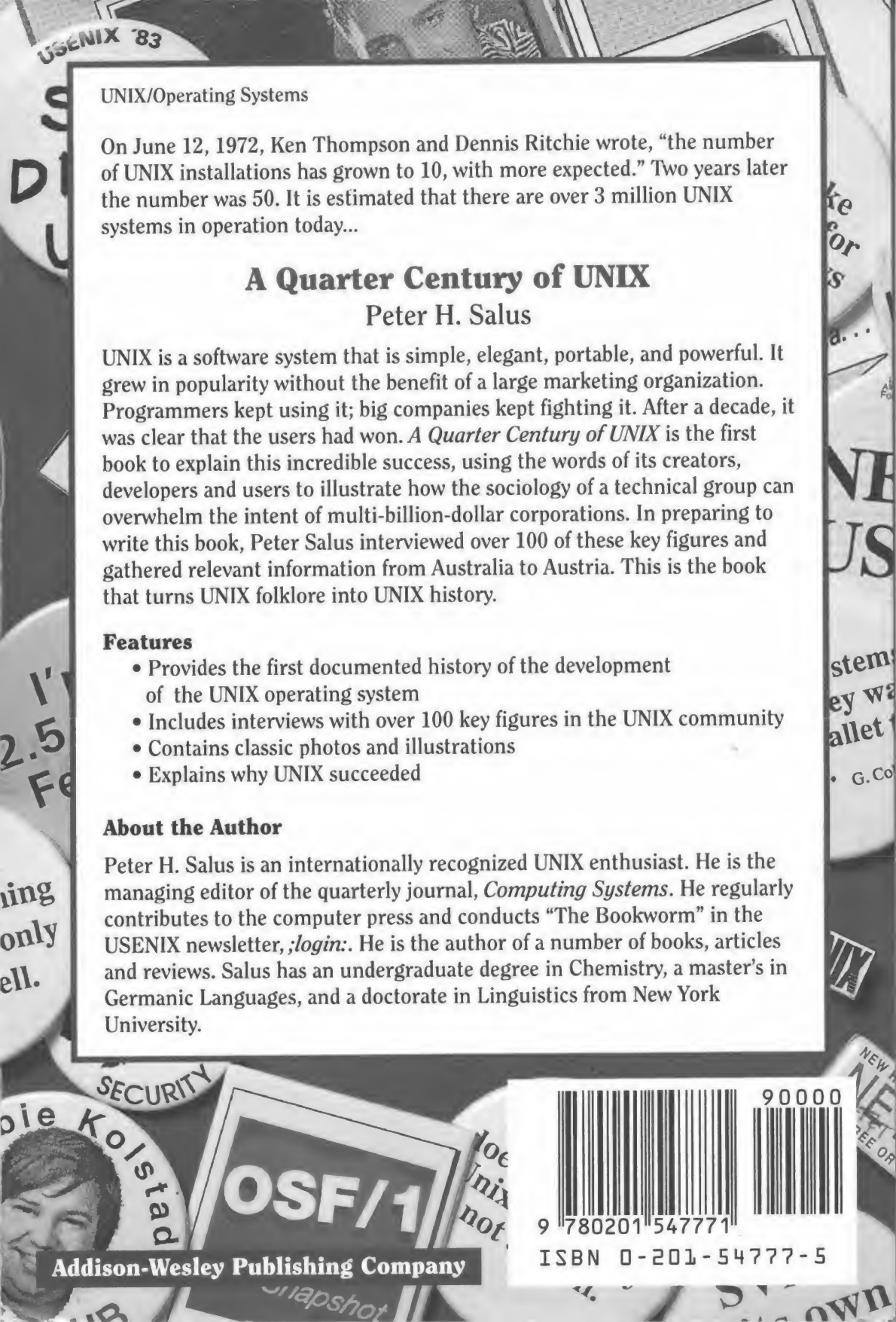
UNIX is a software system that is simple, elegant, portable, and powerful. It grew in popularity without the benefit of a large marketing organization. Programmers kept using it; big companies kept fighting it. After a decade, it was clear that the users had won. *A Quarter Century of UNIX* is the first book to explain this incredible success, using the words of its creators, developers and users to illustrate how the sociology of a technical group can overwhelm the intent of multi-billion-dollar corporations. In preparing to write this book, Peter Salus interviewed over 100 of these key figures and gathered relevant information from Australia to Austria. This is the book that turns UNIX folklore into UNIX history.

Features

- Provides the first documented history of the development of the UNIX operating system
- Includes interviews with over 100 key figures in the UNIX community
- Contains classic photos and illustrations
- Explains why UNIX succeeded

About the Author

Peter H. Salus is an internationally recognized UNIX enthusiast. He is the managing editor of the quarterly journal, *Computing Systems*. He regularly contributes to the computer press and conducts "The Bookworm" in the USENIX newsletter, *;login:*. He is the author of a number of books, articles and reviews. Salus has an undergraduate degree in Chemistry, a master's in Germanic Languages, and a doctorate in Linguistics from New York University.



90000
9 780201 547771
ISBN 0-201-54777-5