Projet VirtualFS

Utilisation du projet :

./virtualFS C disque.bin : crée un disque virtuel
./virtualFS S disque.bin nom1 nom2 : stocke le fichier nom1 dans le disque virtuel, sous le nom nom2.
./virtualFS X disque.bin nom1 nom2 : extrait le fichier nom1 du disque virtuel stocké dans disque.bin, dans le fichier nom2.
./virtualFS R disque.bin nom1 : supprime le fichier nom1 du disque.
./virtualFS L disque.bin : liste toutes les tables d'adresses.
./virtualFS F disque.bin : liste tous les fichiers du disque.
./virtualFS D disque.bin : défragmente les fichiers du disque.

Fonctionnalité mise en place :

-Ajout d'un fichier
-Création d'un système de fichier
-Suppression d'un fichier
-Table d'espace libre
-Récupération d'un fichier
-Défragmentation
-Troncage du fichier si il y a un espace libre à la fin (inclut dans la défragmentation).
-Affichage de tous les fichiers
-Affichage des tables d'index

Méthode de conception du projet :

Toute les tables d'adresse ont été implémentées par liste chaînées. Quand on arrive à la fin d'une table, on en crée une nouvelle et la première pointera vers la seconde.
La première chose qui est au début du disque virtuel est la table d'adresse. La seconde chose qui la suit juste après est la table d'espace libre, il faut au minimum l'initialisation de ces deux choses-là pour la création d'un disque virtuel.

Lors de l'ouverture du disque virtuel, ces deux tables sont chargées en mémoire (et mises à jour en fonction des actions).

Tous les objets, sauf les deux premiers peuvent être n'importe où dans le disque virtuel.

La défragmentation déplace toutes les données les plus loin dans l'espace libre le plus proche puis recherche l'objet le plus loin pour troquer le fichier pour réduire sa taille.

Fichier et signification :

Adressetable : Fichier où se trouve la structure de table d'adresse. La table d'adresse est associée à un inode pour stocker les offset où se trouvent les données.

FileSystem : Fichier où se trouve la représentation du système de fichier. On peut ajouter, supprimer des fichiers, défragmenter, ouvrir, créer et fermer des système de fichiers (disque virtuel).

FreeSpaceTable : contient la représentation de la table d'espace libre. C'est dans ces fichiers que l'on a des fonctions qui nous retournent où mettre les données.

IndexTable : Contient la représentation de la table d'index. Cette table contient l'adresse de tous les inode des fichiers qui se trouvent sur le système de fichiers. C'est dans ce fichier que l'on a des fonctions qui nous retournent si un fichier existe ou non, où se trouve l'objet le plus lointain dans le fichier ainsi que les fonctions de gestion de cette index.

Inode : Contient les informations d'un fichier à l'intérieur du système de fichiers. L'inode se présente comme suis : sont nom est dynamique, il a une table data qui lui permet de stocker des offset des données dans le fichier. Si ces données ne tiennent plus dans sa table, une liste chaînée de tables d'adresse sera crée. Les fichiers inode contiennent toutes les fonctions pour gérer les Inode.

Main : Invoque les fonctionnalités des fonctions du système de fichiers en fonction des choix de l'utilisateur (appel des fonctions du fichier FileSystem).

Limitation de l'implémentation :

- Lorsque un fichier est supprimé, les blocs, un par un sont ajoutés à la table d'espace libre avec leur taille.
  Lorsque l'on supprime un inode (taille variable) et qu'il reste de l'espace libre, cet espace est ajouté à la table d'espace libre.

A force de supprimer et recopier des données, les cases ajoutées seront plus petites et à terme, pour un bloc donné, laisseront des trous.

Par exemple :
Un bloc de 2048 octets sera scindé à force de déplacement d'inode dans la table d'espace libre en x bloc de taille variable. Par exemple, un bloc de 150 octets, un autre de 125, un autre de 100 qui se suivent et qui sont vides.

Résolution : écrire une procédure qui recherchera les blocs d'espace libre qui se suivent et les fusionnera.

- Implémentation par liste chaînée entre blocs donc lente
  Résolution :
  Une implémentation en arbre pourrait augmenter drastiquement les performances.

- Ne gère pas les dossiers
  Résolution : Ajouter un token dossier à l'inode et modifier les algo de recherche de fichiers