




**Module: Systèmes D'Exploitation**

# Théorie des systèmes d'exploitation

---

**Abdellah OUAGUID**  
 Laboratoire : Informatique Intelligence Artificielle et  
 Cyber Sécurité (2IACS)  
 ENSET, Université Hassan II Casablanca, Maroc  
 Email : [ouaguid@enset-media.ac.ma](mailto:ouaguid@enset-media.ac.ma)



<b>Généralités sur les systèmes d'exploitation</b>	01	 <h2 style="margin: 0;">Théorie des systèmes d'exploitation (prévisionnel)</h2>
<b>Définitions, Fonctions et Types de Système d'exploitation</b>	02	
<b>Architecture et concepts des systèmes d'exploitation</b>	03	
<b>Principaux services des systèmes d'exploitation</b>	04	
<b>Gestion des processus</b>	05	
<b>Ordonnancement</b>	06	
<b>Communication interprocessus</b>	07	
<b>Gestion de la mémoire</b>	08	
<b>Gestion de fichiers</b>	09	
<b>Gestion d'Entrées/Sorties</b>	10	
<b>Les interruptions</b>	11	

Systèmes d'exploitation
2
Copyright (C) 2025 - All rights reserved.

# Travaux dirigés

## TD – Exercice 1 : Eléments de réponse

Exercice 1 :

1.

**fork()** est un appel système utilisé pour créer un nouveau processus, qui est un duplicata du processus appelant (processus père).

L'appel retourne une valeur différente dans le processus parent et le processus enfant :

- dans le processus enfant, il retourne **0**,
- tandis que dans le processus parent, il retourne le **PID** du processus enfant.
- Il retourne aussi **-1** en cas d'erreur.

## TD : Eléments de réponse

Exercice 1 :

2.

<p>Je suis le père avec pid 1901 Je suis le fils avec pid 1902 user@ubuntu:~\$</p>	<p>Le père a précédé le fils dans l'exécution puis le fils s'est exécuté (cette exécution est la plus répandue).</p>
<p>Je suis le fils avec pid 1902 Je suis le père avec pid 1901 user@ubuntu:~\$</p>	<p>le fils a précédé le père dans l'exécution puis le père a terminé son exécution.</p>

## TD : Eléments de réponse

Exercice 1 :

2.

Je suis le père avec pid 1901  
user@ubuntu:~\$ Je suis le fils avec pid 1902

- Sur un système **lent**, on peut avoir cet output
- La console a réaffiché l'invite (**user@ubuntu:~\$**) avant l'affichage du second message. Ceci s'explique par le fait que le processus père, qui a été lancé à partir de la console, **s'est terminé avant le processus fils**, et a donc redonné le contrôle à la console avant que le fils ne puisse afficher son message.

## TD : Eléments de réponse

Exercice 1 :

3.

```

1 #include <sys/types.h>
2 #include <sys/wait.h> // Nécessaire pour wait()
3 #include <unistd.h>
4 #include <stdio.h>
5
6 int main() {
7     pid_t fils_pid;
8     fils_pid = fork();
9
10    if (fils_pid == 0) {
11        // Ceci est exécuté par le processus enfant.
12        printf("Je suis le fils avec pid %d\n", getpid());
13        exit(0); // Utilisation de exit() pour terminer immédiatement le processus enfant
14    } else if (fils_pid > 0) {
15        int status;
16        // Ceci est exécuté par le processus parent.
17        printf("Je suis le père avec pid %d\n", getpid());
18        wait(& status); // Le parent attend ici que le processus enfant se termine.
19        printf("Le processus enfant avec pid %d s'est terminé avec le statut %d\n", fils_pid, WEXITSTATUS(status));
20    } else {
21        // fork() retourne -1 en cas d'erreur.
22        printf("Erreur dans la creation du fils\n");
23    }
24
25    return 0;
26 }

```

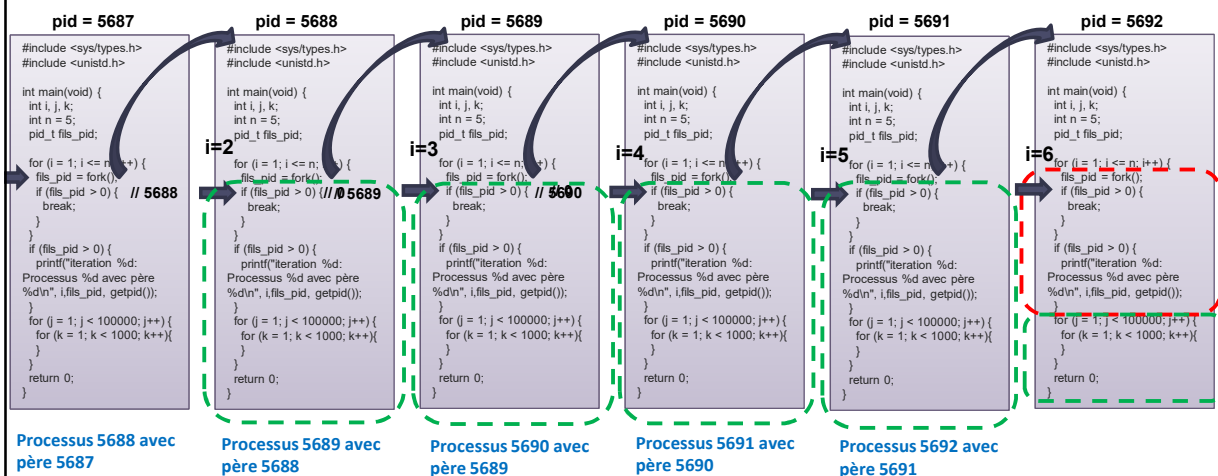
Systèmes d'exploitation

7

Copyright (C) 2025 - All rights reserved.

## TD : Eléments de réponse

Exercice 2 :



Systèmes d'exploitation

8

Copyright (C) 2025 - All rights reserved.

## TD : Eléments de réponse

Exercice 2 :

- Les parents peut afficher les messages (5 au total) pour les enfants dans **l'ordre de leur création** .

Processus 5688 avec père 5687

Processus 5689 avec père 5688

Processus 5690 avec père 5689

Processus 5691 avec père 5690

Processus 5692 avec père 5691

- Ou bien les parents peuvent afficher leurs messages dans **un ordre différent**, selon **l'ordonnancement** du système d'exploitation. Mais en fin de compte, le nombre total de messages qui seront affichés **sera de 5**, tout comme dans le premier scénario.

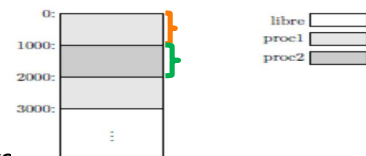
-8-

# Gestion de la mémoire

## Définition

- **Objectif** : Gérer les ressources de la mémoire et de les affecter aux différents processus en cours d'exécution

- Un processus **doit être chargé** dans la mémoire centrale pour être exécuté
- Ce processus (proc1) sera alors situé (par exemple) aux adresses physiques 0 à 999.
- Un deuxième processus se verra **attribué** à espace mémoire correspondant aux adresses physiques 1000 à 1999
- Une fois le processus est **terminé**, l'espace mémoire qui lui est alloué est donc **libéré**



## Définition

- De façon générale, plus la quantité de mémoire est **importante**, plus vous pouvez lancer d'applications **simultanément**.
- D'autre part, plus celle-ci est **rapide** plus votre système **réagit vite**,
- La gestion de la mémoire est un difficile compromis entre les **performances** (temps d'accès) et la **quantité** (espace disponible).

## Les concepts de base

### 3 types de mémoires :

#### Mémoire morte (appelée également mémoire **non volatile**)

- mémoire ROM (Read-Only Memory)
- mémoire ne s'effaçant pas en absence de courant électrique
- mémoire conservant les données nécessaires au démarrage de l'ordinateur
- Temps d'accès : **100 ns à 10 ns** selon la technologie.

#### Mémoire vive (appelée également mémoire **volatile**)

- mémoire RAM (Random Access Memory)
- données s'efface en l'absence de courant électrique, 2 types de mémoire RAM : DRAM et SRAM
- temps d'accès pour la DRAM de l'ordre de **50ns**, temps d'accès pour la SRAM de l'ordre de **10ns**

#### Mémoire flash

- compromis entre la mémoire RAM et la mémoire ROM (Exemples : **SSD, clés USB, cartes SD**) :
  - non volatilité de la mémoire morte
  - accès en lecture/écriture : Lecture rapide (**~50-100 ns**), écriture plus lente (**~100 µs - 1 ms**)

## Les concepts de base

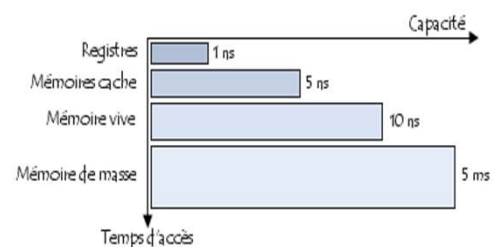
### 2 niveaux de gestion de la mémoire :

#### Niveau matériel

- les registres du processeur
- la mémoire cache

#### Niveau système d'exploitation

- la mémoire principale (appelée également mémoire centrale ou interne)
- la mémoire secondaire (appelée également mémoire de masse ou physique)

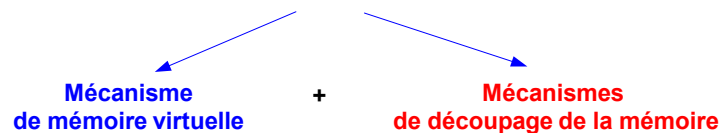


→ **Rôle du gestionnaire de mémoire du SE**

## Le gestionnaire de mémoire du SE

- **Objectifs du gestionnaire de mémoire** du système d'exploitation :

- **Partager la mémoire** (système multi-tâche)
- **Allouer des blocs de mémoire** aux différents processus
- **Protéger les espaces mémoire** utilisés
- **Optimiser la quantité de mémoire disponible**



## Le gestionnaire de mémoire du SE

### La mémoire virtuelle

La **mémoire virtuelle** est une technique permettant d'exécuter des programmes dont la taille **excède** la taille de la mémoire réelle

La **mémoire virtuelle** permet :

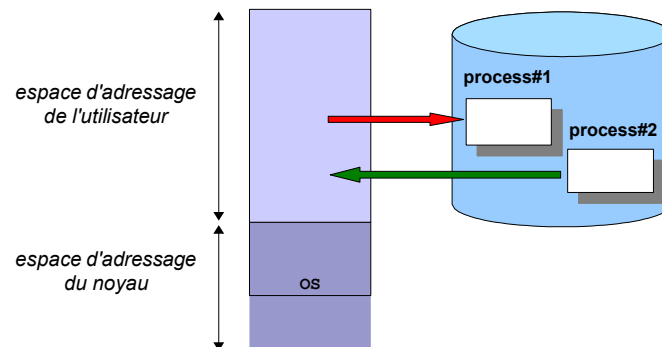
- **d'augmenter** le nombre de processus présents simultanément en mémoire centrale
- de mettre en place des mécanismes de **protection** de la mémoire
- de **partager** la mémoire entre processus



## Le gestionnaire de mémoire du SE

### La mémoire virtuelle

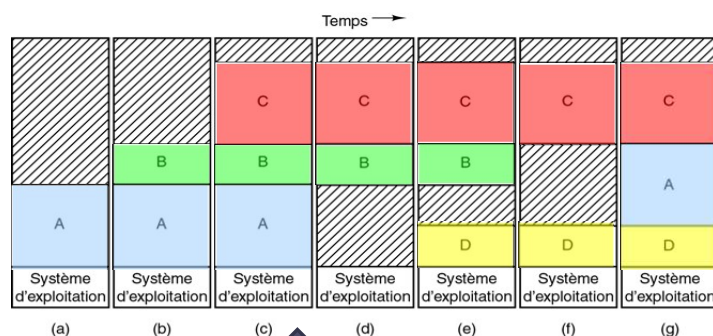
Une partie de l'espace d'adressage d'un processus peut être enlevé temporairement de la mémoire centrale au profit d'un autre (**swapping**)



## Le gestionnaire de mémoire du SE

### La mémoire virtuelle

Illustration du mécanisme de **va-et-vient** (swapping)



## Le gestionnaire de mémoire du SE

### La mémoire virtuelle

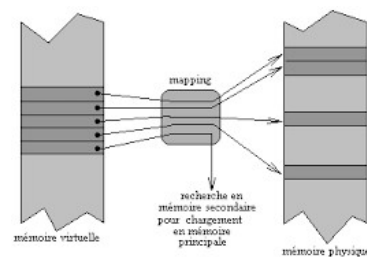
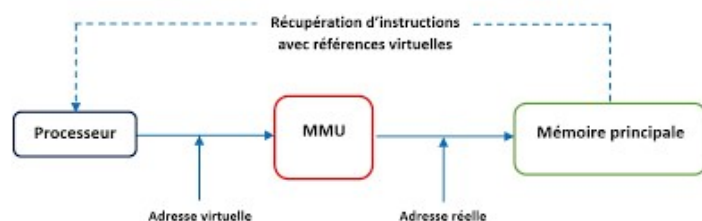
- Au lancement d'un processus, son espace d'adressage est majoritairement stocké en **mémoire secondaire**
- Au fur et à mesure de l'exécution du processus, des parties de son espace d'adressage sont chargées en **mémoire principale**

## Le gestionnaire de mémoire du SE

### La mémoire virtuelle

- Les **adresses virtuelles** doivent être traduites en **adresses physiques**
- Cette traduction est assurée par un **circuit matériel** spécifique pour la gestion de la mémoire :

→ la **MMU** (*Memory Management Unit*)



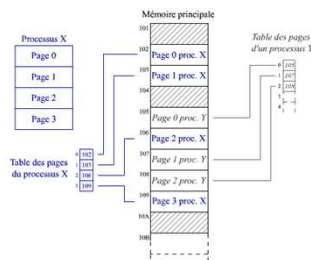
# Le gestionnaire de mémoire du SE

## Les mécanismes de découpage de la mémoire

La mémoire principale peut être découpée de 3 façons :

### Par pagination

Elle consiste à diviser la mémoire en blocs, et les programmes en pages de **longueur fixe**.



### Par segmentation

les programmes sont découpés en parcelles ayant des **longueurs variables** appelées **segments**.



**Par segmentation paginée** : certaines parties de la mémoire sont segmentées, d'autres paginées

## Problèmes communément rencontrés

**La fragmentation mémoire**



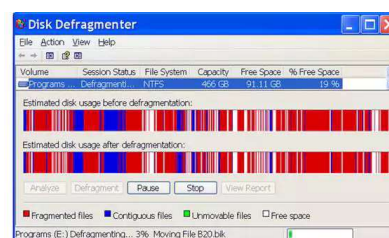
**Les défauts de pages**



## La fragmentation

### La fragmentation

- Une **mémoire fragmentée** est une mémoire dans laquelle plusieurs blocs de mémoire non contigus sont libres
  - ✓ La **fragmentation interne** dans les **systèmes paginés**
    - entre chaque partition de taille fixe, un peu de mémoire est perdue
  - ✓ La **fragmentation externe** dans les **systèmes segmentés**
    - des espaces entre les segments existent suite au retrait de programmes



## Les défauts de pages

### Les défauts de pages

- Un défaut de page est une **interruption du processeur** générée par le **matériel** lorsqu'un programme se réfère à une page de mémoire virtuelle **qui n'est pas actuellement chargée** dans la mémoire principale.
- Il se produit lorsqu'un programme tente d'accéder à une page de mémoire qui n'est pas actuellement **stockée** dans la RAM, et qui doit être **récupérée à partir** d'un stockage secondaire comme un disque dur.
- Les défauts de page sont provoqués lorsqu'un programme **tente d'accéder** à une page de mémoire qui n'est pas actuellement stockée dans la mémoire principale. Cela peut être dû à divers facteurs:
  - une quantité insuffisante de RAM
  - la fragmentation
  - ou un manque d'espace d'échange disponible, etc.

## Les algorithmes de remplacement de page

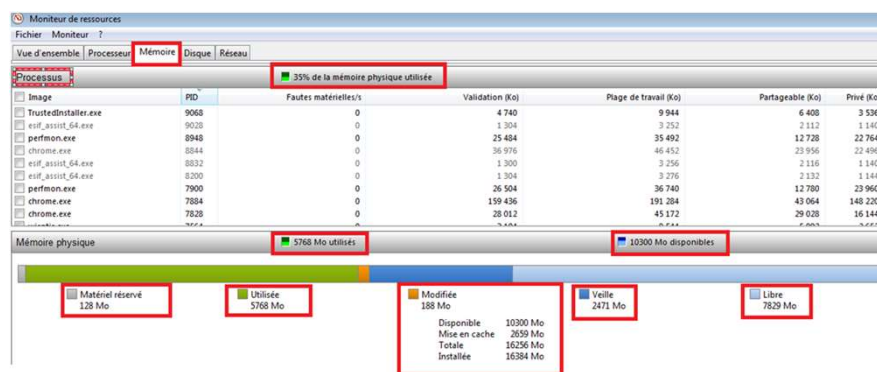
**Objectif** : choisir la page à retirer de manière à **minimiser** le nombre de défauts de page

### Algorithmes existants :

- **Algorithme FIFO** : les pages sont remplacées dans l'ordre où elles sont arrivées dans la mémoire,
- **Algorithme LRU** (Least Recently Used) : remplace la ligne **utilisée le moins récemment**. L'idée sous-jacente est de garder les données récemment utilisées,
- **Algorithme LFU** (Least Frequently Used) : garde trace de la fréquence d'accès de ces lignes et remplace la **moins fréquemment utilisée**



## Moniteur des ressources : Mémoire





**QUESTIONS**

**-9-**

**Gestion de fichiers**

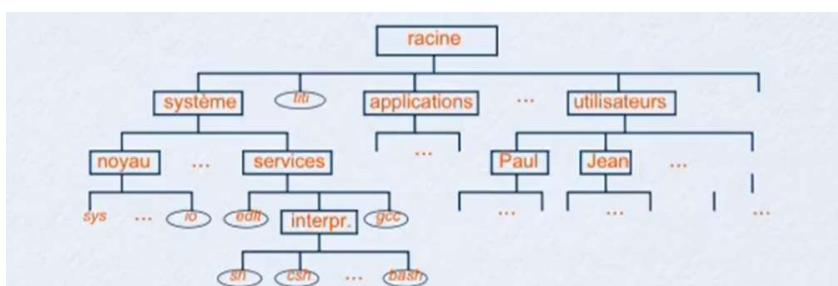
## Gestion de fichiers

- Un fichier est une **collection d'informations** liées défini par son créateur
  - ➔ Communément, les fichiers représentent les **programmes** (source ou objets) et les **données**
- L'OS est **responsable** des activités suivantes en relation avec la gestion des fichiers :
  - **Création** et **suppression** de fichiers
  - **Création** et **suppression** de répertoires
  - **Support de primitives** pour la manipulation des fichiers et des répertoires
  - **Mapper** les fichiers en mémoire secondaire
  - **Backup** de fichiers sur un média de stockage non volatile
  - ...



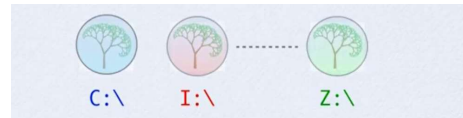
## Gestion des fichiers

- Les informations sur disque sont stockées dans des **dossiers** (répertoires) et des **fichiers**.
- Un **Système de Gestion de Fichiers** (SGF) organisé de façon hiérarchique.

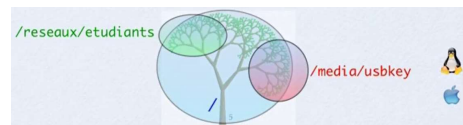


## Gestion des fichiers

- Les différents SFG peuvent être soit gérés comme des **arbres séparés** (cas de Microsoft Windows)



- Soit toute l'arborescence est intégrée en un **seul arbre**



## Gestion des fichiers

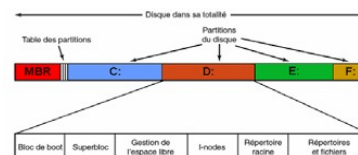
Quelques opérations basiques qu'un système de fichier peut assurer :

- création
- renommage
- copie
- suppression
- changement des droits d'accès
- compression
- recherche de fichier ou dossier
- ...



## Organisation du système de fichiers

- La structure des fichiers est en général en **arborescence**.
- Un disque dur peut être partitionné en **plusieurs partitions**, chacune contenant un système de fichiers.
- Le Secteur 0 d'un disque contient :**
  - MBR (Master Boot Record)** : Ancien format de table de partitions.
  - GPT (GUID Partition Table)** : Nouveau format, plus adapté aux disques modernes.
- Chaque partition contient :
  - Un **secteur de boot** (MBR) ou une **partition système EFI (ESP)** (GPT).
  - Les informations du **système de fichiers**
  - Les informations sur les **espaces libres et occupés**.
  - Les **fichiers et données** stockés dans la partition.



## Organisation du système de fichiers

### Comparaison MBR vs GPT :

Caractéristique	MBR	GPT
<b>Nombre max de partitions</b>	4 primaires (ou 3 primaires + 1 étendue)	Jusqu'à 128 (sur Windows)
<b>Capacité disque max</b>	2 To	9,4 Zo (Zettaoctets)
<b>Mode de démarrage</b>	BIOS (Legacy)	UEFI
<b>Redondance</b>	Une seule table de partitions	Plusieurs copies de la table de partitions

### Boot et démarrage du système

#### Démarrage avec MBR (BIOS) :

- Le BIOS charge le MBR du secteur 0.
- Le MBR contient un bootloader de 512 octets (ex: GRUB, Windows Boot Manager).
- Le bootloader charge le noyau du système d'exploitation.

#### Démarrage avec GPT (UEFI) :

- L'UEFI charge directement un fichier de démarrage dans la partition ESP (ex: \EFI\Boot\bootx64.efi).
- Pas de limite de taille ou de partitions comme en MBR.
- Plus sécurisé avec Secure Boot.

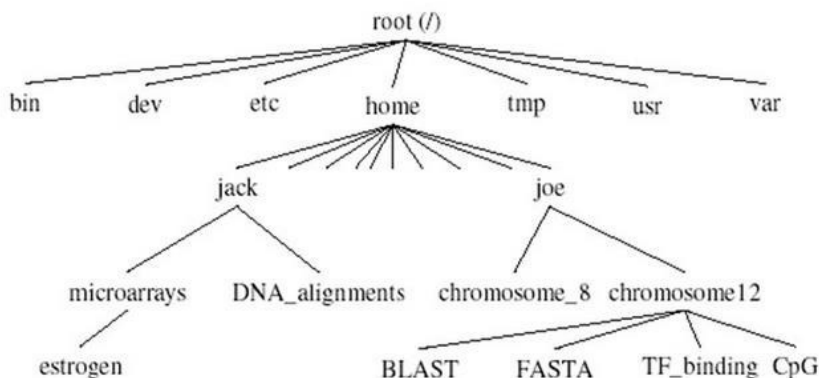
## Implantation des fichiers

Généralement, un fichier n'occupe pas une zone contiguë (consécutive) sur disque, mais un ensemble ordonné de blocs disjoints.

- **Liste chaînée**
  - Chaque fichier est représenté par une **liste chaînée** de blocs qui indique l'ordre des fragments. (Accès séquentiel **lent**, car il faut suivre la chaîne pour lire un fichier.)
- **FAT** (File Allocation table)
  - Une **table d'allocation** conserve la liste des blocs utilisés par chaque fichier. (La table peut devenir **très grande** sur un disque volumineux, ce qui impacte les **performances**.)
- **Inodes** : C'est un mélange des 2 solutions :
  - Chaque fichier a un inode qui contient des **pointeurs directs** vers les **premiers** blocs. Si l'inode est plein, des pointeurs indirects renvoient à d'autres tables pour référencer les blocs suivants. (Légèrement **plus complexe** à gérer, mais bien **plus performant**.)

## Gestion des fichiers

Structure d'un système de fichier :  
Exemple LINUX



## Gestion des fichiers

### Structure d'un système de fichier : Exemple LINUX

Chaque fichier est associée une **entrée dans la table des inodes**. Dans cette entrée, on trouve :

- **type de fichier** : fichier régulier, un répertoire, un lien symbolique,
- **bits de protection** : Ces bits déterminent les permissions ou les droits d'accès au fichier.
- **nombre de liens sur ce fichier** : Cela indique combien de noms différents (liens) pointent vers cet inode
- **taille du fichier**
- **adresses des 10 premiers blocs occupés par le fichier** : Un inode contient les adresses des blocs de disque qui stockent le contenu du fichier.
- dates et heures du dernier accès, de la dernière modification du fichier et de l'inode
- ....

## Gestion des fichiers

### Structure d'un système de fichier : Exemple Windows

- La **MFT** (Master File Table) est utilisée par NTFS pour stocker les **métadonnées des fichiers**.
- Chaque entrée de la MFT a une taille d'environ 1 Ko (variable selon la configuration) et contient : Le nom du fichier, ses attributs, et la liste des clusters utilisés.
- Les fichiers sont représentés sous forme de paires (**cluster début - cluster fin**).
- Si un fichier est très fragmenté, NTFS ajoute des **File Record Segments** (FRS) pour **stocker plus de références**.
- Les petits fichiers sont stockés **directement** dans la MFT (fichiers "résidents")

## Gestion des fichiers

### Gestion des blocs libres

- Le SE doit savoir quels sont les **blocs libres** dans le système de fichiers. On peut utiliser :
  - une **liste chaînée** des blocs libres (**UNIX**)
  - une **table de bits** (1 bit par bloc = libre/occupé) (**WINDOWS**)
- Cas des CD/DVD-ROM : c'est plus simple parce que tous les fichiers sont contigus.

# -10- Gestion d'Entrées/Sorties

## Les périphériques d'E/S (1)

- Un périphérique d'E/S est un matériel informatique assurant une **communication** entre **l'unité centrale** de l'ordinateur et le **monde extérieur**
- On distingue 3 types de périphériques :
  - Les périphériques **d'entrée**
  - Les périphériques de **sortie**
  - Les périphériques **d'entrée/sortie**
- Les périphériques peuvent être **internes** ou **externes**

## Exemples de périphériques d'entrée

### • Dispositifs de pointage



### • Dispositifs d'acquisition



## Exemples de périphériques de sortie



Systèmes d'exploitation

43

Copyright (C) 2025 - All rights reserved.

## Exemples de périphériques d'entrée/sortie



Systèmes d'exploitation

44

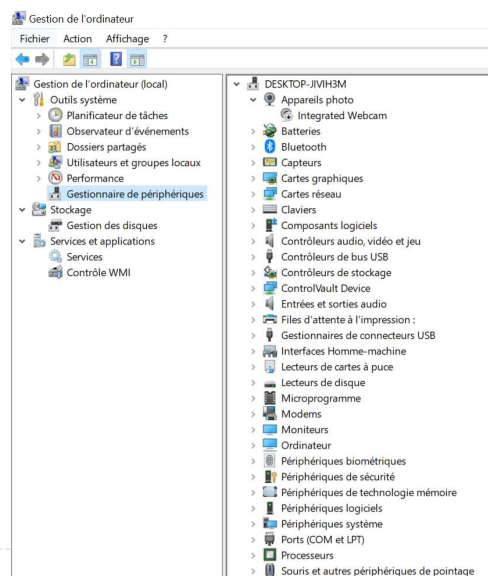
Copyright (C) 2025 - All rights reserved.

## La Gestion des Entrées/Sorties

- Les SE gèrent les **Entrées/Sorties**. On parle d'entrées-sorties dès qu'il s'agit **d'échanger** des informations entre **l'unité centrale** et les **matériels** périphériques (écran, clavier, souris, disque dur, imprimante, modem ...).
- Ces informations sont échangées par l'intermédiaire **d'interfaces** qui réalisent la conversion des données, ex : touche de clavier transformée en représentation binaire suivant le code ASCII.



## La Gestion des Entrées/Sorties



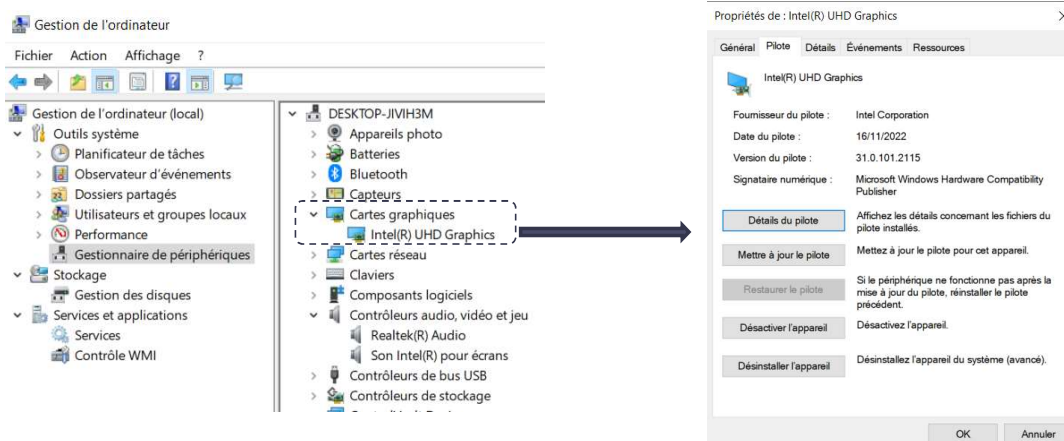
## La Gestion des Entrées/Sorties

Pour gérer les E/S sur les périphériques, les SE utilisent :

- Des programmes spécifiques aux périphériques, qu'on appelle **pilotes (drivers)**.
- Le **BIOS** sur la carte mère ou sur les différents contrôleurs d'entrées/sorties.
- Aujourd'hui, la plupart des systèmes proposent la détection et le paramétrage automatique des périphériques au moment du démarrage de la machine (**plug and play**) ou même pendant son utilisation (**hot plug**).

## La Gestion des Entrées/Sorties

### Pilotes (drivers)

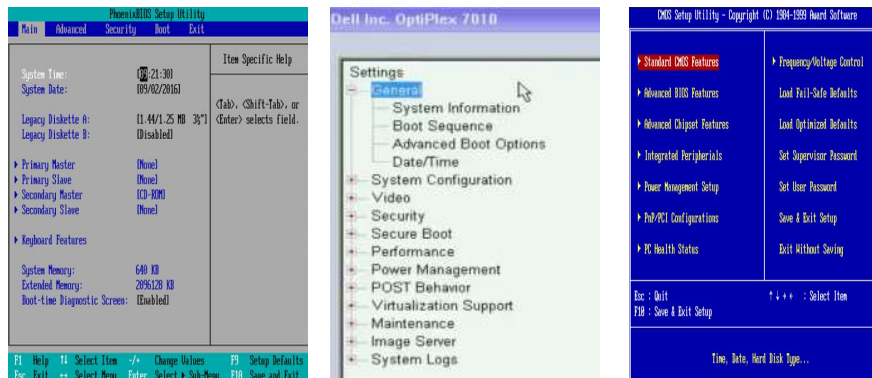




## La Gestion des Entrées/Sorties

### BIOS

Le **BIOS** (Basic Input Output System) est un programme de configuration qui permet de démarrer l'ordinateur et de reconnaître les principaux composants



Systèmes d'exploitation

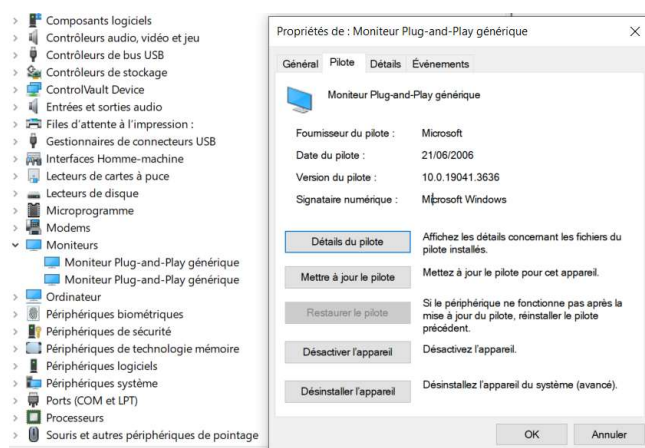
49

Copyright (C) 2025 - All rights reserved.

## La Gestion des Entrées/Sorties

### Plug and Play

- **Plug and Play** fait référence à la capacité d'un système à reconnaître et configurer automatiquement un périphérique.



Systèmes d'exploitation

50

Copyright (C) 2025 - All rights reserved.

## La Gestion des Entrées/Sorties

### Hot Plug

- Les périphériques **hot-plug** sont ceux que l'on peut connecter ou déconnecter d'un ordinateur pendant que le système est en marche. Ils sont dits connectés ou déconnectés « **à chaud** »
- Cette fonction est souvent utilisée dans des serveurs ou des systèmes nécessitant une disponibilité continue, où il **n'est pas pratique d'arrêter** le système pour ajouter ou remplacer du matériel.



Systèmes d'exploitation

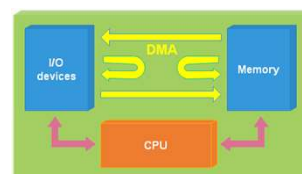
51

Copyright (C) 2025 - All rights reserved.

## La Gestion des Entrées/Sorties

- Le SE contrôle tous les organes d'E/S : il en récupère les **Interruptions (ou IRQ pour Interrupt Request)** et fournit les **fonctions d'accès**.

### Aspects matériels



#### Contrôleur de périphérique

On le pilote par ses registres internes :

- o commandes
- o états
- o Données

#### DMA (Direct Memory Access)

- o Pour faciliter les transferts d'info avec les périphériques et soulager le CPU, on utilise le **DMA**. Il s'agit d'un **dispositif physique** auquel le CPU peut demander de faire un transfert entre registres de données du contrôleur de périphériques et mémoire dans un sens ou dans l'autre  
 → Le DMA qui **règle sa vitesse sur celle du périphérique** et génère une **IT** vers le CPU quand le transfert est terminé
- o Un contrôleur de DMA offre plusieurs **canaux** → il peut faire plusieurs transferts en même temps avec plusieurs périphériques.

Systèmes d'exploitation

52

Copyright (C) 2025 - All rights reserved.

## Bus

- Un Bus est un groupement de **conducteurs électriques** permettant une **connexion physique** et le **transport** de signaux entre les différents **composants de l'ordinateur**
- Catégories de Bus :

### Direction de transmission

- **Simplex** – unidirectionnel
- **Half duplex** – bidirectionnel, une direction un certain temps
- **Full duplex** – bidirectionnel simultané

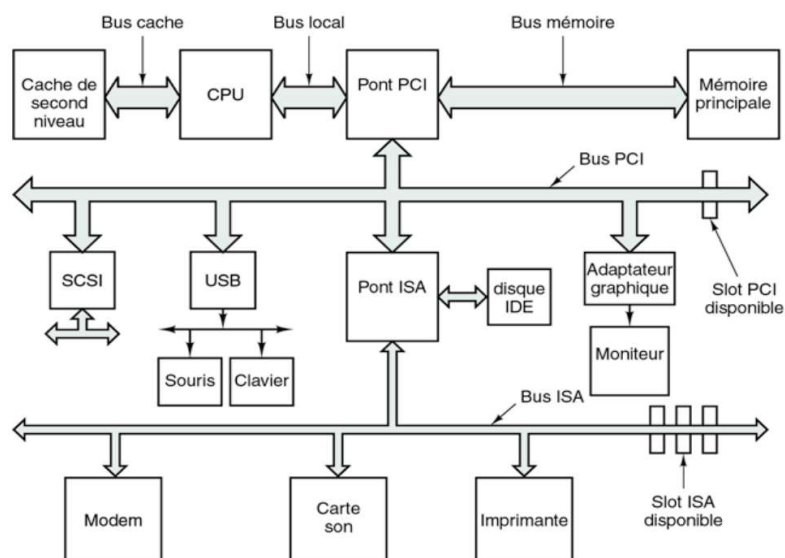
### Méthode de l'interconnexion

- **Point-à-point** – liaison: source à destination
  - Câbles – Les bus point-à-point qui connectent vers un dispositif externe
- **Bus Multipoint** – [broadcast bus]
  - Connecte les points multiple vers un autre bus
  - Type de bus dans les réseau Ethernet (nécessite une technique d'adressage ≠ Bus point-à-point)

### Architectures de communication

- Les bus **parallèles**
- Les bus **série**

## Bus



## Bus

### Les bus parallèles vs. Les bus série

- **Les bus parallèles :**
  - Ce sont des bus **simples** constitués **d'autant de fils** qu'il y a de bits à transporter.
  - Ces bus sont **coûteux** et **peu fiables** pour des distances importantes.
  - Ils sont utilisés sur des distances courtes, par exemple, pour relier le processeur, la mémoire et les unités d'échanges.
- **Les bus série :**
  - Ils permettent des transmissions sur de **grandes distances**.
  - Ils utilisent une seule voie de communication sur laquelle les bits sont sérialisés et envoyés **les uns à la suite des autres**

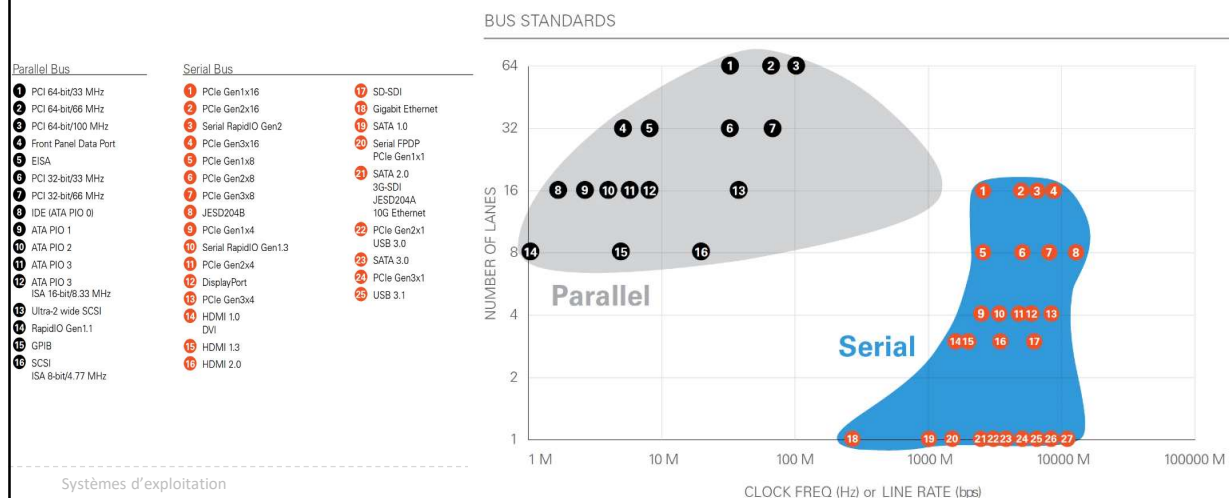
## Enjeux liés à la gestion des E/S (1)

- La gestion des E/S par le système d'exploitation est un véritable défi lié aux **différences multidimensionnelles** des périphériques :
  - rapidité du périphérique
  - volume des informations
  - service proposé
  - direction du flux d'informations
  - protocole de communication



## Enjeux liés à la gestion des E/S (2)

Débits de données de certains périphériques courants:



## Enjeux liés à la gestion des E/S (3)

- Le système d'exploitation doit :
  - ✓ offrir de bonnes **abstractions** aux programmes
  - ✓ **gouverner** leur utilisation par les processus
  - ✓ **coordonner** efficacement les périphériques
- Défi : Soucis d'**uniformisation de l'interface** offerte par le SE pour les E/S





## QUESTIONS

## Sources et références

- Systèmes d'exploitation, Gestion des processus, Cours SYE, Prof. Daniel Rossier, Version 2.3 (2009-2010)
- KHIAT, Azeddine. Cours sur les « Systèmes d'Exploitation », ENSET Mohammedia, Université Hassan 2
- M. DALMAU - IUT de Bayonne - Les systèmes d'exploitation 56
- Cours de Systèmes d'Exploitation, Hugues DELALIN, IUT de Lens - Université d'Artois
- Andrew Tanenbaum, Systèmes d'exploitation, 3<sup>ème</sup> édition,
- Laurent Bloch, Splendeurs et servitudes des Systèmes d'exploitation, Histoire, fonctionnement, enjeux
- Gael Le Mignot, Système d'exploitation Processus et threads, Pilot Systems –INSIA SRT - 2007
- Azouagh Driss , « Théorie des systèmes d'exploitation »
- Systèmes D'exploitation & Réseaux Informatiques, Pierre Nugues
- Système D'exploitation II - Errais Mohammed
- <https://tech-lib.fr/>
- Introduction aux systèmes informatiques, Stefan Monnier
- <https://www.ni.com/en/shop/electronic-test-instrumentation/digital-instruments/high-speed-serial-explained.html>
- ...