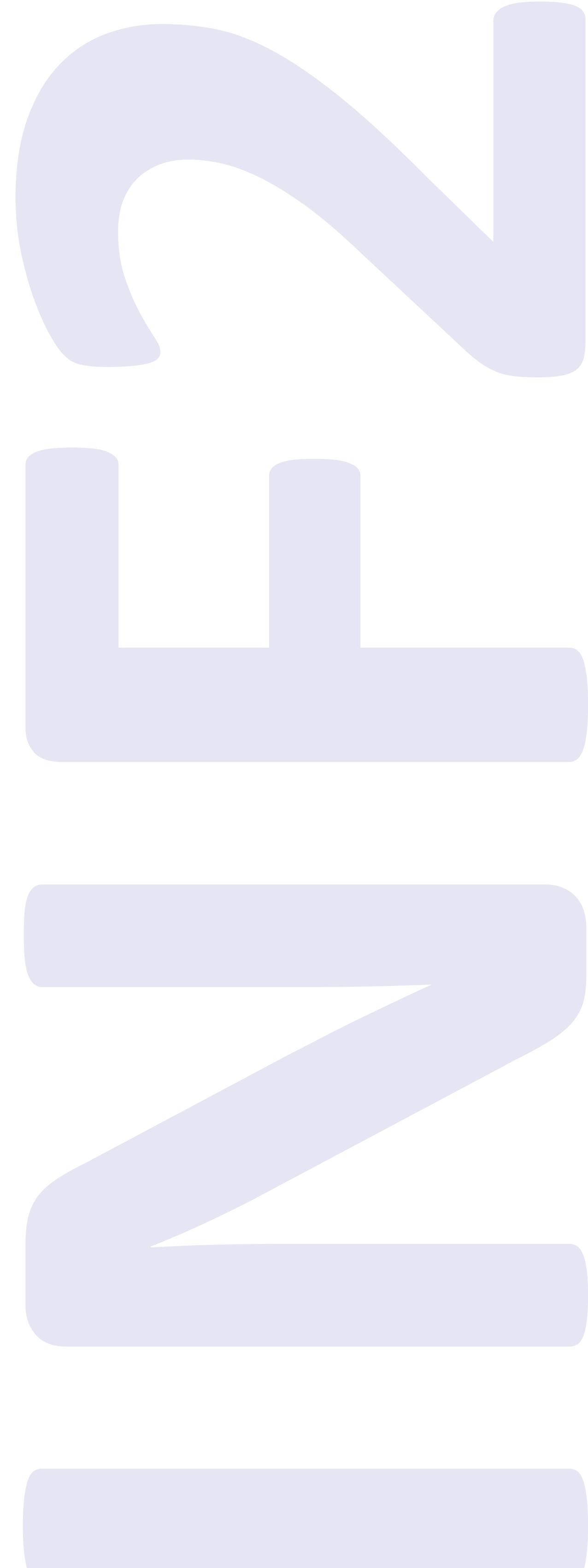


NF2

Principes des
Systèmes
Informatiques

la notion de système

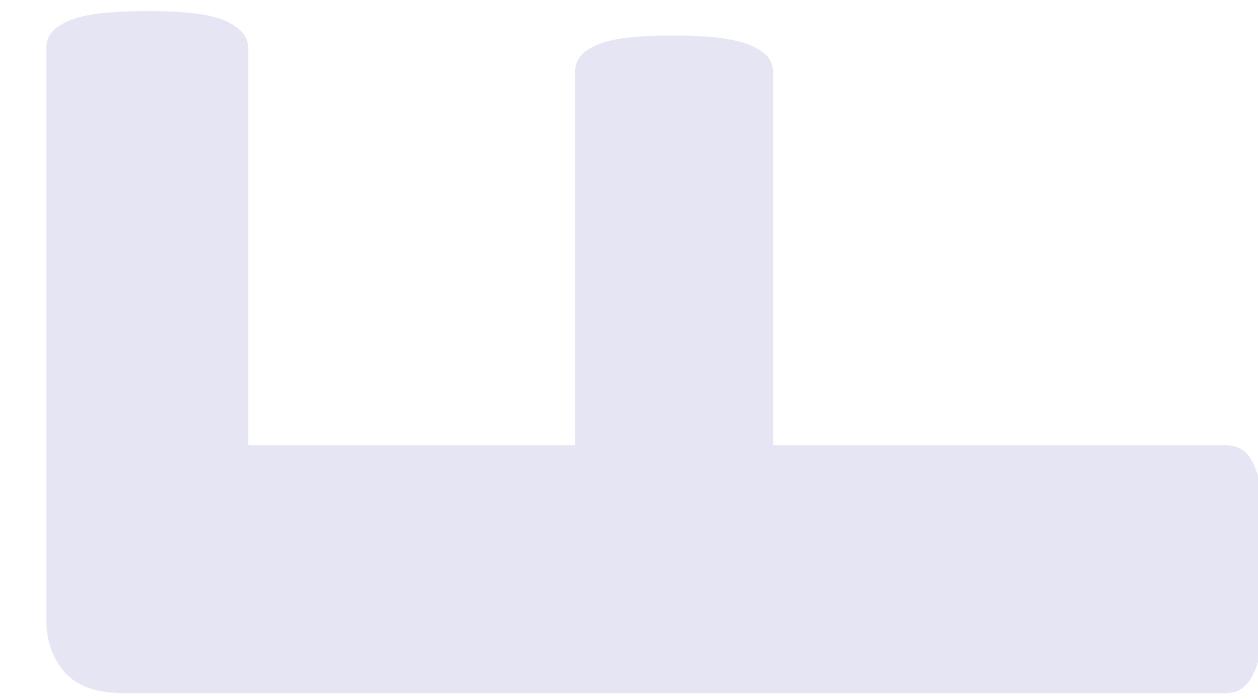
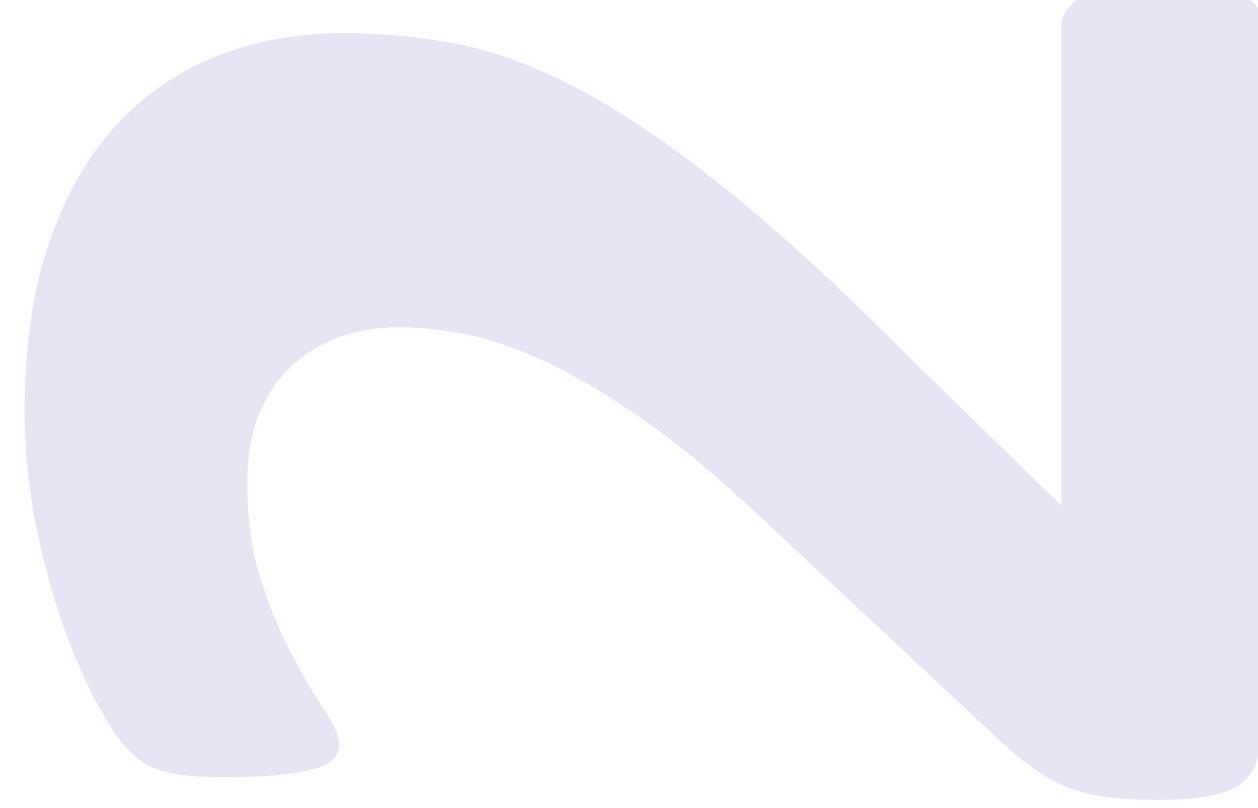
Olivier Ridoux



Plan

- La notion de système
- Systèmes informatiques

La notion de système



La notion de système

Des **entités** connectées par des **relations**

– **entités**

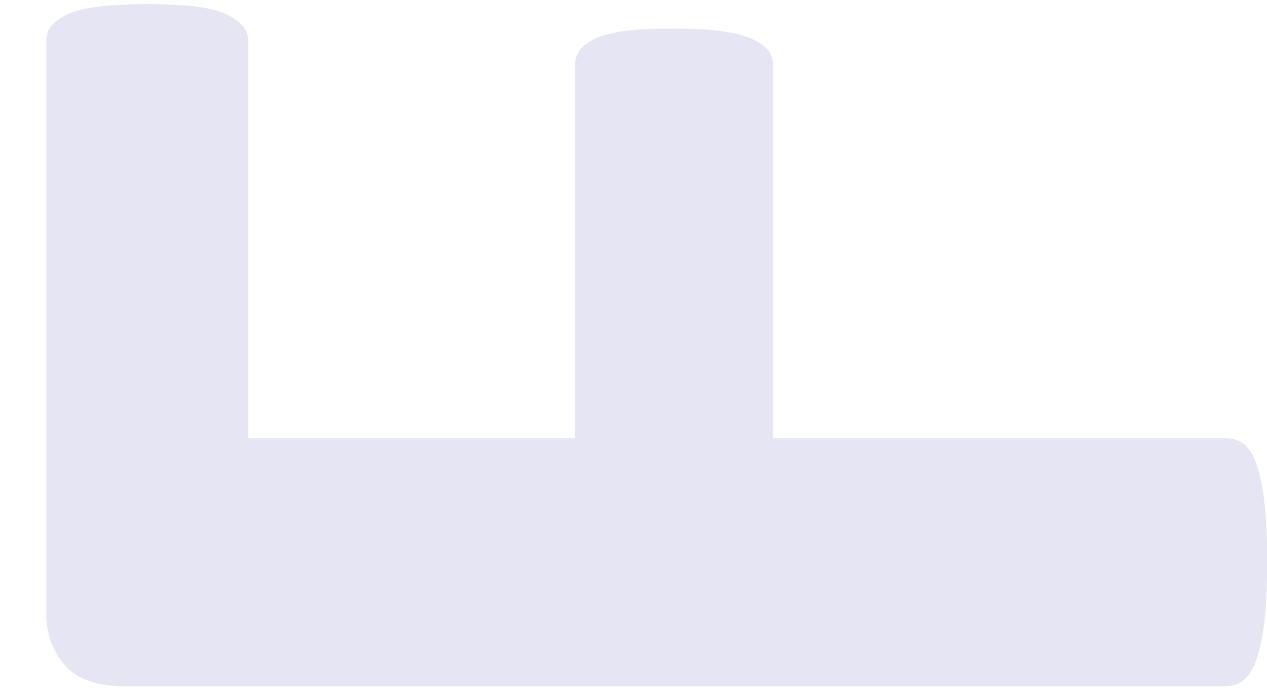
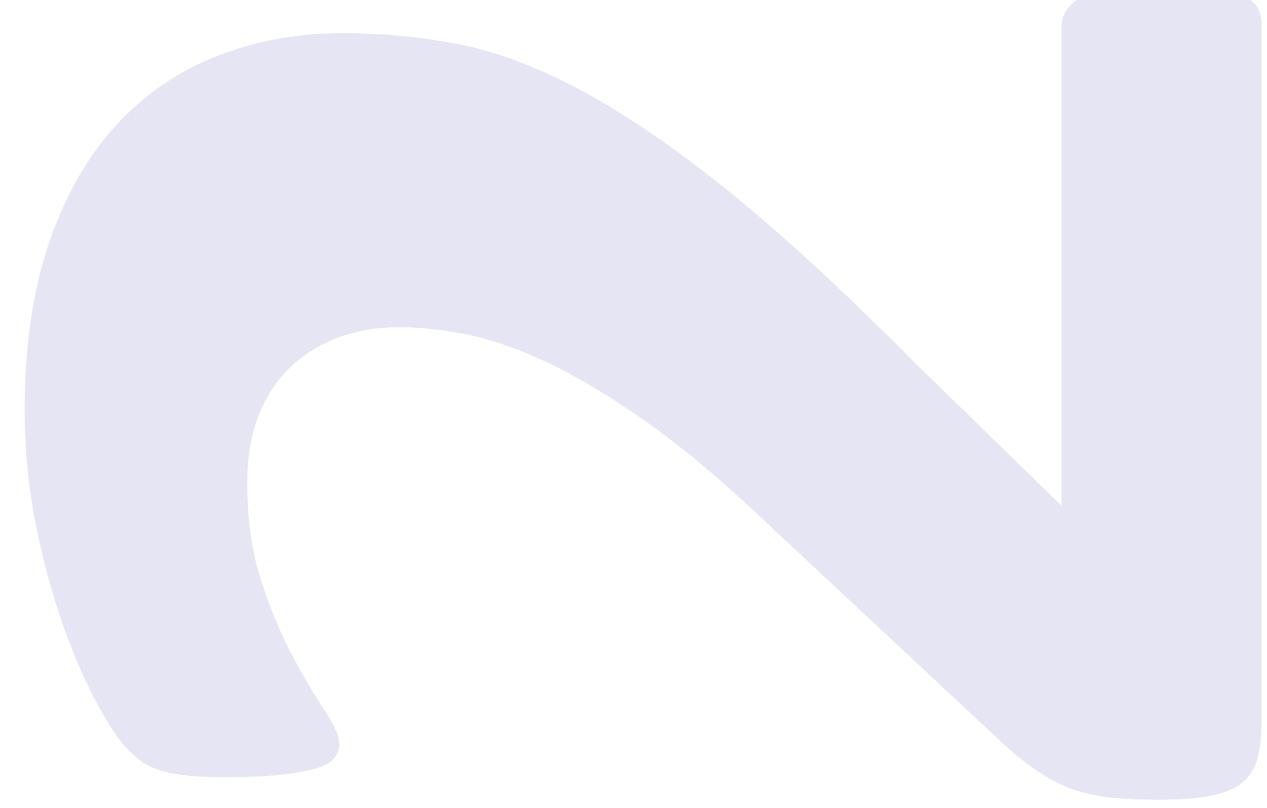
- planètes, cellules, individus

...

– **relations**

- attire, repousse, vend, achète, ...

...



Naturel ou artificiel

- Un système peut avoir un objet **naturel** ou **artificiel**
 - **naturel**
 - système solaire, système nerveux, cellule
 - ...
 - **artificiel**
 - système judiciaire, réseau routier
 - ...

Système vue de l'esprit

- Même **naturel** un système peut être une **vue de l'esprit** pour simplifier l'analyse
 - système Terre-Lune
 - effet du Soleil ?
 - système solaire
 - effet des autres étoiles de la galaxie ? Des autres galaxies ?
 - système judiciaire
 - rôle de la société, de l'économie, etc ?

Contour d'un système

- Un système peut avoir un **contour concret...**
 - membrane d'une cellule
- ...ou **abstrait**
 - limite d'attraction par le Soleil
- Être **dans ou hors** du système



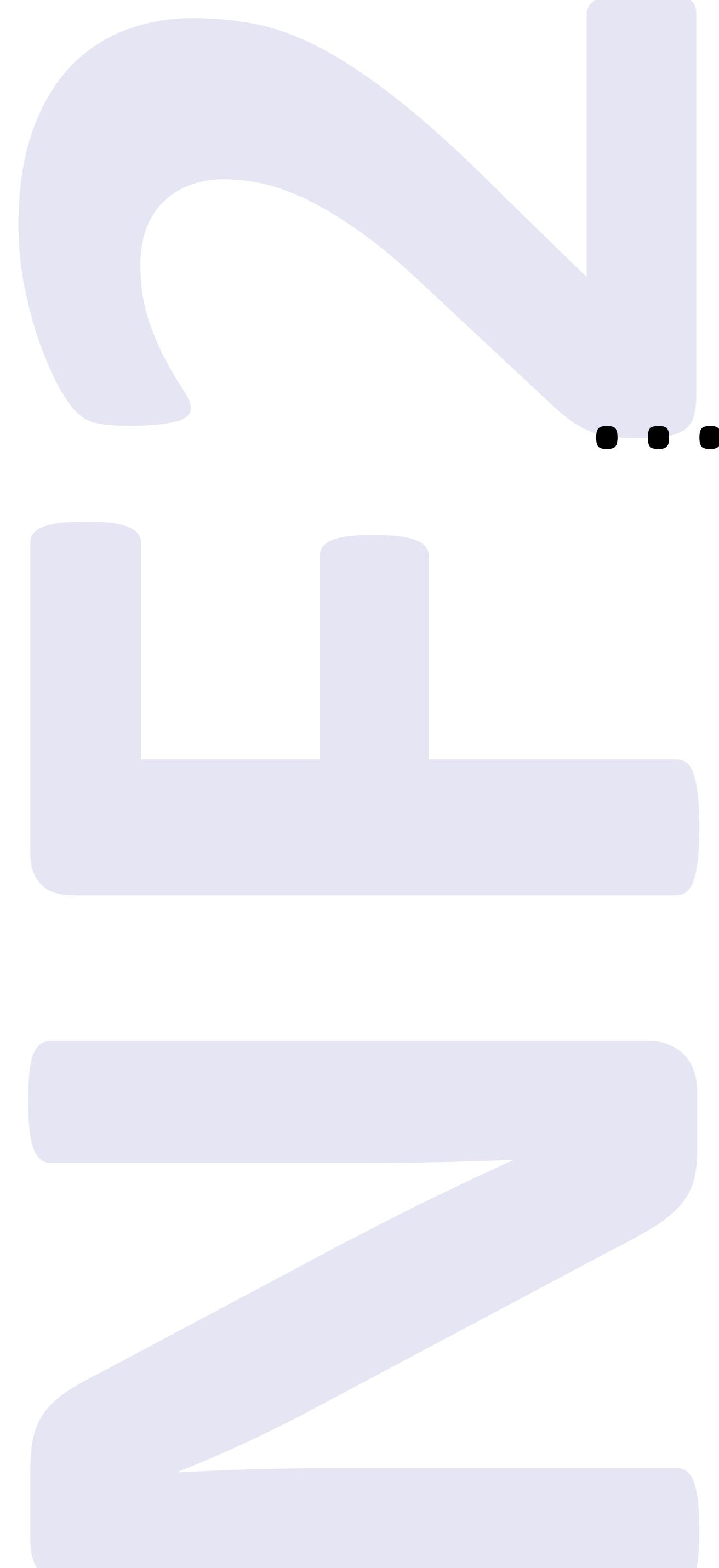
« Le tout est plus que la somme des parties »

Structure d'un système

- Un système peut être **composé** de sous-systèmes
- Une propriété peut **émerger** de cette composition
 - la cellule vivante / les protéines
 - la matière / les particules

Relations avec l'extérieur d'un système

- Un système peut être **ouvert**...
 - des relations avec son extérieur...
...qu'on ignore ou pas
- ...ou **fermé**
 - pas de relation avec son extérieur
 - ex. certains systèmes idéaux
de la thermodynamique



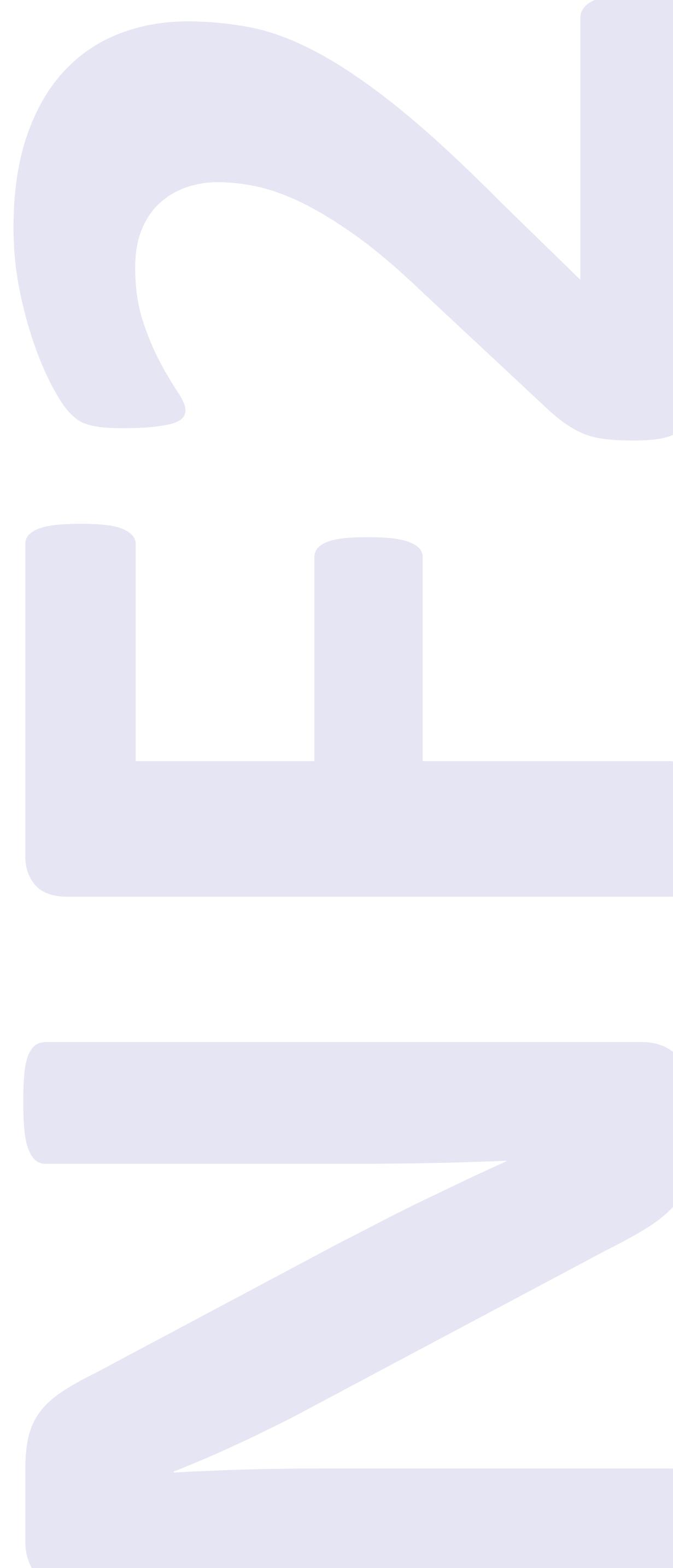
Vie d'un système

• Un système peut n'avoir qu'un régime **établi**...

- la cellule, le système Terre-Lune,
Internet pour les nuls

...ou avoir des régimes **transitoires**,
voire un **début** et une **fin**

- abiogenèse, cosmologie,
Internet pour l'administrateur réseau



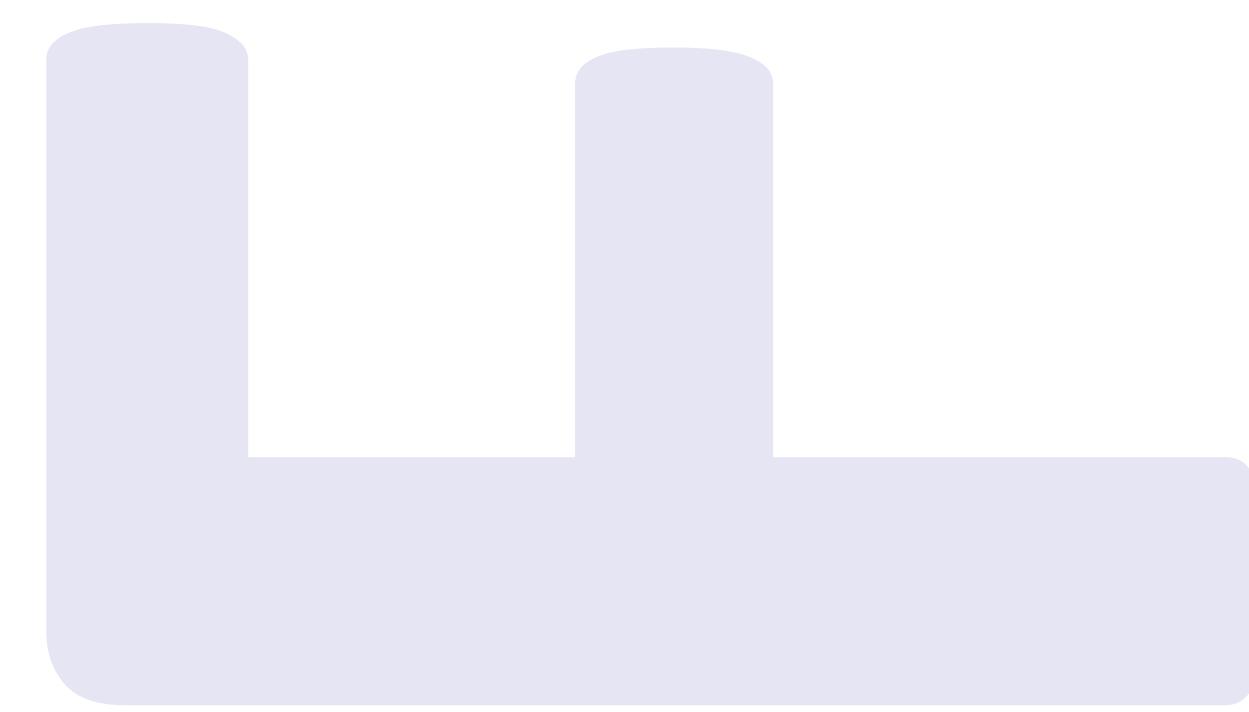
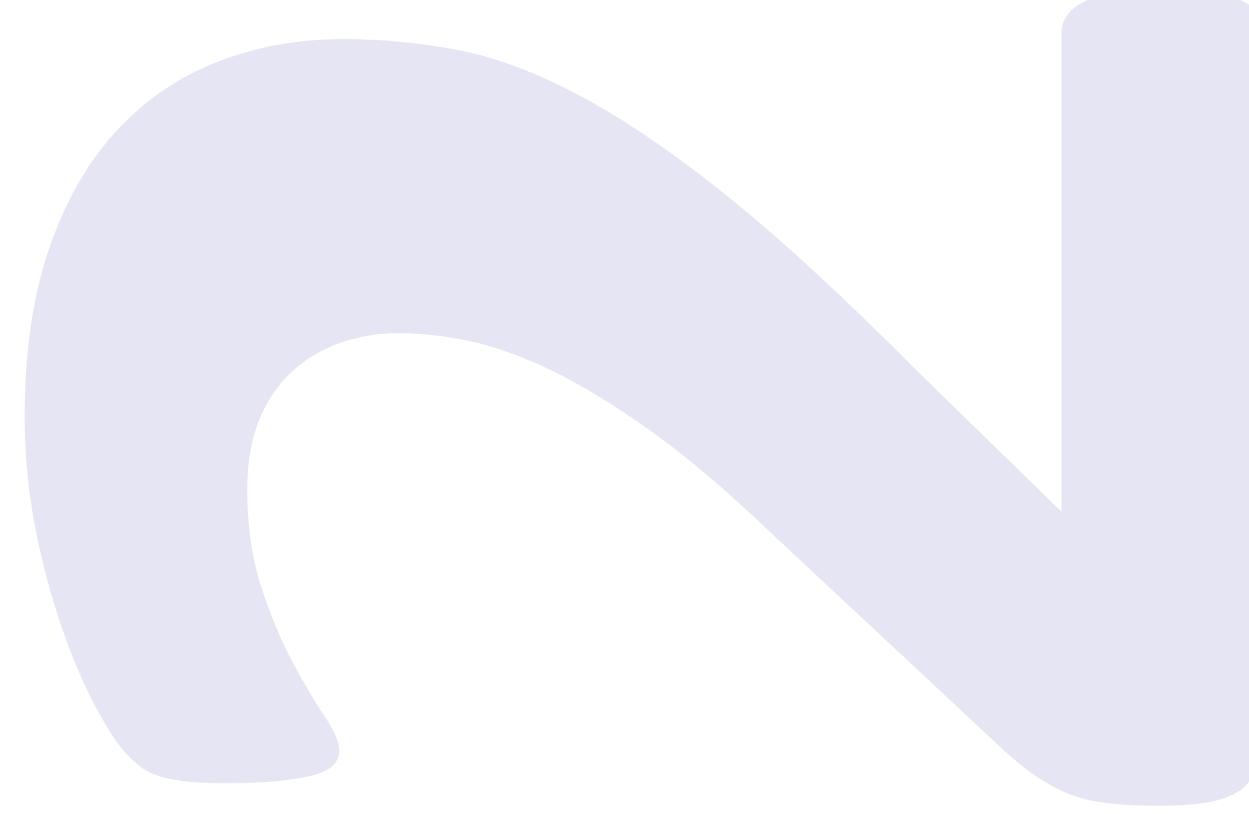
À retenir

- Système = entités + relations
- Système
 - naturel / artificiel / vue de l'esprit
 - concret / abstrait
 - composition
 - propriété émergente
 - ouvert / fermé
 - stable / transitoire

Systèmes informatiques

Les systèmes informatiques

- Des entités qui **calcurent** et qui **mémorisent** des **symboles**
- Des entités qui **communiquent** **des symboles** entre elles et avec l'extérieur

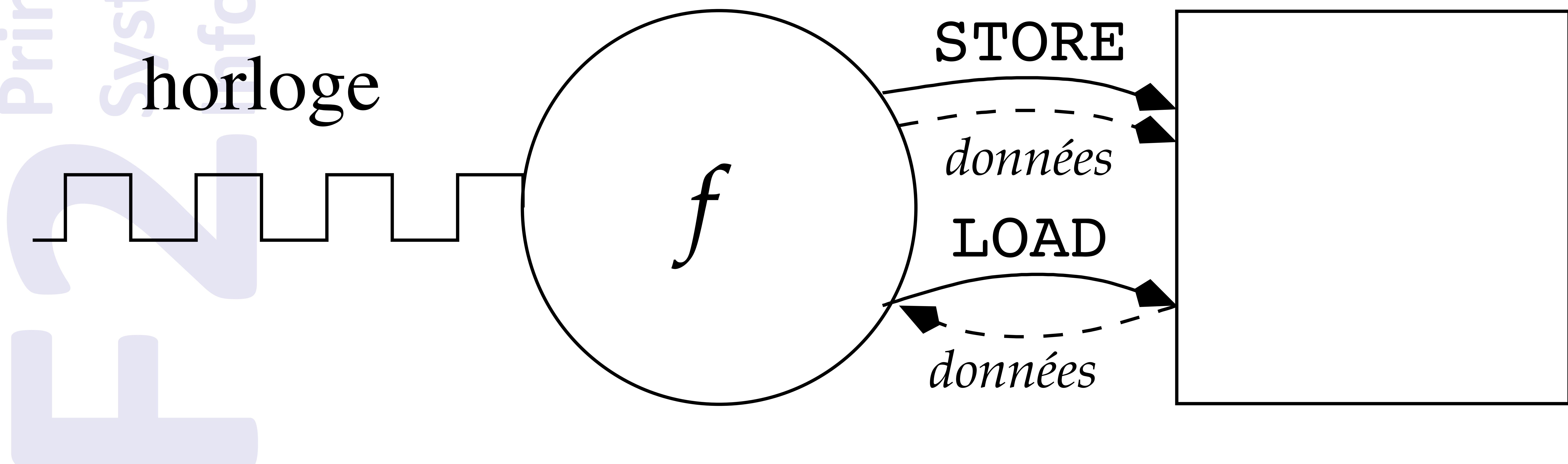


L'entité de calcul (1)

- Une **unité de calcul**
 - caractérisée par sa fonction f
- Une **mémoire**
 - pour écrire et lire des données
- Une **horloge**
 - pour synchroniser l'unité de calcul et la mémoire

L'entité de calcul (2)

unité de calcul mémoire



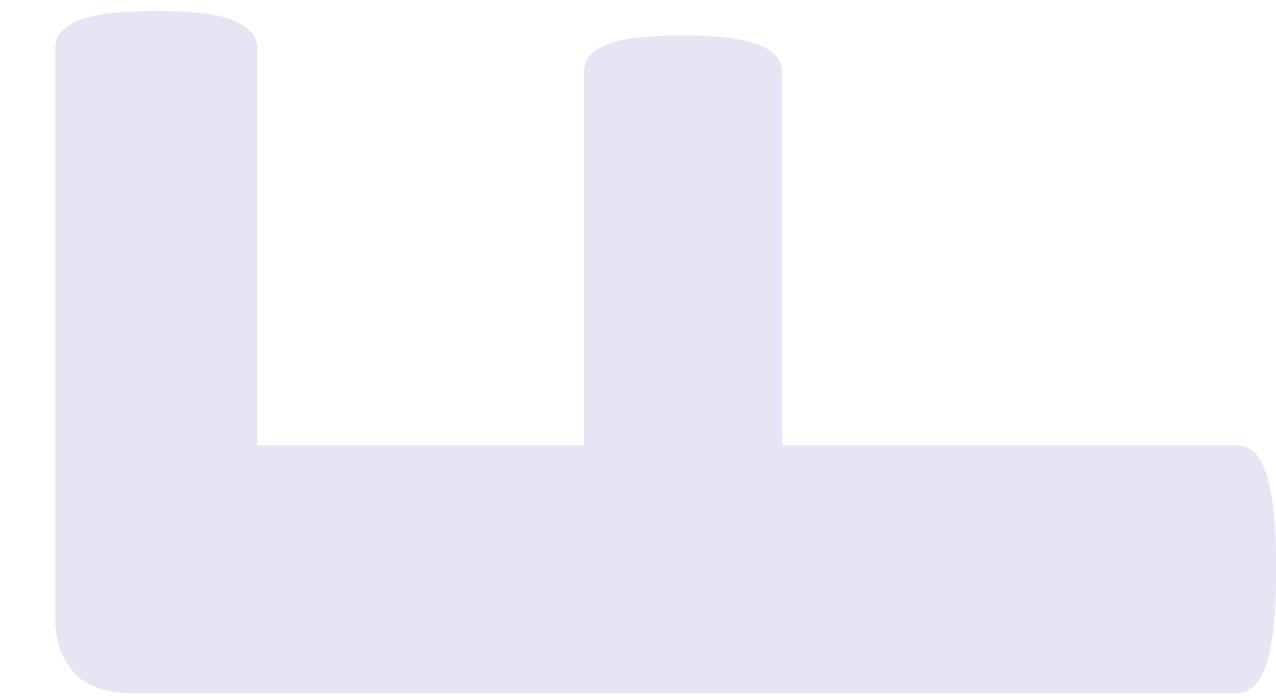
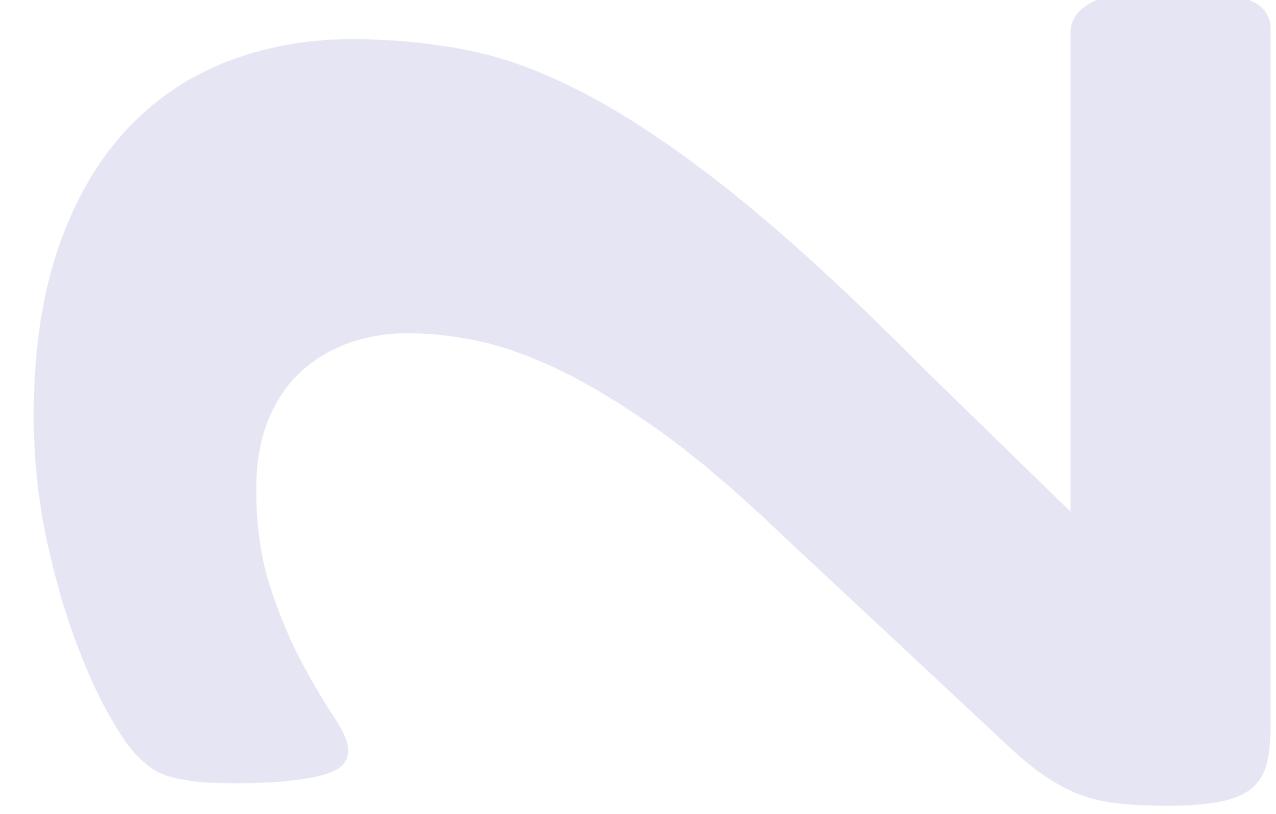
→ sens de la demande

→ sens de la propagation des données



L'entité de calcul (3)

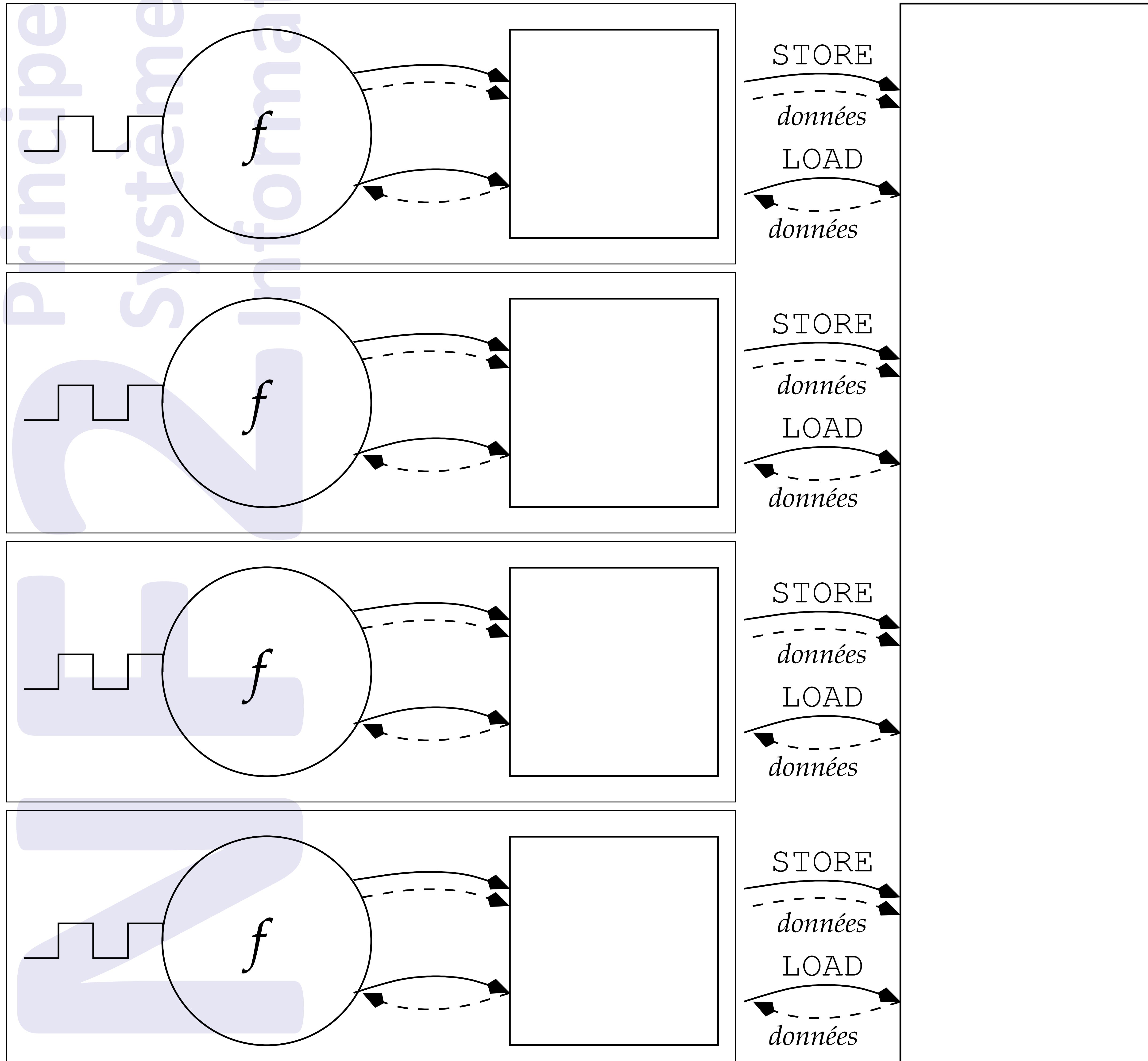
- L'unité de calcul itère une même fonction répéter
 - $m \leftarrow \text{STORE}(\ f(\ \text{LOAD}(m) \))$
- f est la fonction caractéristique de l'unité de calcul



L'entité de calcul (4)

- Modèle sommaire mais nombreuses variations possibles
 - multiplier les mémoires
 - multiplier les entités de calcul
 - Complexifier les mémoires ou les entités de calcul

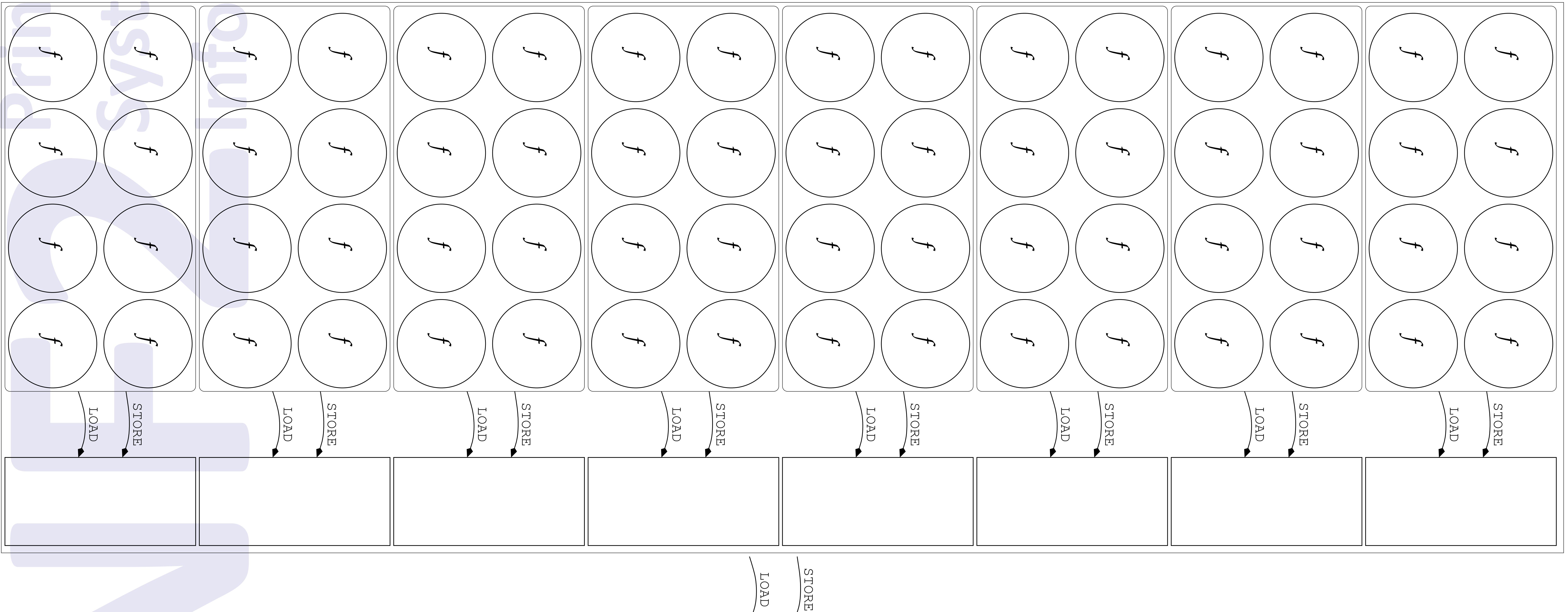
L'entité de calcul (5)



- Le processeur multi-cœurs

L'entité de calcul (6)

- Le processeur graphique (GPU)



Ce que calcule l'entité informatique (1)

- $c_0, f(c_0), f(f(c_0)), f(f(f(c_0))), f(f(f(f(c_0)))), \dots$

- Si tout va bien, on trouve

$$c = f(c) = f^n(c_0)$$

on dit que l'entité a calculé c à partir de c_0

- c est un **point-fixe** de f

- L'entité calcule les points-fixes de f

$f^*(i) = \text{point-fixe de } f \text{ pour input } i$



La boucle

Points fixes de

$$x \mapsto x + 1 \ ?$$

$$x \mapsto 2 \times x \ ?$$

$$x \mapsto x / 2 \ ?$$

- Il n'existe pas toujours de point-fixe
- Ils ne sont pas toujours accessibles

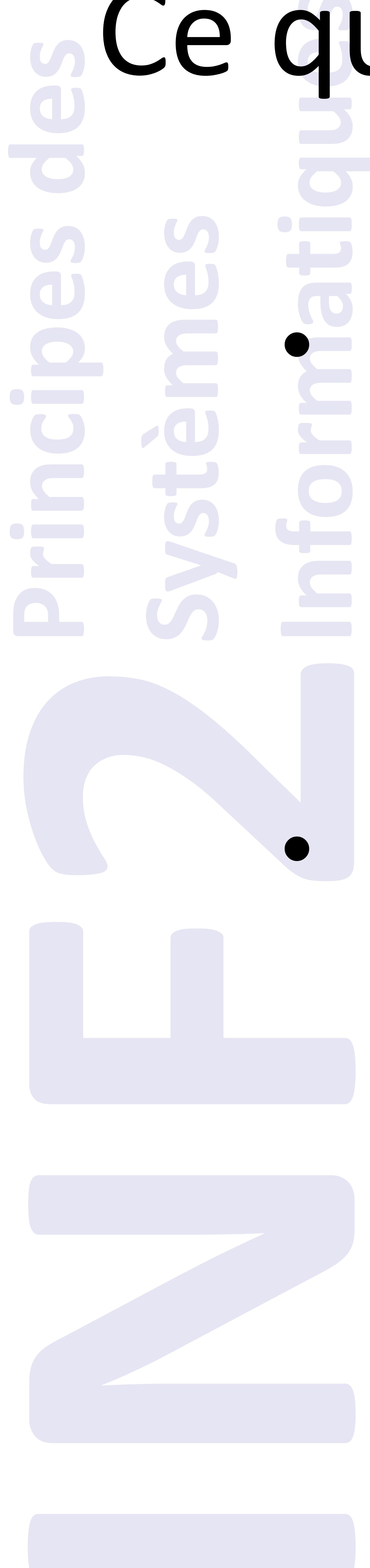
boucle !

Inévitable ! [Turing 1936]

Ce que calcule l'entité informatique (2)

- L'unité de calcul a une fonction caractéristique f **unique**
- L'entité calcule un f^* **qui dépend de l'input i**
- Glisser un programme dans l'input !

**Un ordinateur calcule
de multiples fonctions !**



Ce que calcule l'entité informatique (3)

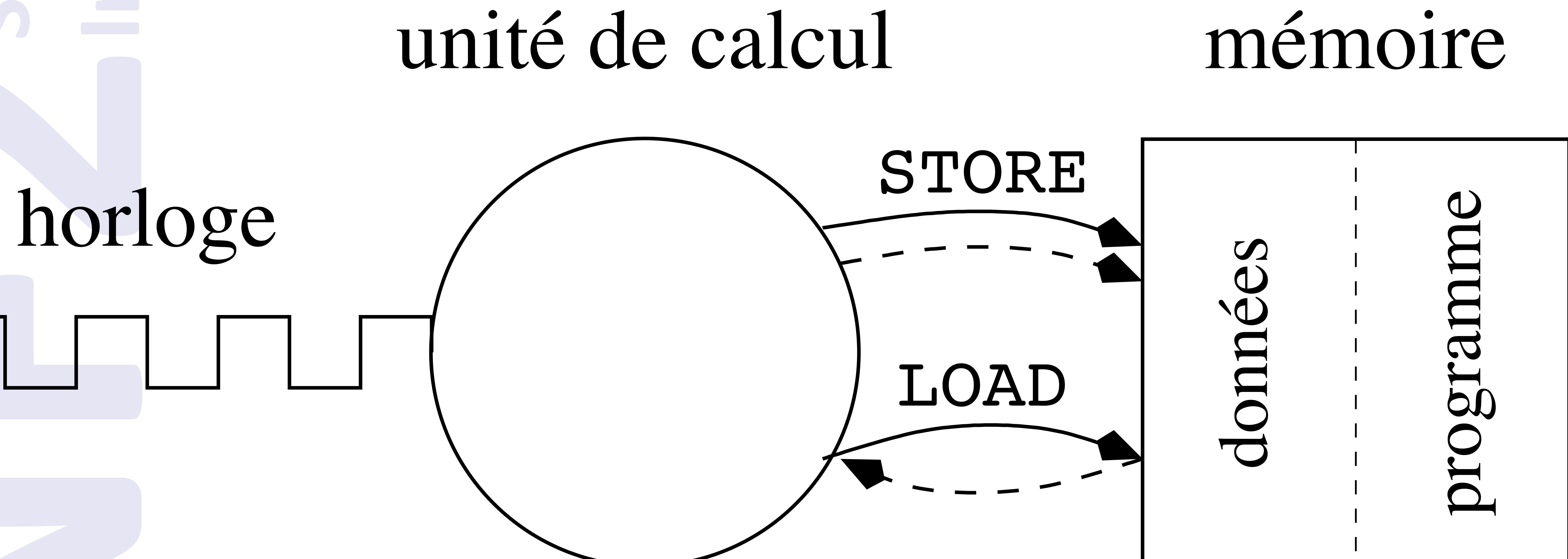
- Modèle abstrait de **Turing** (1936)
 - modèle mathématique du calcul
 - voir en L2 informatique
- Modèle concret de **von Neumann** (1945)

Les deux sont toujours d'actualité !

De très vieux principes !!!

Modèle de von Neumann (1)

Stocker un **programme en mémoire**



Modèle de von Neumann (2)

Stocker un **programme en mémoire**

répéter

données <

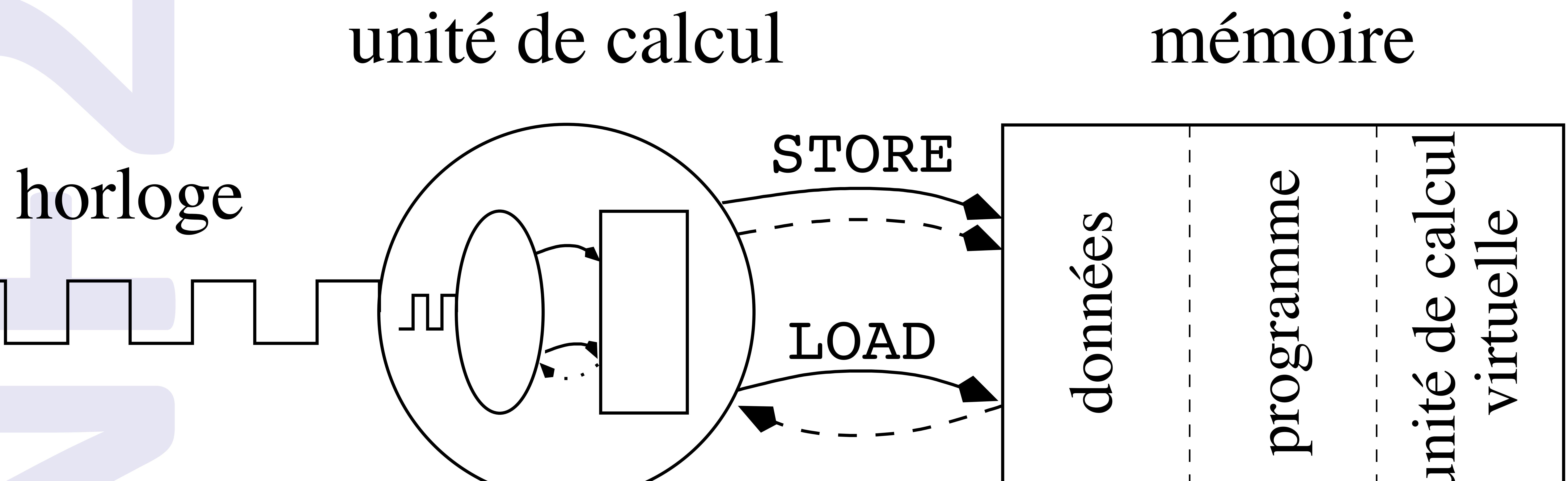
```
STORE( f( LOAD(programme),  
LOAD(données) ) )
```

- Le **calculateur programmable**

- flexibilité
- virtualisation
- portabilité

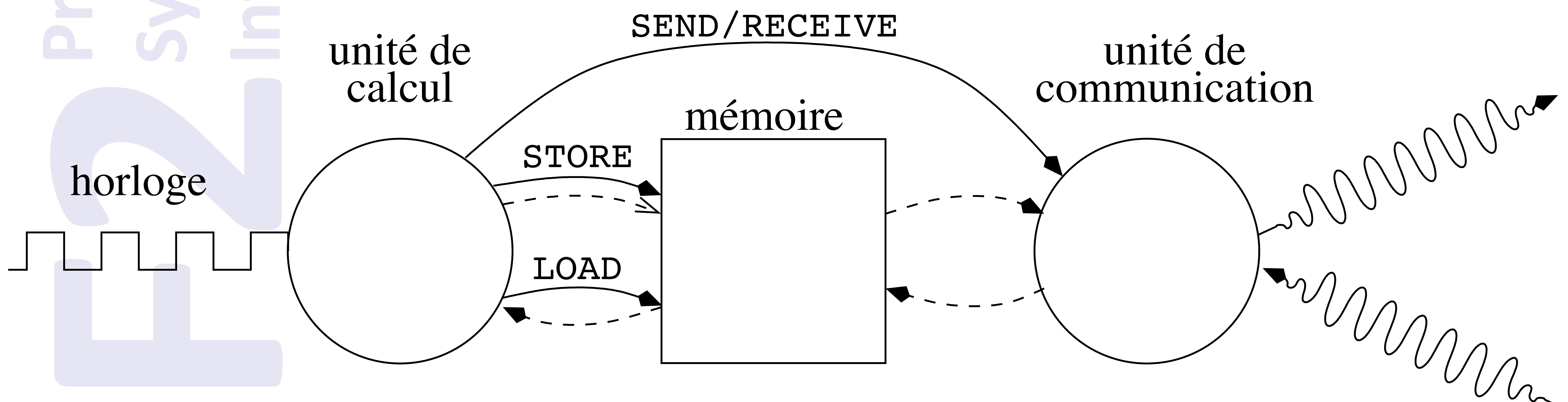
Modèle de von Neumann (3)

- # Virtualisation à tous les étages



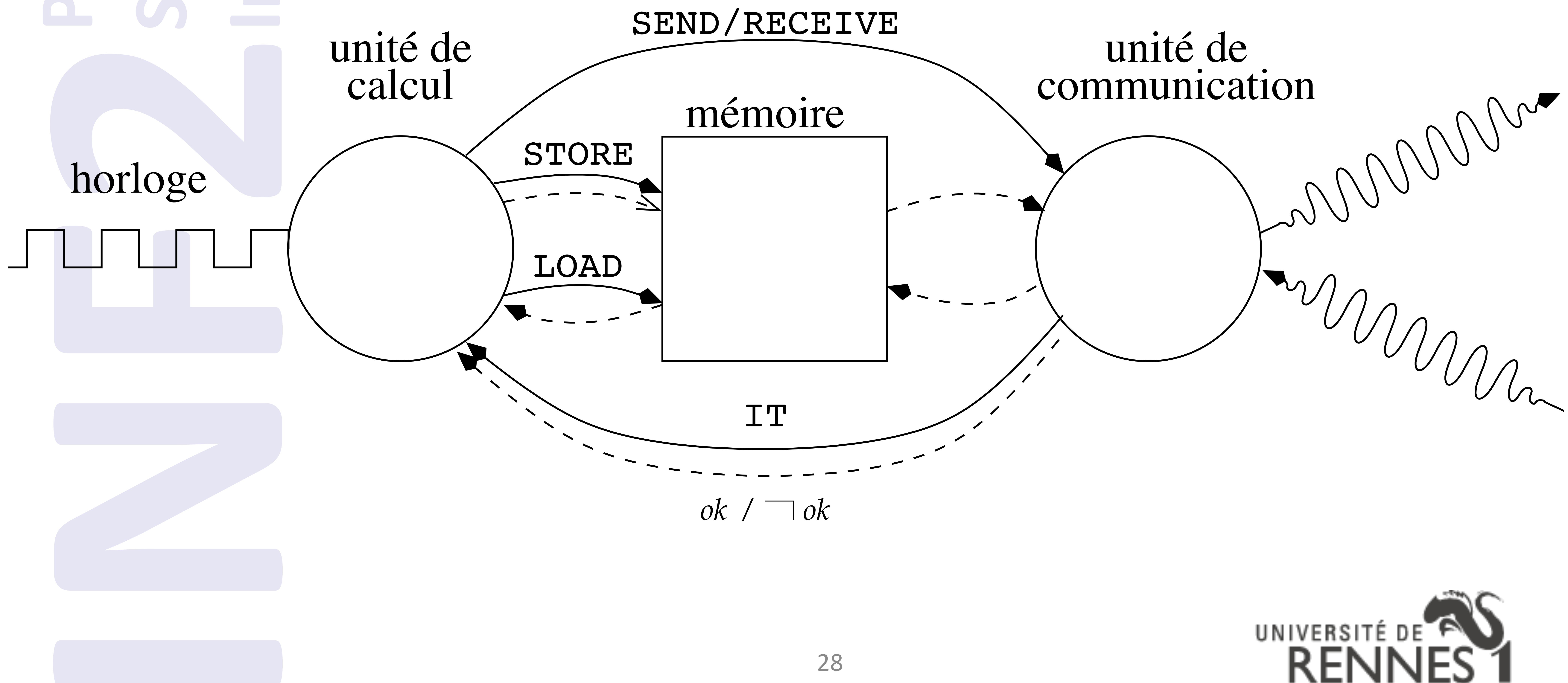
La relation de communication (1)

- Échange de message



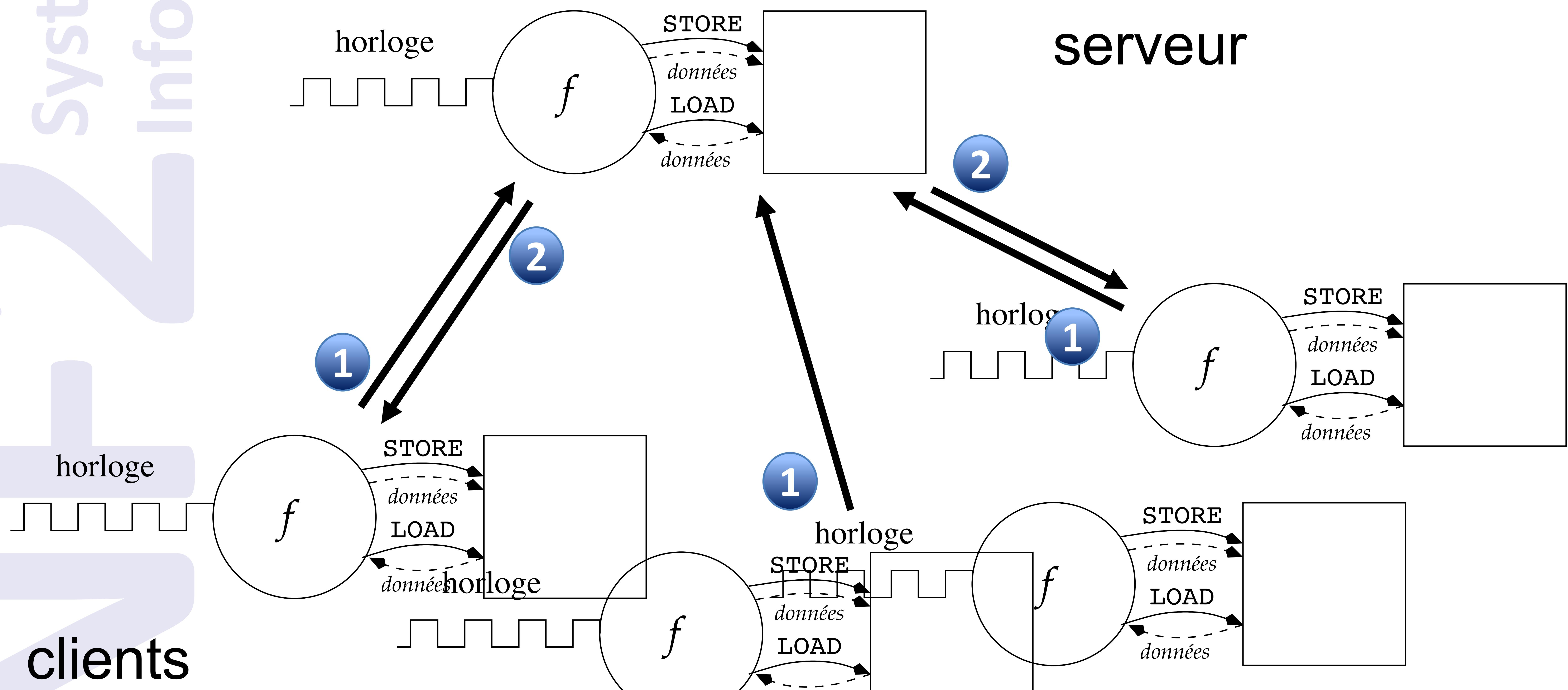
La relation de communication (2)

- Échange **asynchrone** de message



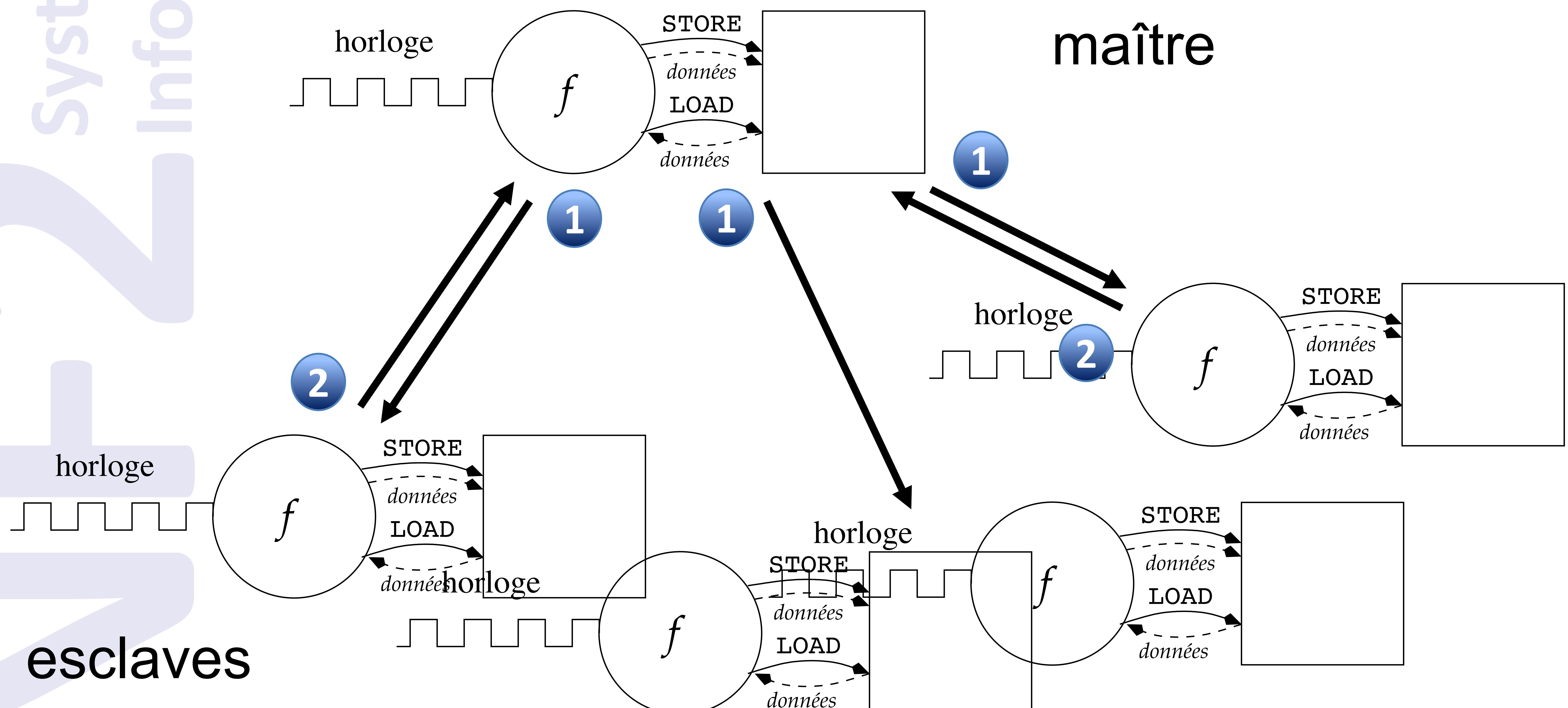
La relation de communication (3)

- Client-serveur – ex. web



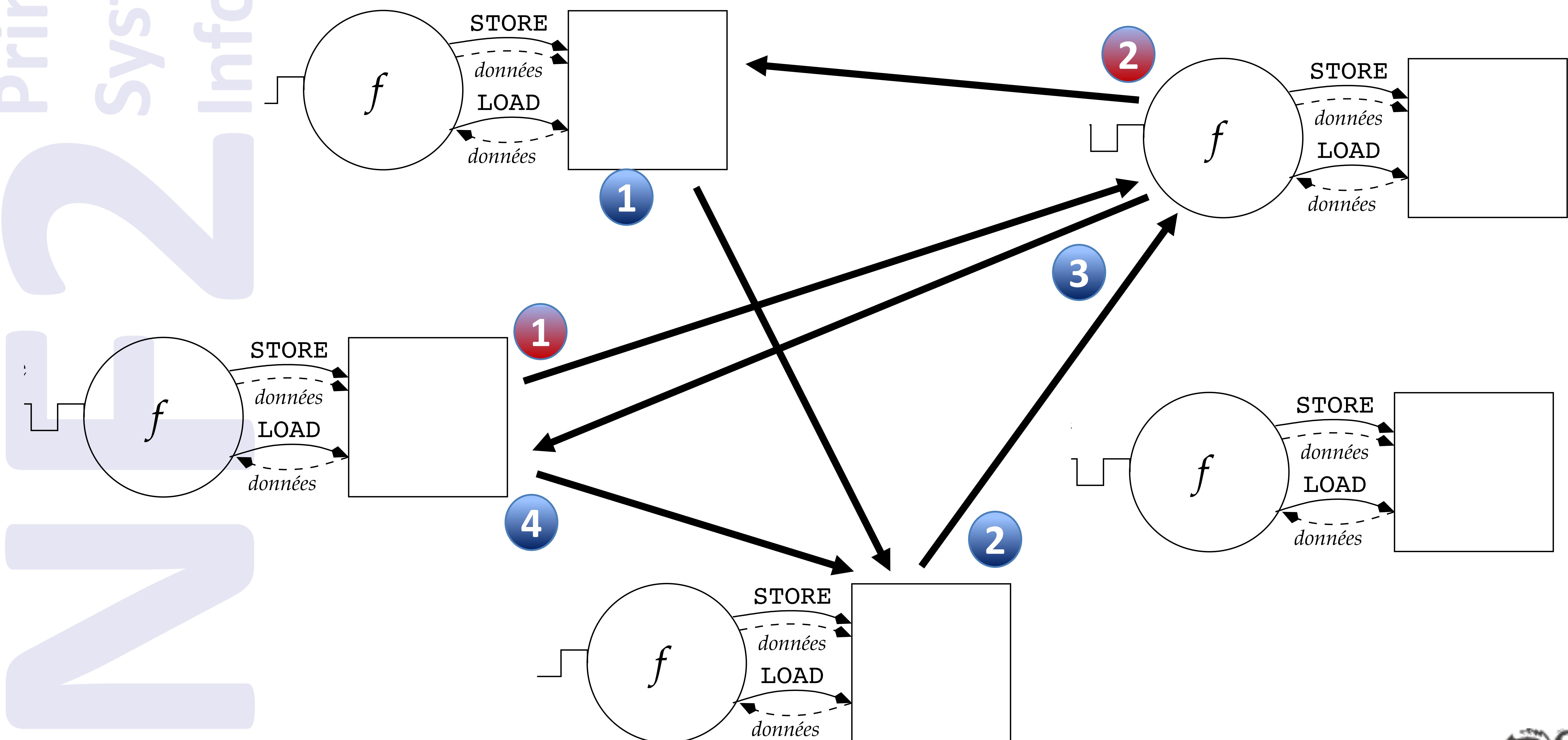
La relation de communication (4)

- Maître-esclave – ex. USB



La relation de communication (5)

- pair-à-pair – ex. web



Que contient la mémoire ? (1)

- La mémoire contient des **symboles**
- Les symboles représentent
l'état du système
- Fonction de **changement d'état**

Que contient la mémoire ? (2)

- Exemple : recevoir des messages Morse

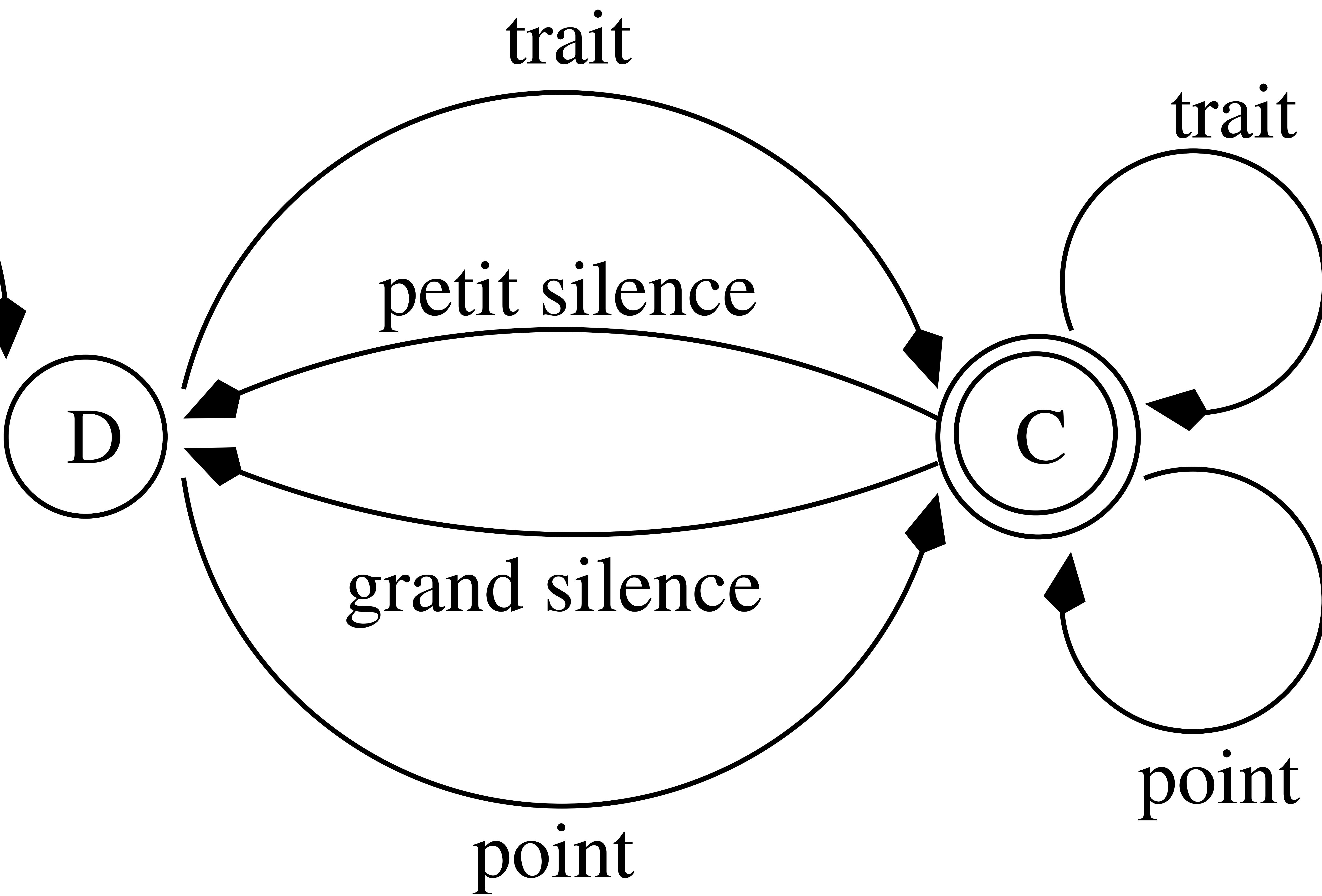
- Combien de symboles ?

- TRAIT et POINT, évident
 - PETIT et GRAND silences entre les lettres et les mots

- Ne peut pas commencer ni finir par un silence
- Ne pas avoir de silence consécutifs

Que contient la mémoire ? (3)

- **Fonction de transition**



Que contient la mémoire ? (4)

Modélisation

par **automate à nombre fini d'états**

– **états**, en nombre fini : Q

– état **initial** $q_0 \in Q$

– états **finaux** $F \subset Q$

– **symboles**, en nombre fini : V

– **relation de transition**, $\delta : (Q \times V) \rightarrow Q$

$$\delta(q_1, v) = q_2 \quad \text{ou} \quad q_1 \xrightarrow{v} q_2$$

dans l'état q_1 aller dans l'état q_2 si on reçoit un v

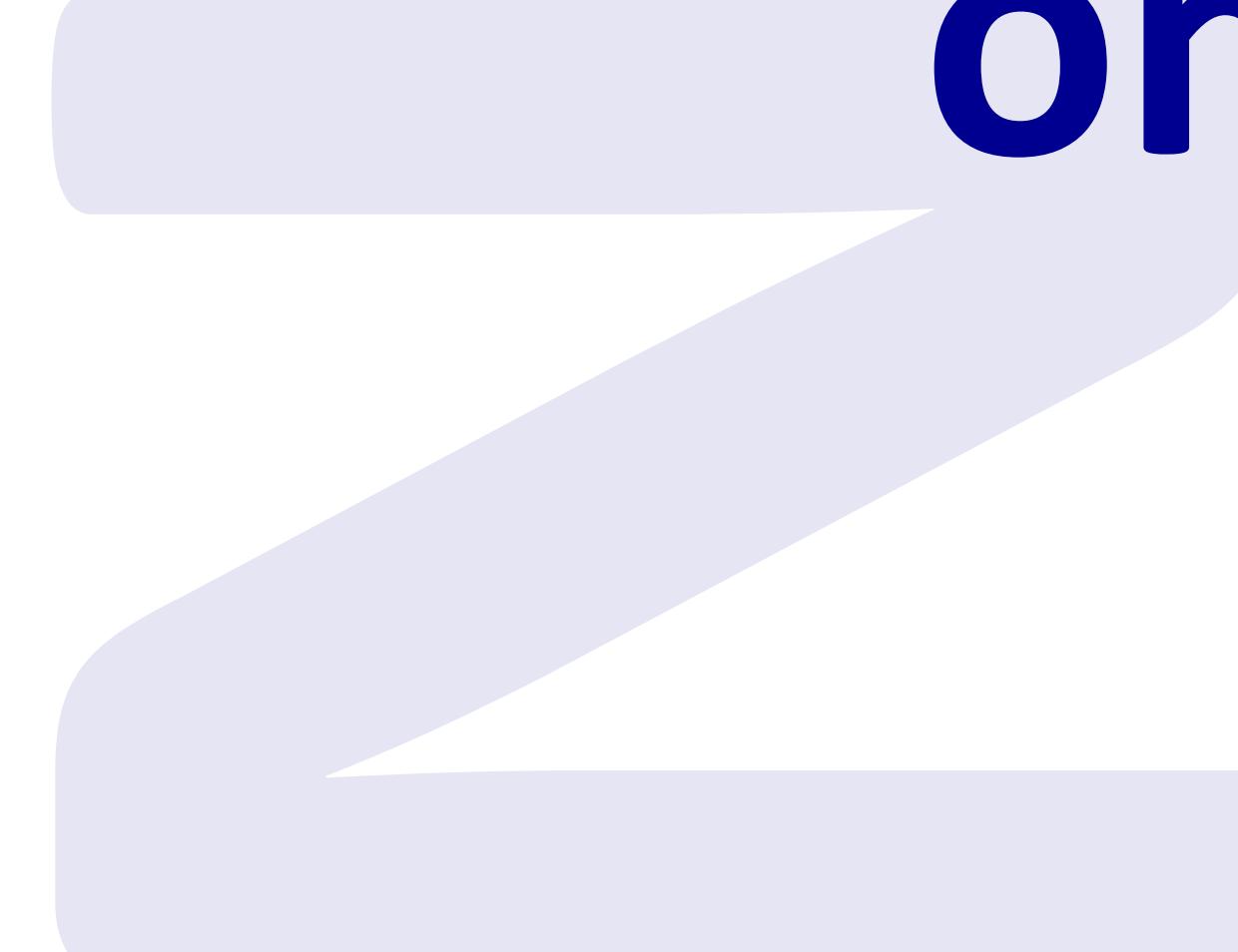
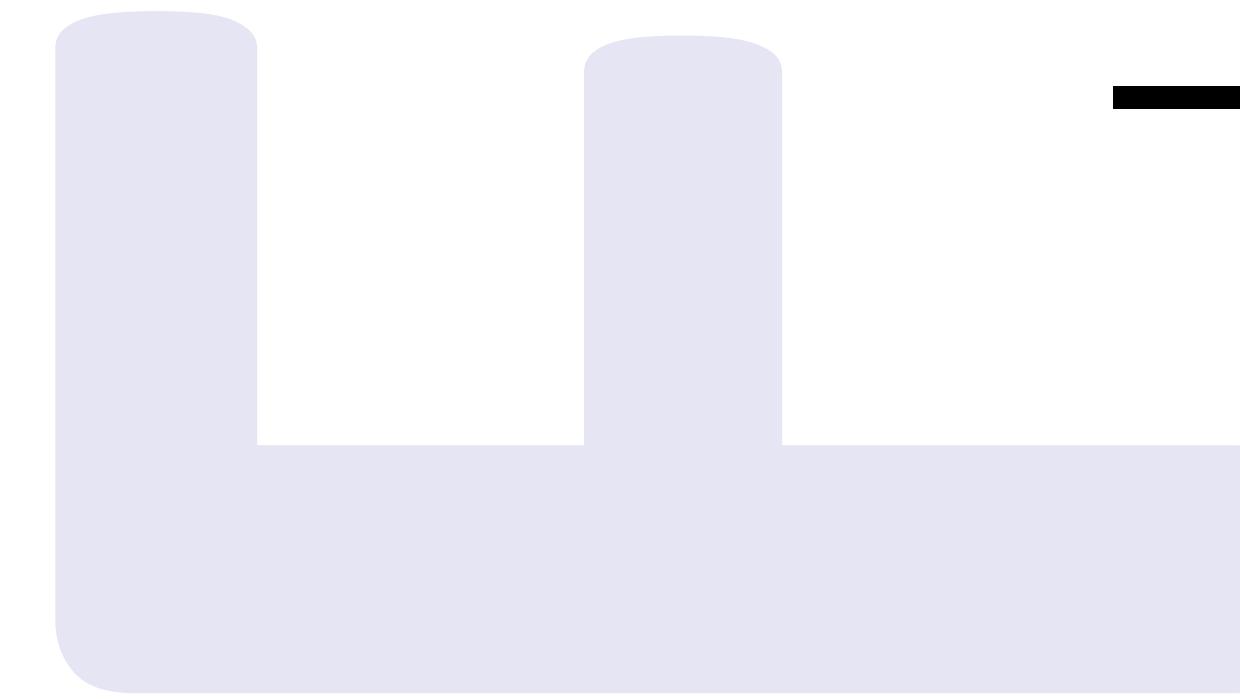
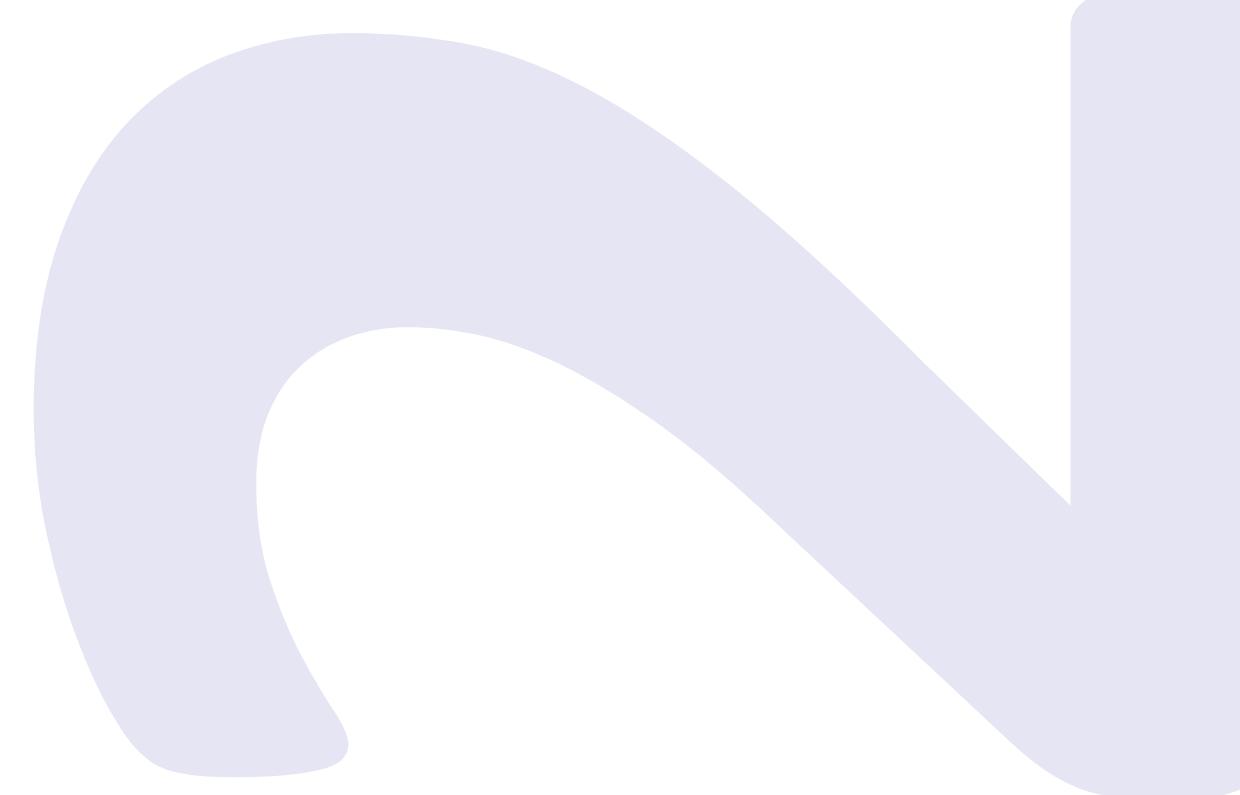
Automate fini (1)

• Fonctionnement

- placer l'automate dans l'**état initial**
- lire les symboles en effectuant les **transitions**

$$q_0 \xrightarrow{v^1} q_1 \xrightarrow{v^2} q_2 \xrightarrow{v^3} q_3 \dots \xrightarrow{v^n} q_n$$

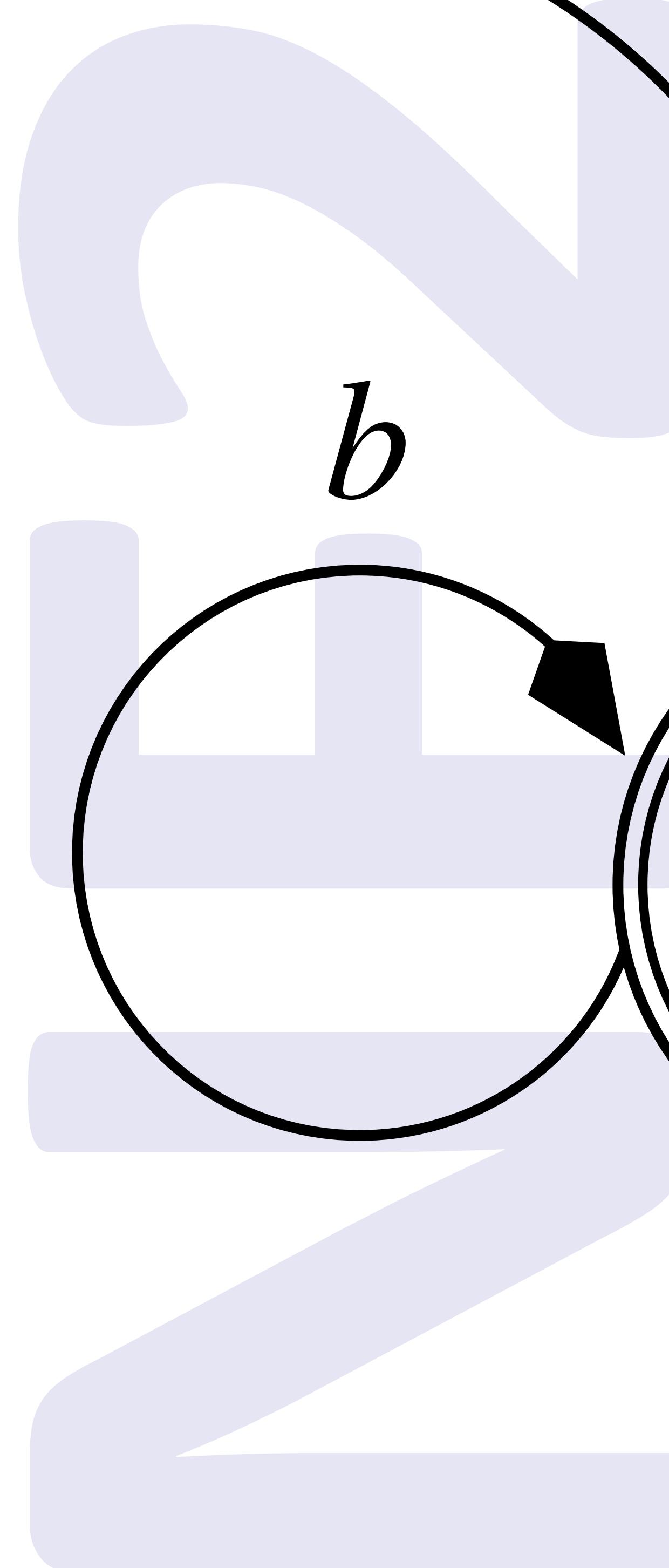
- ne s'arrêter que dans un **état final**



Automate fini (2)

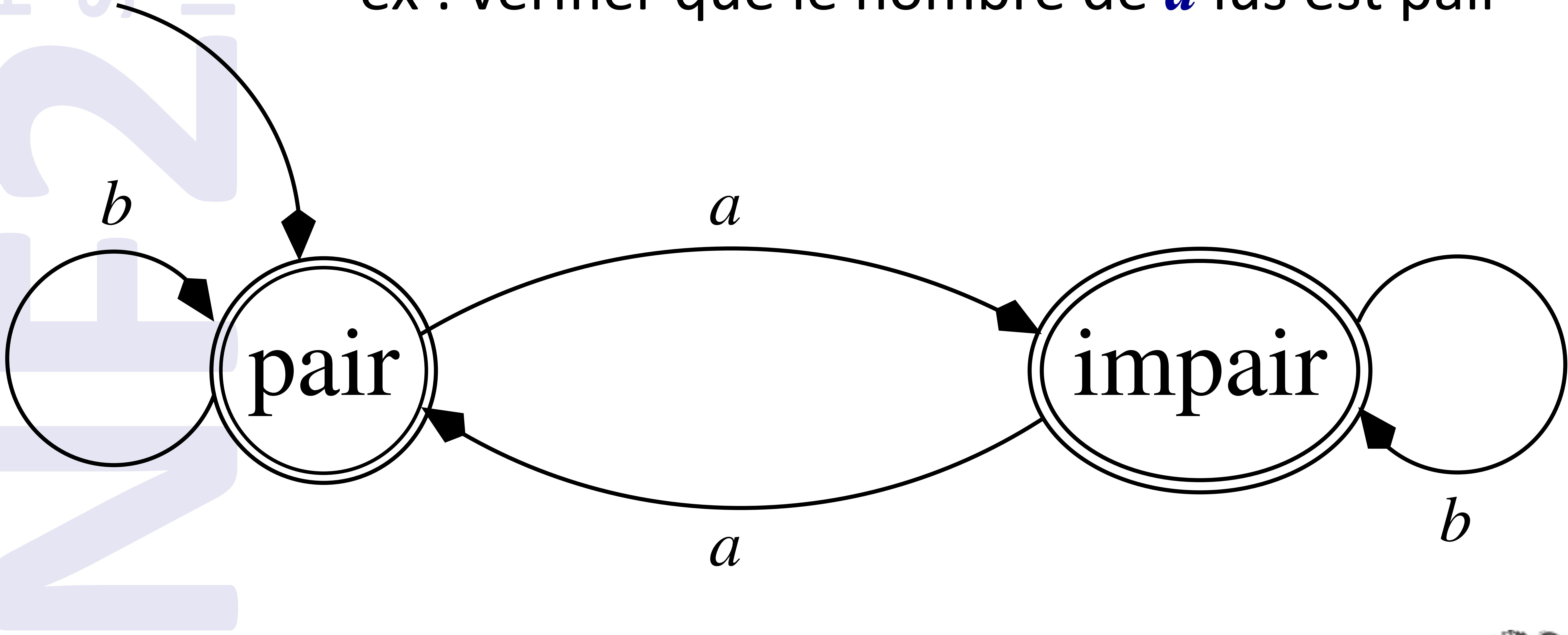
- Formalisme extrêmement répandu
 - simple et puissant
 - **mémoire limitée**
 - **ne sait pas compter**
 - des variétés plus puissantes existent

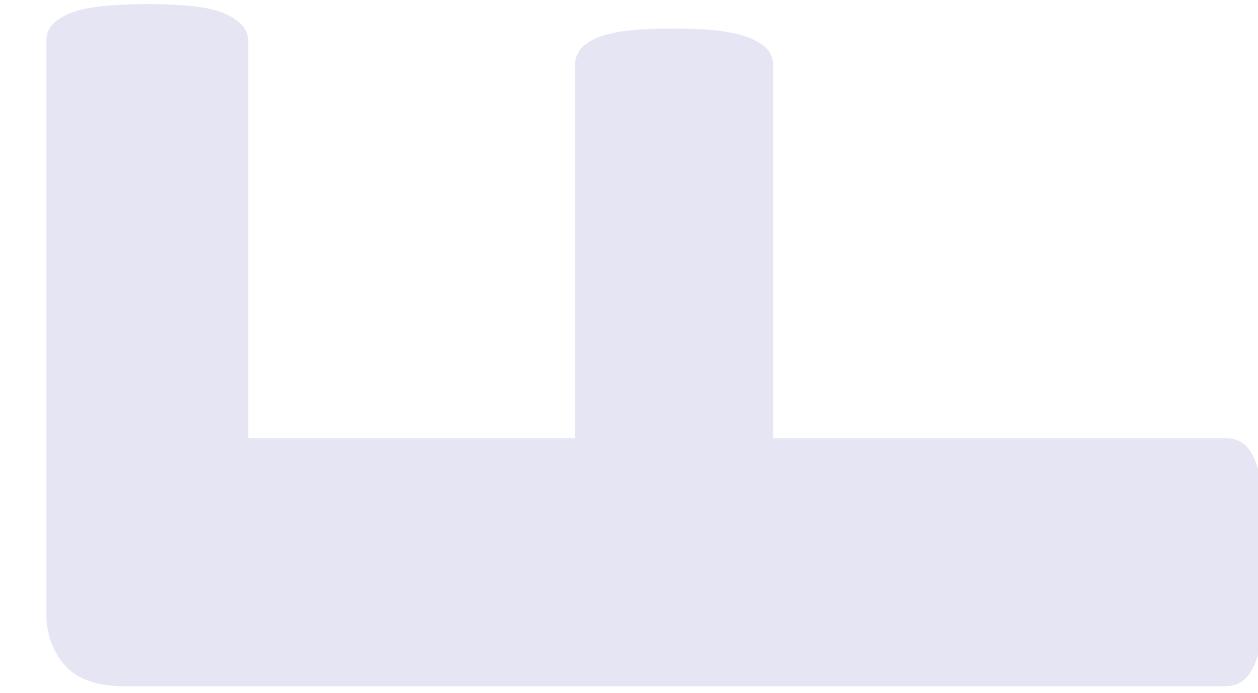
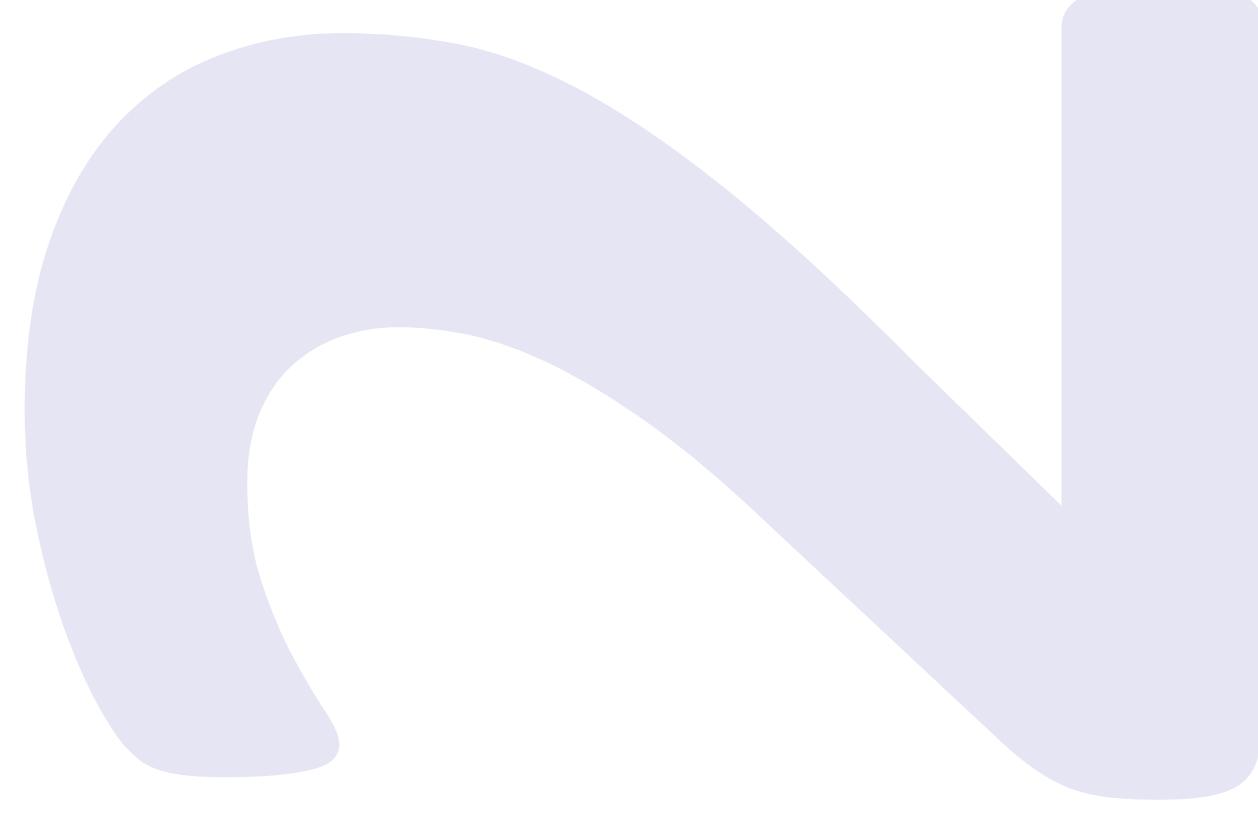
**on a moins souvent besoin de compter
qu'on le croit !**



Automate fini (3)

- Vérifier sans compter !
 - ex : vérifier que le nombre de *a* lus est pair





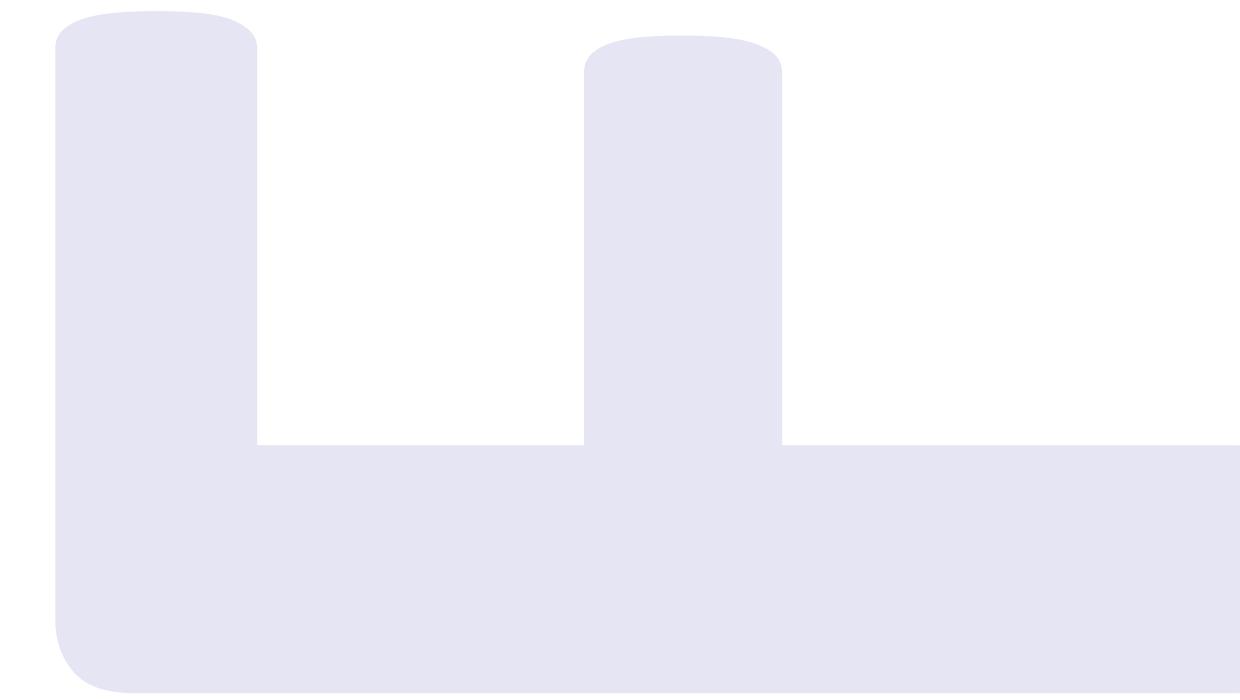
Conclusion (1)

- Systèmes
 - entité & relation
 - naturel / artificiel
 - ouvert / fermé
 - dedans / dehors
 - vue de l'esprit
 - propriété émergente



Conclusion (2)

- Entité informatique
 - calculante
 - communicante
 - mémorisante
- Modèle connu depuis 1936 (Turing) et exploité depuis 1945 (von Neumann)



Conclusion (3)

- Calcul itératif via la mémoire
 - $c_0 \rightarrow f(c_0) \rightarrow f(f(c_0)) \rightarrow f(f(f(c_0))) \rightarrow \dots$
 - recherche d'un point-fixe, $c = f(c)$
- Contenu de mémoire
 - = symbole représentant un état
 - automate
 - fonction de transition