Unix Architecture Evolution: Milestones and Lessons Learned

Diomidis Spinellis
Department of Management Science and Technology
Athens University of Economics and Business

www.spinellis.gr
@CoolSWEng


LIVE AID


WWW


FORECLOSURE
You Tube
digg

Overview

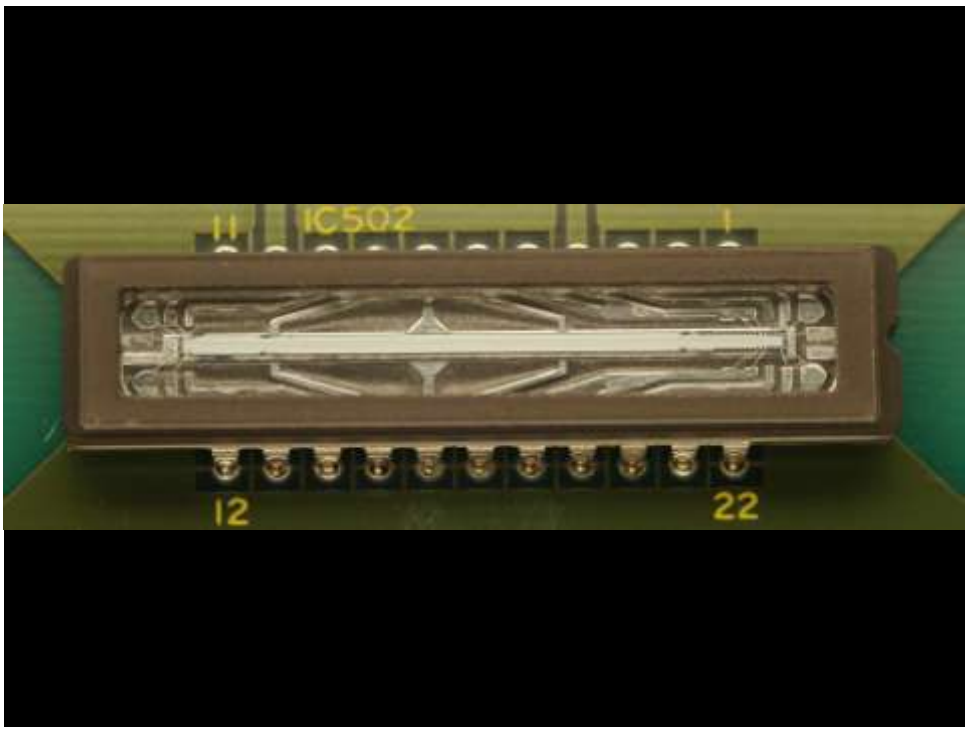
- Groundwork
- Sources
- Important architectural milestones
- Evolution in numbers
- Evolution in words

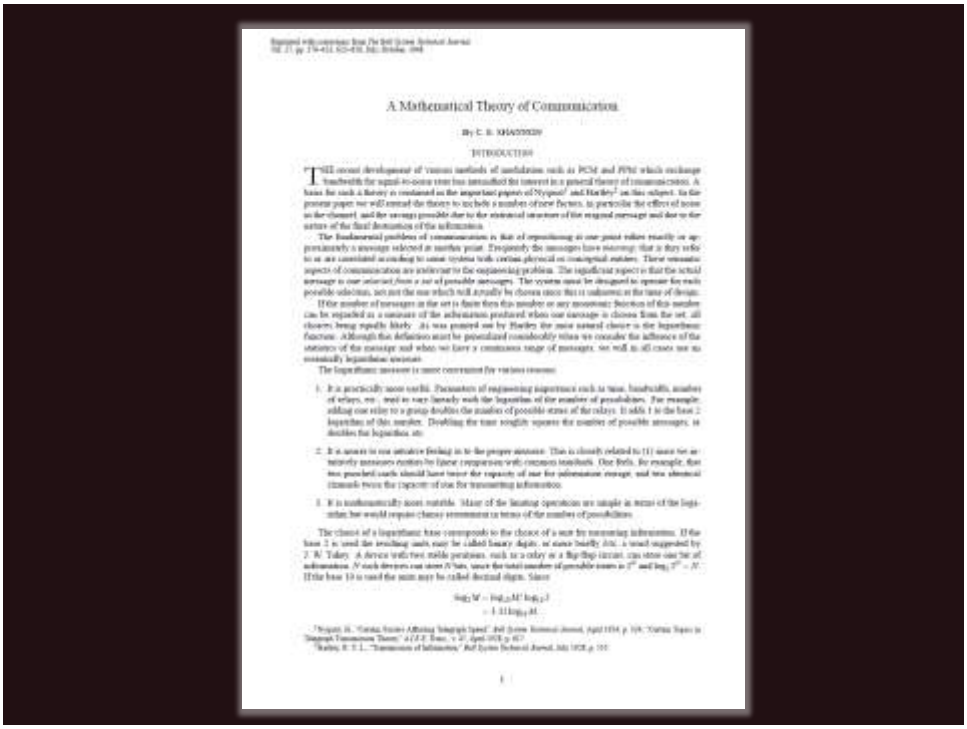

VAX 11/780


ALFR.
NOBEL


ACM
A.M. TURING AWARD


microelectronics group
Lucent Technologies
Bell Labs Innovations
A replica of the first transistor, invented at Bell Labs, December 23, 1947
50 Years and Counting...


IC502
11
1
12
22



Reprinted with corrections from *The Bell System Technical Journal*
Vol. 27, pp. 379–423, 623–656, July, October, 1948.

A Mathematical Theory of Communication

By C. E. SHANNON

INTRODUCTION

THE recent development of various methods of modulation such as PCM and PPM which exchange bandwidth for signal-to-noise ratio has intensified the interest in a general theory of communication. A basis for such a theory is contained in the important papers of Nyquist[1] and Hartley[2] on this subject. In the present paper we will extend the theory to include a number of new factors, in particular the effect of noise in the channel, and the savings possible due to the statistical structure of the original message and due to the nature of the final destination of the information.

The fundamental problem of communication is that of reproducing at one point either exactly or approximately a message selected at another point. Frequently the messages have meaning; that is they refer to or are correlated according to some system with certain physical or conceptual entities. These semantic aspects of communication are irrelevant to the engineering problem. The significant aspect is that the actual message is one *selected from* a set of possible messages. The system must be designed to operate for each possible selection, not just the one which will actually be chosen since this is unknown at the time of design.

If the number of messages in the set is finite then this number or any monotonic function of this number can be regarded as a measure of the information produced when one message is chosen from the set, all choices being equally likely. As was pointed out by Hartley the most natural choice is the logarithmic function. Although this definition must be generalized considerably when we consider the influence of the statistics of the message and when we have a continuous range of messages, we will in all cases use an essentially logarithmic measure.

The logarithmic measure is more convenient for various reasons:

1. It is practically more useful. Parameters of engineering importance such as time, bandwidth, number of relays, etc., tend to vary linearly with the logarithm of the number of possibilities. For example, adding one relay to a group doubles the number of possible states of the relays. It adds 1 to the base 2 logarithm of this number. Doubling the time roughly squares the number of possible messages, or doubles the logarithm, etc.

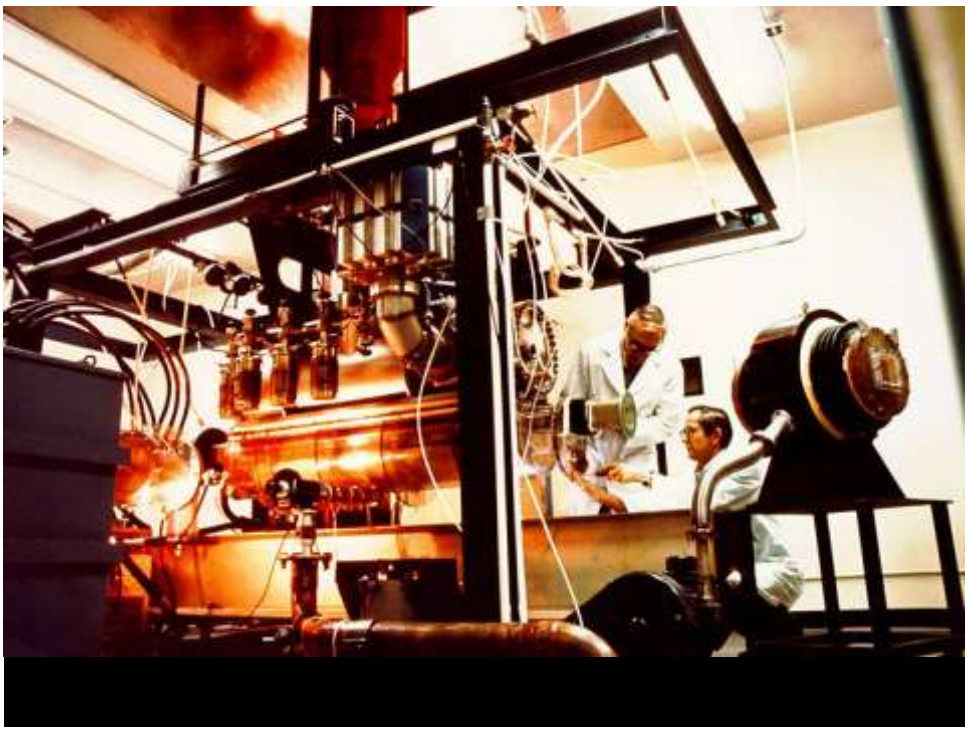
2. It is nearer to our intuitive feeling as to the proper measure. This is closely related to (1) since we intuitively measures entities by linear comparison with common standards. One feels, for example, that two punched cards should have twice the capacity of one for information storage, and two identical channels twice the capacity of one for transmitting information.

3. It is mathematically more suitable. Many of the limiting operations are simple in terms of the logarithm but would require clumsy restatement in terms of the number of possibilities.

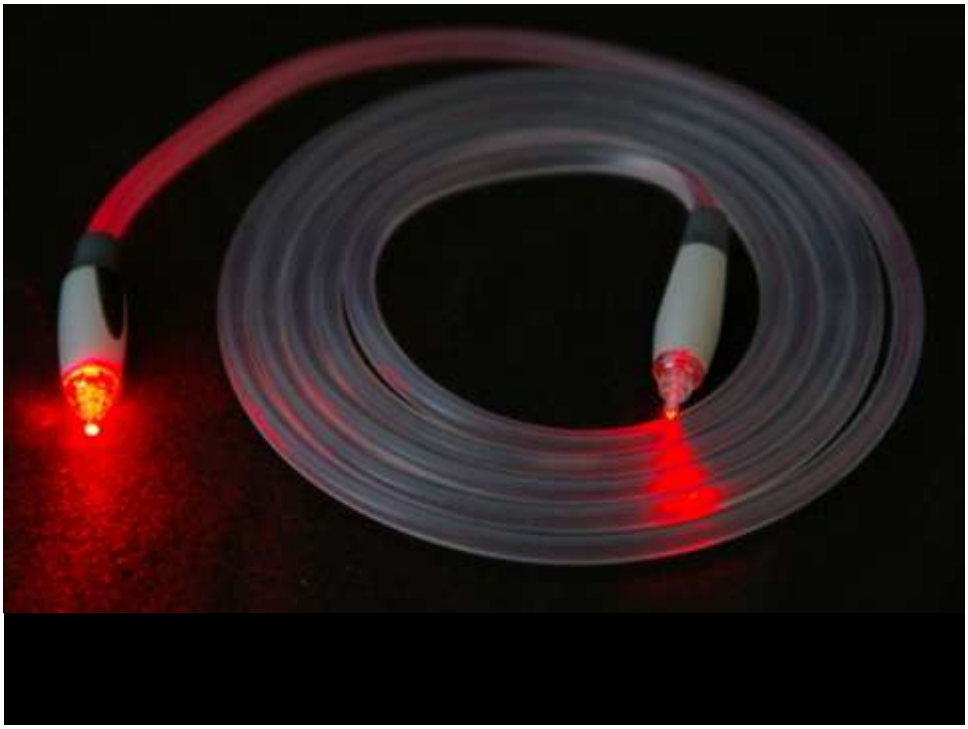
The choice of a logarithmic base corresponds to the choice of a unit for measuring information. If the base 2 is used the resulting units may be called binary digits, or more briefly *bits*, a word suggested by J. W. Tukey. A device with two stable positions, such as a relay or a flip-flop circuit, can store one bit of information. N such devices can store N bits, since the total number of possible states is 2^N and $\log_2 2^N = N$. If the base 10 is used the units may be called decimal digits. Since

$$\log_2 M = \log_{10} M / \log_{10} 2$$
$$= 3.32 \log_{10} M$$

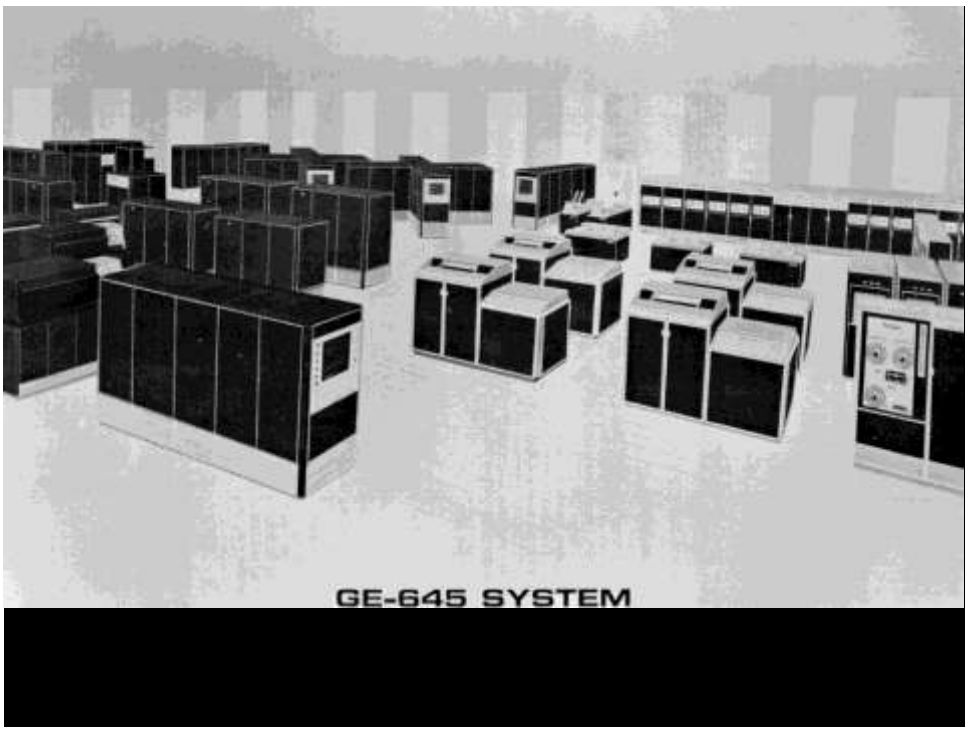
[1] Nyquist, H., "Certain Factors Affecting Telegraph Speed", *Bell System Technical Journal*, April 1924, p. 324; "Certain Topics in Telegraph Transmission Theory," *A.I.E.E. Trans.*, v. 47, April 1928, p. 617.
[2] Hartley, R. V. L., "Transmission of Information," *Bell System Technical Journal*, July 1928, p. 535.

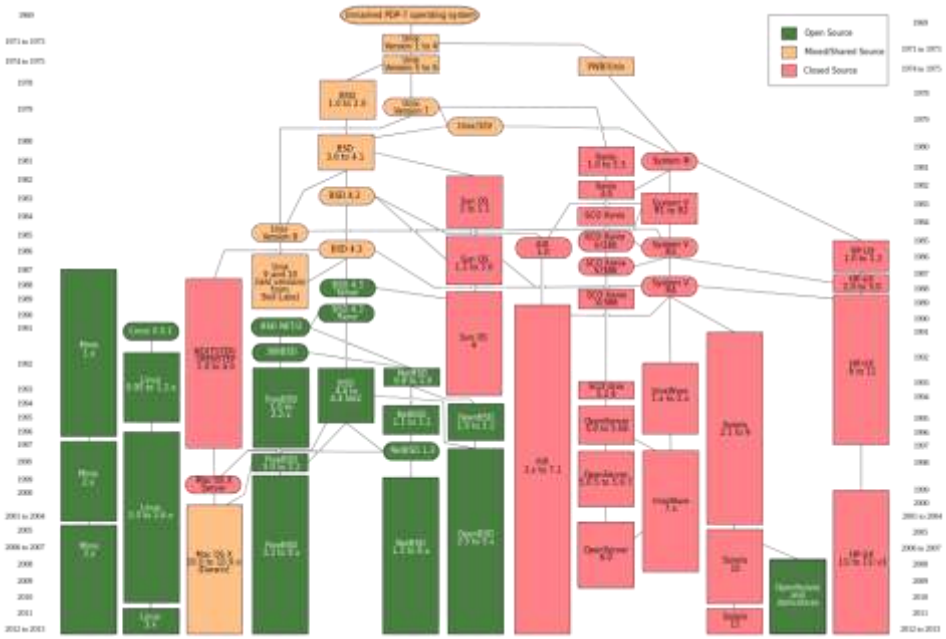



THE
C
PROGRAMMING
LANGUAGE
Brian W. Kernighan • Dennis M. Ritchie
THE
C++
PROGRAMMING LANGUAGE
BJARNE
STROUSTRUP
The AWK
Programming
Language




LIVE FREE OR DIE
UNIX*
TRADEMARK OF BELL LABS*


GE-645 SYSTEM



Why Unix is important

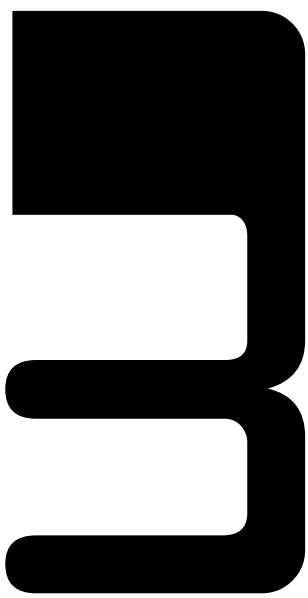
- Exemplar design
- Technical contributions,
- Impact
- Development model
- Widespread use
- “unusual simplicity, power, and elegance”

System technology

- Hierarchical file system
- Compatible file, device, networking, and inter-process I/O
- Pipes and filters architecture
- Virtual file systems
- The shell as a user-selectable regular process

Associated Technologies

- C and C++
- Parser and lexical analyzer generators
- Software development environments
- Document preparation tools and declarative markup
- Scripting languages
- TCP/IP networking
- Configuration management systems



A. T. & T. SETTLES ANTITRUST CASE; SHARES PATENTS

U.S. Hails Consent Decree as Major Victory — Company Calls Terms 'Stringent'

By ANTHONY LEWIS

Special to The New York Times.

WASHINGTON, Jan. 24—An antitrust suit against the American Telephone and Telegraph Company was settled today on terms described by Government lawyers as a major victory.

Herbert Brownell Jr., Attorney General, announced the signing of a consent decree in the Federal Court in Newark, N. J. Under the terms of the settlement A. T. & T. must:

¶License 8,600 existing patents to all applicants without royalties.

¶License all its other patents, present and future, to any American concern at "reasonable and nondiscriminatory" rates.

¶Get out of all business not directly connected with the communications field.

¶Maintain uniform cost accounting methods for its manufacturing subsidiary, Western Electric.

One of 'Most Important'

Stanley N. Barnes, Assistant Attorney General in charge of the Justice Department's Antitrust Division, said the decree was "one of the most important" in antitrust history. Another department lawyer called it "miraculous."

In New York, Cleo F. Craig, president of A. T. & T., acknowledged that the terms of the consent decree were "stringent." However, he said, the settlement will leave intact "the unique combination and teamwork of the operating companies, the Bell Telephone Laboratories and the Western Electric Company that over the years has produced for the people of this country the finest, most widely used and most progressive telephone service in the world."

The A. T. & T. case was one of three major antitrust suits brought by the Government in the electronics field since World War II. The others, involving the Radio Corporation of America and International Business Machines, also are in negotiation for possible consent settlement. The I. B. M. negotiations are believed to be almost finished.

Through subsidiary Bell operating companies, A. T. & T. controls a majority of the country's telephone lines. Western Electric, its wholly owned subsidiary, makes the equipment for all Bell companies.

U. S. Pressed Civil Suit

On Dec. 31, 1954, the assets of A.T.&T. and the Bell system were estimated at $13,000,000,000.

The Government complaint, filed in 1949, charged that A. T. & T. and Western Electric had "unlawfully restrained and monopolized trade and commerce in the manufacture, distribution, sale and installation of telephone equipment."

It was a civil suit. The Government was not calling for a fine but wanted the courts to order changes in A. T. & T.'s structure. Specifically, the Government asked that the parent corporation give up its interest in Western Electric, that Western Electric be dissolved and its assets divided among three other companies.

The judgment entered today allows Western Electric to continue as manufacturer to the Bell System. However, several

Continued on Page 16, Column 4

The New York Times
Published: January 25, 1956

CALDERA

240 West Center Street
Orem, Utah 84057
Tel: 801.765.4999 Fax 801.765.4481

January 23, 2002

Dear UNIX® enthusiasts,

Caldera International, Inc. hereby grants a fee free license that includes the right to use, modify and distribute this named source code, including creating derived binary products created from the source code. The source code for which Caldera International, Inc. grants rights are limited to the following UNIX Operating Systems that operate on the 16-bit PDP-11 CPU and only versions of the 32-bit UNIX Operating System, with specific exclusion of UNIX System III and UNIX System V and successor operating systems:

32-bit AT/V UNIX
16 bit UNIX Versions 1, 2, 3, 4, 5, 6, 7

Caldera International, Inc. makes no guarantees or commitments that any source code is available from Caldera International, Inc.

The following copyright notice applies to the source code files for which this License is granted.

Copyright(C) Caldera International Inc. 2001-2002. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code and documentation must retain the above copyright notice, this list of conditions and the following disclaimer. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

All advertising materials mentioning features or use of this software must display the following acknowledgement:

This product includes software developed or owned by Caldera International, Inc.

Neither the name of Caldera International, Inc. nor the names of other contributors may be used to endorse or promote products derived from this software without specific prior written permission.

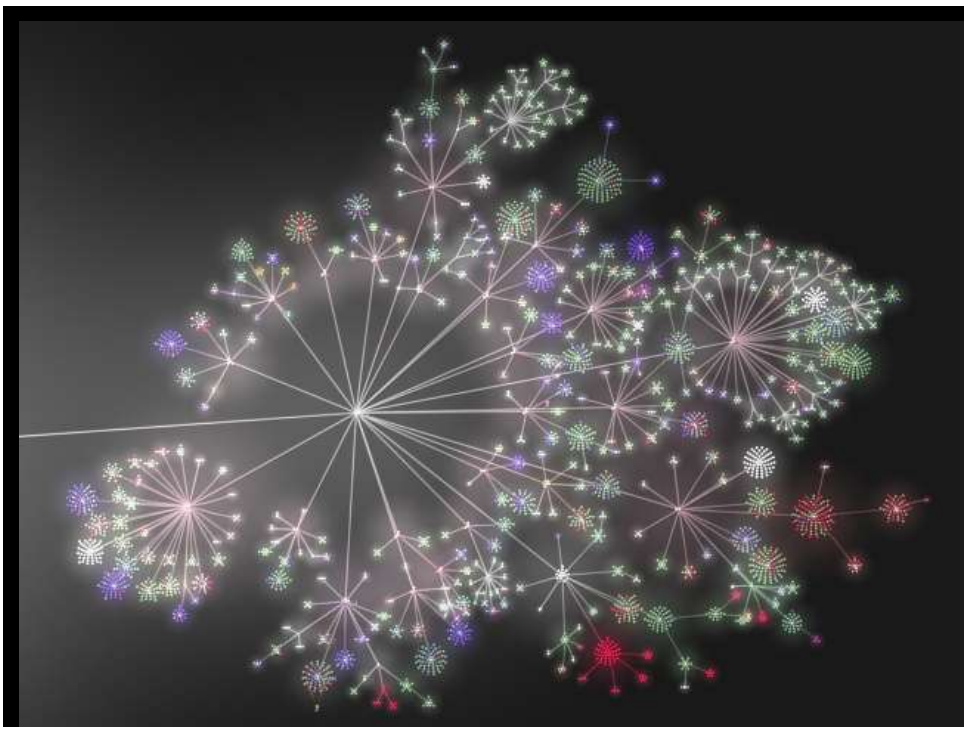
USE OF THE SOFTWARE PROVIDED FOR UNDER THIS LICENSE BY CALDERA INTERNATIONAL, INC. AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL CALDERA INTERNATIONAL, INC. BE LIABLE FOR ANY DIRECT, INDIRECT INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Very truly yours,

/signed/ Bill Broderick

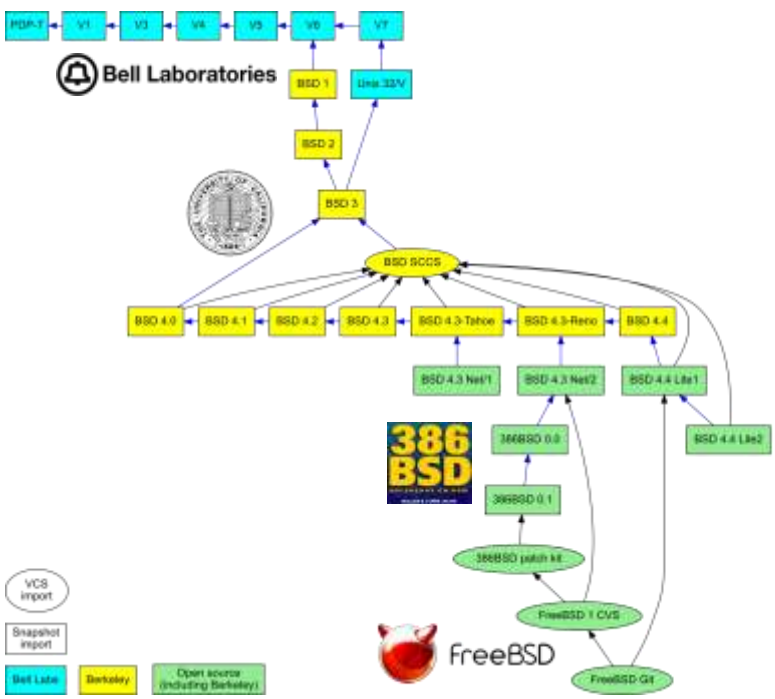
Bill Broderick
Director, Licensing Services

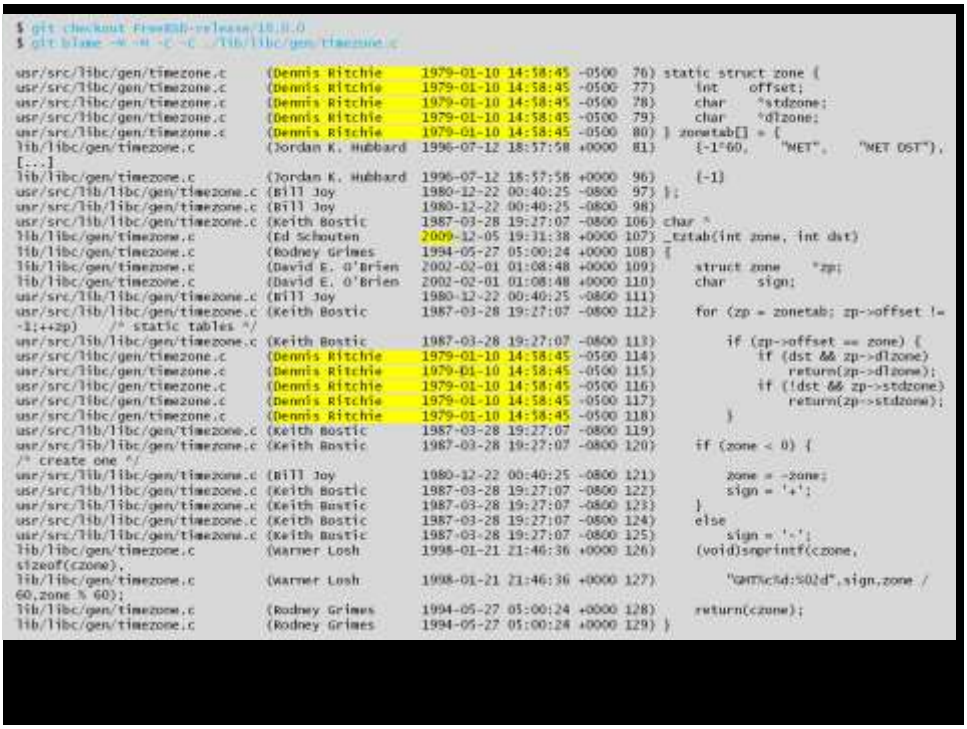
* UNIX is a registered trademark of The Open Group in the US and other countries.



Motivation

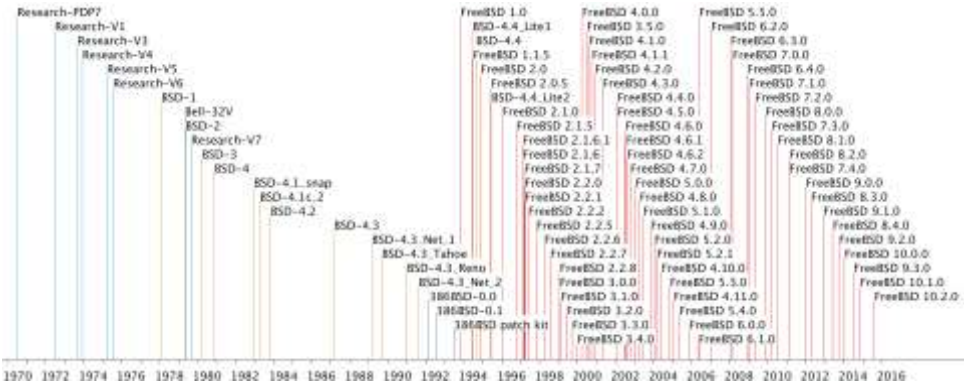
- Explore evolution of programming style
- Consolidate digital artifacts of historical importance
- Collect and record history that is fading away
- Provide a data set of digital archeology and repository mining

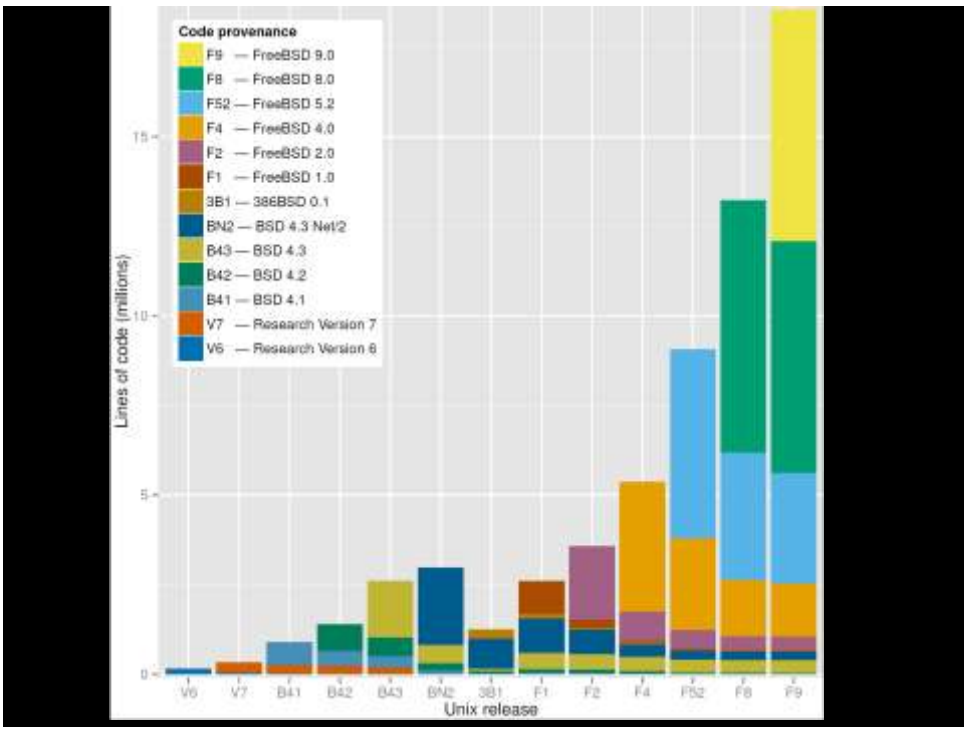


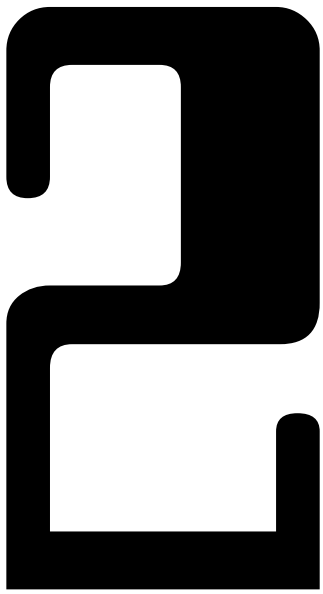


In Numbers …

Metric	Unix history	Linux history
Start date	30/06/1970	17/09/1991
Start files	43	92
Start lines	11,500	917,812
End files	63,049	51,396
End lines	27,388,943	21,525,436
Data set size (.git)	1.1GB	1.0GB
Number of commits	495,622	611,735
Number of merges	2,523	48,821
Number of authors	973	18,465
Days with activity	13,004	5,126


Research-PDP7
Research-V1
Research-V3
Research-V4
Research-V5
Research-V6
BSD-1
Bell-32V
BSD-2
Research-V7
BSD-3
BSD-4
BSD-4.1_snap
BSD-4.1c.2
BSD-4.2
BSD-4.3
BSD-4.3_Net.1
BSD-4.3_Tahoe
BSD-4.3_Reno
BSD-4.3_Net.2
386BSD-0.0
386BSD-0.1
386BSD patch kit
FreeBSD 1.0
BSD-4.4_Lite1
BSD-4.4
FreeBSD 1.1.5
FreeBSD 2.0
FreeBSD 2.0.5
BSD-4.4_Lite2
FreeBSD 2.1.0
FreeBSD 2.1.5
FreeBSD 2.1.6.1
FreeBSD 2.1.6
FreeBSD 2.1.7
FreeBSD 2.2.0
FreeBSD 2.2.1
FreeBSD 2.2.2
FreeBSD 2.2.5
FreeBSD 2.2.6
FreeBSD 2.2.7
FreeBSD 2.2.8
FreeBSD 3.0.0
FreeBSD 3.1.0
FreeBSD 3.2.0
FreeBSD 3.3.0
FreeBSD 3.4.0
FreeBSD 4.0.0
FreeBSD 3.5.0
FreeBSD 4.1.0
FreeBSD 4.1.1
FreeBSD 4.2.0
FreeBSD 4.3.0
FreeBSD 4.4.0
FreeBSD 4.5.0
FreeBSD 4.6.0
FreeBSD 4.6.1
FreeBSD 4.6.2
FreeBSD 4.7.0
FreeBSD 5.0.0
FreeBSD 4.8.0
FreeBSD 5.1.0
FreeBSD 4.9.0
FreeBSD 5.2.0
FreeBSD 5.2.1
FreeBSD 4.10.0
FreeBSD 5.3.0
FreeBSD 4.11.0
FreeBSD 5.4.0
FreeBSD 6.0.0
FreeBSD 6.1.0
FreeBSD 5.5.0
FreeBSD 6.2.0
FreeBSD 6.3.0
FreeBSD 7.0.0
FreeBSD 6.4.0
FreeBSD 7.1.0
FreeBSD 7.2.0
FreeBSD 8.0.0
FreeBSD 7.3.0
FreeBSD 8.1.0
FreeBSD 8.2.0
FreeBSD 7.4.0
FreeBSD 9.0.0
FreeBSD 8.3.0
FreeBSD 9.1.0
FreeBSD 8.4.0
FreeBSD 9.2.0
FreeBSD 10.0.0
FreeBSD 9.3.0
FreeBSD 10.1.0
FreeBSD 10.2.0
1970 1972 1974 1976 1978 1980 1982 1984 1986 1988 1990 1992 1994 1996 1998 2000 2002 2004 2006 2008 2010 2012 2014 2016

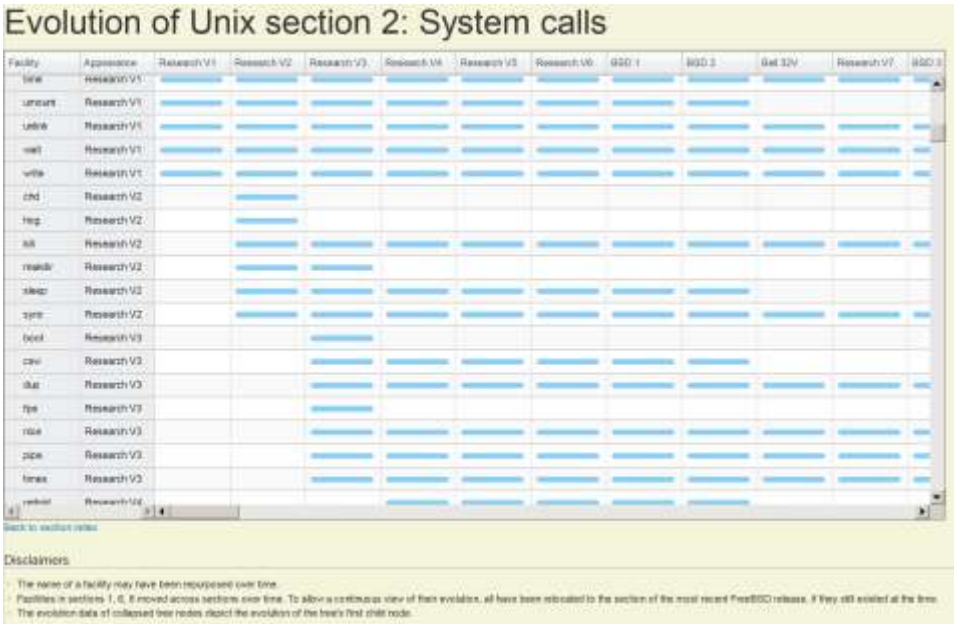

Code provenance
F9 — FreeBSD 9.0
F8 — FreeBSD 8.0
F52 — FreeBSD 5.2
F4 — FreeBSD 4.0
F2 — FreeBSD 2.0
F1 — FreeBSD 1.0
3B1 — 386BSD 0.1
BN2 — BSD 4.3 Net/2
B43 — BSD 4.3
B42 — BSD 4.2
B41 — BSD 4.1
V7 — Research Version 7
V6 — Research Version 6
Lines of code (millions)
0
5
10
15
V6
V7
B41
B42
B43
BN2
3B1
F1
F2
F4
F52
F8
F9
Unix release

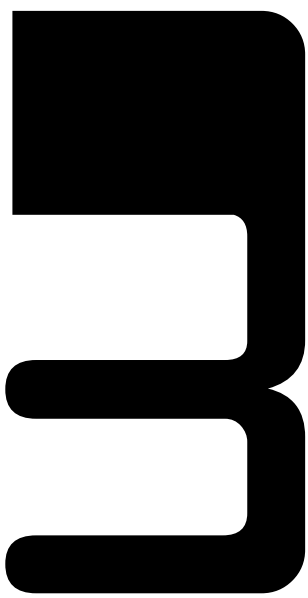


dspinellis.github.io/unix-history-man

Evolution of Unix Facilities

1. User commands
2. System calls
3. C library functions
4. Devices and special files
5. File formats and conventions
6. Games et. al.
7. Miscellanea
8. System maintenance procedures and commands
9. System kernel interfaces


Evolution of Unix section 2: System calls
Facility
Appearance
Research V1
Research V2
Research V3
Research V4
Research V5
Research V6
BSD 1
BSD 3
Net 32V
Research V7
BSD 3
Research V1
umount
Research V1
unlink
Research V1
wait
Research V1
write
Research V1
chd
Research V2
fstg
Research V2
kill
Research V2
mkdir
Research V2
sleep
Research V2
sync
Research V2
boot
Research V3
csv
Research V3
dup
Research V3
fpe
Research V3
nice
Research V3
pipe
Research V3
times
Research V3
Back to section index
Disclaimers
The name of a facility may have been repurposed over time.
Facilities in sections 1, 6, 8 moved across sections over time. To allow a continuous view of their evolution, all have been relocated to the section of the most recent FreeBSD release, if they still existed at the time.
The evolution data of collapsed tree nodes depict the evolution of the tree's first child node.



UNIX PROGRAMMER'S MANUAL

Third Edition

K. Thompson

D. M. Ritchie

February, 1973

UNIX PROGRAMMER'S MANUAL

Fourth Edition

K. Thompson
D. M. Ritchie

November, 1973


Milestones


Observations
Numbers & Insights

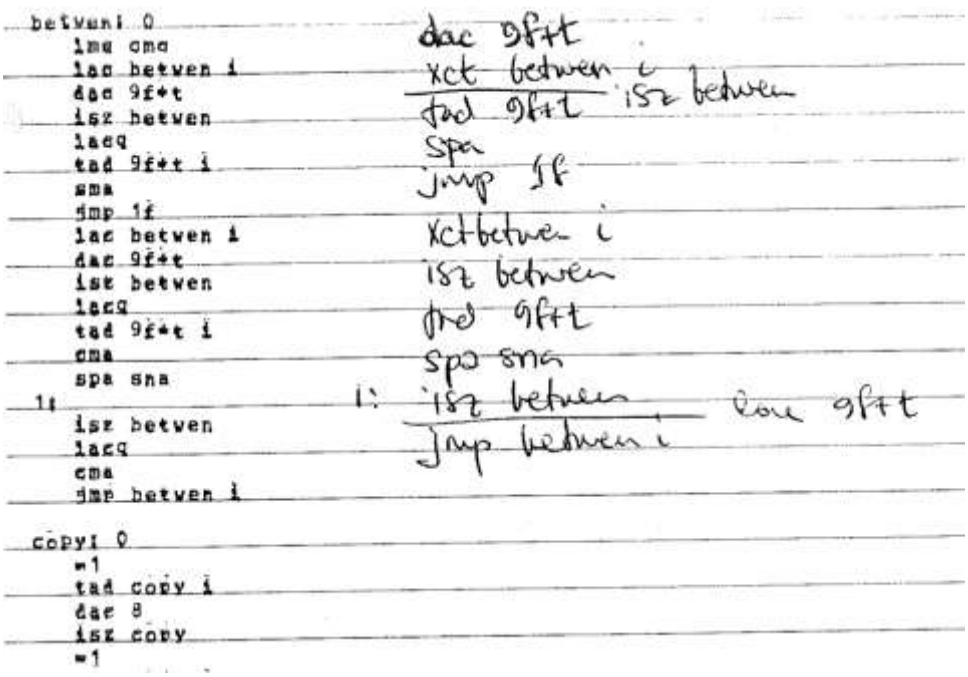
Things to Take Away …

- Architectural evolution in practice
- 1.1GB Git repository
  - github.com/dspinellis/unix-history-repo
- Open source project
  - github.com/dspinellis/unix-history-make
- Architectural lessons to apply

- Joint work with **Paris Avgeriou** Professor (chair of Software Engineering) and Head of the SEARCH (Software Engineering and Architecture) Group of the Johann Bernoulli Institute for Mathematics and Computer Science at the University of *Groningen* (RuG).

PDP-7 [Unix] (1970)

- Kernel
- Layering and Partitioning
- System Call
- Data Scoping
- Interpreter
- Monolithic Implementation
- Process Management
- Descriptor Management
- Separation of File Metadata from File Naming
- Devices as Files
- File I/O
- Filesystem



```
between, 0
    lmq cma
    lac between i
    dac 9f+t
    isz between
    lacq
    tad 9f+t i
    sma
    jmp 1f
    lac between i
    dac 9f+t
    isz between
    lacq
    tad 9f+t i
    cma
    spa sna
1:  isz between
    lacq
    cma
    jmp between i

copy, 0
    -1
    tad copy i
    dac 8
    isz copy
    -1
```

```
dac 9f+t
xct between i
tad 9f+t   isz between
spa
jmp 1f
xct between i
isz between
tad 9f+t
spa sna
i: isz between,   law 9f+t
jmp between i
```

Kernel

- 2584 lines
- Loads and executes user-level commands
- Provides the file abstraction
- Virtualizes the hardware interfaces
- Establishes ownership of files

Layering and Partitioning

adm.s	cat.s	dskio.s	init.s	s6.s
ald.s	check.s	dskres.s	lcase.b	s7.s
apr.s	chmod.s	dsksav.s	maksys.s	s8.s
as.s	chown.s	ds.s	s1.s	s9.s
bc.s	chrm.s	dsw.s	s2.s	scope.v
bi.s	cp.s	ed1.s	s3.s	sop.s
bl.s	db.s	ed2.s	s4.s	trysys.s
cas.s	dmabs.s	ind.b	s5.s	

Process Management (fork)

```
.fork1
    jms lookfor; 0 / not-used
        skp
        jms error
    dac 9f*t
    isz uniqpid
    lac uniqpid
    dac u.ac
    lav sysexit
    dac u.swapret
    lac o200000
    tad u.ulistp i
    dac u.ulistp i
    jms dsksvap; 07000
    lac 9f*t
    dac u.ulistp
    lac o100000
    xor u.ulistp i
    dac u.ulistp i
    lac u.pid
```

Descriptor Management

```
fget: 0
    jms betwen; d0; d9
      jmp fget i
    cli; mul; 8
    iac
    tad ofilepp
    dac 9f+t
    dac .+2
    jms copy; ..; fnode; 3
    isz fget
    jmp fget i

fput: 0
    iac 9f+t
    dac .+3
    jms copy; fnode; ..; 3
    jmp fput i
t = t+1
```

Separation of File Metadata from File Naming

```
inode:
    i.flags: .=.+1
    i.dskps: .=.+7
    i.uid: .=.+1
    i.nlks: .=.+1
    i.size: .=.+1
    i.uniq: .=.+1
        . = inode+12
```

```
namei: 0
    jms iget
    -1
    tad namei i
    dac 9f+t+1
    isz namei
    iac i.flags
    and o20
    sna
    jmp namei i
    -8
    tad i.size
    cma
    irss 3
    dac 9f+t
    sna
    jmp namei i
    dzm di
```

Devices as Files

```
ttyin:
    <tt>;<yi>;<n 040;040040
ttyout:
    <tt>;<yo>;<ut>; 040040
keybd:
    <ke>;<yb>;<oa>;<rd>
displ:
    <di>;<sp>;<la>;<y 040
sh:
    <sh>; 040040;040040;040040
system:
    <sy>;<st>;<em>; 040040
```

File I/O

- open
- read
- write
- seek
- tell
- close

Filesystem

- creat
- rename
- link
- unlink

Interpreter

```
main 5(
   extrn read, write;
   auto i, c, state, line 100;

loop:
   state = i = 0;
loop1:
   c = read();
   if(c==4) return;
   if(c=='*' & state==0) state = 2;
   if((c<'0' | c>'9'&c<'a' | c>'z') & state==0) state = 1;
   line[i] = c;
   i = i+1;
   if(c!=012) goto loop1;
   if(state==2 | i==1) goto noi;
   write('  ');
   write(' ');
noi:
   i = 0;
loop3:
   c = line[i];
   write(c);
   i = i+1;
   if(c!=012) goto loop3;
   goto loop;
}
```

ind.b

/* ind */

First Research Edition (Nov 1971)

- System Calls
- Binary-Code API
- Abstraction of Standard I/O
- Generic File I/O Layer
- User-Contributed Tools and Games
- The Shell as a User Program
- Interoperability through Documented File Formats
- Tree Directory Structure
- Mountable Filesystem Interface

Bell Laboratories

subject: Study of UNIX

date: September 14, 1972

from: T. R. Bashkow

Messrs. W. S. Bartlett
D. P. Clayton
D. H. Copp
Mmes. G. J. Hansen
J. Hintz
Mr. L. J. Kelly
Miss R. L. Klein
Messrs. J. J. Ludwig
J. F. Maranzano
Mrs. G. Pettit
Messrs. J. E. Ritacco
B. A. Tague
D. W. Vogel
Mrs. L. S. Wright

On Tuesday, September 19, at 9:30 a.m. in Room 2A-418 at Murray Hill, I will give a talk on my study of the UNIX operating system. The emphasis will be on the structure, functional components, and internal operation of the system.

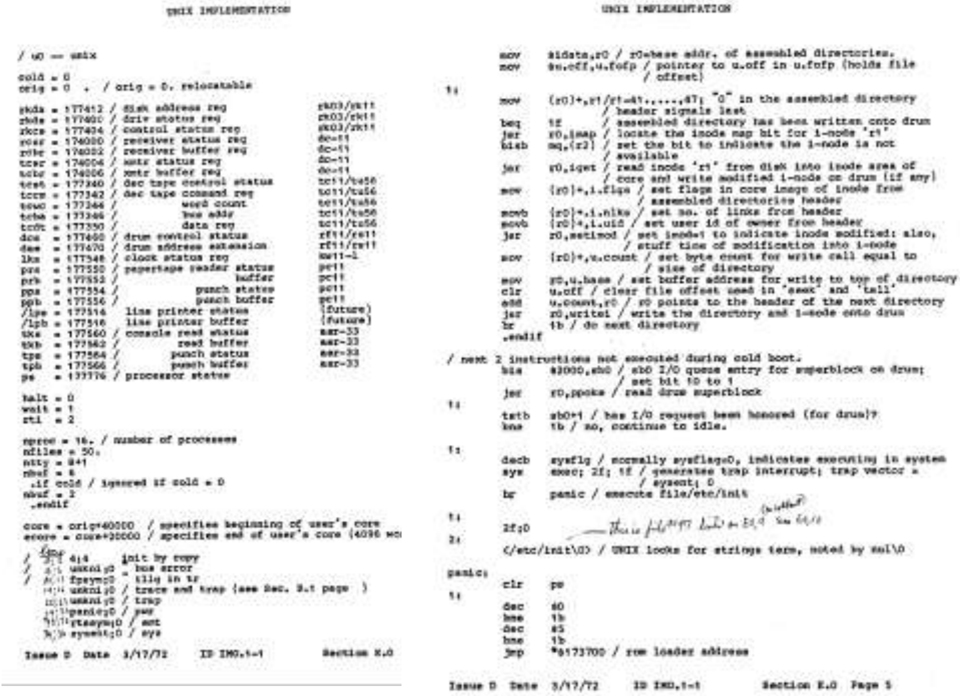
MH-8234-TRB-mbh

T. R. Bashkow

T R Bashkow

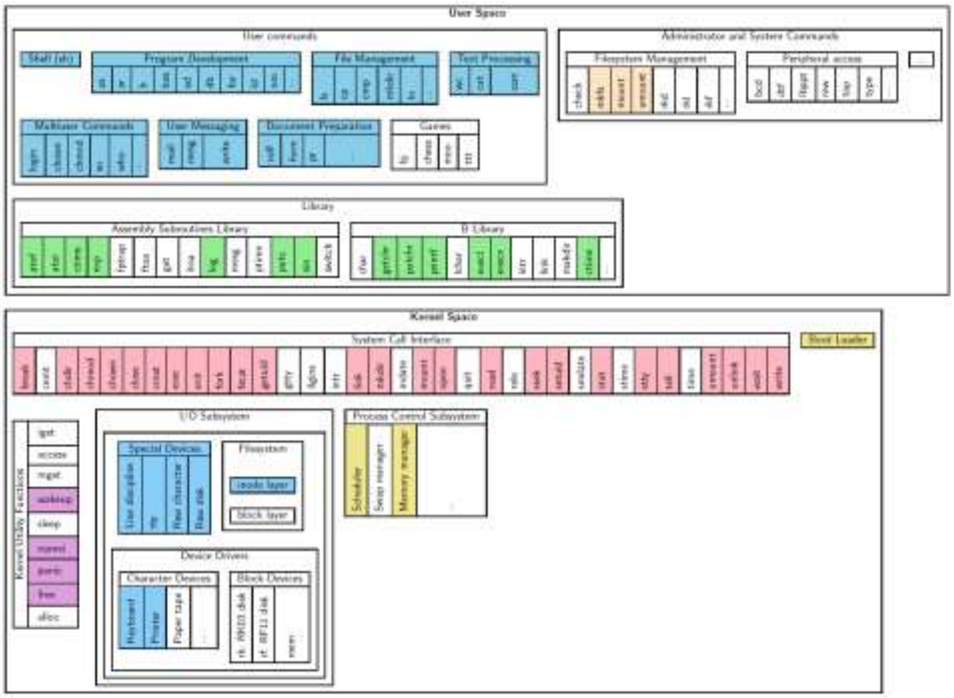
Copy to
Mr. G. L. Baldwin

```
/ initialize inodes for special files (inodes 1 to 40.)

        mov     $40.,r1 / set r1=i-node-number 40.
1:
        jsr     r0,iget / read i-node 'r1' from disk into inode area of
                        / core and write modified inode out (if any)
        mov     $100017,i.flgs / set flags in core image of inode to indi-
                               / cate allocated, read (owner, non-owner),
                               / write (owner, non-owner)
        movb    $1,i.nlks / set no. of links = 1
        movb    $1,i.uid / set user id of owner = 1
        jsr     r0,setimod / set imod=1 to indicate i-node modified, also
                           / stuff time of modification into i-node
        dec     r1 / next i-node no. = present i-node no.-1
        bgt     1b / has i-node 1 been initialized; no, branch

/ initialize i-nodes r1.,...,47. and write the root device, binary, etc.,
/ directories onto fixed head disk.  user temporary, initialization prog.
```



```
UNIX IMPLEMENTATION

/ u0 -- unix

cold = 0
orig = 0   .  / orig = 0. relocatable

rkda = 177412 / disk address reg              rk03/rk11
rkds = 177400 / driv status reg               rk03/rk11
rkcs = 177404 / control status reg            rk03/rk11
rcsr = 174000 / receiver status reg           dc-11
rcbr = 174002 / receiver buffer reg           dc-11
tcsr = 174004 / xmtr status reg               dc-11
tcbr = 174006 / xmtr buffer reg               dc-11
tcst = 177340 / dec tape control status       tc11/tu56
tccm = 177342 / dec tape command reg          tc11/tu56
tcwc = 177344 /          word count           tc11/tu56
tcba = 177346 /          bus addr             tc11/tu56
tcdt = 177350 /          data reg             tc11/tu56
dcs  = 177460 / drum control status           rf11/rs11
dae  = 177470 / drum address extension        rf11/rs11
lks  = 177546 / clock status reg              kw11-L
prs  = 177550 / papertape reader status       pc11
prb  = 177552 /                  buffer       pc11
pps  = 177554 /             punch status      pc11
ppb  = 177556 /             punch buffer      pc11
/lps = 177514   line printer status           (future)
/lpb = 177516   line printer buffer           (future)
tks  = 177560 / console read status           asr-33
tkb  = 177562 /         read buffer           asr-33
tps  = 177564 /        punch status           asr-33
tpb  = 177566 /        punch buffer           asr-33
ps   = 177776 / processor status

halt = 0
wait = 1
rti  = 2

nproc = 16. / number of processes
nfiles = 50.
ntty = 8+1
nbuf = 6
 .if cold / ignored if cold = 0
nbuf = 2
 .endif

core = orig+40000  / specifies beginning of user's core
ecore = core+20000 / specifies end of user's core (4096 wo

/     4;4     init by copy
/     unkni;0 - bus error
/     fpsym;0 - illg in tr
      unkni;0 / trace and trap (see Sec. B.1 page  )
      unkni;0 / trap
      panic;0 / pwr
      rtssym;0 / emt
      sysent;0 / sys

Issue D  Date  3/17/72     ID IMO.1-1          Section E.0
```

```
UNIX IMPLEMENTATION

        mov     $idata,r0 / r0=base addr. of assembled directories.
        mov     $u.off,u.fofp / pointer to u.off in u.fofp (holds file
                               / offset)
1:
        mov     (r0)+,r1 / r1=41.,...,47; "0" in the assembled directory
                        / header signals last
        beq     1f      / assembled directory has been written onto drum
        jsr     r0,imap / locate the inode map bit for i-node 'r1'
        bisb    mq,(r2) / set the bit to indicate the i-node is not
                        / available
        jsr     r0,iget / read inode 'r1' from disk into inode area of
                        / core and write modified i-node on drum (if any)
        mov     (r0)+,i.flgs / set flags in core image of inode from
                             / assembled directories header
        movb    (r0)+,i.nlks / set no. of links from header
        movb    (r0)+,i.uid / set user id of owner from header
        jsr     r0,setimod / set imod=1 to indicate inode modified; also,
                          / stuff time of modification into i-node
        mov     (r0)+,u.count / set byte count for write call equal to
                              / size of directory
        mov     r0,u.base / set buffer address for write to top of directory
        clr     u.off / clear file offset used in 'seek' and 'tell'
        add     u.count,r0 / r0 points to the header of the next directory
        jsr     r0,writei / write the directory and i-node onto drum
        br      1b / do next directory
        .endif

/ next 2 instructions not executed during cold boot.
        bis     $2000,sb0 / sb0 I/O queue entry for superblock on drum;
                          / set bit 10 to 1
        jsr     r0,ppoke / read drum superblock
1:
        tstb    sb0+1 / has I/O request been honored (for drum)?
        bne     1b / no, continue to idle.

1:
        decb    sysflg / normally sysflag=0, indicates executing in system
        sys     exec; 2f; 1f / generates trap interrupt; trap vector =
                             / sysent; 0
        br      panic / execute file/etc/init

1:
        2f;0
2:
        </etc/init\0> / UNIX looks for strings term, noted by nul\0

panic:
        clr     ps
1:
        dec     $0
        bne     1b
        dec     $5
        bne     1b
        jmp     *$173700 / rom loader address

Issue D  Date  3/17/72     ID IMO.1-1          Section E.0  Page 5
```



First Edition Unix 1972 FreeBSD 11.1 2018

```
sysrele / 0
sysexit / 1
sysfork / 2
sysread / 3
syswrite / 4
sysopen / 5
sysclose / 6
syswait / 7
syscreat / 8
syslink / 9
sysunlink / 10
```

```
0   { int nosys(void); } syscall nosys_args int
1   { void sys_exit(int rval); } exit \
        sys_exit_args void
2   { int fork(void); }
3   { ssize_t read(int fd, void *buf, \
        size_t nbyte); }
4   { ssize_t write(int fd, const void *buf, \
        size_t nbyte); }
5   { int open(char *path, int flags, int mode); }
6   { int close(int fd); }
7   { int wait4(int pid, int *status, \
        int options, struct rusage *rusage); }
8   { int creat(char *path, int mode); }
9   { int link(char *path, char *link); }
10  { int unlink(char *path); }
```

Issue D Date 3/17/72 ID IMO.1-1 Section E.1 Page 1

The Shell as a User Program

11/3/71 PASSWD (V)

NAME passwd -- password file

SYNOPSIS --

DESCRIPTION passwd contains for each user the following information:

name (login name)
password
numerical user ID
default working directory
program to use as Shell

This is an ASCII file. Each field within each user's entry is separated from the next by a colon. Each user is separated from the next by a new-line. If the password field is null, no password is demanded; if the Shell field is null, the Shell itself is used.

Abstraction of Standard I/O

11/3/71 SH (I)

Two characters cause the immediately following string to be interpreted as a special argument to the shell itself, not passed to the command. An argument of the form "<arg" causes the file arg to be used as the standard input file of the given command; an argument of the form ">arg" causes file "arg" to be used as the standard output file for the given command.

Interoperability through Documented File Formats

V. FILE FORMATS

a.out	assembler and loader output
archive	archive file
bppt	binary paper tape format
core	core image file
directory	directory format
file system	file system format
passwd	password file
uids	map names to user ID's
utmp	logged-in user information

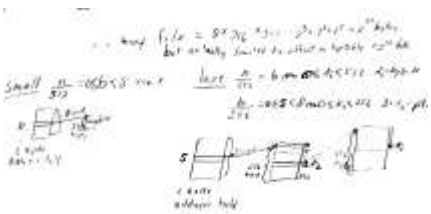
Format	Description	Clients
a.out	Assembler and linker output	as, ld, strip, nm, un
Archive	Object code libraries	ar, ld
Core	Crashed program image	Kernel, db
Directory	File system directories	du, find, ls, ln, mkdir, rmdir
File system	File system format	check, dump,* mkfs, restor*
Ident	GECOD ident card format	opr
Password	User accounts and passwords	chown, find, getpw,* login,* ls, passwd*
Tape*	DECtape file format	mt,* tap*
Uid	User identifier to name map	chown
utmp	Logged in users	init, login,* who,* write*
wtmp*	Users login history	acct, date, init, login, tacct, who

User-Contributed Tools and Games

```
VI.  USER MAINTAINED PROGRAMS

basic ..................... DEC supplied BASIC
bj ........................ the game of black jack
cal ....................... print calendar
chess ..................... the game of chess
das ....................... disassembler
dli ....................... load DEC binary paper tapes
dpt ....................... read DEC ASCII paper tapes
moo ....................... the game of MOO
sort ...................... sort a file
ttt ....................... the game of tic-tac-toe
```

Tree Directory Structure

- mkdir(II)
- chdir(II)



- chdir(I)
- find(I)
- ln(I)
- ls(I)
- stat(I)
- mkdir(I)
- mv(I)
- rm(I)
- rmdir(I)

Mountable Filesystem Interface

- mount(II)
- umount(II)
- mount(I)
- umount(I)

Second Research Edition (Jun 1972)

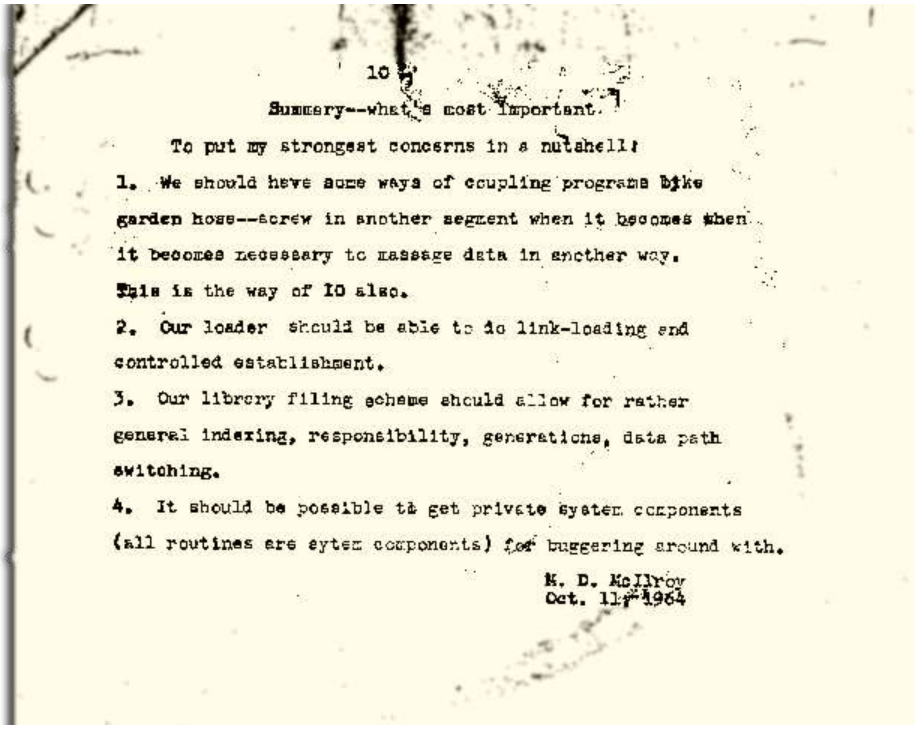
- Software Library

III. SUBROUTINES

atan	arctangent
atof	convert ASCII to floating
atoi	convert ASCII to integer
const	floating-point constants
ctime	convert time to ASCII
exp	exponential function
fptrap	floating-point simulator
ftoa	convert floating to ASCII
gerts	communicate with GCOS
getc	get character
hypot	compute hypotenuse
itoa	convert integer to ASCII
log	logarithm base e
mesg	print string on typewriter
nlist	read name list
ptime	print time
putc	write character or word
qsort	quicker sort
salloc	storage allocator
sin	sine, cosine
sqrt	square root
switch	transfer depending on value

Third Research Edition (Feb 1973)

- Pipes and Filters



10

Summary--what's most important.

To put my strongest concerns in a nutshell:

1. We should have some ways of coupling programs like garden hose--screw in another segment when it becomes then it becomes necessary to massage data in another way. This is the way of IO also.

2. Our loader should be able to do link-loading and controlled establishment.

3. Our library filing scheme should allow for rather general indexing, responsibility, generations, data path switching.

4. It should be possible to get private system components (all routines are sytem components) for buggering around with.

M. D. McIlroy
Oct. 11, 1964

Fourth Research Edition (Nov 1973)

- Structured Programming
- User Groups
- Language-Independent API
- Data Structure Definition Reuse
- Dynamic Resource Management
- Device Driver Abstraction
- Buffer Cache

Structured Programming

- Kernel implemented in “New B”
- 6373 lines New B
- 768 lines PDP-11 assembly
- 105 functions + 50 assembly symbols
- vs 248 global symbols in the First Edition

Language-Independent API

PIPE (II) 8/5/73 PIPE (II)

NAME

pipe – create a pipe

SYNOPSIS

(pipe = 42.)
sys pipe
(read file descriptor in r0)
(write file descriptor in r1)

pipe(fildes)
int fildes[2];

DESCRIPTION

The *pipe* system call creates an I/O mechanism called a pipe. The file descriptors returned can be used in read and write operations. When the pipe is written using the descriptor returned in r1 (resp. fildes[1]), up to 4096 bytes of data are buffered before the writing process is suspended. A read using the descriptor returned in r0 (resp. fildes[0]) will pick up the data.

It is assumed that after the pipe has been set up, two (or more) cooperating processes (created by subsequent *fork* calls) will pass data through the pipe with *read* and *write* calls.

The shell has a syntax to set up a linear array of processes connected by pipes.

Read calls on an empty pipe (no buffered data) with only one end (all write file descriptors closed) return an end-of-file. Write calls under similar conditions are ignored.

Data Structure Definition & Reuse

```
buf.h   filsys.h  proc.h   text.h
conf.h  inode.h   reg.h    tty.h
file.h  param.h   systm.h  user.h
```

Dynamic Resource Management

```
int     coremap[CMAPSIZ];
int     swapmap[SMAPSIZ];

struct map {
        char *m_size;
        char *m_addr;
};

malloc(mp, size)
struct map *mp;
{
 …
```

Device Driver Abstraction

IV. SPECIAL FILES

cat	phototypesetter interface
da	voice response unit
dc	DC-11 communications interface
dn	dn11 ACU interface
dp	dp11 201 data-phone interface
kl	KL-11/TTY-33 console typewriter
mem	core memory
pc	PC-11 paper tape reader/punch
rf	RF11/RS11 fixed-head disk file
rk	RK-11/RK03 (or RK05) disk
rp	RP-11/RP03 moving-head disk
tc	TC-11/TU56 DECtape
tiu	Spider interface
tm	TM-11/TU-10 magtape interface
vs	voice synthesizer interface
vt	11/20 (vt01) interface

Driver Interface

```
struct {
        int     (*d_open)();
        int     (*d_close)();
        int     (*d_strategy)();
} bdevsw[];

struct {
        int     (*d_open)();
        int     (*d_close)();
        int     (*d_read)();
        int     (*d_write)();
        int     (*d_sgtty)();
} cdevsw[];
```

Buffer Cache

Fourth Edition

```
#define B_READ  01
#define B_DONE  02
#define B_ERROR 04
#define B_BUSY  010
#define B_XMEM  060
#define B_WANTED 0100
#define B_RELOC 0200
#define B_ASYNC 0400
#define B_DELWRI 01000
```

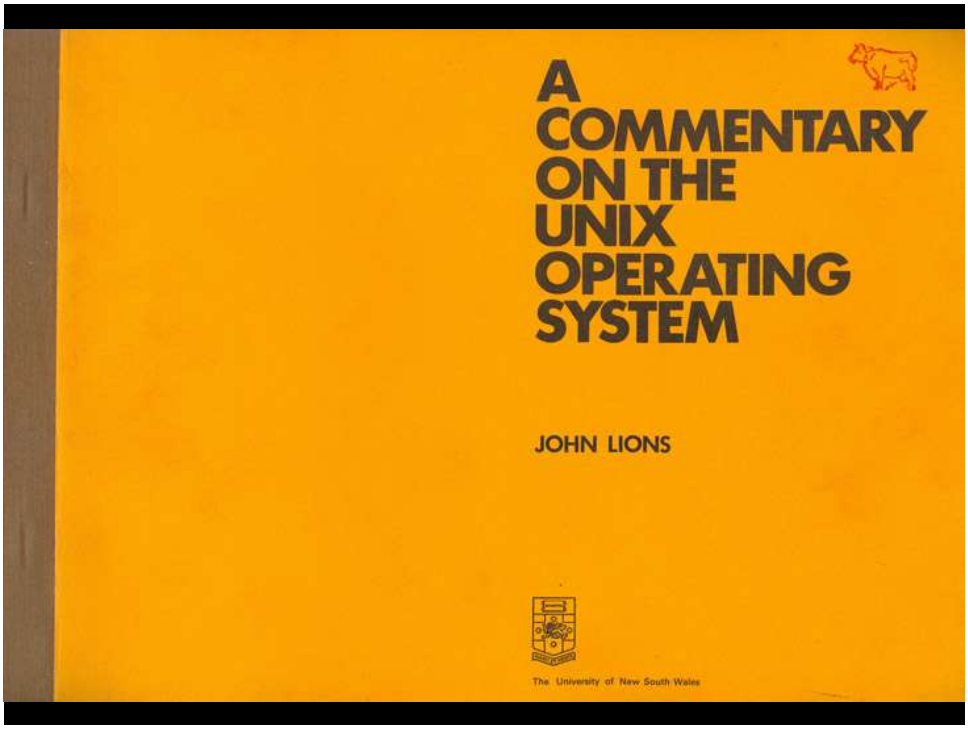
FreeBSD 11.1

```
#define B_ASYNC 0x00000004
 /* Start I/O, do not wait.
  */
[…]
#define B_DELWRI 0x00000080
/* Delay I/O until buffer
  reused. */
#define B_DONE 0x00000200
/* I/O completed. */
```

Fifth Research Edition (Jun 1974)

- Command Files

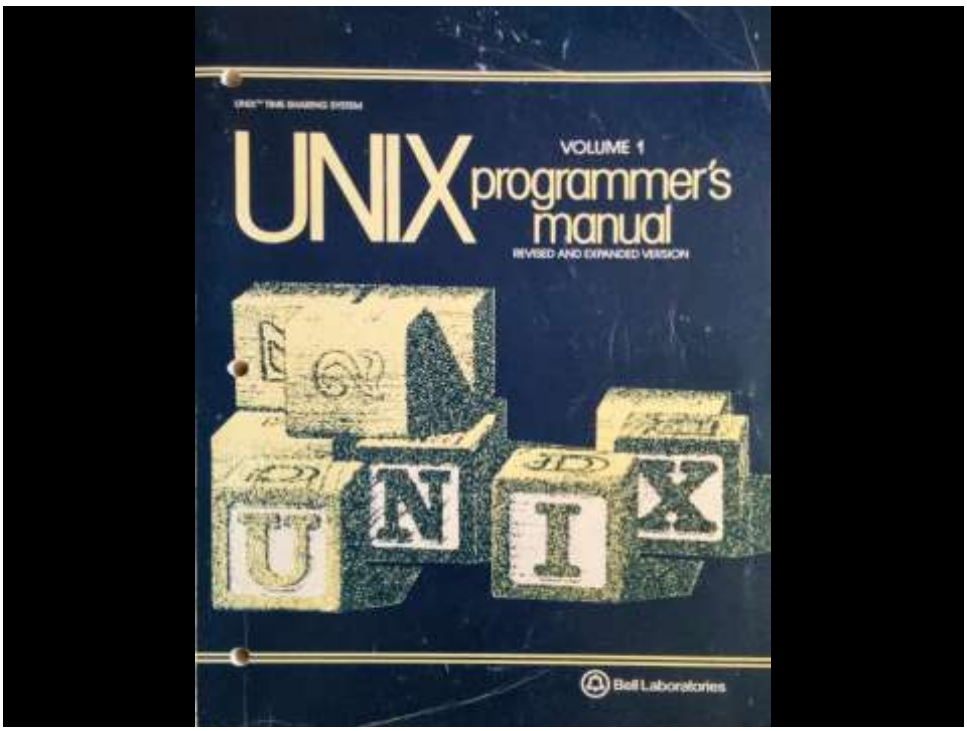
```
chdir /usr/source/s3
cc -c ctime.c
ar r /lib/liba.a ctime.o
rm ctime.o
chdir /usr/source/s1
cc -s -n date.c
cp a.out /bin/date
cc -s -n dump.c
cp a.out /bin/dump
cc -s -n ls.c
cp a.out /bin/ls
rm a.out
```



Sixth Research Edition (May 1975)

- Portable C Library

```
alloc.c   clenf.c   makbuf.c  scan1.c
calloc.c  copen.c   maktab.c  scan2.c
cclose.c  cputc.c   nexch.c   scan3.c
ceof.c    cwrd.c    nodig.c   system.c
cerror.c  dummy.s   printf.c  tmpnam.c
cexit.c   ftoa.c    putch.c   unget.c
cflush.c  getch.c   puts.c    unprnt.s
cfree.c   gets.c    relvec.c  wdleng.c
cgetc.c   getvec.c  revput.c
ciodec.c  iehzap.c  run
```



Seventh Research Edition (Jan 1979)

- Unix as a Virtual Machine
- Dynamic Memory Allocation
- Static Analysis
- Environment Variables
- Language Development Tools
- Domain-Specific Languages
- Filesystem Directory Hierarchy

Unix as a Virtual Machine

Also, about this time [1973] I had a fateful discussion with Dennis, in which he said
"I think it may be easier to port Unix to a new piece of hardware than to port a complex application from Unix to a new OS"

— Steve Johnson

Dynamic User Memory Allocation

- malloc(3), free(3)
- 26 programs: awk cc col cron dc dcheck diff ed eqn expr graph icheck learn ls m4 neqn nm quot ratfor spline struct tar tsort uucp xsend quiz
- stdio(3), mp(3)

Environment Variables

- KEY=value
- Kernel
- Shell
- C Library

ENVIRON(5) UNIX Programmer's Manual ENVIRON(5)

NAME
environ – user environment

SYNOPSIS
extern char **environ;

DESCRIPTION
An array of strings called the 'environment' is made available by *exec*(2) when a process begins. By convention these strings have the form 'name=value'. The following names are used by various commands:

PATH The sequence of directory prefixes that *sh*, *time*, *nice*(1), etc., apply in searching for a file known by an incomplete path name. The prefixes are separated by ':'. *Login*(1) sets PATH=:/bin:/usr/bin.

HOME A user's login directory, set by *login*(1) from the password file *passwd*(5).

TERM The kind of terminal for which output is to be prepared. This information is used by commands, such as *troff* or *plot*(1), which may exploit special terminal capabilities. See *term*(7) for a list of terminal types.

Further names may be placed in the environment by the *export* command and 'name=value' arguments in *sh*(1), or by *exec*(2). It is unwise to conflict with certain Shell variables that are frequently exported by '.profile' files: MAIL, PS1, PS2, IFS.

SEE ALSO
exec(2), sh(1), term(7), login(1)

Language Development Tools

- lex(1)
- yacc(1)
- 12 clients: awk bc cpp egrep eqn lex m4 make pcc neqn struct
-

THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 57, No. 6, July-August 1978
Printed in U.S.A.

UNIX Time-Sharing System:

Language Development Tools

By S. C. JOHNSON and M. E. LESK
(Manuscript received December 27, 1977)

The development of new programs on the UNIX system is facilitated by tools for language design and implementation. These are frequently program generators, compiling into C, which provide advanced algorithms in a convenient form, while not restraining the user to a preconceived set of jobs. Two of the most important such tools are Yacc, a generator of LALR(1) parsers, and Lex, a generator of regular expression recognizers using deterministic finite automata. They have been used in a wide variety of applications, including compilers, desk calculators, typesetting languages, and pattern processors.*

Domain-Specific Languages

- sh
- awk
- sed
- find
- expr
- egrep
- m4
- make

Filesystem Directory Hierarchy

First and Second Berkeley Software Distributions (1978)

- Software Packages
  - csh
  - ex
  - Mail
  - Pascal
  - termlib

3BSD (1979)

- Virtual Memory Paging
  - vm_*.c
  - 2808 out of 16039 C source code
  - 17% of kernel source code
  - vread(2), vwrite(2), vfork(2)

4BSD (Oct 1980)

- Regular Expression Library: regex(3)
- Optimized Screen Handling

Regular Expression Library: regex(3)

– 5 implementations: awk, sed, ed, grep, expr
– 1 client: more(1)
– 2 more by 4.3: dbx(1), rdist(1)
– 4 replacements in FreeBSD: ed, grep, sed, expr

Optimized Screen Handling

– curses(3)
– termcap(5)


login: root
Password:
Last login: Tue Jun 13 11:45:38 on console
4.3 BSD #3: Tue Jun 25 15:36:53 JST 1991
You have mail.
Terminal type is vt100
TTYPE is ascii
Erase set to Backspace
Kill set to Ctrl-U
Interrupt is Ctrl-C
Jun 13 11:46
SONY NET WORK STATION, Model NWS-1850, Machine ID #20910
4.3 BSD #3: Tue Jun 25 15:36:53 JST 1991
NEWS-OS Release 3.3ap [LP,RS] Copyright (c) 1988,1989 by Sony Corporation
real mem = 16777216
avail mem = 15409960
using 1510 buffers [dynamic allocate buffer]
Enable Internal Inst/Data Cache

4.2BSD (Sep 1983)

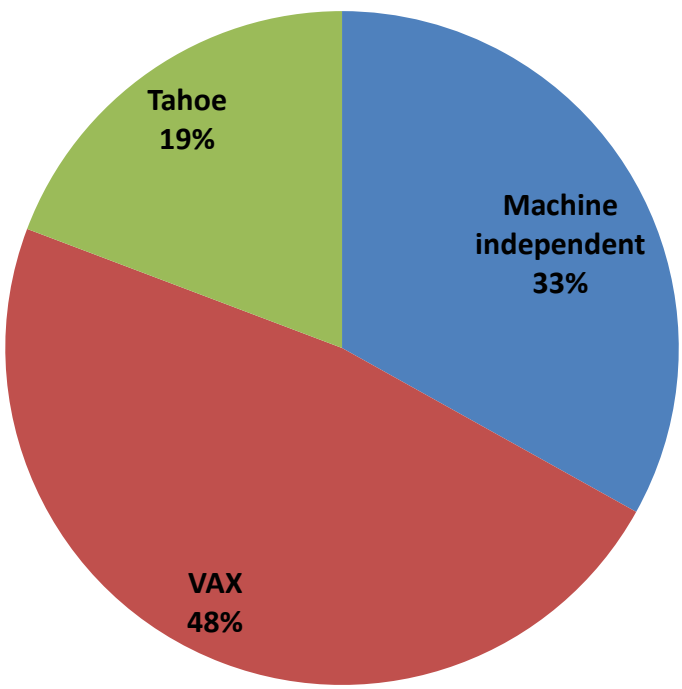
- Internet Protocol Family
  - ARP, IP, TCP, UDP, ICMP
- Local and Remote Interprocess Communication
  - socket(2), etc.
- Network and User Database Access
  - getfsent(3x), getgrent(3), gethostent(3n), getnetent(3n), getprotoent(3n), getpwent(3), getservent(3n)
- Pseudo-Terminal Driver
  - pty(4)

- 2 library functions
  - rcmd(3x)
  - rexec(3x)
- 11 system daemons
  - comsat(8c)
  - ftpd(8c)
  - gettable(8c)
  - implogd(8c)
  - rexecd(8c)
  - rlogind(8c)
  - af(8c)
  - rshd(8c)
  - rwhod(8c)
  - telnetd(8c)
  - tftpd(8c)
- 8 user-mode programs
  - ftp(1c)
  - rlogin(1c)
  - rsh(1c)
  - talk(1c)
  - telnet(1c)
  - tftp(1c)
  - whois(1c)
  - sendmail(1c)

System call	Uses
bind	23
connect	15
accept	13
select	12
listen	11
sendto	10
shutdown	9
recvfrom	8
getsockname	6
recv	2
send	2
sendmsg	1
getsockopt	0
recvmsg	0
socketpair	0

4.3BSD Tahoe (Jun 1988)

- Multiple CPU Architecture Support
  - VAX
  - CCI Power 6/32 (Tahoe)
- Timezone Handling
  - Community contribution
  - Separation of timezone rules from code



4.3BSD Reno (Jun 1990)

- Kernel Packet Forwarding Database
  - route(4)
  - routed(8), XNSrouted(8)
- Virtual Filesystem Interface
  - …

vnode

```
/*
 * Operations on vnodes.
 */
struct vnodeops {
        int     (*vn_lookup)(           /* ndp */ );
        int     (*vn_create)(           /* ndp, fflags, vap, cred */ );
        int     (*vn_mknod)(            /* ndp, vap, cred */ );
        int     (*vn_open)(             /* vp, fflags, cred */ );
        int     (*vn_close)(            /* vp, fflags, cred */ );
        int     (*vn_access)(           /* vp, fflags, cred */ );
        int     (*vn_getattr)(          /* vp, vap, cred */ );
        int     (*vn_setattr)(          /* vp, vap, cred */ );

        int     (*vn_read)(             /* vp, uiop, offp, ioflag, cred */ );
        int     (*vn_write)(            /* vp, uiop, offp, ioflag, cred */ );
        int     (*vn_ioctl)(            /* vp, com, data, fflag, cred */ );
        int     (*vn_select)(           /* vp, which, cred */ );
        int     (*vn_mmap)(             /* vp, ..., cred */ );
        int     (*vn_fsync)(            /* vp, fflags, cred */ );
        int     (*vn_seek)(             /* vp, (old)offp, off, whence */ );
….
```

4.3BSD Net/2 (Jun 1991)

- Stream I/O Functions
  - funopen(3)
  - GNU funopencookie(3) added in FreeBSD 11

4.4BSD (Jun 1994)

- Stackable Filesystems
  - mount_null(8)
  - mount_union(8)
- Generic System Control Interface (MIB)
  - sysctl(1)
  - sysctl(3)
  - sysctl(9)

386BSD Patch Kit (1992-1993)

- Organized Community Contributions
  - From open source software …
  - … to an open source **project**
- Patch metadata
  - title
  - author
  - description
  - prerequisites

FreeBSD 1.1 (May 1994)

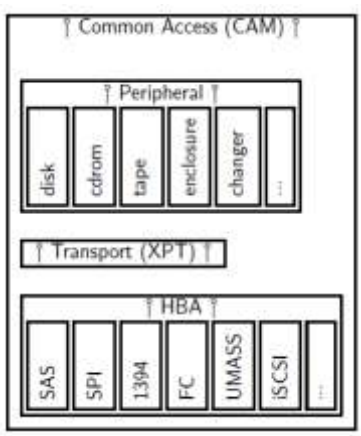
- Package Manager
  - Patch
  - Compile
  - Install
  - Uninstall
  - Handling of dependencies

FreeBSD 2.0 (Nov 1994)

- Process Filesystem
  - procfs(5)
- Dynamically Loadable Kernel Modules
  - lkm(4), then kld(4)
  - device drivers
  - file systems
  - emulators
  - system calls
  - **992** modules in FreeBSD 11.1

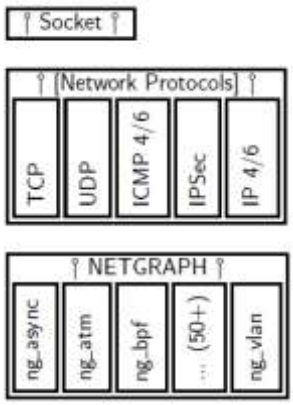
FreeBSD 3.0.0 (Jan 1999)

- Common Access Method I/O Subsystem (CAM)



FreeBSD 3.4.0 (Dec 1999)

- Graph-based Kernel Networking and User Library (NETGRAPH)

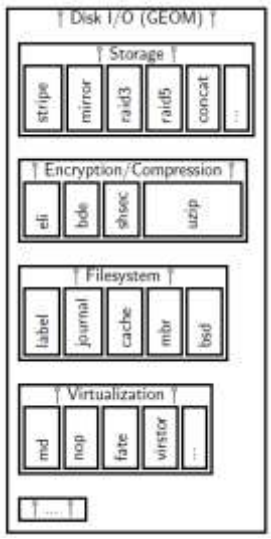


FreeBSD 4.0.0 (Mar 2000)

- OpenSSL Secure Sockets Layer and Transport Layer Security framework
  - Version 0.9.4
  - 1127 files, 227118 lines
  - libssl(3), libcrypto(3), openssl(1)
- Jail: Isolate a process and its descendants

FreeBSD 5.0.0 (May 2006)

- Modular Disk I/O Request Transformation Framework (GEOM)



FreeBSD 5.3.0 (Nov 2004)

- Streaming Archive Access Library
- Miniport Driver Wrapper

FreeBSD 7.0.0 (Feb 2008)

- ZFS Filesystems and Storage Pools

FreeBSD 7.1.0 (Dec 2008)

- Dynamic Tracing

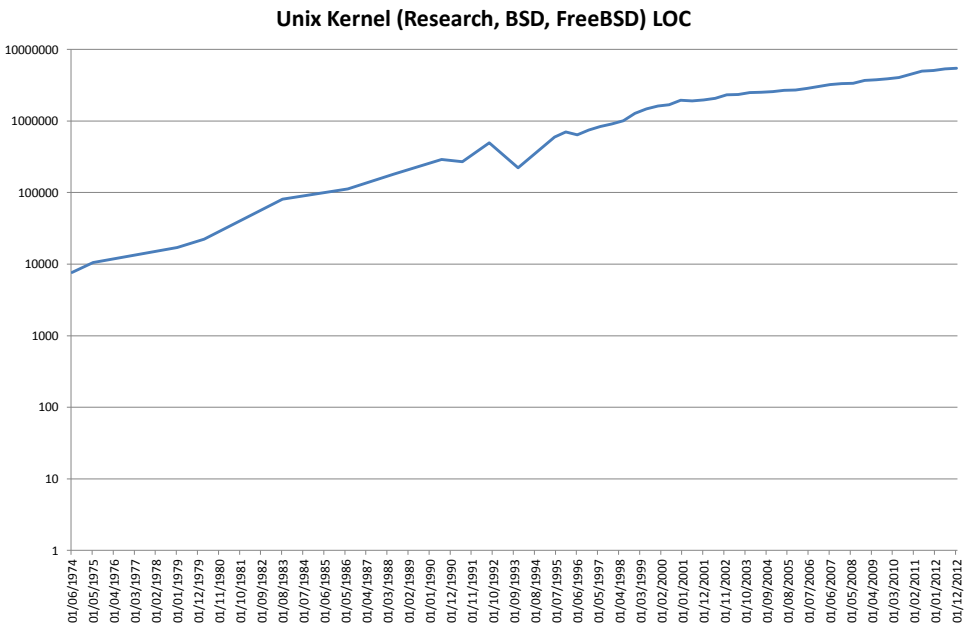
FreeBSD 8.0.0 (Nov 2009)

- Packet Capture Library
  - pcap(3)

FreeBSD 9.0.0 (Jan 2012)

- Infiniband / RDMA High-Speed Low-Latency Switched-Fabric Interconnect Library

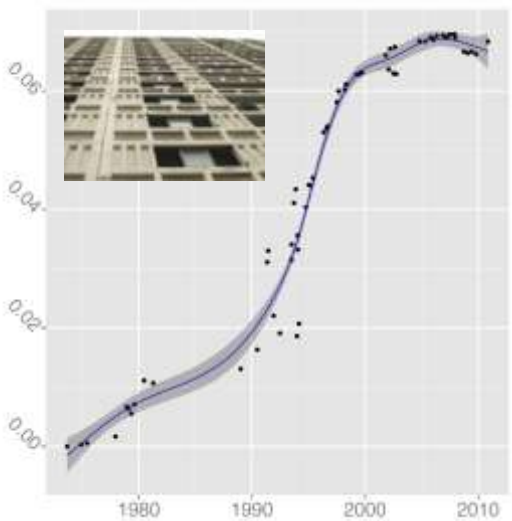

Observations
Numbers & Insights


Unix Kernel (Research, BSD, FreeBSD) LOC
10000000
1000000
100000
10000
1000
100
10
1
01/06/1974
01/05/1975
01/04/1976
01/03/1977
01/02/1978
01/01/1979
01/12/1979
01/11/1980
01/10/1981
01/09/1982
01/08/1983
01/07/1984
01/06/1985
01/05/1986
01/04/1987
01/03/1988
01/02/1989
01/01/1990
01/12/1990
01/11/1991
01/10/1992
01/09/1993
01/08/1994
01/07/1995
01/06/1996
01/05/1997
01/04/1998
01/03/1999
01/02/2000
01/01/2001
01/12/2001
01/11/2002
01/10/2003
01/09/2004
01/08/2005
01/07/2006
01/06/2007
01/05/2008
01/04/2009
01/03/2010
01/02/2011
01/01/2012
01/12/2012

Modularity increases with code size

Increase in number of **static** declarations / statement

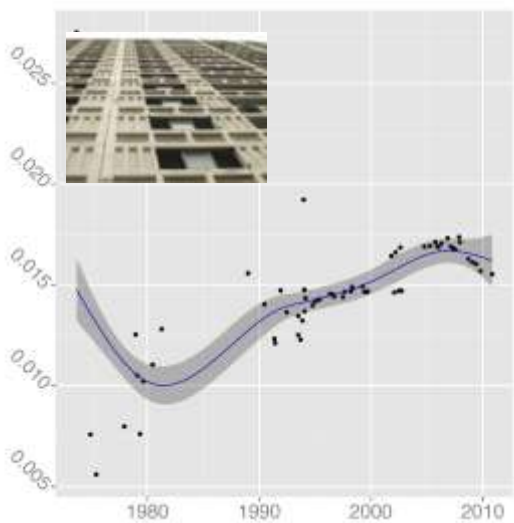
static short splice;



Modularity increases with code size

Increase in number of **#include** directives / line

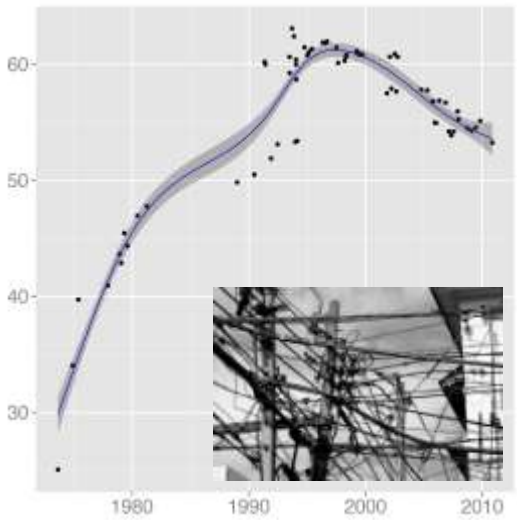
#include "if_uba.h"



Software complexity evolution follows self correction

Mean lines / function

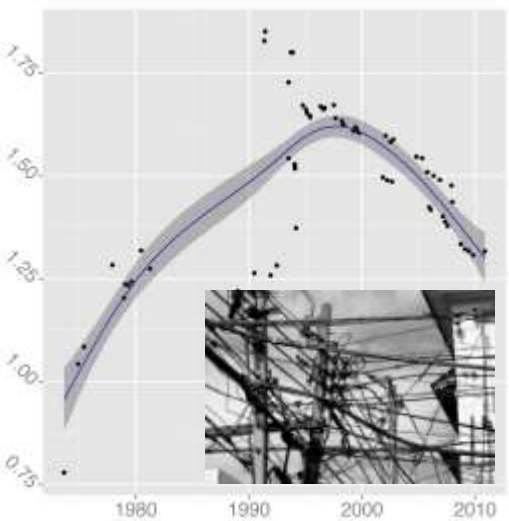
{
….
….
}



Software complexity evolution follows self correction

Mean statement nesting

```
if (a)
  while (b)
    for (;;)
      if (
```

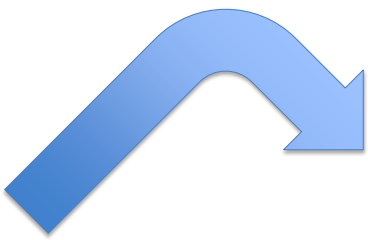


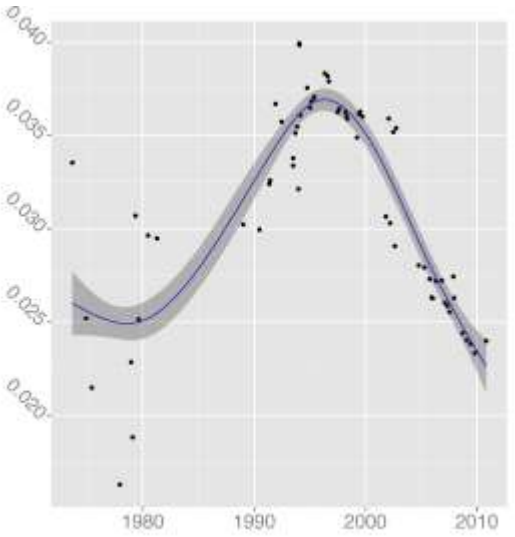
Software complexity evolution follows self correction

Density of C preprocessor non-include directives

#define

#if





Letters to the Editor

Go To Statement Considered Harmful

Key Words and Phrases: go to statement, jump instruction, branch instruction, conditional clause, alternative clause, repetitive clause, program intelligibility, program sequencing
CR Categories: 4.22, 5.23, 5.24

Editor:

For a number of years I have been familiar with the observation that the quality of programmers is a decreasing function of the density of go to statements in the programs they produce. More recently I discovered why the use of the go to statement has such disastrous effects, and I became convinced that the go to statement should be abolished from all "higher level" programming languages (i.e. everything except, perhaps, plain machine code). At that time I did not attach too much importance to this discovery; I now submit my considerations for publication because in very recent discussions in which the subject turned up, I have been asked to do so.

My first remark is that, although the programmer's activity ends when he has constructed a correct program, the process taking place under control of his program is the true subject matter of his activity, for it is this process that has to accomplish the desired effect; it is this process that in its dynamic behavior has to satisfy the desired specifications. Yet, once the program has been made, the "making" of the corresponding process is delegated to the machine.

My second remark is that our intellectual powers are rather dynamic progress is only characterized when we also give to which call of the procedure we refer. With the inclusion of procedures we can characterize the progress of the process via a sequence of textual indices, the length of this sequence being equal to the dynamic depth of procedure calling.

Let us now consider repetition clauses (like *while B repeat A* or *repeat A until B*). Logically speaking, such clauses are now superfluous, because we can express repetition with the aid of recursive procedures. For reasons of realism I don't wish to exclude them: on the one hand, repetition clauses can be implemented quite comfortably with present day finite equipment; on the other hand, the remaining patterns known as "indexical" seem to well equipped to retain our intellectual grasp on the processes generated by repetition clauses. With the inclusion of the repetition clauses textual indices are no longer sufficient to describe the dynamic progress of the process. With each entry into a repetition clause, however, we can associate a so-called "dynamic index," inexorably counting the ordinal number of the corresponding current repetition. As repetition clauses (just as procedure calls) may be applied tacitly, we find that now the progress of the process can always be uniquely characterized by a (mixed) sequence of textual and/or dynamic indices.

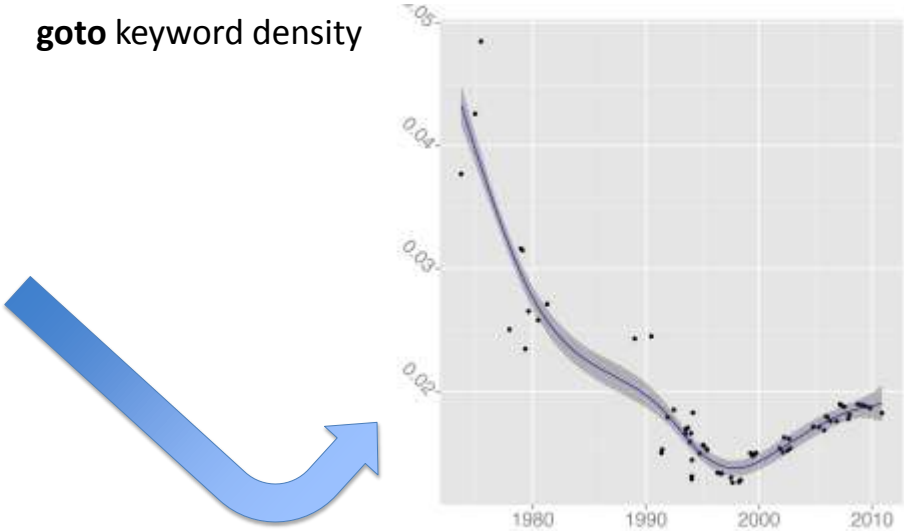
The main point is that the values of these indices are outside programmer's control; they are generated (albeit by the write-up of his program or by the dynamic evolution of the process) whether he wishes or not. They provide independent coordinates in which to describe the progress of the process.

EDSGER W. DIJKSTRA
Technological University
Eindhoven, The Netherlands

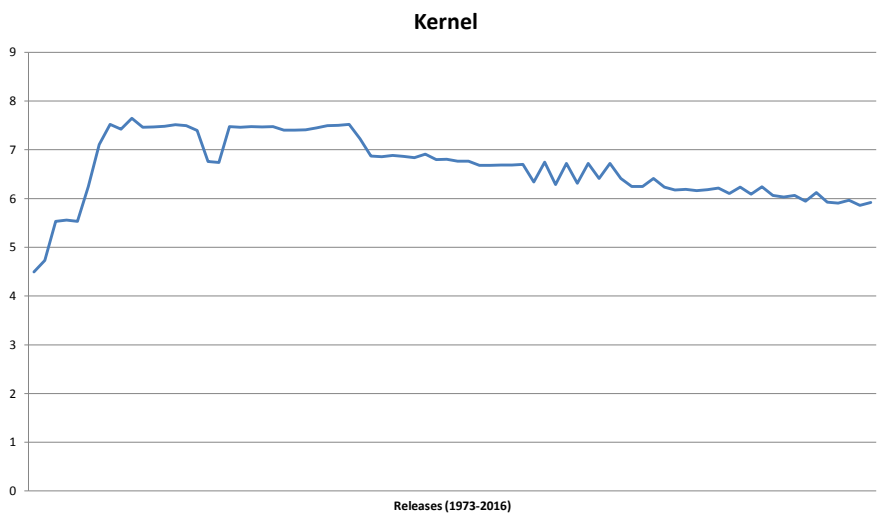
Volume 11 / Number 3 / March, 1968 Communications of the ACM 147

Software complexity evolution follows self correction

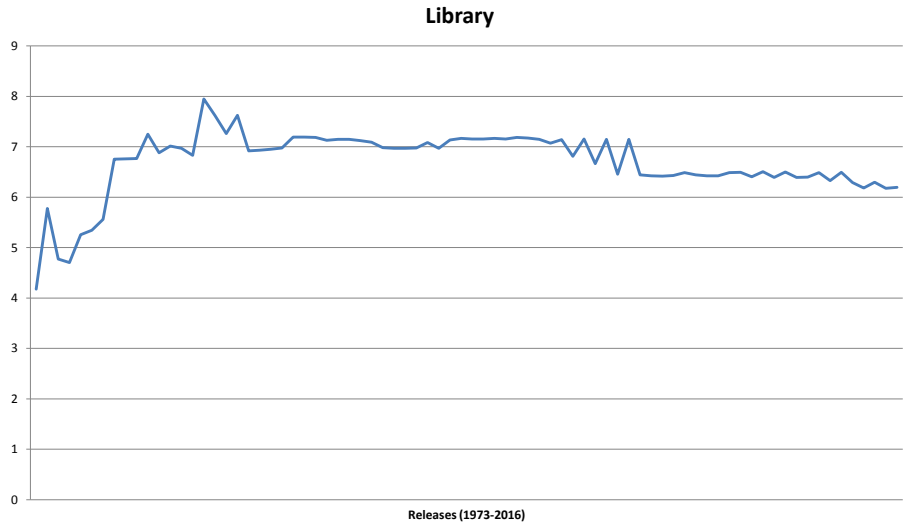
goto keyword density



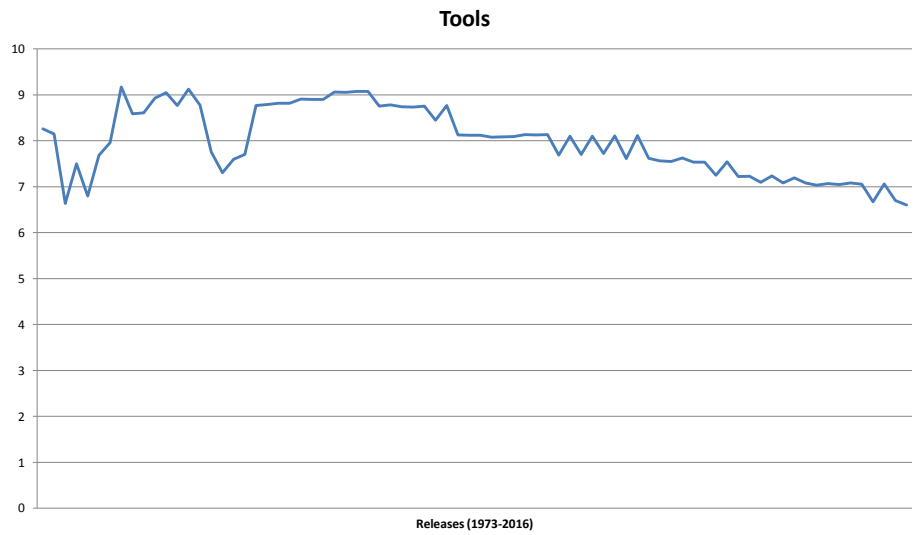
Cyclomatic Complexity



Cyclomatic Complexity



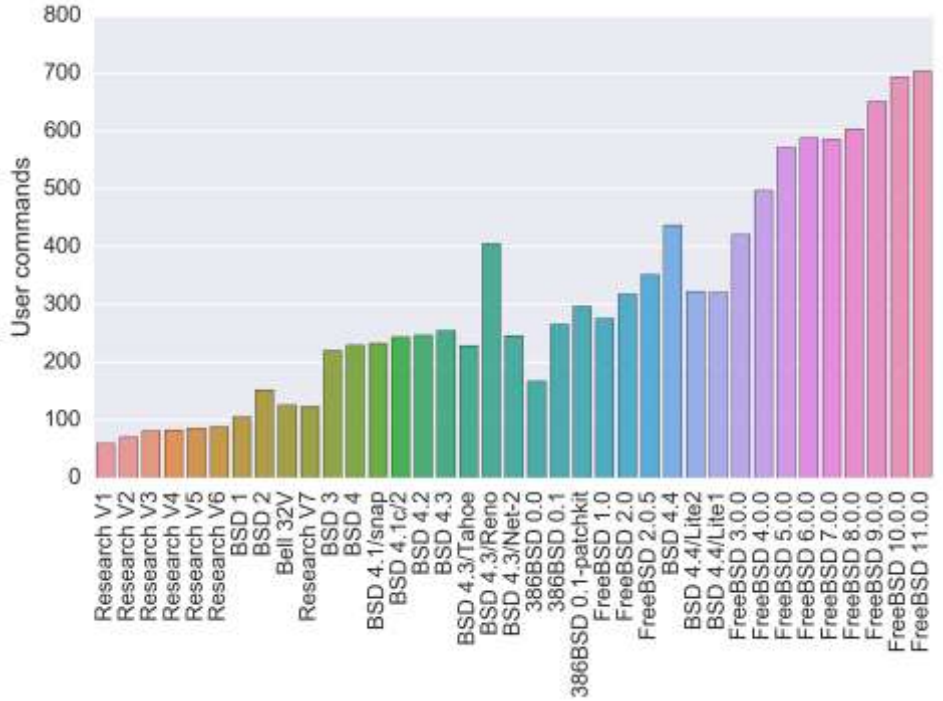
Cyclomatic Complexity

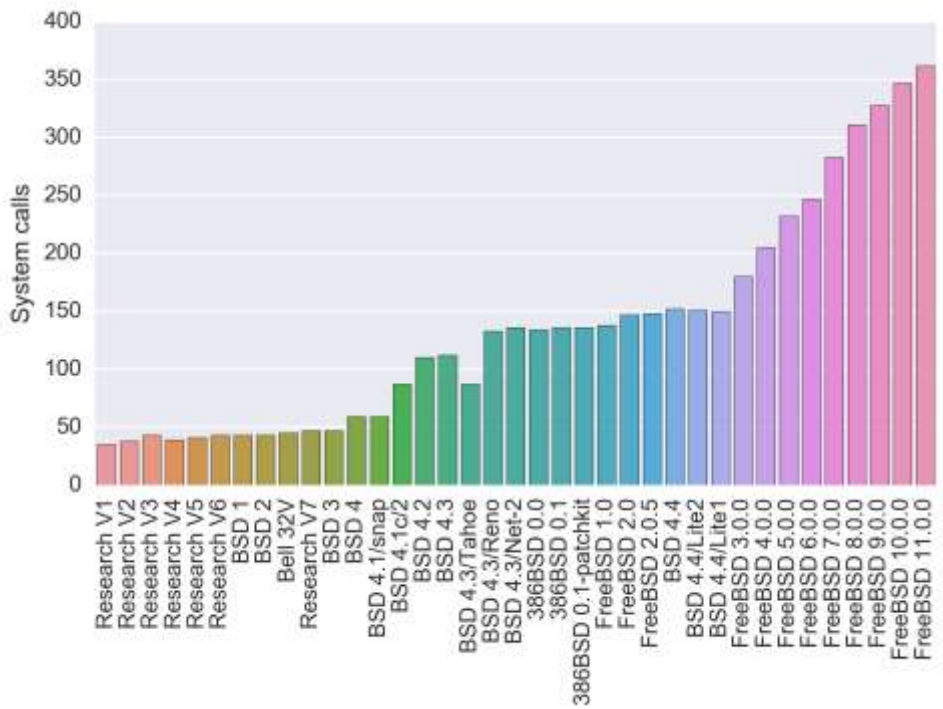


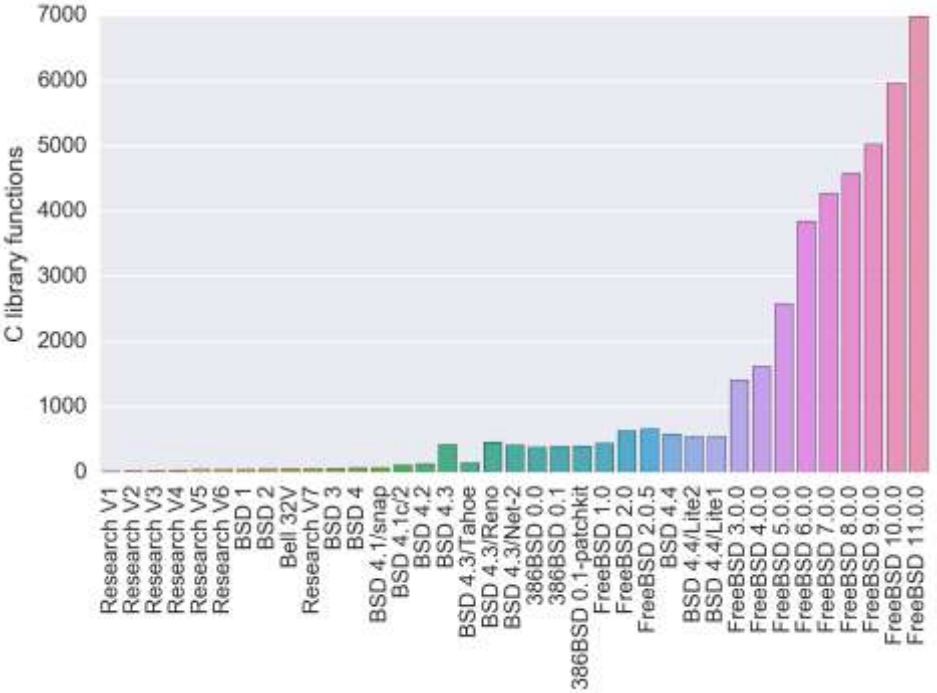
Agreement with Lehman's Laws

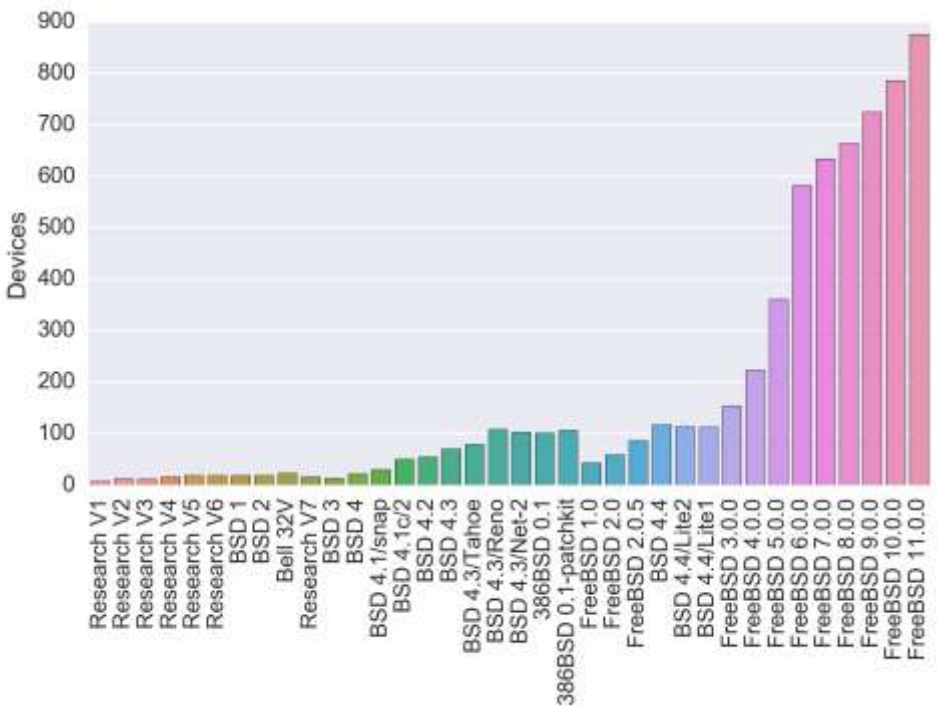
- Increasing Complexity
- Conservation of Familiarity
- Declining Quality
- Feedback System

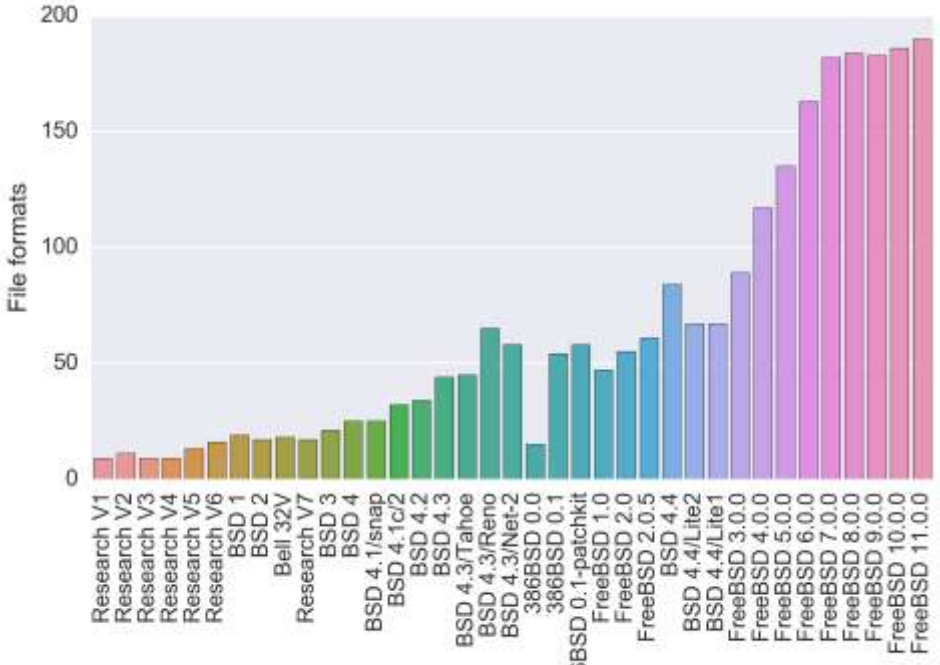
Growth in Facilities

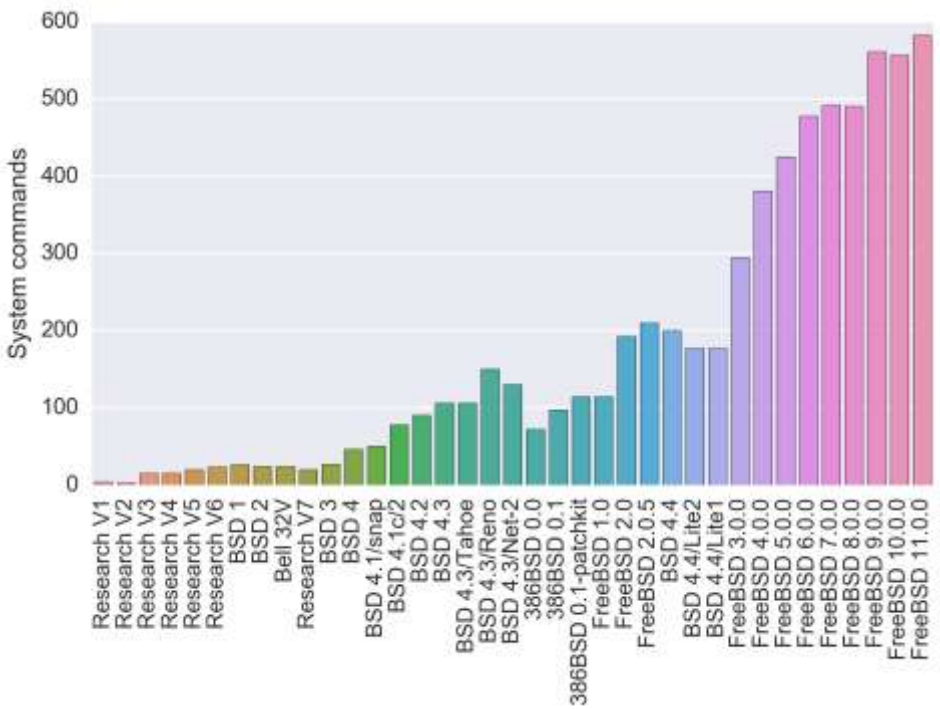

User commands
800
700
600
500
400
300
200
100
0
Research V1
Research V2
Research V3
Research V4
Research V5
Research V6
BSD 1
BSD 2
Bell 32V
Research V7
BSD 3
BSD 4
BSD 4.1/snap
BSD 4.1c/2
BSD 4.2
BSD 4.3
BSD 4.3/Tahoe
BSD 4.3/Reno
BSD 4.3/Net-2
386BSD 0.0
386BSD 0.1
386BSD 0.1-patchkit
FreeBSD 1.0
FreeBSD 2.0
FreeBSD 2.0.5
BSD 4.4
BSD 4.4/Lite2
BSD 4.4/Lite1
FreeBSD 3.0.0
FreeBSD 4.0.0
FreeBSD 5.0.0
FreeBSD 6.0.0
FreeBSD 7.0.0
FreeBSD 8.0.0
FreeBSD 9.0.0
FreeBSD 10.0.0
FreeBSD 11.0.0

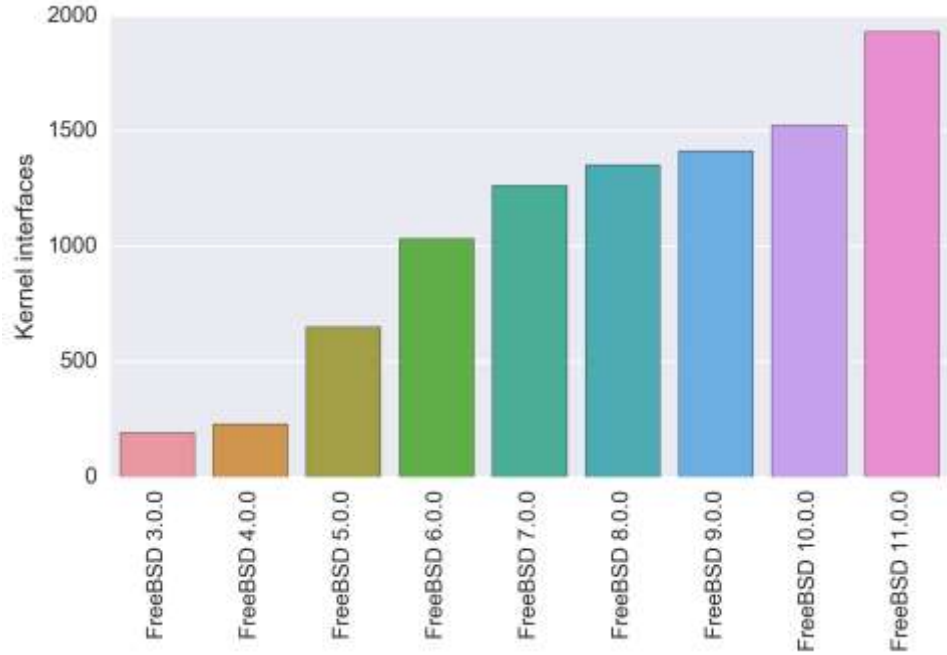

System calls
400
350
300
250
200
150
100
50
0
Research V1
Research V2
Research V3
Research V4
Research V5
Research V6
BSD 1
BSD 2
Bell 32V
Research V7
BSD 3
BSD 4
BSD 4.1/snap
BSD 4.1c/2
BSD 4.2
BSD 4.3
BSD 4.3/Tahoe
BSD 4.3/Reno
BSD 4.3/Net-2
386BSD 0.0
386BSD 0.1
386BSD 0.1-patchkit
FreeBSD 1.0
FreeBSD 2.0
FreeBSD 2.0.5
BSD 4.4
BSD 4.4/Lite2
BSD 4.4/Lite1
FreeBSD 3.0.0
FreeBSD 4.0.0
FreeBSD 5.0.0
FreeBSD 6.0.0
FreeBSD 7.0.0
FreeBSD 8.0.0
FreeBSD 9.0.0
FreeBSD 10.0.0
FreeBSD 11.0.0


C library functions
7000
6000
5000
4000
3000
2000
1000
0
Research V1
Research V2
Research V3
Research V4
Research V5
Research V6
BSD 1
BSD 2
Bell 32V
Research V7
BSD 3
BSD 4
BSD 4.1/snap
BSD 4.1c/2
BSD 4.2
BSD 4.3
BSD 4.3/Tahoe
BSD 4.3/Reno
BSD 4.3/Net-2
386BSD 0.0
386BSD 0.1
386BSD 0.1-patchkit
FreeBSD 1.0
FreeBSD 2.0
FreeBSD 2.0.5
BSD 4.4
BSD 4.4/Lite2
BSD 4.4/Lite1
FreeBSD 3.0.0
FreeBSD 4.0.0
FreeBSD 5.0.0
FreeBSD 6.0.0
FreeBSD 7.0.0
FreeBSD 8.0.0
FreeBSD 9.0.0
FreeBSD 10.0.0
FreeBSD 11.0.0


Devices
900
800
700
600
500
400
300
200
100
0
Research V1
Research V2
Research V3
Research V4
Research V5
Research V6
BSD 1
BSD 2
Bell 32V
Research V7
BSD 3
BSD 4
BSD 4.1/snap
BSD 4.1c/2
BSD 4.2
BSD 4.3
BSD 4.3/Tahoe
BSD 4.3/Reno
BSD 4.3/Net-2
386BSD 0.1
386BSD 0.1-patchkit
FreeBSD 1.0
FreeBSD 2.0
FreeBSD 2.0.5
BSD 4.4
BSD 4.4/Lite2
BSD 4.4/Lite1
FreeBSD 3.0.0
FreeBSD 4.0.0
FreeBSD 5.0.0
FreeBSD 6.0.0
FreeBSD 7.0.0
FreeBSD 8.0.0
FreeBSD 9.0.0
FreeBSD 10.0.0
FreeBSD 11.0.0

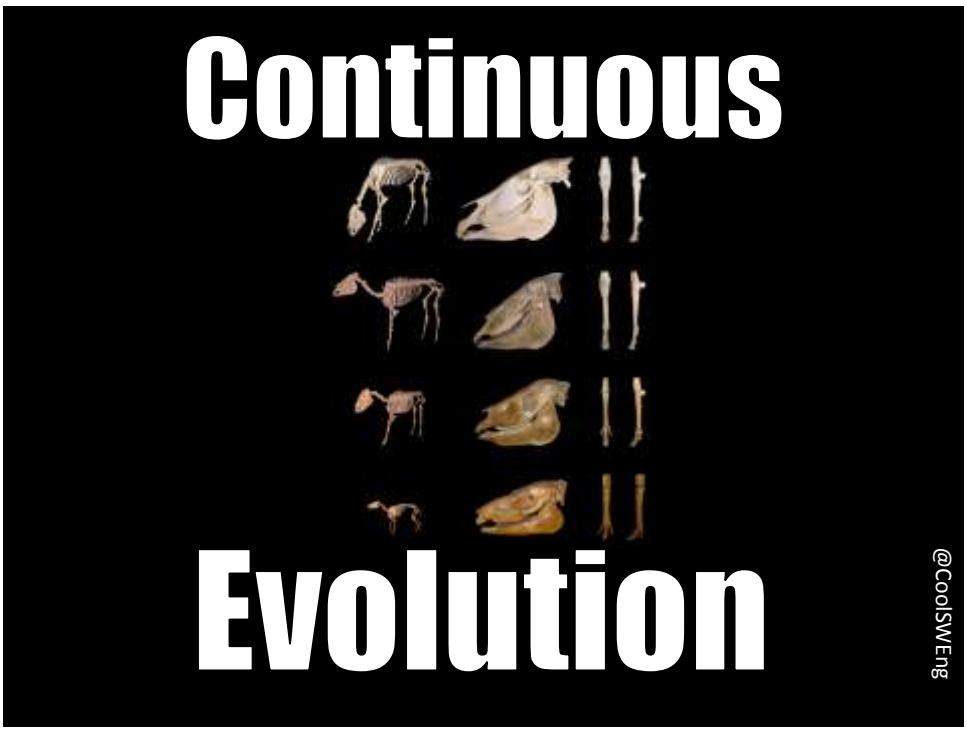

File formats
0
50
100
150
200
Research V1
Research V2
Research V3
Research V4
Research V5
Research V6
BSD 1
BSD 2
Bell 32V
Research V7
BSD 3
BSD 4
BSD 4.1/snap
BSD 4.1c/2
BSD 4.2
BSD 4.3
BSD 4.3/Tahoe
BSD 4.3/Reno
BSD 4.3/Net-2
386BSD 0.0
386BSD 0.1
386BSD 0.1-patchkit
FreeBSD 1.0
FreeBSD 2.0
FreeBSD 2.0.5
BSD 4.4
BSD 4.4/Lite2
BSD 4.4/Lite1
FreeBSD 3.0.0
FreeBSD 4.0.0
FreeBSD 5.0.0
FreeBSD 6.0.0
FreeBSD 7.0.0
FreeBSD 8.0.0
FreeBSD 9.0.0
FreeBSD 10.0.0
FreeBSD 11.0.0


System commands
0
100
200
300
400
500
600
Research V1
Research V2
Research V3
Research V4
Research V5
Research V6
BSD 1
BSD 2
Bell 32V
Research V7
BSD 3
BSD 4
BSD 4.1/snap
BSD 4.1c/2
BSD 4.2
BSD 4.3
BSD 4.3/Tahoe
BSD 4.3/Reno
BSD 4.3/Net-2
386BSD 0.0
386BSD 0.1
386BSD 0.1-patchkit
FreeBSD 1.0
FreeBSD 2.0
FreeBSD 2.0.5
BSD 4.4
BSD 4.4/Lite2
BSD 4.4/Lite1
FreeBSD 3.0.0
FreeBSD 4.0.0
FreeBSD 5.0.0
FreeBSD 6.0.0
FreeBSD 7.0.0
FreeBSD 8.0.0
FreeBSD 9.0.0
FreeBSD 10.0.0
FreeBSD 11.0.0

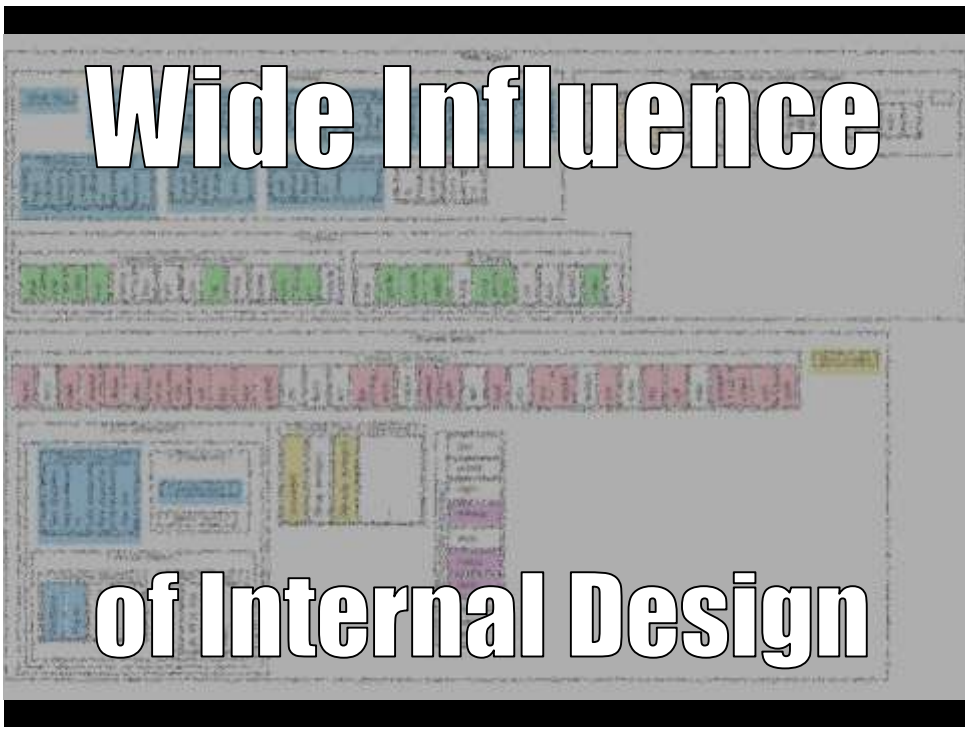

2000
1500
1000
500
0
Kernel interfaces
FreeBSD 3.0.0
FreeBSD 4.0.0
FreeBSD 5.0.0
FreeBSD 6.0.0
FreeBSD 7.0.0
FreeBSD 8.0.0
FreeBSD 9.0.0
FreeBSD 10.0.0
FreeBSD 11.0.0


Observations
Insights

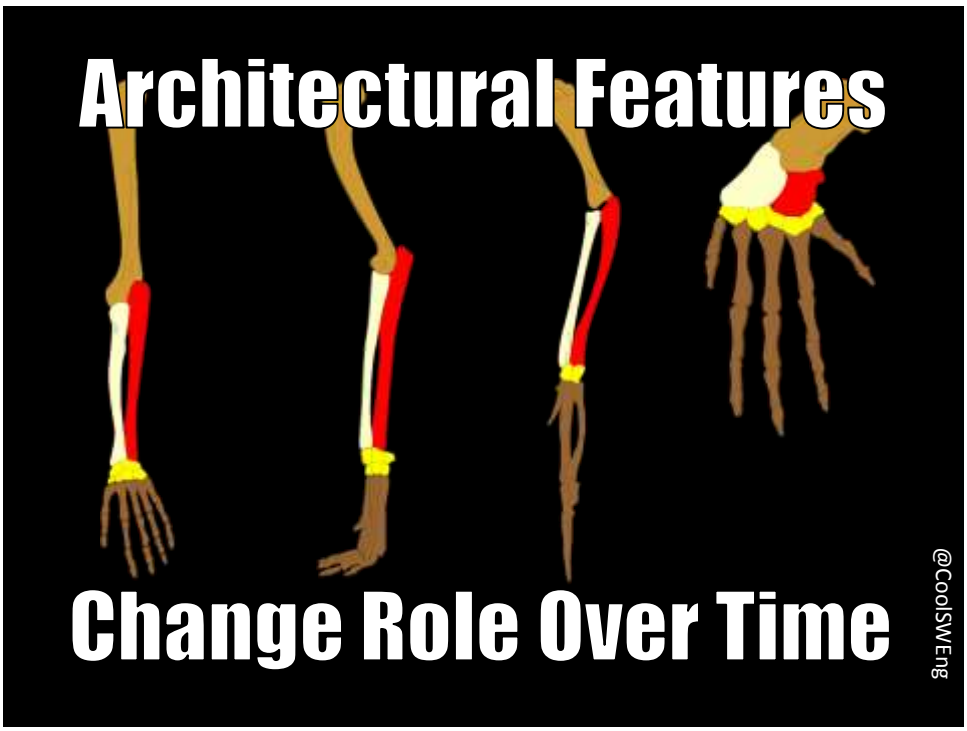

Conventions
Over Rigid Enforcement


Continuous
Evolution
@CoolSWEng

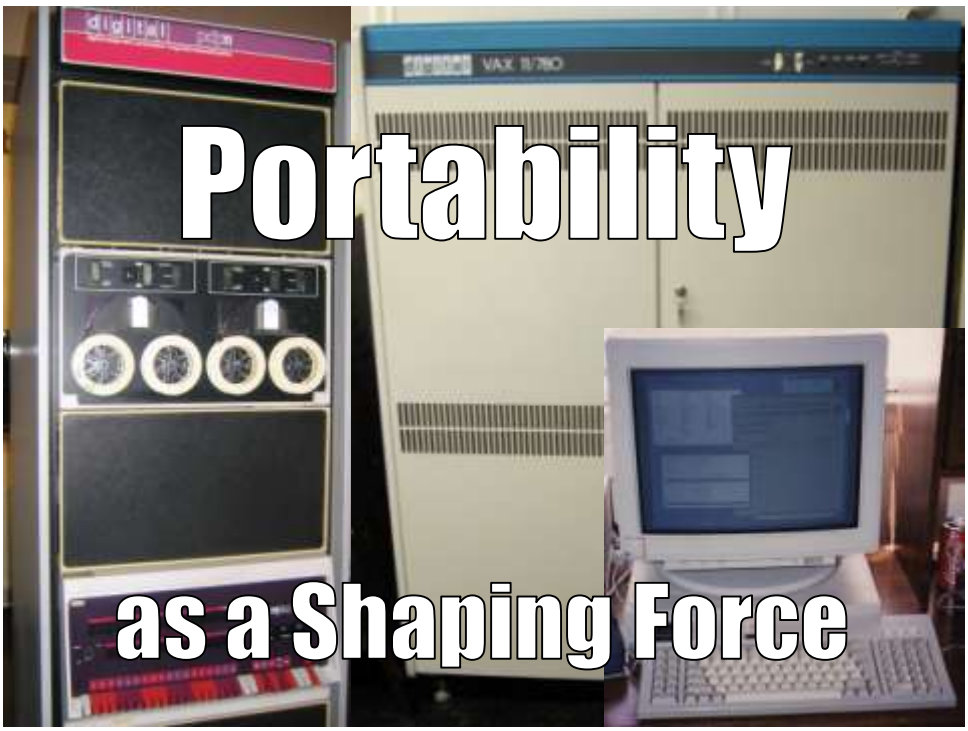

Slowdown
of Architectural Evolution


Wide Influence
of Internal Design

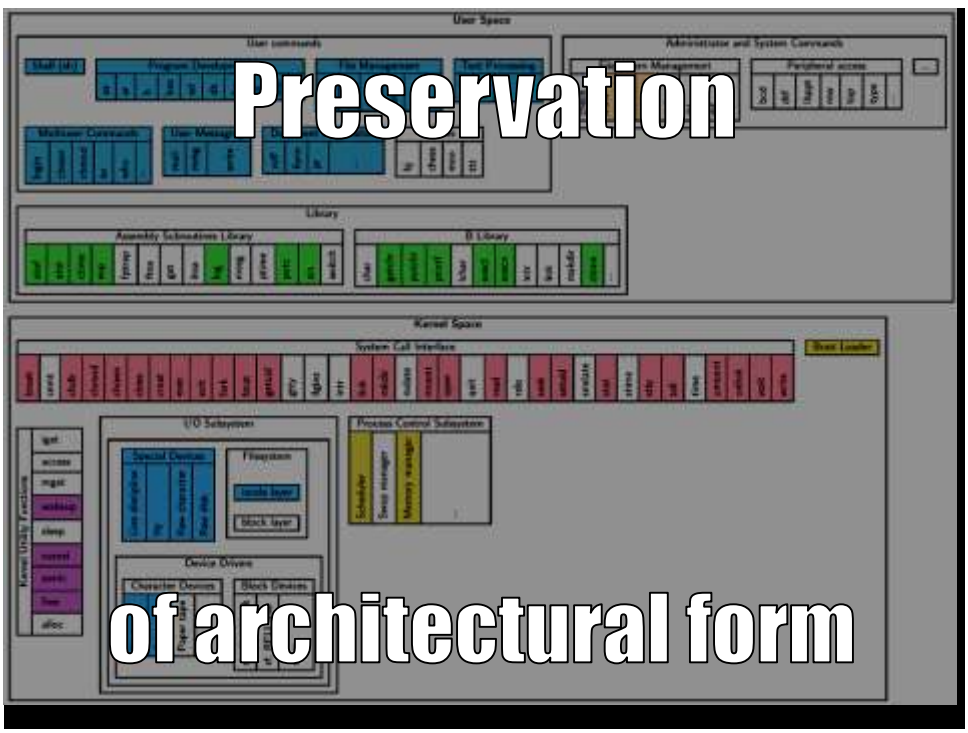

Features can be Added
and Not Be Used

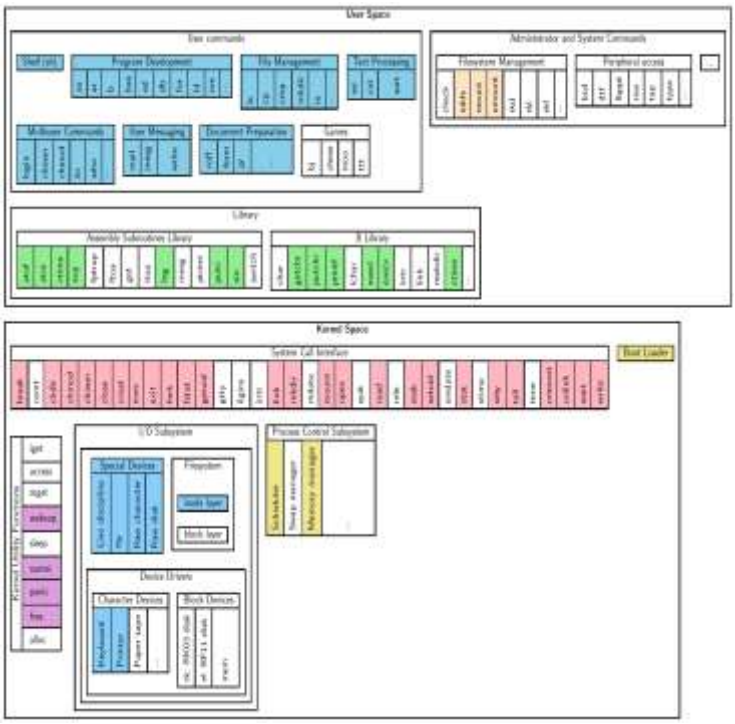

Architectural Features
Change Role Over Time
@CoolSWEng

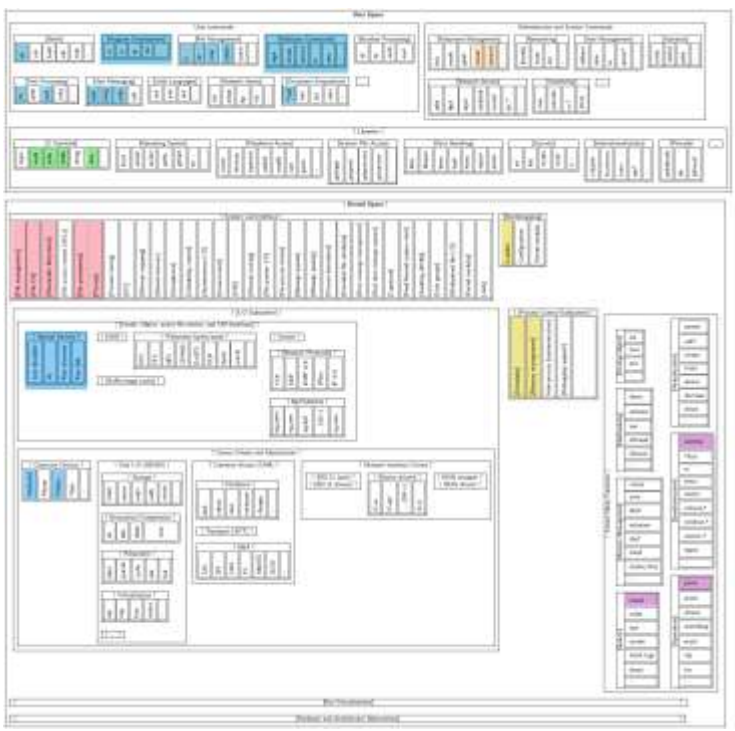

COUPON BOND
OF AMERICA
TEN THOUSAND DOLLARS
10⅛%
TREASURY BOND
OF 1994
Inadvertent
Technical Debt
@CoolSWEng

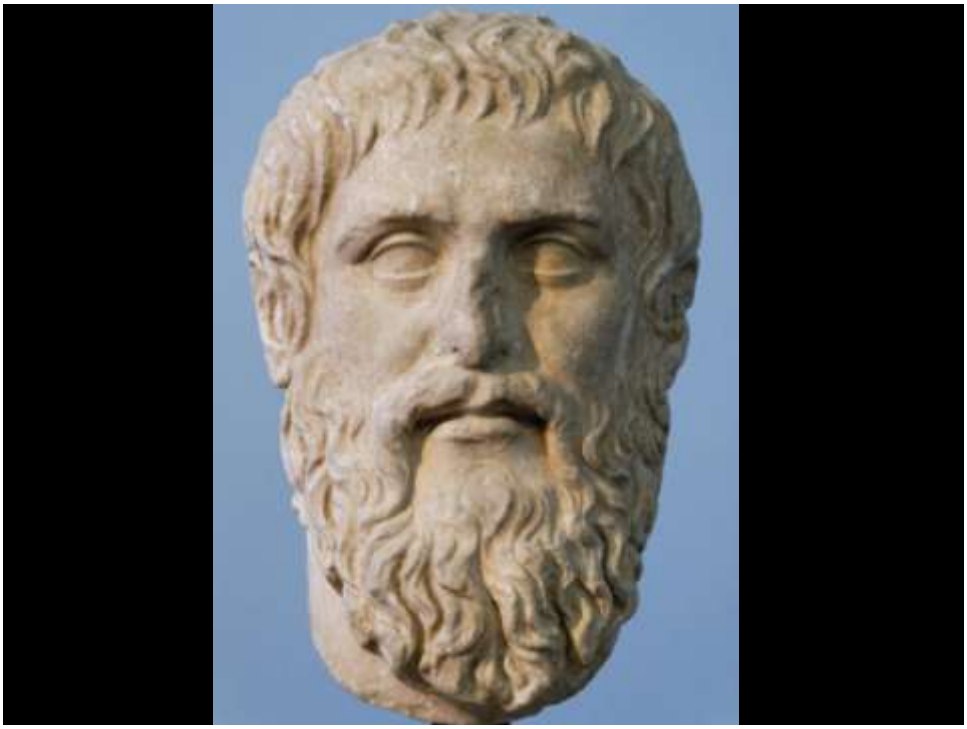

digital
VAX 11/780
Portability
as a Shaping Force

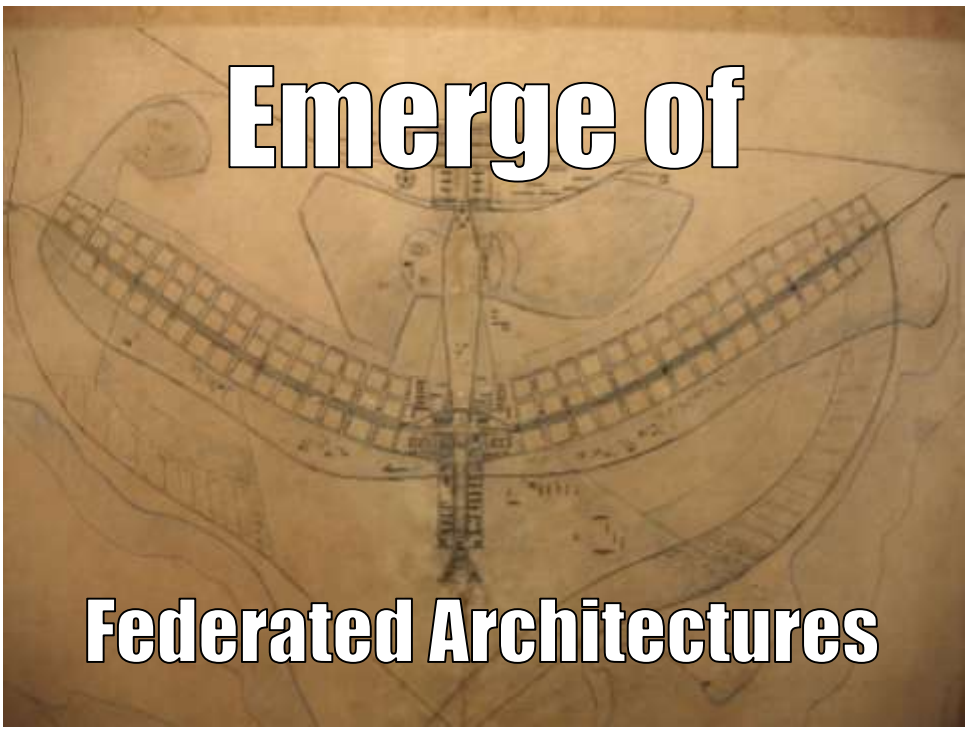

Competition Between
Alternative Architectural Styles


Preservation
of architectural form


User Space
User commands
Shell (sh)
Program Development
File Management
Text Processing
Multiuser Commands
User Messaging
Document Preparation
Games
Administrator and System Commands
Filesystem Management
Peripheral access
Library
Assembly Subroutines Library
C Library
Kernel Space
System Call Interface
Boot Loader
I/O Subsystem
Special Devices
Filesystem
Block layer
Device Drivers
Character Devices
Block Devices
Process Control Subsystem
Scheduler
Kernel Utility Processes






Emerge of
Federated Architectures


Importance
of a Gifted Team


Go and architect
something great!

Thank you!

github.com/dspinellis/unix-history-repo

www.spinellis.gr

@CoolSWEng

Funding Credit

The research described has been partially carried out as part of the CROSSMINER Project, which has received funding from the European Union's Horizon 2020 Research and Innovation Programme under grant agreement No. 732223.

Image Credits

- Decades: Wikipedia (1970,80,90,2000)
- ASR-33 Teletype: Rama & Musée Bolo
- VT100: Jason Scott
- VAX 11/780: Joe Mabel
- PDP 11/20: Image courtesy of Computer History Museum
- VAX in use: Photo courtesy of Berkeley Lab © 2010 The Regents of the University of California, through the Lawrence Berkeley National Laboratory.
- Pentium: Iorsh
- Hypotheses: Niklas Morberg
- Modules: Suatu Ketika
- Reading glasses: Walt Stoneburner
- Cables: christof tof
- Chemical flasks: Joe Sullivan
- Snake Oil cover: Clark Stanley
- Haswell Chip: Intel Free Press
- Sparcstation 10: Thomas Kaiser
- Gold coins: Anonimski
- Go to statement considered harmful: Edsger W. Dijkstra and ACM
- Manny Lehman: © Copyright 2009 Imperial College London
- Saladin and Guy de Lusignan after battle of Hattin in 1187: Said Tahsine

(Creative commons licenses)

- PDP11/40: Stefan_Kögl, CC BY-SA 3.0
- Digital VAX 11/780: Emiliano Russo, PD
- SPARCstation, Fourdee, PD
- Hologous bones: Волков Владислав Петрович, CC BY-SA 4.0
- Skeletal evolution: H. Zell, CC BY-SA 3.0
- Company Shocked at a Lady Getting up to Ring the Bell: James Gillray, PD
- SLOW: Monyo Kararan, CC BY-SA-ND 2.0
- Bond: JHerbstman, PD
- SSL mixer console: Rebecca Wilson CC-BY-2.0
- Ancient Indian observatory at Delhi: Thomas Daniell, PD
- Roman milestone: Júlio Reis , CC BY-SA 3.0
- Florence Cathedral: Bruce Stokes, CC BY-SA 2.0