RAPPORT
RS SYSTEME

Extracteur d'archive tar durable et parallèle

Version 1.0

Github repository :

Maxime ESCAMEZ
Quentin TARDIVON

Table des matières

1 - Introduction

Le but de ce projet à été de mettre en place un programme permettant de désarchiver des fichiers **tar**. Il a fallu, pour créer et manipuler des fichiers et répertoires, utiliser les fonctions de l'API POSIX et non de la bibliothèque standard C. De plus, le programme doit être durable et l'écriture des données doit être parallélisée grâce à l'utilisation de threads.

2 - Choix de conception et difficultés rencontrées

2.1 Listeur simple

Le développement du listeur simple devait se faire avec précaution puisque c'est dans cette partie qu'a été implémenté le parcours de l'archive. De cette partie dépend donc non seulement le listing, mais aussi l'écriture correcte des fichiers. On parcourt l'archive par bloc de 512 octets. Dans chaque premier bloc est indiqué la taille de l'élément inspecté, on peut donc utiliser cette taille pour connaître le nombre de bloc utilisé par cet élément et trouver le header du prochain élément. Pour cela on utilise la fonction lseek avec le nombre de bloc à sauter, on trouve alors le prochain header. Pour savoir si on est toujours dans l'archive, on test la valeur magic, si elle est égale à ustar on est toujours dans l'archive et on imprime le nom de l'élément, si elle est nulle, on est hors de l'archive et on arrête le programme.

2.2 Traitements des options de liste de commande

Cette partie est complètement indépendante du reste du programme. Il a juste fallu implémenter un gestionnaire d'options de commande en utilisant la fonction "getopt". Nous n'avons pas rencontré de difficultés particulières pour cette partie, il faut juste penser aux différentes façon d'écrire les options de l'utilisateur (-x -l est valide, -xl est valide, -xlzp 5 est valide, -xlzp n'est pas valide car p a besoin d'un entier, -xlpz 5 n'est pas valide). On notera que n'importe quelle autre lettre que x,l,z ou p ne sera pas validé. Selon les options choisies, on activera un flag pour la suite du programme.

2.3 Listeur détaillé

Tout d'abord, pour les permissions nous avons différencié les fichiers, répertoires et symlink pour rajouter l'élément précédant les autres permissions à l'aide de l'attribut *typeflag* de la structure ustar . Les permissions sont déterminées en traitant l'attribut *mode* avec une fonction "modeReading" qui rendra les permissions correspondantes. Pour l'uid, le

gid, le nom et la taille de fichier nous avons utilisé les attributs *uid*, *gid name* et *size* de ustar. Ensuite pour la date de modification des données, nous l'avons récupéré grâce à l'attribut *mtime* et formaté avec la fonction C "strftime". Enfin dans le cas d'un symlink,on rajoutera sa destination avec l'attribut *linkname*.

2.4 Extracteur

Pour effectuer l'extraction on récupère d'abord les permissions du fichier original avec l'attribut *mode*. Selon la nature du fichier à créer on utilisera la fonction adéquate. Pour un fichier nous implémentons notre fonction "createFile" qui prend en argument le nom du fichier, sa taille, le file descriptor et les permissions récupérées auparavant. Dans cette fonction on crée un buffer contenant la totalité des données du fichier ce qui sera avantageux pour le multi-threading mais problèmatique dans le cas d'un fichier très volumineux. Pour un dossier, on utilise l'appel système "mkdir" qui prend lui aussi les permissions en paramètre et le nom du répertoire original. Pour un symlink on utilise l'appel système "symlink"qui prend en paramètre le nom du symlink et le fichier destination. De plus on applique l'appel système "chown" pour appliquer l'uid et le gid des fichiers sources à nos fichiers extraits.
En ce qui concerne l'affichage de la date, nous avons dû reparcourir l'archive pour la mettre à jour. En effet les opérations nécessaires pour obtenir l'extraction sont effectué dans la boucle while principale qui parcours le buffer. Lors de cette extraction, les informations modifiées écrase la date modifiée lors du premier parcours de l'archive. Il est donc nécessaire de la reparcourir.

2.5 Ajout de la durabilité et parallélisation

Nous avons commencé la parallélisation, mais n'étant pas fonctionnelle, elle n'est pas intégrée à la branche master. Elle est disponible sur la branche Thread du repository Github. Le problème rencontré lors de la parallélisation concerne la sauvegarde des différents threads pour les lancer au bon moment. La stratégie établie était la suivante : le thread principale lit l'archive et passe un buffer d'information lu à un thread d'écriture disponible qui va se charger d'écrire les informations sur le disque.

3 Temps de travail

Maxime ESCAMEZ

Etape	Conception	Codage	Tests	Rédaction du rapport
1	1h	0h	0h	4h
2	1h	4h	1h	
3	1h	3h	1h30min	
4	1h	16h	2h30min	
5	1h	0h	0h	

Temps total : 37h

Quentin TARDIVON

Etape	Conception	Codage	Tests	Rédaction du rapport
1	1h	4h	1h	0h30min
2	0h	0h	0h30min	
3	2h	5h	1h30min	
4	1h	16h	2h30min	
5	1h	1h	0h30min	

Temps total : 37h30

4 - Conclusion

Ce projet nous a permis de comprendre le fonctionnement de l'archiveur tar et de programmer à un niveau proche du système. Nous avons aussi pu remarquer les différences d'UNIX entre Debian (notre OS de test) et Mac (notre OS de travail). En effet il a fallu être particulièrement prudent car MacOS n'utilise pas forcément les standards POSIX et les implémentations de tar sont différentes.

5 - Remerciements

Nous avons réalisé ce projet en nous aidant des sites webs suivants :
ftp ://gd.tuwien.ac.at/languages/c/programming-bbrown/c_075.htm
http ://manpagesfr.free.fr/man/man2/open.2.html
http ://man7.org/linux/man-pages/man3/opterr.3.html
http ://man7.org/linux/man-pages/man2/mkdir.2.html
http ://man7.org/linux/man-pages/man2/utime.2.html
https ://doc.ubuntu-fr.org/permissions
http ://man7.org/linux/man-pages/man2/symlink.2.html
http ://permissions-calculator.org/
https ://www.gnu.org/software/libc/manual/html_node/File-Times.html
https ://www.freebsd.org/cgi/man.cgi ?query=lutimessektion=2

Nous avons récuperé la fonction OctalToDecimal sur ce site :
https ://www.programiz.com/c-programming/examples/octal-decimal-convert