UML Exercices de base

Exercice 1

Dessiner les diagrammes (d'objets, de classes) correspondant aux situations suivantes :

(a) La France est frontalière de l'Espagne. Le Canada est frontalier des Etats-Unis.

(b) Un polygone est constitué de points. Un point possède une abscisse et une ordonnée.

(c) Une médiathèque possède des médias, empruntables par les abonnés de la médiathèque.

(d) Un client demande une réparation. Une réparation est effectuée par un mécanicien. Elle nécessite des compétences. Un mécanicien possède des compétences.

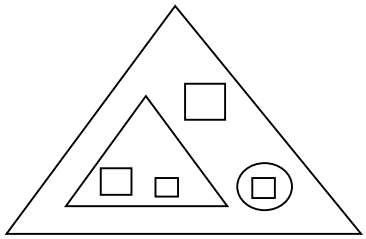
(e) Une galerie expose des œuvres, faites par des créateurs, et représentant des thèmes. Des clients, accueillis par la galerie, achètent des œuvres.

Dessiner les diagrammes (d'objets, de classes, de généralisation) correspondant à la situation suivante :

(f) Un bateau contient des cabines, occupées par des personnes qui effectuent des activités. Les personnes sont ou bien des guides, ou bien des animateurs, ou bien des passagers. Les guides expliquent des visites aux passagers et les animateurs animent des animations pour les passagers.

Exercice « FIGURE »

Le dessin ci-dessous représente des figures (triangles, carrés ou cercles) emboîtés. Les triangles contiennent une ou plusieurs figures. Les carrés ne contiennent rien. Les cercles contiennent exactement une figure. Les figures possèdent des « côtés ». On dira que les cercles ont un seul côté, les triangles trois côtés et les carrés quatre côtés.



1) A partir du texte précédent, déterminer les classes du domaine et dessiner le diagramme de généralisation des classes.

2) Dessiner un diagramme d'instances correspondant au dessin sans dessiner les instances de la classe "Côté".

3) Dessiner un diagramme de classes correspondant à la figure. Le diagramme comprendra les classes "Figure", "Cercle", "Carré", "Triangle" et "Coté" et des associations à déterminer.

4) Placer les ordres de multiplicité sur ce diagramme.

Exercice « JARDINIER »

Un jardinier effectue deux types de travaux : l'arrosage et le piochage. L'arrosage consiste à arroser des plantes (tulipes, eucalyptus ou géraniums) avec un outil (arrosoir ou tuyau) contenant de l'eau et le piochage consiste à retourner la terre avec un outil (pioche ou pelle) pour y mettre de l'engrais. Autrement dit, le jardinier utilise un outil (arrosoir, tuyau, pelle ou pioche) pour mettre une ressource (eau ou engrais) sur un objet naturel (terre ou plante) ; celui-ci est produit par un travail (arrosage ou piochage).

1) Dessiner le diagramme de généralisation des classes. On généralisera les classes du domaine avec une classe « Objet » .

2) Dessiner un diagramme d'objets correspondant au texte suivant :

jacques est un jardinier qui arrose 3 géraniums avec un arrosoir rempli d'eau.
jules est un jardinier qui pioche la terre avec une pelle pour y mettre de l'engrais.

3) Dessiner un premier diagramme de classes avec les classes Jardinier, Arrosage, Arrosoir, Eau, Geranium. Puis dessiner un autre diagramme de classes similaire au premier mais avec des classes plus générales : Jardinier, Travail, Outil, Ressource, ObjetNaturel.

4) Quelle difficulté posent les classe Eau, Terre, Engrais ?.

5) Placer les ordres de multiplicité sur les 2 diagrammes de classes précédents. Quelle remarque peut-on faire ?

Exercice « TRIATHLON »

Un triathlète utilise trois types de moyens de déplacement : la nage, le cyclisme et la course à pied. La nage consiste à nager une distance courte avec un maillot de bain dans un liquide (lac ou mer). Le cyclisme consiste à pédaler sur une distance longue avec un vélo sur une route. La course a pied consiste à courir une distance moyenne avec des chaussures sur une route. Autrement dit, le triathlète possède des équipements (vélo, maillot ou chaussure) pour effectuer une distance (courte distance, moyenne distance ou longue distance) sur un site (liquide ou route) en utilisant un moyen de déplacement (nage, cyclisme ou course à pied).

1) Dessiner le diagramme de généralisation des classes. On généralisera les classes du domaine avec une classe « Objet » .

2) Dessiner un diagramme d'objets correspondant au texte suivant :

thierry est un triathlète qui court à pied une moyenne distance sur la route départementale 3 avec ses chaussures.
timothée est un triathlète qui nage une courte distance dans la mer méditerranée avec un maillot de bain.

3) Dessiner un premier diagramme de classes avec les classes Triathlète, Nage, Maillot, Mer, CourteDistance. Puis dessiner un autre diagramme de classes similaire au premier mais avec des classes plus générales : Triathlète, MoyenDeplacement, Equipement, Site, Distance.

4) Placer les ordres de multiplicité sur les 2 diagrammes de classes précédents. Quelle remarque peut-on faire ?

Exercice « EXPRESSION »

Soit l'expression suivante : (X+Y/2)/(X/3+Y)

1) Identifier des classes pertinentes correspondant à cette expression.
2) Faire un diagramme de généralisation.
3) Faire un diagramme de classes.
4) Préparer le diagramme d'objets correspondant à l'expression.

Exercice « classification »

Classer les relations suivantes en généralisation, instanciation, agrégation, lien ou association. Argumenter les réponses.

(a) Un pays possède une capitale.
(b) Un philosophe qui dîne utilise une fourchette.
(c) Un joueur de rugby est un avant, un demi ou un arrière.
(d) Une équipe de rugby est composée de 8 avants, 2 demis et 5 arrières.
(e) Dédé programme son simulateur de vol en Java sur son PC.
(f) Java, C++, Eiffel sont des langages orientés objet.
(g) La Tour Eiffel a 3 étages et 3 millions de boulons.
(h) L'agrégation est un examen.

Exercice « brain-storming »

Préparer un diagramme d'objets montrant au moins 10 relations parmi les classes d'objets suivantes. Inclure les associations les agrégations et les généralisations. Placer les ordre de multiplicité.

(a) école, terrain de jeu, proviseur, conseil de classe, salle de classe, livre, élève, professeur, cafétéria, ordinateur, bureau, chaise, porte.
(b) château, douve, pont-levis, tour, fantôme, escalier, donjon, plancher, couloir, salle, fenêtre, pierre, seigneur, dame, cuisinier.
(c) Automobile, roue, frein, moteur, porte, batterie, silencieux, pot d'echappement.

Exercice « attributs méthodes de classe ou d'instance »

Pour chaque propriété des classes UML suivantes, nous avons volontairement omis de la souligner quand cela était nécessaire. En vous aidant du nom de la propriété et de sa signification, indiquer si la propriété est "d'instance" ou bien "de classe".

Objet
nombreObjets : int ; mesObjets : Liste ; numéro : int ;

Château
muraille : Muraille ; listeTours : Liste ; listeChateaux : Liste ;
void imprimerMuraille() ; void imprimerTours() ; void imprimerChateaux() ; void imprimer() ;

Exercice «de C++ à UML»

Le « Reverse engineering » est une technique de développement informatique qui consiste à concevoir un logiciel à partir du code source d'un logiciel existant. « engineering » signifie concevoir et « reverse » signifie que l'on travaille en sens inverse au sens habituel : en général dans le cycle en V, la conception précède la production du code source.

Cet exercice est un exemple de reverse engineering. A partir d'un code C++, vous devez retrouver la conception UML. On a les déclarations et définitions C++ suivantes:

```
#include <iostream.h>

class Objet { public:
      static int nombre_objets = 0;
      int numero ;
      Objet();
      ~Objet();
};

class Porte : public Objet { public:
      Porte();
      ~Porte();
};

class Batiment : public Objet { public:
      Porte * porte;
      int surface;
      Batiment(int);
      ~Batiment();
};

class Tour : public Objet { public:
      int hauteur;
      Muraille * muraille;
      Tour(int, Muraille *);
      ~Tour();
};

class Donjon : public Tour { public:
      Chateau * chateau;
      Donjon(Chateau *, int);
      ~Donjon();
};

class Muraille : public Objet { public:
      int longueur;
      Tour * tour1;
      Tour * tour2;
      Chateau * chateau;
      Muraille(int, int, int, Chateau *);
      ~Muraille();
};

class Chateau : public Batiment { public:
      Donjon * donjon;
      Muraille * muraille;
      Chateau(int, int, int, int, int);
      ~Chateau();
};

int Objet::nombre_objets = 0 ;

Objet::Objet() {
      cout << "++ Objet debut" << endl;
      nombre_objets++;
      numero = nombre_objets ;
      cout << "++ Objet fin" << nombre_objets << endl;
}
Objet::~Objet() {
```

```
	cout << "-- Objet debut " << endl;
	nombre_objets--;
	cout << "-- Objet fin " << endl;
}

Porte::Porte() {
	cout << "++ Porte debut" << endl;
	cout << "++ Porte fin" << endl;
}
Porte::~Porte() {
	cout << "-- Porte debut" << endl;
	cout << "-- Porte fin" << endl;
}

Batiment::Batiment(int s) {
	cout << "++ Batiment debut" << endl;
	porte = new Porte();
	surface = s;
	cout << "++ Batiment fin" << surface << endl;
}
Batiment::~Batiment() {
	cout << "-- Batiment debut" << endl;
	delete porte;
	cout << "-- Batiment fin" << endl;
}

Tour::Tour(int h, Muraille * m) {
	cout << "++ Tour debut" << endl;
	hauteur = h;
	muraille = m;
	cout << "++ Tour fin " << hauteur << endl;
}
Tour::~Tour() {
	cout << "-- Tour debut" << endl;
	cout << "-- Tour fin" << endl;
}

Donjon::Donjon(Chateau * c, int h) : Tour(h, NULL) {
	cout << "++ Donjon debut" << endl;
	chateau = c;
	cout << "++ Donjon fin" << endl;
}
Donjon::~Donjon() {
	cout << "-- Donjon debut" << endl;
	cout << "-- Donjon fin" << endl;
}
Muraille::Muraille(int l, int h1, int h2, Chateau * c) {
	cout << "++ Muraille debut" << endl;
	longueur = l;
	tour1 = new Tour(h1, this);
	tour2 = new Tour(h2, this);
	chateau = c;
	cout << "++ Muraille fin " << longueur << endl;
}
Muraille::~Muraille() {
	cout << "-- Muraille debut" << endl;
	delete tour1;
	delete tour2;
	cout << "-- Muraille fin" << endl;
}

Chateau::Chateau(int l, int h1, int h2, int hd, int s) : Batiment(s) {
	cout << "++ Chateau debut" << endl;
	muraille = new Muraille(l, h1, h2, this);
	donjon = new Donjon(this, hd);
	cout << "++ Chateau fin" << endl;
}
Chateau::~Chateau() {
	cout << "-- Chateau debut" << endl;
	delete muraille;
	delete donjon;
	cout << "-- Chateau fin" << endl;
}
```

```
main() {
      Chateau chateau(400, 40, 60, 80, 10000);
}
```

1) Dessiner le de généralisation UML.

2) Dessiner un diagramme de classes UML. Placer les ordres de multiplicité.

3) Dessiner un diagramme d'objets UML correspondant à l'état du programme après la déclaration du château dans le programme principal main() et avant la fin de celui-ci.

Exercice « classe-association-instance tamaguchi»

On suppose que l'on a 3 classes : Tamaguchi, Faim et Table. Ces classes sont reliées par la sémantique suivante : *Pour satisfaire sa Faim, un Tamaguchi doit aller à Table.*

1) Dessiner un diagramme de classe avec une unique association ternaire.

2) Redessiner ce diagramme avec 3 associations binaires. On trouvera des noms pour chacune des associations binaires.

3) Pour chaque association binaire, donner un nom à chaque rôle de l'association.

L'utilisateur dispose d'une société de Tamaguchi dont il doit s'occuper. Il dispose d'un certain nombre de tables pour leur donner à manger. Il peut en mettre 4 sur une même table et chaque Tamaguchi vivant possède une instance de la classe Faim. Un Tamaguchi mort n'est plus relié à aucune instance de la classe Faim.

4) Placer les ordres de multiplicités sur le diagramme de classes de la question 4).

Les classes possèdent les attributs suivants :

Tamaguchi : nom, etat
Faim : satisfaitp
Table : nombreTamaguchis

état peut prendre les valeurs "autonome", "non satisfait", "satisfaction en cours", "satisfaction terminée" et "mort".
satisfaitp peut prendre les valeurs true ou false, true quand le Tamaguchi a fini de manger ou est autonome, false quand il a faim ou qu'il est en train de manger.
nombreTamaguchis peut prendre des valeurs entières, il indique le nombre de Tamaguchi attablés à la table.

5) Dessiner un diagramme d'instances correspondant à la situation suivante :

Une table. Toto et Tutu sont attablés à la table. Toto mange. Tutu pleure.
Titi est autonome. Tata est mort. Tété pleure.

On indiquera précisément la valeur des attributs des instances et les liens entre les instances.