Projet d'Introduction au Système
Licence $2^{ème}$ année

Gilles Roussel

6 mai 2012

À rendre à votre chargé de TD pour le 17 mai 2012.
Le code source du projet doit être accompagné du petit rapport au format `pdf` de 5 pages maximum dans lequel vous décrirez vos choix.

L'objectif de ce projet est d'écrire une bibliothèque pour la manipulation de fichier `tar` (Wikipedia) et un petit interpréteur de commande qui l'utilise.

0 Le fichier `tar`

Un fichier `tar` est un fichier qui regroupe à l'intérieur d'un même fichier une arborescence de fichiers de façon structurée. Elle permet de la récupérer avec les noms de fichiers et divers autres attributs.

Le format du fichier est décrit sous l'URL suivant : `http://en.wikipedia.org/wiki/Tar_(file_format)#Format_details`. On utilisera dans le projet l'extension *USTAR* et on se contentera de traiter les fichiers normaux et les répertoires avec des noms courts. Un exemple d'archive que vous devrez être capable de traiter est accessible sous ce lien.

Pour créer un fichier `tar` de nom `root.tar` à partir d'un arborscence de fichier dont la racine est `root`, il suffit d'utiliser la commande suivante : `tar cvf root.tar root`

1 La bibliothèque

La bibliothèque que vous devez écrire devra avoir pour nom `libtar.so` et elle devra contenir au moins les fonctions suivantes dont les profils seront décrits dans un fichier `tar.h` :

- une fonction `TAR * open_tar(char *name)` qui retourne un pointeur vers une structure opaque `TAR` que vous aurez définie et qui représente le fichier `tar` ouvert ;
- une fonction `int close_tar(TAR* tarfile)` qui ferme le fichier `tar` ;
- une fonction `int tar_chdir(TAR* tarfile, const char *path)` comparable à `chdir` qui permet de changer le répertoire courant. Après ouverture, le répertoire courant est la racine de l'archive ;
- une fonction `char *tar_getcwd(TAR* tarfile, char *buf, size_t size)` comparable à `getcwd` ;
- des fonctions `TARDIR *tar_opendir(TAR* tarfile, const char *name)`, `struct tar_dirent *tar_readdir(TARDIR *dir)` et `int tar_closedir(TARDIR *dir)` comparables `opendir`, `readdir` et `closedir`. La structure `tar_dirent` devra permettre d'obtenir les informations récupérées dans le fichier `tar` et qui concerne le fichier trouvé ;
- des fontions `TARFILE* tar_fopen_r(TAR* tarfile, char *name)` comparable à `fopen` en lecture ;
- des fonctions `size_t tar_fread(void *ptr, size_t size, size_t nmemb, TARFILE *stream)`, `int tar_fseek(TARFILE *stream, long offset, int whence)` et `int tar_fclose(TARFILE *fp)` comparables à `fread`, `fseek` et `fclose`.

Les noms de fichiers utilisés seront relatifs au répertoire courant dans l'archive ou à la racine de l'archive s'ils commencent par `/`.

2 Interpréteur de commande

Écrire un petit interpréteur de commande qui utilise les fonctions de la bibliothèque que vous avez écrite. Cet interpréteur contiendra au moins les commandes suivantes :

- `opentar <fichier>` qui ouvre un fichier `tar`. Un seul fichier pourra être ouvert à la fois ;
- `closetar` qui ferme le fichier `tar` courant ;
- `cd <rep>` qui permet de se déplacer dans l'arborescence du fichier `tar`. Après ouverture avec `opentar`, le répertoire courant est la racine de l'archive ;
- `pwd` qui affiche le répertoire courant ;
- `ls` qui liste le contenu du répertoire courant. Avec l'option `-l`, elle retourne en plus les attributs des fichiers trouvés dans l'archive ;
- `cat <fichier>` qui affiche le contenu du fichier ;
- `part <debut> <fin> <fichier>` qui affiche le contenu du fichier `<fichier>` entre les octets `<debut>` et `fin`

3 Étapes possibles

1. Écrire une commande `readtar` qui affiche le nom, le type (répertoire ou fichier régulier) et la taille des fichiers contenus dans un fichier `tar` dont le nom est passé en argument.
2. Écrire un interpréteur de commandes à l'aide la bibliothèque `readline` et de la fonction `strtok` qui ne fait que vérifier que la commande entrée fait bien partie des commandes acceptées (`opentar`, `pwd`, etc.) et qu'elle a bien le bon nombre d'arguments, puis qui l'affiche si elle est valide.
3. Modifier la commande `readtar` pour qu'elle affiche, en plus, le contenu des fichiers réguliers.
4. Modifier la commande `readtar` pour qu'elle affiche, en plus, le nom des fichiers contenus dans les répertoires.
5. Écrire la bibliothèque partagée `libtar.so`.
6. Intégrer la bibliothèque `libtar.so` dans l'interpréteur de commandes.

4 Évaluation

Le respect des noms de fichiers et de fonctions précisés dans ce sujet sont un aspect important de l'évaluation finale du projet car ceux-ci seront testés de façon automatique.

Une attention particulière devra être portée à la détection et à la gestion des erreurs qui pourront se produire lors de l'utilisation de votre bibliothèque.

Votre projet sera à rendre sous la forme d'une archive compressée `tar.gz` contenant : les sources et un `Makefile` pour les compiler dans un répertoire `src`, les bibliothèques dans un répertoire `lib`, les commandes dans un répertoire `bin` et une documentation simple sur votre projet au format `pdf` dans un répertoire `doc`.