1 Histoire

Pour comprendre l'intérêt des systèmes d'exploitation il faut faire un peu d'Histoire :

1. **1945-1957 : Première Informatique : l'ère des ordinateurs**

   (a) Technologie principale : Les lampes à vide.

   (b) Les ordinateurs sont peu nombreux coûtent très cher et consomment beaucoup d'énergie

   (c) Ils sont principalement utilisés par les universités et l'Armée pour des calculs scientifiques

   (d) Les ordinateurs ont à leur service de nombreux chercheurs, ingénieurs et techniciens pour leur fonctionnement

   (e) Progressivement apparaissent des modifications pour améliorer leur **ergonomie** (adaptation d'un environnement de travail (outils, matériel, organisation...) aux besoins de l'utilisateur), ainsi en 1953 avec l'IBM 701 l'entrée des données se fait avec des bandes magnétiques pour gagner du temps

   (f) Avec l'IBM 704 en 1955 (image ci-dessus) **pour optimiser le temps de calcul** d'une machine qui a coûté plusieurs millions de dollars, deux des principaux clients ayant acheté cet ordinateur, General Motors et North American Aviation ont créé **le premier système d'exploitation**, nommé GM-NAA I/O pour General Motors and North American Aviation Input Output System.

   Pourquoi et comment ?

   A l'époque le temps de calcul d'une machine comme l'IBM 704 était organisé suivant un **traitement par lots** :

Un **lot** est constitué d'une entrée des données et du programme (par bandes magnétiques) puis de l'exécution du programme et enfin l'impression des résultats du programme sur un télétype (une sorte d'imprimante)

Les lots devaient ensuite se succéder en cadence pour profiter au mieux de l'ordinateur

Mais la phase d'impression, purement mécanique, ralentissait la cadence aussi les ingénieurs ont eu l'idée de détourner la phase d'impression vers une phase de mémorisation sur une mémoire auxiliaire électromagnétique et **surtout que la gestion de toutes ces phases pour tous les lots allait se faire par un super-programme**, une sorte d'arbitre des programmes, placé en premier dans la mémoire de l'ordinateur
A l'époque on a appelé ce programme, un moniteur

Pour optimiser encore plus le temps, le moniteur pouvait faire chevaucher un temps de calcul pour un programme P_1 et un temps d'entrées-sorties pour un autre programme P_2 et ainsi de suite.

A partir de maintenant les programmes clients du processeur sont appelés des processus gérés par le système d'exploitation

(g) **En gestation :**

Le transistor apparaît en 1947 il remplacera progressivement les lampes à vide

2. **1957-1970 : Deuxième Informatique : l'ère des mini-ordinateurs**

Les transistors ont remplacé les tubes à vide, la taille des ordinateurs a diminué on parle maintenant de mini-ordinateurs (la taille d'un réfrigérateur pour le PDP-8 de Digital Equipment (image ci-dessus))

Cependant un transistor occupe encore une certaine place, l'époque des circuits intégrés n'est pas encore arrivée

De plus la mémoire des ordinateurs est constituée de tores de ferrites ce qui occupe aussi de la place

Pour optimiser le rendement du processeur les systèmes d'exploitation évoluent aussi :

1961 : CTSS (Compatible Time-Sharing System), est le **premier système d'exploitation de temps partagé** développé au MIT

L'échelle de temps du processeur (la micro seconde à l'époque, la nano seconde actuellement en 2022) n'est pas celle de l'Humain (la seconde) Bien que **le concept de machine de Von Neumann implique que le processeur exécute chaque instruction l'une après l'autre** on peut encore gagner du temps en entremêlant les instructions des processus, le système d'exploitation étant une sorte d'agent de la circulation faisant circuler tel processus P_1 puis tel processus P_2 etc...

L'Humain a ainsi l'illusion que plusieurs processus s'exécutent en même temps ce qui est un progrès par rapport au traitement par lots

Avec le système CTSS apparaît un langage interprété **le shell** qui permet le lancement des programmes et le "dialogue" entre l'utilisateur et le système (on verra quelques exemples plus loin)

1965 : Multics (Multiplexed Information and Computing Service), successeur de CTSS, améliore la gestion et la sécurité de la mémoire :

(a) Mémoire segmentée

(b) Mémoire virtuelle paginée

(c) Système de gestion de fichiers (que l'on regardera un peu en détail plus loin)

1969 : **Unix**, successeur de Multics et ancêtre des systèmes d'exploitation,Linux, Mac Os des ordinateurs Mac de Apple, iOs et Android

2 Principes généraux d'un système d'exploitation

Lorsqu'on met sous tension un ordinateur un ensemble de programmes composant le système d'exploitation est alors exécuté.

Le système d'exploitation contient :

1. Les pilotes de périphériques (claviers ,disques durs, cartes graphiques ...)
2. La gestion de la connexion au réseau
3. Les différents systèmes de fichiers (disques, clé usb ...)

De plus lorsque différents programmes sont lancés (on parle alors de processus), le système d'exploitation attribue à ces derniers les ressources dont ils ont besoin

1. quantité de mémoire (vive)

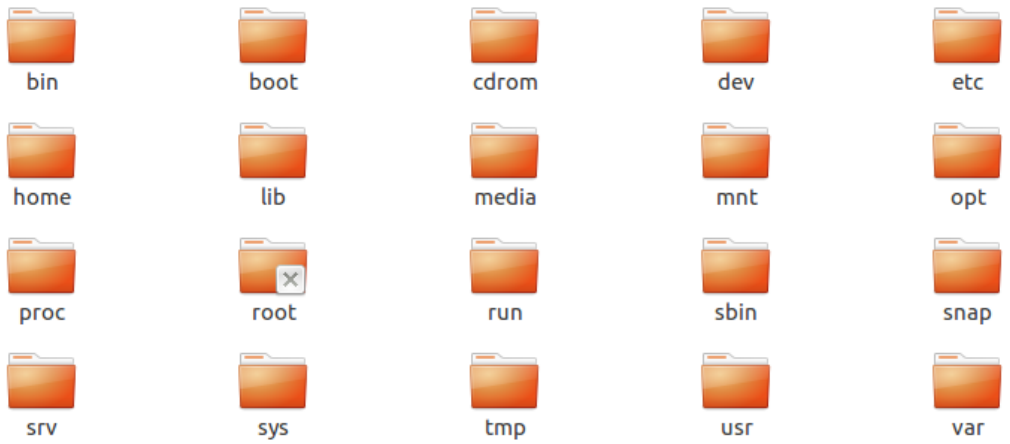
2. temps de processeur
3. accès aux entrées sorties

De nos jours l'interface d'utilisation d'un ordinateur est essentiellement graphique ainsi lancer un programme revient à cliquer sur une icône ou toucher du doigt une icône.

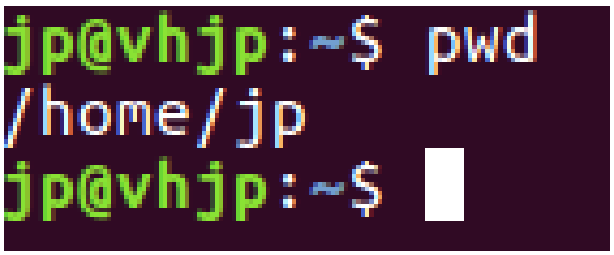
Cependant il existe encore un terminal où l'on peut exécuter un programme appelé **shell** ou **shell bash** constitué de commandes permettant à l'utilisateur d'interagir avec le système d'exploitation et de lancer des programmes

3 Découverte du système de gestion de fichiers d'Unix et du shell d'Unix

Le système de gestion de fichiers d'Unix, structure la mémoire du disque dur (attention ! ce n'est pas la mémoire vive) sous la forme d'un arbre dont les noeuds sont soit des répertoires soit des fichiers
Un répertoire peut contenir d'autres répertoires ou des fichiers.
Voici la disposition des répertoires pour le système Ubuntu (Linux)



1. Quelque soit le système UNIX que vous avez sous la main ouvrir un **terminal**
2. Le terminal nous permettra d'entrer des commandes du langage interprété **shell** et entrez la commande `pwd` pour `print working directory` qui affiche le **chemin absolu du répertoire actuel** dans lequel se trouve l'utilisateur qui a ouvert une session avec son mot de passe
   Vous devez observer ceci



   Il nous faut préciser ici la notion de chemin absolu et de chemin relatif, que nous avons déjà vu lors du projet Eisbar

Un fichier ou un répertoire est **repéré absolument par rapport à la racine** /, ainsi le répertoire jp dans l'exemple ci-dessus a pour chemin absolu

/home/jp

le fichier reduction.py a pour chemin absolu :

/home/jp/Documents/reduction.py

Puisqu'il se trouve dans le répertoire Documents qui est inclus dans le répertoire jp qui est contenu dans le répertoire home qui est un "enfant" de la racine

L'inconvénient du repérage absolu est sa longueur, on lui préfère en général le repérage relatif

Qui dit relatif dit relatif par rapport à un autre fichier ou à un autre répertoire, regardons cela sur l'exemple suivant

3. Pour se déplacer dans l'arbre on peut soit "remonter" par la commande `cd ..` (pour change directory) les deux points pour signifier "vers le haut"

   Soit "descendre" vers un répertoire précis par exemple en entrant la commande `cd Documents` et

   le répertoire Documents a pour chemin relatif **par rapport à jp**

   Documents

4. Documents et Téléchargements sont deux répertoires "frères" contenus dans jp, pour passer de Documents à Téléchargements je peux entrer la commande avec une adresse relative :

   cd ../Téléchargements

   ou la commande suivante avec une adresse absolue

   cd /home/jp/Téléchargements

5. Vous pouvez faire `ls` pour visualiser le contenu du répertoire Documents

   On observe dans le répertoire Documents un autre répertoire 2019-2020 (les répertoires n'ont pas d'extension) et d'autres fichiers comme pstree.png qui est une image et reduction.py qui est un fichier Python

6. la commande `mkdir` pour make directory permet de créer un répertoire.

   Ainsi l'année prochaine on pourra créer un répertoire 2020-2021 dans le répertoire Documents pour y ranger les fichiers de l'année scolaire 2020-2021

   On peut le faire avec le langage shell en entrant la commande `mkdir 2020-2021` en s'assurant au préalable être placé dans le répertoire Documents

   Créer un répertoire 2020-2021 dans Documents

7. Insérer la clé USB

   Chercher son chemin absolu avec l'interface graphique

8. Pour copier le fichier reduction.py de Documents vers la clé USB il y a deux possibilités

(a) "couper" c'est à dire déplacer le fichier avec la commande `mv <source> <cible>`, mv pour move

En étant déjà dans Documents on va donc entrer `mv reduction.py /media/jp/cle-JP/reduc`

(b) "copier-coller" avec la commande

`cp <source> <cible>`, cp pour copy

En étant déjà dans Documents on va donc entrer

`cp reduction.py /media/jp/cle-JP/reduction.py`

(c) Pour copier **tous** les fichiers ayant la même extension .py du répertoire Documents vers la clé USB on va faire

`cp *.py /media/jp/cle-JP`

9. Pour supprimer un fichier on a la commande `rm` pour remove ainsi si on a fait un copier-coller du fichier reduction.py de Documents vers la clé USB et qu'après coup on veut aussi le supprimer de Documents alors on peut entrer la commande après s'être assuré d'être déjà dans Documents

   `rm reduction.py`

10. Une dernière chose sur le shell lorsqu'on a oublié les spécificités d'une commande on utilise le manuel avec la commande `man`

    Ainsi en tapant `man ls` on aura toutes les options de la commande `ls`

4 Droits des fichiers

Chaque utilisateur de la machine a un identifiant et un mot de passe lui permettant d'utiliser la machine

Chaque utilisateur a un espace personnel situé dans le répertoire `/home` à son nom dans lequel il pourra stocker ses fichiers

Chaque utilisateur de la machine est identifié en plus par un numéro unique `uid` (user identifier) et appartient forcément à un groupe principal et possède aussi un numéro d'identification `gid` (group identifier) ceci servira par la suite à la définition des droits d'accès aux fichiers

Pour connaitre son uid et son gid il faut entrer la commande `id` on observe alors quelque chose comme :

```
MacBook-Pro-de-vallon:~ vallon$ id
uid=501(vallon) gid=20(staff) groups=20(staff),701(com.apple.sharepoint.group.1)
,12(everyone),61(localaccounts),79(_appserverusr),80(admin),81(_appserveradm),98
```

Les droits d'accès pour un fichier ordinaire sont définis relativement à trois publics différents :

1. L'utilisateur propriétaire du fichier
2. Le groupe principal du propriétaire
3. Les autres utilisateurs de la machine

Avec la commande `ls -l` avec l'option -l pour long on a des informations sur les droits d'un fichier

```
MacBook-Pro-de-vallon:2020-2021 vallon$ ls -l par_dessus.png
-rw-r--r--@ 1 vallon  staff  139980 13 avr 10:57 par_dessus.png
```

Ainsi pour l'image nommée `par_dessus.png` on a les informations suivantes
`-rw-r-r-@ 1 vallon staff 139180 13 avr 10:57 par_dessus.png`
Ce qui signifie que

1. le premier tiret - signifie que c'est un fichier ordinaire si c'était un répertoire il y aurait eu d comme directory
2. Ensuite les trois premiers symboles `rw-` signifient que pour le propriétaire de cette image, il peut lire (read) et écrire (write)(modifier) cette image, le tiret est à la place de x (pour executable) donc cette image ne peut être un exécutable (un script ou un programme)
3. Les trois symboles suivants `r-` signifient que les membres du groupe principal du propriétaire, appelé `staff` ne peuvent que visualiser l'image sans la modifier, ni l'exécuter
4. De même pour les autres utilisateurs de la machine

Pour un répertoire les droits sont :

1. les droits de lecture = on peut lister les fichiers contenus dans le répertoire
2. les droits d'écriture = on peut ajouter ou supprimer des fichiers dans le répertoire et renommer des fichiers
3. les droits d'exécution = l'utilisateur peut se positionner dans ce répertoire avec la commande `cd` par exemple

Le propriétaire d'un fichier peut changer les droits des fichiers avec la commande `chmod` (pour change mode)
La syntaxe générale est la suivante `chmod public opération droit nom_du_fichier`
Où :

1. public est `u` pour l'utilisateur, `g` pour le groupe principal et `o` pour les autres
2. opération est + pour ajouter le droit,- pour l'enlever et = pour affecter un droit
3. droit est r,w,x

Par exemple on peut donner aux utilisateurs du groupe `staff` le droit de modifier l'image `par_dessus.png`

```
MacBook-Pro-de-vallon:2020-2021 vallon$ chmod g+w par_dessus.png
MacBook-Pro-de-vallon:2020-2021 vallon$ ls -l par_dessus.png
-rw-rw-r--@ 1 vallon  staff  139980 13 avr 10:57 par_dessus.png
```

5 Exercices

Question 1 Quel est l'effet de la commande shell suivante
$cd ..

☐ Ejecter le cd

☐ Changer le répertoire courant vers le répertoire supérieur
☐ Supprimer le répertoire courant
☐ Copier le répertoire courant

Question 2 Que peut on dire du système de fichiers suite à l'exécution des commandes suivantes ?

$ls
`entier.py float.py chat.png`
$mkdir X
$mv *.py X

☐ Le répertoire X contient deux fichiers d'extension .py
☐ Le répertoire contient les résultats de l'exécution des fichiers `entier.py` et `float.py`
☐ Les fichiers `entier.py float.py` et X ont été déplacés dans le répertoire de l'utilisateur
☐ L'utilisateur X est propriétaire des fichiers `entier.py float.py`

Question 3 Une seule des phrases suivantes est fausse

☐ Le premier système d'exploitation est apparu en même temps que les premiers ordinateurs
☐ Linux, Android,iOs et mac Os sont des descendants de Unix
☐ Unix a été écrit en C
☐ Le shell est un langage interprété comme Python