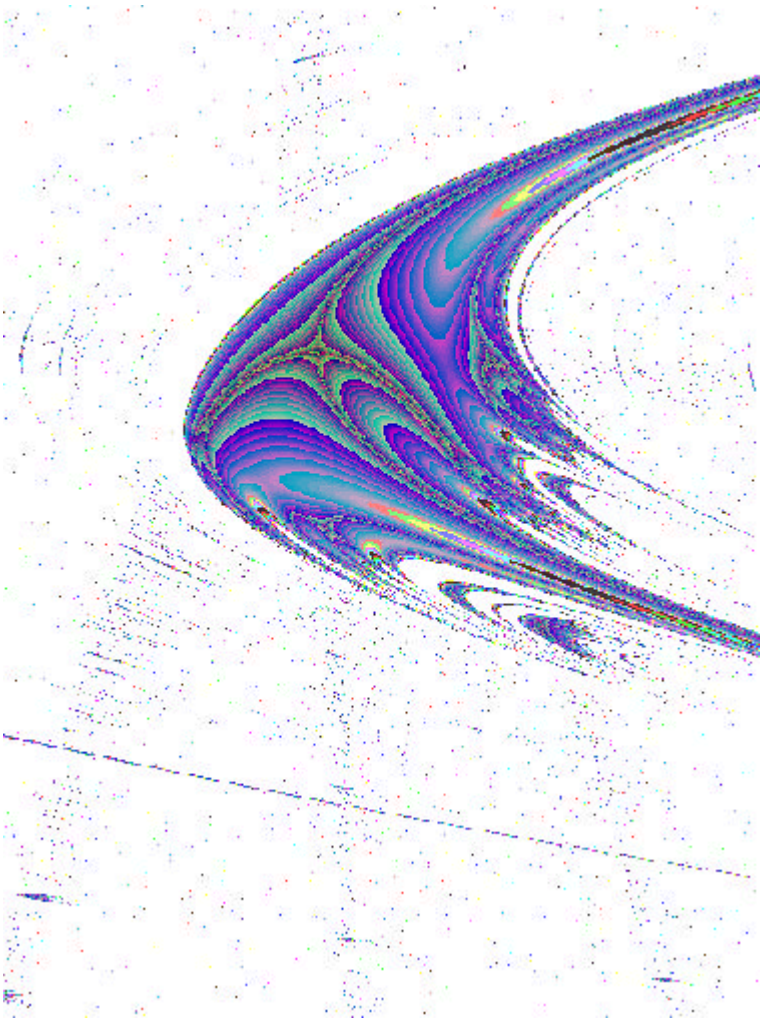
CHAPITRE 10

# *UML, les diagrammes d'objets*



## 10.1 *But du diagramme d'objets*

Alors que le diagramme de classes montre les relations existant entre les diverses classes du système en cours de développement (puis terminé), il ne peut pas mettre en évidence les relations existant à un instant déterminé de la vie du système, entre les diverses instances des classes composant le système. Il ne peut montrer qu'une instance particulière (ayant des valeurs d'attributs bien définies) est nécessaires dans un cas d'utilisation particulier.

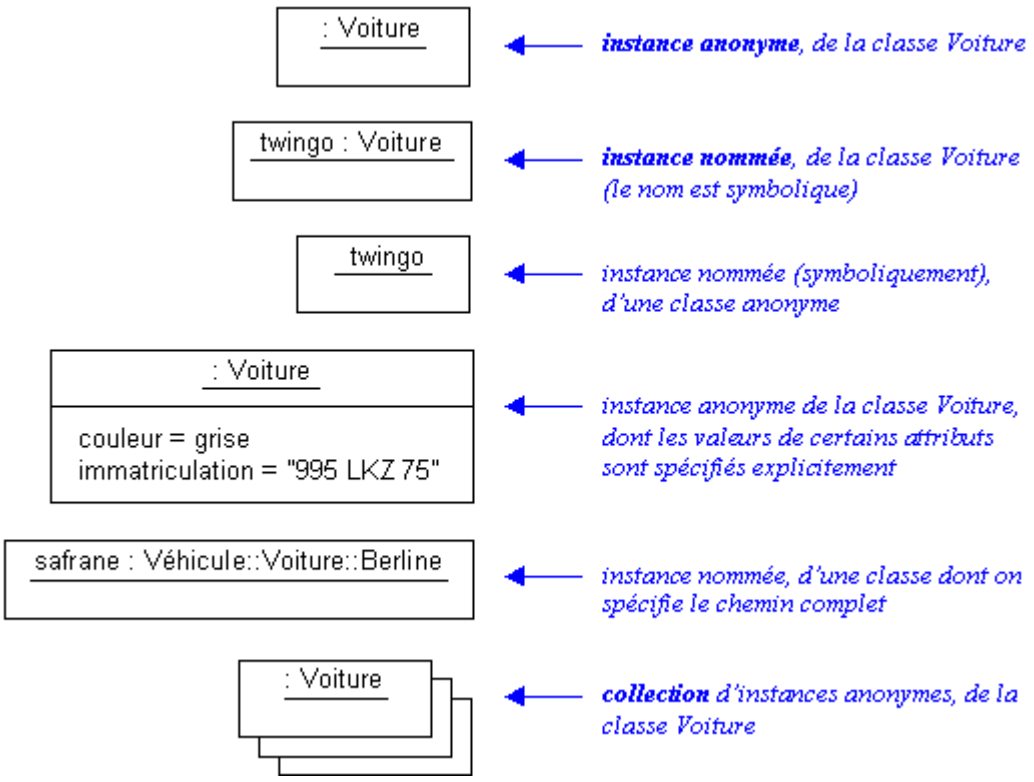
C'est là le rôle du diagramme d'objets. Le diagramme d'objets ne permet pas de générer du code : il ne s'agit que d'un outil de documentation et de communication (ce qui au demeurant n'est déjà pas si mal que cela).

### 10.1.1 *La notation*

Une instance est définie par un rectangle nommé correspondant en principe au nom de l'instance dans le cours du programme (mais ce n'est aucunement une obligation). La dénomination peut mentionner le nom de la classe, le nom de l'instance, ou les deux.

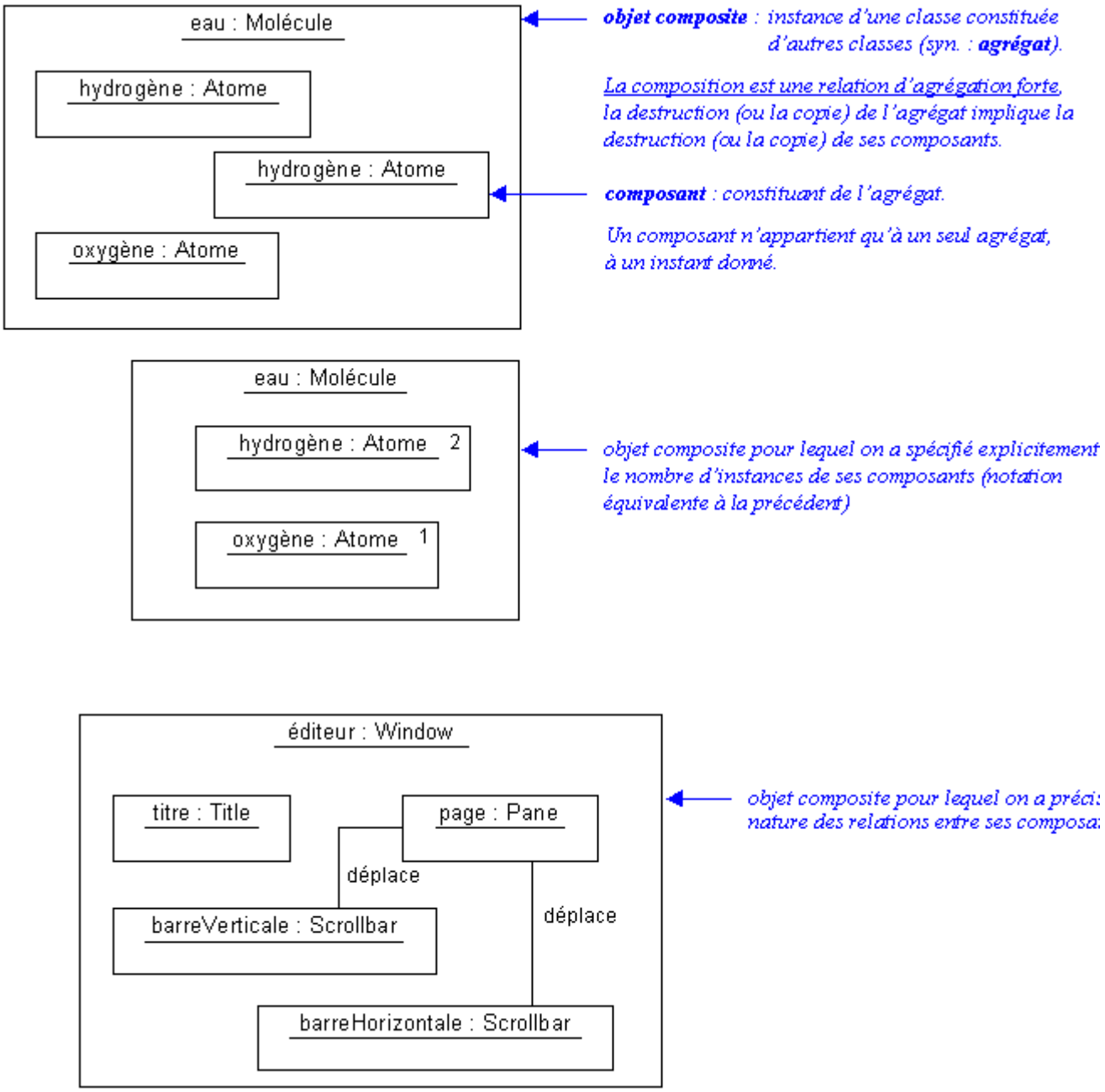
Lorsque des valeurs particulières d'attributs sont nécessaires, elles seront documentées dans le diagramme d'objets (figure10.1, page108).

**FIGURE 10.1 Exemples de notations d'objets**



Un objet peut être composite, donc constitué d'autres objets. Cette décomposition apparaît normalement dans le diagramme des classes (et devrait de toutes façons être documenté dans ce diagramme!), mais la vue des objets permet de spécifier l'état de certaines composantes dans un cas d'utilisation particulier. Ainsi, si un objet nécessite des collaborations avec d'autres objets dans un état bien particulier, ceci peutm être documenté dans le diagramme d'objets.

**FIGURE 10.2 Exemples de diagrammes d'objets**



### 10.1.2 *Utilité du diagramme d'objets*

Il n'est généralement pas possible de définir tous les diagrammes d'objets possibles pour un système, et ce serait assez fastidieux et vain que d'esayer. Par contre, le diagramme d'objets permet de préciser certains cas d'utilisation particulièrement difficiles : il est aussi très utile lors de la phase de débogage de l'application, puisqu'il définit exactement quel devrait être l'état de chaque objet impliqué dans le cas d'utilisation.

**FIGURE 10.3 Exemple de diagramme d'objets**