

Université de Jijel
Faculté des Sciences exactes et Informatique
Département d'informatique





SYSTÈME D'EXPLOITATION I

Pour 2ème Année
Licence - informatique





PLAN DU COURS

- ❖ **Chapitre1**: Introduction aux système d'exploitation
 - ❖ **Chapitre2**: Mécanismes de base d'exécution des programmes
 - ❖ **Chapitre3**: Gestion des processus
 - ❖ **Chapitre4**: Gestion de la mémoire
 - ❖ **Chapitre5**: gestion des entrées/sorties
- 
- 

Chapitre 1

INTRODUCTION AUX SYSTÈME D'EXPLOITATION



Introduction

Le système d'exploitation d'un ordinateur ou d'une installation informatique est un ensemble de programmes qui remplissent deux grandes fonctions :

- ❑ gérer les ressources de l'installation matérielle en assurant leurs partages entre un ensemble plus ou moins grand d'utilisateurs
- ❑ assurer un ensemble de services en présentant aux utilisateurs une interface mieux adaptée à leurs besoins que celle de la machine physique



Définition

Le programme « système d'exploitation » est le programme fondamental des programmes systèmes. Il contrôle les ressources de l'ordinateur et fournit la base sur laquelle seront construits les programmes d'application.

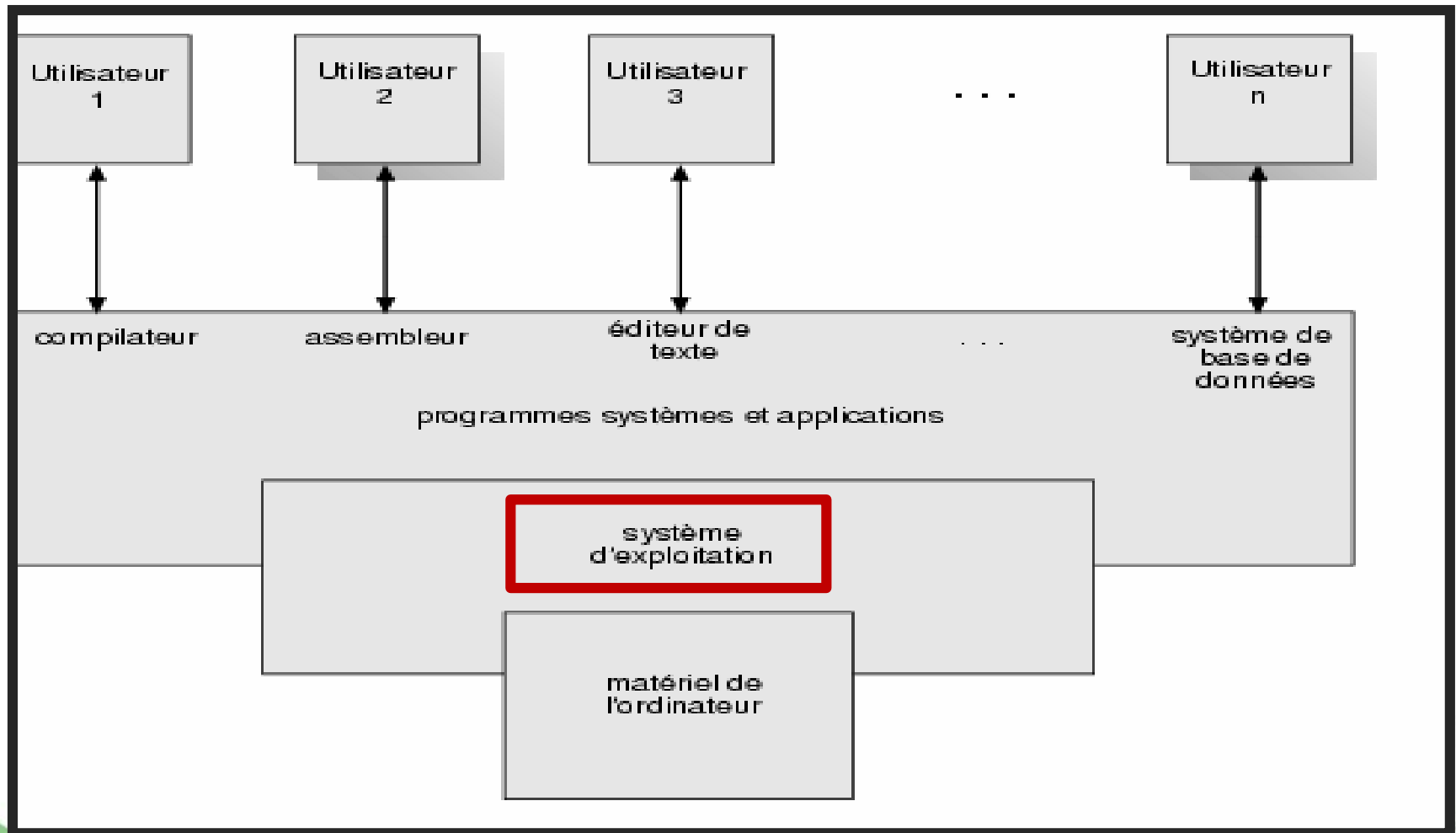


Les rôles d'un système d'exploitation

Le système d'exploitation (Operating System, O.S.) est l'intermédiaire entre un ordinateur (ou en général un appareil muni d'un processeur) et les applications qui utilisent cet ordinateur ou cet appareil. Son rôle peut être vu sous deux aspects complémentaires



Vue abstraite d'un SE



Les rôles d'un système d'exploitation

- Un système d'exploitation permet de répondre à deux besoins qui ne sont pas forcément liés :
 1. le système d'exploitation en tant que **machine étendue** (ou « machine virtuelle »),
 2. le système d'exploitation en tant que **gestionnaire de ressources**.



En tant que machine étendue

- ❑ Le système d'exploitation correspond à « l'interface » entre les applications et le matériel.
- ❑ De ce point de vue le système d'exploitation peut être assimilé à une machine étendue ou virtuelle plus facile à programmer ou à utiliser que le matériel :
 - Un programmeur va utiliser le système d'exploitation par l'intermédiaire " d'appels système ".



En tant que gestionnaire de ressources

- ❑ Les différents composants d'un ordinateur doivent coopérer et partager des ressources.
 - ❑ Dans cette optique, le travail du système d'exploitation consiste à :
 - ❖ ordonnancer,
 - ❖ contrôler l'allocation des ressources :
 - processeurs,
 - mémoires,
 - périphériques d'E/S,
 - ...
- entre les différents programmes qui y font **appel**.



Historique des systèmes d'exploitation

Pour comprendre ce que sont les SE, nous devons tout d'abord comprendre comment ils se sont développés. Cela permettra de voir comment les composants des SE ont évolué comme des solutions naturelles aux problèmes des premiers SE.



Porte ouverte ou exploitation self service (1945-1955)

Afin d'utiliser la machine, la procédure consistait:

- ❑ à allouer des tranches de temps directement aux Utilisateurs (programmeurs), qui se réservent toutes les ressources de la machine à tour de rôle pendant leur durée de temps.
- ❑ Les périphériques d'entrée/sortie en ce temps étaient respectivement le lecteur de cartes perforées et l'imprimante. Un pupitre de commande était utilisé pour manipuler la machine et ses périphériques.



Porte ouverte ou exploitation self service (1945-1955)

Chaque utilisateur, assurant le rôle d'opérateur, devait lancer un ensemble d'opérations qui sont :

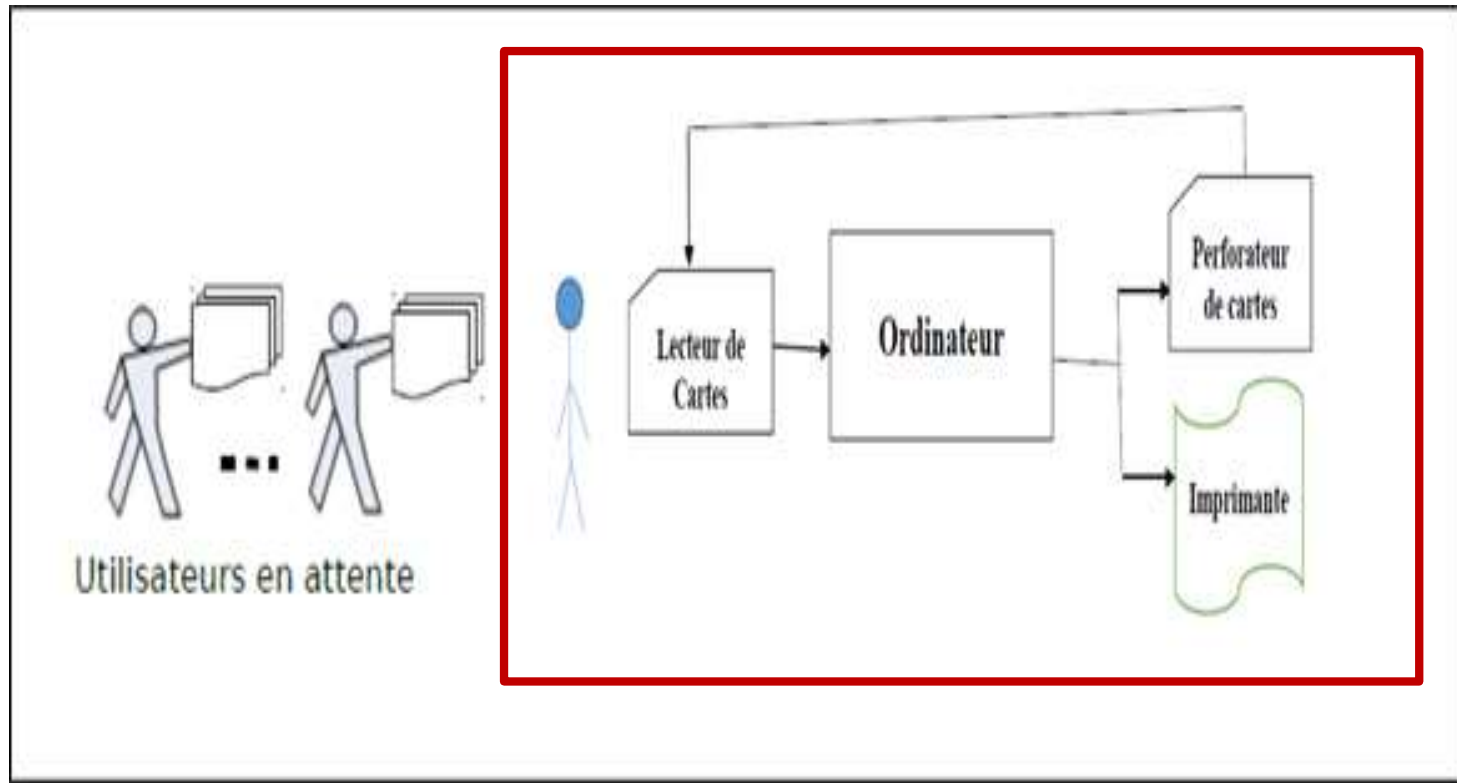
- Placer les cartes du programme dans le lecteur de cartes.

Initialiser un programme de lecteur des cartes.

- Lancer la compilation du programme utilisateur.
- Placer les cartes données s'il y en a, dans le lecteur de cartes.
- Initialiser l'exécution du programme compilé.
- Détecter les erreurs au pupitre et imprimer les résultats.



Porte ouverte ou exploitation self service (1945-1955)



Cartes perforées



Opérateur lisant un paquet de cartes perforées



Traitement par lots (Batch Processing) (1955-1965)

Ce sont des systèmes réalisant le séquençement des jobs ou travaux selon l'ordre des cartes de contrôle à l'aide d'un **moniteur d'enchaînement**. L'objectif était de réduire les pertes de temps .



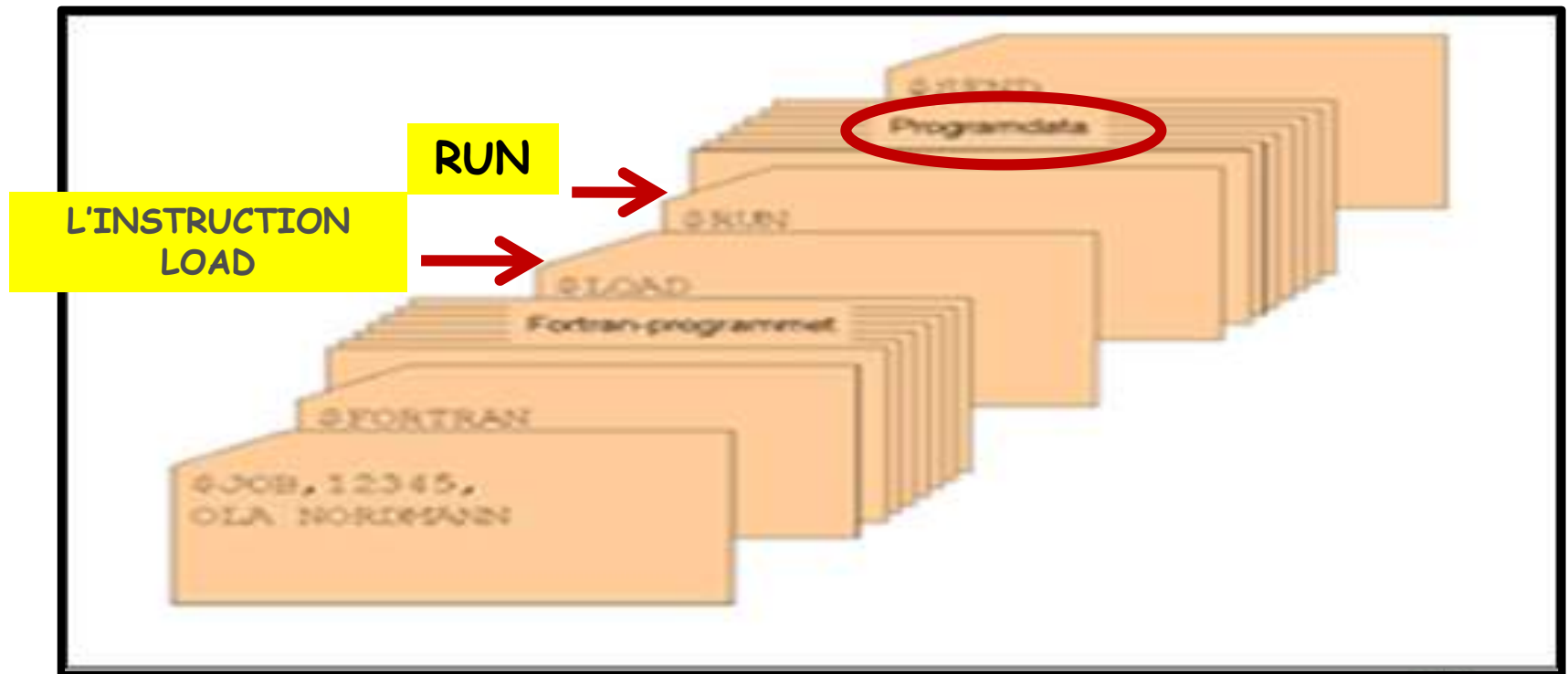
Traitement par lots (Batch Processing) (1955-1965)

Le programmeur soumet une **job** à un opérateur

- Programme suivi par données
- L'opérateur place un lot de plusieurs jobs sur le dispositif de lecture
- Un programme, **le moniteur**, gère l'exécution de chaque programme du lot
- Le moniteur est toujours en mémoire et prêt à être exécuté
- Les utilitaires du moniteur sont chargés au besoin
- Un seul programme à la fois en mémoire, programmes sont exécutés en séquence



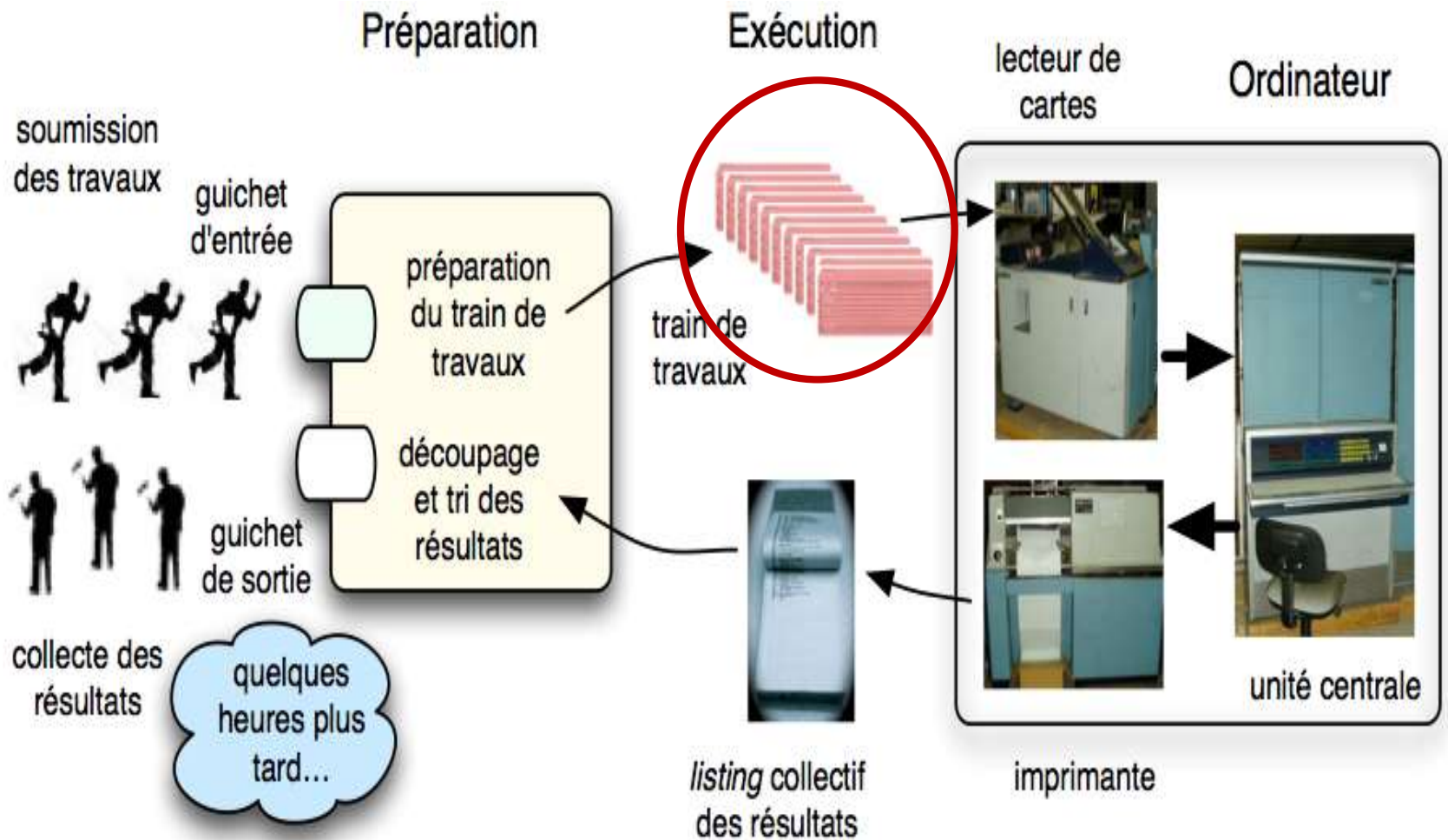
Traitement par lots (Batch Processing) (1955-1965)



Structure d'un travail FMS typique



Traitement par lots (Batch Processing) (1955-1965)



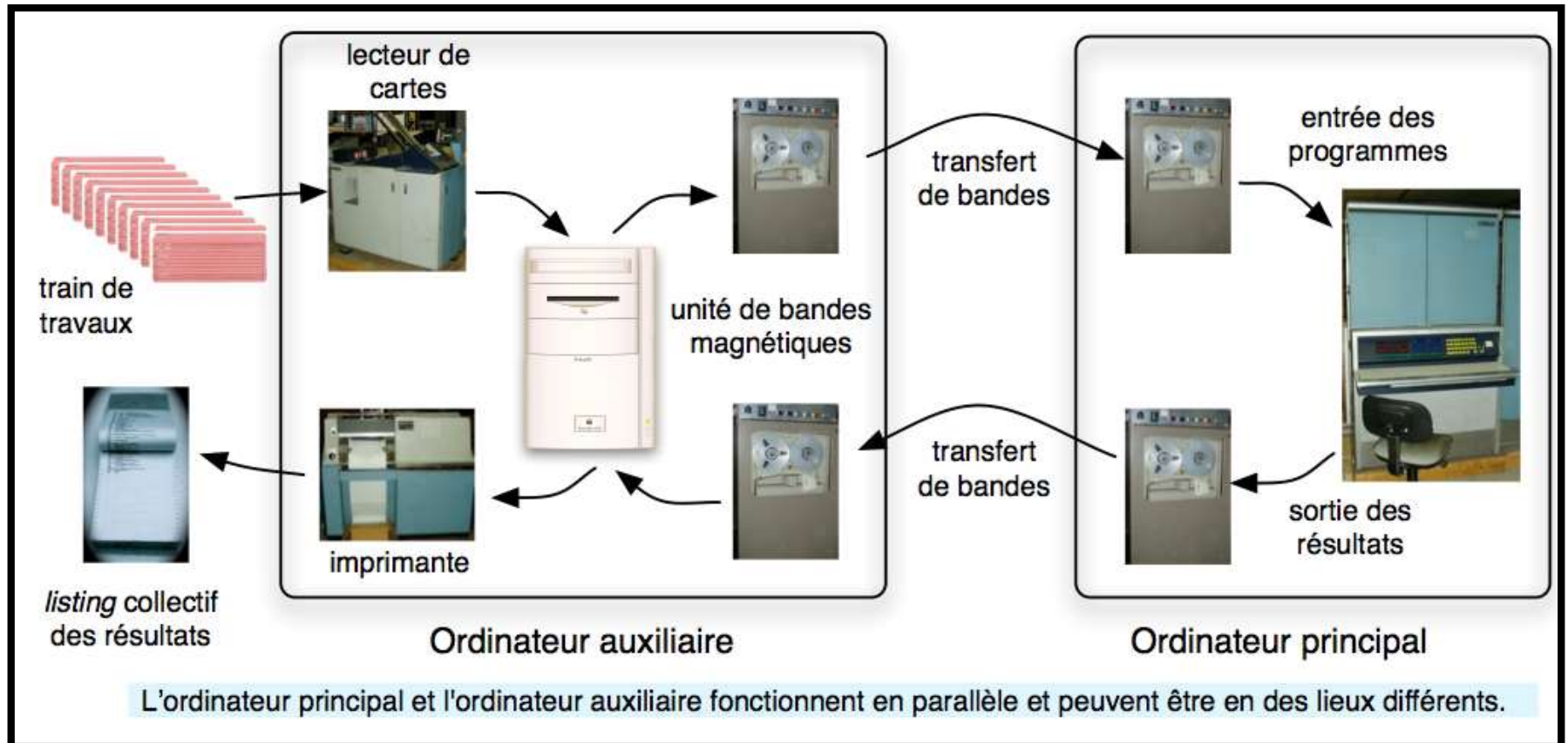
Traitement par lots (Batch Processing) (1955-1965)



La bande magnétique



Traitement par lots (Batch Processing) (1955-1965)



Traitement par lots avec ordinateur auxiliaire



Traitement par lots (Batch Processing) (1955-1965)



Multiprogrammation (Multiprogramming) (1965-1970)

L'introduction des **circuits intégrés** dans la construction des machines a permis d'offrir un meilleur rapport coût/performance.
L'introduction de la technologie des disques a permis au système d'exploitation de conserver tous les travaux sur un disque.



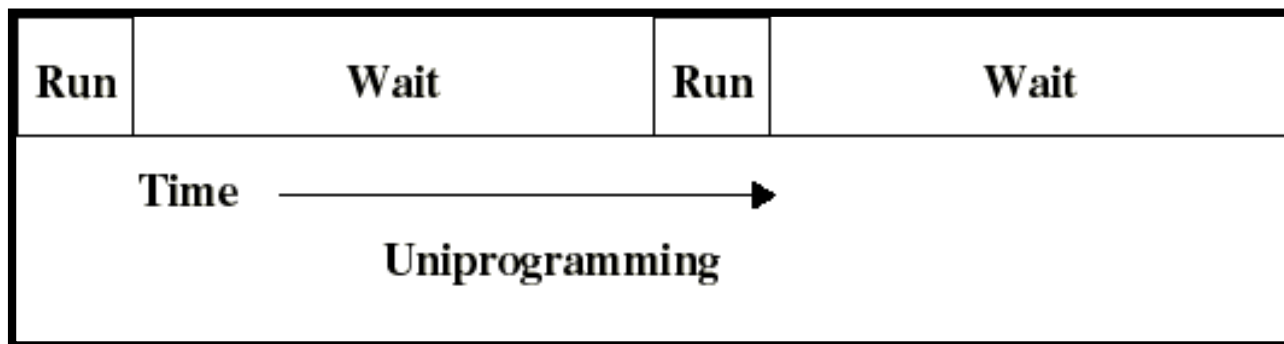
Multiprogrammation (Multiprogramming) (1965-1970)

- ☐ Amélioration des coûts et des performances (circuits intégrés).
- ☐ Une famille d'ordinateurs compatibles entre eux.
- ☐ Une seule architecture et un même jeu d'instructions.
- ☐ Des ordinateurs uniques pour les calculs scientifiques et commerciaux.
- ☐ Apparition du spouleur (spool, Simultaneous Peripheral Operation On Line) pour le transfert des travaux des cartes vers le disque.



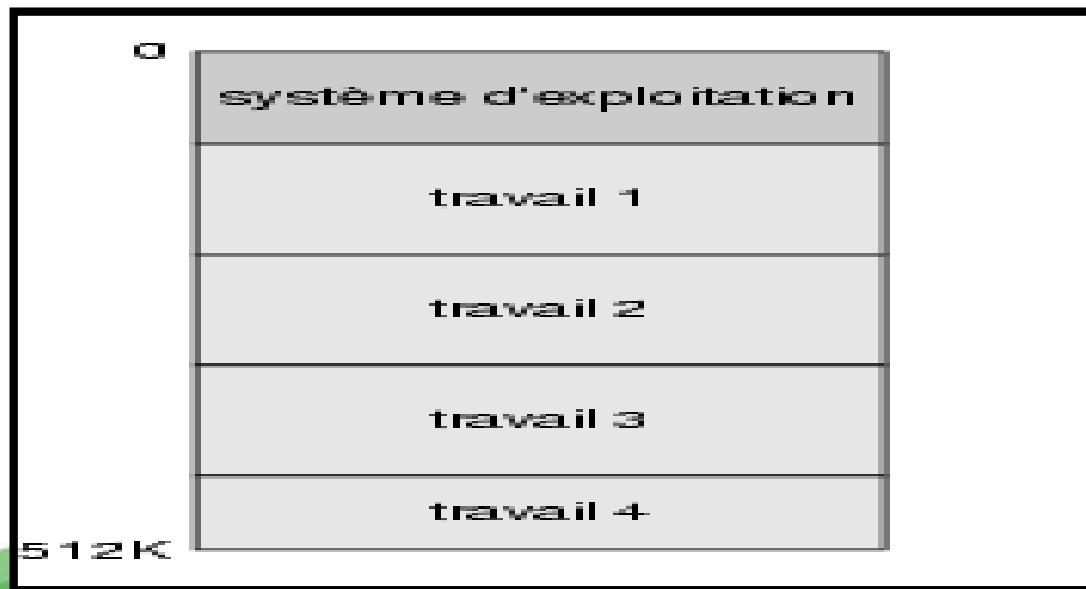
Multiprogrammation (Multiprogramming) (1965-1970)

- ❑ Les opérations E/S sont extrêmement lentes (comparé aux autres instructions)
- ❑ Même avec peu d'E/S, un programme passe la majorité de son temps à attendre
- ❑ Donc: pauvre utilisation de l'UCT lorsqu'un seul programme se trouve en mémoire

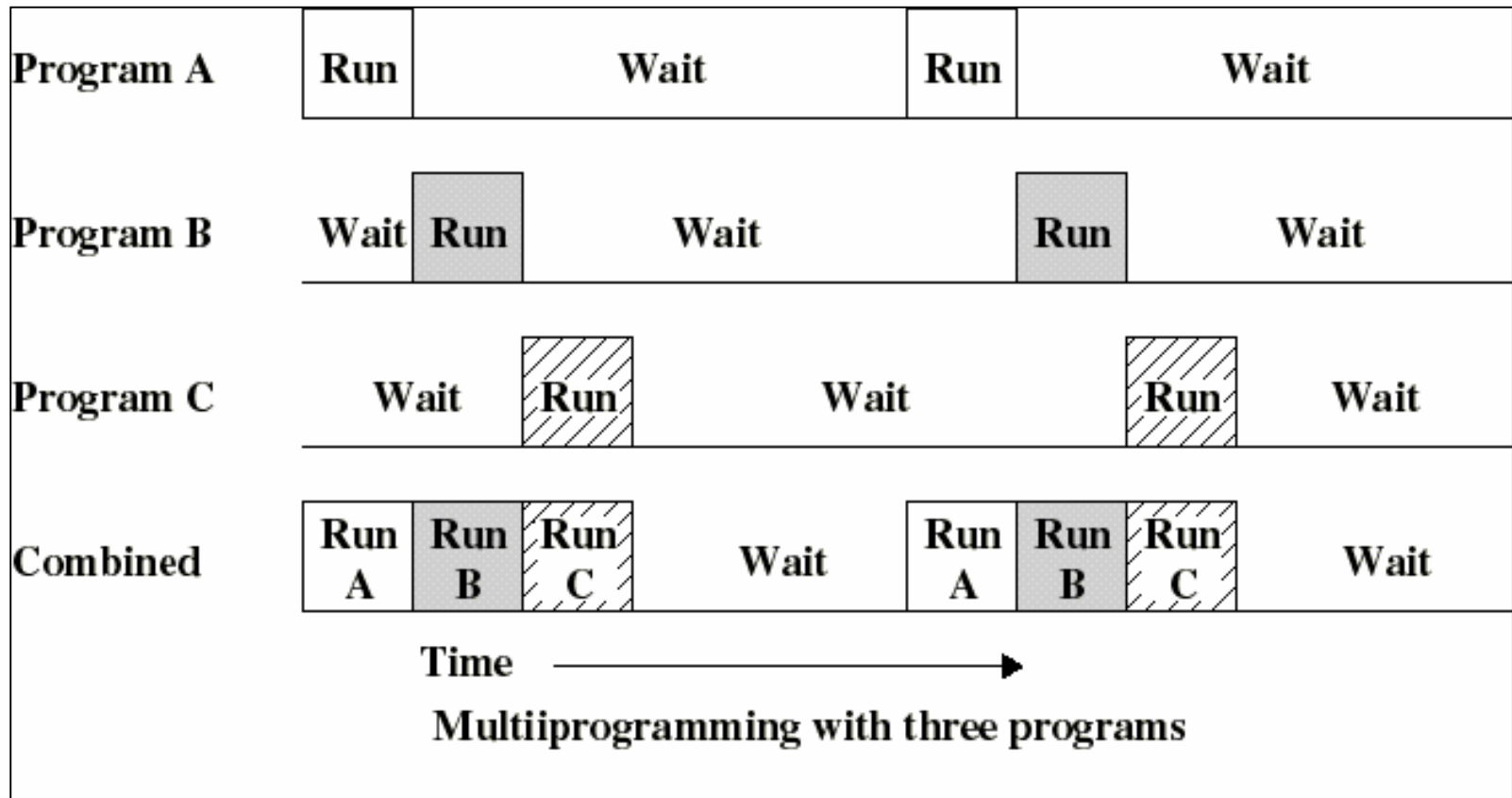


Multiprogrammation (Multiprogramming) (1965-1970)

- ❑ Si la mémoire peut contenir plusieurs programmes, l'UCT peut exécuter un autre programme lorsqu'un programme attend une E/S



Multiprogrammation (Multiprogramming) (1965-1970)

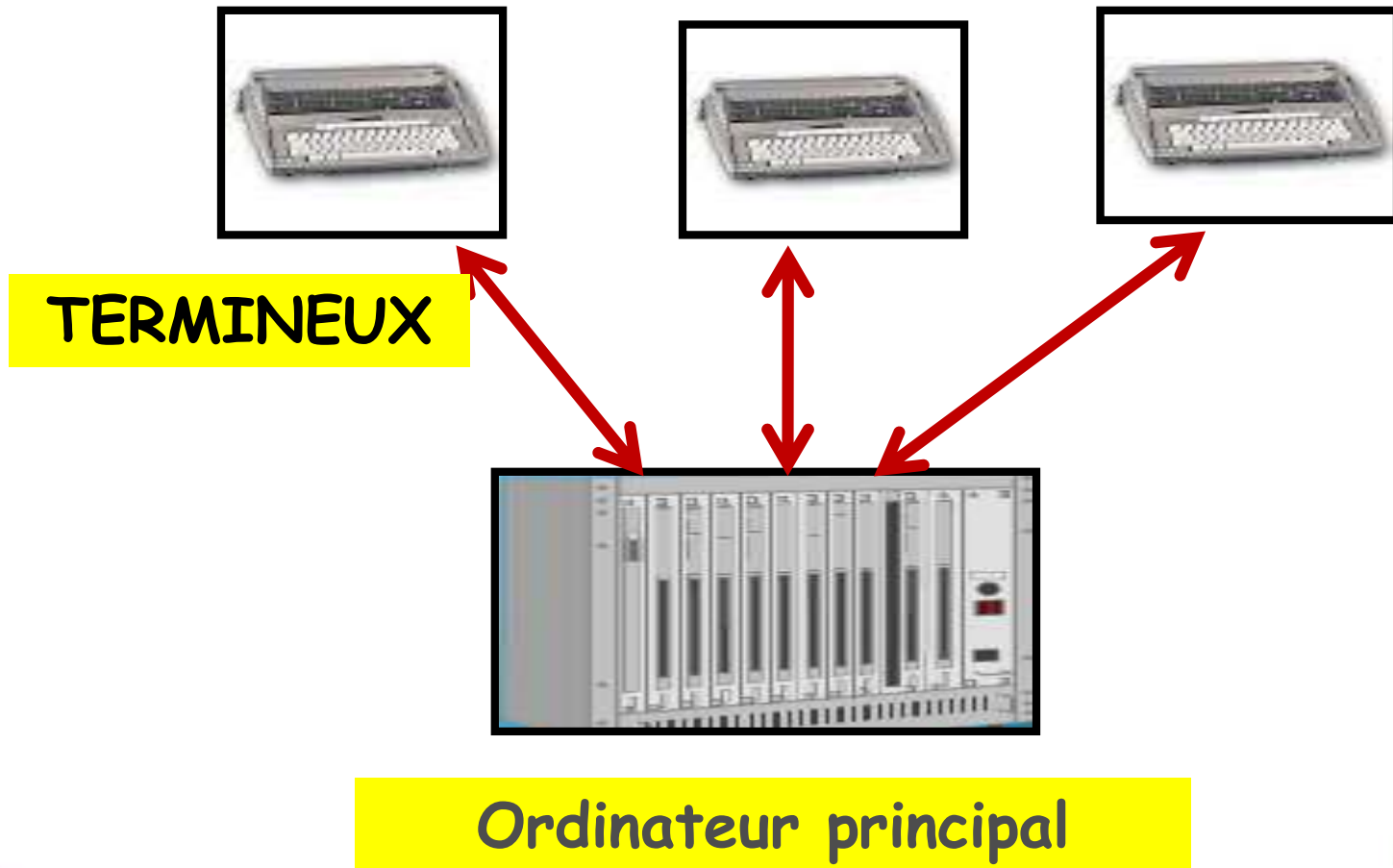


Temps partagé (Time Sharing, 1970-)

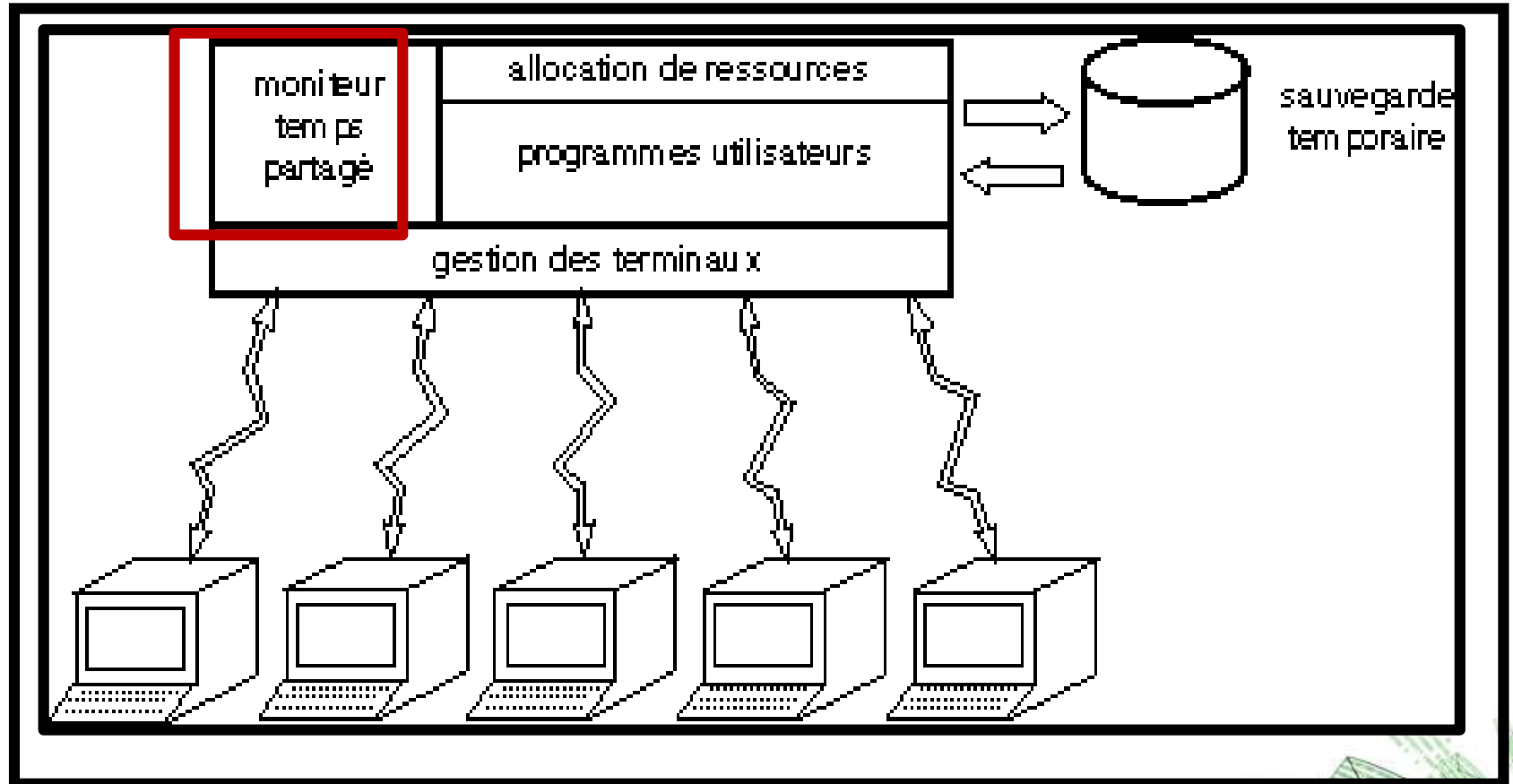
C'est une variante du mode multiprogrammé où le temps CPU est distribué en petites tranches appelées quantum de temps



Systemes à temps partagé



Systemes à temps partagé



Composants d'un système d'exploitation

plusieurs systèmes modernes possèdent les composants suivants :

- ☐ Gestion de processeurs.
- ☐ Gestion mémoire principale.
- ☐ Gestion mémoire auxiliaire.
- ☐ Gestion du système d'entrée/sortie.
- ☐ Gestion des fichiers.
- ☐ Système de protection.
- ☐ Gestion de réseaux.
- ☐ Système interpréteur de commande.



Classification des SE

- ☐ Système monotâche
- ☐ Multitâche
- ☐ Multi-utilisateurs
- ☐ Systèmes temps réel



Structuration des SE

La conception d'un SE est basée sur une structure à couches. Elle consiste à découper le SE en un certain nombre de couches (niveaux), chacune d'entre elles étant construite au-dessus des couches inférieures



Structuration des SE

Utilisateurs

Utilitaires (Shell, éditeurs, compilateurs, ...)

Bibliothèque standard (appels système)

Noyau (Gestion des processus, de la mémoire, des fichiers, des E/S, ...)

Matériel (CPU, mémoire, disques, terminaux, ...)



Chapitre 2

Mécanismes de base d'exécution des programmes



INTRODUCTION

Dans cette partie du cours , nous étudierons les mécanismes de base utilisés par les systèmes d'exploitation pour réaliser leurs fonctions fondamentales .



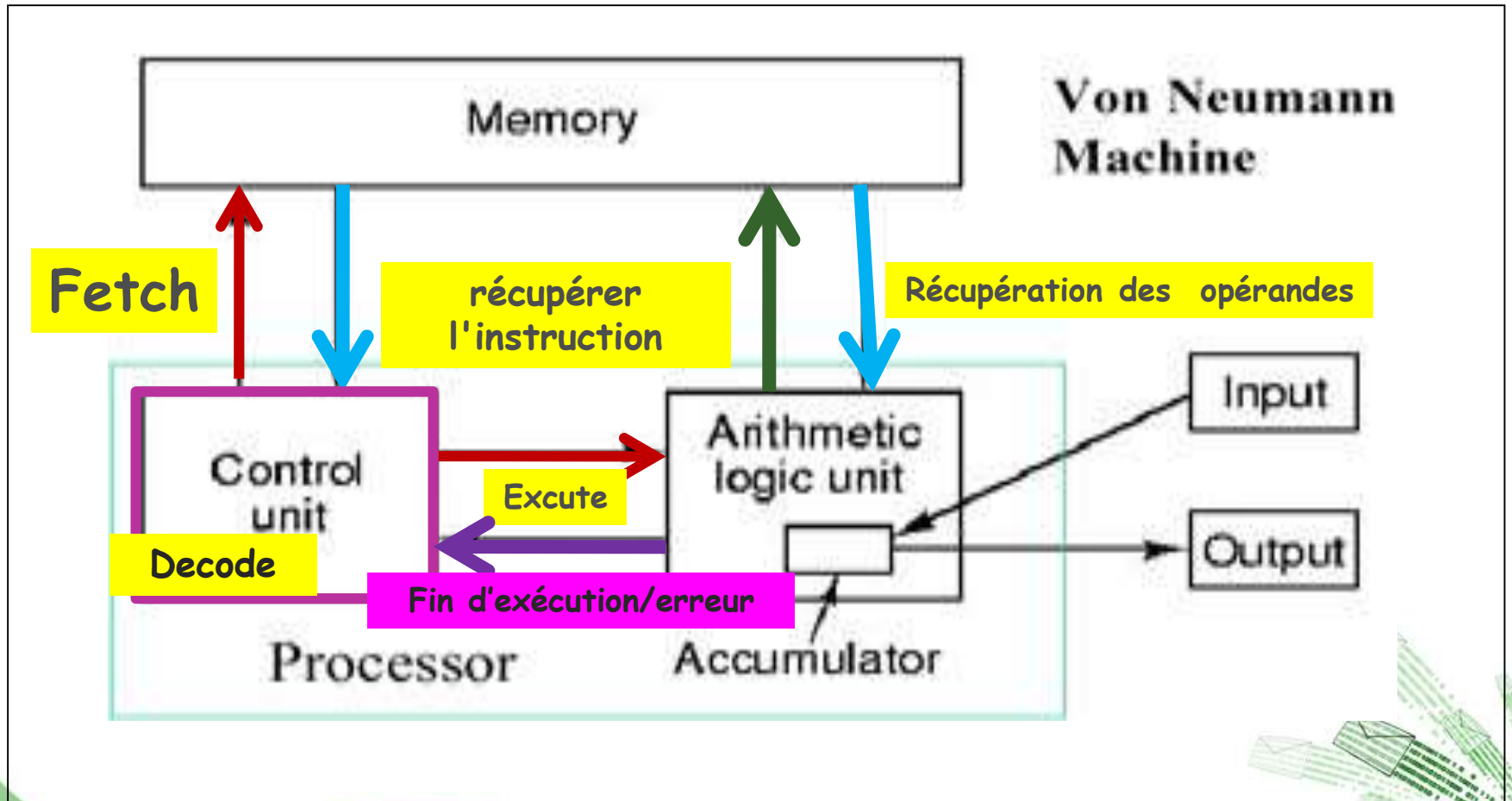
Architecture matérielle d'une machine Von Neumann

John Von Neumann est à l'origine (1946) d'un modèle de machine universelle (non spécialisée) qui caractérise les machines possédant les éléments suivants :

- ❑ une mémoire contenant programme (instructions) et données,
- ❑ une unité arithmétique et logique (UAL ou ALU),
- ❑ une unité permettant l'échange d'information avec les périphériques : l'unité d'entrée/sortie (E/S ou I/O),
- ❑ une unité de commande (UC).



Architecture matérielle d'une machine Von Neumann



Les registres du processeur central

- ❑ Le Compteur Ordinal (CO)
- ❑ Le Registre d'Instruction (RI)
- ❑ Les Registres Généraux : Registres Accumulateurs , Registres d'Index , Registre de Base , Registres Banalisés
- ❑ Le Registre Pile (Stack Pointer, SP)
- ❑ Registre Mot d'Etat (PSW, Program Status Word)



Registre Mot d'Etat (PSW)

Il contient plusieurs types d'informations ; à savoir :

- ❑ **Les valeurs courantes des codes conditions (Flags)** qui sont des bits utilisés dans les opérations arithmétiques et comparaisons des opérands.
- ❑ **Le mode d'exécution:** Pour des raisons de protection, l'exécution des instructions et l'accès aux ressources se fait suivant des modes d'exécution. deux modes d'exécution existent généralement
- ❑ **masque d'interruptions**



Registre Mot d'Etat (PSW)

- ❑ **Mode privilégié ou maître (ou superviseur).** Il permet : L'exécution de tout type d'instruction. Les instructions réservées au mode maître sont dites **privilégiées** (Ex. instructions d'E/S, protection, ...etc.). L'accès illimité aux données.
- ❑ **Mode non privilégié ou esclave (ou usager).** Il permet : Exécution d'un répertoire limité des instructions.



Etat du processeur (Processor State Information)

- ❑ Il est décrit par le contenu des registres du processeur. Le contenu des registres concerne le processus en cours d'exécution. Quand un processus est interrompu, l'information contenue dans les registres du processeur doit être sauvegardée afin qu'elle puisse être restaurée quand le processus interrompu reprend son exécution.
- ❑ **Remarque** : L'état d'un processeur n'est observable qu'entre deux cycles du processeur.



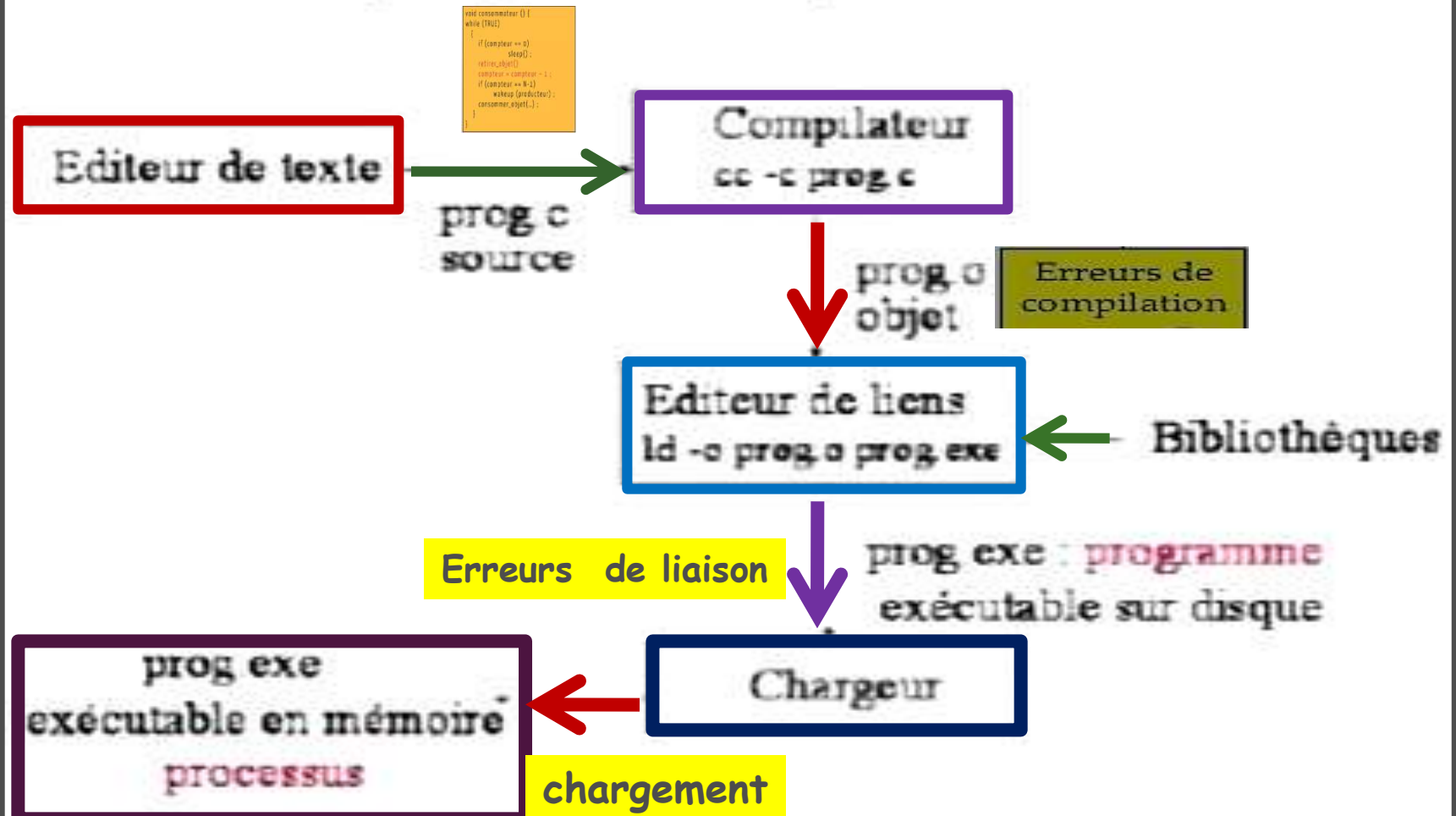
cheminement d'un programme dans un système

La chaîne de production de programme transforme un programme écrit dans un langage de haut niveau en un programme dit exécutable, écrit en langage machine. Cette transformation s'effectue à l'aide de plusieurs étapes



cheminement d'un programme dans un système

Du programme au processus



le processus

Les processus correspondent à l'exécution de tâches : les programmes des utilisateurs, les entrées-sorties, ... par le système. Un système d'exploitation doit en général traiter plusieurs tâches en même temps.



le processus

Un processus est une entité dynamique correspondant à l'exécution d'une suite d'instructions : un programme qui s'exécute, ainsi que ses données, sa **pile**, son **compteur ordinal**, son **pointeur de pile** et les autres contenus de registres nécessaires à son exécution.



le processus

Le Rôle du SE pour les processus :

- ☐ (Ré)Activer un processus
- ☐ Suspendre un processus
- ☐ Tuer un processus
- ☐ Contrôler l'exécution d'un processus De manière optimale pour le CPU

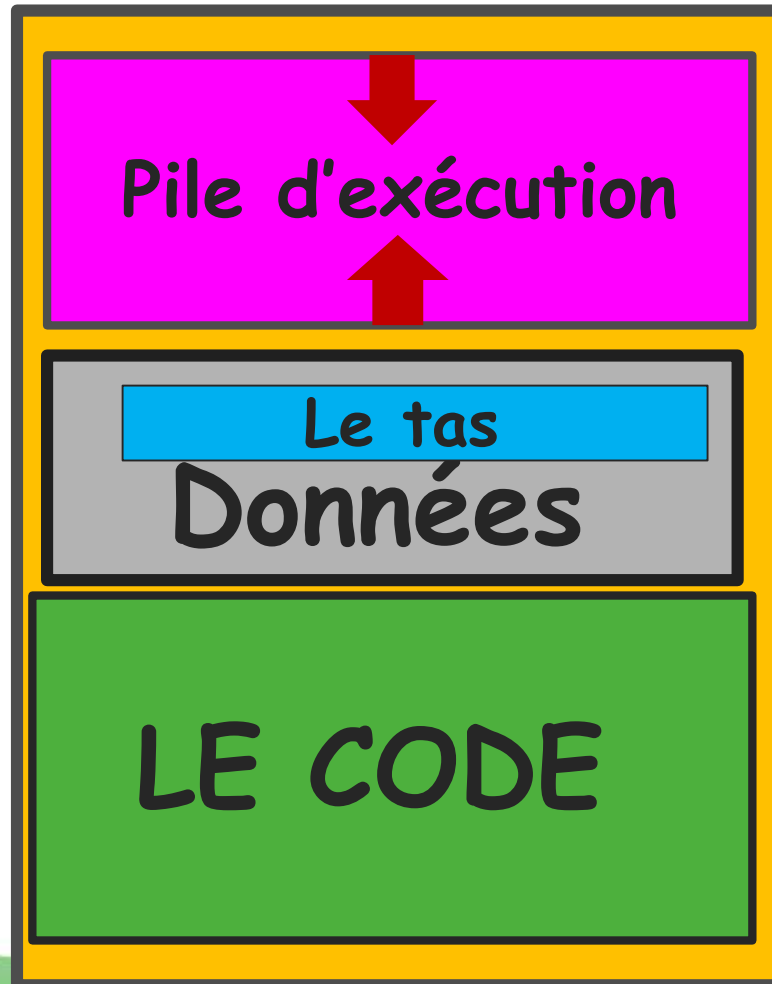


le processus

Les segments d'un processus : Les processus sont composés d'un espace de travail en mémoire formé de 3 segments : **la pile**, **les données** et **le code** et d'un contexte.



le processus



les variables globales ou
statiques du programme

+

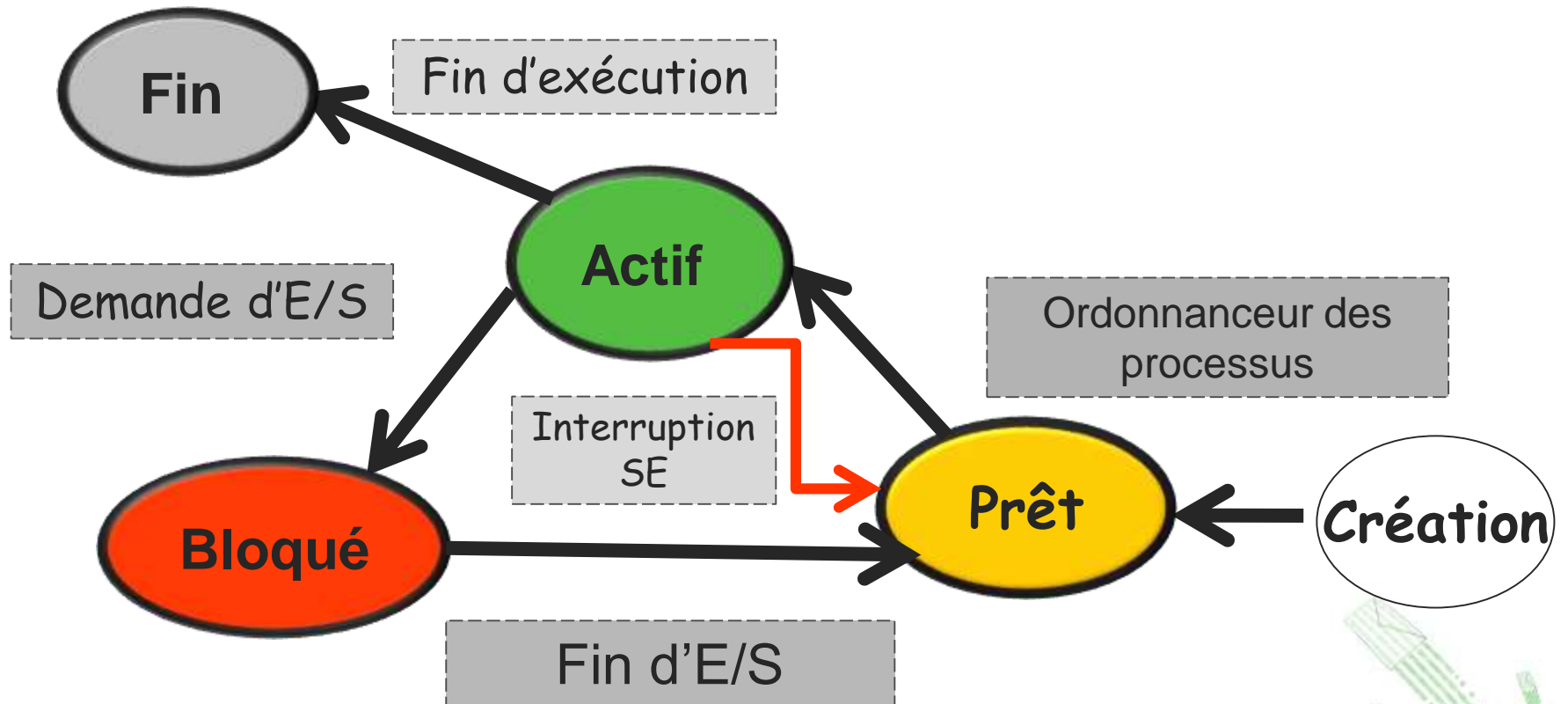
Les données dynamiques

les appels de
fonctions, avec leurs
paramètres et leurs
variables locales

Le code correspond
aux **instructions**, en
langage d'assemblage,
du programme à
exécuter.



Le Cycle de vie d'un processus



Le contexte d'un processus (PCB)

Le contexte est formé des données nécessaires à la gestion des processus. Une table contient la liste de tous les processus et chaque entrée conserve leur contexte. C'est l'ensemble des informations nécessaire et suffisante pour la reprise d'un processus au cas où il aurait été interrompu, Le contexte est utilisé :

- ☐ En lecture lors de la **restauration**.
- ☐ En écriture lors de la **sauvegarde**



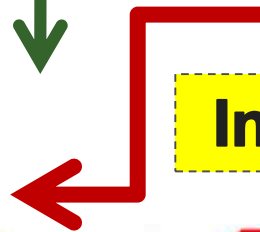
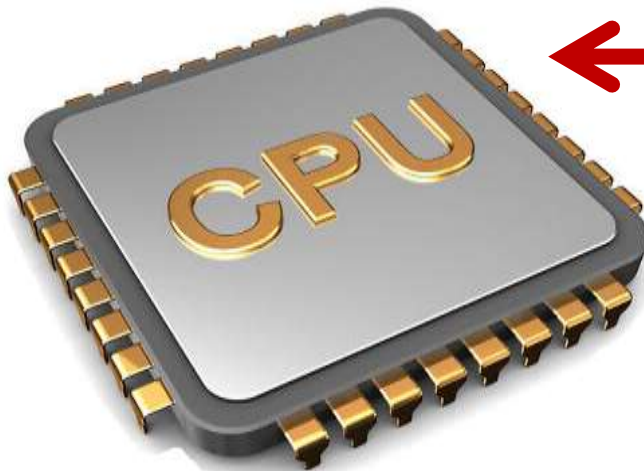
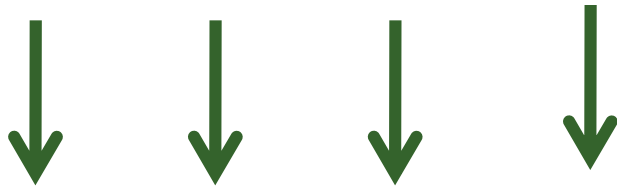
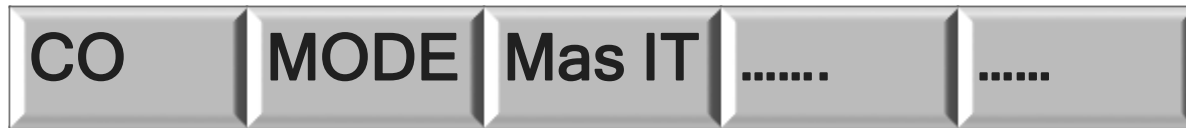
Le contexte d'un processus (PCB)

- ❑ à tout processus, on associe un bloc de contrôle de processus (BCP).
- ❑ Il comprend généralement :
 - une copie du PSW au moment de la dernière interruption du processus
 - l'état du processus : prêt à être exécuté, en attente, suspendu, ...
 - des informations sur les ressources utilisées (mémoire principale, temps d'exécution, périphériques d'E/S en attente)
 - files d'attente dans lesquelles le processus est inclus, etc...
 - toutes les informations nécessaires pour assurer la reprise du processus en cas d'interruption

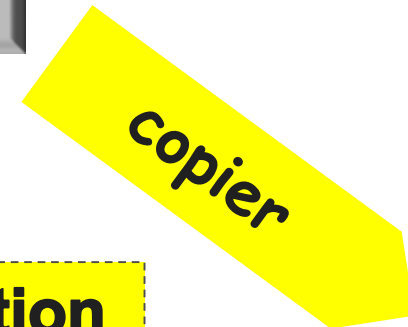


Le contexte d'un processus

PSW: le mot d'état de processeur



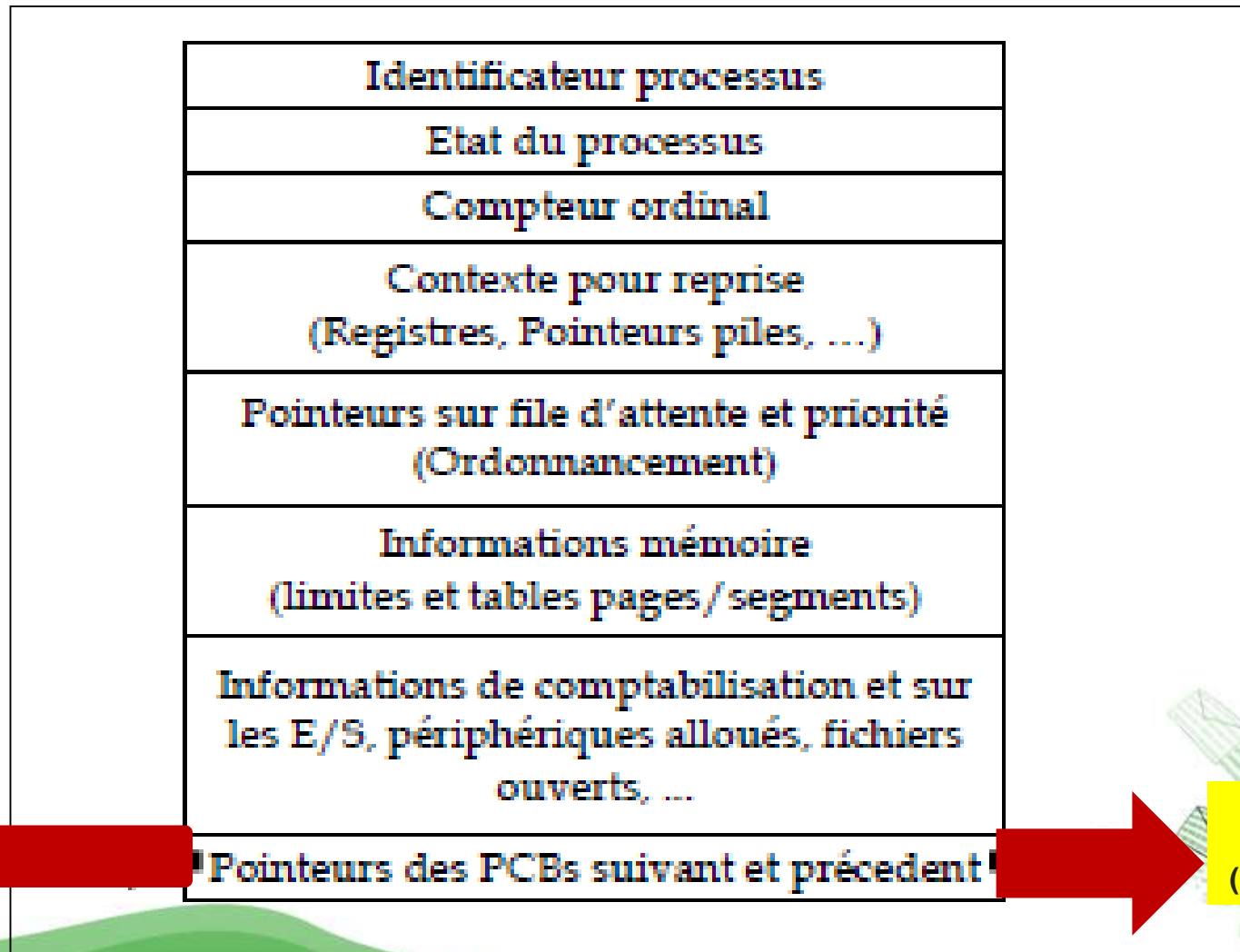
Interruption



PCB (processus)



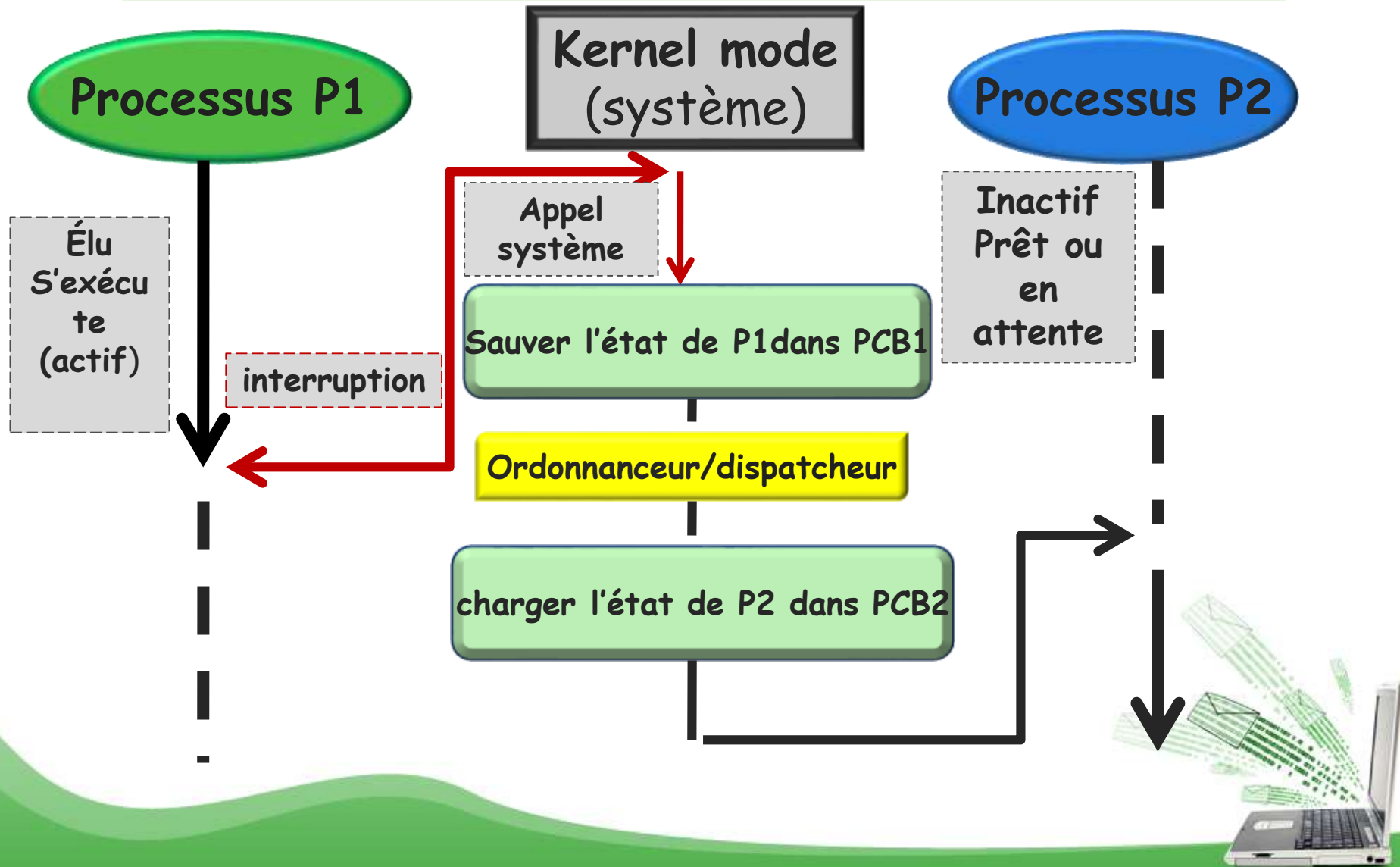
Le contexte d'un processus (PCB)



PCB
(suivant)

PCB
(Précédent)

Mécanisme de commutation de contexte



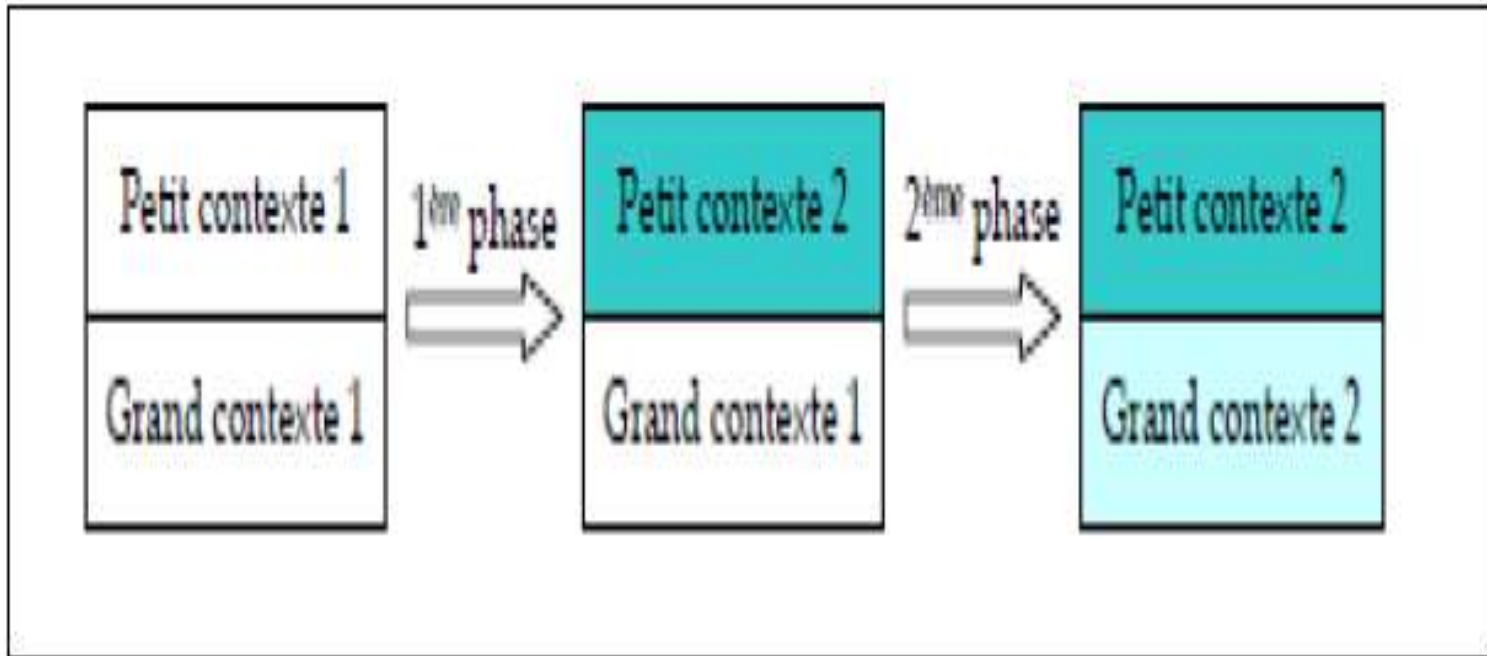
Mécanisme de commutation de contexte

La commutation du contexte se fait en deux phases :

- ❑ La première phase consiste à commuter le petit contexte (PSW) par une instruction indivisible.
- ❑ La deuxième phase consiste quant à elle à commuter le grand contexte par celui du nouveau processus.



Mécanisme de commutation de contexte



La cause d'une commutation de contexte

La commutation de contexte constitue la réponse du système à l'occurrence d'une **interruption** qui est un signal généré par le matériel comme conséquences à un événement qui s'est produit durant l'exécution du programme.



Les interruptions

Une interruption est un signal déclenché par un événement interne à la machine ou externe, provoquant l'arrêt d'un programme en cours d'exécution

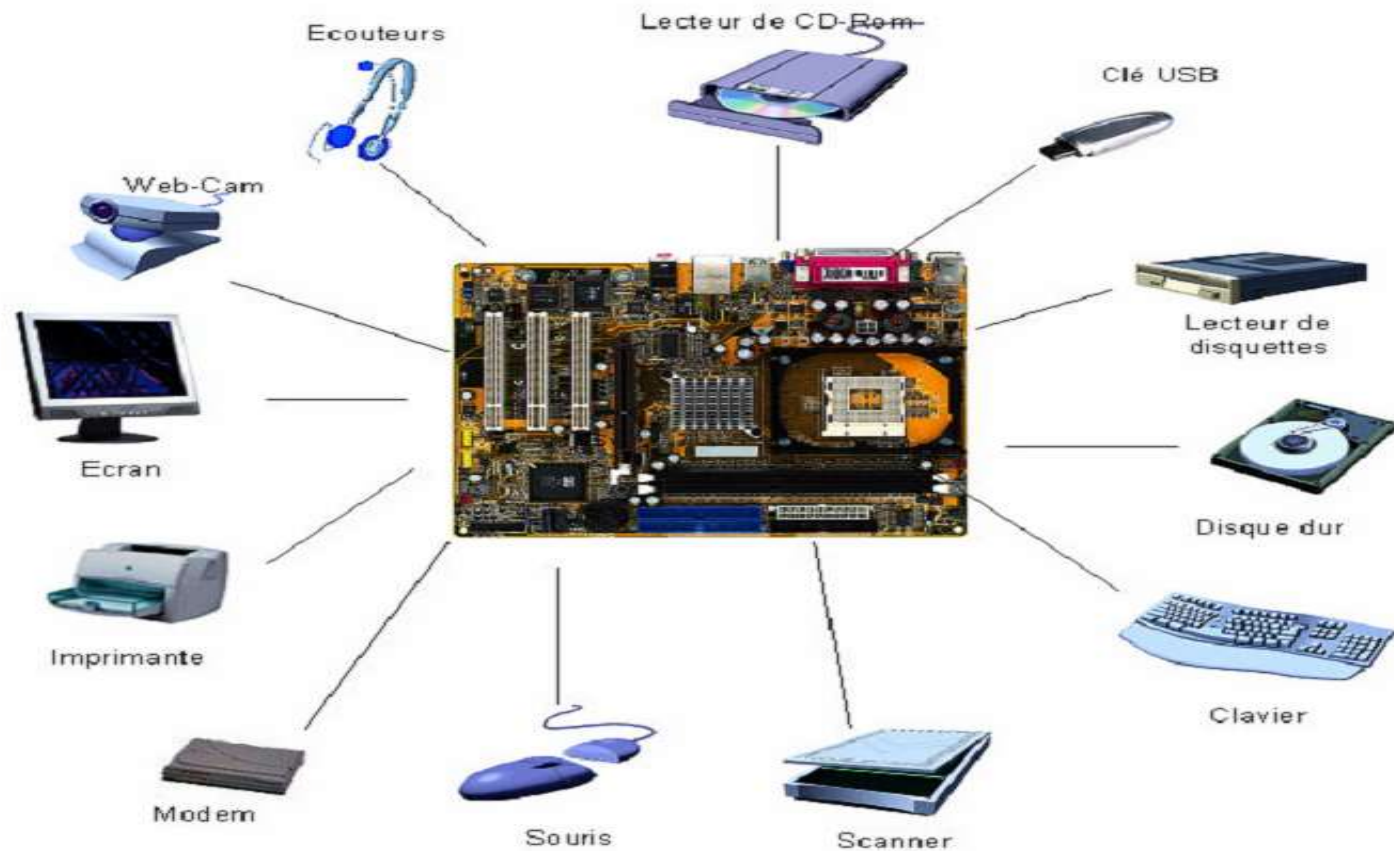


Types d'interruptions

- ☐ interruptions matérielles ou externe
- ☐ interruptions internes ou déroutement
- ☐ interruptions logiques ou appel système (SVC)

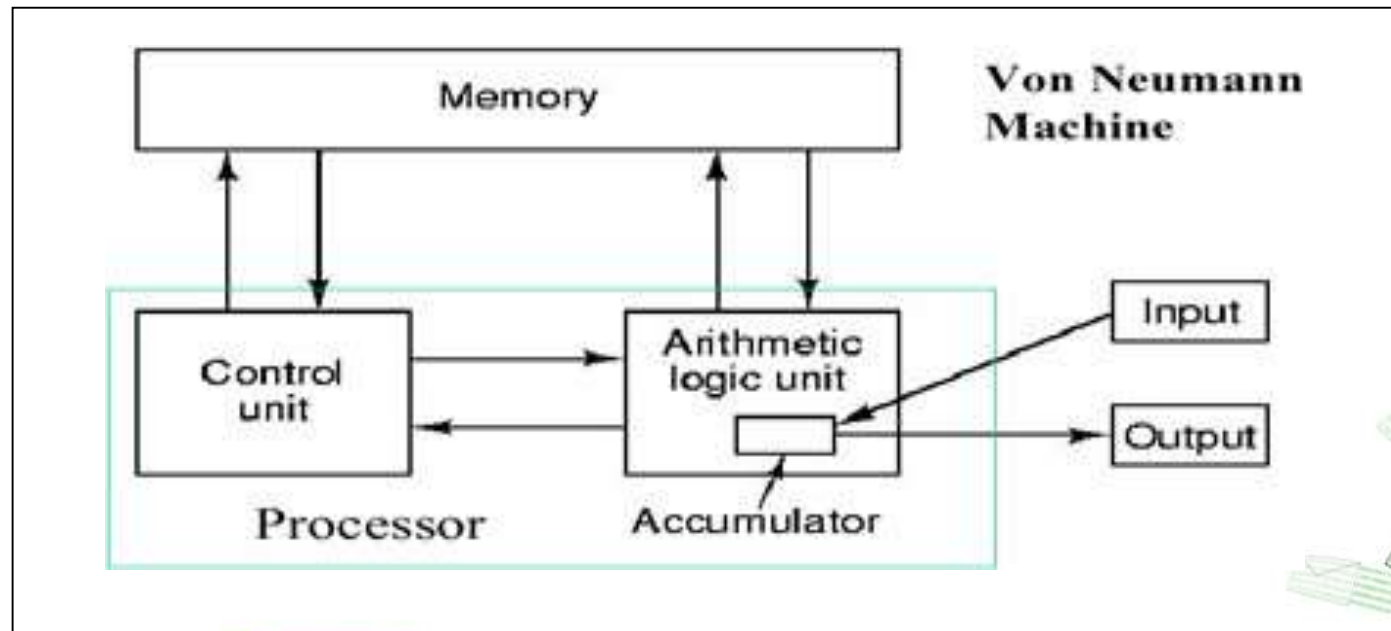


Les interruptions matérielles ou externe



Interruptions internes ou déroutement

erreur d'adressage, code opération
inexistant, problème de parité...)



Interruptions logiques ou appel système (SVC)

Processus

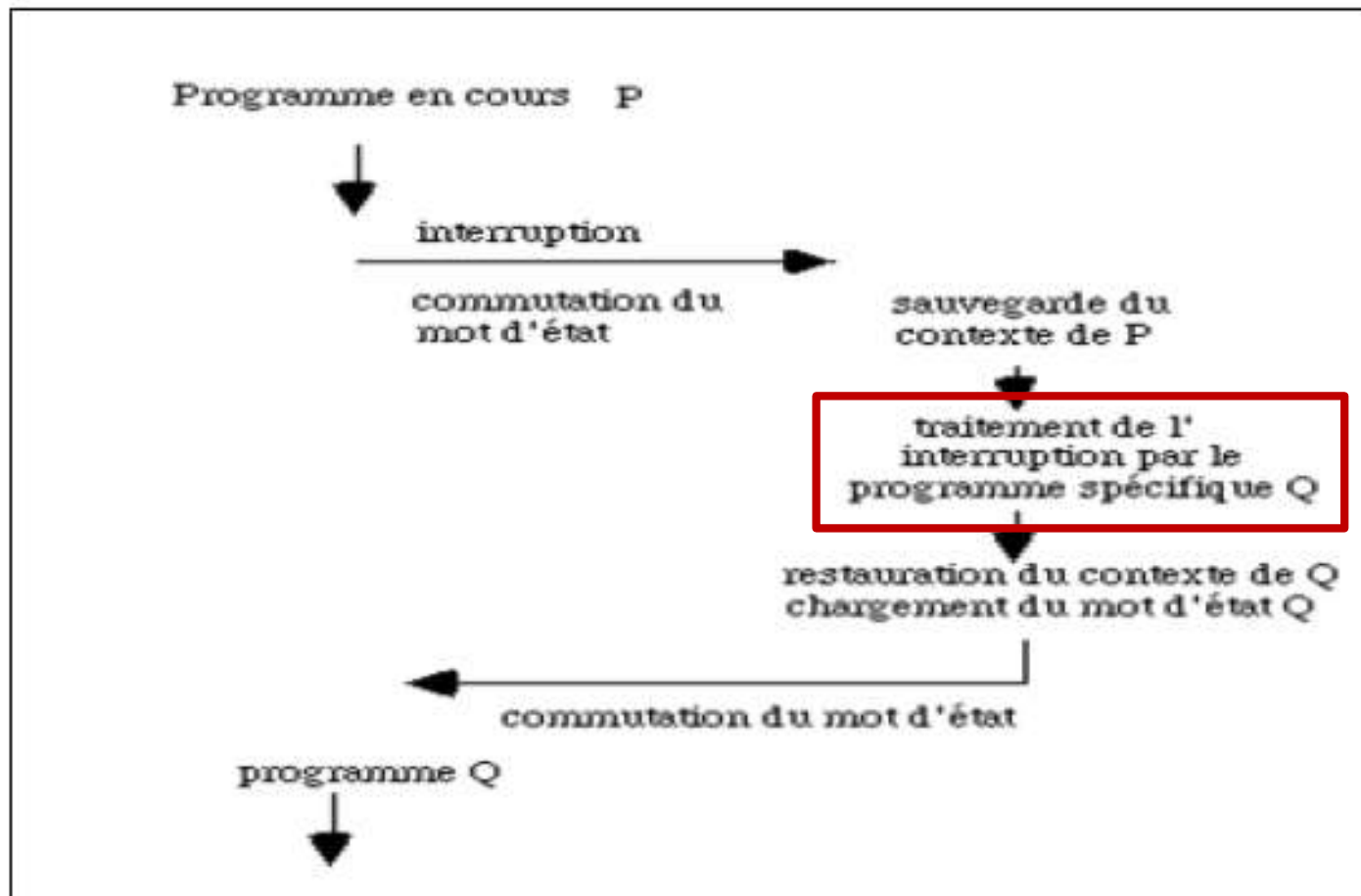
```
void consommateur () {  
    while (TRUE)  
    {  
        if (compteur == 0)  
            sleep();  
        retirer_objet();  
        compteur = compteur - 1 ;  
        if (compteur == N-1)  
            wakeup (producteur);  
        consommer_objet(.);  
    }  
}
```

SVC

Systeme
d'exploitation



Le traitement d'une interruption



Les opérations sur les interruptions

- ❑ **Masquage des interruptions:** une interruption masquée n'est pas ignorée : elle est prise en compte dès qu'elle est démasquée.
- ❑ **Désarmer une interruption :** elle sera ignorée. Par défaut, les interruptions sont évidemment armées.



Le traitement d'une interruption





Chapitre 3

La Gestion des processus



Introduction

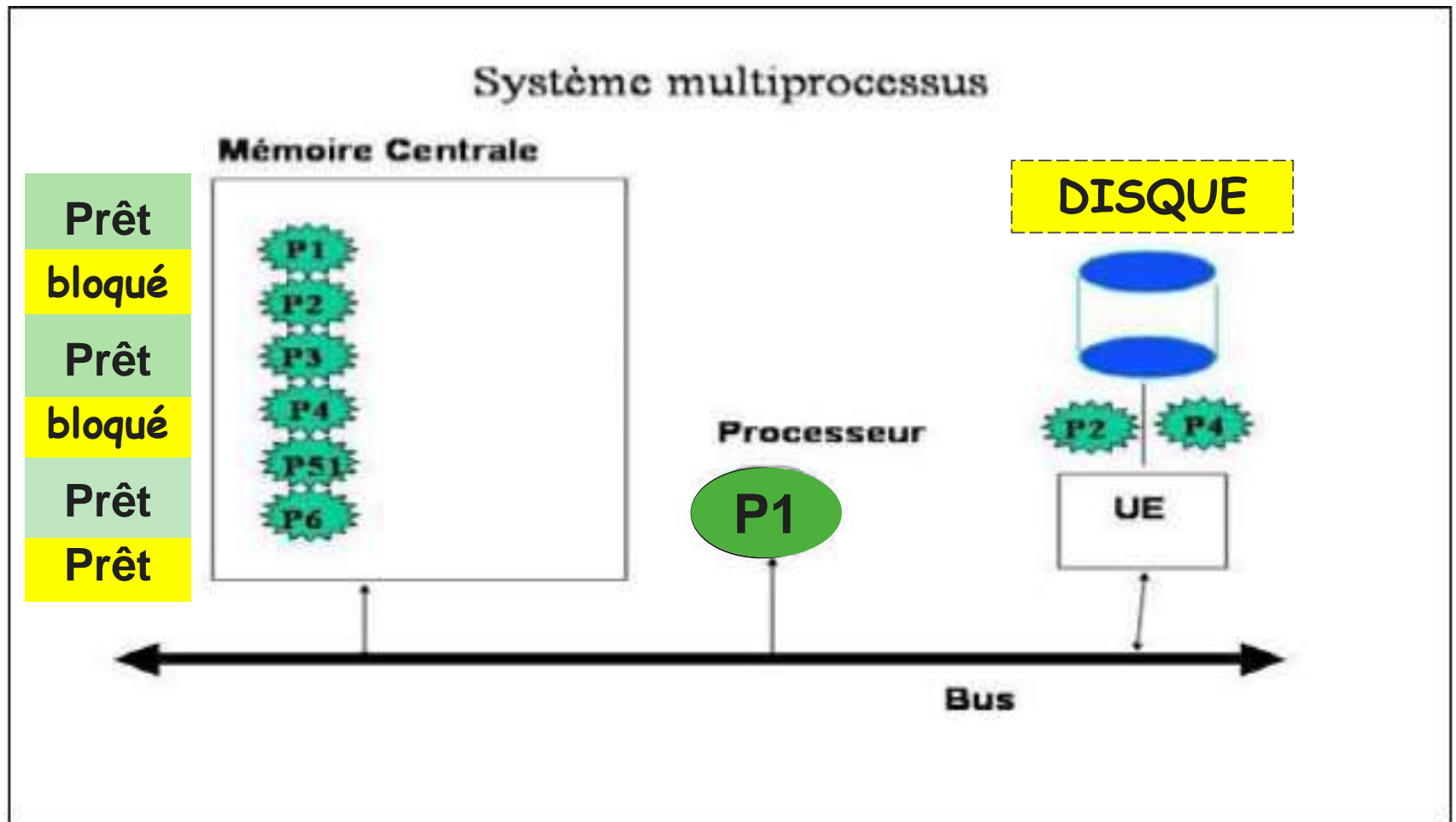
Le S.E choisit un processus qui deviendra actif parmi ceux qui sont prêts. Tout processus qui se bloque en attente d'un événement (par exemple l'attente de la frappe d'un caractère au clavier lors d'un scanf) passe dans l'état bloqué tant que l'événement attendu n'est pas arrivé. Lors de l'occurrence de cet événement, le processus passe dans l'état prêt. Il sera alors susceptible de se voir attribuer le processeur pour continuer ses activités.



Introduction

- ❑ Le changement d'état d'un processus peut être provoqué par :
 - a) un autre processus (qui lui a envoyé un signal, par exemple)
 - b) le processus lui-même (appel à une fonction d'entrée-sortie bloquante,...)
 - c) une interruption (fin de quantum, terminaison d'entrée-sortie, ...)
- ❑ le passage de l'état actif à l'état prêt est provoqué par le système en fonction de sa politique **d'ordonnancement** (fin de quantum, préemption du processus actif si un processus plus prioritaire devient prêt)

Qu'est-ce que l'ordonnancement de processus ?



Qu'est-ce que l'ordonnancement de processus ?

Définition : La fonction d'ordonnancement gère le partage du processeur entre les différents processus en attente pour s'exécuter, c'est-à-dire entre les différents processus qui sont dans l'état prêt. L'opération d'élection consiste à allouer le processeur à un processus.



Qu'est-ce que l'ordonnancement de processus ?

D'une manière plus concrète, cette tâche est prise en charge par deux routines système en l'occurrence le **Dispatcheur** et **L'ordonnanceur (Scheduleur)**.



le Dispatcheur

Il s'occupe de l'allocation du processeur à un processus sélectionné par l'Ordonnanceur du processeur. Une fois allouer, le processeur doit réaliser les tâches suivantes :

- A. Commutation de contexte** : sauvegarder le contexte du processus qui doit relâcher le processeur
- B. Commutation du mode d'exécution** : basculer du mode Maître (mode d'exécution du dispatcheur) en mode utilisateur
- C. Branchement** : se brancher au bon emplacement dans le processus utilisateur pour le faire démarrer.



L'Ordonnanceur

deux types d'Ordonnanceurs :

- a) **Ordonnanceur du processeur** : c'est un Ordonnanceur court terme opère sur une ensemble du processus présents en mémoire.
- b) **Ordonnanceur de travail** : ou Ordonnanceur long terme, utilisé en cas d'insuffisance de **mémoire**, son rôle est de sélectionné le sous ensemble de processus stockés sur un disque et qui vont être chargés en mémoire.



Ordonnancement préemptif ou non préemptif

La préemption correspond à une opération de réquisition du processeur, c'est-à-dire que le processeur est **retiré** au processus élu alors que celui-ci dispose de toutes les ressources nécessaires à la poursuite de son exécution. Cette réquisition porte le nom de préemption.



Ordonnancement préemptif

actif

si l'ordonnancement est **préemptif**, la transition de l'état élu vers l'état prêt est **autorisée**

Prêt



Ordonnancement non préemptif

actif



si l'ordonnancement est **non préemptif**, la transition de l'état élu vers l'état prêt est **interdite**

Prêt



Objectifs des politiques d'ordonnancement

- ❑ Systèmes de traitements par lots. Le but est de maximiser le débit du processeur ; c'est-à-dire le nombre moyen de processus traités par unité de temps.
- ❑ le taux d'occupation du processeur



Critères d'évaluation de performances

- ❑ **Le rendement d'un système** : est le nombre de processus exécutés par unité de temps.
- ❑ **Le temps de réponse** : est le temps qui s'écoule entre le moment où un processus devient prêt à s'exécuter et le moment où il finit de s'exécuter (temps d'accès mémoire + temps d'attente dans la file des processus éligibles + temps d'exécution dans l'unité centrale + temps d'attente + temps d'exécution dans les périphériques d'entrée/sortie).



Critères d'évaluation de performances

Le temps de réponse moyen décrit la moyenne des dates de fin d'exécution

$$TRM = \sum_{i=1}^n TR_i / n, \text{ avec } TR_i = \text{date fin} - \text{date arrivée.}$$

□ **Le temps d'attente** : est le temps passé dans la file des processus éligibles. On utilise en général un temps moyen d'attente calculé pour tous les processus mis en jeu pendant une période d'observation donnée.

$$TAM = \sum_{i=1}^n TA_i / n, \text{ avec } TA_i = TR_i - \text{temps d'exécution}$$



Ordonnancement non préemptif

- ❑ Ordonnancement **FCFS** (first come first Served)
- ❑ L'algorithme du Plus Court d'abord(SJF)
- ❑ Ordonnancement basé sur les **priorités**



Ordonnancement FCFS

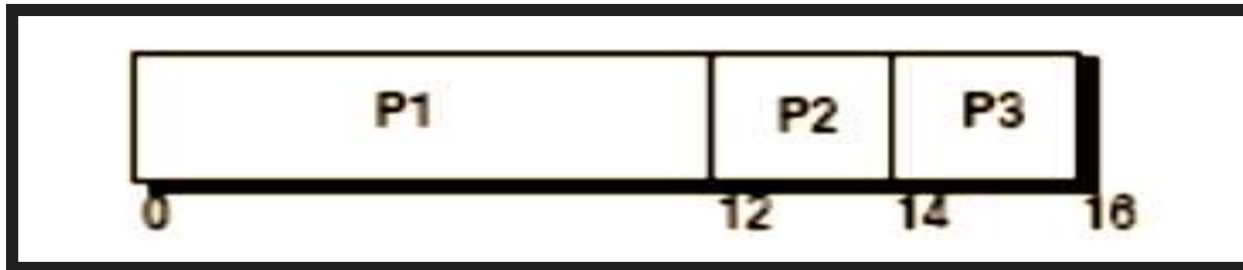
Dans cet algorithme ; connu sous le nom **FIFO** (First In, First Out) ; les processus sont rangés dans la file d'attente des processus prêts selon leur ordre d'arriver. Les règles régissant cet ordonnancement sont :

- Quand un processus est prêt à s'exécuter, il est mis en queue de la file d'attente des processus prêts.
- Quand le processeur devient libre, il est alloué au processus se trouvant en tête de file d'attente des processus prêts.



Ordonnancement FCFS

Exemple: On a trois processus 1, 2, 3 à exécuter de durées d'exécution respectives 12, 2 et 2 unités de temps.



L'algorithme du Plus Court d'abord(SJF)

Cet algorithme (en anglais Shortest Job First : SJF) affecte le processeur au processus possédant le temps d'exécution le plus court. Si plusieurs processus ont la même durée, une politique FIFO sera alors utilisée pour les départager.

- **Problème** : comment prédire la durée d'exécution des processus a priori ?
- **Solutions** : faire de la prédiction sur la durée d'exécution des processus. faire accompagner la demande d'exécution de chaque programme d'une durée maximum d'exécution autorisée qui est utilisée comme durée d'exécution.



L'algorithme du Plus Court d'abord(SJF)

Exemple : On soumet au système quatre processus P1, P2, P3 et P4 dont les durées d'exécution sont données par le tableau suivant

Processus	Date d'arrivée	Temps CPU
P1	0	6
P2	2	3
P3	3	7
P4	3	4



Ordonnancement basé sur les priorités

Dans cet algorithme, les processus sont rangés dans la file d'attente des processus prêt par ordre décroissant de priorité. L'ordonnancement dans ce cas est régit par les règles suivantes :

- A. Quand un processus est admis par le système, il est inséré dans la file d'attente des processus prêts à sa position appropriée (dépend de la valeur de priorité).
- B. Quand le processeur devient libre, il est alloué au processus se trouvant en tête de file d'attente des processus prêts (le plus prioritaire).
- C. Un processus élu relâche le processeur s'il se termine ou se bloque (E/S ou autre).



Ordonnancement basé sur les priorités

Exemple: On dispose de 5 processus ayant des priorités différentes, comme le montre ce tableau :

processus	Durée d'exécution	Priorité
P1	10	2
P2	1	4
P3	2	2
P4	1	1
P5	5	3



Ordonnancement préemptif

- A. L'algorithme de **Round Robin**(**Tourniquet**)
- B. Ordonnancement **SRTF** (plus court temps d'exécution restant **PCTER**) .
- C. avec **priorité** (avec réquisition)



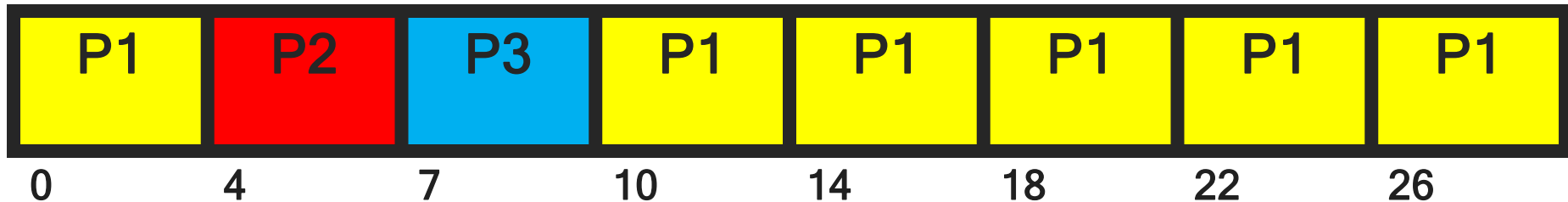
L'algorithme de Round Robin (**RR**) (Tourniquet)

L'algorithme tourniquet, appelé aussi Round Robin, a été conçu pour des systèmes à temps partagé. Il alloue le processeur aux processus à tour de rôle, pendant une tranche de temps appelée quantum. Dans la pratique le quantum s'étale entre 10 et 100 ms.



L'algorithme de Round Robin (RR) (Tourniquet)

Exemple : On dispose de 3 processus P1, P2 et P3 ayant comme durée d'exécutions, respectivement **24**, **3** et **3** ms. En utilisant un algorithme Round Robin, avec un quantum de 4 ms, on obtient le diagramme de Gantt suivant :



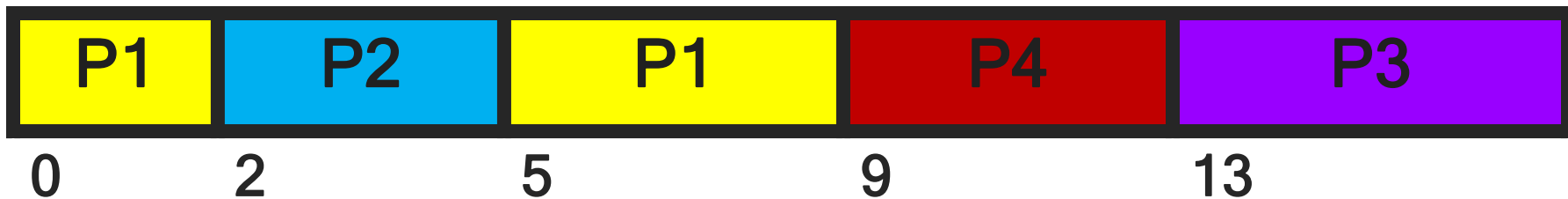
Ordonnancement SRTF

choisit le processus dont le temps d'exécution restant est le plus court. C'est une variante de l'algorithme **SJF**. Cet algorithme est non implantable parce qu'il suppose, entre autres, connu le temps d'exécution réel de chaque processus pour pouvoir calculer le temps restant.



Ordonnancement SRTF

Processus	Date d'arrivée	Temps CPU
P1	0	6
P2	2	3
P3	3	7
P4	3	4



L'ordonnancement avec priorité (avec réquisition)

On associe une **priorité** à chaque processus et on choisit dans la file des processus prêts le processus qui a **la priorité la plus haute**. Si 2 processus ont la priorité la plus haute égale, on choisit le 1er dans la file d'attente. Lorsqu'un processus arrive dans l'état prêt, le système d'exploitation regarde si la priorité de ce processus est **strictement supérieure** à celui du processus en exécution.



L'ordonnancement avec priorité (avec réquisition)

Processus	Date d'arrivée	Temps CPU	Priorité
P1	0	6	2
P2	2	3	4
P3	3	7	3
P4	3	4	2



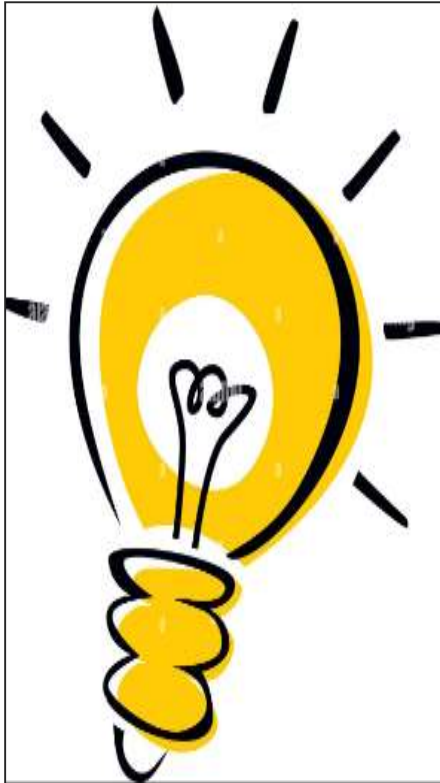
Priorités dynamiques



Si le système est très chargé, les processus de faible priorité n'obtiennent jamais le CPU (**famine** ou **starvation**), pour pallier à ce défaut, la priorité doit être un paramètre dynamique qui permettra de changer le processus de file (files réactives), l'ajustement dynamique de la priorité d'un processus est calculé par l'ordonnanceur.



Priorités dynamiques



- ☐ Recyclage avec priorité variable
- ☐ Les files multi niveaux
- ☐ Les files multi niveaux avec feedback



Recyclage avec priorité variable

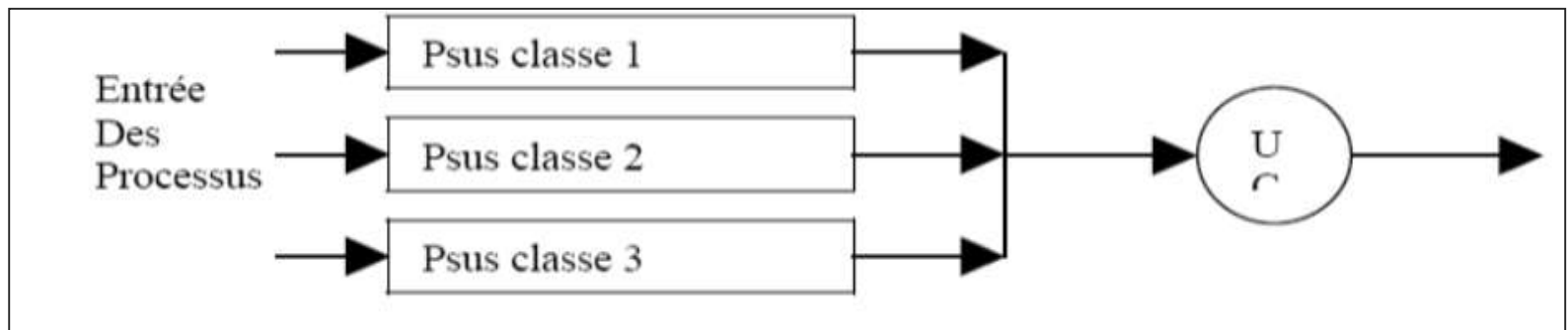
Les priorités sont calculées de deux façons différentes:

- ❑ Si un processus est arrêté par un **événement E** (entrée-sortie, fonctions wait, pause ...), il est réactivé avec une priorité qui dépend du type de cet événement.
- ❑ La priorité des **processus prêts** est directement liée à la consommation de temps CPU et au temps qui s'est écoulé depuis leur dernier accès au processeur.



Les files multi niveaux

Les processus sont regroupés par classe, et à chacune est associée une priorité et politique d'allocation.



Les files multi niveaux avec feedback

Plusieurs queues (politiques) les processus peuvent passer d'une queue à une autre processus prêt sont g rer dans plusieurs files de priorit s, et il est possible de faire passer un processus d'une file   une autre.

