Architecture TSH

Introduction :

Ce document sert à expliquer la démarche prise pour l'implémentation du shell pour les TarBall, TSH.

I. Déroulement du développement de notre projet

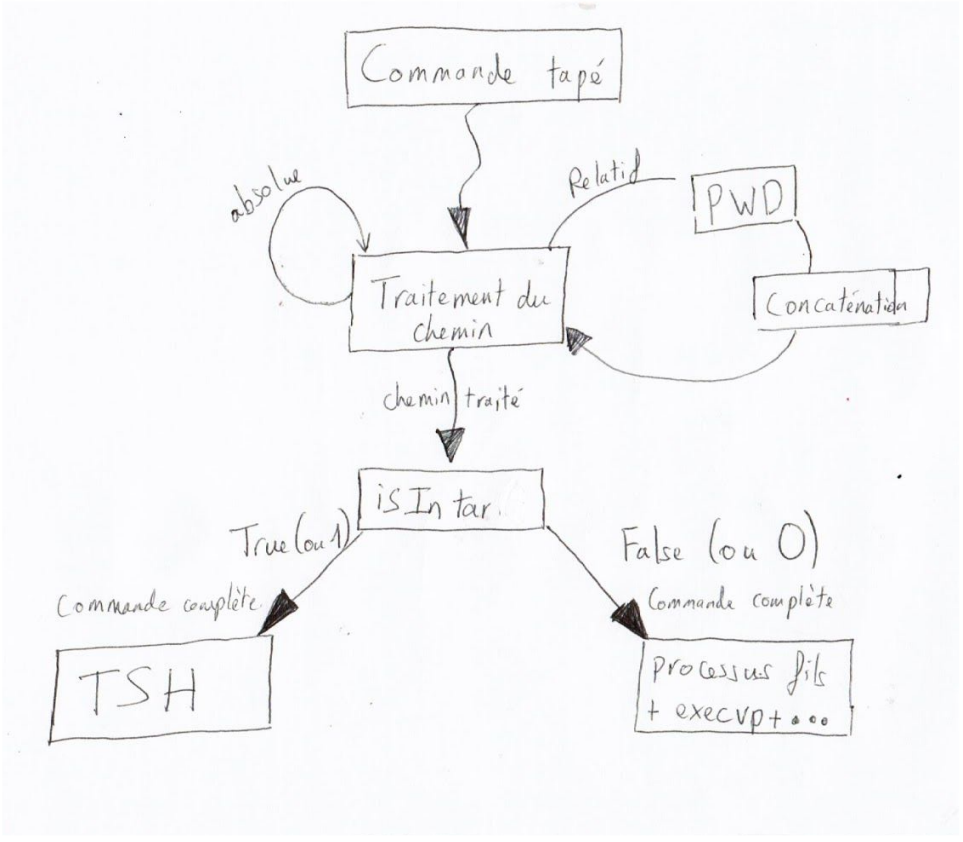
Au début, nous étions partis sur le développement de chacune des commandes sans impliquer les tar. Nous avons vite remarqué que c'était inutile et que la manipulation des tarballs nécessitait des fonctions auxiliaires avant de commencer. Nous avions ainsi à la 3-4ème semaine les fonctions auxiliaires de lecture du tar pouvoir réaliser les commandes de lecture, soit pwd, ls, cd , cat.
Nous avions aussi en parallèle pu faire un système pour pouvoir gérer les chemins pour le projet.
Suite au rendu 1, nous avons commencé à développer un shell pour pouvoir tester de façon dynamique nos commandes, commandes que l'on testait à la main auparavant.
Nous nous sommes ensuite concentrés à écrire des fonctions auxiliaire d'écriture de base pour pouvoir implémenter toutes les fonctions restantes, c'est-à-dire mkdir, rmdir, mv, cp et rm.
Il ne nous restait ainsi plus que les options à finaliser, nous avons fait en sorte que les options utilisent en général les versions des commandes de base.
Finalement nous avons ajouté des vérifications un peu partout dans les commandes pour éviter en générale les mauvaises utilisation des commandes.
Pour les deux dernières semaines du projet, nous nous sommes concentrés à faire la redirection et les combinaisons de commande. Cependant nous n'avons pas pu finir les combinaisons de commandes par manque de temps.

II. Gestion des chemins dans les tar.

Tout d'abord, pour sauvegarder le chemin courant du programme, nous avons choisi d'utiliser une variable globale cwd(current working directory) comme contenant.

Pour savoir si un chemin est dans un tar, il nous suffit de voir si l'extension “.tar” est présente dans le chemin.
Si un fichier est .. (retour en arrière) le chemin supprimera le fichier précédent.
exemple : aa/bb.tar/../cc -> aa/cc.
Dans le programme, tous les chemins sont convertis en chemin absolu sans . et .. pour pouvoir simplifier l’utilisation.
Ainsi pour savoir si une commande implique un tarball, il suffit de voir si l'extension est présente dans le chemin convertis. Cela marche ainsi peu importe les cas. (qu’on revienne d’un tar, qu’on change de tar , etc ...)

Fonction importante dans la gestion des chemins :
DividePathWithTar : Fonction qui permet de découper un chemin en deux parties, la première est le chemin vers l’archive, la deuxième est le nom/chemin du fichier.
exemple : /home/aaa/archive.tar/rep/fifi ->/home/aaa/archive.tar | rep/fifi



III. Fonctionnement générale d'une commande dans notre shell

Une commande fonctionnera de manière TRÈS générale de cette manière :

1/ La commande reçoit argv, dont les chemins seront déjà traités (ils seront convertis en chemin absolu et n'auront aucun . et ..)

2/ Il vérifie avant tout si il peut utiliser la commande avec ces arguments (vérifie si les fichiers n'existe pas déjà, si le chemin est valide, si la destination est un répertoire .....)

3/ Il vérifie si les arguments impliquent les tar à l'aide de la fonction isInTar() et si ce n'est pas le cas, utilise la fonction executeCommandExterne() pour utiliser exec facilement la commande.

4/ Dans le cas du tar, il exécute la commande en ouvrant l'archive à partir du chemin découpé (en utilisant la fonction dividePathWithTar() qui sépare un chemin en 2: le chemin sur l'archive et le chemin qui est dans l'archive).

IV. Implémentations des différentes fonctionnalité du TSH dans le tar

cat : Ouvre l'archive. Cherche le header avec le nom du fichier. Et si le header a été trouvé, récupère son contenu et l'affiche dans le STDOUT.
pwd : print la variable globale cwd.

cd : Ouvre l'archive. Si le répertoire existe, modifie cwd avec le nouveau chemin. Si pas dans le tar, utilise chdir et getcwd pour mettre à jour cwd. CD ne pouvait pas ici être fait avec exec ici car il ne modifierait pas notre variable globale cwd.

ls : Ouvre l'archive. Parcours l'archive. Si un header est dans le répertoire rechercher, le print.
-l : change le print en utilisant les informations du header.

mkdir: Créer un header dans le tar avec le nom correspondant (après avoir vérifié si il était possible de le créer à l'emplacement voulu), un typeflag = '5' et avec size = 0. Ouvre l'archive. Écrit le header dans le tar à la toute fin de l'Archive.

rm et rmdir: Ouvre l'archive. Cherche le fichier à supprimer. Met le curseur après le contenu du fichier. Sauvegarde tout le reste du contenu de l'archive dans un buffer. Cherche le fichier à supprimer. Supprime tout archive (à partir du fichier à supprimer). Passe l'archive. Écrit le buffer de sauvegarde dans l'archive. rmdir et rm se différencie par une vérification du type du fichier.
-r : parcours l'archive. Si le fichier à comme préfix le nom du répertoire à supprimer, le supprimer avec rm/rmdir. Revenir au début de l'archive et recommencer.

cp :

tar -> tar : Ouvre l'archive. Sauvegarde le header et le contenu. modifie le nom/chemin du header. Écrit dans l'archive le nouveau header et contenu à la fin de l'archive.

tar -> ext : Ouvre l'archive. Sauvegarde le header et le contenu du fichier. Ouvre la destination en Création. Écrit le contenu.

ext -> tar : Ouvre le fichier source. Recupère ses informations ainsi que son contenu. Créer un header avec ces informations. Écrit le header et son contenu dans l'archive.

-r : En général. Liste les fichiers et répertoires à créer avec des parcours. Applique soit mkdir soit cp selon le type des fichiers.
(tar -> ext : petite subtilité car il faut faire un tri par insertion par nombre de slash(ordre de priorité), aaa/ est plus prioritaire que aaa/bbb/ car sans aaa/ impossible de créer le second)

mv: répertoire : rmdir puis mkdir si répertoire
fichier : cp puis rm.

redirection :
ext : Ouvre le fichier en création. Redirige Stdout dans le fichier. exécute la commande. Rétablit Stdout.

tar : Ouvre le fichier en création un fichier trash. Redirige Stdout dans le fichier trash. Exécute la commande. supprime le fichier dans la destination. Copie le trash dans la destination. Supprime le fichier trash. Rétablit Stdout.