

# Méthodes pour l'informatisation - compléments

---

Diagrammes de collaboration  
Diagrammes d'états

Christine Plumejeaud

Doctorante Informatique UJF

**CNAM - centre de Grenoble**

# Plan

---

1. Diagramme de collaboration
2. Diagramme d'état

# Diagramme de collaboration

---

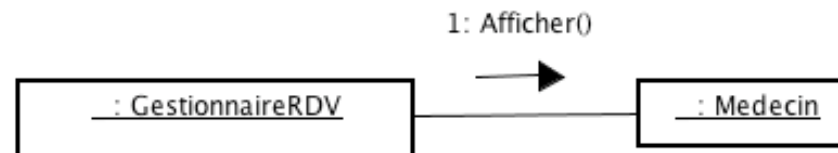
- Il montre des interactions entre objets, en insistant sur la structure spatiale statique qui permet la mise en collaboration d'un groupe d'objets.

- C'est une extension du diagramme d'objets

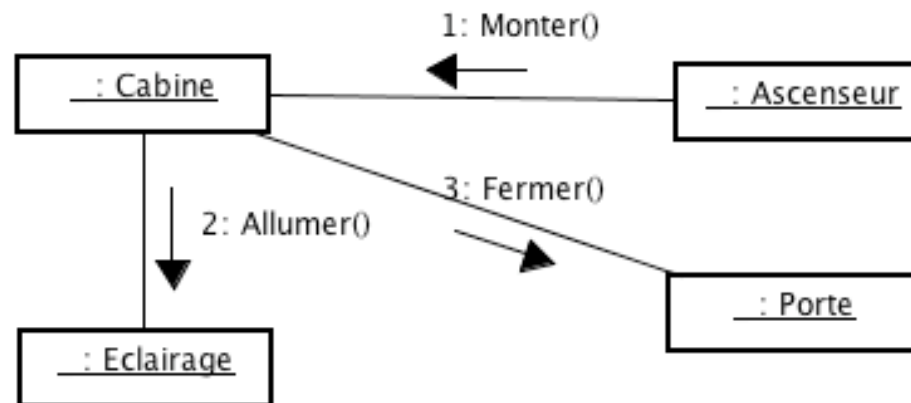
## Représentation des interactions

- Le contexte d'une interaction comprend les arguments, les variables locales créées pendant l'exécution, ainsi que les liens entre les objets qui participent à l'interaction
- Une interaction est réalisée par un groupe d'objets qui collaborent en échangeant des messages

# Diagramme de collaboration

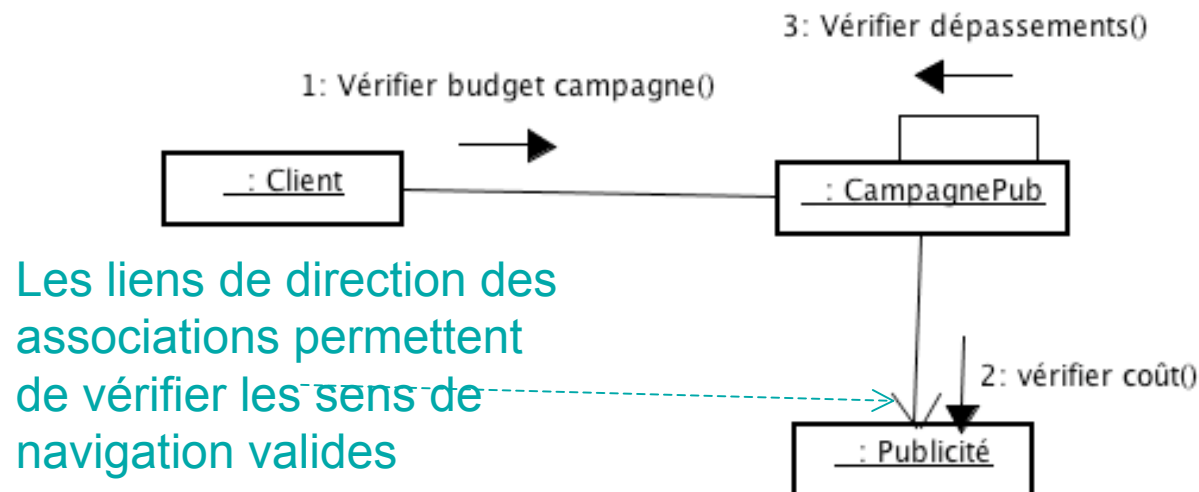


- Le temps n'est pas représenté de manière explicite : les messages sont numérotés pour indiquer l'ordre des envois



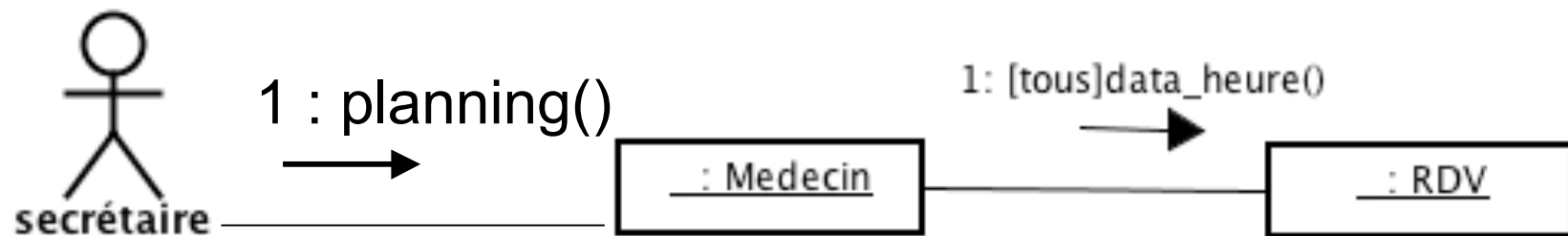
# Diagramme de collaboration

Les diagrammes de collaboration montrent simultanément les interactions entre les objets (**acteurs**) et les relations structurelles qui permettent ces interactions



## Place de l'utilisateur dans un diagramme de collaboration

La notation permet de faire figurer un acteur afin de représenter le déclenchement des interactions par un élément externe au système.

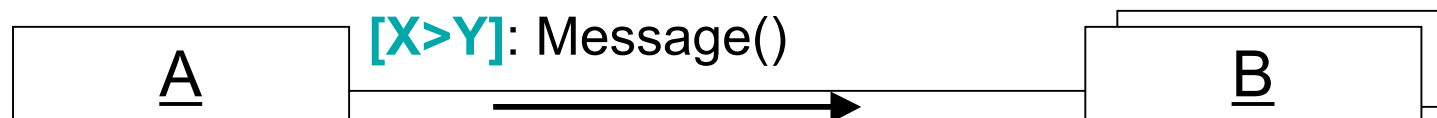


## Les messages dans un diagramme de collaboration

1. La clause d'itération est optionnelle; elle est exprimée par **\*[clause]**:



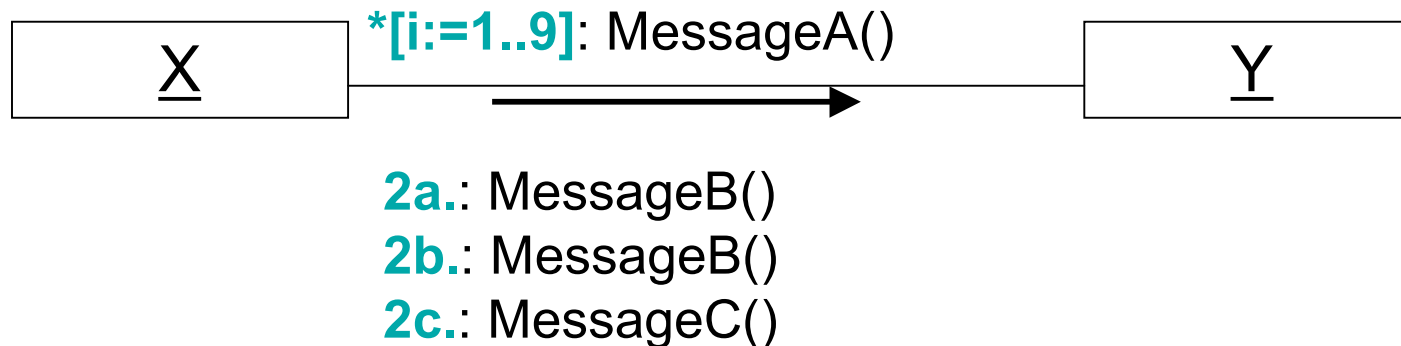
2. La clause de condition valide ou non l'envoi des messages contenus par le bloc. Elle est exprimée par **[condition]**:



# Les messages dans un diagramme de collaboration

---

## Poster des messages en parallèle



L'invocation des messages `2.a:messageB`,  
`2.b:messageB`, `2.c:messageC` se fait en  
même temps, après celle de `1.messageA`

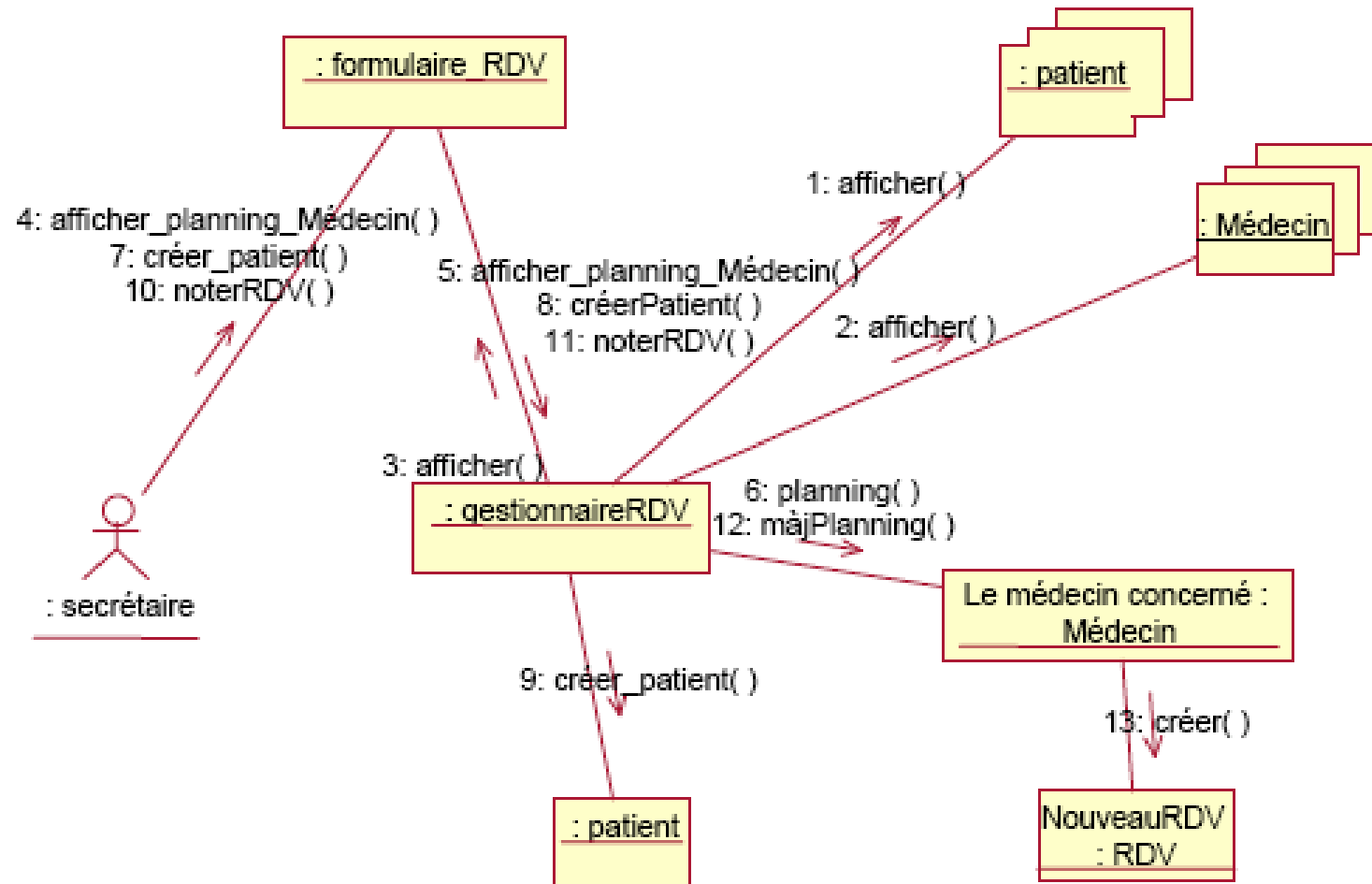


# Vérifier la cohérence des modèles

---

1. La définition des **opérations** dans les classes doit être cohérente avec les message, et les signatures de ces opérations avec les arguments des messages
2. Tout objet à l'origine d'un message doit avoir la référence de l'objet destination du message
  - A travers une association, ou un chemin d'associations
  - Cette étape est importante pour la définition des associations dans le diagramme de classe
  - Bien vérifier tous les chemins de messages
3. Les diagrammes de séquence et de collaboration doivent être cohérents entre eux, et avec les diagrammes d'état transition des objets impliqués

# Exemple



# Les diagrammes d'états-transition

---

Ils permettent principalement de

- décrire le comportement des objets d'une classe.

Mais aussi,

- les aspects dynamiques d'un cas d'utilisation, d'un acteur, d'un système, d'un sous-système

Ils sont inspirés des « statecharts » de David Harel.

Il s'agit d'automates hiérarchiques qui peuvent être mis en parallèle.

# L'état

---

L'état d'un objet correspond à l'ensemble:

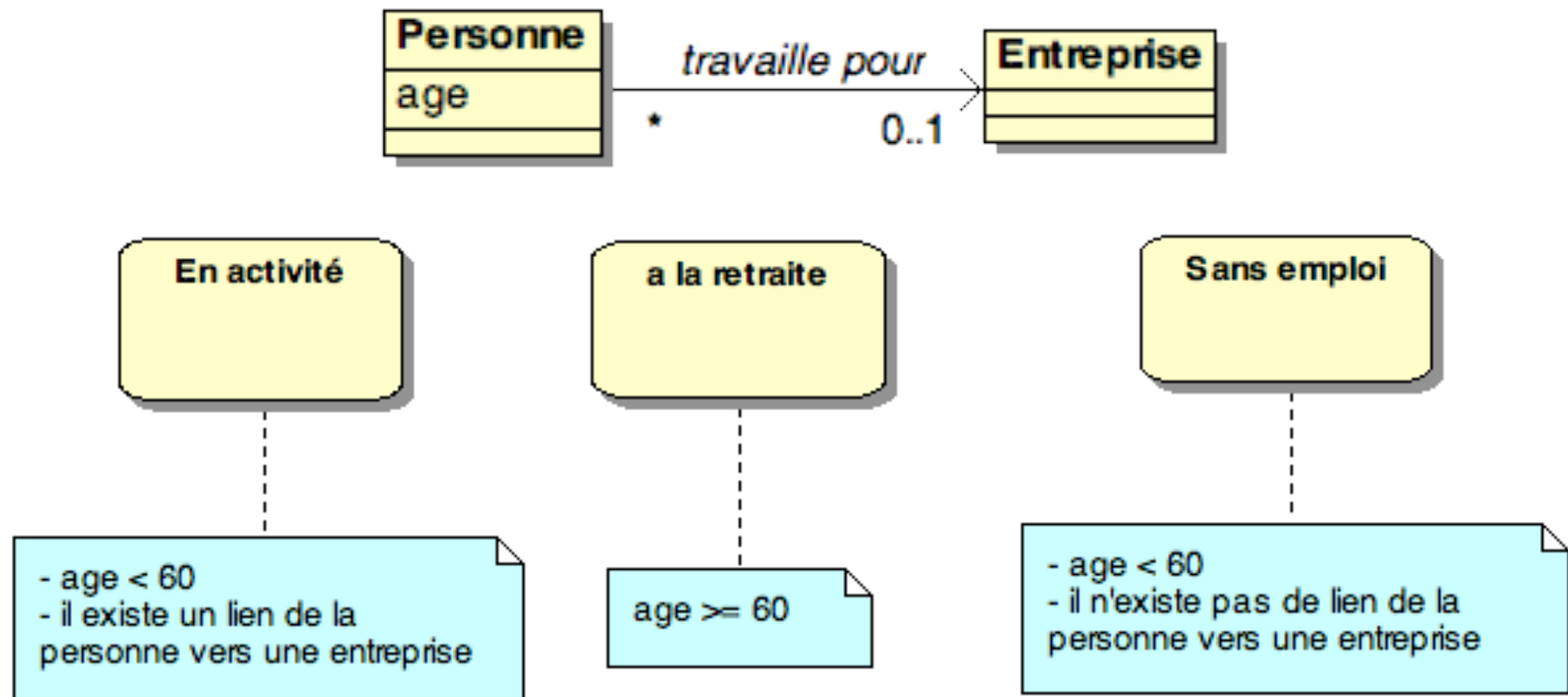
- des valeurs de ses attributs
- des liens qu'il entretient avec les autres objets.

Un état est une condition ou **une situation dans la vie d'un objet** qui dure un certain temps pendant lequel cet objet satisfait une condition, effectue une activité ou attend un événement

L'état d'un automate peut être vu comme une abstraction représentant l'ensemble des états de l'objet.

# Exemple

## 1. L'emploi des personnes (l'abstraction)



# Etats initial et final

---

1. Etat initial = pseudo-état permettant de montrer l'état dans lequel un objet se trouve au moment de sa création



Etat initial

2. Etat final = pseudo-état permettant de montrer la fin du comportement d'un objet, en particulier le moment de sa destruction.



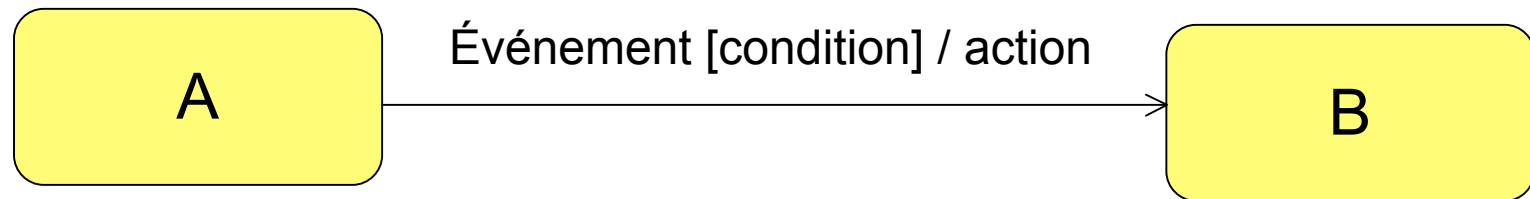
Etat final

# Les transitions

Les transitions permettent à un objet de changer d'état, en fonction des événements qu'il reçoit.

Une transition comporte:

- un état source,
- un état destination,
- un événement,
- une condition (appelée garde),
- une action.



# Évènement

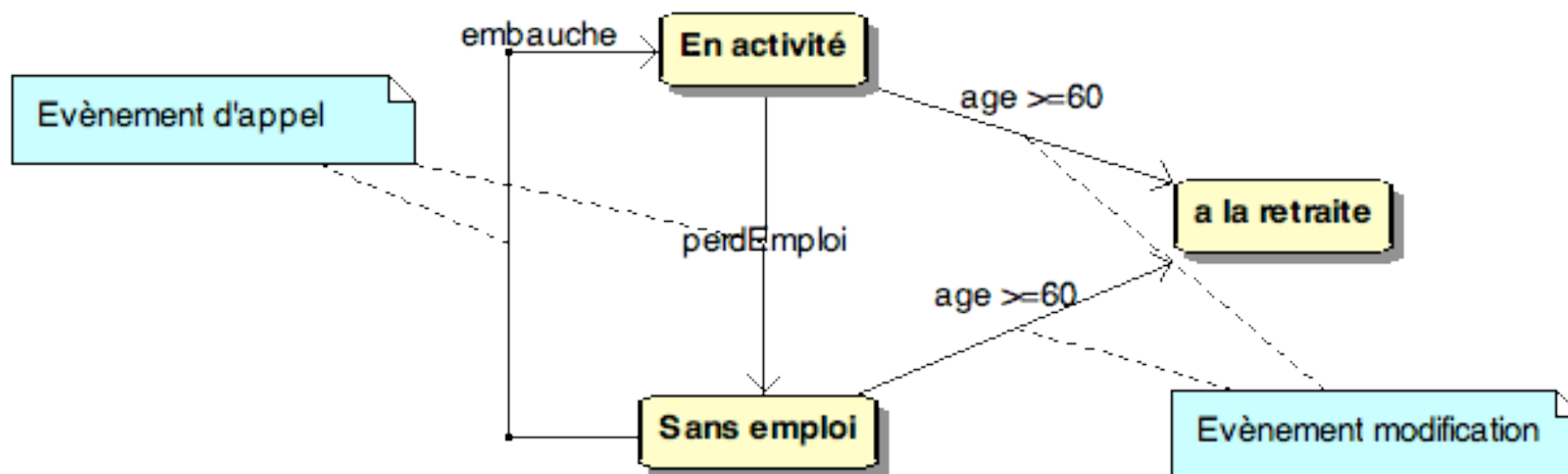
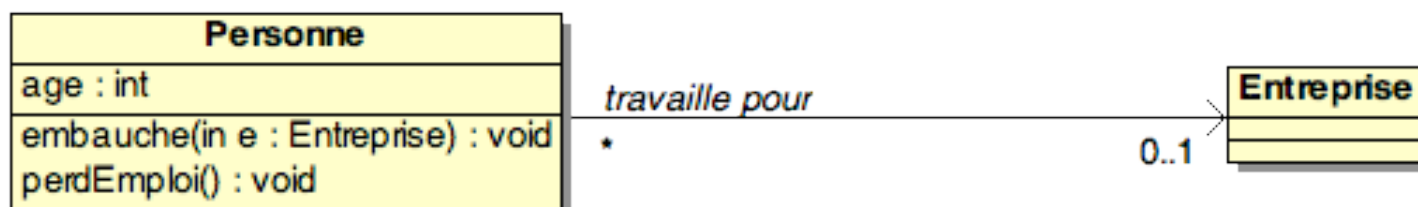
---

## 4 sortes d'évènements en UML

1. Évènement d'appel (« *call event* ») : causé par l'appel d'une opération, noté  $op(x_1, x_2, \dots, x_n)$  où  $op$  est une opération de la classe
2. Événement modification (« *change event* ») : causé par le passage de la valeur faux à vrai, suite à un changement de valeur d'un attribut ou d'un lien
3. Évènement temporel (« *time event* ») : survient quand une temporisation expire. Une temporisation peut-être relative (délai) ou absolue (heure de la transition).
4. Événement signal (« *signal event* ») : stimulus asynchrone entre 2 objets (clic de souris)

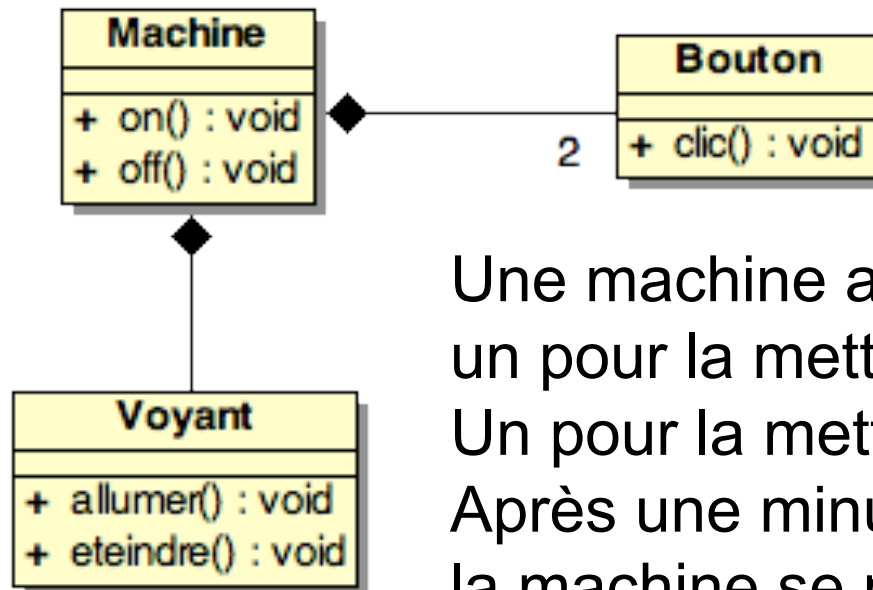


# Exemple 1



## Exemple 2

### Machine avec bouton et voyant d'allumage



Une machine avec 2 boutons :  
un pour la mettre sous tension, (signal = on)  
Un pour la mettre hors tension, (signal=off)  
Après une minute sans utilisation,  
la machine se met automatiquement  
hors tension

## Exemple 2 suite

Diagramme d'état-transitions  
associé à la classe Voyant

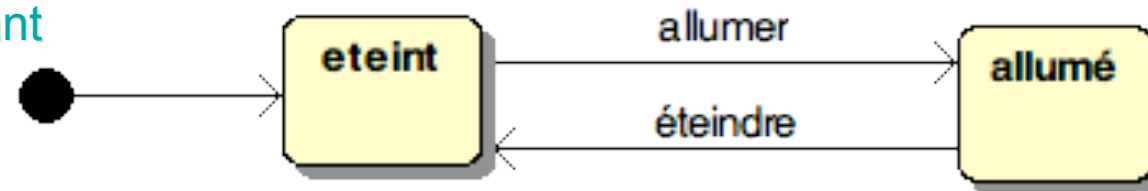
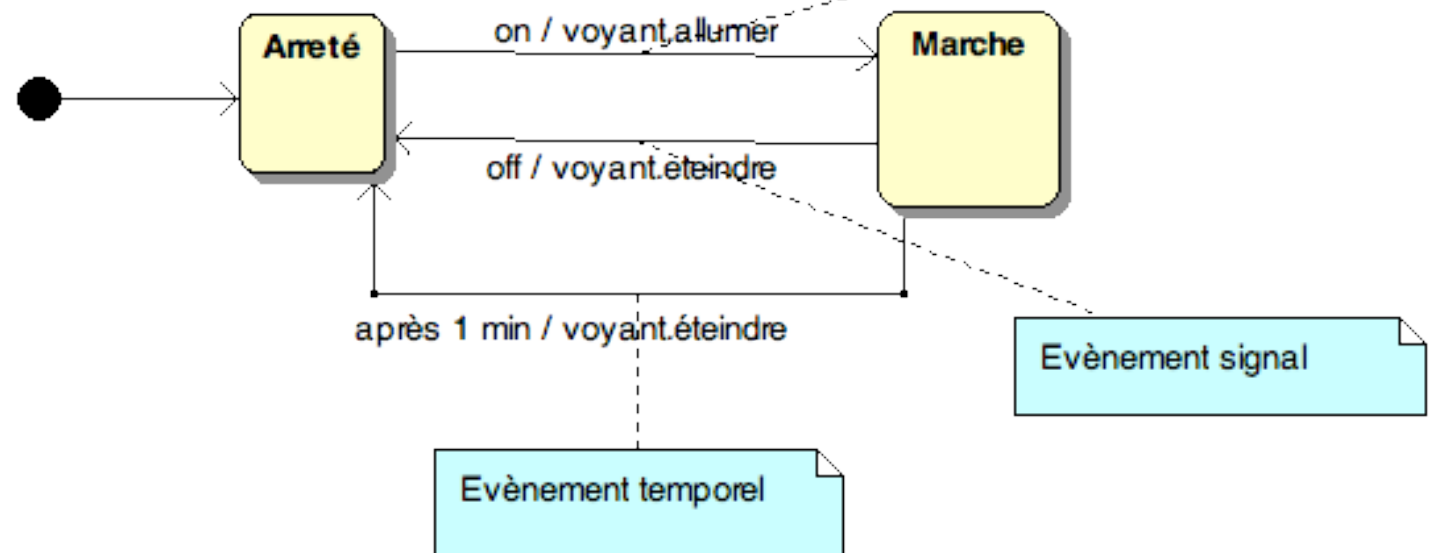


Diagramme d'état-transitions  
associé à la classe Machine



# Garde

---

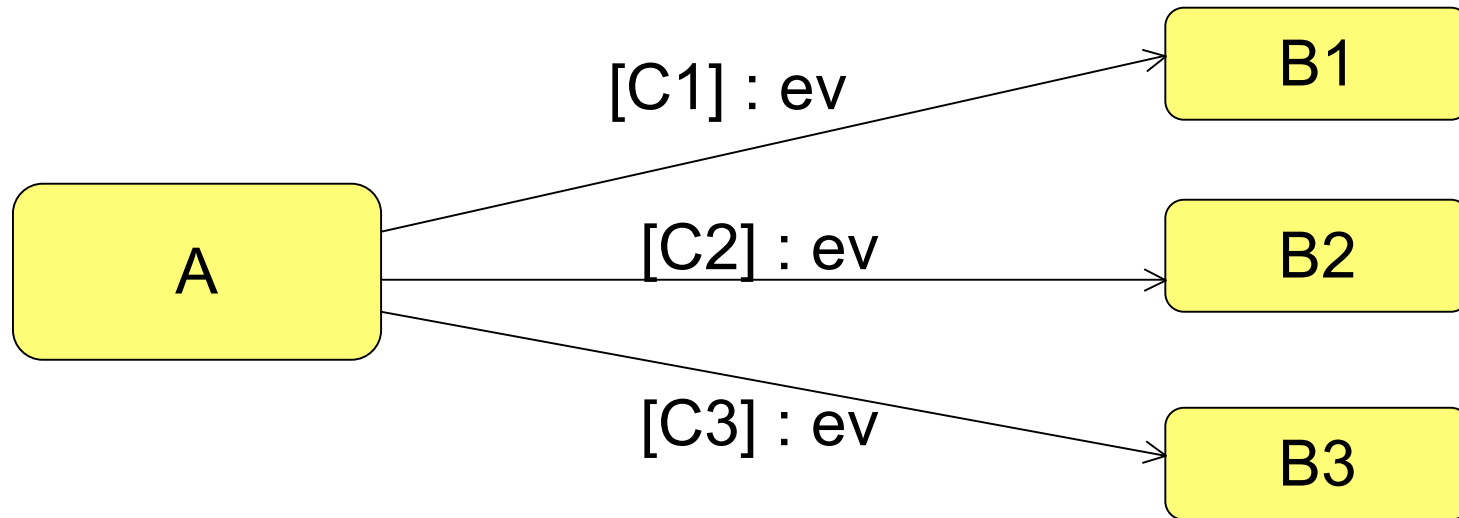
Une garde est une *condition booléenne* notée entre crochets. Elle est évaluée lorsque l'événement se produit.

Il ne faut pas confondre un événement de changement et une garde :

- Un événement de changement est un événement qui déclenche la transition lorsque la condition passe à vraie;
- une garde est une condition booléenne évaluée lorsque l'événement se produit.

## Garde : exemple

---



- Si C1, C2, et C3 sont exclusives alors l'automate est déterministe.
- Si aucune des conditions n'est remplie, l'objet ne change pas d'état.

# Actions et activité

---

Une action consiste en la génération d'un signal ou l'invocation d'une opération. Elle est *instantanée* (temps d'exécution négligeable) et *atomique* (non interruptible)

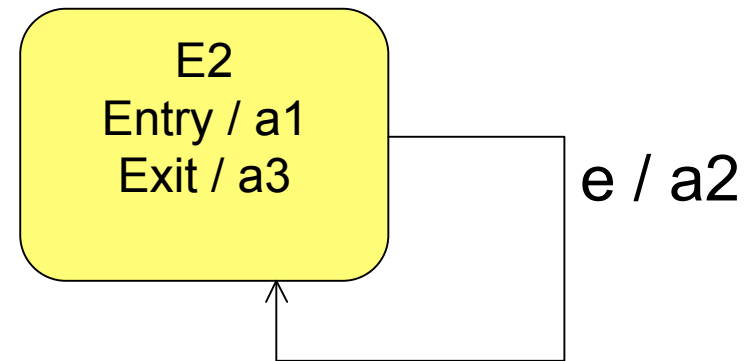
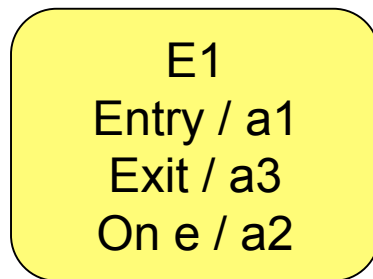
Une activité correspond à une opération qui prend un temps non négligeable et qui peut être interrompue.

Associations entre actions et états:

- Entry / action
- On événement / action
- Exit / action
- Do / activité

# Exemple

On entre dans l'un des états E1 ou E2,  
qu'on reçoit une suite d'évènements e,  
puis qu'on sort de l'état



Pattern d'exécution

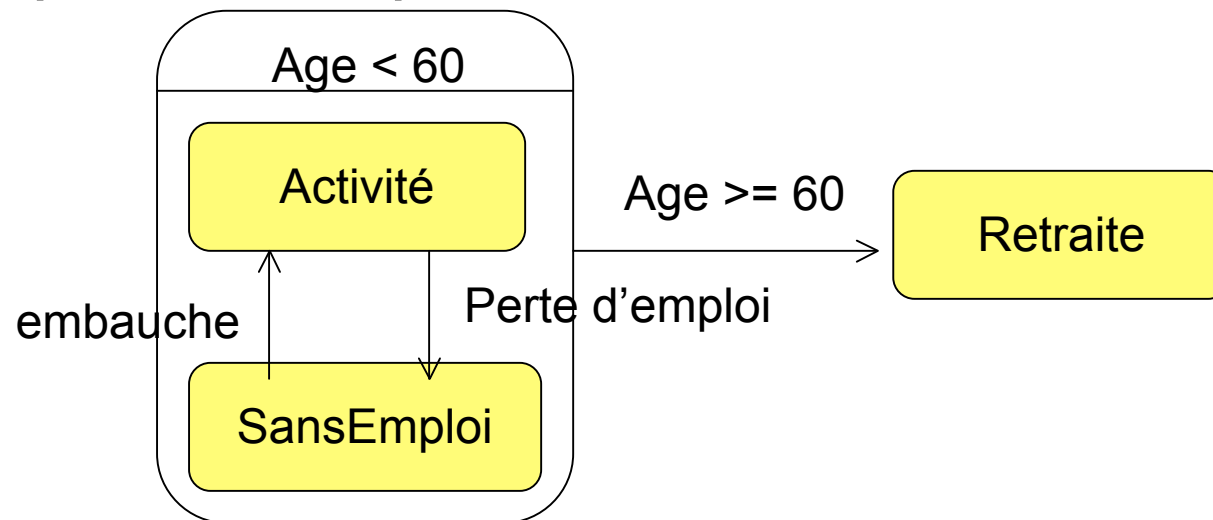
$a1\ a2^*\ a3$

Pattern d'exécution

$a1\ (a3\ a2\ a1)^*\ a3$

# Etats composites

Un état composite est un état qui se décompose en plusieurs sous-états. Ils permettent de structurer les automates pour les rendre plus lisibles. Et de factoriser les états similaires qui partent de plusieurs états.

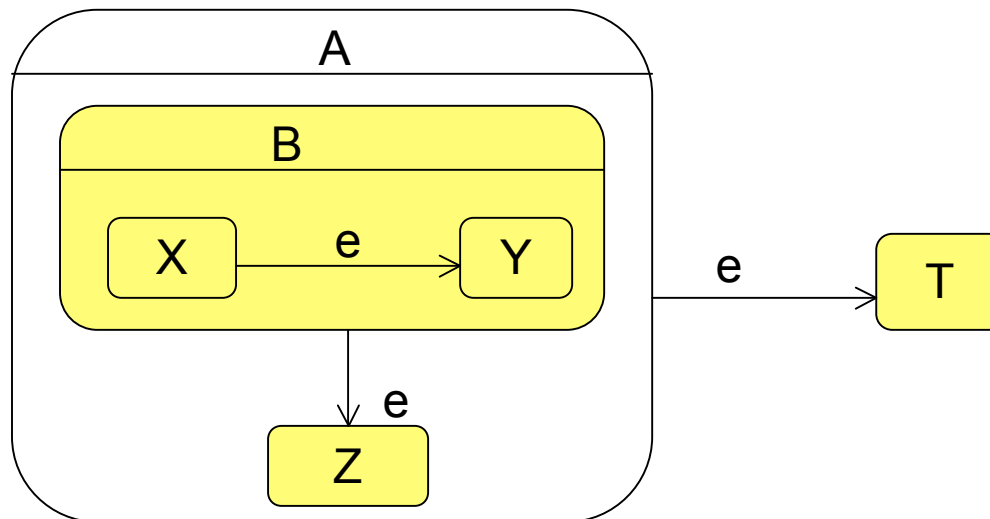




# Etats composites et transitions

Lorsque deux transitions peuvent être effectuées, l'une sur un sous-état et l'autre sur l'état-englobant, c'est la transition sur le sous-état (la plus « spécialisée ») qui est effectuée.

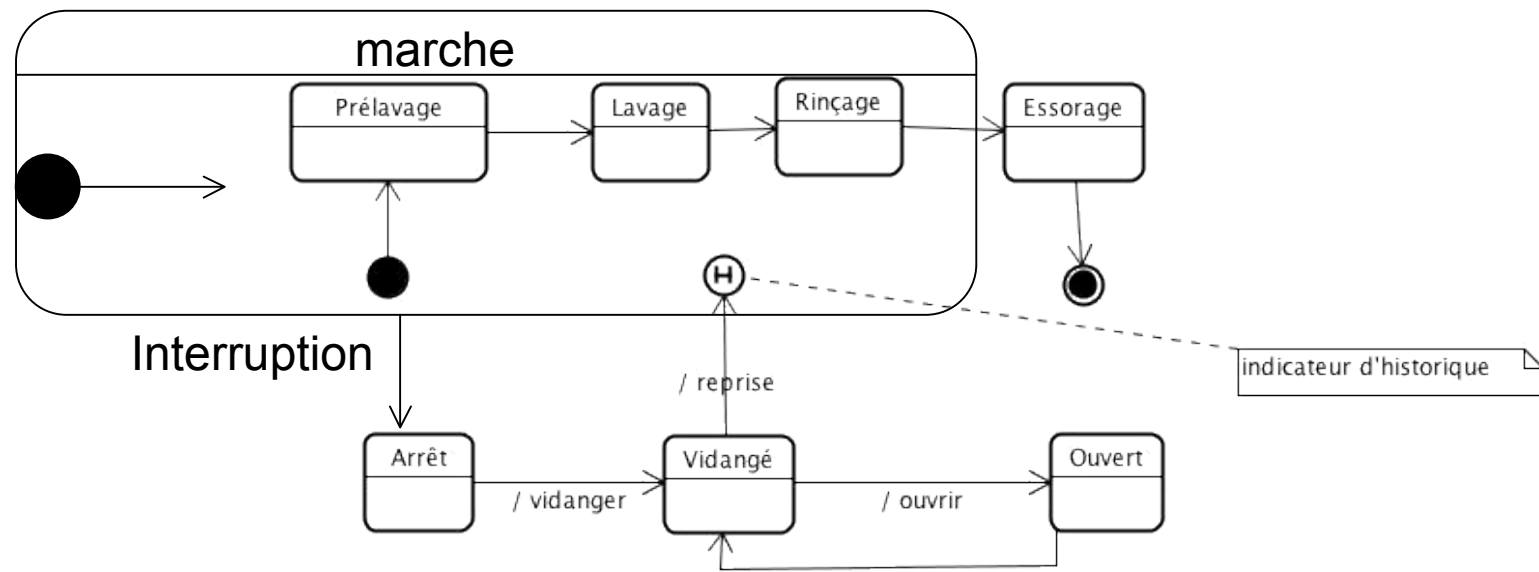
Exemple : si l'objet est dans l'état X, et l'événement e survient, alors l'objet passe dans l'état Y et non Z ou T.



# Indicateurs d'historique

L'indicateur d'historique, noté « **H** », est un pseudo-état permettant de mémoriser le dernier état visité d'un automate pour y retourner ultérieurement.

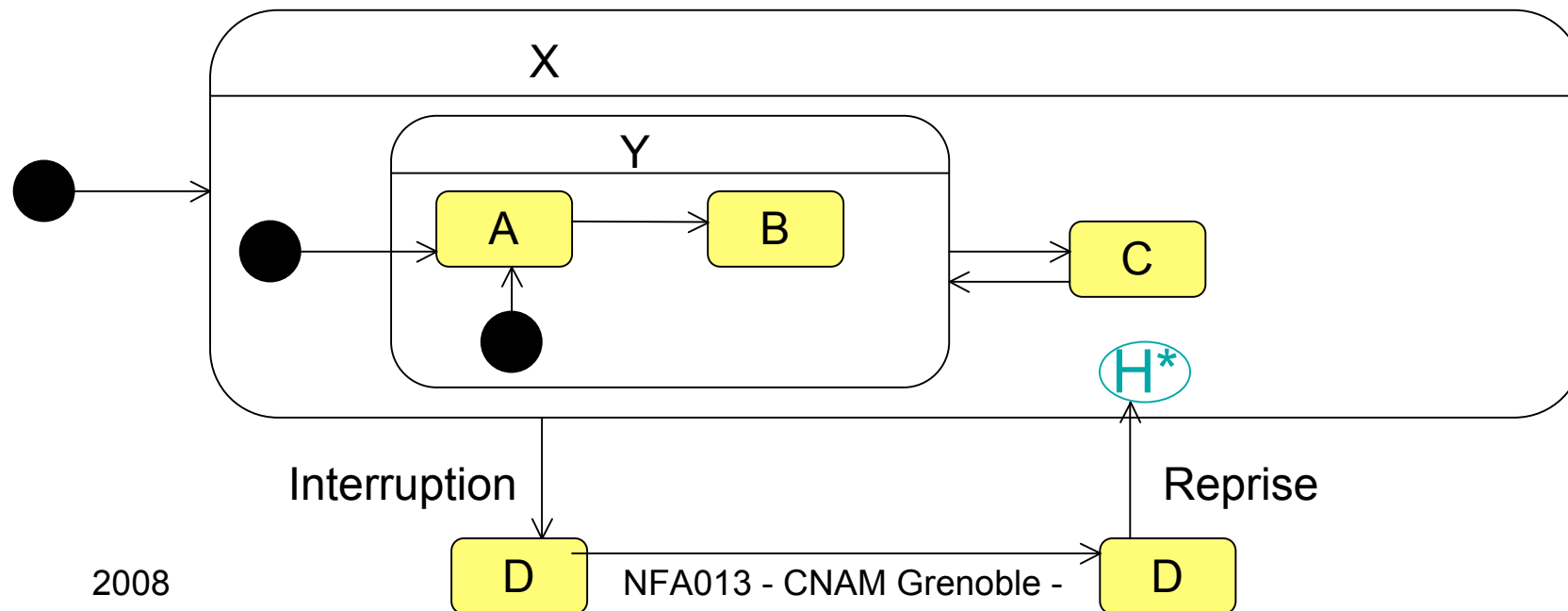
Exemple : la machine à laver. L'ouverture de la porte de la machine en marche nécessite la pause du programme, et la vidange. Une fois la porte refermée, la machine reprend normalement.



# Indicateurs d'historique

L'indicateur d'historique à un niveau quelconque, noté « $H^*$ », est un pseudo-état permettant de mémoriser le dernier état visité d'un automate pour y retourner ultérieurement, à un niveau d'imbrication quelconque.

Exemple : lorsque la reprise est effectuée, l'automate revient dans l'état où il était lors de l'interruption : A, B, ou C



# Automates en parallèles

---

A l'intérieur d'un état, plusieurs automates peuvent s'exécuter en parallèle. Chaque sous-automate a un état initial et un certain nombre d'états terminaux.

L'activité d'un tel état se termine lorsque tous les sous-automates parviennent à un état final

Lorsqu'un événement se produit, toutes les transitions possibles (sensibles à cet événement) sont effectuées.

Cas d'utilisation : objet formé de la composition ou de l'agrégation d'autres objets, en particulier, si le comportement du composite correspond à la mise en parallèle des comportements des composants.

# Exemple d'automates parallèles

