

```

linux/kernel/panic.c
Copyright (C) 1991, 1992 Linus Torvalds
*/

/*
 * This function is used through-out the kernel (unless you
 * to indicate a major problem.
 */
#include <linux/config.h>
#include <linux/module.h>
#include <linux/sched.h>
#include <linux/delay.h>
#include <linux/reboot.h>
#include <linux/notifier.h>
#include <linux/init.h>
#include <linux/signal.h>
#include <linux/syscalls.h>
#include <linux/interrupt.h>
#include <linux/mm.h>

int panic_timeout;
int panic_on_oops;
int tainted;

EXPORT_SYMBOL(panic_timeout);

struct notifier_block *panic_notifier_list;

EXPORT_SYMBOL(panic_notifier_list);

static int __init panic_setup(char *str)
{
    panic_timeout = simple_strtoul(str, NULL, 10);
    return 0;
}

__setup("panic=", panic_setup);

/**
 * panic - halt the system
 * @fmt: The text string to print
 *
 * Display a message, then perform a hard reboot. Functions
 * in the panic_notifier_list are called after the message is
 * printed. This function never returns.
 */
void panic(const char *fmt, ...)
{
    static char buf[1024];
    va_list args;
    if (defined(CONFIG_MAGIC_SYSRQ) && magic_sysrq)
        unsigned long caller = (unsigned long) __builtin_return_address(0);
    if (caller == 0)
        return;

    bust_spinlocks(1);
    va_start(args, fmt);
    vsprintf(buf, size_t(1024), fmt, args);
    va_end(args);
    printk(KERN_EMERG "Kernel panic - not syncing: %s\n", buf);
    bust_spinlocks(0);

    if (CONFIG_SMP)
        smp_send_stop();

    notifier_call_chain(panic_notifier_list, 0, buf);

    if (panic_timeout > 0)
        delay(panic_timeout);
}

```

## Chapitre 1

# Système d'Exploitation UNIX

## Introduction à l'UNIX

BOUKRI  
KHALIL

# Systemes d'Exploitation: Rappels

Le **Système d'Exploitation** (Operating System: **OS**) est un ensemble de programmes qui assure la gestion de l'ordinateur et ses périphériques.

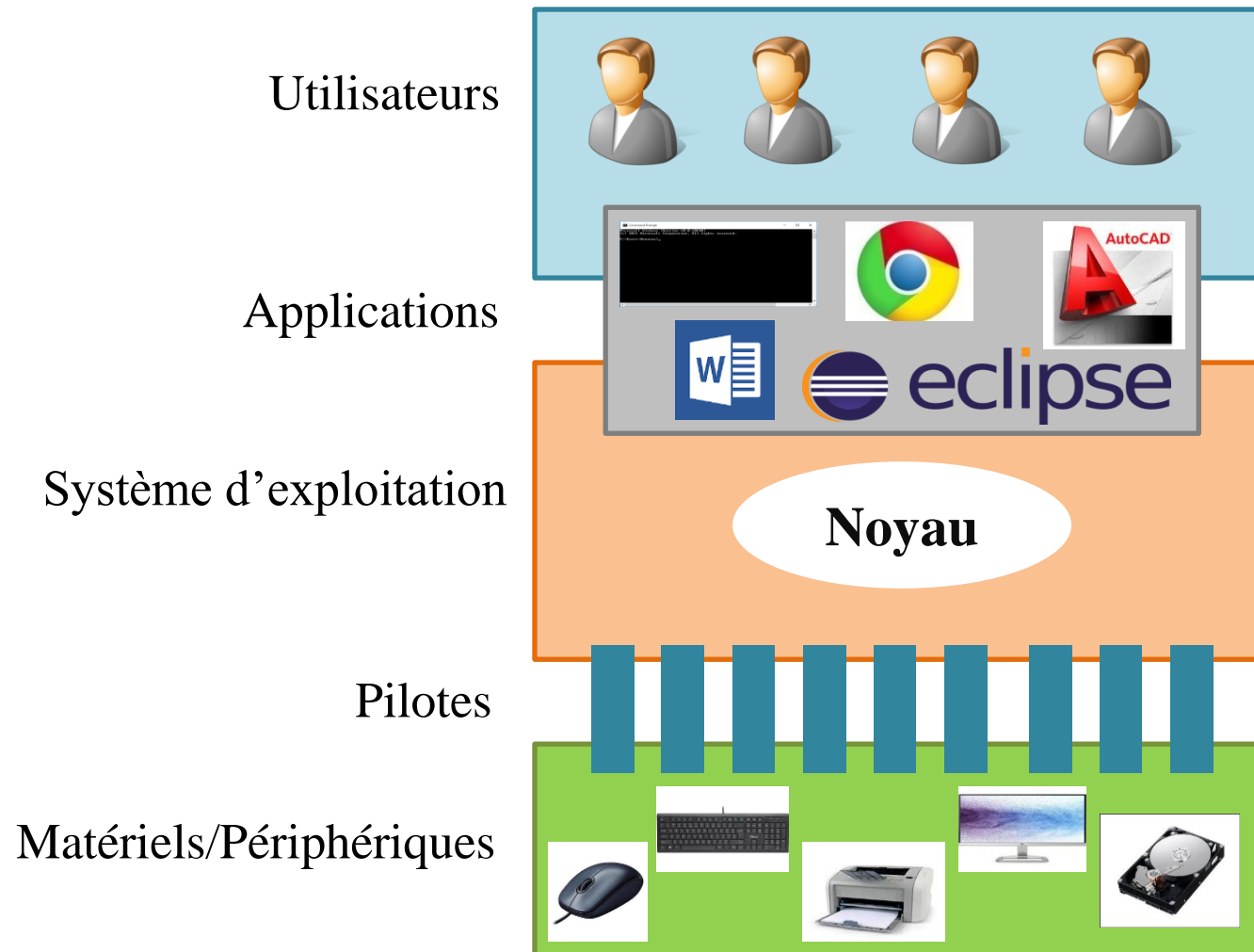
Le système d'exploitation sert d'interface entre matériel (hardware) et applications (software).

Les fonctions principales d'un **S.E.** sont :

- Contrôle des ressources (allocation et gestion du CPU et de la mémoire)
- Contrôle des processus
- Contrôle des périphériques

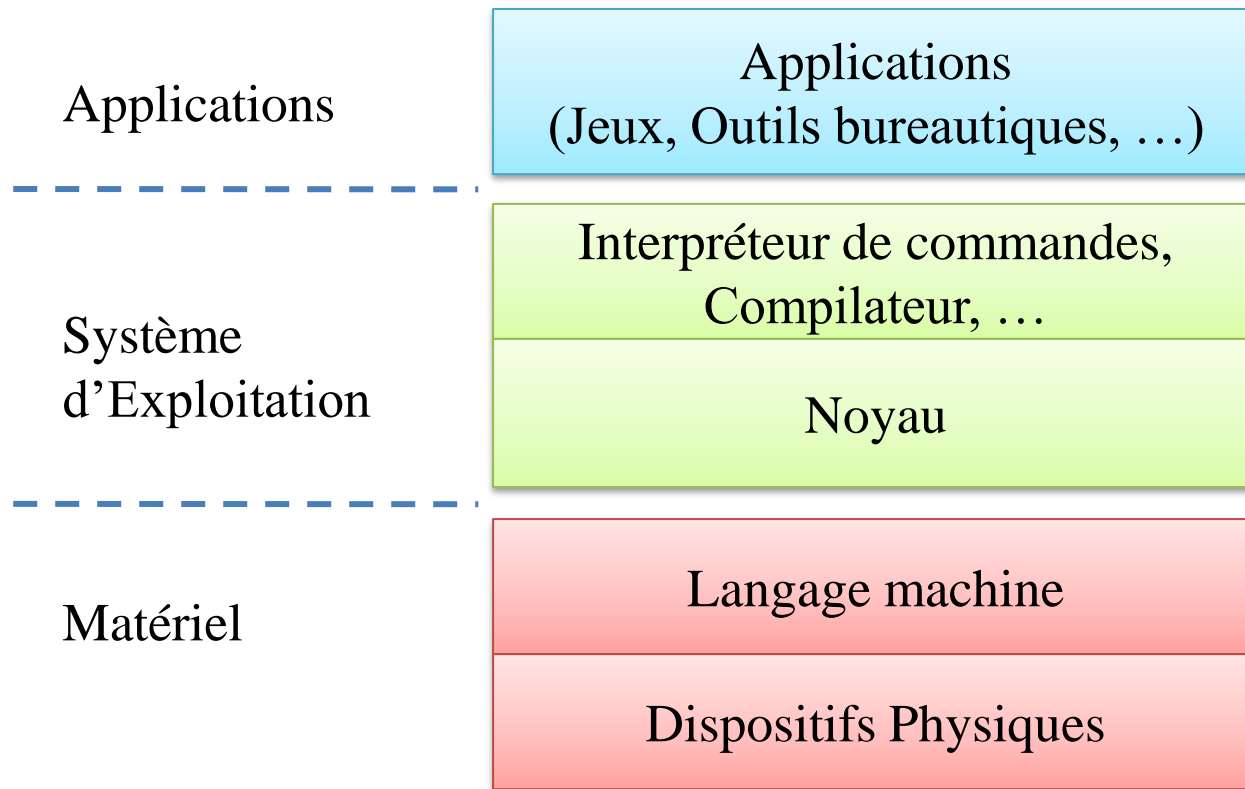
Il contient des outils de gestion utilisables par les applications, tels que la manipulation de fichiers, gestion d'impressions, date...

# Systemes d'Exploitation: Rappels



# Systemes d'Exploitation: Rappels

## Architecture-Type:



### **Système mono-tâche:**

- Gère une seule tâche à la fois.
- Le seul programme lancé utilise les ressources de la machine et ne rend la main au système d'exploitation qu'en fin d'exécution, ou en cas d'erreur.

**Exemple:** MS-DOS

### **Système multi-tâches:**

- Gère plusieurs tâches simultanément sur une même machine.
- Le système partage le temps du processeur entre plusieurs programmes.
- Le principe est d'allouer du temps à différentes applications qui sont découpées en séquence d'instructions (Tâche ou Processus), ces tâches seront tour à tour actives, en attente, suspendues ou détruites suivant la priorité qui leur est associée.

**Exemples:** OS2 d'IBM, Windows 95, 98 ...

### **Système multi-utilisateurs:**

- Gère l'environnement propre à chaque utilisateur (identification, ressources propres)
- Gère la sécurité d'accès aux programmes et aux données
- Notion de droits d'accès

**Exemples:** Windows 2000, 2003, XP, Unix (HP-UX, Solaris, AIX, Linux, FreeBSD...)

### **Système multi-processeurs:**

- C'est un système nécessairement multi-tâches puisqu'on lui demande d'une part de pouvoir exécuter simultanément plusieurs applications, mais surtout d'organiser leur exécution sur les différents processeurs (qui peuvent être identiques ou non).
- Ces systèmes peuvent être soit architecturés autour d'un processeur central qui coordonne les autres processeurs, soit avec des processeurs indépendants qui possèdent chacun leur système d'exploitation, ce qui leur vaut de communiquer entre eux par l'intermédiaire de protocoles.

### **Système multi-utilisateurs:**

- Gère l'environnement propre à chaque utilisateur (identification, ressources propres)
- Gère la sécurité d'accès aux programmes et aux données
- Notion de droits d'accès

**Exemples:** Windows 2000, 2003, XP, Unix (HP-UX, Solaris, AIX, Linux, FreeBSD...)

### **Système multi-processeurs:**

- C'est un système nécessairement multi-tâches puisqu'on lui demande d'une part de pouvoir exécuter simultanément plusieurs applications, mais surtout d'organiser leur exécution sur les différents processeurs (qui peuvent être identiques ou non).
- Ces systèmes peuvent être soit architecturés autour d'un processeur central qui coordonne les autres processeurs, soit avec des processeurs indépendants qui possèdent chacun leur système d'exploitation, ce qui leur vaut de communiquer entre eux par l'intermédiaire de protocoles.

# Systemes d'Exploitation: Rappels

Systeme	Codage	Mono-utilisateur	Multi-utilisateurs	Mono-tâches	Multi-tâches
DOS	16 bits	X		X	
Windows 3.1	16/32 bits	X			Non préemptif
Windows 95/98/Me	32 bits	X			coopératif
Windows NT/2000	32 bits		X		X
Windows XP	32/64 bits		X		X
Unix/Linux	32/64 bits		X		X
MAC/OS X	32 bits		X		X
...					



# Historique d'UNIX

- **1969** : Bell Laboratories, centre de recherches commun à AT&T et Western Electric, Ken Thompson travaille sur **MULTICS** (Multiplexed Information and Computing Service). Bell se retire du projet, Multics est abandonné. Ken Thompson décide de développer son propre OS, en s'éloignant volontairement de tout existant et écrit **UNICS** (Unified Information and Computing System) sur DEC PDP-7. Équipe : Dennis Ritchie, Rudd Canaday, puis Brian Kernighan.
- **1970** : Premier portage sur DEC PDP-11/20, avec le premier compilateur C, conçu spécialement pour rendre cet OS portable.
- **1971** : Version 1 d'Unix sur PDP/11-20 avec un système de fichiers, fork(), roff, ed, suite à la demande de AT&T qui avait besoin d'un système de traitement de textes pour l'aide à l'écriture de ses brevets.
- **1973** : La V2 intègre les tubes (pipes)
- **1974** : AT&T ne voyant pas d'avenir commercial à Unix, décide de distribuer le code source aux universités selon quatre critères de licence. Unix gagne donc la faveur des universitaires. Entre 1974 et 1977 les versions de la V3 à la V6 voient le jour.

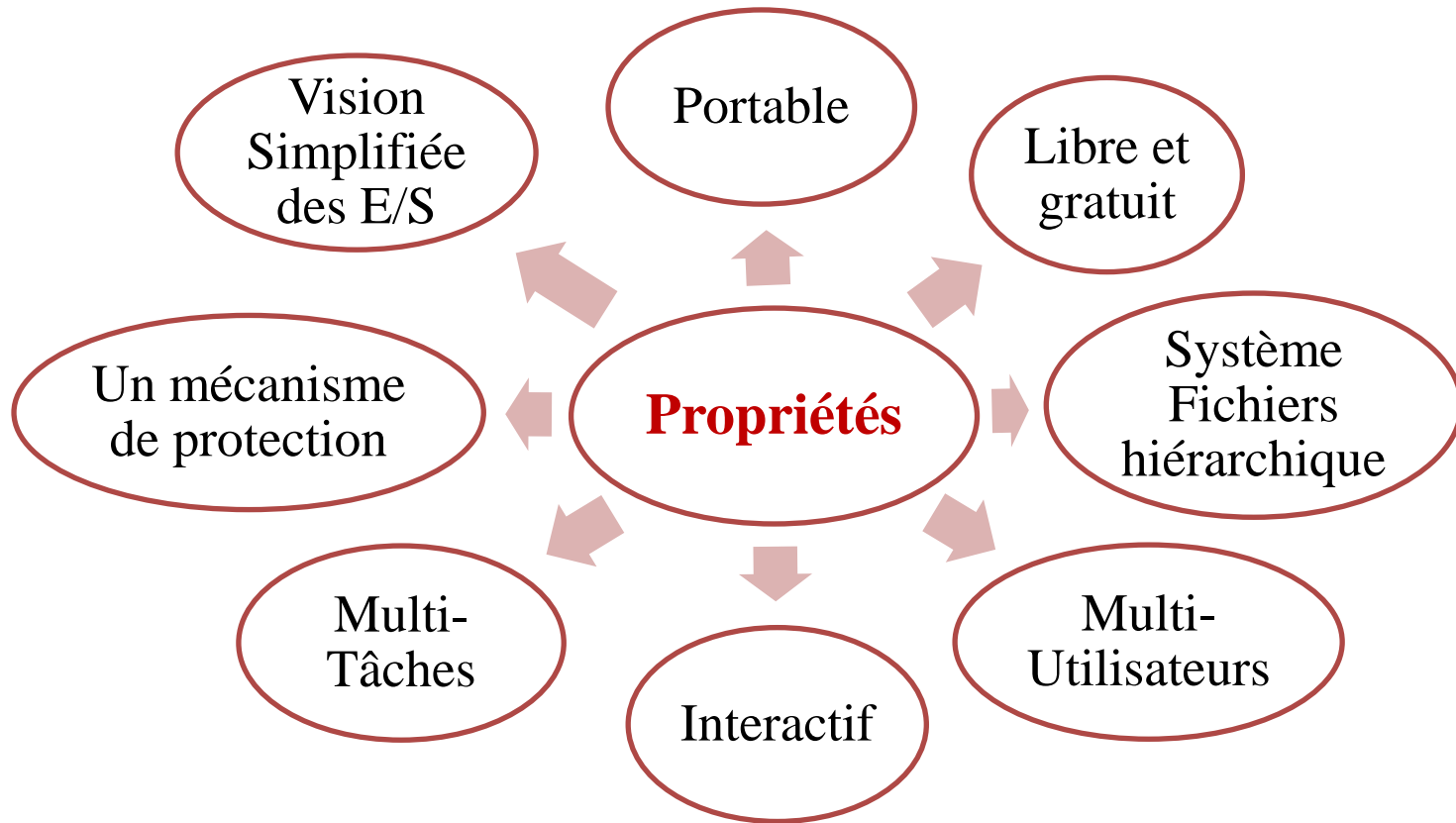
# Historique d'UNIX

- **1979** : Le coût des licences Unix incite l'université de Californie à Berkeley à continuer ses travaux sur les sources diffusées avant la licence, et crée sa propre variante : BSD Unix. Le DARPA décide d'utiliser Unix pour ses développements, notamment BSD Unix.
- **1983** : AT&T met en vente la version commerciale de Unix SYSTEM V.
- **1986** : Première ébauche des normes POSIX sur la standardisation des appels systèmes et des fonctions.
- **1987** : Création de X-Window, interface C/S graphique développée au sein du MIT. System V v3, premiers Unix propriétaires de HP et IBM suite à la modification de la licence de SYSTEM V. BSD 4.3, Unification de BSD et SYSTEM V (Sun et AT&T), d'où abandon des particularités de chaque système.
- **1988** : Troisième version de X/Open Portability Guide, servant de référence pour tous les développements d'Unix ultérieurs (commandes, appels système, langages, requêtes, graphique, internationalisation, réseau).

# Historique d'UNIX

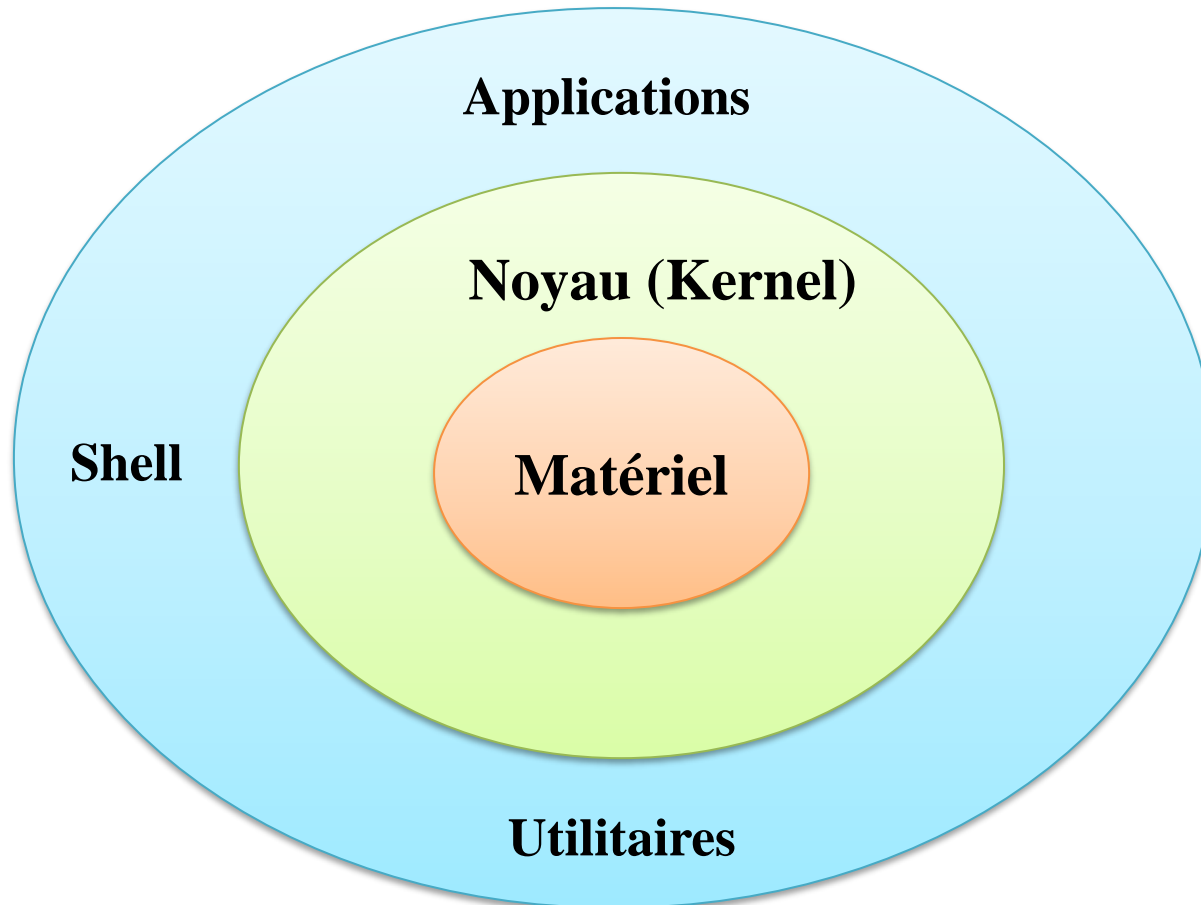
- **1990** : System V v4 de AT&T, nouveaux standards d'unification avec Sun. Les autres constructeurs se sentent menacés et fondent OSF (Open Software Foundation).
- **1991** : OSF/1. Apparition des premiers clones Unix comme Linux et FreeBSD.
- **1992** : Sun sort Solaris (SunOS), dérivé de System V v4, avec la gestion des threads. AT&T crée USL (Unix Software Laboratories) et transfère toutes les licences à cette société.
- **1993** : Novell rachète USL, puis transfère les droits de licences à X/Open.
- **Depuis 1993** : S'il existe un grand nombre d'Unix propriétaires, la plupart restent conformes aux normes et standards établis (X/Open, Posix). On distingue deux grandes branches SYSTEM V et BSD. Les deux sont compatibles. L'arrivée de **Linux** (dérivé de System V mais avec pas mal d'améliorations issues de BSD) a changé la donne.
- Les code source d'Unix appartient aujourd'hui à la société Caldera issue de Novell, mais les droits et la force de proposition sont transférés à l'Open Group.

# Caractéristiques d'UNIX



# Caractéristiques d'UNIX

## Architecture:



## Une session Unix:

- Travailler sous Linux implique une connexion au système.
- Login: Identification de l'utilisateur:
  - Login** : <tapez ici votre nom d'utilisateur>
  - Password** : <tapez ici votre mot de passe>
- Le mot de passe n'apparaît pas en clair et doit être tapé en aveugle. En cas d'erreur, un message indiquera :
  - Login incorrect*
- Un super-utilisateur existant dans le système qui a tous les droits c'est l'utilisateur: **root**.
- Logout: NE PAS ETEINDRE une machine “sauvagement”, commande “logout” dans la console

## Interpréteur de commandes:

### Le shell:

Après le login, dans une console, vous voyez le prompt (par exemple):

*[user@machine~]\$ \_*

Le prompt:

**user:** nom d'utilisateur

**@** : dans

**machine:** nom de la machine où vous êtes connecté

**~** : le répertoire courant (dossier personnel ou home)

**\$** : le niveau d'autorisation donnée

\$ signifie que vous êtes en train d'utiliser un compte utilisateur « normal », avec des droits limités (il ne peut pas modifier les fichiers système les plus importants).

# : signifie que vous êtes en mode super utilisateur, c'est-à-dire que vous êtes connectés sous le pseudonyme « root ». Le root est l'utilisateur maître qui a le droit de tout faire sur sa machine.

## Interpréteur de commandes:

### Le shell:

Le shell est un programme qui interprète vos commandes et les transmet au noyau unix (OS). Ils permettent en plus de définir un environnement.

Son rôle consiste ainsi à lire la ligne de commande, interpréter sa signification, exécuter la commande, puis retourner le résultat sur les sorties.

Il existe plusieurs shells, les plus communs sont: bash et tcsh.

### Définir/modifier un environnement de shell:

Votre environnement de shell est défini par des variables, par exemple: **PATH**

Cette variable définit la liste des répertoires où le Shell cherche le binaire correspondant à la commande que vous tapez (p.e.: passwd ).

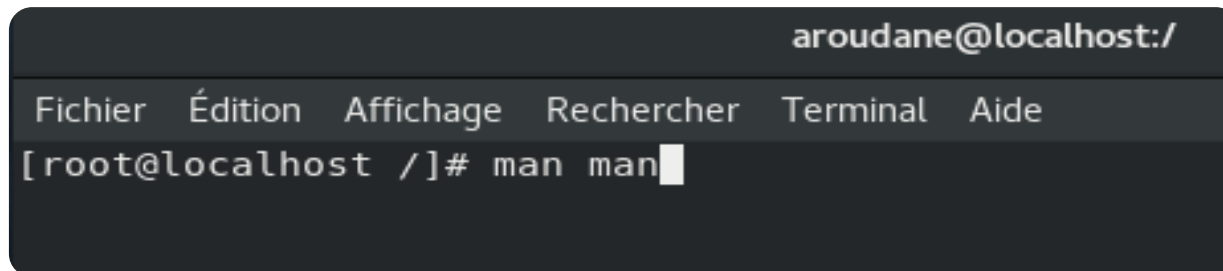
Pour connaître le contenu d'une variable:

```
[user@machine~]$ echo $PATH
```



## Manuel des commandes: La commande « man »

- La liste des commandes n'est pas exhaustive, seules les plus courantes et les plus adaptées à votre enseignement sont présentées.
- De la même façon, seules les options les plus courantes seront indiquées pour chaque commande.
- La commande **man** permet d'afficher une documentation souvent très complète sur la commande indiquée.
- La commande **man** est utile, en particulier, pour explorer les options possibles d'une commande.



```
aroudane@localhost:/  
Fichier  Édition  Affichage  Rechercher  Terminal  Aide  
[root@localhost /]# man man
```

# Caractéristiques d'UNIX

## Manuel des commandes: La commande « man »

```
aroudane@localhost:/  
Fichier  Édition  Affichage  Rechercher  Terminal  Aide  
MAN(1)          Utilitaires de l'afficheur des pages de manuel          MAN(1)  
  
NOM  
man - Interface de consultation des manuels de référence en ligne  
  
SYNOPSIS  
man [-C file] [-d] [-D] [--warnings[=warnings]] [-R encoding] [-L  
locale] [-m system[,...]] [-M path] [-S list] [-e extension] [-i|-I]  
[--regex|--wildcard] [--names-only] [-a] [-u] [--no-subpages] [-P  
pager] [-r prompt] [-7] [-E encoding] [--no-hyphenation] [--no-justifi-  
cation] [-p string] [-t] [-T[device]] [-H[browser]] [-X[dpi]] [-Z]  
[[section] page[.section] ...] ...  
man -k [options d'apropos] expression_rationnelle ...  
man -K [-w|-W] [-S liste] [-i|-I] [--regex] [section] term ...  
man -f [options de whatis] page ...  
man -l [-C fichier] [-d] [-D] [--warnings[=avertissements]] [-R enco-  
dage] [-L locale] [-P afficheur] [-r invite] [-7] [-E encodage] [-p  
chaîne] [-t] [-T[périphérique]] [-H[navigateur]] [-X[ppp]] [-Z] fichier  
...  
man -w|-W [-C fichier] [-d] [-D] page ...  
man -c [-C fichier] [-d] [-D] page ...  
man [-?V]  
  
Manual page man(1) line 1 (press h for help or q to quit)
```