Patterns in OCL

Burkhart Wolff

Université Paris-Sud

Pattern-Matching Lambdas

- Proposal:
  - Hidden second-order combinators, implicitly accepting a lambda buried under first-order notation, are:

->iterate ->exists ->forall ->select
->collect ->any ->isUnique.

S->select(PATTERN | P x)

for example:

S->select(Seq{_, 3, a, ...} | a >= 15)

or

S->select(Tuple{name='mueller', sex=male, age= x, ...} | x >= 21)

Pattern-Matching Lambdas

- Proposal:
  - Hidden second-order combinators, implicitly accepting a lambda buried under first-order notation, are:

for example

```
S->select(Seq{_, 3, a, ...} | a >= 15)
```

or

```
S->select(Tuple{name=`mueller', sex=male, age= x, ...} | x >= 21)
```

or

```
S->select(a in Employee | P a )    for
     (S->select(a   |    a.oclIsKindOf(Employee) and P a ))
```

Pattern-Matching Lambdas

- Proposal:
  - Hidden second-order combinators, implicitly accepting a lambda buried under first-order notation, are:

  - Possibility: implicit Tuple-notation for Classes:

```
class Employee is Person
+ salary : Integer[0..1]
+ department_id : Integer [1]
end
```

Example:

- S->select(Employee{salary = x, department_id=5, ... }
  | x <> null and x>2000)