Introduction to the basic concepts of computational reflection

Alexander Tamm

6th October 2003

Abstract

Computational reflection (hereafter *"reflection"*) is the ability of a programme to observe or change its code and aspects of its programming language at runtime. This paper is an introction to the basic concepts of reflection. Different methods and aspects of reflection are descussed. *Reification* is defined as a method by which a programme can express aspects of itself or the programming language used. Aspects of computer programmes can be observed and acted upon using *inspection.*

Reflection in computer programmes are supported by languages with *reflective architectures.* Two distinct styles of reflective architectures can be recognised, *procedural reflection* and *declarative reflection.* Declarative reflection is the ability of a language to reify a running programme and abstract data types used. Procedural reflection is the method by which a programme can affect aspects of the programming language used, ie. change the meaning of the operation. A new language *L'* is formed, as it were, based on - and expressed in - the original language *L* used. Since aspects of *L'* can be modified and a new language *L"* constructed, we stumble upon *reflective towers*, which are also discussed in this paper.

Object-oriented programming can receive a new aspect of abstraction through reflection. Object describing objects, or *meta-objects*, can be used. Meta-objects are discussed, as well as *meta-object protocols* or *MOP.*