

Review of topics raised during the workshop

Tooling

1. XText is a nice tool for generating OCL editors.
2. OCL should be able to be integrated with other languages and tools.
3. An IDE4OCL is a good idea – it could be used to validate the UML definition process.

Foundations

1. OCL can be used to generate test cases (but tooling needed for industry use)
2. OCL scales to very large models – needs work on the translation.
3. The UML spec has holes – use OCL in test driven way to solve this.
4. Core UML should/should not have bi-directional links.

Textual Modelling

1. Keeping text and graphics in sync.
2. Keeping multiple overlapping views of a system in sync.
3. Path expressions could be a useful extension to OCL
4. OCL needs user defined libraries.
5. Transitive Closure operator added to OCL

OCL in the OMG

1. OCL spec is too formal (!) at least given the (perceived) Lack of tools to validate its formal aspects.
2. Lack of effort to work on this.

What are the key issues that should be addressed by the next generation OCL and Textual Modelling Languages?

1. User defined parameterized types.
 - a. Should we have round, square or angled brackets?
2. Adding helpers to meta-classes.
3. Adding anonymous functions (lambda expressions, closures).
4. Overloading and dynamic dispatch in OCL
5. Support for stereotypes.
6. Primitive type incompatibilities.
7. Remove implicit 'asSet' and 'collect' operations.
 - a. ... but they are useful.
 - b. confusion between '.' and '->' operations with collections.
 - c. have a new name 'all' for the implicit 'collect' operation.
 - d. Instead of 7(c) 'this op cannot be used when you have an implicit collection cast' e.g. sizeOf()
 - e. tooling should indicate the type of the expression as it is typed.
8. Possible conflict between -> and . since OCL 2.2
9. Use of '=' most languages use two versions:
 - a. Use '=='?
10. OCL reflection (a meaning for self.oclType().oclType().ownedOperation)
11. Concrete syntax to abstract syntax mapping.
12. Add a type construction algorithm to the OCL standard.
13. Adding framing conditions: 'modifies only':
 - a. Anything with 'modifies only' means everything else stays the same.
14. Adding transitive closure to OCL.
15. UML semantics with UML replaced by OCL to UML Executable transform
16. Plan for OCL > 2.2, each chapter defined by a tool checked model.
17. Have a clear separation between core with no casting and an upper-layer with appropriate syntax for casting. The spec should define a mapping from the upper layer to the lower layer. Tools must implement this.
18. Core OCL:
 - a. Theorem proving back end