Ext3Viewer

Programme à but éducatif permettant de naviguer à travers le système de fichiers ext3 et de comprendre ses structures et fonctionnalités.

Cahier des charges

1 Description des fonctionnalités attendues par l'utilisateur

1.1 Produit attendu et intérêt

Le rôle du logiciel est de permettre à l'utilisateur de naviguer à travers le système de fichiers ext3. Nous l'avons choisi car c'est l'un des systèmes de fichier les plus utilisés sous Linux, et car c'est le système de fichiers étudié en cours. Notre programme permettra aux étudiants de mieux le comprendre. Il devra se présenter comme un explorateur de fichier mais permettra d'analyser les structures de données des partitions formatées en ext3. De plus, il devra permettre à l'utilisateur d'étudier le journal du système de fichiers. Ext3Viewer devra être utilisable de deux manières : un mode texte dans lequel l'utilisateur entrera les options dans la console, et un mode graphique qui permettra de naviguer et d'afficher les structures du système de fichiers, essentiellement à l'aide de la souris. Dans le mode graphique, l'utilisateur pourra également obtenir, pour chaque structure, des « astuces » : un récapitulatif lui permettant de mieux appréhender le système de fichiers.

Le but principal de Ext3Viewer sera donc de faciliter la compréhension du système de fichiers ext3 pour les personnes étudiant l'informatique.

1.2 Les différentes fonctionnalités

1.2.1 Visualisation des structures ext3

Le programme permettra de visualiser les différentes structures du système de fichiers ext3 ainsi que les valeurs prises par chacun des champs. Les valeurs seront lues à partir d'une partition et de la sélection de l'utilisateur. Chaque champ sera commenté pour permettre à l'utilisateur d'en comprendre l'utilité.
Cette fonctionnalité de visualisation sera la clé du programme : c'est grâce à elle que l'utilisateur comprendra l'organisation du système de fichiers et le stockage des fichiers que l'utilisateur à l'habitude de manipuler.

1.2.2 Navigation à travers le système de fichiers

L'interface graphique permettra de naviguer à travers le système de fichiers, tandis que le mode console permettra de visualiser les structures (sans explications). Ce dernier sera donc surtout une sorte de 'débugger' réservé aux utilisateurs initiés au fonctionnement de ext3. Grâce à l'interface graphique, l'utilisateur pourra voir ses fichiers et dossiers habituels et tout en naviguant il pourra afficher la structure de chaque *inode*. Il aura accès aux informations concernant le *boot sector*, les *group blocks*, le *superblock*, les différentes cartes des *inodes* et des *blocks*. Il pourra se situer, lors de la navigation, dans les structures de la partition. De plus, il lui sera possible de consulter le *journal* du système de fichiers. Lorsque l'utilisateur ne comprendra pas les informations que le programme lui présente, il pourra à tout moment afficher une explication sur la structure ou le champ de la structure qu'il consulte.
Néanmoins, certaines actions comme la suppression d'un fichier, la création d'un lien, etc... ne peuvent être comprises par une simple description des structures impliquées. Le programme devra donc présenter différents cas d'utilisation du système de fichiers et ses conséquences dans les structures concernées. Ces cas d'usage du système de fichiers seront rassemblés dans une partie aide, accessible depuis le menu.

1.2.3 Correspondance nom --> inode et inversement

Le logiciel comportera une fonctionnalité qui permettra de passer d'un nom de fichier (ou d'un chemin) au numéro d'inode correspondant. Cette fonction servira également pour la recherche de fichier dans une partition. Inversement, le programme proposera de retrouver un fichier, un dossier, ou autre, à partir de son numéro d'inode. On pourra aussi afficher à quel *inode* est rattaché un *block* en tapant son numéro de *block*.

1.2.4 Recherche de structures libres

Le programme permettra de trouver l'*inode* libre suivant sur le disque. Il permettra aussi de trouver le *block* libre suivant. De plus, on pourra tester si un *inode* ou un *block* est utilisé, et, dans ce cas, se rendre au bon endroit dans l'arborescence et afficher la structure du fichier ou dossier.

1.3 Plate-formes d'utilisation

Le système de fichiers ext3 étant utilisé essentiellement sous Linux, notre programme sera donc compatible Linux et développé pour une utilisation sous Linux. Cependant l'interface graphique sera programmée avec une API également compatible avec Windows.
Windows, par défaut, ne sait pas lire le système de fichiers ext3, ce qui ne rendra donc possible l'utilisation du programme sous Windows qu'avec une partition exemple sauvegardée dans un fichier. L'utilisation sous Linux est plus intéressante car l'utilisateur peut monter sa partition ext3, la modifier, puis étudier directement les conséquences de ses modifications sur le système de fichiers.
(Il est possible que le programme soit compatible avec MacOs.)

2 Description des fonctionnalités du point de vue développeur

2.1 Études des avancés dans ce domaine

debugfs - ext2/ext3 file system debugger : *The debugfs program is an interactive file system debugger. It can be used to examine and change the state of an ext2 file system.*
Ce programme permet d'afficher des structures du système de fichiers, mais ne nous donne pas accès à toutes les structures de ext2 ni les nouvelles structures propres à ext3. De plus ce programme est fait pour 'débugger', il ne comporte aucune explication, cependant il permet d'écrire sur une partition au format ext2/ext3.

dumpe2fs - dump ext2/ext3 filesystem information : *dumpe2fs prints the super block and blocks group information for the filesystem present on device.*
dumpe2fs permet d'afficher les valeurs des champs des structures du *superblock* et du groupe de *blocks* (pas d'accès a la structure *inode*). Mais encore une fois cet utilitaire n'affiche pas toutes les structures du système de fichiers, et de plus il n'a pas pour but d'expliquer ces structures. Comme le programme précédent, il ne permet ni d'afficher les données relatives au journal ni celles des listes d'accès (ACL : *access control list*).

e2view : programme qui permet de visualiser les structures du système de fichiers ext2. Notre programme est en quelque sorte une évolution de ce programme pour expliquer les nouvelles structures du système de fichiers, qui en est à sa version 3 (journalisation, access control lists, etc ...). *E2view* est programmé entièrement en C et utilise TCL pour l'interface graphique. Cette dernière n'étant pas fonctionnelle,d'après son auteur , nous avons décidé de ne pas utiliser ce langage.

2.2 Description modulaire

2.2.1 Lecture des données sur le disque dur

Avant de pouvoir analyser une partition, il faut ouvrir le système de fichiers et accéder aux structures du système de fichiers ext3. Il y aura donc un module chargé des accès en lectures sur le disque. Ce module se présentera sous la forme d'un catalogue de fonctions comme *lire_superbloc()* ou *lire_inode()*.

2.2.2 Visualisation des structures par la console

Le mode console sera une interface entre l'utilisateur et les fonctions de lecture des structures. L'utilisateur tapera une commande, avec des options, qui seront analysée par une fonction. Le programme affichera ensuite les informations souhaitées en faisant appel aux fonctions de lecture, et aux fonctions d'affichage.

2.2.3 Navigation grâce à l'interface graphique

L'interface graphique aura une fonctionnalité de navigation dans le système de fichiers. La fenêtre graphique sera divisée en 5 grandes parties :

- le menu (fichier, aide, about ...)
- une première barre indiquant la position dans les structures *bootsector* + *superblock* + *groupes*
- une seconde barre contenant *block bitmap* + *inode table* + *inode bitmap*
- menu de navigation sous la forme d'un arbre déroulant
- affichage des valeurs prises par chaque élément dans la structure sélectionnée.

Après avoir ouvert le système de fichiers grâce au menu situé en haut de la fenêtre graphique, l'utilisateur aura accès à un arbre déroulant de ses fichiers et dossiers. Cet arbre déroulant représentera l'arborescence habituelle, lors de l'utilisation de son système de fichiers sous Linux. Le premier dossier sera la racine « / ». En cliquant dessus, le programme cherchera dans les structures du système de fichiers les fichiers contenus dans ce répertoire

(d*irectory_entries*), et les affichera. En plus de cet affichage, on pourra lire pour chaque fichier son numéro d'*inode.* Lorsque l'on cliquera sur un fichier, la structure de l'*inode* du fichier concerné sera affichée dans la partie centrale de la fenêtre graphique.

2.2.4 Affichage des structures dans la fenêtre graphique

Pour chaque structure sélectionnée, soit dans la barre de navigation à gauche, soit dans une des barres en haut présentant l'organisation des structures sur le disque, on pourra afficher les valeurs prises par chaque champ de la structure, dans le cadre central de l'application. Pour chaque champ, l'utilisateur aura accès à une aide qui lui expliquera à quoi sert le champ sélectionné.

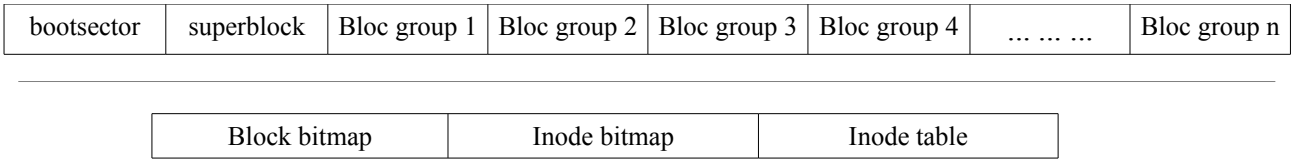
Les différentes structures affichables seront :

- les structures ACL : *ext3_acl_header*, *ext3_acl_entry, ext3_group_desc*
- la structure d'un *inode* sur le disque : *ext3_inode*
- la structure du *superblock* : *ext3_super_block*
- la structure d'une entrée dans un répertoire : *ext3_dir_entry, ext3_dir_entry2* (nouvelle version utilisée)
- le journal du système de fichiers

Ces structures sont stockées dans un certain ordre sur le disque dur, on peut modéliser cette répartition selon un schéma des structures. Ce schéma sera affiché en haut de la fenêtre et permettra à l'utilisateur de se situer dans l'organisation du disque dur.

2.2.5 Affichage de l'emplacement dans le système de fichiers

Le schéma représentant l'organisation des structures sur le disque et permettant de se situer dans les structures aura un allure semblable au schéma ci dessous :



Ces barres qui serviront en même temps à la navigation seront situés juste en dessous du menu du logiciel. En cliquant sur l'une de ces structures, l'utilisateur pourra lire les champs contenus par ces structures dans la partie principale de la fenêtre. Par exemple, si l'utilisateur clique sur le groupe de bloc 1, puis sur *inode bitmap*, la carte des inodes sera affichée et montrera quels inodes sont libres et occupés.

2.3 Analyse des risques

Le risque principal est la sous-estimation des délais nécessaires à la maîtrise des technologies employées : il sera nécessaire de connaître parfaitement le système de fichiers ext3, afin de produire un logiciel clair et pratique pour des personnes souhaitant découvrir comment ce système fonctionne. De même, l'utilisation d'une API telle que GTK sera une découverte, ce qui peut demander un temps d'adaptation, préjudiciable à la bonne marche du projet.
Dans la même optique, la fonction open() utilisée habituellement pour ouvrir un descripteur de fichier ne fonctionne qu'avec des fichiers jusqu'à 2 Go. Il faudra donc trouver une solution permettant d'ouvrir un système de fichiers actuel, pouvant aller jusqu'à des centaines de giga-octets.

2.4 Plate-forme et outils de développement

Le logiciel sera programmé en C pour le module de lecture et le module d'affichage en console. Il sera aussi programmé en C (ou C++) pour le module graphique, en utilisant l'API GTK. L'une des raisons pour laquelle nous avons choisi la bibliothèque graphique GTK est sa capacité à s'interfacer avec un langage compilé, contrairement à TCL, qui est interprété. De plus GTK est compatible avec Linux, Windows et MacOs. Nous aurions pu utiliser Qt qui est également multiplateforme, mais sa politique un peu floue sur sa licence d'utilisation nous a finalement décidé pour GTK (licence LGPL, utilisable pour un projet libre ou non).

Toutes les fonctions d'accès aux structures ext3 font partie de la bibliothèque *libc*, nous utiliserons donc gcc et g++ pour compiler notre programme.

3 Prévision du planning

mi-décembre : l'accès aux structures, la lecture des informations sur le disque ainsi que les différentes fonctions comme la navigation et la recherche dans l'arborescence devront être terminés. Le mode texte du logiciel devra lui aussi être fini. Durée: 1+1/2 mois
fin février : L'interface graphique, avec ses fonctions de navigation et d'affichage des structures, devra être terminée. Les fonctions d'aide à l'utilisateur dans l'interface graphique devront être bien entamées. Durée : 2+1/2 mois
mi-mars : Les fonctions d'aide à l'utilisateur en mode graphique devront être terminées. La rédaction d'une documentation sur le fonctionnement d'ext3 devra être écrite, ainsi que la documentation utilisateur et la documentation développeur. Une création de fichiers d'internalisations pourrait être faite pour que le programme fonctionne en 2 langues (français et anglais). Ceci rendrait facile le rajout d'une langue, par la modification de quelques fichiers prévus à cet effet. Durée : 1 mois

4 Répartition du travail

La répartition du travail sera faite de telle sorte que chaque développeur programme une partie des modules de l'interface, et une partie des fonctions nécessaires à la lecture disque. De cette manière, les techniques employées seront maîtrisées par tous les membres de l'équipe.

5 Apprentissage technique nécessaire à la réalisation du projet

Pour la réalisation de notre projet, nous aurons l'occasion d'aborder de nouvelles connaissances techniques. Voici une liste de ce qui nous sera nécessaire pour notre projet :

- utilisation de l'API graphique GTK
- apprentissage des structures de ext3
- apprentissage des « standards » pour la création d'un script de configuration pour préparer la compilation suivant la plate-forme *./configure* et suivant des options, que la personne qui compile désire activer ou non.
- utilisation des codes d'erreur standard définis dans *errno.h* sont utilisés dans un projet.
- utilisation de *groff,* pour la création de *pages manuel*
- organisation de la programmation pour permettre l'internationalisation du logiciel (support de langues étrangères)