What You Always Wanted to Know about OCL and Never Dared to Ask

Martin Gogolla

What You Always Wanted to Know about OCL and Never Dared to Ask

Martin Gogolla

Personally, i never use OCL

What You Always Wanted to Know about OCL and Never Dared to Ask

Martin Gogolla

Personally, i never use OCL, i always use USE OCL

OCL is a nice, useful language,
for driving with your finger through a class diagram.

But there are some things worth to be debated ...

Technical aspects of OCL leading to debates

- Collection operations collect, collectNested; missing backward compatibility of collect
- Navigation and typing in n-ary associations; missing resp. wrong functionality
- Dot VS arrow; attention required after 20 OCL years; Seq{Set{7,8},Set{9}}->size, Seq{Set{7,8},Set{9}}.size
- Dot shortcut very useful, but difficult to explain technically
- 0..1/1..1 handling; changes to * (or the other way round) invalidates expressions
- Keyword/name harmonization and length: Set, Bag, Sequence, OrderedSet; at VS subOrderedSet
- Ordered set properties: Ord{7,8,7} =/<> Ord{8,7,7} (long) unclear in standard

Technical aspects of OCL leading to debates

- 3-valued VS 4-valued (VS 2-valued) logic;
  relationship to and harmonization with SQL
- Set{} VS oclEmpty(Set(Integer)), oclEmpty(Set(Person)), ...
- size() VS size [as in OCL 1.X]
- Referential transparency (A=B => f(A)=f(B)): E=E,
  Set{7,8}->asSeq()=Set{7,8}->asSeq() not required in standard
- Workarounds for explicitly typing of expressions with let,
  given albert:Prof > Person
  `let albertP:Person=albert in expr-albertP-is-Person`

Views on and uses of OCL

- Several implementations, used in many projects, no UML/MODELS conference without OCL papers
- OCL @ USE (University of Bremen) for 20 years
- Following quotes come from memory, not liable for trial
- Perdita Stevens: 'OCL is probably the best part of UML'
- Oystein Haugen: 'OCL is completely useless and inacceptable'
- Albert Zündorf: 'OCL has strange typing rules' (navigation results in Set(...), Bag(...), ...)
- Jean Bezivin: 'OCL navigation is too tedious, more automatism'
- OCL: object *CONSTRAINT* language;
  SQL: structured *QUERY* language; both good for *C* and *Q*
- OCL independent of UML, embedding into other langs (ATL, ...)
- OCL to large degree object-independent, collections and collections operations [OCL heart] applicable without objects

Thanks for your attention!