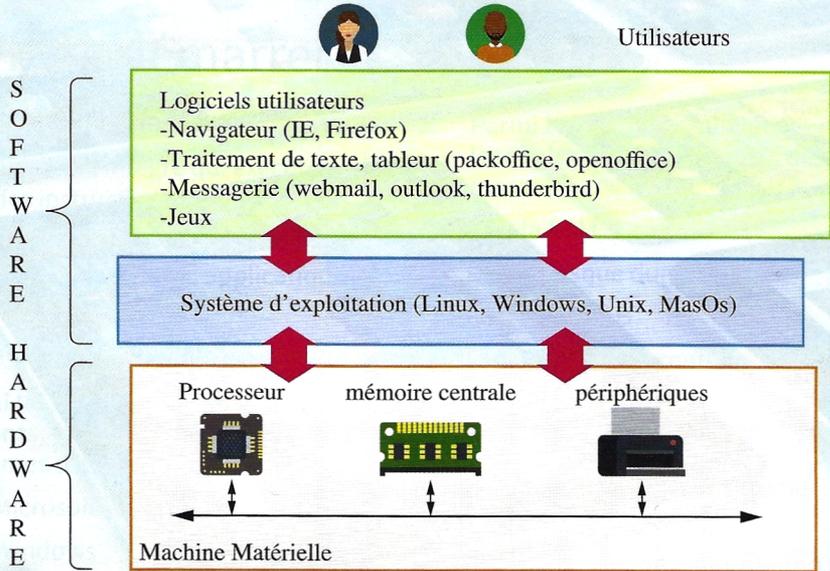


1 Rôle et fonctions des systèmes d'exploitation

Définition

Un **système d'exploitation** est un ensemble de programmes qui réalise l'interface entre le matériel de l'ordinateur et les applications des utilisateurs. Il a deux objectifs principaux :

- la construction au-dessus du matériel d'une machine virtuelle plus facile d'emploi et plus conviviale ;
- la prise en charge de la gestion de plus en plus complexe des ressources et de leur partage.



Comme son nom le suggère, le système d'exploitation a en charge l'exploitation de la machine pour en faciliter l'accès, le partage et pour l'optimiser.

Les principaux systèmes d'exploitation sont : Windows, Linux, Unix, Mac OS, Android.



La société Microsoft a été fondée en 1975 par deux étudiants, Bill Gates et Paul Allen. Cette société est surtout connue pour ses logiciels, dont le système d'exploitation Windows.

1.1 Rôles du système d'exploitation

1.1.1 Assurer le partage de la machine physique

La machine physique et ses différents composants, s'ils offrent des mécanismes permettant de faciliter leur partage entre différents programmes, ne sont pas conçus pour supporter et gérer d'eux-mêmes ce partage. C'est là le premier rôle du système d'exploitation dans un environnement multiprogrammé que de gérer le partage de la machine physique et des ressources matérielles entre les différents programmes.

Ce partage des ressources va concerner principalement le processeur, la mémoire centrale et les périphériques d'entrées-sorties. De manière concrète, les questions suivantes vont devoir être résolues :

- dans le cadre du partage du processeur : parmi tous les programmes chargés en mémoire centrale, lequel doit s'exécuter ?

- dans le cadre du partage de la mémoire centrale : comment allouer la mémoire centrale aux différents programmes ? Comment disposer d'une quantité suffisante de mémoire pour y placer tous les programmes nécessaires en optimisant le taux d'utilisation du processeur ? Comment assurer la protection¹ entre différents programmes utilisateurs ?
- dans le cadre du partage des périphériques : dans quel ordre traiter les requêtes d'entrées-sorties pour optimiser les transferts ?

1.1.2 Rendre conviviale la machine physique

Faciliter l'accès à la machine physique constitue le second rôle du système d'exploitation. Par le biais d'une interface de haut niveau, composée d'un ensemble de primitives attachées à des fonctionnalités qui gèrent elles-mêmes les caractéristiques matérielles sous-jacentes et offrent un service à l'utilisateur, le système d'exploitation construit au-dessus de la machine physique, une machine virtuelle plus simple d'emploi et plus conviviale. Ainsi, pour réaliser une opération d'entrées-sorties, l'utilisateur fera appel à une même primitive `ECRIRE` (données) quel que soit le périphérique concerné. C'est la primitive `ECRIRE` et la fonction de gestion des entrées-sorties du système d'exploitation à laquelle cette primitive est rattachée qui feront la liaison avec les caractéristiques matérielles.

1.2 Fonctions du système d'exploitation

Le système d'exploitation réalise donc une couche logicielle placée entre la machine matérielle et les applications. Le système d'exploitation peut être découpé en plusieurs grandes fonctionnalités :

- La fonctionnalité de **gestion du processeur** : le système doit gérer l'allocation du processeur aux différents programmes pouvant s'exécuter. Cette allocation se fait par le biais d'un **algorithme d'ordonnancement** qui planifie l'exécution des programmes. Selon le type de système d'exploitation, l'algorithme d'ordonnancement répond à des objectifs différents (par exemple, offrir le plus court temps d'exécution).
- La fonctionnalité de **gestion de la mémoire** : le système doit gérer l'allocation de la mémoire centrale entre les différents programmes pouvant s'exécuter, c'est-à-dire qu'il doit trouver une place libre suffisante en mémoire centrale pour que le chargeur puisse y placer un programme à exécuter. Les principales méthodes d'allocation actuelles sont la **segmentation** et la **pagination**.
- La fonctionnalité de **gestion des entrées-sorties** : le système doit gérer l'accès aux périphériques, c'est-à-dire faire la liaison entre les appels de haut niveau des programmes utilisateurs (exemple `getchar()`) et les opérations de bas niveau de l'unité d'échange responsable du périphérique (unité d'échange clavier) : c'est le pilote d'entrées-sorties qui assure cette correspondance.

1. Par protection, on entend ici veiller à ce qu'un programme donné n'accède pas à une plage mémoire allouée à un autre programme.



■ *Apple est une société américaine fondée en 1975 par Steve Jobs. Elle est connue pour ses ordinateurs Macintosh, ses tablettes Ipad, ses téléphones iPhone et son système d'exploitation Mac OS.*

- La fonctionnalité de **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 non volatile (disque dur, clé USB, CD-rom...). La gestion de l'allocation des mémoires de masse ainsi que l'accès aux données stockées s'appuient sur la notion de **fichiers** et le **système de gestion de fichiers** (SGF).
- La fonctionnalité de **gestion de la concurrence** : comme plusieurs programmes coexistent en mémoire centrale, ceux-ci peuvent avoir à communiquer pour échanger des données. Par ailleurs, il faut synchroniser l'accès aux données partagées afin de maintenir leur cohérence. Le système offre des outils de communication et de synchronisation entre programmes.
- La fonctionnalité de **gestion de la protection** : le système doit fournir des mécanismes garantissant que ses ressources (processeur, mémoire, fichiers) ne peuvent être utilisées que par les programmes auxquels les droits nécessaires ont été accordés. Il faut notamment protéger le système et la machine des programmes utilisateurs (mode d'exécution utilisateur et superviseur).

1.3 Les différents types de systèmes d'exploitation

Les systèmes d'exploitation multiprogrammés peuvent être classés selon différents types qui dépendent des buts et des services qu'ils offrent. Trois grandes classes peuvent être définies : les systèmes à **traitements par lots**, les systèmes **multi-utilisateurs interactifs** et les systèmes **temps réels**.

1.3.1 Les systèmes à traitements par lots

Les systèmes à traitement par lots ou systèmes *batch* constituent en quelque sorte les ancêtres de tous les systèmes d'exploitation.

Le principe du traitement par lots s'appuie sur la composition de lots de travaux ayant des caractéristiques ou des besoins communs, la formation de ces lots visant à réduire les temps d'attente du processeur en faisant exécuter les uns à la suite des autres ou ensemble, des travaux nécessitant les mêmes ressources. Par ailleurs, dans un système à traitements par lots, la caractéristique principale est qu'il n'y a pas d'interaction possible entre l'utilisateur et la machine durant l'exécution du programme soumis. Le programme est enregistré avec les données qu'il doit traiter et l'utilisateur récupère les résultats de l'exécution ultérieurement, soit dans un fichier, soit sous forme d'une impression.

L'objectif poursuivi dans les systèmes à traitements par lots est de maximiser l'utilisation du processeur et le débit des travaux, c'est-à-dire le nombre de travaux traités sur une tranche de temps.

Un exemple de système d'exploitation orienté *batch* est le système MVS.

1.3.2 Les systèmes interactifs

La particularité d'un système d'exploitation interactif est que, contrairement aux systèmes précédents, l'utilisateur de la machine peut interagir avec l'exécution de son programme. Typiquement, l'utilisateur lance l'exécution de son travail et attend le résultat derrière le clavier et l'écran. S'il s'aperçoit que l'exécution n'est pas conforme à son espérance, il peut immédiatement agir pour l'arrêter et analyser les raisons de l'échec.

Puisque l'utilisateur attend derrière son clavier et son écran et que par nature, l'utilisateur de la machine est un être impatient, le but principal poursuivi par les systèmes interactifs est de faire en sorte que chaque exécution se termine au plus tôt. Pour parvenir à ce but, la plupart des systèmes interactifs travaillent en temps partagé.

Un système en temps partagé permet aux différents utilisateurs de partager l'ordinateur simultanément, tout en ayant par ailleurs la sensation d'être seul à utiliser la machine. Ce principe repose notamment sur un partage de l'utilisation du processeur par les programmes des différents utilisateurs. Chaque programme occupe à tour de rôle le processeur pour un court laps de temps et les exécutions se succèdent suffisamment rapidement sur le processeur pour que l'utilisateur ait l'impression que son travail est le seul à disposer du processeur.

Des exemples de systèmes interactifs fonctionnant en temps partagé sont le système Unix ou le système Linux.

1.3.3 Les systèmes temps réel

Les systèmes temps réel sont des systèmes liés au contrôle de **procédé** pour lesquels la caractéristique primordiale est que les exécutions de programmes sont soumises à des contraintes temporelles, c'est-à-dire qu'une exécution de programme est notamment qualifiée par une date butoir de fin d'exécution, appelée **échéance**, au-delà de laquelle les résultats de l'exécution ne sont plus valides.

Par exemple le pilotage d'un métro automatique ou encore le dispositif de surveillance d'un réacteur de centrale nucléaire requièrent des systèmes temps réel. Le procédé désigne ici soit le métro, soit le réacteur nucléaire. Dans le cas du métro, le système temps réel sera même qualifié de **système embarqué**, puisqu'il est installé à bord du train. Dans de nombreux cas par ailleurs, le système temps réel est également qualifié de **système réactif**, car il se doit de réagir dans les temps impartis aux informations qu'il reçoit.

Il ne s'agit pas de rendre le résultat le plus vite possible, mais simplement à temps. L'échelle du temps relative à la contrainte temporelle varie d'une application à l'autre : elle peut être par exemple de l'ordre de la microseconde dans des applications de contrôle radars, mais peut être de l'ordre de l'heure pour une application de contrôle chimique. Par contre, il est souvent primordial de respecter la contrainte temporelle, sous peine de graves défaillances, pouvant mettre en péril le procédé lui-même et son environnement. Tout retard de réaction face à une situation anormale au sein du réacteur nucléaire, peut évidemment mener à une situation catastrophique ! LynxOs de la société LynuxWorks ou VxWorks la firme Wind River sont des exemple de systèmes temps réel.



■ Linux est un système d'exploitation logiciel libre créé au début des années 1990 par Linus Torvalds. La mascotte de Linux est le manchot Tux.