Prod. Alain MENU
Lycée La Fayette - 0770920G
iR!S
Informatique et Réseaux pour l'Industrie et les Services techniques

# Les threads et la synchronisation sous Linux et Windows

af

2006

Prof. Alain MENU
Lycée La Fayette - 0770920G
IRIS
Informatique et Réseaux pour l'Industrie et les Services techniques

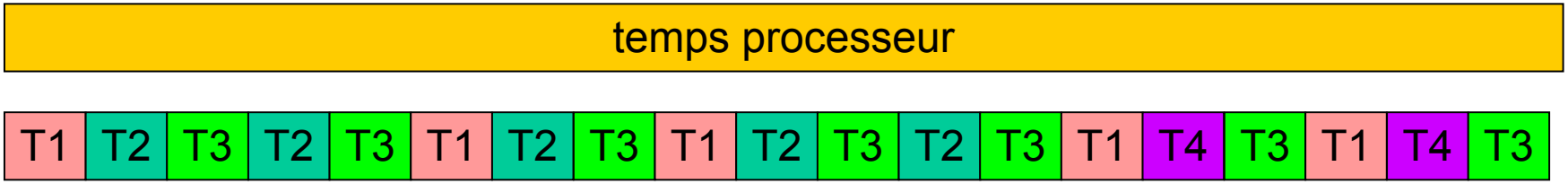
# Sommaire

- Rappels
- Définitions
- Création d'une thread
- Exemple industriel sans synchronisation.
- Une solution: Le verrou
- Exemple industriel avec synchronisation.
- Problème : Le verrou mortel
- Un autre outil de synchronisation
- Conclusion

Prod. Alain MENU
Lycée La Fayette - 0770920G
iRiS
Informatique et Réseaux pour l'Industrie et les Services techniques

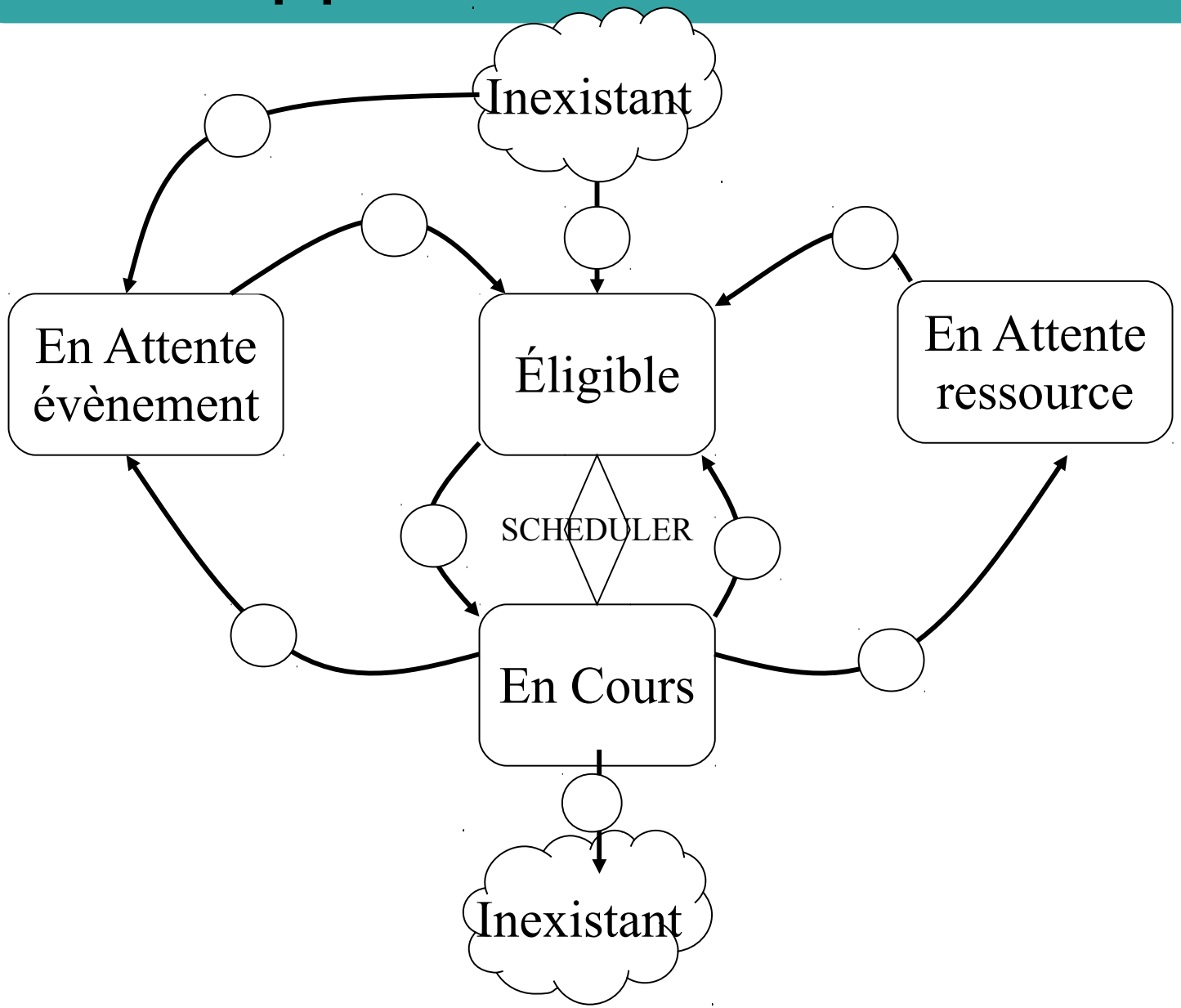
# Rappels: Un OS Multitâche

- Partage du temps processeur entre les tâches ou processus.




- Le scheduler choisit la tâche à exécuter suivant un algorithme d'ordonnancement.

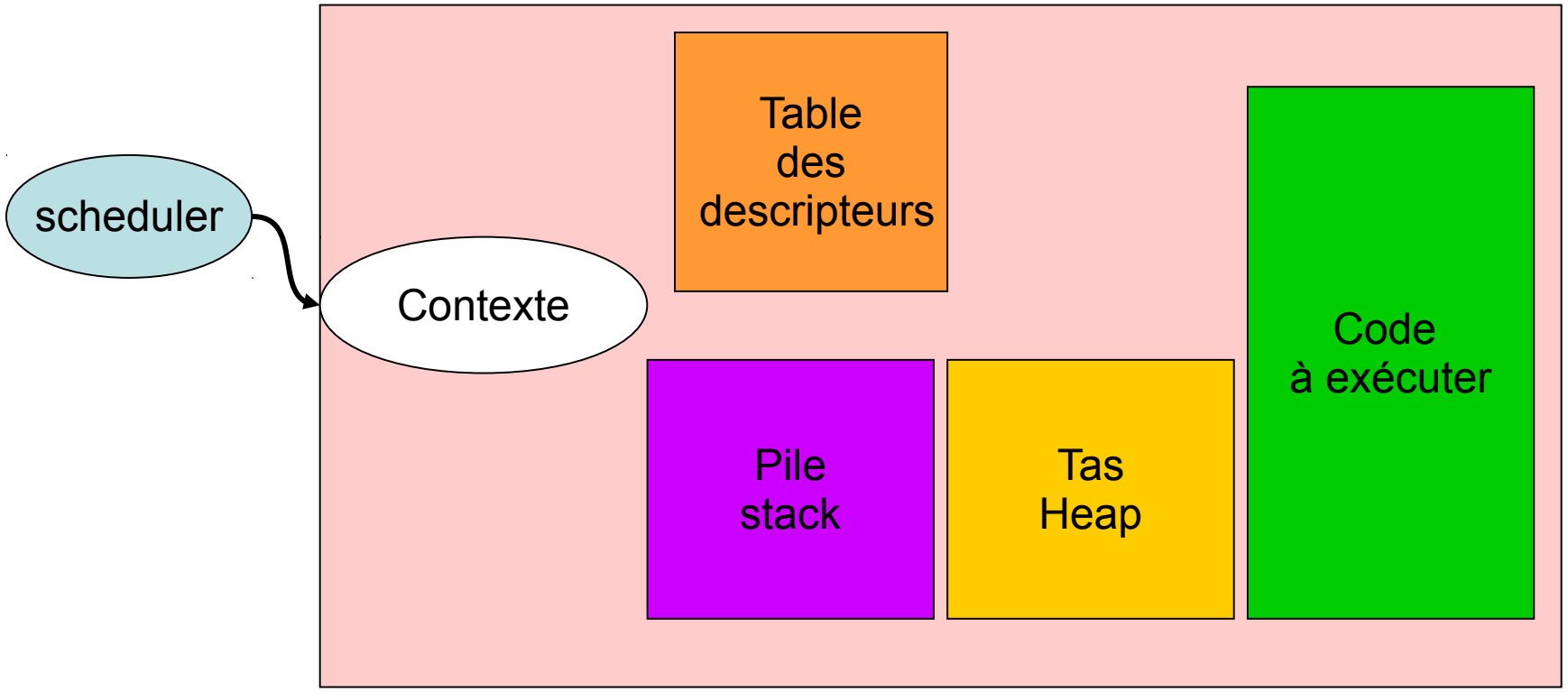
# Rappels: État des tâches



Prod. Alain MENU
Lycée La Fayette - 0770920G
iRIS
Informatique et Réseaux pour l'Industrie et les Services techniques

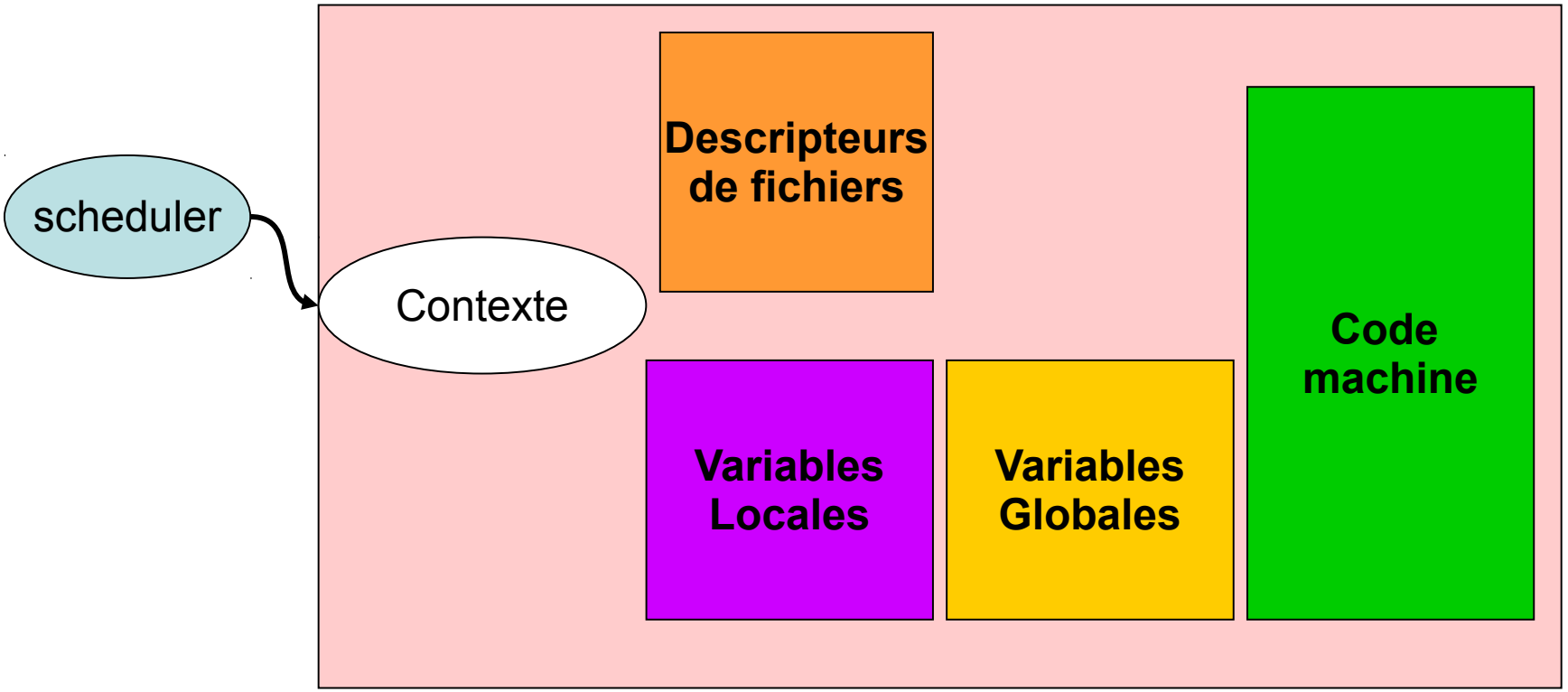
# Rappels: Un processus

- Un processus est composé de:

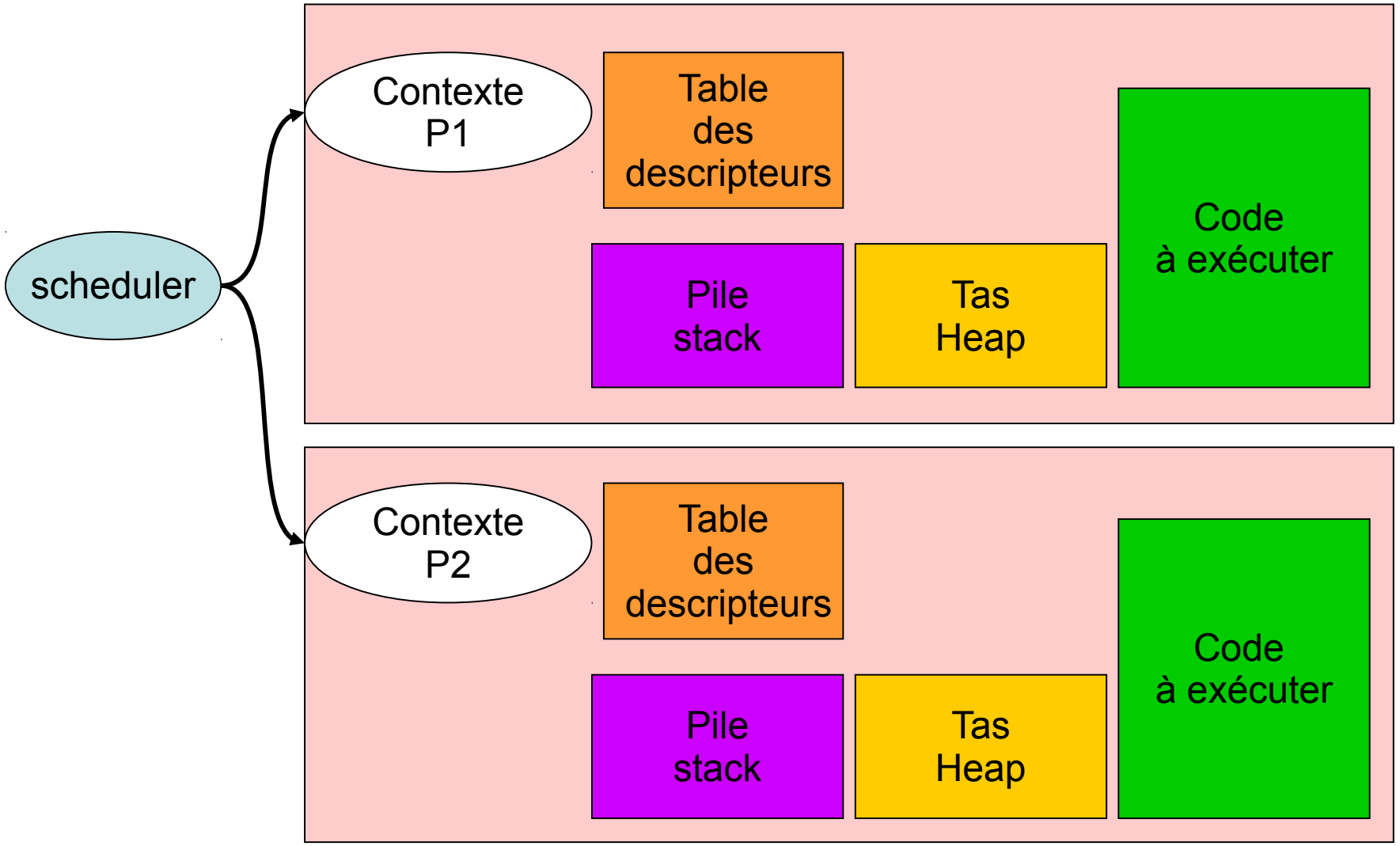


# Rappels: Un processus

- Exemple



# Rappels: Deux processus



# Les processus

- Une programmation utilisant plusieurs processus à quelques inconvénients:
  - Difficultés d'échanges de données entre processus
  - Moins d'outils de synchronisation
  - Lenteur : Changement de contexte long

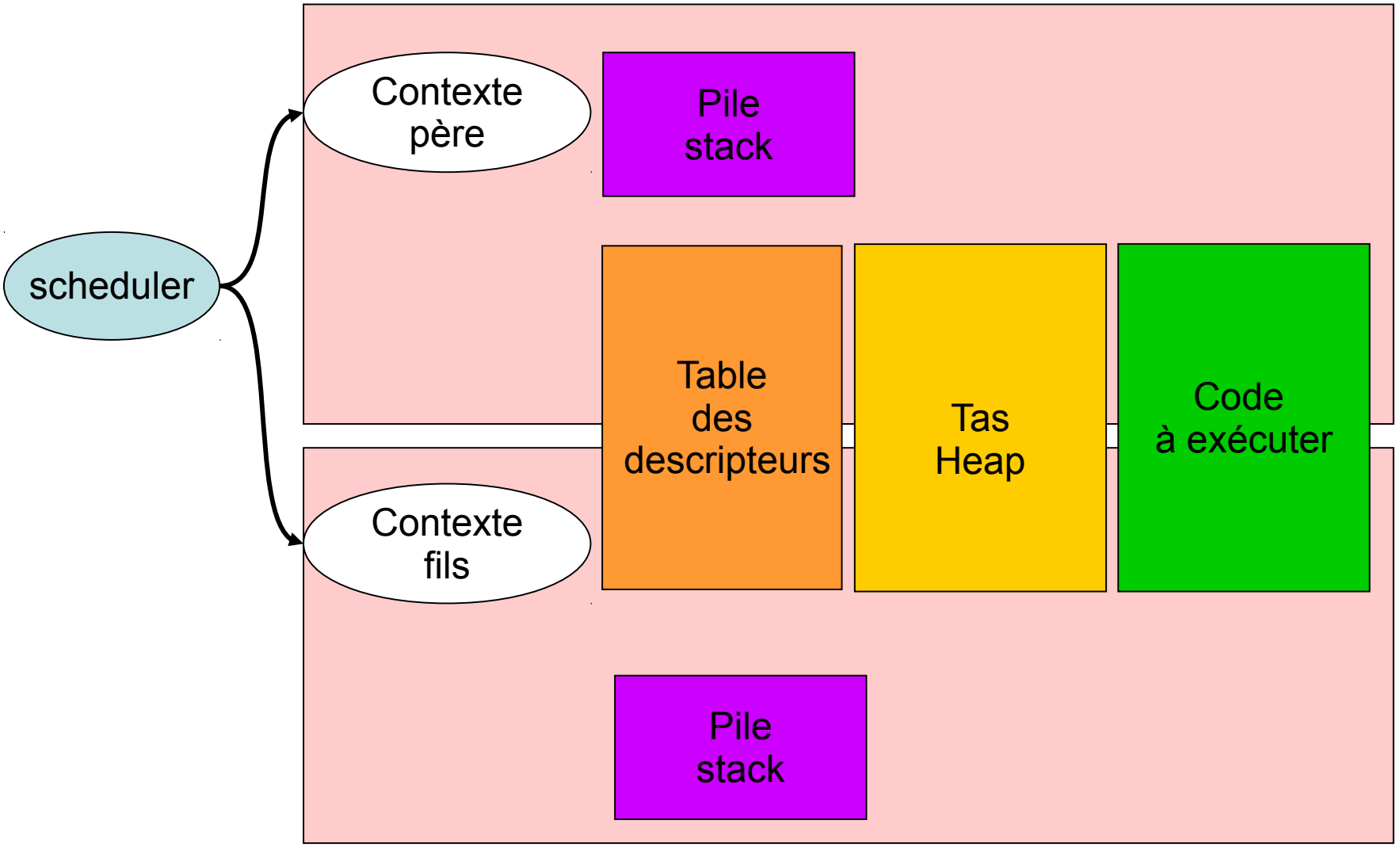
- D'où l'utilité des threads

# Définition: Un Thread

- Thread :
  - activité,
  - fil d'exécution,
  - *lightweight process* (*lwp*) ou processus léger

- Par la suite nous utiliserons le terme «thread » ou « processus léger »

# Définition: Un Thread



# Création de thread

- Les paramètres:
  - Un pointeur sur une fonction à exécuter.
  - Les arguments de la fonction.
  - Les attributs.

- Valeur de retour:
  - Un descripteur du thread fils.

Prod. Alain MENU
Lycée La Fayette - 0770920G
IRIS
Informatique et Réseaux pour l'Industrie et les Services techniques

# Création de thread sous Linux

```
int pthread_create(
   pthread_t *pth,
   const pthread_attr_t *pattr,
   void (*start)(void *),
   void *arg
);
```

**Attributs : NULL = attributs par défaut**

- **Caractéristiques d'ordonnancement**
- **Attributs de copie de descripteur**

# Création de thread sous W32

**Paramètres de la fonction CreateThread() :**

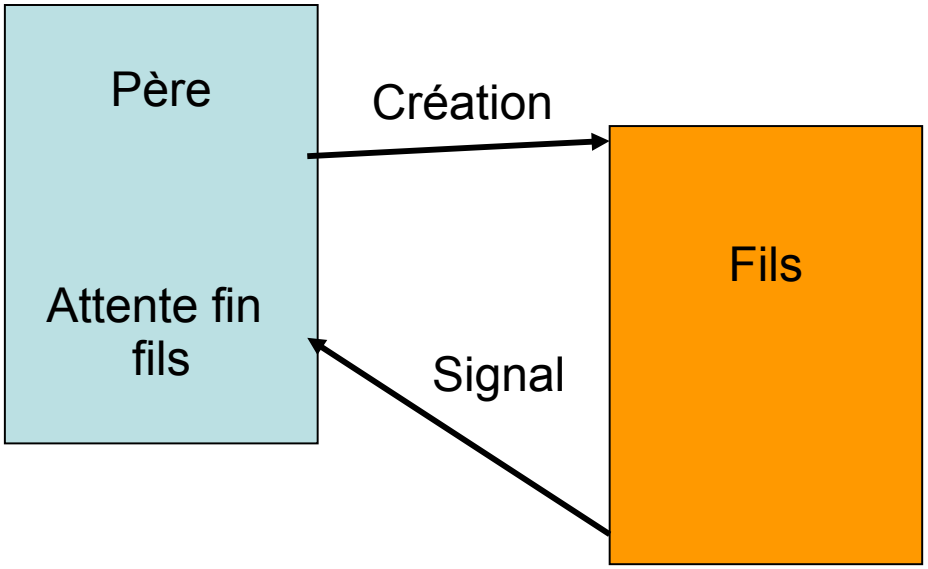
```
HANDLE CreateThread(

    LPSECURITY_ATTRIBUTES
lpThreadAttributes,
    DWORD  dwStackSize,
    LPTHREAD_START_ROUTINE  lpStartAddress,
    LPVOID  lpParameter,
    DWORD  dwCreationFlags,
    LPDWORD  lpThreadId
 );
```

**CreateThread() retourne un HANDLE (descripteur) si création réussie sinon NULL (GetLastError())**

# Attente de la fin du fils

- Le père doit attendre la fin du fils
- Lors de sa terminaison, le fils envoie un signal à son père



# Les fonctions d'attente

- La fonction sous linux

```
int pthread_join(
        pthread_t thread,
        void **statusp );
```

- La fonction sous Windows

```
DWORD WaitForSingleObject(
    HANDLE hHandle,
    DWORD dwMilliseconds );
```

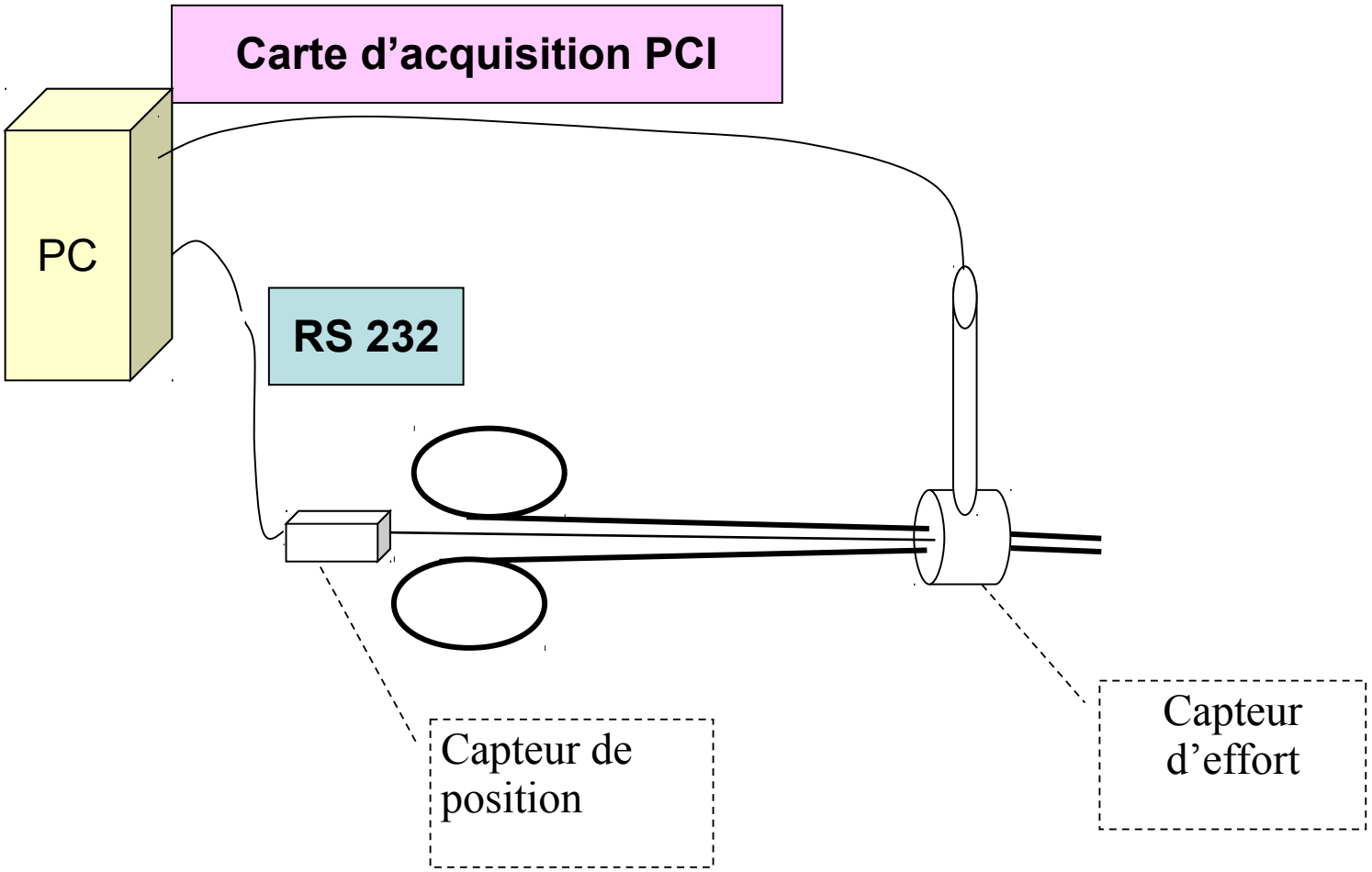
Prod. Alain MENU
Lycée La Fayette - 0770920G
IRIS
Informatique et Réseaux pour l'Industrie et les Services techniques

# La Synchronisation

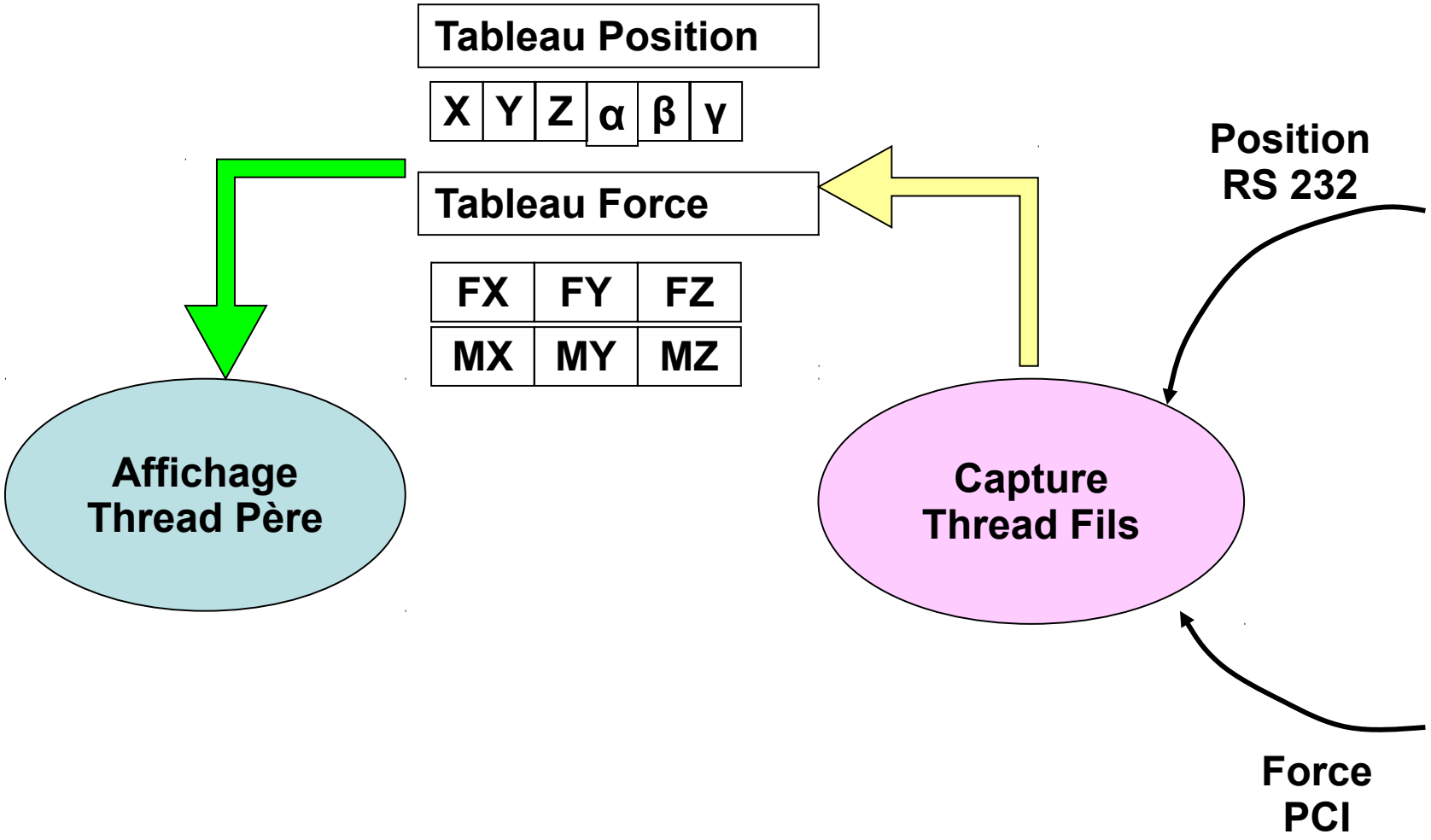
- Exemple industriel sans synchronisation.
- Une solution: Le verrou
- Exemple industriel avec synchronisation.
- Problème : Le verrou mortel
- Un autre outil de synchronisation.
- Conclusion

# Exemple industriel (1)

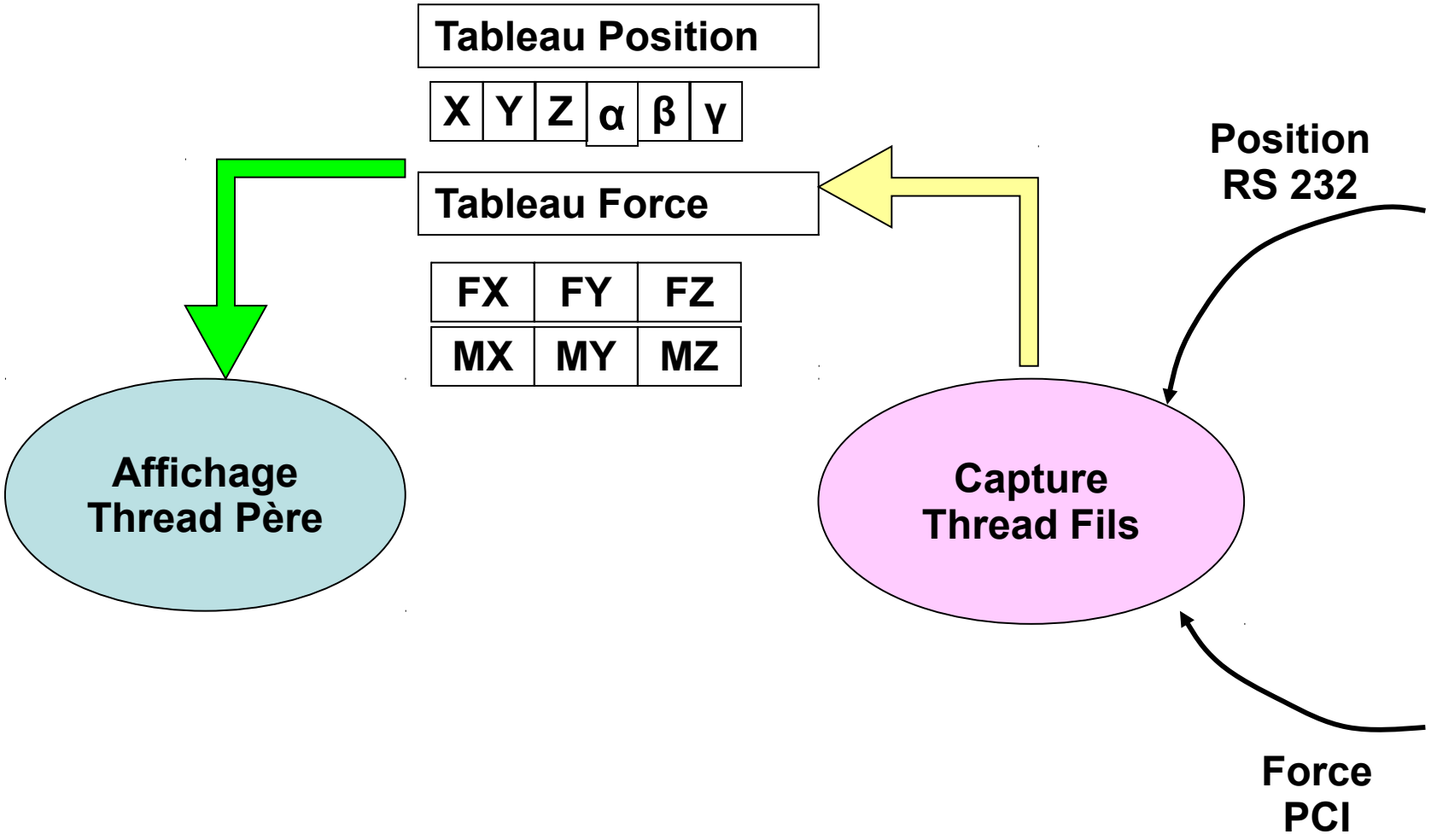
## Capture de gestes chirurgicaux



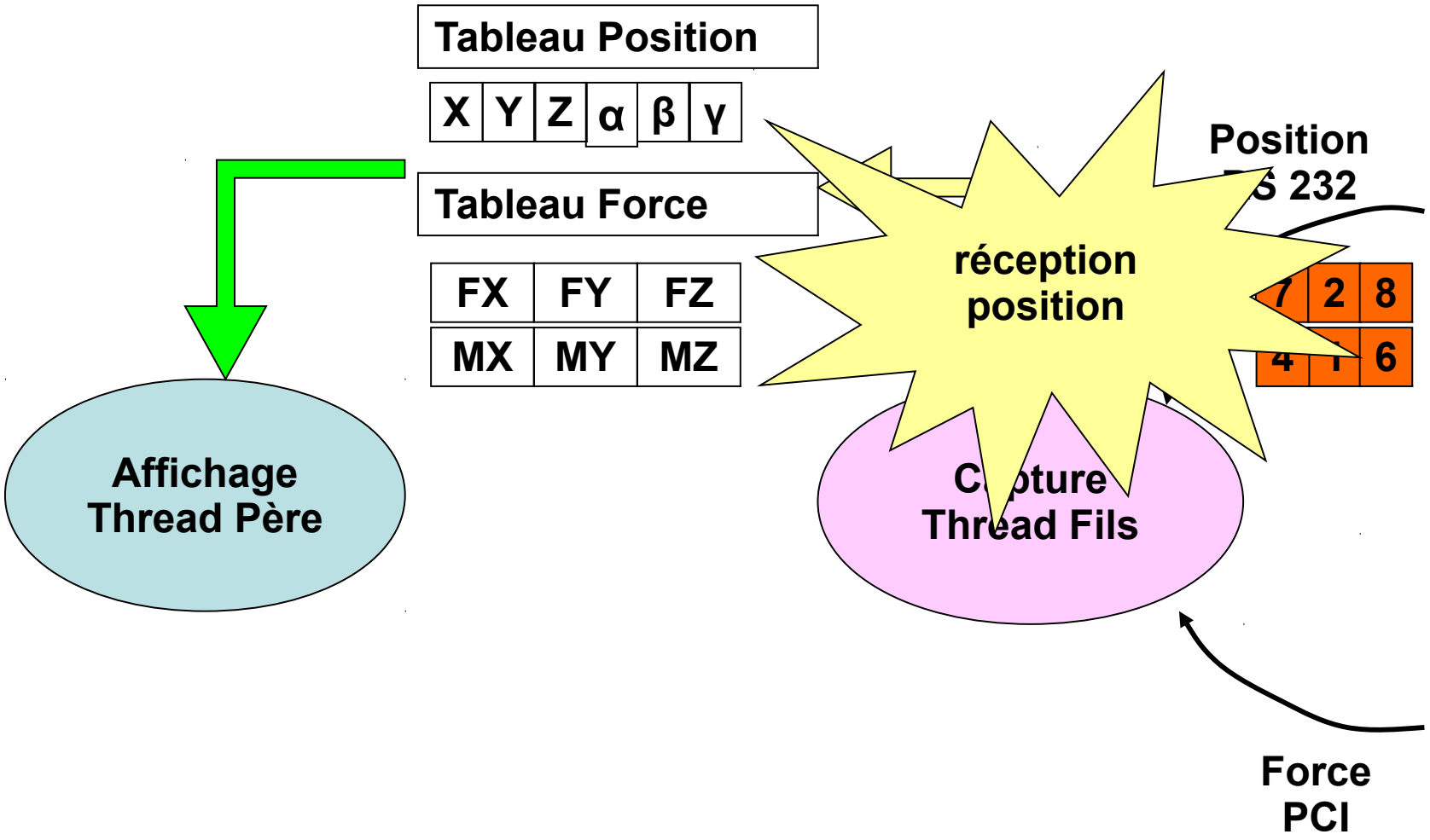
# Exemple industriel (2)



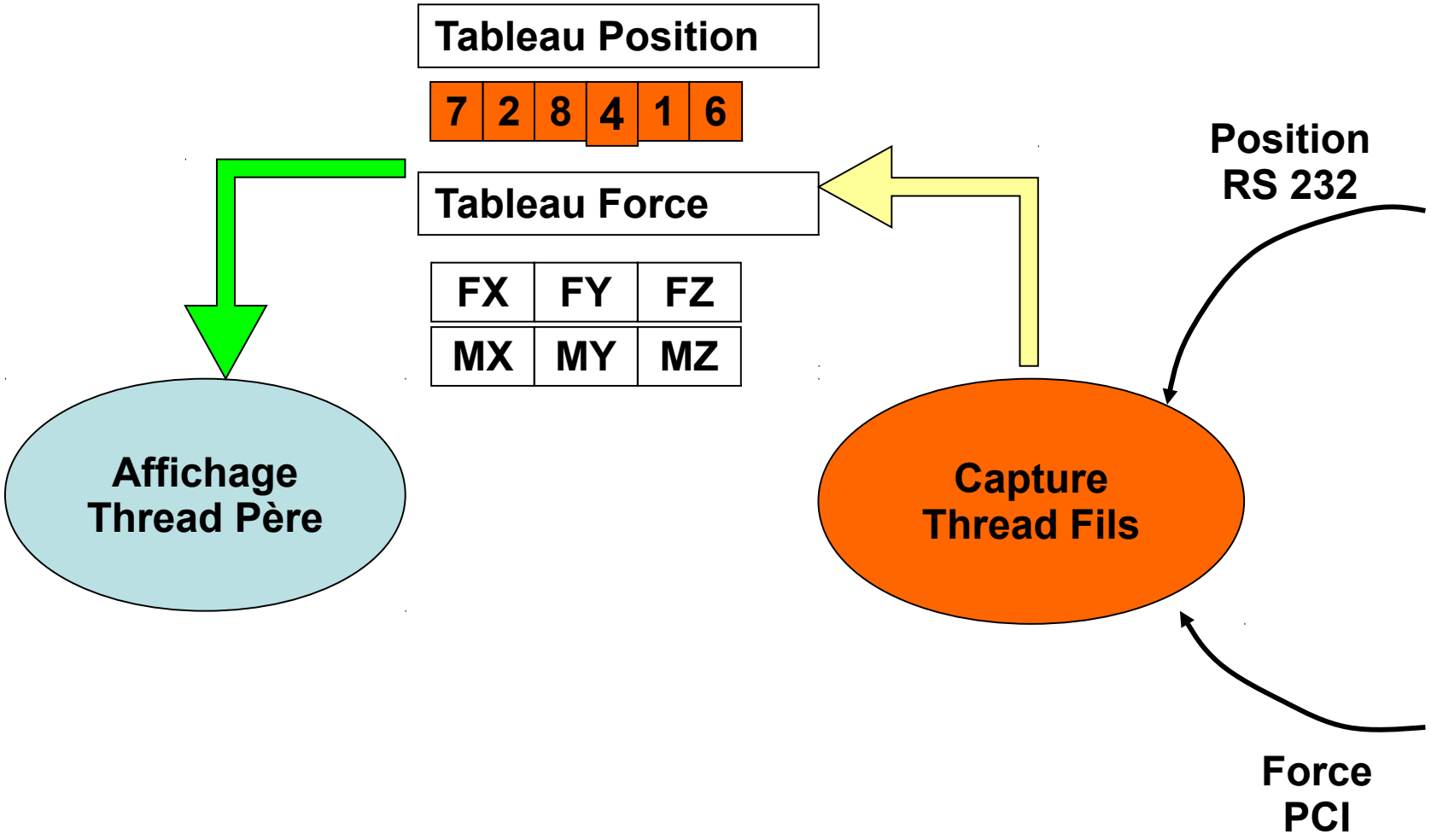
# Exemple industriel (3) Sans synchro.



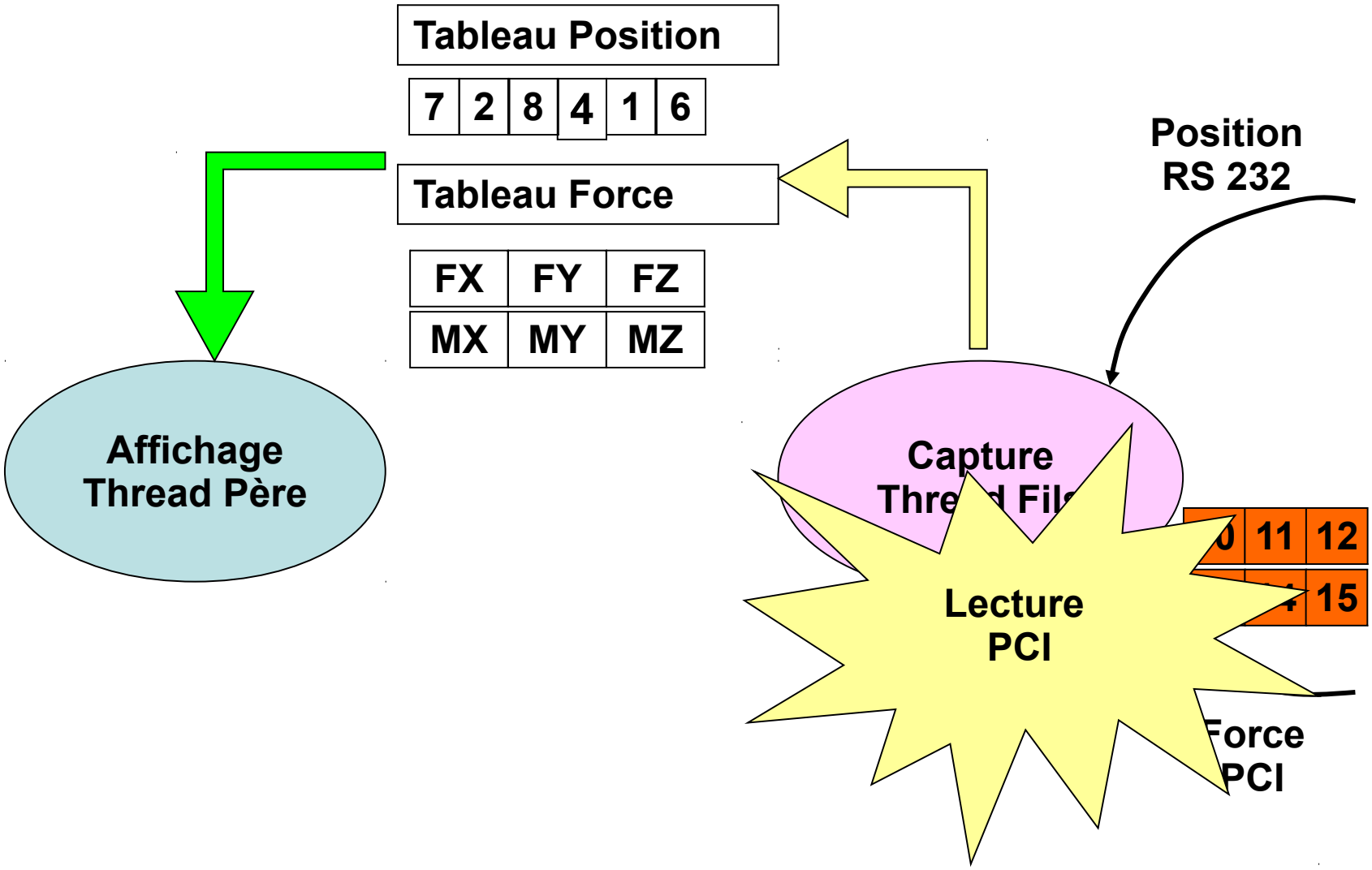
# Exemple industriel (4) Sans synchro



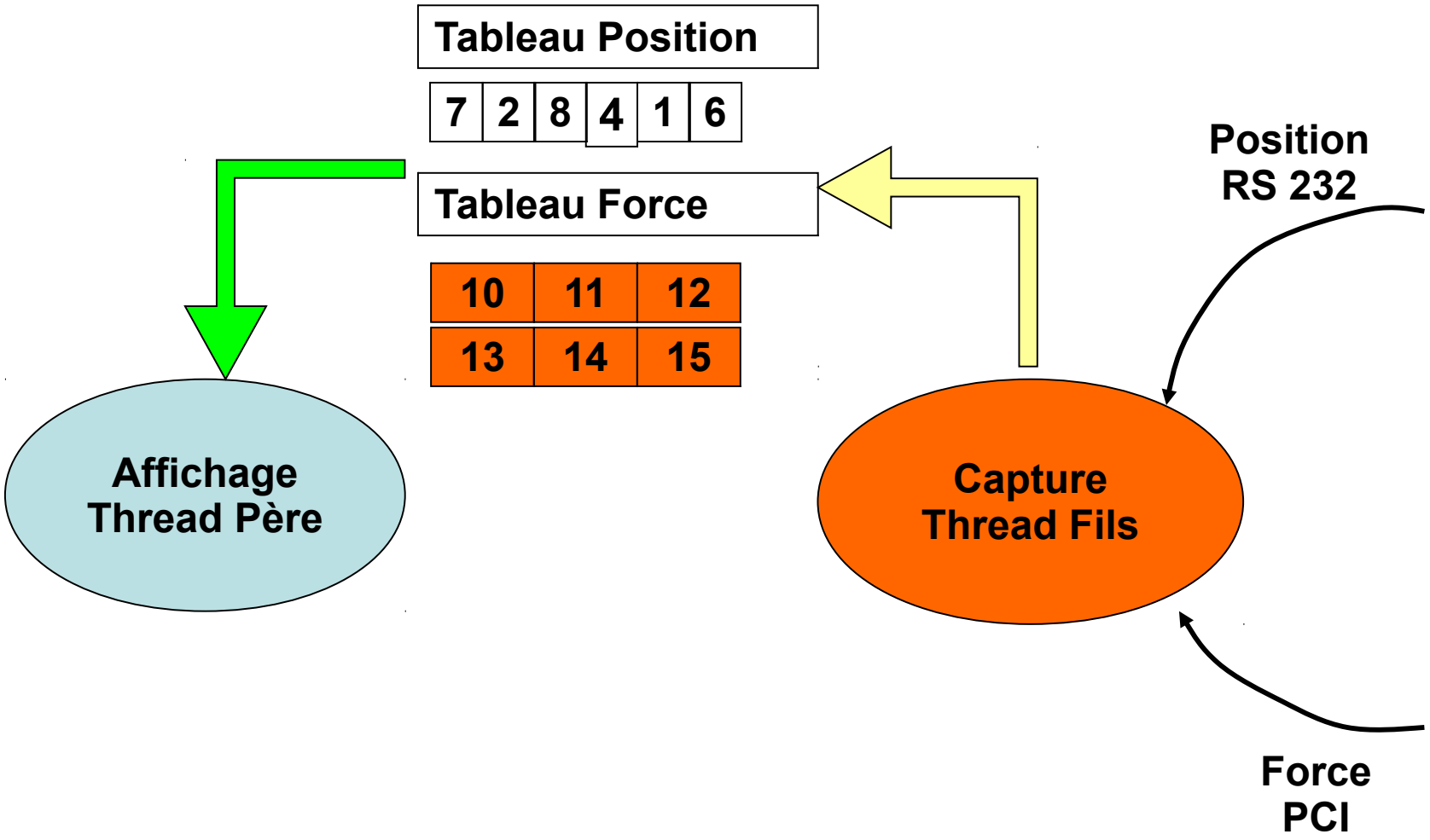
# Exemple industriel (5) Sans synchro



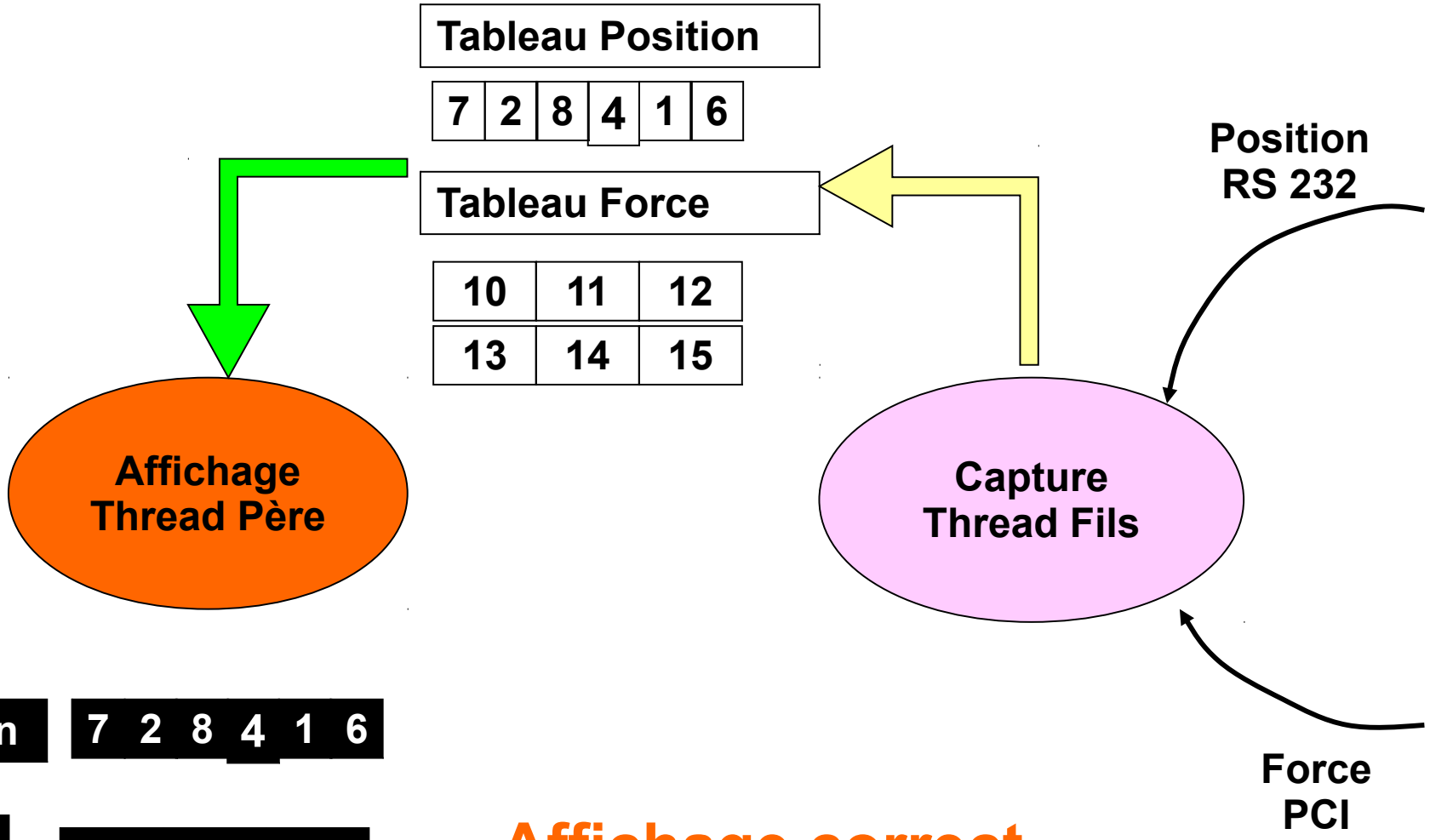
# Exemple industriel (6) Sans synchro



# Exemple industriel (7) Sans synchro



# Exemple industriel (8) Sans synchro

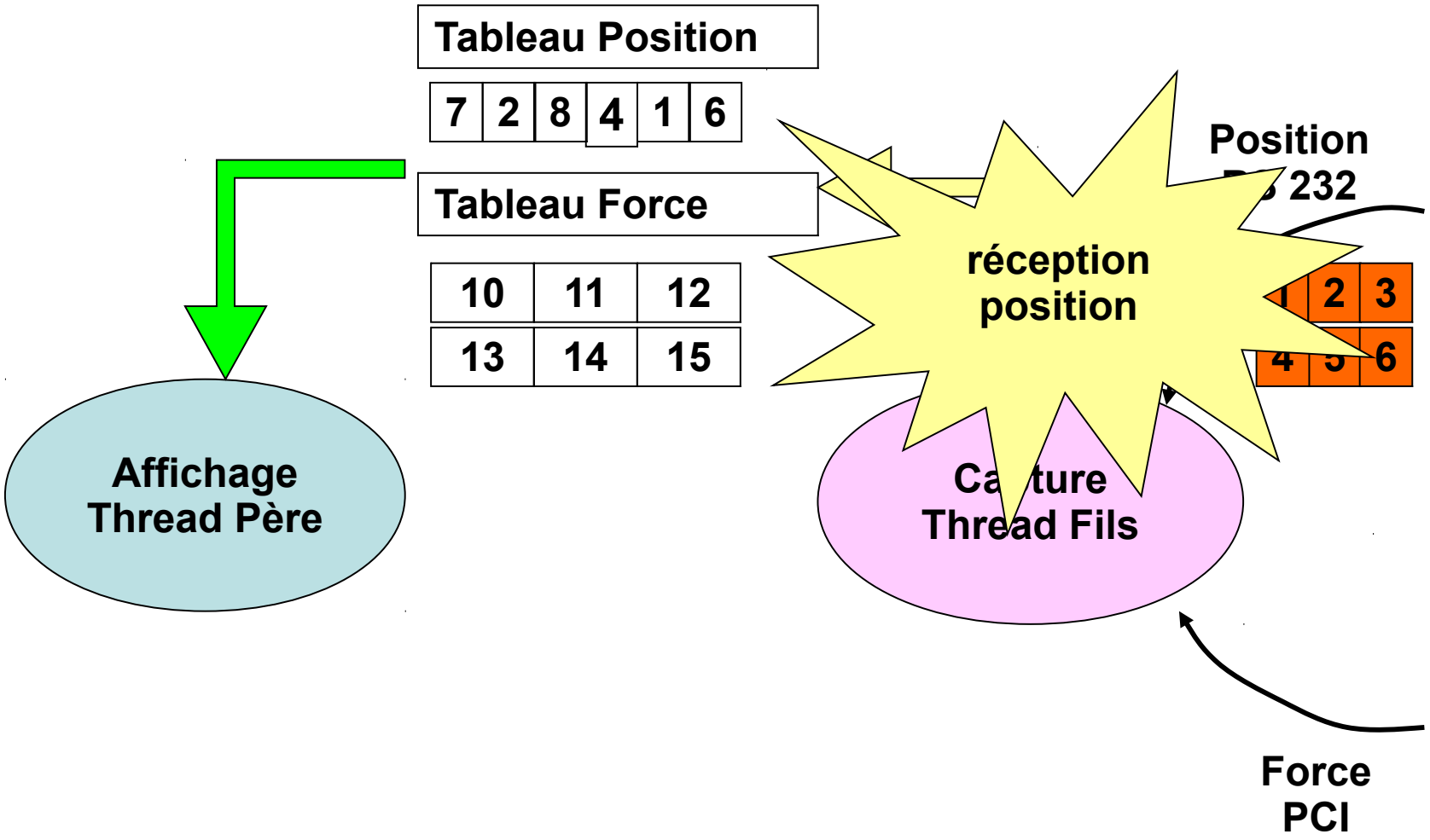



| Position | 7 | 2 | 8 | 4 | 1 | 6 |
|---|---|---|---|---|---|---|

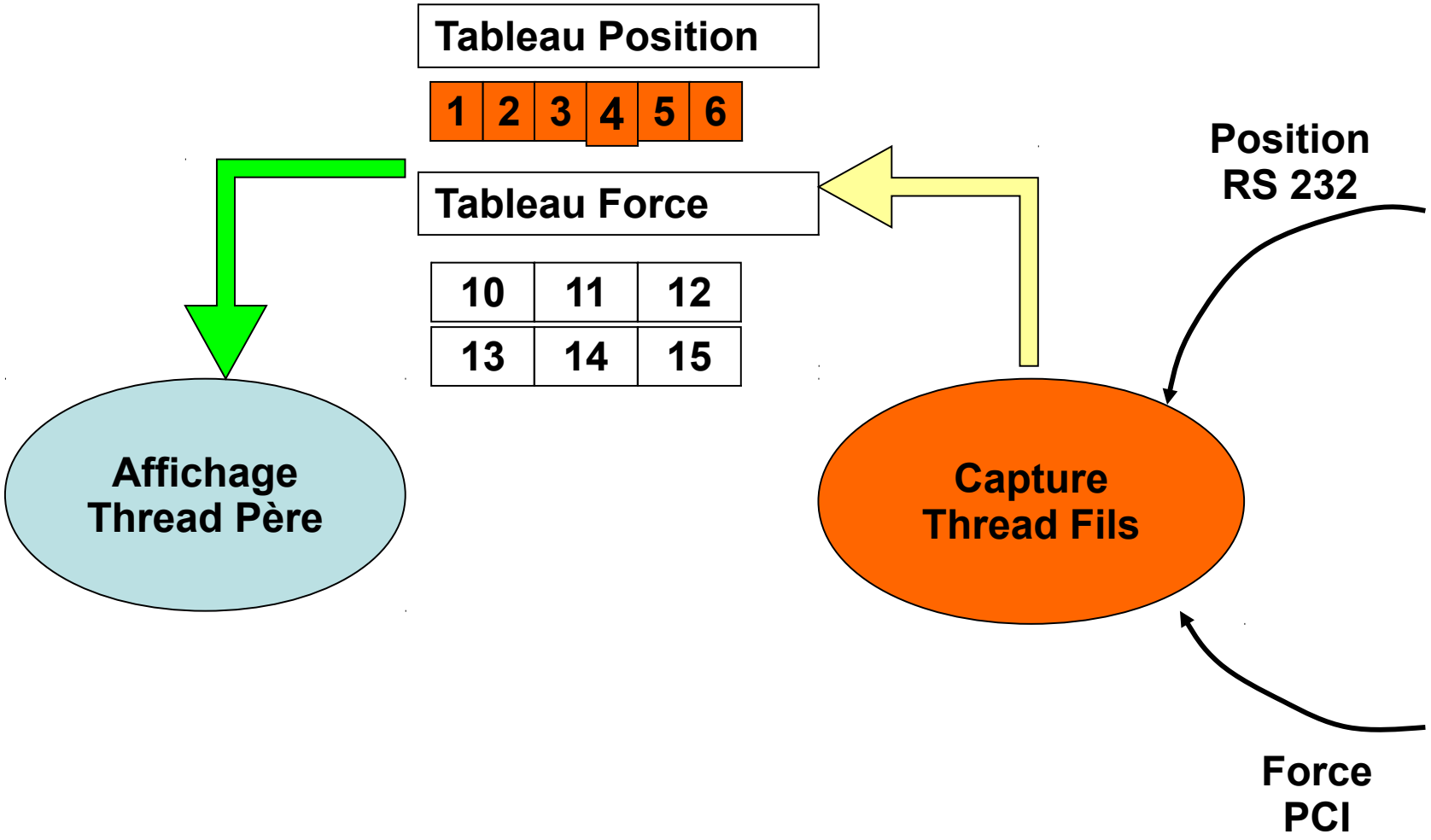
| Force | 10 | 11 | 12 |
|---|---|---|---|
| | 13 | 14 | 15 |

**Affichage correct**

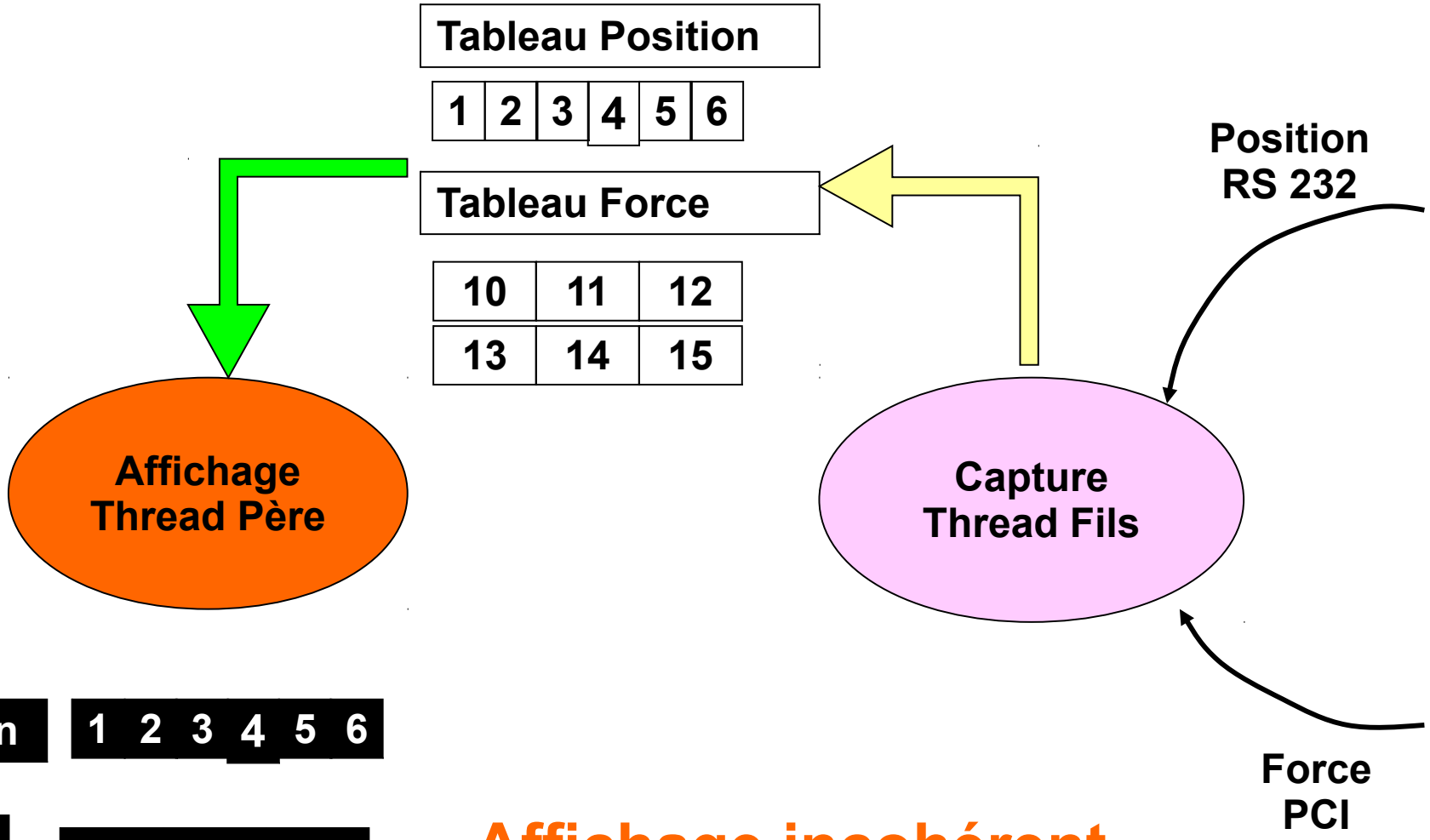
# Exemple industriel (9) Sans synchro



# Exemple industriel (10) Sans synchro



# Exemple industriel (11) Sans synchro




| Position | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|

| Force | | | |
|---|---|---|---|
| | 10 | 11 | 12 |
| | 13 | 14 | 15 |

**Affichage incohérent**

# Exemple industriel (12)

- Problème:
  - Pas de synchronisation entre les deux threads.
  - Affichage aléatoire.

- Besoin de synchronisation

# La synchronisation

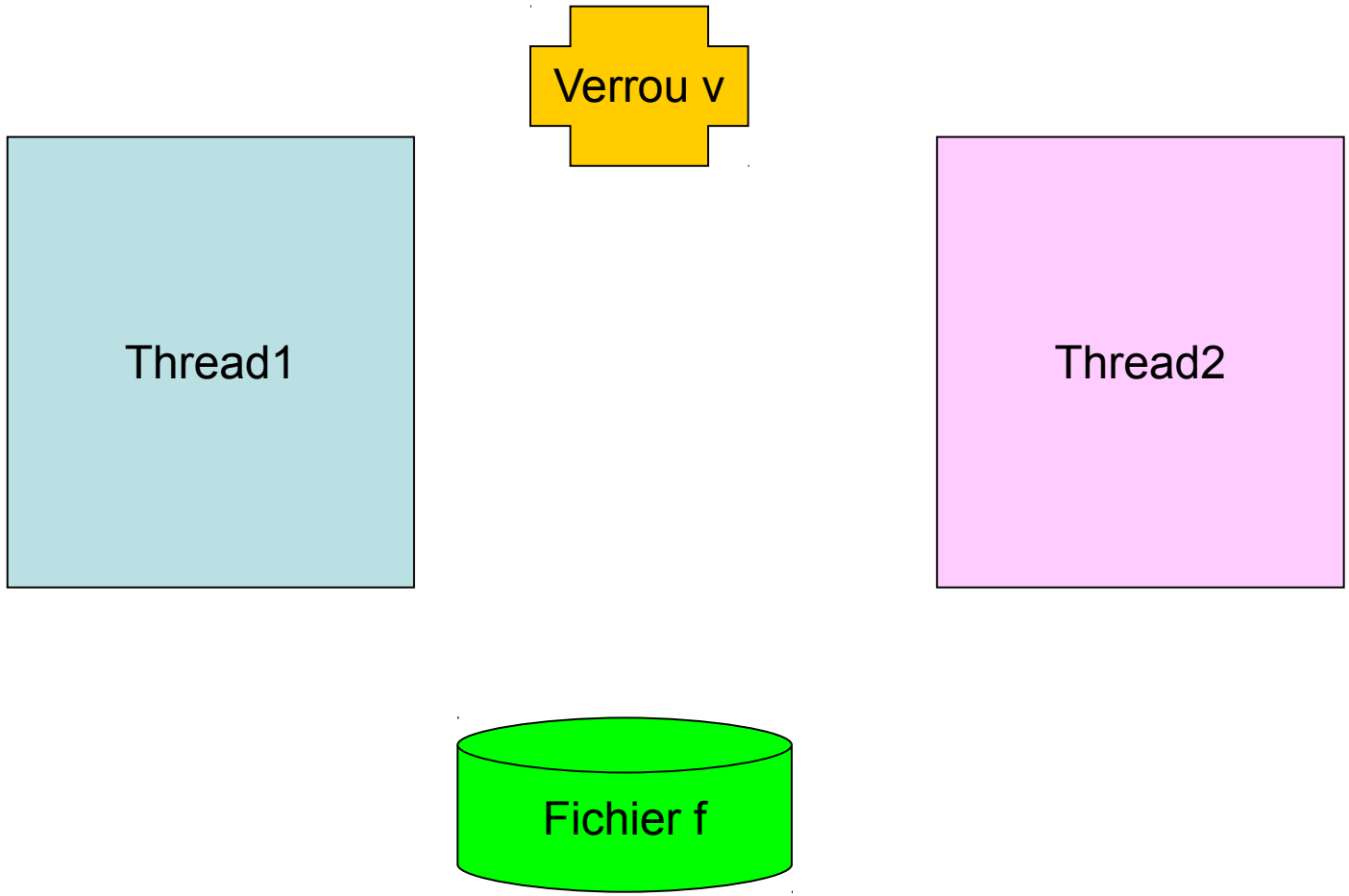
- Une solution :
  - Synchronisation par Verrou.

# Les Verrous

- Le verrou va permettre de synchroniser l'accès à une ressource.
- Le verrou est un outil simple pour qu'une ressource ne soit pas utilisée en même temps par deux threads.
- le verrou doit être rendu par le thread qui l'a pris.

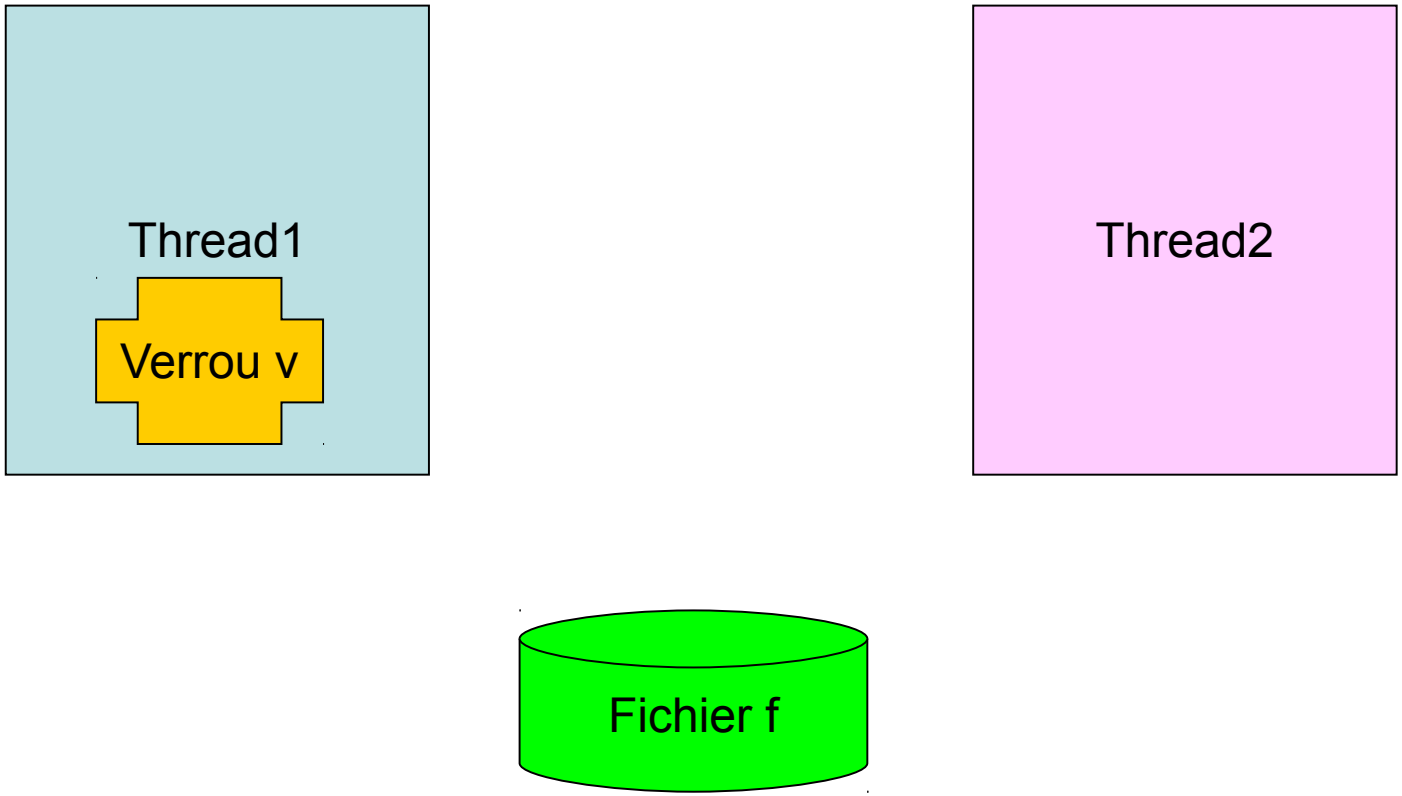
# Les Verrous (1)

**Le verrou v est associé à l'écriture dans le fichier f.**



# Les Verrous (2)

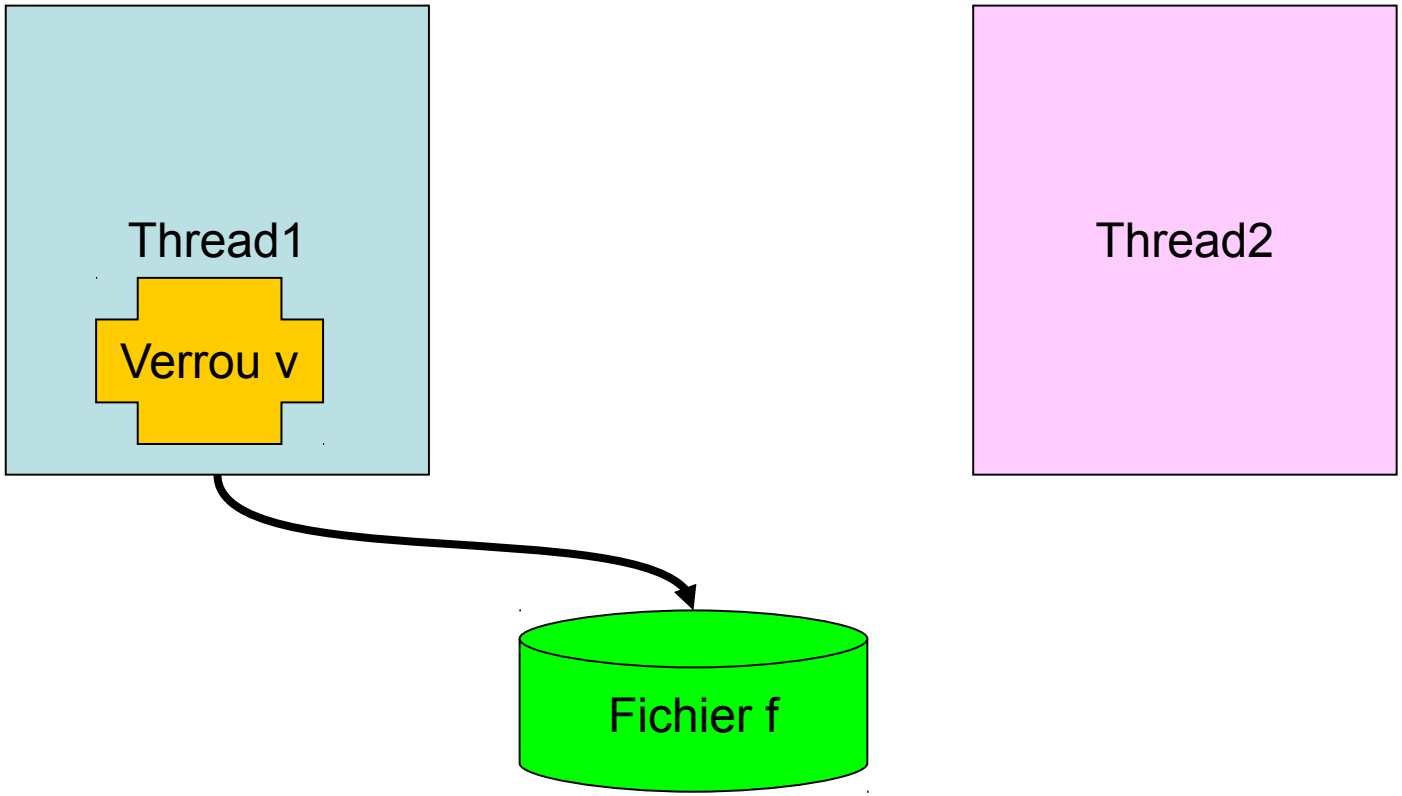
**Verrouillage du verrou v par la thread 1**



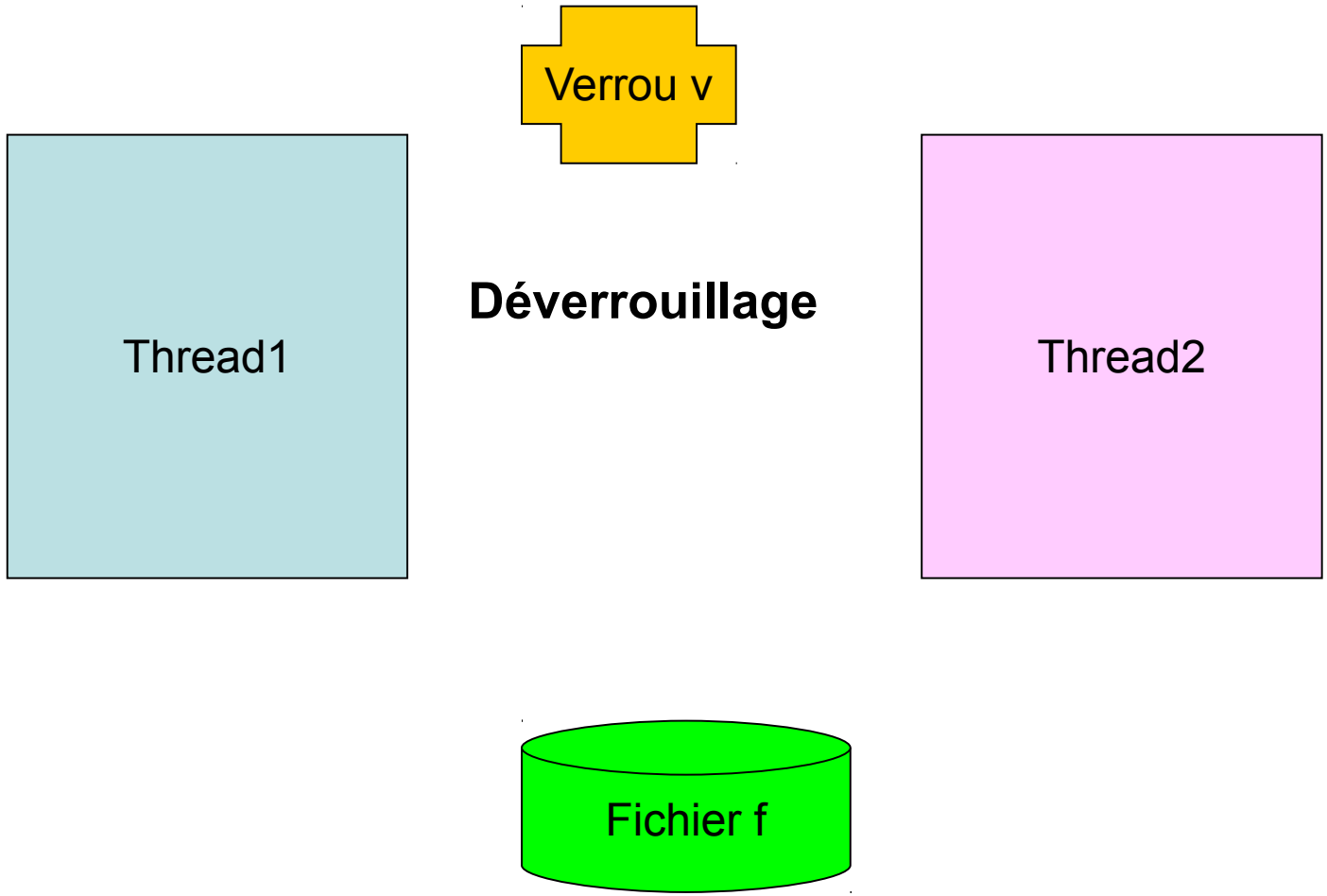
# Les Verrous (3)

**écriture dans le fichier**

**La thread 2 doit attendre le déverrouillage du verrou v pour écrire dans le fichier f.**

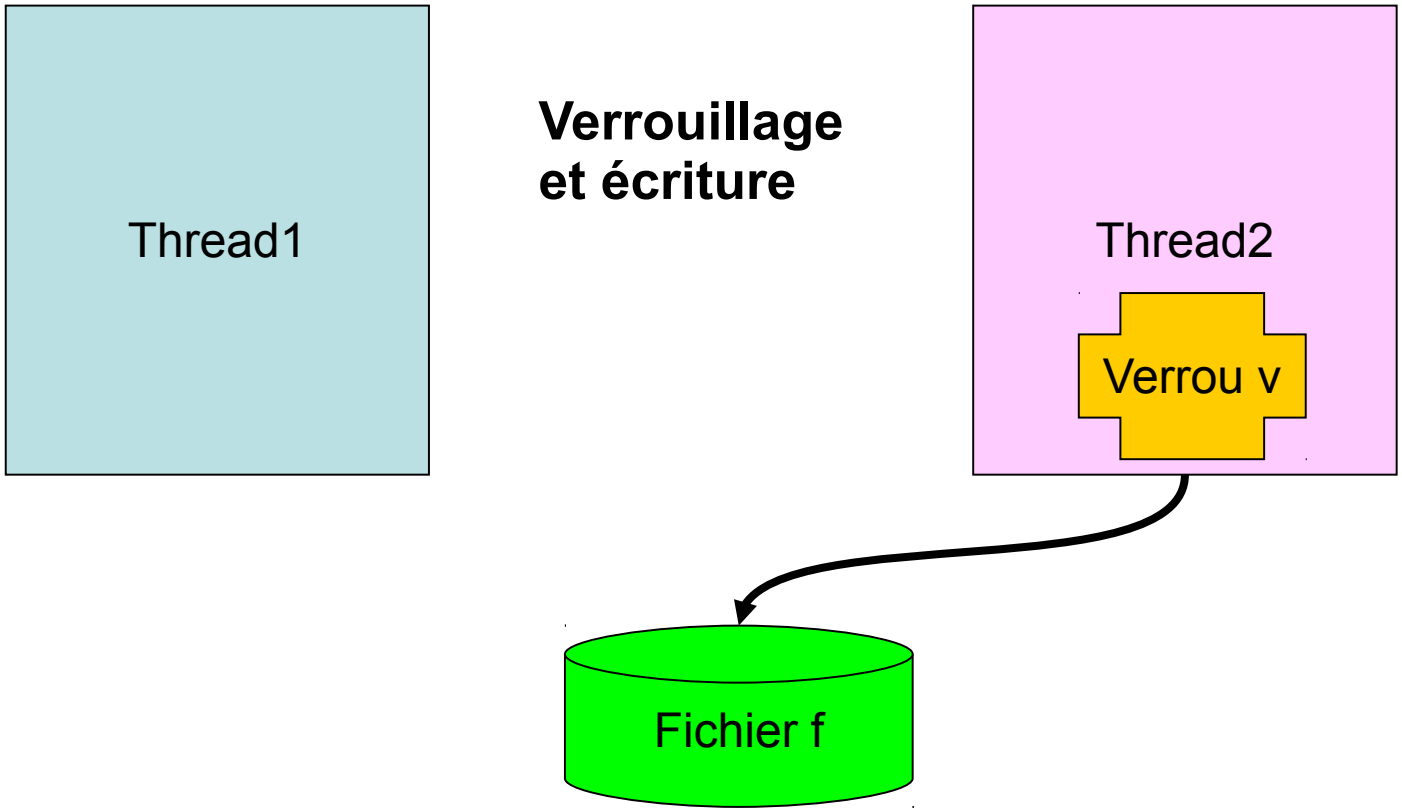


# Les Verrous (4)



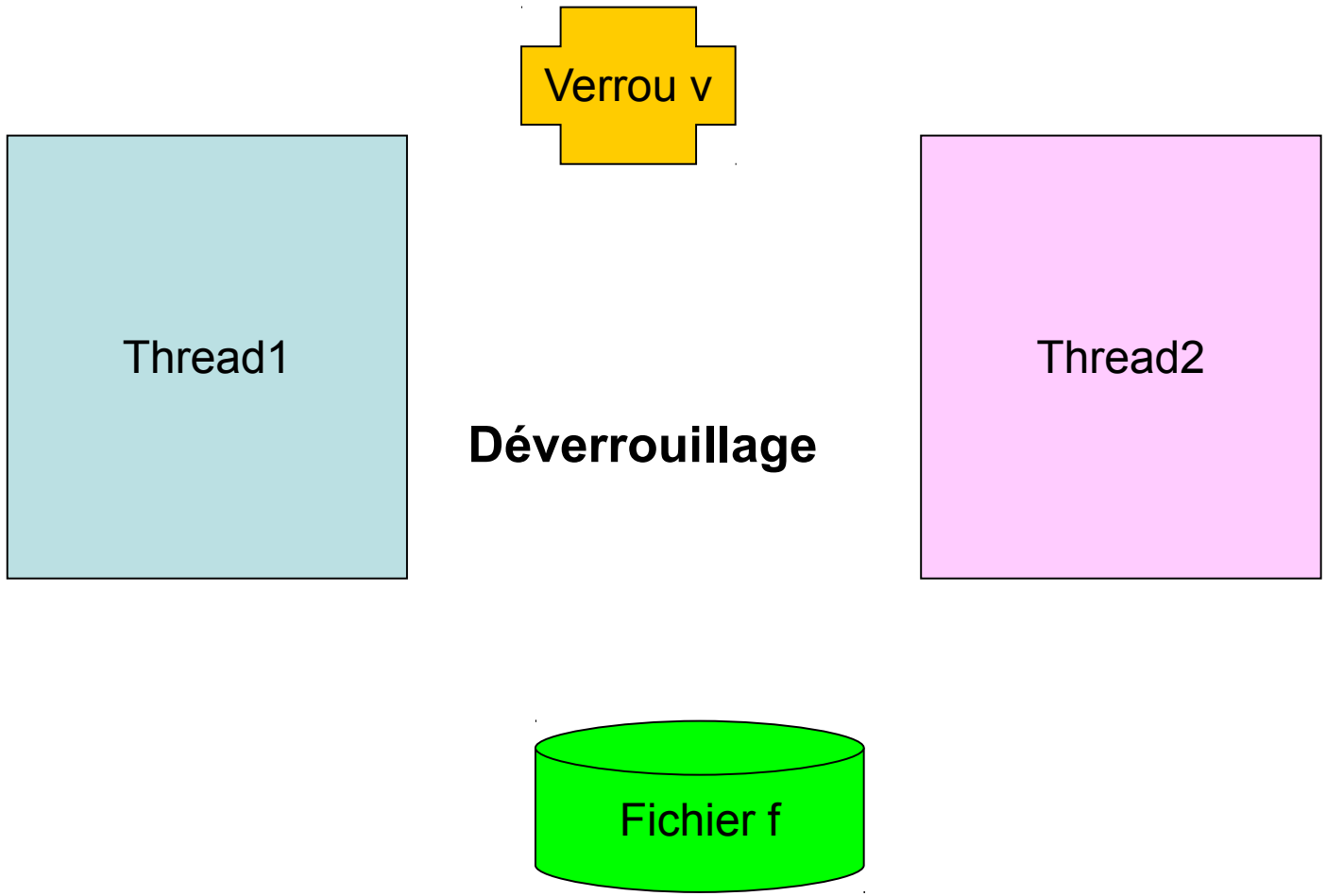
Prod. Alain MENU
Lycée La Fayette - 0770920G
iRIS
Informatique et Réseaux pour l'Industrie et les Services techniques

# Les Verrous (5)

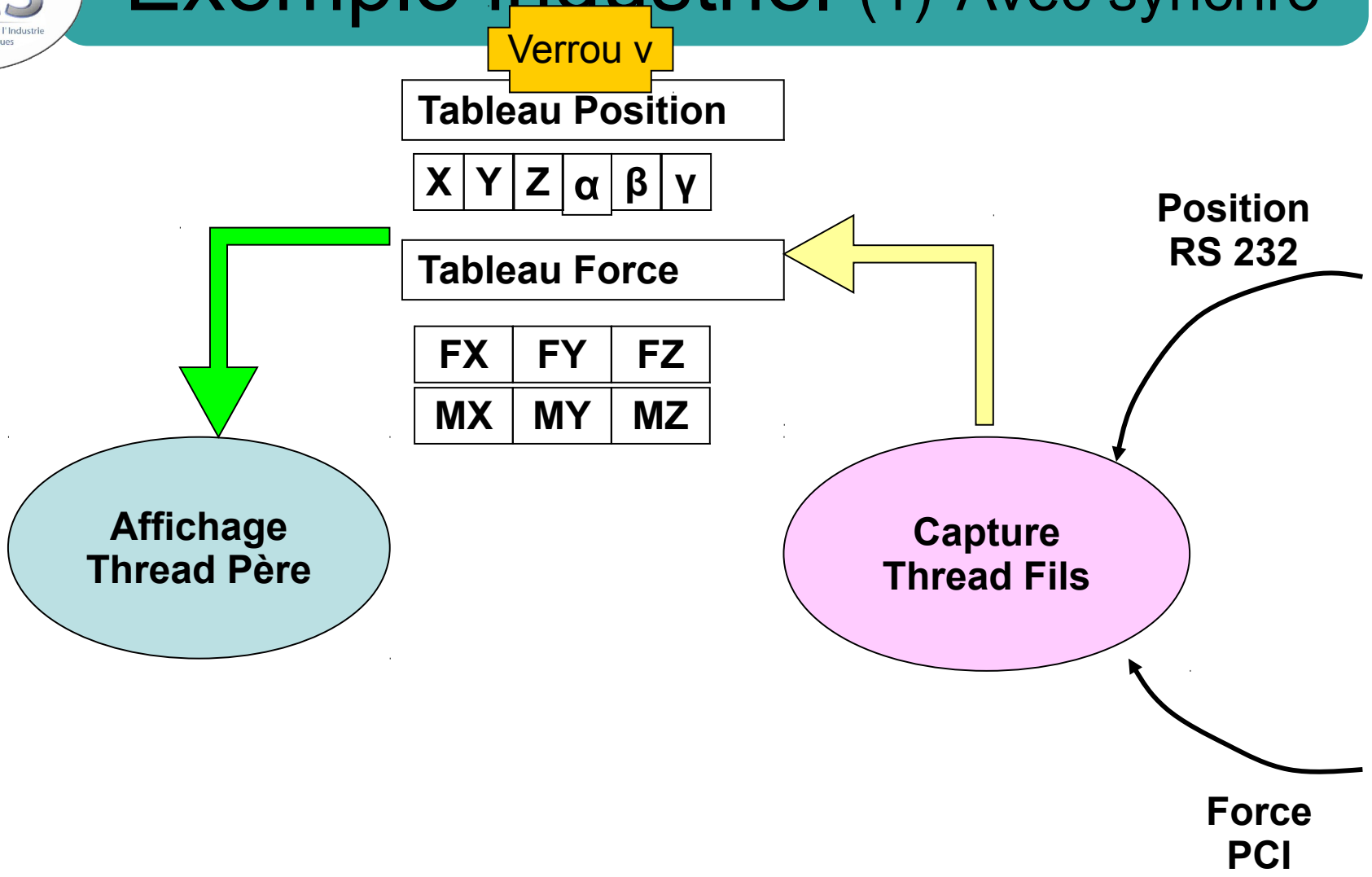


Prod. Alain MENU
Lycée La Fayette - 0770920G
iRiS
Informatique et Réseaux pour l'Industrie et les Services techniques

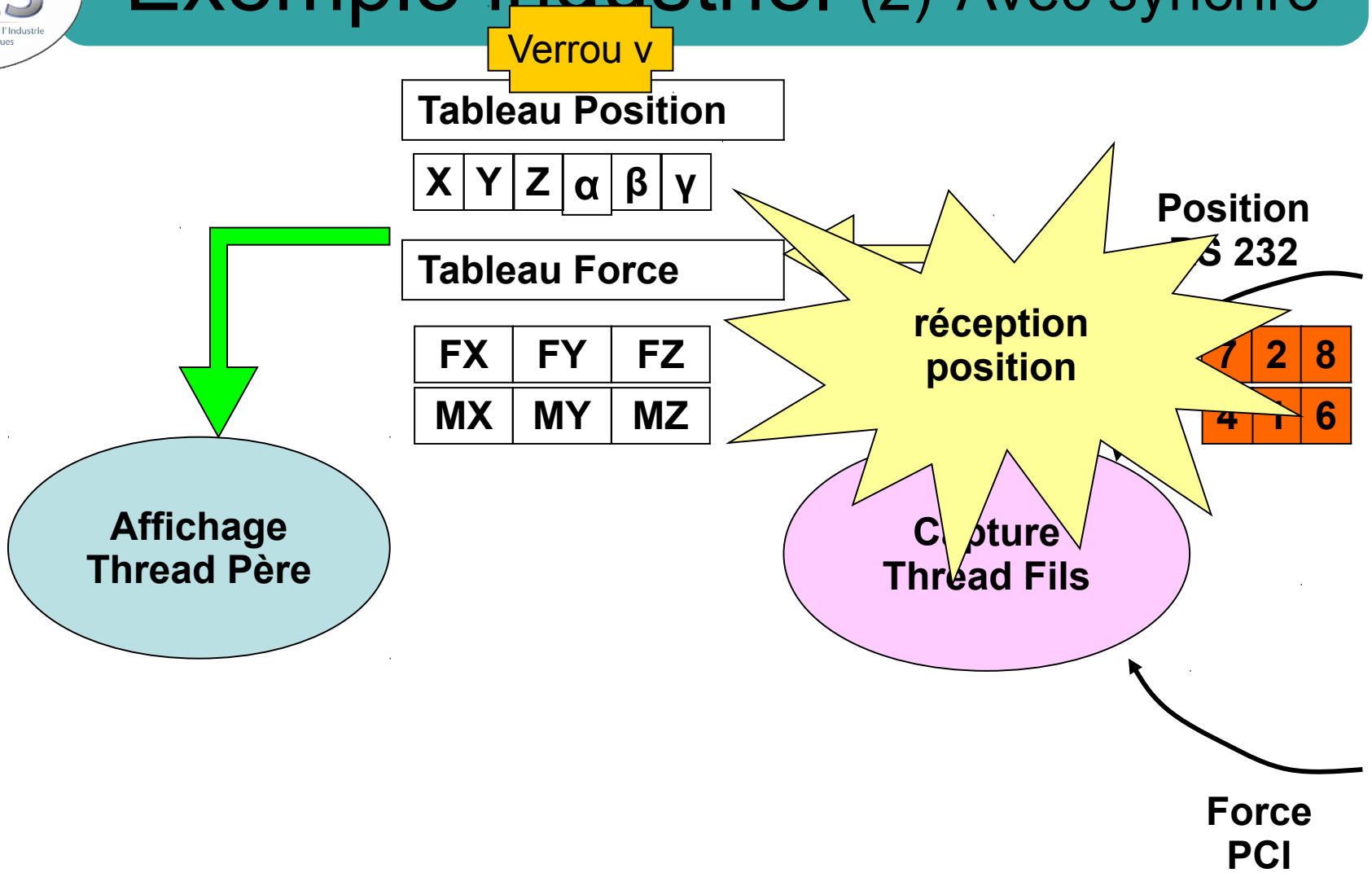
# Les Verrous (6)



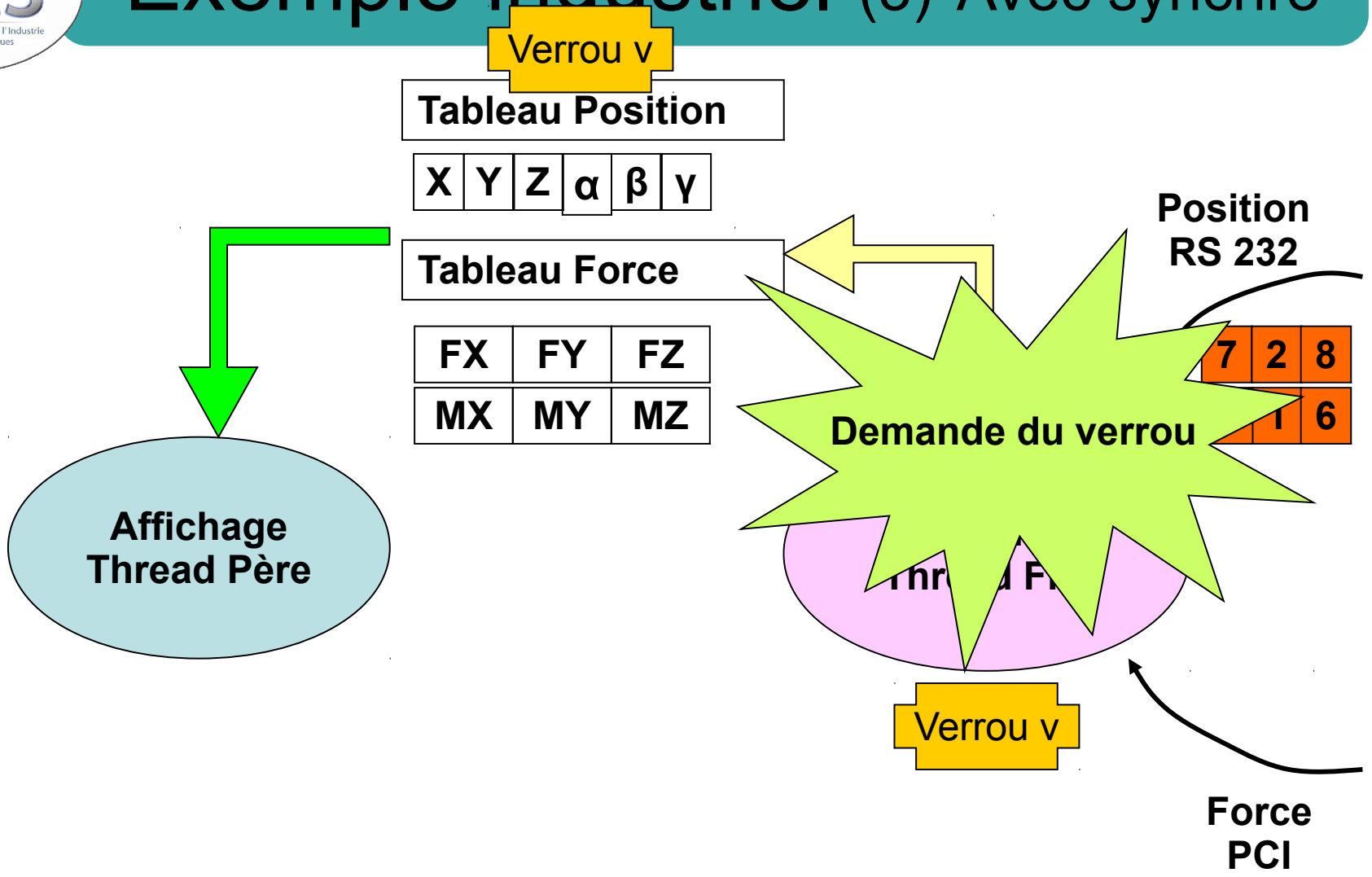
# Exemple industriel (1) Avec synchro



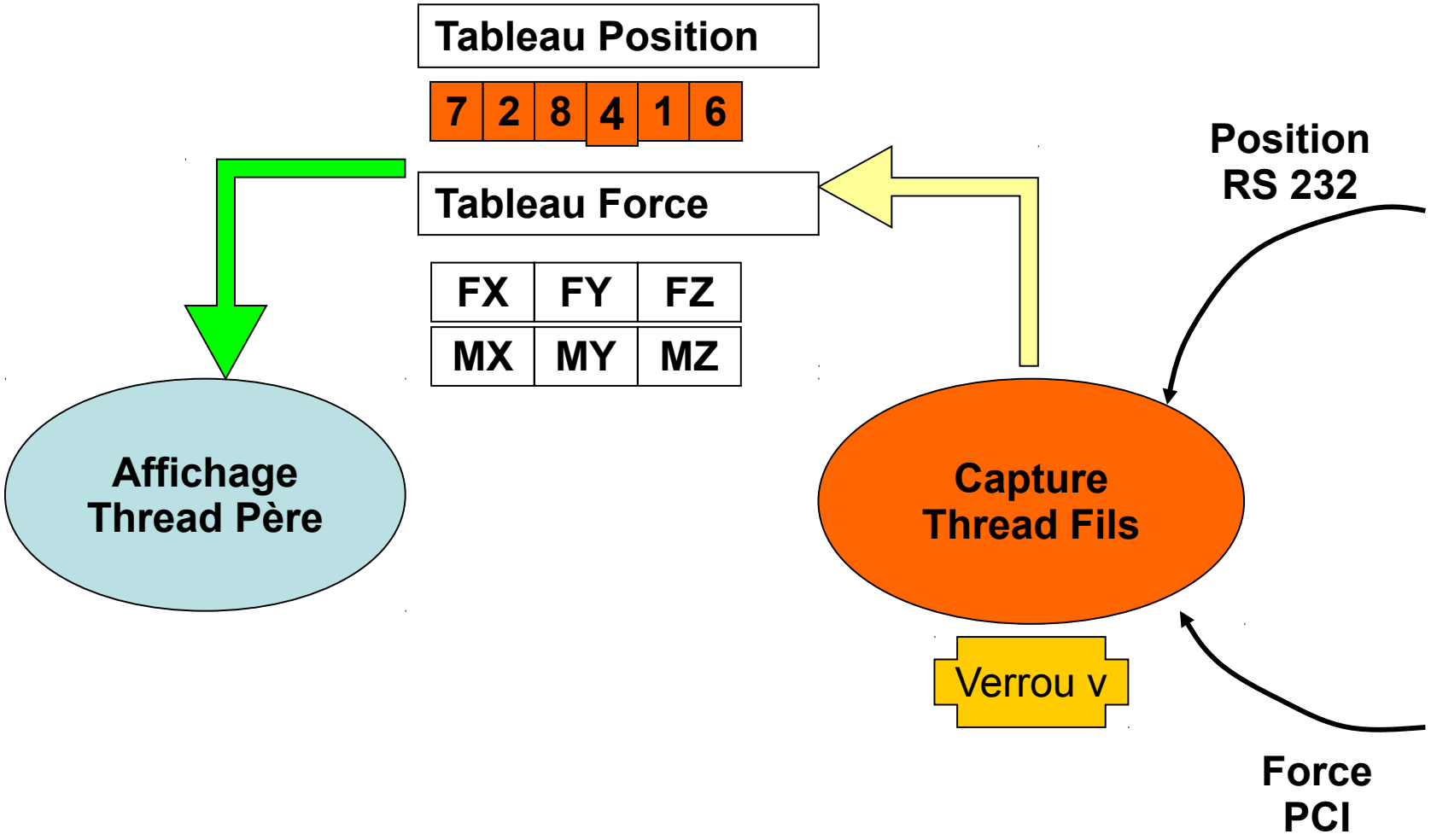
# Exemple industriel (2) Avec synchro



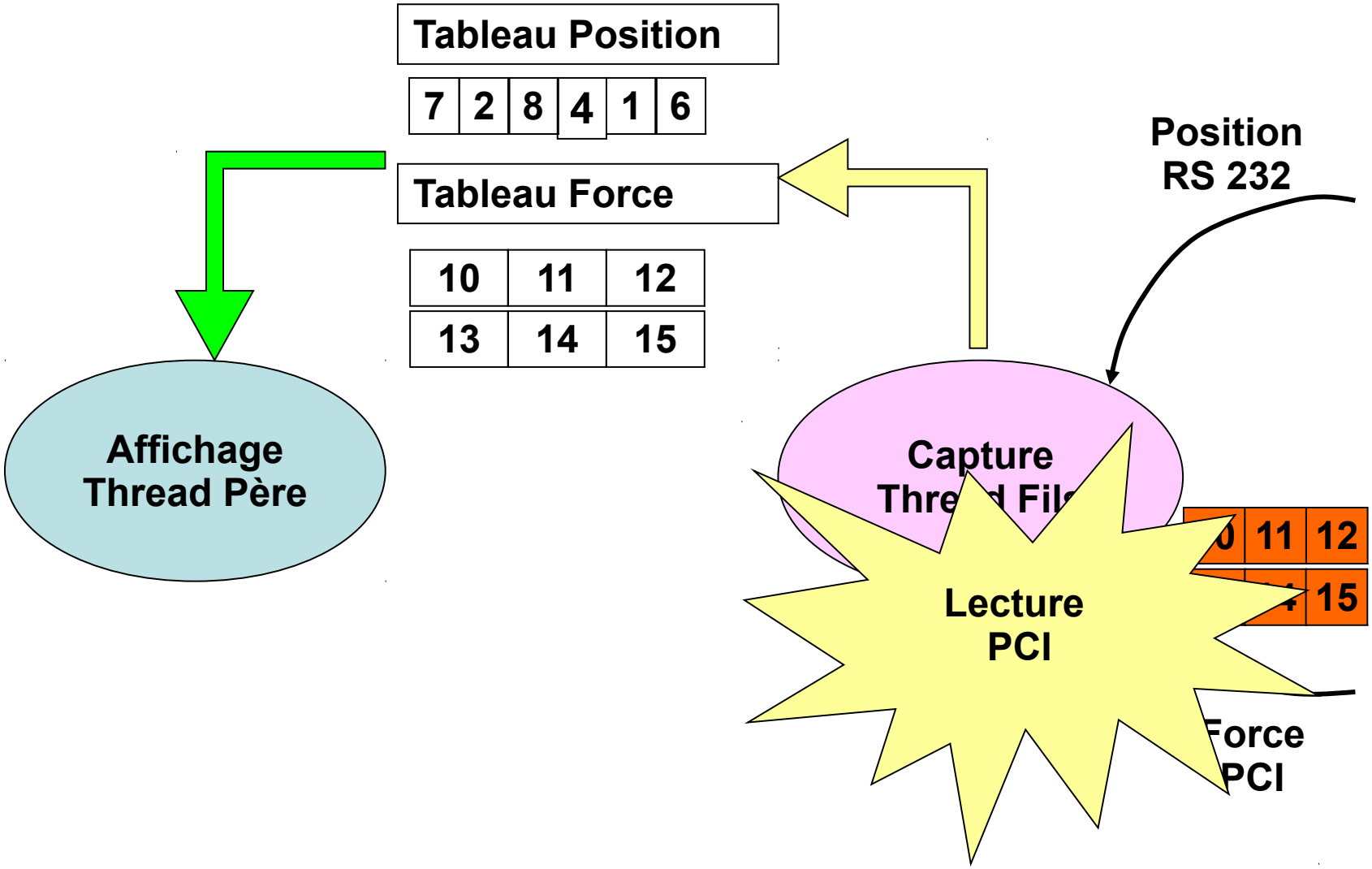
# Exemple industriel (3) Avec synchro



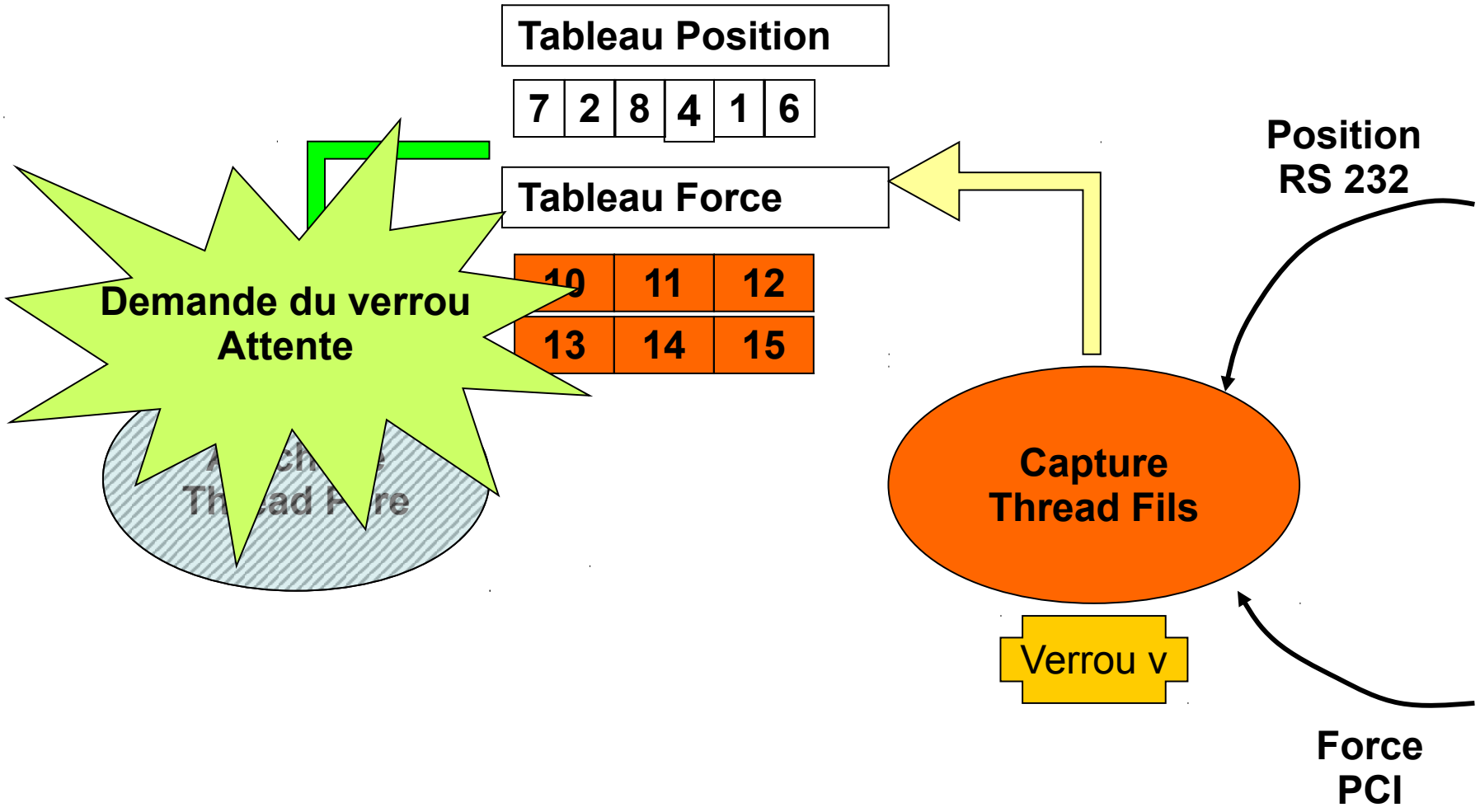
# Exemple industriel (4) Avec synchro



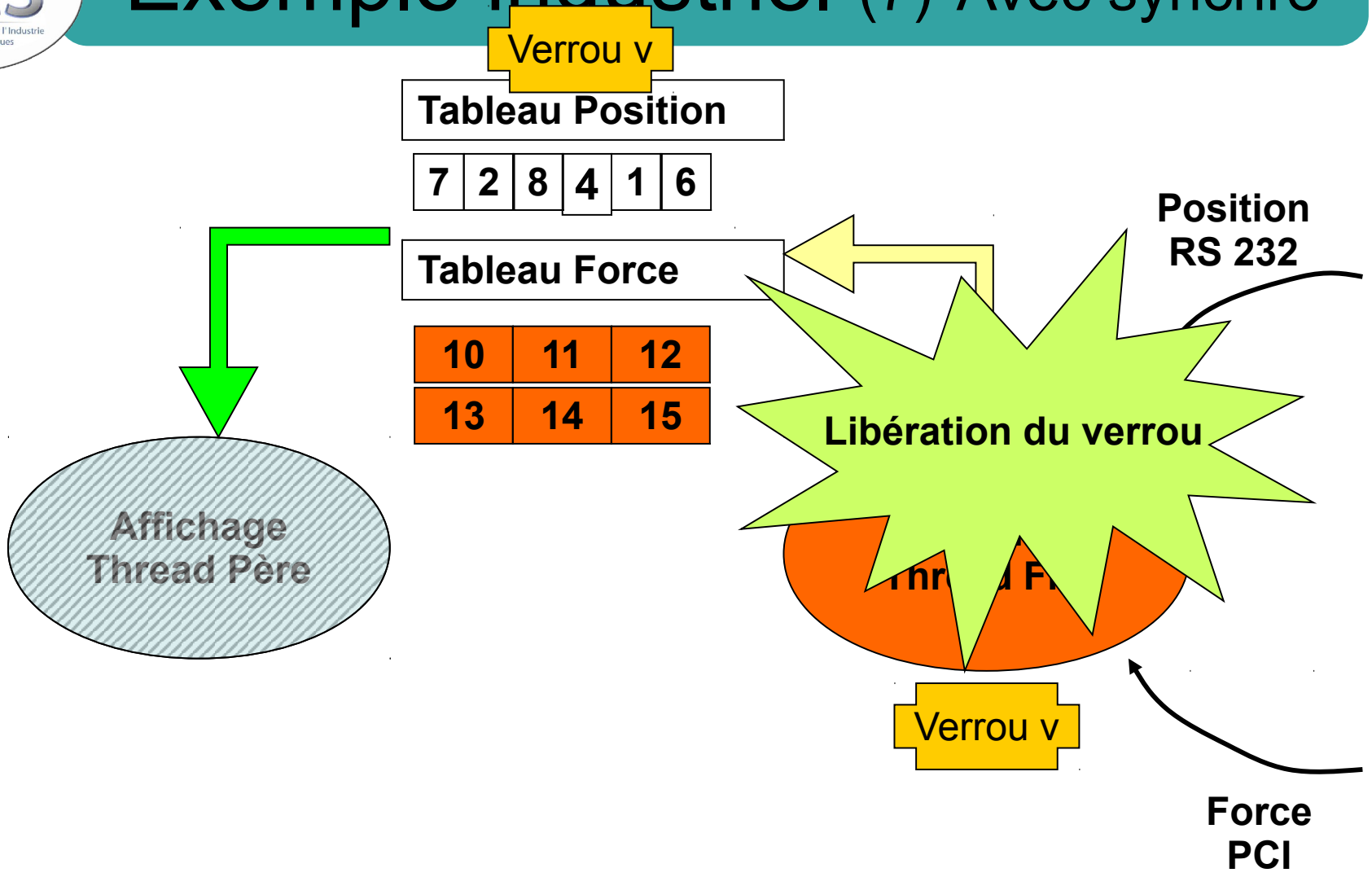
# Exemple industriel (5) Avec synchro



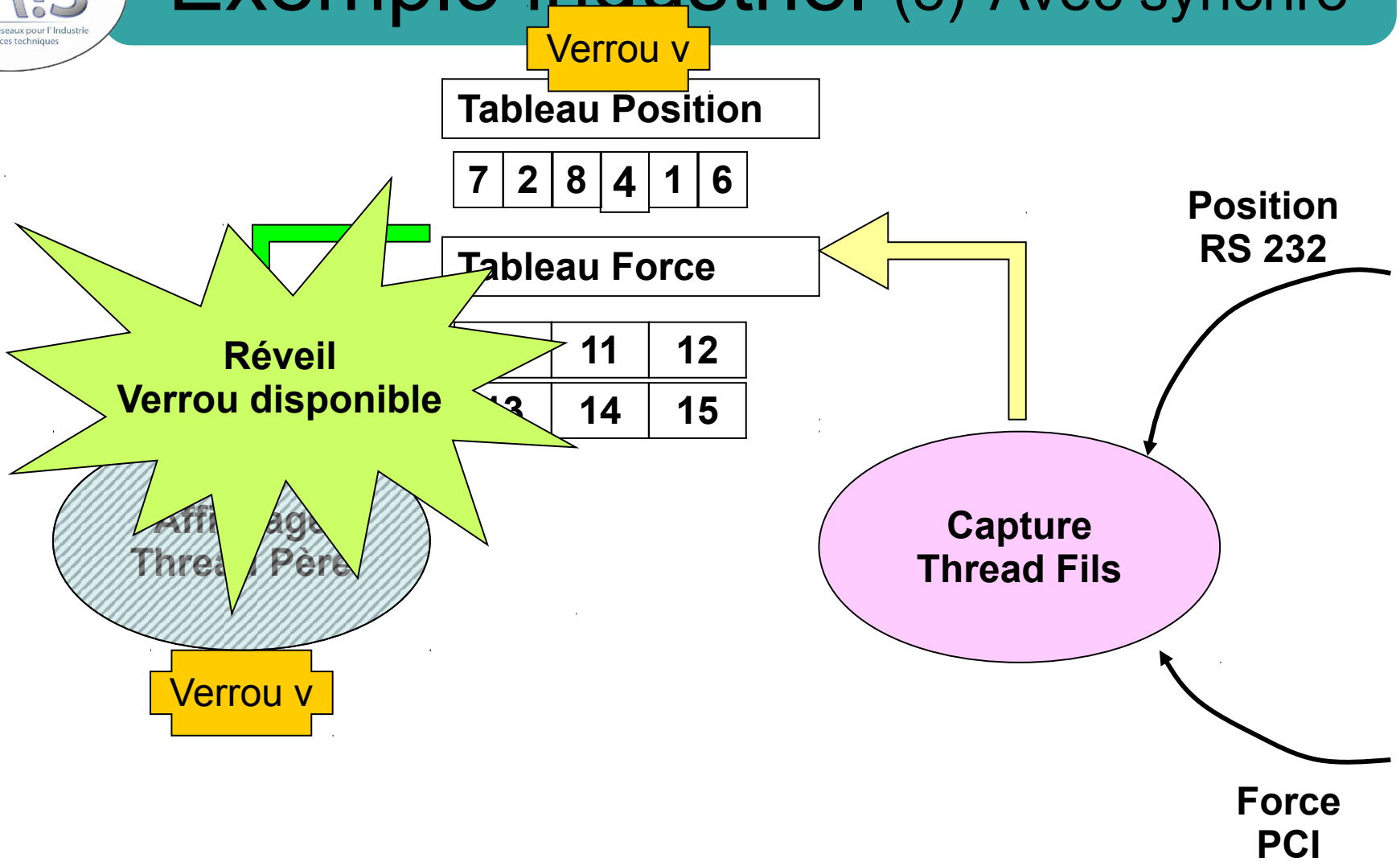
# Exemple industriel (6) Avec synchro



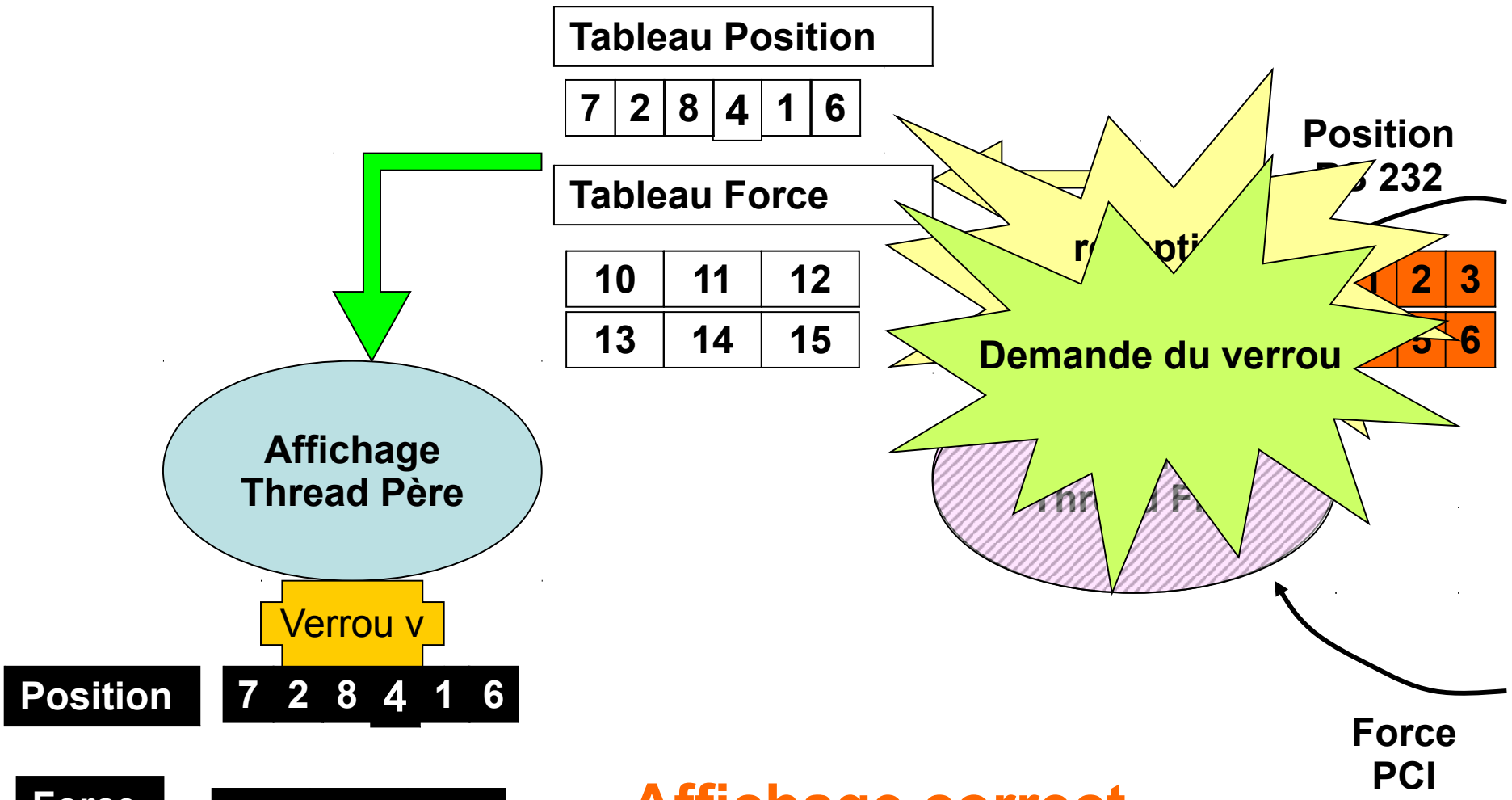
# Exemple industriel (7) Avec synchro



# Exemple industriel (8) Avec synchro

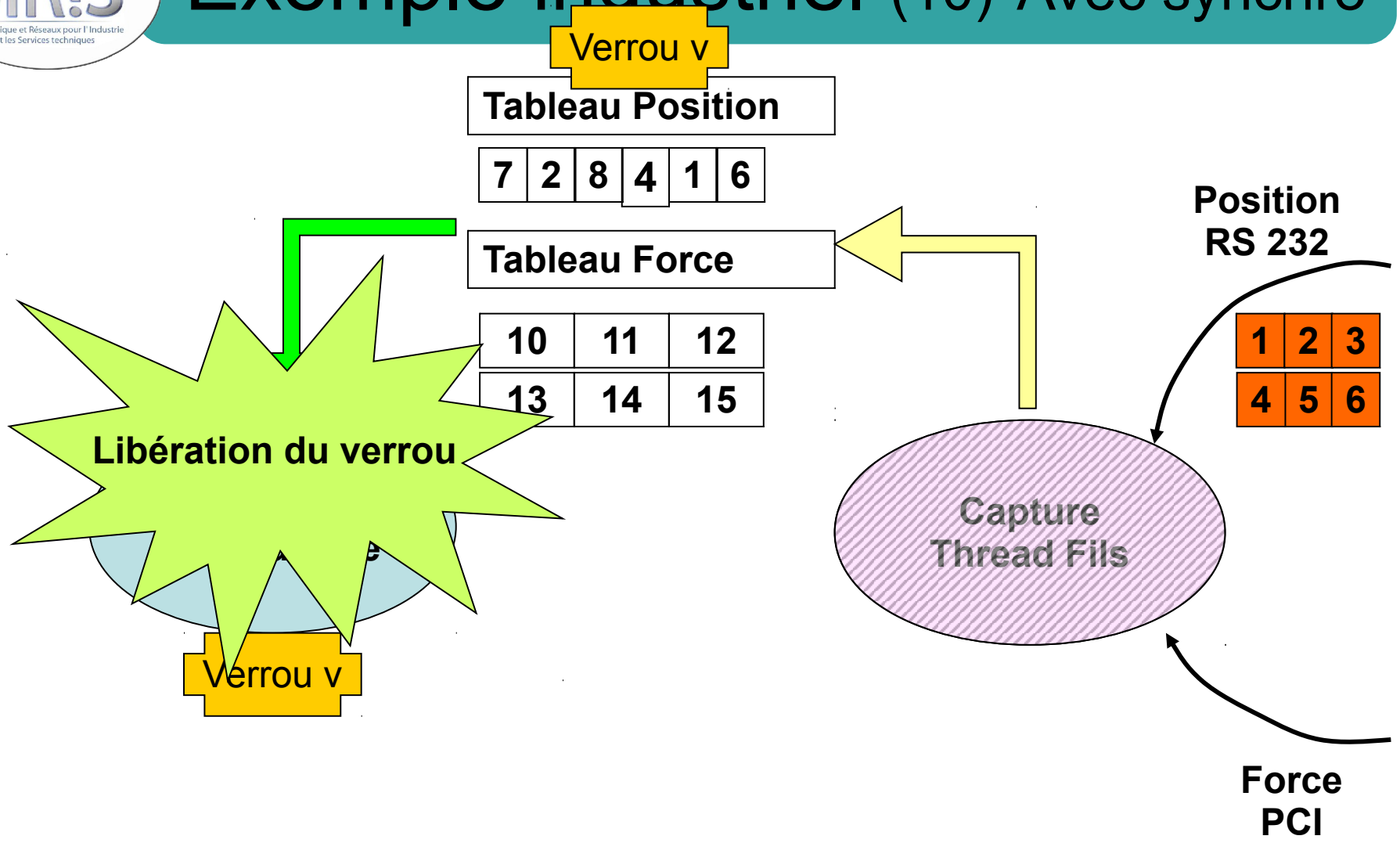


# Exemple industriel (9) Avec synchro

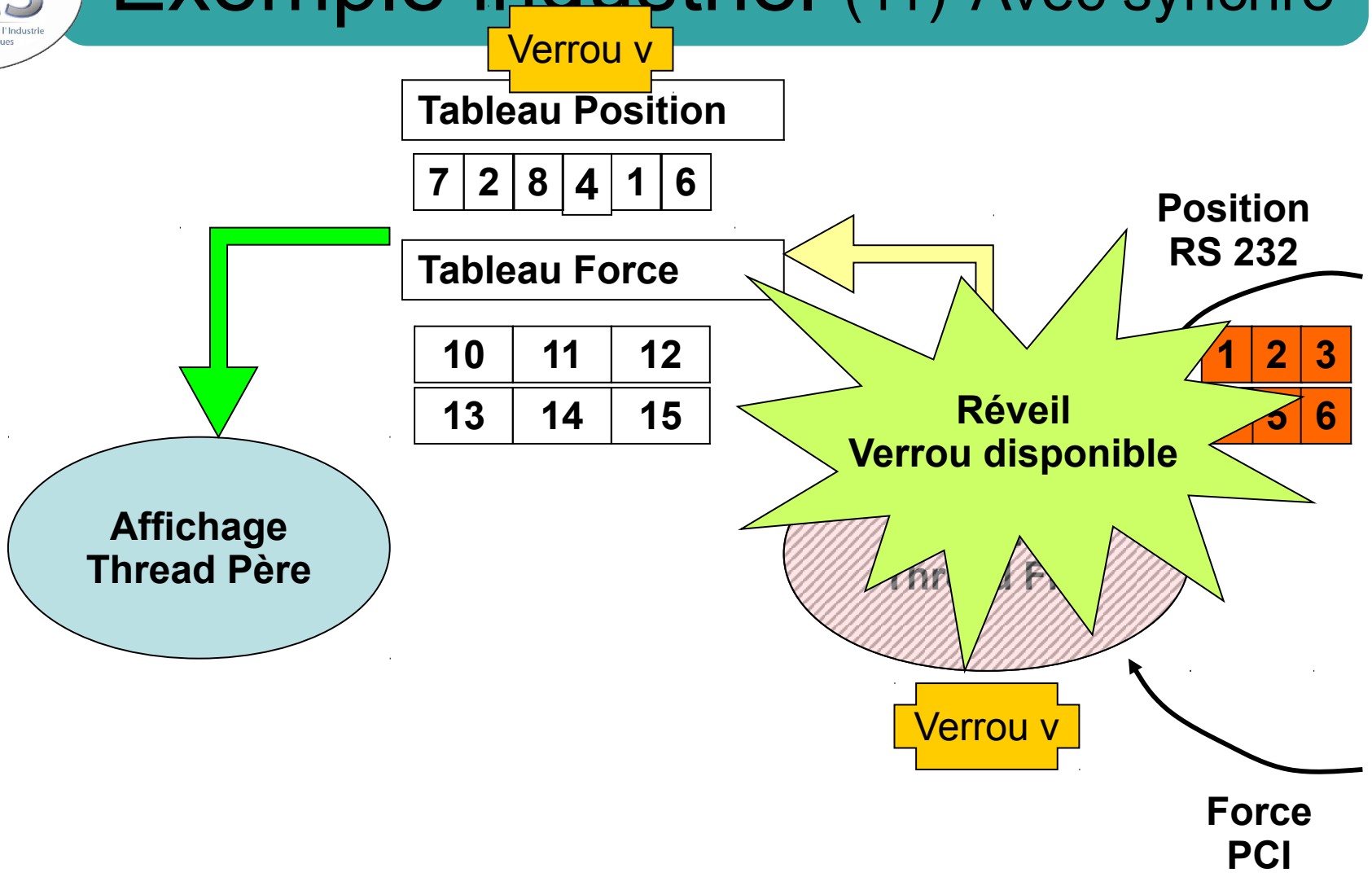



**Affichage correct**

# Exemple industriel (10) Avec synchro



# Exemple industriel (11) Avec synchro



# Les Verrous (généralisation)

- Fonctions utiles aux verrous
  - Création du verrou
  - Destruction du verrou
  - Attente du verrou
  - Prise du verrou
  - Libération du verrou

Fonction unique

# Les Verrous sous Linux

- pthread_mutex_init (...)
  - le verrou est créé et mis à l'état "unlock"
- pthread_mutex_destroy(...)
  - le verrou est détruit
- pthread-mutex_lock (...)
  - si le verrou est déjà pris, le thread est bloqué
- pthread_mutex_unlock (...)
  - rend le verrou et libère
- pthread_mutex_trylock(...)
  - renvoie une erreur si le verrou est déjà pris, le thread n'est pas bloqué

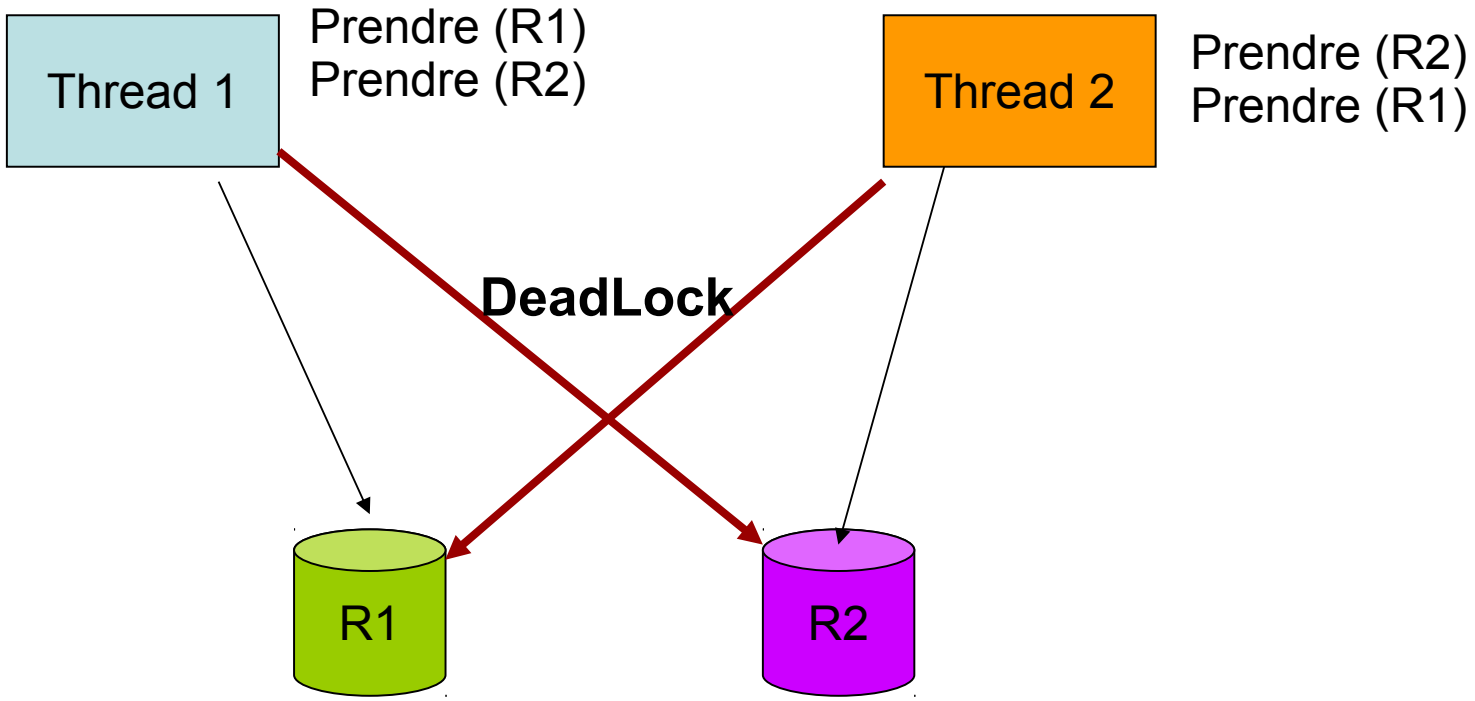
Prof. Alain MENU
Lycée La Fayette - 0770920G
IRIS
Informatique et Réseaux pour l'Industrie et les Services techniques

# Les Verrous sous Windows

- hMutex = **CreateMutex**(NULL, FALSE,« NomMutex »);
  - le verrou est créé et mis à l'état "unlock"
- **CloseHandle**(hMutex);
  - le verrou est détruit
- **WaitForSingleObject(** hMutex, INFINITE**);**
  - Si le verrou est déja pris, le thread est bloqué
- **WaitForSingleObject(** hMutex, 100**);**
  - Attends 100 ms le déblocage du verrou.
- **ReleaseMutex**(hMutex)
  - rend le verrou et le libère

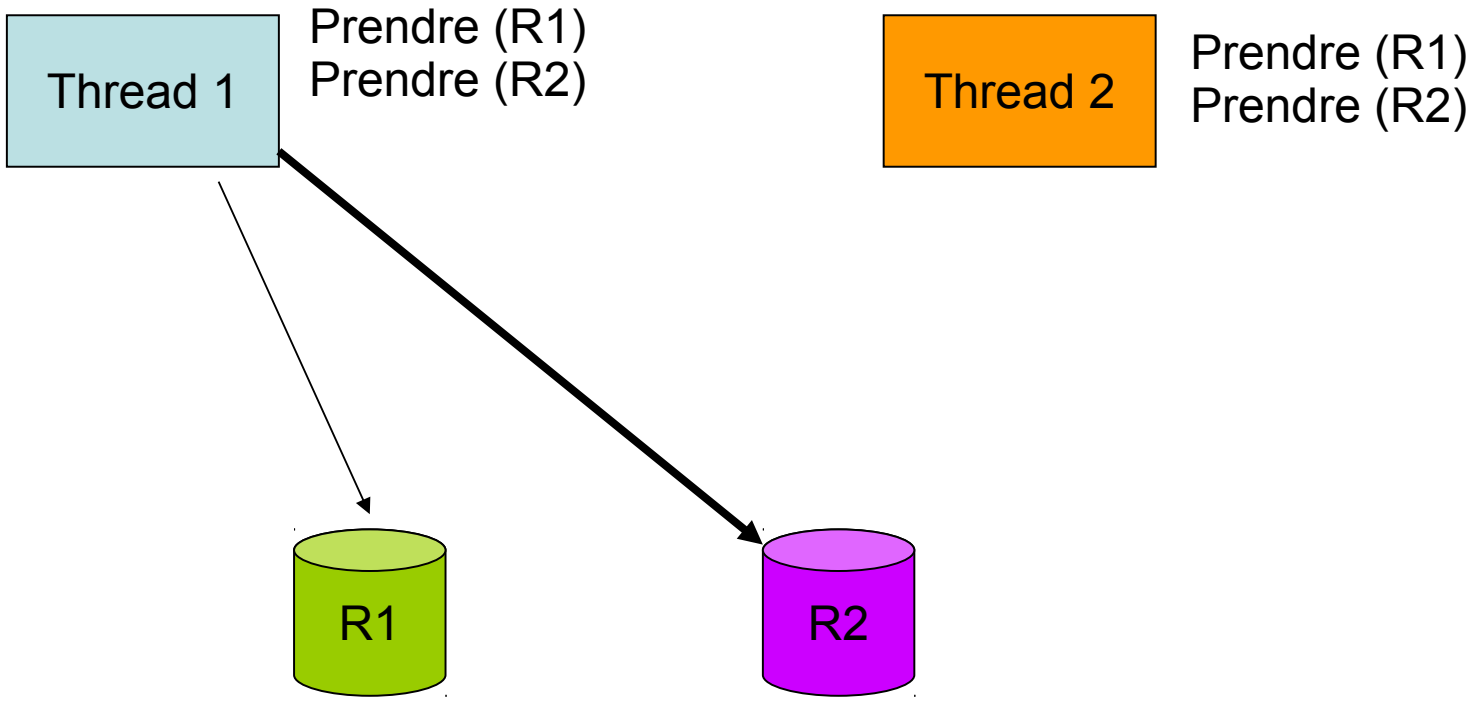
# Verrou Mortel ( DeadLock )

- Deux threads ont besoin de deux ressources



# Prévention du DeadLock

- prendre les ressources toujours dans le même ordre



# La synchronisation

- Synchronisation avec file d'attente:
  - Les sémaphores
- Un sémaphore est un outil de synchronisation complexe.
- Un sémaphore peut-être vu comme un ou plusieurs jetons.
- Utile pour limiter le nombre d'accès à une ressource.

# Exemple

- Limite de création de socket à un nombre donné.
- Un serveur TCP peut être limité à 10 sockets.

# Exemple sémaphore (1)

- Un sémaphore à deux jetons

Thread 1

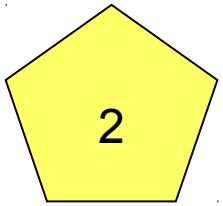
Thread 2

Thread 3

Thread 4

Thread 5

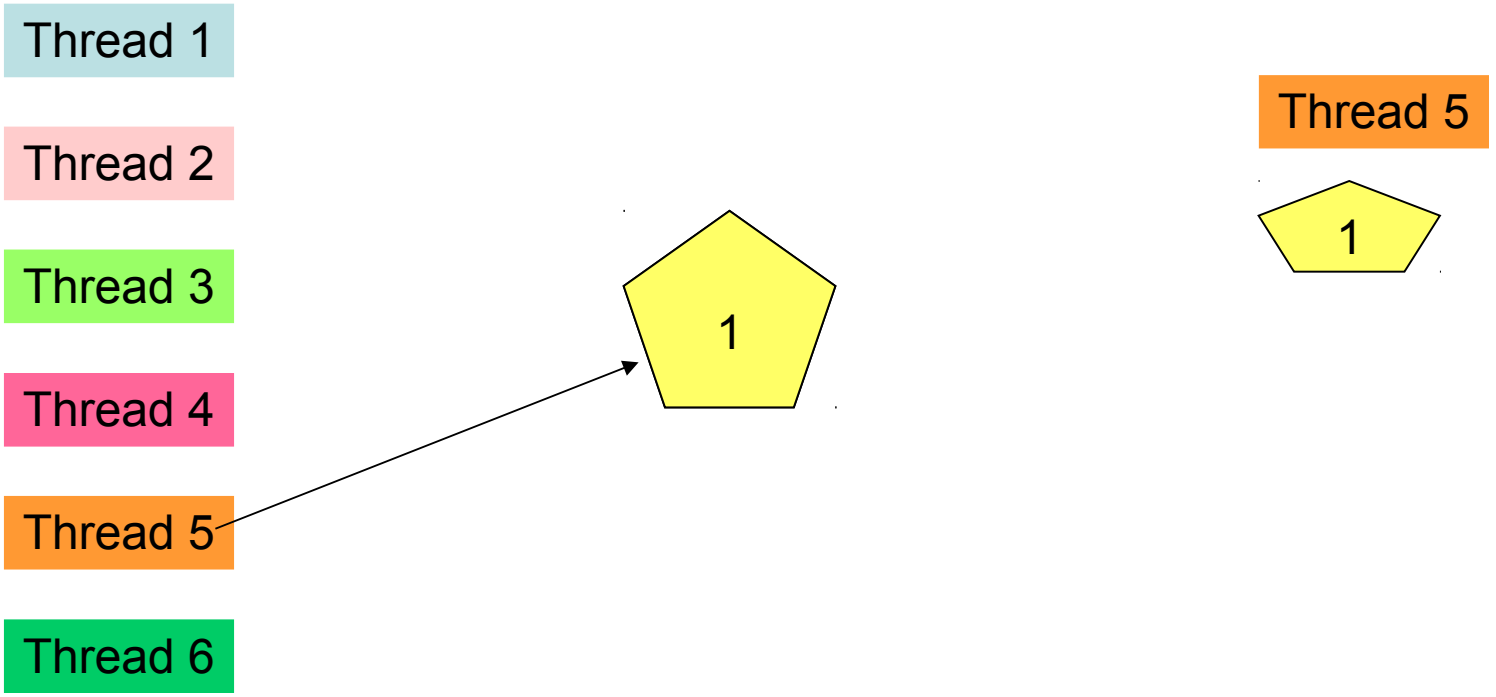
Thread 6



Prod. Alain MENU
Lycée La Fayette - 0770920G
IRIS
Informatique et Réseaux pour l'Industrie et les Services techniques

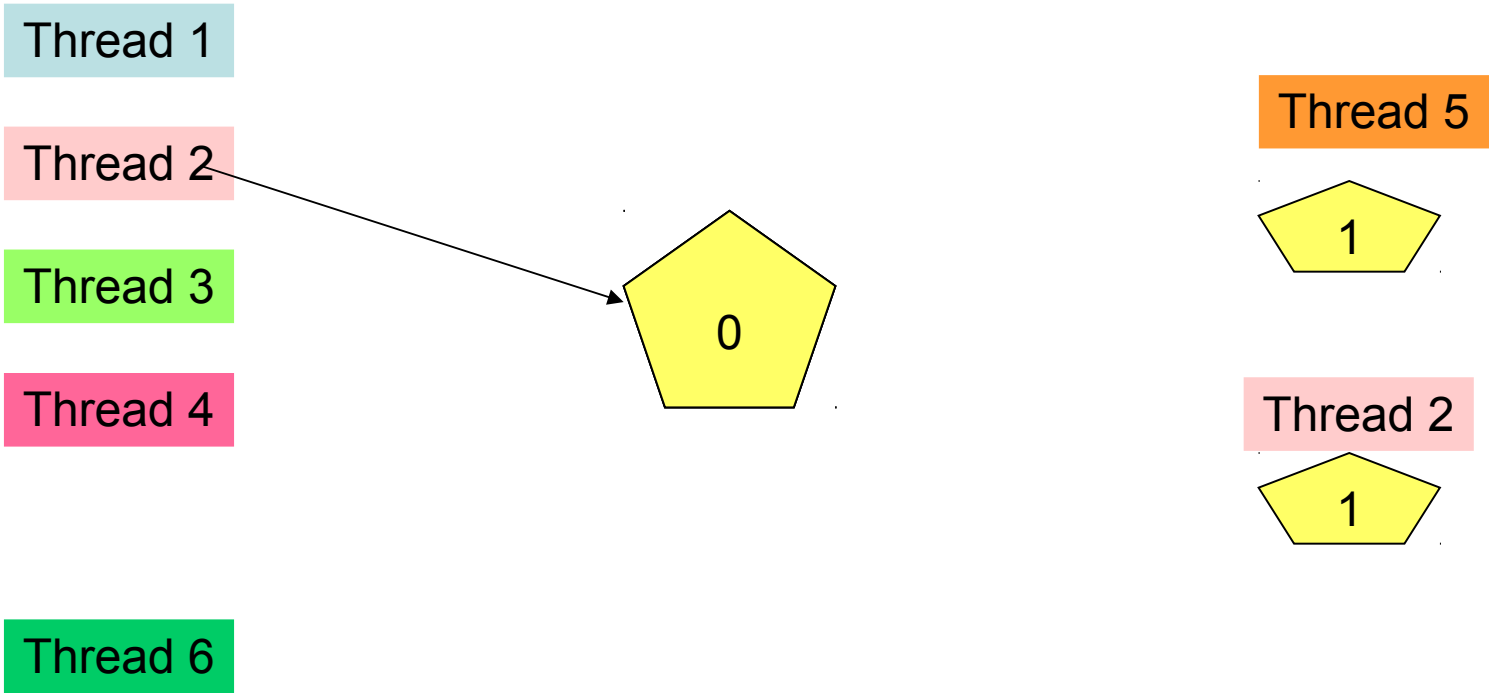
# Exemple sémaphore (2)

- Un sémaphore à deux jetons



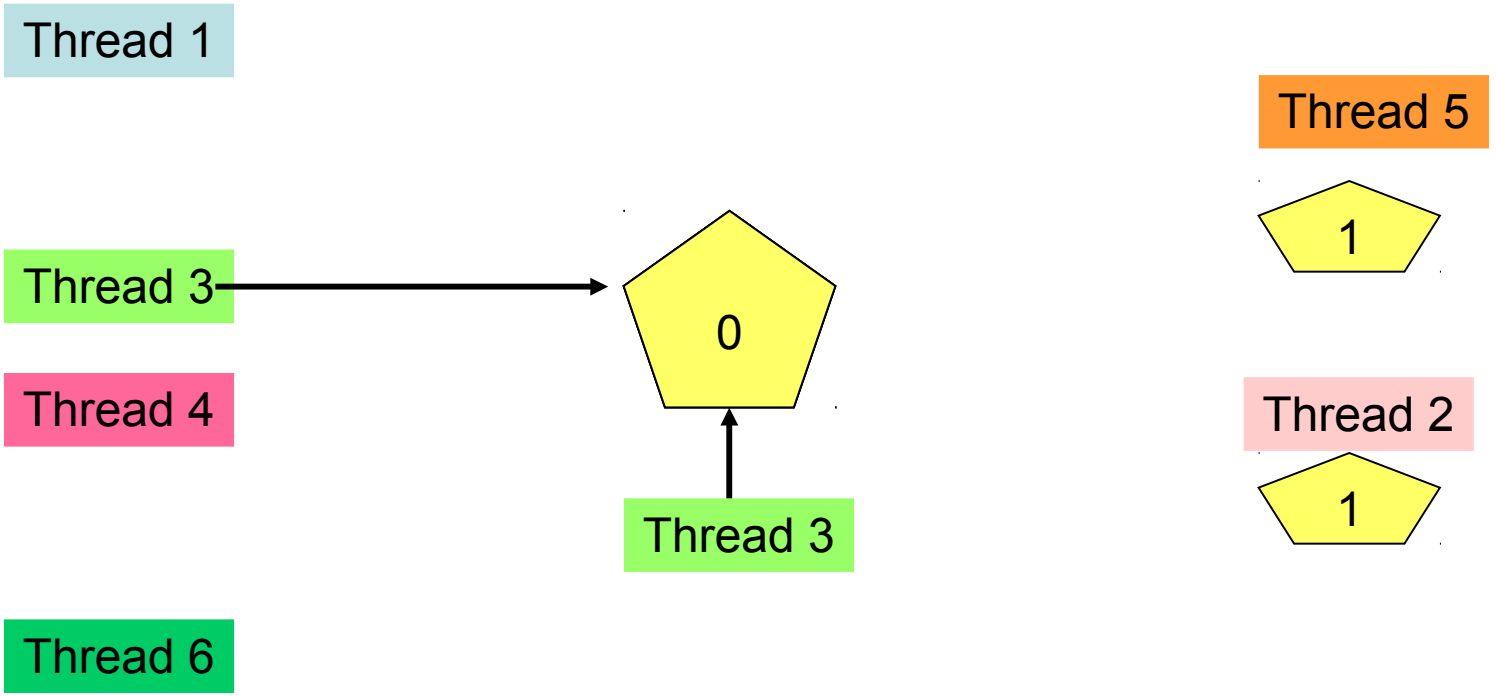
# Exemple sémaphore (3)

- Un sémaphore à deux jetons



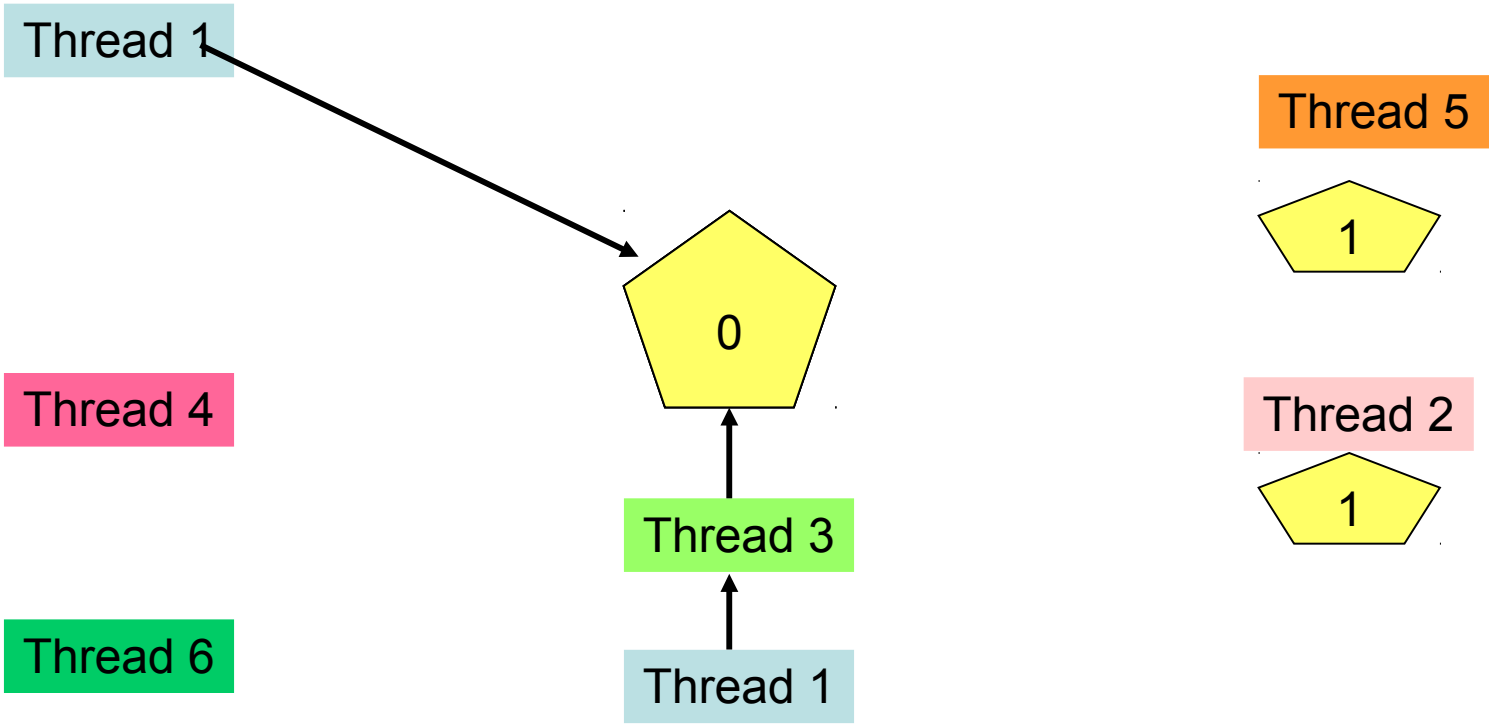
# Exemple sémaphore (4)

- Un sémaphore à deux jetons



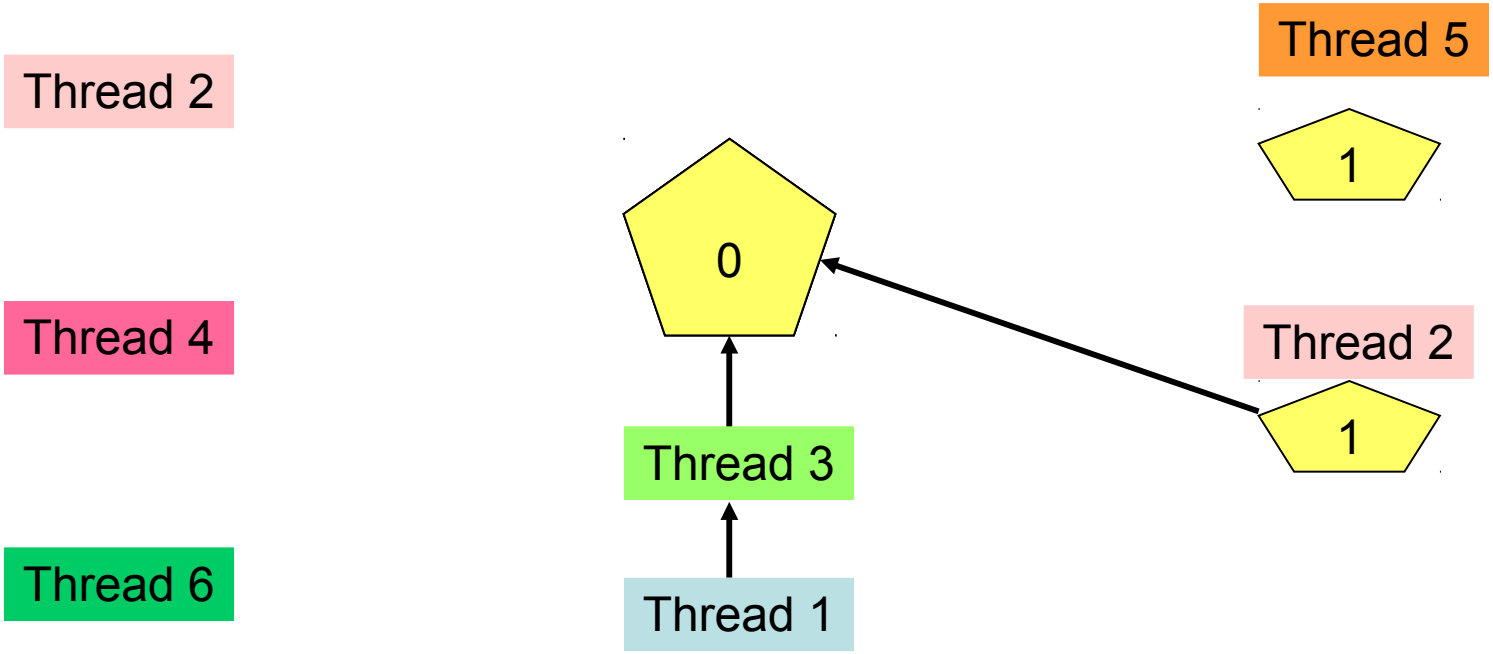
# Exemple sémaphore (5)

- Un sémaphore à deux jetons



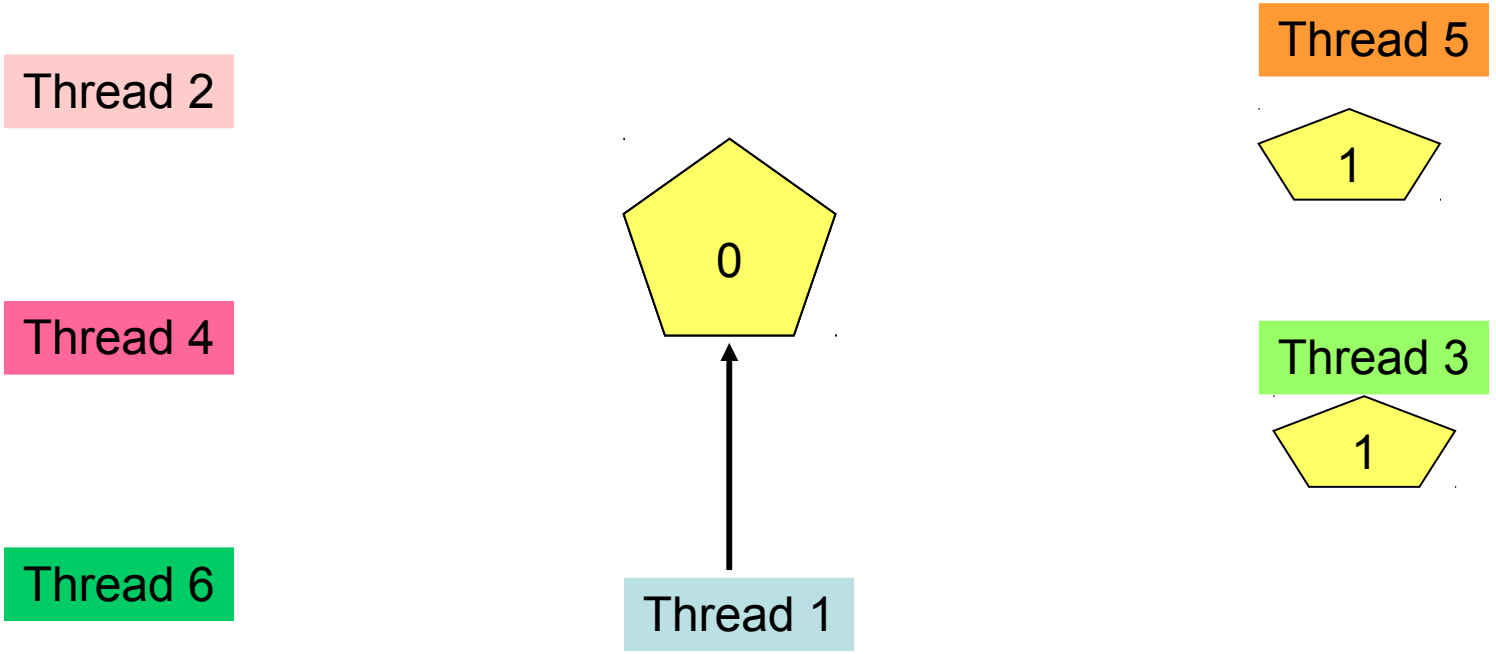
# Exemple sémaphore (6)

- Un sémaphore à deux jetons



# Exemple sémaphore (7)

- Un sémaphore à deux jetons



# Les sémaphores sous Linux

- sem_init (&S, , Compt,)
  - initialisation du sémaphore
- sem_destroy (&S)
  - suppression du sémaphore
- sem_wait (&S)
  - attend que la valeur du compteur du sémaphore soit positive, puis la décrémente
- sem_trywait (&S)
  - décrémente le compteur s'il est positif, sinon erreur
- sem_post (&S)
  - incrémente le compteur et libère éventuellement un thread.

# Les Sémaphores sous Windows

- hSem = **CreateSémaphore**(…);
  - Créé le sémaphore
- **CloseHandle**(hSem);
  - ferme le sémaphore.
- **ReleaseSemaphore**(hSem)
  - rend le sémaphore et le libère
- **WaitForSingleObject(** hSem, INFINITE**);**
  - Si le sémaphore est déjà pris, le thread est bloqué
- **WaitForSingleObject(** hSem, 100**);**
  - Attends 100 ms le déblocage du sémaphore.

# Conclusion

- **Le thread:**
  - il est moins coûteux en temps cpu qu'un processus, il permet un partage des données plus facile.
  - La principale difficulté dans l'utilisation des threads est la gestion des accès aux ressources communes.

# Conclusion

- **le verrou (*lock*)**
  - l'outil le plus rapide et le moins consommateur de mémoire. Il doit être rendu par le thread qui l'a pris. Il s'utilise surtout pour sérialiser l'accès à une ressource ou assurer la cohérence des données.
- **le sémaphore**
  - consomme plus de mémoire et s'utilise dans les cas où on a besoin d'une file d'attente et d'un nombre d'accès à une ressource supérieur à 1.
- pour éviter les inter-blocages (dead lock), utiliser les verrous ou les sémaphores suivant un ordre fixe.

Prod. Alain MENU
Lycée La Fayette - 0770920G
iRIS
Informatique et Réseaux pour l'Industrie et les Services techniques

# Questions ?

- ?