UNIVERSITÉ PARIS-SUD 11

Méthodes et vues

Jan Van den Bussche, Limburg
Emmanuel Waller, Orsay

BDA 98

Présentation :
Clément de Groc, Orsay

UNIVERSITÉ PARIS-SUD 11

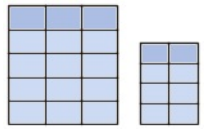
Outline

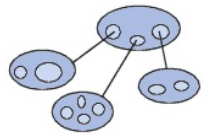
1. Introduction
2. Motivation
3. Solution proposée
4. Conclusion

Pré-requis : Base de données et vues

Base de données

- Ensembre structuré et organisé de données auxquelles on accède au moyen d'une **requête**
- Différents types (relationnelles, orientées objet . . .)





Vue

- Une synthèse d'une requête sur la base de données
- Intérets :
  - Sécurité
  - Confort

Pré-requis : Orientation objet

Schemas de méthodes (Kanellakis et al. 90)

- On appelle schema racine l'ensemble des classes et des prototypes de méthodes
- On appelle instance du schema racine les objets correspondants ainsi que la definition de ces méthodes (ou comportements)

Les vues orientées objet

- Contiennent classes + attributs

Travaux liés

O_2Views : thèse Souza / Souza, Abiteboul, Delobel EDBT94

- Axé sur l'aspect structurel
- Méthodes : appelées par des requêtes lors du calcul des attributs virtuels
- Sémantique : fonctionne parfaitement mais obscur
- Programmation **E. Waller** (application pour la revue Esprit)

A formal model of views : Guerrini et al, TAPOS97

- Axé sur l'aspect structurel
- Méthodes : possibilité d'utiliser dans la vue un code de la base racine
- Sémantique : non fournie

Exemple 1/3

- Base de données d'une compagnie d'assurance :

```
classe Client
attributs   :  ville, age, risque
methode     :  prime(age, risque) = age + risque
```

- La compagnie envisage pour les clients parisiens :
  - un risque spécifique (fixé à 20)
  - un contrat promotionnel (50%)

- Pour gérer cette situation, elle crée une **vue comportementale** qui contient :

```
classe Parisien
attributs   :  age, risque=20
methode     :  prime(age, risque) = age * risque
               promo_prime(prime) = prime / 2 (50%)
```

- Dans cette vue seront répertoriés automatiquement tout les clients ayant pour ville Paris.

Exemple 2/3

- On crée une instance de Client *c*

```
Client c : ville=Paris, age=42, risque=10
```

- Dans la vue est exécuté :

```
promo_prime(c)
> prime(c)  /   2
> ???       /   2
```

⇒ **Comment et dans quel contexte doit être exécutée prime(c)?**

Exemple 3/3

Calcul de *prime(c)* dans la vue

$\Rightarrow$ (age * risque) / 2

$\Leftrightarrow$ (42 * 20) / 2

$\Leftrightarrow$ 420

Calcul de *prime(c)* dans la base

$\Rightarrow$ (age + risque) / 2

$\Leftrightarrow$ (42 + 10) / 2

$\Leftrightarrow$ 26

Calcul de *prime(c)* dans la vue avec le code de la base

$\Rightarrow$ (age + risque) / 2

$\Leftrightarrow$ (42 + 20) / 2

$\Leftrightarrow$ 31

Motivation

Vue comportementale

- On souhaite ajouter des méthodes (comportements) spécifiques aux vues
- On défini donc une **vue comportementale** comme une instance du schema racine soit un ensemble d'attributs, d'implémentations de méthodes et de classes.

Problématique

⇒ Le programmeur doit pouvoir choisir le contexte et le mode d'exécution de ces méthodes

⇒ Vérifier alors qu'il n'y a jamais d'appel à des méthodes indéfinies

Formalisme

Importation de valeur : **import value**

- Spécifie d'utiliser le résultat d'une méthode depuis la base
- Exemple: nouvelle_prime@Parisien = 2 * **import value** prime

Importation de code : **import code**

- Spécifie d'utiliser le code d'une méthode de la base et de l'exécuter dans la vue
- Cependant, on souhaite pouvoir spécifier d'utiliser un attribut de la base ou de la vue ⇒ mot clé **with**.
- Exemple: nouvelle_prime@Parisien = **import code** prime
  with (age : **import value**) *[risque : vue]*

Décidabilité

Théorème

- La cohérence des vues comportementale $\equiv$ cohérence des schémas de méthodes
- $\Rightarrow$ Elle est donc décidable pour les cas de méthodes unaires et/ou non récursives
- cf. papier ou article Kanellakis, Ramaswamy, Waller et Abiteboul 90

Cohérence : aucune exécution n'aboutie à un appel de méthodes indéfini

Conclusion

Apport

- Un formalisme pour spécifier le contexte et le mode d'exécution d'une méthode
- Décidabilité (dans certains cas) de la correction

Extensions et perspectives

- Variantes de *import code* et *import value*
- Masquage
- Objets imaginaires (cf. article)