IPT : Cours 1

Informatique : le hard et le soft (2h)

MPSI-Schwarz : Prytanée National Militaire

Cours inspiré du livre
"Informatique pour Tous en CPGE" - ed EYROLLE

16 janvier 2016

Il s'agit dans ce chapitre :

1. De définir ce que l'on désigne habituellement sous le terme d'*ordinateur*,
2. De voir que l'utilisation d'un ordinateur se fait au travers d'un *système d'exploitation*,
3. De montrer que l'utilisateur d'un ordinateur peut construire ses propres programmes à l'aide d'un *environnement de développement*.

1 Qu'est-ce qu'un ordinateur ?

Un smartphone, une tablette numérique, un ordinateur portable ou de bureau, tous entrent dans la catégorie des ordinateurs ? Pourquoi ?

Ces différents appareils ont en commun les caractéristiques suivantes :

- Ils ont besoin d'une source d'énergie
- Ils reçoivent des informations de la part de l'utilisateur par l'intermédiaire :
  (a) D'un clavier (b) D'un écran tactile (c) D'une souris (d) D'un haut-parleur (e) D'un réseau (wifi, filaire...)
- Ils renvoient des informations par l'intermédiaire :
  (a) D'un écran (b) D'un haut-parleur (c) D'un réseau

Une automobile ou un thermostat vérifient également ces caractéristiques...
Les considère-t-on pour autant comme des ordinateurs ? Non car ils différent :

1. dans leurs objectifs : l'automobile a pour but de mouvoir des passagers et non de traiter une information
2. dans leur spécificité : le thermostat traite bien une information mais a un objectif trop particulier.

Quelle définition alors donner au mot *ordinateur* ?

1.1 L'ordinateur, une machine universelle

La première idée derrière la notion d'ordinateur est que l'information peut-être codée et son traitement peut être automatisé. Pensez par exemple au métier à tisser mis au point par Jacquard en 1801, à un orgue de barbarie ou encore aux premières machines à calculer mécaniques (Pascal 1642). Les exemples précédents ont cependant le gros défaut d'être trop spécialisés. Au XXème siècle, on s'est donc demandé s'il n'était pas possible de construire une machine polyvalente pouvant recueillir toutes sortes d'informations et de les traiter de toutes les façons envisageables. Turing imagine alors un modèle de machine qui possède les caractéristiques suivantes :

1. Il suppose l'existence de machines pouvant lire des données (sur un ruban...), les traiter et renvoyer des données issues du traitement.
2. Il démontre alors qu'il existe une machine universelle capable de remplacer n'importe laquelle des machines précédentes et prétend que cette machine est alors capable de résoudre tous les problèmes pour lesquels une méthode de résolution existe. Selon Turing, une machine n'a plus besoin d'être construite pour exister, il suffit qu'elle soit codée et entrée comme données dans cette machine universelle. Vous aurez ici sans doute reconnu le concept de *programme*.
3. Il démontre cependant que cette machine universelle reste incapable de résoudre certains problèmes comme la terminaison d'un traitement ou la correction de certaines propriétés mathématiques.

Définition 1 : **Ordinateur**
De nos jours, on considère qu'un ordinateur est la réalisation concrête d'une machine de turing, à savoir, une machine traitant des informations et capable en principe de prendre comme donnée n'importe quel algorithme et de l'exécuter.

Remarque 1. Certains appareils comme une box internet ou l'ordinateur de bord d'une voiture s'apparentent à des ordinateurs dans la mesure où ils peuvent être programmer. Cependant, seul le constructeur a la possibilité de changer les programmes et ils ne sont donc pas considérés comme des ordinateurs.

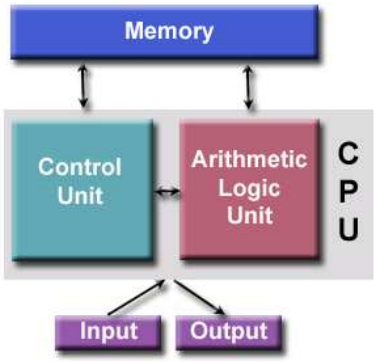
Il ne reste plus qu'à passer à la conception d'une telle machine...

1.2 Architecture des ordinateurs

Les ordinateurs actuels sont tous construits autour du même modèle architectural théorique : l'architecture de Van Newmann.

Un machine suivant l'architecture de Neumann est constituée :

1. d'une *mémoire vive* : la RAM (mémoire de travail)
2. d'un *processeur* (CPU pour Central Processing Unit) conceptuellement décomposé en :
   - un *registre*,
   - une *unité de contrôle* (UC) et
   - une *unité de calcul arithmétique et logique* (UAL)
3. de *périphériques* (pour les Entrées et Sorties d'informations)
4. de canaux de communication (les *bus*) entre la mémoire, le processeur et les périphériques



Architecture de Von Neumann

• La mémoire vive ou RAM (Ramdom Access Memory) :

Il s'agit d'une suite de chiffres codés en binaire (*bits*) et regroupés en paquets de 8 bits (*octets*) et en *mots mémoire* en général de 32 ou 64 bits. Il existe au moins 3 types de mémoire vive : SD, SSD et DDR-SDRAM (d'accès plus rapide).

Barettes de mémoire vive

Elle possède les caractéristiques suivantes :

1. un mot dans la mémoire peut aussi bien représenter une instruction dans un programme, qu'un entier ou une couleur selon l'interprétation que lui donne l'utilisateur.
2. elle est inerte dans le sens où elle sert juste de support de stockage de l'information.
3. tout mot mémoire possède une adresse (codé sous la forme d'un entier) lui permettant d'être lu rapidement par le microprocesseur.

Remarque 2. Lorsque la mémoire vive arrive à saturation en raison d'une utilisation instantanée intensive de l'ordinateur, une partie du disque dur peut-être utilisée par le microprocesseur : on appelle cela le *swap*. Dans ce cas, le temps d'exécution des instructions est beaucoup plus lent (de l'ordre de 1000 fois).

• Les périphériques :

Il s'agit de mémoires additionnelles que l'ordinateur peut lire ou encore écrire (disque dur - DVD - CD - clé USB ...). Ils diffèrent de la RAM en particulier par le fait qu'ils ne sont pas nécessairement inertes et peuvent ainsi eux-même obéïr à des instructions (imprimante...)

Disque dur

Remarque 3.

1. Pour garantir une plus grande sécurité, les disques durs sont en général *partitionnés* en un minimum de deux parties.
   - le "disque C : \" (dîte *partition active*) sur lequel est enregistré le système d'exploitation utilisé et
   - le "disque D : \" sur lequel on trouve les données de l'utilisateur et les logiciels installés.

2. On trouve souvent également une *partition de restauration* qui permet de remettre le disque dur dans l'état où il était au moment de la première utilisation de l'ordinateur. Cette manipulation engendre donc la perte de tous les logiciels et fichiers installés par l'utilisateur. Les utilisateurs de Microsoft Windows préféreront donc en général utiliser les *points de restauration* proposé par leur système d'exploitation. Il est toutefois conseillé de créer sa propre *image de restauration* (ce que permet la plupart des systèmes d'exploitation actuels) et de l'enregistrer (environ 8 go) sur un autre périphérique (DVD - DD externe - clé USB - carte SD). Son installation se fait alors à l'aide d'un *utilitaire* du type "Ghost 11.5" qui permet des copies de DD.

3. Le disque dur possède aussi une *zone de catalogue* dans laquelle se trouvent l'ensemble des adresses des différents logiciels et fichiers contenu dans le disque dur. Un *formatage de bas-niveau* consiste à simplement supprimer ce catalogue tandis qu'un formatage complet remet à 0 tous les bits du dusque dur et fait donc disparaître toutes les données.

• Le processeur (CPU) :

Le *processeur* (ou micro-processeur) est considéré comme le coeur de l'ordinateur. C'est lui qui attribut des tâches simples aux périphériques et à la RAM tandis qu'il exécute les tâches les plus compliquées. Il est responsable de l'exécution d'un programme.

Micro-processeur (CPU)

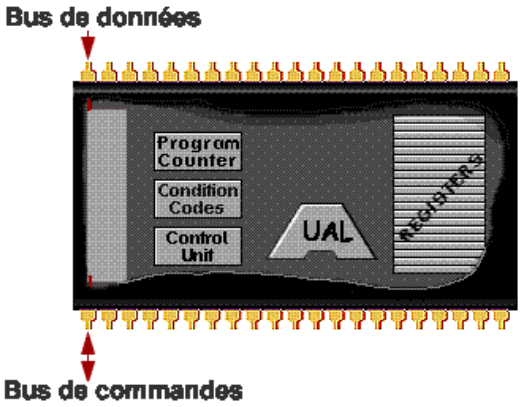
1. Les *registres* : le micro-processeur dispose d'une toute petite mémoire d'au plus quelques centaines de mots appelés des *registres* et habituellement notés R_1, R_2, ...

2. L'*unité de calcul arithmétique et logique* (UAL) :
   C'est elle qui effectue les opérations algébriques usuelles (somme, différence, produit, rapport) sur deux des nombres inscrits dans deux des registres pour stocker le résultat obtenu dans un troisième. Elle peut également effectuer des opérations logiques de base (disjonction, conjonction, négation).

3. L'*unité de contrôle* :
   Via le bus, l'unité de contrôle accède à la RAM. Elle peut par exemple aller chercher l'information contenue à l'adresse R_1 dans la RAM et la stocker dans le registre R_2, ou inversement, copier le contenu du registre R_2 dans la RAM à l'adresse indiquée dans R_1.

   Les instructions que doit suivre l'unité de contrôle sont elles aussi inscrites dans la RAM. Pour les exécuter, l'unité de contrôle utilise un registre particulier appelé PC (Program Counter) et exécute en boucle la séquence d'actions suivantes :

   (a) *Lire instruction* : Aller lire le mot stocké à l'adresse mémoire inscrite dans le registre PC et le stocker dans un registre spécial appelé IR (Registre d'Instruction). Les instructions sont des mots mémoires codés en binaire que l'on appelle le *langage machine*.
   (b) *Incrémenter PC* : Ajouter 1 au mot stocké dans le registre PC et enregistrer le résultat dans ce même registre.
   (c) *Décoder instruction* : Décoder l'instruction contenue dans IR, c'est à dire, reconnaître s'il s'agit d'une opération arithmétique et logique, d'un accès à la mémoire vive ou d'un branchement (voir plus loin).
   (d) *Exécuter instruction* : Exécuter l'instuction décodée.



On constate que les adresses des différentes instructions du programme à exécuter sont successives sauf lorsque le registre PC contient une instruction de branchement. Une telle instruction rompt la lecture séquentielle en indiquant une nouvelle adresse pour l'instruction suivante. Les branchement sont utilisés lorsqu'il s'agit de répéter une série d'instructions ou lors de l'exécution d'une alternative selon la valeur d'un résultat donné.

Remarque 4. La description précédente est en fait un peu simpliste car :

1. si le registre PC stocke bien des mots, en revanche les adresses sont stockées sur uniquement un octet. A chaque itération, le registre PC stocke donc 4 ou 8 adresses (selon qu'il s'agit d'un CPU 32 ou 64 bits) et est incrémenté soit de 4 soit de 8.

2. à chaque itération, pour gagner du temps, le processeur exécute les 4 ou 8 instructions de façon simultanée.

3. il existe deux structures usuelles de microprocesseurs : l'architecture CISC (adapté au traitement d'instructions complexes, plus cher et moins rapide) et l'architecture RISC (traite des instructions simples, moins cher et plus rapide). Les compilateurs utilisés lors de la création de programmes sont compatibles avec l'une ou l'autre de ces deux architectures : pas les deux !

4. De plus en plus, pour gagner en rapidité, les ordinateurs fonctionnent avec soit plusieurs microprocesseurs, soit des multi-processeurs qui incorporent 2 (double-core) ou 4 (quatre-core) microprocesseurs. Notons de plus qu'en moyenne, la vitesse de calcul des microprocesseurs double en moyenne tous les 18 mois.

1.3 Inconvénients de l'architecture de Von Neumann

Si l'architecture de Newmann se caractérise par sa grande souplesse, elle présente en revanche trois gros inconvénients :

1. **Une exécution séquentielle :** L'architecture de Von Neumann impose une exécution séquentielle des instructions. Avec ce modèle, il n'est donc pas possible d'effectuer différents travaux en parallèle.

2. **Un goulet d'étranglement :** Alors que le microprocesseur peut exécuter des instructions très rapidement, celles-ci sont de toute façon ralenties par la vitesse de transfert des informations dans les bus de communication.

3. **Faible robustesse :** Le fait que les données et les programmes soient mélangés engendre une fragilité du modéle de Neumann : si une donnée est enregistrée à l'emplacement d'un programme, le comportement de l'ordinateur peut devenir complètement incohérent.

Ainsi, afin d'optimiser la performance des ordinateurs, les concepteurs d'ordinateurs ont été amenés à s'éloigner du modèle théorique décrit par Von Neumann. En particulier, pour éviter de surcharger le micro-processeur :

1. les instructions relatifs à l'affichage de données à l'écran sont prises en charge par *carte graphique* qui contient son propre micro-processeur et sa propre mémoire vive.

2. un système appelé DMA (Direct Memory Access) permet aux périphériques de lire et écrire des données directement dans la RAM, sans passer par le micro-processeur.

1.4 Volatilité de la RAM

Les informations de la RAM disparaissent lors de la moindre coupure d'électricité : on dit que la RAM est une mémoire *volatile*. Afin de conserver sur le long terme les informations stockées, on a recourt à deux types de mémoire non volatile :

1. **La mémoire morte (ROM ou BIOS) :** Il s'agit d'un type de mémoire installée dans les ordinateurs permettant un accès rapide en lecture mais ne permettant pas d'accès en écriture. Elle contient un programme (appelé un *firmware*) mis en place lors de la conception de l'ordinateur que l'on appelle le BIOS (Basic Input/Output System) et qui est le premier programme à être exécuté lors de l'allumage de l'ordinateur.

La mémoire morte (ROM ou BIOS)

Il indique à l'ordinateur les instructions à effectuer lors de sa mise sous tension avec en particulier :

(a) la ré-initialisation de tous les composants de la carte mère.

(b) l'identification de l'ensemble des périphériques connectés à la carte mère et la vérification de leur bon fonctionnement

(c) le démarrage du système d'exploitation présent sur le premier périphérique disponible.

Remarque 5. Le BIOS est maintenant installé sur une *mémoire flash* qu'il est possible de modifier à l'aide d'un logiciel appelé un *firmware*. L'opération de mise à jour du BIOS s'appelle *flasher le BIOS*.

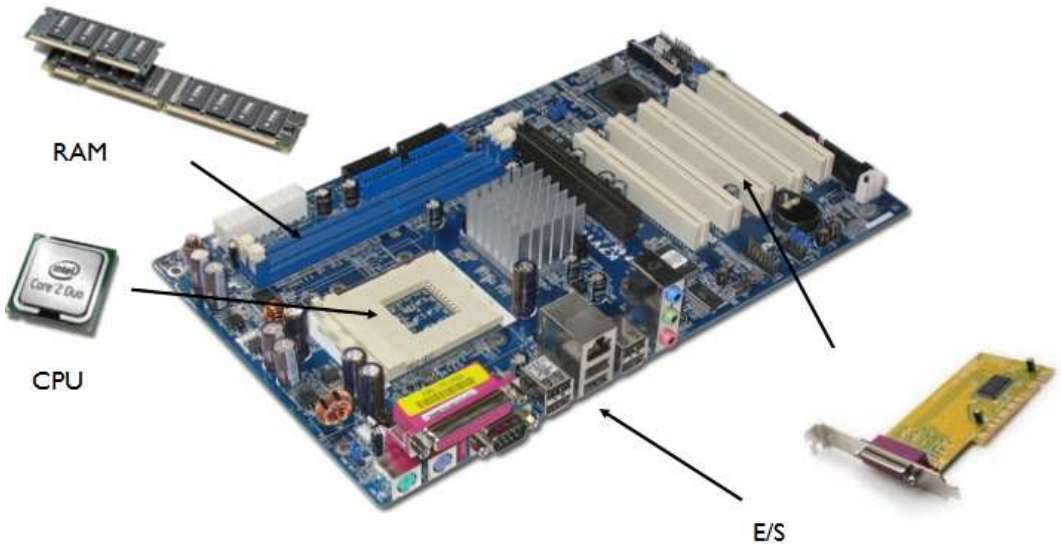
2. **Les mémoires de masse :** Pour sauvegarder les données, le système d'exploitation ou les programmes, l'ordinateur dispose d'une ou plusieurs *mémoire de masse*. Outre le fait d'être non volatiles, elles présentent également l'avantage d'être accessible en écriture. En revanche, ce type de mémoire est généralement lente en lecture (environ 1000 fois plus lente que la lecture de la RAM).

Mémoire flash

La principale mémoire de masse est le *disque dur* pour un ordinateur de bureau ou la *mémoire flash* pour un smartphone, mais parmi les mémoires de masse on trouve aussi : les clés USB, les cartes SD (dans les appareils photos numériques), les CD, les DVD...

1.5 La carte mère

La *carte mère* permet la coordination de l'ensemble des éléments intervenant dans le fonctionnement d'un ordinateur.



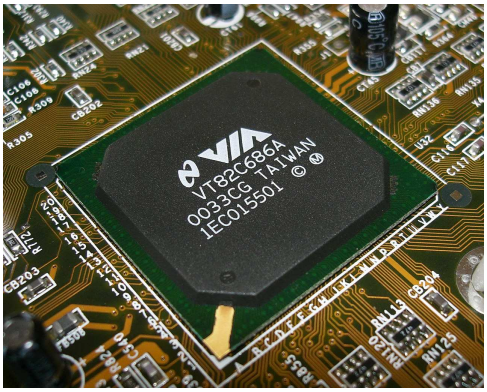
On y trouve en particulier :

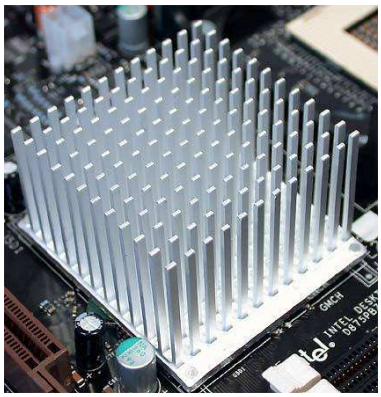
1. Des connecteurs :

   - essentiels : connecteur de la RAM, du micro-processeur, de l'alimentation électrique, du ventilateur
   - d'extension :
     — les ports PCI pour brancher des carte réseau, carte son, carte de connection (USB...), carte TV (antenne), carte DV (caméra), carte SCSI (scanner, graveur)...
     — un port AGP pour la carte graphique (plus performant qu'un port PCI)

2. Des composants :

   - le *chipset* (décomposé en 2 parties : le northbridge et le bridge d'entrée / sortie) qui regroupe les circuits électroniques permettant de gérer les interconnections





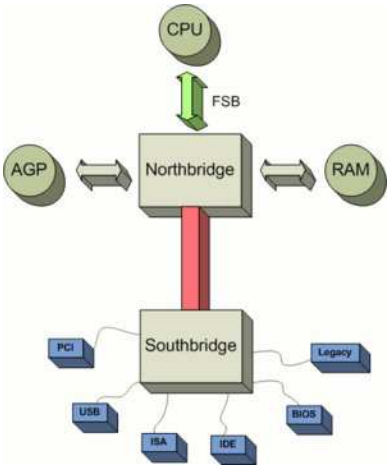
(Bridge d'entrée / sortie Relié aux périphériques) - Le chipset - (Northbridge relié au CPU et à la RAM)

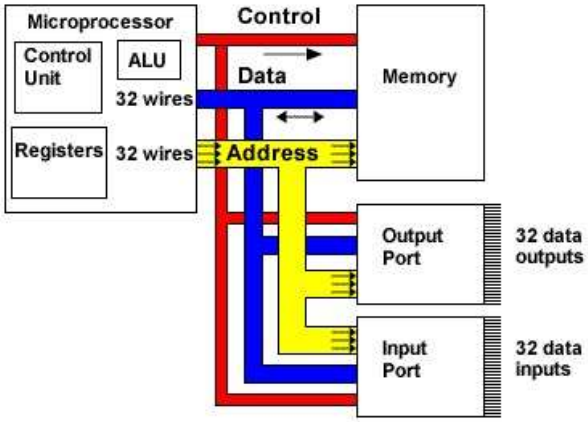
- une *horloge* (composée de deux cristaux) qui détermine la vitesse d'exécution des instructions du micro-processeur et des périphériques

- une mémoire morte (ROM) ou mémoire flash contenant le BIOS.

3. Des *bus* permettant le transfert d'informations :

   - le FSB (Front Side Bus) qui permet la communication entre le micro-processeur et le chipset.
   - le bus mémoire qui relie la RAM au chipset
   - le bus d'entrées/sorties qui relie le microprocesseur aux entrées/sorties et aux connecteurs d'extension



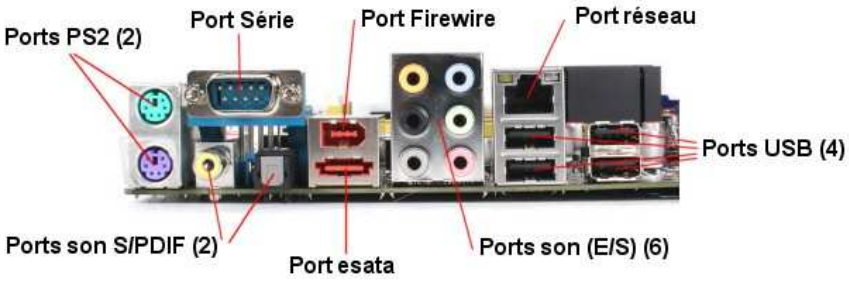


Chacun des bus précédents se décompose en *bus d'adresses*, en *bus de données* et en *bus de commandes* qui comportent 32 fils (pour un micro-processeur 32 bits).

4. Des *connexions* diverses, présentes ou non :

   - les ports USB (Universal Serial Bus) très polyvalents et assurant un transfère rapide des données
   - le port PS2 pour la souris et le clavier (remplacé aujourd'hui par un port USB)
   - le port parallèle pour les anciennes imprimantes (remplacé aujourd'hui par un port USB)
   - le connecteur RJ45 ou RS232 pour la connexion à des réseaux informatiques (remplace le *port série* : COM1 et COM2)

- les connecteurs VGA (analogique) et/ou DVI (numérique) pour la connexion d'un écran
- les connecteurs audio
- les connecteurs audio/vidéo HDMI ou Displayport
- les connecteurs ATA (ou anciennement IDE) pour les disques durs et e-SATA pour les périphériques de stockage externe



Remarque 6. Il y a autant de cartes mère que de types d'ordinateur.
Afin d'accroître les performances d'un ordinateur, certaines d'entre elles :

1. permettent la connection de plusieurs microprocesseurs : jusqu'à 8 !
2. disposent d'un branchement (socket) pour le micro-processeur permettant à l'utilisateur de changer celui-ci.

2 Le Système d'Exploitation

Le hard que nous venons de décrire, sert à exécuter des programmes, lire et stocker des données dans la mémoire de masse. En pratique, l'utilisateur d'un ordinateur utilise de nombreux programmes, parfois même de façon simultanée. Les instructions et les données relatives à ces programmes doivent être stockées de façon très structurée afin qu'il n'y ait pas de problème dans leur exécution.

Le rôle principal du *système d'exploitation* est de :

- repérer les endroits où sont stockés les programmes et les données
- de gérer le lancement des programmes
- de stocker de façon cohérentes les données générées.

Après le BIOS, le système d'exploitation est le deuxième programme lancé par l'ordinateur : il est stocké en mémoire vive à l'allumage de l'ordinateur et y reste jusqu'au moment où l'on éteint la machine.

On distingue deux catégories de systèmes d'exploitation :

1. Les systèmes issus d'Unix utilisés dans quasiment tous les domaines à l'exception des ordinateurs personnels. On les trouve donc dans les téléphones portables (Androïd), dans les ordinateurs personnels (Linux, Max OS X), dans les box internets ainsi que dans les serveurs web et les super-calculateurs.

2. Les systèmes de la famille Microsoft Windows qui ont su s'imposer dans la plupart des ordinateurs personnels (90%) en raison de leur convivialité mais surtout de l'agressivité de la politique commerciale de Microsoft.

En complément des 3 tâches principales précédentes, un système d'exploitation permet également en général de :

1. donner l'illusion qu'un ordinateur est multitâche en coordonnant l'exécution simumltanée des programmes
2. identifier les différents utilisateurs afin de contrôler leur accès aux différentes ressources de l'ordinateur
3. servir de garde-fou en cas de mauvaise utilisation des ressources de l'odinateur.

Nous allons décrire ces différentes fonctions dans les parties suivantes...

2.1 Le multitâche

Nous avons vu qu'un ordinateur n'est capable d'exécuter qu'une instruction à la fois. Pourtant, grâce au système d'exploitation, nous avons l'impression que celui-ci est capable d'exécuter plusieurs tâches en même temps.

Cette illusion provient de la conjonction de deux effets : d'une part, les instructions exécutées par l'ordinateur sont extrêmement rapides (de l'ordre de quelques dizaines de millisecondes) et d'autre part, le système d'exploitation utilise les temps d'attente dans l'exécution d'une tâche pour orienter les ressources de l'ordinateur (micro-processeur + RAM...) vers l'exécution d'une autre tâche.

En fait, de plus en plus d'ordinateurs sont dotés de plusieurs processeurs (souvent des *dualcore* ou *quadricore*) ce qui leur permet d'exécuter réellement plusieurs tâches simultanément. Malgré tout, le nombre d'instructions effectuées simultanément reste habituellement très inférieur au nombre d'applications en fonctionnement simultané.

2.2 Identification des utilisateurs

Les systèmes d'exploitation sont en général *multi-utilisateurs*. Lors de l'ouverture du système d'exploitation, les utilisateurs sont amenés à s'identifier à l'aide d'un *identifiant* et d'un *mot de passe*.

Lors de la création d'un *compte utilisateur*, l'administrateur du réseau est amené à associer l'utilisateur à un ou plusieurs groupes, lui donnant ainsi certains droits d'accès et de modification des ressources en place sur l'ordinateur (ou parle aussi de *serveur* lorsque le nombre d'utilisateurs est important).

Au moment de la connexion d'un utilisateur, celui-ci est reconnu par le système d'exploitation qui ouvre alors un environnement (*shell*) propre à cet utilisateur. Les environnements habituels de travail sont souvent très conviviaux et autorise l'utilisation de la souris ou d'un écran tactile.

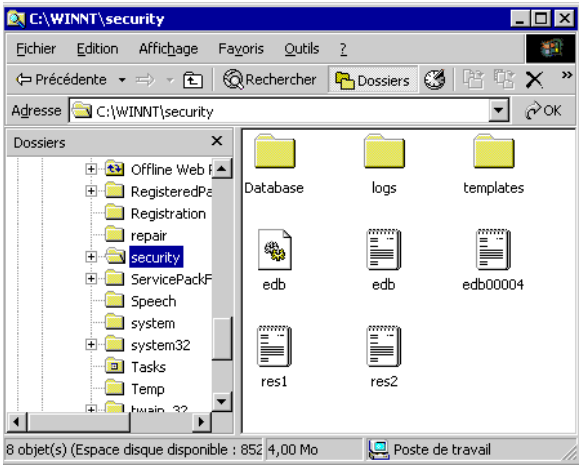
Remarque 7. Cependant, il est aussi possible d'utiliser un environnement bien plus sommaire appelé *interprète en mode texte* qui se présentent sous la forme d'un écran noir avec une ligne d'instruction en mode texte. Les premiers ordinateurs fonctionnaient sous ce type d'environnement (le *DOS*) mais ceux-ci n'ont pas disparus car ils permettent aux administrateurs de systèmes informatiques d'effectuer des instructions de façon beaucoup plus rapides.

Interprète en mode texte

2.3 Système de fichiers

La mémoire de masse est en générale organisée en un *système de fichiers* permettant aux utilisateurs d'enregistrer leurs données et leurs applications.

Le nombre de fichiers contenu dans le disque dur d'un ordinateur dépasse souvent plusieurs centaines de milliers. Ils sont classés en une structure arborescente de répertoires. Du point de vue de l'utilisateur, il s'agit de répertoires contenant des fichiers et des sous-répertoires.



Sous Windows, le disque dur est souvent partitionné en deux sections :

- C : destiné à stocker tous les éléments du système d'exploitation
- D : dans lequel on stocke les données (applications et fichiers) des différents utilisateurs.

L'instruction permettant d'accéder à un fichier "toto.txt" est alors la suivante "c :\ rep1 \ rep2... \ repn \ toto.txt" où rep1, ..., repn sont les différents répertoires emboités dans lesquels se trouvent le fichier "toto.txt".

Organisation des fichiers sur le disque dur :

Un fichier ou une application se présente sous la forme d'une succession de mots mémoire de 32 ou 64 bits. Rien a priori ne les distingue donc les uns des autres et c'est le logiciel de lecture qui permettra de leur donner un sens.

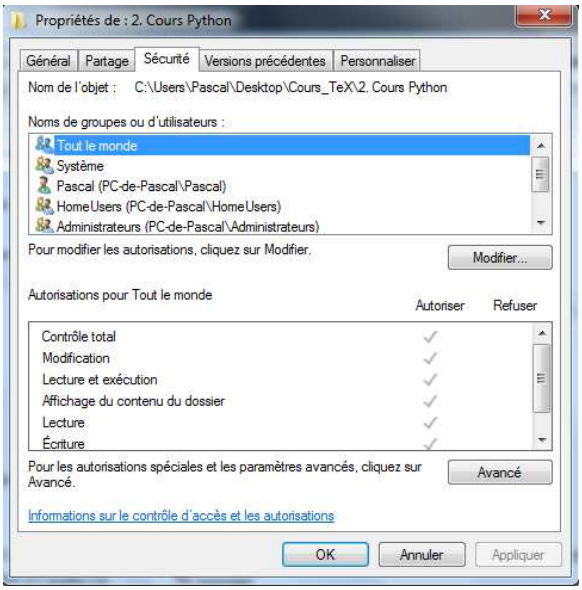
Ces fichiers sont répertoriés dans des *tables* (*table des inodes*) contenant des informations relatives à ces fichiers appelées *métadonnées*. Parmi elles, se trouvent :

1. les dates de créations, de dernière modification, de dernière lecture
2. la taille du fichier
3. l'emplacement des données du fichier sur le disque
4. un numéro identifiant le fichier (appelé *inode* sous Unix)

Ainsi, du point de vue du système d'exploitation, un fichier correspond à un couple $(n,\ i)$ où n correspond au nom du fichier et i à son inode. La table des inodes permet alors d'identifier la nature du fichier : fichier de données, application ou répertoire.

2.4 Contrôle d'accès

Parmi les informations contenues dans la table des inodes, on trouve les droits d'accès en lecture et écriture relatifs aux différents fichiers et répertoires qui s'y trouvent. Sous Microsoft Windows, on a accès à ces informations en choisissant les options "propriétés" puis "sécurité" avec avoir effectué un clic droit avec la souris.



Chaque fichier et répertoire attribue à chacun des groupes d'utilisateurs définis par le système, des droits définis parmi la liste suivante :

- Contrôle total
- Modification
- Lecture et exécution
- Affichage du contenu du dossier
- Lecture
- Ecriture
- Autorisations spéciales

En général, le seul statut permettant d'effectuer ce que l'on veut dans un système de fichiers est le statut *administrateur*. Si vous ne disposez pas des autorisations suffisantes, il ne faut donc pas vous étonner si apparaît un message d'erreur lorsque :

- Vous tentez d'ouvrir un répertoire
- Vous tentez d'ouvrir un fichier
- Vous tentez d'enregistrer un fichier dans un dossier
- Vous souhaitez modifier le contenu d'un fichier
- Vous voulez lancer une application

Remarque 8. De même, des autorisations sont attribuées par le système d'exploitation pour l'utilisation des différents périphériques tels que : l'écran, la carte son, le clavier, la souris, les imprimantes et le réseau.

2.5 Lancement des applications

• **Pour lancer une application** (un programme), l'utilisateur dispose des possibilités suivantes :

1. Dans le shell graphique (Windows) :
   (a) soit cliquer directement sur l'îcone représentant l'application
   (b) soit cliquer sur un raccourci renvoyant sur l'application
   (c) soit cliquer sur un fichier dont l'extension renvoie sur l'application

2. Taper une commande dans l'interpréteur en mode texte (DOS) :
   Cette méthode devient nécessaire dès que l'on souhaite passer des arguments à un programme ou manipuler des fichiers en masse. Elle demande cependant des connaissances informatiques avancées.

• **Que se passe-t-il lors du lancement d'une application ?**

1. Le système d'exploitation commence par vérifier que l'utilisateur dispose bien des droits l'autorisant à lancer l'application.
2. Si c'est le cas, il réserve un espace mémoire dans la mémoire vive de l'ordinateur pour stocker les instructions du programme et les données utilisées et/ou produites.
3. Il copie alors le programme (fichier exécutable) dans la mémoire vive. Il s'agit d'une série d'instructions codées sous forme d'une suite de bits que l'on appelle le *langage machine* et qui sont directement compréhensibles par le micro-processeur. Pour exécuter ces instructions, il suffit de "brancher" le micro-processeur sur la première instruction du programme.

• **Comment écrire et lancer son propre programme ?**

Un utilisateur qui souhaite écrire son propre programme a la choix entre 3 types de langage :

1. **Ecrire son programme en langage machine :**
   C'est une tâche particulièrement difficile car le langage machine n'est pas adapté à la transcription d'instruction un minimum complexe. On dit que c'est un langage de *bas niveau*. De plus, les micro-processeurs étant différents d'un ordinateur à l'autre, un programme en langage machine ne peut être exécuté que par la machine sur laquelle il a été conçu.

2. **Utiliser un langage compilé :**
   L'utilisateur commence par rédiger son programme dans un langage évolué (par opposition au "langage machine") qui est alors traduit en langage machine par un programme appelé *un compilateur*.

3. **Utiliser un langage de script :**
   L'utilisateur commence par rédiger son programme dans un langage évolué puis utilise un application permettant de lire et d'exécuter le programme instruction par instruction (appelé *un interpréteur*).

Remarque 9.

1. Par rapport à l'interpréteur, le compilateur présente l'inconvénient de retarder l'exécution d'un programme mais une fois compilé, le fichier exécutable obtenu est particulièrement rapide.

2. Le choix entre "compilateur" et "interpréteur" dépend beaucoup du langage dans lequel on programme : certains langages ne proposent qu'une seule de ces deux possibilités tandis que d'autres comme Python laissent le choix.

3. Il existe des langages intermédiaires où un compilateur commence par traduire les instructions dans un langage proche du langage machine, puis donne la main à un interpréteur qui se charge alors de l'exécution du programme (on parle de *bytecode*).

3 Les environnements de développement

Un ordinateur est une machine universelle et à ce titre, tout utilisateur peut écrire et exécuter des programmes. Nous avons vu qu'un programme se présentait sous la forme d'une série d'instructions codées en binaire (le langage machine). Heureusement pour l'utilisateur, il existe des applications lui permettant de rédiger des programmes dans un langage proche du langage courant.

La première chose à choisir est donc le langage de programmation. Il en existe de très nombreux, certains obsolètes (le basic), certains polyvalents (Python, C++, Java, Caml...) et d'autres plus spécifiques à une utilisation donnée (HTML,

SAS, TeX...). Le langage choisi pour le cours d'*Informatique pour Tous* de CPGE est le langage PYTHON tandis que le langage utilisé dans l'*Option Informatique* est le langage CAML.

Une fois le langage choisi, le programmeur doit choisir un *environnement de développement intégré* (IDE).
Il s'agit d'un logiciel permettant au minimum d'écrire un programme dans un éditeur adapté au langage et souvent également :

1. d'exécuter les programmes que l'on a écrits ;
2. de corriger les erreurs dans le programme ;
3. et éventuellement de consulter de la documentation.

Remarque 10.

1. Un IDE peut être spécifique à un langage de programmation donné ou commun à plusieurs langages de programmation.
2. Un IDE peut être basique en n'incluant que des fonctions d'éditeur de texte ou bien très complet avec en plus de l'éditeur, une aide à la rédaction de programmes, une fenêtre d'exécution, une console de travail...

• Quelques exemples d'IDE adaptés à Python

1. **IDLE :** spécifique à Python, fourni avec la distribution standard de Python et adapté à une programmation basique.
2. **Eclipse :** adapté à la programmation dans plusieurs langages de programmation (y compris Python en installant le plugin *PyvDev*)
3. **Emacs ou Vim :** ce sont de simples éditeurs de texte qui peuvent également servir à la rédaction de programmes dans différents langages.
4. **Spyder :** spécifique à Python et inclus dans de nombreuses distributions telles que Anaconda, WinPython ou Python(x,y) sous Windows ou MacPorts sous Mac OS. Il présente l'avantage d'inclure les bibliothèques utilisées en CPGE et de contenir un débogueur.