Abstraheren van modellen

Geert Delanote

29 juni 2006

Geert.Delanote@cs.kuleuven.be

Overzicht

- # Operatie
- Archief
- @ Operatie
- Abstracte methode
- Afhankelijkheden tussen gebeurtenissen
- Partiële properties

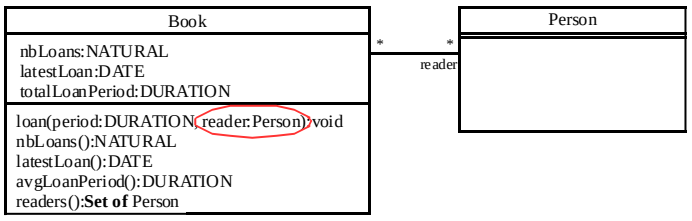
Ideeën => Niets is (volledig) uitgewerkt!

Operatie (1/8)

- Query 1 : De (verschillende) lezers van een boek.
- Query 2 : De (verschillende) boeken die iemand gelezen heeft.
- Model
  - Met properties
  - Met reïficatie
  - Met # operatie

Operatie (2/8)

- Model met properties



Context Book::loan(period:DURATION,
reader:Person) : void
post: self.nbLoans = self.nbLoans@pre + 1
post: self.latestLoan = now
post: self.totalLoanPeriod =
self.totalLoanPeriod@pre + period
post: self.reader =
self.reader@pre union {reader}

Context Book::nbLoans() : NATURAL
post: result = self.nbLoans

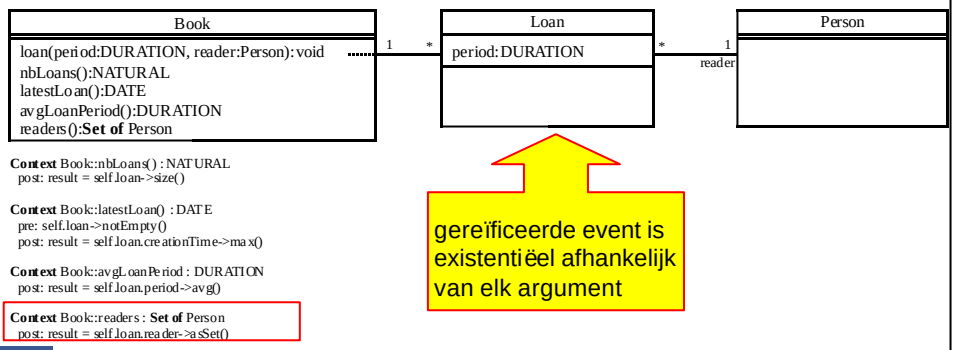
Context Book::readers():**Set of** Person
post: result = self.reader

Context Book::latestLoan() : DATE
pre: self.nbLoans > 0
post: result = self.latestLoan

Context Book::avgLoanPeriod() : DURATION
post: result =
if self.nbLoans = 0
then 0
else self.totalLoanPeriod/self.nbLoans

Operatie (3/8)

- Model met reïficatie



Context Book::nbLoans() : NATURAL
post: result = self.loan->size()

Context Book::latestLoan() : DATE
pre: self.loan->notEmpty()
post: result = self.loan.creationTime->max()

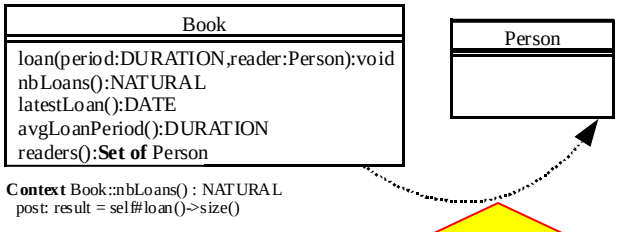
Context Book::avgLoanPeriod : DURATION
post: result = self.loan.period->avg()

Context Book::readers : **Set of** Person
post: result = self.loan.reader->asSet()

gereïficeerde event is existentiëel afhankelijk van elk argument

Operatie (4/8)

- Model met # operatie



Context Book::nbLoans() : NATURAL
post: result = self#loan()->size()

Context Book::latestLoan() : DATE
pre: self#loan()->notEmpty()
post: result = self#loan().executionTime->max()

Context Book::avgLoanPeriod() : DURATION
post: result = self#loan().period->avg()

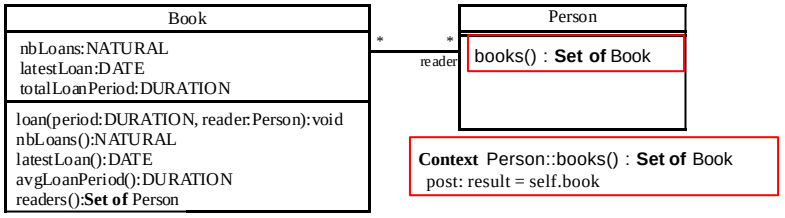
Context Book::readers() : **Set of** Person
post: result = self#loan().reader->asSet()

Dependency

Low coupling

Operatie (5/8)

- Model met properties



Context Person::books() : **Set of** Book
post: result = self.book

Context Book::loan(period:DURATION,
reader:Person) : void
post: self.nbLoans = self.nbLoans@pre + 1
post: self.latestLoan = now
post: self.totalLoanPeriod =
self.totalLoanPeriod@pre + period
post: self.reader =
self.reader@pre union {reader}

Context Book::nbLoans() : NATURAL
post: result = self.nbLoans

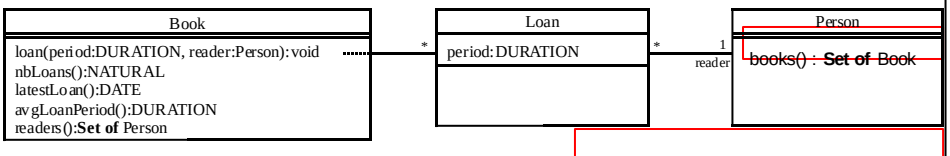
Context Book::readers():**Set of** Person
post: result = self.reader

Context Book::latestLoan() : DATE
pre: self.nbLoans > 0
post: result = self.latestLoan

Context Book::avgLoanPeriod() : DURATION
post: result =
if self.nbLoans = 0
then 0
else self.totalLoanPeriod/self.nbLoans

Operatie (6/8)

- Model met reïficatie



Context Person::books() : **Set of** Book
post: result = self.loan.book->asSet()

Context Book::nbLoans() : NATURAL
post: result = self.loan->size()

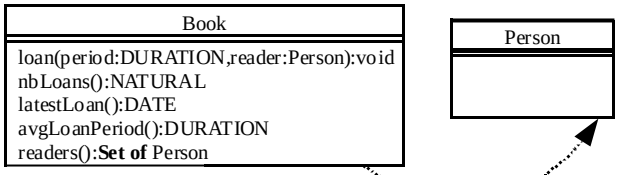
Context Book::latestLoan() : DATE
pre: self.loan->notEmpty()
post: result = self.loan.creationTime->max()

Context Book::avgLoanPeriod : DURATION
post: result = self.loan.period->avg()

Context Book::readers : **Set of** Person
post: result = self.loan.reader->asSet()

Operatie (7/8)

- Model met # operatie



Context Book::nbLoans() : NATURAL
post: result = self#loan()->size()

Context Book::latestLoan() : DATE
pre: self#loan()->notEmpty()
post: result = self#loan().executionTime->max()

Context Book::avgLoanPeriod() : DURATION
post: result = self#loan().period->avg()

Context Book::readers() : **Set of** Person
post: result = self#loan().reader->asSet()

Context Person::books() : **Set of** Book
post: result = {b:Book|b#loan()->select(reader=self)->notEmpty()}

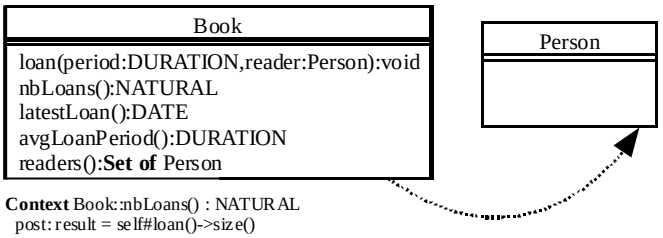
Context Book::books(person:Person) : **Set of** Book
post: result = {b:Book|b#loan()->select(reader=person)->notEmpty()}

Low coupling

Operatie (8/8)

- Research issues
  - Regels formuleren
    - Welke “artificiële” objecten wegwerken?
  - Transformatie van “statische” queries
  - Model met # operatie beter?
    - Low coupling

Archief



Context Book::nbLoans() : NATURAL
post: result = self#loan()->size()

Context Book::latestLoan() : DATE
pre: self#loan()->notEmpty()
post: result = self#loan().executionTime->max()

Context Book::avgLoanPeriod() : DURATION
post: result = self#loan().period->avg()

Context Book::readers() : **Set of** Person
post: result = self#loan().reader->asSet()

- Populatie ?
- Archief ?
- Populatie + archief ?

@ Operatie

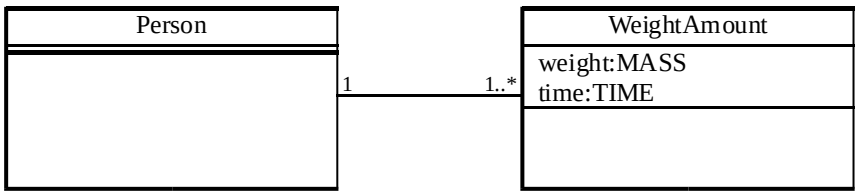
- Opvragen van een eigenschap van een object op een gegeven tijdstip
  - Voorbeeld
    - Gewicht (leeftijd) van een persoon op het tijdstip van uitlening
- Model
  - Met properties
  - Met reïficatie
  - Met # operatie

@ Operatie

- Model met properties
  - Collectie tupels <new weight,time> bijhouden
    - Postconditie van setWeight()

@ Operatie

- Model met reïficatie



@ Operatie

- Model met @ operatie

Person
weight:MASS
getWeightAt(t:TIME):MASS

Context Person::getWeightAt(t:TIME):MASS
post: result = self@t.getWeight()

Abstracte methode (1/6)

- Model met properties

Account
balance:MoneyAmount
withdraw(amount:MoneyAmount):void deposit(amount:MoneyAmount):void transferTo(amount:MoneyAmount, target:Account):void getBalance():MoneyAmount

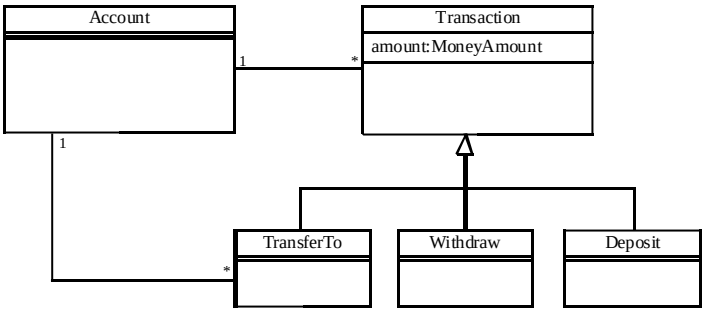
Context Account::withdraw(amount:MoneyAmount):void
post: self.balance = self.balance@pre - amount

Context Account::deposit(amount:MoneyAmount):void
post: self.balance = self.balance@pre + amount

Context Account::transferTo(amount:MoneyAmount,target:Account):void
pre: self<>target
post: self.balance = self.balance@pre - amount
post: target.balance = target.balance@pre + amount

Abstracte methode (2/6)

- Model met reïficatie



Abstracte methode (3/6)

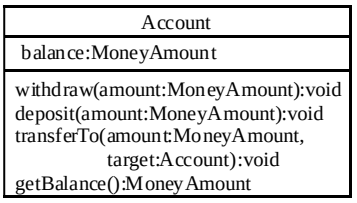
- Model met abstracte methode

Account
transaction(amount:MoneyAmount):void withdraw():void deposit():void transferTo(target:Account):void

Context Account::transferTo(target:Account):void
pre: self<>target

Abstracte methode (4/6)

- Model met properties : getBalance()



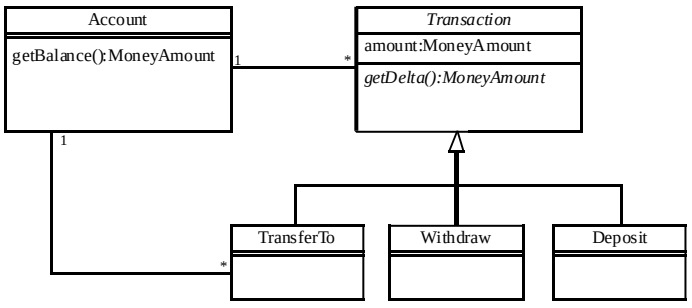
Context Account::withdraw(amount:MoneyAmount):void
post: self.balance = self.balance@pre - amount

Context Account::deposit(amount:MoneyAmount):void
post: self.balance = self.balance@pre + amount

Context Account::transferTo(amount:MoneyAmount,target:Account):void
pre: self<>target
post: self.balance = self.balance@pre - amount
post: target.balance = target.balance@pre + amount

Abstracte methode (5/6)

- Model met reïficatie : getBalance()



Context Account::getBalance():MoneyAmount
post: result = self.transaction.getDelta()->sum()+self.transferTo.getAmount()->sum()

Context Withdraw::getDelta():MoneyAmount
post: result = -self.getAmount()

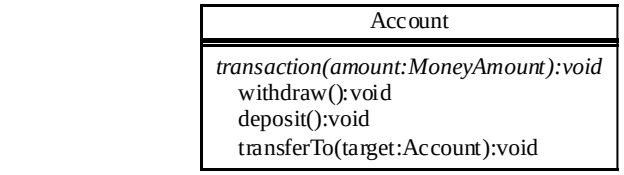
Context Deposit::getDelta():MoneyAmount
post: result = self.getAmount()

Context TransferTo::getDelta():MoneyAmount
post: result = -self.getAmount()

Specificatie onafhankelijk van de subklassen

Abstracte methode (6/6)

- Model met abstracte methode : getBalance()



Context Account::transferTo(target:Account):void
pre: self<>target

Context Account::getBalance():MoneyAmount
post: self#deposit().amount->sum() -
self#withdraw().amount->sum() -
self#transferTo().amount->sum() +
{tr:transaction|tr.target=self}.amount->sum()

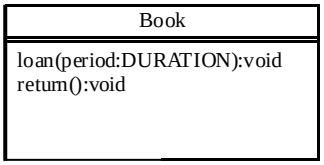
Specificatie afhankelijk van concrete methodes

Context Account::getBalance():MoneyAmount
post: self#transaction().delta->sum() +
{tr:transaction|tr.target=self}.amount->sum()

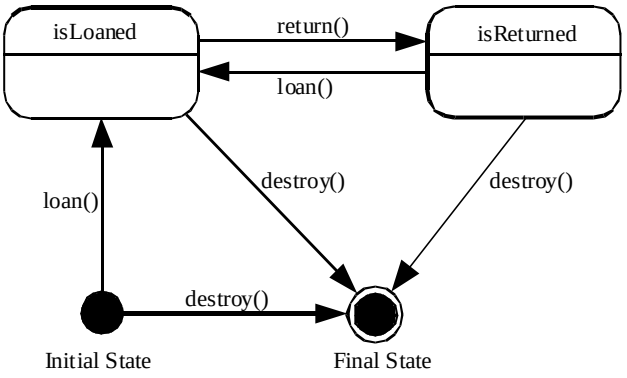
"delta" is een abstracte eigenschap van *transaction()*

Afhankelijkheden tussen gebeurtenissen

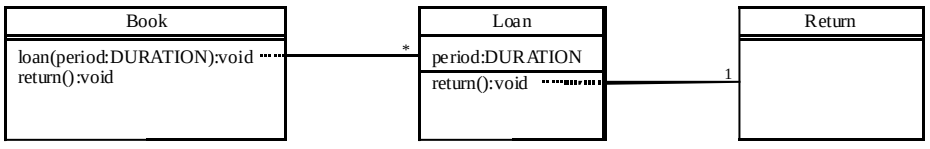
- Een boek kan pas teruggebracht worden *na* uitlening



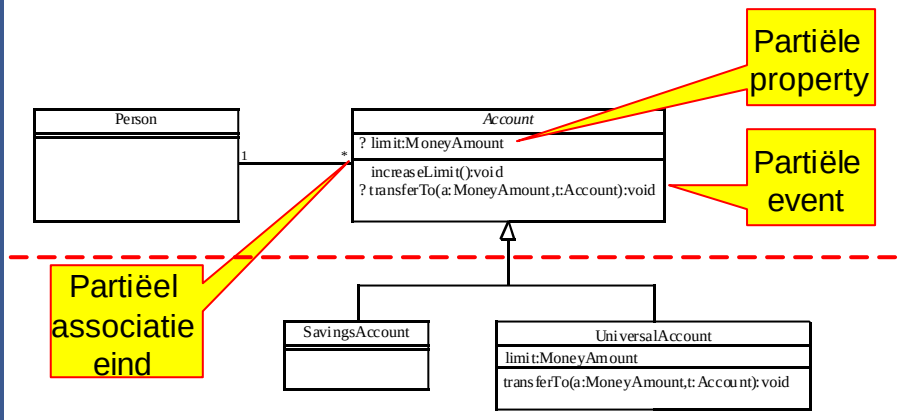
Afhankelijkheden tussen gebeurtenissen



Afhankelijkheden tussen gebeurtenissen



Partiële properties



Besluit

- Klassificatie
  - Abstracties
    - # Operatie
    - Abstracte methode
    - Afhankelijkheden tussen gebeurtenissen
    - @ Operatie
  - Lagen
    - Partiële properties
  - Notatie voor vaak voorkomend patroon
    - Archief