DE LA RECHERCHE À L'INDUSTRIE

cea

ensiie
Paris-Évry | Strasbourg

www.cea.fr

Architecture d'un système d'exploitation : Les composants d'un système d'exploitation (cours)

Gilles.wiber@cea.fr

Septembre 2017

Plan

Introduction

Historique des systèmes d'exploitation

Besoins du HPC

Les principaux concepts d'un système d'exploitation

Le démarrage d'un système d'exploitation

Introduction / objectifs

Objectifs

Le cours d'aujourd'hui est un cours d'introduction au système d'exploitation dans le domaine du calcul haute performance (HPC), qui vous permettra de :

- Savoir ce qu'est un système d'exploitation (OS) ?
  - A quoi cela sert
  - Ses principaux composants
- Connaître / comprendre son fonctionnement général

Remarque :

Les séances suivantes permettront d'approfondir le fonctionnement des différents composants et leur importance dans le domaine du calcul haute performance (HPC).

Un système d’exploitation

Rappel : qu'est ce qu'un système d'exploitation ?

- **Ensemble de logiciels qui tournent en permanence sur un ordinateur et le contrôlent à partir de son démarrage (boot) et tant que celui-ci est allumé.**

- Exemples :
  - **Unix** : Créé en 1969, rapidement multi-utilisateurs, écrit en langage C.
  - **Linux** : Clone gratuit d'UNIX pour les PC, open source (1992)
  - **Mac OS** : Premier à proposer le concept des fenêtres, du glisser-déposer, la corbeille, le plug-and-play; aujourd'hui possède le noyau Linux, avec une interface graphique élégante et ergonomique, et optimisation particulière des traitements multimédia.
  - **MS-DOS** (Microsoft disque operating system) : OS des premiers PC, mono-utilisateur, mono-tâche, interface ligne de commande.
  - **MS-Windows** : Inspiré par l'interface Macintosh; tout d'abord, une coquille graphique pour DOS. Seulement à partir de Windows 95 nous commençons à assister à un transfert de nombreuses fonctionnalités de DOS vers Windows.
  - **Windows NT** : Système d'exploitation indépendant de DOS. Techniquement nettement supérieur à Windows
  - Sur des systèmes embarqués : **iOS**, **Android** (basé sur Linux), **Firefox OS**, **Windows Mobile**, **Maemo**, **Symbian**, ...

Historique des systèmes d’exploitation

Historique des systèmes d'exploitation

- Fin des années 40 : **Organisation par « roulement utilisateur »**
  - Le temps d'accès aux machines est découpé en période de 30min
  - Chaque période est alloué à un seul utilisateur
  - A la fin de chaque période, la machine est réinitialisée

- Début des années 50 : **Moniteur logiciel d'enchaînement séquentiel de travaux**

  Introduction des cartes perforées (cartes de programme, cartes de commande)
  - Lecture (des cartes), exécution, impression des travaux
  - Enchaînement des travaux

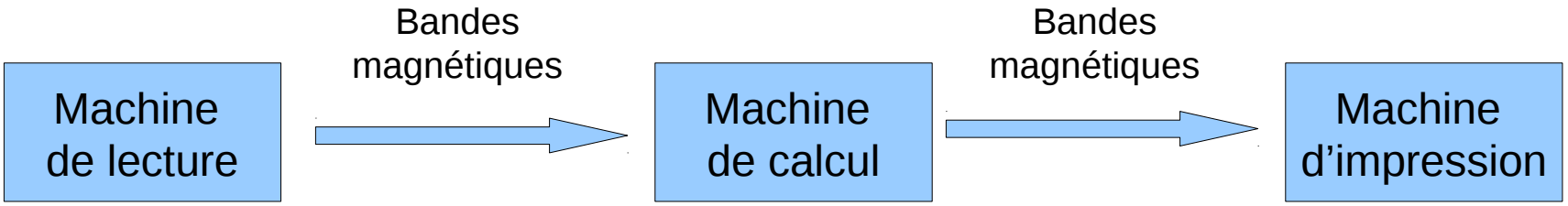
  Rem :

  Tout cela est fait séquentiellement (traitement du travail 1 puis 2, ...) et le moniteur est rechargé avant chaque travaux

- Milieu des années 50 : **Moniteur logiciel résidant d'enchaînement séquentiel des travaux**
  - Enchaînement automatique des travaux
  - Protection de la mémoire
  - Limitation de la durée (introduction d'une horloge)
  - Supervision des entrées / sorties

Historique des systèmes d'exploitation

- Fin des années 50 : **Apparition du traitement par lots (batch)**
  - Parallélisme entre les tâches de lecture, calcul et impression (avec spécification des machines et introduction des bandes magnétiques)



Remarque :

Avec la généralisation du disque magnétique, l'OS peut alors coordonner les 3 activités simultanément (mécanisme de spool). Chaque travail effectue des cycles de CPU (exécution de code) et des cycles d'entrées / sorties

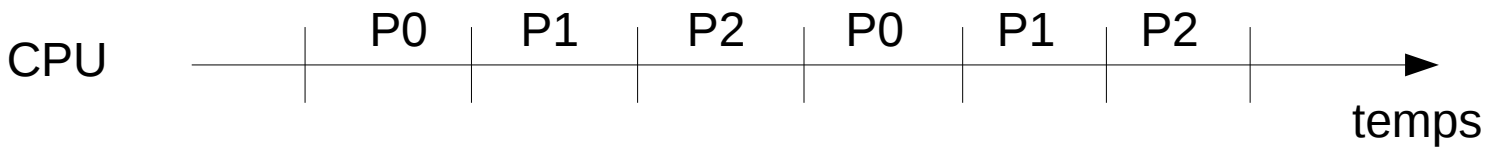
- Début des années 60 : **Multiprogrammation**
  - Plusieurs programmes résident simultanément en mémoire centrale (et lorsqu'un programme est en mode E/S, on peut lancé un autre en calcul)

=> Ré-implémentation du code, protection de la mémoire et E/S tamponnées

Historique des systèmes d'exploitation

- Dans les années 60/70 : **Apparition du temps partagé**
  - Le temps d'exécution du CPU est découpé en tranche de temps

    => cela répondait mieux à la demande des utilisateurs qui étaient connectés à des terminaux

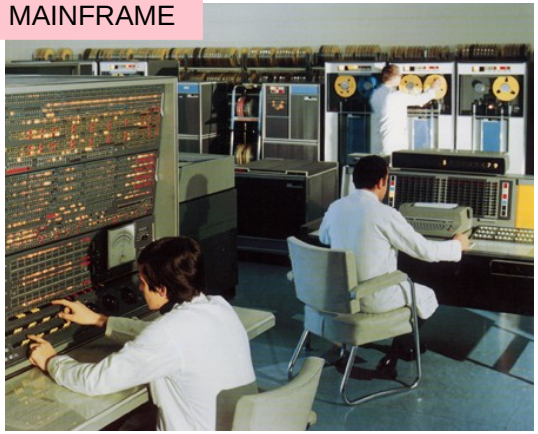


- A partir de 1980 : **Systèmes réparties**
  - Répartition d'une activité entre plusieurs machines reliées par un réseau

- Puis poussée toujours par la technologie (en particulier celle des réseaux devenant beaucoup + rapide) **Apparition des clusters de calcul** (ou grappe de calcul)
  - + évolutif, - spécifique et + robuste

Introduction au Calcul Haute Performance
Définitions, origines et cas d'utilisation

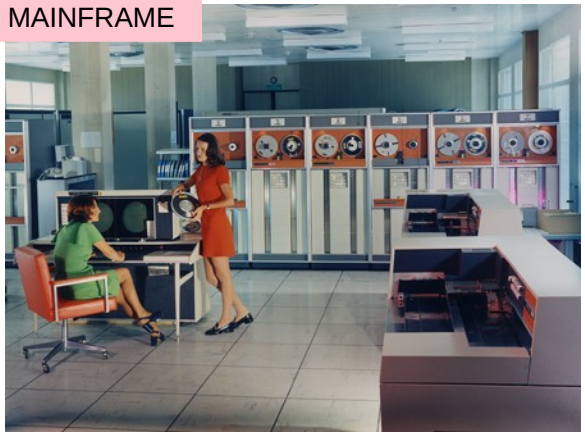
MAINFRAME



IBM 7030

1963…

MAINFRAME



CDC 6400

1970…

VECTORIEL

CRAY 1S

1982…

VECTORIEL

CRAY YMP

1990…

VECTORIEL

CRAY T90

1996…

MASSIVEMENT PARALLELE

TERA 1

2001…

Voir : www.cea-hpc.fr

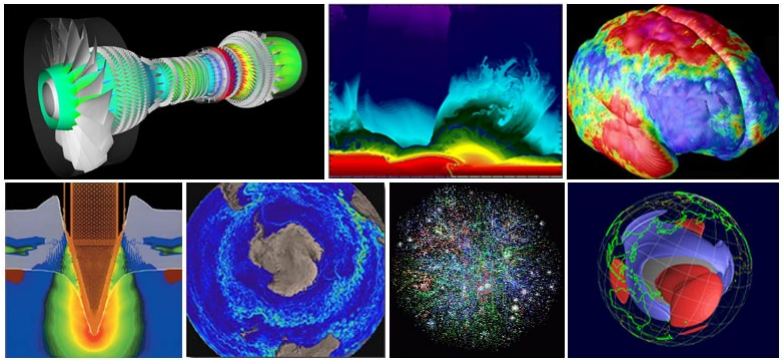
Besoins
du High Performence Computing

Le calcul haute performance (HPC)

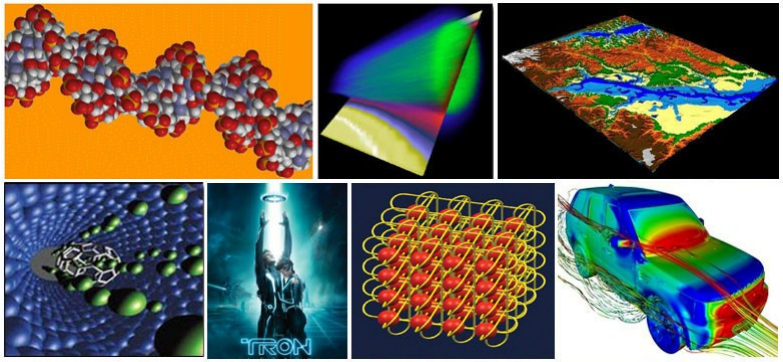
- **Résolution de problèmes scientifiques complexes nécessitant une très grande quantité de calculs informatiques**
  - Manipulant généralement de gros volumes de données
- **Utilisation des technologies informatiques les plus avancées**
  - Logicielles et matérielles
  - Afin de maximiser les performances et de réduire les délais de résolution
- **Notion de super-calculateur fournissant les ressources nécessaires**
  - Le terme HPC fait généralement référence au terme *supercomputer*
- **Mise en œuvre d'algorithmes de calcul complexes et adaptés aux architectures des super-calculateurs**
  - Permettant d'optimiser les applications pour maximiser les rendements de calcul

La simulation numérique

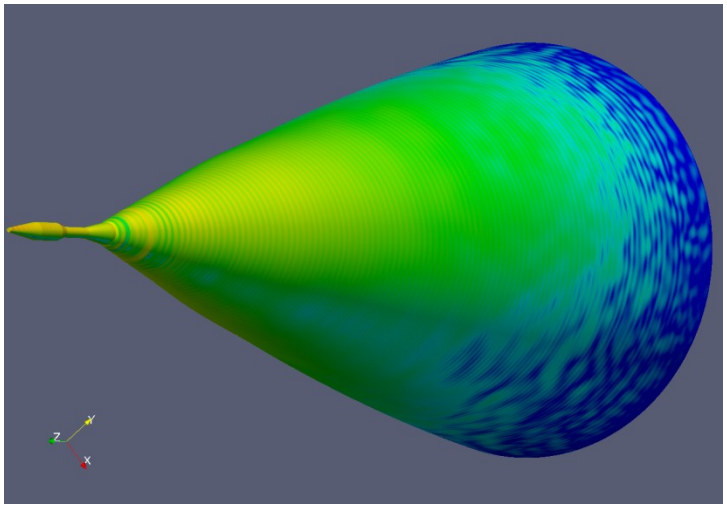
- **De nombreux domaines d'utilisation**



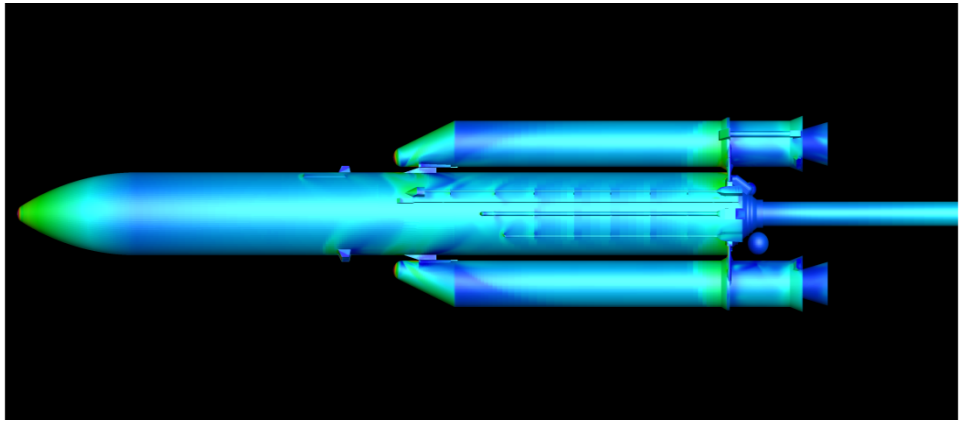
·Scientifiques
  - Mécanique
  - Environnement
  - Biochimie, Bio-informatique
  - Géologie, sismologie
  - Défense
  - ...



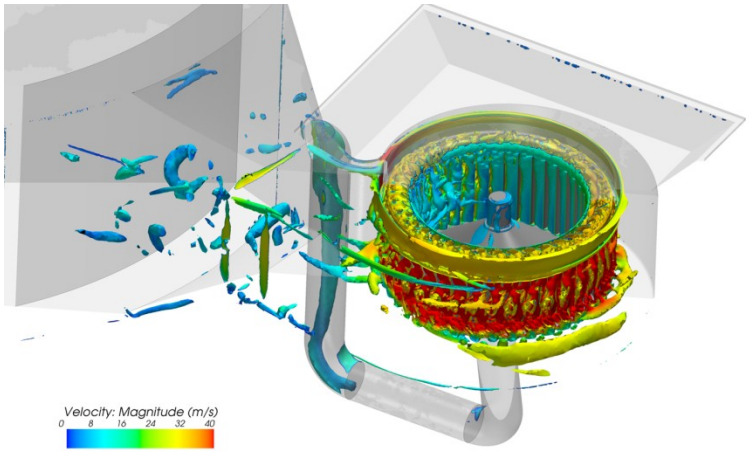
·Industriels, commerciaux, ...
  - Pharmacologie
  - Imagerie médicale
  - Exploration pétrolière/gazière
  - Finance
  - Multimédia
  - ...



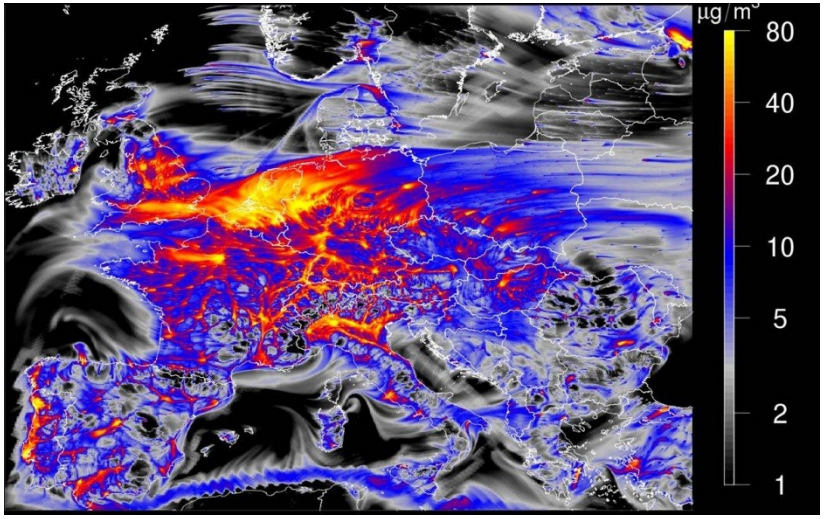
Courant de surface sur un radome de pointe avant d'avion de combat- **Thalès SAS**



Simulations aérodynamiques sur Ariane 5 – **Airbus DS**



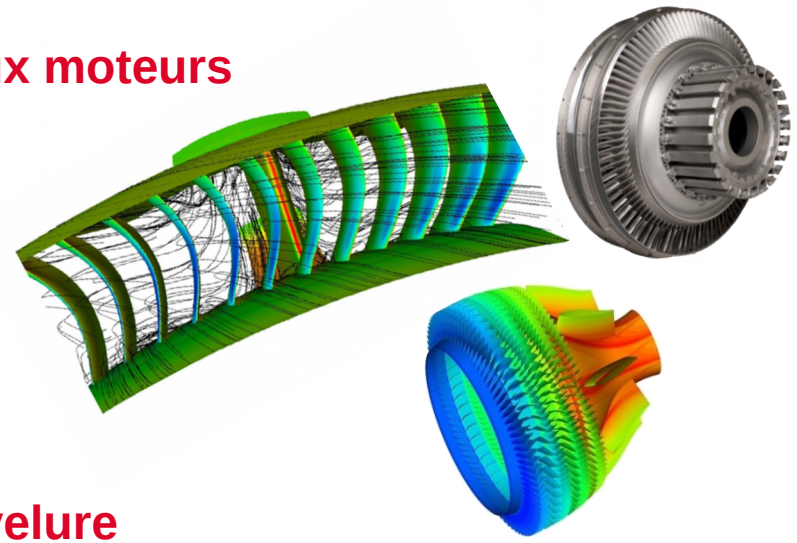
Simulations de la climatisation des véhicules - **Valéo**



Simulation d'un épisode de pollution particulaire d'ampleur transnationale à 2km de résolution sur l'Europe - **Inéris**

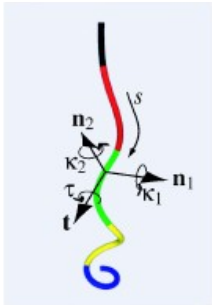
Safran : développement de 3 nouveaux moteurs

> Apport de la simulation :
Réduction de consommation d'env. 15 %
pour cette nouvelle génération de moteurs
grâce à la simulation

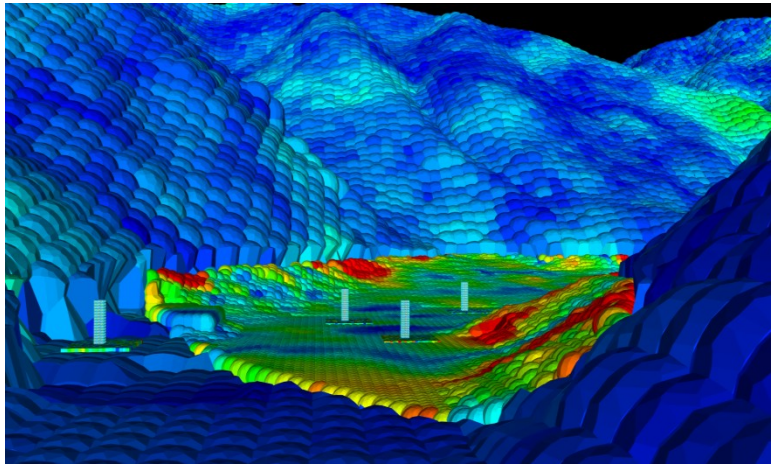


L'Oréal R&D : modélisation de la chevelure

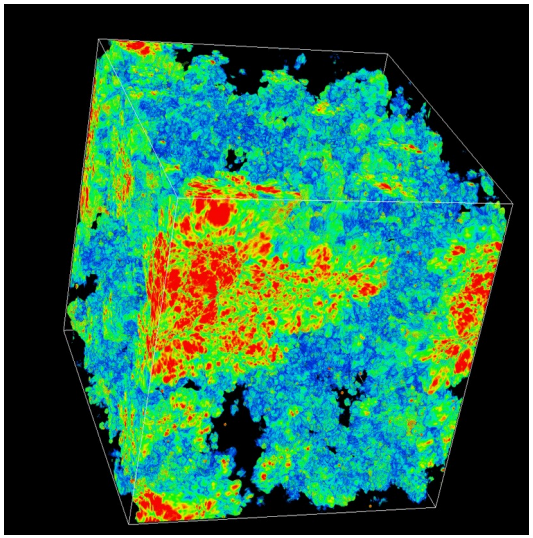
> Apport de la simulation :
Modélisation physiquement très réaliste de la chevelure.



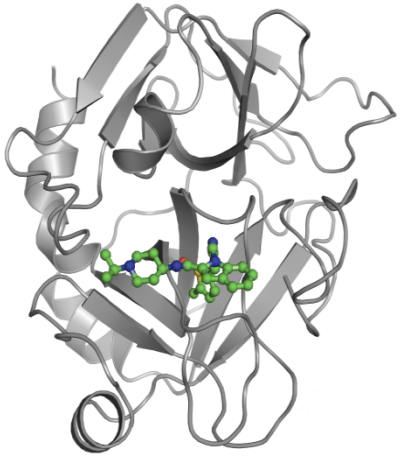
De la physique de la fibre au mouvement d'une chevelure.
Collaboration L'Oréal – Inria.



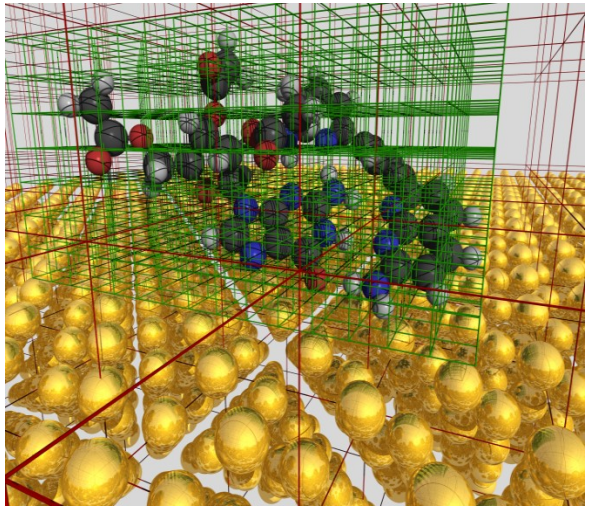
Simulation d'effets sismiques sur des bâtiments - CEA/DAM



Distribution du rayonnement ionisant de l'Univers…800 millions d'années après le Big-Bang CEA/DSM



Incidence de mutations sur des interactions protéine/ligand - CEA/DSV



Molécule sur une surface métallique, simulée par le code BigDFT- CEA/DSM

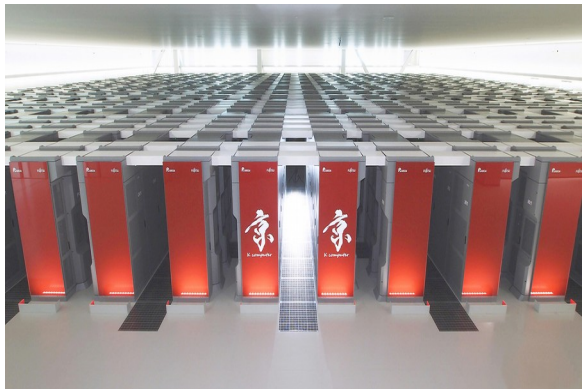
Les super-calculateurs

- **Très dépendants des technologies matérielles qui leur sont contemporaines**
  - L'objet désigné par le terme évolue donc au cours du temps

- **Aujourd'hui, un super-calculateur est dans la majorité des cas**
  - Une grappe de machines de calcul (cluster) et d'espaces de stockage fédérés au travers d'un ou plusieurs réseaux à haut débit et faible latence
  - Une machine fortement parallèle nécessitant des algorithmes de calcul adaptés

1997 – ASCI Red (Intel)

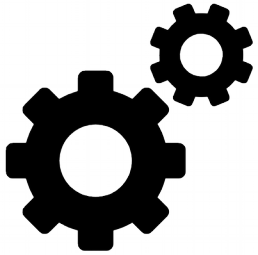
2008 – RoadRunner (IBM)

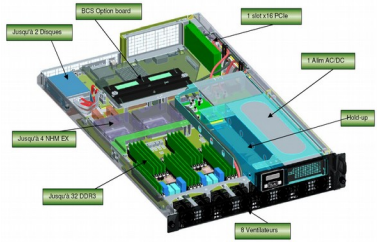


2012 – K (Fujitsu)

Introduction au Calcul Haute Performance
Composantes principales d'un super-calculateur

Logiciel

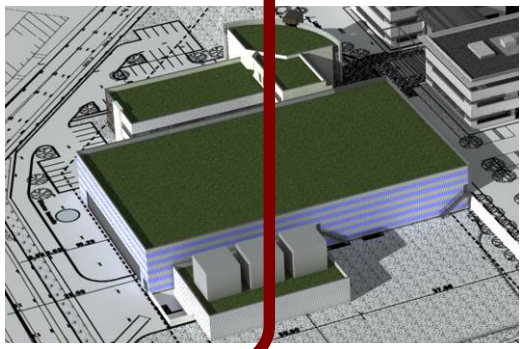




Nœuds de calcul

Interconnexion

Stockage

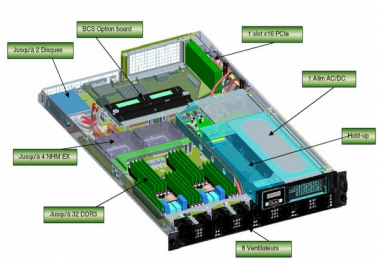


Infrastructures

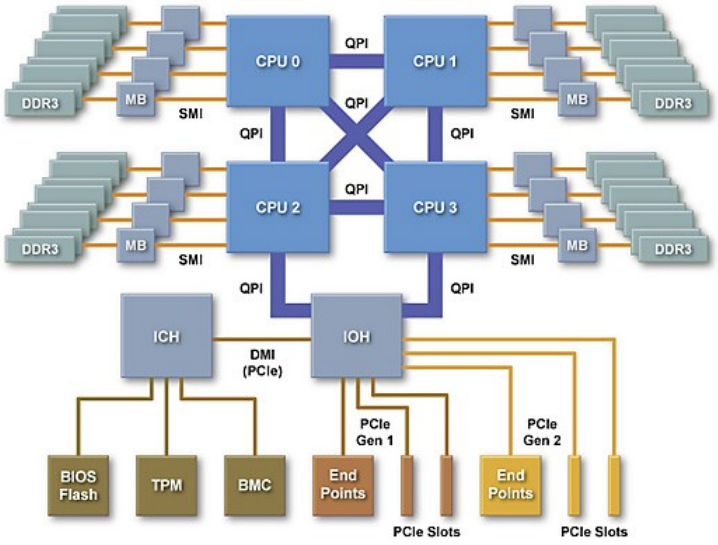
Composantes principales d'un super-calculateur
Les nœuds de calcul

Objectifs

- Fournir la puissance de calcul unitaire du système
  - Architecture multi-processeurs, multi-cœurs
    - Majoritairement NUMA (Non-Uniform Memory Access)
- Fournir le volume mémoire nécessaire à la résolution des problèmes
  - Ainsi que des débits mémoires optimums
- Fournir les accès aux réseaux de communication du calculateurs
  - Réseaux d'administration et de gestion
  - Réseaux rapides de calcul/stockage
- Minimiser l'espace/volume nécessaire
  - Pour maximiser le nombre de serveurs sur une même empreinte au sol
- Garantir un fonctionnement constant et optimal



Nœuds de calcul



Composantes principales d'un super-calculateur
Le réseau d'interconnexion

Objectifs

- Fédérer l'ensemble des entités de calcul et de stockage
  - Fournir l'ensemble des connexions point à point entre tous ces composants
- Fournir une très bonne qualité de service aux applications et aux systèmes de fichiers distribués
  - En terme de bande passante (débit)
  - En terme de latence
- Assurer la tolérance aux pannes des équipements d'interconnexion
  - Dans la limite de la conservation de la connectivité
  - En dégradant le service par une diminution des débits (congestion plus forte)

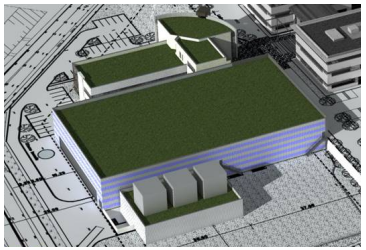
Composantes principales d'un super-calculateur
Le stockage

Objectifs

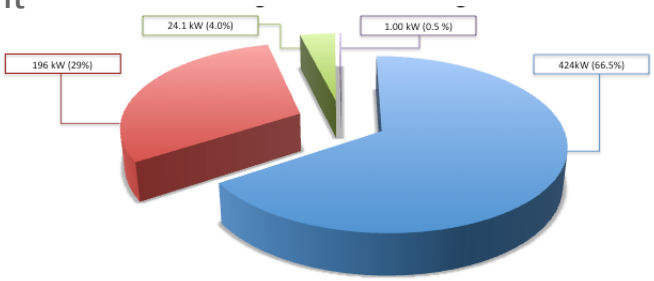
- Fournir un service de systèmes de fichiers distribués
  - Exp : NFS, GPFS, Lustre, ...
- Fournir les capacités de stockage nécessaires au bon déroulement des applications
  - Fonction du type d'applications exécutées et de leurs volumes produits
- Fournir les débits d'accès globaux attendus en lecture/écriture pour l'ensemble du calculateur
  - Plusieurs applications s'exécutent simultanément en production sur un calculateur
- Assurer la cohérence et la robustesse du service de stockage
  - Tolérance aux pannes matérielles courantes
- Assurer la conservation des données critiques sur de longues périodes de temps
  - Données coûteuses à générer nécessitant un archivage à long terme

Objectifs

- Accueillir l'ensemble des composants matériels
- Fournir la puissance électrique nécessaire
  - De l'ordre de plusieurs Mégawatt (MW)
    - Équivalent d'une ville de plusieurs milliers d'habitants
    - Pour alimenter le matériel informatique
    - Pour alimenter les servitudes
- Fournir la puissance de refroidissement nécessaire
  - Même ordre de grandeur que la puissance électrique
    - La plus grande partie de l'énergie électrique étant transformée en chaleur par effet Joules
- Optimiser la consommation énergétique globale
  - Notion de PUE (Power Usage Effectiveness)

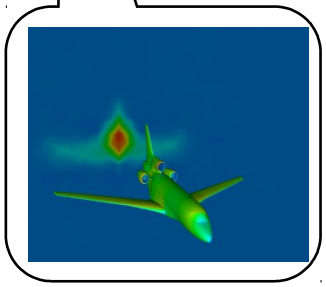


Infrastructures



Supercalculateur

Réseau haut débit



Climatisation, refroidissement

Ligne électrique

Stockage de données

Des équipes d'experts pluridisciplinaires pour piloter ces moyens exceptionnels et accompagner les utilisateurs

HPC : une optimisation à tous les niveaux

Afin de fournir la meilleure puissance de calcul, il faut optimiser à tous les niveaux sur tous les composants du HPC :

- Matériel
- Logiciel

=> optimisation / résilience au plus bas niveau

- Au niveau du nœud
  - Matériel
  - Système d'exploitation
- Pour aller + loin :
  - Actuellement OS généraliste mais on n'a pas besoin de tout :
    => différence entre un OS pour un PC portable et un nœud de calcul (n'étant la que pour faire des calculs)
  - Possibilité de gagner de la performance en simplifiant l'accès aux ressources locales (processeur, mémoire, ...)

Un système d’exploitation

Différents types de systèmes d'exploitation

De nos jours, les systèmes d'exploitation sont des systèmes de multi-programmation dirigés vers certains types d'applications, nous citons les trois types d'application les plus significatifs :

- **Système interactif**

  Un tel système a vocation à permettre à l'utilisateur d'intervenir pratiquement à toutes les étapes du fonctionnement du système et pendant l'exécution de son programme (Windows Xp, Linux sont de tels systèmes).

- **Système temps réel**

  Comme son nom l'indique, un système de temps réel exécute et synchronise des applications en tenant compte du temps : par exemple un système gérant une chaîne de montage de pièces à assembler doit tenir compte des délais de présentation d'une pièce à la machine d'assemblage, puis à celle de soudage etc…

- **Système embarqué**

  C'est un système d'exploitation dédié à des applications en nombre restreint et identifiées : par exemple un système de gestion et de contrôle des mesures à l'intérieur d'une sonde autonome, un système pour assistant personnel de poche, système pour téléphone portables se connectant à internet etc…

Rôles d'un système d'exploitation

Le système d'exploitation (SE ou OS pour Operating system) est chargé d'assurer la liaison entre les ressources matérielles, l'utilisateur et les applications.

L'OS a 2 rôles :

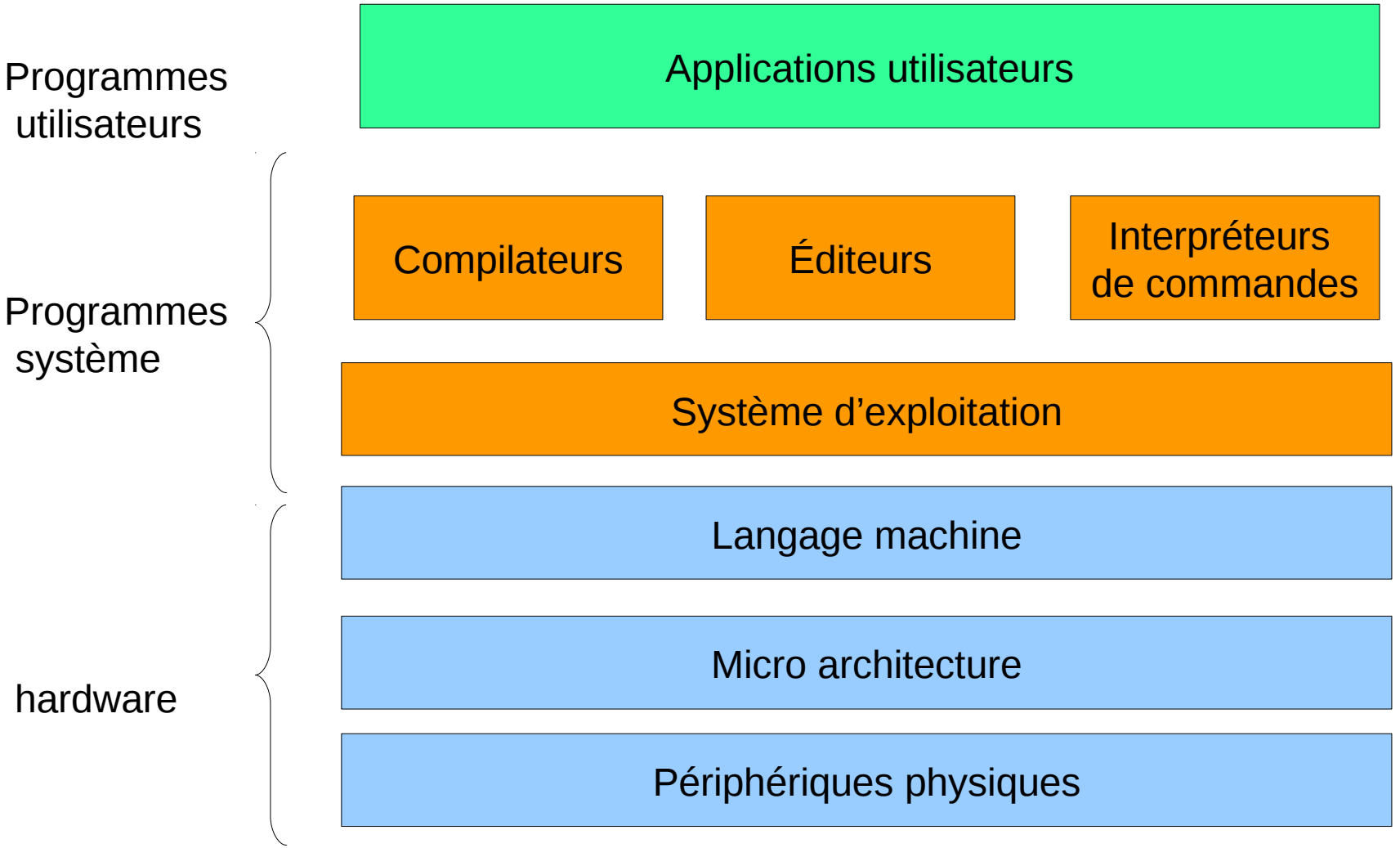
- Fournir à l'utilisateur une machine étendue ou virtuelle, plus simple à programmer
  - L'OS présente au programmeur une interface d'accès aux ressources de l'ordinateur (sous forme d'appels système). Ainsi le programmeur peut faire abstraction des détails de fonctionnement des ressources.
  - L'OS sécurise ainsi l'accès à la ressource matériel.

- Administrer et gérer efficacement les ressources
  - L'OS gère l'utilisation des ressources par différents utilisateurs et les éventuels conflits.
  - L'OS partage les ressources en 2 dimensions (multiplexage) : temps, espace

Remarque :

Les OS se développent pour faciliter l'utilisation sous-jacente du matériel

La place du système d'exploitation

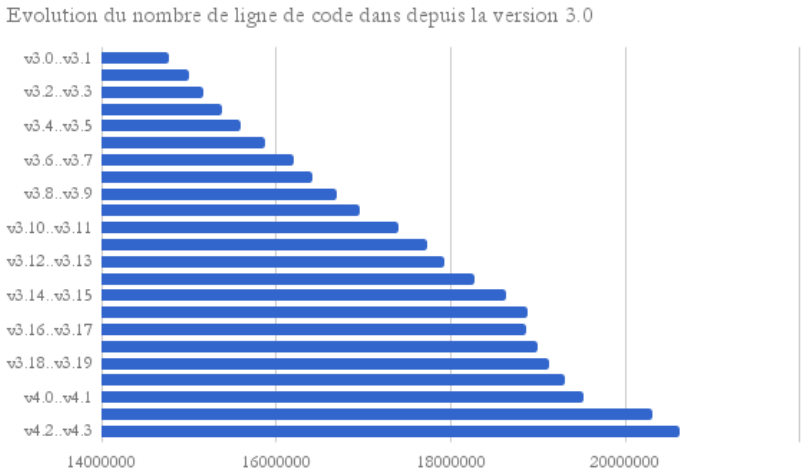


Tâches d'un système d'exploitation

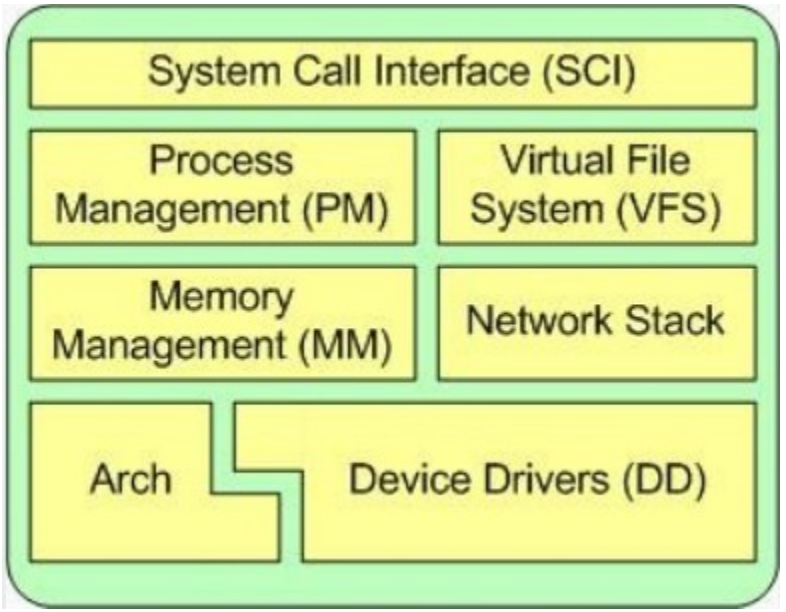
- Les différentes tâches d'un système d'exploitation :
  - Gestion de processus
  - Gestion de la mémoire
  - Gestion des fichiers
  - Gestion des E/S
- C'est le noyau (kernel) qui effectue ces tâches :
  Il tourne constamment dans la machine
- Les programmes utilisateurs peuvent accéder à ces différentes fonctionnalités à l'aide des appels système.

Le noyau (kernel)

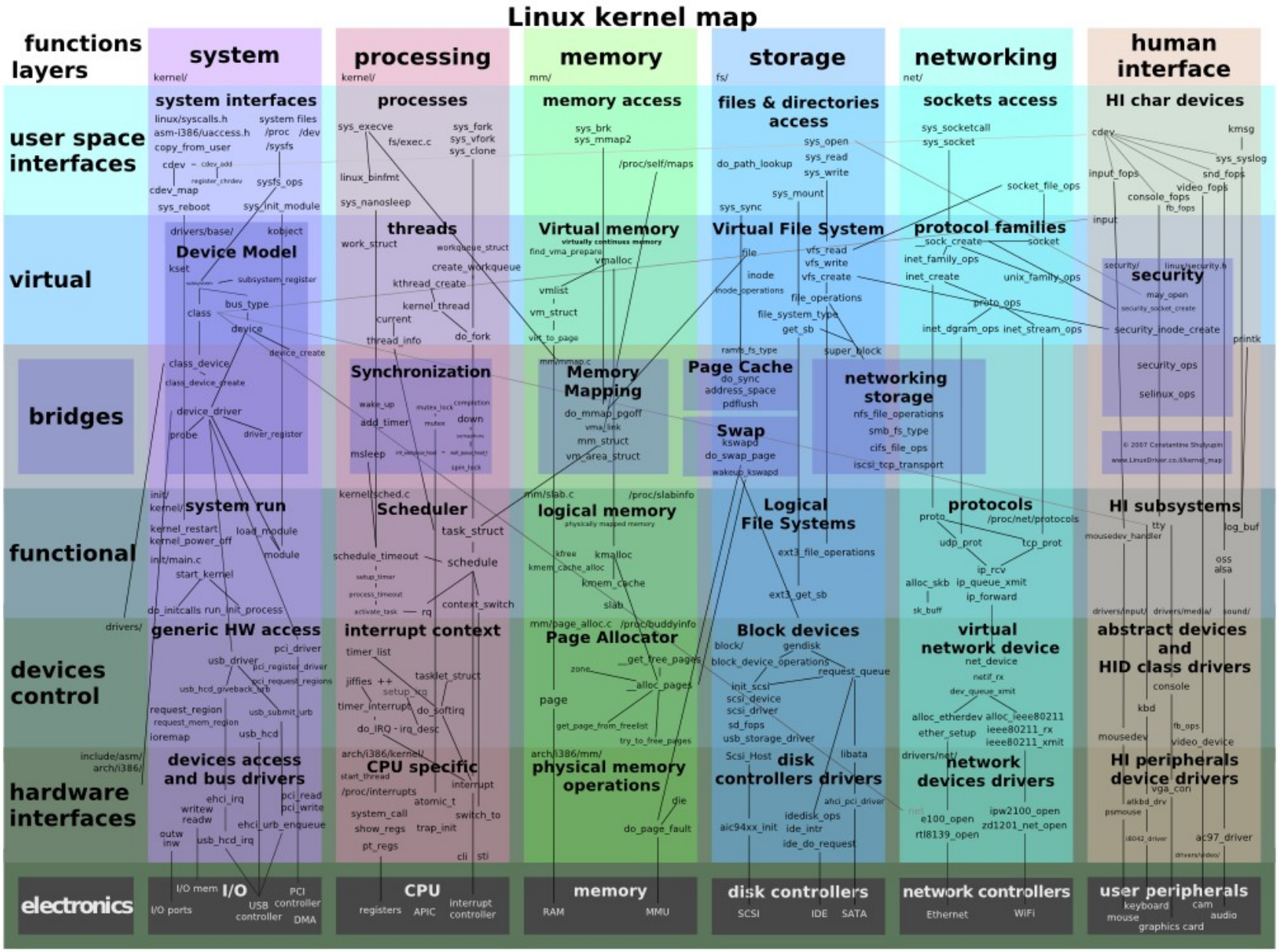
- Le noyau est un programme très complexe :
  + de 22,3 millions de lignes de code (Linux 4.9)



- En reprenant les tâches de l'OS développées précédemment, il est possible de découper son fonctionnement en quelques briques élémentaires, suffisantes pour une première approche de son fonctionnement.



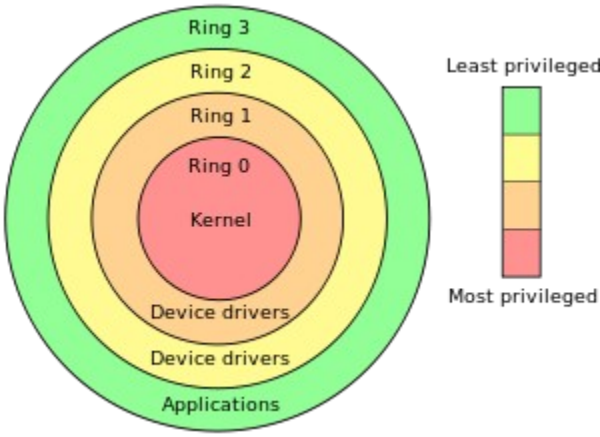
Le noyau (kernel)



Protection d'un système d'exploitation

Comment protéger le noyau des applications utilisateur ?

Avec une protection matérielle



- Le matériel permet 2 modes de fonctionnement :
  - **Mode moniteur ou superviseur ou système ou privilégié** (Monitor mode, supervisor mode, system mode) : exécution de la part de l'OS uniquement
    - Instructions privilégiées: instructions machine risquant de nuire.
    - Les instructions privilégiées sont exécutées seulement en mode superviseur.
  - **Mode utilisateur** (User Mode) : exécution de la part de l'utilisateur :
    S'il se produit une tentative d'exécuter une instruction privilégiée, le matériel ne la réalise pas mais traite l'instruction comme illégale et l'indique à l'OS.

Double modes de fonctionnement

- Au moment d'initialiser le système, le matériel démarre en mode superviseur
- Ensuite l'OS est chargé et démarre les processus utilisateurs en mode utilisateur
- Chaque fois qu'un événement (déroutement (trap) ou interruption (notions abordées un peu plus loin)) se produit, le matériel commute du mode utilisateur en mode superviseur :

  ⇒ Chaque fois que l'OS prend le contrôle de l'ordinateur, il est en mode superviseur

  ⇒ Le système commute toujours au mode utilisateur avant de donner la main à un programme utilisateur

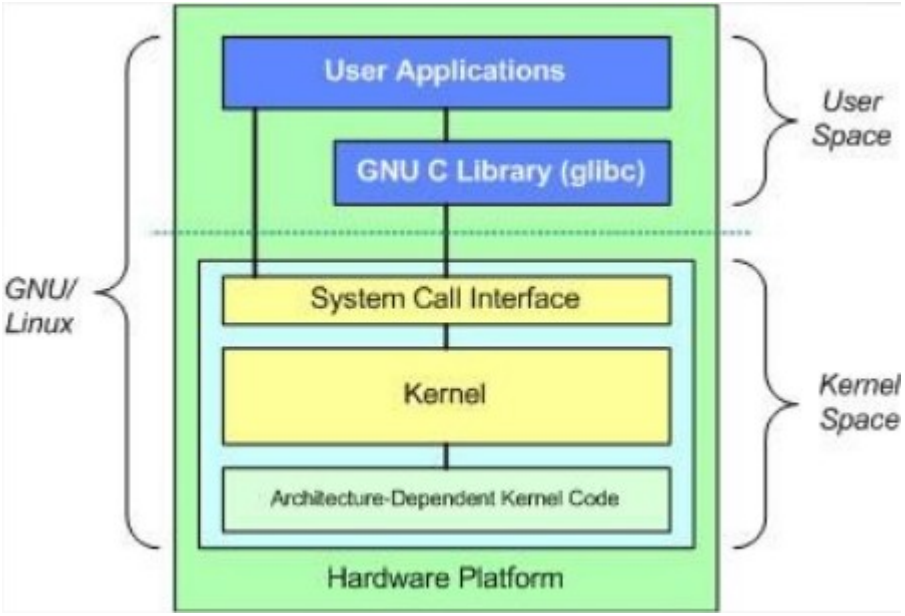
Remarque :

Lorsque l'on se logge root (administrateur) sur un système Linux par exemple, on est en mode utilisateur

Double modes de fonctionnement

- Certaines instructions sont privilégiées, comme les instructions E/S : donc utilisable qu'en mode superviseur ...
- Comment un programme utilisateur peut donc exécuter des E/S ?

En utilisant les appels système



Appels système (1/2)

- Un appel système est une fonction fournie par le noyau (kernel) d'un OS et utilisée par les programmes s'exécutant dans l'espace utilisateur (en d'autres termes, tous les programmes distincts du noyau).

- Quelques appels systèmes classiques :
  - open, read, write et close qui permettent les manipulations sur les systèmes de fichiers,
  - alloc, free pour allouer et désallouer de la mémoire.

- Sur la majorité des systèmes d'exploitations, les appels système peuvent être utilisés comme de simples fonctions écrites en C.

  Remarque :

  Sous Linux ils se trouvent dans la glibc.

- Sur la plupart des noyaux (notamment les noyaux monolithiques comme le Noyau Linux) les appels systèmes sont implémentés par une instruction machine (interupt, supervisor call, ...) qui fait basculer le processeur dans le noyau en mode superviseur (en ayant convenablement passé les paramètres de l'appel système, par exemple dans les registres).

Appels système (2/2)

Le « system call interface » est donc le composant qui gère l'interaction entre l'espace utilisateur et l'espace noyau. Il existe plus de 300 appels systèmes qui répondent à la norme POSIX (Portable OS for Unix) sous linux, les principaux sont :

- **Contrôle de processus** :
  - charger, exécuter, créer, terminer des processus, obtenir, signaler des événements, libérer de la mémoire, etc
- **Manipulation de fichiers** :
  - créer, supprimer, ouvrir, fermer, lire, écrire, repositionner,etc
- **Gestion de périphériques** :
  demander, libérer, obtenir, attacher, etc
- **Entretient d'information** :
  - obtenir, définir l'heure ou la date, définir les données du systèmes
- **Communications** :
  - créer, supprimer des connexions de communication, envoyer, recevoir de messages, transférer des informations sur les états, etc

Remarque :

Vu du programme applicatif, un appel système est atomique (il s'est exécuté -éventuellement en erreur- ou pas).

Les principaux concepts

Les composants principaux

Les différentes tâches d'un système d'exploitation :

- Gestion de processus
- Gestion de la mémoire
- Gestion des E/S
- Gestion des fichiers

Les processus

Processus : définition

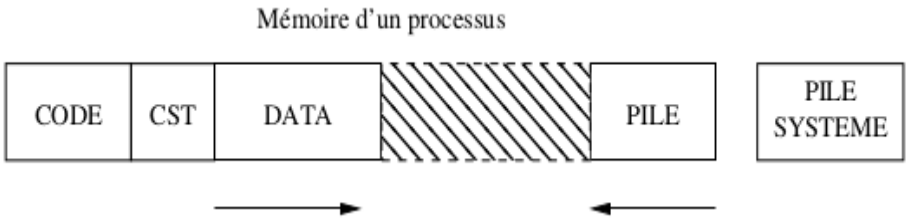
Objectifs

- Toutes les applications semblent progresser simultanément :
  - Les processeurs physiques les exécutent en alternance
  - Chaque application a l'illusion d'être seule sur la machine
- Partager les ressources disponibles entre de multiples applications

=> introduction de la notion de processus

C'est un **programme en cours d'exécution**. Chaque processus possède :

- un espace d'adressage qui contient :
  - le programme exécutable
  - ses données
  - sa pile



- un ensemble de registres dont :
  - le compteur ordinal (contient l'adresse de la prochaine instruction)
  - le (ou les) pointeur(s) de pile
- d'autres registres matériels et informations nécessaires.

Processus : rappel de l'exécution

Une exécution d'un programme c'est une évolution *discrète* de l'état de la machine

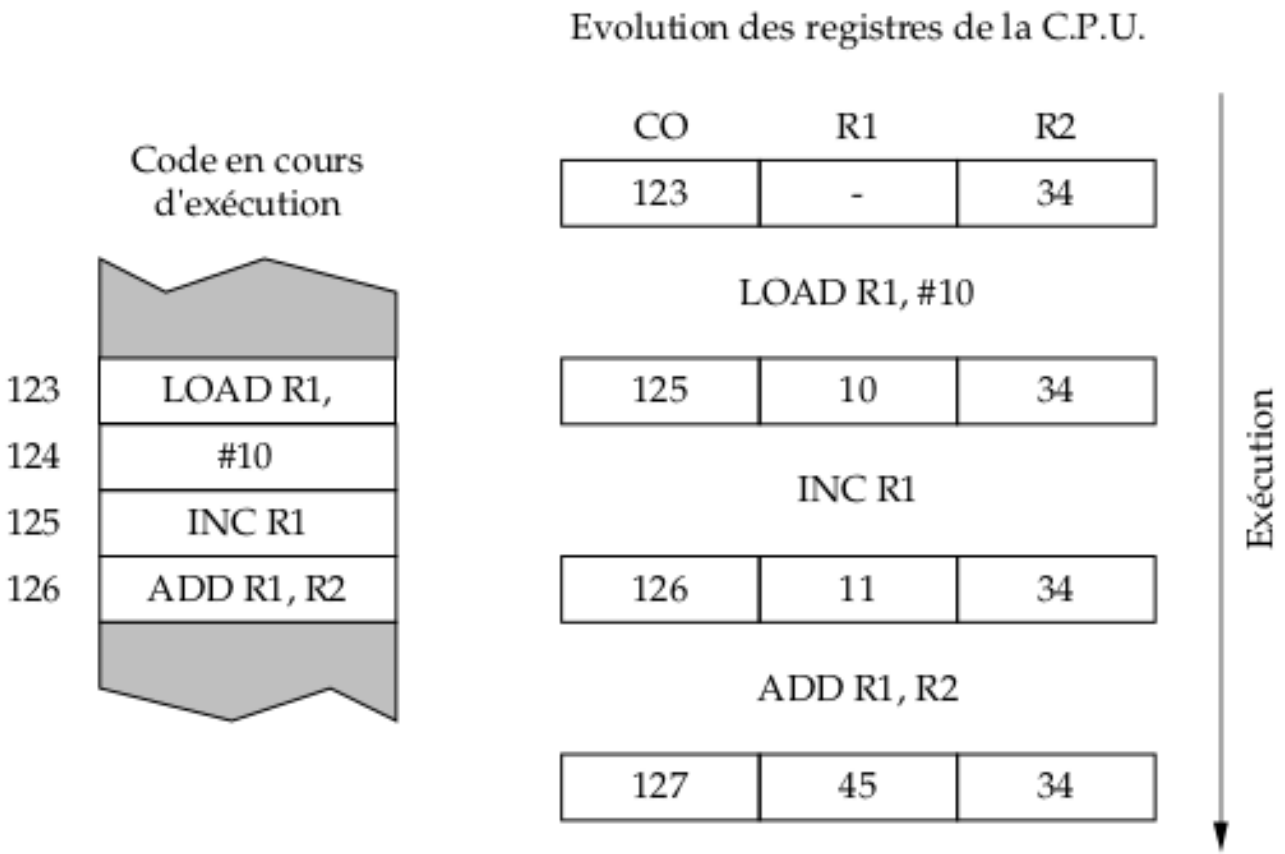


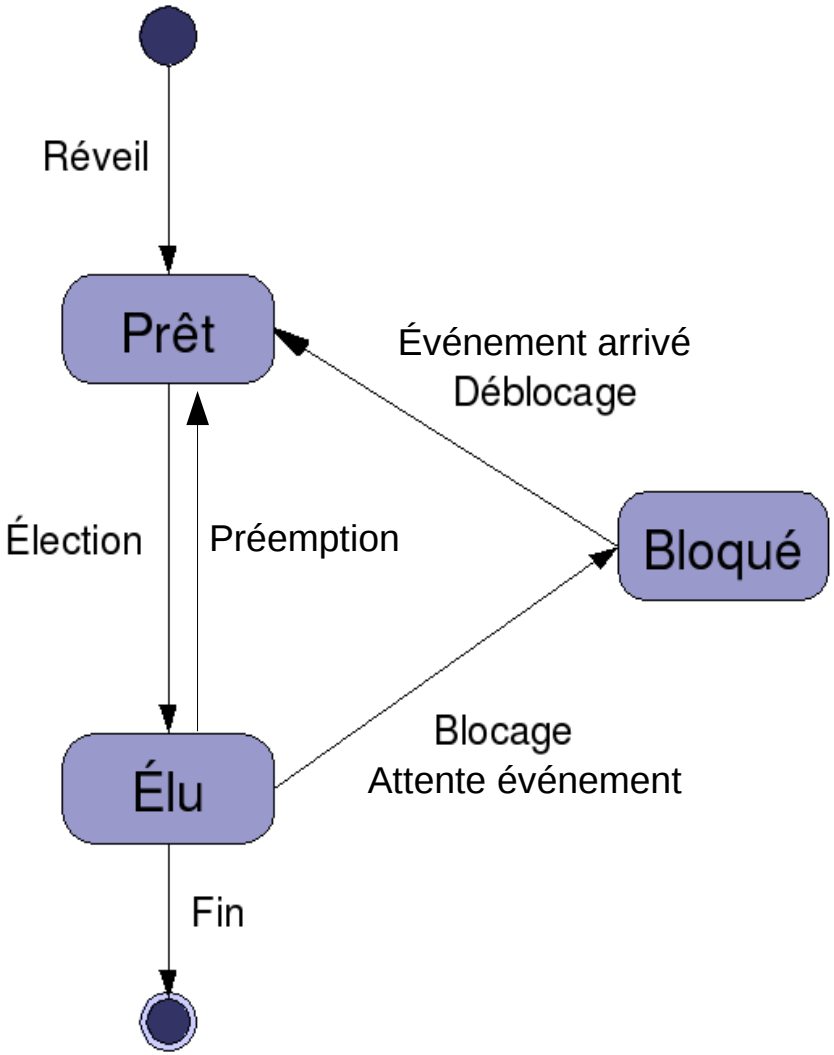
FIG. 2.1 – Un exemple d'exécution

Processus : états

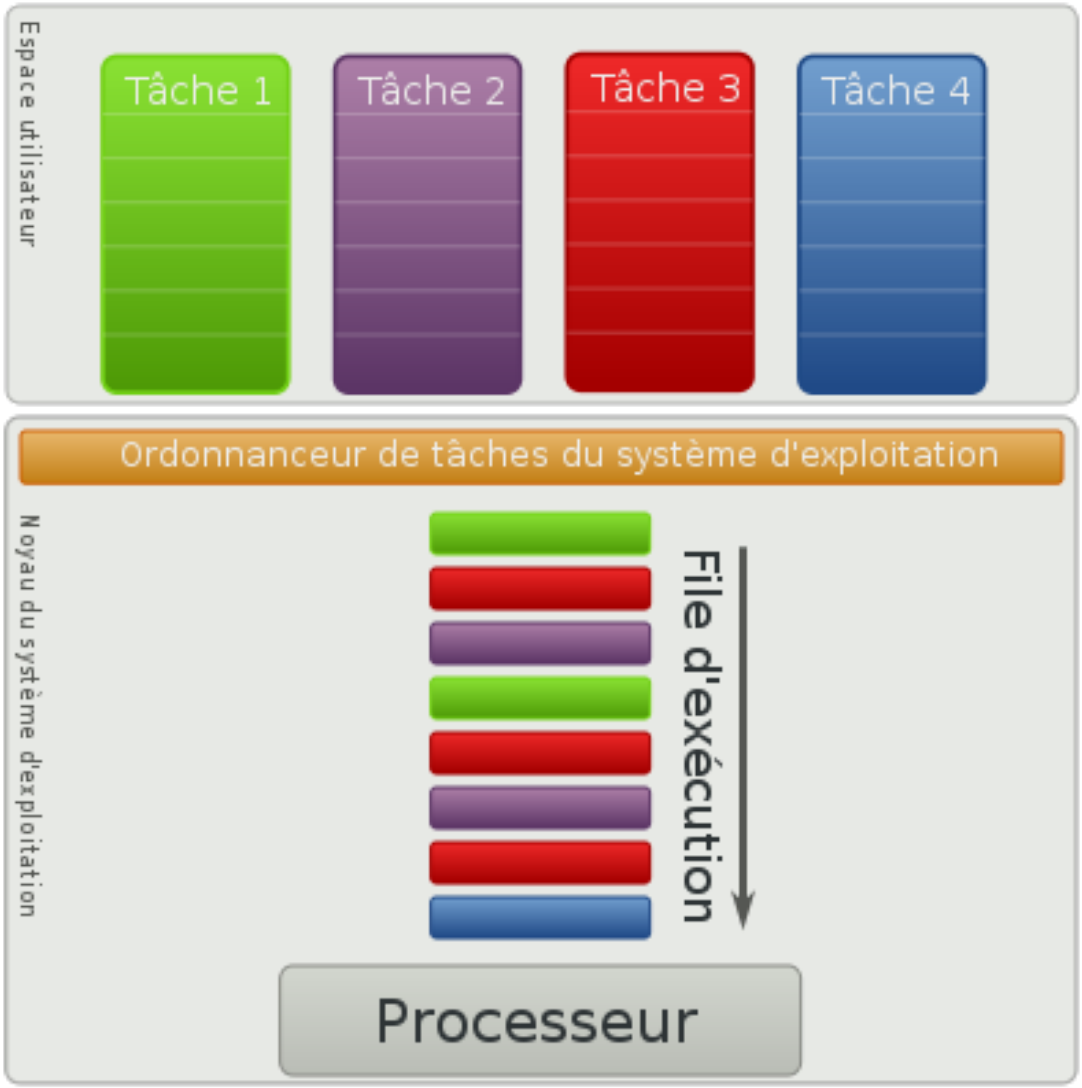
- Un processus peut prendre un de ces 3 états :
  - Élu (le programme utilise le processeur)
  - Prêt (exécutable, temporairement arrêté pour laisser un autre processus)
  - Bloqué (ne peut pas s'exécuter tant qu'un événement externe ne se produit pas)

- Le passage de Prêt à Élu et inversement est géré par l'ordonnanceur de processus (grâce aux interruptions)

  Remarques :
  - Le processeur bascule constamment et très rapidement d'un processus à l'autre (time sharing).
  - Beaucoup de gestion de files (d'attente, d'exécution, ...) dans le noyau



Processus : fonctionnement



Interruptions : définition

- Une Interruption est un événement qui modifie le flux de commande d'un programme
- Il existe 2 types d'interruption :
  - **Interruptions matérielles** :
    Elles permettent la prise en compte d'une requête de service système (mémoire, contrôleur de périphérique, clavier, lecteur, ...). À tout moment le matériel peut activer une interruption.
  - Interruptions logicielles :
    Elles sont activées par l'exécution d'un appel système (system call ou monitor call)

Interruptions : traitement d'interruption (déroutement)

- Le programme en cours est arrêté.
- Le système d'exploitation préserve l'état de la CPU (sauvegarde des registres et du compteur ordinal).
- L'OS détermine le type d'interruption.
- Pour chaque type d'interruption, une partie de code de l'OS détermine l'action qui doit être prise.
- Dès que cette procédure est terminée, le programme interrompu reprend son exécution
- Lors de la reprise, la machine doit se trouver exactement dans l'état où elle était au moment de la prise en compte de l'interruption.

Remarque :

Les OS modernes sont orientés interruptions (interrupt driven). Ils sont alors plus réactifs et consomment moins de ressources.

Processus : création / arrêt

- Événements conduisant à la création d'un nouveau processus
  - Initialisation du système
  - Exécution d'un appel système de création de processus par un processus en cours
  - Requête utilisateur sollicitant la création d'un nouveau processus
  - Initiation d'un travail en traitement par lots

- Événements conduisant à l'arrêt d'un processus :
  - Arrêt normal (volontaire) (exit)
  - Arrêt pour erreur (volontaire)
  - Arrêt pour erreur fatale (involontaire)
  - Le processus est arrêté par un autre processus (volontaire) (kill)

La mémoire

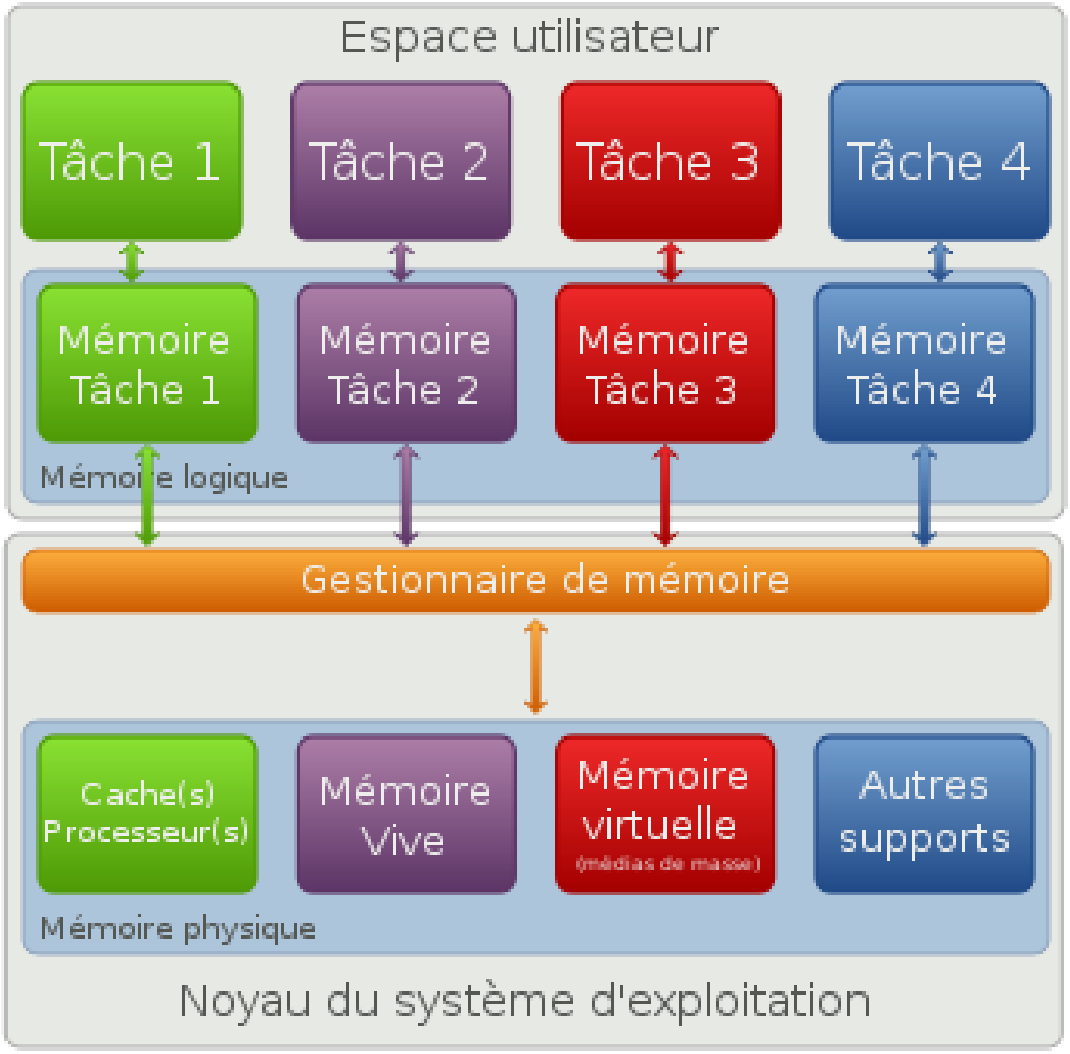
Rôle du gestionnaire de mémoire

- Fournir une abstraction pour le développeur pour simplifier / optimiser son utilisation
- Séparer au niveau mémoire les différents processus
- Supporter / coordonner la hiérarchisation de la mémoire (cache, RAM, disque dur)
- Gérer la mémoire des processus
  - conserver la trace de la mémoire en cours d'utilisation ou pas
  - allouer la mémoire aux processus qui en ont besoin
  - gérer le va-et-vient (swapping) entre mémoire principale et disque

Remarque :

La mémoire est utilisée par petits blocs (pages) de même taille : c'est la pagination. Ce mécanisme permet une meilleure gestion mémoire (peu de fragmentation)

Gestion de la mémoire



Mémoire virtuelle

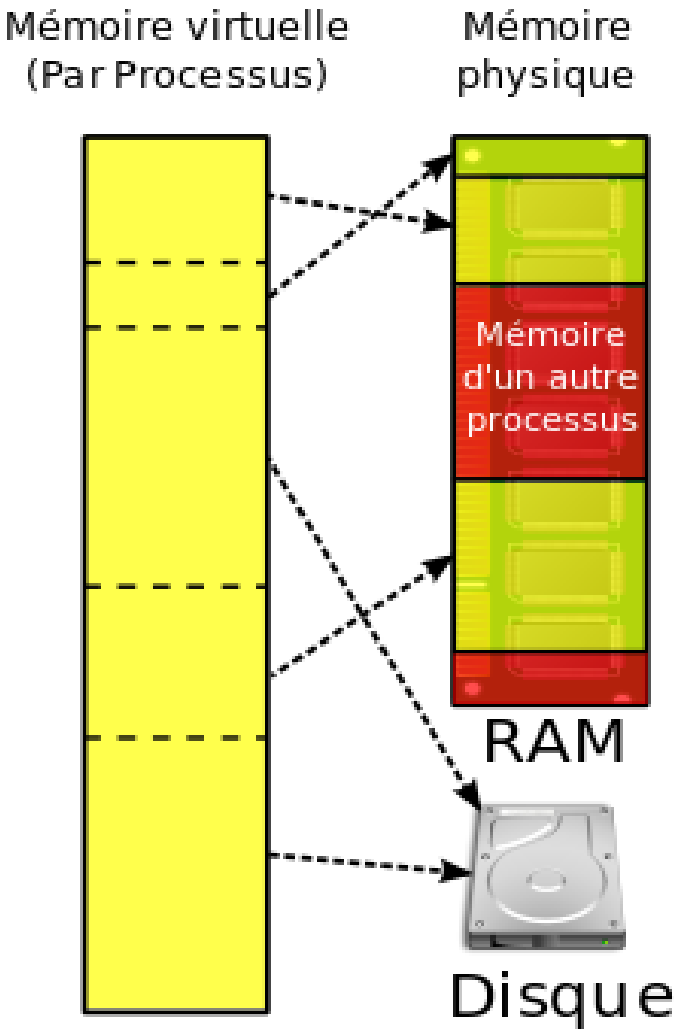
Objectifs

- Avoir pour chaque processus une mémoire “linéaire”
- Dépasser la capacité de mémoire physique disponible utilisée pour le programme, les données ou la pile
  - L’OS conserve les parties de programme en cours d’utilisation dans la mémoire principale, et le reste sur le disque
- Avoir une meilleure protection / contrôle / sécurité
  - Isolation de processus, ...
- Partager la mémoire entre différents processus

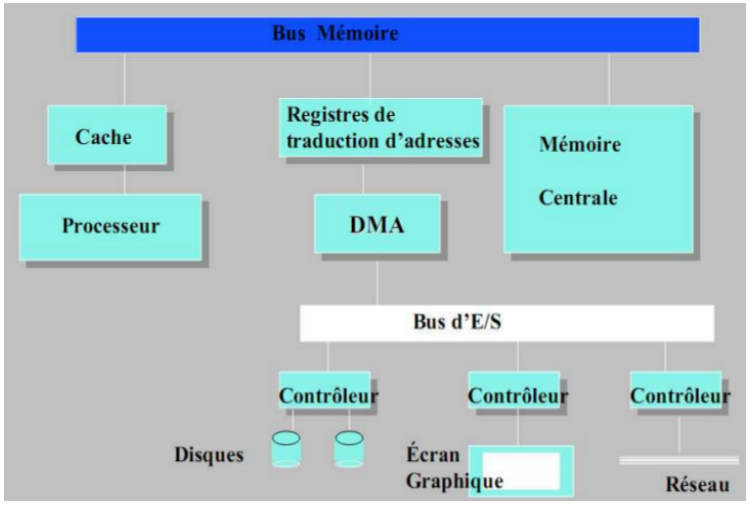
=> introduction d’adresses virtuelles

- Indépendant du stockage sous-jacent
- Les adresses virtuelles sont traduites au dernier moment par le MMU (memory management unit)
  - => gestion des tables de pages (plusieurs niveaux)

Mémoire virtuelle



L'accès direct à la mémoire (DMA / RMDA)



Notions importantes en particulier pour le HPC

- **DMA : Direct memory access**
  - Lecture et écriture directement en mémoire
  - Disponible uniquement s'il y a un contrôleur DMA
  - Le contrôleur DMA a accès au bus système sans dépendre du processeur
    → Allègement du travail du processeur
  - Réduit le nombre d'interruptions et permet un recouvrement total (le processeur pouvant faire autre chose)

Rem :

En cas de conflit d'accès à la mémoire, la priorité est donné au DMA

- **RDMA : Remote direct memory access**

Accès distant en lecture et écriture à la mémoire

=> utile pour MPI (Message passing Interface), système de fichiers parallèle

Les entrées / sorties (E/S)

Les entrées / sorties

L'OS a la tâche importante de contrôler les périphériques d'entrées/sorties (E/S)

- Fonctions
  - Émission des commandes vers les périphériques
  - Interception des interruptions
  - Gestion des erreurs
- But
  - Fournir une interface simple entre les périphériques et le système
  - Interface identique pour tous les périphériques

Les entrées / sorties : les unités

- Deux catégories :
  - **périphériques par bloc** : informations stockées par bloc de taille fixe, chacun possédant sa propre adresse (e.g. disque)
  - **périphériques par caractère** : l'information circule sous la forme d'un flot de caractères, sans aucune structure de bloc (e.g. : clavier, imprimante, souris)
- Deux parties dans une unité :
  - un composant mécanique, le périphérique (e.g. : disque)
  - un composant électronique, le contrôleur de périphérique (e.g. : contrôleur IDE)

Les entrées / sorties : la communication

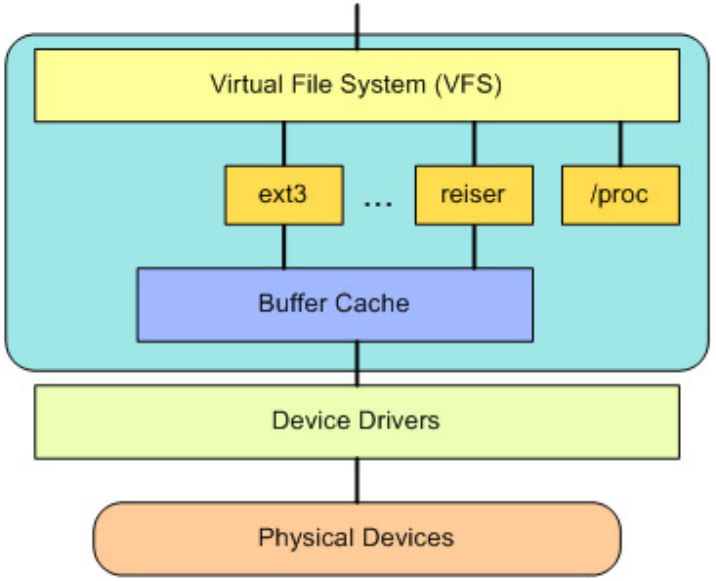
- Interface entre contrôleur et périphérique de très bas niveau
- Le contrôleur possède des registres qui permettent la communication avec le processeur
  - Écriture dans ces registres : l'OS ordonne au périphérique de délivrer des données, d'en accepter ou d'effectuer une action donnée
  - Lecture dans ces registres : l'OS peut connaître l'état du périphérique, savoir s'il est capable d'accepter une nouvelle commande
- Certains périphériques sont équipés d'un tampon de données que l'OS peut lire ou écrire

Le système de fichiers

Le système de fichiers

- Enregistrement d'une grande quantité d'informations
- Informations conservées après la fin du processus qui les utilise (persistance)
- Plusieurs processus doivent pouvoir avoir accès simultanément à une information
- Fichier : Mécanisme d'abstraction (l'utilisateur ne voit pas où et comment sont stockées les informations)
- Répertoire ou catalogue : Système à répertoires hiérarchiques
  - permet regroupement logique des fichiers
  - notion de chemin d'accès
    - chemin d'accès absolu (depuis la racine)
    - chemin d'accès relatif (depuis le répertoire courant)

=> HPC : nécessité de système de fichiers parallèle



Le démarrage d’un système

Le démarrage d'un système : BIOS

- Début du démarrage
  - Tout est vierge : mémoire vide, pas d'instruction dans les processeurs

- Puis le processeur 0 (bootstrap processor) démarra. Il est programmé pour exécuter le contenu d'une adresse spéciale

- Et il exécute le BIOS : **Basic Input/Output System**
  (ou UEFI (Unified Extensible Plaform Interface))

  - Stocké sur un circuit 1 de la carte mère
  - Configurable par un menu lors du démarrage du système.
  - Détermine ou chercher le système à démarrer (quel disque, lecteur CD ou DVD, clef USB, disquette, par le réseau, . . .)

- Chaque disque contient une zone particulière qui est destinée au démarrage du système (MBR : Marter Boot Record).
  => Lancement du binaire qui se trouve dans cette zone du disque indiqué par le BIOS.

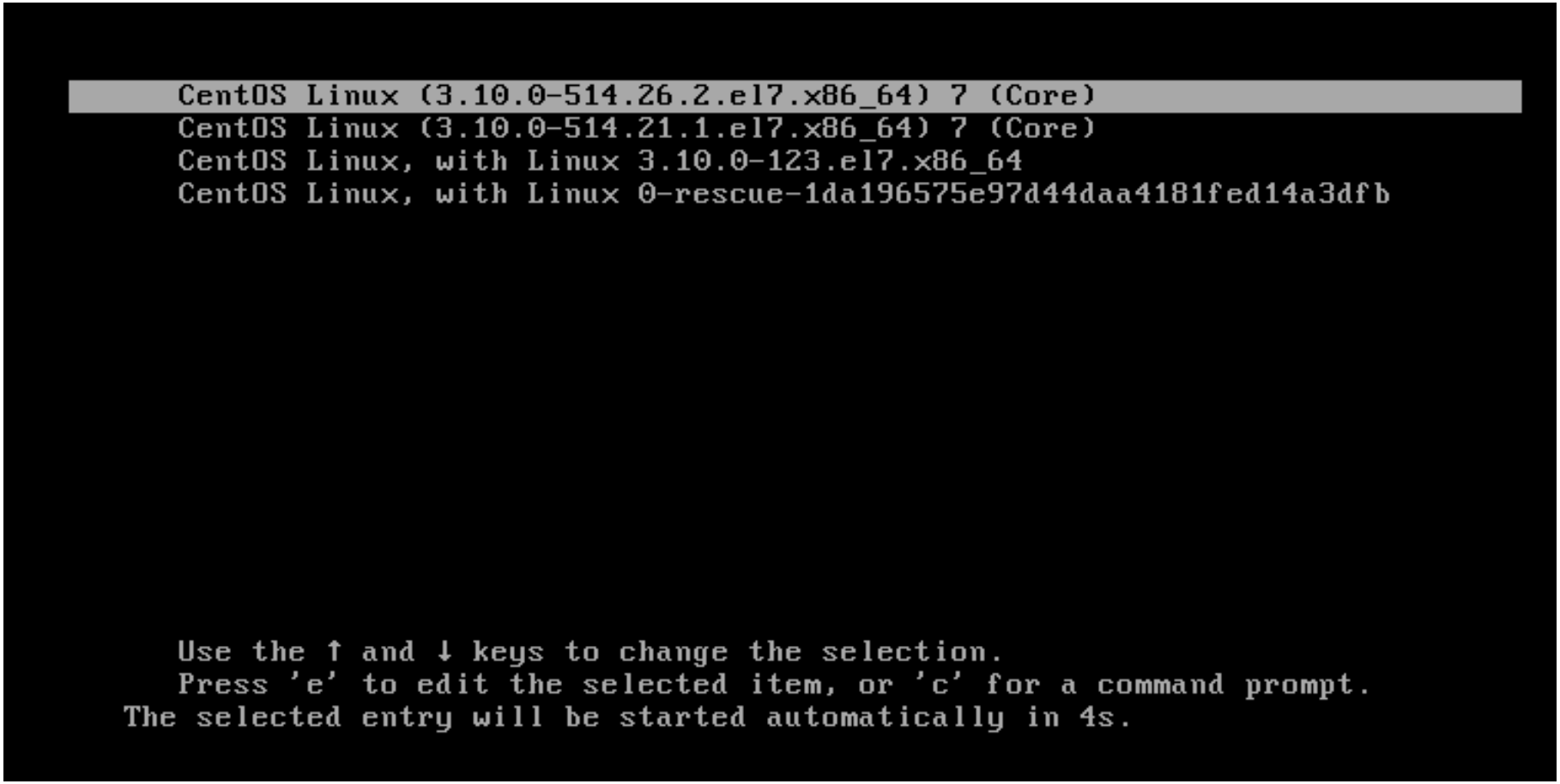
Le démarrage d'un système : le chargeur ou Boot loader

Puis 2 possibilités :

- Lancement du système propre
- Lancement d'un chargeur (bootloader)
  Il est utilisé souvent dans le cas de systèmes multi-boot

- Le chargeur :
  - Logiciel qui réside sur un support comme disque dur, clef USB, ...
  - Peut être modifié ou configuré par l'utilisateur, l'installation de ce logiciel nécessite des outils spécialisés car le chargeur réside sur une zone spéciale du disque.
  - Ce logiciel détermine (éventuellement après un choix fait par l'utilisateur) quel système d'exploitation lancer.
  - Le chargeur connaît une liste de système d'exploitation installés (par exemple Windows, Linux, BSD), avec le disque et l'endroit sur le disque où se trouve le noyau.
  - Exemples de chargeur (GNU/Linux) : LILO, Grub

Le démarrage d'un système : le chargeur GRUB



Le démarrage du système : GNU/Linux

- Le chargeur lance le noyau :
  - Détection des ressources grâce au BIOS
  - Réservation des zones mémoire pour la gestion des ressources du système
  - Le noyau parcourt les arguments transmis par le bootloader (ex : GRUB)
  - Initialise / configure la mémoire, les interruptions et un minimum de périphériques
  - Création d'un / temporaire en mémoire
  - Monte l' « init RAM disk »
    - charge des modules / drivers
    - initialise des périphériques
  - Monte le vrai système de fichiers / (procédure du pivot_root)
  - Lancement d'un premier processus init qui sert comme racine de l'arbre de processus.

Remarque :

Ce premier processus est en mode **userspace**

=> le noyau est alors pleinement opérationnel

Le démarrage du système : GNU/Linux

- Poursuite du lancement complet du système d'exploitation (au travers du processus Init)
  - Initialisation des périphériques restants
  - Montage des autres systèmes de fichiers
  - Lancement des différents services :
    - init crée (par exemple) des processus qui attendent qu'un utilisateur entre son login et mot de passe, puis (quand authentification de l'utilisateur réussie) se transforment dans un interpréteur de commande avec les privilèges de l'utilisateur.

Remarque :

Init sera aussi en charge de l'exécution des process nécessaires à l'arrêt de la machine.

Le démarrage d'un système


```
[    0.432921] NetLabel:  domain hash size = 128
[    0.433350] NetLabel:  protocols = UNLABELED CIPSOv4
[    0.433795] NetLabel:  unlabeled traffic allowed by default
[    0.434339] Switched to clocksource kvm-clock
[    0.439373] pnp: PnP ACPI init
[    0.439821] ACPI: bus type PNP registered
[    0.440579] pnp: PnP ACPI: found 5 devices
[    0.441004] ACPI: bus type PNP unregistered
[    0.446633] NET: Registered protocol family 2
[    0.447203] TCP established hash table entries: 8192 (order: 4, 65536 bytes)
[    0.447730] TCP bind hash table entries: 8192 (order: 5, 131072 bytes)
[    0.448208] TCP: Hash tables configured (established 8192 bind 8192)
[    0.448678] TCP: reno registered
[    0.449092] UDP hash table entries: 512 (order: 2, 16384 bytes)
[    0.449573] UDP-Lite hash table entries: 512 (order: 2, 16384 bytes)
[    0.450054] NET: Registered protocol family 1
[    0.450510] pci 0000:00:00.0: Limiting direct PCI/PCI transfers
[    0.450996] pci 0000:00:01.0: PIIX3: Enabling Passive Release
[    0.451537] pci 0000:00:01.0: Activating ISA DMA hang workarounds
[    0.471334] ACPI: PCI Interrupt Link [LNKB] enabled at IRQ 10
[    0.511326] ACPI: PCI Interrupt Link [LNKC] enabled at IRQ 11
[    0.553157] ACPI: PCI Interrupt Link [LNKD] enabled at IRQ 11
[    0.593728] ACPI: PCI Interrupt Link [LNKA] enabled at IRQ 10
[    0.613406] Unpacking initramfs...
```

Le démarrage d'un système : pivot_root

```
         Starting Reload Configuration from the Real Root...
[  OK  ] Started Reload Configuration from the Real Root.
[  OK  ] Reached target Initrd File Systems.
[  OK  ] Reached target Initrd Default Target.
         Starting dracut pre-pivot and cleanup hook...
[  OK  ] Started dracut pre-pivot and cleanup hook.
         Starting Cleaning Up and Shutting Down Daemons...
         Starting Plymouth switch root service...
[  OK  ] Stopped target Timers.
[  OK  ] Stopped Cleaning Up and Shutting Down Daemons.
[  OK  ] Stopped dracut pre-pivot and cleanup hook.
         Stopping dracut pre-pivot and cleanup hook...
[  OK  ] Stopped target Remote File Systems.
[  OK  ] Stopped target Remote File Systems (Pre).
[  OK  ] Stopped dracut initqueue hook.
         Stopping dracut initqueue hook...
[  OK  ] Stopped target Initrd Default Target.
[  OK  ] Stopped target Basic System.
[  OK  ] Stopped target Sockets.
[  OK  ] Stopped target Paths.
[  OK  ] Stopped target System Initialization.
[  OK  ] Stopped target Swap.
[  OK  ] Stopped target Local File Systems.
[  OK  ] Stopped Apply Kernel Variables.
         Stopping Apply Kernel Variables...
         Stopping udev Kernel Device Manager...
[  OK  ] Stopped udev Coldplug all Devices.
         Stopping udev Coldplug all Devices...
[  OK  ] Stopped target Slices.
[  OK  ] Stopped udev Kernel Device Manager.
[  OK  ] Stopped dracut pre-udev hook.
         Stopping dracut pre-udev hook...
[  OK  ] Stopped dracut cmdline hook.
         Stopping dracut cmdline hook...
[  OK  ] Stopped Create Static Device Nodes in /dev.
         Stopping Create Static Device Nodes in /dev...
[  OK  ] Stopped Create list of required static device nodes for the current kernel.
         Stopping Create list of required static device nodes for the current kernel...
[  OK  ] Closed udev Kernel Socket.
[  OK  ] Closed udev Control Socket.
         Starting Cleanup udevd DB...
[  OK  ] Started Cleanup udevd DB.
[  OK  ] Reached target Switch Root.
[  OK  ] Started Plymouth switch root service.
         Starting Switch Root...
[    2.545337] systemd-journald[87]: Received SIGTERM from PID 1 (systemd).
[    2.592998] type=1404 audit(1504473129.895:2): enforcing=1 old_enforcing=0 auid=4294967295 ses=4294967295
```

Le démarrage d'un système : lancement des services

```
[  OK  ] Listening on udev Kernel Socket.
[    1.313227] systemd[1]: Listening on udev Kernel Socket.
[    1.317415] systemd[1]: Starting udev Kernel Socket.
[  OK  ] Reached target Sockets.
[    1.357404] systemd[1]: Reached target Sockets.
[    1.357862] systemd[1]: Starting Sockets.
[  OK  ] Reached target Slices.
[    1.360040] systemd[1]: Reached target Slices.
[    1.362365] systemd[1]: Starting Slices.
[    1.363192] systemd[1]: Starting dracut cmdline hook...
         Starting dracut cmdline hook...
[  OK  ] Reached target Local File Systems.
[    1.367896] systemd[1]: Reached target Local File Systems.
[    1.371595] systemd[1]: Starting Local File Systems.
[  OK  ] Started Journal Service.
[    1.373867] systemd[1]: Started Journal Service.
[  OK  ] Started Create list of required sta...ce nodes for the current kernel.
[  OK  ] Started Apply Kernel Variables.
         Starting Create Static Device Nodes in /dev...
[  OK  ] Started Create Static Device Nodes in /dev.
[    1.416346] usb 1-1: new high-speed USB device number 2 using ehci-pci
[  OK  ] Started Setup Virtual Console.
[  OK  ] Started dracut cmdline hook.
         Starting dracut pre-udev hook...

_
```

Le démarrage d'un système : lancement des services

```
Welcome to CentOS Linux 7 (Core)!

[  OK  ] Stopped Switch Root.
[  OK  ] Stopped Journal Service.
         Starting Journal Service...
         Mounting NFSD configuration filesystem...
[  OK  ] Reached target RPC Port Mapper.
[  OK  ] Set up automount Arbitrary Executable File Formats File System Automount Point.
[  OK  ] Stopped target Switch Root.
[  OK  ] Listening on Delayed Shutdown Socket.
[  OK  ] Listening on LVM2 poll daemon socket.
[  OK  ] Created slice User and Session Slice.
[  OK  ] Stopped target Initrd Root File System.
[  OK  ] Created slice system-selinux\x2dpolicy\x2dmigrate\x2dlocal\x2dchanges.slice.
[  OK  ] Listening on LVM2 metadata daemon socket.
         Starting Create list of required static device nodes for the current kernel...
[  OK  ] Listening on udev Kernel Socket.
[  OK  ] Listening on /dev/initctl Compatibility Named Pipe.
         Mounting POSIX Message Queue File System...
[  OK  ] Reached target Login Prompts.
[  OK  ] Stopped target Initrd File Systems.
         Starting Remount Root and Kernel File Systems...
[  OK  ] Created slice system-systemd\x2dfsck.slice.
         Starting Apply Kernel Variables...
[  OK  ] Reached target Slices.
         Mounting Debug File System...
[  OK  ] Created slice system-getty.slice.
[  OK  ] Listening on udev Control Socket.
[  OK  ] Reached target Host and Network Name Lookups.
[  OK  ] Reached target User and Group Name Lookups.
[  OK  ] Listening on Device-mapper event daemon FIFOs.
[    3.425086] RPC: Registered named UNIX socket transport module.
[    3.425087] RPC: Registered udp transport module.
[    3.425087] RPC: Registered tcp transport module.
[    3.425088] RPC: Registered tcp NFSv4.1 backchannel transport module.
         Starting Monitoring of LVM2 mirrors, snapshots etc. using dmeventd or progress polling...
         Mounting Huge Pages File System...
         Starting Kernel Module supporting RPCSEC_GSS...
[  OK  ] Mounted Debug File System.
[  OK  ] Mounted POSIX Message Queue File System.
[  OK  ] Mounted Huge Pages File System.
```

Conclusion

L’évolution des systèmes d’exploitation

- **Micro-kernel :**
  - Factoriser les tâches d’un kernel en un kernel minimal.
  - Simplifier / spécialiser le kernel

    => optimisations plus faciles à faire, kernel mieux adapté

  Rem :

  Couplage possible entre un kernel standard et les micro-kernel

- **Virtualisation :**

  Faire tourner un système complet sur une simulation logicielle d’une machine (machine virtuelle). C’est également une veille idée (OS/360 des années 60) qui est aujourd’hui couramment utilisée.

  => très utile pour cloisonner des applications, tester / débugger de nouveaux OS, ...

Conclusion

- Ce cours est une introduction au fonctionnement générale d'un système d'exploitation mettant en avant les besoins pour le calcul haute performance (HPC)

- Système d'exploitation :
  - Description
  - Fonctionnement global
  - Première approche des composants critiques :
    - Processus
    - Mémoire
    - E/S
    - Système de fichiers

- Les séances suivantes permettront de rentrer beaucoup plus en détail dans le fonctionnement des différents composants du système d'exploitation Linux.

Approfondissement des composants d'un OS

Dans la suite du cours :

- Approfondissent des composants suivants :
  - Allocateur mémoire
  - Ordonnanceur de tâches
  - Systèmes de fichiers locaux
- La sécurité :
  Description du rôle et des éléments du système participant à la Sécurité du système
- Analyse de crash et debbuging du noyau
- Optimisations du système d'exploitation

Questions ?

Backup

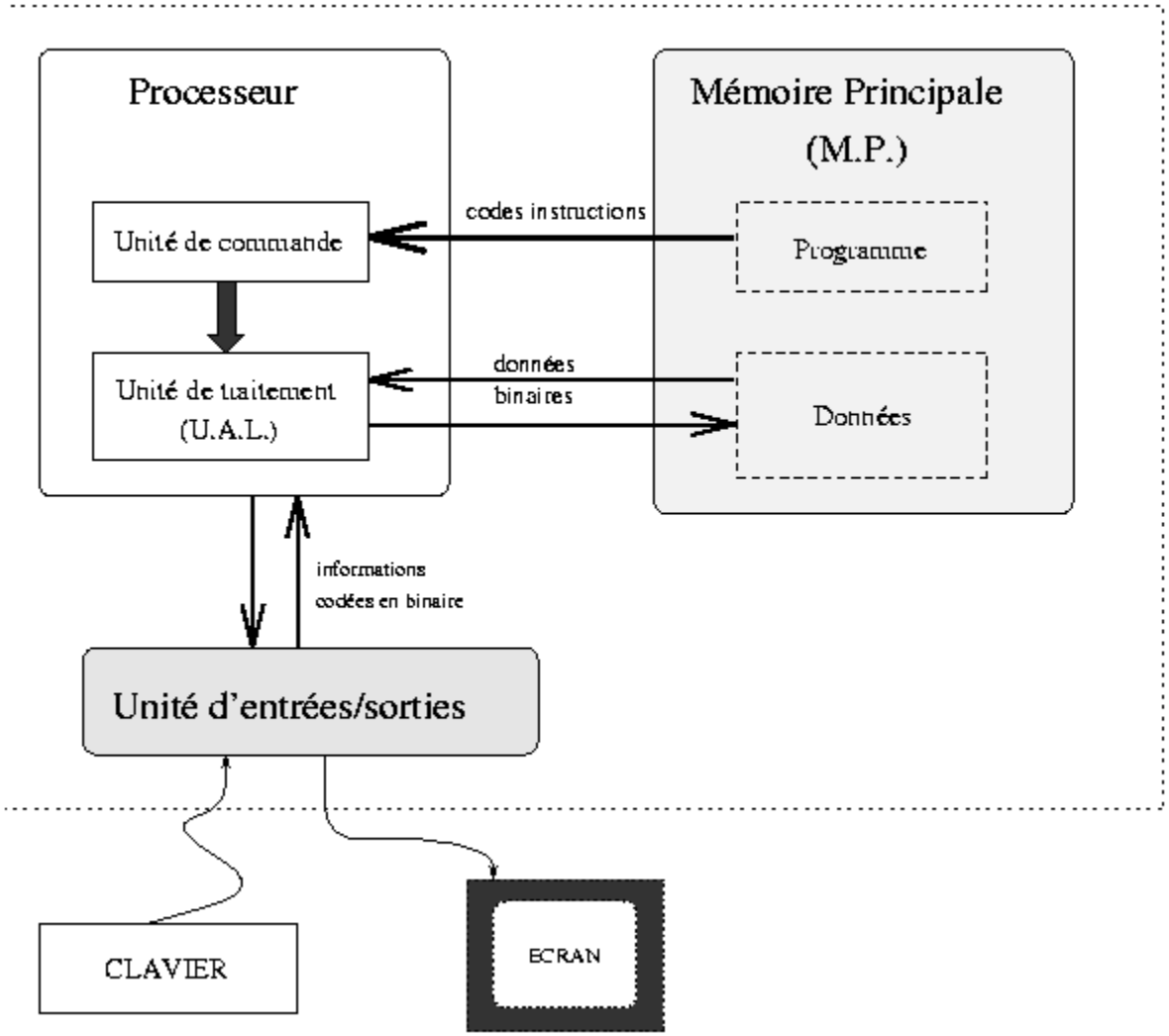
Rappel :
Fonctionnement d’un ordinateur

Rappel : fonctionnement d'un ordinateur

La machine de von Neumann : EDVAC (Electronic Discrete Variable Automatic Computer) est composée de :

- une mémoire principale (MP) qui contient données et instructions
- une unité arithmétique et logique (UAL) capable de fonctionner sur des données binaires
- une unité de contrôle (UC) qui interprète les instructions en mémoire et en entraîne l'exécution
- un dispositif d'entrée et de sortie (E/S) pris en charge par l'unité de contrôle

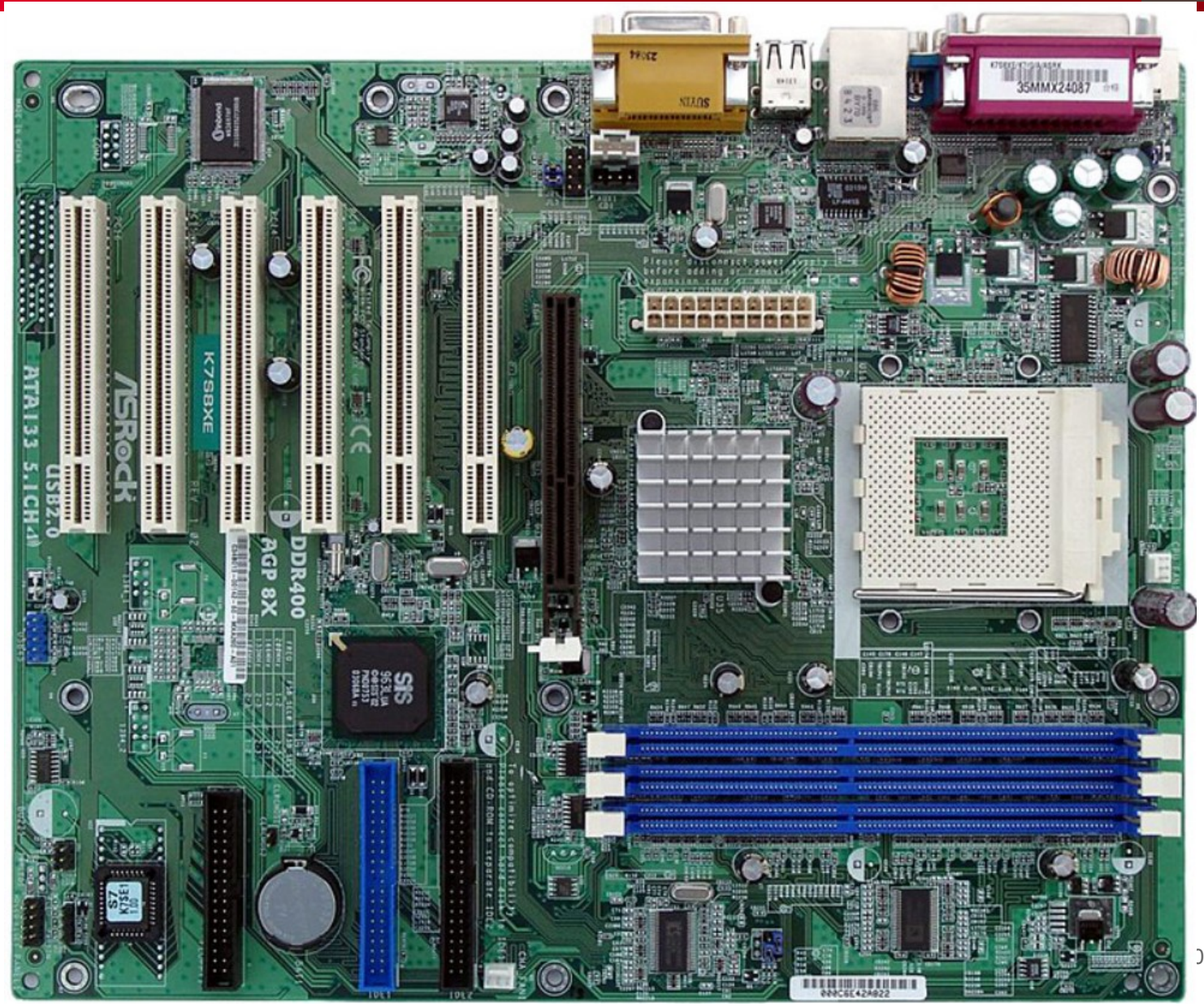
Rappel : fonctionnement d'un ordinateur



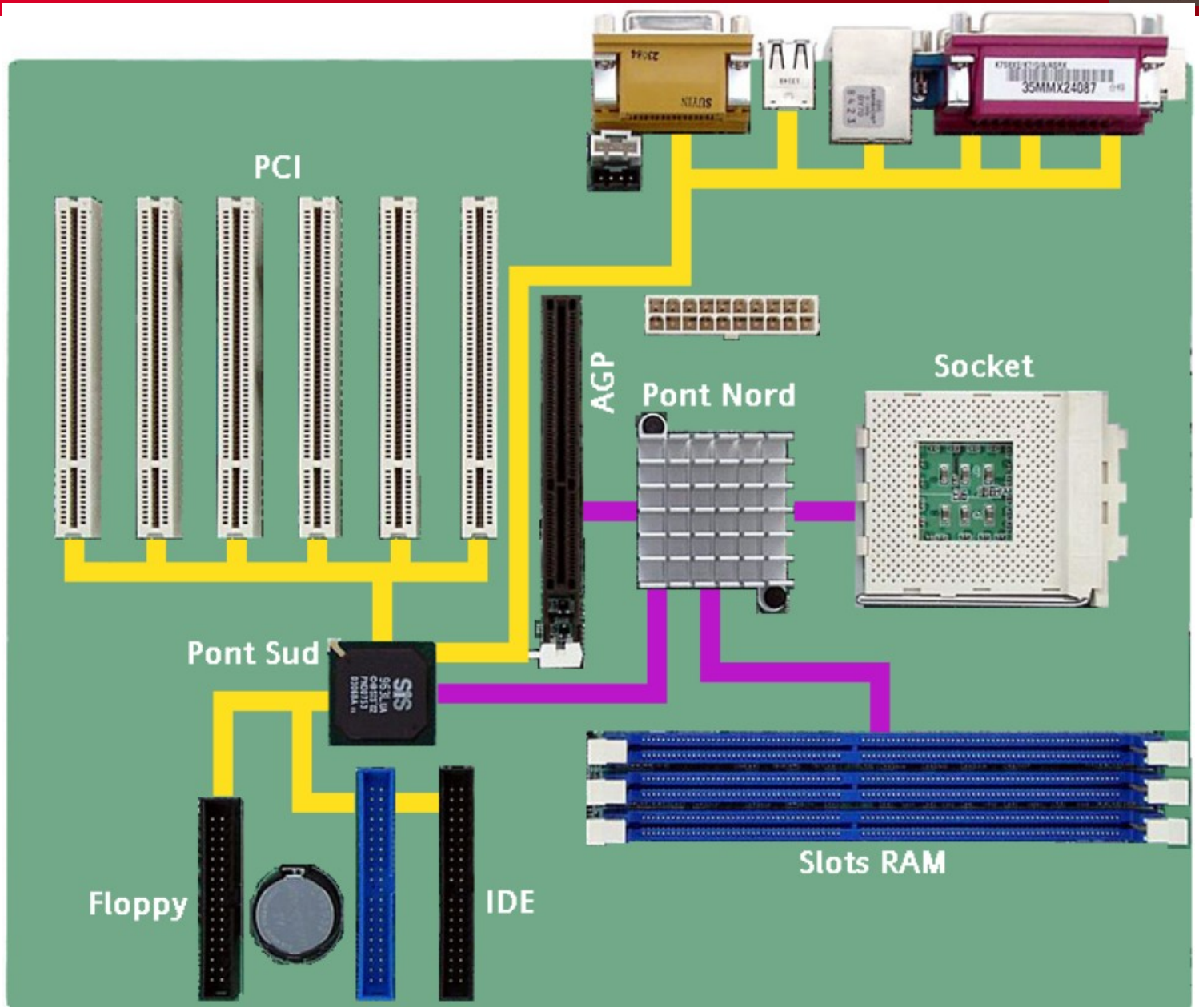
Rappel : fonctionnement d'un ordinateur

- Registres : mémoire haute vitesse qui se trouve dans le processeur
- Processeur :
  réunion de l'UAL (unité de calcul), de l'UC (unité de commandes) et de registres
- Disque magnétique :
  plateau circulaire à face simple ou double équipé d'une surface magnétisable capable de stocker les données
- Unité centrale :
  boîte (tour par exemple) qui contient le processeur, la mémoire principale et le disque
- PC : réunion de l'unité centrale, du clavier et de la souris
- Instruction : traitement effectué à un instant donné par le système (action atomique)
- Programme :
  suite d'instructions effectuant un certain traitement (le nombre de programmes possibles est théoriquement infini)

Rappel : fonctionnement d'un ordinateur



Rappel : fonctionnement d'un ordinateur



Commissariat à l'énergie atomique et aux énergies alternatives
Centre de Saclay | 91191 Gif-sur-Yvette Cedex
T. +33 (0)1 69 26 40 00 | F. +33 (0)1 69 26 XX XX

DAM
DSSI
SISR

Etablissement public à caractère industriel et commercial | RCS Paris B 775 685 019