

# Introduction à la micro-informatique

---

Cours Solvay 2ème Bachelor

# Comprendre la technologie du moteur

JUSQU'À 250€ D'ÉCONOMIE + FRAIS DE PORTS OFFERTS  
+ VOTRE COULEUR FAVORITE POUR 1€ DE PLUS  
+ UNE SACOCHÉ BELKIN® ROUGE ET NOIRE POUR 1€ DE PLUS !!

Offres valables sur une sélection de systèmes du 26/06 au 09/07/2008.

XPS ONE™  
**1249€ TTC\*\*** ~~1399€~~  
150€ d'économies + frais de port offerts !<sup>1)</sup>  
E-REF: PPFRS-D07X103

- Processeur Intel® Core™2 Duo E6550 (2.33GHz, 4Mo L2 Cache, 1333MHz FSB)
- Windows Vista® Édition intégrale authentique
- Mémoire 2Go DDR2
- Disque dur 320Go SATA
- Moniteur 20" intégré XPS ONE™
- Carte Graphique ATI® Radeon™ HD 2400XT avec 256Mo

XPS™ M1330  
**899€ TTC\*\*** ~~949€~~  
50€ d'économies + frais de port offerts !<sup>1)</sup>  
E-REF: PPFRS-N07X3305

- Processeur Intel® Core™2 Duo T6750 (2GHz, 2Mo L2 Cache, 1067MHz FSB)
- Windows Vista® Édition Famille Premium authentique
- Mémoire 2048Mo bicanale DDR2 SDRAM, 667MHz
- 250Go de disque dur (5400tpm)
- Écran 13.3" TrueLife™ avec caméra intégrée 2.0MP

LE VÔtre A DES COULEURS  
Optez pour l'une de nos 3 superbes couleurs.  
● ● ●

cliquez !  
**WWW.DELL.FR**

MICROSOFT® OFFICE FAMILLE ET ETUDIANT 2007 POUR 110€ TTC

DELL™ RECOMMANDÉ NOTRE

Volant publicitaire SVM N°272 juillet-août 2008 ■ FUTUR, dans 1 an, dans 10 ans... tout ce qui vous attend 7 miniportables 12 APN comm... 100% recyclé

★ MÉTHODOLOGIE

**Chaque mois, notre labo teste les matériels selon des procédures strictes. Les résultats sont combinés au jugement du rédacteur pour établir la note finale sous forme d'étoiles.**

**Notre PC de référence**

Pour obtenir des résultats cohérents et comparables, tous les tests de périphériques (cartes graphiques, imprimantes, supports de stockage, webcams...) sont réalisés sur une machine de référence. Nous en possédons cinq exemplaires identiques. Il s'agit de PC HP dc7800 avec un processeur Intel Core 2 Duo E6750 à 2,66 GHz, FSB 1333 MHz, chipset Q35, 2 048 Mo de mémoire DDR2, contrôleur vidéo intégré Intel GMA 3100 et disque dur Seagate S-ATA de 160 Go à 7 200 tours par minute. C'est en comparant les résultats obtenus par ce PC de référence à ceux des matériels testés que nous évaluons le niveau de performance des ordinateurs.

**LE TEST DES PC DE BUREAU ET PORTABLES**

**BUREAUTIQUE**

Ce test, réalisé grâce au programme PCMark Vantage, de FutureMark, mesure les performances des PC dans le cadre d'une utilisation bureautique (compression de fichiers, scan antivirus, etc.). Effectués au moins trois fois, ces tests sont invalidés si l'on constate une différence de plus de 5 % entre chaque itération.

**CRÉATION**

Cette épreuve, assez exigeante pour le matériel, est effectuée grâce aux programmes PCMark Vantage et 3DMark 06 de FutureMark. Elle sert à mesurer l'aptitude du micro-ordinateur à créer des contenus multimédias. Plus cette mesure – également réalisée au moins trois fois – est élevée, plus le PC est puissant.

**COMPOSANTS**

Les performances du processeur, du disque dur et du système vidéo sont notamment mesurées à l'aide des logiciels HD Tach, de Simpli Software, Sandra de SiSoftware et des utilitaires de gravure de Nero. La consommation électrique des PC de bureau et portables est observée par une sonde Energy Monitor 3000 de Voltcraft.

**EST DES PÉRIPHÉRIQUES**

**IMAGE**

Sur les écrans sont testés avec une mire de référence pour apprécier le contraste, le rendu des polices à différentes résolutions, la qualité des dégradés et des aplats. Nous lançons ensuite divers jeux et films pour juger la fluidité d'image et l'angle.

**AFFICHAGE**

Les écrans sont testés avec une mire de référence pour apprécier le contraste, le rendu des polices à différentes résolutions, la qualité des dégradés et des aplats. Nous lançons ensuite divers jeux et films pour juger la fluidité d'image et l'angle.

**IMPRESSION**

Les imprimantes et les multifonctions subissent plusieurs tests d'impression chronométrés : texte de 5 pages en mode rapide, rapport de 10 pages en couleur, diverses photos en qualité maxi. Et pour les imprimantes photo 10 x 15, dix photos de

**MOBILITÉ**

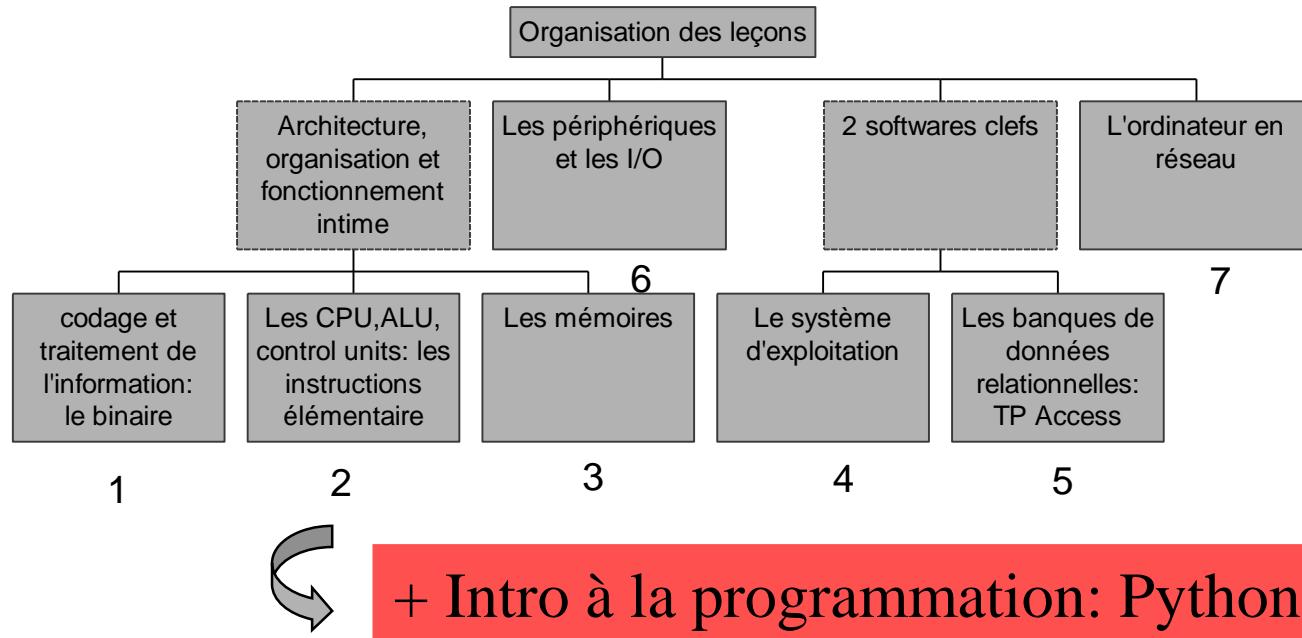
Cette catégorie regroupe tous les matériels "mobiles" (baladeurs audio/vidéo, téléphones portables, PDA...). Notre laboratoire teste leur autonomie et la vitesse de transfert des données, mais aussi leurs qualités de restitution sonore et vidéo, sans oublier

**PHOTO/VIDÉO**

C'est évidemment la qualité d'image, fixe ou animée, qui prime pour les tests de ces périphériques. La qualité de la balance des blancs est évaluée à l'aide d'une mire Kodak, et pour le respect chromatique, c'est le mode L-A-R de Phottixton d'Arbois

# Plan du cours

## Plan du cours



# Introduction

---

- L'ordinateur est aujourd'hui la machine la plus importante, vitale, complexe, complète, omniprésente, polyvalent.
- Les plus récentes révolutions technologiques trouvent leur source dans l'informatique: automatisation, robotique, courrier électronique, Smartphone, Internet, Multimédia, ..
- Il s'est infiltré partout: domotique, outils, électroménagers, informatique embarquée, automobile, finance, etc.
- Pourquoi ?? L'ordinateur est la machine qui peut se substituer au plus grand nombre d'objets, de fonctions ou d'autres machines de ce monde:

- 
- dans son rôle de stockage organisé de données: bibliothèque, vidéothèque, eBook, ipod, médiathèque, album photos, banque de données, Cloud, sites Web --> Mémoire du monde
  - dans sa fonction de présentation organisée et interactive de ces mêmes données: c'est son côté TV, CD-ROM, Album, Vidéo, Multimédia, DVD,..
  - dans son support à la communication: Internet, téléphone (VoIP), Smartphone, email,....
  - Calcul à grande vitesse: Modélisation, simulation, analyses de données

# L'ordinateur, omniprésent



- 
- Il peut se substituer à l'Homme ?? Intelligence Artificielle (IA):
    - » jeux d'échec,
    - » système expert, aide à la décision
    - » Les robots trader ultrarapides → 50% du marché boursier
    - » vision artificielle, reconnaissance de la parole,
    - » créativité picturale et musicale,
    - » Classement automatique de documents , recommandations

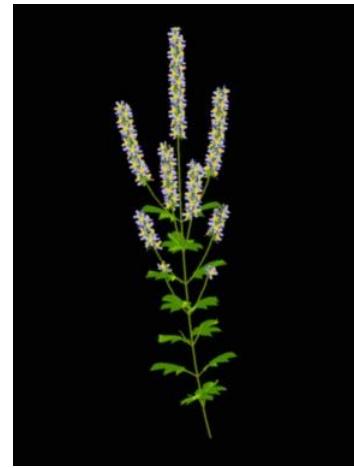
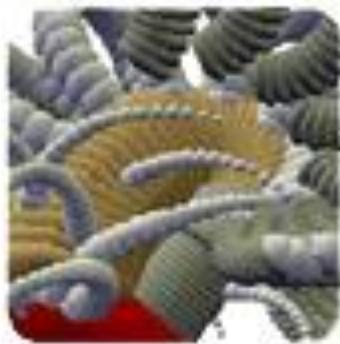


# Les exploits de l'Intelligence Artificielle



- 
- il peut se substituer à la Vie: Vie Artificielle (VA):
    - » robotique animale
    - » jeux de la vie
    - » simulations biologiques, morphogenèses des plantes et organismes
    - » algorithmes génétiques
    - » réseaux de neurones
  - il peut se substituer à la Nature: Réalité Virtuelle
    - » jeux interactifs (3D), simulateurs de vol,
    - » Simulateurs de systèmes complexes

Tous ces artefacts sont-ils réellement « vivants » ?



virus

a  
u  
t  
o  
r  
o  
p  
e  
r  
o  
t  
o  
n

# Pourquoi l'informatique intéresse l'économiste?

---

- Comme sujet d'étude:
  - L'informatique est un booster de croissance et de productivité
  - L'information dématérialisée est un bien très particulier
  - Les marchés électroniques ont des propriétés particulières
  - Les effets de la mise en réseau

# Pourquoi l'informatique intéresse l'économiste?

---

- Sous l'angle du gestionnaire:
  - L'un des principaux centres de coûts des entreprises (e.g. Banque globale: 7 milliards de \$)
  - Un moteur de transformation (industrielle et organisationnelle)  
(e.g. e-banking, industrie musicale, photographie, etc.)
  - Un outil de gestion  
(Comptabilité, gestion des clients (CRM), de la chaîne d'approvisionnement (SCM), des ressources humaines (HRM), gestion intégrée (ERP), veille stratégique (BI), etc.)

# Pourquoi l'informatique intéresse l'économiste?

---

- Comme outil de travail:
  - Statistique, économétrie, simulations se passent difficilement de l'informatique
  - Echange et partage de données
  - Diffusion de travaux et accès à la connaissance

# Au fait, qu'est-ce l'informatique?

SYSTEMES D'INFORMATION (SI)  
sont composés de:

TECHNOLOGIE

=

Technologies de  
l'information et de  
la communication  
(ICT)

=

Technologies de  
- Stockage  
- Traitement  
- Transmission  
de l'information

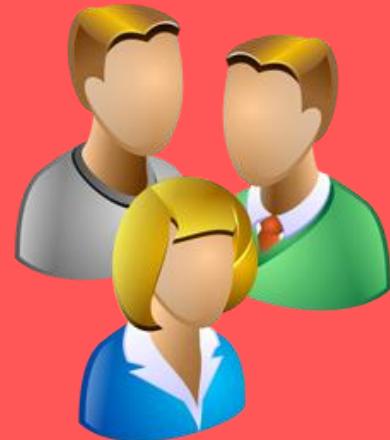
PROCESSUS



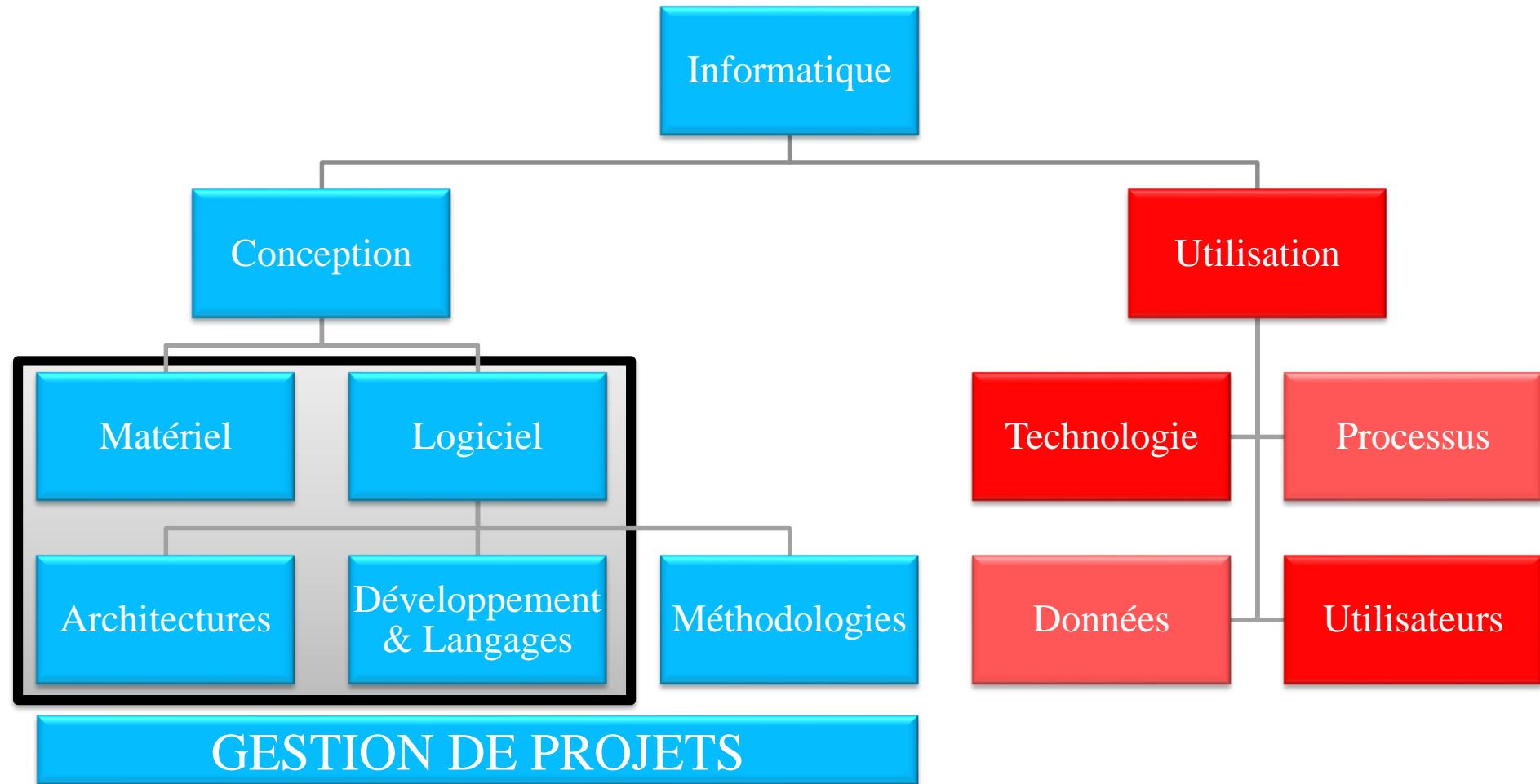
DONNEES



UTILISATEURS



# Les deux facettes de l'informatique



# Objectifs du cours

---

- Vous initier aux fondements et aux développements récents de l'informatique
  - Contribuer à faire de vous des utilisateurs de l'informatique plus avertis
  - Vous sensibiliser à l'impact de l'informatique sur l'entreprise et sur l'économie
  - Vous permettre de dialoguer plus efficacement avec des informaticiens
- Vous familiariser avec la démarche algorithmique et la programmation, outils vitaux de l'économiste

# Philosophie du cours

---

- Choix d'un enseignement transversal qui balaie plutôt que n'approfondit
- Acquisition d'une culture informatique attaquant les différents niveaux informatique, mais en superficie
- Le cours passe en revue le codage binaire, le microprocesseur, les mémoires, les I/O, les OS, les bases de données (DB), les réseaux, les bases de l'algorithme et de la programmation
- Le cours est complété par le stage intensif et par des lectures

# Organisation du cours théorique

---

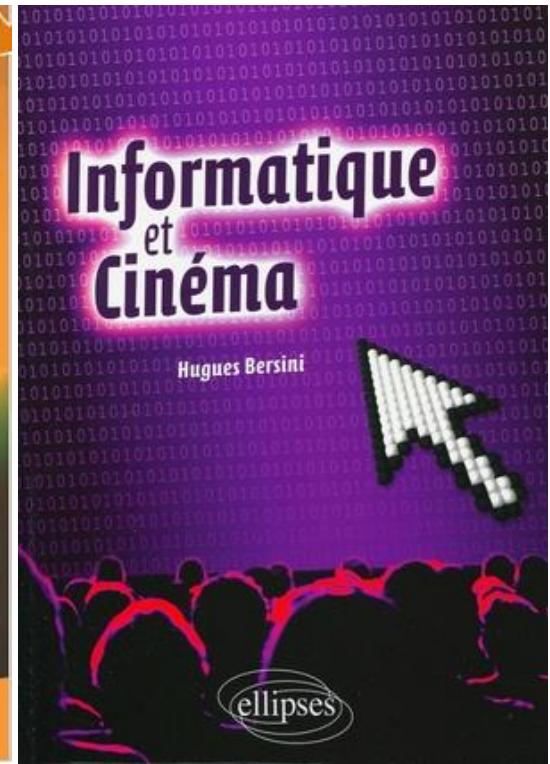
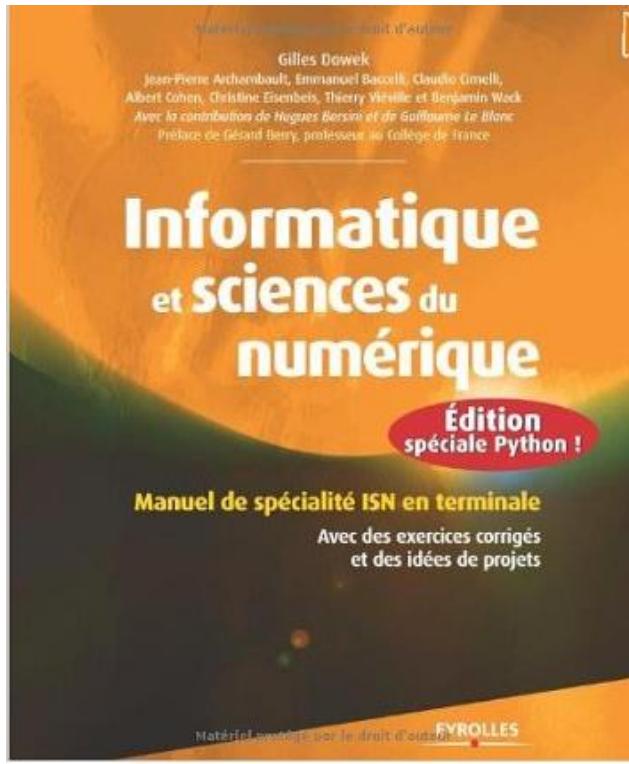
- Matériel: Organisation et fonctionnement intime
  - Codage de l'information: le binaire
  - Traitement de l'information: processeur et instructions élémentaires
  - Les mémoires
  - Les périphériques et les entrées-sorties
- Logiciels
  - Le systèmes d'exploitation
  - Les bases de données relationnelles
  - Logiciels commerciaux et logiciels d'entreprise
- Introduction à l'algorithme et à la programmation
  - Démarche algorithmique
  - Programmation procédurale
  - Langages et architectures de base
- Réseaux

# Des lectures complémentaires sont obligatoires

---

- Ce cours écrit ne se suffit pas à lui-même !!
- Les transparents soulignent les principaux éléments de la matière.
- L'avantage c'est la flexibilité et l'adaptabilité, capitales en informatique, le désavantage c'est la “rudesse” de la présentation
- Un transparent est un support à une présentation orale, ou la conséquence d'un “stabilo boss”
- A vous de compenser par des lectures additionnelles.

# Références

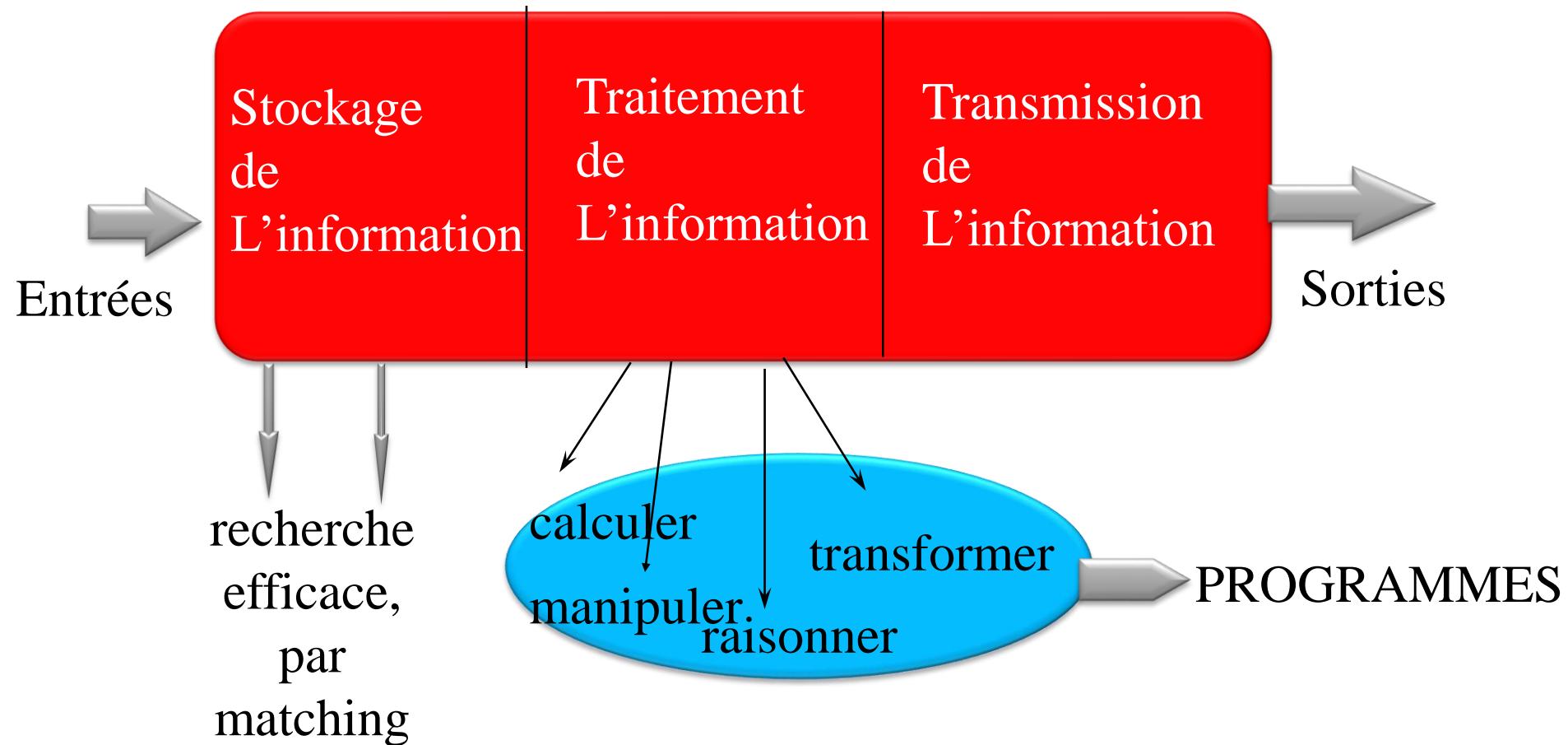


# I. Codage et traitement de l'information: le binaire

---

# Fonctionnement intime de l'ordinateur

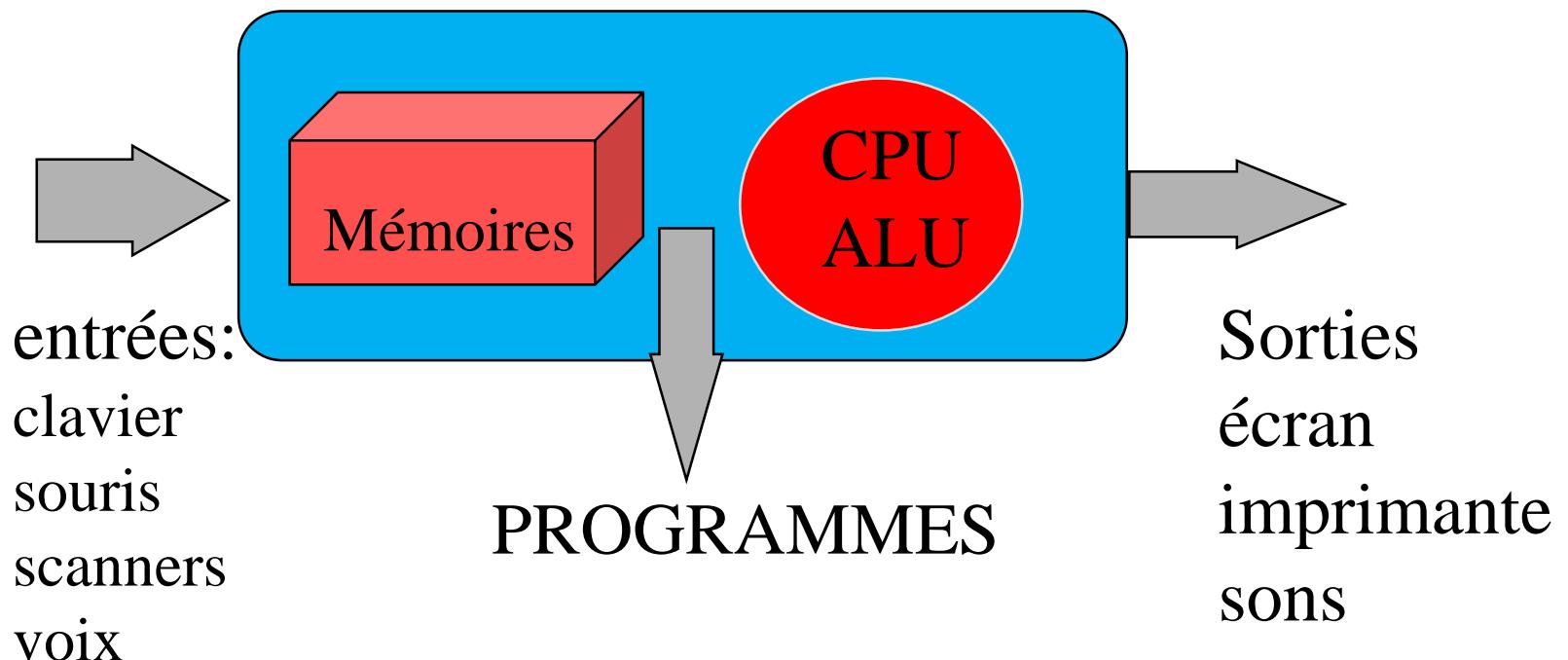
## *Structure et Fonctions Premières de L'Ordinateur*



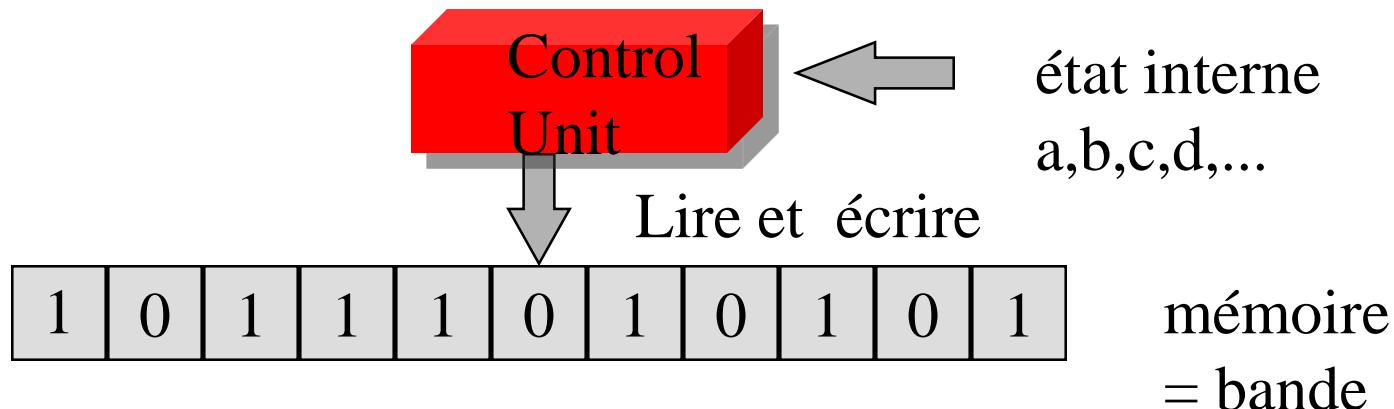
# La Partie Intelligente de l'Ordinateur: Le Programme

---

D'où son organisation Hardware



# L'abstraction suprême d'un ordinateur = la machine de Turing: tout ordinateur peut se ramener à cela.



L  
e

Programe	Etat courant	lu sur la bande	écrit sur la bande	bouge gau/dro	Nouvel état
	a	1	0	g	c
	b	0	0	d	a

# Alan Turing (1912-1954)

---



# Le binaire: en stockage et en traitement

---

- Pourquoi: le courant passe ou passe pas, idem pour la lumière et le magnétisme.
- C'est plus robuste et résiste beaucoup mieux aux perturbations.
- Plus simple et plus économique à réaliser.
- Les transistors (composants fondamental des microprocesseurs - 100000000 sur une puce) font office d'interrupteur + amplification du signal (utile quand ils sont en série)
- Circuiterie électronique = circuiterie logique

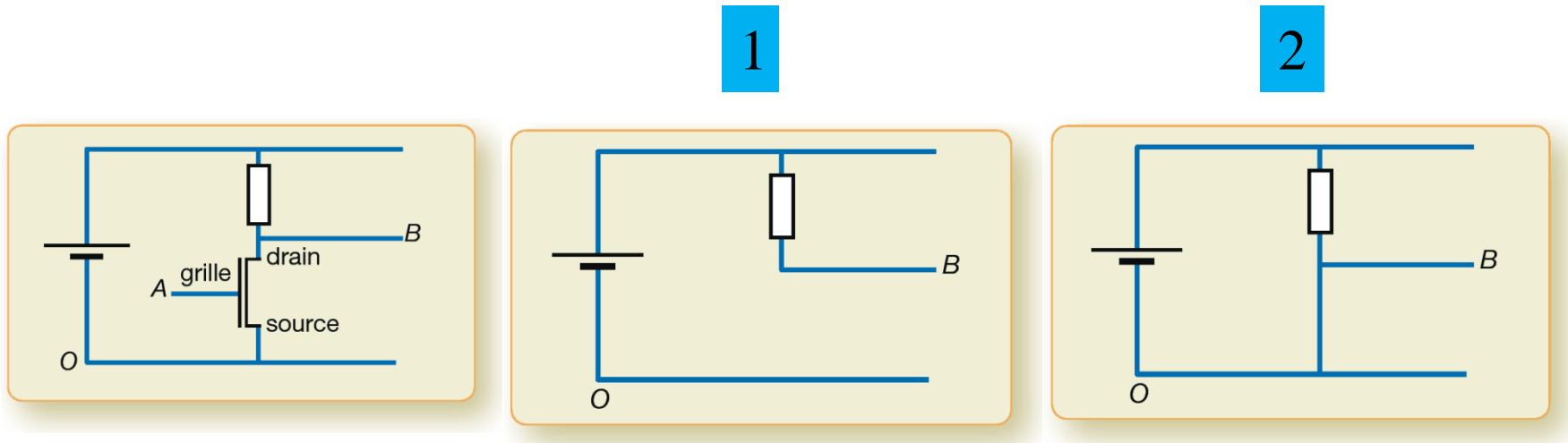
- 
- Les circuits sont des implémentations matérielles des fonctions logiques (booléennes)
  - Utilisation de la logique booléenne (binaire) pour le traitement calculatoire, logique ou symbolique de l'information
  - Stockage des données binaire permanent: magnétique, optique - ou volatile: électronique
  - Toute information est stockée en binaire: lettres, son, image,..

# Transistor

---

- Chaque élément doit pouvoir réaliser 3 fonctions
  - Amplifier le courant (utile s'ils sont en série)
  - Bloquer le courant
  - Inverser le courant
- Idéalement, il faut un élément capable de réaliser ces opérations le plus vite possible, si possible à une vitesse proche de celle de la lumière
- C'est exactement ce que fait le transistor

# Transistor = Inverseur ou interrupteur → Principe de fonctionnement



Si tension petite entre A et O, le transistor est bloqué comme en 1 et la tension entre BO = alimentation (bit A = 0, bit B = 1).  
Si tension grande entre A et O, le transistor est passant et la tension BO = 0 (bit A = 1, bit B = 0).

# Transistor

---

- A la base: le silicium (i.e. du sable), le plus abondant après l'oxygène et un semi-conducteur.
  - Sous forme de cristaux purs: parfaitement isolant
  - Mélangé
    - » A du phosphore: libère des électrons → Cristaux de type N
    - » A du bore: capture des électrons → Cristaux de type P
  - Assemblage de cristaux des 2 types → Courant circule volontiers de N vers P, mais pas dans l'autre sens
  - Différentes combinaisons possibles des deux types fournissent les différents types de transistors

# Transistor

---

- Exemple de transistor: MOS (Métal et Oxyde de Silicium)
  - Catégorie des FET (Field-Effect Transistors)
  - 2 Zones de type N: Source et Drain
  - Séparées et posées à la surface d'un substrat de type P
  - Recouverts d'une mince couche isolante d'oxyde de silicium
  - Surmontés d'une mince grille métallique (e.g. aluminium)

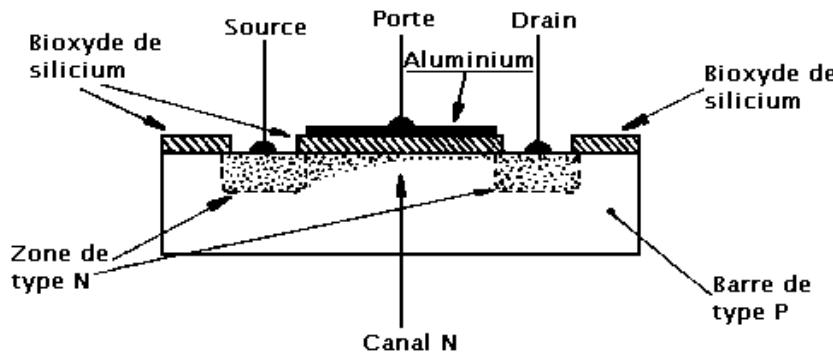
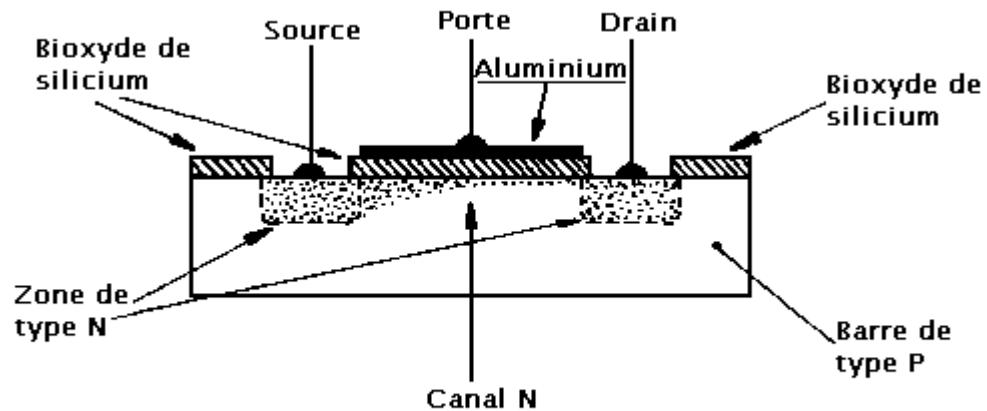


Fig. 6. – Structure d'un MOS FET avec canal N par enrichissement.

# Transistor

- Exemple de transistor: MOS (Métal et Oxyde de Silicium)
  - Si on applique une charge négative à la source, il ne se passe rien (courant ne circule pas du substrat P vers le drain N)
  - Par contre si on applique en même temps une tension positive à la grille, champ électrique se crée qui attire les électrons du substrat et crée un canal N qui permet à la charge de la source d'affluer vers le drain
  - Sitôt qu'on enlève la tension de la grille, les électrons refluent dans le substrat, et le courant cesse de circuler

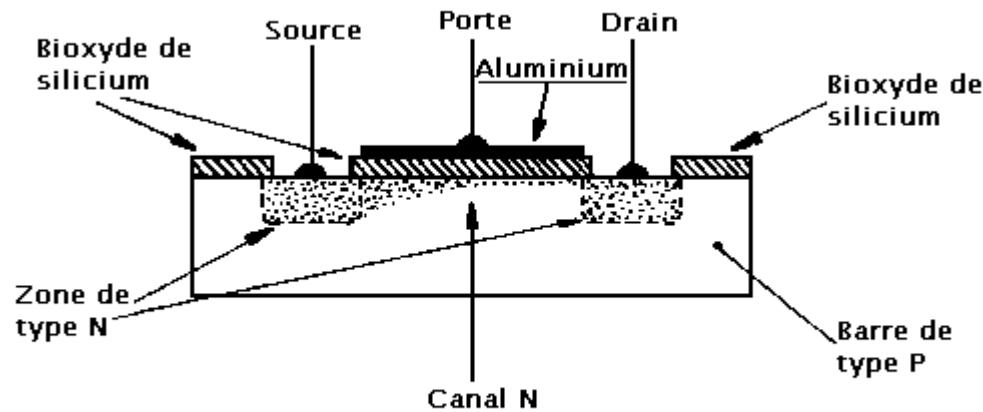


**Fig. 6. – Structure d'un MOS FET avec canal N par enrichissement.**

# Transistor

---

- Exemple de transistor: MOS (Métal et Oxyde de Silicium)
  - C'est donc bien un interrupteur de courant entre la source et le drain, commandé par le courant appliqué à la grille
  - En l'occurrence, c'est un interrupteur rapide, miniaturisé, et qui consomme très peu d'énergie



**Fig. 6. – Structure d'un MOS FET avec canal N par enrichissement.**

# Transistors et circuiterie

---

- Un processeur moderne, c'est un ensemble intégré (une puce de quelques centimètres carrés) qui contient des centaines de millions de transistors
- Au lieu de les produire séparément et de les assembler sur un circuit imprimé (cf. les années 1950), on les grave désormais au sein d'une seule galette (wafer) de silicium
- La gravure repose sur des procédés industriels d'une extrême précision (on grave à la dizaine de nanomètre près) et sur des matériaux d'une extrême pureté

# Transistors et circuiterie

---

- Plus finement on grave, plus on peut mettre de transistors sur une même puce
  - augmente la capacité de traitement
  - et réduit le délai de transmission entre deux transistors
- Mais
  - on augmente aussi la complexité et le coût de la gravure,
  - on atteint peu à peu les limites quantiques (en gravant trop finement, les électrons finissent par ‘sauter’ par effet de tunnel, provoquant des dysfonctionnements),
  - et les puces dégagent de plus en plus de chaleur
- Néanmoins, jusqu’à ce jour, la ‘loi de Moore’ se vérifie
  - Tous les 18 mois, la densité des transistors sur une puce double
  - Entre les années 1970 et 2000 : 20000 fois plus de transistors

# Transistors et circuiterie

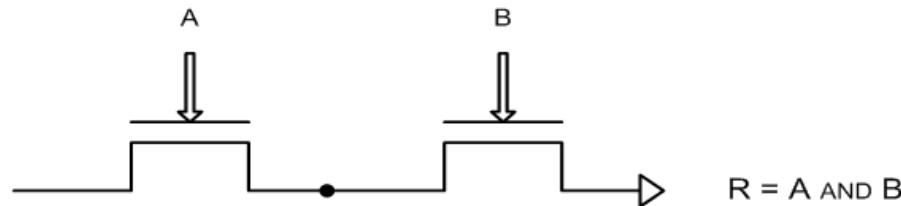
---

- Traitement de l'information binaire
  - Mis en série ou en parallèle, les transistors peuvent réaliser des fonctions booléennes élémentaires comme le “AND”, le “NOR”, le “XOR”,...
  - Circuiterie électronique = circuiterie logique
    - » Les circuits sont des implémentations matérielles des fonctions logiques (booléennes)
  - Utilisation de la logique booléenne (binaire) pour le traitement calculatoire, logique ou symbolique de l'information

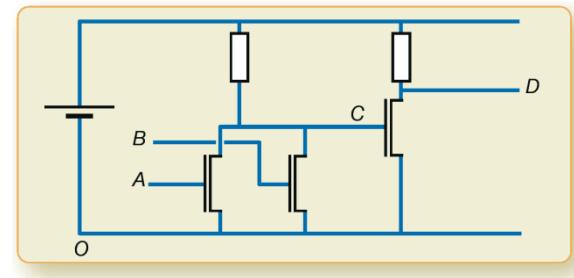
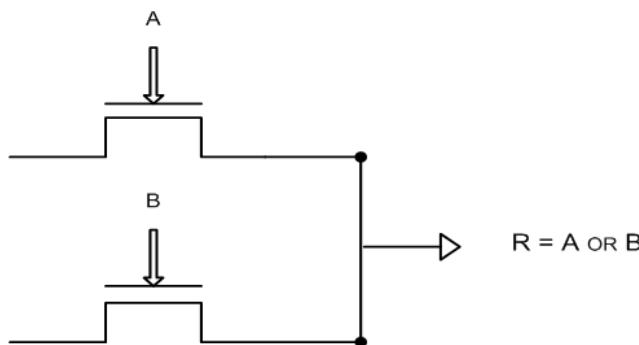
# Transistors et circuiterie

---

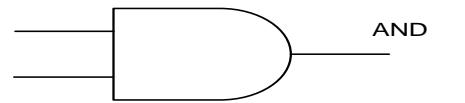
- Mise en série de 2 transistors (AND)



- Mise en parallèle de 2 transistors (OR)



# Représentation de la logique binaire

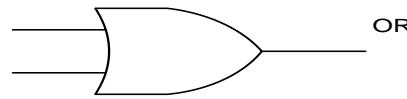


AND

A	B	R
0	0	0
0	1	0
1	0	0
1	1	1

R est VRAI si sont VRAIS

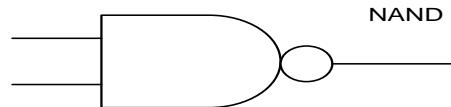
A et B



OR

A	B	R
0	0	0
0	1	1
1	0	1
1	1	1

A ou B



NAND

A	B	R
0	0	1
0	1	1
1	0	1
1	1	0

Pas (A et B)



NOR

A	B	R
0	0	1
0	1	0
1	0	0
1	1	0

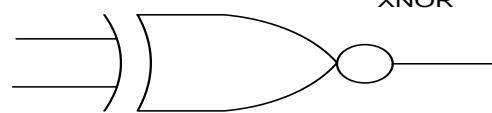
Pas (A ou B)



XOR

A	B	R
0	0	0
0	1	1
1	0	1
1	1	0

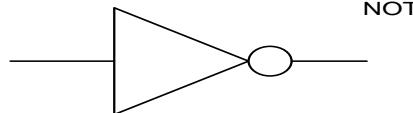
(A ou B) et pas (A et B)



XNOR

A	B	R
0	0	1
0	1	0
1	0	0
1	1	1

(Ni A ni B) ou (A et B)



NOT

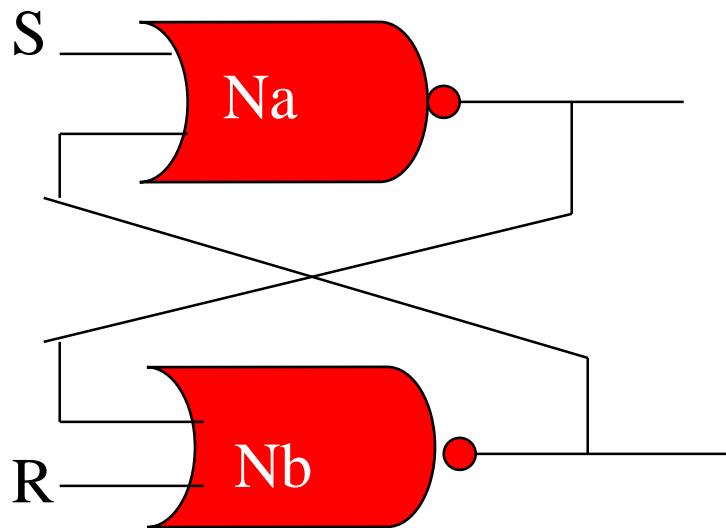
A	R
0	1
1	0

R est l'inverse de A

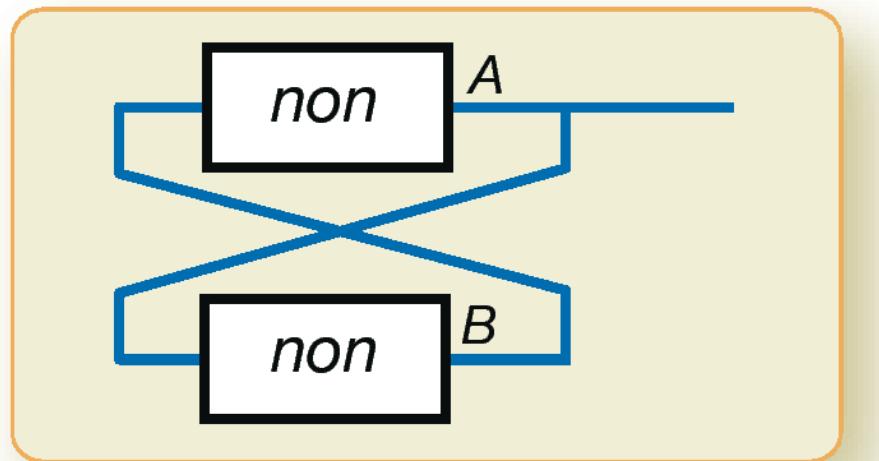
# Transistors et circuiterie

Le bistable (flip-flop) = un élément de mémoire à deux états stables.

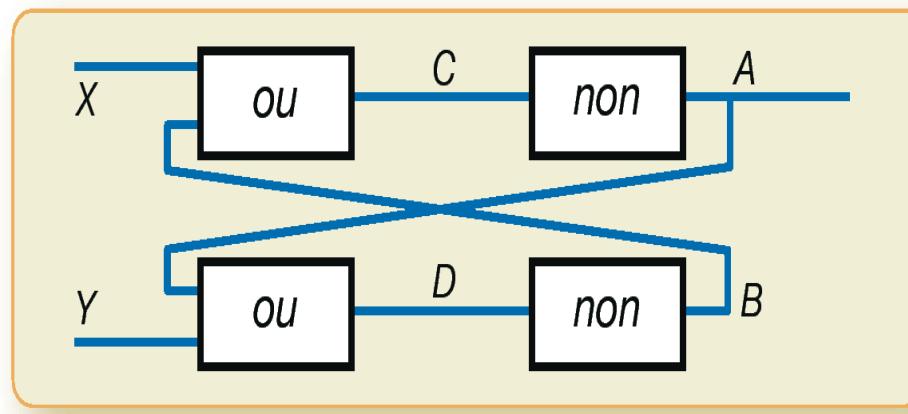
- Mis en boucle, les transistors peuvent “pièger” la valeur d’un signal électrique à un instant donné et cette valeur restera stable → on aura “mémorisé” cette valeur. C’est la mémoire électronique (RAM)
- En fait, c’est deux portes NOR mises en parallèle et bouclées sur elles-mêmes (sortie de Na = entrée de Nb et inversement)



# Mémoire d'un bit → principe



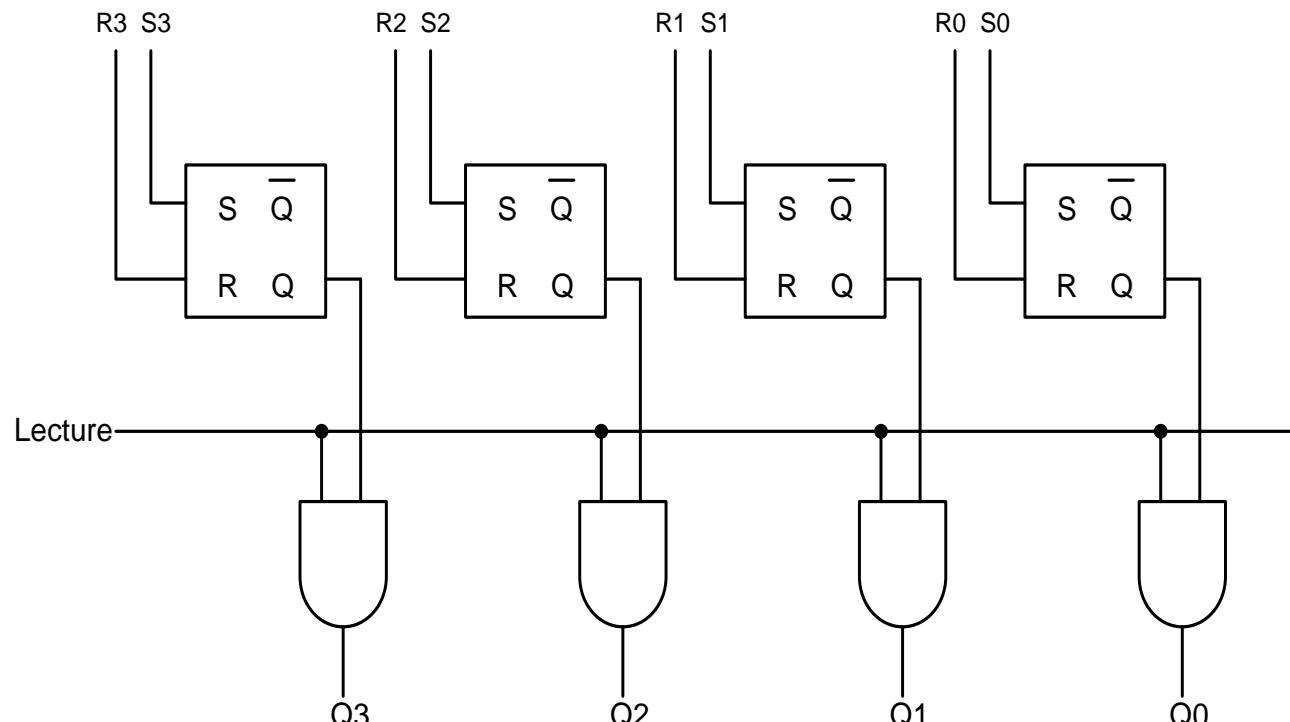
$A = 1, B = 0$  est stable  
 $A = 0, B = 1$  est stable



Si  $A = 1$ , les deux portes « ou » sont neutres. Mais si  $X = 1$ ,  $A$  passe à 0 et y reste. Et si  $Y = 1$ ,  $A$  passe à 1 et y reste. Ce circuit mémorise une valeur 0 ou 1 et, en stimulant l'entrée  $X$  ou l'entrée  $Y$ , on change la valeur mémorisée.

# Transistors et circuiterie

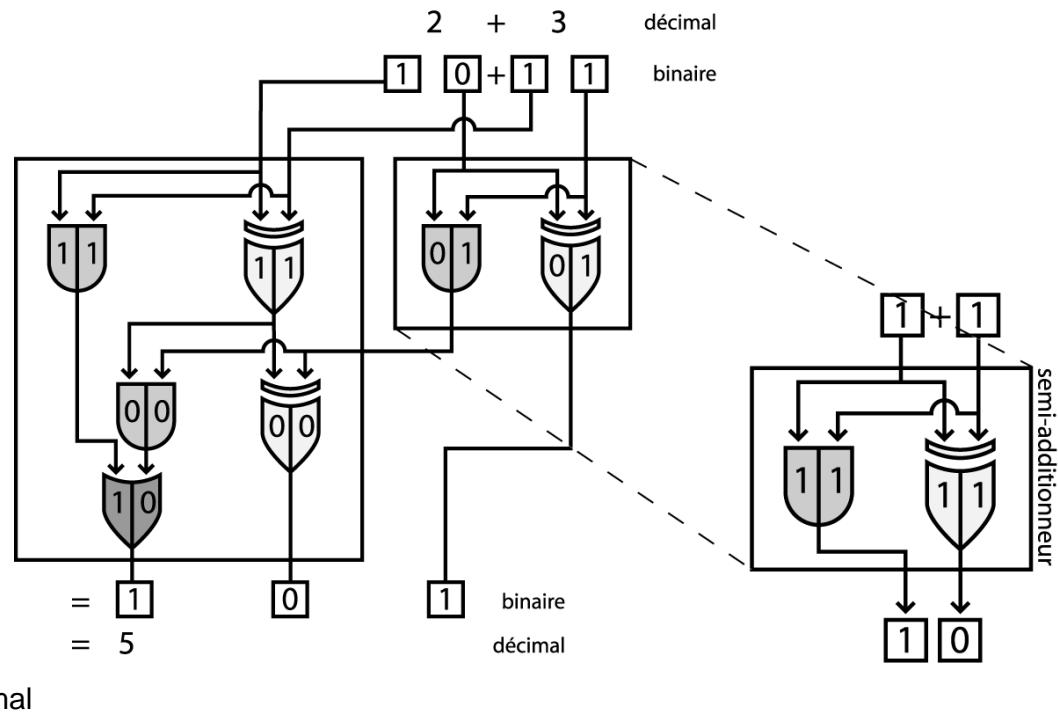
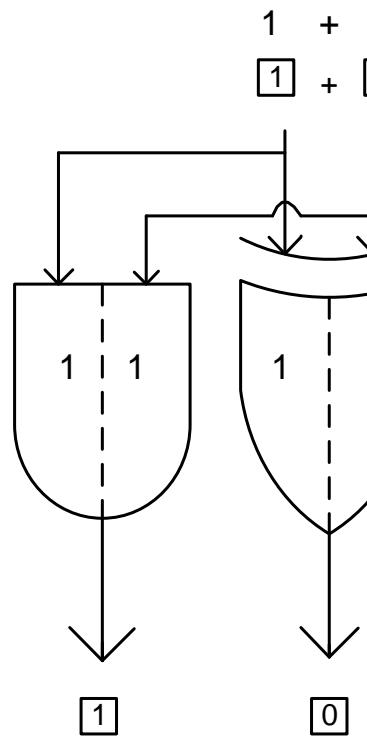
- Un registre à 4 bistables (= une mémoire de 4 bits)



**Registre à 4 bistables**

# Transistors et circuiterie

- Le transistor peut aussi effectuer des opérations arithmétiques (un XOR pour l'addition d'un bit, un AND pour la retenue, et un étage par degré à additionner)



# Transistors et circuiterie

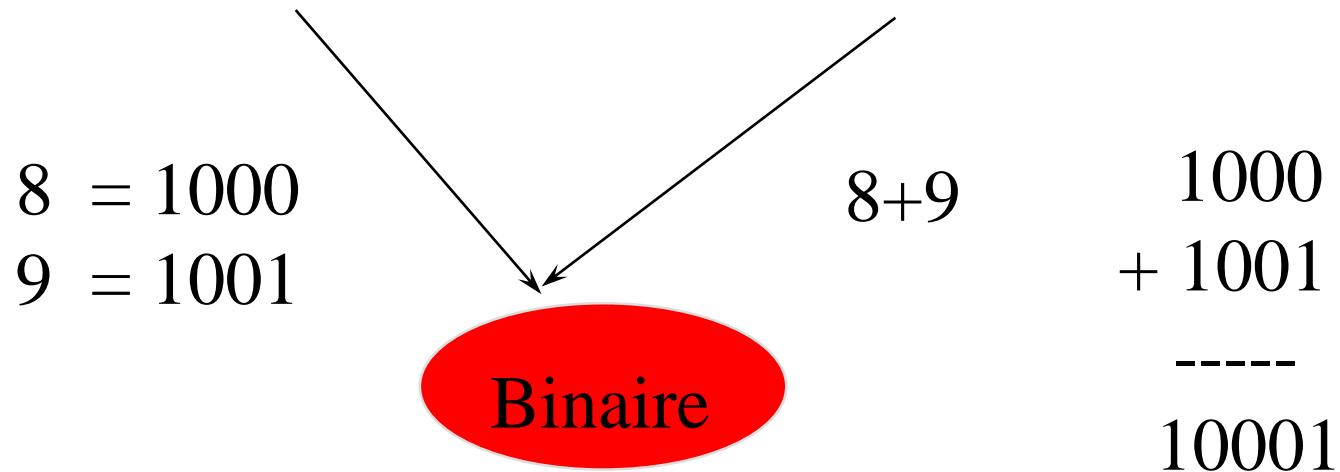
---

- Par leur fonctionnement propre, les transistors en série ré-amplifient le signal qu'ils reçoivent
- En changeant le potentiel en entrée, on change le potentiel en sortie. Le temps de changement est inférieur à la nanoseconde mais il explique pourquoi tous les circuits logiques ont un temps de réponse
- Les mêmes “briques” électroniques permettent donc de: raisonner, calculer, mémoriser,...
- Recherche en cours: interrupteur plus rapide :
  - Les nanotubes de carbone: plus petit, moins consommateur
  - Transistors optiques: 100000 fois plus rapide.

# INFORMATIQUE =

---

Information + Traitement de cette Information



Le sens d'un bit dépendra de son contexte d'utilisation  
D'où la nécessité d'établir des standards

# Information Binarisée: Ecriture, Nombres, Images, Sons

---

- La valeur d'un mot binaire dépend du contexte d'utilisation: mot, images, sons,...
- On vise la standardisation
- Ecriture = Code ASCII → UTF-8
  - Sur 7 bits --> 128 caractères, Sur 8 bits --> 256 caractères, par ex.  
«a» = 1100001
  - Equivalence Bytes (8 bits) --> Texte
  - 1.4 MBytes = 500 pages (1 page = 3000 char)
  - UNICODE (16 bits ou 32 bits) pour étendre les caractères
  - latin-1, latin-2, UTF-32, UTF-8 (le standard actuel)

Caractère	Code binaire	Caractère	Code binaire	Caractère	Code binaire
0	00110000				
1	00110001	A	01000001	a	01100001
2	00110010	B	01000010	b	01100010
3	00110011	C	01000011	c	01100011
4	00110100	D	01000100	d	01100100
5	00110101	E	01000101	e	01100101
6	00110110	F	01000110	f	01100110
7	00110111	G	01000111	g	01100111
8	00111000	H	01001000	h	01101000
9	00111001	I	01001001	i	01101001
		J	01001010	j	01101010
		K	01001011	k	01101011
		L	01001100	l	01101100
		M	01001101	m	01101101
		N	01001110	n	01101110
		O	01001111	o	01101111
		P	01010000	p	01110000
		Q	01010001	q	01110001
		R	01010010	r	01110010
		S	01010011	s	01110011
		T	01010100	t	01110100
		U	01010101	u	01110101
		V	01010110	v	01110110
		W	01010111	w	01110111
		X	01011000	x	01111000
		Y	01011001	y	01111001
		Z	01011010	z	01111010

Extrait d'une table de conversion ASCII

# Codage binaire: les nombres entiers

---

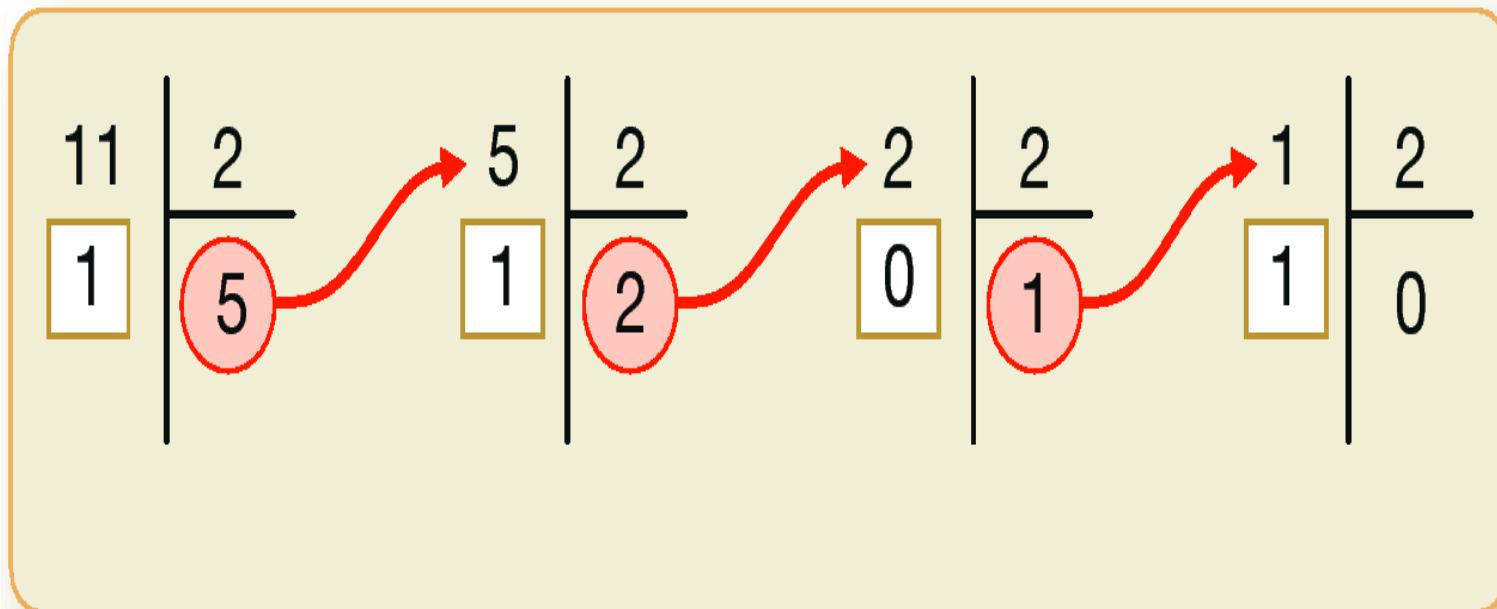
## □ La clé: comment interpréter les bits?

- Supposons des chiffres entiers non signés codés sur 8 bits

Position	8	7	6	5	4	3	2	1
Signific.	$2^7$	$2^6$	$2^5$	$2^4$	$2^3$	$2^2$	$2^1$	$2^0$
Valeur	128	64	32	16	8	4	2	1

- Exemples:
  - »  $00000000 = 0$
  - »  $00000001 = 1$
  - »  $10000000 = 128$
  - »  $01010101 = 85 = 0 \times 128 + 1 \times 64 + 0 \times 32 + 1 \times 16 + 0 \times 8 + 1 \times 4 + 0 \times 2 + 1 \times 1$
  - »  $11111111 = 255 = 1 \times 128 + 1 \times 64 + 1 \times 32 + 1 \times 16 + 1 \times 8 + 1 \times 4 + 1 \times 2 + 1 \times 1$

# Trouvez la valeur en base 2 de 11



Le résultat est : 1011

# les nombres entiers

---

- Codage binaire des nombres
  - Entiers non-signés:
    - » Sur 8 bits:
      - De 0 à 255 = 256 valeurs en tout ( $2^8$ )
    - » Sur 16 bits:
      - De 0 à 65535 = de 0 à  $2^{16}-1$
    - » Sur 32 voire 64 bits?
      - Quel est le plus grand entier codé sur 32 bits?
        - $2^{32}-1 = 4.294.967.295$
      - Et sur 64 bits?
        - $2^{64}-1$

# Codage binaire: les nombres entiers

---

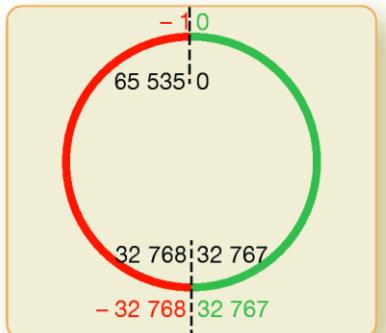
- Arithmétique élémentaire: l'addition d'entiers non signés (ici codés sur 8 bits)

$$\begin{array}{rcl} 8 & = & 00001000 \\ + 9 & = & 00001001 \\ \hline & & \\ = 17 & = & 00010001 \end{array}$$

# les nombres entiers négatifs

---

- Codage binaire des nombres
  - Entiers signés: notation dite en 2's complément
    - » Le premier bit informe sur le signe et est appelé: «le bit le plus significatif»
    - » Pour les nombres positifs: à part le premier bit à 0, cela ne change rien
    - » Pour les nombres négatifs:
      - Prendre la notation binaire de la valeur absolue
      - Inverser chaque bit
      - Ajouter la valeur 1 au résultat
      - Exemple sur 4 bits:  $3 = 0011 \rightarrow -3 = 1100 + 0001 = 1101$
    - » L'espace codable est compris entre  $-2^{n-1}$  et  $2^{n-1} - 1$  où n représente le nombre de bits.  
la notation en 2's complément permet de n'avoir qu'un seul zéro et de traiter l'addition de nombres négatifs et positifs de la même manière



---

Nombre	Binaire
-8	1000
-7	1001
-6	1010
-5	1011
-4	1100
-3	1101
-2	1110
-1	1111
0	0000
+1	0001
+2	0010
+3	0011
+4	0100
+5	0101
+6	0110
+7	0111

# Nombres entiers: arithmétique élémentaire

---

- Travaillons avec 4 bits en 2'complement

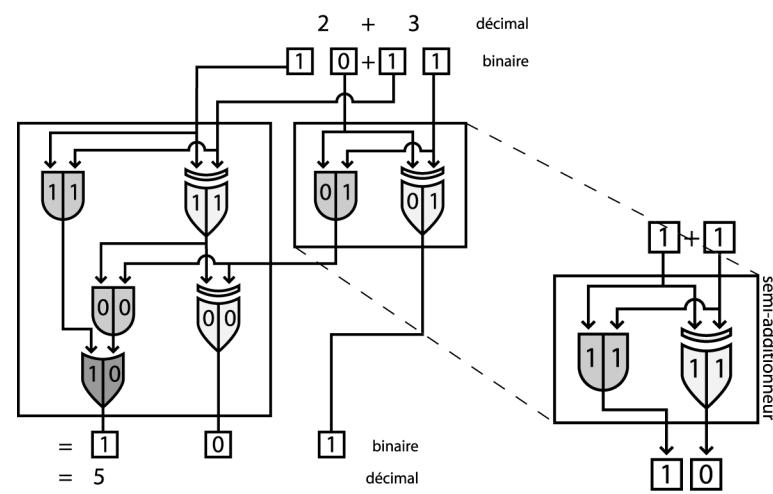
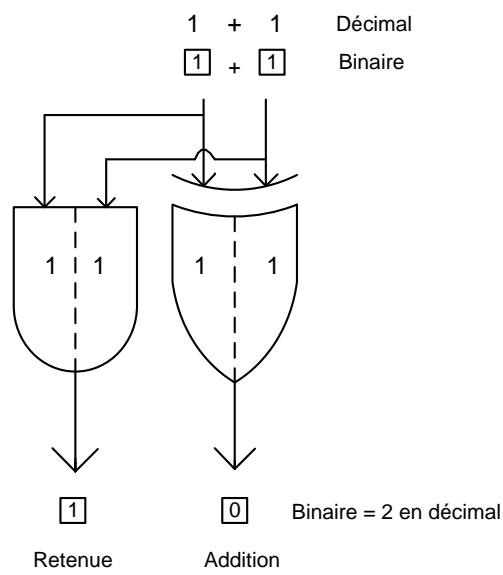
$$\begin{array}{r} 0011 \text{ (3)} \\ + 0010 \text{ (2)} \\ \hline 0101 \end{array} \quad \begin{array}{r} 0011 \text{ (3)} \\ + 1110 \text{ (-2)} \\ \hline 0001 \end{array} = 0001 \text{ (on se débarrasse du dernier bit)}$$

- $$\begin{array}{r} 1010 \\ + 1100 \\ \hline 0110 \end{array}$$
(impossible, overflow car le bit significatif est différent des deux nombres --->  
détection automatique des overflows)

# multiplication additions

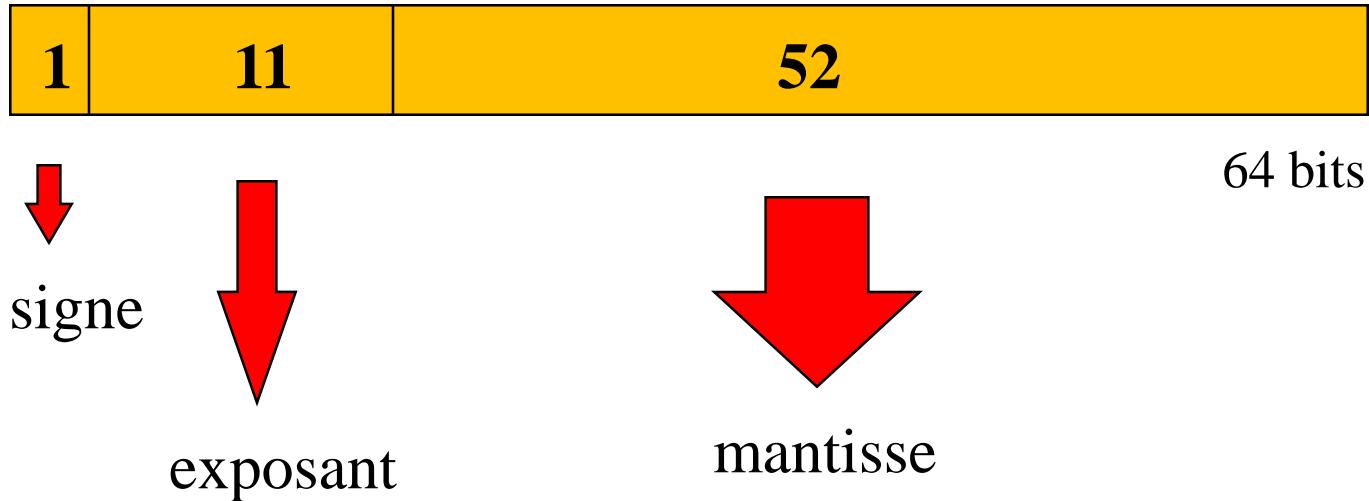
---

$$\begin{array}{r} 00011 \quad (3) \\ * \quad \underline{00011} \quad (3) \\ 00011 \qquad \qquad \text{ADD et SHIFT} \\ 00011 \\ 00000 \\ 00000 \\ \underline{\underline{00000}} \\ 000001001 \end{array}$$



# Codage et calcul de nombres à virgule flottante (floating-point)

---



En général, les nombres sont représentés après la virgule:  $1.32 \cdot 10^7$ . Donc, le « 1 » est par défaut. L'exposant est compris entre -1022 et 1023 (on le trouve en supprimant 1023). La mantisse contient 52 chiffres après la virgule (1,.....).

---

Trouver le nombre à virgule représenté par le mot  
1100010001101001001110000110000000000000000000000  
0000000000000000

*Le signe est représenté par 1.*

*L'exposant est représenté par 10001000110.*

*La mantisse est représentée par*

*10010011100001100000000000  
000000000000000000000000.*

*Le signe du nombre est donc -.*

*Le nombre 100 0100 0110 est égal à 1 094 et  
l'exposant du nombre est n = 1094 - 1023 = 71.*

---

$$\begin{aligned}m &= 1.1001\ 0011\ 1100\ 0011\ 1000\ 0000\ 0000\ 0000\ 0000\ 0000\\&= 1 + 1/2 + 1/2^4 + 1/2^7 + 1/2^8 + 1/2^9 + 1/2^{10} + 1/2^{15} + 1/2^{16} + \\&\quad 1/2^{17}\\&= (2^{17} + 2^{16} + 2^{13} + 2^{10} + 2^9 + 2^8 + 2^7 + 2^2 + 2 + 1) / 2^{17}\\&= 206727/131072.\end{aligned}$$

*Le nombre représenté est donc  $-206\ 727 / 131\ 072 \times 2^{71}$   
=  $-3.724\dots \times 10^{21}$ .*

- 
- il existe maintenant des ANSI/IEEE standards pour ces représentations et opérations des virgules flottantes. Tous les processeurs les traitent de la même façon.
  - les additions et soustraction sont plus compliquées que les multiplications car il faut aligner les nombres (SHIFT) puis renormaliser le résultat. Pour la multiplication, il suffit de faire un «shift» final.
  - il y a un grand nombre d'opérations arithmétiques qui portent et sur l'exposant et sur la mantisse.

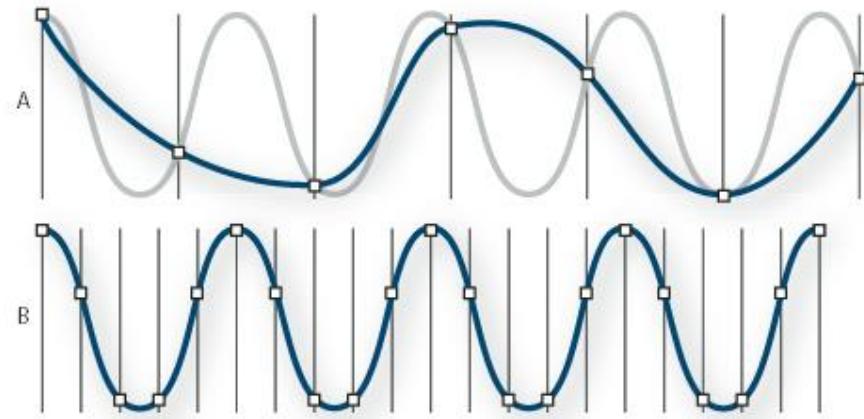
# Images

---

- Version BitMap
  - photos, peintures, vidéo, scanner
  - chaque pixel est encodé: niveau de gris ou couleur
  - le format (.gif): décomposition en blocs,taille,palette de couleur, résolution.
  - Le format (.jpeg): des blocs de 64 pixels codés fréquentiellement
- Version PostScript
  - l’information géométrique (segments, courbes définis mathématiquement) et les positions relatives --> information sur l’image ---> codage économique.
  - Facile à redimensionner
  - l’imprimante ou l’écran peuvent convertir le ps en bitmap.

# Codage binaire: sons

- Pour numériser du son enregistré par micro ou créé par un instrument acoustique ou électrique
  - On découpe le son en échantillons de temps et on code chaque échantillon sur un certain nombre de bits → Analog to Digital Converter (ADC)
  - Fréquence d'échantillonnage: nombre d'unités de sons (échantillons) codées par seconde
    - » Ex. CD: 44,1 kHz (44100 échantillons / seconde)
    - » Rem: Seuil audible pour l'oreille humaine: +/- 20 kHz, mais il faut échantillonner 2 fois plus vite que la fréquence du son pour la capturer
  - Nombre de bits par échantillon
    - » Ex. CD: 16 bits
  - Nombre de canaux:
    - » mono (1) v. stéréo (2)
  - Pour rejouer le son, il faut un Digital to Analog Converter (DAC)
  - → 1 minute d'enregistrement en stéréo et qualité CD:
    - =  $2 * 16 * 44100 * 60 = 84,672,000$  bits
    - = 10,584,000 bytes  $\approx 10\text{Mb} / \text{min.}$



# Regroupement et compression des données

---

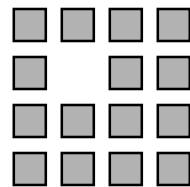
Nombre de bits	Appellation anglaise	Appellation française
1	Bit	Bit
4	Half-byte	Demi-octet
8	Byte	Octet
16	Word	Mot
32	Double Word	Mot double
64	Quad Word	Mot quadruple

---

Unité	Puissance 10	Valeur	Puissance 2	Valeur
Kilo	$10^3$	1.000	$2^{10}$	1024
Méga	$10^6$	1.000.000	$2^{20}$	1.048.576
Giga	$10^9$	1.000.000.000	$2^{30}$	1.073.741.824
Téra	$10^{12}$	1.000.000.000.000	$2^{40}$	1.099.511.627.776

# Comprimer l'information (zip,Jpeg, MP3,.....)

- Les mécanismes de compression perdent ou non de l'information:
  - quand ils n'en perdent pas, ils exploitent les redondances: une lettre ou un mot qui revient souvent dans un texte sera codé sur moins de bits. Dans une image, si plusieurs pixels sont égaux on indiquera seulement leur nombre. Ainsi le groupe “ABBA” pourrait s'écrire sur 4 bits: 0110 plutôt que 32. Et le dessin :



pourrait se coder: 5 1 10

- quand ils en perdent, on parle de compaction, ils ne dégradent que quelque peu l'information: MP3, MPEG4,... On supprime les fréquences inaudibles, on diminue la résolution de l'image, ....
- La compression est symétrique, pas la compaction

# Corriger l'information

---

- Corriger les erreurs de transfert exige des bits additionnels.  
Par exemple 1001 deviendrait 111000000111.  
Chaque bit est répété trois fois
- Possibilité plus économique de détecter les erreurs de transfert et de les corriger par l'addition de bits de parité. Le plus simple est 1 bit (pair/impair) mais on peut recourir à plusieurs bits qui permettent alors la correction d'erreur. Soit un message de 16 bits.

					Colonne de contrôle
	0	0	1	1	0
	0	1	0	1	0
	1	1	0	1	1
	0	1	1	1	1
Ligne de Contrôle	1	1	0	0	

# Encrypter l'information

---

- Notion de clef privée:
  - $N = 2$
  - « informatique » deviendrait « kphqtocvkswg »
  - Encryptage symétrique → Problème: la clé circule !!!
- Clef publique, clef privée:
  - Encryptage asymétrique
  - Le destinataire reçoit la clef publique et encrypte le message que seul, vous, avec la clef privée pouvez déchiffrer

## II. Fonctionnement intime du processeur

---

- 
- L'ordinateur exécute des programmes
  - Un programme = une séquence d'instructions, généralement écrites dans un langage de programmation, puis automatiquement traduites en instructions élémentaires pour un processeur particulier.
  - Le programme exécute un algorithme contenant des instructions en séquence ainsi que des boucles et des tests conditionnels.
  - Une fois traduites en instructions élémentaires, PHO prend le relais.

# Schéma général de fonctionnement de la machine de Von Neuman:

---

## □ *Le Petit Homme Ordinateur: PHO*

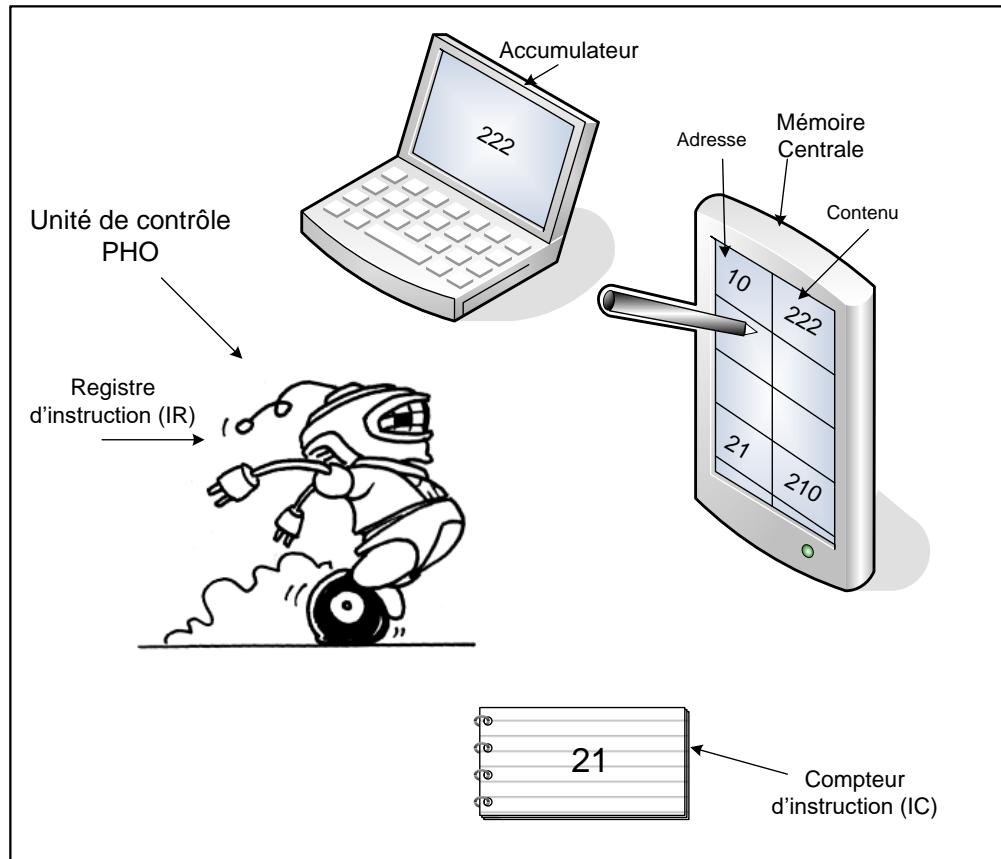
- Il cherche l'adresse de l'instruction (indexée par le compteur)
- Il la lit et la décode
- Il l'exécute (the fetch-execute cycle)
- Il incrémente le compteur
- C'est la conception de Von Neumann, toujours d'actualité  
---> Informatique Séquentielle et Programme chargé et stocké en mémoire. Cela fonctionne de la même manière depuis 50 ans et n'a pas pris une ride.
- Elle implique un CPU, ALU, mémoires et registres.

# Von Neumann (1903-1957)

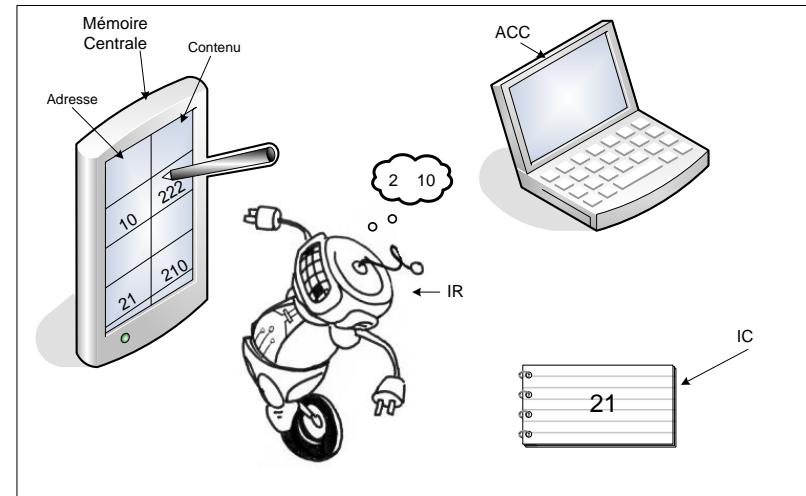
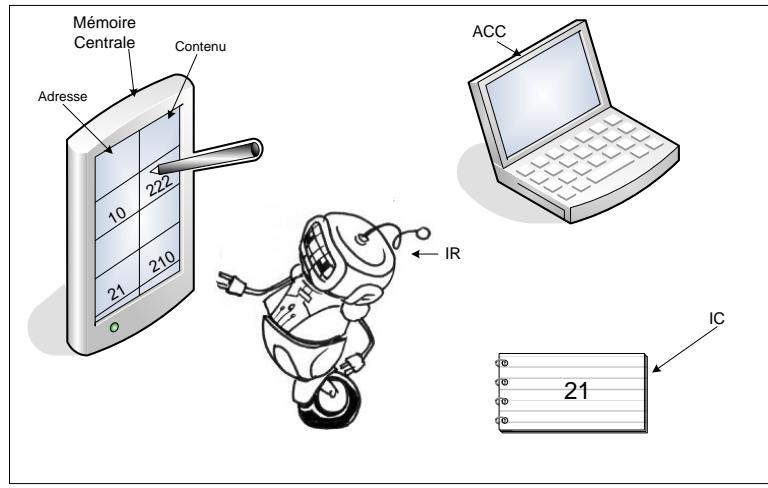
---

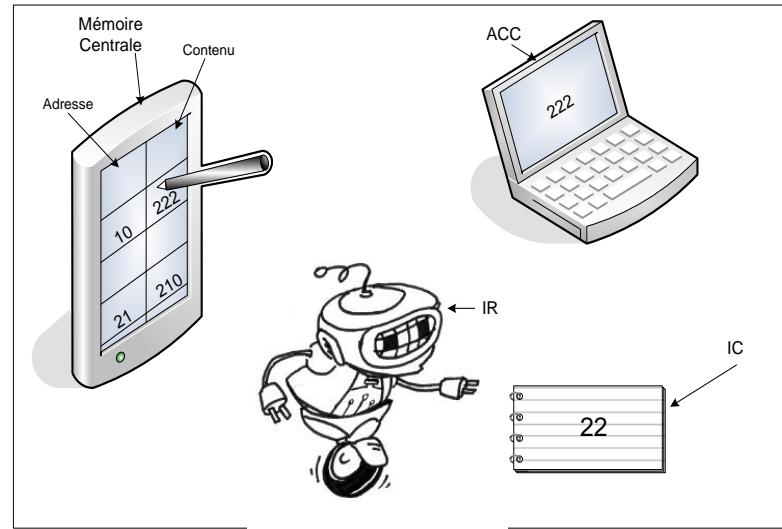
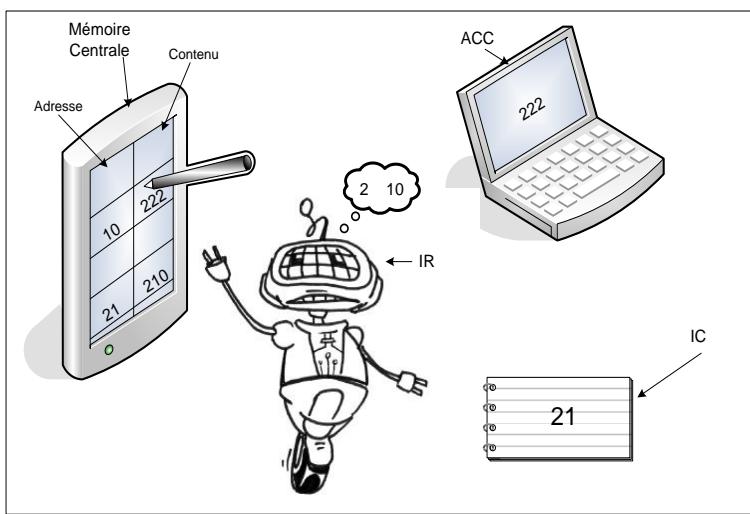


# PHO

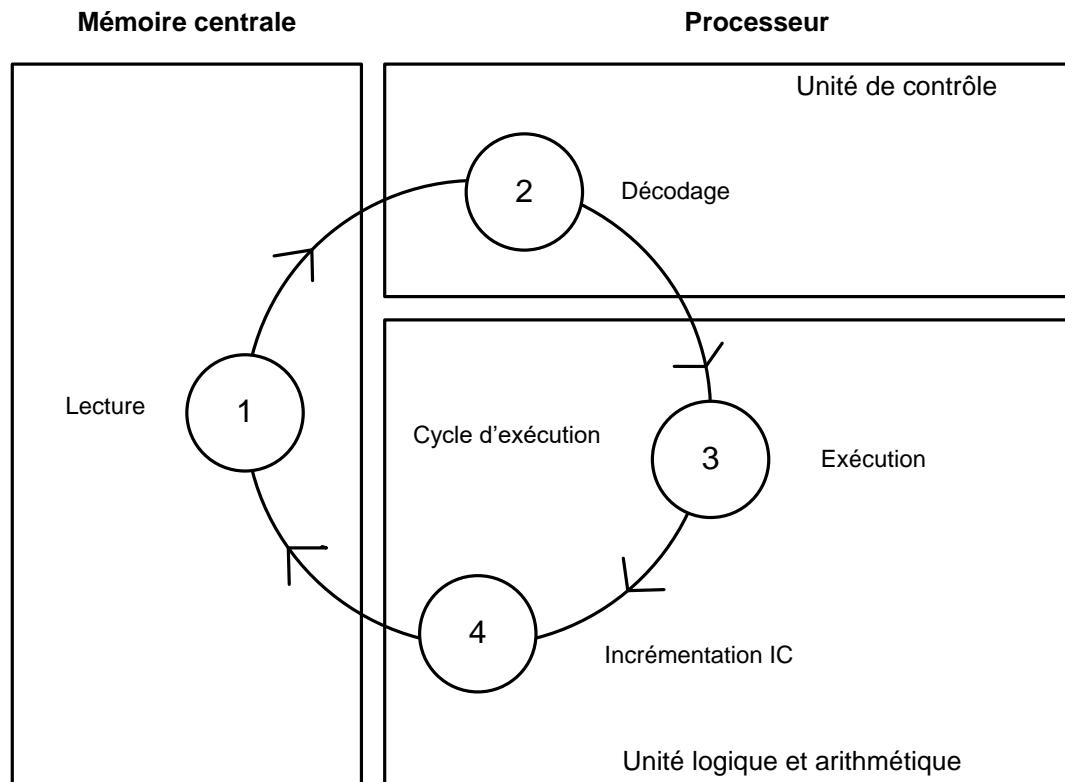


# PHO à l'oeuvre





# Le cycle fondamental



# Les quatre classes d'instructions élémentaires sont:

---

- Déplacement de données:
  - Mémoire  $\leftrightarrow$  Mémoire
  - Registre  $\leftrightarrow$  Mémoire
  - Pérophérique  $\leftrightarrow$  Mémoire
  - Pérophérique  $\leftrightarrow$  Registre
- Opérations Arithmétiques et Logiques:
  - $a + b$  ; z est-il égal à 0 ?
- Opérations Conditionnelles et Branchement:
  - Si  $x < 0$  alors aller a l'instruction 95
- Opérations d'entrée/sortie

# L'écriture des programmes

---

- Au tout début → instructions élémentaires
- Mais depuis, langages programmation de haut niveau (LPH) → Java, C++, Python, Fortran, Cobol ...
- Une instruction en LPH → plusieurs instructions élémentaires.
- “ $c=a+b$ ” est plus simple que “load a, reg1”, “load b, reg2”, “add reg3, reg1, reg2”, “move c, reg3”.

# Jeu d'instructions simplifié

## Jeu d'instructions simplifié

Codes instruction			Description de l'instruction
Symbole Assembleur	Valeur en binaire	Valeur en décimal	
LOAD	010	2	Copie dans l'accumulateur le contenu de la mémoire dont l'adresse est donnée dans l'opérande
STORE	011	3	Copie le contenu de l'accumulateur dans la mémoire à l'adresse donnée dans l'opérande
ADD	100	4	Ajoute à l'accumulateur le contenu de la mémoire dont l'adresse est donnée dans l'opérande
MPY	101	5	Multiplie l'accumulateur avec le contenu de la mémoire dont l'adresse est donnée dans l'opérande
BR	001	1	Interrompt inconditionnellement le déroulement séquentiel du programme par chargement dans le registre d'instruction de l'adresse donnée dans l'opérande
BRG	000	0	A condition que le contenu de l'accumulateur soit plus grand que zéro, interrompt le déroulement séquentiel du programme par chargement dans le registre d'instruction de l'adresse donnée dans l'opérande; dans le cas contraire, passage à l'instruction suivante.

---

## Adresses des données et des instructions

		Adresse
Symbol	Valeur en décimal	Valeur en binaire 16 8 4 2 1
<b>Données</b>		
A	10	0 1 0 1 0
B	11	0 1 0 1 1
C	12	0 1 1 0 0
D	13	0 1 1 0 1
E	14	0 1 1 1 0
F	15	0 1 1 1 1
Z	20	1 0 1 0 0
<b>Instructions</b>		
LB (LeBoulot)	04	0 0 1 0 0
SEQ2	27	1 1 0 1 1

# Du LPH au binaire

---

## Représentation détaillée d'instructions

Classes d'instructions	Fonctions en langage de haut niveau	Langage assembleur Symboles des Code-instruction + opérande		Langage machine Binaire Code-instruction + opérande	
Copie de données	b = a	LOAD STORE	A B	010 011	01010 01011
Opérations arithmétiques	c=d + e * f	LOAD MPY ADD STORE	F E D C	010 101 100 011	01111 01110 01101 01100
Branchements	goto séquence2  if z > 0 goto leBoulot	BR  LOAD BRG	SEQ2  Z LB	001 010 000	11011 10100 00100

# Les différents types d'instruction

Copie de A vers B: b = a

Adresse de l'instruction	Langage assembleur: Symboles		Langage machine : Binaire	
	Code instruction	Adresse opérande	Code instruction	Adresse opérande
21	LOAD	A	010	01010
22	STORE	B	011	01011

---

## Opération arithmétique: $c = d + e \times f$

Adresse de l'instruction	Langage assembleur: Symboles		Langage machine : Binaire	
	Code instruction	Adresse opérande	Code instruction	Adresse opérande
23	LOAD	F	010	01111
24	MPY	E	101	01110
25	ADD	D	100	01101
26	STORE	C	011	01100

---

## Branchement et boucle: while z > 0 do LeBoulot

Adresse de l'instruction	Langage assembleur : Symboles		Langage machine : Binaire	
	Code instruction	Adresse opérande	Code instruction	Adresse opérande
04 LB(LeBoulot)	**	**	**	**
05	**	**	**	**
06	**	**	**	**
07	**	**	**	**
08	BR	SEQ2	001	11011
.....				
27 (SEQ2)	LOAD	Z	010	10100
28	BRG	LB	000	00100
29	**	**	**	**

# Toute l'informatique repose sur le principe de l'abstraction fonctionnelle

---

- Un ordinateur fonctionne à différent niveaux d'abstraction. On peut travailler à un niveau supérieur sans se soucier du niveau inférieur.



*Java/Python*



*assembleur*

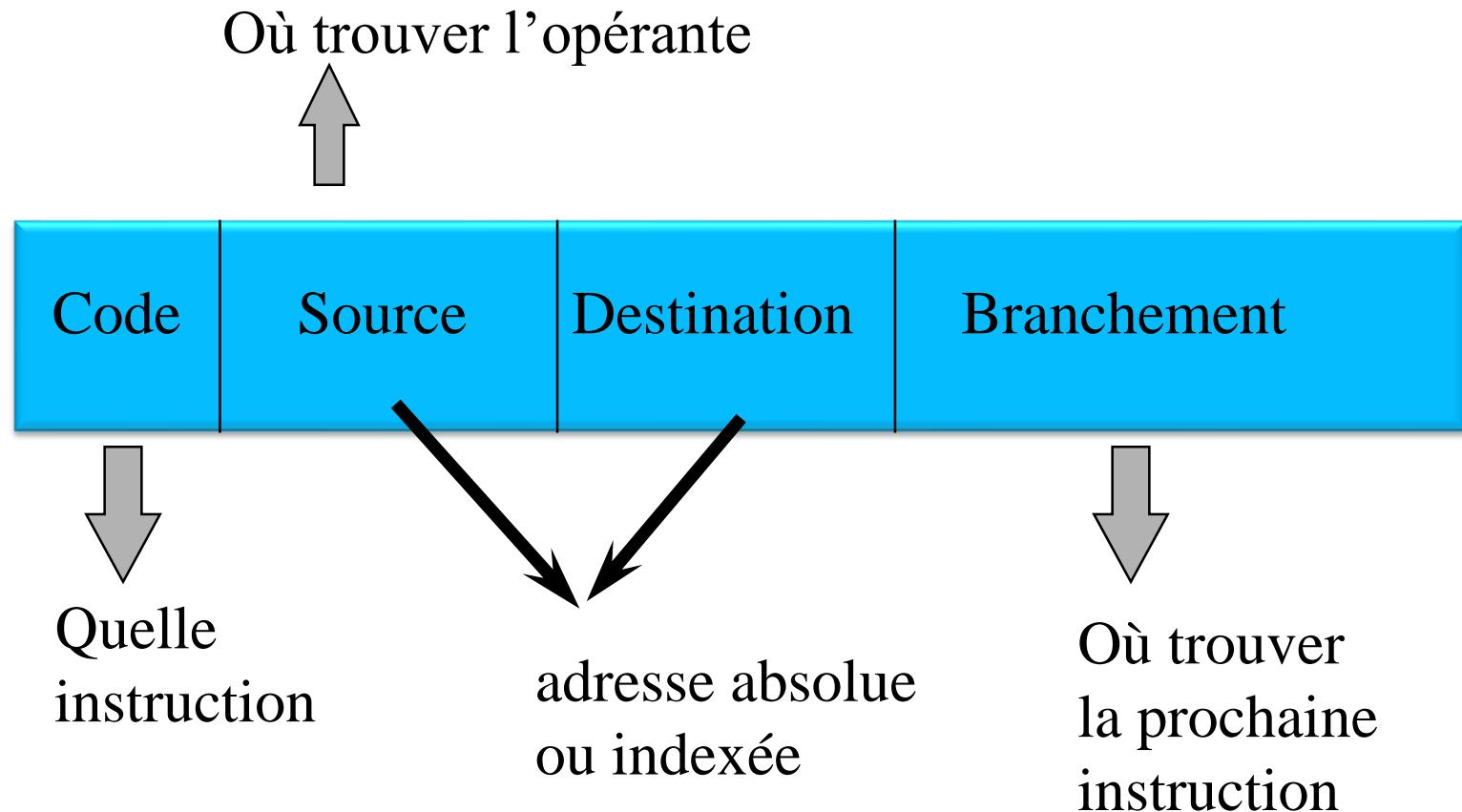


*logique*



*electronique*

# Une Instruction Élémentaire



# Eléments fondamentaux de l'architecture

---

- les registres
  - MAR,MDR,IC,IR,
- les mémoires
  - RAM,DRAM,cache,virtuelle....
- les bus
  - IC--->MAR
  - MDR-->IR
  - les bus pour I/O et les périphériques

# Les Registres

---

- ils contiennent les données actuellement manipulées
- l'instruction exécutée: IR
- les I/O adresses à accéder
- le compteur de programme: IC
- l'adresse mémoire: MAR, la donnée mémoire: MDR
- le registre d'état du processeur informe sur la dernière opération et indique si elle a produit un zéro, ou une retenue, ou le bit significatif du résultat,...
- ils interviennent dans les instructions pour les données ou les adresses des opérantes
- ils peuvent être chargés,sommés,permutés,translatés
- le transfert entre registre conditionne la vitesse du CPU

# Les mémoires

---

- avec un MAR de 32 bits, on peut aller jusque  $2^{32}$  (4 GigaBytes de mémoires principales).
- le MDR = 1 Byte et il faut alors accéder plusieurs bytes successivement, ou 2 ou 4 bytes.
- la mémoire principale est ROM (fusibles), RAM (flip-flop) ou DRAM (condensateurs). La RAM est plus chère et plus rapide que la DRAM ---> mémoire cache. Mais nécessite plus d'énergie, plus de transistors (plus d'espace). Il s'agit de mémoire volatile avec Random Access ( RAM = Random Access Memory).

# Les bus

---

- ils peuvent être locaux et connecter des registres entre eux. Plus il y en a, plus d'information pourront être transmises simultanément, plus le CPU ira vite.
- ils peuvent également connecter le CPU à la mémoire et le CPU aux périphériques
- En général les bus sont parallèles au sein du CPU et séries pour connecter des périphériques plus distants
- un bus contient un ensemble de lignes avec données, adresses ou information de contrôle (par ex. le timing ou des informations read/write).

# Les instructions élémentaires

---

- 4 grandes classes
  - 1. transfert ou mouvement de données: registres -> mémoire --> registres
  - 2. arithmétique et logique + translation ou permutation
  - 3. instruction de branchements
  - 4. entrée/sortie → Pérophérique
- l'instruction comprend plusieurs champs: code et opérantes (1,2 ou 3) à tenir sur 2,4 ou 6 bytes d'où naît un problème d'adressage.
- dans un processeur RISC, toutes les instructions sont de taille égale, souvent 32 bits. Pas du tout le cas pour les CISC jusqu'à 11 bytes (vax ou pentium).

# RISC vs CISC

## RISC

Code instruction	Rs	Rd	Valeur immédiate	
31	25	20	15	0
Code instruction	Rb		Adresse de branchement	
31	25	20	15	0

## CISC

Code instruction	Rs	Code instruction	Rd	
15				0
Code instruction		Valeur immédiate (format court)		
31		15		0
Code instruction		Valeur immédiate (format long)		
47		31		0
Code instruction		Adresse mémoire source		
79		63		32
Adresse mémoire destination				0
		31		

---

## Comparaison entre un processeur Cisc et Risc

Cisc	Risc
Motorola MC68000, Intel 486, Pentium (entre les deux)	PowerPC, Alpha(ex-Digital), Sparc (Sun) Représente la tendance actuelle
Une large variété d'instructions, de tailles d'instruction (jusqu'à 30 octets) et de modes d'adressage	Instructions toutes de même taille (32 ou 64 bits) Adressage simplifié, par l'utilisation massive de registres
Plus cher en mémoire	Moins cher en mémoire
Plus compliqué, plus flexible mais plus lent	Plus simple, plus rapide
Programme plus court contenant moins d'instructions, mais exécution d'instruction plus lente, une instruction pouvant avoir besoin de 20 étapes	Utilisation massive des registres (nécessite beaucoup de registres) et des transferts entre registres. Les programmes s'exécutent plus localement par les registres et la « cache »
Difficile pour le pipeline mais recours intensif à la microprogrammation	Plus adéquat pour le pipeline → une instruction par cycle d'horloge
Beaucoup d'accès mémoire dans la majorité des instructions	Accès mémoires plus rares: uniquement par les instructions « load » et « store »

# RISC vs. CISC

---

## □ Intel Pentium:

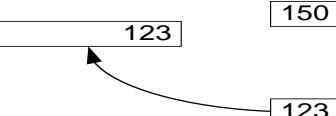
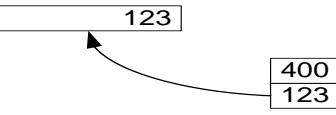
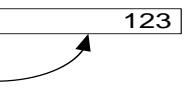
- Architecture générique baptisée Intel Architecture 32 bits (IA32)
  - » Physiquement, ce sont des processeurs de type RISC
  - » Mais ils doivent être compatibles avec les logiciels écrits pour les anciennes générations de processeurs Intel (x86)
  - » Dès lors le processeur (de type RISC) avec des instructions codées sur 32 bits simule un processeur CISC (i.e. traduit les instructions x86 en instructions IA32)
- Dernière génération (Core i7) est une architecture à 64 bits qui contient plusieurs jeux d'instructions (x86, MMX, SSE, SSE2, SSE3, SSSE3, x86-64, SSE4)



# L'adressage direct ou indirect

---

- Pour réduire l'espace utilisé à l'adressage, on peut utiliser, via des registres, des adressages indirects ou indexés.
- Cela permet des adresses plus courtes et est tout à fait en phase avec la logique de programmation qui fonctionne modulairement, avec des variables locales, boucle, matrice ou pointeur.
- On peut adresser des registres, ou à partir « d 'offset »,...
- les instructions élémentaires font vraiment la différence entre les types de processeur.
- Différences fondamentales entre CISC et RISC
- Le RISC doit adresser beaucoup plus à partir des registres pour maintenir des adressages courts et donc des petites instructions

Type d'adressage	Instruction	Résultat dans Accumulateur	Contenu Registre ou Mémoire	Numéro Registre ou Adresse Mémoire	Adresse de l'opérande
Absolu:	Load 400		123	R0 R1	200 400
Registre	Load R1		123	R0 R1	R1
Relatif	Load 250, R0		150	R0 R1	200 400
Indirect	Load ind [200]		123	200 400	250+150 =400
Immédiat:	Load "123"		123	R0 R1	"123"

# Fonctionnement du processeur

---

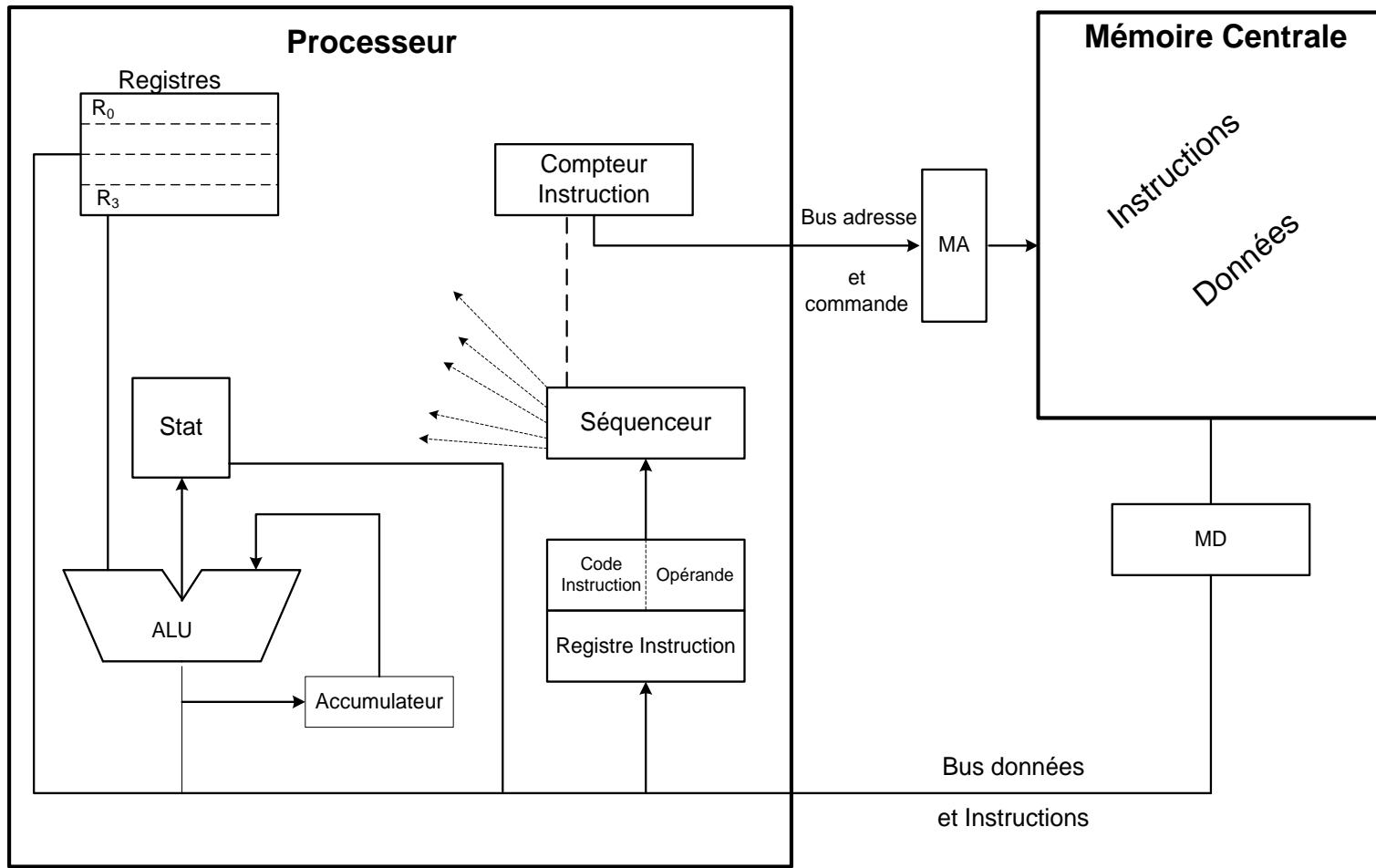
## □ Les registres:

- A chaque registre correspond une utilisation spécifique.
- Tout le fonctionnement de l'ordinateur se ramène à des transferts de registres
- C'est cette vitesse de transfert que l'on qualifie avec “les fameux GHz” indiquant la vitesse du processeur.
- Un transfert de registres est une “étape atomique”
- Le déroulement d'une instruction élémentaire comprend plusieurs étapes atomiques.
- Combien d'étapes atomiques peut-on exécuter en 1 seconde →  $\text{GHz} = 1000000000$

---

Registre	Fonction
IC	Instruction Counter : contient l'adresse de la prochaine instruction à exécuter
IR	Instruction Register : contient l'instruction à exécuter (code-instruction et opérande)
ACC	Accumulator : Registre spécialisé pour le calcul sur des données alimentées depuis les registres ou la mémoire
R <sub>n</sub>	General Purpose Register : 4 registres de travail banalisés de 1 octet, source ou destination de données en mémoire ou dans l'accumulateur
STAT	Status : Reflète (entre autres) l'égalité à zéro du nombre contenu dans ACC, et est examiné par les instructions de branchement conditionnel
MA	Memory Address : reçoit l'adresse de l'emplacement en mémoire où un contenu sera lu ou écrit ; MA est complété par un signal, pour commander la lecture ou l'écriture en mémoire, et par un signal (TERM) indiquant que la mémoire a terminé la lecture ou l'écriture
MD	Memory Data reçoit l'octet lu ou à écrire en mémoire à l'adresse indiquée dans MA.

# Le schéma fondamental



# Déroulement des instructions élémentaires

---

- Soit:

LOAD	R1
ADD	R2

- Première phase: le chargement:

Cycle de base	1	2
Etape	MA $\leftarrow$ IC	IR $\leftarrow$ MD

- Seconde phase:  
l'exécution:

Le chronogramme logique complet de l'instruction LOAD R1 devient donc le suivant:

Cycle de base	1	2	3
Phase	Chargement		Exécution
Etape	MA $\leftarrow$ IC	IR $\leftarrow$ MD	ACC $\leftarrow$ R <sub>1</sub>

Celui de l'instruction ADD R 2 est semblable, à la phase d'exécution près :

Cycle de base	1	2	3
Phase	Chargement		Exécution
Etape	MA $\leftarrow$ IC	IR $\leftarrow$ MD	ACC + R <sub>2</sub>

# Parallélisme entre étapes atomiques

---

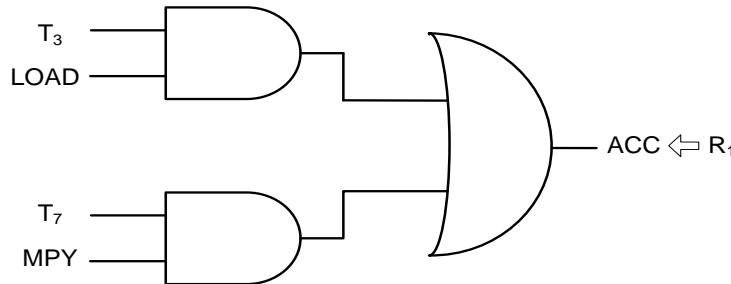
Cycle de base	1	2	3	4
Etape	$MA \leftarrow IC$	$IR \leftarrow MD$	$ACC \leftarrow R_1$	...
	...	$MA \leftarrow IC$	$IR \leftarrow MD$	$ACC + R_2$

Mais il est important de maintenir une indépendance logique entre les instructions se suivant dans la séquence.

→ Pipeline

# Séquenceur: la succession des étapes atomiques

- Séquenceur câblé:

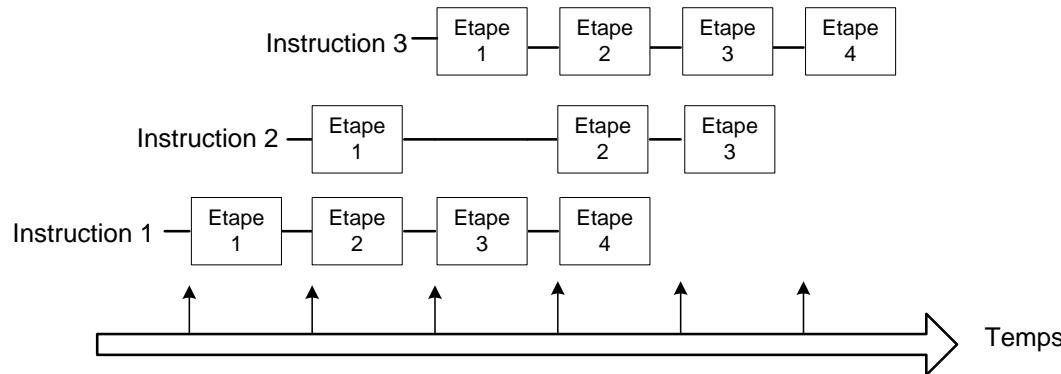


- Séquenceur microprogrammé :

Instruction	Etape	Branchement si Count NOT 0	Adresse branchement	R <sub>1out</sub>	ACC <sub>in</sub>	MD <sub>in</sub>
LOAD	...					
	T3			1	1	
	...					
ADD	...					
	T7			1	1	
	...					

- 
- Les instructions sont cadencées (ou séquencée) par une horloge électronique
  - le temps d'exécution d'un programme sera dépendant du nbre d'instructions, nbre d'étapes par instruction et la durée d'une étape
  - la durée minimale d'une étape est la durée requise pour un transfert entre deux registres → GHz

# Le pipeline: premier parallélisme



- **Limites du pipeline**
  - OK:  $c = a + b$  puis  $f = d + e$
  - Pas OK:  $c = a * b$  puis  $f = c + d$
  - problèmes: 1. dépendances entre instructions (le «load» de la deuxième est le «store» de la première) - 2. instruction de branchement (la «suivante» n'est pas la «suivante»).
  - résolu: soit à la compilation, soit en créant des retards, soit en utilisant plusieurs lignes de pipeline, soit en ré-ordonnant les instructions.
  - Pentium: 20 étages de pipeline

# Du superscalaire à l'informatique distribuée

---

- Superscalaire
  - exécute plusieurs instructions simultanément
  - possède plusieurs CPU
  - divise les instructions en 3 classes: entier, virgule flottante et branchement, et aiguille chaque type vers un CPU dédié
  - le powerPC 601: 3 CPU, 1 pour les entiers avec 1 pipeline à 4 étapes, 1 pour les virgules flottantes avec 1 pipeline à 5 étapes et 1 pour les branchements.
- Systèmes multi-processeurs, chacun exécutant simultanément la même partie du programme ou des parties très séparées.
- Grid computing

# Améliorations courantes et constantes

---

- accélération du CPU, cependant:
- la vitesse de l'ordinateur ne dépend pas que de la fréquence de l'horloge mais également de la durée des accès RAM ou accès I/O
- bus plus larges: 64 bits
- mémoire RAM étendue ---> mémoire virtuelle
- accès mémoire accéléré: «cache»
- accès disque accéléré
- processeurs en parallèle.
- grid computing: « the computer is the network »
- parallélisation croissante: informatique quantique ou biologique

## **III. Les mémoires centrales**

---

# Caractéristiques des mémoires centrales

---

- capacité: dépend de la taille du MAR. Soit un MAR de 32 bits --->  $2^{32}$  x (la taille du MDR) – aujourd’hui 64 bits.
- le CPU envoie un «REQUEST» et indique la nature de l’interaction «R/W». Une fois les données transmises, la mémoire envoie un «COMPLETE»
- la qualité des mémoires est fonction des temps d'accès aux données («latency») et des durées de transmission de ces données (bandwidth ou «bande passante»).
- cela induit une hiérarchie des mémoires, des plus rapides aux plus lentes: registre --> cache --> principale --> disque --> bande. Les plus rapides étant les plus chères. Les passages d'un niveau à l'autre sont invisibles à l'utilisateur.

- 
- mémoire RAM: accès random, pourquoi ??
  - le décodage d'adresse:  $n \rightarrow 2^n$  exige toute une circuiterie logique lourde et coûteuse.
  - pour simplifier ces circuits, le décodage est souvent fait en série: on sélectionne une fois, puis on re-sélectionne dans la partie juste sélectionnée.
  - les DRAM sont moins chers, moins lourds en électronique, mais plus lent, et exigent l'addition d'opération de rafraîchissement très fréquente. Le «timing» doit donc inclure des périodes de rafraîchissement → mémoire principale et non la cache.
  - la mémoire principale peut être structurée en différents modules de mémoires avec des adressages adaptés.

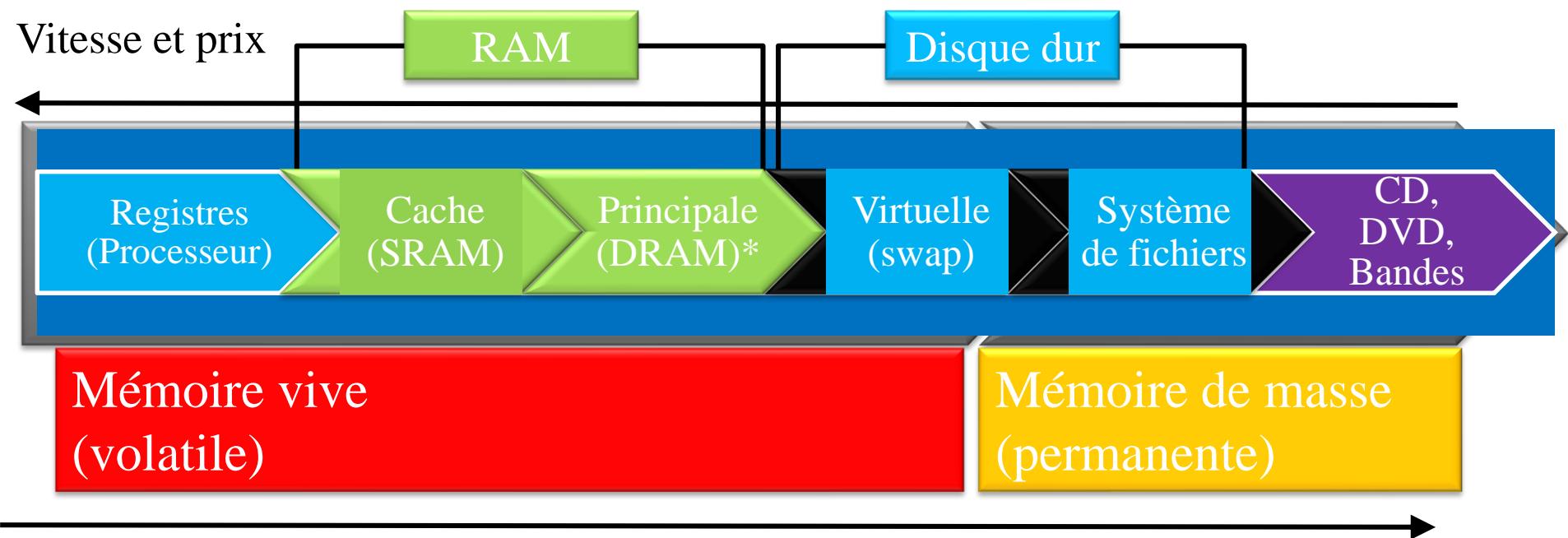
- 
- Des progrès constants dans l'accélération des mémoires qui accompagnent l'accélération des CPU
  - Aujourd'hui aux deux mémoires classiques SRAM (cache) et DRAM se rajoutent des mémoires SDRAM qui fonctionnent à 100MHz, les RDRAM, les VRAM, WRAM, SGRAM, etc....
  - Intel vers les DRDRAM → 1,6 Go/s.
  - Les capacités des mémoires RAM évoluent de manière spectaculaire. Les PC des prochaines années pourraient disposer de mémoire RAM de dizaines de Giga, à l'accès synchronisé sur le fonctionnement du processeur.
  - La plus éloignée, la plus permanente, la plus grosse et la moins chère.

# Mémoires secondaires et d'archivage

---

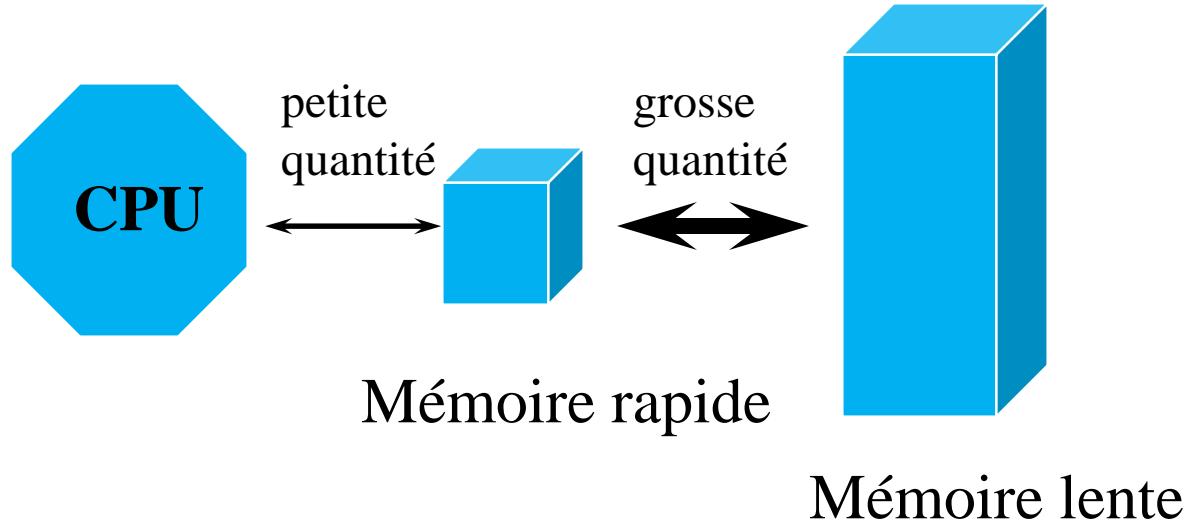
- mémoires secondaires: permanentes et à accès direct: disque dur.
- mémoires d'archivage: permanentes, très éloignées physiquement et à accès plus indirect comme séquentiel: bandes magnétiques.

# Vue d'ensemble de la mémoire



\* DRAM et ses dérivées: SDRAM, DDR  
SDRAM, etc.

# Les niveaux hiérarchiques: mémoire cache et mémoire virtuelle



- Deux niveaux de mémoire, une rapide transférant à haute fréquence peu de données, l'autre plus lente transférant moins fréquemment beaucoup de données.
- Le principe de **localité** rend cela efficace. Un programme s'exécute localement en mémoire (boucle, matrice, routine,...). C'est un principe tant spatial (on reste dans la même zone mémoire) que temporel (on reste avec les mêmes variables)

- 
- Le premier niveau contient des blocs de mots situés consécutivement dans le deuxième niveau (les blocs deviennent des pages dans le cas de la mémoire virtuelle)
  - Des transferts de blocs s'effectuent entre les 2 niveaux.
  - Le CPU s'adresse toujours au premier niveau qui fait appel au deuxième quand il ne peut satisfaire la demande
  - Le deuxième niveau a un temps de latence et un temps de transfert plus long
  - Quand le disque dur est le deuxième niveau (mémoire virtuelle), l'adresse des données est longue: le «disque», la «surface», la «piste», le «bloc».
  - Une adresse se compose maintenant du numéro de bloc et de l'adresse du mot dans le bloc: **bloc** | mot

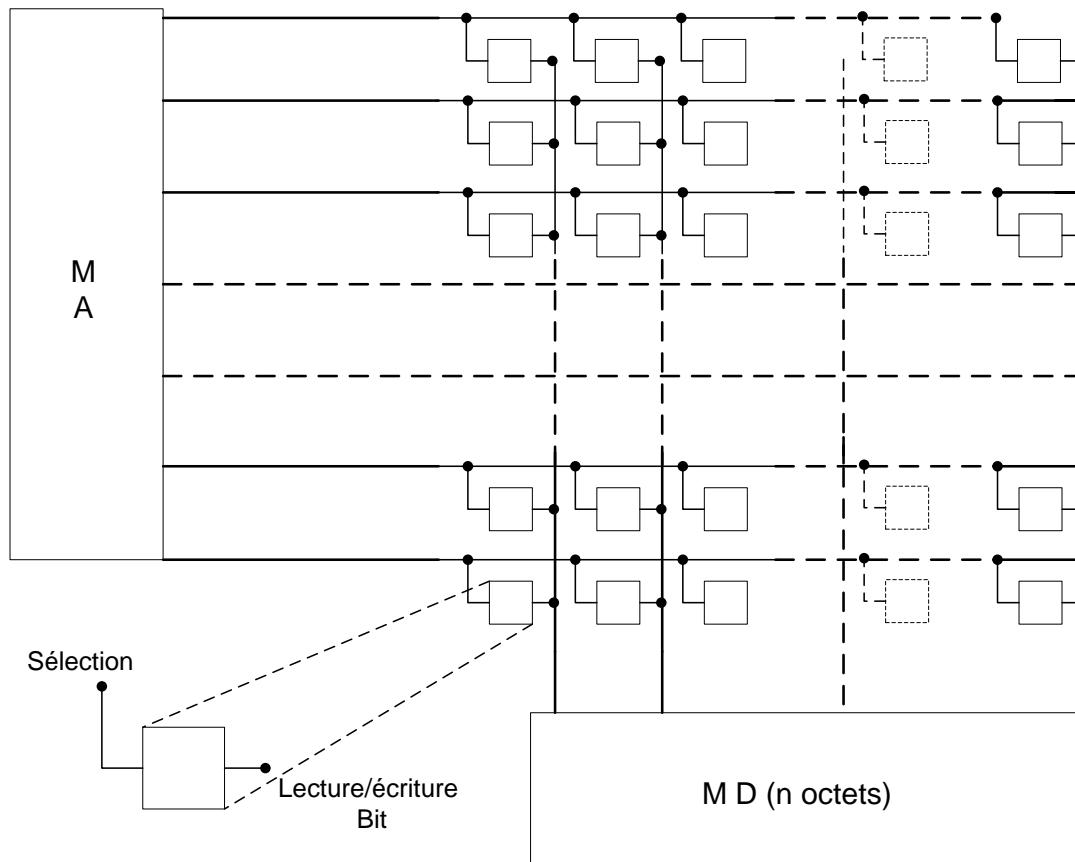
# Décomposition de l'adresse

3	1	8	4	2	1	5	2	1	6	3	1	8	4	2	1	Adressage en continu			
2	6	8	0	0	0	5	2	2	6	4	1	8	4	2	1				
7	3	1	0	4	2	1	5	2	8	4	2	6	4	2	1				
6	8	9	9	8	4	2	1	5	8	4	2	6	4	2	1				
8	4	2	6	0	1	0	0	0	0	0	0	0	0	0	0	Binaire	Adresse 10250		
2				8				0				A				Hexadécimal			
1	6	3	1	8	4	2	1	1	2	6	3	1	8	4	2	1	Adressage en blocs de 256 octets		
2	4	2	6	0	1	0	0	0	0	8	4	2	1	0	0	Bloc 40	Adresse 10		
8	0	1	0	2				8				0				A			
2	0	5	2	1	5	2	6	3	1	8	4	2	1	0	0	Adressage en blocs de 4096 octets			
4	2	1	5	2	6	8	4	2	6	8	4	2	1	0	0	Bloc 2	Adresse 2058		
8	4	2	1	8	4	2	6	8	4	2	6	8	4	2	1	Hexadécimal			

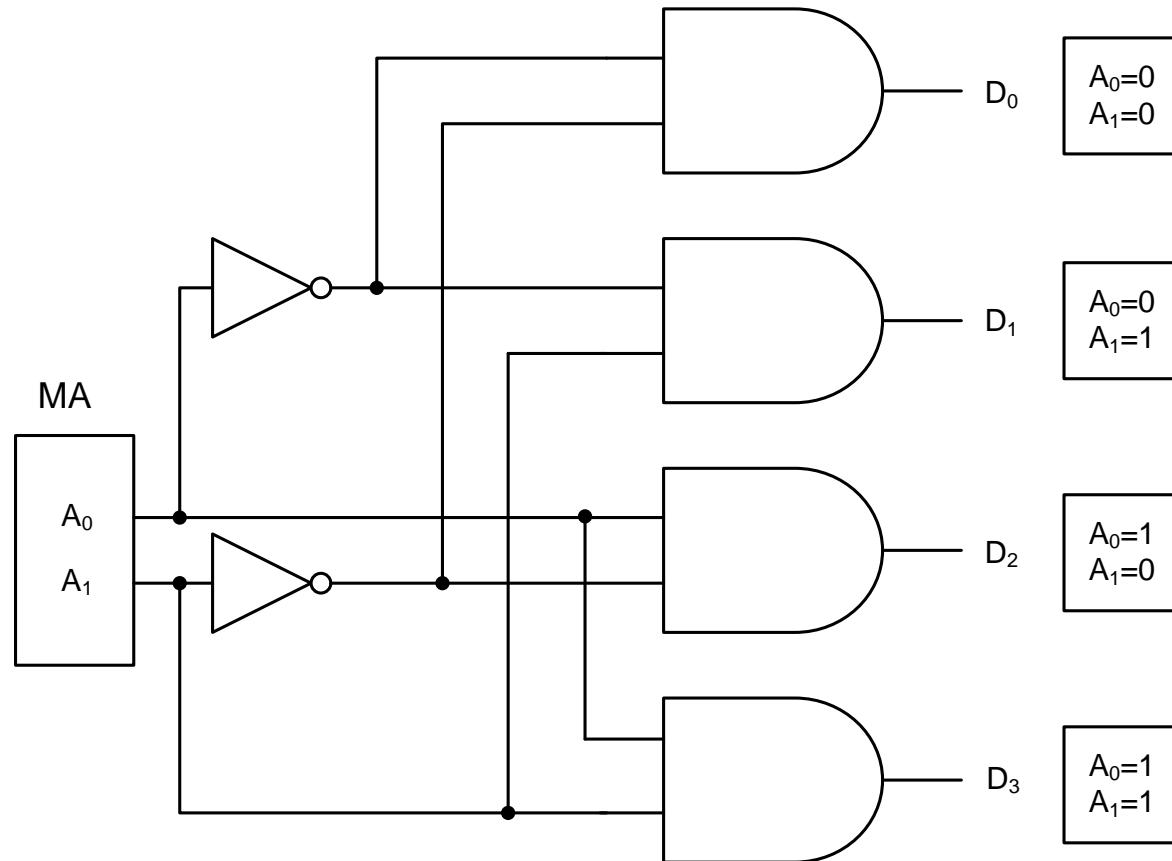
- 
- C'est à partir de la partie "numero du bloc" qu'on peut dire si l'adresse concerne une information contenue dans le premier niveau de mémoire ou dans le deuxième.
  - Les blocs de la mémoire RAM et cache sont les pages de la mémoire virtuelle mais le principe reste le même.
  - Une table intermédiaire vérifie si le bloc ou la page sont dans le premier niveau. Si ce n'est pas le cas, l'adresse est redirigée vers le 2ème niveau. L'information est transférée ainsi que tout un bloc d'informations voisines et la table de redirection est mise à jour.
  - Il faut choisir qui remplacer par les nouveaux venus dans la mémoire rapide, en fonction de la durée de séjour, fréquence ou récence d'utilisation.
  - Ramener les remplacés dans la mémoire lente si ceux-ci ont été modifiés durant leur séjour dans la mémoire rapide.

- 
- Le nombre d'appels ratés au premier niveau doit être petit par rapport aux succès.
  - Pour la cache, le premier niveau fonctionne à la même vitesse que le processeur, le deuxième niveau 10x plus lentement, le disque 1000000x plus lentement.
  - Pour la mémoire virtuelle, quand on fait appel au disque, le CPU peut faire autre chose et s'occuper d'un autre process, pour la «cache» le CPU attend le transfert du bloc.
  - Un bon taux de raté pour la «cache» est 1-2 % et pour la mémoire virtuelle 0.001%

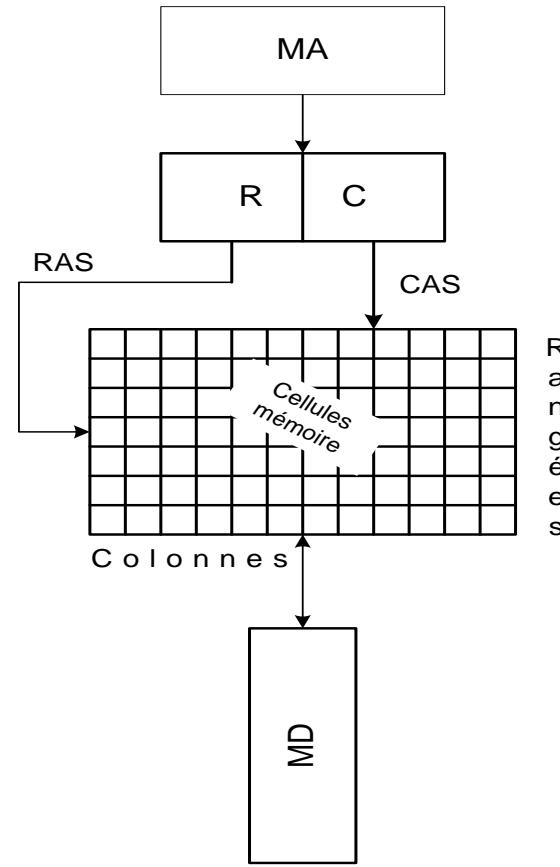
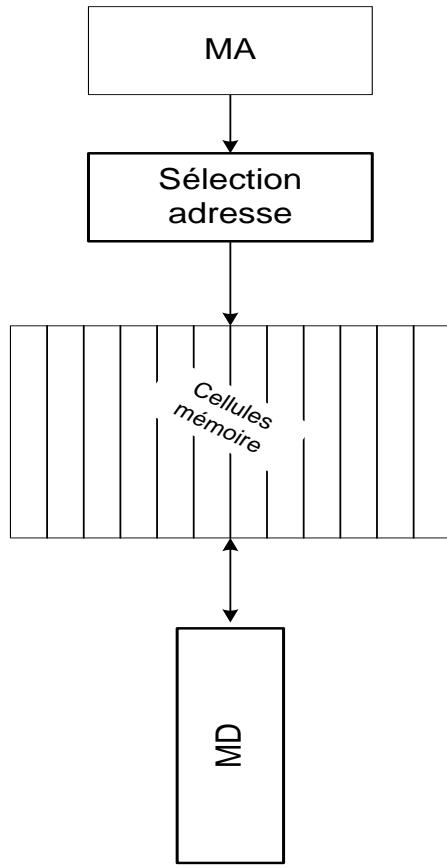
# Fonctionnement de la mémoire centrale ou RAM



# Le décodeur d'adresse: une sacré circuiterie



# que l'on peut simplifier par des mémoires bidimensionnelles



# La mémoire virtuelle

---

- Le principe est d'utiliser le niveau secondaire, dans ce cas-ci le disque dur, pour étendre la capacité de la mémoire principale
- Très utile pour les machines multi-task et pour les serveurs. Chaque tâche peut utiliser la mémoire principale comme si elle était seule. L'illusion est parfaite, d'où l'expression de « mémoire virtuelle ».
- N'oubliez pas que les RAM aujourd'hui classiques sont plusieurs giga et que nombre d'applications (dont l'OS) nécessitent autant sinon plus d'espace mémoire. La mémoire RAM est coûteuse.
- La table intermédiaire est un MMU (Memory Management Unit) entre le CPU et la mémoire principale, pour traduire toute adresse logique en une adresse physique (soit dans la RAM soit sur le disque)
- Une partie du disque dur est alors partitionnée en pages (la partie « swap »).

# Pourquoi la mémoire virtuelle:

---

- Taille de la mémoire insuffisante
- Parties des programmes temporairement superflues
- Multiplication des programmes présents simultanément
- Variations et fractionnement de l'espace mémoire disponible.
- Les avantages sont: 1. adressage simplifié, relatif au programme, 2. espace mémoire étendu sans augmenter en RAM coûteux, 3. contrôle des adressages dans des espaces réservés au système, pour des raisons de protection, on peut facilement séparer les espaces mémoires réservés à différents process.

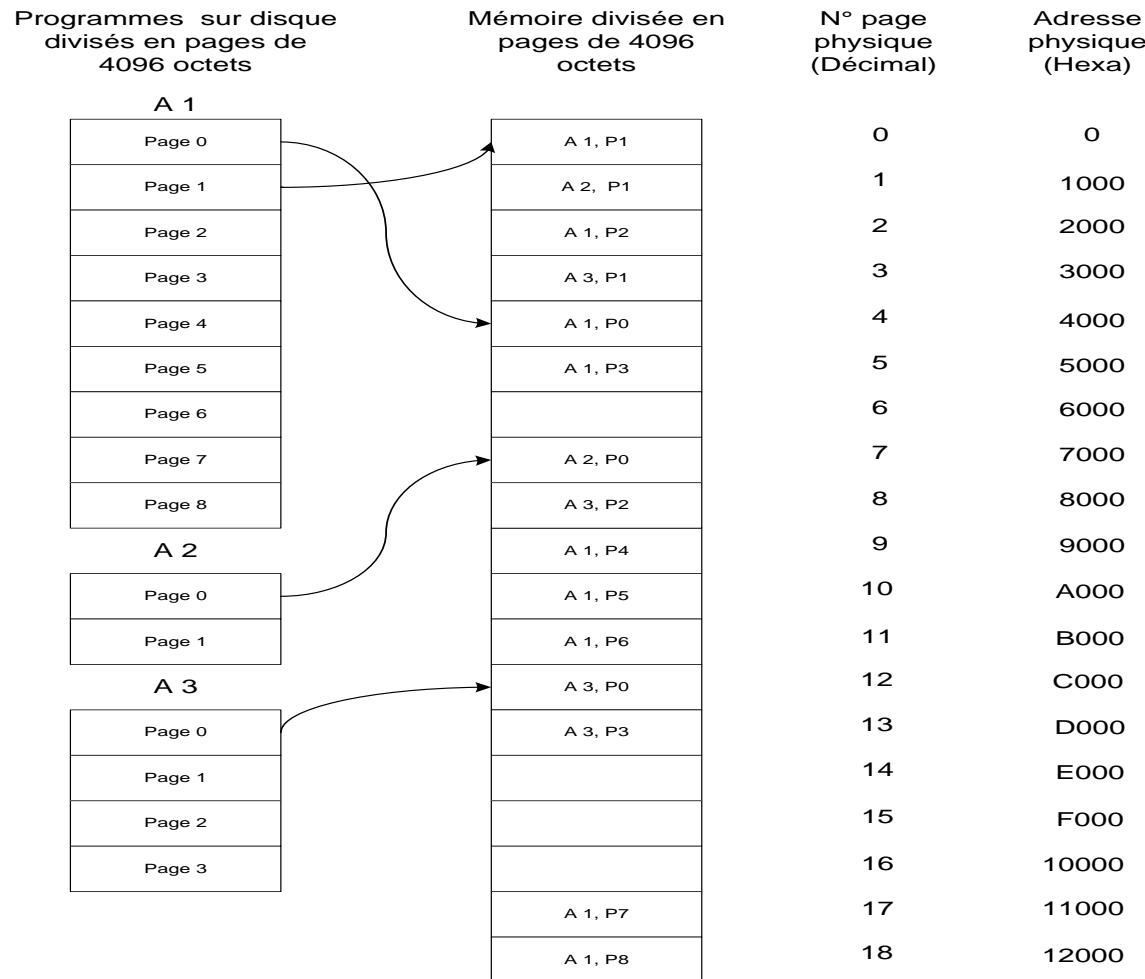
- 
- Une page = 512 bytes à 8Kbytes (dans la figure 4096 bytes)
  - La MMU transforme les adresses logiques en adresse physique, en transformant l'adresse de la page logique en une adresse physique et par concaténation de cette adresse avec l'adresse du mot spécifique. En général, il y aura un MMU par programme.
  - Si l'adresse logique de la page n'est pas dans la table, c'est un raté, le CPU doit s'interrompre (ou changer de tâche) et il faut aller chercher la page sur le disque dur pour la chargé dans la mémoire principale.
  - comme pour le bloc dans la cache, la page a remplacer dans la RAM sera la moins utilisée ou la moins récemment utilisée
  - Normalement, la mémoire virtuelle doit être explorée avant la mémoire cache. Cela peut être long et on utilise alors un TLB (Translation lookaside buffer), une petite cache qui contient les pages les plus récentes. Si la page est trouvée on cherche l'adresse physique en mémoire, sinon on va dans le MMU normal.

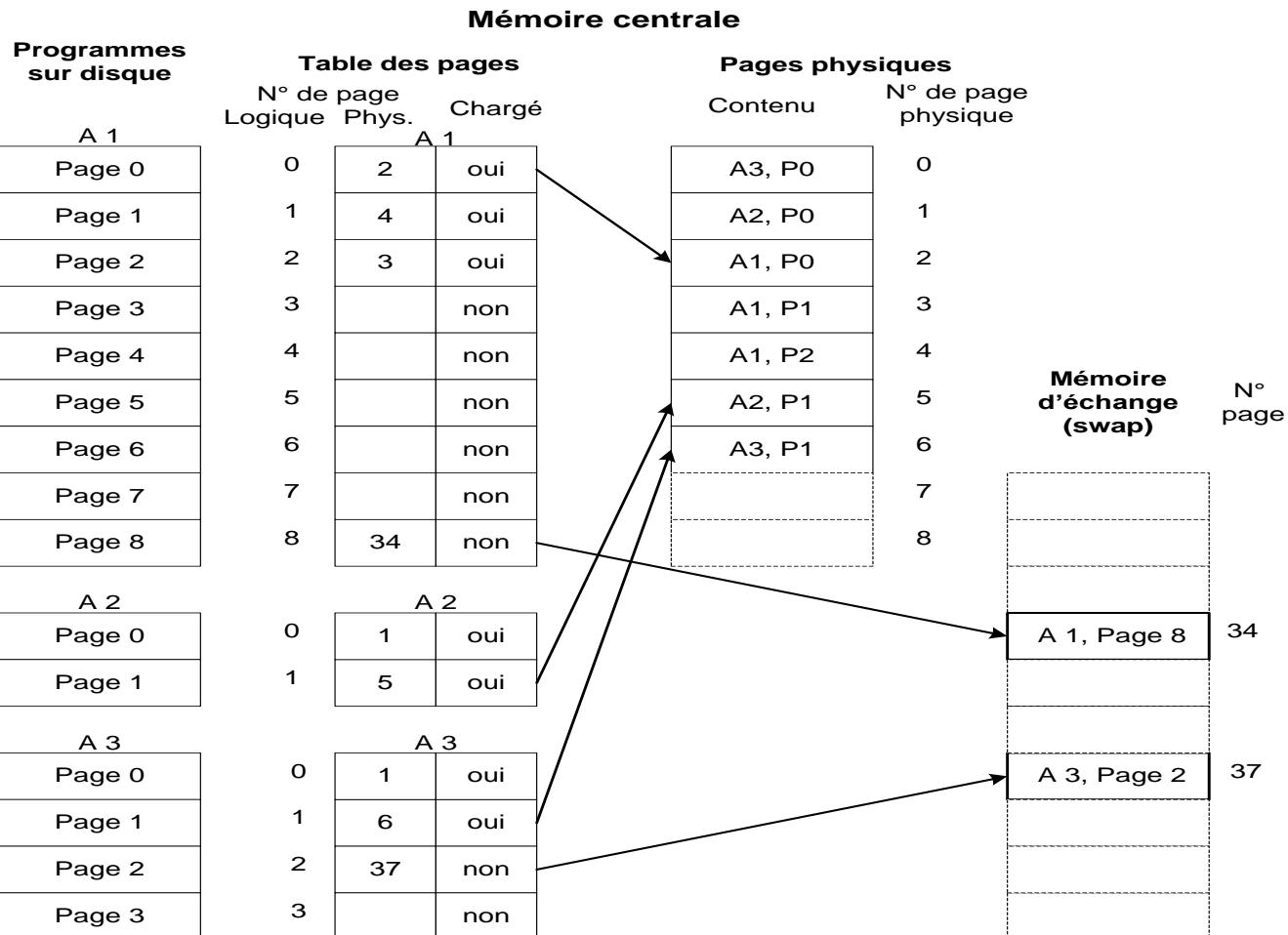
# La pagination

---

Espace adressable virtuel		4094	4095	4096	4097
Espace paginé	Page logique	0	0	1	1
	Octet	4094	4095	0	1

# Fragmentation des programmes en pages





# Le TLB: mémoire associative

- Pour très vite savoir si la page se trouve en RAM ou sur le disque dur
- Mémoire associative car les pages ne sont plus dans une séquence logique et qui doit fonctionner très rapidement.

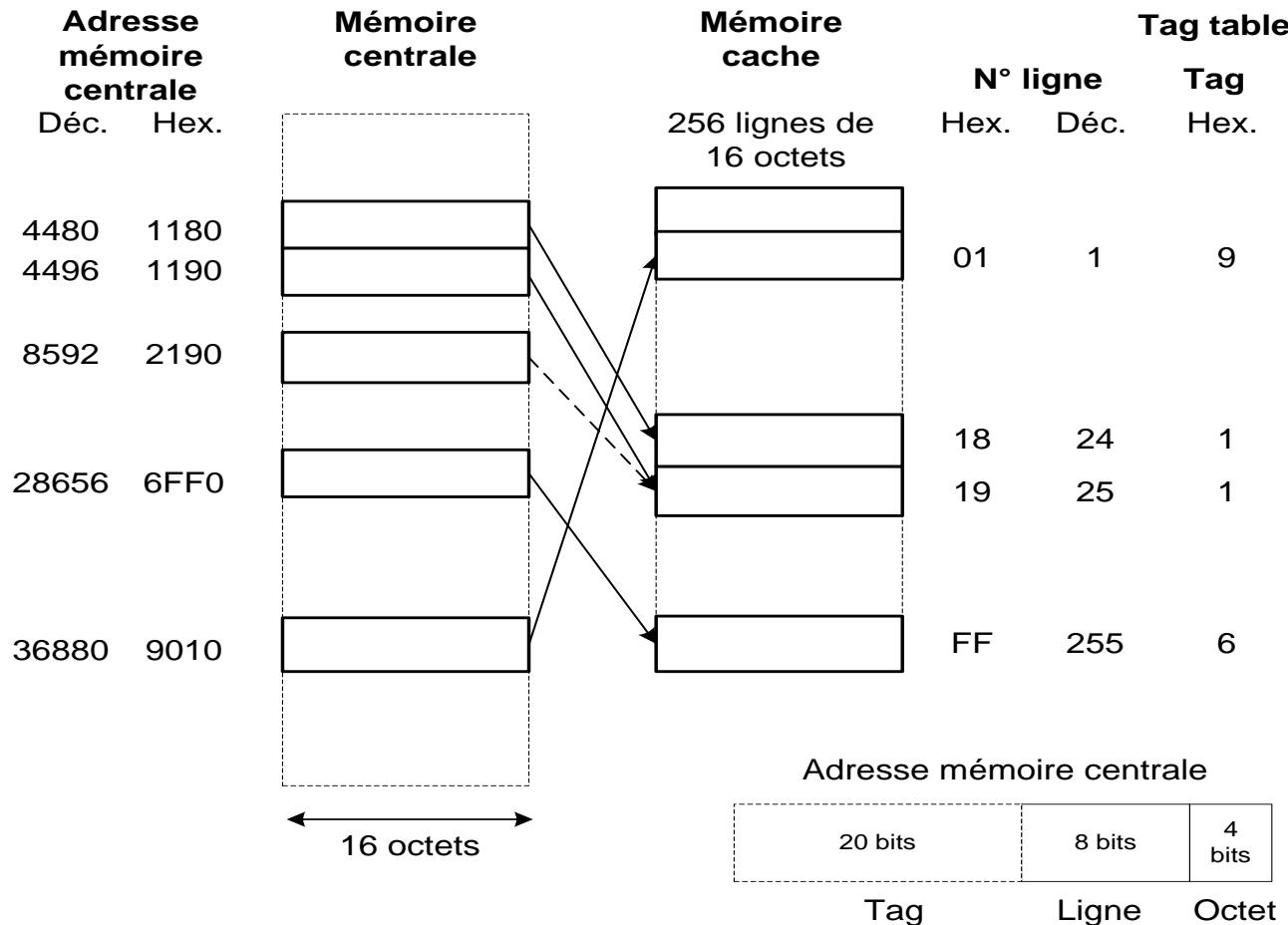
Table des pages en mémoire		Table des pages en TLB	
Adresse dans la table (n° de page logique)	N° de page physique en Hexadécimal	N° de page logique	N° de page physique en Hexadécimal
0	F	Page 4	A
1	C	Page 0	F
2	4	Page 1	C
3	9	.....	.....
4	A	.....	.....
5	.....	.....	.....
6	.....	.....	.....
7	.....	.....	.....
8	.....	.....	.....

# La mémoire cache

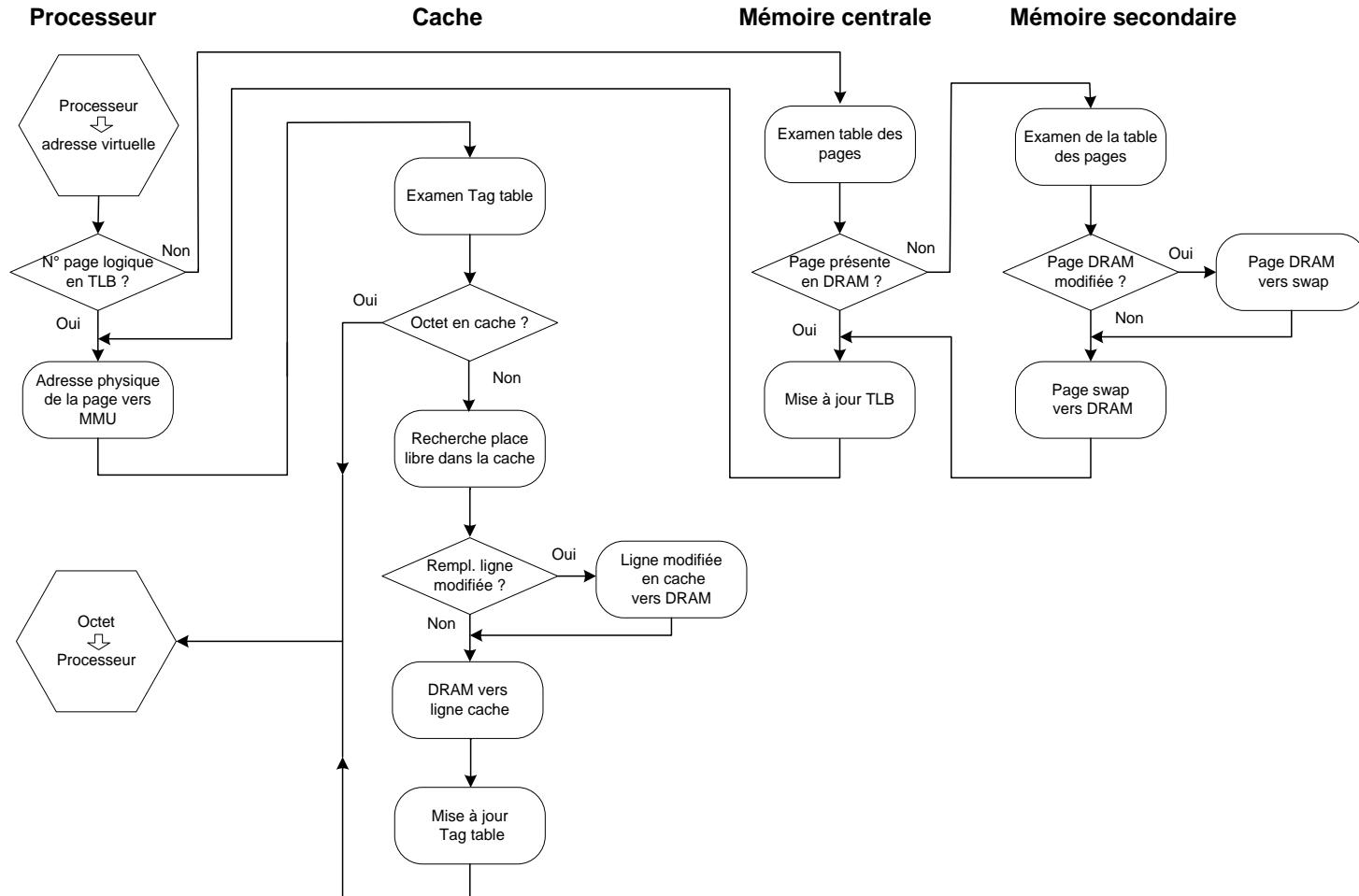
---

- Le programme ne sait pas si la donnée recherchée sera prise dans la cache ou dans le deuxième niveau.
- Les adresses sont découpées en trois parties: tag, bloc, octet.
- Une fonction d'association détermine où se situe le bloc. La méthode la plus simple est la «cache associative» où la «tag table». On utilise un adressage en parallèle pour faire vite (la cache doit être rapide).
- Soit par exemple un bloc de 16 bytes et une cache de 256 blocs ou lignes dans la figure.
- Le deuxième niveau est mis à jour si l'information a été altérée quand on remplace dans le premier niveau un bloc par un autre
- Souvent on remplace le bloc dont l'utilisation remonte le plus loin dans le temps.

# Fonctionnement de la cache



# Accès à la mémoire dans sa globalité

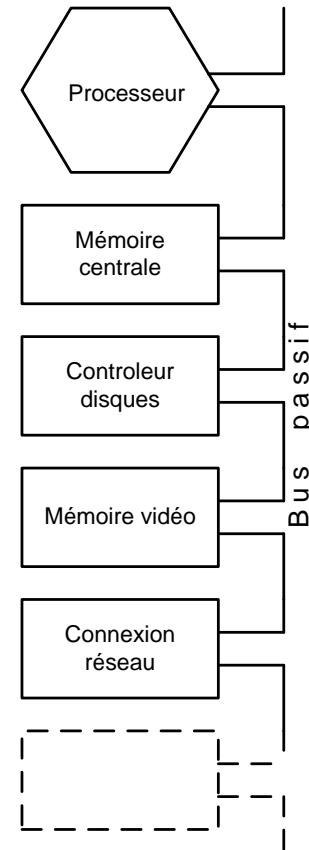


- 
- La gestion complète de la mémoire peut être très complexe
  - D'un accès rapide à la cache à un accès très lent sur le disque dur avec remplacement sur le disque des pages présentes dans la RAM.
  - Dans les Pentium, on sépare la «cache» instruction de la «cache» données car les instructions sont structurées différemment des données en mémoire.
  - Pour optimiser la mémoire, on peut jouer sur beaucoup de paramètres: la taille des blocs, la vitesse de transfert,...
  - On essaie d'accélérer les mémoires et surtout le disque dur qui est très lent.
  - Les accroissements mémoires sont plus importants que les accélérations CPU.

# Interconnexions dans l'unité centrale

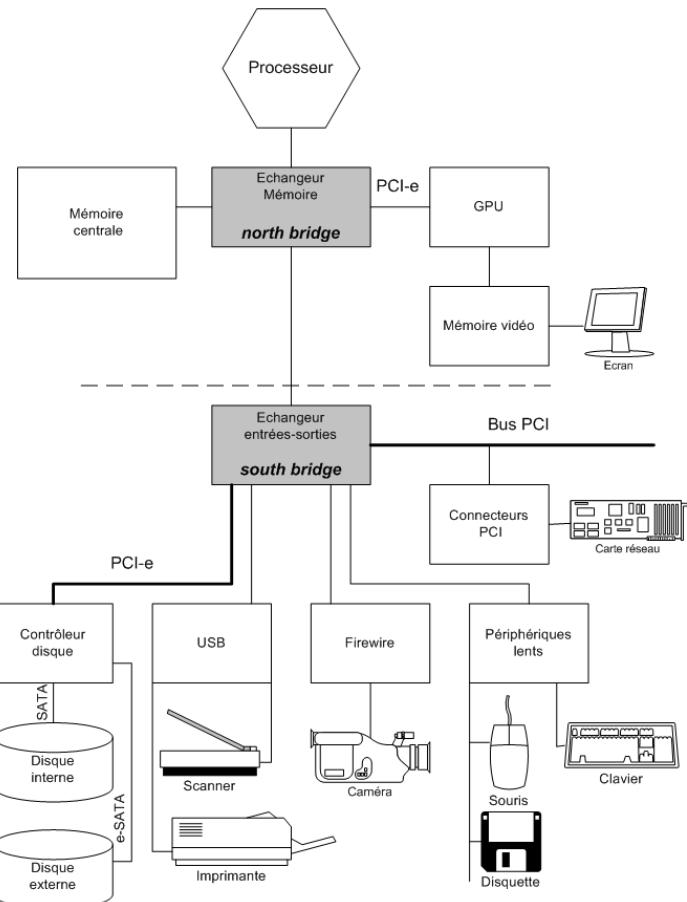
---

- Regrouper les liaisons dans des bus
- Bus multiples ou bus unique.
- Tenter de réduire le nombre de lignes (multiplexage)
- Bus parallèles ou bus séries.
- Bus unique
  - Mais problèmes de vitesse dûs aux composants lents
- → Bus multiples



# Jeu de composants ou “chipset”

- Bus distincts
- Points de jonction
- North et south bridge
- North bridge: bus très rapide
- South bridge: les périphériques.
- La carte mère est le squelette de l'ordinateur: dans laquelle on installe les composants.



# **IV. Entrées/Sorties et Pérophériques**

---

# Périphériques - Généralités

---

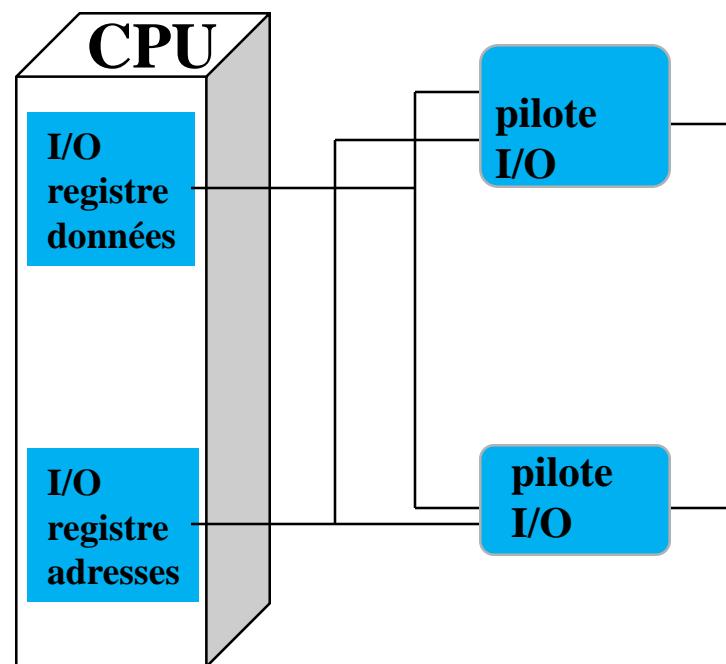
- Sans ses périphériques, les effets du CPU sont inaccessibles. L'ordinateur interagit avec nous via ses périphériques. Il faut organiser les interactions du CPU avec ses périphériques (les I/O). Ces interactions se caractérisent par plusieurs aspects:
  - les interactions peuvent être sous le contrôle du CPU ou asynchrones (se produire indépendamment du déroulement normal du CPU - comme les interruptions)
  - il y a plusieurs périphériques avec lequel le CPU interagit et il faut pouvoir les différentier et organiser éventuellement des communications simultanées.
  - Les périphériques fonctionnent avec des débits de données et des contraintes internes extrêmement différent; ceci exige que chaque périphérique s'accompagne de son contrôleur pour s'interfacer au CPU (qui voit alors tous les périphériques de manière semblable)

- 
- Il est important que le CPU ne différencie pas les périphériques entre eux et que les pilotes de chacun des périphériques se chargent de cette différentiation. Cela simplifie grandement le design du CPU.
  - Les périphériques sont de plus en plus « intelligents » et prennent de plus en plus l'initiative dans leur interaction avec le processeur.
  - Le pilote et le contrôleur se chargent de gérer ces adresses physiques, de structurer les données (par ex. assembler les bits en bytes), de synchroniser la communication, de réaliser les interruptions et de corriger les erreurs de transferts.
  - Les bus I/O et le bus de transferts de données peuvent être séparés ou communs jusqu'à un certain point. Un bus commun de type PCI se retrouve mais ce sont les bus USB qui tentent à standardiser et unifier la connexion aux périphériques.

## Périph.

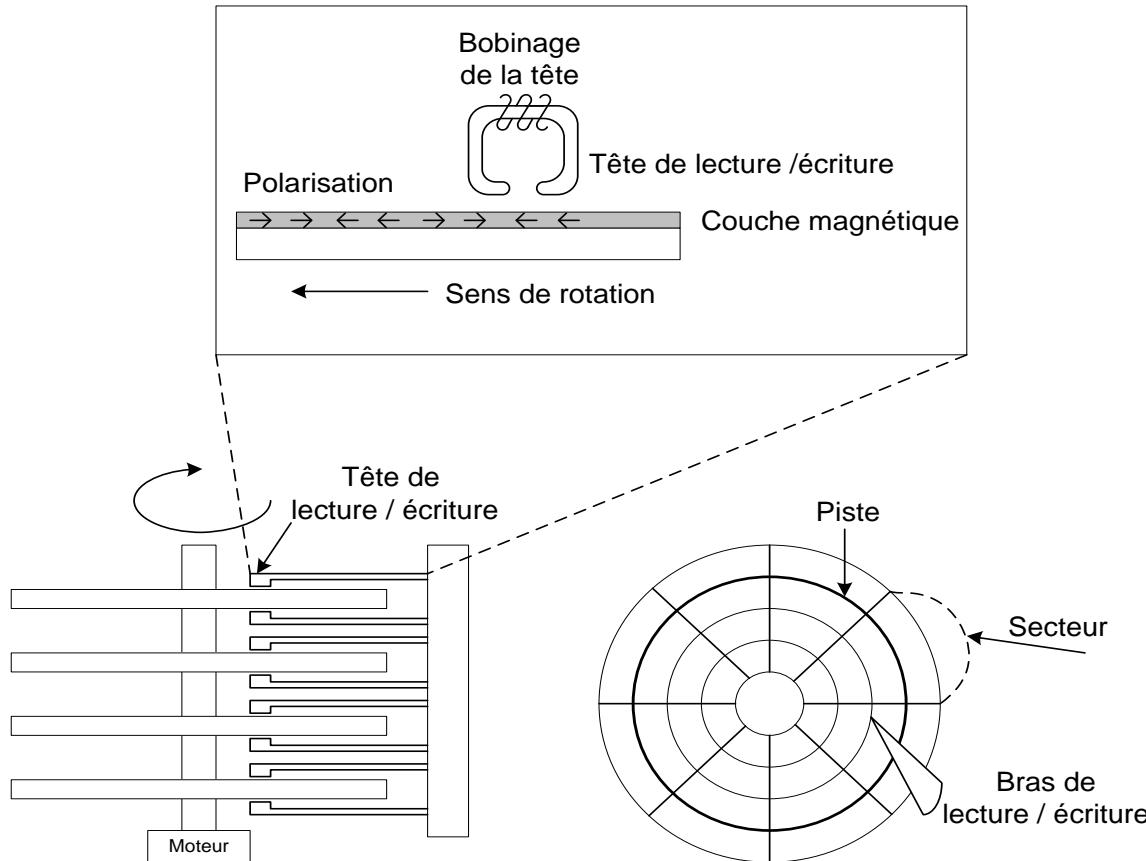
## Transfert KB/sec

Clavier	0.03
Souris	0.02
Voix	0.02
Scanner	200
Imprimante matricielle	0.5
CD	153
Disque Dur	150000



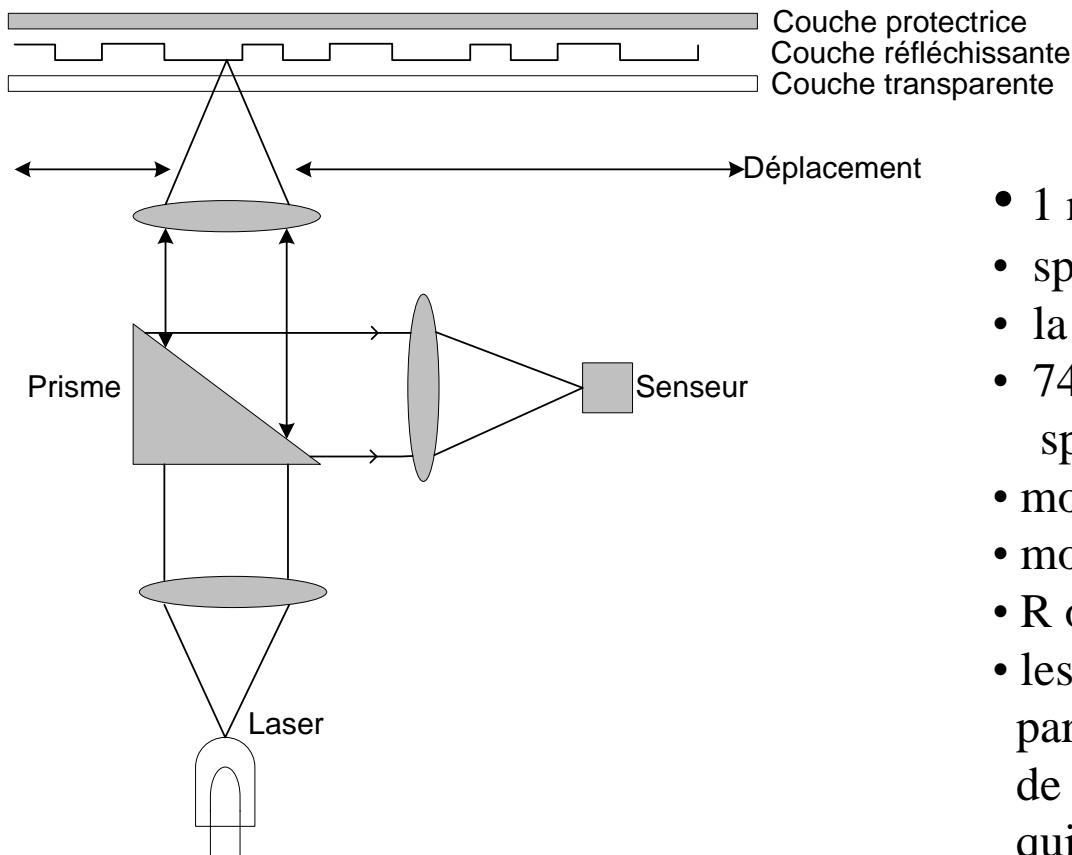
L  
E  
S  
  
P  
E  
R  
I  
P  
H  
E  
R  
I  
Q  
U  
E  
S

# Les mémoires de masse: disque dur



- $10000 \text{ t/m}$
- 1 micron sépare disque et tête
- bras lent
- dizaine de Gbyte
- densité d'écriture variable
- Un exemple: 1024 pistes,
- 64 secteurs.
- Un bloc = 512 Kbytes.
- encodage par magnétisation
- le disque est fourni formaté.

# Disques optiques



- 1 mm sépare la tête du disque
  - spirale plutôt que piste
  - la vitesse s'adapte
  - 74 mn pour la spirale de 6km
  - moins rapide que le disque dur
  - moins encombrant
  - R ou RW
  - les bits sont codés par la présence de réflecteurs ou de bosses qui dispersent la lumière.
- Aujourd'hui → Blu-ray (100Go)

# Mémoires électroniques non volatiles

---

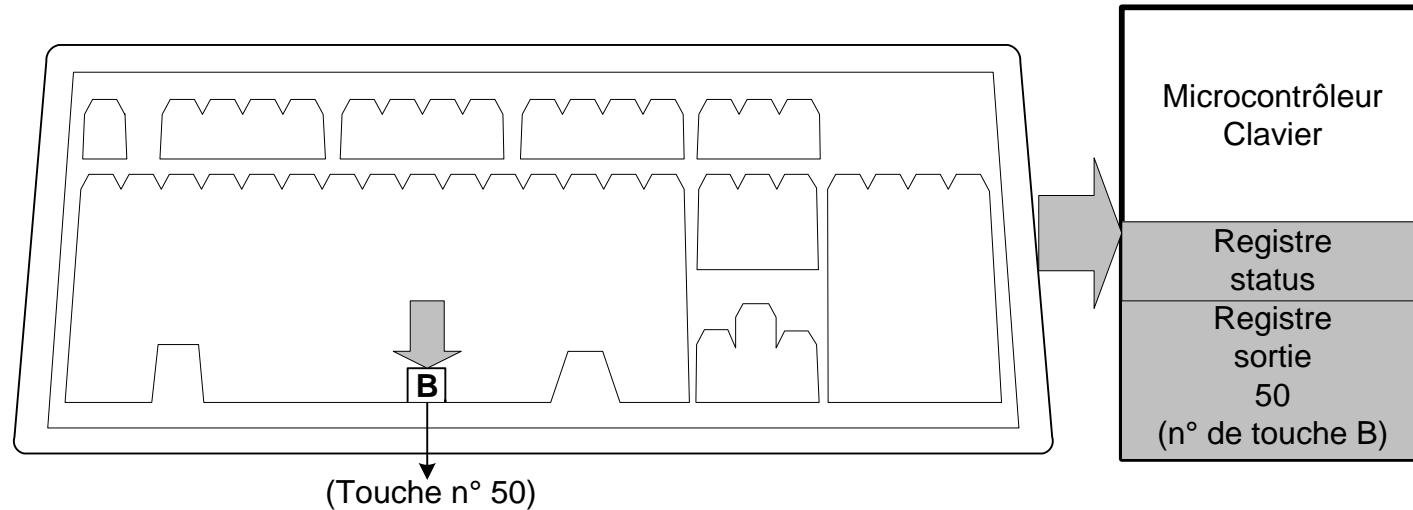
- Mémoire flash:

- Principe: Mémoire électronique de type EEPROM (transistors MOS), fonctionne comme la mémoire vive mais est préservée même sans alimentation. Consomme très peu de courant, très résistante aux chocs
- Supports courants:
  - » Sticks (Clés) USB
  - » Cartes SD (appareils photos, PDA, etc.)
  - » Disques durs SSD (Solid State Drive)
    - ➔ Moins volumineux mais beaucoup plus rapides que les disques magnétiques

- Cartes à puces:

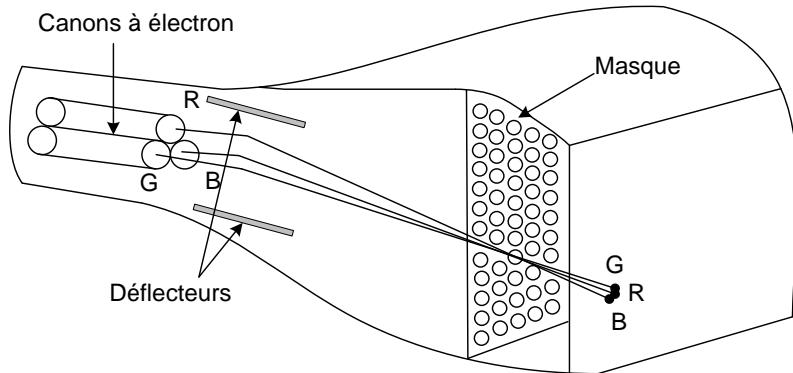
- Mémoires de taille limitée mais cryptée et résistante aux chocs

# Le clavier

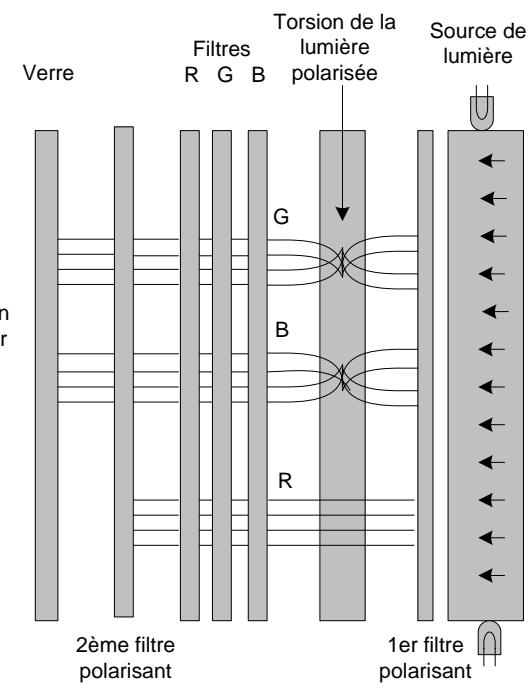


Quand une touche est pressée (variation de résistance),  
un signal est envoyé au contrôleur du clavier.  
Le signal est traduit en ascii et envoyé (sur un bus série) au CPU.  
Un différent traitement est appliqué pour les touches modificatrices  
dont le code est enregistré en mémoire afin d'être traité avec la touche  
successive.

# L'écran



Ecran cathodique



Ecran plat

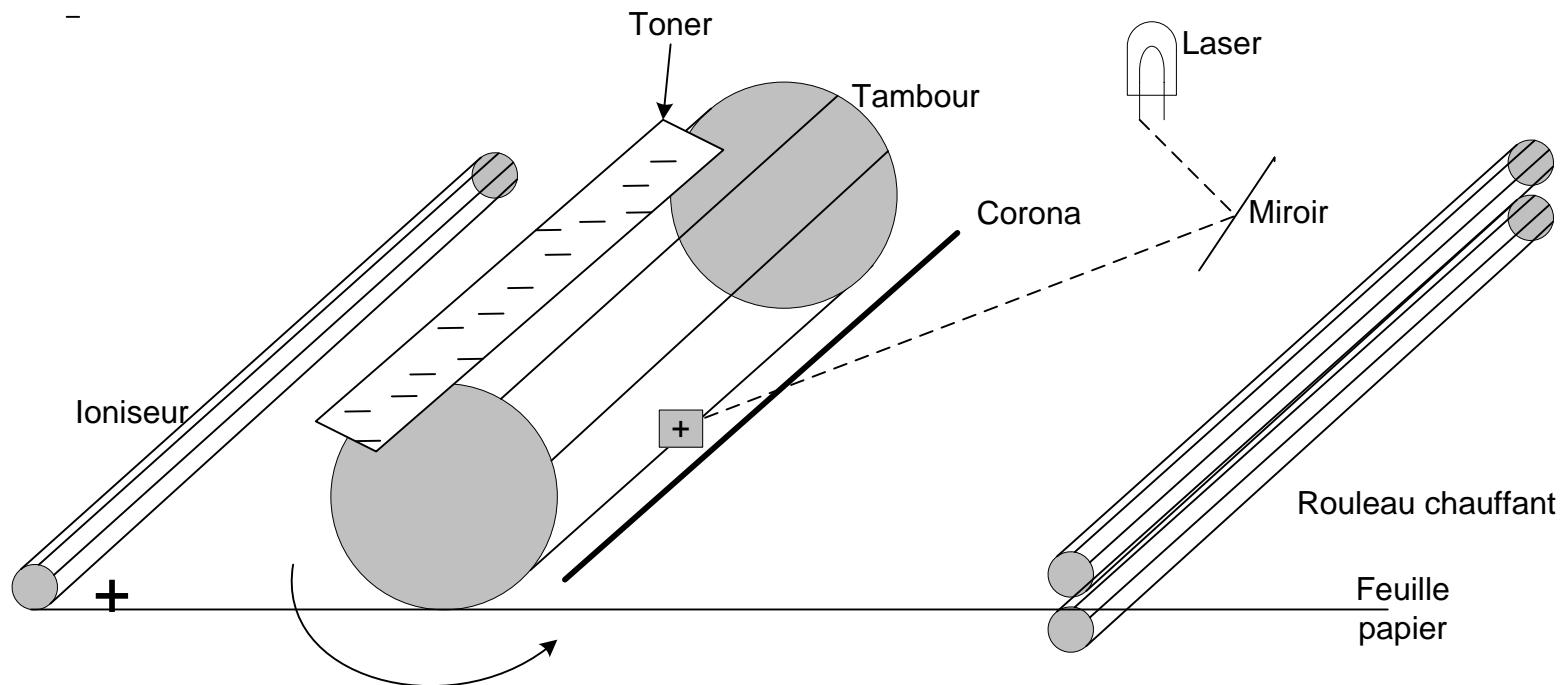
---

## □ Les écrans (les moniteurs)

- il est nécessaire d'avoir un bon contrôleur du vidéo, la carte graphique qui traduit l'information digitale en analogique
- la dimension = la diagonale 15 ou 17 pouces, la fréquence de rafraîchissement - 80 Hz (80 écrans / sec.)
- les principes des moniteurs semblables à l'écran télé mais percée récente des écrans plats (ou à cristaux liquides)
- les signaux numériques envoyés par le CPU contiennent des informations de couleur sur les pixels et des informations de déflection horizontale et verticale qu'il faut synchroniser avec le balayage d'écran.
- les écrans classiques, trois pastilles de couleur (RVB) reçoivent 3 faisceaux d'électron et émettent une brillance proportionnelle à l'intensité du faisceau d'électron.

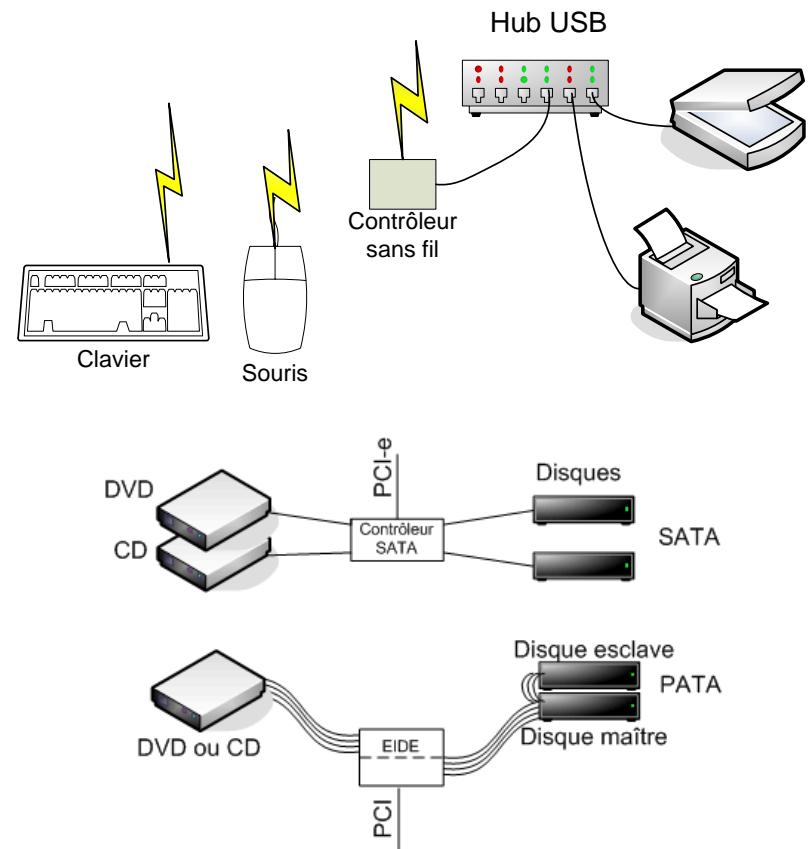
- 
- la résolution est également importante: le nombre de pixels qui constituent l'image sur l'écran. Typiquement 768 lignes de 1024 pixels.
  - s'il y avait une large possibilité de couleur, cela créerait une pression trop forte sur les transferts de données entre le CPU et le moniteur. D'où des choix restreint --> 256 couleurs (1 byte).
  - l'image est transférée en binaire, du CPU dans la mémoire interne du moniteur. Cette mémoire est ensuite balayée et à chaque pixel, l'information est transformée en analogique pour déterminer les intensités des faisceaux d'électron et la déflexion du faisceau.

# L'imprimante



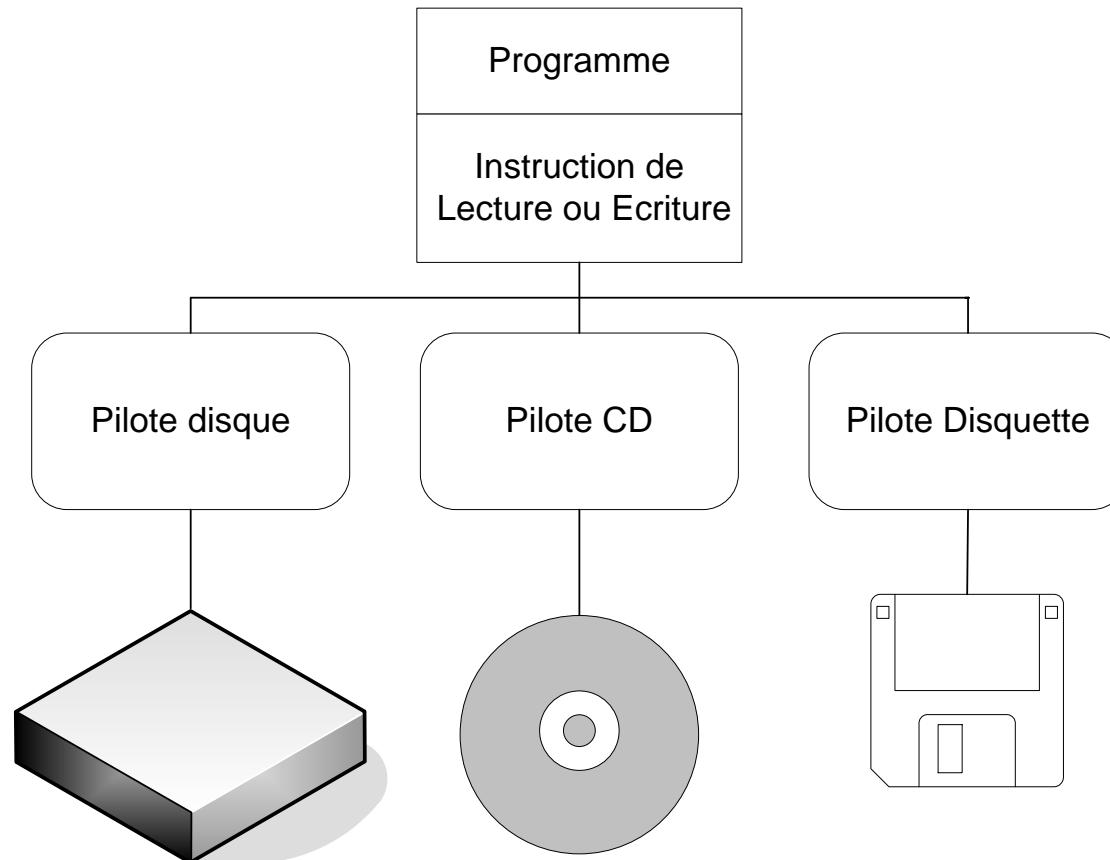
# Raccordement des périphériques

- Un domaine en constante évolution et tendant à l'uniformisation.
- Port: relie un seul périphérique au travers d'une interface spécialisée: port clavier, ou COM-x, port PCI -e/GPU (pour l'affichage), port parallèle, port LAN
- Bus: PCI, USB, IEEE 1394
- Contrôleur: assure le dialogue entre le périphérique et le CPU.  
IDE ou SCSI pour les disques durs:



# Les périphériques en action

---

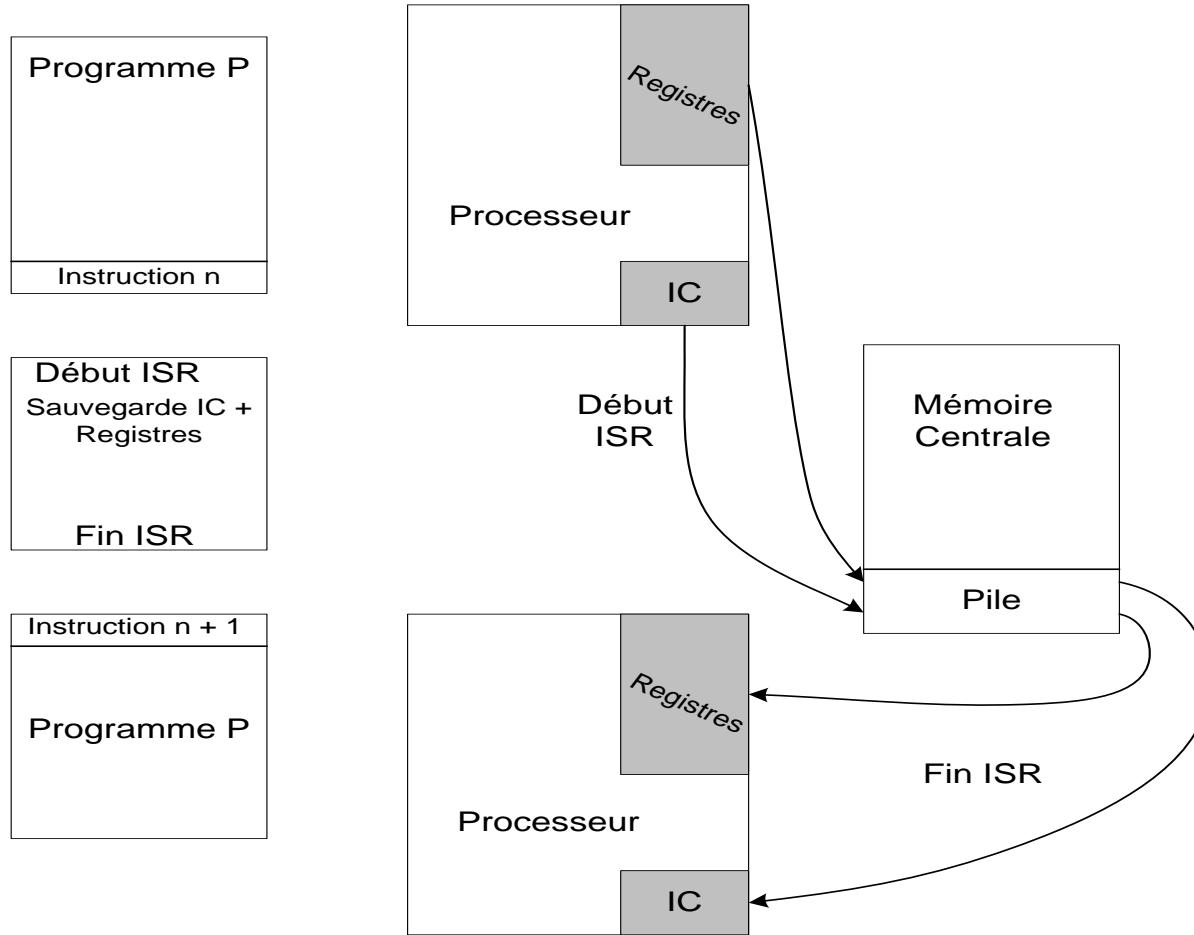


# Le mécanisme des interruptions

---

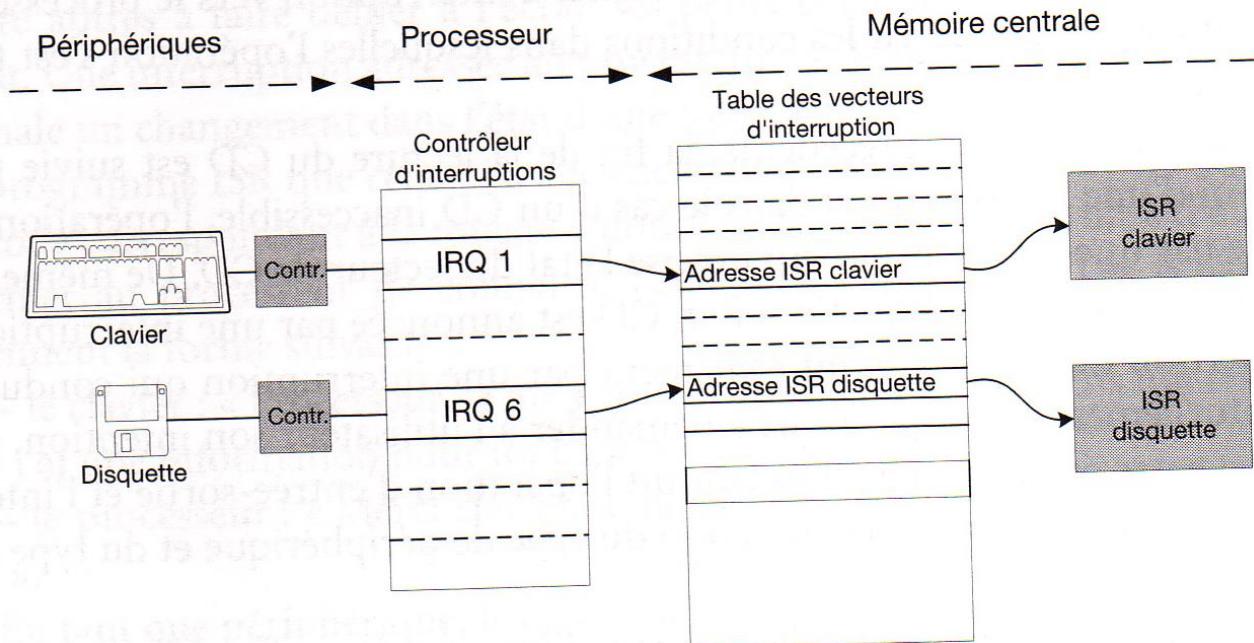
## □ Les interruptions

- interrompent le cours normal du programme et donc sortent du contrôle du CPU.
- Elles permettent plus de flexibilité.
- elles peuvent provenir du clavier, d'un I/O qui signale qu'il a fini, d'un événement extérieur inattendu (panne de courant,...), d'un programme, ou permettre le multitasking (allouer du temps CPU à différentes tâches).
- le CPU est connecté à plusieurs lignes d'interruption IRQ1 - IRQ15.
- le périphérique qui demande l'interruption: clavier ou fin de I/O doit s'identifier auprès du CPU, ainsi que la nature du service demandé par le périphérique. Ce service est une routine spécifique qui prendra possession du CPU. Cette identification peut se faire simplement via la ligne ou par un code pré-définit.



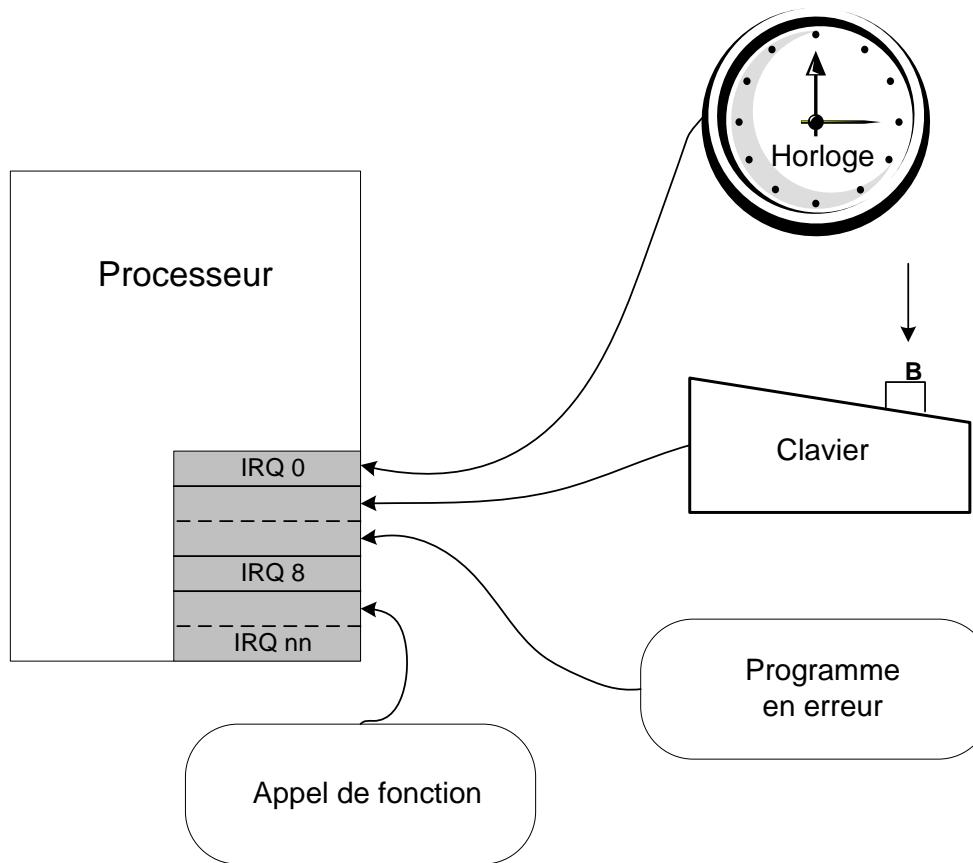
- 
- toute l'information concernant l'état actuel du programme est sauvee sur un «stack» et les registres. Ensuite, la première instruction du «service d'interruption» est chargee dans l'IR.
  - Quand la routine est terminée, elle peut soit rendre le contrôle au programme ou modifier complètement le cours des choses (par exemple, une interruption d'imprimante pour dire qu'elle est sans papier)
  - une interruption d'un événement anormal peut venir de l'extérieur (panne) ou être générée par le CPU lui-même (divisé par 0)
  - les interruptions peuvent s'imbriquer exigeant alors un ordre hiérarchique de leur priorité. Une plus haute priorité pourra interrompre une plus basse priorité. Une panne a une haute priorité, le clavier une moyenne, une fin d'I/O une basse.
  - les interruptions peuvent être désactivées (masquées)

- 
- Déroulement du traitement d'une interruption:
    - On reçoit un vecteur d'interruption (informant sur sa nature)
    - A partir de cette information, on cherche le programme de gestion de cette interruption
    - On sauve l'état actuel du programme: le registre d'état (flags), le contenu du segment de code et surtout le contenu du PC
    - On charge le PC avec l'adresse du programme de gestion de l'interruption
    - A la suite de ce programme, on restaure tout ce que l'on avait sauvé précédemment.



**Figure 4.15** Cheminement d'une demande d'interruption

# D'où proviennent les interruptions

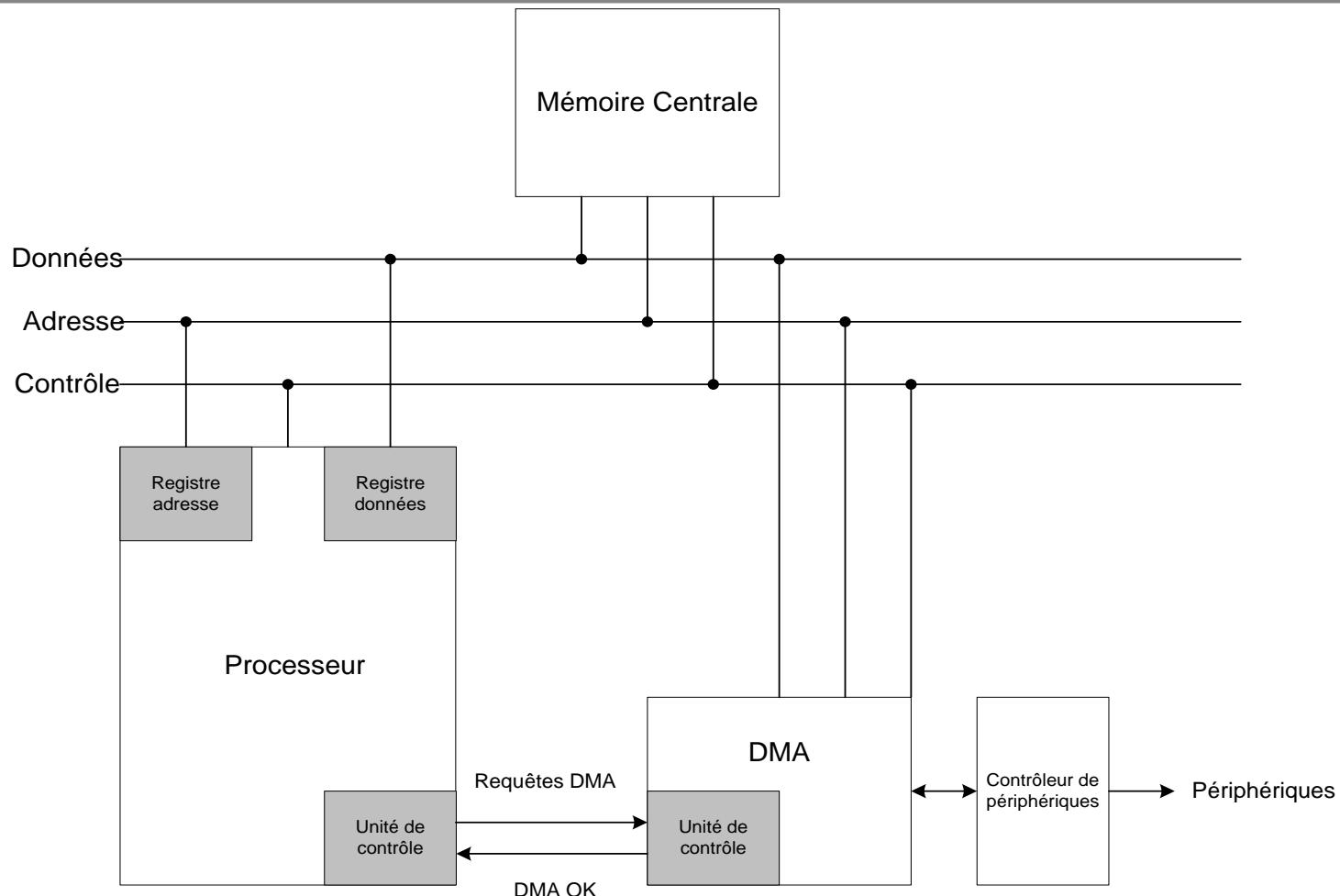


# Transfert de données et DMA

---

## □ Les DMA (Direct Memory Transfer)

- nécessaires pour transférer des blocs de données, rapidement.
- Sur les PC, il existe 8 canaux DMA
- on court-circuite le CPU. Il peut donc s'occuper d 'autres process.
- le transfert se fait sous le contrôle du contrôleur du périphérique
- il est initié par une interruption, puis le CPU disparaît de la circulation.
- la fin du transfert est signalé par une interruption
- le CPU peut faire autre chose (utile pour le multitasking ou le multiusers)
- le contrôleur I/O doit connaître l'adresse au niveau du périphérique et de la mémoire, la quantité de données à transférer et s'il s'agit d'une lecture ou écriture.
- possibilité de détecter les erreurs de transfert et de les corriger par l'addition de bits de parité. Le plus simple est 1 bit (pair/impair) mais on peut recourir a plusieurs bits qui permettent alors la correction d 'erreur.



# Le futur des interfaces

---

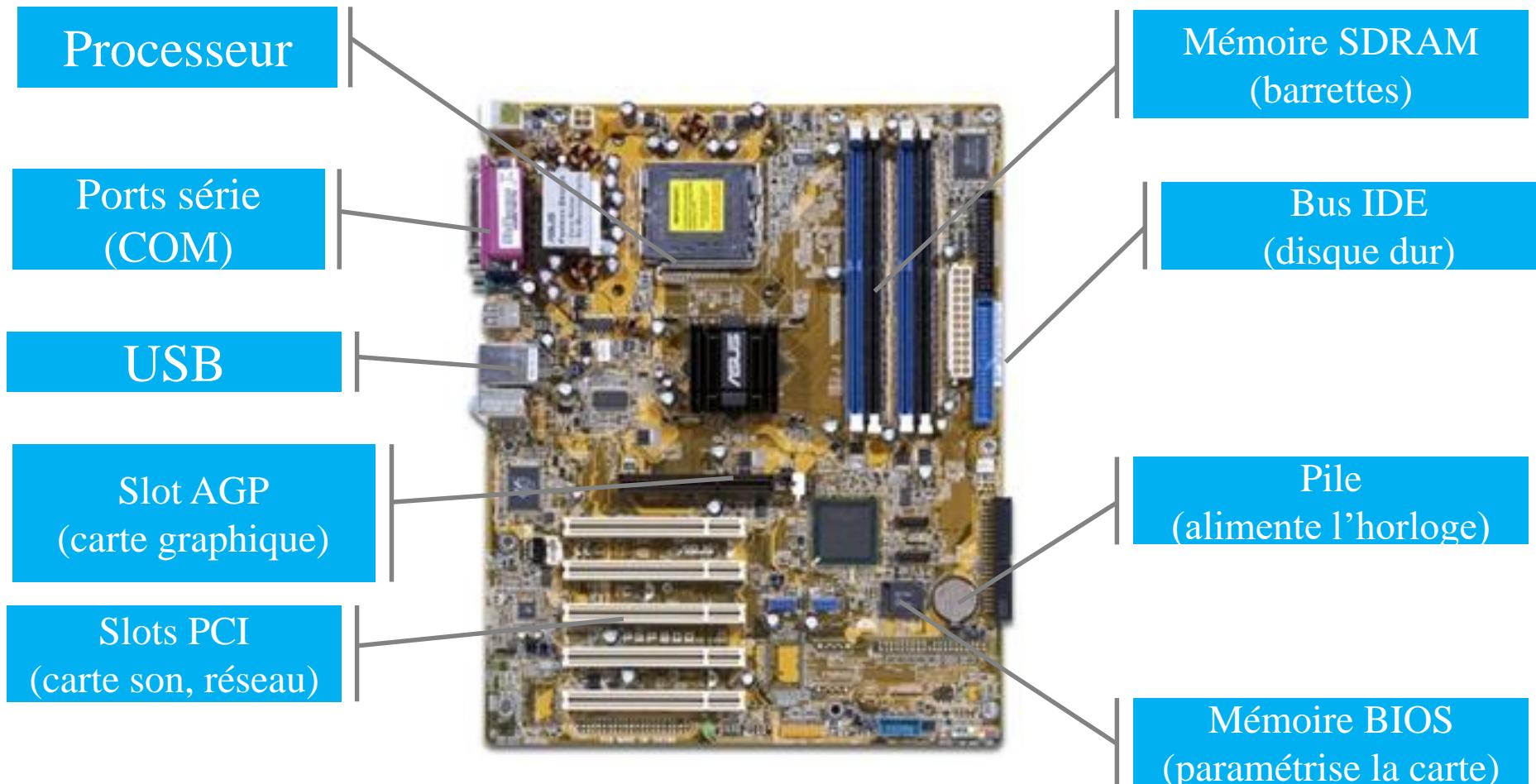


# De nouvelles modalités sensorielles

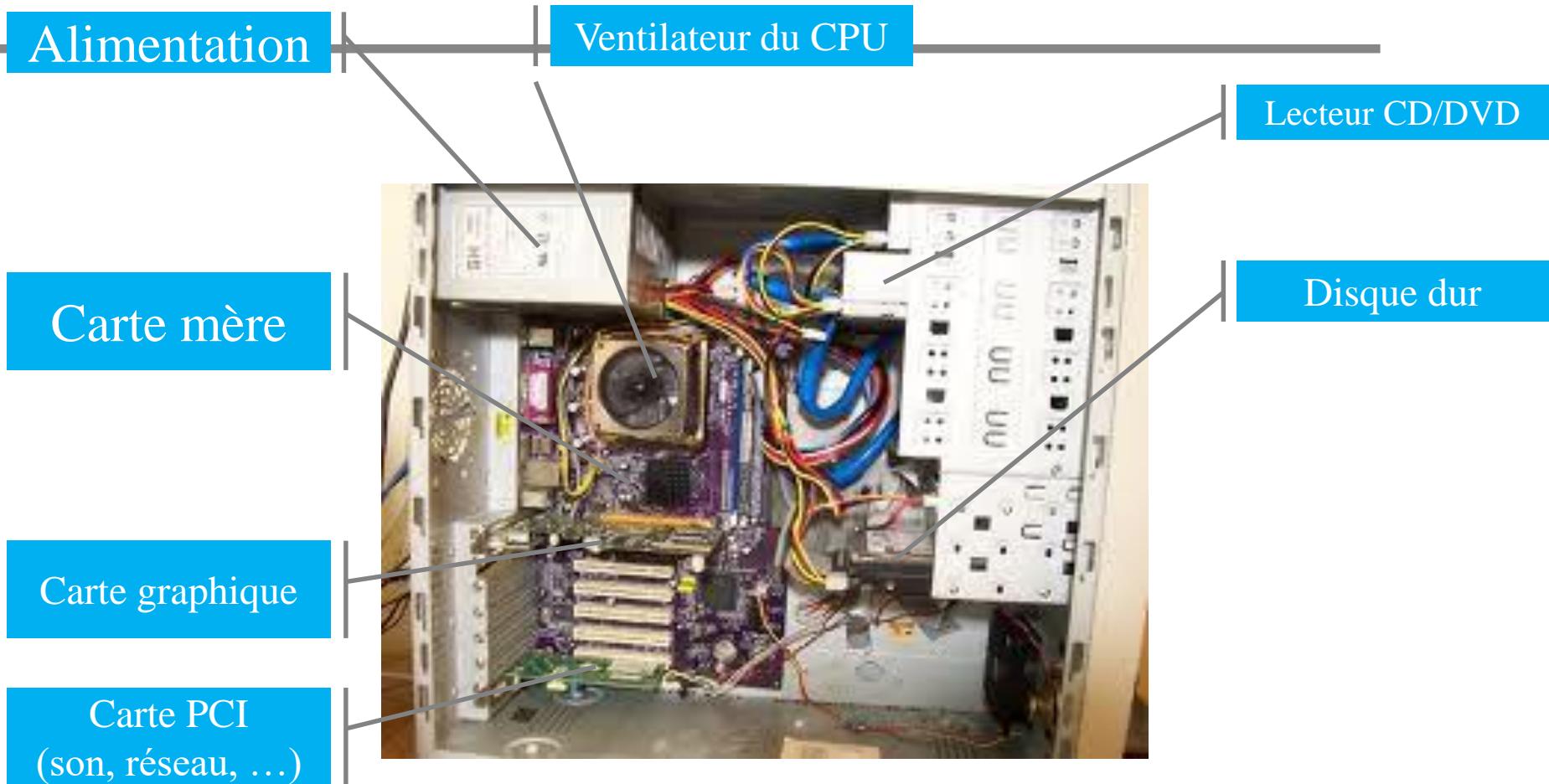
---

- Aujourd’hui surtout le toucher mais pas les oreilles et le yeux.
- Reconnaissance vocale, 98% de qualité, sans bruit de fond et pour un locuteur unique. Parfait, lorsque les mains sont occupées.
- Pavé tactile (le doigt crée une conductance électrique)
- Écran tactile (écran ultrason ou infrarouge). Extension au multi-doigts.
- Accéléromètre
- Reconnaissance gestuelle → réalité virtuelle
- Communication télépathique

# Vue d'ensemble de l'ordinateur: Carte mère



# Vue d'ensemble de l'ordinateur



# V. Les logiciels

---

# Le système d'exploitation (OS) - généralités

---

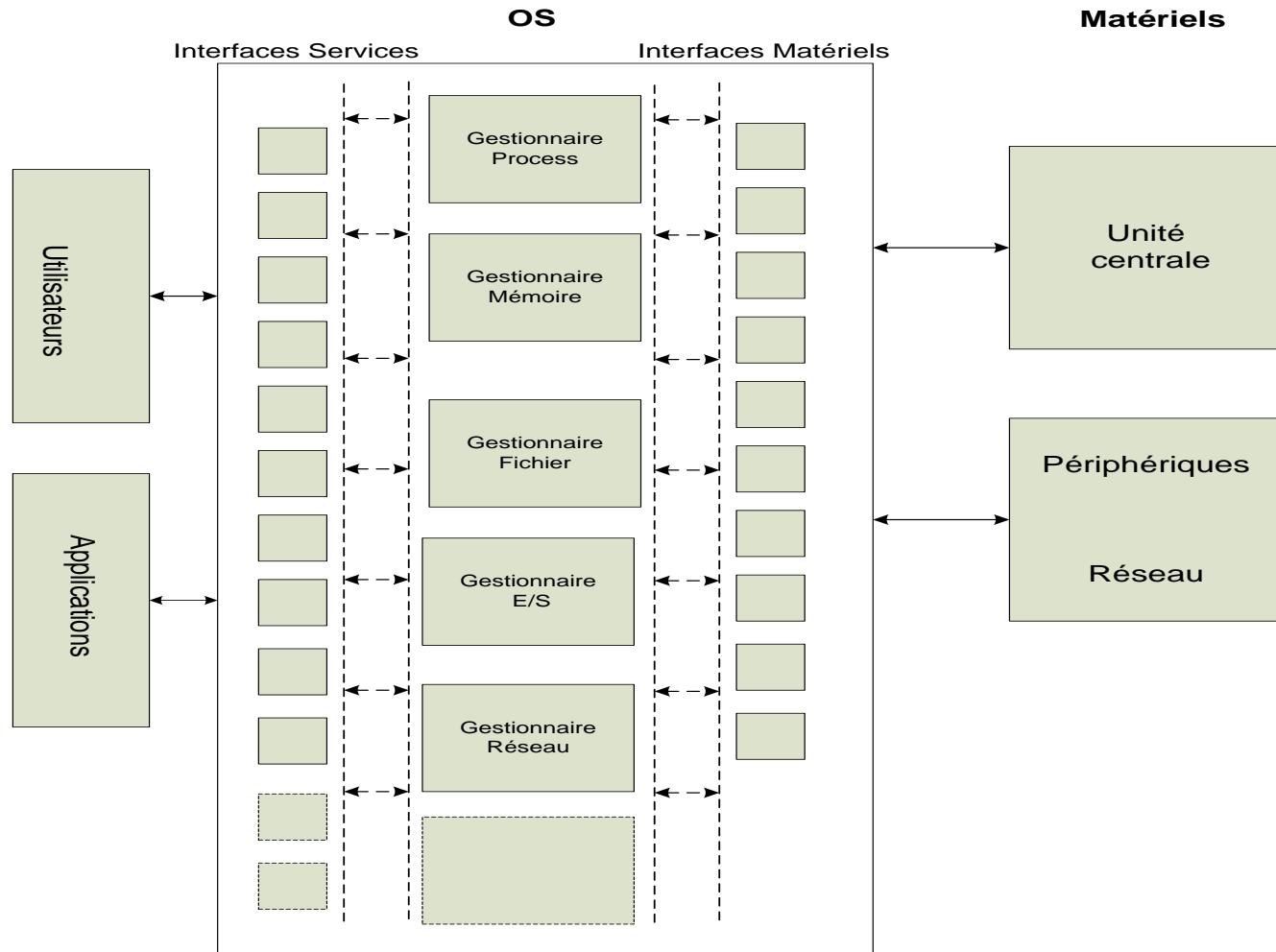
- La définition classique de l'OS: les 3 fonctions
  - » 1: présenter une interface unifiée pour les services les plus usités.
  - » 2: gérer les ressources de l'ordinateur (processeur, RAM, périphériques); assurer, le cas échéant, leur partage entre un ensemble d'utilisateurs et organiser leur répartition entre les tâches multiples assignées à l'ordinateur
  - » 3: assurer cet ensemble de services en présentant aux utilisateurs (humains ou programmes) une interface mieux adaptée à leurs besoins que celle de la machine physique; cette interface est celle d'une « machine virtuelle » qui fournit un ensemble de fonctions pour la gestion et la communication, et pour la réalisation de logiciel d'application.
- D'abord: permettre une interface plus conviviale entre l'utilisateur et le hardware: pour la gestion des fichiers, pour la gestion des I/O, pour lancer des programmes. Ceci tant de l'extérieur d'un programme qu'à l'intérieur d'un programme. Il s'agit du langage de communication entre l'ordinateur et l'utilisateur.

- 
- ensuite: faire une gestion des ressources et du hardware de manière à optimiser leur utilisation. Ceci est d'autant plus capital que ces ressources sont soumises à de multiples demandes: multitâche, multiutilisateur, réseau, ....
  - il y a beaucoup d'OS et de philosophies d'OS différents: UNIX, Linux, DOS, WINDOWS 9X et NT, 2000, XP, VISTA, MAC-OS, Léopard (???)....
  - C'est l'enjeu d'une terrible bataille commerciale et stratégique car l'OS est le « maître de l'ordinateur »
  - il existe autre chose que Windows !!!!!
  - Windows vs Linux
  - L'UNIX (multi-utilisateurs, multi-tâches) est l'ancêtre vers lequel tout le monde retourne.
  - l'OS tourne toujours en arrière-plan pour reprendre la main quand un programme la laisse pour l'exécution d'un service.
  - L'OS est le premier programme qui se charge dans la RAM.
  - C'est le « booting » de l'ordinateur.
  - Les applications ne peuvent “dialoguer” qu'avec un OS donné

# Logiciels libres (open source)

---

- Programmeurs = chercheurs, son code doit être ouvert.
- Programmeurs = créateur de valeur → concurrence → confidentialité → son code doit être fermé.
- Situation paradoxale !!
- Linux est open source → son code est disponible à tous.
- Ainsi qu'Apache, MySQL, Mozilla, Java
- Libre n'est pas synonyme de gratuit !!!!
- Ainsi IBM s'enrichit considérablement au départ de logiciels libres.
- Wikinomics = arme de collaboration massive
- Le deuxième Web.
- Le Web = division du travail



- 
- L'OS est responsable du traitement des fichiers en réponse aux requêtes de l'utilisateur: copy, open, execute, move, ...
  - L'OS est responsable du traitement et la gestion des I/O en réponse aux requêtes de l'utilisateur: print, save, keyboard, modem, internet,...
  - Il gère le démarrage de l'ordinateur, et le «bootstrapping» - «il doit se charger lui-même». Charger l'OS s'appelle le «booting» et s'exécute à partir du BIOS.
  - Il permet l'exécution «apparemment» simultanée de plusieurs applications ou programmes et l'utilisation «apparemment» simultanée de plusieurs utilisateurs.
    - pour cela, il gère les ressources demandées par les utilisateurs et les programmes: CPU, mémoire, I/O,...
    - il permet des protections et des communications entre utilisateurs et programmes
    - Il fournit des informations pour qu'un utilisateur privilégié: « le system manager » gère les ressources et optimise le système.

- 
- de manière synthétique, on peut catégoriser historiquement les OS en 4 classes:
    - utilisateur unique, tâche unique
    - utilisateur unique, tâches multiples
    - utilisateurs multiples, tâches multiples
    - systèmes distribués
  - l'OS est un programme de type «event-driven», il n'intervient que quand on lui demande d'intervenir par une commande ou via un programme, en réponse à une commande fichier, un I/O, un input de clavier, une demande de mémoire par un programme, ....
  - L'OS prend le contrôle, ou en réponse à une requête de l'utilisateur ou la par le jeu des interruptions (qui permet par exemple l'échange du CPU entre les programmes).

- 
- les OS peuvent tourner sur un large ensemble de plates-formes informatiques. Les deux OS les plus fréquents sont Windows et Unix. Partager les OS entre plates-formes permet une meilleure «portabilité» des programmes et des applications (qui interagissent avec cet OS)
  - les OS sont rendus «universels» par leur écriture dans un langage de programmation de haut niveau qui peut être compilé et exécuté sur de nombreuses plates-formes. C et C++ sont des langages idéaux d'écriture d'OS. De fait, l'OS est un programme comme les autres mais qui se trouve en amont de tous les autres et peut interagir avec tous les autres.
  - en plus, une même plate-forme peut tourner avec plusieurs OS. Sur votre même PC, vous pouvez exécuter Windows, NextStep ou Linux
  - il faut différentier le «kernel» de l'OS (le cœur) du «shell» qui peut-être de type GUI (Graphical User Interface), les plus répandus aujourd'hui, ou de type ligne de commande (CLI - Command Line Interface).

- 
- un kernel peut donc donner lieu à plusieurs shell, ce qui est typiquement le cas avec UNIX (avec une seule commande on peut changer de shell, on peut aussi configurer le shell initial) et le contraire avec Mac-OS.
  - certains OS (Unix, MS-Dos) permettent l'écriture de fichier reprenant une suite de commandes OS qui peuvent être exécutés comme un programme normal (les fichiers BAT en MS-Dos). On y intègre, par exemple, des fonctionnalités comme le «piping» ou la redirection des I/O,... Unix est le plus complet en cette matière.
  - de manière plus complète, les 10 occupations de l'OS sont: l'interface utilisateur, le service fichier, le service I/O, la gestion des «process» et des «threads», la gestion mémoire, le «dispatching», la gestion des mémoires disque et autres supports secondaires, la sécurité et la protection, réseaux et communications, le support au «system manager» ou «system administrator» (le «superuser» en UNIX)

# Multi-tâches et multi-utilisateurs: Organisation en processus.

---

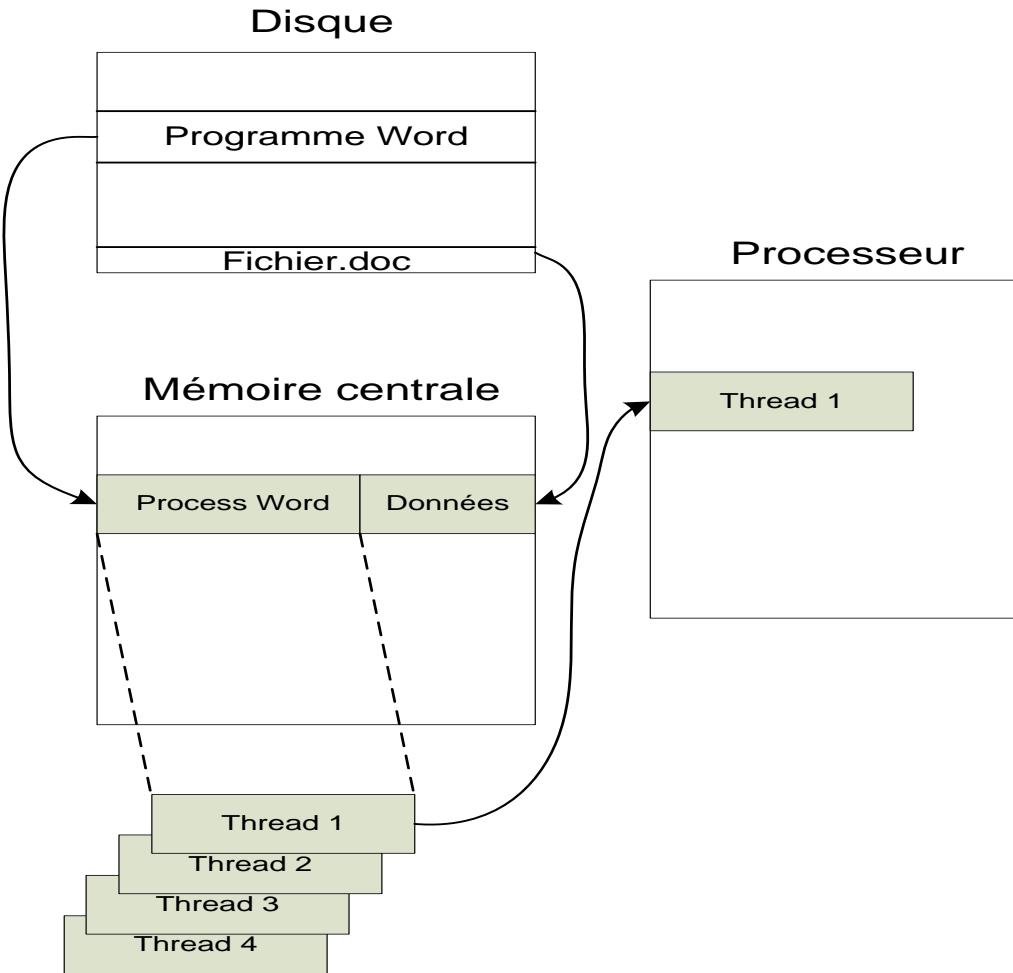
- Un seul programme est peu économique pour le CPU: notamment à cause du temps perdu pour les I/O.
- Aujourd’hui tous les ordinateurs permettent le multitâche.
- Le rôle supplémentaire de l’OS devient d’allouer les ressources CPU et mémoire entre les tâches. C’est la procédure de «dispatching» qui prend en compte les I/O (quand un programme attend pour son I/O, un autre peut s’exécuter) et une répartition équitable du temps CPU entre les programmes. C’est aussi la gestion mémoire qui doit partitionner la RAM entre les programmes. Les différentes gestions ne sont pas indépendantes: plus on met de programmes dans la RAM, plus le «dispatching» doit gérer la répartition des autres ressources
- la notion de «processus» - le «processus» est un programme qui s’exécute avec ses ressources I/O, les fichiers ouverts, sa mémoire.

# Le bloc descripteur de processus

---

- A chaque processus est associé un «bloc descripteur» qui contient toutes les informations suivantes:
  - un identificateur du processus,
  - l'état courant du processus,
  - un espace pour la sauvegarde du contenu des registres du processeur lorsque le processus est provisoirement interrompu dans son exécution,
  - l'adresse de sa table de correspondance entre pages virtuelles et pages réelles,
  - la liste des ressources nécessaires en termes de mémoire et fichiers,
  - le niveau de priorité éventuel à considérer dans l'affectation des ressources,
  - une spécification de ses permissions d'accès (la zone mémoire occupée par le processus est-elle accessible par d'autres processus)
  - le propriétaire du processus (par exemple l'utilisateur ou le processus l'ayant déclenché),
  - la liste des processus enfants (c'est-à-dire les processus déclenchés à partir de celui-ci).

- 
- quand un processus est créé et prêt à être exécuté, il est «ready» et doit passer au «dispatching» qui, s'il est choisi, le fait passer au stade «running». Plusieurs processus peuvent être bloqués ou «ready» mais un seul peut être «running». Quand un processus demande un «I/O», il devient «bloqué». Quand l'I/O est terminé, il redevient «ready».
  - les «threads» sont des espèces de «mini-processus». Les threads ont leur propre «program counter», registres, mais ils appartiennent au même processus, avec les mêmes données. En gros les threads sont des petites routines d'un même programme qui peuvent être traités comme des «mini-processus». Les «threads» ont pris de l'importance avec l'intensification des programmes «event-driven» et des programmes animés. On les retrouve dans les langages de programmation de avancés: C++ et JAVA.

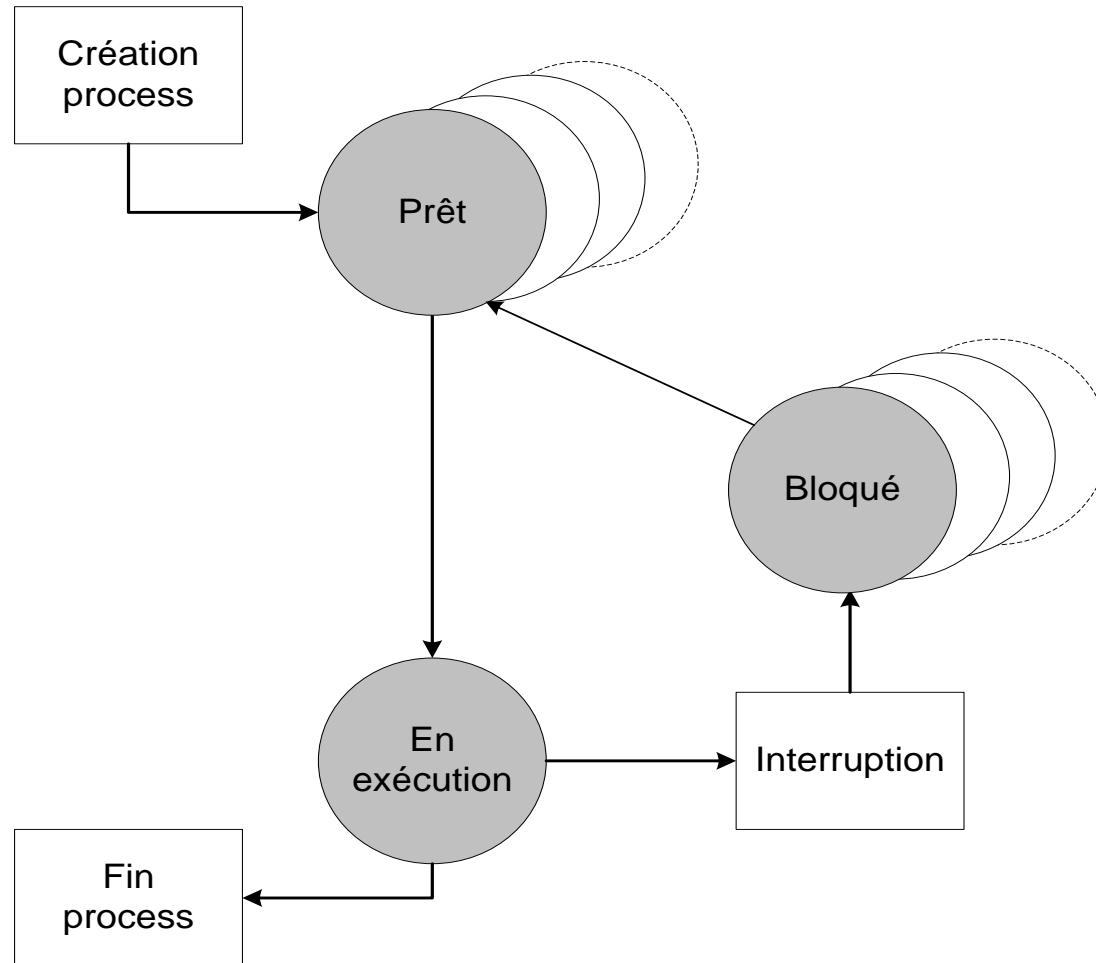


# Gestion du processeur

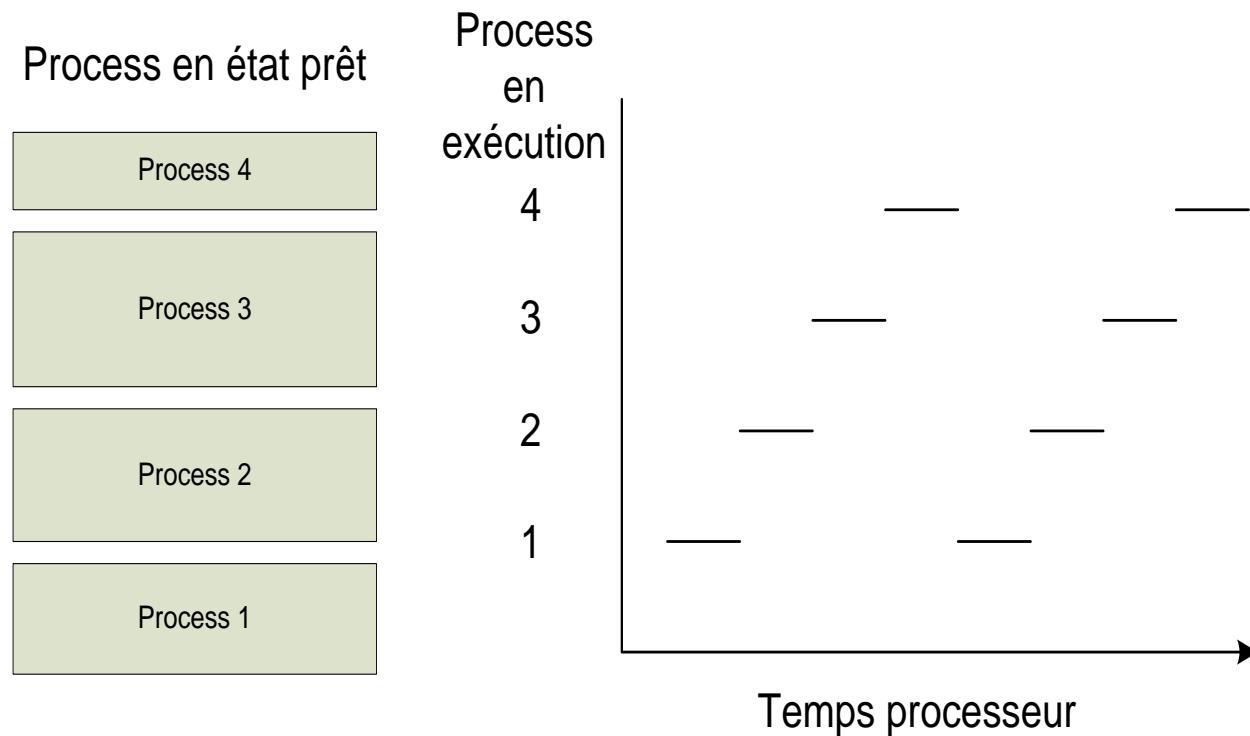
---

- le but du «dispatching» (allocateur) est de sélectionner le prochain «processus» pour le CPU. Une fois choisi, l'OS se limitera à mettre dans le registre de compteur d'instruction, l'information, reprise dans le registre spécifique du «bloc descripteur», de l'adresse de l'instruction à exécuter.
- c'est le mécanisme d'interruption qui est mis à l'œuvre.
- il y a une multitude de possibilités pour l'algorithme de «dispatching». Les anciens algorithmes étaient tels qu'une fois un processus sélectionné, il s'exécutait jusqu'à la fin. Le choix était de type: First-in-First-out, ou d'abord les petits programmes ou encore sur base de priorité assigné à chacun des programmes.
- les algorithmes plus récents sont plus flexibles et «démocratique». Le «round robin» alloue à chaque processus un certain «quantum» de temps et tous les «processus» tournent. Les processus faisant de l'I/O sont pénalisés car ils tourneront beaucoup plus que les autres. Dès lors, certaines variations favorisent ceux qui font beaucoup d'I/O car ils laisseront naturellement leur place en cas d'I/O.

# Les différents états d'un processus



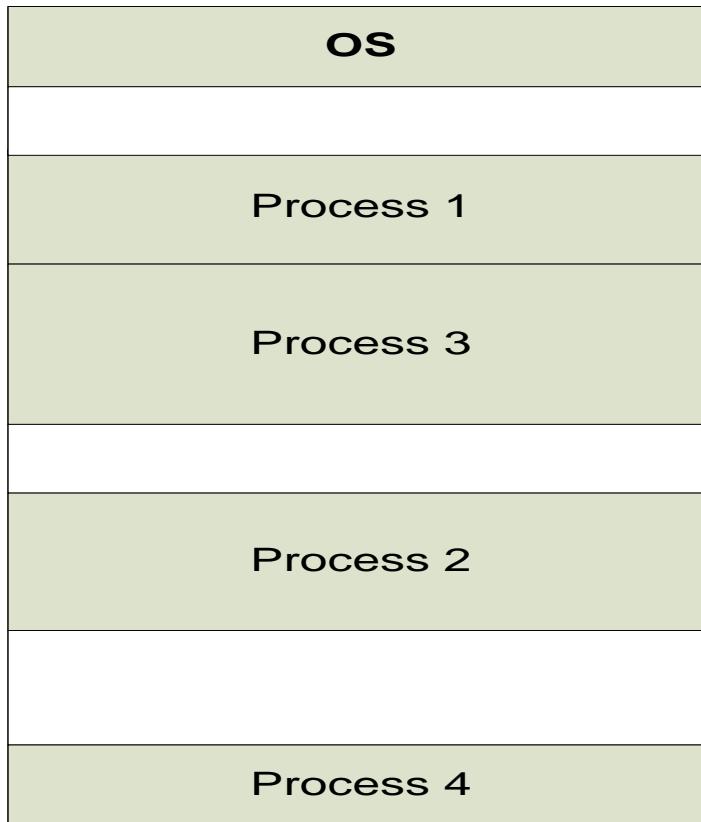
# Allocation du temps processeur aux processus: stratégie préemptive.



# La gestion mémoire

---

- la gestion mémoire consiste à allouer à chaque programme à exécuter un espace mémoire. La mémoire est partitionnée en plusieurs programmes. La partition est soit totalement équitable soit prend en considération la taille du programme. Mais l'emplacement mémoire ne sera pas continu.
- on retrouve des problèmes de fragmentation semblables à ceux que l'on retrouve sur disque mais pour la RAM cette fois-ci.
- L'installation de programmes dans différents emplacement mémoire justifie également des modes d'adressage relatifs plutôt qu'absolus.
- Un programme pourra se « planter » quand on essaiera d'adresser une information qui se trouve hors de l'espace mémoire qui lui a été alloué.
- Il faut bien sûr toujours et surtout protéger l'espace mémoire de l 'OS.
- il faut rajouter à cela la mémoire virtuelle qui doit être gérée également par l'OS. La mémoire virtuelle permet d'accroître considérablement l'espace mémoire (par l'espace disque) en utilisant des adresses logiques dont la correspondance avec les adresses physiques se fait dans un deuxième temps.



# La mémoire virtuelle

---

- via la mémoire virtuelle, chaque processus tourne avec sa propre mémoire et sa propre table de correspondance (table des pages)
- en cas d'absence de la page requise dans la mémoire RAM, l'OS va chercher sur le disque dur la page absente pour la charger dans la RAM.
- cela marche dû au principe de localité
- Si toutes les places sont occupées dans la RAM, on remplace une page par une autre en suivant l'un ou l'autre critère: First-in-First-out, la moins récemment utilisée, la moins fréquemment utilisée, ....
- Dans certains cas, la page remplacée appartient au même processus, dans d'autres cas, non.

# La gestion des I/O (entrées/sorties)

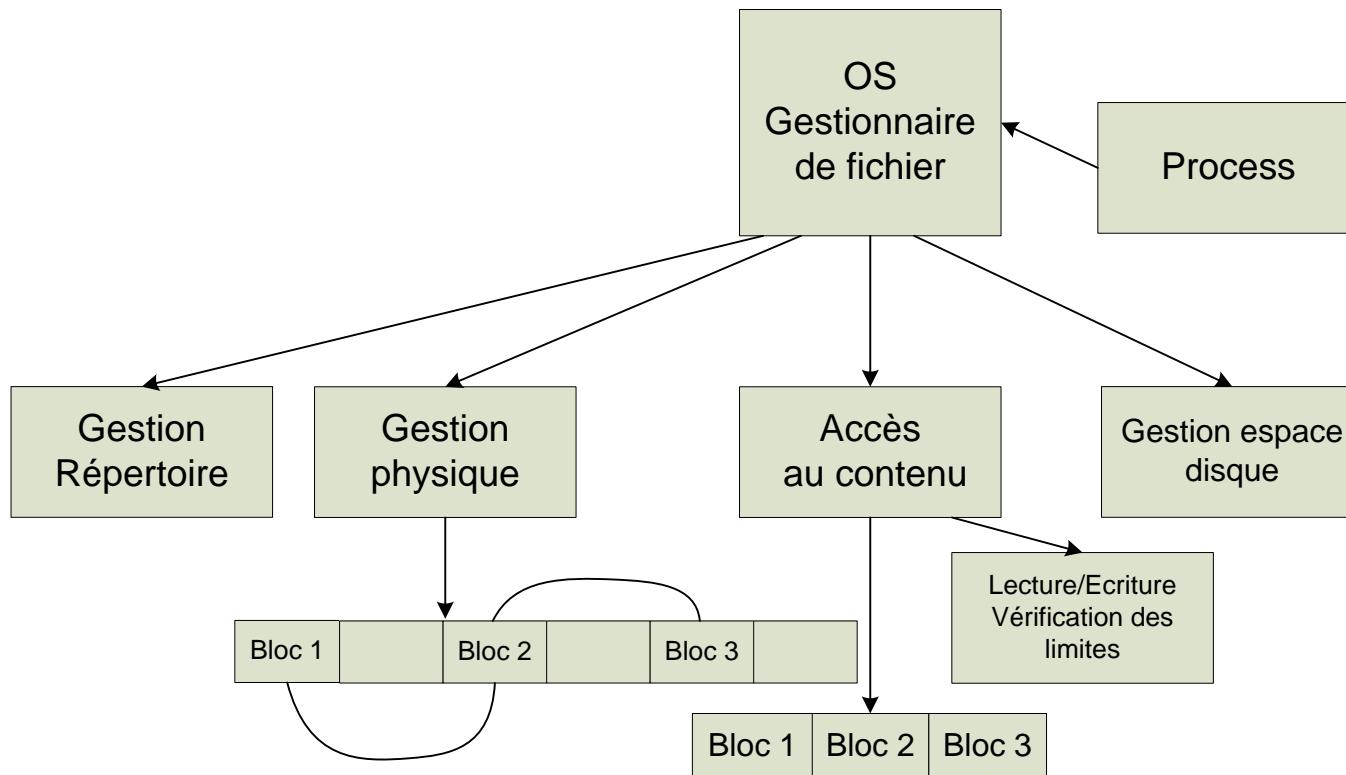
---

- Plug and play – pilotes, côté OS, et contrôleurs, côté périphérique.
- Il peut y avoir plusieurs appels I/O à satisfaire simultanément.
- C'est le mécanisme d'interruption qui gère l'ensemble.
- Là encore, cela peut se faire en suivant une procédure de type First-in-First-out ou en suivant une procédure plus sophistiquée (de priorité)
- Le problème du «deadlock» se produit quand plusieurs processus ont besoin de la même ressource (par exemple plusieurs programmes veulent imprimer) et quand le premier a besoin d'une ressource utilisée par le deuxième qui lui-même a besoin d'une ressource utilisée par le premier. Par exemple un programme occupe de la mémoire qu'il libérera s'il peut imprimer. Mais un autre processus imprime jusqu'à ce qu'il puisse occuper de la mémoire. Cela peut se produire avec plusieurs processus qui sont organisés circulairement.
- On résout ce problème soit en le «prévoyant», soit en «l'évitant» soit en le «corrigeant». Il faut également synchroniser la communication entre les processus.

# La gestion des fichiers

---

- Tout sur le disque dur est sous forme de fichiers.
- Chaque fichier est repérable et accessible par son nom.
- Son contenu dépend du programme qui le traite
- Toute application a une vision assez unique d'un fichier, une séquence de bytes.
- Cela devient facile de le lire, le copier, le déplacer, l'effacer...
- L'essentiel de vos manipulations et de celles de l'OS
- Chaque OS a un système de gestion de fichiers qui lui est propre et souvent différent d'un autre.



- 
- Gestion du répertoire: un système de “dénomination” des fichiers.
  - Structure arborescente
  - Chaque répertoire est lui-même un fichier avec un tas d’infos sur son contenu.
  - L’accès aux fichiers est conditionné par des droits d'accès.
  - Pour Unix et Windows, la gestion du contenu dépend des applications et non pas de l’OS qui voit tout fichier comme une séquence non structurée de bytes.

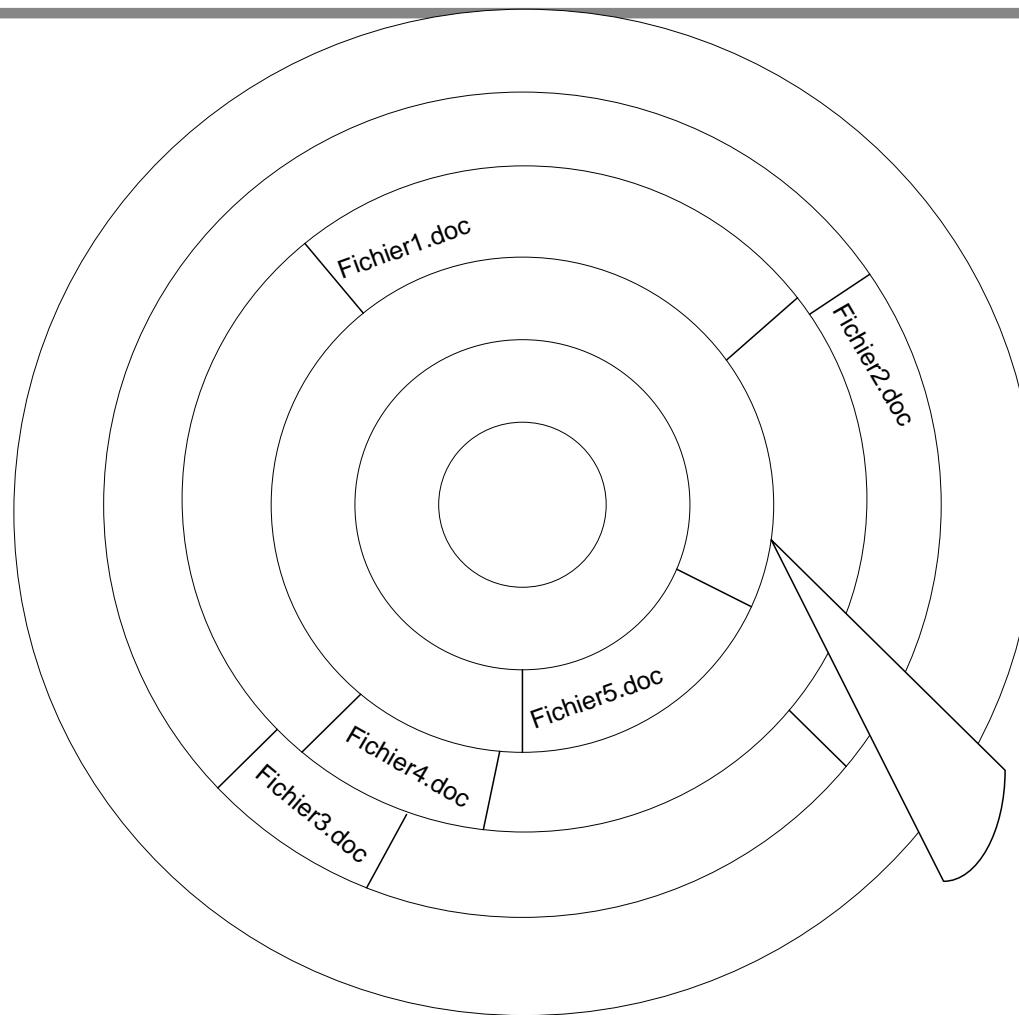
# Stockage physique des fichiers

---

- Le disque dur peut être divisé en partitions – gestion facilitée, table des contenus plus petites.
- L’unité d’allocation et de transfert pourrait être le secteur disque, 512 bytes. Mais la table renseignant sur l’emplacement des fichiers serait trop grosse → grossissement de l’unité à plusieurs secteurs: des agrégats ou cluster de secteurs. Par exemple: 8 secteurs.
- Le mode de stockage le plus simple est le stockage contigu, accès et retrait simplifié.
- Mais problèmes de fragmentation évidents.
- D’où le stockage en liste lié qui convient aussi bien pour des fichiers séquentiels.
- Le problème c’est la vulnérabilité si une des unités est endommagée.
- → Stockage selon table indexée: accès facilité car indexé mais problème du stockage des tables d’index (sur disque et dans RAM)

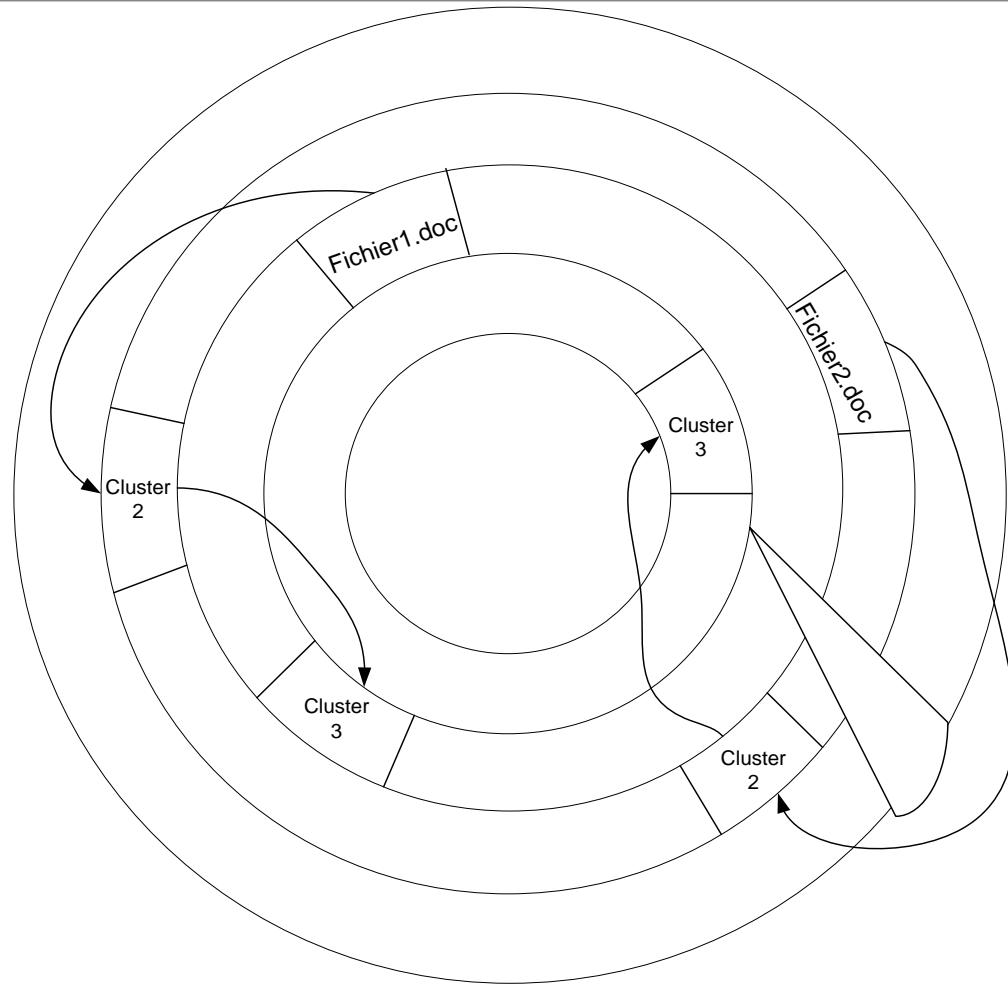
# Stockage contigu des fichiers

---

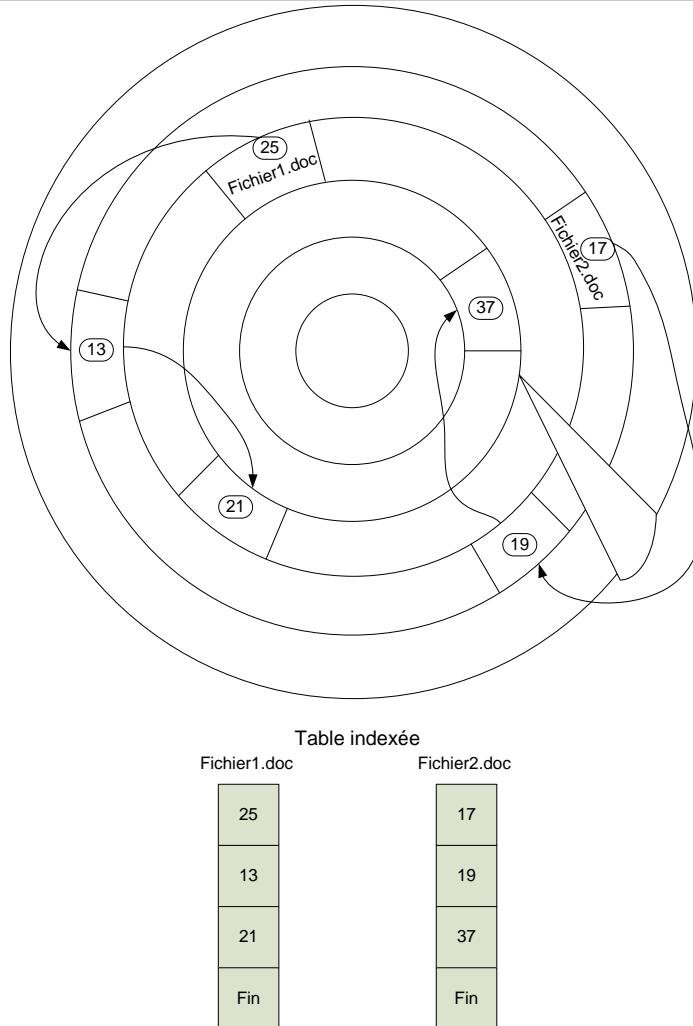


# Stockage des fichiers en liste liée

---



# Stockage selon table indexée

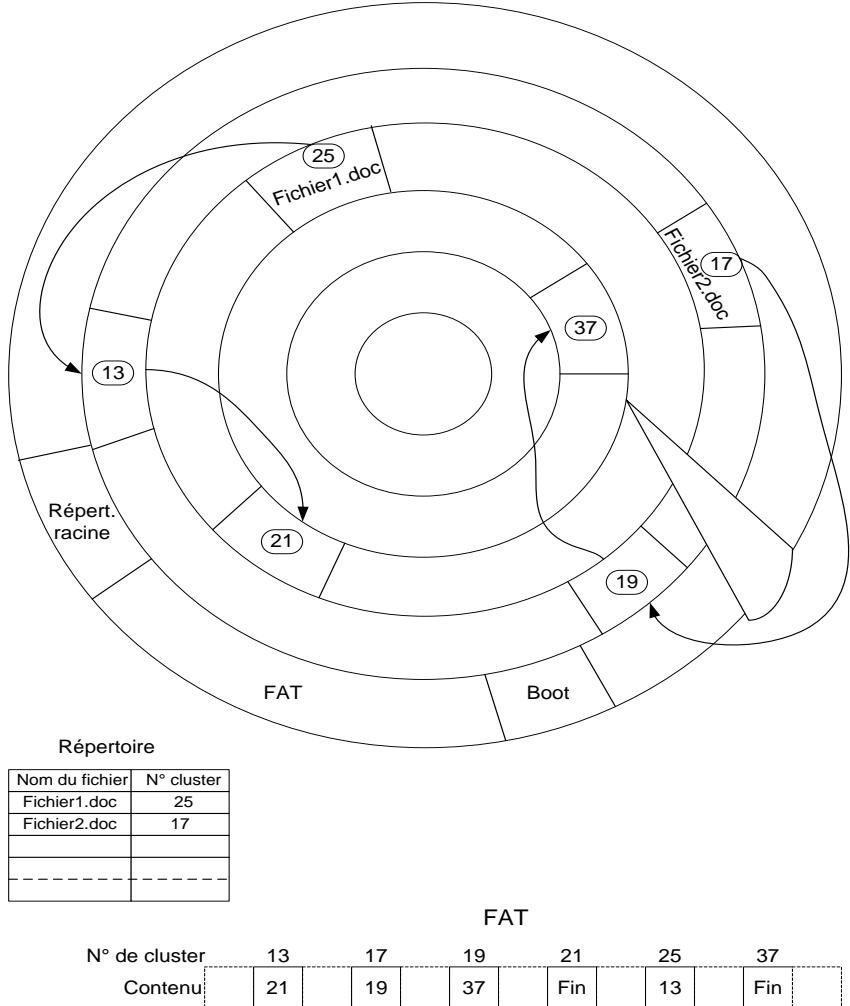


# Stockage logique des fichiers: FAT sous Windows (mélange indexé et liste liée)

---

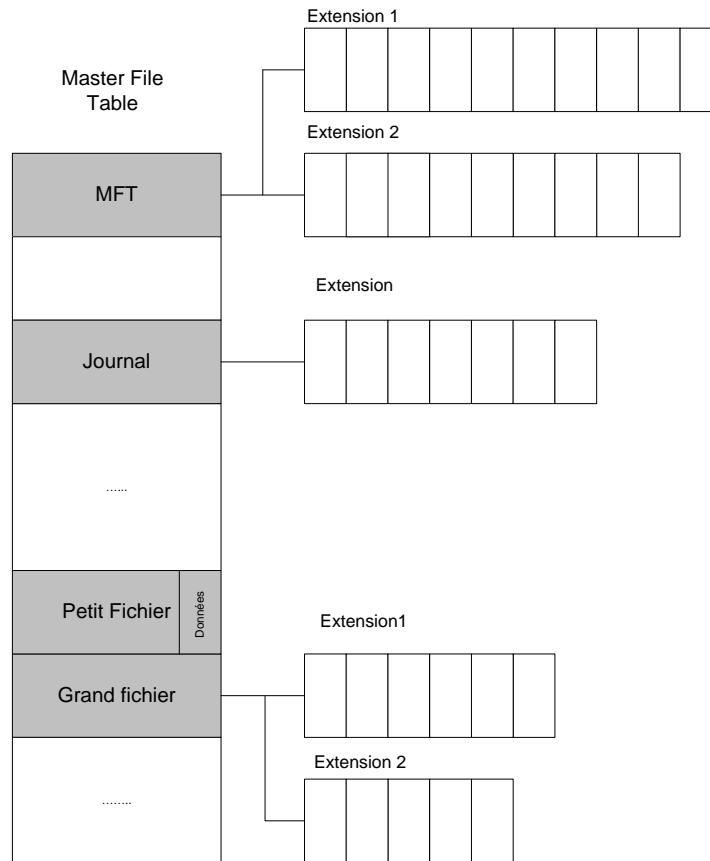
<b>Répertoire</b>		
<b>Champ</b>	<b>Signification</b>	<b>Exemple</b>
Nom du fichier	Le nom du fichier (limité à 8 octets)	Fichier1
Extension	Oriente vers une application	.doc (pour un fichier Word)
Date/heure	Mise à jour la plus récente	24 janvier 2004 17h20
Numéro cluster	Adresse du premier cluster du fichier	25
Nombre d'octets	Longueur du fichier	11200

<b>File Allocation Table – FAT</b>		
<b>Numéro d'entrée dans la FAT</b>	<b>Contenu</b>	<b>Remarque</b>
12	0	Cluster disponible
13	21	Adresse cluster suivant
	...	
21	Fin	Dernier cluster du fichier
25	13	Adresse cluster suivant (Fichier1.doc débute en 25)

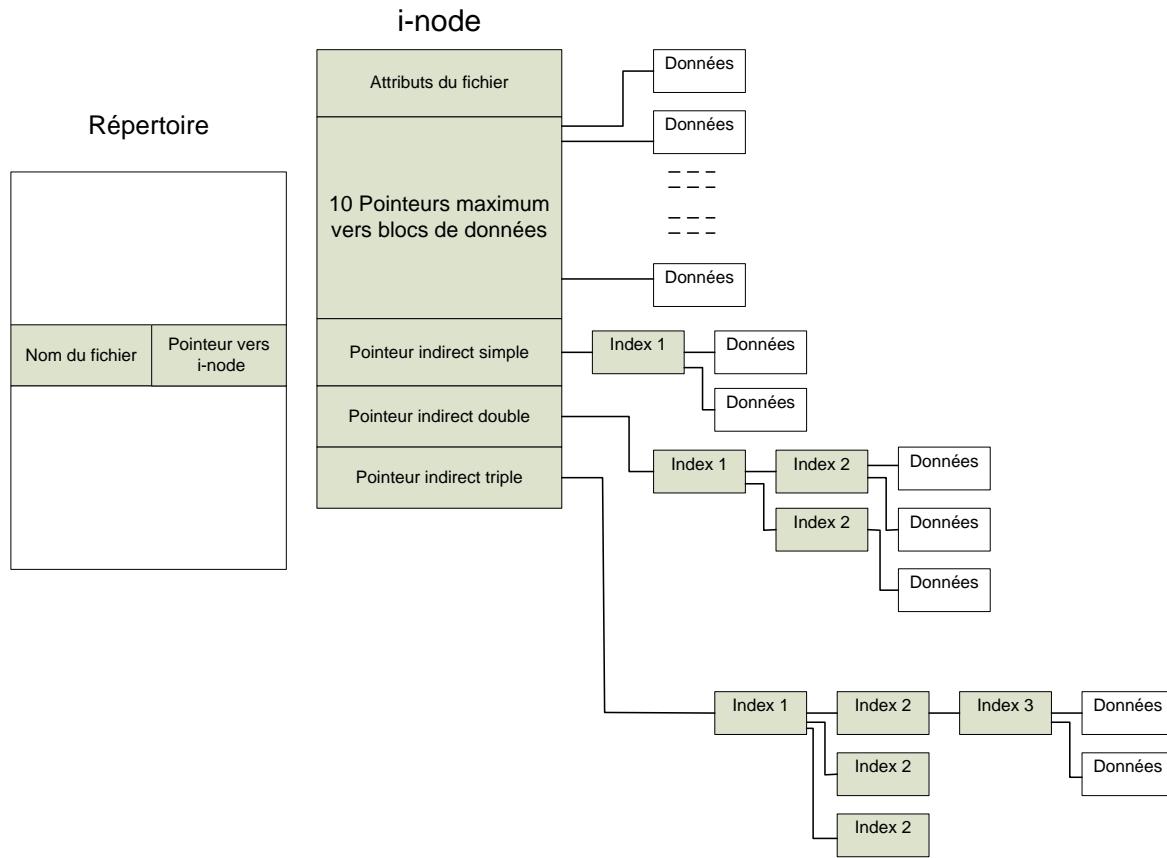


# NTFS sous Windows: plutôt indexé

## □ Depuis Windows NT

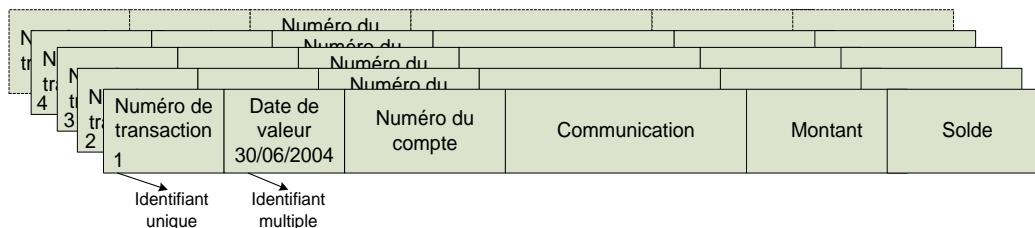
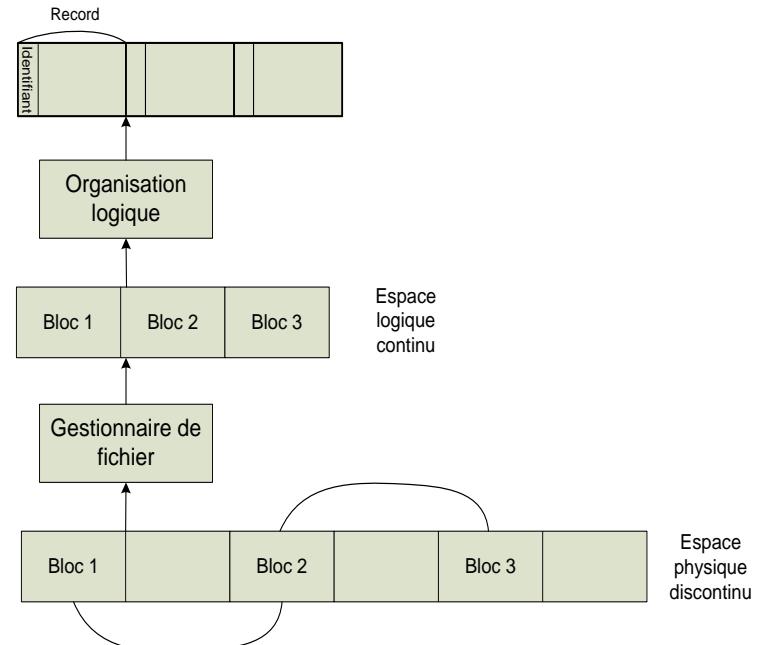


# Sous Unix (indexé NTFS = Unix)



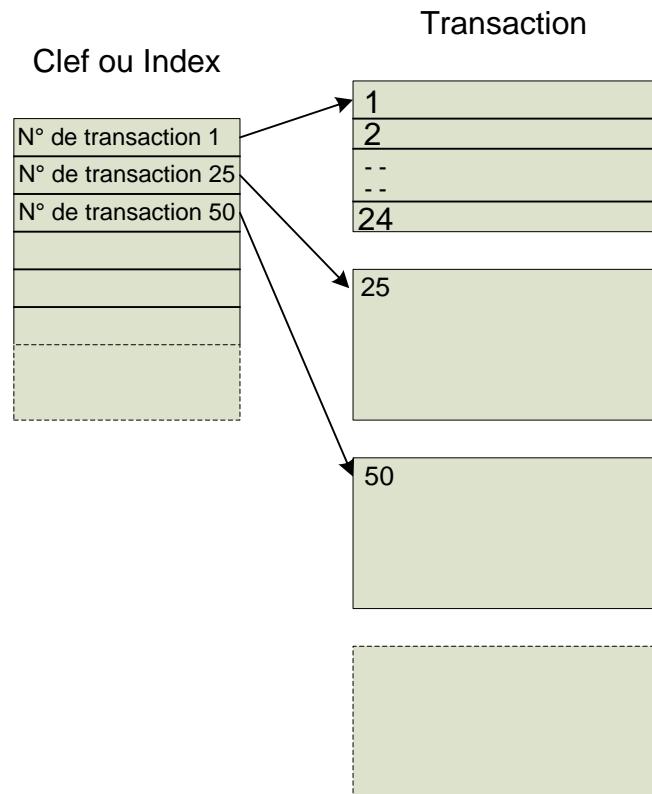
# Structuration logique des fichiers

- Un fichier = collection séquentielle d'octets
- Sur le disque: agrégats dispersés
- L'OS ne s'occupe pas de la structuration interne des fichiers.
- Or ce contenu peut avoir une organisation précise.
- Sous forme d'une suite de records.
- Chaque record possède son identifiant.
- Exemple des transactions



# D'où l'accès logique associé

- Accès séquentiel: seul possible pour les fichiers contigus ou en liste liée
- Le plus standard
- Parfait pour du texte mais pas pour des transactions
- → accès indexé: possible pour des fichiers stockés selon la table indexée.
- Table encombrante et parcours de la table à la recherche de la valeur de l'index très long.
- → accès direct par adresse calculée: adresse se calcule à partir de la valeur de la clef.
- → BASES DE DONNEES.



# LES BASES DE DONNEES

---

# Bases de données: quoi et pourquoi?

---

- Les données en entreprise doivent généralement être:
  - Structurées
  - Validée et sécurisées
  - Partagées et accessibles concurrentiellement
  - Sauvegardées de manière systématique
    - » Et l'historique des modifications conservé
  - Volumineuses
- Ex: Banque
  - Clients, Comptes, Opérations, etc.

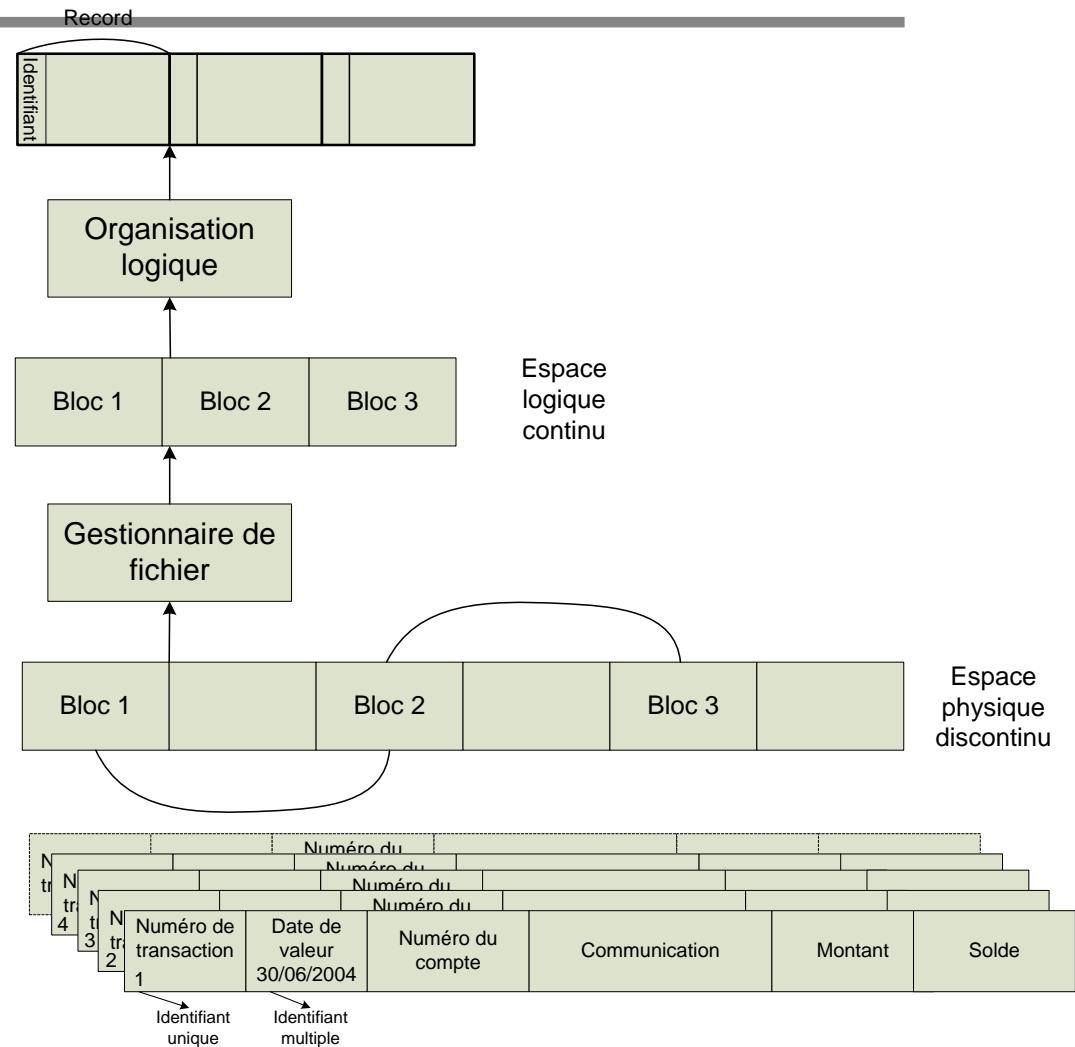
# Bases de données: quoi et pourquoi?

---

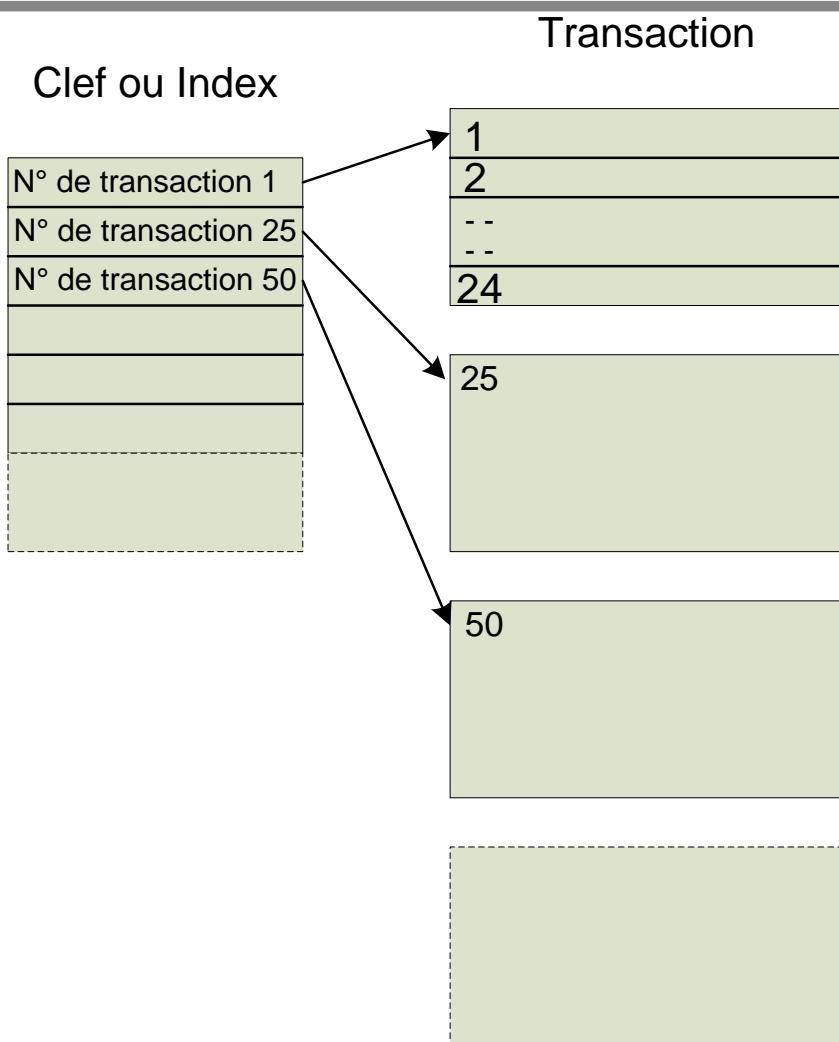
- Comment satisfaire ces différents besoins?
  - Système de fichiers (dont on reparlera) n'est pas la solution:
    - » Ne connaît ni ne gère la structuration interne du contenu des fichiers
    - » Ne permet pas d'accès logique en fonction du contenu
    - » Ne permet pas d'accès concurrentiels
    - » Ne permet pas de gérer des transactions et leur historique
    - » Ne permet de réaliser que des sauvegardes intégrales

# Structuration logique des fichiers

- Un fichier = collection séquentielle d'octets
- Sur le disque: agrégats dispersés
- L'OS ne s'occupe pas de la structuration interne des fichiers.
- Or ce contenu peut avoir une organisation précise.
- Sous forme d'une suite de records.
- Chaque record possède son identifiant.
- Exemple des opérations



# D'où l'accès logique associé



- → accès direct par adresse calculée:  
adresse se calcule à partir de la valeur de la clé
- → BASES DE DONNEES

# Bases de données: quoi et pourquoi?

---

## □ Définition:

- Une base de données est un lot d'informations stockées dans un dispositif informatique structuré
- Elle permet d'organiser et de structurer les données de manière à pouvoir facilement les manipuler et stocker efficacement de très grandes quantités d'informations
- L'organisation logique des données se fait selon un modèle de données
- La structure physique des fichiers comporte des index destinés à accélérer les opérations de recherche et de tri
- Les données sont gérées, consultées et modifiées par un langage de requêtes (SQL)
- Modèle le plus courant: Bases de données relationnelles

# Modélisation relationnelle

---

- Chaque table représente une **entité** (un groupe d'information conceptuel)
- Chaque colonne d'une table représente une composante de l'entité (**attributs**)
- Une ligne du tableau représente est un **tuple**
- Un attribut est repéré par un nom et un domaine de définition, c'est-à-dire l'ensemble des valeurs qu'il peut prendre (entier, booléen, chaîne de caractères).

Attributs

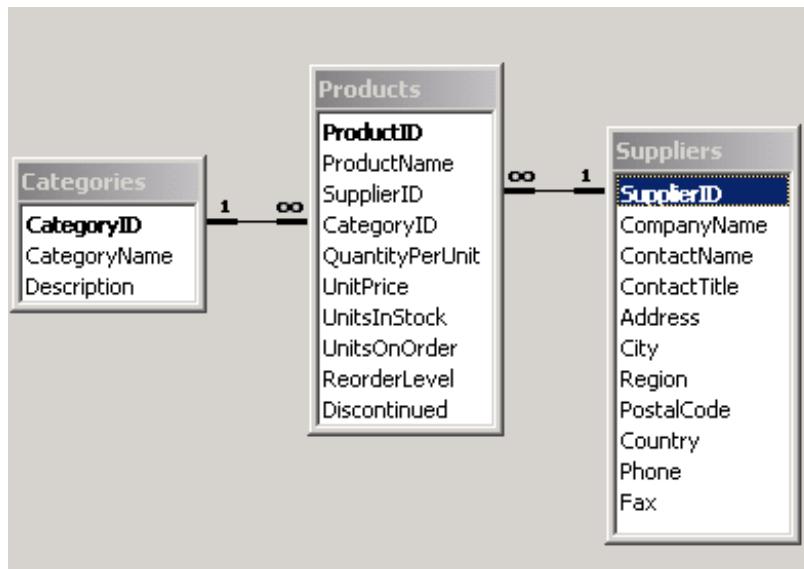
Marque	Modèle	Série	Numéro
Renault	18	RL	4698 SJ 45
Peugeot	309	Chorus	5647 ABY 82
Ford	Escort	Match	8562 EV 23

Tuples  
(N-uplets)

# Modèle de données

---

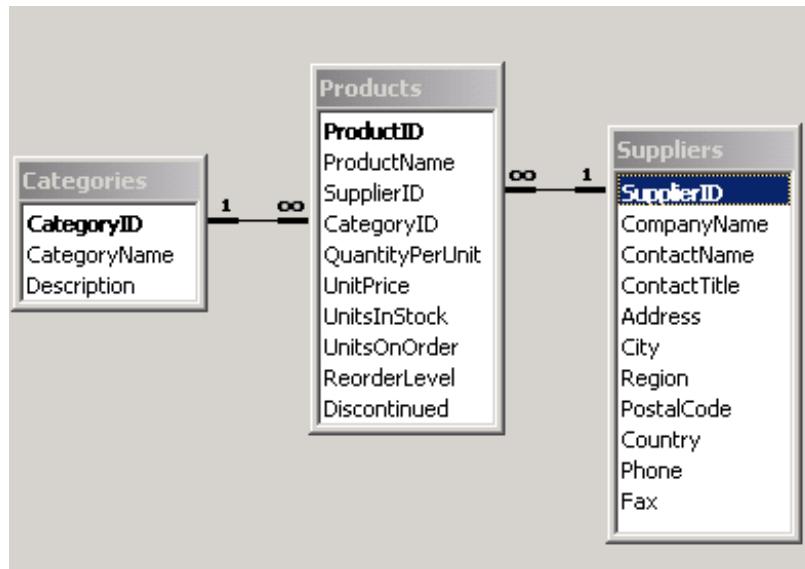
- schéma d'une entité : description d'une table par ses attributs (nom et domaine).
- schéma d'une base de données relationnelle : l'ensemble des tables et des relations qui la composent.



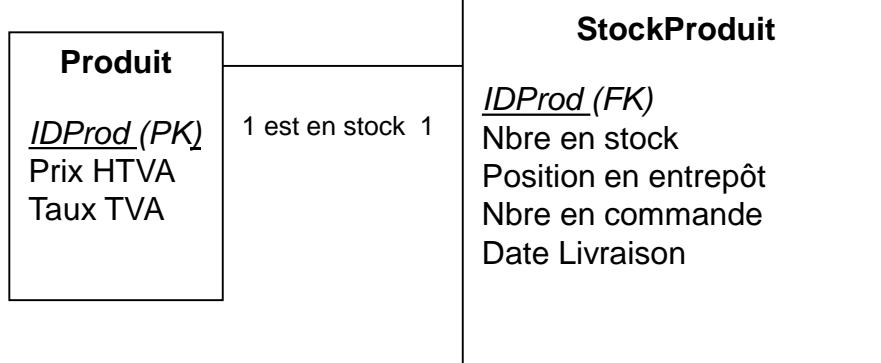
# Clé primaire et clé étrangère

---

- La clé primaire (PK) d'une relation est l'attribut, ou l'ensemble d'attributs, permettant de désigner de façon unique un tuple.
- Une clé étrangère (FK) est une clé faisant référence à une clé appartenant à une autre table.



# Relation 1-1



La table Produit représente l'ensemble des produits proposés par un magasin. Chaque ligne de cette table contient un élément IDProd (la clé primaire) permettant d'identifier ce produit de manière unique. Le produit est également caractérisé par un prix et un taux de TVA.

La table StockProduit représente l'état du stock pour chacun des produits figurant dans la table Produit. Chaque ligne de la table Produit est liée à une ligne de la table StockProduit, et réciproquement, nous avons donc une *relation* entre ces 2 tables.

Cette table ne possède pas de clé propre, elle utilise la même clé que la table Produit (on parle alors de *clé étrangère*). L'utilisation de ce mécanisme permet de créer une *relation* entre ces 2 tables.

# Relation 1-1: Fusion des entités conceptuelles

---

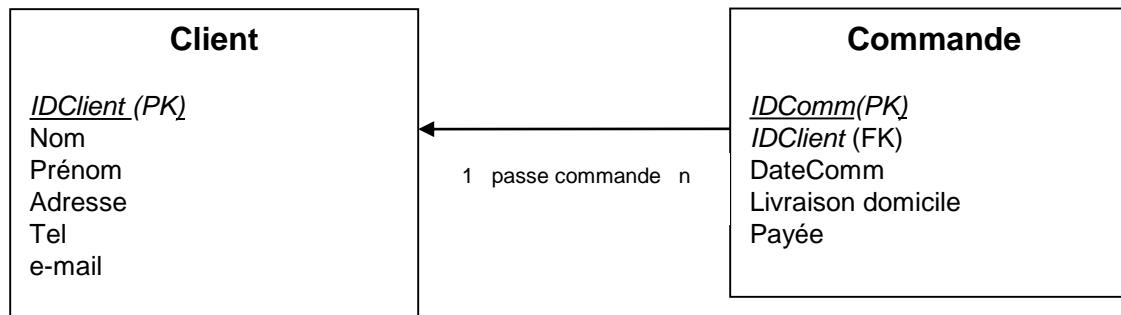
- Ces 2 tables sont des entités conceptuelles, elles représentent 2 concepts différents. Toutefois chaque élément d'une table étant lié à un et un seul élément de la seconde, on peut rassembler ces 2 tables en une dans notre base de données.

<b>ProduitEnStock</b>
<u>IDProd (PK)</u>
Prix HTVA
Taux TVA
Nbre en stock
Position en entrepôt
Nbre en commande
Date Livraison

# Relation 1-N

---

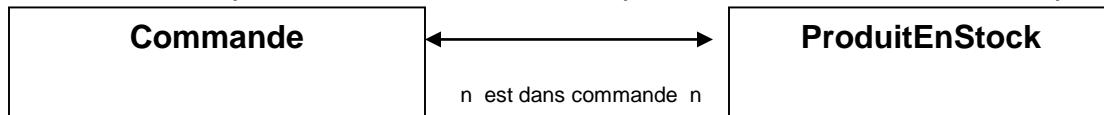
- Un client peut passer plusieurs commandes.
- Une commande est liée à un seul client
- On doit créer une relation entre Commande et Client.
- On place donc la clé étrangère IDComm dans la table Commande.
- Cette clé nous permet de retrouver le client associé à une commande.
- Plusieurs tuples de la table commande pourront avoir la même valeur pour l'attribut IDClient



# Relation N-N

---

- Une commande comporte un ou plusieurs produits.
- Un produit peut figurer dans plusieurs commandes.
- On doit créer une relation N-N entre Commande et ProduitEnStock.
- On place donc la clé étrangère IDComm dans la table Commande.
- Cette clé nous permet de retrouver le client associé à une commande.
- Plusieurs tuples de la table commande pourront avoir la même valeur pour l'attribut IDClient



IDComm	DateComm	Livraison	Payée
1	01/04/2002	O	O
2	08/04/2002	N	N

IDProd	Prix	NbreStock	IDComm
1	100	50	1
2	50	44	1
2	50	44	2

Le produit 2 figure dans 2 commandes et ça génère des redondances d'information dans la table produits.  
Ce modèle n'est pas adapté, on dit qu'il n'est pas *normalisé*.

# Relation N-N: Création d'une table de jointure

Nous allons transformer la relation N-N en 2 relations 1-N



<i>IDComm</i>	<i>DateComm</i>	<i>Livraison</i>	<i>Payée</i>
1	01/04/2002	O	O
2	08/04/2002	N	N

<i>IDComm</i>	<i>IDProd</i>
1	1
1	2
2	2

La table intermédiaire ne contient que les clés étrangères des 2 relations 1-N. Ceci permet de dupliquer les lignes sans induire de redondance dans le modèle.

<i>IDProd</i>	<i>Prix</i>	<i>NbreStock</i>
1	100	50
2	50	44
2	50	44

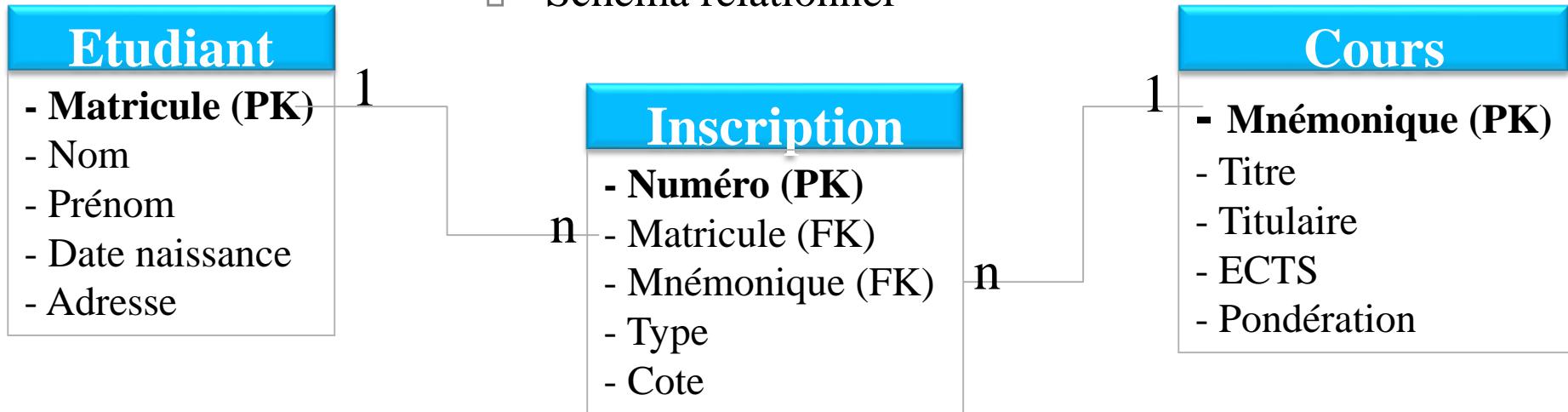
# Exemple synthétique

---

- Cotes d'étudiants aux cours
  - 3 Tables: Etudiants, Cours et Cotes
  - Etudiants:
    - » Clé primaire = Matricule
    - » Champs: Nom, prénom, adresse, date de naissance, etc.
  - Cours:
    - » Clé primaire = Mnémonique (code du cours)
    - » Champs: Titre, Titulaire(s), ECTS, heures, pondération, etc.
  - Cotes:
    - » Clé primaire = Numéro ad-hoc
    - » Champs: étudiant (FK), cours (FK), cote (obligatoirement entre 0 et 20), etc.

# Exemple synthétique: schéma relationnel

## □ Schéma relationnel



## □ Créons cette DB dans Access

# SQL, *Structured Query Language*

---

SQL (*Langage de requêtes structuré*) est

:

- un langage de définition de données
  - Création des tables et des relations
- un langage de manipulation de données
  - Consultation, insertion, modification de tuples
- un langage de protection de données
  - Définition des permissions au niveau des utilisateurs pour les bases de données relationnelles.

# Manipulation de données

---

## □ **Syntaxe de la commande SELECT**

*SELECT [ALL] | [DISTINCT]*

*<liste des noms de colonnes> | \**

*FROM <Liste des tables>*

*[WHERE <condition logique>]*

<b>Marque</b>	<b>Modele</b>	<b>Serie</b>	<b>Numero</b>
Renault	18	RL	4698 SJ 45
Renault	Kangoo	RL	4568 HD 16
Renault	Kangoo	RL	6576 VE 38
Peugeot	106	KID	7845 ZS 83
Peugeot	309	chorus	7647 ABY 82
Ford	Escort	Match	8562 EV 23

<b>Modele</b>	<b>Serie</b>
18	RL
Kangoo	RL
Kangoo	RL
106	KID
309	chorus
Escort	Match

SELECT Modele, Serie FROM VOITURES

<b>Modele</b>	<b>Serie</b>
18	RL
Kangoo	RL
106	KID
309	chorus
Escort	Match

# Expression booléenne

---

comparateurs arithmétiques

=  
!=  
>  
<  
>=  
<=  
<>  
!>  
!<

comparateurs de chaîne:

IN  
BETWEEN  
LIKE

opérateurs logiques

AND

OR

NOT

opérateurs arithmétiques:

+

-

\*

/

%

&

|

Marque	Modele	Serie	Numero	Compteur
Renault	18	RL	4698 SJ 45	123450
Renault	Kangoo	RL	4568 HD 16	56000
Renault	Kangoo	RL	6576 VE 38	12000
Peugeot	106	KID	7845 ZS 83	75600
Peugeot	309	chorus	7647 ABY 82	189500
Ford	Escort	Match	8562 EV 23	

*SELECT \* FROM OCCAZ  
WHERE (Compteur < 100000)*

Marque	Modele	Serie	Numero	Compteur
Renault	Kangoo	RL	4568 HD 16	56000
Renault	Kangoo	RL	6576 VE 38	12000
Peugeot	106	KID	7845 ZS 83	75600
Ford	Escort	Match	8562 EV 23	

Marque	Modele	Serie	Numero	Compteur
Renault	Kangoo	RL	4568 HD 16	56000
Peugeot	106	KID	7845 ZS 83	75600

*SELECT \* FROM OCCAZ  
WHERE (Compteur <= 100000) AND (Compteur  
>= 30000)*

# Le prédicat *LIKE*

---

- permet de faire des comparaisons sur des chaînes grâce à des caractères, appelés caractères *jokers*:
  - Le caractère **%** permet de remplacer une séquence de caractères (éventuellement nulle)
    - » Microsoft Access utilise l'astérisque (\*) plutôt que le %
  - Le caractère **\_** permet de remplacer un caractère

*SELECT \* FROM OCCAZ  
WHERE Marque LIKE "\_E%"*

Marque	Modele	Serie	Numero	Compteur
Renault	18	RL	4698 SJ 45	123450
Renault	Kangoo	RL	4568 HD 16	56000
Renault	Kangoo	RL	6576 VE 38	12000
Peugeot	106	KID	7845 ZS 83	75600
Peugeot	309	chorus	7647 ABY 82	189500

# Restriction sur les valeurs manquantes

---

- Lorsqu'un champ n'est pas renseigné, le SGBD lui attribue une valeur spéciale que l'on note *NULL*. La recherche de cette valeur ne peut pas se faire à l'aide des opérateurs standards, il faut utiliser les prédictats *IS NULL* ou bien *IS NOT NULL*.

*SELECT \* FROM OCCAZ  
WHERE Compteur IS NULL*

Marque	Modele	Serie	Numero	Compteur
Ford	Escort	Match	8562 EV 23	

# Tri des résultats

---

La clause *ORDER BY* est suivie des mots clés *ASC* ou *DESC*, qui précisent respectivement si le tri se fait de manière croissante (par défaut) ou décroissante.

*SELECT \* FROM VOITURE ORDER BY Marque ASC, Compteur DESC*

Marque	Modele	Serie	Numero	Compteur
Ford	Escort	Match	8562 EV 23	
Peugeot	309	chorus	7647 ABY 82	189500
Peugeot	106	KID	7845 ZS 83	75600
Renault	18	RL	4698 SJ 45	123450
Renault	Kangoo	RL	4568 HD 16	56000
Renault	Kangoo	RL	6576 VE 38	12000

# Traitements statistiques et regroupement des résultats

---

Utilisation de la clause *GROUP BY*, suivie du nom de chaque colonne sur laquelle on veut effectuer des regroupements (pour des traitements statistiques).

Les principales fonctions pouvant être effectuées par groupe sont:

- **AVG**: Calcule la moyenne d'une colonne
- **COUNT**: Calcule le nombre de lignes d'une table
- **MAX**: Calcule la valeur maximale d'une colonne
- **MIN**: Calcule la valeur minimale colonne
- **SUM**: Effectue la somme des valeurs d'une colonne

En combinant avec la clause GROUP BY, on peut faire un calcul sur chaque groupe produit

```
SELECT Marque, AVG(Compteur) AS Moyenne  
FROM VOITURE GROUP BY Marque
```

Marque	Moyenne
Renault	63816.6
Peugeot	132550
Ford	

# Jointures entre tables

---

- Une jointure est un produit cartésien de deux tables.
- Une équijointure est une jointure dont la qualification est une égalité entre deux colonnes.
- Exemple :  

```
SELECT T1.Att1, T2.Att2
FROM T1, T2
WHERE T1.ClePrim=T2.CleEtrang
```

Marque	Modele	Serie	Numero	Compteur
Renault	18	RL	4698 SJ 45	123450
Renault	Kangoo	RL	4568 HD 16	56000
Renault	Kangoo	RL	6576 VE 38	12000
Peugeot	106	KID	7845 ZS 83	75600
Peugeot	309	chorus	7647 ABY 82	189500
Ford	Escort	Match	8562 EV 23	
Fiat	Punto	GTI	8941 UD 61	
Audi	A4	Quattro	7846 AZS 75	21350

Nom	Pays
Renault	France
Fiat	Italie
Peugeot	France
Volkswagen	Allemagne
Ford	Etats-Unis

*SELECT Occaz.Marque, Occaz.Modele, Societe.Pays  
 FROM OCCAZ,SOCIETE  
 WHERE Occaz.Marque = Societe.Nom*

*requête équivalente avec les « alias » ...*

*SELECT O.Marque, O.Modele, S.Pays  
 FROM OCCAZ O,SOCIETE S  
 WHERE O.Marque = S.Nom*

Marque	Modele	Pays
Renault	18	France
Renault	Kangoo	France
Renault	Kangoo	France
Peugeot	106	France
Peugeot	309	France
Ford	Escort	Etats-Unis
Fiat	Punto	Italie

# Sous requêtes

---

Une sous-requête doit être placée à la suite d'une clause *WHERE* ou *HAVING*, et doit remplacer une constante ou un groupe de constantes qui permettraient en temps normal d'exprimer la qualification.

- lorsque la sous-requête remplace une constante utilisée avec des opérateurs classique, elle doit obligatoirement renvoyer une seule réponse (une table d'une ligne et une colonne).

Ex: *SELECT ---- FROM ---- WHERE ---- < (SELECT ---- FROM ----)*

- lorsque la sous-requête remplace une constante utilisée dans une expression mettant en jeu les opérateurs *IN*, *EXISTS*, *ALL* ou *ANY*, elle doit obligatoirement renvoyer une seule ligne.

Ex : *SELECT ---- FROM ---- WHERE ---- IN (SELECT ---- FROM ----)*

Marque	Modele	Serie	Numero	Compteur
Renault	18	RL	4698 SJ 45	123450
Renault	Kangoo	RL	4568 HD 16	56000
Renault	Kangoo	RL	6576 VE 38	12000
Peugeot	106	KID	7845 ZS 83	75600
Peugeot	309	chorus	7647 ABY 82	189500
Ford	Escort	Match	8562 EV 23	
Fiat	Punto	GTI	8941 UD 61	
Audi	A4	Quattro	7846 AZS 75	21350

Moyenne(Compteur) = 79650

*SELECT \* FROM OCCAZ*

*WHERE Compteur < (SELECT AVG(Compteur) FROM OCCAZ)*

NB : Les tuples dont le champ compteur est vide ne font pas partie du recordset retourné.

Marque	Modele	Serie	Numero	Compteur
Renault	Kangoo	RL	4568 HD 16	56000
Renault	Kangoo	RL	6576 VE 38	12000
Peugeot	106	KID	7845 ZS 83	75600
Audi	A4	Quattro	7846 AZS 75	21350

# L'opérateur UNION

---

- Cet opérateur permet d'effectuer une UNION des tuples sélectionnés par deux clauses *SELECT*
- Les deux tables sur lesquelles on travaille devant avoir le même schéma.

*SELECT ---- FROM ---- WHERE -----  
UNION  
SELECT ---- FROM ---- WHERE -----*

- Par défaut les doublons sont automatiquement éliminés. Pour conserver les doublons, il est possible d'utiliser une clause *UNION ALL*.

# Insertion de données

---

- L'insertion de nouvelles données dans une table se fait grâce à l'ordre **INSERT**, qui permet d'insérer de nouvelles lignes dans la table.  
L'ordre **INSERT** attend la clause *INTO*, suivie du nom de la table, ainsi que du nom de chacune des colonnes entre parenthèses (les colonnes omises prendront la valeur **NUL** par défaut).
- Les valeurs à insérer peuvent être précisées de deux façons:
  - **avec la clause *VALUES***
  - **avec la clause *SELECT***

# Insertion de données avec la clause *VALUES*:

---

une seule ligne est insérée, elle contient comme valeurs, l'ensemble des valeurs passées en paramètre dans la parenthèse qui suit la clause *VALUES*.

Les données sont affectées aux colonnes dans l'ordre dans lequel les colonnes ont été déclarées dans la clause *INTO*

```
INSERT INTO Nom_de_la_table  
(colonne1,colonne2,colonne3,...)  
VALUES (Valeur1,Valeur2,Valeur3,...)
```

Lorsque chaque colonne de la table est modifiée, l'énumération de l'ensemble des colonnes est facultatif

# Insertion de données avec la clause *SELECT*

---

Plusieurs lignes peuvent être insérées, elle contiennent comme valeurs, l'ensemble des valeurs découlant de la sélection. Les données sont affectées aux colonnes dans l'ordre dans lequel les colonnes ont été déclarées dans la clause *INTO*

```
INSERT INTO Nom_de_la_table(colonne1,colonne2,...)  
SELECT colonne1,colonne2,...  
FROM Nom_de_la_table2  
WHERE qualification
```

Lorsque l'on remplace un nom de colonne suivant la clause *SELECT* par une constante, sa valeur est affectée par défaut aux tuples.

*NB : Nom\_de\_la\_table* doit être différent de *Nom\_de\_la\_table2*

# Modification de données

---

La modification à effectuer est précisé après la clause *SET*. Il s'agit d'une affectation d'une valeur à une colonne grâce à l'opérateur = suivi d'une expression algèbrique, d'une constante ou du résultat provenant d'une clause *SELECT*.

*UPDATE Nom\_de\_la\_table  
SET Colonne = Valeur  
[WHERE qualification]*

La clause *WHERE* permet de préciser les tuples sur lesquels la mise à jour aura lieu

# Suppression de données

---

Grâce à l'ordre *DELETE* suivi de la clause *FROM*, précisant la table sur laquelle la suppression s'effectue, puis d'une clause *WHERE* qui décrit la qualification, c'est-à-dire l'ensemble des lignes qui seront supprimées.

*DELETE FROM Nom\_de\_la\_table  
WHERE qualification*

L'ordre *DELETE* est à utiliser avec précaution car l'opération de suppression est irréversible.

# SQL: Exemples

---

- Programmons quelques requêtes sur notre DB de cotes:
  - Extraction et affichage: SELECT
    - » Affichage du listing des étudiants
    - » Affichage du nom et prénom des étudiants <20 ans
      - <20 ans ou matricule >80000 et matricule<110000
    - » Affichage des étudiants dont le nom commence par ‘Ma’
    - » Affichage des étudiants triés par âge décroissant
    - » Affichage de la moyenne des cotes par cours
    - » Affichage du relevé de notes de chaque étudiant
    - » Affichage de la moyenne, min et max de chaque étudiant
    - » Affichage des cotes d’INFO-D202 inférieures à la moyenne du cours
    - » Affichage des 10 meilleurs et des 10 moins bons étudiants (classés par moyenne totale)
    - » Affichage des étudiants ayant une moyenne correspondant à une distinction (i.e. moyenne  $\geq 12$  et  $< 14$ )

# SQL: Exemples

---

- Programmons quelques requêtes sur notre DB de cotes:
  - Insertion: INSERT
    - » Insérer un nouveau cours dans la table des cours
    - » Insérer tous les étudiants dans la table cotes pour ce nouveau cours
  - Mise à jour: UPDATE
    - » Mettre 12 à tous les étudiants pour le nouveau cours
    - » Ajouter 1 point en informatique à tous les étudiants
    - » Remonter la cote d'informatique des étudiants en échec à 10
  - Suppression: DELETE
    - » Supprimer le cours ajouté
      - → Quel est le problème?

# Fonctionnalités des SGBD

---

- Fonctions de base
  - Gestion des tables et du système d'indexation
  - Intégration du langage de requêtes SQL
  - Interface graphique pour gérer les bases de données, créer des requêtes, etc.
  - Gestion des utilisateurs et droits d'accès fins
  - Gestion des transactions
  - Gestion des sauvegardes et de la réplication
- Outils de création d'applications intégrées (Access)
  - Formulaires, Rapports (Etats), Macros

# Quelques exemples de SGBD

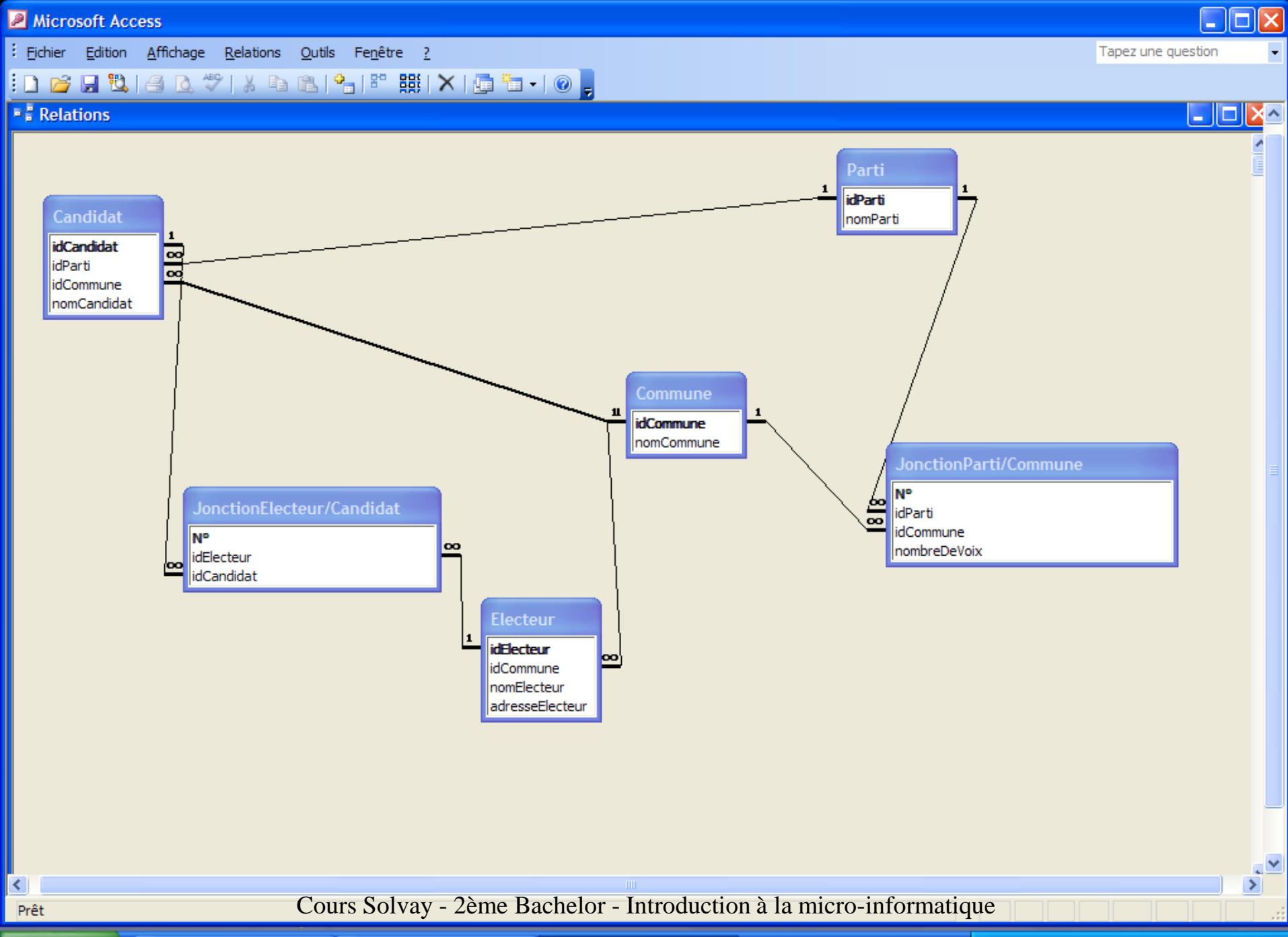
---

- Systèmes de Gestion de Bases de Données (SGBD)
  - Microsoft Access
  - MySQL & phpMyAdmin (open source)
  - Oracle
  - Microsoft SQL Server
  - IBM DB2

# Exercice 1

---

- Lors d'une élection communale, faisant fi de tout secret électoral, un informaticien malfaisant réussit à se procurer et à stocker dans une base de données relationnelle les informations suivantes : pour chaque commune (avec son nom et son nombre d'électeurs potentiels) les électeurs ayant voté (nom, prénom, adresse et heure du vote), les candidats (nom, prénom, position sur la liste du parti) pour lesquels ils ont voté (un électeur pouvant en effet voter pour plusieurs candidats) ainsi que le parti pour lequel ils ont voté (les candidats pour lesquels ils votent doivent être du même parti). Notre informaticien veut aisément pouvoir comptabiliser le nombre de voix attribuées à chaque parti et à chaque candidat pour chaque commune et aisément retrouver les électeurs ayant voté dans chaque commune pour chaque candidat et chaque parti. Réalisez les tables (avec leurs attributs) et le schéma relationnel décrivant la base de données en question et permettant à notre informaticien d'acquérir cette connaissance.



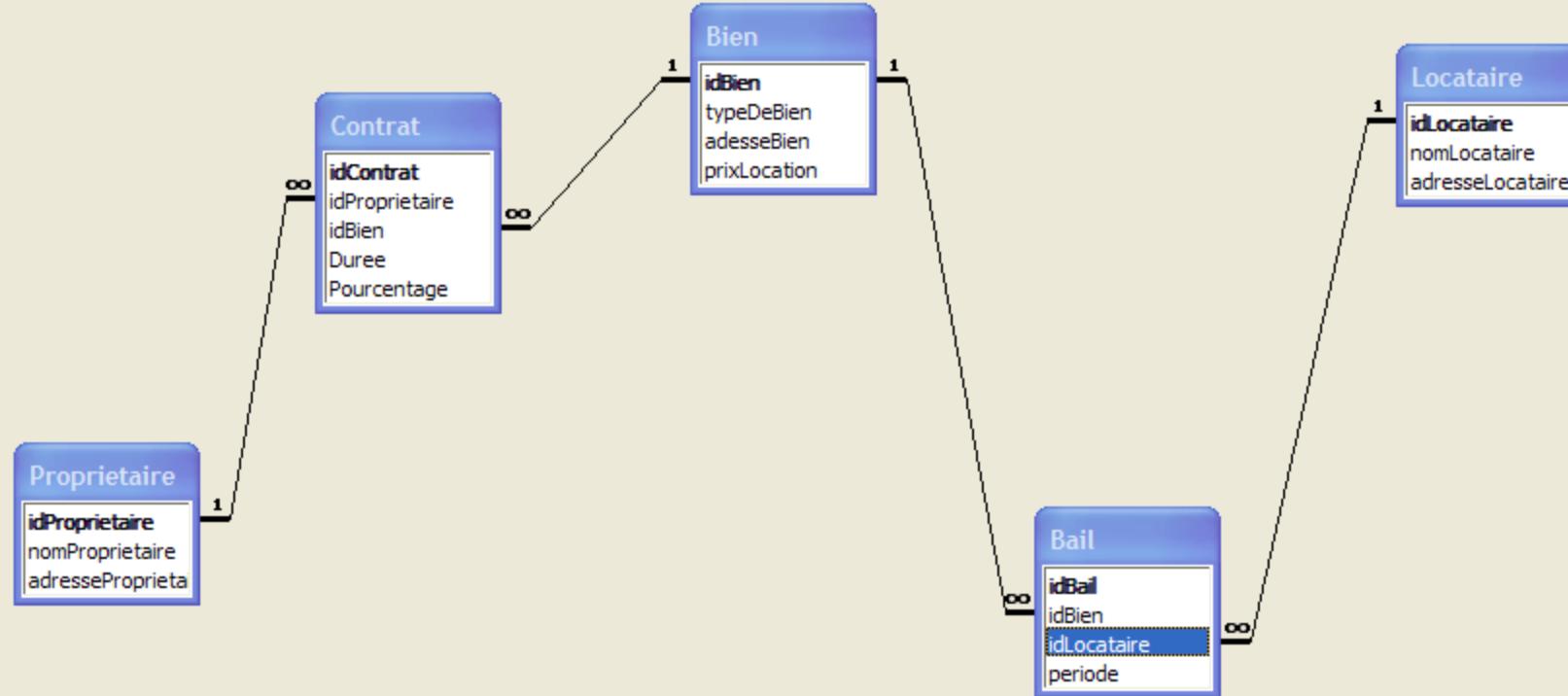
# Exercice 2

---

- Une agence immobilière gère la location d'appartements et de maisons pour le compte de propriétaires. Cette agence maintient une base de données relationnelles dans laquelle chaque propriétaire (nom, prénom, adresse, tel.) remet en gestion à l'agence plusieurs appartements ou maisons (taille, adresse, prix de location) selon un contrat (durée, pourcentage) propre à chaque bien. Chaque appartement ou maison est donnée en location par un bail identifié par un numéro et pour une période déterminée, à un locataire dont on enregistre le nom, prénom, adresse, tél. Réalisez le schéma relationnel d'une telle base de données.



tions



# L'interface utilisateur

---

- C'est ce qui doit rendre l'OS « user-friendly »
- **Gérer les applications:** ajouter ou retirer une application, démarrer une application, dialoguer avec l'application, passer des informations entre applications, par exemple par copier-coller.
- **Gérer ses répertoires et fichiers:** créer et effacer des répertoires, copier, sauvegarder, déplacer, effacer ses fichiers en utilisant des fonctions qui lui permettent de ne pas avoir à connaître la structure interne du contenu. Octroyer des droits d'accès différenciés (lecture, écriture...)
- **Gérer la configuration** matérielle ou logicielle du système: Procéder à la défragmentation du disque, gérer les supports des fichiers en lançant leur formatage ou leur découpe en partitions, installer une nouvelle version du système d'exploitation

- 
- l'interface peut être de type GUI ou CLI (commande par ligne). Attention le multi-fenêtrage n'implique pas le «vrai» multitâche. Le GUI est plus facile mais le CLI est moins coûteux, plus flexible et plus puissant (on peut programmer en CLI).
  - le GUI rend le réseaux très lourd car une application qui tourne sur un poste doit envoyer toutes les informations graphiques sur un autre. «X Window» tente de résoudre ce problème. Il sépare le programme qui produit l'image, du programme qui crée l'image et la montre sur l'écran. Un programme tournant sur un poste éloigné peut utiliser les facilités sur un poste local pour montrer les images. L'application peut dès lors s'exécuter en local ou sur un poste éloigné, ce qui ne changera rien pour l'utilisateur.
  - En UNIX, différentes fenêtres sur votre écran peuvent montrer des résultats produits sur différentes machines. Certains terminaux deviennent alors, de simples «terminaux X». X-Window est standard pour tous les OS et peut donc montrer une fenêtre de type «Windows» ou «Sun» ou «Mac-OS», etc...

# Les logiciels d'application

---

- Du plus étroit (un jeu) au plus large (SAS, Office, OS)
- I) Programme sur mesure: jeu, BLAST (pour le séquençage génétique)
- II) Progiciels:
  - Large domaine d'activités, gestion de production, compatibilité, administration du personnel, mais paramétrables: SAP
- III) Outils : à vocation universelle
  - Excel, Matlab, SAS.
  - Le navigateur

# Le rôle de l'administrateur système

---

- le superuser peut entrer dans le système en tant que tel, il a des droits d'accès privilégiés
- il vérifie le bon fonctionnement du système
- il a accès à tous les fichiers du système
- il peut ajouter un nouvel utilisateur avec son login, son mot de passe, son directory, son «login shell» et configurer son fichier d'init.
- Le fichier d'init est généralement configurable dans tous les système. Il comprend l'initialisation des «path», des «prompt», des périphériques, ...
- il peut partitionner le disque, le réparer, mesurer l'utilisation du disque, du CPU et l'espace mémoire.
- il peut faire des «backups» réguliers ou retrouver des données perdues
- vérifier et configurer la sécurité, les accès
- installer de nouveaux softwares et mettre à jour les softwares existant, y compris l'OS.
- installe les antivirus

# Virus et antivirus

---

- Le danger s'accroît avec "l'ouverture" de l'ordinateur (problèmes d'Internet)
- Le virus se rajoute à un programme hôte dont il détourne les instructions
- Trois parties:
  - L'infection responsable des dégâts
  - L'auto-réPLICATION recopiant le virus dans d'autres programmes hôtes
  - Marquer les fichiers déjà infectés
- Vers et cheval de troie
- Antivirus:
  - Faire correspondre les fichiers entrant avec des bouts de codes répertoriés et considérés comme nocifs.

# Confidentialité, intégrité, fiabilité

---

- Confidentialité des informations vous concernant → cryptographie (http → https)
- Intégrité: se prémunir d'une modification illégitime et nuisible de vos données → Processus d'authentification (mot de passe, biométrie), filtrer les accès (pare-feu)
- Fiabilité: attaque par « déni de service », spams et encombres intempestifs, la complexité nous échappe.

# Architectures de base

---

- Tout système logiciel est constitué d'au minimum 3 éléments:
  - Des données
  - Du code (=la logique applicative ou ‘programme’)
  - Une interface utilisateur, elle-même constituée
    - » D'un affichage
    - » D'un mécanisme de contrôle par l'utilisateur

# Architectures de base

---

## □ Notion d'architecture:

- Essentiellement la question de l'organisation de ces 3 éléments
  - » Sont-ils séparés ou forment-ils un tout?
  - » Sur quelles (et combien de) machines s'exécute chaque élément?
  - » Comment l'information circule-t-elle entre les différents composants (ou ‘tiers’)?

# Architectures de base

---

- Architectures principales en entreprise
  - Client simple
  - Client – Serveur (=2-tier)
    - » Classique (terminal)
    - » Data Server
  - 3-tier
  - Architecture web classique (multi-tier)
  - Architecture d'ensemble et intégration

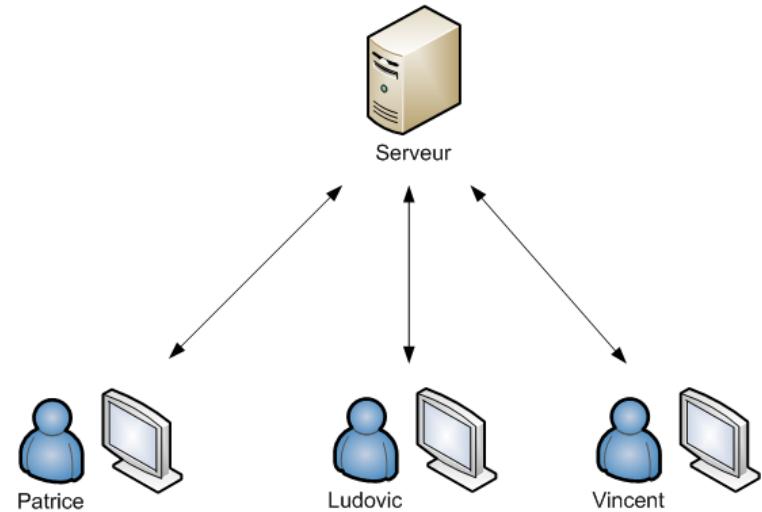
# Architectures de base

---

- Client simple: Logiciels personnels
  - Les logiciels personnels sont généralement de simples applications dites ‘client’
    - » Données, programme et interface utilisateur forment un tout
    - » Qui s’exécute intégralement sur un seul ordinateur
  - Exemples:
    - » Microsoft Word, Excel, Access
    - » Media Player, PhotoShop, Adobe Acrobat
    - » Jeux
    - » Etc.

# Architectures de base

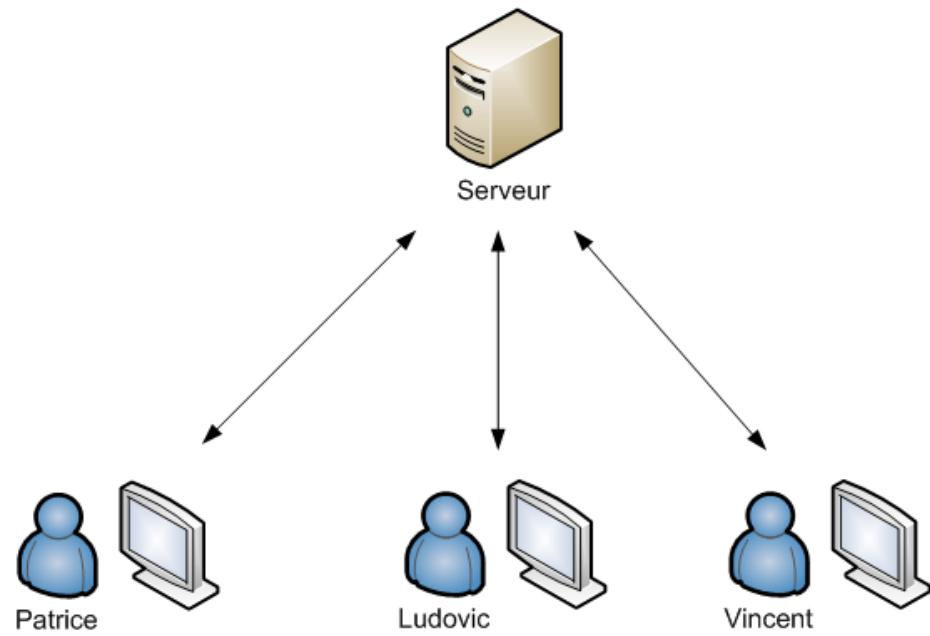
- Client – Server classique
  - Applications et données sur un serveur central
  - Terminaux (clients) ne possèdent qu'une interface et la connexion au serveur
  - Client envoie des requêtes au serveur qui renvoie le résultat



# Architectures de base

## □ Client – Data server

- Base de données sur un serveur central
- Clients exécutent la logique applicative et l'interface graphique + connexion au serveur
- Client envoie des requêtes au serveur de bases de données qui renvoie les données demandées
- Client traite et affiche les données



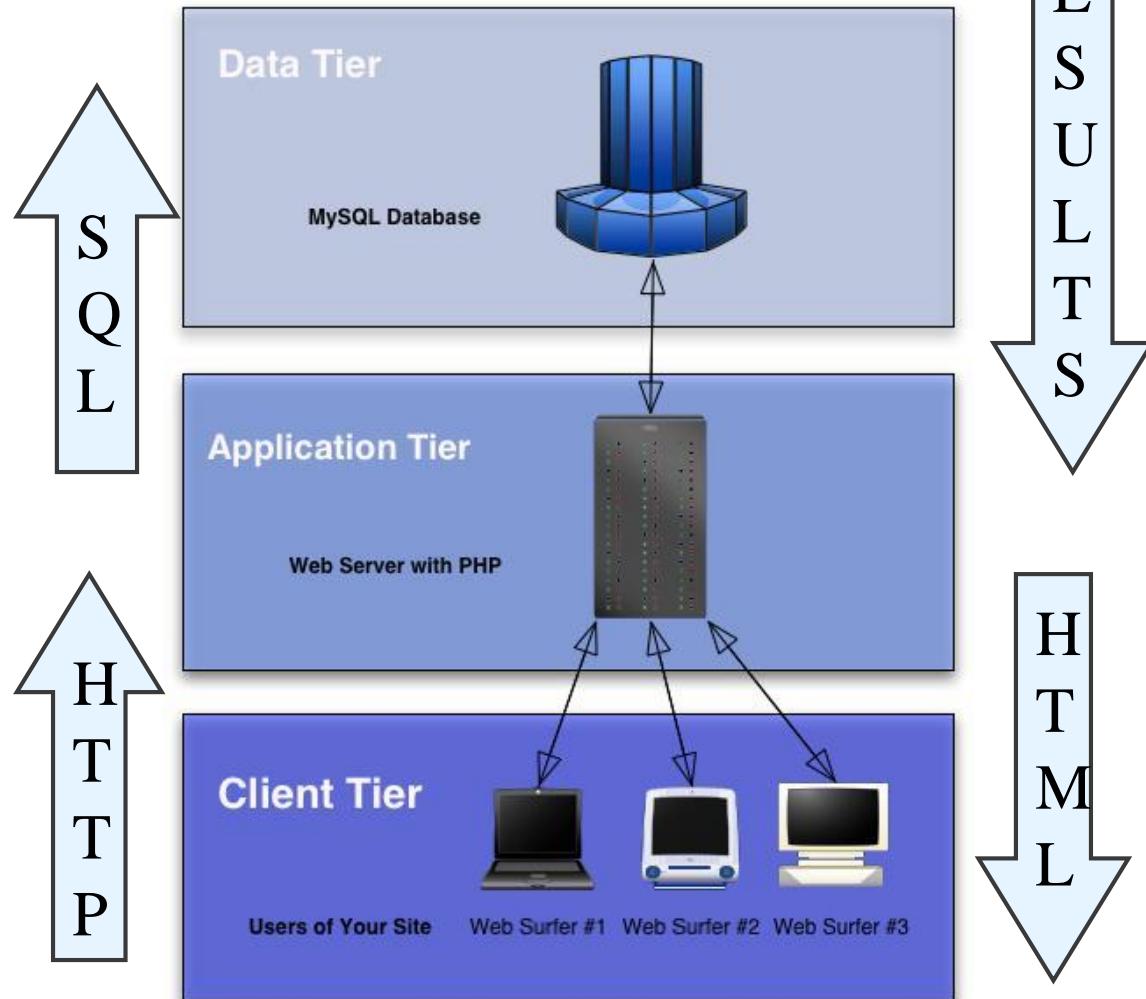
# Architectures de base

R  
E  
S  
U  
L  
T  
S

H  
T  
M  
L

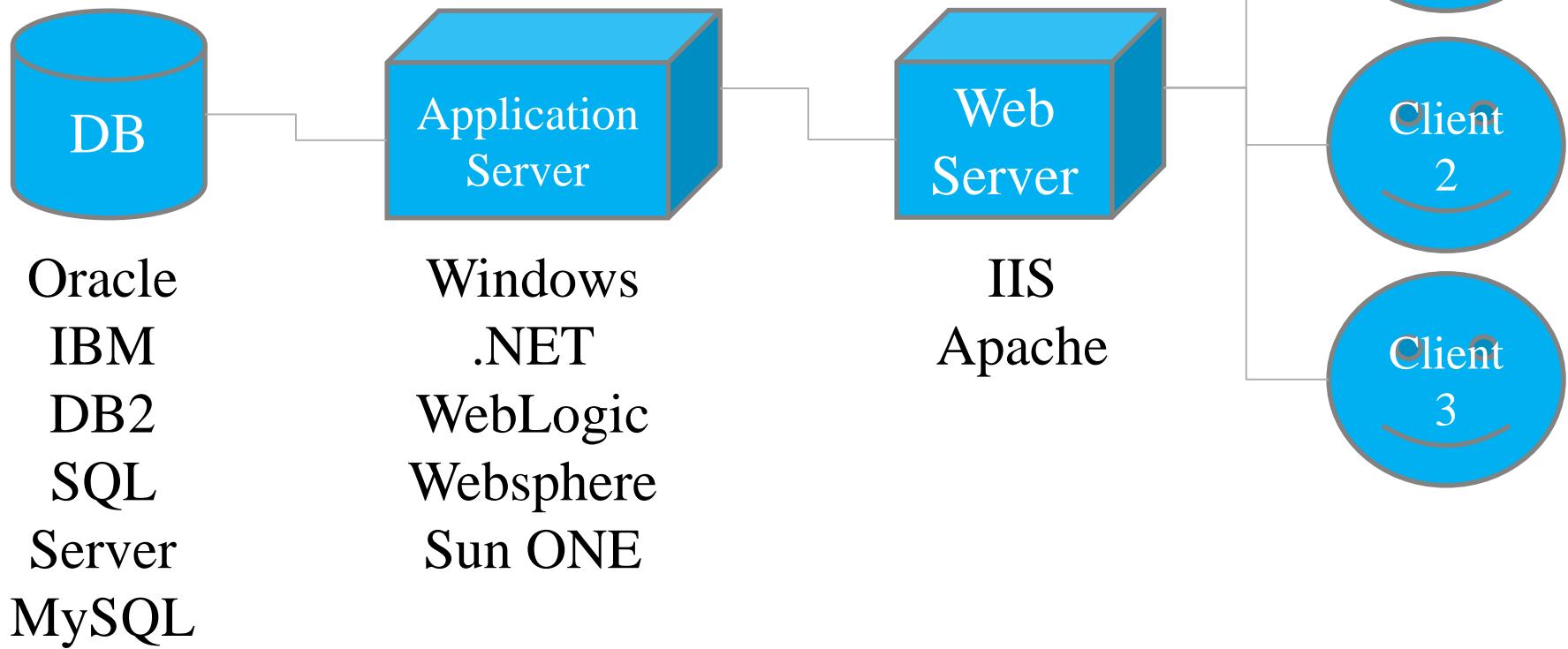
## □ 3-tiers

- Données  
(DB)
- Logique  
(Applications)
- Interface  
(Utilisateurs)
  - + Séparer  
Affichage et  
Contrôle  
(Paradigme MVC)

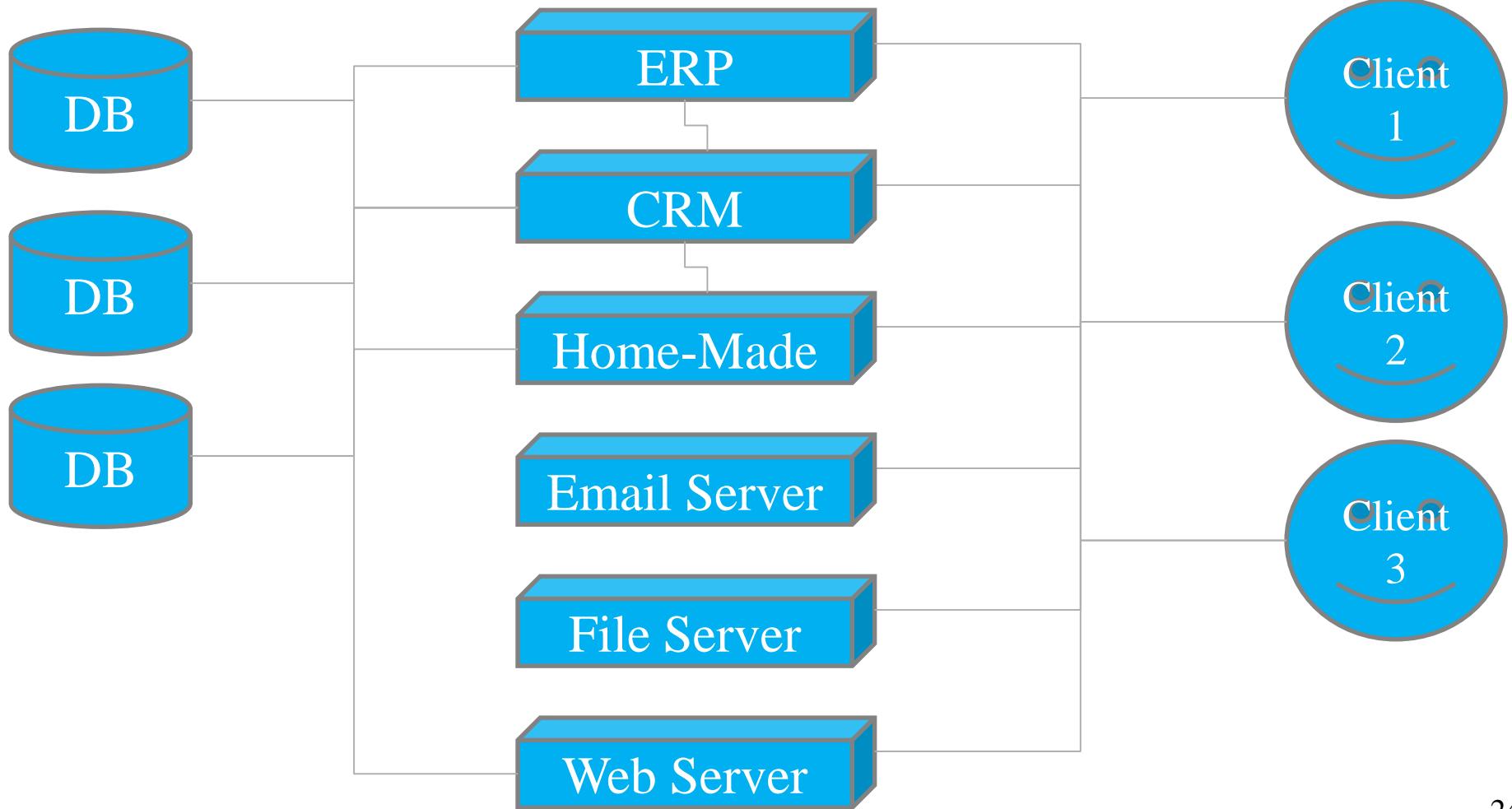


# Architectures de base

## □ Architecture Web classique

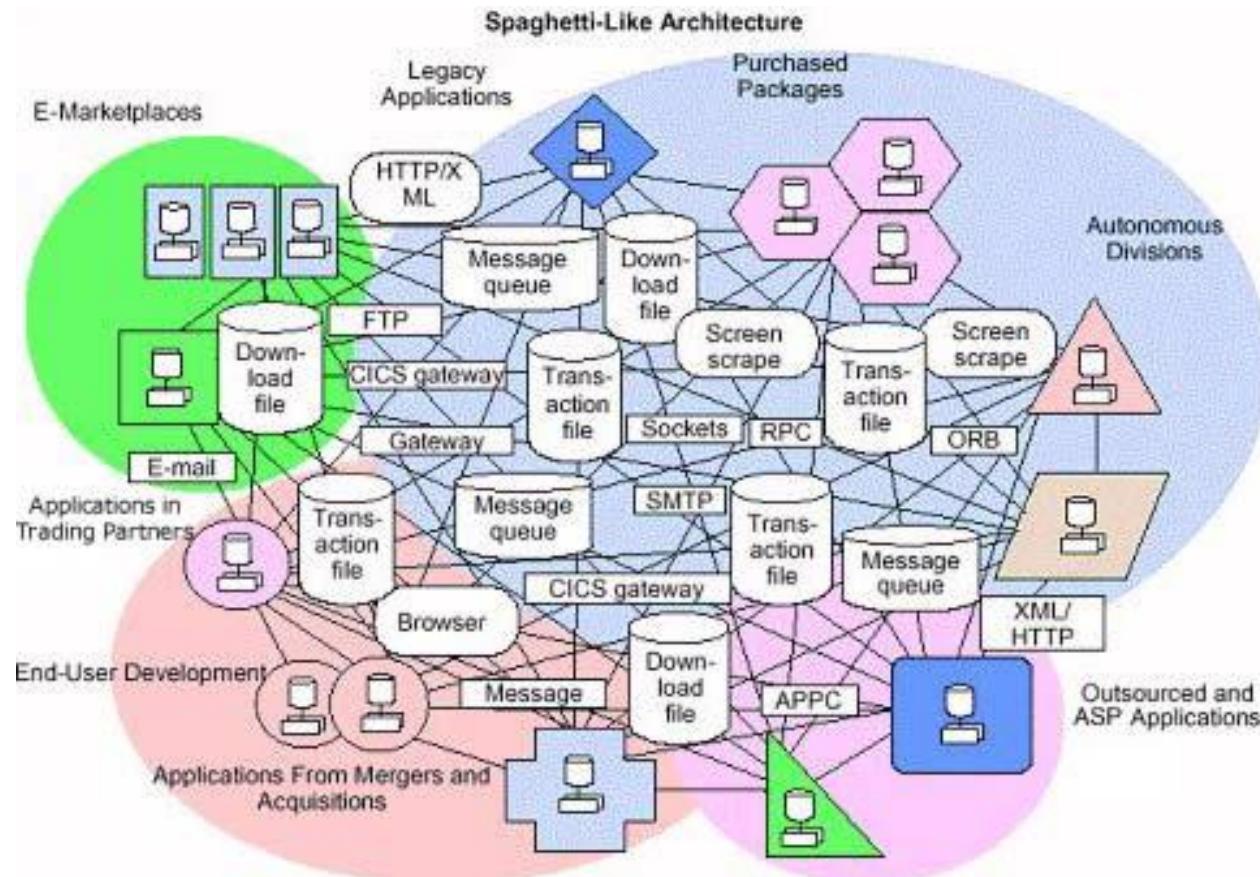


# Architectures de base: Simplified enterprise architecture



# Architectures de base

- Le risque: ‘Architecture Spaghetti’ → Risques?



# Architectures de base

---

- Risques liés aux architectures spaghetti
  - Coûts croissants de maintenance et de développement
    - » Complexité croissante
    - » Interdépendance des applications et des bases de données
  - Confusion des utilisateurs
  - Perte de temps en développement et en exploitation

# Architectures de base

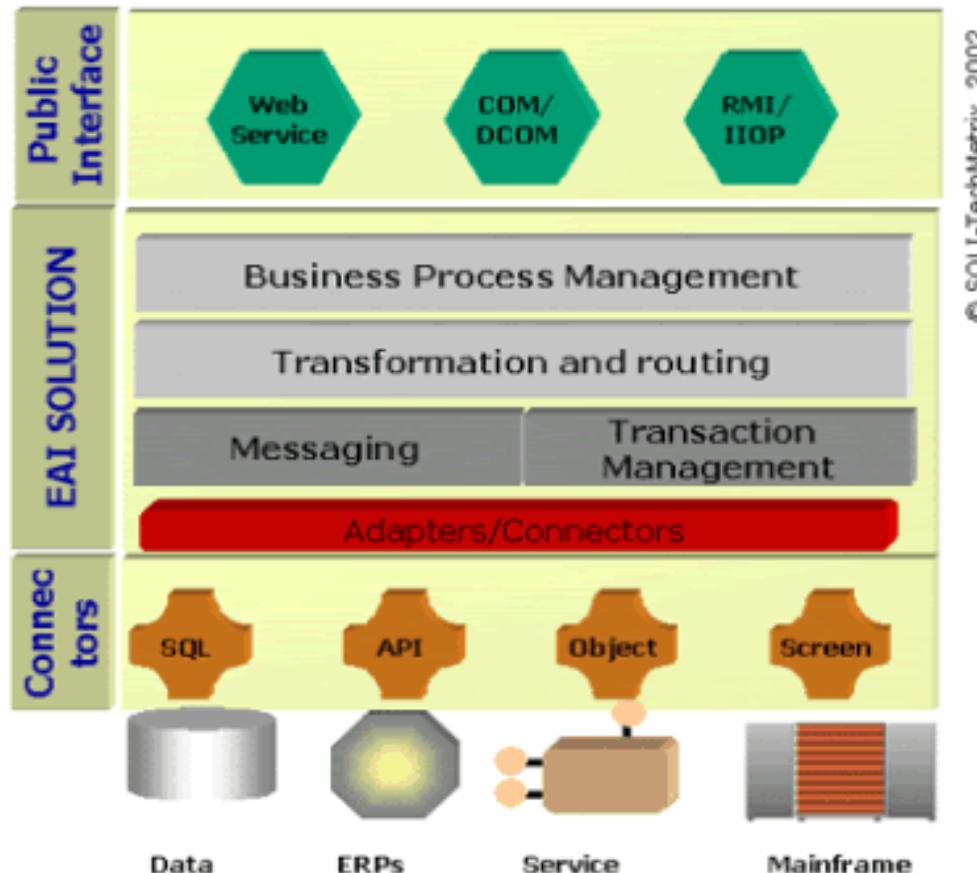
---

## □ Solutions au code spaghetti?

- Remettre l'architecture à plat et investir dans une ou plusieurs applications mieux conçues et mieux intégrées  
➔ Mais temps et argent?
- Séparer le ‘front-end’ du ‘back-office’ et ajouter une ou plusieurs couches d’abstraction qui neutralisent la complexité des applications back-office et présentent un ensemble de services intégrés au front-end (clients)
  - » Plusieurs solutions (complémentaires)
    - Enterprise Application Integration (EAI) et SOA
      - Microsoft Biztalk, IBM Websphere, BEA WebLogic, Oracle SOA Suite...
    - Portail d’entreprise (front-end intégré via interface web)
      - Microsoft SharePoint, IBM Websphere Portal, Plumtree, etc.
    - Business Intelligence Software
      - SAS, Cognos (IBM), Business Objects

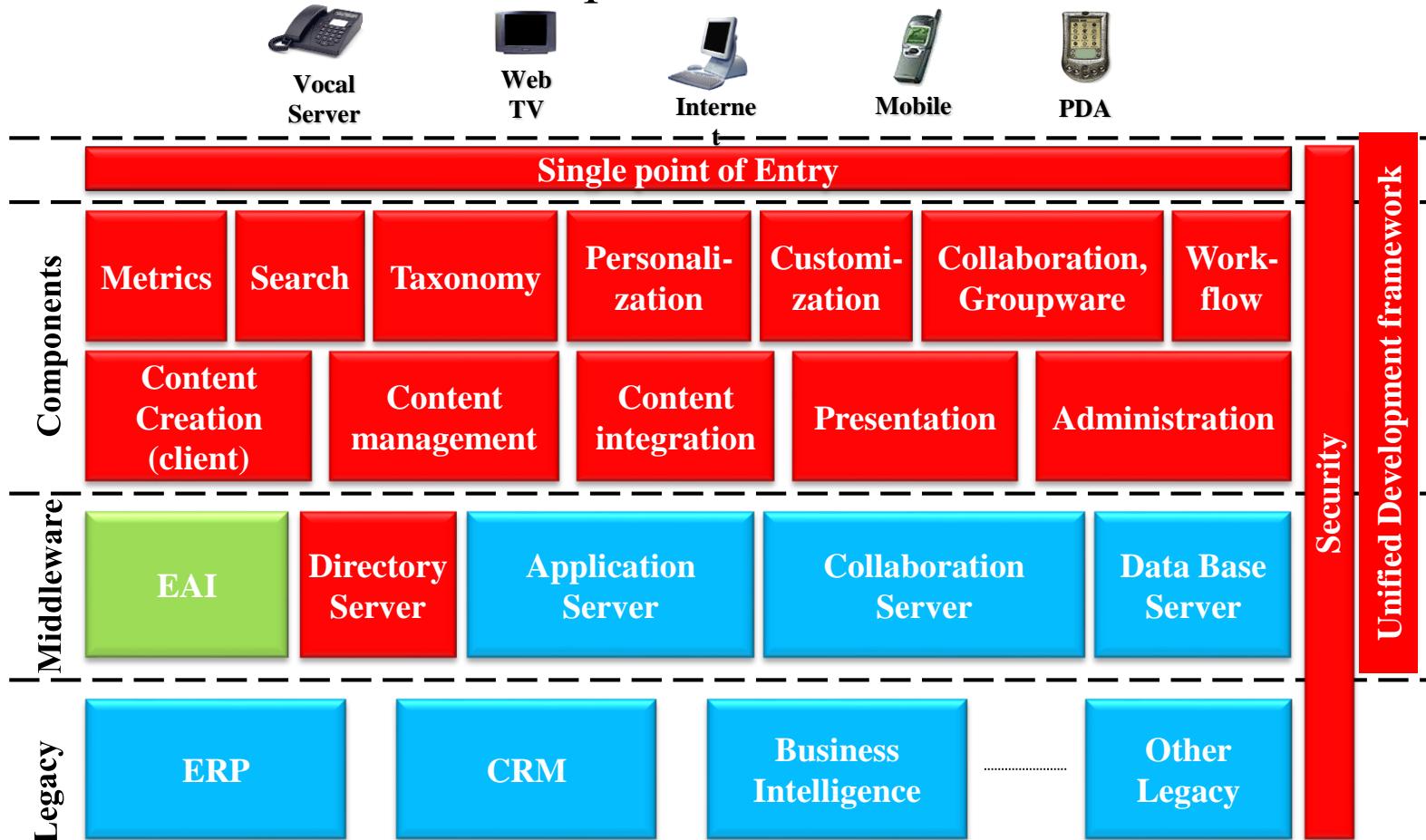
# Architectures de base

## □ Enterprise Application Integration



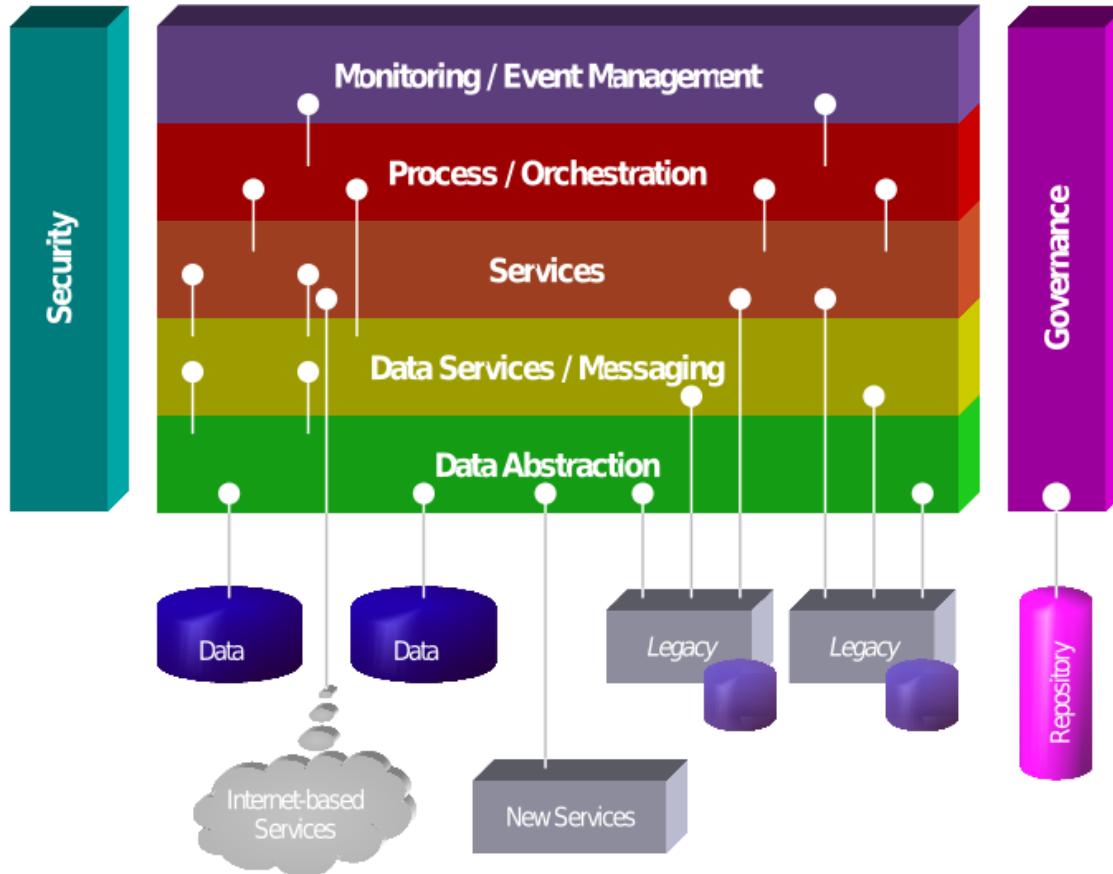
# Architectures de base

## □ EAI et Portail d'entreprise



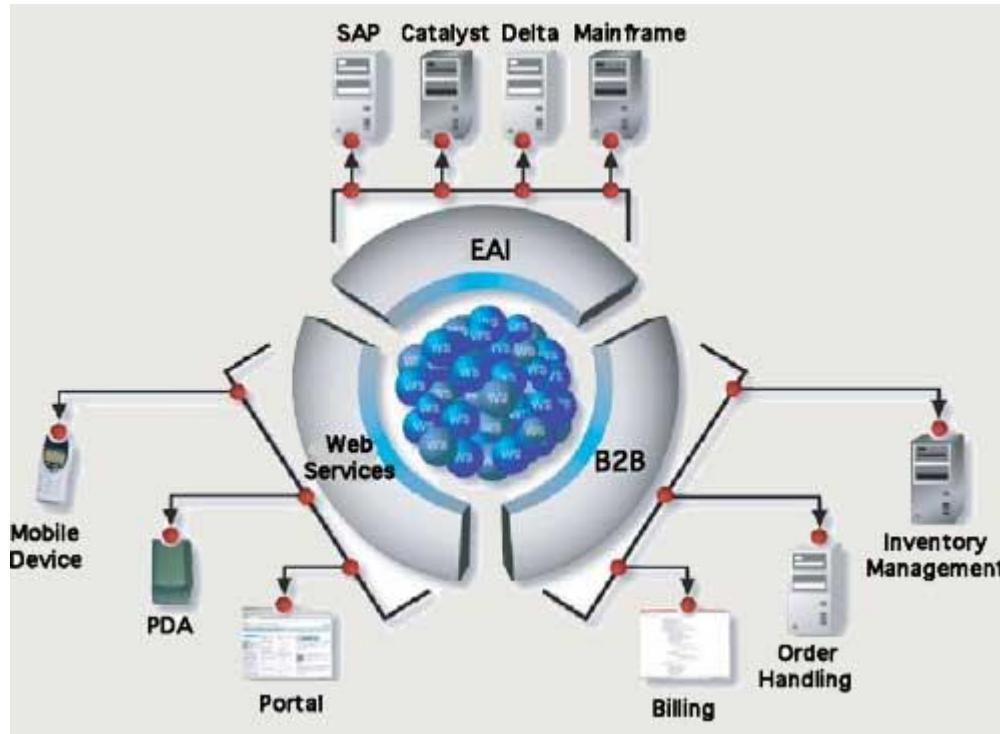
# Architectures de base

## □ Service-Oriented Architecture (SOA)



# Architectures de base

## □ Service-Oriented Architecture (SOA)



# Architectures de base

---

- Développement en web services
  - Développement en 5 couches
    - » Données
      - Dans des bases de données ou dans les applications back office
    - » Connecteurs
      - Adaptés aux différentes sources de données
    - » Services de base = fonctionnalités de base
      - Ex: Contrôler un utilisateur, modifier la quantité d'un produit dans une commande, etc.
    - » Business Processes = assemblage de services
      - Ex: Processus de création de commande
    - » Applications = interfaces d'accès aux processus
  - Avantages:
    - » Abstraction permet de se concentrer sur la conception des processus
    - » Implémentation des process facile et rapide par assemblage de services
    - » Services sont accessibles de l'intérieur comme de l'extérieur
      - Clients, fournisseurs, etc.

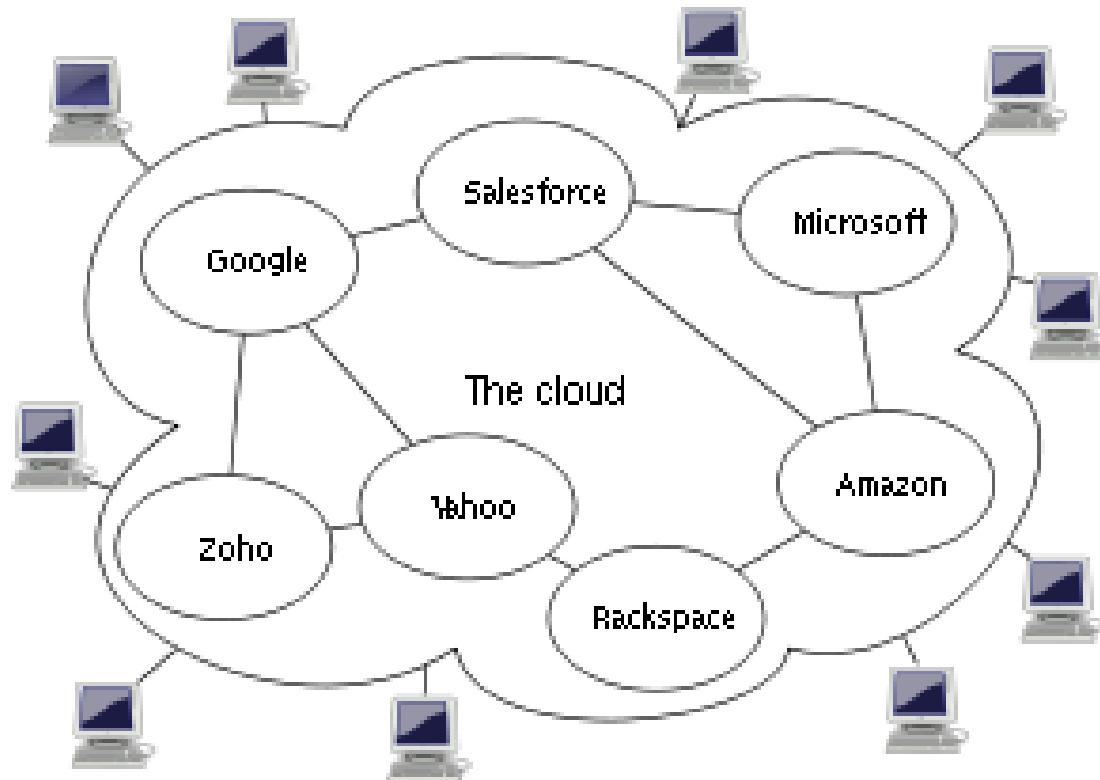
# Architectures de base

---

- Tendances actuelles
  - Mise en place de serveur d'intégration (EAI)
    - » Définition d'une taxonomie des données et mise en place de metadata (Harmonisation des champs de données)
    - » Ex: 'client\_zip\_code' = 'zipcode' dans SAP = 'customerZIP' dans Siebel
  - Implémentation de portails d'entreprise modulaires qui intègrent l'ensemble des fonctionnalités disponibles et offrent un accès personnalisé aux fonctionnalités pertinentes pour chaque utilisateur
  - Développement par modules ou assemblage de services (SOA)
  - Cloud computing: sharing of resources, software & information over the internet on demand, like the electricity grid
    - ➔ 'Software as a service' (SaaS)

# Architectures de base

- Tendances actuelles: cloud computing



# **VI. Les réseaux**

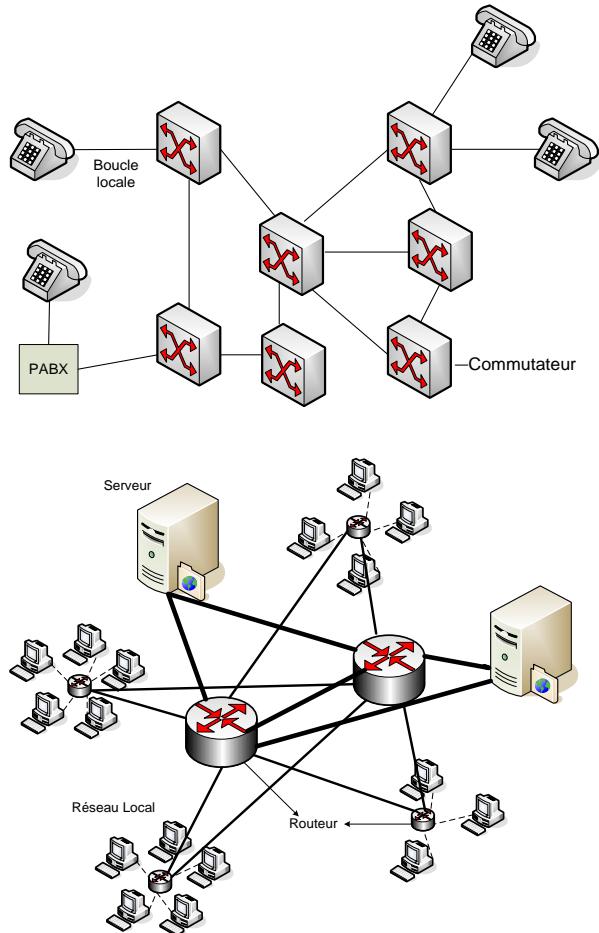
---

# Généralités: Pourquoi les réseaux

---

- partage des ressources: mémoires et CPU
- Informatique distribuée: Peer to peer, seti@home
- ASP (Application Service Provider)
- travail délocalisé
- fiabilité accrue: origine historique d'Internet (le projet Darpanet)
- économie: plusieurs petites machines en réseau est moins cher qu'une grosse machine
- communication: email, téléphone, vidéoconférence. La communication remplace les déplacements de personnes.
- bibliothèque virtuelle, globale et multimédia: le WEB + information personnalisée
- jeux, amusement,...

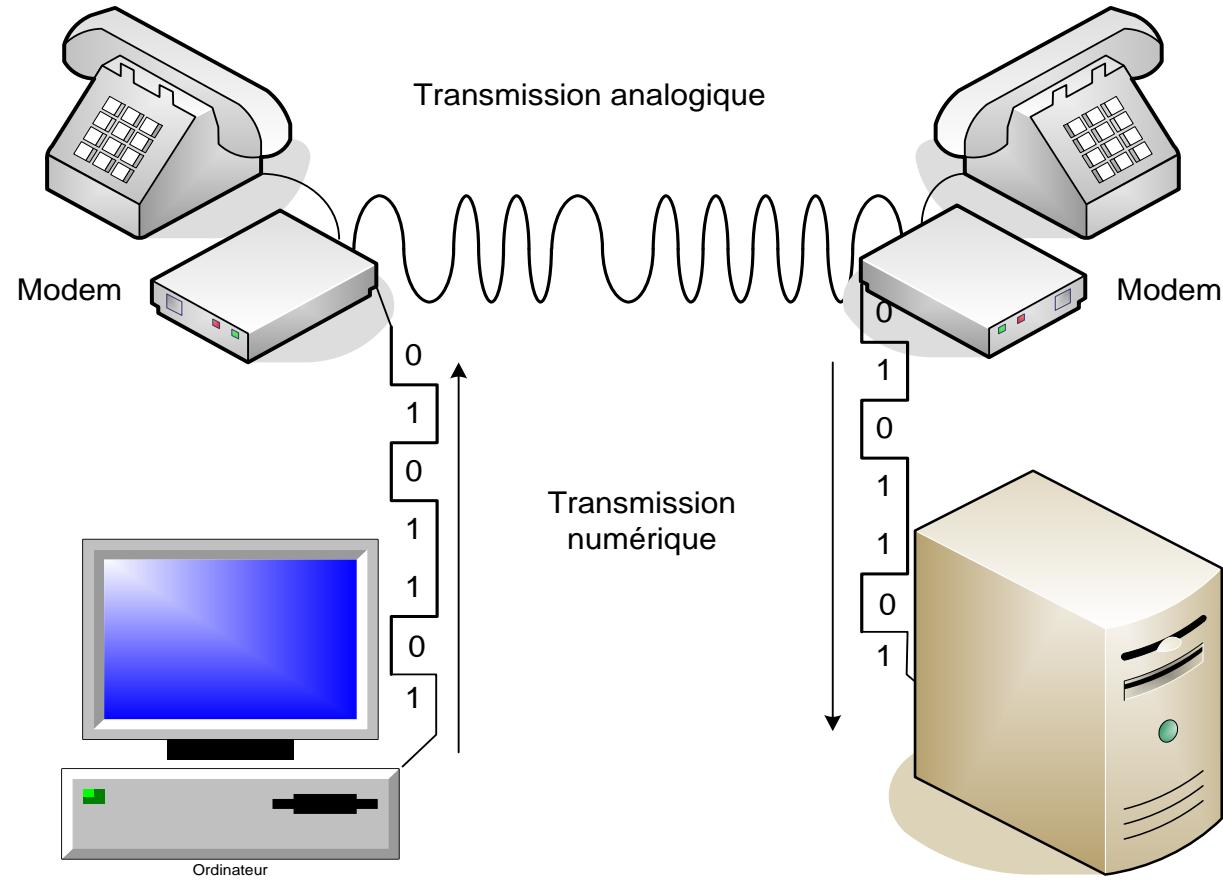
- Communication et dématérialisation de l'information
- Économie d'accès ou flux d'expérience
- Impact des communications téléphoniques
- Mais la voix, les films ou les fichiers n'ont pas les mêmes exigences.
- E\_commerce



# Types de support physique

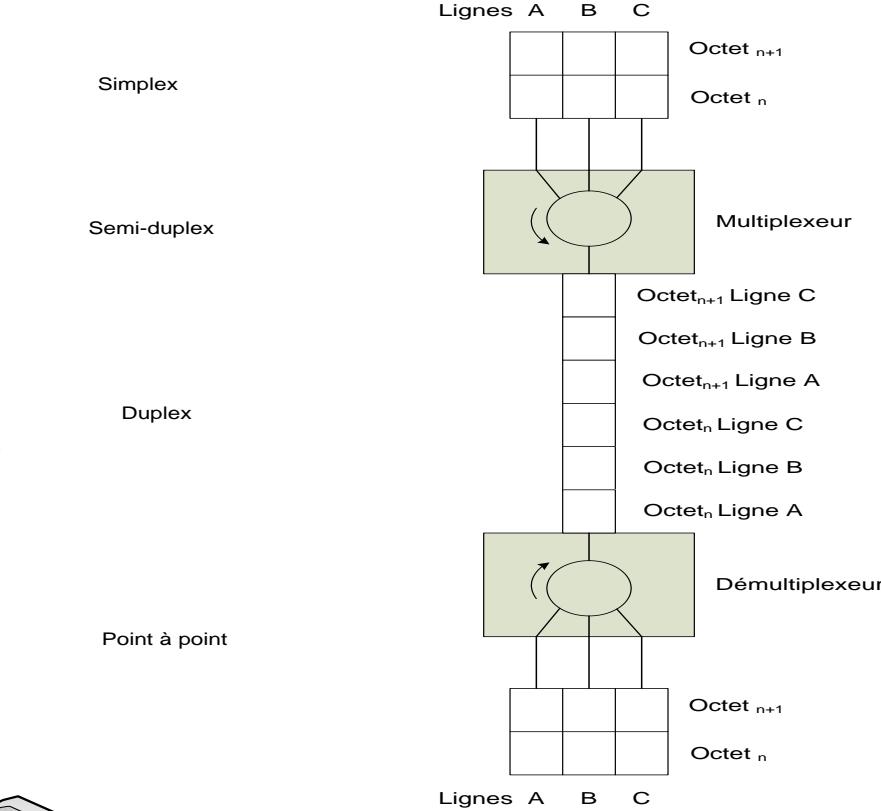
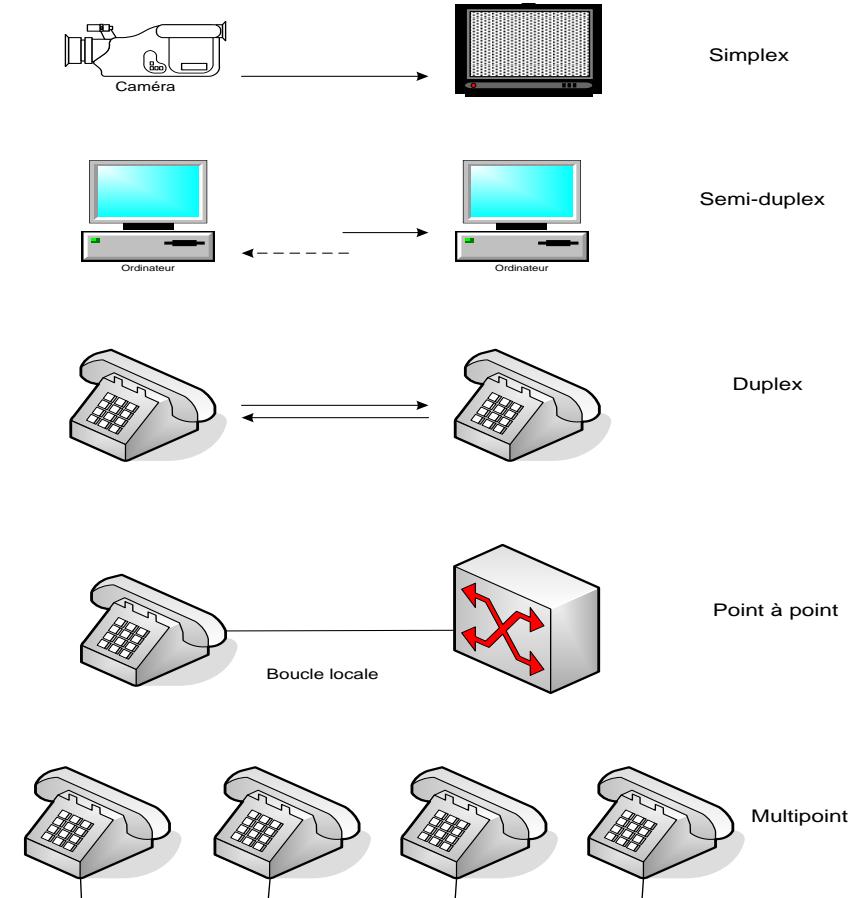
---

- la nature des données transmises: analogique ou digitale.
  - analogique: le signal varie de manière continue dans le temps
  - digital: le signal est discrétisé en binaire
  - la nature digitale ou analogique du signal dépend des besoins de la source et du destinataire. Les connexions téléphoniques se doivent d'être en partie analogique, toute connexion informatique peut être uniquement digitale.
  - il existe des systèmes de conversion analogique/digital comme les modems à connecter au téléphone (voir figure).
  - la tendance étant à la digitalisation généralisée de l'information, il s'en suit que les signaux à transmettre sont de plus en plus souvent uniquement digitaux.



- 
- La tendance est à la digitalisation (ou numérisation) généralisée de l'information tant pour son stockage que pour son transport. Une fois qu'elle s'est faite digitale et discrète, cette information est non seulement plus commode à stocker, à transporter, à multiplexer, à compresser ou encore à encrypter mais elle est encore plus fiable durant le transport, ayant moins à subir de l'effet des interférences.
  - Homogénéisation de toutes les informations
  - Les réseaux téléphoniques sont pour la plupart quasi entièrement numérisés au niveau de leur infrastructure. Seule la paire de fils (dite boucle locale) reliant le client à son commutateur de raccordement reste encore largement en modulation analogique, le remplacement du parc d'appareils téléphoniques analogiques par des appareils numériques sensiblement plus coûteux ne se justifiant que rarement.
  - Les conversations sont donc généralement acheminées en modulation analogique jusqu'au commutateur de raccordement, pour se poursuivre en modulation numérique jusqu'au commutateur final. Une conversion en modulation analogique y sera opérée pour envoi vers l'oreille du destinataire final.

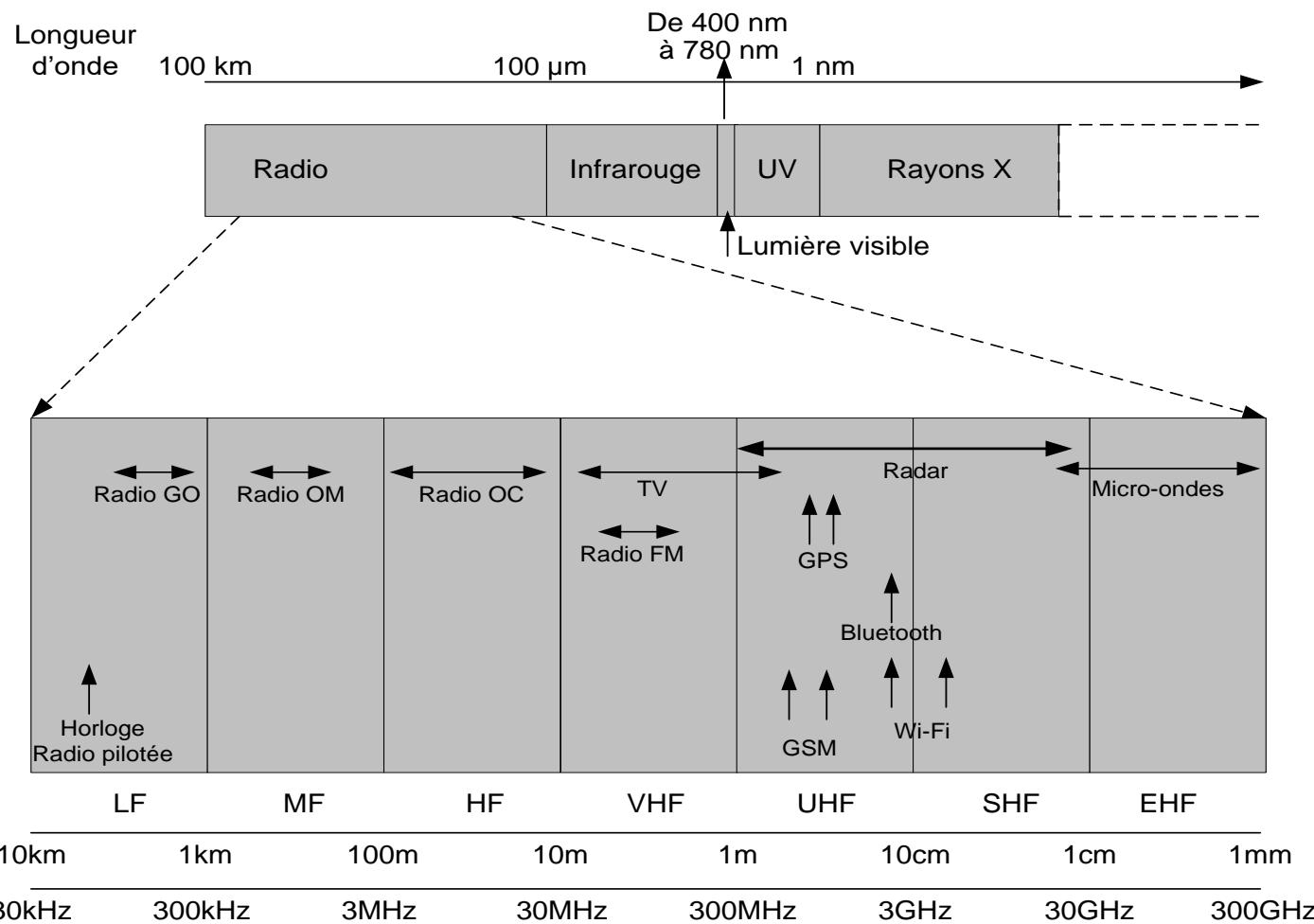
- 
- La directionalité du signal:
    - simplex: dans une direction - half-duplex: dans les deux direction mais une à la fois - full-duplex: dans les deux simultanément.
  - «multiplexer» les données: envoyer simultanément des données sur un même support. On peut multiplexer «en fréquence» et en «temps». L'envoi de plusieurs canaux TV est un «multiplexage» en fréquence. On divise la bande passante en n fréquences. Mais dans la plupart des cas (surtout dans la transmission digitale), on «multiplex» dans le temps en attribuant des quantum de temps à chacun des signaux.
  - la transmission peut être synchrone, à un rythme régulier, mais de plus en plus souvent, on permet des transmissions asynchrones qui exigent des «bits» de début et de «fin».
  - chaque signal peut être bruité ou atténué, il faut donc renforcer (ou «répéter») le signal lors de son transfert et également prévoir des détecteur et des correcteurs de signaux binaires (comme pour les échanges avec le CPU)

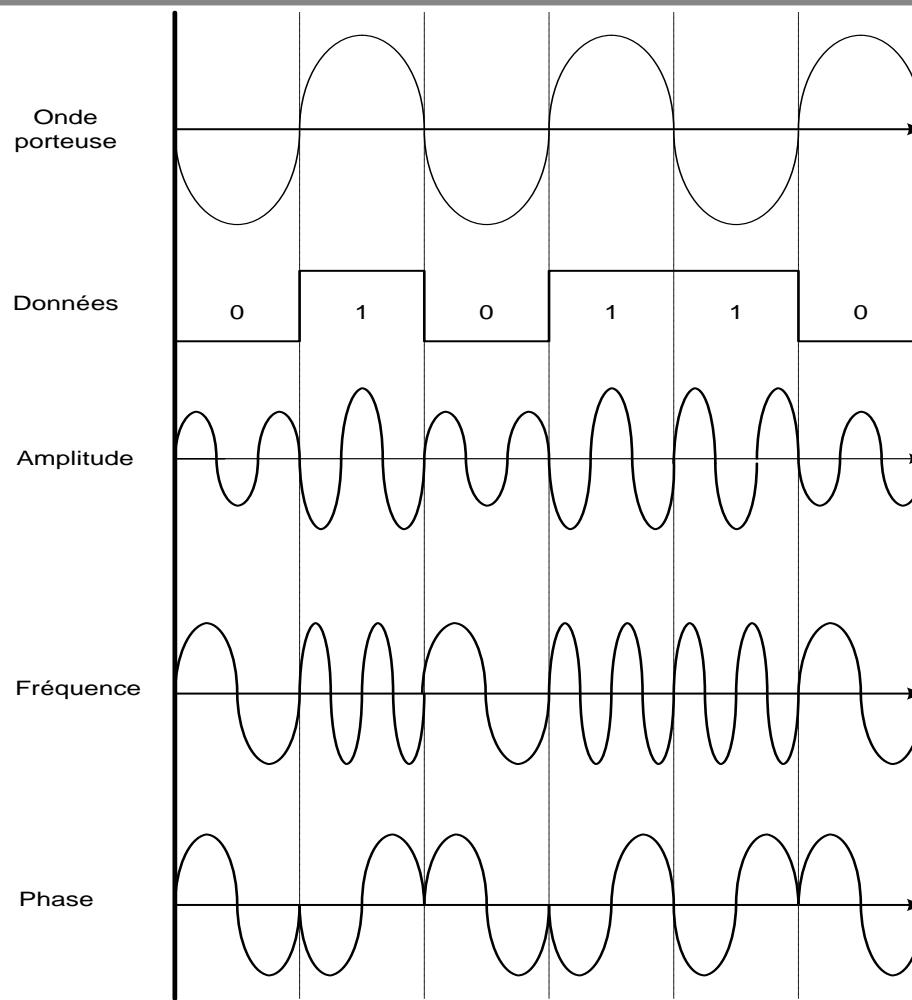


## Multiplexer

- 
- Le type de support de la communication: câbles (ou fils), fibre optique, ondes électromagnétiques - avec comme différence: la vitesse de transmission, la bande passante, la fiabilité, l'atténuation (et le besoin de «répéteur» successif), le prix, la simplicité,...
    - Paire de fils torsadé (téléphone) ou câble coaxial (télévision). Le câble coaxial est plus fiable, plus rapide: centaines Mb/s est possible, large bande passante (utile pour les canaux de TV) et permet des distances de communication plus grande.
    - fibre optique: plus fine qu'un cheveux et peut mesurer des milliers de km. Hyperrobuste au bruit, extrêmement rapide (on rejoint les terabits/sec, le problème devenant la conversion électricité/lumière), extrêmement large bande passante, permet des communications très distantes et est moins coûteux à installer. LA technologie de communication de l'avenir (et déjà du présent). Le seul problème est que la communication doit être guidée, point-to-point, et encore difficile à complètement maîtrisée

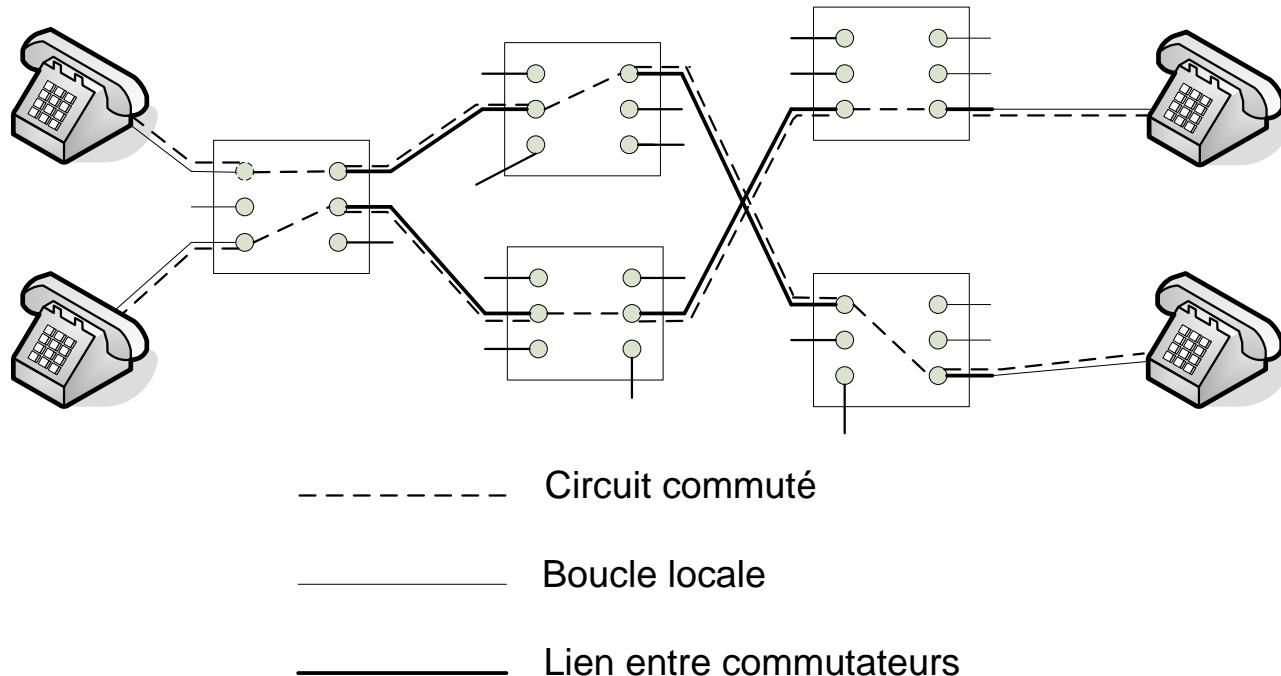
- 
- les ondes électromagnétiques - Elles permettent la mobilité des «utilisateur», et sont plus simples à «installer». Elles n’ont pas de support particulier, souvent elles sont utilisées dans le spectre  $> 1 \text{ GHz}$  (les micro-ondes). Elles ne sont pas guidées (bien que parfois on peut plus ou moins les diriger vers une antenne) et sont préférentiellement utilisée en «broadcast». Les grandes fréquences vont plus vite mais s’atténuent très facilement et sont très sensibles aux interférences. Les fréquences ne se mélangeant pas.
  - L’importance de la modulation: on module l’information dans un signal périodique de fréquence donnée. De ce signal, on peut utiliser soit la fréquence, soit l’amplitude, soit la phase pour transmettre l’information.
  - Les micro-ondes sont largement utilisées pour les communications télép. de longue distance, les téléphones cellulaires, la distribution TV. De plus en plus, on entre dans le spectre de la lumière (par ex. l’infrarouge).
  - La technologie « bluetooth » ou « wifi » des réseaux sans fil, communiquant par onde radio à petite distance et sans trop d ’obstacle.





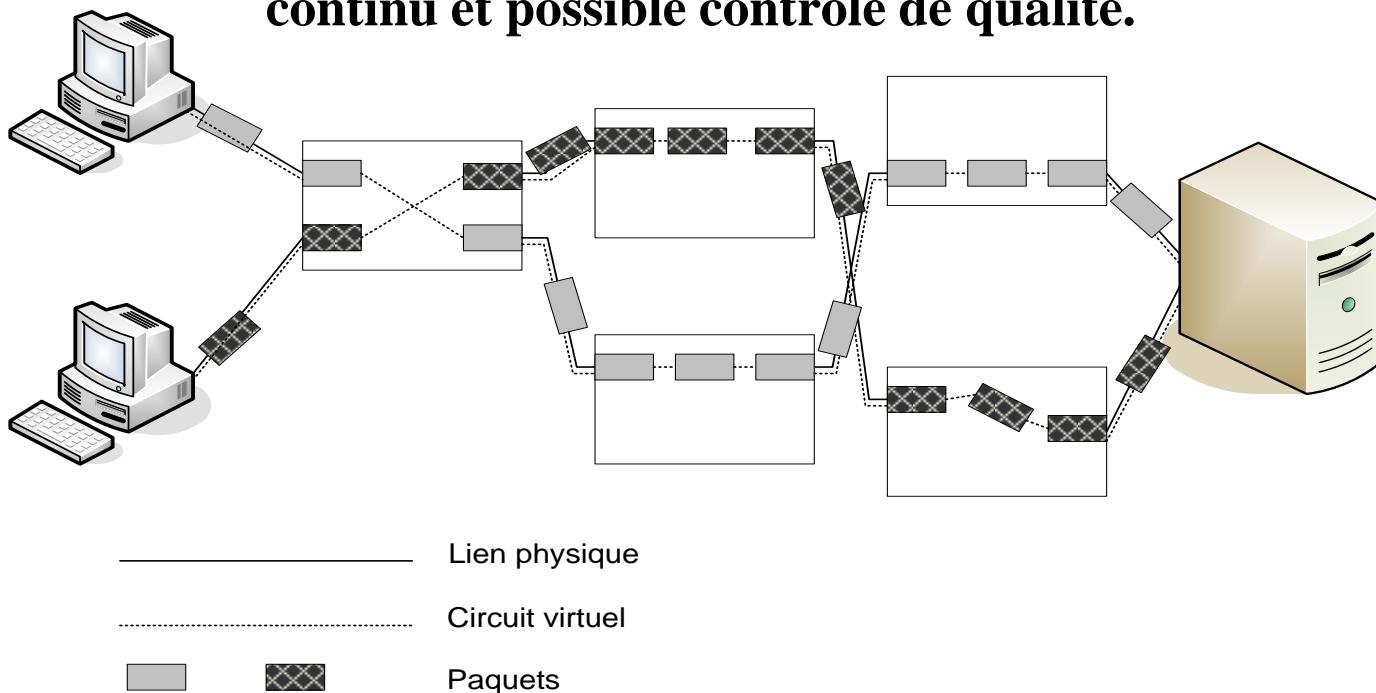
# Types de commutations

- Commutation de circuit: garantit un flux continu de parole.

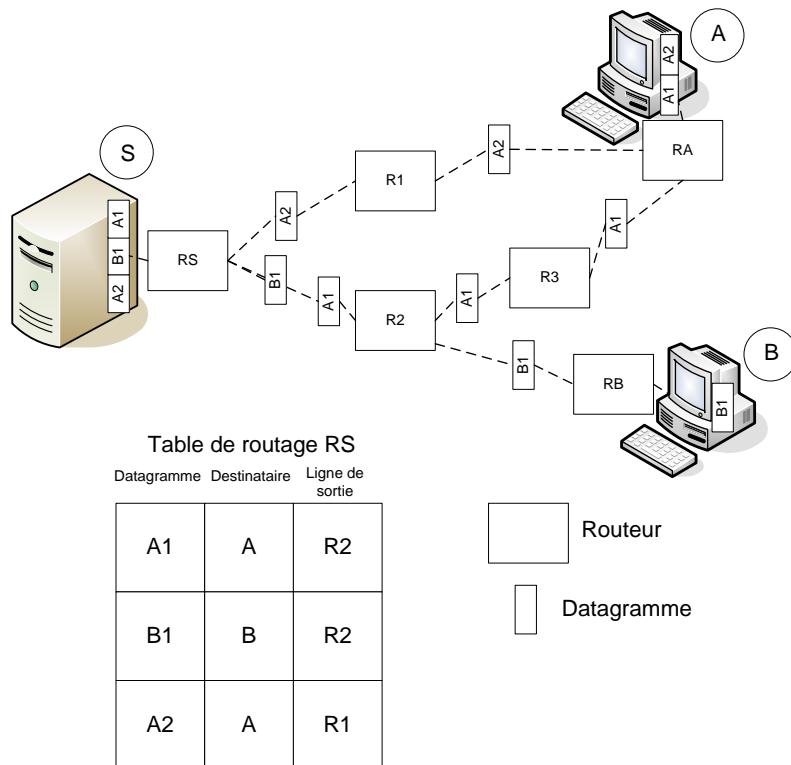


---

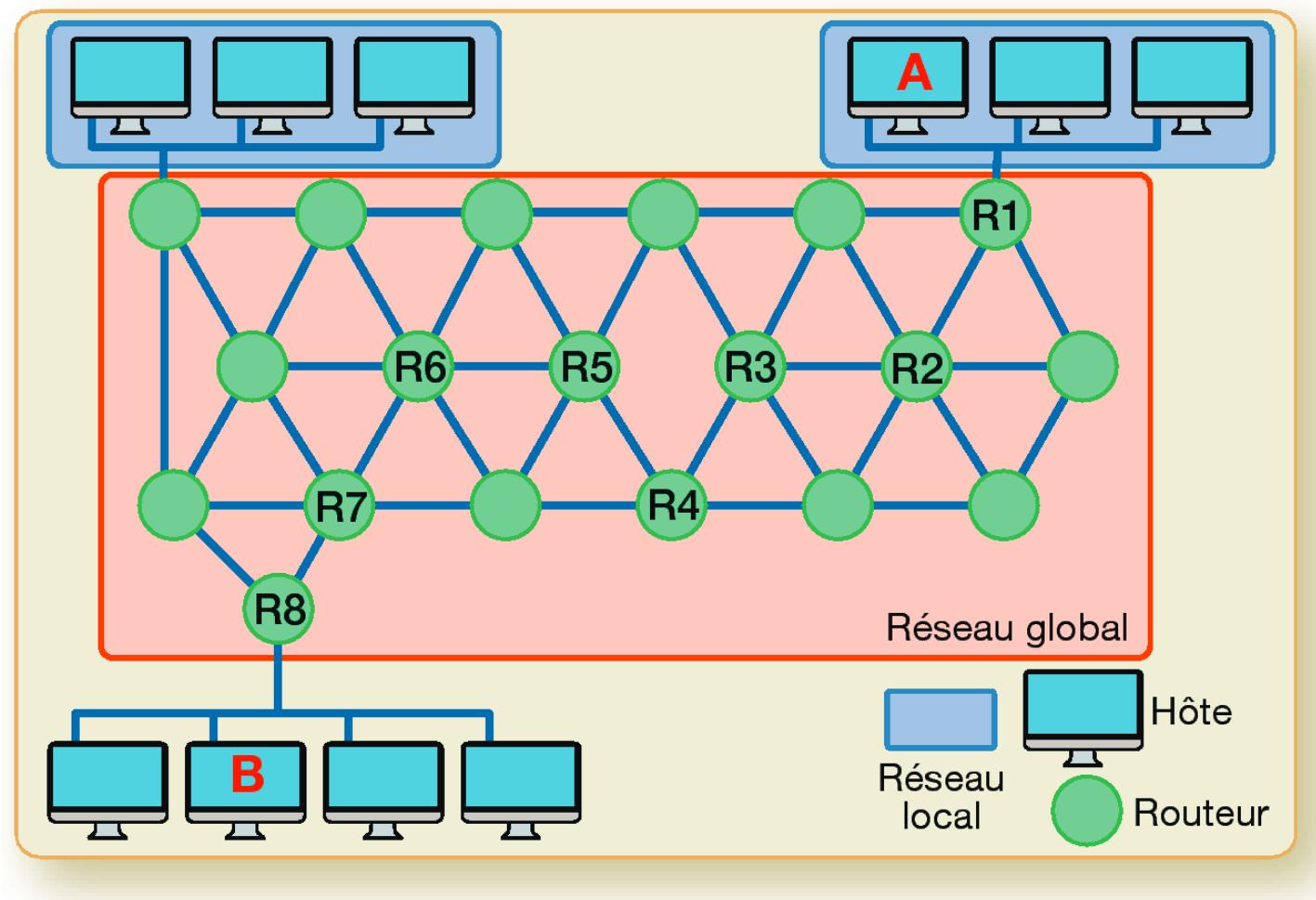
## **Commutation de paquets en mode connecté: mélange de paquets mais flux continu et possible contrôle de qualité.**



## Commutation de paquets en mode non connecté: plus efficace mais aucun contrôle de qualité.



# Routage des paquets de A à B



# Types de réseaux

---

- Réseau téléphonique commuté
- RNIS ou ISDN
- ADSL: accroissement des débits de la boucle locale par l'addition de hautes fréquences
- GPRS (general packet radio system)
- UMTS
- Bluetooth: réseau sans fil à faible intensité et à connexion très courte
- Wifi: le plus répandu. Présence fréquente de hot spot dans les lieux publics.

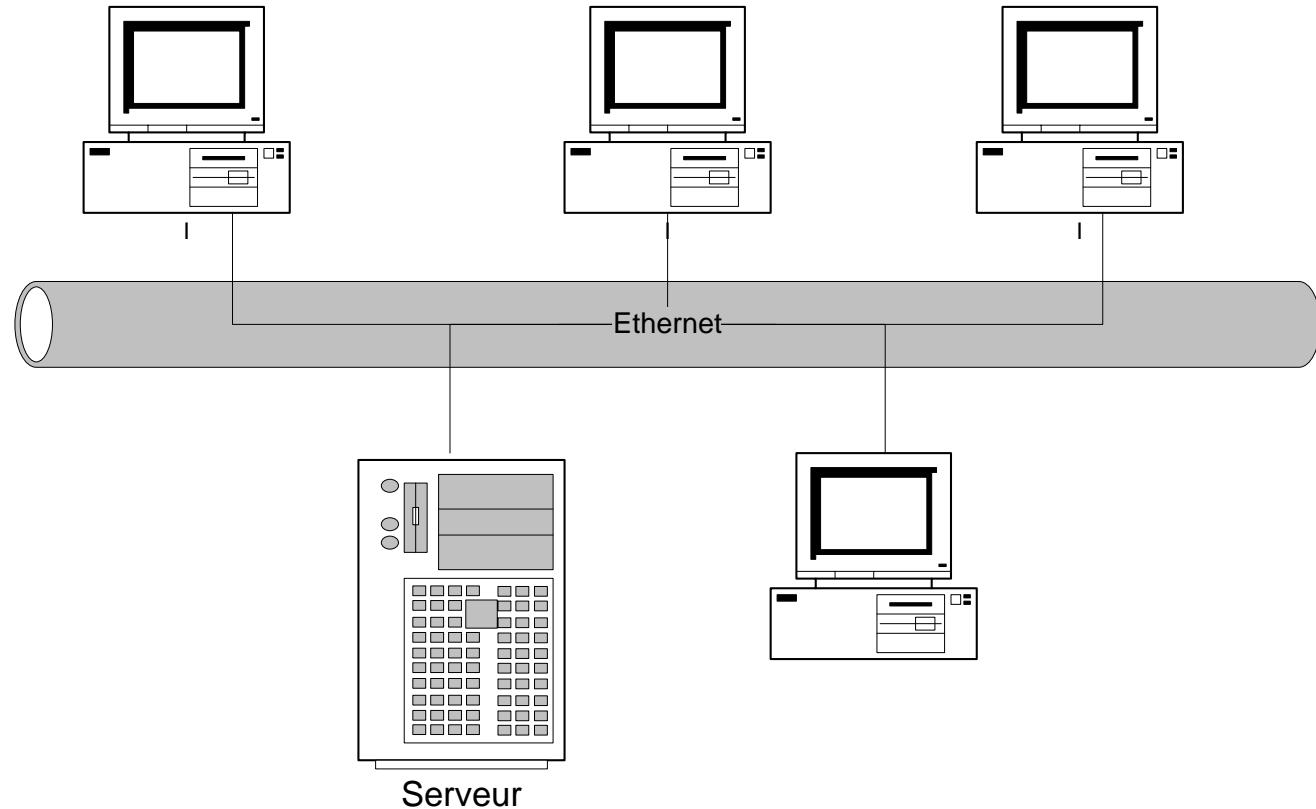
# Topologie des réseaux LAN ou locaux

---

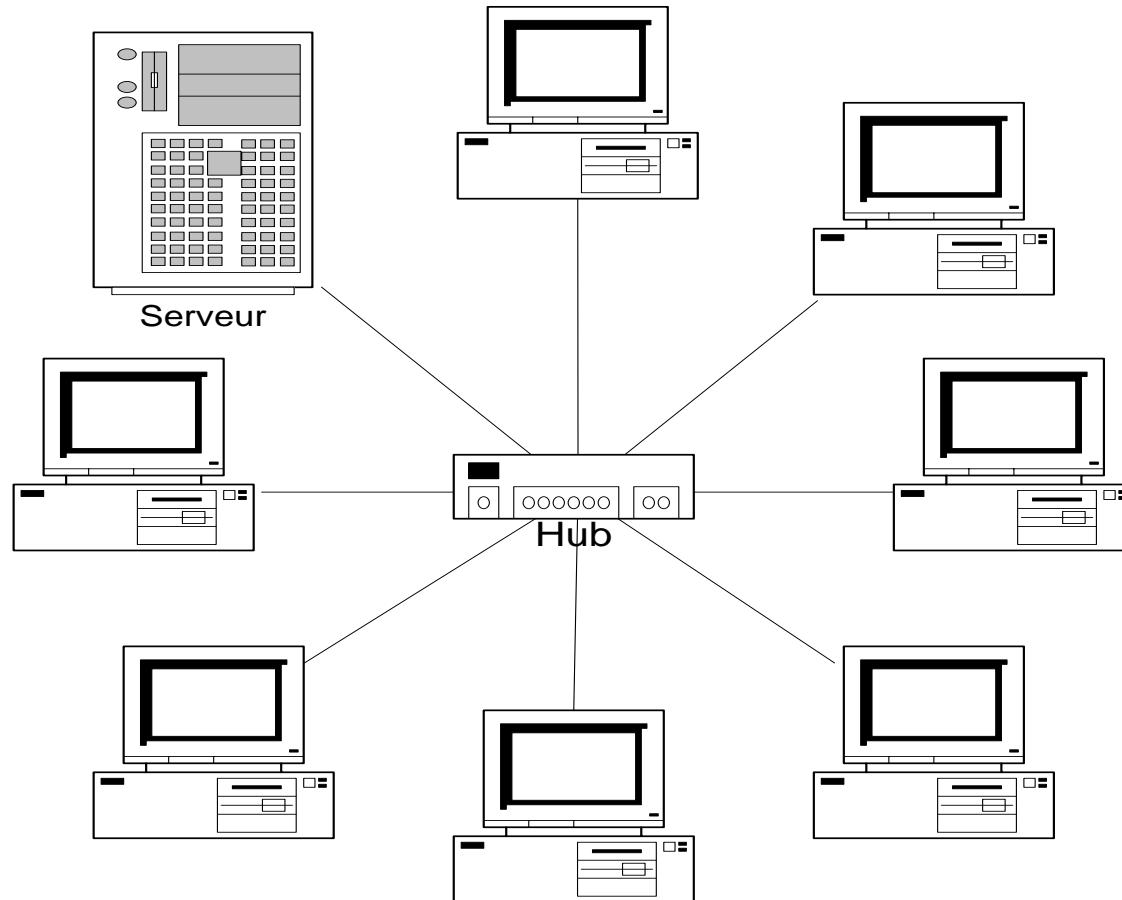
- en bus (type ethernet) - chacun des postes est indépendant et communique quand ça lui chante et que la voie est libre, il n'y a pas de coordination centrale. Chaque poste scrute l'adresse de la « trame » et vérifie si elle lui est adressée. Des collisions sont détectées et gérées -> peu fiable.
- en «étoile» (aussi ethernet): il existe un poste central, le « hub » qui relie les stations entre elles. Si le hub commute entre les différentes paires de «postes communicateurs» alors il devient « switch ». Les commutations sont ouvertes et fermées à tour de rôle -> plus fiable, plus efficace.
- en «token-ring» ou « anneau » : les informations circulent dans une seule direction. On évite les «collisions» par l'utilisation du réseau uniquement possible par le poste qui s'approprie le jeton.

# En bus

---

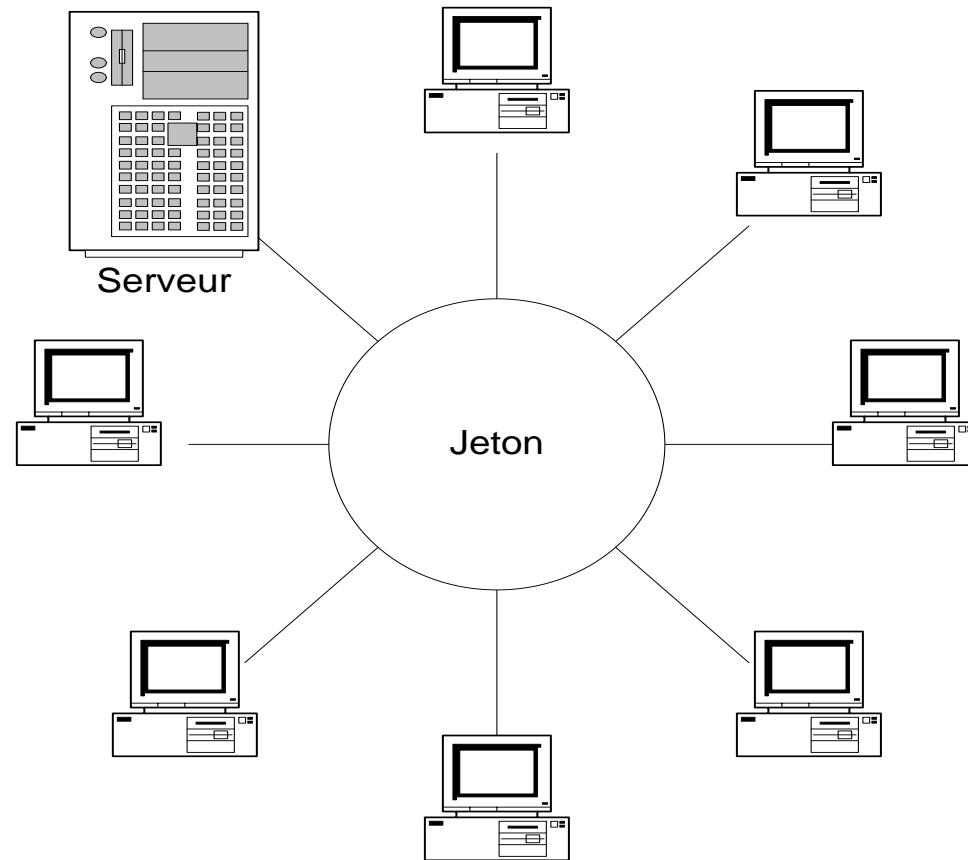


# En étoile



# En anneau

---



# Les réseaux WAN ou globaux

---

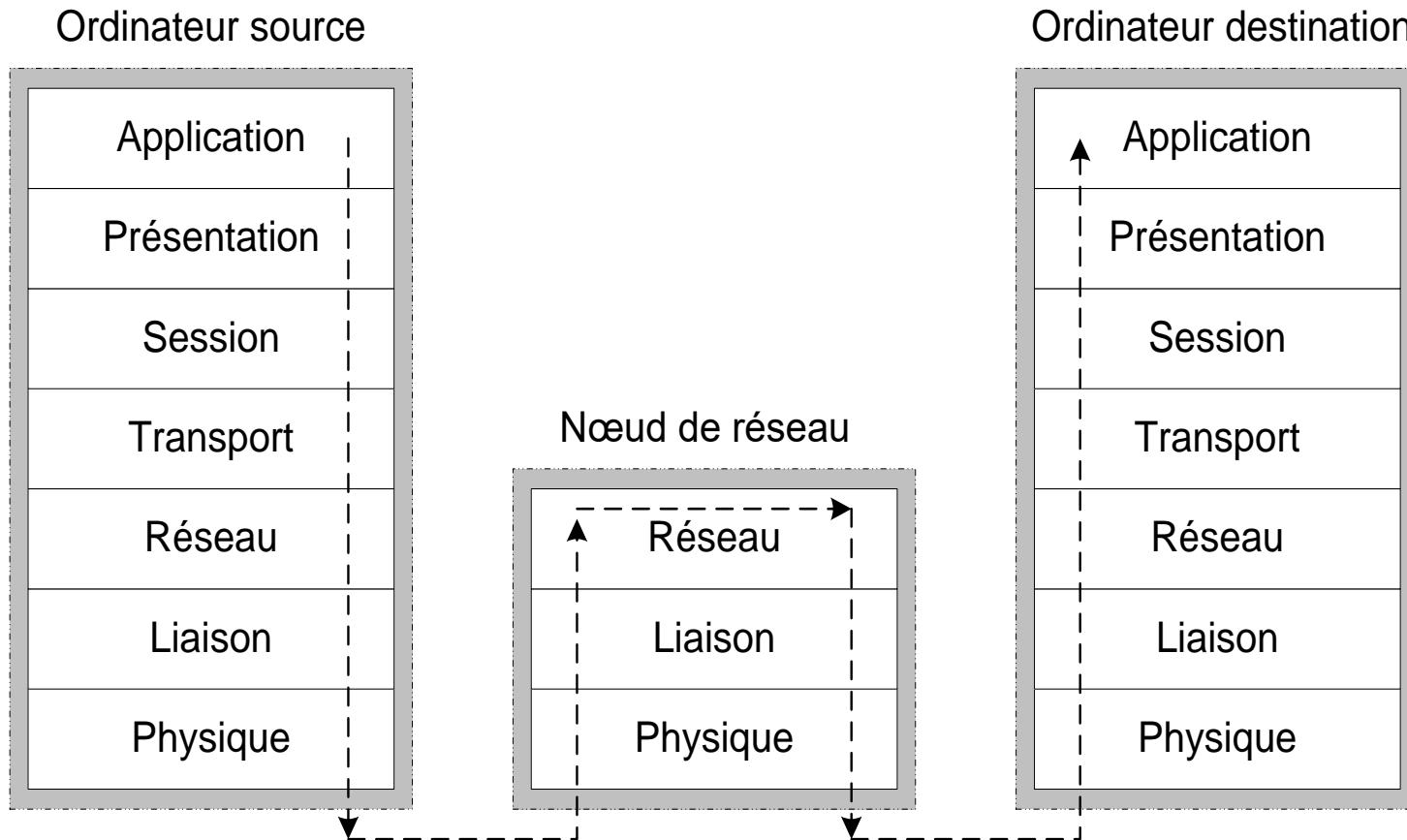
- Les réseaux WAN (Wide Area Network) ont une étendue très large. Le réseau comprend des sous-réseaux (LAN) et des «ponts» ou des «routeurs». Les «ponts» se limitent à répéter le signal et à isoler les LAN entre eux. Les «routeurs» sont plus «intelligents». Soit, ils transmettent le message localement soit ils le re-dirigent vers un autre routeur (en maintenant une «table de route»). Internet est le plus célèbre des WAN.

# Les protocoles de communication

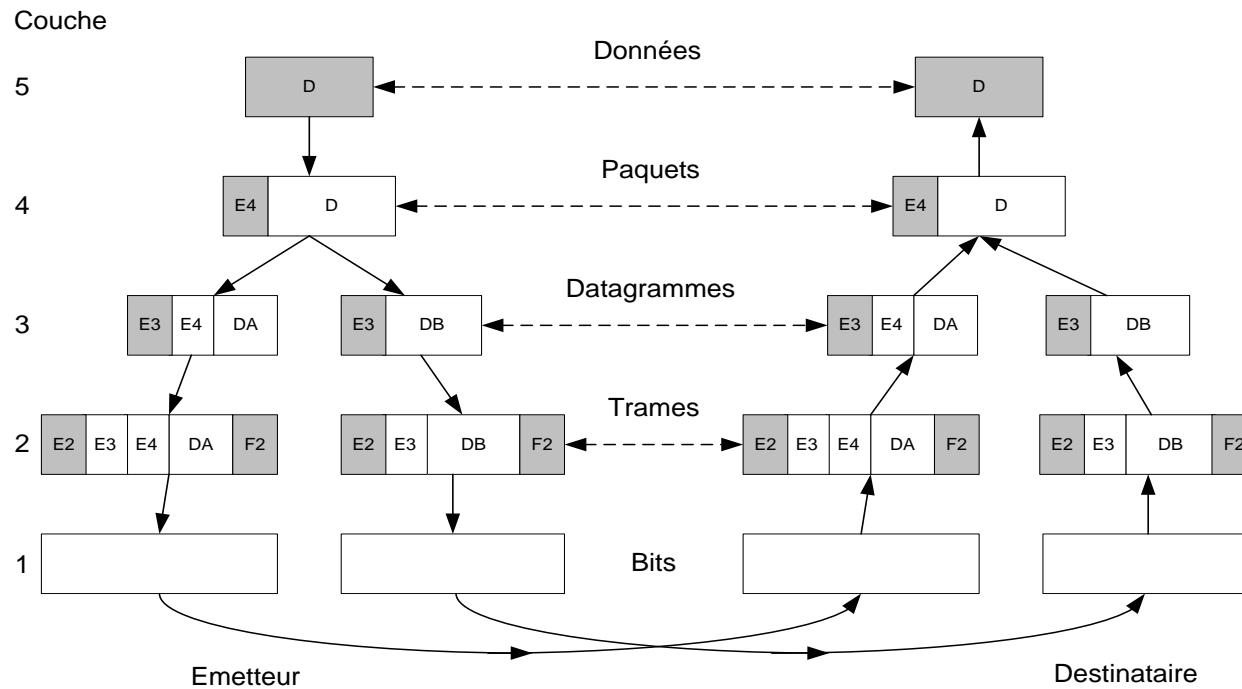
---

- Les protocoles sont hiérarchisés ou superposés en couche, une communication établie à un haut niveau ne se préoccupe pas du déroulement effectif de cette communication au niveau juste en dessous. Par exemple, une communication avec traduction simultanée. La communication se fait en haut, les traducteurs l'exécutent effectivement en bas. Les niveaux se parlent entre eux sans s'occuper du dessous. Cela permet de stabiliser les protocoles, rien ne change au-dessus si cela change en-dessous.
- Une application tournant sur un ordinateur devrait pouvoir communiquer avec une autre application tournant sur un autre ordinateur sans se soucier de la manière dont celle-ci s'effectue pratiquement.
- Le protocole idéal de communication dans les WAN est le protocole OSI (Open System Interconnexion) qui se décompose en 7 niveaux.

# Modèle OSI



- 
- Ainsi les différents niveaux dans un WAN devrait être (du plus haut au plus bas):
    - Physique: Au niveau le plus bas: transfert de bits sur paire torsadée
    - Liaison: Envoi des données sous forme de trames
    - Réseau: Données en paquets + optimisation du chemin + contrôle de congestion
    - Transport: Détection des erreurs de transfert uniquement chez les interlocuteurs
    - Session: ouverture et synchronisation du dialogue
    - Présentation: Syntaxe des données échangées
    - Application: Telnet, http, email, ....
  - Une simplification de ce protocole est le protocole TCP/IP en 4 couches. Toute nouvelle application Internet (email, http,...) doit se conformer à ce protocole. Pour les LANs le protocole Ethernet s'est bien imposé.
  - Les protocoles peuvent s'appeler entre eux: dès qu'un message global est distribué dans un LAN, TCP/IP peut passer la main à Ethernet. Les OS réseaux aussi doivent se connecter au protocole: UNIX et TCP/IP ou Novell et Ethernet.



D = Donnée

E = En-tête

$F = F_{in}$

# Le protocole ethernet

---

- la plupart des LAN existent pour le partage des ressources: fichier, programme et imprimante.
- aujourd’hui le protocole ethernet pour les LAN configurés en bus ou même en étoile est le plus répandu. On parle de plus en plus d’ Ethernet Gbits.
- le niveau le plus bas: physique: câble type coaxial: supportant un débit de de Gbits et un nombre de postes pouvant aller jusqu’à 1024
- un niveau plus haut: le «data layer», les données sont morcelées en trames de 500 bytes en moyenne. La connexion est de type «broadcast» car tous les postes ont accès à l’information. Quand une collision est détectée, le poste émetteur attend une durée aléatoire avant de re-émettre.
- chaque poste a une carte Ethernet (adresse MAC) qui, lorsqu’il reçoit un paquet, compare l’adresse avec l’adresse du poste. Si le paquet est accepté, on vérifie s’il y a des erreurs.

- 
- une adresse Ethernet est longue de 6 bytes. Chaque vendeur de carte Ethernet doit s'enregistrer à l'IEEE (la société d'ingénierie américaine) qui lui confère les 3 premiers bytes, les autres seront propres à l'ordinateur qui la contient.
  - chaque paquet contient l'adresse de la source et du destinataire, le type de données (pour Internet, pour Novell, pour AppleTalk, ou tout autre protocole de plus haut niveau), les données, et enfin des bits de correction d'erreur.
  - la plupart des utilisations d'Ethernet se font via un protocole de plus haut niveau. Il est donc important d'inclure dans le paquet qui transite par Ethernet des informations concernant le routeur actuel, la session, le transport (tous les niveaux propres au protocole du plus haut niveau) et la prochaine destination (adresse du prochain routeur,...)
  - Ethernet est très bien pour des réseaux courts, avec un trafic léger, car il est très simple et très flexible (pas de contrôle central) mais se dégrade quand les probabilités de collision s'accroissent: réseau plus long ou trafic plus intense.

# Sans fil – la norme IEEE 802.11

---

- Ondes radio— par exemple 2.5 GHz
- Hot spot (point d'accès) couvre une cellule
- Topologie ESS = Ethernet entre les points d'accès.
- Le protocole de communication est de type Ethernet avec détection des collisions et écoute du réseau avant d'émettre.
- Transmission accélérée par « étalement de spectre » pour atteindre des dizaines de mégabits. Saut entre les fréquences pour éviter les collisions et interférences si plusieurs transferts doivent se faire simultanément dans une même cellule → Cela permettait à l'origine une forme de dissimulation.

# Internet et TCP/IP

---

- La partie TCP (le haut niveau - Transmission Control Protocol) reprend d'OSI les niveaux «présentation», «session» et «transport» - la partie IP (le bas niveau, Internet Protocol) reprend les interactions de bas niveau propres à d'autres protocoles.
- les applications Internet telles: Web,email,ftp,telnet sont au-dessus de TCP/IP mais utilisent ce protocole pour leur déroulement.
- une adresse IP est un entier de 32 bits, elle spécifie l'adresse d'un domaine sur le WAN, les 3 premiers bytes, puis ensuite l'adresse de l'hôte sur ce domaine. Les adresses peuvent être de 3 classes: A, B et C suivant que l'on attribue au domaine un grand nombre d'hôtes. Par ex. un domaine de classe A pourra accueillir 16000000 d'hôtes.
- C'est le DNS (Domain Name Server) qui traduit les noms symboliques en les adresses Internet:colorado.edu en 128.130.0.0. Tous les hôtes du domaine comme cs.colorado.edu auront une adresse: 128.130.244.9

## Classe

A	0	Adresse réseau sur 7 bits	Adresse station sur 24 bits
---	---	------------------------------	-----------------------------

B	10	Adresse réseau sur 14 bits	Adresse station sur 16 bits
---	----	----------------------------	-----------------------------

C	110	Adresse réseau sur 21 bits	Adresse station sur 8 bits
---	-----	----------------------------	----------------------------

---

## □ **Le protocole hiérarchique:**

- au niveau de l’application, on fixe l’adresse IP du destinataire (en recherchant dans le DNS) et on passe les données à transmettre
- au niveau TCP, on fait les paquets, on établit la session (le premier paquet s’en occupe), on envoie les paquets, on s’assure que tous les paquets sont reçus dans le bon ordre. On s’occupe aussi de fournir les bits de détection et correction d’erreur.
- au niveau IP, on s’occupe du routage. On regarde si le destinataire est local au réseau, si c’est le cas, on modifie son adresse en une adresse locale (par ex. adresse Ethernet) et on est relayé par le protocole local, sinon on indique comme nouvelle adresse l’adresse du prochain routeur. Si le routeur courant est le destinataire, on passe la main au niveau TCP.

- 
- Il est intéressant de subdiviser un gros réseau en de multiples LAN avec des routeurs qui peuvent isoler les LANs entre eux. Par exemple un administrateur de domaine de classe B peut décider d'avoir 64 sous-réseaux LAN et de dédier 6 bits des 16 qui lui sont alloués pour identifier le sous-réseau et les 10 autres pour identifier l'hôte dans le sous-réseau.
  - Il faut alors recourir au mécanisme de «submask» pour que l'on puisse bien retrouver l'adresse de l'hôte en la décomposant en l'adresse du sous-réseau et l'adresse de l'hôte dans ce sous-réseau.
  - Aujourd'hui tous les OS supportent TCP/IP.
  - Toutes les applications Internet courantes: email, telnet, WWW, ftp et celles plus récentes: Internet Phone et la vidéoconférence tournent au-dessus de TCP/IP.
  - Le protocole devrait bien donner lieu à une 6ème version avec: 128 bits d'adresse, des routeurs hiérarchiques, l'encryptage de l'information, des priorités permises sur les paquets, des facilités de type «Plug and Play» pour chaque nouvel hôte qui se connecte au réseau.

# L'avenir en matière de réseau

---

- accélération des supports et de leur bande passante: fibre optique + onde lumineuse
- amélioration des protocoles --> IP6
- utilisation d'un réseau commun pour le transfert des différents types d'information: téléphone, TV, Internet et autres communications informatiques
- homogénéisation des modes de transmission de données: entre les différents modes de commutation, par exemple ATM est une commutation de cellules (très petits paquets) en mode connecté.
- Le choix des circuits virtuels est fait en fonction des priorités et de la nature des données.
- Tarification, fonction du volume acheminé et de la qualité de service demandé.
- Homogénéisation et concurrence.

# Les répercussions

---

- Technologiques: (courrier numérique, téléphonie sur IP et télévision interactive, distribution des documents textuels et multimédias, informatique distribuée)
- Economiques (achat en ligne, disparition des intermédiaires, e-business, open source, peer-to-peer, commerce personnalisé, économie de l'accès plutôt que de l'appropriation)
- Sociologiques (« chat », nouvelles méthodes de travail à distance, rencontres par le Web)
- Culturelles (vidéothèque, photothèque, médiathèque, bibliothèque, tout en ligne, que faire avec ces tonnes de documentation validées par personne mais si facilement accessibles)
- Politiques (campagne email, propagande électronique, vote électronique, terrorisme et malveillance informatique, espionnage électronique)
- Juridiques (droit d'auteur, utilisation du Web à des fins subversives ou amorales)
- Scientifiques (le grid computing, par exemple, qui fait d'Internet l'ordinateur le plus puissant à ce jour).