Recurring Return on Modeling Investment: A Conceptual Modeling Language and Extensible Compiler

Quenio Cesar Machado dos Santos[1] and Raul Sidnei Wazlawick[2]

[1] Computer Sciences,
UFSC - Universidade Federal de Santa Catarina, Brazil,
queniodossantos@gmail.com
[2] Associate Professor of Computer Sciences Department,
UFSC - Universidade Federal de Santa Catarina, Brazil,
raul@inf.ufsc.br

Abstract. Proposes a textual programming language that enables conceptual modeling (similarly to UML classes/associations and OCL constraints) and a compiler that allows code generation (via extensible textual templates) to any target language or technology. Together, the language and the compiler make it feasible to specify (in a single high-level language) the information of ever-changing, increasingly distributed software systems. From this single source, the automated code generation keeps the implementations (across the different platforms and technologies) consistent with the specification. Also, as the technology landscape evolves, these textual models allow the recurring use of the investment made on their specification. Unlike other approaches, such as MDA and MPS, the built-in tooling support, along with the textual nature of this programming language and its extensible templates, facilitates the integration to the workflow of software developers, which is expected to promote its adoption.

Keywords: conceptual modeling, omg, mda, uml, ocl, mof, mps, mde, mdsd, er, entity-relationship model, programming language, compiler, code generation, model-driven software development, model-driven engineering, modeling investment, classes, associations, constraints, specification, software tools, metaprogramming, generative programming

1 Introduction

In order to address the challenges of the ever-changing, increasingly distributed technologies used on software systems, the Model-Driven Architecture (MDA [1]) initiative by the Object Management Group (OMG) has been promoting model-driven software development. In particular, MDA has guided the use of high-level models (created with OMG standards, such as UML [3], OCL [2] and MOF [4]) to derive software artifacts and implementations via automated transformations. As one of its value propositions, the MDA guide [1] advocates:

> "Automation reduces the time and cost of realizing a design, reduces the time and cost for changes and maintenance and produces results that ensure consistency across all of the derived artifacts. For example, manually producing all of the web service artifacts required to implement a set of processes and services for an organization is difficult and error-prone. Producing execution artifacts from a model is more reliable and faster."

The Metaprogramming System (MPS [5]) has shown that tooling can assist in integrating high-level, domain-specific models in the development workflow.

References

1. OMG: Object management group model driven architecture (mda) mda guide rev. 2.0 (2014). URL: http://www.omg.org/cgi-bin/doc?ormsc/14-06-01
2. OMG: OMG Object Constraint Language (OCL), Version 2.4 (2014). URL: http://www.omg.org/spec/OCL/2.4
3. OMG: Unified modeling language (uml), superstructure, version 2.5 (2015). URL: http://www.omg.org/spec/UML/2.5
4. OMG: Meta object facility (mof) core specification, version 2.5.1 (2016). URL: http://www.omg.org/spec/MOF/2.5.1
5. Voelter, M.: Generic tools, specific languages. Ph.D. thesis, Delft University of Technology (2014). URL: http://repository.tudelft.nl/view/ir/uuid%3A53c8e1e0-7a4c-43ed-9426-934c0a5a6522