Synchronisation et communication entre processus

Principes : exclusion mutuelle et interblocage

Introduction

Nous nous intéressons au développement d'applications multiprocessus concurrents c'est à dire d'applications composées de plusieurs processus indépendants et en concurrence pour l'accès aux ressources du système.

Les processus sont ordonnancés indépendamment les uns des autres.

Une **ressource** désigne toute entité dont a besoin un processus pour s'exécuter. Il existe deux types de ressources :

- ressource matérielle (processeur, périphérique)
- ressource logicielle (fichier)

Introduction

Dans Système multiprocessus, l'ordonnancement "entrelace" les exécutions des processus

Les processus peuvent ne pas être indépendants :

→ accès concurrents aux ressources

Notion de ressources

Une ressource désigne toute entité dont a besoin un processus pour s'exécuter. Il existe deux types de ressources :

- Ressource matérielle (processeur, périphérique)
- Ressource logicielle (fichier, variable).

Une ressource est caractérisée

- par un état : libre / occupée
- par son nombre de points d'accès (nombre de processus pouvant l'utiliser en même temps)

Notion de ressources

le cnam

Utilisation d'une ressource par un processus

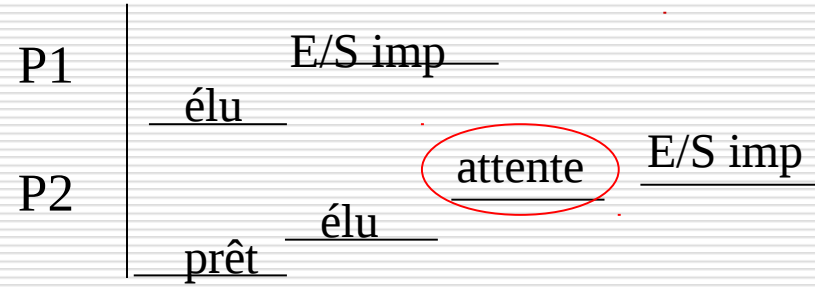
- Trois étapes : Allocation
  Utilisation
  Restitution

- Les phases d'allocation et de restitution doivent assurer que le ressource est utilisée conformément à son nombre de points d'accès

Une ressource critique a un seul point d'accès.

Notion de ressources : exemple

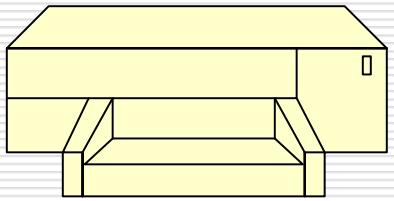
Ressource matérielle : imprimante



Processus P1

Print (texte)

SE
Gestion des entrées/sorties
PRINT



LIBRE
1 point d'accès

Print (texte)

Allouée P1

OCCUPEE
1 point d'accès

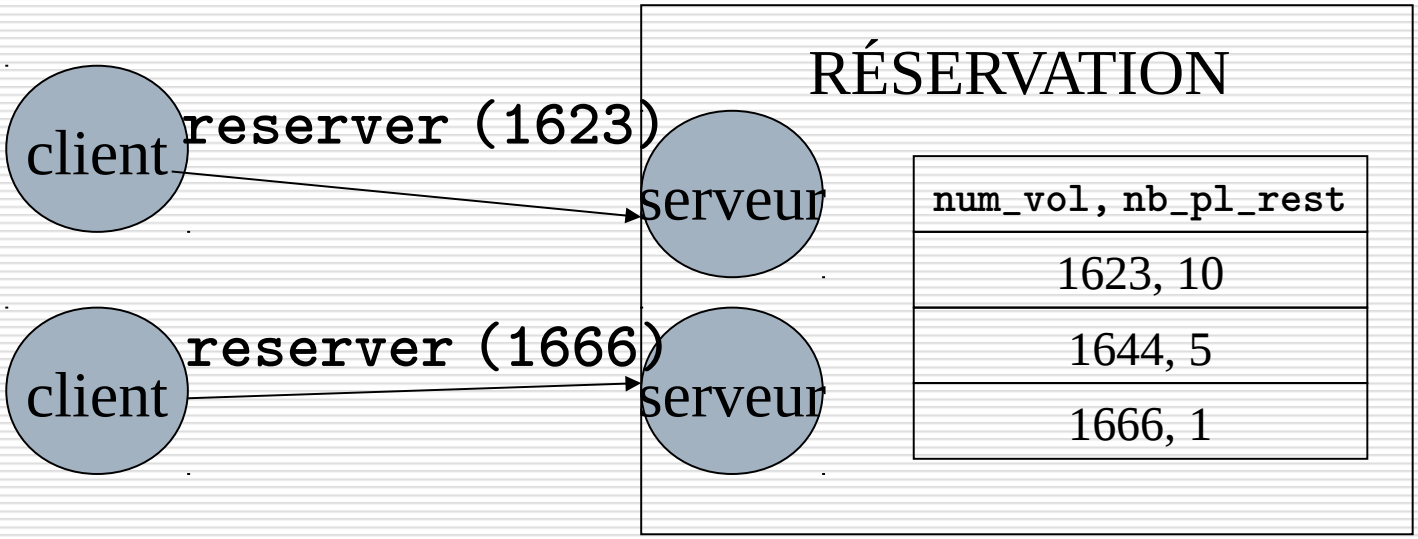
Processus P2

P2 en attente jusqu'à
libération par P1

Notion d'exclusion mutuelle entre processus

Un système de réservation de vols

On considère un système permettant à des clients de réserver une place dans un avion donné **`num_vol`**.



Exclusion mutuelle entre processus

le cnam

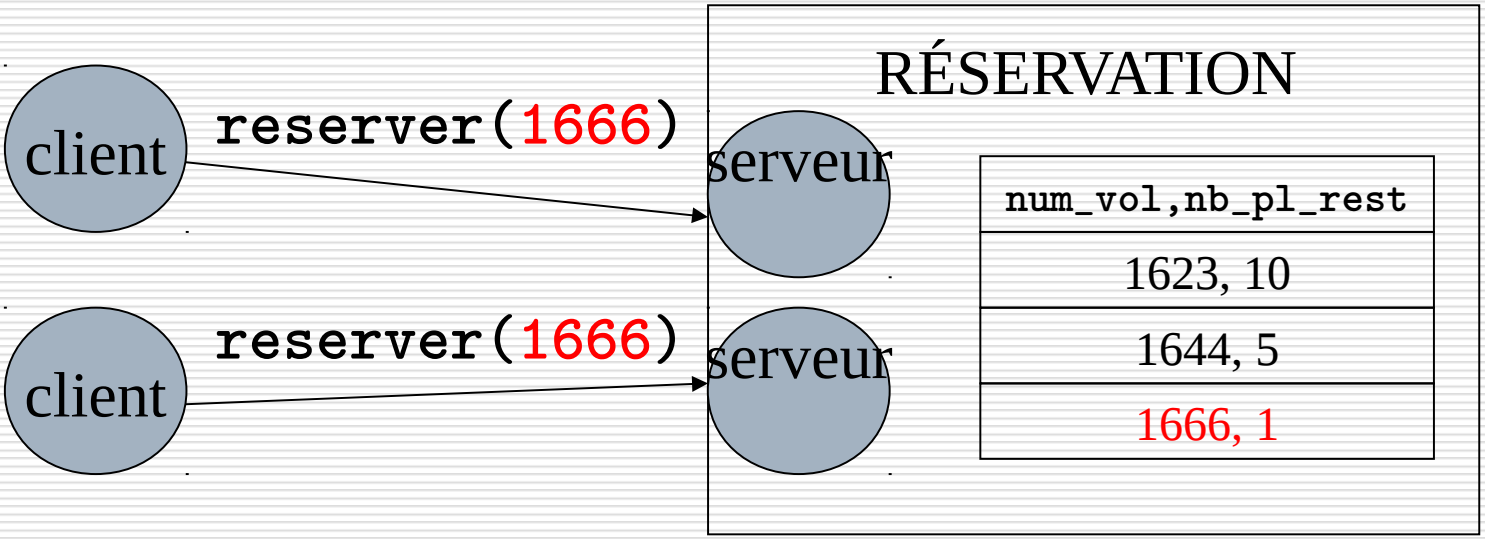
Client 1

Client 2

```
reserver (num_vol)
debut
    acceder a (num_vol) dans la table des vols
    si ( nb_pl_rest > 0 ) alors
      reserver(num_vol)
      nb_pl_rest = nb_pl_rest - 1
     finSi
fin
```

Exclusion mutuelle entre processus

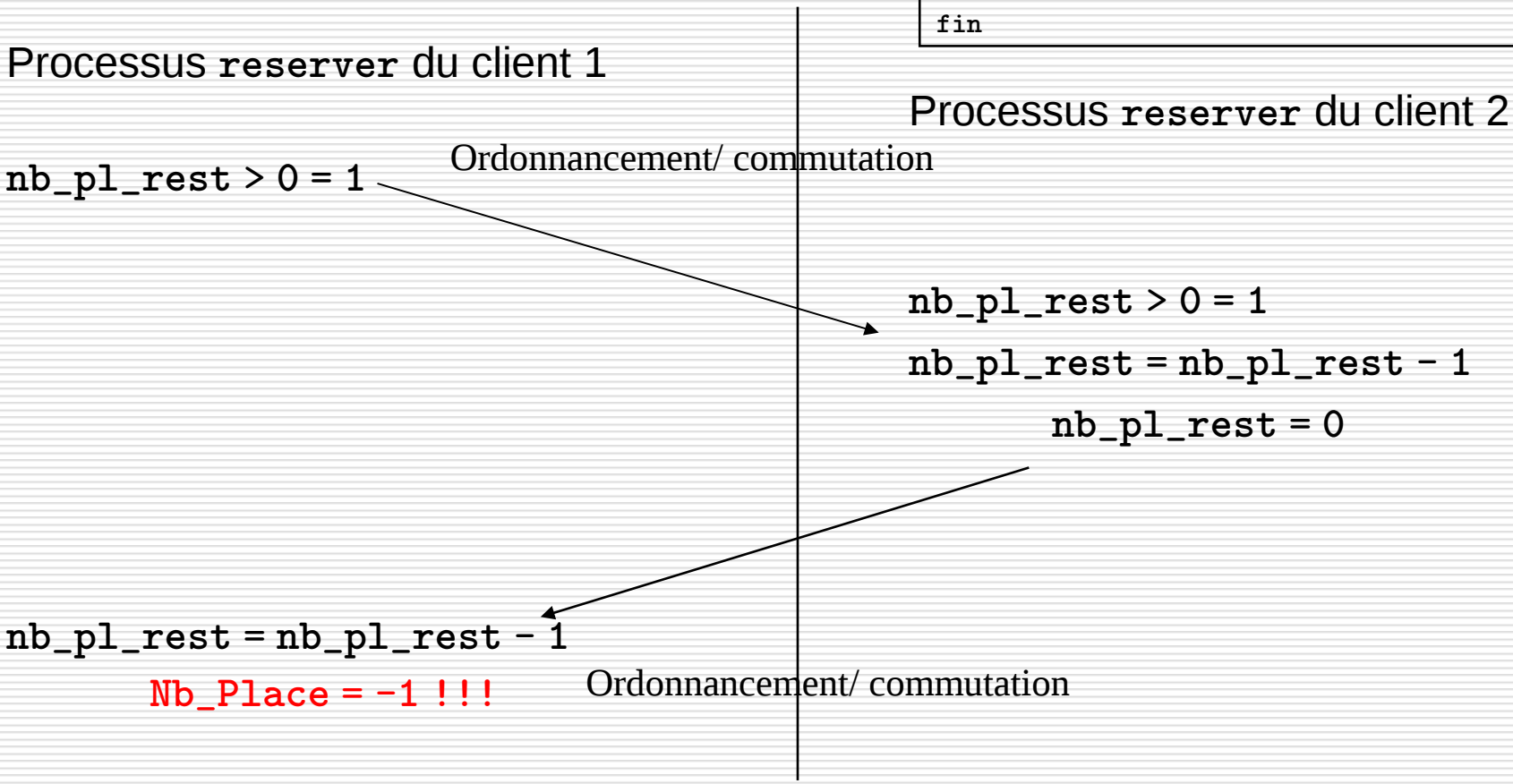
On considère la situation où deux clients demandent simultanément la réservation d'une place sur le vol 1666 pour lequel reste une seule place disponible



Exclusion mutuelle entre processus

Les deux processus **reserver** s'exécutent en concurrence ; le SE ordonnance les deux processus via un algorithme en temps partagé

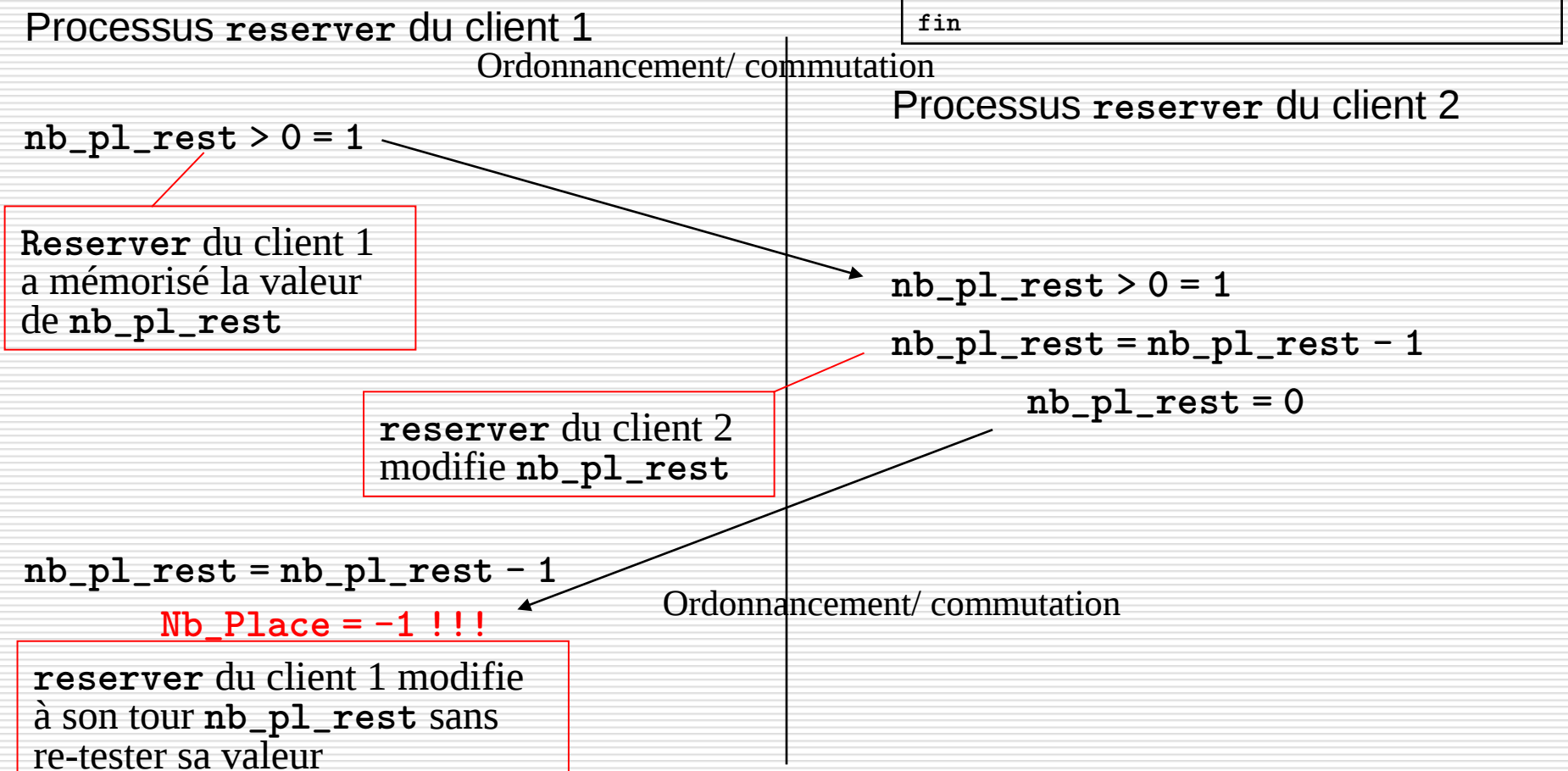
```
reserver (num_vol)
debut
      acceder a (num_vol) dans la table des vols
      si ( nb_pl_rest > 0 ) alors
         reserver(num_vol)
         nb_pl_rest = nb_pl_rest - 1
       finSi
fin
```



Exclusion mutuelle entre processus

Que s'est-il passé ? L'entrelacement des exécutions a permis à `reserver` du client_2 de modifier `nb_pl_rest` alors que `reserver` du client_1 accédait déjà à cette variable

```
reserver (num_vol)
debut
      acceder a (num_vol) dans la table des vols
      si ( nb_pl_rest > 0 ) alors
         reserver(num_vol)
         nb_pl_rest = nb_pl_rest - 1
       finSi
fin
```



Exclusion mutuelle entre processus

le cnam

Que faut-il faire ?

Interdire au processus Processus `reserver` du client 2 de modifier la variable `nb_pl_rest` pendant que le Processus `reserver` du client 1 cette variable

→ un seul processus à la fois accède à `nb_pl_rest`

`nb_pl_rest` est une ressource critique à un seul point d'accès.

Processus `reserver` du client 1

```
si ( nb_pl_rest > 0 ) alors
    reserver(num_vol)
    nb_pl_rest = nb_pl_rest - 1
finSi
```

Processus `reserver` du client 2

`nb_pl_rest` non accessible pour le client 2 tant que client 1 manipule cette variable

Section critique et exclusion mutuelle

le cnam

Processus
Début

Entrée Section Critique

Ressource Critique
`nb_pl_rest`

SECTION CRITIQUE (code d'utilisation de la ressource critique)

Sortie Section Critique

Fin

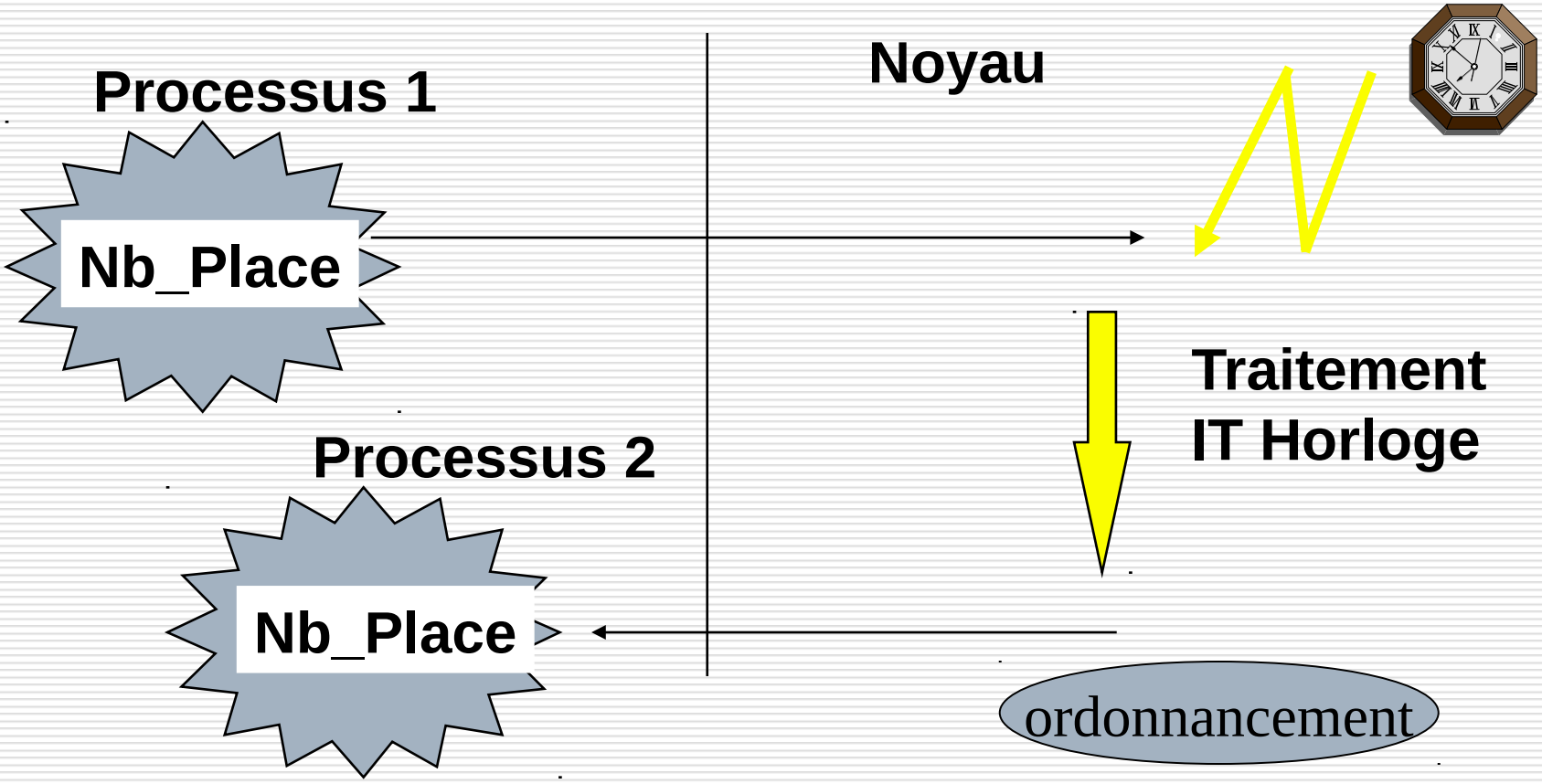
L'entrée et la sortie de Section Critique doivent assurer qu'à tout moment, un seul processus s'exécute en Section Critique.

→ principe de l'exclusion mutuelle.

Notion d'exclusion mutuelle entre processus

le cnam

Le processus `reserver` du client 2 a pu s'exécuter car l'ordonnanceur a préempté `reserver` du client 1 et élu `reserver` du client 2.



Notion d'exclusion mutuelle entre processus
Solution matérielle

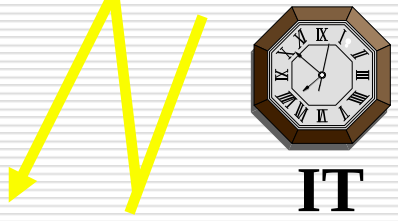
le cnam

Une première solution pour empêcher **`reserver`** du client 2 de s'exécuter est de masquer les interruptions.

Mais exécution d'un programme utilisateur en mode superviseur

Processus 1

```
masquer IT
si ( nb_pl_rest > 0 ) alors
    reserver(num_vol)
    nb_pl_rest = nb_pl_rest - 1
finSi
demasquer IT
```



IT

Non prise en compte

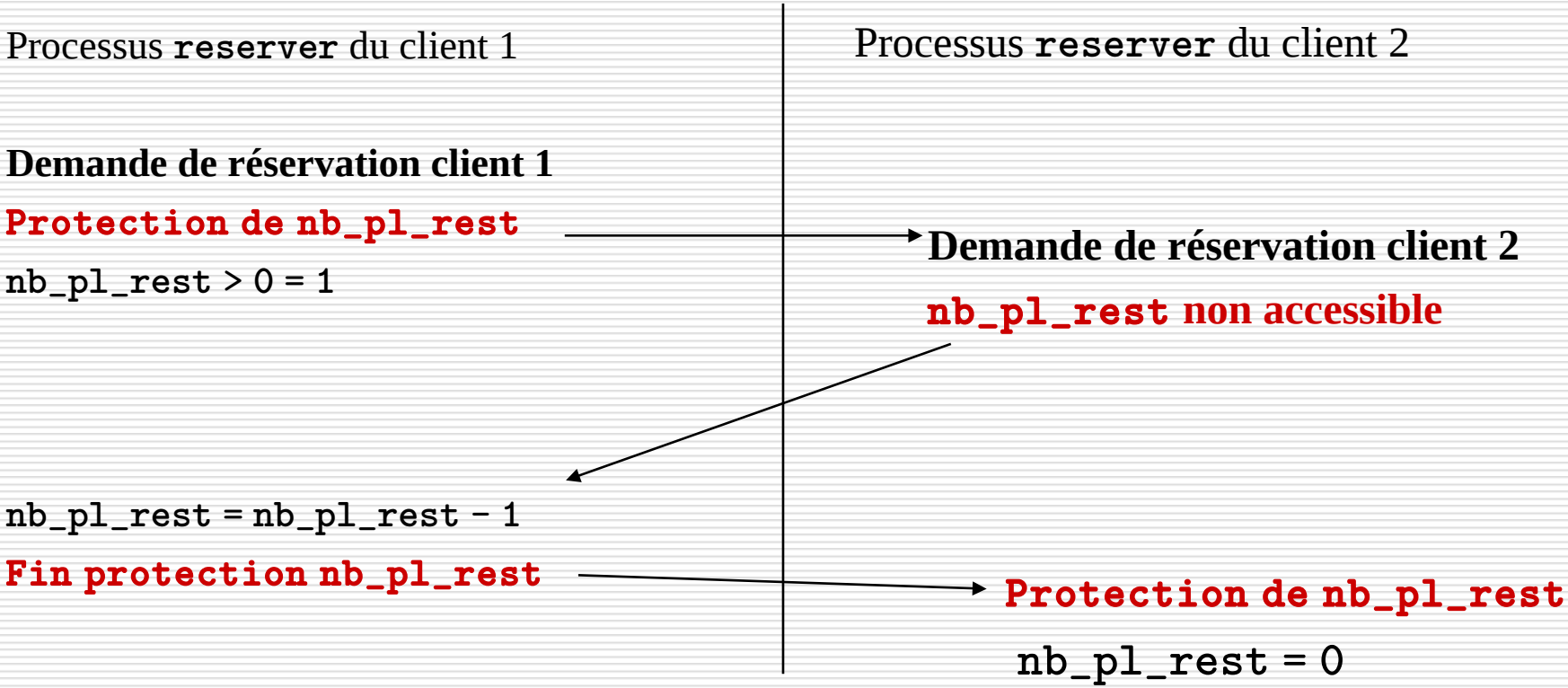
Processus 2

Attente

Notion d'exclusion mutuelle entre processus
Solution logicielle

le cnam

On ne souhaite plus interdire l'entrelacement des processus.
On protège l'accès à **`nb_pl_rest`** : le processus **`reserver`** du client 1 « verrouille » l'accès à **`nb_pl_rest`** tant qu'il utilise la variable



Notion d'exclusion mutuelle entre processus
Solution logicielle : le verrou

le cnam

Un mécanisme proposé pour permettre de résoudre l'exclusion mutuelle d'accès à une ressource est le mécanisme de *verrou.* Un verrou est un objet système à deux états (libre/occupé) sur lequel deux opérations sont définies..

- `verrouiller (v)` permet au processus d'acquérir le verrou *v* s'il est libre. S'il n'est pas disponible, le processus est bloqué en attente de la ressource.
- `déverrouiller (v)` permet au processus de libérer le verrou *v* qu'il possédait. Si un ou plusieurs processus étaient en attente de ce verrou, un seul de ces processus est réactivé et reçoit le verrou.

En tant qu'opérations systèmes, ces opérations sont indivisibles, c'est-à-dire que le système qu'elles s'exécutent interruptions maquées.

Notion d'exclusion mutuelle entre processus
Solution logicielle : le verrou

`reserver` du client 1

`V_nb_place : verrou;`

Demande de réservation client 1

`Protection de nb_pl_rest`

`nb_pl_rest > 0 = 1`

`verrouiller (V_nb_place)`
Si `V_nb_place` libre alors autoriser l'accès et mettre `V_nb_place` à l'état occupé sinon bloquer le processus

`nb_pl_rest = nb_pl_rest - 1`

`Fin protection nb_pl_rest`

`deverrouiller (V_nb_place)`
Si un processus bloqué en attente pour accéder à `nb_pl_rest`, le débloquer, Sinon `V_nb_place` libre

Notion d'exclusion mutuelle entre processus
Solution logicielle

le cnam

`V_nb_place : verrou;` -- verrou libre

`reserver` du client 1

Demande de réservation du client 1

`Verrouiller (V_nb_place)`

`V_nb_place` libre

`nb_pl_rest > 0 = 1`

`reserver` du client 1

Demande de réservation du client 2

`Verrouiller (V_Nb_Place)`

`V_Nb_Place` **occupé par le client 1**

`reserver` **du client 2 bloqué**

`nb_pl_rest = nb_pl_rest - 1`

`Deverrouiller (V_nb_place)`

Réveil de `reserver` **du client 2**

`nb_pl_rest = 0`

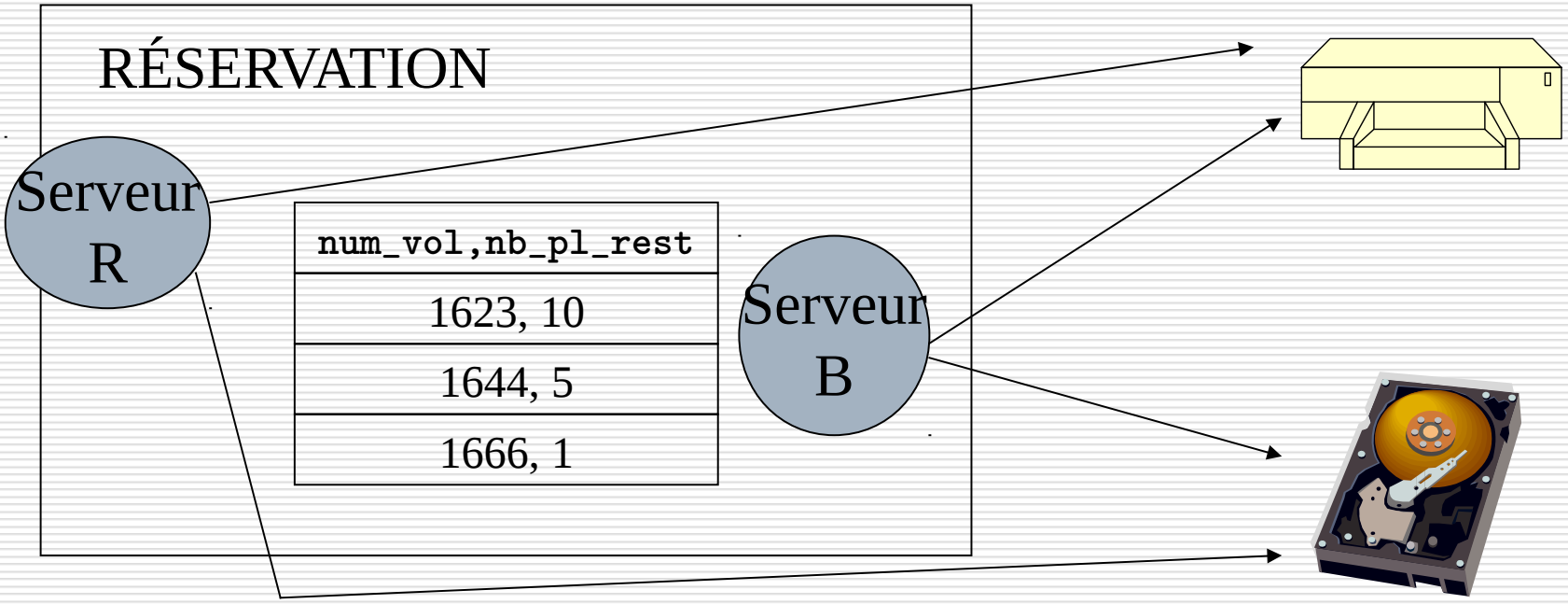
`deverrouiller (v_nb_Place)`

`V_nb_place : état libre`

Notion d’interblocage

Un système de réservation de vols

On considère un système permettant à des clients de réserver une place dans un avion donné `num_vol`.



Notion d'interblocage

- Le système de réservation dispose de deux ressources : une imprimante, un disque

- Le processus `reservation_client`, une fois la réservation effectuée, imprime une facture pour le client et écrit sur le disque dans un fichier commande, un enregistrement correspondant (commande en cours, paiement en attente)

- Un processus `billet`, régulièrement, parcourt dans le fichier commande, les enregistrements commande et pour chacun d'eux pour lesquels un paiement a été reçu, passe la commande à état payé édite sur l'imprimante un billet.

Notion d'interblocage

le cnam

On programme comme suit les deux processus

Processus `reservation_client`	Processus `billet`

`Effectuer réservation client`
`verrouiller (V_imprimante)`
`verrouiller (V_fichier_commande)`
`imprimer (facture)`
`ecrire(commande,fichier_commande)`
`deverrouiller (V_imprimante)`
`deverrouiller(V_fichier_commande)`
	`toutes les P unités de temps faire`
	`  verrouiller (V_fichier_commande)`
	`  verrouiller (V_imprimante)`
	`  tant que (commandes)`
	`    si paiement mettre etat paye`
	`      imprimer (billet)`
	`    finSi`
	`  finTantQue`
	`  deverrouiller (V_fichier_commande)`
	`  deverrouiller (V_imprimante)`

Notion d'interblocage

Processus `reservation_client`	Processus `billet`

`Effectuer réservation client`
`verrouiller (V_imprimante)`
Imprimante libre allouée à `reservation_client`
 	
`verrouiller (V_fichier_commande)`
`reservation_client` bloqué
	`toutes les P unités de temps faire`
	`verrouiller (V_fichier_commande)`
	fichier commande libre alloué à `billet`
	`verrouiller (V_imprimante)`
	`billet` bloqué

`imprimer (facture)`
`ecrire(commande,fichier_commande)`
`deverrouiller (V_imprimante)`
`deverrouiller(V_fichier_commande)`
	`tant que (commandes)`
	`si paiement mettre etat paye`
	`imprimer (billet)`
	`finSi`
	`finTantQue`
	`deverrouiller (V_fichier_commande)`
	`deverrouiller (V_imprimante)`

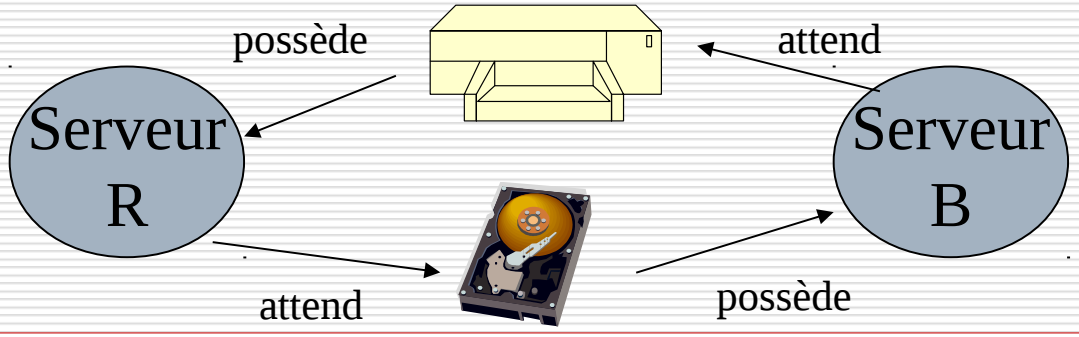
Notion d'interblocage

le cnam

Processus `reservation_client`	Processus `billet`

`Effectuer réservation client`
`verrouiller (V_imprimante)`
Imprimante libre allouée à `reservation_client`
 	
`verrouiller (V_fichier_commande)`
`reservation_client` bloqué
	`toutes les P unités de temps faire`
	`verrouiller (V_fichier_commande)`
	fichier commande libre alloué à `billet`
	`verrouiller (V_imprimante)`
	`billet` bloqué

INTERBLOCAGE : Ensemble de n processus attendant chacun une ressource déjà possédée que par un autre processus de l'ensemble



Notion d'interblocage

Les deux processus demandent l'accès aux ressources dans un ordre différent.

→ imposer un ordre de demande des ressources

Processus `reservation_client`

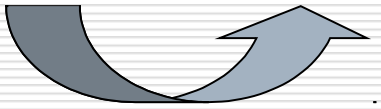
```
verrouiller (V_imprimante)
verrouiller (V_fichier_commande)
...
deverrouiller (V_imprimante)
deverrouiller (V_fichier_commande)
```

Processus `billet`

```
Verrouiller (V_imprimante)
Verrouiller (V_fichier_commande)
...
Deverrouiller (V_imprimante)
Deverrouiller
(V_fichier_commande)
```

Processus `billet`

```
verrouiller (V_fichier_commande)
verrouiller (V_imprimante)
...
deverrouiller (V_imprimante)
deverrouiller (V_fichier_commande)
```



Notion d'interblocage : prévention

Processus `reservation_client`	Processus `Billet (B)`

`Effectuer réservation client`
`verrouiller (V_imprimante)`
(imprimante libre allouée à `reservation_client`)	

	`Reveil de billet`
	`verrouiller (V_imprimante)`
	(`billet` bloqué)

`Verrouiller (V_fichier_commande)` (Fichier libre alloué à `reservation_client`)	
`imprimer (facture)` `ecrire (commande, fichier_commande)`	`billet` débloqué, acquiert `V_fichier_commande` et s'exécute
`deverrouiller (V_imprimante)` `deverrouiller (V_fichier_commande)`	