Système utilisateur
.:: Système UNIX ::.

Un système d'exploitation

Exemples connus :

Windows

Linux

Qu'apportent-ils ?

Un système d'exploitation

Qu'apportent-ils ?

La possibilité d'utiliser l'ordinateur

Par une interface graphique ou plutôt une interface homme – machine.

Permet de lancer des programmes

Permet de copier/déplacer/... des fichiers

Permettre aux programmes de fonctionner quel que soit le matériel

Permet de 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 de l'ordinateur : 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

Un système d'exploitation

Pour aller plus loin

L'OS (Operating System) gère

- Les utilisateurs : gérer les droits d'accès aux fichiers, comme au matériel
- La standardisation des programmes : offre des interfaces de programmation simplifiées et standardisées. (exemple directX)

Un peu d'histoire

Premier système permettant d'effectuer des calculs de façon automatique : des Abaques réalisées au 16ème siècle par John Napier (Ecosse). Il s'agit de grille faisant la relation entre une valeur d'entrée et son cube. (comme une table de multiplication).

Au début du 17ème, Wilhem Schickard réalise une horloge à calculer utilisée en astronomie.

Un peu d'histoire

La première machine traitant des données entrées par l'utilisateur : Blaise Pascal au milieu du 17ème siècle avec la Pascaline (voir musée clermontois)

Jusque là il s'agit de systèmes mécaniques

Un peu d'histoire

Première moitié du 20 ème siècle : naissance de l'informatique telle que nous la connaissons

1939 – Atanasoff Berry Computer
1943 – Colossus (Turing) / Enigma (cryptage/décryptage)
1944 – Mark I, premier matériel programmable et donc premiers logiciels
1946 – Eniac (Von Neuman)

Un peu d'histoire

Seconde moitié du 20 ème siècle : la révolution du transistor

1947 – invention du transistor (remplace les lampes)

Avant, la logique qui compose les ordinateurs est réalisée par des lampes (comme une ampoule dont l'allumage est commandée par un signal électrique) – système encombrant, fragile et chauffe et grille facilement.

Après, cette logique est réalisée avec des semi-conducteurs : peu encombrant, consommant moins, utilisable sans préchauffage ...

En informatique, une lampe, ou un transistor sont simplement des interrupteurs commandés électriquement.

Un peu d'histoire

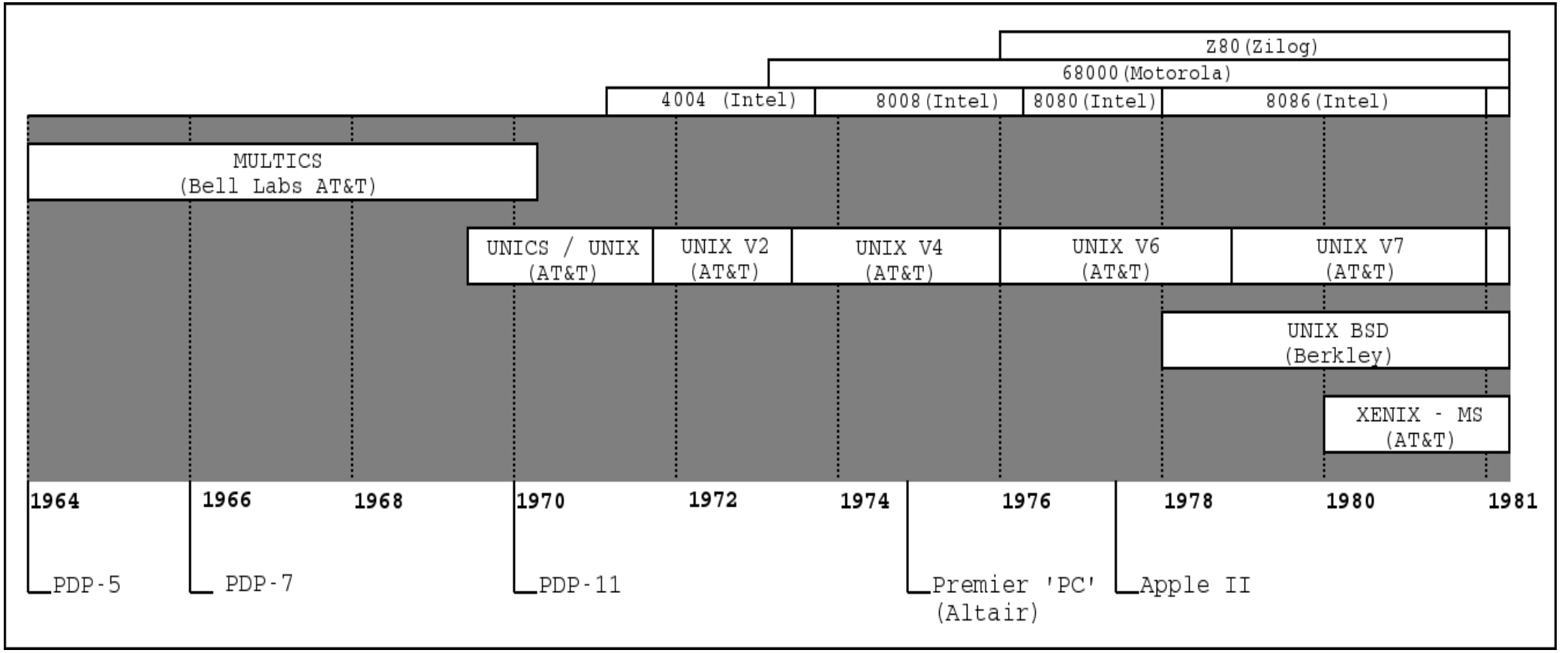
Seconde moitié du 20 ème siècle : la révolution du transistor

1960 – invention des circuits intégrés : regrouper dans un même composant plusieurs transistor pour réaliser une opération plus complexe.

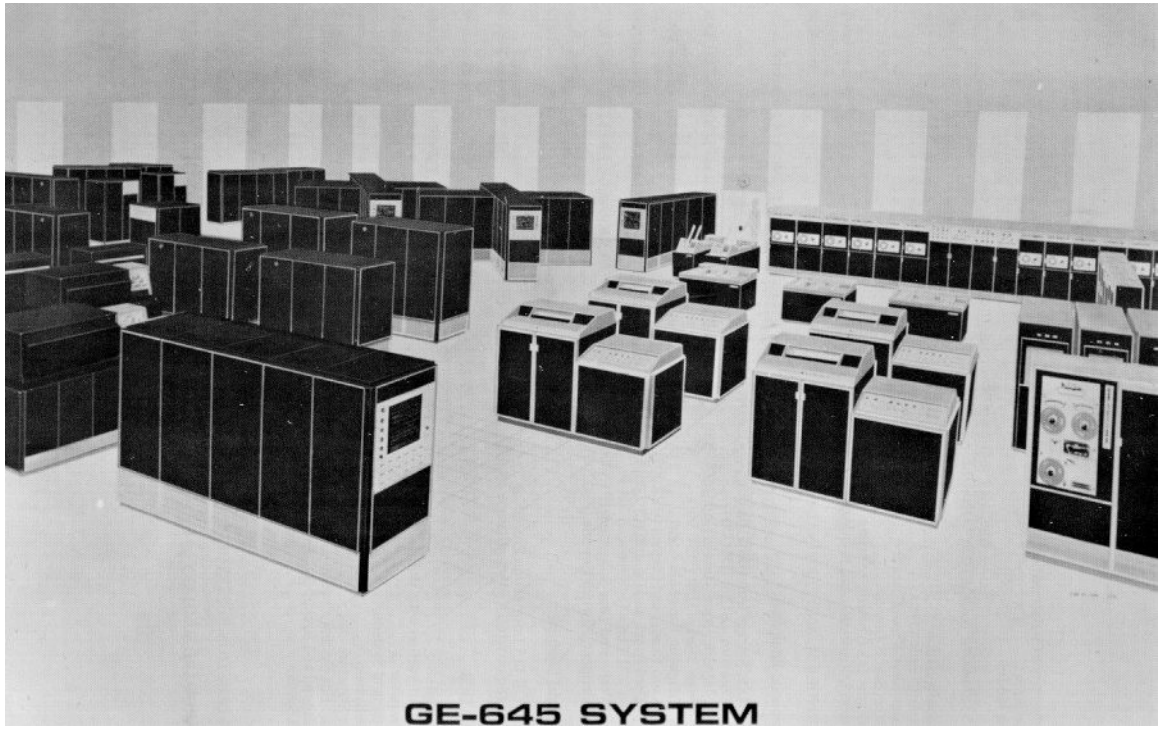
196x – les premiers langages “évolués” COBOL (1961) puis BASIC (1964)

1969 – premiers systèmes UNIX

Premiers systèmes UNIX et naissance de l'informatique personnelle

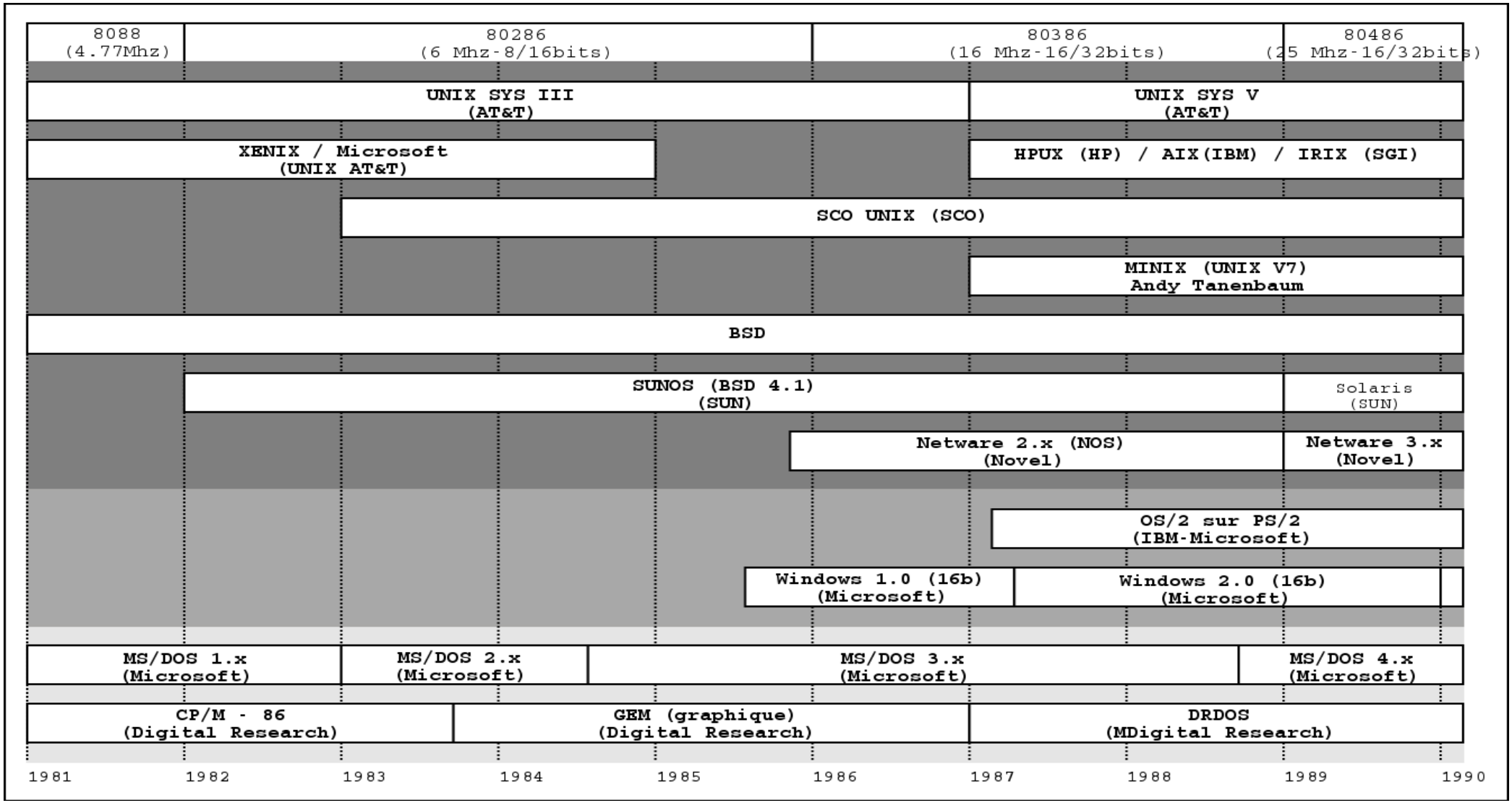


Machine sous MULTIX, une config de rêve : 16 Mo de RAM, 136Mo de HD et CPU à 430KHz

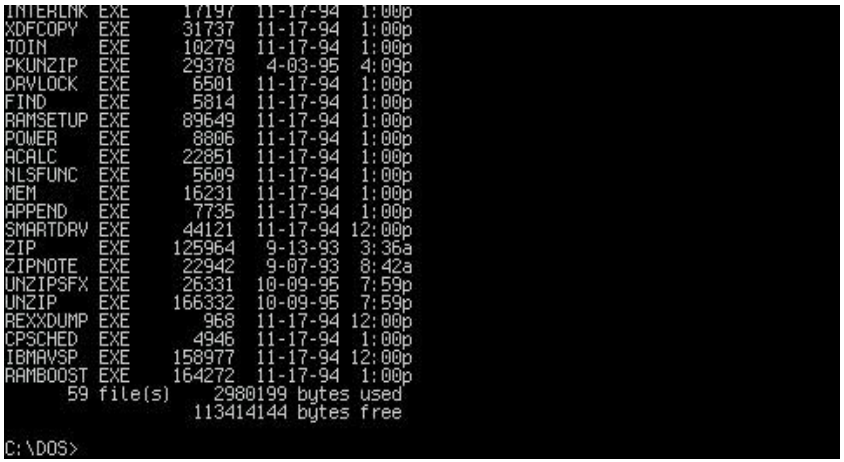


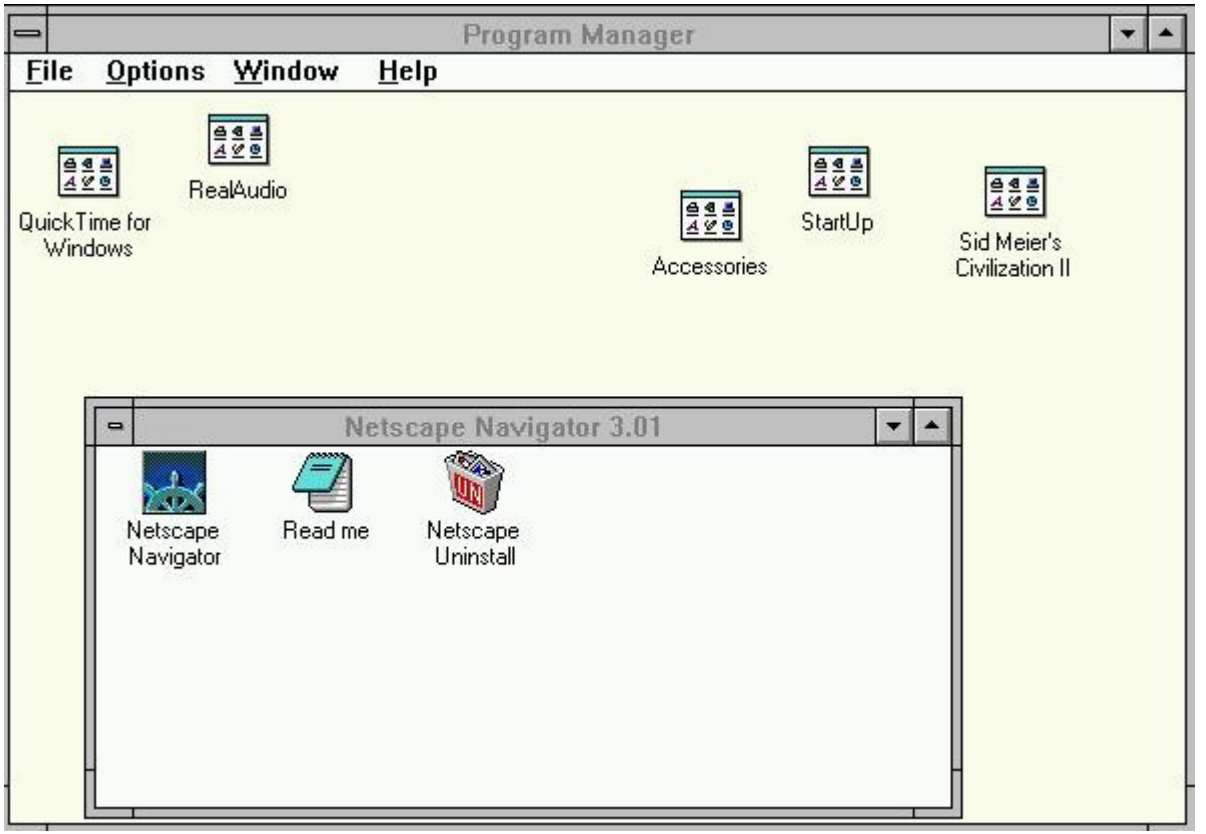
Machine TRS 80 sous XENIX, mode texte et caractères graphiques

Multiplication des UNIXs, les premiers systèmes graphiques

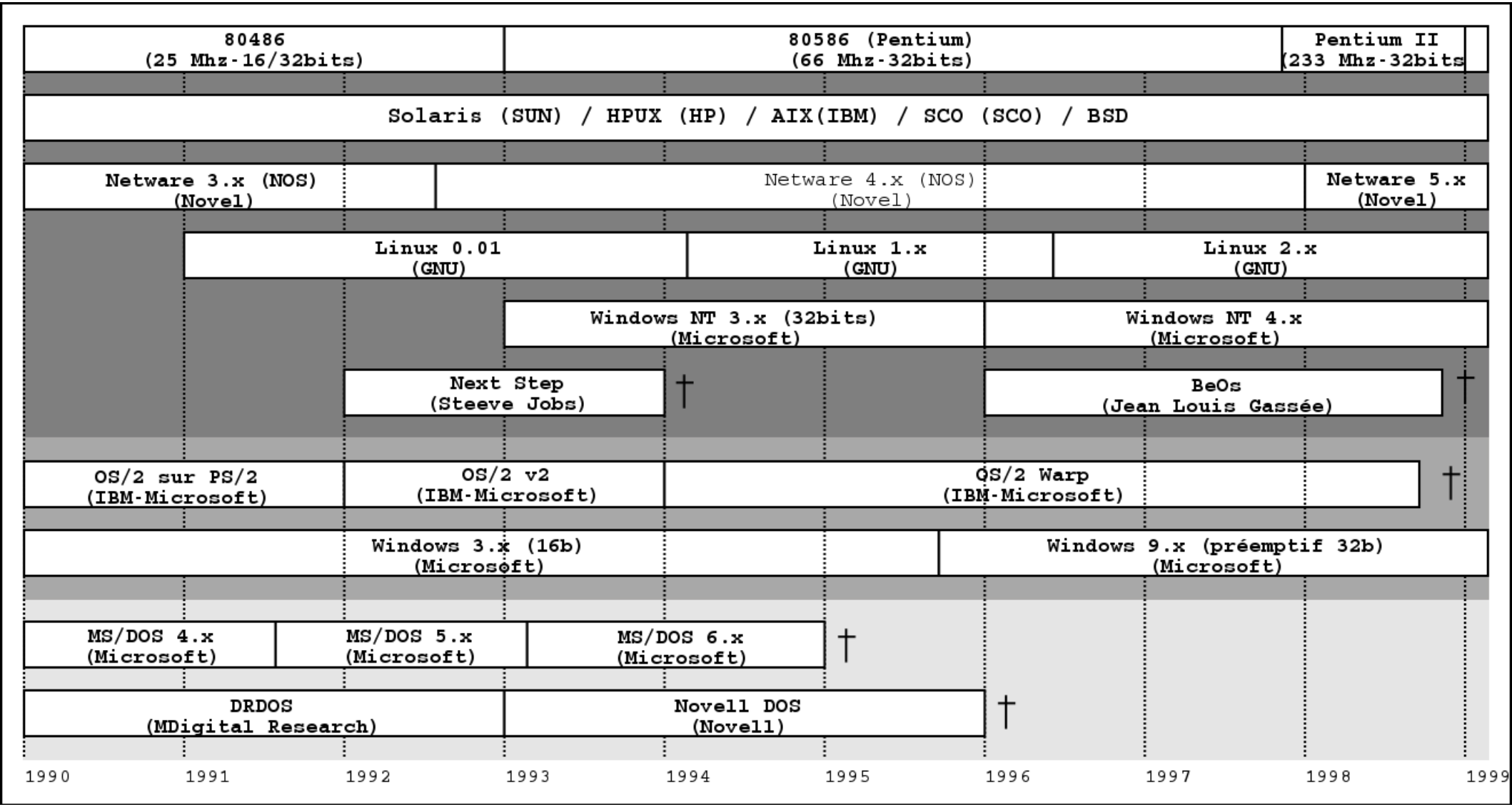


Windows complète le DOS et apparaît sur PC l'OS graphique

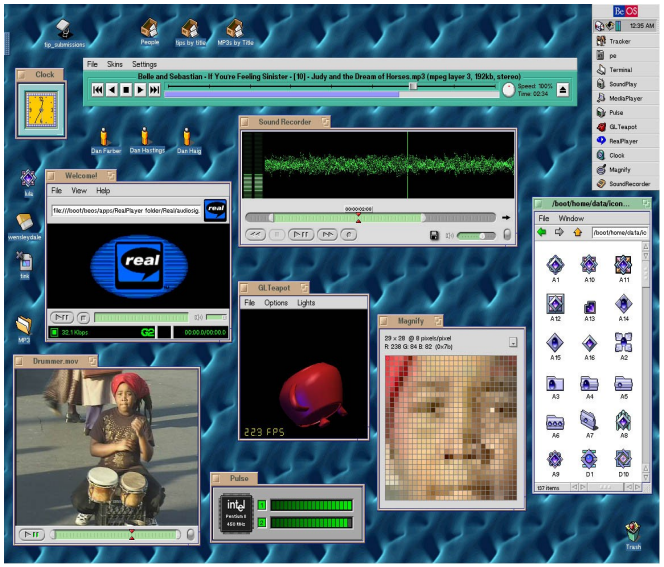


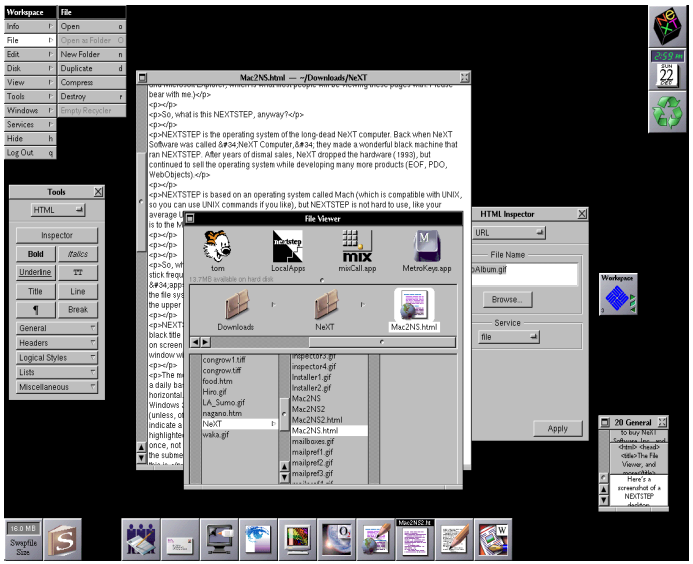


Nouvelles générations, OS 32bits et NOS (réseaux)

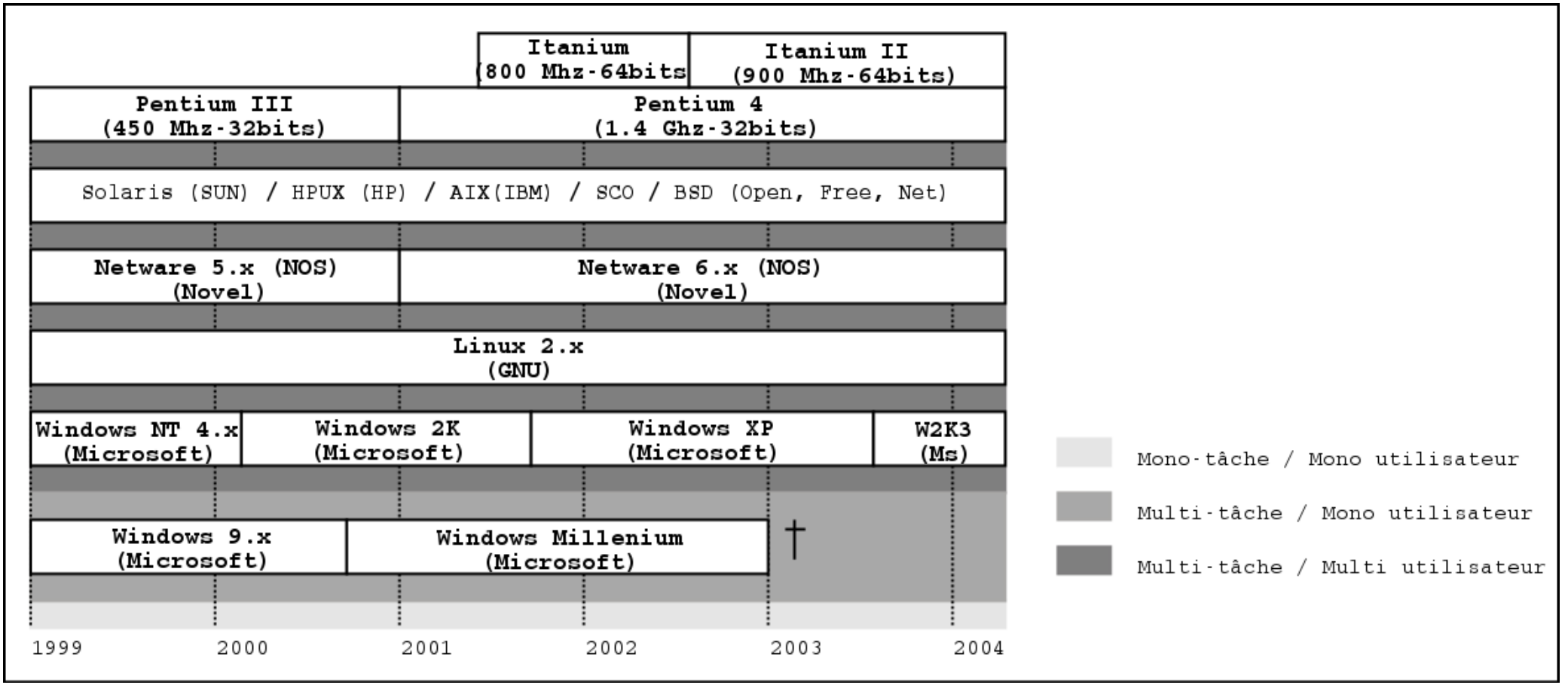


Be-OS / Next-Step / OS/2

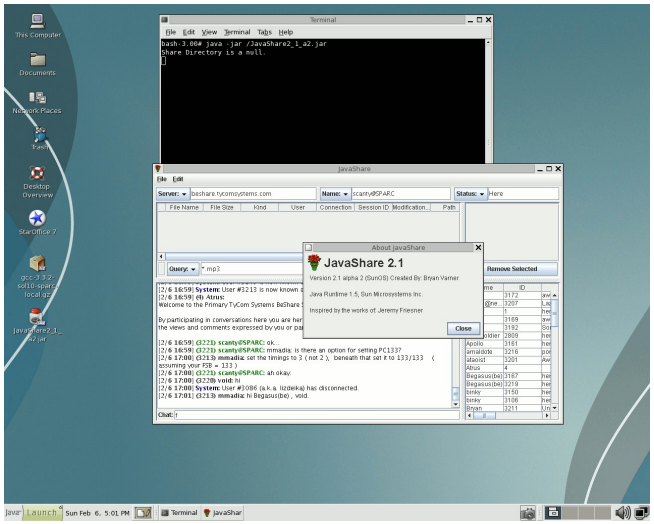


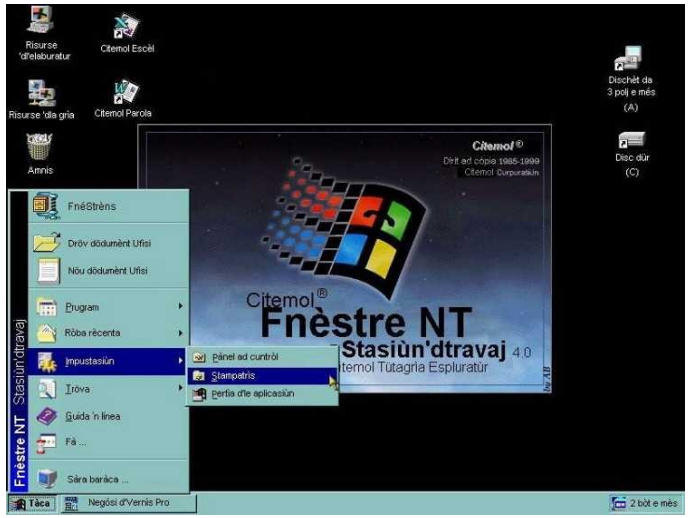


Systèmes actuels



Mac OS-X / SunOS / Win-NT





Comment sont nés les systèmes ?

Au tout départ il n'y a pas de système : les fonctions de l'ordinateur sont très limitées – type calculatrice. Ils n'exécutent qu'un unique programme pour lequel ils ont été spécifiquement conçu. (Pascaline, Enigma)

Il n'y a donc pas d'OS dans ces équipements, comme encore dans beaucoup de micro-contrôleur embarqués dans de l'électronique d'usage courant : TV, machine à laver, micro-onde ...

Comment sont nés les systèmes ?

La génération suivante est capable d'exécuter des programmes => fabrication non spécifique.

Les premiers systèmes sont avant tout des librairies d'accès au matériel pour éviter aux programmes de devoir les ré-écrire à chaque fois.
(un peu comme un dictionnaire vous évite de redéfinir chaque mots que vous utilisez)

Époque des cartes perforées. Encore aujourd'hui, des équipements comme certaines consoles de jeux fonctionnent sur ce type : à l'allumage un “loader” se charge de lire le jeu sur un DVD ou une carte mémoire.

Comment sont nés les systèmes ?

Évolution suivante : le système est conservé en mémoire après chaque exécution de programme et enchaine les programmes (Jobs) les uns à la suite des autres (traitement Batchs) et JCL (Job Control Language)

Jusque là, les systèmes ne gèrent ni les utilisateurs, ni le partage du processeur entre plusieurs programmes. Les ordinateurs ne sont pas “interactifs” ce sont de grosses calculatrices à qui l'on passe un lot de données à traiter.

Comment sont nés les systèmes ?

Apparaissent ensuite les terminaux :
plusieurs personnes peuvent utiliser l'ordinateur, il sert par exemple à consulter un stock, prendre des commandes...

S'il est utilisé par plusieurs personnes, alors il doit pouvoir partager le processeur entre plusieurs tâches.

Naissance dans ce but d'UNIX dans la fin des années 60

De multiples UNIX

UNIX est né dans le Bells Labs (AT&T) sur un PDP 11. Il est au départ mono-utilisateur mais multi-processus.

Il devient multi-utilisateurs en 1971 et ré-écrit en C (Dennis Ritchie, Brian Kernigham) en 1973. Le code source et **donné** aux universités et donne naissance aux branches BSD (Berkeley)

En 1977, le code est **donné** aux entreprises.

Une famille nombreuse

Chaque entreprise crée sa version propre d'UNIX :

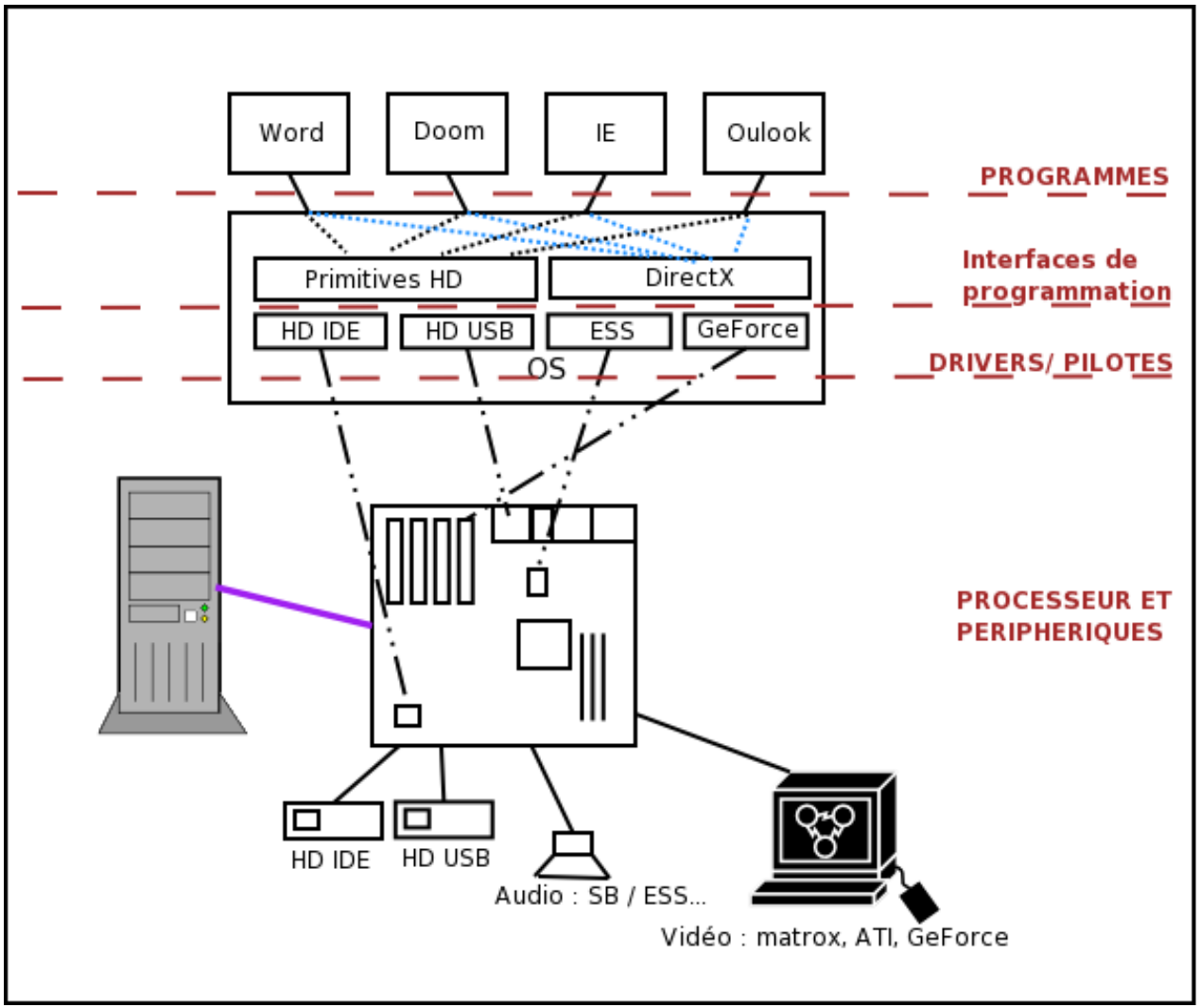
- IBM avec AIX depuis 1990
- Sun avec Solaris
- HP avec HP-UX depuis 1986
- DEC avec Ultrix
- Silicon Graphics avec IRIX
- Novell avec Unixware
- SCO avec SCO-UNIX depuis 1979
- Compaq avec Tru64 UNIX

... dont certains membres sont libres

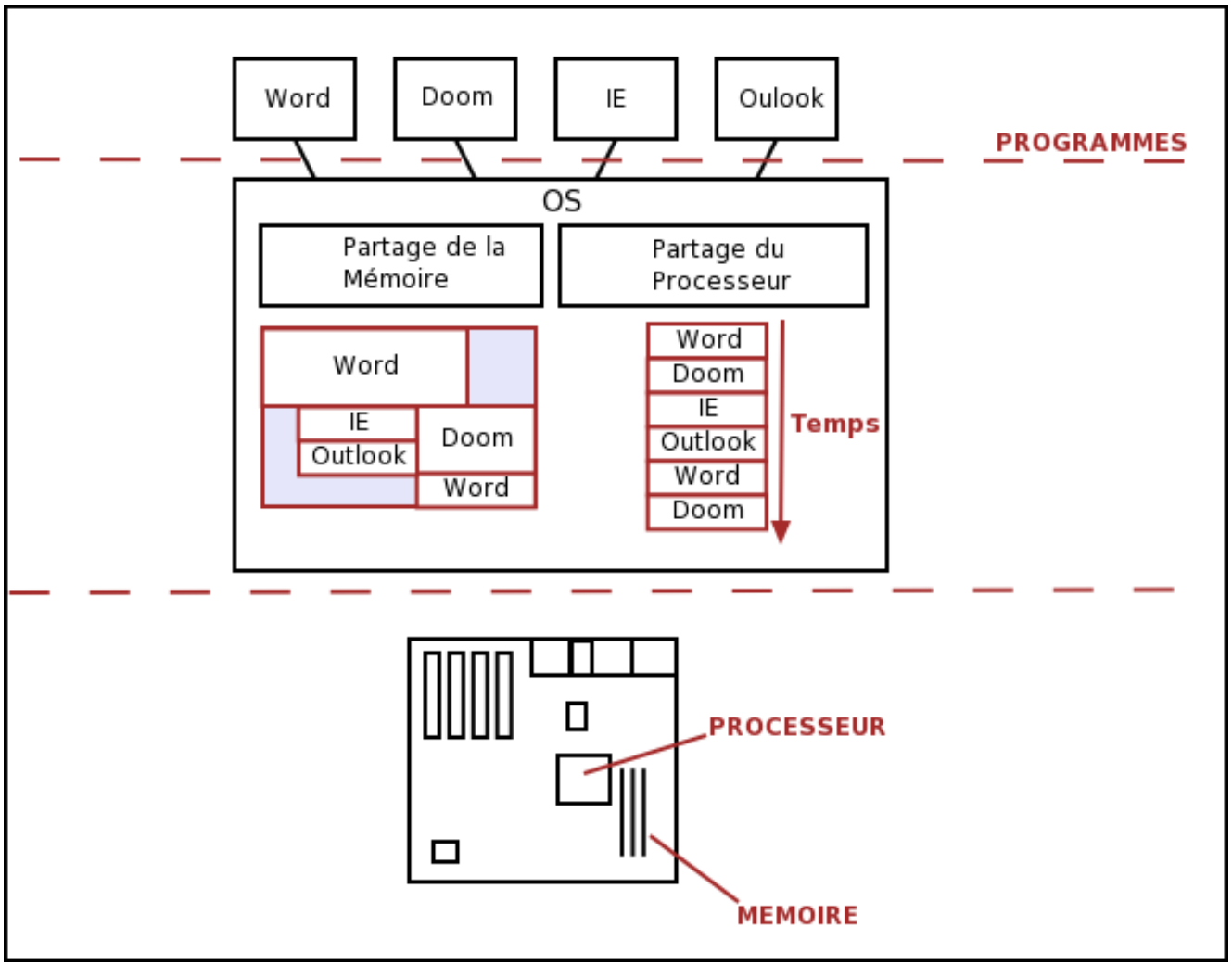
Les **free** – software :
Au tout début : MINIX (Tannenbaum)
La famille des BSD – le code est ouvert et libre, il peut être copié et le résultat de cette copie peut être vendu sans distribuer les modifications apportées. openBSD, netBSD, freeBSD, Mac-OSX

Les **ouverts** – GNU :
Ceux dont les modifications doivent être publiées.
Linux, créé de rien par Linux Torwalds en 1991 (debian, opensuse, redhat, mandriva, ubuntu... tous ont le même noyau – coeur du système)
Hurd, le noyau de GNU (micro-noyau) qui n'est pas un UNIX

Principe d'abstraction matériel



Principe de partage du matériel



Aux utilisateurs la simplicité du mode graphique

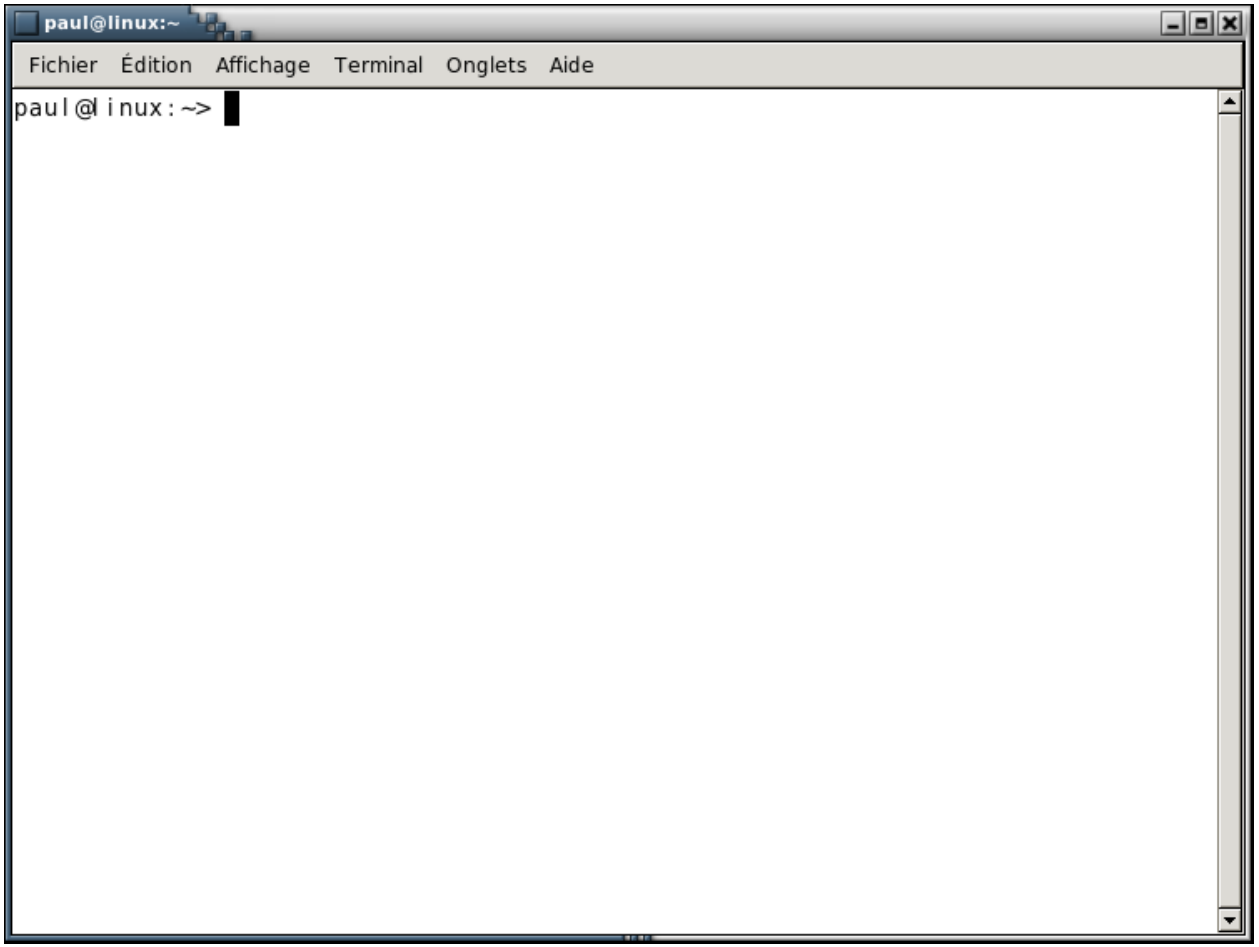
Aux informaticiens la puissance du mode texte !

Le mode texte ou SHELL

Le shell est plus rapide et plus stable que les manipulations graphique. Il permet d'automatiser des taches, de programmer des séquences itératives et conditionnelles.

Un shell est une fenêtre dans laquelle nous allons saisir des commandes. Ces commandes sont interprétées par le système.

Le mode texte ou SHELL



Les commandes

Le SHELL exécute des commandes qui peuvent être

Des instructions internes au shell – if then else, while, set... C'est donc un langage de programmation

Des programmes externes (ls, ps, openoffice, firefox)

Des alias : alias ll='ls -l' (raccourcis vers une autre commande)

Les commandes d'aide

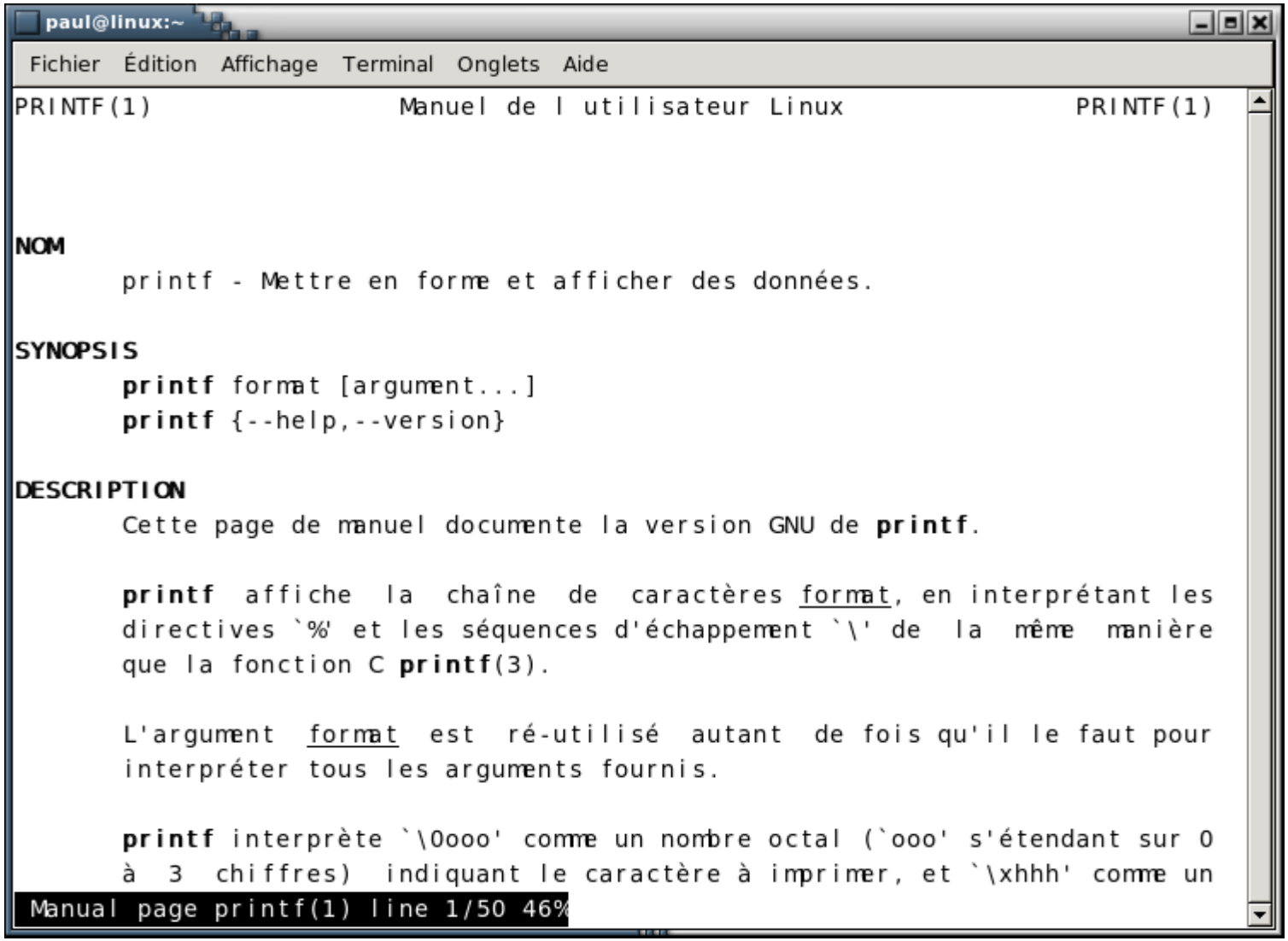
Une commande des plus importantes : **man**

=> permet d'afficher la documentation d'une commande

=> comme il existe des homonymes, la documentation est rangée par section
1 pour les commandes
3 pour les fonctions (programmation)

Exemple :
=> le manuel de la commande printf man 1 printf
=> le manuel de la fonction C printf man 3 printf

Les commandes d'aide



Les commandes d'aide

Pour trouver quelle commande utiliser pour effectuer une opération particulière : **apropos**

Exemple :
=> comment copier des fichiers ?

```
paul@linux:~> apropos "copier des fichiers"
cp (1)              - Copier des fichiers.
install (1)         - Copier des fichiers et positionner leurs attributs.
```

Les commandes d'aide

Accéder à l'aide embarquée dans un programme :
=> Passer à la commande une option qui peut être par exemple –help

Exemple :
=> comment fonctionne la commande de copie de fichiers cp ?

```
paul@linux:~> cp --help
Usage: cp [OPTION]... SOURCE CIBLE
  ou:  cp [OPTION]... SOURCE... RÉPERTOIRE
  ou:  cp [OPTION]... --target-directory=RÉPERTOIRE SOURCE...
Copier la SOURCE vers la DESTINATION, ou de multiples SOURCES vers
un RÉPERTOIRE.
[...]
```

L'organisation des fichiers

Un fichier est une suite de caractères, lisibles ou non (texte ou binaire) qui sont regroupés ensemble. Un fichier peut être un programme, un document, une musique, un film ...

Un fichier est identifié par un nom, une extension peut permettre d'identifier rapidement le type d'un fichier (monCour.doc est sans doute un fichier WORD) toutefois, sous UNIX ceci n'est pas une contrainte : l'habit ne fait pas le moine.

La commande **file** permet de connaitre le type réel d'un fichier à partir de son contenu et non de son nom.

Exemple :
paul@linux:~/tempo> **file monCours.doc**
monCours.doc: GIF image data, version 89a, 100 x 76

L'organisation des fichiers

Les fichiers sont rangés dans des répertoires.

=> Sous Windows l'organisation est très liée à l'organisation physique de la machine

```
Bureau
    |-Poste de travail
    |        |-Disque C
    |        |     |-   Windows
    |        |     |-   Mes Documents
    |        |     |-   Program Files
    |        |-Disque D
    |        |-CDRom
    |-Corbeille
```

L'organisation des fichiers

=> Sous UNIX, l'organisation est logique à partir d'un répertoire appelé « racine » dont le nom est « / »

```
/                                   Répertoire racine
    |- home                         Répertoire des utilisateurs
    |      |-paul
    |      |-sebastien
    |      |-pierre
    |- dev                          Répertoire des périphérique
    |      |-sound
    |      |-cdrom
    |-etc                           Répertoire de configuration
    |-mnt                           Répertoire de montage des périphériques
    |      |-cdrom
    |      |-clé-usb
    |-...
```

L'organisation des fichiers

=> Organisation sous forme d'un arbre, on parle donc d'arborescence.

=> A chaque noeud il est possible d'attacher une partition ou un périphérique (faire le lien entre l'organisation logique et l'organisation physique). Ont dit aussi monter un périphérique.

=> Le chemin d'un fichier est « l'adresse » de ce fichier dans l'arborescence :

=> Chemin absolu : adresse depuis le répertoire racine

```
paul@linux> cat /home/paul/cours/monCours.doc
```

=> Chemin relatif : adresse depuis le répertoire courant

```
paul@linux> cat ./cours/monCours.doc
paul@linux> cat cours/monCours.doc
paul@linux> cat ~/cours/monCours.doc
```

=> Le répertoire « . » est le répertoire courant ; « .. » le répertoire père.
=> Le répertoire de l'utilisateur courant (home) est aussi appelé « ~ ».

L'organisation des fichiers

Sous UNIX, tout est fichier : périphériques (scanner, carte son, écran), répertoire, socket de communication ... Pour les distinguer des uns des autres on utilise un type :

- d pour les répertoire
- \- pour les fichiers ordinaires (ou réguliers)
- l pour les lien symboliques
- c pour les périphériques caractères
- b pour les périphériques bloc
- ...

Les informations concernant un fichier sont affichées par la commande « **ls** ».

=> Le type du fichier
=> Les droits du fichier : qui peut effectuer quelle opération (r/w/x)
=> Le nombre de lien sur ce fichier
=> L'utilisateur propriétaire
=> Le groupe propriétaire
=> La taille en octet
=> La date de dernière modification
=> Le nom du fichier

L'organisation des fichiers

Exemple :

paul@linux> ls -l support_cours_2005.sxi

-rw-r--r-- 1 paul users 1686700 2005-07-31 15:00 support_cours_2005.sxi

L'utilisateur « paul » peut lire et écrire ce fichier
Les utilisateurs du groupe « users » peuvent seulement le lire
Les autres peuvent aussi seulement le lire

C'est un fichier régulier de 1,6 Mo environ modifié pour la dernière fois le 31 Juillet 2005 à 15:00.

L'organisation des fichiers

=> L'organisation physique est totalement indépendante :

```
/                                   Peut être sur le premier disque dur, partition 1 (C:)
    |- home                              Peut être sur le premier disque dur, partition 2 (D:)
    |     |-paul
    |     |-sebastien                 Peut être sur un disque réseau (\\serveur\sebhome)
    |     |-pierre
    |- dev
    |     |-sound
    |     |-cdrom
    |-etc                                 Peut être sur le second disque dur ...
    |-mnt
    |     |-cdrom                       Correspond à n'importe quel lecteur de cdrom
    |     |-clé-usb                     ...
    |-...
```

Monter un périphérique

=> Pour connecter un périphérique physique (clef usb, disquette) à un lecteur logique et ainsi lire / écrire sur le périphérique.

=> la commande « **mount** » permet d'effectuer cette opération et lit les paramètres nécessaires dans le fichier de configuration /etc/fstab

=> mount /floppy va associer le lecteur de disquette au répertoire /floppy

=> après utilisation, avant de retirer le périphérique, il doit être déconnecté (démonté) à l'aide de la commande « **umount** ».

Les variables SHELL

=> Le shell permet de mémoriser des données dans des variables, celles-ci sont utilisées par exemple pour paramétrer le shell.
PATH : mémorise les chemins par défaut où chercher les programmes
PS1 : définir le prompt de l'utilisateur
LANG : définir la langue de l'utilisateur ex fr_FR pour le français de France.

=> Pour afficher une variable on utilise la commande « **echo** » et les variables sont identifiée par le caractère « **$** » qui précède.

```
paul@linux> echo $LANG
fr_FR.UTF-8
```

Plusieurs SHELL

=> Le shell est un programme ; il existe donc plusieurs SHELL comme il existe plusieurs interface graphiques sous UNIX

- sh
- bash
- tcsh
- ksh
- ...

=> Tous ont leur petits plus, à chacun sa préférence ; mais la programmation de scripts peut être spécifique à un shell ou un autre.

=> Pour les interfaces graphiques :

- Kde, Gnome, WindowMaker (AfterStep), fvwm2 (win95), iceWM (léger)

Vos premières commandes

=> Afficher du texte à l'écran
echo bonjour - affiche « bonjour » à l'écran

=> Afficher le contenu d'un fichier à l'écran
cat nomDuFichierAAfficher

=> Afficher le contenu d'un fichier page par page
more nomDuFichierAAfficher

=> Afficher le contenu d'un fichier en mode interactif
less nomDuFichierAAfficher

Vos premières commandes

=> Connaître le répertoire courant

pwd

=> Naviguer dans les répertoires

cd - pour Change Directory

on ajoute en paramètre le répertoire où l'on souhaite se rendre

- en relatif (par rapport au répertoire courant)

cd ./Documents/mp3 => ./ indique que l'on part du répertoire courant

- en absolu (par rapport à la racine)

cd /tmp => / indique que l'on part de la racine

=> Afficher le contenu d'un répertoire

ls

Vos premières commandes

=> Créer un alias

alias l='ls' – de cette façon on utilisera l plutot que ls (plus court)

=> Supprimer une alias

unalias l – supprime l'alias l créé précédemment

=> Créer une variable

MaVariable=/tmp

=> Afficher une variable

echo $MaVariable

=> Composer une variable

PATH=$PATH:$MaVariable