

# Système d'Exploitation (SE) -Linux-

Dr. Mohammed BOUGAA  
mbougaa@itescia.fr

ITESCIA\_2018/2019: Système d'Exploitation –Linux- [M2I L3]



## Plan du cours

- ❖ C'est quoi un Système d'Exploitation
- ❖ Linux
  - Présentation
  - Installation et prise en main
- ❖ Commandes
- ❖ Manipulation de Fichiers

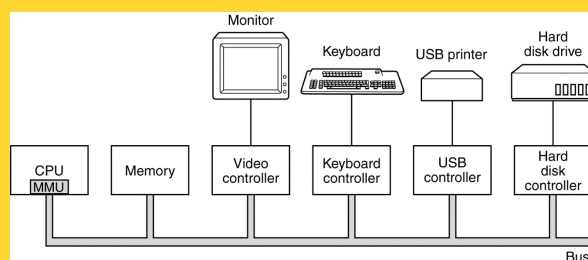
## Partie 1: Système d'Exploitation

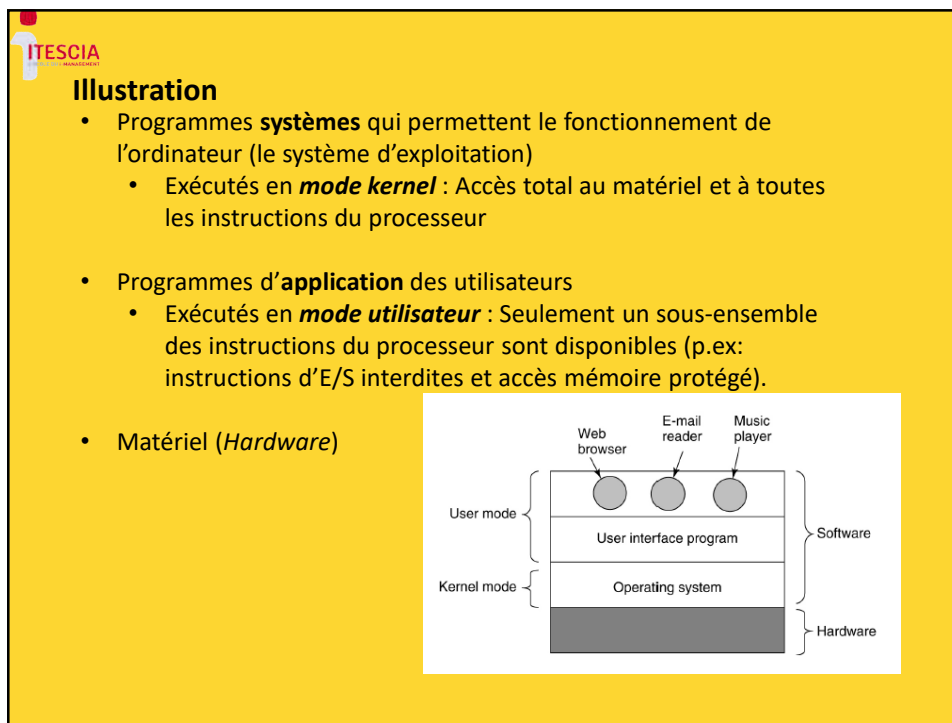
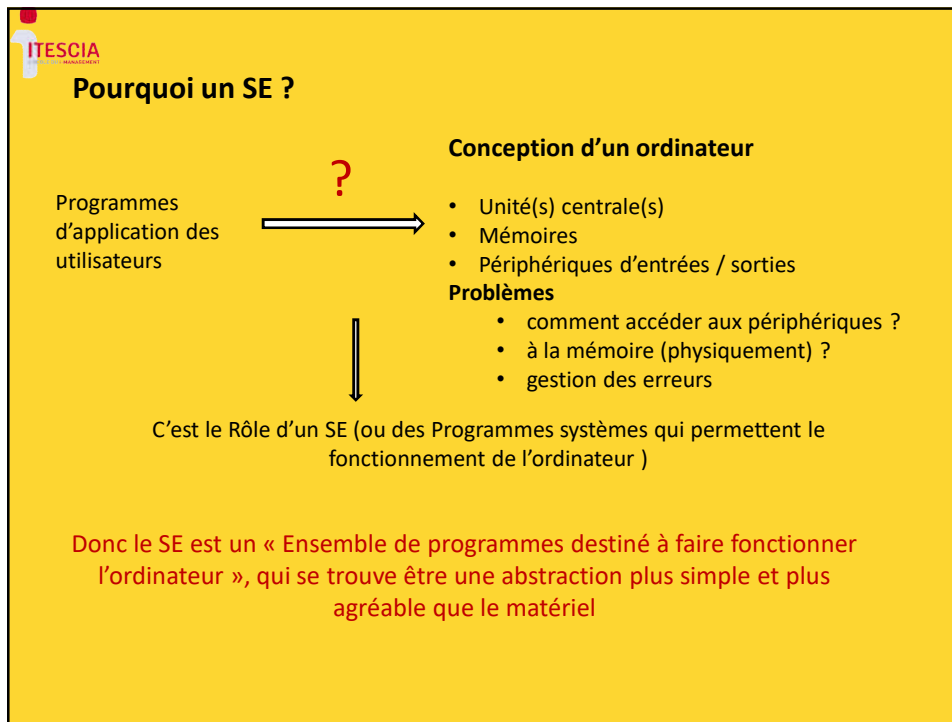



### C'est quoi un SE ?

#### Composants de l'ordinateur moderne :

- Un ou plusieurs microprocesseurs
- Mémoires
- Disques
- Clavier
- Souris
- Displays (écran)
- Interfaces réseau
- D'autres périphériques d'entrées / sorties
- Etc.







**Principales fonctions (Concepts de base pour l'utilisation d'un SE):**

**Allocateur et gestionnaire de ressources**


- gestion de entrées-sorties
- gestion des processus (charger, exécuter, terminer) → ordonne et contrôle l'allocation des processeurs
- gestion de la mémoire centrale

**Fournit également :**

- Système de Gestion des Fichiers SGF
- Briques et protocoles pour la gestion du Réseau et de la Sécurité → protège les utilisateurs dans le cas d'usage partagé
- Interface utilisateur, ... etc


**Deux grands types de SE**

- systèmes constructeurs
- systèmes ouverts




**Partie 2:**

**Linux**



### C'est quoi Linux:

- Un SE 32/64-Bit de type Unix
- Utilitaires Unix comme sed, awk, et grep
- Compilateurs C, C++, Fortran, Small talk, Ada, ....
- Utilitaires réseaux comme telnet, ftp, ssh, ping, traceroute
- Gère Multi-processeurs
- X Windows GUI
- Interopérabilité avec d'autres SE
- Fonctionne sur différentes machines
- Code source disponible –Logiciel libre



### C'est quoi Linux:

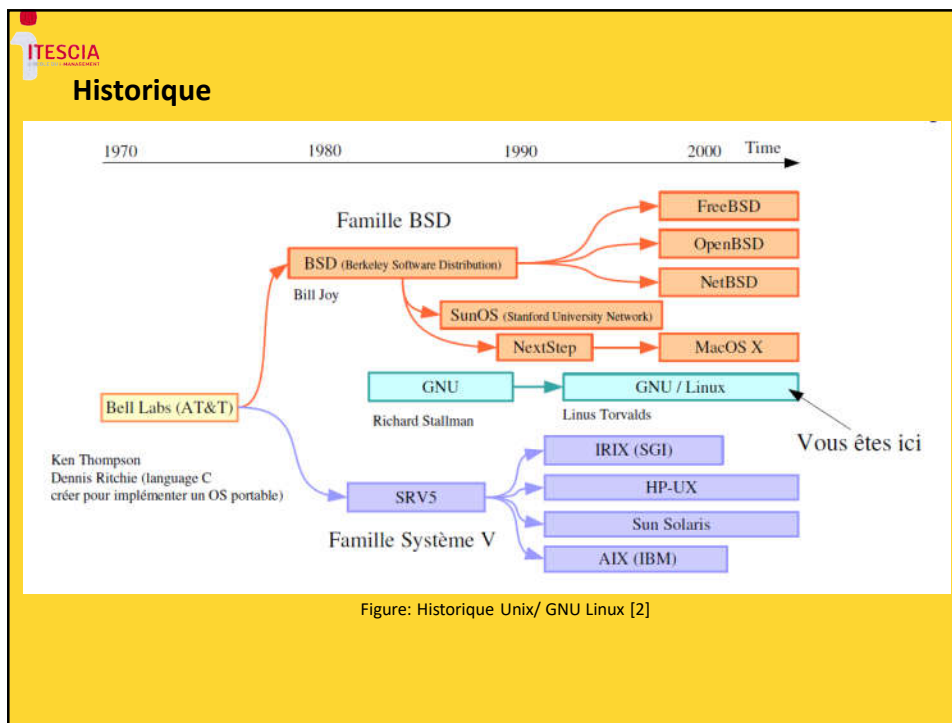
Unix est un système d'exploitation permettant de contrôler un PC et ses différents périphériques. Unix se distingue par les caractéristiques suivantes :

- Multi-utilisateurs (qui peut être utilisé simultanément par plusieurs personnes)
- Multitâches (un utilisateur peut exécuter plusieurs programmes en même temps)
- Repose sur un noyau (kernel) utilisant 4 concepts principaux : **fichiers, droits d'accès, processus et communication interprocessus (IPC)**

**ITESCIA**

## Historique

- 1969 Unix –Laboratoires Bell
  - K. Thompson –PDP7 Sep
- 1973 réécriture en C du système
  - D. Ritchie
- 1977 Transport InterData32 –
- 1977 UnixV5 –BSD
- 1980-> UnixV7, BSD4.3, Ultrix, SunOs, AIX-IBM, FreeeBSD, ...
- **1991 Linux**
  - Linus Torvalds
  - Avec l'aide de programmeurs du monde entier
  - 1er version postée sur Internet en 1991
- 1994 Linux 1.0 et en 2003 Linux 2.6
- Slackware, Debian, Ubuntu, Gentoo, etc.
- LiveCD: Knoppix
- Aujourd'hui, utilisésur7 à 10 millions d'ordinateurs, mais MS 98% de parts de marché






### **Linux est un Logiciel Open Source:**

- Tout le monde peut prendre le logiciel sur internet pour le lire, le modifier, le corriger, l'adapter et le redistribuer.
- Processus collaboratif à l'échelle mondiale,

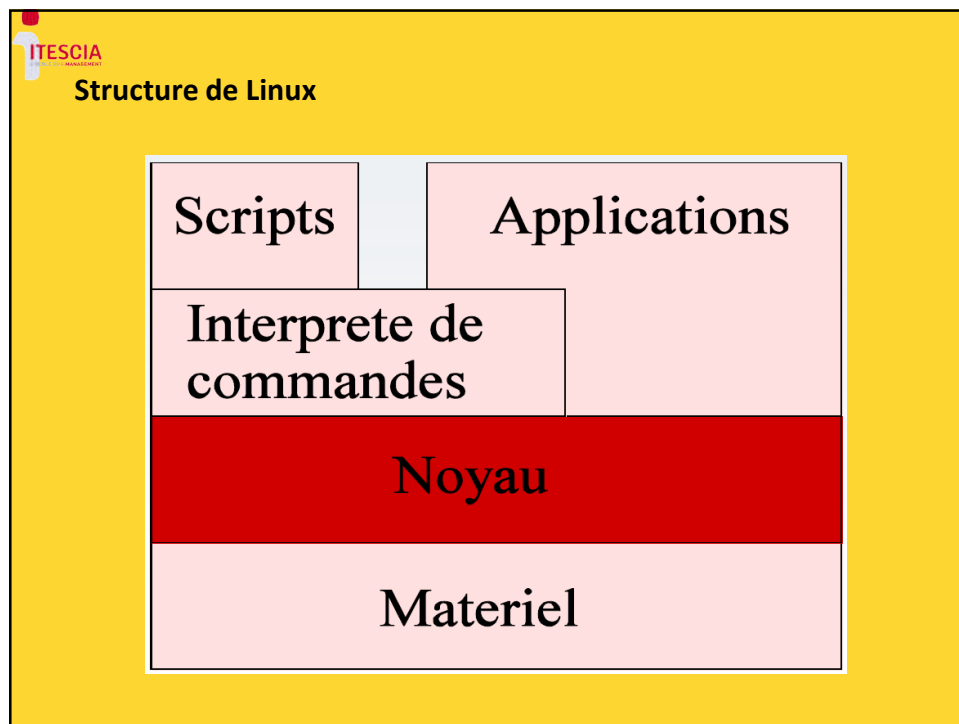
### **Où l'obtenir ?**

- Le plus simple par Internet sur le site d'un distributeur :
  - <http://www.debian.org/index.fr.html>
  - <http://www.ubuntu-fr.org/>
- Télécharger les images iso



### **Points forts:**


- Puissance
  - Tourne sur de nombreux ordinateurs différents
  - Rapide et stable
  - Très nombreux logiciels
  - Pilotes pour une majorité de périphériques
- Libre et Gratuit (Licence GPL)



Partie 3:  
TP: Installation et prise en main de Linux

The slide features a yellow background with the ITESCIA logo in the top right corner. A red banner with white text is centered on the slide, indicating the topic of the third part of the course.






## Machine Virtuelle

❖ C'est quoi une VM?

Une **machine virtuelle** (anglais virtual machine. VM) est une **illusion d'un appareil informatique** créée par un logiciel d'émulation. Le logiciel d'émulation **simule** la présence de ressources matérielles et logicielles telles que la mémoire, le processeur, le disque dur, **voire le système d'exploitation** et les pilotes, permettant d'exécuter des programmes dans les mêmes conditions que celles de la machine simulée. [Wikipedia]

**L'émulation** est le fait de mettre en œuvre les fonctionnalités d'un appareil en utilisant un autre appareil offrant des fonctionnalités différentes.




## Machine Virtuelle

❖ **Intérêts des Machines Virtuelles**

- Pouvoir s'abstraire des caractéristiques de la machine physique utilisée
- Isoler des applications pour des raisons de sécurité,
- Augmenter la robustesse d'un serveur en limitant l'impact des erreurs système
- Émuler plusieurs machines sur une seule machine physique (virtualisation).
- Permettre des tests de variantes d'installation sur des machines simulées possédant de tailles de RAM, de mémoire graphique, et des nombres de processeurs divers.

❖ **Inconvénients des Machines Virtuelles**

- Performances brutes sensiblement inférieures à une exécution sur le matériel en natif
- Consommer une petite partie de la mémoire réelle pour leur propre fonctionnement



## Machine Virtuelle


- ❖ Installation de « **Virtual Box** »

**VirtualBox** est un logiciel de virtualisation de systèmes d'exploitation. En utilisant les **ressources matérielles de votre ordinateur** (*système hôte*), VirtualBox permet la création d'un ou de plusieurs ordinateurs virtuels (*machines virtuelles*) dans lesquels s'installent d'autres systèmes d'exploitation (*systèmes invités*).

Les *systèmes invités* fonctionnent **en même temps** que le *système hôte*, mais **seul** ce dernier a accès directement au véritable **matériel** de l'ordinateur. Les *systèmes invités* exploitent du matériel générique, simulé par un « faux ordinateur » (*machine virtuelle*) créé par VirtualBox.

VirtualBox permet de faire fonctionner un ou plusieurs système(s) d'exploitation en même temps en toute **sécurité**. En effet, les *systèmes invités* **n'interagissent pas directement avec le système hôte**, et n'interagissent pas entre eux. Le champ d'action des *systèmes invités* est confiné, **limité à leur propre machine virtuelle**.

[doc.ubuntu-fr.org]



## Machine Virtuelle

- ❖ Télécharger la dernière version de « **Virtual Box** », pour votre machine  
**<https://www.virtualbox.org/wiki/Downloads>**
- ❖ Installer la « **Virtual Box** »
- ❖ Créer une nouvelle Machine Virtuelle sur « **Virtual Box** » ayant la configuration suivante:
  - 10 GB de disque dur
  - 2 GB RAM

– Dans l'unité DVD, mettez l'image iso d'ubuntu 64 bits server.


**ITESCIA**  
MANAGEMENT

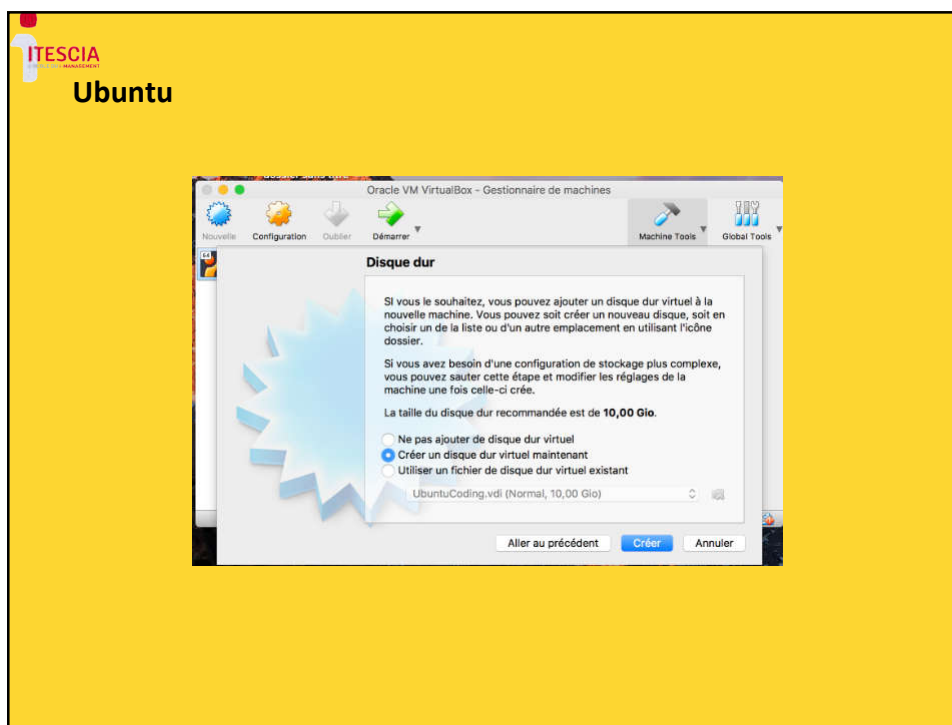
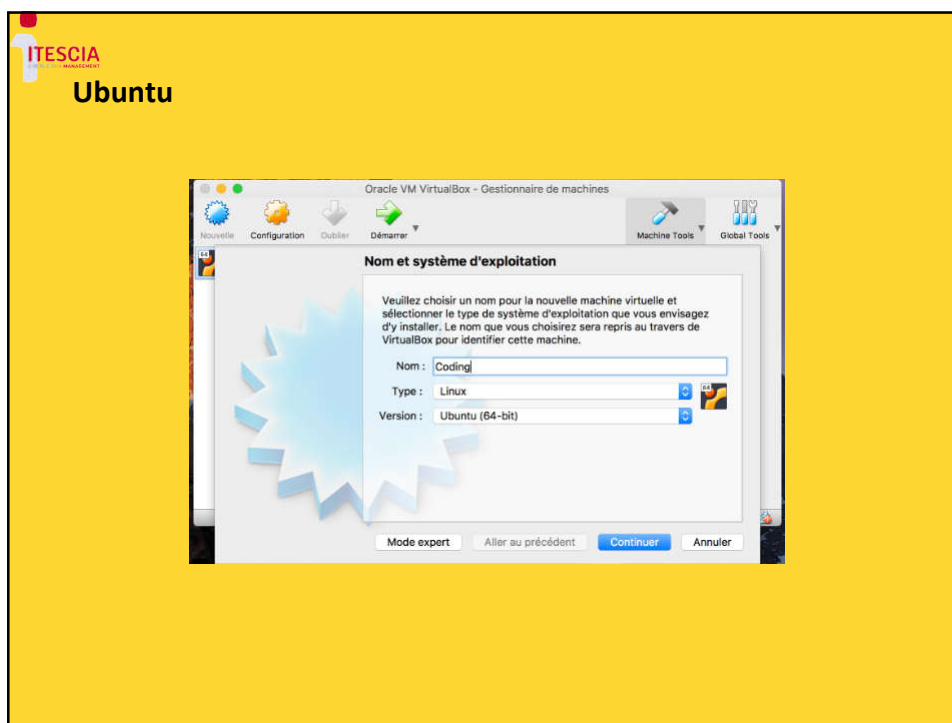
## Ubuntu

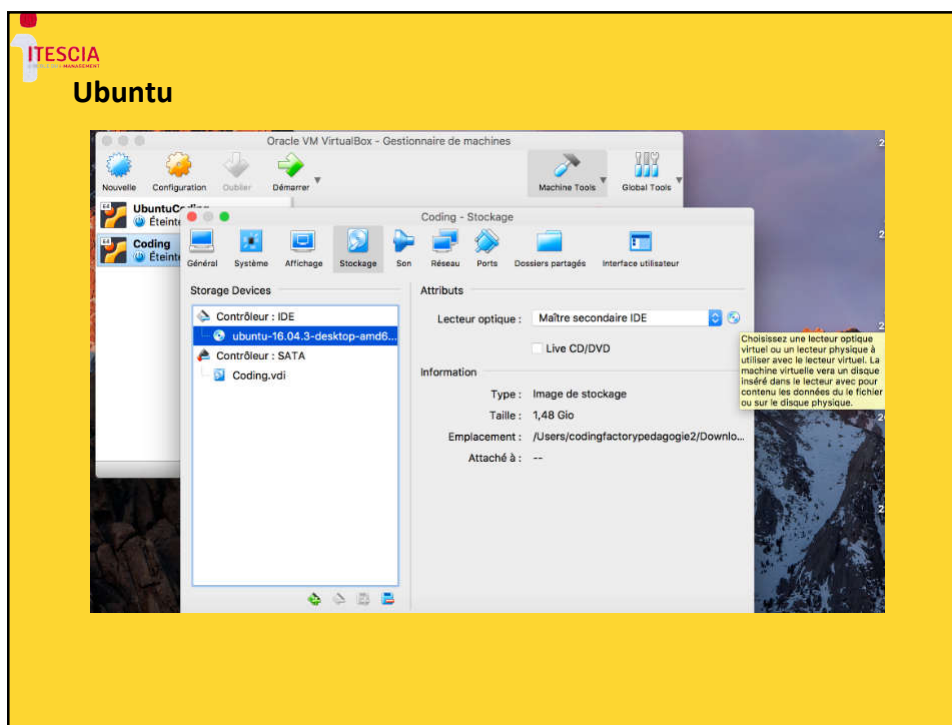
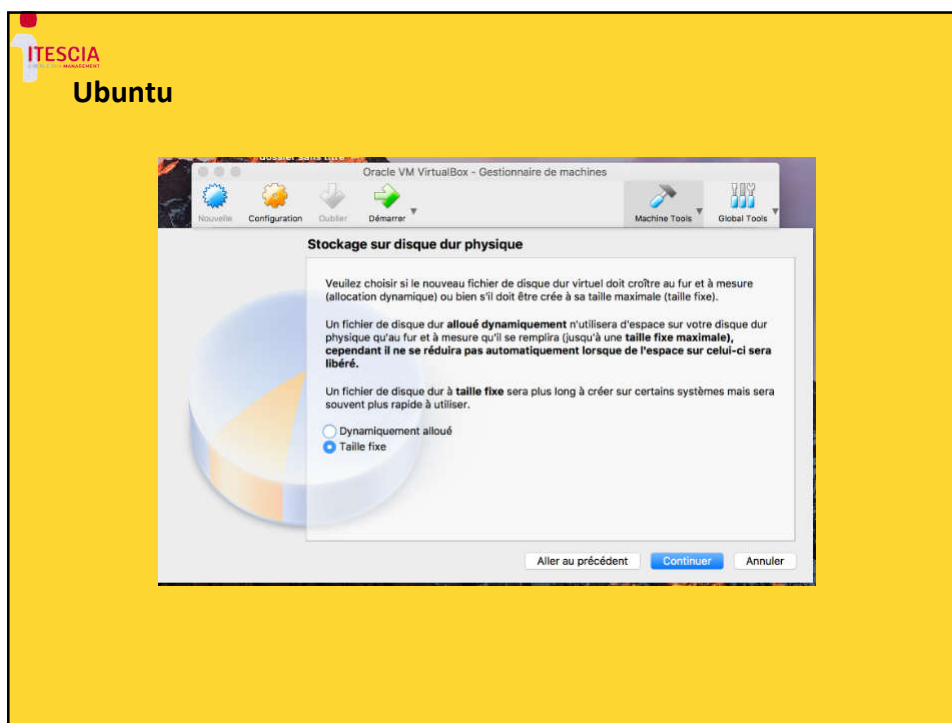
- ❖ Télécharger l'image ISO de la dernière version stable du **Ubuntu (Version 16.04.3 LTS)**  
<https://www.ubuntu.com/download/desktop>
- ❖ Installer Ubuntu sur la machine virtuelle que vous avez créé précédemment:
  - Dans l'unité DVD, mettez l'iso d'Ubuntu.
  - Démarrez la machine virtuelle et suivez les instructions d'installation d'Ubuntu de sorte qu'on arrive à l'écran de partitionnement du disque dur. Pour cela, il faudra d'abord configurer les paramètres suivants :
  - Langue française
  - Pays : France
  - Clavier français
  - Heure de Paris

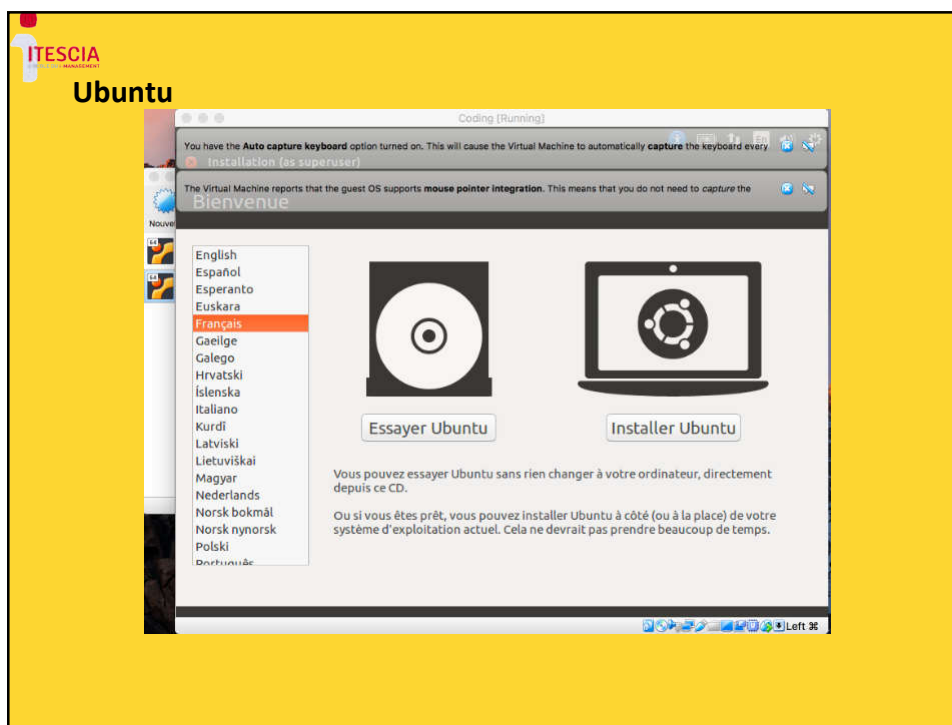
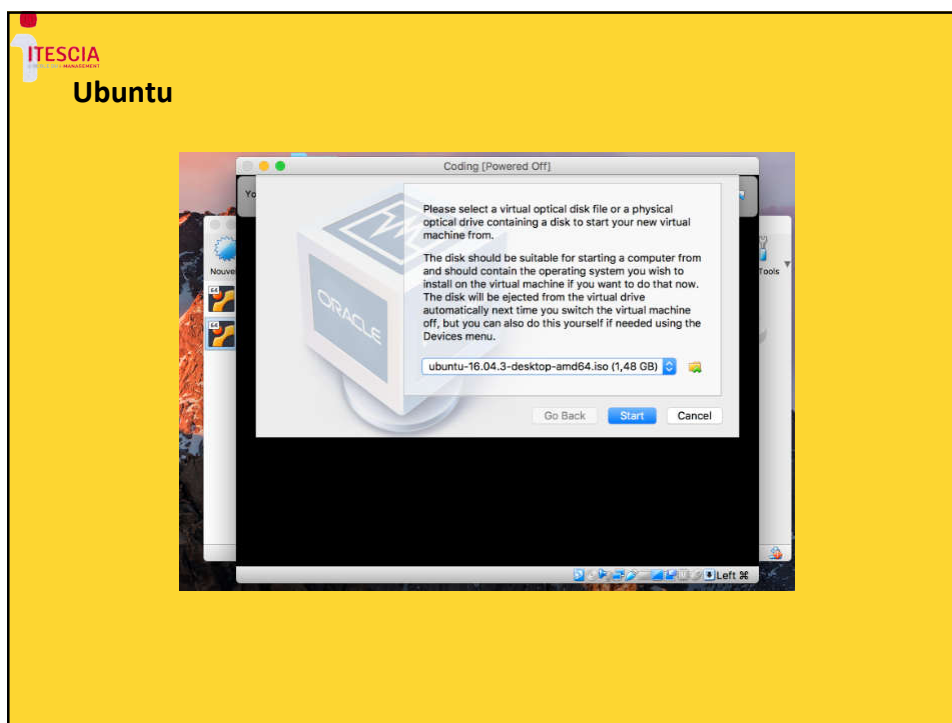
**ITESCIA**  
MANAGEMENT

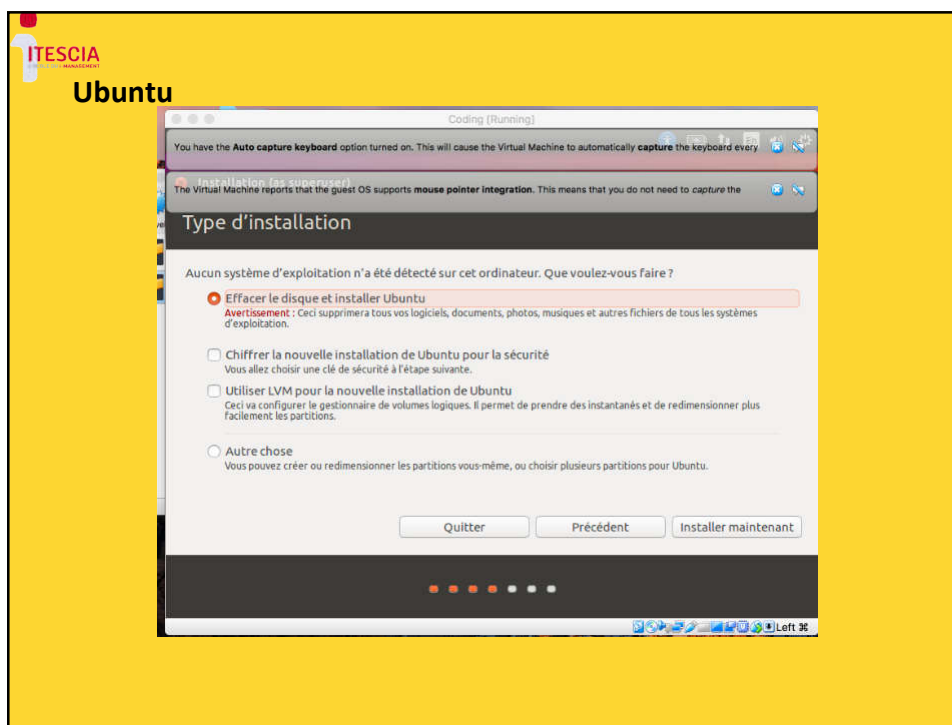
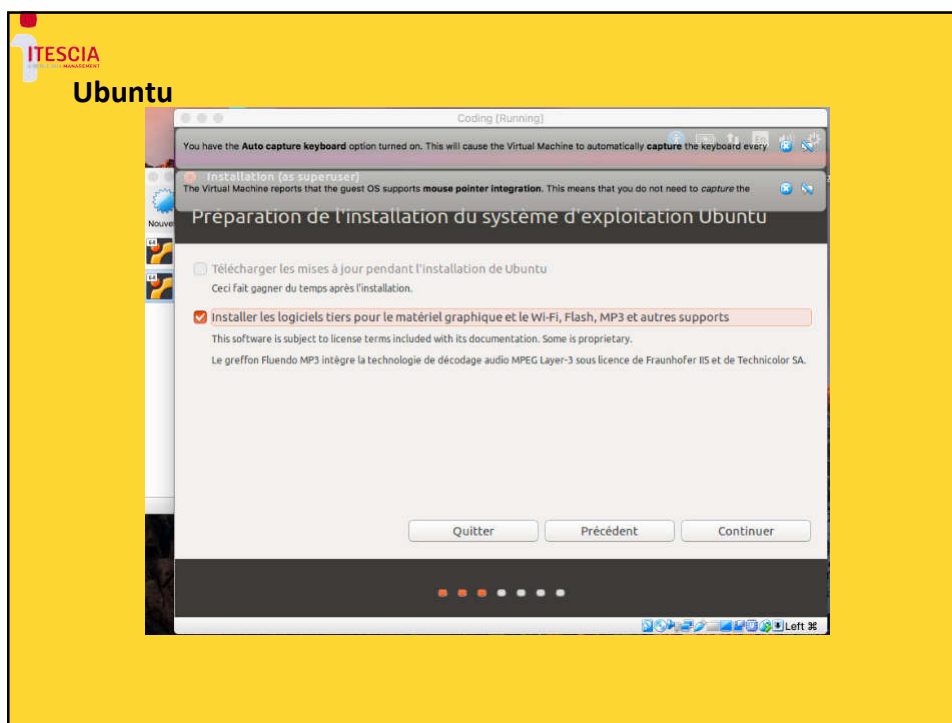
## Ubuntu





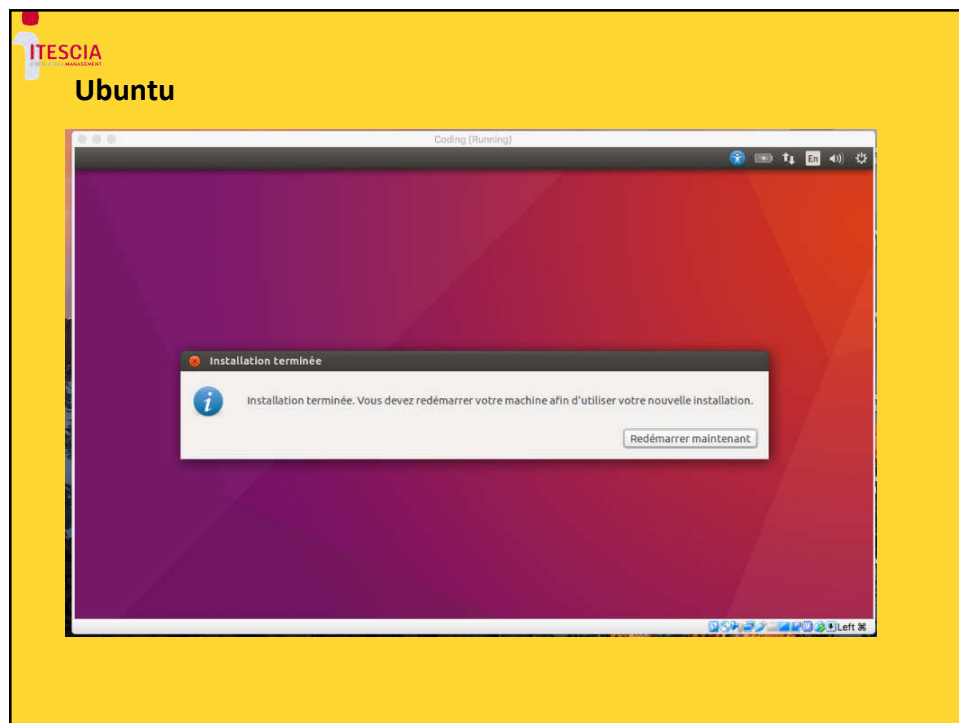
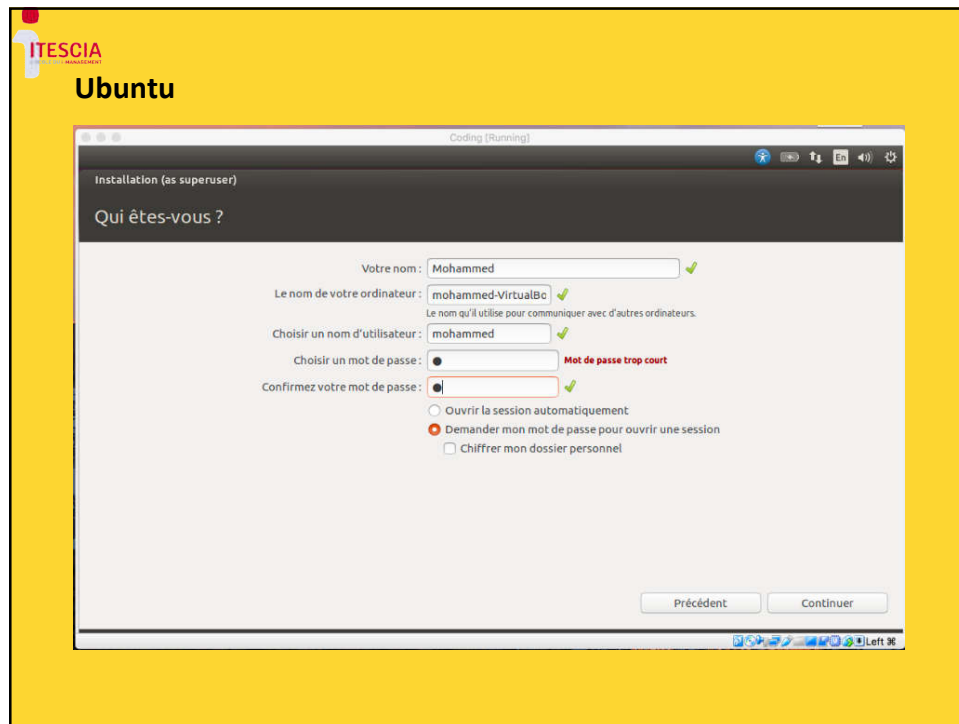












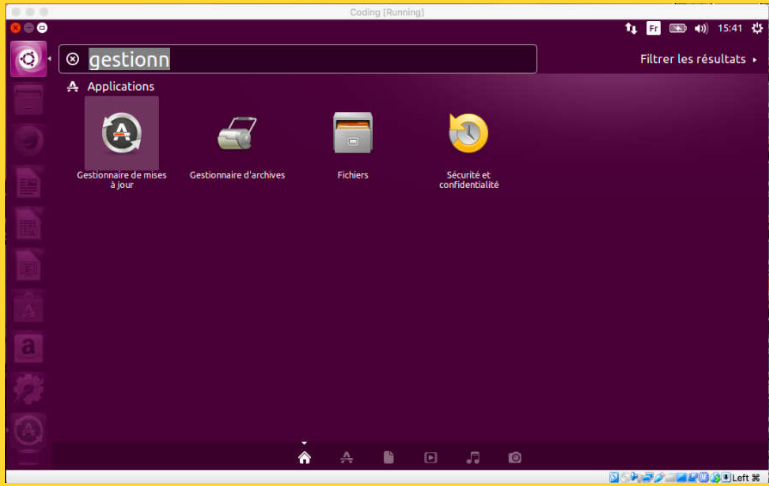
**ITESCIA**  
INFORMATION TECHNOLOGY MANAGEMENT

## Ubuntu\_Mise à jour du système (Via l'Interface graphique du gestionnaire de mises à jour)

- Mise à jours système
- Installation de paquets à partir des dépôts officiels
- Installation à partir de paquets non présent dans les dépôts
- Installation à partir de code source

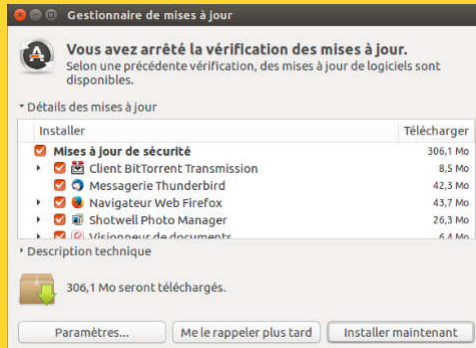
**ITESCIA**  
INFORMATION TECHNOLOGY MANAGEMENT

## Ubuntu\_Mise à jour du système (Via l'Interface graphique du gestionnaire de mises à jour)



**ITESCIA**

### Ubuntu\_Mise à jour du système (Via l'Interface graphique du gestionnaire de mises à jour)



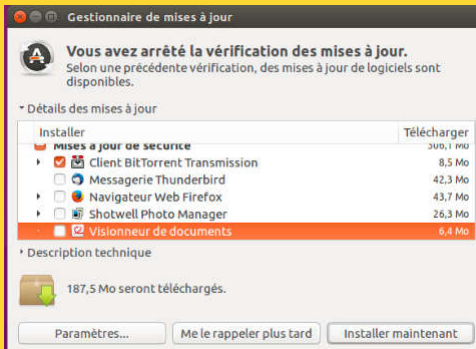
Logiciel	Télécharger
<input checked="" type="checkbox"/> Mises à jour de sécurité	306,1 Mo
<input checked="" type="checkbox"/> Client BitTorrent Transmission	8,5 Mo
<input checked="" type="checkbox"/> Messagerie Thunderbird	42,3 Mo
<input checked="" type="checkbox"/> Navigateur Web Firefox	43,7 Mo
<input checked="" type="checkbox"/> Shotwell Photo Manager	26,3 Mo
<input checked="" type="checkbox"/> Visionneur de documents	6,4 Mo

306,1 Mo seront téléchargés.

Paramètres... Me le rappeler plus tard Installer maintenant

**ITESCIA**

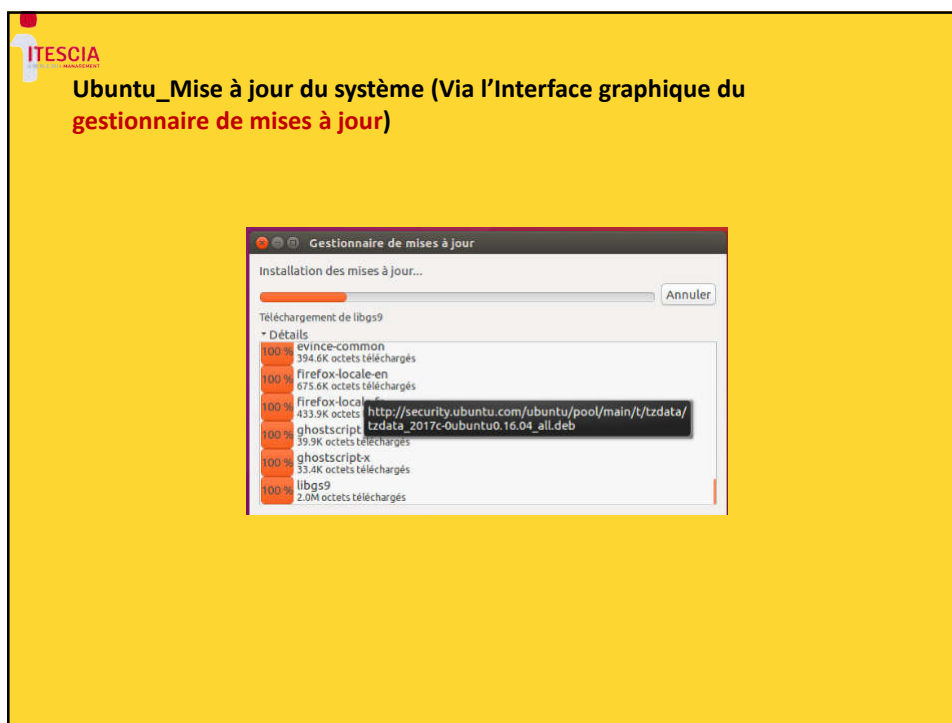
### Ubuntu\_Mise à jour du système (Via l'Interface graphique du gestionnaire de mises à jour)



Logiciel	Télécharger
<input type="checkbox"/> Mises à jour de sécurité	306,1 Mo
<input checked="" type="checkbox"/> Client BitTorrent Transmission	8,5 Mo
<input type="checkbox"/> Messagerie Thunderbird	42,3 Mo
<input type="checkbox"/> Navigateur Web Firefox	43,7 Mo
<input type="checkbox"/> Shotwell Photo Manager	26,3 Mo
<input checked="" type="checkbox"/> Visionneur de documents	6,4 Mo

187,5 Mo seront téléchargés.

Paramètres... Me le rappeler plus tard Installer maintenant



**ITESCIA**


## Ubuntu\_Mise à jour du système

### 1.2 Via un terminal

Pour maintenir à jour son système depuis un **terminal**, on peut utiliser la **commande**:

```
sudo apt-get update && sudo apt-get upgrade -y
```

Voir la page [apt-get mise à jour des dépôts](#) pour des informations complémentaires.

**ITESCIA**  
Informatique - Télécoms - Services - Cloud

# Ubuntu

## 4.1 Mise à jour dépôts

```
sudo apt-get update
```

- L'option **update** met à jour la liste des fichiers disponibles dans les dépôts APT présents dans le fichier de configuration `/etc/apt/sources.list`. L'exécuter régulièrement est une bonne pratique, afin de maintenir à jour votre liste de paquets disponibles.

Modifier

## 4.2 Mise à jour de paquets

```
sudo apt-get install <paquet(s)> -v
```

- L'option **install** met à jour les paquets indiqués déjà installés, vers leurs dernières versions (rarement utilisé).

```
apt-get --simulate upgrade
```

- L'option **simulate** simule la mise à jour des paquets sans réellement les installer (rarement utilisé).

```
sudo apt-get upgrade
```

- L'option **upgrade** met à jour tous les paquets installés sur le système vers les dernières versions (couramment utilisé).


```
sudo apt-get dist-upgrade
```

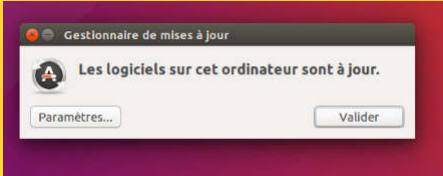
- L'option **dist-upgrade** met à jour tous les paquets installés vers les dernières versions en installant de nouveaux paquets si nécessaire, par opposition à l'upgrade simple qui n'ajoute pas de nouveaux paquets.


Remarque : taper "man apt-get" dans un terminal donne une réponse en français et une explication plus complète et plus claire.

Modifier

A screenshot of a terminal window titled "mohammed@mohammed-VirtualBox ~". The terminal displays the output of a system update command. It lists several packages being downloaded from the Ubuntu archive, including bdpkg-perl, bpopplers, libgail, and libgtk2.0-bin, along with their sizes and download progress. The output also shows the installation of various updates for the xentia package. At the bottom, it indicates that 62% of the updates are installed, totaling 103 files and 15.3 MB, with a remaining size of 43.7 MB and 35% space used.

 **Ubuntu\_Mise à jour du système (Via l'Interface graphique du gestionnaire de mises à jour)**



 **Pour une:**

- Installation de paquets à partir des dépôts "officiels"
- Installation à partir de paquets non présent dans les dépôts
- Installation à partir de code source

**Regarder les procédures sur:**

[https://doc.ubuntu-fr.org/tutoriel/comment\\_installer\\_un\\_paquet](https://doc.ubuntu-fr.org/tutoriel/comment_installer_un_paquet)




## Partie 4: Système de Fichiers

### Le système de fichiers

Presque tous dans Unix est un fichier!

▶ Fichiers ordinaires	▶ Périphériques et dispositifs La lecture et l'écriture à partir d'un dispositif se fait comme un fichier
▶ Répertoires Les répertoires ne sont juste que des fichiers listant plusieurs fichiers	▶ Pipes Utiliser pour mettre en cascade plusieurs programmes <code>cat *.log   grep error</code>
▶ Liens symboliques Fichiers faisant référence au nom d'un autre fichier	▶ Sockets Communication inter processus




## Manipulation des fichiers

Stockage persistant, non volatile. Un système de fichiers est **une structure de données** permettant de **stocker les informations** et de les organiser dans des fichiers sur des **mémoires secondaires**.

Les fichiers sont gérés par le système d'exploitation. C.à.d:  
La manière dont ils sont

- structurés
- nommés
- utilisés
- protégés

**est à la charge du SE.**



## Manipulation des fichiers

Depuis le début d'Unix, les noms de fichiers ont les caractéristiques suivantes:

- ▶ Sensibles aux majuscules / minuscules
- ▶ Pas de longueur limite évidente
- ▶ Peuvent contenir tous caractères (incluant l'espace, à l'exception de /).  
Les types de fichiers sont stockés dans un fichier ("nombre magique").  
Les extensions d'un nom de fichier n'ont pas besoin et ne sont pas interprétés. Ils sont justes utilisés pour les utilisateurs .
- ▶ Exemples de noms de fichiers:

README	.bashrc	Windows Buglist
index.htm	index.html	index.html.old



**Manipulation des fichiers**

- Le SE Linux masque les spécificités des disques.
- Offre des **Appels systèmes** : création, suppression, lecture écriture, ouverture, fermeture.
- Regroupe les fichiers en hiérarchie arborescente (répertoires + fichiers).
- Assure la protection des fichiers.

```


graph TD
    Root[Root directory] --> Students[Students]
    Root --> Faculty[Faculty]
    Students --> Robert[Robert]
    Students --> Matty[Matty]
    Students --> Leo[Leo]
    Faculty --> ProfBrown[Prof.Brown]
    Faculty --> ProfGreen[Prof.Green]
    Faculty --> ProfWhite[Prof.White]
    ProfBrown --> Courses[Courses]
    ProfBrown --> Papers[Papers]
    ProfBrown --> Grants[Grants]
    ProfBrown --> Committees[Committees]
    Courses --> CS101[CS101]
    Courses --> CS105[CS105]
    Papers --> Files[Files]
    Grants --> SOSP[SOSP]
    Committees --> COST11[COST-11]
  
```

**Manipulation des fichiers**

- Montage d'un système de fichiers:
  - (a) Avant montage les fichiers du CD ne sont pas accessibles.
  - (b) Après montage, ils font partie de la hiérarchie des fichiers.

```

graph TD
    subgraph (a)
        Root1[Root] --> a1[a]
        Root1 --> b1[b]
        a1 --> c1[c]
        a1 --> d1[d]
        CDROM1[CD-ROM] --> x1[x]
        CDROM1 --> y1[y]
    end
    subgraph (b)
        Root2[Root] --> a2[a]
        Root2 --> b2[b]
        a2 --> c2[c]
        a2 --> d2[d]
        b2 --> x2[x]
        b2 --> y2[y]
    end
  
```



## L'arborescence de fichiers


- Le système de fichier correspond à une arborescence que l'on parcourt de la racine (root) vers les feuilles
- La racine se note / (slash)
- Il s'agit d'un répertoire contenant les sous-répertoires suivants :

**/bin** exécutables essentiels pour le système, directement utilisable par les utilisateurs

**/boot** contient les fichiers permettant à Linux de démarrer

**/dev** contient les points d'entrée des périphériques (=device)

**/etc** configuration du réseau, contient les commandes et les fichiers nécessaires à l'administrateur du système (fichiers passwd, group, inittab, ld.so.conf, lilo.conf, ...)



## L'arborescence de fichiers

Sous-répertoires de la racine (suite) :

**/home** répertoire personnel des utilisateurs

**/lib** contient des bibliothèques partagées essentielles au système lors du démarrage et pour les commandes dans /bin

**/mnt** contient les points de montage des partitions temporaires (cd-rom, disquette, ...), parfois nommé media

**/opt** contient des packages d'applications supplémentaires

**/proc** fichiers contiennent des infos sur la mémoire, E/S, périphérique, compatibilité pour le noyau, ...

**/root** répertoire de l'administrateur root

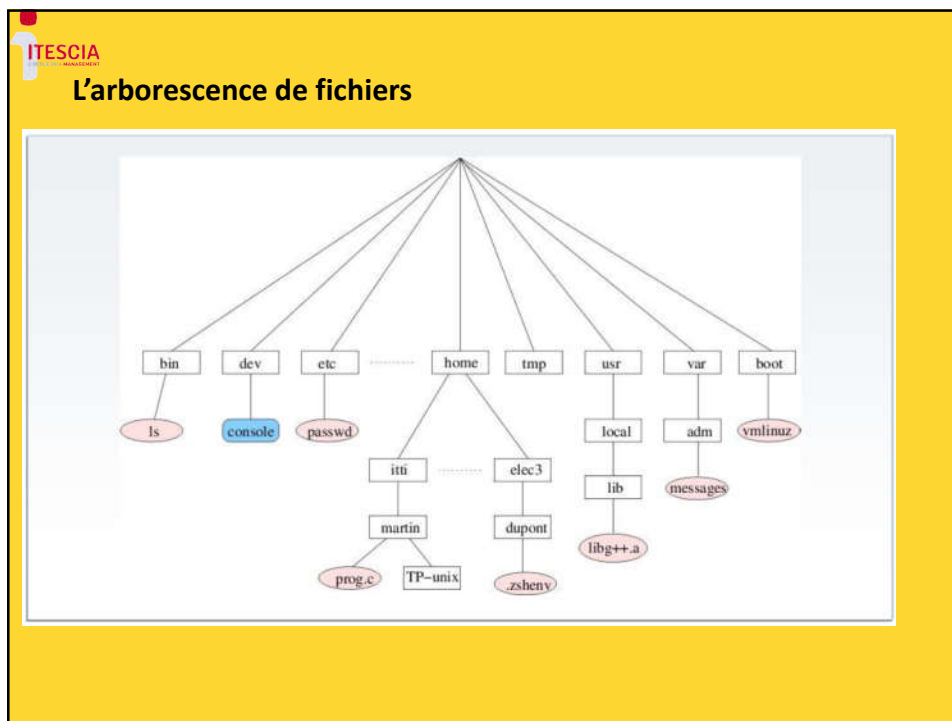
**/usr** hiérarchie secondaire (utilisateurs)

**/var** contient des données diverses (variables), telles que la boîte mail, des fichiers temporaires et des fichiers journaux

**/tmp** contient les fichiers temporaires

**/usr/bin** : commandes complémentaires de l'utilisateur

**/usr/include** : fichiers d'en-tête des langages



**ITESCIA**

### Chemins d'accès

**Notation absolue**

- /usr/include/sys/home/elec3/dupont

**Notation relative**

- prog.c, adm/messages
- ./lib, ../../elec3

**Nom spéciaux**

- /La racine
- .Le répertoire courant
- .. Le répertoire père
- ~Le répertoire utilisateur(home)




## Partie 5: Les Commandes



### L'interpréteur de commande

- Shell : interface entre l'utilisateur et le système d'exploitation ("coquille")
- Application (fichier exécutable) chargée d'interpréter les commandes des utilisateurs et de les transmettre au système
- Différents types de shell, les principaux étant :
  - **sh** (Bourne shell)
  - **bash** (Bourne again shell)
  - **csh** (C shell)
  - **Tcsh** (Tenex C shell)
  - **ksh** Korn shell
  - **zsh** Zero shell
- Le nom du shell correspond généralement au nom de l'exécutable :  
% /bin/bash




### Utilisation du shell

- Le shell correspond à une fenêtre présentant un prompt, encore appelé invite de commande. Celle-ci est paramétrable et par défaut en **bash** se compose comme suit :  

```
login@machine$
```

  
(suffixe **\$** → utilisateur normal, suffixe **#** → super utilisateur/ administrateur)  
On saisit les commandes à la suite du prompt

- Pour stopper la commande en cours : **Ctrl-C**
- Pour mettre en attente la commande en cours : **Ctrl-Z**
- Pour terminer l'entrée standard (les éventuelles paramètres données par l'utilisateur via le clavier) : **Ctrl-D**



### Syntaxe des commandes

**nom\_commande [options] [arguments]**

**Exemple:**


**commande ls**

**options :**

- ls -l -a
- ls -la
- ls -l --color (option à plusieurs caractères)

**arguments :** fichier, expression

- grep toto monFichier
- tar -cv -f archive.tar MonRepertoire



### Méta-caractères

- **\*** : suite de caractères
- **?** : un seul caractère
- **[ ]** : un des caractères dans les crochets
- un ensemble : [hg]
- un intervalle : [a-k]


On verra ça en détaille par la suite

**ls \***  
afficher le contenu du répertoire courant

**ls \*.exe**  
afficher tous les fichiers se terminant par .exe


**ls ????**  
afficher tous les fichiers dont le nom est composé de 4 caractères exactement

**ls [ct]\***  
afficher tous les fichiers dont le nom commence par c ou par t




### Aide

- **man** commande  
obtenir le manuel d'une commande
- **info** commande  
obtenir de l'aide (renvoie souvent à man)
- commande **--help**  
afficher une aide succincte (aide mémoire) et liste les arguments qui peuvent être passés à commande



### Manipulation des fichiers : chemin

- **pwd**  
Afficher le répertoire courant  
**Exemple:**  
yannick@nausicaa:~/toto \$ **pwd**  
/home/yannick/toto
- **cd [chemin]**  
Changer le répertoire courant, se déplacer dans l'arborescence  
**sans argument** : retour au répertoire de connexion  
Alias :
  - . : répertoire courant
  - .. : répertoire parent**Exemples :**  
\$ pwd → /home/yannick/toto  
\$ cd .. → /home/yannick/  
\$ cd projet → /home/yannick/projet  
\$ cd /usr/local → /usr/local




### Manipulation des fichiers : Listing

**ls [option] [chemin]**  
Liste le contenu d'un répertoire avec plus ou moins de détails  
Remarques:

- fichier : afficher description
- répertoire : afficher contenu

**Exemples :**  
\$ ls l\* → liste tous les fichiers commençant par l  
\$ ls -l → liste tous les fichiers du répertoire courant, en donnant les attributs des fichiers (droits, taille, etc)  
\$ ls -a → liste tous les fichiers du répertoire courant (y compris les fichiers cachés dont le nom commence par un ".")  
\$ man ls → affiche la page de manuel de la commande ls




## Manipulation des fichiers : Visualisation

**cat** [option] [chemin vers le fichier1, fichier2, etc]  
affiche le contenu d'un fichier

**Exemples :**  
\$ cat .bash\_profile → affiche le contenu du fichier caché **.bash\_profile**  
\$ cat toto > tata → écrit le contenu du fichier toto dans un fichier nommé tata  
(> on verra ça plutard)

**more** [fichier]  
Visualiser le contenu d'un fichier page à page

**less** [fichier]  
Visualiser le contenu d'un fichier dans un flux




## Manipulation des fichiers : Edition

**wc** [option] [chemin vers le fichier]  
Obtenir des statistiques sur le contenu d'un fichier (affiche le nombre de mots / lignes / caractères d'un Fichier)

**Exemples :**  
\$ wc -l toto → affiche le nombre de lignes du fichier toto  
\$ wc -c toto → affiche le nombre de caractères du fichier toto  
\$ ls | wc -l → affiche le nombre de fichiers dans le répertoire courant

**Comment éditer un fichier ?**  
emacs [fichier]  
vim [fichier]  
gedit [fichier]  
... (On verra ça par la suite)






### Manipulation des fichiers : changement de droits

Toute ressource (fichier, répertoire, ...) a :

- un identificateur
- un propriétaire
- un ensemble de droits d'accès (en lecture, en écriture, en exécution) :
  - les droits du propriétaire
  - les droits du groupe auquel appartient le propriétaire
  - les droits des autres utilisateurs
- Root, administrateur




### Manipulation des fichiers : changement de droits

- **chmod** [options] droits fichier1, fichier2, ...  
changer les droits d'accès au fichier.

Les droits peuvent être spécifiés de **deux façons**, avec des lettres ou avec des nombres en Octal.

- Pour les lettres, il existe les opérateurs de changement d'état + et - pour ajouter ou retirer un type de droit aux droits courants, et l'opérateur = pour les écraser.
- Pour l'octal, il faut additionner les nombres pour chaque type de possesseur.

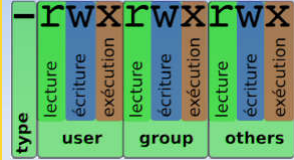


### Manipulation des fichiers : changement de droits (Première Méthode)


- chmod** [options] droits fichier1, fichier2, ...

Changer les droits d'accès au fichier. Les droits sont définis comme suit :

- u** droits de l'utilisateur (user)
- g** droits des utilisateurs du groupe (group)
- a** droits de tous les utilisateurs (all)
- +r** droit en lecture accordé **-r** droit en lecture retiré
- +w** droit en écriture accordé **+x** droit d'exécution accordé

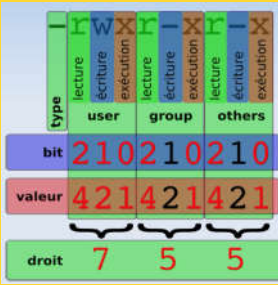


Type	Droit	Destinataire
b	Block device	r Read u User (propriétaire)
c	Character device file	w Write g Groupe
d	Répertoire (Directory)	x Execute (fichier) o Others (les autres, ni 'u' ni 'g')
l	Lien symbolique	x chdir (répertoire)
s	Socket	s SUID bit
p	FIFO	
-	Fichier normal	




### Manipulation des fichiers : changement de droits (Deuxième méthode)

- Dans la deuxième façon de faire, les permissions sont (valeurs octales entre parenthèses): **r (4)** : autorisation de lecture **w (2)** : autorisation d'écriture **x (1)** : autorisation d'exécution




Droit	Valeur alphanumérique	Valeur octale
aucun droit	---	0
exécution seulement	--X	1
écriture seulement	-W-	2
écriture et exécution	-WX	3
lecture seulement	r--	4
lecture et exécution	r-X	5
lecture et écriture	rW-	6
tous les droits (lecture, écriture et exécution)	rwX	7



## Manipulation des fichiers : changement de droits

**Exemples:**

- `chmod a+r fichier` → tout (a) le monde a le droit de lire.
- `chmod 444 fichier` → idem
- `chmod u+x fichier` → l'utilisateur a le droit d'exécuter.
- `chmod 744 fichier` → L'utilisateur a tous les droits, le groupe et les autres ne peuvent que consulter le fichier
- `chmod 755 mon_dossier` → donne au propriétaire tous les droits, aux membres du groupe et aux autres les droits de lecture et d'accès. C'est un droit utilisé traditionnellement sur les répertoires.
- `chmod og-w fichier` → les autres (o) et le groupe (g) n'ont pas le droit d'écrire.




## Manipulation des fichiers : commandes d'administration

**chown** [options] utilisateur.groupe fichier  
change le propriétaire d'un fichier  
NB: nécessite d'être administrateur (super-user)

Exemple :  
`sudo chown -R paul.L1 projet/`  
Définit l'utilisateur paul et le groupe L1 au répertoire projet et à tous ses fichiers

**useradd** [options] login  
Ajouter un utilisateur

**passwd** [options] login  
Changer de mot de passe



### Manipulation des fichiers : copie

- **cp** [-ipr] source dest
- **cp** [option] [chemin vers fichier source] [chemin vers fichier destination]

➤ **Source = fichier**

➤ **Dest = fichier** : copie un fichier source en le renommant si le chemin du fichier destination contient un nom de fichier

➤ **Dest = répertoire** : recopier dans dest

options


- ❖ **-i** : confirmation en cas d'écrasement
- ❖ **-p** : préserve les attributs (propriétaire, groupe, date de création)
- ❖ **-r** : copie récursive (pour les répertoires imbriqués)

**Exemples :**

\$ cp toto /tmp/ → copie le fichier local toto dans /tmp (toujours nommé toto)

\$ cp toto /tmp/tata → copie le fichier local toto dans /tmp en le nommant tata

\$ cp -r projet /tmp → copie le contenu du répertoire projet dans le répertoire /tmp/projet



### Manipulation des fichiers : Déplacement

**mv** [option] [chemin vers fichier source] [chemin vers fichier destination]


Déplace un fichier source en le renommant si le chemin du fichier destination contient un nom de fichier

**Exemples :**

\$ mv toto /tmp/ → déplace le fichier local toto dans /tmp (toujours nommé toto)

\$ mv toto /tmp/tata → déplace le fichier local toto dans /tmp en le nommant tata

\$ mv -i toto /tmp → déplace le fichier toto dans /tmp en prévenant l'utilisateur s'il existe déjà un fichier /tmp/toto




### Manipulation des fichiers : Suppression

**rm** [option] [chemin vers fichier]  
supprime un fichier

**Exemples :**

- \$ rm toto → supprime le fichier toto
- \$ rm -i toto → supprime le fichier toto en demandant confirmation à l'utilisateur
- \$ rm -f toto\* → supprime les fichiers dont le nom commence par toto, sans demander confirmation à l'utilisateur
- \$ rm -r projet → efface récursivement le contenu du répertoire projet **non vide**



### Manipulation des fichiers Créer / supprimer un répertoire


**mkdir** [chemin vers répertoire]  
créer un répertoire

**rmdir** [chemin vers répertoire]  
Supprimer un répertoire **vide**

Sécurité: ne fonctionne que quand les répertoires sont vides  
Alternative: **rm -r**

**Exemples :**

- \$ mkdir toto → crée le répertoire toto
- \$ rmdir toto → supprime le répertoire vide toto
- \$ rmdir projet → rmdir: projet/: Directory not empty




## Recherche

- **find** [options]  
chercher dans une hiérarchie de répertoires les fichiers vérifiant certains caractéristiques données en options.


### Exemples

- **find** / -name charte -print  
chercher à partir de la racine tous les fichiers dont le nom est charte.
- **find** . -name \*.kwd -print  
chercher à partir du répertoire courant tous les fichiers dont l'extension est .kwd
- **which** commande  
effectue une recherche dans la liste des exécutables de la commande donnée et retourne le chemin d'accès complet d'une commande.  
Ex. **which echo** ! → **bin/echo**



## Commandes diverses

- **who**  
lister des utilisateurs connectés au système
- **date**  
afficher date et heure
- **file** fichier  
déterminer le type du fichier
- **head** [-n] fichier  
afficher les n premiers lignes du fichier
- **tail** [+n] -n] fichier  
+n : afficher à partir de la ligne numéro n  
-n : afficher le n dernières lignes
- **more** fichier  
afficher le fichier page par page
- **sort** fichier  
Trier le contenu d'un fichier
- **wc** [-cwl] fichier  
-c : nombre de caractères  
-w : nombre de mots  
-l : nombre de lignes



## Commandes diverses

Connaitre l'espace occupé par un répertoire / disque ?

**du** [option] fichier


Donne la taille en octets d'un fichier

**df** [option]

Donne la taille des données présentes sur chaque disque

**Exemples :**

**du** -sh projet → 4.0K projet/



## Manipulation d'archives et compressions

**tar** cf projet.tar projet/\*  
crée une archive contenant le contenu du répertoire projet et nommée projet.tar


**tar** xf projet.tar  
extraite le contenu de l'archive nommée projet.tar

**tar** zcf projet.tar projet/\*  
crée et compresse une archive contenant le contenu du répertoire projet et nommée projet.tar.gz

**tar** zxf projet.tar.gz  
extraite le contenu de l'archive compressée nommée projet.tar.gz

options

- **-c** : créer un archive
- **-x** : extraire les fichiers
- **-v** : obtenir une description du contenu archivé
- **-f** : pour spécifier un nom pour l'archive (en paramètre)
- **-z** : zipper/dézipper l'archive




## Manipulation d'archives et compressions

**gzip** fichier  
Comprime un fichier au format .gz (algorithme deflate)

**gunzip** fichier  
Décompresse un fichier au format .gz

Exemples :  
gzip toto.txt → toto.txt.gz  
gunzip toto.txt.gz → toto.txt



## Manipulation de texte

- **grep** [options] motif fichier1 fichier2 .. (motif= expression régulière)  
Chercher les lignes contenant le motif décrit par l'expression régulière dans les fichiers

**options**

- **-c** : indique seulement le nombre de lignes
- **-n** : indique les numéros des lignes trouvés
- **-i** : ne distingue pas majuscule et minuscules

Exemples :

\$ grep "listeria" /home/Cath/cours/\* → cherche, dans les fichiers du répertoire cours, des fichiers contenant le motif listeria

\$ grep -n "listeria" /home/Cath/cours/\* → idem, mais en affichant le numéro de ligne


\$ grep -c "listeria" /home/Cath/cours/\* → idem, mais en donnant le nombre d'occurrences du motif

- **sort** [-r] fichier  
Trier les lignes du fichier ou l'entrée standard.

**option**

- **-r** : renverser le tri





## Gestion de processus

- **Ps** [options]  
Afficher les informations sur les processus en cours d'exécution  
Exemple: **ps** ux

USER	PID	%CPU	%MEM	VSZ	RSS	TTY	STAT	START	TIME	%COMMAND
yannick	6316	0.0	0.0	13272	1728	?	SL	09:26	0:00	/bin/echo


- **top**  
Afficher les processus les plus actifs en temps réel, donne des informations sur l'activité du système (ressources occupées, etc)
- **Kill** [option] PID  
Envoyer un message à un processus donné, généralement pour y mettre fin
  - **signal SIGTERM (15)** par défaut : arrêter le processus proprement
  - **signal SIGKILL (9)** : terminer brutalement un processus



## Les entrées-sorties standards

Lors de l'exécution d'une commande, un processus est créé. Celui-ci va alors ouvrir trois flux :


- **stdin** l'entrée standard, par défaut le clavier, identifiée par l'entier **0** (descripteur)
- **stdout** la sortie standard, par défaut l'écran, identifiée par l'entier **1**
- **stderr** la sortie d'erreur standard, par défaut L'écran, identifiée par l'entier **2**



## Les redirections

Il est possible de rediriger les flux d'entrée-sortie au moyen d'opérateurs spécifiques :

- > redirection de la sortie standard (par exemple dans un **fichier**)
- < redirection de l'entrée standard
- >> redirection de la sortie standard avec concaténation
- > & redirection des sorties standard et d'erreur
- >| redirection avec écrasement de fichier
- | redirection de la sortie standard vers l'**entrée standard** (pipe)




## Redirection des entrées/sorties

### Redirection de l'entrée

**Exemple**

- cat < fichier  
afficher le contenu du fichier
- sort < participants.txt  
L'entrée standard de sort est prise dans le fichier indiqué.




## Redirection des entrées/sorties

### Redirection de la sortie

**Exemple**

- `ls > résultats`  
Créer un fichier nommé résultats contenant le contenu du répertoire courant
- `ls >> resultats`  
Ajouter à la fin du fichier résultats le contenu du répertoire courant
- `ls ~saddam/* > ~gwb/weapons_mass_destruction.txt`
- `cat obiwan_kenobi.txt > starwars_biographies.txt`
- `cat han_solo.txt >> starwars_biographies.txt`



## Redirection des entrées/sorties

### Redirection entrée et sortie:

**Exemple**

- `cat < fichier >> résultat` → ajouter au fichier **résultat** le contenu de **fichier** après l'avoir récupéré avec `cat`,


### Redirection vers des tubes (pipes): |

Les pipes Unix sont très utiles pour rediriger la sortie standard d'une commande vers l'entrée standard d'une autre commande.

**Exemple:**

- `ls | sort` → afficher le contenu du répertoire courant trié
- `cat *.log | grep -i error | sort`
- `grep -ri error . | grep -v "ignored" | sort u \ > serious_errors.log`
- `cat /home/*/homework.txt | grep mark | more`

Il s'agit d'une des fonctionnalités les plus puissantes des shells Unix!



## Liens Symboliques et Physiques

Un lien est un type spécial de fichier qui permet à plusieurs noms de fichiers de faire référence au même fichier sur le disque. (lien physique "dur" vs lien symbolique)


**Liens Symboliques:**

- Un lien symbolique est un fichier spécial qui est juste une référence au nom d'un autre (fichier ou répertoire).
- Utile pour simplifier et réduire l'utilisation du disque quand deux fichiers ont le même contenu.

**Exemple:**  
biographie\_anakin\_skywalker -> biographie\_darth\_vador

- Comment distinguer les liens symboliques:


**ls l :** affiche -> et le fichier référencé par le lien  
**GNU ls** affiche les liens avec **une couleur** différente



## Création de liens Symboliques:

- Pour créer un lien symbolique (même ordre que dans cp):  
**ln -s nom\_fichier nom\_lien**
- Pour créer un lien vers un fichier dans **un autre répertoire**, avec le même nom:  
**ln -s ../LISEZ\_MOI.txt**
- Pour créer plusieurs liens d'un coup dans un dossier donné:  
**ln -s fich1 fich2 fich3 ... Rep**
- Pour supprimer un lien:  
**rm nom\_lien**

Bien sûr, cela ne supprime pas le fichier référencé par le lien!



**Création de liens Physiques:**

- Un *lien physique* vers un fichier est un fichier ordinaire, avec exactement le même contenu physique
- Bien qu'ils économisent toujours de la place, les liens physiques sont indiscernables des fichiers d'origine.
- Si vous supprimez le fichier d'origine, cela n'affecte pas le contenu du lien physique. Le contenu est supprimé quand il n'y a plus aucun fichier (lien physique) qui y fait référence.

**Par défaut, *ln* crée des liens physiques**

***ln*** [options] fichier\_source lien


**Exemples :**

\$ ***ln*** /home/yannick/cours.pdf /home/yannick/projet/cours.pdf  
le fichier cours.pdf du répertoire projet du répertoire yannick est un lien vers le fichier cours.pdf du répertoire yannick

\$ ***ln -s*** /home/yannick/cours.pdf /home/yannick/projet/cours.pdf  
idem avec un lien symbolique



**Partie 6:**  
**Les Expression Régulières**




### ❖ C'est quoi une expression régulière?

Une expression régulière est **une suite de caractères** typographiques (qu'on appelle plus simplement « **motif** » – « **pattern** » en anglais) décrivant un ensemble de chaînes de caractères.

→ Par exemple l'ensemble de mots « ex-équo, ex-equo, ex-aequo et ex-æquo » peut être condensé en un seul motif:  
« `ex-(a?e|æ|é)quo` ».

Les mécanismes de base pour former de telles expressions sont basés sur des **caractères spéciaux de substitution**, de **groupement** et de **quantification**. [Définition Wikipedia]



### Expressions régulières

- Analyser des chaînes de caractères
- Pattern matching
- **Utilisation :**
  - Commandes : grep, sed, ...
  - Éditeurs de textes : vi, nedit, ...
  - Langages de programmation : php, perl, ...



### Les symboles ^ et \$

^ : début d'un pattern

\$ : fin d'un pattern


**Exemples :**

^at : chaîne de caractères qui commence par « at »

cot\$ : chaîne de caractères qui se finit « cot »

^mot\$ : chaîne de caractères « mot »

test : chaîne de caractères qui contient « test »




### Les symboles \*, + et ?

Nombre de fois qu'un caractère (suite de caractères) puisse apparaître

- \* : aucune fois ou plusieurs fois
- + : une fois ou plusieurs fois
- ? : aucune fois ou une et une seule fois

**Exemples :**

- ab\* : chaîne de caractère contenant un a suivi d'un, de plusieurs, ou d'aucun b ("a", "ab", "abb", ...)
- ab+ : chaîne de caractère contenant un a suivi d'au moins un b ("ab", "abb", "abbb", ...)
- ab? : chaîne de caractère contenant un a suivi d'un ou d'aucun b ("a", "ab", mais pas "abb")
- a?b+\$ : chaîne de caractères composée d'aucun ou d'un seul a, suivi d'un ou de plusieurs b, le tout étant situé à la fin de la chaîne




**Les accolades {}**

Nombre d'occurrences de la chaîne

**Exemples :**

- $ab\{2\}$  : chaîne de caractère composée d'un a suivi d'exactly deux b ("abb")
- $ab\{2,\}$  : chaîne de caractère composée d'un a suivi d'au moins deux b ("abb", "abbb",...)
- $ab\{,5\}$  : chaîne de caractère composée d'un a suivi de jusqu'à cinq b ("a", "ab" ... et "abbbbb")
- $ab\{3,5\}$  : chaîne de caractère composée d'un a suivi de trois à cinq b ("abbb", "abbbb" et "abbbbb")




**Les parenthèses ()**

Quantifier une chaîne de caractères

**Exemples :**

- $a(bc)^*$  : chaîne de caractères commençant par un a suivi d'aucune ou de plusieurs séquence de caractères "bc"
- $a(bc)\{1,5\}$  : chaîne de caractères commençant par un a suivi d'une à cinq fois la séquence de caractères "bc"





**Le symbole |**

Comme opérateur booléen OU

**Exemples :**

- `toto|titi` : chaîne de caractères contenant le mot "toto" ou le mot "titi"
- `(b|cd)ef` : chaîne de caractères qui contient la séquence de caractères "bef" ou bien la séquence de caractères "cdef"

A noter: `(b|cd)ef` équivaut `(bef|cdef)`


- `(a|b)*c` : chaîne de caractères qui contient une alternance de a et de b, se terminant par un c ("`bababbbaac`", "`c`", "`bc`")

**Le symbole .**

N'importe quel caractère unique.

**Exemple :**

- `^. {3}$` : chaîne de caractères comportant exactement trois caractères




**Les crochets []**

Les caractères permis à un endroit précis d'un modèle

**Exemples :**

- `[ab]` : chaîne de caractères contenant un "a" ou un "b"
- `[a-d]` : chaîne de caractères qui contient les lettres minuscules comprises entre le "a" et le "d"
- `^[a-zA-Z]` : chaîne de caractères qui commence par une lettre minuscule ou bien par une lettre majuscule
- `[0-9]%` : chaîne de caractères qui contient un pourcentage à un seul chiffre
- `,[a-zA-Z0-9]$` : chaîne de caractères qui finit par une virgule suivi d'un caractère alphanumérique



**Le ^ dans les crochets**

- ^ comme premier symbole dans les crochets: « tout sauf »
- ^[a]: chaîne de caractères qui ne commence pas par « a »

**Notez**

Si l'on veut qu'un méta-caractère apparaisse tel quel, il faut le précéder d'un **backslash**

Pour que les méta-caractères ?, +, {, }, |, (, et ) gagnent leurs significations spéciales dans un terminal, il faut utiliser un backslash :

- \?, \+, \{, \}, \|, \(, et \).

**Exemple:**

grep "**^a\b\\c\\{1,3\\}**" fichier permet de reconnaître a**(b|c){1,3}**

Par exemple, **[a-z]\\[0-9]\\** permet de trouver **c[8]**.



**Bibliographie:**

- [1] Cours de **Vincent Granet** (Polytech'Nice-Sophia)
- [2] Introduction à Unix et GNU / Linux, par Michael Opdenacker (Free Electrons), Traduction française par Julien Boibessot. Mise à jour Fabien Deleu (Département GTR de l'IUT de Béthune)
- Introduction à LINUX, de M. Abdallah ELKHYARI, Univ. Jean Monet St Etienne
- <https://moodle.polymtl.ca/mod/url/view.php?id=47398>
- Cours "Systèmes d'exploitation", Audrey Queudet, Univ Nantes, 2010

**Pour aller plus loin:**

**Livres**

- *Linux pour les nuls*, Dee-Ann Leblanc, First Interactive, 2006.
- *Linux en pratique*, Arnold Robbins, Campus Press, 2007.
- *Linux programmation système et réseau , cours exemples et exercices corrigés en C-C++*, Joëlle Delacroix, Dunod, 2007

**Sites web**

- <http://www.linux.org/>
- <http://www.linux-france.org/>