INF5151 – Génie logiciel : analyse et modélisation

Modèle du domaine

Jacques Berger

Objectifs

Introduire le modèle du domaine et le diagramme de classes

Prérequis

Aucun

Modèle du domaine

Représentation visuelle des concepts touchés par le logiciel

Souvent nommé le modèle conceptuel

Modèle du domaine

Le modèle du domaine est optionnel dans le processus de développement d'un logiciel

Outil de communication dans une équipe de développement

Permet de synthétiser la compréhension du domaine

Domaine

Champ d'activité du logiciel

Le domaine possède un vocabulaire qui lui est propre

Le vocabulaire peut être spécifié ou clarifié dans un glossaire annexé au modèle du domaine

Modèle du domaine

On représente visuellement le modèle à l'aide d'un diagramme de classes UML

La modélisation du domaine est la décomposition du domaine en un ensemble de classes conceptuelles

Modèle du domaine

Une classe conceptuelle est un objet du monde réel dans un domaine spécifique

Diagramme de classes

Un diagramme de classes peut modéliser plusieurs choses

- Un modèle du domaine
- Une conception détaillée
- Une implémentation d'un logiciel

Diagramme de classes

Dans un modèle du domaine, on spécifie les classes, mais on ne spécifie pas les opérations sur les classes

On peut y mettre :

- Les classes conceptuelles
- Les associations entre les classes
- Les attributs des classes

Diagramme de classes

Les classes conceptuelles doivent être des objets du monde réel

Les classes ne sont pas nécessairement des objets logiciels (ex. classes en Java)

Les fichiers, bases de données, fenêtres, interfaces, etc. ne sont pas représentés dans ce modèle

Diagramme de classes

Ce n'est pas un modèle de données

Aucun élément relatif à une base de données (clé primaire, clé étrangère, index, etc.)

Il est possible d'avoir une classe sans attribut

Classe conceptuelle

Concept du domaine
- Une idée, un objet, une chose
- Tangible ou intangible

Décalage des représentations

Le modèle du domaine inspire souvent le modèle de données et la conception du logiciel

Si la conception est proche du modèle du domaine, ça diminue le décalage des représentations et facilite la compréhension du code

Création du modèle

Selon la portée de la modélisation :

- Identifier les classes conceptuelles
- Représenter les classes dans un diagramme
- Ajouter les attributs et associations

Création du modèle

Utilisation du vocabulaire du domaine
- Utiliser les termes existants du domaine
- Exclure les éléments non pertinents
- Ne pas ajouter des éléments inexistants

Associations

Une relation entre des classes suffisamment significative pour la souligner

Représentées par des lignes entre les classes avec un nom d'association (en PascalCase)

Associations

Les noms des associations doivent être clairs et précis

Éviter des noms génériques comme :

- A
- Possède
- Contient
- Utilise

Associations

Les associations peuvent avoir une multiplicité

Indique une relation numérique entre les classes

Multiplicité

* : 0, 1 ou plus
1..* : 1 ou plus
7 : exactement 7
3..5 : de 3 à 5
0..1 : 0 ou 1
1,5,9 : exactement 1, ou 5, ou 9

Attributs

Les attributs sont des données logiques dans une classe

Si une donnée est complexe, c'est-à-dire qu'elle n'est pas uniquement composée d'une valeur élémentaire (entier, réel, string, etc.), c'est probablement une autre classe

Attributs

On spécifie les attributs lorsque les exigences logicielles implique de mémoriser certaines données

Les attributs sont dans une section de la boîte d'une classe

Les attributs peuvent avoir un type, une visibilité, une multiplicité, une propriété et une valeur par défaut optionnel

Attributs

Notation

visibilité nom : type multiplicité = défaut {propriété}

Exemples :

age
+ age : entier
+ age : entier 1 = 45 {readOnly}
- age : [0..100]

Attributs

Lorsqu'un attribut est une valeur calculée à partir des autres données du modèle, on peut l'indiquer à l'aide d'un / devant le nom de l'attribut

/quantite

Esquisse

Lorsqu'on dessine des esquisses sur un tableau, on ne ferme pas les boîtes à droite et en bas

Ça permet de les agrandir au besoin

Exactitude

Il n'existe pas de modèle du domaine complet et exact

Il s'agit d'un outil de communication et de réflexion

Cas 1

Gestion de prêts dans une bibliothèque

Cas 1

Une installation par bibliothèque

Pas d'échange de livres entre bibliothèques

Le système doit offrir un catalogue de livres avec des recherches par titre, auteurs, sujets et code ISBN

Cas 1

Le catalogue peut être consulté sur un poste de la bibliothèque et sur le site web de la bibliothèque

Un membre de la bibliothèque peut consulter la disponibilité d'un livre et faire une réservation d'une durée de 3 jours

Cas 1

Un membre peut emprunter un livre pour une durée de 21 jours

Si un livre n'est pas retourné à l'intérieur du 21 jours, une pénalité de 0.50$ par jour par livre est facturé au membre

Cas 1

Un membre qui doit de l'argent à la bibliothèque ne peut pas emprunter de livre

Si un livre n'est pas retourné après un délai de 3 mois, le montant du livre est facturé au membre

Cas 1

Il faut conserver une trace de tous les prêts, réservations, émissions de facture et paiements du membre

Cas 2

Aspirateur robotisé

Cas 2

L'aspirateur doit mémoriser l'emplacement de sa base de chargement

Lorsque la pile de l'aspirateur ne contient que 10% ou moins de sa charge, l'aspirateur va se charger automatiquement sur sa base

Cas 2

L'aspirateur possède un contenant où placer les saletés aspirées, un capteur permet de vérifier si ce contenant est plein

Lorsque le contenant à saletés est plein, l'aspirateur retourne sur sa base et émet un signal lumineux

Cas 2

Des capteurs devant l'aspirateur lui permettent de détecter des obstacles devant lui

Lorsque l'aspirateur est devant un obstacle, il doit légèrement reculer, faire une rotation sur lui-même et partir dans une nouvelle direction

Cas 2

L'aspirateur doit mémoriser ses déplacements entre 2 charges afin de ne pas toujours aspirer les mêmes endroits

Après plusieurs nettoyages, si l'aspirateur a détecté une route optimale pour nettoyer la pièce où il se trouve, il la mémorise et l'applique automatiquement les fois suivantes

Cas 2

Une personne peut positionner 2 bornes physiques dans la pièce

La ligne droite entre les 2 bornes sera considérée comme un obstacle pour l'aspirateur

Plus loin...

UML 2 et les design patterns
Craig Larman
Pearson, 2005
Chapitre 9