Chap. IV : Le système de fichiers (1ère partie)

Laurent Poinsot

UMR 7030 - Université Paris 13 - Institut Galilée

Cours "Architecture et Système"

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 utilise.
- 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 fichiers (ou file system).

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 utilise.
- 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 fichiers (ou file system).

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 utilise.
- 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 fichiers (ou file system).

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 utilise.
- 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 fichiers (ou file system).

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 utilise.
- 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 fichiers (ou file system).

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 utilise.
- 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 fichiers (ou file system).

Nom des fichiers

Tous les systèmes permettent des noms de fichiers comprenant entre un et huit caractères, certains autorisent jusqu'à deux cent cinquante-cinq caractères et permettent les caractères spéciaux. Les O.S. de la famille UNIX distinguent les caractères minuscules et majuscules (on dit qu'ils respectent la casse), Windows non.
Beaucoup de systèmes d'exploitation gèrent des noms de fichiers comprenant deux parties :

- La première partie située avant le point ;
- La seconde partie située après le point appelée extension et qui permet d'indiquer le type du fichier.

Sous UNIX, on peut avoir deux extensions ou plus, par exemple " .c.z " correspond 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 un programme correspondant lui est associé (par exemple, Acrobat Reader peut être associé à une extension " .pdf ").

Nom des fichiers

Tous les systèmes permettent des noms de fichiers comprenant entre un et huit caractères, certains autorisent jusqu'à deux cent cinquante-cinq caractères et permettent les caractères spéciaux. Les O.S. de la famille UNIX distinguent les caractères minuscules et majuscules (on dit qu'ils respectent la casse), Windows non.
Beaucoup de systèmes d'exploitation gèrent des noms de fichiers comprenant deux parties :

- La première partie située avant le point ;
- La seconde partie située après le point appelée extension et qui permet d'indiquer le type du fichier.

Sous UNIX, on peut avoir deux extensions ou plus, par exemple " .c.z " correspond 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 un programme correspondant lui est associé (par exemple, Acrobat Reader peut être associé à une extension " .pdf ").

Nom des fichiers

Tous les systèmes permettent des noms de fichiers comprenant entre un et huit caractères, certains autorisent jusqu'à deux cent cinquante-cinq caractères et permettent les caractères spéciaux. Les O.S. de la famille UNIX distinguent les caractères minuscules et majuscules (on dit qu'ils respectent la casse), Windows non.

Beaucoup de systèmes d'exploitation gèrent des noms de fichiers comprenant deux parties :

- La première partie située avant le point ;
- La seconde partie située après le point appelée extension et qui permet d'indiquer le type du fichier.

Sous UNIX, on peut avoir deux extensions ou plus, par exemple " .c.z " correspond 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 un programme correspondant lui est associé (par exemple, Acrobat Reader peut être associé à une extension " .pdf ").

Nom des fichiers

Tous les systèmes permettent des noms de fichiers comprenant entre un et huit caractères, certains autorisent jusqu'à deux cent cinquante-cinq caractères et permettent les caractères spéciaux. Les O.S. de la famille UNIX distinguent les caractères minuscules et majuscules (on dit qu'ils respectent la casse), Windows non.

Beaucoup de systèmes d'exploitation gèrent des noms de fichiers comprenant deux parties :

- La première partie située avant le point ;
- La seconde partie située après le point appelée extension et qui permet d'indiquer le type du fichier.

Sous UNIX, on peut avoir deux extensions ou plus, par exemple " .c.z " correspond 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 un programme correspondant lui est associé (par exemple, Acrobat Reader peut être associé à une extension " .pdf ").

Nom des fichiers

Tous les systèmes permettent des noms de fichiers comprenant entre un et huit caractères, certains autorisent jusqu'à deux cent cinquante-cinq caractères et permettent les caractères spéciaux. Les O.S. de la famille UNIX distinguent les caractères minuscules et majuscules (on dit qu'ils respectent la casse), Windows non.

Beaucoup de systèmes d'exploitation gèrent des noms de fichiers comprenant deux parties :

- La première partie située avant le point ;
- La seconde partie située après le point appelée extension et qui permet d'indiquer le type du fichier.

Sous UNIX, on peut avoir deux extensions ou plus, par exemple " .c.z " correspond 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 un programme correspondant lui est associé (par exemple, Acrobat Reader peut être associé à une extension " .pdf ").

Nom des fichiers

Tous les systèmes permettent des noms de fichiers comprenant entre un et huit caractères, certains autorisent jusqu'à deux cent cinquante-cinq caractères et permettent les caractères spéciaux. Les O.S. de la famille UNIX distinguent les caractères minuscules et majuscules (on dit qu'ils respectent la casse), Windows non. Beaucoup de systèmes d'exploitation gèrent des noms de fichiers comprenant deux parties :

- La première partie située avant le point ;
- La seconde partie située après le point appelée extension et qui permet d'indiquer le type du fichier.

Sous UNIX, on peut avoir deux extensions ou plus, par exemple " .c.z " correspond 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 un programme correspondant lui est associé (par exemple, Acrobat Reader peut être associé à une extension " .pdf ").

Nom des fichiers

Tous les systèmes permettent des noms de fichiers comprenant entre un et huit caractères, certains autorisent jusqu'à deux cent cinquante-cinq caractères et permettent les caractères spéciaux. Les O.S. de la famille UNIX distinguent les caractères minuscules et majuscules (on dit qu'ils respectent la casse), Windows non. Beaucoup de systèmes d'exploitation gèrent des noms de fichiers comprenant deux parties :

- La première partie située avant le point ;
- La seconde partie située après le point appelée extension et qui permet d'indiquer le type du fichier.

Sous UNIX, on peut avoir deux extensions ou plus, par exemple " .c.z " correspond 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 un programme correspondant lui est associé (par exemple, Acrobat Reader peut être associé à une extension " .pdf ").

Nom des fichiers

Tous les systèmes permettent des noms de fichiers comprenant entre un et huit caractères, certains autorisent jusqu'à deux cent cinquante-cinq caractères et permettent les caractères spéciaux. Les O.S. de la famille UNIX distinguent les caractères minuscules et majuscules (on dit qu'ils respectent la casse), Windows non.
Beaucoup de systèmes d'exploitation gèrent des noms de fichiers comprenant deux parties :

- La première partie située avant le point ;
- La seconde partie située après le point appelée extension et qui permet d'indiquer le type du fichier.

Sous UNIX, on peut avoir deux extensions ou plus, par exemple " .c.z " correspond 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 un programme correspondant lui est associé (par exemple, Acrobat Reader peut être associé à une extension " .pdf ").

Nom des fichiers

Tous les systèmes permettent des noms de fichiers comprenant entre un et huit caractères, certains autorisent jusqu'à deux cent cinquante-cinq caractères et permettent les caractères spéciaux. Les O.S. de la famille UNIX distinguent les caractères minuscules et majuscules (on dit qu'ils respectent la casse), Windows non. Beaucoup de systèmes d'exploitation gèrent des noms de fichiers comprenant deux parties :

- La première partie située avant le point ;
- La seconde partie située après le point appelée extension et qui permet d'indiquer le type du fichier.

Sous UNIX, on peut avoir deux extensions ou plus, par exemple " .c.z " correspond 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 un programme correspondant lui est associé (par exemple, Acrobat Reader peut être associé à une extension " .pdf ").

Nom des fichiers

Tous les systèmes permettent des noms de fichiers comprenant entre un et huit caractères, certains autorisent jusqu'à deux cent cinquante-cinq caractères et permettent les caractères spéciaux. Les O.S. de la famille UNIX distinguent les caractères minuscules et majuscules (on dit qu'ils respectent la casse), Windows non.
Beaucoup de systèmes d'exploitation gèrent des noms de fichiers comprenant deux parties :

- La première partie située avant le point ;
- La seconde partie située après le point appelée extension et qui permet d'indiquer le type du fichier.

Sous UNIX, on peut avoir deux extensions ou plus, par exemple " .c.z " correspond 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 un programme correspondant lui est associé (par exemple, Acrobat Reader peut être associé à une extension " .pdf ").

Nom des fichiers

Tous les systèmes permettent des noms de fichiers comprenant entre un et huit caractères, certains autorisent jusqu'à deux cent cinquante-cinq caractères et permettent les caractères spéciaux. Les O.S. de la famille UNIX distinguent les caractères minuscules et majuscules (on dit qu'ils respectent la casse), Windows non.
Beaucoup de systèmes d'exploitation gèrent des noms de fichiers comprenant deux parties :

- La première partie située avant le point ;
- La seconde partie située après le point appelée extension et qui permet d'indiquer le type du fichier.

Sous UNIX, on peut avoir deux extensions ou plus, par exemple " .c.z " correspond 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 un programme correspondant lui est associé (par exemple, Acrobat Reader peut être associé à une extension " .pdf ").

Nom des fichiers

Tous les systèmes permettent des noms de fichiers comprenant entre un et huit caractères, certains autorisent jusqu'à deux cent cinquante-cinq caractères et permettent les caractères spéciaux. Les O.S. de la famille UNIX distinguent les caractères minuscules et majuscules (on dit qu'ils respectent la casse), Windows non.
Beaucoup de systèmes d'exploitation gèrent des noms de fichiers comprenant deux parties :

- La première partie située avant le point ;
- La seconde partie située après le point appelée extension et qui permet d'indiquer le type du fichier.

Sous UNIX, on peut avoir deux extensions ou plus, par exemple " .c.z " correspond 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 un programme correspondant lui est associé (par exemple, Acrobat Reader peut être associé à une extension " .pdf ").

Nom des fichiers

Tous les systèmes permettent des noms de fichiers comprenant entre un et huit caractères, certains autorisent jusqu'à deux cent cinquante-cinq caractères et permettent les caractères spéciaux. Les O.S. de la famille UNIX distinguent les caractères minuscules et majuscules (on dit qu'ils respectent la casse), Windows non.
Beaucoup de systèmes d'exploitation gèrent des noms de fichiers comprenant deux parties :

- La première partie située avant le point ;
- La seconde partie située après le point appelée extension et qui permet d'indiquer le type du fichier.

Sous UNIX, on peut avoir deux extensions ou plus, par exemple " .c.z " correspond 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 un programme correspondant lui est associé (par exemple, Acrobat Reader peut être associé à une extension " .pdf ").

Nom des fichiers

Tous les systèmes permettent des noms de fichiers comprenant entre un et huit caractères, certains autorisent jusqu'à deux cent cinquante-cinq caractères et permettent les caractères spéciaux. Les O.S. de la famille UNIX distinguent les caractères minuscules et majuscules (on dit qu'ils respectent la casse), Windows non.
Beaucoup de systèmes d'exploitation gèrent des noms de fichiers comprenant deux parties :

- La première partie située avant le point ;
- La seconde partie située après le point appelée extension et qui permet d'indiquer le type du fichier.

Sous UNIX, on peut avoir deux extensions ou plus, par exemple " .c.z " correspond 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 un programme correspondant lui est associé (par exemple, Acrobat Reader peut être associé à une extension " .pdf ").

Structure des fichiers

Il existe trois structures possibles :

1. 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.
2. 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).
3. Dans la troisième, la plus élaborée, le fichier est un arbre d'enregistrement. Les enregistrement peuvent être de longueur différente, mais possède tous une clef, située à la même position. L'arbre est trié en fonction des clefs. On accède aux enregistrement en précisant la clef. Le Système de Gestion de Base de Données (SGBD) fait alors aussi office d'O.S.

Structure des fichiers

Il existe trois structures possibles :

1. 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.
2. 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).
3. Dans la troisième, la plus élaborée, le fichier est un arbre d'enregistrement. Les enregistrement peuvent être de longueur différente, mais possède tous une clef, située à la même position. L'arbre est trié en fonction des clefs. On accède aux enregistrement en précisant la clef. Le Système de Gestion de Base de Données (SGBD) fait alors aussi office d'O.S.

Structure des fichiers

Il existe trois structures possibles :

1. 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.
2. 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).
3. Dans la troisième, la plus élaborée, le fichier est un arbre d'enregistrement. Les enregistrement peuvent être de longueur différente, mais possède tous une clef, située à la même position. L'arbre est trié en fonction des clefs. On accède aux enregistrement en précisant la clef. Le Système de Gestion de Base de Données (SGBD) fait alors aussi office d'O.S.

Structure des fichiers

Il existe trois structures possibles :

1. 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.
2. 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).
3. Dans la troisième, la plus élaborée, le fichier est un arbre d'enregistrement. Les enregistrement peuvent être de longueur différente, mais possède tous une clef, située à la même position. L'arbre est trié en fonction des clefs. On accède aux enregistrement en précisant la clef. Le Système de Gestion de Base de Données (SGBD) fait alors aussi office d'O.S.

Structure des fichiers

Il existe trois structures possibles :

1. 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.
2. 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).
3. Dans la troisième, la plus élaborée, le fichier est un arbre d'enregistrement. Les enregistrement peuvent être de longueur différente, mais possède tous une clef, située à la même position. L'arbre est trié en fonction des clefs. On accède aux enregistrement en précisant la clef. Le Système de Gestion de Base de Données (SGBD) fait alors aussi office d'O.S.

Structure des fichiers

Il existe trois structures possibles :

1. 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.
2. 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).
3. Dans la troisième, la plus élaborée, le fichier est un arbre d'enregistrement. Les enregistrement peuvent être de longueur différente, mais possède tous une clef, située à la même position. L'arbre est trié en fonction des clefs. On accède aux enregistrement en précisant la clef. Le Système de Gestion de Base de Données (SGBD) fait alors aussi office d'O.S.

Structure des fichiers

Il existe trois structures possibles :

1. 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.
2. 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).
3. Dans la troisième, la plus élaborée, le fichier est un arbre d'enregistrement. Les enregistrement peuvent être de longueur différente, mais possède tous une clef, située à la même position. L'arbre est trié en fonction des clefs. On accède aux enregistrement en précisant la clef. Le Système de Gestion de Base de Données (SGBD) fait alors aussi office d'O.S.

Structure des fichiers

Il existe trois structures possibles :

1. 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.
2. 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).
3. Dans la troisième, la plus élaborée, le fichier est un arbre d'enregistrement. Les enregistrement peuvent être de longueur différente, mais possède tous une clef, située à la même position. L'arbre est trié en fonction des clefs. On accède aux enregistrement en précisant la clef. Le Système de Gestion de Base de Données (SGBD) fait alors aussi office d'O.S.

Structure des fichiers

Il existe trois structures possibles :

1. 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.
2. 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).
3. Dans la troisième, la plus élaborée, le fichier est un arbre d'enregistrement. Les enregistrement peuvent être de longueur différente, mais possède tous une clef, située à la même position. L'arbre est trié en fonction des clefs. On accède aux enregistrement en précisant la clef. Le Système de Gestion de Base de Données (SGBD) fait alors aussi office d'O.S.

Structure des fichiers

Il existe trois structures possibles :

1. 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.
2. 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).
3. Dans la troisième, la plus élaborée, le fichier est un arbre d'enregistrement. Les enregistrement peuvent être de longueur différente, mais possède tous une clef, située à la même position. L'arbre est trié en fonction des clefs. On accède aux enregistrement en précisant la clef. Le Système de Gestion de Base de Données (SGBD) fait alors aussi office d'O.S.

Structure des fichiers

Il existe trois structures possibles :

1. 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.
2. 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).
3. Dans la troisième, la plus élaborée, le fichier est un arbre d'enregistrement. Les enregistrement peuvent être de longueur différente, mais possède tous une clef, située à la même position. L'arbre est trié en fonction des clefs. On accède aux enregistrement en précisant la clef. Le Système de Gestion de Base de Données (SGBD) fait alors aussi office d'O.S.

Structure des fichiers

Il existe trois structures possibles :

1. 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.
2. 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).
3. Dans la troisième, la plus élaborée, le fichier est un arbre d'enregistrement. Les enregistrement peuvent être de longueur différente, mais possède tous une clef, située à la même position. L'arbre est trié en fonction des clefs. On accède aux enregistrement en précisant la clef. Le Système de Gestion de Base de Données (SGBD) fait alors aussi office d'O.S.

Structure des fichiers

Il existe trois structures possibles :

1. 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.
2. 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).
3. Dans la troisième, la plus élaborée, le fichier est un arbre d'enregistrement. Les enregistrement peuvent être de longueur différente, mais possède tous une clef, située à la même position. L'arbre est trié en fonction des clefs. On accède aux enregistrement en précisant la clef. Le Système de Gestion de Base de Données (SGBD) fait alors aussi office d'O.S.

Types de fichier

Il existe plusieurs types de fichiers :

- Les fichiers ordinaires : ils contiennent des programmes, des données de programmes, du texte, de la musique, de la vidéo, etc.
- Les répertoires : fichiers mémorisant la structure du système de fichiers, c'est-à-dire la façon dont sont stockés les fichiers sur l'ordinateur.
- Les fichiers spéciaux : ils servent à modéliser (représenter) les périphériques d'entrées/sorties (imprimantes, disques, etc.).

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. Un tel fichier contient une suite de bits dont seul un programme associé reconnaît sa structure interne. Par exemple, le logiciel Acrobat Reader reconnaît - et sait donc traiter - les fichiers de type " pdf ". Les systèmes d'exploitation doivent connaître la structure de leurs propres fichiers exécutables.

Types de fichier

Il existe plusieurs types de fichiers :

- Les fichiers ordinaires : ils contiennent des programmes, des données de programmes, du texte, de la musique, de la vidéo, etc.
- Les répertoires : fichiers mémorisant la structure du système de fichiers, c'est-à-dire la façon dont sont stockés les fichiers sur l'ordinateur.
- Les fichiers spéciaux : ils servent à modéliser (représenter) les périphériques d'entrées/sorties (imprimantes, disques, etc.).

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. Un tel fichier contient une suite de bits dont seul un programme associé reconnaît sa structure interne. Par exemple, le logiciel Acrobat Reader reconnaît - et sait donc traiter - les fichiers de type " pdf ". Les systèmes d'exploitation doivent connaître la structure de leurs propres fichiers exécutables.

Types de fichier

Il existe plusieurs types de fichiers :

- Les fichiers ordinaires : ils contiennent des programmes, des données de programmes, du texte, de la musique, de la vidéo, etc.
- Les répertoires : fichiers mémorisant la structure du système de fichiers, c'est-à-dire la façon dont sont stockés les fichiers sur l'ordinateur.
- Les fichiers spéciaux : ils servent à modéliser (représenter) les périphériques d'entrées/sorties (imprimantes, disques, etc.).

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. Un tel fichier contient une suite de bits dont seul un programme associé reconnaît sa structure interne. Par exemple, le logiciel Acrobat Reader reconnaît - et sait donc traiter - les fichiers de type " pdf ". Les systèmes d'exploitation doivent connaître la structure de leurs propres fichiers exécutables.

Types de fichier

Il existe plusieurs types de fichiers :

- Les fichiers ordinaires : ils contiennent des programmes, des données de programmes, du texte, de la musique, de la vidéo, etc.
- Les répertoires : fichiers mémorisant la structure du système de fichiers, c'est-à-dire la façon dont sont stockés les fichiers sur l'ordinateur.
- Les fichiers spéciaux : ils servent à modéliser (représenter) les périphériques d'entrées/sorties (imprimantes, disques, etc.).

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. Un tel fichier contient une suite de bits dont seul un programme associé reconnaît sa structure interne. Par exemple, le logiciel Acrobat Reader reconnaît - et sait donc traiter - les fichiers de type " pdf ". Les systèmes d'exploitation doivent connaître la structure de leurs propres fichiers exécutables.

Types de fichier

Il existe plusieurs types de fichiers :

- Les fichiers ordinaires : ils contiennent des programmes, des données de programmes, du texte, de la musique, de la vidéo, etc.
- Les répertoires : fichiers mémorisant la structure du système de fichiers, c'est-à-dire la façon dont sont stockés les fichiers sur l'ordinateur.
- Les fichiers spéciaux : ils servent à modéliser (représenter) les périphériques d'entrées/sorties (imprimantes, disques, etc.).

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. Un tel fichier contient une suite de bits dont seul un programme associé reconnaît sa structure interne. Par exemple, le logiciel Acrobat Reader reconnaît - et sait donc traiter - les fichiers de type " pdf ". Les systèmes d'exploitation doivent connaître la structure de leurs propres fichiers exécutables.

Types de fichier

Il existe plusieurs types de fichiers :

- Les fichiers ordinaires : ils contiennent des programmes, des données de programmes, du texte, de la musique, de la vidéo, etc.
- Les répertoires : fichiers mémorisant la structure du système de fichiers, c'est-à-dire la façon dont sont stockés les fichiers sur l'ordinateur.
- Les fichiers spéciaux : ils servent à modéliser (représenter) les périphériques d'entrées/sorties (imprimantes, disques, etc.).

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. Un tel fichier contient une suite de bits dont seul un programme associé reconnaît sa structure interne. Par exemple, le logiciel Acrobat Reader reconnaît - et sait donc traiter - les fichiers de type " pdf ". Les systèmes d'exploitation doivent connaître la structure de leurs propres fichiers exécutables.

Types de fichier

Il existe plusieurs types de fichiers :

- Les fichiers ordinaires : ils contiennent des programmes, des données de programmes, du texte, de la musique, de la vidéo, etc.
- Les répertoires : fichiers mémorisant la structure du système de fichiers, c'est-à-dire la façon dont sont stockés les fichiers sur l'ordinateur.
- Les fichiers spéciaux : ils servent à modéliser (représenter) les périphériques d'entrées/sorties (imprimantes, disques, etc.).

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. Un tel fichier contient une suite de bits dont seul un programme associé reconnaît sa structure interne. Par exemple, le logiciel Acrobat Reader reconnaît - et sait donc traiter - les fichiers de type " pdf ". Les systèmes d'exploitation doivent connaître la structure de leurs propres fichiers exécutables.

Types de fichier

Il existe plusieurs types de fichiers :

- Les fichiers ordinaires : ils contiennent des programmes, des données de programmes, du texte, de la musique, de la vidéo, etc.
- Les répertoires : fichiers mémorisant la structure du système de fichiers, c'est-à-dire la façon dont sont stockés les fichiers sur l'ordinateur.
- Les fichiers spéciaux : ils servent à modéliser (représenter) les périphériques d'entrées/sorties (imprimantes, disques, etc.).

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. Un tel fichier contient une suite de bits dont seul un programme associé reconnaît sa structure interne. Par exemple, le logiciel Acrobat Reader reconnaît - et sait donc traiter - les fichiers de type " pdf ". Les systèmes d'exploitation doivent connaître la structure de leurs propres fichiers exécutables.

Types de fichier

Il existe plusieurs types de fichiers :

- Les fichiers ordinaires : ils contiennent des programmes, des données de programmes, du texte, de la musique, de la vidéo, etc.
- Les répertoires : fichiers mémorisant la structure du système de fichiers, c'est-à-dire la façon dont sont stockés les fichiers sur l'ordinateur.
- Les fichiers spéciaux : ils servent à modéliser (représenter) les périphériques d'entrées/sorties (imprimantes, disques, etc.).

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. Un tel fichier contient une suite de bits dont seul un programme associé reconnaît sa structure interne. Par exemple, le logiciel Acrobat Reader reconnaît - et sait donc traiter - les fichiers de type " pdf ". Les systèmes d'exploitation doivent connaître la structure de leurs propres fichiers exécutables.

Types de fichier

Il existe plusieurs types de fichiers :

- Les fichiers ordinaires : ils contiennent des programmes, des données de programmes, du texte, de la musique, de la vidéo, etc.
- Les répertoires : fichiers mémorisant la structure du système de fichiers, c'est-à-dire la façon dont sont stockés les fichiers sur l'ordinateur.
- Les fichiers spéciaux : ils servent à modéliser (représenter) les périphériques d'entrées/sorties (imprimantes, disques, etc.).

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. Un tel fichier contient une suite de bits dont seul un programme associé reconnaît sa structure interne. Par exemple, le logiciel Acrobat Reader reconnaît - et sait donc traiter - les fichiers de type " pdf ". Les systèmes d'exploitation doivent connaître la structure de leurs propres fichiers exécutables.

Types de fichier

Il existe plusieurs types de fichiers :

- Les fichiers ordinaires : ils contiennent des programmes, des données de programmes, du texte, de la musique, de la vidéo, etc.
- Les répertoires : fichiers mémorisant la structure du système de fichiers, c'est-à-dire la façon dont sont stockés les fichiers sur l'ordinateur.
- Les fichiers spéciaux : ils servent à modéliser (représenter) les périphériques d'entrées/sorties (imprimantes, disques, etc.).

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. Un tel fichier contient une suite de bits dont seul un programme associé reconnaît sa structure interne. Par exemple, le logiciel Acrobat Reader reconnaît - et sait donc traiter - les fichiers de type " pdf ". Les systèmes d'exploitation doivent connaître la structure de leurs propres fichiers exécutables.

Types de fichier

Il existe plusieurs types de fichiers :

- Les fichiers ordinaires : ils contiennent des programmes, des données de programmes, du texte, de la musique, de la vidéo, etc.
- Les répertoires : fichiers mémorisant la structure du système de fichiers, c'est-à-dire la façon dont sont stockés les fichiers sur l'ordinateur.
- Les fichiers spéciaux : ils servent à modéliser (représenter) les périphériques d'entrées/sorties (imprimantes, disques, etc.).

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. Un tel fichier contient une suite de bits dont seul un programme associé reconnaît sa structure interne. Par exemple, le logiciel Acrobat Reader reconnaît - et sait donc traiter - les fichiers de type " pdf ". Les systèmes d'exploitation doivent connaître la structure de leurs propres fichiers exécutables.

Types de fichier

Il existe plusieurs types de fichiers :

- Les fichiers ordinaires : ils contiennent des programmes, des données de programmes, du texte, de la musique, de la vidéo, etc.
- Les répertoires : fichiers mémorisant la structure du système de fichiers, c'est-à-dire la façon dont sont stockés les fichiers sur l'ordinateur.
- Les fichiers spéciaux : ils servent à modéliser (représenter) les périphériques d'entrées/sorties (imprimantes, disques, etc.).

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. Un tel fichier contient une suite de bits dont seul un programme associé reconnaît sa structure interne. Par exemple, le logiciel Acrobat Reader reconnaît - et sait donc traiter - les fichiers de type " pdf ". Les systèmes d'exploitation doivent connaître la structure de leurs propres fichiers exécutables.

Par exemple, un fichier exécutable sous Unix est structuré de la façon suivante : un en-tête (header) comprenant un nombre magique qui identifie un exécutable, suivi de la taille du code, la taille des données, la taille BSS (Below Stack Segment), taille de la table des symboles, le point d'entrée (début des instructions), enfin un espace libre. Après cet en-tête suivent les indicateurs, le code, les données, les bits de translation, et la table des symboles (utilisée pour deboggage).

Par exemple, un fichier exécutable sous Unix est structuré de la façon suivante : un en-tête (header) comprenant un nombre magique qui identifie un exécutable, suivi de la taille du code, la taille des données, la taille BSS (Below Stack Segment), taille de la table des symboles, le point d'entrée (début des instructions), enfin un espace libre. Après cet en-tête suivent les indicateurs, le code, les données, les bits de translation, et la table des symboles (utilisée pour deboggage).

Par exemple, un fichier exécutable sous Unix est structuré de la façon suivante : un en-tête (header) comprenant un nombre magique qui identifie un exécutable, suivi de la taille du code, la taille des données, la taille BSS (Below Stack Segment), taille de la table des symboles, le point d'entrée (début des instructions), enfin un espace libre. Après cet en-tête suivent les indicateurs, le code, les données, les bits de translation, et la table des symboles (utilisée pour deboggage).

Par exemple, un fichier exécutable sous Unix est structuré de la façon suivante : un en-tête (header) comprenant un nombre magique qui identifie un exécutable, suivi de la taille du code, la taille des données, la taille BSS (Below Stack Segment), taille de la table des symboles, le point d'entrée (début des instructions), enfin un espace libre. Après cet en-tête suivent les indicateurs, le code, les données, les bits de translation, et la table des symboles (utilisée pour deboggage).

Par exemple, un fichier exécutable sous Unix est structuré de la façon suivante : un en-tête (header) comprenant un nombre magique qui identifie un exécutable, suivi de la taille du code, la taille des données, la taille BSS (Below Stack Segment), taille de la table des symboles, le point d'entrée (début des instructions), enfin un espace libre. Après cet en-tête suivent les indicateurs, le code, les données, les bits de translation, et la table des symboles (utilisée pour deboggage).

Par exemple, un fichier exécutable sous Unix est structuré de la façon suivante : un en-tête (header) comprenant un nombre magique qui identifie un exécutable, suivi de la taille du code, la taille des données, la taille BSS (Below Stack Segment), taille de la table des symboles, le point d'entrée (début des instructions), enfin un espace libre. Après cet en-tête suivent les indicateurs, le code, les données, les bits de translation, et la table des symboles (utilisée pour deboggage).

Par exemple, un fichier exécutable sous Unix est structuré de la façon suivante : un en-tête (header) comprenant un nombre magique qui identifie un exécutable, suivi de la taille du code, la taille des données, la taille BSS (Below Stack Segment), taille de la table des symboles, le point d'entrée (début des instructions), enfin un espace libre. Après cet en-tête suivent les indicateurs, le code, les données, les bits de translation, et la table des symboles (utilisée pour deboggage).

Par exemple, un fichier exécutable sous Unix est structuré de la façon suivante : un en-tête (header) comprenant un nombre magique qui identifie un exécutable, suivi de la taille du code, la taille des données, la taille BSS (Below Stack Segment), taille de la table des symboles, le point d'entrée (début des instructions), enfin un espace libre. Après cet en-tête suivent les indicateurs, le code, les données, les bits de translation, et la table des symboles (utilisée pour deboggage).

Par exemple, un fichier exécutable sous Unix est structuré de la façon suivante : un en-tête (header) comprenant un nombre magique qui identifie un exécutable, suivi de la taille du code, la taille des données, la taille BSS (Below Stack Segment), taille de la table des symboles, le point d'entrée (début des instructions), enfin un espace libre. Après cet en-tête suivent les indicateurs, le code, les données, les bits de translation, et la table des symboles (utilisée pour deboggage).

Par exemple, un fichier exécutable sous Unix est structuré de la façon suivante : un en-tête (header) comprenant un nombre magique qui identifie un exécutable, suivi de la taille du code, la taille des données, la taille BSS (Below Stack Segment), taille de la table des symboles, le point d'entrée (début des instructions), enfin un espace libre. Après cet en-tête suivent les indicateurs, le code, les données, les bits de translation, et la table des symboles (utilisée pour deboggage).

Par exemple, un fichier exécutable sous Unix est structuré de la façon suivante : un en-tête (header) comprenant un nombre magique qui identifie un exécutable, suivi de la taille du code, la taille des données, la taille BSS (Below Stack Segment), taille de la table des symboles, le point d'entrée (début des instructions), enfin un espace libre. Après cet en-tête suivent les indicateurs, le code, les données, les bits de translation, et la table des symboles (utilisée pour deboggage).

Par exemple, un fichier exécutable sous Unix est structuré de la façon suivante : un en-tête (header) comprenant un nombre magique qui identifie un exécutable, suivi de la taille du code, la taille des données, la taille BSS (Below Stack Segment), taille de la table des symboles, le point d'entrée (début des instructions), enfin un espace libre. Après cet en-tête suivent les indicateurs, le code, les données, les bits de translation, et la table des symboles (utilisée pour deboggage).

Par exemple, un fichier exécutable sous Unix est structuré de la façon suivante : un en-tête (header) comprenant un nombre magique qui identifie un exécutable, suivi de la taille du code, la taille des données, la taille BSS (Below Stack Segment), taille de la table des symboles, le point d'entrée (début des instructions), enfin un espace libre. Après cet en-tête suivent les indicateurs, le code, les données, les bits de translation, et la table des symboles (utilisée pour deboggage).

Par exemple, un fichier exécutable sous Unix est structuré de la façon suivante : un en-tête (header) comprenant un nombre magique qui identifie un exécutable, suivi de la taille du code, la taille des données, la taille BSS (Below Stack Segment), taille de la table des symboles, le point d'entrée (début des instructions), enfin un espace libre. Après cet en-tête suivent les indicateurs, le code, les données, les bits de translation, et la table des symboles (utilisée pour deboggage).

Attributs des fichiers (1/2)

En plus de son nom, à chaque fichier est associé un certain nombre d'informations, appelées attributs du fichier, qui varient selon l'O.S.
Exemples d'attributs :

- Attribut 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.

Attributs des fichiers (1/2)

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

Exemples d'attributs :

- Attribut 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.

Attributs des fichiers (1/2)

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

Exemples d'attributs :

- Attribut 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.

Attributs des fichiers (1/2)

En plus de son nom, à chaque fichier est associé un certain nombre d'informations, appelées attributs du fichier, qui varient selon l'O.S.
Exemples d'attributs :

- Attribut 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.

Attributs des fichiers (1/2)

En plus de son nom, à chaque fichier est associé un certain nombre d'informations, appelées attributs du fichier, qui varient selon l'O.S.
Exemples d'attributs :

- Attribut 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.

Attributs des fichiers (1/2)

En plus de son nom, à chaque fichier est associé un certain nombre d'informations, appelées attributs du fichier, qui varient selon l'O.S.
Exemples d'attributs :

- Attribut 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.

Attributs des fichiers (1/2)

En plus de son nom, à chaque fichier est associé un certain nombre d'informations, appelées attributs du fichier, qui varient selon l'O.S.
Exemples d'attributs :

- Attribut 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.

Attributs des fichiers (1/2)

En plus de son nom, à chaque fichier est associé un certain nombre d'informations, appelées attributs du fichier, qui varient selon l'O.S.
Exemples d'attributs :

- Attribut 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.

Attributs des fichiers (1/2)

En plus de son nom, à chaque fichier est associé un certain nombre d'informations, appelées attributs du fichier, qui varient selon l'O.S.
Exemples d'attributs :

- Attribut 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.

Attributs des fichiers (1/2)

En plus de son nom, à chaque fichier est associé un certain nombre d'informations, appelées attributs du fichier, qui varient selon l'O.S.
Exemples d'attributs :

- Attribut 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.

Attributs des fichiers (1/2)

En plus de son nom, à chaque fichier est associé un certain nombre d'informations, appelées attributs du fichier, qui varient selon l'O.S.
Exemples d'attributs :

- Attribut 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.

Attributs des fichiers (1/2)

En plus de son nom, à chaque fichier est associé un certain nombre d'informations, appelées attributs du fichier, qui varient selon l'O.S.
Exemples d'attributs :

- Attribut 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.

Attributs des fichiers (1/2)

En plus de son nom, à chaque fichier est associé un certain nombre d'informations, appelées attributs du fichier, qui varient selon l'O.S.
Exemples d'attributs :

- Attribut 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.

Attributs des fichiers (1/2)

En plus de son nom, à chaque fichier est associé un certain nombre d'informations, appelées attributs du fichier, qui varient selon l'O.S.
Exemples d'attributs :

- Attribut 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.

Attributs des fichiers (1/2)

En plus de son nom, à chaque fichier est associé un certain nombre d'informations, appelées attributs du fichier, qui varient selon l'O.S.
Exemples d'attributs :

- Attribut 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.

Attributs des fichiers (1/2)

En plus de son nom, à chaque fichier est associé un certain nombre d'informations, appelées attributs du fichier, qui varient selon l'O.S.
Exemples d'attributs :

- Attribut 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.

Attributs des fichiers (1/2)

En plus de son nom, à chaque fichier est associé un certain nombre d'informations, appelées attributs du fichier, qui varient selon l'O.S.
Exemples d'attributs :

- Attribut 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.

Attributs des fichiers (1/2)

En plus de son nom, à chaque fichier est associé un certain nombre d'informations, appelées attributs du fichier, qui varient selon l'O.S.
Exemples d'attributs :

- Attribut 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.

Attributs des fichiers (1/2)

En plus de son nom, à chaque fichier est associé un certain nombre d'informations, appelées attributs du fichier, qui varient selon l'O.S.
Exemples d'attributs :

- Attribut 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.

Attributs des fichiers (2/2)

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

- Longueur d'un enregistrement ;
- Position de la clef ;
- Longueur de la clef.

Attributs des fichiers (2/2)

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

- Longueur d'un enregistrement ;
- Position de la clef ;
- Longueur de la clef.

Attributs des fichiers (2/2)

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

- Longueur d'un enregistrement ;
- Position de la clef ;
- Longueur de la clef.

Attributs des fichiers (2/2)

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

- Longueur d'un enregistrement ;
- Position de la clef ;
- Longueur de la clef.

Opérations sur les fichiers

Pour manipuler l'ordinateur, notamment les fichiers, on utilise les services de l'O.S. via un appel système.

- Opération create : opération de création sans donnée, indique le nom et certains paramètres ;
- 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 un certain endroit du fichier ;
- Opération read : lire à partir de la position courante ;
- Opération write : écrire à partir de la position courante (si on est en fin de fichier, alors cela augmente sa taille) ;
- Opération get attribut : pour obtenir certains attributs ;
- Opération set attribut : pour modifier certains attributs.

Opérations sur les fichiers

Pour manipuler l'ordinateur, notamment les fichiers, on utilise les services de l'O.S. via un appel système.

- Opération create : opération de création sans donnée, indique le nom et certains paramètres ;
- 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 un certain endroit du fichier ;
- Opération read : lire à partir de la position courante ;
- Opération write : écrire à partir de la position courante (si on est en fin de fichier, alors cela augmente sa taille) ;
- Opération get attribut : pour obtenir certains attributs ;
- Opération set attribut : pour modifier certains attributs.

Opérations sur les fichiers

Pour manipuler l'ordinateur, notamment les fichiers, on utilise les services de l'O.S. via un appel système.

- Opération create : opération de création sans donnée, indique le nom et certains paramètres ;
- 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 un certain endroit du fichier ;
- Opération read : lire à partir de la position courante ;
- Opération write : écrire à partir de la position courante (si on est en fin de fichier, alors cela augmente sa taille) ;
- Opération get attribut : pour obtenir certains attributs ;
- Opération set attribut : pour modifier certains attributs.

Opérations sur les fichiers

Pour manipuler l'ordinateur, notamment les fichiers, on utilise les services de l'O.S. via un appel système.

- Opération create : opération de création sans donnée, indique le nom et certains paramètres ;
- 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 un certain endroit du fichier ;
- Opération read : lire à partir de la position courante ;
- Opération write : écrire à partir de la position courante (si on est en fin de fichier, alors cela augmente sa taille) ;
- Opération get attribut : pour obtenir certains attributs ;
- Opération set attribut : pour modifier certains attributs.

Opérations sur les fichiers

Pour manipuler l'ordinateur, notamment les fichiers, on utilise les services de l'O.S. via un appel système.

- Opération create : opération de création sans donnée, indique le nom et certains paramètres ;
- 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 un certain endroit du fichier ;
- Opération read : lire à partir de la position courante ;
- Opération write : écrire à partir de la position courante (si on est en fin de fichier, alors cela augmente sa taille) ;
- Opération get attribut : pour obtenir certains attributs ;
- Opération set attribut : pour modifier certains attributs.

Opérations sur les fichiers

Pour manipuler l'ordinateur, notamment les fichiers, on utilise les services de l'O.S. via un appel système.

- Opération create : opération de création sans donnée, indique le nom et certains paramètres ;
- 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 un certain endroit du fichier ;
- Opération read : lire à partir de la position courante ;
- Opération write : écrire à partir de la position courante (si on est en fin de fichier, alors cela augmente sa taille) ;
- Opération get attribut : pour obtenir certains attributs ;
- Opération set attribut : pour modifier certains attributs.

Opérations sur les fichiers

Pour manipuler l'ordinateur, notamment les fichiers, on utilise les services de l'O.S. via un appel système.

- Opération create : opération de création sans donnée, indique le nom et certains paramètres ;
- 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 un certain endroit du fichier ;
- Opération read : lire à partir de la position courante ;
- Opération write : écrire à partir de la position courante (si on est en fin de fichier, alors cela augmente sa taille) ;
- Opération get attribut : pour obtenir certains attributs ;
- Opération set attribut : pour modifier certains attributs.

Opérations sur les fichiers

Pour manipuler l'ordinateur, notamment les fichiers, on utilise les services de l'O.S. via un appel système.

- Opération create : opération de création sans donnée, indique le nom et certains paramètres ;
- 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 un certain endroit du fichier ;
- Opération read : lire à partir de la position courante ;
- Opération write : écrire à partir de la position courante (si on est en fin de fichier, alors cela augmente sa taille) ;
- Opération get attribut : pour obtenir certains attributs ;
- Opération set attribut : pour modifier certains attributs.

Opérations sur les fichiers

Pour manipuler l'ordinateur, notamment les fichiers, on utilise les services de l'O.S. via un appel système.

- Opération create : opération de création sans donnée, indique le nom et certains paramètres ;
- 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 un certain endroit du fichier ;
- Opération read : lire à partir de la position courante ;
- Opération write : écrire à partir de la position courante (si on est en fin de fichier, alors cela augmente sa taille) ;
- Opération get attribut : pour obtenir certains attributs ;
- Opération set attribut : pour modifier certains attributs.

Opérations sur les fichiers

Pour manipuler l'ordinateur, notamment les fichiers, on utilise les services de l'O.S. via un appel système.

- Opération create : opération de création sans donnée, indique le nom et certains paramètres ;
- 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 un certain endroit du fichier ;
- Opération read : lire à partir de la position courante ;
- Opération write : écrire à partir de la position courante (si on est en fin de fichier, alors cela augmente sa taille) ;
- Opération get attribut : pour obtenir certains attributs ;
- Opération set attribut : pour modifier certains attributs.

Opérations sur les fichiers

Pour manipuler l'ordinateur, notamment les fichiers, on utilise les services de l'O.S. via un appel système.

- Opération create : opération de création sans donnée, indique le nom et certains paramètres ;
- 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 un certain endroit du fichier ;
- Opération read : lire à partir de la position courante ;
- Opération write : écrire à partir de la position courante (si on est en fin de fichier, alors cela augmente sa taille) ;
- Opération get attribut : pour obtenir certains attributs ;
- Opération set attribut : pour modifier certains attributs.

Opérations sur les fichiers

Pour manipuler l'ordinateur, notamment les fichiers, on utilise les services de l'O.S. via un appel système.

- Opération create : opération de création sans donnée, indique le nom et certains paramètres ;
- 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 un certain endroit du fichier ;
- Opération read : lire à partir de la position courante ;
- Opération write : écrire à partir de la position courante (si on est en fin de fichier, alors cela augmente sa taille) ;
- Opération get attribut : pour obtenir certains attributs ;
- Opération set attribut : pour modifier certains attributs.

Opérations sur les fichiers

Pour manipuler l'ordinateur, notamment les fichiers, on utilise les services de l'O.S. via un appel système.

- Opération create : opération de création sans donnée, indique le nom et certains paramètres ;
- 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 un certain endroit du fichier ;
- Opération read : lire à partir de la position courante ;
- Opération write : écrire à partir de la position courante (si on est en fin de fichier, alors cela augmente sa taille) ;
- Opération get attribut : pour obtenir certains attributs ;
- Opération set attribut : pour modifier certains attributs.

Opérations sur les fichiers

Pour manipuler l'ordinateur, notamment les fichiers, on utilise les services de l'O.S. via un appel système.

- Opération create : opération de création sans donnée, indique le nom et certains paramètres ;
- 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 un certain endroit du fichier ;
- Opération read : lire à partir de la position courante ;
- Opération write : écrire à partir de la position courante (si on est en fin de fichier, alors cela augmente sa taille) ;
- Opération get attribut : pour obtenir certains attributs ;
- Opération set attribut : pour modifier certains attributs.

Opérations sur les fichiers

Pour manipuler l'ordinateur, notamment les fichiers, on utilise les services de l'O.S. via un appel système.

- Opération create : opération de création sans donnée, indique le nom et certains paramètres ;
- 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 un certain endroit du fichier ;
- Opération read : lire à partir de la position courante ;
- Opération write : écrire à partir de la position courante (si on est en fin de fichier, alors cela augmente sa taille) ;
- Opération get attribut : pour obtenir certains attributs ;
- Opération set attribut : pour modifier certains attributs.

Opérations sur les fichiers

Pour manipuler l'ordinateur, notamment les fichiers, on utilise les services de l'O.S. via un appel système.

- Opération create : opération de création sans donnée, indique le nom et certains paramètres ;
- 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 un certain endroit du fichier ;
- Opération read : lire à partir de la position courante ;
- Opération write : écrire à partir de la position courante (si on est en fin de fichier, alors cela augmente sa taille) ;
- Opération get attribut : pour obtenir certains attributs ;
- Opération set attribut : pour modifier certains attributs.

Opérations sur les fichiers

Pour manipuler l'ordinateur, notamment les fichiers, on utilise les services de l'O.S. via un appel système.

- Opération create : opération de création sans donnée, indique le nom et certains paramètres ;
- 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 un certain endroit du fichier ;
- Opération read : lire à partir de la position courante ;
- Opération write : écrire à partir de la position courante (si on est en fin de fichier, alors cela augmente sa taille) ;
- Opération get attribut : pour obtenir certains attributs ;
- Opération set attribut : pour modifier certains attributs.

Opérations sur les fichiers

Pour manipuler l'ordinateur, notamment les fichiers, on utilise les services de l'O.S. via un appel système.

- Opération create : opération de création sans donnée, indique le nom et certains paramètres ;
- 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 un certain endroit du fichier ;
- Opération read : lire à partir de la position courante ;
- Opération write : écrire à partir de la position courante (si on est en fin de fichier, alors cela augmente sa taille) ;
- Opération get attribut : pour obtenir certains attributs ;
- Opération set attribut : pour modifier certains attributs.

Opérations sur les fichiers

Pour manipuler l'ordinateur, notamment les fichiers, on utilise les services de l'O.S. via un appel système.

- Opération create : opération de création sans donnée, indique le nom et certains paramètres ;
- 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 un certain endroit du fichier ;
- Opération read : lire à partir de la position courante ;
- Opération write : écrire à partir de la position courante (si on est en fin de fichier, alors cela augmente sa taille) ;
- Opération get attribut : pour obtenir certains attributs ;
- Opération set attribut : pour modifier certains attributs.

Opérations sur les fichiers

Pour manipuler l'ordinateur, notamment les fichiers, on utilise les services de l'O.S. via un appel système.

- Opération create : opération de création sans donnée, indique le nom et certains paramètres ;
- 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 un certain endroit du fichier ;
- Opération read : lire à partir de la position courante ;
- Opération write : écrire à partir de la position courante (si on est en fin de fichier, alors cela augmente sa taille) ;
- Opération get attribut : pour obtenir certains attributs ;
- Opération set attribut : pour modifier certains attributs.

Les répertoires (1/5)

Les systèmes de fichiers ont en général des répertoires (directory) ou dossiers qui gardent une trace des fichiers, ces répertoires étant eux-même traiter comme des fichiers dans la plupart des systèmes. Le répertoire racine est celui qui contient tous les autres répertoires. Il existe plusieurs niveaux de répertoires selon les systèmes.

Les répertoires (1/5)

Les systèmes de fichiers ont en général des répertoires (directory) ou dossiers qui gardent une trace des fichiers, ces répertoires étant eux-même traiter comme des fichiers dans la plupart des systèmes. Le répertoire racine est celui qui contient tous les autres répertoires. Il existe plusieurs niveaux de répertoires selon les systèmes.

Les répertoires (1/5)

Les systèmes de fichiers ont en général des répertoires (directory) ou dossiers qui gardent une trace des fichiers, ces répertoires étant eux-même traiter comme des fichiers dans la plupart des systèmes. Le répertoire racine est celui qui contient tous les autres répertoires. Il existe plusieurs niveaux de répertoires selon les systèmes.

Les répertoires (1/5)

Les systèmes de fichiers ont en général des répertoires (directory) ou dossiers qui gardent une trace des fichiers, ces répertoires étant eux-même traiter comme des fichiers dans la plupart des systèmes. Le répertoire racine est celui qui contient tous les autres répertoires. Il existe plusieurs niveaux de répertoires selon les systèmes.

Les répertoires (1/5)

Les systèmes de fichiers ont en général des répertoires (directory) ou dossiers qui gardent une trace des fichiers, ces répertoires étant eux-même traiter comme des fichiers dans la plupart des systèmes. Le répertoire racine est celui qui contient tous les autres répertoires. Il existe plusieurs niveaux de répertoires selon les systèmes.

Les répertoires (2/5)

- Systèmes à un seul niveau de répertoires : le répertoire racine contient tous les fichiers (en vrac !).
- 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 (ou arborescents) : c'est le système que nous connaissons sur les PC actuels. Les répertoires et fichiers sont organisés sous la forme d'un arbre dont la racine est le répertoire racine (!). Les feuilles de cet arbres sont les fichiers. Une branche entre deux répertoires (ou entre un répertoire et un fichier) s'interprète (dans le sens racine $\rightarrow$ feuilles) comme une inclusion.

Les répertoires (2/5)

- Systèmes à un seul niveau de répertoires : le répertoire racine contient tous les fichiers (en vrac !).
- 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 (ou arborescents) : c'est le système que nous connaissons sur les PC actuels. Les répertoires et fichiers sont organisés sous la forme d'un arbre dont la racine est le répertoire racine (!). Les feuilles de cet arbres sont les fichiers. Une branche entre deux répertoires (ou entre un répertoire et un fichier) s'interprète (dans le sens racine $\rightarrow$ feuilles) comme une inclusion.

Les répertoires (2/5)

- Systèmes à un seul niveau de répertoires : le répertoire racine contient tous les fichiers (en vrac !).
- 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 (ou arborescents) : c'est le système que nous connaissons sur les PC actuels. Les répertoires et fichiers sont organisés sous la forme d'un arbre dont la racine est le répertoire racine (!). Les feuilles de cet arbres sont les fichiers. Une branche entre deux répertoires (ou entre un répertoire et un fichier) s'interprète (dans le sens racine → feuilles) comme une inclusion.

Les répertoires (2/5)

- Systèmes à un seul niveau de répertoires : le répertoire racine contient tous les fichiers (en vrac !).
- 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 (ou arborescents) : c'est le système que nous connaissons sur les PC actuels. Les répertoires et fichiers sont organisés sous la forme d'un arbre dont la racine est le répertoire racine (!). Les feuilles de cet arbres sont les fichiers. Une branche entre deux répertoires (ou entre un répertoire et un fichier) s'interprète (dans le sens racine → feuilles) comme une inclusion.

Les répertoires (2/5)

- Systèmes à un seul niveau de répertoires : le répertoire racine contient tous les fichiers (en vrac !).
- 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 (ou arborescents) : c'est le système que nous connaissons sur les PC actuels. Les répertoires et fichiers sont organisés sous la forme d'un arbre dont la racine est le répertoire racine (!). Les feuilles de cet arbres sont les fichiers. Une branche entre deux répertoires (ou entre un répertoire et un fichier) s'interprète (dans le sens racine → feuilles) comme une inclusion.

Les répertoires (2/5)

- Systèmes à un seul niveau de répertoires : le répertoire racine contient tous les fichiers (en vrac !).
- 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 (ou arborescents) : c'est le système que nous connaissons sur les PC actuels. Les répertoires et fichiers sont organisés sous la forme d'un arbre dont la racine est le répertoire racine (!). Les feuilles de cet arbres sont les fichiers. Une branche entre deux répertoires (ou entre un répertoire et un fichier) s'interprète (dans le sens racine → feuilles) comme une inclusion.

Les répertoires (2/5)

- Systèmes à un seul niveau de répertoires : le répertoire racine contient tous les fichiers (en vrac !).
- 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 (ou arborescents) : c'est le système que nous connaissons sur les PC actuels. Les répertoires et fichiers sont organisés sous la forme d'un arbre dont la racine est le répertoire racine (!). Les feuilles de cet arbres sont les fichiers. Une branche entre deux répertoires (ou entre un répertoire et un fichier) s'interprète (dans le sens racine → feuilles) comme une inclusion.

Les répertoires (2/5)

- Systèmes à un seul niveau de répertoires : le répertoire racine contient tous les fichiers (en vrac !).
- 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 (ou arborescents) : c'est le système que nous connaissons sur les PC actuels. Les répertoires et fichiers sont organisés sous la forme d'un arbre dont la racine est le répertoire racine (!). Les feuilles de cet arbres sont les fichiers. Une branche entre deux répertoires (ou entre un répertoire et un fichier) s'interprète (dans le sens racine → feuilles) comme une inclusion.

Les répertoires (2/5)

- Systèmes à un seul niveau de répertoires : le répertoire racine contient tous les fichiers (en vrac !).
- 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 (ou arborescents) : c'est le système que nous connaissons sur les PC actuels. Les répertoires et fichiers sont organisés sous la forme d'un arbre dont la racine est le répertoire racine (!). Les feuilles de cet arbres sont les fichiers. Une branche entre deux répertoires (ou entre un répertoire et un fichier) s'interprète (dans le sens racine → feuilles) comme une inclusion.

Les répertoires (2/5)

- Systèmes à un seul niveau de répertoires : le répertoire racine contient tous les fichiers (en vrac !).
- 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 (ou arborescents) : c'est le système que nous connaissons sur les PC actuels. Les répertoires et fichiers sont organisés sous la forme d'un arbre dont la racine est le répertoire racine (!). Les feuilles de cet arbres sont les fichiers. Une branche entre deux répertoires (ou entre un répertoire et un fichier) s'interprète (dans le sens racine $\rightarrow$ feuilles) comme une inclusion.

Les répertoires (3/5)

Dans un système hiérarchique, il faut pouvoir préciser les chemins d'accès c'est-à-dire la façon de parcourir la hiérarchie (l'arborescence) afin d'aboutir à un certain fichier (de cette hiérarchie). On distingue les chemins d'accès :

- absolus, qui commencent par la racine, par ex. sous UNIX : " /user/joe/pgmC/exo1.c " ou sous Windows : " \user\joe\pgmC\exo1.c ".
- relatifs, qui partent du répertoire courant (répertoire de travail), par exemple, si on travaille sous le répertoire " /user/joe ", alors il suffit de préciser " pgmC/exo1.c " pour accéder au fichier " exo1.c ".

Les répertoires (3/5)

Dans un système hiérarchique, il faut pouvoir préciser les chemins d'accès c'est-à-dire la façon de parcourir la hiérarchie (l'arborescence) afin d'aboutir à un certain fichier (de cette hiérarchie). On distingue les chemins d'accès :

- absolus, qui commencent par la racine, par ex. sous UNIX : " /user/joe/pgmC/exo1.c " ou sous Windows : " \user\joe\pgmC\exo1.c ".
- relatifs, qui partent du répertoire courant (répertoire de travail), par exemple, si on travaille sous le répertoire " /user/joe ", alors il suffit de préciser " pgmC/exo1.c " pour accéder au fichier " exo1.c ".

Les répertoires (3/5)

Dans un système hiérarchique, il faut pouvoir préciser les chemins d'accès c'est-à-dire la façon de parcourir la hiérarchie (l'arborescence) afin d'aboutir à un certain fichier (de cette hiérarchie). On distingue les chemins d'accès :

- absolus, qui commencent par la racine, par ex. sous UNIX : " /user/joe/pgmC/exo1.c " ou sous Windows : " \user\joe\pgmC\exo1.c ".
- relatifs, qui partent du répertoire courant (répertoire de travail), par exemple, si on travaille sous le répertoire " /user/joe ", alors il suffit de préciser " pgmC/exo1.c " pour accéder au fichier " exo1.c ".

Les répertoires (3/5)

Dans un système hiérarchique, il faut pouvoir préciser les chemins d'accès c'est-à-dire la façon de parcourir la hiérarchie (l'arborescence) afin d'aboutir à un certain fichier (de cette hiérarchie). On distingue les chemins d'accès :

- absolus, qui commencent par la racine, par ex. sous UNIX : " /user/joe/pgmC/exo1.c " ou sous Windows : " \user\joe\pgmC\exo1.c ".
- relatifs, qui partent du répertoire courant (répertoire de travail), par exemple, si on travaille sous le répertoire " /user/joe ", alors il suffit de préciser " pgmC/exo1.c " pour accéder au fichier " exo1.c ".

Les répertoires (3/5)

Dans un système hiérarchique, il faut pouvoir préciser les chemins d'accès c'est-à-dire la façon de parcourir la hiérarchie (l'arborescence) afin d'aboutir à un certain fichier (de cette hiérarchie). On distingue les chemins d'accès :

- absolus, qui commencent par la racine, par ex. sous UNIX : " /user/joe/pgmC/exo1.c " ou sous Windows : " \user\joe\pgmC\exo1.c ".
- relatifs, qui partent du répertoire courant (répertoire de travail), par exemple, si on travaille sous le répertoire " /user/joe ", alors il suffit de préciser " pgmC/exo1.c " pour accéder au fichier " exo1.c ".

Les répertoires (3/5)

Dans un système hiérarchique, il faut pouvoir préciser les chemins d'accès c'est-à-dire la façon de parcourir la hiérarchie (l'arborescence) afin d'aboutir à un certain fichier (de cette hiérarchie). On distingue les chemins d'accès :

- absolus, qui commencent par la racine, par ex. sous UNIX : " /user/joe/pgmC/exo1.c " ou sous Windows : " \user\joe\pgmC\exo1.c ".
- relatifs, qui partent du répertoire courant (répertoire de travail), par exemple, si on travaille sous le répertoire " /user/joe ", alors il suffit de préciser " pgmC/exo1.c " pour accéder au fichier " exo1.c ".

Les répertoires (3/5)

Dans un système hiérarchique, il faut pouvoir préciser les chemins d'accès c'est-à-dire la façon de parcourir la hiérarchie (l'arborescence) afin d'aboutir à un certain fichier (de cette hiérarchie). On distingue les chemins d'accès :

- absolus, qui commencent par la racine, par ex. sous UNIX : " /user/joe/pgmC/exo1.c " ou sous Windows : " \user\joe\pgmC\exo1.c ".
- relatifs, qui partent du répertoire courant (répertoire de travail), par exemple, si on travaille sous le répertoire " /user/joe ", alors il suffit de préciser " pgmC/exo1.c " pour accéder au fichier " exo1.c ".

Les répertoires (3/5)

Dans un système hiérarchique, il faut pouvoir préciser les chemins d'accès c'est-à-dire la façon de parcourir la hiérarchie (l'arborescence) afin d'aboutir à un certain fichier (de cette hiérarchie). On distingue les chemins d'accès :

- absolus, qui commencent par la racine, par ex. sous UNIX : " /user/joe/pgmC/exo1.c " ou sous Windows : " \user\joe\pgmC\exo1.c ".
- relatifs, qui partent du répertoire courant (répertoire de travail), par exemple, si on travaille sous le répertoire " /user/joe ", alors il suffit de préciser " pgmC/exo1.c " pour accéder au fichier " exo1.c ".

Les répertoires (3/5)

Dans un système hiérarchique, il faut pouvoir préciser les chemins d'accès c'est-à-dire la façon de parcourir la hiérarchie (l'arborescence) afin d'aboutir à un certain fichier (de cette hiérarchie). On distingue les chemins d'accès :

- absolus, qui commencent par la racine, par ex. sous UNIX : " /user/joe/pgmC/exo1.c " ou sous Windows : " \user\joe\pgmC\exo1.c ".
- relatifs, qui partent du répertoire courant (répertoire de travail), par exemple, si on travaille sous le répertoire " /user/joe ", alors il suffit de préciser " pgmC/exo1.c " pour accéder au fichier " exo1.c ".

Les répertoires (4/5)

Sous Linux, une écriture de la forme " ./toto " est un chemin relatif vers le fichier toto en partant du répertoire courant, " ../rep-courant/toto " est un chemin relatif vers le fichier toto en partant du répertoire parent (on remonte d'un répertoire) du répertoire courant rep-courant. Il est tout à fait possible de remonter de plusieurs répertoires : " ../../tata/tutu " permet de remonter de deux répertoires à partir d'un répertoire courant.

Les répertoires (4/5)

Sous Linux, une écriture de la forme " ./toto " est un chemin relatif vers le fichier toto en partant du répertoire courant, " ../rep-courant/toto " est un chemin relatif vers le fichier toto en partant du répertoire parent (on remonte d'un répertoire) du répertoire courant rep-courant. Il est tout à fait possible de remonter de plusieurs répertoires : " ../../tata/tutu " permet de remonter de deux répertoires à partir d'un répertoire courant.

Les répertoires (4/5)

Sous Linux, une écriture de la forme " ./toto " est un chemin relatif vers le fichier toto en partant du répertoire courant, " ../rep-courant/toto " est un chemin relatif vers le fichier toto en partant du répertoire parent (on remonte d'un répertoire) du répertoire courant rep-courant. Il est tout à fait possible de remonter de plusieurs répertoires : " ../../tata/tutu " permet de remonter de deux répertoires à partir d'un répertoire courant.

Les répertoires (4/5)

Sous Linux, une écriture de la forme " ./toto " est un chemin relatif vers le fichier toto en partant du répertoire courant, " ../rep-courant/toto " est un chemin relatif vers le fichier toto en partant du répertoire parent (on remonte d'un répertoire) du répertoire courant rep-courant. Il est tout à fait possible de remonter de plusieurs répertoires : " ../../tata/tutu " permet de remonter de deux répertoires à partir d'un répertoire courant.

Les répertoires (5/5)

Quelques appels systèmes concernant les répertoires :

- Create : qui crée un répertoire vide ;
- Delete : qui supprime un répertoire vide ;
- Open dir : charge certaines informations 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.

Les répertoires (5/5)

Quelques appels systèmes concernant les répertoires :

- Create : qui crée un répertoire vide ;
- Delete : qui supprime un répertoire vide ;
- Open dir : charge certaines informations 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.

Les répertoires (5/5)

Quelques appels systèmes concernant les répertoires :

- Create : qui crée un répertoire vide ;
- Delete : qui supprime un répertoire vide ;
- Open dir : charge certaines informations 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.

Les répertoires (5/5)

Quelques appels systèmes concernant les répertoires :

- Create : qui crée un répertoire vide ;
- Delete : qui supprime un répertoire vide ;
- Open dir : charge certaines informations 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.

Les répertoires (5/5)

Quelques appels systèmes concernant les répertoires :

- Create : qui crée un répertoire vide ;
- Delete : qui supprime un répertoire vide ;
- Open dir : charge certaines informations 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.

Les répertoires (5/5)

Quelques appels systèmes concernant les répertoires :

- Create : qui crée un répertoire vide ;
- Delete : qui supprime un répertoire vide ;
- Open dir : charge certaines informations 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.

Les répertoires (5/5)

Quelques appels systèmes concernant les répertoires :

- Create : qui crée un répertoire vide ;
- Delete : qui supprime un répertoire vide ;
- Open dir : charge certaines informations 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.

Les répertoires (5/5)

Quelques appels systèmes concernant les répertoires :

- Create : qui crée un répertoire vide ;
- Delete : qui supprime un répertoire vide ;
- Open dir : charge certaines informations 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.

Les répertoires (5/5)

Quelques appels systèmes concernant les répertoires :

- Create : qui crée un répertoire vide ;
- Delete : qui supprime un répertoire vide ;
- Open dir : charge certaines informations 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.

Les répertoires (5/5)

Quelques appels systèmes concernant les répertoires :

- Create : qui crée un répertoire vide ;
- Delete : qui supprime un répertoire vide ;
- Open dir : charge certaines informations 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.

Les répertoires (5/5)

Quelques appels systèmes concernant les répertoires :

- Create : qui crée un répertoire vide ;
- Delete : qui supprime un répertoire vide ;
- Open dir : charge certaines informations 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.

Les répertoires (5/5)

Quelques appels systèmes concernant les répertoires :

- Create : qui crée un répertoire vide ;
- Delete : qui supprime un répertoire vide ;
- Open dir : charge certaines informations 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.

Les répertoires (5/5)

Quelques appels systèmes concernant les répertoires :

- Create : qui crée un répertoire vide ;
- Delete : qui supprime un répertoire vide ;
- Open dir : charge certaines informations 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.

Les répertoires (5/5)

Quelques appels systèmes concernant les répertoires :

- Create : qui crée un répertoire vide ;
- Delete : qui supprime un répertoire vide ;
- Open dir : charge certaines informations 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.

Les répertoires (5/5)

Quelques appels systèmes concernant les répertoires :

- Create : qui crée un répertoire vide ;
- Delete : qui supprime un répertoire vide ;
- Open dir : charge certaines informations 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.

Implantation physique des fichiers (1/4)

Il existe plusieurs façon d'implanter (stocker) 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. Par contre l'accès aléatoire devient lent car il faut lire le premier mot de chaque bloc précédant le bloc cherché.

Implantation physique des fichiers (1/4)

Il existe plusieurs façon d'implanter (stocker) 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. Par contre l'accès aléatoire devient lent car il faut lire le premier mot de chaque bloc précédant le bloc cherché.

Implantation physique des fichiers (1/4)

Il existe plusieurs façon d'implanter (stocker) 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. Par contre l'accès aléatoire devient lent car il faut lire le premier mot de chaque bloc précédant le bloc cherché.

Implantation physique des fichiers (1/4)

Il existe plusieurs façon d'implanter (stocker) 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. Par contre l'accès aléatoire devient lent car il faut lire le premier mot de chaque bloc précédant le bloc cherché.

Implantation physique des fichiers (1/4)

Il existe plusieurs façon d'implanter (stocker) 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. Par contre l'accès aléatoire devient lent car il faut lire le premier mot de chaque bloc précédant le bloc cherché.

Implantation physique des fichiers (1/4)

Il existe plusieurs façon d'implanter (stocker) 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. Par contre l'accès aléatoire devient lent car il faut lire le premier mot de chaque bloc précédant le bloc cherché.

Implantation physique des fichiers (1/4)

Il existe plusieurs façon d'implanter (stocker) 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. Par contre l'accès aléatoire devient lent car il faut lire le premier mot de chaque bloc précédant le bloc cherché.

Implantation physique des fichiers (1/4)

Il existe plusieurs façon d'implanter (stocker) 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. Par contre l'accès aléatoire devient lent car il faut lire le premier mot de chaque bloc précédant le bloc cherché.

Implantation physique des fichiers (1/4)

Il existe plusieurs façon d'implanter (stocker) 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. Par contre l'accès aléatoire devient lent car il faut lire le premier mot de chaque bloc précédant le bloc cherché.

Implantation physique des fichiers (1/4)

Il existe plusieurs façon d'implanter (stocker) 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. Par contre l'accès aléatoire devient lent car il faut lire le premier mot de chaque bloc précédant le bloc cherché.

Implantation physique des fichiers (1/4)

Il existe plusieurs façon d'implanter (stocker) 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. Par contre l'accès aléatoire devient lent car il faut lire le premier mot de chaque bloc précédant le bloc cherché.

Implantation physique des fichiers (1/4)

Il existe plusieurs façon d'implanter (stocker) 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. Par contre l'accès aléatoire devient lent car il faut lire le premier mot de chaque bloc précédant le bloc cherché.

Implantation physique des fichiers (1/4)

Il existe plusieurs façon d'implanter (stocker) 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. Par contre l'accès aléatoire devient lent car il faut lire le premier mot de chaque bloc précédant le bloc cherché.

Implantation physique des fichiers (1/4)

Il existe plusieurs façon d'implanter (stocker) 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. Par contre l'accès aléatoire devient lent car il faut lire le premier mot de chaque bloc précédant le bloc cherché.

Implantation physique des fichiers (2/4)

Il existe également 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 et 8, et qu'un fichier B soit constitué de trois blocs situés sur 3, 9, 5. On obtient dans ce cas la table suivante.

Implantation physique des fichiers (2/4)

Il existe également 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 et 8, et qu'un fichier B soit constitué de trois blocs situés sur 3, 9, 5. On obtient dans ce cas la table suivante.

Implantation physique des fichiers (2/4)

Il existe également 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 et 8, et qu'un fichier B soit constitué de trois blocs situés sur 3, 9, 5. On obtient dans ce cas la table suivante.

Implantation physique des fichiers (2/4)

Il existe également 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 et 8, et qu'un fichier B soit constitué de trois blocs situés sur 3, 9, 5. On obtient dans ce cas la table suivante.

Implantation physique des fichiers (2/4)

Il existe également 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 et 8, et qu'un fichier B soit constitué de trois blocs situés sur 3, 9, 5. On obtient dans ce cas la table suivante.

Implantation physique des fichiers (2/4)

Il existe également 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 et 8, et qu'un fichier B soit constitué de trois blocs situés sur 3, 9, 5. On obtient dans ce cas la table suivante.

Implantation physique des fichiers (2/4)

Il existe également 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 et 8, et qu'un fichier B soit constitué de trois blocs situés sur 3, 9, 5. On obtient dans ce cas la table suivante.

Implantation physique des fichiers (3/4)

Blocs physiques	Chaînage
0	
1	
2	4
$B \Rightarrow 3$	9
4	8
5	-1
$A \Rightarrow 6$	2
7	
8	-1
9	5

(1)

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

Implantation physique des fichiers (3/4)

Blocs physiques	Chaînage
0	
1	
2	4
$B \Rightarrow 3$	9
4	8
5	-1
$A \Rightarrow 6$	2
7	
8	-1
9	5

(1)

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

Implantation physique des fichiers (3/4)

Blocs physiques	Chaînage
0	
1	
2	4
$B \Rightarrow 3$	9
4	8
5	-1
$A \Rightarrow 6$	2
7	
8	-1
9	5

(1)

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

Implantation physique des fichiers (3/4)

Blocs physiques	Chaînage
0	
1	
2	4
$B \Rightarrow 3$	9
4	8
5	-1
$A \Rightarrow 6$	2
7	
8	-1
9	5

(1)

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

Implantation physique des fichiers (4/4)

Il existe encore une autre méthode d'implantation qui consiste à attribuer à chaque fichier une structure, appelée i-node (pour index node ou noeud d'information en français). Les i-nodes sont créés lors de la création du système de fichiers. La quantité d'i-nodes (généralement déterminée lors du formatage et dépendant de la taille de la partition) indique le nombre maximum de fichiers que le système de fichiers peut gérer. À chaque fichier correspond un numéro d'i-nodes (i-number) dans le système de fichiers dans lequel il réside. Les i-nodes peuvent, selon les systèmes de fichiers, contenir aussi des informations concernant le fichier, tel que son créateur (ou propriétaire), son type d'accès (droits d'accès), etc. Le numéro d'i-node d'un fichier " toto " peut être affiché, sous Linux, en tapant la commande " ls -i toto ".

Implantation physique des fichiers (4/4)

Il existe encore une autre méthode d'implantation qui consiste à attribuer à chaque fichier une structure, appelée i-node (pour index node ou noeud d'information en français). Les i-nodes sont créés lors de la création du système de fichiers. La quantité d'i-nodes (généralement déterminée lors du formatage et dépendant de la taille de la partition) indique le nombre maximum de fichiers que le système de fichiers peut gérer. À chaque fichier correspond un numéro d'i-nodes (i-number) dans le système de fichiers dans lequel il réside. Les i-nodes peuvent, selon les systèmes de fichiers, contenir aussi des informations concernant le fichier, tel que son créateur (ou propriétaire), son type d'accès (droits d'accès), etc. Le numéro d'i-node d'un fichier " toto " peut être affiché, sous Linux, en tapant la commande " ls -i toto ".

Implantation physique des fichiers (4/4)

Il existe encore une autre méthode d'implantation qui consiste à attribuer à chaque fichier une structure, appelée i-node (pour index node ou noeud d'information en français). Les i-nodes sont créés lors de la création du système de fichiers. La quantité d'i-nodes (généralement déterminée lors du formatage et dépendant de la taille de la partition) indique le nombre maximum de fichiers que le système de fichiers peut gérer. À chaque fichier correspond un numéro d'i-nodes (i-number) dans le système de fichiers dans lequel il réside. Les i-nodes peuvent, selon les systèmes de fichiers, contenir aussi des informations concernant le fichier, tel que son créateur (ou propriétaire), son type d'accès (droits d'accès), etc. Le numéro d'i-node d'un fichier " toto " peut être affiché, sous Linux, en tapant la commande " ls -i toto ".

Implantation physique des fichiers (4/4)

Il existe encore une autre méthode d'implantation qui consiste à attribuer à chaque fichier une structure, appelée i-node (pour index node ou noeud d'information en français). Les i-nodes sont créés lors de la création du système de fichiers. La quantité d'i-nodes (généralement déterminée lors du formatage et dépendant de la taille de la partition) indique le nombre maximum de fichiers que le système de fichiers peut gérer. À chaque fichier correspond un numéro d'i-nodes (i-number) dans le système de fichiers dans lequel il réside. Les i-nodes peuvent, selon les systèmes de fichiers, contenir aussi des informations concernant le fichier, tel que son créateur (ou propriétaire), son type d'accès (droits d'accès), etc. Le numéro d'i-node d'un fichier " toto " peut être affiché, sous Linux, en tapant la commande " ls -i toto ".

Implantation physique des fichiers (4/4)

Il existe encore une autre méthode d'implantation qui consiste à attribuer à chaque fichier une structure, appelée i-node (pour index node ou noeud d'information en français). Les i-nodes sont créés lors de la création du système de fichiers. La quantité d'i-nodes (généralement déterminée lors du formatage et dépendant de la taille de la partition) indique le nombre maximum de fichiers que le système de fichiers peut gérer. À chaque fichier correspond un numéro d'i-nodes (i-number) dans le système de fichiers dans lequel il réside. Les i-nodes peuvent, selon les systèmes de fichiers, contenir aussi des informations concernant le fichier, tel que son créateur (ou propriétaire), son type d'accès (droits d'accès), etc. Le numéro d'i-node d'un fichier " toto " peut être affiché, sous Linux, en tapant la commande " ls -i toto ".

Implantation physique des fichiers (4/4)

Il existe encore une autre méthode d'implantation qui consiste à attribuer à chaque fichier une structure, appelée i-node (pour index node ou noeud d'information en français). Les i-nodes sont créés lors de la création du système de fichiers. La quantité d'i-nodes (généralement déterminée lors du formatage et dépendant de la taille de la partition) indique le nombre maximum de fichiers que le système de fichiers peut gérer. À chaque fichier correspond un numéro d'i-nodes (i-number) dans le système de fichiers dans lequel il réside. Les i-nodes peuvent, selon les systèmes de fichiers, contenir aussi des informations concernant le fichier, tel que son créateur (ou propriétaire), son type d'accès (droits d'accès), etc. Le numéro d'i-node d'un fichier " toto " peut être affiché, sous Linux, en tapant la commande " ls -i toto ".

Exemple d'i-node

Sous Unix, un i-node contient les informations suivantes :

- Taille du fichier (en octets) ;
- Identifiant du disque qui contient le fichier ;
- UID : User IDentifier du propriétaire du fichier ;
- GID : Group IDentifier du propriétaire du fichier ;
- Droits d'accès (en lecture, écriture, exécution) ;
- Nombres de liens (physiques) ;
- atime (access time) : Date de dernière ouverture du fichier ;
- mtime (modification time) : Date de dernière modification du fichier ;
- ctime (change Time) : Date de dernière modification de l'i-node lui-même ;
- Adresse : pointeur sur les blocs du disque sur lesquels le fichier est stocké.

Exemple d'i-node

Sous Unix, un i-node contient les informations suivantes :

- Taille du fichier (en octets) ;
- Identifiant du disque qui contient le fichier ;
- UID : User IDentifier du propriétaire du fichier ;
- GID : Group IDentifier du propriétaire du fichier ;
- Droits d'accès (en lecture, écriture, exécution) ;
- Nombres de liens (physiques) ;
- atime (access time) : Date de dernière ouverture du fichier ;
- mtime (modification time) : Date de dernière modification du fichier ;
- ctime (change Time) : Date de dernière modification de l'i-node lui-même ;
- Adresse : pointeur sur les blocs du disque sur lesquels le fichier est stocké.

Exemple d'i-node

Sous Unix, un i-node contient les informations suivantes :

- Taille du fichier (en octets) ;
- Identifiant du disque qui contient le fichier ;
- UID : User IDentifier du propriétaire du fichier ;
- GID : Group IDentifier du propriétaire du fichier ;
- Droits d'accès (en lecture, écriture, exécution) ;
- Nombres de liens (physiques) ;
- atime (access time) : Date de dernière ouverture du fichier ;
- mtime (modification time) : Date de dernière modification du fichier ;
- ctime (change Time) : Date de dernière modification de l'i-node lui-même ;
- Adresse : pointeur sur les blocs du disque sur lesquels le fichier est stocké.

Exemple d'i-node

Sous Unix, un i-node contient les informations suivantes :

- Taille du fichier (en octets) ;
- Identifiant du disque qui contient le fichier ;
- UID : User IDentifier du propriétaire du fichier ;
- GID : Group IDentifier du propriétaire du fichier ;
- Droits d'accès (en lecture, écriture, exécution) ;
- Nombres de liens (physiques) ;
- atime (access time) : Date de dernière ouverture du fichier ;
- mtime (modification time) : Date de dernière modification du fichier ;
- ctime (change Time) : Date de dernière modification de l'i-node lui-même ;
- Adresse : pointeur sur les blocs du disque sur lesquels le fichier est stocké.

Exemple d'i-node

Sous Unix, un i-node contient les informations suivantes :

- Taille du fichier (en octets) ;
- Identifiant du disque qui contient le fichier ;
- UID : User IDentifier du propriétaire du fichier ;
- GID : Group IDentifier du propriétaire du fichier ;
- Droits d'accès (en lecture, écriture, exécution) ;
- Nombres de liens (physiques) ;
- atime (access time) : Date de dernière ouverture du fichier ;
- mtime (modification time) : Date de dernière modification du fichier ;
- ctime (change Time) : Date de dernière modification de l'i-node lui-même ;
- Adresse : pointeur sur les blocs du disque sur lesquels le fichier est stocké.

Exemple d'i-node

Sous Unix, un i-node contient les informations suivantes :

- Taille du fichier (en octets) ;
- Identifiant du disque qui contient le fichier ;
- UID : User IDentifier du propriétaire du fichier ;
- GID : Group IDentifier du propriétaire du fichier ;
- Droits d'accès (en lecture, écriture, exécution) ;
- Nombres de liens (physiques) ;
- atime (access time) : Date de dernière ouverture du fichier ;
- mtime (modification time) : Date de dernière modification du fichier ;
- ctime (change Time) : Date de dernière modification de l'i-node lui-même ;
- Adresse : pointeur sur les blocs du disque sur lesquels le fichier est stocké.

Exemple d'i-node

Sous Unix, un i-node contient les informations suivantes :

- Taille du fichier (en octets) ;
- Identifiant du disque qui contient le fichier ;
- UID : User IDentifier du propriétaire du fichier ;
- GID : Group IDentifier du propriétaire du fichier ;
- Droits d'accès (en lecture, écriture, exécution) ;
- Nombres de liens (physiques) ;
- atime (access time) : Date de dernière ouverture du fichier ;
- mtime (modification time) : Date de dernière modification du fichier ;
- ctime (change Time) : Date de dernière modification de l'i-node lui-même ;
- Adresse : pointeur sur les blocs du disque sur lesquels le fichier est stocké.

Exemple d'i-node

Sous Unix, un i-node contient les informations suivantes :

- Taille du fichier (en octets) ;
- Identifiant du disque qui contient le fichier ;
- UID : User IDentifier du propriétaire du fichier ;
- GID : Group IDentifier du propriétaire du fichier ;
- Droits d'accès (en lecture, écriture, exécution) ;
- Nombres de liens (physiques) ;
- atime (access time) : Date de dernière ouverture du fichier ;
- mtime (modification time) : Date de dernière modification du fichier ;
- ctime (change Time) : Date de dernière modification de l'i-node lui-même ;
- Adresse : pointeur sur les blocs du disque sur lesquels le fichier est stocké.

Exemple d'i-node

Sous Unix, un i-node contient les informations suivantes :

- Taille du fichier (en octets) ;
- Identifiant du disque qui contient le fichier ;
- UID : User IDentifier du propriétaire du fichier ;
- GID : Group IDentifier du propriétaire du fichier ;
- Droits d'accès (en lecture, écriture, exécution) ;
- Nombres de liens (physiques) ;
- atime (access time) : Date de dernière ouverture du fichier ;
- mtime (modification time) : Date de dernière modification du fichier ;
- ctime (change Time) : Date de dernière modification de l'i-node lui-même ;
- Adresse : pointeur sur les blocs du disque sur lesquels le fichier est stocké.

Exemple d'i-node

Sous Unix, un i-node contient les informations suivantes :

- Taille du fichier (en octets) ;
- Identifiant du disque qui contient le fichier ;
- UID : User IDentifier du propriétaire du fichier ;
- GID : Group IDentifier du propriétaire du fichier ;
- Droits d'accès (en lecture, écriture, exécution) ;
- Nombres de liens (physiques) ;
- atime (access time) : Date de dernière ouverture du fichier ;
- mtime (modification time) : Date de dernière modification du fichier ;
- ctime (change Time) : Date de dernière modification de l'i-node lui-même ;
- Adresse : pointeur sur les blocs du disque sur lesquels le fichier est stocké.

Exemple d'i-node

Sous Unix, un i-node contient les informations suivantes :

- Taille du fichier (en octets) ;
- Identifiant du disque qui contient le fichier ;
- UID : User IDentifier du propriétaire du fichier ;
- GID : Group IDentifier du propriétaire du fichier ;
- Droits d'accès (en lecture, écriture, exécution) ;
- Nombres de liens (physiques) ;
- atime (access time) : Date de dernière ouverture du fichier ;
- mtime (modification time) : Date de dernière modification du fichier ;
- ctime (change Time) : Date de dernière modification de l'i-node lui-même ;
- Adresse : pointeur sur les blocs du disque sur lesquels le fichier est stocké.

Exemple d'i-node

Sous Unix, un i-node contient les informations suivantes :

- Taille du fichier (en octets) ;
- Identifiant du disque qui contient le fichier ;
- UID : User IDentifier du propriétaire du fichier ;
- GID : Group IDentifier du propriétaire du fichier ;
- Droits d'accès (en lecture, écriture, exécution) ;
- Nombres de liens (physiques) ;
- atime (access time) : Date de dernière ouverture du fichier ;
- mtime (modification time) : Date de dernière modification du fichier ;
- ctime (change Time) : Date de dernière modification de l'i-node lui-même ;
- Adresse : pointeur sur les blocs du disque sur lesquels le fichier est stocké.

Exemple d'i-node

Sous Unix, un i-node contient les informations suivantes :

- Taille du fichier (en octets) ;
- Identifiant du disque qui contient le fichier ;
- UID : User IDentifier du propriétaire du fichier ;
- GID : Group IDentifier du propriétaire du fichier ;
- Droits d'accès (en lecture, écriture, exécution) ;
- Nombres de liens (physiques) ;
- atime (access time) : Date de dernière ouverture du fichier ;
- mtime (modification time) : Date de dernière modification du fichier ;
- ctime (change Time) : Date de dernière modification de l'i-node lui-même ;
- Adresse : pointeur sur les blocs du disque sur lesquels le fichier est stocké.

Exemple d'i-node

Sous Unix, un i-node contient les informations suivantes :

- Taille du fichier (en octets) ;
- Identifiant du disque qui contient le fichier ;
- UID : User IDentifier du propriétaire du fichier ;
- GID : Group IDentifier du propriétaire du fichier ;
- Droits d'accès (en lecture, écriture, exécution) ;
- Nombres de liens (physiques) ;
- atime (access time) : Date de dernière ouverture du fichier ;
- mtime (modification time) : Date de dernière modification du fichier ;
- ctime (change Time) : Date de dernière modification de l'i-node lui-même ;
- Adresse : pointeur sur les blocs du disque sur lesquels le fichier est stocké.

Exemple d'i-node

Sous Unix, un i-node contient les informations suivantes :

- Taille du fichier (en octets) ;
- Identifiant du disque qui contient le fichier ;
- UID : User IDentifier du propriétaire du fichier ;
- GID : Group IDentifier du propriétaire du fichier ;
- Droits d'accès (en lecture, écriture, exécution) ;
- Nombres de liens (physiques) ;
- atime (access time) : Date de dernière ouverture du fichier ;
- mtime (modification time) : Date de dernière modification du fichier ;
- ctime (change Time) : Date de dernière modification de l'i-node lui-même ;
- Adresse : pointeur sur les blocs du disque sur lesquels le fichier est stocké.

Exemple d'i-node

Sous Unix, un i-node contient les informations suivantes :

- Taille du fichier (en octets) ;
- Identifiant du disque qui contient le fichier ;
- UID : User IDentifier du propriétaire du fichier ;
- GID : Group IDentifier du propriétaire du fichier ;
- Droits d'accès (en lecture, écriture, exécution) ;
- Nombres de liens (physiques) ;
- atime (access time) : Date de dernière ouverture du fichier ;
- mtime (modification time) : Date de dernière modification du fichier ;
- ctime (change Time) : Date de dernière modification de l'i-node lui-même ;
- Adresse : pointeur sur les blocs du disque sur lesquels le fichier est stocké.

Exemple d'i-node

Sous Unix, un i-node contient les informations suivantes :

- Taille du fichier (en octets) ;
- Identifiant du disque qui contient le fichier ;
- UID : User IDentifier du propriétaire du fichier ;
- GID : Group IDentifier du propriétaire du fichier ;
- Droits d'accès (en lecture, écriture, exécution) ;
- Nombres de liens (physiques) ;
- atime (access time) : Date de dernière ouverture du fichier ;
- mtime (modification time) : Date de dernière modification du fichier ;
- ctime (change Time) : Date de dernière modification de l'i-node lui-même ;
- Adresse : pointeur sur les blocs du disque sur lesquels le fichier est stocké.

Implantation physique 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.

Implantation physique 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.

Implantation physique 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.

Implantation physique 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.

Implantation physique 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.

Fichiers partagés

Il peut être nécessaire de pouvoir faire apparaître dans des répertoires différents un même fichier, 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.

Fichiers partagés

Il peut être nécessaire de pouvoir faire apparaître dans des répertoires différents un même fichier, 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.

Fichiers partagés

Il peut être nécessaire de pouvoir faire apparaître dans des répertoires différents un même fichier, 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.

Fichiers partagés

Il peut être nécessaire de pouvoir faire apparaître dans des répertoires différents un même fichier, 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.

Fichiers partagés : liens physiques

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. Dans ce cas, si le fichier d'origine est supprimé, le fichier est en réalité encore accessible par le lien : le fichier n'est pas détruit physiquement. Cela correspond à peu près à une *copie* d'un fichier. Sous linux on crée un lien symbolique par la commande "`ln fichier Nouveaufichier`".

Fichiers partagés : liens physiques

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. Dans ce cas, si le fichier d'origine est supprimé, le fichier est en réalité encore accessible par le lien : le fichier n'est pas détruit physiquement. Cela correspond à peu près à une *copie* d'un fichier. Sous linux on crée un lien symbolique par la commande "`ln fichier Nouveaufichier`".

Fichiers partagés : liens physiques

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. Dans ce cas, si le fichier d'origine est supprimé, le fichier est en réalité encore accessible par le lien : le fichier n'est pas détruit physiquement. Cela correspond à peu près à une *copie* d'un fichier. Sous linux on crée un lien symbolique par la commande "`ln fichier Nouveaufichier`".

Fichiers partagés : liens physiques

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. Dans ce cas, si le fichier d'origine est supprimé, le fichier est en réalité encore accessible par le lien : le fichier n'est pas détruit physiquement. Cela correspond à peu près à une *copie* d'un fichier. Sous linux on crée un lien symbolique par la commande "`ln fichier Nouveaufichier`".

Fichiers partagés : liens physiques

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. Dans ce cas, si le fichier d'origine est supprimé, le fichier est en réalité encore accessible par le lien : le fichier n'est pas détruit physiquement. Cela correspond à peu près à une *copie* d'un fichier. Sous linux on crée un lien symbolique par la commande "`ln fichier Nouveaufichier`".

Fichiers partagés : liens physiques

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. Dans ce cas, si le fichier d'origine est supprimé, le fichier est en réalité encore accessible par le lien : le fichier n'est pas détruit physiquement. Cela correspond à peu près à une *copie* d'un fichier. Sous linux on crée un lien symbolique par la commande "`ln fichier Nouveaufichier`".

Fichiers partagés : liens logiques

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. Un lien symbolique est donc traité comme un fichier à part entière (puisqu'il dispose de son propre i-node). Cela permet d'avoir un seul fichier physiquement présent sur le disque et de le désigner sous plusieurs noms différents. Si le fichier d'origine est détruit, le lien l'est également. Par exemple, sous Linux, on crée un lien symbolique " nomlien " vers le fichier " tp3 " par la commande " `ln -s tp3 nomlien` ". Il s'agit en quelque sorte d'un raccourci pour désigner un fichier.

Fichiers partagés : liens logiques

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. Un lien symbolique est donc traité comme un fichier à part entière (puisqu'il dispose de son propre i-node). Cela permet d'avoir un seul fichier physiquement présent sur le disque et de le désigner sous plusieurs noms différents. Si le fichier d'origine est détruit, le lien l'est également. Par exemple, sous Linux, on crée un lien symbolique " nomlien " vers le fichier " tp3 " par la commande " `ln -s tp3 nomlien` ". Il s'agit en quelque sorte d'un raccourci pour désigner un fichier.

Fichiers partagés : liens logiques

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. Un lien symbolique est donc traité comme un fichier à part entière (puisqu'il dispose de son propre i-node). Cela permet d'avoir un seul fichier physiquement présent sur le disque et de le désigner sous plusieurs noms différents. Si le fichier d'origine est détruit, le lien l'est également. Par exemple, sous Linux, on crée un lien symbolique " nomlien " vers le fichier " tp3 " par la commande " `ln -s tp3 nomlien` ". Il s'agit en quelque sorte d'un raccourci pour désigner un fichier.

Fichiers partagés : liens logiques

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. Un lien symbolique est donc traité comme un fichier à part entière (puisqu'il dispose de son propre i-node). Cela permet d'avoir un seul fichier physiquement présent sur le disque et de le désigner sous plusieurs noms différents. Si le fichier d'origine est détruit, le lien l'est également. Par exemple, sous Linux, on crée un lien symbolique " nomlien " vers le fichier " tp3 " par la commande " `ln -s tp3 nomlien` ". Il s'agit en quelque sorte d'un raccourci pour désigner un fichier.

Fichiers partagés : liens logiques

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. Un lien symbolique est donc traité comme un fichier à part entière (puisqu'il dispose de son propre i-node). Cela permet d'avoir un seul fichier physiquement présent sur le disque et de le désigner sous plusieurs noms différents. Si le fichier d'origine est détruit, le lien l'est également. Par exemple, sous Linux, on crée un lien symbolique " nomlien " vers le fichier " tp3 " par la commande " `ln -s tp3 nomlien` ". Il s'agit en quelque sorte d'un raccourci pour désigner un fichier.

Fichiers partagés : liens logiques

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. Un lien symbolique est donc traité comme un fichier à part entière (puisqu'il dispose de son propre i-node). Cela permet d'avoir un seul fichier physiquement présent sur le disque et de le désigner sous plusieurs noms différents. Si le fichier d'origine est détruit, le lien l'est également. Par exemple, sous Linux, on crée un lien symbolique " nomlien " vers le fichier " tp3 " par la commande " `ln -s tp3 nomlien` ". Il s'agit en quelque sorte d'un raccourci pour désigner un fichier.

Fichiers partagés : liens logiques

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. Un lien symbolique est donc traité comme un fichier à part entière (puisqu'il dispose de son propre i-node). Cela permet d'avoir un seul fichier physiquement présent sur le disque et de le désigner sous plusieurs noms différents. Si le fichier d'origine est détruit, le lien l'est également. Par exemple, sous Linux, on crée un lien symbolique " nomlien " vers le fichier " tp3 " par la commande " `ln -s tp3 nomlien` ". Il s'agit en quelque sorte d'un raccourci pour désigner un fichier.

Organisation générale du disque

Les fichiers sont enregistrés sur des disques qui sont le plus souvent découpés en partitions, les systèmes de fichiers étant indépendants sur chaque partition. Le secteur 0 du disque est réservé au démarrage de la machine, il est appelé MBR (Master Boot Record) et 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 lors du démarrage de la machine. 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, etc.). 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.

Organisation générale du disque

Les fichiers sont enregistrés sur des disques qui sont le plus souvent découpés en partitions, les systèmes de fichiers étant indépendants sur chaque partition. Le secteur 0 du disque est réservé au démarrage de la machine, il est appelé MBR (Master Boot Record) et 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 lors du démarrage de la machine. 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, etc.). 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.

Organisation générale du disque

Les fichiers sont enregistrés sur des disques qui sont le plus souvent découpés en partitions, les systèmes de fichiers étant indépendants sur chaque partition. Le secteur 0 du disque est réservé au démarrage de la machine, il est appelé MBR (Master Boot Record) et 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 lors du démarrage de la machine. 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, etc.). 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.

Organisation générale du disque

Les fichiers sont enregistrés sur des disques qui sont le plus souvent découpés en partitions, les systèmes de fichiers étant indépendants sur chaque partition. Le secteur 0 du disque est réservé au démarrage de la machine, il est appelé MBR (Master Boot Record) et 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 lors du démarrage de la machine. 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, etc.). 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.

Organisation générale du disque

Les fichiers sont enregistrés sur des disques qui sont le plus souvent découpés en partitions, les systèmes de fichiers étant indépendants sur chaque partition. Le secteur 0 du disque est réservé au démarrage de la machine, il est appelé MBR (Master Boot Record) et 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 lors du démarrage de la machine. 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, etc.). 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.

Organisation générale du disque

Les fichiers sont enregistrés sur des disques qui sont le plus souvent découpés en partitions, les systèmes de fichiers étant indépendants sur chaque partition. Le secteur 0 du disque est réservé au démarrage de la machine, il est appelé MBR (Master Boot Record) et 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 lors du démarrage de la machine. 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, etc.). 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.

Organisation générale du disque

Les fichiers sont enregistrés sur des disques qui sont le plus souvent découpés en partitions, les systèmes de fichiers étant indépendants sur chaque partition. Le secteur 0 du disque est réservé au démarrage de la machine, il est appelé MBR (Master Boot Record) et 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 lors du démarrage de la machine. 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, etc.). 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.

Organisation générale du disque

Les fichiers sont enregistrés sur des disques qui sont le plus souvent découpés en partitions, les systèmes de fichiers étant indépendants sur chaque partition. Le secteur 0 du disque est réservé au démarrage de la machine, il est appelé MBR (Master Boot Record) et 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 lors du démarrage de la machine. 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, etc.). 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.

Organisation générale du disque

Les fichiers sont enregistrés sur des disques qui sont le plus souvent découpés en partitions, les systèmes de fichiers étant indépendants sur chaque partition. Le secteur 0 du disque est réservé au démarrage de la machine, il est appelé MBR (Master Boot Record) et 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 lors du démarrage de la machine. 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, etc.). 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.

Organisation générale du disque

Les fichiers sont enregistrés sur des disques qui sont le plus souvent découpés en partitions, les systèmes de fichiers étant indépendants sur chaque partition. Le secteur 0 du disque est réservé au démarrage de la machine, il est appelé MBR (Master Boot Record) et 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 lors du démarrage de la machine. 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, etc.). 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.

Organisation générale du disque

Les fichiers sont enregistrés sur des disques qui sont le plus souvent découpés en partitions, les systèmes de fichiers étant indépendants sur chaque partition. Le secteur 0 du disque est réservé au démarrage de la machine, il est appelé MBR (Master Boot Record) et 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 lors du démarrage de la machine. 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, etc.). 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.

Organisation générale du disque

Les fichiers sont enregistrés sur des disques qui sont le plus souvent découpés en partitions, les systèmes de fichiers étant indépendants sur chaque partition. Le secteur 0 du disque est réservé au démarrage de la machine, il est appelé MBR (Master Boot Record) et 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 lors du démarrage de la machine. 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, etc.). 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.