UNIVERSITEIT ANTWERPEN

SDM Transformatie voor Method Classes

Pieter Van Gorp

Universiteit Antwerpen

9th May 2005

UNIVERSITEIT ANTWERPEN

Motivatie

Afspraak op vergadering 7 maart:

- voorstel om # operator te ondersteunen door in PSM imperatief (i.p.v. declaratief in contracten) de nodige metadata te onderhouden,
- transformatie formaliseren in SDM
- link met MoTMoT illustreren

UNIVERSITEIT ANTWERPEN

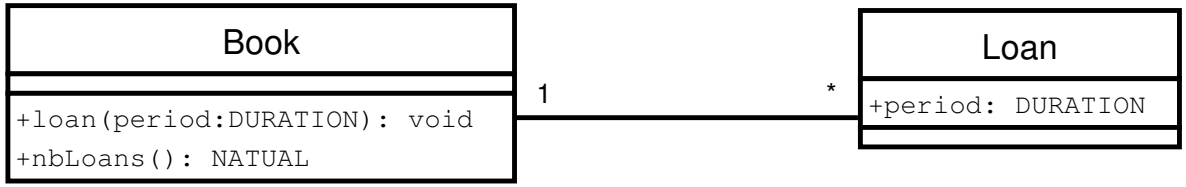
Overzicht

UNIVERSITEIT ANTWERPEN

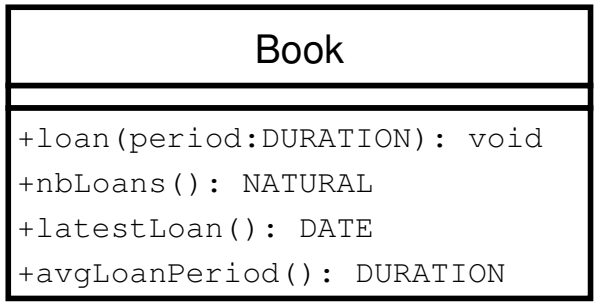
Case Study: Books

Voorbeeld

- Basisverantwoordelijkheid van Book: *loan()*



- Bovendien: tijdstip laatste uitlening, gemiddelde uitleentermijn, ...



- Observatie: complexe UML modellen
- Doel:
  - UML uitbreidingen voor conceptuele modellering
  - automatische vertaling

UNIVERSITEIT ANTWERPEN

#-Operator

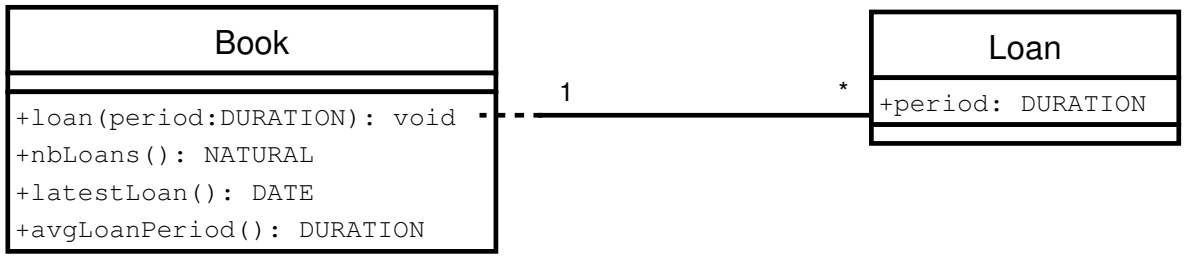
- Nieuwe Abstractie: #-operator
  - Betekenis: *obj#event* geeft de verzameling van alle voorkomens van de event *event()* op het object *obj* terug.
  - Voorbeeld:

```
context Book::avgLoanPeriod()
post:
result=avg((this#loan())->period)
```

- Gerealiseerd via *Method Class*
  - Cfr. Association Class

Method Classes

- Method Class:
  - Elke instantiatie van de method class is een uitvoering van de event “loan()” (en omgekeerd)
  - Elke karakteristiek van een event is ook een karakteristiek van objecten van de klasse
  - Elke parameter van de event is een property van de klasse
- Voorbeeld: loan(period)



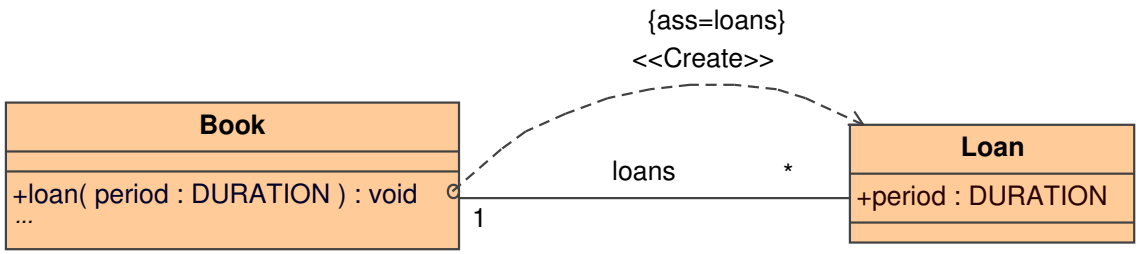
UNIVERSITEIT ANTWERPEN

Feedback

- PSM 2 (“niet reificeren”) werd onderbelicht

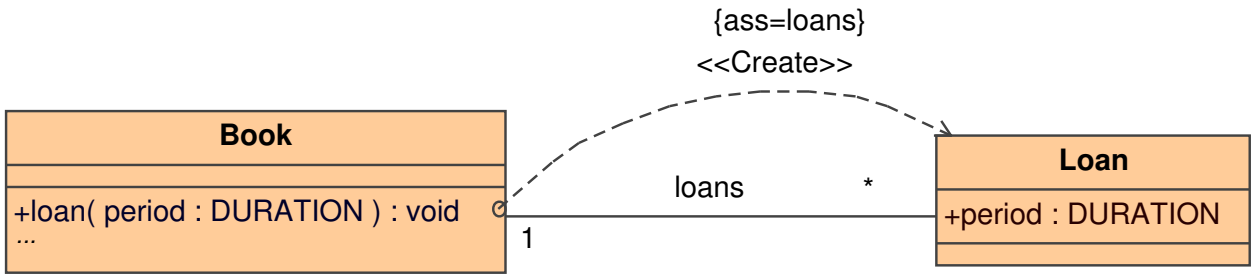
Book
-nbLoans: NATURAL -latestLoan: DATE -totalLoanPeriod: DURATION -loasIsExecuted: BOOLEAN
+loan(period:DURATION): void +nbLoans(): NATURAL +latestLoan(): DATE +avgLoanPeriod(): DURATION

- Voorstel: Modelleer meta-informatie bij Loan *klasse*
- Motivatie: #-operator wordt overbodig, dus simpelere theorie (standaard OO)



UNIVERSITEIT ANTWERPEN

Lifecycle modellering: Alternatief voor “#”?



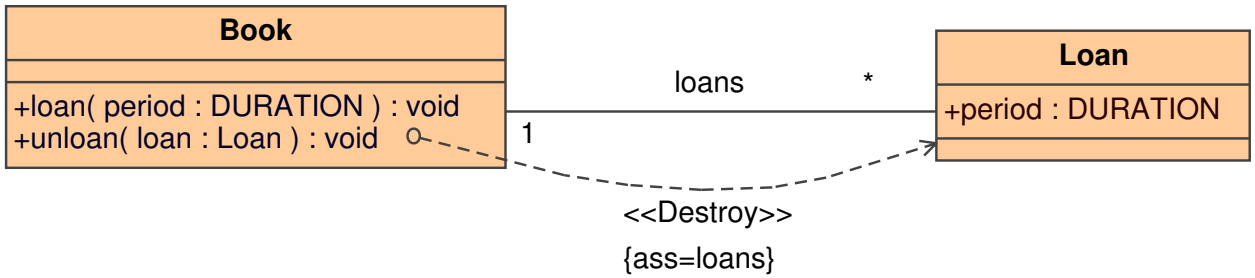
- *Book* en *Loan* zijn conventionele klassen
- “*loans*” is conventionele associatie
- “*loan*” methode wordt gedecoreerd met lifecycle informatie:
  - bij elke uitvoering wordt impliciet een nieuw Loan object aan “loans” toegevoegd
  - Terug naar het voorbeeld:

```
context Book::avgLoanPeriod()
post:
result=avg((this#loan())->period)
context Book::avgLoanPeriod()
post:  result=avg(this.loan.period)
```

- Creation time etc. voorzien

UNIVERSITEIT ANTWERPEN

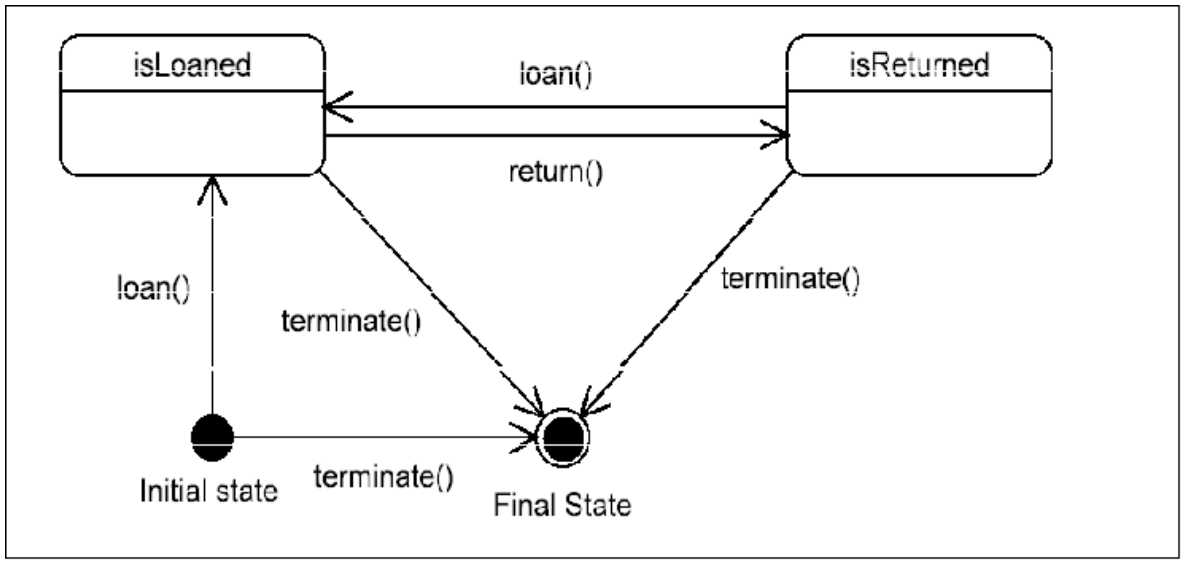
Analoge modellering: <<Destroy>>



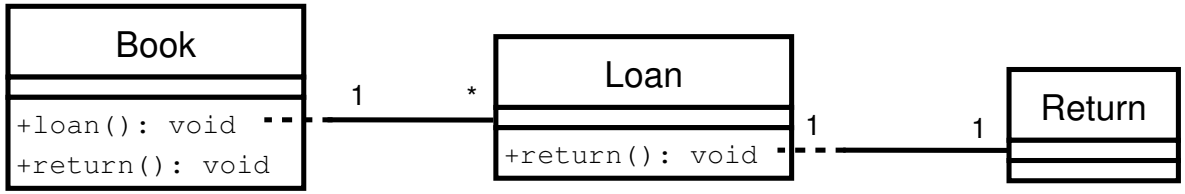
Toepassing op Case 2

Case 2: "Verbanden tussen methodes"

- Terugbrengen van uitgeleende boeken



- Voorgesteld gebruik van Method Classes (7 maart)

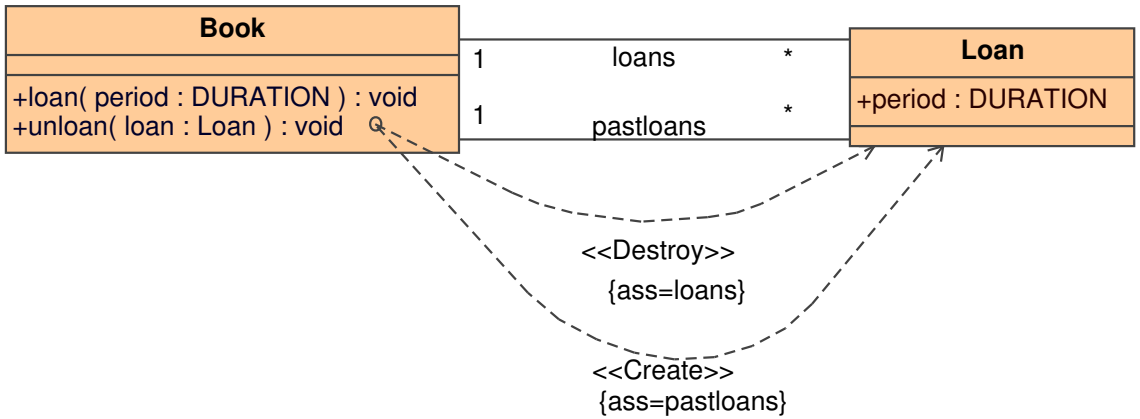


- Vraag: hoe modelleren met lifecycle decoraties?

UNIVERSITEIT ANTWERPEN

Case 2 via lifecycle decoraties

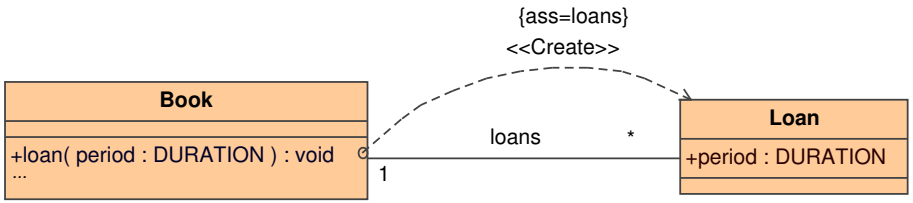
- Vraag: hoe modelleren met lifecycle decoraties?
- Antwoord:
  - voeg “*pastloans*” associatie toe, met 2 lifecycle decoraties
- Merk op:
  - “historiek” expliciet in domein model
  - juiste abstractieniveau (publiek)



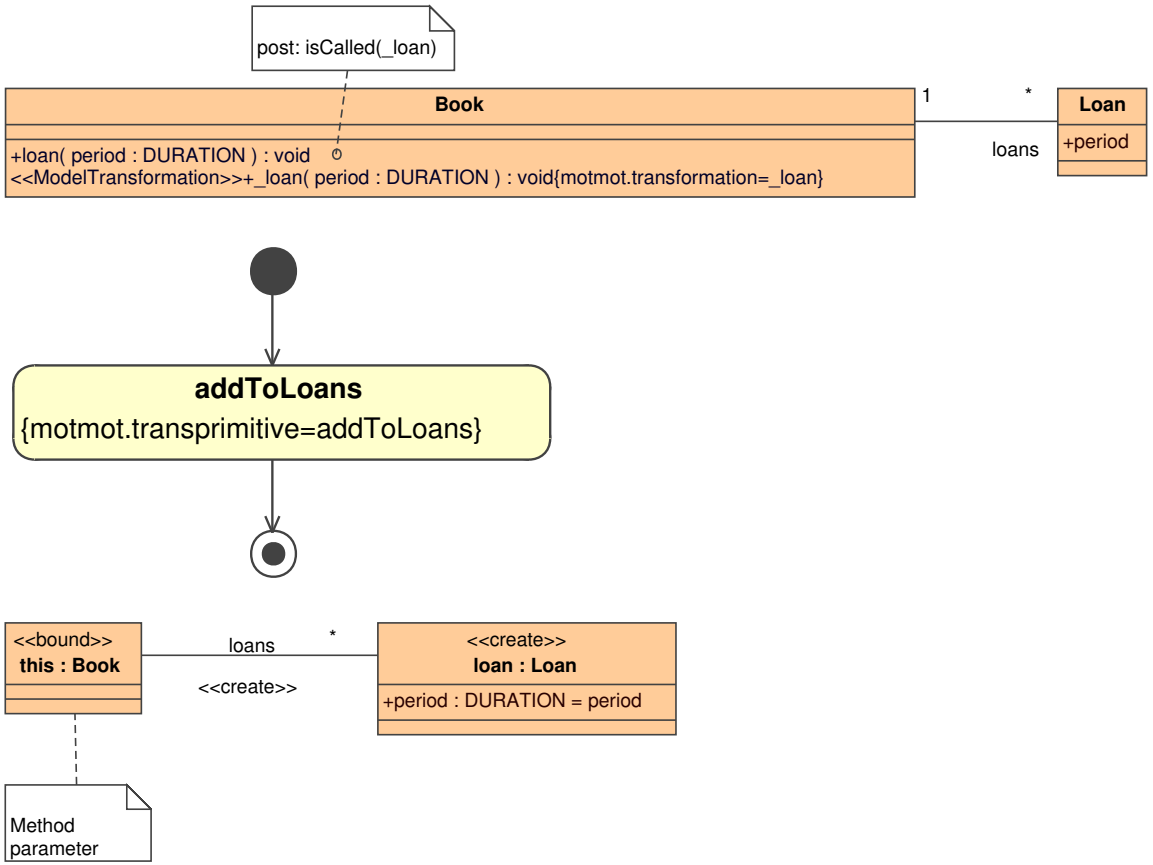
UNIVERSITEIT ANTWERPEN

Book PIM2PSM Voorbeeld

- PIM



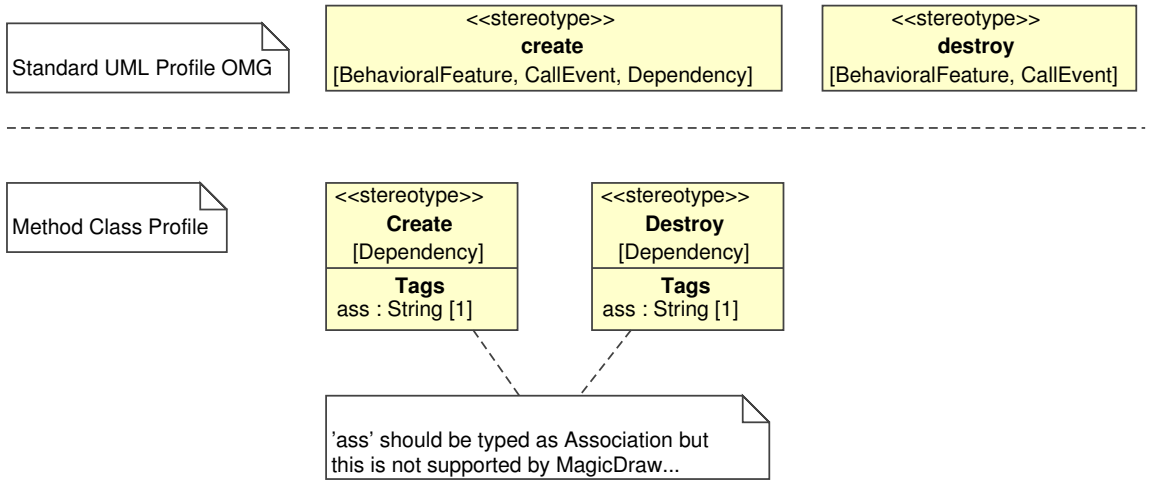
- PSM



UNIVERSITEIT ANTWERPEN

SDM Transfo voor lifecycle decoraties

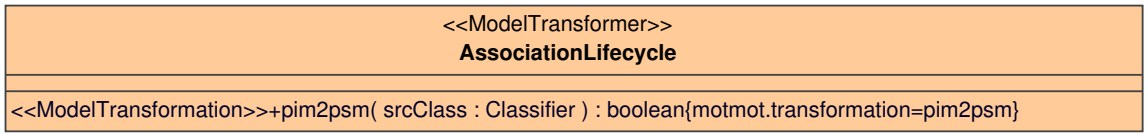
- Input metamodel: UML
- Gebruik van “lifecycle” profiel



- Output metamodel: UML
- Gebruik van “SDM” profiel (voor gedrag “add/delete”)
  - Details profiel: zie MoTMoT docs

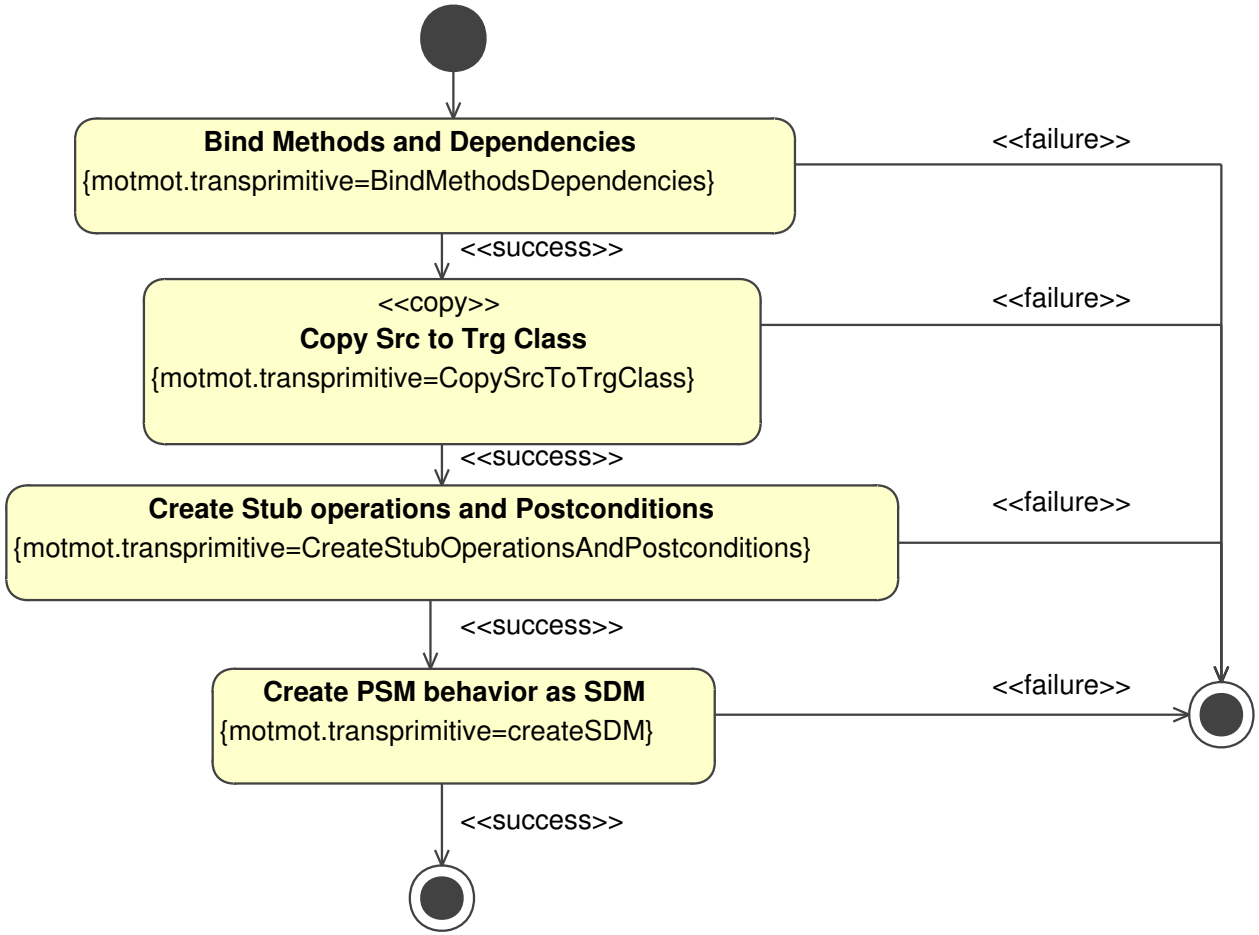
Transformatie klasse

- Transformatieklasse met één methode: *“pim2psm”*
  - één parameter: klasse die met lifecycle informatie gedecoreerde methodes bevat
  - elk van de <<Create>> methodes wordt vertaald zoals in ons PIM2PSM voorbeeld voor Book



UNIVERSITEIT ANTWERPEN

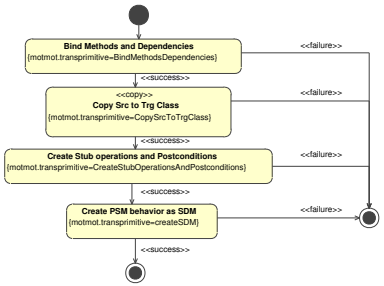
Transformatie Flow

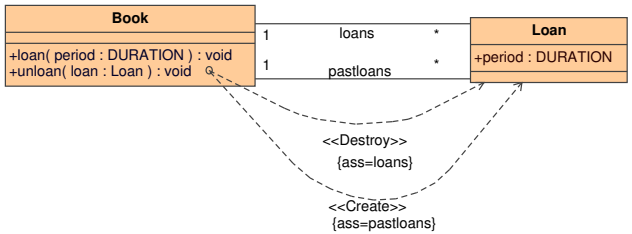


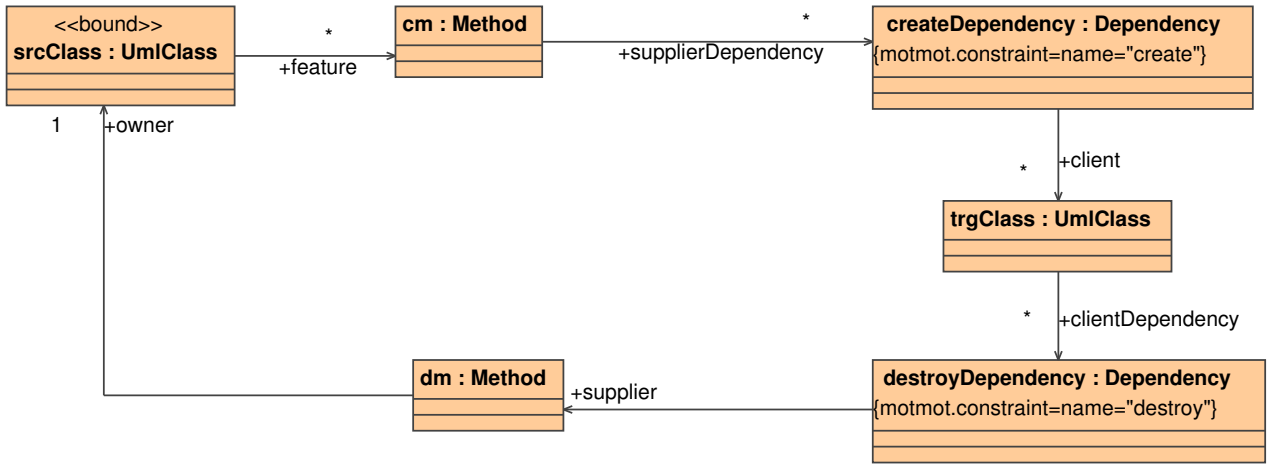
UNIVERSITEIT ANTWERPEN

Transformatie Primitieve: *Bind*

Story 1: match methodes en dependencies

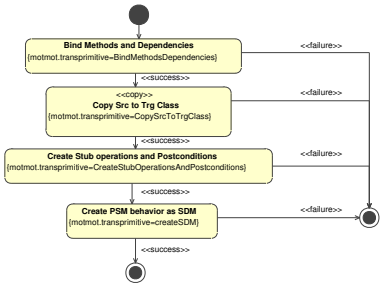


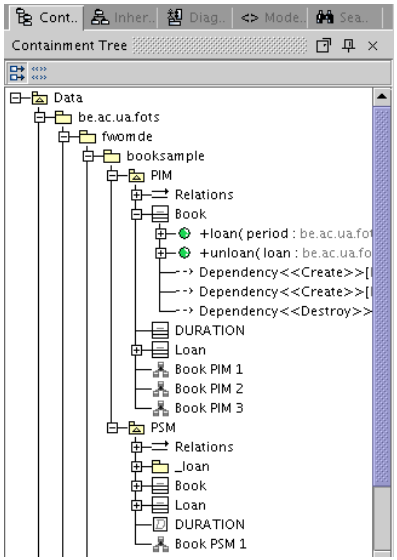


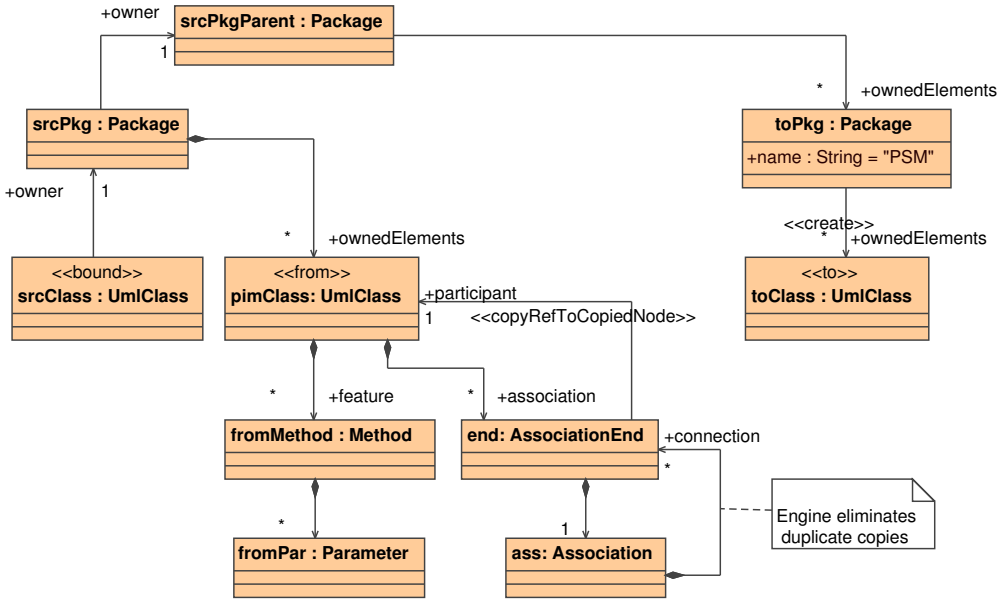


Transformatie Primitieve: *Copy*

Story 2: copy PIM class to PSM class

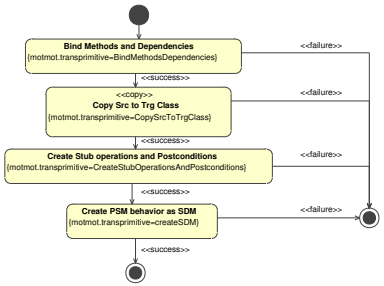


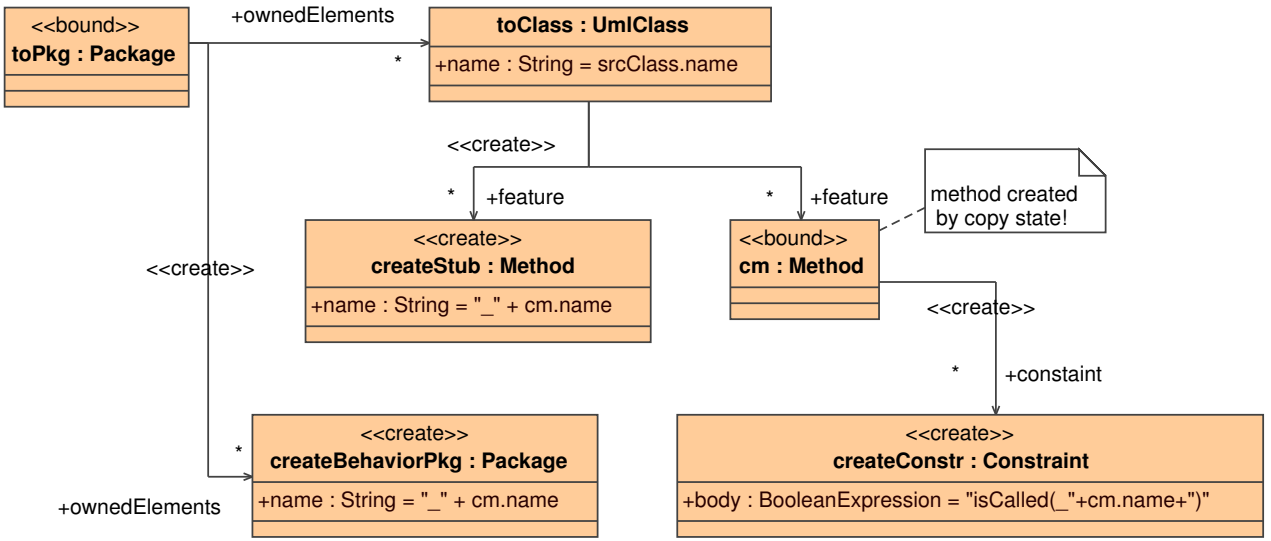




Transformatie Primitieve: *Create* (1)

Story 3: create stub operation (e.g., “_loan”) and postcondition (“isCalled(_loan)”)

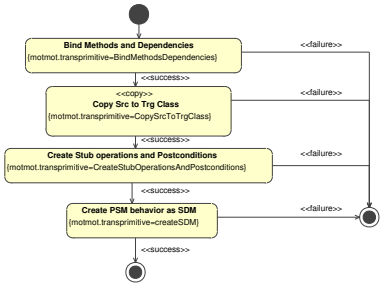


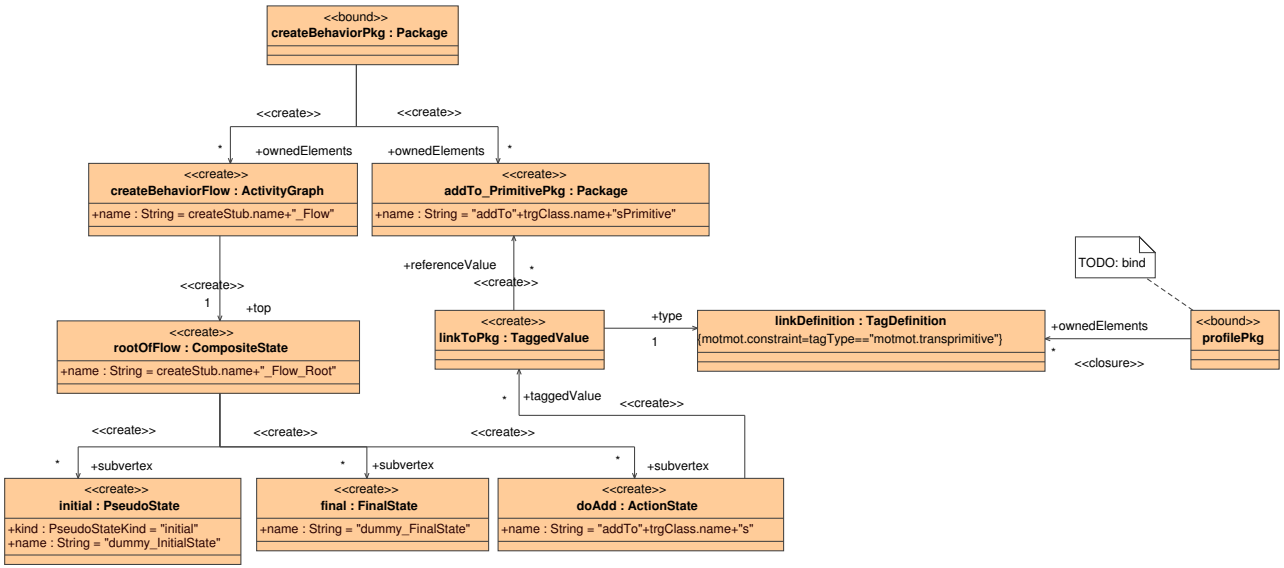


UNIVERSITEIT ANTWERPEN

Transformatie Primitieve: *Create* (2)

Story 4: Create behavior of stub method (e.g. "_loan") as SDM





UNIVERSITEIT ANTWERPEN

Conclusies

- “#”-operator en Method Classes
  - Expressiviteit geëvenaard met UML profiel voor *lifecycle modellering*
  - OCL tools e.d.m. hierdoor onaangepast
  - Simpele theorie
- Transformaties
  - Formalisering in SDM
  - Compacte specificatie
  - Introductie van GT in FWO-MDE project
- Basis voor licentiaatsthesis?

UNIVERSITEIT ANTWERPEN