Date limite : vendredr 14 janvier 2005

Projet systèmes d'exploitation
L3, parcours informatique, 2004-2005

Projet Mish (Shell minimal)

Description du projet

Dans le sixième cours on a étudié le comportement d'un shell, et en particulier du shell Bash. Il s'agit d'abord d'une interface avec le système d'exploitation permettant à l'utilisateur l'exécution d'un programme ou d'une suite de programmes. Dans le projet Mish, on implémentera un shell minimal. On se chargera, en particulier, d'implémenter les points suivants.

Exécuter des commandes. Mieux, il faudra pouvoir exécuter des suites des commandes. Ces suites peuvent être construites à l'aide des opérations `;` `&` `|`, dont la sémantique est comme pour le shell Bash :
`;` : exécution séquentielle,
`&` : exécution asynchrone en arrière plan,
`|` : exécution asynchrone avec redirection du `stdout` du premier processus sur le `stdin` du deuxième processus.

Redirection de Entrée/Sorties. Il faudra
– pouvoir rediriger le `sdtin` à partir d'un fichier donné à l'aide de l'opération `<`,
– pouvoir rediriger le `sdtout` sur un fichier donné à l'aide de l'opération `>`.

Intercepter les signaux. Par exemple, un `CTRL-C`qui provient de la console n'est pas destiné au shell, mais à la programme que le shell est en train de faire dérouler en avant-plan.

Commandes internes. Il faudra pouvoir sortir du shell. On implémentera donc une commande interne `exit`. Aussi on aimerait tester la valeur de retour d'une commande. Pour cela on peut penser d'implémenter une commande telle que `$?` (sans nécessairement implémenter un mécanisme de substitution de variables. On implémentera aussi un commande interne `help` donnant une concise description de comment utiliser votre shell.

Mécanisme d'expansion/quoting. Par exemple, on peut implémenter une mécanisme de recherche de tous les fichiers dans le répertoire courant à l'aide `*` : dans ce cas la signification de la chaîne `\*` sera le caractère étoile (déprivé de sa signification de recherche de fichier).

Gestion des taches. Ce mécanisme est implicite dans notre travail : par exemple on veut parler de processus en avant-plan ou arrière-plan. Il faudra en particulier penser à la précédence entre les opérations `;`,`&`,`|`,`<`,`>`, et donc comment interpréter une ligne de commande pour construire une unité logique.
On peut penser d'implémenter de autres mécanismes, mais cela est laissé a votre goût et temps à disposition.

Modalités de travail.

On travaille impérativement par binômes. Inclure le noms du binôme dans le fichier `README` et dans chaque fichier source.

Modalités de soumission.

Préparer un répertoire contenant le code source, le fichier `README`, le fichier `MAKEFILE`, et rien d'autre. Prendre contacte avec Cyrill Terrioux, ou bien avec Luigi Santocanale, selon votre groupe (plus de détailles à venir). Date limite de soumission : vendredi 14 janvier 2005.

Modalités d'évaluation

Dans l'évaluation de votre projet importance particulière sera donnée aux critères suivants :

Documentation. Votre projet contiendra un fichier `README` contenant :
- La description du programme adressé à un utilisateur non-expert.
- Une description concise du programme (adressés aux évaluateurs) décrivant les choix d'implémentation, les structures de données utilisés, les résultats accomplis par rapport à chaque tache décrite ci-dessus.

Qualité du Code C. Le code source sera bien commenté, les noms des variables, des fonctions, des structures, et des types seront bien choisis. On donnera préférence à des fonction de petite taille. On utilisera de macro-constantes aux lieux des constantes explicites.

Organisation du programme en unités logiques. Chaque unité logique sera contenue dans un fichier séparé `nom.c` (avec `nom` est bien choisi) ayant un fichier en tête `nom.h` par défaut. La liste des unités logiques (donc la liste du code source) apparaîtra dans le fichier `README`, la compilation sera géré par le programme `make` à l'aide d'un fichier `Makefile`.

Fonctionnement de votre code. Le code source sera compilé au moment de l'évaluation, possiblement sur des machines différentes des machine SUN. Écrire donc votre code en suivant le standard POSIX. Votre présence n'est pas nécessaire par l'évaluation (toute explication éventuelle doit être contenue dans le fichier `README`). Le fonctionnement de votre code sera testé par rapport aux résultats déclarés accomplis dans le fichier `README`.