Chap6 : Diagramme de Communication et de Séquence

1. Diagramme de communication

a) Introduction

Les diagrammes de communication montrent des interactions entre les objets. Ils permettent de représenter le contexte d'une interaction car on peut y présenter les états des objets qui interagissent.

Un diagramme de communication rend compte de l'organisation spatiale des participants à l'interaction, il est souvent utilisé pour illustrer un cas d'utilisation ou pour décrire une opération. Le diagramme de communication aide à valider les associations du diagramme de classe en les utilisant comme support de transmission des messages.

b) Syntaxe des messages

Dans un diagramme de communication, les messages sont généralement ordonnés selon un numéro de séquence croissant.

Un message est, habituellement, spécifié sous la forme suivante:

['['<cond>']' [<séq>] [*[||] ['['<iter>']']] :] [<var> :=] <msg>([<par>])

<cond>
est une condition sous forme d'expression booléenne entre crochets.

<séq>
est le numéro de séquence du message. On numérote les messages par envoi et sous-envoi désignés par des chiffres séparés par des points : ainsi l'envoi du message 1.4.4 est postérieur à celui du message 1.4.3, tous deux étant des conséquences (*i.e.* des sous-envois) de la réception d'un message 1.4. La simultanéité d'un envoi est désignée par une lettre : les messages 1.6*a* et 1.6*b* sont envoyés en même temps.

<iter>
spécifie (en langage naturel, entre crochets) l'envoi séquentiel (ou en parallèle, avec ||) de plusieurs message. On peut omettre cette spécification et ne garder que le caractère * (ou *||) pour désigner un message récurrent envoyé un certain nombre de fois.

<var>
est la valeur de retour du message, qui sera par exemple transmise en paramètre à un autre message.

<msg>
est le nom du message.

<par>

désigne les paramètres (optionnels) du message.

Cette syntaxe un peu complexe permet de préciser parfaitement l'ordonnancement et la synchronisation des messages entre les objets du diagramme de communication . La direction d'un message est spécifiée par une flèche pointant vers l'un ou l'autre des objets de l'interaction, reliés par ailleurs avec un trait continu (connecteur).

3 : bonjour () : Ce message a pour numéro de séquence «3 »

[Heure=midi] 1 : manger () : ce message n'est envoyé que s'il est midi.

1.3.6*: ouvrir() : ce message est envoyé de façon séquentiel un certain nombre de fois.

3/*II (i :=1..5] :fermer() : Représente l'envoie en parallèle de 5 messages ; Ces messages ne seront envoyés qu'après l'envoie du message 3.

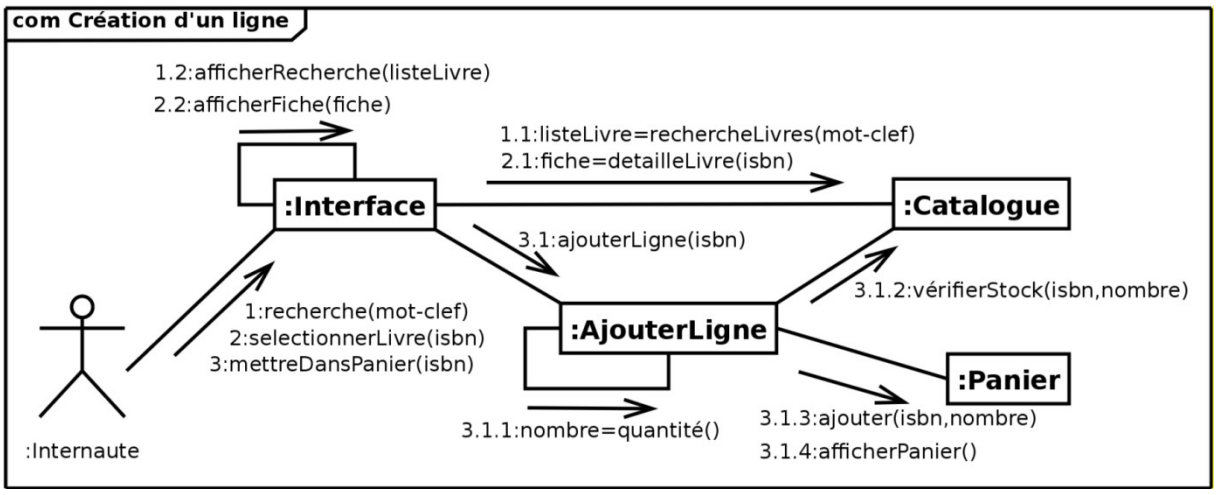
1.3,2.1/[t<10s].2.5 :=âge=demande [nom, prénom] Ce message 2.5 ne sera envoyé qu'après les messages 1.3 et 2.1 et que si t<10s

1.3/[disk full] 1.7.a* :deleteTempfile

1.3/ [disk full] 1.7.b : reduceswapfile(20%) : Ces deux derniers messages ne seront envoyés qu'près l'envoie du 1.3 et si la condition disk full est réalisée. Si cela est le cas, les messages 1.7.a et 1.7.b seront envoyé simultanément. Plusieurs messages 1.7.a peuvent être envoyés.

c) Exemple

Diagramme de communication illustrant la recherche puis l'ajout, dans son panier virtuel, d'un livre lors d'une commande sur Internet.



2. Diagramme de séquence

a) Sémantique

Les diagrammes de séquences permettent de représenter des collaborations entre objets selon un point de vue temporel. On y met l'accent sur la chronologie des envoies de messages.

Contrairement au diagramme de communication, on y décrit par le contexte ou l'état des objets, la représentation se concentre sur l'expression des interactions.

Les diagrammes de séquence peuvent servir à illustrer un cas d'utilisation ou une opération.

L'ordre d'envoie des messages est déterminé par sa position sur l'axe vertical du diagramme; le temps s'écoule « de haut en bas » de cet axe.

La disposition des objets sur l'axe horizontal n'a pas de conséquence pour la sémantique du diagramme.

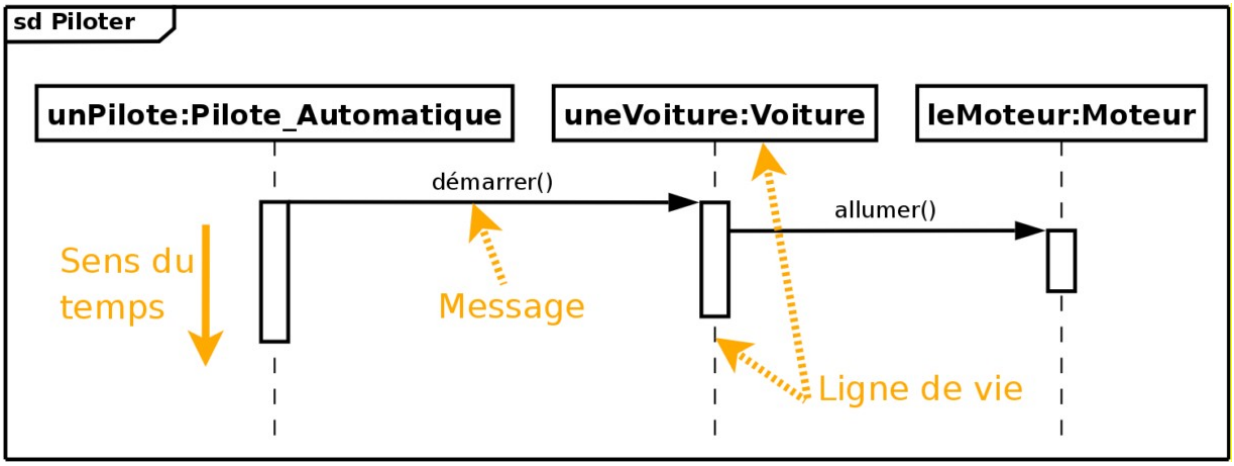
Sur un diagramme de séquence il est aussi possible de représenter de manière explicite les différentes périodes d'activité d'un objet superposé à la ligne de vie de l'objet.

b) Représentation des messages

Un message définit une communication particulière entre des lignes de vie. Plusieurs types de messages existent, les plus communs sont :

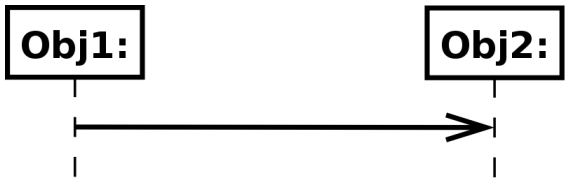
- l'envoi d'un signal ;
- l'invocation d'une opération ;
- la création ou la destruction d'une instance.

c) Exemple



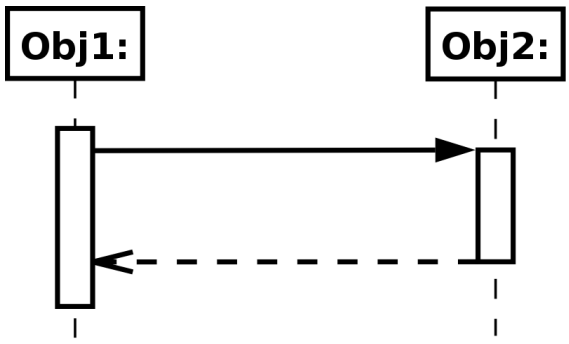
d) **Messages asynchrones**

Ce type de message n'interrompt pas l'exécution de l'expéditeur. Le message envoyé peut être pris en compte par le récepteur à tout moment ou ignoré (jamais traité).



Un signal est, par définition, un message asynchrone. Ils n'attendent pas de réponse et ne bloquent pas l'émetteur qui ne sait pas si le message arrivera à destination, le cas échéant quand il arrivera et s'il sera traité par le destinataire.

e) **Messages synchrones**



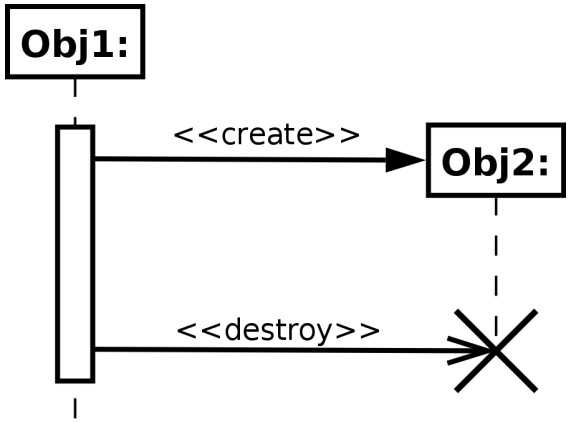
Bloque l'expéditeur jusqu'à prise en compte du message par le destinataire. Le flot de contrôle passe de l'émetteur au récepteur (l'émetteur devient passif et le récepteur devient actif) à la prise en compte du message.

L'invocation d'une opération est le type de message le plus utilisé en programmation objet. L'invocation peut être asynchrone ou synchrone. Dans la pratique, la plupart des

invocations sont synchrones, l'émetteur reste alors bloqué le temps que dure l'invocation de l'opération.

Le message synchrone peut être suivi d'une réponse qui se représente par une flèche en pointillé.

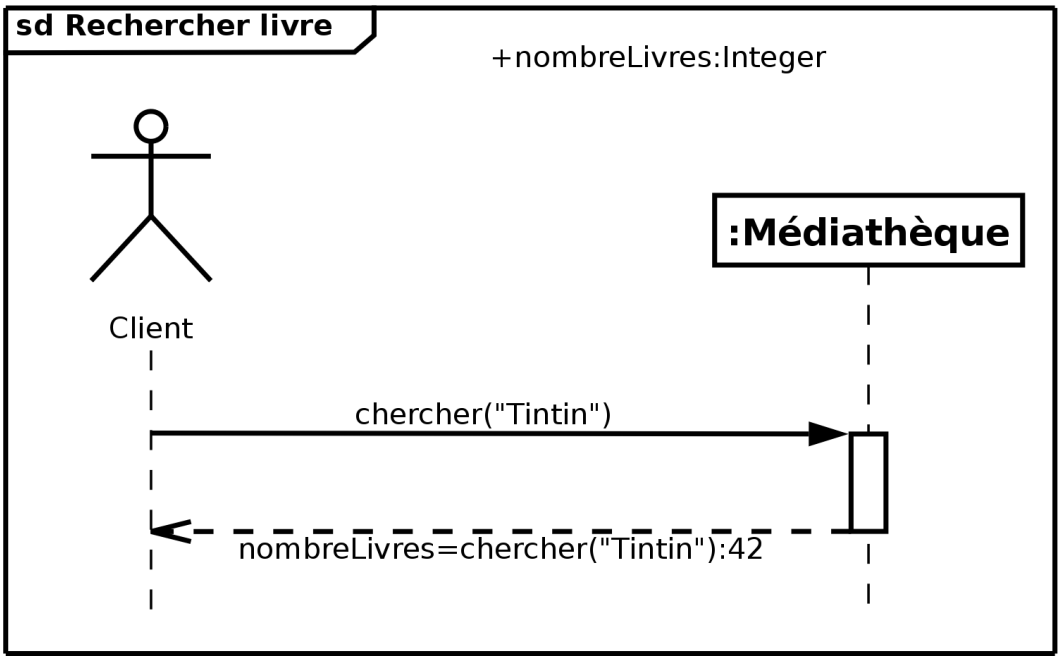
Messages de création et destruction d'instance



La création d'un objet est matérialisée par une flèche qui pointe sur le sommet d'une ligne de vie.

La destruction d'un objet est matérialisée par une croix qui marque la fin de la ligne de vie de l'objet. La destruction d'un objet n'est pas nécessairement consécutive à la réception d'un message.

f) **Syntaxe des messages et des réponses**

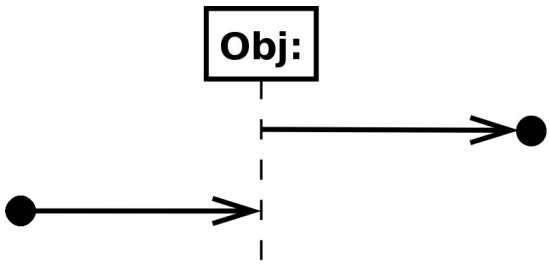


Dans la plupart des cas, la réception d'un message est suivie de l'exécution d'une méthode d'une classe. Cette méthode peut recevoir des arguments et la syntaxe des messages permet de transmettre ces arguments. La syntaxe de ces messages est la même que pour un diagramme de communication, excepté deux points :

- la direction du message est directement spécifiée par la direction de la flèche qui matérialise le message, et non par une flèche supplémentaire au dessus du connecteur reliant les objets comme c'est le cas dans un diagramme de communication ;
- les numéros de séquence sont généralement omis puisque l'ordre relatif des messages est déjà matérialisé par l'axe vertical qui représente l'écoulement du temps.

g) **Message perdu et trouvé**

Représentation d'un message perdu et d'un message trouvé.



Un message complet est tel que les événements d'envoi et de réception sont connus. Comme nous l'avons déjà vu, un message complet se représente par une simple flèche dirigée de l'émetteur vers le récepteur.

Un message perdu est tel que l'événement d'envoi est connu, mais pas l'événement de réception. Il se représente par une flèche qui pointe sur une petite boule noire.

Un message trouvé est tel que l'événement de réception est connu, mais pas l'événement d'émission. Une flèche partant d'une petite boule noire représente un message trouvé.