Chapitre 1 : Introduction aux Systèmes d'Exploitation

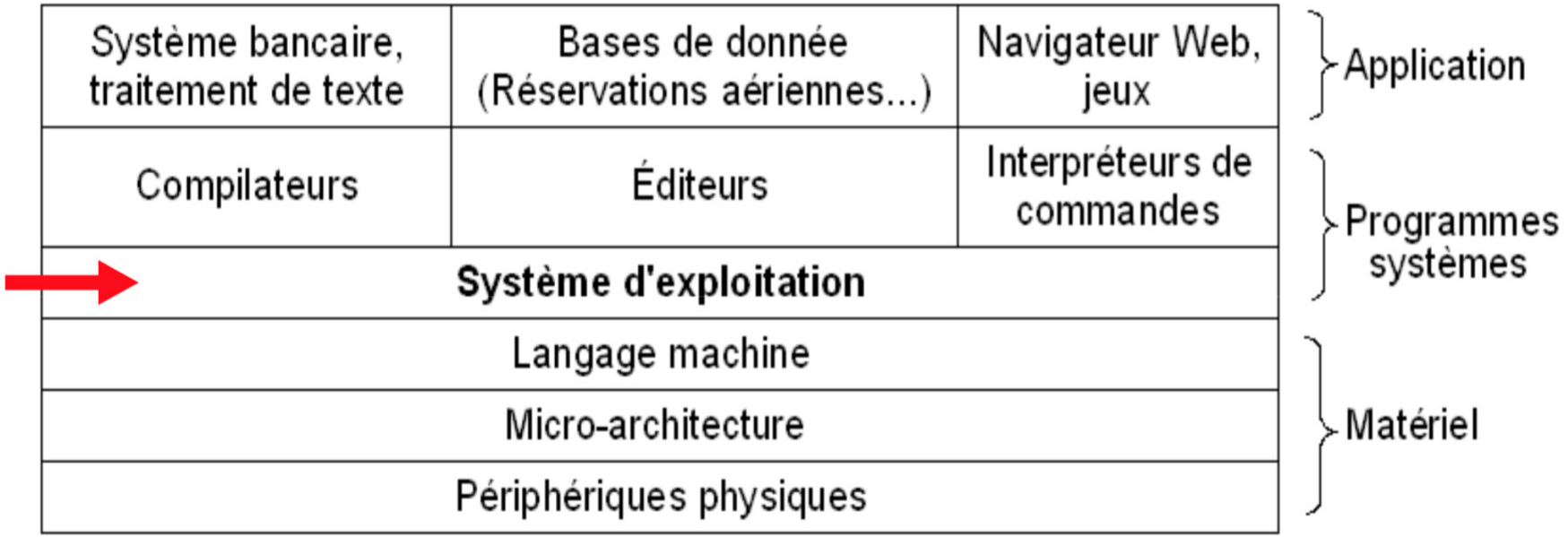
Dr Mandicou BA
mandicou.ba@esp.sn
http://www.mandicouba.net

Diplôme D'Ingénieur de Conception (DIC, 1^{e} année)
en Informatique / Télécommunications-Réseaux
Licence Professionnelle
en Génie Logiciel et Systèmes d'Information (GLSI)

Plan du Chapitre

Sommaire

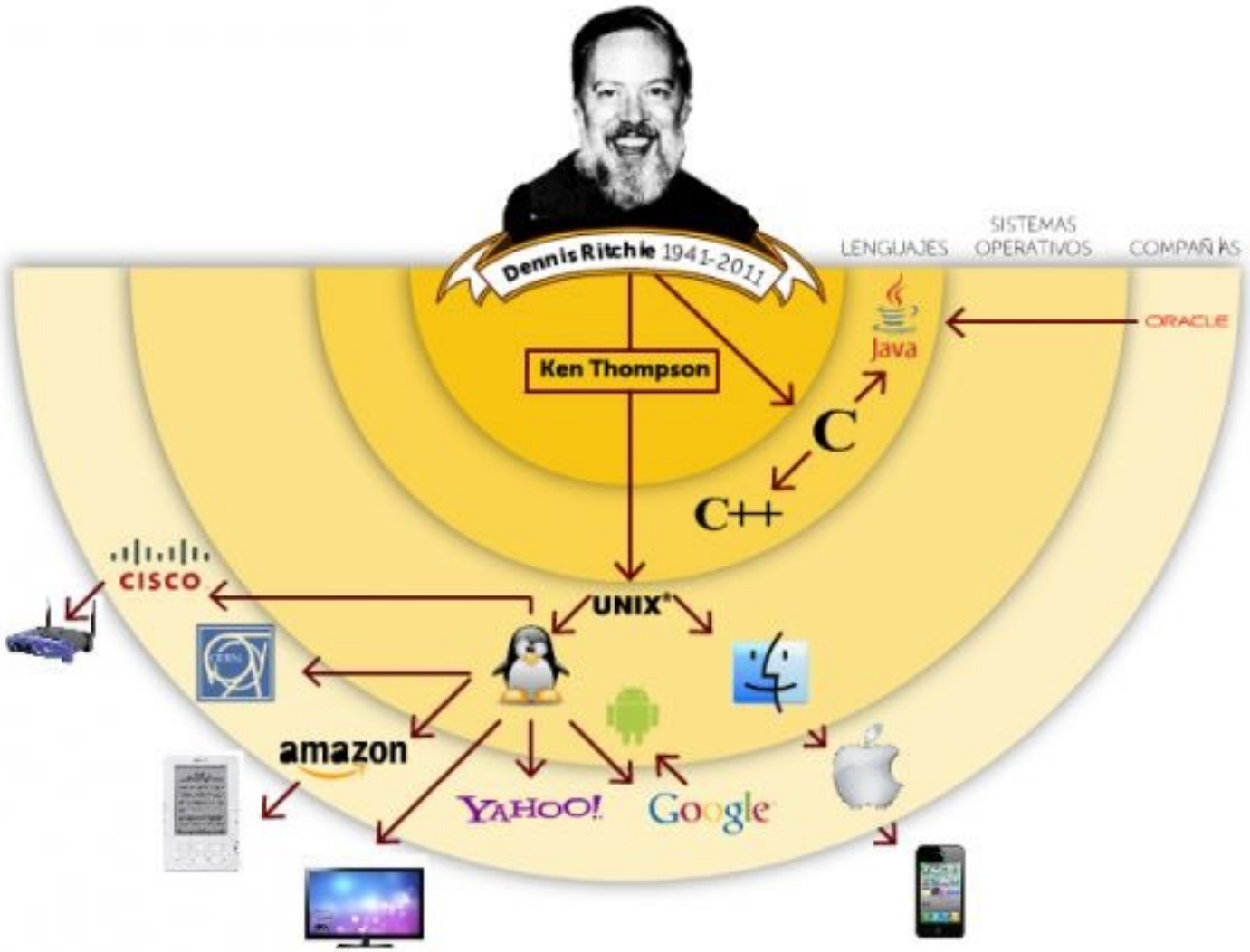
Structuration et Rôles des SE



Positionnement du système d'exploitation dans la couche des programmes qui contribuent à l'utilisation d'un ordinateur

Sommaire

Le cas du système d'exploitation UNIX : origine et évolution



Le cas du système d'exploitation UNIX

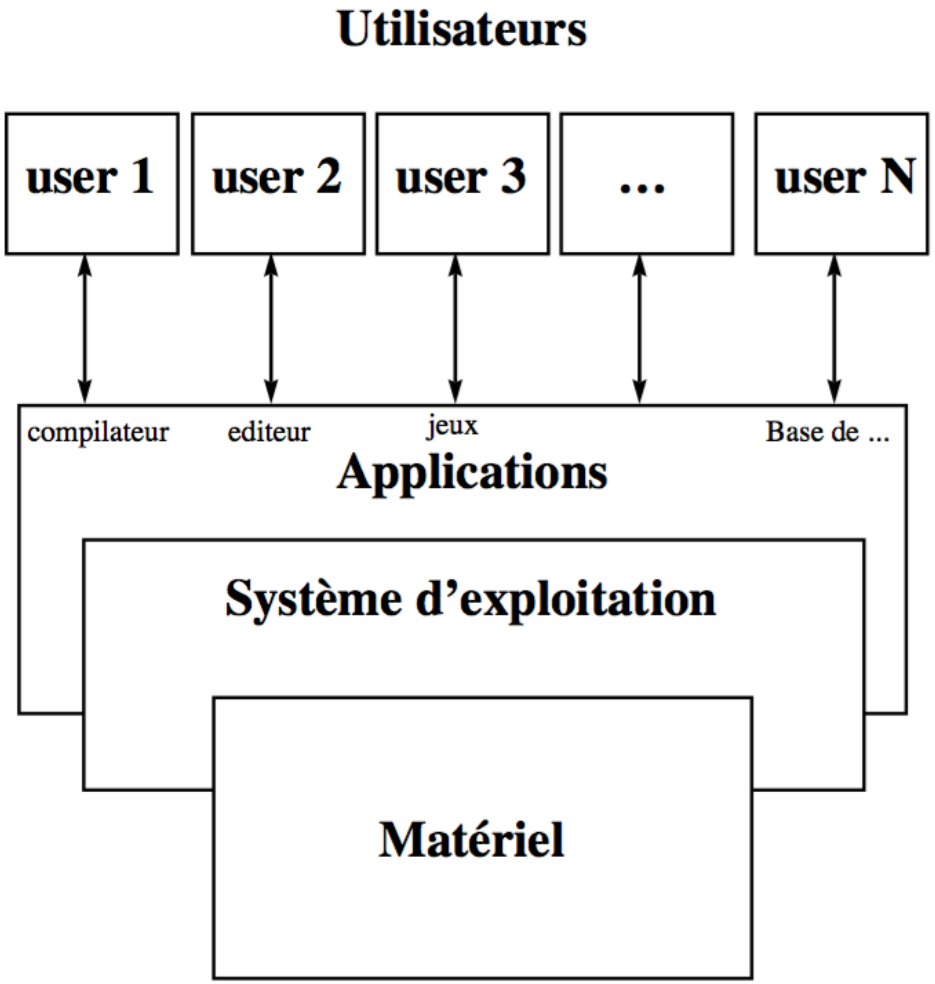


Figure: Vue générale du système

Le cas du système d'exploitation : UNIX

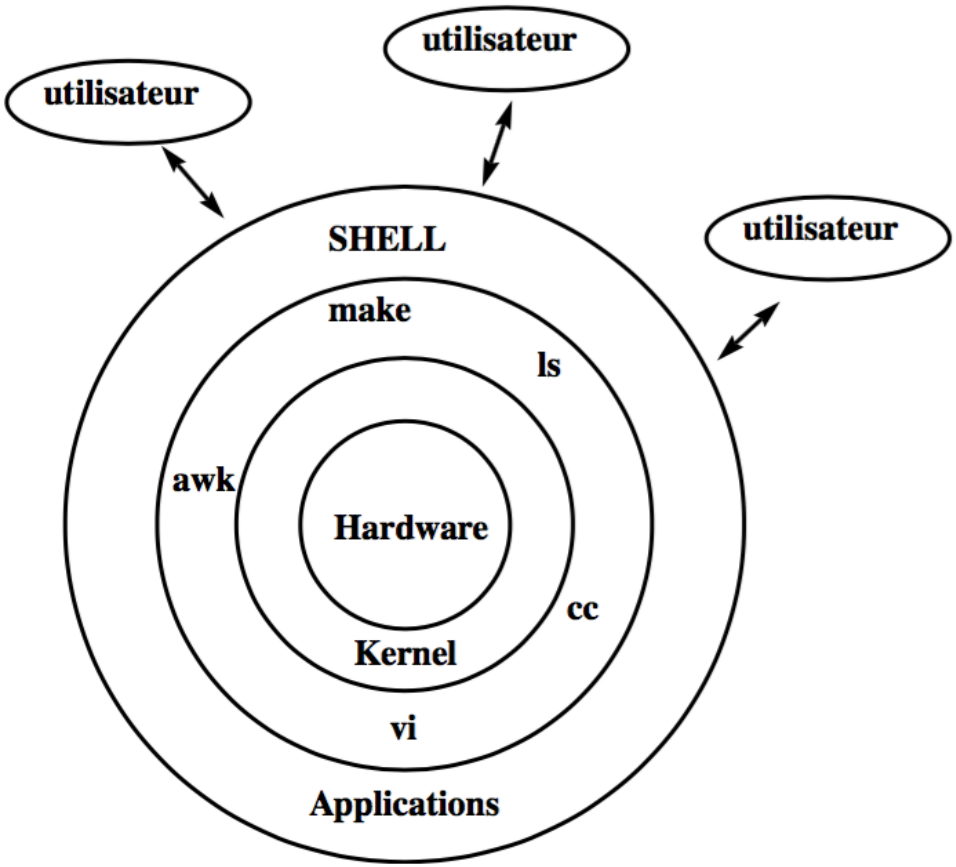


Figure: Point de vue utilisateur

Le cas du système d'exploitation : UNIX

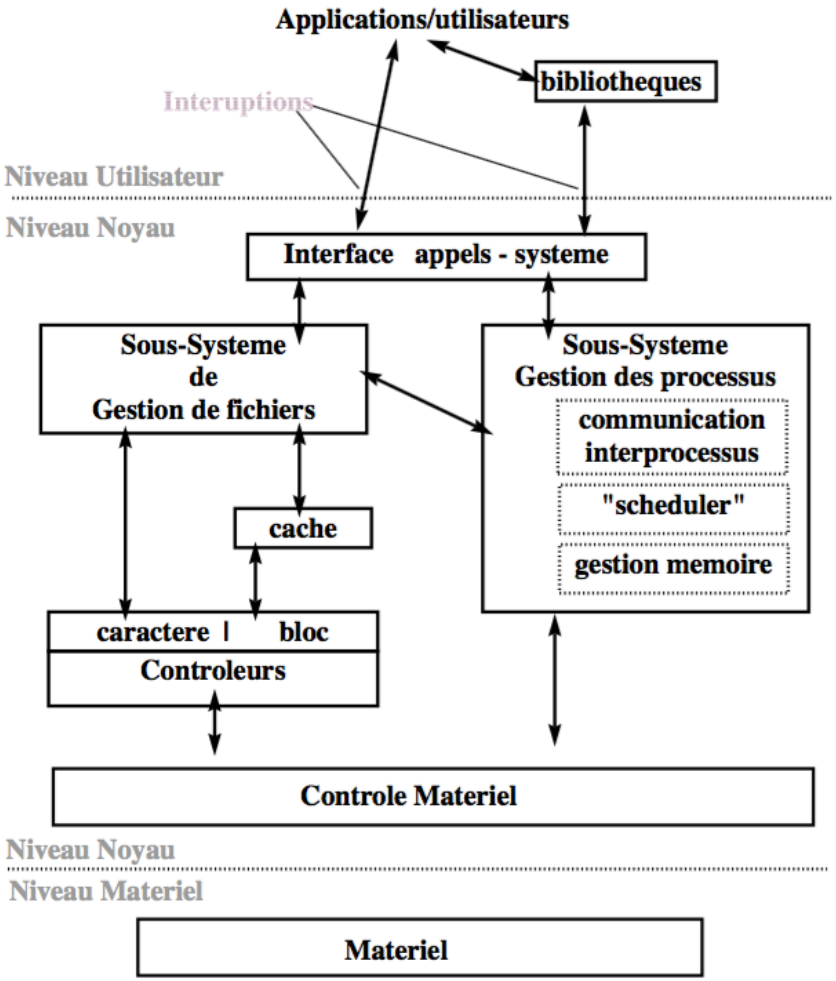


Figure: Architecture Noyau

Sommaire

Idée de base

- ☛ Faire croire à un programmeur C qu'il dispose de toute la machine, pour qu'il n'ait pas à changer sa façon de travailler
- ☛ pour donner cette illusion, le système doit disposer d'un type d'objet particulier: le processus (process)

Rappels sur la multi-programmation

Mono-programmation

- Un seul programme (processus) s'exécute sans interruption :
  - Si le processus contient une instruction d'E/S, restera inactif durant une « longue période » en attendant que cette instruction se termine

Multi-programmation

- Plusieurs programmes (processus) se partagent les ressources (mémoire, périphérique, etc.) de l'ordinateur :
  - problème de protection, de concurrence et contrôle
- Le processeur exécute un autre processus au lieu de restant inactif pendant tout le temps pris par l'instruction d'E/S du 1^e processus
- Cela donne à l'utilisateur que tous les processus s'exécutent en même temps : pseudo-parallélisme
  - sur une machine mono-processeur, un seul processus est exécuté à un instant donné

Processus

Définition 1

- Processus = instance d'exécution d'un programme crée par le SE ou l'utilisateur
  - il possède son compteur ordinal, ses registres et ses variables en mémoire
- Ne pas confondre **processus** et **programme** !
  - Un même programme peut avoir plusieurs exécutions simultanées
- Les processus sont protégés les uns des autres par le système
- Pour communiquer, ils doivent passer par des appels système (IPC)

- Composants d'un processus
  - le code (texte) : programme
  - les données : variables globales
  - pile d'exécution
  - registres internes

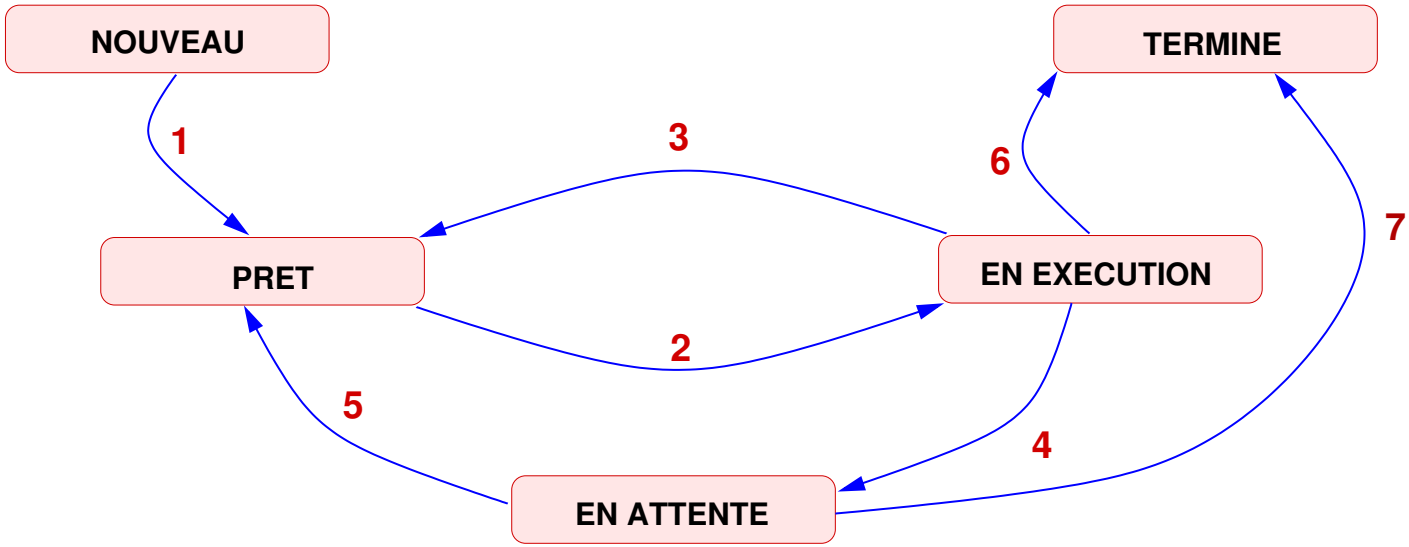
Les processus ça sert à quoi ?

- Faire plusieurs activités « en même temps ». Exemples :
  - Faire travailler plusieurs utilisateurs sur la même machine. Chaque utilisateur a l'impression d'avoir la machine à lui tout seul
  - Compiler tout en lisant son mail
- **Problème** : Un processeur ne peut exécuter qu'une seule instruction à la fois.
- L'exécution d'un processus doit progresser séquentiellement, i.e, à n'importe quel moment une seule instruction au plus est exécutée au nom du processus
- **BUT**: Partager un (ou plusieurs) processeur entre différents processus
- Attention!!! Ne pas confondre **processus** et **processeur**

États d'un processus

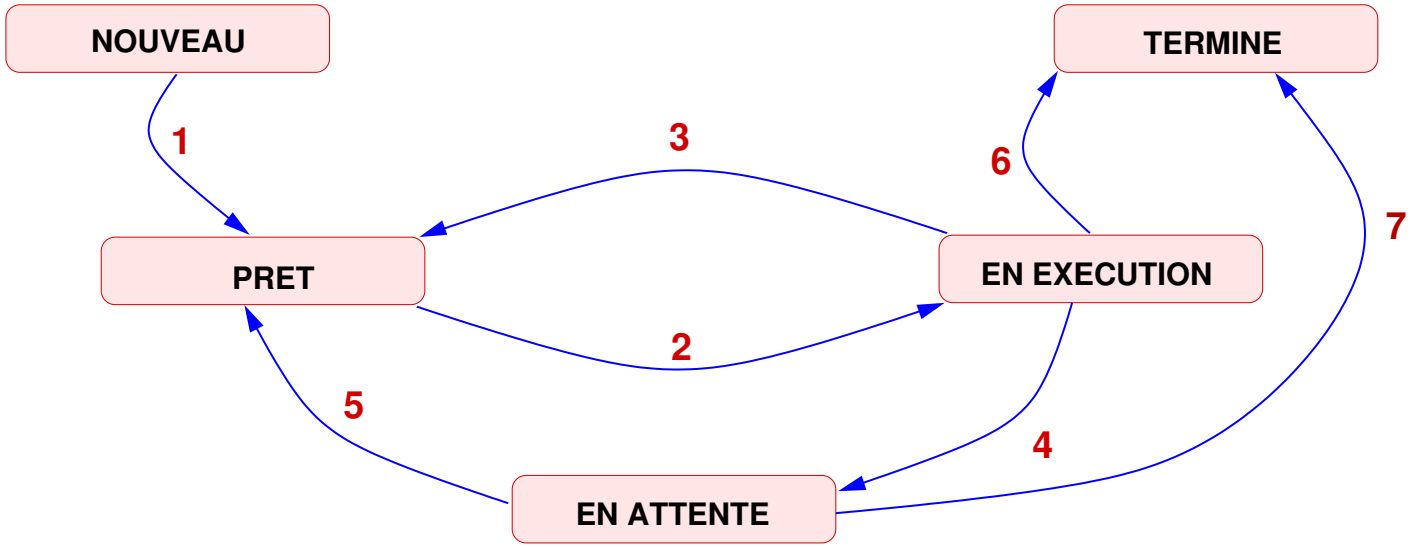
- Quand un processus s'exécute, il change d'état.
- Chaque processus peut se trouver dans chacun des états suivants :
  - En **exécution** ((process)): Les instructions sont en cours d'exécution (en train d'utiliser la CPU)
  - En **attente** : Le processus attend qu'un événement se produise.
  - **Prêt** : Le processus attend d'être affecté à un processeur.
- Un seul processus peut être en exécution sur n'importe quel processeur à tout moment.
- Toutefois, plusieurs processus peuvent être prêts et en attente

Transitions entre les états d'un processus 1/2



1. **Nouveau** $\mapsto$ **Prêt** : Tout nouveau processus créé est « admis d'abord dans la file d'attente des **prêts** »
2. « **Prêt** $\longmapsto$ **En Exécution (actif)** » : se produit quand le SE sélectionne le processus en question pour l'exécuter
3. « **En Exécution (actif)** $\longmapsto$ **Prêt** » : se produit si le **quantum** alloué est épuisé ou si un processus plus prioritaire (processus urgent ou processus système) arrive

Transitions entre les états d'un processus 2/2



4 « **En Exécution** $\longmapsto$ **En Attente (bloqué)** » : se produit quand le processus est en cours d'exécution et a besoin d'une ressource non disponible

5 « **En Attente** $\longmapsto$ **Prêt** » : se produit quand l'événement externe attendu par le processus se produit

6 « **En Attente** $\longmapsto$ **Terminé** » : se produit quand l'événement externe attendu par le processus ne peut se réaliser (ex. inter-blocage)

7 « **En Exécution** $\longmapsto$ **Terminé** » : Le processus à finis son exécution

Interruption d'un processus

Interruption

- Dans le cas des transitions « **actif** $\longmapsto$ **bloqué** » et « **actif** $\longmapsto$ **prêt** » on parle d'**interruption (IT)**

D'où viennent les IT

- Quand le processus à atteint une instruction d'E/S
- Le quantum attribué au processus est écoulé
- Un processus plus urgent doit être exécuté
- Un processus nécessite une ressource (matérielle ou logicielle) ou une donnée (un résultat calculé par un autre processus, ou un ensemble d'instructions qui ne sont pas encore chargées en chargé en mémoire) détenue par un autre processus (elle n'est pas encore disponible)

Interruption d'un processus

Traitement d'une IT

1. Arrivée d'une IT : Le processus en cours est interrompu et une gestionnaire d'IT est chargé dans les registres du processeurs et s'exécute pour traiter l'IT en question
2. Une fois le signal IT reconnu, le gestionnaire d'IT accède à la table des vecteurs d'IT et recherche l'adresse du programme associé et l'exécute
3. Une fois l'IT traitée, le SE charge un autre processus à partir de la file d'attente et l'exécute

Interruption d'un processus

Traitement d'une IT

1. Une IT est provoquée par un signal généré soit par un événement interne soit par un événement externe :
   - ☛ **Événement interne** : lié au processus
     - Appel système
     - Déroutement : dû généralement aux erreurs telles que division par zéro, débordement de la mémoire, exécution d'une instruction non autorisée, etc
   - ☛ **Événement externe** : panne, intervention de l'utilisateur à l'aide d'une frappe au clavier. C'est l'exemple de «Ctrl+Alt+supp», bouton «reset», etc.

Interruption d'un processus

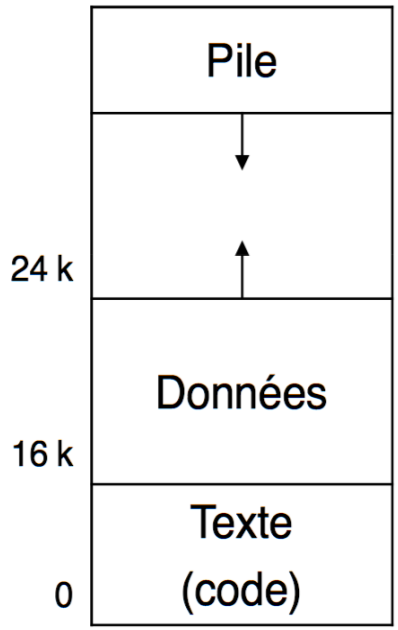
Traitement d'une IT

1. Deux sortes d'interruptions : Matérielles et Logicielles
   - ☛ **IT Matérielles (IRQ)** : générées par les périphériques. Parviennent au processeur par l'intermédiaire d'un contrôleur d'IT
   - ☛ **IT Logicielles** : des IT internes, c'est le processus qui appelle cette IT à l'aide du numéro d'IT.
     - : Par exemple, pour appeler une IT DOS, appeler l'IT N°21H
   - ☛ Si plusieurs interruptions arrivent au même temps, alors celle qui a le plus petit numéro qui a la plus grande priorité :
     - Exemple : IRQ horloge système = 0, IRQ port parallèle = 7

Espace mémoire d'un processus (1/5)

☛ Le mémoire utilisée par un processus est divisée en plusieurs zones, exactement 4.

1. Segment de code
2. Segment de données
3. Pile
4. Tas

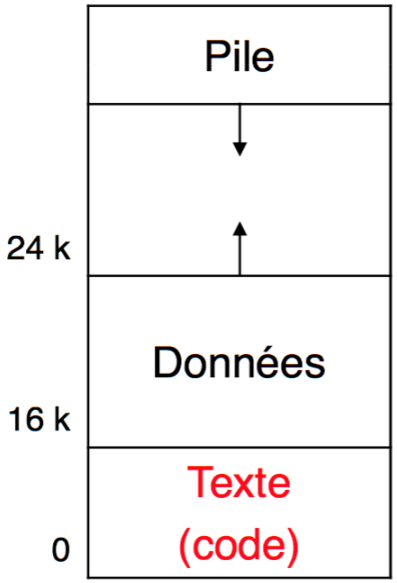


Structure de l'espace d'adressage d'un processus

Espace mémoire d'un processus (2/5)

Sagement de code

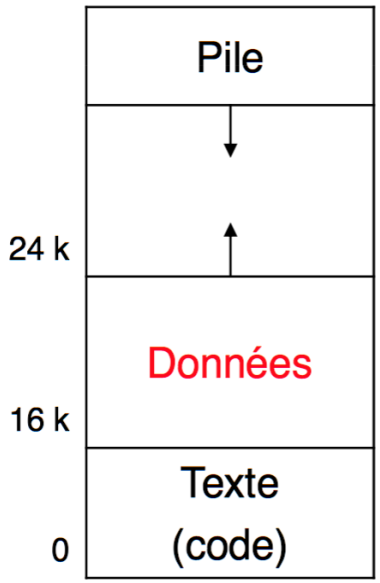
- Copie du segment de code du fichier exécutable
- Placé dans des zones fixes de la mémoire (début de la zone disponible)
- La prochaine instruction à exécuter dans ce segment est repérée par le pointeur d'instruction
- Cette zone est en lecture seule
- Elle peut être partagée par tous les programmes exécutant le même programme. Ce qui n'est pas le cas pour les segment de données et de pile



Espace mémoire d'un processus (3/5)

Sagement de données

- Il se trouve au dessus du segment de code
- Il est amené à grandir ou à rétrécir durant l'exécution
- il est composé de :
  1. **Un segment de données initialisées** : copié directement de l'exécution. Par exemple, les données initialisées correspondent aux variables globales et static initialisées d'un programme C
  2. **Un segment de données non initialisées** : crée dynamiquement. Les données non initialisées correspondent aux variables globales et static non initialisées



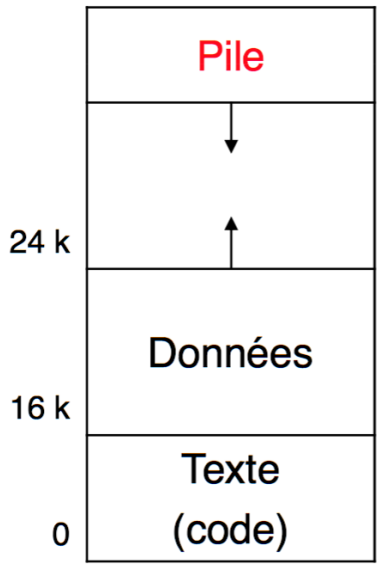
Espace mémoire d'un processus (4/5)

Pile

- Sert à stocker les données obtenues en cours d'exécution. Son nom, **pile**, (*stack en anglais*) vient de la manière dont elle est géré : empiler puis dépiler les données
- Le plus souvent en haut de l'espace d'adressage et croit vers le bas

Exemple 1 (Appel d'une fonction)

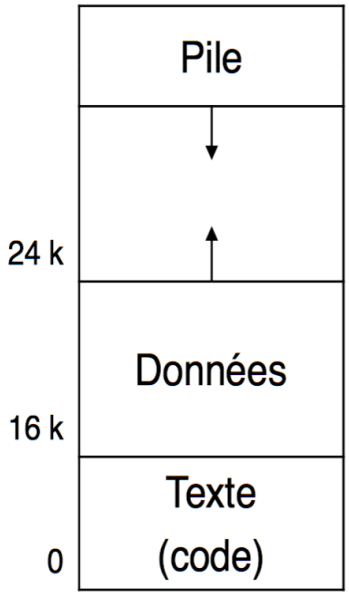
- *Empiler le nom de fonction, lui passer les paramètres et variables locales*
- *Exécuter la fonction. Une fois la fonction terminée, le SE dépile les données utilisées et retrouve les données d'avant*
- *Poursuivre l'exécution du programme*



Espace mémoire d'un processus (5/5)

Tas

☛ Est un autre segment utilisé par le système d'exploitation pour les allocations dynamiques.



Structure de données pour la gestion d'un processus 1/4

Structure de données

☛ Pour générer un processus, le SE manipule deux structures de données :

1. Le **bloc contexte** d'un processus
2. La **table des processus**

Structure de données pour la gestion d'un processus 2/4

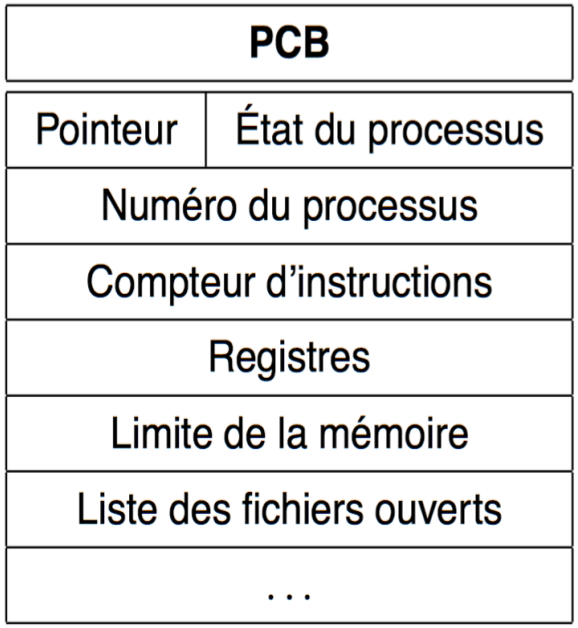
Contexte d'un processus

- Structures de données qui décrivent un processus en cours d'exécution
- info sauvegardées par le SE lors d'une IT d'un processus
- Crées au même temps que le processus et sont mises à jours lors d'une IT d'un processus
- Les données d'un contexte d'un processus sont :
  - le compteur ordinal adresse la prochaine instruction
  - le contenu des registres généraux
  - les registres d'occupation mémoire
  - le registre de variable d'état
  - la valeur d'horloge d'un processus
  - la priorité du processus

Structure de données pour la gestion d'un processus 3/4

Bloc de contrôle d'un processus

- Un processus est un programme en cours d'exécution dans un ordinateur.
- C'est une entité active avec son propre compteur ordinal (instruction pointer) et l'ensemble des ressources qui lui sont associées.
- Ces informations sont stockées, pour chaque processus dans le PCB (Process Control Block)



Structure de données pour la gestion d'un processus 4/4

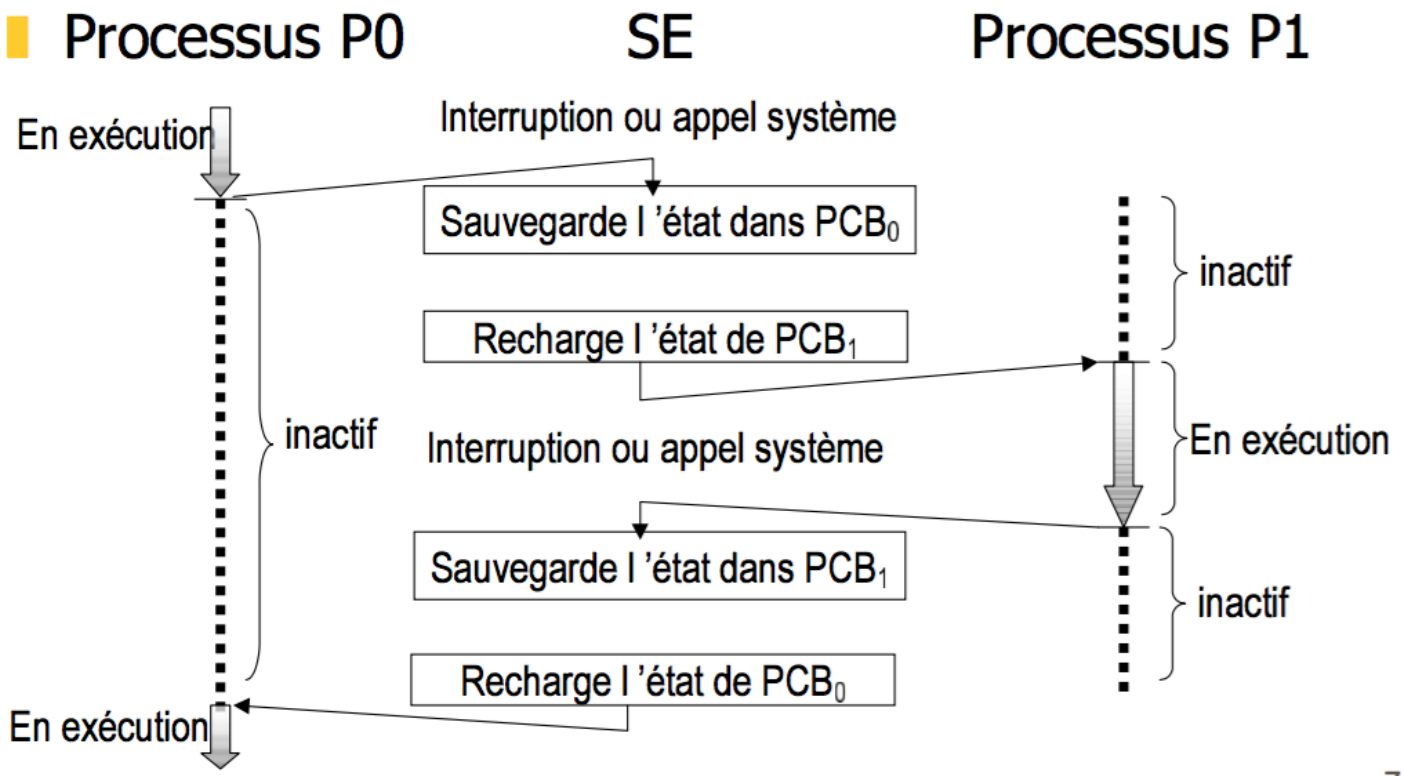
Table des processus

- ☛ Tout processus contient une entrée dans cette table
- ☛ Cette table contient toutes les informations indispensable au SE pour assurer une gestion cohérente des processus
- ☛ Ces informations sont :
  - Un pointeur vers le bloc de contexte du processus
  - l'identifiant du processus
  - son lien de parenté
  - les fichiers qu'il a ouvert
  - Occupation mémoire (pointeur sur le segment de code, de données et de pile)
- ☛ Cette table est stockée dans l'espace mémoire du SE
  - ☛ Donc **aucun processus ne peut y accéder**

Commutation de contexte (context switch) 1/2

- Lorsque le SE change le processus auquel est alloué le processeur, **une commutation de contexte** se produit.
- le SE doit sauvegarder le contexte du processus qui tourne sur le CPU afin d'etre en mesure de le restaurer par la suite
- Le contexte est représenté par le PCB du processus, et c'est lui qui est sauvegardé, puis restauré
- La commutation de contexte est une opération relativement coûteuse (quelques millisecondes, habituellement) et qui ne fait pas « avancer » les processus eux-mêmes
- Chaque processus dispose donc de son propre processeur virtuel
- La commutation entre les différents processus en cours d'exécution dans la mémoire donne cette « impression de virtualité »

Commutation de contexte (context switch) 2/2



☛ *Un **algorithme d'ordonnancement** détermine à quel moment il faut suspendre un processus et à quel moment en repartir un autre*

Chapitre 1 : Introduction aux Systèmes d'Exploitation

Dr Mandicou BA
mandicou.ba@esp.sn
http://www.mandicouba.net

Diplôme D'Ingénieur de Conception (DIC, 1^{e} année)
en Informatique / Télécommunications-Réseaux
Licence Professionnelle
en Génie Logiciel et Systèmes d'Information (GLSI)