PROGRAMMATION OBJET POUR LES BIOLOGISTES

# Introduction

programmation « spaghetti » : Basic, fortran

programmation structurée : C (lasagne)

Minsky : structuré : != du mode de pensée humain. Nos on identifie d’abord les objets puis on y associe la fonction. 🡪 programmation objet. (ravioli)

claude delannoy : Programmer en C++ (1998) 4e édition. Eyrolles

: Exercices en langage C++ - Programmation orientée objet – e édition revue et augmentée – Eyrolles 1996

Historique

1966 : *Simula* projet destiné à la simulation. C’était pas l’objectif du projet de mettre au point la POO. regroupement de données et de traitements en « modules ».

1972 : Smalltalk était le 1er langage objet. (créé par l’équipe de Simula).

Deux écoles à l’origine de la POO :

1. génie logiciel : comment faire un logiciel « propre » (sur lequel certaines pptés sont respectées)
2. inteligence artificielle : école représentée par les frames.

La difficulté de concevoir des logiciels de tailles importantes a imposé la conception structurée.

**Programmation classique** : séquentielle.

Instruction1

Instruction2

…

Instruction n

Toute modification dans le programme a forcément un impact sur l’ensemble du programme.

**Programmtion objet**

Si un objet est supprimé, les autres restent connectés entre eux (à l’image de notre société).

Hiérarchisation : héritage

Classe1

O1 O2 O3

O2.1

# Notion de classe dans la POO

La descritpion d’une famille d’objets qui partagent la même structure et le même comportement.

* Différents attributs sont commun à la classe : la composante statique. (Pour une classe Personne : le nom, l’âge, …)
* Il y a aussi toutes les fonctions, les actions qui peuvent être appliquées sur la classe. C’est la composante dynamique. (les méthodes marcher, parler, …).

Une classe encapsule les données et les fonctions qui lui sont liées.

Deux types de classes :

* abstraite : Personne, Véhicule
* concrète : Marie, Laguna

Exemple : Gestion d’un stock de magasin.

Essayer de créer des objets très génériques (très abstrait) :

Article

attributs : référence, désignation, prixHT, quantité.

Méthodes :

prixTTC() retourner taxe\*prixHT, prixTransport(), retirer(q), ajouter(q)

# Notion d’instance

Le représentant d’une classe est une instance. Chemise1 est une instance de la classe Article. Le mécanisme qui donne naissance à l’instance est l’instanciation. Normalement, quand on parle d’objet, on parle d’une instance (mais il y a beaucoup d’abus de langage). Par l’instanciation, on donne des valeurs aux attributs de la classe. La classe définit la structure de façon générique.

Une instance est un objet qui respecte les propriétés d’une classe. Elle est est le résultat du moulage de la classe.

La classe est la représentation conceptuelle qui permet de construire un objet. Elle joue le rôle de moûle. Le moulage est partiel : une instance évalue les attributs d’une classe.

Ceopendant, les méthodes et un type d’attribut (variable de classe (se rapproche un peu des variables globales en structural)) ont des comportements communs pour l’ensemble des oinstances de la classe.

Deux types d’attributs :

* attributs avec des comportements locaux (varaible d’instance)
* attributs avec des comportements partagés (variables de classes).

# Communication par envoi de message

POO : Ensemble d’instances qui communiquent entre elles *via* des messages. Un objet est une entité indépendante ont la structure est connue de lui seul.

Il ne peut pas, en principe, agir sur un autre objet qu’à travers les méthodes appartenant à l’interface de cet autre objet.

INTERFACE M2 M5

M1

M2

M3 M4 M5

Agir sur une méthode appartenant à l’interface d’un objet revient à envoyer un message, qui demande l’execution de la méthode en question.

Le receveur : chemise231

Le selecteur de la méthode : (sans le corps de la méthode) prixTTC()

Arguments : coordonnées

Quand une instance y, de la Classe Y, s’adresse à l’instance x, elle passe donc par une interface qui appartient à la Classe X.

Une relation entre deux instances veut dire qu’il a une relation entre les deux classes auxquelles elles appartiennent. Il faut qu’un protocole de communication entre les 2 classes soit pré-établit.

# La relation d’héritage (spécialisation)

Cette relation se fait entre plusieurs classes (et pas entre instances). Une sous-classe hérite d’une classe : la sous-classe récupère l’ensemble du comportement et de la structure de la classe mère.

Relations verticales :

* Héritage (spécialisation) : classe – classe
* Instanciation : classe – instance

super-classe

Etre-vivant

classe

spécialisation

généralisation

Homme

Personne

sous-classe

Relations horizontales :

* Agrégation (composition)

Quand on hérite, on reste dans le même concept. Quand on compose, les choses n’appartiennent pas forcément au meme concept, mais on les assemble pour former une classe composite.

Roue

Moteur

Carosserie

Voiture

Ici, Voiture est une classe composite et les 3 autres sont des classes composantes.

Ces deux notions sont un peu semblables et on les confond parfois.

Dans l’héritage, il y a les notions d’inclusion, de connexion et de similarité. Dans la composition, on retrouve l’inclusion et la connexion. 🡪 Quand il y a un doute, se poser la question sur la similarité des concepts.

Héritage multiple : exemple : l’étudiant salarié hérite à la fois de la classe Etudiant et à la fois de la classe Salarié.

Relation de spécialisation est : Transitive et anti-symétrique (A donne B mais B ne donne pas A).

Collision sémantique (problème de conception du problème) : le vrai problème de l’héritage multiple.

Solution : donner 2 noms différents.

Parcours en I, en N, ou en Z : quand il y a un conflit, un choix de parcours doit être fait.

Surcharge

Redéfinition :quand on redéfinit la signature d’une méthode.

Surdéfinition : on change le code associé. Les deux méthodes ont le meme nom mais n’ont pas le meme code.

# La compilation

Une fois que le compilateur a anlysé les classes, il analyse l’executable : le main (instanciation + méthodes).

POO : instances qui communiquent entre elles.

Deux niveaux :

Niveau modélisation : modéliser le problème. Analyser toutes les composantes… : **UML** (Unified Language Model) : on retrouve les concepts objets traduits sous forme de diagramme.

Niveau programmation. : **JAVA**

# U.M.L.

Les Objets (Instances)

Objet = Etat + Comportement + Identité

Etat : structure. Regroupe les valeurs des attributs de l’objets.

Comportement : méthodes.

Identité : nom de l’instance.

Les objets peuvent être regroupés en 3 catégories selon la nature de leur comportement :

* acteurs : toujours à l’origine d’une intéraction : ce sont des objets actifs.
* serveurs : ne sont jamais à l’origine d’une intéraction, mais sont toujours destinataires des messages : ce sont des objets passifs.
* agents : cummulent les caractéristiques des acteurs et des serveurs : ils peuvent intéragir.

Catégories des messages

* **message simple :** convient aux système à un seul flot d’execution, dans lesquels un seul objet à la fois est actif.
* **message synchrone :** il ne déclenche une opération que lorsque le desinataire accepte le message. Une fois le message envoyé, l’expéditeur est bloqué jusqu’à ce que le destinataire accepte le message. L’expéditeur accepte d’attendre.
* **message dérobant :** déclenche une opération seuleemnt si le destinataire s’est préalablement mis en attente de message. Ce type de synchronisation correspond à une tolérance d’attente invers de celle de l’envoi synchrone. *i.e.* Le destinataire accepte d’attendre.
* **message minuté :** Bloque l’expéditeur pendant un temps donné, en attendant la prise en compte par le destinataire. L’expéditeur est libéré si la prise en compte n’a pas eu lieu au bout du temps spécifié dans la description de l’envoi.
* **message asynchrone :** il n’interrompt pas l’exectution de l’expédieur, l’expéditeur envoi le message sans savoir si le destinataire attend un message.

Les 3 diagrammes les plus utilisés en UML : diag d’objets, diag de séquence et diag de classe.

On peut définir une appli avec un ensemble de diagrammes, mais la cohérence entre les diagrammes n’est pas assurée.

Un enfant peut avoir 0 ou 2 parents, et un parent peut avoir 0 ou plusieurs enfants