# Chapitre 6 : Documenter avec UML

- Diagrammes d’états
- Diagrammes de séquence

# Pourquoi documenter ?

# Pourquoi documenter ?

- Travail en équipe
- Support pour les tests
- Maintenance

# Diagramme d'états

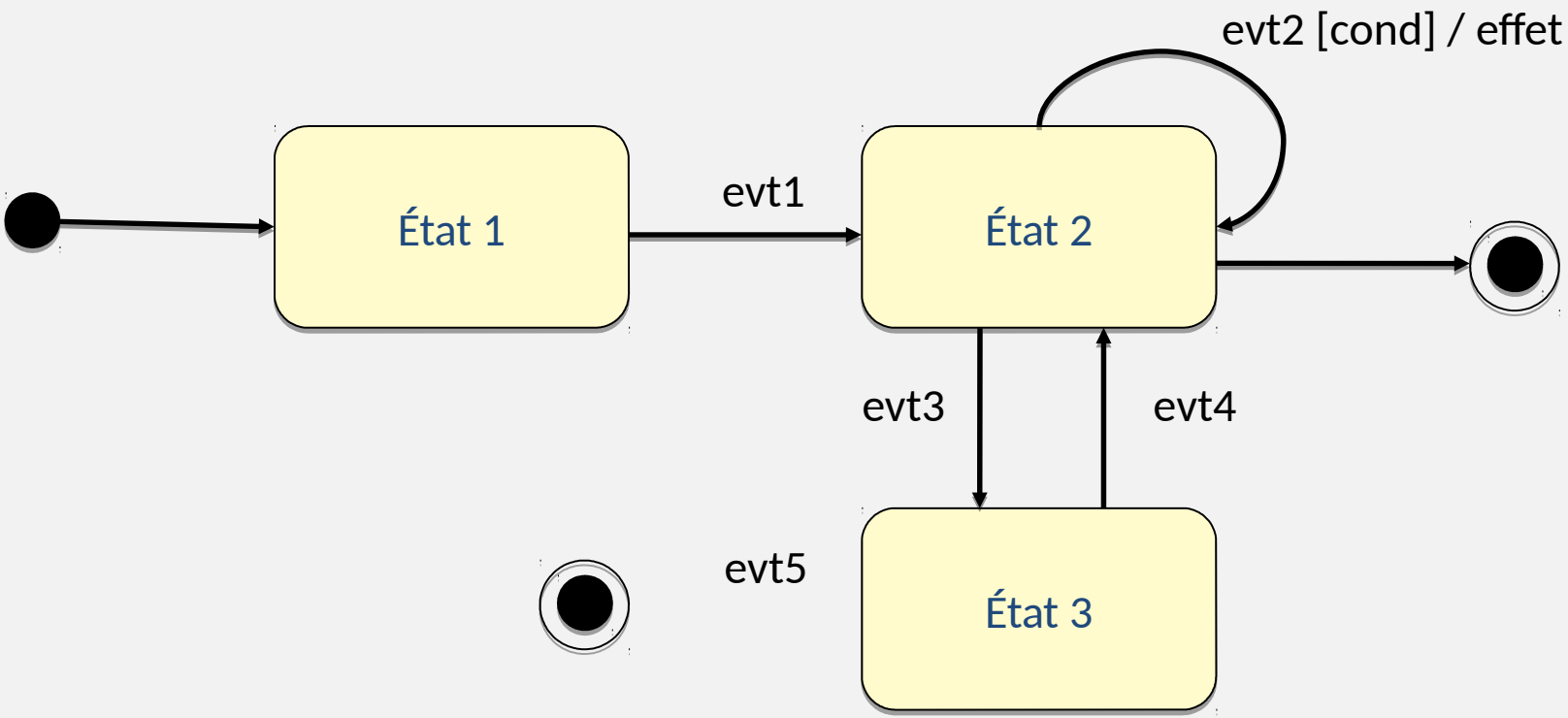
Description **dynamique** d'une classe à l'aide d'un automate fini (déterministe) :

Représentation des différents cycles de vie possibles d'un objet.

**Note :** tous les diagrammes d'états ne sont pas pertinents.

# Diagramme d'états

- Représentation graphique:



# Diagramme d'états

- **État :** situation où l'objet :
  - satisfait certaines conditions,
  - peut attendre un **événement**
  - peut exécuter une **activité**.

- Représentation graphique:


**pseudo états**

# Diagramme d'états

- **Transition :** relation orientée entre deux états.
  - peut être automatique ou déclenchée par un **évènement**.
  - peut être conditionnée par une **garde**.
  - peut également spécifier un **effet** (instructions de base, appel de méthode…)

- Représentation graphique et notation :

# Diagramme d'états

- Différents types d'**évènements** :
  - Réception d'un **signal** : en général envoyé par un autre objet (ex : un clic souris)
  - **Appel** d'une opération : ce sont des méthodes de l'objet
  - Écoulement du **temps** : mot clé `after(durée)`
  - **Changement** d'une condition : mot clé `when(condition)`

# Diagramme d'états

- **Transition interne** : définies dans un état.

  Évènements spécifiques :

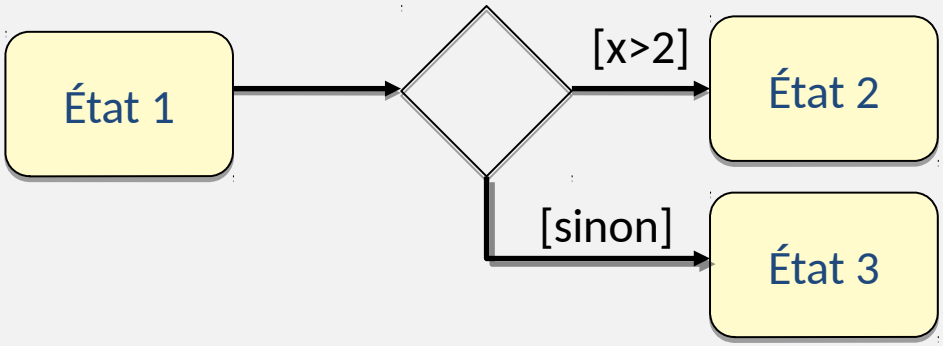
  - **entry** / effet : à l'arrivée dans l'état
  - **exit** / effet : à la sortie de l'état
  - **do** / effet : pendant l'état.

- **Transitions propre** : état départ = état arrivée

Note : une transition propre déclenche à chaque fois les effets entry et exit.

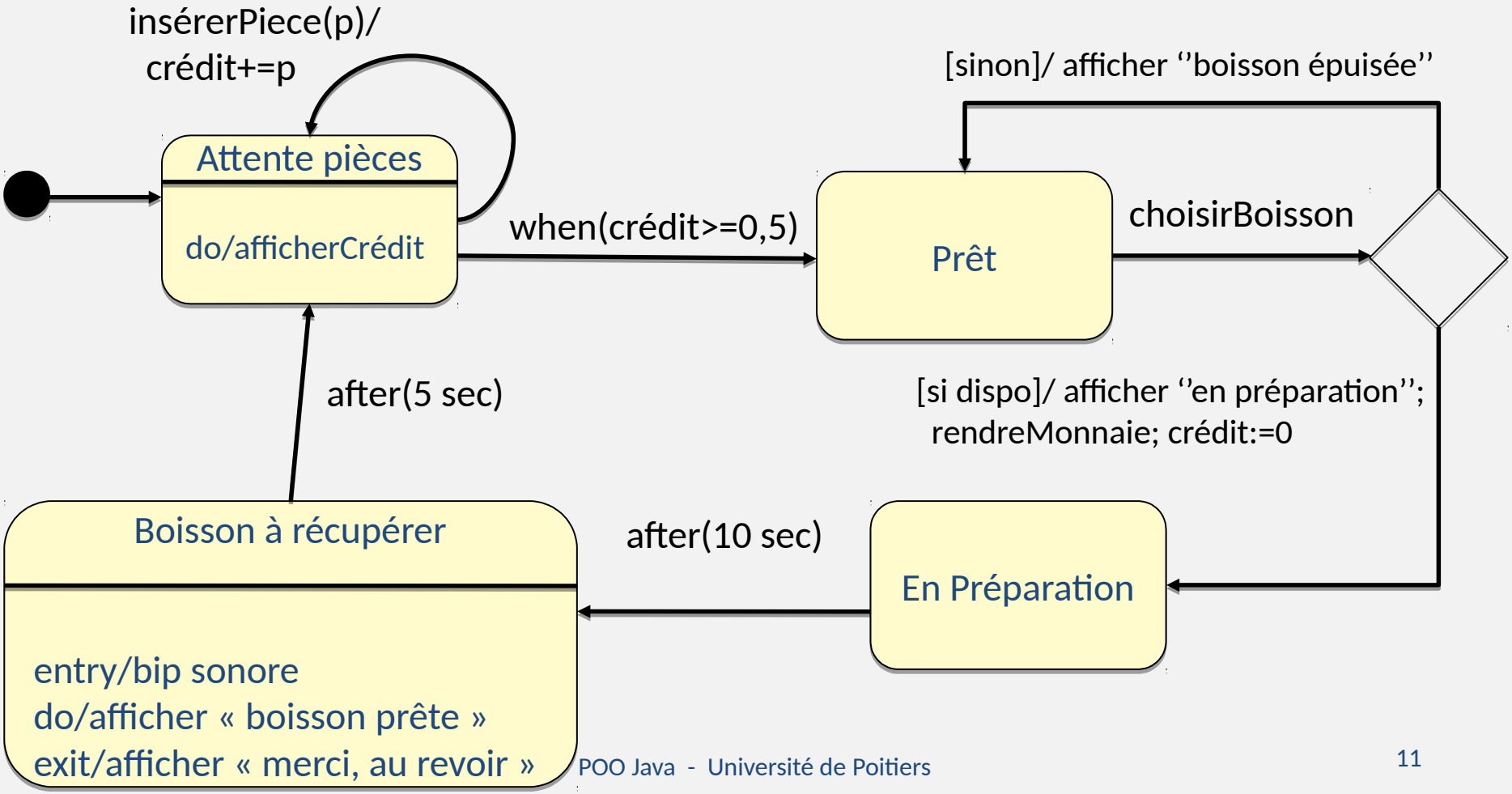
# Diagramme d’états

- Le pseudo état **point de choix :** une transition entrante et au moins deux transitions sortantes.



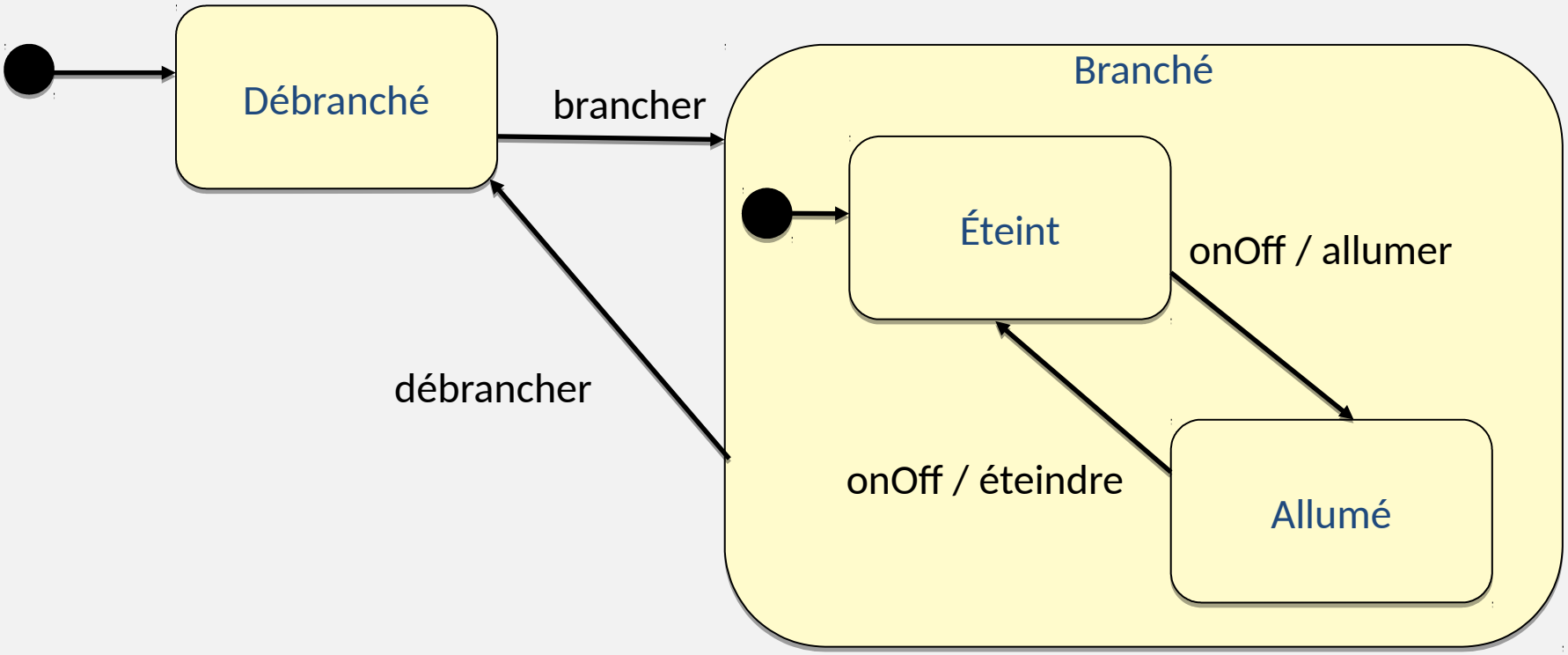
# Diagramme d'états

## Exemple : distributeur de café



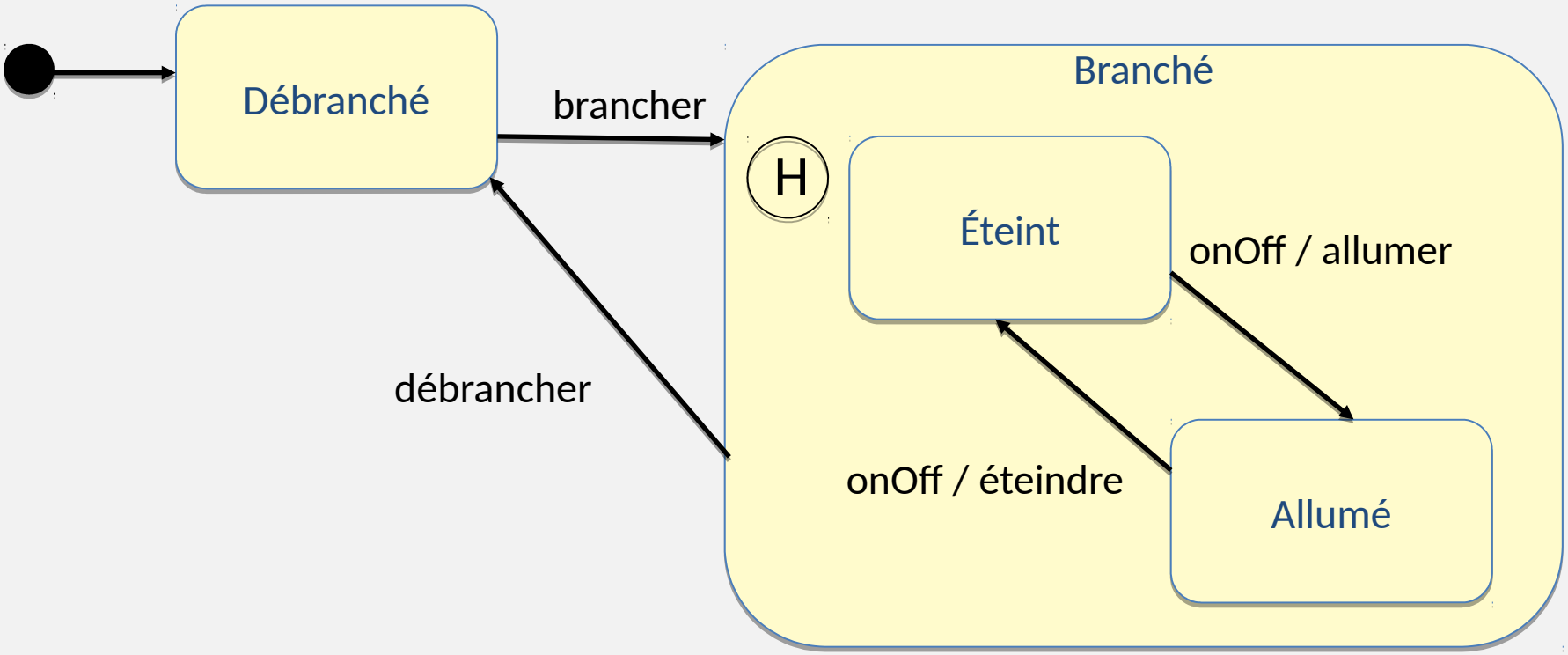
# Diagramme d'états

- Un état peut éventuellement être décomposé en sous-états, on parle alors d'**état composite**



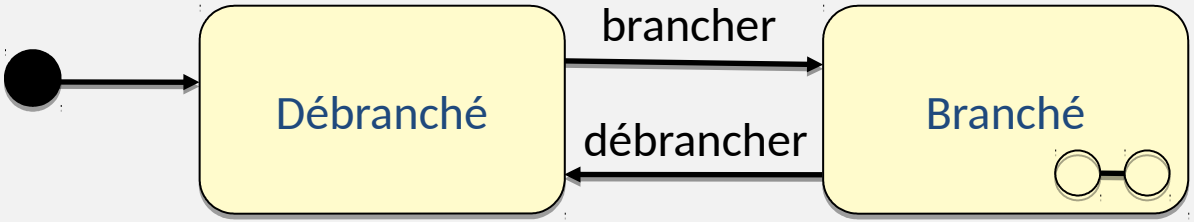
# Diagramme d'états

- Utilisation d'un état historique.



# Diagramme d’états

- On peut indiquer qu’un état est un état composite, sans le détailler :



# Diagramme d'états

Exemple : du distributeur de café

- **ATTENTION** : dans notre exemple, le diagramme d'états ne spécifie pas entièrement la classe `DistributeurCafé`.

- Par exemple, que fait la méthode `insérerPièce` si l'instance en état *en préparation* ? (c.f. chapitre sur les tests)

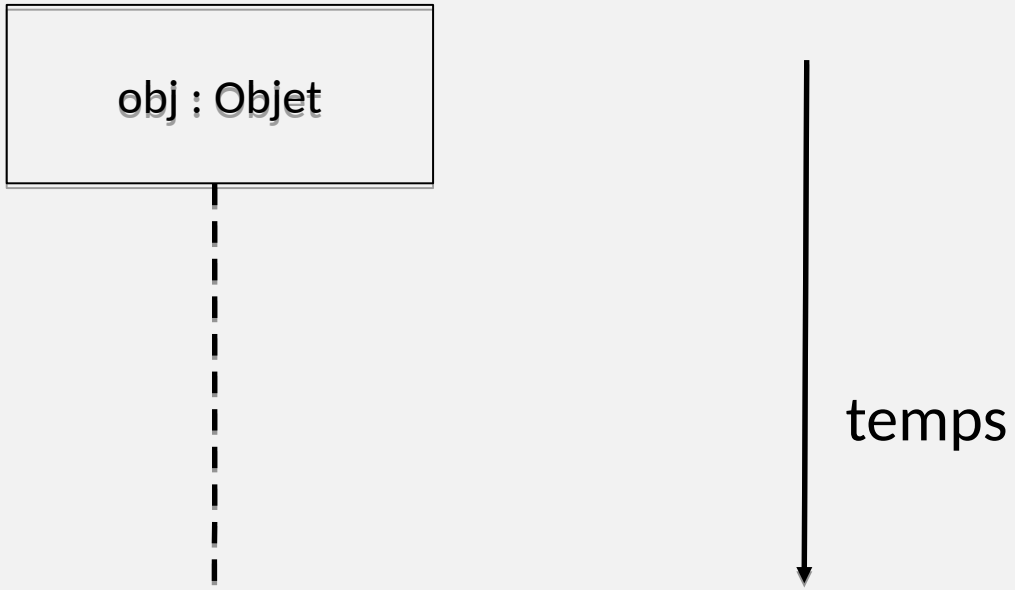
# Diagramme de séquences

- Un diagramme de séquences décrit **une fonctionnalité**, selon un point de vue temporel.

- permet de représenter des **interactions entre des classes**.

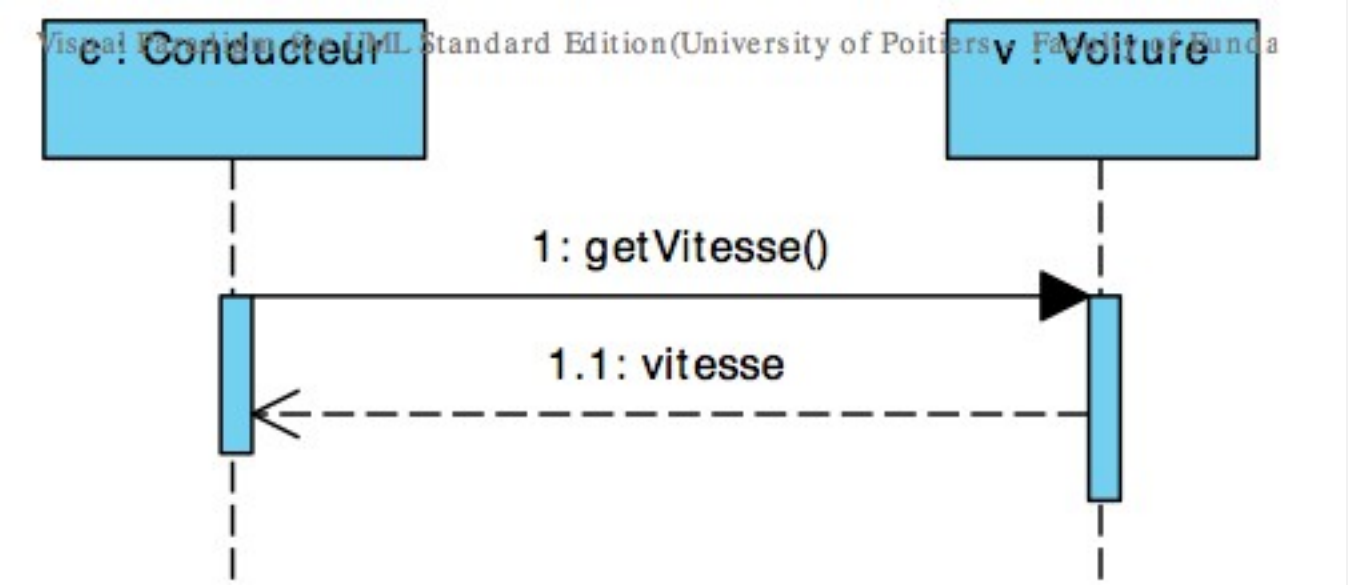
# Diagramme de séquences

- Ligne de vie d'un objet : le **temps est représenté par l'axe vertical** (temps s'écoule de haut en bas).



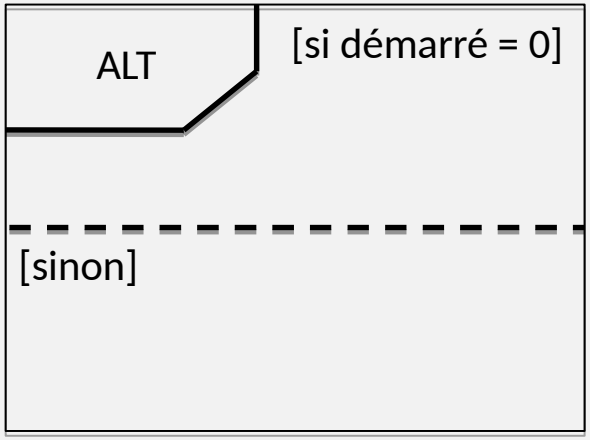
# Diagramme de séquences

- Échange de message entre objets

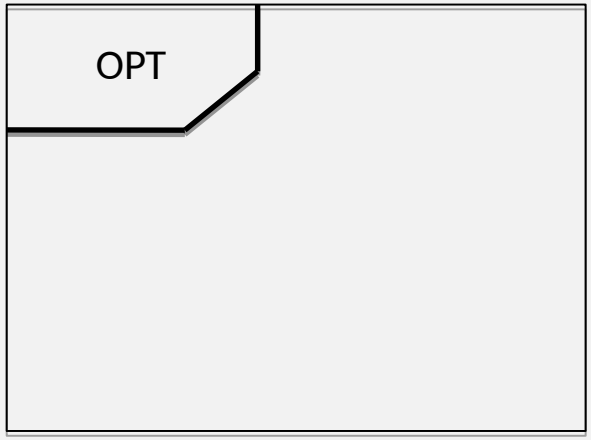


# Diagramme de séquences

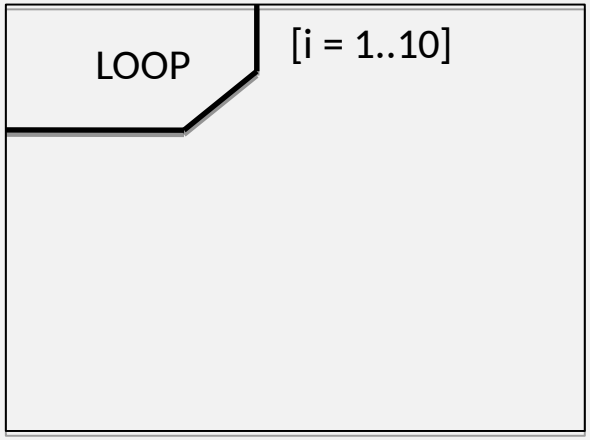
- **Fragments combinés** permettent de décrire entre autres les structures de contrôle usuelles.




**If then else**




**if**




**loop**

# Diagramme de séquences

- Les acteurs :

Rôle joué par quelque chose ou quelqu'un d'extérieur au système et qui interagit avec (client, gestionnaire, imprimante…).

client

# Diagramme de séquences