Systèmes d'Exploitation et Programmation Concurrente

Dr. Faïza NAJJAR

Responsable du cours SE&PC des II2

ENSI – Dept. SRSI (Bureau 109)

Faiza.Najjar@ensi-uma.tn

Systèmes d'Exploitation et Programmation Concurrente

Cours intégré (semestre 3)

67,5 Heures (4H30/semaine)

Evaluation (ancienne!): 35% Contrôle Continu (CC) + 65% Examen

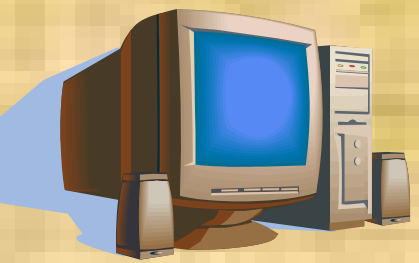
CC: 55%DS + 45% Note 2 (Participation + TPs)!

Motivations --Pourquoi étudier les SEs?

- **Domaine mâture**
  - Meilleurs programmes (corrects, performants, complexes, … etc)
  - Besoin de comprendre l'interaction entre logiciel et matériel
    - Tout utilisateur est concerné → Meilleure maîtrise
    - Tout programme est concerné → Améliorer l'efficacité
- Les serveurs de haute performance sont *confrontés aux mêmes problèmes* → afin de ne pas réinventer la roue
- **Challenges**
  - Programmation multithread (**multicore**)/Programmation Parallèle
  - Consommation de ressources (batterie, …)
  - Sécurité

Motivations –où sont les SEs?

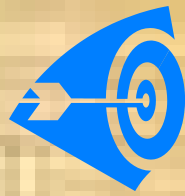
- Les SEs sont-ils seulement dans les ordinateurs??? →NON
  - Actuellement PARTOUT!!! Là où il y a un ordinateur il y a un SE!



Où sont les SEs? (suite)

Source: S. Helal, introduction to pervasive computing, Univ. Of Florida

Objectifs du cours



- Mieux comprendre les concepts et les mécanismes de base d'un SE multitâches/multithreads
  - ⇨ Techniques et algorithmes de gestion des ressources (processus, threads, mémoire centrale, …) …etc.
- Maîtriser les éléments de la **programmation système** (et non programmation **du** système!) → Focaliser sur Unix/Linux et le multithread devient nécessaire
  - ⇨ Fork, wait, exec, pipe, ….
  - ⇨ pthread_create, pthread_mutex, sem_wait, …
- ✪ **Cours vital** → Utiliser et/ou adapter les techniques et les services du système d'exploitation pour concevoir des codes plus fiables et plus performants.
- **Pré-requis:** pratique du langage C et shell Unix

Contenu du cours prévu

1) **Introduction aux systèmes d'exploitation**
   - ⇨ Principe de base d'un SE
   - ⇨ Structuration des SEs
   - ⇨ Bref historique
2) **Gestion des Processus et des Threads**
   - ⇨ Notion de ressource/processus/threads.
   - ⇨ Concurrence dans les SEs
   - ⇨ Commutation de contexte d'un processus
   - ⇨ Programmation système sous Unix (+Programmation multithreadée POSIX/java threads)
3) **Ordonnancement (Scheduling)**
   - ⇨ Notion de files d'attentes
   - ⇨ Politiques d'ordonnancement et comportement des processus/threads

Contenu du cours prévu (suite)

4) **Synchronisation et communication des processus/threads –IPC**
   - ⇨ Les mécanismes de synchronisation
   - ⇨ Verrous (mutex de Pthread), sémaphores, moniteurs, passage par messages, les signaux Unix, les tubes Unix.

5) **Gestion de la mémoire (physique + virtuelle)**
   - ⇨ Concepts fondamentaux; allocation statique/dynamique; politiques d'allocation

6) **Interblocage (deadlock)**
   - ⇨ Notion d'interblocage
   - ⇨ Solutions à l'interblocage (détection/guérison, prévention, évitement)

7) **Protection et sécurité**
   - ⇨ Protection : mécanismes de protection
   - ⇨ Sécurité : principes et authentification

Références Principales

Meilleurs livres les plus référencés (OS)

1. **A. Silberschatz**, & **P. Glavin** and G. Gagne. *Operating Systems Concepts*, 9th edition, John Wiley & sons, inc. 2012 (**ISBN-13:** 978-1118063330).
2. **A. Tanenbaum**, & Herbert Bos, *Modern Operating Systems*, 4th edition, Pearson, **ISBN-13:** 978-0133591620; (2014)

Autres cours utiles sur Internet (vieux)

3. Introduction aux systèmes et aux réseaux (**S. Krakowiak**, Grenoble)
   - ⇨ Recommandé: source de nombreux transparents présents ici
   - ⇨ *http://sardes.inrialpes.fr/~krakowia*
4. Operating Systems and System Programming (B. Pfa, Stanford)
   - ⇨ *http://cs140.stanford.edu/*

Sites d'information

- ⇨ *http://systeme.developpez.com/cours/*

Livre recommandé:.....NEA!!

Systèmes d'Exploitation & Programmation Concurrente

Chap.I. Introduction aux Systèmes d'Exploitation

Dr. Faïza NAJJAR

ENSI – Dept. SRSI (Bureau 109)

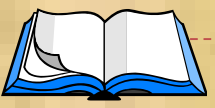
Faiza.Najjar@ensi-uma.tn

Contenu du cours

1. Définition d'un système d'exploitation "classique"
2. Concepts de base des systèmes d'exploitation
3. Structuration des systèmes d'exploitation
4. Bref historique

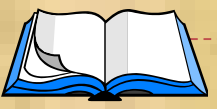
Terminologie Système

- **Système?**
  - Ensemble de composants qui interagissent;
  - Gestion des ressources communes et de l'infrastructure;
  - Lié de manière étroite au matériel (Hardware) sous-jacent.
- ***Exemples de systèmes :***
  - Système d'exploitation (le plus courant!): gestion de chaque élément.
  - Système de communication: échange d'information entre les éléments.
  - *Système de Gestion de Bases de Données* -- SGBD

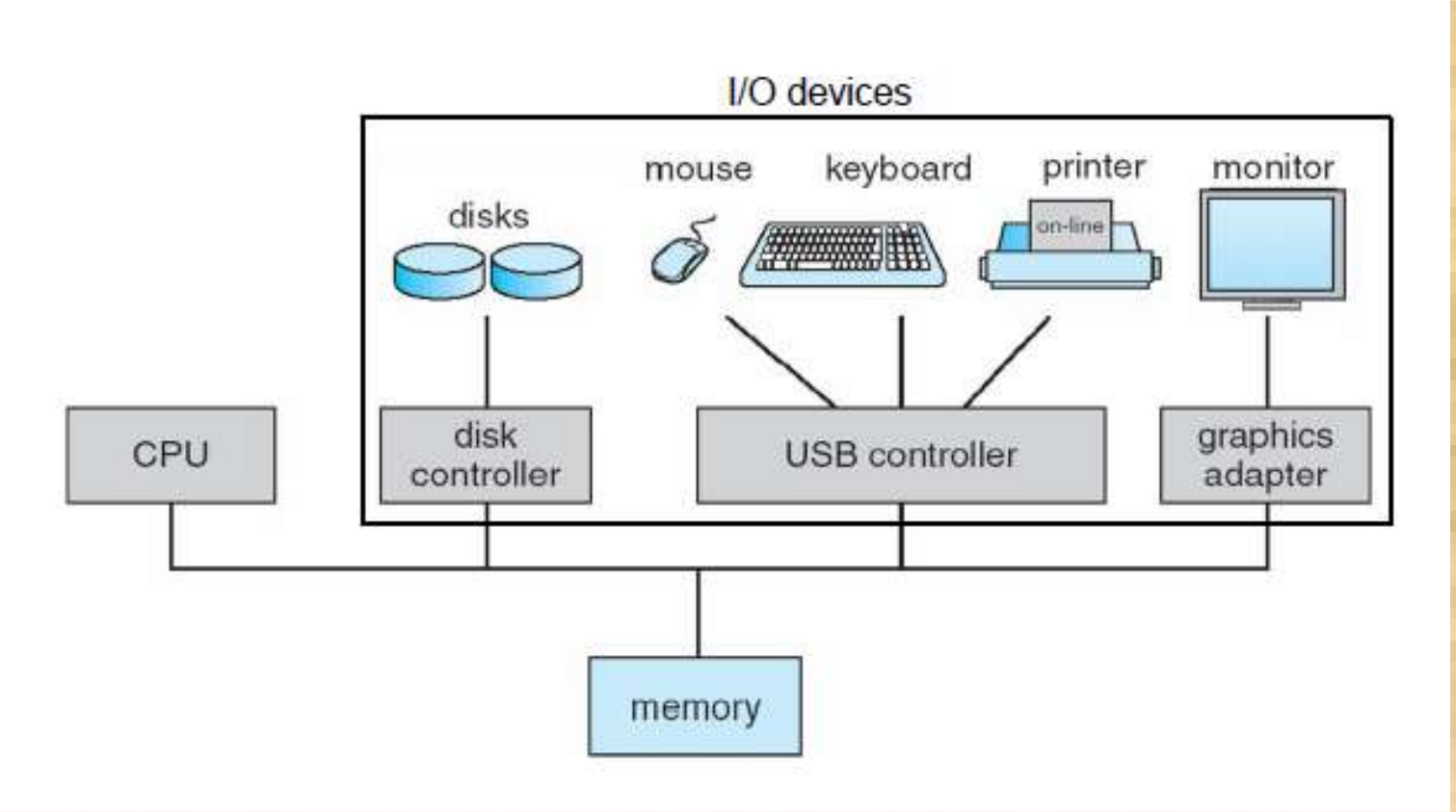


Systèmes vs. Applications?

- **Propriétés des systèmes:**
  - Les systèmes cachent la complexité du matériel et des communications
  - Ils fournissent des services communs de plus haut niveau d'abstraction.
- **Application?:** réponse à un problème spécifique, en utilisant les services généraux fournis par le système
- *La distinction n'est pas évidente!*
  - Exp. les systèmes embarqués

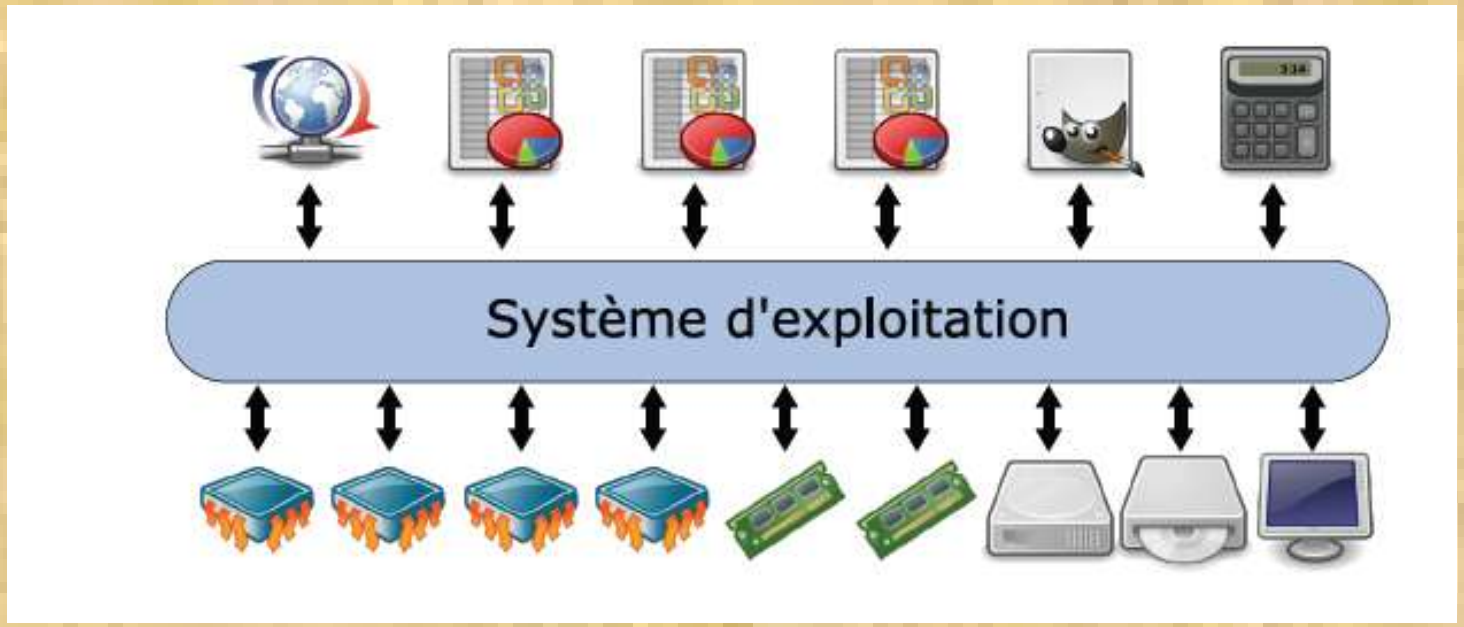


Organisation du Matériel



Qu'est-ce qu'un système d'exploitation?

- **Définition:** *Couche logicielle qui permet et coordonne l'utilisation du matériel entre les différentes applications et/ou utilisateurs (finaux/ordinateurs/autres)*



Source: R. Bonidal, systèmes d'exploitation, Loria France, 2010.

Rôle d'un système d'exploitation

- Deux aspects complémentaires [Krakowiak]:
  - **Adaptation d'interface**
    - *"Machine Virtuelle"* : plus facile d'emploi et plus conviviale
      - Cacher les détails de mise en œuvre du matériel
      - Cacher les limitations physiques (taille mémoire)
  - **Gestion des ressources (matérielles et logicielles)**
    - Utilisation cohérente, en bon ordre, des commandes matérielles
    - Protéger les autres programmes des erreurs de votre programme

Interfaces d'un système d'exploitation (1)

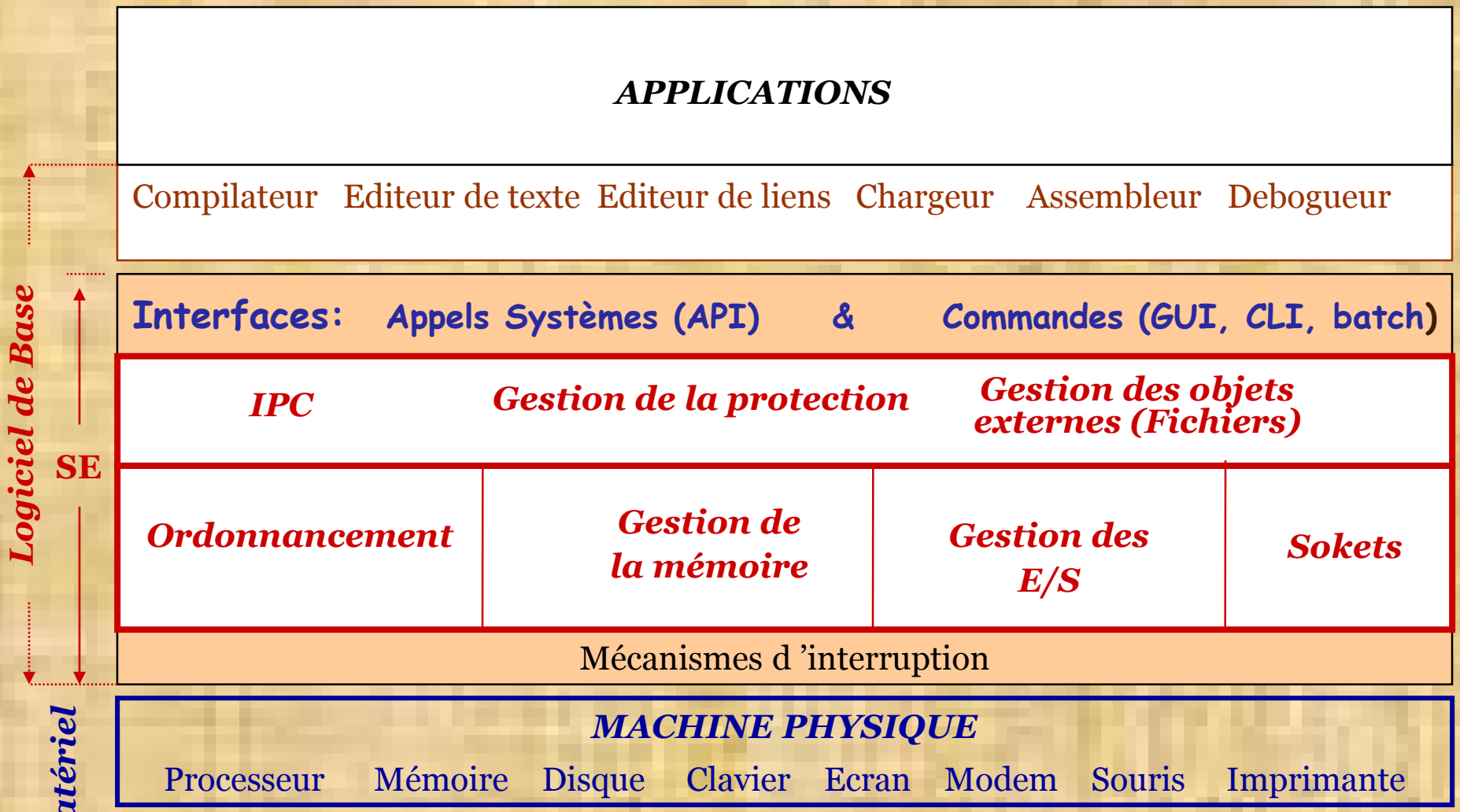
Notion d'interface (service)

- L'interface est l'ensemble des fonctions accessibles aux utilisateurs du service
  - ⇨ Chaque fonction est définie par son format (syntaxe), sa spécification (sémantique).
  - ⇨ Ces descriptions doivent être précises, complètes (y compris les cas d'erreur), non ambiguës.
- ***Une interface permet l'accès à un service***
- **Principe de base:**
  - ⇨ séparation entre ***interface*** et ***mode de réalisation***

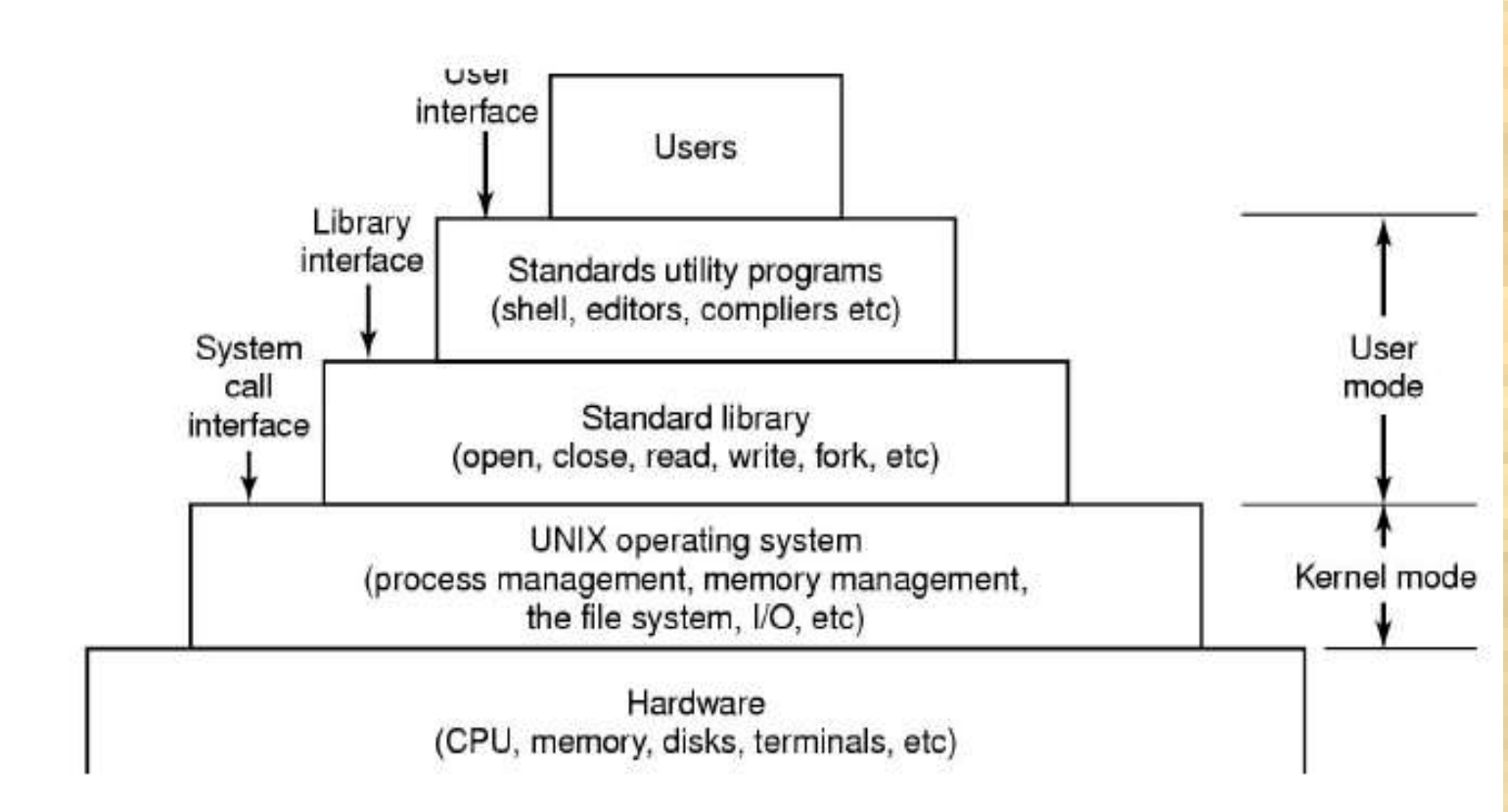
Interfaces d'un système d'exploitation (2)

- Au moins deux interfaces d'un SE:
  - ***Appels systèmes (System call)*:** *Interface programmatique,*
    - Fonctions fournies par le SE aux applications utilisateurs
    - *Généralement accédée par des **API** (Application programming interface)* afin de comprendre les réponses des appels systèmes.
    - Les 3 plus courantes : Win32 API, POSIX API et JAVA API
    - Exp. en C: *read (FD, &buffer, nbytes)*
  - ***Interface de commandes*** (textuelle **–CLI** ou graphique **—GUI**, ou batch)
    - CLI: *rm *.ps* *GUI: déplacer l'icône du fichier vers la corbeille*

Services principaux d'un SE Classique



Différents niveaux d'un Système



Source: [2]

Notions de Base d'un Système d'Exploitation

- Les fonctionnalités du SE sont accessibles par le biais de deux interfaces systèmes:
  - ***Commandes*** (GUI, CLI (shell Unix, invite DOS), batch), ou
  - ***Appels systèmes*** (open(…), read(…), printf(…)).
- Deux **modes d'exécution** :
  - Le ***mode superviseur (noyau, maître, …)***, mode privilégié qui autorise notamment l'appel à des instructions interdites en ***mode utilisateur*** (manipulation des interruptions).
  - Ce mode assure la protection du système d'exploitation assisté par le matériel

Notions de Base d'un SE (2)

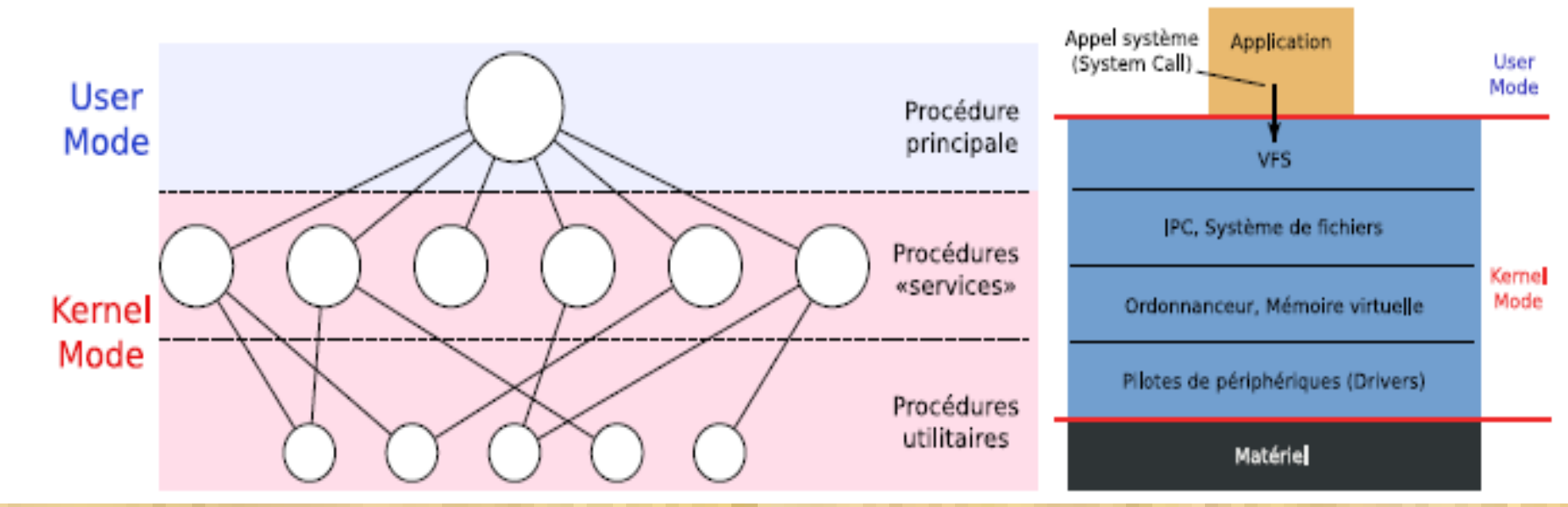
- Le passage du mode utilisateur vers le mode superviseur est soit provoqué par un ***appel système***, soit par une ***exception*** (déroutement en cas d'opération illicite), soit par l'arrivée d'une ***interruption***
- Une ***interruption*** est provoquée par un signal provenant du monde extérieur au processeur, et modifiant le comportement de celui-ci.
- Le passage entre les modes utilisateur/noyau s'accompagne de ***commutations de contexte*** (sauvegarde du contexte utilisateur - changement de mode d'exécution - restauration du contexte utilisateur).

Structure d'un SE [3]

- **A base de noyau:**
  - **Monolithique:** «Tout en un» seul morceau
    - Plus faciles à écrire
    - Moins élégants que les micro-noyaux
    - Plus performants (ou pas)
  - **Micronoyaux: Client/serveur**
    - Plus difficiles à écrire
    - Plus résistants aux bugs (donc plus sûres)
  - **Hybride:** combiner monolithique + micronoyau

Noyau monolithique

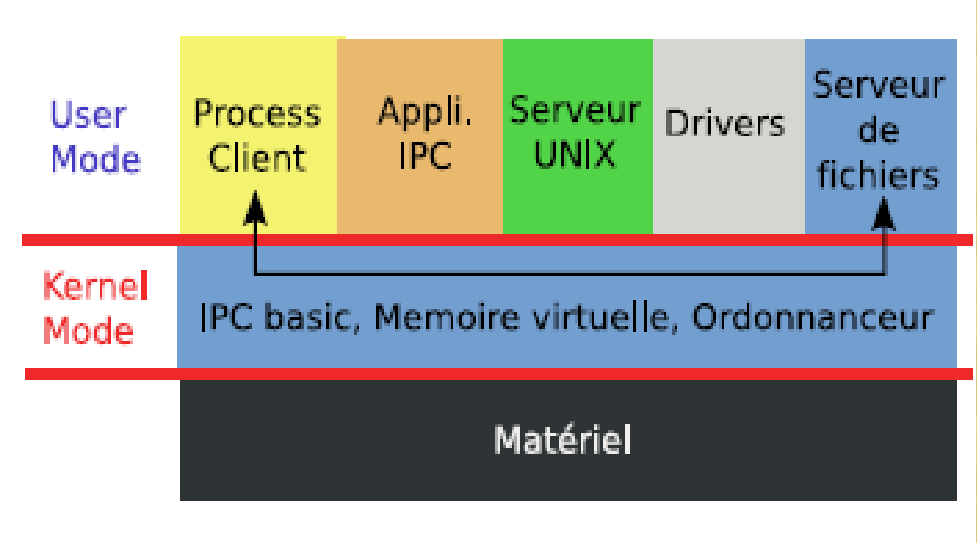
- Un seul programme → lourd et difficile à débugger
- Gâchis de mémoire (tout est chargé)
- Possibilité de modularité (Linux), exemple : Linux



Source: [4]

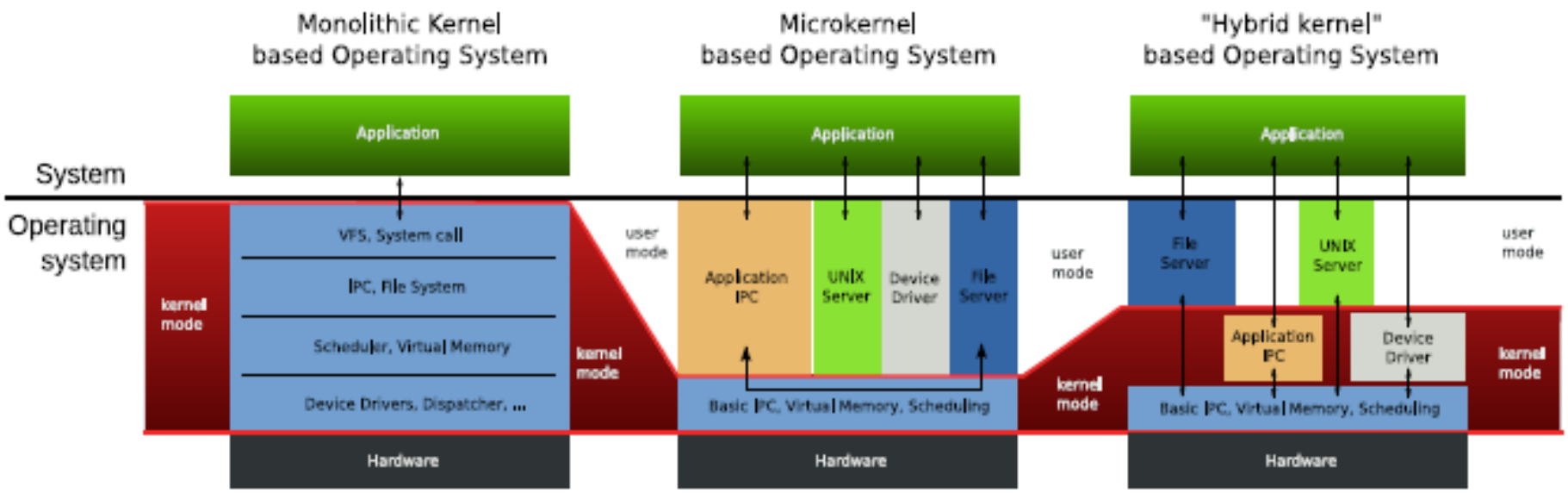
MicroNoyau [4]

- Noyau réduit au presque minimum (microkernel), le reste chargé en serveurs
  - ⇨ Structure client/serveur
- Gère principalement l'ordonnancement et les transferts de messages entre les programmes
- Les drivers et les applications s'exécutent en mode utilisateur
- Portable et facilement maintenable
- Exemples : Mach, Minix, L4, ...etc.



Structure Hybride [4]

- Compromis entre le monolithique et le micronoyau
- Les drivers s'exécutent en mode noyau → Optimisation
- Exemple : Windows NT



Structure Hybride [4]

- Compromis entre le monolithique et le micronoyau
- Les drivers s'exécutent en mode noyau → Optimisation
- Exemple : Windows NT

Bref Historique des Systèmes d'Exploitation (1)

- **1950s : Traitement par lots (batch processing)**
  - Apparition des transistors (permet des ordinateurs moins chers)
  - Assembleur, premiers langages évolués (Fortran, Cobol)
  - Le 1er SE gère l 'ordonnancement des jobs (résultats sur cartes/bandes)
- **1960s : Multiprogrammation**
  - Le SE partage ses ressources entre les utilisateurs
- **1970s : Systèmes interactifs à temps partagé**
  - Le SE partage son temps entre utilisateurs
    - Amélioration des SE (Performances, sécurité) et intégration des réseaux
- **1980s : Ordinateurs Personnels -- PCs**
  - Même SE pour des matériels différents, convivialité sans cesse accrue

Bref Historique des Systèmes d'Exploitation (2)

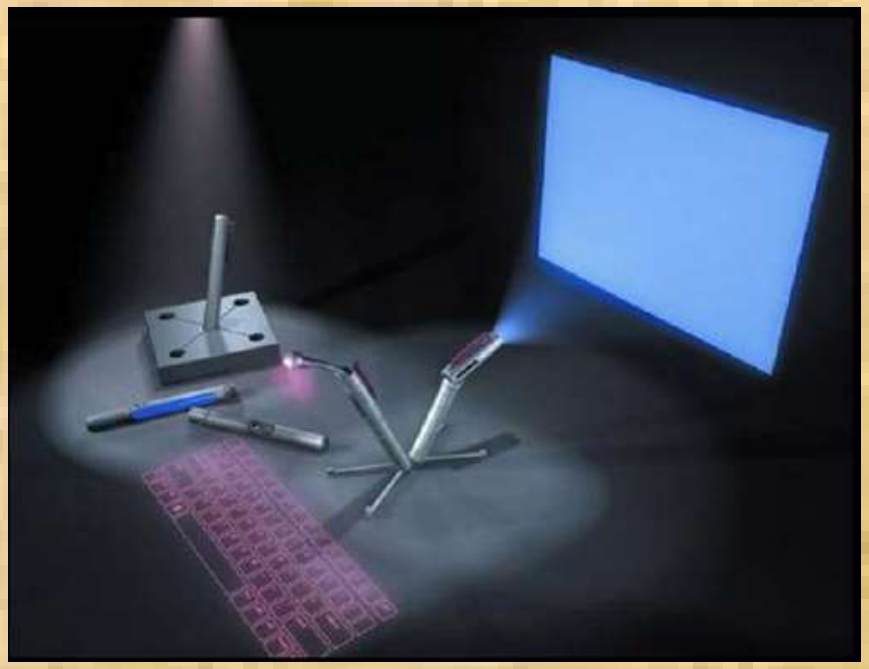
- **1990s : Systèmes distribués / Internet**
  - Ressources disponible à travers le réseau
  - Partage de données et des services sur une très grande échelle
    - World Wide Web -- développé au CERN (Genève)
  - Tout le monde (presque) est en-ligne
  - ***Systèmes Clients-Serveurs***
    - Un serveur est une machine assez puissante; les clients y accèdent via le réseau avec des machines standard
    - Solaris (Sun) sur les stations de travail, Windows NT sur PC
- **2000 : Réseau d'ordinateurs**
  - Matériel pas cher
  - Internet : Répertoire géant des ressources (prog., info., serveurs)
  - Commerce électronique
  - Recherche
    - Grands enjeux de sécurité (cryptographie)

Aujourd'hui

- **Les systèmes actuels :**
  - Les systèmes temps partagé (Unix, Windows NT); les systèmes temps réel (commande de procédés industriels)
  - Les systèmes transactionnels (grande BD; MAJ de la BD par des transactions)
  - Les systèmes multi-cœurs (**[5] --lecture recommandée)**
  - Les systèmes distribués
- **Personal and mobile Computer Systems (Mobile computing - Wireless computing)**
  - *Systèmes mobiles*: périodiquement connectés au reste du système, autonomes le reste du temps.
  - Les unités/dispositifs mobiles:
    - Ordinateurs portables, smartphones, tablettes, ...
    - Réseaux sans fil (WiFi, RFID, BLE, ...)

et Demain …

- **Ubiquitous and pervasive computing:**
  - 98% des processeurs ne sont pas dans les PCs! Il sont partout!!!
  - Tout dispositif pourrait communiquer (les bâtiments, les frigos, les voitures, …)



Conclusions: Challenges d'un SE Moderne
source: [1]

- Architecture micronoyaux (client/serveur)
  - ⇨ Le noyau contient le minimum nécessaire → Conception OO
- Multithreading/Mprogrammation multicores
  - ⇨ Exécution multi-threadée en intra-processus
- Systèmes concurrents
  - ⇨ Systèmes multi-processeurs (parallèles)
  - ⇨ Systèmes répartis/distribués
  - ⇨ Multi-threaded/multicore processors

Conclusions: Challenges d'un SE Moderne (suite) source: [1]

- Illusion de mémoire unique
  - sur un espace de mémoire secondaire → Plateforme à base de clusters
- Systèmes temps réels (Real-time OS)
  - For time-critical applications, multimedia, …
- Systèmes embarqués (Embedded OS)
  - Contraintes: Ressources limitées, fonctionnalités particulières de gestion de ressources

Références et Lectures Complémentaires

[1]	**Computer and OS System Overview:** ***http://www.cse.chalmers.se/edu/year/2010/course/EDA092/SLIDES/1-2-csos-overview-10.pdf***
[2]	Noyau d'un système d'exploitation, INF2610 , Ecole Polytechnique de Montréal, 2009.
[3]	*F. Najjar. Les micronoyaux, Module Conception des Systèmes et Systèmes Ouverts, II2-SRI. ENSI, 2010.*
[4]	*R. Bonidale, Systemes d'exploitation, Loria (France), 2010*
[5]	**Jernej Barbic, Multi-core architectures 15-213, Spring 2007**

Mot de fin (première séance) …

Annexe 1: Rappel sur les Unités de Mesure

Spatiale

De plus en plus grand

- **K**ilo-**O**ctet (**KB**) 2^{10} (10^3)
- **M**éga-**O**ctet (**MB**) 2^{20} (10^6)
- **G**iga-**O**ctet (**GB**) 2^{30} (10^9)
- **T**éra**O**ctet (**TB**) 2^{40} (10^{12})
- **P**éta-**O**ctet (**PB**) 2^{50} (10^{15})
- **E**xa-**O**ctet (**EB**) 2^{60} (10^{18})
- **Z**eta-**O**ctet (**ZB**) 2^{70} (10^{21})
- **Y**otta-**O**ctet (**YB**) 2^{80} (10^{23})

Temporelle

De plus en plus petit

- Milli (**ms**) 10^{-3}
- Micro (**µs**) 10^{-6}
- Nano (**ηs**) 10^{-9}
- Pico (**ps**) 10^{-12}
- Femto (**fs**) 10^{-15}

Annexe 2: Caractéristiques techniques

Level	1	2	3	4
Name	registers	cache	main memory	disk storage
Typical size	< 1 KB	> 16 MB	> 16 GB	> 100 GB
Implementation technology	custom memory with multiple ports, CMOS	on-chip or off-chip CMOS SRAM	CMOS DRAM	magnetic disk
Access time (ns)	0.25 – 0.5	0.5 – 25	80 – 250	5,000.000
Bandwidth (MB/sec)	20,000 – 100,000	5000 – 10,000	1000 – 5000	20 – 150
Managed by	compiler	hardware	operating system	operating system
Backed by	cache	main memory	disk	CD or tape

Source: Silberschatz et al.

Annexe 3: Architecture Matérielle (single-core vs. Multi-core)