Object Constraint Language (OCL)

Tim Schooley

UCL ComputerScience 1

SLIDES ARE BORING

(apologies in advance for this)

UCL ComputerScience 2

Origins

- In 1996, OMG requested proposals on Object Analysis and Design.
- In 1997, IBM and ObjecTime jointly submitted a proposal, which included the OCL.
- After merging parts from both companies, the OCL was placed into what is now the UML 1.1 specification.
- OCL was developed by Jos Warmer as a language for business modelling within IBM.
- It is derived from the Syntropy method of Steve Cook and John Daniels (early 1990's OO design and analysis method).

UCL ComputerScience 3

What is it ?

- A language for specifying constraints on objects in the UML.
- Specifies conditions that must hold for the system being modelled.

UCL ComputerScience 4

Why ?

- UML isn't refined enough to describe all relevant aspects of a specification.
- Natural language was used, which led to ambiguities.
- Formal languages were developed, but very mathematical (so not easy to use etc)
- _ In comes OCL, to fill the gaps.....

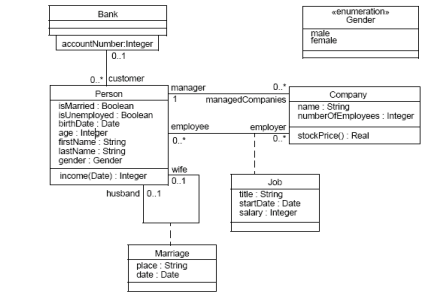
UCL ComputerScience 5

What is it ? (2)

- From the OCL specification:
  - "OCL is a pure **specification** language; therefore, an OCL expression is guaranteed to be without side effect. When an OCL expression is evaluated, it simply returns a value. It cannot change anything in the model. "
  - "This means that the state of the system will never change because of the evaluation of an OCL expression, even though an OCL expression can be used to *specify* a state change (e.g., in a post-condition). "
  - "OCL is *not* a programming language; therefore, it is not possible to write program logic or flow control in OCL."

UCL ComputerScience 6

Class Diagram



Where can OCL be used ?

- As a query language
- To specify invariants on classes and types in the class model
- To specify type invariant for Stereotypes
- To describe pre and post conditions on Operations and Methods
- To describe Guards
- To specify target (sets) for messages and actions
- To specify constraints on operations
- To specify derivation rules for attributes for any expression over a UML model.

As a query language

- OCL can be used to indicate the result of a query operation, using the following syntax:

context Typename::operationName(param1 : Type1, ...): ReturnType
body: -- some expression

- For example:

context Person::getCurrentSpouse() : Person
pre: self.isMarried = true
body: self.mariages->select(m | m.ended = false).spouse

Specifying Invariants

- The OCL expression can be part of an Invariant which is a Constraint stereotyped as an «invariant»
- For example (in context of type "Company"):

  self.numberOfEmployees > 50

- Specifies the invariant that the number of employees in the Company must be greater than 50
- With a little more detail:

  context c : Company **inv** enoughEmployees:
  c.numberOfEmployees > 50

Pre/Post Conditions

- Used to specify pre-conditions or post-conditions associated with an operation.
- For example:

context Typename::operationName(param1 : Type1, ...): ReturnType
pre parameterOk: param1 > ...
post resultOk : result = ...

Guards

- A guard is a Boolean condition that may or may not validate the triggering of an event occurrence. For example:

```
context ExpressionInOcl
inv: not self.guard.transition.getStateMachine().context.oclIsUndefined()
and
self.guard.transition.getStateMachine().context.oclIsKindOf(Classifier)
implies
contextualClassifier =
self.guard.transition.getStateMachine().context.oclAsType(Classifier)
and
self.bodyExpression.type.name = 'Boolean'
```

Messages

- To specify that communication has taken place, the hasSent ('^') operator is used.
- For example:

context Subject::hasChanged()
post: observer^update(12, 14)

More Features

- Collections
  - There is rarely a model where you don't need to express collections
  - select() and reject() operations used to define new collections from existing ones
  - Example of select():
    context Company **inv**:
    self.employee->select(age > 50)->notEmpty()
  - This specifies that the set of employees over the age of 50 is not empty.
  - Example of reject():
    context Company **inv**:
    self.employee->reject(isMarried)->isEmpty()
  - This specifies that the set of employees not married is empty.

Broken Constraints

- OCL can be used to check object states in a system
  - manually by hand
  - Use an OCL parser tool (various ones available)
- If an Object breaks a given constraint, an error in the implementation (or specification) has been found.
- OCL doesn't specify how to fix the problem (not it's job)

Summary

- OCL helps to specify an OO design
  - Precise and unambiguous specifications
    - Constraints on all types of objects and expressions
    - Pre/Post conditions
    - Set operations
  - A formal language
    - No ambiguity
    - Industry Standards