
Principes des Systèmes d'exploitation

Catherine Faron Zucker, UNSA, EPU SI1
faron@polytech.unice.fr

1

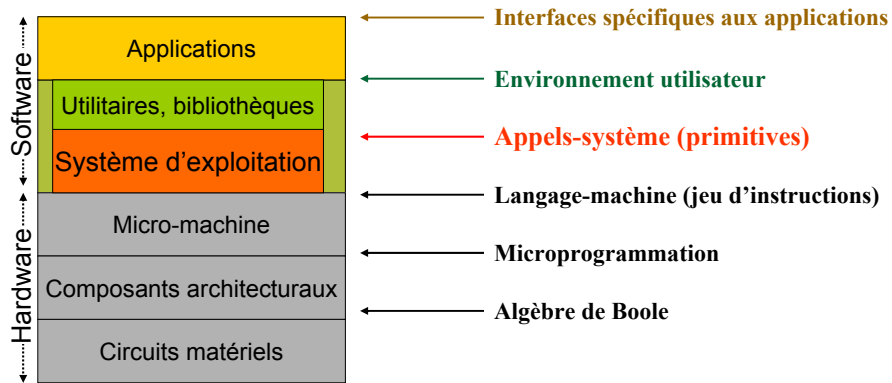
Introduction aux systèmes d'exploitation

Chapitre 1

2

Définition fonctionnelle d'un OS

■ Interface Utilisateur - Hardware



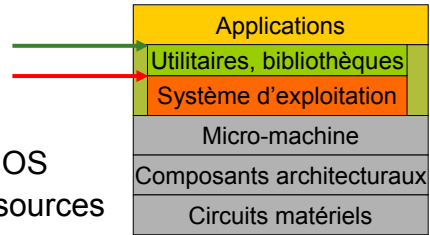
Définition fonctionnelle d'un OS

■ Contrôle, Gestion, ordonnancement des ressources

- Ressources physiques
 - Processeur(s), mémoire, disques, périphériques
- Programmes, processus
- Information
 - Stockage, désignation, localisation
 - Partage et échange entre usagers
- Protection, sécurité, confidentialité

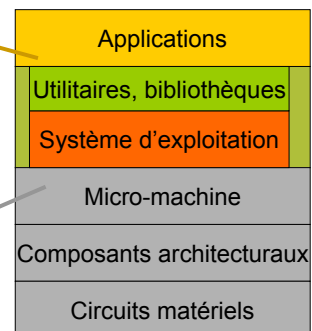
Appels système

- Interface entre applications et OS
Créer, détruire, utiliser les ressources gérées par l'OS
 - ❑ Créer des processus
 - ❑ Gérer la mémoire
 - ❑ Lire et écrire dans des fichiers
 - ❑ Gérer les entrées/sorties (lire au terminal, imprimer, etc.)
- Les procédures de bibliothèques appellent l'instruction TRAP pour démarrer l'OS qui exécute l'instruction correspondante
 - ❑ Par exemple la procédure C *read* invoque l'appel système READ
 - ❑ Les appels systèmes apparaissent comme des appels de procédures ordinaires



Classification des OS

- Par types d'application
 - ❑ Ordinateurs individuels
 - ❑ Systèmes embarqués
 - ❑ Systèmes « temps partagé »
 - ❑ Systèmes transactionnels
 - ❑ Systèmes graphiques, CAO
- Par types d'architecture de système
 - ❑ Système centralisé
 - ❑ Système multiprocesseur
 - ❑ Système réparti



Histoire des OS

- Fortes corrélations avec :
 - ❑ Les architectures des ordinateurs
 - ❑ Les avancées technologiques
 - ❑ Les langages de programmation
- Première génération (1945-1955)
 - ❑ Tubes à vide
 - ❑ Von Neumann
 - ❑ Gros calculateurs à usage militaire
 - ❑ Programmes en langage machine

Histoire des OS

- Seconde génération (1955-1965)
 - ❑ Transistors : les ordinateurs deviennent plus fiables
 - Nette séparation entre concepteurs, constructeurs, opérateurs, programmeurs, personnel de maintenance
 - Diffusion à 2 usages : calcul scientifique et commercial
 - ❑ 2 lignes distinctes de produits incompatibles
 - ❑ Machines incompatibles les unes avec les autres
 - ❑ Programmes en assembleur ou en fortran
 - ❑ Interaction par des cartes perforées et listings
 - ❑ Système de batch: traitement (séquentiel) par lot
 - ❑ Premiers OS: FMS (Fortran Monitor System), IBYS (IBM)

Histoire des OS

- Troisième génération (1965-1980)
 - Circuits intégrés
 - Augmentation de la performance et diminution du prix
 - IBM 360 : série unique de machines compatibles
 - Même architecture et même jeu d'instructions, le software écrit pour l'une peut s'exécuter sur les autres
 - Les modèles diffèrent selon la taille de la mémoire, la vitesse du processeur, les périphériques, etc.
 - Multitâches: conçues à la fois pour du calcul scientifique et des usages commerciaux

Histoire des OS

- Troisième génération (1965-1980)
 - OS/360 et ses successeurs
 - Enormes et très complexes : millions de lignes de code en langage assembleur
 - Multiprogrammation
 - Pour pallier aux temps d'attente en E/S
 - CTSS et MULTICS
 - Temps partagé

Histoire des OS

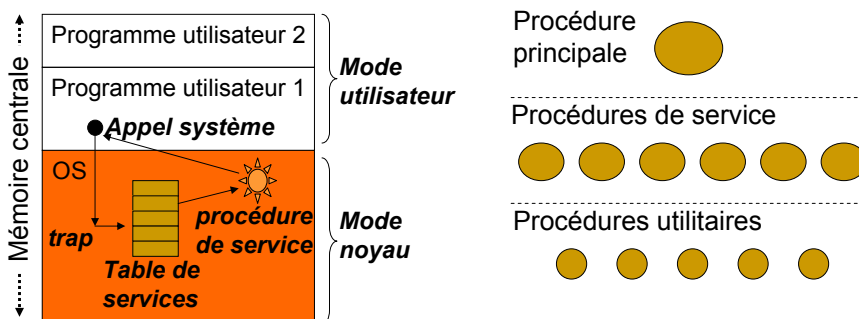
■ Quatrième génération (1980-)

- ❑ Circuits LSI
 - Stations de travail et ordinateurs personnels (prix abordable)
 - Nécessité d'interfaces utilisateur conviviales
- ❑ Langages de programmation de haut niveau
- ❑ MSDOS
 - dans les IBM PC et autres machines ayant une CPU Intel 8088 et ses successeurs
- ❑ UNIX
 - dans les stations et ordinateurs personnels à composants RISC
- ❑ Windows NT, MAC OS, Linux (multiplateforme)

Structures d'OS

■ Système monolithique

- ❑ OS est un seul module
- ❑ Séparation des espaces d'adressage système/utilisateur



Structures d'OS

■ Système en couches

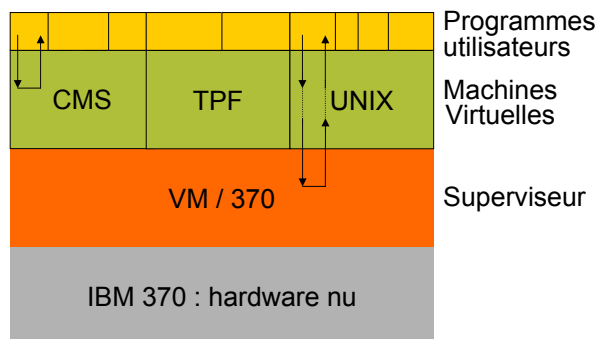
- Généralisation du modèle en 3 couches
- Chaque couche est construite sur celle inférieure
- THE, Dijkstra (1968)

5	Opérateur (utilisateur)
4	Programmes utilisateurs
3	Gestion d'E/S
2	Communication opérateur/processus
1	Gestion de la mémoire
0	Gestion de la multiprogrammation

Structures d'OS

■ Machine virtuelle

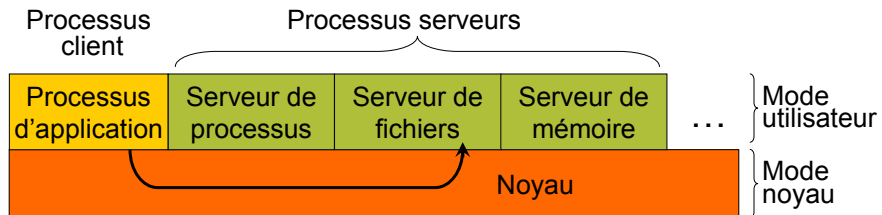
- VM / 370 (IBM)



Structures d'OS

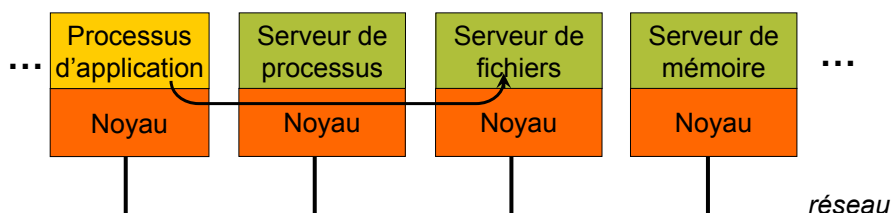
■ Modèle client-serveur, micro-noyau

- Windows NT, MACH, CHORUS



Structures de OS

■ Modèle client-serveur réparti



Principes de l'exécution d'un programme

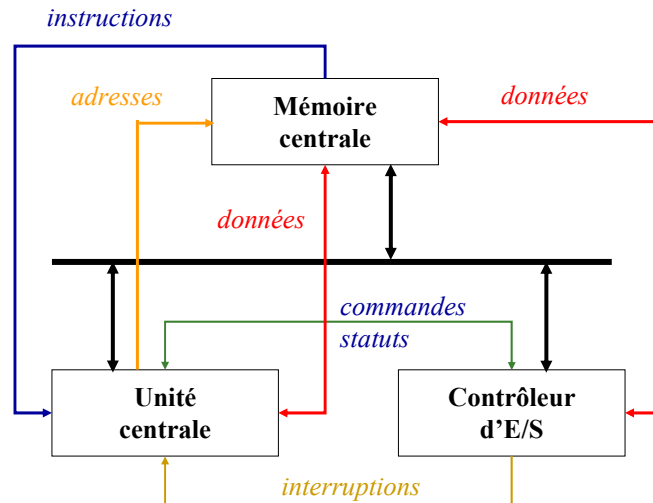
Chapitre 2

17

Modèle de von Neumann

- Architecture d'ordinateur qui utilise une mémoire unique pour stocker programme et données
 - Mémoire centrale : programme et données
 - CPU : registres de travail et opérateurs
 - Jeu d'instructions
 - Transferts, calculs, ruptures de séquence, *privilégiées*
 - Modes maître et esclave
 - Maître: le processeur accède à toutes les ressources
 - Esclave: certaines instructions sont interdites au processeur

Modèle de von Neumann



Processus

- Procédure et activité
 - Procédure : élément structurant d'un programme
 - Activité : exécution d'une procédure
- Programme et processus
 - Programme
 - Ensemble de procédures,
 - Résultat statique d'une compilation/édition de liens (binaire exécutable)
 - Processus
 - Activité dynamique résultant de l'exécution d'un programme
 - Enchaînement d'activités

Processus

■ Contexte

- Information caractérisant l'état courant d'un processus
 - Sauvegardé lors de la suspension du processus et rechargé lors de sa reprise
- Contexte matériel
 - Mot d'état, compteur ordinal, registres de l'unité centrale
- Contexte logiciel (ou système)
 - Segments de texte (instructions), de données, de pile...
 - Attributs : identification utilisateur, droits d'accès, priorité...
 - Ressources utilisées : fichiers ouverts...

■ Table des processus

- Maintenu par l'OS: Tableau de structures dédiées chacune à un processus existant et qui contient son contexte
- processus = adresse mémoire + entrée dans la table des processus

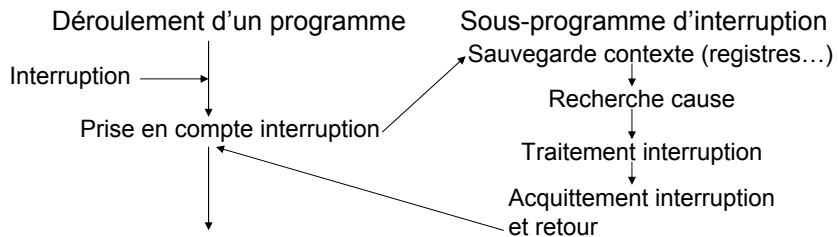
Processus

■ Espace d'adressage logique d'un processus

- Segment de données système *protégé*
 - Contexte matériel et logiciel
- Segment de texte *taille fixe*
 - Code, instructions
- Segment de pile *taille variable*
 - Variables locale, gestion des sous-programmes
- Segment de données *taille variable*

Interruptions

- Événement asynchrone (inopiné)
 - Fin d'entrée-sortie
- Rôle : interrompre le déroulement normal d'un programme
 - Problème: sauvegarde du contexte
- Procédure de traitement d'interruption



Cycle d'exécution d'une instruction

