

Unix et Programmation Shell

Philippe Langevin

département d'informatique
UFR sciences et technique
université du sud Toulon Var

Automne 2013

brouillon en révision

- site du cours :
<http://langevin.univ-tln.fr/cours/UPS/upsh.html>
- localisation du fichier :
<http://langevin.univ-tln.fr/cours/UPS/doc/shell.pdf>

dernières modifications

man.tex	2017-09-07	12:27:47.738251920	+0200
perm.tex	2016-09-30	09:41:54.766553521	+0200
file.tex	2016-09-30	09:19:02.810595120	+0200
bash.tex	2016-09-15	12:09:09.887948313	+0200
term.tex	2016-09-14	18:50:05.124091515	+0200
upsh.tex	2015-10-25	18:09:36.027434338	+0100
proc.tex	2015-10-20	22:09:35.450391618	+0200
shell.tex	2015-09-10	19:31:04.581529236	+0200
prologue.tex	2015-09-07	09:06:31.773157847	+0200
tools.tex	2015-07-11	09:04:38.890915266	+0200
pipe.tex	2014-10-02	19:10:22.426127326	+0200
direct.tex	2014-10-02	07:49:17.162784238	+0200
syntaxe.tex	2014-10-01	23:52:29.859357485	+0200
part.tex	2014-10-01	23:52:29.372363438	+0200

1 - shell unix

- origine
- unices
- GNU/linux
- distribution
- shell unix
- GUI vs CLI
- C-production

naissance de unix

Parmi les nombreux *hackers* du 20e, deux pionniers des BELL LABS sont à l'origine du système **unix** :

naissance de unix

Parmi les nombreux *hackers* du 20e, deux pionniers des BELL LABS sont à l'origine du système **unix** :



naissance de unix

Parmi les nombreux *hackers* du 20e, deux pionniers des BELL LABS sont à l'origine du système **unix** :



1969 Ken Thompson crée le système UNICS

naissance de unix

Parmi les nombreux *hackers* du 20e, deux pionniers des BELL LABS sont à l'origine du système **unix** :



1969 Ken Thompson crée le système UNICS

1971 Dennis Ritchie crée le langage C

naissance de unix

Parmi les nombreux *hackers* du 20e, deux pionniers des BELL LABS sont à l'origine du système **unix** :



1969 Ken Thompson crée le système UNICS

1971 Dennis Ritchie crée le langage C

[PSLC] prog. système en langage C sous Linux, par C. Blaess.

naissance de unix

Parmi les nombreux *hackers* du 20e, deux pionniers des BELL LABS sont à l'origine du système **unix** :



1969 Ken Thompson crée le système UNICS

1971 Dennis Ritchie crée le langage C

[PSLC] prog. système en langage C sous Linux, par C. Blaess.

[EPI] **unix** et l'informatique pédagogique ont le même âge !

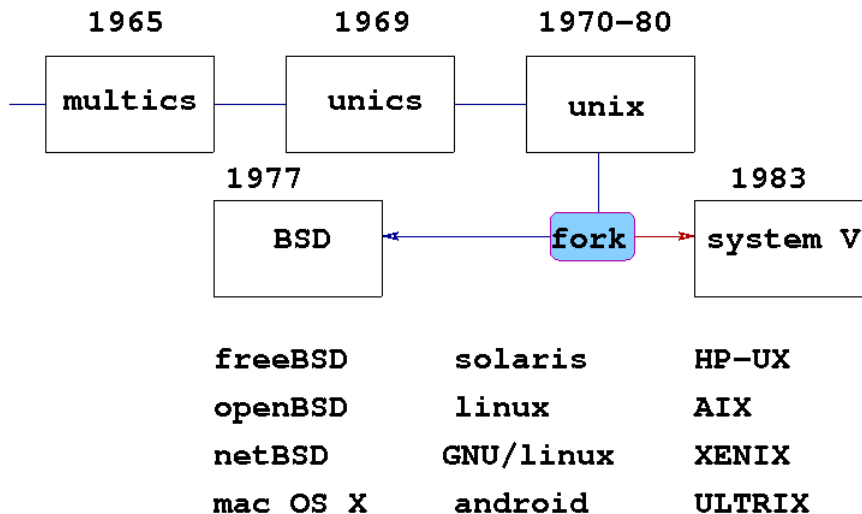
MULTICS — UNICS — UNIX



1964 MIT, General Electric et les Bell Labs d'AT&T lancent le projet MULTiplexed Information and Computing Service pour développer un nouveau système d'exploitation à temps partagé.

1969 Les Bell Labs se retire du projet.

chronologie



unix populaires

unix a donné naissance à une famille de systèmes, les *unices* dont les plus populaires sont :

1983	System V	Bell labs, AT&T.
1977	BSD	Berkeley Software Distribution
1990	GNU/Linux	Logiciel Libre
1999	OS X	next, apple.
2003	android	androïde, google.

L'ensemble des industriels acteurs du développement du système **unix** sont regroupés dans l'[opengroup](#) propriétaire de la marque **unix** dont le *Single UNIX Specification* certifie les systèmes **unix**.

influences

Trois groupes influent sur la normalisation des systèmes **unix** :

- **POSIX** : Portable Operating System Interface (IEEE).
- **BSD**
- **GNU** : Gnu is Not Unix, logiciel libre.

Je vous recommande

- la description du **projet GNU** par R. Stallman
- la lecture de **la cathédrale et le bazar** par E. Raymond.

dialecte

PS(1) Linux User's Manual

NAME

`ps` – report a snapshot of the current processes.

DESCRIPTION

`ps` displays information about a selection of the active processes. If you want a repetitive update of the selection and the displayed information, use `top`.

This `ps` version accepts several kinds of options:

- 1 `UNIX` options, must be preceded by a dash.
- 2 `BSD` options, must not be used with a dash.
- 3 `GNU` long options, preceded by two dashes.

Options of different types may be freely mixed,

but conflicts can appear

GNU/linux

Dans les salles de travaux-pratiques, vous utiliserez un système d'exploitation **GNU/linux**, fusion des composantes du logiciel libre :

- noyau **linux** (**Linus Torvalds**, 1991),
- utilitaires **GNU** (**Richard Stallman**, 1983).

Plus précisément, une distribution **ubuntu**, basée sur **debian**. Il s'agit d'un environnement de travail **unix** de qualité issu du logiciel libre ? !.

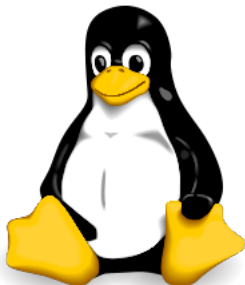
- **hurd** : le noyau **GNU** n'est pas encore opérationnel.

GNU/linux

libérez unix!



Hello everybody out there using



minix

Le nom GNU/Linux fut utilisé pour la première fois par debian en 1994 pour le nom de leur distribution du système d'exploitation basé sur le noyau Linux et des logiciels GNU.

le 27 septembre 1983 : libérez unix !

À partir de Thanksgiving je vais écrire un système logiciel complet compatible avec Unix appelé GNU, et le distribuer librement à quiconque voudra l'utiliser. Il y a grand besoin de contributions sous forme de temps, d'argent, de programmes et d'équipement. Pour commencer, GNU comprendra un noyau ainsi que tous les utilitaires requis pour écrire et faire tourner des programmes C : éditeur, interpréteur de commandes, compilateur C, éditeur de liens, assembleur et quelques autres encore. Par la suite, nous ajouterons un formateur de texte, un YACC, un jeu Empire, un tableur et des centaines d'autres choses. À terme, nous espérons fournir tous les composants utiles venant normalement avec un système Unix, ainsi que tout autre composant utile, y compris de la documentation en ligne et imprimée...

– Richard Stallman

paquets gnu (2012)

a2ps acct acm adns aetherspace alive anubis archimedes aris aspell
auctex autoconf autoconf-archive autogen automake avl
ballandpaddle barcode bash bayonne bazaar bc bfd binutils bison bool
bpel2owfn c-graph ccaudio ccide ccrtp ccscript cflow cgicc chess cim
classpath classpathx clisp cobol combine commoncpp complexity
config coreutils cpio cppi cssc dap dc ddd ddrescue dejagnu denemo
dia dico diction diffutils dionysus dismal djgpp dmd dominion dotgnu
dotgnu-forum dotgnu-pnet dr-geo ed edma electric emacs
emacs-muse emms enscript eprints epsilon fdisk ferret findutils
fontutils freedink freefont freeipmi freetalk fribidi gama garpd gawk
gcal gcc gcide gcl gcompris gdb gdbm gengen gengetopt gettext
gforth ggradebook ghostscript gift gimp gleem glib global glpk glue
gmediaserver gmorph gmp gnash gnat gnats gnatsweb gnome
gnowsys gnu-arch gnu-c-manual gnu-crypto

gnaue gnubatch gnubg gnubiff gnubik gnucap gnucash gnucomm
gnue gnufm gnugo gnuit gnujdoc gnujump gnukart gnulib gnumach
gnumed gnumeric gnump3d gnun gnunet gnupg gnupod
gnuprologjava gnuradio gnurobots gnuschool gnushogi gnuskies
gnusound gnuspeech gnuspool gnustandards gnustep gnutls
gnutrition gnuzilla goptical gorm gpaint gperf gprolog grabcomics
greg grep gretl groff grub gsas gsegrafx gsl gsrc gss gtick gtk+
gtypist guile guile-dbi guile-gnome guile-gtk guile-ncurses guile-rpc
gurgle gv gvpe gxmessage gzip halifax health hello help2man hp2xx
httpunnel hurd hyperbole icecat idutils ignuit indent inetutils
intlfonts jacal java-getopt jdresolve jel jwhois kawa kopi leg less libc
libcdio libextractor libffcall libgcrypt libiconv libidn libmatheval
libmicrohttpd libredwg librejs libsigsegv libtasn1 libtool libunistring
libxmi lightning lilypond linux-libre liquidwar6 lispintro lrzsz lsh m4
macchanger mailman mailutils

make marst maverik mc mcron mcsim mdk mediagoblin melting
metaexchange metahtml mifluz mig miscfiles mit-scheme moe motti
mpc mpfr mtools myserver nana nano ncurses nettle network ocrad
octave oleo orgadoc osip packaging panorama paperclips parallel
parted pascal patch paxutils pcb pdf pem pexec pgccfd
phantom-home phpgroupware pies pipo plotutils polyxmass
powerguru proxyknife pspp psychosynth pth pythonwebkit qexo
quickthreads radius rcs readline recutils reftex rottlog rpge rush
sather scm screen sed serveez sharutils shishi shmm shtool sipwitch
slib smalltalk smarteiffel snakecharmer social solfege sourceinstall
spacechart speex spell sqlltutor src-highlite stalkerfs stow stump
superopt swbis sysutils talkfilters tar termcap termutils tesseq
teximpatient texinfo texmacs thales time tramp trans-coord trueprint
units unrtf userv uucp vc-changelog vc-dwim vcdimager vera vmgen
vmslib w3 wb wdiff websocket4j webstump wget which womb xaos
xboard xhippo xlogmaster xmlat xnee xorriso zile

On 25 August 1991, newsgroup comp.os.minix

Hello everybody out there using minix.

I'm doing a (free) operating system (just a hobby, won't be big and professional like gnu) for 386(486) AT clones. This has been brewing since april, and is starting to get ready. I'd like any feedback on things people like/dislike in minix, as my OS resembles it somewhat (same physical layout of the file-system (due to practical reasons) among other things). I've currently ported bash(1.08) and gcc(1.40), and things seem to work. This implies that I'll get something practical within a few months, and I'd like to know what features most people would want. Any suggestions are welcome, but I won't promise I'll implement them :-)

Linus (torvalds@kruuna.helsinki.fi)

PS. Yes – it's free of any minix code, and it has a multi-threaded fs. It is NOT portable (uses 386 task switching etc), and it probably never will support anything other than AT-harddisks, as that's all I have :-(. —Linus Torvalds

monolithique vs micro-noyau

Le système `minix` est système minimaliste créé en 1987 par [Andrew Tannenbaum](#) pour illustrer son fameux cours sur les systèmes d'exploitation.

- *To me, writing a monolithic system in 1991 is a truly poor idea.*

monolithique vs micro-noyau

Le système **minix** est système minimaliste créé en 1987 par **Andrew Tannenbaum** pour illustrer son fameux cours sur les systèmes d'exploitation.

- *To me, writing a monolithic system in 1991 is a truly poor idea.*
- *Je persiste à penser que concevoir un noyau monolithique en 1991 est une erreur fondamentale. Estime-toi heureux de ne pas être un de mes étudiants. Tu n'obtiendrais pas une bonne note pour une telle conception :-)*

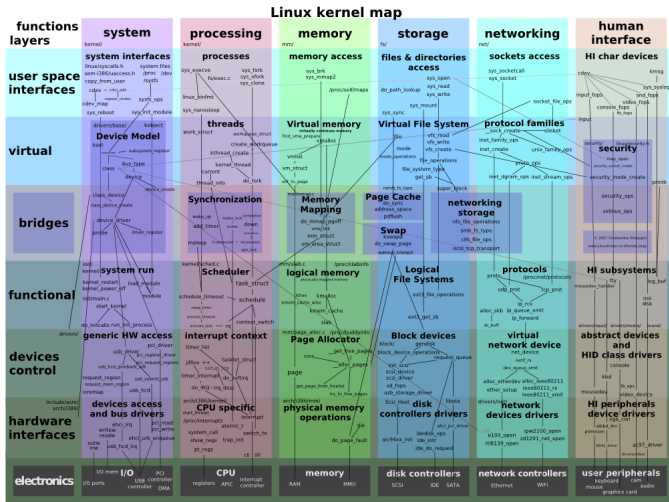
monolithique vs micro-noyau

Le système `minix` est système minimaliste créé en 1987 par [Andrew Tannenbaum](#) pour illustrer son fameux cours sur les systèmes d'exploitation.

- *To me, writing a monolithic system in 1991 is a truly poor idea.*
- *Je persiste à penser que concevoir un noyau monolithique en 1991 est une erreur fondamentale. Estime-toi heureux de ne pas être un de mes étudiants. Tu n'obtiendrais pas une bonne note pour une telle conception :-)*

`linux >2` est un noyau monolithique modulaire.

noyau linux



distribution

Le système **GNU/Linux** est mis en forme au sein de plusieurs distributions qui intègrent le noyau et les utilitaires avec

- une politique de distribution,
- un système de maintenance,
- une communauté

```
~> uname -m -o -r -s
```

```
Linux 2.6.34.9-69.fc13.i686 i686 GNU/Linux
```

```
~> ssh pl@192.168.0.150 uname -mors
```

```
Linux 3.8.13-100.fc17.i686.PAE i686 GNU/Linux
```

```
~> ssh pl@imath01
```

```
Linux 2.6.18-348.4.1.el5.centos.plus x86_64 GNU/  
Linux
```

uname -help (man uname)

```
~> uname --help
```

```
Utilisation: uname [OPTION]
```

```
Affiche certaines informations systeme
```

```
-a, --all
```

```
-s, --kernel-name          nom du noyau
```

```
-n, --nodename             hostname
```

```
-r, --kernel-release      version du noyau
```

```
-v, --kernel-version      version du kernel
```

```
-m, --machine             materiel
```

```
-p, --processor           type de processeur
```

```
-i, --hardware-platform  plate-forme
```

```
-o, --operating-system    systeme
```

distribution



distribution

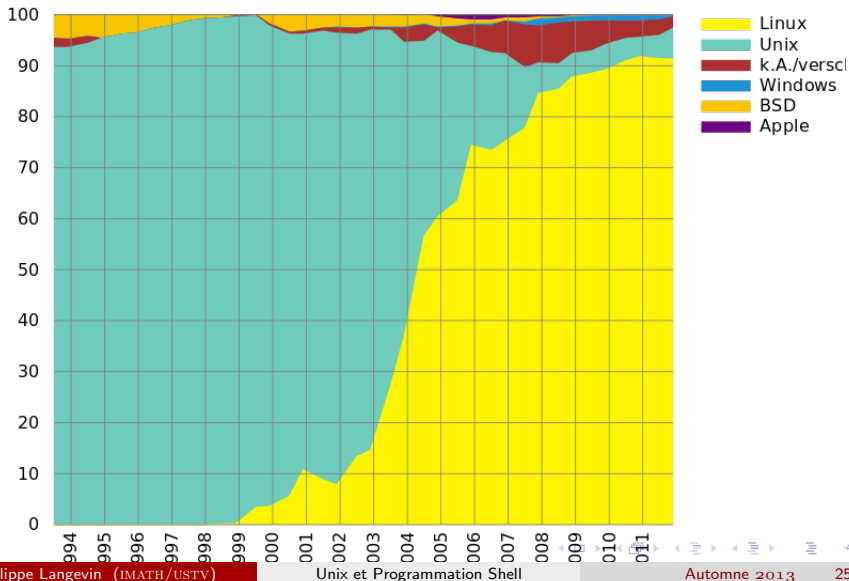


position

D'après la [linux foundation](#), en juin 2012, 98% des [supercalculateurs](#) du top500 sont essentiellement équipés d'un système `linux` :

	2012
<code>linux</code>	92.4
<code>unix</code>	5
<code>windows</code>	0.4
<code>BSD</code>	0.2

systemes des supercalculateurs



position

- En 2013, United Space Alliance remplace les `windows` et `scientific linux` des ordinateurs de la station spatiale ISS par une distribution `debian` de `GNU/linux`.

position

- En 2013, United Space Alliance remplace les `windows` et `scientific linux` des ordinateurs de la station spatiale ISS par une distribution `debian` de `GNU/linux`.
- La plupart des serveurs du réseau internet tournent sous un système `unix`.

position

- En 2013, United Space Alliance remplace les `windows` et `scientific linux` des ordinateurs de la station spatiale ISS par une distribution `debian` de `GNU/linux`.
- La plupart des serveurs du réseau internet tournent sous un système `unix`.
- matériel embarqué, matériel réseau

position

- En 2013, United Space Alliance remplace les `windows` et `scientific linux` des ordinateurs de la station spatiale ISS par une distribution `debian` de `GNU/linux`.
- La plupart des serveurs du réseau internet tournent sous un système `unix`.
- matériel embarqué, matériel réseau
- tablette, téléphone

shell ?

La terminologie de la communauté **unix** comprend quelques bizarreries, sigles et acronymes plus ou moins célèbres :

- Portable Operating System Interface X
- foo, bar ??
- biff
- shell ? bash ! shabang #!

[[ESR's jargon file](#)]

[[RFC-3092](#)] International Engineering Task Force.

[[FAQ](#)] : **unix** Frequently Asked Questions.

cowsay -W30 'it is easier...

```

____
/  it is easier to port a  \
|  shell than a shell     |
\  script (Larry Wall)   /
____

```

```

      ^  _  ^
      --  --
      (oo)\ -----
      (--)\           )\ \
           ||-----w  |
           ||           ||

```

La déclaration de **Wall** fait référence au partage d'un shell **unix** sur un système **windows**. Le hacker est l'inventeur d'un langage de manipulation de fichiers textes. Lequel ?

cowsay -W30 'it is easier...

```

____
/  it is easier to port a  \
|  shell than a shell     |
\  script (Larry Wall)   /
____

```

```

      ^  _  ^
      --  --
     (oo)\ -----
     (--)\          )\ \
          ||-----w  |
          ||          ||

```

La déclaration de **Wall** fait référence au partage d'un shell **unix** sur un système **windows**. Le hacker est l'inventeur d'un langage de manipulation de fichiers textes. Lequel ?

ystème d'exploitation

hello world

utilisateur

commande
application
bibliothèque

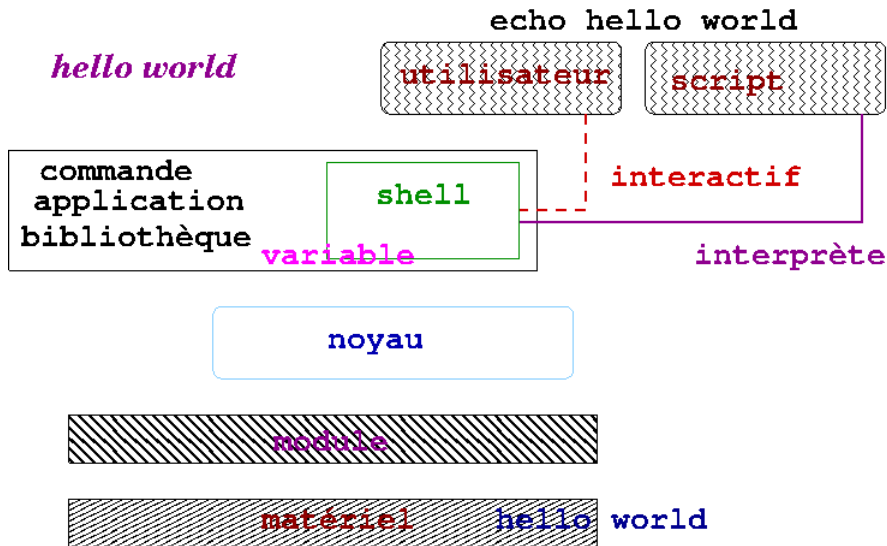
interface

noyau

module

matériel `hello world`

ystème d'exploitation



Thompson shell

La notion de **shell** apparaît dans le système MULTICS, il s'agit d'une application à l'interface entre le noyau et les utilitaires. Le shell développé au début des années 70 par Thompson est un interprète de commandes qui introduit la syntaxe des deux opérateurs fondamentaux des systèmes **unix**, la *redirection* :

```
commande > destination < source
```

et le *pipeline* de Douglas McIlroy

```
commande | commande
```

[[man 1 sh](#)]

shell unix

Un *shell unix* est une commande qui permet d'accéder aux fonctionnalités du système d'exploitation :

- utilisation des fichiers et commandes externes
- contrôle des processus

Un *shell* est un langage de programmation qui traitent des variables : globale, locale et exportée au moyen de commande interne. Un *processus shell* possède un des modes d'exécutions :

- login (initialisation)
- interactif (ligne de commande)
- interprète (script)

traçage des appels

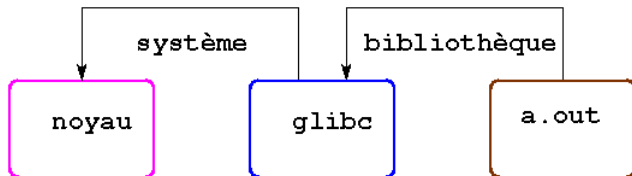
Les débogueurs `ltrace` et `strace` renseignent sur les appels bibliothèques et systèmes effectués par un processus :

```
1 #include <stdio.h>
2 int main( void )
3 {
4 puts("hello world");
5 return 0;
6 }
```

traçage des appels

Les débogueurs `ltrace` et `strace` renseignent sur les appels bibliothèques et systèmes effectués par un processus :

```
1 #include <stdio.h>
2 int main( void )
3 {
4 puts("hello world");
5 return 0;
6 }
```



```
puts("hello world")
```

```
1 #!/bin/bash
2 gcc -Wall hw.c
3 ltrace ./a.out |& cut -c1-50 > hw.out
4 strace -e trace=read,write,file ./a.out\  
5 |& cut -c1-50 >> hw.out
```

```
puts("hello world")
```

```

__libc_start_main(0x80483b4, 2, 0xbfa92db4, 0x8048
puts("hello world") =
hello world
+++ exited (status 0) +++
execve("./a.out", ["./a.out"], [/* 52 vars */]) =
access("/etc/ld.so.preload", R_OK) = -1 ENOEN
open("/etc/ld.so.cache", O_RDONLY) = 3
open("/lib/libc.so.6", O_RDONLY) = 3
read(3, "\177ELF\1\1\1\3\0\0\0\0\0\0\0\0\3\0\3\0\1
write(1, "hello world\n", 12hello world
) = 12

```

echo hello world

Un exemple plus complexe

```
1 #!/bin/bash
2 ltrace  echo hello world |& \  
3   cut -c1-50 >hello-lib.out
4
5 strace -e trace=open,read,write,file \  
6   echo hello world |& \  
7   cut -c1-50 >hello-sys.out
```


echo hello world

```

__libc_start_main(0x8049070, 3, 0xbfa69174, 0x804b
getenv(" POSIXLY_CORRECT")           =
strchr(" echo", '/')                  =
setlocale(6, "")                       =
bindtextdomain(" coreutils", "/usr/share/locale") =
textdomain(" coreutils")              =
__cxa_atexit(0x804a000, 0, 0, 0x804e5b8, 0xbfa690c
fputs_unlocked(0xbfa695ad, 0xc5e4e0, 0xc5c1e0, 0,
fputs_unlocked(0xbfa695b3, 0xc5e4e0, 0xc5c1e0, 0,
exit(0 <unfinished ... >
__fpending(0xc5e4e0, 0xc5e4e0, 0xb42a24, 0xbfa68fd
fclose(0xc5e4e0hello world
)                                       = 0
__fpending(0xc5e580, 0xc5e4e0, 0xb42a24, 0xbfa68fd
fclose(0xc5e580)                       =
+++ exited (status 0) +++

```

echo hello world

```

execve("/bin/echo", ["echo", "hello", "world"], [
access("/etc/ld.so.preload", R_OK)          = -1 ENOEN
open("/etc/ld.so.cache", O_RDONLY)          = 3
open("/lib/libc.so.6", O_RDONLY)           = 3
read(3, "\177ELF\1\1\1\3\0\0\0\0\0\0\0\3\0\3\0\1
open("/usr/lib/locale/locale-archive", O_RDONLY|O_
write(1, "hello world\n", 12)hello world
)
= 12

```

Bourne [Again] shell

Le **Bourne shell** est introduit dans le système UNIX V7. Il a été développé par **Stephen Bourne** avec l'objectif de créer un outil de scriptage des commandes pour faciliter l'administration du système. **sh** est devenu populaire pour sa facilité d'emploi et sa rapidité, standard des systèmes UNIX, c'est toujours le shell par défaut du compte **root**, de certaines commandes **make**.

Bourne [Again] shell

Le **Bourne shell** est introduit dans le système UNIX V7. Il a été développé par **Stephen Bourne** avec l'objectif de créer un outil de scriptage des commandes pour faciliter l'administration du système. **sh** est devenu populaire pour sa facilité d'emploi et sa rapidité, standard des systèmes UNIX, c'est toujours le shell par défaut du compte **root**, de certaines commandes **make**.

Le shell de base **sh**

Les **alias**, l'historique des commandes et le contrôle des processus ne sont pas gérés par **sh**.

Le **Bourne Again shell** est un projet GNU **bash** démarré en 1980 par Brian Fox, actuellement maintenu par Chet Ramey.

shell par défaut

```
~> /bin/sh  
sh-4.1$ exit  
exit
```

```
~> which sh  
/bin/sh
```

```
~> ls -l /bin/sh  
lrwxrwxrwx. 1 root root 4 9 aout 08:47 /bin/sh  
-> bash
```

```
~> echo $SHELL  
/bin/bash
```

shell par défaut

```
~> echo $SHELL  
/bin/bash
```

```
~> echo -e 'all:\n\techo $(SHELL)' > /tmp/  
makefile
```

```
~> make -c /tmp  
make: entrant dans le repertoire /tmp  
echo /bin/sh  
/bin/sh  
make: quittant le repertoire /tmp
```

shell populaires

Les shells font légion.

shell	shabang
sh	6 490 000
bash	5 380 000
ash	345 000
ksh	323 000
csch	254 000
tcsh	116 000
zsh	86 000

TABLE : popularité de quelques shells

Ces slides concernent mon shell préféré `bash`...

Après le `lambis` bien entendu !

[comparatif des shells]

Interfaces

L'utilisateur d'un ordinateur interagit avec le système d'exploitation au moyen d'une interface graphique ou bien d'une interface texte.

- Graphic User Interface
- Command Line Interface

La notion de `shell` est étendue à l'ensemble des interfaces y compris graphique : `gnome`, `kde`, `lxde`, `xfce`, `awesome` ...

- L'évolution des GUIs fait souvent débat !
- Le monde des CLIs paraît plus zen...

suggestion : essayez les interfaces !

CLI vs GUI

- Le mode graphique se veut intuitif et facile d'emploi.
- Le mode textuel est puissant mais peu intuitif.

Il n'y a pas lieu d'opposer les deux modes qui sont parfaitement complémentaires, les tâches sur les fichiers textes sont plus faciles à réaliser par la console.

En général, lors d'une session graphique, les utilisateurs ouvrent des pseudos terminaux pour effectuer certaines tâches avec l'interprète de commande.

C-production

Evaluer une C-production

combien de lignes de codes C dans le répertoire `./CC?`

C-production

Evaluer une C-production

combien de lignes de codes C dans le répertoire `./CC?`

Pas de solution graphique !

Difficile d'interpréter cette question par des mouvements de souris. . .

C-production

Evaluer une C-production

combien de lignes de codes C dans le répertoire ./CC ?

Pas de solution graphique !

Difficile d'interpréter cette question par des mouvements de souris. . .

Plusieurs solutions textuelles

Au fil des années, les applications `unix` ont été développées, améliorées pour répondre efficacement ce type de questions.

Une ligne de commande

```
$ find ~/CC -name "*.c" | xargs cat | wc -l
```

```
400942
```

Une ligne de commande

```
$ find ~/CC -name "*.c" | xargs cat | wc -l  
400942
```

Une solution facile à comprendre plus difficile à reproduire sans connaître les usages des commandes et arguments passés par la ligne de commande.

- `find` : outil pour la recherche de fichiers.
- `wc` : compter les lignes, mots, octets.

Paul Rubin et [David MacKenzie](#)

- `xargs` : construire et exécuter des lignes de commandes.

Une ligne de commande

```
$ find ~/CC -name "*.c" | xargs cat | wc -l
```

```
400942
```

Une solution facile à comprendre plus difficile à reproduire sans connaître les usages des commandes et arguments passés par la ligne de commande.

- `find` : outil pour la recherche de fichiers.
- `wc` : compter les lignes, mots, octets.

Paul Rubin et [David MacKenzie](#)

- `xargs` : construire et exécuter des lignes de commandes.

```
$ cat $(find ~/CC -name '*.c' ) 2>/dev/null | wc  
-l
```

```
400942
```

Une autre solution

Une solution basée sur le filtre `awk` ?

Une autre solution

Une solution basée sur le filtre `awk` ?

```
$ find ~/CC -name "*.c" -exec wc -l {} \;  
  | awk 'BEGIN {s=0} {s=s+$0} END {print s}'
```

```
400942
```

Une autre solution

Une solution basée sur le filtre `awk` ?

```
$ find ~/CC -name "*.c" -exec wc -l {} \;  
    | awk 'BEGIN {s=0} {s=s+$0} END {print s}'
```

400942

- `find` : outil pour la recherche de fichiers.
- `wc` : compter les lignes, mots, octets.
- `awk` : commande d' [Alfred Aho](#), [Peter Weinberger](#) et [Brian Kernighan](#) pour filtrer les lignes d'un fichier texte.

script

Le rôle d'un informaticien est de réaliser des travaux avec une machine, le plus souvent, dans un temps limité :-).

Il convient d'éviter des pertes de temps qui ont pour origine

- erreurs
- lacunes

Un *script* est un fichier de commandes pour le shell. La première ligne du script peut préciser le shell d'exécution :

```
1 #!/bin/myshell  
2 ...
```

c'est le fameux **shabang**.

exemple

```
1 #!/bin/bash
2 sum=0
3 while read num rem
4 do
5 echo $num $rem
6 let sum+= $num
7 done <<(find $1 -name $2 -exec wc -l {} \;)
8 echo $sum
```

Après avoir sauvé ces lignes dans un fichier `count.sh` :

```
~> chmod u+x count.sh
```

pour rendre le script exécutable, on peut alors lancer

```
~> ./count.sh ~/CC '*.c'
```

commande complexe

Avec de la pratique, il est possible de d'exécuter une tâche complexe directement en ligne de commande :

```
↪ find ~ -name "*.txt" -exec wc -l {} \;  
| ( sum=0; while read num rem ;  
do let sum+=num; done; echo $sum) > sum.txt
```

Il s'agit d'une *commande composée* s'articulant sur des *commandes simples* incluant un *pipeline* et une *redirection*.