Systèmes de fichiers

Le système d'exploitation doit permettre le stockage de l'information, avec les contraintes suivantes :

- Pouvoir enregistrer une très grande quantité d'information
- Pouvoir conserver les informations après la fin du processus qui les utilisent
- Permettre l'accès simultané à l'information pour deux processus différents

Ceci est réalisé à l'aide des fichiers, sur des disques ou autres supports externes. La partie du système d'exploitation qui s'occupe de la gestion des fichiers est appelée système de fichier (file system).

I – Le point de vue utilisateur

1) Noms de fichier

Tous les systèmes permettent des noms de fichiers comprenant entre 1 et 8 caractères, certains autorisent jusqu'à 255 caractères et permettent les caractères spéciaux. UNIX distinguent les caractères minuscules et majuscules, Windows non. Beaucoup de systèmes d'exploitation gèrent des noms de fichiers comprenant deux parties :

- La 1ère partie située avant le point
- La 2nde partie située après le point appelé extension qui indique le type du fichier

Sous UNIX, on peut avoir deux extensions ou plus, par exemple .c.z correspondra à l'extension source C compressé. Pour certains systèmes, comme UNIX ou MAC OS, l'extension n'est qu'une convention, pour d'autres, comme Windows, l'extension est reconnue et le programme correspondant lui est associé.

2) Structure des fichiers

Il existe trois structures possibles :

-Dans la première, la plus flexible, le système d'exploitation ne s'occupe pas du contenu, le fichier n'est qu'une suite d'octets. C'est le cas d'UNIX et de Windows.

-Dans la deuxième, le fichier est une suite d'enregistrements de longueur fixe, chaque enregistrement ayant une structure interne bien définie.
Une opération de lecture renvoie un enregistrement (de même une opération d'écriture écrit un enregistrement complet).

-Dans la troisième, la plus élaborée, le fichier est un arbre d'enregistrement. Les enregistrement peuvent être de longueur différente, mais possède tous une clé, située à la même position. L'arbre est trié en fonction des clés. On accède aux enregistrement en précisant la clé. Ce type de structure peut être utilisée par certains gros systèmes de traitement de données (le SGBD (Système de Gestion de Base de Donnée) fait alors aussi office d'OS).

3) Types de fichier

Il existe plusieurs types de fichiers :

- Les fichiers ordinaires : ils contiennent des programmes, des données de programmes, des fichiers texte, utilisateur ou système.
- Les répertoires : fichiers mémorisant la structure du système de fichier
- Les fichiers spéciaux : ils servent à modéliser les périphériques d'entrées/sorties. Il y a les fichiers spéciaux caractères qui modélisent les choses tel que l'imprimante, les fichiers spéciaux blocs qui modélisent les disques.

Il existe encore d'autres types de fichier. En général, un fichier ordinaire peut être un fichier ASCII (lignes de texte), ou un fichier binaire (seul le programme associé reconnaît leur structure interne). Les systèmes d'exploitation doivent connaître la structure de leurs propres fichiers exécutables.

On donne ci-dessous par exemple la structure d'un exécutable système UNIX :

Nombre magique (identifie un exécutable)
Taille du code
Taille des données
Taille BSS (Below Stack Segment)
Taille de la table des symboles
Point d'entrée (début des instructions)
/////// Espace libre \\\\\\\
Indicateurs
Code
Données
Bits de translation (relocation)
Table des symboles (debuggage)

4) Attributs des fichiers

En plus de son nom, à chaque fichier est associé un certain nombre d'informations, appelées attributs du fichiers, qui varient d'un système à l'autre.

Exemples d'attributs :

- De protection, qui peut accéder au fichier
- Mot de passe : pour pouvoir accéder au fichier
- Qui est propriétaire
- Indicateur de lecture seul : 0 lecture-écriture, 1 lecture seule
- Indicateur de fichier caché
- Indicateur de fichier système
- Indicateur d'archivage
- Indicateur de fichier ASCII et binaire
- Indicateur de fichier temporaire
- Date de création
- Date de dernier accès
- Taille courante

Pour les fichiers dont les enregistrements peuvent être accessibles avec une clé, on a aussi :

- Longueur d'enregistrement
- Position de la clé
- Longueur de la clé

5) Opérations sur les fichiers

Les principaux appels systèmes concernant les fichiers sont :

- Opération create : opération de création sans donnée, indique le nom et certain paramètre
- Opération delete : opération de destruction, libère la mémoire prise par l'opération create
- Opération open : opération d'ouverture, on charge les attributs et les adresses sur le disque
- Opération close : libère l'espace mémoire prise par open
- Opération seek : pour se placer à tel endroit du fichier
- Opération read : lire à partir de la position courante
- Opération write : écrire à partir de la position courante si en fin de fichier on augmente sa taille
- Opération get attribut : pour obtenir certains attributs
- Opération set attribut : pour modifier certains attributs

6) Répertoires

Les systèmes de fichiers ont en général des répertoires (directory) ou dossiers qui garde une trace des fichiers, ces répertoires sont eux-même des fichiers dans la plupart des systèmes. Il existe plusieurs niveaux de répertoires selon les systèmes. Systèmes à un seul niveau de répertoires : le répertoire racine contient tous les fichiers. Systèmes à deux niveaux de répertoires : le répertoire racine contient un répertoire système et un répertoire par utilisateur, ensuite tous les fichiers sont dans l'un des répertoires précédents (pas d'autre sous-répertoire). Systèmes à répertoire hiérarchique, c'est le système que nous connaissons sur les PC actuels. Dans ce cas, il faut pouvoir préciser les chemins d'accès. On distingue les chemins d'accès :

- Absolu, qui commence par la racine, par ex. sous UNIX : /user/joe/pgmC/exo1.c ; ou sous Windows : \user\joe\pgmC\exo1.c
- Relatif, qui partent du répertoire courant (répertoire de travail), par exemple si on travaille sous le répertoire /user/joe il suffit de préciser pgmC/exo1.c

Quelques appels systèmes concernant les répertoires :

- Create : qui crée un répertoire vide
- Delete : qui supprime un répertoire vide
- Open dir : charge certaine information concernant le répertoire
- Close dir : libère la place mémoire prise par open dir
- Read dir : qui renvoie la prochaine entrée d'un répertoire ouvert
- Link : qui permet à un fichier d'apparaître dans plusieurs répertoires
- Unlink : supprime une entrée du répertoire, si le fichier est présent dans un seul répertoire il va être supprimer de l'ensemble du fichier, si il est dans plusieurs répertoire seul l'entrée correspondant à l'entrée est effacée, les autres restent

II – Le point de vu concepteur

1) Implantation des fichiers

Il existe plusieurs façon d'implanter des fichiers sur un disque.

La plus simple est l'allocation contiguë : les fichiers sont stockés en entiers, les uns à la suite des autres. Elle permet de calculer facilement les adresses (il suffit de connaître l'adresse du premier bloc et le nombre de blocs) et la lecture des données est rapide puisque les informations sont proches physiquement. Par contre elle conduit à une fragmentation du disque et pose problème lorsque la taille d'un fichier augmente.

L'allocation par liste chaînée : consiste à implanter un fichier par blocs non nécessairement contigus, le premier mot de chaque bloc d'un même fichier servant de pointeur sur le bloc suivant (le reste du bloc contenant les données). Cela résout le problème de fragmentation (hormis la fragmentation interne du premier bloc). Par contre l'accès aléatoire devient lent car il faut lire le premier mot de chaque bloc précédent le bloc cherché.

Une alternative est l'allocation par liste chaînée avec table en mémoire, cette fois le pointeur de chaque bloc est mémorisé dans une table stockée en mémoire, appelée FAT (File Allocation Table) ou table d'allocation des fichiers. Elle se présente sous la forme suivante : supposons qu'un fichier A soit constitué de 4 blocs, situés respectivement dans l'ordre sur les blocs physiques 6, 2, 4, 9, et qu'un fichier B soit constitué de trois blocs situés sur 3, 9, 5, on obtiendra la table suivante :

Blocs Physiques	Chaînage
0	
1	
2	4
B => 3	9
4	8
5	-1
A => 6	2
7	
8	-1 (fin de fichier)
9	5
10	
...	

Il suffit de connaître le premier bloc correspondant au début du fichier, ensuite de parcourir le tableau jusqu'au bloc voulu, le nombre –1 indiquant la fin du fichier.

Il existe encore une autre méthode qui consiste à attribuer à chaque fichier une structure, appelée i-node (index node), nœud d'informations, qui mémorise les blocs qui lui sont alloués, elle peut se présenter sous la forme :

Attributs du fichier
Adresse disque du bloc 0
Adresse disque du bloc 1
Adresse disque du bloc 2
Adresse disque du bloc 3
Adresse disque du bloc 4
Adresse disque du bloc 5
Adresse disque d'un bloc de pointeur

La dernière case pouvant servir au cas où le fichier augmente et contient plus de blocs que d'entrées dans la table. L'avantage est que l'on peut se contenter de ne charger que les i-nodes des fichiers ouverts.
Le choix de la taille des blocs doit concilier les deux aspects suivants :

- Plus le bloc est gros plus il y a de fragmentation interne
- Plus le bloc est petit plus il en faut et plus la lecture nécessite de nombreux déplacement donc plus de temps

En général, la taille d'un bloc se situe entre 512 o et 8 Ko.

2) Implantation des répertoires

Pour ouvrir un fichier, le système d'exploitation utilise un chemin d'accès pour trouver l'entrée du répertoire dans lequel se trouve le fichier. Chaque entrée du répertoire correspond à un fichier. Le répertoire associe à chaque fichier les informations nécessaires pour le localiser (adresse disque du début de fichier pour l'allocation contiguë, numéro de 1er bloc pour les listes chaînées, ou encore numéro i-node). Selon les systèmes de fichiers, le répertoire peut aussi contenir les attributs de chaque fichier, ou juste le numéro d'i-node si celui-ci contient les attributs.

Il existe plusieurs façon de mémoriser les noms de fichiers : soit on réserve une partie fixe correspondant à la longueur maximale du nom de fichier, soit on réserve une partie fixe qui contient un pointeur sur le nom du fichier. Lorsque le répertoire contient un très grand nombre de fichiers, on peut utiliser une table de hachage, si cette table a n entrées, on associera à chaque nom de fichier un nombre compris entre 0 et n-1, si à la même entrée correspond plusieurs noms de fichiers, cette entrée pointe sur une liste chaînée contenant les informations sur les différents fichiers.
Exemple :
On suppose qu'on retient que la 1ère lettre du nom de fichier => 26 entrées dans la table.
On aura par exemple pour l'entrée n l'adresse 522.
Entrée n => 522
522 => must : pointeur suivant 538
attributs de must
538 => marguerite : pointeur 603
attributs de marguerite
603 => etc.

3) Fichiers partagés

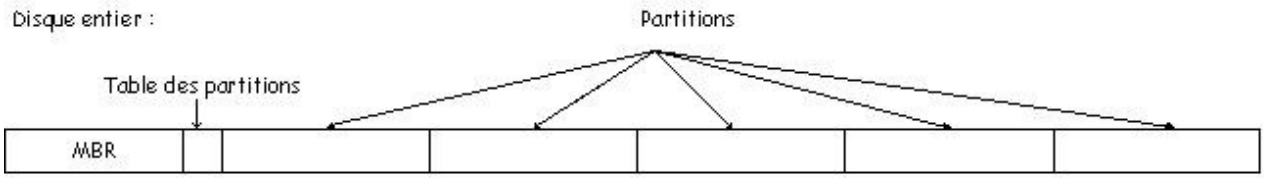
Il peut être nécessaire de pouvoir faire apparaître dans des répertoires différents un même fichiers, de façon à ce qu'il puisse être accessible et modifiable par deux utilisateurs par exemple. Une méthode consiste à créer un lien sur le fichier. On peut le faire de deux façons :

- Le lien physique (ou lien matériel) : on crée une entrée supplémentaire dans le répertoire avec le même numéro d'i-node que le fichier d'origine, mais un nouveau nom (si on demande la suppression du fichier d'origine, l'entrée correspondante est détruite, la nouvelle reste, par contre le propriétaire reste celui d'origine).
- Le lien symbolique : on crée un nouvel i-node, dans le répertoire, on associe au nouveau nom le chemin d'accès du fichier d'origine (il faut alors que le lien soit détruit si le fichier d'origine est détruit).

4) Organisation générale du disque

Les fichiers sont enregistrés sur des disques qui sont le plus souvent découpés en partitions, les systèmes de fichiers étant indépendant sur chaque partition. Le secteur 0 du disque est réservé au démarrage de la machine, il est appelé MBR (Master Boot Record), il contient la table des partitions (adresse début et fin de chaque partition). Toutes les partitions commencent par un bloc appelé bloc de boot. L'une d'elles, appelée partition active, contient le système d'exploitation qui sera chargé dans la mémoire centrale.

Ensuite l'organisation de la partition varie selon les système de fichiers. Il peut y avoir un super bloc contenant des indications sur le système de fichier (nombre de blocs, taille de différentes tables, …). On peut avoir une table contenant des informations sur les blocs libres, suivie du tableau des i-nodes avec une entrée par fichier, puis le répertoire racine et enfin les répertoires et fichiers.



Une partition :

Bloc de boot	Superbloc	Table blocs libres	Table i-nodes libres	Table i_nodes	Répertoires et Fichiers

La gestion de l'espace libre peut se faire suivant deux méthodes :

- La liste chaînée de blocs contenant chacun un numéro de bloc libre
- La table de bits, un par bloc, indiquant si le bloc est libre ou pas

5) Sauvegarde

Il existe plusieurs stratégies. On peut effectuer par exemple une copie intégrale hebdomadaire des données, et une copie incrémentable (uniquement les fichiers modifiés) journalière. En générale, on compresse préalablement les données (différents algorithmes de compression). La sauvegarde peut nécessiter une organisation très technique si le système de fichiers est toujours en activité.

On distingue les copies physiques (blocs successifs, problème blocs défectueux) et les copies logiques (commence par un répertoire et tous ses fichiers modifiés depuis une certaine date, etc.).

6) Cohérence

Il existe des utilitaires qui vérifient la cohérence des systèmes de fichiers, comme scandisk pour Windows, fsck pour UNIX. Ils s'appuient sur la redondance des informations. Par exemple fsck établit une liste des blocs utilisés et une liste des blocs libres, chaque liste contient un compteur pour chaque bloc qui sera incrémenté en parcourant les i-nodes pour la 1ère et le tableau de bits ou la liste des blocs libres pour la 2nde. Si le système est cohérent, on obtient :

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	N° de bloc
1	1	0	0	1	0	1	1	1	0	0	1	0	0	1	0	Blocs utilisés
0	0	1	1	0	1	0	0	0	1	1	0	1	1	0	1	Blocs libres

Si un des blocs a 0 dans les deux listes, il va être considéré comme bloc manquant et il sera ajouté aux blocs libres.
Si un de blocs à 1 dans les deux listes, il sera enlevé aux blocs libres.
Si un des blocs est marqué deux fois libre (compteur à 2 au lieu de 1), la liste des blocs libres sera modifié pour qu'il n'apparaissent qu'une seule fois.
Si un des blocs est marqué deux fois utilisé, le bloc sera recopié dans un bloc libre et attribué à l'un des fichiers, un message d'erreur préviendra l'utilisateur du système.