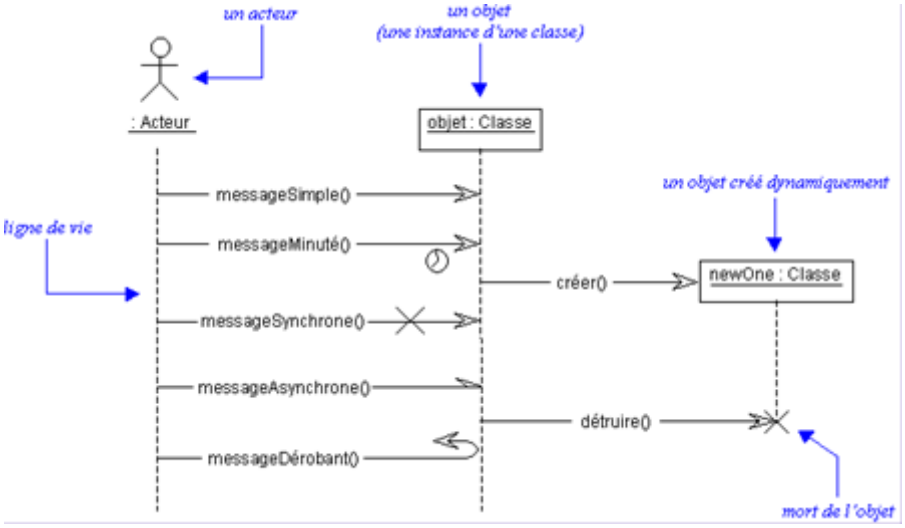
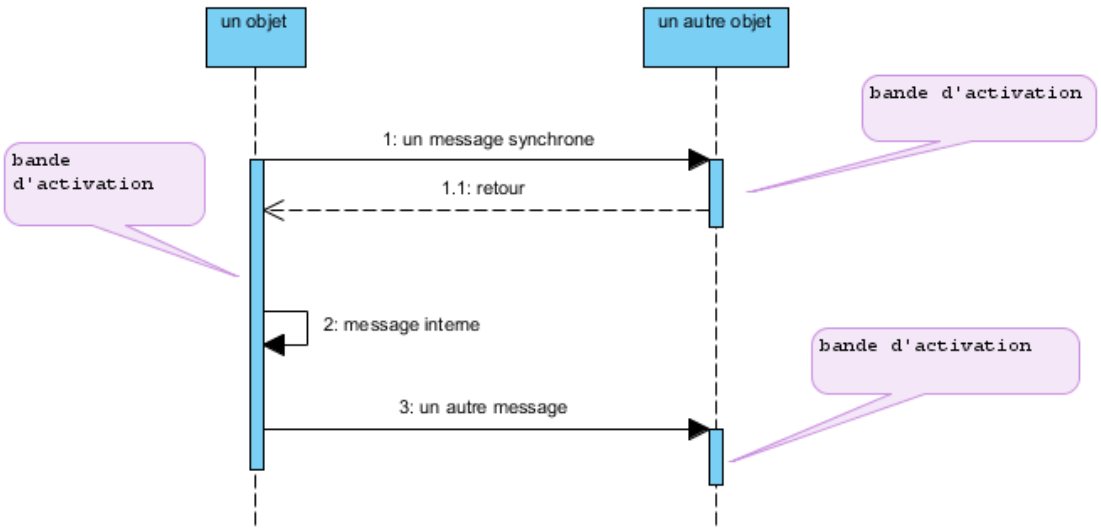
## Introduction :

- Les diagrammes de séquences permettent de représenter des interactions (des messages) entre objets (objet au sens large du terme : instances de classe, acteurs, autres systèmes…) selon un point de vue temporel ; on y met l'accent sur l'ordre d'envoi des messages.
- Axe horizontal : les objets (également appelés participants) ; la disposition des objets sur l'axe horizontal n'a pas d'importance.
- Axe vertical : c'est la ligne de vie ; représente le temps qui s'écoule (de haut en bas).
- Les diagrammes de séquences peuvent servir à illustrer un cas d'utilisation, les interactions entre objets dans du code, …
- Vue dynamique du système (ici notion de temps).

## Représentation graphique :




Remarque : ne pas confondre la période d'activation d'un objet avec sa création ou sa destruction. Un objet peut être actif plusieurs fois au cours de son existence.

## Nature des messages échangés :

- **Message simple** : message dont on ne spécifie aucune caractéristique d'envoi ou de réception particulière.
- **Message synchrone** : bloque l'expéditeur jusqu'à prise en compte du message par le récepteur (envoi d'un message de retour du destinataire vers l'expéditeur)
- **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.
- **Message interne** (self message) : appelle une méthode de l'objet lui-même.
- **…**

Remarque : la représentation graphique des messages peut être différente suivant les logiciels utilisés pour la création du diagramme et la version UML utilisée.

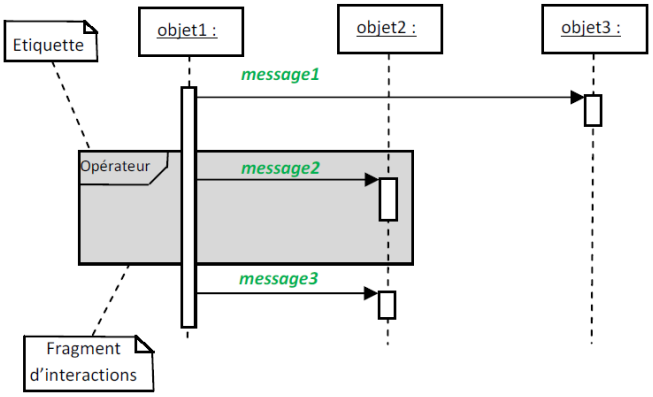
## Les fragments combinés :

Dans un diagramme de séquences, les fragments combinés permettent de représenter :

- des blocs optionnels,
- des blocs conditionnels,
- des blocs itérés,
- …

➔ **Des boucles et des tests**

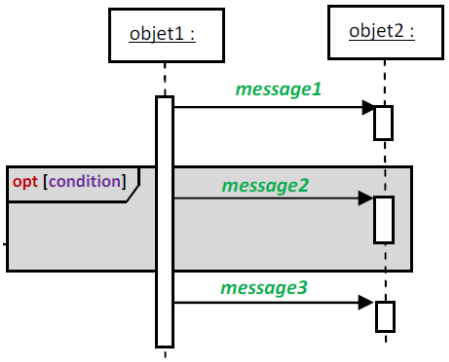
## Représentation graphique d'un fragment :




Remarque : l'opérateur précise le type de fragment.

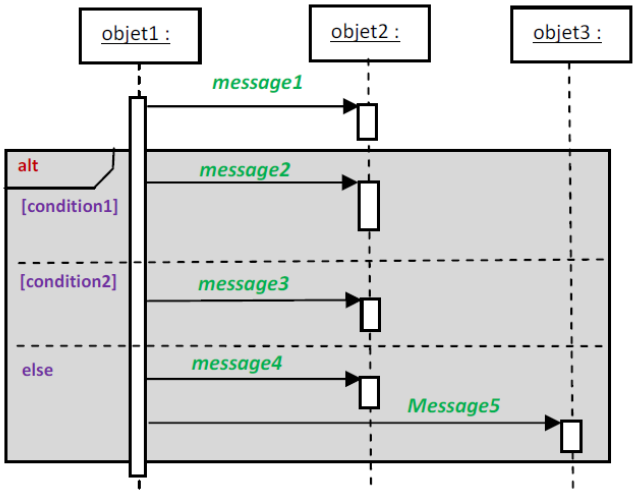
## Les fragments les plus utilisés :

Fragment **opt** (option) : correspond en pseudo-code à la structure : « **si…alors** »



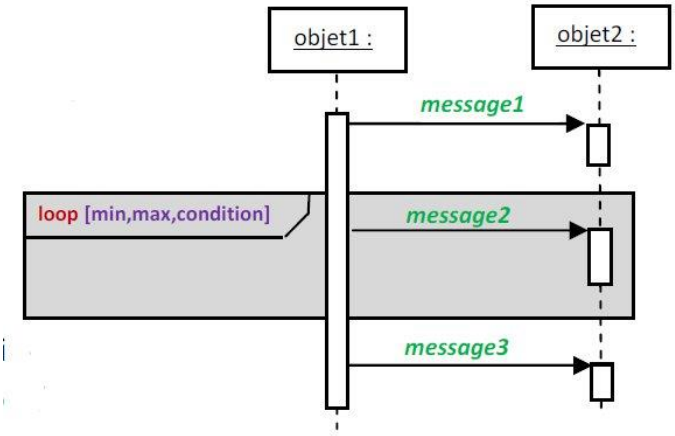

→Le message est envoyé si la condition est vraie.

Fragment **alt** (alternative) : correspond en pseudo-code aux structures
« **si…alors…sinon** » ou **« choix multiples →switch…case** »




condition 1 vraie → exécution de message2
condition 2 vraie → exécution de message3
si aucune condition n'est vraie → exécution du message4 puis message5

Fragment **loop** (boucle) : correspond en pseudo-code aux structures « **Tant Que Condition… Alors…Fin Tant Que** » ou « ***Pour…*** »




On exécute le fragment un certain nombre de fois. Chaque paramètre (***min***, ***max*** et ***condition***) est optionnel→ création de structures C comme **while** et **for**.