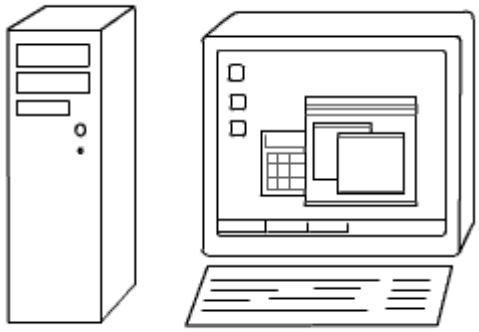
Structure en couches des systèmes informatiques

Vue simplifiée d'un système informatique

Ce que le simple utilisateur perçoit « à première vue » d'un système informatique :



- Le boîtier (tour, desktop ou portable) contient : le processeur, la mémoire, des disques, CD-ROM, DVD, …
- Des périphériques : écran, clavier, souris, imprimante, connexion réseau …
- A l'écran, des éléments graphiques: icônes, fenêtres, barre de tâches
- Dans les fenêtres: titre, menu, barre d'outils, barre d'état … boîtes de dialogues, documents
- Dans les disques, des concepts plus abstraits : les données organisées via le système de fichiers.
- En mémoire, l'ordinateur ne gère pas uniquement des données, il gère les programmes tels que le système d'exploitation, les applications, des tâches de fond.

Types de systèmes informatiques

Il n'y a pas que les PC ! Les systèmes informatiques se présentent sous des formes très variées allant des plus gros, les supercalculateurs aux plus petits tels que les cartes à puce. Les PC deviennent de plus en plus puissants et remplacent de plus en plus souvent les super-ordinateurs dans les grands centres de calculs.

C'est donc sous la forme de PC (*Personal Computer)* que se présentent le plus généralement les systèmes informatiques mais ils se présentent fréquemment aussi sous d'autres formes plus discrètes.

On trouve par exemple les « Systèmes embarqués ». Il s'agit de tous les systèmes automatisés construits autour d'un microprocesseur ou d'un microcontrôleur. Le microcontrôleur est un circuit intégré qui regroupe sur la même puce tous les éléments d'un système informatique : processeur, mémoire, et interfaces d'entrées/sorties. On trouve des systèmes embarqués dans les voitures, les appareils photos, les photocopieuses, bref dans toutes les machines où des processus peuvent être automatisés.

Les systèmes informatiques se banalisent tellement qu'on pourrait même parler d'ordinateurs jetables. C'est le cas pour certains objets sans grande valeur tels que les cartes à puce ou les cartes de vœux qui pour votre anniversaire carillonnent l'air de « *Happy birthday to you* ».

Hardware / Software

Toute description un peu avancée d'un système informatique commence par faire la distinction entre le « matériel » informatique ou « *hardware* » et ce qui est impalpable, les programmes que l'on désigne dans ce contexte par le mot « logiciel » ou « *software* »

Le hardware

PC, câbles, écran, imprimante, disques durs, CD-ROM, ports USB …
Carte mères, cartes d'extension, …
Processeur, interruptions, DMA, coprocesseurs, …

Le software

L'essentiel de la masse d'un système informatique réside dans le logiciel, les programmes.

Modularité des systèmes informatiques

Pour le simple utilisateur, le système informatique n'est autre que le support des ses applications. Il a appris à utiliser l'interface du système d'exploitation mais son fonctionnement interne ainsi que celui du matériel le concerne assez peu. Pour notre part, c'est précisément le fonctionnement du système ou plus modestement la structure de ce système qui nous intéresse et plus particulièrement celle du logiciel : « l'essentiel de la masse d'un système informatique. »

Qu'est-ce qui fait que l'utilisateur a l'impression de manipuler non pas des composants électroniques, des disques ou des codes binaires mais bien des documents, le traitement de ses données ou la communication ?

Entre le matériel dont la complexité nous échappe et l'utilisateur, quelque chose a été fait pour que les composants matériels et logiciels soient parfaitement **interchangeables** et, pour ce faire, d'une **modularité exceptionnelle**.

Modularité du hardware

Le hardware est fait d'une multitude de composants

- d'origines diverses
- conçus indépendamment les uns des autres
- d'âges différents
- de technologies différentes

Tous collaborent comme s'ils ne faisaient qu'un.

Modularité du software

Une série de modules dont les plus fondamentaux sont invisibles à l'utilisateur.

Ces modules doivent pouvoir être modifiés ou échangés facilement.

Ils doivent donc autant que possible être indépendants les uns des autres

=> Ces modules sont idéalement superposés pour former une **structure en couche**.

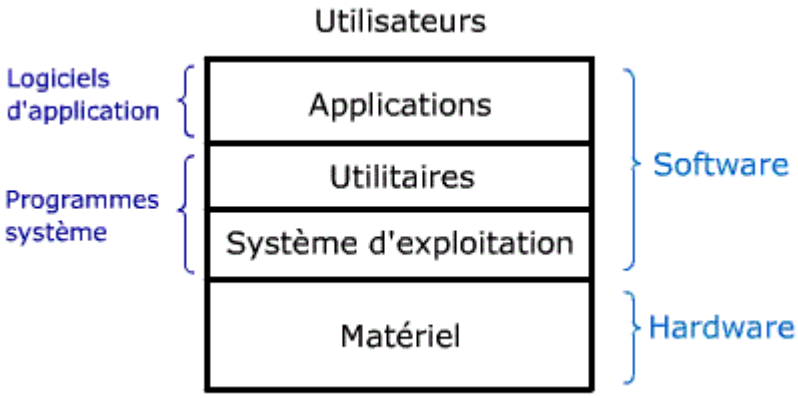
Chaque couche n'a de liaisons qu'avec les deux couches attenantes.

Couche d'abstraction

La notion de couche d'abstraction (*abstraction layer*) nous permet de décrire les systèmes informatiques comme s'il s'agissait d'empilement de couches qui se superposent en apportant à chaque niveau supplémentaire de nouvelles fonctions de plus en plus élaborées et reposant sur les fonction plus élémentaires assurées par les couches sous-jacentes.

Ces notions de **couches d'abstraction** ou de **structure en couches** seront utilisées plusieurs fois dans ces notes de cours, ici pour situer le rôle du système d'exploitation,

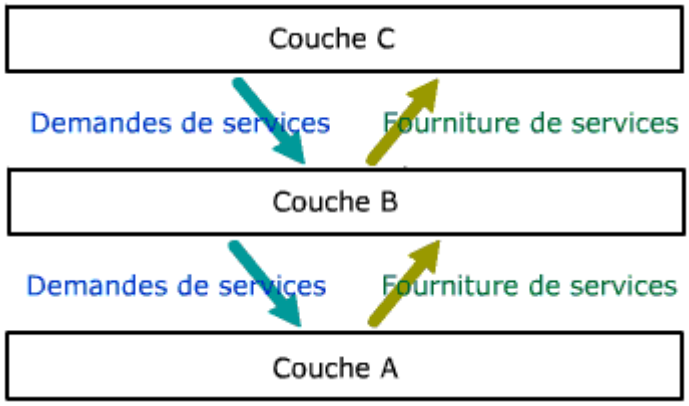
tantôt pour décrire les langages de programmation, et d'autres fois encore pour décrire l'architecture des machines, la structure des logiciels, les communications des réseaux, etc.



Matériel	Circuits électroniques / circuits logiques. On dira que c'est au niveau du matériel que se trouve la couche d'abstraction la plus basse.
S.E.	Elément le plus déterminant d'un système informatique
Applications	Traitement de texte, gestionnaire de bases de données, tableurs etc. Compilateurs, *debugger* …
Utilitaires	Services de base aux utilisateurs. Ex. interface graphique, interpréteur de commandes, gestionnaires divers qui tournent en tâches de fond, impriment, cherchent le courrier etc.
Utilisateurs	C'est à eux que le système informatique est destiné. Les utilisateurs interagissent avec la couche de plus haut niveau.

Structure en couches des logiciels

- Chaque couche est construite sur la couche précédente.
  Elle est une sorte de **machine virtuelle** qui permet de faire **abstraction des détails** qui composent les couches sous-jacentes.



- Chaque couche offre des services à la couche qui lui est supérieure et est cliente de la couche sous-jacente.
- Elle ne communique avec ces deux couches adjacentes qu'au travers d'interfaces bien définies.
- Chaque couche est seule responsable de son fonctionnement interne. Les éventuelles modifications de ce fonctionnement ne doivent pas influencer les autres couches.

Exemple : Le système de fichier

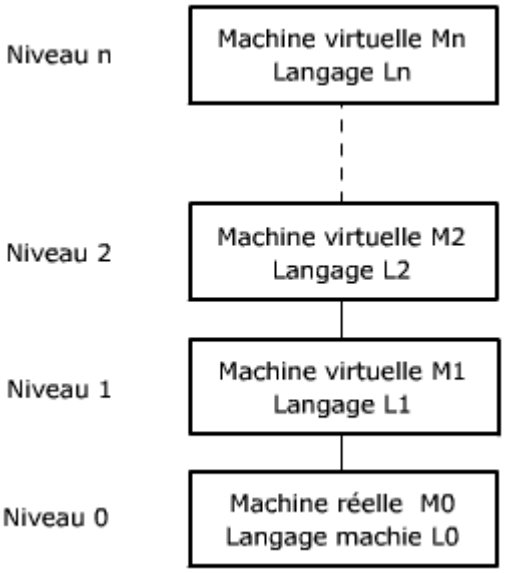
Les développeurs d'applications disposent de primitives pour ouvrir, lire, écrire et refermer un fichier comme on le ferait avec un document quelconque, un livre par exemple. La programmation de fichier est donc possible sans devoir connaître le mécanisme interne des disques, des disquettes ou d'autres lecteurs amovibles.

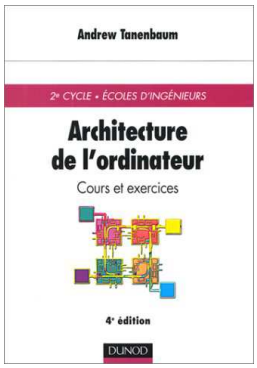
Le système de fichier est donc pour le développeur une couche logicielle dont il ne doit connaître que l'interface d'application pour pouvoir l'utiliser en pouvant faire abstraction des détails propres à la mise en œuvre des fichiers.

Couches de langages et machines virtuelles

L'illustration de cette vue en couches qui suit est empruntée à un ouvrage d'Andrew Tanenbaum dont la référence est donnée ci-dessous.

Les langages informatiques sont à considérer à plusieurs niveaux allant du plus bas, le plus proche des composants électronique au plus proche de l'utilisateur, plus indépendant du matériel.





http://www.cs.vu.nl/~ast/books/book_software.html

- Un programme est une suite d'instructions
- Soit L0, le langage machine dont les instructions sont exécutées "en dur" par les circuits électroniques de la machine M0.
- L0 est parfaitement adapté aux circuits électroniques mais pour nous, humains, son code est fastidieux !
- Il faut donc un langage L1 plus proche de l'utilisateur.
- C'est comme si une machine hypothétique M1 exécutait directement les instructions L1.Le langage L1 n'est pas très différent de L0

  On a dès lors besoin d'un langage L2 plus proche de l'utilisateur et moins dépendant de la machine
- Chaque langage s'appuie sur son prédécesseur et devient un peu plus pratique que le précédent.

Andrew TANENBAUM conçoit le système informatique comme un empilement de couches qui correspondent à des langages de plus en plus évolués, compris par des machines virtuelles de plus en plus abordables pour l'utilisateur et de moins en moins dépendantes du matériel.

Exécution d'un programme écrit en L1

Comment fonctionne la machine virtuelle M1 ?

Compilation

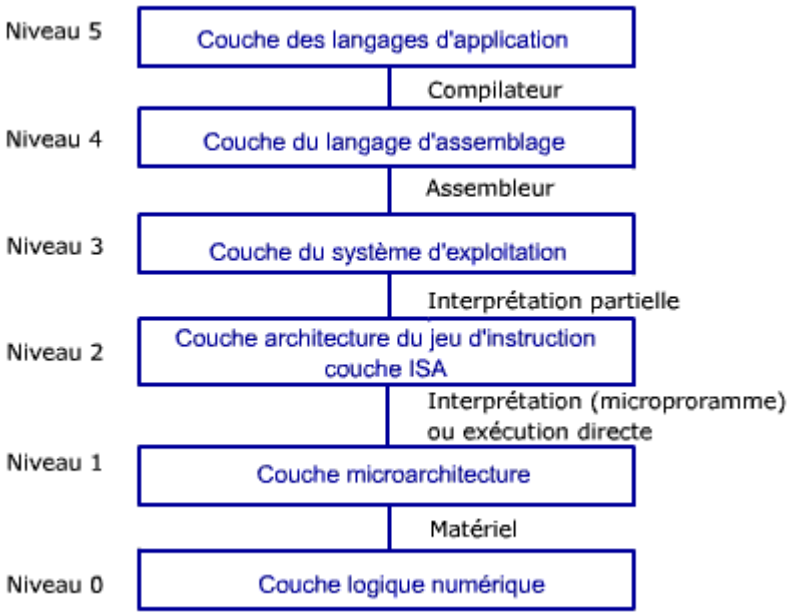
Chaque instruction L1 est remplacée par une suite d'instructions L0.
La machine M0 exécute ensuite le programme en langage L0.

Interprétation

Un programme écrit en L0 examine les instructions L1 et exécute directement des séquences d'instructions qui correspondent aux tâches demandées.

Machines multicouches

Nous nous référons ici aussi à l'approche en six couches que Andrew Tanenbaum développe dans son ouvrage « Architecture de l'ordinateur ». Maintenant que nous savons comment interagissent les langages de niveaux successivement de plus en plus élaborés, voyons quels sont réellement ces langages dans nos machines informatiques.



Niveau 0 *Couche logique numérique ou couche physique*

Tout ordinateur est conçu à partir de composants de base rudimentaires : des portes logiques qui effectuent des fonctions simples (ET, OU, Inversion, …)

Niveau 1 *Couche micro-architecture*

L'***ALU*** et les ***registres*** construits à partir des circuits logiques du niveau précédent.

L'ALU fait des opérations arithmétiques et logiques à partir de données contenues dans les registres où il range aussi les résultats de ses calculs. Ce mécanisme est commandé,

soit par un microprogramme qui interprète les instructions du niveau suivant, soit directement par le matériel.

Initialement, on l'appelait « ***couche microprogrammée*** » car un interpréteur logiciel commandait les registres et l'ALU.

On parle actuellement de « ***chemin des donnés*** » car l'ALU et les registres sont commandés par des circuits matériels.

Niveau 2 *Couche « langage machine » ou Couche ISA (Instruction Set Architecture)*

Elle développe l'architecture du jeu d'instructions = entre 50 et 300 instructions que doivent connaître ceux qui programment la machine en langage assembler. Elle est la vision que les compilateurs ont de la machine.

Une bonne architecture ISA

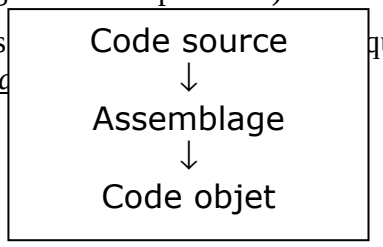
- donne une machine performante
- permet aux compilateurs de générer des codes optimisés au mieux
- permet d'exécuter les programmes anciens sans les modifier (*compatibilité amont*)

Niveau 3 *Couche système d'exploitation*

Pour les programmeurs système, cette couche ajoute aux instructions de la couche ISA de nouvelles instructions essentielles pour l'écriture du système d'exploitation.
Ces dernières sont souvent appelées « appels système » (*system calls*)

Elles apportent les trois techniques suivantes:

- La mémoire virtuelle
  pour faire comme si la machine avait plus de mémoire qu'elle n'en a réellement
- Les entrées/sorties sur fichiers
  Un fichier est une abstraction permettant de localiser l'écriture ou la lecture d'une suite d'octets (Exemple: une unité de stockage ou une imprimante)
- Le parallélisme permet à plusieurs processus quer et de se synchroniser*Niveau 4* *Couche langage d'*



Langage symbolique

- Mnémoniques et adresses symboliques
- Directives
- Macros

Le langage d'assemblage est une forme symbolique d'un des langages sous-jacents.

Des mnémoniques qui représentent les instructions sont plus faciles à utiliser pour l'écriture et la lecture du code.

Les adresses s'écrivent sous forme symbolique. Il est possible de donner des noms aux variables. Des directives d'assemblages permettent de segmenter le code et de contrôler la portée des variables ou des procédures (locales et globales / privées ou publiques)

Les macros allègent l'écriture en englobant des séries d'instructions répétitives.

Le code assembleur (*code source*) est ensuite traduit en langage machine (*code objet*).
Le programme de traduction = *assembleur*.

Il y a autant de langages d'assemblage que de microprocesseurs.

Niveau 5 *Couche langage d'application*

Langages de haut niveau :

Basic Pascal / Delphi C / C++ Java C# Python, Perl, PHP …	Les langages évolués permettent de se préoccuper uniquement de la logique du programme sans se soucier de l'organisation matérielle de la machine