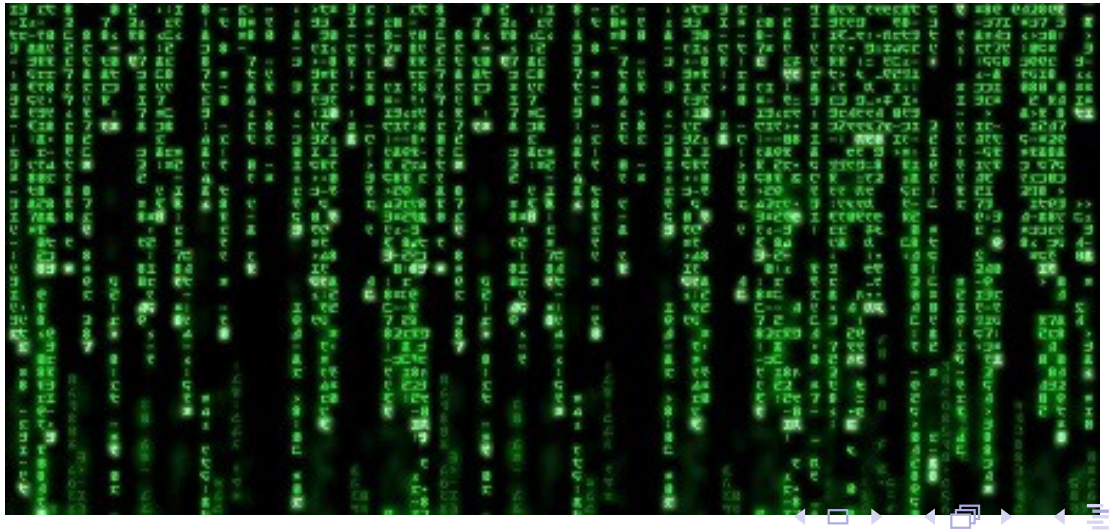
Quelques notions essentielles sur la ligne de commandes (cli en anglais)

raphael.pasquier@univ-antilles.fr

Université des Antilles

20 février 2024



Brève histoire des interfaces Homme-Machine

Prochaine interface du cluster **Exocet**...

Traitement par lots ou **batch**

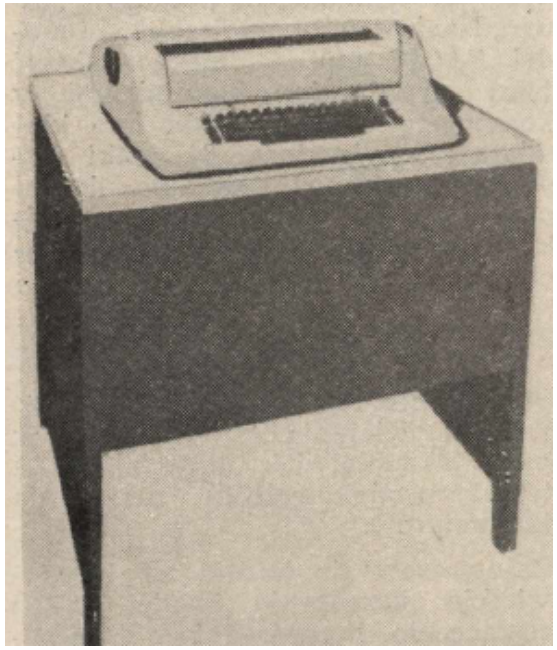
Des années 40 jusqu'aux années 60. Fonctionnement :

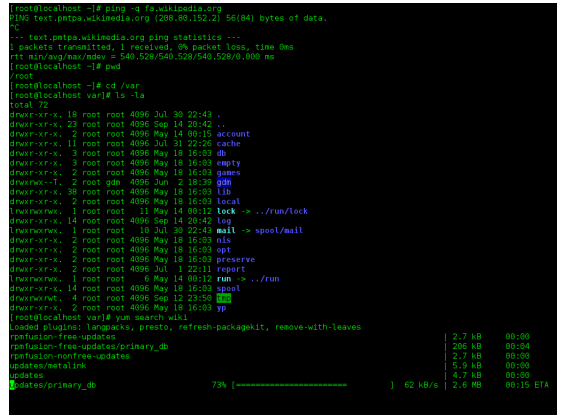
1. On entre les données et le programme sous forme de carte-perforées ou de bandes magnétiques.
2. On démarrage l'exécution du programme (ensuite il n'y a plus d'intéraction avec l'ordinateur).
3. On récupére les résultats (corrects ou erronés) sur papier imprimé ou bandes magnétiques.

Photo de la console de l'UNIVAC 1 où on voit Grace Hopper.

Interface de ligne de commande (CLI)

Des années 60 aux années 80. Apparition des télétypes puis des terminaux. Début de l'interface de ligne de commande comme première et véritable communication intéractive HM sous forme de **session** avec un clavier et un écran (Louis Pouzin).

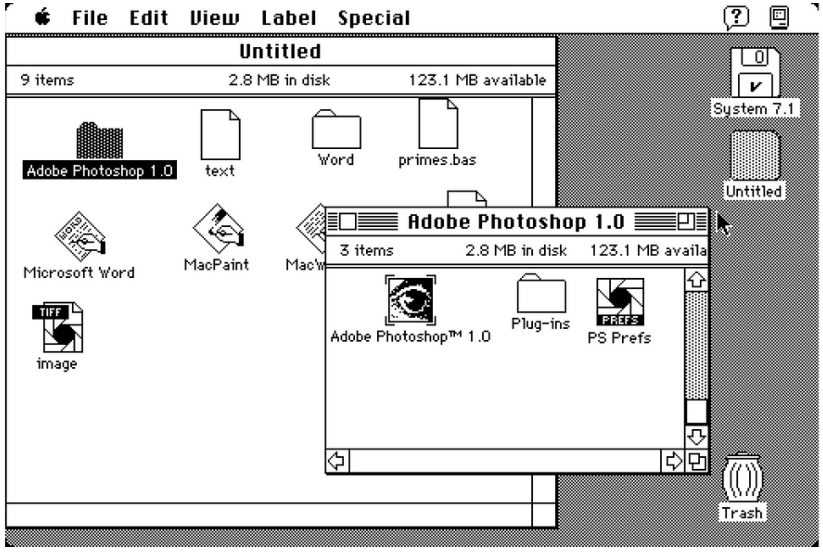


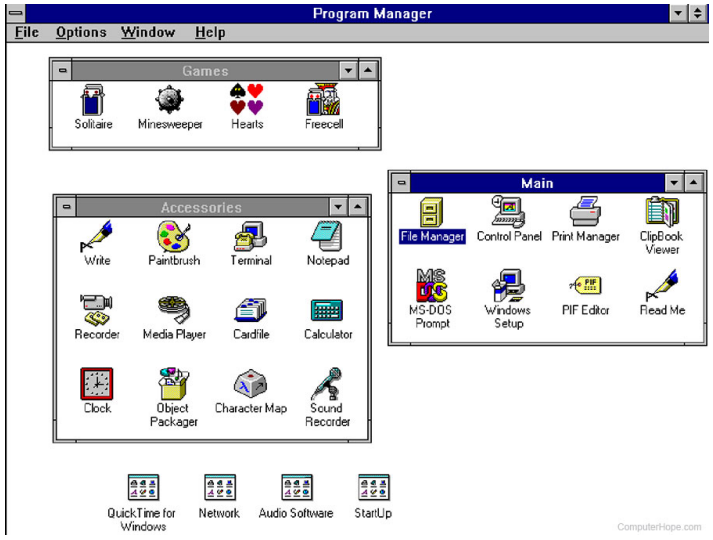


Ils permettaient à plusieurs utilisateurs de se connecter à un seul ordinateur qui était très cher.

Interfaces graphiques du type environnement de bureau

À partir des années 80, apparition des interfaces graphiques du type **environnement de bureau** (Macintosh, Windows 3.1, etc...) avec crayon optique puis souris (avec ou sans boule).





Interfaces graphiques avec écran tactile

À partir des années 2000, apparition des écrans tactiles. L'interface suit le paradigme "Web" : on passe d'une fenêtre à une autre.

Présentation

Une interface de ligne de commande (**Bash**) ressemble à :

```
martin@monPC:~$
```

L'interface de ligne de commandes (CLI ou **Shell**) est en fait un petit programme qui permet à un utilisateur d'écrire une séquence de caractères (ligne) après le prompt (symbole $ pour Bash) pour exécuter un ou des programmes. Cette ligne doit contenir au moins le nom d'un programme avec d'éventuelles options et données. Une fois la touche "Entrée" enfoncée (validation de la ligne), le programme lit la ligne, puis l'interprête. Si la syntaxe de la ligne est correcte, le programme dont le nom apparaît dans la ligne sera exécuté. Une fois l'exécution terminée, une nouvelle invite (ou prompt) apparaîtra en dessous et l'utilisateur pourra de nouveau taper une nouvelle ligne de commande(s).
Il existe plusieurs **Shell** (ex. **Bash**, **ksh**, **PowerShell pwsh**, etc).

Syntaxe pour exécuter un seul programme en ligne de commande

```
martin@monPC:~$ program param_1 ...  param_n [&]
```

où `param_i` est soit :

- une ou plusieurs options du programme (souvent la chaîne de caractères commence par le signe - ou deux signes -),
- un nom de fichier,
- une donnée.

program peut aussi avoir aucun paramètre.
Exemples :

```
$ ls
$ gunzip projet.zip
$ rm *.log
$ latexila article.tex &
$ g++ -O3 -lm main.cpp -o mon_appli
$ sleep 10
```

Liste des commandes (programmes) les plus utiles :

- `ls` : affiche les fichiers et les répertoires d'un répertoire dans l'écran du terminal,
- `cd` : change le répertoire courant ou permet de revenir à son répertoire personnel,
- `pwd` : affiche à l'écran le répertoire courant,
- `cp` : copie un fichier, ou des fichiers, ou un ou plusieurs répertoires,
- `mv` : déplace ou renomme un fichier, ou un répertoire, ou plusieurs,
- `mkdir` : crée un répertoire (ou plusieurs),
- `touch nom_fichier` : crée un fichier nommé `nom_fichier`,
- `rm` : supprime un ou plusieurs fichiers (voir des répertoires). **ATTENTION! :** Il est impossible d'annuler l'opération.

De nombreux programmes sont utilisables en ligne de commandes.

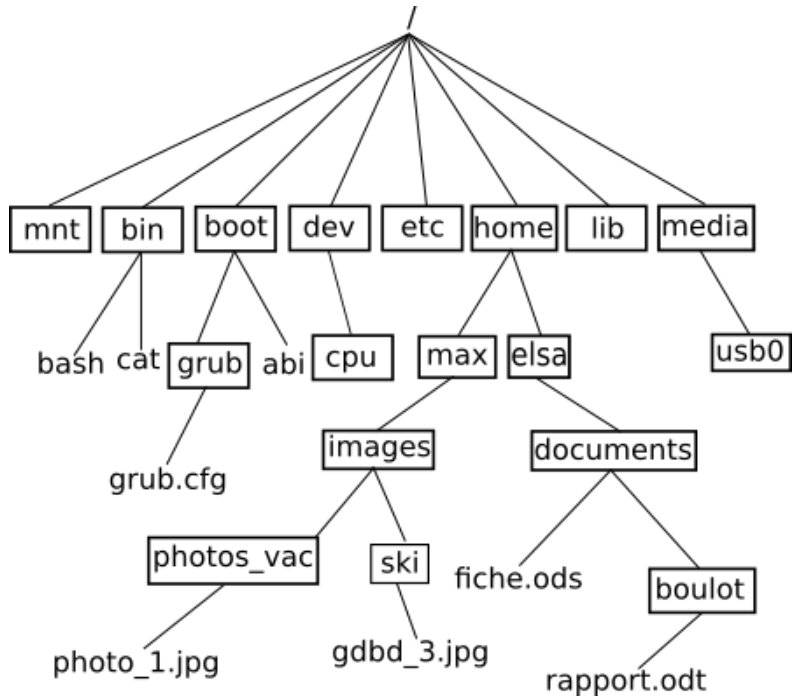
Quelques propriétés de la ligne de commandes

* On peut se déplacer dans la ligne avec les flèches $\leftarrow$ et $\rightarrow$ du clavier pour corriger des erreurs (avant de valider bien sûr) et `Ctrl+A` pour venir en début de ligne,
* On peut remonter dans l'historique des commandes déjà tapées avec la flèche $\uparrow$ ou redescendre $\_$.
* On peut "compléter" le nom d'un programme, d'un nom de fichier ou de répertoire du répertoire courant avec la touche tabulation du clavier $\twoheadrightarrow$.
* On peut avoir de l'aide sur la syntaxe d'un programme avec le programme `man`. Exemple :
  - `man ls`

  Pour quitter le programme `man`, on tape sur la lettre q sans valider.
* Pour arrêter un programme, taper la combinaison `Ctrl+C`.

Organisation des fichiers sur un système UNIX

Les fichiers sont organisés à l'aide du concept de répertoire. Un répertoire peut contenir des fichiers ou des répertoires. On représente cette organisation sous la forme d'un arbre (structure informatique) :



Le répertoire qui contient tous les répertoires s'appelle le répertoire **racine** (root en anglais) (symbole slash /).

Chemin absolu

Le chemin qui permet d'atteindre un fichier (ou un répertoire) depuis la racine s'appelle **le chemin absolu** du fichier (ou du répertoire). Exemples :

- `/etc` (chemin absolu du répertoire `etc`),
- `/bin/bash` (chemin absolu du fichier `bash`),
- `/boot/grub/grub.cfg`

Le symbole `/` sert de séparateur entre les noms.

ATTENTION : Ne pas confondre un séparateur `/` avec la racine.

Quel est le chemin absolu du fichier `gdbd_3.jpg` ?

À chaque fichier (ou répertoire) correspond un **unique** chemin absolu (nom complet du fichier ou du répertiore).

Chemin relatif

Le chemin qui permet d'atteindre un fichier (ou un répertoire) depuis un répertoire $\mathcal{R}$ donné s'appelle le **le chemin relatif** du fichier (ou du répertoire) par rapport à ce répertoire $\mathcal{R}$. Par exemple, le chemin relatif du fichier gdbd_3.jpg depuis le répertiore `/home/max` est :

- `images/ski/gdbd_3.jpg`

Un chemin relatif ne commence jamais par / .
Si on concatène le chemin absolu de $\mathcal{R}$ avec le chemin relatif, on obtient le chemin absolu du fichier ou du répertoire.

Contexte d'exécution

On associe à une ligne de commande une **liste de variables d'environnement**. Cette liste constitue l'**environnement d'exécution** des programmes qui seront exécutés dans le shell.

Une variable d'environnement est un couple (name, value) où "name" est le nom de la variable (chaîne de caractères en majuscule) et "value" est sa valeur (chaîne de caractères) .
Exemple :

- (`PWD`, `/home/elsa`)

Une variable très importante est la variable `PWD` qui contient le chemin absolu d'un répertoire : c'est le chemin du "répertoire courant". On peut afficher sa valeur en exécutant le programme `pwd` :

- `pwd`

On peut changer sa valeur avec le programme `cd` (Change Directory).
La valeur de la variable `PWD` peut être utilisée par les programmes exécutés dans le **Shell**.

Définition de quelques concepts relatif aux chemins

1. Le tilde ~ représente le répertoire personnel ("**home**") de l'utilisateur . Par exemple, pour l'utilisateur Max :
   - ~ est égal à `/home/max`
2. Par convention, le symbole `..` signifie "répertoire parent du précédent répertoire dans le chemin".
   Exemples :
   - `/dev/cpu/..` est égal à `/dev`,
   - `/home/max/images/../..` est égal à `/home`,
   - Le chemin `/home/max/../elsa/documents` existe,
   - `..` est égal au répertoire parent du répertoire courant (convention),
   - `../..` est égal au répertoire parent du répertoire parent du répertoire courant,
3. Enfin le symbole point `.` représente "répertoire courant".

Exemples d'utilisation de commandes :

- `ls .`
  ou simplement
- `ls`
- `cd ..`
- `mv ../fichier.fasta .`
- `ls ..`
- `ls ~/documents`
- `rm gen*.fasta`

Sujets importants non abordés :

- Les éditeurs de texte en ligne de commandes (`vim`, `emacs`, `nano`). Ces programmes n'ont pas besoin d'une interface graphique.
- Les flux (d'octets), entrée et sortie standards.
- La communication entre plusieurs programmes d'une même ligne de commandes (pipe |). Dans une ligne de commande, la sortie d'un programme peut devenir l'entrée d'un autre programme.
- Ls fichiers spéciaux UNIX (lien symbolique, fichier représentant un périphérique, etc).

Ressources pour les exercices :

- https://doc.ubuntu-fr.org/tutoriel/console_commandes_de_base
- https://doc.ubuntu-fr.org/commande_shell
- https://doc.ubuntu-fr.org/tutoriel/console_ligne_de_commande

Références :

Livre :

- Histoire illustrée de l'informatique de E. Lazard et al
- LINUX Initiation et utilisation de J.P. Armspach et al.
  Chapitre 1, 2, 3, 4, 6, 7 et 11

Web : Ligne de commandes :

- https://fr.wikipedia.org/wiki/Interface_en_ligne_de_commande

Organisation des fichiers sur Linux :

- Page Wikipedia Répertoire_(informatique)
- https://doc.ubuntu-fr.org/arborescence
- https://doc.ubuntu-fr.org/chemins
  Chapitre 1 et 2

Chemin absolu, chemin relatif :

- https://doc.ubuntu-fr.org/terminal
  Chapitre 1, 3, 4 et 5
- https://doc.ubuntu-fr.org/tutoriel/console_ligne_de_commande
  Tous les chapitres sauf le 12

Les commandes de bases :

- https://doc.ubuntu-fr.org/tutoriel/console_commandes_de_base
  Chapitre 1 et 2

Histoire des interfaces HM :

- https://fr.wikipedia.org/wiki/Traitement_par_lots
- https://fr.wikipedia.org/wiki/Interface_en_ligne_de_commande