

Systeme Web PHP

Ryan Cassel
cassel@limsi.fr

Université Paris XI

Plan

● Système *4h*

● Web *2h*

● PHP *4h*

Plan

- **Systeme**

- Systeme d'exploitation

- Systeme de gestion de fichier

- Processus

Plan

- **Systeme d'exploitation**

- Couche fonctionnelle

- Architecture du systeme

- Architecture du noyau

Systeme d'exploitation

Qu'est ce qu'un systeme d'exploitation ?

C'est un allocateur et gestionnaire des ressources.

Systeme d'exploitation

- Ressources matérielles d'un ordinateur
 - unités centrales
 - mémoires
 - Périphériques des entrées/sorties
- Pour exécuter un programme il faut des ressources
- Les ressources sont limités
 - gestion des ressources

Systeme d'exploitation

○ Interface entre l'utilisateur et l'ordinateur

Systeme d'exploitation

Quelques systemes :

- Windows NT / XP / 2003 / ...
- Linux
- Mac OS
- UNIX
- Palm OS

Systeme d'exploitation

Couches fonctionnelles

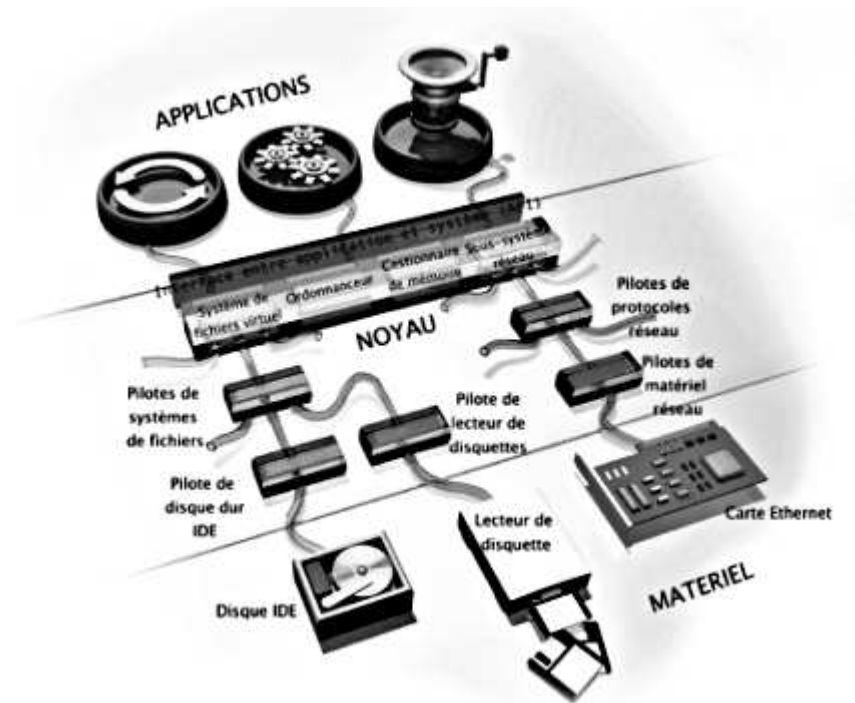
- Programmes utilisateurs
- Programmes d'application (éditeurs/tableurs/SGBD/CAO...)
- Programmes système (assembleurs/compilateurs...)
- Systeme d'exploitation
- Langage machine
- Microprogramme
- Machine physique

Systeme d'exploitation

Architecture du systeme

OSysteme par couches

- Couche utilisateur
- Couche systeme
- Couche materielle

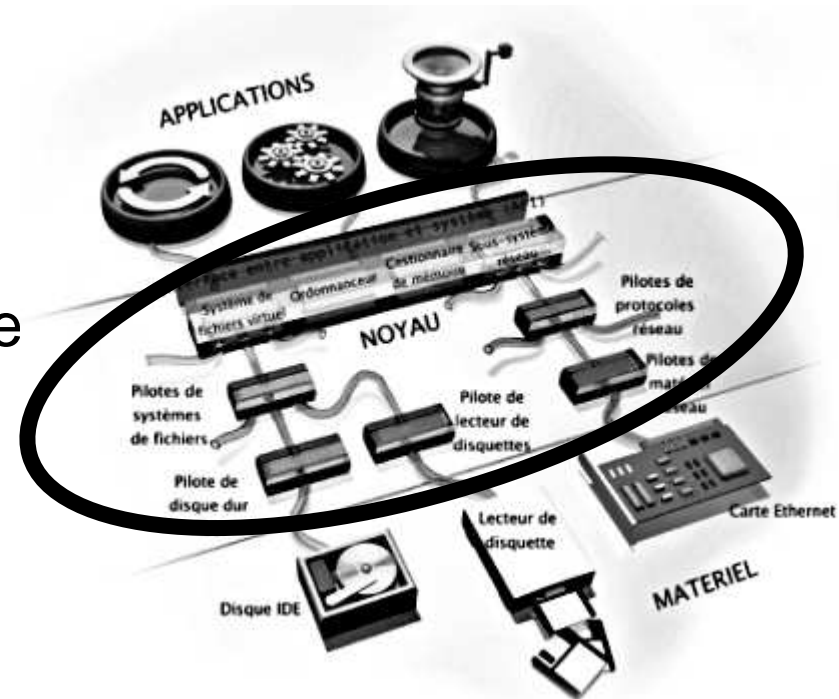


Systeme d'exploitation

Architecture du noyau (kernel)

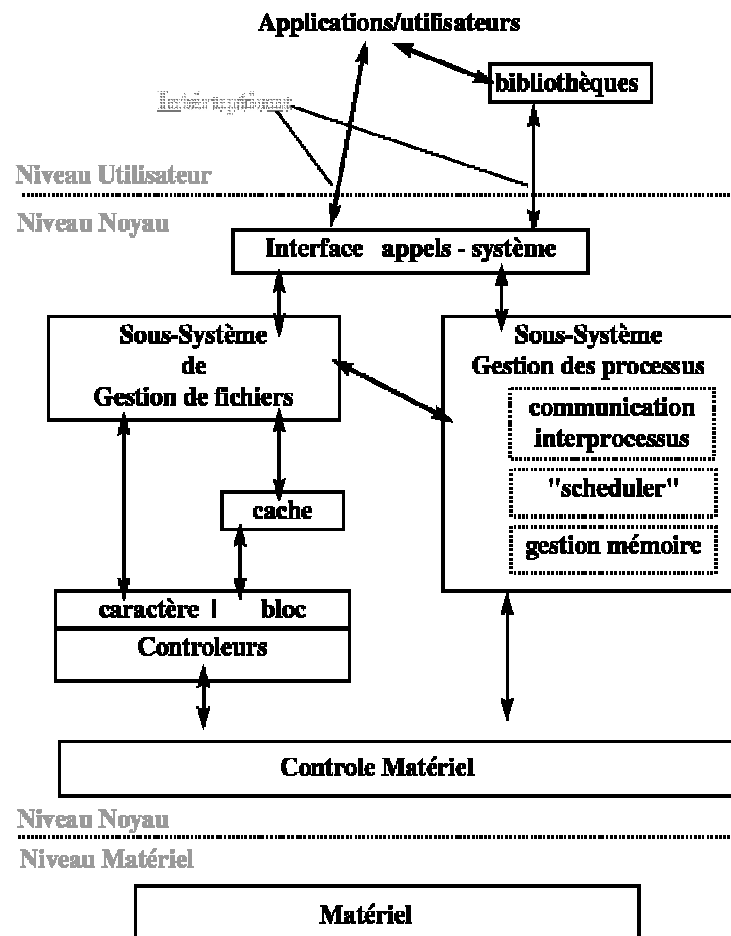
- Couche système

Interface entre le matériel et les applications



Systeme d'exploitation

Architecture du noyau (kernel)



Plan

Systeme de gestion de fichier (SGF)

Pourquoi un SGF ?

- Plusieurs raisons:
 - On veut pouvoir conserver de grandes quantités d'informations
 - On veut un stockage permanent → les données doivent être présentes même si la machine est mise hors tension
 - On veut partager les informations avec plusieurs utilisateurs
- Les fichiers permettent de satisfaire ces demandes
- C'est le rôle du SE de gérer les fichiers

Le concept de fichier

- Abstraction présentée à l'utilisateur
- Collection d'informations stockées sur une mémoire secondaire à laquelle on donne un nom.
- Le nom d'un fichier peut contenir son type
 - extensions dans UNIX et Windows par ex.
- Exemple de Types de fichiers
 - Fichiers multimédia, Répertoires
 - Fichiers exécutables
 - Fichiers spéciaux
 - Archives

Le concept de fichier

- Certains SE supportent et reconnaissent le type des fichiers:
 - Windows utilise les suffixes des noms de fichiers : .exe, .bat, .txt, .com...
 - Un type est associé à une application
 - .htm, .html -> netscape.exe
 - .pdf -> acroread.exe
 - MacOS utilise un typage présent au début du fichier
 - Il est formé de 4 lettres pour l'application et de 4 lettres pour le type
 - Unix ne supporte pas le typage des fichiers
 - Cependant l'utilitaire file peut donner le type

Types de fichiers et extensions

Type de fichier	Extension	Caractéristiques
Executable	exe, com, bin	Fichier executable → programme
Objet	obj, o	Résultat d'une compilation sans édition de liens
Code Source	c, cpp, pas, java, asm, a	Code source dans plusieurs langages
Traitement par lots (batch)	bat, sh	Commandes pour un interpréteur de commandes (shell)
Multimédia	mpeg, mpg, mp3, avi	Fichiers contenant des informations audio et/ou vidéo
Document d'un Traitement de Texte	doc, tex, rtf, etc.	Différents format utilisés par des applications.
Bibliothèque	lib, a	Bibliothèque de routines
Impression ou vue	ps, pdf, dvi, gif, jpg, bmp	Fichiers dans un format d'impression ou de visualisation
Archive	gz, arc, zip, tar, bz2	Regroupement de plusieurs fichiers, parfois compressés.

Structure d'un fichier

- *Aucune* – seulement une suite d'octets
(Unix, MS-DOS, ...)
- *Enregistrements simples*
 - Lignes de longueur fixée ou variable
- *Structure complexe*
 - Document formaté, Document multi-media
- De la responsabilité:
 - Du système d'exploitation
 - De l'application
 - D'un SGBD

Les attributs de fichiers

- **Nom** – information à destination de l'utilisateur → doit être lisible
- **Type** – nécessaire pour les systèmes qui supportent différents types.
- **Location** – pointeur vers l'adresse du fichier sur le support physique.
- **Taille** – taille du fichier
- **Protection** – contrôle qui peut lire, écrire ou exécuter un fichier.
- **Temps, date, et identification de l'utilisateur** – données pour la protection, la sécurité et le suivi de l'utilisation des fichiers.
- Les informations sur les fichiers sont conservées dans une structure spéciale sur le disque.

Opérations sur les fichiers

- Création - create
- Ecriture - write
- Lecture - read
- Déplacement – seek
- Suppression - delete
- Troncature - truncate
- open(filename) – ouvre le fichier dont le nom est “filename” et déplace son contenu en mémoire.
- close (filename) – fermeture du fichier → n’est plus accessible en mémoire.

Méthodes d'accès

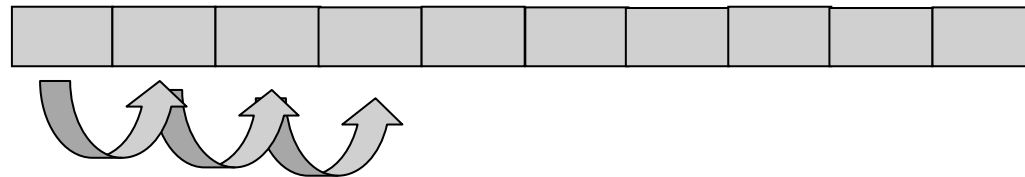
2 type d'accès:

- Accès séquentiel
- Accès direct (aléatoire : random access)

Méthodes d'accès

- Accès séquentiel

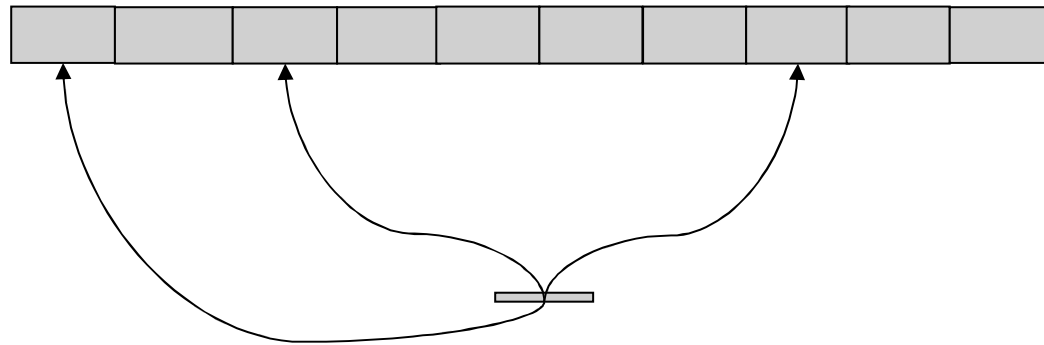
- Lecture ou écriture des éléments dans l'ordre avec possibilité de revenir au début.



- Adapté aux supports de stockage séquentiels: bandes magnétiques

Méthodes d'accès

- Accès direct (aléatoire : random access)
 - Lecture ou écriture à n'importe quel endroit



- Adapté aux supports de stockage à accès direct (disques)

Les répertoires

- Objectif

- Organiser les fichiers

- Plusieurs organisations

- Répertoire à un niveau

- Structure de répertoire à deux niveaux

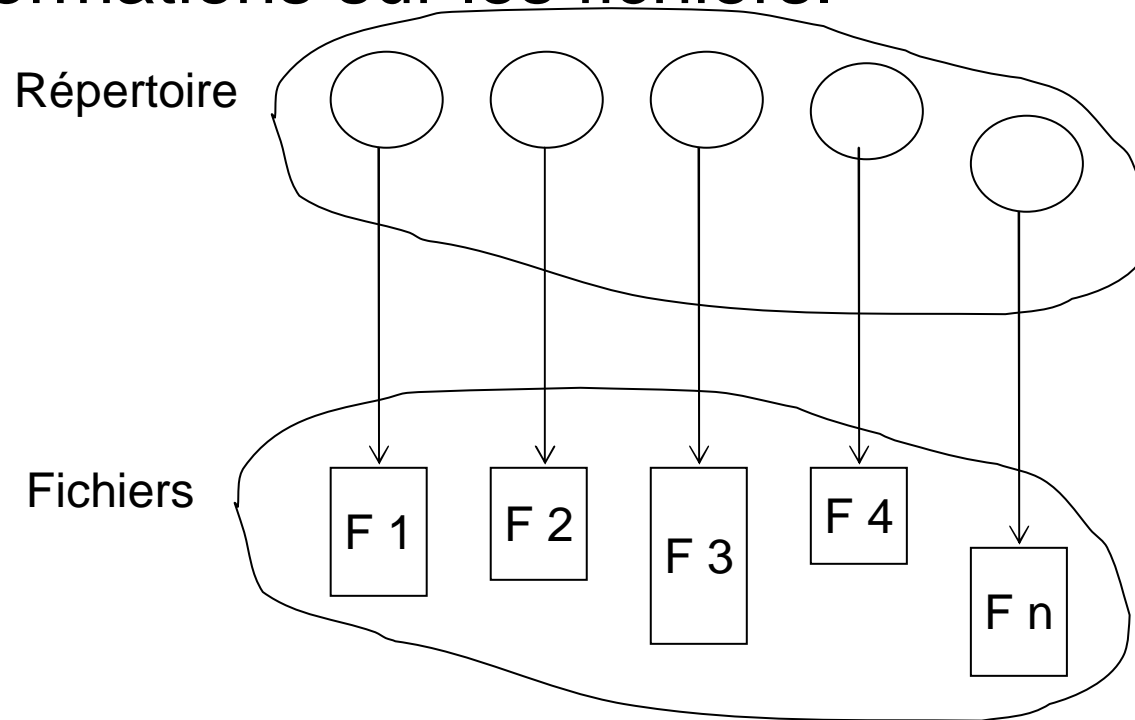
- Organisation arborescente (Unix, Ms-dos)

- Opérations

- Parcours, listage, renommage, ...

Structure de répertoire

- Une collection de noeuds contenant des informations sur les fichiers.



La structure de répertoire et les fichiers résident sur le disque.

Répertoire: informations

- Nom
- Type
- Adresse physique
- Longueur actuelle
- Longueur maximale
- Date de dernier accès
- Date de dernière mise à jour
- Identifiant du Propriétaire
- Protection

Opérations sur les répertoires

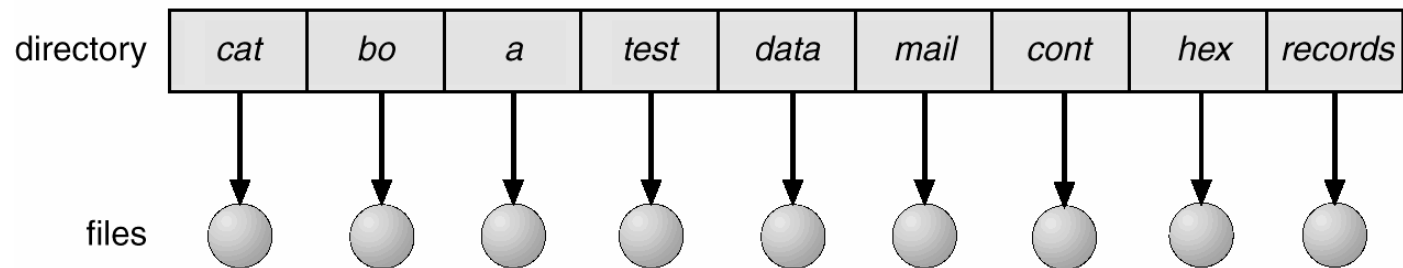
- Rechercher un fichier
- Créer un fichier
- Supprimer un fichier
- Renommer un fichier
- Afficher le contenu d'un répertoire
- Parcourir les sous-répertoires

Organisation logique

- Efficacité – pour localiser rapidement n'importe quel fichier.
- Nommage – très utile pour les utilisateurs
 - Deux utilisateurs peuvent donner un même nom à des fichiers différents.
 - Un même fichier peut avoir plusieurs noms.
- Regroupement – en fonction des caractéristiques des fichiers (sources, jeux, applications, ...)

Répertoire à un seul niveau

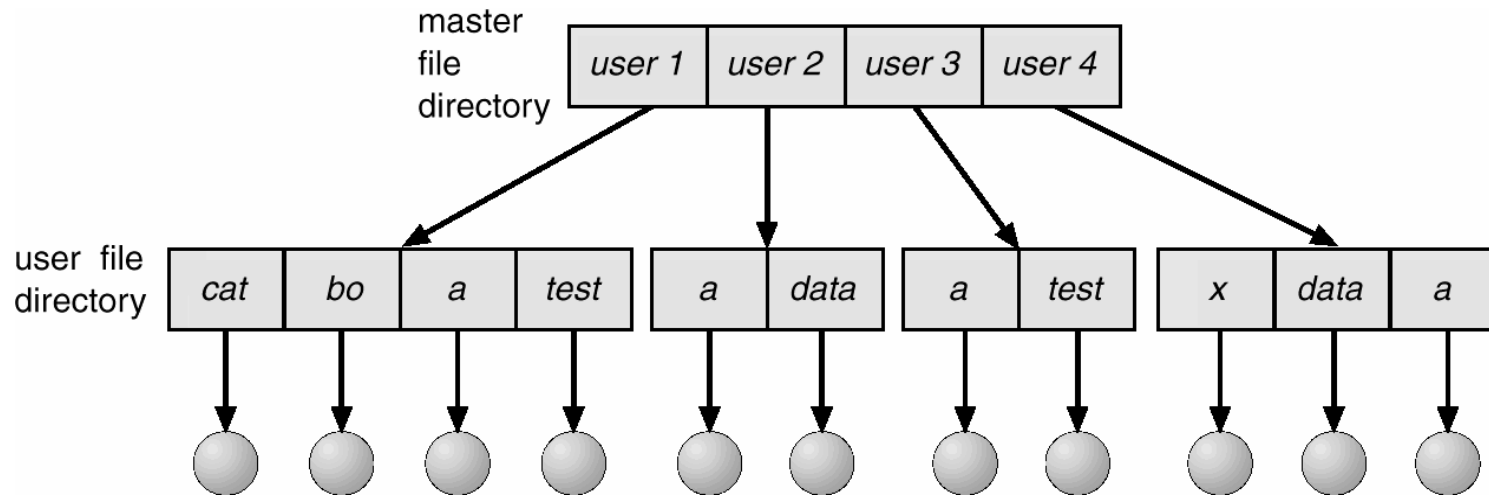
- Un seul niveau de répertoire pour tout le monde



- Problème pour nommer les fichiers
- Problèmes pour les regrouper

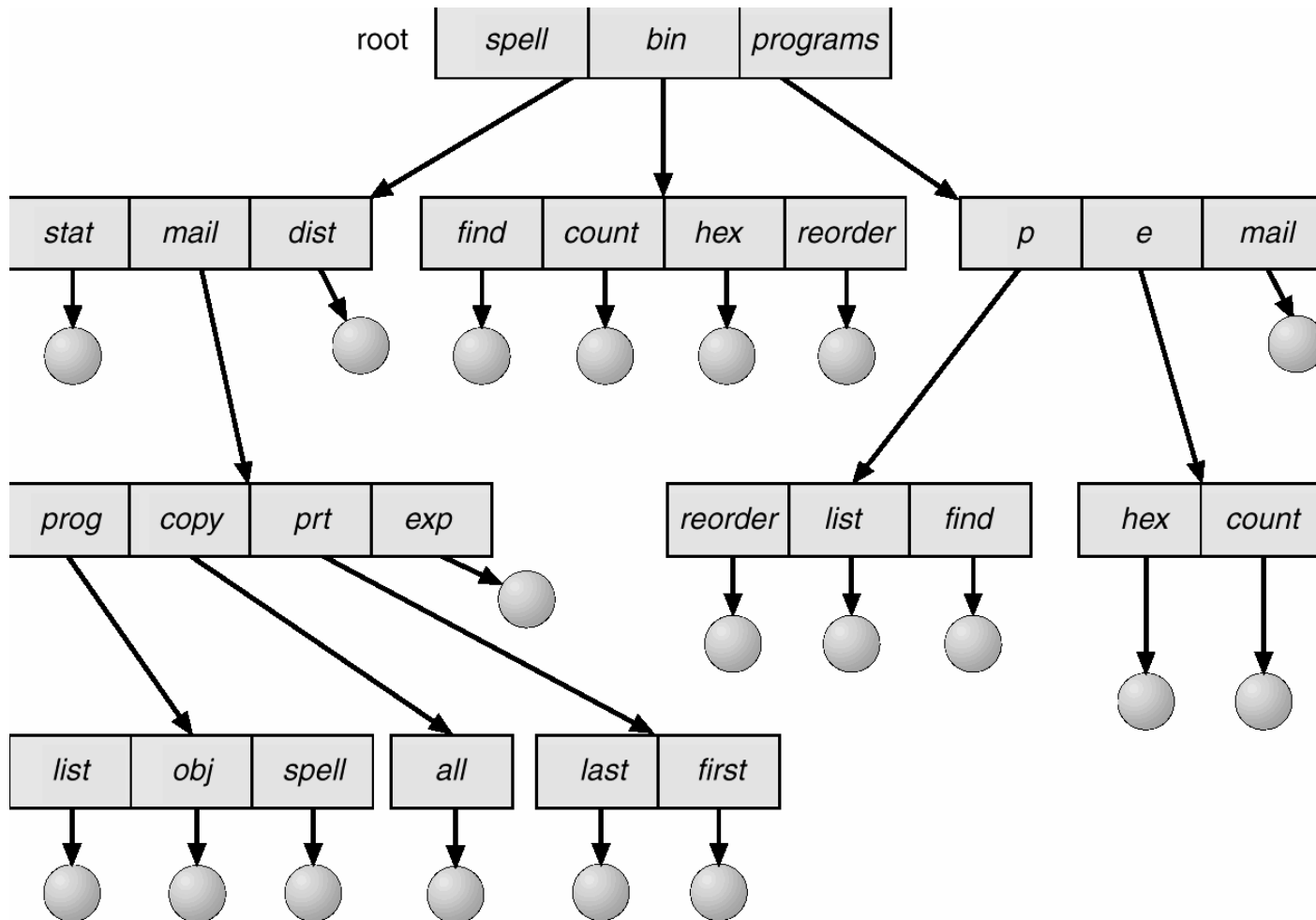
Répertoires à 2 niveaux

- Un répertoire pour chaque utilisateur



- Nom de chemin
- Des utilisateurs distincts peuvent avoir des fichiers portant le même nom
- La recherche d'un fichier est efficace

Répertoires à structure d'arbre



Répertoires à structure d'arbre

- Recherche de fichiers efficace
- Permet des regroupements
- Répertoire courant (répertoire de travail)

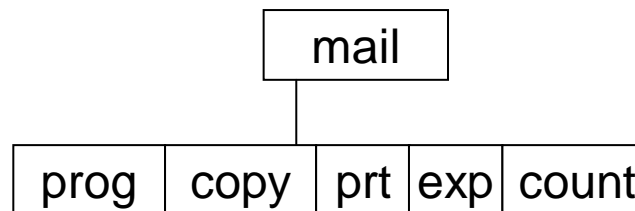
Ocd ~

Répertoires à structure d'arbre

- Chemin **relatif** ou **absolu**
- La création d'un nouveau fichier se fait dans le répertoire courant.
- Suppression d'un fichier
rm <file-name>
- Création d'un sous-répertoire dans le répertoire courant.
mkdir <dir-name>

Exemple: si on est dans /spell/mail

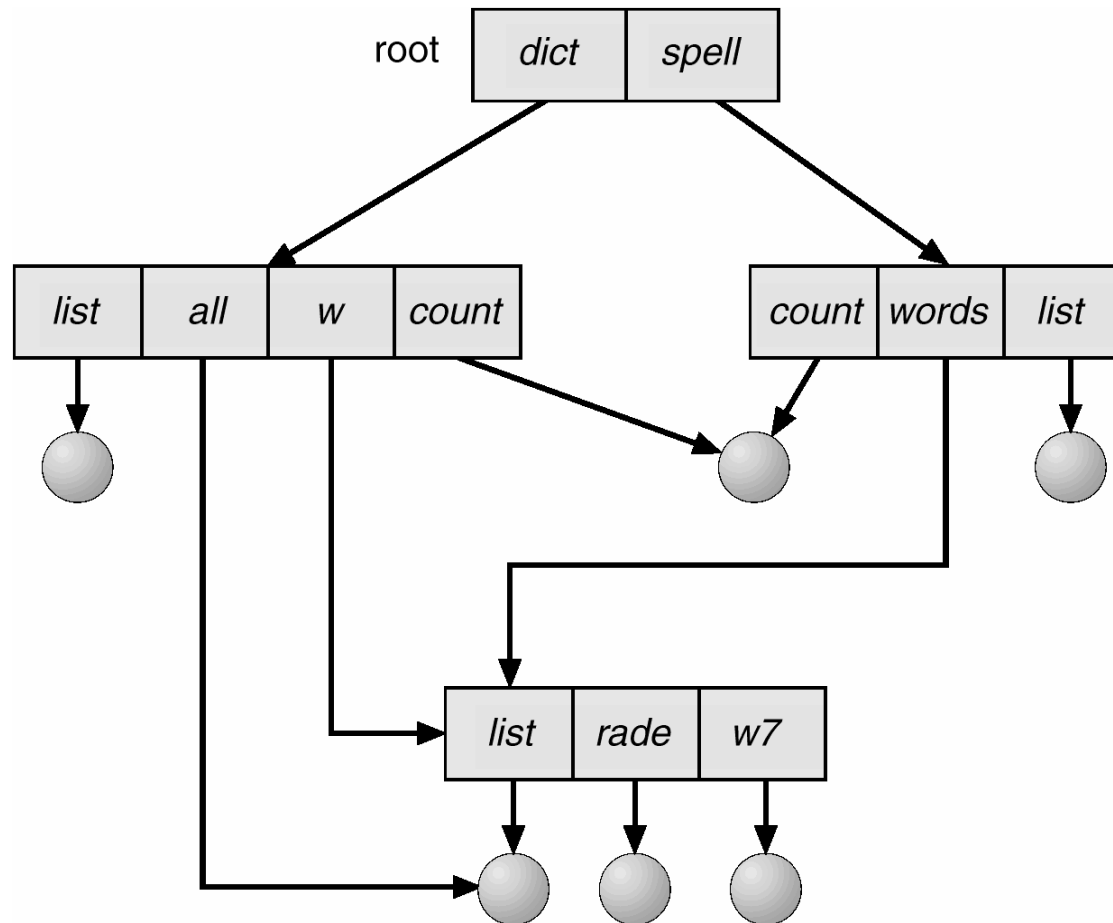
mkdir count



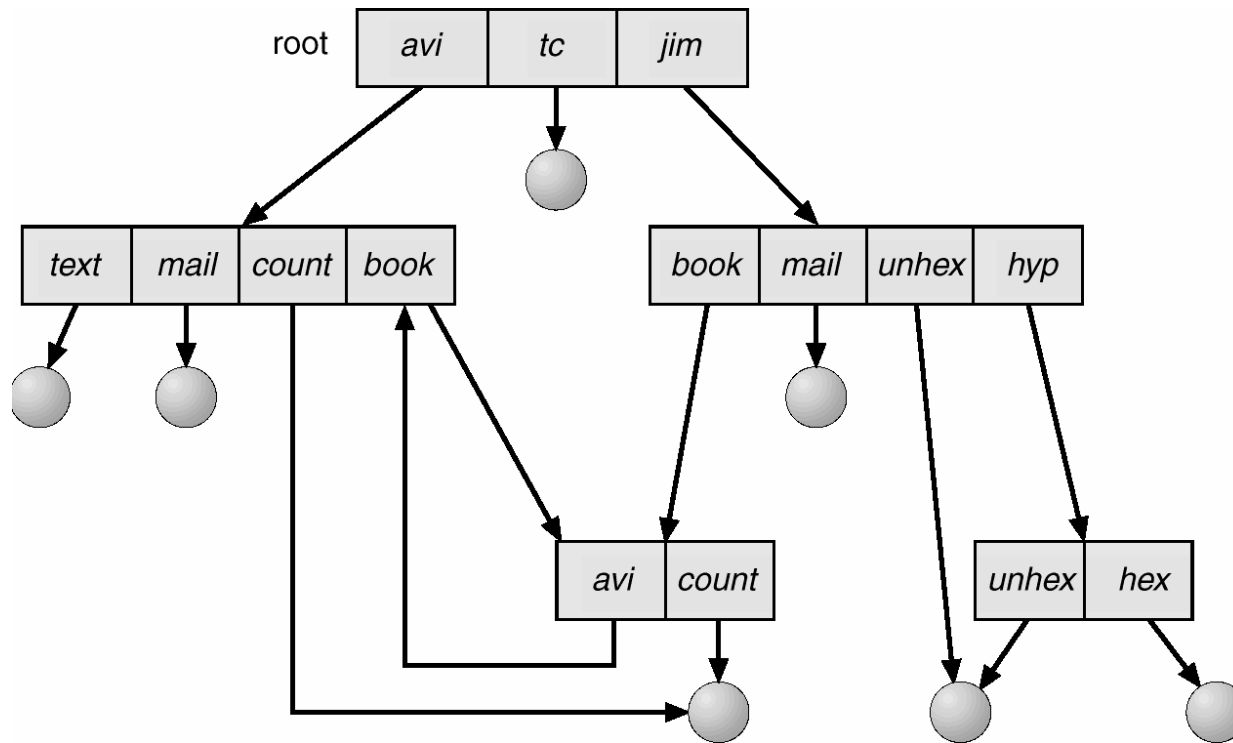
Suppression de mail \Rightarrow suppression du sous-arbre dont la racine est le sous-répertoire 'mail'

Structure de graphe acyclique

- Permet de partager des sous-répertoires et des fichiers



Structure de graphe général



Structure de graphe général

- Comment empêcher les cycles ?
 - Permettre seulement des liens vers des fichiers et non des sous-répertoires.
 - A chaque ajout d'un lien → algorithme de détection de cycle. Autoriser le lien seulement s'il n'y a pas de cycle.

Protection

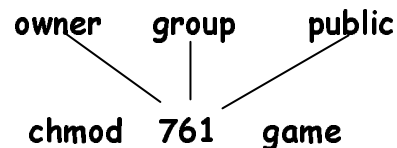
- Le propriétaire/créateur d'un fichier doit pouvoir contrôler:
 - Ce qui peut être fait
 - Par qui
- Types d'accès
 - Lecture
 - Ecriture
 - Exécution
 - Suppression
 - ...

Groupes et accès

- Type d'accès: lecture, écriture, exécution
- Trois types d'utilisateurs

a) propriétaire	7	⇒	RWX 1 1 1
b) groupe	6	⇒	RWX 1 1 0
c) public	1	⇒	RWX 0 0 1

- Des utilisateurs peuvent appartenir à un même groupe G.
- Pour chaque fichier et sous-répertoire, on définit des règles d'accès.



On peut changer le group auquel appartient un fichier:

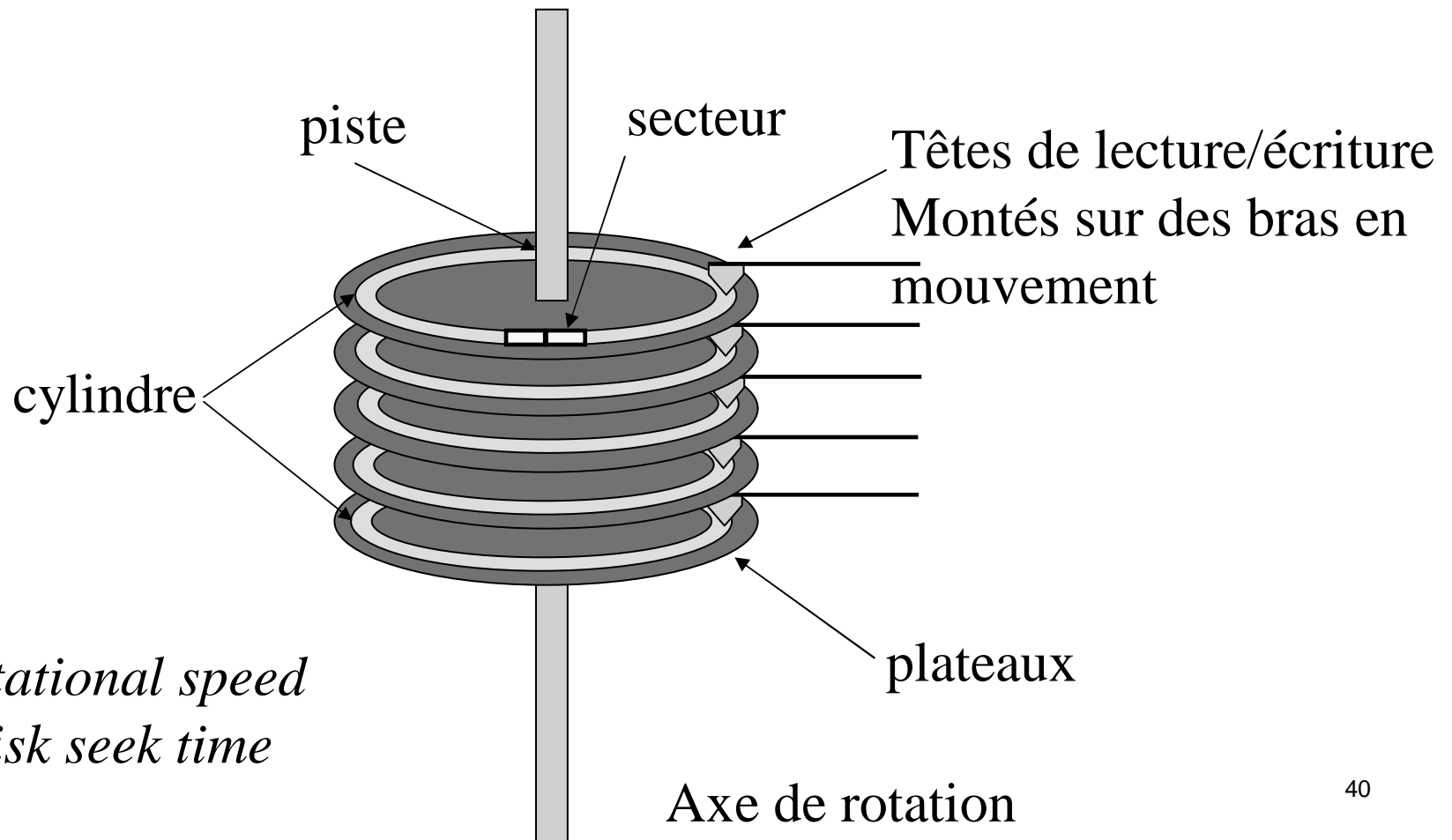
chgrp *G* *game*

Conception d'un SGF

- Organisation des disques
- Méthodes d'allocation
- Gestion de l'espace libre
- Performance

Disque magnétique

- Disque organisé en tête, cylindre, secteur

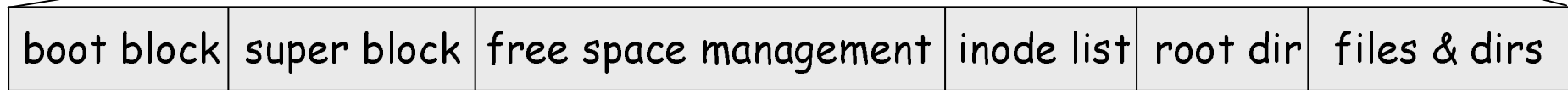


Disque: Organisation logique

- Le disque est divisé en une ou plusieurs partitions
 - Chaque partition a un SGF indépendant
 - Secteur 0 : contient le MBR (Master Boot Record)
 - MBR contient la table de partition
 - Une partition est marquée comme active
 - Block de démarrage (boot) – le 1er de la partition active
 - BIOS lit et exécute le MBR qui lit le bloc de boot et l'exécute.
 - Le programme dans le bloc de boot charge le SE et l'exécute.
 - Souvent un SGF a un superbloc qui contient des paramètres importants.

Exemple: Disque et SGF

Partition Table



Allocation Contiguë

- Chaque fichier est rangé dans une série de blocs adjacents sur le disque.



Fichier 1

Fichier 2

- Fait coïncider l'implantation physique avec la vision logique qu'a l'utilisateur d'un fichier

Allocation Contiguë

- Avantages:

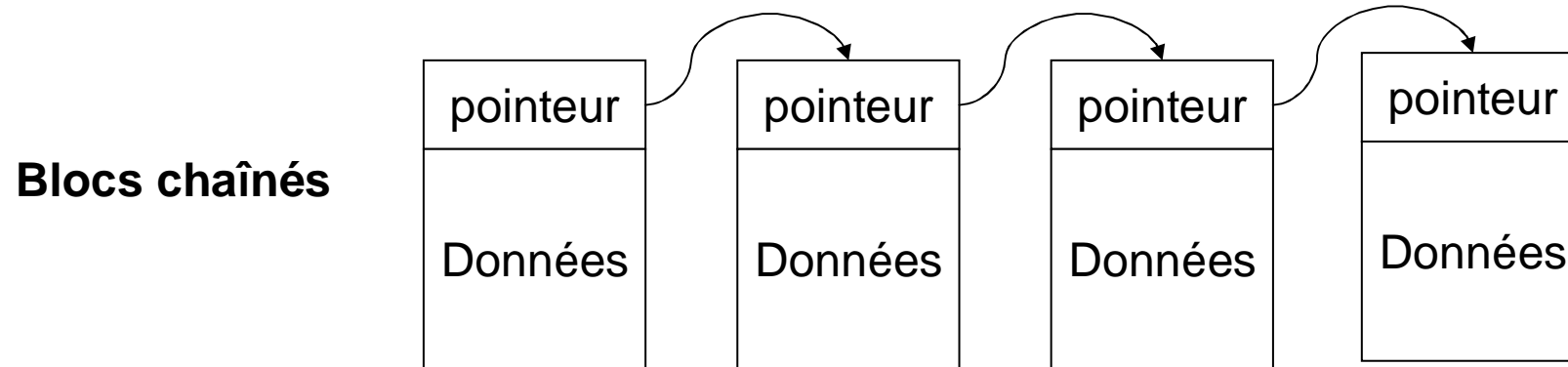
- Simple: on a besoin que de la position de départ (bloc n°x) et de la longueur.
- Accès direct à un bloc en temps constant
- Adapté aux supports “Write Once” (CDROM)

- Inconvénients:

- Gaspillage d'espace : ***fragmentation interne et externe.***
- On doit connaître à l'avance la taille du fichier
- L'extension peut nécessiter un déplacement du fichier

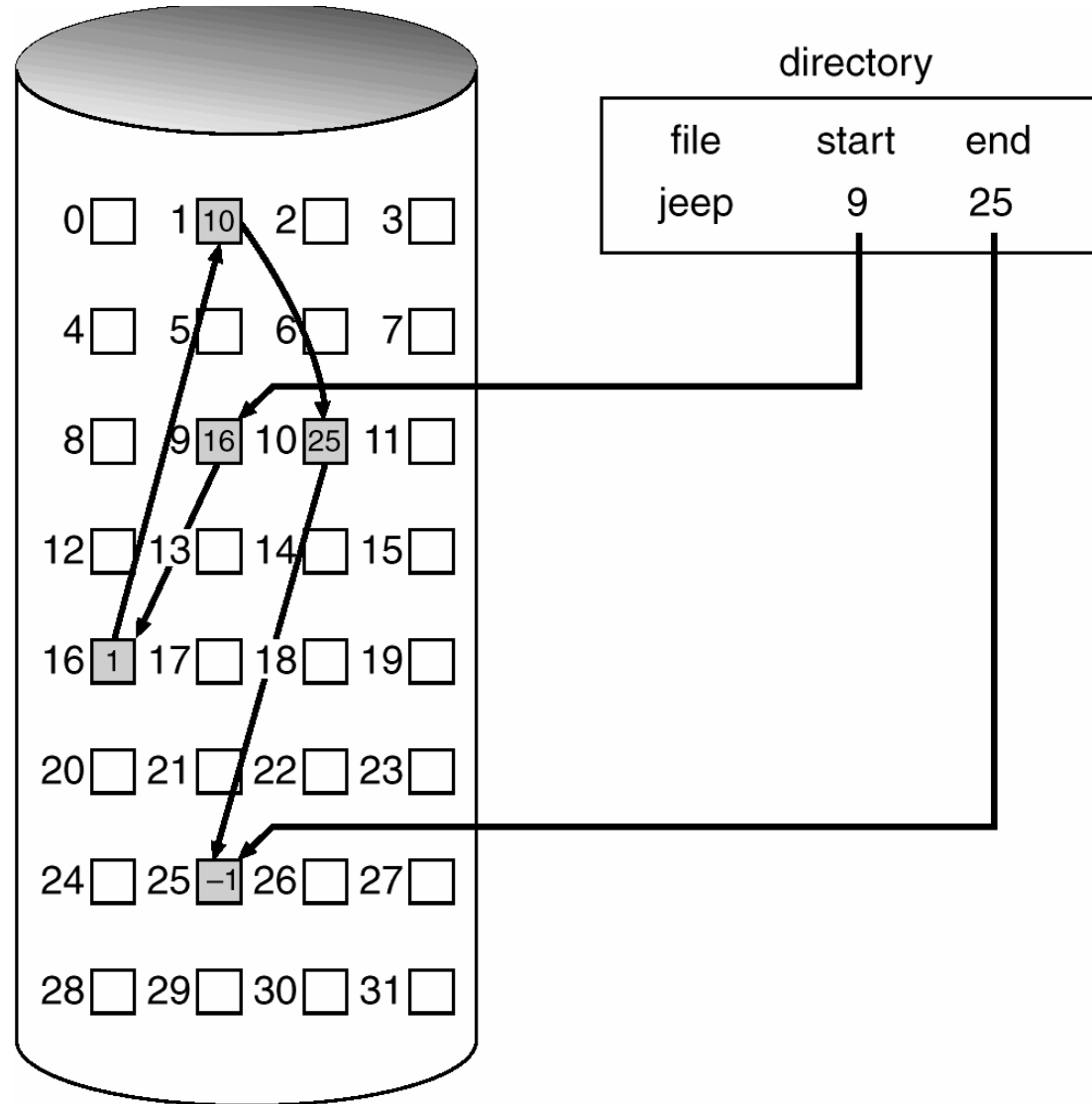
Allocation chaînée

- Chaque fichier est une liste chaînée de blocs: les blocs ne sont pas forcément contigus.



Les blocs sont alloués en fonction des besoins.

Allocation chaînée



Allocation chaînée

- Avantages:

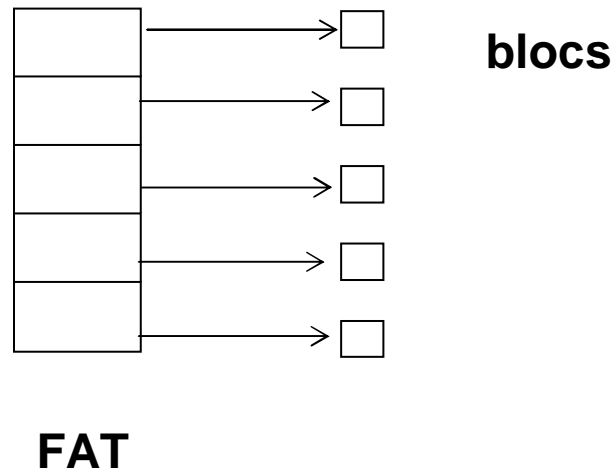
- On peut facilement étendre un fichier
- Simple – on a besoin que de l'adresse de départ
- Pas de gaspillage d'espace → pas de fragmentation externe

- Inconvénients:

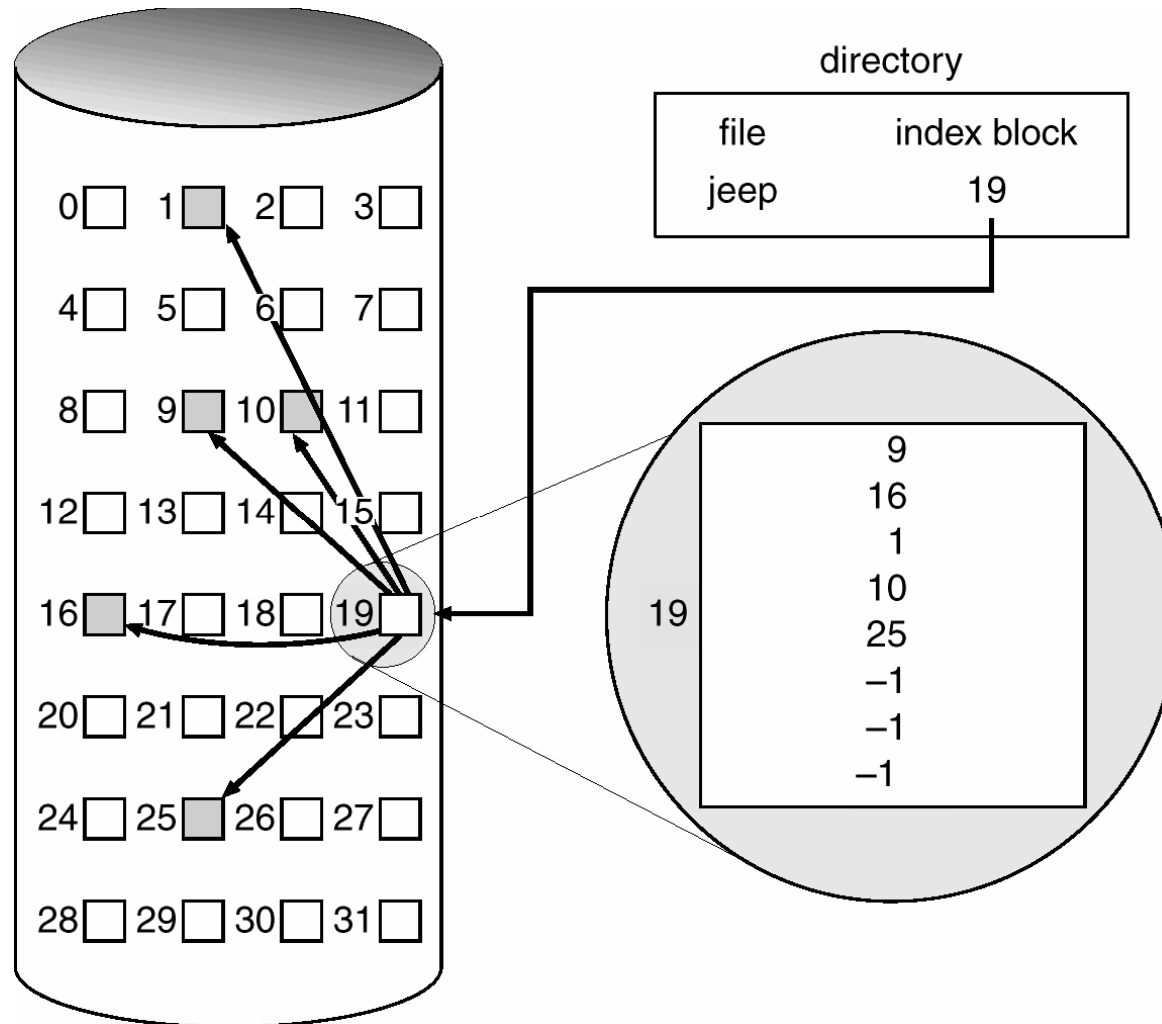
- L'accès à un bloc nécessite d'accéder à tous les blocs qui le précèdent → non adapté à l'accès direct
- Les pointeurs sont stockés sur disque

Allocation Indexée

- Tous les pointeurs sont regroupés dans une table d'allocation de fichiers (FAT: File Allocation Table)
- Elle est stockée dans un bloc spécial: le bloc d'index



Allocation Indexée



Allocation Indexée

- **Avantages:**

- Les fichiers sont facilement extensibles

- Accès direct facile

- Pas de fragmentation externe mais besoin d'un bloc d'index

- **Inconvénients:**

- Gaspillage d'espace pour le bloc d'index

- Problème pour les disques de grande taille

- **Solution: indexation à plusieurs niveaux**

Indexation à plusieurs niveaux

- Idée

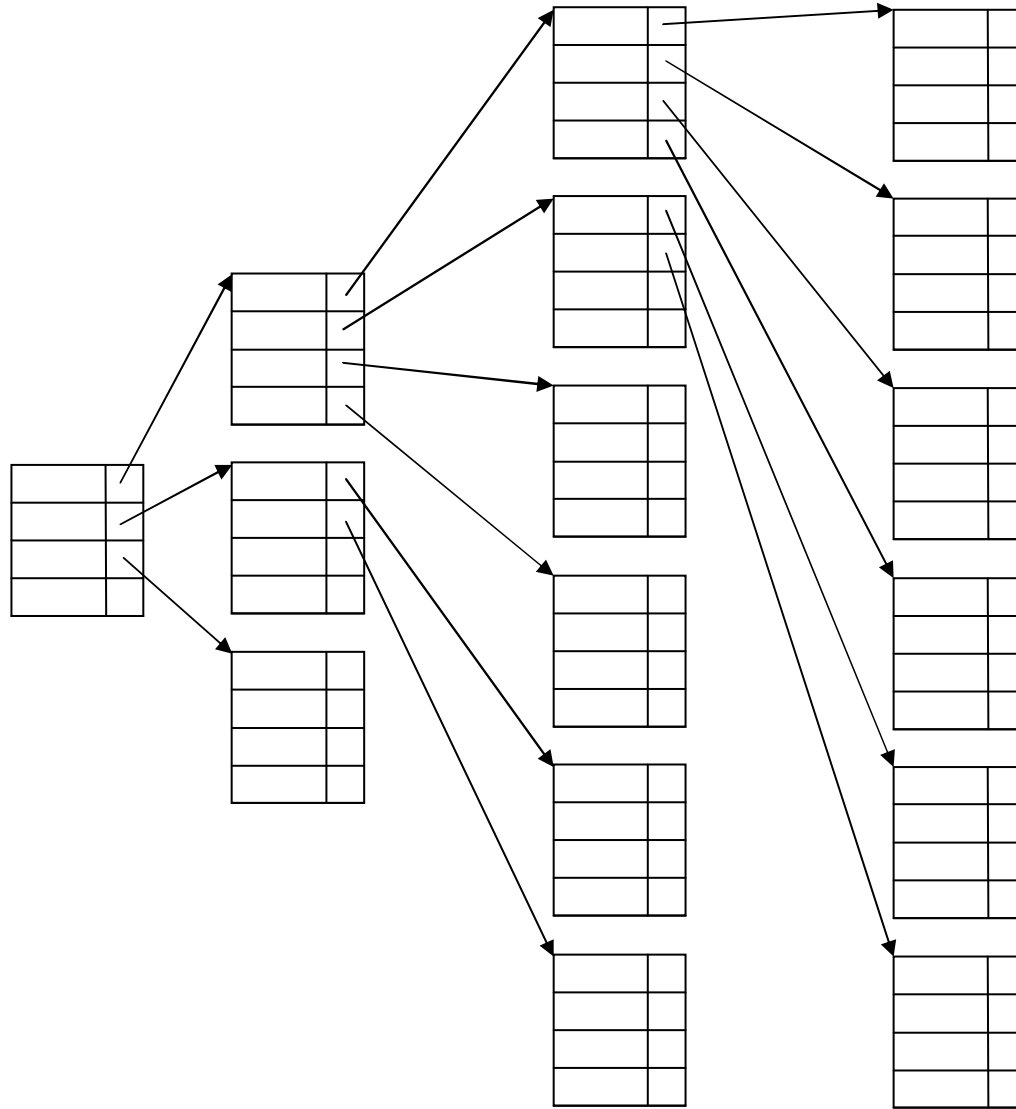
- Utiliser plusieurs niveaux de blocs

- Ex.: avec 3 niveaux:

- Les blocs de niveau 1 contiennent des pointeurs vers des blocs de niveau 2 qui eux-mêmes contiennent des pointeurs vers des blocs de niveau 3 qui pointent vers des blocs de données

- Si un bloc contient 128 pointeurs, avec 3 niveaux, on peut adresser $128 \times 128 \times 128 = 2097152$ blocs

Indexation à plusieurs niveaux



Implémentation des répertoires

- Liste linéaire de noms de fichiers avec des pointeurs vers les blocs de données.
 - Facile à mettre en oeuvre
 - Pas très performant
- Table de hachage.
 - Réduit le temps de recherche dans un répertoire
 - *Pb. des collisions* : situations où deux noms de fichiers ont la même clé de hachage
 - Taille fixée

Efficacité et performance

- L'efficacité dépend :
 - De l'allocation sur le disque et des algorithmes de recherche dans les répertoires
 - Du type de données contenues dans une entrée d'un répertoire
- Performance
 - Cache du disque – les blocs les plus souvent utilisés sont mis en mémoire
 - Techniques pour optimiser l'accès séquentiel
 - Utilisation d'une partie de la RAM en tant que disque virtuel → risque de perte de données.