Notes SE

Chapitre 1 : Filesystems

- Fichier = Unité logique de stockage
- Deux types de fichiers :
    - **Fichiers réguliers (text,bin,executable)**
    - **Fichiers spéciaux (dir, dev)**
- *On peut visualiser le type de fichier sous unix avec **file***
- Attributs de fichiers : taille - dates création modif accès - propriétaire - droits d'accès - types
- Structures internes de fichiers :
    - **Seq d'octets**
    - **Seq d'enregistrements (enregistrements de long fixe)**
    - **Arborescence d'enregistrements (enregistrements de long var)**
- Structures logique :
    - **Plate ou à un niveau (Racine - Fichiers)**
    - **À deux niveaus (Racine - User Dir - Fichiers)**
    - **Multi-niveaux**
- Montage :
    - Intégration d'un système de fichiers spécial - permettre et faciliter l'accès aux données qui se trouve dans le système "monté"
    - Point de montage: est un répertoire à partir duquel sont accessibles les données se trouvant dans le système de fichiers qui a été intégré
    - Le montage se fait sur des répertoires vides
    - Si le point de montage contient déjà des fichiers, ces derniers deviennent inaccessibles jusqu'au démontage
- Méthodes d'accés à un fichier
    - Sequentiel (Bande magnétique)
    - Direct (CD - Flash)
- Méthodes d'accés au contenu
    - Seq (Compilateur * imprimante)
    - Direct (BD)
- Disque dur physiquement
    - Ensemble de **plateaux** formés de **pistes** formés de **secteurs**
- Disque dur logiquement
    - Ensemble de blocs
    - Bloc 0 contient le MBR Master Boot Record
- Méthodes d'allocation :

Allocation Contiguë

Mise en œuvre :

L'entrée à un Fichier est structurée de la façon suivante :

Nom Fichier	Attributs	Adresse de début	Longueur (nombre de blocs)

Avantages

+ Simple

+ Accès direct et accès séquentiel facile

Inconvénients

– Nécessité de connaitre en avance la taille de Fichier !!!

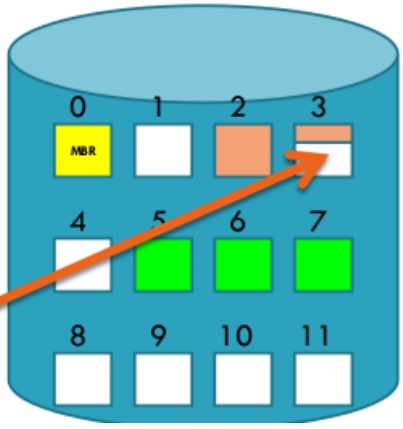
– Fragmentation externe

– Fragmentation interne

F1 de taille 900 Ø

Taille bloc : 512 Ø

Fragmentation interne = 2*512 – 900 = 124 Ø



K. ElBedoui-Maktouf

- **La fragmentation externe peut etre corrigé avec le compactage (defrag)**

Allocation Chaînée

Mise en œuvre :

L'entrée à un Fichier est structurée de la façon suivante :

Nom Fichier	Attributs	Adresse de début

Avantages

+ Pas de fragmentation externe

+ Pas de problème de taille de Fichier (il n'est pas nécessaire de connaître la taille de Fichier en avance).

Inconvénients

– Fragmentation interne

– Accès direct impossible

– Accès séquentiel lent

– La perte d'un pointeur engendre la perte du Fichier

Mélange entre donnée utilisateur et donnée système

Données

Allocation chaînée avec table FAT

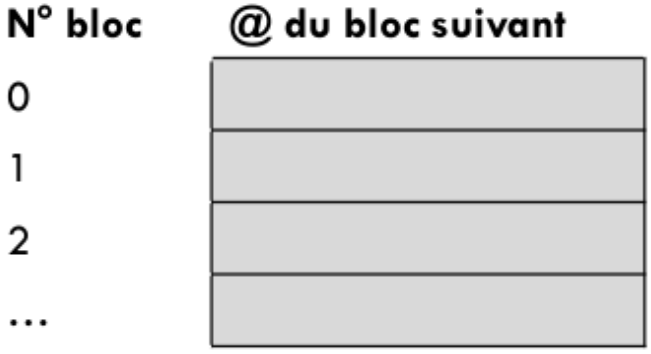
Principe :

Le chaînage entre les blocs d'un fichier est stocké dans une table et non pas au niveau du bloc

La table est dite FAT (File Allocation Table).

Principe :

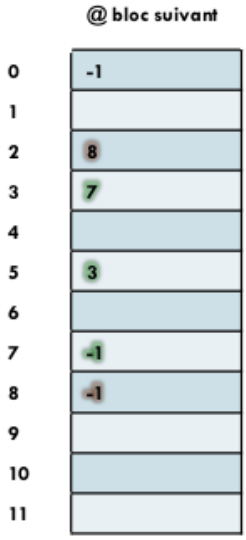
La table FAT



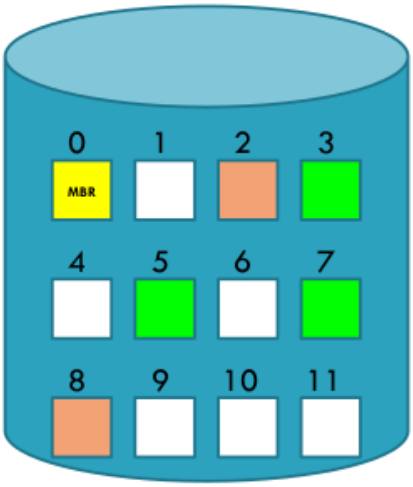
Exemple :

Nom	Att	début
F1	...	2
F2	...	5

Répertoire A



FAT



. ElBedoui-Maktouf

❁ Allocation chaînée avec table FAT

Avantages

+ Pas de fragmentation externe

+ Pas de problème de taille de Fichier (il n'est pas nécessaire de connaître la taille de Fichier en avance).

+ pas de mélange entre donnée utilisateur et donnée système

Inconvénients

– Fragmentation interne

– Retour régulier à la table FAT

– Accès séquentiel et direct lent

– Consommation de l'espace mémoire pour le stockage de FAT (et ceci quelque soit le nombre de Fichiers)

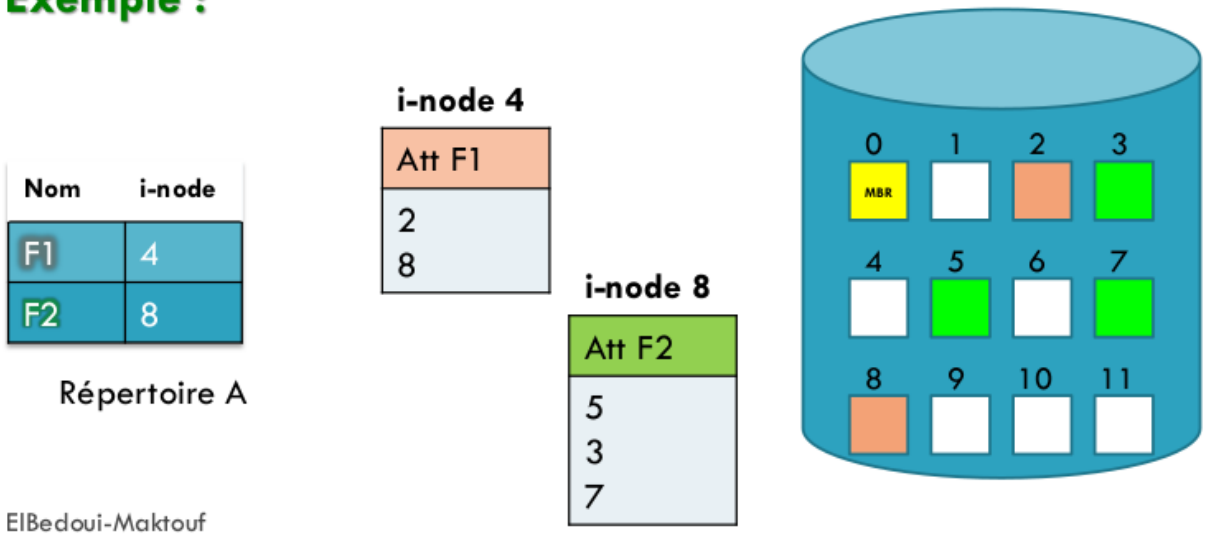
❁ Allocation indexée

Mise en œuvre :

L'entrée à un Fichier est structurée de la façon suivante :

Nom Fichier	Adresse de l'i-node

Exemple :



K. ElBedoui-Maktouf

- **Sous Unix, pour connaître le numéro d'i-node d'un Fichier, on utilise la commande ls -i**

Avantages

+ Pas de fragmentation externe

+ Pas de problème de taille de Fichier (il n'est pas nécessaire de connaître la taille de Fichier en avance).

+ pas de mélange entre donnée utilisateur et donnée système

Inconvénients

– Fragmentation interne (au niveau de l'i-node et du Fichier)

– Retour régulier à l'i-node

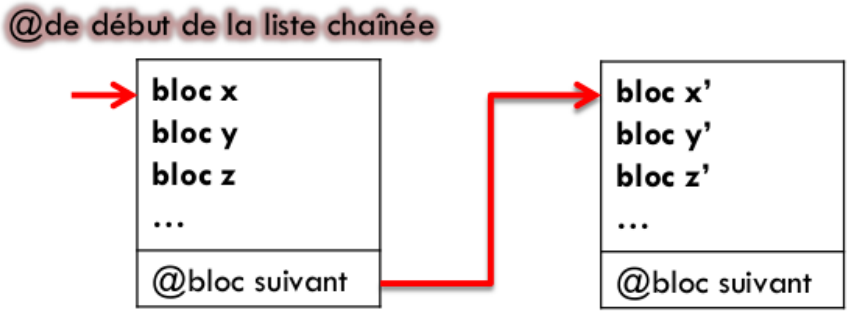
– Accès séquentiel et direct lent

– Consommation de l'espace mémoire pour le stockage des i-nodes

- Gestion de l'espace libre :
  - **Méthode statique :**
    - **Table de bits.. Chaque bit fait réf à un bloc si bit = 1 bloc libre**
  - **Méthode Dynamique :**
    - **Liste chainée de <u>blocs</u>**

Se base sur l'utilisation d'une liste chaînée de blocs

spéciaux qui contiennent les numéros des blocs libre



- **<u>CAS DE LINUX :</u>**

  - **Bloc 0 : MBR**
  - **Super Bloc : Bloc 1 : @deb de la liste de blocs libre - nb blocs libre - taille de table inode - nombre de blocs…**
  - **Table de inodes de 1 à max inode :**
    - **Inode 1 liste des blocs defectueux**
    - **Inode 2 racine**

- **INODE :**

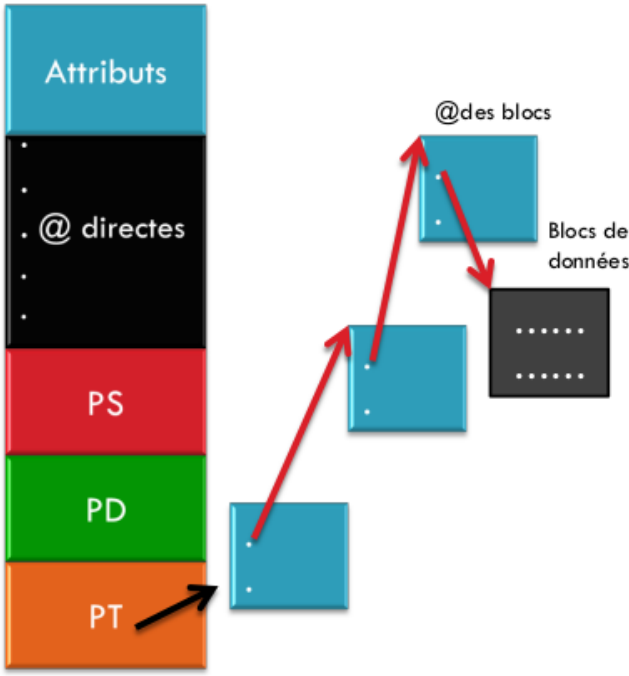
Structure d'un i-node

Si les adresses directes ne sont pas suffisantes pour lister les adresses des blocs des données qui composent le Fichier, on peut utiliser :

un Pointeur d'indirection Simple

un Pointeur d'indirection Double

un Pointeur d'indirection Triple



- **CAS DE WINDOWS**
  - **FAT**

    - **Cluster = Ensembe de blocs adjacents**
  - **NTFS (New Technologie File System)**
    - **Utilise la table MFT (Master File Table) depuis Windows NT**

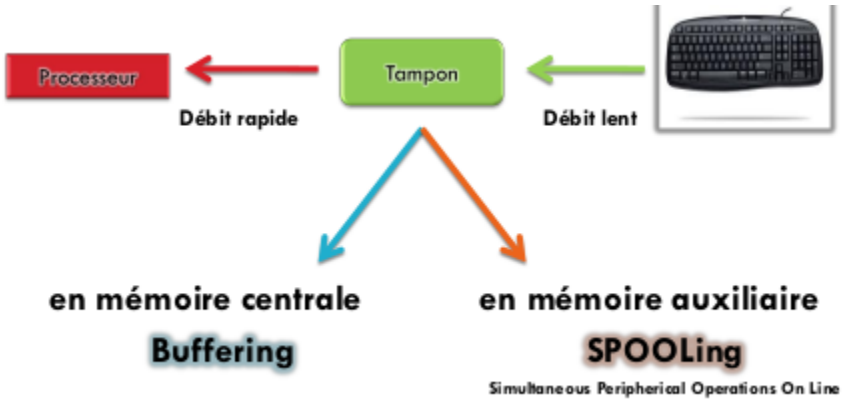
Chapitre 2 : E/S

Aspects Matériels :

- Péripherique :
  - En bloc : blocs de taille fixe - chaque bloc est adressable (Disque dur - CD)
  - En caractére : flot non adressable (Ecran - imprimante)
- Constroleur :
  - Carte electronique qui controle chaque péripherique
  - A son microprocesseur, tampon et registre

    - Registres d'etat : etat du périphérique
    - Registre de controle : permet d'initier une commande
    - Tampon d'entrée : recupérer des données
    - Tampon de sortie : transmettre des données
- Unités d'échange :

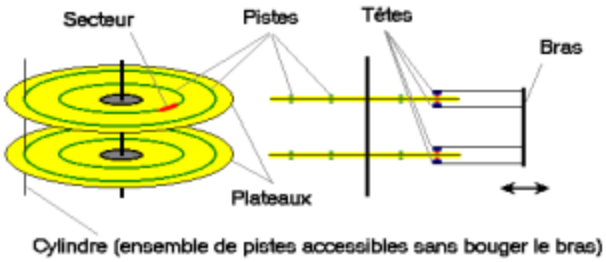
- Adressage d'un péripherique : un périphérique est designé par un adresse (Num controleur - Num periphérique) :
    - Num majeur : designe le type de péripherique (controleur) : 4 terminaux, 3 HDD, 6 imprimantes
    - Num mineur : désigne le periphérique : pour les imprimantes : 0 lp0 et 1 à lp1
    - Ls -l /dev
- Tamponnage : resoudre la différence de vitesse entre le processeur et les organes peripheriques



Aspectes logiciels :
3 maniéres de transfert des E/S
- Par scrutation
- Par interruption
- Par DMA Direct Access Memoy

Driver (Pilote) :
- Logiciel qui commende le fonctionnement elementaire d'un périphérique
- Contient :
    - Une procédure traitant l'initialisation d'un transfert
    - Une procédure de traitement de l'interruption associée à une fin de transfert

Cas du Disque Dur :



- Adressage des blocs :
  - Cylinder/Head/Sector CHS :

Cylindre : piste (commence de 0)
Tête : surface (commence de 0)
Secteur : (commence de 1)

NC : nombre total de cylindre
NH : nombre total de tête
NS : nombre total de secteur par piste

Nombre total de secteur sur un disque est : NC*NH*NS

  - Logical Block Adressing LBA :

Adresse linéaire

$$AL = (C \times NH \times NS) + (H \times NS) + S - 1$$

Conversion inverse

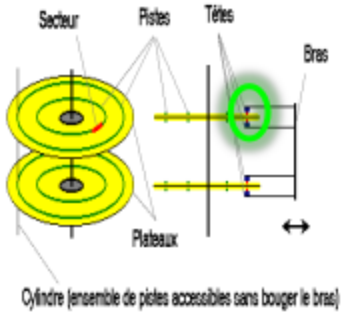
$$S = (AL \ \% \ NS) + 1$$
$$H = (AL - S + 1) / NS \ \% \ NH$$
$$C = (AL - S + 1) / NS \div NH$$

- Paramétres de performance :

1. **Seek Time :**
temps de déplacement de bras du disque au cylindre demandé

2. **Latency Time :**
temps de rotation de l'axe du disque au secteur demandé

3. **Transmission Time :**
Temps nécessaire pour effectuer une opération e lecture ou écriture



- Algorithme FCFS : First come first served

Requêtes	40	10	60	0	80	30	70	50

Ordre	15	40	10	60	0	80	30	70	50
Coût	25	30	50	60	80	50	40	20	355

- Algorithme SSTF : Shortest Seek Time First

Requêtes	40	10	60	0	80	30	70	50

Ordre	15	10	0	30	40	50	60	70	80
Coût	5	10	30	10	10	10	10	10	95

- Algorithme SCAN : Ascenseur

Requêtes	40	10	60	0	80	30	70	50

Ordre	15	30	40	50	60	70	80	10	0
Coût	15	10	10	10	10	10	70	10	145

- Le temps d'accés sur le disque = S + L + T
  - **S : Seek Time =**

I (Initiation) + C(Nombre de cylindres à parcourir) * H(Temps de passage entre deux cylindres)

  - **L : Latency Time = en secondes**

30 / R (rpm)

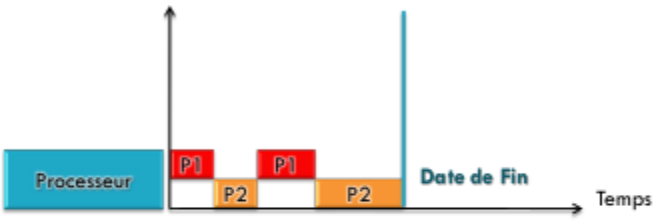
  - **T : Transmission Time = en ms**

(60 * B(Volume à lire en octets)) / (R*N(Nbr octets dans une piste))

Chapitre 4 : Processus

Définition :

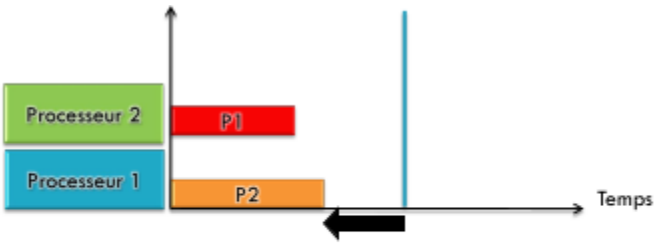
- Programme séquentiel en cours d'exécution
- Le programme statique <> processus dynamique
- Chaque exécution du même programme engendre un processus différent
- Système monoprocesseur :



- À un instant donné un seul processus tourne
- Sur un intervalle de temps important tous les processus ont fait leurs exécutions

C'est le pseudo-parallélisme

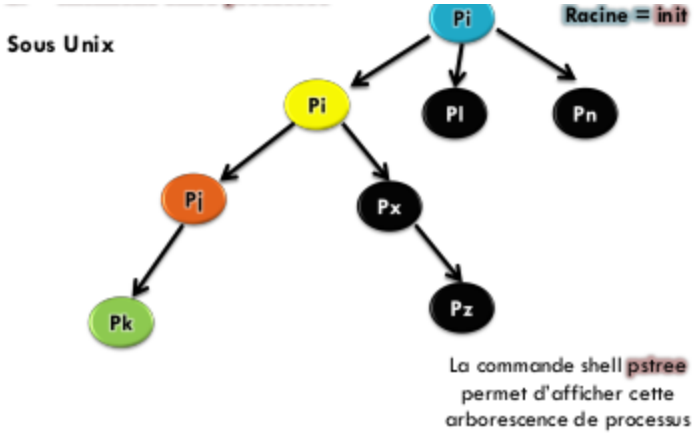
- Système multiprocesseur :



Sur un système multiprocesseurs

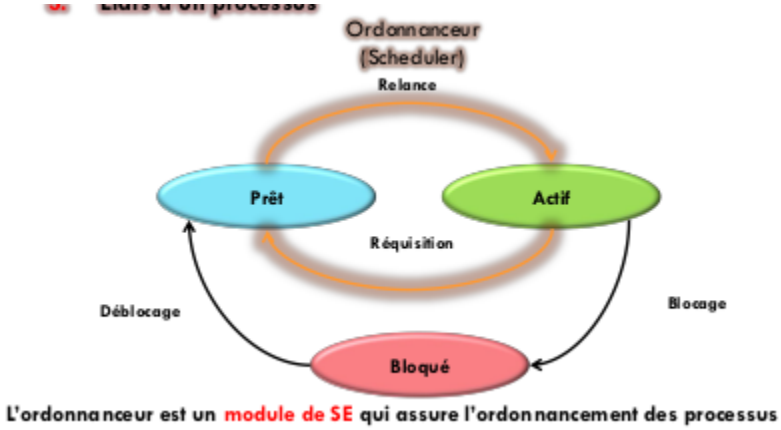
C'est le parallélisme

Relations entre processus :



- Racine = Ancétre
- Les processus peuvent être:
    - en coopération : chacun réalise une partie du travail
    - En compétition : pour les ressources partagés (processeur, memoire, fichier)

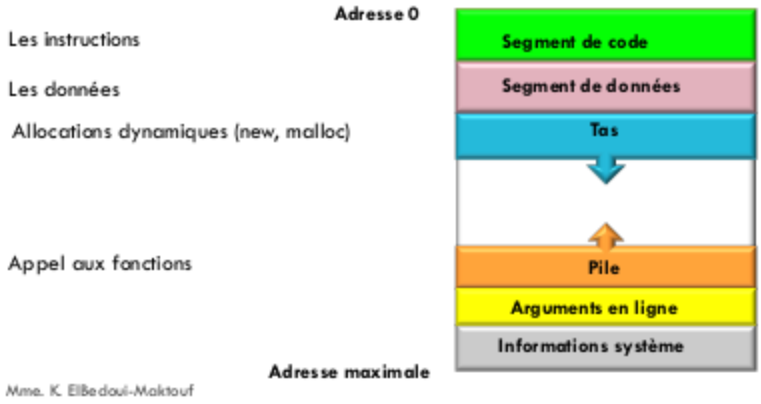
Etats d'un processus :



- Actif : En cours sur le processeur
- Pret : Attend la libération du processeur
- Bloqué : Attend un événement autre que la libération du processeur

Structure du processus :

Le processus dispose de son propre espace d'adressage

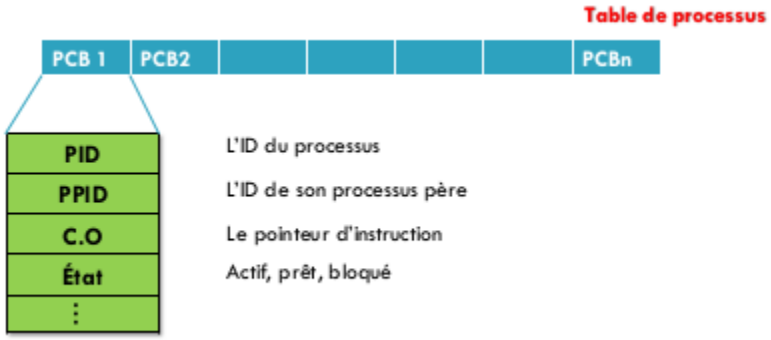


Le processus est géré via une structure de donnée dite

PCB (Process Control Bloc) qui est une structure de données

Le PCB contient les informations nécessaires pour la relance d'un processus

Les PCB sont rangés dans une table (dite table des processus) qui se trouve dans l'espace mémoire du système



- Le passage d'un processus à un autre engendre :
  - Sauvegarde du contexte actuel (PCB du process actif)
  - Chargement d'un nouveau contexte (PCB d'un autre process)
- La commutation se fait par
  - Interruption
  - Déroutement

a) Interruption (it)

Signal émis par un dispositif externe au processeur.

Le traitement d'une interruption se fait en se référant à un **vecteur d'interruptions** et en tenant compte de sa **priorité**

- Indexé par un numéro de périphérique
- Indique pour chaque it l'adresse de la fonction de traitement correspondante

it	@ Routine
0	@ fonction 0
1	@ fonction 1
2	@ fonction 2
…	…

Mme. K. ElBedoui-Maktouf

a) Interruption (it)

Étapes de traitement d'une it :

a) Sauvegarde du processus courant

b) Passage en mode noyau

c) Identification de l'it

d) Chargement de la routine correspondante

e) Relance de processus interrompu

it	@ Routine
0	@ fonction 0
1	@ fonction 1
2	@ fonction 2
…	…

Mme. K. ElBedoui-Maktouf

Sous Unix, on distingue les interruptions suivantes :

it	Cause
0	Horloge
1	Disque
2	Terminaux
3	Périphériques
4	Logiciels (exp. trap)
5	Autres

Engendré par le processus actif et ce suite à une erreur d'exécution :

- Débordement d'un tableau
- Instruction incorrecte (exp. division par zéro)
- Violation d'une protection
- ...

Le traitement d'un déroutement ne peut pas être différé (pas de mise en attente possible

Le traitement d'un déroutement est toujours plus prioritaire que celui d'une interruption

Interruption (it) Vs Déroutement (dt)

it	dt
Externe au processeur	Interne au processeur
Attente possible	Traitement immédiat
Traitée sur n'importe quel processeur (cas multiprocesseurs)	Traité par le processeur concerné
Niveaux de priorité	Pas de priorité

Ordonnencement :

- La gestion de la concurrence des processus prêts pour l'obtention de processeur est effectuée par l'ordonnanceur.
- Un algorithme d'ordonnancement doit assurer :
  - Équité : chaque processus doit avoir sa part du temps processeur
  - Efficacité : utiliser le temps processeur à 100%
  - Rendement : Augmenter le nombre de travaux effectués par UT
- Il existe deux types d'algo d'ordonnancement :
  - Nom préemptif (sans réquiistion): Un processus est exécuté sans interruption :
  - Préemptif (avec réquisition) : l'ordennenceur peut interrompre l'execution d'un processus pour allouer le cpu à un autre process

Il existe deux type d'algorithme d'ordonnancement :

- Ordonnancement non préemptif (sans réquisition)
  - FCFS (First Come First Served)
  - SJF (Shortest Job First)

- Ordonnancement préemptif (avec réquisition)
  - RR (Round Robin)
  - Priorité

- Paramètres de performance

× Temps de Réponse d'un processus = date de sa fin – date de son arrivée

× Temps de Réponse Moyen (TRM) = Σ TR des processus/Nombre de processus

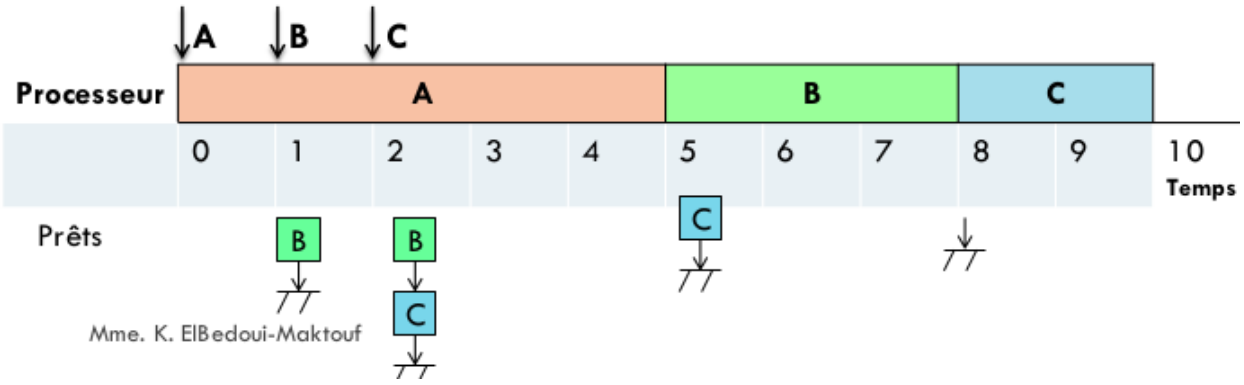
× Temps d'Attente d'un processus = TR du processus – sa durée

× Temps d'Attente Moyen (TAM) = (Σ TR - Σ durée)/ Nombre de processus

× Nombre de commutation de contexte = nbre changement de contexte

- **FCFS**

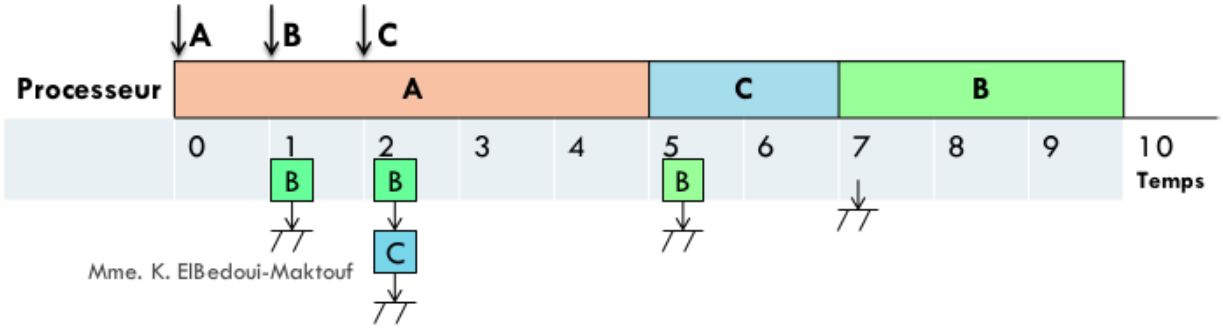
Processus	Durée	Date d'arrivée
A	5	0
B	3	1
C	2	2



3. Algorithmes

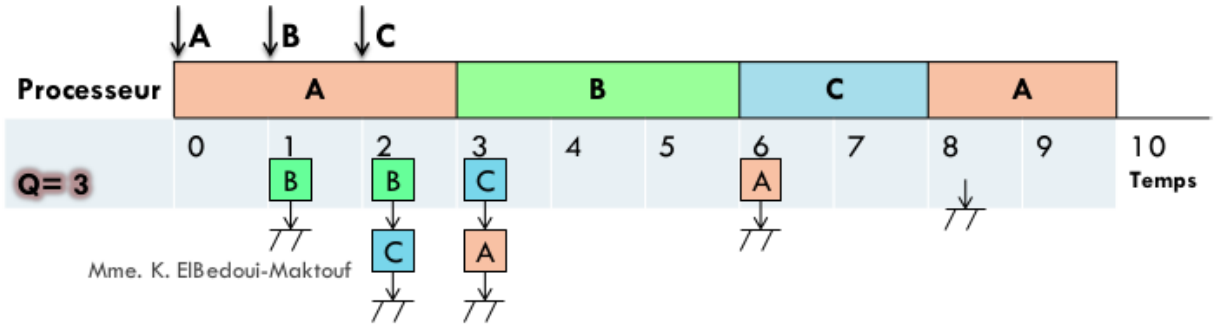
- **SJF**

Processus	Durée	Date d'arrivée
A	5	0
B	3	1
C	2	2



- **RR (Round Robin)**

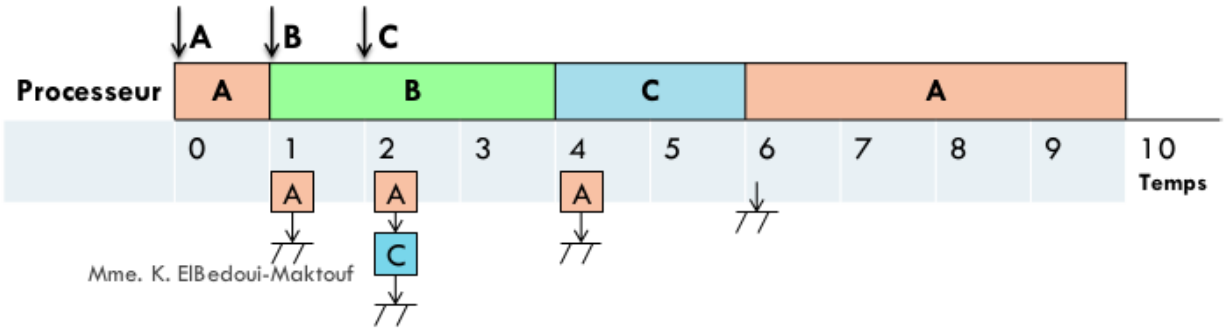
Processus	Durée	Date d'arrivée
A	5	0
B	3	1
C	2	2



- **Priorité**

Statique

Processus	Durée	Date d'arrivée	Priorité ↗
A	5	0	1
B	3	1	3
C	2	2	2



Chapitre 5 : RAM

Chaque processus a un espace d'adressage dans lequel il fait ses accès aux instructions et aux données

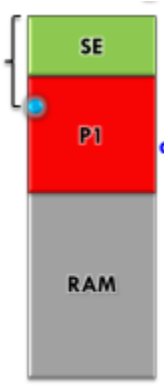
L'accès à une instruction ou à une donnée se passe en précisant sa **référence** (position) dans cet espace

Adresse logique **est une référence par rapport au processus lui-même**

est indépendante de la position de processus en RAM

L'ensemble des adresses logiques d'un processus forme son

Espace d'adressage logique



Une fois placé en mémoire centrale, toute

adresse logique aura son correspondant en adresse physique

Ainsi, l'adresse physique désigne la position

en mémoire physique

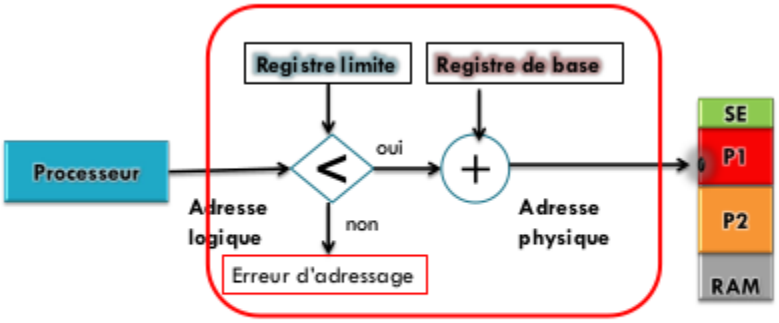
Autrement dit :

L'adresse physique est une référence par rapport à la mémoire physique (RAM)

Une fois chargé en RAM, un processus aura deux paramétres :

- Adresse de base (Physique)
- Taille limite

Le processeur utilise les adresses logiques.



MMU (Memory Manager Unit)

Ce dispositif qui fait la transformation d'adresse est dit MMU

Registre limite : contient la taille limite du processus actif -> indique la taille de son espace d'adressge.

Registre de base : contient l'adresse de base du process actif -> indique son adresse de début dans le RAM.

Gestion de la mémoire uniforme :

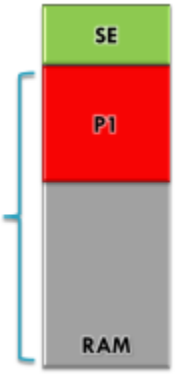
- **Monoprogrammation :**

a) **Sans recouvrement**

La gestion est faite sans recouvrement

Si la taille de processus ne dépasse pas

la taille de l'espace utilisable



b) **Avec recouvrement**

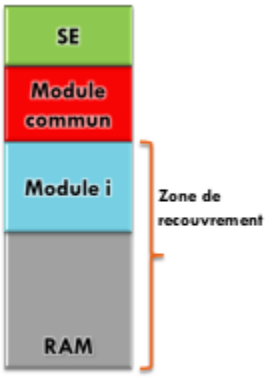
Si la taille de processus dépasse la taille

de l'espace utilisable

Alors le programmeur doit diviser son processus

en des modules (overlays) :

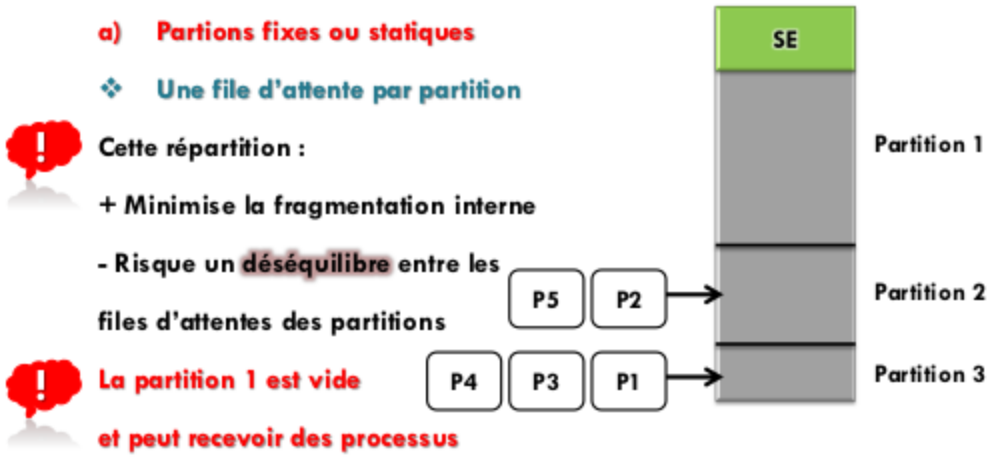
- module commun
- et des modules de recouvrements



Les modules de recouvrement sont chargés en RAM à la demande

- **Multiprogrammation :**
  - Partitions statique(nombre, tailles, localisation fixe) :
    Chaque partition peut recevoir au maximum un processus

    File d'attente par partition :
    Chaque processus est affecté à la partiton de la plus petite taille que possible et qui peut le contenir.



File d'attente commune : p41

Algorithmes de pagination :

Algorithmes de remplacement de page

OPT (OPTimal)

		0	1	0	2	0	1	3	0	2	3
Cadre 0		0	0	0	0	0	0_1				
Cadre 1			1	1	1	1	1_∞				
Cadre 2					2	2	2_2				
D.P		x	x	-	x	-	-				

Algorithmes de remplacement de page

FIFO (First in, First Out)

		0	1	0	2	0	1	3	0	2	3
Cadre 0		0	0	0	0	0	0	3	3	3	3
Cadre 1			1	1	1	1	1	1	0	0	0
Cadre 2					2	2	2	2	2	2	2
D.P		x	x	-	x	-	-	x	x	-	-

Algorithmes de remplacement de page

LRU (Least Recently Used) = moins récemment utilisée

Principe

- **Choisir la page la moins récemment référencée c.à.d. celle qui a restée non utilisée le plus de temps**
  - **Chaque page est alors indexée par la date du dernier accès**