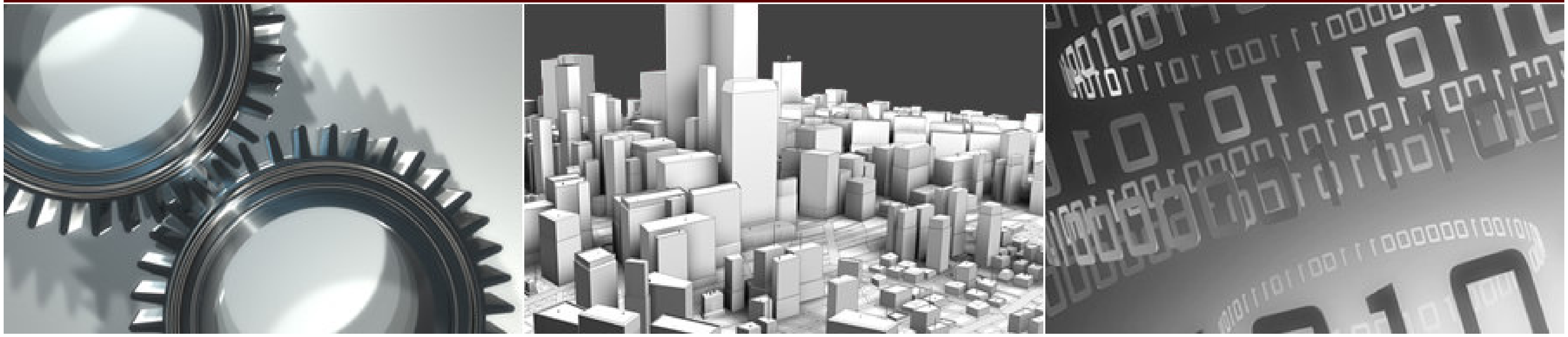


Very Short OCL Tutorial

Contact: info@nomos-software.com

Content

1. Background on OCL

2. Different Types of OCL Rules
   - Checking Attribute values
   - Implies Rules
   - Rules across classes
   - Rules on collections

3. Advantages of OCL

4. Learn More

nomos software

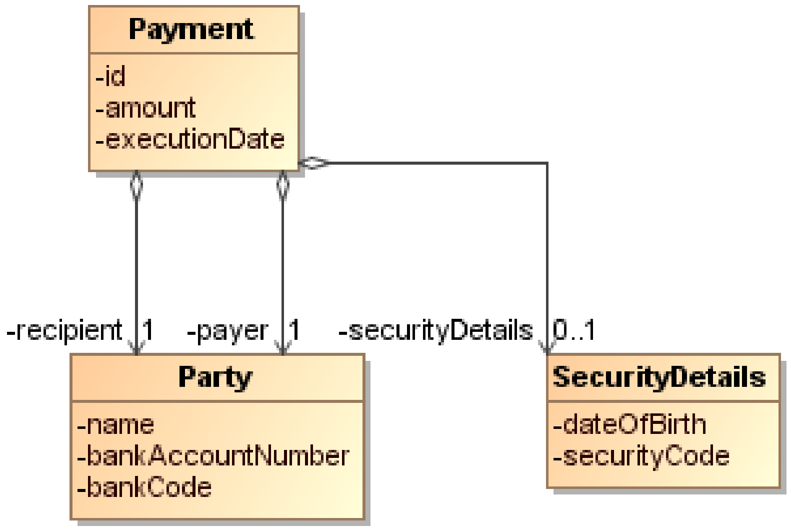
What is OCL?

- Object Constraint Language
    - Standardised, non-proprietary rules language, partner standard to UML
    - For adding constraints / business rules to models
- Supplements natural language rules
    - Express rules unambiguously in OCL
    - Then generate code
- Supported in modelling environments
    - E.g. MagicDraw, IBM Rational Software Architect, Eclipse

nomos software

Example

- A class model can define the structure of data
  - *"A payment must include a payer and a recipient"*
- But OCL is needed to define interdependencies between the data
  - *"The payer and the recipient cannot be the same"*
  - *payer.name <> recipient.name*



nomos software

OCL Resources and References

- Book
  - The Object Constraint Language: Getting your Models Ready for MDA (2nd Edition). Jos Warmer and Anneke Kleppe, Addison Wesley
- OCL Specification
  - http://www.omg.org/spec/OCL//
- LinkedIn Group
  - OCL users
- Jordi Cabot’s Modelling Languages Portal
  - http://modeling-languages.com/
- Dresden’s OCL Portal
  - http://st.inf.tu-dresden.de/oclportal/
- Eclipse OCL Project
  - http://www.eclipse.org/modeling/mdt/?project=ocl

nomos software

OCL Rule Basics

Need to define four things for a rule

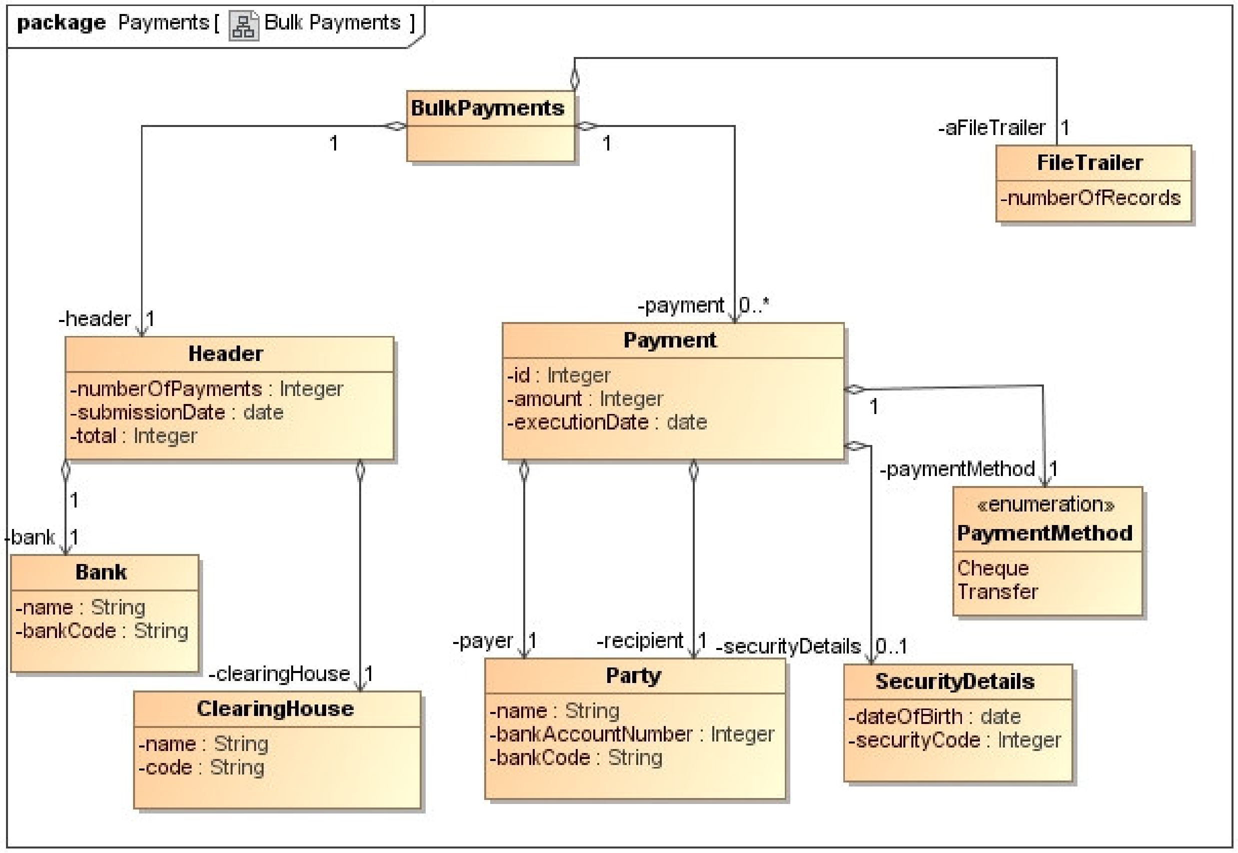
Context	The classifier with which the rule is associated
Name	The name of the rule
OCL	The rule expressed precisely in OCL
Error Message	A textual description of the business rule using the vocabulary of the business user.

Example Model

Class Diagram of Model used for Tutorial – Bulk Payments

A list of payments (money transfers between two bank accounts) that a bank has been requested to make on a particular day. The money transfer can be either an electronic funds transfer or a cheque. The list of payments is sent to a clearing house for processing.

Loosely based on real UML models for new generation European payments (ISO20022 and SEPA)



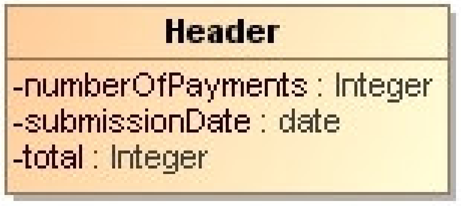
Rule Types

- Checking Attribute Values
- Implies / if …then rules
- Rules across classes
- Rules on collections

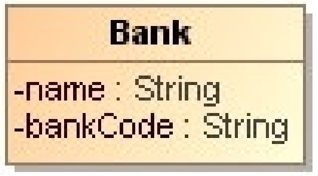
nomos software

Checking Attribute Values

Context	Header
Name	maxNumberOfPayments
OCL	numberOfPayments <= 100
Error Message	A maximum of 100 payments can be included in a bulk payments file.



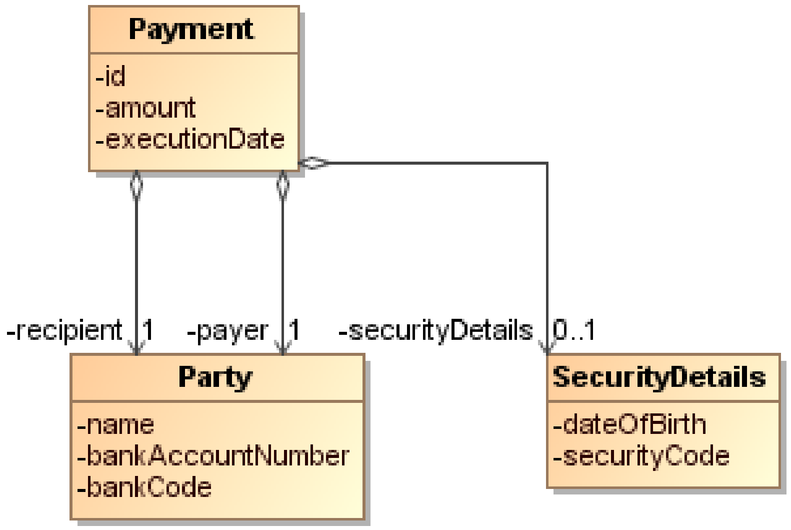
Context	Bank
Name	bankCodeSize
OCL	bankCode.size() = 8 or bankCode.size() = 11
Error Message	The bank code must be either 8 characters (primary code for the bank) or 11 characters (branch code) long.



Implies Rule

Used for rules of form ‘if … then …’. The most common type of rule.

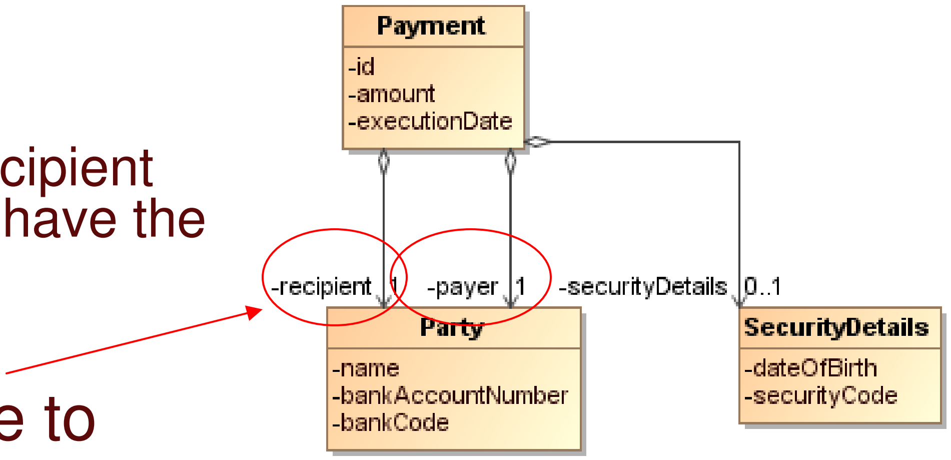
Context	Payment
Name	securityDetailsRequired
OCL	amount > 1000 implies securityDetails->notEmpty()
Error Message	If the amount of a Payment is greater than 1000, securityDetails must be provided.



nomos software

Rules Across Associations

- Many rules need to reference 'associated' class instances
- Example:
  - The payer Party and the recipient Party to a payment cannot have the same name
- Use associationEnd name to reference the class
  - payer.name <> recipient.name

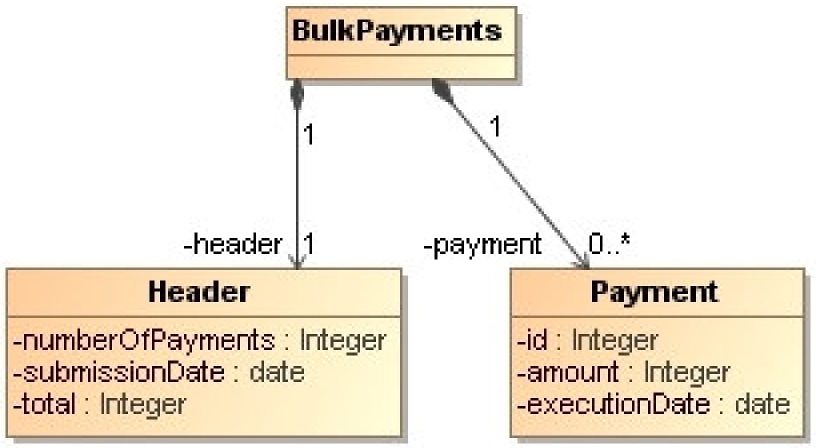


nomos software

Examples

Context	BulkPayments
Name	numOfPaymentsMatches
OCL	header.numberOfPayments = payment->size()
Error Message	The number of payments in the bulkPayments file must match the numberOfPayments in the header.

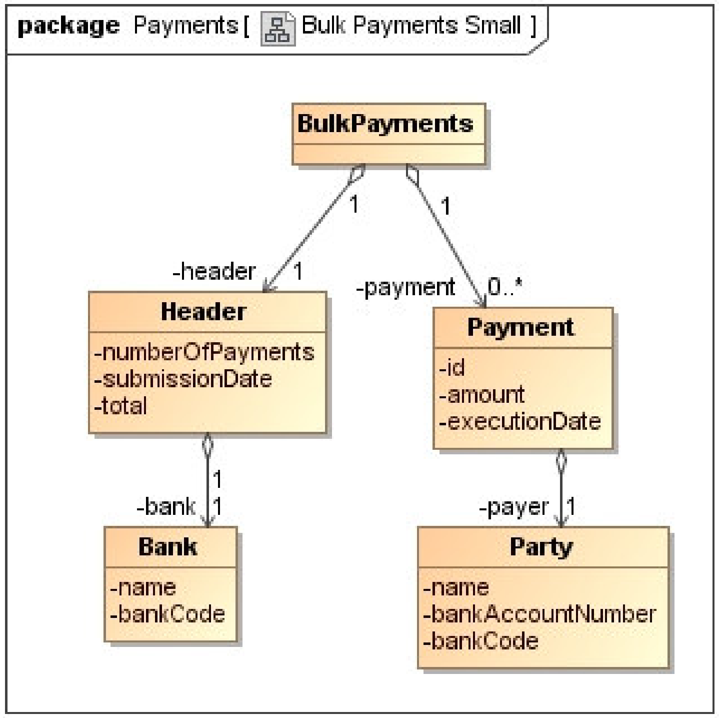
Context	BulkPayments
Name	totalInHeaderMatches
OCL	header.total = payment.amount->sum()
Error Message	The payment total in the bulkPayments header must be the same as the sum of the amounts in each payment.



nomos software

Collections

- You often need to write rules on collections of things
  - as opposed to on one thing
- Examples
  - The bankCode in the header and the bankCode for each payment payer must be the same
  - There must be at least one payment with value greater than 100 USD included in a bulk payments file
- OCL provides special operations for collections
  - ForAll, Exists, Select, Size etc
- Collections are very powerful

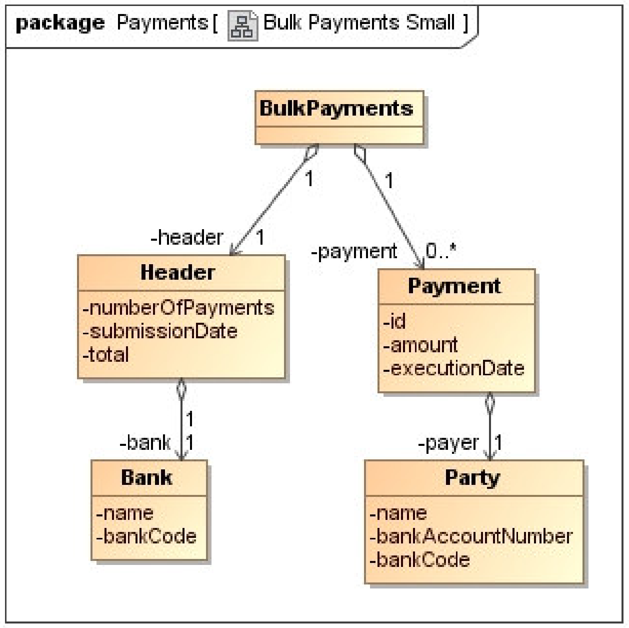


nomos software

Collection Examples

Context	BulkPayments
Name	headerBankCodeMatchesPayerBankCode
OCL	payment->forAll(p | p.payer.bankCode = header.bank.bankCode)
Error Message	The bankCode in the header and the bankCode for the payer in each payment must be the same.

Context	BulkPayments
Name	atLeastOnePaymentGreaterThan100
OCL	payment->exists(p | p.amount > 100)
Error Message	There must be at least one payment with value greater than 100 us dollars included in the bulk payments.



nomos software

More Examples

Context	Header
Name	lowOrHighValueclearingHouse
OCL	(total > = 10000 implies clearingHouse.code = ‘TARGET2’) and (total < 10000 implies clearHouse.code = ‘STEP2’)
Error Message	If the total payments amount is greater than 100,000 US dollars, the clearing house code must be set to TARGET2. If the total payments amount is less than 100,000, the clearing house code must be set to STEP2.

Context	BulkPayments
Name	executionDateSameAsSubmissionDate
OCL	payment->forAll(a | a.executionDate = header.submissionDate)
Error Message	The executionDate for each payment must be the same as the submissionDate in the BulkPayments header.

nomos software

More Examples

Context	BulkPayments
Name	samePaymentMethod
OCL	payment->forAll(a | a.paymentMethod = PaymentMethod::Cheque) or payment->forAll(a | a.paymentMethod = PaymentMethod::Transfer)
Error Message	All payments in a bulk payment file must use the same payment method. That is, they must all be cheque payments, or all be electronic transfers.

Context	BulkPayments
Name	paymentIdsUnique
OCL	payment.id->size() = payment.id->asSet()->size()
Error Message	The id for each payment must be different.

Advantages of OCL

- Is a standard
  - Lots of rules systems use proprietary languages
- Can use to write simple rules
  - But also to write very powerful rules
- Can write the rules in a modelling environment
  - Suitable for systems engineers and business analysts
- Can be used for autogeneration
  - Using Nomos products, can autogenerate executable code for XML data
  - for deployment in test and production environments

nomos software

Learn More

Learn more about OCL

Find out about OCL code generation solutions

Contact us at info@nomos-software.com