Commutative Short Circuit Operators

Edward D. Willink

Willink Transformations Ltd, Reading, England,
ed_at_willink.me.uk

Lightning presentation at the 17th International Workshop in OCL and Textual Modeling, July 20, 2017, Marburg, Germany.

1 Commutative Short Circuit Operators - E.D.Willink

OCL's 4-level logic has been a source of much unhappiness and while various solutions have been suggested, none have met with enthusiasm. We look at where the unhappiness comes from and thereby suggest a new solution.

The OCL designers defined an underlying model in which all expressions have types. Consequently the mathematical concept of truth was reified by a `Boolean` type with associated Boolean library operations. The designers chose to avoid exceptions. This in combination with UML conformance required a `null` value for the missing value of properties with optional multiplicity, and an `invalid` value for everything bad that might be evaluated.

Unfortunately `null` and `invalid` pollute the simplicity of truths and so the Amsterdam Manifesto [1] elaborates Boolean operators with short-circuit like functionality for problems such as:

```
a <> null and a.doSomething()
```

However the operators remain commutative and so it is suggested that all terms are evaluated in parallel until the result is knowable. A Karnaugh Map defines the mapping from the `true` (T), `false` (F), `null` (ϵ) and `invalid` ($\perp$) values of `Left` and `Right` inputs to the `and` output.

Left	Right	and	requires	'and2'
T	T	T	T	T
T	F	F	F	F
T	$\perp$,ϵ	$\perp$	$\perp$	$\perp$
F	-		F	
F	T,F	F		F
F	$\perp$,ϵ	F		$\perp$
$\perp$,ϵ	-		$\perp$	
$\perp$,ϵ	T,F,$\perp$,ϵ	$\perp$		$\perp$

Parallel execution is an implementation nightmare and the intermediate `invalid` results can be inefficient. If we eliminate commutative short circuits, we find that `invalid` results are exceptional rather than normal.

```
a <> null requires a.doSomething()
```

A new `requires` operator imposes a left argument first evaluation order for `and`. This avoids the spurious `invalid` results from the right argument and clearly indicates the intent to handle non-truths. The `and` operator can then be used for truths only. Once static analysis verifies that neither left nor right input of an `and` operator can be `null` or `invalid`, an implementation may implement a regular `'and2'` operation that returns `invalid` for any `null` or `invalid` input.

A new `obviates` operator is also needed to regularize `or` short circuiting.

References

1. Cook, s., Kleppe, A., Mitchell, R., Rumpe, B., Warmer, j., Wills, A.: The Amsterdam Manifesto on OCL. December 1999. http://www4.informatik.tu-muenchen.de/publ/papers/CKR+99.pdf