True Lies: Lazy Contracts for Lazy Languages

Faithfulness is Better than Laziness

Markus Degen Peter Thiemann Stefan Wehr
{degen,thiemann,wehr}@informatik.uni-freiburg.de

Design by contract [Mey97] is a methodology for constructing correct software. It equips each operation with a contract, which consists of a precondition and a postcondition: All input must fulfill the precondition and then the output is also obliged to meet the postcondition. Thus, a contract provides a (partial) specification of an operation and an implementation of the operation must fulfill that contract.

Contract monitoring validates contracts dynamically: Each operation checks the precondition before performing its computation and checks the postcondition before returning to its caller. On violation of the precondition the operation raises an exception blaming its caller. Conversely, on violation of the postcondition the operation blames itself.

In the context of eager languages like Eiffel, Java, and Scheme, there is only one useful and sensible mode of contract monitoring, which we call *eager monitoring*. If the contract predicates are side-effect free, then eager monitoring guarantees meaning reflection and meaning preservation. Moreover, it reports contract violations faithfully and behaves idempotently.

Contributions. We have identified three principal modes of contract monitoring in lazy languages, thus arriving at a clearly defined design space for this problem. *Eager monitoring* fully evaluates all preconditions on function entry and all postconditions on function exit. *Semi-eager monitoring* does not evaluate preconditions for arguments which are not demanded by the function. Evaluation of postconditions is also driven by demand of the function's result. *Lazy monitoring* delays evaluation of a condition until all values that it depends on are evaluated by the program proper.

We provide implementations for the novel notion of eager monitoring as well as for lazy monitoring, using a new implementation strategy.

We assess the effectiveness of the three modes along five dimensions: meaning reflection, meaning preservation, faithfulness, idempotence, and implementability. The central insight is that contract monitoring in a lazy language cannot be faithful and meaning preserving at the same time.

References

[Mey97] Bertrand Meyer. *Object-Oriented Software Construction*. Prentice-Hall, 2nd ed., 1997.