Architecture matérielle et logicielle d'un ordinateur.

MP2I - Informatique

Anthony Lick
Lycée Janson de Sailly

Un peu d'histoire

Les premiers “ordinateurs”

Qu’est-ce qu’un ordinateur ?

En anglais : “computer”, qui pourrait se traduire en français par “calculateur”.

Les premiers “ordinateurs”

Qu’est-ce qu’un ordinateur ?

En anglais : “computer”, qui pourrait se traduire en français par “calculateur”.

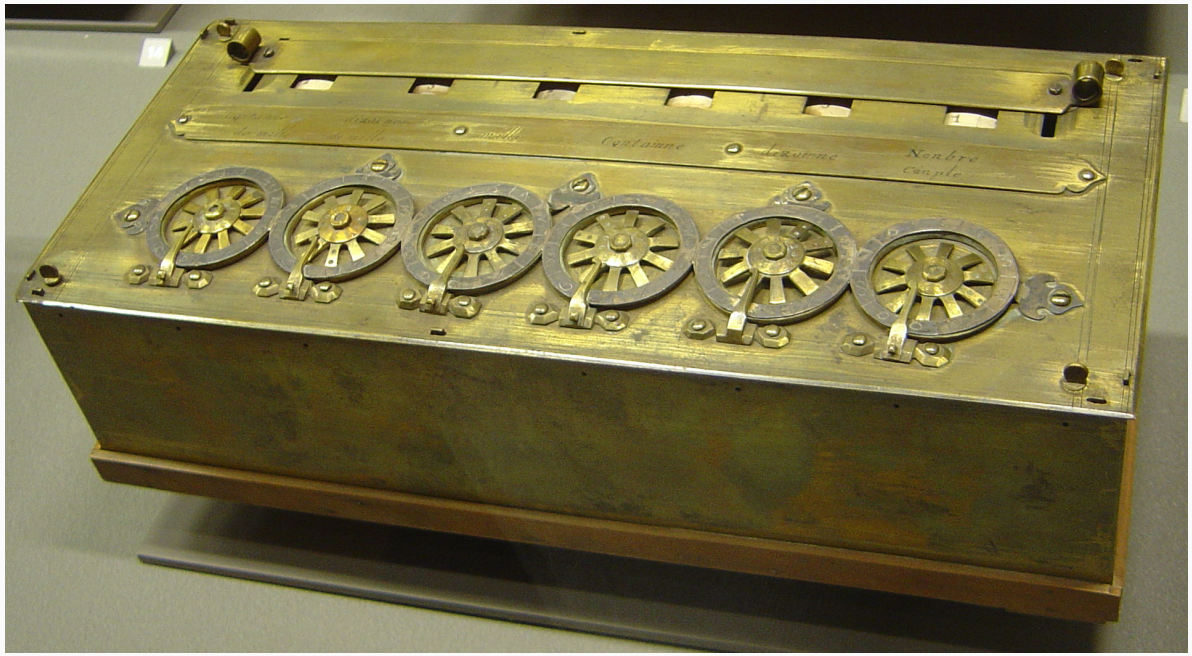


Figure 1 – Une Pascaline (inventée par Blaise Pascal en 1642), exposée au au musée des arts et métiers de Paris

Les premiers “ordinateurs”

Qu’est-ce qu’un ordinateur ?

En français, le mot “ordinateur” insiste sur le fait qu’on veut **ordonner** des données afin de les traiter automatiquement.

Les premiers “ordinateurs”

Qu’est-ce qu’un ordinateur ?

En français, le mot “ordinateur” insiste sur le fait qu’on veut **ordonner** des données afin de les traiter automatiquement.

Organisation des données

Il y a environ 5000 ans, les premières villes sont apparues en Mésopotamie. Il a fallu les organiser, et pour cela gérer diverses informations :

- état des stocks
- codification des règles régissant ces stocks

Ainsi, les premiers systèmes de numération et d’écriture se sont développés.

Les premiers “ordinateurs”

Automatisation

Au début du XIXème siècle, on réalise que si des informations sont transmises sous un format prédéfini, elles peuvent être traitées automatiquement par une machine.

Les premiers “ordinateurs”

Automatisation

Au début du XIXème siècle, on réalise que si des informations sont transmises sous un format prédéfini, elles peuvent être traitées automatiquement par une machine.

Exemple

En 1801, Jacquard met au point une machine à tisser dont les motifs peuvent être **programmés** à l'aide de cartes perforées.

Figure 2 – Mécanisme Jacquard, exposé au musée des arts et métiers de Paris

L'ordinateur, une machine universelle

Qu'est-ce qu'un ordinateur ?

Un ordinateur est une machine qui :

- prend des données en entrée ;
- est capable de traiter automatiquement ces données ;
- puis renvoie un résultat.

L'ordinateur, une machine universelle

Qu'est-ce qu'un ordinateur ?

Un ordinateur est une machine qui :

- prend des données en entrée ;
- est capable de traiter automatiquement ces données ;
- puis renvoie un résultat.

Est-ce suffisant ?

Selon cette définition, les appareils suivants pourraient être considérés comme des ordinateurs :

- Les smartphones et les tablettes ;

L'ordinateur, une machine universelle

Qu'est-ce qu'un ordinateur ?

Un ordinateur est une machine qui :

- prend des données en entrée ;
- est capable de traiter automatiquement ces données ;
- puis renvoie un résultat.

Est-ce suffisant ?

Selon cette définition, les appareils suivants pourraient être considérés comme des ordinateurs :

- Les smartphones et les tablettes ;
- Un thermostat d'ambiance contrôlant une chaudière.

L'ordinateur, une machine universelle

Comment améliorer cette définition ?

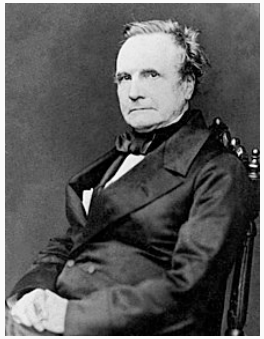
- Un thermostat d'ambiance semble trop **spécialisé** pour être considéré comme un ordinateur.
- Tous comme les ordinateurs, les smartphones et tablettes semblent être des **machines universelles**.

La machine analytique de Babbage

Concept de machine universlle

En 1834, **Charles Babbage** imagine sa **machine analytique**, qui est le premier prototype d'une machine universelle, que l'on pourrait programmer à notre guise.

Malheureusement, limité par la technologie de son temps, il ne parviendra jamais à la construire.



(a) Charles Babbage

(b) Une partie de la machine analytique

Le premier programme informatique

Le premier algorithme

En 1843, **Ada Lovelace** écrit le premier programme informatique.

Il était destiné à fonctionner sur la **machine analytique** de **Babbage**, pour calculer des **nombres de bernouilli**.

Cet algorithme possède une **boucle while**, une révolution pour l'époque.

L'informatique théorique

Les début de l'informatique théorique

Au cours du XXème siècle, l'informatique théorique commence à se développer, et l'on s'intéresse aux questions suivantes :

- Quelle est une bonne notion d'algorithme ?
- Quels sont les problèmes qu'on peut résoudre à l'aide d'un algorithme ? (on parle de problème **décidable**)
- Existe-t-il des problèmes **indécidables**, i.e. tels qu'aucun algorithme ne peut les résoudre ?

L'informatique théorique

Problème de décision

En 1928, **David Hilbert** et **Wilhelm Ackermann** demandent s'il existe un **algorithme** capable de décider si une proposition mathématique est vraie ou fausse.

(a) David Hilbert

(b) Wilhelm Ackermann

Les algorithmes

Définition (algorithme)

Un algorithme est une suite finie non ambigüe d'opérations ou d'instructions permettant de résoudre un problème.

Donald Knuth

L'informaticien **Donald Knuth** liste **cinq** propriétés comme prérequis d'un algorithme.

Les algorithmes

Prérequis d'un algorithme (Donald Knuth)

- **finitude** : un algorithme doit toujours se terminer après un nombre fini détapes.
- **définition précise** : chaque étape d'un algorithme doit être définie précisément et sans ambiguïté.
- **entrées** : des quantités (prises dans un ensemble spécifié) qui lui sont données avant qu'un algorithme ne commence.
- **sorties** : des quantités ayant une relation spécifiée avec les entrées.
- **rendement** : toutes les opérations que l'algorithme doit accomplir doivent être suffisamment basiques pour pouvoir être en principe réalisées dans une durée finie par un humain utilisant un papier et un crayon.

Les machines de Turing

Turing

En 1936, **Alan Turing** introduit un modèle théorique pour représenter des machines effectuant des calculs : les **machines de Turing**.

Les machines de Turing

Turing

En 1936, **Alan Turing** introduit un modèle théorique pour représenter des machines effectuant des calculs : les **machines de Turing**.

Lien avec les algorithmes

À chaque machine de Turing correspond un algorithme, et à chaque algorithme correspond une machine de Turing.

Exemple

Il existe une machine de Turing qui calcule les additions.

Il existe une machine de Turing qui cherche un mot dans un texte.

Les machines de Turing

Problème de décision

En 1936, **Alan Turing** et **Alonzo Church** résolvent indépendamment le problème posé par **Hilbert** et **Ackermann** : un tel algorithme n'existe pas.

(a) Alan Turing

(b) Alonzo Church

La machine de Turing universelle

Machine de Turing universelle

Alan Turing démontre également qu'il existe une **machine universelle**, capable de simuler toutes les autres. C'est-à-dire, une machine qui :

- prend en entrée la description d'une autre machine, ainsi que les données d'entrée de la machine à simuler ;
- simule la machine de Turing spécifiée en entrée sur ces données d'entrée.

La machine de Turing universelle

Machine de Turing universelle

Alan Turing démontre également qu'il existe une **machine universelle**, capable de simuler toutes les autres. C'est-à-dire, une machine qui :

- prend en entrée la description d'une autre machine, ainsi que les données d'entrée de la machine à simuler ;
- simule la machine de Turing spécifiée en entrée sur ces données d'entrée.

Lien avec les ordinateurs

Un ordinateur est l'implémentation concrète d'une machine de Turing universelle.

Architecture d'un ordinateur

Les premiers ordinateurs

Z3 (allemand, 1941)

Première réalisation concrète d'une machine de Turing (électromécanique).

Figure 6 – Réplique d'un Z3

ENIAC (américain, 1943-1946)

Première réalisation entièrement électronique.

La machine est utilisée pour calculer des tables indiquant les paramètres de tir d'une batterie d'artillerie en fonction de la distance à la cible, du vent, etc.

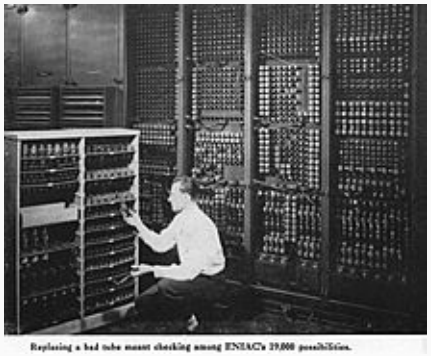


Figure 7 – Photo de l'ENIAC

Architecture de von Neumann

Architecture de von Neumann

John von Neumann participe à l'élaboration de l'ENIAC à partir de 1944, ainsi qu'à la conception de l'EDVAC (un autre ordinateur électronique).

En 1945, il publie un rapport présentant l'architecture de ces machines.

Architecture de von Neumann

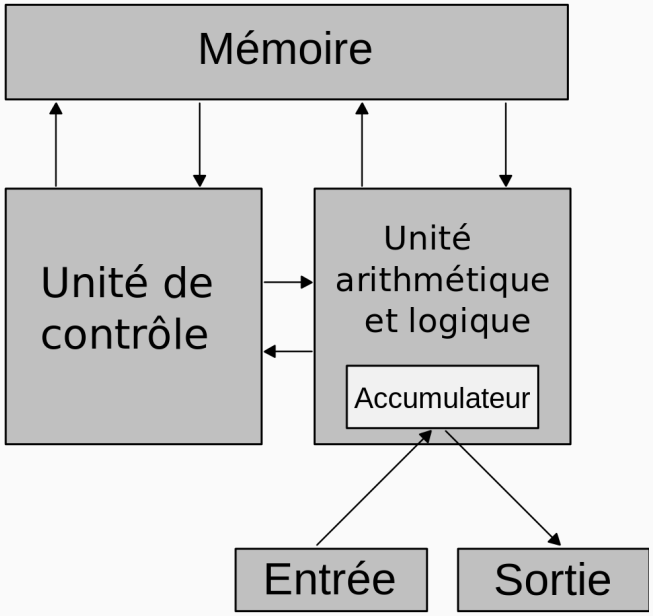


Figure 8 – Architecture de von Neumann

Architecture de von Neumann

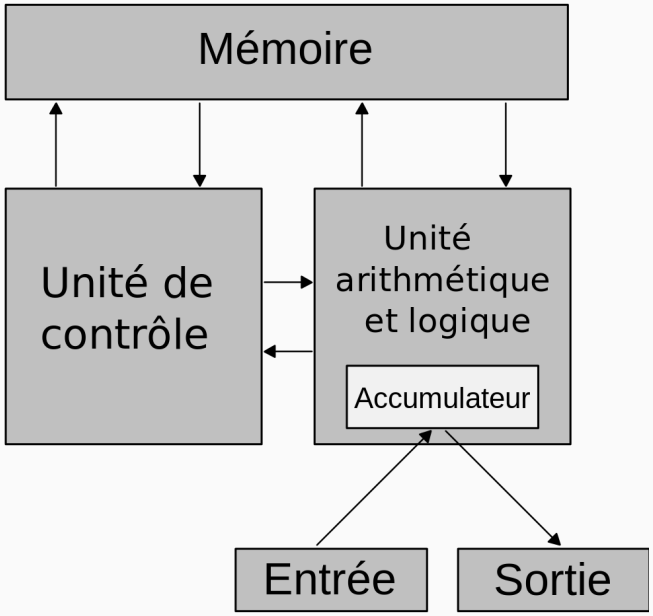


Figure 8 – Architecture de von Neumann

Unité Arithmétique et Logique

Capable de calculer des additions, soustractions, multiplications, ainsi que des disjonctions, conjonctions, négations.

Fait parti du processeur dans les ordinateurs actuels.

Architecture de von Neumann

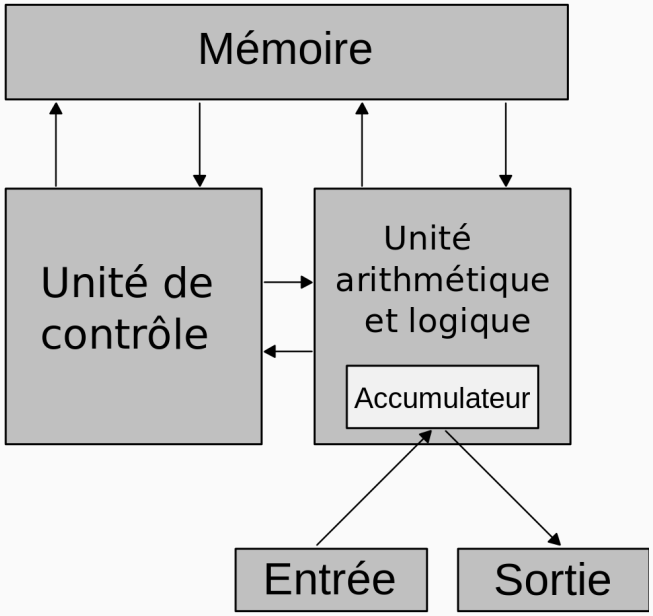


Figure 8 – Architecture de von Neumann

Unité de contrôle

Décide quel est la prochaine action à effectuer.

Fait parti du processeur dans les ordinateurs actuels.

Architecture de von Neumann

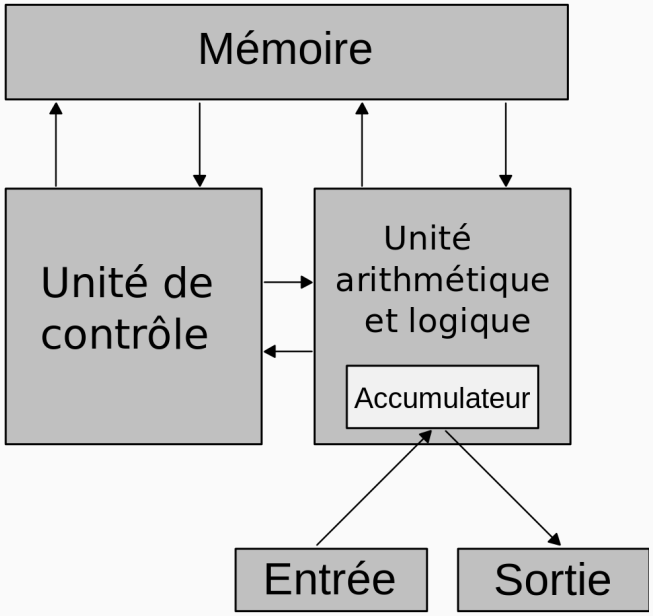


Figure 8 – Architecture de von Neumann

Mémoire

Contient une suite de 0 et de 1 représentant les données et les programmes. Plusieurs types de mémoire sont utilisés dans les ordinateurs actuels : la mémoire vive, mémoire de masse, etc.

Architecture de von Neumann

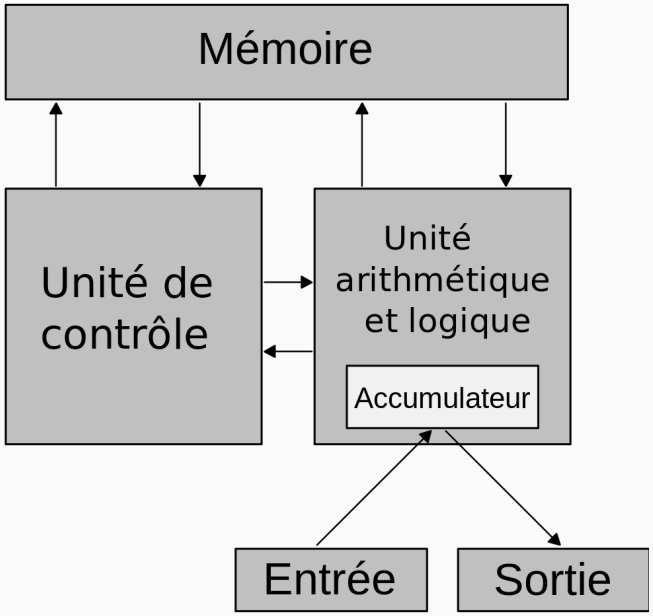


Figure 8 – Architecture de von Neumann

Périphériques d'entrée/sortie

Moyens d'interaction entre l'utilisateur et la machine.

(clavier, souris, écran, réseau, etc)

Organisation de la mémoire : la RAM

Mémoire vive (RAM)

La mémoire vive est une suite de chiffres binaires (**bits**), organisée en pratique en **octets** (paquets de 8 bits) ou en **mots mémoire** (64 bits sur les machines récentes).

Organisation de la mémoire : la RAM

Caractéristiques

- **Pas de sens a priori** : un mot mémoire est une suite de 0 et de 1 qui peut aussi bien représenter un entier, du texte, une instruction d'un programme, la couleur d'un pixel d'une image, ...

  C'est à l'utilisateur ou au reste de l'ordinateur d'interpréter un mot mémoire de la bonne façon !
- **Inertie** : La mémoire vive est inerte, c'est à dire qu'elle n'effectue aucun calcul.
- **Accès direct** : tout mot mémoire possède une adresse (un nombre entier). On peut lire ou écrire directement le contenu d'un mot mémoire d'adresse donnée.

Organisation de la mémoire

RAM

- Avantage : lire/écrire dans la RAM est très **rapide**.
- Inconvénient : la RAM est une mémoire **volatile**, son contenu est perdu si l'on éteint la machine.

Organisation de la mémoire

RAM

- Avantage : lire/écrire dans la RAM est très **rapide**.
- Inconvénient : la RAM est une mémoire **volatile**, son contenu est perdu si l'on éteint la machine.

Mémoire morte

Le programme permettant de démarrer l'ordinateur est en général stocké dans une mémoire **morte** (Read-Only Memory, ROM). Lire dans cette mémoire est également rapide, mais on ne peut pas y écrire.

Organisation de la mémoire

RAM

- Avantage : lire/écrire dans la RAM est très **rapide**.
- Inconvénient : la RAM est une mémoire **volatile**, son contenu est perdu si l'on éteint la machine.

Mémoire morte

Le programme permettant de démarrer l'ordinateur est en général stocké dans une mémoire **morte** (Read-Only Memory, ROM). Lire dans cette mémoire est également rapide, mais on ne peut pas y écrire.

Mémoire de masse

Le reste des informations est stockée dans la mémoire de masse (disque dur, SSD). L'accès à cette mémoire est plus lente, mais elle a l'avantage de ne pas être volatile (et moins cher).

Organisation de la mémoire

Utilisation de la RAM

- Lors de l'exécution d'un programme, les valeurs intermédiaires utiles au programme sont sotckées dans la RAM.
- Au lancement d'un programme installé sur la mémoire de masse, toutes les informations utiles sont chargées dans la RAM.
- Si la RAM est remplie, on utilise à la place la mémoire de masse (on parle de **swapping**).
  Cela impacte les performances du programme !

Exécution d'un programme

Registres

Les processeurs actuels possèdent des registres (une dizaine ou une centaine), qui peuvent stockés des mots mémoire.

Le processeur peut par exemple demander à son unité arithmétique et logique d'additionner le contenu de deux de ses registres, et de stocker le résultat dans un troisième registre.

Lors de l'exécution d'un programme stocké dans la mémoire, deux de ces registres jouent un rôle particulier :

Exécution d'un programme

Registres

Les processeurs actuels possèdent des registres (une dizaine ou une centaine), qui peuvent stockés des mots mémoire.

Le processeur peut par exemple demander à son unité arithmétique et logique d'additionner le contenu de deux de ses registres, et de stocker le résultat dans un troisième registre.

Lors de l'exécution d'un programme stocké dans la mémoire, deux de ces registres jouent un rôle particulier :

- le registre PC (program counter), qui se souvient de l'adresse du mot mémoire contenant la prochaine instruction du programme ;

Exécution d'un programme

Registres

Les processeurs actuels possèdent des registres (une dizaine ou une centaine), qui peuvent stockés des mots mémoire.

Le processeur peut par exemple demander à son unité arithmétique et logique d'additionner le contenu de deux de ses registres, et de stocker le résultat dans un troisième registre.

Lors de l'exécution d'un programme stocké dans la mémoire, deux de ces registres jouent un rôle particulier :

- le registre PC (program counter), qui se souvient de l'adresse du mot mémoire contenant la prochaine instruction du programme ;
- le registre IR (instruction register), qui sert à stocker cette instruction.

Exécution d'un programme

Comportement de l'unité de contrôle

Si lors de l'exécution d'un programme, le registre PC contient l'adresse n, l'unité de contrôle va :

- **lire l'instruction** : aller lire le mot stocké à l'adresse mémoire contenu dans PC, et stocker ce mot dans IR ;
- **incrémenter PC** : remplacer la valeur de PC (n) par $n+1$;
- **décoder l'instruction** contenue dans IR ;
- **exécuter l'instruction**.

Exécution d'un programme

Comportement de l'unité de contrôle

Si lors de l'exécution d'un programme, le registre PC contient l'adresse n, l'unité de contrôle va :

- **lire l'instruction** : aller lire le mot stocké à l'adresse mémoire contenu dans PC, et stocker ce mot dans IR ;
- **incrémenter PC** : remplacer la valeur de PC (n) par $n+1$;
- **décoder l'instruction** contenue dans IR ;
- **exécuter l'instruction**.

Instructions

- Opération arithmétique ou logique.
- Accès à la mémoire.
- Instruction de branchement : modifier la valeur de PC.

Système d'exploitation

Les systèmes d'exploitation

Unix

La première grande famille de systèmes d'exploitation sont issus d'Unix (Mac OS X, iOS, GNU/Linux, Android, FreeBSD, NetBSD).

Dans tous les domaines sauf celui des ordinateurs personnels, ces systèmes sont majoritaires. (téléphonie, routeurs, serveurs web, supercalculateurs).

Windows

La deuxième grande famille de systèmes d'exploitation sont basés sur Microsoft Windows. Grâce aux pratiques commerciales très agressives de Microsoft, Windows est en situation de quasi-monopole (90%) sur les ordinateurs personnels depuis plus de 20 ans.

Les fonctions d'un système d'exploitation

À quoi sert un système d'exploitation ?

Le système d'exploitation a les responsabilités suivantes :

- donner l'illusion que l'ordinateur est multitâche ;
- identifier les utilisateurs ;
- contrôler l'accès aux données du disque dur et de manière générale aux ressources de l'ordinateur ;
- servir de garde-fou en cas de tentative de mauvaise utilisation des ressources de l'ordinateur ;
- gérer le lancement des différentes applications utilisées ;
- gérer l'organisation du disque dur et de ses fichiers.

Le multitâche

Noyau

Le noyau du système d'exploitation choisit quelle application est exécutée à tout instant.

Multitâche

Lorsqu'on exécute plusieurs programmes en même temps, le noyau exécute plusieurs instructions du premier programme, puis le met en pause et passe à l'exécution d'un morceau du programme suivant, et ainsi de suite.

Chaque exécution d'un morceau de programme dure environ une dizaine de millisecondes.

Exemple : saisir du texte en écoutant de la musique

Exemple : saisir du texte en écoutant de la musique

- Le noyau donne d'abord la main à votre lecteur de musique, qui calcule les prochaines notes et les envoie à vos enceintes ;

Exemple : saisir du texte en écoutant de la musique

- Le noyau donne d'abord la main à votre lecteur de musique, qui calcule les prochaines notes et les envoie à vos enceintes ;
- Le noyau reprend la main, et votre clavier l'informe que vous avez appuyé sur une touche ;

Exemple : saisir du texte en écoutant de la musique

- Le noyau donne d'abord la main à votre lecteur de musique, qui calcule les prochaines notes et les envoie à vos enceintes ;
- Le noyau reprend la main, et votre clavier l'informe que vous avez appuyé sur une touche ;
- Le noyau sait que votre logiciel de traitement de texte est en attente d'une telle action, il lui envoie donc l'information et lui donne la main ;

Exemple : saisir du texte en écoutant de la musique

- Le noyau donne d'abord la main à votre lecteur de musique, qui calcule les prochaines notes et les envoie à vos enceintes ;
- Le noyau reprend la main, et votre clavier l'informe que vous avez appuyé sur une touche ;
- Le noyau sait que votre logiciel de traitement de texte est en attente d'une telle action, il lui envoie donc l'information et lui donne la main ;
- Le logiciel de traitement de texte affiche la lettre en question à l'écran ;

Exemple : saisir du texte en écoutant de la musique

- Le noyau donne d'abord la main à votre lecteur de musique, qui calcule les prochaines notes et les envoie à vos enceintes ;
- Le noyau reprend la main, et votre clavier l'informe que vous avez appuyé sur une touche ;
- Le noyau sait que votre logiciel de traitement de texte est en attente d'une telle action, il lui envoie donc l'information et lui donne la main ;
- Le logiciel de traitement de texte affiche la lettre en question à l'écran ;
- Pendant ce temps, votre périphérique audio informe le noyau qu'il a fini de lire les notes qui lui ont été envoyées ;

Exemple : saisir du texte en écoutant de la musique

- Le noyau donne d'abord la main à votre lecteur de musique, qui calcule les prochaines notes et les envoie à vos enceintes ;
- Le noyau reprend la main, et votre clavier l'informe que vous avez appuyé sur une touche ;
- Le noyau sait que votre logiciel de traitement de texte est en attente d'une telle action, il lui envoie donc l'information et lui donne la main ;
- Le logiciel de traitement de texte affiche la lettre en question à l'écran ;
- Pendant ce temps, votre périphérique audio informe le noyau qu'il a fini de lire les notes qui lui ont été envoyées ;
- Le noyau redonne la main à votre lecteur de musique ;

Exemple : saisir du texte en écoutant de la musique

- Le noyau donne d'abord la main à votre lecteur de musique, qui calcule les prochaines notes et les envoie à vos enceintes ;
- Le noyau reprend la main, et votre clavier l'informe que vous avez appuyé sur une touche ;
- Le noyau sait que votre logiciel de traitement de texte est en attente d'une telle action, il lui envoie donc l'information et lui donne la main ;
- Le logiciel de traitement de texte affiche la lettre en question à l'écran ;
- Pendant ce temps, votre périphérique audio informe le noyau qu'il a fini de lire les notes qui lui ont été envoyées ;
- Le noyau redonne la main à votre lecteur de musique ;
- ...

Identification des utilisateurs

Utilisateurs

Les systèmes d'exploitation actuels sont multi-utilisateurs : chaque utilisateur dispose d'un identifiant auprès du système (et en général d'un mot de passe).

Droits d'accès

Tous les utilisateurs n'ont pas accès à l'intégralité des fichiers (voire même des applications).
Le système d'exploitation gère les droits d'accès des différents utilisateurs.

Exemple : droits d'accès sur le réseau du lycée

Exemple : droits d'accès sur le réseau du lycée

- Chaque utilisateur possède un dossier personnel, auquel il est le seul à avoir accès en lecture et en écriture ;

Exemple : droits d'accès sur le réseau du lycée

- Chaque utilisateur possède un dossier personnel, auquel il est le seul à avoir accès en lecture et en écriture ;
- Les utilisateurs peuvent appartenir à un ou plusieurs groupes, ce qui leur fournit d'autres droits d'accès. Par exemple :

Exemple : droits d'accès sur le réseau du lycée

- Chaque utilisateur possède un dossier personnel, auquel il est le seul à avoir accès en lecture et en écriture ;
- Les utilisateurs peuvent appartenir à un ou plusieurs groupes, ce qui leur fournit d'autres droits d'accès. Par exemple :
  - Le groupe `eleve` a accès à un dossier commun en lecture et en écriture ;

Exemple : droits d'accès sur le réseau du lycée

- Chaque utilisateur possède un dossier personnel, auquel il est le seul à avoir accès en lecture et en écriture ;
- Les utilisateurs peuvent appartenir à un ou plusieurs groupes, ce qui leur fournit d'autres droits d'accès. Par exemple :
  - Le groupe `eleve` a accès à un dossier commun en lecture et en écriture ;
  - Un dossier `enseignement` est accessible en lecture et écriture aux membres du groupe `professeur`, et seulement en lecture au groupe `eleve` ;

Exemple : droits d'accès sur le réseau du lycée

- Chaque utilisateur possède un dossier personnel, auquel il est le seul à avoir accès en lecture et en écriture ;
- Les utilisateurs peuvent appartenir à un ou plusieurs groupes, ce qui leur fournit d'autres droits d'accès. Par exemple :
  - Le groupe `eleve` a accès à un dossier commun en lecture et en écriture ;
  - Un dossier `enseignement` est accessible en lecture et écriture aux membres du groupe `professeur`, et seulement en lecture au groupe `eleve` ;
  - Le groupe `professeur` a accès à un dossier spécial, auquel le groupe `eleve` n'a aucun accès ;

Exemple : droits d'accès sur le réseau du lycée

- Chaque utilisateur possède un dossier personnel, auquel il est le seul à avoir accès en lecture et en écriture ;
- Les utilisateurs peuvent appartenir à un ou plusieurs groupes, ce qui leur fournit d'autres droits d'accès. Par exemple :
  - Le groupe `eleve` a accès à un dossier commun en lecture et en écriture ;
  - Un dossier `enseignement` est accessible en lecture et écriture aux membres du groupe `professeur`, et seulement en lecture au groupe `eleve` ;
  - Le groupe `professeur` a accès à un dossier spécial, auquel le groupe `eleve` n'a aucun accès ;
  - Chaque classe peut donner lieu à un groupe d'utilisateurs ;

Exemple : droits d'accès sur le réseau du lycée

- Chaque utilisateur possède un dossier personnel, auquel il est le seul à avoir accès en lecture et en écriture ;
- Les utilisateurs peuvent appartenir à un ou plusieurs groupes, ce qui leur fournit d'autres droits d'accès. Par exemple :
  - Le groupe `eleve` a accès à un dossier commun en lecture et en écriture ;
  - Un dossier `enseignement` est accessible en lecture et écriture aux membres du groupe `professeur`, et seulement en lecture au groupe `eleve` ;
  - Le groupe `professeur` a accès à un dossier spécial, auquel le groupe `eleve` n'a aucun accès ;
  - Chaque classe peut donner lieu à un groupe d'utilisateurs ;
  - Le groupe `administrateur` est le seul à avoir accès à certains fichiers.

Lancement d'applications

Connexion utilisateur

Une fois connecté, un utilisateur a accès aux fichiers qui lui sont autorisés, ainsi qu'à certaines applications.

Lanceur d'applications

L'utilisateur a accès à un lanceur d'application (interface graphique ou ligne de commandes) permettant d'ouvrir des programmes.

Système de fichiers

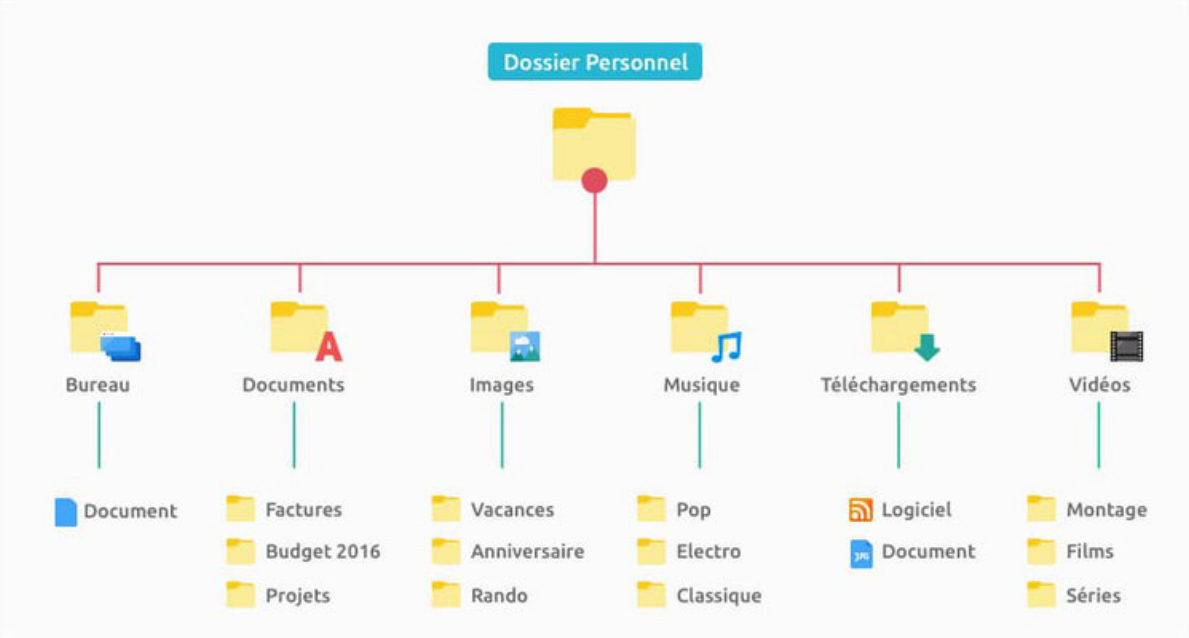
Arborescence

La mémoire de masse est généralement organisée en un **système de fichiers** qui permet aux utilisateurs d'enregistrer leurs données et leurs programmes.

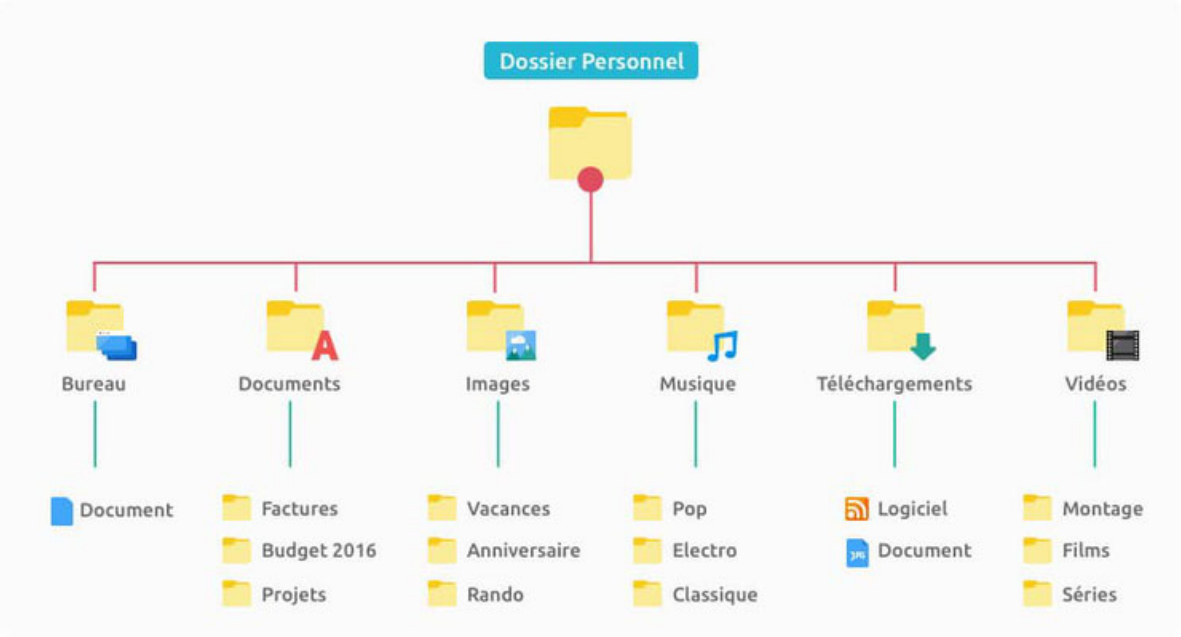
Ce système de fichiers à une structure arborescente :

- La **racine** du système de fichiers est un dossier contenant tous les autres ;
- chaque dossier peut contenir des fichiers, ou d'autres dossiers, contenant eux-mêmes des fichiers ou des dossiers ;
- L'**adresse** d'un fichier ou d'un dossier est la liste des dossiers à ouvrir dans l'ordre depuis la racine pour accéder à celui-ci.

Exemple de système de fichiers



Exemple de système de fichiers



Linux

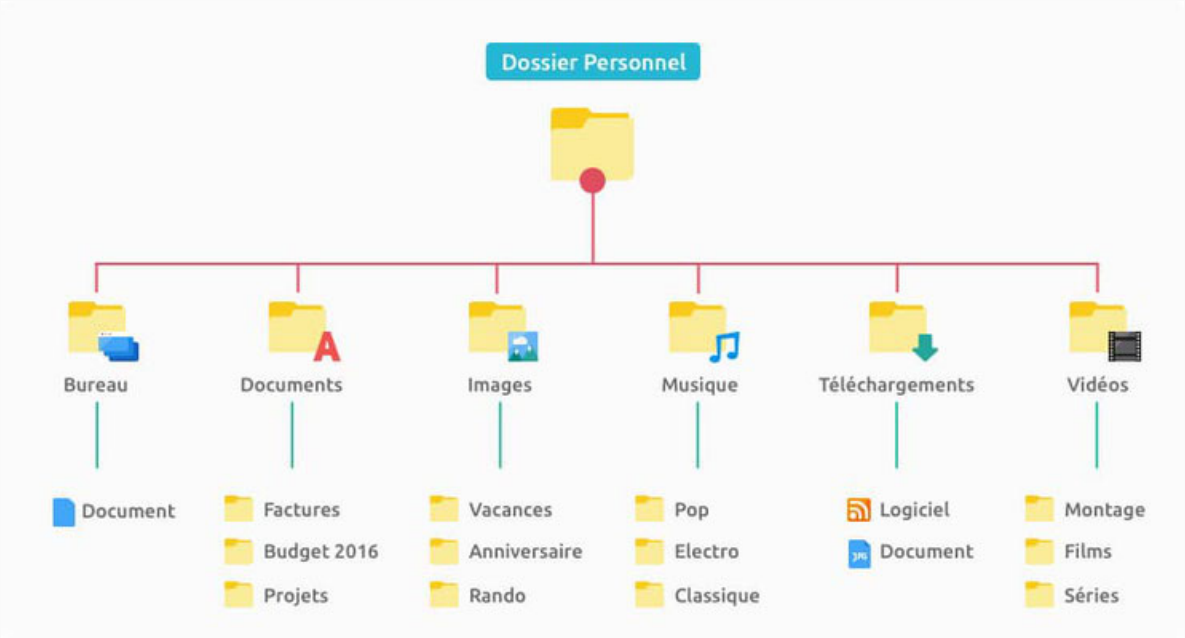
Le dossier personnel de l'utilisateur dupond est à l'adresse :

`/home/dupond`

Sa photo de vacances `plage.jpg` est à l'adresse :

`/home/dupond/Images/Vacances/plage.jpg`

Exemple de système de fichiers



Windows

Le dossier personnel de l'utilisateur dupond est à l'adresse :

`C:\Users\Dupond`

Sa photo de vacances `plage.jpg` est à l'adresse :

`C:\Users\Dupond\Images\Vacances\plage.jpg`

Langages de programmation

Comment décrire un algorithme à une machine ?

Les langages de programmation

En pratique, pour décrire un algorithme à un ordinateur, on n'écrit évidemment pas une suite de 0 et de 1 à la main !

Deux nombreux **langages de programmation** ont été développés. On peut y décrire un algorithme dans une syntaxe compréhensible par un humain, et qui est ensuite traduite en **langage machine**.

Deux grandes catégories de langages

Langages compilés

- La traduction de notre programme se fait d'un coup, lors de la **compilation**, qui génère un fichier exécutable.
- Une fois compilé, les machines utilisant le programme n'ont besoin que de l'exécutable.
- Exemples : **C**, C++, **OCaml**

Langages interprétés

- La traduction se fait en temps réel : notre programme est lu et traduit à la volée par un interpréteur.
- L'interpréteur doit être installé sur toutes les machines utilisant le programme.
- Exemples : Python, Mathlab, **Ocaml**

Comment écrire un programme ?

Le code

Le code d'un programme est simplement écrit dans un fichier texte. De ce fait, on peut choisir l'éditeur de texte que l'on veut, certains étant mieux adaptés à un langage donné (menus ad-hoc, auto-complétion, coloration syntaxique). Dans ce cas, on parle d'**environnement de développement intégré** (IDE).

Évolution des outils de programmation

Au début, les ordinateurs n'étaient pas assez puissants pour faire tourner des IDE comme aujourd'hui. La vie d'un programmeur n'était alors pas facile : après avoir taper du code dans un terminal, il fallait l'imprimer pour le relire et chercher les bugs...

Comment écrire un programme ?

Apollo 11

Ce fut le cas par exemple pour le programme qui a envoyé Apollo 11 sur la Lune en 1969.

Sur la photo ci-contre, la cheffe d'équipe des programmeurs de la NASA, **Margaret Hamilton**, pose à côté du code d'Apollo 11.

Si vous êtes curieux, ce code est disponible sur GitHub depuis 2016 :

`https://github.com/chrislgarry/Apollo-11/`

Comment écrire un programme ?

Éditeurs

Certains IDE sont spécialisés pour un seul langage de programmation.

Par exemple, pour écrire du code Python, on peut utiliser :

- Pyzo
- Spyder
- Idle
- ...

Comment écrire un programme ?

Éditeurs

D'autres IDE peuvent être utiliser pour tous les langages de programmation :

- Eclipse
- Atom
- SublimeText
- **VSCode**
- **Emacs**
- **Vim**
- ...

Comment écrire un programme ?

Éditeurs

On peut également utiliser des éditeurs de texte plus simples :

- NotePad++
- le BlocNote de Windows
- ...

La MP2I à Janson de Sailly

Qu'allons-nous utiliser ?

Système d'exploitation

En TP, nous utiliserons le système d'exploitation **Debian 11**.

IDE

Pour coder en C ou en OCaml, vous pourrez utiliser votre éditeur préféré si vous en avez déjà un.
Si vous n'en avez pas, je vous conseille de commencer avec **VSCode**.

Visual Studio VSCode