Développement Orienté Objets

Diagrammes de séquence UML

Arnaud Lanoix Brauer
Arnaud.Lanoix@univ-nantes.fr

Département informatique

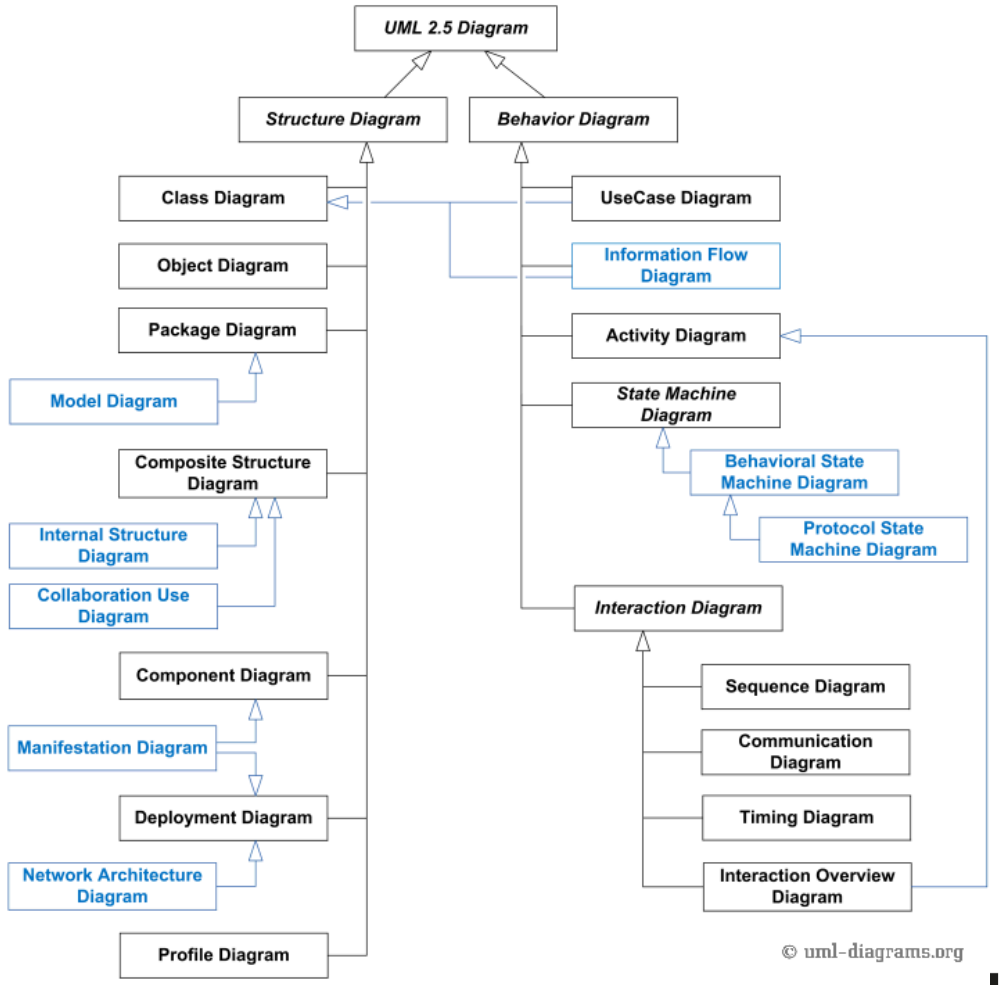
Diagrammes en UML 2.5

Diagrammes de structure (statiques)

- **Diagramme de classes**
- Diagramme d'objets
- Diagramme de paquets
- Diagramme de structure composite
- Diagramme de composants
- Diagramme de déploiement
- Diagramme de profils

Diagrammes de comportement (dynamiques)

- Diagramme des cas d'utilisation
- Diagamme d'activité
- Diagramme états-transitions
- *Diagrammes d'interaction*
  - **Diagramme de séquence**
  - Diagramme de communication
  - Diagramme de temps
  - Diagramme global d'interaction



IUT Nantes

Sommaire

IUT Nantes

Diagramme d'objets

Un diagramme d'objets UML représente des instances de classes, c-à-d des objets :

- est **forcément** associé à un diagramme de classes
  - **Attention :** les différents diagrammes doivent être cohérents
- illustre aussi l'état de l'objet
- illustre les associations entre les différents objets
- utile lorsque les les associations possible sont complexes

= photographie (instantanée) des objets instanciés

- permet d'illustrer une configuration particulière
- utile lorsque les associations possible entre objets sont complexes

IUT Nantes

Diagramme d'objets

Un diagramme d'objets UML représente des instances de classes, c-à-d des objets :

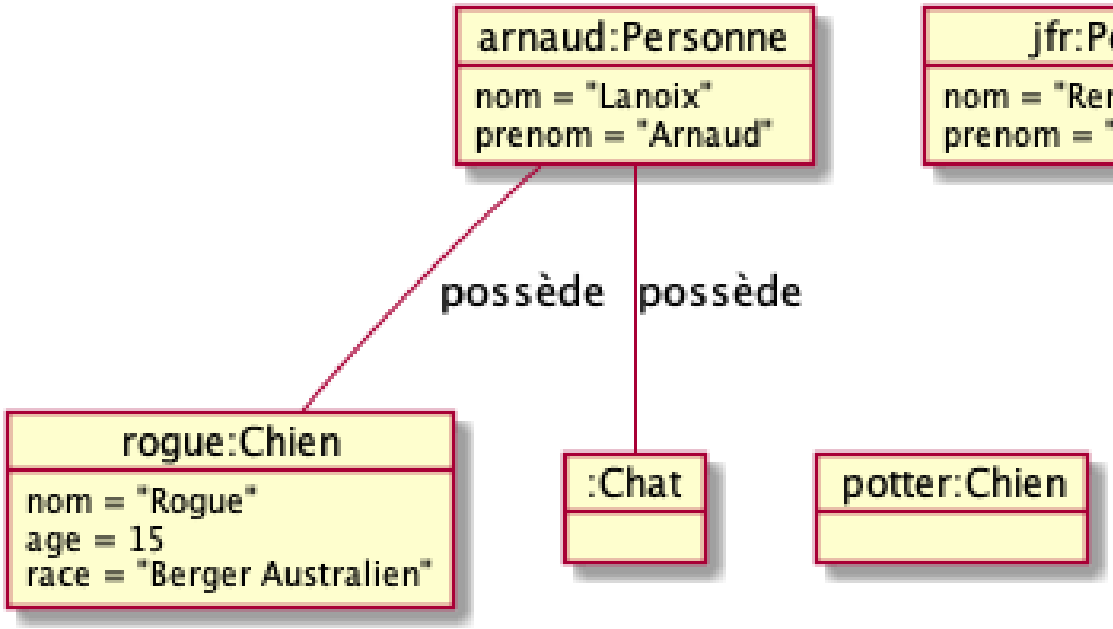
- est **forcément** associé à un diagramme de classes
  - **Attention :** les différents diagrammes doivent être cohérents
- illustre aussi l'état de l'objet
- illustre les associations entre les différents objets
- utile lorsque les les associations possible sont complexes

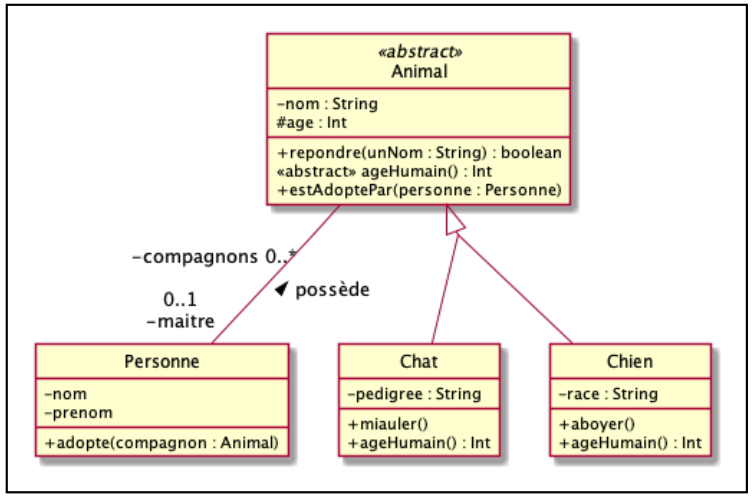
= **photographie (instantanée) des objets instanciés**

- permet d'illustrer une configuration particulière
- utile lorsque les associations possible entre objets sont complexes

IUT Nantes

Exemple de diagramme d'objets



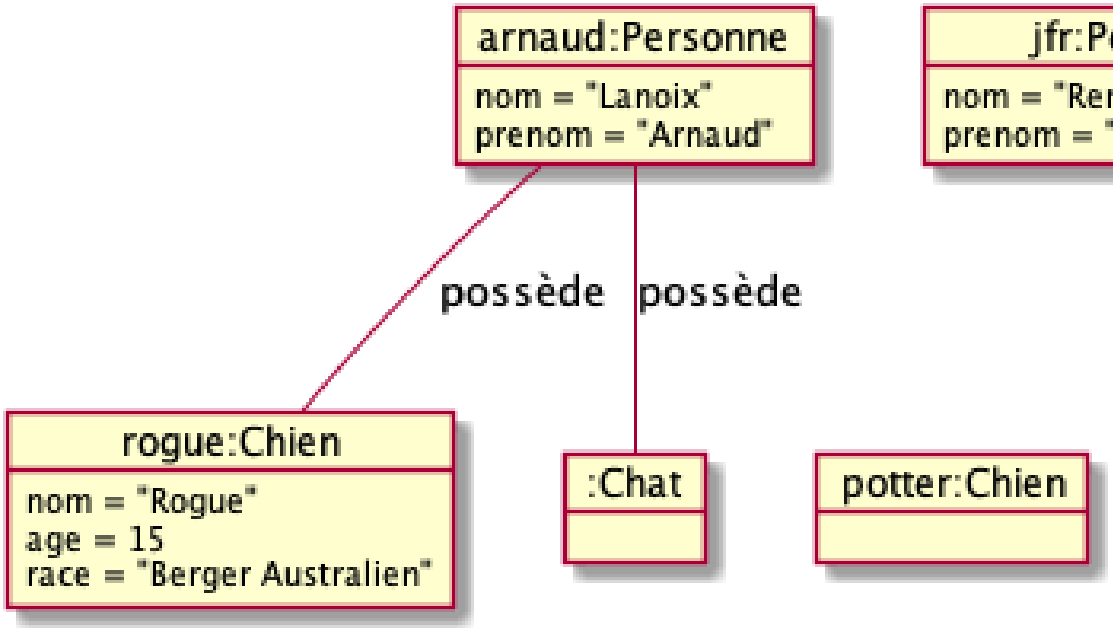


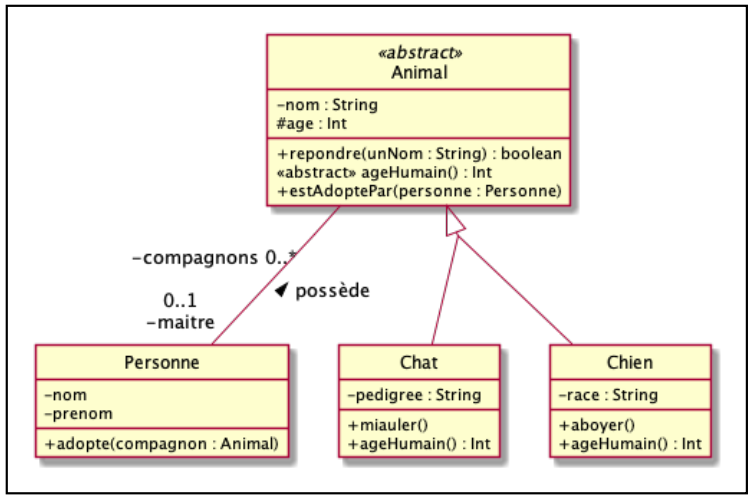
- on peut identifier (ou pas) les objets : `rogue:Chien`
- on peut valuer (ou pas) les attributs d'un objet

Les deux diagrammes sont cohérents

IUT Nantes

Exemple de diagramme d'objets





- on peut identifier (ou pas) les objets : `rogue:Chien`
- on peut valuer (ou pas) les attributs d'un objet

Les deux diagrammes sont cohérents

IUT Nantes

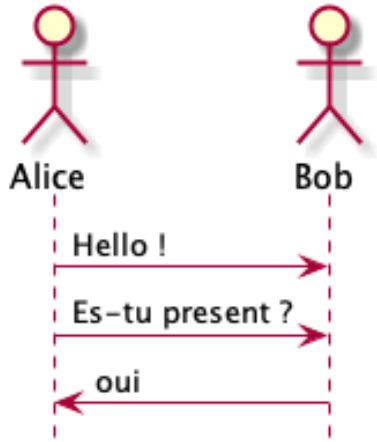
Sommaire

IUT Nantes

Diagramme de séquence

Un diagramme de séquence UML illustre les interactions entre acteurs de manière chronologique.

- Le "passage du temps" est représenté verticalement

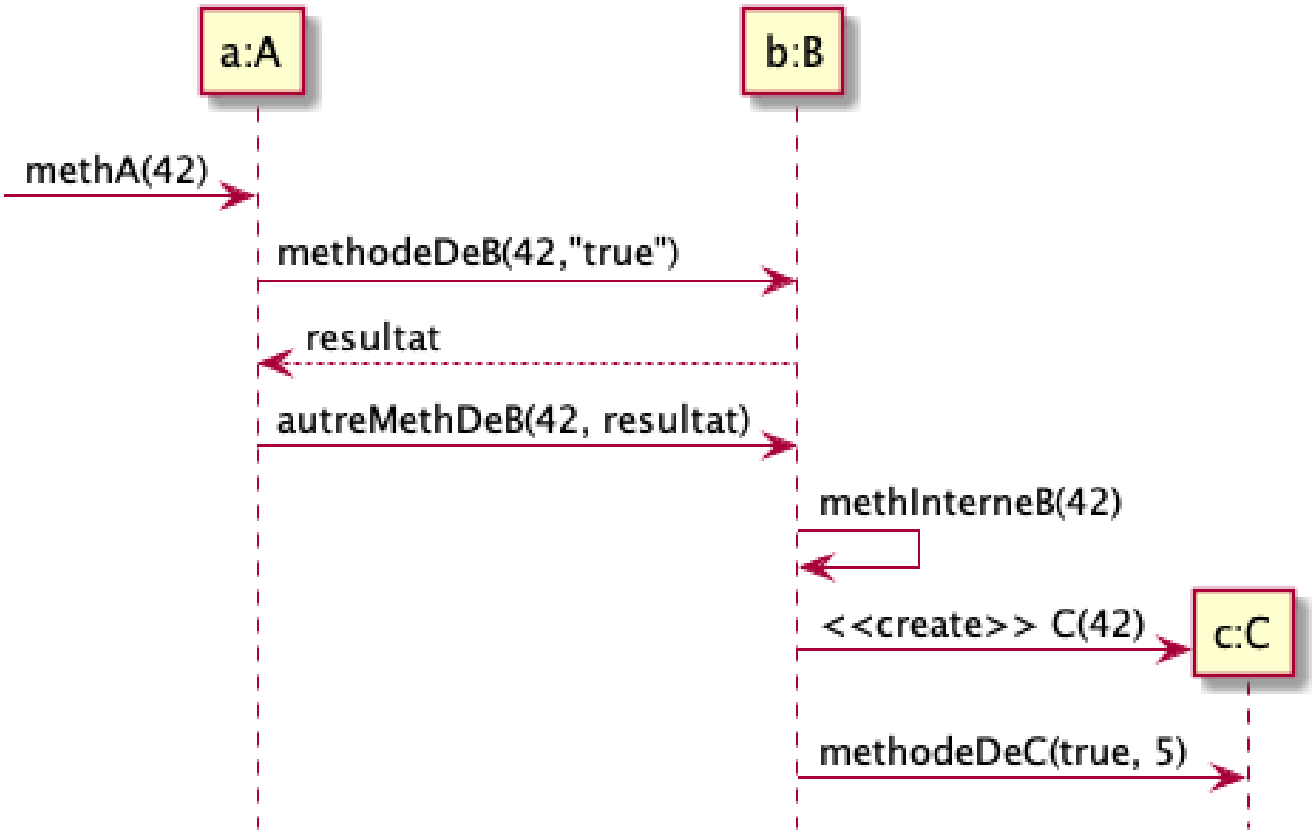


Nous concernant

- Les acteurs seront des objets (instances de classes) qui interagiront
- Interaction = appel de méthodes
- sera **forcément** associé à un diagramme de classe
  - **Attention :** les différents diagrammes devront être cohérents

IUT Nantes

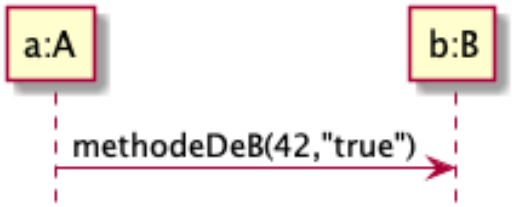
Syntaxe des diagrammes de séquence



- Les acteurs (en haut) sont des instances de classes
- Le temps est représenté par les lignes pointillés verticales
- Les appels de méthodes s'enchaînent successivement

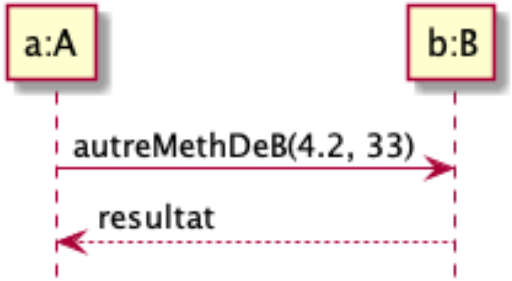
IUT Nantes

Details



Appel de méthode :

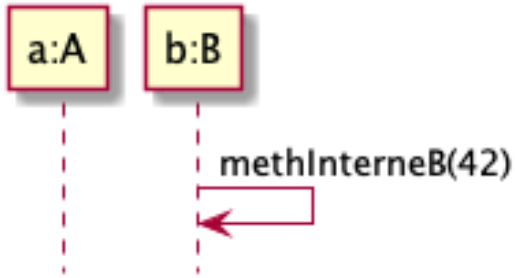
a instance de `A` appelle la méthode `methodeDeB(...)` sur l'instance `b` de la classe `B`



Appel avec retour :

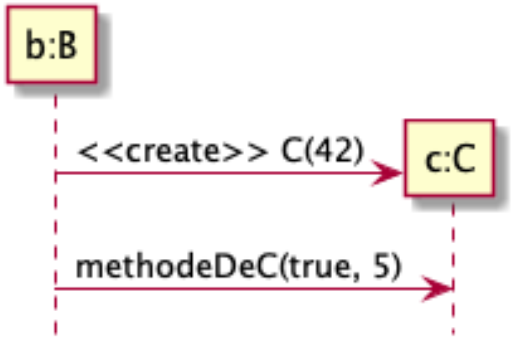
l'appel à la méthode `autreMethDeB(...)` renvoie un `resultat` à l'instance appelante

---



Appel à une méthode "de soi-même" :

`b:B` appelle un autre méthode de la classe `B`

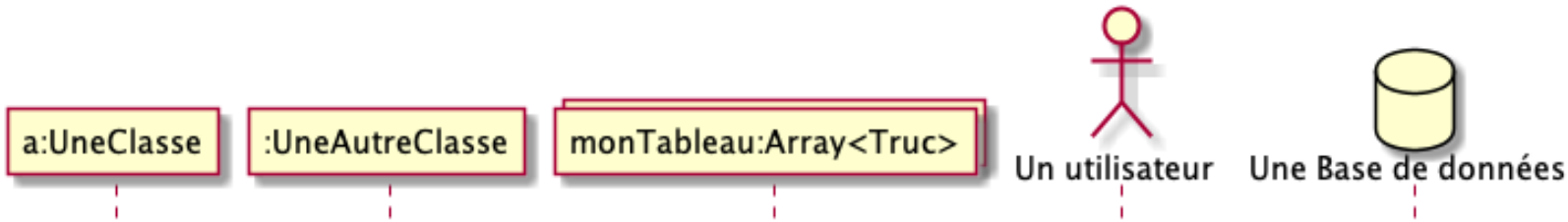


Instantiation d'un nouvel objet :

`b:B` créé une nouvelle instance de `C` en appelant son constructeur

IUT Nantes

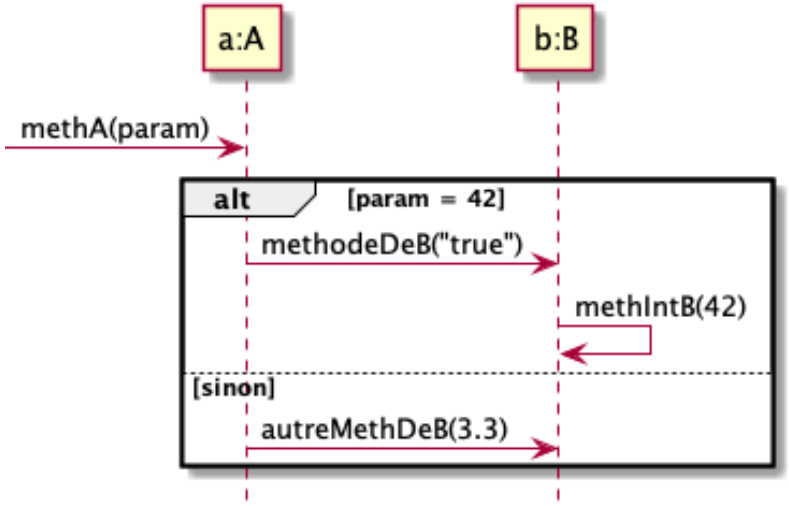
Différents acteurs dans un diagramme de séquence



- Les acteurs peuvent être des instances de classes, nommées ou non
- Un acteur peut aussi représenter une collection d'objets d'un type donné
- Un acteur peut être un utilisateur du système modélisé
- Un acteur peut être une base de données
- ...

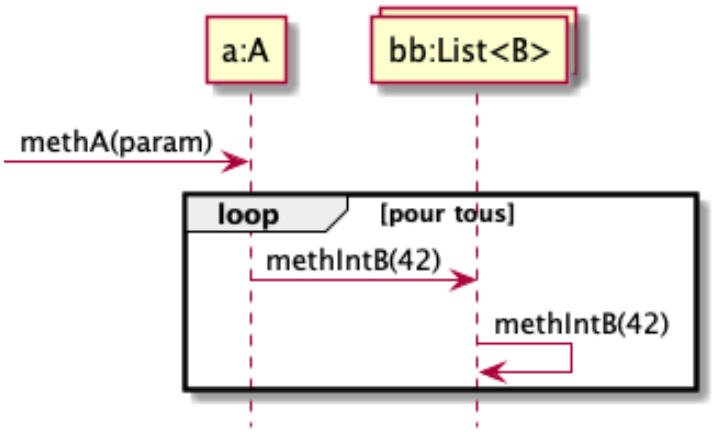
IUT Nantes

Fragments combinés



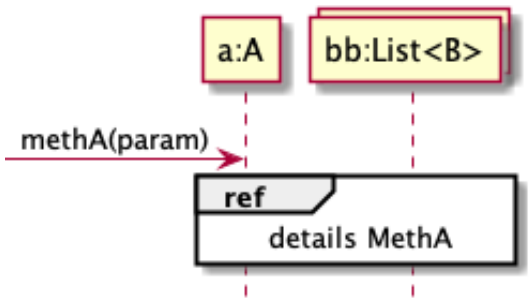
`alt` pour alternative : exprime un comportemement conditionnel

- Les conditions `[...]` sont exprimées en langue naturelle



`loop` : exprime une boucle

- La condition `[...]` est exprimée en langue naturelle

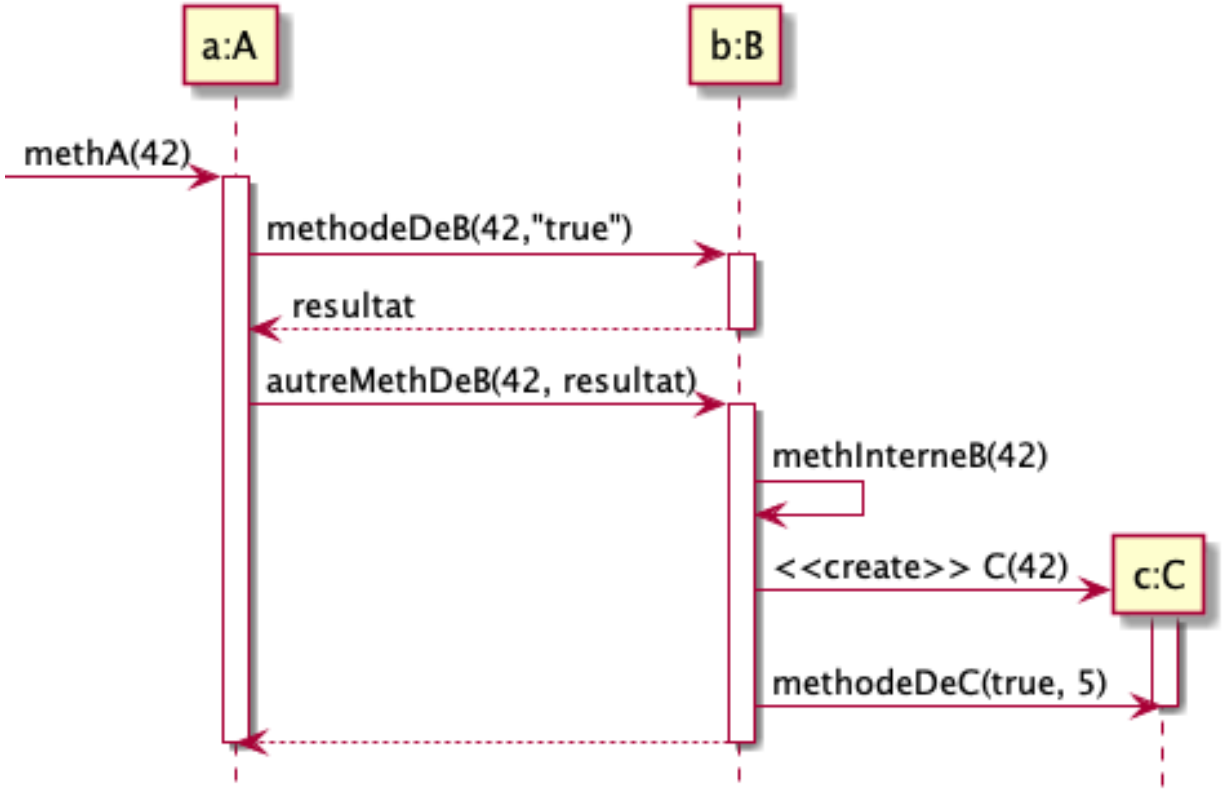


`ref` : permet de référencer un autre diagramme

IUT Nantes

Lignes de vie

= "rectangles superposés sur la ligne de temps"

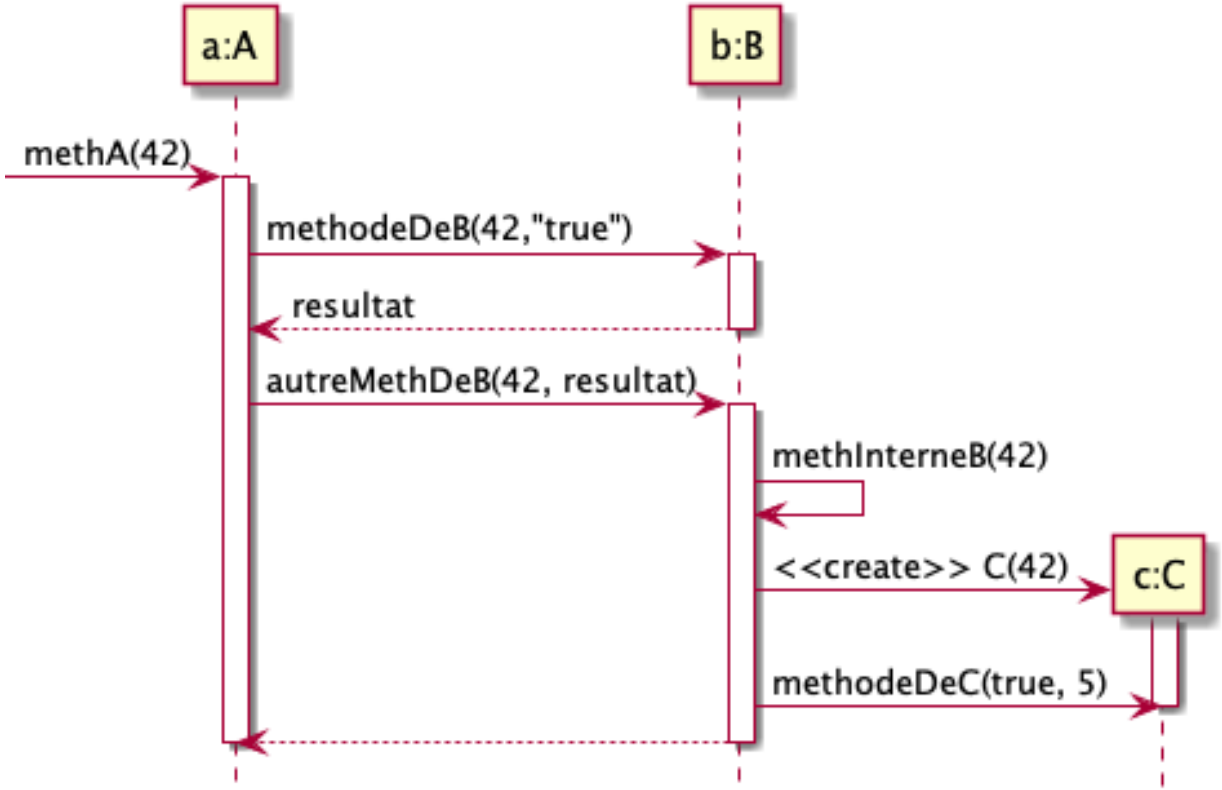


- permet d'indiquer les périodes d'activité des différents acteurs
- pas très utile dans notre contexte

IUT Nantes

Lignes de vie

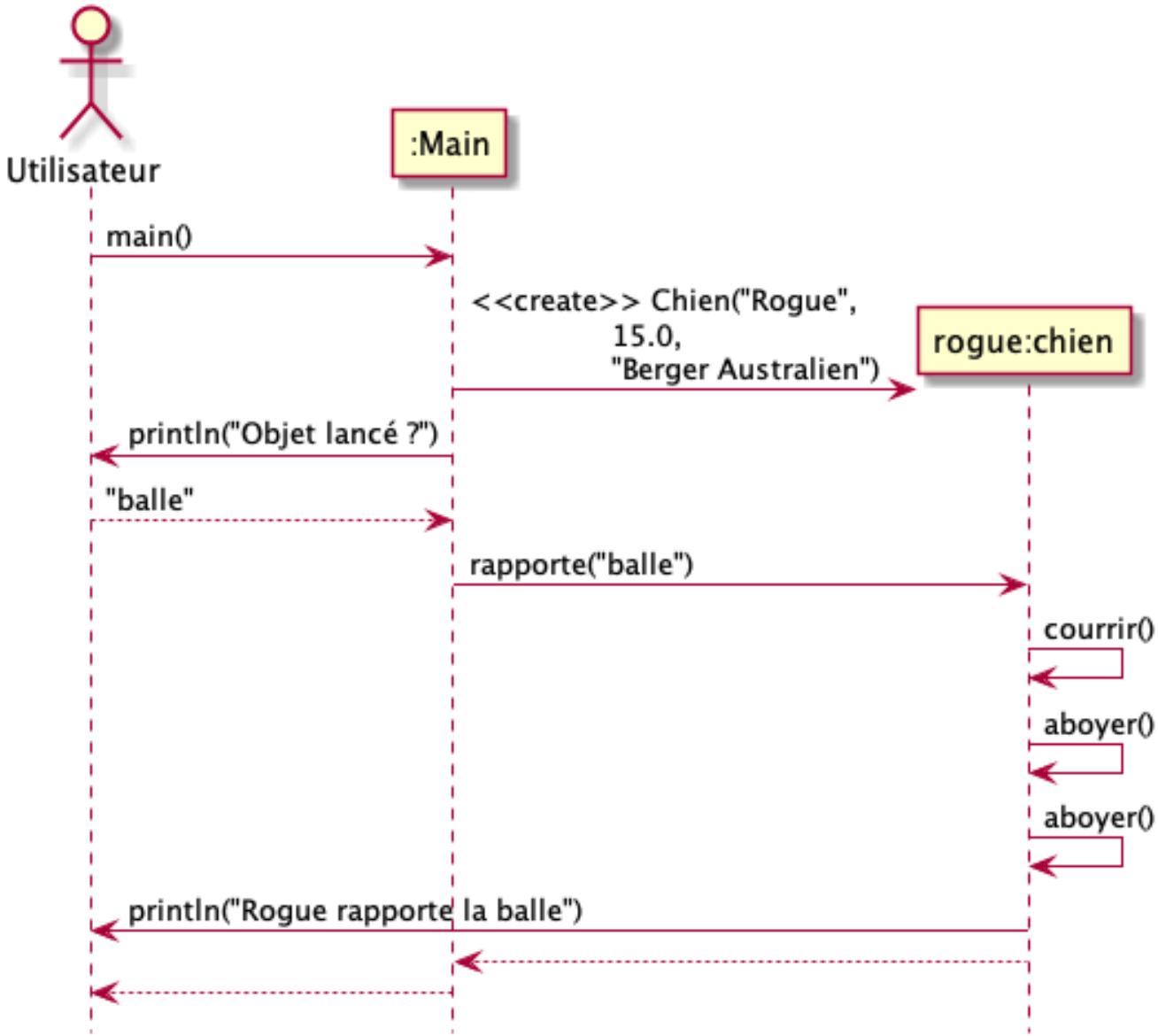
= "rectangles superposés sur la ligne de temps"



- permet d'indiquer les périodes d'activité des différents acteurs
- pas très utile dans notre contexte

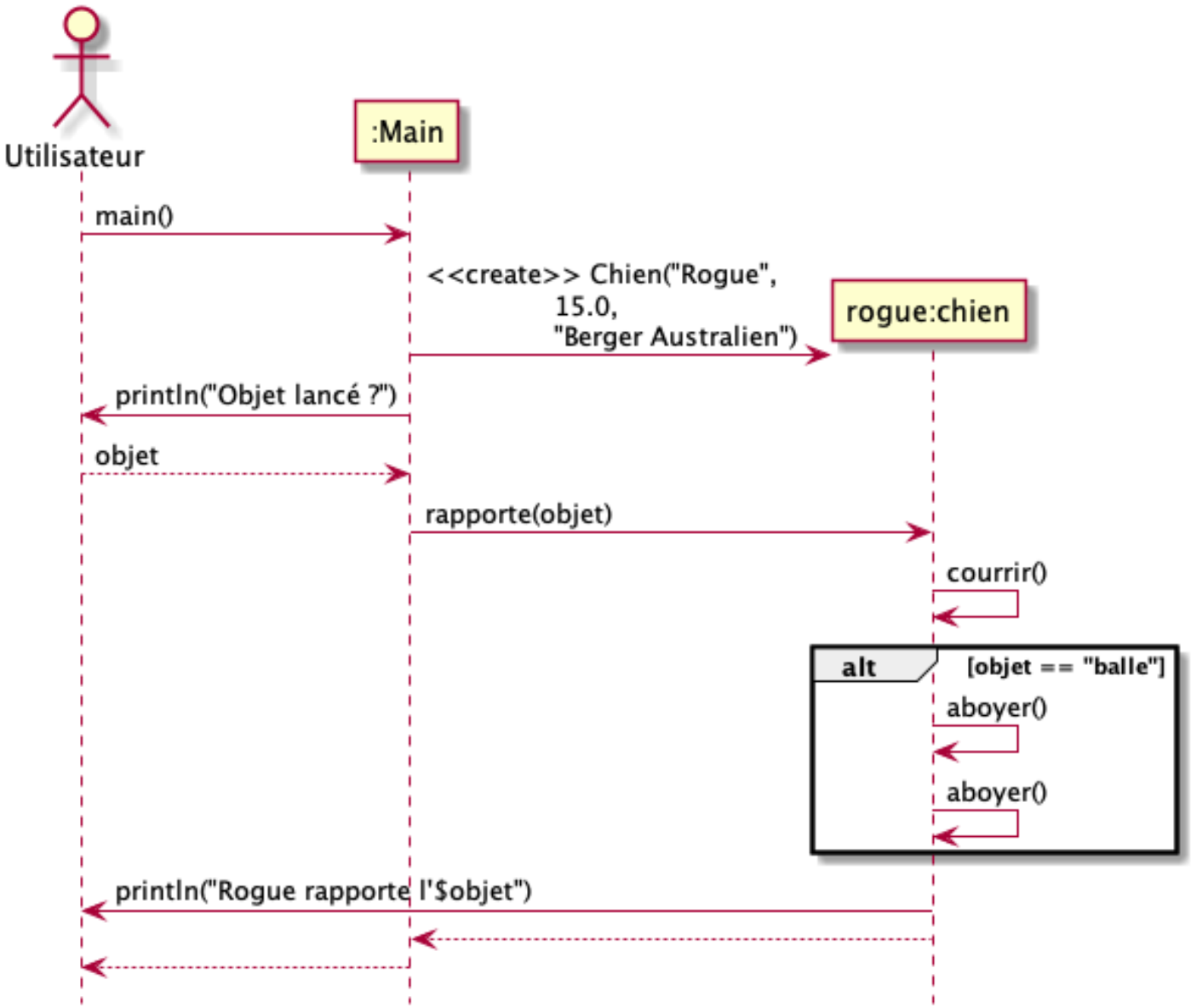
IUT Nantes

Exemple



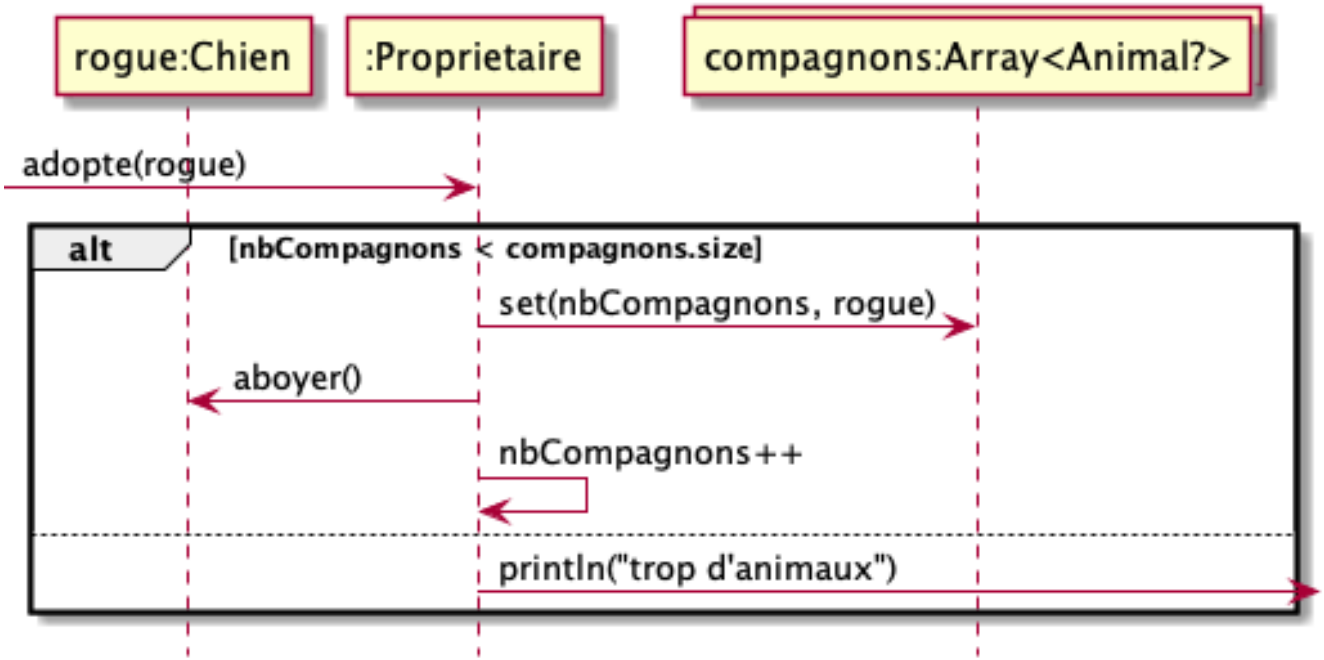
IUT Nantes

Exemple : `alt`



IUT Nantes

Exemple : `alt` (2)



Exemple : `loop`

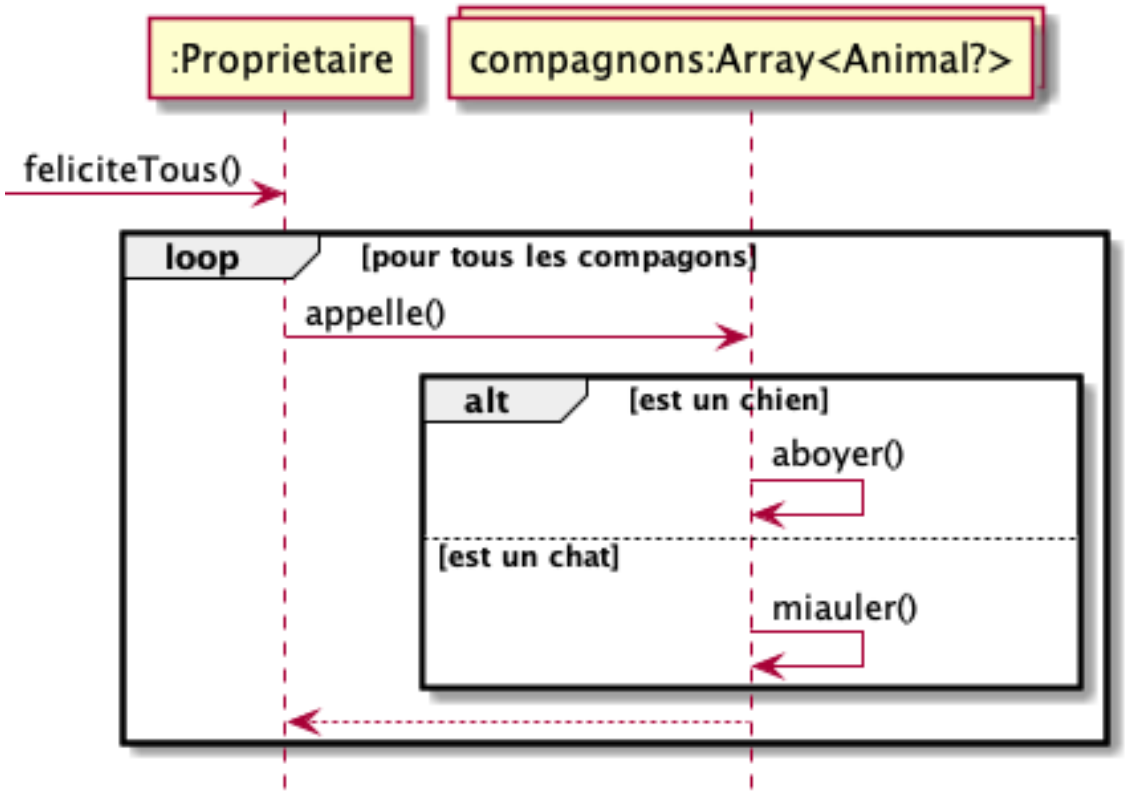
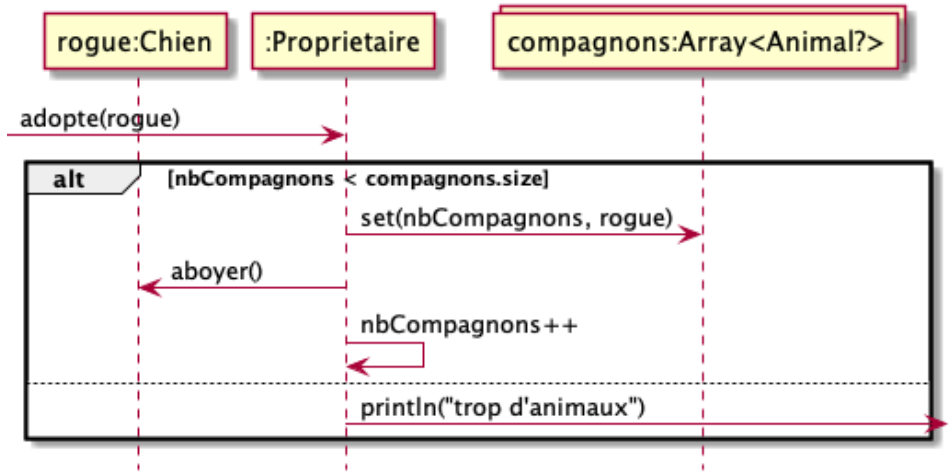


Diagramme de séquence + code Kotlin

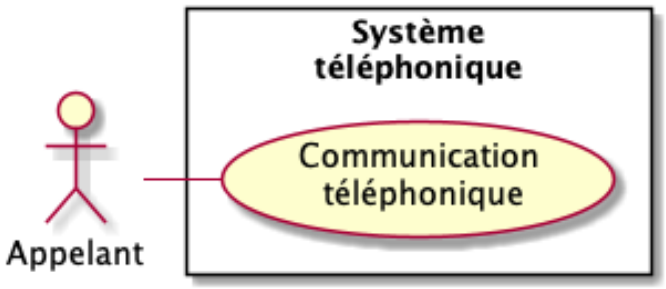



```kotlin
class Proprio(...) {
  compagnons : Array<Animal?>
  nbCompagnons : Int = 0
  ...
  fun adopte(nouveau : Chien) {
    if (nbCompagnons < compagnons.size) {
      compagnons[nbCompagnons] = nouveau
      nouveau.aboyer()
      nbCompagnons++
    }
    else {
      println("trop d'animaux")
  }
```

IUT Nantes

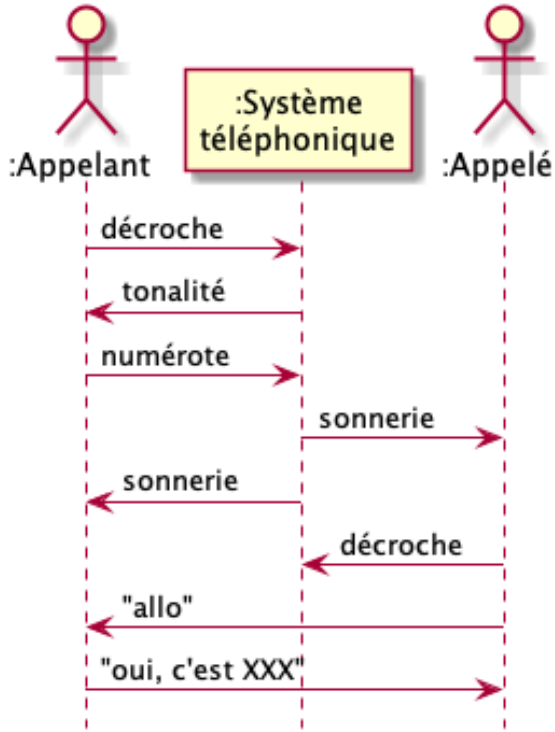
Diagramme de cas d'utilisation + diagramme de séquence

Les diagrammes de séquences peuvent aussi servir à illuster/documenter des cas d'utilisation, en illustrant de scénarios particuliers



Use case : Communication téléphonique

Scénario nominal :



IUT Nantes