Les Systèmes d'Exploitation

Table des matières

Un système d'exploitation est un ensemble de programmes qui dirige l'utilisation des capacités d'un ordinateur. Le système d'exploitation est le premier programme exécuté lors de la mise en marche de l'ordinateur, après l'amorçage. Il offre une suite de services généraux qui facilitent la création de logiciels applicatifs et sert d'intermédiaire entre ces logiciels et le matériel informatique.

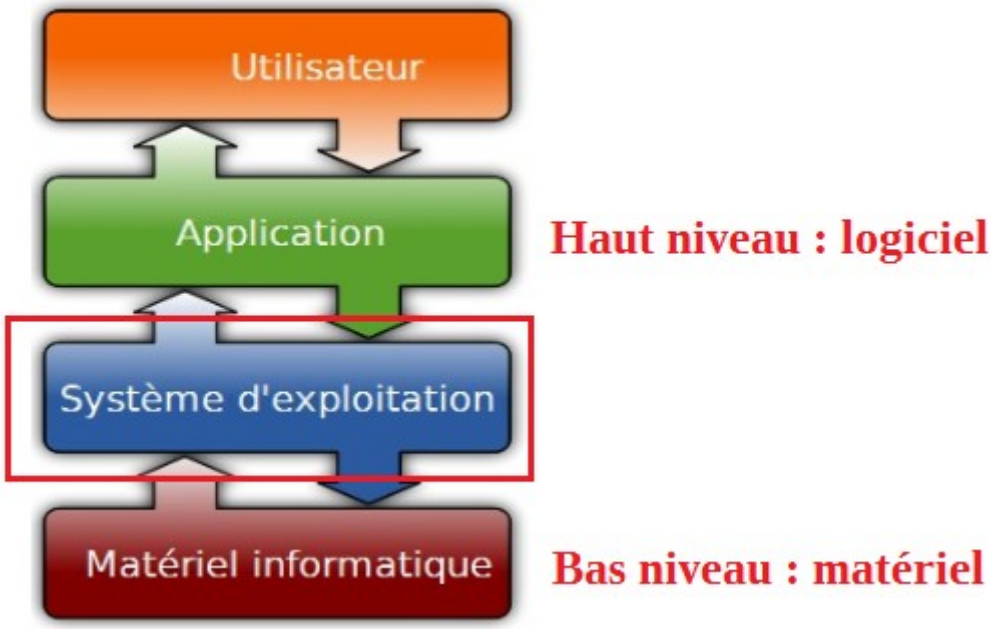
1. Introduction

Le système d'exploitation (SE) est un ensemble de programmes fondamentaux sur un appareil informatique qui sert d'interface entre le matériel et les logiciels applications. Il est souvent désigné par l'abrégé OS pour operating system en anglais.

Le système d'exploitation va gérer les disques durs, les périphériques, la mémoire, l'affichage, etc.. et permettre à l'utilisateur de lancer des programmes (messagerie, traitement de texte, …).

Le SE soustrait le matériel au regard du programmeur et offre une présentation agréable des fichiers. Un SE a ainsi deux objectifs principaux :

- présentation : Il propose à l'utilisateur une abstraction plus simple et plus agréable que le matériel : une machine virtuelle
- gestion : il ordonne et contrôle l'allocation des processeurs, des mémoires, des icônes et fenêtres, des périphériques, des réseaux entre les programmes qui les utilisent. Il assiste les programmes utilisateurs. Il protège les utilisateurs dans le cas d'usage partagé.



Les premiers systèmes d'exploitation ont été créés dans les années 1960.

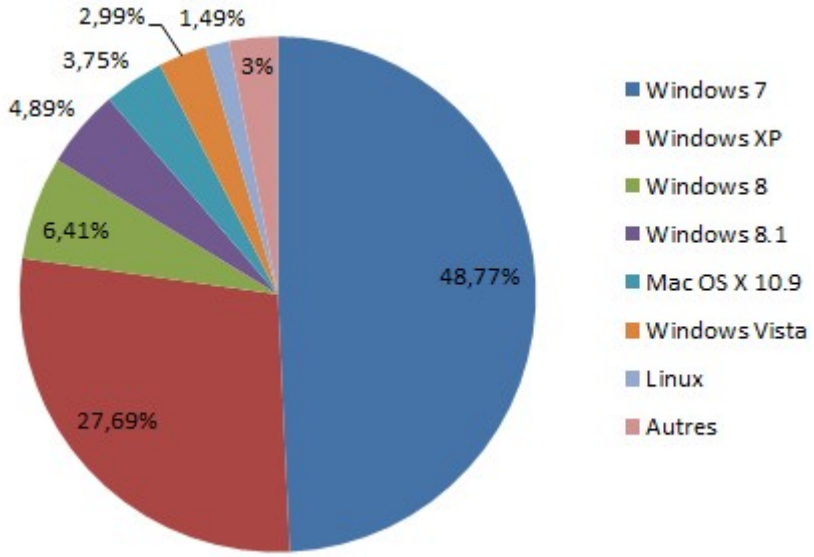
En 2010 les deux familles de systèmes d'exploitation les plus populaires sont Unix (dont Mac OS X et Linux) et Windows.

- La famille Windows détient le quasi-monopole sur les ordinateurs personnels, avec plus de 90 % de part de marché depuis 15 ans, tandis que les parts de marché des systèmes d'exploitation Unix sont de presque 50% sur les serveurs. Système d'exploitation et base matérielle.
- UNIX est le nom d'un système d'exploitation multitâche et multi- utilisateur créé en 1969.

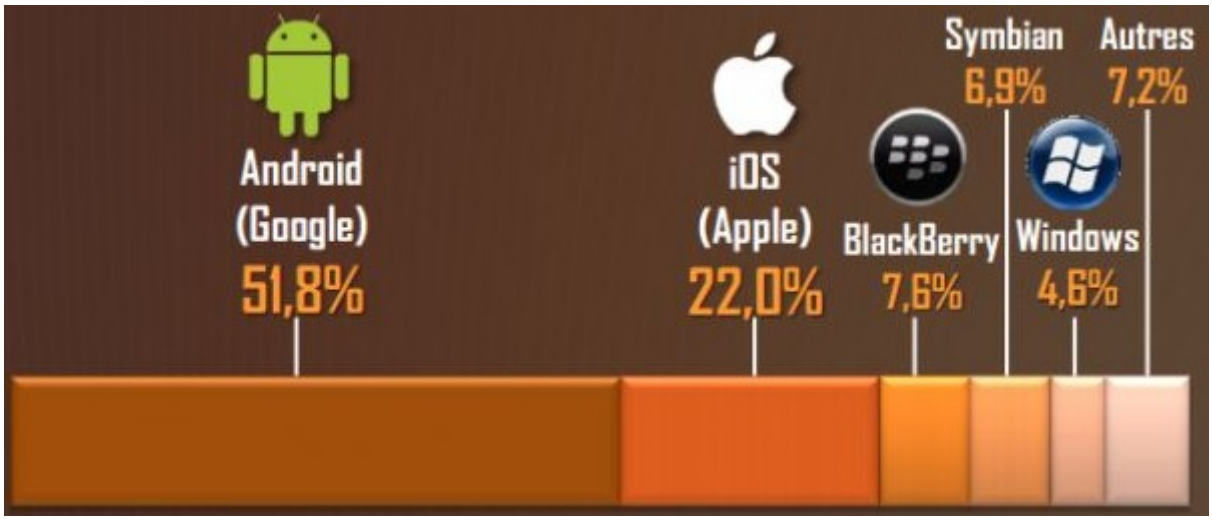
  Il a donné naissance à une famille de systèmes, dont les plus populaires à ce jour sont System V, BSD, GNU/Linux et Mac OS X .

  On nomme « famille Unix » l'ensemble de ces systèmes. On dit encore qu'ils sont de « type

Unix » ou « Unix like ».



Part de marché des systèmes d'exploitation, dans le monde et sur desktop, pour mars 2014



Part de marché des systèmes d'exploitation, en France et sur smartphone, pour mars 2013

Il existe plusieurs grandes familles de processeurs utilisés pour la fabrication des ordinateurs :

- x86 (Intel) pour les PC
- PowerPC et Motorola 68000 pour les Mac et autres Apple
- ARM pour les smartphones

Chaque OS supporte généralement mieux une famille de processeurs donnée :

- x86 : Windows, Gnu/Linux
- Mac OsX pour les Apple
- Android pour ARM

Un des enjeux actuels est de porter un OS vers d'autres familles, ce qui se fait avec plus ou moins de facilité...

2. Historique

Les premiers ordinateurs étaient mis à la disposition d'un programmeur selon un calendrier de réservation : un usager avec un travail unique utilisait seul la machine à un moment donné. Puis vint l'époque du traitement par lots (batch) : enchaînement, sous le contrôle d'un moniteur, d'une suite de travaux avec leurs données, confiés à l'équipe d'exploitation de la machine (inconvénient : temps d'attente des résultats pour chaque utilisateur).

Cette pratique a nécessité trois innovations :

- le contrôle des E/S et leur protection pour éviter le blocage d'un lot.
- un mécanisme de comptage de temps et de déroutement autoritaire des programmes pour éviter le blocage d'un lot à cause d'une séquence trop longue. Ce furent les première interruptions.
- les premiers langages de commande (JCL[1]).

Historiquement, on peut dire que les SE sont vraiment nés avec les ordinateurs de la 3ème génération (ordinateurs à circuits intégrés apparus après 1965). Le premier SE digne de ce nom est l'OS/360, celui des IBM 360, famille unique de machines compatibles entre elles, de puissances et de configurations différentes. Bien que son extrême complexité (due à l'erreur de couvrir toute la gamme 360) n'ait jamais permis d'en réduire le nombre de bogues, il apportait deux concepts nouveaux :

- la multiprogrammation : partitionnement de la mémoire permettant au processeur d'accueillir une tâche dans chaque partie et donc d'être utilisé plus efficacement par rapport aux temps d'attente introduits par les périphériques (le processeur est ré-alloué)
- les E/S tamponnées : adjonction à l'UC[2] d'un processeur autonome capable de gérer en parallèle les E/S ou canal ou unité d'échange. Cela nécessite une politique de partage du bus ou d'autres mécanismes (vol de cycle, DMA[3]).

Au MIT[4], F.J. CORBATO et son équipe ont réalisé dès 1962, sur un IBM 7094 modifié, le premier SE expérimental à temps partagé (mode interactif entre plusieurs utilisateurs simultanés), baptisé CTSS. Une version commerciale, nommée MULTICS[5], a été ensuite étudiée par cette équipe, les Bell Laboratories et General Electric. Les difficultés ont fait que MULTICS n'a jamais dépassé le stade expérimental sur une douzaine de sites entre 1965 et 1974. Mais il a permis de définir des concepts théoriques importants pour la suite.

La technologie à base de circuits intégrés de la 3ème génération d'ordinateurs a permis l'apparition des mini-ordinateurs et leur diffusion massive (précédant celle des microordinateurs).

En 1968, l'un des auteurs de MULTICS, Ken Thompson a effectué une adaptation de MULTICS mono-utilisateur sur un mini-ordinateur PDP-11 de DEC inutilisé dans son Laboratoire des Bell Laboratories. Son collègue Brian Kernighan la nomma UNICS (Uniplexed - à l'opposé de Multiplexed - Information and Computer Service), qui devint ensuite UNIX . En 1972, son collègue Dennis Ritchie traduisit UNIX en C, langage qu'il venait de mettre au point avec Kernighan... L'ère des grands SE avait commencé.

1 Job Control Language
2 Unité centrale
3 Direct Memory Access
4 Massachusetts Institute of Technology
5 MULTiplexed Information and Computing Service

Avec la grande diffusion des micro-ordinateurs, l'évolution des performances des réseaux de télécommunications, deux nouvelles catégories de SE sont apparus :

- les SE en réseaux : ils permettent à partir d'une machine de se connecter sur une machine distante, de transférer des données. Mais chaque machine dispose de son propre SE.
- les SE distribués ou répartis : l'utilisateur ne sait pas où sont physiquement ses données, ni où s'exécute son programme. Le SE gère l'ensemble des machines connectées. Le système informatique apparaît comme un mono-processeur.

3. Éléments de base d'un système d'exploitation

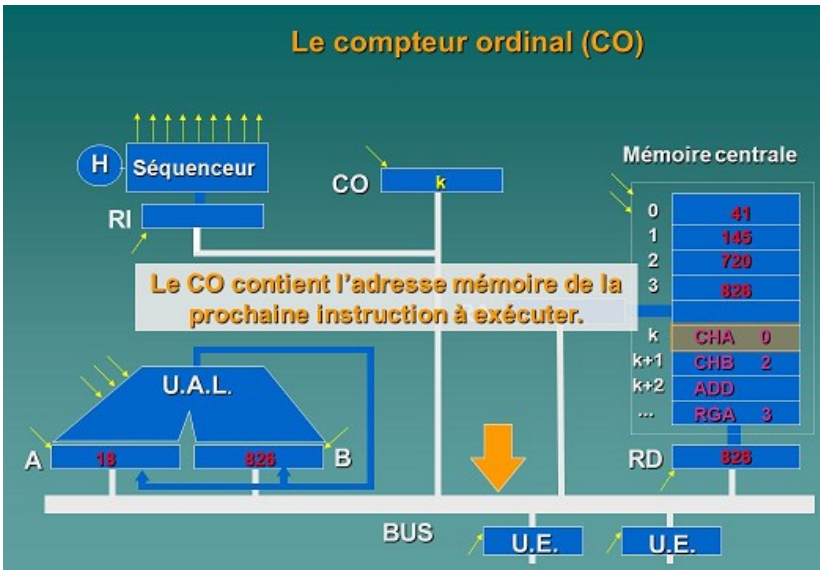
Les principales fonctions assurées par un SE sont les suivantes :

- gestion de la mémoire principale et des mémoires secondaires,
- exécution des E/S à faible débit (terminaux, imprimantes) ou haut débit (disques, bandes),
- multiprogrammation, temps partagé, parallélisme : interruption, ordonnancement, répartition en mémoire, partage des données
- lancement des outils du système (compilateurs, environnement utilisateur,...) et des outils pour l'administrateur du système (création de points d'entrée, modification de privilèges,...),
- lancement des travaux,
- protection, sécurité,
- réseaux

L'interface entre un SE et les programmes utilisateurs est constituée d'un ensemble d'instructions étendues, spécifiques d'un SE, ou appels système. Généralement, les appels système concernent soit les processus, soit le système de gestion de fichiers (SGF).

3.1. Les processus

Un processus est 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.



Attention : ne pas confondre un processus (aspect dynamique, exécution qui peut être suspendue, puis reprise), avec le texte d'un programme exécutable (aspect statique).

Les appels système relatifs aux processus permettent généralement d'effectuer au moins les actions suivantes :

- création d'un processus (fils) par un processus actif (d'où la structure d'arbre de processus gérée par un SE)
- destruction d'un processus
- mise en attente, réveil d'un processus
- suspension et reprise d'un processus, grâce à l'ordonnanceur de processus (scheduler)
- demande de mémoire supplémentaire ou restitution de mémoire inutilisée
- attendre la fin d'un processus fils
- remplacer son propre code par celui d'un programme différent
- échanges de messages avec d'autres processus
- spécification d'actions à entreprendre en fonction d'événements extérieurs asynchrones
- modifier la priorité d'un processus

Dans une entité logique unique, généralement un mot, le SE regroupe des informations-clés sur le fonctionnement du processeur : c'est le mot d'état du processeur (PSW[6]). Il comporte généralement :

- la valeur du compteur ordinal
- des informations sur les interruptions (masquées ou non)
- le privilège du processeur (mode maître ou esclave)
- etc.... (format spécifique à un processeur)

A chaque instant, un processus est caractérisé par son état courant : c'est l'ensemble des informations nécessaires à la poursuite de son exécution (valeur du compteur ordinal, contenu des différents registres, informations sur l'utilisation des ressources). A cet effet, 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...
- et toutes les informations nécessaires pour assurer la reprise du processus en cas d'interruption

6 Processor Status Word

Les BCP sont rangés dans une table en mémoire centrale à cause de leur manipulation fréquente.

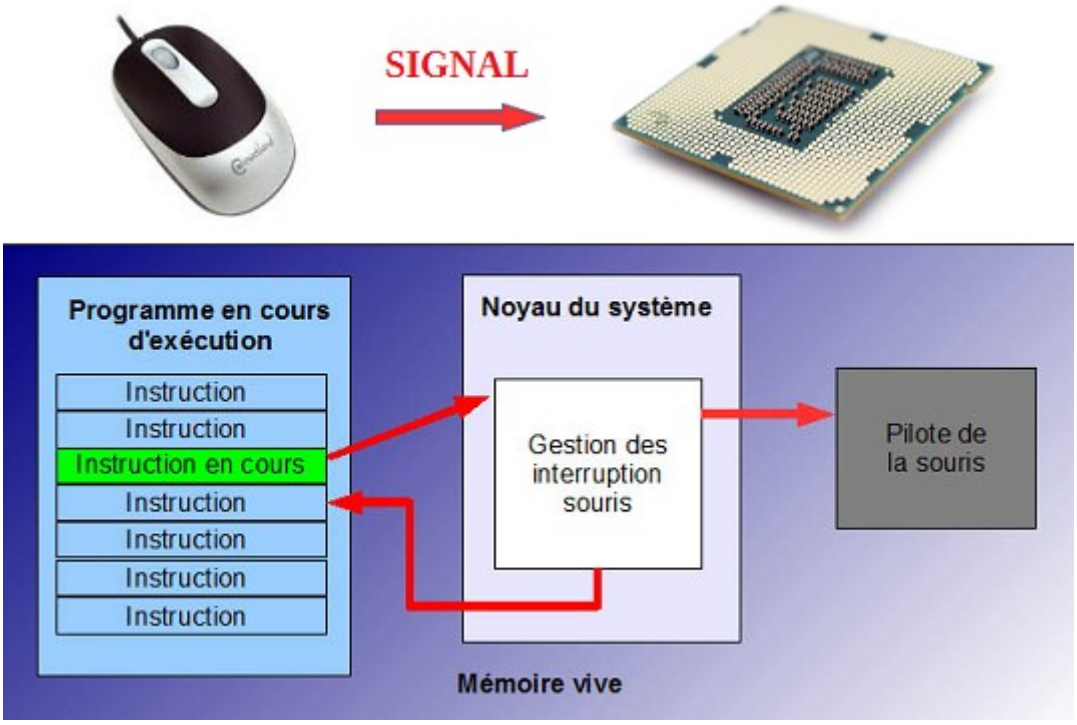
3.2. Les interruptions

Une interruption est une commutation du mot d'état provoquée par un signal généré par le matériel. Ce signal est la conséquence d'un événement interne au processus, résultant de son exécution, ou bien extérieur et indépendant de son exécution. Le signal va modifier la valeur d'un indicateur qui est consulté par le SE. Celui-ci est ainsi informé de l'arrivée de l'interruption et de son origine. A chaque cause d'interruption est associé un niveau d'interruption. On distingue au moins 3 niveaux d'interruption :

- les interruptions externes : panne, intervention de l'opérateur, ...
- les déroutements qui proviennent d'une situation exceptionnelle ou d'une erreur liée à l'instruction en cours d'exécution (division par 0, débordement, ...)
- les appels système

UNIX admet 6 niveaux d'interruption : interruption horloge, interruption disque, interruption console, interruption d'un autre périphérique, appel système, autre interruption.

Le chargement d'un nouveau mot d'état provoque l'exécution d'un autre processus, appelé le traitant de l'interruption. Le traitant réalise la sauvegarde du contexte du processus interrompu (compteur ordinal, registres, indicateurs,....). Puis le traitant accomplit les opérations liées à l'interruption concernée et restaure le contexte et donne un nouveau contenu au mot d'état : c'est l'acquittement de l'interruption.



Généralement un numéro de priorité est affecté à un niveau d'interruption pour déterminer l'ordre de traitement lorsque plusieurs interruptions sont positionnées. Il est important de pouvoir retarder, voire annuler la prise en compte d'un signal d'interruption. Les techniques que l'on utilise sont le masquage et le désarmement des niveaux d'interruption :

- le masquage d'un niveau retarde la prise en compte des interruptions de ce niveau. Pour cela, on positionne un indicateur spécifique dans le mot d'état du processeur. Puisqu'une interruption modifie le mot d'état, on peut masquer les interruptions d'autres niveaux pendant l'exécution du traitant d'un niveau. Lorsque le traitant se termine par un acquittement, on peut alors démasquer des niveaux qui avaient été précédemment masqués. Les interruptions intervenues pendant l'exécution du traitant peuvent alors être prises en compte
- le désarmement d'un niveau permet de supprimer la prise en compte de ce niveau par action sur le mot d'état. Pour réactiver la prise en compte, on réarme le niveau. Il est évident qu'un déroutement ne peut être masqué; il peut toutefois être désarmé.

3.3. Les ressources

On appelle ressource tout ce qui est nécessaire à l'avancement d'un processus (continuation ou progression de l'exécution) : processeur, mémoire, périphérique, bus, réseau, compilateur, fichier, message d'un autre processus, etc... Un défaut de ressource peut provoquer la mise en attente d'un processus.

Un processus demande au SE l'accès à une ressource. Certaines demandes sont implicites ou permanentes (la ressource processeur). Le SE alloue une ressource à un processus. Une fois une ressource allouée, le processus a le droit de l'utiliser jusqu'à ce qu'il libère la ressource ou jusqu'à ce que le SE reprenne la ressource (on parle en ce cas de ressource préemptible, de préemption).

On dit qu'une ressource est en mode d'accès exclusif si elle ne peut être allouée à plus d'un processus à la fois. Sinon, on parle de mode d'accès partagé. Un processus possédant une ressource peut dans certains cas en modifier le mode d'accès. Exemple : un disque est une ressource à accès exclusif (un seul accès simultané), une zone mémoire peut être à accès partagé.

Le mode d'accès à une ressource dépend largement de ses caractéristiques technologiques. Deux ressources sont dites équivalentes si elles assurent les mêmes fonctions vis à vis du processus demandeur. Les ressources équivalentes sont groupées en classes afin d'en faciliter la gestion par l'ordonnanceur.

3.4. L'ordonnancement

Dans un système d'exploitation, il est courant que plusieurs processus soient simultanément prêts à s'exécuter. II faut donc réaliser un choix pour ordonnancer dans le temps les processus prêts sur le processeur, qui est dévolu à un ordonnanceur.

Pour les systèmes (le traitement par lots. l'algorithme d'ordonnancement est relativement simple, puisqu'il consiste à exécuter le programme suivant de la file dès qu'un emplacement se libère dans la mémoire de l'ordinateur (multi programmation).

Pour les systèmes multi-utilisateurs, mufti-tâches, et multi-processeurs. L' algorithme d'ordonnancement peut devenir très complexe.

Le choix d'un algorithme d'ordonnancement dépend de l'utilisation que l'on souhaite faire de la machine. et s'appuie sur les critères suivants:

- équité : chaque processus doit pouvoir disposer de la ressource processeur; efficacité: l'utilisation du processeur doit être maximale.
- temps de réponse : il faut minimiser l'impression de temps de réponse pour les utilisateurs

interactifs.

- temps d'exécution : il faut minimiser le temps d'exécution pris par chaque travail exécuté en traitement par lots.
- rendement : le nombre de travaux réalisés par unité de temps doit être maximal.

En fait; plusieurs de ces critères sont mutuellement contradictoires, et l'on a montré[7] que tout algorithme d'ordonnancement qui favorise une catégorie de travaux le fait au détriment d'une autre.

Qui plus est, rien ne permet de connaître à l'avance les demandes en ressources de chacun des processus (E/S, mémoire, processeur) au cours de leur exécution, et donc le temps passé entre deux appels système. Pour assurer l'équité entre processus, il est donc nécessaire de mettre en œuvre un mécanisme de temporisation, afin de rendre la main à l'ordonnanceur pour que celui-ci puisse déterminer si le processus courant peut continuer ou doit être suspendu au profit d'un autre. On effectue alors un ordonnancement avec réquisition[8] du processeur, bien plus complexe à réaliser que le simple ordonnancement par exécution jusqu'à achèvement, car il implique la possibilité de conflits d'accès qu'il faut prévenir au moyen de mécanismes délicats (sémaphores ou autres).

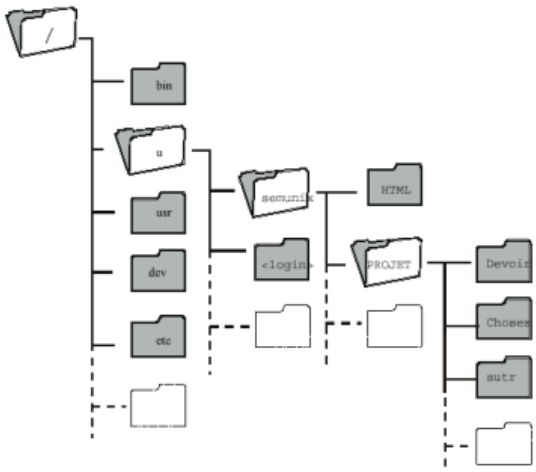
3.5. Le système de gestion de fichiers

Le stockage persistant, rapide, et fiable de grandes quantités de données (et de petites !) est un critère déterminant de l'efficacité d'un système d'exploitation.

Pour ce faire, on a très vite formalisé la notion de fichier, correspondant à un objet nommé, résidant en dehors de l'espace d'adressage des processus. mais disposant d'interfaces permettant la lecture et l'écriture de données dans ce dernier. Le nom même de « fichier » provient de l'histoire de l'informatique, lorsque les premières machines mécanographiques étaient exclusivement dédiées au classement et à la gestion de fichiers de cartes perforées. L'espace des fichiers et son organisation interne sont appelées génériquement « système de (gestion de) fichiers ».

Une des fonctions d'un SE est de masquer les spécificités des disques et des autres périphériques d'E/S et d'offrir au programmeur un modèle de manipulation des fichiers agréable et indépendant du matériel utilisé.

Les appels système permettent de créer des fichiers, de les supprimer, de lire et d'écrire dans un fichier. Il faut également ouvrir un fichier avant de l'utiliser, le fermer ultérieurement. Les fichiers sont regroupés en répertoires arborescents; ils sont accessibles en énonçant leur chemin d'accès (chemin d'accès absolu à partir de la racine ou bien chemin d'accès relatif dans le cadre du répertoire de travail courant).



Selon les systèmes, différentes organisations sont proposées aux utilisateurs pour organiser les données dans les fichiers. Ceux ci peuvent être organisés comme :

7 Kleinock 1975
8 Preemptive scheduling

- des suites d'octets : c'est l'organisation conceptuellement la plus simple. Le système de fichiers ne gère que des suites d'octets sans structure visible ; c'est leur interprétation par les différents programmes et le système (pour les fichiers considérés comme des exécutables) qui leur donne une signification. C'est l'organisation adoptée par de nombreux systèmes, comme les Un*x, DOS, ... ;
- des suites d'enregistrements : les fichiers sont structurés en enregistrements de taille fixe, qui ne peuvent être lus et écrits qu'en totalité, sans possibilité d'insertion au milieu de la liste. C'était en particulier le cas de CP/M;
- un arbre d'enregistrements de taille variable : les fichiers sont organisés en enregistrements de taille variable, indexés chacun par une clé, et groupés par blocs de façon hiérarchique. L'ajout d'un nouvel enregistrement en une position quelconque peut provoquer un éclatement d'un bloc en sous-blocs, tout comme la suppression d'un enregistrement peut provoquer la fusion de blocs peu remplis. Cette organisation, de type « fichier indexé », est proposée par le système de fichiers ISAM[9] d'IBM.

3.5.1. Types de fichiers

Dans tous les systèmes de fichiers, la plupart des informations de structure sont elles aussi considérées comme des fichiers (spéciaux), de même que certains moyens de communication inter-processus. Les types de fichiers les plus couramment définis sont les suivants :

- fichiers ordinaires : ils contiennent les données des utilisateurs.
- répertoires (ou catalogues) : structure du système de fichiers permettant d'indexer d'autres fichiers, de façon hiérarchique.
- fichiers spéciaux de type « caractère » : modélisent des périphériques d'entrée/sortie travaillant caractère par caractère, comme les terminaux (claviers et écrans), les imprimantes, ...
- fichiers spéciaux de type « bloc » : modélisent des périphériques d'entrée sortie travaillant par blocs, comme les disques.

3.5.2. Fichiers ordinaires

Dans la plupart des systèmes d'exploitation, les fichiers ordinaires sont subdivisés en plusieurs types en fonction de leur nature. Ce typage peut être :

- un typage fort : dans ce cas, le nomnmage des fichiers fait intervenir la notion d'extension, qui est gérée partiellement par le système (par exemple, sous DOS, un fichier doit posséder l'extension « bin », « com », ou « exe » pour pouvoir être exécuté par le système) .
- un typage déduit : les extensions des noms de fichiers ne sont qu'indicatives, et le système détermine la nature des fichiers par inspection de leur contenu (voir en particulier la commande « file » d'Unix) .
- un typage polymorphe : les fichiers représentent la sérialisation d'objets persistants dans des langages orientés objet ou fonctionnels, comme en Java par exemple. De façon interne, ces fichiers contiennent la description de la classe dont ils sérialisent une instance, ce qui les rapproche du typage déduit.

9 Indexed Sequential Access Method

La structure interne d'un fichier ordinaire dépend de son type. Elle peut être simple, comme dans le cas des fichiers texte, constitués de séquences de lignes terminées par des caractères spéciaux (« CR », ou « CR-LF »), lisibles sur un terminal sans traitement spécial. Elle peut être complexe lorsque les données sont organisées selon une structure interne dépendant du type du fichier, comme par exemple pour les fichiers exécutables d'Unix, dont l'information peut être extraite au moyen de nombreux outils (od, strings, nm).

3.6. La gestion de la mémoire

Plus que la ressource processeur, la mémoire constitue la ressource la plus critique des systèmes d'exploitation. dont le mésusage peut avoir des effets dramatiques sur les performances globales du système.

Les fonctionnalités attendues d'un gestionnaire efficace de la mémoire sont les suivante :

- connaître les parties libres de la mémoire physique.
- allouer de la mémoire aux processeurs, en évitant autant que possible le gaspillage.
- récupérer la mémoire libérée par la terminaison d'un processus.
- offrir aux processus des services de mémoire virtuelle, de taille supérieure à celle de la mémoire physique disponible, au moyen des techniques de va-et-vient[10] et de pagination.

Le système UNIX fonctionne en mémoire virtuelle paginée. Ceci permet de faire fonctionner des processus demandant une quantité d'espace mémoire supérieure à la mémoire physique installée.

Lorsqu'un processus demande l'allocation d'une page de mémoire et qu'il n'y en a pas de disponible en mémoire centrale, le noyau traite un défaut de page (voir le cours de système). Il choisit une page (qui n'a pas été utilisé depuis long- temps) et l'écrit sur une partition spéciale du disque dur. La place libérée est alors attribuée au processus demandeur.

Ce mécanisme demande la réservation d'une (ou plusieurs) partition spéciale sur l'un des disques durs, nommée partition de swap . La mémoire disponible pour les processus est donnée par la somme de la taille de mémoire physique (RAM) et des partitions de swap. Bien entendu, les performances du système se dégradent lorsque la fréquence des défauts de page augmente ; dans ce cas, il faut augmenter la mémoire physique.

Sur un système typique, la partition de swap est deux à trois fois plus grande que la mémoire centrale (exemple : PC avec 32Mo de RAM, partition de swap de 64Mo).

4. Structure d'un système d'exploitation

Le noyau est le programme qui assure la gestion de la mémoire, le partage du processeur entre les différentes tâches à exécuter et les entrées/sorties de bas ni- veau. Il est lancé au démarrage du système (le boot) et s'exécute jusqu'à son arrêt.

C'est un programme relativement petit, qui est chargé en mémoire principale. Le rôle principal du noyau est d'assurer une bonne répartition des ressources de l'ordinateur (mémoire, processeur(s), espace disque, imprimante(s), accès ré- seaux) sans intervention des utilisateurs. Il s'exécute en mode superviseur , c'est à dire qu'il a accès à toutes les fonctionnalités de la machine : accès à toute la mémoire, et à tous les disques connectés, manipulations des interruptions, etc.

10 swap

Tous les autres programmes qui s'exécutent sur la machine fonctionnent en mode utilisateur : ils leur est interdit d'accéder directement au matériel et d'utiliser certaines instructions. Chaque programme utilisateur n'a ainsi accès qu'à une certaine partie de la mémoire principale, et il lui est impossible de lire ou écrire les zones mémoires attribuées aux autres programmes.

Lorsque l'un de ces programmes désire accéder à une ressource gérée par le noyau, par exemple pour effectuer une opération d'entrée/sortie, il exécute un appel système . Le noyau exécute alors la fonction correspondante, après avoir vérifié que le programme appelant est autorisé à la réaliser.

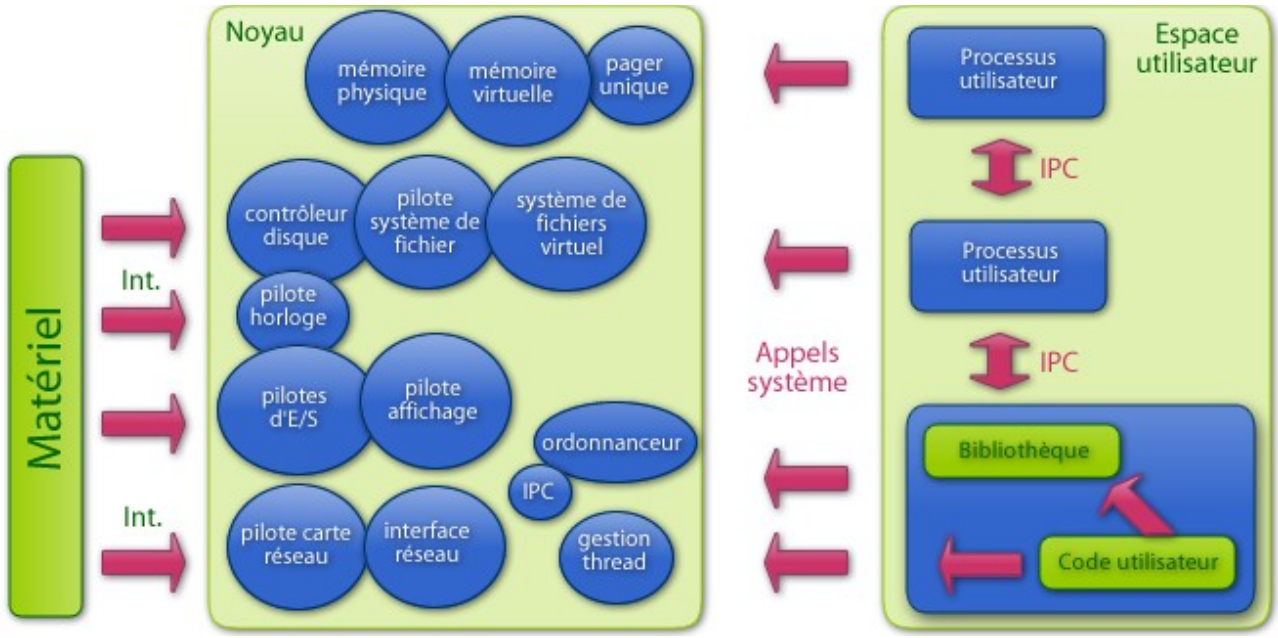
On peut distinguer quatre grandes catégories de SE.

4.1. Les systèmes monolithiques

Le SE est un ensemble de procédures, chacune pouvant appeler toute autre à tout instant. Pour effectuer un appel système, on dépose dans un registre les paramètres de l'appel et on exécute une instruction spéciale appelée appel superviseur ou appel noyau. Son exécution commute la machine du mode utilisateur[11] au mode superviseur ou noyau[12] et transfère le contrôle au SE. Le SE analyse les paramètres déposés dans le registre mentionné plus haut et en déduit la procédure à activer pour satisfaire la requête. A la fin de l'exécution de la procédure système, le SE rend le contrôle au programme appelant.

Généralement, un tel SE est organisé en 3 couches :

- une procédure principale dans la couche supérieure, qui identifie la procédure de service requise
- des procédures de service dans la couche inférieure à la précédente qui exécutent les appels système
- des procédures utilitaires dans la couche basse qui assistent les procédures système. Une procédure utilitaire peut être appelée par plusieurs procédures systèmes.



11 User space
12 Kernel space

4.2. Les systèmes en couches

On peut généraliser la conception précédente et concevoir un SE composé de plusieurs couches spécialisées, chaque couche ne pouvant être appelée que par des procédures qui lui sont immédiatement inférieures. Citons par exemple le premier SE de cette nature proposé par Dijkstra en 1968 :

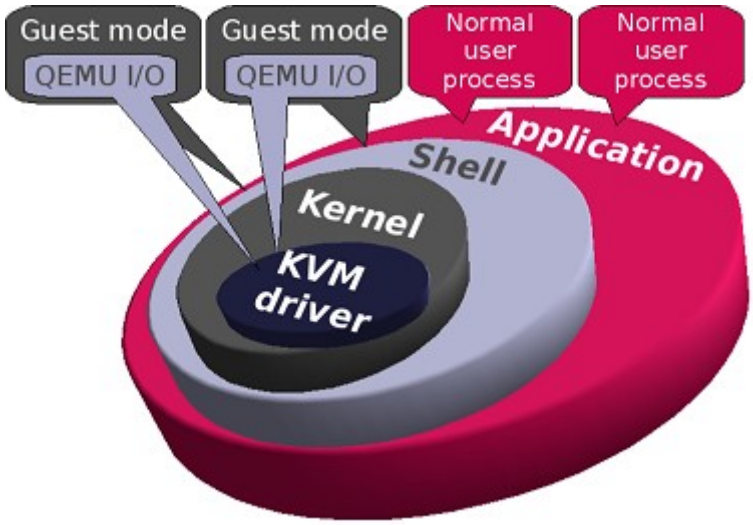
- couche 0 : allocation du processeur par commutation de temps entre les processus, soit à la suite d'expiration de délais, soit à la suite d'interruption (multiprogrammation de base du processeur)
- couche 1 : gestion de la mémoire, allocation d'espace mémoire pour les processus (pagination)
- couche 2 : communication entre les processus et les terminaux
- couche 3 : gestion des E/S (échanges d'information avec des mémoires tampons, c'est à dire avec des périphériques abstraits, dégagés des spécificités matérielles)
- couche 4 : programmes utilisateurs

4.3. Les machines virtuelles

Une des premiers SE à gérer le concept de machine virtuelle a été l'adaptation temps partagé de l'OS/360 d'IBM, proposé vers 1968 sous le nom de CP/CMS, puis sous le nom de VM/370 en 1979.

Le cœur du SE, appelé moniteur de machine virtuelle ou VM/370, s'exécute à même le matériel et fournit à la couche supérieure plusieurs machines virtuelles. Ces machines virtuelles sont des copies conformes de la machine réelle avec ses interruptions, ses modes noyau/utilisateur, etc...

Chaque machine virtuelle peut exécuter son propre SE. Lorsqu'une machine virtuelle exécute en mode interactif un appel système, l'appel est analysé par le moniteur temps partagé de cette machine, CMS. Toute instruction d'E/S, toute instruction d'accès mémoire est convertie par VM/370 qui les exécute dans sa simulation du matériel. La séparation complète de la multiprogrammation et de la machine étendue rend les éléments du SE plus simples et plus souples. VM/370 a gagné en simplicité en déplaçant une grande partie du code d'un SE dans le moniteur CMS.



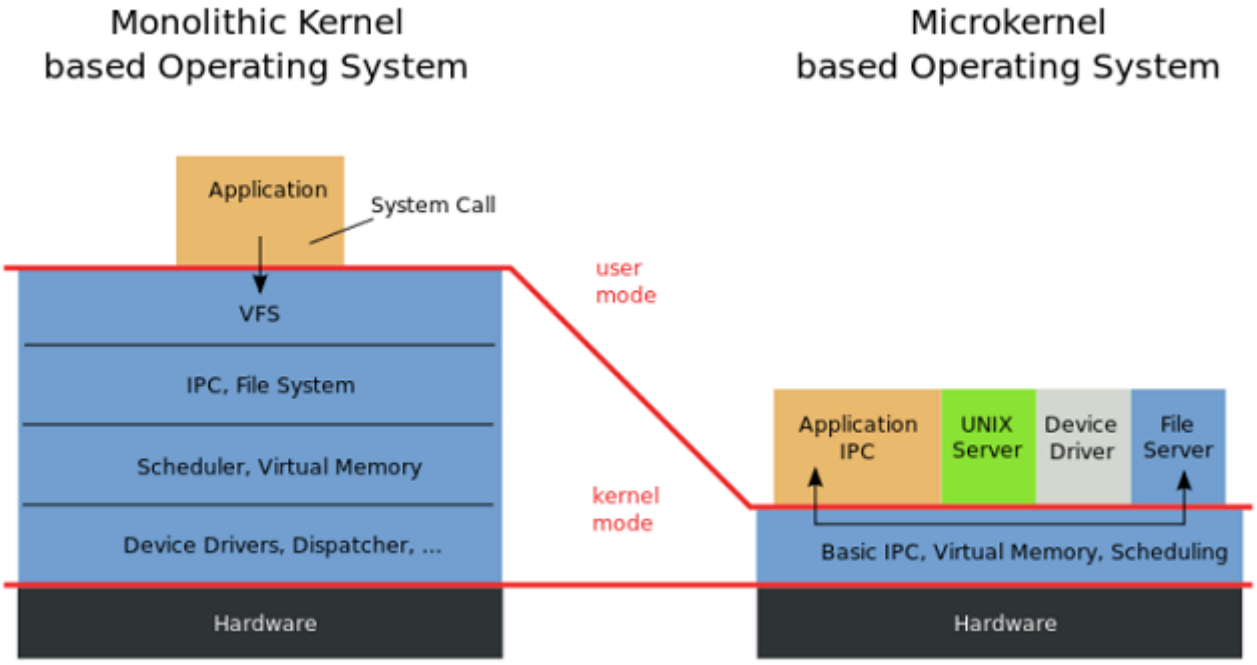
4.4. L'architecture client/serveur

Cette tendance s'est accentuée dans les SE contemporains en tentant de réduire le SE à un noyau minimal[13]. Une des formes les plus accentuées de cette évolution est l'architecture client/serveur.

La plupart des fonctionnalités d'un SE sont reportées dans des processus utilisateurs. Pour demander un service comme la lecture d'un bloc de fichier, le processus utilisateur ou processus client envoie une requête à un processus serveur qui effectue le travail et envoie une réponse. Le noyau ne gère que la communication entre les clients et les serveurs. Cependant, le noyau est souvent obligé de gérer certains processus serveurs critiques comme les pilotes de périphériques qui adressent directement le matériel.

La décomposition du SE en modules très spécialisés le rend facile à modifier. Les serveurs s'exécutent comme des processus en mode utilisateur et non pas en mode noyau. Comme ils n'accèdent donc pas directement au matériel, une erreur n'affecte que le serveur et pas l'ensemble de la machine.

En outre, ce modèle est bien adapté aux systèmes distribués. Un client n'a pas besoin de savoir si le SE fait exécuter sa requête par un serveur de sa propre machine ou celui d'une machine distante.



5. superordinateur

Dans le petit monde de l'informatique, les superordinateurs occupent une place à part : ces grosses machines, souvent utilisées pour faire des simulations numériques de phénomènes météorologiques, physiques, chimiques, ou autres, affichent des puissances de calcul impressionnantes.

En novembre dernier, à Austin (Texas), le nouveachine_smu classement des 500 ordinateurs les plus rapides du monde a été dévoilé. Ce classement, appelé le Top 500, parait deux fois par an depuis 1993 : en juin et en novembre, à l'occasion de la conférence International SuperComputing.

Ces ordinateurs, appelés supercalculateurs ou superordinateurs, sont des machines spéciales pour

13 micro kernel

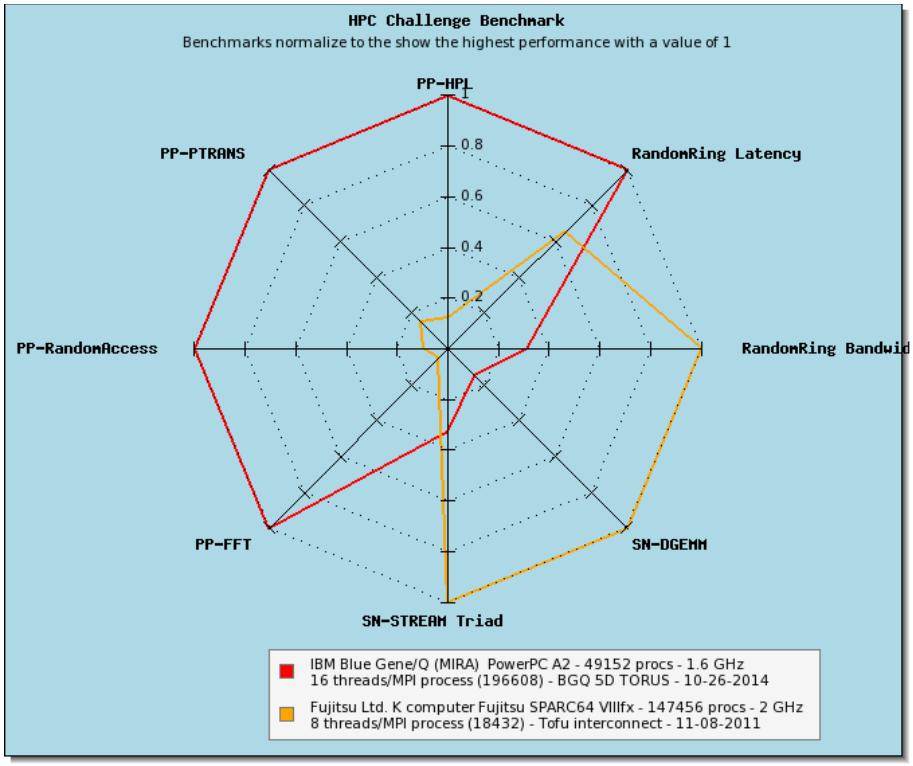
effectuer des calculs trop gros pour être faits par une machine de bureau. Ils ont une architecture spécifique, celle-ci ayant évolué au cours du temps. Ils sont pour la plupart hébergés dans des centres de calcul, et sont utilisés par un grand nombre d'utilisateurs qui se partagent leurs capacités.

5.1. Comment mesurer la performance des supercalculateurs ?

Tout d'abord, qu'est-ce que la performance des supercalculateurs ? C'est tout simplement la rapidité avec laquelle ils sont capables d'effectuer un calcul. Le Top 500 fournit deux chiffres : RPEAK et RMAX. Le RPEAK correspond à la puissance maximale théorique que peut fournir la machine. C'est une mesure très optimiste, qui ne tient pas compte des conditions réelles dans lesquelles les opérations sont effectuées. Cette mesure n'est donc pas vraiment équivalente à la performance du supercalculateur.

Le RMAX, lui, est mesuré par un vrai calcul, aux caractéristiques bien connues et représentatif des futurs calculs effectués sur ces machines : c'est ce qu'on appelle un benchmark. Le benchmark utilisé pour le Top 500 s'appelle LINPACK. Il effectue un calcul sur des matrices, très courant dans les applications de calcul scientifique, comme les résolutions de systèmes d'équations ou la simulation numérique. La performance est obtenue en divisant le nombre d'opérations de calcul effectuées par le benchmark par le temps pris par ce calcul. Il existe d'autres benchmarks, le plus connu étant le HPC Challenge : lui ne donne pas qu'un seul chiffre, mais un ensemble de mesures de différentes caractéristiques bien précises. Les résultats sont présentés sous forme d'un schéma appelé diagramme de cible.

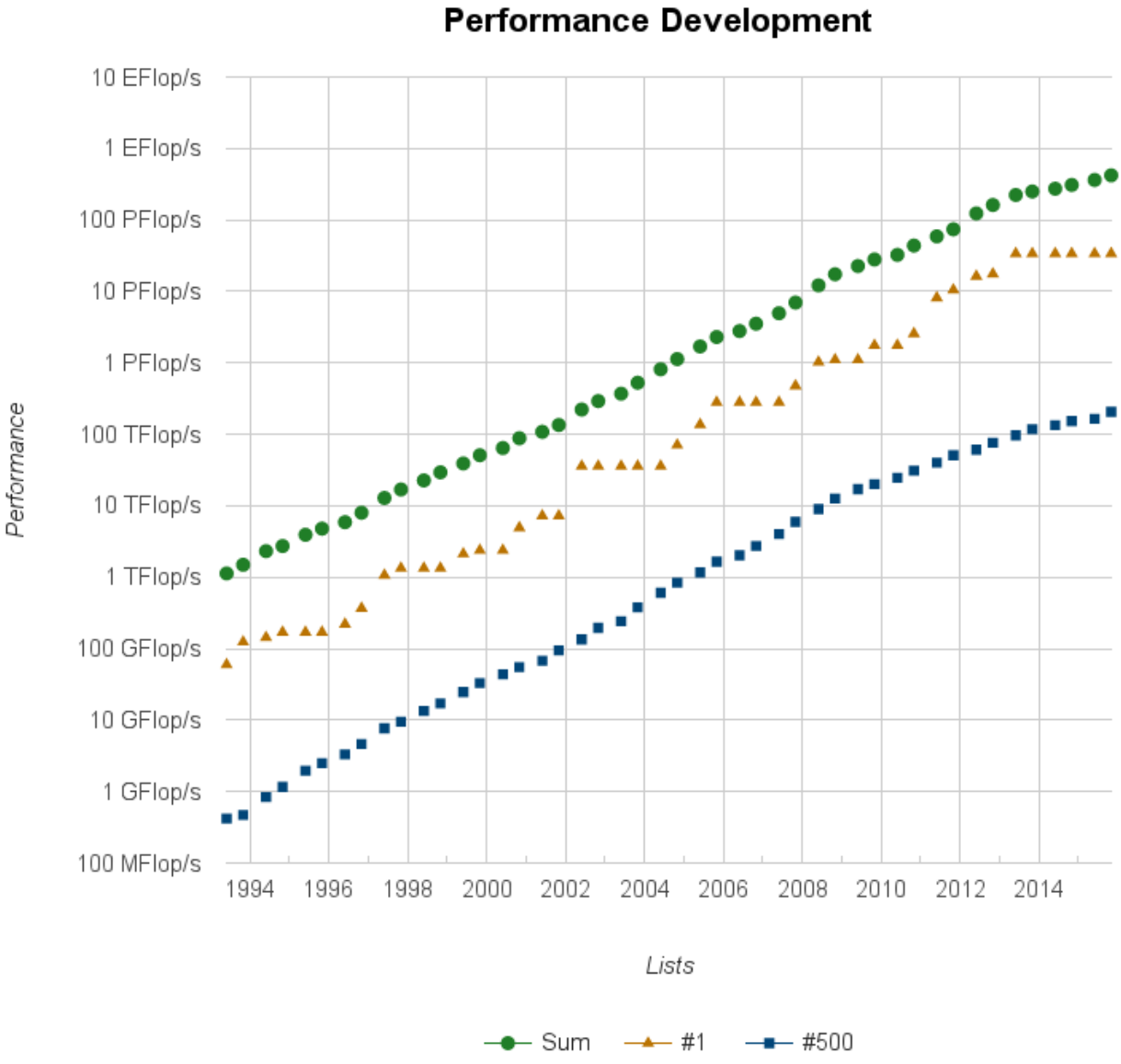
Par exemple, sur la figure suivante nous voyons le diagramme obtenu pour deux machines qui ont figuré tout en haut du Top 500 : la machine K, qui fut première en 2011 et actuellement quatrième, et la machine Mira, actuellement cinquième. On voit que, suivant ce qui est testé, ces deux machines ne s'illustrent pas du tout sur les mêmes caractéristiques.



5.2. Que nous apporte ce classement ?

Les retombées de ce classement sont multiples. Tout d'abord, ne le cachons pas, être dans le peloton de tête constitue une formidable vitrine pour plusieurs acteurs. Le propriétaire de la machine s'affiche ainsi en tant que centre de calcul majeur, mais le prestige revient aussi au constructeur de la machine, aux constructeurs des différentes pièces, à l'institution qui a financé la machine… Même pour des machines situées à des rangs plus modestes, afficher une entrée dans le classement Top 500 permet de donner un retour aux organismes de financement : « voici ce que nous avons fait de l'argent que vous nous avez confié, le centre de calcul possède une machine qui fait partie des plus rapides du monde et le monde entier le sait ».

Observer les caractéristiques des machines est aussi très intéressant, notamment si l'on regarde leur évolution au cours du temps. On peut voir par exemple l'évolution des architectures matérielles. Aujourd'hui, les machines les plus puissantes sont des clusters, c'est-à-dire des grappes de petits processeurs astucieusement reliés entre eux, dont les nœuds disposent souvent d'accélérateurs, comme des GPGPU (processeurs de cartes graphiques, en français ou presque) ou des processeurs Cell. On voit aussi que la performance des machines du Top 500 double tous les 18 mois.



Récemment, la consommation électrique des machines est apparue dans les tableaux de résultats du Top 500. Une liste alternative, le Green 500, classe les supercalculateurs en fonction du nombre d'opérations par seconde et par watt.

5.3. Le classement

La tête du classement est restée inchangée, avec en première position la machine Tianhe-2 du National Super Computer Center à Guangzhou. Cette machine est dotée de plus de trois millions de cœurs : 16.000 nœuds de calcul, chacun équipé de deux microprocesseurs Intel Xeon (12 cœurs chacun) et de deux accélérateurs Intel Xeon Phi. Les nœuds sont reliés par un réseau très rapide créé par le centre de recherche où la machine est située, appelé TH Express-2. Derrière cette machine on trouve Titan, un Cray XK7 appartenant au laboratoire national d'Oak Ridge (Tennessee), et en troisième position Sequoia, un IBM Blue Gene/Q appartenant au laboratoire national Lawrence Livermore (Berkeley, Californie), ces deux laboratoire dépendant du ministère américain de l'énergie. Quant aux machines françaises, elles pointent aux 33e (Total), 44e (CINES) et 53e (CEA) places.