SYSTEMES D'EXPLOITATION

Motivations

- Complexité du matériel
  - écriture difficile et fastidieuse des programmes
  - maîtriser le fonctionnement des divers matériels
  - allongement de la formation de programmeurs
  - remise à niveau des programmeurs due à l'évolution des matériels

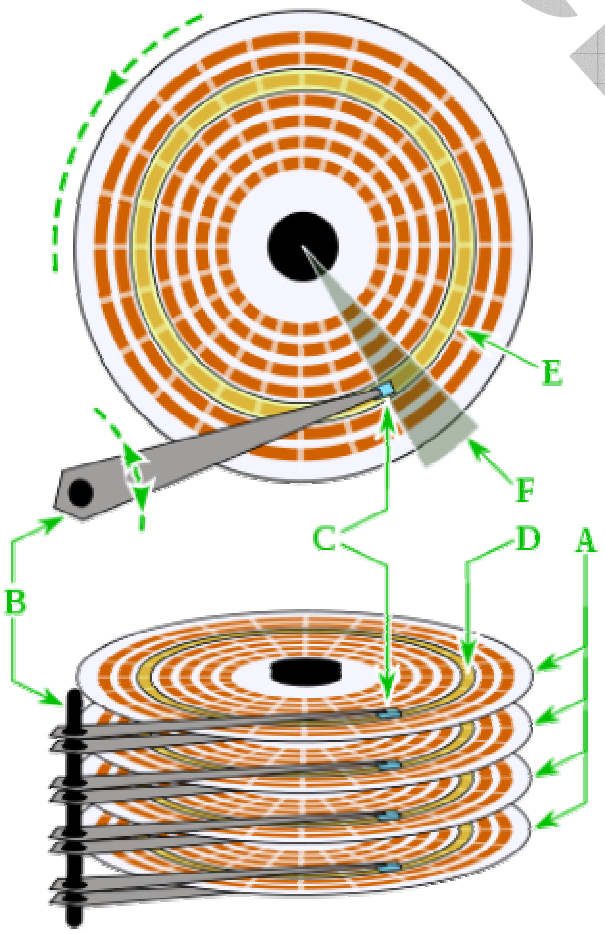
☞ faible productivité et complication inutile des programmes

portabilité nulle

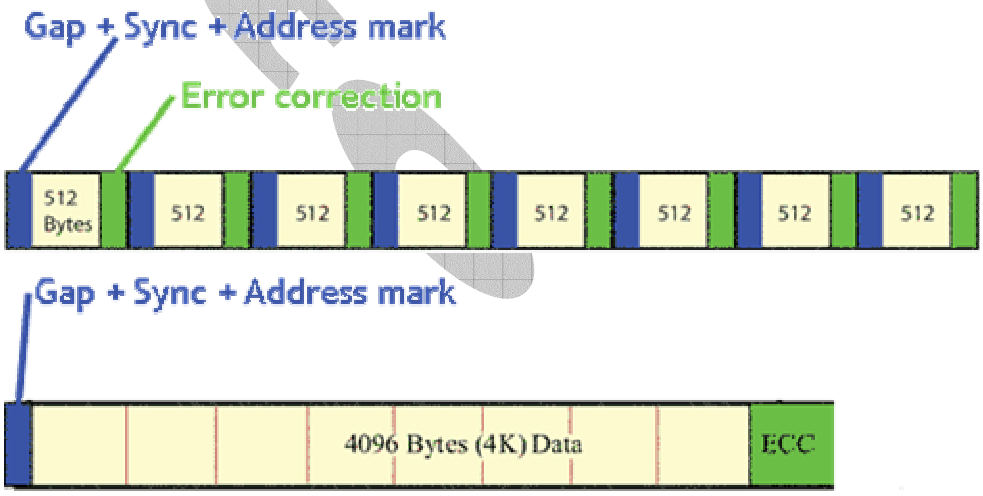
SYSTEMES D'EXPLOITATION

Motivations

- Exemple: Un disque dur



A: plateaux - B:Bras - C:Tête - D:cylindre - E:piste - F:secteur



VISION PHYSIQUE

Disque = cylindres + pistes + secteurs

Opération de lecture: 13 paramètres tels que:

En entrée:

- adresse du secteur à lire
- nombre de secteur par piste
- mode d'enregistrement utilisé
- distance entre 2 secteurs consécutifs

Résultats: 23 paramètres d'état et d'erreur

VISION LOGIQUE

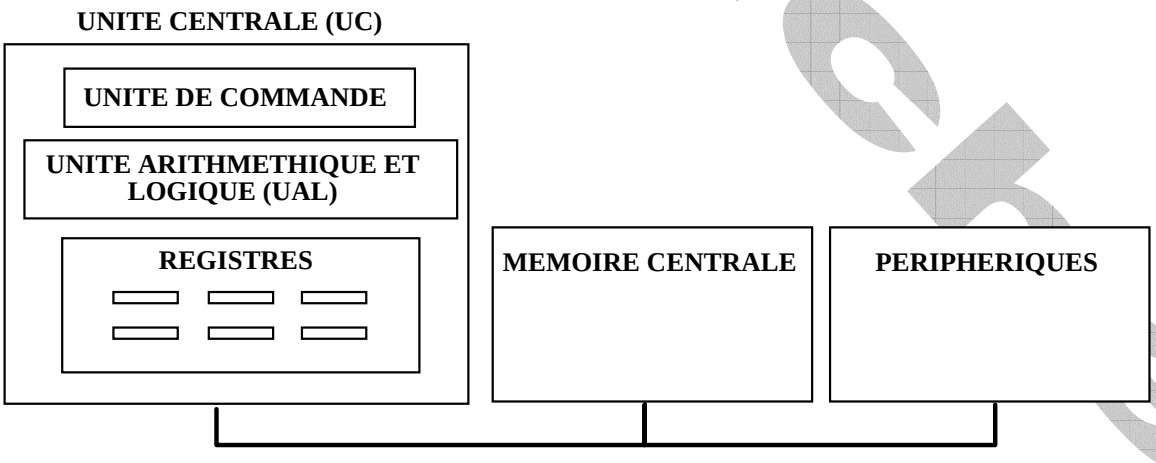
Disque = ensemble de fichiers

Appel système de lecture: 3 paramètres

nblu := read (fichier, tampon, nombre octets)

ARCHITECTURE DE L'ORDINATEUR

NOTIONS INDISPENSABLES



- Mémoire centrale (bit, octet, mot)
- Registre instruction (décodage)
- Compteur ordinal
- Exécution d'une instruction

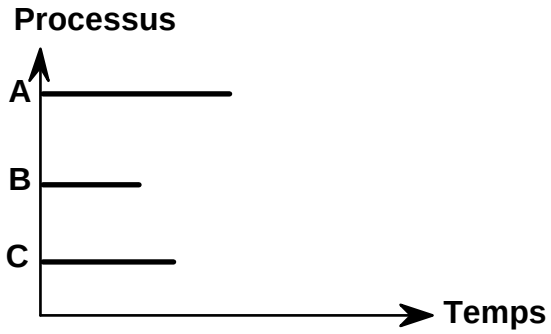
1) Chargement de la prochaine instruction à exécuter depuis la mémoire jusque dans le registre instruction
2) Modification du compteur ordinal pour qu'il pointe sur l'instruction suivante
3) Décodage de l'instruction que l'on vient de charger
4) Localisation dans la mémoire des données utilisées par l'instruction
5) Chargement des données, si nécessaire, dans les registres internes de l'UC
6) Exécution de l'instruction
7) Stockage des résultats à leur destination respective
8) Retour à l'étape 1)

GESTION DES RESSOURCES

Gestion du processeur

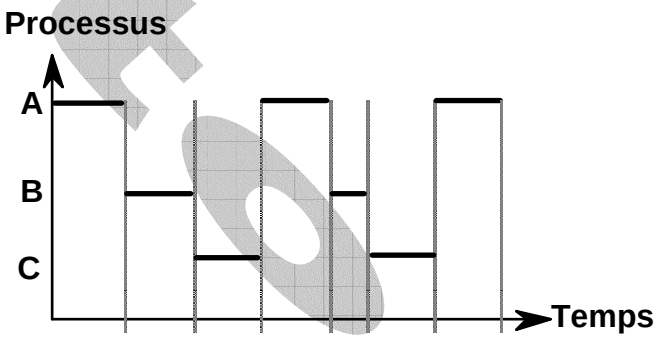
- Parallélisme

  n processus –> *m* processeurs



ici, n=3 et m=3

- Pseudo-parallélisme
  - *n* > *m*
    - à un instant donné, il n'y a que *m* processus actifs au maximum



ici n>=3 et m=1

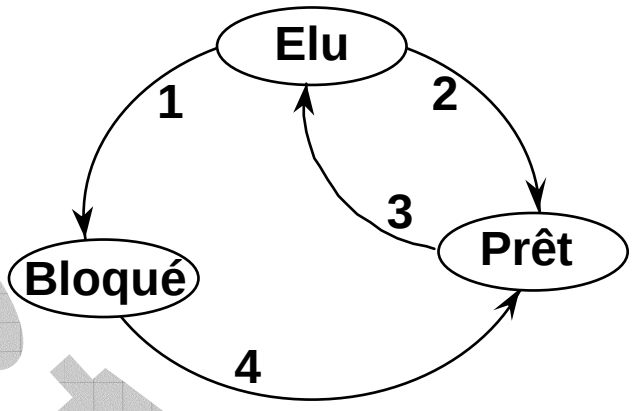
GESTION DES RESSOURCES

Gestion du processeur

- Etats d'un processus
  - Elu ou actif (running): en cours d'exécution
  - Bloqué (blocked): en attente de données

    => exécution impossible
  - Prêt (ready): en attente du processeur

- Transitions entre les états



1. Les données ou ressources nécessaires ne sont pas disponibles
2. Le système choisit un autre processus
3. Le système choisit ce processus
4. Les données ou ressources deviennent disponibles

GESTION DU PROCESSEUR

Ordonnancement des processus

- Ordonnanceur *(scheduler)*
  - choix du processus à exécuter selon *la stratégie (ou algorithme) d'ordonnancement* en vigueur.
- Objectifs de la stratégie d'ordonnancement

1. Equité
2. Efficacité
3. Temps de réponse : minimiser le temps de réponse pour les utilisateurs en mode interactif sans pénaliser les utilisateurs qui travaillent en traitement par lots.
4. Bon taux d'utilisation des ressources
5. Equilibre entre temps de réponse et taux d'utilisation des ressources
6. Dégradation progressive sous forte charge

Certains des objectifs sont contradictoires

=> Aucune stratégie d'ordonnancement n'est idéale

GESTION DU PROCESSEUR

Stratégies d'ordonnancement

DEUX FAMILLES DE STRATEGIES

- Stratégie sans réquisition du processeur*(non preemptive scheduling)*

  Un processus actif garde le processeur jusqu'à son achèvement.

  Avantages:

  - simple
  - peu d'overhead

  Inconvénients:

  - les processus courts sont retardés par les processus longs
  - mauvaise utilisation des ressources

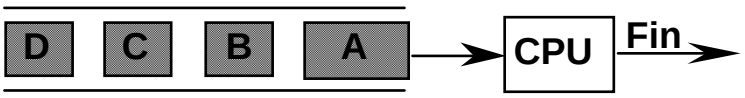
- Stratégie est avec réquisition du processeur *(preemptive scheduling)*

  Le processeur peut être retiré à un processus avant sa terminaison

GESTION DU PROCESSEUR

Stratégies d'ordonnancement

- Premier arrivée, premier sorti
  - les travaux sont exécutés dans leur ordre d'arrivée
  - sans réquisition



Avantages:

- très simple
- overhead minimal

Inconvénients:

- les travaux courts sont pénalisés

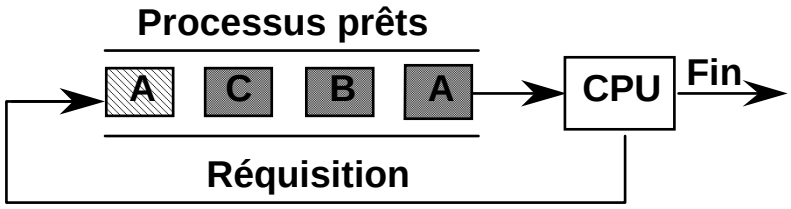
GESTION DU PROCESSEUR

Stratégies d'ordonnancement

- Tourniquet *(round robin)*

  - les processus prêts sont gérés dans une file FIFO
  - allocation du processeur pour un temps limité

    => quantum *(time-slice)*



  - efficace dans les environnements temps partagé où il est nécessaire de garantir des temps de réponses raisonnables
  - faible overhead

GESTION DE LA MEMOIRE

Hiérarchie des mémoires

- Mémoire secondaire
  - magnétique (bande, disque, disquette)
  - optique (CD laser).

    => Exploitée par les périphériques

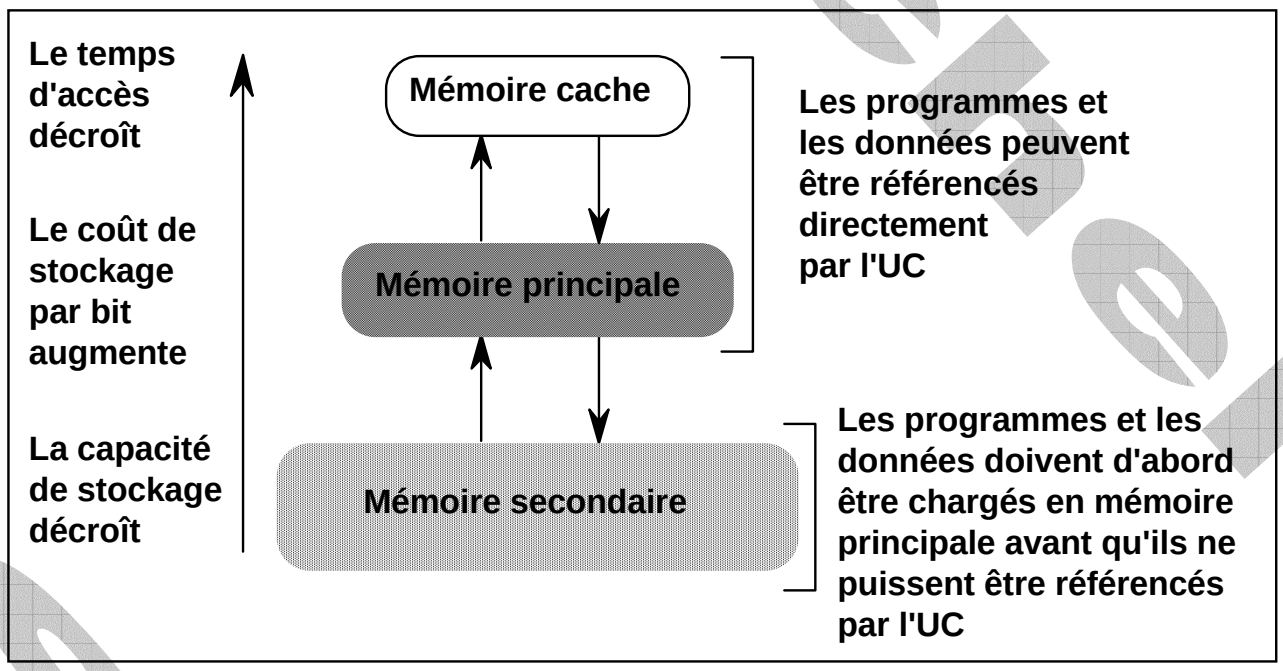
- Mémoire principale: circuits électroniques

- Mémoire cache: circuits électroniques à temps d'accès très rapides (50 ns)
  - transparente aux programmes utilisateurs.
  - contient toujours les parties du programme et des données les plus utilisées.

    => Accélérer l'exécution des programmes.

GESTION DE LA MEMOIRE

Hiérarchie des mémoires



Le rapport des temps d'accès entre la mémoire cache et la mémoire secondaire est de 10^9 (50 s pour accéder à n'importe quel endroit d'une bande magnétique)

GESTION DE LA MEMOIRE

Organisation et gestion

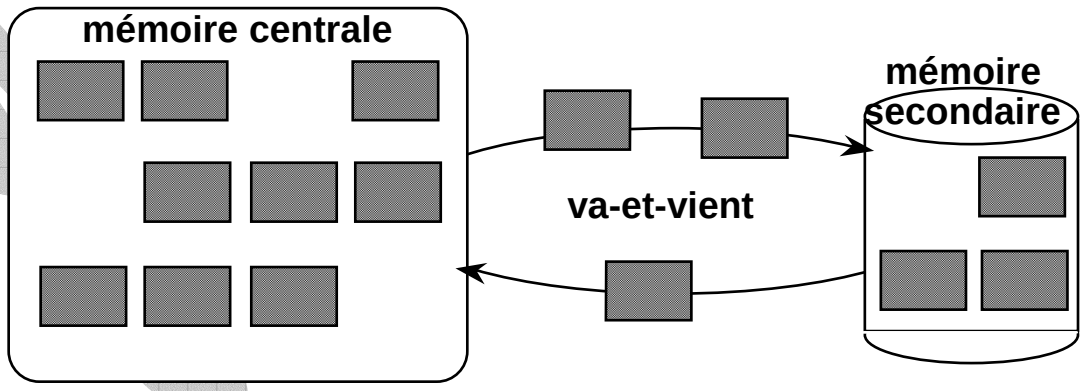
- L'organisation définit comment la mémoire est vue par le système d'exploitation :
  - un seul utilisateur à la fois ?
  - plusieurs utilisateurs à la fois ?
  - chaque utilisateur se voit-il alloué la même quantité de mémoire ?
  - partitionne-t-on la mémoire principale de manière rigide ou en fonction des besoins des processus utilisateurs ?
  - un processus s'exécute-t-il dans une partition spécifique ou peut-il s'exécuter n'importe où dans la mémoire ?
  - chaque processus doit-il être placé dans des blocs de mémoire contigus ou peuvent-ils être découpés en blocs placés n'importe où dans la mémoire ?

GESTION DE LA MEMOIRE

Organisation et gestion

- Gestionnaire de mémoire
  - obtenir des performances optimales de l'organisation

- Gestion avec va-et-vient
  - mémoire secondaire utilisée comme extension de la mémoire principale



- Gestion sans va-et-vient
  - Dépassée !!

GESTION DE LA MEMOIRE

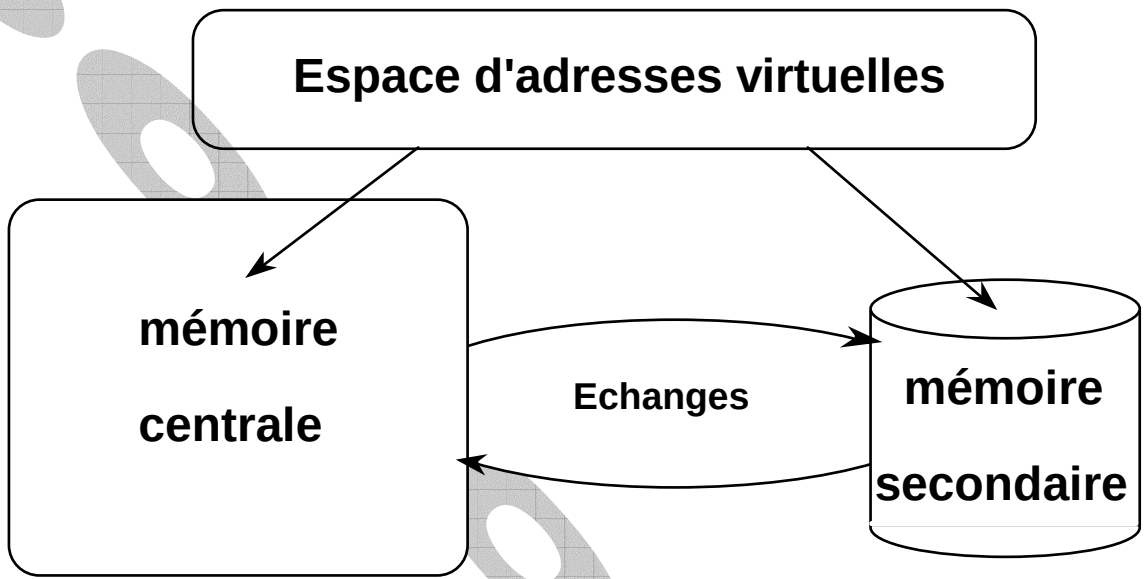
Mémoire virtuelle

- Propriétés
  - dissociation des adresses virtuelles et physiques
  - fractionnement de l'espace virtuel d'un processus en **blocs**
  - allocation non-contiguë

    => Espace virtuel et physique de tailles différentes

    => Seuls les blocs utiles sont en mémoire centrale

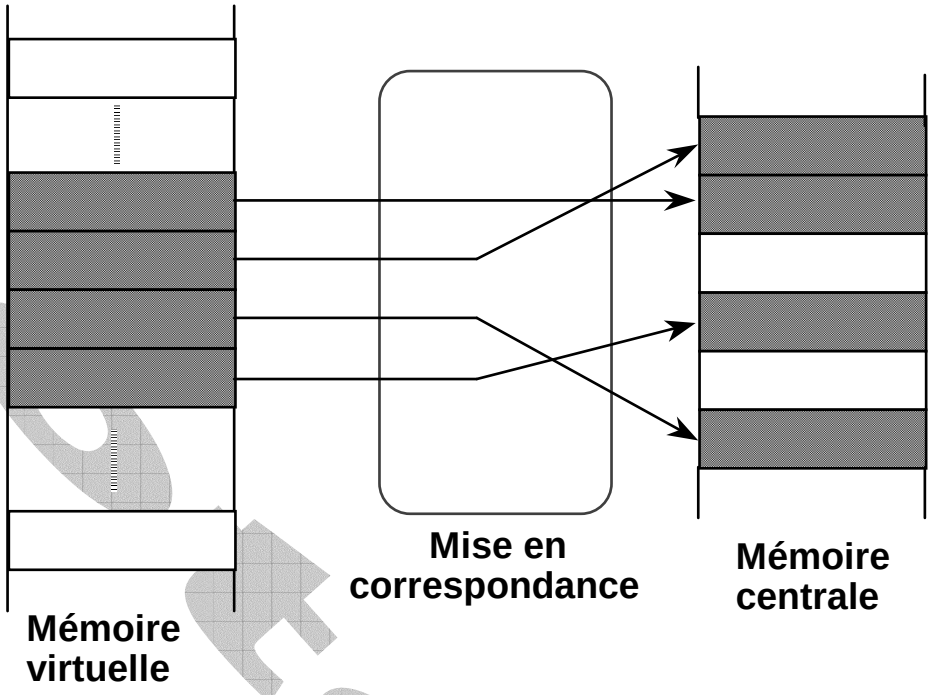
    => Mise en correspondance dynamique des adresses virtuelles avec les adresses physiques



Memoire virtuelle

Pagination

- Principe
  - l'espace d'adresses virtuelles est divisé en blocs de **taille fixe** appelés **pages**
  - la mémoire centrale est divisée en blocs de même taille que les pages appelés **cases mémoire**



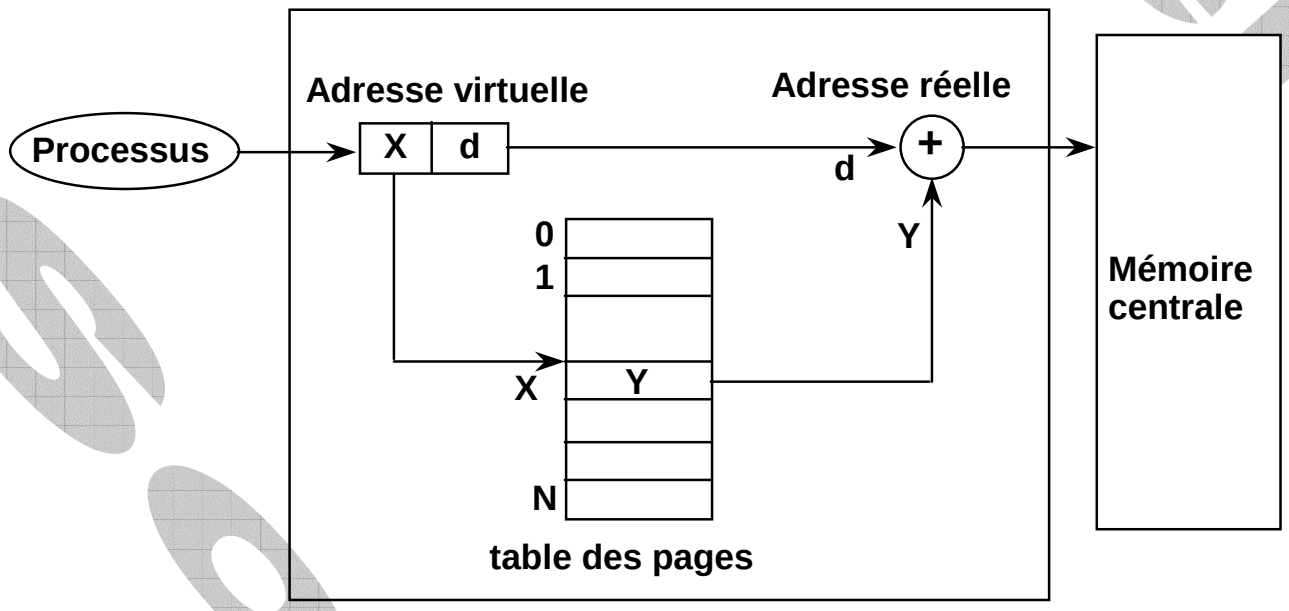
Memoire virtuelle

Pagination

- Correspondance adresse virtuelle → adresse réelle
  - adresse virtuelle:

    (**X**: n° de page, **d**: déplacement dans la page)

  Cette correspondance est assurée par un dispositif matériel spécifique



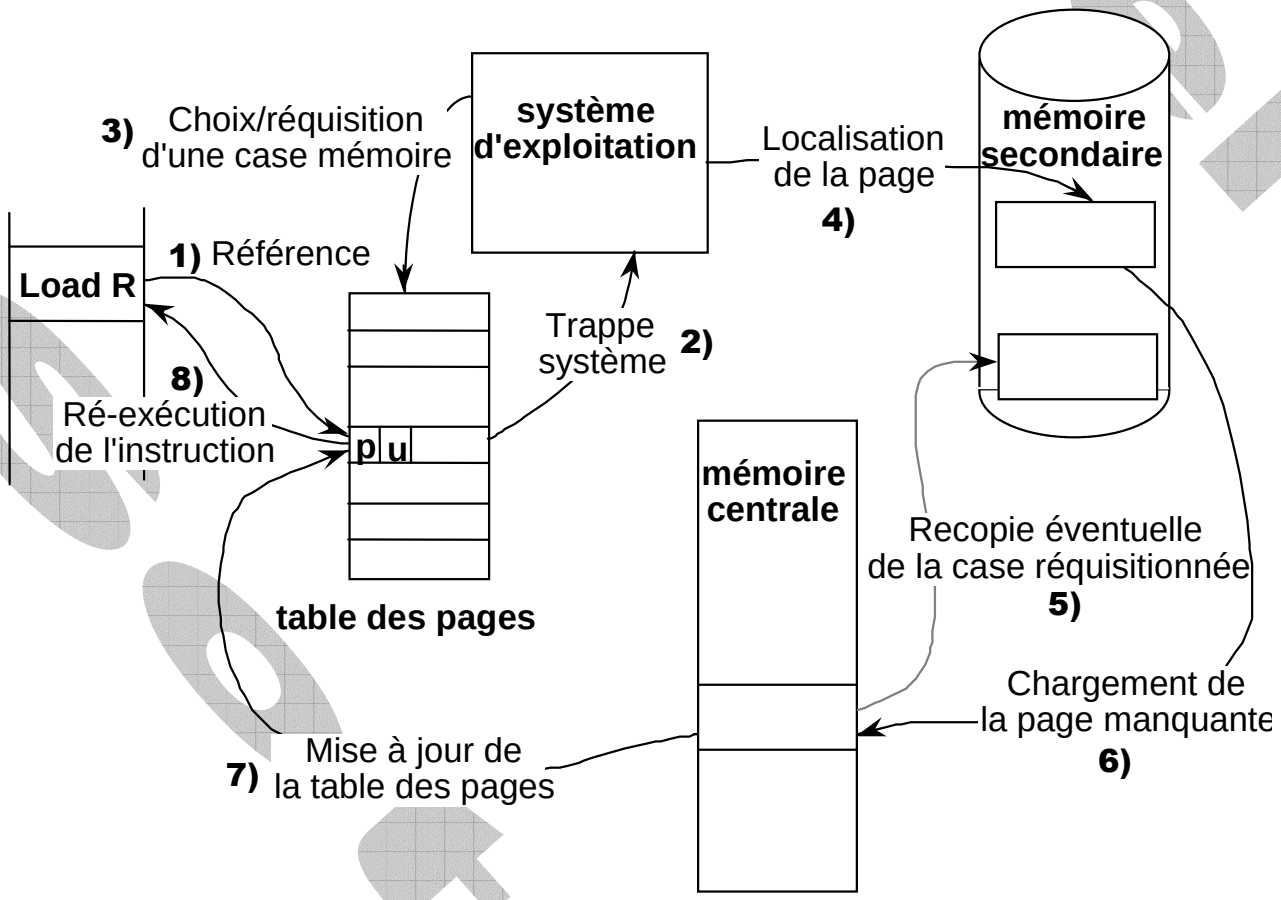
- Table des pages
  - ensemble de registres ou - mémoire centrale
  - une table par processus

Memoire virtuelle

Pagination

- Défaut de page

  Un processus référence une page qui n'est pas en mémoire centrale



GESTION DE LA MEMOIRE

Stratégies de remplacement

- Remplacement optimal

  Le gestionnaire de mémoire réquisitionne la page qui sera référencée dans le futur le plus lointain.

Caractéristiques:

- Méthode optimale
- Irréalisable puisqu'il faut prévoir l'avenir

  ➡ Stratégie de référence

- Remplacement aléatoire

  Lorsqu'un défaut de page se produit, le gestionnaire de mémoire choisit au hasard la page à remplacer.

Caractéristiques:

- faible overhead
- équitable
- risque de choisir la prochaine page référencée
- rarement utilisée du fait de son aspect "quitte ou double"

GESTION DE LA MEMOIRE

Stratégies de remplacement

- Première arrivée-première remplacée *(FI FO)*

  Le gestionnaire de mémoire choisit de remplacer la page qui est dans la mémoire depuis le plus long temps.

Caractéristiques:

- faible overhead
- équitable
- risque de choisir la prochaine page référencée
- rarement utilisée

GESTION DE LA MEMOIRE

Stratégies de remplacement

- Page la moins récemment utilisée *(LRU)*

  Le gestionnaire de mémoire choisit de remplacer la page qui est resté inutilisée depuis la plus longue période de temps.

  Chaque page est estampillée chaque fois qu'elle est référencée.

Caractéristiques:

- bonne approximation de la stratégie optimale
- overhead non négligeable
- rarement utilisée, on essaie d'approximer cette méthode

GESTION DE LA MEMOIRE

Stratégies de remplacement

- Page la moins fréquemment utilisée
  - approximation de la stratégie LRU
  - un compteur est associé à chaque page et incrémenté chaque fois que la page est référencée.
  - le gestionnaire de mémoire remplace la page dont le compteur contient la valeur la plus faible.

Caractéristiques:

- cette stratégie n'oublie rien: des pages qui ont été fortement référencées seront gardées en mémoire alors qu'elles sont devenues inutiles.

GESTION DE LA MEMOIRE

Stratégies de remplacement

- Page non récemment utilisée
  - approximation des plus populaire de la stratégie LRU
  - deux bits sont associés à chaque page:

    Bit de référence: 0, page non référencée

    1, page référencée

    Bit de modification: 0, page non modifiée

    1, page modifiée
  - le matériel repositionne périodiquement tous les bits de référence à zéro
  - 4 groupes de pages:

    Groupe 1: non référencée, non modifiée

    Groupe 2: non référencée, modifiée

    Groupe 3: référencée, non modifiée

    Groupe 4: référencée, modifiée
  - Le gestionnaire de mémoire cherche d'abord la page à remplacer dans le groupe 1 puis 2, etc.