I/ Introduction

Le diagramme de séquence décrit la dynamique du système. À moins de modéliser un très petit système, il est difficile de représenter toute la dynamique d'un système sur un seul diagramme. Aussi la dynamique globale sera représentée par un ensemble de diagrammes de séquence, chacun étant généralement lié à une sous-fonction du système.

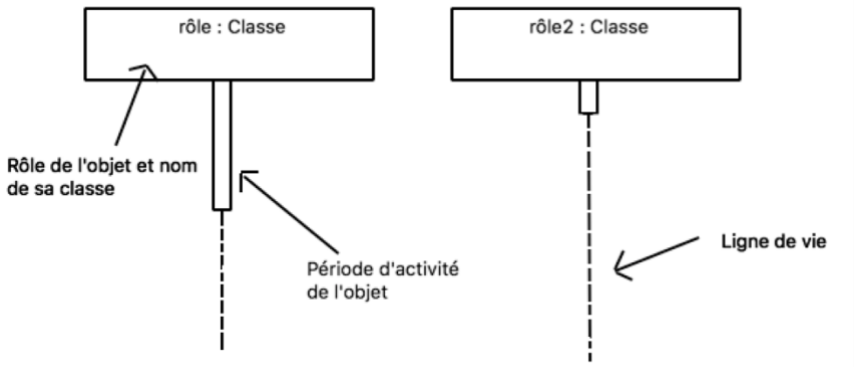
Le diagramme de séquence décrit les interactions entre un groupe d'objets en montrant, de façon séquentielle, les envois de message qui interviennent entre les objets. Le diagramme peut également montrer les transmissions de données échangées lors des envois de messages.

Pour interagir entre eux, les objets s'envoient des messages. Lors de la réception d'un message, un objet devient actif et exécute la méthode de même nom. Un envoi de message est donc un appel de méthode.

II/ La ligne de vie d'un objet

Comme il représente la dynamique du système, le diagramme de séquence fait entrer en action les instances des classes intervenant dans la réalisation de la sous-fonction qui lui est liée. À chaque instance est associé une ligne de vie qui montre ses actions et réactions, ainsi que les périodes pendant lesquelles elle est active, c'est à dire où elle exécute l'une de ses méthodes.

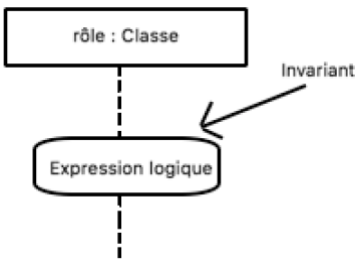
La représentation graphique de la ligne de vie est illustrée ci-dessous :



La notation « rôle : Classe » *représente le rôle d'une instance suivi du nom de sa classe. Le rôle de l'instance est optionnel si une seule instance de cette classe participe au diagramme de séquence. Le nom de la classe peut également être omis dans le cas d'une étape préliminaire de la modélisation, mais il doit être spécifié dès que possible.*

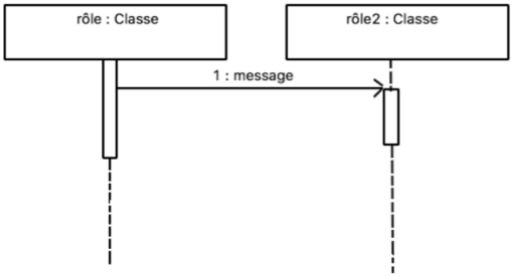
Un diagramme de séquence contient plusieurs lignes de vie car il traite des interactions entre plusieurs objets.

La ligne de vie peut commencer par l'introduction d'un invariant d'état qui est une expression logique qui doit être vérifiée tout au long du déroulement de la ligne de vie.



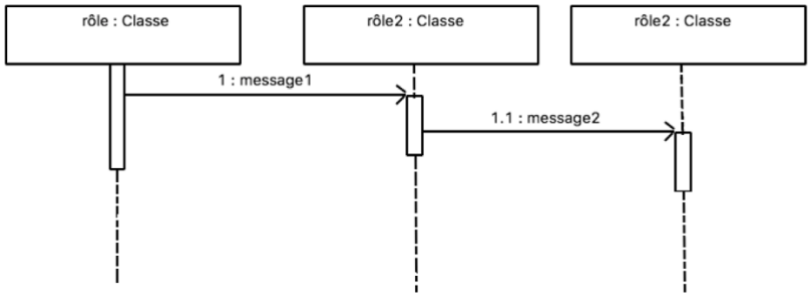
III/ Envoi de message

Les envois de messages sont représentés par des flèches horizontales reliant la ligne de vie de l'objet destinataire.



Dans la figure ci-dessus, l'objet de gauche envoie un message à l'objet de droite. En programmation, ce message donne lieu à l'exécution de la méthode message de l'objet de droite, ce qui provoque son activation. Le nom du message n'est pas obligatoire, il est possible de l'omettre lors de la spécification d'un envoi de message. Dans ce cas, le nom du message est remplacé par le caractère *.

Les messages sont numérotés séquentiellement, à partir de 1. Si un message est envoyé alors que le traitement du précédent n'est pas terminé, il est possible d'utiliser une numérotation composée (voir figure ci-dessous) où l'envoi du message 2 intervient pendant l'exécution du message 1.

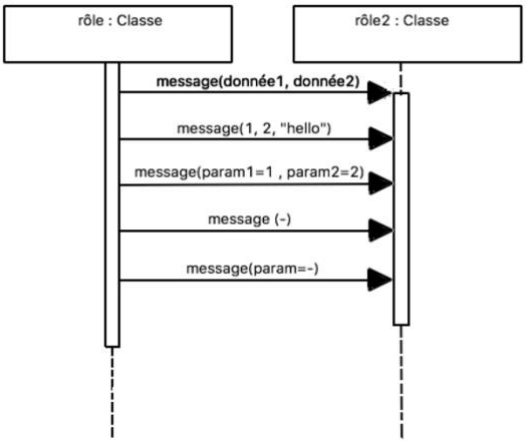


La numérotation des messages n'est pas obligatoire. Elle reste toutefois pratique pour montrer les activations imbriquées.

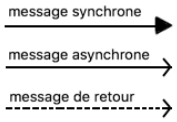
La transmission de données est également possible ; elle est représentée par des paramètres transmis avec le message. La valeur de chaque paramètre transmis est fournie par la valeur de variables comme donnée1 et donnée2 ou par la valeur de constantes. Par défaut, la valeur des paramètres est fournie dans l'ordre de ceux-ci. Il est aussi possible de nommer les paramètres afin de leur affecter leur valeur. L'utilisation du caractère – signifie que la valeur de paramètres n'est pas spécifiée. Ainsi, dans l'exemple message (-), la valeur d'aucun paramètre n'est spécifiée.

Il existe différents types d'envois de message. La figure ci-dessous en fournit une explication graphique.

Le message synchrone est le plus fréquemment utilisé. Dans ce cas, l'expéditeur attend que l'activation de la méthode invoquée chez le destinataire soit terminée avant de continuer son activité.

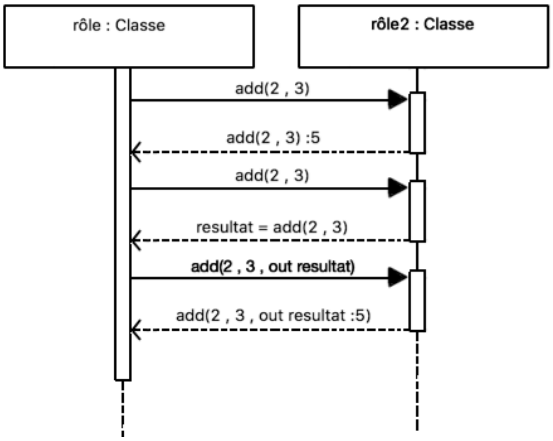


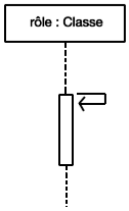
Dans le cas du message asynchrone, l'expéditeur n'attend pas la fin de l'activation de la méthode invoquée chez le destinataire. Ceci se produit lors de la modélisation d'un système où les objets peuvent fonctionner en parallèle (cas des systèmes multithreads où les traitements sont effectués en parallèle).



Le message de retour de l'invocation d'une méthode n'est pas systématique, toutes les méthodes ne retournant pas un résultat.

La figure ci-contre illustre des envois de message avec un message de retour qui transmet un résultat, soit comme résultat d'une fonction, soit comme paramètre de retour.

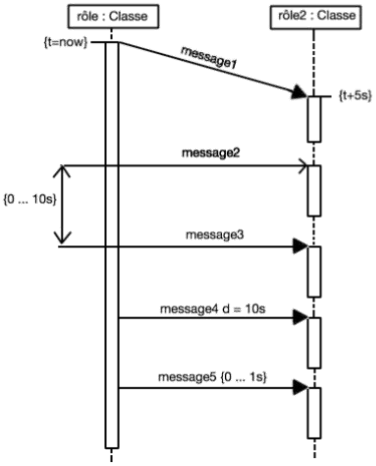




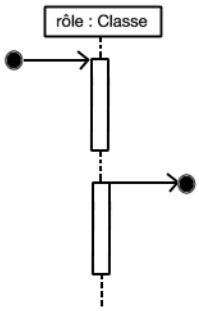
Un objet peut s'envoyer un message à lui-même. La représentation d'un tel message est illustré à la figure ci-dessous.

L'introduction du temps se fait au travers de conditions illustrées :

- Le message 1 est envoyé à now (temps courant) et il arrive 5 secondes plus tard (t+5s) ;
- Le message 3 est envoyé au plus tard 10 secondes après le message 2 ;
- La transmission du message 4 dure 10 secondes. Le paramètre d est utilisé pour fixer la durée de la transmission d'un message ;
- La durée de transmission du message 5 est comprise entre 0 et 1 seconde.



La figure ci-dessous illustre un message trouvé et un message perdu. Le premier message est un message trouvé. Ceci signifie que le récepteur de message est connu, mais pas l'émetteur. Le second message est un message perdu. L'émetteur du message est connu, mais pas le récepteur.



IV/ La création et la destruction d'objets

Le diagramme de séquence décrivant la dynamique d'un système, celle-ci contient fréquemment des créations et des destructions d'objets.

La création d'objet est représentée par un message spécifique qui donne lieu au début de la ligne de vie du nouvel objet.

La destruction d'objet est un message envoyé à un objet existant et qui donne lieu à la fin de sa ligne de vie. Il est représenté par une croix.