Centre Universitaire de Mila

3 ème année licence LMD Informatique

Module : Systèmes d'exploitation 2

Bessouf Hakim

CHAPITRE 3:

Communication entre processus (Les IPC InterProcess Communication)

- Coopération de processus
- Communication entre processus (IPC : InterProcess Communication)
  - La mémoire partagé (shared memory region)
  - Le transfert de messages (Message passing)

Coopération de processus

Les système d'exploitation actuels permettent la coopération entre plusieurs processus, ceci offre plusieurs avantages on cite:

- **Partages d'informations** : les processus peuvent accéder d'une manière concurrente à une ressource partagé
- **Accélération des calculs** : Si par exemple on a plusieurs processeurs, le travail peut être divisé entre eux, pour accélérer les calculs
- **Modularité** : la conception modulaire des programmes, minimise les risque d'erreurs et permet une réutilisation des modules,
- **Commodité** : faire coopérer les processus peut être très commode à certaines applications même si on a un seul processeur, par exemple un processus peut lire les données un autre fait les calcul et un troisième affiche et imprime les résultats.

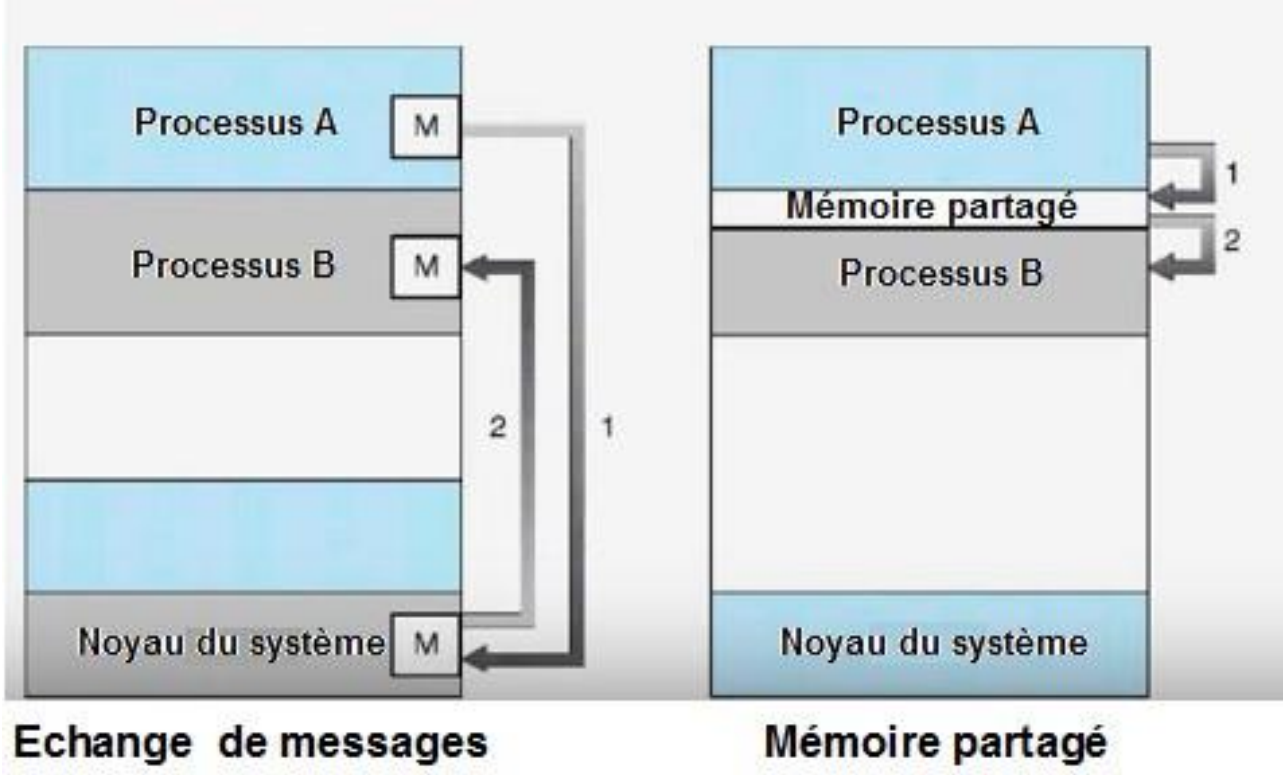
Communication entre processus

- **(IPC:InterProcess Communication)**

Pour coopérer, les processus doivent impérativement pouvoir communiquer ensemble.

Deux modèles principaux de communication existe :

La mémoire partagé et **le transfert de messages**



La mémoire partagé (shared memory region)

- Généralement, chaque processus a un espace d'adressage propre, auquel les autres processus ne peuvent pas accéder.
- Dans la mémoire partagé, les processus coopérants dans un espace partagé (hors de l'espace d'adressage du système) auquel ils peuvent accéder directement, cet espace est utilisé pour échanger directement des messages sans faire appel au système.
- Ce sont les processus eux même qui gèrent et synchronisent l'accès à la mémoire partagé et non pas le système, le système va juste réserver cette zone mémoire et l'attacher à l'espace d'adressage des processus coopérants.
- Le paradigme de partage de mémoire et celui du **Producteur/Consommateur**, un des processus dépose le message dans une zone tampon partagé et l'autre processus le récupère. (chapitre 2)
- Le problème avec le modèle de la mémoire partagé est qu'il ne peut pas être utilisé dans les système distribué où les processus s'exécutent dans des machines distantes connecté par un réseau physique.

Le transfert de messages (Message passing)

- Ce paradigme peut être utilisé même dans les système distribué. Dans ce cas chaque processus doit disposer de deux primitives de communication:
- Une primitive **send** pour envoyer un message,
- Une primitive **receive** pour recevoir un message.
- Le message peut être de taille fixe ou variables.

Le transfert de messages (Message passing)

- L'appel des primitives send et receive peut être bloquant ou non bloquant; ça veut dire que le processus appelant peut ou non, se bloquer en attendant l'envoi ou la réception du message.

Émetteur (***send***)	Récepteur (***receive***)	
Non bloqué	Non bloqué	Utilisation d'un tampon nécessaire
bloqué	Non bloqué	Problème de synchronisation
Non bloqué	bloqué	Problème de synchronisation
bloqué	bloqué	Problème de synchronisation rendez-vous

- Pour échanger des messages les processus doivent établir des liens de communication (communication link) entre eux .
- Physiquement un lien de communication peut être une mémoire partagé (Dans les machines monoprocesseur), ou un bus de communication physiques (dans le cas des machines multiprocesseurs) ou des liaisons réseaux physiques (Dans le cas des systèmes distribués).

Le transfert de messages (Message passing)

Un lien de communication a plusieurs propriétés logiques qui le caractérise :

- La capacité : combien de messages peut-il transférer ?
- Est ce qu'il est unidirectionnel ou bidirectionnel ?
- Est ce qu'il peut être utilisé par plus de deux processus ?
- Combien de lien peuvent être utilisé par un même processus ?
- Comment établir le lien entre les processus ?
- La communication entre les processus peut être directe ou indirecte.

Communication directe entre processus

- Dans ce cas quant un processus A veut envoyer un message à un autre processus B, le message de A est directement transféré à B.
- Un lien de communication est donc **automatiquement** établi par le système entre chaque paire de processus.
- Pour utiliser cette méthode chaque processus doit impérativement **connaître le nom du processus** avec lequel il veut communiquer,

Communication directe entre processus

- le problème est que si on change le nom d'un processus il faut reprogrammer tous les processus émetteurs à ce processus pour que ce nouveau nom soit pris en compte
- Les primitives de communications suivent le schéma suivant :

  (Le processus A envoi un message au processus B)
  - **send (B, message)** : envoyer le message au processus B,
  - **receive (A, message)** : recevoir un message du processus A.

Communication indirecte entre processus

- Dans ce cas quant un processus A veut envoyer un message à un autre processus B le message de A est enregistrer dans une zone mémoire appelé boite aux lettres (Mailbox) , le processus B vas ensuite le récupérer de la boite aux lettres.
- Les primitives de communications suivent le schéma suivant :

  (Le processus A envoi un message au processus B)

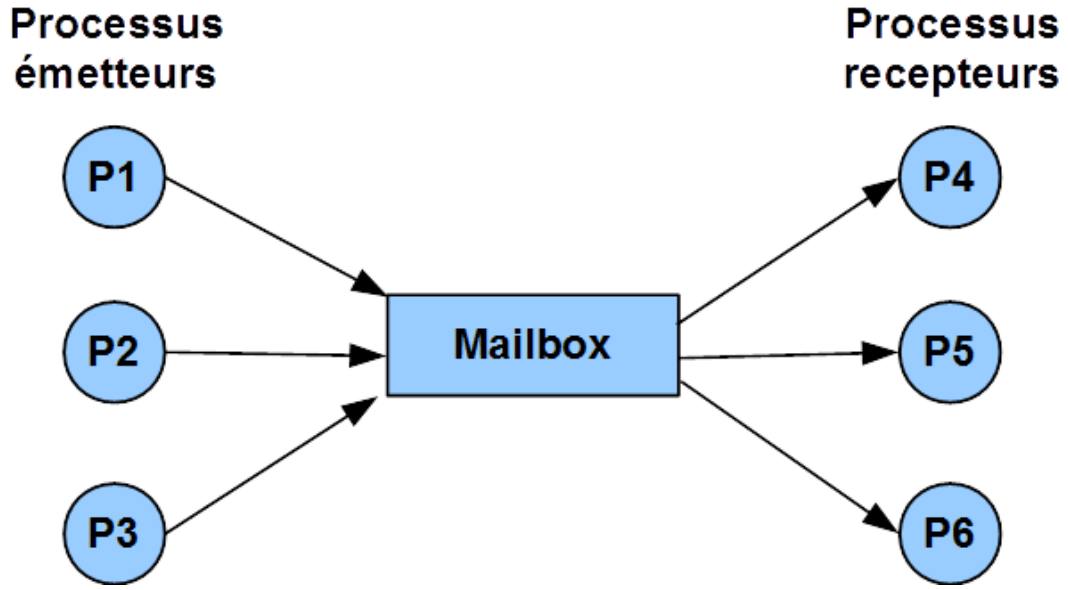
  - **send (MB, message):** envoyer le message à la boite aux lettres MB,
  - **receive (MB, message):** recevoir un message de la boite aux lettres MB.

Communication indirecte entre processus

- Chaque processus dispose de routines pour :
  - Créer une nouvelle boit aux lettres,
  - Envoyer et recevoir des message d'une boite aux lettres,
  - Détruire une boite aux lettres.
- Chaque boite aux lettres a un identifiant unique, pour que deux processus peuvent communiquer ils doivent avoir une boite aux lettres commune.

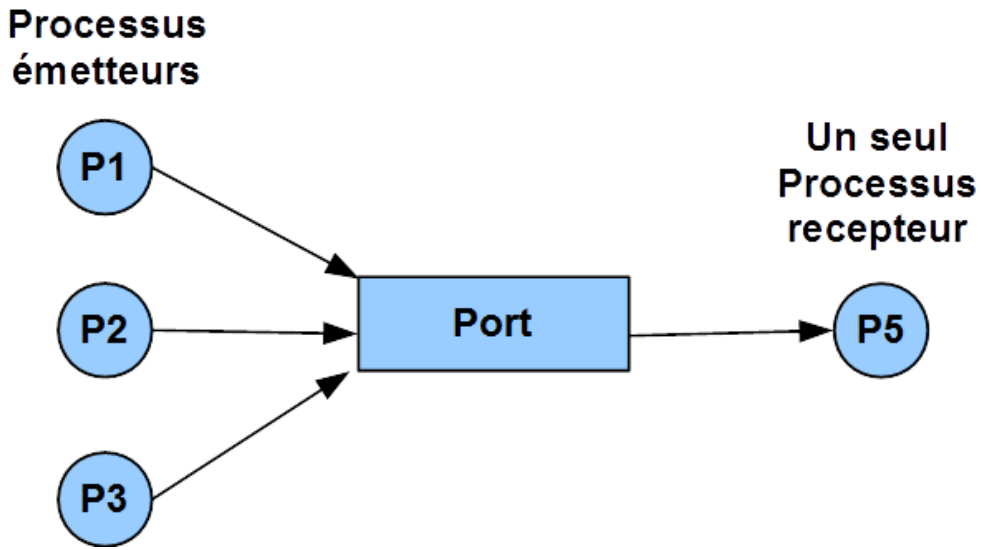
Communication indirecte entre processus

- Une boite aux lettres peut être associé avec un seul émetteurs et un seul récepteurs ou bien avec plusieurs émetteurs et plusieurs récepteurs
- une boite aux lettre est créer par un processus qui veut communiquer avec d'autres processus
- elle sera détruite quand le processus qui l'a créé demande sa suppression.



Communication indirecte entre processus

- Il existe un type particulier de boite aux lettres appelé port
- Un port est une boite aux lettres qui peut être seulement associé avec un seul récepteur et éventuellement plusieurs émetteurs
- Un port est créer par un processus qui veut recevoir des messages
- il est détruit automatiquement quand le processus émetteur se termine.

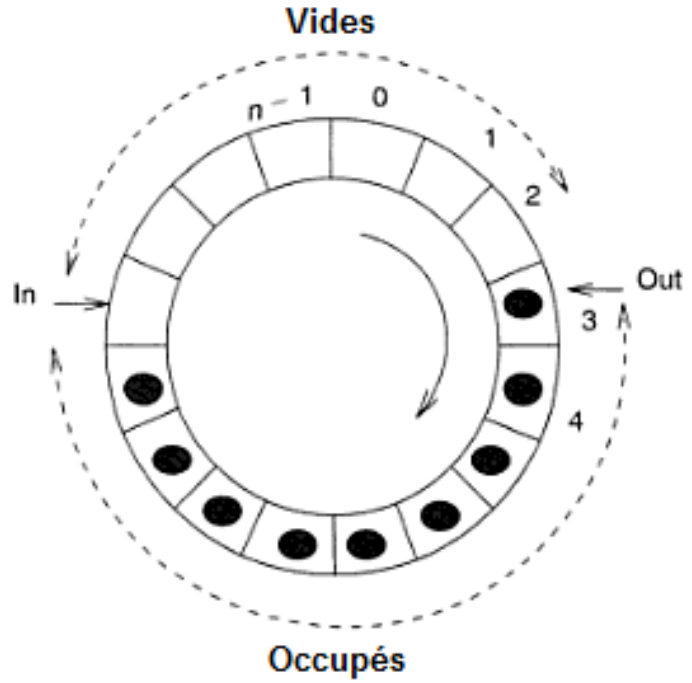


Exemples de IPC (InterProcess Communication)

[ref : « Operating Systems » : Sibsankar Haldar,Alex Alagarsamy Aravind]

1- Message Queue :

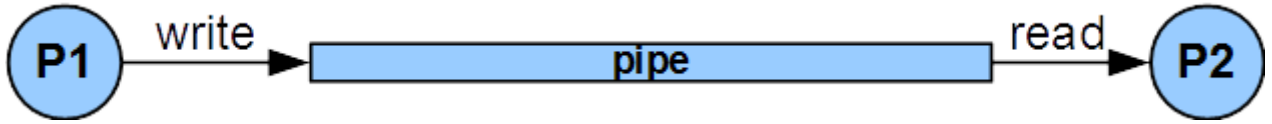
- Ils sont utilisé dans le système UNIX
- un Message Queue est une structure de données de taille prédéfinit qui réside dans l'espace d'adressage du système dans laquelle les processus peuvent ajouter ou retirer des informations
- les information ajouté a la queue sont ordonnées selon la politique FIFO
- un processus ne peut pas ajouter de messages si la queue est pleine aussi il ne peut pas retirer un message si la queue est vide
- Les message Queue sont en quelques sorte des boites au lettres (mailbox) avec un **tampon circulaire** de taille prédéfini.



Exemples de IPC *[ref : « Operating Systems » : Sibsankar Haldar,Alex Alagarsamy Aravind]*

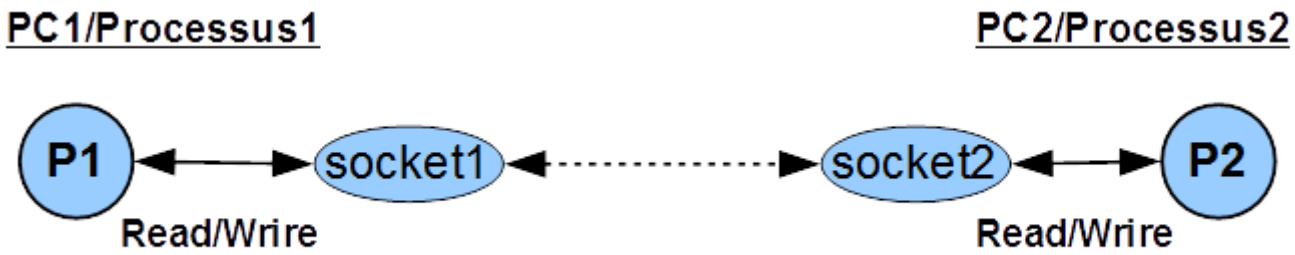
2- Les pipes (tubes)

Les pipes sont utilisées dans les systèmes UNIX pour faire **communiquer deux processus** un premier processus émetteur copie les messages dans le pipe et un second processus récepteur le récupère. La communication dans le pipe est **unidirectionnelle**.



Exemples de IPC *[ref : « Operating Systems » : Sibsankar Haldar,Alex Alagarsamy Aravind]*

- **3- Les sockets** (prise)
  Une **socket** est une en quelque sorte un **pipe** mais qui permet une communication bidirectionnelle
- les **sockets** ont été développé spécialement pour les communication dans les systèmes distribués mais ils peuvent être utilisé avec un seul PC.
- Pour faire communiquer deux processus il faut deux sockets connecté entre eux,
- A chaque socket est associé une **adresse IP** et un **numéro de port**. Le protocole TCP/IP est utilisé pour transmettre les messages.



Exemples de IPC *[ref : « Operating Systems » : Sibsankar Haldar,Alex Alagarsamy Aravind]*

- **Autre techniques:**
- **Les RPC (Remote Procedure Call) :**

Sa consiste à invoquer des procédures a distance (le programme et la procédure ne se trouent pas dans le même ordinateur)

- **Les RMI (Remote Method Invocation) :**

C'est une technique utilisé dans le langage Java. Elle permet d'invoquer une méthode d'un objet qui se trouve dans une machine virtuelle distante. c'est en quelque sorte des RPCs pour le langage Java.

- **Remarque**

Les pipes, les Sockets, les IPCs et les RMIs sont des techniques de communication dite Client/Serveur.