

Système d'exploitation

Thomas Vantroys

thomas.vantroys@polytech-lille.fr

Polytech'Lille
Université de Lille

IMA4
2020 - 2021



Université de Lille



POLYTECH
VILLE

Architecture des OS

Introduction

Le système d'exploitation contrôle les ressources de l'ordinateur et fournit la base pour les autres applications

Le système d'exploitation fonctionne sous deux modes différents :

- mode **noyau**
- mode **utilisateur**

Il doit remplir deux fonctions :

① Machine virtuelle :

- enrobe et masque le matériel avec une couche logicielle
- présente une interface plus facile à comprendre et à programmer

② Gestionnaire de ressources :

- ordonner et contrôler l'allocation des ressources (processeurs, mémoire, entrées/sorties, ...) entre les différents programmes qui y font appel.
- crucial pour les systèmes multi-tâches

Classes de systèmes d'exploitation

Il existe différentes classes de systèmes d'exploitation en fonction :

- des services rendus
- de leur architecture logique
- de l'architecture matérielle

Services rendus

- **Mono/Multi Tâches :**
capacité du système à pouvoir (ou pas) exécuter plusieurs processus simultanément
- **Mono/Multi Utilisateur :**
capacité à pouvoir (ou pas) gérer un panel d'utilisateurs utilisant simultanément les mêmes ressources matérielles

Architecture des systèmes

• Systèmes centralisés :

- l'ensemble du système est entièrement présent sur la machine
 - les machines éventuellement reliées sont vues comme des entités étrangères disposant elles aussi d'un système centralisé
 - ne gère que les ressources de la machine sur laquelle il est présent

- Systèmes réparties ou distribués :

- les différentes abstractions du système sont réparties sur un ensemble (domaine) de machines (sites)
 - le système apparaît aux yeux de l'utilisateur comme une machine unique (virtuelle)
 - l'utilisateur ne se soucie pas de la localisation des ressources
 - exploitation des capacités de parrallélisme d'un domaine
 - assez résistant aux pannes

Architecture matérielle

① **Monoprocesseur** : temps partagé ou multiprogrammation

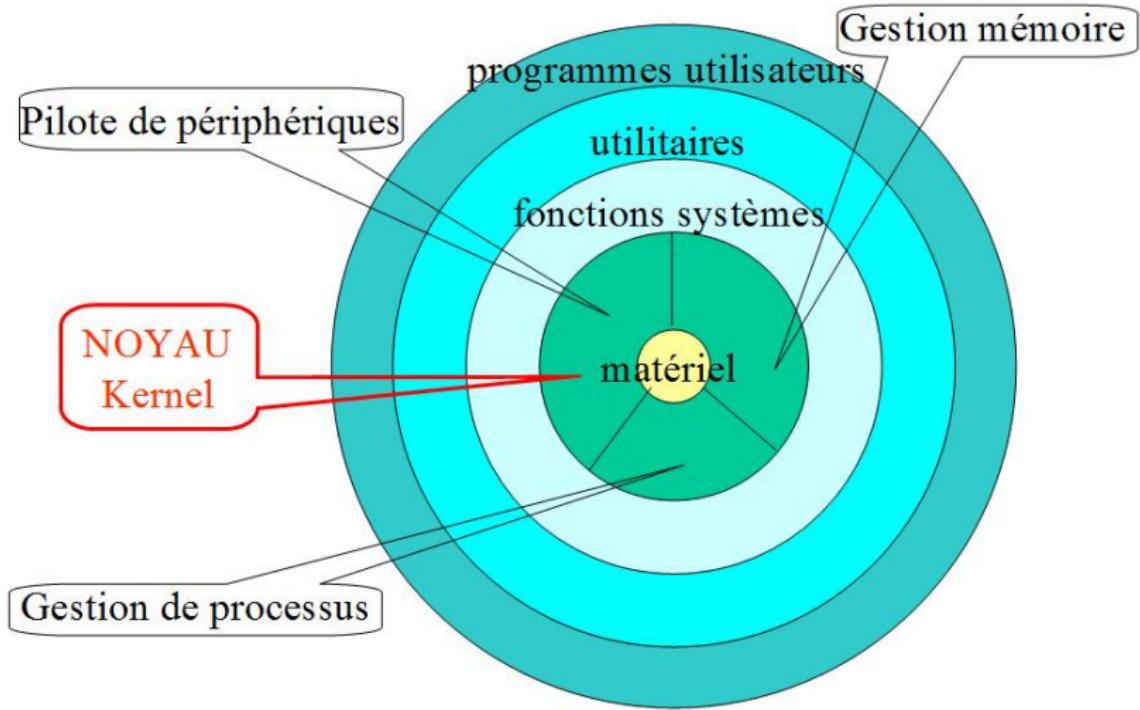
Mécanisme de gestion des processus pour offrir un pseudo parallélisme à l'utilisateur. En fait c'est une communication rapide entre processus qui permet de donner l'illusion du parallélisme.

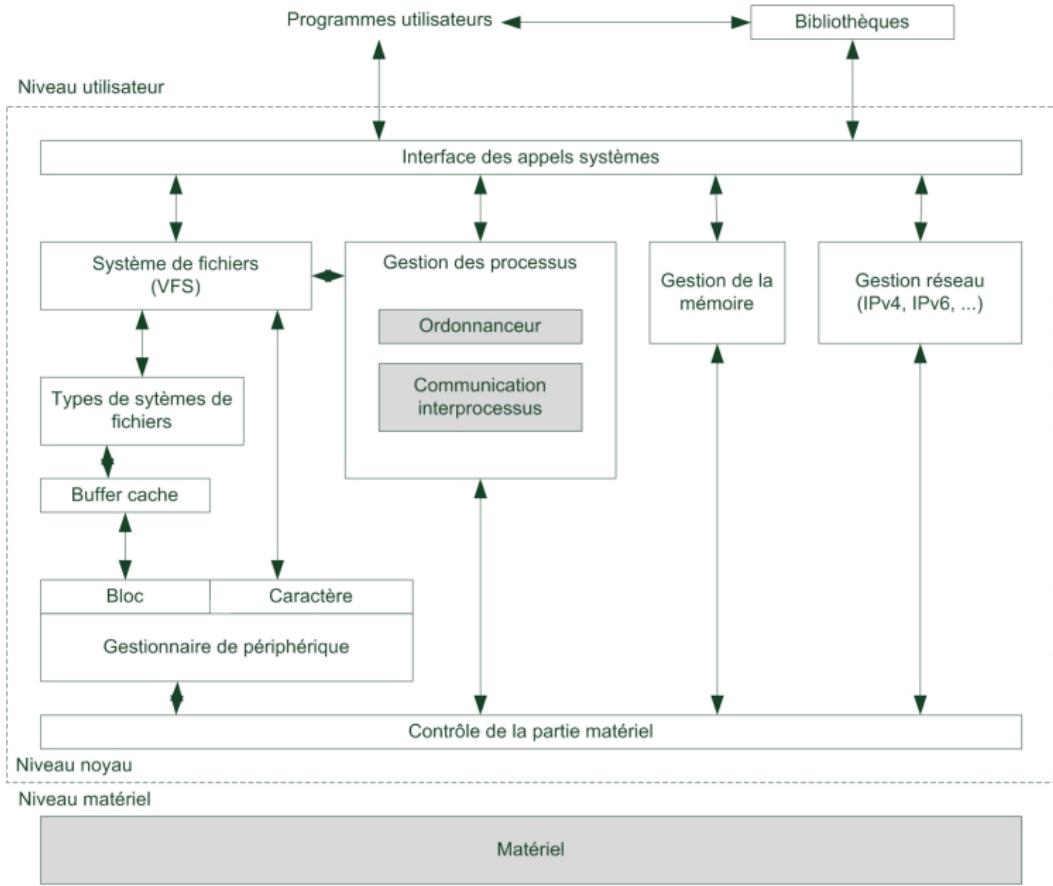
② **Multiprocesseur** : parallélisme

Différentes variétés :

- SIMD (Single Instruction Multiple Data) : tous les processus exécutent les mêmes instructions mais sur des données différentes
- MIMD (Multiple Instruction Multiple Data) : chaque processeur est complètement indépendant et exécute des instructions sur des données différentes
- Pipeline : les différentes unités d'exécution sont mises en chaîne et font chacune une partie du traitement à effectuer.

Couches du système d'exploitation





Aperçu du système de fichiers

- Vue physique

- pour accéder à un bloc de 512 octets il faut donc connaître le cylindre, le plateau et le secteur sur lequel il se trouve ce qui est très fastidieux d'où une interface utilisateur, appelé système de gestion de fichiers.

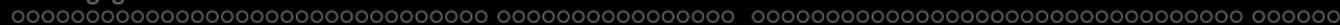
- Vue logique

- le système de gestion de fichiers fournit une vue logique faisant abstraction des propriétés physiques des dispositifs de stockage.

Fichier

C'est un ensemble d'informations en relation entre elles :

- Les fichiers de *données* peuvent être numériques, alphabétiques, alphanumériques ou binaires.
- Un fichier *texte* est une séquence de caractères organisée en lignes, éventuellement en pages.
- Un fichier *source* est fichier texte ayant une structure formée de séquences de routines et de fonctions organisées pour être reconnu par un compilateur.
- Un fichier *objet* est une séquence d'octets organisée de façon compréhensible pour un éditeur de liens.
- Un fichier *exécutable* est une séquence de segments de code que le chargeur peut amener en mémoire et exécuter.
- ...



Fichier

- ...
- Il peut être sans format particulier.
- C'est une séquence de bits, octets, lignes ou enregistrement dont la sémantique est définie par le créateur et l'utilisateur

En conclusion, c'est un concept extrêmement général.

Fichier (nommage)

Un fichier est nommé pour la commodité d'utilisation des humains que nous sommes.

- Le nom est généralement une chaîne de caractères
- Certains OS font une différence entre majuscules et minuscules

Fichier (attributs)

- Autre attributs que le nom pouvant varier selon les systèmes de gestion de fichiers.
 - Type (si le système supporte différents types de fichiers)
 - Emplacement : pointe sur le fichier
 - Taille : taille courante du fichier (octets, mots ou blocs)
 - Protection : information concernant le contrôle d'accès (qui peut lire, écrire, exécuter etc.)
 - Date et heure de création, dernière modification, dernière utilisation
- Ces informations sont généralement rangées dans la structure de répertoire qui réside également sur le support.

Fichier (opérations sur)

- Créer un fichier :
 - Trouver de la place dans le système de fichiers
 - Créer une entrée dans le répertoire
 - L'entrée du répertoire mémorise le nom du fichier et son emplacement dans le système de fichiers
- Écrire
 - Au travers d'un appel système, il faut rechercher le répertoire afin de trouver l'emplacement du fichier
 - Le système doit maintenir un pointeur d'écriture vers l'emplacement dans le fichier où l'écriture suivante doit avoir lieu
 - Ce pointeur doit être actualisé à chaque fois que l'on écrit

Fichier (opérations sur)

- Lire
 - Avec utilisation d'un appel système on spécifie le nom du fichier et où placer l'information (en mémoire) à lire
 - Le système maintient un pointeur de lecture vers l'emplacement où la lecture suivante doit avoir lieu
 - Une fois la lecture terminée, le pointeur de lecture doit être actualisé.
- Détruire un fichier
 - Recherche du fichier nommé
 - Libération de tout l'espace disque occupé par ce fichier
 - Suppression de l'entrée dans le répertoire

Structure des répertoires

- Stockage de milliers de fichiers
- Organisation des données
- 1^{ere} découpe sous forme de partitions
- Chaque partition contient de l'information sur les fichiers qui la composent
- Information maintenue dans les entrées d'un **répertoire**
- Ce répertoire mémorise l'information (nom, emplacement, taille, ...) sur tous les fichiers de cette partition

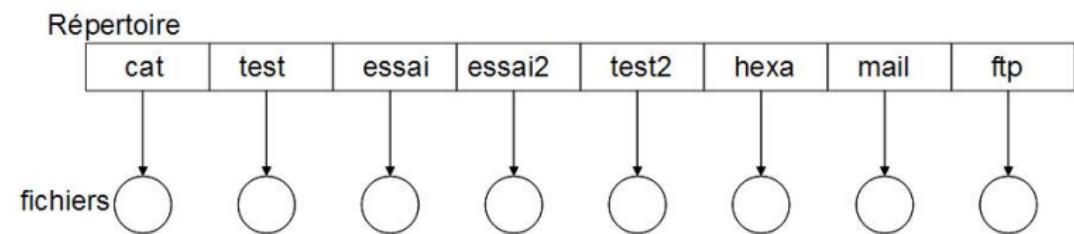
Structure des répertoires

- Le répertoire peut être organisé de plusieurs façons mais nous souhaitons pouvoir insérer, supprimer, chercher une entrée, énumérer les entrées.
- Rechercher un fichier. Il faut être capable de trouver l'entrée d'un fichier particulier
- Créer un fichier. Créer une nouvelle entrée dans le répertoire
- Détruire un fichier. Éliminer une entrée dans le répertoire
- Renommer un fichier. Changer le nom de l'entrée dans le répertoire

Structure des répertoires

- Répertoire à un niveau

- Structure la plus simple. Tous les fichiers se trouvent dans le même répertoire
- Tous les fichiers doivent avoir des noms uniques
- Problème évident en cas de plusieurs usagers.



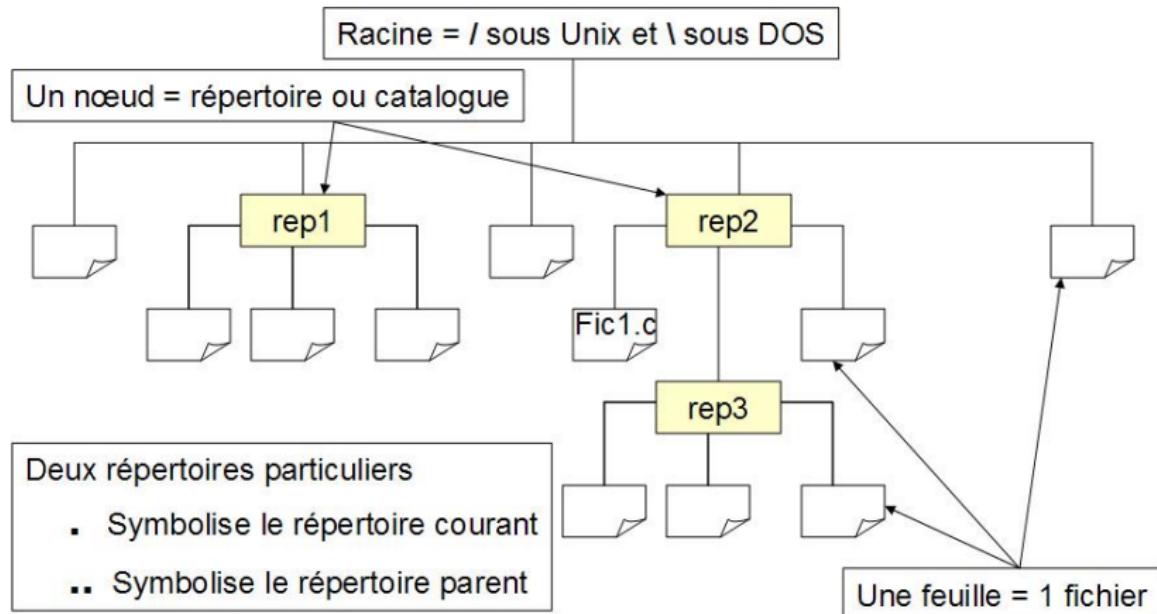
Structure des répertoires

- Structure arborescente

- Généralisation permettant aux utilisateurs de créer leurs propres sous-répertoires et d'organiser leurs données
- Chaque fichier possède un chemin d'accès unique.
- Un répertoire (ou sous-répertoire) contient un ensemble de fichiers ou de sous-répertoires
- Un sous-répertoire est un fichier traité de manière particulière
- Il existe des primitives spéciales de création et de destruction de répertoire (`mkdir`, `rmdir`, ...)
- La destruction passe généralement par le "vidage" préalable du répertoire

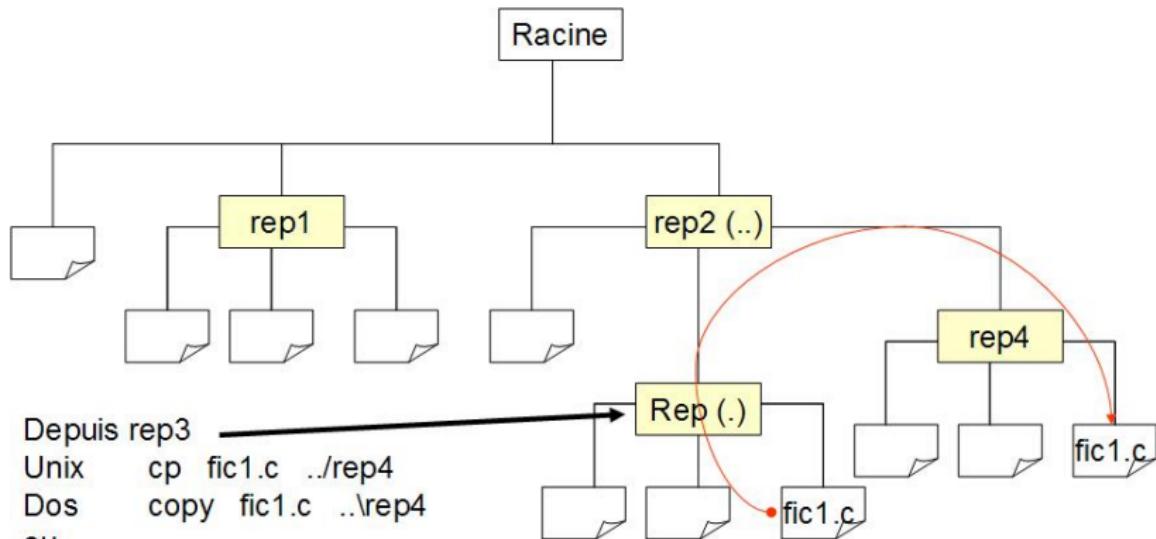
Le répertoire

Structure arborescente



Le répertoire

Structure arborescente



Depuis rep3

Unix cp fic1.c ..\rep4

Dos copy fic1.c ..\rep4

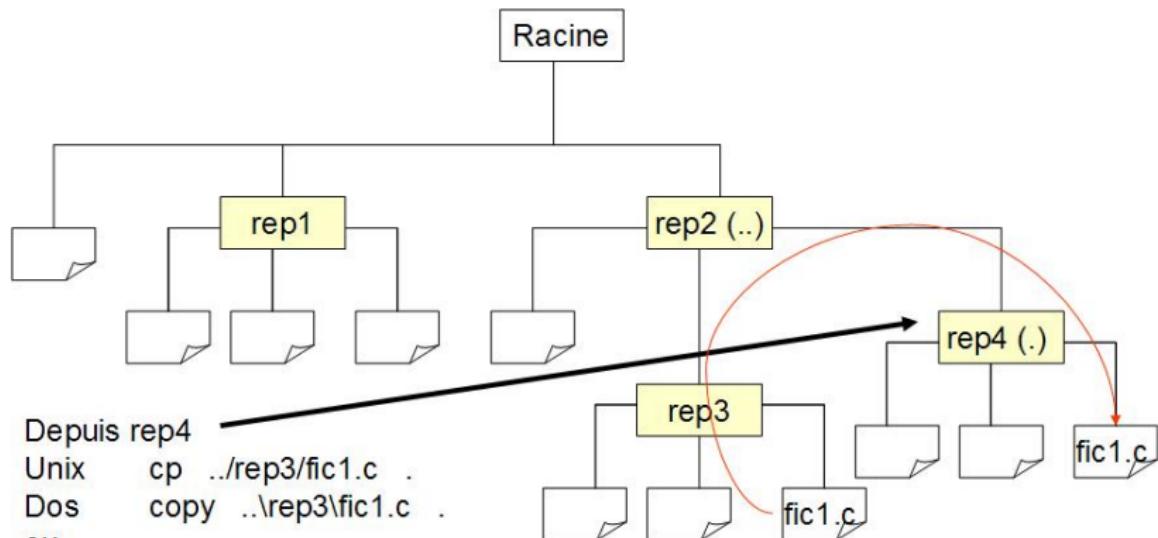
ou

Unix cp fic1.c /rep2/rep4

Dos copy fic1.c \rep2\rep4

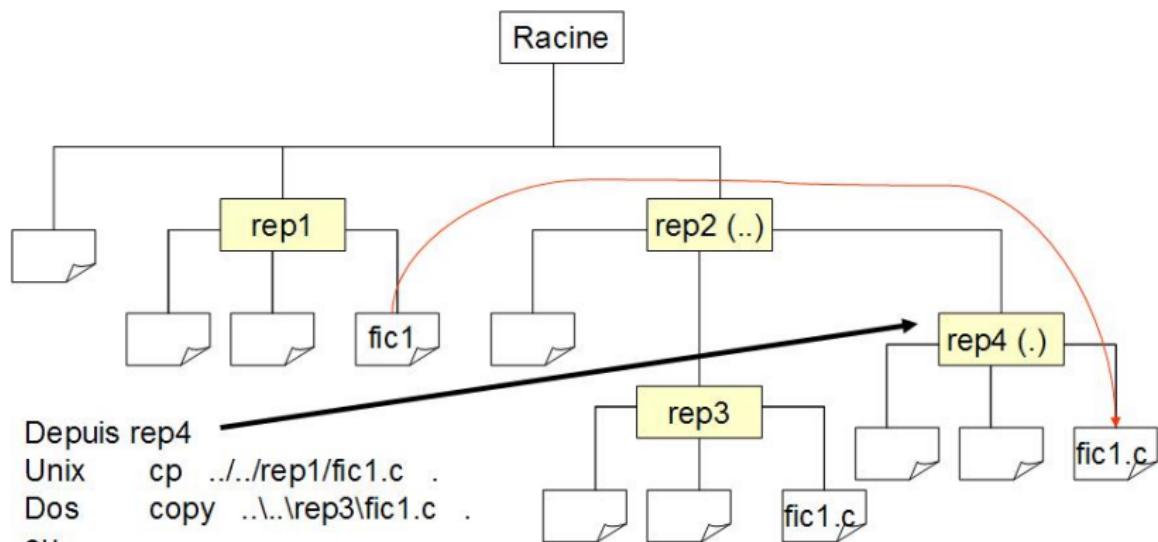
Le répertoire

Structure arborescente



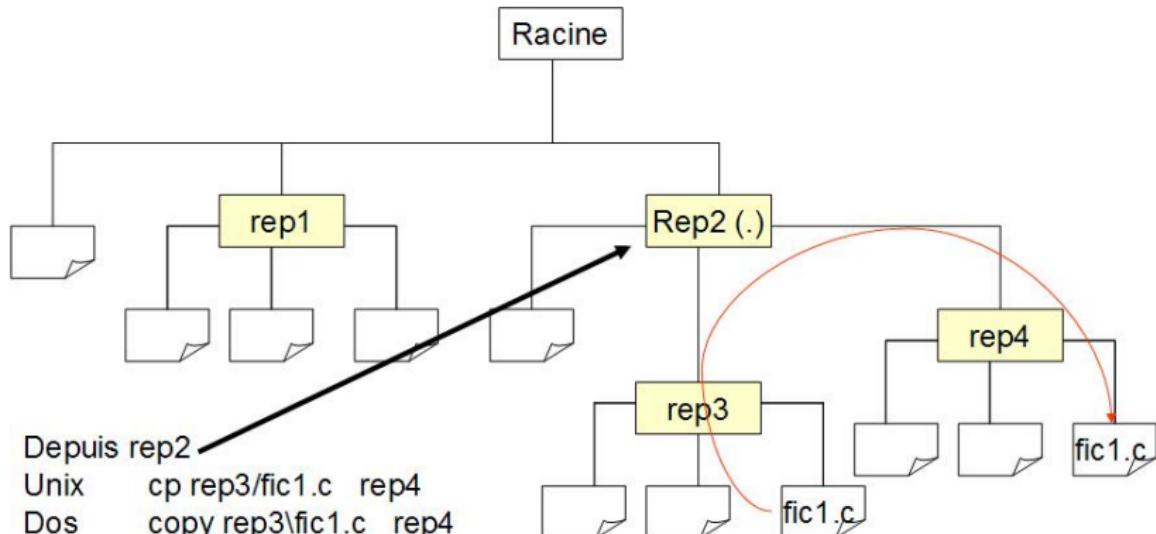
Le répertoire

Structure arborescente



Le répertoire

Structure arborescente



Unix cp /rep2/rep3/fic1.c /rep2/rep4

Le répertoire

- Déplacement dans l'arborescence

- Unix :

- cd <rep/.../repn> descente dans le répertoire <repn>
 - cd .. retour au répertoire parent
 - cd / retour au répertoire racine
 - cd retour au répertoire privé
 - cd - retour au répertoire précédent

- DOS :

- cd <rep\...\repn> descente dans le répertoire <repn>
 - cd .. retour au répertoire parent
 - cd \ retour au répertoire racine

- Quel est le répertoire courant ?

- Unix : pwd
 - DOS : cd

Le répertoire

Lister le contenu d'un répertoire :

- Unix :

- ls affiche les noms seuls
- ls -l avec les détails (date, taille, protection, etc)
- ls -a avec les fichiers cachés (commençant par .)
- ls -lt avec détails et chronologiquement
- ls -laR récursif
- de nombreux autres possibilités (faire man ls)

- DOS :

- dir affiche les noms avec les détails (date, taille, ...)
- dir /w affiche les noms seuls
- dir /od avec détails et chronologiquement à leur création
- dir /s récursif
- dir /? pour connaître les autres options

Le répertoire

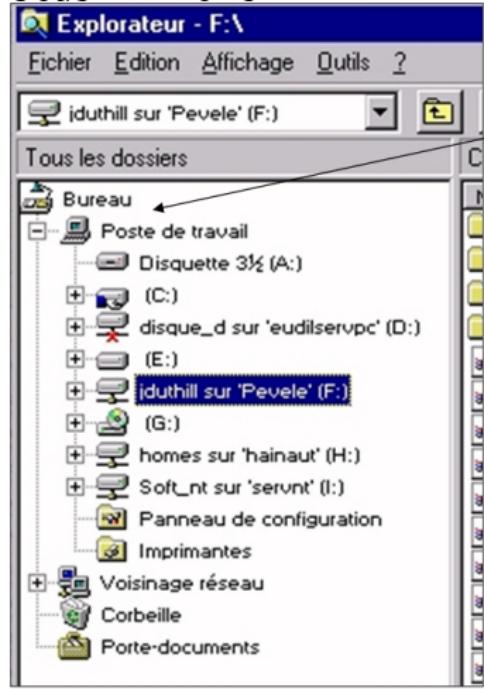
Sous Unix

- pas de notion de disque
- répertoire unique partant de la racine, nommé "/"
- commande df

```
jduthill@hainaut:~ >df -k
Filesystem      1k-blocks   Used   Available  Use% Mounted on
/dev/sda4        792800    533356    218479  71% /
/dev/sda5        497667     8850    463115   2% /tmp
/dev/sda6        792800    42958    708878   6% /var
/dev/sdb1        8634569   7041939   1144820  86% /home
/dev/sda7       1981000   227144   1651444  12% /usr/local
```

Le répertoire

Sous Windows

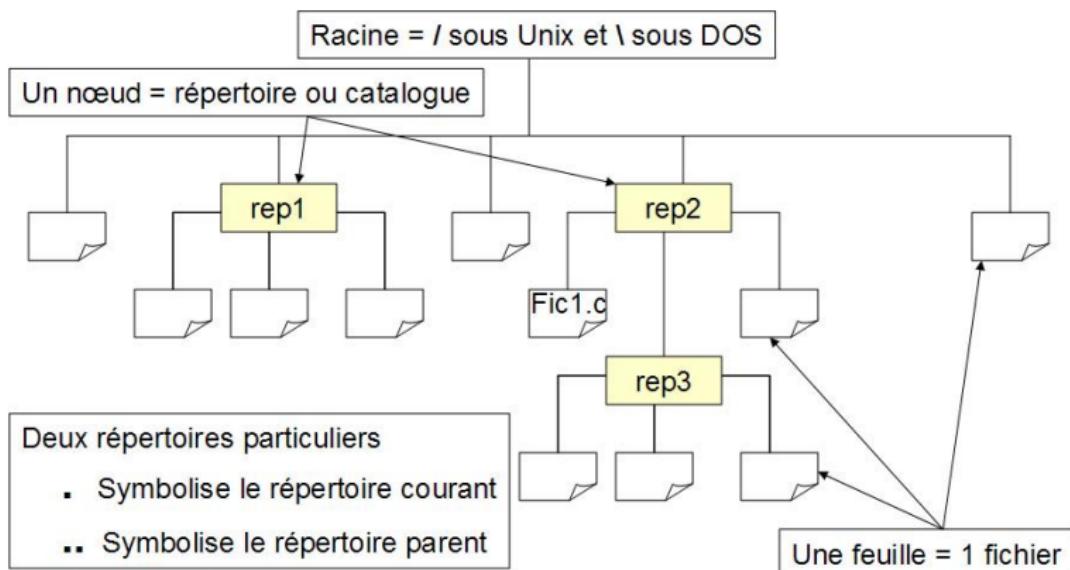


Apparence d 'arborescence alors qu 'il s 'agit de disques différents

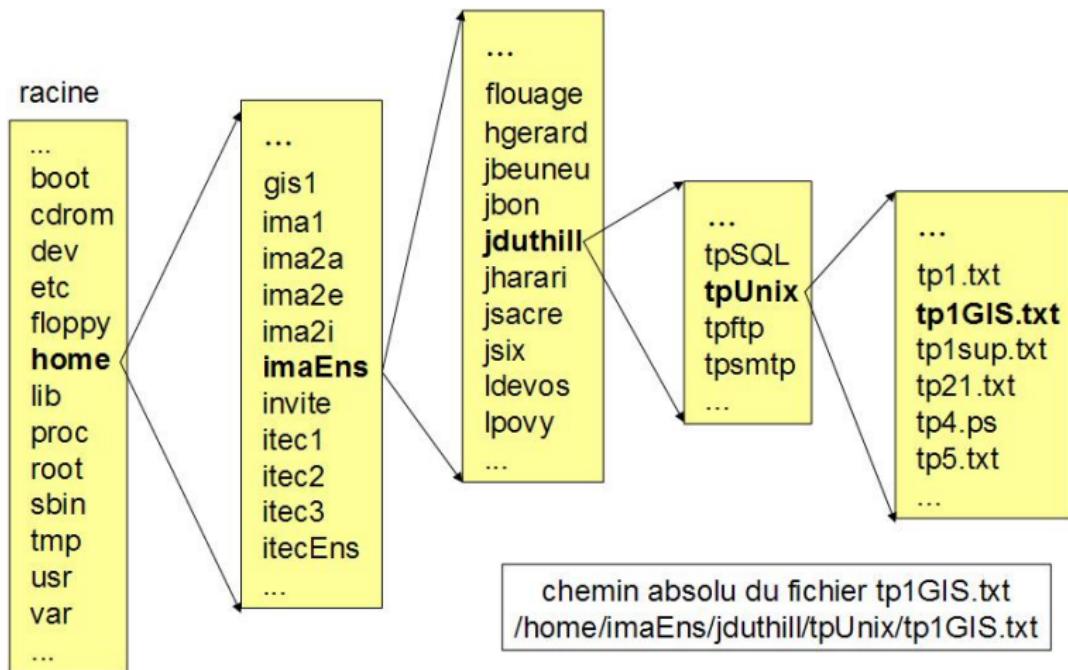


Structure des répertoires

Structure arborescente



Exemple



Le répertoire

- Référence absolue d'un fichier
 - Nom "complet" du fichier = chemin d'accès depuis la racine.
 - Ex sous Unix : /rep2/fic1.c
- Répertoire de travail ou courant
 - Répertoire sur lequel un utilisateur est positionné à un moment donné (nommé .)
- Référence relative d'un fichier
 - Référence d'un fichier à partir du répertoire de travail
- Répertoire privé (sous Unix)
 - Le répertoire sur lequel est positionné un utilisateur suite au login

UNIX : type de fichiers

- - : ordinaire (texte, binaire, exécutable, ...)
- d : répertoire (*directory*)
- c : fichier périphérique (*device*) de type caractère
- b : fichier périphérique (*device*) de type bloc
- s : socket
- l : lien (*link*)
- p : tube (*pipe*) nommé

UNIX : droits

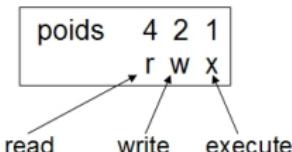
- Un utilisateur est caractérisé par :
 - un numéro de compte uid
 - un numéro de groupe gid

```
jduthill@weppes:/etc >id  
uid=1127(jduthill) gid=1011(imadns) groups=1011(imadns)
```

- Catégories d'utilisateurs :
 - l'utilisateur (*user*)
 - les membres du groupe (*group*)
 - les autres utilisateurs (*others*)
 - tous les utilisateurs (*all*)

UNIX : droits

- Un ensemble de 3 bits pour l'utilisateur, le groupe et les autres



droit	Fichier	Répertoire
r	lecture possible	Liste possible
w	écriture possible	Modification possible
x	exécution possible	Traversée possible

UNIX : droits

- Pour combiner plusieurs droits, il suffit de les additionner

$$0 = 0+0+0 = ---$$

$$1 = 0+0+1 = --x$$

$$2 = 0+2+0 = -w-$$

$$3 = 0+2+1 = -wx$$

$$4 = 4+0+0 = r--$$

$$5 = 4+0+1 = r-x$$

$$6 = 4+2+0 = rw-$$

$$7 = 4+2+1 = rwx$$

UNIX : gestion des permissions

- changement de propriétaire (*change owner*)

```
chown [option] owner[.group] nom_fichier
```

```
pevele:/home/imaEns/jduthill# ls -la cent
-rw-r--r--  1 root      root           4 Jan 20  1999 cent
pevele:/home/imaEns/jduthill# chown jduthill.imaEns cent
pevele:/home/imaEns/jduthill# ls -la cent
-rw-r--r--  1 jduthill  imaEns        4 Jan 20  1999 cent
```

- changement de groupe (*change group*)

```
chgrp [option] group nom_fichier
```

```
pevele:/home/imaEns/jduthill# ls -la cent
-rw-r--r--    1 jduthill imaEns        4 Jan 20  1999 cent
pevele:/home/imaEns/jduthill# chgrp root cent
pevele:/home/imaEns/jduthill# ls -la cent
-rw-r--r--    1 jduthill root        4 Jan 20  1999 cent
```

UNIX : gestion des permissions

- positionnement des droits

```
chmod <permissions> nom_fichier
```

```
pevèle:/home/imaEns/jduthill# chmod 777 cent
pevèle:/home/imaEns/jduthill# ls -la cent
-rwxrwxrwx 1 jduthill imaEns 4 Jan 20 1999 cent
pevèle:/home/imaEns/jduthill# chmod u=rwx,g-x,o-wx cent    OU
pevèle:/home/imaEns/jduthill# chmod 764 cent
pevèle:/home/imaEns/jduthill# ls -la cent
-rwxrw-r-- 1 jduthill imaEns 4 Jan 20 1999 cent
```

- positionnement du masque des droits

```
umask
```

```
jduthill@servnx:~/mail >umask 000;touch fichier;ls -la fichier;rm fichier
-rw-rw-rw- 1 jduthill imaEns 0 Feb 15 16:55 fichier
jduthill@servnx:~/mail >umask 022;touch fichier;ls -la fichier;rm fichier
-rw-r--r-- 1 jduthill imaEns 0 Feb 15 16:55 fichier
jduthill@servnx:~/mail >
```

UNIX : permissions

- 12 bits par groupe de 3
- les 3 premiers bits : 4000 setuid
 - le processus résultant d'un fichier exécutable s'exécute avec les droits du propriétaire du programme

```
pevele:/home/imaEns/jduthill# ls -la cent
-rwxrwxrwx    1 jduthill imaEns      4 Jan 20  1999 cent
pevele:/home/imaEns/jduthill# chmod 4777 cent
pevele:/home/imaEns/jduthill# ls -la cent
-rwsrwxrwx    1 jduthill imaEns      4 Jan 20  1999 cent
pevele:/home/imaEns/jduthill# chmod 777 cent
pevele:/home/imaEns/jduthill# chmod u+s cent
pevele:/home/imaEns/jduthill# ls -la cent
-rwsrwxrwx    1 jduthill imaEns      4 Jan 20  1999 cent
```

Le SUID s'écrit **S** si le propriétaire n'a pas le droit d'exécution, **s** sinon

UNIX : permissions

- 12 bits par groupe de 3
- les 3 premiers bits : 2000 setgid
 - Fichiers exécutables :
le processus résultant d'un fichier exécutables possède les droits du groupe du propriétaire du fichier
 - Répertoires :
les fichiers créés dans le répertoire ont pour groupe le groupe du propriétaire du répertoire

```
pevele:/home/imaEns/jduthill# chmod 6777 cent
pevele:/home/imaEns/jduthill# ls -la cent
-rwsrwsrwx 1 jduthill imaEns 4 Jan 20
pevele:/home/imaEns/jduthill# chmod 777 cent
pevele:/home/imaEns/jduthill# chmod u+s,g+s cent
pevele:/home/imaEns/jduthill# ls -la cent
-rwsrwsrwx 1 jduthill imaEns 4 Jan 20 1999 cent
```

Le SGID s'écrit **S** si
le groupe propriétaire
n'a pas le droit
d'exécution, **s** sinon.

UNIX : permissions

- 12 bits par groupe de 3
- les 3 premiers bits : 1000 sticky bit
 - Fichiers exécutables :
le processus résultant d'un fichier exécutable reste en mémoire et son chargement est rapide
 - Répertoires :
la suppression d'un fichier dans un répertoire n'est possible que par le propriétaire

le Sticky bit s'écrit T si les autres n'ont pas le droit d'exécution, t sinon.

```
pevole:/home/imaEns/jduthill# chmod 777 cent
pevole:/home/imaEns/jduthill# chmod 1777 cent
pevole:/home/imaEns/jduthill# ls -la cent
-rwxrwxrwt    1 jduthill imaEns        4 Jan 20  1999 cent
pevole:/home/imaEns/jduthill# chmod 777 cent
pevole:/home/imaEns/jduthill# chmod o+t cent
pevole:/home/imaEns/jduthill# ls -la cent
-rwxrwxrwt    1 jduthill imaEns        4 Jan 20  1999 cent
```



UNIX : gestion des permissions

- changement de groupe

```
newgrp [-] [group]
```

```
jduthill@gayant08:~ >id  
uid=1127(jduthill) gid=1011(imaEns) groups=1011(imaEns),1059(sinfo)  
jduthill@gayant08:~ >newgrp - sinfo  
jduthill@gayant08:~ >id  
uid=1127(jduthill) gid=1059(sinfo) groups=1011(imaEns),1059(sinfo)  
jduthill@gayant08:~ >exit  
jduthill@gayant08:~ >id  
uid=1127(jduthill) gid=1011(imaEns) groups=1011(imaEns),1059(sinfo)
```

- retour au groupe initial : exit

UNIX : extension ACL

- connaître les droits sur un fichier

```
getfacl [options] fichier
```

```
jduthill@weppes:~ >ls -la ssh
-rw----- 1 jduthill imaEns 0 Feb 21 2004 ssh
jduthill@weppes:~ >getfacl ssh
# file: ssh
# owner: jduthill
# group: imaEns
user::rw-
group::---
other::---
```

UNIX : extension ACL

- donner des droits sur un fichier

```
setfacl [options] [{-m|-x} acl] fichier
```

m pour mettre des droits

x pour supprimer des droits

```
jduthill@weppes:~ >setfacl -m ndevesa:r,ocaron:w ssh
jduthill@weppes:~ >ls -la ssh
-rw-r----+ 1 jduthill imaEns 0 Feb 21 2004 ssh
jduthill@weppes:~ >getfacl ssh
# file: ssh
# owner: jduthill
# group: imaEns
user::rw-
user:ndevesa:r--
user:ocaron:-w-
group::---
mask::rw-
other::---
```

acl

acl_spec
[u[ser]:]uid [:perms]
g[roup]:gid [:perms]

UNIX : liens

- lien dur (*hard link*)

- autre nom pour un fichier
- un lien de ce type peut être détruit sans affecter les autres
- dès qu'un fichier n'a plus de liens, il est détruit

```
jduthill@weppes:~/tp >ls -li essailien
556622253 -rw-r--r--    1 jduthill imaEns      547 Apr  3 15:31 essailien
jduthill@weppes:~/tp >cd
jduthill@weppes:~ >ln tp/essailien ressailien
jduthill@weppes:~ >ln tp/essailien ressailien2
jduthill@weppes:~ >ls -li ressailien*
556622253 -rw-r--r--    3 jduthill imaEns      547 Apr  3 15:31 ressailien
556622253 -rw-r--r--    3 jduthill imaEns      547 Apr  3 15:31 ressailien2
jduthill@weppes:~ >rm tp/essailien
jduthill@weppes:~ >ls -li ressailien*
556622253 -rw-r--r--    2 jduthill imaEns      547 Apr  3 15:31 ressailien
556622253 -rw-r--r--    2 jduthill imaEns      547 Apr  3 15:31 ressailien2
```

UNIX : liens

- lien symbolique (*soft link*)

- c'est un pointeur sur un fichier
- si l'original est détruit, le lien persiste mais ne fonctionne plus
- un lien sur un répertoire doit être un lien symbolique

```
jduthill@weppes:~/tp >ls -li essailien
556621998 -rw-r--r-- 1 jduthill imaEns 547 Apr 3 15:37 essai
jduthill@weppes:~/tp >cd
jduthill@weppes:~ >ln -s tp/essailien ressailien
jduthill@weppes:~ >ls -li ressailien
555788376 lrwxrwxrwx 1 jduthill imaEns 12 Apr 3 15:38 ressai -> tp/essai
jduthill@weppes:~ >rm tp/essailien
jduthill@weppes:~ >cat ressailien
cat: ressailien: No such file or directory
jduthill@weppes:~ >ls -li ressailien
555788376 lrwxrwxrwx 1 jduthill imaEns 12 Apr 3 15:38 ressai -> tp/essai
```

UNIX : principaux répertoires

- / : la racine
- bin : commandes binaires utilisateurs essentielles
- boot : fichiers statiques du chargeur de lancement
- dev : fichiers de périphériques
- etc : configuration système de la machine
- home : répertoires personnels des usagers
- lib : bibliothèques partagées essentielles
- mnt : point de montage temporaire
- proc : système de fichiers virtuels d'information du noyau et des processus
- root : répertoire personnel de root
- sbin : binaires systèmes
- tmp : fichiers temporaires
- usr : deuxième section majeure du système
- var : contient des fichiers de données variables



UNIX : principaux répertoires

/bin :

- les principales commandes du systèmes pour tous les utilisateurs

```
jduthill@pevele:/bin >ls
arch      df          fuser      mkdir     ping6      sh        uname
bash      dir         grep       mknod    ps          sleep     uncompressed
cat       dmesg      gunzip    mktemp   pwd        stty      vdir
chgrp    dnsdomainname gzip      more     rbash     readlink sync
chmod    echo        hostname  mount    rm        tar       zcat
chown   ed          kill      mt      rmdir    tempfile
cp       egrep       ln        mv      run-parts touch
cpio    false       loadkeys  netstat  sed      true
date    fdflush    login     pifof   setserial umount
dd      fgrep       ls        ping
```

commandes de restitution

commandes réseau

UNIX : principaux répertoires

/boot :

- contient tout pour le démarrage, sauf les fichiers de configuration

/dev :

- contient tous les fichiers spéciaux, liens entre le logiciel et les périphériques.
- exemples :
 - fd* : floppy disk
 - sd* : SCSI disk
 - tty* : terminaux
 - hd* : hard disk
 - mt* : magnetic disk



Shell UNIX

Unix ABC

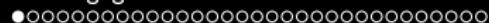
A is for awk, which runs like a snail, and
B is for biff, which reads all your mail.
C is for cc, as hackers recall, while
D is for dd, the command that does all.
E is for emacs, which rebinds your keys, and
F is for fsck, which rebuilds your trees.
G is for grep, a clever detective, while
H is for halt, which may seem defective.
I is for indent, which rarely amuses, and
J is for join, which nobody uses.
K is for kill, which makes you the boss, while
L is for lex, which is missing from DOS.
M is for more, from which less was begot, and
N is for nice, which it really is not.
O is for od, which prints out things nice, while
P is for passwd, which reads in strings twice.
Q is for quota, a Berkeley-type fable, and
R is for ranlib, for sorting ar table.
S is for spell, which attempts to belittle, while
T is for true, which does very little.
U is for uniq, which is used after sort, and
V is for vi, which is hard to abort.
W is for whoami, which tells you your name, while
X is, well, X, of dubious fame.
Y is for yes, which makes an impression, and
Z is for zcat, which handles compression.

Philosophie Unix

- ➊ Small is beautiful
- ➋ Make each program do one thing well
- ➌ Build a prototype as soon as possible
- ➍ Choose portability over efficiency
- ➎ Store data in flat ASCII files
- ➏ Use shell scripts to increase portability
- ➐ Avoid captive user interfaces
- ➑ If possible, make a program a filter

source :

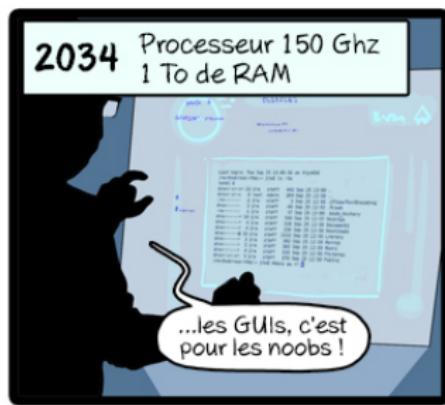
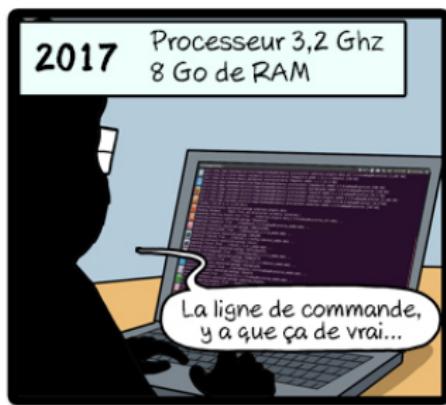
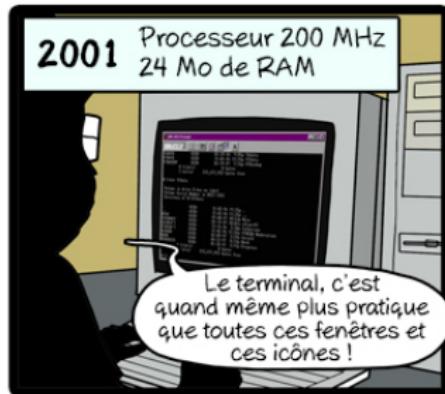
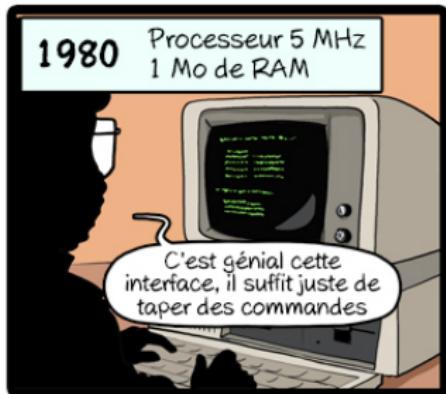
<http://www.math.harvard.edu/computing/unix/frame.html>

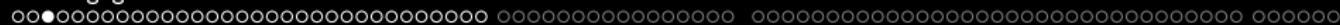


Les langages de commande

Interface entre l'utilisateur et le système

- exécution de programmes et de commandes
- contrôle de l'environnement
- redirection des entrées/sorties
- substitution des variables
- traitement des caractères spéciaux
- fournit un langage de programmation





Choix d'un langage de commande

- sous UNIX (Bourne Shell et dérivés)
 - sh : bourne shell (shell original)
 - bash : bourne again shell
 - ksh : korn shell
 - zsh : Z shell
- sous UNIX (C Shell et dérivés)
 - csh : C shell, développé par Berkeley
 - tcsh : C shell amélioré
- sous DOS
 - command.com

Format d'une commande

Format général d'une commande (UNIX)

Invite du shell (prompt)

\$commande [-options] [arg1 arg2 arg3 ...]

return

arguments séparés
par un espace

option précédée de -

nom de la commande

Commandes de base

Les caractères génériques ou jokers

- ? caractère quelconque (UNIX & DOS)
- * 0, un ou plusieurs caractères (UNIX & DOS)
- [] caractère parmi un ensemble (UNIX)
- [^] caractère en dehors d'un ensemble (UNIX)
- ~ désigne le répertoire home de l'utilisateur

Gestion de ces jokers

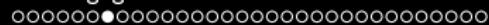
- UNIX : expansion avant l'exécution (le processus fils du shell reçoit une commande préalablement traitée par le shell)

Commandes de base

Gestion de ces jokers : Astuce !

- Unix : expansion avant l'exécution (le processus fils du shell reçoit une commande préalablement traitée par le shell)

```
jduthill@septiles:~/sgbd05/public_html >ls *.php
etudiant.php insertvoeu.php stage.php voeux.php
jduthill@septiles:~/sgbd05/public_html >strace ls *.php 2>&1
| grep execve
execve("/bin/ls", ["ls", "etudiant.php", "insertvoeu.php",
"stage.php", "voeux.php"], /* 30 vars */) = 0
jduthill@septiles:~/sgbd05/public_html >
```



Commandes de base

- affichage de la date et heure
 - UNIX : date
 - DOS : date et time
- aide en ligne
 - UNIX : man [section] <commande>
 - Section 1 : commandes utilisateur
 - Section 2 : appels système
 - Section 3 : fonctions diverses
 - Section 4 : format de fichiers
 - Section 5 : divers
 - UNIX : man -k <mot>
 - DOS : <commande> /?
- effacement de l'écran
 - UNIX : clear
 - DOS : cls

Commandes de base

- afficher le contenu d'un fichier
 - UNIX : cat <fich> ou more <fich>
 - DOS : type <fich> ou more <fich>
- afficher le contenu de plusieurs fichiers
 - UNIX : cat <fich1> ... <fichn>
 - DOS : type <fich1> ... <fichn>
 - DOS : copy <fich1>+...+<fichn> CON:
- destruction de fichiers
 - UNIX : rm [-if] <fich1> ... <fichn>
 - -i : confirmation avant destruction
 - -f : force la destruction des fichiers sans autorisation d'écriture
 - DOS : del <fich> [/p]
 - /p : confirmation avant destruction

Commandes de base

Destruction de fichiers (exemple)

- UNIX : **rm [-if] <fich1> ... <fichn>**

```
p166-jm-~>ls -la a*
-rw-r--r-- 1 jm      users          822 Feb 27 1999 adrr
-rw-r--r-- 1 jm      users         960 Feb 19 1999 appeloffre
-rw-r--r-- 1 jm      users        1354 Jan 28 1999 archiv-ad
-rrr--r-- 1 jm      users        431 Dec 19 1998 atos

p166-jm-~>rm -i adrr
rm: détruire `adrr'? n

p166-jm-~>rm -if adrr
p166-jm-~>rm -i atos
rm: détruire `atos', en outrepassant le mode 0444? n

p166-jm-~>rm -if atos
p166-jm-~>
```

Commandes de base

- renommer un fichier
 - UNIX : `mv [-if] <fich1> <fich2>`
 - -i : confirmation avant renommage
 - -f : force le renommage
 - DOS : `rename <fich1> <fich2>`
- création de répertoire(s)
 - UNIX : `mkdir [-p] <rep>[/<rep>]`
 - -p : crée les répertoires parents si besoin
 - DOS : `mkdir <rep>[\ <rep>]`
- renommer un répertoire
 - UNIX : `mv [-if] <rep1> <rep2>`
 - DOS : `move <rep1> <rep2>`

Commandes de base

Renommer un ensemble de fichiers

- UNIX : mv [-if] *.<exten1> <*.exten2> ???

```
-rw-r--r--    1 jeanmich jeanmich    14779 fév  3 14:16 td1_ps.eps
-rw-r--r--    1 jeanmich jeanmich    11313 fév  3 14:16 td2_ps.eps
-rw-r--r--    1 jeanmich jeanmich    12594 fév  3 14:16 td3_ps.eps
-rw-r--r--    1 jeanmich jeanmich    12273 fév  3 14:16 td4_ps.eps
-rw-r--r--    1 jeanmich jeanmich   279964 fév  3 14:17 unix_ps.eps
[jeanmiche@p166bis jeanmiche]$ mv      *.eps      *.tru
mv: Lors du déplacement de fichiers, le dernier paramètre doit être
un répertoire. Pour en savoir davantage, faites: `mv --help'.
[jeanmiche@p166bis jeanmiche]$
```

- Voir ce qui a été dit à propos de l'expansion des caractères génériques

Commandes de base

- destruction de répertoires
 - UNIX : rmdir [-if] -r <rep1> ... <repn>
 - UNIX : rm -rf <rep1> ... <repn>
 - DOS : del [/s/p] <rep1> ... <repn>
 - /s : dans tous les répertoires
 - /p : confirmation
- copie de plusieurs fichiers
 - UNIX : cp [-if] <fich1> ... <fichn> <rep>
 - DOS : copy <fich> <rep>
- copie récursive de répertoire
 - UNIX : cp [-if] -r <rep1> ... <repn> <rep>
 - DOS : xcopy [/p] /e <rep1> <rep2>
 - /e : copie répertoire et sous répertoires

Commandes de base

Déplacement de fichiers

- UNIX : `mv [-fiu] <fic1> ... <fic2>`
 - `-u (update)` : seulement si le fichier source est plus récent que la destination
- DOS : `move [/Y/-Y] <fic1> ... <fic2>`
 - `/Y` : supprime la demande de confirmation
 - `/-Y` : impose la confirmation

```
jduthill@douaisis:~$ ls -la
-rw-r--r-- 1 jduthill imaEns 214 Feb 1 10:29 address
drwxr-xr-x 2 jduthill imaEns 4096 Feb 1 10:26 essai
jduthill@douaisis:~$ mv -i address essai/
mv: overwrite `essai/address'? n
jduthill@douaisis:~$ ls -la essai/address
-rw-r--r-- 1 jduthill imaEns 214 Feb 1 10:27 essai/address
jduthill@douaisis:~$ mv -iu address essai/
mv: overwrite `essai/address'? y
jduthill@douaisis:~$ ls -la essai/address
-rw-r--r-- 1 jduthill imaEns 214 Feb 1 10:29 essai/address
jduthill@douaisis:~$
```

Commandes de base

- remplissage d'un fichier depuis le clavier

- UNIX : `cat > <fich>`
`<texte frappé au clavier>`
 + D

- DOS : `copy con: <fich>`
`<texte frappé au clavier>`
 + Z

- tri

- UNIX : `sort <fich>` (de nombreuses possibilités)
 - DOS : `sort <fich>` (quelques possibilités)

Commandes de base

- comparaison de deux fichiers
 - UNIX : diff [-i] <fich1> <fich2>
 - DOS : fc [/c] <fich1> <fich2>
 - /c : ignore la casse
- recherche d'un fichier
 - UNIX : find . -name <fich> -print (voir plus loin)
 - DOS : dir [/s] <fich>
 - /s : récursif
- affichage d'une chaîne
 - UNIX : echo chaîne
 - DOS : echo [on|off] chaîne

Les variables des Shells

- affectation
 - UNIX (Bourne Shell) : `<var>=<valeur>`
 - UNIX (C Shell) : `set <var>=<valeur>`
 - DOS : `set <var>=<valeur>`
- désigner la valeur d'une variable
 - UNIX : `${<var>}`
 - DOS : `%<var>%`
- affecter une variable d'environnement
 - UNIX (Bourne Shell) : `export <var>=<valeur>`
 - UNIX (C Shell) : `setenv <var>=<valeur>`
 - DOS : `set <var>=<valeur>`

Les variables des Shells

Quelques variables d'environnement

- UNIX

- HOME répertoire racine de l'utilisateur
- SHELL nom du shell utilisé par l'utilisateur
- PATH répertoire des commandes
- UID, USER identité de l'utilisateur
- MANPATH chemin pour trouver les pages du manuel
- PS1, PS2 invites utilisées par le shell
- ...

- DOS

- COMSPEC chemin d'accès à l'interpréteur de commandes
- PATH chemin de recherche des commandes
- PROMPT invite

Les variables des Shells

Exécution d'une commande

- UNIX : l'exécution d'une commande s'effectue en recherchant dans les répertoires se trouvant dans la variable PATH. Si la commande est trouvée, elle est exécutée.
Si la commande est dans un répertoire non précisé dans la variable PATH, on peut alors l'exécuter en donnant explicitement le chemin où elle se trouve.
Ex : /chemin/commande ou ./commande
- DOS : La recherche est d'abord effectuée dans le répertoire courant (.), puis dans les répertoires spécifiés dans la variable PATH.
Si la commande est dans un répertoire non précisé dans la variable PATH, on peut alors l'exécuter en donnant explicitement le chemin où elle se trouve.
Ex : \chemin\commande ou .\commande

Redirection des flux

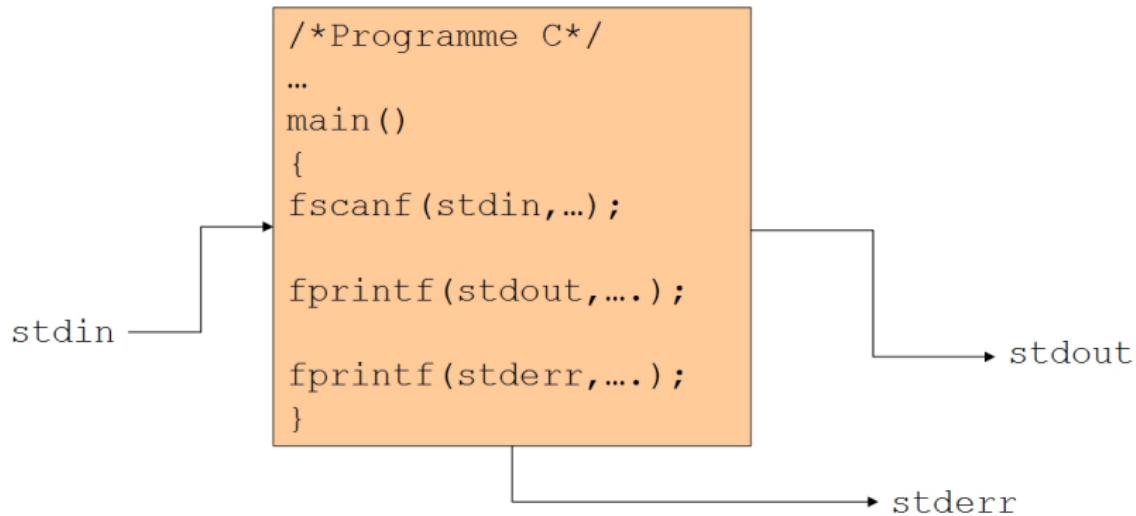
- sous UNIX un processus est créé avec 3 flux
 - entrée standard (stdin ou 0)(*)¹
 - sortie standard (stdout ou 1)(*)
 - sortie erreur (stderr ou 2)(*)
- manipulation de ces flux

sh	csh	effet
<	<	redirige stdin(*)
>	>	redirige stdout(*)
2 >	_	redirige stderr(*)
>>	>>	ajoute stdin(*)
2 >>	2 >>	ajoute stderr(*)
> &	> &	redirige stdout et stderr
2 > &1		redirige stderr sur stdout(*)

1. (*) fonctionne aussi sous DOS

Redirection des flux

- Sous UNIX un processus est créé avec 3 flux
- Manipulation de ces flux



Redirection des flux

Transfert de la sortie d'une commande vers l'entrée d'une autre commande

- Utilisation du caractère | (*pipe*)
- UNIX : exemple : ls -la | sort | more
- DOS : exemple : dir | sort | more

le résultat de la commande ls (UNIX) ou dir (DOS) est triée par la commande sort puis affichée page par page par la commande more

Commandes de base

Enchaînement de commandes :

- Commande simple (avant plan)
 - \$> cde [arg ...]
- Commande simple (arrière plan)
 - \$> cde [arg ...] &
- Pipeline (tube de communication)
 - \$> cde1 [arg ...] | cde2 [arg ...]
- Séquence de commandes
 - \$> cde1 [arg ...] ; cde2 [arg ...] ; cde3 [arg ...]; ...
- Exécution par un nouveau shell
 - \$> sh cde [arg ...]
 - \$> (cde1 [arg ...] ; cde2 [arg ...] ; ...); cde3 [arg ...]; ...
- Prolongement de la commande sur plusieurs lignes
 - apparition d'un "prompt" de suite (variable PS2), tant que la commande n'est pas terminée

Commandes de base

Enchaînement de commandes (exemples) :

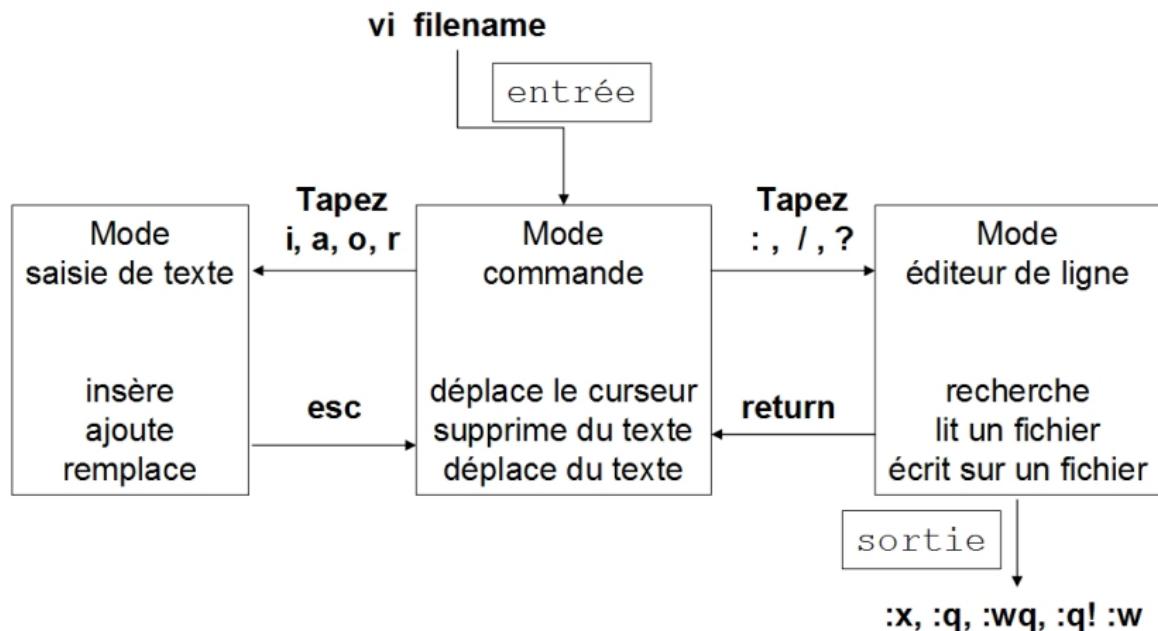
```
jduthill@septiles:~/tpGIS >(cd /$HOME ; pwd) ; pwd
//home/jduthill
/home/jduthill
/home/jduthill/tpGIS
jduthill@septiles:~/tpGIS >
jduthill@septiles:~/tpGIS >ls -la $HOME/tpc |
suite>sort |
suite>more
-rw-r--r--    1 jduthill jduthill      280 Dec 18 16:02 exo2.c
-rw-r--r--    1 jduthill jduthill      397 Dec 20 13:59 exo3.c
-rwxr-xr-x    1 jduthill jduthill    5152 Dec 18 16:02 exo2
-rwxr-xr-x    1 jduthill jduthill    5348 Dec 20 13:59 exo3
drwxr-sr-x   2 jduthill jduthill    4096 Dec 20 13:59 .
drwxr-sr-x  31 jduthill jduthill    4096 Feb  6 14:04 ..
total 32
jduthill@septiles:~/tpGIS >
```

Éditeur vi

Éditeur vi

- disponible sur tout UNIX (et sur Windows)
- deux modes : insertion et commandes
- appel de l'éditeur : vi <nomfic> <nomfic2> ...
- mode commandes au démarrage
- passage en mode insertion :
 - a (*after*) insertion après la position courante
 - A (*append*) insertion à la fin de la ligne
 - i (*insert before*) insertion avant la position courante
 - I (*insert before*) insertion avant le 1^{er} caractère non blanc
- revenir en mode commandes : <esc>

Éditeur vi





CommitStrip.com

Éditeur vi

Sortie de l'éditeur (doit être en mode commandes)

- :w (*write*) enregistre le fichier sans quitter vi
- :w <fic> (*write*) enregistre le fichier sous le nom <fic> sans quitter vi
- :q (*quit*) quitte l'éditeur si le fichier a été renregistré ou n'a pas été modifié
- :wq (*write and quit*) enregistre le fichier et quitte l'éditeur.
- :q! (*quit*) quitte l'éditeur sans tenir compte des modifications effectuées

Éditeur vi

Modification du texte (mode commandes)

- **r** (*replace*) remplace le caractère courant
- **x** (*exchange*) supprime le caractère courant
- **J** (*joint*) joint la ligne suivante à la ligne courante
- **u** (*undo*) annule la dernière opération
- **cw** (*change word*) insère en lieu et place du mot courant
- **~** bascule le caractère courant en majuscule/minuscule

Suppression de lignes (mode commandes)

- **ndd** supprime n lignes. Si n est omis, supprime la ligne courante

Éditeur vi

Recherche et remplacement (mode commandes)

- /<chaîne> recherche la chaîne
- n recherche la chaîne suivante
- :n,m s/<chaîne1>/<chaîne2>/options recherche la chaîne <chaîne1> de la ligne n à la ligne m et remplace par <chaîne2>. \$ signifie dernière ligne. options peut être :
 - g pour global, change toutes les occurrences d'une même ligne
 - c pour confirmer

Vim Visual Cheat Sheet

Movement/Range

- Character: h, j, k, l, Ctrl-B, Ctrl-F
- Line: 0, \$, begin/end of line
- Paragraph, Block: {, }, [,], %
- Window, File: ^W, ^W p, ^W w, ^W b, ^W f, ^W k, ^W zt, ^W zz, ^W G, ^W G

General Commands

- ESC: enter normal mode
- v: enter visual mode
- V: enter visual line mode
- C-v**: enter visual block mode
- i: enter insert mode
- R: enter replace mode
- a: append at end of line
- y: yank/copy (range)
- d: delete/cut (range)
- c: modify (range)
- x: delete/cut (character)
- D: delete to end of line
- P: modify to end of line
- p: paste after cursor
- J: join lines
- r: replace (character)
- >: indent
- <: indent leftward
- .: redo
- u: undo

EX Commands

- :w: save(:wq) save and quit)
- :q: quit(:q! quit anyway)
- :e x: edit file x
- :n: new window
- :h: vim help
- :xx: jump to line #xx

Auto-completion [insert mode]

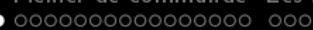
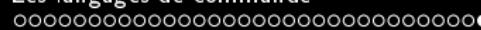
- C-N: auto-complete next/prev keyword
- C-X C-F: auto-complete file name

Search

- */?/: find current word backward/forward
- f/x: to character x to right
- g/d: to definition of current word
- /xxx: search xxx
- n/N: next/prev search result

Split windows

- VS: vertically/horizontally split
- SP: split and diff
- ^W p: to last accessed window
- ^W w: to next window



Les scripts

Fichier de commande

- Sous UNIX :

- n'importe quel fichier contenant des commandes UNIX. Par simple convention .sh
- indiquer sur la première ligne le shell utilisé :
#!/bin/sh ou #!/bin/csh
- **ATTENTION** : nécessite l'autorisation d'exécution (bit x)

```
p166-jm-> ls -la esscsh
-rw-r--r-- 1 jm      users
p166-jm-> chmod 744 esscsh
p166-jm-> ls -la esscsh
-rwxr--r-- 1 jm      users
24 Jan 29 16:25 esscsh
```

- Sous DOS :

- le fichier doit contenir des commandes DOS et avoir le suffixe .BAT (ex : autoexec.bat)

Fichier de commande

Un fichier de commande (script shell) subit les mêmes règles de programmation que n'importe quel langage (commentaires, indentation, déclaration de constantes) et doit être lisible.

Fichier de commande

Sous UNIX :

```
jduthill@septiles://home/jduthill >cdbin
bash: cdbin: command not found
```

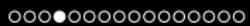
```
#!/bin/bash
# fichier cdbin
pwd
cd tpc
pwd
```

```
jduthill@septiles://home/jduthill > ./cdbin
//home/jduthill
//home/jduthill/tpc
jduthill@septiles://home/jduthill >
```

Le shellscript s'exécute dans son propre environnement. Pas d'incidence sur l'environnement de travail.

```
jduthill@septiles://home/jduthill > . ./cdbin
//home/jduthill
//home/jduthill/tpc
jduthill@septiles://home/jduthill/tpc >
```

Le shellscript est exécuté directement dans le shell de l'environnement de travail.



Fichier de commande

Passage de paramètres

- Sous UNIX

Exemple : commande paramètre1 paramètre2 paramètre3

\$* donne la liste des paramètres (\$argv en csh)

\$# donne le nombre de paramètres (\$#argv en csh)

\$0 donne le nom de la commande (\$argv[0] en csh)

\$1 donne le premier paramètre, \$2, \$3 ... (\$argv[n] en csh)

\$? donne la valeur renvoyée par le programme

- Sous DOS

Exemple : commande paramètre1 paramètre2 paramètre3

%1 donne le premier paramètre, %2, %3, ...

Fichier de commande

Caractères spéciaux (UNIX)

- # introduit un commentaire
- \ préserve la valeur du caractère suivant
- , , délimitent une chaîne de caractères.
Les variables qu'elle contient ne sont pas évaluées
- " " délimitent une chaîne de caractères.
Les variables qu'elle contient sont évaluées
- ` ` ce qui est entre les deux accents graves est interprété comme une commande
- ; sépare les instructions successives d'une ligne de commande

Fichier de commande

Caractères spéciaux (UNIX), exemple avec " et '

```
p166-jm-> a=essai  
p166-jm-> b="Ceci est un $a"  
p166-jm-> echo $b  
Ceci est un essai  
p166-jm->  
p166-jm->  
p166-jm-> c='Ceci est un $a'  
p166-jm-> echo $c  
Ceci est un $a
```

dans une chaîne entre " ", ce qui apparaît sous la forme de \$nom est évalué

dans une chaîne entre ' ' ce qui apparaît sous la forme \$nom n'est pas évalué

```
p166-jm->echo "la date du jour est date"  
la date du jour est date  
p166-jm->echo "la date du jour est `date`"  
la date du jour est Sun Feb 6 17:50:34 CET 2000  
p166-jm->
```

Ici la commande date est interprétée

test

`test <expression>` ou `[&<expression>]`

- l'expression est évaluée et si la valeur est vraie alors le code de retour de la commande est nul
- sur les chaînes de caractères
 - `test -z <chaîne>` est vraie si la chaîne est vide
 - `test -n <chaîne>` est vraie si la chaîne n'est pas vide
 - `test <chaîne1> = <chaîne2>` (`!=` différent)



test

- sur les chaînes numériques (c'est une suite de chiffre précédée ou non de + ou -)

```
test <chaîne1> -eq <chaîne2>
```

-eq	pour equal	-le	pour less or equal
-ne	pour not equal	-gt	pour greater than
-lt	pour less than	-ge	pour greater or equal

- sur les fichiers

- `test -f <fichier>` est vrai ssi `fichier` est un fichier ordinaire
- `test -d <repert>` est vrai ssi `repert` est un répertoire

- opérateurs sur les expressions

- négation : `!`
- conjonction : `-a`
- disjonction : `-o`

Fichier de commande

Instruction conditionnelle (sh)

```
if <liste de commandes>
    then <liste de commandes>
    else <liste de commandes>
fi
```

cascade de if

```
if <liste de commandes>
    then <liste de commandes>
else if <liste de commandes>
    then ....
else if ....
    fi
fi
fi
```

```
#!/bin/sh
if test -f $1
    then echo $1 existe
    else echo $1 n'existe pas
fi
ou
if [ -f $1 ]
    then echo $1 existe
    else echo $1 n'existe pas
fi
```

Fichier de commande

Aiguillage (sh)

```
case <chaine> in
    <motif>) <liste de commandes>;;
    <motif>) <liste de commandes>;;
    .....
    .....
esac
```

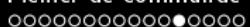
```
#!/bin/sh
case $# in
    0) pwd;;
    1) if [ -f $1 ]
        then echo $1 existe
        fi;;
    2) cp $1 $2;;
    *) echo trop de paramètres;;
esac
```

Fichier de commande

Itération bornée (sh)

```
for <variable> in <chaine>
    do <liste de commandes>
done
```

```
#!/bin/sh
for f in $* ; do
    if [ -f $f ]
        then
            echo $f existe
        else echo $f n'existe pas
    fi
done
#
# autre exemple
#
for i in `ls -a` ;do echo $i ;done
```



Fichier de commande

Boucle avec Bash version 3.0+

```
1 #!/bin/bash
2
3 echo "Bash version ${BASH_VERSION}..."
4 for i in {0..10..2}
5 do
6     echo "Welcome $i times"
7 done
```

Fichier de commande

Itération non bornée (sh)

```
while <liste de commandes>
    do <liste de commandes>
done
```

```
until <liste de commandes>
    do <liste de commandes>
done
```

```
#!/bin/sh
echo -n 'répondre par oui ou par non : '
read reponse
while [ "$reponse" != oui -a "$reponse" != non ]
    do echo -n 'il faut répondre par oui ou par non : '
        read reponse
done
```

Fichier de commande

Définition d'abréviation : alias

- (csh) alias <nom_alias> <commande>
 - possibilité de définir des alias avec paramètres. Un paramètre est désigné par son numéro précédé de la chaîne \!: ainsi \!:2 désignera le 2^{eme} paramètre.
 - \!\$ désigne le dernier paramètre
 - \!^ désigne le premier paramètre
 - \!* donne la liste des paramètres
- (sh) alias <nom_alias> <commande>
 - n'admet pas de paramètres
- alias sans paramètre donne la liste des alias

```
alias dir='ls -la|more'
alias dirlt='ls -lat|more'
alias rm='rm -i'
alias cp='cp -i'
alias mv='mv -i'
```

Fichier de commande

Suppression d'abréviation : unalias

- (sh) **unalias [-a] <nom_alias>**
 - option a, supprime TOUS les alias

```
jduthill@septiles:~ >alias dirr='ls -la'  
jduthill@septiles:~ >alias  
alias dirr='ls -la'  
jduthill@septiles:~ >unalias dirr  
jduthill@septiles:~ >alias  
jduthill@septiles:~ >
```

Fichier de commande

Définition d'abréviation : fonction

- (sh) on peut écrire des fonctions. Les \$1, \$2, ... sont remplacés par les paramètres d'appel de la fonction.

nom-fonction ()

{

.....

.....

}

<--- attention doit être en début de ligne

```
lt()
{
ls -lt $1 | more
}
```

```
lt() { ls -lt $1 | more; }
```

Les filtres

```
oooooooooooooooooooooooooooo Fichier de commande Les filtres oooooooooooooo cmd et f  
oooooooooooooo ooooooooooooooo ooooooooooooooo ooooooooooooooo ooooooooooooooo ooooooooooooooo
```

cut

- permet de découper les lignes d'un fichier en champ et d'extraire des champs particuliers

```
cut -f<liste> [-d<caractère>]
```

```
jduthill@servnx:~ >jour=`date`  
jduthill@servnx:~ >echo $jour  
Mon Feb 7 09:57:24 MET 2000  
jduthill@servnx:~ >day=`echo $jour | cut -f1 -d\ `  
jduthill@servnx:~ >echo $day  
Mon  
jduthill@servnx:~ >day=`date | cut -f2 -d\ `  
jduthill@servnx:~ >echo $day  
Feb
```

WC

```
wc [options] [<fichier> ...]
```

- compte le nombre de lignes, de mots et de caractères des fichiers

option:

- l nombre de lignes
- w nombre de mots
- c nombre de caractères

```
jduthill@servnx:~ >wc /etc/passwd  
      1979    2180  132805 /etc/passwd  
jduthill@servnx:~ >wc -l /etc/pas*  
      1979 /etc/passwd  
      199 /etc/passwd.local  
     2178 total
```

grep ou egrep

egrep [options] <expression> [<fichier>]

- permet de sélectionner des lignes de fichier contenant un motif
- egrep présente l'avantage de permettre d'utiliser des expressions régulières plus détaillées (egrep "none|aborted")

Quelques options (voir man pour d'autres options) :

- **-i** pas de différence entre majuscule et minuscule
- **-n** affiche le numéro de la ligne
- **-c** compte le nombre de lignes
- **-v** inverse le résultat

```
jduthill@douaisis:~ >egrep duthil /etc/passwd
jduthill:x:1127:1011:Jean-Michel Duthilleul:/home/imaEns/jduthill:/bin/bash
rduthill:*:1475:1136:Richard Duthilleul:/home/sm2004/rduthill:xbinxbash
jduthill@douaisis:~$ egrep -c duthil /etc/passwd
2
jduthill@douaisis:~$ egrep -cvD DUTHILL /etc/passwd
4204
jduthill@douaisis:~$
```

tail

- donne les dernières lignes d'un fichier

tail [+|-<nombre>] [-f] [fichier]

nombre de lignes depuis le début du fichier
nombre de lignes depuis la fin du fichier (par défaut 10)

!!! l'option -f permet d'attendre la suite du "remplissage" du fichier

```
jduthill@servnx:~ >tail -5 /etc/passwd
dcazin:x:2774:1058:David.Cazin:/home/deaiaa/dcazin:xbinxbash
jsaab:x:2775:1058:Joseph.Saab:/home/deaiaa/jsaab:xbinxbash
bsaidane:x:2776:1058:Brahim.Saidane:/home/deaiaa/bsaidane:xbinxbash
dsaihi:x:2777:1058:Dhouha.Saihi:/home/deaiaa/dsaihi:xbinxbash
tvanbell:x:2778:1058:Tony.Vanbelle:/home/deaiaa/tvanbell:xbinxbash
```

head

```
head [-number | -n number] [filename...]
```

- donne les n premières lignes du fichier. Par défaut, n = 10

```
jduthill@servnx:~ >head -7 /etc/passwd
adm:x:4:4:0000-Admin(0000):/var/adm:
bin:x:2:2:0000-Admin(0000):/usr/bin:
daemon:x:1:1:0000-Admin(0000):/:
listen:x:37:4:Network Admin:/usr/net/nls:
lp:x:71:8:0000-lp(0000):/usr/spool/lp:
noaccess:x:60002:60002:uid no access:/:
nobody:x:60001:60001:uid no body:/:
```

```
oooooooooooooooooooooooooooo Fichier de commande Les filtres cmd et f  
oooooooooooooooooooooooooooo oooooooo●oooooooooooooooooooooooooooo
```

uniq

```
uniq [option] [input[output]]
```

- supprime les lignes dupliquées d'un fichier préalablement trié

```
jduthill@pevele:~ >ps aux | tail +2 | cut -f1 -d\ |  
> sort | uniq  
daemon  
irc  
jduthill  
mail  
qf6kpo  
rex  
root  
www-data  
jduthill@pevele:~ >
```

```
oooooooooooooooooooooooooooo Fichier de commande Les filtres cmd et f  
oooooooooooooooooooo●oooooooooooooooooooo
```

tr

```
tr [option] set1 [set2]
```

- transpose ou supprime des caractères

```
jduthill@septiles:~ >a="essai"  
jduthill@septiles:~ >echo "$a: chaine en minuscules" | tr a-z A-Z  
ESSAI: CHAINE EN MINUSCULE  
jduthill@septiles:~ >  
jduthill@septiles:~ >echo "$a: chaine en minuscules" | tr [:lower:] [:upper:]  
ESSAI: CHAINE EN MINUSCULE  
jduthill@septiles:~ >echo "chaine en minuscules" | tr -s "" "" | tr a-z A-Z  
CHAINE EN MINUSCULES  
jduthill@septiles:~ >
```

sort

```
sort [options] [fichier]
```

- sort trie les lignes d'un fichier texte
- principales options :
 - -n numérique
 - -r reverse
 - -t séparateur
 - -k champ

```
septiles:>cat /etc/passwd | grep gis1 | sort -tl -k 2r
hzime:x:5326:1047:Harold Zime:/home/gis1/hzime:/bin/false
lzhao:x:5335:1047:Lei Zhao:/home/gis1/lzhao:/bin/false
nzebboud:x:5574:1047:Nadir Zebboudj:/home/gis1/nzebboud:/bin/false
lyanzeu:x:5312:1047:Lionel Yanzeu:/home/gis1/lyanzeu:/bin/false
yxie:x:5223:1047:Yu Xie:/home/gis1/yxie:/bin/false
jvazel:x:5319:1047:Johann Vazel:/home/gis1/jvazel:/bin/false
lteixeir:x:5562:1047:Laetitia Teixeira:/home/gis1/lteixeir:/bin/false
jsamson:x:1092:1047:Jean-Francois Samson:/home/gis1/jsamson:/bin/false
mraoulx:x:4966:1047:Michael Raoulx:/home/gis1/mraoulx:/bin/false
.../...
```



Filtre

- basename permet d'éliminer le chemin d'accès et le suffixe d'un nom de fichier

```
basename <chemin> [ chaîne ]
```

- dirname permet d'éliminer le dernier composant délimité par un slash du nom_de_fichier

```
dirname <chemin>
```

```
jduthill@septiles:~ >edirname=`find . -name article.php`
jduthill@septiles:~ >echo $edirname
./tmp/article.php
jduthill@septiles:~ >dirname $edirname
./tmp
jduthill@septiles:~ >basename $edirname .php
article
```

join

```
join [options] fichier1 fichier2
```

- affiche sur la sortie standard une ligne pour chaque paire de lignes d'entrée, l'une provenant de fichier1 et l'autre de fichier2, qui disposent de champs de fusion identiques.

```
[jm@p166bis tmp]$ cat passwd2
games:x:12:100:games:/usr/games:
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:
news:x:9:13:news:/var/spool/news:
```

```
[jm@p166bis tmp]$ cat shadow2
ftp:*:11006:0:99999:7:::
games:*:11006:0:99999:7:::
gopher:*:11006:0:99999:7:::
halt:*:11006:0:99999:7:::
mail:*:11006:0:99999:7:::
news:*:11006:0:99999:7:::
```

```
[jm@p166bis tmp]$ join -j2 1 -t: passwd2 shadow2 > result
[jm@p166bis tmp]$ cat result
games:x:12:100:games:/usr/games::*:11006:0:99999:7:::
halt:x:7:0:halt:/sbin:/sbin/halt::*:11006:0:99999:7:::
mail:x:8:12:mail:/var/spool/mail::*:11006:0:99999:7:::
news:x:9:13:news:/var/spool/news::*:11006:0:99999:7:::
```

```
oooooooooooooooooooooooooooo Fichier de commande Les filtres cmd et f  
oooooooooooooooooooo●oooooooooooooooooooo
```

paste

```
paste [options] [fichier...]
```

- **paste** affiche des lignes regroupant les lignes correspondantes de chaque fichier, séparées par des tabulations ou le caractère précisé par l'option "-d", et terminées par un saut de ligne (NewLine)

```
septiles:/etc# cat passwd2
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
```

```
septiles:/etc# cat shadow2
root:$1$CTAt.bgy$Oir:12424:0:99999:7:::
daemon:*:12390:0:99999:7:::
bin:*:12390:0:99999:7:::
sys:*:12390:0:99999:7:::
sync:*:12390:0:99999:7:::
games:*:12390:0:99999:7:::
```

```
septiles:/etc# paste -f -d! passwd2 shadow2
root:x:0:0:root:/root:/bin/bash!root:$1$CTAt.bgy$Oir:12424:0:99999:7:::
daemon:x:1:1:daemon:/usr/sbin:/bin/sh!daemon:*:12390:0:99999:7:::
bin:x:2:2:bin:/bin:/bin/sh!bin:*:12390:0:99999:7:::
sys:x:3:3:sys:/dev:/bin/sh!sys:*:12390:0:99999:7:::
sync:x:4:65534:sync:/bin:/bin/sync!sync:*:12390:0:99999:7:::
```

find

find [chemin] [expression...]

- rechercher des fichiers dans une hiérarchie de répertoires

```
septiles:/etc# find /etc -type d -group root -print 2>/dev/null
```

/etc/mc

/ext/X11

/etc/X11/xkb

/etc/X11/xkb/rules

/etc/X11/xkb/types

/etc/X11/xkb/semantics

.../...

```
septiles:/etc# find . -mtime 0 -type f
```

/etc/mail/sendmail.cf

/etc/mail/aliases

/etc/mail/renaliases

/etc/mail/routes

/etc/r

De très nombreuses possibilités
type = type de fichiers

group = nom du groupe

mtime n = fichier modifi

n fois 24h

avec +n plus grand que n

-n plus petit que n

n exactement n

.../...

find (suite)

```
find [chemin] [expression...] [commande]
```

- rechercher des fichiers dans une hiérarchie de répertoires et exécuter la commande

```
jduthill@douaisis:~$ find /etc -type f -mtime 1 -exec ls -la {} \;
-rw-r--r-- 1 root root 16 Feb 12 10:47 /etc/network/ifstate
-rwxr-xr-x 1 root root 30593 Feb 12 10:39 /etc/init.d/sendmail
-rw-r--r-- 1 root root 374 Feb 12 10:47 /etc/mtab
-rw-r--r-- 1 root root 44 Feb 12 10:45 /etc/adjtime
.../...
```

! espace

```
jduthill@douaisis:~$ find /etc -type f -mtime -1 -exec ls -la {} \;
-r--r---- 1 www-data proxy 41407 Feb 13 12:26 /etc/group.web
-r--r---- 1 www-data proxy 188472 Feb 13 12:26 /etc/passwd.web
-rw-r--r-- 1 root root 8 Feb 13 12:48 /etc/ntp.drift
-rw-r--r-- 1 root root 957 Feb 12 17:41 /etc/vmware/vmnet8/dhcpd/dhcpd.leases
-rw-r--r-- 1 root root 1317 Feb 12 17:11 /etc/vmware/vmnet8/dhcpd/dhcpd.leases~
.../...
```

find (suite)

```
find [chemin] [expression...]
```

- rechercher des fichiers dans une hiérarchie de répertoires et exécuter la commande

```
jduthill@douaisis:~$ ls -la `find /etc -type f -mtime 1`  
-rw-r--r-- 1 root root 16 Feb 12 10:47 /etc/network/ifstate  
-rwxr-xr-x 1 root root 30593 Feb 12 10:39 /etc/init.d/sendmail  
-rw-r--r-- 1 root root 374 Feb 12 10:47 /etc/mtab  
-rw-r--r-- 1 root root 44 Feb 12 10:45 /etc/adjtime  
.../...
```

```
jduthill@douaisis:~$ ls -la `find /etc -type f -mtime -1`  
-r--r----- 1 www-data proxy 41407 Feb 13 12:26 /etc/group.web  
-r--r----- 1 www-data proxy 188472 Feb 13 12:26 /etc/passwd.web  
-rw-r--r-- 1 root root 8 Feb 13 12:48 /etc/ntp.drift  
-rw-r--r-- 1 root root 957 Feb 12 17:41 /etc/vmware/vmnet8/dhcpd/dhcpd.leases  
-rw-r--r-- 1 root root 1317 Feb 12 17:11 /etc/vmware/vmnet8/dhcpd/dhcpd.leases~  
.../...
```

Arithmétique

expr :

- les arguments sont interprétés comme des expressions ;
- les différents termes de l'expression doivent être séparés par des espaces

```
jduthill@servnx:~ >expr 5 + 6  
11  
jduthill@servnx:~ >a=10  
jduthill@servnx:~ >a=`expr $a + 1`  
jduthill@servnx:~ >echo $a  
11
```

Arithmétique

```
eval
jduthill@servnx:~ >a=machin
jduthill@servnx:~ >machin=9
jduthill@servnx:~ >echo $a
machin
jduthill@servnx:~ >echo \$\$a
$machin
jduthill@servnx:~ >echo $machin
9
jduthill@servnx:~ >eval echo \$\$a
9
jduthill@servnx:~ >eval $a=0
jduthill@servnx:~ >echo $a
machin
jduthill@servnx:~ >echo \$\$a
$machin
jduthill@servnx:~ >echo $machin
0
```

Script

Sous UNIX :

jduthill:x:1127:1011:Jean-Michel.Duthilleul:/home/imaEns/jduthill:/bin/bash

```
#!/bin/sh
#
#       Quelques constantes
#
PASSWD=passwd3
GROUPES=" "
TEMPFILE=/tmp/accounting
#
#       mise à zéro des compteurs
#
listgroup=`ls /home`
for group in ${listgroup} ; do
    if [ -d /home/$group ] ; then
        GROUPES=${GROUPES}" "$group
        eval $group=0
    fi
done
```

```
#
#       calculs
#
while read line
do
    group=`echo $line | cut -f6 -d:`
    group=`echo $group | cut -f3 -d/`
    eval value=\$group
    eval $group=`expr $value + 1`
done < ${PASSWD}
#
#       sortie des resultats
#
for group in ${GROUPES} ; do
    eval echo "$group: \$group" \
    >> ${TEMPFILE}
done
```

Script

Code de retour d'une commande :

- toute commande UNIX renvoie un code de retour
- 0 si tout est correct
- entier > 0 sinon
- ce code ne s'affiche pas, il est stocké dans la variable " \$?"

```
jduthill@septiles:~ >ls -la testmail
-rw-r--r-- 1 jduthill imaEns 0 Jan 16 10:00 testmail
jduthill@septiles:~ >echo $?
0
jduthill@septiles:~ >ls -la test
ls: test: No such file or directory
jduthill@septiles:~ >echo $?
1
jduthill@septiles:~ >
```

Script

Lecture d'une variable

- `read <nom_variable>`
- `read <var1> ... <varN> < fichier`

```
jduthill@septiles:~ >./lecture
```

nom du répertoire à lister :**mail**

```
drwx----- 12 jduthill imaEns 4096 Jan 16 09:59 mail
```

c'est bon

```
jduthill@septiles:~ >./lecture
```

nom du répertoire à lister :**machin**

pas de répertoire

```
jduthill@septiles:~ >
```

```
# lecture ligne à ligne d'un fichier
```

```
while read ligne ; do
```

```
    echo $ligne
```

```
done < fichier
```

```
#!/bin/sh
# fichier lecture
echo -n nom du répertoire à lister :
read rep
ls -ld $rep 2> /dev/null
if [ $? -eq 0 ]
    then echo c'est bon
    else echo pas de répertoire
fi
```

Script

- décaler les paramètres de position
 - shift [n] (fonctionne aussi sous MS-DOS)
- sortie d'un shellscript
 - exit [n] (fonctionne aussi sous MS-DOS)

```
jduthill@septiles:~ >./decalage essai test truc
```

```
essai
```

```
test
```

```
jduthill@septiles:~ >echo $?
```

```
2
```

```
jduthill@septiles:~ >
```

```
#!/bin/sh
# fichier decalage test si fichier
nb=0
while test "$1" ; do
    if [ -f $1 ] ; then
        nb=`expr $nb + 1` ; echo $1
    fi
    shift
done
exit $nb
```

Script

Mise au point de programme de commande

```
jduthill@septiles:~ >sh -x decalage chemin cours decalage
+ nb=0
+ test chemin
+ '[' -f chemin ']'
++ expr 0 + 1
+ nb=1
+ shift
+ test cours
+ '[' -f cours ']'
+ shift
+ test decalage
+ '[' -f decalage ']'
++ expr 1 + 1
+ nb=2
+ shift
+ test "
+ exit 2
jduthill@septiles:~ >
```

```
#!/bin/sh
# fichier decalage
nb=0
while test "$1" ; do
    if [ -f $1 ] ; then
        nb=`expr $nb + 1`
    fi
    shift
done
exit $nb
```

Expressions régulières ou expression rationnelles

- permettent de désigner des ensembles de chaînes de caractères, en particulier dans les adressages et les substitutions
- peuvent être utilisées dans la plupart des commandes sous UNIX
- une expression régulière atomique est :
 - un caractère normal
 - un caractère spécial précédé de \
 - le caractère ? représentant un caractère quelconque
 - une chaîne de caractères non vide placée entre []

Expressions régulières

Une expression régulière est :

- une expression régulière atomique
 - une expression régulière atomique suivie du caractère d'itération *
 - la concaténation d'expression régulières
 - une expression atomique suivie de :
 - $\{n\}$ avec $0 \leq n < 256$ pour exactement n occurrences de l'expression
 - $\{n, \}$ pour au moins n occurrences de l'expression
 - $\{m, n\}$ pour un nombre d'occurrences compris entre m et n
 - expression comprise entre $\backslash($ et $\)$ pour délimiter des champs dans une ligne

Expressions régulières

Exemples :

- . caractérise n'importe quel caractère
 - [] un des caractères entre crochets ou si le premier caractère est ^, alors caractérise ceux qui ne correspondent pas avec ceux entre crochet
 - [abc] a, b ou c
 - [a-z] une lettre minuscule
 - [a-d5-8w-z] a, b, c, d, 5, 6, 7, 8, w, x, y, z
 - [^0-9] pas un chiffre
 - [^a-zA-Z] pas une lettre
 - ^ début de ligne
 - \$ fin de ligne, [ab\$] ligne finissant par ab

Expressions régulières

Exemples :

- * signifie de 0 à n fois le caractère qui précède
- a* 0 à n fois a
- aa* au moins un a
- .* n'importe quelle chaîne de caractères (y compris la chaîne vide)
- ^[0-9][0-9]*\$ ligne qui ne contient que des chiffres
- \(\) isoler des sous-chaînes. On peut les réutiliser grâce à \1 \2



Filtre-éditeur sed

- fonctionne en mode batch
- c'est une commande puissante
- peut être considéré comme un filtre et un éditeur
- son utilisation courante :
 - recevoir, en entrée chaque ligne d'un fichier
 - lui faire subir des modifications
 - l'envoyer sur la sortie standard

Filtre-éditeur sed

Exemple 1 (en détail)

```
[user@p166bis user]$ echo Ceci est un poids: 523Kg | sed -e 's/^[^0-9]*([0-9][0-9]*).*$/valeur = \1/'  
valeur = 523  
[user@p166bis user]$
```

substitution

sed -e 's/ ^[^0-9]* ([0-9][0-9]*).*/ valeur = \1 / ,

Depuis le début de la ligne
prendre un caractère qui n'est pas
un chiffre, répété « * » fois

Capturer un chiffre suivi
d'un autre chiffre, répété « * » fois

Filtre-éditeur sed

Exemple 2

```
jduthill@servnx:~ >cat /etc/passwd|grep duthill
duthille:x:13103:131:Jean-Michel Duthilleul:/home1/eudilima/duthille:/bin/csh
jduthill:x:1127:1011:Jean-Michel.Duthilleul:/home/imaEns/jduthill:/bin/bash

jduthill@servnx:~ >cat /etc/passwd|grep duthill|cut -f7 -d:
/bin/csh
/bin/bash

jduthill@servnx:~ >cat /etc/passwd|grep duthill|cut -f7 -d:|sed -e "s///x/g"
xbincsh
xbinxbash

jduthill@servnx:~ >
```

Filtre-éditeur sed

Exemple 3

```
jduthill@hainaut:~ >cat passwd
frozenbe:*:1846:1031:Franck Rosenberg:/home/ima3m/frozenbe:xbinxbash
sweiss:*:1375:1031:Sebastien Weiss:/home/ima3m/sweiss:xbinxbash
calloggi:*:1166:1031:Cyril Alloggia:/home/ima3m/calloggi:xbinxbash
fvictoir:*:2129:1047:Frederic Victoire:/home/gis1/fvictoir:xbinxbash
psitarz:*:2156:1047:Przemyslaw Sitarz:/home/gis1/psitarz:xbinxbash
gtyrou:*:2130:1047:Gregory Tyrou:/home/gis1/gtyrou:xbinxbash
.../...
jduthill@hainaut:~ >cat passwd | sed -e 's/^:(.*:)!:(.*\!)(.*!)!/1\!rep\!poub\!3/'
frozenbe:*:1846:1031:rep/poub/frozenbe:xbinxbash
sweiss:*:1375:1031:rep/poub/sweiss:xbinxbash
calloggi:*:1166:1031:rep/poub/calloggi:xbinxbash
fvictoir:*:2129:1047:rep/poub/fvictoir:xbinxbash
psitarz:*:2156:1047:rep/poub/psitarz:xbinxbash
gtyrou:*:2130:1047:rep/poub/gtyrou:xbinxbash
.../...
jduthill@hainaut:~ >cat passwd
```

Filtre-éditeur sed

Exemple 3 (en détail)

```
jduthill@hainaut:~ >cat passwd  
frozenbe:*:1846:1031:Franck Rosenberg:/home/ima3m/frozenbe:xbinxbash  
jduthill@hainaut:~ >cat passwd | sed -e 's/^(.*:|)(.*\\|)(.*|)/\\1\\rep\\poub\\3/'  
frozenbe:*:1846:1031:Franck Rosenberg:/rep/poub/frozenbe:xbinxbash  
jduthill@hainaut:~ >cat passwd
```

```
sed -e 's/ ^(.*:|)(.*\\|)(.*|) / \\1\\rep\\poub\\3/'
```

(1) Depuis le début de ligne,
prendre tous les caractères
suivis de :

(3) Prendre tous les caractères
qui suivent

(2) Prendre tous les caractères
suivis de /

Filtre-éditeur sed

Exemple 4 : cas du mv "multiple"

```
-rw-r--r-- 1 jeanmich jeanmich 14779 fév 3 14:16 td1_ps.eps
-rw-r--r-- 1 jeanmich jeanmich 11313 fév 3 14:16 td2_ps.eps
-rw-r--r-- 1 jeanmich jeanmich 12594 fév 3 14:16 td3_ps.eps
-rw-r--r-- 1 jeanmich jeanmich 12273 fév 3 14:16 td4_ps.eps
-rw-r--r-- 1 jeanmich jeanmich 279964 fév 3 14:17 unix_ps.eps
```

```
[jeanmiche@p166bis jeanmiche]$ mv *.eps *.tru
```

mv: Lors du déplacement de fichiers, le dernier paramètre doit être un répertoire.
Pour en savoir davantage, faites: `mv --help'.

```
[jeanmiche@p166bis jeanmiche]$ ls -la
```

```
drwxrwxr-x 2 qf6kpo associat 4096 Feb 8 13:40 .
drwxrwxr-x 4 qf6kpo associat 4096 Feb 8 13:25 ..
-rw-rw-r-- 1 qf6kpo associat 0 Feb 8 13:25 truc.txt
```

```
[jeanmiche@p166bis jeanmiche]$ rename 's/>.txt$/truc/' *.txt
```

```
[jeanmiche@p166bis jeanmiche]$ ls -la
```

```
drwxrwxr-x 2 qf6kpo associat 4096 Feb 8 13:40 .
drwxrwxr-x 4 qf6kpo associat 4096 Feb 8 13:25 ..
-rw-rw-r-- 1 qf6kpo associat 0 Feb 8 13:25 truc.truc
```

Expression régulière

utilisable dans toutes les commandes UNIX

```
jduthill@septiles:~/enseignement/tp/gis/sujetTP >ls -l
-rw----- 1 jduthill jduthill 5931 Sep  6 2001 tp1.txt
-rw----- 1 jduthill jduthill 2696 Sep  6 2001 tp2.txt
-rw----- 1 jduthill jduthill 3123 Sep  6 2001 tp2gis
-rw----- 1 jduthill jduthill 4360 Sep  6 2001 tp2gis_corr
-rw-r--r-- 1 jduthill jduthill 66976 Feb 28 2005 tpFindEtMake.ps
-rw-r--r-- 1 jduthill jduthill 81002 Feb 28 2005 tpProcessus.ps
-rw-r--r-- 1 jduthill jduthill 68742 Feb 28 2005 tpRedirection.ps
-rw-r--r-- 1 jduthill jduthill 59022 Feb 28 2005 tpSecurite.ps
-rw-r--r-- 1 jduthill jduthill 89857 Feb 28 2005 tpShell.ps
-rw-r--r-- 1 jduthill jduthill 177835 Feb  1 14:31 tpSysteme.tgz
jduthill@septiles:~/enseignement/tp/gis/sujetTP >ls -l tp[12]*
-rw----- 1 jduthill jduthill 5931 Sep  6 2001 tp1.txt
-rw----- 1 jduthill jduthill 2696 Sep  6 2001 tp2.txt
-rw----- 1 jduthill jduthill 3123 Sep  6 2001 tp2gis
-rw----- 1 jduthill jduthill 4360 Sep  6 2001 tp2gis_corr
jduthill@septiles:~/enseignement/tp/gis/sujetTP >
```

Commandes et fichiers utiles

Fichiers d'initialisation (Bash)

```
/bin/bash
    The bash executable
/etc/profile
    The systemwide initialization file, executed for login shells
~/.bash_profile
    The personal initialization file, executed for login shells
~/.bashrc
    The individual per-interactive-shell startup file
~/.bash_logout
    The individual login shell cleanup file, executed when a
    login shell exits
```

Fichiers d'initialisation (Bash)

- au login
 - si `/etc/profile` existe alors l'exécuter
 - si `~/.bash_profile` existe
 - alors l'exécuter
 - sinon si `~/bash_login` existe
 - alors l'exécuter
 - sinon si `~/.profile` existe
 - alors l'exécuter
 - au logout
 - si `~/.bash_logout` existe alors l'exécuter

Fichiers d'initialisation

/etc/profile

```
# /etc/profile
#
PATH="/usr/local/bin:/usr/bin:/bin:/usr/bin/X11"
PS1="\$ "
HOSTNAME=`/bin/hostname`
HISTSIZE=1000
#
# export PATH PS1 HOSTNAME HISTSIZE
#
if [ -e $HOME/.bashrc ] ; then
        source $HOME/.bashrc
fi
```

Fichiers d'initialisation

~/.bashrc

```
#!/bin/bash
export PS1='\u@\h:\w >'
export PS2='suite>'
PATH=/bin:/sbin:/usr/bin:/usr/sbin:/usr/ucb:$PATH:~/.
export PATH
export LC_CTYPE=iso_8859_1
#
#
# quelques alias
#
#
alias which="type -path"
alias elm="cd ~; elm"
alias vt='export TERM=vt100'
alias whois='whois -h whois.univ-lille1.fr'
alias dirt='ls -lat|more'
```

\u username
\h hostname
\w répertoire courant

-initialisation propre à
l'utilisateur
-situé dans le répertoire HOME
de l'utilisateur
-redéfinition du prompt
-définition des alias
-définition de fonctions

Prise en compte immédiate du
fichier .bashrc
\$> source .bashrc

Commandes et fichiers utiles

at (UNIX)

```
at [-option] [-f fichier] HEURE
```

- lance les commandes indiquées à une heure précise
- si le fichier /etc/at.allow existe, seuls les utilisateurs dont les noms sont mentionnés dans ce fichier peuvent utiliser at
- si le fichier /etc/at.allow n'existe pas, at vérifie sur /etc/at.deny existe et tous les utilisateurs non-mentionnés dans ce fichier ont le droit d'invoquer at
- si aucun de ces deux fichiers n'existe, seul le super-utilisateur a le droit d'appeler at

Commandes et fichiers utiles

mail

envoi d'un e-mail

```
mail [option] [-s subject] ... to-addr ...
```

```
#!/bin/bash
ADMMAIL= "sinfo@polytech-lille.fr"
SUBJECT="message de la part de root"
```

```
mail -s "${SUBJECT}" ${ADMMAIL} <<!
La sauvegarde s'est terminée correctement
!
```

stty

stty [configuration ...]

- sert à régler les différents paramètres des terminaux
- très grand nombre de combinaison (voir la page man)

```
jduthill@pevele:/var/home/imaEns/jduthill > stty -a
speed 9600 baud; rows 0; columns 0; line = 0;
intr = ^C; quit = ^\; erase = ^?; kill = ^U; eof = ^D; eol = <undef>;
eol2 = <undef>; start = ^Q; stop = ^S; susp = ^Z; rprnt = ^R; werase = ^W;
lnext = ^V; flush = ^O; min = 1; time = 0;
)parenb -parodd cs8 -hupcl -cstopb cread -clocal -crtscs
-ignbrk -brkint -ignpar -parmrk -inpck -istrip -inlcr -igncr icrnl ixon -ixoff
-iuclc -ixany -imaxbel
opost -olcuc -ocrnl onlcr -onocr -onlret -ofill -ofdel nl0 cr0 tab0 bs0 vt0 ff0
isig icanon iexten echo echoe echok -echonl -noflsh -xcase -tostop -echoprt
echoctl echoke
```

stty erase '^?'
stty quit '^Z'
stty intr '^C'

touch

`touch [-option] [-r référence | -t date] fichier`

- touch modifie la date d'accès et la date de modification de chaque fichier indiqué.
- ces horodatages sont remplacés par la date et l'heure actuelles, sauf si l'option -r est utilisée, auquel cas ils sont remplacés par les horodatages correspondants du fichier référence, ou si l'option -t est utilisée, auquel cas on utilise la date mentionnée
- les fichiers n'existant pas sont créés, leur contenu est vide

```
[jm@p166bis tmp]$ ls -la touch_essai
ls: touch_essai: Aucun fichier ou répertoire de ce type
[jm@p166bis tmp]$ touch touch_essai
[jm@p166bis tmp]$ ls -la touch_essai
-rw-r--r-- 1 jm      jm          0 Feb 27 18:09 touch_essai
```

Commandes et fichiers utiles (astuce)

Problème posé par les noms de fichiers avec un espace et l'interprétation de ~ ou \$HOME

```
#!/bin/sh
# fichier astuce
echo -n nom du répertoire à lister :
read rep

# ce qui suit permet l'interprétation de
# ~ ou $HOME
nomrep=`eval echo $rep` 

# lecture ligne à ligne (permet de lire
# des noms de fichiers avec un espace)
ls $nomrep | while read fichier ; do
    if [ -d "$nomrep/$fichier" ] ; then
        echo $fichier est un répertoire
    else if [ -f "$nomrep/$fichier" ]
        then echo $fichier est un fichier
    fi
done
```

```
jduthill@weppes:~>ls tpGis/
comptage2
fichier essai
tp.txt
tp3
jduthill@weppes:~>./astuce
nom du répertoire à lister :~/tpGis
comptage2 est un fichier
fichier essai est un fichier
tp.txt est un fichier
tp3 est un fichier
jduthill@weppes:~>
```

telnet

```
telnet [-options] [-l username] host
```

- connexion sur un ordinateur hôte
- tend à disparaître au profit de ssh

```
jduthill@pevele:~ >telnet chouia.univ-lille1.fr -l ima2ie
Password:
Last login: Sun Feb 20 18:06:33 from pevele.eudil.fr

Digital UNIX V4.0F (Rev. 1229); Wed Aug 18 15:56:59 MET DST 1999

*****
UNIVERSITE DES SCIENCES ET TECHNOLOGIES DE LILLE
CENTRE DE RESSOURCES INFORMATIQUES
*****
You have new mail.
ima2ie@chouia.univ-lille1.fr% mail
```

ssh (*remote login*)

```
ssh [-l username] host
```

- login sur un ordinateur hôte de manière sécurisée

```
jduthill@pevele:/>ssh weppes
The authenticity of host 'weppes (2001:660:.....:b5ff:fe02:a905)' can't be established.
RSA key fingerprint is f3:04:c0:81:90:9d:59:e7:08:ec:92:e7:be:bc:22:66.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'weppes,2001:660:.....' (RSA) to the list of known hosts.
jduthill@weppes's password:
Bienvenue sur les machines de Polytech'Lille.
jduthill@weppes:~>exit
logout
Connection to weppes closed.
jduthill@pevele:/>ssh weppes
jduthill@weppes's password:
Bienvenue sur les machines de Polytech'Lille.
Last login: Mon Feb 16 10:21:33 2004 from pevele.escaut.net
jduthill@weppes:~>
```

exit : permet de quitter

Commandes et fichiers utiles

Ajout d'utilisateur (UNIX)

```
adduser | useradd [-d home_dir] [-g group] [-p passwd]
                  [-s shell] [-u uid] login
```

- **-d home_dir** : répertoire d'accueil au login de l'utilisateur (\$HOME)
- **-g group** : nom ou numéro du groupe auquel appartient l'usager. Ce groupe doit exister.
- **-p passwd** : le mot de passe de l'utilisateur
- **-s shell** : le nom du shell qui sera utilisé au login de l'usager.
- les informations sont rangées dans le fichier /etc/passwd (fichier en lecture pour tous) sauf le mot de passe qui est stocké dans le fichier /etc/shadow (en lecture pour root)

Commandes et fichiers utiles

Ajout de groupe (UNIX)

```
addgroup | groupadd [-g gid] group
```

- **-g gid** : c'est la valeur numérique de l'identifiant de groupe (GID). Il doit être UNIQUE. En général, les valeurs comprises entre 0 et 499 sont réservées au système. Par défaut ce sera la première valeur supérieure à tous les autres groupes.
- les informations sont rangées dans le fichier /etc/group

Commandes et fichiers utiles

Le mot de passe

- peut être changé par la commande `passwd`
- **NE PAS UTILISER** un mot du dictionnaire (pas seulement français !)
- **NE PAS UTILISER** votre nom ou prénom (ni d'un parent, enfant, animal, ...) ou toute variation de votre nom de login
- **NE PAS UTILISER** une information vous concernant (nb de téléphone, nb de voiture, nb INSEE, date d'anniversaire, ...)
- **UTILISER** une combinaison de majuscule, minuscule, chiffre, caractère spéciaux et au moins 12 caractères

Commandes et fichiers utiles

su (substitute user)

```
su [-] [nom [argument]]
```

- permet de changer d'utilisateur sans avoir besoin de se "déloguer" et de se "reloguer"
- la commande su sans paramètre permet de passer en mode super-utilisateur
- exit permet de revenir à l'utilisateur d'origine

```
jm@p166bis jm]# pwd  
/home/jm  
[jm@p166bis jm]$ su  
Password:  
[root@p166bis jm]# pwd  
/home/jm  
[root@p166bis jm]$ exit  
exit  
[jm@p166bis jm]$
```

```
[jm@p166bis jm]# pwd  
/home/jm  
[jm@p166bis jm]$ su -  
Password:  
[root@p166bis /root]# pwd  
/root  
[root@p166bis /root]# exit  
logout  
[jm@p166bis jm]$
```

Commandes et fichiers utiles

Qui est connecté

- who [-options]
- users

```
jduthill@weppes:~ >who
jduthill pts/0          Mar  1 08:39 (douaisis.escaut.net)
mcastele pts/1          Feb 24 14:31 (172.26.70.218)
dvivier :0              Feb 28 14:01
ocaron  pts/2          Feb 27 11:00 (baleares.staff.yser.net)

jduthill@weppes:~ >users
jduthill mcastele dvivier ocaron
jduthill@weppes:~ >
```

Commandes et fichiers utiles

fichiers /etc/passwd, /etc/shadow et /etc/group

```
[root@p166bis /etc]# cat group
root:x:0:root
bin:x:1:root,bin,daemon
daemon:x:2:root,bin,daemon
sys:x:3:root,bin,adm
adm:x:4:root,adm,daemon
tty:x:5:
disk:x:6:root
lp:x:7:daemon,lp
jm:x:503:
```

```
[root@p166bis jm]# cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:
daemon:x:2:2:daemon:/sbin:
adm:x:3:4:adm:/var/adm:
lp:x:4:7:lp:/var/spool/lpd:
jm:x:500:503::/home/jm:/bin/bash
```

```
[root@p166bis jm]# cat /etc/shadow
root:$1$AaYrU0.H$v84TWFOOdJGo3TM.RVL1z.:11006:0:99999:7:::
bin:*:11006:0:99999:7:::
daemon:*:11006:0:99999:7:::
adm:*:11006:0:99999:7:::
lp:*:11006:0:99999:7:::
jm:$1$XQ6CQ60R$5hKGfOQqptQzM27/OZT1p0:11011:0:99999:7:::
```

Commandes et fichiers utiles

fichiers /etc/passwd et /etc/group

- Groupe principal auquel appartient l'usager identifié par son GID
- L'usager peut appartenir à des groupes additionnels

```
jduthill@douaisis:~$ cat /etc/passwd | grep duthill
jduthill:x:1127:1011:Jean-Michel Duthilleul:/home/imaEns/jduthill:/bin/bash
rduthill:x:1475:1136:Richard Duthilleul:/home/sm2004/rduthill:xbinxbash
```

```
jduthill@douaisis:~$ cat /etc/group | egrep "duthill|1011"
sinfo::1059:nom1,nom2,jduthill,nom3,nom4,
gisEns::1046:user1, user2,user3, jduthill, user4, user5, user6
imaEns::1011:User1, User2, User3,
```

Commandes et fichiers utiles

fichiers /etc/passwd et /etc/group

```
jduthill@douaisis:~$ cat /etc/passwd | grep duthill
jduthill:x:1127:1011:Jean-Michel Duthilleul:/home/imaEns/jduthill:/bin/bash
rduthill:x:1475:1136:Richard Duthilleul:/home/sm2004/rduthill:xbinxbash
```

login:x:uid:gid:commentaire:répertoire_accueil:shell

```
jduthill@douaisis:~$ cat /etc/group | egrep "duthill|1011"
sinfo::1059:nom1,nom2,jduthill,nom3,nom4,
gisEns::1046:user1, user2,user3,jduthill,user4,user5,user6
imaEns::1011:User1, User2,User3,
```

groupe::gid:[user,user,...,]

Commandes et fichiers utiles

Connaître l'uid et le gid d'un usager

```
id [login]
```

```
jduthill@douaisis:~$ cat /etc/passwd | grep duthill
jduthill:x:1127:1011:Jean-Michel Duthilleul:/home/imaEns/jduthill:/bin/bash
rduthill:x:1475:1136:R:
```

```
jduthill@douaisis:~$ cat /etc/group | egrep "duthill|1011"
sinfo::1059:nom1,nom2,jduthill,nom3,nom4,
gisEns::1046:user1, user2,user3,jduthill,user4,user5,user6
imaEns::1011:User1, User2,User3,
```

```
jduthill@douaisis:~$ su
Password:
douaisis:/home/imaEns/jduthill# id
uid=0(root) gid=0(root) groups=0(root)
douaisis:/home/imaEns/jduthill# id jduthill
uid=1127(jduthill) gid=1011(imaEns) groups=1011(imaEns),1059(sinfo),1046(gisEns)
douaisis:/home/imaEns/jduthill#
```

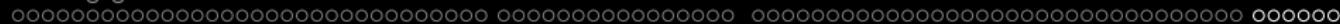
Commandes et fichiers utiles

Afficher les informations d'un utilisateur

finger [login]

```
jduthill@gayant01:~$ cat /etc/passwd | grep duthill
jduthill:x:1127:1011:Jean-Michel Duthilleul:/home/imaEns/jduthill:/bin/bash
rduthill:x:1475:1136:Richard Duthilleul:/home/sm2004/rduthill:/bin/bash
```

```
jduthill@weppes:~ >finger
Login Name Tty Idle Login Time Office Office Phone
jduthill Jean-Michel Duthilleul pts/0 Mar 1 08:39 (douaisis.escaut.net)
mcastele Maxime Castelein pts/1 4d Feb 24 14:31 (172.26.70.218)
ocaron Olivier Caron pts/2 1d Feb 27 11:00 (baleares.staff.yser.net)
jduthill@weppes:~ >finger rduthill
Login: rduthill Name: Richard Duthilleul
Directory: /home/sm2004/rduthill Shell: /bin/bash
Never logged in.
No mail.
No Plan.
jduthill@weppes:~ >
```



UNIX man (Nowhere Man)

He's a real UNIX Man
Sitting in his UNIX LAN
Making all his UNIX .plans
For nobody.

Knows the blocksize from du
Cares not where /dev/null goes to
Isn't he a bit like you And me?
UNIX Man, please listen
My lpd is missin' UNIX Man
The wo-o-o-orld is at your command.
He's as wise as he can be
Uses lex and yacc and C
UNIX Man, can you help me At all ?

UNIX Man, don't worry
Test with time, don't hurry UNIX Man
The new kernel boots, just like you had planned.
He's a real UNIX Man Sitting in his UNIX LAN
Making all his UNIX .plans For nobody ...
Making all his UNIX .plans For nobody.

Index I

Index II

addgroup, 150	dirname, 113	man, 62	su, 152
adduser, 149	echo, 70	mkdir, 65	tail, 108
alias, 101	eval, 120	more, 63	telnet, 147
at, 142	exit, 124	mv, 65, 68	test, 94
basename, 113	export, 71	newgrp, 45	touch, 145
cat, 63, 69	expr, 119	paste, 115	tr, 111
cd, 28	find, 70, 116	pwd, 28	umask, 40
chgrp, 39	finger, 158	read, 123	unalias, 102
chmod, 40	getfacl, 46	rmdir, 67	uniq, 110
chown, 39	grep, 107	rm, 63, 67	vi, 80
clear, 62	head, 109	sed, 130	wc, 106
cp, 67	id, 157	setfacl, 47	who, 153
cut, 105	join, 114	shift, 124	Redirection, 74
date, 62	ln, 48	sort, 69, 112	variable shell,
df, 30	ls, 29	ssh, 148	71
diff, 70	mail, 143	stty, 144	

