Cours de systèmes d'exploitation centralisés

Chapitre 5

Systèmes de gestion de fichiers

Séance 2

Mme L. BOUZAR

2021-2022

4.2 Organisation des partitions en répertoires(étape2)

- **Définition d'un répertoire (catalogue ou dossier)**
  - C'est un ensemble de descripteurs de fichiers qui contiennent toutes les informations (attributs) définissant les fichiers d'un disque.
- **Utilisation des répertoires**

  Un répertoire est utilisé lors des opérations suivantes :
  - **Création d'un fichier** : Cela consiste à rajouter une entrée (descripteur de fichier) dans le répertoire,
  - **Destruction de fichier**: C'est une suppression logique(Ex: FAT mettre '**E5**' dans les 1er octet du nom du fichier ➔ entrée supprimée),
  - **Ouverture/fermeture d'un fichier**: Pour lire ou écrire des articles,
  - **Lister le répertoire**: Exemple commande « dir » de DOS ou « ls » de UNIX/Linux,
  - **Sauvegarde, restauration, copie de fichiers**, …
- Donc, un répertoire doit être organisé de manière à faciliter toutes ces opérations.

4.2.1 Descripteur de fichier (rappel)

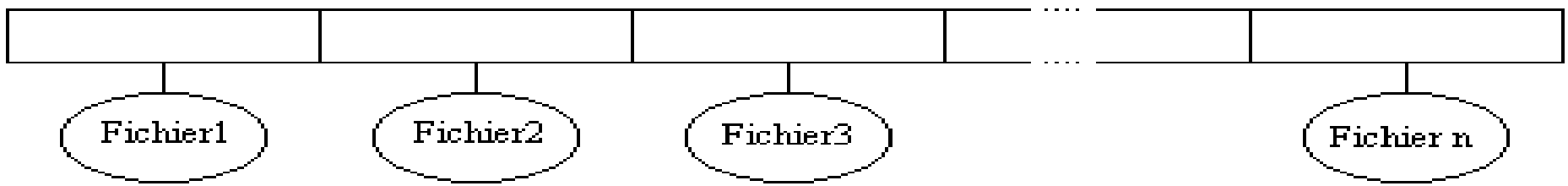
- Les informations contenues dans un descripteur varient selon les systèmes de fichiers. On peut trouver:
  - Nom du fichier, Extension du nom(ex: DOS),
  - Numéro de version du fichier (ex: système VMS),
  - Type(exécutable, binaire, texte,...),
  - Taille du fichier,
  - Taille d'un bloc,
  - Longueur d'article,
  - Organisation, Format des articles,
  - Protection,
  - Date de création, date de la dernière utilisation, date de la dernière modification,
  - Nombre d'utilisateurs simultanés (fichier partageable ou non),
  - Adresse d'implantation sur disque, …

4.2.2 Structure des répertoires

- Un répertoire peut être à un ou plusieurs niveaux.

4.2.2.1 Répertoire à un niveau

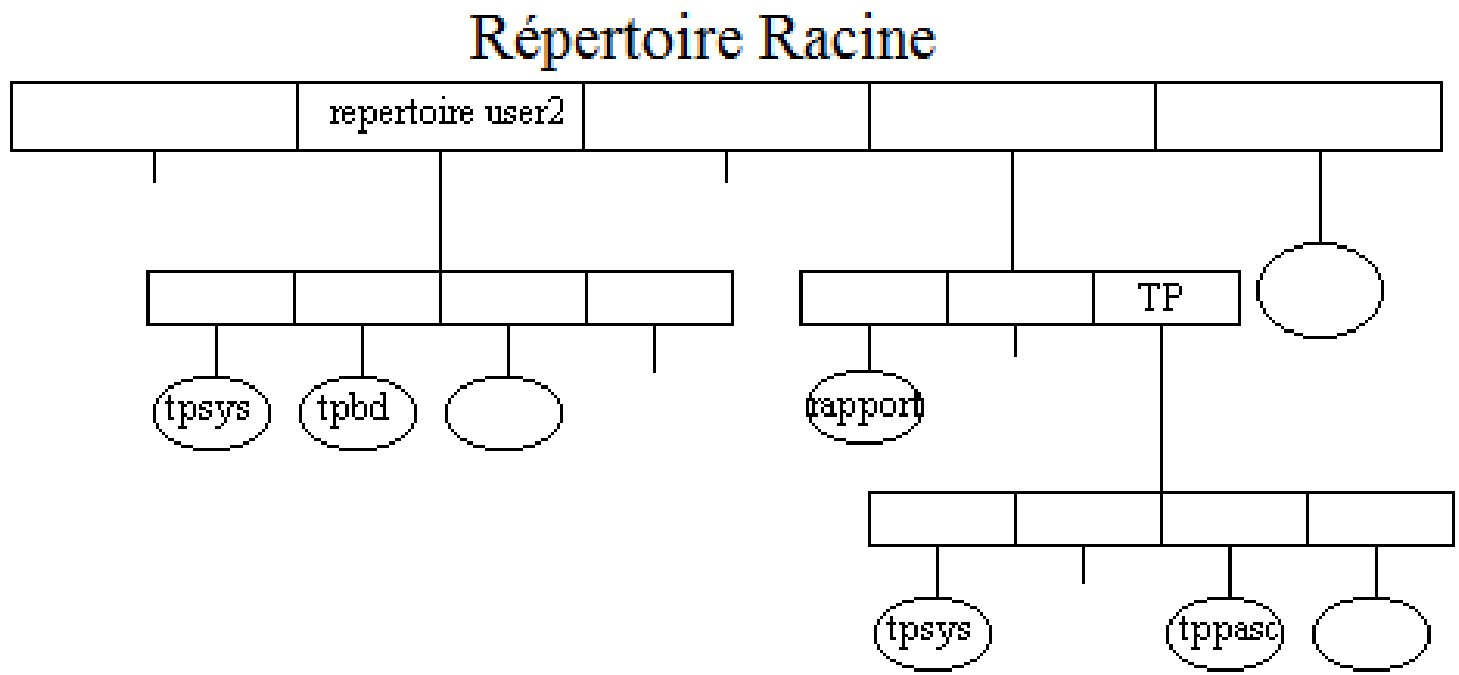
- Un disque(physique ou partition) dispose d'un seul répertoire, tous les descripteurs de fichiers se trouvent dans un même répertoire.
- Cette structure est facile à implanter;
  Cependant, elle ne permet pas à deux utilisateurs de donner le même nom à leur fichier.



Exemple: La première version du PC-DOS utilisait un répertoire à un seul niveau.

4.2.2.2 Répertoire hiérarchisé ou à plusieurs niveaux

- Un disque physique ou partition dispose d'un répertoire racine et éventuellement de un ou plusieurs « sous répertoires ».
- Une entrée (descripteur) d'un répertoire peut identifier un fichier ou un répertoire(sous répertoire).
- Le nombre de niveaux et le nombre d'entrées par répertoire dépendent du système de fichiers.



4.3 L'allocation de l'espace disque

- Les disques permettent de stocker un grand nombre de fichiers.
- Le problème principal consiste à savoir comment allouer de l'espace à ces fichiers, afin que l'espace disque soit utilisé de façon efficace et que l'on accède rapidement aux fichiers.
- Il existe deux stratégies pour stocker un fichier de n octets sur disque :
  1) On alloue n octets consécutifs sur le disque, c'est ce que l'on appelle l'allocation contiguë.
  2) On alloue plusieurs blocs de taille fixe pas nécessairement contigus, c'est ce que l'on appelle l'allocation non contiguë.

- La plupart des systèmes de fichiers actuels utilisent l'allocation non contiguë.

4.3.1 L'allocation contiguë

- Dans cette méthode l'espace nécessaire au fichier est demandé à la création, si le système ne peut pas satisfaire cette demande le fichier n'est pas créé.
- L'espace libre est représenté à l'aide d'une liste des zones libres.
- Au début le disque est constitué d'une seule zone.
- Des zones sont créées au fur et à mesure de la création(allocation de l'espace) et destruction(libération de l'espace) de fichiers.
- Il existe plusieurs stratégies de sélection d'une zone ; les plus importantes sont: first-fit et best-fit, ...

4.3.2 L'allocation non contiguë

- Pour remédier au problème de l'émiettement de l'espace disque, on considère cet espace comme un ensemble de blocs de même taille et l'unité d'allocation est le bloc.

4.3.3.1 Taille des blocs

- Adressage CHS, Un disque est composé de cylindres, pistes et secteurs.
- Avec l'adressage LBA, un disque est considéré comme un ensemble de secteurs numérotés de 0 à n-1.
- La plus petite quantité d'informations que l'on peut transférer, en mode bloc, entre le disque et la mémoire centrale est le secteur.
- **Taille d'un bloc doit être un multiple de secteur**.
- Si taille de bloc bien choisie ➔
  - ➢ Meilleure occupation de l'espace disque et
  - ➢ Diminution du temps d'E/S en traitement séquentiel.

4.3.3.2 Représentation des blocs libres

- Comment représenter l'espace libre ou les blocs libres ?
- Le système de fichiers doit maintenir tous les blocs libres dans une structure de données adéquate.
- Cette structure de données est utilisée lors de la création et la destruction des fichiers.
- On peut l'implanter à l'aide d'une table de bits(vecteurs de bits) ou d'une liste chaînée.

a) Tables de bits

- L'espace disque est représenté à l'aide d'une table de bits, un bit par bloc.
  - ✓ bit = 0 ➔ bloc libre,
  - ✓ bit = 1 ➔ bloc occupé.

- **Exemple:**

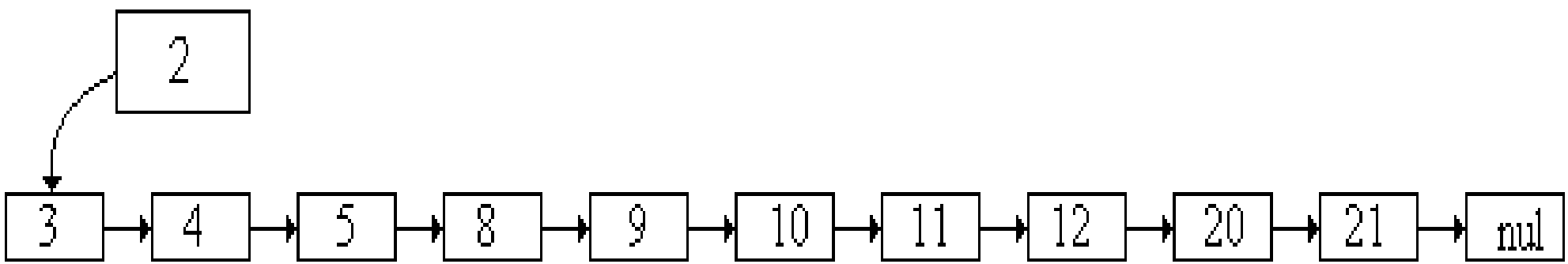
  Les blocs suivants sont libres: 2, 3, 4, 5, 8, 9, 10, 11, 12, 20, 21.

1	1	0	0	0	0	1	1	0	0	0	0	0	1	1	1	1	1	1	1	0	0	1	1	...

b) Liste des blocs libres (listes de blocs chaînés)

- Les blocs libres sont chaînés entres eux et constitue une liste des blocs libres.

  Les blocs suivants sont libres: 2, 3, 4, 5, 8, 9, 10, 11, 12, 20, 21.

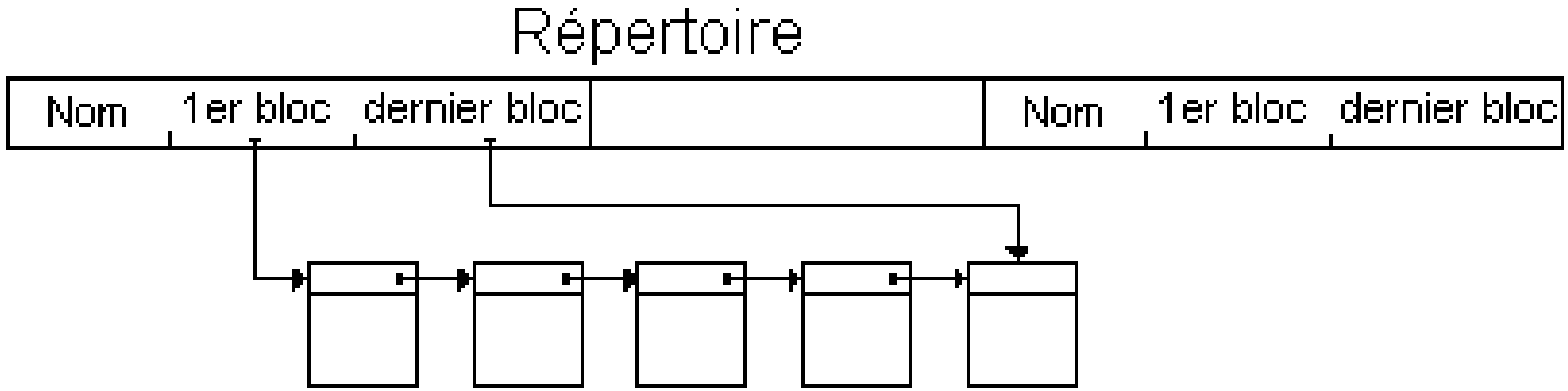


4.3.3.3 Représentation de l'espace disque (blocs) d'un fichier

- Comment représenter l'espace disque ou blocs alloués à un fichier ?
- Principales techniques utilisées:
  - ✓ Blocs chainés,
  - ✓ Bloc d'index d'allocation,
  - ✓ Fichiers d'allocation(Mapping).

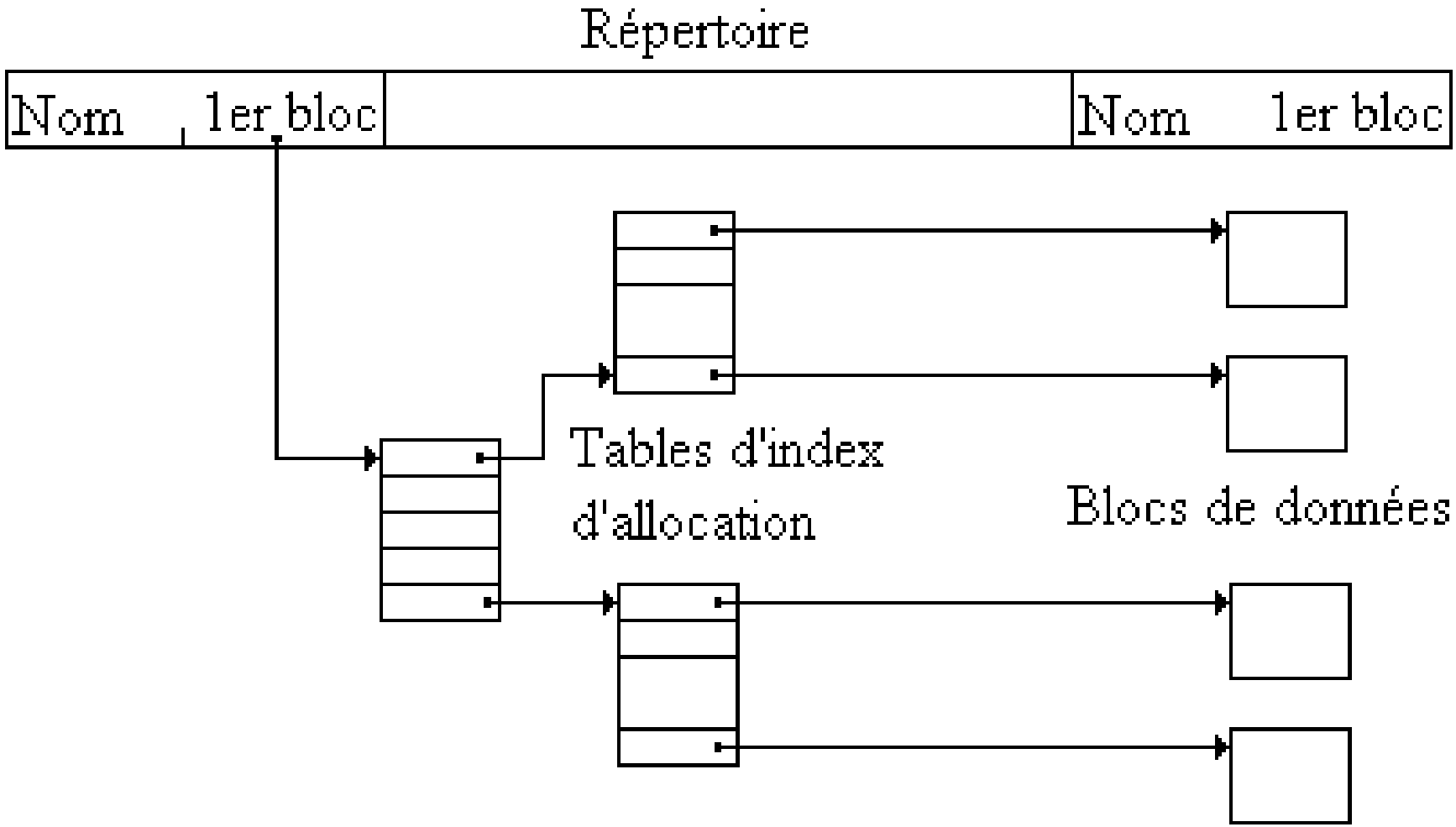
a) Blocs chaînés

- Le fichier est constitué d'un ensemble de blocs chaînés entre eux. ➔
- Un bloc est composé de deux parties : Un pointeur et des données.
- Le descripteur du fichier contient:
  - ✓ l'adresse du premier bloc du fichier (tête de liste),
  - ✓ l'adresse du dernier bloc.
- Ce mode d'allocation est bien adapté à l'accès séquentiel. Cependant, il est très coûteux pour l'accès direct.



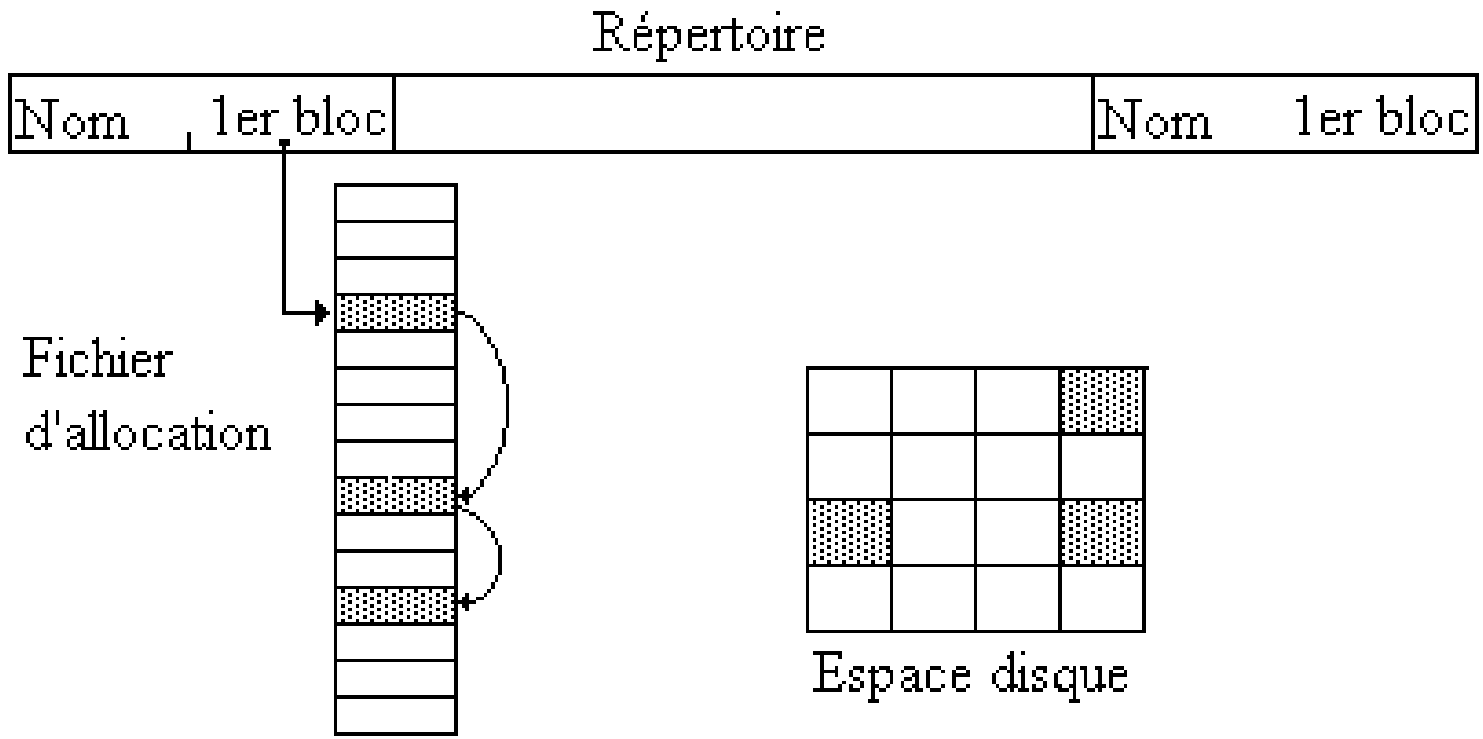
b) Tables d'index d'allocation

- Un fichier est constitué :
  - de tables d'index d'allocation (blocs),
  - un ensemble de blocs de données.
- Les blocs d'index d'allocation sont chaînés entre eux et pointent les blocs de données.
- Le descripteur du fichier contient l'adresse de la première table d'allocation.


Répertoire
Nom
1er bloc
Nom
1er bloc
Tables d'index
d'allocation
Blocs de données

c) Le fichier d'allocation (Mapping)

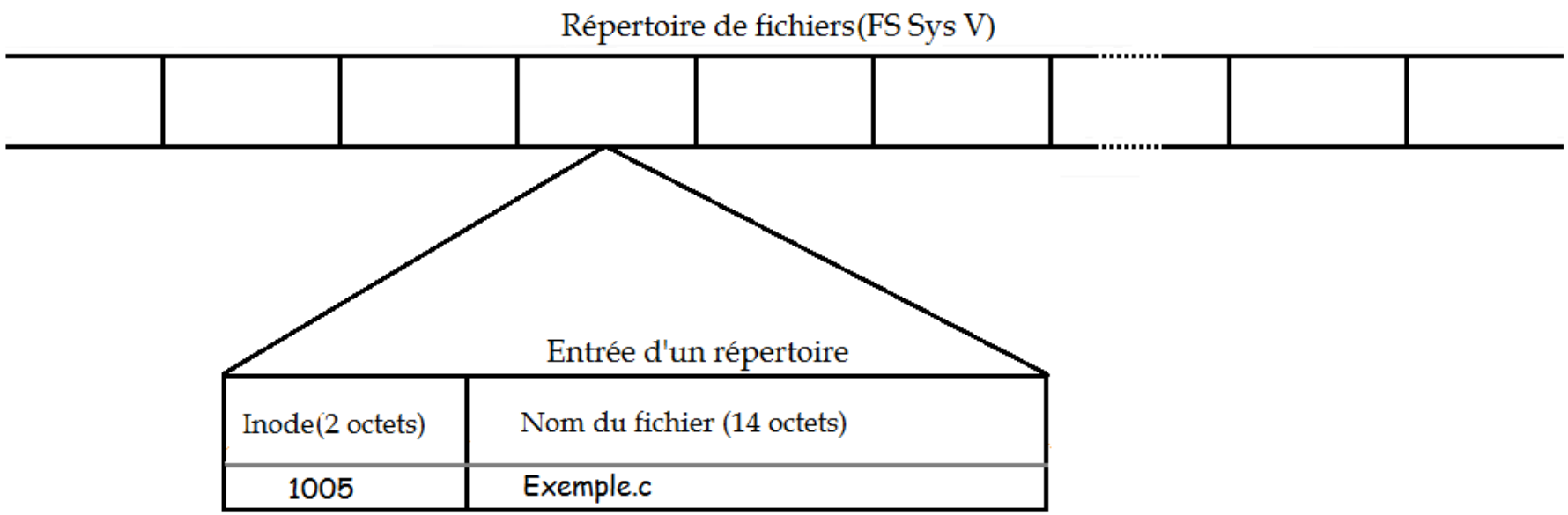
- L'espace disque est représenté à l'aide d'une table d'allocation(fichier).
- Une entrée de cette table (fichier) correspond à un bloc.
- La table contient autant d'entrées qu'il y a de blocs de données dans le disque.
- Les blocs alloués à un fichier sont chaînés entre eux.
- Le descripteur du fichier contient l'adresse(index) du premier bloc dans la table d'allocation(fichier d'allocation).



4.4 Système de fichiers Unix

- Le système de fichiers **Unix** est hiérarchisé : il est organisé en une structure arborescente dont les nœuds sont des répertoires et les feuilles sont des fichiers.
- L'arbre est construit à partir d'un répertoire racine unique que l'on appelle **root** désigné par '**/**' .
- Principaux types de fichiers :
  - ✓ fichiers ordinaires(-),
  - ✓ répertoires(d),
  - ✓ Liens symboliques(l),
  - ✓ Fichiers spéciaux :
    - ❖ Périphériques mode caractères(c),
    - ❖ Périphériques mode bloc(b),
    - ❖ …

- Un répertoire est un fichier dont le contenu est une suite de «couples » formés d'un nom de fichier et d'un index (numéro de l'inode).
- L'index permet d'accéder à la table des descripteurs de fichiers que l'on appelle table des inodes.
- Dans les premières versions de Unix les noms de fichiers étaient limités à 14 caractères.
- Une entrée a donc 16 octets :
  - ✓ 2 octets pour le numéro de l'inode ,
  - ✓ 14 octets pour le nom du fichier.

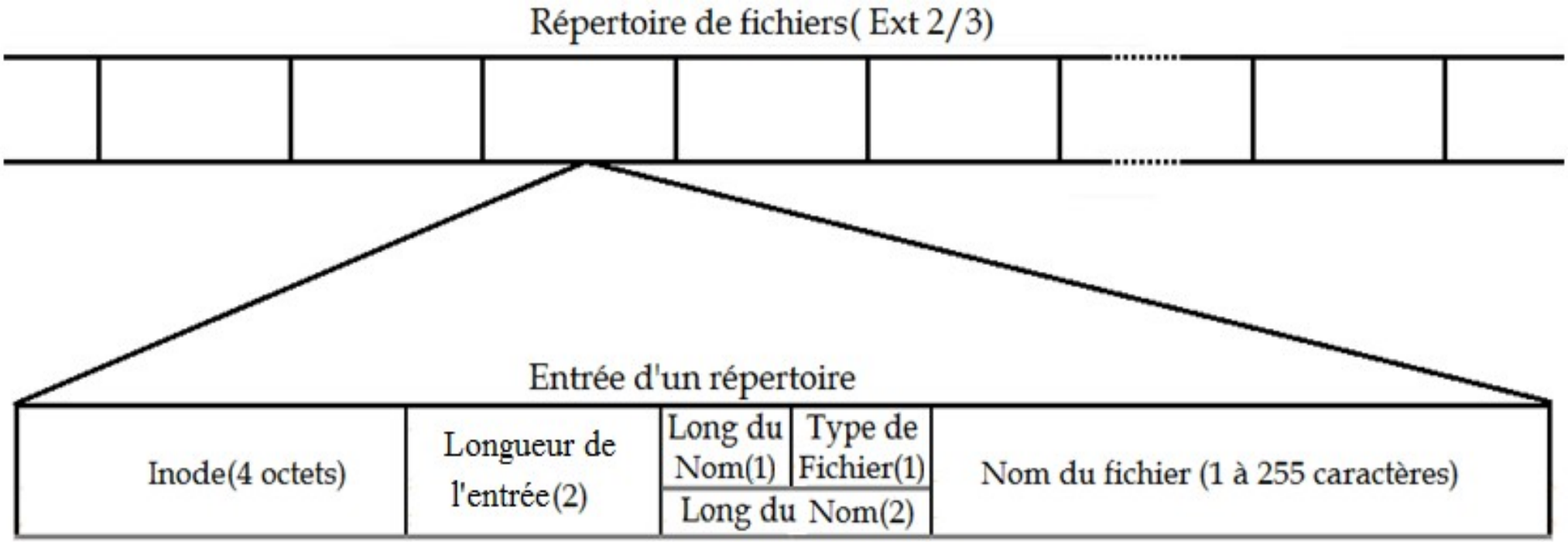


- Versions BSD(à partir de BSD4.2) et linux(ext2,ext3,...),
  - Les noms de fichiers sont de longueur variable, jusqu'à 255 octets ➔ les entrées de répertoires sont donc de longueur variable.

    Un Répertoire est un fichier en format variable.

4.4.1 Structure d'une entrée d'un répertoire du système de fichiers Ext2/ext3 de linux

- Chaque entrée contient :
  - ✓ Le numéro de l'inode(index du nœud) ,
  - ✓ La longueur totale de l'entrée (ou déplacement jusqu'à l'entrée suivante),
  - ✓ La longueur du nom du fichier,
  - ✓ Le type de fichier (n'existe pas dans toutes les versions),
  - ✓ Le nom du fichier.

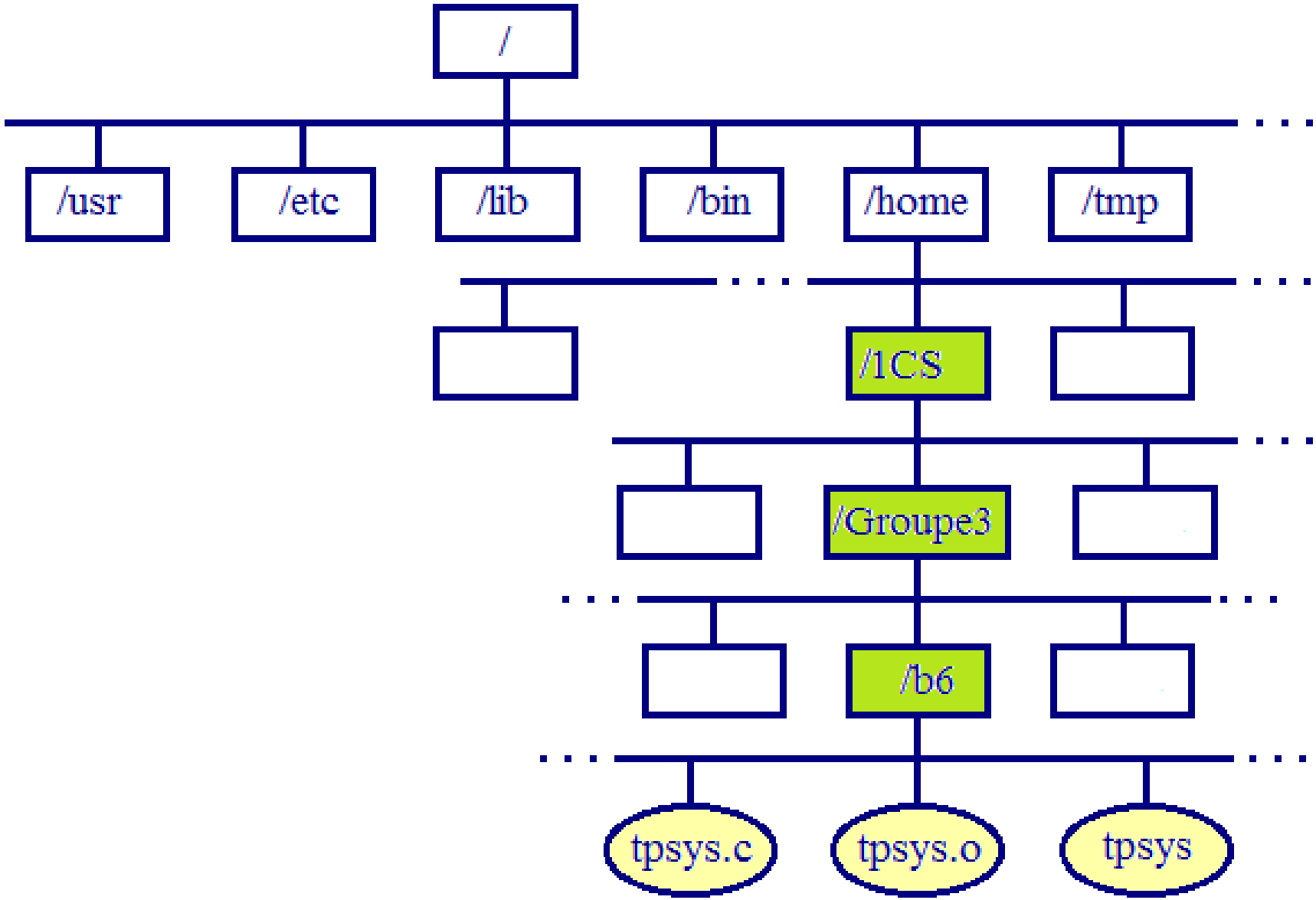


- A la création d'un répertoire, 2 entrées «**.**» et «**..**» sont créées
  - ✓ «.» : fait référence au répertoire courant,
  - ✓ «..» : fait référence au répertoire père.

- L'utilisateur référence un fichier avec un nom,
- Le système emploie le numéro de l'inode pour l'accès au fichier.

- Dans le système Unix plusieurs répertoires sont créés lors de l'installation du système: usr, sbin, etc, dev, home, ...

- **Exemple de système de fichiers**



4.4.2 Désignation des fichiers

- Le nom d'un fichier Unix est composé d'une suite de caractères;
- Unix fait la différence entre les majuscules et les minuscules (la casse).
- Le caractère « - » est déconseillé au début d'un nom de fichier (certaines commandes pourraient l'interpréter comme une option).
- Un fichier peut être désigné à l'aide d'un chemin d'accès absolu ou un chemin d'accès relatif.

- **Chemin absolu**
  - Un chemin d'accès absolu démarre de la racine « / » du système de fichiers et contient tous les noms de répertoires constituant le chemin, ces noms de répertoires sont séparés par le caractère /.
  - Le dernier nom du chemin d'accès correspond au nom du répertoire ou du fichier auquel on désire accéder.
  - **Exemple1 :** Chemin absolu du fichier *tpsys* : */home/1cs/Groupe3/b6/tpsys*

- **Chemin relatif**
  - Pour accéder à un répertoire ou à un fichier du répertoire courant, on utilise le nom du fichier ou répertoire désiré.
  - Un chemin d'accès relatif commence à partir du répertoire courant.

  - **Exemple**:
    - ✓ Supposons que le répertoire courant soit */home/1cs/Groupe3/b6*
    - ✓ Le chemin relatif *tpsys* est équivalent au chemin absolu */home/1cs/Groupe3/b6/tpsys*

4.4.3 Structure d’un disque logique (volume)

- Un périphérique peut être un disque interne, un disque externe, un flash disque, …
- Un disque peut être découpé en disques logiques (partitions).
- Chacun d'eux contient son propre système de fichiers logique.
-

4.4.3.1 Structure d'un disque logique (Unix System V)

- **Remarque:** Cette structure ne correspond pas aux systèmes de fichiers BSD et linux(ext2, ...)

- Le super bloc contient les informations suivantes :
  - ✓ Taille du système de fichiers,
  - ✓ Nom du système de fichiers,
  - ✓ Nombre de blocs,
  - ✓ Liste des blocs libres,
  - ✓ Pointeur vers le premier bloc libre,
  - ✓ Nombre d'inodes,
  - ✓ Liste des inodes libres,
  - ✓ Pointeur vers le premier inode libre,
  - ✓ Caractéristiques du disque.

- L’inode est une structure contenant toutes les informations définissant un fichier, il contient:
  - Type de fichier,
  - Nombre de liens,
  - Identificateur du propriétaire,
  - Identificateur du groupe,
  - Taille du fichier en octets,
  - Date de création ou Date de la dernière modification,
  - Date de la dernière utilisation(dernier accès),
  - Date de la dernière modification de l’inode,
  - Protection ou droit d'accès (12 bits),
  - **Table d’index d’allocation du fichier (13 mots de 4 octets),**
  - *Autres informations selon les versions.*

- **Quelques différences entre Unix SystemV et Unix BSD :**

1. Les disques sous BSD sont organisés par groupes de cylindres

   Chacun de ces groupes a la même organisation que les disques logiques System V ➔ permet de réduire le déplacement des têtes de lecture.

2. Les blocs de données sont plus grands (4K ou 8K).
3. La Table d'index d'allocation du fichier contient :
   - ✓ 12 adresses directes,
   - ✓ une adresse indirecte,
   - ✓ une adresse indirectes doubles
   - ✓ une adresse indirection triple (cette dernière n'est pas utilisée car avec des blocs de taille 4k octets la taille du fichier dépasse les 2^{32} octets permis).
4. Noms des fichiers de **1 à 255** octets ➔ Les répertoires sont composés d'enregistrements de tailles variables.

4.4.3.2 Structure d'un disque logique du système de fichier ext2/3 de linux

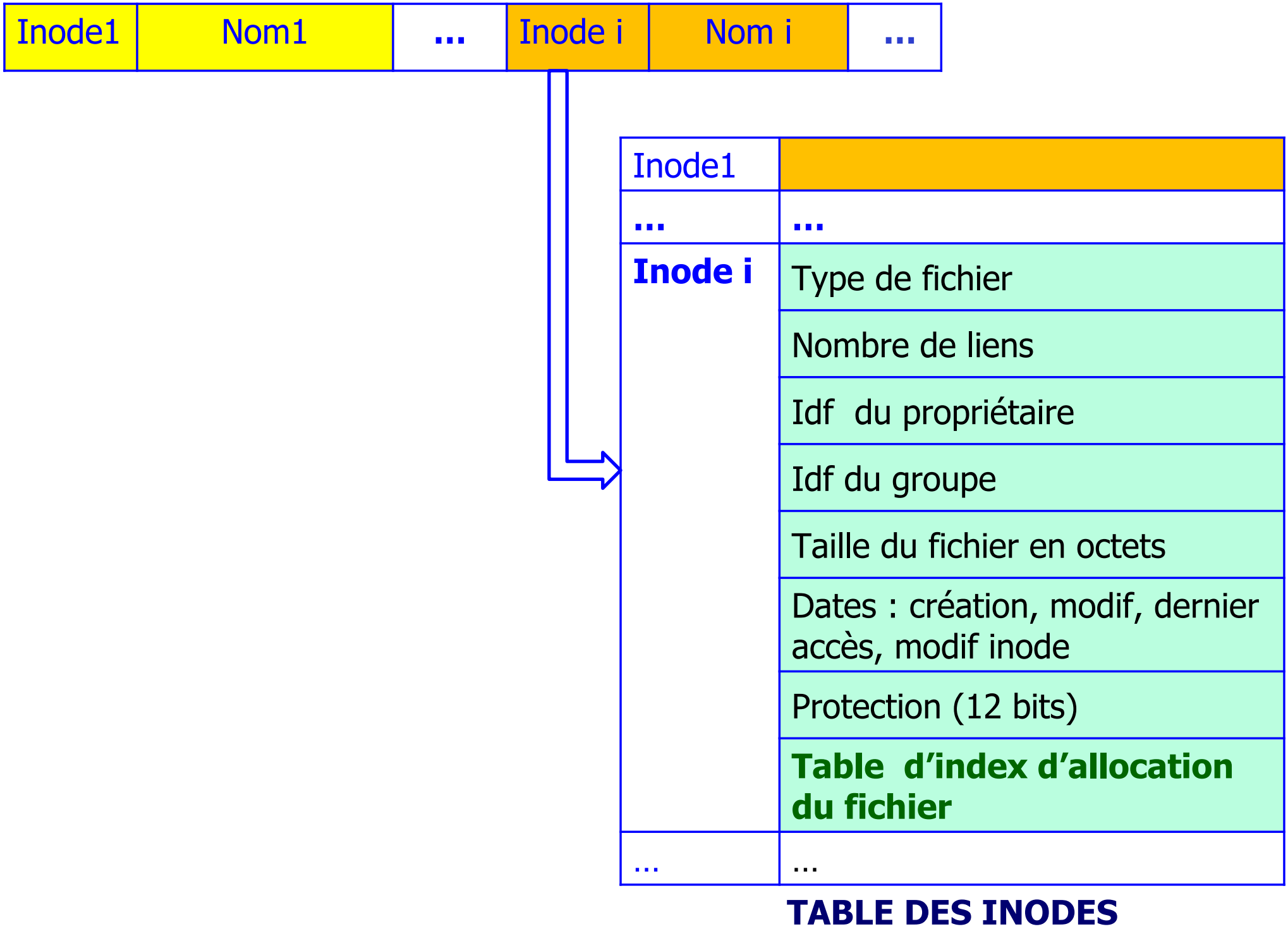
- Un disque logique ou volume (partition ou disque dur) du système de fichiers Ext2/Ext3 est divisé en groupes de blocs(similaires aux groupes de cylindres des systèmes de fichiers BSD).
- Chaque groupe de blocs contient :
  - ✓ Super bloc,
  - ✓ Table des descripteurs qui localise les différentes tables du groupe,
  - ✓ Table de bitmaps des blocs du groupe,
  - ✓ Table de bitmaps des inodes du groupe,
  - ✓ Table des inodes du groupe,
  - ✓ Blocs de données contenant les répertoires et fichiers.

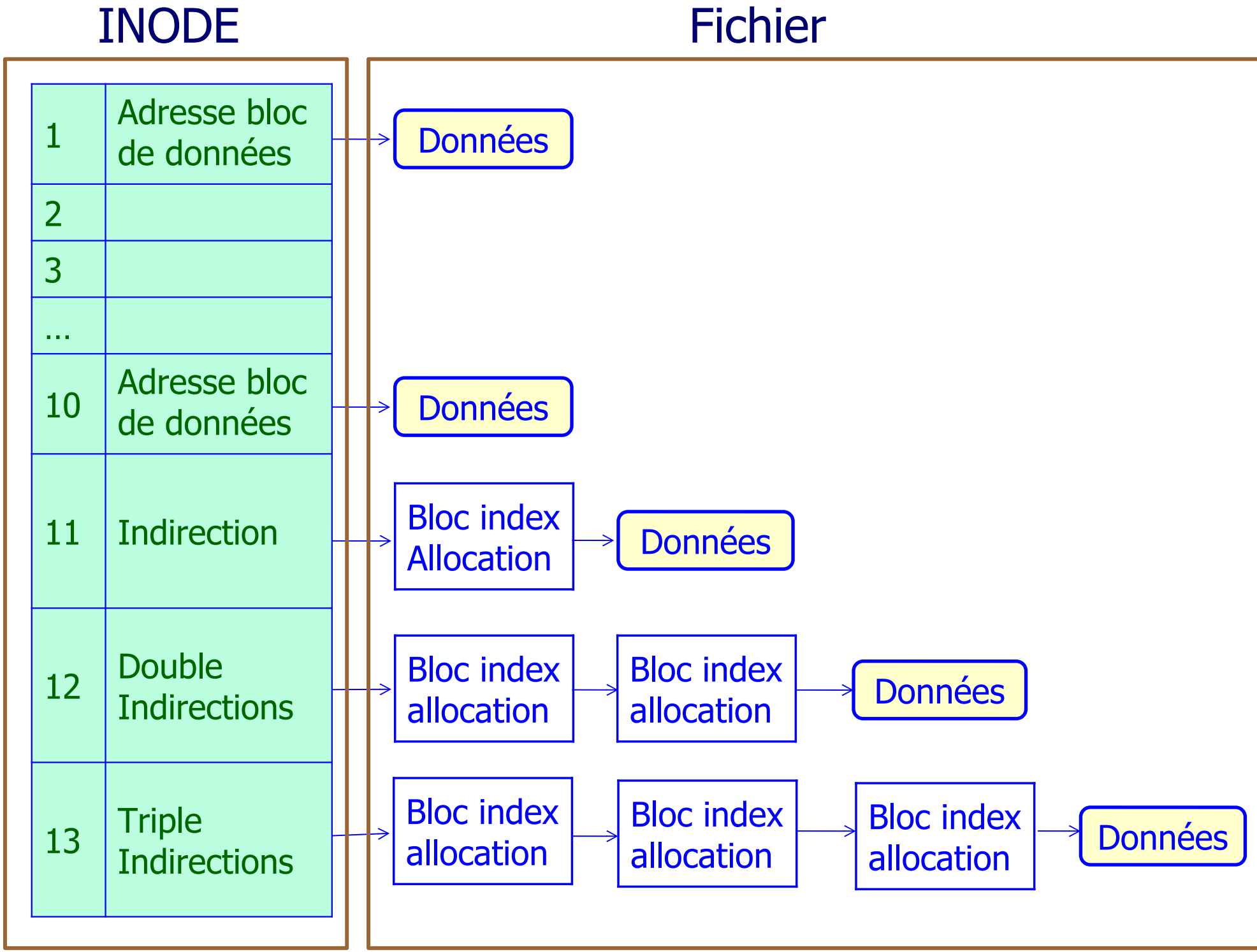
- Dans chacun des groupes, on trouve une copie du super bloc et une copie de la table des descripteurs. ➔ Meilleure fiabilité du système de fichiers linux (ext2/3).

Secteur Boot						
Groupe 1 de blocs	**Groupe 2 de blocs**	**Groupe 3 de blocs**				**Groupe n de blocs**
Super bloc						
Table des descripteurs						
Tables de bitmaps des blocs						Super bloc
Tables de bitmaps des Inodes		Super bloc				Table des descripteurs
Table des inodes	Super bloc	Table des descripteurs				
Blocs de données	Table des descripteurs					

4.4.4 Allocation de l'espace disque dans le système Unix system V

- Espace disque = ensemble de blocs.
- Unité d'allocation = bloc. Allocation dynamique par bloc.
- Taille des blocs varie selon les versions des systèmes: 512, 1024, 2048, …
- La taille la plus utilisée est 1024 octets.
- L'espace libre est décrit à l'aide d'une liste de blocs libres qui se trouve dans le super bloc.
- L'espace disque alloué (blocs alloués) à un fichier est représenté à l'aide de tables d'index d'allocation:
  - ✓ La première table d'index d'allocation composée de 13 entrées se trouve dans l'inode du fichier,
  - ✓ Selon la taille du fichier on peut utiliser d'autres index (blocs d'index): ces index font partie du fichier,
  - ✓ Taille d'une entrée de la table d'index d'allocation ou d'un bloc d'index d'allocation = 4 octets.


Inode1
Nom1
...
Inode i
Nom i
...
Inode1
...
...
Inode i
Type de fichier
Nombre de liens
Idf du propriétaire
Idf du groupe
Taille du fichier en octets
Dates : création, modif, dernier accès, modif inode
Protection (12 bits)
Table d'index d'allocation du fichier
...
...
TABLE DES INODES


INODE
Fichier
1
Adresse bloc de données
2
3
...
10
Adresse bloc de données
11
Indirection
12
Double Indirections
13
Triple Indirections
Données
Données
Bloc index Allocation
Données
Bloc index allocation
Bloc index allocation
Données
Bloc index allocation
Bloc index allocation
Bloc index allocation
Données

- **La taille réelle du fichier = blocs d'index d'allocation + blocs de données.**
- **Exemple:** Taille d'un bloc = **1024**; Longueur d'une entrée = **4 octets**.
- **Le nombre maximal de blocs de données d'un fichier =**
  $10 + 1024/4 + (1024/4)^2 + (1024/4)^3 =$ **$10 + 256 + 256^2 + 256^3$**
  = 16843018 blocs
- **Taille réelle maximale =**
  $10+(1+256)+(1+256+256^2)+(1+256+256^2+256^3)$ = **16 909 069 blocs.**

- **Remarques :**
  - Le temps d'accès aux blocs du fichier n'est pas uniforme :
    les 10 premiers blocs sont privilégiés.
  - Cette représentation permet un accès rapide aux petits fichiers.