IUP MIAGE
Système d'exploitation

TP — Séquence de commandes

Ce TP illustre une utilisation des primitives `getenv ()`, `fork ()`, `wait ()`, `waitpid ()`, `exec ()` (avec ses variantes), et `dup2 ()`.

Consultez les documents de TD et/ou les pages de manuels pour ces différentes primitives.

1 La commande

Le but de ce TP est d'écrire un programme C qui réalise un équivalent de la commande

```
% (echo '#' ; cat ~/.cshrc ; echo '# eof') | sed '/^$/d'
```

Cette commande produit dans un premier temps un résultat formé de la concaténation des résultats des deux `echo` et du `cat`. Ce résultat est ensuite passé à `sed` qui supprime les lignes vides.

Pour tester la commande, on pourra créer un petit fichier `~/.cmd` et implanter une version modifiée de la commande :

```
% (echo '#' ; cat ~/.cmd ; echo '# eof') | sed '/^$/d'
```

Remarquez que les quotes n'apparaissent dans la commande ci-dessus que pour combattre la tendance du shell à interpréter lui-même les caractères spéciaux. Elles sont inutiles avec `execlp ()`.

Par contre, le sens du caractère ~ est géré par le shell, donc `execlp ()` ne le comprendra pas. Il sera donc nécessaire de récupérer la valeur de la variable `$HOME` par la primitive `getenv ()`.

2 Les processus

L'exécution de cette commande repose sur 5 processus :
– un processus pour chacune des commandes `echo`, `cat`, et `sed` ;
– un processus père de toutes ces commandes.

Les trois commandes de l'ensemble `echo`, `cat`, `echo` s'exécutent les unes après les autres. Chacune d'elles produit son résultat dans un même tube.

La commande `sed` peut être lancée dès le début ; elle lit ses données depuis ce même tube.

À la terminaison des 3 commandes (`echo`, `cat`, `echo`), `sed` peut se terminer. Cette terminaison est détectée par la fermeture du tube.

Le processus père se termine à la terminaison de tous ses fils.

Id: tpcmd.tex,v 1.9 2001/02/15 15:25:30 marquet Exp