Unix/Linux I

Dr Mikaël A. Mousse

Institut Universitaire de Technologie

Université de Parakou

1. Introduction

2. Systèmes d'exploitation, Unix et Linux
   - Fonctions et spécifité d'Unix
   - Architecture

3. Connexion-Déconnexion

4. Commandes Unix

5. Système de fichiers
   - Fichier Unix
   - Arborescence de fichiers
   - Quelques commandes sur les fichiers
   - Chemins d'accès
   - lien symbolique

Un système d'exploitation

Exemples connus :

▷ Windows, ▷ Linux, ▷ OS X

Qu'apportent-ils ?

- ▷ La possibilité d'utiliser l'ordinateur par une interface graphique ou plutôt une interface homme-machine
  - lancer des programmes
  - copier/déplacer/$\cdots$ des fichiers
- ▷ Permettre aux programmes de fonctionner quelque soit le matériel
  - jouer à un jeu vidéo quelque soit la carte vidéo et sa performance, avec plus ou moins d'options

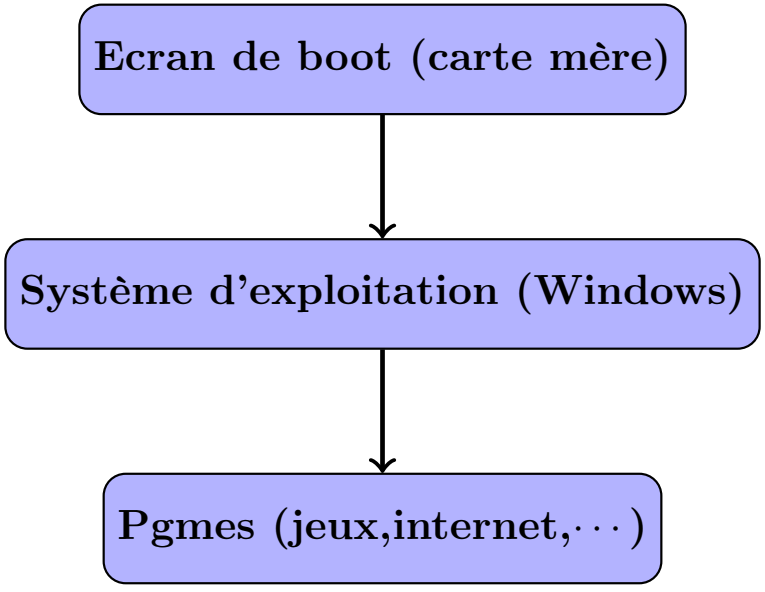
Un système d'exploitation

Pour aller plus loin :

L'OS (Operating System) gère

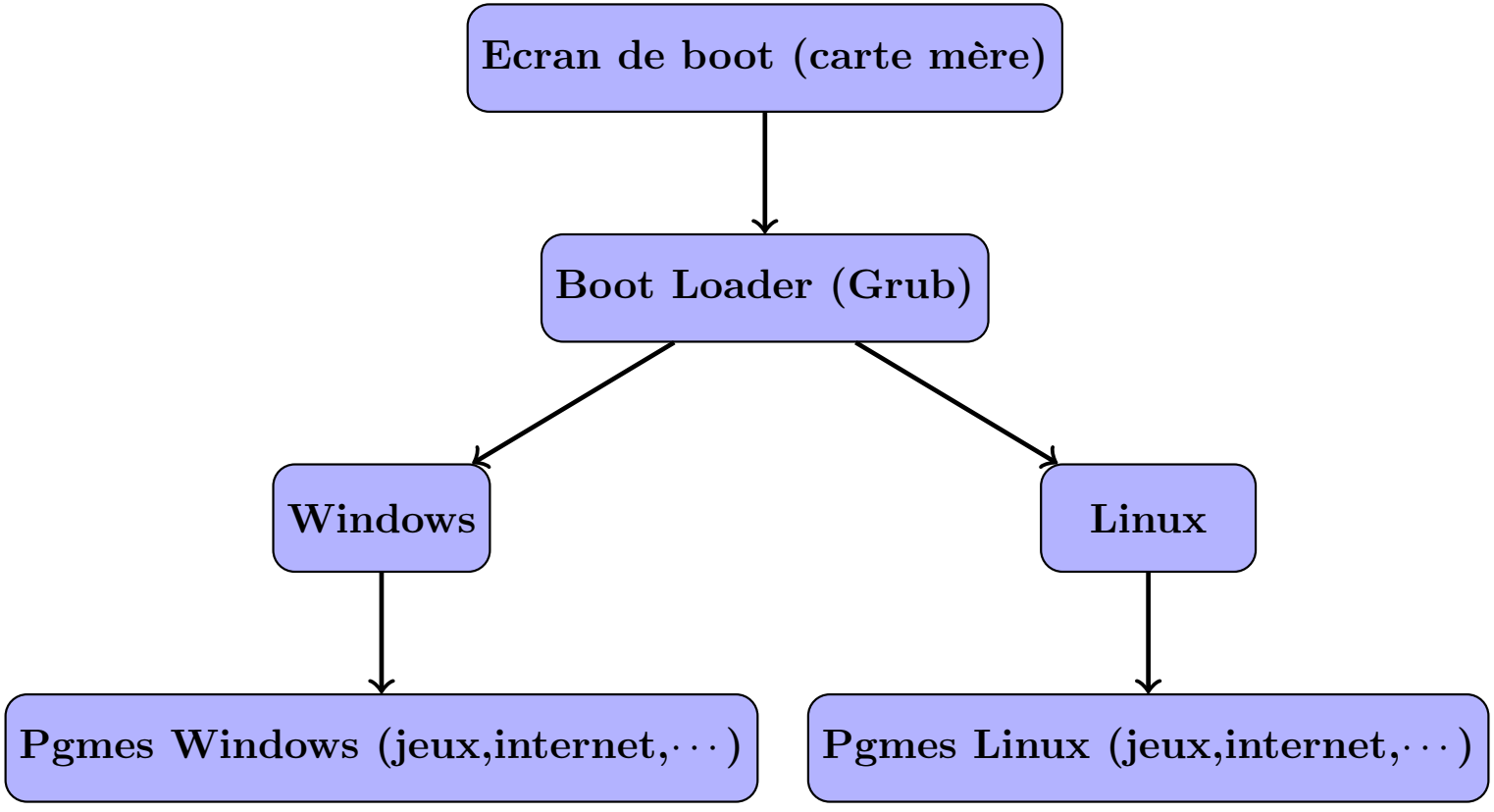
- **La mémoire** : il la partage entre tous les programmes
- **Les périphériques** : écran, imprimante, disque dur, réseau. Il s'assure que les programmes puissent les utiliser de façon standard.
- **Le processeur** : il le partage entre tous les programmes pour qu'ils aient l'air de fonctionner parallèlement
- **Les utilisateurs** : gérer les droits d'accès aux fichiers, comme au matériel
- **La standardisation des programmes** : offre des interfaces de programmation simplifiées et standardisées.

Vue générale



Mais pas que!!! Peut on avoir 2 OS sur son ordinateur???

Vue générale



Système d'exploitation

Définition (Système d'Exploitation)

Un système d'exploitation (**SE**) est un ensemble de programmes responsables de la liaison entre les ressources matérielles d'un ordinateur et les applications informatiques de l'utilisateur (traitement de textes, vidéo,...).
Il fournit aux programmes applicatifs des points d'entrées génériques pour les périphériques.

Le système Unix est un système d'exploitation multi-utilisateur et multi-tâche

Unix est multi-utilisateurs

Multi-User : Plusieurs utilisateurs sous Unix. Chacun dispose de l'ensemble des ressources du système. Comme tout système multi-utilisateur, Unix comporte des mécanismes d'identification et de protection permettant d'éviter toute interférence entre utilisateurs.

2 types de Users :

1. Users *normaux* : compte avec
   - Login
   - password
   - Espace de travail protégé (**rep.** privé `-home directory`)
   - mail box
2. *Super-User* `root` gère tout le système

Unix est multi-tâche

Multi-tâches : Unix est multi-tâche car plusieurs programmes peuvent être en cours d'exécution en même temps sur une même machine.
Un **processus** est une tâche en train de s'exécuter. On appelle **processus**, l'image de l'état du processeur et de la mémoire au cours de l'exécution du programme.
En fait, à chaque instant, le processeur ne traite qu'au plus un seul des programmes lancés. La gestion des processus est effectuée par le système.

Fonctions principales d'Unix

- Partage des ressources équitables : veiller au partage équitable des ressources entre tous les processus.
- Interface avec le matériel :passage par des fichiers spéciaux gérés par le SE. pour accéder à une ressource matériel (disque dur, lecteur de disquettes,...)
- Gestion de la mémoire :partage correct de la RAM entre processus.
- Gestion des fichiers : Unix fournit un mécanisme de protection des fichiers.

Unix fonctionne par couche

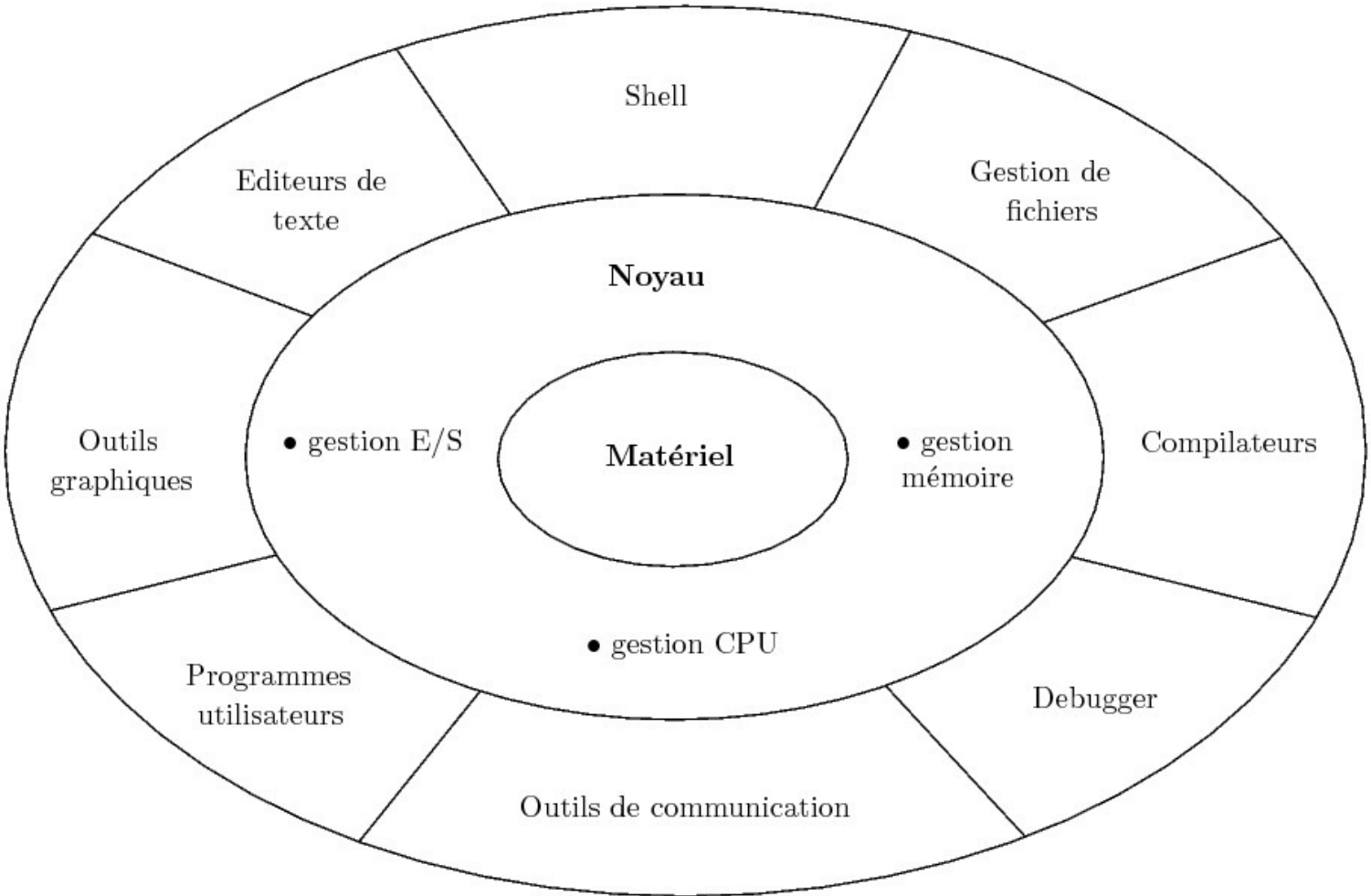
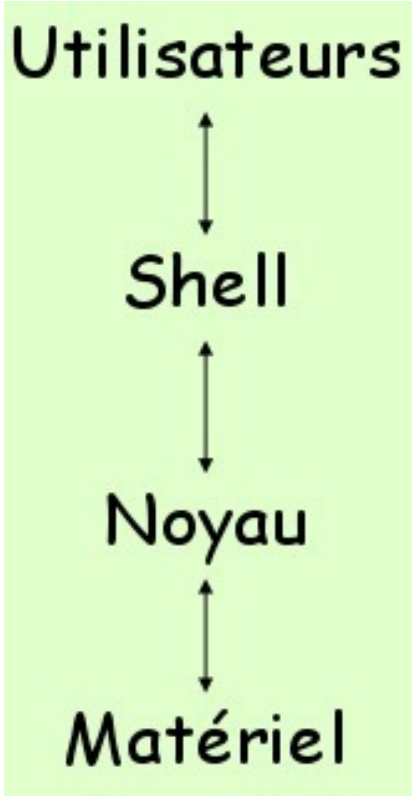


Schéma d'exploitation de la machine



- **shell :** interpréteur de commandes Unix (vérifie, interprète les commandes, exécute et renvoie les réponses). Le Shell envoie des appels au noyau en fonction des requêtes des utilisateurs
- **noyau :** couche logicielle la plus interne du S.E Unix dédiée à la gestion des composants matériels : processeur, mémoire, périph.
- Autour du noyau gravite un certain nombre d'utilitaires.

Connexion-Déconnexion

Connexion : S'identifier pour ouvrir une session (de travail) :

- Entrer nom de connexion aprés le message `login`
- Entrer mot de passage aprés le message `password`

$\rightsquigarrow$ L'utilisateur de trouve alors dans son répertoire privé correspondant à son `login` (`home directory`)

Déconnexion : En l'absence d'environnement graphique, une simple commande `exit` suffit pour terminer ma session de travail.

Choisir son mot de passe

Un bon mot de passe :

- posséder entre 7 et 8 caractères
- posséder au moins une lettre majuscule
  au moins un chiffre
  et un caractère de ponctuation
- ne pas contenir de donnés relatives à votre identité
- ne pas appartenir à un dictionnaire
- ne pas contenir de répétition de caractères
- ...

Commande Unix en console

Unix fonctionne en mode ligne de commandes et non en mode graphique ⇒ permet des opérations plus complexes.
Une commande est un programme. Pour l'exécuter ⇒ taper son nom éventuellement suivi d'options et d'arguments.
Syntaxe :

```
nom_commande [-liste_options][liste_arguments]
```

Exemple : `ls -l` ↩
Lors de l'appui sur la touche `Entrée`, le shell analyse la ligne de commande et l'interprète.
Différence entre majuscules et minuscules. : On dit que la console Unix est sensible à la casse.

Commande Unix en console

Aide en ligne : Doc. de référence organisée en 9 sections

1	Commande users
2	Appels système
3	Fct. bib. standard
4	Formats fichier
5	Tables
6	Jeux
7	Drivers périph.
8	Commandes admin.
9	Commandes locales

Visualiser une page du manuel :

`man[-s section] nom_commande`

Recherche page qui se rapporte a un mot clé :

`man -k mot-clé`

Quelques commandes :

- `who` Affiche les users actuellement connectés
- `date` Consulter date et heure
- `cal [mois[année]]` Affiche calendrier

Les fichiers sous Unix

Définition (fichier)

fichier : objet recevant et délivrant des données, constitué d'une chaîne de caractères non structurée.

Type de Fichiers :

- Fich. ordinaire : données stockées sur un disque
- Répertoire : ensemble d'informations permettant l'accés à d'autres fichiers
- Fich. spécial : dispositif d'entrée/sortie (terminal, lecteur,...)

Description de Fichiers : dans un **i-nœud** (**inode**) comportant

- type de fichier, mode de protection, nb. liens, num. propriétaire
- num groupe, taille fichier, adr.physique direct
- date et heure dernière modif., date heure dernier accés,....

Les fichiers sous Unix

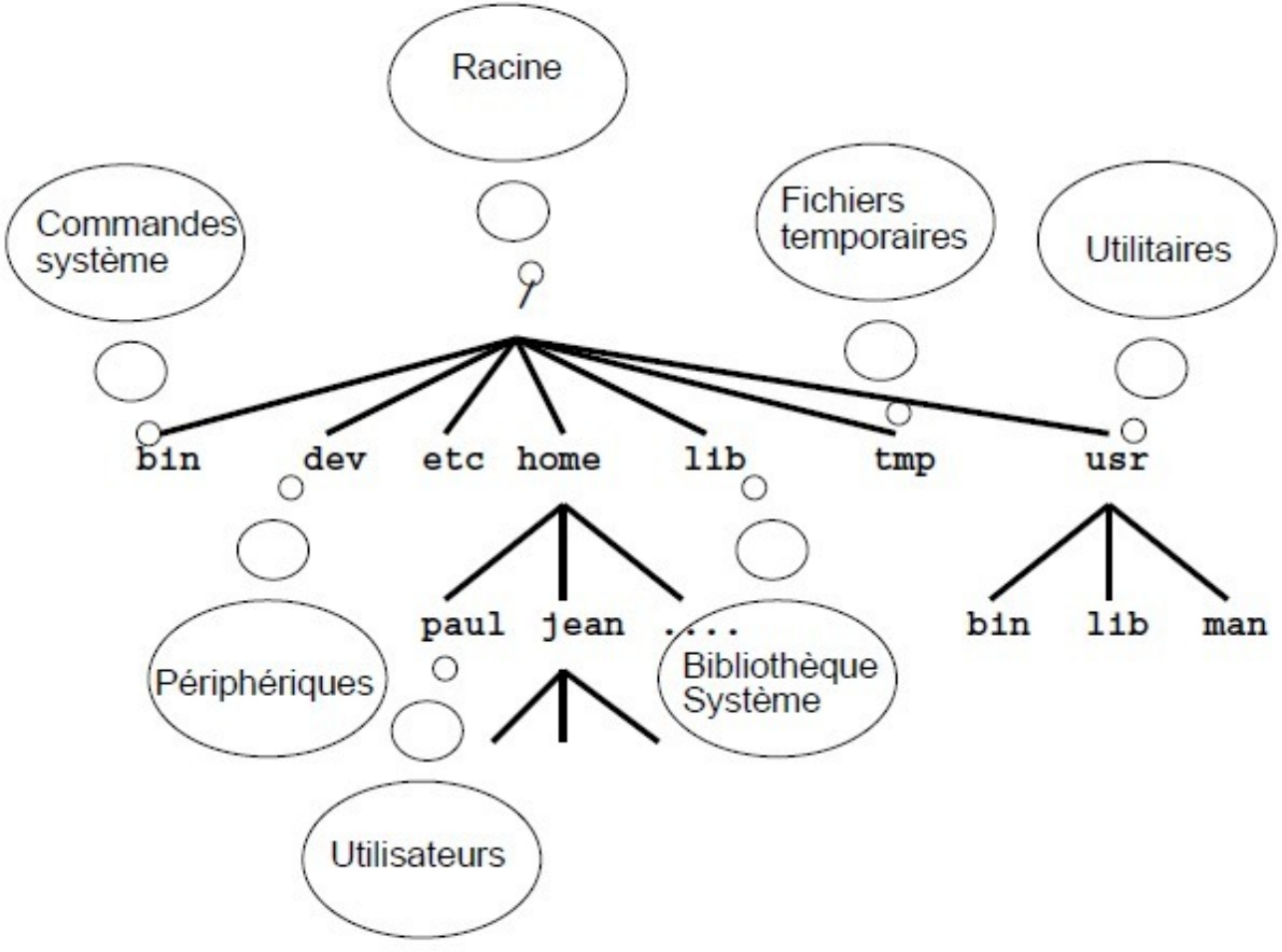
`ls -i fich` : numéro i-œud du fichier fich.
`ls -l rep` : liste contenu repertoire rep `-l` fournit des détails des fichiers contenu dans le répertoire rep.

```
-rw-rw-r- 1 nicolas nicolas 3205 août 24 09:53 demenagement.org
-rw-rw-r- 1 nicolas nicolas 2453 juil. 18 16:07 recherche.org
```

Opérations élémentaires sur les fichiers

`cat fich`	affiche sur la sortie standard le contenu de `fich`
`more fich`	affiche contenu de `fich` page par page
`head fich`	affiche début de `fich`
`tail fich`	affiche fin de `fich`
`sort fich`	trie lignes de `fich`
`ls rep`	affiche contenu de `rep`

Arborescence des fichiers



".." désigne le répertoire parent.

Accès aux fichiers

Atteindre un fichier :

- Référence absolue : chemin à partir de la racine (`/usr/local/bin`)
- Réference relative : chemin à partir du répertoire de travail

Commandes :

`pwd`	indique la réf. absolue du rep de travail
`cd`	`cd ..` permet de remonter dans l'arbre
`ls -R`	liste récursivement les sous-rep. et leur contenu
`mkdir`	création d'un rep

À sa création, un rep contient deux liens :
(index, .) : un lien sur lui-mème
(index, ..) : un lien sur son père
Le nombre de liens sur un rep. est $\geqslant 2$
(index, nom) dans le rep. père
(index, .) dans lui-mème

Opérations sur les fichiers

cp - to copy - copier

`cp f_source f_dest` recopie physique de `f_source` dans `f_dest`

rm - to remove - supprimer

`rm fich` suppression de `fich`

`rm -r fich` suppression du rep `fich` et de son contenu

mv - to move - déplacer

`mv f_source f_dest` renommer le fichier `f_source` en `f_dest`

`mv f rep_accueil` déplace le fichier `f`

Chemin absolu et relatif

Notion de chemin d'accès :

- Pour identifier un fichier : suite de noms étiquetant les arêtes le long de l'arborescence.
- racine absolue : /
- / sert aussi de séparation entre sous-répertoires.

Référence absolue : chemin d'accès `pathname` depuis la racine (permettant le repérage sans ambiguité)
Exemple : `/home/prot1/formation/M1IR`
Référence relative : Selon l'endroit où l'on se situe (répertoire de travail = `working directory`), repérer un fichier peut s'effectuer de manière relative.
Exemple : `../../DESS`

Des fichiers physiques différents appartenant à des disques logiques distincts peuvent avoir le même index de i-nœud ⇒ impossible de créer des liens

Le système Unix permet de créer des liens symboliques entre des fichiers.

Définition (Lien symbolique)

fichier contenant la référence absolue d'un autre fichier. Toute opération sur ce fichier (lecture, écriture, ...) s'effectue sur le fichier référencé. Un lien est créé pour pouvoir accéder au même fichier à différents endroits de l'arborescence.

Commandes : (à utiliser dans un rÃ©pertoire de travail)

`ln -s f_cible lien_nom` crée un lien symbolique `lien_nom` contenant la référence à `f_cible`

`ls -l` fait apparaître le lien sous la forme `fich_dest -> fich_source`