CHM BLOG FROM THE COLLECTION , SOFTWARE HISTORY CENTER

THE EARLIEST UNIX CODE: AN ANNIVERSARY SOURCE CODE RELEASE

By David C. Brock | October 17, 2019

What is it that runs the servers that hold our online world, be it the web or the cloud? What enables the mobile apps that are at the center of increasingly on-demand lives in the developed world and of mobile banking and messaging in the developing world? The answer is the operating system Unix and its many descendants: Linux, Android, BSD Unix, MacOS, iOS—the list goes on and on. Want to glimpse the Unix in your Mac? Open a Terminal window and enter “man roff” to view the Unix manual entry for an early text formatting program that lives within your operating system.

Dennis Ritchie—began the construction of a new operating system, using a then-aging DEC PDP-7 computer at the labs. As Ritchie would later explain:

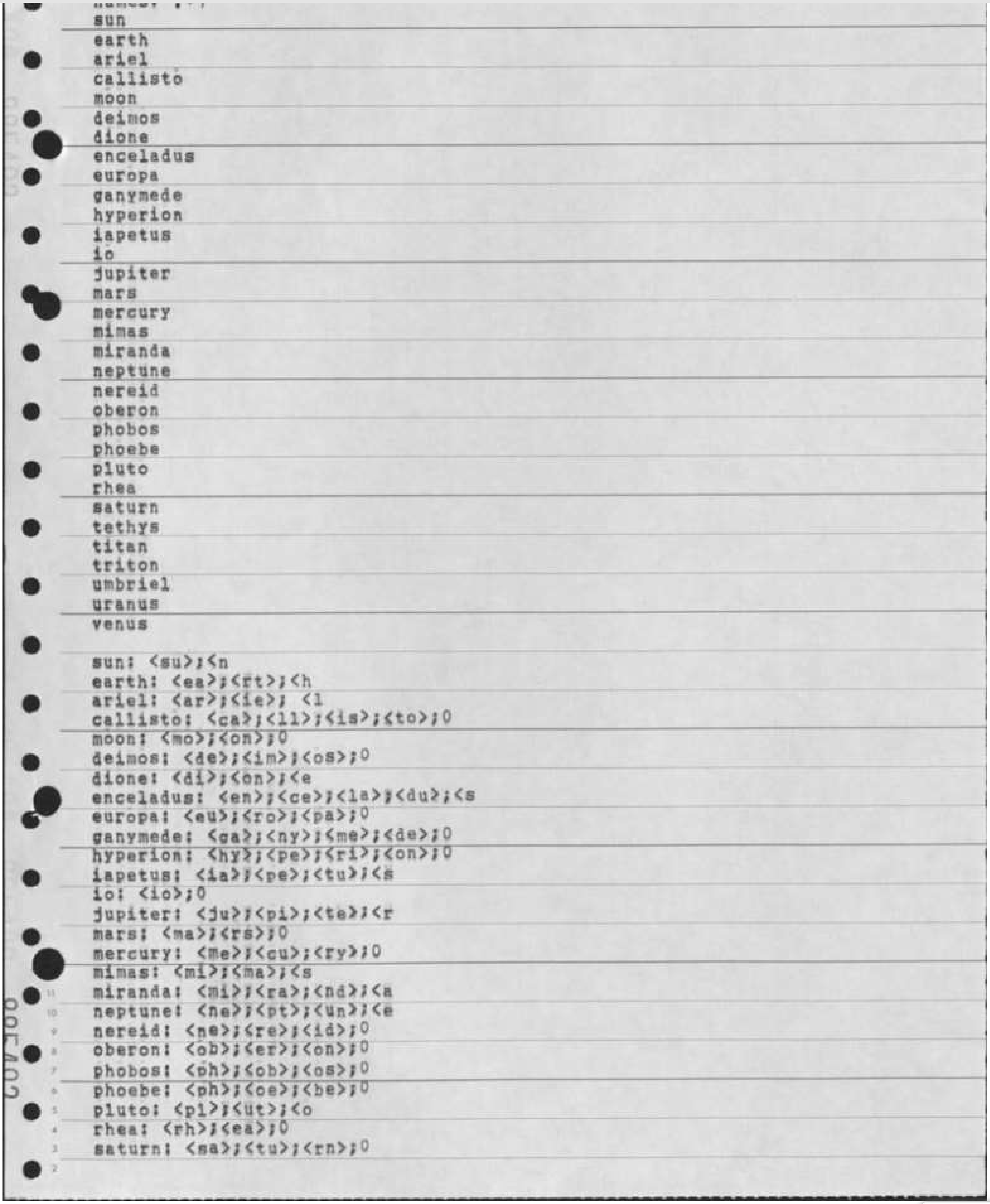
> “What we wanted to preserve was not just a good environment to do programming, but a system around which a fellowship could form. We knew from experience that the essence of communal computing, as supplied from remote-access, time-shared machines, is not just to type programs into a terminal instead of a keypunch, but to encourage close communication.”[1]

Ken Thompson (seated) and Dennis Ritchie (standing) with the DEC PDP-11 to which migrated the Unix effort in 1971. Collection of the Computer History Museum, 102685 tremendously influential. C and languages inspired by it (C++, C#, Java) predominate the list of the most popular programming languages to the present. Indeed, they account for 4 of the 10 most popular programming languages in 2019 according to the IEEE.[2] C itself is #3.

To mark Unix's 50th anniversary, the CHM Software History Center is delighted to make publicly accessible for the first time some of the earliest source code produced in the Unix story.

Recently, CHM was entrusted to preserve the papers of Dennis Ritchie by the Ritchie family. Within these papers, I identified a black binder with the hand-label "Unix Book II" containing nearly 190 pages of printed source code listings, written in PDP-7 assembly code.

With invaluable early review of these listings from Warren Toomey of The Unix Heritage Society and from John Mashey, an early Unix contributor and CHM trustee, we can date these listings to 1970, perhaps early 1971, before the Unix effort migrated to a new PDP-11. A PDF of the listings contained in this Unix Book II binder is available for download via this catalog record.



```
sun
earth
ariel
callisto
moon
deimos
dione
enceladus
europa
ganymede
hyperion
iapetus
io
jupiter
mars
mercury
mimas
miranda
neptune
nereid
oberon
phobos
phoebe
pluto
rhea
saturn
tethys
titan
triton
umbriel
uranus
venus

sun: <su>;<n
earth: <ea>;<rt>;<h
ariel: <ar>;<ie>; <l
callisto: <ca>;<ll>;<is>;<to>;0
moon: <mo>;<on>;0
deimos: <de>;<im>;<os>;0
dione: <di>;<on>;<e
enceladus: <en>;<ce>;<la>;<du>;<s
europa: <eu>;<ro>;<pa>;0
ganymede: <ga>;<ny>;<me>;<de>;0
hyperion: <hy>;<pe>;<ri>;<on>;0
iapetus: <ia>;<pe>;<tu>;<s
io: <io>;0
jupiter: <ju>;<pi>;<te>;<r
mars: <ma>;<rs>;0
mercury: <Me>;<cu>;<ry>;0
mimas: <mi>;<ma>;<s
miranda: <mi>;<ra>;<nd>;<a
neptune: <ne>;<pt>;<un>;<e
nereid: <ne>;<re>;<id>;0
oberon: <ob>;<er>;<on>;0
phobos: <ph>;<ob>;<os>;0
phoebe: <ph>;<oe>;<be>;0
pluto: <pl>;<ut>;<o
rhea: <rh>;<ea>;0
saturn: <sa>;<tu>;<rn>;0
```

A page from the source code listing for ***Space Travel*****, ca. 1970.** *Space Travel* was critica the start of the Unix story. Ken Thompson began implementing the science fictional ga developing a full, if fledgling, operating system for the computer that incorporated file system and other ideas that he and others in his computer science department had been considering. Ritchie and other colleagues were soon attracted to the system and its development. That early system, the start of Unix, and programs for it are represented in this source code release. Collection of the Computer History Museum, Dennis M. Ritchie Papers, 102788942.

These programs were written by the first participants in the Unix effort at the Bell Telephone Laboratories, starting in 1969. Ken Thompson and Dennis Ritchie were the two central, and initial, participants. Other early participants include Rudd Canaday, Doug McIlroy, Brian Kernighan, Bob Morris, and Joe Ossanna. It is likely that much of the work represented in this binder is due to the work of Thompson and Ritchie.

The binder is also likely to have been originally kept in the “Unix Room” at Bell Labs and was part of a collection there of the earliest Unix code. The collection appears to have been divided into two binders, presumably the Unix Book II now preserved at the Museum and another companion binder, perhaps labeled “Unix Book I.” The listings within this companion binder were photocopied by Norman Wilson in the later 1980s and, in 2016, scanned and made available through The Unix Heritage Society. The current location of this companion binder is unknown.

Provisional identifications and notes on the program listings in Unix Book II, keyed to the page numbers of the PDF, follow. Our sincere thanks to Warren Toomey and John Mashey for their vital assistance in these provisional identifications and notes. We are excited to see what additional identifications and insights will come from the examination of these source code listings by the public. To share your ideas and insights with us, please join the discussion at the end of this post or email the CHM Software History Center.

Happy golden anniversary, Unix!

UNIX BOOK II IDENTIFICATIONS AND NOTES

These may be PDP-7 assembly listings for the floating-point arithmetic operations that was among the first software that Ken Thompson and Dennis Ritchie had to create in order to develop the game *Space Travel* for the PDP-7 starting in 1969.

In Ritchie's "The Evolution of the Unix Time-sharing System" he writes: "Also during 1969, Thompson . . . and I rewrote *Space Travel* to run on this machine ['a little-used PDP-7 computer with an excellent display processor; the whole system was used as a Graphic-2 terminal']. The undertaking was more ambitious than it might seem; because we disdained all existing software, we had to write a floating-point arithmetic package, the point-wise specification of the graphic characters for the display, and a debugging system . . . All this was written in assembly language . . ."

Warren Toomey believes that the "fops" listing from pp. 2–15 represents mathematics functions like multiplication, division, sine, cosine, square root, and others.

pp.17–18
PDP-7 assembly listing for "ln"

Unix command for creating links to files.

pp. 20–24
PDP-7 assembly listing for "ls"

Unix command for listing file names.

pp. 26–34
PDP-7 assembly listing for "moo"

Number guessing game, available in 1970 on Multics and in 1968 on University of Cambridge mainframe. A version of the mind or paper game Bulls and Cows.

pp. 36–39
PDP-7 assembly listing for "nm"

pp. 42–43
“op”

A list of definitions, showing the instruction values for all of the PDP-7 assembly mnemonic codes and also numbers for the Unix system calls.

pp. 45–63
PDP-7 assembly listing for what may be a simulation or game for billiards or pool.

pp. 65
PDP-7 assembly listing for “pd”

Unidentified program.

Might “pd” stand for “previous directory”?

pp. 67–71
PDP-7 assembly listing for “psych”

Unidentified program.

pp. 73
PDP-7 assembly listing for “rm”

Unix command for removing files.

pp. 75
PDP-7 assembly listing for "rn"

Possibly a Unix command for renaming, or moving, files that was later implemented as the “mv” command.

pp. 77–92
PDP-7 assembly listing for “roff”

The first Unix text-formatting program.

Unix command for file system salvage, reconstructing the file system.

pp. 100–106
PDP-7 assembly listing for "sh"

The Thompson Shell, the Unix command interpreter.

pp. 109–136
PDP-7 assembly listing for *Space Travel*.

Space Travel is a computer game that was central to the beginning of the Unix effort at Bell Labs.

pp. 138–139
PDP-7 assembly listing for "stat"

Unix command that provides status information about files.

pp. 141–142
PDP-7 assembly listing for "tm"

A command that performs the system call time and performs a conversion. Likely an early version of the Unix command "date."

pp. 145–169
PDP-7 assembly listings for "t1," "t2," "t3," "t4," "t5," "t6," "t7," and "t8"

Unidentified program.

Perhaps an interpreter for a programming language? B?

pp. 172–183
PDP-7 assembly listings for "ttt1"

Tic Tac Toe game.

Unidentified program.

Presumably related to Tic Tac Toe.

pp. 190
PDP-7 assembly listing for “un”

Unix command for finding undefined symbols.

NOTES

1. https://www.bell-labs.com/usr/dmr/www/hist.html

2. https://spectrum.ieee.org/computing/software/the-top-programming-languages-2019

ABOUT THE AUTHOR

David C. Brock is an historian of technology and director of the CHM Software History Center. He focuses on histories of computing, electronics and instrumentation, as well as on oral history.

JOIN THE DISCUSSION

6 Comments CHM Website 🔒 Login

♡ Favorite 1 Tweet Share Sort by Best

OR SIGN UP WITH DISQUS ?

Name

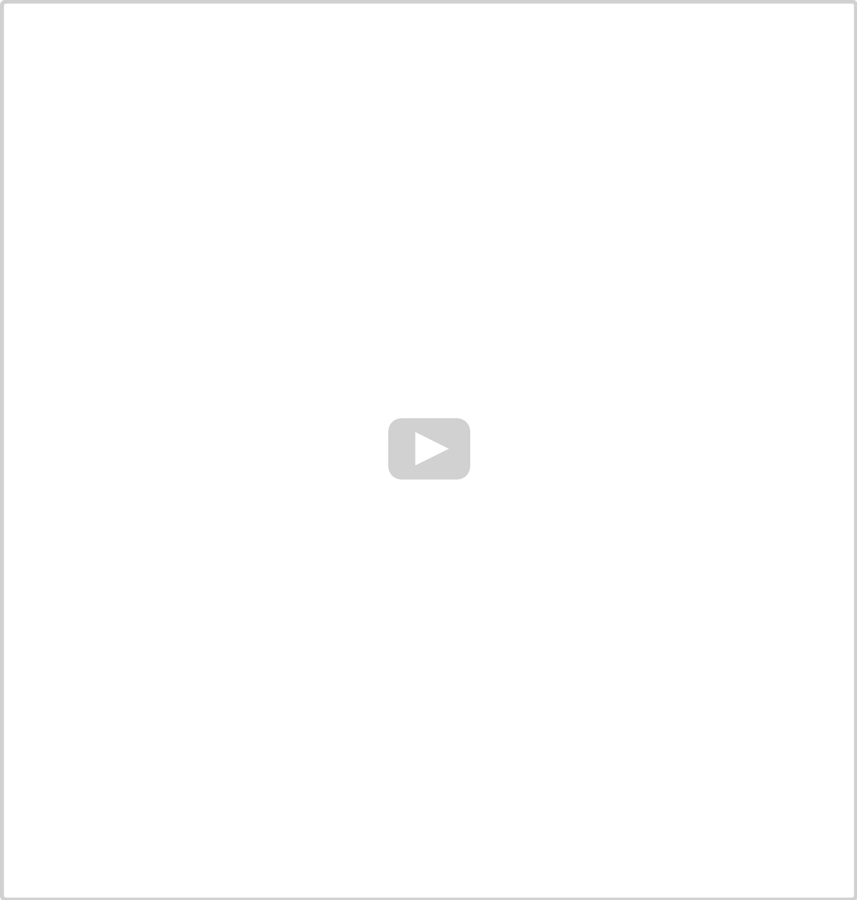
Shabbadeux • 2 years ago

pd looks somehow involved in finding the "parent directory," given how it traverses "..". It may have been involved in linking the current directory to the parent directory, based on some of the shenanigans I see at the end for writing info about "." into the file created for "..".

I have to say I'm deep in heavy guessing here, though.

1 ^ | ⌄ • Reply • Share ›

Cynde Moya • 2 years ago



See this listing of Unix System 0 booting again on Living Computers: Museum + Labs restored PDP-7 computer.

^ | ⌄ • Reply • Share ›

since Unix was developed. The OS's developed prior to Unix were not only rock solid, but they were, and for those still in use, secure. Face it, using a language that doesn't natively protect from buffer overflow is guaranteed to create security issues.

⌃ | ⌄ 1 • Reply • Share ›

Carlos A. Osuna ➦ Mike Ober • 2 years ago

This is a classic flexibility versus security dilema.

Corporate companies who wanted rock solid security flooded towards the IBM AS/400 which was Object Oriented, built like a Fortress and permanently struck in its past.

On the other hand, both the PDP, the VAX and latter on, the PCs were open systems whose flexibility far outweighed the compromise of inherent insecurity.

Ironically, the Apple Lisa and Apple Macintosh, with their tightly integrated Windowing and Program structure in Object Pascal oversaw most of the problems coming, but were also defense less against the power of flexibility.

So your argument is basically mute as just LPARing a C based OS like Linux inside a close system like the PowerSystems running IBM i can correct the deficiencies you stated above.

1 ⌃ | ⌄ • Reply • Share ›

sodapop7 ➦ Mike Ober • 2 years ago

Master it

⌃ | ⌄ • Reply • Share ›

Mike Ober ➦ sodapop7 • 2 years ago

Master what?

C, C++, or the millions of lines of code in a modern operating system. The Linux's driver for one of the most common WiFi chip sets was just

CHM BLOG

Meet the Adams Brothers

November 09, 2021

CHM BLOG

For Whom Does Data Work?

March 09, 2021

CHM BLOG

Thinking about Machines and Thinking

January 28, 2021

View all articles

SHARE