Conception Object

Le paradigme des objets

Alain Plantec

UBO

17 septembre 2012

Qu'est ce qu'un objet ?

Langages Classe-Instance :

- C'est une *entité* constitué d'un **état** privé et qui sait répondre à des **message**s.
  - **État** variable encapsulée
  - **Message** invocations de service d'un objet à un autre (nom d'une méthode avec éventuellement des arguments)
  - **Interface** ensemble des messages reconnus par un objet
  - **Méthode** le code que doit exécuter l'objet lors de la réception d'un message

Le modèle objet *Classe-Instance*

Le système

Un système se constitue d'un ensemble d'objets qui communiquent entre eux par envois de messages (i.e., invocation de méthodes)

- Un objet est l'instance d'une classe.
- Une classe définit le comportement de ces instances au moyen de méthodes et leur structure au moyen de variables d'instances.
- Chaque classe hérite de la description de son comportement et de sa structure à partir d'une seule super-classe (Smalltalk)
- Une classe particulière (Object) est la racine de l'arbre d'héritage (la classe qui définit le comportement commun à tous les objets du système).
- Les classes sont aussi des instances d'une classe (méta-classe).

Notion d'Interface

C'est l'ensemble des messages qu'un objet peut comprendre - Ensemble des services qu'un objet sait rendre.

- Un objet comprend au minimum une interface mais peut en comprendre plusieurs ;
- Une interface peut ne pas être mise en œuvre du tout ou bien être mise en œuvre par plusieurs classes ;
- Un service se définit par sa signature ;
- Attention : le concept d'*interface* est indépendant de celui d'*héritage*.

Anatomie d'un objet

Point de vue externe

En principe :

- Seules les interfaces qu'un objet met en œuvre sont connues (utilisées par les autres).
- Un objet client sait comment invoquer les services et quel type de résultat on obtient,
- mais, ne sait pas de quoi l'objet est fait, quelle est sa structure,
- et ne sait pas comment il fait pour répondre aux messages.

En pratique : On connait les interfaces mises en oeuvre

- par la documentation (commentaires),
- par les exemples et les tests et ...
- par la lecture du code !

Anatomie d'un objet

Point de vue interne

Spécification de la structure et du comportement

La classe comprend la définition des variables d'instance (structure) et spécifie le comportement exécuté par ses instances lors de la réception des messages.

- Une signature par message compris par les instances.
- Pour chaque signature, la classe comporte le code correspondant.
- Le code gère et maintient l'état de l'objet.
- L'état d'un objet se constitue de l'ensemble des valeurs de ses variables d'instance.

Attention : L'état d'un objet doit toujours être valide !

L'instanciation

Mécanisme fondamental : toute classe met en oeuvre ce service pour créer des instances conformes.

Deux étapes :

1. Réservation d'un espace de vie pour le nouvel objet et attribution d'un identifiant unique (la référence).
2. Initialisation des variables d'instance (pour que l'objet débute sa vie dans un état valide).

Constructeur

Permet d'indiquer comment initialiser les variables d'instances (étape 2)

Instanciation et identité des objets

L'identité est une propriété fondamentale d'un objet qui lui est attribuée lors de son instanciation

- A un moment T, il n'existe pas deux objets ayant même identité.
- Attention, la notion d'identité est différente de celle d'égalité entre objets : l'égalité est calculée à partir des valeurs des variables d'instance

Encapsulation

Un objet encapsule données et comportement.

- Les variables d'instance ne sont manipulées que par les objets auquels elles appartiennent ;
- L'exécution d'une méthode (réponse à un message) ne doit pas déstabiliser l'état interne.
- Composition entre objets : une variable d'instance peut contenir la référence d'un autre objet.

Héritage

Relation de spécialisation entre classes

Toute classe spécifiée par le développeur est définie par spécialisation ou raffinement d'une classe existante. On dit que **la nouvelle classe hérite de la classe raffinée**.

- Une classe hérite des variables d'instances et du comportement de sa super-classe.
- Une sous-classe peut être enrichie par de nouvelles variables d'instance et de nouvelles méthodes.
- Une sous-classe peut redéfinir ou masquer le comportement hérité (Attention)

Polymorphisme

Propriété fondamentale des langages à objet qui est liée à la notion d'interface.

- Le même message peut être compris par des objets de classes différentes.
- Pour un même message il peut y avoir plusieurs comportements différents (Attention).

Liaison dynamique

Le polymorphisme est mis en œuvre grâce à la liaison dynamique.

Le comportement d'un objet (le code exécuté en réponse à un message) est définit par la classe de l'objet qui répond. C'est l'objet qui reçoit le message qui décide.

- C'est un mécanisme dynamique (pendant l'exécution).
- Par opposition, si c'est la classe déclarée d'un objet qui définit le comportement, alors on parle de lien statique (liaison à la compilation)