Chapitre :
Architecture
Réseau
systèmes d'exploitation

I. Architecture Von Neumann

⊛ **Activité :** Architecture des ordinateurs

Dans un système informatique on trouve des composants physiques (la partie matérielle, en anglais hardware) et des programmes ou logiciels (le software) qui s'exécutent sur cette partie matérielle. Les programmes sont plus ou moins efficaces suivant les caractéristiques du matériel. Il est donc important de connaître le fonctionnement interne d'une machine pour optimiser les algorithmes et les rendre efficaces, rapides, etc.
L'architecture des ordinateurs actuels repose sur le modèle de Von Neumann, basé sur l'idée de programme enregistré. Cela signifie que la mémoire de l'ordinateur, dans laquelle sont stockées les données, contient également les programmes qui s'exécutent. Autrement dit, programmes et données se partagent l'espace mémoire de l'ordinateur.
Une machine contient :

- Une **mémoire** qui contient à la fois les données et les programmes ;
- Une **unité arithmétique et logique** (UAL) qui effectue les calculs et les tests logiques ;
- Une **unité de contrôle** qui gère la séquentialité des opérations ;
- Des dispositifs d'**entrée et de sortie** qui permettent la communication depuis et vers l'extérieur.

Le tout est rythmé par une **horloge interne** qui détermine la fréquence du processeur.
Entre ces différentes composantes, la communication de données, d'adresses, etc. se fait par des **bus**.
Il existes différents types de mémoire, qui peuvent être placées à différents endroits dans l'ordinateur. On peut retenir que généralement les mémoires les plus proches du processeur (les **registres**) sont les plus rapides et les plus petites, quand les mémoires les plus éloignées sont les plus lentes et les plus grandes (comme les disques durs).

Le temps d'exécution a une grande importance en informatique, et la vitesse de calcul des machines n'a cessé d'augmenter. Pour cela, le principal levier était la fréquence d'horloge. Cependant cela implique un échauffement du processeur. Depuis quelques années, on augmente le nombre d'unités de calcul, autrement dit le nombre de processeurs (ou de cœurs). Un **cœur** est en fait un ensemble composé d'unités de calculs et de commandes avec des registres. Il faut alors écrire des programmes capables d'utiliser plusieurs cœurs pour partager les temps de calcul (le **parallélisme**).

⊛ **Activité :** Réalisation de circuits logiques (nécessite le logiciel logisim)

II. Réseau

Pour ce thème, lire les ressources disponibles sur Internet et les noter sur feuille de cours pour les retenir.
Par exemple, sur ces sites de cours de spécialité NSI : le site numerique-sciences-informatiques.fr et le site pixees.fr (dans la section Architectures matérielles et systèmes d'exploitation, en commençant par la partie Introduction réseau).
Utiliser éventuellement d'autres sources.
On s'attachera à retenir les informations concernant le programme officiel de la spécialité, à savoir :

- Transmission de données dans un réseau ;
- Protocoles de communication ;
- Architecture d'un réseau ;
- Mettre en évidence l'intérêt du découpage des données en paquets et de leur encapsulation ;
- Dérouler le fonctionnement d'un protocole simple de récupération de perte de paquets, celui du bit alterné (voir plus loin) ;
- Simuler ou mettre en œuvre un réseau. On trouve une vidéo d'utilisation du logiciel Filius sur le site pixees.fr, qu'il suffira en première de regarder. Le logiciel pourra être utilisé en terminale.

En particulier, les éléments suivants seront à savoir et à comprendre :

- Les différentes tailles de réseau (PAN, LAN,...) ;
- Les différentes topologies de réseau (étoile, maillé,...) ;
- Les deux modèles pour les réseaux : TCP/IP et OSI. Connaître un peu leurs couches, et en particulier leur nombre ;
- À propos de l'encapsulation, savoir ce que sont et comment s'imbriquent trames, paquets et segments ; connaître un peu leur contenu, en particulier quels types d'adresses sont données dans chacun. Savoir dans quelles couches du modèle OSI ils sont traités ;
- Savoir quelle est la forme d'une adresse IP, en particulier IPv4. La connaissance de la forme des adresses IPv6 n'est pas requise mais conseillée ;
- Savoir également ce qu'est une adresse MAC et savoir à quelle couche du modèle OSI elle est utilisée (et le savoir aussi pour l'adresse IP) ;
- Savoir ce qu'est un masque de sous-réseau (défini plus loin) et savoir calculer le nombre de machines que l'on peut connecter sur un réseau étant donné son masque de sous-réseau. Savoir en particulier ce qu'est l'adresse de broadcast ;
- Connaître ce que sont les protocoles TCP et UDP et leurs différences. Savoir à quelle couche du modèle OSI ils correspondent ;
- Connaître la différence entre les hubs, les switchs (commutateurs) et les routeurs ;
- Connaître le protocole du bit alterné (défini plus loin).

Une fois ces informations lues et notées, faire la fiche d'exercice sur les réseaux.
Un devoir surveillé avec des questions sur les points à connaître permettra de vérifier que les connaissances ont bien été retenues et comprises.

1. Masque de sous-réseau et notation CIDR

Historiquement, on appelle sous-réseau chacun des réseaux connectés à Internet.
Les machines d'un même réseau partagent le même début de leur adresse IP.
Plus précisément, l'adresse IPv4 d'une machine d'un réseau est composée de deux parties :

- celle du (sous-)réseau ;
- celle de l'hôte (qui identifie de manière unique la machine sur le réseau).

Un **masque de sous-réseau** est une information donnée en plus d'une adresse IP afin de déterminer quelle est la partie réseau et quelle est la partie hôte dans l'adresse de la machine.
Rappelons qu'une adresse IPv4 s'écrit sous la forme `A.B.C.D` avec `A`, `B`, `C` et `D` écrits sous forme décimale et prenant des valeurs allant de 0 à 255, mais qu'il s'agit en fait de la donnée de 4 octets, donc de $4 \times 8 = 32$ bits.
En binaire, un masque de sous-réseau est une suite de 32 bits qui commence par des 1 (indiquant la partie réseau) et termine par des 0 (indiquant la partie hôte).

Exemple Soit une machine dont l'adresse IPv4 est :

`124.32.197.6`

En binaire, son adresse est :

`01111100.00100000.11000101.00000110`

Considérons par exemple que le masque de sous-réseau soit :

`11111111.11111111.11110000.00000000`

On a bien une suite de 1 (pour la partie réseau) puis une suite de 0 (pour la partie hôte).

L'adresse du réseau est alors celle obtenue à partir de l'adresse IP de la machine en conservant les bits de la partie réseau, et en remplaçant par des 0 ceux de la partie hôte.
Autrement dit, l'adresse du réseau de l'exemple est :

`01111100.00100000.11000000.00000000`

En notation décimale, le masque de sous-réseau a pour adresse :

`255.255.240.0`

C'est sous cette forme que l'on donne généralement le masque de sous-réseau.
L'adresse du réseau est, quant à elle :

`124.32.192.0`

Plutôt que de donner le masque de sous-réseau sous forme d'une adresse IP (`255.255.240.0`), on peut utiliser une notation, appelée **CIDR**, dans laquelle, en plus de l'adresse IP de la machine, on indique le nombre de 1 du masque de sous-réseau.
Dans notre exemple, le nombre de 1 du masque de sous-réseau est 20.
La notation CIDR de l'adresse de la machine est alors :

`124.32.197.6/20`

Il n'est en fait pas fréquent que la longueur de la partie hôte soit égale à 20 comme dans l'exemple. Généralement, il s'agit d'un multiple de 8 (on prend des octets complets pour l'adresse du réseau). Cela simplifie l'obtention de l'adresse du masque de réseau en notation décimale car un octet ne contenant que des 1 vaut 255. De même, il est immédiat de donner l'adresse réseau.

Exemples

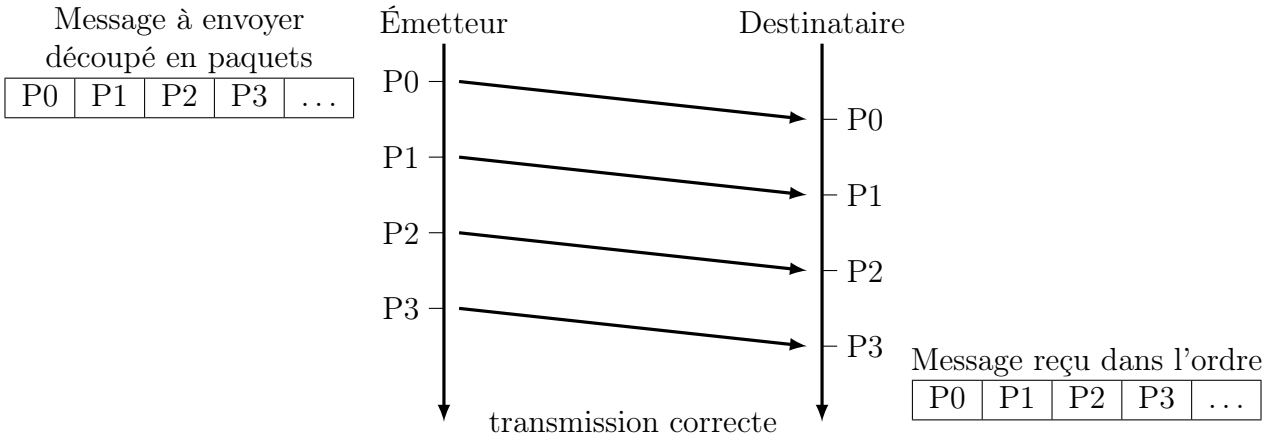
- Pour une adresse machine `124.32.197.6/8`,
  le masque de sous-réseau a pour adresse `255.0.0.0`
  et le réseau a pour adresse `124.0.0.0`
- Pour une adresse machine `124.32.197.6/16`,
  le masque de sous-réseau a pour adresse `255.255.0.0`
  et le réseau a pour adresse `124.32.0.0`
- Pour une adresse machine `124.32.197.6/24`,
  le masque de sous-réseau a pour adresse `255.255.255.0`
  et le réseau a pour adresse `124.32.197.0`

Le nombre de machines que l'on peut connecter sur un réseau dépend du nombre de bits de la partie hôte. À retenir : deux adresses ne peuvent pas être données à une machine : celle du réseau lui-même, et celle de broadcast (où l'on remplace tous les 0 de la partie hôte par des 1).

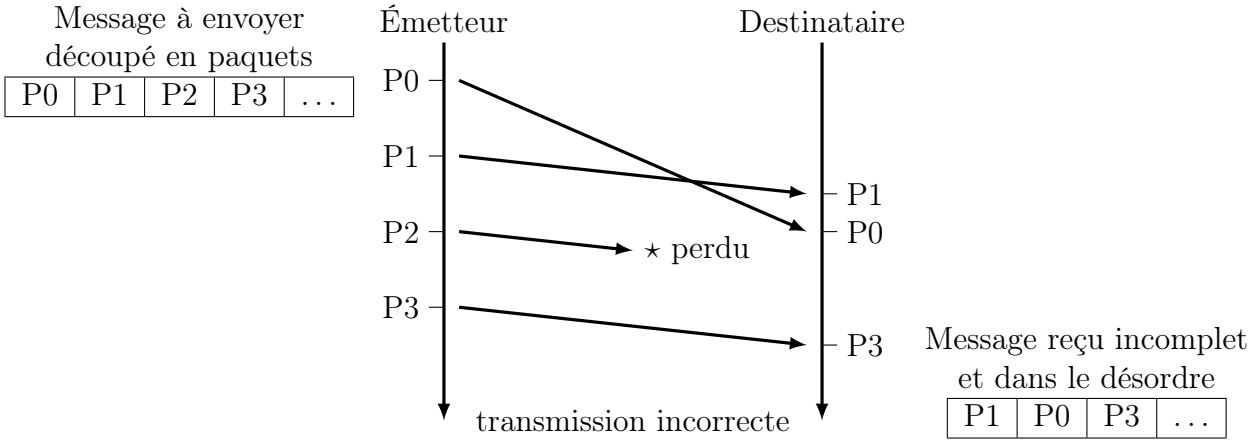
2. Protocole du bit alterné

Quand on veut envoyer un message, celui-ci étant généralement assez grand, il est coupé en plusieurs morceaux (paquets, ou trames).

Dans l'idéal, on envoie les paquets dans l'ordre, et le destinataire les reçoit tous, et dans l'ordre :



Dans la réalité, même si les paquets sont envoyés dans l'ordre, ils n'arrivent pas forcément dans le même ordre (la route empruntée n'est pas toujours la même, par exemple), et il arrive même que des paquets soient perdus :

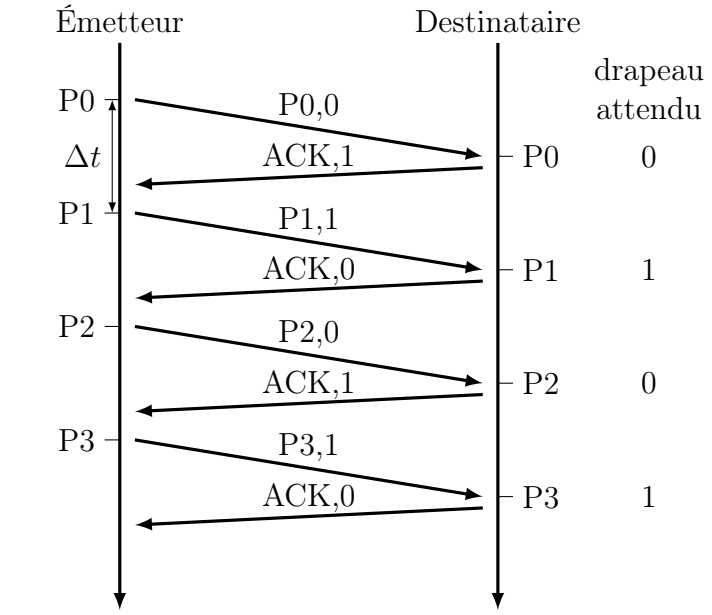


Pour tenter de résoudre les problèmes de perte de paquet et de temporalité, le **protocole du bit alterné** utilise plusieurs outils :

- l'émission par le destinataire d'un signal d'accusé de réception, noté ACK (pour *acknowledgement* en anglais) afin que l'émetteur sache que son envoi a réussi : dès que l'émetteur a reçu un paquet, il émet le signal ACK en retour ;
- l'utilisation d'une horloge du côté de l'émetteur pour estimer qu'un problème de réception (de paquet chez le destinataire ou de ACK chez l'émetteur) a eu lieu : si, au bout d'une durée fixée à l'avance Δt après l'envoi d'un paquet, l'émetteur n'a toujours pas reçu le signal ACK correspondant, celui-ci émet à nouveau le même paquet ;
- l'utilisation d'un drapeau (*flag* en anglais), un bit supplémentaire (0 ou 1) ajouté aux paquets et aux signaux ACK émis, informant de ce qui est envoyé par l'émetteur et de ce qui est attendu par le destinataire :
  - Le premier paquet envoyé a pour drapeau 0, le second aura pour drapeau 1, puis le troisième a pour drapeau 0, etc. il y a donc une alternance des valeurs des drapeaux (d'où le nom du protocole) ;

∗ Quand le destinataire reçoit un paquet avec le drapeau 0 (et si c'est ce qu'il attend), il émet un ACK avec le drapeau 1 pour indiquer qu'il a bien reçu le paquet et attend le paquet suivant, et vice-versa. Le destinataire vérifie à chaque réception d'un paquet que celui ci a bien le drapeau attendu, sans quoi il rejette le paquet reçu et réémet un ACK avec le drapeau attendu.

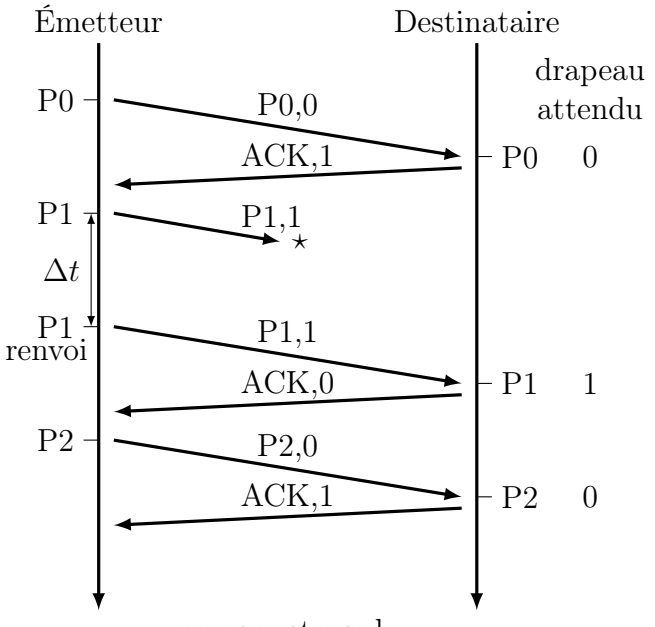
Voici le schéma d'une transmission parfaite, avec les drapeaux indiqués le long des flèches :



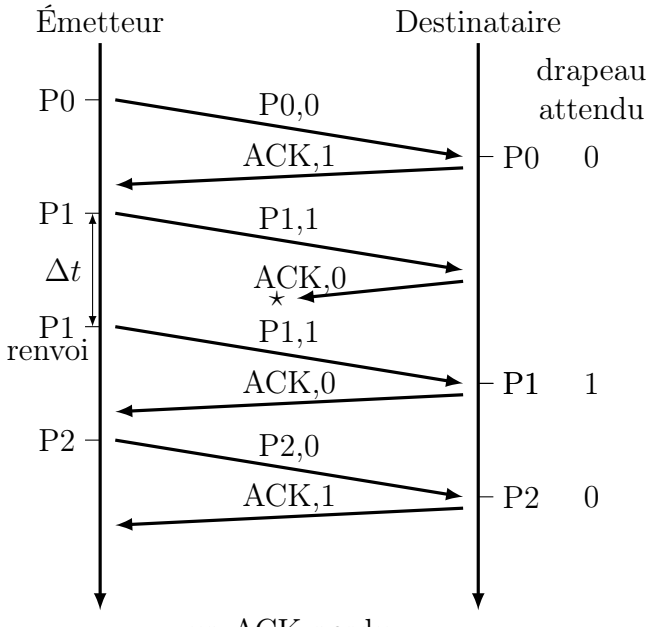
transmission avec le protocole du bit alterné

Remarque Il est possible de choisir, dans le protocole, que le drapeau de l'ACK soit de même valeur que celui du paquet reçu. Autrement dit, on peut choisir qu'à la réception du drapeau 0 (attendu), le destinateur émette un ACK avec le même drapeau 0 (et non 1 comme décrit plus haut).

Montrons deux situations problématiques possibles où le protocole permet tout de même la bonne réception du message :

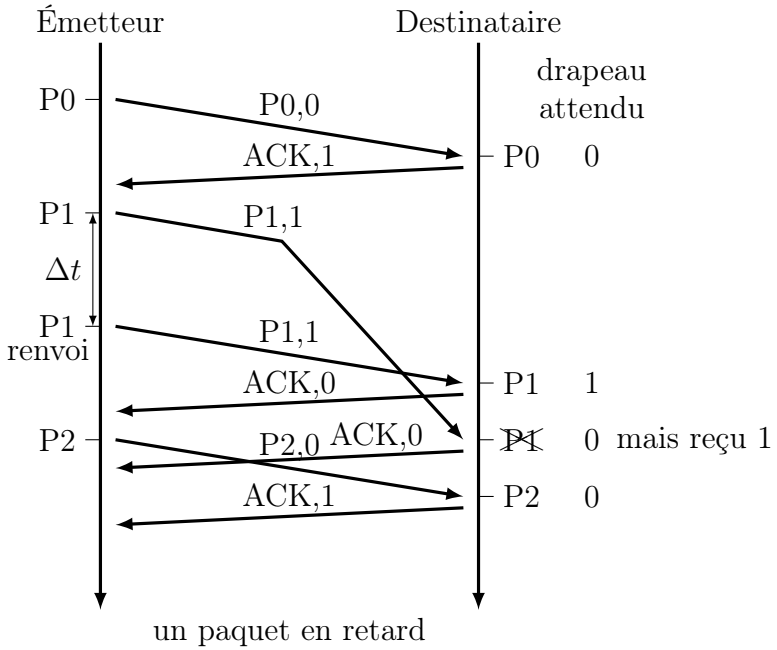


un paquet perdu



un ACK perdu

Plus compliqué, voici ce qui se produit lorsqu'un paquet n'est pas perdu mais reçu tardivement.
Le paquet barré dans la ligne du destinataire est rejeté par celui-ci :



Remarque Tout se passe bien encore ici, donc, mais c'est une autre histoire si le premier paquet P1 arrive finalement après le paquet P2 !

Ce protocole du bit alterné est donc un exemple simple de fiabilisation du transfert de données.
Il a longtemps été utilisé au sein de la couche 2 du modèle OSI (distribution des trames Ethernet).
Simple et léger, il peut toutefois être facilement mis en défaut, ce qui explique qu'il ait été remplacé par des protocoles plus performants.

III. Systèmes d'exploitations

Comme nous avons pu le voir, programmer en langage machine (ou même en langage assembleur) n'est pas facile, bien que ce soit le langage que comprend la machine. On voit donc la nécessité d'avoir un intermédiaire entre l'humain et la machine qui gère les ressources de la machine. C'est de là que naît le système d'exploitation.
Un **système d'exploitation** (SE en français, OS en anglais pour Operating System) a pour tâches de :

- Fournir une interface entre l'humain et la machine ;
- Gérer les ressources de l'ordinateur (mémoire, processeur, périphériques, etc.) ;
- Gérer les utilisateurs, en particulier leurs droits d'accès et d'exécution des fichiers ;

Un système d'exploitation rend concret ce qui ne l'est pas : un fichier est par essence abstrait, c'est seulement une suite de 0 et de 1 placés quelque part dans la mémoire de la machine, mais le système d'exploitation nous permet d'en avoir une notion concrète, par une adresse.
Il existe beaucoup de systèmes d'exploitation, tels Windows 10, Linux, mac OS X.
Chaque système d'exploitation possède un **terminal**, qui est un programme permettant d'exécuter des tâches à l'aide de commandes données au clavier.
Il y a des commandes qui sont généralement communes aux systèmes d'exploitation comme :

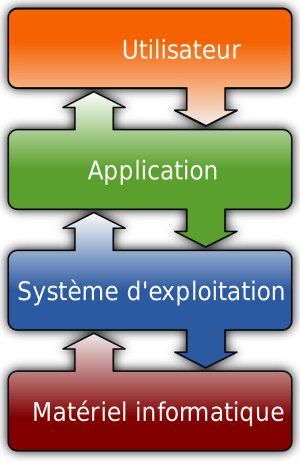
- `ls` qui liste les fichiers et répertoires du répertoire courant ;
- `cd` qui change de répertoire (Change Directory) ;
- `rm` qui supprime un fichier ou répertoire (ReMove).

D'autres commandes ne sont pas communes à tous les systèmes d'exploitation.
Nous utiliserons ici un système d'exploitation libre, à savoir Linux, dont il existe de nombreuses distributions, et qui est basé sur le système UNIX (sur lequel est également basé mac OS X).

⊛ **Activité :** Console Linux

1. Rôle du système d'exploitation

Un système d'exploitation est un logiciel qui fait le lien entre les ressources matérielles d'un ordinateur (processeur, disque dur...) et les logiciels de l'utilisateur.
Le système d'exploitation permet aux logiciels d'application de fonctionner indépendamment du matériel : il offre ce qu'on appelle une **couche d'abstraction** : plutôt que se s'adresser directement aux différents périphériques, les logiciels passent par le système d'exploitation, qui traduit leurs appels auprès du matériel.



Les logiciels d'application n'ont pas besoin de savoir comment fonctionne le matériel de l'ordinateur. Leur seul interlocuteur est le système d'exploitation qui les accueille. En revanche, le SE doit être capable de dialoguer avec les différents périphériques de l'ordinateur, d'où la nécessité de pilotes (drivers) pour interagir avec le matériel.
Un système d'exploitation a de nombreuses responsabilités :

- Permettre l'exécution et le contrôle des logiciels applicatifs ;
- Gérer les périphériques matériels (processeur, mémoire vive, carte graphique...) ;
- Fournir une Interface Homme-Machine (IHM) à un ou plusieurs utilisateurs.

2. Fonctionnalités d'un système d'exploitation

a. Gestion des applications

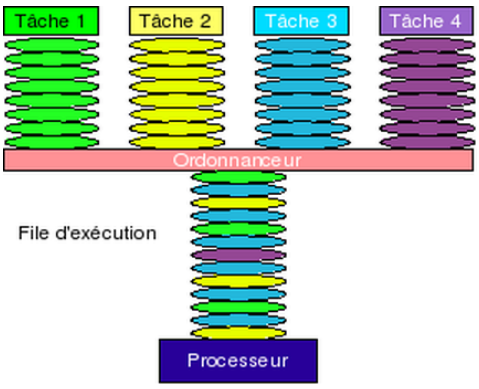
Systèmes monotâche
Comme son nom l'indique, un système **monotâche** ne peut gérer qu'une seule tâche (un seul programme) à la fois. Un programme lancé par un système mono-tâche monopolise les ressources de la machine jusqu'à sa fin d'exécution.
Ses deux principaux inconvénients sont :

- Elle ne permet pas d'utiliser plusieurs programmes simultanément ;
- En cas de blocage du programme, tout le système est arrêté : il faut redémarrer l'ordinateur...

Les systèmes mono-tâches ont quasiment disparu du paysage des SE.
Systèmes multitâche
Un système multitâche permet l'exécution simultanée de plusieurs programmes sur une machine. Cette simultanéité n'est qu'apparente : en réalité, chaque programme est exécuté tour à tour à un rythme très rapide, ce qui donne l'illusion d'une exécution en parallèle.
Il existe deux variantes possibles du multitâche. En multitâche **coopératif**, chaque application décide elle-même du moment où elle "rend la main" pour permettre aux autres de s'exécuter. Le système est dépendant des applications qu'il accueille. Un programme mal conçu peut monopoliser les ressources ou bloquer le système en cas de crash.
Ce type de multitâche, présent sur les SE des années 1990, a été remplacé par le multitâche **préemptif**. Ici, le système d'exploitation distribue les temps d'exécution entre les différents programmes, à la manière d'un chef d'orchestre. Les ressources sont mieux partagées entre les applications, et une erreur dans un programme ne menace plus l'ensemble du système.

b. Gestion du processeur

Le processeur est l'unité centrale d'exécution d'un ordinateur. A chaque top de son horloge interne, soit plusieurs millions de fois par seconde, il exécute une instruction (ou une partie d'instruction). A un instant donné, un processeur ne peut donc effectuer qu'un traitement à la fois.
Pour obtenir un multitâche préemptif, le SE dispose d'un ordonnanceur qui gère l'ordre dans lequel les instructions des différents programmes sont exécutées. Des priorités peuvent être affectées à chaque tâche en fonction de leur importance.



c. Gestion de la mémoire

Pour fonctionner, les programmes occupent et utilisent de la place en mémoire. Le système d'exploitation gère l'allocation de cette mémoire aux applications. Quand la place en mémoire vive (RAM, Random Access Memory) vient à manquer, une partie du disque dur est utilisée comme mémoire virtuelle (plus lente que la RAM).
Un système d'exploitation consomme aussi de la RAM pour son propre usage. Les systèmes récents ont tendance à être de plus en plus "gourmands" en RAM. A titre d'exemple, Windows 10 nécessite 4 Go de RAM pour un usage confortable.

d. Gestion des partitions

Un système d'exploitation doit piloter les différents périphériques de stockage présents sur l'ordinateur pour pouvoir y lire et écrire des données de manière persistante. Un périphérique de stockage (disque dur, SSD, clé USB...) est divisé en zones spécifiques appelées **partitions**. Une partition est une unité de stockage logique, qui peut être **formatée** pour stocker des informations telles que des fichiers de données ou des applications.
Pendant la phase d'installation, la plupart des systèmes d'exploitation partitionnent et formatent automatiquement l'espace disque.

e. Stockage des fichiers

La gestion des données persistantes sur les périphériques de stockage est assurée par le système de gestion de fichiers (SGF) du système d'exploitation. Les fichiers sont découpés sous forme de blocs et répartis sur le support de stockage.
Un SGF offre les fonctionnalités suivantes :

- Manipulation des fichiers et des répertoires ;
- Localisation des fichiers ;
- Sécurité et contrôle des fichiers.

Il existe de nombreux SGF offrant des niveaux de fonctionnalités variables, comme FAT32 (File Allocation Table), NTFS (New Technology File System), HFS (Hierarchical File System) ou ext4 (EXTended File System version 4). Le SGF est choisi lors du formatage d'une partition.
Le système d'exploitation propose également des mécanismes de gestion des fichiers et des répertoires : création, copie, suppression, formatage...

f. Interface Homme-Machine

L'IHM d'un système d'exploitation permet à l'utilisateur d'interagir avec le système : lancement de commandes, utilisation de logiciels, accès aux fichiers...
Jusqu'au milieu des années 1980, les systèmes disposaient uniquement d'une IHM textuelle.

De nos jours, la plupart des systèmes offrent une IHM graphique conviviale, ou GUI (Graphical User Interface). Il reste possible de les utiliser en mode texte grâce à un terminal. Elle présente à l'utilisateur un bureau virtuel et permet le pilotage du système à la souris.

Cette vidéo illustre l'évolution de l'IHM des OS de la famille Windows.

g. Gestion des utilisateurs

On peut classer les systèmes d'exploitation en deux catégories, selon leurs possibilités de gestion des utilisateurs.

Systèmes mono-utilisateur

Un système mono-utilisateur ne peut gérer qu'un seul utilisateur à la fois. Les systèmes d'exploitation des ordinateurs personnels (Windows, Mac OS), ont longtemps été mono-utilisateur.

Systèmes multi-utilisateurs

Un tel système permet à plusieurs utilisateurs d'exploiter simultanément les ressources de la machine, ce qui introduit de nouvelles problématiques :

- L'environnement propre à chaque utilisateur (identification, ressources propres) ;
- La sécurité de l'accès partagé aux données et aux programmes ;
- Les droits de chaque utilisateur (accès aux fichiers, exécution de programmes).

Pour des raisons de sécurité, tous les utilisateurs n'ont pas les mêmes droits sur le système. Seul un utilisateur particulier (`Administrateur` sous Windows, `root` sous Linux) possède tous les droits. La majorité des systèmes d'exploitation récents sont multi-utilisateurs.

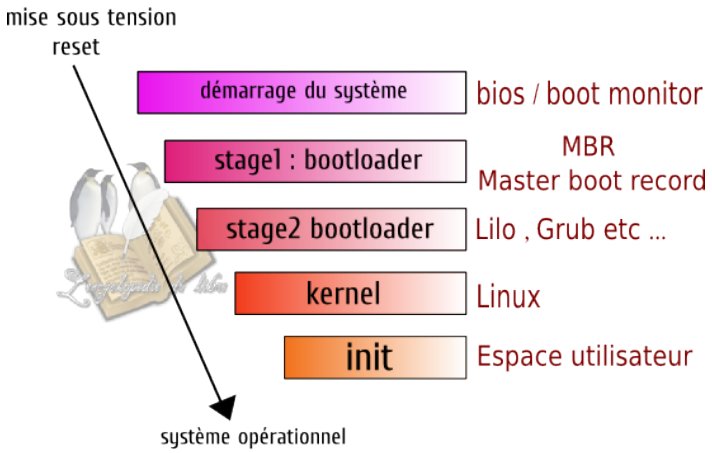
3. Architecture d'un système d'exploitation

Un système d'exploitation est typiquement composé :

- D'un noyau ;
- De bibliothèques ;
- D'un ensemble d'outils système ;
- De plusieurs applications de base.

a. Le noyau

Le noyau (ou kernel) est la partie fondamentale du système d'exploitation. Il est chargé en mémoire vive durant le démarrage de l'ordinateur (boot sequence).



Après le démarrage, la mémoire est divisée en deux parties :

- L'espace noyau, réservé au noyau lui-même.
- L'espace utilisateur, dédié aux applications.

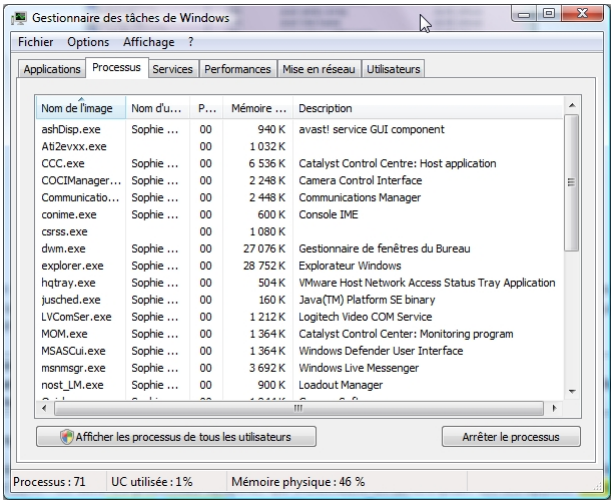
Cette division permet d'augmenter la robustesse du système : aucun programme de l'espace utilisateur ne peut accéder à la mémoire du noyau.

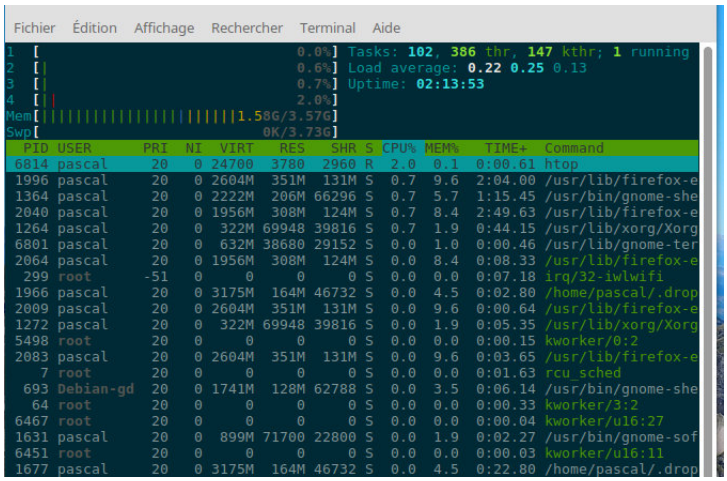
Le noyau remplit les fonctions essentielles du système d'exploitation :

- Gestion du matériel ;
- Exécution des programmes (sous la forme de processus) ;
- Échanges entre matériel et logiciels.

b. Notion de processus

Un **processus** est l'entité qui représente l'exécution d'un programme sur un ordinateur. Son état évolue au cours du temps : il a un début, un déroulement et une fin.
Le noyau du système d'exploitation offre des fonctions de contrôle des processus en cours d'exécution (exemple : gestionnaire des tâches sous Windows).





Un logiciel lancé par l'utilisateur est manipulé par le noyau sous la forme d'un ou plusieurs processus. Chaque processus se voit attribuer des droits (souvent ceux de l'utilisateur).

Processus système

Il existe des processus lancés par le système d'exploitation lui-même pour assurer des fonctions de base : ce sont les processus système, qui s'exécutent en arrière-plan. On les appelle **services** sous Windows et **démons** (daemons) sous UNIX et Linux.

Processus légers

Certains programmes s'exécutent sous la forme d'un ensemble de processus légers (ou threads) appartenant à un même processus père. Cela permet de faciliter la communication entre les différents threads d'un processus et d'améliorer les performances de certaines opérations (traitements en parallèle).

c. Les bibliothèques

Une bibliothèque (ou librairie) regroupe un ensemble de fonctions souvent utilisées par les programmes. Elle peut contenir des fonctions simples (exemples : sinus, cosinus) ou très complexes (exemples : rendu graphique 3D). Elles sont partagées par les logiciels qui s'exécutent et leur offrent des API (Application Programming Interface).
Une bibliothèque évite à chaque programme de devoir réécrire manuellement les fonctions qu'elle propose.
La plupart des systèmes d'exploitation récents supportent les bibliothèques dynamiques, qui sont chargées par les programmes sans en faire directement partie. Cela permet de diminuer la taille des fichiers exécutables et la place occupée en mémoire. Elles sont identifiées par l'extension .dll sous Windows et .so sous Linux.

d. 32 bits, 64 bits

Un système d'exploitation peut être disponible en versions 32 bits et 64 bits afin de s'adapter au processeur de l'ordinateur. Ce nombre de bits correspond à la taille des registres mémoire du processeur. L'augmentation de la largeur des registres permet, entre autres, d'adresser une quantité plus important de mémoire vive. La quantité de RAM utilisable est limitée à 4 Go avec un processeur 32 bits. Un processeur 64 bits peut faire tourner un OS 32 bits, mais l'inverse n'est pas vrai.

Les systèmes d'exploitation de la famille Windows existent en version 64 bits depuis Windows XP.

4. Le marché des systèmes d'exploitation

Il existe plusieurs familles de systèmes d'exploitation, adaptées à différents types de machines.

a. Les serveurs

Un serveur est une machine dont le rôle est d'héberger des données ou des programmes afin de répondre aux demandes de ses clients.
Exemples de systèmes d'exploitation pour serveurs : Windows 2008 Server, Solaris, FreeBSD, Linux Debian.

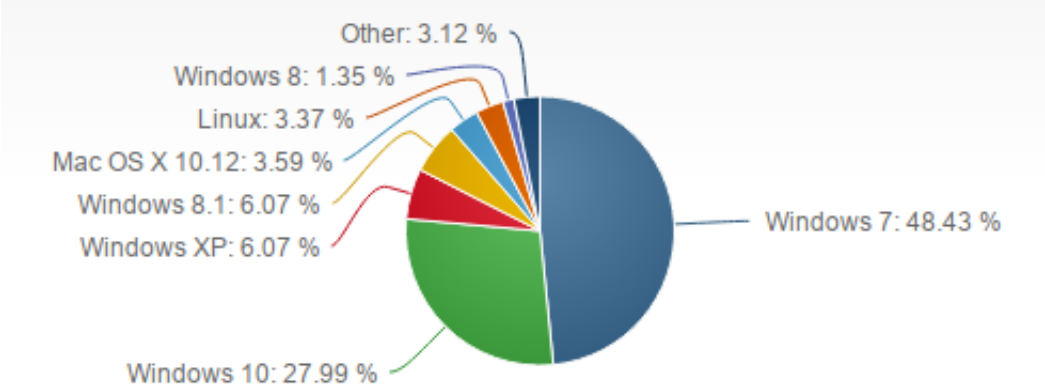
b. Les postes de travail

Un poste de travail est destiné à un seul utilisateur, pour un usage professionnel ou personnel. La grande majorité des postes de travail utilise un système d'exploitation de la famille Windows.
Exemples de systèmes d'exploitation pour postes de travail : Windows 7, Windows 10, Mac OS X, Linux Ubuntu.

c. Les terminaux mobiles

Ces dernières années ont vu apparaître un grand nombre d'appareils mobiles de plus en plus sophistiqués : téléphones portables, tablettes... Ces périphériques embarquent des systèmes d'exploitation spécifiques ou des versions modifiées de systèmes existants.
Exemples de systèmes d'exploitation pour appareils mobiles : iOS, Android, Windows Phone.

d. Quelques statistiques pour les OS des ordinateurs de bureaux



(données de netmarketshare.com mars 2017)

Il existe des systèmes d'exploitation pour tous les appareils électroniques, pèse-personnes, montres connectées, etc...

e. Un exemple : GNU/Linux

GNU est un projet qui a apporté des tas d'utilitaires au noyau Linux, tel que le fameux compilateur gcc, et les milliers d'utilitaires (tar, tail, man, bash...). Ces utilitaires GNU, associés au noyau Linux, constituent le système d'exploitation GNU/Linux.
Linux est donc un noyau. GNU est un ensemble de programmes utilitaires. GNU/Linux est le système d'exploitation.
(https://fr.wikipedia.org/wiki/Linux)

(Tux, mascotte de Linux)

Les grands noms associés à Linux sont :

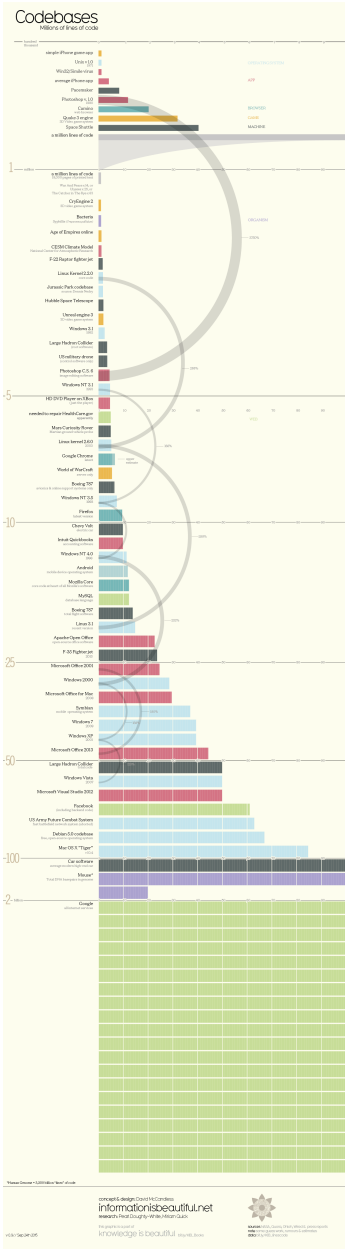
Linus Torwals

Richard Stallman

Il existe de nombreuses distributions de Linux, parmi lesquelles :

Mint, Suse, Fedora
Mageia, Ubuntu, Debian

Évolution du nombre de lignes de codes pour plusieurs systèmes et programmes :



IV. Entrées et sorties

Les dispositifs d'entrée/sortie sur un ordinateur sont constitués de périphériques connectés par câble sur des ports ou via une technologie sans fil (bluetooth, wifi, etc.). Les exemples les plus fréquents pour les ordinateurs sont les claviers, moniteurs, souris, imprimantes, etc.
L'utilisateur de ces périphériques envoi un signal à l'ordinateur (en entrée) qui réagit en fonction d'un algorithme, selon le principe de gestion des événements comme nous avons pu le voir par exemple avec JavaScript.
Pour faire cela, l'ordinateur fait en permanence fonctionner un programme qui boucle tant que l'ordinateur est allumé, et qui est à l'écoute des événements en provenance de ses périphériques, et qui réagit en fonction des données reçues et des programmes qui écoutent ces événements.
Dans d'autres cas, l'exécution d'un algorithme produit en sortie des données, par exemple sur un moniteur ou une imprimante.
Certains périphériques portent plutôt le nom de capteurs ou d'actionneurs. Ils équipent les systèmes embarqués comme les robots et les objets connectés.
Il s'agit par exemple de capteurs de son, de lumière, de température, etc. et les actionneurs sont par exemple des moteurs ou des diodes.
Autrement dit, un capteur est un dispositif d'acquisition de données (physiques), et un actionneur est un dispositif permettant au système d'agir.
Ces éléments sont mis en œuvre dans les robots, qui sont des objets équipés de systèmes embarqués, éventuellement connectés par wifi à des ordinateurs qui peuvent les programmer.
Généralement, chaque système embarqué a son propre mode de programmation, mais le principe est toujours le même.
Il y a des fonctions qui interrogent les capteurs et d'autres qui commandent les actionneurs. Des algorithmes permettent alors d'établir le comportement souhaité en fonction des données captées.

V. Interface homme machine

Une interface homme machine (IHM) est une interface utilisateur permettant à une personne de dialoguer avec un système. Il peut s'agir d'un écran qui affiche des informations lors du fonctionnement de la machine. Une interface graphique donne des moyens à l'utilisateur d'agir (en cliquant sur des boutons, en tapant du texte), mais aussi à la machine d'informer de l'état du fonctionnement.
Les premières interfaces homme machine se trouvent sous la forme des télétypes (ou téléscripteurs), qui sont essentiellement un système d'un clavier pour les entrées et une imprimante pour la sortie (il n'y avait pas d'écran). L'influence de ces systèmes est telle que sous Linux, les périphériques de type Console (ou Terminal) sont nommés, tty en référence à ces interfaces.
Les premiers environnements graphiques, avec écran, datent de 1968, où Douglas Engelbart, à Stanford, présenta un pointeur permettant d'ouvrir ou de fermer des fenêtres. C'est l'invention, autrement dit, de la souris.
À partir de 1983 naissent les premières interfaces utilisateur graphiques (GUI pour Graphical User Interface en anglais). Elles rendent l'utilisation plus simple des ordinateurs, permettant ainsi leur démocratisation.
Depuis quelques années se développent des interfaces naturelles (NUI pour Natural User Interface) : on utilise le toucher, la voix, le mouvement, etc. pour commander les machines.
Lorsque l'on doit réaliser une IHM, il faut commencer par établir un cahier des charges. Autrement dit on détaille les spécificités du système, les besoins en capteurs et en actionneurs pour réaliser les

tâches souhaitées, les conditions nécessaires au bon fonctionnement, les moyens d'interagir avec la système, et la description des tâches que la machine doit effectuer.
Dans certains cas, on peut tester des systèmes, comme par exemple des robots, par émulateurs. Autrement dit on teste le fonctionnement dans un environnement numérique qui simule une réalité physique dans lequel on place une représentation du robot que l'on laisse agir comme il le ferait en vrai, selon les instructions codées par le programmeur.