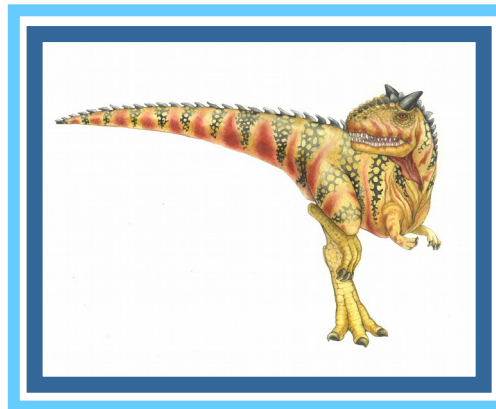


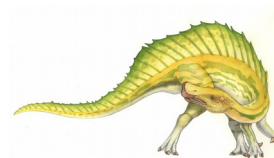
Chapitre 1: Introduction





Chapitre 1: Introduction

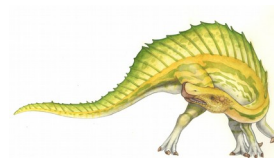
- Ce que fait un système d'exploitation
- Système informatique
 - Organisation
 - Architecture
 - Structure
 - Opérations
- Gestion
 - Processus
 - Memoire
 - Stockage
- Protection et sécurité
- Structures de données du noyaux
- Environnements informatiques





Objectifs

- Décrire l'organisation de base d'un système informatique
- Faire le tour des composants principaux d'un système d'exploitation
- Donner une vue d'ensemble des différents environnements informatiques
- Explorer les systèmes d'exploitation libres

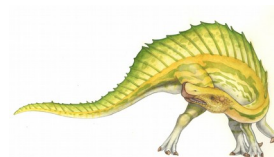




Un système d'exploitation?

- Un programme qui fait l'intermédiaire entre l'utilisateur d'un système informatique et le matériel

- Buts du système d'exploitation:
 - Exécuter les programmes de l'utilisateur
 - Rendre le système informatique plus facile d'usage
 - Utiliser le matériel de manière efficace





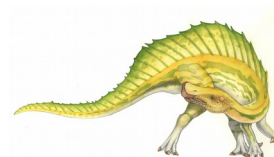
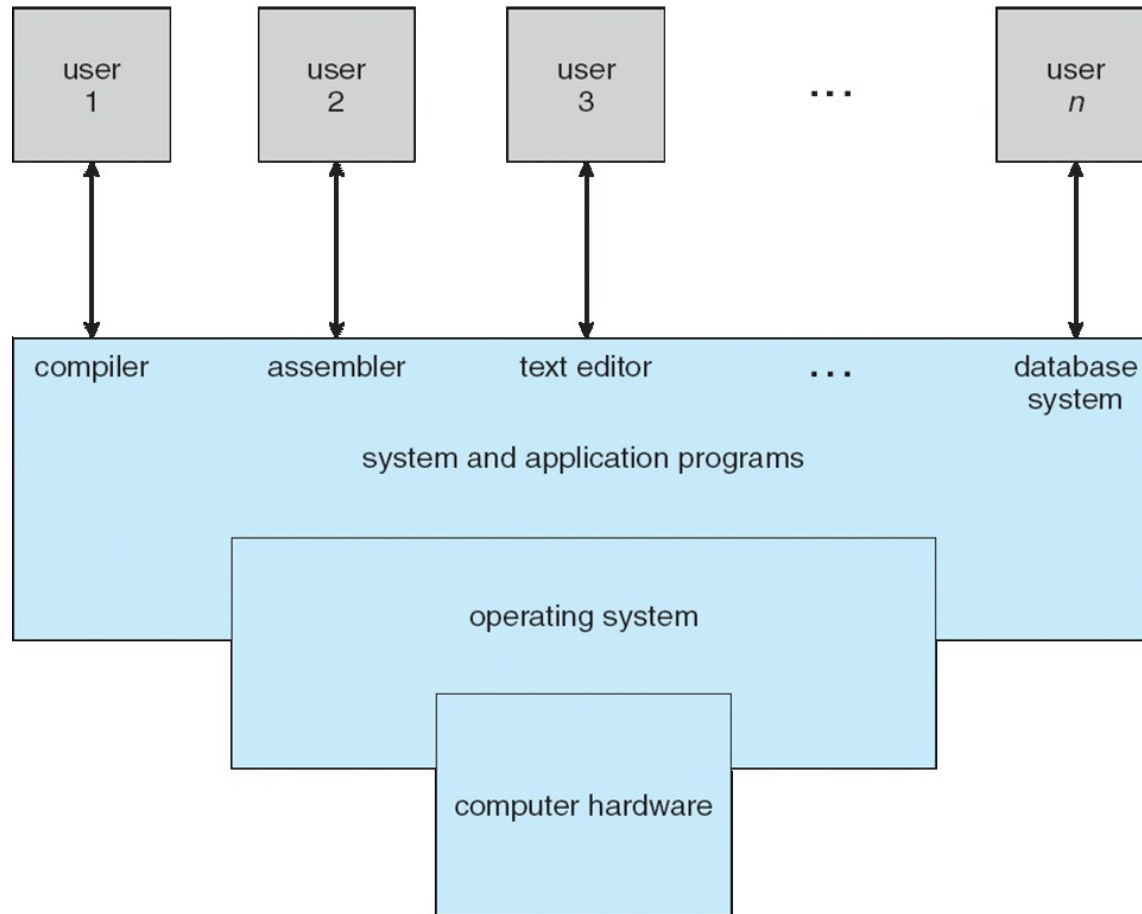
Structure d'un système informatique

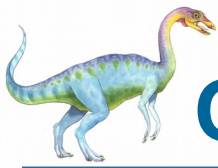
- Un système informatique peut se décomposer en 4 parties:
 - Matériel – fournis les ressources informatiques fondamentales
 - ▶ CPU, mémoire, périphériques d'entrée/sortie (I/O)
 - Système d'exploitation
 - ▶ Contrôle et coordone l'usage du materiel entre les différents utilisateurs et applications
 - Programmes d'application – défini comment les ressources du système sont utilisées pour résoudre les problèmes des utilisateurs
 - ▶ Traitements de textes, compilateurs, navigateurs, système de bases de données, jeux vidéo
 - Utilisateurs
 - ▶ Toi et moi, mais aussi d'autres machines





Quatre éléments d'un système informatique





Que fait un système d'exploitation

- Ça dépend à qui on demande
- Sur un système partagé (serveur, **mainframe** ou **minicomputer**), partager les ressources pour satisfaire tous les utilisateurs
- Utilisateurs de système dédiés (**workstations**, PC) partagent aussi souvent des ressources sur un serveur
- Utilisateurs isolés veulent de la **facilité d'utilisation**
 - L'usage des ressources est moins important
- Tablettes et téléphones ont des ressources limitées, où il faut optimiser la durée de la batterie
- Certains ordinateurs n'ont pas d'interface utilisateur, tels que les systèmes embarqués dans des appareils ou des véhicules

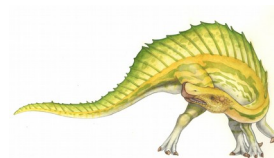




Définition d'un système d'exploitation

- SE est un **allocateur de resource**
 - Gère toutes les ressources
 - Décide entre plusieurs requêtes en conflit, pour rendre l'usage des ressources efficace et juste

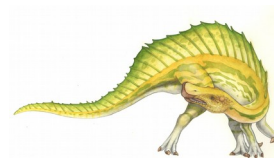
- SE est un **programme de contrôle**
 - Contrôle l'exécution des programmes pour éviter les erreurs et usages incorrect de la machine

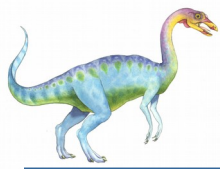




Déf. d'un système d'exploitation (suite)

- Pas de définition acceptée universellement
- “Tout ce qu'un vendeur donne quand on lui demande un système d'exploitation” est une bonne première approximation
 - Mais cela varie exagérément
- “Le ou les programmes qui sont toujours en fonctionnement”
- Le **noyau**. Tout le reste est soit un “programme système” (vient avec le système d'exploitation) ou un programme d'application





Computer Startup

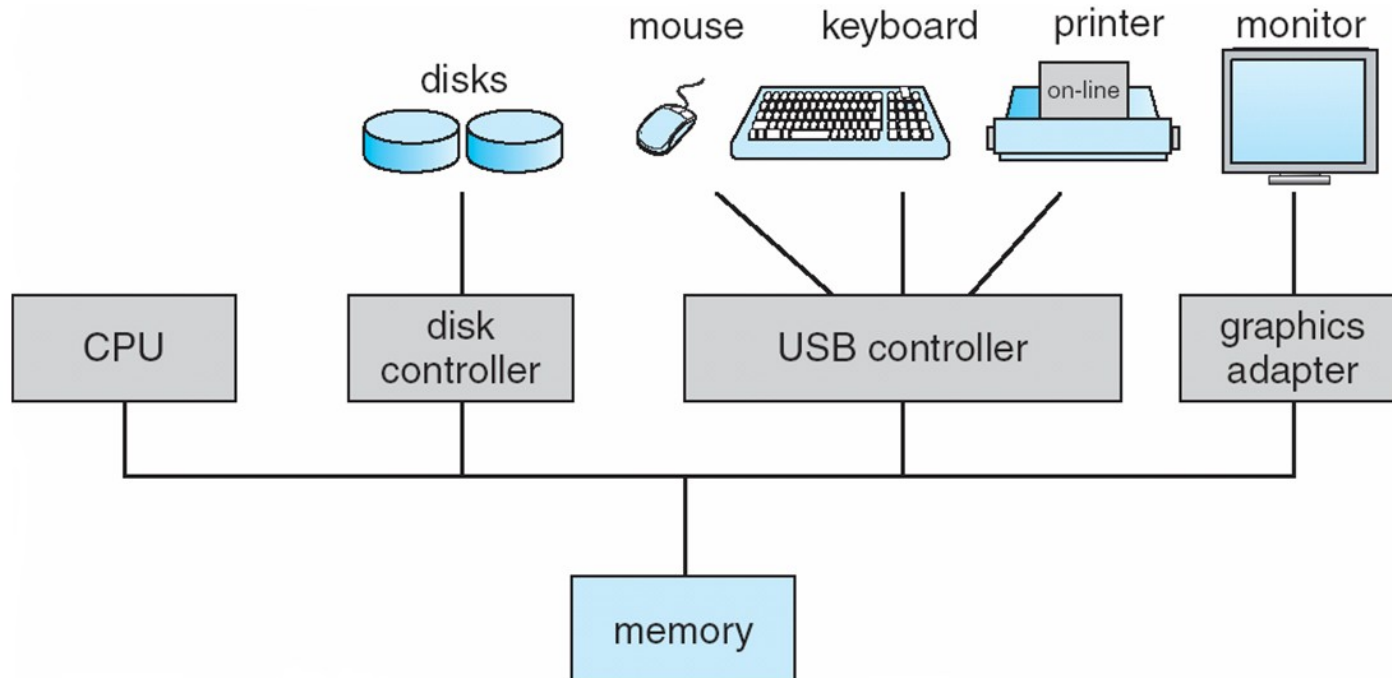
- **Programme de bootstrap** est chargé au démarrage
 - Habituellement stocké en mémoire Flash ou ROM (read-only memory), généralement dénommé **firmware**
 - Initialise tout le matériel du système
 - Charge le noyau du système d'exploitation et lance son exécution





Organisation d'un système informatique

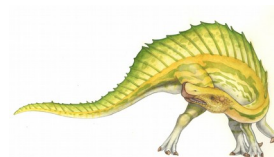
- Opération d'un système informatique
 - Les CPUs et contrôleurs de périphériques sont connectés par un bus commun donnant accès à une mémoire partagée
 - Exécution concurrente des CPUs and des périphériques qui luttent pour accéder au bus et à la mémoire

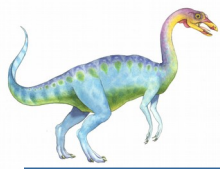




Opération d'un système informatique

- CPUs et périphériques d'E/S peuvent exécuter concurremment
- Chaque contrôleur de périphérique se charge d'un type particulier de périphérique
- Chaque contrôleur de périphérique a un tampon local
- CPU déplace les données de/vers la mémoire centrale vers/de les tampons locaux
- Les E/S se font entre le périphérique et le tampon local du contrôleur
- Le contrôleur de périphérique informe le CPU qu'il a terminé une opération en signalant une **interruption**





Interruptions

- Une interruption cause un transfert de contrôle à la routine de service d'interruptions, généralement via un **vecteur d'interruptions**, qui contient les adresses des différentes routines de service
- Le CPU sauve l'adresse de l'instructions interrompue
- Un “**trap**” ou une **exception** est une sorte d'interruption générée par software causée par une erreur ou une requête
- Un système d'exploitation est **guidé par les interruptions**



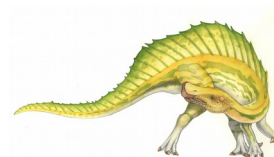


Gestion des interruptions

- Le système d'exploitation préserve l'état antérieur du CPU, tel que le contenu des registres

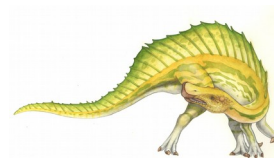
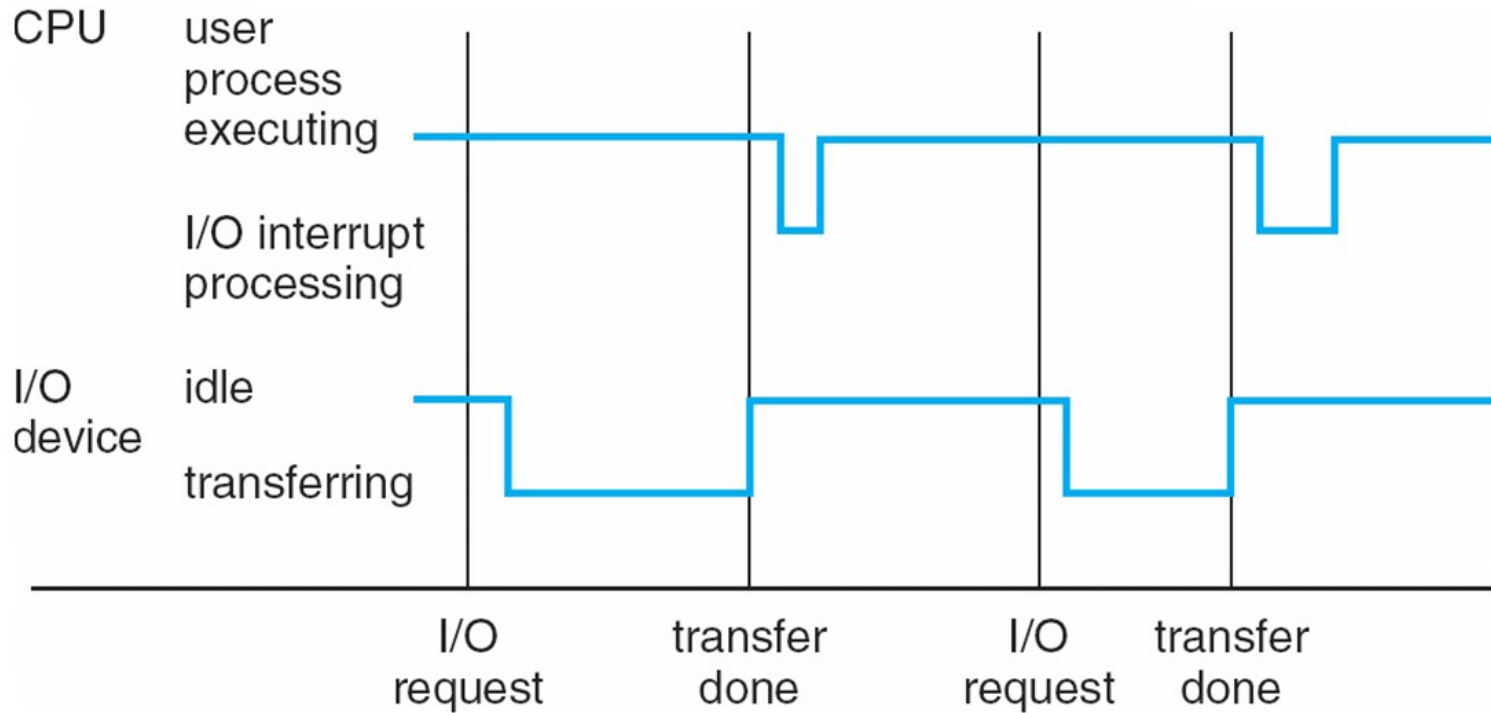
- Détermine quel type d'interruption est arrivée
 - Par scrutation (**Polling**)
 - Par **vecteur d'interruption**

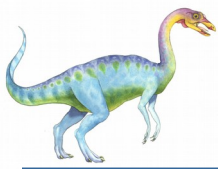
- Différent segments de code déterminent l'action à prendre pour chaque type d'interruption





Ligne de temps d'une interruption





Structure des E/S

- Après le début d'une E/S, le contrôle retourne au programme seulement quand l'E/S est complétée:
 - Des instructions d'attente mettent le CPU en pause jusqu'à la prochaine interruption
 - Boucle d'attente active
 - Pas plus d'une requête d'E/S en attente à la fois, pas d'E/S simultanées

- Après le début d'une E/S, le contrôle retourne au programme sans attendre la fin de l'E/S:
 - **Appels systèmes** – requêtes au SE pour permettre au programme d'attendre la fin de l'E/S
 - **Table d'état des périphériques** contient le type, l'adresse, et l'état de chaque périphérique





Storage Definitions and Notation Review

bit. Un bit contient soit 0 soit 1. Tout stockage est basé sur une collection de bits.

Un **byte** = 8 bits, et pour la majorité des systèmes, c'est la plus petite unité facile à manipuler.

Par exemple, la majorité des CPU n'a pas d'instruction pour déplacer un bit, mais en a pour déplacer un byte.

b=bit; **B**=byte

Un **mot** est l'unité "naturelle" d'un système informatique. E.g. 32bit.

Un **kilobyte** (ou **KB**) = 1,024 bytes

Un **megabyte** (or **MB**) = $1,024^2$ bytes

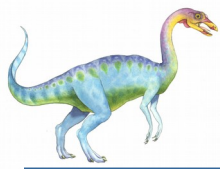
Un **gigabyte** (ou **GB**) = $1,024^3$ bytes

Un **terabyte** (ou **TB**) = $1,024^4$ bytes

Un **petabyte** (ou **PB**) = $1,024^5$ bytes

Il est courant d'arrondir ces nombres en milliers, et de dire par exemple qu'un megabyte est 1 million de bytes and a gigabyte is 1 billion bytes. Les vitesses de communication sont souvent mesurées en multiple de bits plutôt que de bytes.

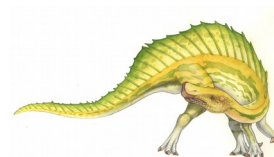




Structure du DMA

DMA = Accès Direct à la Mémoire

- Utilisé pour les périphériques rapides, capables de transférer des données à haut débit
- Le contrôleur de DMA transfère des blocs de données directement de/vers la mémoire centrale sans intervention du CPU
- Une seule interruption par transfert plutôt qu'une pour chaque byte





Structures de stockage

- Mémoire centrale – seul stockage auquel le CPU peut accéder directement
 - **Accès aléatoire**
 - Typiquement **volatile**
- Mémoire de masse – extension de la mémoire centrale qui offre une plus grande capacité et n'est **pas volatile**
- Disque magnétique – plateaux de metal ou verre recouvert d'un matériau magnétique d'enregistrement
 - La surface est divisée en **pistes**, elles mêmes subdivisées en **secteurs**
 - Le **contrôleur de disque** détermine l'interaction entre le périphérique et l'ordinateur
- **disques SSD** – plus rapides que les disques magnétiques mais quand même nonvolatils
 - Diverses technologies
 - Très populaire de nos jours



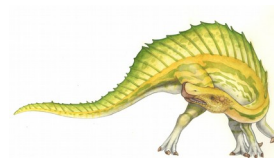


Hiérarchie de stockage

- Les systèmes de stockage sont organisés en hiérarchie
 - Vitesse
 - Coût
 - Volatilité

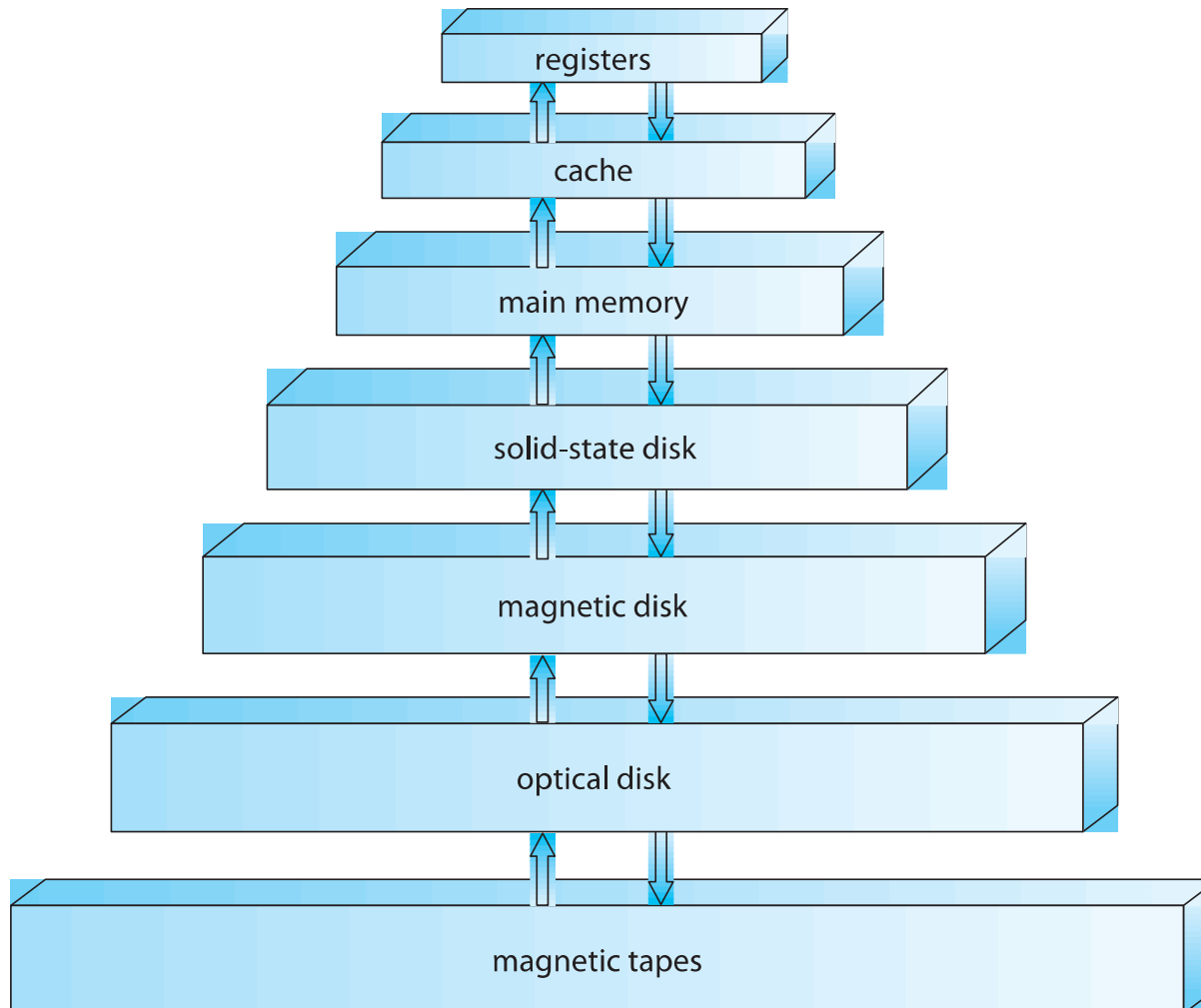
- **Caching** – copier l'information dans un système de stockage plus rapide; la mémoire centrale peut être utilisée comme un cache de la mémoire de masse

- **Pilote de périphérique** pour chaque contrôleur de périphérique pour gérer les E/S
 - Fourni une interface uniforme entre le contrôleur et le noyau





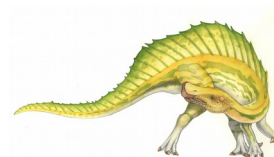
Hiérarchie de stockage





Caching

- Principe important, utilisé à toute sortes d'endroits et de niveaux dans un ordinateur (en matériel, par le SE, par les applications)
- Copier l'information active vers une mémoire plus rapide
- Vérifier d'abord si la donnée est dans le cache
 - Si elle y est, on peut l'utiliser directement (rapide)
 - Si non, on la copie d'abord et on l'utilise après
- Pour être plus rapide, le cache est plus petit
 - Le design de la gestion du cache est très important
 - Taille du cache et règles de remplacement

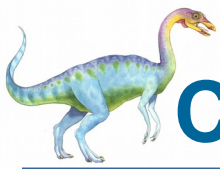




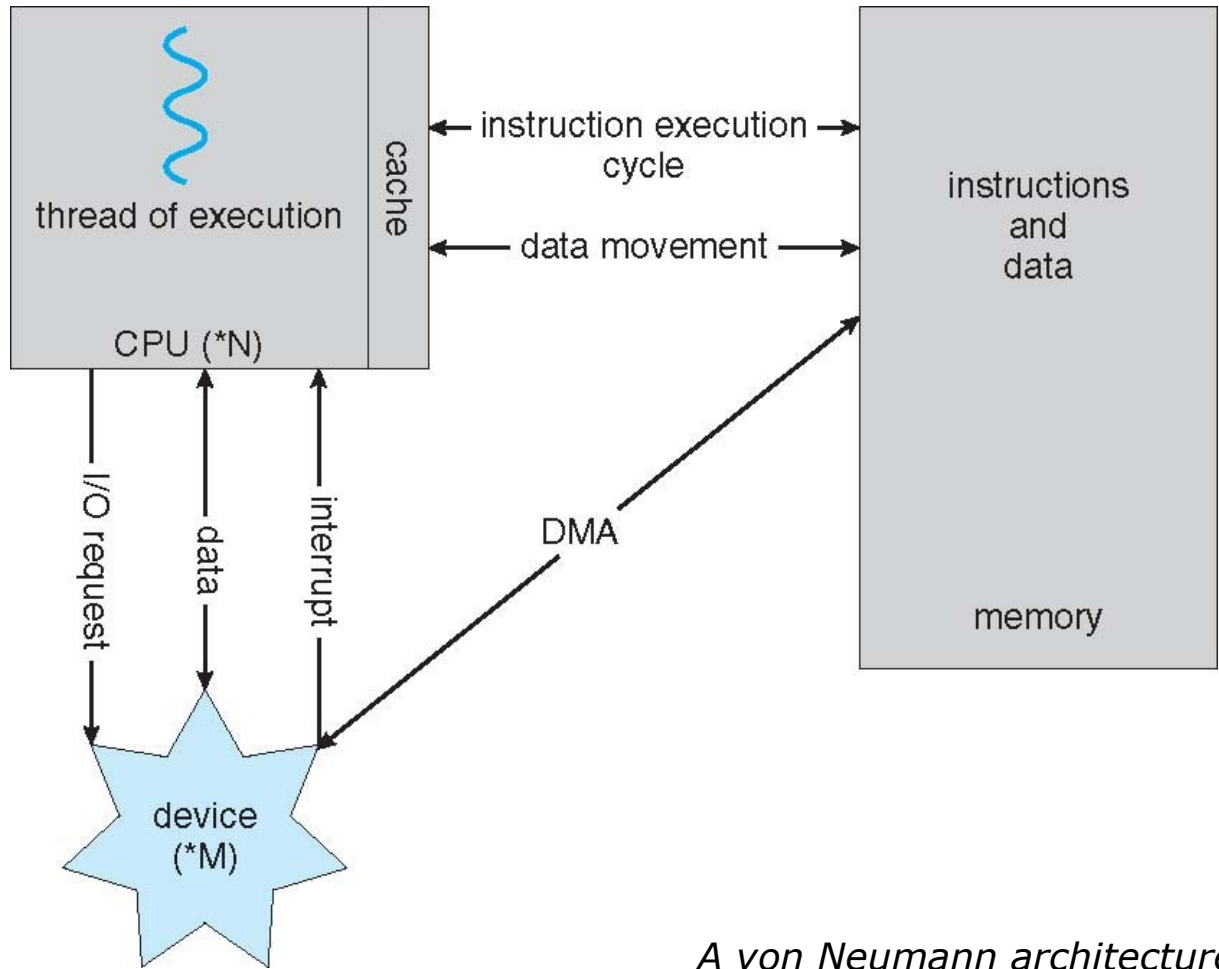
Architecture des systèmes

- Beaucoup de système n'ont qu'un seul CPU
 - La majorité ont aussi des processeurs spécialisés
- Les systèmes **multiprocesseurs** sont toujours plus courants et importants
 - Aussi nommés **systèmes parallèles**, **systèmes fortement couplés**
 - Avantages:
 1. **Plus haute performance**
 2. **Économies d'échelle**
 3. **Meilleure fiabilité – dégradation progressive, tolérance aux pannes**
 - Deux types:
 - 4 **Multiprocessing asymétrique** (un CPU gère les autres)
 - 4 **Multiprocessing symétrique** (tous les CPUs sont égaux)

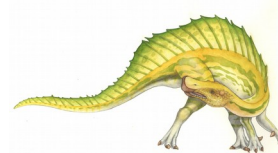




Comment fonctionne un ordinateur

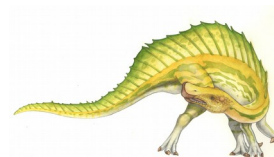
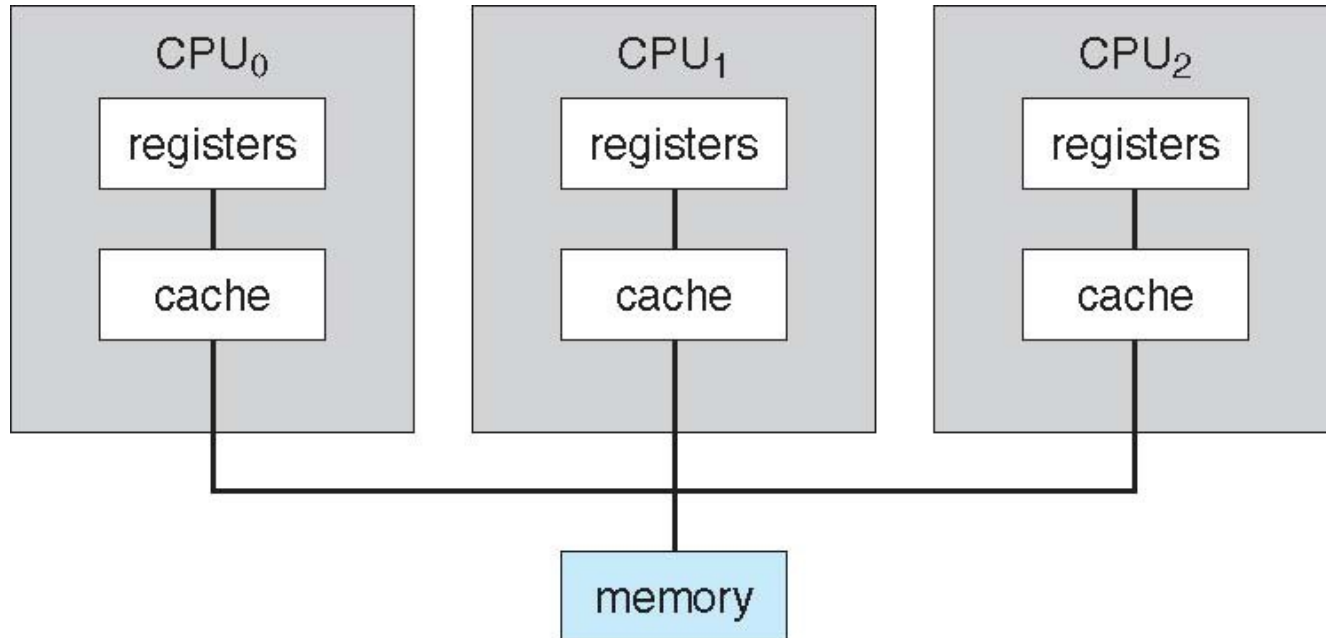


A von Neumann architecture





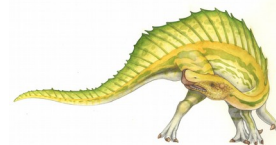
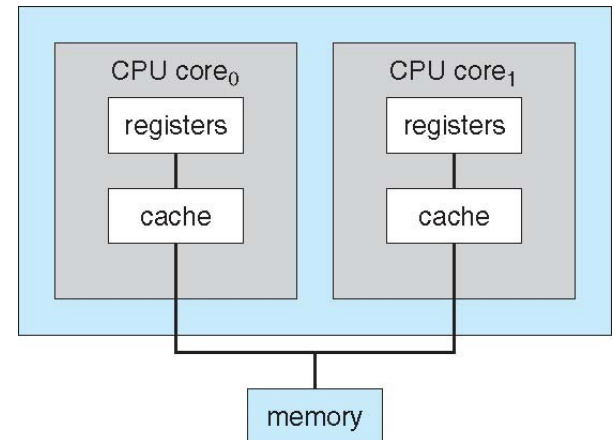
Architecture multiprocesseur symétrique





Un système dual-core

- Variations **UMA** et **NUMA**
(accès mémoire uniforme et nonuniforme)
- Multi-chip ou **multicore**
- Systèmes contenant tout les chips vs **serveurs blade**
 - Chassis contenant plusieurs systèmes séparés (avec leur OS, I/O, réseau, etc.)





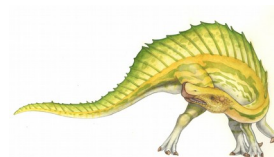
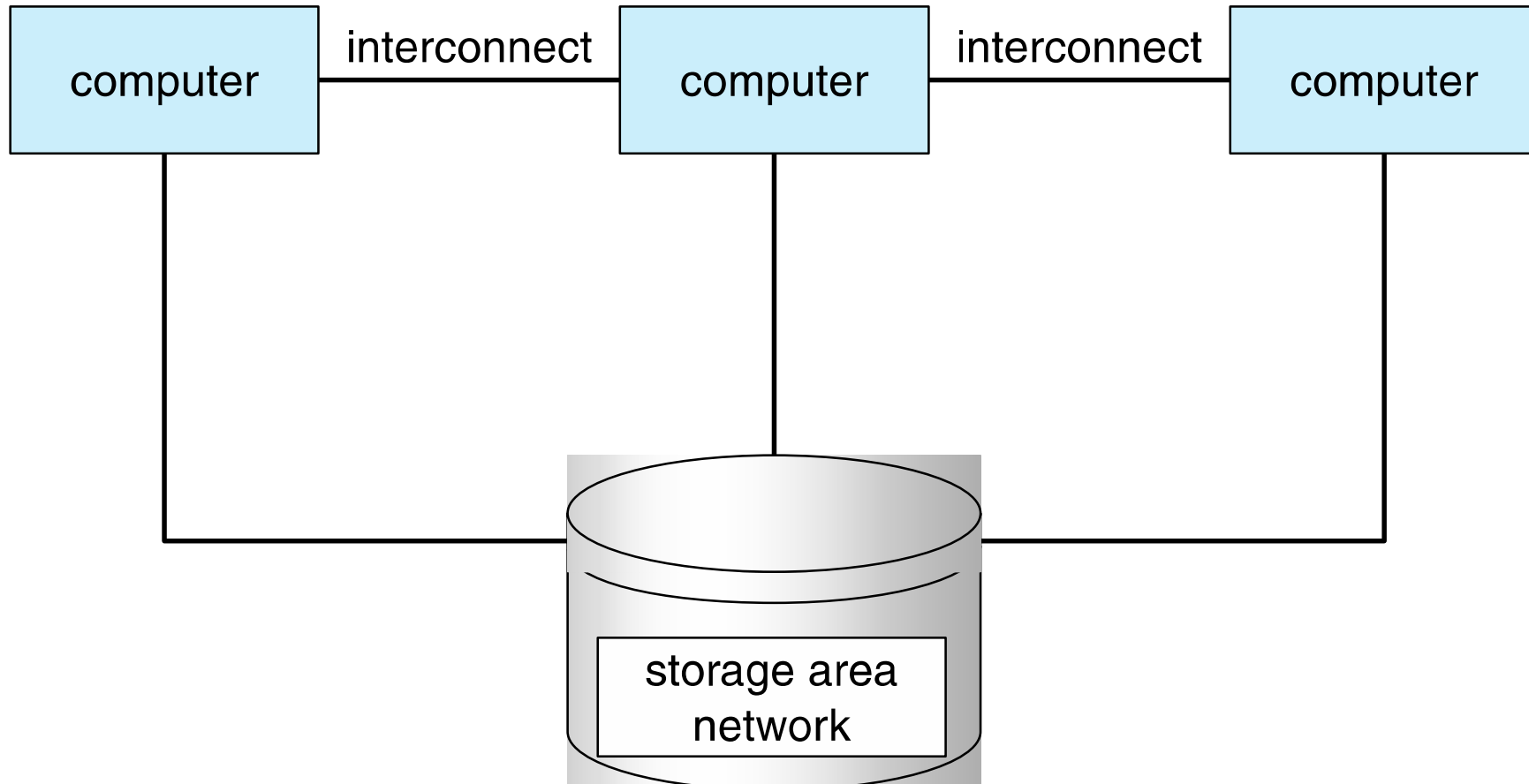
Systemes en grappes

- Comme systemes multiprocesseurs, mais multiples systemes travaillant ensemble
 - Partage du stockage via un **storage-area network (SAN)**
 - Offre un service **high-availability** qui survit aux pannes
 - ▶ **Grappes asymmetriques** avec une machine en hot-standby
 - ▶ **Grappes symmetriques** où les noeuds se surveillent mutuellement
 - Grappes de calcul **high-performance computing (HPC)**
 - ▶ Applications écrites pour utiliser la **parallélisation**
 - Usage de **distributed lock manager (DLM)** pour éviter les conflits dans certaines opérations





Systemes en grappes

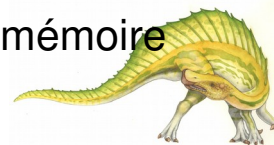




Structure des systèmes d'exploitation

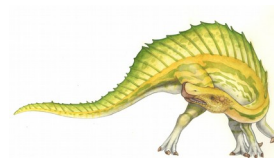
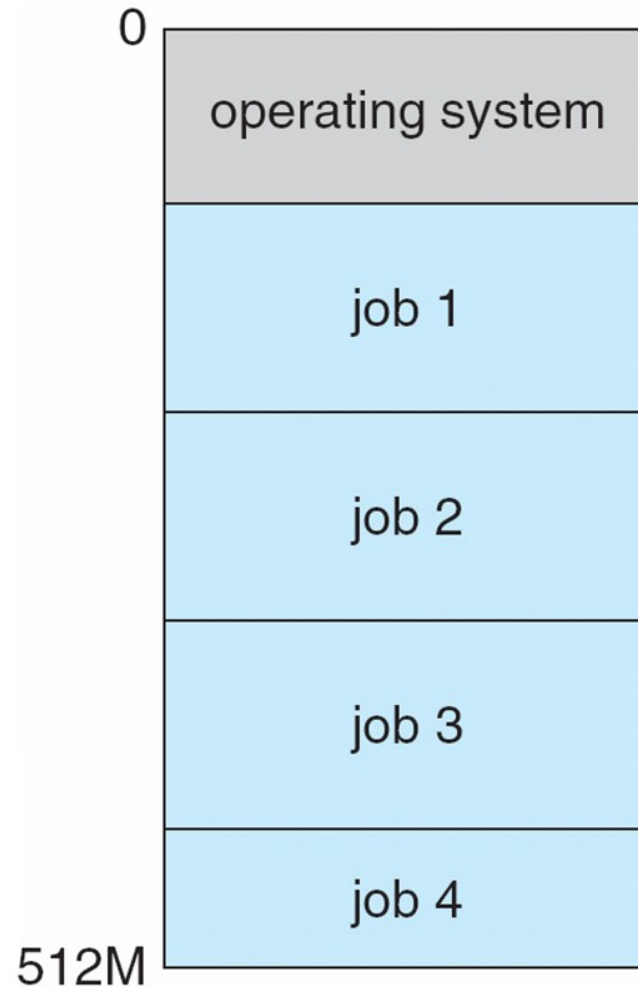
- Besoin de **multiprogrammation** pour l'efficacité
 - Un seul utilisateur ne peut garder le CPU et les E/S occupées
 - Multiprogrammation organise le travail (code and data) pour que le CPU aie toujours quelque chose à faire
 - Un sous ensemble des tâches gardé en mémoire
 - Choix d'exécution des tâches par **job scheduling**
 - En cas d'attente (par exemple pour E/S), SE passe à une autre tâche

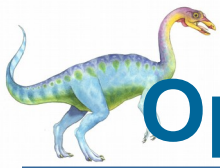
- **Timesharing (multitasking)** une extension naturelle où le CPU change de tâche si souvent que l'utilisateur peut interagir avec chacune d'elle: **interactive computing**
 - Temps de réponse devrait être < 1 second
 - Chaque utilisateur a au moins une tâche en mémoire □ **process**
 - Plusieurs tâches prêtes en même temps □ **CPU scheduling**
 - Tâches occupent plus de mémoire que disponible: **swapping** sur le disque
 - **Mémoire virtuelle** permet l'exécution de tâches partiellement en mémoire





Layout mémoire de système multiprogrammé

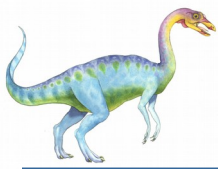




Opérations de systèmes d'exploitation

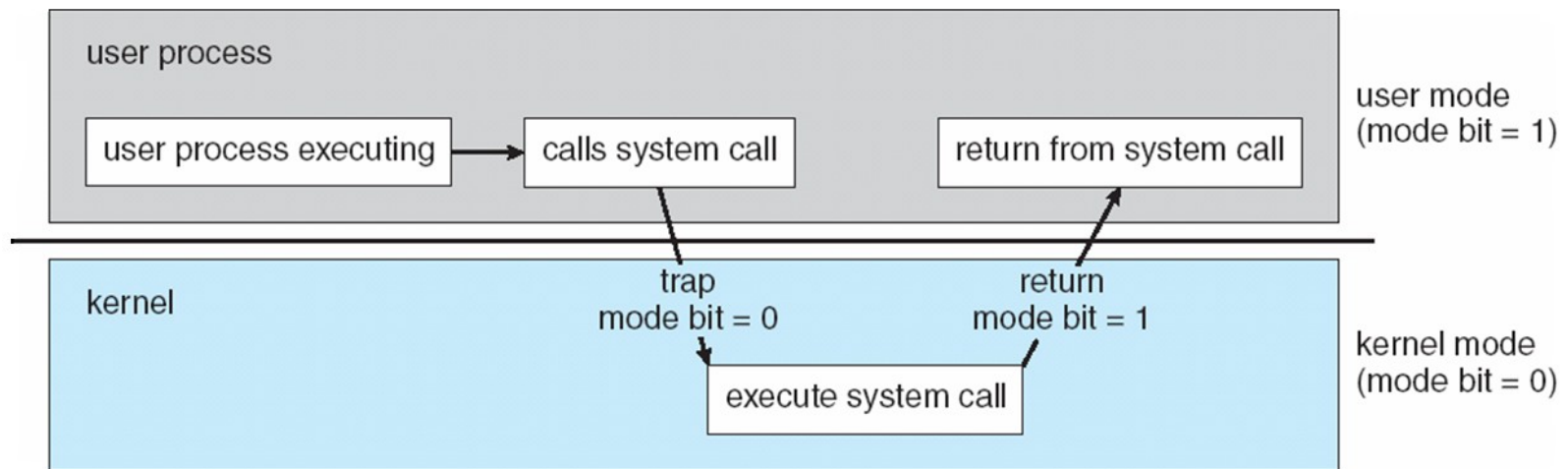
- **Guidé par les interruptions** par le matériel
- Erreur logicielle et requêtes créent des **exceptions ou trap**
 - Division par zéro, appel système
- Éviter les problèmes tels que les boucles infinies, les processus qui en modifient d'autres ou qui modifient le SE lui-même
- Opération en **deux modes** permet au SE de protéger les éléments
 - **Mode utilisateur** et **mode noyau**
 - **Bit de mode** fourni par le processeur
 - ▶ Permet de distinguer l'exécution dans le noyau
 - ▶ Les instructions “**privileged**”, autorisées seulement en mode noyau
 - ▶ Interruption et exceptions passent en mode noyau





Transition au mode noyau

- Pour éviter les boucles sans fin, le noyau doit garder un œil ouvert
 - Utiliser un timer pour générer une interruption dans le futur
 - Avant de retourner en mode user, le SE arme le timer
 - Quand le timer sonne, l'interruption redonne la main au noyau





Gestion des processus

- Un processus est un programme en cours d'exécution. Un programme est une **entité passive**, un processus est une **entité active**.
- Un processus a besoin de ressources pour accomplir sa tâche
 - CPU, mémoire, E/S, fichiers
 - Initialisation des données
- À la fin d'un processus, il faut récupérer ces ressources
- Un processus "single-threaded" a un **program counter** qui indique l'adresse de la prochaine instruction à exécuter
 - Le processus exécute ses instructions séquentiellement, une à la fois, jusqu'à épuisement
- Un processus "multi-threaded" a plusieurs program counters, un par thread, qui peuvent ainsi exécuter en parallèle
- Habituellement un système a toujours beaucoup de processus actifs, certains utilisateurs, d'autres appartenant au système, qui fonctionnent tous concurremment sur un ou plusieurs CPUs
 - Multiplexer les CPUs parmi les processus / threads

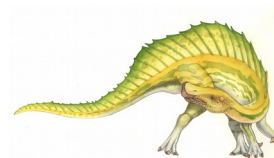




Activités de gestion des processus

Le système d'exploitation est responsable de:

- Créer et détruire les processus utilisateurs et système
- Suspendre et réveiller les processus
- Fournir des mécanismes pour la synchronisation entre processus
- Fournir des mécanismes pour la communication entre processus
- Fournir des mécanismes pour la gestion des étreintes mortelles





Gestion mémoire

- Avoir les données en mémoire avant de les manipuler
- Avoir les instructions en mémoire avant de les exécuter
- La gestion mémoire détermine quoi garder en mémoire pour
 - Optimiser l'usage du CPU et le temps de réponse à l'utilisateur
- Activités de gestion mémoire
 - Garder trace de qui utilise quelle partie de la mémoire
 - Decider quelles informations de quels processus transférer de/vers la mémoire
 - Allouer et désallouer les espaces mémoire au besoin





Gestion du stockage

- Se fournit une vue uniforme des capacités de stockage
 - Abstraires les détails physiques en unités logiques – les **fichiers**
 - Each medium is controlled by device (i.e., disk drive, tape drive)
 - ▶ Varying properties include access speed, capacity, data-transfer rate, access method (sequential or random)
- File-System management
 - Files usually organized into directories
 - Access control on most systems to determine who can access what
 - OS activities include
 - ▶ Creating and deleting files and directories
 - ▶ Primitives to manipulate files and directories
 - ▶ Mapping files onto secondary storage
 - ▶ Backup files onto stable (non-volatile) storage media





Mass-Storage Management

- Usually disks used to store data that does not fit in main memory or data that must be kept for a “long” period of time
- Proper management is of central importance
- Entire speed of computer operation hinges on disk subsystem and its algorithms
- OS activities
 - Free-space management
 - Storage allocation
 - Disk scheduling
- Some storage need not be fast
 - Tertiary storage includes optical storage, magnetic tape
 - Still must be managed – by OS or applications
 - Varies between WORM (write-once, read-many-times) and RW (read-write)

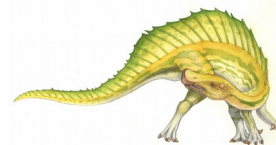




Performance of Various Levels of Storage

	128GB	1TB	4TB	DIRO	
Level	1	2	3	4	5
Name	registers	cache	main memory	solid state disk	magnetic disk
Typical size	< 1 KB	< 16MB	< 64GB	< 1 TB	< 10 TB
Implementation technology	custom memory with multiple ports CMOS	on-chip or off-chip CMOS SRAM	CMOS SRAM	flash memory	magnetic disk
Access time (ns)	0.25 - 0.5	0.5 - 25	80 - 250	25,000 - 50,000	5,000,000
Bandwidth (MB/sec)	20,000 - 100,000	5,000 - 10,000	1,000 - 5,000	500	20 - 150
Managed by	compiler	hardware	operating system	operating system	operating system
Backed by	cache	main memory	disk	disk	disk or tape

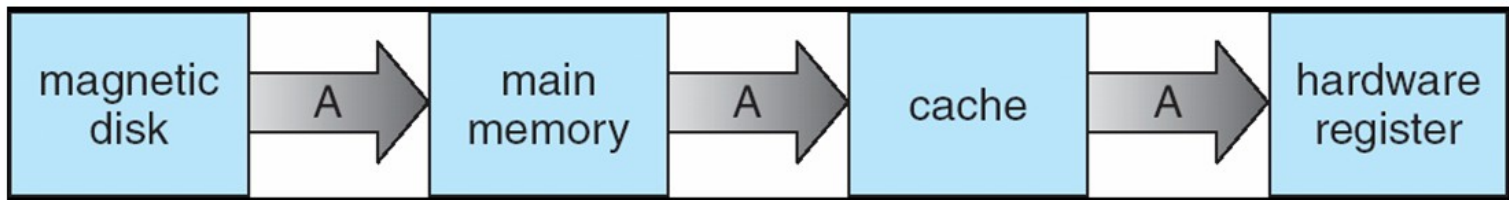
- Movement between levels of storage hierarchy can be explicit or implicit





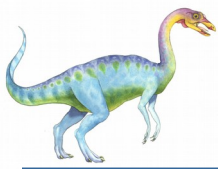
Migration of Integer A from Disk to Register

- Multitasking environments must be careful to use most recent value, no matter where it is stored in the storage hierarchy



- Multiprocessor environment must provide **cache coherency** in hardware such that all CPUs have the most recent value in their cache
- Distributed environment situation even more complex
 - Several copies of a datum can exist
 - Various solutions covered in Chapter 17





I/O Subsystem

- One purpose of OS is to hide peculiarities of hardware devices from the user

- I/O subsystem responsible for
 - Memory management of I/O including
 - ▶ buffering: storing data temporarily while it is being transferred
 - ▶ caching: storing parts of data in faster storage for performance
 - ▶ spooling: the overlapping of output of one job with input of other jobs
 - General device-driver interface
 - Drivers for specific hardware devices





Protection and Security

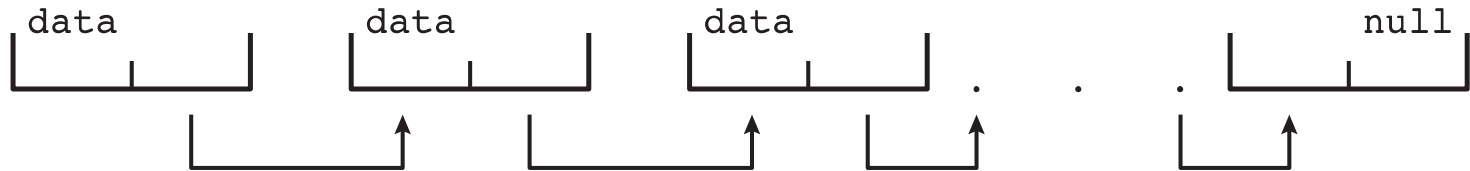
- **Protection** – any mechanism for controlling access of processes or users to resources defined by the OS
- **Security** – defense of the system against internal and external attacks
 - Huge range, including denial-of-service, worms, viruses, identity theft, theft of service
- Systems generally first distinguish among users, to determine who can do what
 - User identities (**user IDs**, security IDs) include name and associated number, one per user
 - User ID then associated with all files, processes of that user to determine access control
 - Group identifier (**group ID**) allows set of users to be defined and controls managed, then also associated with each process, file
 - **Privilege escalation** allows user to change to effective ID with more rights



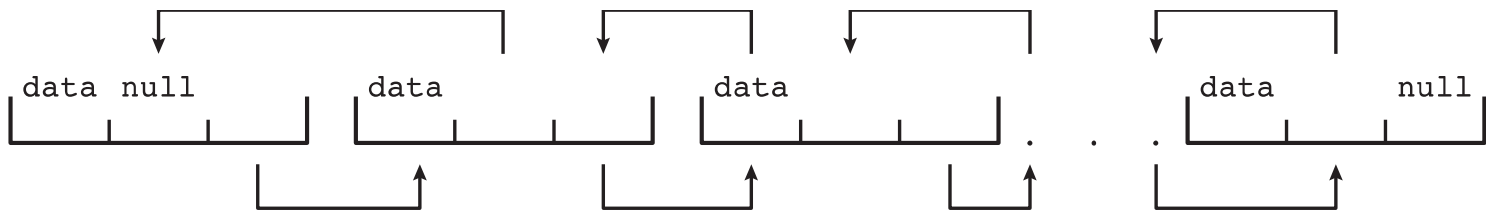


Kernel Data Structures

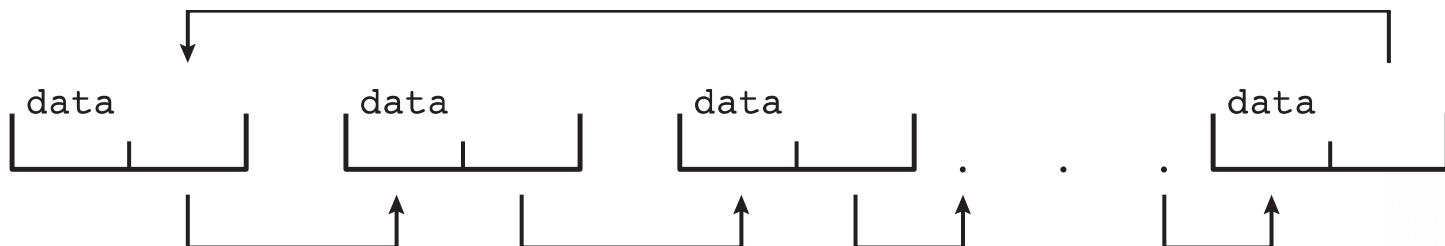
- Many similar to standard programming data structures
- ***Singly linked list***



- ***Doubly linked list***



- ***Circular linked list***



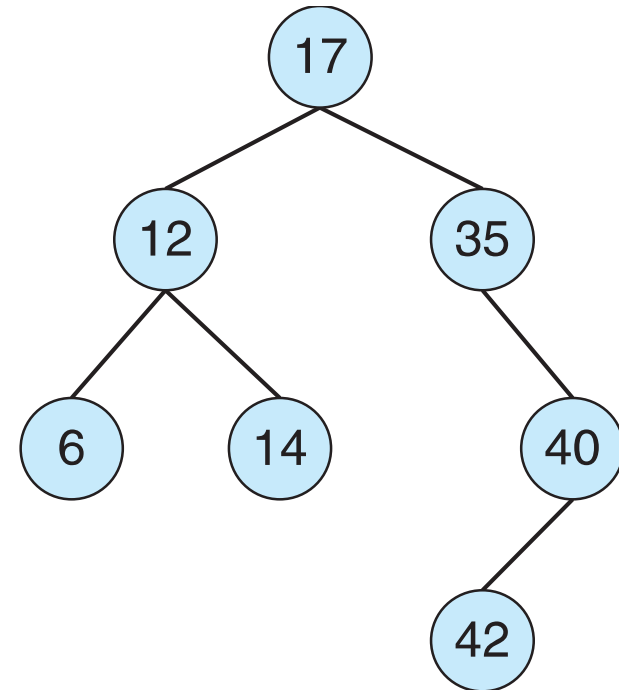


Kernel Data Structures

■ Binary search tree

left \leq right

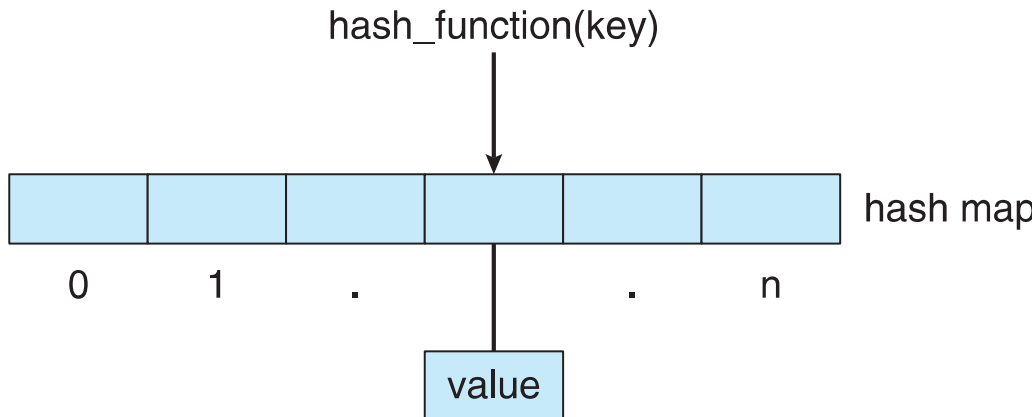
- Search performance is $O(n)$
- **Balanced binary search tree** is $O(\log n)$





Kernel Data Structures

- **Hash function** can create a **hash map**



- **Bitmap** – string of n binary digits representing the status of n items
- Linux data structures defined in ***include*** files
`<linux/list.h>`, `<linux/kfifo.h>`,
`<linux/rbtree.h>`





Computing Environments - Traditional

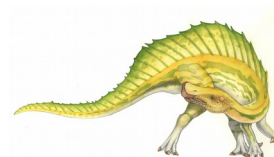
- Stand-alone general purpose machines
- But blurred as most systems interconnect with others (i.e., the Internet)
- **Portals** provide web access to internal systems
- **Network computers** (**thin clients**) are like Web terminals
- Mobile computers interconnect via **wireless networks**
- Networking becoming ubiquitous – even home systems use **firewalls** to protect home computers from Internet attacks





Computing Environments - Mobile

- Handheld smartphones, tablets, etc.
- What is the functional difference between them and a “traditional” laptop?
- Extra feature – more OS features (GPS, gyroscope)
- Allows new types of applications like ***augmented reality***
- Use IEEE 802.11 wireless, or cellular data networks for connectivity
- Leaders are **Apple iOS** and **Google Android**





Computing Environments – Distributed

■ Distributed

- Collection of separate, possibly heterogeneous, systems networked together
 - ▶ **Network** is a communications path, **TCP/IP** most common
 - **Local Area Network (LAN)**
 - **Wide Area Network (WAN)**
 - **Metropolitan Area Network (MAN)**
 - **Personal Area Network (PAN)**
 - **Network Operating System** provides features between systems across network
 - ▶ Communication scheme allows systems to exchange messages
 - ▶ Illusion of a single system

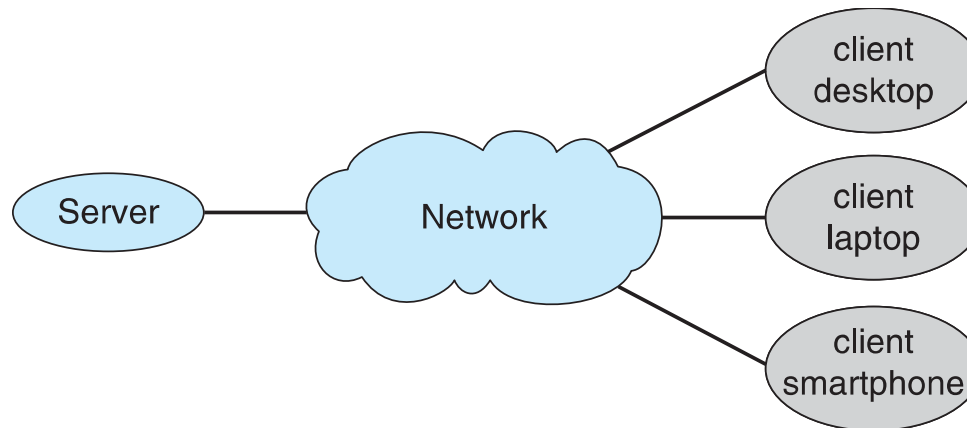




Computing Environments – Client-Server

■ Client-Server Computing

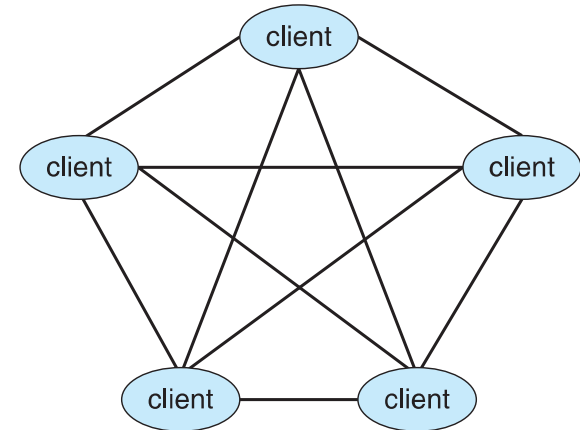
- Dumb terminals supplanted by smart PCs
- Many systems now **servers**, responding to requests generated by **clients**
 - ▶ **Compute-server system** provides an interface to client to request services (i.e., database)
 - ▶ **File-server system** provides interface for clients to store and retrieve files





Computing Environments - Peer-to-Peer

- Another model of distributed system
- P2P does not distinguish clients and servers
 - Instead all nodes are considered peers
 - May each act as client, server or both
 - Node must join P2P network
 - ▶ Registers its service with central lookup service on network, or
 - ▶ Broadcast request for service and respond to requests for service via **discovery protocol**
 - Examples include Napster and Gnutella, **Voice over IP (VoIP)** such as Skype





Computing Environments - Virtualization

- Allows operating systems to run applications within other OSES
 - Vast and growing industry
- **Emulation** used when source CPU type different from target type (i.e. PowerPC to Intel x86)
 - Generally slowest method
 - When computer language not compiled to native code – **Interpretation**
- **Virtualization** – OS natively compiled for CPU, running **guest** OSES also natively compiled
 - Consider VMware running WinXP guests, each running applications, all on native WinXP **host** OS
 - **VMM** provides virtualization services





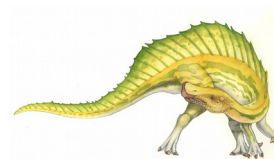
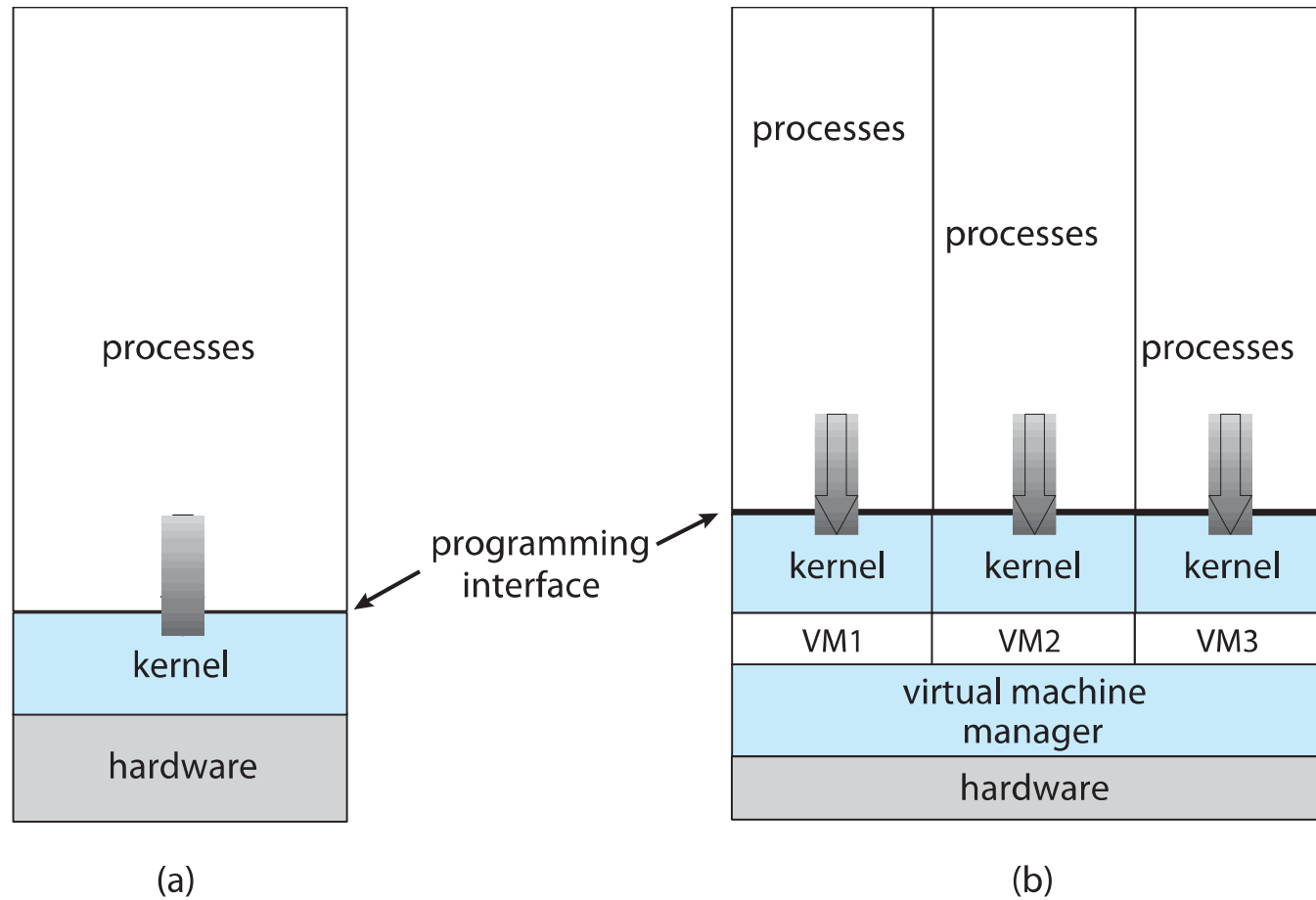
Computing Environments - Virtualization

- Use cases involve laptops and desktops running multiple OSES for exploration or compatibility
 - Apple laptop running Mac OS X host, Windows as a guest
 - Developing apps for multiple OSES without having multiple systems
 - QA testing applications without having multiple systems
 - Executing and managing compute environments within data centers
- VMM can run natively, in which case they are also the host
 - There is no general purpose host then (VMware ESX and Citrix XenServer)





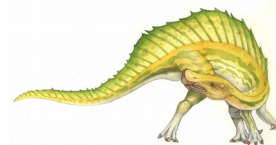
Computing Environments - Virtualization





Computing Environments – Cloud Computing

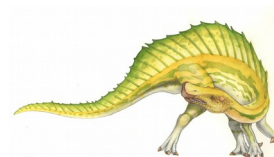
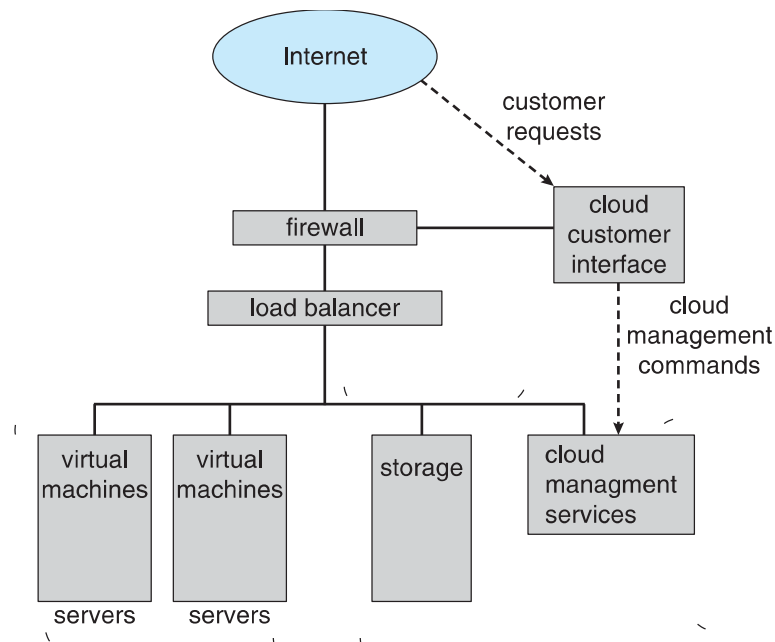
- Delivers computing, storage, even apps as a service across a network
- Logical extension of virtualization as based on virtualization
 - Amazon **EC2** has thousands of servers, millions of VMs, PBs of storage available across the Internet, pay based on usage
- Many types
 - **Public cloud** – available via Internet to anyone willing to pay
 - **Private cloud** – run by a company for the company's own use
 - **Hybrid cloud** – includes both public and private cloud components
 - Software as a Service (**SaaS**) – one or more applications available via the Internet (i.e. word processor)
 - Platform as a Service (**PaaS**) – software stack ready for application use via the Internet (i.e. a database server)
 - Infrastructure as a Service (**IaaS**) – servers or storage available over Internet (i.e. storage available for backup use)

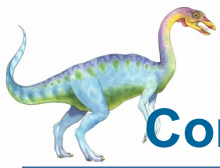




Computing Environments – Cloud Computing

- Cloud compute environments composed of traditional OSES, plus VMMs, plus cloud management tools
 - Internet connectivity requires security like firewalls
 - Load balancers spread traffic across multiple applications





Computing Environments – Real-Time Embedded Systems

- Real-time embedded systems most prevalent form of computers
 - Vary considerably, special purpose, limited purpose OS, **real-time OS**
 - Use expanding
- Many other special computing environments as well
 - Some have OSES, some perform tasks without an OS
- Real-time OS has well-defined fixed time constraints
 - Processing ***must*** be done within constraint
 - Correct operation only if constraints met



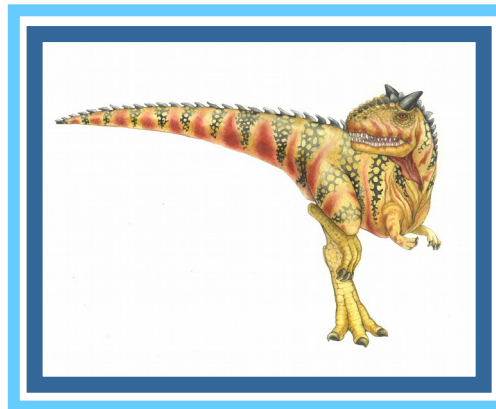


Open-Source Operating Systems

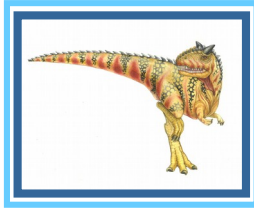
- Operating systems made available in source-code format rather than just binary **closed-source**
- Counter to the **copy protection** and **Digital Rights Management (DRM)** movement
- Started by **Free Software Foundation (FSF)**, which has “copyleft” **GNU Public License (GPL)**
- Examples include **GNU/Linux** and **BSD UNIX** (including core of **Mac OS X**), and many more
- Can use VMM like VMware Player (Free on Windows), Virtualbox (open source and free on many platforms - <http://www.virtualbox.com>)
 - Use to run guest operating systems for exploration



End of Chapter 1



Chapitre 1: Introduction





Chapitre 1: Introduction

- Ce que fait un système d'exploitation
- Système informatique
 - Organisation
 - Architecture
 - Structure
 - Opérations
- Gestion
 - Processus
 - Memoire
 - Stockage
- Protection et sécurité
- Structures de données du noyaux
- Environnements informatiques





Objectifs

- Décrire l'organisation de base d'un système informatique
- Faire le tour des composants principaux d'un système d'exploitation
- Donner une vue d'ensemble des différents environnements informatiques
- Explorer les systèmes d'exploitation libres





Un système d'exploitation?

- Un programme qui fait l'intermédiaire entre l'utilisateur d'un système informatique et le matériel

- Buts du système d'exploitation:
 - Exécuter les programmes de l'utilisateur
 - Rendre le système informatique plus facile d'usage
 - Utiliser le matériel de manière efficace





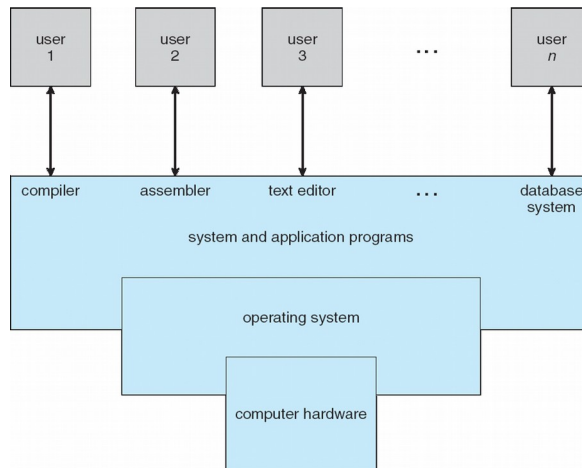
Structure d'un système informatique

- Un système informatique peut se décomposer en 4 parties:
 - Matériel – fournis les ressources informatiques fondamentales
 - CPU, mémoire, périphériques d'entrée/sortie (I/O)
 - Système d'exploitation
 - Contrôle et coordone l'usage du materiel entre les différents utilisateurs et applications
 - Programmes d'application – défini comment les ressources du système sont utilisées pour résoudre les problèmes des utilisateurs
 - Traitements de textes, compilateurs, navigateurs, système de bases de données, jeux vidéo
 - Utilisateurs
 - Toi et moi, mais aussi d'autres machines





Quatre éléments d'un système informatique





Que fait un système d'exploitation

- Ça dépend à qui on demande
- Sur un système partagé (serveur, **mainframe** ou **minicomputer**), partager les ressources pour satisfaire tous les utilisateurs
- Utilisateurs de système dédiés (**workstations**, PC) partagent aussi souvent des ressources sur un serveur
- Utilisateurs isolés veulent de la **facilité d'utilisation**
 - L'usage des ressources est moins important
- Tablettes et téléphones ont des ressources limitées, où il faut optimiser la durée de la batterie
- Certains ordinateurs n'ont pas d'interface utilisateur, tels que les systèmes embarqués dans des appareils ou des véhicules





Définition d'un système d'exploitation

- SE est un **allocateur de ressource**
 - Gère toutes les ressources
 - Décide entre plusieurs requêtes en conflit, pour rendre l'usage des ressources efficace et juste

- SE est un **programme de contrôle**
 - Contrôle l'exécution des programmes pour éviter les erreurs et usages incorrect de la machine





Déf. d'un système d'exploitation (suite)

- Pas de définition acceptée universellement
- “Tout ce qu'un vendeur donne quand on lui demande un système d'exploitation” est une bonne première approximation
 - Mais cela varie exagérément
- “Le ou les programmes qui sont toujours en fonctionnement”
- Le **noyau**. Tout le reste est soit un “programme système” (vient avec le système d'exploitation) ou un programme d'application



Microsoft a perdu une poursuite parce qu'il introduisait trop d'applications directement dans son OS, et limitait la compétition avec son monopole. Mais aujourd'hui, ça redevient de plus en plus présent...



Computer Startup

- **Programme de bootstrap** est chargé au démarrage
 - Habituellement stocké en mémoire Flash ou ROM (read-only memory), généralement dénommé **firmware**
 - Initialise tout le matériel du système
 - Charge le noyau du système d'exploitation et lance son exécution



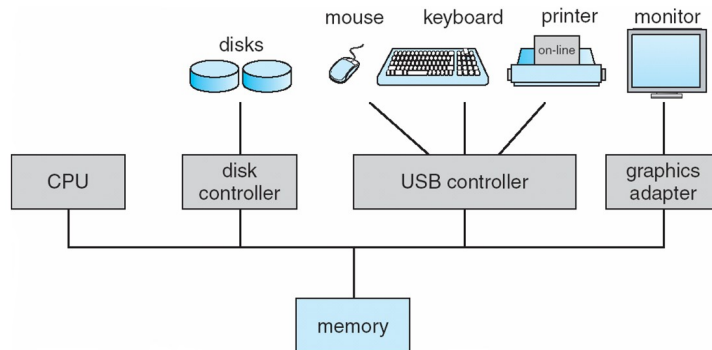
Dans le cours d'architecture?

Panneau de leviers binaires du PDP-11 a ULaval



Organisation d'un système informatique

- Opération d'un système informatique
 - Les CPUs et contrôleurs de périphériques sont connectés par un bus commun donnant accès à une mémoire partagée
 - Exécution concurrente des CPUs and des périphériques qui luttent pour accéder au bus et à la mémoire



La memoire a aussi un controleur



Opération d'un système informatique

- CPUs et périphériques d'E/S peuvent exécuter concurremment
- Chaque contrôleur de périphérique se charge d'un type particulier de périphérique
- Chaque contrôleur de périphérique a un tampon local
- CPU déplace les données de/vers la mémoire centrale vers/de les tampons locaux
- Les E/S se font entre le périphérique et le tampon local du contrôleur
- Le contrôleur de périphérique informe le CPU qu'il a terminé une opération en signalant une **interruption**





Interruptions

- Une interruption cause un transfert de contrôle à la routine de service d'interruptions, généralement via un **vecteur d'interruptions**, qui contient les adresses des différentes routines de service
- Le CPU sauve l'adresse de l'instructions interrompue
- Un **"trap"** ou une **exception** est une sorte d'interruption générée par software causée par une erreur ou une requête
- Un système d'exploitation est **guidé par les interruptions**





Gestion des interruptions

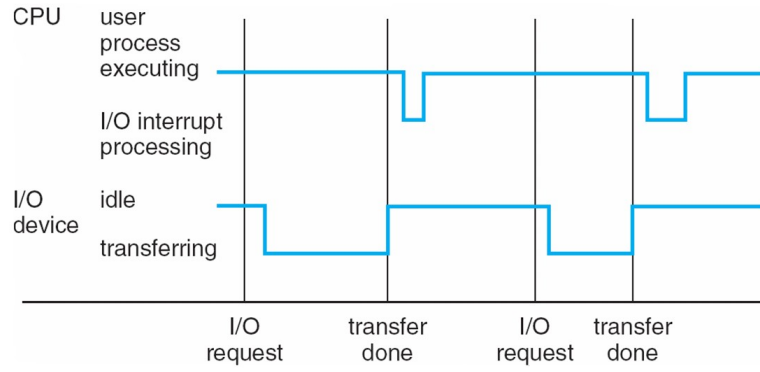
- Le système d'exploitation préserve l'état antérieur du CPU, tel que le contenu des registres
- Détermine quel type d'interruption est arrivée
 - Par scrutation (**Polling**)
 - Par **vecteur d'interruption**
- Différent segments de code déterminent l'action à prendre pour chaque type d'interruption



polling attend (busy-waiting) jusqu'à ce que le périphérique réponde



Ligne de temps d'une interruption





Structure des E/S

- Après le début d'une E/S, le contrôle retourne au programme seulement quand l'E/S est complétée:
 - Des instructions d'attente mettent le CPU en pause jusqu'à la prochaine interruption
 - Boucle d'attente active
 - Pas plus d'une requête d'E/S en attente à la fois, pas d'E/S simultanées

- Après le début d'une E/S, le contrôle retourne au programme sans attendre la fin de l'E/S:
 - **Appels systèmes** – requêtes au SE pour permettre au programme d'attendre la fin de l'E/S
 - **Table d'état des périphériques** contient le type, l'adresse, et l'état de chaque périphérique





Storage Definitions and Notation Review

bit. Un bit contient soit 0 soit 1. Tout stockage est basé sur une collection de bits.

Un **byte** = 8 bits, et pour la majorité des systèmes, c'est la plus petite unité facile à manipuler.

Par exemple, la majorité des CPU n'a pas d'instruction pour déplacer un bit, mais en a pour déplacer un byte.

b=bit; **B**=byte

Un **mot** est l'unité "naturelle" d'un système informatique. E.g. 32bit.

Un **kilobyte** (ou **KB**) = 1,024 bytes

Un **megabyte** (or **MB**) = $1,024^2$ bytes

Un **gigabyte** (ou **GB**) = $1,024^3$ bytes

Un **terabyte** (ou **TB**) = $1,024^4$ bytes

Un **petabyte** (ou **PB**) = $1,024^5$ bytes

Il est courant d'arrondir ces nombres en milliers, et de dire par exemple qu'un megabyte est 1 million de bytes and a gigabyte is 1 billion bytes. Les vitesses de communication sont souvent mesurées en multiple de bits plutôt que de bytes.





Structure du DMA

DMA = Accès Direct à la Mémoire

- Utilisé pour les périphériques rapides, capables de transférer des données à haut débit
- Le contrôleur de DMA transfère des blocs de données directement de/vers la mémoire centrale sans intervention du CPU
- Une seule interruption par transfert plutôt qu'une pour chaque byte



DMA



Structures de stockage

- Mémoire centrale – seul stockage auquel le CPU peut accéder directement
 - Accès aléatoire
 - Typiquement **volatile**
- Mémoire de masse – extension de la mémoire centrale qui offre une plus grande capacité et n'est **pas volatile**
- Disque magnétique – plateaux de metal ou verre recouvert d'un matériau magnétique d'enregistrement
 - La surface est divisée en **pistes**, elles mêmes subdivisées en **secteurs**
 - Le **contrôleur de disque** détermine l'interaction entre le périphérique et l'ordinateur
- **disques SSD** – plus rapides que les disques magnétiques mais quand même nonvolatils
 - Diverses technologies
 - Très populaire de nos jours



solid-state drive: flash (coûts diminuent, mais encore 7-8x plus chers que disques)



Hierarchie de stockage

- Les systèmes de stockage sont organisés en hiérarchie
 - Vitesse
 - Coût
 - Volatilité

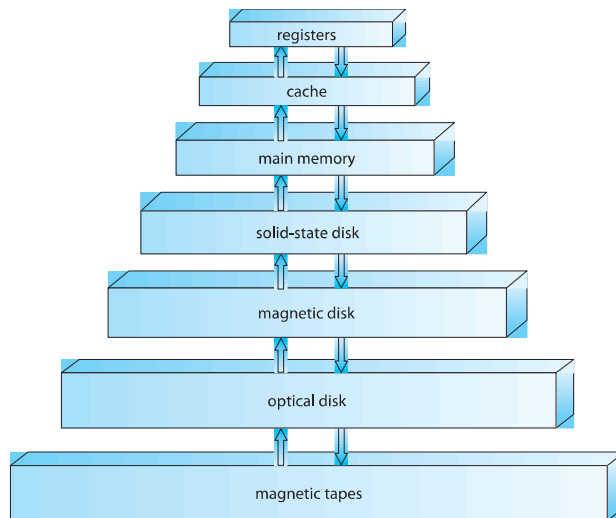
- **Caching** – copier l'information dans un système de stockage plus rapide; la mémoire centrale peut être utilisée comme un cache de la mémoire de masse

- **Pilote de périphérique** pour chaque contrôleur de périphérique pour gérer les E/S
 - Fournir une interface uniforme entre le contrôleur et le noyau





Hiérarchie de stockage





Caching

- Principe important, utilisé à toute sortes d'endroits et de niveaux dans un ordinateur (en matériel, par le SE, par les applications)
- Copier l'information active vers une mémoire plus rapide
- Vérifier d'abord si la donnée est dans le cache
 - Si elle y est, on peut l'utiliser directement (rapide)
 - Si non, on la copie d'abord et on l'utilise après
- Pour être plus rapide, le cache est plus petit
 - Le design de la gestion du cache est très important
 - Taille du cache et règles de remplacement





Architecture des systèmes

- Beaucoup de système n'ont qu'un seul CPU
 - La majorité ont aussi des processeurs specialisés
- Les systèmes **multiprocesseurs** sont toujours plus courants et importants
 - Aussi nommés **systèmes parallèles**, **systèmes fortement couplés**
 - Avantages:
 1. **Plus haute performance**
 2. **Économies d'échelle**
 3. **Meilleure fiabilité – dégradation progressive, tolérance aux pannes**
 - Deux types:
 - 4 **Multiprocessing asymétrique** (un CPU gère les autres)
 - 4 **Multiprocessing symétrique** (tous les CPUs sont égaux)

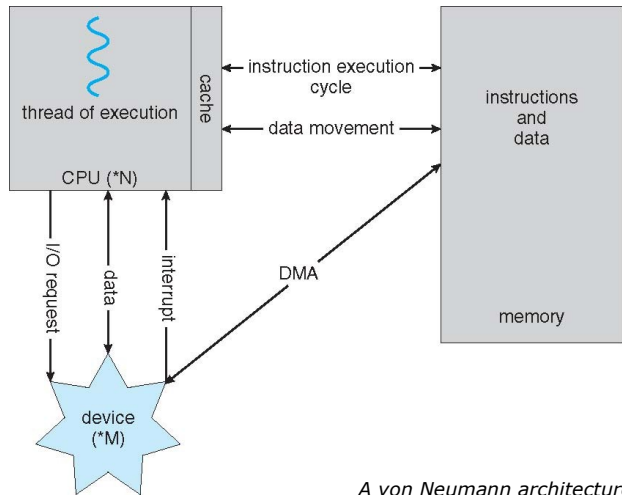


N processeurs accomplissent moins qu'un facteur de speed-up de n: synchronisation, partage ressources

Asymétrique désigne un boss-CPU, symétrique indique tous les CPUs sont égaux, et le plus populaire



Comment fonctionne un ordinateur

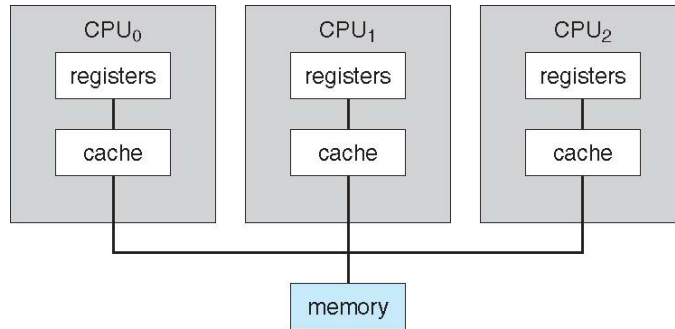


A von Neumann architecture





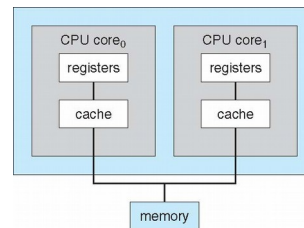
Architecture multiprocesseur symétrique





Un système dual-core

- Variations **UMA** et **NUMA**
(accès mémoire uniforme et nonuniforme)
- Multi-chip ou **multicore**
- Systèmes contenant tout les chips vs **serveurs blade**
 - Chassis contenant plusieurs systèmes séparés (avec leur OS, I/O, réseau, etc.)



Uniform memory access, prend le meme temps d'accès, alors que nonuniform MA non
Communication sur multi-chip est plus longue que sur multi-cores sur un meme chip,
et multicore demande moins d'electricite

Multicore est multiproc, mais pas tous les multiprocs sont multicores

Blade servers: chaque board a son propre OS, I/O, reseaux, etc.



Systemes en grappes

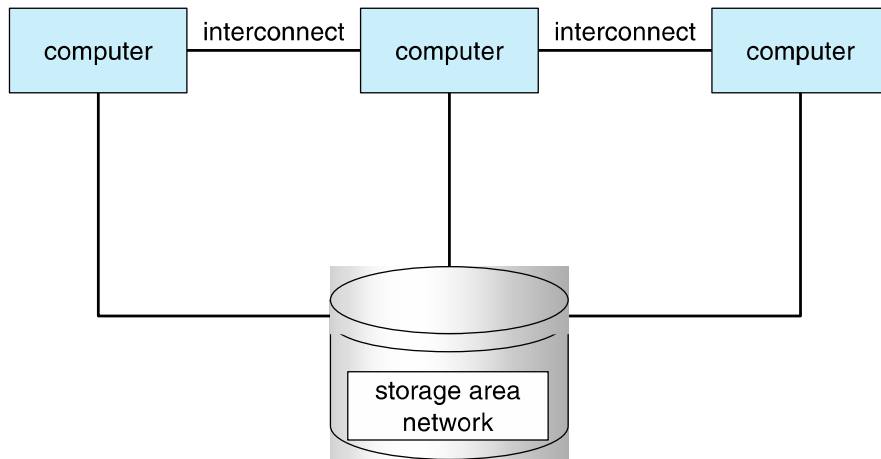
- Comme systemes multiprocesseurs, mais multiples systemes travaillant ensemble
 - Partage du stockage via un **storage-area network (SAN)**
 - Offre un service **high-availability** qui survit aux pannes
 - **Grappes asymmetriques** avec une machine en hot-standby
 - **Grappes symmetriques** où les noeuds se surveillent mutuellement
 - Grappes de calcul **high-performance computing (HPC)**
 - Applications ecrites pour utiliser la **parallélisation**
 - Usage de **distributed lock manager (DLM)** pour éviter les conflits dans certaines operations



Hot-standby mode: moniteur les autres



Systemes en grappes





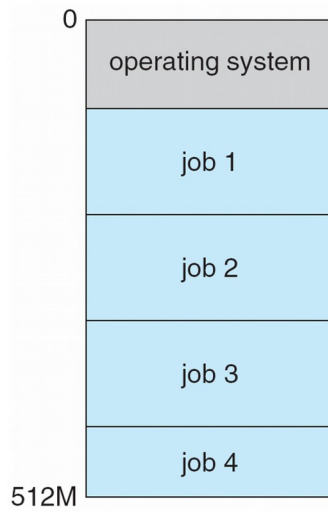
Structure des systèmes d'exploitation

- Besoin de **multiprogrammation** pour l'efficacité
 - Un seul utilisateur ne peut garder le CPU et les E/S occupées
 - Multiprogrammation organise le travail (code and data) pour que le CPU aie toujours quelque chose à faire
 - Un sous ensemble des tâches gardé en mémoire
 - Choix d'exécution des tâches par **job scheduling**
 - En cas d'attente (par exemple pour E/S), SE passe à une autre tâche
- **Timesharing (multitasking)** une extension naturelle où le CPU change de tâche si souvent que l'utilisateur peut interagir avec chacune d'elle: **interactive computing**
 - Temps de réponse devrait être < 1 second
 - Chaque utilisateur a au moins une tâche en mémoire □ **process**
 - Plusieurs tâches prêtes en même temps □ **CPU scheduling**
 - Tâches occupent plus de mémoire que disponible: **swapping** sur le disque
 - **Mémoire virtuelle** permet l'exécution de tâches partiellement en mémoire





Layout mémoire de système multiprogrammé





Opérations de systèmes d'exploitation

- **Guidé par les interruptions** par le matériel
- Erreur logicielle et requêtes créent des **exceptions ou trap**
 - Division par zéro, appel système
- Éviter les problèmes tels que les boucles infinies, les processus qui en modifient d'autres ou qui modifient le SE lui-même
- Opération en **deux modes** permet au SE de protéger les éléments
 - **Mode utilisateur** et **mode noyau**
 - **Bit de mode** fourni par le processeur
 - Permet de distinguer l'exécution dans le noyau
 - Les instructions "**privilegées**", autorisées seulement en mode noyau
 - Interruption et exceptions passent en mode noyau

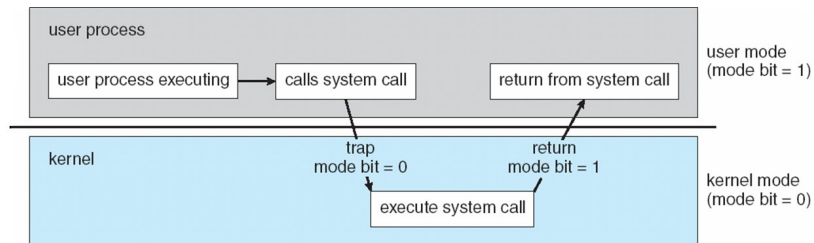


L'instruction pour switcher en mode kernel est une instruction privileged
VMM a besoin de plus de bits mode pour gerer ses propres processus



Transition au mode noyau

- Pour éviter les boucles sans fin, le noyau doit garder un œil ouvert
 - Utiliser un timer pour générer une interruption dans le futur
 - Avant de retourner en mode user, le SE arme le timer
 - Quand le timer sonne, l'interruption redonne la main au noyau



Instructions qui ont change le timer sont en mode privileged



Gestion des processus

- Un processus est un programme en cours d'exécution. Un programme est une **entité passive**, un processus est une **entité active**.
- Un processus a besoin de ressources pour accomplir sa tâche
 - CPU, mémoire, E/S, fichiers
 - Initialisation des données
- À la fin d'un processus, il faut récupérer ces ressources
- Un processus "single-threaded" a un **program counter** qui indique l'adresse de la prochaine instruction à exécuter
 - Le processus exécute ses instructions séquentiellement, une à la fois, jusqu'à épuisement
- Un processus "multi-threaded" a plusieurs program counters, un par thread, qui peuvent ainsi exécuter en parallèle
- Habituellement un système a toujours beaucoup de processus actifs, certains utilisateurs, d'autres appartenant au système, qui fonctionnent tous concurremment sur un ou plusieurs CPUs
 - Multiplexer les CPUs parmi les processus / threads





Activités de gestion des processus

Le système d'exploitation est responsable de:

- Créer et détruire les processus utilisateurs et système
- Suspendre et réveiller les processus
- Fournir des mécanismes pour la synchronisation entre processus
- Fournir des mécanismes pour la communication entre processus
- Fournir des mécanismes pour la gestion des étreintes mortelles





Gestion mémoire

- Avoir les données en mémoire avant de les manipuler
- Avoir les instructions en mémoire avant de les exécuter
- La gestion mémoire détermine quoi garder en mémoire pour
 - Optimiser l'usage du CPU et le temps de réponse à l'utilisateur
- Activités de gestion mémoire
 - Garder trace de qui utilise quelle partie de la mémoire
 - Decider quelles informations de quels processus transférer de/vers la mémoire
 - Allouer et désallouer les espaces mémoire au besoin





Gestion du stockage

- Se fournit une vue uniforme des capacités de stockage
 - Abstraires les détails physiques en unités logiques – les **fichiers**
 - Each medium is controlled by device (i.e., disk drive, tape drive)
 - Varying properties include access speed, capacity, data-transfer rate, access method (sequential or random)
- File-System management
 - Files usually organized into directories
 - Access control on most systems to determine who can access what
 - OS activities include
 - Creating and deleting files and directories
 - Primitives to manipulate files and directories
 - Mapping files onto secondary storage
 - Backup files onto stable (non-volatile) storage media





Mass-Storage Management

- Usually disks used to store data that does not fit in main memory or data that must be kept for a “long” period of time
- Proper management is of central importance
- Entire speed of computer operation hinges on disk subsystem and its algorithms
- OS activities
 - Free-space management
 - Storage allocation
 - Disk scheduling
- Some storage need not be fast
 - Tertiary storage includes optical storage, magnetic tape
 - Still must be managed – by OS or applications
 - Varies between WORM (write-once, read-many-times) and RW (read-write)





Performance of Various Levels of Storage

	128GB	1TB	4TB	DIRO	
Level	1	2	3	4	5
Name	registers	cache	main memory	solid state disk	magnetic disk
Typical size	< 1 KB	< 16MB	< 64GB	< 1 TB	< 10 TB
Implementation technology	custom memory with multiple ports CMOS	on-chip or off-chip CMOS SRAM	CMOS SRAM	flash memory	magnetic disk
Access time (ns)	0.25 - 0.5	0.5 - 25	80 - 250	25,000 - 50,000	5,000,000
Bandwidth (MB/sec)	20,000 - 100,000	5,000 - 10,000	1,000 - 5,000	500	20 - 150
Managed by	compiler	hardware	operating system	operating system	operating system
Backed by	cache	main memory	disk	disk	disk or tape

- Movement between levels of storage hierarchy can be explicit or implicit



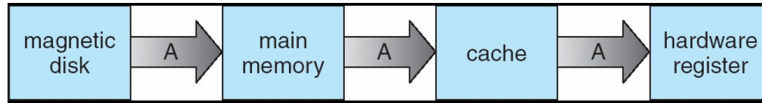
transfert explicite: cache vers CPU/registres ne requiert pas le OS

transfert implicite: disque vers memoire est gere par le OS



Migration of Integer A from Disk to Register

- Multitasking environments must be careful to use most recent value, no matter where it is stored in the storage hierarchy



- Multiprocessor environment must provide **cache coherency** in hardware such that all CPUs have the most recent value in their cache
- Distributed environment situation even more complex
 - Several copies of a datum can exist
 - Various solutions covered in Chapter 17





I/O Subsystem

- One purpose of OS is to hide peculiarities of hardware devices from the user

- I/O subsystem responsible for
 - Memory management of I/O including
 - ▶ buffering: storing data temporarily while it is being transferred
 - ▶ caching: storing parts of data in faster storage for performance
 - ▶ spooling: the overlapping of output of one job with input of other jobs
 - General device-driver interface
 - Drivers for specific hardware devices





Protection and Security

- **Protection** – any mechanism for controlling access of processes or users to resources defined by the OS
- **Security** – defense of the system against internal and external attacks
 - Huge range, including denial-of-service, worms, viruses, identity theft, theft of service
- Systems generally first distinguish among users, to determine who can do what
 - User identities (**user IDs**, security IDs) include name and associated number, one per user
 - User ID then associated with all files, processes of that user to determine access control
 - Group identifier (**group ID**) allows set of users to be defined and controls managed, then also associated with each process, file
 - **Privilege escalation** allows user to change to effective ID with more rights

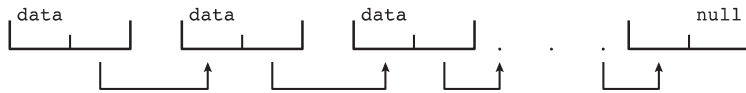




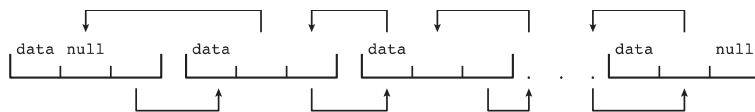
Kernel Data Structures

- Many similar to standard programming data structures

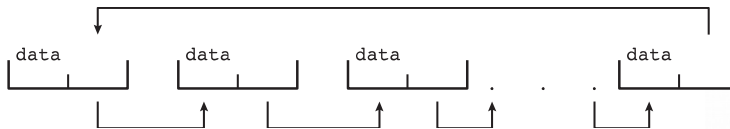
- **Singly linked list**



- **Doubly linked list**



- **Circular linked list**



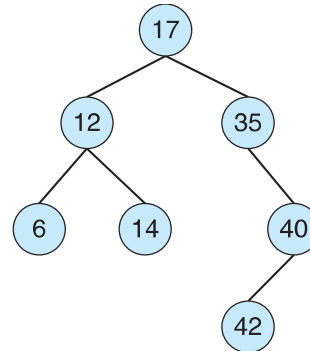


Kernel Data Structures

■ Binary search tree

left \leq right

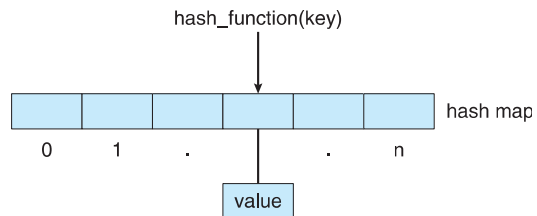
- Search performance is $O(n)$
- **Balanced binary search tree** is $O(\log n)$





Kernel Data Structures

- **Hash function** can create a **hash map**



- **Bitmap** – string of n binary digits representing the status of n items
- Linux data structures defined in **include** files
`<linux/list.h>`, `<linux/kfifo.h>`,
`<linux/rbtree.h>`





Computing Environments - Traditional

- Stand-alone general purpose machines
- But blurred as most systems interconnect with others (i.e., the Internet)
- **Portals** provide web access to internal systems
- **Network computers** (**thin clients**) are like Web terminals
- Mobile computers interconnect via **wireless networks**
- Networking becoming ubiquitous – even home systems use **firewalls** to protect home computers from Internet attacks





Computing Environments - Mobile

- Handheld smartphones, tablets, etc.
- What is the functional difference between them and a “traditional” laptop?
- Extra feature – more OS features (GPS, gyroscope)
- Allows new types of applications like **augmented reality**
- Use IEEE 802.11 wireless, or cellular data networks for connectivity
- Leaders are **Apple iOS** and **Google Android**



La consommation de la batterie est un soucis serieux en mobile



Computing Environments – Distributed

- Distributed
 - Collection of separate, possibly heterogeneous, systems networked together
 - **Network** is a communications path, **TCP/IP** most common
 - **Local Area Network (LAN)**
 - **Wide Area Network (WAN)**
 - **Metropolitan Area Network (MAN)**
 - **Personal Area Network (PAN)**
 - **Network Operating System** provides features between systems across network
 - Communication scheme allows systems to exchange messages
 - Illusion of a single system

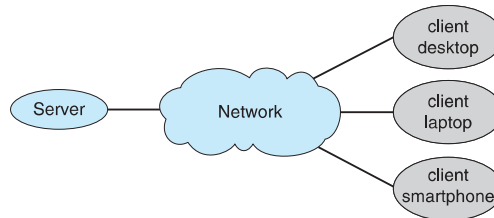


PAN: un cell et les ecouteurs, ou un cell et un PC



Computing Environments – Client-Server

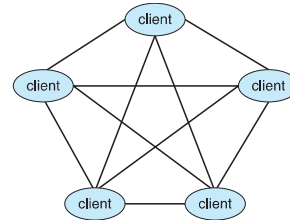
- Client-Server Computing
 - Dumb terminals supplanted by smart PCs
 - Many systems now **servers**, responding to requests generated by **clients**
 - ▶ **Compute-server system** provides an interface to client to request services (i.e., database)
 - ▶ **File-server system** provides interface for clients to store and retrieve files





Computing Environments - Peer-to-Peer

- Another model of distributed system
- P2P does not distinguish clients and servers
 - Instead all nodes are considered peers
 - May each act as client, server or both
 - Node must join P2P network
 - Registers its service with central lookup service on network, or
 - Broadcast request for service and respond to requests for service via **discovery protocol**
 - Examples include Napster and Gnutella, **Voice over IP (VoIP)** such as Skype



Skype a un serveur central pour se connecter/identifier, mais permet a deux peers de communiquer



Computing Environments - Virtualization

- Allows operating systems to run applications within other OSes
 - Vast and growing industry
- **Emulation** used when source CPU type different from target type (i.e. PowerPC to Intel x86)
 - Generally slowest method
 - When computer language not compiled to native code – **Interpretation**
- **Virtualization** – OS natively compiled for CPU, running **guest** OSes also natively compiled
 - Consider VMware running WinXP guests, each running applications, all on native WinXP **host** OS
 - **VMM** provides virtualization services



VMM: virtual machine manager



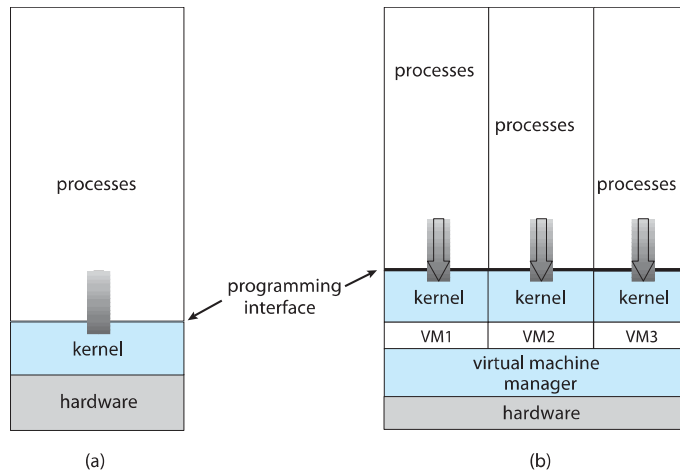
Computing Environments - Virtualization

- Use cases involve laptops and desktops running multiple OSes for exploration or compatibility
 - Apple laptop running Mac OS X host, Windows as a guest
 - Developing apps for multiple OSes without having multiple systems
 - QA testing applications without having multiple systems
 - Executing and managing compute environments within data centers
- VMM can run natively, in which case they are also the host
 - There is no general purpose host then (VMware ESX and Citrix XenServer)





Computing Environments - Virtualization





Computing Environments – Cloud Computing

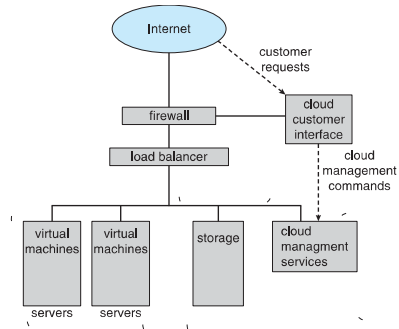
- Delivers computing, storage, even apps as a service across a network
- Logical extension of virtualization as based on virtualization
 - Amazon **EC2** has thousands of servers, millions of VMs, PBs of storage available across the Internet, pay based on usage
- Many types
 - **Public cloud** – available via Internet to anyone willing to pay
 - **Private cloud** – run by a company for the company's own use
 - **Hybrid cloud** – includes both public and private cloud components
 - Software as a Service (**SaaS**) – one or more applications available via the Internet (i.e. word processor)
 - Platform as a Service (**PaaS**) – software stack ready for application use via the Internet (i.e. a database server)
 - Infrastructure as a Service (**IaaS**) – servers or storage available over Internet (i.e. storage available for backup use)





Computing Environments – Cloud Computing

- Cloud compute environments composed of traditional OSes, plus VMs, plus cloud management tools
 - Internet connectivity requires security like firewalls
 - Load balancers spread traffic across multiple applications





Computing Environments – Real-Time Embedded Systems

- Real-time embedded systems most prevalent form of computers
 - Vary considerably, special purpose, limited purpose OS, **real-time OS**
 - Use expanding
- Many other special computing environments as well
 - Some have OSes, some perform tasks without an OS
- Real-time OS has well-defined fixed time constraints
 - Processing **must** be done within constraint
 - Correct operation only if constraints met





Open-Source Operating Systems

- Operating systems made available in source-code format rather than just binary **closed-source**
- Counter to the **copy protection** and **Digital Rights Management (DRM)** movement
- Started by **Free Software Foundation (FSF)**, which has “copyleft” **GNU Public License (GPL)**
- Examples include **GNU/Linux** and **BSD UNIX** (including core of **Mac OS X**), and many more
- Can use VMM like VMware Player (Free on Windows), Virtualbox (open source and free on many platforms - <http://www.virtualbox.com>)
 - Use to run guest operating systems for exploration



Mac OS X et iOS ont un open-source kernel, Darwin, mais aussi des composantes privées

End of Chapter 1

