

Génie Logiciel TD n° 6, 7

UML, placement des informations

Objectif du TD

Savoir placer correctement les informations décrivant un domaine à modéliser. Correctement signifie trouver les bonnes classes, les bonnes associations, les bons attributs, de classe ou d'instance, puis les bonnes méthodes.

Exercice 1 (DEVELOPPEUR)

Soit le texte suivant:

Une *équipe* d'informatique {est composée de} développeurs. Une équipe {travaille pour} une *entreprise*. Une équipe possède un logo. Un *développeur* {utilise} un *ordinateur* qui lui est personnel. Un développeur peut être un *programmeur* ou un *spécifieur*. Le logo des ordinateurs des programmeurs est identique au logo de son équipe. Le logo des ordinateurs des spécifieurs est toujours ☺. Les développeurs réfléchissent. Le spécifieur dialogue avec les utilisateurs.

1) On suppose que les classes du texte précédent correspondent aux mots *en italique*. On suppose qu'un logo n'est pas une classe mais un entier. Dessiner le diagramme de généralisation. On généralisera les classes du domaine avec une classe "Objet".

2) On suppose que les associations correspondent aux mots {entre accolades}. Dessiner un diagramme de classes. On utilisera l'agrégation si cela est possible. On placera les ordres de multiplicité.

3) Soit les classes suivantes : Développeur, Programmeur, Specifieur, Equipe, Ordinateur, Entreprise. Placer les attributs 'logo', 'mesDeveloppeurs', 'monOrdinateur', 'mesOrdinateurs', 'monEquipe', 'mesEquipes', 'mesOrdinateurs', 'monEntreprise' et 'monDeveloppeur' dans les classes adéquates. Idem pour les méthodes 'creerMesDeveloppeurs', 'dialoguerAvecLUtilisateur' et 'Reflechir'. Compléter la description avec les types des propriétés (int, Equipe, Developpeur, Ordinateur, Liste, void) et les soulignements là où cela est nécessaire.

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

germain et gertrude sont des programmeurs. gérard est un spécifieur. Ils font partie de l'équipe "gereflehi" qui représente l'entreprise "gertoupourvou". Le logo de l'équipe "gereflehi" est ☺. Tous les développeurs utilisent un ordinateur.

Exercice 2 (FOOTBALL)

Soit le texte suivant:

Une *équipe* de football {est composée de} joueurs. Une équipe {représente} un *pays*. Une équipe possède une couleur. Un *joueur* {porte} un *maillot*. Un joueur peut être un *joueur de champ* ou un *gardien*. Les maillots des joueurs de champ est de la couleur de son équipe. Le maillot du gardien est toujours d'une couleur noire. Les joueurs ont le droit d'utiliser leurs pieds. Le gardien a le droit d'utiliser ses mains.

1) On suppose que les classes du texte précédent correspondent aux mots *en italique*. On suppose qu'une couleur n'est pas une classe mais un entier. Dessiner le diagramme de généralisation. On généralisera les classes du domaine avec une classe "Objet".

2) On suppose que les associations correspondent aux mots {entre accolades}. Dessiner un diagramme de classes. On utilisera l'agrégation si cela est possible. On placera les ordres de multiplicité.

3) Soit les classes suivantes : Joueur, JoueurDeChamp, Gardien, Equipe, Maillot, Pays. Placer les attributs 'couleur', 'mesJoueurs', 'monMaillot', 'monEquipe', 'mesMaillots' et 'monJoueur' dans les classes adéquates. Idem pour les méthodes 'creerMesJoueurs', 'utiliserLesMains' et 'utiliserLesPieds'. Compléter la description avec les types des propriétés (int, Equipe, Joueur, Maillot, Liste, void) et les soulignements là où cela est nécessaire.

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

ronaldo et bebeto sont des joueurs de champ. taffarel est un gardien. Ils font partie de la "selecao" qui représente le brésil. La couleur de l'équipe du brésil est le jaune. Tous les joueurs portent un maillot.

Exercice 3 (RESTAURANT)

Soit le texte suivant:

Un *restaurant* {est composée de} *tables*. Il est midi. Des *clients* {sont à} table. Des *plats* et des *boissons* (de la *nourriture*) {sont posés sur} les tables. Un client {mange son} plat et {boit sa} boisson. Un client peut être un *adulte* ou un *enfant*. Les boissons peuvent être des *bouteilles de vin*, des *carafes d'eau* ou des *tasses de café*. Un plat peut être une *entrée*, un *plat du milieu* ou un *dessert*. Plusieurs clients peuvent boire la même boisson. Un client mange un seul plat mais peut boire plusieurs boissons. Les bouteilles de vin et les plats ont des prix variables, un café coûte 10 francs et une carafe d'eau est gratuite. Un enfant ne boit ni vin ni café.

1) On suppose que les classes du texte précédent correspondent aux mots ou groupe de mots *en italique*. On suppose qu'un prix est un entier. Dessiner le diagramme de généralisation. On généralisera les classes du domaine avec une classe "Objet".

2) On suppose que les associations correspondent aux mots {entre accolades}. Dessiner un diagramme de classes. On placera les ordres de multiplicité 1 ou *.

3) Soit les classes suivantes : Restaurant, Table, Client, Adulte, Enfant, Nourriture, Boisson, Plat, CarafeEau, Café. Placer les attributs 'prix', 'monClient', 'mesClients', 'maTable', 'mesTables', 'maBoisson', 'mesBoissons', 'monPlat', 'mesPlats', dans les classes adéquates. Idem pour les méthodes 'void seMettreATable(Table)', 'void poserSurLaTable(Table)', 'void poserNourritureSur (Nourriture)', 'void debarrasserLesTables()', 'void boireDuVin(Vin)', 'void boireDeLEau(Eau)', 'void afficherLesClients()'. Compléter la description avec les types des propriétés (int, Liste, classes... , void) et les soulignements là où cela est nécessaire.

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

le restaurant "bacchus" comprend trois tables. laurence et paul ont une fille léa. ils sont à la table 1 avec valérie. laurence et valérie boivent une bouteille de bourgogne. laurence mange un riz cantonnais. valérie mange une salade de tomates. paul boit une tasse de café. léa mange son dessert et boit de l'eau. a la table 2, benoit et anne-laure boivent du vin mais n'ont pas faim. leur fille zoé mangeait une glace à la vanille qui est tombée par terre. la table 3 est vide.

Indiquer les instanciations, généralisations, spécialisations, agrégations, associations ou attributions présentes dans les bouts de phrases suivants :

- a) roger est un restaurateur.
- b) il tient le restaurant "bacchus" qui est un restaurant.
- c) un restaurateur est une personne,
- d) donc roger est une personne.
- e) un repas complet est composé d'une entrée, d'un milieu et d'un dessert.
- f) un plat est une entrée ou un milieu ou un dessert.
- g) la table 3 est une table du restaurant ;
- h) elle est vide.
- i) un vin est une boisson ;
- j) une boisson est soit du vin, soit de l'eau, soit du café.
- k) l'eau est froide ; le café est chaud.
- l) le café va bien avec le dessert ;
- m) d'ailleurs le dessert contient du sucre.

Exercice 4 (DEMON LOGE)

1) Pour chaque alinéa, dessiner le(s) diagramme(s) UML éventuel(s) correspondant.

A • Les démons sont de deux sortes : les fermions et les bosons.

(un diagramme de généralisation)

B1 • Un être vivant possède une ou plusieurs loges dans lesquelles viennent se placer des démons. Un démon est ubiquiste, cela signifie qu'il peut être présent dans zéro ou plusieurs loges.

(un diagramme de classes)

B2 • Les bosons sont toujours à plusieurs dans une loge. Dans ce cas, la loge est dite bosonique. Un fermion est toujours seul dans une loge. Dans ce cas, la loge est dite fermionique. Une loge peut aussi être vide.

(un diagramme de généralisation, deux diagrammes de classes)

C • Le boson sOrath est nécessaire à la vie d'un être vivant et il se trouve au moins dans une des loges d'un être vivant. Sans sa présence dans une de ses loges, un être ne reste vivant que très peu de temps. Avec le boson sOrath présent dans une de ses loges, un être vivant peut survivre longtemps.

(un diagramme d'instances)

D • Initialement, les êtres vivants ne possédaient qu'une seule loge. Puis, les êtres humains sont apparus avec une deuxième loge. D'abord, chez les Albaspinas, première sorte d'être humain, cette deuxième loge fut vide, la première étant remplie de bosons. Ensuite, chez les Luisants, la deuxième loge fut remplie soit par des bosons, soit par un fermion.

(un diagramme de généralisation, un diagramme d'instances avec une Albaspina et un Luisant)

E • Maintenant, en 1999, 95% des être humains sont banaux, ils possèdent deux loges bosoniques (remplies de bosons). 5% sont hors norme, ils possèdent une loge avec des bosons et une loge fermionique (avec un fermion). 0,000001% sont rarissimes, ils possèdent deux loges avec un fermion.

(un diagramme de généralisation, des diagrammes de classes)

F • Mistral est le seul être humain connu actuellement à avoir eu deux loges occupées par le fermion oRca, et il n'a pas survécu longtemps.

(un diagramme d'instances)

G • Un démon possède une puissance, représentée par un nombre. Pour un boson, ce nombre est entier, il s'appelle le charme et il est compris entre 0 et 9. Pour un fermion, ce nombre est demi-entier, il s'appelle la résistance et il est compris entre $\frac{1}{2}$ et $9\frac{1}{2}$.

(des diagrammes classe-attribut)

H • Il existe plusieurs sortes de bosons : les hypnotiques, les processionnaires et les caracoles. Les hypnotiques ont un charme variable, les caracoles ont un charme 1, les processionnaires ont un charme 2 et sOrath un charme 9. Chez les fermions, oRca possède une résistance $9\frac{1}{2}$ et l'eTranger une résistance $5\frac{1}{2}$.

(un diagramme de généralisation, des diagrammes classe-attribut, des diagrammes d'instances)

2) Synthétiser les diagrammes de généralisation précédents en un seul.

3) Synthétiser au mieux les diagrammes de classes précédents.

4) Décrire les classes suivantes : Démon, Fermion, Boson, Caracole, Hypnotique, Processionnaire, EtreVivant, EtreHumain, Loge, avec les attributs 'puissance', 'charme', 'resistance', 'maPremiereLoge', 'maSecondeLoge', 'monEtreVivant', 'monFermion', 'mesBosons', 'mesDemons', 'mesLoges'. Idem avec les méthodes 'void occuperUneLoge(Loge)', 'void ecrireCharme(Entier)', 'DemiEntier lireResistance()', 'void associerUneLoge(Loge)', 'void afficherLesBosons()', 'void afficherMonFermion()'. Compléter la description avec les types des propriétés (Liste, Entier, DemiEntier, classes... , void) et les soulignements là où cela est nécessaire.

Exercice 5 (IMMEUBLE APPARTEMENT)

1) Pour chaque alinéa, dessiner le(s) diagramme(s) UML éventuel(s) correspondant.

A • Un appartement possède plusieurs pièces. Il a une surface et un prix.

B • Une pièce est soit une salle d'eau, soit une salle de séjour. Certaines salles de séjour sont des chambres.

C • Un appartement possède au moins une porte, une fenêtre, une salle d'eau et une salle de séjour.

D • Un immeuble contient plusieurs appartements et une cage d'escalier. Celle-ci contient un escalier et une porte d'entrée qui donne sur la rue et plusieurs portes donnant chacune sur un appartement. L'immeuble est voisin d'une ou plusieurs rues et éventuellement d'une cour intérieure.

E • Une pièce possède au moins une porte et éventuellement des fenêtres. Elle a une surface et des pièces voisines.

F • Une fenêtre donne soit sur une rue, soit sur une cour intérieure. Une porte relie deux pièces voisines ou bien une pièce à une cage d'escalier ou bien la cage d'escalier à une rue.

2) Faire un nouveau diagramme UML B' en reprenant l'alinéa B ci-dessus et en ajoutant les informations de l'alinéa B' ci-dessous.

B' • Une entrée est une sorte de pièce, qui est une sorte d'espace intérieur, tout comme la cage d'escalier. Une rue et une cour intérieure sont des sortes d'espaces extérieurs. Finalement un espace est soit un espace intérieur soit un espace extérieur.

3) En utilisant la nouvelle généralisation de l'alinéa B' ci-dessus faire un nouveau diagramme UML EF', très synthétique, exprimant les éventuelles relations qui existent entre les portes, les fenêtres et les différents espaces.

4) Critiquer le diagramme C. On jugera sa cohérence avec les diagrammes A, B' et EF'. Proposer une nouvelle solution.

5) Faire un diagramme d'instances avec l'alinéa G ci-dessous. On vérifiera sa cohérence avec les diagrammes précédents.

G • L'appartement A contient une entrée avec une porte donnant sur la cage d'escalier. Cette entrée possède une fenêtre (donnant sur une cour intérieure) et deux autres portes. La première porte donne sur la chambre qui a deux fenêtres donnant sur le boulevard B. La deuxième porte donne sur la salle de séjour qui a deux fenêtres donnant sur l'avenue C. La salle de séjour communique avec la cuisine par une porte. La fenêtre de la cuisine donne sur la cour intérieure. La chambre communique avec la salle de bains par une porte. La salle de bains n'a pas de fenêtre.

6) En respectant le formalisme UML, placer les attributs suivants dans les bonnes classes:

'prix', 'surface', 'mesImmeubles', 'mesAppartements', 'mesPièces', 'mesFenêtres', 'mesPortes', 'mesRues'.

Idem avec les méthodes suivantes :

'int calculerPrix()', 'void afficherMesPiecesVoisines()', 'void afficherSurface()', 'void afficherMesPieces()'.

7) Compléter la description avec les types des propriétés (Liste, int, classes... , void) et les soulignements là où cela est nécessaire.

Exercice 6 (METRO)

Pendant la phase de spécifications des besoins, l'utilisateur du futur logiciel *Métro* a écrit le texte suivant :

"Le logiciel *Métro* est destiné aux utilisateurs du métro. L'utilisateur indique la *gare* de départ et la *gare* d'arrivée et le logiciel *Métro* conseille un *trajet* reliant ces deux *gares*. Les *gares* ont un nom et {sont situées sur} des *lignes*. Une *ligne* a une couleur. Une *gare* est soit un *terminus*, soit une *correspondance*, soit une *gare normale*. Une *ligne* {possède} deux *terminus*, des *gares normales* et au moins une *correspondance*. Une *correspondance* {relie} au moins deux *lignes*. Un *trajet* est {composé} d'une *gare* de départ, d'une liste de *segments* et d'une *gare* d'arrivée. Un *segment* {correspond à} une *ligne* et {possède} une *gare* origine et une *gare* destination."

1) Dessiner des diagrammes de classes et un diagramme de généralisation UML correspondant à ce texte. Les mots en italique du texte ci-dessus correspondent à des classes. Les mots entre accolades correspondent à des associations. On placera les multiplicités de manière identique à celle vue en cours.

2) Pour chaque classe, donner les attributs intrinsèques et les attributs associatifs de la classe, de manière cohérente avec les diagrammes de classes de la question 1).

3) Dans la classe *Gare*, écrire la signature des deux méthodes répondant à la phrase suivante :

"Pour une *ligne* et un *terminus* donnés, une *gare* a éventuellement une *gare* suivante et une *gare* précédente".

4) Effectuer la spécification d'interfaces correspondant au cas d'utilisation dans lequel l'utilisateur demande le trajet reliant la gare de départ *coccinelle*, située sur la ligne *rouge*, à la gare d'arrivée *océan*, située sur la ligne *bleue*. On supposera que le trajet trouvé par le logiciel *Métro* contient deux correspondances, *orange* et *herbe*, situées sur la ligne *jaune*. L'interface du logiciel est en mode clavier-écran. Pour les distinguer des sorties écran, on fera précéder les entrées tapées au clavier par un prompt *Métro*> .

5) Dessiner un diagramme d'instances UML cohérent avec le trajet de la question 4) et avec les diagrammes de classes de la question 1). Dans un souci de clarté et de simplicité, on ne dessinera que les instances pertinentes. En particulier, on ne dessinera pas toutes les gares du réseau.