

CWRU LINUX CLUB OCT 24
Prof. Ronald Loui rxl637

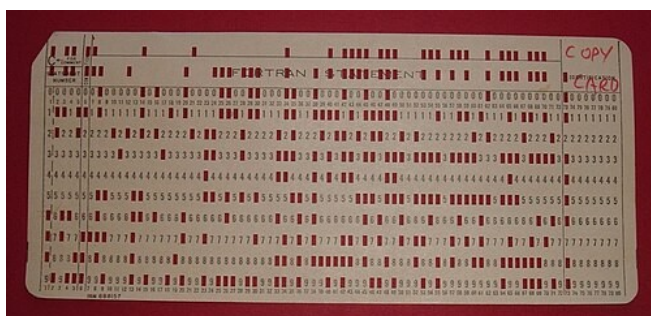
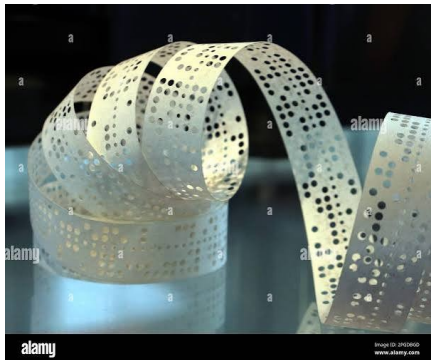
UNIX BEEN BERRY GOOD TO ME

Started on mini HP2100 in BASIC with teletype

Dartmouth BASIC OS: the language WAS THE OS:
maybe had commands to RUN, EDIT, LOAD@ (no
save – save where?)

Not great for doing real things
But excellent way to learn ALL!

Though I did write a program that could store 12 insults and randomly send them back to you.

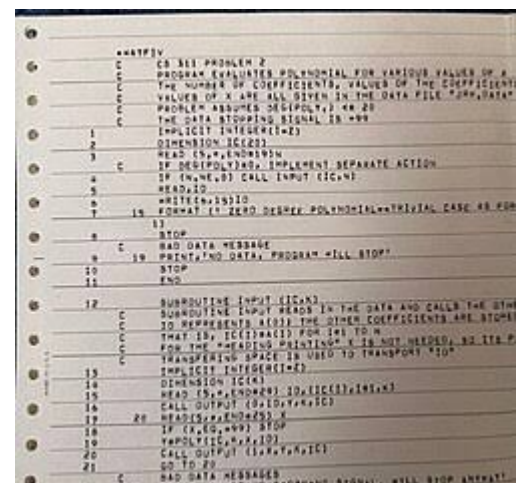


**Powerful
Computing Genie:
\$4900**

READY, WILLING AND ABLE.

[illegible]

- personal computer



ADDS/MENTOR/PICK/ENGLISH DB/PROC/BASIC/PR1ME

and

VAX 11/780 VMS WITH FORTRAN/EDT

and

DEC HUDSON SUMMER INTERNSHIP WITH VAXes of all kinds, C

Fantastic for getting real work done!

Faster in fact, for what it was designed for. Hmm...

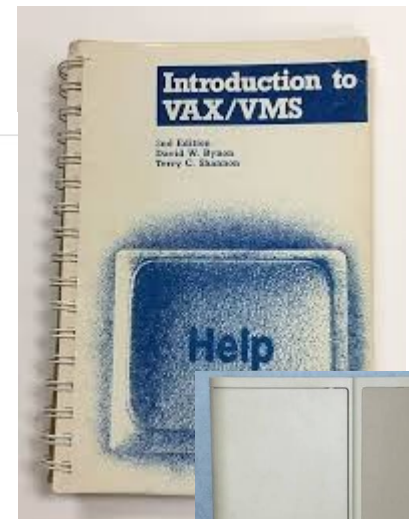


Performs constant or variable assignment at compile time. A *symbol* is similar to a variable name, with the exception that it may not be modified by placing it to the left of the equal sign in an assignment expression. Here are some examples:

```
EQU CLEAR.SCREEN TO CHAR(12)
EQU RING.BELL TO CHAR(7)
EQU ATTRIBUTE.MARK TO CHAR(254)
EQU CUSTOMER.NAME.ATTRIBUTE TO 1
EQU INVOICE.NUMBER TO CUSTOMER.ITEM(13)
EQU TITLE TO "PICK POCKET GUIDE"
```

EXECUTE TCL.expression {RETURNING return.variable} . . . {CAPTURING capture.variable}

Issues any valid TCL expression, and continues execution of the program at the next line upon completion. The TCL expression may be any valid TCL command, including ACCESS sentences, verbs, PROCs, or other cataloged PICK/BASIC programs. Any input that may be needed by the EXECUTE statement may be provided with the DATA statement.



➡ ACCESS Verbs



The third, and final class of verbs are ACCESS verbs. These are, by far, the most sophisticated of all verbs and require at minimum a verb and a file name. See the ACCESS section for further information. To determine if a verb belongs in this category, issue the LISTVERBS command. If the second column (attribute) contains a "35", it's an ACCESS verb. Here are some examples of ACCESS commands:

```
>SORT CUSTOMERS BY NAME NAME ADDRESS CSZ LPTR
>SSELECT INVOICES WITH NO PAYMENT.AMOUNT BY DATE
>SSELECT ORDERS BY SHIP.DATE
```

THEN THE PHD PROGRAM!

Tops20 on DEC 20

Xerox Alto and D-machines

and someone introduced me to UNIX (actually Jerry Dunietz taught me vi at H but it was not easy to access UNIX in college)

At first:

This is whack:



This is stupid. Where is the database?

Stupid, the file is the database. -- Leo Hartman

How do you access the records and fields, stupid?

Stupid, you use grep. -- Leo Hartman

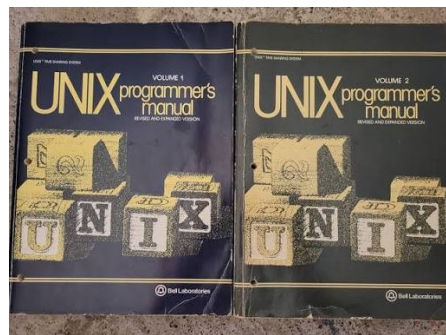
That's not formatting the results for me, stupid!

Stupid, you use awk. -- Leo Hartman (who also taught me squash)

This took years and years, but I NEVER LOOKED BACK

Never tempted by Windows products, Apple products, NeXt, OS2. Lisp Machines, etc

Though I put W98 on a lot of machines in the 90s (mea culpa) but often to talk to Unix servers



SO WHAT DID IT GIVE ME?

Independence

Authority / Cred

Cost Savings

Speed

Control

Reliability (*not perfect, but relatively speaking)

Freedom from fad

Brotherhood with the Best

Reminder of Real Computing Brilliance (sorry, but ... not sorry; Usenix Lifetime > Turing Award)

Skills (even longevity, and teaching longevity)

Apps Galore! (once I discovered gawk, I started writing 100 scripts a year for 30yrs)

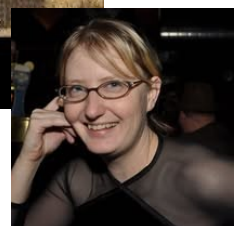
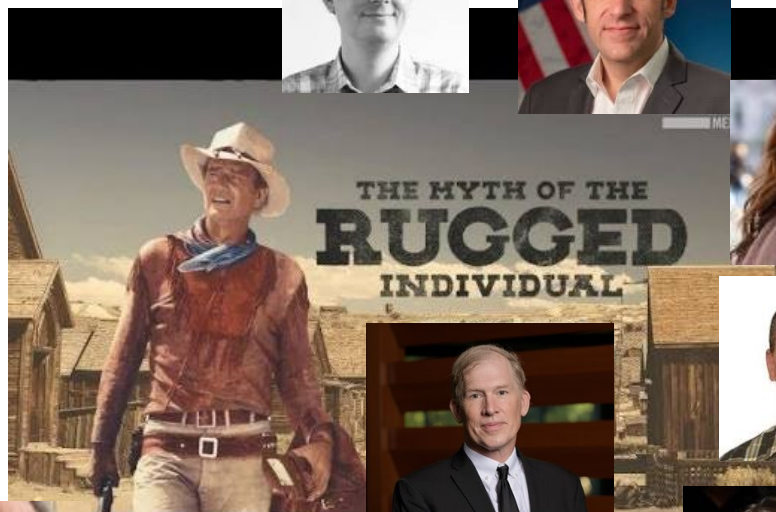
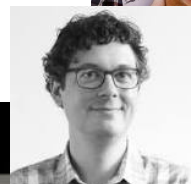
Web servers – prototypes – ideas!

Computing, Generally

Lots of cheap machines (currently running 16++ linux)

Joy

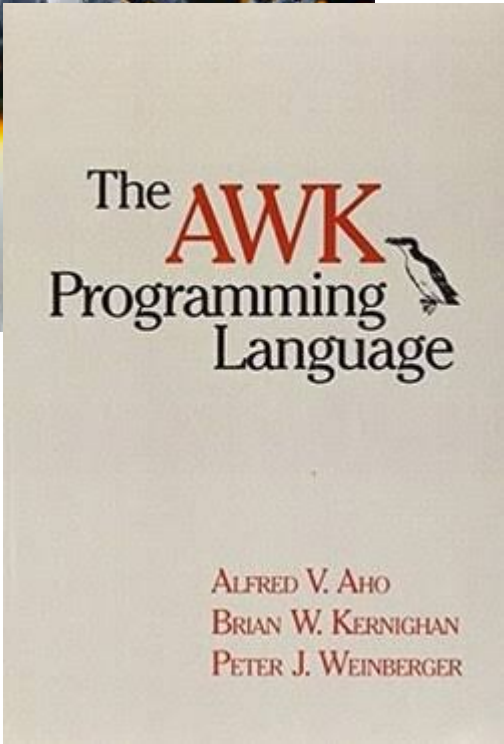
Not Typesetting, sadly





Unix filters

Simple	Intermediate	Complex
tail tr wc sort and more...	grep	sed awk custom python programs etc...



ODD THINGS I DO NOW IN LINUX

I boot from live usb

store to servers, 2nd usb or ext drive, do not configure

I learned long ago that config files don't follow you around
so practice remembering what you need to know

install software on demand

I monitor memory usage in top a lot

zram lockup with no swap can surprise you

I use 2-3-4gb ram machines

prefer lubuntu18

but browser/sites require bloated 24+ (boo!)

I worry about security much more than command line errors

I work out of /tmp and use absolute path names with tab-completion

I use vim/insert as my screen saver

I try to stay in plain text as much as possible

I use google and LLMs much more than man pages

I rarely vibe code because practice/discipline/ownership/process

I can still rewrite in awk faster than I can find where I put a program >1yr old or tell LLM what 2do
dev process evolves what you want to do (UX CLASS)

Note my keyboard effort tool: <http://codereserve.us/cgi/keyboardeffort.cgi>

AWK 581 APL 722 LOGO 1152 BASH 1168 BASH 1174 RUBY 1186 TCL 1329 FORTH 1366 LLM-AWK
1416 DBASE 1421 ICON 1495 SMALLTALK 1542 HASKELL 1545 GROOVY 1561 PYTHON 1572
POWERSHELL 1599 LUA 1620 SAS 1627 KOTLIN 1629 SCALA 1636 CLOJURE 1693 MATLAB 1749 PHP 1750
R 1765 PICK.BASIC 1806 SCHEME 1840 XSLT 1860 COMMON.LISP 1887 SNOBOL 1890 OCAML 1917 D 2006
ML 2017 PERL 2041 ALGOL 2065 NODE.JS 2074 TYPESCRIPT 2086 CSHARP 2124 VB.NET 2200 JAVA 2202
RPG 2226 SWIPROLOG 2277 SWIFT 2447 OCCAM 2470 MYSQL 2473 EXCEL.VBA.MACRO 2485 RUST 2519
APPLESCRIPT 2617 ERLANG 2730 C 2796 PL/I 2823 JULIA 2902 GOLANG 3117 COBOL 3151 EIFFEL 3163 C+
+ 3193 OBERON 3231 PASCAL 3698 OBJECTIVEC 3708 ADA 3714 MODULA 4008 FORTRAN 4499

Short identifiers are your friend when doing real (fast) work (unshared)

Python too heavy pour moi, R ok if you do it a lot, Bash unsuited but ok if you do it a lot

Maybe Lua? Stanford's Ousterhout still likes TCL.

(SCRIPTING CLASS)

The trick is to change mentality to be mercenary in scripting pragmatics

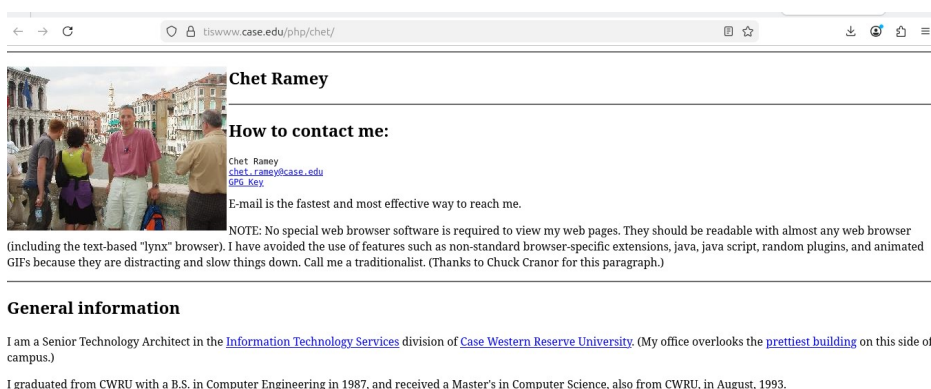
Make peace with short regexps

I use a lot of ctrl-shift-v

I hate vim (students suggest alternatives, ..., but need it for its undo stack)

I am still not a fan of bash (an old csh/tcsh guy, call out to awk!)

so sorry Chet Ramey (who got the albatross from Brian Fox)



Chet Ramey

How to contact me:

Chet Ramey
chet_ramey@case.edu
[GPG Key](#)

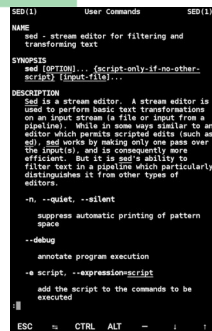
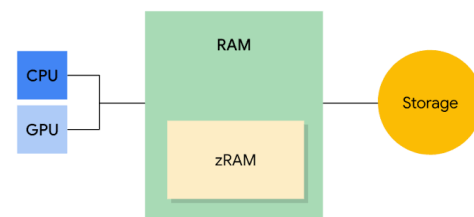
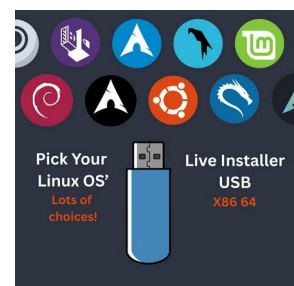
E-mail is the fastest and most effective way to reach me.

NOTE: No special web browser software is required to view my web pages. They should be readable with almost any web browser (including the text-based "lynx" browser). I have avoided the use of features such as non-standard browser-specific extensions, java, java script, random plugins, and animated GIFs because they are distracting and slow things down. Call me a traditionalist. (Thanks to Chuck Cranor for this paragraph.)

General information

I am a Senior Technology Architect in the [Information Technology Services](#) division of [Case Western Reserve University](#). (My office overlooks the [prettiest building](#) on this side of campus.)

I graduated from CWRU with a B.S. in Computer Engineering in 1987, and received a Master's in Computer Science, also from CWRU, in August, 1993.



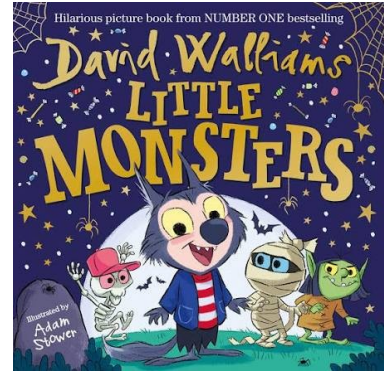
WHAT I SHOW YOU TODAY: FUN LITTLE MONITORS IN AWK

awk -f topmon.awk (ctrl-C or ctrl-Z)

topmon.awk

```
BEGIN {
    max = 20
    jscale = 5
    iscale = 10
    for (e=1; e<=1000; e++) doit()
}
function doit() {
    delete sto
    com = "top -b -n 1"
    while (com | getline) {
        pid = $1; if (pid !~ /^ *[0-9]+$/ ) continue
        cpu = $9; mem = $10; i = cpu; j = mem

        i = int(i*iscale+.5); j = int(j*jscale+.5)
        if (i > max) i = max; if (j > max) j = max
        sto[i,j]++
    }; close(com)
    system("clear")
    print com, e
    print "%cpu V" %mem >"
    printf "|"; for (i=0; i<=max; i++) printf "--"; print "|"
    for (j=max; j>=0; j--) {
        printf "|"
        for (i=0; i<=max; i++) {
            count = sto[i,j]-0*hold[i,j]
            if (count > 9) count = "*"
            else if (count == 0) count = " "
            else if (count < 0) count = "-"
            printf count " "
        }
        print "|"
    }
    printf "|"; for (i=0; i<=max; i++) printf "__"; print "|"
    for (x in sto) hold[x] = sto[x]
    # system("sleep 1")
}
```



lubuntu@lubuntu: /media/lubuntu/ab8bbe50-585b-44df-b4d7-f8845773

top -b -n 1 652

%cpu V %mem >

```
-----|
1      | 1      |
1      | 1      |
1      |         |
1      |         |
1      |         |
1      |         |
2      |         |
2      |         |
3      |         |
2      |         |
6      |         |
5      |         |
*      |         |
*      |         |
-----|
```

File Actions Edit View Help

lubuntu@lubuntu: /media/lubuntu/ab8bbe50-585b-44df-b4d7-f8845773259/home/guest/ux1

```
find: '/rofs/var/log/private': Permission denied
find: '/rofs/var/log/speech-dispatcher': Permission denied
find: '/rofs/var/spool/cron/crontabs': Permission denied
find: '/rofs/var/spool/rsyslog': Permission denied
lubuntu@lubuntu: /media/lubuntu/ab8bbe50-585b-44df-b4d7-f88457
me/guest/ux1$ find -mtime 1 / > /dev/null
find: paths must precede expression: '/'
find: possible unquoted pattern after predicate '-mtime'?
lubuntu@lubuntu: /media/lubuntu/ab8bbe50-585b-44df-b4d7-f88457
me/guest/ux1$ find / -mtime 1 > /dev/null 2>/dev/null
lubuntu@lubuntu: /media/lubuntu/ab8bbe50-585b-44df-b4d7-f88457
me/guest/ux1$ find / -mtime 1 > /dev/null 2>/dev/null &
[7] 1490188
lubuntu@lubuntu: /media/lubuntu/ab8bbe50-585b-44df-b4d7-f88457
me/guest/ux1$ find / -mtime 1 > /dev/null 2>/dev/null &
[8] 1490214
lubuntu@lubuntu: /media/lubuntu/ab8bbe50-585b-44df-b4d7-f88457
me/guest/ux1$ find / -mtime 1 > /dev/null 2>/dev/null &
[9] 1490227
lubuntu@lubuntu: /media/lubuntu/ab8bbe50-585b-44df-b4d7-f88457
me/guest/ux1$ find / -mtime 1 > /dev/null 2>/dev/null &
[10] 1490241
lubuntu@lubuntu: /media/lubuntu/ab8bbe50-585b-44df-b4d7-f88457
me/guest/ux1$
```

modmon.awk

```
BEGIN {
    max = 20; jscale = 1; iscale = .1
    for (e=1; e<=1000; e++) doit()
}
function doit() {
    delete sto
    com = "find /home /dev -type f -mtime -1 -ls 2>/dev/null"
    while (com | getline) {
        size = log($7); name = length($11); i = name; j = size

        i = int(i*iscale+.5); j = int(j*jscale+.5)
        if (i > max) i = max; if (j > max) j = max
        sto[i,j]++
    }; close(com)
    system("clear")
    print com, e
    print "%log(size) V %namelen >"
    printf "|"; for (i=0; i<=max; i++) printf "--"; print "|"
    for (j=max; j>=0; j--) {
        printf "|"
        for (i=0; i<=max; i++) {
            count = sto[i,j]-0*hold[i,j]
            if (count > 9) count = "*"
            else if (count == 0) count = " "
            else if (count < 0) count = "-"
            printf count " "
        }
        print "|"
    }
    printf "|"; for (i=0; i<=max; i++) printf "__"; print "|"
    for (x in sto) hold[x] = sto[x]
    # system("sleep 1")
}
```

WHERE THE WILD THINGS ARE



STORY AND PICTURES BY MAURICE SENDAK

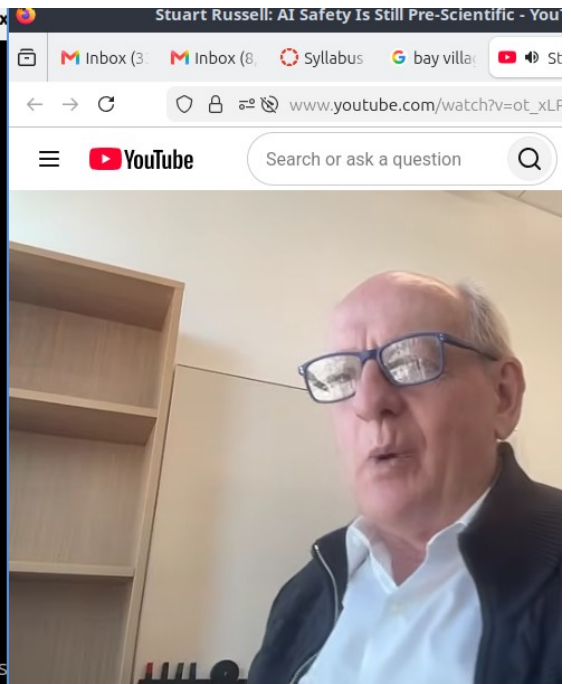

```

BEGIN {
    max = 20; jscale = 1; iscale = .3
    for (e=1; e<=1000; e++) doit()
}
function doit() {
    delete sto
    com = "sudo timeout 2 tcpdump -w capture.pcap"
    system(com)
    com = "tcpdump -nn -r capture.pcap"
    while (com | getline) {
        # 03:30:16.008762 IP6
2600:1700:5470:5c50:a9ff:6a4b:c809:6e92.47214 >
2001:4860:482b:7700::.443: Flags [P.], seq 901:940, ack 587, win
292, options [nop,nop,TS val 2018852842 ecr 1860755486], length 39
        if ($(NF-1) !~ /length/) continue
        size = log($NF); name = length($3); split(name,tt,/:/); for (i
in tt) { sum=0; gsub(/[0-9]/,"",tt[i]); sum+=tt[i] }
        i = sum; j = size; i = int(i*iscale+.5); j = int(j*jscale+.5);
if (i > max) i = max; if (j > max) j = max
        print i,j
        sto[i,j]++
    }; close(com)
    system("clear")
    print com, e
    print "%log(size) V                                %iphash >"
    printf "|"; for (i=0; i<=max; i++) printf "--"; print "|"
    for (j=max; j>=0; j--) {
        printf "|"
        for (i=0; i<=max; i++) {
            count = sto[i,j]-0*hold[i,j]
            if (count > 9) count = "*"
            else if (count == 0) count = " "
            else if (count < 0) count = "-"
            printf count " "
        }
        print "|"
    }
    printf "|"; for (i=0; i<=max; i++) printf "__"; print "|"
    for (x in sto) hold[x] = sto[x]
    # system("sleep 1")
}

```



```
lubuntu@lubuntu: /media/lubuntu/ab8bbe50-585b-44df-b4d7-f88457737259/home/guest/ux
tcpdump -nn -r capture.pcap 35
%log(size) V %iphash >
-----
1          7
          2
          1
          1
1          1
          1          1
          1          7
-----
tcpdump: listening on wlp2s0, link-type EN10MB (Ethernet), s
```



Stuart Russell: AI Safety Is Still Pre-Scientific

TRY THEM ON YOUR OWN MACHINE!

TURN THEM INTO AUDIO! <https://ieeexplore.ieee.org/document/7293430>
EVEN BETTER THAN TEXT!

Server monitoring by systems administrators is important for computer security, and usually requires well **focused visual attention**.

For example, running a top or netstat repeatedly requires that the sysadmin interrupt other tasks and read the output ...

*Moreover, one man's **Stairway to Heaven** musical alert is another man's **Copacabana**.*

For rare events like incoming phone calls, it may be acceptable to have an auralization that captures entertains, enthralls, startles, or captivates.

Although this is **clearly a good idea**, it has not found its way into **standard practice**. One problem is that tools that permit operators to generate sounds do not give much **guidance on what sounds to generate**. At best, this leaves the sysadmin with a problem of **assigning “ring tones”** to various events ...

2822	27690	716228	/usr/bin/cc
14	255	17976	/usr/bin/cd-create-profile
33	414	17768	/usr/bin/cd-fix-profile
10	93	9576	/usr/bin/cd-icdump
3419	17881	395876	/usr/bin/cdrecord
72	560	22136	/usr/bin/c++filt
32	230	9764	/usr/bin/chacl
200	1243	49420	/usr/bin/chage
159	469	4459	/usr/bin/chardet
123	1405	68272	/usr/bin/charmap
16	156	9712	/usr/bin/chattr
243	1722	59176	/usr/bin/chcon

Since sounds cannot be included in a paper, and our descriptions are bit subjective, we reproduce a part of a binary here, repeated three times:

A#H D#:#:#:#:#:#A #A#:#C #] #C A#A
K#AA#(D#:#:#:#:#C #C#:#hA#(p#8:#:#:#C #C#:#h
#L:#:#:#:#:#A#:#A#:#A#:#F#

Conferences > 2015 IEEE International Confe... ?

Auralization of process and port status using program binaries to generate semantically meaningful non-fatiguing noise as canonical sound signals

Publisher: IEEE Cite This PDF

Ronald Loui; Dylan Dozier; Evan Barber; Jeswanth Harish All Authors

60 Full Text Views



Abstract
Document Sections
I. Introduction
II. Main Idea
III. Warbled Whistling
IV. Assigning

Abstract:
We describe using auralization to monitor process stack and netstat status on a server or small group of servers. Using the auditory channel is especially useful for monitoring because it can be done without taking much attention. Our contribution is to suggest generating noise-like sounds that more naturally fade into the background, and are less fatiguing over long monitoring sessions. Although our noises are not musically or tonally interesting, they are nevertheless semantically identifiable and succeed in indicating changes of state.

Published in: 2015 IEEE International Conference on Electro/Information Technology (EIT)