

# UML : diagramme de communication

**Achref El Mouelhi**

Docteur de l'université d'Aix-Marseille  
Chercheur en programmation par contrainte (IA)  
Ingénieur en génie logiciel

`elmouelhi.achref@gmail.com`



**UNIFIED  
MODELING  
LANGUAGE**

# Plan

- 1 Introduction
- 2 Ligne de vie
- 3 Connecteur
- 4 Message

## Diagramme de communication ?

- Un diagramme dynamique d'UML
- Appelée diagramme de collaboration dans UML 1
- Version simplifiée d'un diagramme de séquence
- Permettant aussi de représenter les échanges entre les objets (les acteurs)
- Un graphe dont
  - les sommets : les objets
  - les arêtes : les messages échangés

## Mots-clés associés

- Ligne de vie
- Connecteur
- Message

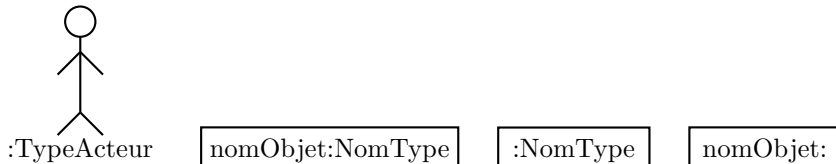
## Ligne de vie

- Exactement comme une ligne de vie dans un diagramme de séquence
- Rectangle au sommet contenant le nom de type (objet, acteur...) **mais sans la ligne verticale pointillée**

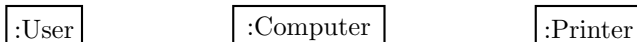
## Plusieurs représentations possibles en UML



## Plusieurs représentations possibles en UML



## Exemple



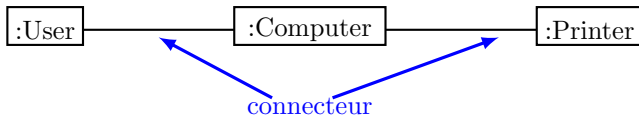
L'acteur `Vendeur` peut être représenté comme un objet.

## Connecteur

- Représenté par une ligne continue
- Définit une communication entre deux lignes de vie
- Pouvant être ornés de multiplicité



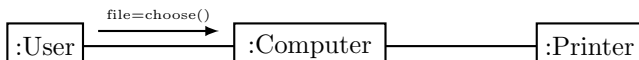
## Voici comment modéliser les connecteurs en UML ?



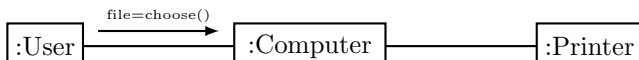
## Message

- Associé à un connecteur
- Représenté par une flèche
  - en traits pleins et à l'extrémité ouverte pour les messages asynchrones
  - en traits pleins et à l'extrémité pleine pour les messages synchrones
- Pouvant être accompagné de plusieurs informations
  - ordre
  - condition de franchissement
  - boucle
  - valeur de retour

## Voici comment modéliser un message dans un diagramme de communication



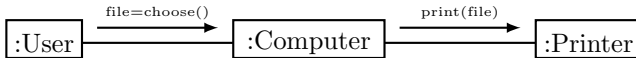
## Voici comment modéliser un message dans un diagramme de communication



### Explication

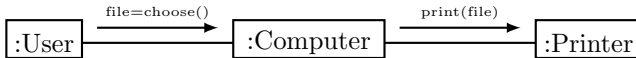
`file` est le fichier choisi par l'utilisateur (la valeur de retour de l'opération `choose`)

**La valeur de retour est optionnelle et peut être utilisée par un autre message**



# UML

**La valeur de retour est optionnelle et peut être utilisée par un autre message**

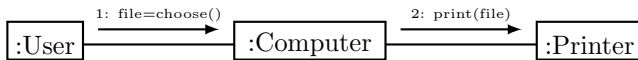


## Question

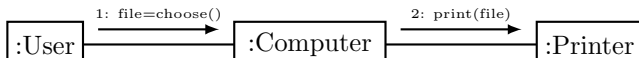
Comment connaître l'ordre ?

- `choose` **ensuite** `print`, ou
- `print` **puis** `choose`

## On peut aussi ordonner les messages



## On peut aussi ordonner les messages

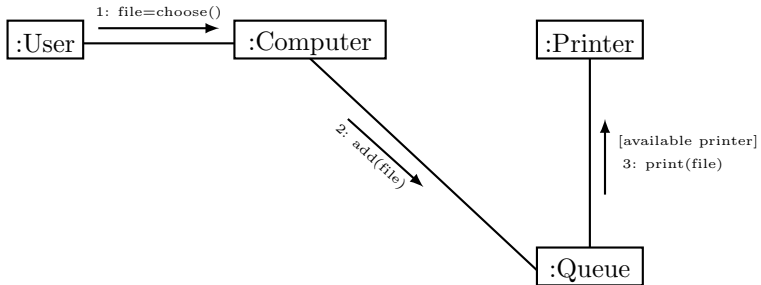


Et si l'imprimante est non disponible ?

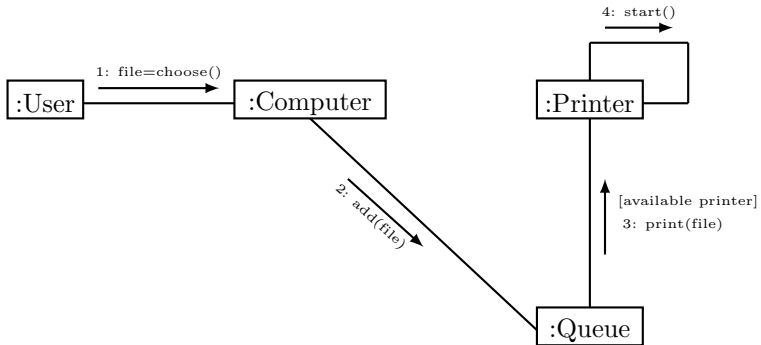
Il faut définir une condition de franchissement



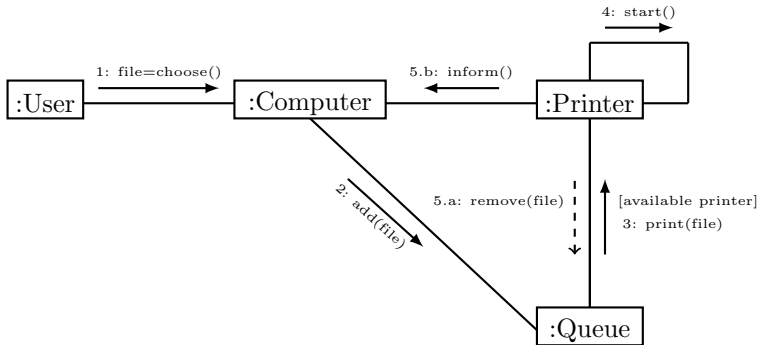
**voici comment on définit les conditions de franchissement**



## Un objet peut envoyer un message à lui même



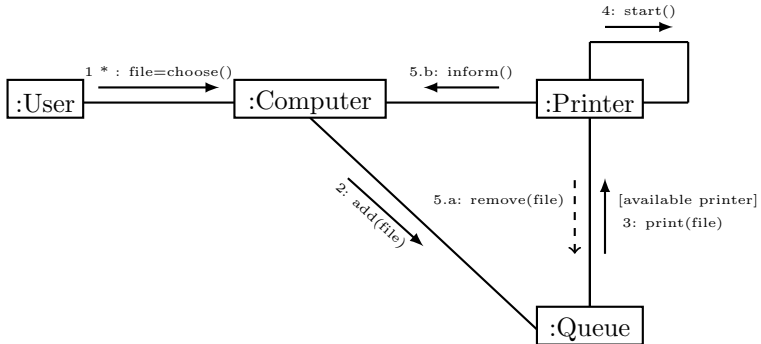
## Il est possible d'envoyer deux messages simultanément



## Remarques

- deux messages simultanés doivent avoir le même numéro de séquence (dans notre exemple 5) suffixés de deux lettres différentes
- un message avec une flèche en pointillé est utilisée pour les messages de retour

## Il est possible de préciser le nombre de répétition d'un message



### Remarques

- \* : plusieurs fois
- \* [i=1..3] : répété trois fois (une boucle allant de 1 à trois)

## On peut aussi définir un contexte

