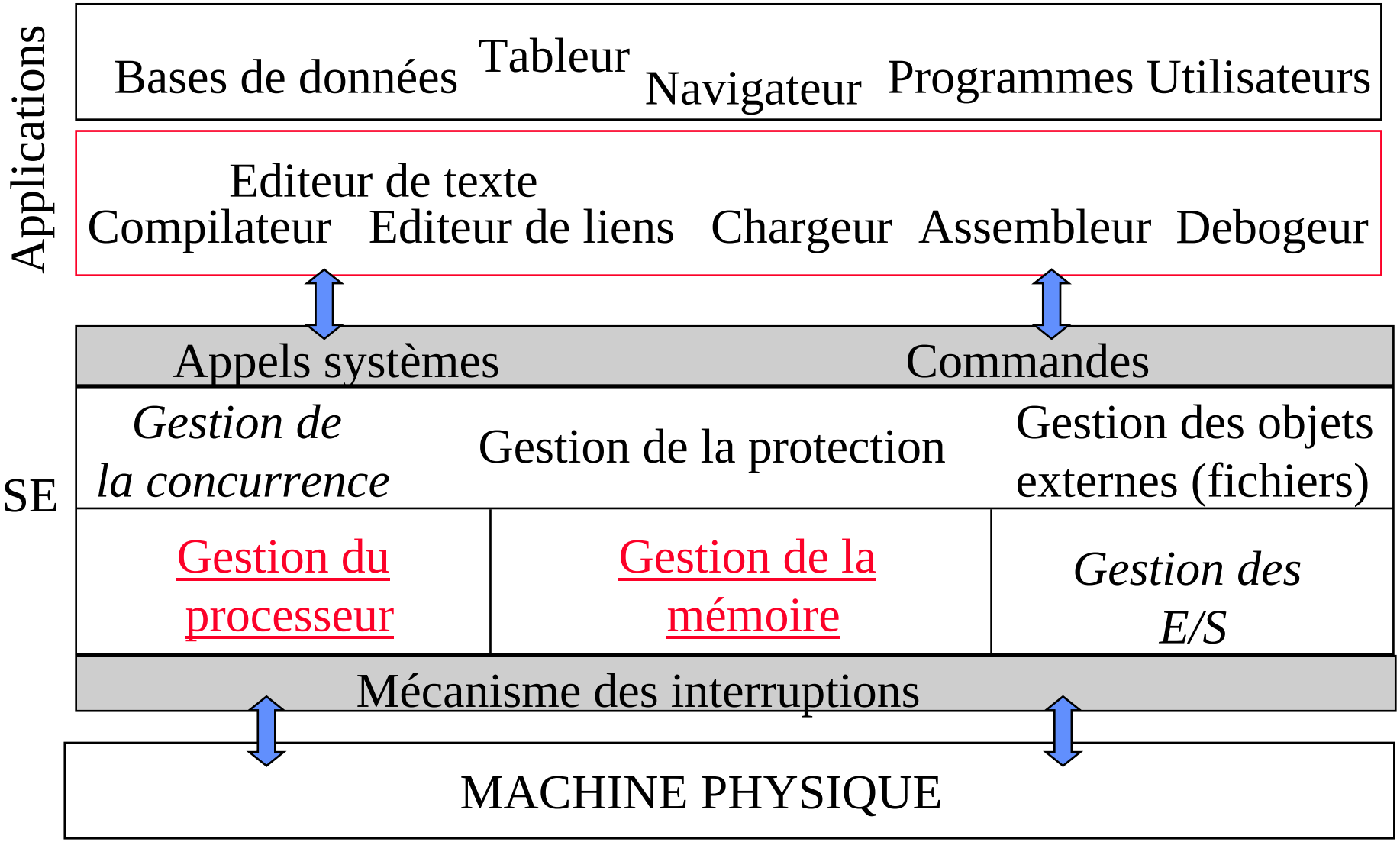
Fonctions d’un système d’exploitation

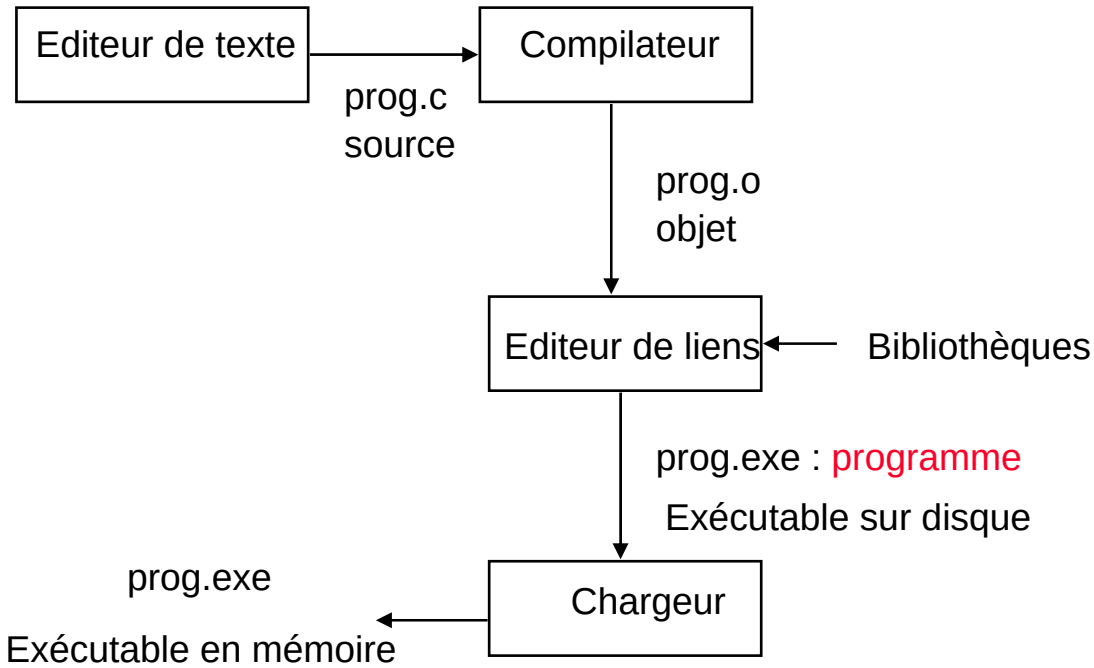
FONCTIONS D'UN SYSTEME D'EXPLOITATION



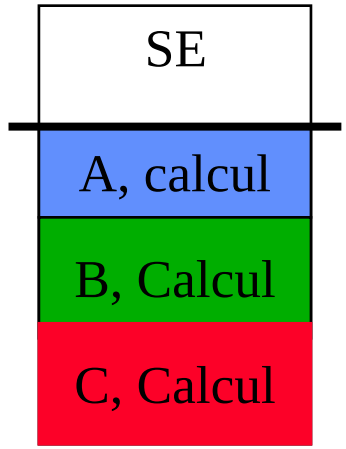
La chaîne de production de programmes

Cette chaîne désigne l'ensemble des étapes nécessaires à la construction d'un programme exécutable à partir d'un programme dit source :

- Compilation
- Edition des liens
- Chargement



FONCTIONS D'UN SYSTEME D'EXPLOITATION



- Gestion du processeur

  – le système doit gérer l'allocation du processeur aux différents programmes pouvant s'exécuter (processus). Cette allocation se fait par le biais d'un **algorithme d'ordonnancement** qui planifie l'exécution des programmes

➢ Un processus est un programme en cours d'exécution

➢ Un algorithme d'ordonnancement est un programme du système qui planifie l'exécution des processus

Ordonnancement : un exemple

1. A, calcul 10 ms 2. B, calcul 20 ms 3. C, calcul 15 ms

A, B et C sont trois programmes à exécuter (processus).

Planification/ordonnancement selon l'ordre d'arrivée

Planification/ordonnancement selon le temps d'exécution

FONCTIONS D'UN SYSTEME D'EXPLOITATION

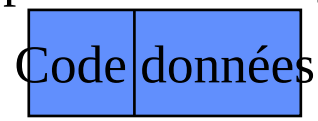
- Gestion de la mémoire

  – le système doit gérer **l'allocation de la mémoire** centrale entre les différents programmes pouvant s'exécuter Comme la mémoire physique est souvent trop petite pour contenir la totalité des programmes, la gestion de la mémoire se fait selon le principe de la **mémoire virtuelle**.

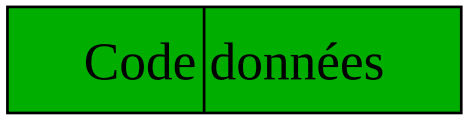
    ↳à un instant donné, seules sont chargées en MC, les parties de code et données utiles à l'exécution

Gestion de ma mémoire centrale : un exemple

Espace d'adressage



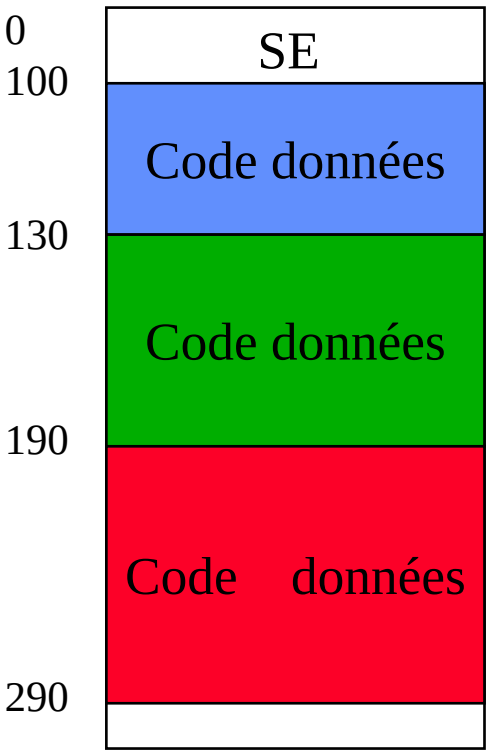
A, 30 Ko

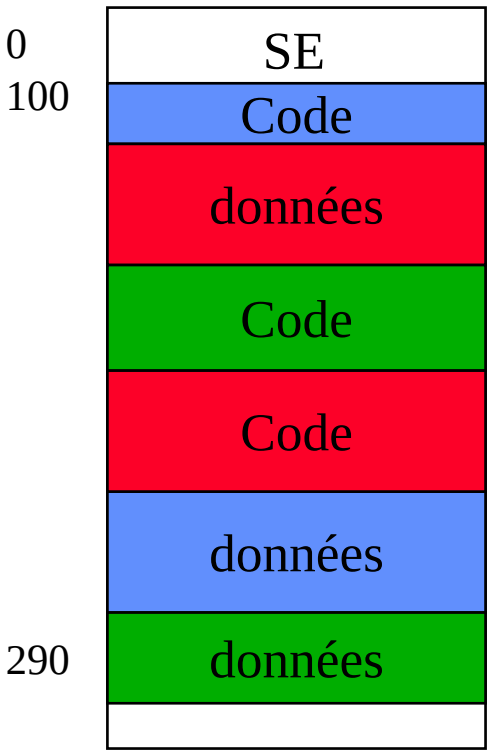


B, 60 Ko

C, 100 Ko

A, B et C sont trois programmes à exécuter (processus).





La **méthode d'allocation** de la mémoire centrale détermine comment l'espace d'adressage de chaque programme (processus) est placé en mémoire centrale

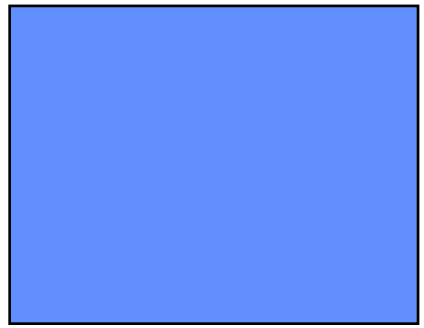
L'espace d'adressage d'un programme correspond à l'ensemble des adresses accessibles par ce programme lors de son exécution

Gestion de ma mémoire centrale : un exemple

```
Pour i = 1 à 10
Faire
A(i) = 2;
i = i + 1;
Fait
Pour i = 1 à 10
Faire
B(i) = 5;
i = i + 1;
Fait
```

```
0 R1 ← 1
1 R2 ← 10
2 comp R1, R2
3 si R1 > R2 aller à 7
4 A(R1)← 2
5 R1 ← R1 + 1
6 aller à 2
```

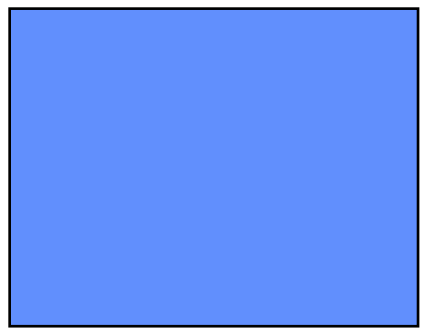
Code M1



Données M3 vecteur A

```
7 R1 ← 1
8 R2 ← 10
9 comp R1, R2
10 si R1 > R2 fin
11 B(R1)← 5
12 R1 ← R1 + 1
13 aller à 9
```

Code M2



Données M4 vecteur B

Références aux « morceaux » : $11(11311)^{10}22(22422)^{10}$

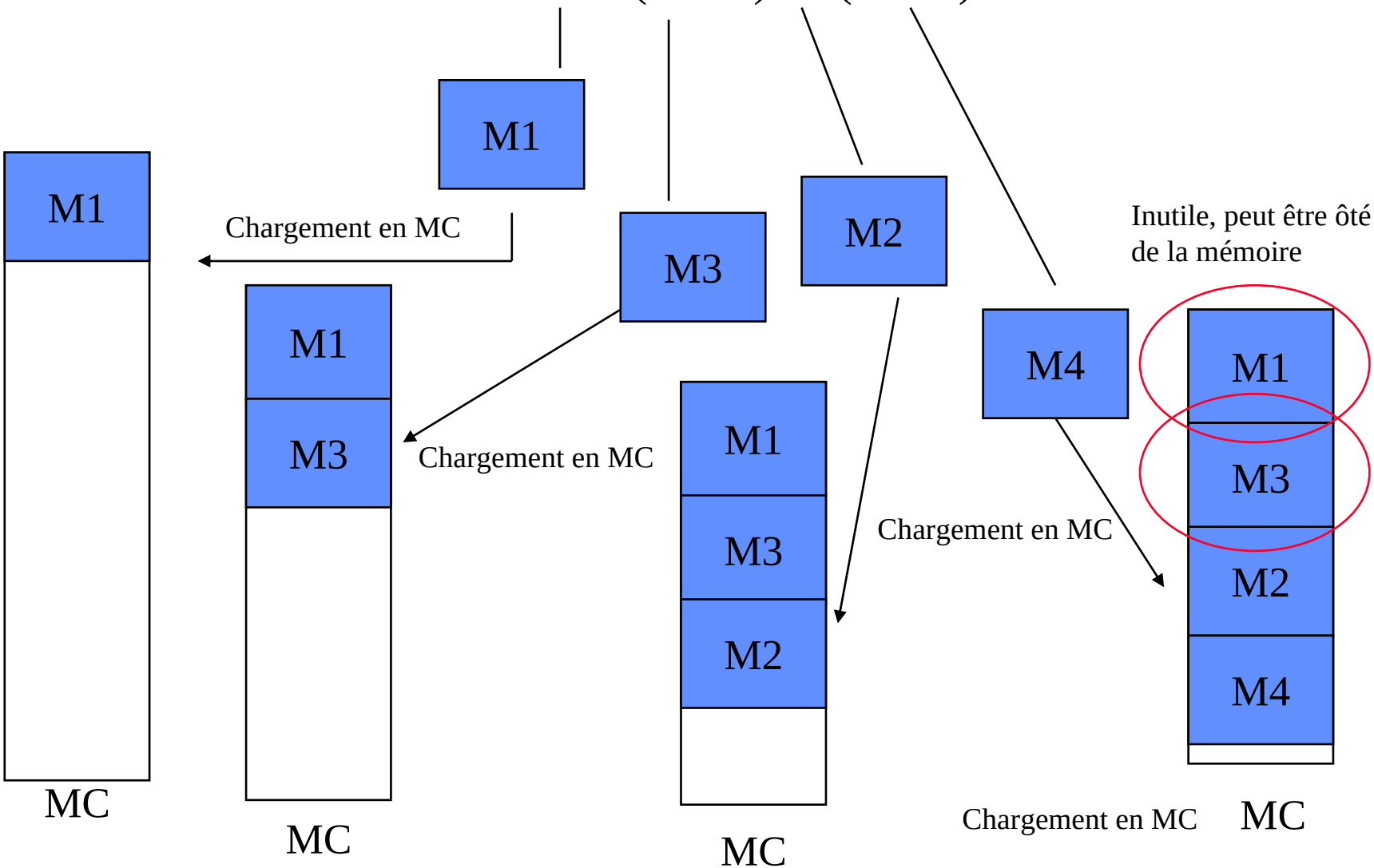
M2, M4 inutile

M1,M3 inutile

Gestion de ma mémoire centrale : un exemple

Seules les parties utiles d'un programme à t sont en mémoire centrale

Références aux « morceaux » : $11(11311)^{10}22(22422)^{10}$



FONCTIONS D'UN SYSTEME D'EXPLOITATION

- Gestion des objets externes
  - La mémoire centrale est une mémoire volatile. Aussi, toutes les données devant être conservées au delà de l'arrêt de la machine, doivent être stockées sur une mémoire de masse (disque dur, disquette, cédérom...)

    ↳ la gestion de l'allocation des mémoires de masse ainsi que l'accès aux données stockées s'appuient sur la notion de **fichiers** et de **système de gestion de fichiers (SGF).**

Système de gestion de fichiers

Le fichier est l'unité de conservation sur le support de masse
Le répertoire du disque permet de connaître les fichiers présents sur le support de masse

Programme :

Pour les élèves
Lire (nom_élève);
Lire (note_examen);
Écrire (table);
fait
Sauvegarder (mon_fichier)

totot	**12**
dupont	**20**
cesar	09
cléopâtre	15

Programme instructions

Programme donnée

SGF
Créer fichier
Lire fichier
Ecrire fichier
…

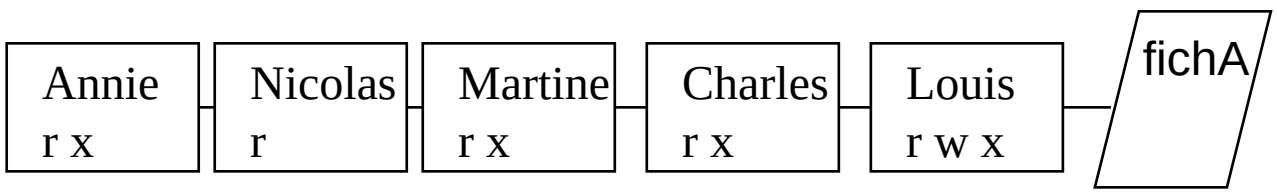
Catalogue du disque (répertoire)
Emplacement des fichiers

00110

Fichier physique
Ensemble de blocs physiques (formatage)

FONCTIONS D'UN SYSTEME D'EXPLOITATION

- Gestion de la protection
  - le système doit fournir des mécanismes garantissant que ses ressources (CPU, mémoire, fichiers) ne peuvent être utilisées que par les programmes auxquels les **droits** nécessaires ont été accordés.
  - Exemple : droits en lecture(r), écriture (w), exécution (x) pour un fichier

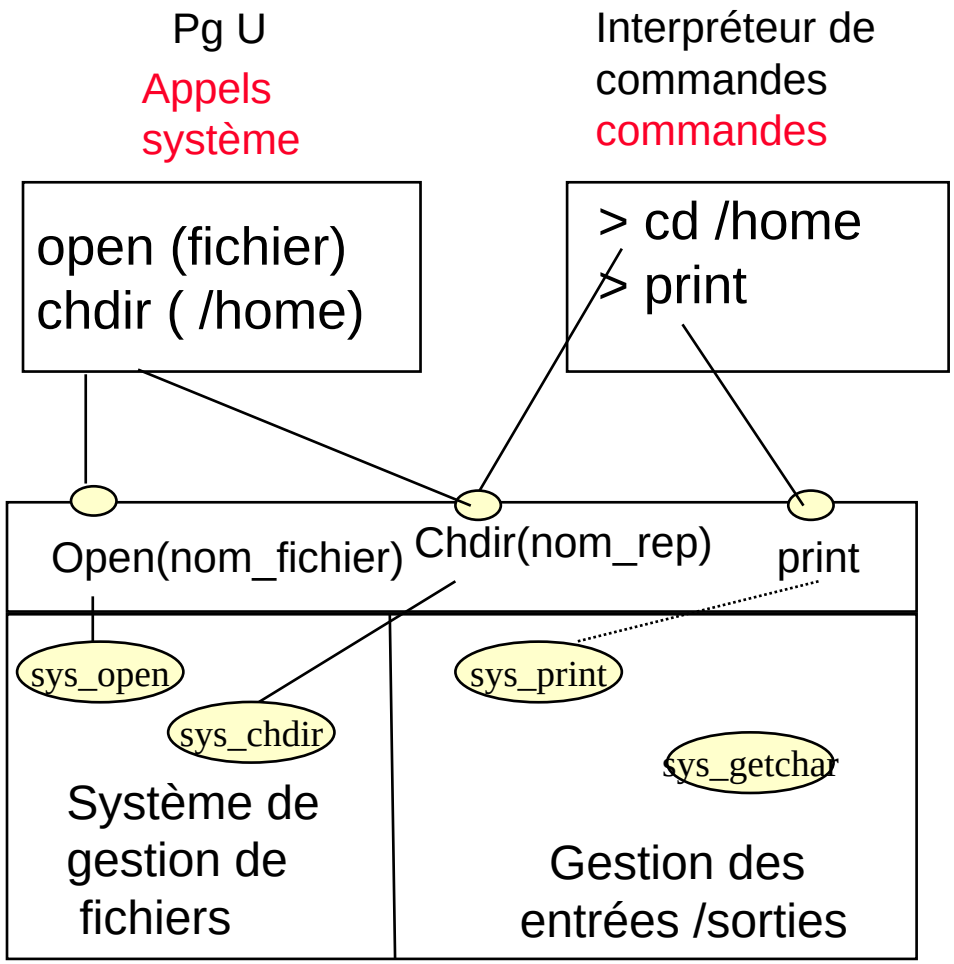


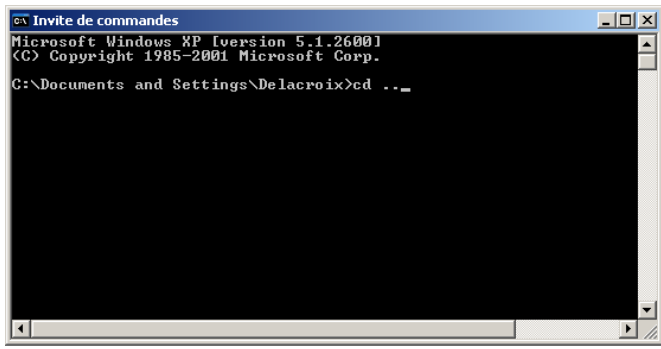
Notions de base

- Modes d’exécutions
- Interruptions logicielles et matérielles

Appels système et commandes

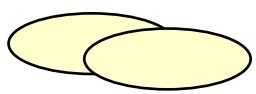
- Les fonctionnalités du système d'exploitation sont accessibles par le biais des **commandes** ou des **appels système**



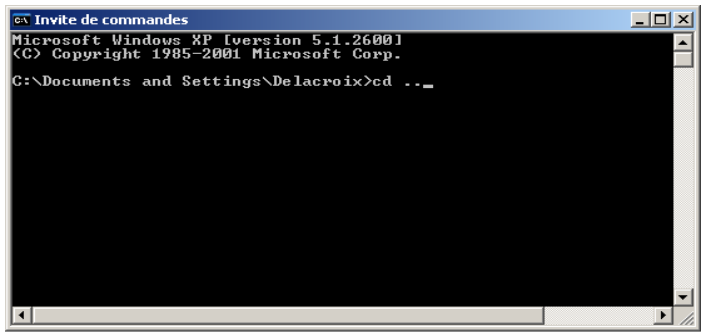


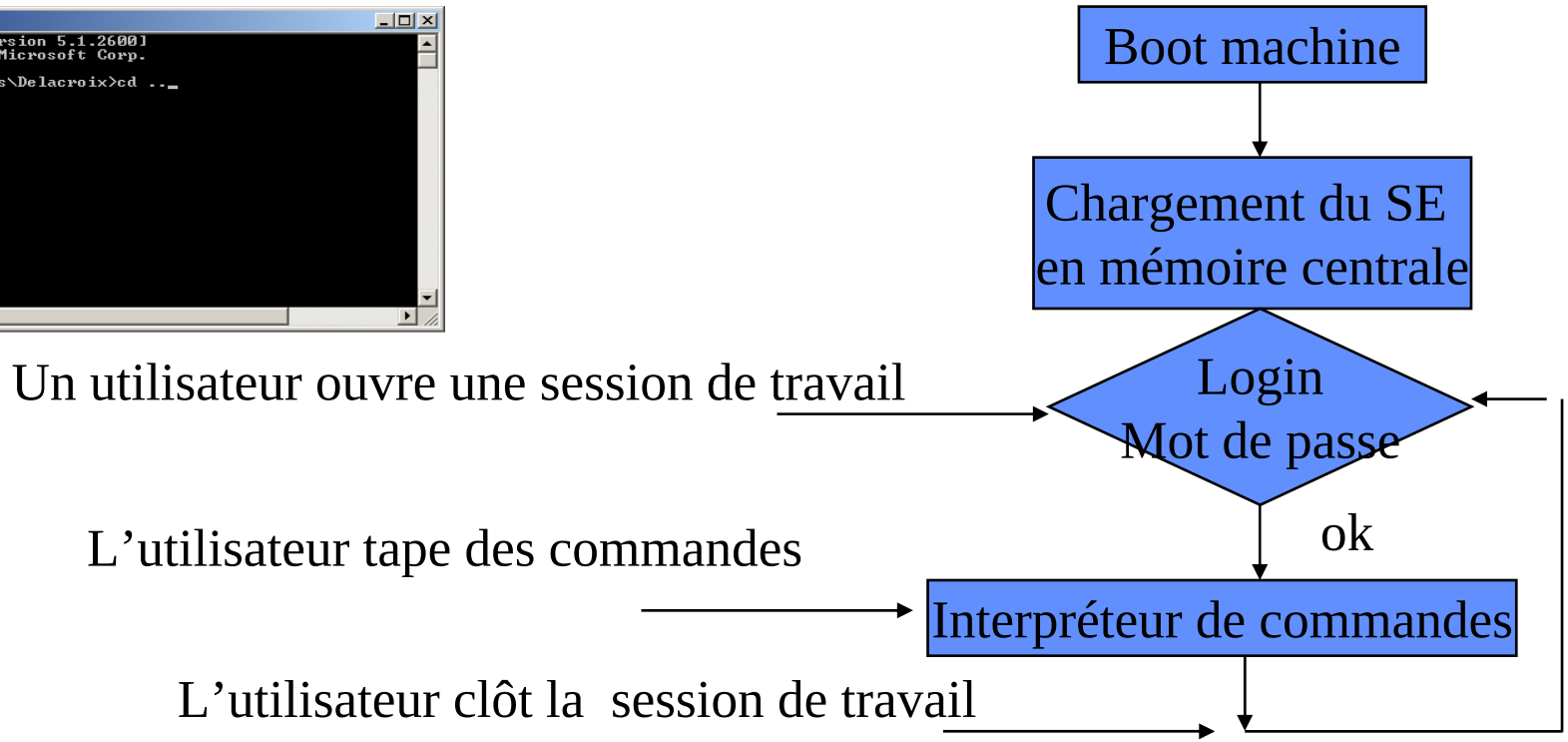
Interface d'appel prototype

SE : ensemble de fonctions sys_X



L’interpréteur de commandes





Un **interpréteur de commandes** est un programme du système d’exploitation. Son rôle est de traiter les commandes tapées au clavier par l'utilisateur.

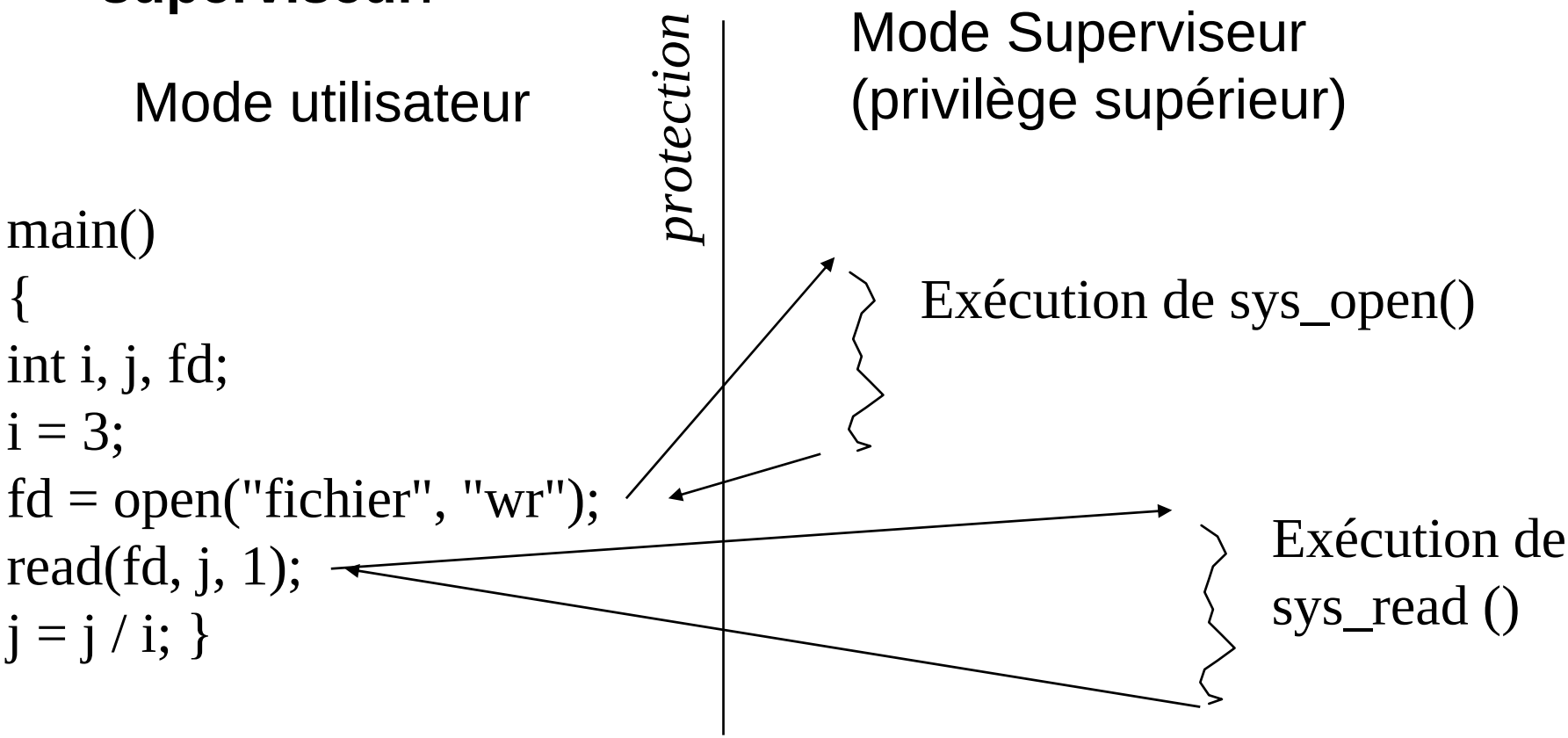
```
TANT QUE l'utilisateur ne ferme pas la session
FAIRE
Emettre un signe d'invite ;
Lire la ligne courante
Exécuter la commande indiquée sur cette ligne
FIN
```

Modes d'exécutions

- Lors de l'appel a une fonction du système, le programme utilisateur passe d'un **mode d'exécution dit utilisateur** à un **mode d'exécution dit superviseur**.

Mode utilisateur

Mode Superviseur
(privilège supérieur)



Modes d'exécutions

- Le mode superviseur est le mode d'exécution du système d'exploitation. C'est un mode privilégié.

- Le mode utilisateur est le mode d'exécution de tout programme utilisateur. Ce mode est restrictif; des instructions machine sont interdites (manipulation des Interruptions notamment).

- Le passage entre les deux modes est appelé changement de contexte.

- Les modes d' exécution sont codés dans le registre d'état du processeur

----------			P	O		I		S	Z						C
31 …….	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Exemple processeur 486

Modes d'exécutions

- Le passage du mode utilisateur au mode superviseur s'accompagne d'opérations de **commutation de contexte : sauvegarde de contexte utilisateur**

Mode utilisateur

protection

Mode Superviseur
(privilège supérieur)

```
main()
{
int i, j, fd;
i = 3;
fd = open("fichier", "wr");
```

Exécution de sys_open()

sauvegarde CO, PSW utilisateur
chargement CO ← adresse de la fonction open
chargement PSW ← mode superviseur

Modes d'exécutions

- Le passage du mode superviseur au mode utilisateur s'accompagne d'opérations de **commutation de contexte : restitution de contexte utilisateur**

Mode utilisateur

protection

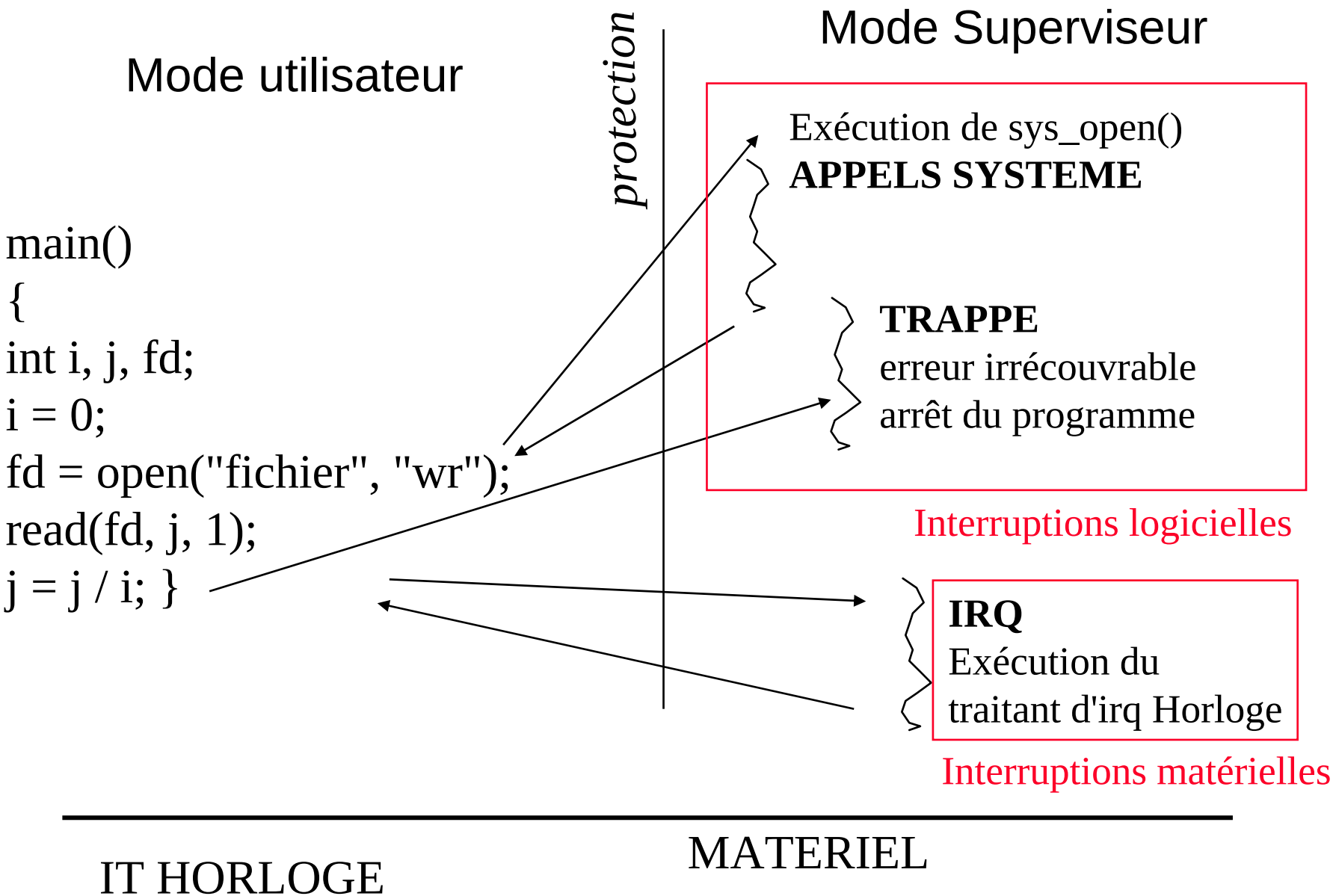
Mode Superviseur
(privilège supérieur)

Exécution de sys_open()

```
main()
{
int i, j, fd;
i = 3;
fd = open("fichier", "wr");
```

restitution du contexte utilisateur
chargement CO ← CO sauvegardé
chargement PSW ←PSW sauvegardé

Passage en mode superviseur d'un programme utilisateur



Notion d'interruptions

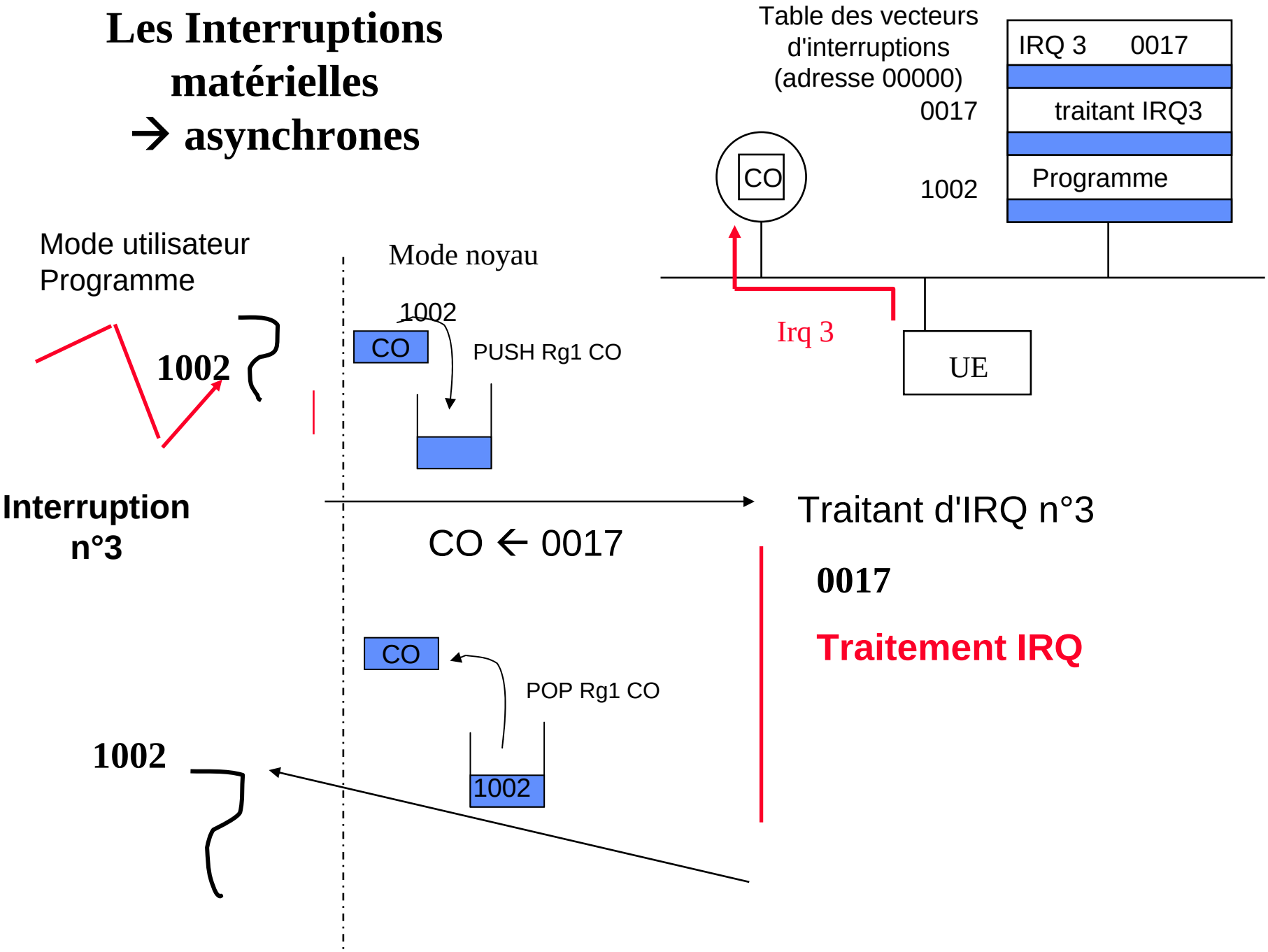
- Une interruption est un mécanisme permettant de stopper l'exécution du programme en cours afin d'aller exécuter une tâche jugée plus prioritaire.
- Elle est caractérisée par un numéro et un traitement associé (la routine ou traitant d'interruption)

On distingue principalement deux types d'événements :

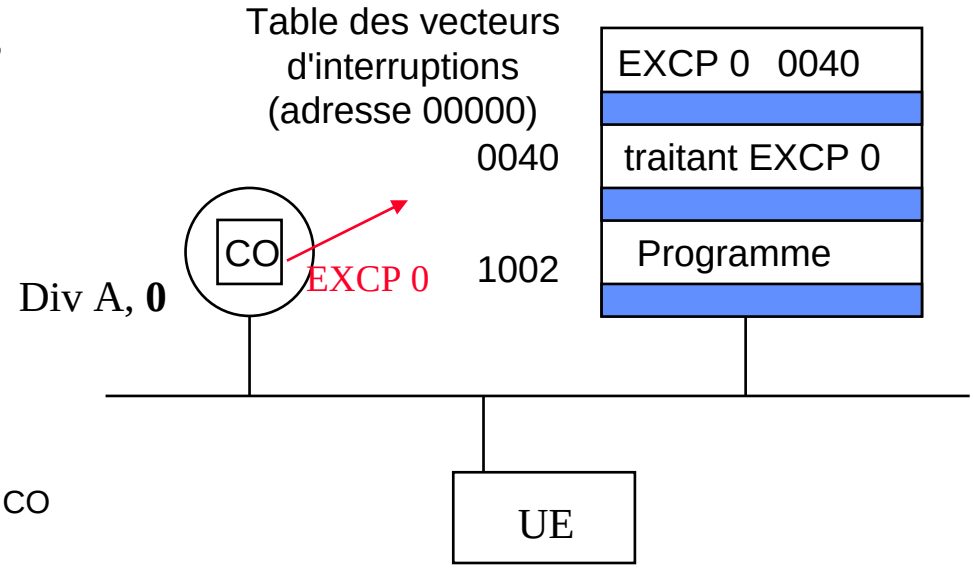
- Les interruptions externes ou matérielles sont émises par les périphériques du processeur de manière asynchrone (fin d'écriture disques, plus de papier imprimante…). Ce sont les IRQs.

- Les interruptions internes ou logicielles sont émises par le processeur lui-même lorsqu'il rencontre une erreur dans l'exécution du programme (division par zéro, accès mémoire illégal). Ce sont les trappes ou exceptions. Elles sont synchrones avec l'exécution en cours.

Les Interruptions matérielles → asynchrones



Les Interruptions logicielles
- trappes
→ synchrones



Mode utilisateur
Programme

I = 0;
A = 12,

A = A / I ; **1002**

Levée exception 0

Mode noyau

1002
CO
PUSH Rg1 CO

CO ← 0040

Traitant Exception 0

0040

Traitement Exception 0
Arrêt du programme

Création d'une image du programme fautif (Core) analysable par un outil débogueur

Les interruptions sous Intel x386

Chaque interruption est identifiée par un numéro sur 8 bits, il y a 256 interruptions différentes.

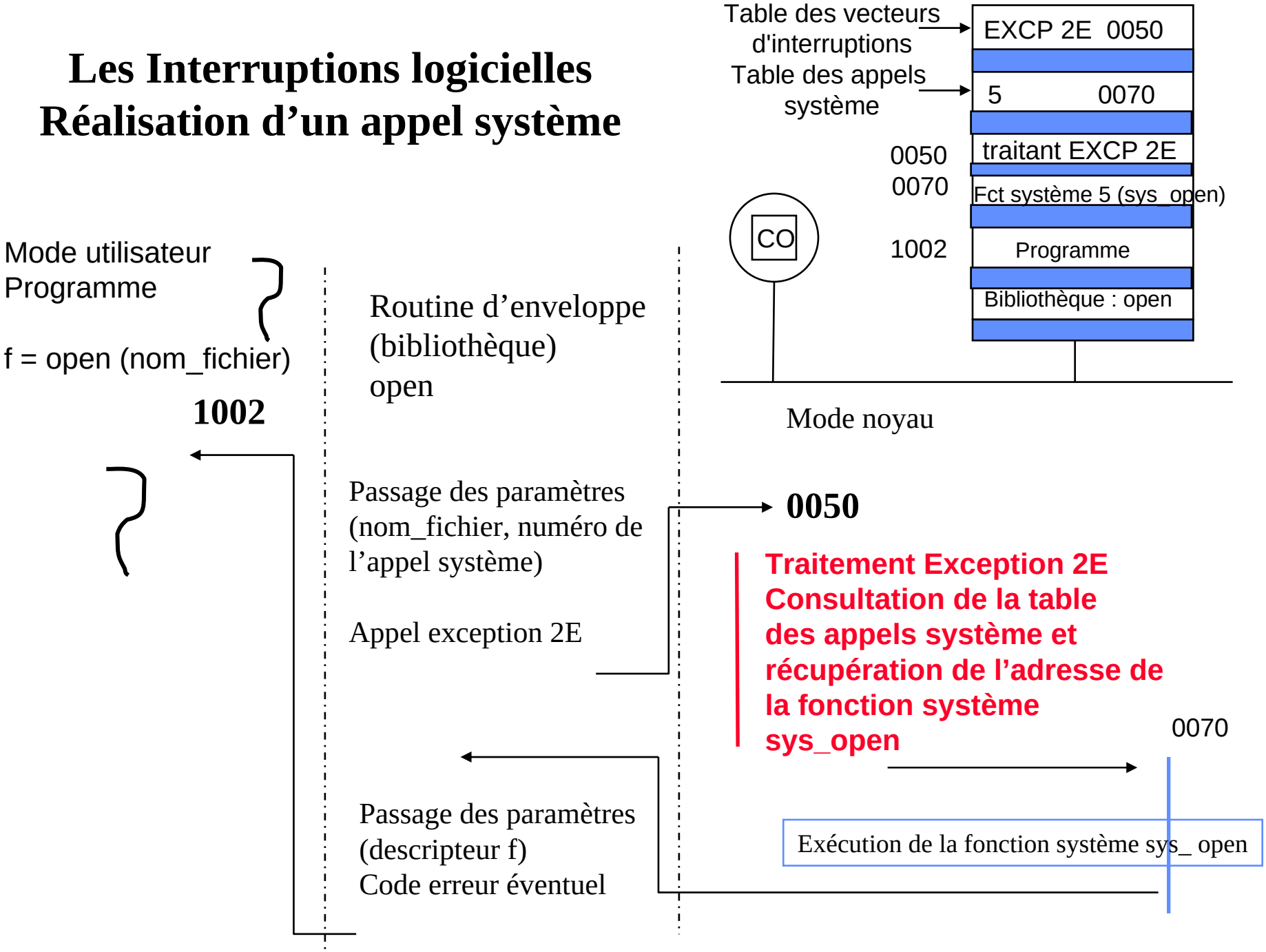
- 32 trappes par défaut du processeur
    - 0 : division par 0;
    - 6 : instruction inconnue ;
    - 13 : accès mémoire illégal.
- 16 interruptions matérielles;
- interruptions logicielles supplémentaires
    - appel système

Le numéro indexe une entrée de la *table des vecteurs d'interruptions* qui associe au numéro l'adresse du gestionnaire de traitement.

Réalisation d'un appel à une fonction du système

- Les fonctions du système sont exécutées en mode noyau.

- Le basculement du mode utilisateur au mode noyau s'effectue par la levée de l'exception particulière (Exception 2Eh sous Intel) réalisée au sein d'une routine d'enveloppe stockée dans une bibliothèque des appels système.

- Chaque fonction du système est connue au niveau du système d'exploitation, par un numéro (exemple open, 5). Une table des appels système associe à chaque numéro de fonction, son adresse en mémoire centrale

Les Interruptions logicielles
Réalisation d'un appel système