Licence 1 Sciences & Technologies
Informatique Generale

Une (brève) introduction aux Systèmes d'Exploitation

Olivier Dalle

Rôles du Système d'Exploitation (1)

- Rôle d'Interface entre Utilisateur et Ordinateur
  - Simplifier la programmation
    - Fournir l'illusion d'une machine virtuelle simple

le programmeur écrit 2 lignes:

```
fichier = open(''toto'', ...)
data = read(fichier,100)
...
```

le système exécute des 100aines de lignes !

A-t-on le droit d'ouvrir/lire toto ?
Quel matériel support ?
Dialoguer avec le support...
Traiter les erreurs, mettre à jour position, garder une copie en cache
...

  - Gommer les différences entre matériels
    - Portabilité des programmes

• *disque dur IDE:*

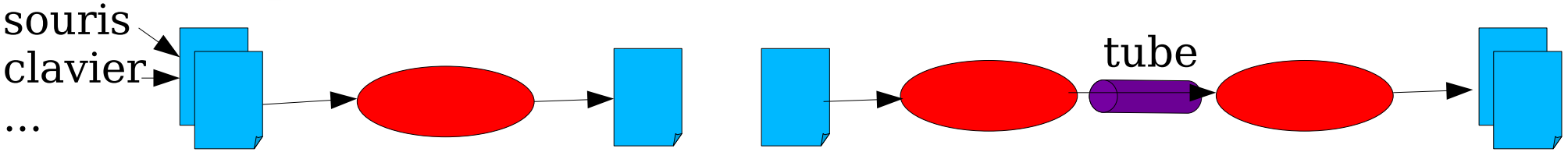
deplacer tete de lecture...
dialogue via bus PCI

• *clef USB:*

pas de pièce mécanique
dialogue via bus USB

Exemple de Machine Virtuelle : le Système Unix

- A l'origine, seulement 2 abstractions !
  - Les fichiers
    - Ils servent à tout et n'importe quoi...
      - Données utilisateur
      - Programmes, librairies (données exécutables)
      - Périphériques (clavier, souris, son, terminaux, ...)
      - Canaux de communication
      - Les programmes, librairies
  - Les processus
    - Un programme en cours d'exécution
      - Eventuellement plusieurs fois le même
    - Chacun a son propre contexte
      - Mémoire utilisée, fichiers ouverts (+position), parents, utilisateur, ...



La Notion de Processus

- Processus = (instance de) programme en cours d'exécution

- Dans un système multi-tâche
  - Les processus sont en compétition pour l'obtention des **ressources**
    - En particulier le processeur
    - Mais aussi : disques, réseau, mémoire, imprimante, ...
  - Partage du CPU (*time-sharing*)
    - Donne l'illusion d'exécution parallèle
      - pseudo-parallélisme sur mono-processeur
      - pseudo et vrai parallélisme sur multi-processeurs

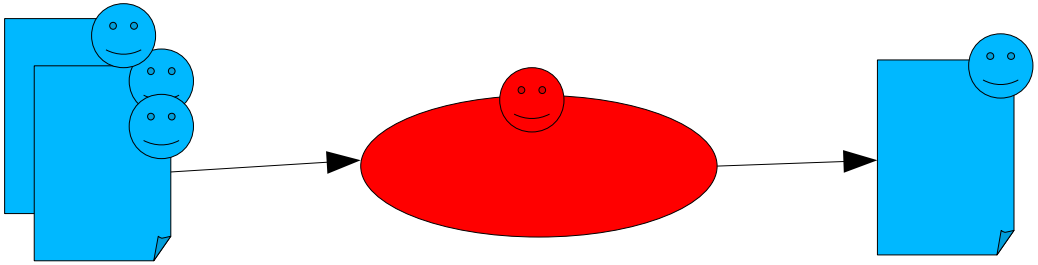
Pourquoi Plusieurs Processus ?

- Utiliser au maximum le processeur
  - Mettre à profit les périodes d'inactivité
    - Ex: attente d'une opération d'entrée/sortie
- Faciliter l'exécution de tâches concurrentes
  - Multi-tâches = exécution **simultanée** de plusieurs programmes
    - Eventuellement plusieurs fois le même
    - Besoin de distinguer les informations propres à chaque exécution
- Permettre à plusieurs utilisateurs de "cohabiter" de façon transparente
  - Multi-utilisateurs : chaque ut. peut exécuter ses propres programmes sans concertation avec les autres

Rôles du Système d'Exploitation (2)

- Rôle de Protection
  - du Matériel
    - Durée de vie limitée (cache des écritures, ...)
    - Utilisation erronée (fréquence maxi moniteur, ...)
    - Mise en sécurité (sonde thermique, accéléromètre MacBook Pro ...)
  - des Utilisateurs (confidentialité)
    - Surtout en contexte MULTI-UTILISATEUR
      - Respect des permissions sur les fichiers
      - Mise en place de stratégies avancées
        - Security Enhanced Linux
        - Stratégie de sécurité windows
    - Firewall (pare-feux)

Permissions sur les Fichiers ?



- action = processus = utilisateur connu
- fichier = permissions (lecture, ecriture, destruction, ...)
  - propriétaire = utilisateur connu (en gal celui qui crée)
  - liste additionnelle d'utilisateurs
- Exemple Unix:
  - 3 catégories d'utilisateurs /fichier
    - user (proprio), group (amis), other...
  - 3 types de permissions
    - read, write/create, execute/traverse

Rôles du Système d'Exploitation (3)

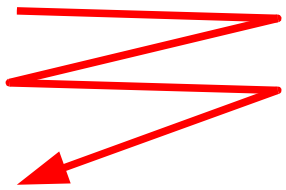
- Rôle d'Autorité (arbitrage)
  - Partage des Ressources
    - Surtout en contexte MULTI-UTILISATEUR
      - Equité (J'attends mon tour depuis 1 heure !!)
      - Efficacité (Phénomènes de " vague verte " ...)
        - Ex: Algorithme de l'ascenseur
      - Evitement des situations de blocage
        - Ex: L'agent qui fait la circulation au carrefour
  - Respect de « la loi »
    - Verrou sur fichier
      - Empêcher deux écritures simultanées (Problèmes dits « d'exclusion mutuelle »)
    - Quotas disque
      - Empêcher qu'un utilisateur s'accapare tout l'espace disponible

Exemple de Problème d'Exclusion Mutuelle

- Une **instruction** assembleur est indivisible, mais pas une **suite d'instructions**

Variable commune: n

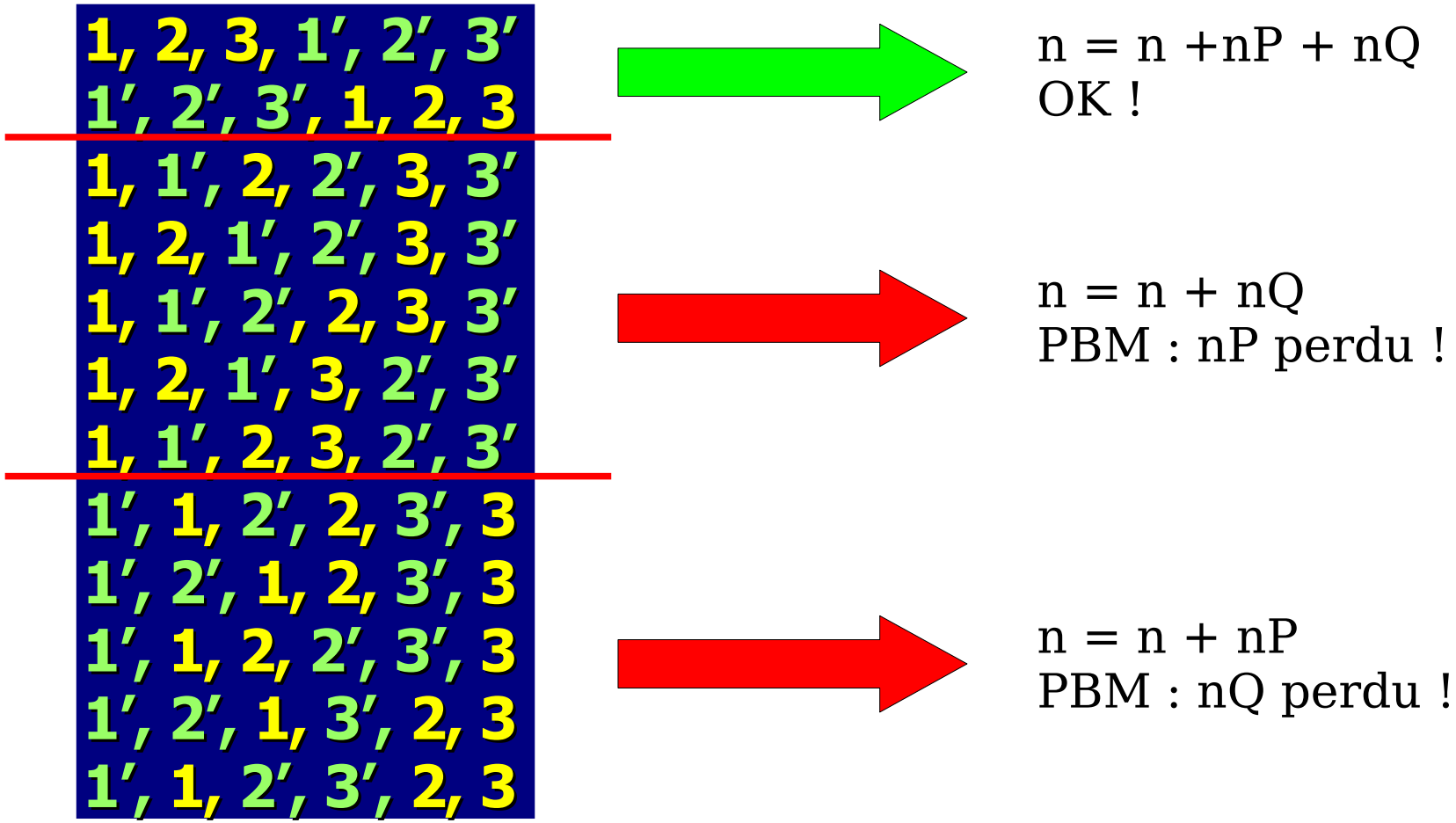
```
P(Virement)
…
n:=n+nP
Actions_P :
1. load Reg_P, n
2. add Reg_P, nP
3. store Reg_P, n
…
```



```
Q(Virement)
…
n:=n+nQ
Actions_Q :
1. load Reg_Q, n
2. add Reg_Q, nQ
3. store Reg_Q, n
…
```

Exemple de Problème d'Exclusion Mutuelle(2)

- Entrelacements possibles lors de l'exécution de P et Q

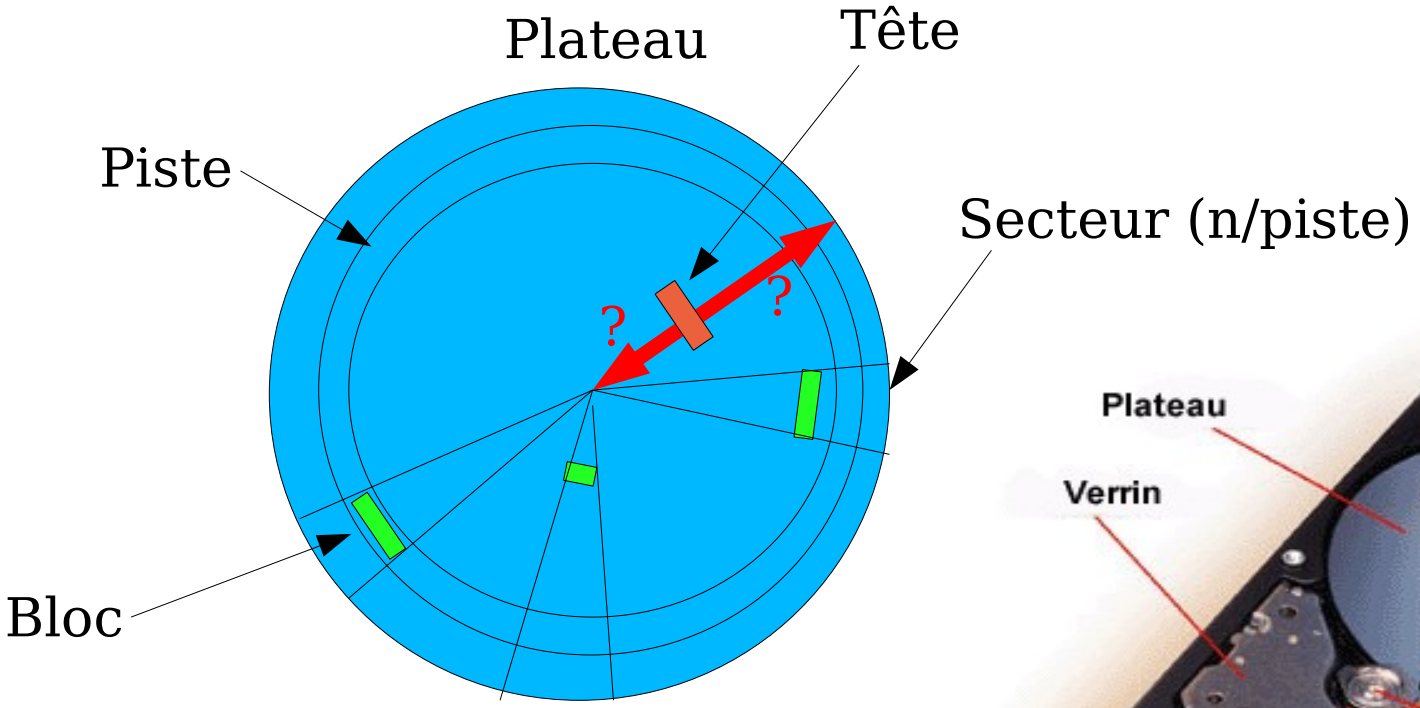


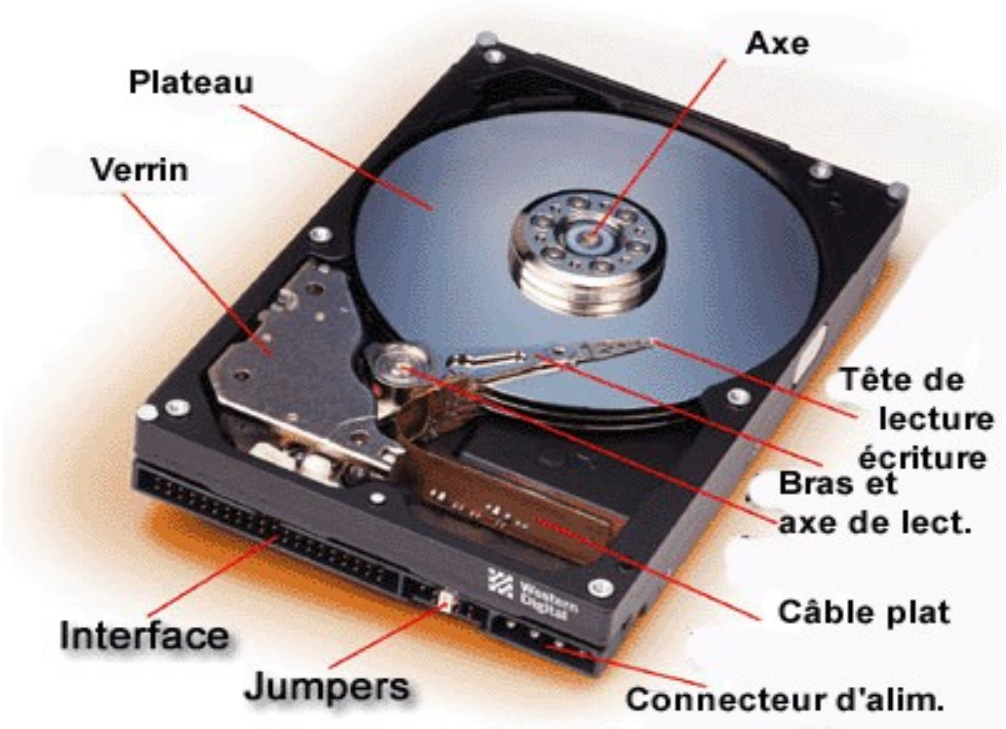
Rôles du Système d'Exploitation (4)

- Rôle d'Optimisation
  - Stratégies d'amélioration
    - Performances d'exécution (vitesse)
      - Exploitation efficace du matériel (caches...)
    - Occupation de l'espace (mémoire vive, disque, ...)
      - Eviter les « miettes »
    - Utilisation de la bande passante (communications)
      - Ratio données utiles /données contrôle
    - Consommation d'énergie (Batteries, dissipation, ...)
      - Arrêt rotation disque, baisse fréquence CPU, ...
    - Déplacement pièces mécaniques (Tête disque dur)
      - Ordonner les lectures au mieux
    - Usure contrôlée
      - Ne pas écrire toujours au même endroit
    - ...

Exemple d'Optimisation

- L'algorithme de l'ascenseur





Système d'Exploitation = Ensemble de Programmes

1. Le noyau

- Chargement lors du démarrage (*boot*)
- Gros programme (Linux = millions de lignes de code)
  - Gestion des évènements Matériels
    - disques, souris, clavier, ...
  - Gestion des ressources
    - Ex: Partage du CPU (time sharing)
  - Gestion Structures de Données
    - Systèmes de Fichiers, Identité des utilisateurs, ...
  - Fournit des « services » élémentaires : Appels Systèmes
    - «Ecrire fichier», «Obtenir la date», «Envoyer message», ...

Système d'Exploitation = Ensemble de Programmes

2. L'Interpréteur de commandes

- Utilisation interactive
  - Shell : Interface minimale homme-machine
  - Intermediaire pour lancer les autres programmes
  - Souvent dissimulé derrière interface graphique
- Utilisation « traitement par lots » (batch)
  - Programmation de séquences d'actions
  - Automatisation de taches
    - Séquence de (re-)démarrage
    - Réaction à certains évènements
      - Panne de courant
      - Erreurs
      - Tâches périodiques
      - Connexion d'un utilisateur

Système d'Exploitation = Ensemble de Programmes

3. Programmes de services supplémentaires

- Bannière de login,
- Partage de fichiers distants,
- Serveur d'impression
- Serveur d'authentification
- Messagerie
- Pare-feux
- ...

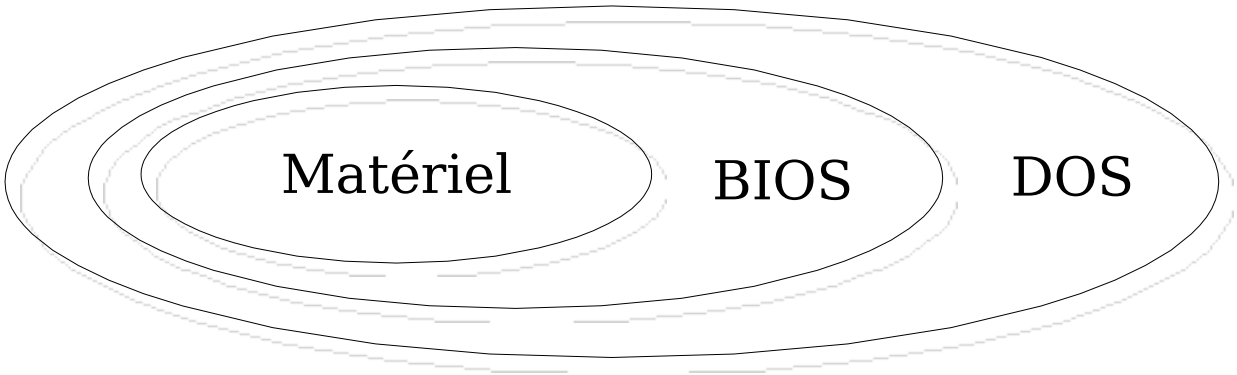
4. Utilitaires de base

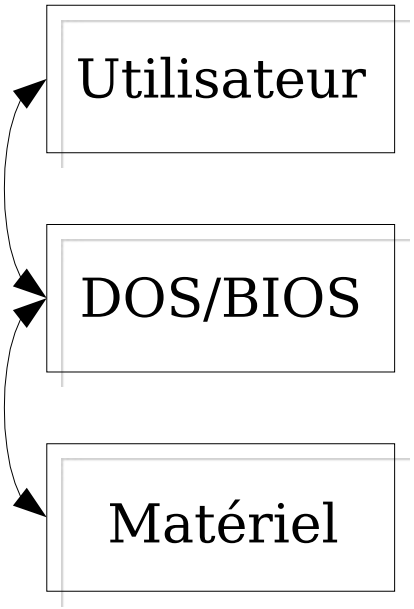
- Manipulations de processus
- Manipulation de fichiers
- Maintenance disques, utilisateurs, ...
- ...

Illustration Concrète : le Cas Simple(iste) MSDOS

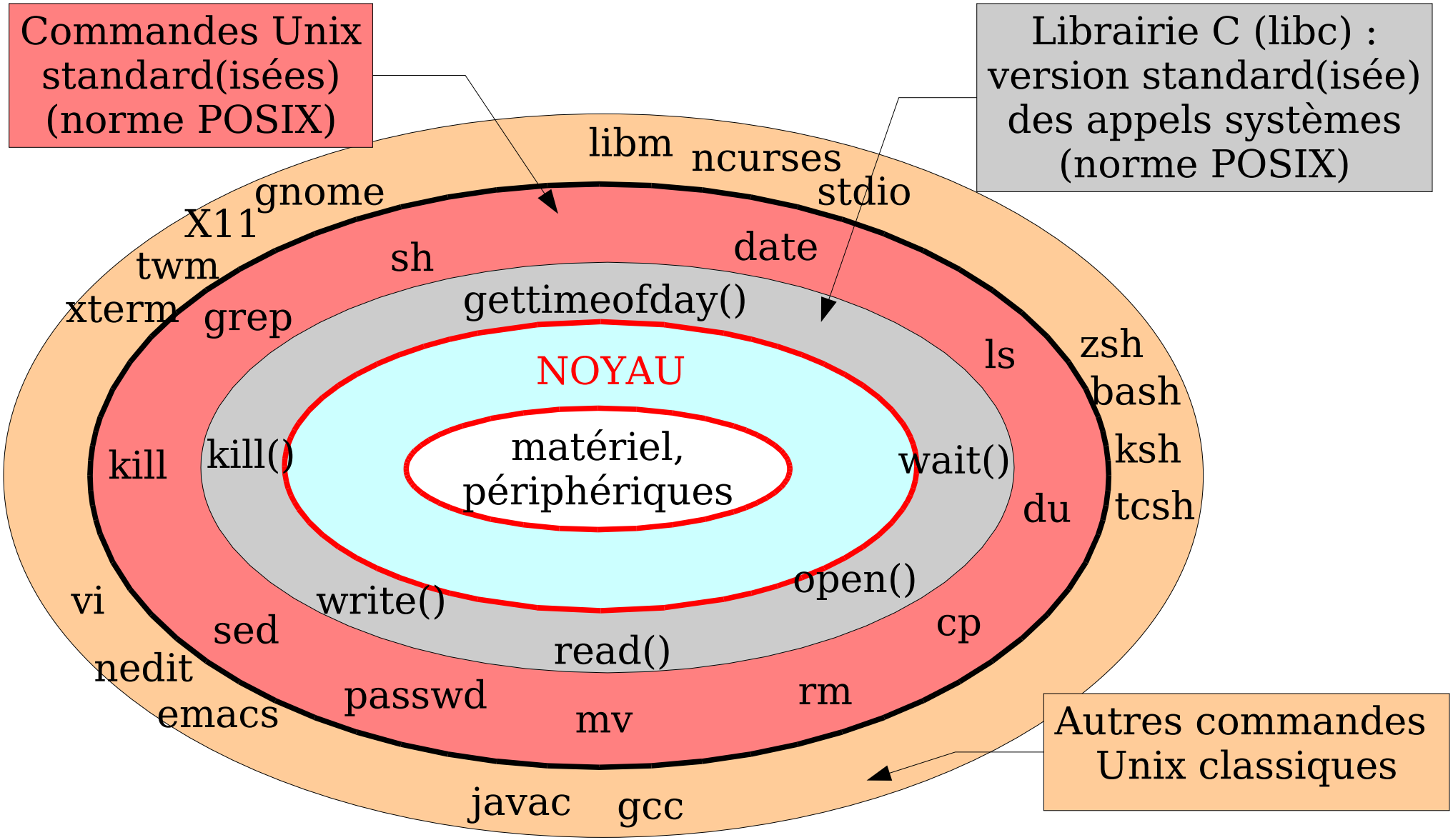
S'appuie sur une couche logicielle intermédiaire : le BIOS

- Fournie par le constructeur
- Offre des services de bas niveau élémentaires
  - Tests (POST), « Trouver boot sector », « lire secteur disque n°XX et le stocker en mémoire à l'adresse mémoire YY », ...
- DOS = 1 seul programme a la fois (mono-tâche)





Le Cas Complexe: Organisation d'Unix



Les Grandes Catégories de Systèmes d'Exploitation

- Généralistes
  - Windows, XP, MSDOS, CPM, ...
  - Unix/Posix (Linux, MacOS, BSD, ...),
  - VMS, OpenVMS...
  - ...
- Temps Reel
  - Utilisation Industrielle, Contraintes de temps
    - Robots, automates, ...
  - Ex: OS9, RTLinux, QNX, VxWorks, ...
- Embarqués
  - Contraintes espqce mémoire/énergie/puissance calcul
  - Telephones mobiles, PDA, Disques en reseaux, Routeurs, iPod, Consoles de jeux, ...
  - Ex: PalmOS, Symbian, Android, WindowsCE, xxLinux

Liste Exhaustive des Systèmes Existants ?

- http://www.csee.wvu.edu/~jdm/classes/cs258/OScat/
  - 14 Catégories, des 100aines de Systèmes !
    - Early Operating Systems, Batch Systems,
    - Timesharing Systems
    - Minicomputer Operating Systems, Microcomputer Operating Systems, Workstation Operating Systems
    - UNIX and its Derivatives
    - Fault-Tolerant and Highly Secure Systems
    - Distributed Operating Systems
    - Real-Time Operating Systems
    - Virtual Machine Systems
    - Parallel and Multiprocessor Operating Systems
    - Educational Operating Systems
    - Miscellaneous Operating Systems

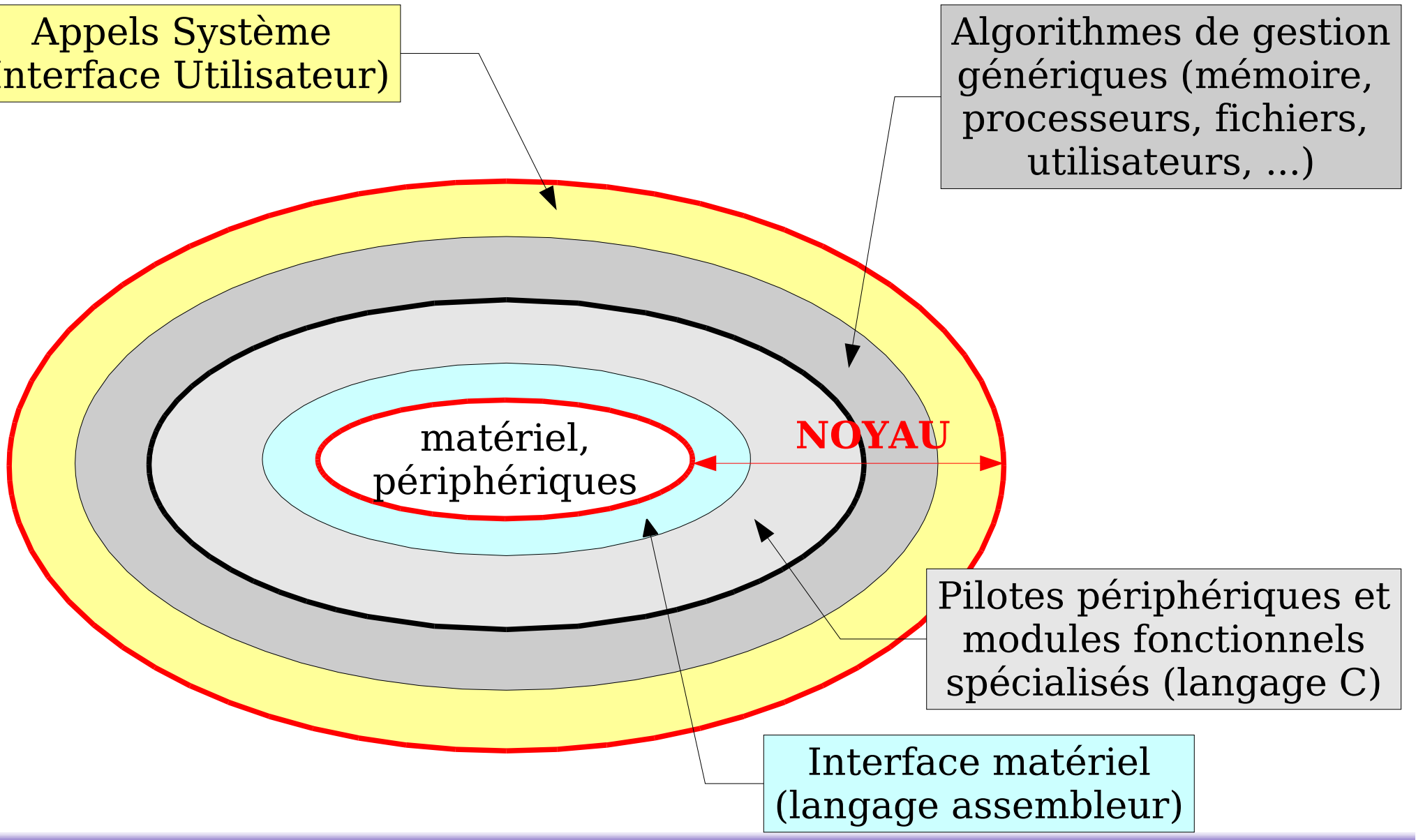
Propriétés Remarquables des Systèmes d'Exploitation

- Mono- ou Multi-tâches
  - Mono = exécution de tâches multiples en séquence
    - *Batch* = traitement par lots
  - Multi = exécution simultanée de plusieurs programmes
    - *Time Sharing* : partage du temps CPU entre les programmes
- Mono- ou Multi-utilisateurs
  - Mono = 1 seul utilisateur
  - Multi- = plusieurs utilisateurs en même temps
    - Chaque nouvel utilisateur lance une session indépendante
    - Interaction au travers d'un terminal (ou console)
      - Physiquement connecté : machine dite « multi-poste »
      - Connexion distante : par le réseau
      - Consoles virtuelles (linux) : plusieurs sessions sur un même terminal

Exemples de Systèmes

- MSDOS : mono-tâche, mono-utilisateur
- Windows95/98 : multi-tâches, mono-utilisateur
  - multi-tâches « coopératif » = peu interactif
- Windows NT/XP : multi-tâches, multi-utilisateurs
  - Une seule session à la fois dans la version de base
  - Sessions multiples possibles sur versions serveur
    - Depuis un autre SE Windows « client »
    - Grâce à certains prog spécialisés (Citrix, Rdesktop, ...)
- Unix (linux, MACOS) : multi-tâches, multi-utilisateurs, multi-sessions, multi-postes ...

Schéma d'Organisation Typique d'un Noyau Unix



Obtenir un service du Système : l'Appel Système

- Principe fondamental : créer des « points de passage » obligatoires
  - Le SE est le seul intermédiaire possible pour obtenir certains services
    - Accès fichiers, lancement programmes, manipulation de partitions, allocation mémoire, transmission de messages,...
  - Appel Système = (seul) moyen de communiquer avec le SE (pour un programme)
- l'AS ressemble à un appel de fonction ...
  - ... Mais son exécution est strictement protégée
    - Protection robuste assurée au niveau matériel (du CPU)
    - Exécution déclenchée par interruption logicielle

Mécanismes de Protection de l'Exécutif Système

- Combinaison de deux mécanismes :
  - Mode d'exécution privilégié
    - CPU bascule (du mode utilisateur) en mode noyau
      - Instructions supplémentaires (ex: masquer interruptions)
      - Registres supplémentaires (ex: gestion mémoire)
    - Bascule strictement contrôlée
      - Les programmes sont libres de basculer quand ils le veulent
      - La bascule entraîne l'exécution d'un (sous-)programme **préexistant**
        - Impossible de modifier les instructions de ce (sous-)programme depuis le mode utilisateur
  - Espace d'adressage système
    - Zone mémoire réservée
      - Inacessible en mode utilisateur
      - Contient le code et les données de l'exécutif système

Illustration : Passage en Mode Noyau