PARIS DESCARTES

DUT INFORMATIQUE — Deux semaines — S1

BASES DE LA CONCEPTION ORIENTEE OBJET

TP N°1 – PROGRAMMATION DIRIGEE PAR LES MODELES

M. OUZIA

Objectifs du TP :

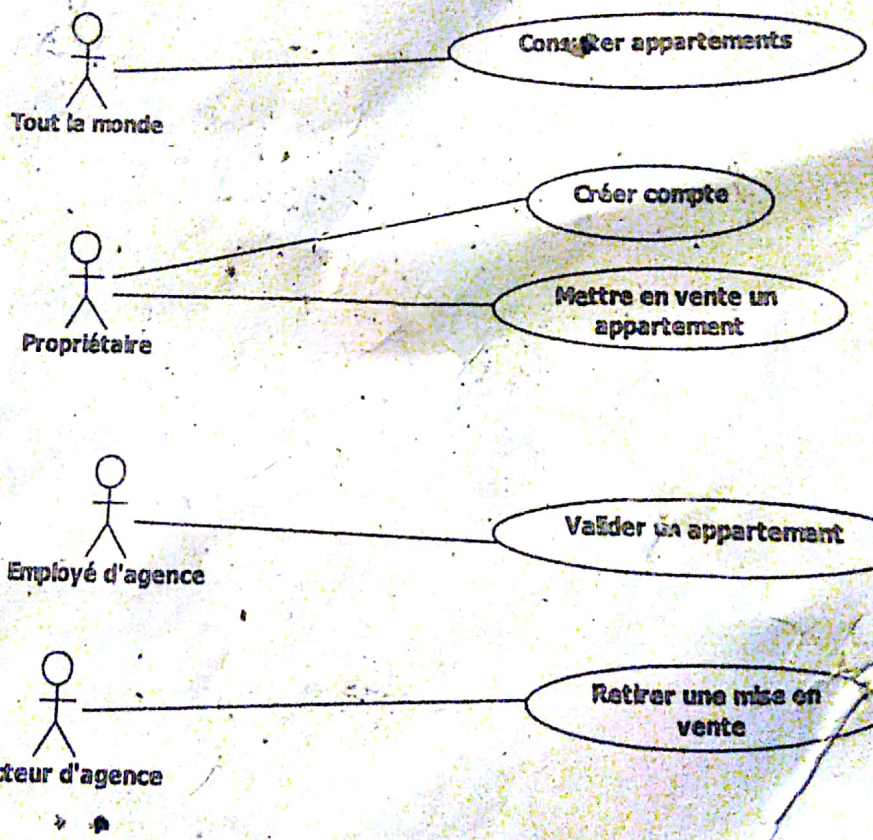
- Utiliser l'AGL Modelio pour dessiner et valider les modèles UML, notamment sur le plan syntaxique.
- Programmer des applications classes Java simples à partir de modèles UML (classes et séquence).

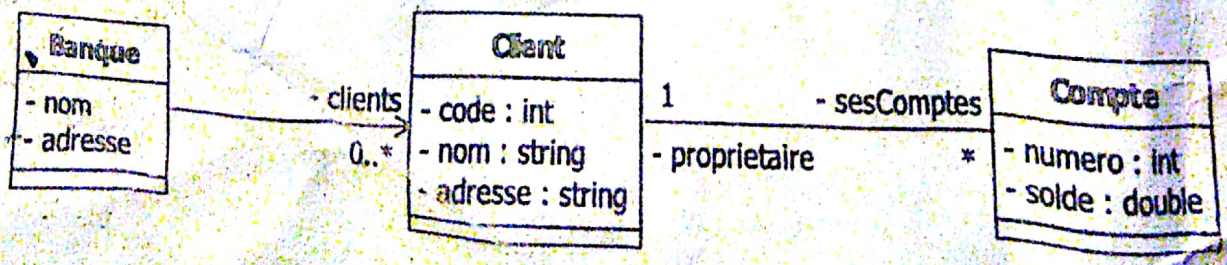
Environnement du TP :

Langage de modélisation : UML, Langage de programmation : Java, EDI : Modelio, Eclipse

Exercice 1 – « Atelier de Génie Logiciel » UML

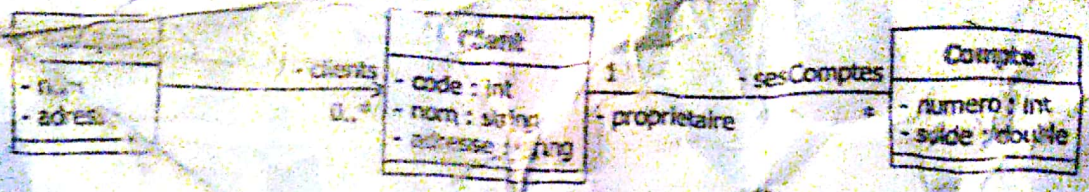
A l'aide de l'AGL *Modelio*, dessiner les diagrammes de cas d'utilisation et de classes UML suivants :





Exercice 2 – Gestion de comptes bancaires

Le diagramme de classes suivant gère les comptes bancaires des clients d'une banque :



Travail demandé

1. Ecrivez, sous Eclipse, le code Java structurel (sans les méthodes) correspondant au diagramme de classes ci-dessus.
2. Pour chaque fonction suivante, vous ferez un diagramme de séquence (il est déjà fait en TD4) directement sur Modelio et le code Java correspondant puis vous ajouterez les méthodes définies dans le diagramme de classes. Vous écrivez des tests automatiques JUnit pour chaque fonction.
   a) Créer la banque. On donnera le nom et l'adresse de la banque.
   b) Consulter les informations de la banque (nom, adresse, liste des clients).
   c) Changer l'adresse de la banque.
   d) Enregistrer un client.
   e) Rechercher un client par son code.
   f) Changer l'adresse d'un client identifié par son code.
   g) Ouvrir un nouveau compte pour un client. On donnera l'objet client.
   h) Ouvrir un nouveau compte pour un client. On donnera le code du client.
   i) Consulter le solde d'un compte appartenant à un client donné. On donnera le numéro du compte et l'objet client.
   j) Consulter le solde d'un compte appartenant à un client donné. On donnera le numéro du compte et le code du client.
   k) Consulter le solde du compte d'un numéro donné.
   l) Créditer une somme sur un compte.
   m) Débiter une somme sur un compte.
   n) Consulter le solde global d'un client. On donnera l'objet client.
   o) Calculer le solde global d'un client. On donnera le code du client.
   p) Calculer le solde total de tous les comptes de la banque.
   q) Donner le propriétaire du compte ayant le solde le plus élevé.
   r) Donner la liste des clients ayant au moins un compte débiteur (solde négatif).
   s) Fermer un compte. On donnera son numéro.
   t) Fermer un compte. On donnera l'objet compte à fermer.

Exercice supplémentaire

Refaire les mêmes questions sur l'application du TD n°2 exercice 4.

PARIS DESCARTES IUT

DUT INFORMATIQUE — Semaines du 14 et 21 mai 2018
BASES DE LA CONCEPTION ORIENTEE OBJET
TP N°3 – HERITAGE ET POLYMORPHISME — M. OUZIRI

Exercice 1 – Gestion des emprunts dans une bibliothèque

Il s'agit de réaliser une gestion simplifiée des emprunts dans une bibliothèque universitaire.

Les adhérents sont soit des étudiants de l'université soit des personnes extérieures. Les étudiants sont décrits par leur numéro d'adhérent, nom et nom du département d'études. Lorsqu'une personne extérieure est inscrite à la bibliothèque, on renseigne obligatoirement son nom et son adresse du domicile. Elle est ensuite enregistrée sous un numéro d'adhérent.

Pour la gestion des emprunts, seul le nombre d'emprunts effectués par les adhérents nous intéresse. Ce nombre est augmenté et diminué à chaque emprunt et retour.

Le nombre d'emprunts maximum autorisé diffère selon que l'adhérent est étudiant de l'IUT ou personne extérieure. Il est actuellement fixé à trois emprunts pour les étudiants de l'IUT contre deux pour les personnes extérieures. La bibliothèque voudrait pouvoir modifier ces limites à tout moment.

Travail demandé

1. Faire le diagramme de classes modélisant à cette gestion.
2. Coder le diagramme de classes (statique) en Java.
3. Faire un diagramme de séquence, le code en Java et les tests correspondants des fonctions suivantes :
   a. Enregistrer un adhérent.
   b. Consulter le type (étudiant, extérieur) d'un adhérent. La méthode *getType()* est polymorphe.
   c. Afficher un adhérent. La méthode *afficher()* (ou toString()) est polymorphe.
   d. Rechercher un adhérent par numéro. Le résultat est un objet Adhérent.
   e. Enregistrer un emprunt et un retour d'emprunt. Ces méthodes sont polymorphes.
   f. Consulter le nombre d'emprunts réalisés par un adhérent donné.
   g. Consulter le nombre d'emprunts effectués par les adhérents d'un type donné.

Exercice 2 – Agence immobilière

Une agence immobilière met en vente des biens immobiliers pour ses clients. Un bien immobilier est soit un appartement soit une maison individuelle. Il est enregistré à l'agence sous un numéro unique.

Lorsqu'un appartement est mis en vente, on renseigne le nom et le numéro de téléphone du propriétaire, l'adresse de l'appartement, le nombre de pièces, l'étage où il se situe dans l'immeuble et le montant de mise en vente. S'il s'agit d'une maison, on renseigne le nom et le numéro de téléphone du propriétaire, l'adresse de la maison, le nombre de pièces, le nombre d'étages, la présence (ou pas) d'un jardin et le montant de mise en vente.

Lorsqu'un appartement ou une maison est vendu, on enregistre la date et le montant réel de vente.

L'agence perçoit des frais sur chaque vente. Ces frais représentent un pourcentage du montant réel de la vente, dépendant du bien concerné. Ces frais s'élèvent actuellement à 3% sur la vente d'appartements et de 2% pour les maisons.

Travail demandé

1. Faire le diagramme de classes modélisant à cette situation.
2. Coder le diagramme de classes obtenu en Java.
3. Faire le diagramme de séquence puis coder en Java les fonctions suivantes :
   a. Enregistrer la mise en vente d'un bien.
   b. Enregistrer la vente d'un bien.
   c. Calculer les frais d'agence sur la vente d'un bien.
   d. Calculer le chiffre d'affaire de l'agence sur une période de gestion donnée.