Mises en œuvre distribuées de programmes synchrones

Environnement de **P**rogrammation pour **A**pplications **T**emps-**R**éel

Pascal Aubry

http://perso. univ-rennes1.fr/pascal.aubry

Distribution

- Pourquoi ?
  - délocalisation géographique (répartition)
  - gain de performance (parallélisme)
  - tolérance aux fautes (redondance)
- Comment ?
  - parallélisme intrinsèque au langage
  - programmation impérative des communications
  - parallélisation automatique
  - ...

Distribution du synchrone

- Pour garder l'approche synchrone
  - systèmes réactifs
  - cadre formel
    - automatisation des transformations
    - outils de validation
  - approche algorithmique
    - indépendance vis-à-vis de l 'architecture

[Girault 94] (Lustre)

[Caillaud 94] (automates)

Signal

- [Chéron 91] :
  - Transformations syntaxiques de programmes Signal
- [Maffeïs 93] :
  - ordonnancement de flots synchrones
- [Kountouris], [Machard] :
  - temps d'exécution, O.S. temps-réel
- [Sorel 92] : Syndex
  - adéquation algorithme architecture

Préambule

théorie

méthodologie

application

modèle

correction

intégration

Préambule

théorie

modèle

méthodologie

correction

application

intégration

Préambule

méthodologie correction

application intégration

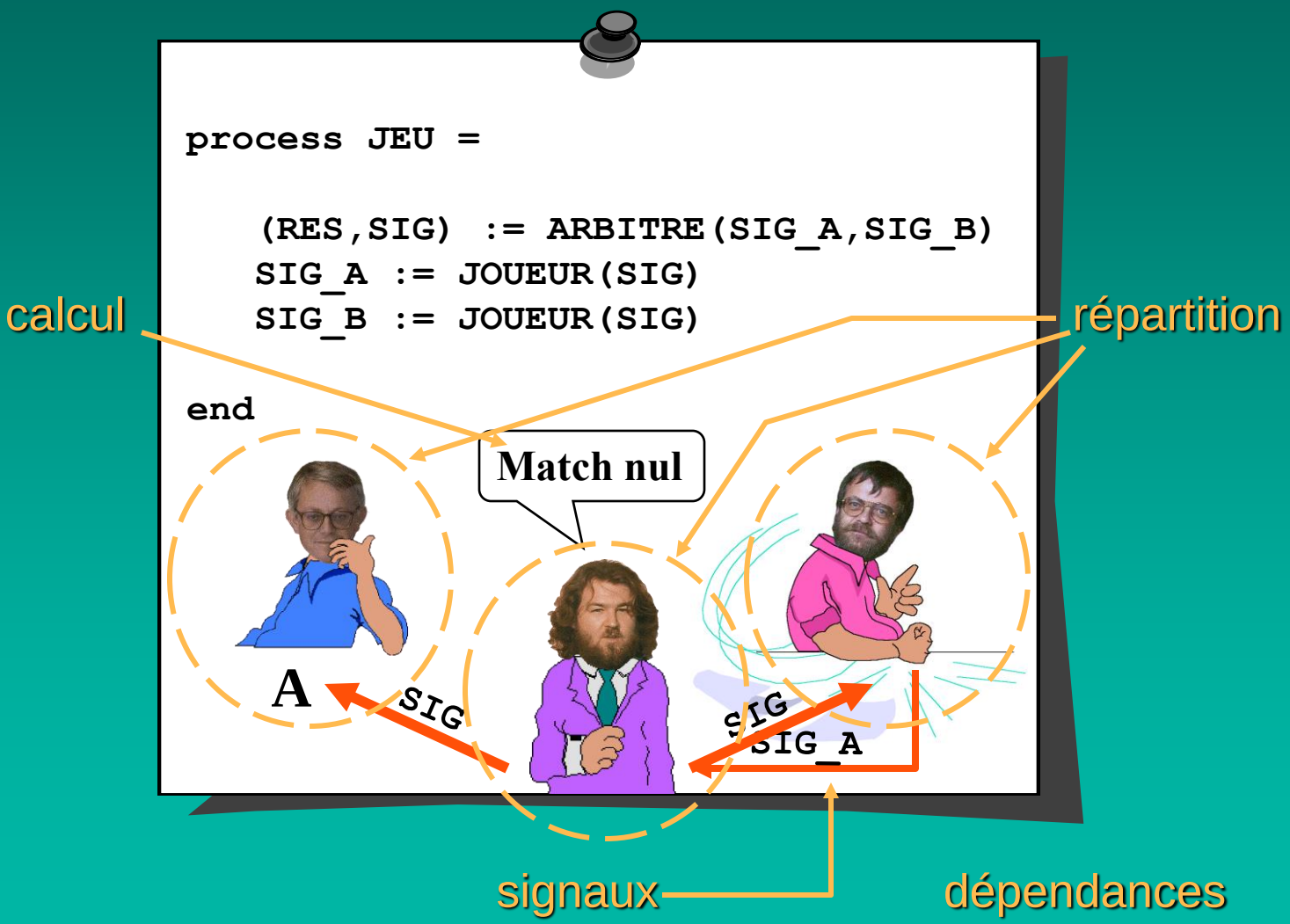
critères qualitatifs de la distribution

Illustration

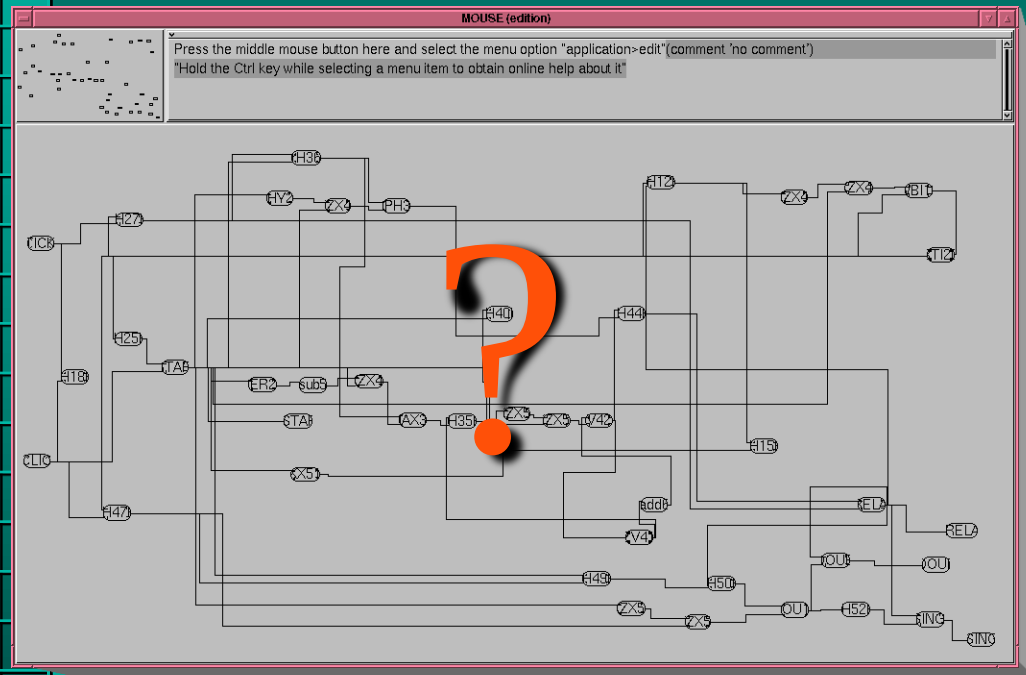
```
process JEU =

    (RES,SIG)  := ARBITRE(SIG_A,SIG_B)
    SIG_A := JOUEUR(SIG)
    SIG_B := JOUEUR(SIG)

end
```

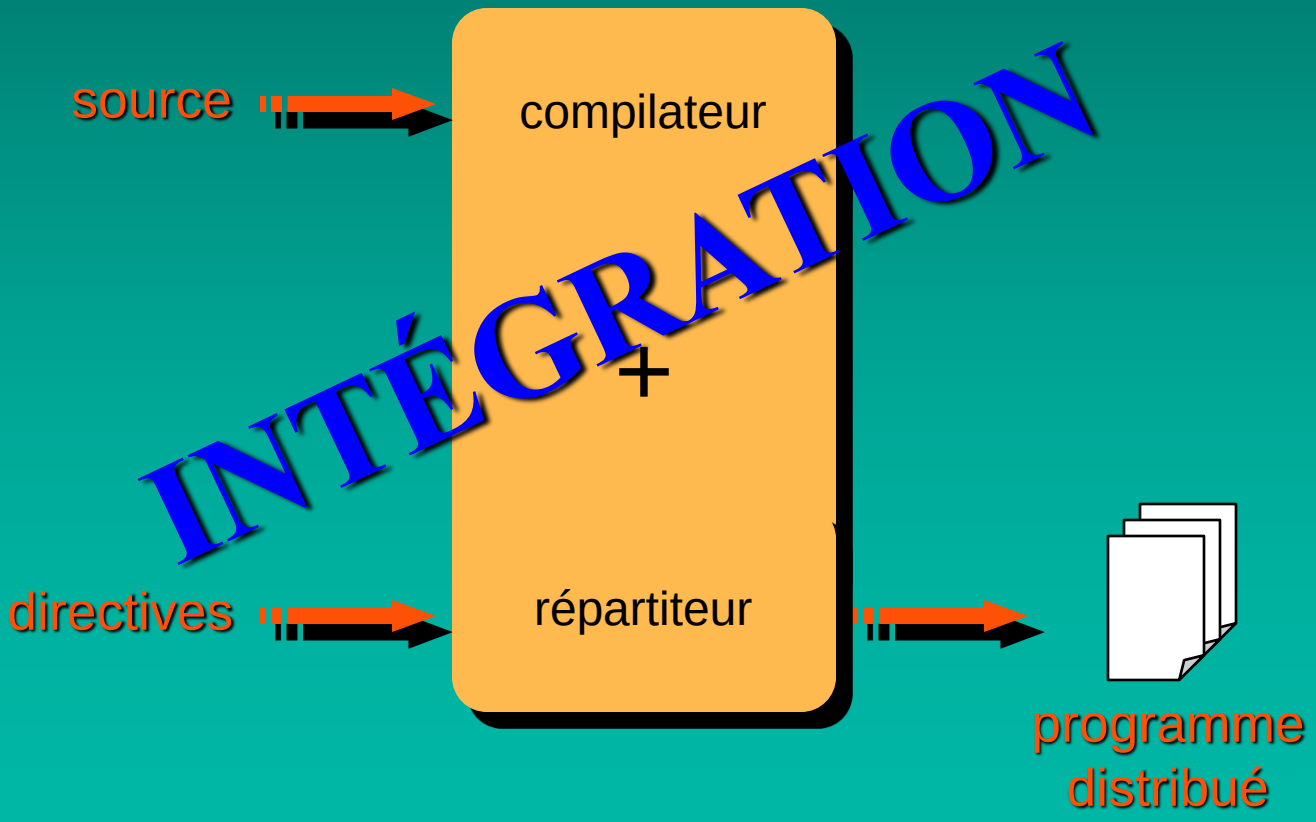


Répartition avec Syndex

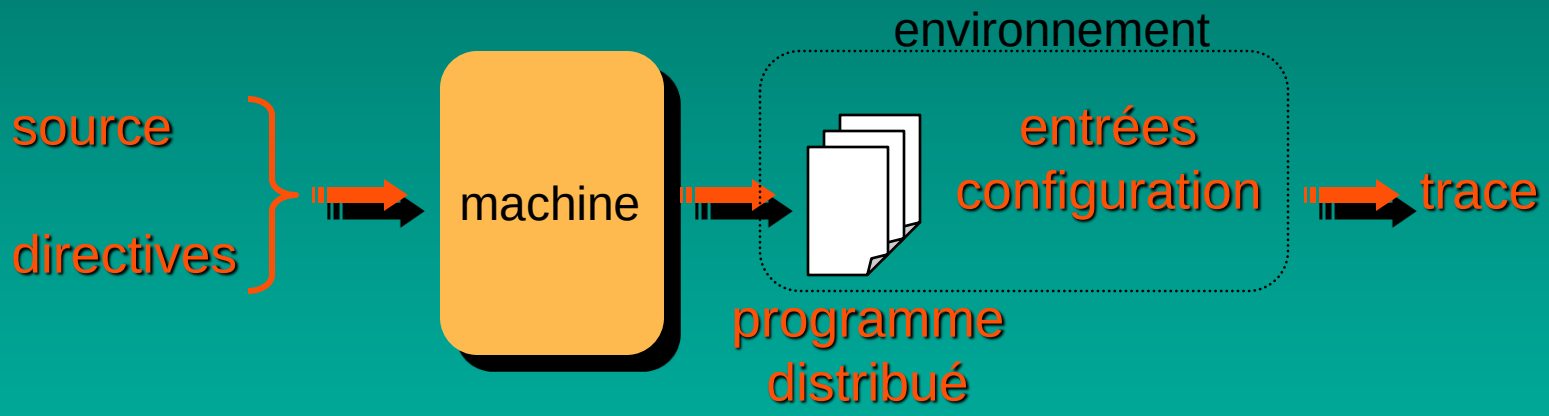


programme distribué

Plus convivial...



Le point de vue de l'utilisateur



Le point de vue de l'utilisateur

source

directives

trace

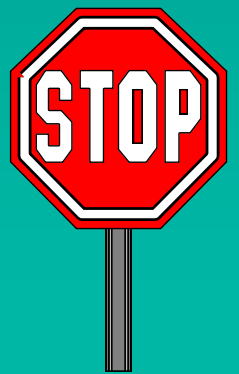
CORRECTIONS (modèle)

Mais au fait, comment ça marche ton truc ?

Euh… je sais pas, chef.

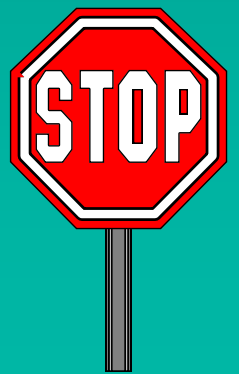
Les corrections obligatoires

- la correction fonctionnelle
  - Les calculs effectués correspondent-ils aux calculs spécifiés ?



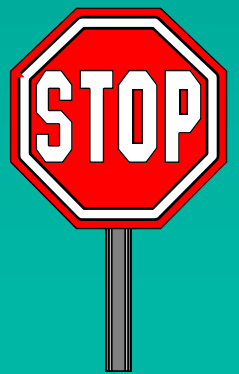
Les corrections obligatoires

- la correction fonctionnelle
- la correction spatiale
  - la mémoire nécessaire à l'exécution est-elle bornée ?



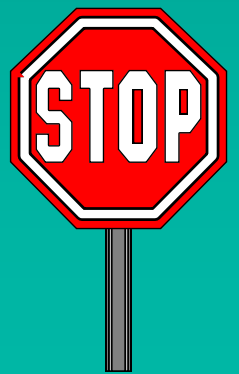
Les corrections obligatoires

- la correction fonctionnelle
- la correction spatiale
- la correction temporelle
  - le temps de réponse du système est-il borné ?



Les corrections obligatoires

- la correction fonctionnelle
- la correction spatiale
- la correction temporelle
- l'équité
  - une partie du programme s'exécute-t-elle infiniment plus vite qu'une autre ?



Les corrections obligatoires

- la correction fonctionnelle
- la correction spatiale
- la correction temporelle
- l'équité

Propriétés classiques

Les corrections facultatives

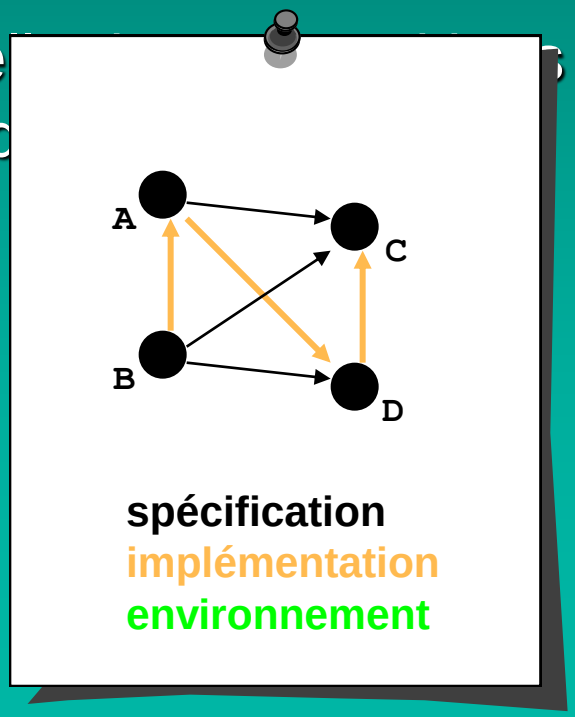
- la correction flot-de-données
  - l'ordre d'observation des signaux est-il fidèle à l 'ordre spécifié ?

Les corrections facultatives

- la correction flot-de-données
- la correction synchrone
  - Les instants logiques peuvent-ils se chevaucher ?

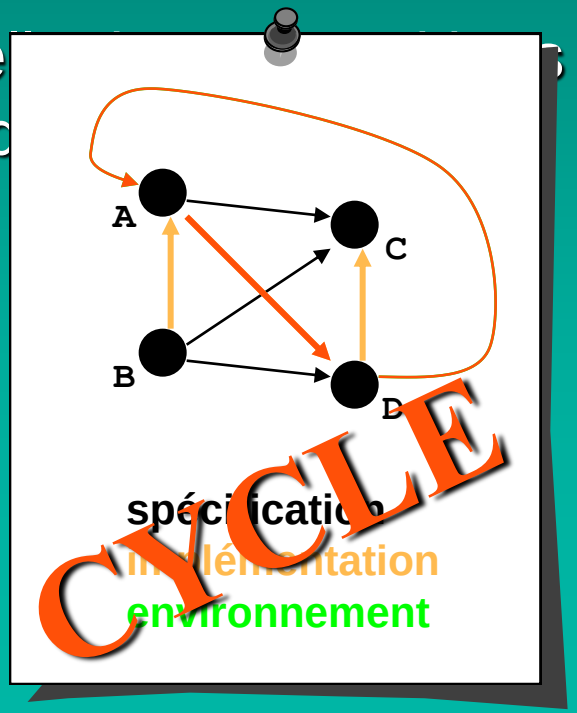
Les corrections facultatives

- la correction flot-de-données
- la correction synchrone
- la correction compositionnelle
  - l'exécution fait-e… sur son environnement c…



Les corrections facultatives

- la correction flot-de-données
- la correction synchrone
- la correction compositionnelle
  - l'exécution fait-e… sur son environnement c…

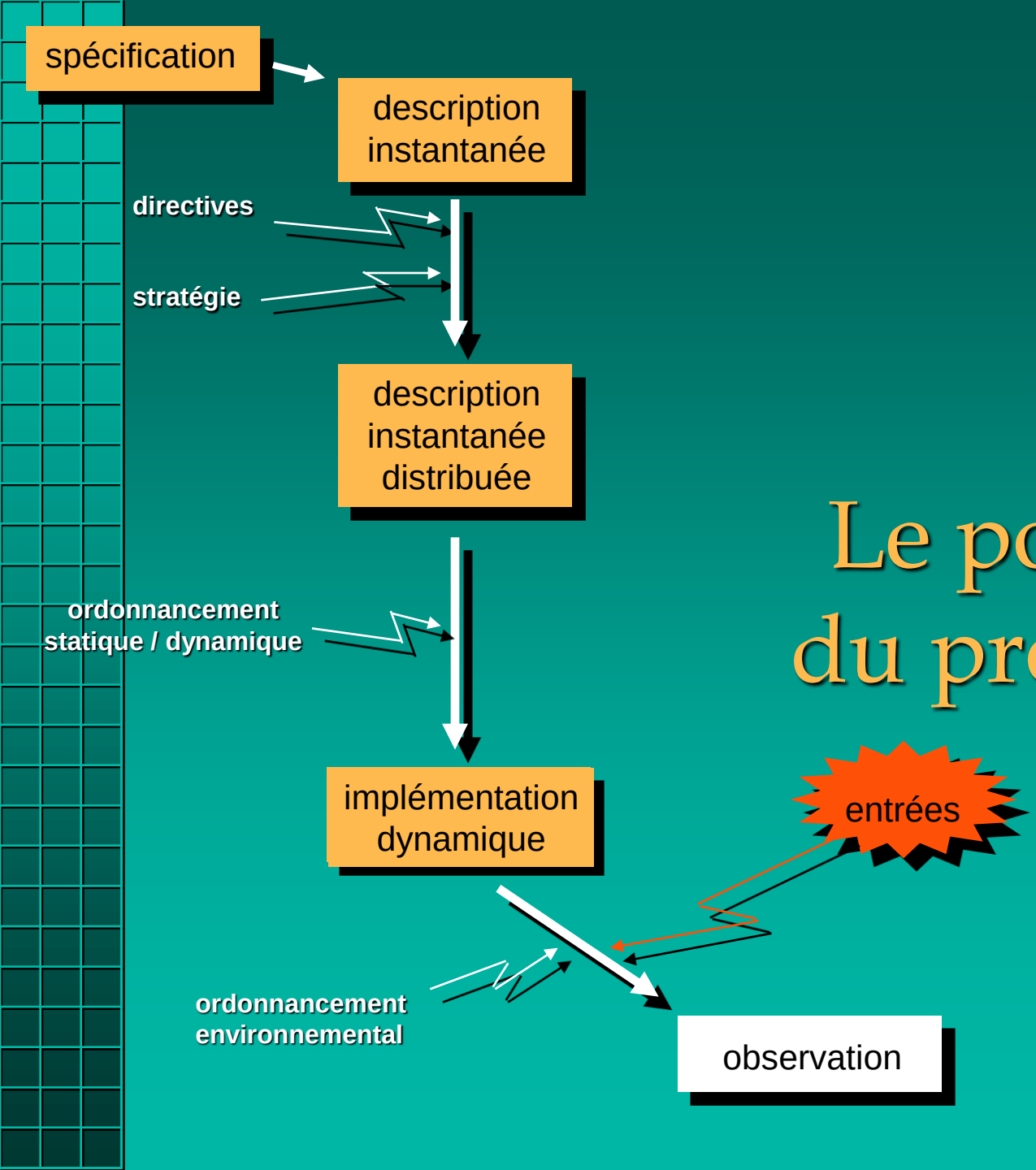


Les corrections facultatives

- la correction flot-de-données
- la correction synchrone
- la correction compositionnelle

Propriétés spécifiques

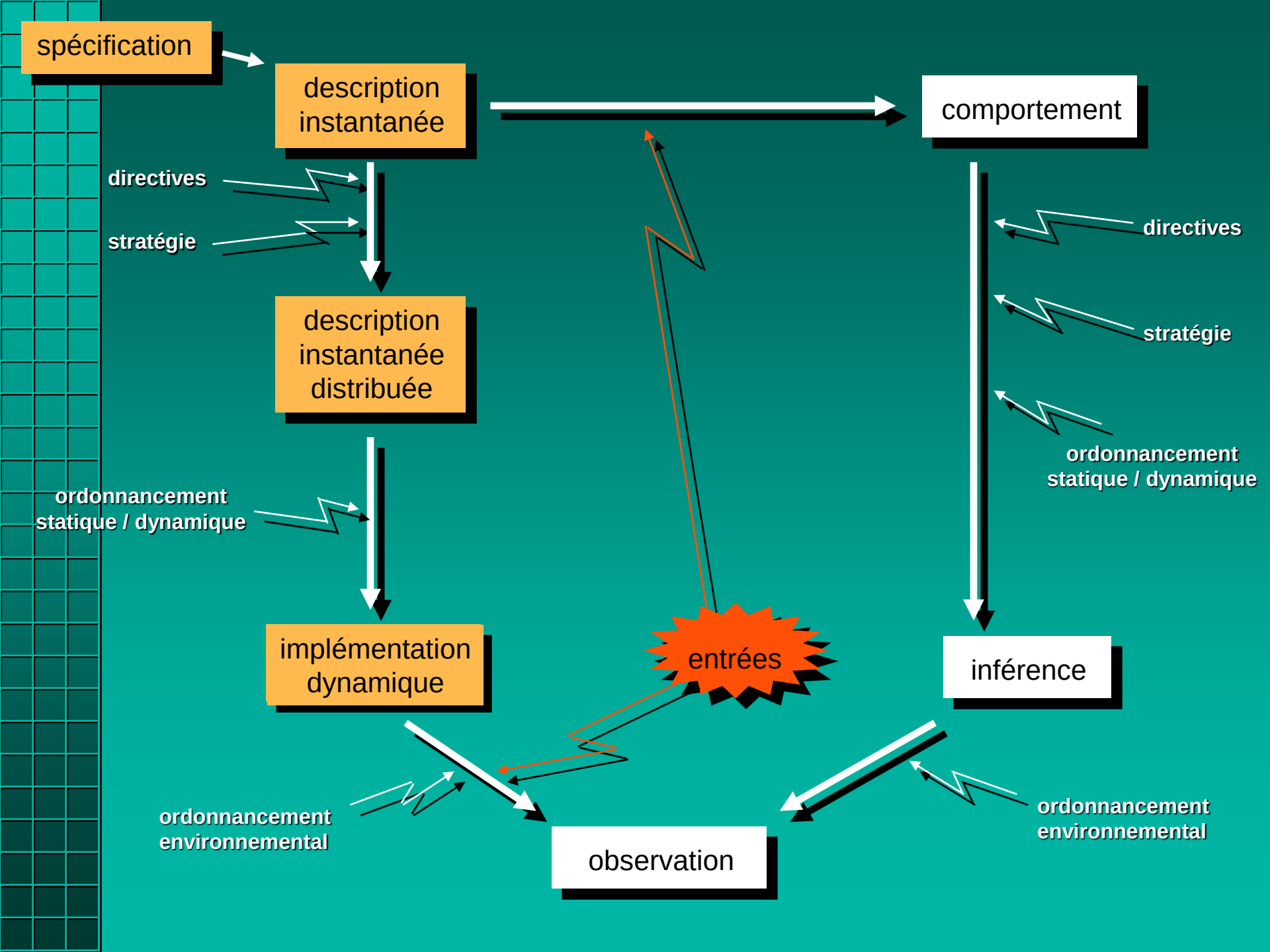
Le point de vue du programmeur

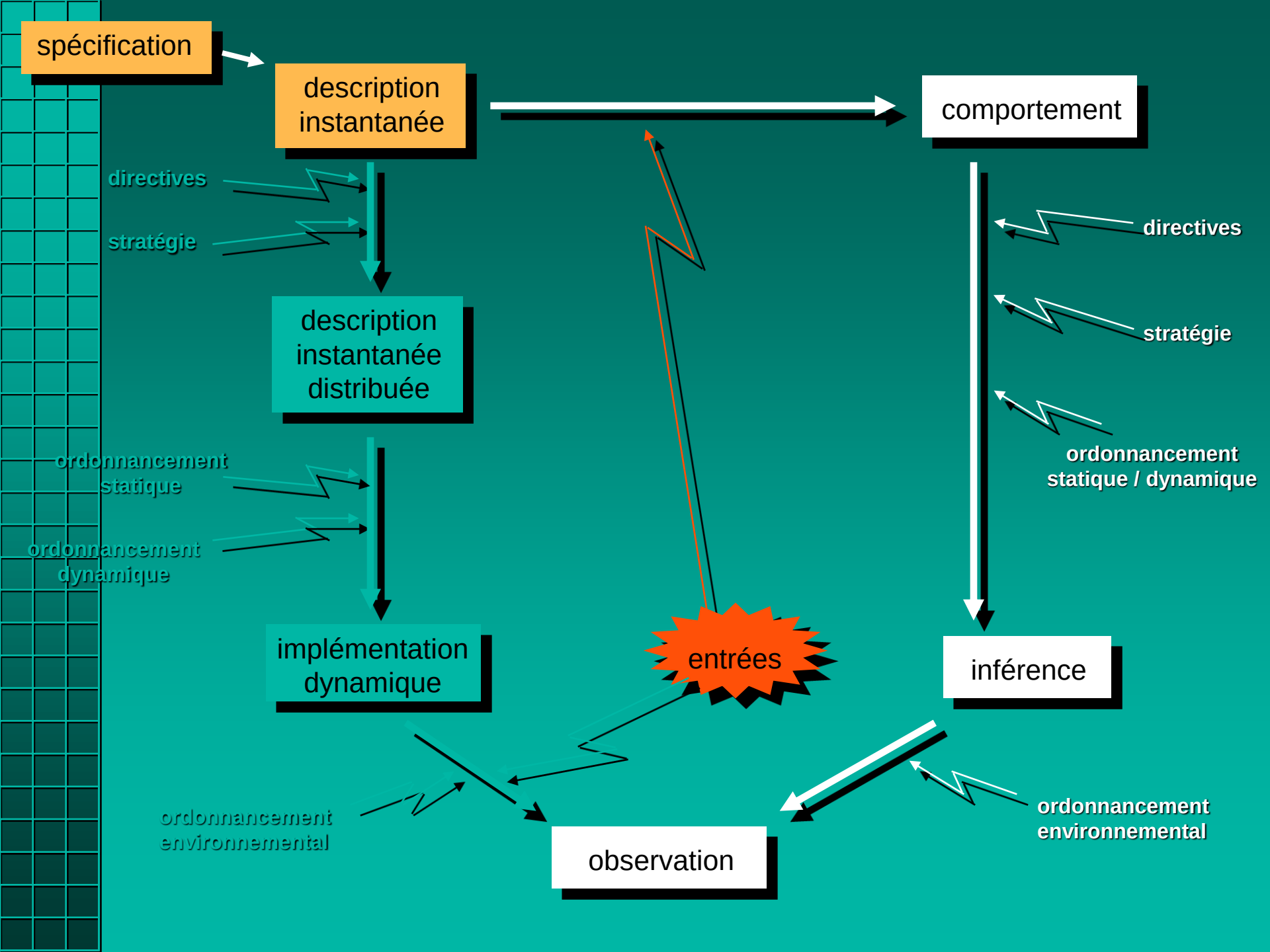


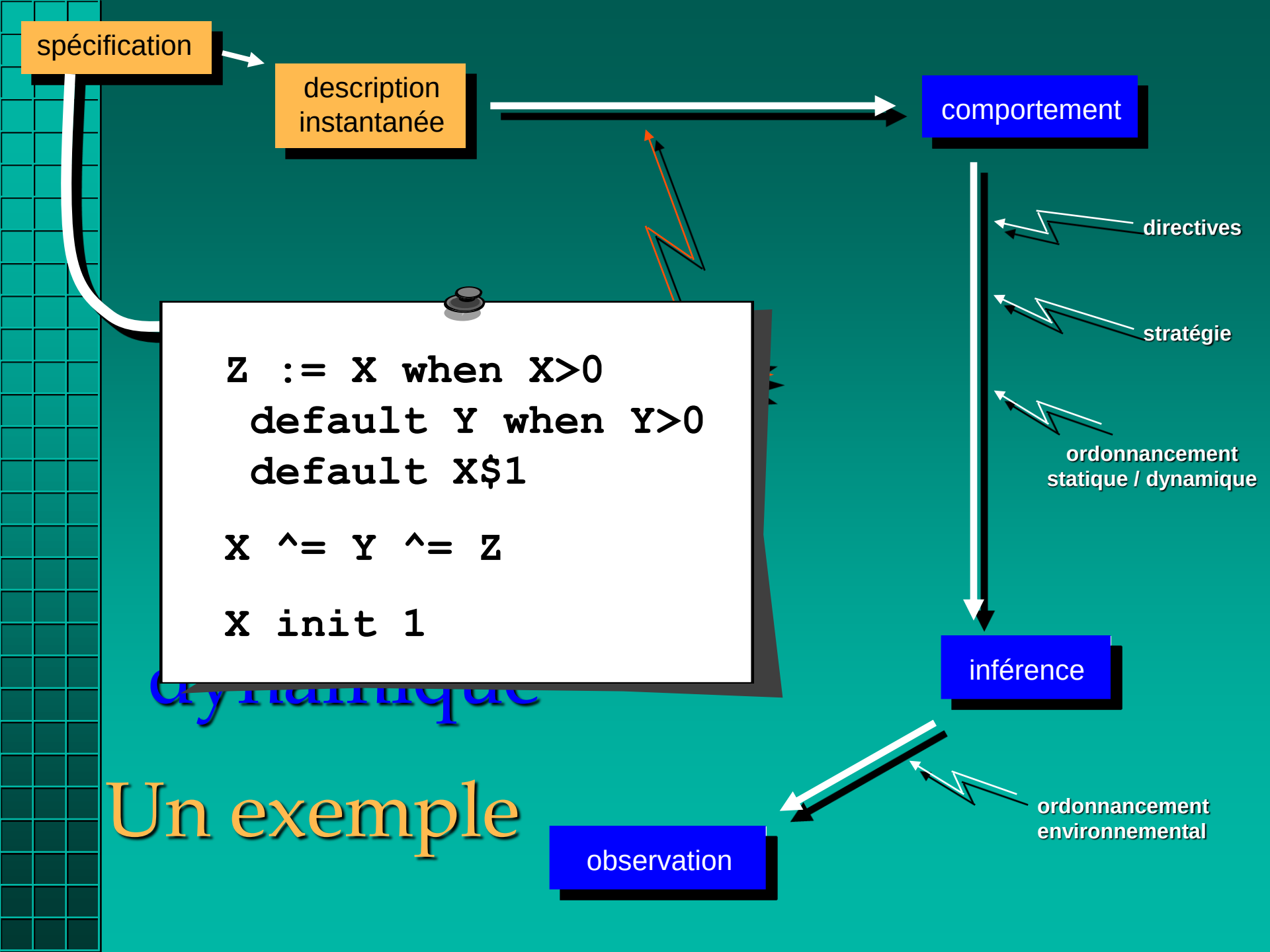
Les corrections d'une mise en œuvre ...

- sont des propriétés sur les occurrences des signaux du programme.
- Les analyses de propriétés statiques ne suffisent pas.

NÉCESSITÉ D'UN MODÈLE DYNAMIQUE


spécification
description instantanée
comportement
directives
stratégie
directives
stratégie
description instantanée distribuée
ordonnancement statique / dynamique
ordonnancement statique / dynamique
implémentation dynamique
entrées
inférence
ordonnancement environnemental
observation
ordonnancement environnemental


spécification
description instantanée
comportement
directives
stratégie
description instantanée distribuée
directives
stratégie
ordonnancement statique / dynamique
ordonnancement statique
ordonnancement dynamique
implémentation dynamique
entrées
inférence
ordonnancement environnemental
observation
ordonnancement environnemental


spécification
description instantanée
comportement
directives
stratégie
ordonnancement statique / dynamique
Z := X when X>0
default Y when Y>0
default X$1
X ^= Y ^= Z
X init 1
inférence
ordonnancement environnemental
observation

Un exemple

Un exemple

spécification

```
Z := X when X>0
  default Y when Y>0
  default X$1
X ^= Y ^= Z
X init 1
```

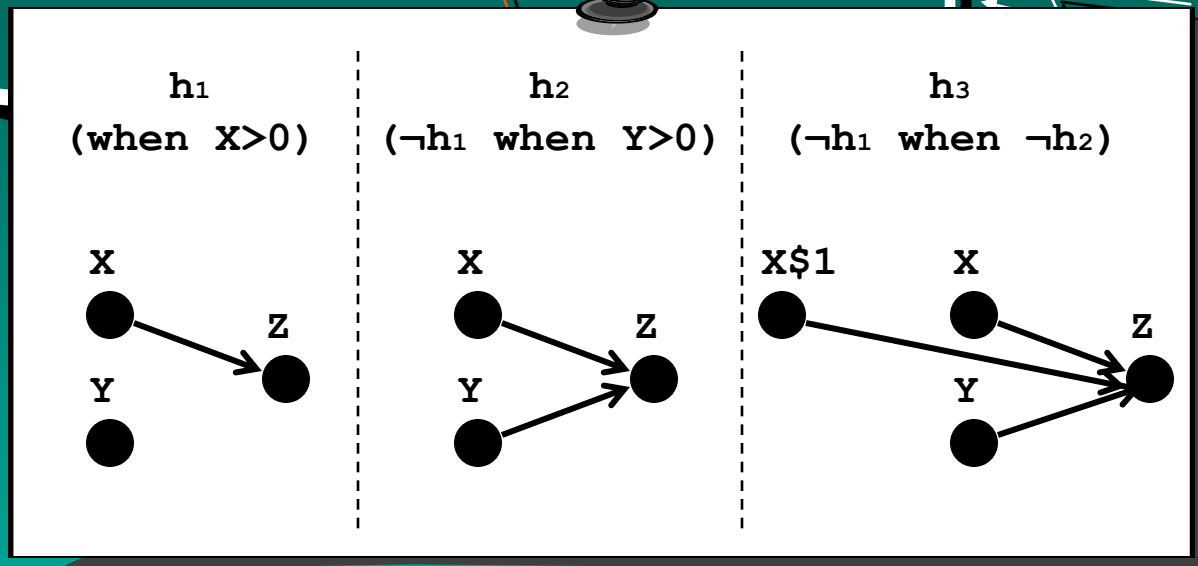
description instantanée

comportement

directives

tratégie

ncement
lynamique



inférence

ordonnancement
environnemental

observation

spécification

```
Z := X when X>0
  default Y when Y>0
  default X$1
X ^= Y ^= Z
X init 1
```

description instantanée

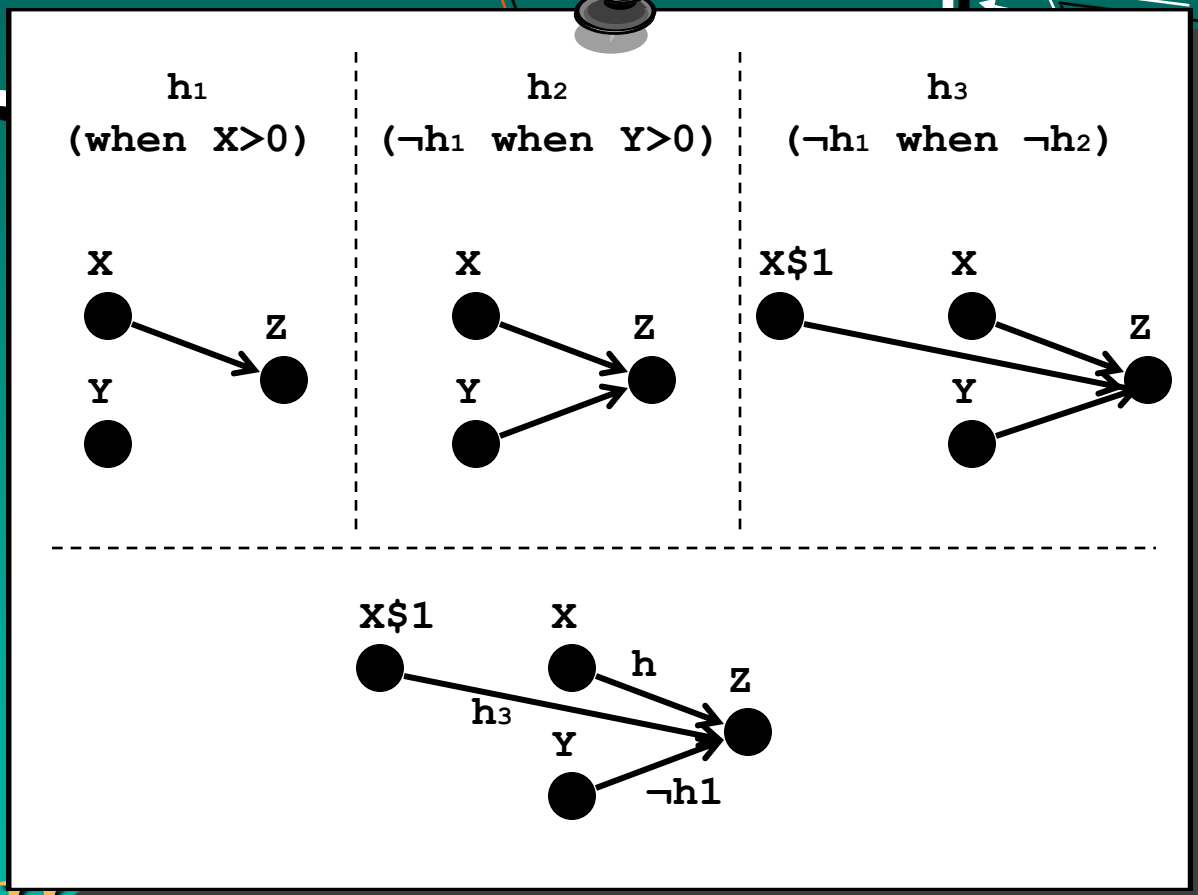
comportement

directives

tratégie

ncement
dynamique

ment
environnemental



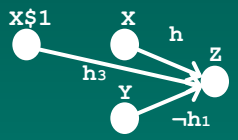
Un exemple

observation

spécification

```
Z := X when X>0
  default Y when Y>0
  default X$1
X ^= Y ^= Z
X init 1
```

description instantanée



comportement

directives

stratégie

ordonnancement statique / dynamique

entrées

```
( X, Y)

(-1, 2)
( 3, 7)
( 0,-1)
( 2, 0)
  ...
```

inférence

ordonnancement environnemental

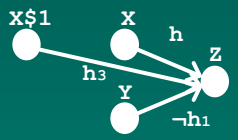
observation

Un ex

spécification

```
Z := X when X>0
  default Y when Y>0
  default X$1
X ^= Y ^= Z
X init 1
```

description instantanée



comportement

(-1, 2)
(3, 7)
(0,-1)
(2, 0)
...

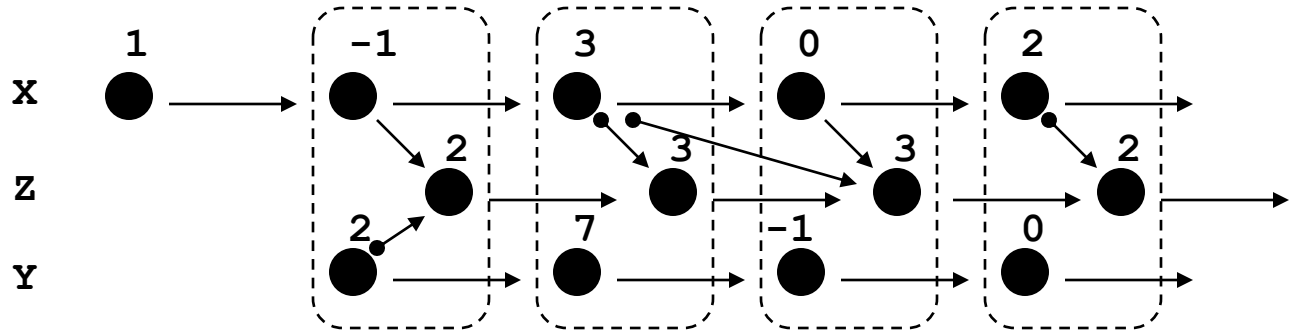
entrées

ectives

stratégie

ordonnancement
statique / dynamique

État initial :



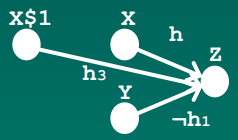
Instants : 1 2 3 4 ...

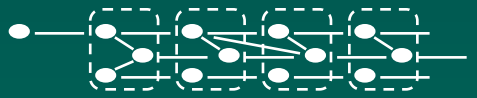
ancement
nemental

spécification

```
Z := X when X>0
  default Y when Y>0
  default X$1
X ^= Y ^= Z
X init 1
```

description instantanée





comportement

directives

stratégie

ordonnancement statique / dynamique

```
(-1, 2)
( 3, 7)
( 0,-1)
( 2, 0)
...
```

entrées

Une mise en œuvre mono-processeur séquentielle

inférence

ordonnancement environnemental

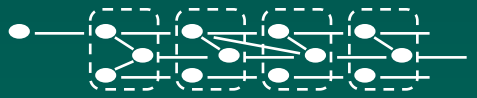
observation

Un exemple

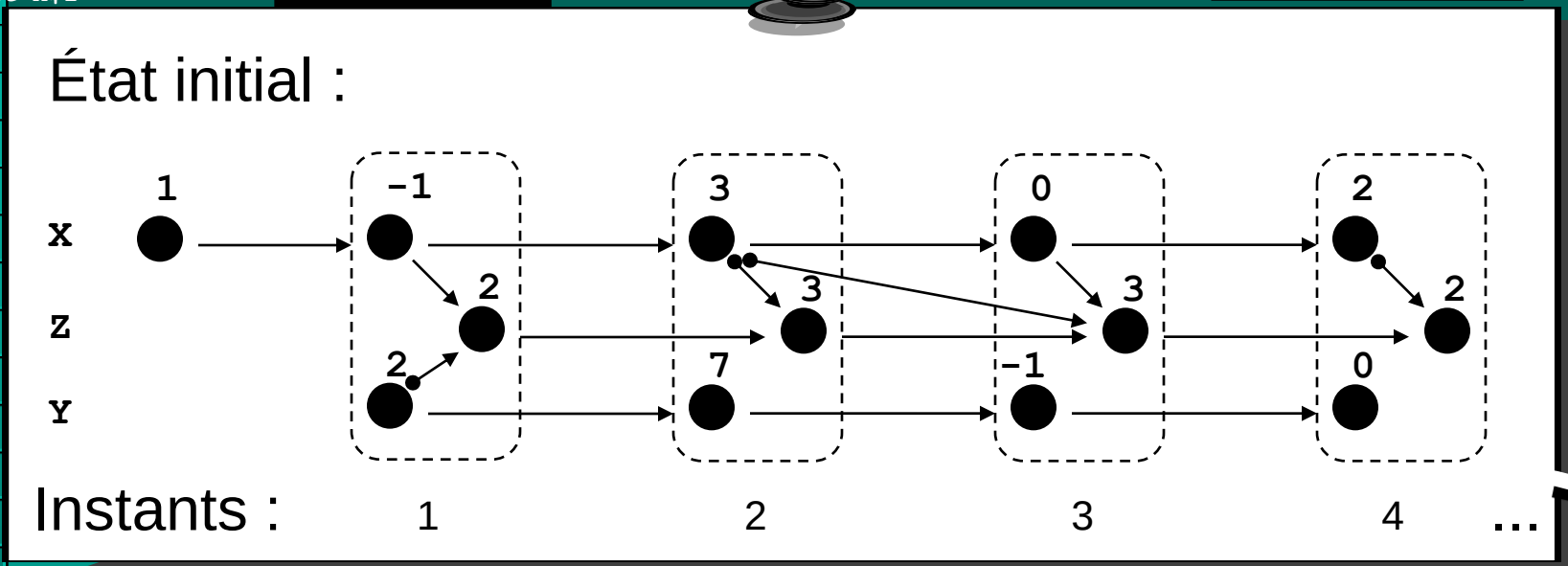
spécification

```
Z := X when X>0
  default Y when Y>0
  default X$1
X ^= Y
X init
```

description instantanée



comportement



ent

mono-processeur séquentielle

inférence

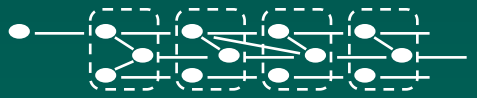
Un exemple

observation

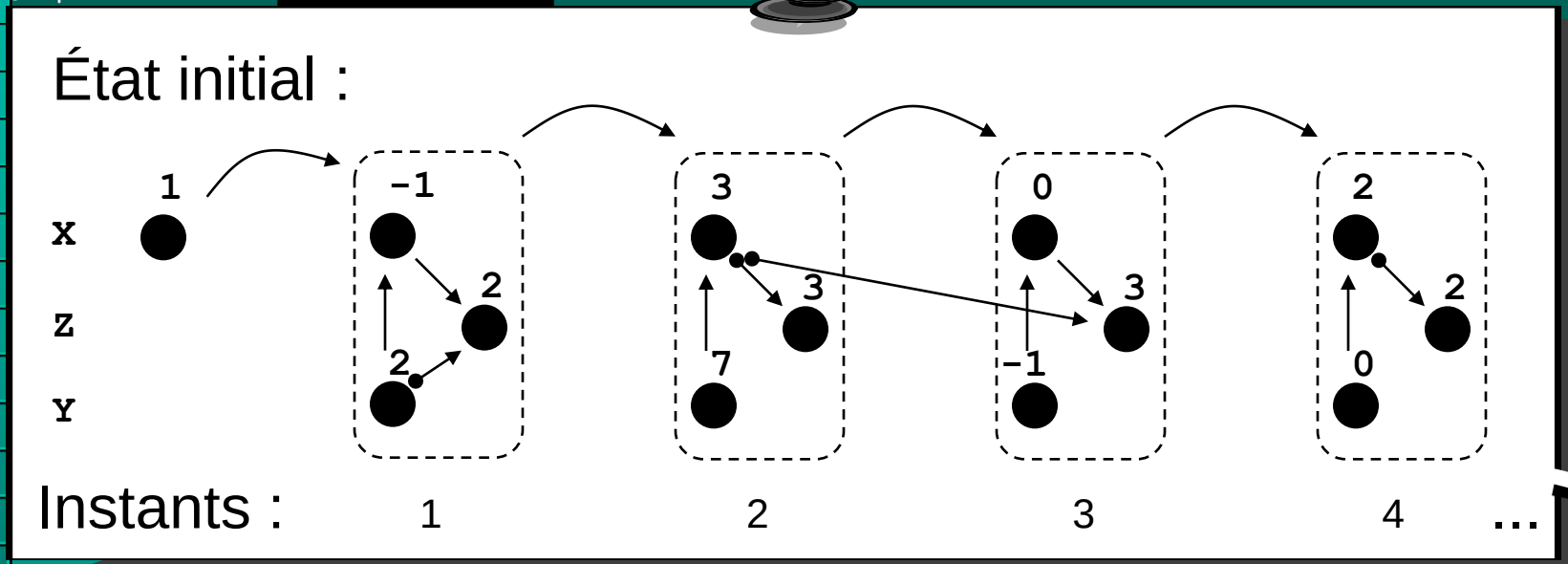
spécification

```
Z := X when X>0
  default Y when Y>0
  default X$1
X ^= Y
X init
```

description instantanée



comportement



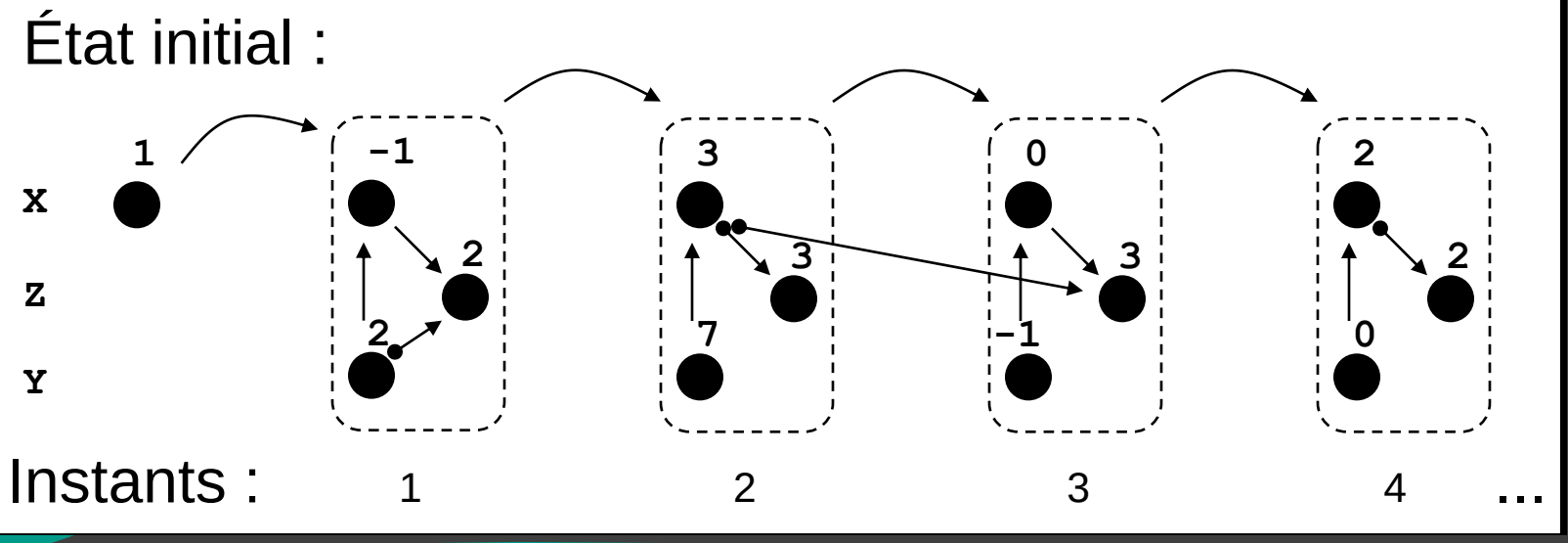
mono-processeur séquentielle

inférence

Un exemple

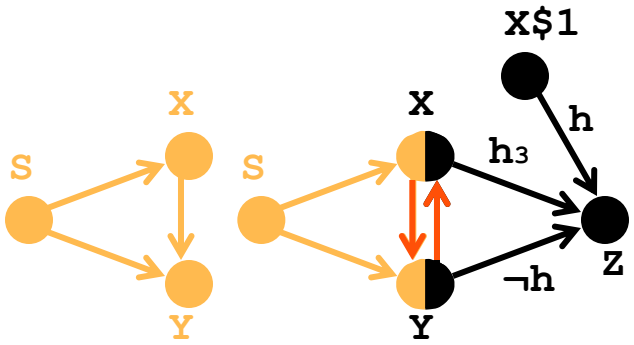
observation

Résultats :



correction fonctionnelle : ✔
correction spatiale : ✔
correction temporelle : ✔
équité : ✔
correction synchrone : ✔
correction flots-de-données : ✔
correction compositionnelle :

Résultats :



correction fonctionnelle : ✔
correction spatiale : ✔
correction temporelle : ✔
équité : ✔
correction synchrone : ✔
correction flots-de-données : ✔
correction compositionnelle : ⊘

C'est dommage...

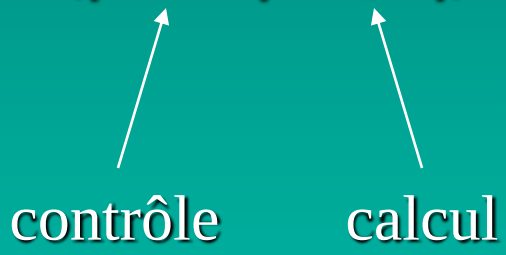
Méthodologie d'implémentation parallèle

- Préservation du formalisme Signal
  - intégration, récupération
  - validation (correction fonctionnelle)
- modularité

Méthodologie d'implémentation parallèle

- compilation

$$P \equiv (| C | Q |)$$



[Besnard 92]

compilation

$P \equiv (| C | Q |)$

Méthodologie d'implémentation parallèle

- répa
  - a
    - .
    - .
  - a
    - .
    - .

)

ls du

seur n

```
process JEU =

topology "MASTER,cheyenne.irisa.fr"
topology "PL_A,  guernica.irisa.fr"
topology "PL_B,  megalo.irisa.fr"

run_on {a} "PL_A"
run_on {b} "PL_B"

m:: (RES,SIG) := ARBITRE(SIG_A,SIG_B)
a:: SIG_A := JOUEUR(SIG)
b:: SIG_B := JOUEUR(SIG)

end
```

compilation

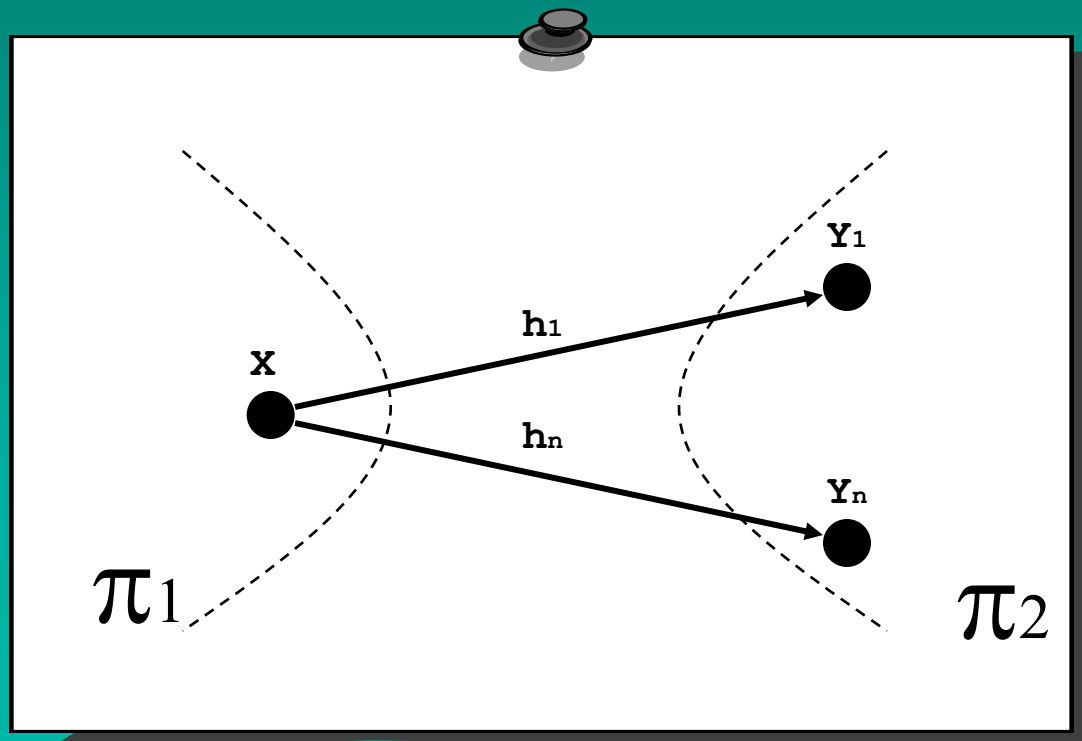
$P \equiv (| C | Q |)$

répartition des données

$Q \equiv (| Q_1 | \ldots | Q_n |)$

Méthodologie d'implémentation parallèle

- communication des données



compilation

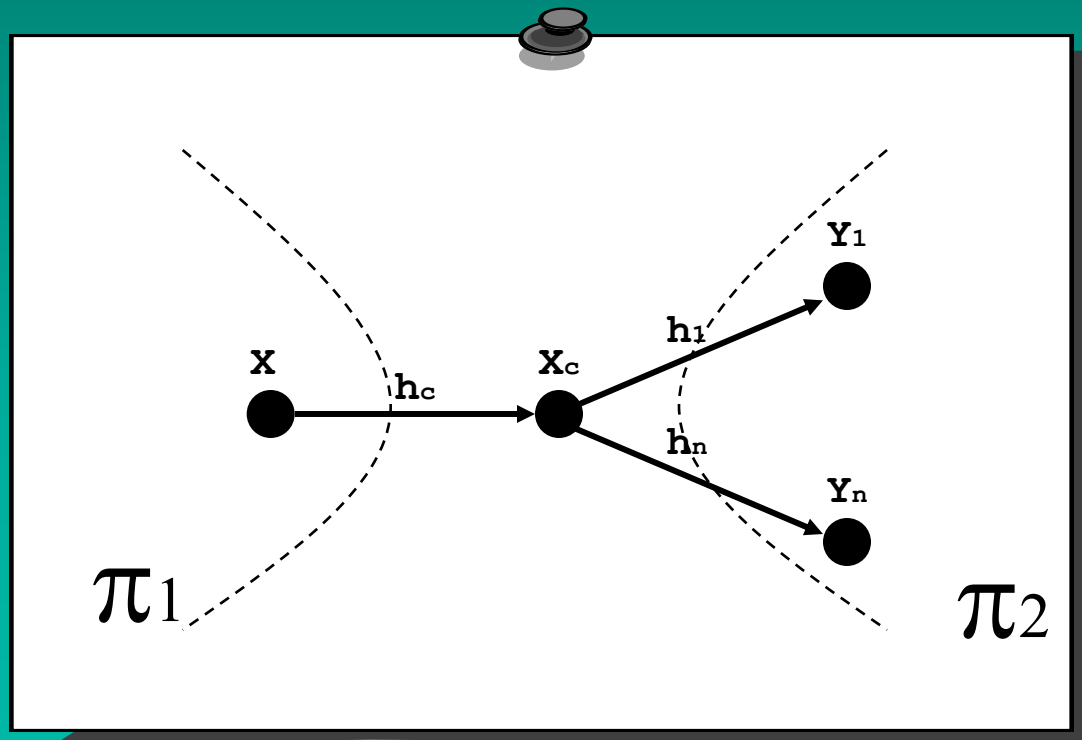
$P \equiv (| C | Q |)$

répartition des données

$Q \equiv (| Q_1 | \ldots | Q_n |)$

Méthodologie d'implémentation parallèle

- communication des données



Méthodologie d'implémentation parallèle

compilation

$P \equiv (| C | Q |)$

répartition des données

$Q \equiv (| Q_1 | \ldots | Q_n |)$

communication des données

$Q \equiv (| Q_1 | \ldots | Q_n |) | (| Qle_1 | \ldots | Qle_n |)$

- répartition du contrôle

$$C \equiv (| C_1 | \ldots | C_n |)$$

contrôle des calculs du processeur 1

contrôle des calcul du processeur n

[Aubry, Machard 96]

Méthodologie d'implémentation parallèle

compilation

$P \equiv (| C | Q |)$

répartition des données

$Q \equiv (| Q_1 | \ldots | Q_n |)$

communication des données

$Q \equiv (| Q_1 | \ldots | Q_n |) | (| Qle_1 | \ldots | Qle_n |)$

répartition du contrôle

$C \equiv (| C_1 | \ldots | C_n |)$

- communication du contrôle

`h`, horloge produite sur $\pi 1$ et utilisée sur $\pi 2$;

une horloge ne peut être communiquée de manière asynchrone.

`r`, une horloge plus rapide ;

```
b := h default not r
```

on communique `b` de $\pi 1$ vers $\pi 2$.

Méthodologie d'implémentation parallèle

compilation

$P \equiv (| C | Q |)$

répartition des données

$Q \equiv (| Q_1 | \ldots | Q_n |)$

communication des données

$Q \equiv (| Q_1 | \ldots | Q_n |) | (| Qle_1 | \ldots | Qle_n |)$

répartition du contrôle

$C \equiv (| C_1 | \ldots | C_n |)$

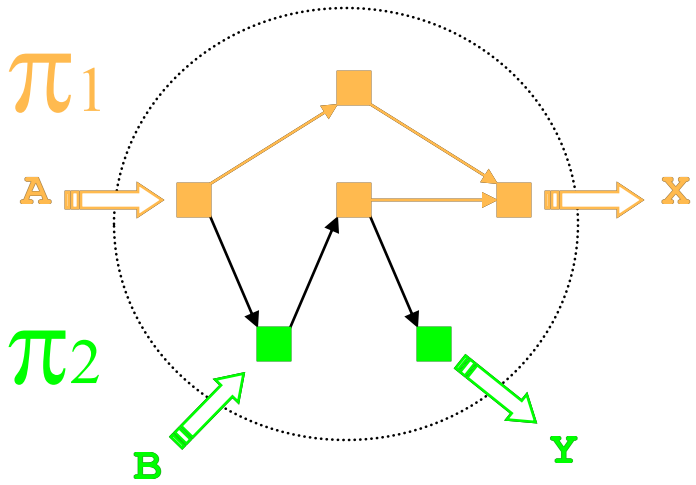
communication du contrôle

$C \equiv (| C_1$

- extraction des programmes distribués

P_i

Un exemple :



Méthodologie d'implémentation parallèle

compilation

$P \equiv (| C | Q |)$

répartition des données

$Q \equiv (| Q_1 | \ldots | Q_n |)$

communication des données

$Q \equiv (| Q_1 | \ldots | Q_n |) | (| Qle_1 | \ldots | Qle_n |)$

répartition du contrôle

$C \equiv (| C_1 | \ldots | C_n |)$

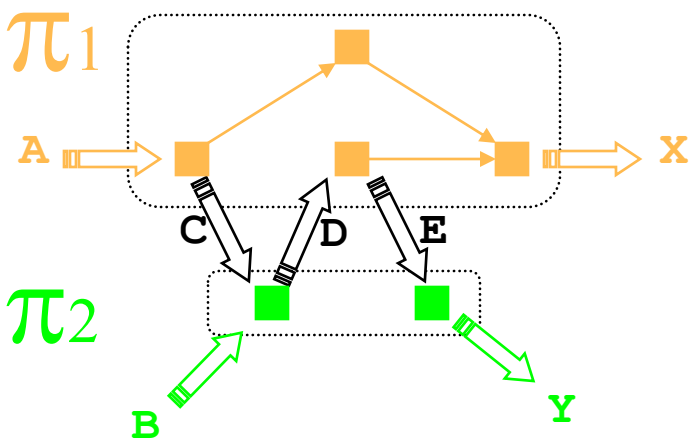
communication du contrôle

$C \equiv (| C_1$

- extraction des programmes distribués

P_i

Un exemple :



Méthodologie d'implémentation parallèle

compilation

$P \equiv (| C | Q |)$

répartition des données

$Q \equiv (| Q_1 | \ldots | Q_n |)$

communication des données

$Q \equiv (| Q_1 | \ldots | Q_n |) | (| Qle_1 | \ldots | Qle_n |)$

répartition du contrôle

$C \equiv (| C_1 | \ldots | C_n |)$

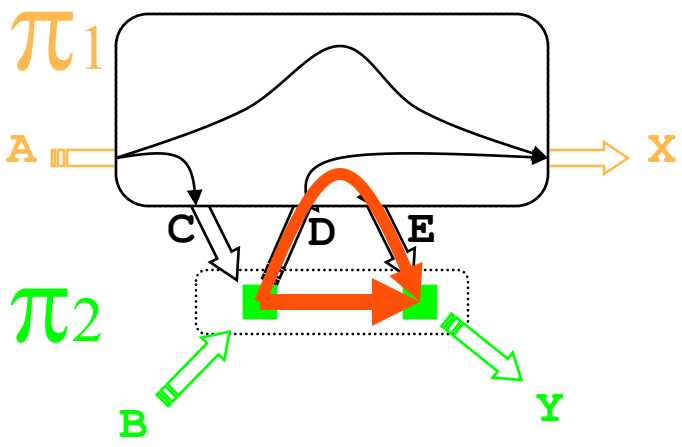
communication du contrôle

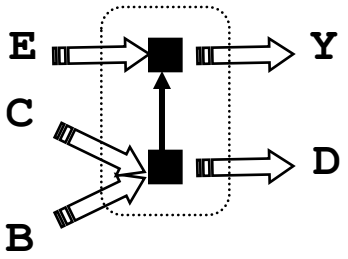
$C \equiv (| C_1$

- extraction des programmes distribués

P_i

Un exemple :





Méthodologie d'implémentation parallèle

compilation

$P \equiv (| C | Q |)$

répartition des données

$Q \equiv (| Q_1 | \ldots | Q_n |)$

communication des données

$Q \equiv (| Q_1 | \ldots | Q_n |) | (| Qle_1 | \ldots | Qle_n |)$

répartition du contrôle

$C \equiv (| C_1 | \ldots | C_n |)$

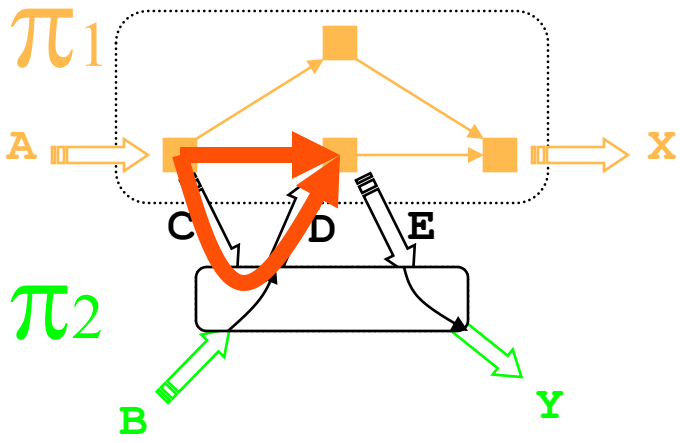
communication du contrôle

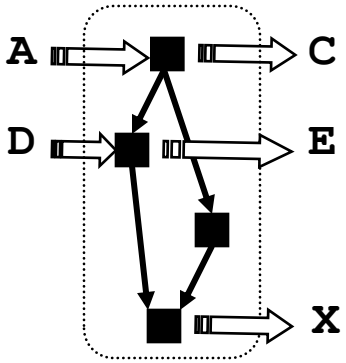
$C \equiv (| C_1$

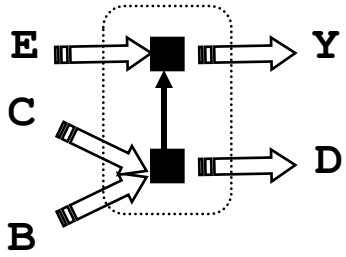
- extraction des programmes distribués

P_i

Un exemple :







Méthodologie d'implémentation parallèle

compilation

$$P \equiv (| C \mid Q |)$$

répartition des données

$$Q \equiv (| Q_1 \mid \ldots \mid Q_n |)$$

communication des données

$$Q \equiv (| Q_1 \mid \ldots \mid Q_n |) \mid (| Qle_1 \mid \ldots \mid Qle_n |)$$

répartition du contrôle

$$C \equiv (| C_1 \mid \ldots \mid C_n |)$$

communication du contrôle

$$C \equiv (| C_1 \mid \ldots \mid C_n |) \mid (| Cle_1 \mid \ldots \mid Cle_n |)$$

extraction des programmes distribués

$$P_i := (| Q_i \mid Qle_i \mid C_i \mid Cle_i \mid Abs(P) |)$$

- Schéma d'exécution
  - dans chaque instant
  - entre les instants

Schéma d'exécution dans l'instant

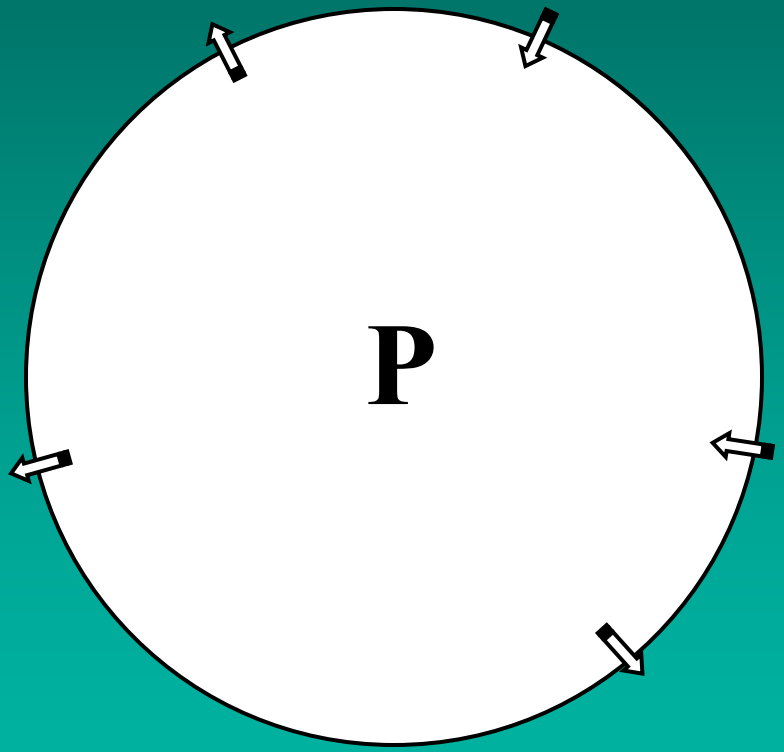
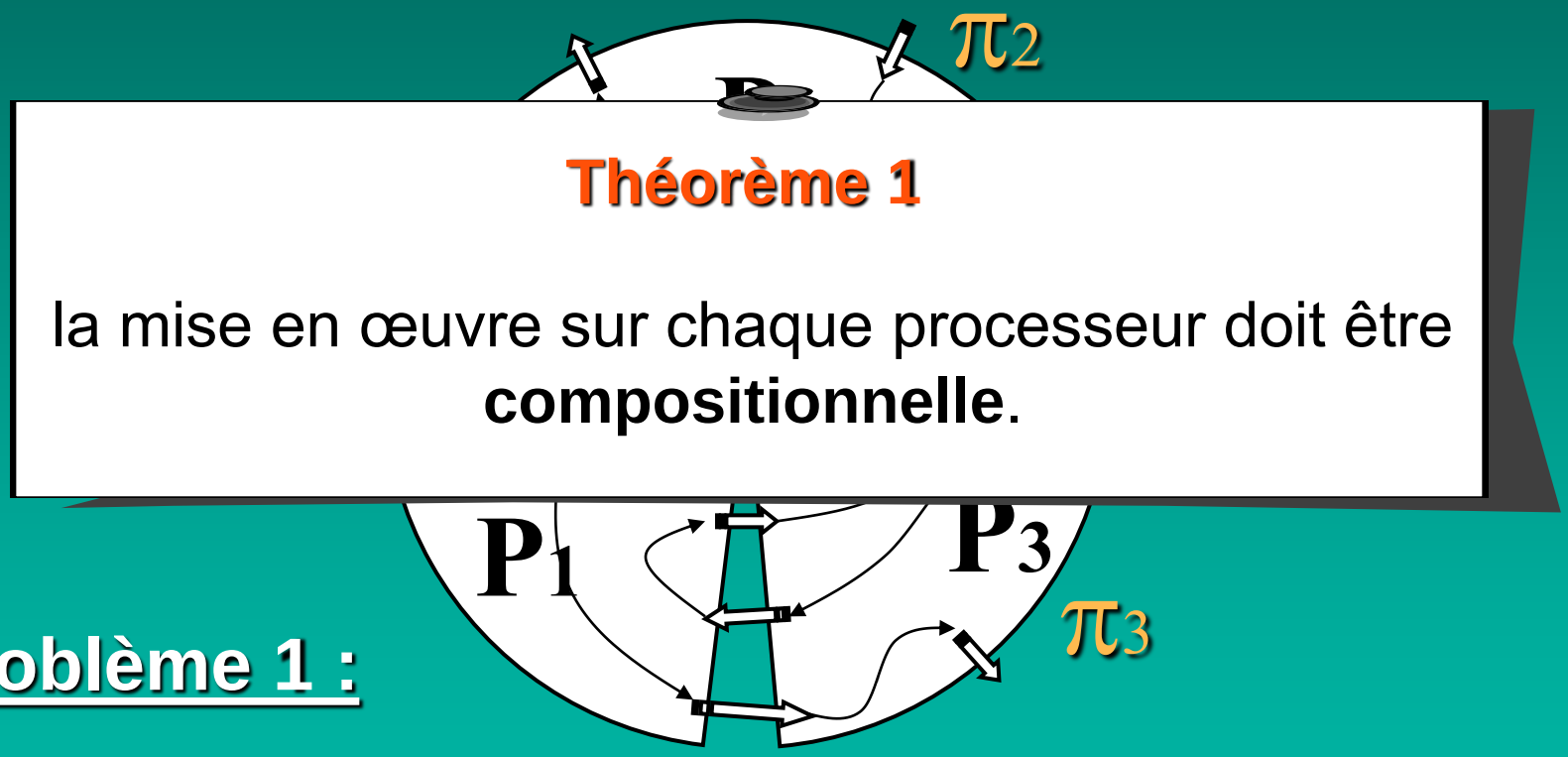


Schéma d'exécution dans l'instant

Il faut mettre en œuvre les processus **Pi** **sans introduire de cycle**.

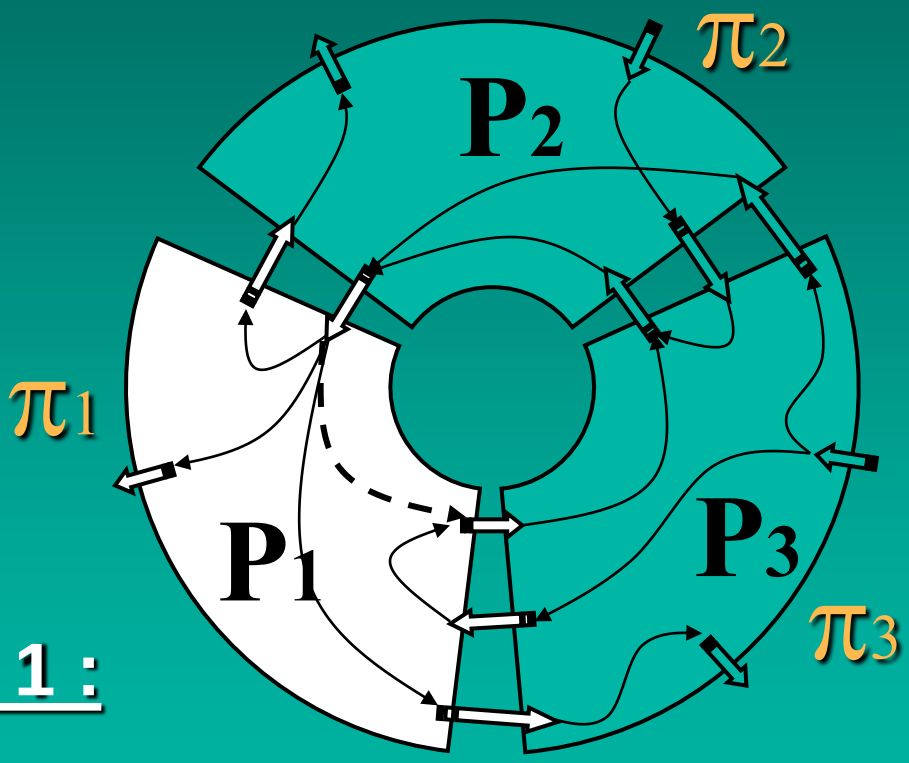


Problème 1 :

La mise en œuvre mono-processeur séquentielle est impossible car non compositionnelle.

Schéma d'exécution dans l'instant

Il faut mettre en œuvre les processus **Pi**
sans introduire de cycle.

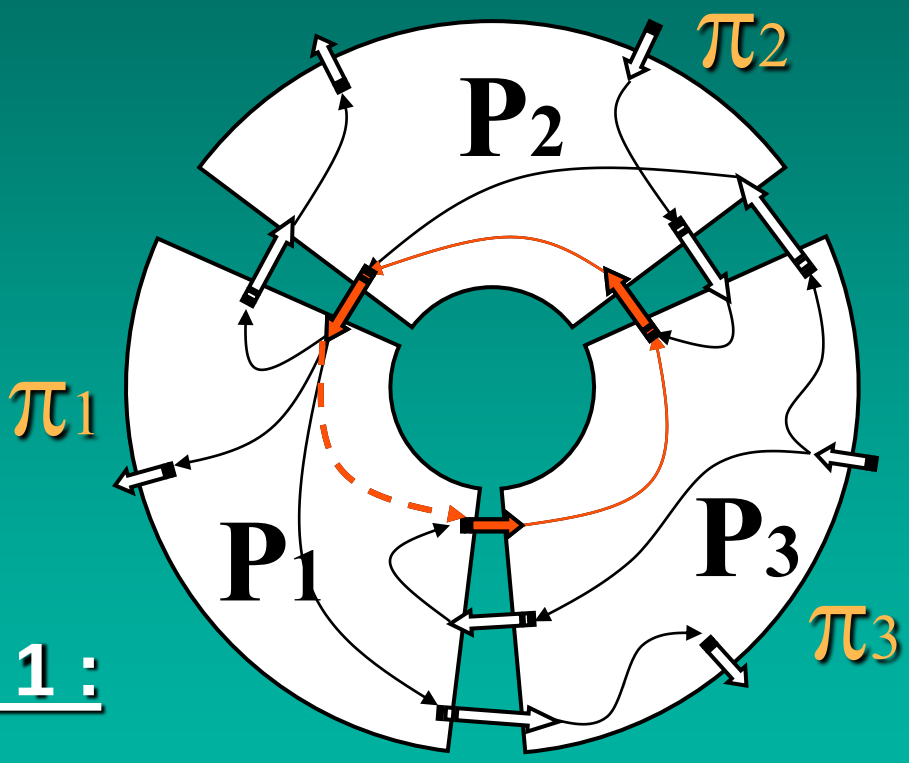


Problème 1 :

La mise en œuvre mono-processeur séquentielle est impossible car non compositionnelle.

Schéma d'exécution dans l'instant

Il faut mettre en œuvre les processus **Pi**
sans introduire de cycle.

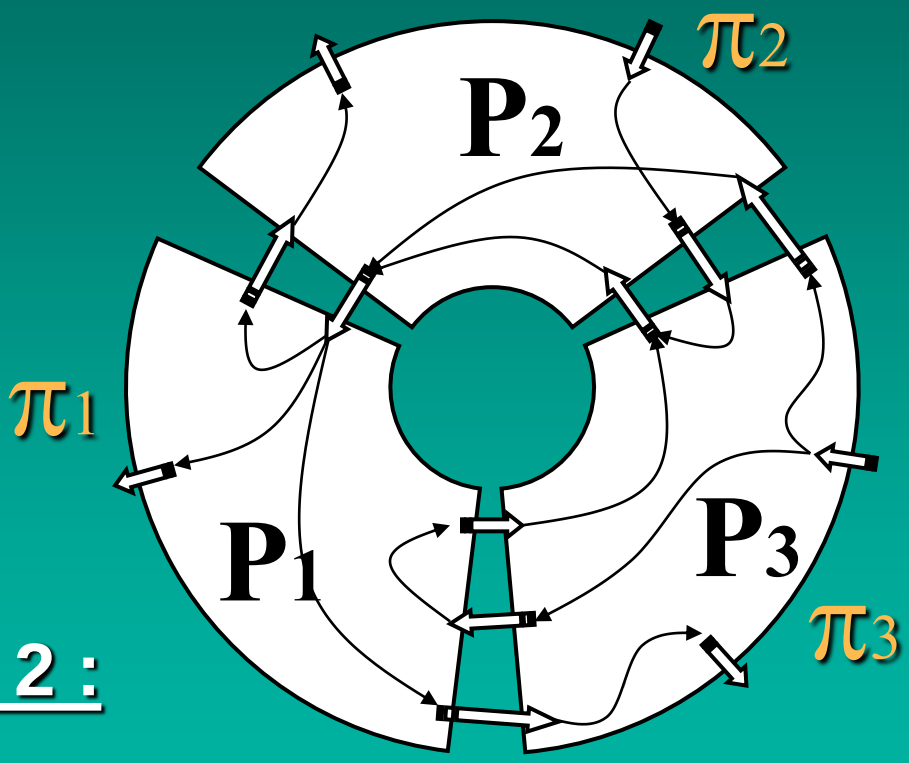


Problème 1 :

La mise en œuvre mono-processeur séquentielle est impossible car non compositionnelle.

Schéma d'exécution dans l'instant

Il faut mettre en œuvre les processus **Pi** **sans introduire de cycle.**



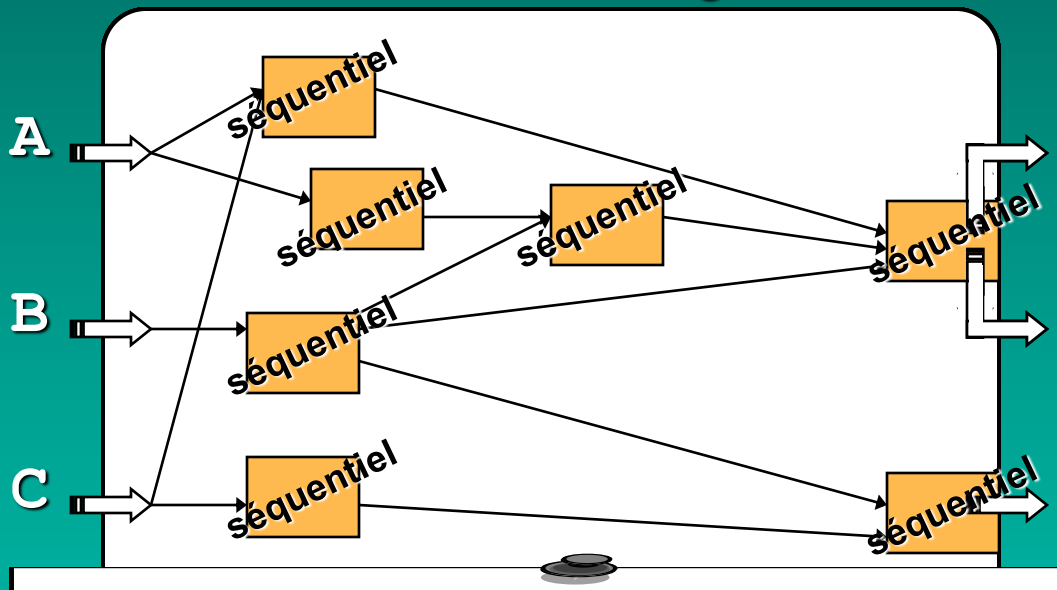
Problème 2 :

Les mises en œuvre compositionnelles sont très coûteuses (ordonnancement dynamique).

Schéma d'exécution dans l'instant

Une solution : **les lignées.**

21 nœuds, 34 arcs : 7 lignées, 12 arcs



Théorème 2

l'ordonnancement total des lignées n'affecte pas la compositionnalité.

Schéma d'exécution entre instants

non synchronisé

Sur chaque processeur :

Initialiser
répéter
exécuter l'instant courant

Schéma d'exécution entre instants

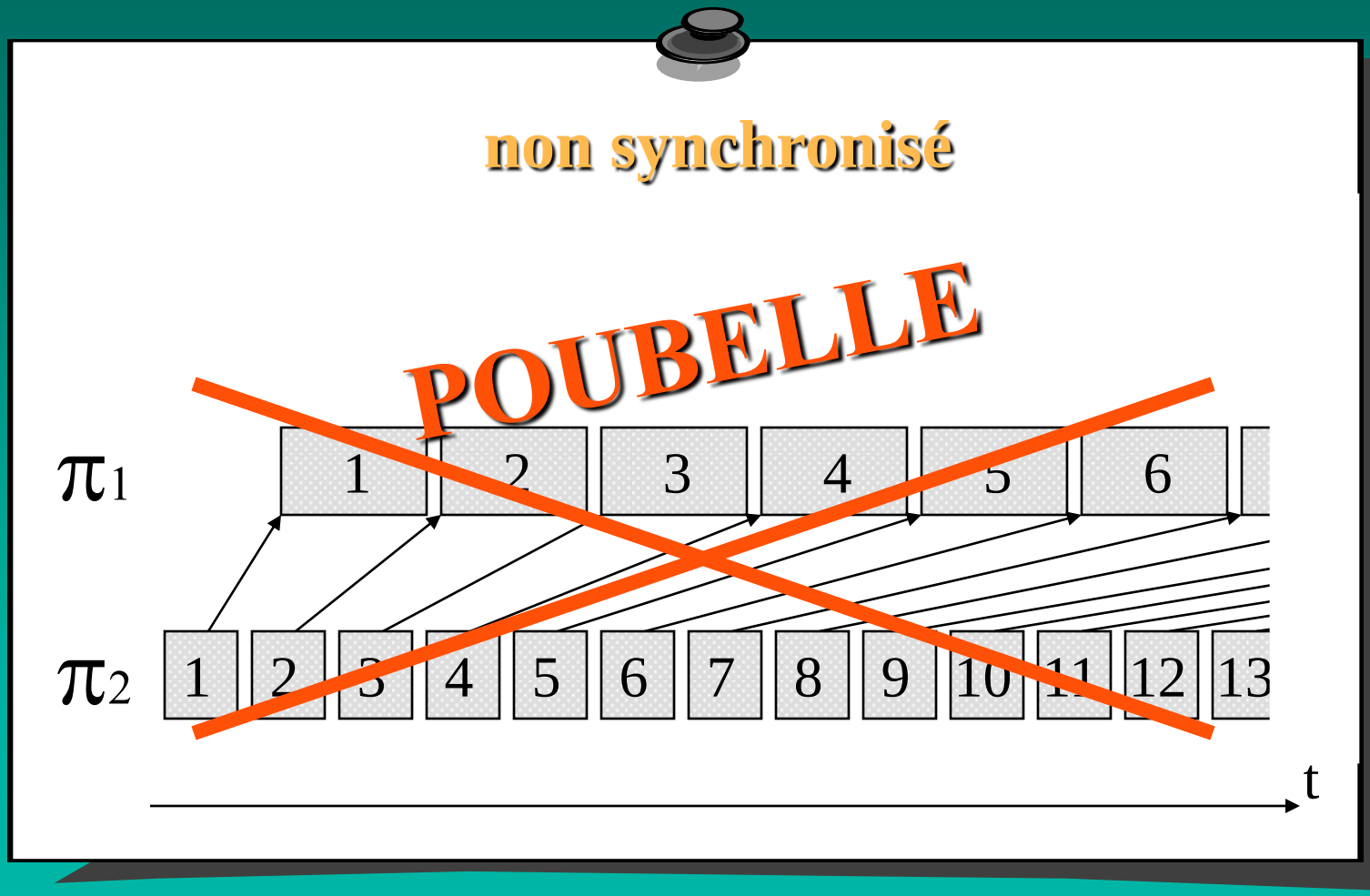


Schéma d'exécution entre instants

fortement synchronisé

Sur chaque processeur :

Initialiser
répéter
exécuter l'instant courant
signaler la fin de l'instant courant
attendre la fin de l'instant courant ailleurs

Schéma d'exécution entre instants

fortement synchronisé

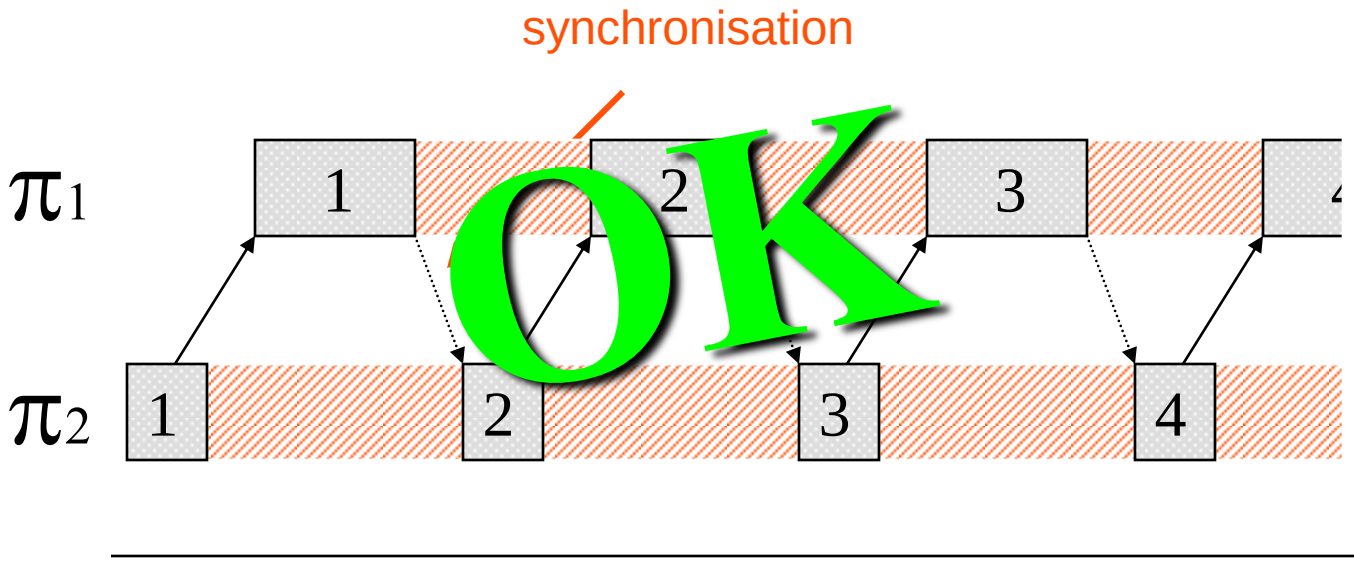


Schéma d'exécution entre instants

faiblement synchronisé

Sur chaque processeur :

Initialiser
n = 1
répéter
exécuter l'instant *n*
signaler la fin de l'instant *n*
attendre la fin de l'instant *n-1* ailleurs
n = n+1

Schéma d'exécution entre instants

faiblement synchronisé
ordre 1

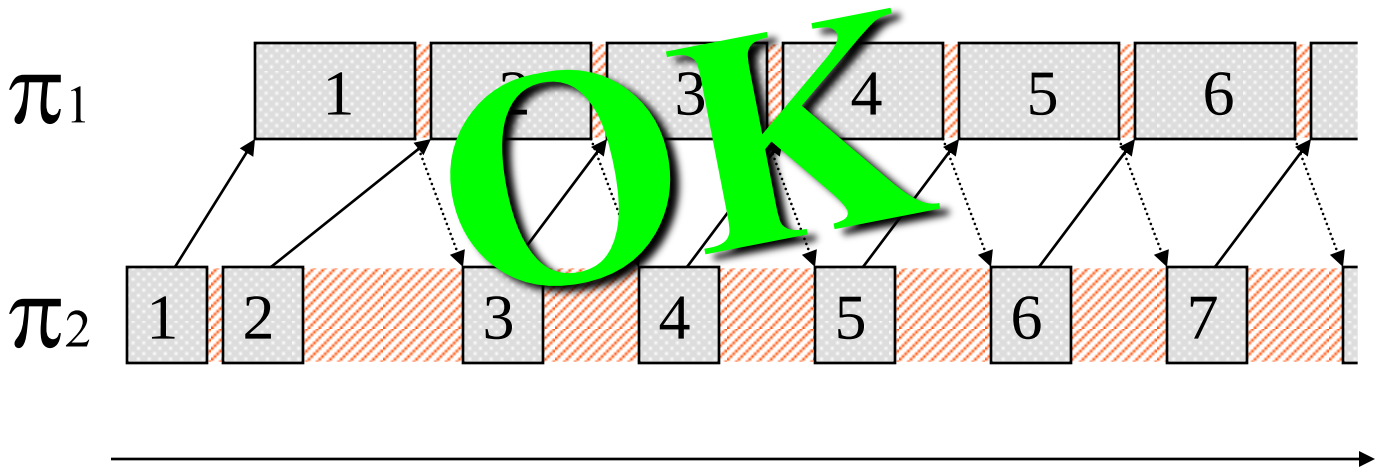


Schéma d'exécution entre instants

faiblement synchronisé
ordre 2

Moniteurs d'interface plus complexes

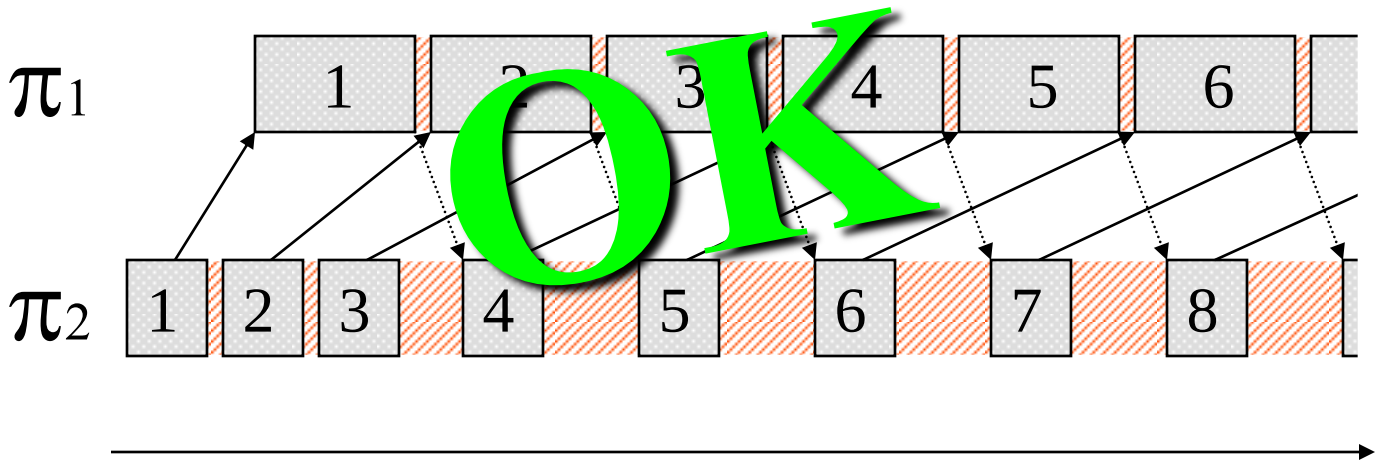


Schéma d'exécution entre instants

problème :

la synchronisation faible n'est valable que pour une petite classe de programmes.

⇦ pour les autres, il faut désynchroniser.

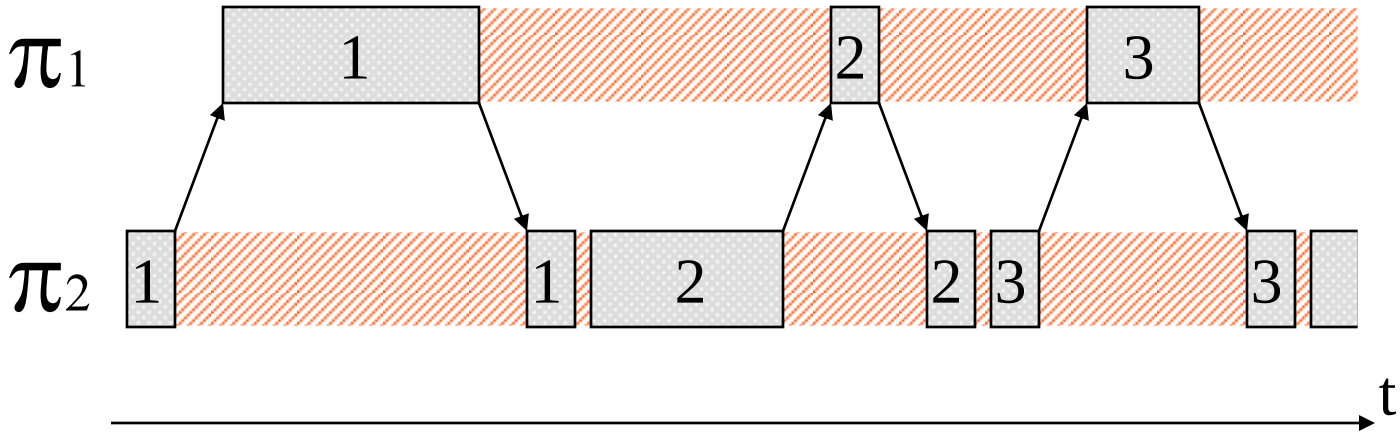


Schéma d'exécution entre instants

1-désynchronisation

Sur chaque processeur, on autorise l'exécution simultanée de 2 instants logiques.

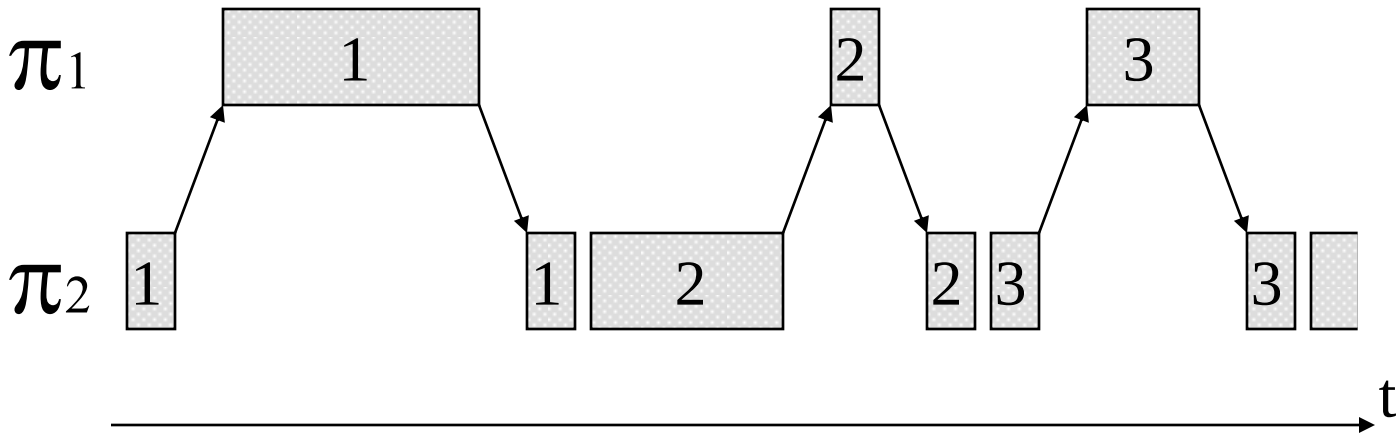


Schéma d'exécution entre instants

1-désynchronisation

Sur chaque processeur, on autorise l'exécution simultanée de 2 instants logiques.

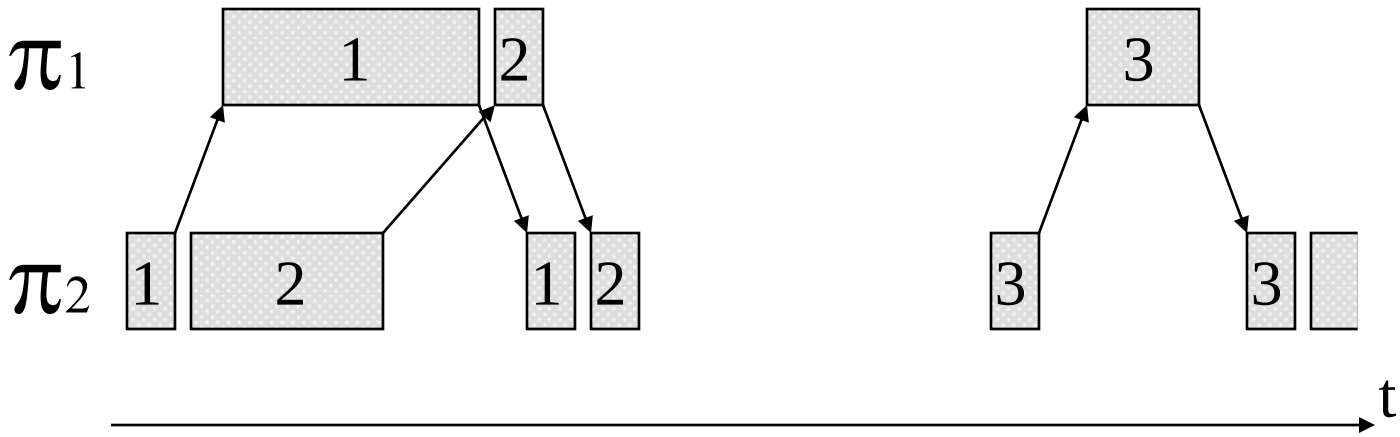


Schéma d'exécution entre instants

1-désynchronisation

Sur chaque processeur, on autorise l'exécution simultanée de 2 instants logiques.

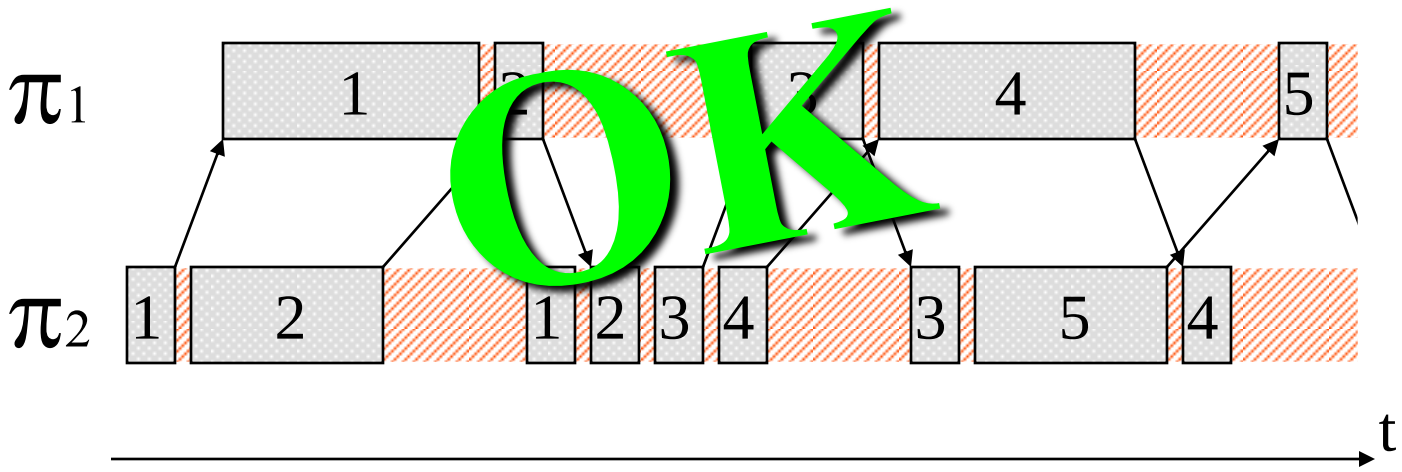


Schéma d'exécution entre instants

2-désynchronisation

Sur chaque processeur, on autorise l'exécution simultanée de 3 instants logiques.

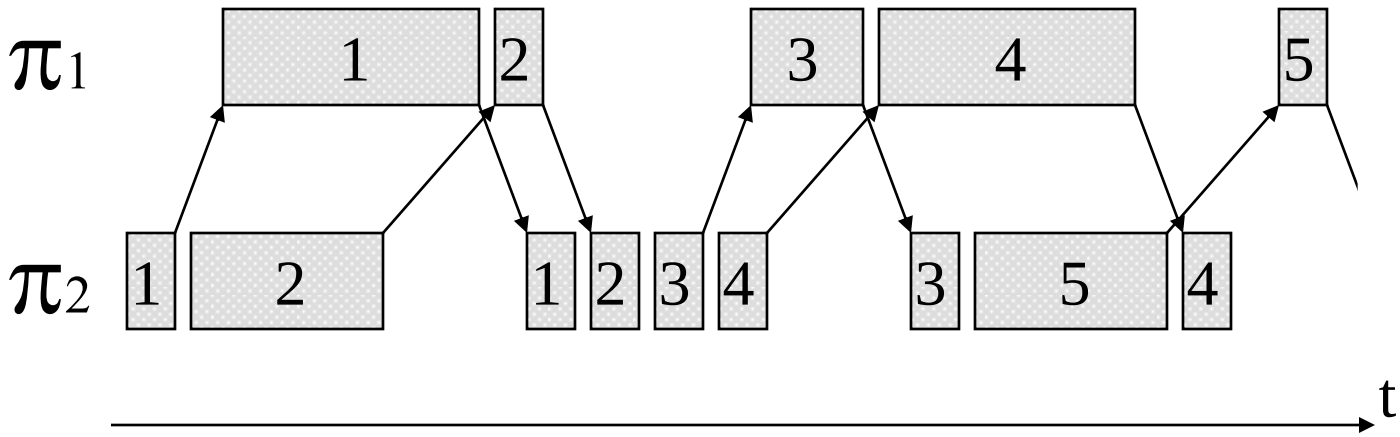


Schéma d'exécution entre instants

2-désynchronisation

Sur chaque processeur, on autorise l'exécution simultanée de 3 instants logiques.

moniteurs d'interface plus complexes
ordonnanceurs dynamiques plus complexes
plus de mémoire nécessaire

Méthodologie d'implémentation parallèle

compilation

$P \equiv (| C | Q |)$

répartition des données générique

$Q \equiv (| Q_1 | ... | Q_n |)$

communication des données

$Q \equiv (| Q_1 | ... | Q_n |) | (| Qle_1 | \ldots | Qle_n |)$

répartition du contrôle générique

$C \equiv (| C_1 | ... | C_n |)$

communication du contrôle

$C \equiv (| C_1 | ... | C_n |) | (| Cle_1 | \ldots | Cle_n |)$

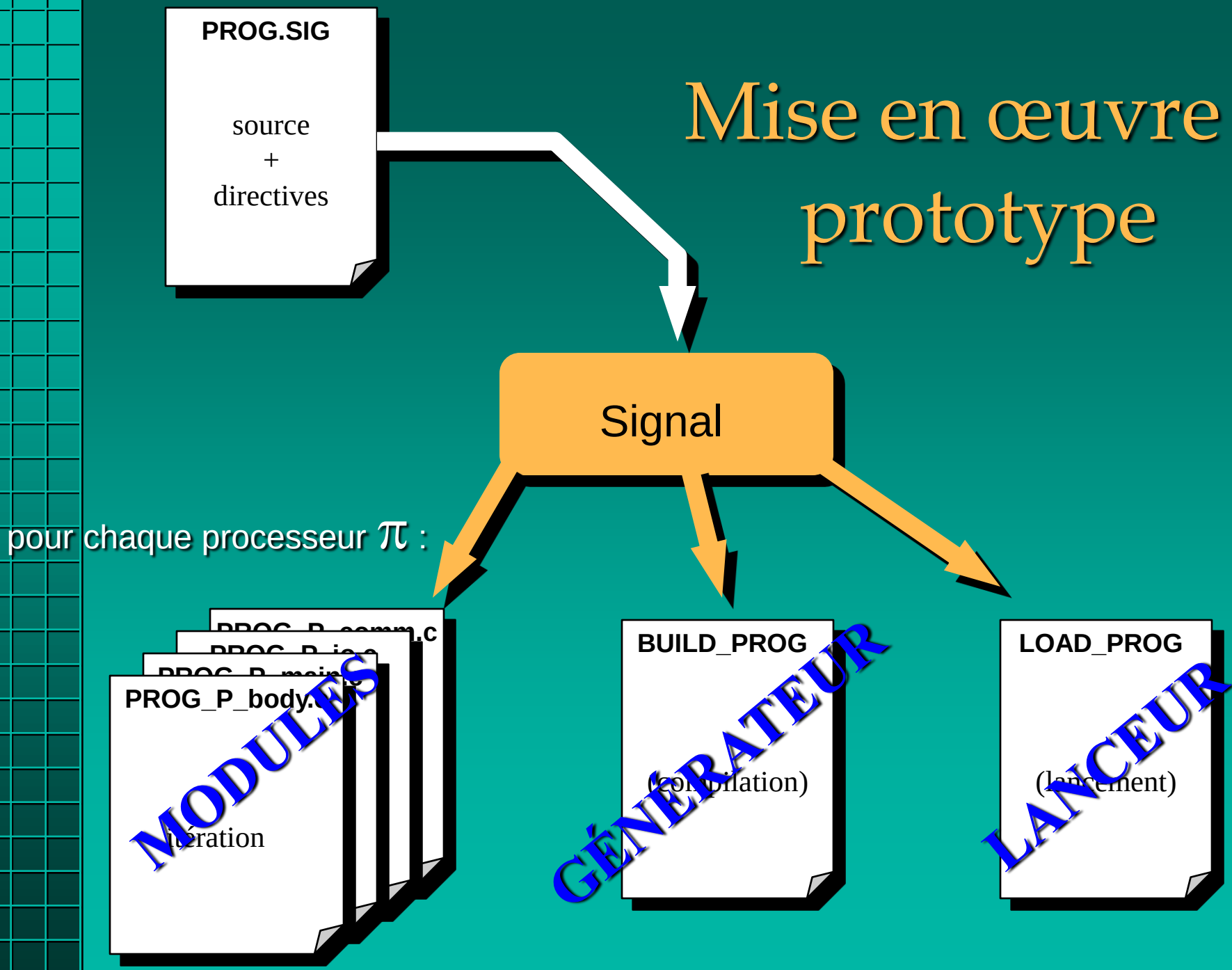
extraction des programmes distribués

$P_i := (| Q_i | Qle_i | C_i | Cle_i | Abs(P) |)$

application d'un schéma d'exécution

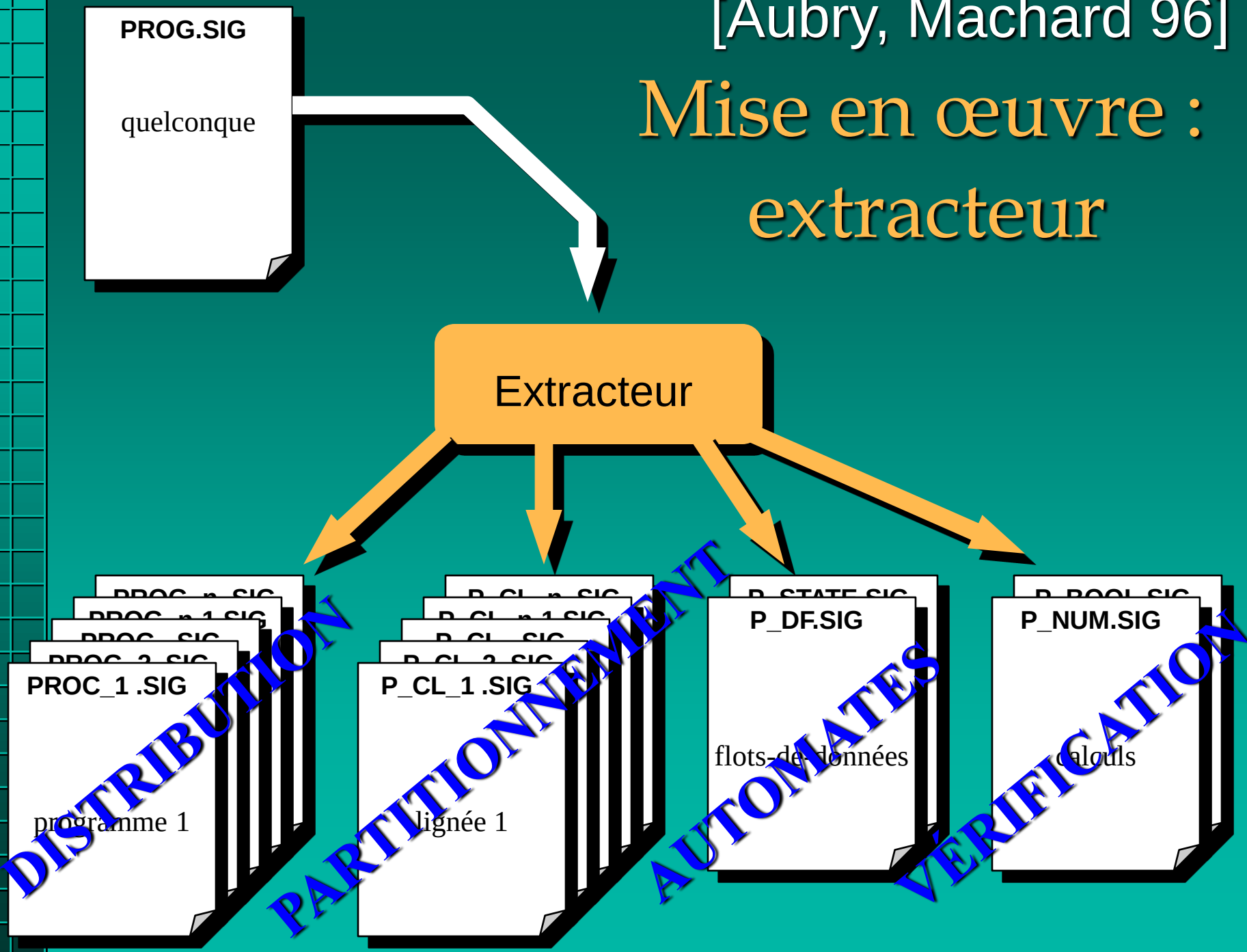
synchronisation, désynchronisation

Mise en œuvre : prototype



[Aubry, Machard 96]

Mise en œuvre : extracteur



Conclusion

- Modèle mathématique
  - dynamique (≠ instantané)
  - corrélation avec l'existant

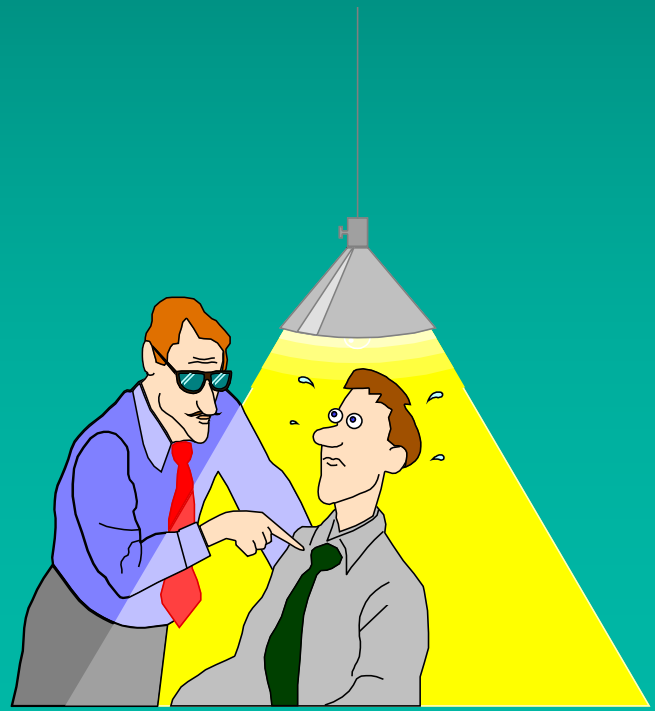
Conclusion

- Modèle mathématique
- Méthodologie
  - complète (du source au exécutifs)
  - modulaire

Conclusion

- Modèle mathématique
- Méthodologie
- Outils nouveaux
  - prototype de distribution
  - extracteur de graphe
  - intégrés dans l'environnement Signal

http://perso.univ-rennes1.fr/pascal.aubry

- Modèle mathématique
- Méthodologie
- Outils nouveaux

SI VOUS AVEZ DES QUESTIONS ...