I Module de gestion d'option, optl

Le dévellopement du module d'option est la partie du projet qui nous a été le plus chronofage. En effet, nous avons dès le début eu pour objectifs de produire un module générique, réutilisable pour de nombreux programme. A l'instar du module getopt qui nous sembler manquer certain fonctionnalités très importante tel que la représentation des options par une chaine de charactère (représentation longue), mais aussi d'autre chose présente dans les option linux.

Dans un premier temps, il nous a fallu effectue des recherches sur la gestion d'option des programmes linux. C'est après ces recherches que nous avons remarquer avec stupeur que les programme linux n'ont pas de norme pour définir pour leur option. Nous avons donc décider de nous en créer une en nous inspirant des commandes tel que cat, ls ou encore rm du système linux.

Les options peuvent être représenter par deux indentificateur, une version courte et une longue. Les deux ne sont pas obligatoire mais une des deux doit au moins exister. Toute option à possibliter d'intérompre le traitement d'option. Toute option peut exiger un arguement pour effectué son traitement. Toutes les options peuvent avoir une description.

1 Défintion d'option

1.1 Les options courte

Une option courte est composer d'un '-' suivit d'un charactère alphanumérique. Pour donner un parametre à une option courte il s'uffit de le séparer d'un espace On peut faire appelle à plusieur option courte en un seul appelle, en suivant le '-' des charactère représentant les options voulus. Cependant, il ne peut y avoir dans cette forme d'appelle qu'un seul parametre demandant un arguement et il doit alors être le dernier de la list.

1.2 Les options longue

Une option longue est préfixé par la chaine '–'. Pour donner un parametre à une longe il faut s'éparer celle-ci de son argument par le charactère '='. <parler de l'autocomplétion des options longues>

1.3 Option help

Nous avons aussi décider de rendre obligatoire une option, l'option 'help'. Cette option représenter par '-h', '–help', doit afficher une possible description du programme, comment l'utiliser mais aussi la liste de toutes les options suivit de leur possible description. Cette option intéromp le traitement des possibles option suivante.

2 Implémentation

2.1 Spécification

Pour l'implémentation nous avons décider de permettre à l'utilisateur de pouvoir modifier certain chose. Notament les préfixe des option courte et longue (On peut notament envisager des utilisateur originaire windows qui préférerait utiliser le ‘
’ au ‘-’), l'indentificateur des deux représentation de l'option ‘help’. Nous avons aussi été contraint d'ajouter un spécifieur permétant de garantir que la valeur suivant celui-ci n'est pas une option (lui aussi est modifiable par l'utilisateur du module). Sans ce spécifieur, l'utilisateur ne pourrait pas rentrer la valeur ‘–help’ en prenant cette valeur non pas comme une option mais une valeur à traiter. Ce spécifieur vaux par défault la chaine ‘–’. De plus, nous avons mis en place un possible traitement sur les éléments qui ne sont pas des options.

2.2 Le code

Pour regrouper toutes les informations nécessaire à la gestion d'une option nous avons donc mis en place un type optparam regroupant toutes ces informations. La fonction opt_init initialise une instance de ce type. La véritable difficulter à été dans la conception de la fonction de traitement, opt_parse. Cette fonctoin peut être diviser de la façon suivante :

Pour chaque argument **de** tableau-argument **faire**
Si argument représente NEXT_NOPT **alors**

Traitement de NEXT_NOPT (le spécifieur que la prochaine valeur ne doit pas être considérer comme une option).

Sinon Si argument est une option longue **alors**

Traitement de l'option longue. Pour cela, un appelle a la fonction opt_parse_long visent a trouver l'option dont argument est le seul préfix. Puis la fonction liée à l'option est éxecuter.

Sinon Si argument est une option courte **alors**

Traitement des possibles options courtes. Pour ce faire, un traitement sur chaque charactère est effectué permettant de trouver toutes les options représenté dans argument. Puis effectue le traitement liés à ces appelles.

Sinon

Traitement de ce qui n'est pas une option, à l'aide la fonction hdl_dlt possiblement fournit par l'utilisateur.

Fin Si
Fin Pour chaque

FIGURE 1 – Traitement du tableau des arguement par la fonction opt_process.

3 Pour et contre du module

Ce module à été consu pour être utiliser dans de nombreux cas. Il pourrait donc reservi pour des futurs projets. Cette généraliter entraine par contre une grande difficulter quand à la compréhension de ces fonctionnalités, on peut notament citer la Spécification de la fonction otp_process qui est trop longue. Un point fort du reste qu'il est complet, il met en pratique toute les fonctionnalités des commandes linux les plus connu (rm, ls, cat), notament d des fonctionnalités tel que 'l'autocomplétion' des options longues, la prise en charge des multiple option courte ou encore le spécifieur NEXT_NOPT (next is not an option). Un autre point fort, la compléxiter du traitement des options, toues les options ne sont parcourue qu'au plus deux fois pour exercuter leur traitement (avec la possible comparaison avec l'option 'help'). De même ce module ne nécessite que le stockage des objets de type optparam pour traiter les options.

II Bibli

Lien :

Présentation des divers 'tradition' de gestion d'option pour unix et gnu, `http://www.catb.org/~esr/writings/taoup/html/ch10s05.html`.

Descrpition de la gestion des option en ligne à la GNU, `https://www.gnu.org/software/gawk/manual/html_node/Options.html`.

Documentation du module getopt, pour possiblement faire une comparaison entre ce module et optl, `https://www.gnu.org/software/libc/manual/html_node/Getopt.html`.