



R 1.04

2023 - 2024

Introduction aux systèmes d'exploitation et à leur fonctionnement

TD N°6

« Communication entre processus : première approche »



ANNE Jean-François
D'après le TD de F. BOURDON

Le but de ce TD est de se familiariser avec les systèmes d'exploitation et avec leur fonctionnement.

« Communication entre processus : première approche »

Notions vues dans ce TD :

« | » (pipe), grep, tee, ps, wc.

Nombre de séance de 2h prévu pour faire ce TD : 1.

PS : Les parties correspondant à du travail à faire sont toutes en italiques ; le restant étant du complément au cours.

1. Exercice 1

Contrairement aux processus lancés simultanément qui s'exécutent sans relation entre eux, des processus concurrents sont synchronisés entre eux par la production d'information de l'un, et la consommation d'information de l'autre. Il est nécessaire pour cela, que le processus producteur soit apte à produire des caractères sur la sortie standard, et que le processus consommateur soit apte à lire des caractères sur l'entrée standard.


C'est ce que réalise la commande « pipe » dont la syntaxe est :

commande1 | commande2

Exemple :


\$ ps -aix | grep bash


La commande « ps » produit la liste des processus qui tournent sur la machine. La commande « grep » affiche les lignes d'un fichier passé en paramètre qui incluent la chaîne de caractères passée également en paramètre ; le pipe « | » sert de raccordement de la sortie standard du processus « ps » sur l'entrée standard du processus « grep ». Ce dernier n'affiche que les lignes contenant la chaîne de caractères « bash ».


 *Exécuter la ligne de commande de l'exemple ci-dessus. Dessinez une boîte par commande en représentant les trois canaux d'entrée-sortie de base (0, 1 et 2) ; reliez sur votre dessin ces boîtes entre elles par le pipe.*


 *Créer un fichier texte de nom « text1 » dont le contenu est le suivant :*

- 1 : la commande *pipe* sert à la communication entre processus
- 3 : la commande *tee* est utile pour capturer les informations qui circulent dans un pipe
- 2 : la commande *tee* recopie son entrée standard sur sa sortie standard et sur un fichier
- 4 : la commande *tee* peut être utilisée pour sauvegarder dans un fichier les traces des informations qui circulent sur sa sortie standard.
- 1 : qu'est-ce qu'un *pipe* et que fait la commande *tee* ?

 En utilisant le pipe « | » écrire une commande qui compte le nombre de ligne contenant le mot « pipe » dans le fichier « text1 » ; proposez une première solution en utilisant les commandes « wc » (après avoir été dans le manuel pour comprendre son fonctionnement) et « grep ».

 Proposez une deuxième solution en utilisant deux pipe et les commandes « cat », « grep » et « wc ». Pour chacune de ces solutions dessinez le schéma avec les boîtes (cf. la question précédente) qui se raccordent entre elles.

 Regarder dans le manuel la commande « tee ». En utilisant cette commande modifier le résultat obtenu précédemment afin de récupérer dans un fichier « text2 » les lignes contenant le mot « pipe ». Comme pour les questions précédentes vous ferez le dessin des boîtes et des raccordements entre elles. Que concluez-vous du rôle de la commande « tee » ?

 Ecrire une ligne de commande qui permette de créer un fichier « text3 » qui contiendra les lignes du fichier « text1 » contenant le mot « pipe ». Ces lignes devront être triées sur le premier champ de chaque ligne. Enfin la commande affiche le nombre de ces lignes. Proposez une première solution avec les commandes « grep », « sort », « tee » et « wc », puis une seconde avec les commandes « cat », « grep », « sort », « tee » et « wc ». Dans les deux cas dessinez les boîtes correspondantes.

 Que fait la ligne de commande suivante ?

```
prompt> df -k . | tail -1 | sed "s/ * / /g" | cut -d " " -f 4
60833156
prompt>
```


Vous pourrez décomposer chaque étape de la chaîne de tubes et vous aider du manuel pour comprendre le résultat obtenu. Il est à noter que la commande « sed » utilise une instruction que nous avons vu avec l'éditeur vi/vim. Elle consiste à substituer la séquence espace-étoile-espace par espace... c'est-à-dire remplacer une suite de blancs par un seul.

 Expliquez ce que va faire la ligne de commandes suivante ? Vous pouvez faire un schéma des processus créés et des liens existants entre eux.

```
prompt> cat | cat
blabla (retour chariot)
...
(CTRL+D)
```

2. Exercice 2

 Combien y-a-t 'il de processus actifs sur le système ? Regardez dans le manuel la commande « ps » et l'option « a ».

 Combien d'utilisateurs sont connectés sur le système ? Que fait l'option « -q » de la commande « who » ? En utilisant la commande « sort », affichez la liste triée de ces utilisateurs connectés :

- par ordre alphabétique.

- selon l'heure de connexion.

- ✍ Combien le répertoire « /etc » a-t-il de fichiers répertoire ? Utilisez les commandes « ls », « grep » et « wc ».
- ✍ Combien le répertoire « /etc » a-t-il de fichiers répertoire ? Utilisez les commandes « ls », « grep » et « wc ». Attention il faut prendre en compte les fichiers cachés. Le caractère « ^ » utilisé dans la commande « grep » désigne le premier caractère d'une ligne.
- ✍ Combien le répertoire « /etc » a-t-il de sous-répertoires ? Utilisez les commandes « ls », « grep » et « wc ». Attention tous les fichiers répertoire d'un répertoire donné, ne sont pas nécessairement des sous-répertoires pour ce dernier. Expliquez cette nuance.
- ✍ Après avoir regardé dans le manuel la fonction « ls », proposez une première solution avec l'option « -l » du « ls », puis une seconde avec l'option « -ld » de cette même commande. Vous préciserez le rôle de l'option « d » pour la commande « ls ».
- ✍ Enfin proposez une troisième solution avec l'option « -p » de la commande « ls ». Cette option « p » permet d'afficher les noms de répertoire avec un « / » à la fin du nom. Pour cette dernière solution le caractère « \$ » placé en fin de chaîne, dans la commande « grep » indique que la chaîne recherchée se place en fin de ligne. En mettant « /\$ » en paramètre du « grep » on recherche les lignes dont le dernier caractère est un « / ».
- ✍ Affichez la liste des fichiers du répertoire courant, triée par ordre de taille des fichiers. La taille des fichiers est renseignée par le 5^{ème} champ (i.e. le champ d'indice 4) avec la commande « ls -l ».
- ✍ Proposez une deuxième solution qui n'affiche cette liste triée par ordre de taille, uniquement pour les fichiers ordinaires (« - » en début de ligne de la commande « ls -l »).

3. Exercice 3

La commande « cat » sans paramètre lit l'entrée standard et affiche sur la sortie standard.











L'opérateur « > » permet de rediriger la sortie vers un fichier. Utilisez la commande « cat » et l'opérateur « > » pour créer d'abord le fichier « fich1 », puis le fichier « fich2 » en leur donnant les contenus suivants, pour le fichier « fich1 » :

- Lecture de l'entrée standard
- Redirection dans le fichier fich1

et pour le fichier « fich2 » :

- Lecture de l'entrée standard
- Redirection dans le fichier fich2

Pour sortir de la saisie dans la commande « cat », après avoir fait un retour chariot (touche « entrée » au clavier), appuyez simultanément sur les touches « CTRL » et « D ».

-  Que constatez-vous si vous refaites l'opération sur le fichier « fich1 » en changeant le contenu entré ?
-  Toujours en utilisant la commande « cat » mais cette fois en regardant le manuel, créez le fichier « fich3 » constitué de la concaténation des fichiers « fich1 » et « fich2 ». Proposez deux solutions, l'une en utilisant l'opérateur « > » et la seconde en utilisant l'opérateur « >> ».
-  Lancez la commande « cat fich1 fich-inexistant » avec le fichier « fich-inexistant » inexistant et le fichier « fich1 » existant. Les messages affichés sortent-ils sur la même sortie de la commande « cat » ? Vous pouvez faire un schéma.
-  Nous pouvons rediriger la sortie standard en utilisant l'opérateur « > » ; par exemple « cat fich1 fich-inexistant > trace » ou « cat fich1 fich-inexistant 1>trace ».
-  Lancez les deux commandes ; que constatez-vous ? Que peut-on dire à propos de la sortie standard et de la sortie d'erreur ?
-  Modifiez la commande précédente (l'une ou l'autre de la question précédente) en redirigeant la sortie d'erreur dans le fichier « err ». Vous pouvez faire un schéma qui fait apparaître la commande « cat » avec ses différentes entrées/sorties.
-  En modifiant la commande obtenue dans cette question, comment peut-on rediriger la sortie standard sur la sortie d'erreur ? Faites un dessin du résultat.
-  L'ordre des redirections est-il important sur la ligne de commande ? Pour s'en rendre compte vous pouvez modifier cet ordre et observer le résultat.
-  La commande « find » permet de rechercher des fichiers, suivant des critères précis, à partir d'un nœud dans l'arbre des fichiers, et de façon récursive dans toutes les branches dépendant de ce nœud. Après avoir regardé le manuel pour la commande « find », lancez la commande suivante :
- ```
prompt> find / -name passwd.
```
-  Modifiez cette commande afin de ne plus voir les messages d'erreur sur votre terminal. Vous utiliserez le terminal « poubelle » /dev/null.

## **II. Webographie :**

- <https://bourdon.users.info.unicaen.fr/cours/IUT-1A/index.html>
-