

Système de fichiers

Thomas Vantroys

thomas.vantroys@polytech-lille.fr

Polytech'Lille
Université de Lille

IMA4
2020 - 2021



Université
de Lille



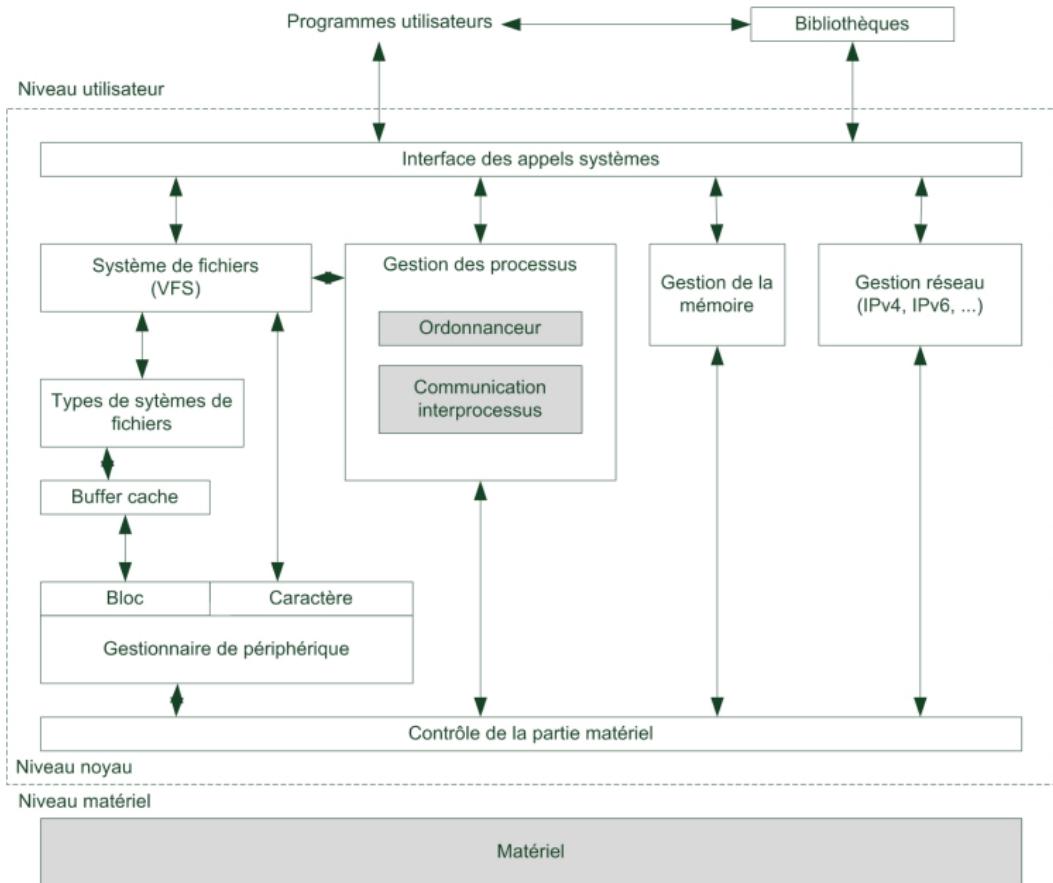
POLYTECH®
LILLE

1 Les disques et systèmes de fichiers

2 E/S

3 Exemple de système de fichiers

- FAT 16/FAT 32
 - HPFS
 - NTFS
 - ext2



Le système de fichier

- **Disque logique**

- c'est une arborescence dont chaque nœud correspond à un répertoire (*directory*) et chaque feuille à un fichier

- **Disque physique**

- Au modèle logique correspond un modèle physique qui correspond au matériel réellement utilisé pour la sauvegarde des informations

Le système de fichier

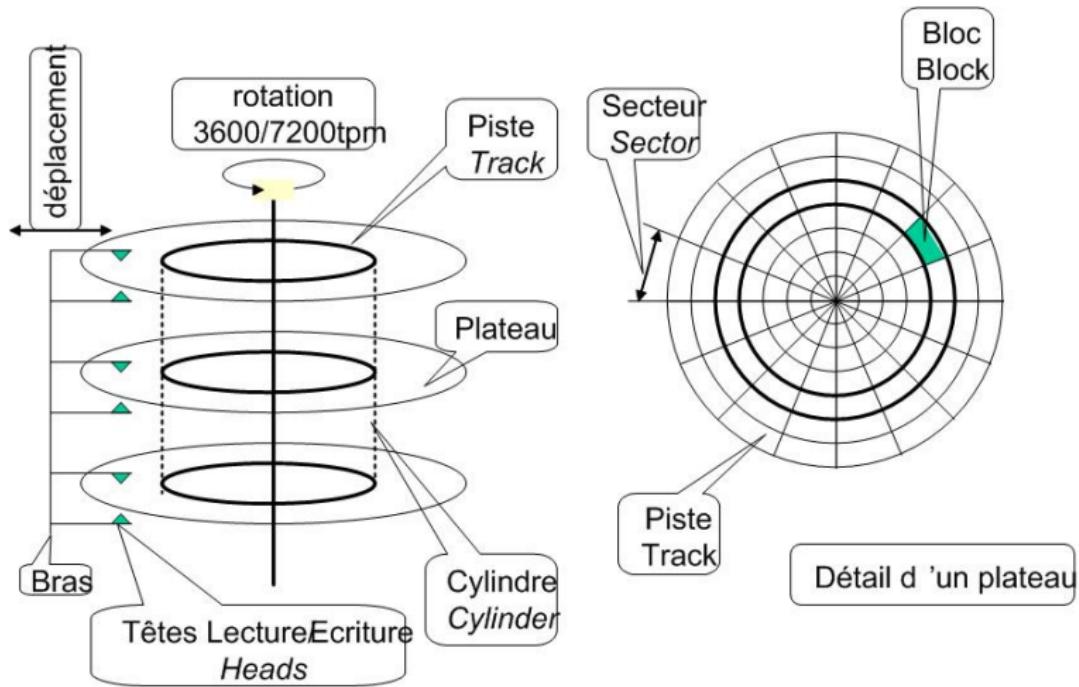
• Vue physique

- pour accéder à un bloc de 512 octets il faut 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.

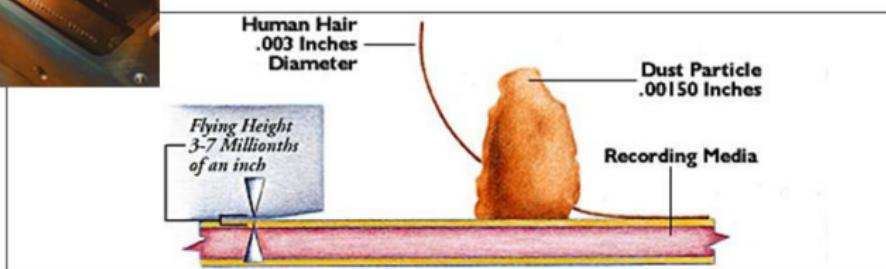
• Vue logique

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

Disque : surfaces magnétique circulaire



Disque : surfaces magnétiques circulaire



Disque : Fonctionnement

- Chaque plateau comporte deux faces
- Une tête de lecture/écriture par face. Toutes les têtes se déplacent simultanément
- Chaque face est divisée en :
 - secteurs : découpage longitudinal
 - pistes : découpage circulaire
- L'intersection secteur et piste est un Bloc
- L'empilement des pistes forme un cylindre
- L'accès est simultané à des données d'un même cylindre

Disque : Calcul de la taille

- secteur de 512 octets
 - 1024 cylindres, 16 têtes et 63 secteurs
 - $512 * 1024 * 16 * 63 = 528.482.304$ octets
 - Attention ! Disque généralement vendu pour 528 Mo alors qu'au formatage on ne trouve plus que 504 Mo
 - Rappel : 1 Ko = 1024 octets et 1 Mo = 1.048.576 octets
 - $1.048.576 * 504 = 528.482.304$ octets

Géométrie des disques CHS (*Cylinder/Head/Sector*)

- BIOS int13
 - C [0 – 1023], H [0 – 255], S [1 – 63]
 - adressage de 8.455.716.864 octets = 7.8 Go
 - Standard IDE
 - C [0 – 65535], H [0 – 15], S [0 – 255]
 - adressage de 137.438.953.471 octets = 128 Go
 - Combinaison des deux
 - C [0 – 1023], H [0 – 15], S [1 – 63]
 - adressage de 528.482.304 octets = 528 Mo
 - Utilisation de C', H', S' : norme LBA (*Logical Block Address*)
 - $C' = \frac{C}{N}$, $H' = H * N$, $S' = S$

Géométrie des disques LBA (*Logical Block Address*)

- Codification sur 28 bits :

- C sur 16 bits (0 - 65535)
- H sur 4 bits (0 - 15)
- S sur 8 bits (0 - 255)
- adressage de $137.438.953.471 \text{ octets} = 128 Go$

- Nouveau codage . . . sur 48 bits

- plus de répartition en C, H, S mais codage d'un numéro de secteur entre 0 et 2^{48} , ce qui donne une capacité d'adressage théorique de
 $2^{48} * 512 = 128 Po$ (144.115.188.075.855.872 octets)

Rappel des abréviations standards

abréviation	appellation	correspondance décimale	correspondance binaire	valeur
Ko	Kilo-octet	1 000	1 024	2,E+10
Mo	Méga-octet	1 000 000	1 048 576	2,E+20
Go	Giga-octet	1 000 000 000	1 073 741 824	2,E+30
To	Téra-octet	1 000 000 000 000	1 099 511 627 776	2,E+40
Po	Péta-octet	1 000 000 000 000 000	1 125 899 906 842 624	2,E+50
Eo	Exa-octet	1 000 000 000 000 000 000	1 152 921 504 606 846 976	2,E+60

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)

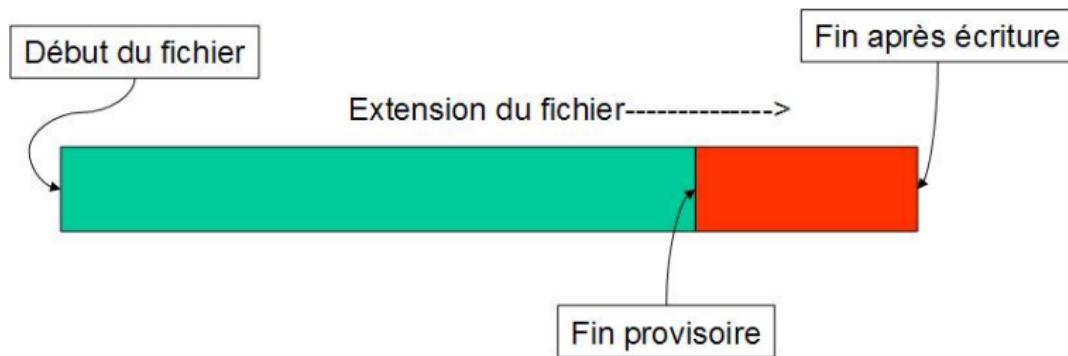
- Créer un fichier :
 - Trouver de la place dans le système de fichiers
 - Créer une entrée dans le répertoire
 - L'entrée du répertoire mémorise le nom du fichier et son emplacement dans le système de fichiers
- Écrire
 - Au travers d'un appel système, il faut rechercher le répertoire afin de trouver 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

Méthode d'accès

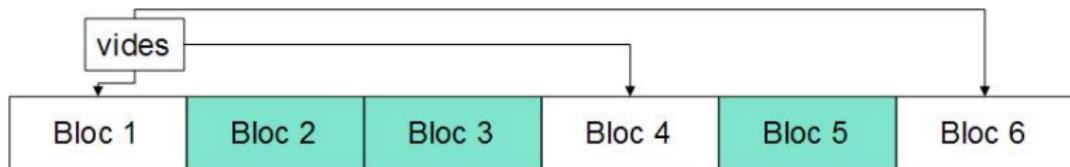
- Accès séquentiel
 - Le plus simple
 - Tout ajout d'information s'effectue à la fin du fichier et à jour le pointeur de fin de fichier



Méthode d'accès

- Accès direct

- Constitué d'enregistrements logiques de longueur fixe
- Permet l'accès immédiat à un enregistrement
- La taille du fichier est connue et peut être réservée à sa création (taille d'un bloc * nombre de blocs)

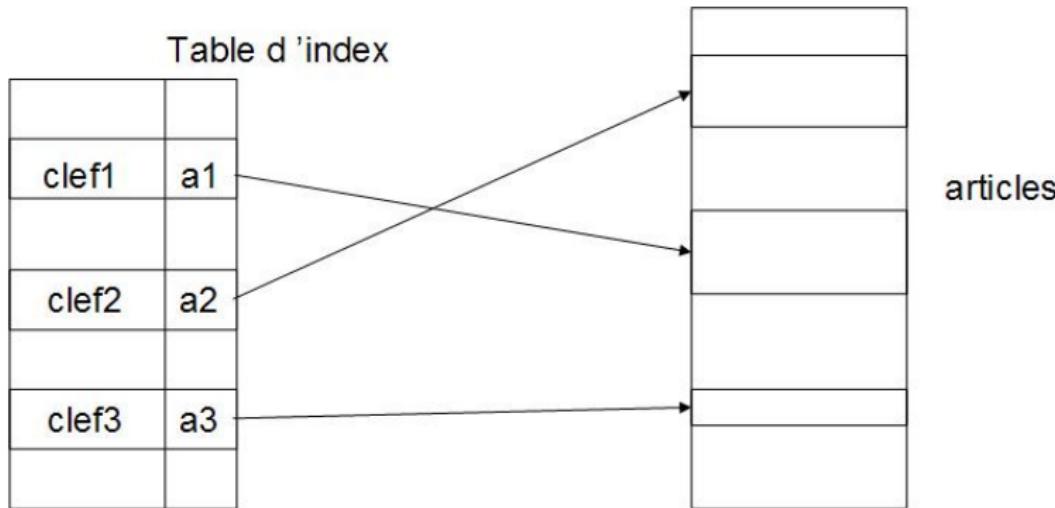


On peut accéder directement au bloc 5 sans que les précédents aient été remplis au préalable.

Méthode d'accès

- Accès indexé

- Nécessite d'avoir un ensemble de clés ordonnées
- Relation entre clés et articles établie au moyen d'une table d'index qui peut être incluse ou séparée du fichier des articles



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 mémorie l'information (nom, emplacement, taille, ...) sur tous les fichiers de cette partition

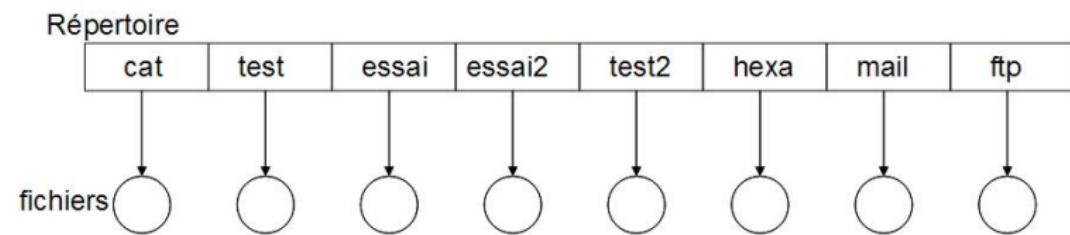
Structure des répertoires

- 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.



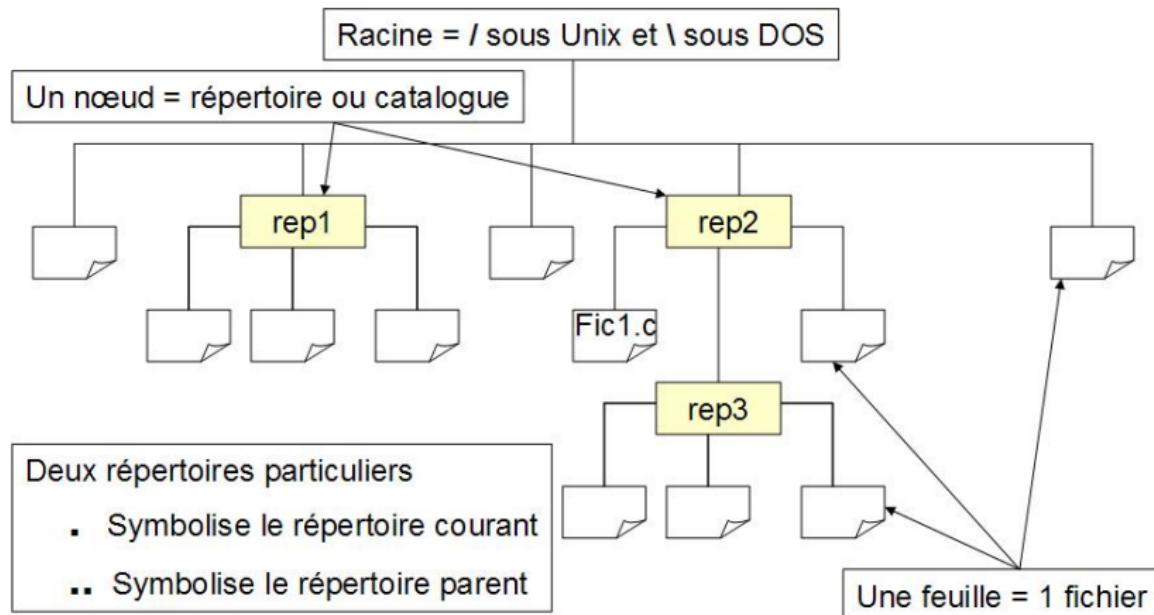
Structure des répertoires

- 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

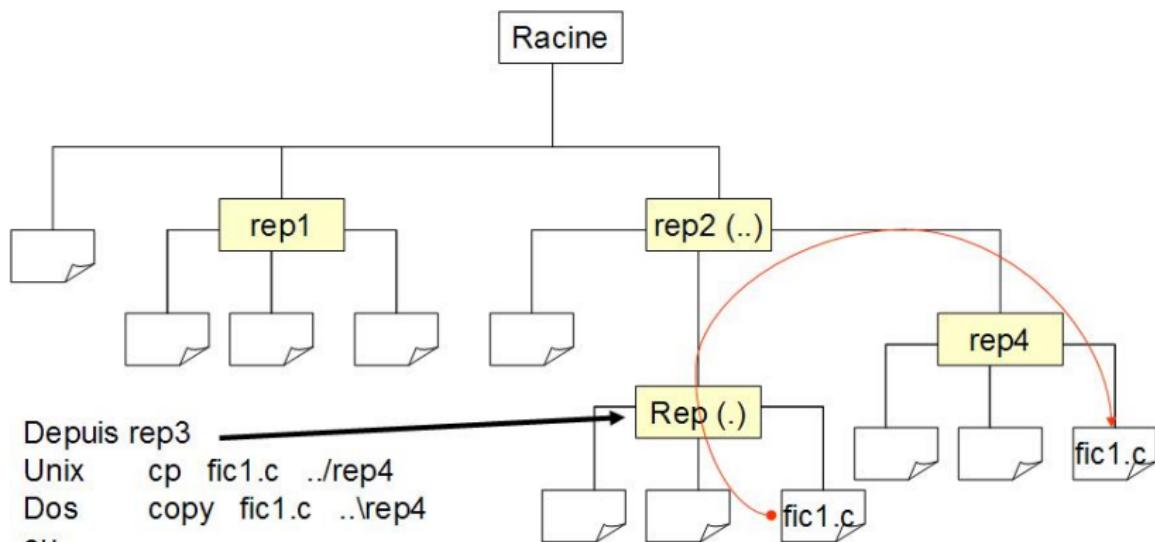
Le répertoire

Structure arborescente



Le répertoire

Structure arborescente



Depuis rep3

```
Unix cp fic1.c ..../rep4
```

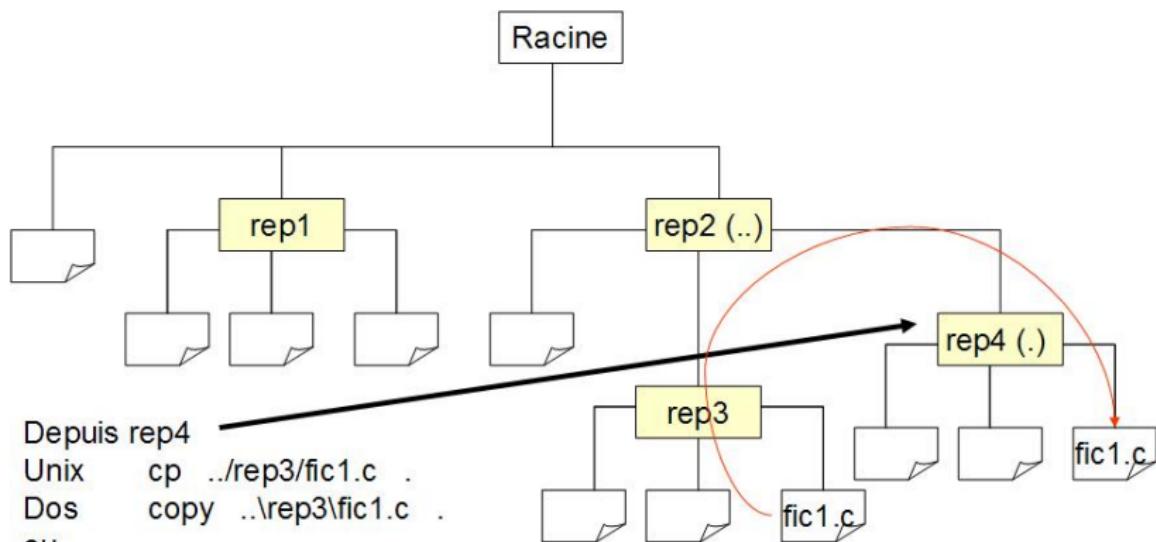
```
Dos      copy fic1.c ..\rep4
```

Unix cp fic1.c /rep2/rep4

Dos copy fic1.c \rep2\rep4

Le répertoire

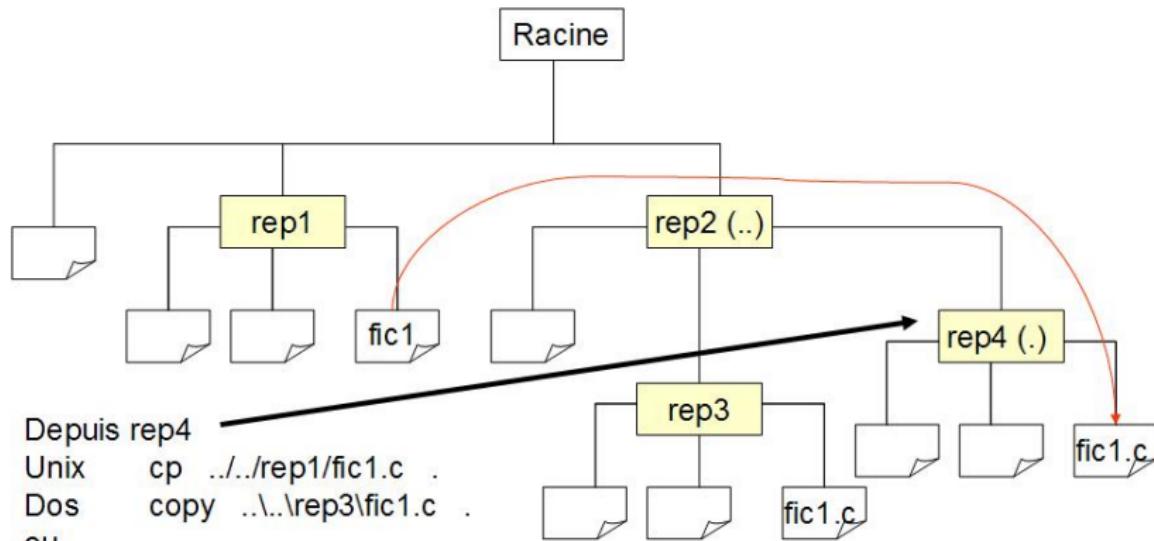
Structure arborescente



```
Depuis rep4
Unix    cp  ..\rep3\fic1.c .
Dos     copy ..\rep3\fic1.c .
ou
Unix    cp  /rep2/rep3\fic1.c .
Dos     copy \rep2\rep3\fic1.c .
```

Le répertoire

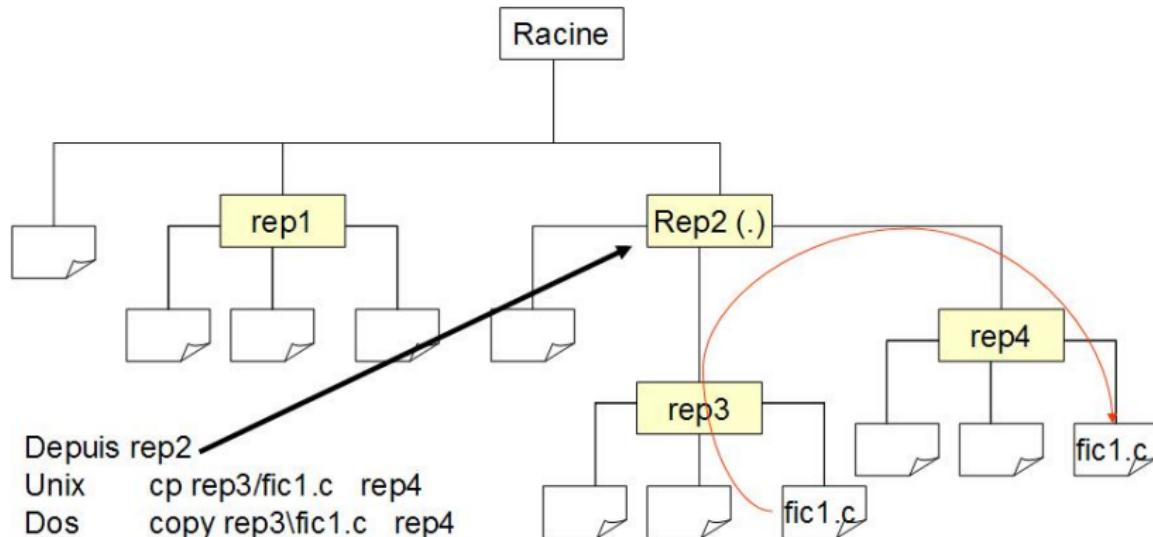
Structure arborescente



```
Depuis rep4
Unix    cp ../../rep1/fic1.c .
Dos     copy ..\rep3\fic1.c .
ou
Unix    cp /rep1/fic1.c .
Dos     copy \rep1\fic1.c .
```

Le répertoire

Structure arborescente



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
- 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

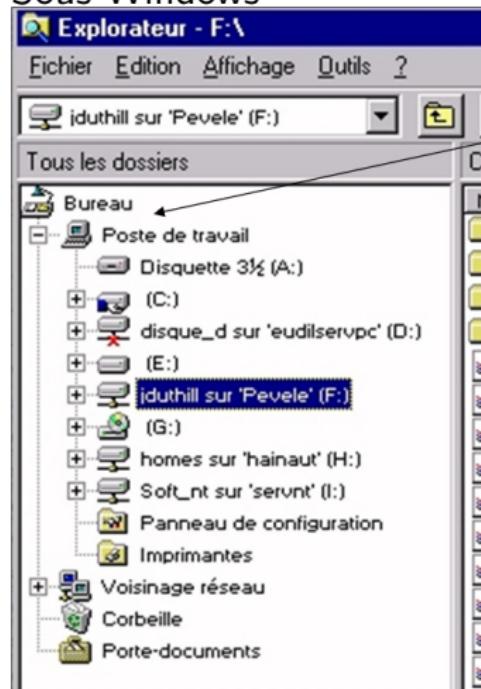
Sous Unix

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

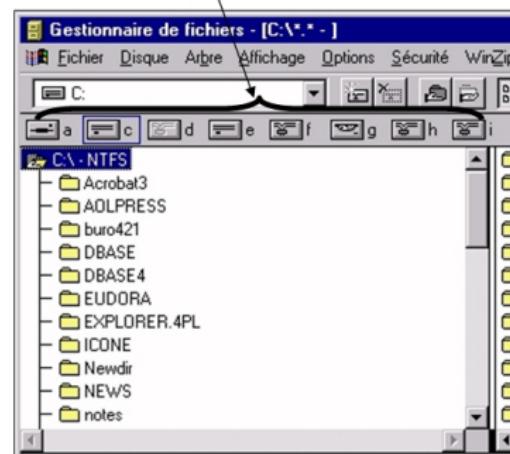
```
jduthill@hainaut:~ >df -k
Filesystem      1k-blocks    Used Available Use% Mounted on
/dev/sda4        792800    533356    218479  71% /
/dev/sda5        497667     8850    463115   2% /tmp
/dev/sda6        792800    42958    708878   6% /var
/dev/sdb1        8634569   7041939   1144820  86% /home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

Le répertoire

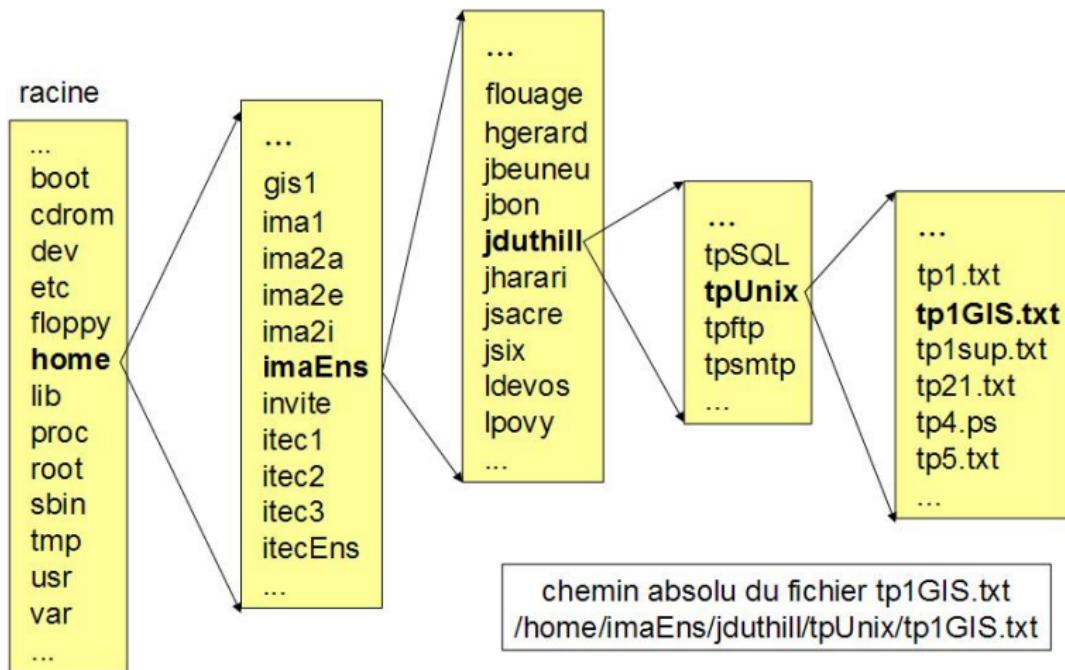
Sous Windows



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



Exemple



Le système de fichier

• Référence de fichiers et répertoires

- référence absolue d'un fichier
 - nom "complet" du fichier = chemin d'accès depuis la racine.
Exemple, sous Unix : /rep2/fic1.c ou sous DOS :
\rep2\fic1.c
- répertoire de travail ou courant
 - répertoire sur lequel un utilisateur est positionné à un moment donnée. Ce répertoire est nommé : .
- référence relative d'un fichier
 - référence d'un fichier à partir du répertoire de travail
- répertoire privé (sous Unix)
 - répertoire sur lequel est positionné un utilisateur suite à son authentification

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

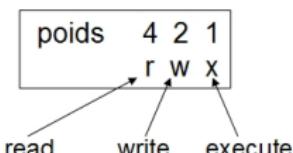
- Un utilisateur est caractérisé par :
 - un numéro de compte uid
 - un numéro de groupe gid

```
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
```

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

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

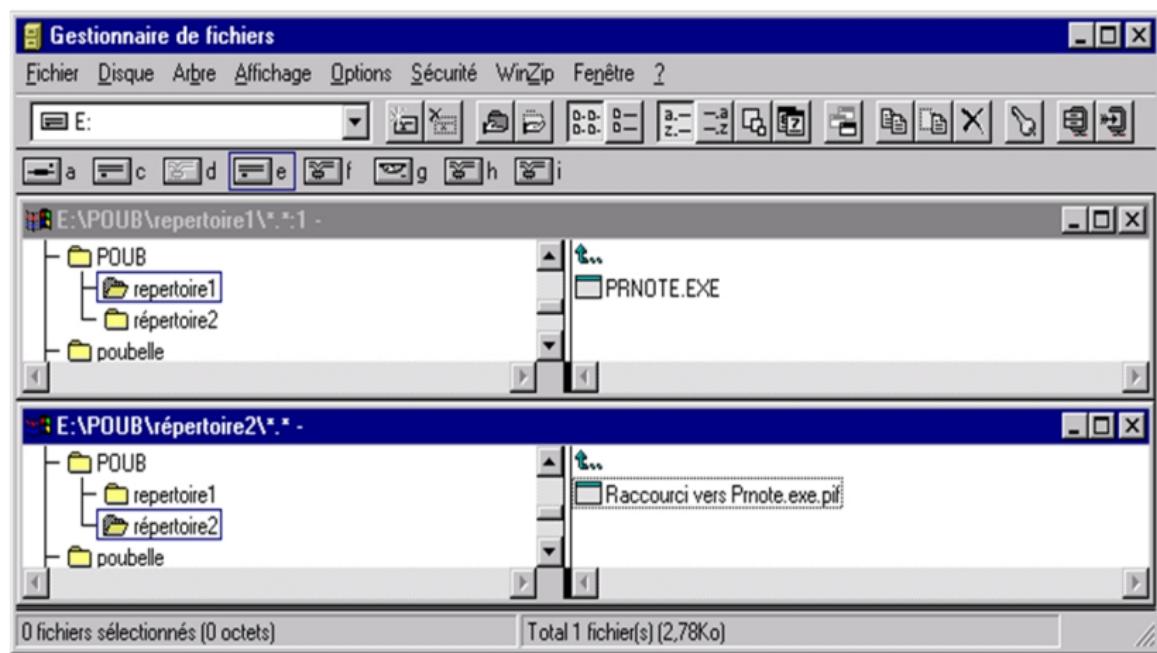
UNIX : liens

- 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
```

WinNT : liens



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 système pour tous les utilisateurs

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

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

Partitionnement des disques

- Une partition est une zone contiguë de secteurs
- Il y a une table des partitions en tête du disque
- Permet des systèmes de fichiers différents
- C'est parfois obligatoire :
 - disques de grande capacité
 - partitions pour la zone d'échange (swap)
 - différents systèmes d'exploitation
- C'est parfois utile :
 - limiter l'espace alloué
 - évite d'exporter un disque entier
 - partition en lecture seule

Partitionnement des disques

- Amorce principale (1^{er} secteur du disque)

Master Boot Record (MBR)



Table de partitions (ex. 2 partitions, disque 6Go)

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
000000000	EB	69	4C	49	4C	4F	01	00	14	00	8D	03	00	00	00	00	éiLILo....□.....
000000010	BF	1B	06	50	1E	1C	81	55	01	1F	1C	81	55	01	1D	1C	¿...P..□U...□U...
000000020	81	55	01	01	00	00	00	00	00	00	21	1C	81	55	01	□U.....!□U.	
000000030	19	1D	81	07	01	1A	1D	81	07	01	1B	1D	81	07	01	1C	..□....□....□...
000000040	1D	81	07	01	1D	1D	81	07	01	1E	1D	81	07	01	1F	1D	.□....□....□....
000000050	81	07	01	20	1D	81	07	01	00	00	00	00	00	00	00	00	□... .□.....
000000060	00	00	00	00	00	00	00	00	00	00	B8	C0	07	8E	D8,À.□Ø	
<hr/>																	
0000001A0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0000001B0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	80	01□.
0000001C0	01	00	00	OB	FE	7F	89	3F	00	00	00	CB	94	60	00	00	...p□?...É"...
0000001D0	41	8A	05	FE	FF	12	0A	95	60	00	49	56	60	00	00	00	AS.pY..•'IV'...
0000001E0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00U*
0000001F0	00	00	00	00	00	00	00	00	00	00	00	00	00	55	AA	

- État de la partition: 00=inactive, 80=active
- début de la partition: tête: 1 octet
- type de partition: cylindre/secteur: 2 octets (voir plus loin)
- fin de partition: tête 01=FAT12,04=FAT16,05=Etendue,06=FAT16,0B=FAT32
- nombre de secteur entre MBR et partition: cylindre/secteur: 2 octets (voir plus loin)
- longueur partition en secteur: 00 00 00 3F, 00 60 95 0A soit 3Go
- longueur partition en secteur: 00 60 94 CB, 00 60 56 49 soit 3Go

00 60 95 0A + 3F = 00 60 95 49 * 512 = 0C12A9200

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
QC12A9200	EB	58	90	4D	53	57	49	4E	34	2E	31	00	02	08	20	00
QC12A9210	02	00	00	00	00	F8	00	00	3F	00	FF	00	3F	00	00	00
QC12A9220	0A	56	60	00	10	18	00	00	00	00	00	00	02	00	00	00
QC12A9230	01	00	06	00	00	00	00	00	00	00	00	00	00	00	00	00
QC12A9240	80	00	29	FA	1A	45	40	44	49	53	51	55	45	5F	44	20
QC12A9250	20	20	46	41	54	33	32	20	20	20	FA	33	C9	8E	D1	BC
QC12A9260	F8	7B	8E	C1	BD	78	00	C5	76	00	1E	56	16	55	BF	22
QC12A9270	05	89	7E	00	89	4E	02	B1	0B	FC	F3	A4	8E	D9	BD	00
QC12A9280	7C	C6	45	FE	0F	8B	46	18	88	45	F9	38	4E	40	7D	25
QC12A9290	8B	C1	99	BB	00	07	E8	97	00	72	1A	83	EB	3A	66	A1
QC12A92A0	1C	7C	66	3B	07	8A	57	FC	75	06	80	CA	02	88	56	02
QC12A9370	5E	72	0A	40	75	01	42	03	5E	0B	49	75	B4	C3	03	18
QC12A9380	01	27	0D	0A	44	69	73	71	75	65	20	6E	6F	6E	20	73
QC12A9390	79	73	74	65	6D	65	20	FF	0D	0A	45	72	72	65	75	72
QC12A93A0	20	64	27	45	2F	53	20	20	FF	0D	0A	52	65	6D	70	6C
QC12A93B0	61	63	65	7A	2D	6C	65	20	65	74	20	61	70	70	75	79
QC12A93C0	65	7A	20	73	75	72	20	75	6E	65	20	74	6F	75	63	68
QC12A93D0	65	20	20	0D	0A	00	00	00	49	4F	20	20	20	20	20	20
QC12A93E0	53	59	53	4D	53	44	4F	53	20	20	20	53	59	53	7E	01
QC12A93F0	00	57	49	4E	42	4F	4F	54	20	53	59	53	00	00	55	AA

Table de partitions (ex. 2 partitions, disque 4Go)

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000000000	EB	69	4C	49	4C	4F	01	00	14	00	8D	03	00	00	00	00
0000000010	E7	B1	CE	37	27	1D	81	07	01	28	1D	81	07	01	26	1D
0000000020	81	07	01	01	00	00	00	00	00	00	00	2A	1D	81	07	01
0000000030	19	1D	81	07	01	1A	1D	81	07	01	1B	1D	81	07	01	1C
0000000040	1D	81	07	01	1D	1D	81	07	01	1E	1D	81	07	01	1F	1D
0000000050	81	07	01	20	1D	81	07	01	00	00	00	00	00	00	00	00
0000000060	00	00	00	00	00	00	00	00	00	00	00	B8	C0	07	8E	D8
.....
0000001A0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0000001B0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	01
0000001C0	01	00	82	FE	3F	10	3F	00	00	00	92	2A	04	00	80	00
0000001D0	01	11	83	FE	BF	0B	D1	2A	04	00	3B	48	7C	00	00	00
0000001E0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0000001F0	00	00	00	00	00	00	00	00	00	00	00	00	00	55	AA	U ^a

État de la partition: 00=inactive, 80=active

début de la partition: tête 1 octet

cylindre/secteur: 2 octets (voir plus loin)

82=swap, 83 linux

type de partition: 82=swap, 83 linux

fin de partition: tête 1 octet

cylindre/secteur: 2 octets (voir plus loin)

2000003F, 00042AD1 soit 128Mo

nombre de secteur entre MBR et partition:

00042A92, 007C483B soit 4Go

longueur partition en secteur

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
0000001B0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	80	01□.
0000001C0	01	00	0B	FE	7F	89	3F	00	00	00	CB	94	60	00	00	00	...b?...E''...
0000001D0	41	8A	05	FE	FF	12	0A	95	60	00	49	56	60	00	00	00	AŠ.bý...•.IV'...
0000001E0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0000001F0	00	00	00	0D	00	00	00	00	00	00	00	00	00	00	55	AAU"

début de la partition: tête → 1 octet
 cylindre/secteur: 2 octets

fin de partition:
 tête ↓ 1 octet
 cylindre/secteur: 2 octets

composition cylindre/secteur

cylindre bits 7 à 0						cyl 9+8			secteur bits 5 à 0							

01 00 -->	00 01 --> cylindre 0,	secteur 1
7F 89 -->	89 7F --> cylindre 189,	secteur 3F
41 8A -->	8A 41 --> cylindre 18A,	secteur 1
FF 12 -->	12 FF --> cylindre 312,	secteur 3F

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
000000000	EB	69	4C	4	00	00	3E	C1	*	512=	7D	82	00	00	00	
0000001B0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
0000001C0	01	01	05	FE	BF	0A	C1	3E	00	00	8A	F5	7F	00	00	
0000001D0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
0000001E0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
0000001F0	00	00	00	00	00	00	00	00	00	00	00	00	00	55	AA	

Disque de 4Go
1 partition étendue
découpée en 4 partitions
logiques de 1Go
(disques E; F; G; H:)

4Go

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0007D8200	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
	$(00\ 20\ 1C\ C3\ *\ 512\) + 7D\ 82\ 00 = 40\ B7\ 08\ 00$															
0007D83B0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	01	
0007D83C0	01	01	05	FE	3F	83	3F	00	00	00	84	1C	20	00	00	
0007D83D0	01	84	05	FE	7F	06	C3	1C	20	00	C3	1C	20	00	00	
0007D83E0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
0007D83F0	00	00	00	00	00	00	00	00	00	00	00	00	00	55	AA	

disque E:

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
040B70800	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
	$(00\ 40\ 39\ 86\ *\ 512\) + 7D\ 82\ 00 = 80\ F0\ 8E\ 00$															
040B709B0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	01	
040B709C0	01	84	05	FE	7F	06	3F	00	00	00	84	1C	20	00	00	
040B709D0	41	07	05	FE	7F	89	86	39	40	00	C3	1C	20	00	00	
040B709E0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
040B709F0	00	00	00	00	00	00	00	00	00	00	00	00	00	55	AA	

disque F:

etc...

Partitionnement des disques

- Exemple de partitionnement (Linux)

```
pevèle:/home/imaEns/jduthill# fdisk -l
```

Disk /dev/sda: 36.4 GB, 36401479680 bytes

Disque sda

255 heads, 63 sectors/track, 4425 cylinders

Units = cylinders of 16065 * 512 = 8225280 bytes

Device	Boot	Start	End	Blocks	Id	System
/dev/sda1		1	609	4891761	83	Linux
/dev/sda2		610	853	1959930	82	Linux
/dev/sda3		854	976	987997+	83	Linux
/dev/sda4		977	4425	27704092+	5	Extended
/dev/sda5		977	1585	4891761	83	Linux
/dev/sda6		1586	2802	9775521	83	Linux
/dev/sda7		2803	3411	4891761	83	Linux
/dev/sda8		3412	4425	8144923+	83	Linux

Disk /dev/sdb: 36.4 GB, 36401479680 bytes

Disque sdb

255 heads, 63 sectors/track, 4425 cylinders

Units = cylinders of 16065 * 512 = 8225280 bytes

Device	Boot	Start	End	Blocks	Id	System
/dev/sdb1		1	1217	9775521	83	Linux
/dev/sdb2		1218	4425	25768260	83	Linux

Partitionnement des disques

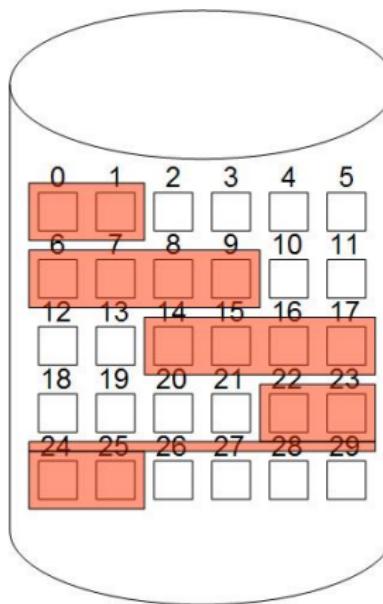
- Exemple de partitionnement (NT)

CD-ROM 0 CD-ROM (I:) Connecté	2 CDrom			
CD-ROM 1 DVD (H:) Connecté				
Disque 0 De base 14,26 Go Connecté	Disque 15 Go (4 partitions)			
(D:) 1,00 Go Sain	4,96 Go NTFS Sain	(C:) 7,27 Go NTFS Sain (Système)	(D:) 1,03 Go FAT32 Sain	
Disque 1 De base 74,53 Go Connecté	(E:) 4,96 Go NTFS Sain	(F:) 4,96 Go NTFS Sain	(G:) 64,62 Go NTFS Sain	Disque 80 Go (3 partitions)
Disque 2 De base 125 Mo Connecté	FlashDisk (K:) 61 Mo FAT Sain (Actif)	64 Mo Sain	Clé USB (2 partitions)	

Système de fichier

- Allocation contiguë de l'espace disque

Fichier	début	lgr
cours	0	2
tr	14	4
mail	6	4
list	22	4



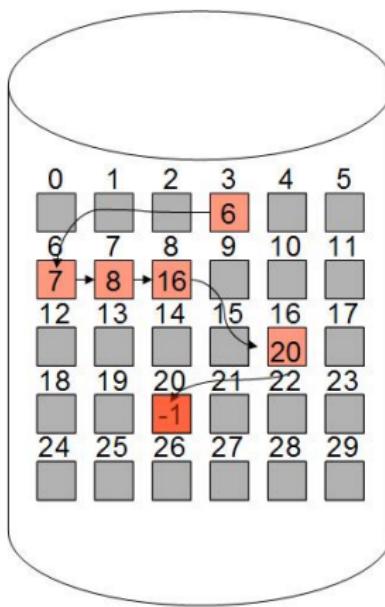
Difficultés:

Déterminer la place nécessaire à un fichier et trouver de la place pour un nouveau fichier

Système de fichier

- Allocation chaînée

Fichier début ligne
cours 3 6

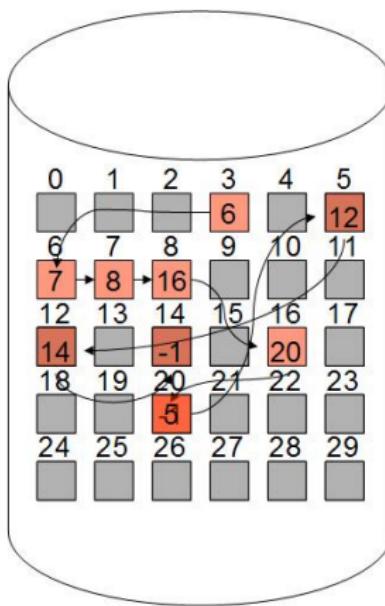


- Chaque fichier est une liste chaînée de blocs.
- Chaque bloc contient un pointeur sur le bloc suivant.

Système de fichier

- Allocation chaînée

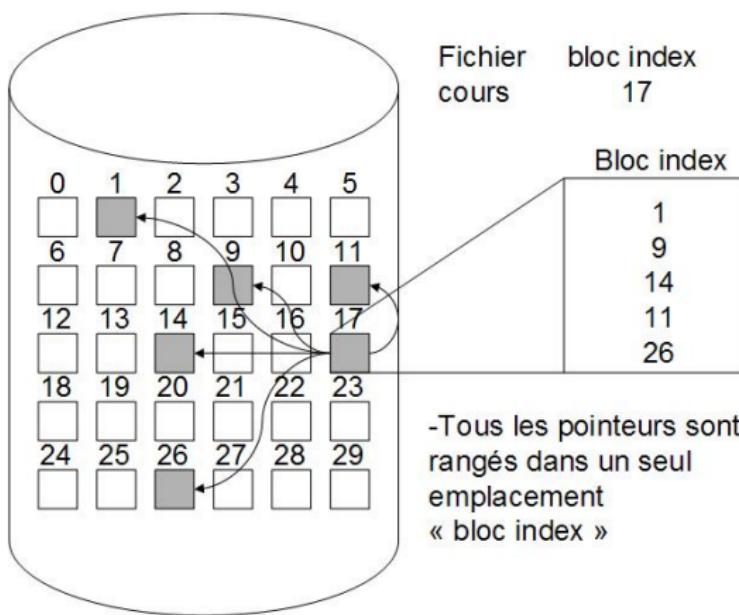
Fichier début lgr
cours 3 69



- Chaque fichier est une liste chaînée de blocs.
- Chaque bloc contient un pointeur sur le bloc suivant.

Système de fichier

- Allocation indexée



Entrées / Sorties sous Unix

Organisation générale des E/S sous UNIX

- Deux types :
 - E/S en mode bloc (ou structuré) : correspond aux entrées/sorties sur disque
 - E/S en mode caractère (non structuré) : correspond aux entrées/sorties sur terminaux
- Opérations réalisées au travers du *buffer cache*

Le buffer cache

- but : minimiser la fréquence d'accès au disque par le noyau en gérant une réserve de tampons en données internes appelé buffer cache, qui contient les données des blocs disque les plus récemment utilisés
- buffer cache = structure logicielle
- différent des caches matériels pour accélérer les références à la mémoire
- lecture d'une donnée :
 - lecture dans le cache
 - donnée présente : pas d'accès disque
 - sinon, accès disque et rangement dans le cache
- écriture d'une donnée :
 - écriture dans le cache : donnée présente si relue plus tard
 - tente de minimiser les opérations d'écriture sur disque
- lecture séquentielle d'un fichier : anticipation sur lecture de bloc : 1 requête synchrone suivie d'une requête asynchrone



Avantages du buffer cache

- permet un accès uniforme au disque (partie d'un fichier, d'un i-nœud, ...)
 - code plus modulaire (interface tous usages pour les E/S)
 - conception du système plus simple
- réduction de l'utilisation du disque
 - augmentation des capacités de traitement du système
 - diminution du temps de réponse
- aide à la sécurité du système en assurant un accès sérialisé de plusieurs processus à un même bloc

Avantage du buffer cache

- remarques :
 - plus le nombre de tampons augmente, plus le trafic diminue
 - MAIS si le nombre de tampons augmente, la mémoire disponible diminue et cela augmente le *swapping*
- transmission d'une quantité faible de données = augmentation des performances (mémorisation des données jusqu'à ce qu'il soit "économique" de les transmettre depuis ou vers le disque)

Inconvénients du buffer cache

- panne fatale (crash) + écriture différée = l'utilisateur n'est jamais sûr qu'un appel système write a finalement conduit à une écriture sur le disque
- tampon intermédiaire = copie de donné intermédiaire
- transmission d'une quantité de données importante = baisse des performances

Les appels systèmes d'E/S

- ouverture d'un fichier : `int open(char *pathname, int flags)`
- création d'un fichier : `int creat(const char *pathname, mode_t mode)`
- lecture dans un fichier : `int read(int fd, void *buf, size_t count)`
- écriture dans un fichier : `int write(int fd, const void *buf, size_t count)`
- fermeture de fichier : `int close(int fd)`

Autres appels système

- création d'un i-nœud : `int mknod(const char *pathname, mode_t mode, dev_t dev)`
- lecture d'un i-nœud : `stat(..), fstat(..), lstat(..)`
- déplacement du pointeur de lecture/écriture dans un fichier : `lseek(..)`
- vérification de la possibilité d'accéder à un fichier : `access(..)`
- création d'un lien : `link(..)`
- suppression d'un lien : `unlink(..)`
- changement de répertoire : `chdir(..)`

La bibliothèque C standard

- /lib/libc.a : le fichier d'archive correspondant
- pas directement d'instruction d'E/S en C => appels système aux fonctions de la bibliothèque standard
- contient les fonctions permettant :
 - les E/S bufférissées
 - manipulation des chaînes de caractères
 - conversion et classification de caractères
 - manipulation de temps
 - allocation mémoire
 - accès aux variables d'environnement

La bibliothèque C standard

- fournit une interface standard et portable entre UNIX et d'autres systèmes d'exploitation qui supportent ces bibliothèques
- Les E/S standard : le cœur de la bibliothèque C standard
 - optimise la taille des données transmise entre les périphériques (disques et terminaux) et les programmes
 - manipulation des informations de la taille d'un bloc logique du disque (=> plus efficace)
 - gestion par une structure de type FILE
- structure FILE :
 - un descripteur de fichier
 - les opérations autorisées sur le fichier
 - un pointeur sur un buffer
- toute opération est effectués par l'intermédiaire d'un pointeur sur une structure FILE
- accès aux E/S standard => inclure le fichier stdio.h
 - accès aux macros, constantes et définitions nécessaires

Le fichier stdio.h

Constantes :

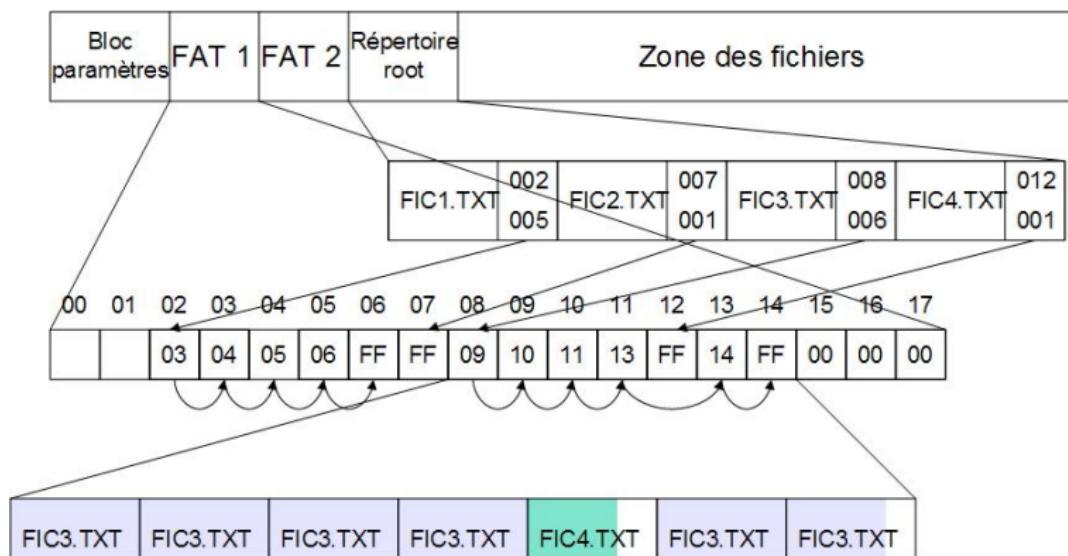
- BUFSIZ : taille tampon en zone utilisateur
- _NFILE : nombre maximum de fichiers pouvant être ouverts
- NULL : pointeur Null
- EOF : caractère *End Of File*
- stdin : entrée standard (descripteur 0)
- stdout : sortie standard (descripteur 1)
- stderr : sortie erreur standard (descripteur 2)

Exemples de systèmes de fichiers

FAT 16 (File Allocation Table)

- Système simple utilisé par DOS, OS/2, Windows
- Structure :
 - Secteur de boot (Master Boot Record) contenant la structure de l'unité physique (nombre de partition, taille, type, etc.)
 - La première table d'allocation de fichier
 - Le répertoire racine avec le nom de volume
 - La zone de données pour les fichiers et sous répertoire
- Noms de fichiers à 11 caractères (8+3)
- Pas de différence entre majuscules et minuscules
- Protection des fichiers rudimentaire
- Volume de 2GO maximum

FAT 16 (File Allocation Table)



FAT 16 (boot)

Le volume dans le lecteur C s'appelle **DISQUE_C**
Le numéro de série du volume est **19DB-1534**

		JMP 3C	NOP	Système BOOT Sect abs disque 0000000									
Déplacement		Codes hex-										Valeur ASCII	
0000(0000)	EB 3C 90 4D 53 44 53 50 36 2E 30 00 02 10 1C 00	<ÉMSDSP6.0 □□□										□ □ □	° □ □
0016(0010)	02 00 02 00 00 F8 E4 00 11 00 0F 00 00 00 00 00	□ □ □ □ □ □ □ □ □ □ □ □ □										□, □) 4S □ DISQU
0032(0020)	14 2C 06 00 00 00 29 34 15 DB 19 44 49 53 51 55	¶, □) 4S □ DISQU										E_C	FAT16 - 3
0048(0030)	45 5F 43 20 20 20 46 41 54 31 36 20 20 20 FA 33	E_C FAT16 - 3										+Ä-+	□□+x 6+7□V
0064(0040)	CO 8E D0 BC 00 7C 16 07 BB 78 00 36 C5 37 1E 56	+Ä-+ □□+x 6+7□V											
octets/secteur	02 00 soit 512	FC E	secteurs cachés 00 00 00 00 soit 0										
secteurs/cluster	10 soit 16	89 4	secteurs larges 00 06 2c 14										
secteurs de boot	00 1c soit 28	13 7	usage du disque 00 00										
nombre de FAT	02 soit 2	26 1	signature 29 (ou 28) pour NT										
nbr entrées ds root	02 00 soit 512	02 0	signature 29 (ou 28) pour NT										
petits secteurs	00 00 soit 0	B8 2	identificateur 19 db 15 34										
support	F8 soit disque dur		01 0	nom du volume 11 octets (ici DISQUE_C)									
secteurs/FAT	00 e4 soit 228	A1 5	système de fichier 8 octets (ici FAT16)										
secteurs/piste	00 11 soit 17	FB B9 0B 00 BE E0 7D F3	□ □ □ □ □ □ □ □ □ □ □ □ □										
nombre de têtes	00 0F soit 15	00 F3 A6 74 18 BE 9E 7D	□ □ □ □ □ □ □ □ □ □ □ □ □										
		1F 8F 04 8F 44 02 CD 19	□ □ □ □ □ □ □ □ □ □ □ □ □										

Boot du secteur 0 au secteur 27 -> $512 * 28 = 14336$ octets

FAT 16

F0 disquette 1.4 Mo
F9 disquette 1.2 Mo
F8 disque dur

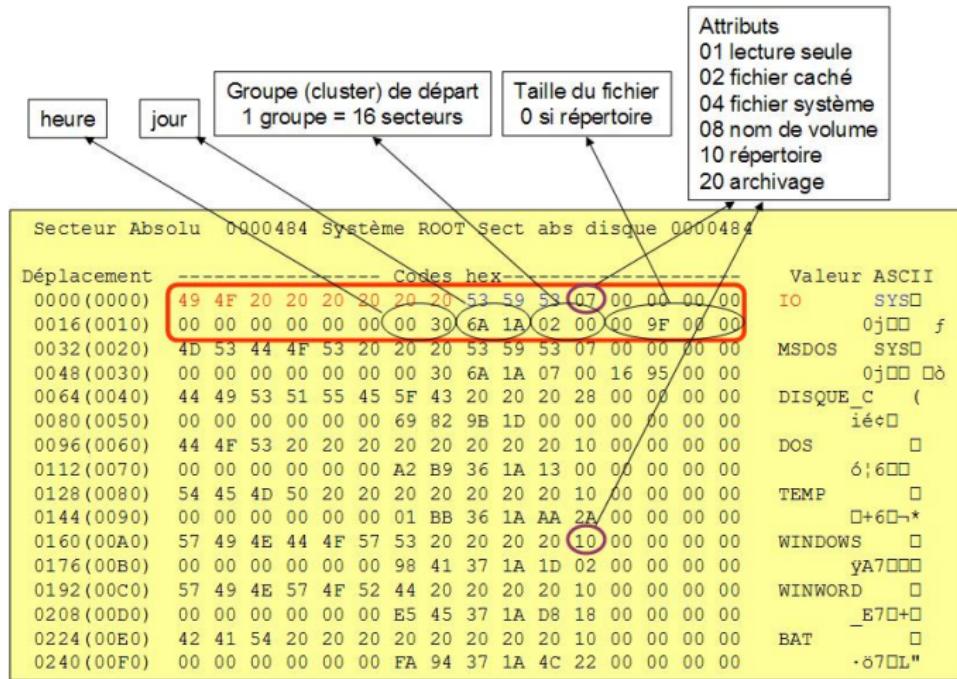
Secteur Absolu	0000028 Système FAT	Sect abs disque	0000028	
Déplacement	Codes hex			Valeur ASCII
0000(0000)	F8 FF FF FF 03 00 04 00 05 00 06 00 FF FF 08 00			° _ □ □ □ □ _ □
0016(0010)	09 00 0A 00 0B 00 FF FF 0D 00 0E 00 0F 00 10 00			□ — □ ✕ □
0032(0020)	11 00 12 00 FF FF FF FF FF FF FF FF FF 18 00			□ □ _____ □
0048(0030)	19 00 FF FF FF FF 1C 00 1D 00 FF FF 1F 00 FF FF			□ _____ □ □
0064(0040)	21 00 FF FF 23 00 24 00 25 00 26 00 FF FF FF FF			! _____ # \$ % & _____

FAT du secteur 28 au secteur 255 -> $512 * 228 = 116736$ octets

Secteur Absolu	0000256 Système FAT	Sect abs disque	0000256	
Déplacement	Codes hex			Valeur ASCII
0000(0000)	F8 FF FF FF 03 00 04 00 05 00 06 00 FF FF 08 00			° _ □ □ □ □ _ □
0016(0010)	09 00 0A 00 0B 00 FF FF 0D 00 0E 00 0F 00 10 00			□ — □ ✕ □
0032(0020)	11 00 12 00 FF FF FF FF FF FF FF FF FF 18 00			□ □ _____ □
0048(0030)	19 00 FF FF FF FF 1C 00 1D 00 FF FF 1F 00 FF FF			□ _____ □ □
0064(0040)	21 00 FF FF 23 00 24 00 25 00 26 00 FF FF FF FF			! _____ # \$ % & _____

copie de la FAT du secteur 256 au secteur 483 -> $512 * 228 = 116736$ octets

FAT 16



FAT 16

Secteur Absolu 0000484 Système ROOT Sect abs disque 0000484

Déplacement	Codes hex-	Valeur ASCII
0000(0000)	49 4F 20 20 20 20 20 20 53 59 53 07 00 00 00 00	IO SYS
0016(0010)	00 00 00 00 00 00 00 30 6A 1A 02 00 00 9F 00 00	0j□□ f
0032(0020)	4D 53 44 4F 53 20 20 20 53 59 53 07 00 00 00 00	MSDOS SYS□

H m s 0011 0000 0000 0000 0001 1010 0110 1010 AM J

+1980

Le volume dans le lecteur C s'appelle DISQUE_C
 Le numéro de série du volume est 19DB-1534
 Répertoire de C:\

IO	SYS				
MSDOS	SYS	40704	03-10-93	6:00a	
DOS		38166	03-10-93	6:00a	
TEMP		<REP>	01-22-93	11:13p	
WINDOWS		<REP>	01-23-93	8:12a	
WINWORD		<REP>	01-23-93	8:47a	
BAT		<REP>	01-23-93	6:39p	

FAT 16

Secteur Absolu 0000484 Système ROOT Sect abs disque 0000484

Déplacement ----- Codes hex ----- Valeur ASCII

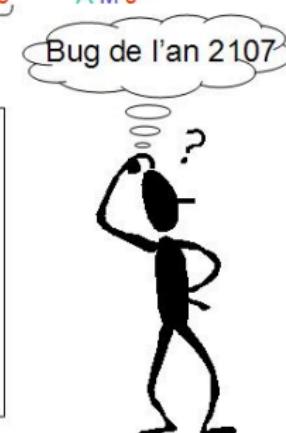
0000(0000)	49 4F 20 20 20 20 20 20 53 59 53 07 00 00 00 00	IO SYS
0016(0010)	00 00 00 00 00 00 00 30 6A 1A 02 00 00 9F 00 00	0j□ f
0032(0020)	4D 53 44 4F 53 20 20 20 53 59 53 07 00 00 00 00	MSDOS SYS□

H m s 0011 0000 0000 0000 0001 1010 0110 1010 +1980 AM J

Bug de l'an 2107

Le volume dans le lecteur C s'appelle DISQUE_C
Le numéro de série du volume est 19DB-1534
Répertoire de C:\

IO	SYS	40704	03-10-93	6:00a
MSDOS	SYS	38166	03-10-93	6:00a
DOS		<REP>	01-22-93	11:13p
TEMP		<REP>	01-22-93	11:24p
WINDOWS		<REP>	01-23-93	8:12a
WINWORD		<REP>	01-23-93	8:47a
BAT		<REP>	01-23-93	6:39p



FAT 16

répertoire															
0080(0050)	00	00	00	00	00	00	69	82	9B	1D	00	00	00	00	iéç□
0096(0060)	44	4F	53	20	20	20	20	20	20	20	10	00	00	00	DOS □
0112(0070)	00	00	00	00	00	00	A2	B9	36	1A	13	00	00	00	6;6□□
0128(0080)	54	45	4D	50	20	20	20	20	20	20	10	00	00	00	TEMP □
0144(0090)	00	00	00	00	00	00	01	BB	36	1A	AA	2A	00	00	□+6□→*

Fichier effacé															
Secteur Absolu	0000788	Groupe	00019	Sect abs disque	0000788										
Déplacement		Codes hex												Valeur ASCII	
0000(0000)	2E	20	20	20	20	20	20	20	20	20	10	00	00	00	.
0016(0010)	00	00	00	00	00	00	A2	B9	36	1A	13	00	00	00	6;6□□
0032(0020)	2E	2E	20	20	20	20	20	20	20	20	20	10	00	00	□
0048(0030)	00	00	00	00	00	00	A2	B9	36	1A	00	00	00	00	6;6□
0064(0040)	45	44	20	20	20	20	20	20	45	58	45	20	00	00	ED EXE
0080(0050)	00	00	00	00	00	00	00	00	00	00	B7	09	40	AE	00
0096(0060)	E5	43	35	30	31	20	20	20	53	59	53	20	00	00	_C501 SYS
0112(0070)	00	00	00	00	00	00	01	18	69	10	23	02	80	0D	00
0128(0080)	4D	4F	55	53	45	20	20	20	44	52	56	20	00	00	MOUSE DRV
0144(0090)	00	00	00	00	00	00	00	60	3C	15	D9	01	73	0E	00
0160(00A0)	4D	4F	55	53	45	20	20	20	53	59	53	20	00	00	MOUSE SYS

FAT 16 (état avant écriture d'un fichier)

Secteur Absolu 0000028 Système FAT Sect abs disque 0000028	
Déplacement	Codes hex----- Valeur ASCII
0256(0100)	81 00 82 00 83 00 84 00 85 00 86 00 87 00 88 00 ù é â à à å ç ê
0272(0110)	FF FF 8A 00 8B 00 8C 00 FF FF 8E 00 FF FF 90 00 è ï ï Ä É
0288(0120)	91 00 FF FF 93 00 FF FF 95 00 96 00 97 00 98 00 æ ô ð û ÿ
0304(0130)	99 00 9A 00 9B 00 9C 00 9D 00 FF FF 9F 00 FF FF ö Ü ç £ ¥ ù ÿ f
0320(0140)	A1 00 A2 00 A3 00 A4 00 A5 00 A6 00 FF FF A8 00 í ó ú ñ Ñ ¢
0336(0150)	FF FF AA 00 AB 00 AC 00 AD 00 AE 00 AF 00 B0 00 ¬ ¤ ¤ ; « »
0352(0160)	B1 00 B2 00 B3 00 B4 00 B5 00 B6 00 B7 00 B8 00 — — — — — —
0368(0170)	B9 00 BA 00 FF FF FF FF FF BE 00 BF 00 C0 00 : : + + + +
0384(0180)	C1 00 C2 00 FF FF C4 00 C5 00 C6 00 C7 00 FF FF - - - + ; -
0400(0190)	C9 00 CA 00 CB 00 FF FF CD 00 FF FF FF D0 00 + - - - - -
0416(01A0)	FF FF D2 00 FF FF FF FF D5 00 D6 00 FF FF 00 00 - - - + + -
0432(01B0)	00 00 00 00 00 00 FF FF C1 02 DE 00 DF 00 E0 00 — — — — — —
0448(01C0)	E1 00 E2 00 E3 00 E4 00 E5 00 E6 00 E7 00 E8 00 ß _ ¶ — —
0464(01D0)	E9 00 EA 00 EB 00 EC 00 ED 00 FF FF EF 00 FF FF — — — — — —
0480(01E0)	FF FF FF F3 00 F4 00 FE FF F6 00 F7 00 F8 00 — — — — — —
0496(01F0)	F9 00 FA 00 FB 00 FC 00 FD 00 FE 00 FF 00 00 01 • . n ¢ — —

Les premiers clusters libres

FAT 16 (création d'un fichier)

Secteur Absolu 0000486 Système ROOT Sect abs disque 0000486		Valeur ASCII
Déplacement	Codes hex	
0256(0100)	43 4F 4E 46 49 47 20 20 53 59 53 20 00 00 00 00	CONFIG SYS `ò+SEPa□
0257(0110)	00 00 00 00 00 00 00 00 00 00 00 00 50 61 01 00 00	DESKTOP INF□
Le volume dans le lecteur C s'appelle DISQUE_C		#qy□ □&
Le numéro de série du volume est 19DB-1534		AUTOEXECBAT □gô%òPç□
Répertoire de C:\		FONTEURO □ -â& & □
A23 BMP	360482 07-25-95 8:39a	A23 BMP ÷D•□+ "ç□
0384(0180)	41 32 33 20 20 20 20 20 42 4D 50 20 00 00 00 00	CONFIG VCT □Nè□òG-
0400(0190)	00 00 00 00 00 00 F6 44 F9 1E D7 00 22 80 05 00	AUTOEXECVCT □Nè□òPG+□
0416(01A0)	43 4F 4E 46 49 47 20 20 56 43 54 20 00 00 00 00	IOTA2CP UPD□ /u>&□
0432(01B0)	00 00 00 00 00 00 00 00 0E 4E 8A 1F 9B 47 02 00 00 00	
0448(01C0)	41 55 54 4F 45 58 45 43 56 43 54 20 00 00 00 00	
0464(01D0)	00 00 00 00 00 00 00 00 0E 4E 8A 1F 9E 47 B8 01 00 00	
0480(01E0)	49 4F 54 41 32 43 50 20 55 50 44 10 00 00 00 00	
0496(01F0)	00 00 00 00 00 00 00 2F 75 3E 26 04 1D 00 00 00 00	
$5*65536 + 8*4096 + 2*16 + 2 = 360482 \text{ octets}$		00D7*2=01AE

FAT 16 (après création du fichier)

Secteur Absolu 0000028 Système FAT Sect abs disque 0000028		
Déplacement	Codes hex	Valeur ASCII
0400(0190)	C9 00 CA 00 CB 00 FF FF CD 00 FF FF FF FF D0 00	+ - - - -
0416(01A0)	FF FF D2 00 FF FF FF FF D5 00 D6 00 FF FF DB 00	- - + + +
0432(01B0)	D9 00 DA 00 A9 02 FF FF C1 02 DE 00 DF 00 E0 00	+ + ☐ - ☐
0448(01C0)	E1 00 E2 00 E3 00 E4 00 E5 00 E6 00 E7 00 E8 00	ß ¶ µ - -

Secteur Absolu 0000030 Système FAT Sect abs disque 0000030		
Déplacement	Codes hex	Valeur ASCII
0320(0140)	A1 02 FF FF A3 02 A4 02 A5 02 A6 02 A7 02 A8 02	íº ñº ñº ñº ñº ñº
0336(0150)	FF FF AC 02 AB 02 FF FF BB 02 AE 02 6C 04 B0 02	¾ ¼ ¼ + ☺ ¼ ¼
0352(0160)	B1 02 B2 02 B3 02 B4 02 B5 02 B6 02 B7 02 B8 02	☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐
0368(0170)	B9 02 BA 02 FF FF BC 02 BD 02 C7 02 FF FF 30 04	; ; ; ; ; ; ; ; ; ;
0384(0180)	FF FF FF FF C3 02 C5 02 F5 02 FF FF C8 02 CB 02	+ + + + + + + +
0400(0190)	C9 02 FF FF CC 02 DB 02 CD 02 CE 02 CF 02 D1 02	+ + + + + + + +
0416(01A0)	8A 03 D2 02 D3 02 DA 02 D5 02 D6 02 FF FF FF FF	è ð - ð + + + +
0432(01B0)	FF FF FF FF E3 02 DC 02 3A 03 FF FF FF FF FF FF	¶ ☐ ☐ ☐ ☐ ☐ ☐
0448(01C0)	E1 02 E2 02 EE 03 E4 02 E5 02 E6 02 FF FF FF FF	ß ☐ ☐ ☐ ☐ ☐ ☐
0464(01D0)	FF FF EA 02 EC 02 FF FF FF FF 63 12 FF FF F0 02	☐ ☐ ☐ ☐ ☐ ☐ ☐

FAT 16 (après création du fichier)

Secteur Absolu 0000028 Système FAT Sect abs disque 0000028		
Déplacement	Codes hex	Valeur ASCII
0400(0190)	C9 00 CA 00 CB 00 FF FF CD 00 FF FF FF FF D0 00	+ - - + + -
0416(01A0)	FF FF D2 00 FF FF FF FF D5 00 D6 00 FF FF DB 00	- - + + + +
0432(01B0)	D9 00 DA 00 A9 02 FF FF C1 02 DE 00 DF 00 E0 00	+ + ☐ - ☐
0448(01C0)	E1 00 E2 00 E3 00 E4 00 E5 00 E6 00 E7 00 E8 00	ß ¶ µ

02A9*2=552 soit 2 secteurs (0200*2=0400 + 152)

Secteur Absolu 0000030 Système FAT Sect abs disque 0000030		
Déplacement	Codes hex	Valeur ASCII
0320(0140)	A1 02 FF FF A3 02 A4 02 A5 02 A6 02 A7 02 A8 02	í ñ ã ã ã ã ã ã
0336(0150)	FF FF AC 02 AB 02 FF FF BB 02 AE 02 6C 04 B0 02	ñ ã ã ã ã ã ã ã
0352(0160)	B1 02 B2 02 B3 02 B4 02 B5 02 B6 02 B7 02 B8 02	ó ã ã ã ã ã ã ã
0368(0170)	B9 02 BA 02 FF FF BC 02 BD 02 C7 02 FE FF 30 04	ó ã ã ã ã ã ã ã
0384(0180)	FF FF FF FF C3 02 C5 02 F5 02 FF FF C8 02 CB 02	þ þ þ þ þ þ þ þ
0400(0190)	C9 02 FF FF CC 02 DB 02 CD 02 CE 02 CF 02 D1 02	þ þ þ þ þ þ þ þ
0416(01A0)	8A 03 D2 02 D3 02 DA 02 D5 02 D6 02 FF FF FF FF	è ã ã ã ã ã ã ã
0432(01B0)	FF FF FF FF E3 02 DC 02 3A 03 FF FF FF FF FF	þ þ þ þ þ þ þ þ
0448(01C0)	E1 02 E2 02 EE 03 E4 02 E5 02 E6 02 FF FF FF FF	þ þ þ þ þ þ þ þ
0464(01D0)	FF FF EA 02 EC 02 FF FF FF FF 63 12 FF FF F0 02	c þ þ þ þ þ þ þ

Etc...

FAT 32 (File Allocation Table)

- Système utilisé par Windows
- Structure :
 - Secteur de boot (Master Boot Record) contenant la structure de l'unité physique (nombre de partition, taille, type, etc.)
 - La première table d'allocation de fichier
 - Une ou plusieurs copies de la table d'allocation
 - Un répertoire racine avec le nom de volume
 - La zone de données pour les fichiers et les sous répertoires
- Noms de fichiers à 255 caractères
- Préserve la casse mais n'en tient pas compte
- Protection des fichiers rudimentaire
- Volume de 128 To maximum

FAT 32 (File Allocation Table)

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
000000000	EB	58	90	4D	53	57	49	4E	34	2E	31	00	02	08	20	00	
000000010	02	00	00	00	00	F8	00	00	3F	00	FF	00	3F	00	00	00	
000000020	CB	94	60	00	20	18	00	00	00	00	00	00	02	00	00	00	
000000030	01	00	06	00	00	00	00	00	00	00	00	00	00	00	00	00	
000000040	80	00	29	F5	1A	3F	2D	4E	4F	20	4E	41	4D	45	20	20	
000000050	20	20	46	41	54	33	32	20	20	20	FA	33	C9	8E	D1	BC	
000000060	50	50	2F	01	DD	70	90	5F	76	00	1E	56	16	55	FF	22	
octets/secteur	02	00	soit	512					0B	FC	F	secteurs/FAT	00	00	18	20	soit 6176
secteurs/cluster	08	soit	8						88	45	F	réserve	(4 octets)				
secteurs de boot	00	20	soit	32								clusters de root	00	00	00	02	soit 2
nombre de FAT	02	soit	2									réserve	00	01	soit	1	
0 en FAT32	00	00										réserve	00	06	soit	6	
0 en FAT32	00	00										réserve	00	00			
support			F8	soit	disque	dur						signature	29				
0 en FAT32	00	00										identification	2D	3F	1A	F5	
secteurs/piste	00	3F	soit	63								nom du volume	11	octets	(ici NO NAME)		
nombre de têtes	00	FF	??									système de fichier	8	octets	(ici FAT32)		
secteurs cachés	00	00	00	3F	soit	63											
secteurs larges	00	06	94	CB													

JMP 58

FAT 32 (File Allocation Table)

OF attribut de nom long

006343760	7C 27 94 27 01 00 B8 53 7C 27 33 98 AD 59 00 00	'...S '3"-Y..
006343776	43 6D 00 6C 00 6F 00 6E 00 67 00 0F 00 84 00 00	Cm.l.o.n.g.....
006343792	FF FF FF FF FF FF FF FF 00 00 FF FF FF FF FF FF	yyyyyyyyyy.yyyy
006343808	02 70 00 6F 00 75 00 72 00 61 00 0F 00 84 76 00	.p.o.u.r.a...v.
006343824	1E 00 69 00 72 00 75 00 6E 00 00 00 6E 00 6F 00	.o.i.r.u.n...n.o.
006343840	01 6D 00 65 00 73 00 73 00 61 00 0F 00 84 67 00	.m.e.s.s.a...g.
006343856	65 00 69 00 6E 00 66 00 6F 00 00 00 64 00 65 00	e.i.n.f.o...d.e.
006343872	4D 45 53 53 41 47 7E 32 20 20 20 20 00 0A 0E 84	MESSAG~2.....
006343888	67 27 98 27 01 00 0F 84 67 27 5D 8B AA 16 00 00	g'"...g']<#...
006343904	E5 45 00 78 00 70 00 6C 00 6F 00 0F 00 17 69 00	àE.x.p.l.o....i.

MESSAG~2 5 802 07/11/99 16:32 messageinfodepouravoirunnomlong

13 caractères de type unicode par entrée

0060C730	75 27 75 27 01 00 CE 82 75 27 42 8D A5 48 00 00	u'u'...I,u'B+H..
0060C740	41 6D 00 65 00 73 00 73 00 61 00 0F 00 64 67 00	Am.e.s.s.a...dg.
0060C750	65 00 69 00 6E 00 66 00 6F 00 00 00 00 00 FF FF	e.i.n.f.o.....ÿ
0060C760	4D 45 53 53 41 47 7E 31 20 20 20 20 00 0A 0E 84	MESSAG~1.....
0060C770	67 27 98 27 01 00 0F 84 67 27 5D 8B AA 16 00 00	g'"...g']<#...
0060C780	41 61 00 6E 00 73 00 65 00 65 00 0F 00 C6 75 00	Aa.n.s.e.e...Eu.

FAT 32 (noms longs)

Offset	Length	Value
0	byte	Ordinal field
1	word	Unicode character 1
3	word	Unicode character 2
5	word	Unicode character 3
7	word	Unicode character 4
9	word	Unicode character 5
11	byte	Attribute
12	byte	Type (reserved; always 0)
13	byte	Checksum
14	word	Unicode character 6
16	word	Unicode character 7
18	word	Unicode character 8
20	word	Unicode character 9
22	word	Unicode character 10
24	word	Unicode character 11
26	word	Cluster (unused; always 0)
28	word	Unicode character 12
30	word	Unicode character 13

Nom long = 255 caractères

13 caractères par entrée
soit 20 entrées.

5 bits sont nécessaires au comptage
le bit 6 sert à marquer la dernière entrée

0100 0000

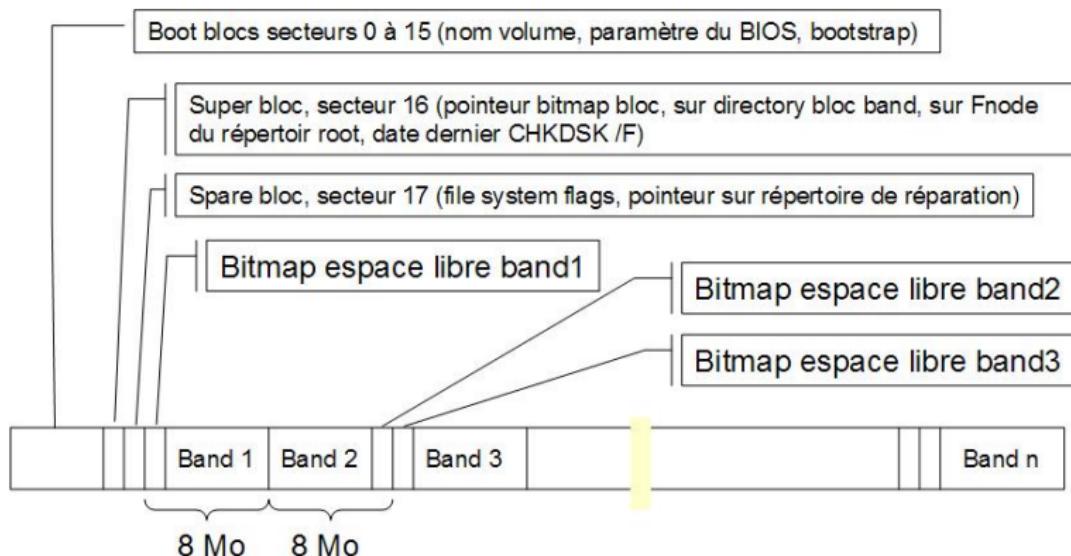
FAT16 et FAT 32 (protection des fichiers)

- Les seules informations de "protection" sont celles vues précédemment :
 - 01 lecture seule
 - 02 fichier caché
 - 04 fichier système
 - 08 nom de volume
 - 10 répertoire
 - 20 archivage
- Pas d'information concernant le "propriétaire" du fichier
- On peut tout obtenir et tout modifier au travers de la commande attrib

HPFS (High Performance File System)

- Système utilisé par OS/2
- Noms de fichier à 255 caractères
- Plus rapide que la FAT
- Attributs de protection
- Consomme 1Mo environ
- Volume limité à 8Go

HPFS (High Performance File System)



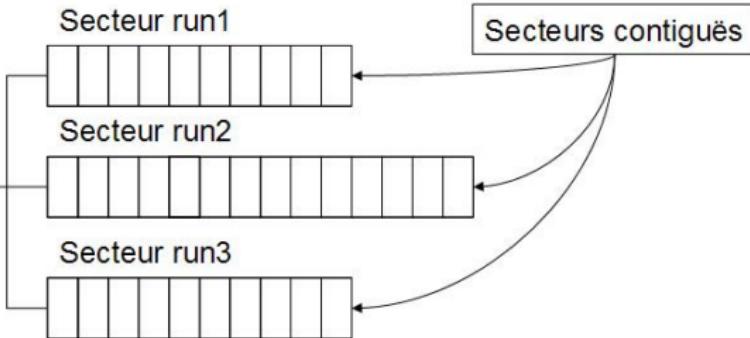
HPFS

Fnode

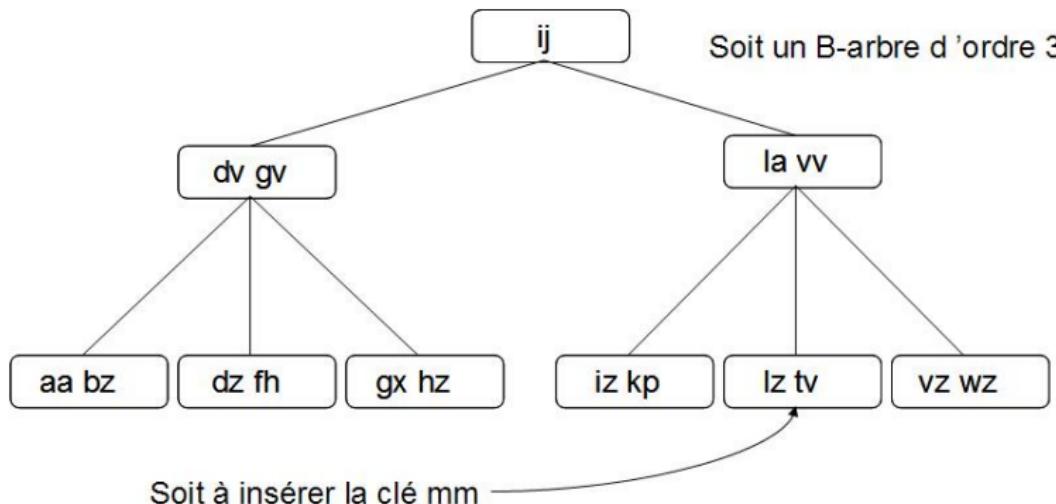
Entête
EA et ACL et/ou pointeurs de rangement pour EA et ACL
Nom tronqué
Allocation

Fnode contient:

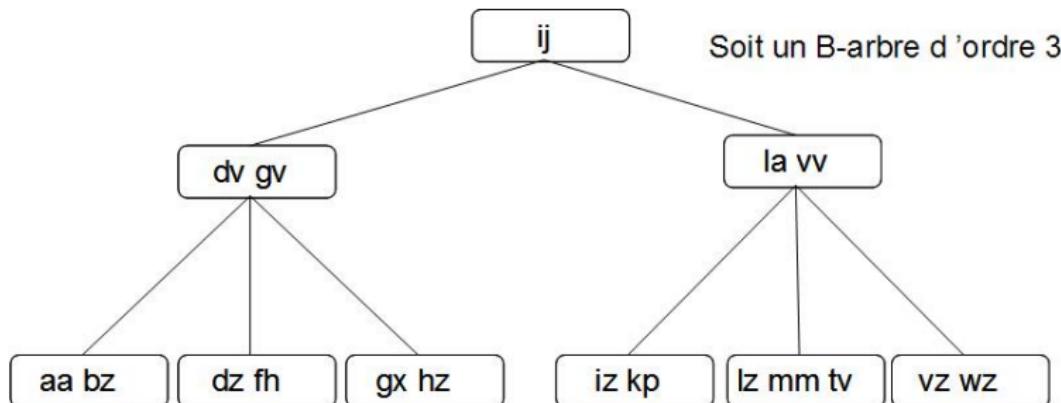
- un entête
- le nom du fichier tronqué à 15 caractères
- la longueur du fichier
- les attributs étendus (EA)
- la liste de contrôle d'accès (ACL)
- la localisation des données (B-arbre)



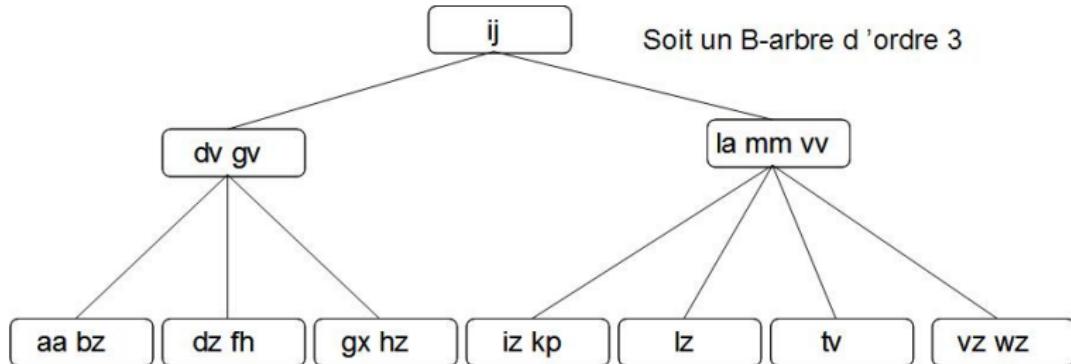
B-arbre (arbre-balancé)



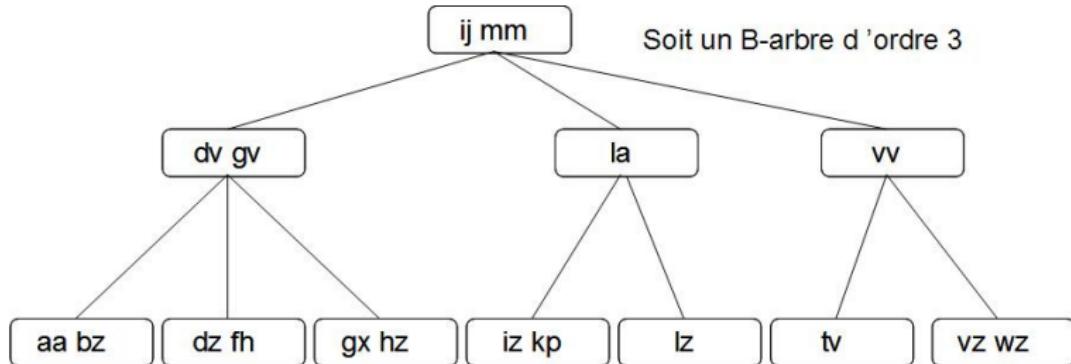
B-arbre (arbre-balancé)



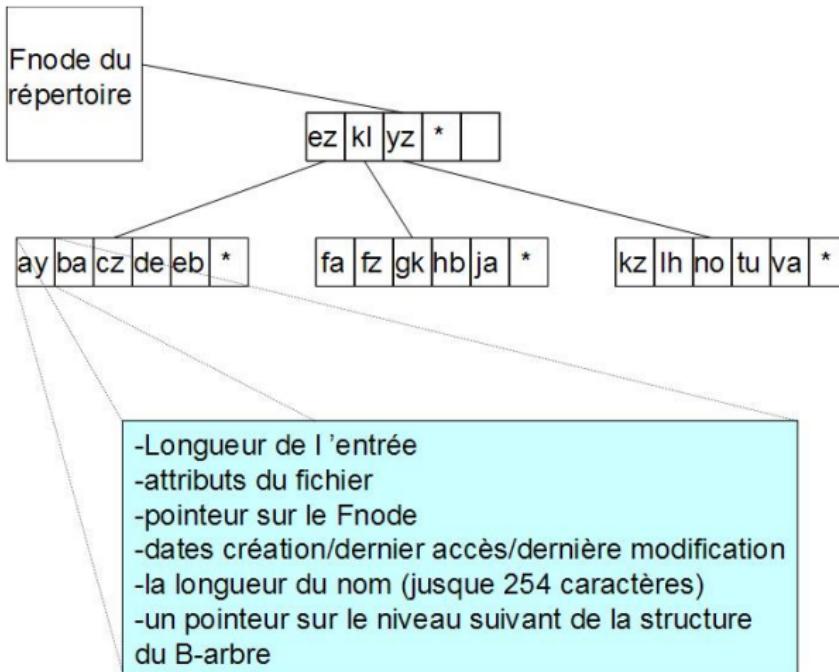
B-arbre (arbre-balancé)



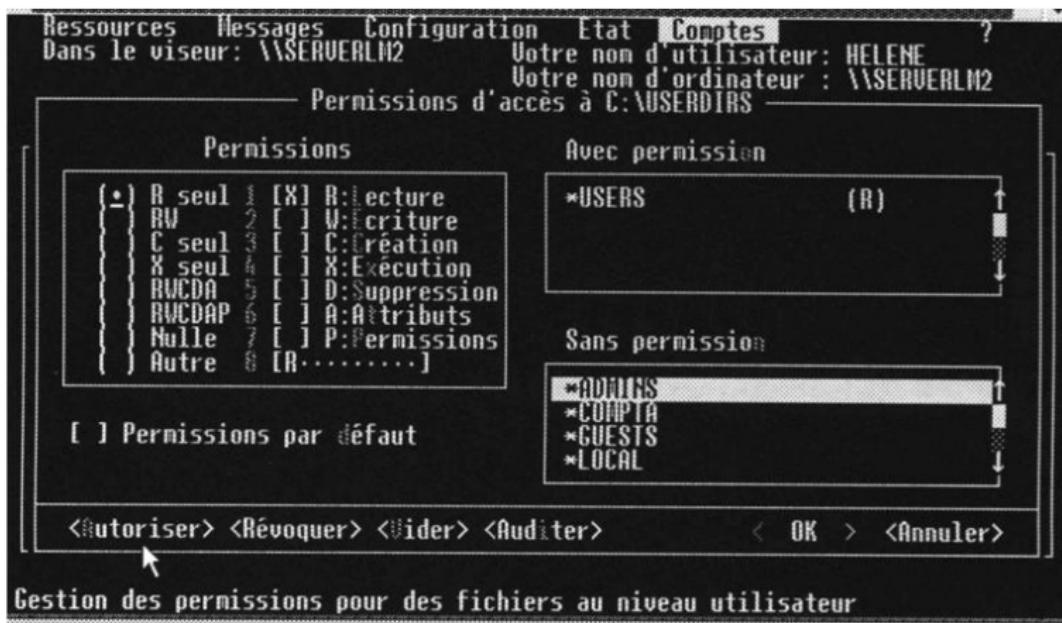
B-arbre (arbre-balancé)



HPFS



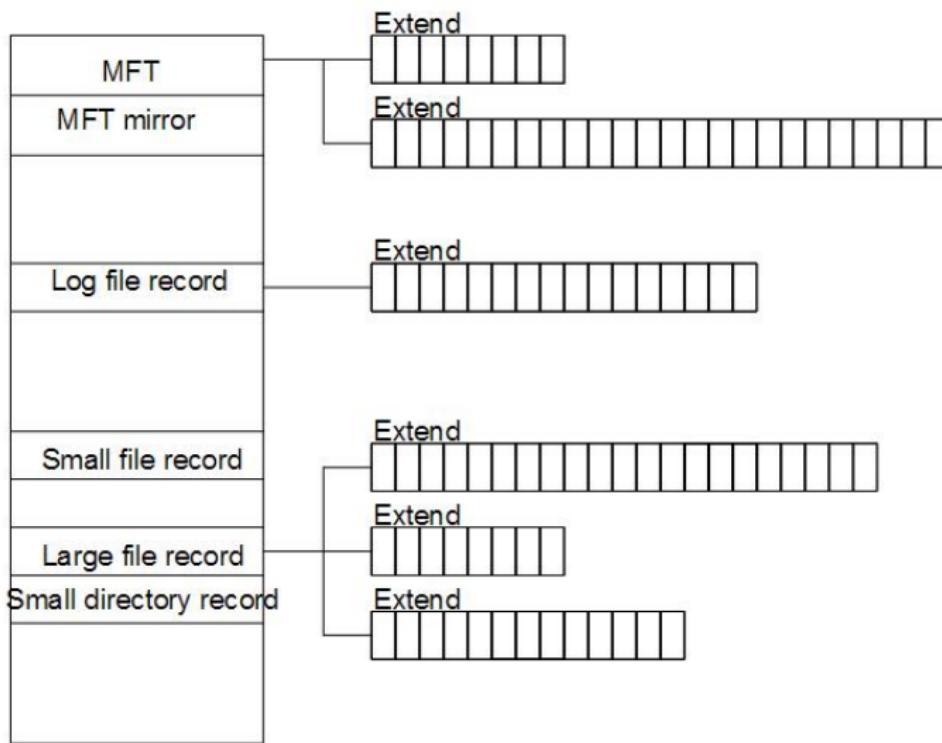
HPFS (droits sur les fichiers)



NTFS (New Technology File System)

- Système utilisé par Windows NT
- Attributs de protection
- Noms de fichier pouvant atteindre 256 caractères
- Préserve la casse mais n'en tient pas compte
- Consomme 4Mo environ
- Volume limité à 16Eo

NTFS



NFTS (boot)

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
000000000	EB	5B	90	4E	54	46	53	28	20	20	20	00	02	01	00	00	Ù[ÉNTFS .□..
000000010	00	00	00	00	00	00	F8	00	00	3F	00	80	00	3F	00	00°...?ç.?...
000000020	00	00	00	00	80	00	80	00	C0	F7	40	00	00	00	00	00ç.ç.+,@.....
000000030	25	1F	03	00	00	00	00	00	E0	7B	20	00	00	00	00	00	%□.....ó{
000000040	02	00	00	00	08	00	00	00	E6	2D	CB	0C	76	CB	0C	B8	□...□...u--□v-□@
000000050	00	00	00	00	00	00	00	00	00	00	00	00	00	FA	33	C03+
000000060	8E	D0	BC	00	7C	FB	B8	C0	07	8E	D8	C7	06	54	00	00	Äö+. ¹@+DÄiÄÖT..
000000070	00	C7	06	56	00	00	00	C7	06	5B	00	10	00	B8	00	0D	.ÄÖV...ÄÖ{.□.®.

octets/secteur	02 00 soit 512
secteurs/cluster	01 soit 1
0 en NTFS	00 00
0 en NTFS	00
0 en NTFS	00 00
0 en NTFS	00 00
support	F8 soit disque dur
0 en FAT32	00 00
secteurs/piste	00 3F soit 63
nombre de têtes	00 80 ??

Secteurs cachés	00 00 00 3F
inutilisés par NTFS	8 octets
Capacité du disque	00 40 F7 C0 soit 2Go
adresse MFT	00 03 1F 25
adresse MFT mirror	00 20 7B E0
MFT record size	00 00 00 02 soit 2 clusters
secteurs/bloc index	00 00 00 08
identificateur	B8 0C CB 76 0C CB 2D E6

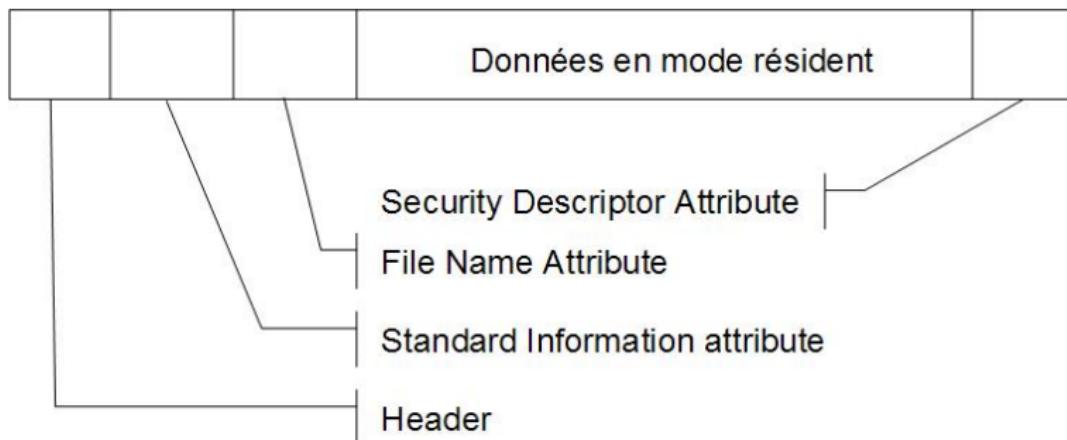
NFTS (MFT)

- table de 1024 octets
- une MFT par fichier ou répertoire

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
063E4A00	46	49	4C	45	2A	00	03	00	DB	8D	C1	0C	00	00	00	00	FILE*.□.¡i-□....
063E4A10	01	00	01	00	30	00	01	00	58	02	00	00	00	04	00	00	□.□.0.□.X□...□..
063E4A20	00	00	00	00	00	00	00	00	08	00	0B	01	54	00	00	00□. □T...
063E4A30	10	00	00	00	48	00	00	00	00	00	18	00	00	00	00	00	□...H.....□....
063E4A40	30	00	00	00	18	00	00	00	00	88	BD	C9	47	56	BD	01	0...□....ê¢+GV¢□
063E4A50	00	88	BD	C9	47	56	BD	01	00	88	BD	C9	47	56	BD	01	.ê¢+GV¢□.ê¢+GV¢□
063E4A60	00	88	BD	C9	47	56	BD	01	06	00	00	00	00	00	00	00	.ê¢+GV¢□□.
063E4A70	00	00	00	00	00	00	00	00	30	00	00	00	68	00	00	000...h...
063E4A80	00	00	18	00	00	00	03	00	4A	00	00	00	18	00	01	00	..□....□.J...□.□.
063E4A90	05	00	00	00	00	00	05	00	00	88	BD	C9	47	56	BD	01	□.....□.ê¢+GV¢□
063E4AA0	00	88	BD	C9	47	56	BD	01	00	88	BD	C9	47	56	BD	01	.ê¢+GV¢□.ê¢+GV¢□
063E4AB0	00	88	BD	C9	47	56	BD	01	00	D8	0F	00	00	00	00	00	.ê¢+GV¢□.Í¤.....
063E4AC0	00	D8	0F	00	00	00	00	00	06	00	00	00	00	00	00	00	.Í¤.....□....
063E4AD0	04	03	24	00	4D	00	46	00	54	00	00	00	00	00	00	00	□□\$.M.F.T.....
063E4AE0	50	00	00	00	80	00	00	00	00	00	18	00	00	00	04	00	P...ç.....□...□.

NTFS

- petit fichier
 - il est logé en totalité dans une MFT

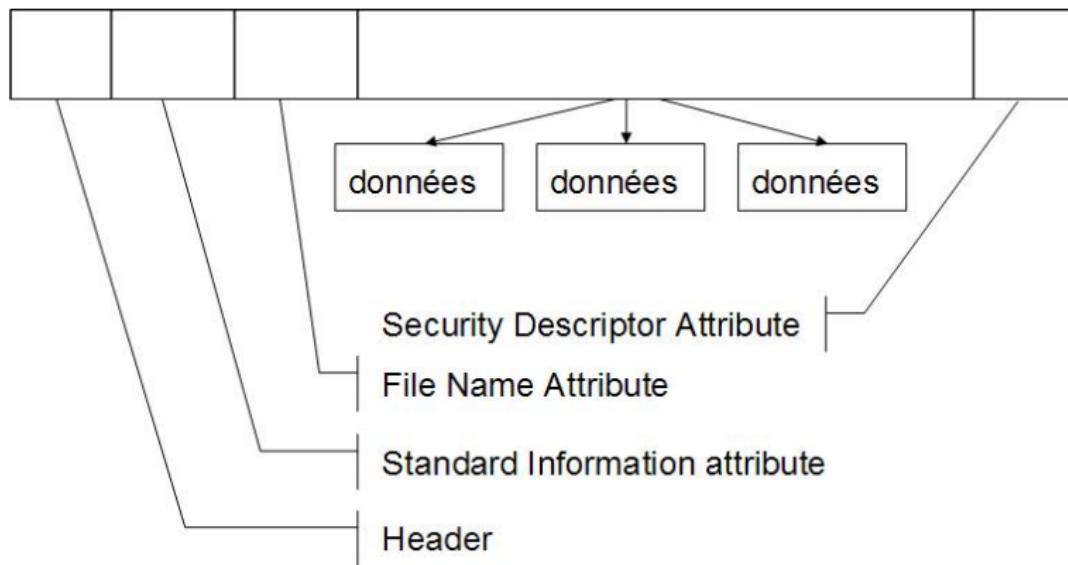


NTFS (petit fichier)

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
017A5000	46	49	4C	45	2A	00	03	00	ED	AC	B6	49	00	00	00	00	FILE*...i-&I....
017A5040	30	00	00	00	18	00	00	00	E0	AC	EE	A9	47	56	BD	01	0.....à-i@GV%
017A5050	B0	70	D5	FA	13	25	BE	01	F0	9E	A7	C2	DB	60	BF	01	°pÓú.%%.ØØSAÚ.;.
017A5060	40	D6	9F	EE	6F	9D	BF	01	25	00	00	00	00	00	00	00	ØÖÝio□.;.%.....
017A5070	00	00	00	00	00	00	00	00	30	00	00	00	70	00	00	000..p...
017A50D0	08	03	62	00	6F	00	6F	00	74	00	2E	00	69	00	6E	00	.b.o.o.t...i.n.
017A50E0	69	00	00	00	00	00	00	00	50	00	00	00	48	00	00	00	i.....P...H...
017A5140	21	01	00	00	18	00	00	00	5B	62	6F	6F	74	20	6C	6F	!.....[boot lo
017A5150	61	64	65	72	5D	0D	0A	74	69	6D	65	6F	75	74	3D	31	ader]..timeout=1
017A5160	30	0D	0A	64	65	66	61	75	6C	74	3D	6D	75	6C	74	69	0..default=multi
017A51D0	49	4E	4E	54	3D	22	57	69	6E	64	6F	77	73	20	4E	54	INNT="Windows NT
017A51E0	20	57	6F	72	6B	73	74	61	74	69	6F	6E	20	56	65	72	Workstation Ver
017A51F0	73	69	6F	6E	20	34	2E	30	30	22	20	0D	0A	6D	24	01	sion 4.00" ..m\$.

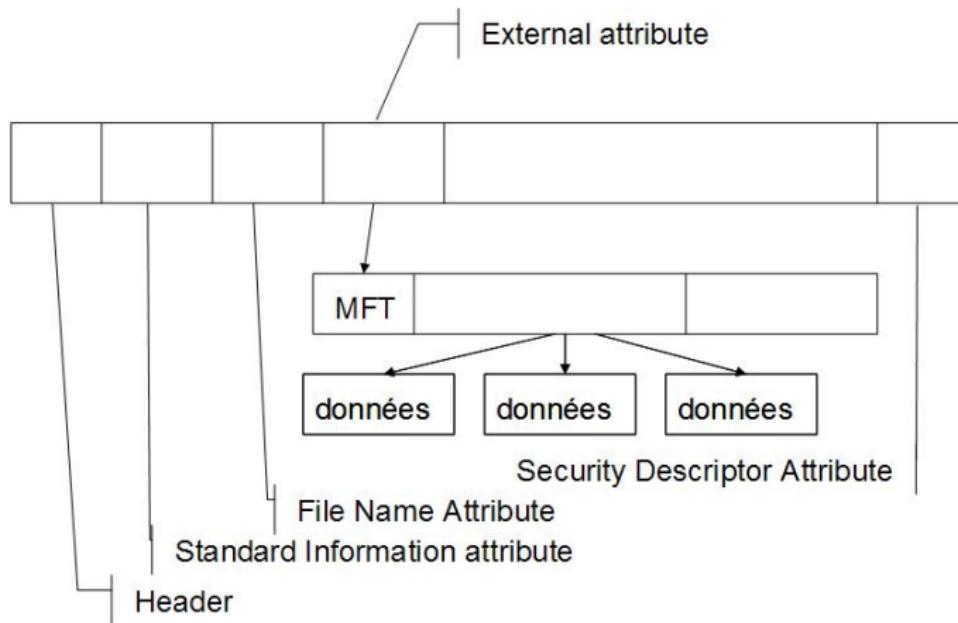
NTFS

- Fichier plus grand



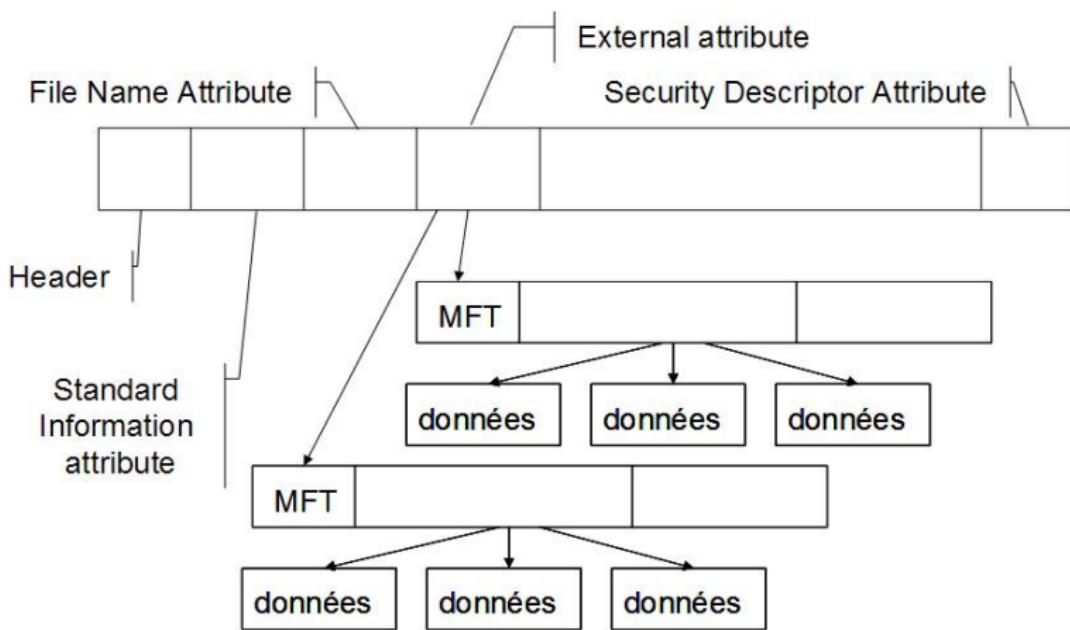
NTFS

- grand fichier



NTFS

- très grand fichier



NTFS (protection des fichiers)

- Droit d'accès sur les fichiers

Droit	R	X	W	D	P	O
Aucun accès	<input type="checkbox"/>					
Lire	<input type="checkbox"/>					
Modifier	<input type="checkbox"/>					
Contrôle total	<input type="checkbox"/>					
Accès spécial	<input type="checkbox"/>					

R= droit de lecture

X= droit d'exécution

W= droit d'écriture

D= droit de suppression

P= modification des droits

O= prend possession du fichier

NTFS (protection des fichiers)

- Droit d'accès sur les répertoires

Droit	R	X	W	D	P	O
Aucun accès	<input type="checkbox"/>					
Lister	<input type="checkbox"/>					
Lire	<input type="checkbox"/>					
Ajouter	<input type="checkbox"/>					
Ajouter et lire	<input type="checkbox"/>					
Modifier	<input type="checkbox"/>					
Contrôle total	<input type="checkbox"/>					
Accès spécial	<input type="checkbox"/>					

R= droit de lecture

X= droit d'exécution

W= droit d'écriture

D= droit de suppression

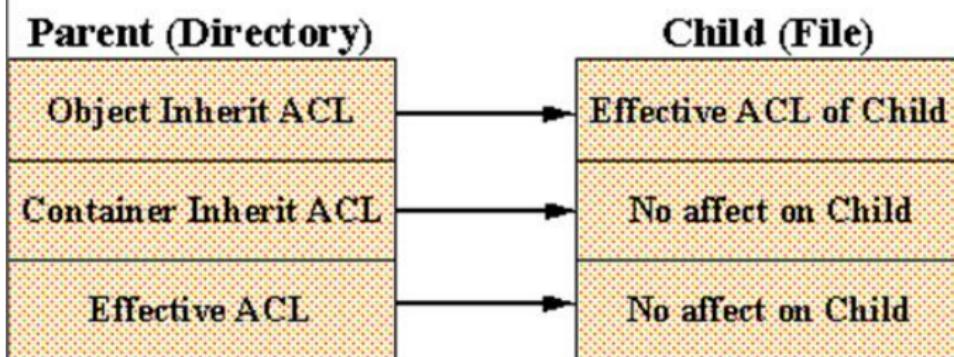
P= modification des droits

O= prend possession du répertoire

NTFS (protection des fichiers)

- Héritage des droits

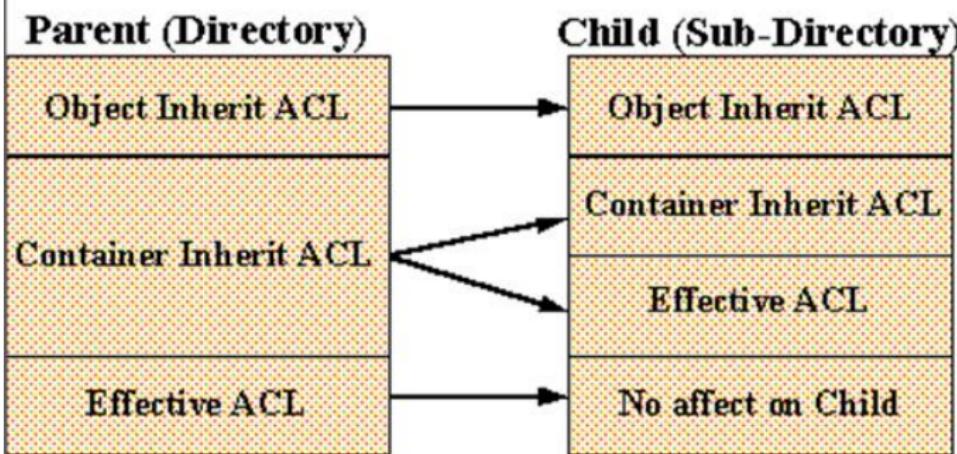
Access Control Inheritance Container to Non-Container Objects



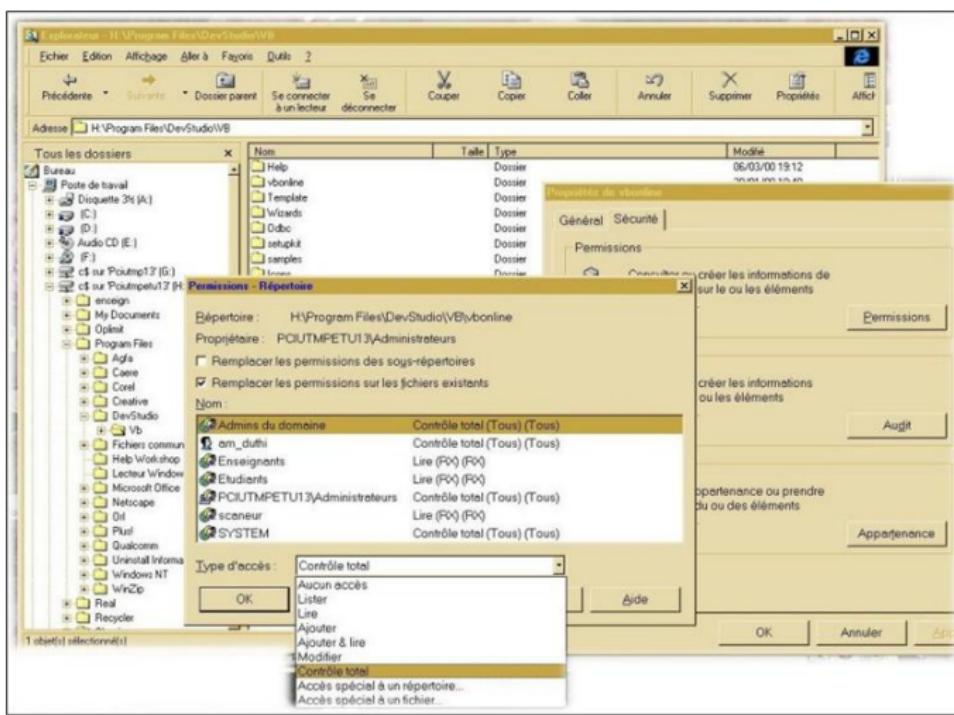
NTFS (protection des fichiers)

- Héritage des droits

Access Control Inheritance Container to Sub-Container Objects



NTFS



NTFS (commande)

**CACLS nom_de_fichier [/T] [/E] [/C] [/G util:perm] [/R util [...]]
[/P util:perm [...]] [/D util [...]]**

nom_de_fichier Affiche les ACLs.

/T Modifie les ACLs des fichiers spécifiés dans le répertoire courant et tous les sous-répertoires.

/E Edite l'ACL au lieu de la remplacer.

/C Continue la modification des ACLs, en ignorant les erreurs.

/G util:perm Donne à l'utilisateur spécifié les droits d'accès.

Perm peut être : R Lecture

C Modification (en écriture)

F Contrôle total

/R util Retire les droits d'accès de l'utilisateur spécifié (valide seulement avec /E).

/P util:perm Remplace les droits d'accès de l'utilisateur spécifié.

Perm peut être : N Aucun

R Lecture

C Modification (en écriture)

F Contrôle total

/D util Refuse l'accès à l'utilisateur spécifié.

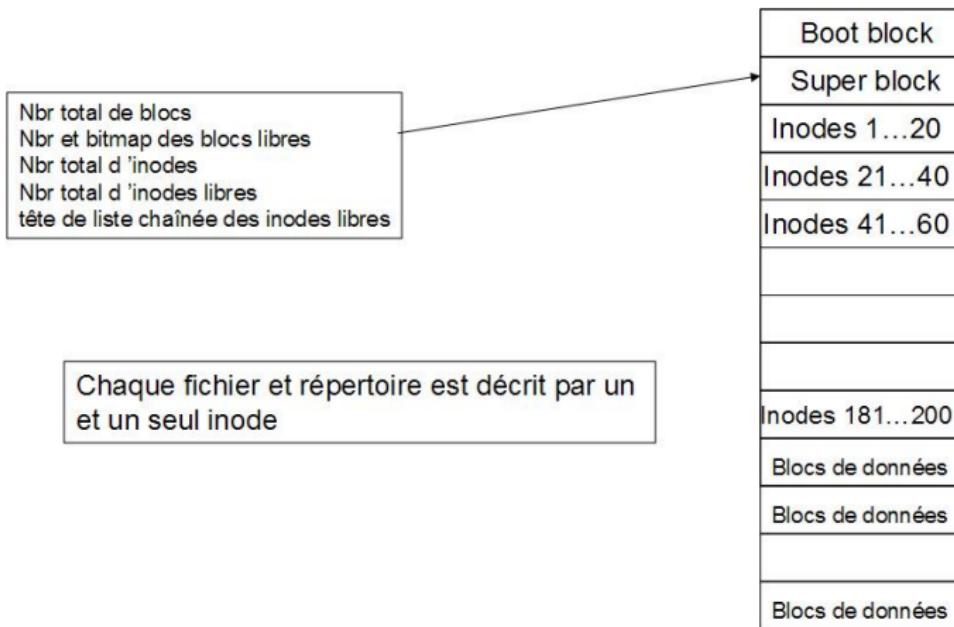
Des caractères génériques peuvent être utilisés pour spécifier plus d'un seul fichier dans une commande.

Vous pouvez spécifier plus d'un utilisateur dans une commande.

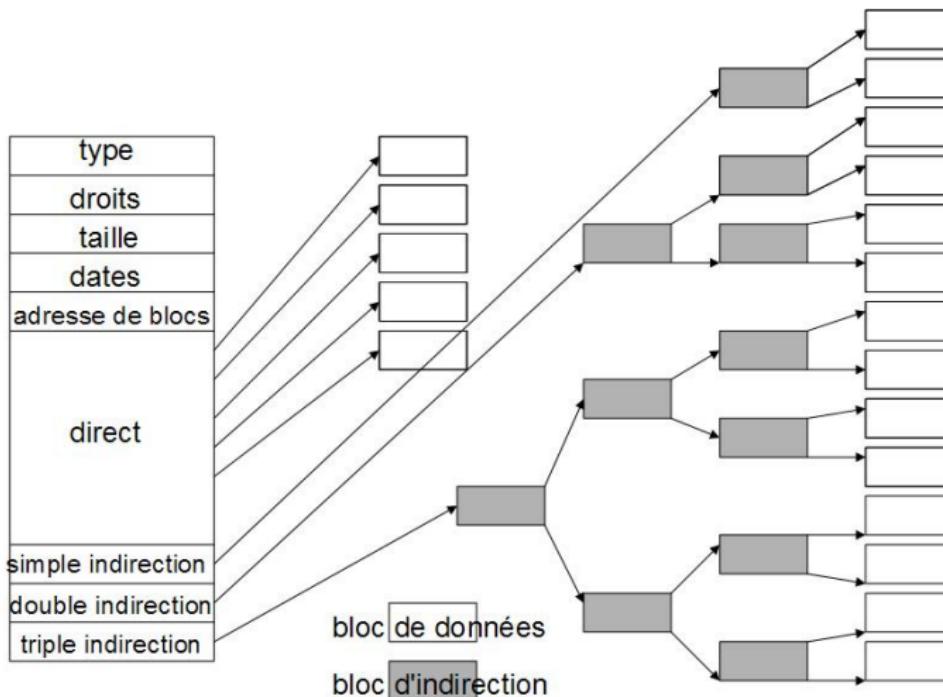
ext2fs (extended 2 File System)

- Système utilisé par Linux
- Attributs de protection
- Noms de fichiers longs
- Distinction entre majuscules et minuscules
- Taille maximum de 4 To

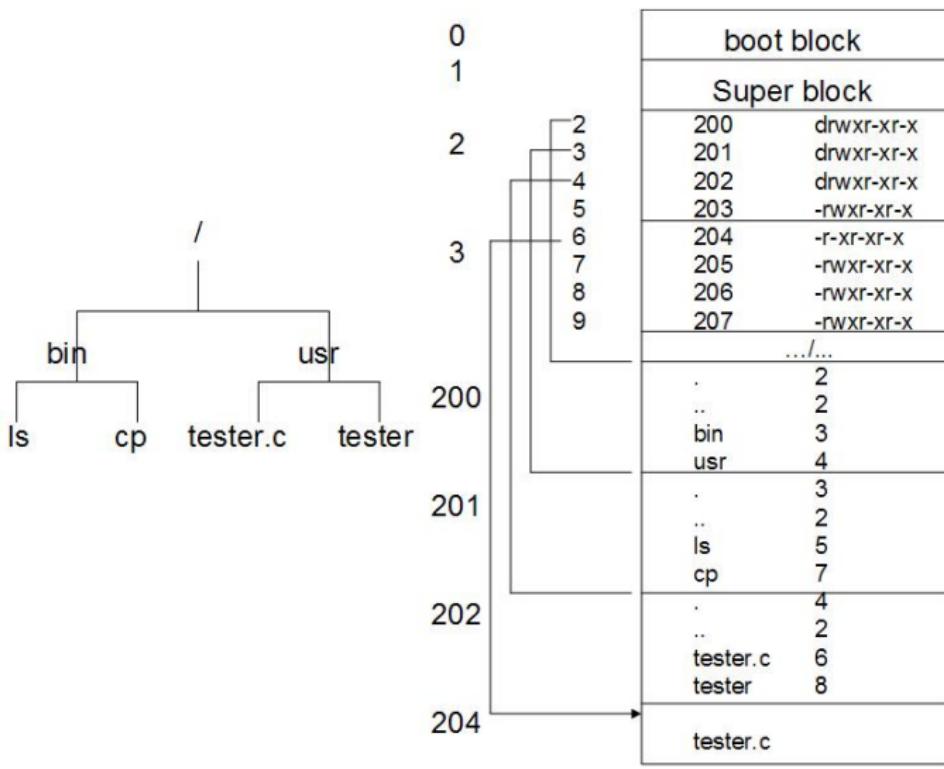
ext2fs (structure d'inode)



ext2fs (structure d'inode)



ext2fs (structure d'inode)



ext2fs (structure d'inode)

- Si on considère des blocs de 1 Ko et qu'une adresse de bloc occupe 4 octets, chaque bloc peut comporter 256 adresses
- Simple indirection :
 - 256 blocs de 1 Ko soit un fichier de 256 Ko maximum
- Double indirection :
 - 256 blocs de 256 adresses de blocs de 1 Ko = 64 Mo
- Triple indirection :
 - 256 blocs pointant sur 256 adresses de 256 adresses de 1 Ko soit un fichier de 16 Go de taille maximum

Journalisation

- ext2fs utilise un cache disque
 - Problème en cas d'arrêt "brutal" du disque
 - Usage de "fsck" au redémarrage : vérification de la totalité du disque => peut prendre des heures sur des gros disques
- Nouveau principe : la journalisation
 - Conservation de toutes les transactions dans un journal
 - En cas de problème : plus de nécessité de vérifier la totalité du disque

ext3fs

- Principe de fonctionnement identique à ext2fs (inode)
 - Nouveau problème : augmentation de la taille des disques
 - Cartographie des blocs libres et utilisés avec un parcours de cette carte
 - Plus le système de fichiers est grand, plus cette carte est grande => ralentissement
 - Plus le disque est "gros" => plus il y a des fichiers => plus la table est grande

ReiserFS

- Structure différente et incompatible avec ext2fs et ext3fs
- Utilise un B-arbre
- Table d'allocation sous forme arborescente
- Les feuilles sont des références vers les fichiers