Théorie des Systèmes d'exploitation : Introduction

P. Trégouët – A. Schaal

Groupe **esaip**

Quelques systèmes d'exploitation

- MsDos
- Windows 3.11 (multitâche coopératif)
- Windows 95 / 98 / millenium
- Windows NT4
- Windows NT5 ou NT2000
- Windows XP

- MacOS 9
- MacOS X (dérivé de Free BSD)

- Unix system V
- SCO Unix Open Server (Caldera)
- Soaris (ex Sun OS)
- HP-UX
- Xenix(Microsoft et SCO)
- Linux (RedHat, Mandrake, SuSE, Debian, slackware...)
- Free BSD

- MVS (IBM)
- OS400 (IBM AS400)
- VMS (DEC: Digital Equipment Corporation)
- GCOS/8 (BULL)

- CPM86
- OS/2

- Vxworks
- OS/9
- QNX
- Windows CE
- Linux RT
- eCos (RedHat Linux)

Architecture des ordinateurs

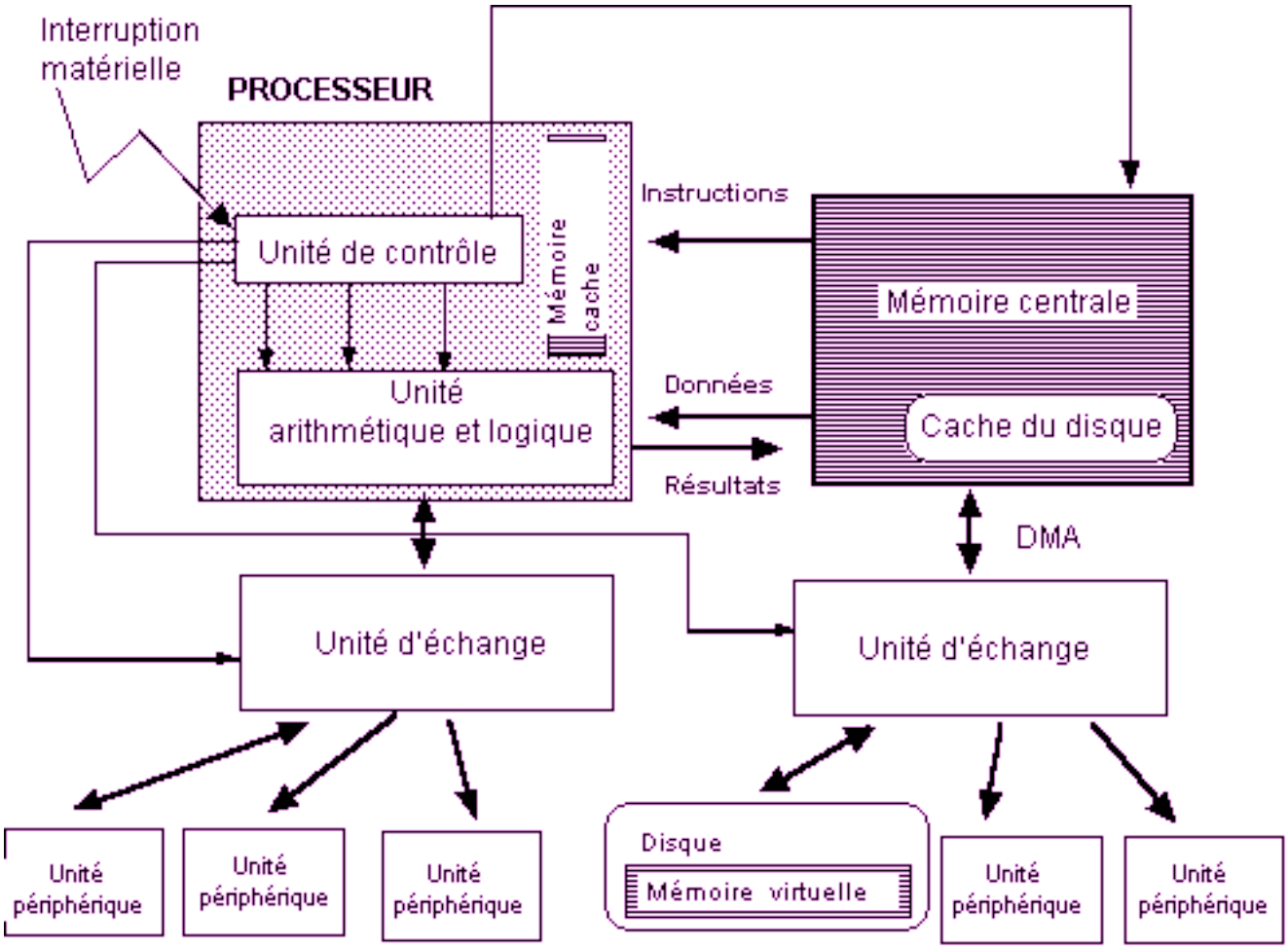
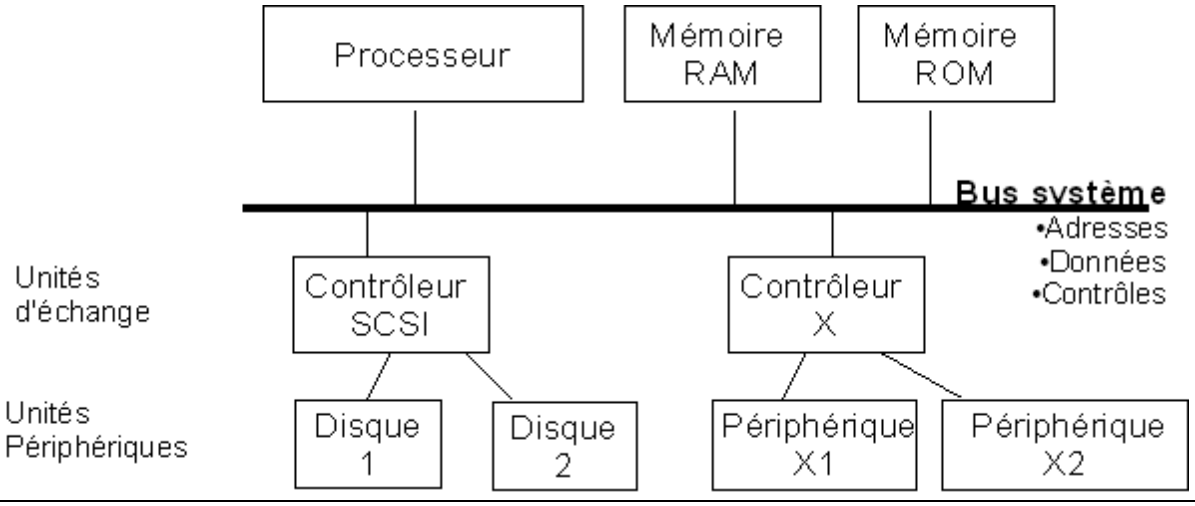
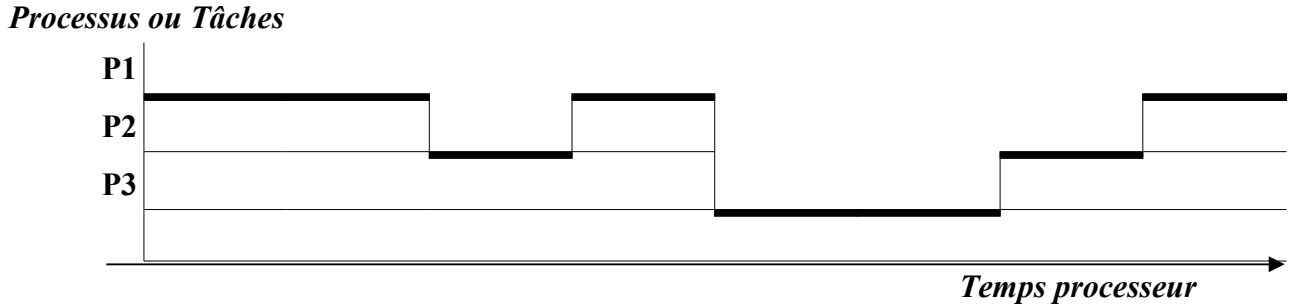


Schéma simplifié :



Les concepts des systèmes d'exploitation

Un des concepts principaux des systèmes d'exploitation est la notion de processus: un processus est un programme en cours d'exécution. Un système d'exploitation sera dit multitâche s'il permet à plusieurs processus de s'exécuter de manière concurrente (partage du temps processeur dans une apparente simultanéité), ce que l'on peut représenter sous la forme d'un diagramme de Gantt:



Les 2 principales fonctions d'un systèmes d'exploitation

1.Normaliser l'accès aux ressources matérielles, en fournisssant des pilotes de périphériques, ou drivers, utilisables par toutes les applications. Ces pilotes permettront un accès standard et simplifié à l'écran, au clavier, à la souris, aux fichiers sur disque, aux ressources du réseau etc...

2.Partager les ressources de l'ordinateur entre les différents processus (les différents utilisateurs): ce partage implique à la fois

- une politique d'allocation de ressources, tenant éventuellement compte d'un niveau priorité alloué à un processus
- et un contrôle des droits d'accès attribués à un processus (ou à un utilisateur) sur les fichiers, la mémoire centrale, les autres processus etc...

Les «appels système », ou API système

Les API[1] (Application Progammer Interface) rendent accessibles au programmeur les opérations suivantes (liste non exhaustive)

- Contrôle des Processus
  - créer ou supprimer un processus
  - charger un programme en mémoire
  - démarrer, arrêter l'exécution d'un processus
  - lire ou modifier les attributs d'un processus
  - attendre jusqu'à l'expiration d'un délai
  - attendre un événement ou signaler l'arrivée d'un événement
  - allouer ou libérer de la mémoire

- Manipulation de fichiers
  - Créer ou supprimer un fichier
  - ouvrir ou fermer un fichier
  - préparer la lecture / écriture à partir d'un numéro d'octet donné
  - lire / écrire des données dans un ficher
  - lire ou modifier les attributs d'un fichier

- Manipulation de périphériques
  - réserver ou libérer l'accès à un périphérique
  - lire / écrire sur un périphérique
  - lire ou modifier les attributs d'un périphérique
  - monter / démonter un système de fichiers local

- Informations de maintenance du système
  - lire ou modifier la date ou l'heure du système
  - lire ou modifier les attributs d'un processus, d'un fichier ou d'un périphérique
  - lire ou modifier la date ou l'heure des données du système

- Communication/réseau
  - Créer / supprimer une connexion réseau
  - envoyer / recevoir un message
  - Envoyer / recevoir des informations sur l'état d'une connexion réseau
  - monter / démonter un système de fichiers réseau

[1] http://www.webopedia.com/

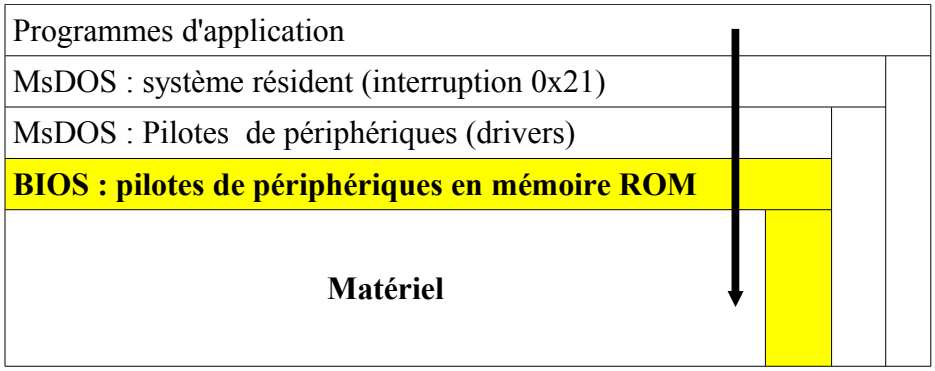
API : Abbreviation of ***a****pplication* ***p****rogram* ***i****nterface*, a set of routines, protocols, and tools for building software applications. A good API makes it easier to develop a program by providing all the building blocks. A programmer puts the blocks together.

Architecture en couches des systèmes d'exploitation

Schéma simplifié

Programmes d'application exemples : openoffice, word, firefox ...
Système d'exploitation exemples : Windows XP, Linux ...
HAL (Hardware Adaptation Layer) masquer aux systèmes d'exploitation les variations de l'architecture matérielle **BIOS**
Matériel processeur, mémoire contrôleurs de périphériques unités périphériques

Architecture de MSDOS



Architecture d'Unix

<table>
<tr><td colspan="3">Utilisateur</td></tr>
<tr><td colspan="3">Shells (interpréteurs de commandes)
Compilateurs et interpréteurs
Bibliothèques système</td></tr>
<tr><td colspan="3">Appels système : interface avec le noyau</td></tr>
<tr><td>•Signaux
•gestion des terminaux
•E/S en mode caractère
•Pilotes de terminaux</td><td>•Systèmes de fichiers
•Mémoire virtuelle
•E/S en mode bloc
•Pilotes de disques et bandes</td><td>•Ordonnancement des processus
•remplacement de pages (mémoire)
•défaut de pages
•mémoire viruelle</td></tr>
<tr><td colspan="3">Interface noyau avec le matériel</td></tr>
<tr><td>Controleur de terminaux</td><td>Controleurs de disques et bandes magnétiques</td><td>Contrôleur de mémoire</td></tr>
</table>

Pour approfondir : mode utilisateur ou mode noyau

Le système d'exploitation comprend le premier code exécutable chargé en mémoire lors du démarrage. Ce **code noyau** gère les différentes composantes de l'ordinateur. C'est par son entremise que vous pouvez charger un programme en mémoire et l'exécuter. Sans lui, rien ne peut être fait.

1. **Le code noyau possède les privilèges nécessaires pour contrôler le matériel.**
2. **Le noyau fournit des services aux programmes par l'intermédiaire des appels système.**

Un programme peut contenir des appels au noyau (ou au système). Les APIs définissent les différentes fonctions disponibles. Le noyau doit comporter des fonctions pour la gestion du CPU (processus), de la mémoire, des entrées/sorties et des fichiers.
Des éléments matériels permettent au noyau de contrôler l'allocation du CPU (l'exécution des processus) et d'assurer la protection mémoire. Un programme (processus) ne peut utiliser que la mémoire qui lui a été allouée.
Un processus est un programme chargé en mémoire et en train d'être exécuté. Le noyau crée et gère les processus.

En plus du **noyau**, un système d'exploitation comporte de nombreux programmes système.
Certains **processus système** sont créés dès l'amorçage du système et demeurent toujours actifs.
D'autres processus (programmes) assurent l'interaction avec les usagers. Un processus "shell" reçoit les commandes de l'utilisateur et permet leur exécution. Il peut au besoin créer un sous−processus (par un appel au noyau) pour exécuter une commande. Il y a des programmes système (commandes) pour gérer les fichiers: copier, détruire, renommer un fichier et afficher le contenu d'un répertoire.

Processus

Le noyau supporte la notion de processus. Pour chaque processus, le noyau garde une structure décrivant son état et les ressources utilisées (mémoire, fichiers, temps CPU). Le processeur (CPU) possède au moins deux modes de fonctionnement: **mode utilisateur et mode système ou noyau.**
En mode noyau, on peut exécuter des instructions privilégiées permettant le contrôle du matériel de protection mémoire, d'interruption et d'E/S. La protection empêche un processus utilisateur d'avoir accès à l'espace mémoire du noyau. **Le mécanisme d'appel au noyau permet à un processus utilisateur de transférer le contrôle du CPU au noyau (avec changement de mode) pour qu'il exécute une fonction nécessitant l'utilisation d'instructions privilégiées.**

Le noyau prend également le contrôle du CPU lors **d'interruption matérielle** pour exécuter une routine de traitement d'interruption. Lorsqu'il a terminé, le noyau détermine quel processus sera redémarré. Il utilise un algorithme d'allocation du CPU pour faire son choix. L'algorithme peut tenir

compte de la priorité, du temps d'utilisation ou d'autres critères.

Gestion de la mémoire

Le code chargé en mémoire (code exécutable d'un programme) contient des adresses relatives à l'espace logique du programme. Lors de l'exécution, il faut convertir ces adresses logiques en adresses physiques correspondant aux positions réelles occupées dans la mémoire physique. Du matériel dédié (le MMU) assure cette conversion. Le MMU assure également la protection mémoire**. Le noyau doit programmer le MMU pour lui indiquer quelle position physique en mémoire occupe le programme.**

Dans un système paginé, un programme n'occupe pas nécessairement des pages contiguës en mémoire. Il faut une table des pages pour faire la translation d'adresse logique à physique. Dans un système avec mémoire virtuelle, toutes les pages d'un programme n'ont pas besoin d'être en mémoire en même temps. Les pages du programme sont chargées en mémoire lorsque c'est nécessaire (sur demande).

Programmation concurrente

Lorsque plusieurs processus partagent des ressources, il faut synchroniser leurs accès aux ressources. Par exemple, lorsque deux processus partagent un espace mémoire, il faut éviter qu'ils essaient simultanément de modifier une même position (ou variable), sinon la valeur finale peut être erronée. "L'exclusion mutuelle" signifie qu'un seul processus à la fois pourra exécuter du code qui accède et modifie cette variable commune. La synchronisation de processus signifie que certains processus seront suspendus jusqu'à ce qu'ils obtiennent l'autorisation de continuer. Un sémaphore peut être vu comme un drapeau ou un feu de signalisation indiquant si l'autorisation de continuer est accordée.

Liens Internet :

http://fr.wikipedia.org/wiki/Noyau_%28informatique%29

http://fr.wikipedia.org/wiki/Espace_noyau