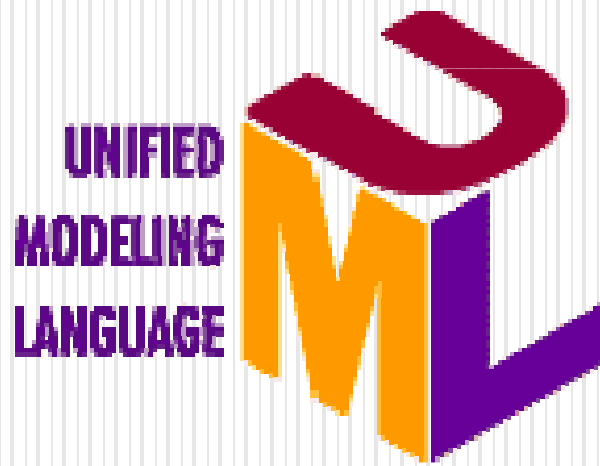
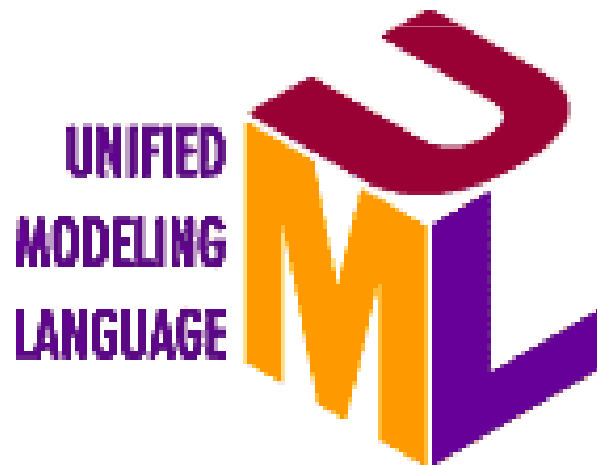


# Analyse : modèles dynamiques



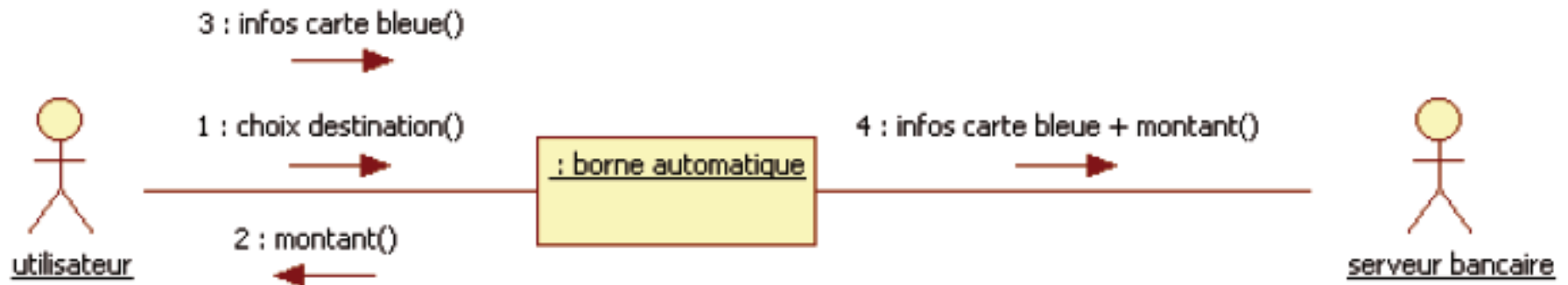
# Les diagrammes de contexte dynamique



## Le diagramme de contexte dynamique

- Le diagramme de contexte dynamique permet de positionner le système dans son environnement selon le point de vue des communications.
- Il reprend les éléments du contexte statique et précise les échanges d'informations qui sont réalisés entre le système et les éléments matériels extérieurs au système.

Ex. : Le diagramme de contexte dynamique d'une borne automatique permettant d'acheter un billet.



# *Table des matières*



- **Introduction**
- **Le diagramme de séquence**
- **Les cadres d'interaction (UML 2)**
- **Le diagramme de communication**
- **Découvrir les objets du système**

# Analyse : modèle dynamique

## *Objectifs*

1. Introduire les notions propres aux diagrammes d'interactions :
  - Notation
  - Création
  - Les relations avec les diagrammes précédemment construits,
2. UML propose 2 diagrammes:  
Diagramme de séquence: aspects temporels  
Diagramme de communication: représentation spatiale

# Diagramme d'interaction

- **Trois usages :**

- **Analyse des spécifications**

- Description de l'interaction du système avec les acteurs extérieurs

- **Analyse dynamique**

- Description de l'interaction moyennant les objets (de l'analyse) du système

- **Conception dynamique**

- Description de l'interaction moyennant les objets (de la conception) du système

# Diagramme d'interaction

Diagramme de séquences

Diagramme de communication

# Diagramme de séquences

- Décrit la dynamique du système: interaction entre un groupe d'objets en montrant de façon séquentielle, les envois des messages qui interviennent entre les objets



## Le diagramme de séquence

**Deux différents types de diagrammes de séquence :**

- **Diagramme de séquence système au niveau de l'*analyse dynamique***

- ➔ représentation des différents scénarios des cas d'utilisation

- ➔ interaction entre les acteurs et le système (une boîte noire) selon un *ordre chronologique*

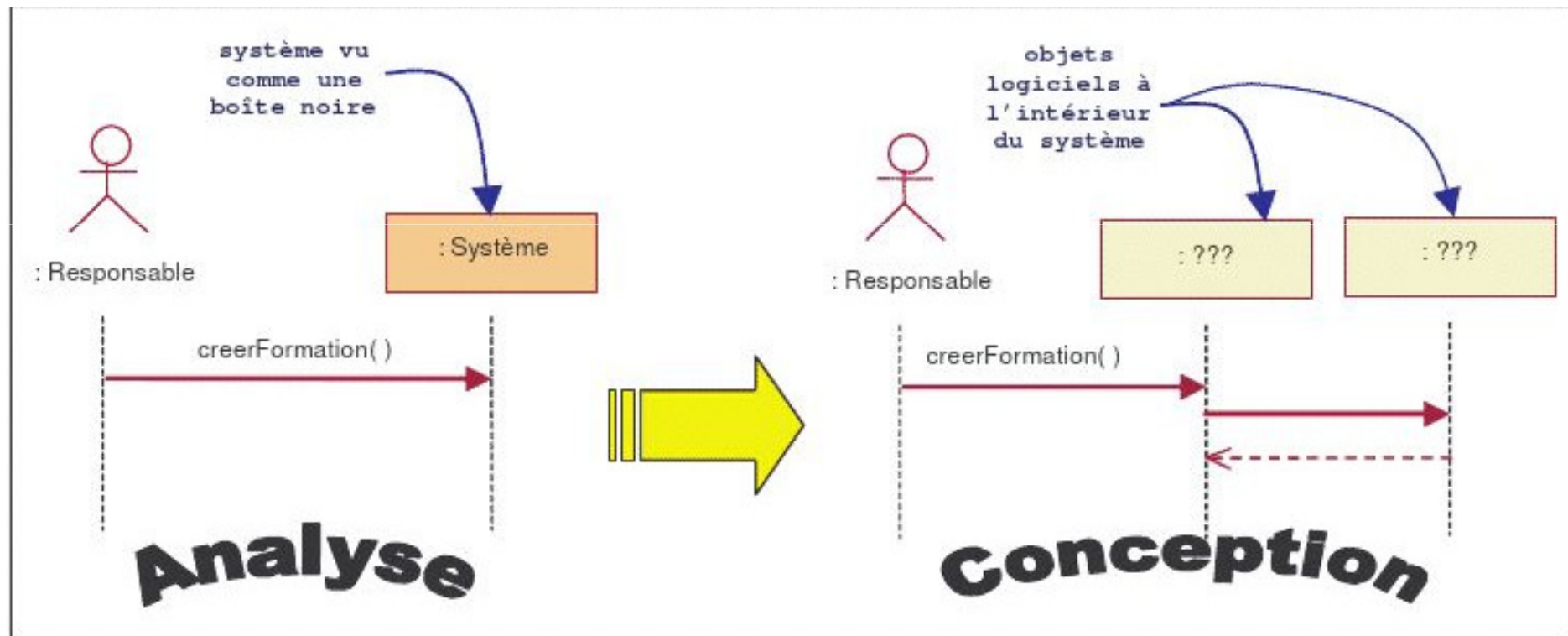
- **Diagramme de séquence détaillé au niveau de la *conception***

- ➔ représentation des *collaborations* entre des objets pour la réalisation d'un cas d'utilisation, selon un point de *vue temporel*.

- ➔ représentation de la chronologie des envois de messages entre les objets.

## Le diagramme de séquence

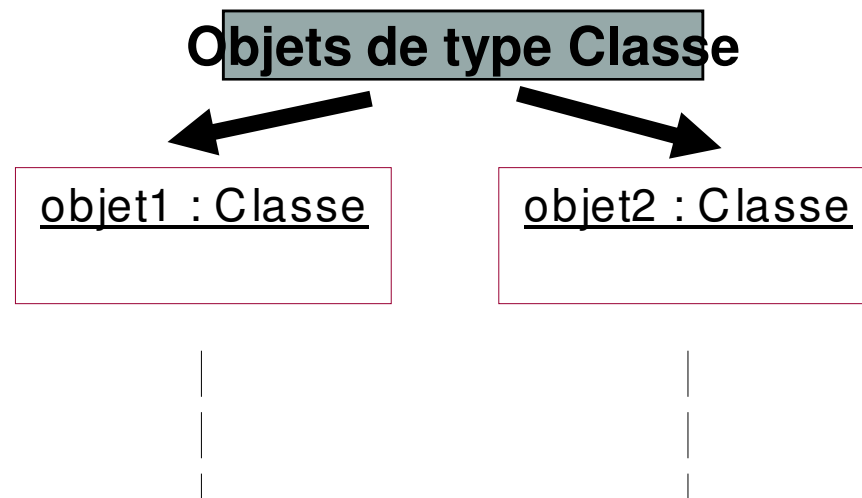
### Du diagramme de séquence système à la conception



## Le diagramme de séquence

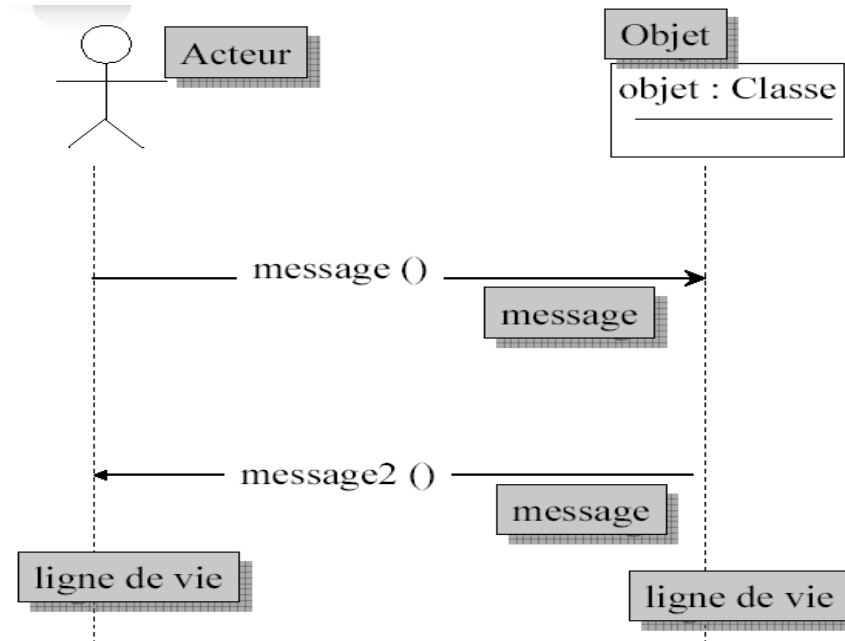
### Représentation de Scénarios

- Il y a autant de diagrammes de séquence qu'il y a de scénarios
- Un Scénario montre une séquence particulière d'interactions entre objets, dans un seul contexte d'exécution du système
- Un scénario peut être vu comme une réponse à un besoin ou une partie d'un besoin du diagramme des Uses Cases.
- On y fait intervenir des objets / système complet, des messages et des événements



## Le diagramme de séquence

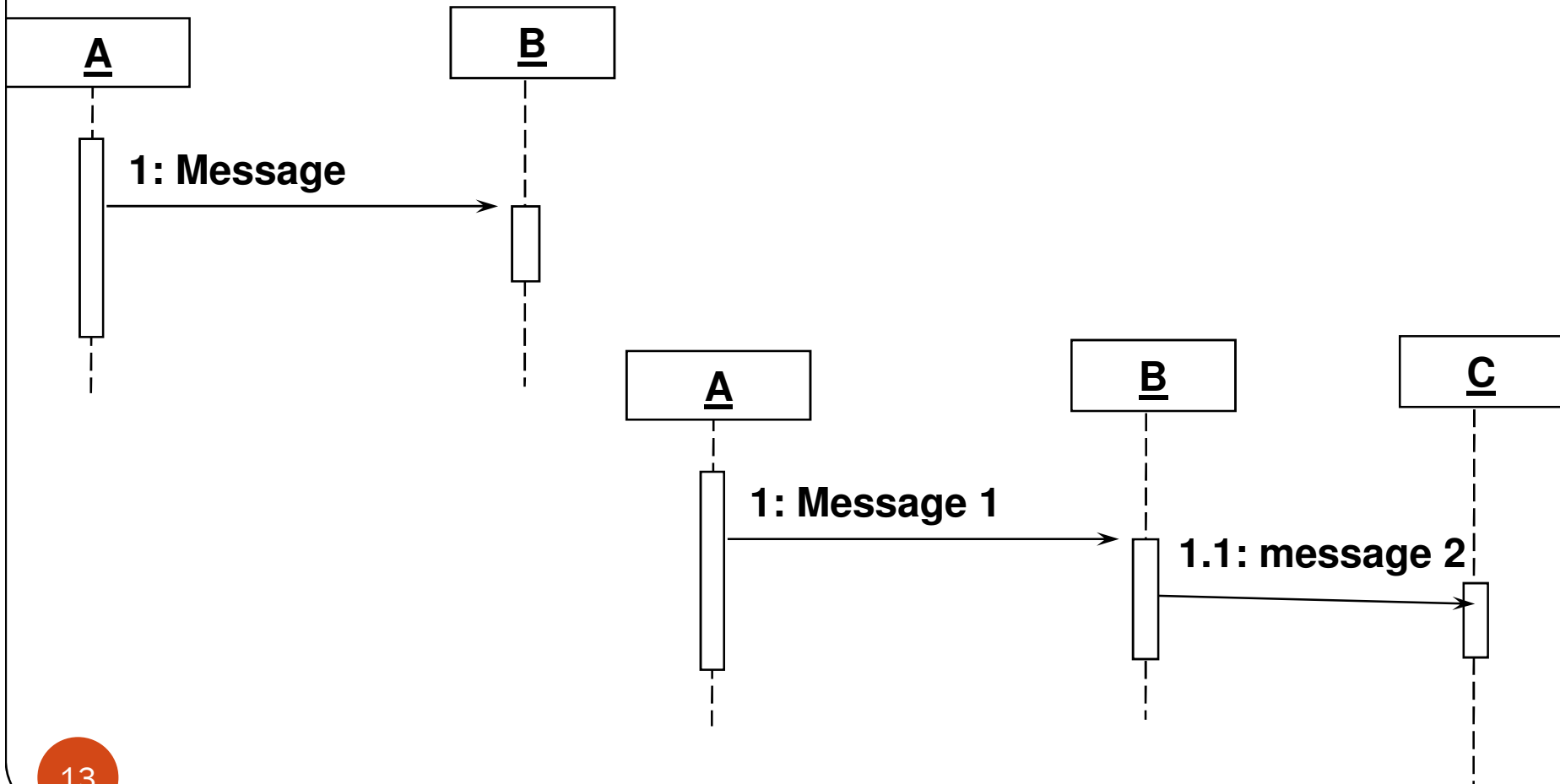
✓ l'ordre d'envoi d'un message est déterminé par sa position sur l'axe vertical du diagramme ; le temps s'écoule "de haut en bas" de cet axe : La ligne de vie. La disposition des objets sur l'axe horizontal n'a pas de conséquence pour la sémantique du diagramme.



## Le diagramme de séquence

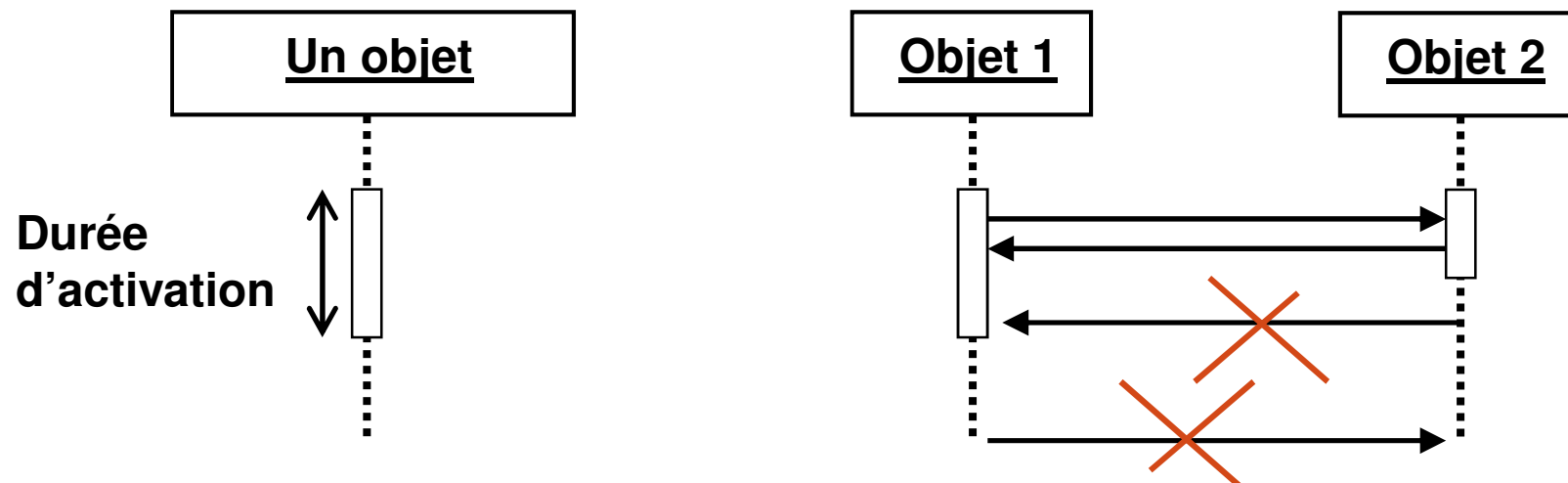
### Numérotation des messages

- Les messages peuvent être numérotés séquentiellement



# Durée d'activation (période d'activation)

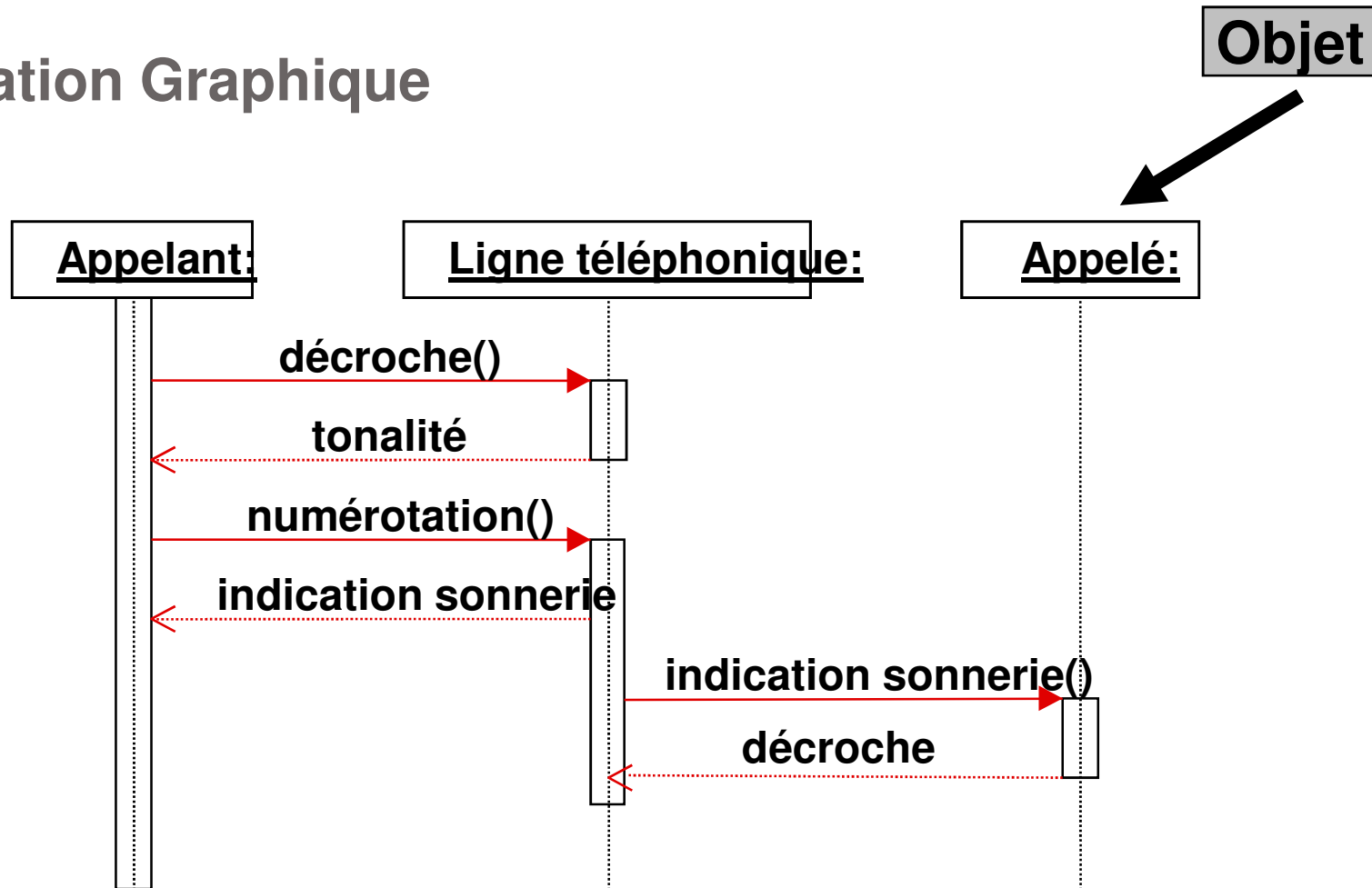
- Correspond au temps pendant lequel un objet effectue une action
- est représentée par une bande rectangulaire :
  - Le long de la ligne de vie de l'objet
  - Dont les extrémités représentent le début et la fin de l'activité.



**Remarque :** souvent quand l'objet est en activité continue, on néglige la représentation de cette période d'activation

## Le diagramme de séquence

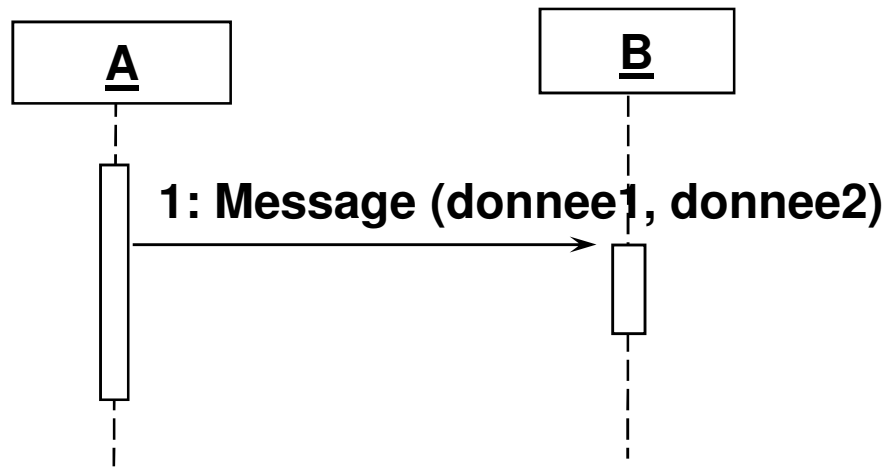
### Notation Graphique



## Le diagramme de séquence

### Transmission de données

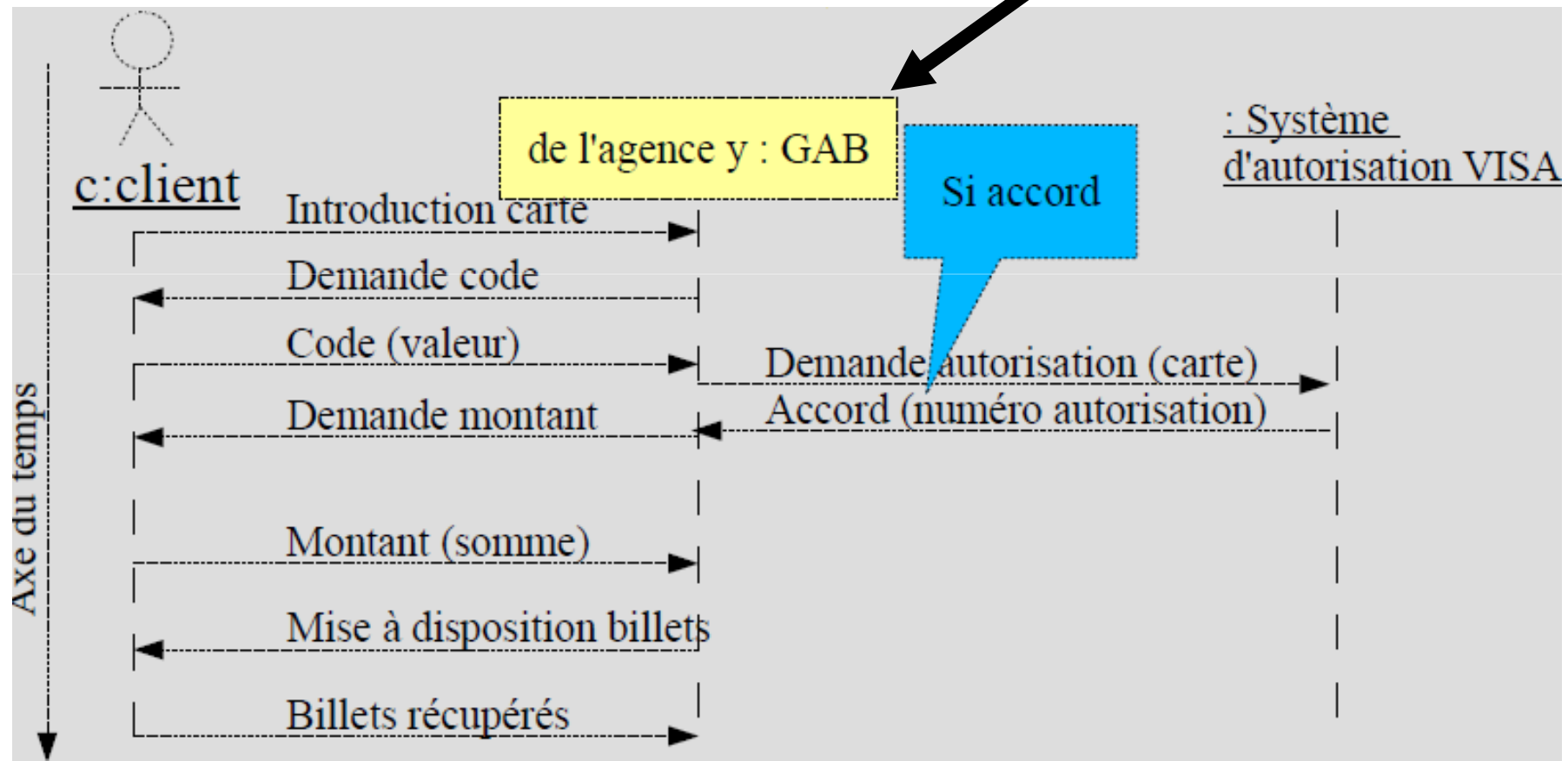
- Les messages peuvent véhiculer des données entre les acteurs et le système ou entre les objets






## Le diagramme de séquence

Représentation graphique Système comme une boîte noire



# Types de messages

**synchrone:** 

- L'expéditeur du message attend que l'activation de la méthode invoquée chez le destinataire soit terminée avant de continuer son activité

**asynchrone:** 

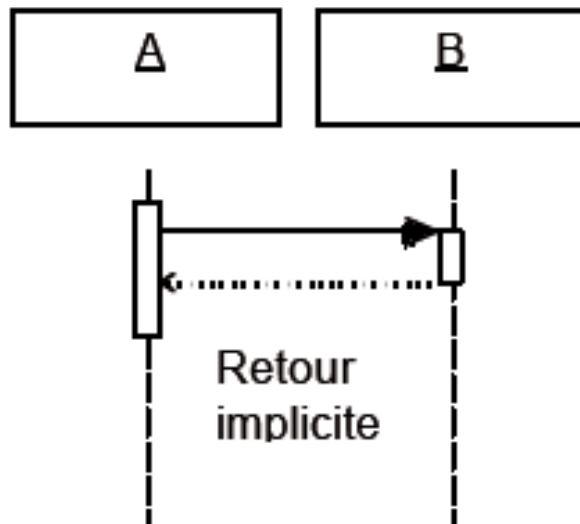
- L'expéditeur n'attend pas la fin de l'activation de la méthode invoquée chez le destinataire: système multi-thread

**Message de retour:** 

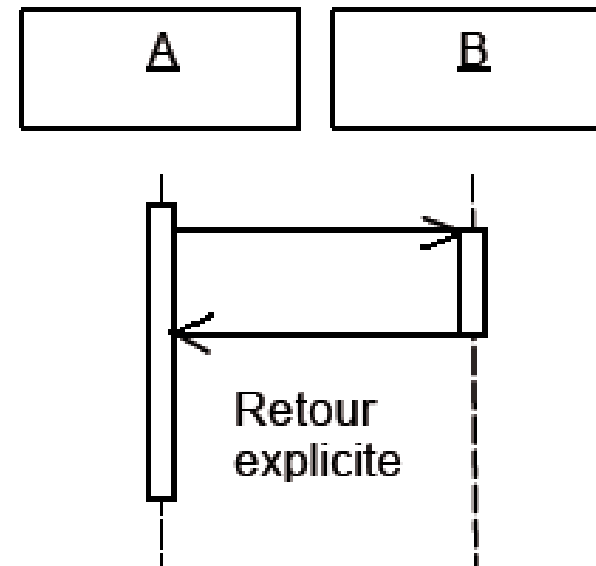
## Le diagramme de séquence

### Les réponses aux messages

**Dans le cas des envois synchrones, le retour peut être implicite en fin d'activités et ne nécessitera pas de représentation particulière**

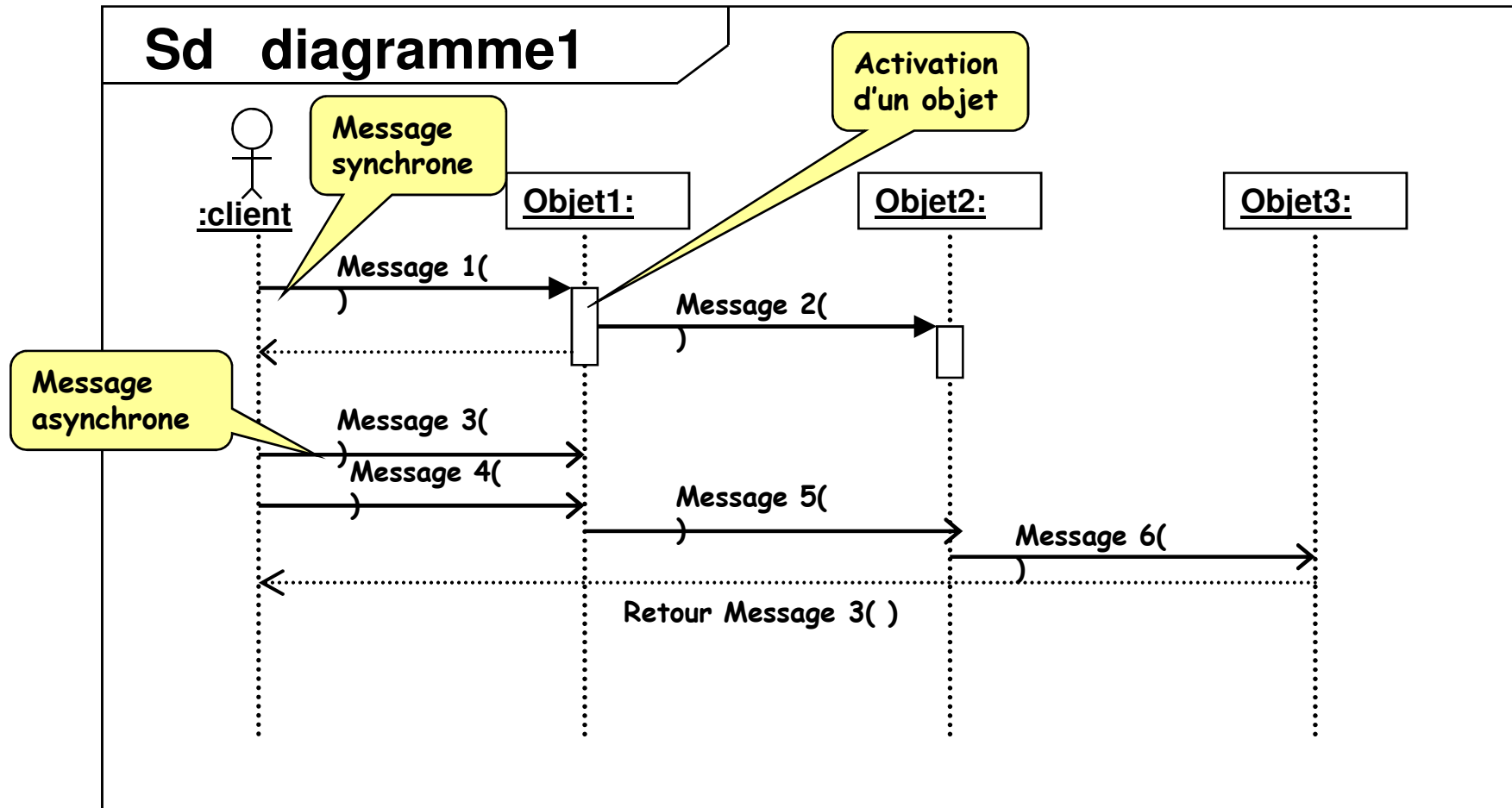


**Dans le cas des envois asynchrones, le retour doit être explicite**



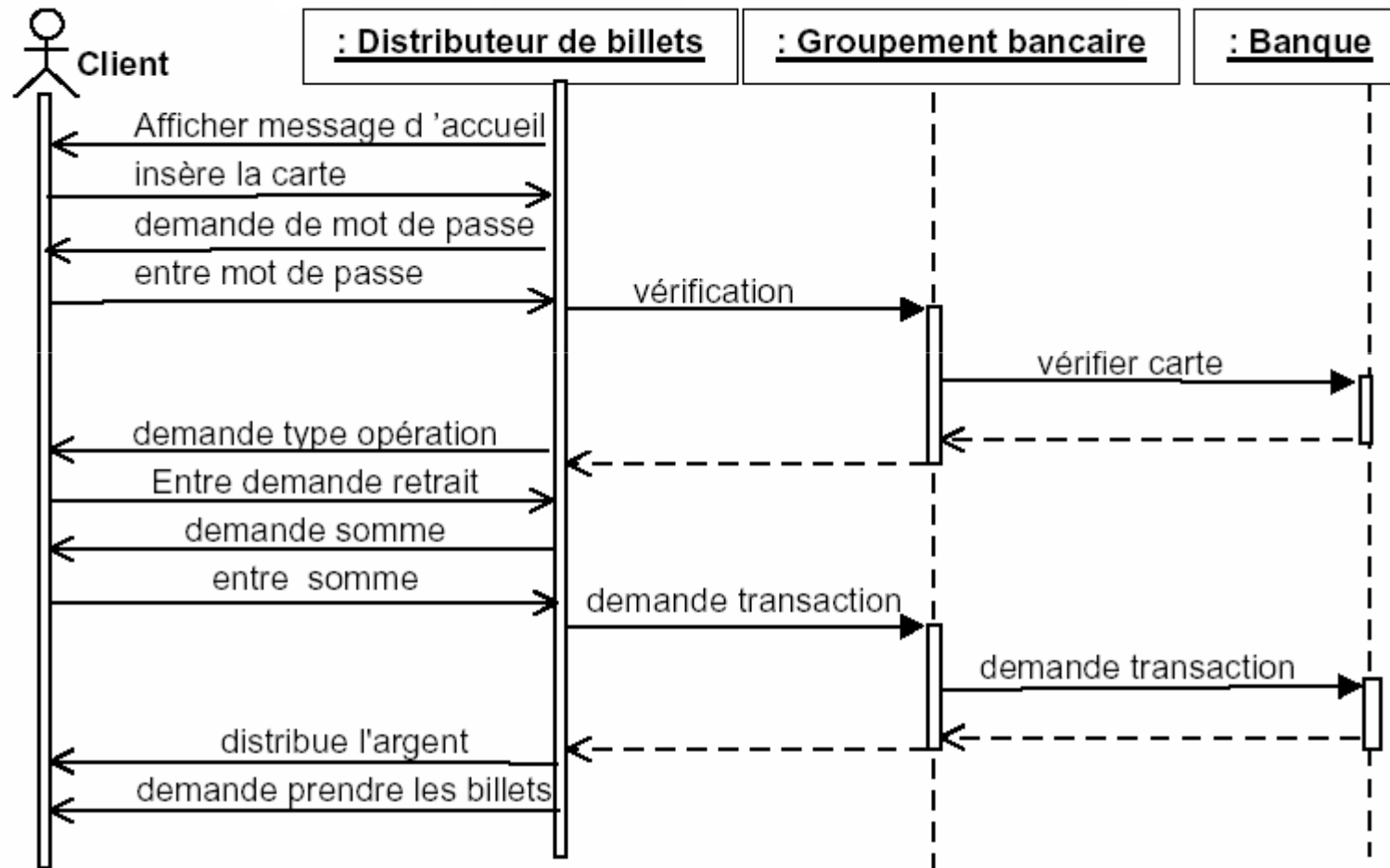
# Le diagramme de séquence

## Les types de messages



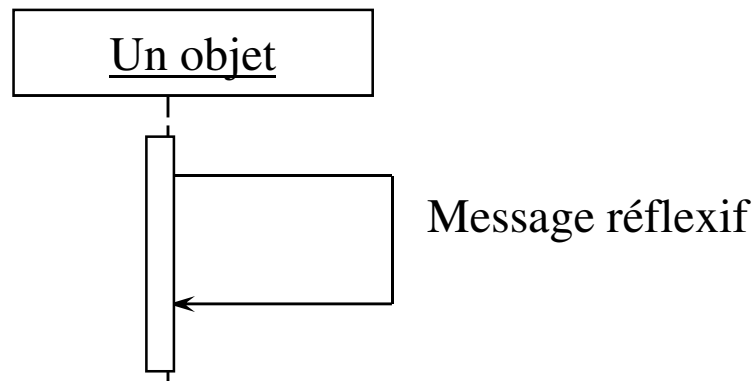
# Le diagramme de séquence

## Exemple



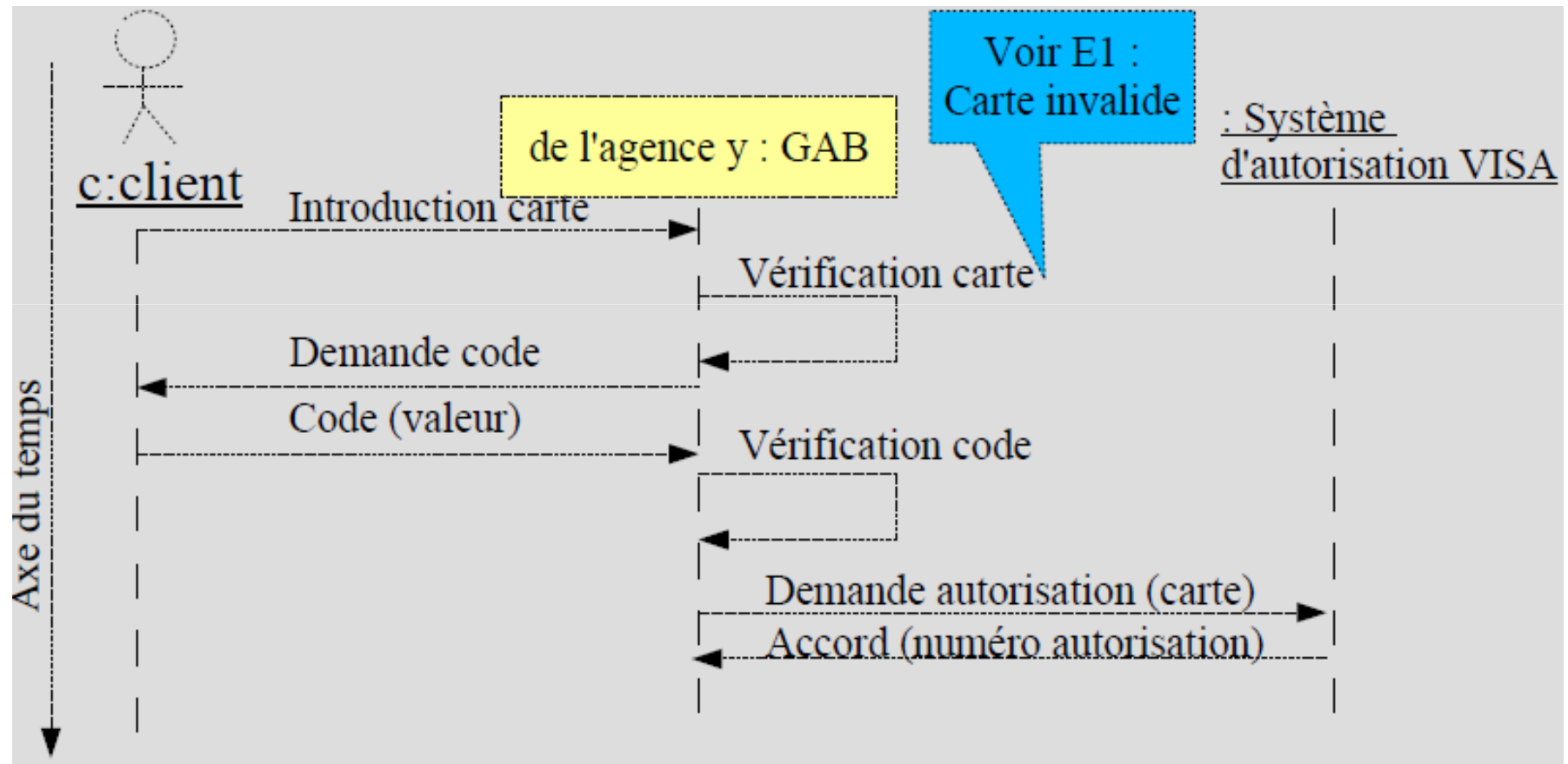
# Envoi d'un message à soi-même

Un objet peut s'envoyer un message à lui-même



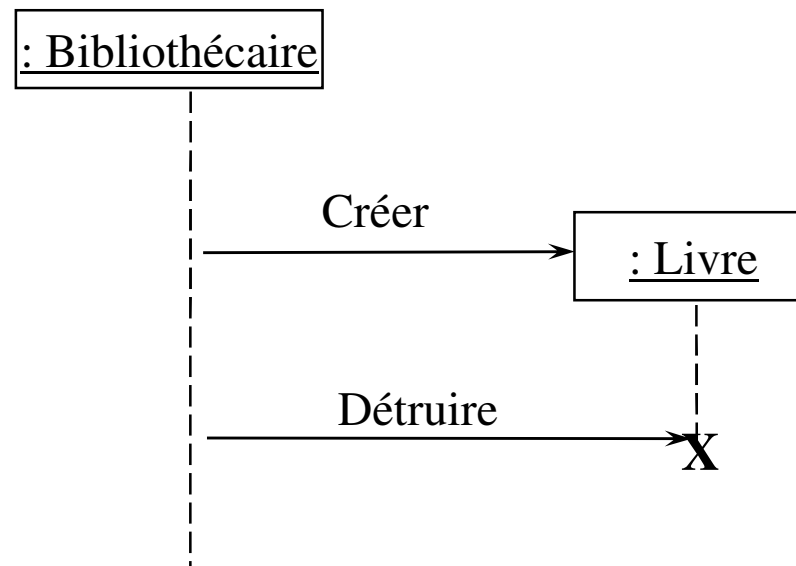
## Le diagramme de séquence

### Représentation graphique enrichie



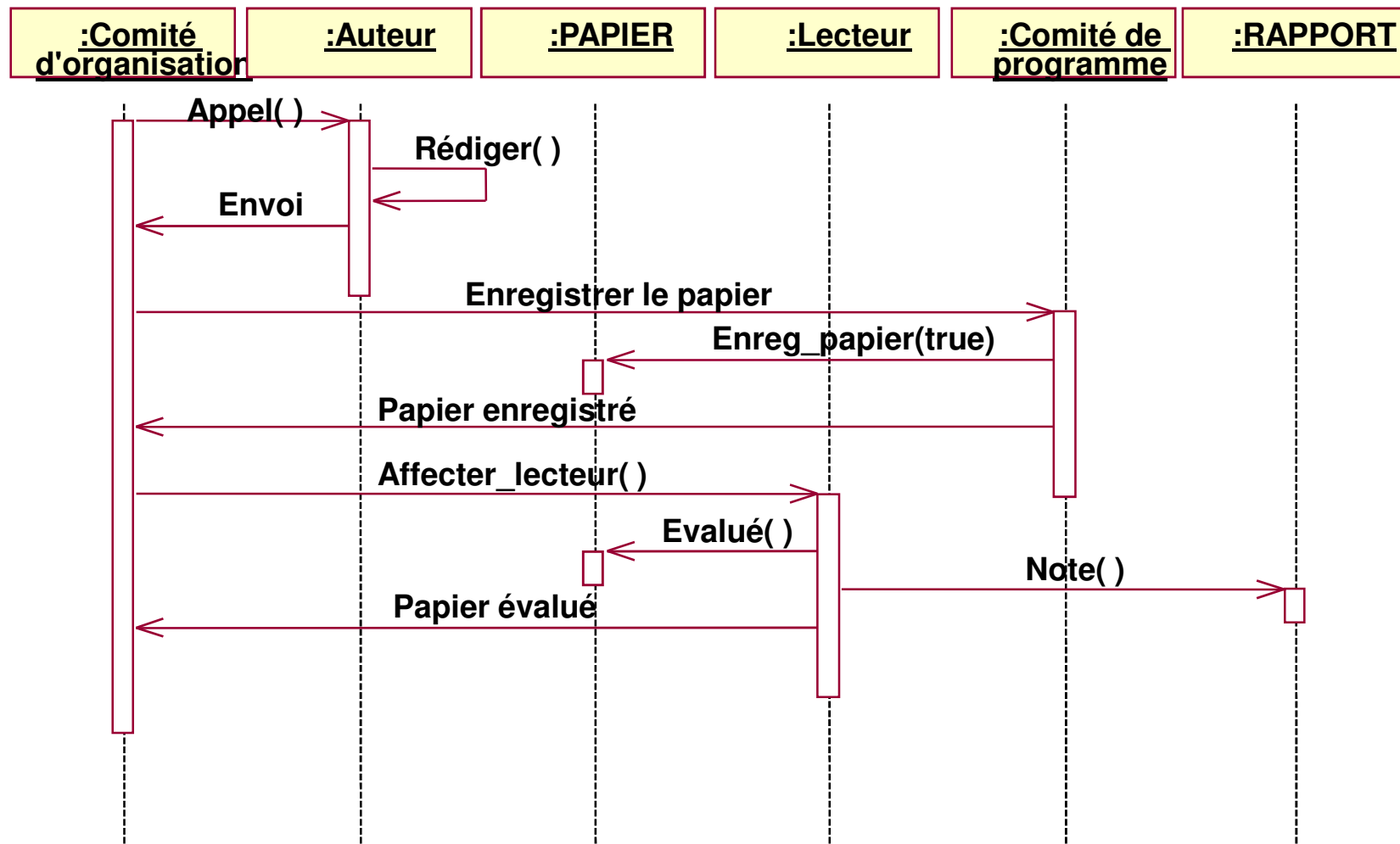
# Création et destruction d'objet

- La création d'un objet se représente par un message pointant sur l'objet crée
- La destruction est indiquée par la fin de la ligne de vie et par la lettre X



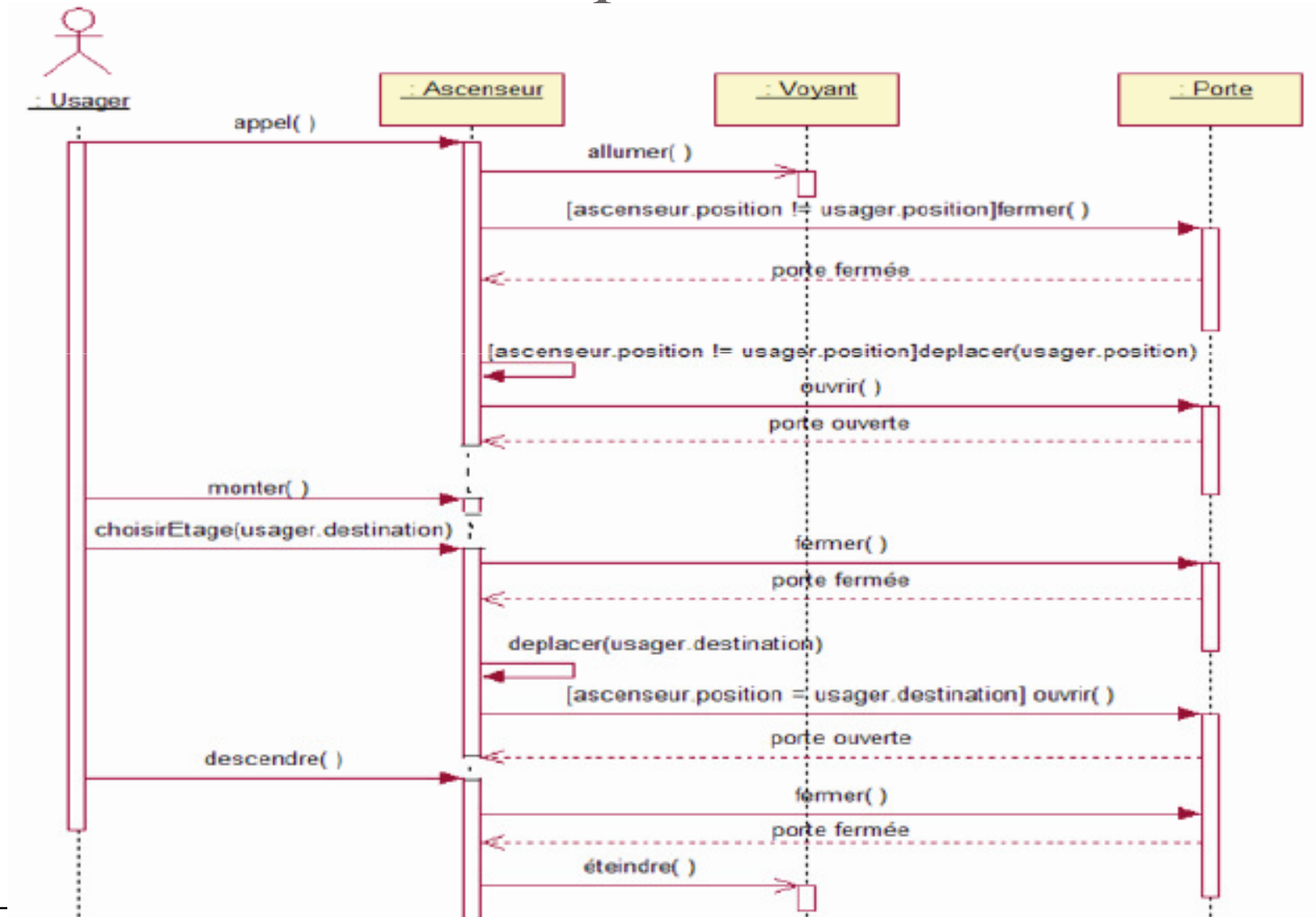


# La description de la dynamique: scénario



## Le diagramme de séquence

### Exemple3



## Le diagramme de séquence

### Autres types de messages

Dans un diagramme de séquence, il peut y avoir d'autres types de messages tels que:

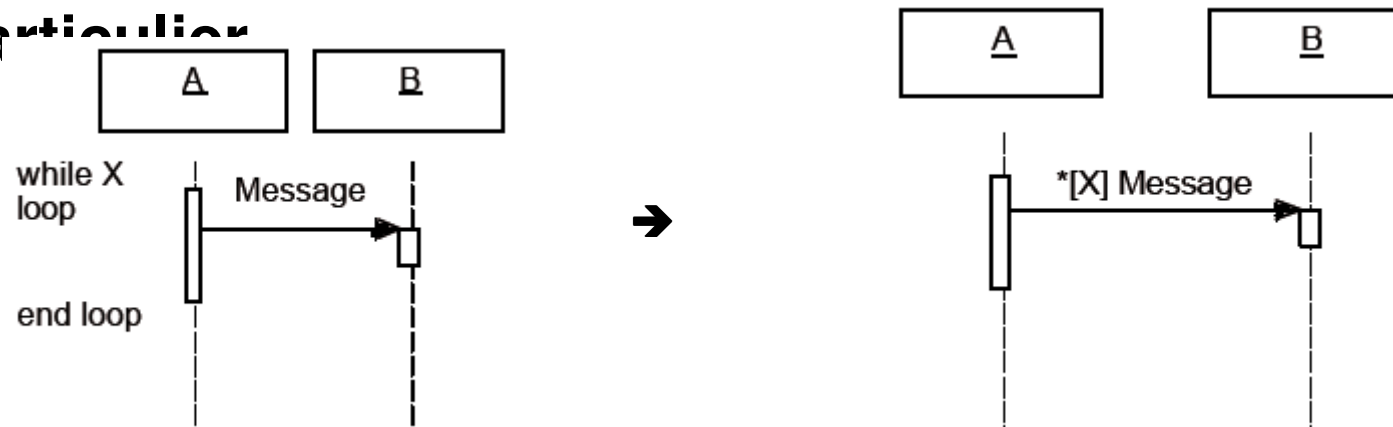
- **message minuté (timeout) :** Bloque l'expéditeur pendant un temps donné (qui peut être spécifié dans une contrainte), en attendant la prise en compte du message par le récepteur. L'expéditeur est libéré si la prise en compte n'a pas eu lieu pendant le délai spécifié.
- **message dérobant :** Le message est mis en attente dans une liste d'attente de traitement chez le récepteur.

## Le diagramme de séquence

### Pseudo-code

L'ajout de pseudo-code sur la partie gauche du diagramme permet la représentation des boucles et des branchements alternatifs

→ les diagrammes de séquence peuvent représenter la forme générale d'une interaction, au-delà de la seule prise en compte d'un scénario particulier



# Les cadres d'interaction (UML2)

## Le diagramme de séquence

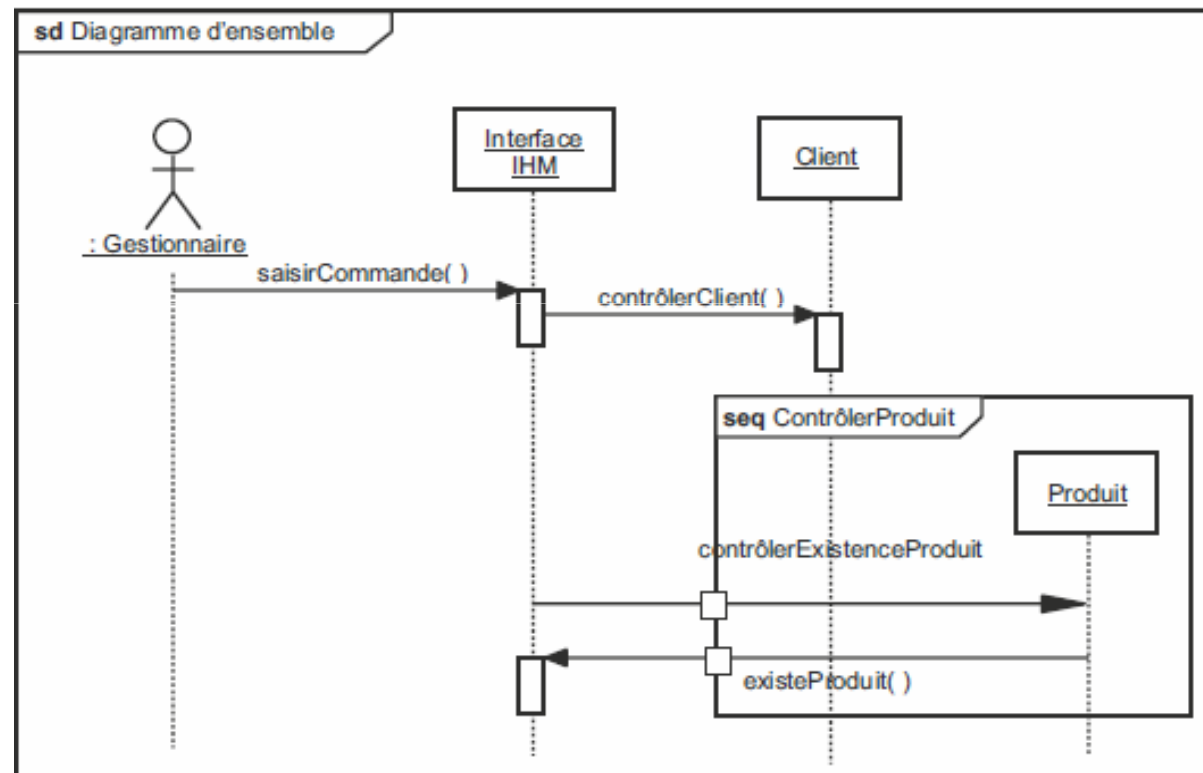
### Diagramme de séquences :

Fragment d'interaction ou fragments combinés

- ❑ Un fragment d'interaction se représente globalement comme un diagramme de séquence dans un rectangle avec indication dans le coin à gauche du nom du fragment.
- ❑ Un port d'entrée et un port de sortie peuvent être indiqués pour connaître la manière dont ce fragment peut être relié au reste du diagramme. Dans le cas où aucun port n'est indiqué c'est l'ensemble du fragment qui est appelé pour exécution

## Le diagramme de séquence

Fragment d'interaction



## Le diagramme de séquence

### Fragment d'interaction

- ❑ Un fragment d'interaction dit combiné correspond à un ensemble d'interactions auquel on applique un opérateur.
- ❑ Un frag combiné se représente globalement comme un diagramme de séquence avec indication dans le coin à gauche du nom de l'opérateur.
- ❑ 13 opérateurs ont été définis dans UML : alt, opt, loop, par, strict/weak, break, ignore/consider, critical, negative, assertion et ref.

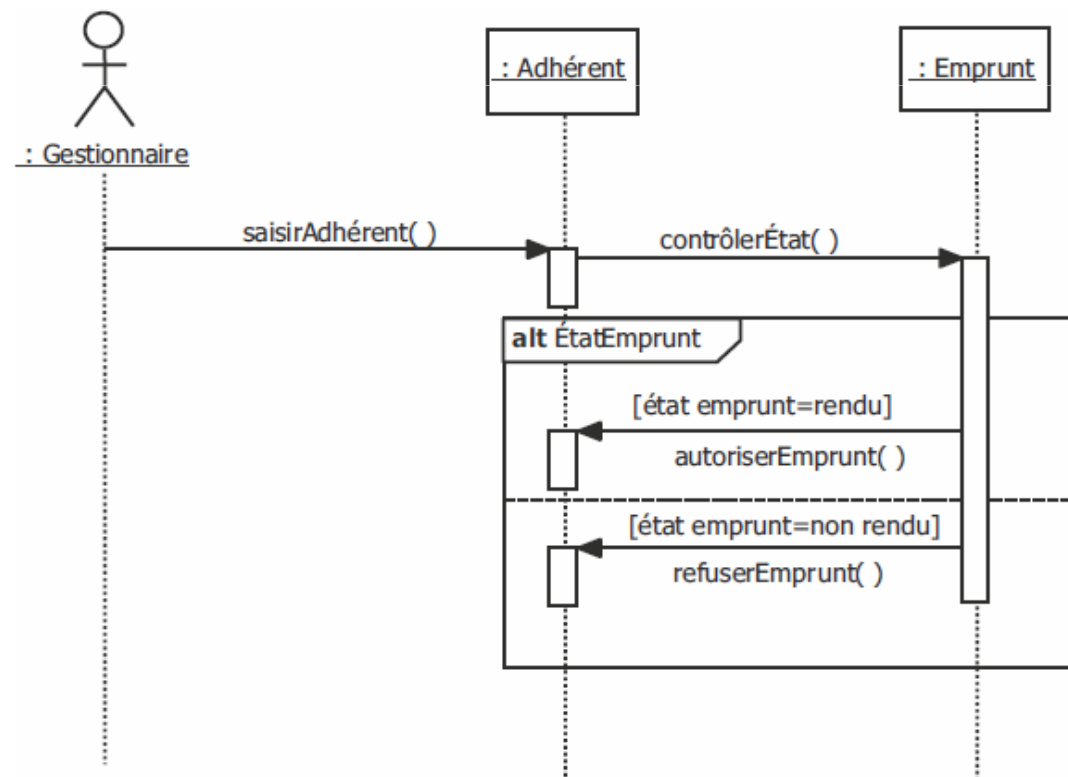


## Le diagramme de séquence

Fragment d'interaction

### ❑ L'opérateur *alt* :

correspond à une instruction de test avec une ou plusieurs alternatives possibles. Il est aussi permis d'utiliser les clauses de type sinon.

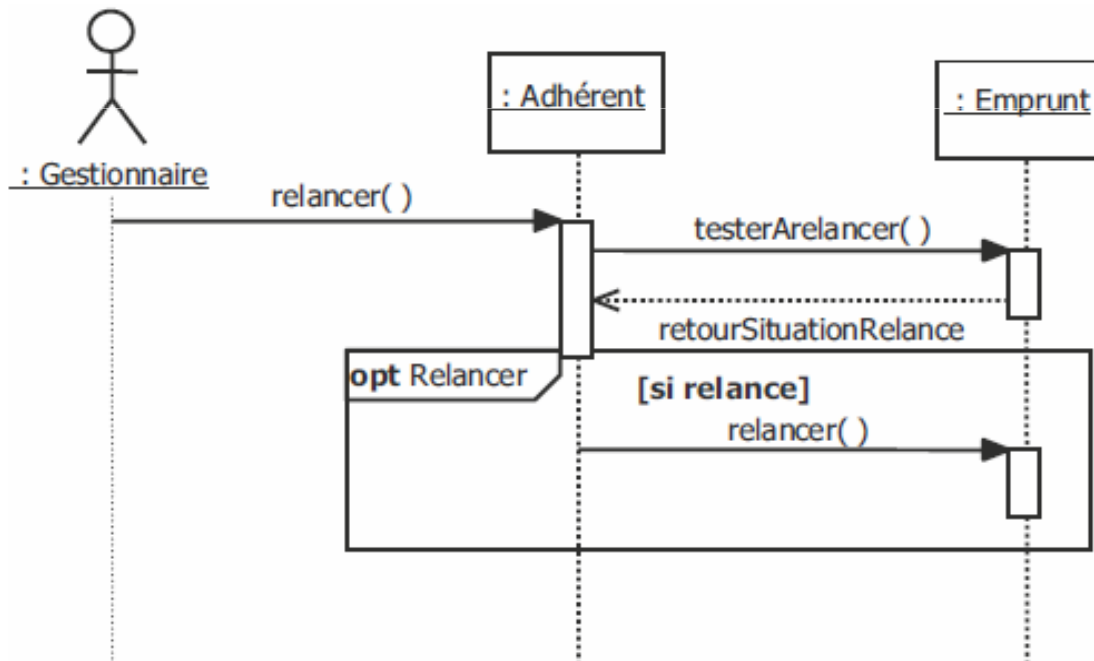


## Le diagramme de séquence

Fragment d'interaction

### ❑ L'opérateur *opt* :

L'opérateur *opt* (optional) correspond à une instruction de test sans alternative (sinon).



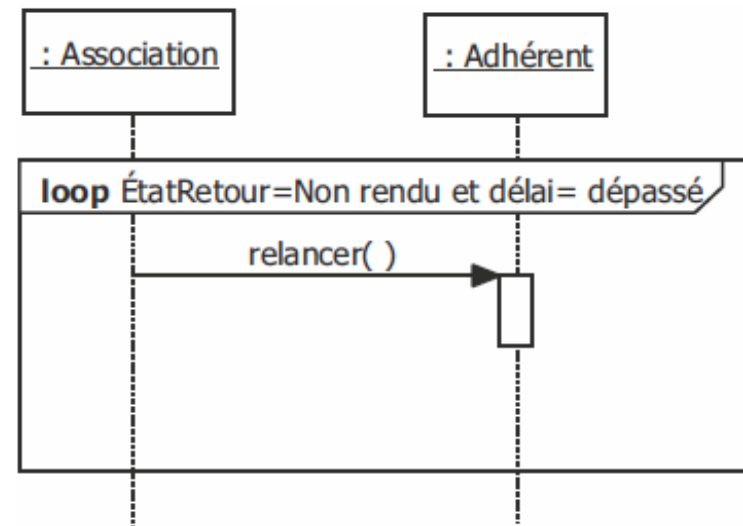
## Le diagramme de séquence

### Fragment d'interaction

#### ❑ L'opérateur *loop*

correspond à une instruction de boucle qui permet d'exécuter une séquence d'interaction tant qu'une condition est satisfaite.

Il est possible aussi d'utiliser une condition portant sur un nombre minimum et maximum d'exécution de la boucle en écrivant : loop min, max. Dans ce cas, la boucle s'exécutera au minimum min fois et au maximum max fois

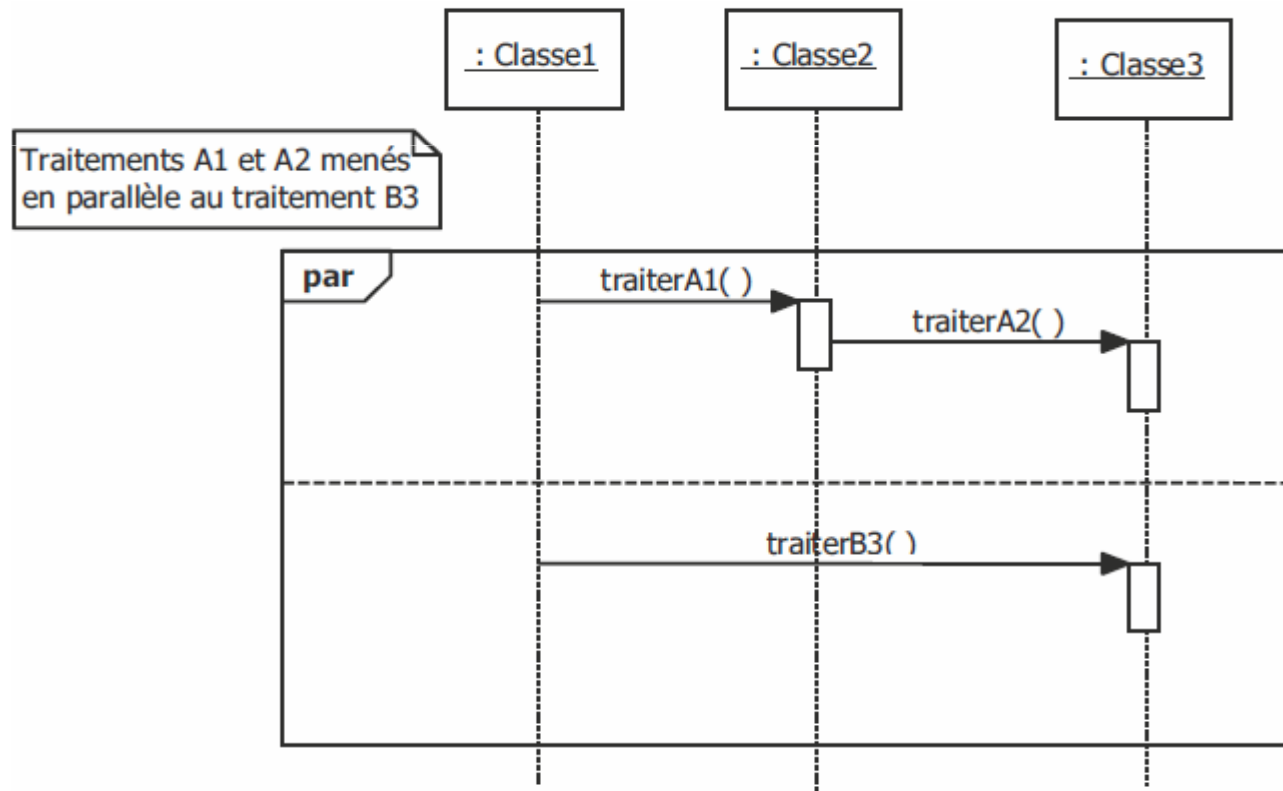


## Le diagramme de séquence

Fragment d'interaction

### ❑ L'opérateur *par*

L'opérateur *par* (parallèle) permet de représenter deux séries d'interactions qui se déroulent en parallèle.



## Le diagramme de séquence

Fragment d'interaction

### ❑ Les opérateurs *strict* et *weak sequencing*

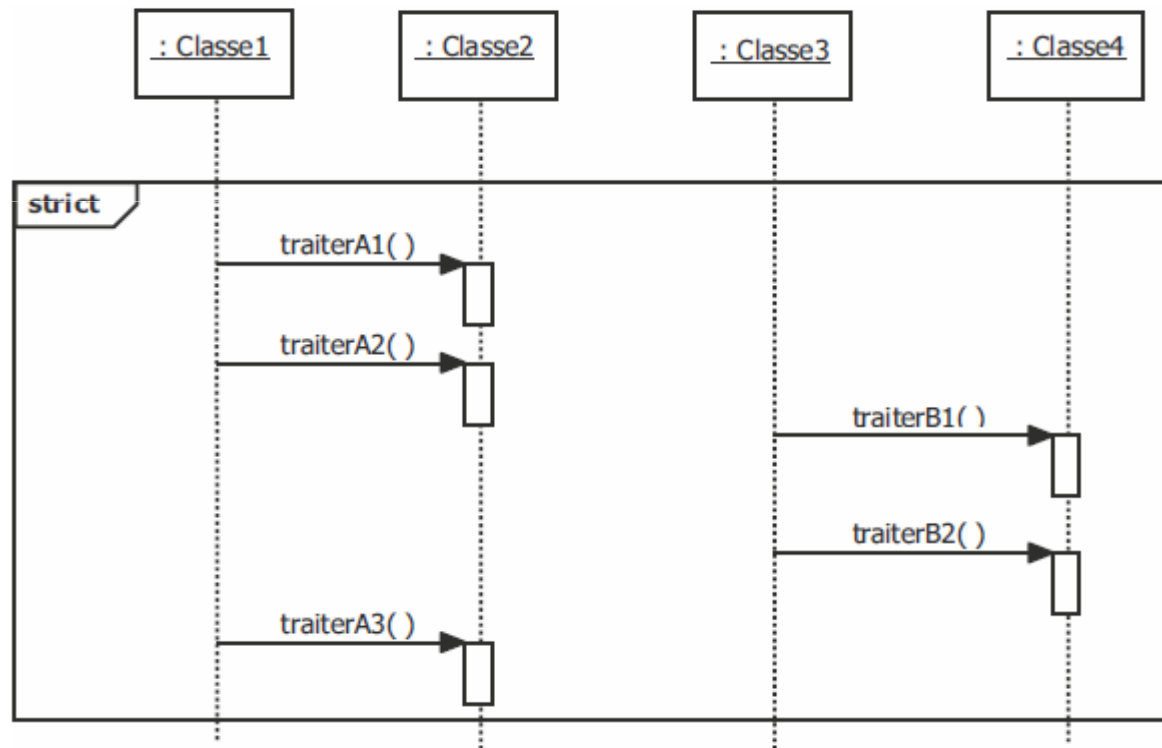
Les opérateurs strict et weak permettent de représenter une série d'interactions dont certaines s'opèrent sur des objets indépendants :

- L'opérateur strict est utilisé quand l'ordre d'exécution des opérations doit être strictement respecté.
- L'opérateur weak est utilisé quand l'ordre d'exécution des opérations n'a pas d'importance.

## Le diagramme de séquence

Fragment d'interaction

### ❑ Les opérateurs *strict* et *weak sequencing*

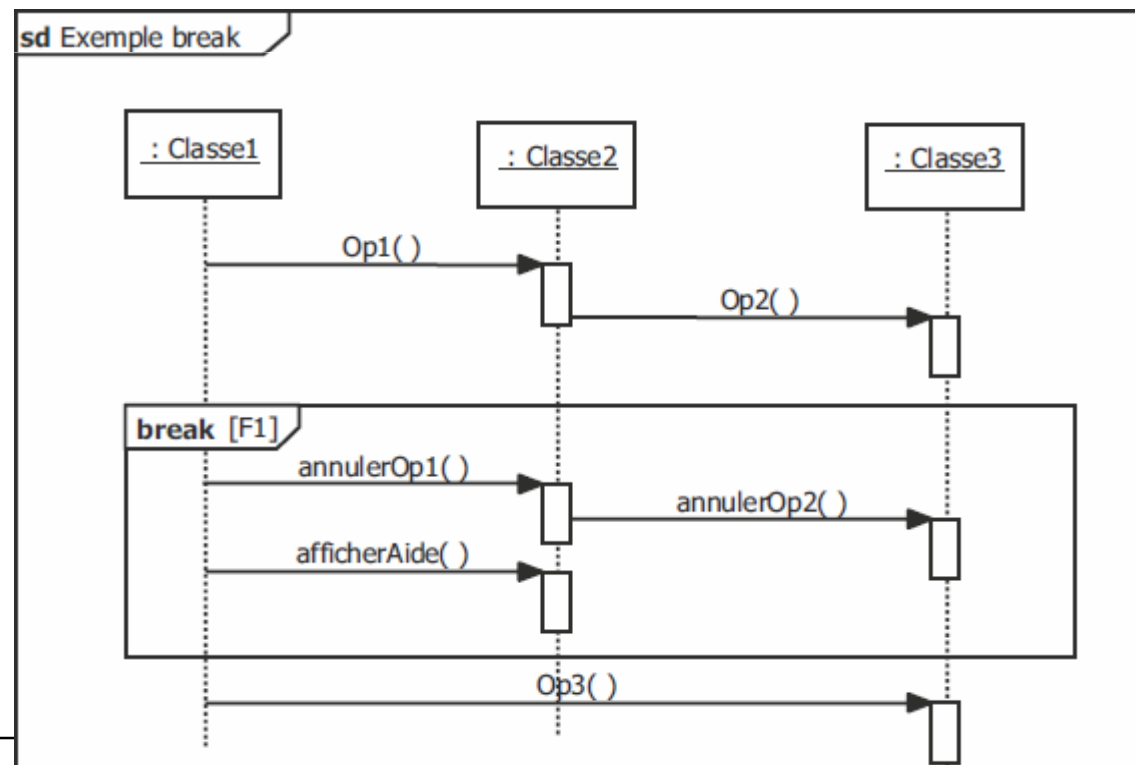


## Le diagramme de séquence

Fragment d'interaction

### ❑ Les opérateurs *break*

L'opérateur break permet de représenter une situation exceptionnelle correspondante à un scénario de rupture par rapport au scénario général. Le scénario de rupture s'exécute si la condition de garde est satisfaite.



## Le diagramme de séquence

Fragment d'interaction

### ❑ Les opérateurs *ignore* et *consider*

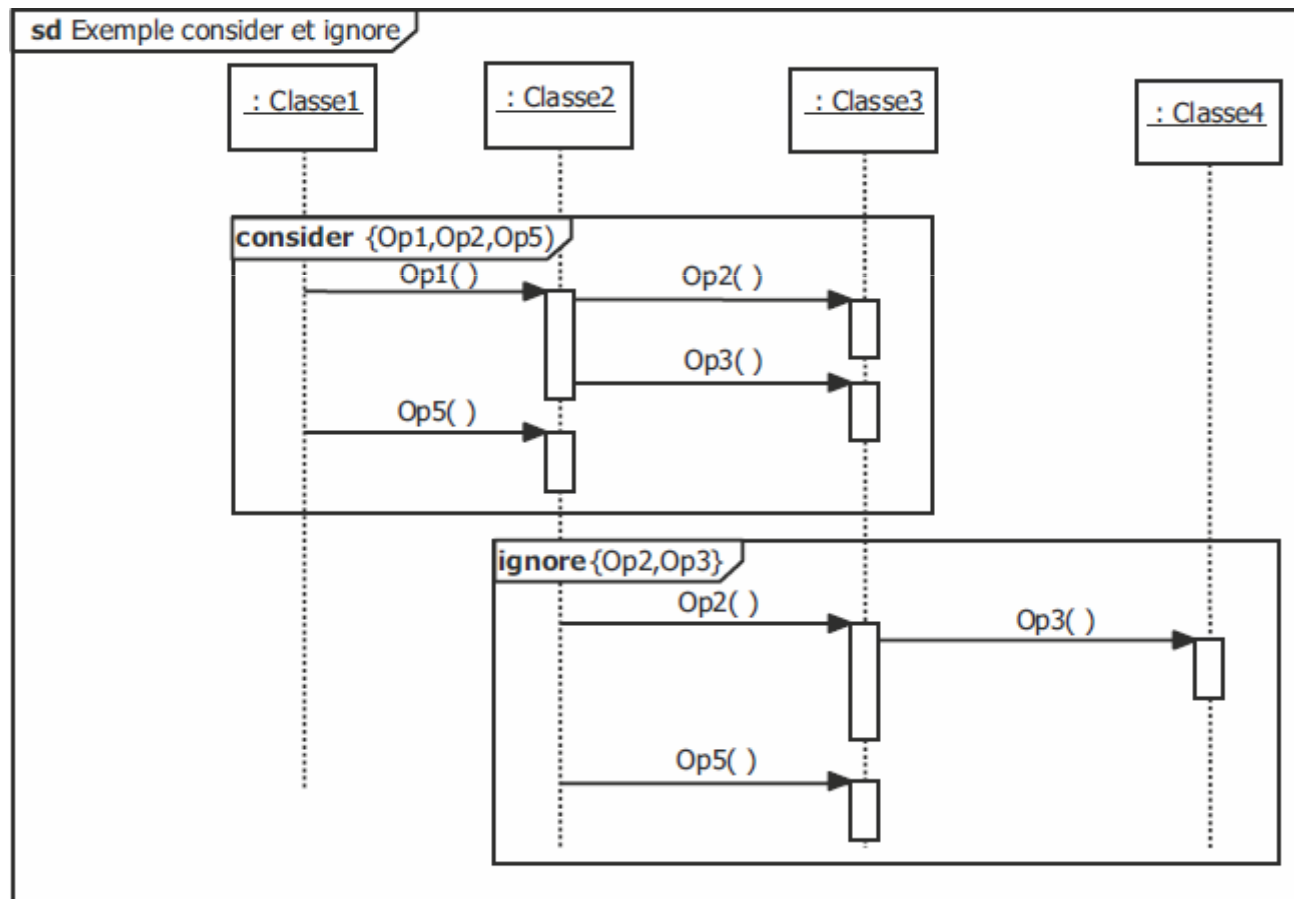
Les opérateurs *ignore* et *consider* sont utilisés pour des fragments d'interactions dans lesquels on veut montrer que certains messages peuvent être soit absents sans avoir d'incidence sur le déroulement des interactions (*ignore*), soit obligatoirement présents (*consider*).



## Le diagramme de séquence

Fragment d'interaction

### ❑ Les opérateurs *ignore* et *consider*



## Le diagramme de séquence

Fragment d'interaction

### ❑ Les opérateurs *critical*

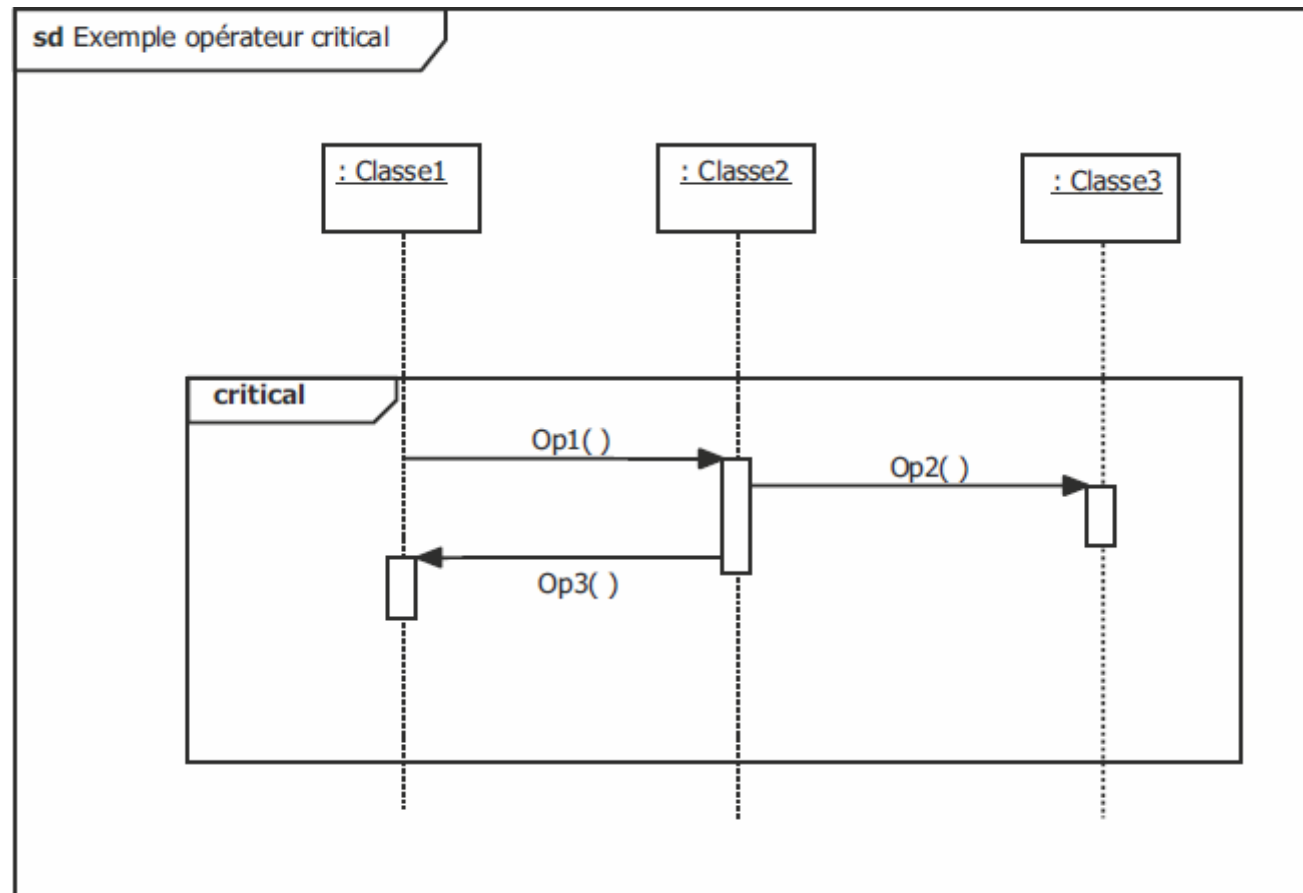
L'opérateur critical permet d'indiquer qu'une séquence d'interactions ne peut être interrompue compte tenu du caractère critique des opérations traitées.

On considère que le traitement des interactions comprises dans la séquence critique est atomique.

## Le diagramme de séquence

Fragment d'interaction

### ❑ L'opérateur *critical*

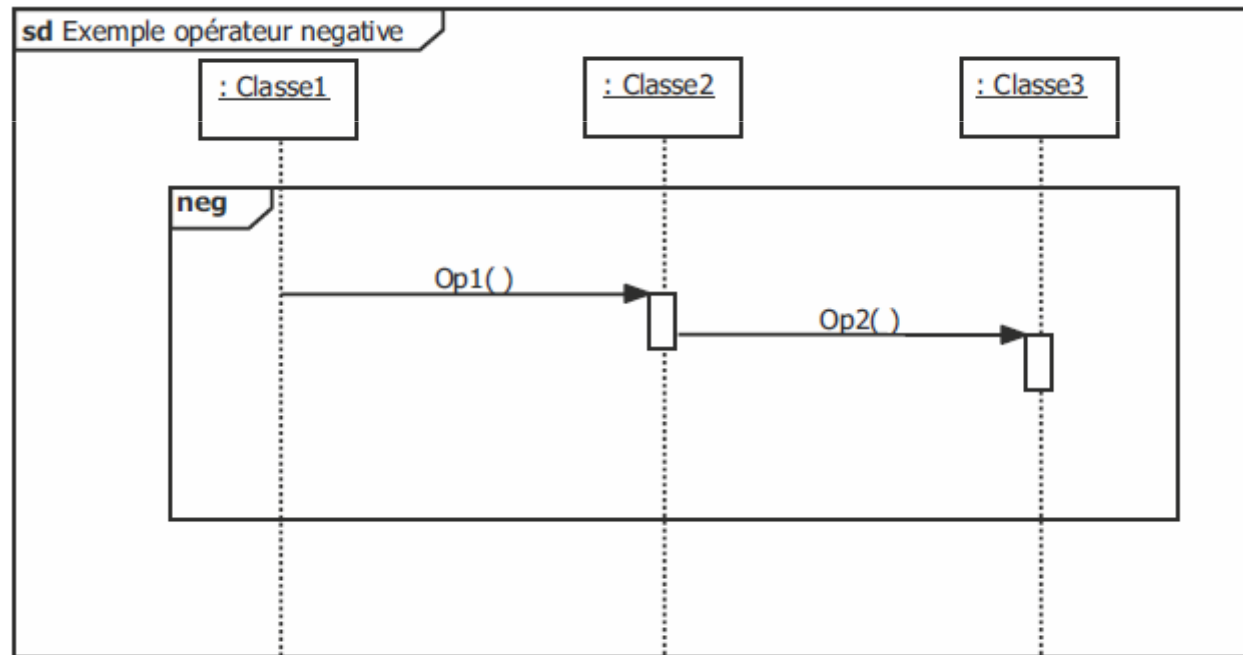


## Le diagramme de séquence

Fragment d'interaction

### ❑ Les opérateurs *neg*

L'opérateur *neg* (negative) permet d'indiquer qu'une séquence d'interactions est invalide.



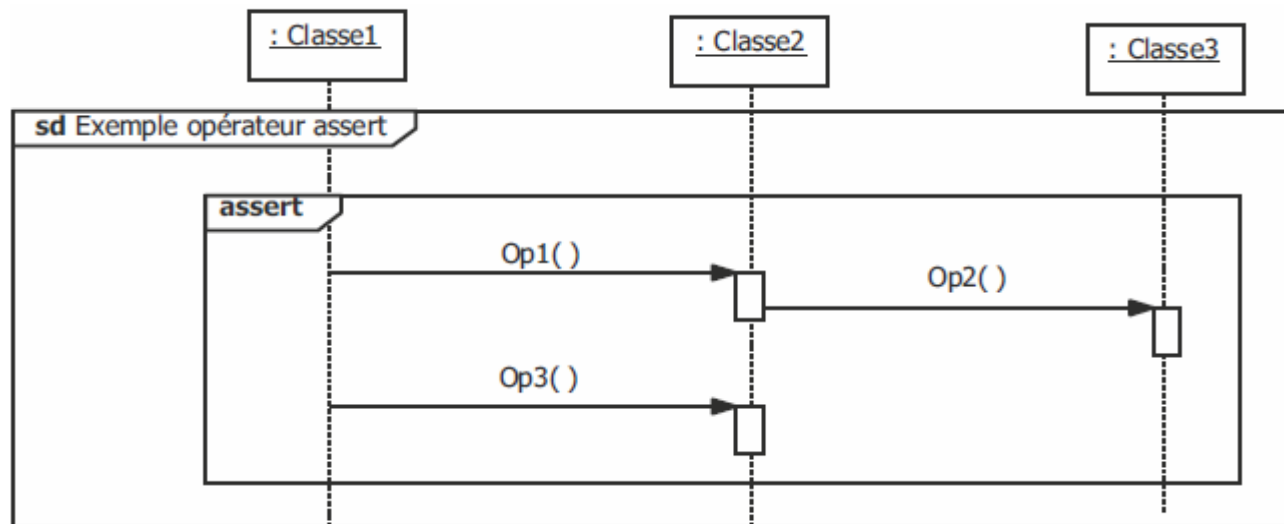
## Le diagramme de séquence

Fragment d'interaction

### ❑ Les opérateurs *assert*

L'opérateur assert (assertion) permet d'indiquer qu'une séquence d'interactions est l'unique séquence possible en considérant les messages échangés dans le fragment.

Toute autre configuration de message est invalide.

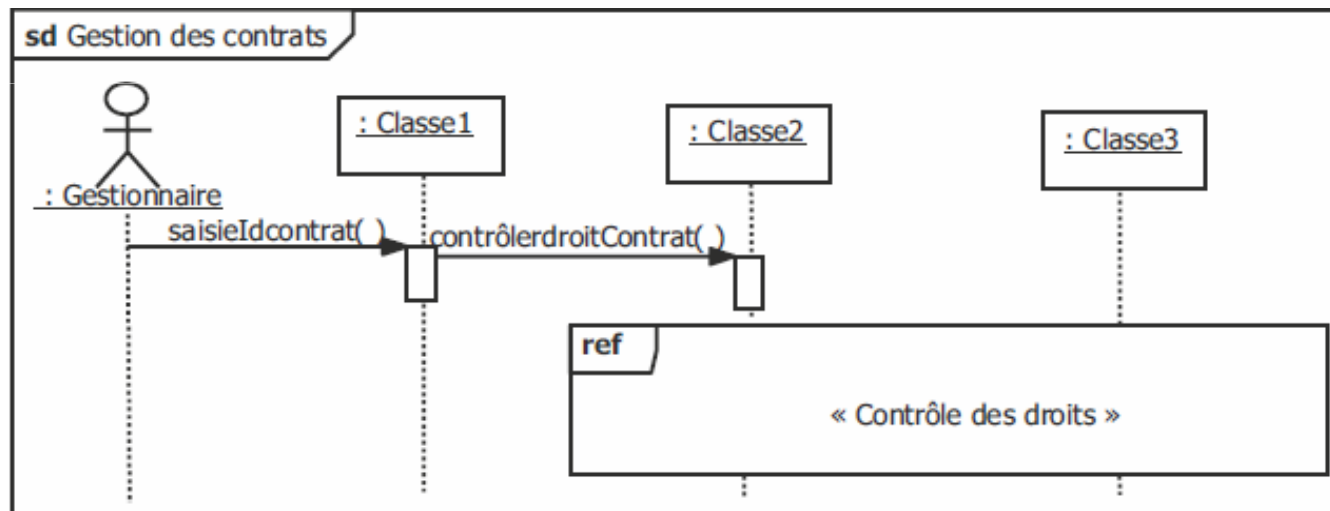


## Le diagramme de séquence

Fragment d'interaction

### ❑ Les opérateurs *ref*

L'opérateur *ref* permet d'appeler une séquence d'interactions décrite par ailleurs constituant ainsi une sorte de sous-diagramme de séquence.



# Diagramme de cas d'utilisation vs Diagramme de Séquence

- Un **cas d'utilisation** apparaît comme un **scénario**, décrit par un diagramme de séquences.
- **Diagramme de séquence**: exprime la séquence des interactions entre objets du système selon un point de vue temporel, pour réaliser le cas d'utilisation.

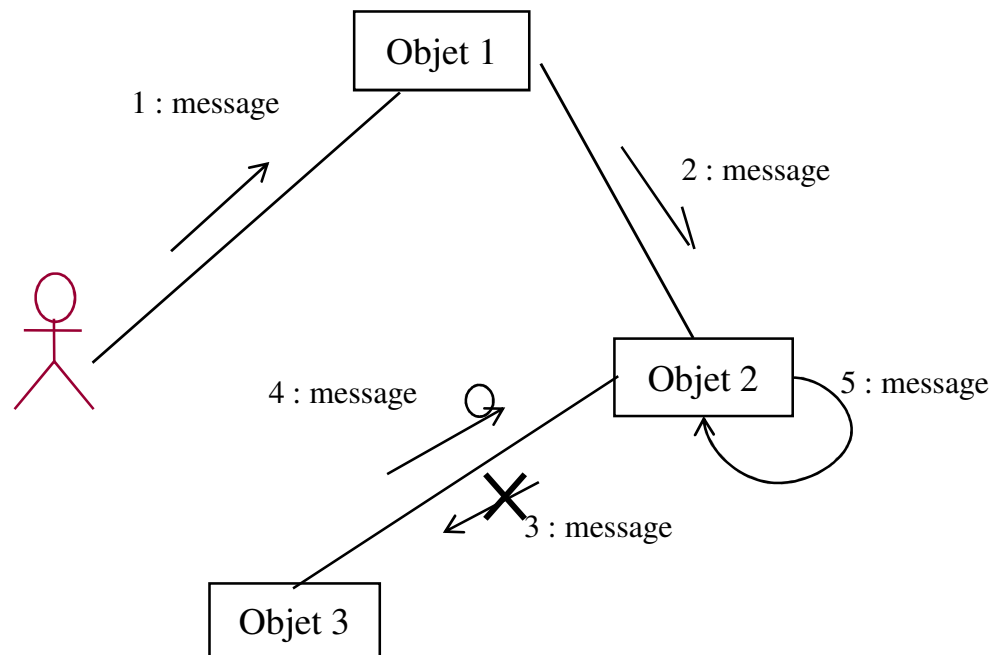
## Diagramme de Communication (Collaboration)



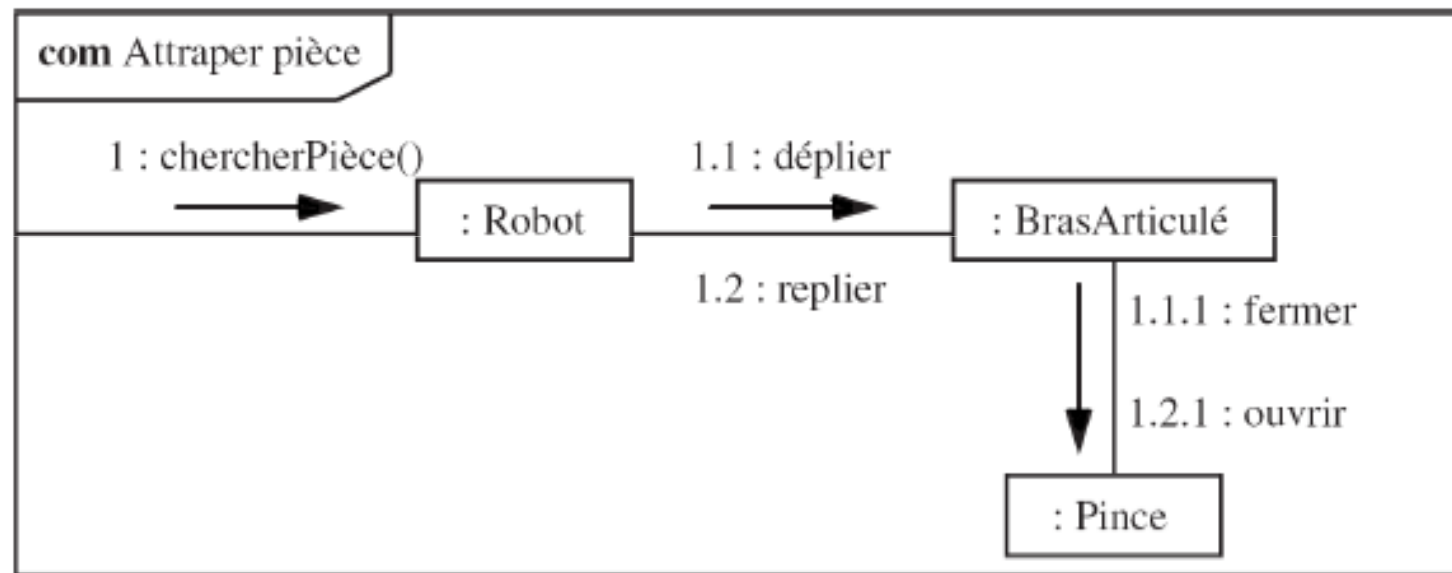
# *Notation*

- Les diagrammes de communication sont un recoupement des diagrammes d'objets et des diagrammes de séquences.
- Chaque objet intervient de la même façon que dans les diagrammes de séquence
- Les messages sont obligatoirement numérotés, la numérotation composée étudiée dans le cadre des diagrammes de séquence pouvant également être employée

# Notation



# Exemple



# Rôles et connecteurs

Le rôle permet de définir le contexte d'utilisation de l'interaction.

- Si la ligne de vie est un objet, celui-ci peut avoir plusieurs rôles au cours de sa vie.

Les relations entre les lignes de vie sont appelées connecteurs .

- Un connecteur se représente de la même façon qu'une association mais la sémantique est plus large : un connecteur est souvent une association transitoire.

Comme pour les diagrammes de séquence, la syntaxe d'une ligne de vie est :

`<nomRole >[< selecteur >]: < nomClasseur >`

# Notation (...)

## La place de l'utilisateur

- La notation permet de faire figurer **un acteur** dans un diagramme de collaboration afin de représenter le déclenchement des interactions par un élément externes au système. Grâce à cet artifice, l'interaction peu être décrite de manière plus abstraite sans entrer dans les détails des objets de l'interface utilisateur.
- Le premier message est envoyé par l'acteur, représenté par un symbole graphique semblable (type bonhomme) au modèle des cas d'utilisation, ou bien par un objet muni d'un stéréotype qui précise sa qualité d'acteur.

# Type de messages

Comme dans les diagrammes de séquence, on distingue deux types de messages :

Un message **synchrone** bloque l'expéditeur jusqu'au retour du destinataire. Le flot de contrôle passe de l'émetteur au récepteur.



Un message **asynchrone** n'interrompt pas l'exécution de l'expéditeur. Le message envoyé peut être pris en compte par le récepteur à tout moment ou ignoré.

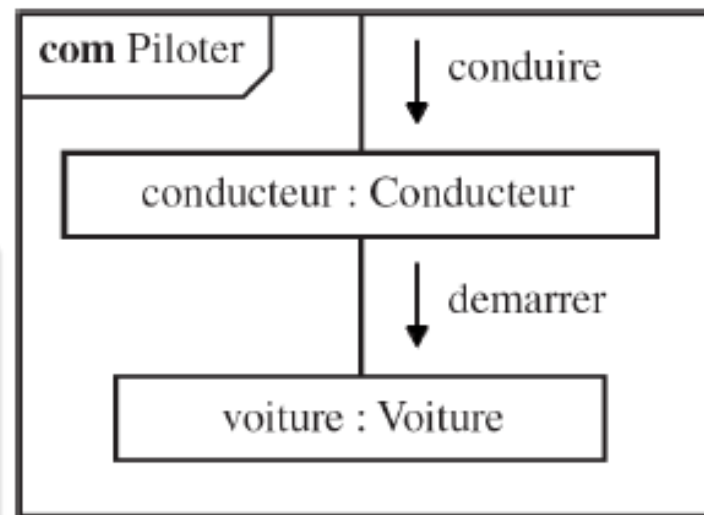


# Représentation des messages

Les flèches représentant les messages sont tracées à côté des connecteurs qui les supportent.

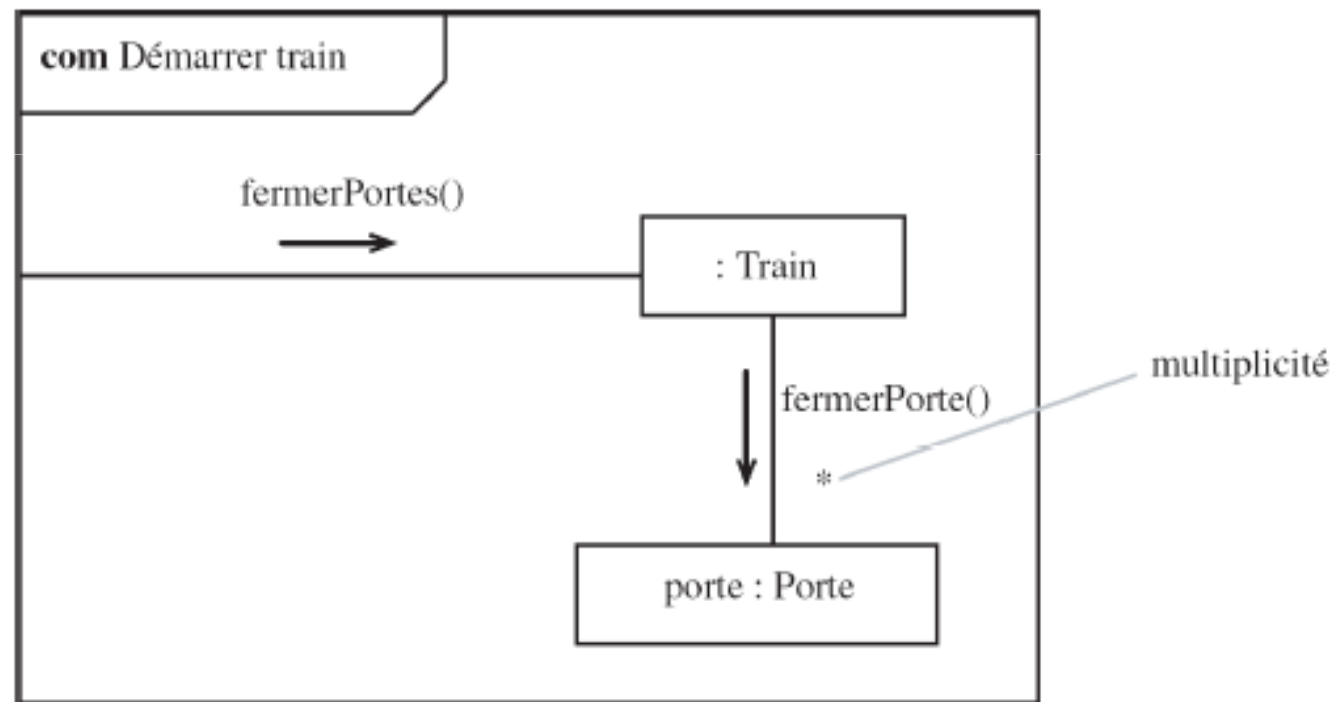
Attention

Bien faire la distinction entre les messages et les connecteurs : on pourrait avoir un connecteur sans message, mais jamais l'inverse.



# Multiplicité

Les connecteurs permettent de rendre compte de la multiplicité.



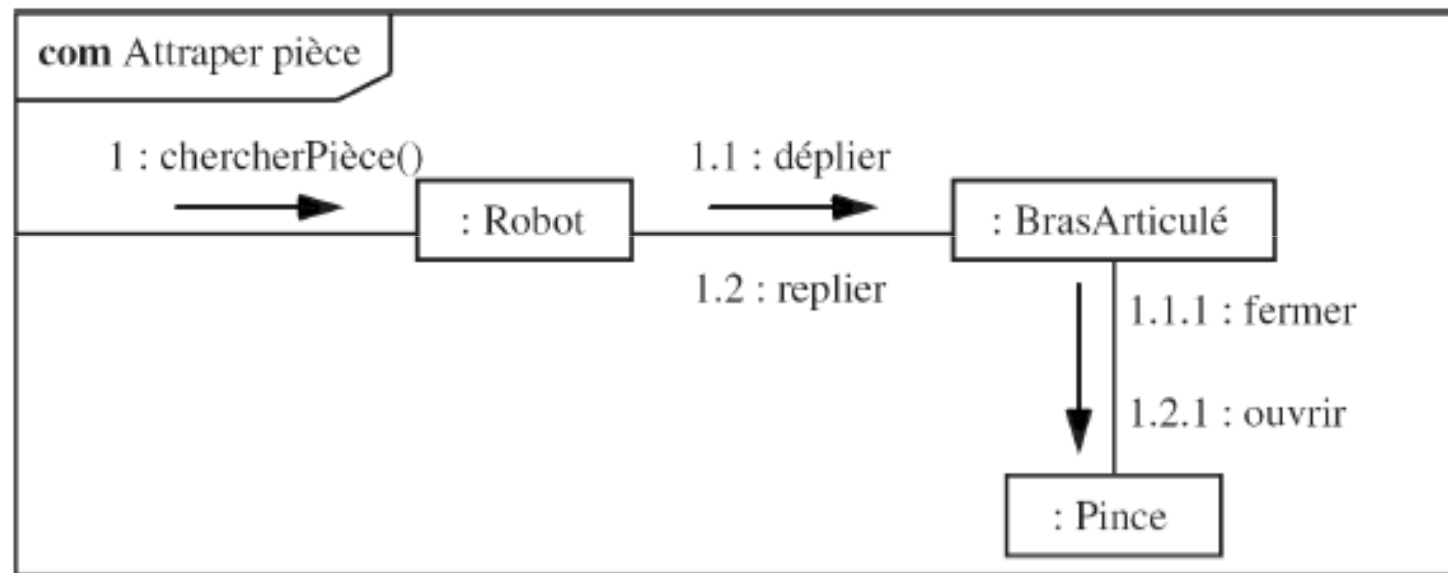


# Numéros de séquence des messages

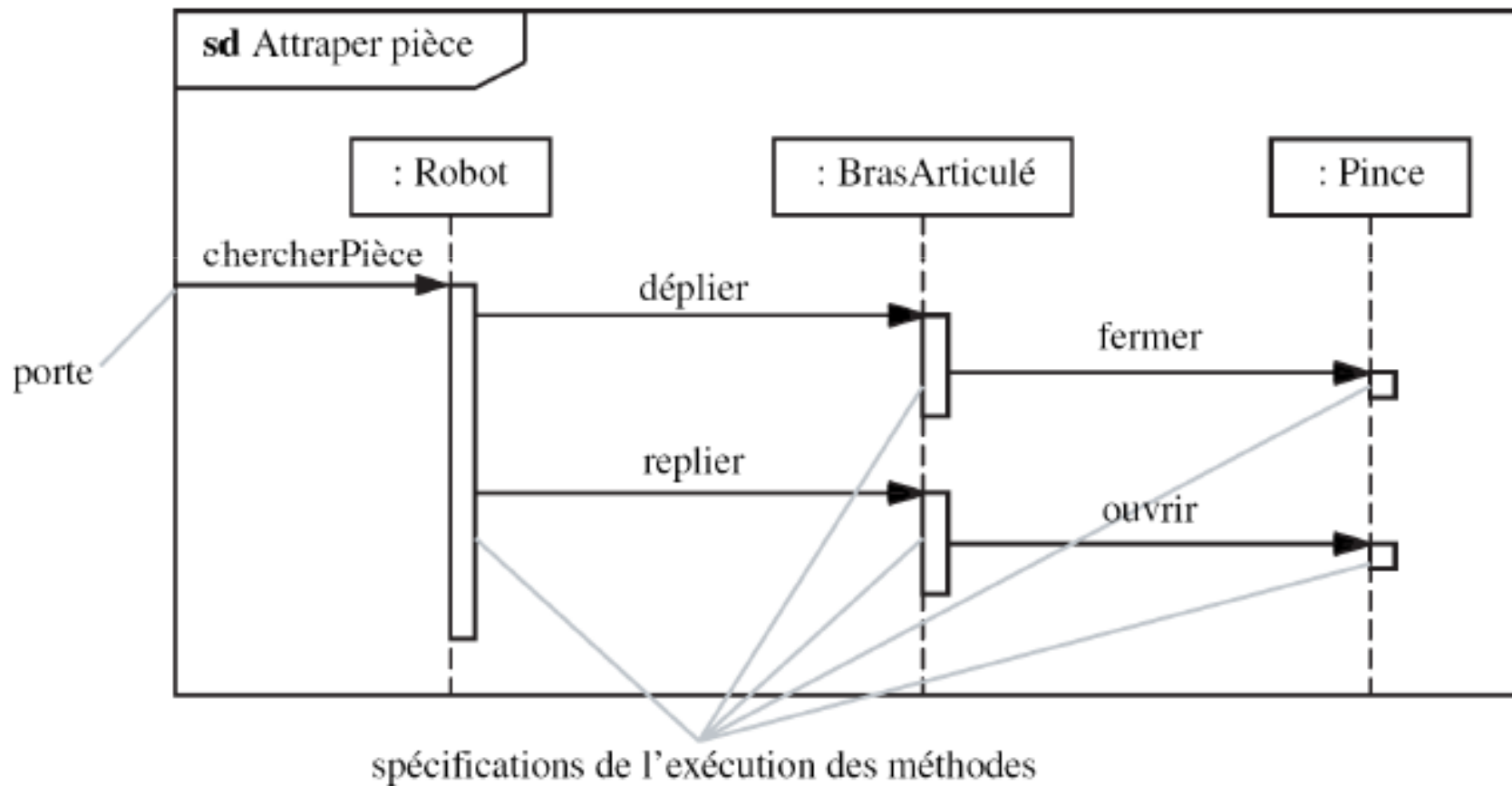
Pour représenter les aspects temporels, on adjoint des numéros de séquence aux messages.

- Des messages successifs sont ordonnés selon un numéro de séquence croissant (1, 2, 3, ... ou encore 3.1, 3.2, 3.3, ...).
- Des messages envoyés en cascade (ex : appel de méthode à l'intérieur d'une méthode) portent un numéro d'emboîtement avec une notation pointée
  - 1.1, 1.2, ... pour des messages appelés par une méthode dont l'appel portait le numéro 1
  - 2.a.1, 2.a.2, ... pour des messages appelés par une méthode dont l'appel portait le numéro 2.a

# Equivalence avec un diagramme de séquence



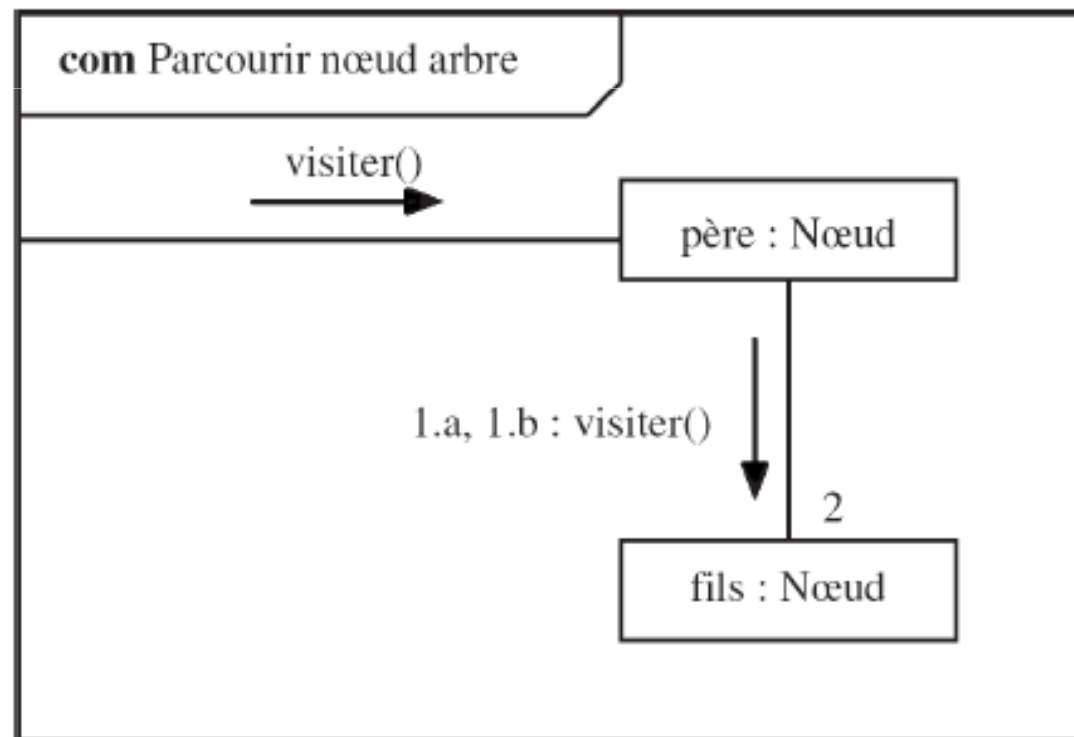
# Equivalence avec un diagramme de séquence



# Messages simultanés

Lorsque des messages sont envoyés en parallèle, on les numérote avec des lettres

- 1.a, 1.b,... pour des messages simultanés envoyés en réponse à un message dont l'envoi portait le numéro 1



# Opérateurs de choix et de boucle

Pas d'opérateurs combinés dans les diagrammes de communication

- *\*[<clause itération>] représente une itération.*
  - La clause d'itération peut être exprimée dans le format  $i:=1..n$
  - Si c'est une condition booléenne, celle-ci représente la condition d'arrêt
- *\*[<clause itération>] représente un choix.* La clause de condition est une condition booléenne.

# Syntaxe des messages

- Syntaxe complète des messages est :

*[<numeroSéquence>][<expression>]:<message>*

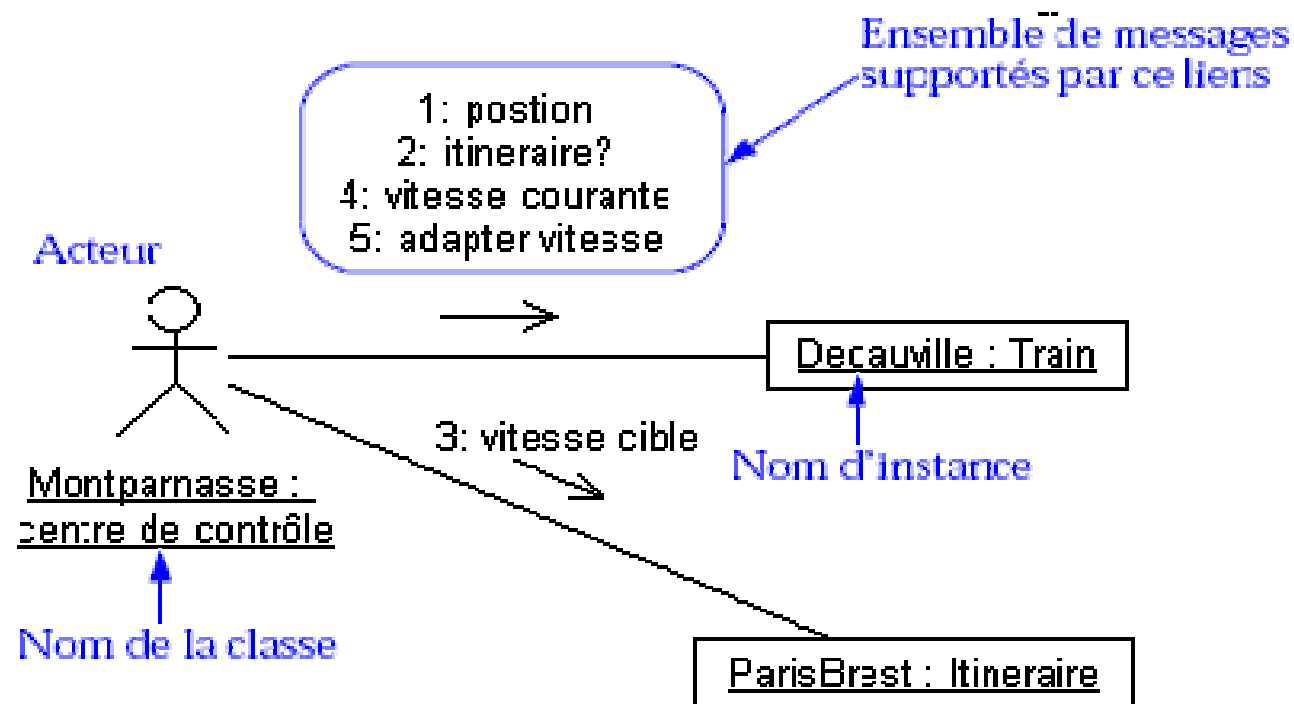
- « message » a la même forme que dans les diagrammes de séquence, « numeroSéquence » représente le numéro de séquençement des messages et « expression » précise une itération ou un embranchement.

- Exemples :

- 2 : affiche(x,y) : message simple.
- 1.3.1 : trouve("Hadock") : appel emboîté.
- 4 [x < 0] : inverse(x,couleur) : conditionnel.
- 3.1 \*[i:=1..10] : recommencer() : itération.

## ❑ Scénario du contrôle de vitesse

Certains attributs de classe peuvent être pris en compte:

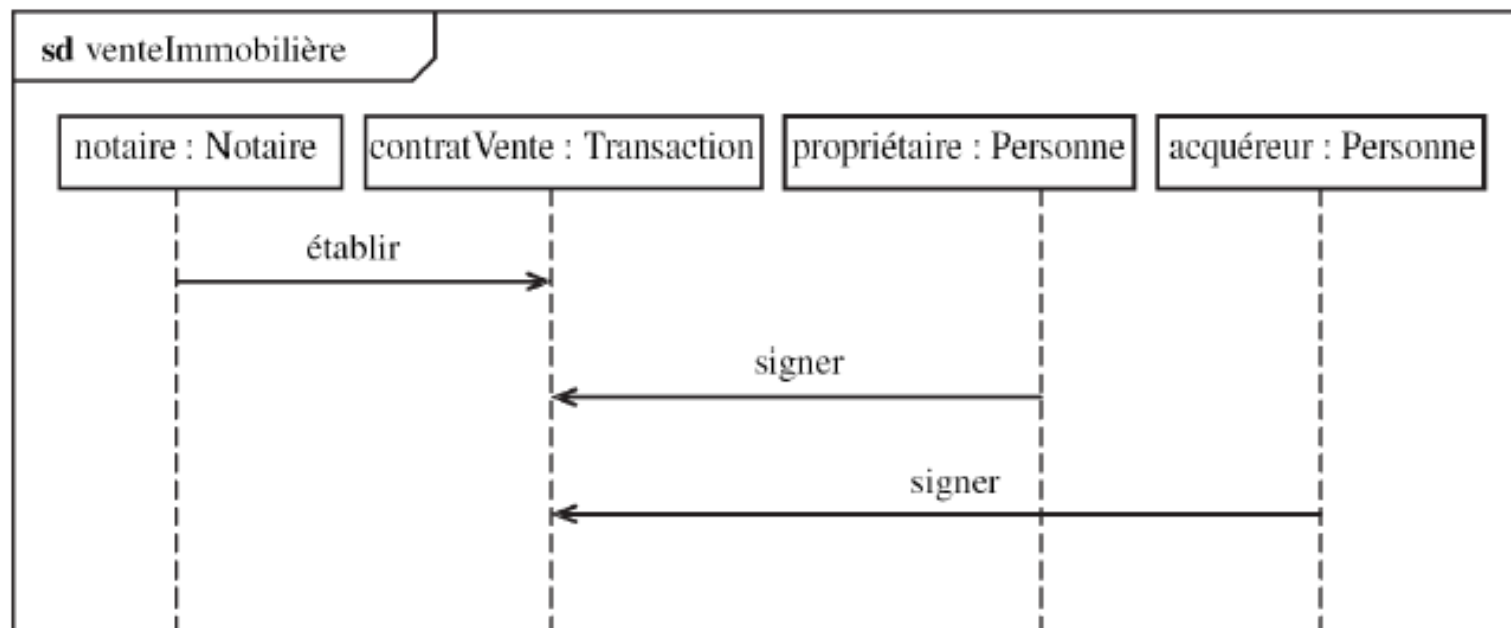


- La classe décrivant la position du train sur l'itinéraire est un exemple pour lequel tout instance doit être explicité avec ces attributs.
- Les messages supportés par les liens peuvent être dans les deux directions.



# Collaborations et interactions

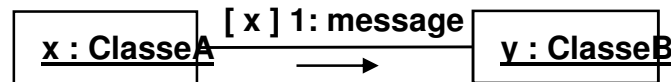
- Les collaborations donnent lieu à des interactions
  - Les interactions documentent les collaborations
  - Les collaborations organisent les interactions.
- Les interactions se représentent indifféremment par des diagrammes de **communication** ou de **séquence**



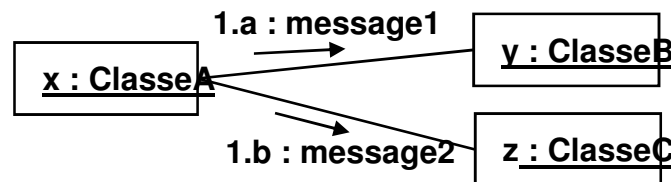
# UML - Le langage Objet unifié

## Les diagrammes d'UML

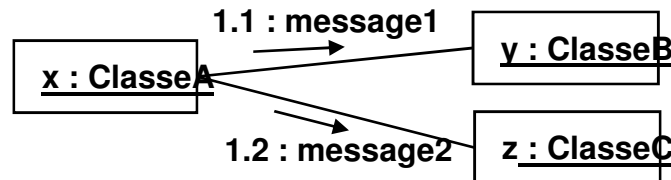
### Diagrammes de collaboration - conventions



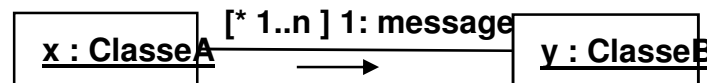
message soumis à la condition x



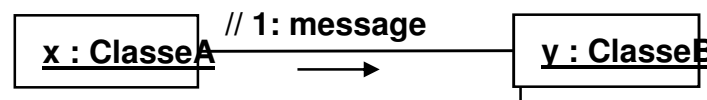
message1 et message 2 en parallèle



choix entre message1.1 et message1.2



message envoyé n fois



message envoyé en parallèle à plusieurs instances de la classe B



a récupère la valeur renvoyée par l'exécution du message

## Le diagramme de communication / collaboration

