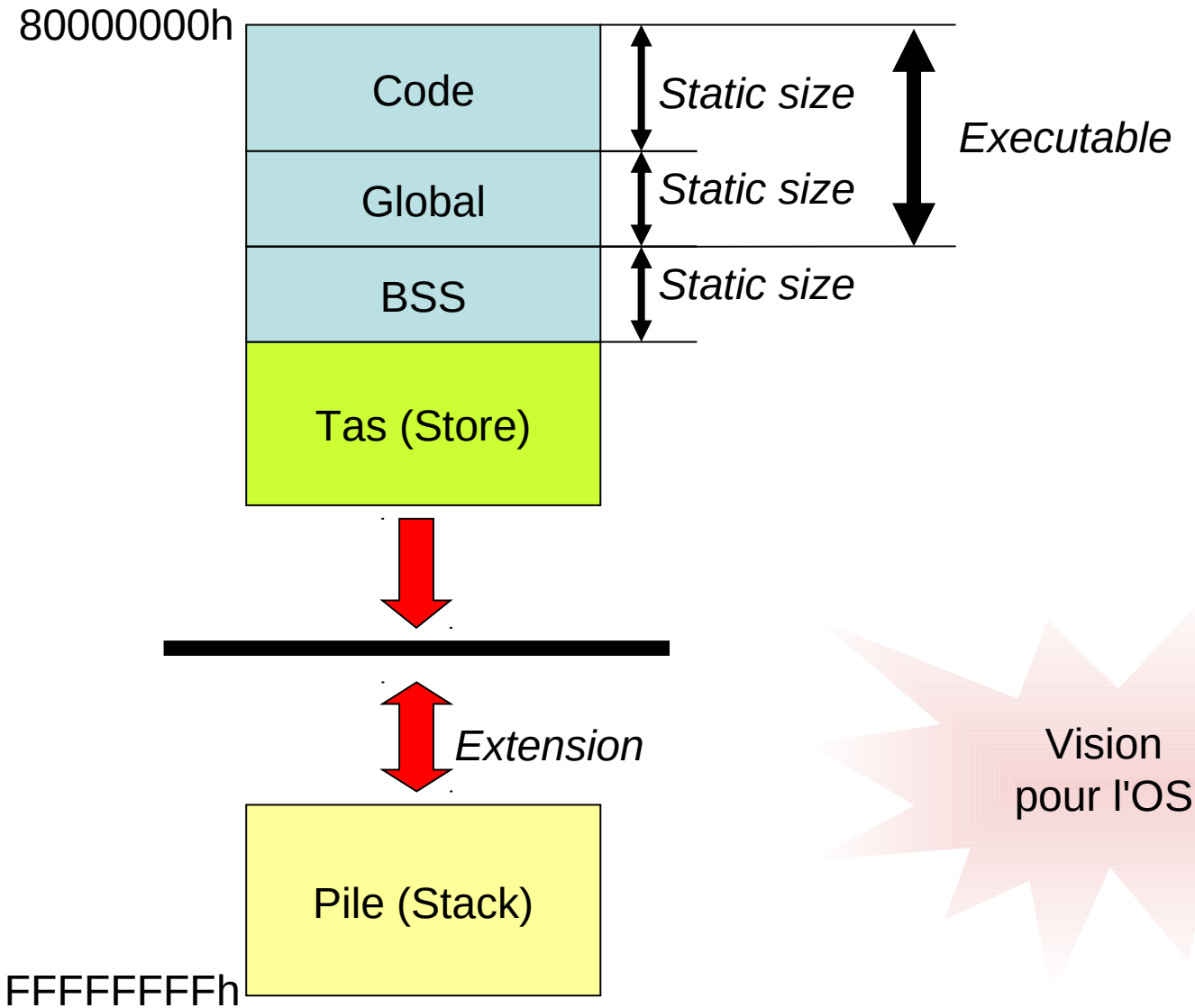
Ramasse-miettes

Benoît Sonntag

---

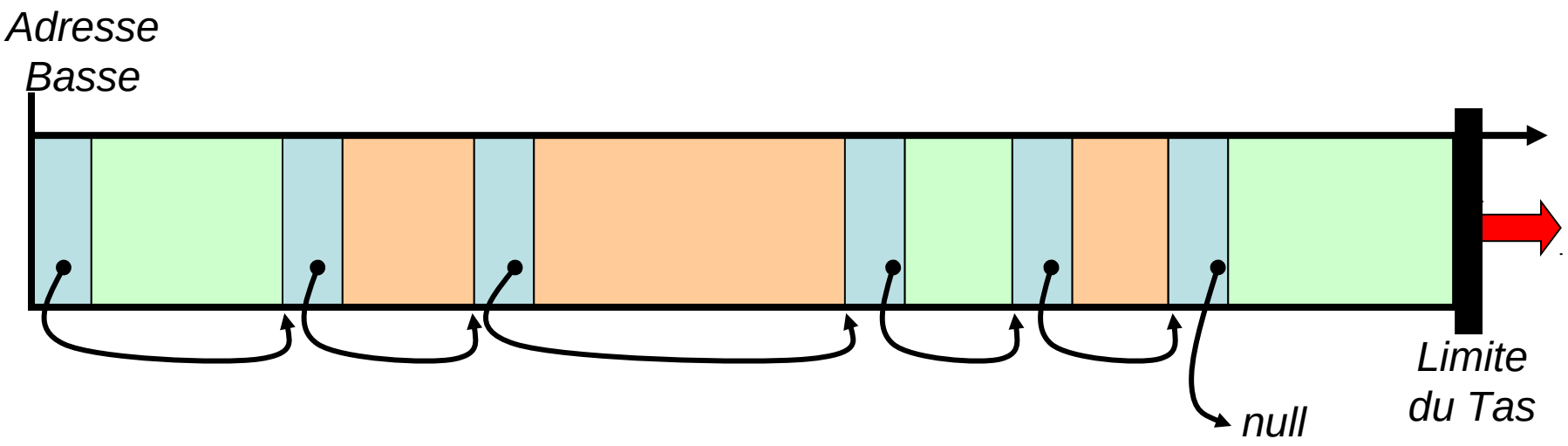
sonntag@icps.u-strasbg.fr
sonntag@IsaacOS.com
benoit@sonntag.fr

La mémoire de Tas (Store)



Vision
pour l'OS

Malloc / Free



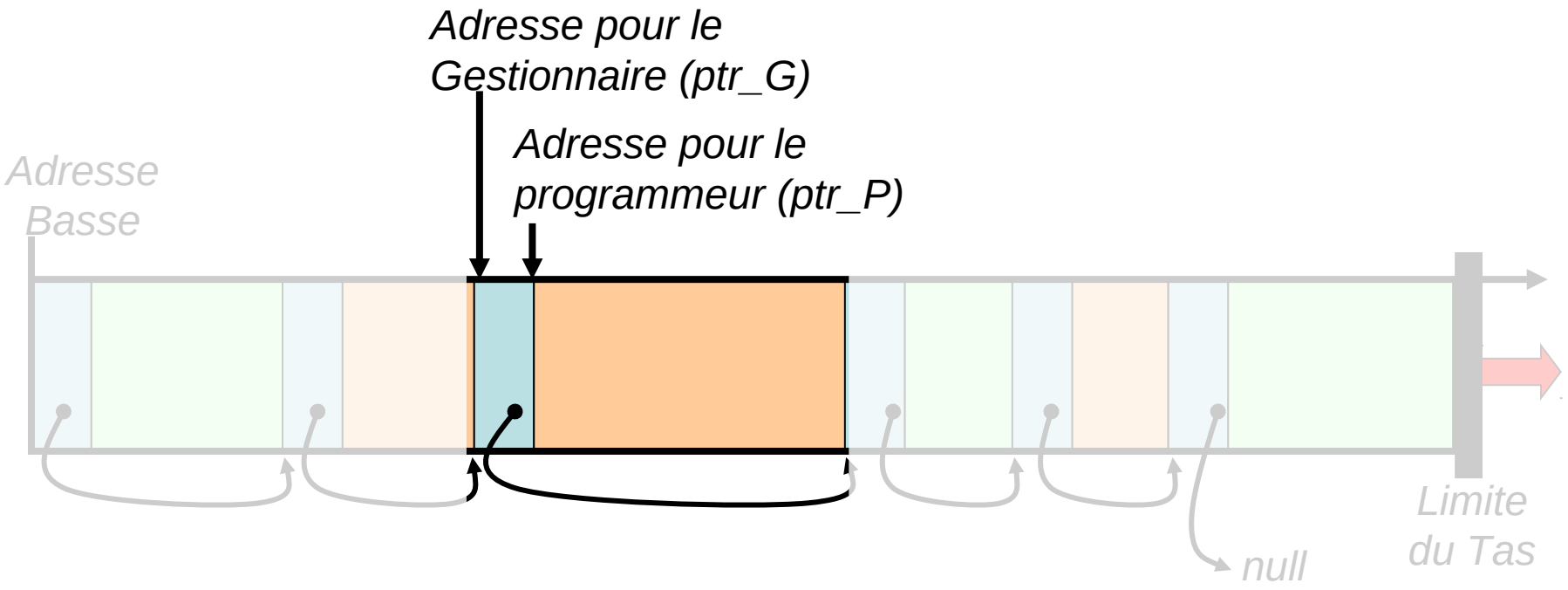
bloc libre

bloc occupé

Entête

Utilisé pour la gestion de la liste chaînée des blocs

Malloc / Free




```
ptr_P = malloc(30);          // ptr_P = ptr_G + constante
free(ptr_P);                 // ptr_G = ptr_P - constante
```

Malloc / Free

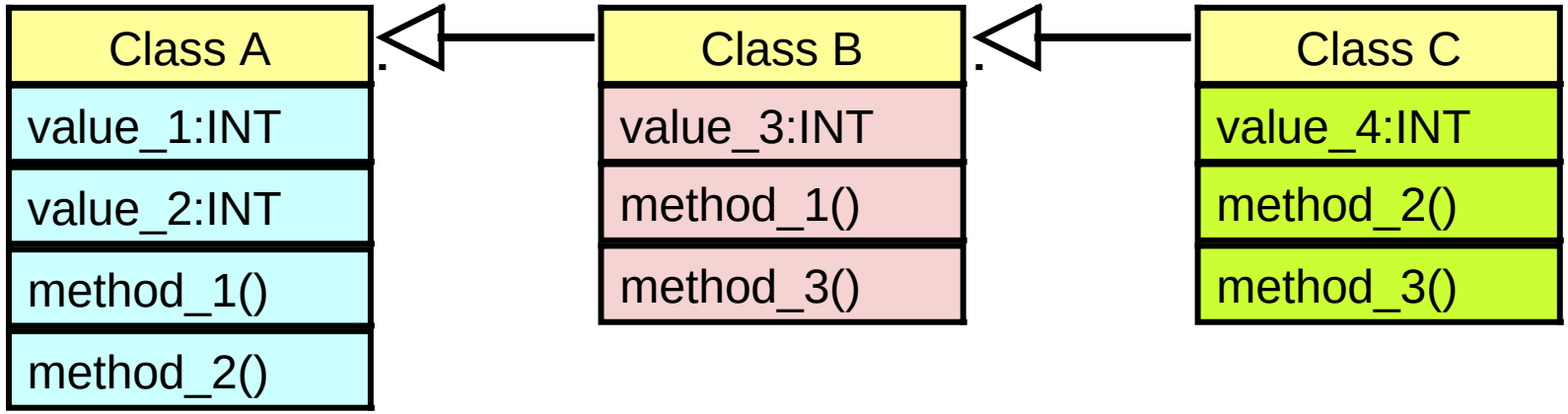
Avantage:

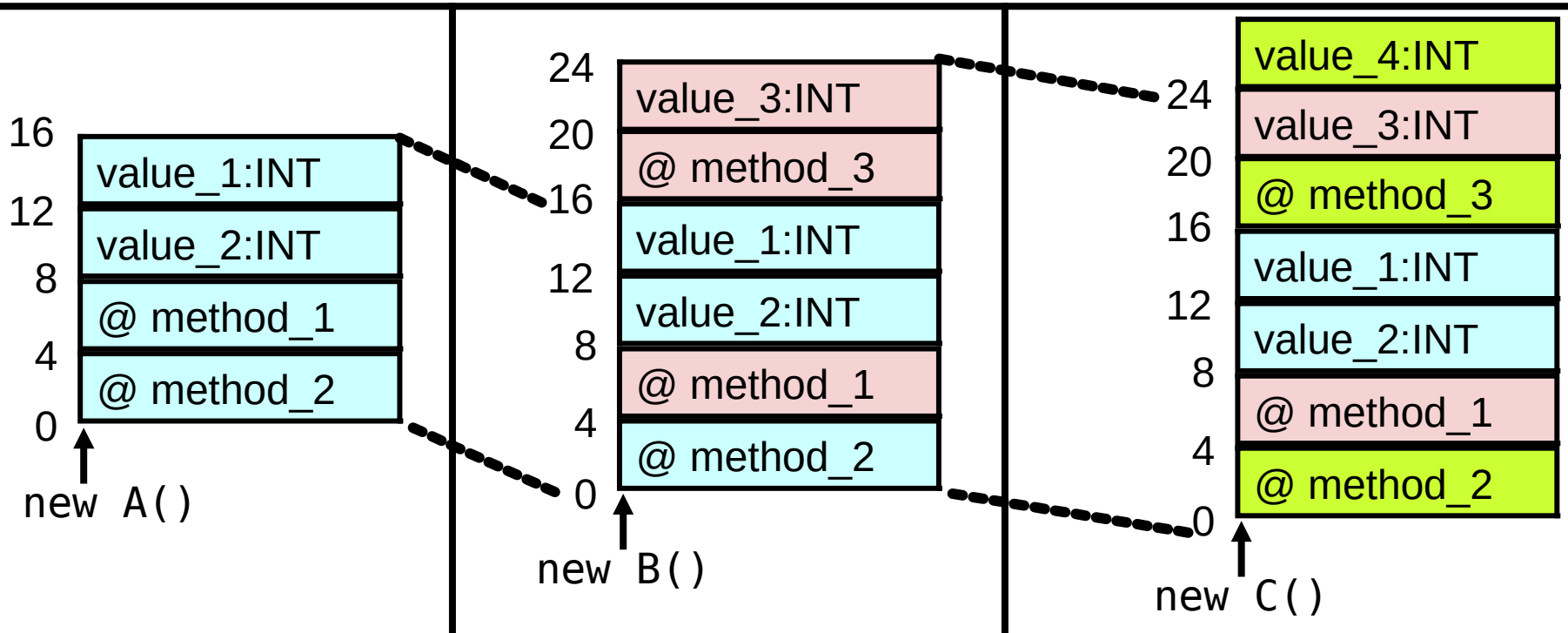
- Allocation de taille flexible

Désavantages:

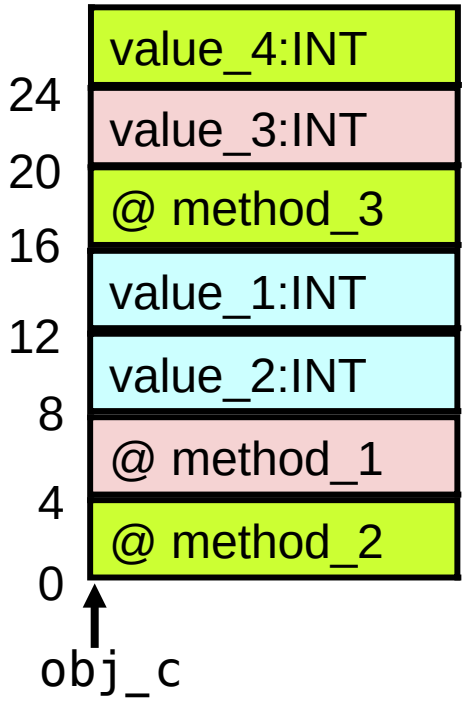
- Allocation non adapté à la programmation objet
- Allocation lente ! *(parcours d'une liste chaînée)*
- Gestion manuelle de la mémoire
- Gâchis mémoire : Entête à chaque bloc
- Fragmentation interne !
- Pas de sécurité

Un objet en mémoire (VFT)



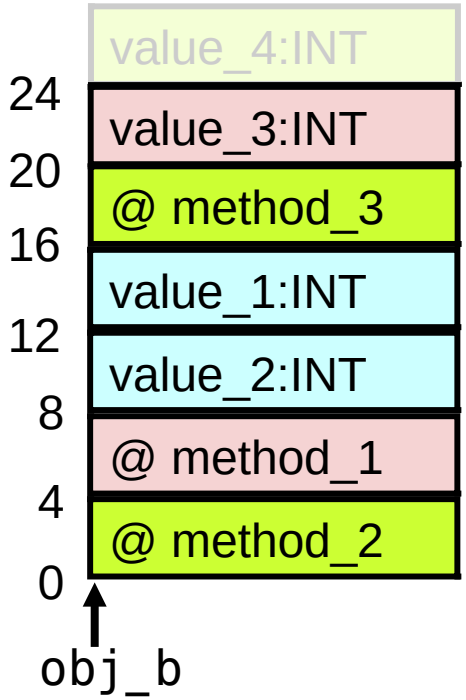


Un objet en mémoire (VFT)



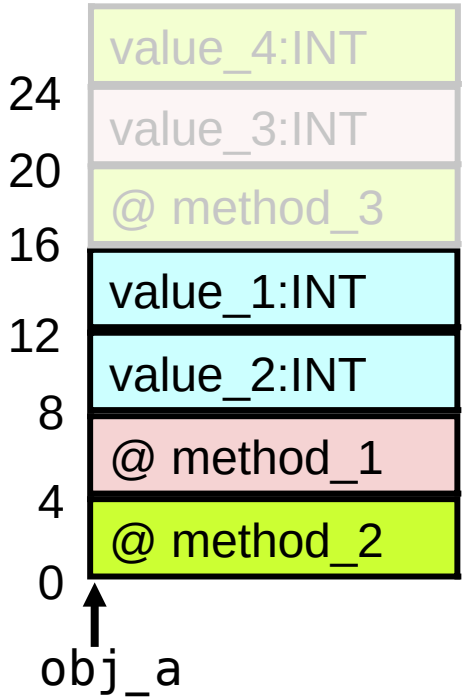

```
obj_a:A;
obj_b:B;
obj_c:C;
obj_c = new C();
```

Un objet en mémoire (VFT)



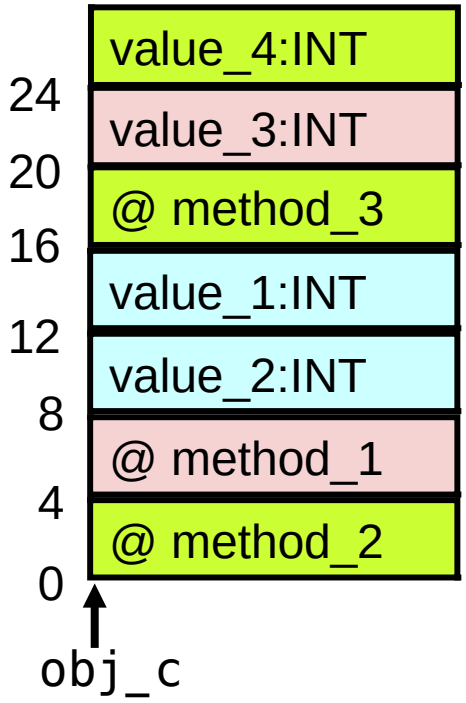
obj_b = obj_c;

Un objet en mémoire (VFT)



obj_a = obj_c;

Un objet en mémoire (VFT)



obj_c = (C)obj_a;

Ramasse-miettes

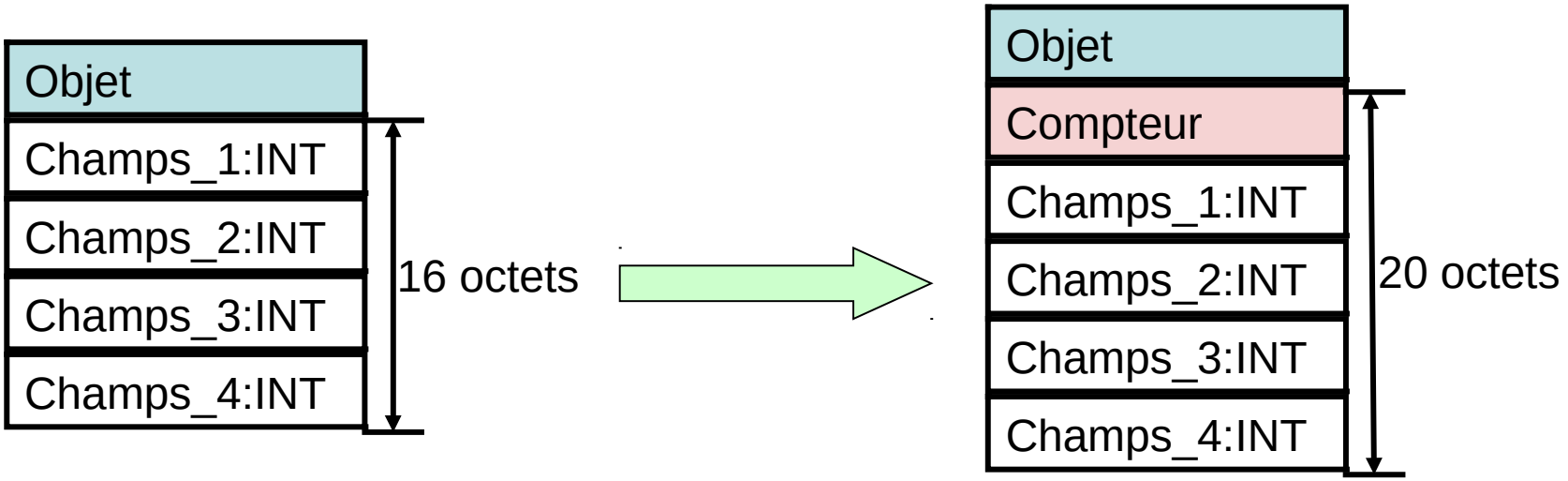
Gestion automatique de la libération de la mémoire *(Garbage Collector)*

Les 3 algorithmes actuellement existant:

1. Le comptage de références *(reference counting)*
2. Le marquage / Balayage *(mark-and-sweep)*
3. La recopie *(copying)*

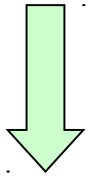
Le comptage de références

Le plus ancien (1960)



Le comptage de références

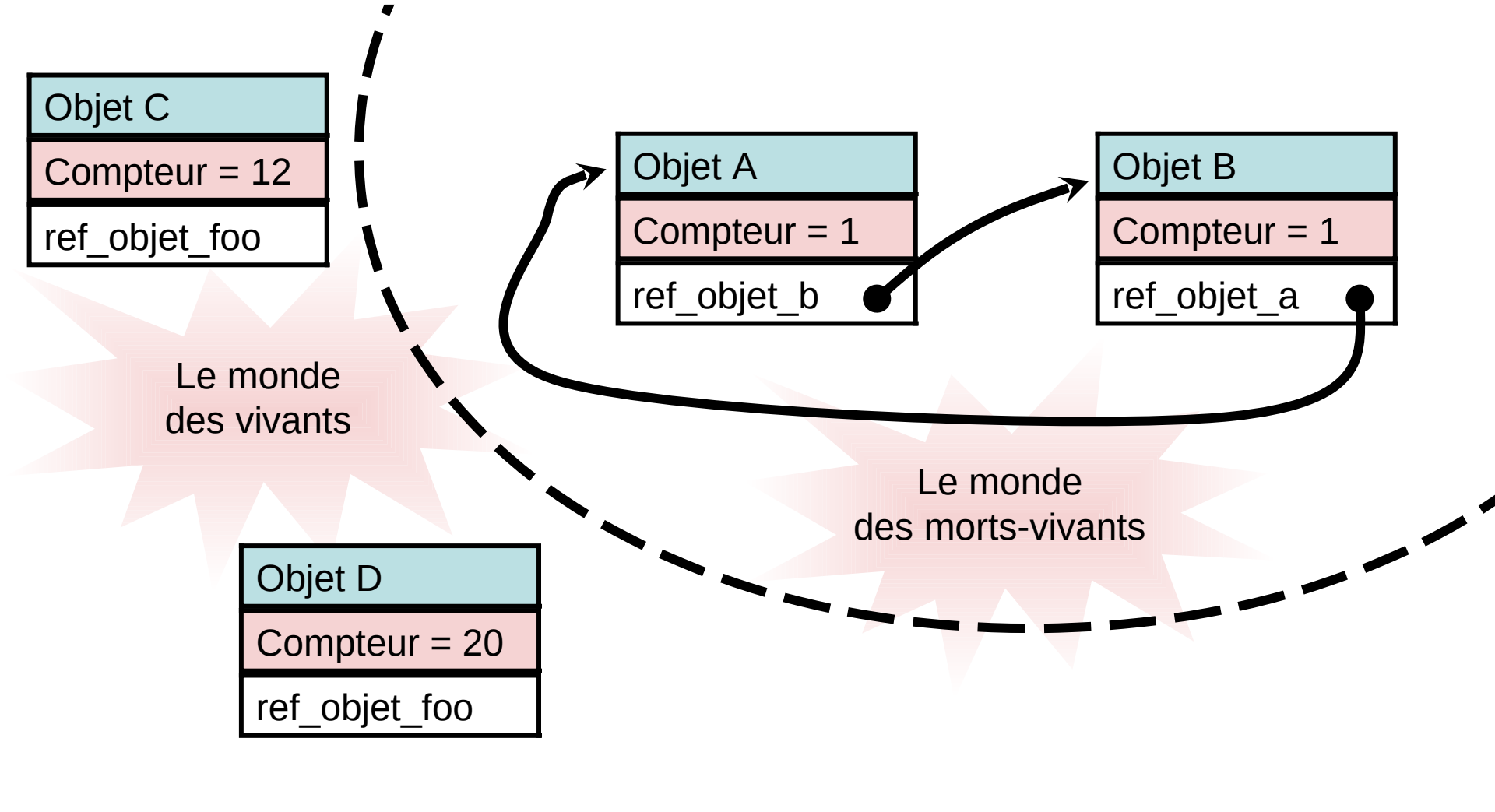
```
a := b
```



```
si (a != NULL) alors
  a.compteur := a.compteur – 1
  si (a.compteur = 0) alors
    libere(a)
  fsi
fsi
si (b != NULL) alors
  b.compteur := b.compteur + 1
fsi
a := b
```

Le comptage de références

<u>Problème</u> : Les références cycliques !



Le comptage de références

Avantages:

- Facile à mettre en place
- Adapté à la programmation temps-réelle

Désavantages:

- Problème des références cycliques
- Très **très très** lents !!!!
- Un compteur en plus pour chaque objet

N'est plus utilisé actuellement…

Marquage-balayage

Mc Carthy (1960)

Étapes :

1. Stopper l'application en cours
2. Marquer toutes les objets "vivants"
3. Libérer les objets "non marqué"

Marquage-balayage

1. Stopper l'application en cours

Périodiquement
(ce n'est pas une bonne solution, sauf pour le temps réel)

OU

Lorsque nous n'avons plus de mémoire disponible
(gâchis)

OU

Selon un seuil variable dépendant de la fréquence des allocations
(bonne solution)

Marquage-balayage

2. Marquer toutes les objets "vivants"
   2.1 Recherche des objets obligatoirement vivants

(Base de la recherche)

Les objets en mémoire global ou bss

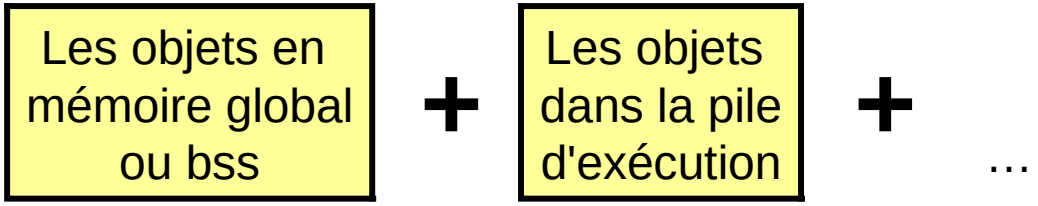
+

...

Marquage-balayage

2. Marquer toutes les objets "vivants"
   2.1 Recherche des objets obligatoirement vivants

(Base de la recherche)

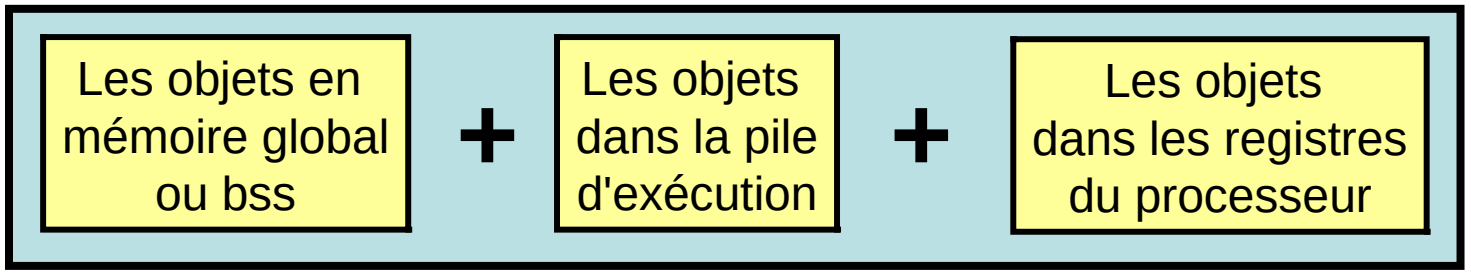


Marquage-balayage

2. Marquer toutes les objets "vivants"

2.1 Recherche des objets obligatoirement vivants

(Base de la recherche)

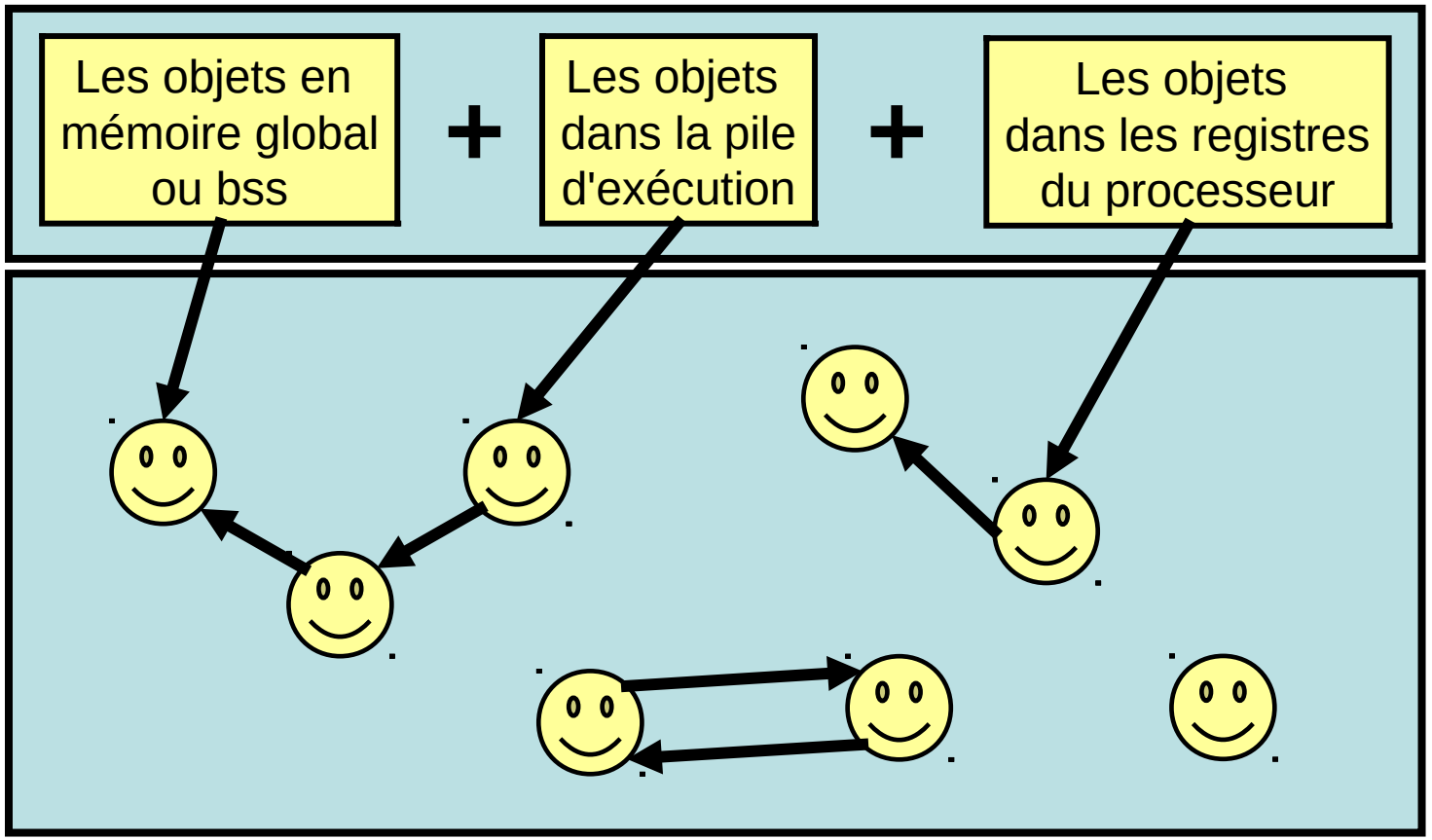


Base des objets vivants

Marquage-balayage

2. Marquer toutes les objets "vivants"
   2.2 Fermeture transitive des objets vivants

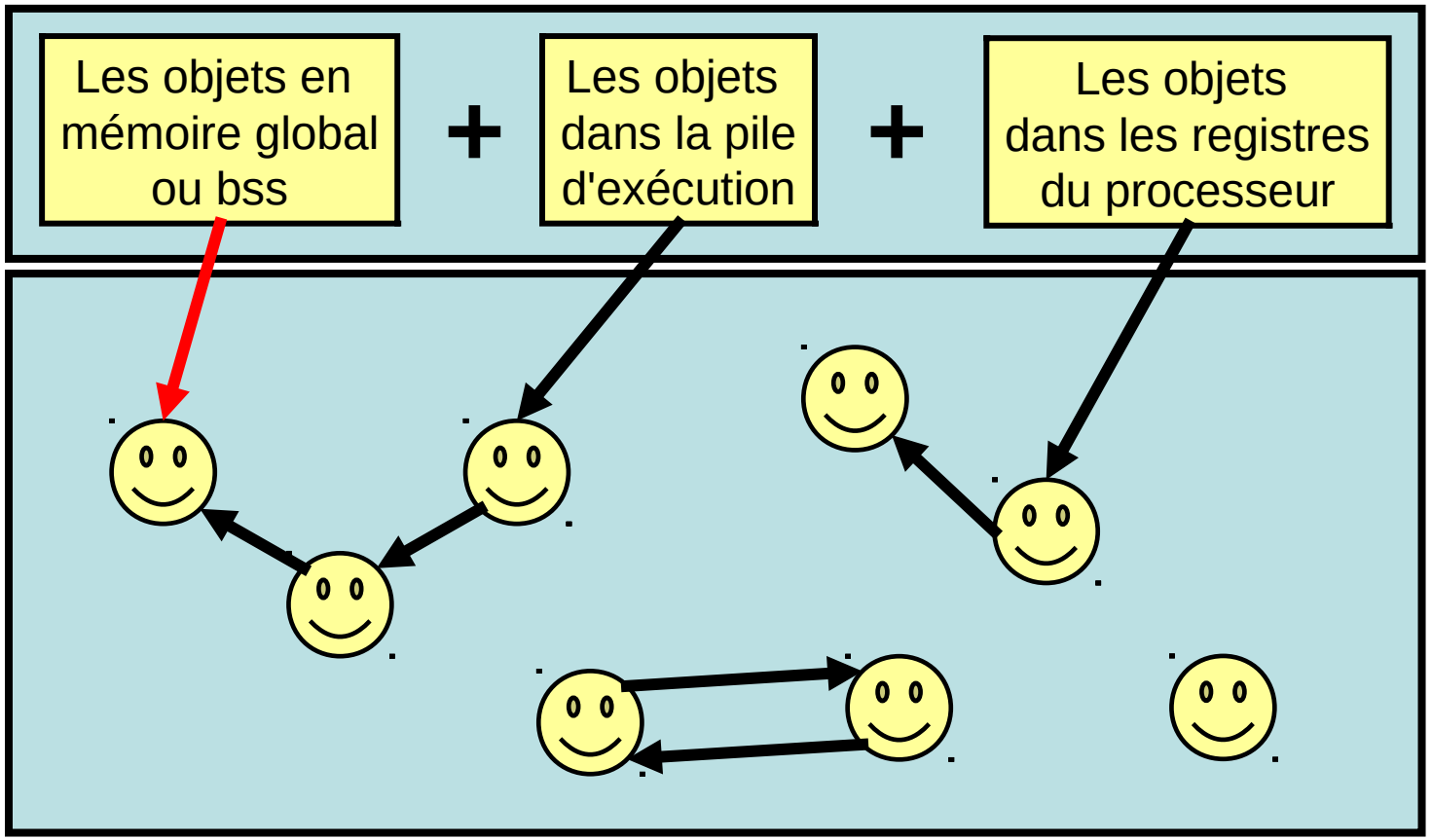
(recherche récursive partant de la base)



Marquage-balayage

2. Marquer toutes les objets "vivants"
   2.2 Fermeture transitive des objets vivants

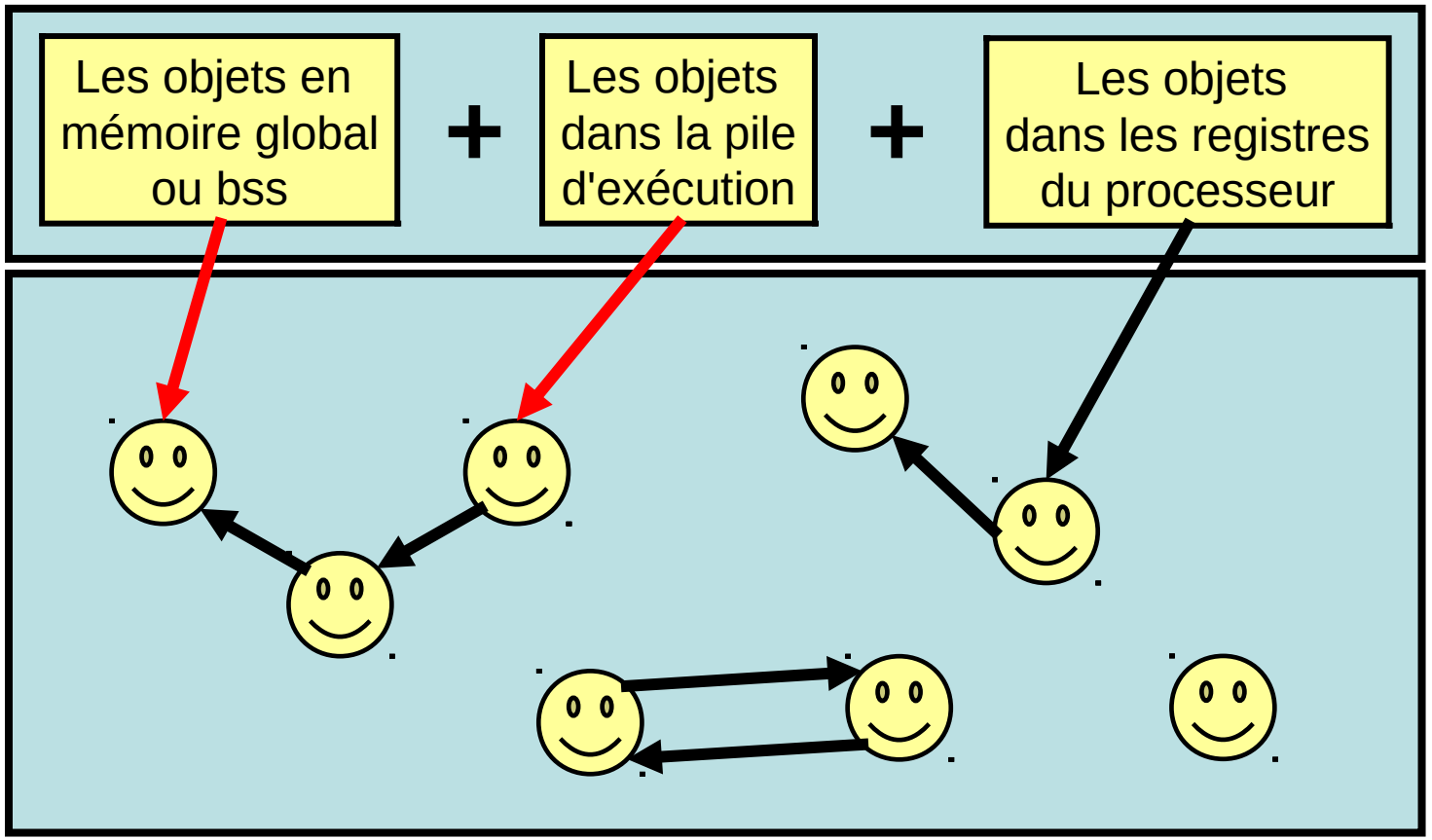
(recherche récursive partant de la base)



Marquage-balayage

2. Marquer toutes les objets "vivants"
   2.2 Fermeture transitive des objets vivants

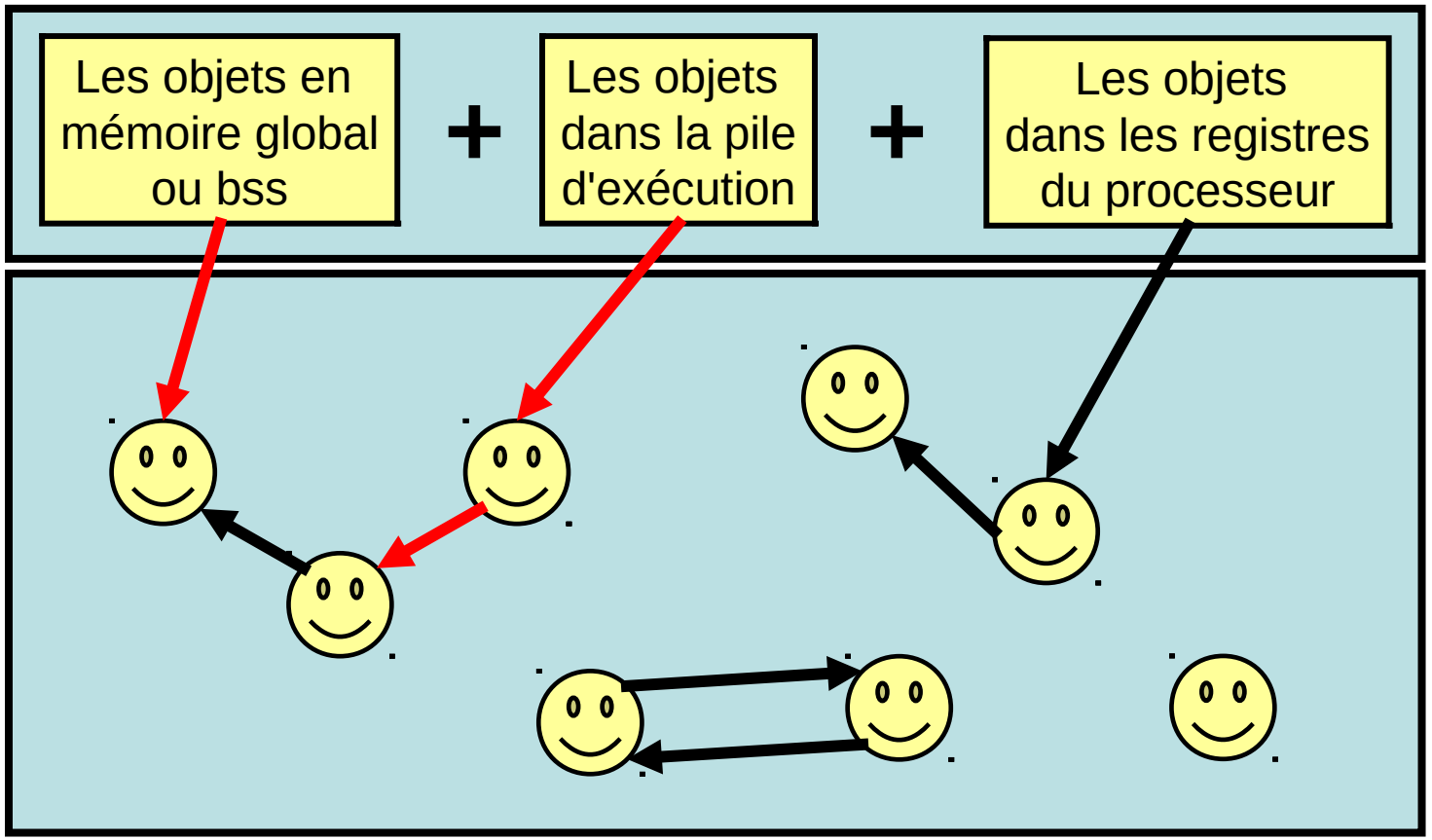
(recherche récursive partant de la base)



Marquage-balayage

2. Marquer toutes les objets "vivants"
   2.2 Fermeture transitive des objets vivants

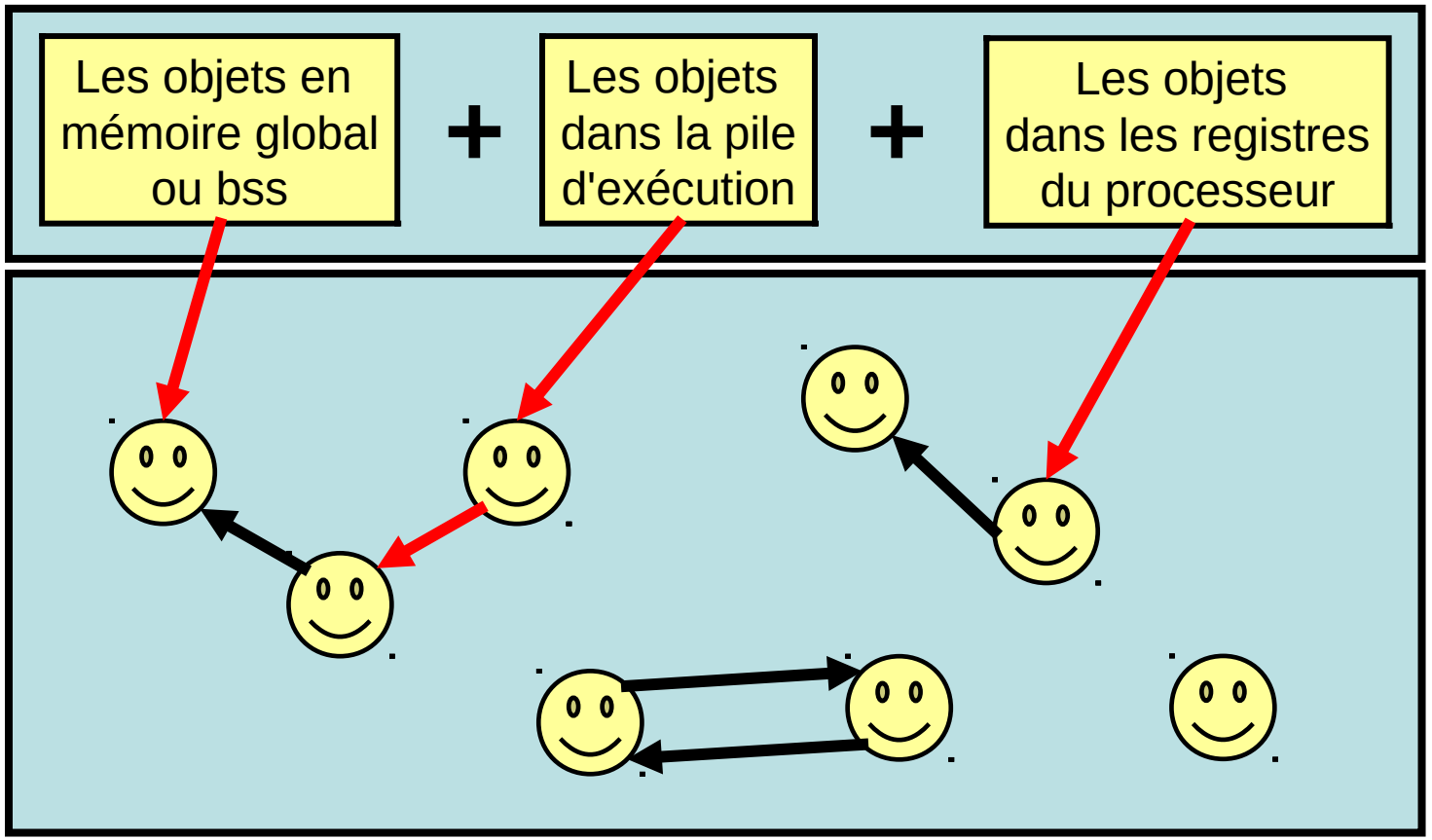
(recherche récursive partant de la base)



Marquage-balayage

2. Marquer toutes les objets "vivants"
   2.2 Fermeture transitive des objets vivants

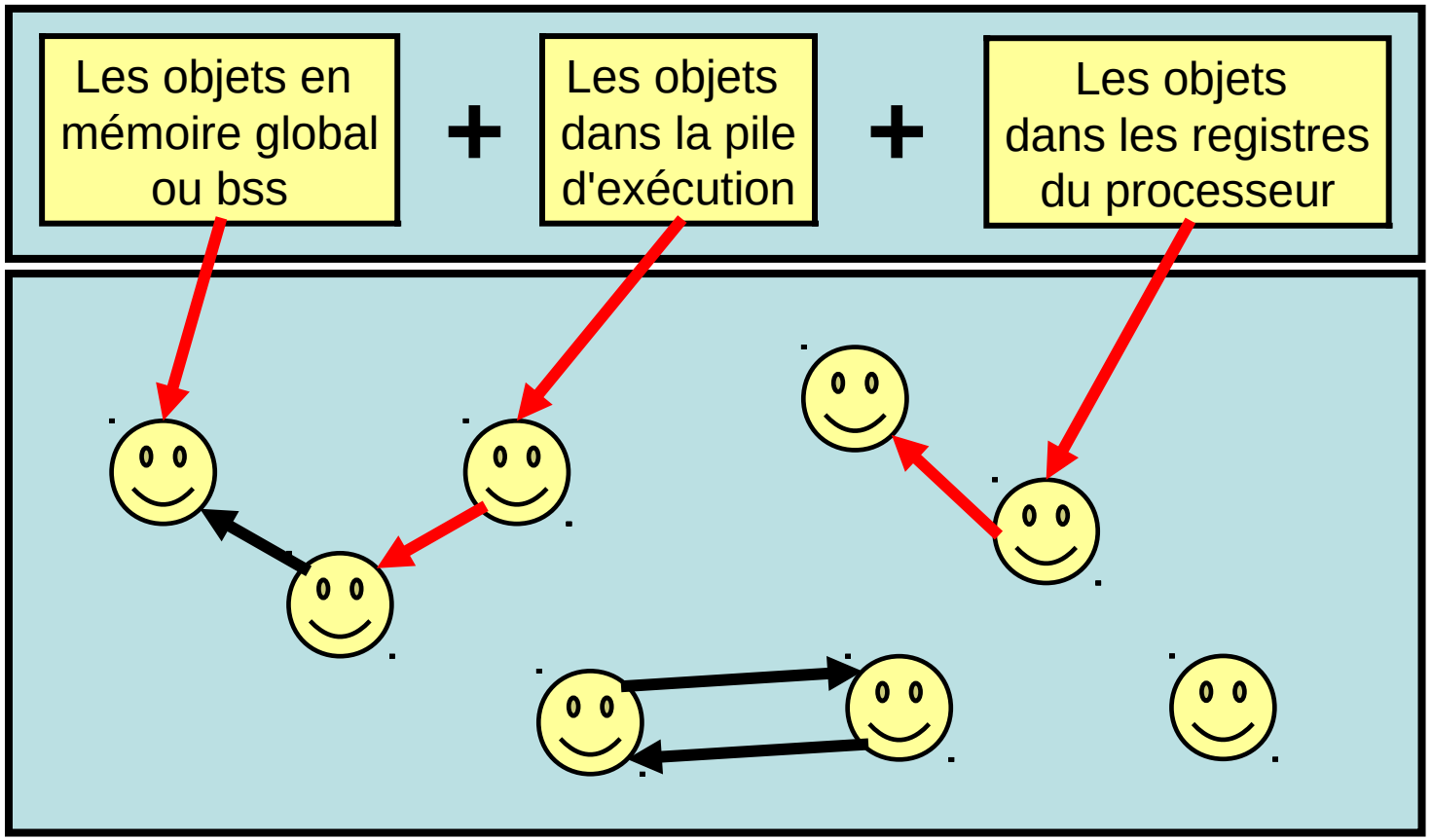
(recherche récursive partant de la base)



Marquage-balayage

2. Marquer toutes les objets "vivants"
   2.2 Fermeture transitive des objets vivants

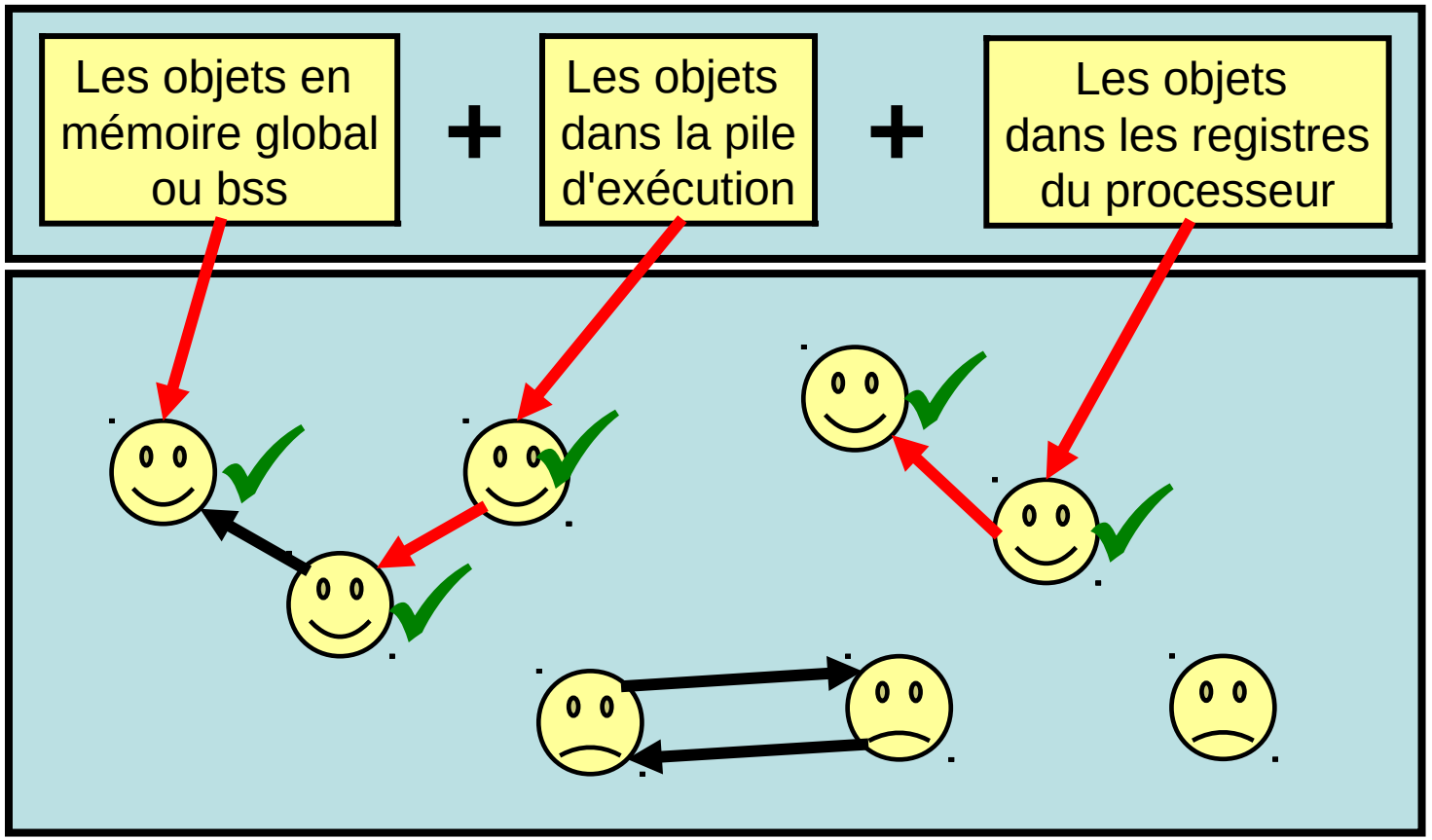
(recherche récursive partant de la base)



Marquage-balayage

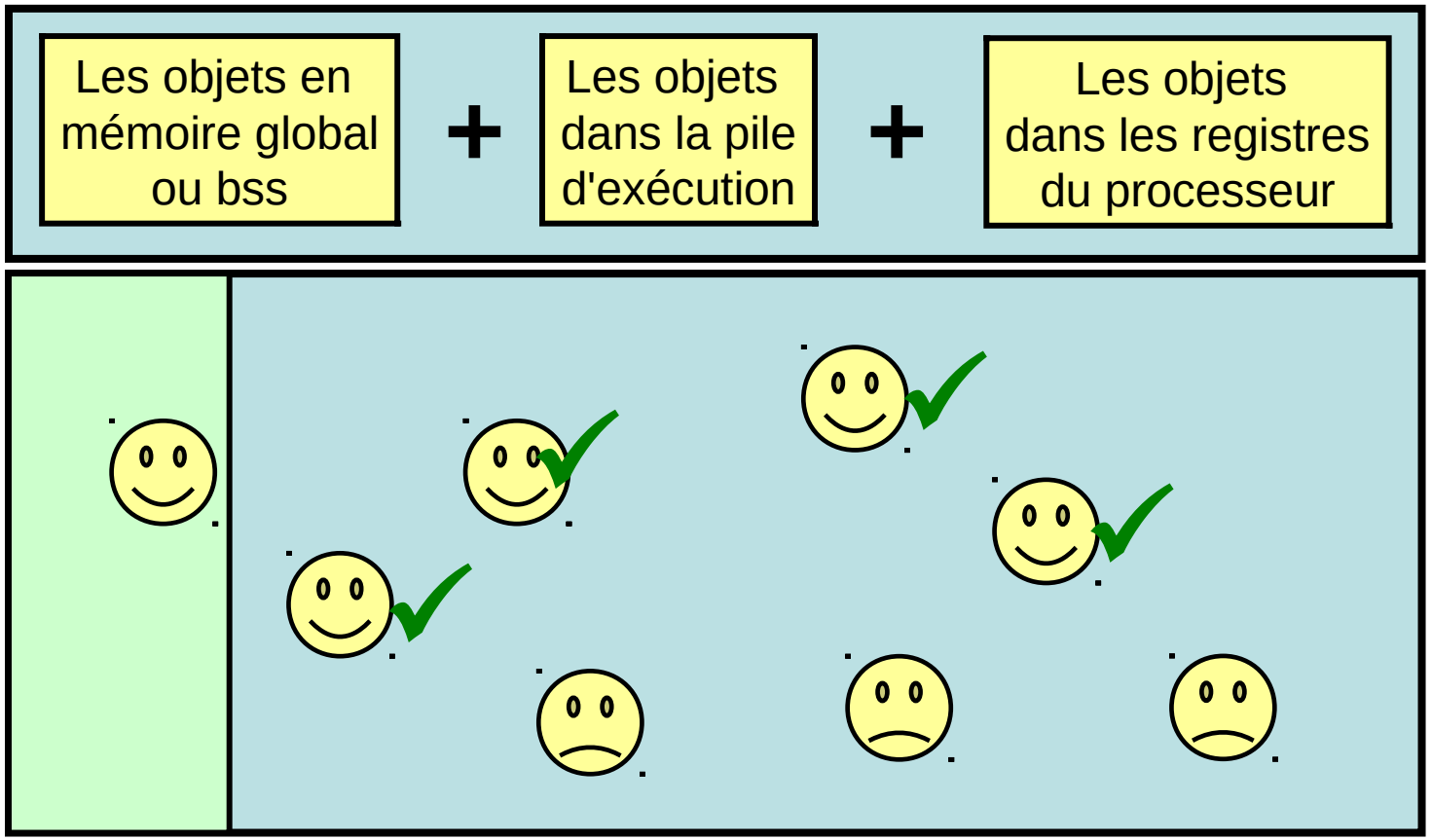
2. Marquer toutes les objets "vivants"
   2.2 Fermeture transitive des objets vivants

(recherche récursive partant de la base)



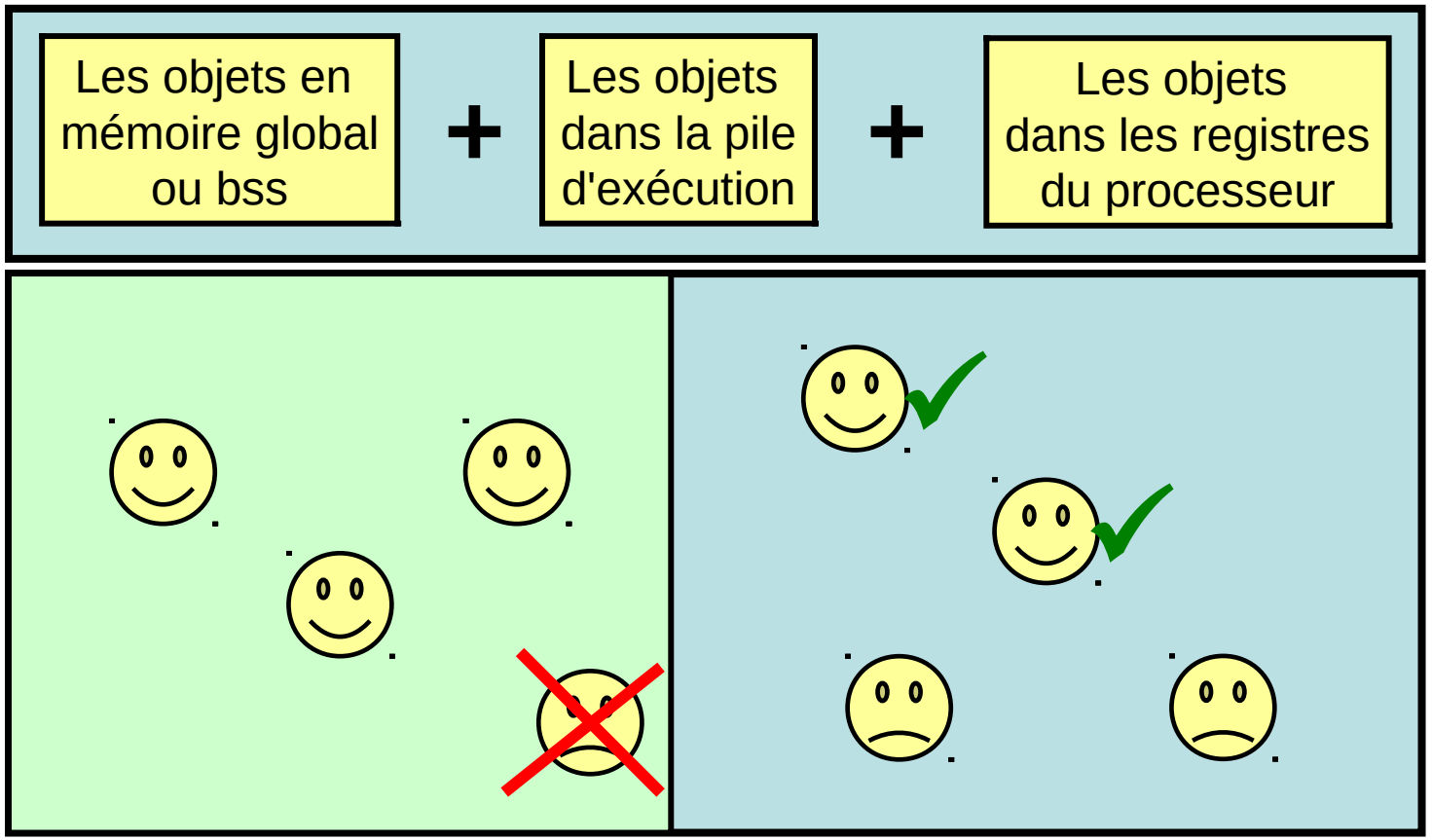
Marquage-balayage

3. Libérer les objets "non marqué"
(Parcours linéaire du tas)



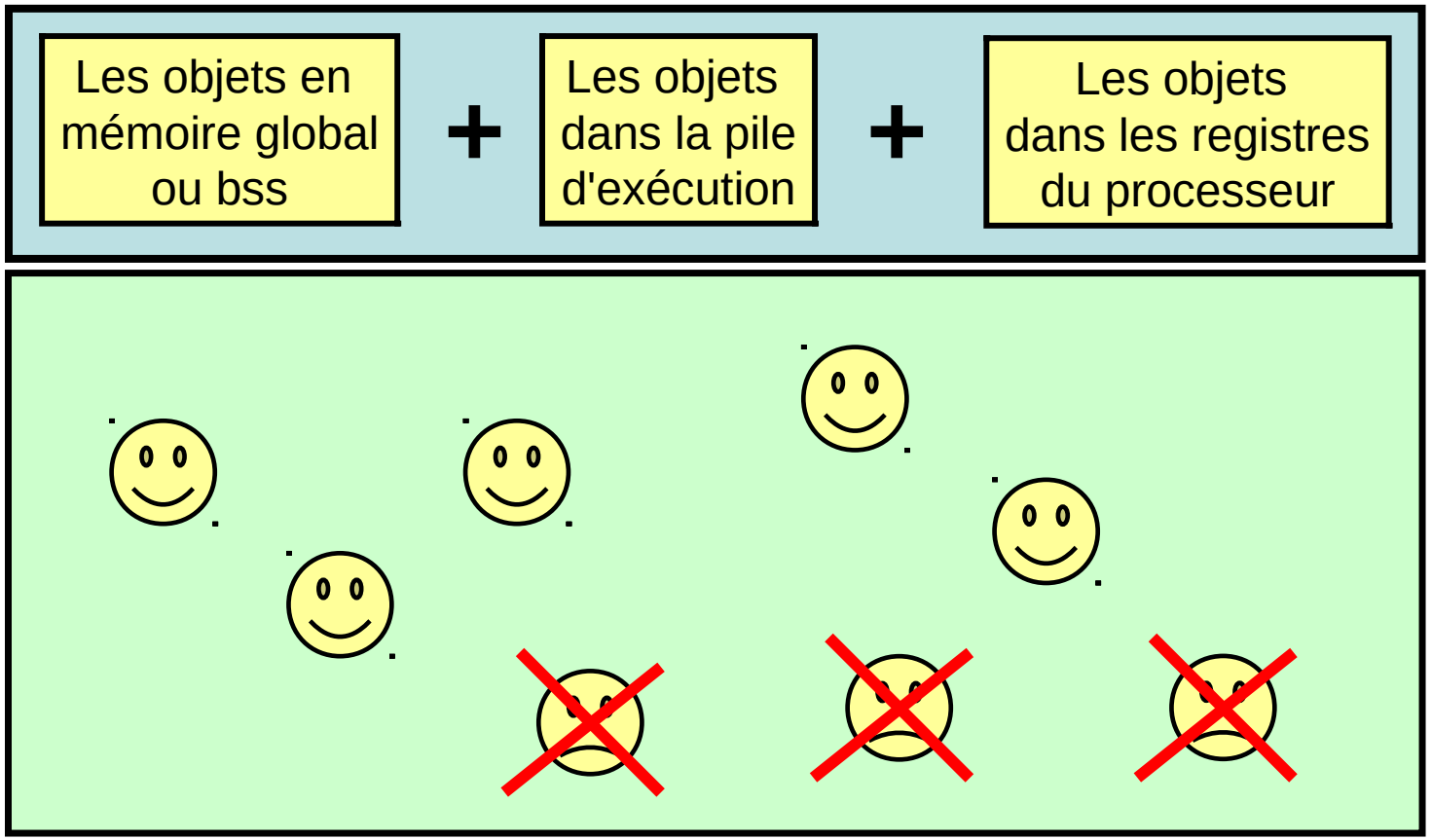
Marquage-balayage

3. Libérer les objets "non marqué"
(Parcours linéaire du tas)



Marquage-balayage

3. Libérer les objets "non marqué"
(Parcours linéaire du tas)



Marquage-balayage

Avantages:

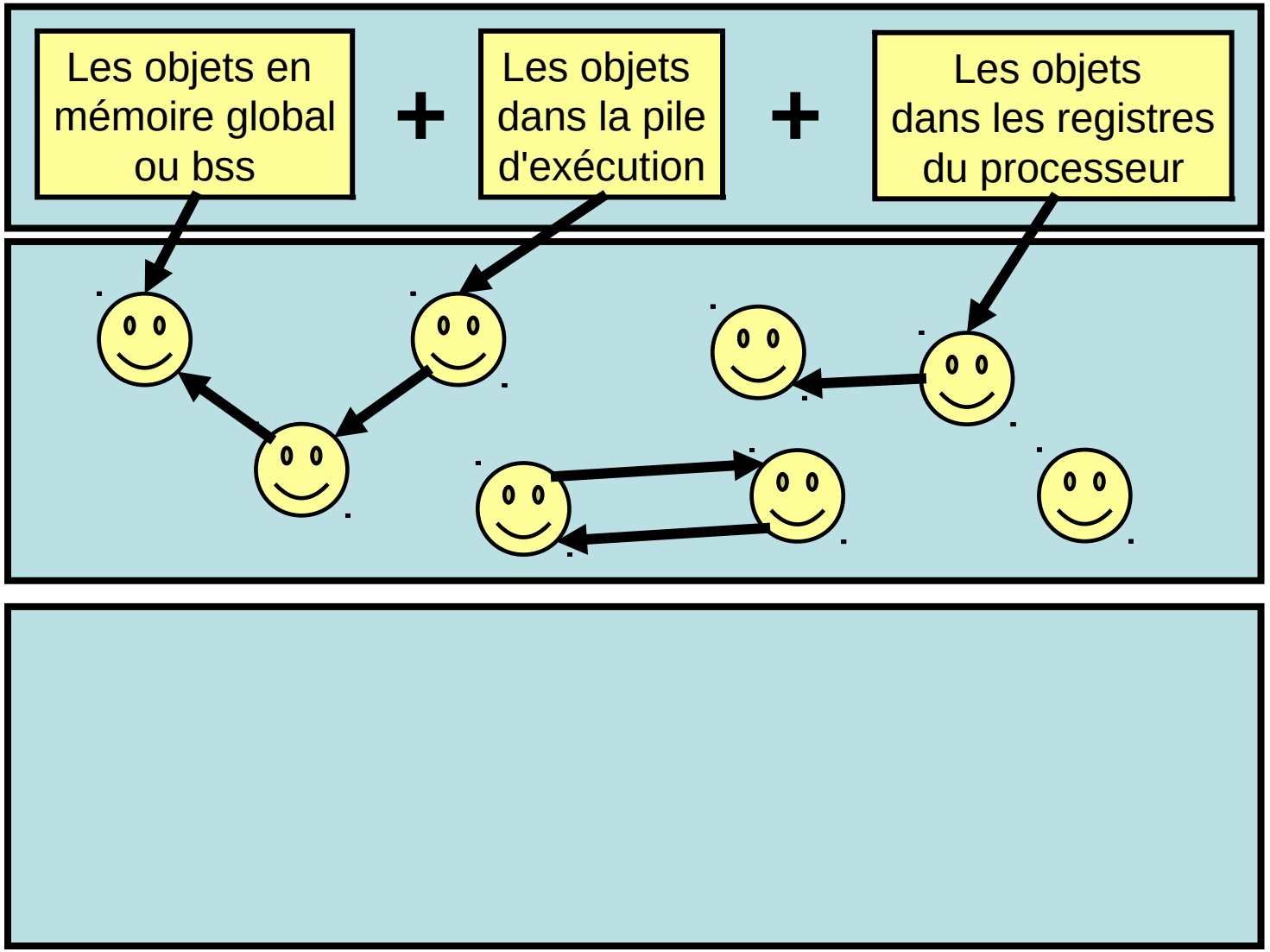
- Le plus Rapide !!!
- Pas de pb avec les références cycliques

Désavantages:

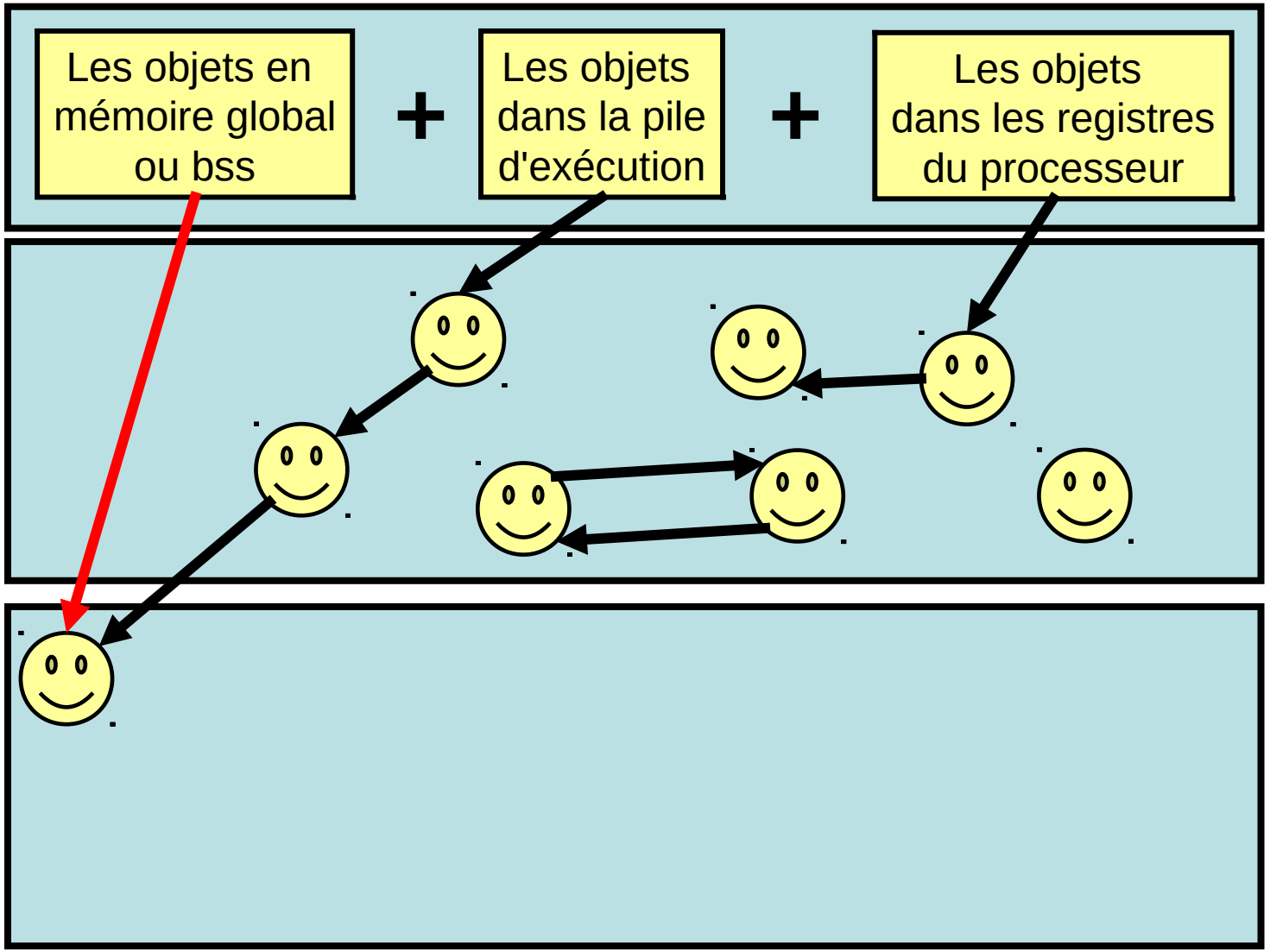
- Difficile à mettre en œuvre
  (dépendant du processeur)
- Programme stoppé
- Nécessité d'un bit de marquage

Utilisé dans Lisaac et SmartEiffel…

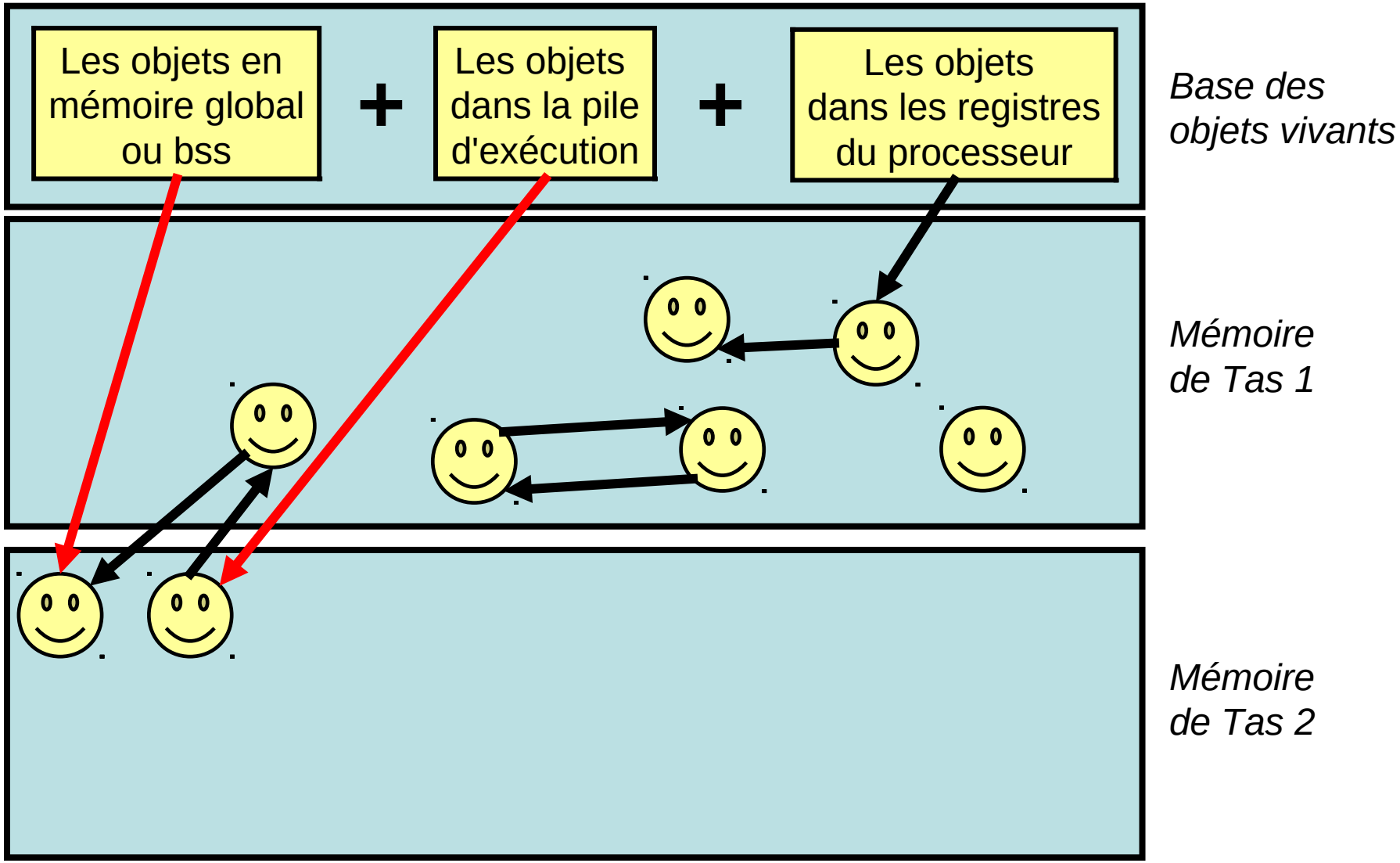
Recopie


Les objets en mémoire global ou bss
+
Les objets dans la pile d'exécution
+
Les objets dans les registres du processeur
Base des objets vivants
Mémoire de Tas 1
Mémoire de Tas 2

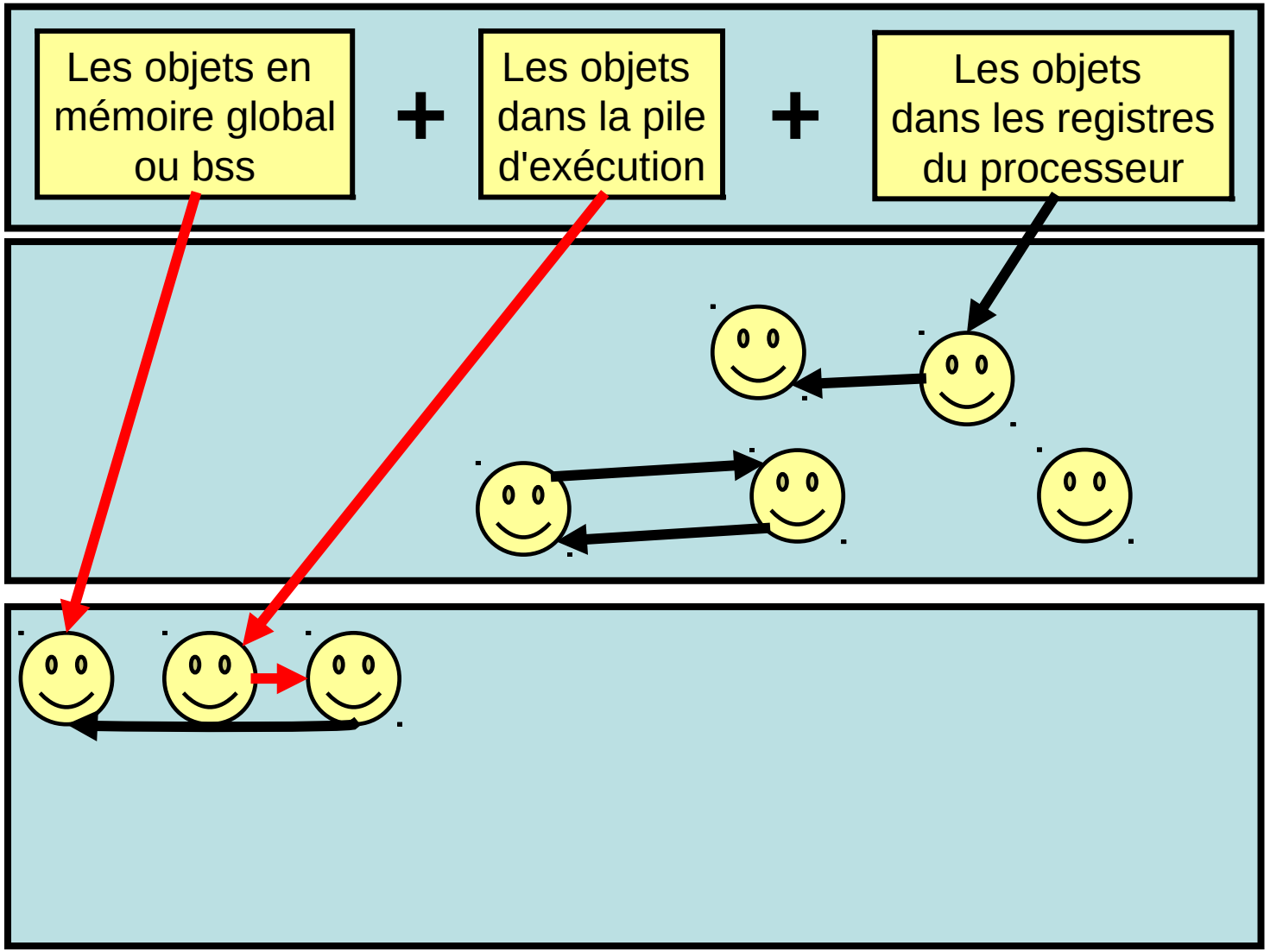
Recopie


Les objets en mémoire global ou bss
+
Les objets dans la pile d'exécution
+
Les objets dans les registres du processeur
Base des objets vivants
Mémoire de Tas 1
Mémoire de Tas 2

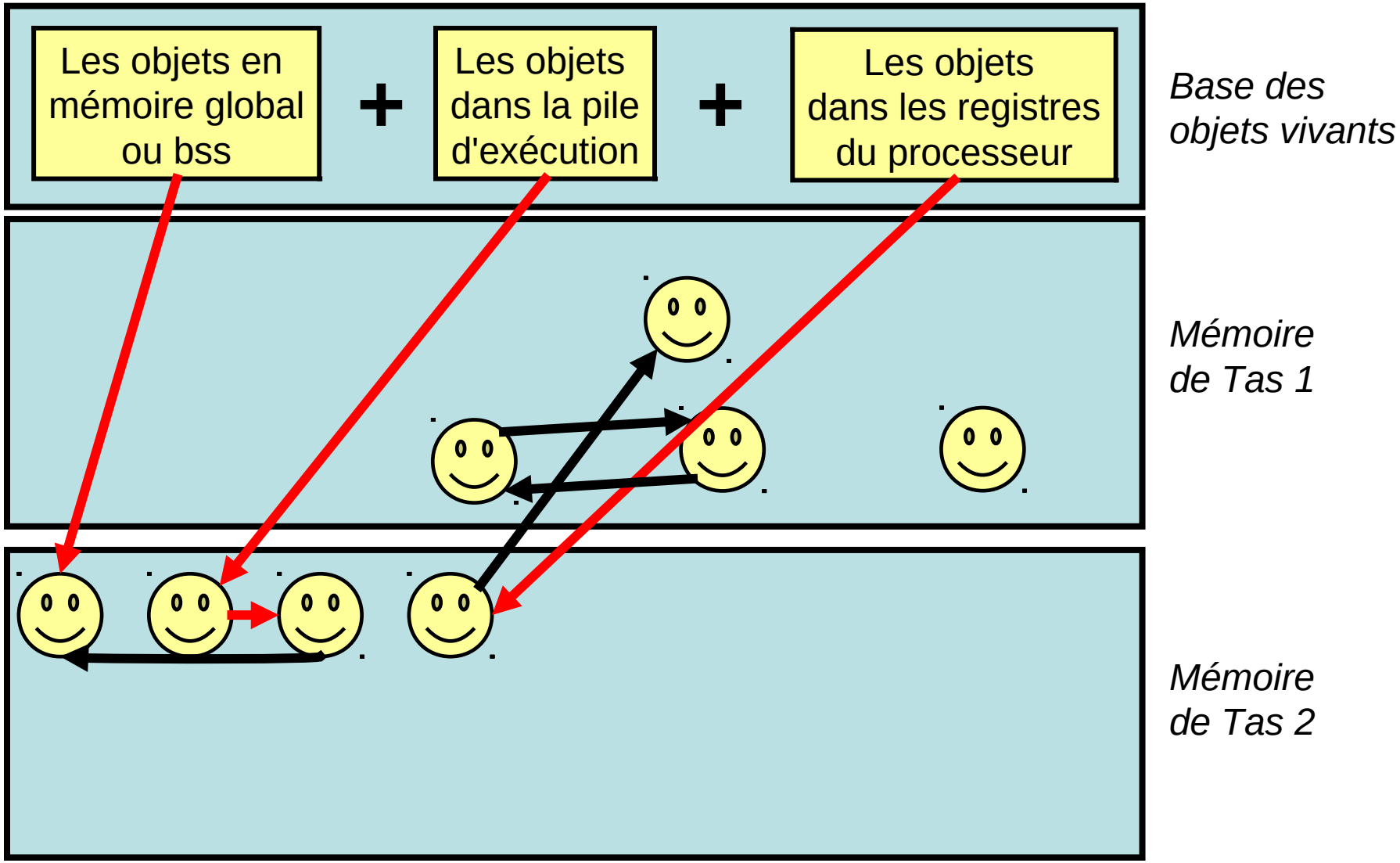
Recopie


Les objets en mémoire global ou bss
+
Les objets dans la pile d'exécution
+
Les objets dans les registres du processeur
Base des objets vivants
Mémoire de Tas 1
Mémoire de Tas 2

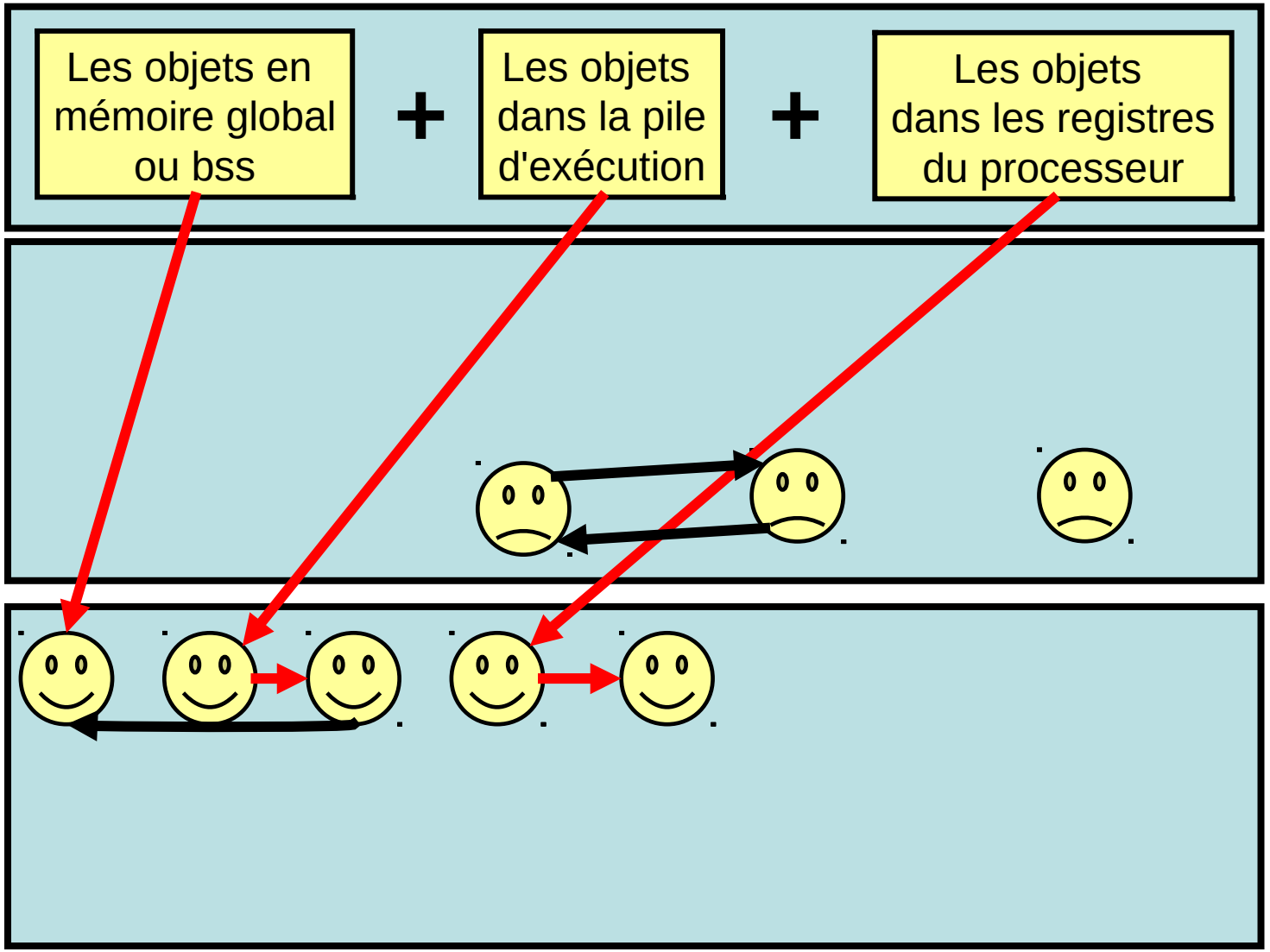
Recopie


Les objets en mémoire global ou bss
+
Les objets dans la pile d'exécution
+
Les objets dans les registres du processeur
Base des objets vivants
Mémoire de Tas 1
Mémoire de Tas 2

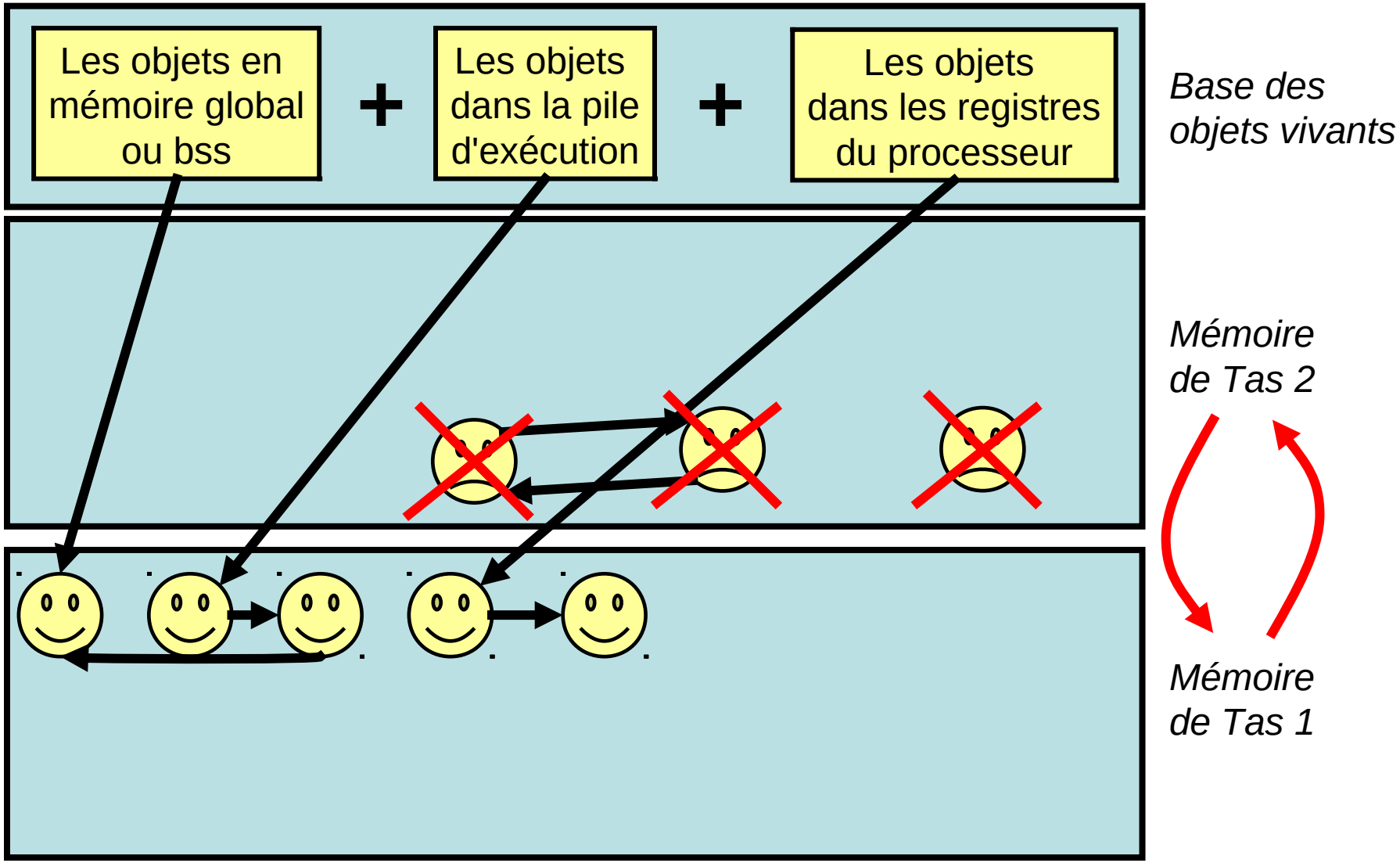
Recopie


Les objets en mémoire global ou bss
+
Les objets dans la pile d'exécution
+
Les objets dans les registres du processeur
Base des objets vivants
Mémoire de Tas 1
Mémoire de Tas 2

Recopie


Les objets en mémoire global ou bss
+
Les objets dans la pile d'exécution
+
Les objets dans les registres du processeur
Base des objets vivants
Mémoire de Tas 1
Mémoire de Tas 2

Recopie


Les objets en mémoire global ou bss
+
Les objets dans la pile d'exécution
+
Les objets dans les registres du processeur
Base des objets vivants
Mémoire de Tas 2
Mémoire de Tas 1

Recopie

Avantages:

- Pas de fragmentation interne
- Pas de bit de marquage

Désavantages:

- Maintenance des références difficiles
- Utilise le double de mémoire
- L'adresse des objets n'est pas fixe

Utilisé dans Lisp, ML et Java…