Model-Driven Engineering Approach for OntoUML

Author: Dan Homola Supervisor: Ing. Robert Pergl, PhD.

Semi-automatic transformation of OntoUML models into object-oriented code

Motivation

OntoUML is an established language for designing ontologically well-founded conceptual models based on the Unified Foundational Ontology (UFO). As such, it is highly suitable for software engineering, as it offers semantically precise models. However, a Model-Driven Engineering approach has not been formulated for OntoUML yet — there are few tools that support the notation and even fewer that are capable of generating source code from a model.

Goals

1. Design and implement a semi-automated tool that transforms OntoUML models into object-oriented code
2. The tool should internally be split into two layers — one general, language-agnostic, and one implementing the transformation to a concrete object-oriented language (C#)

Model-Driven Engineering

Model-Driven Engineering (MDE) is an approach to software development that is based on domain models. These models are created on a conceptual level, therefore they are independent on a final implementation. After validation, they can be transformed (manually – or even better, automatically) into more concrete levels (even into source code) [1]. This very early validation aims to find errors in the solution design as soon as possible and therefore reduce costs.

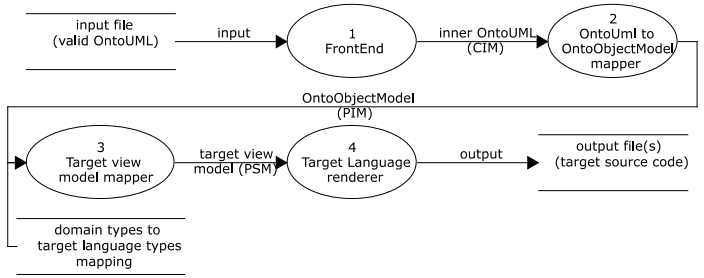
OntoUML

OntoUML is a modelling language (an UML profile) for building ontologically well-founded models. It is grounded in the Unified Foundation Ontology (UFO) models and originated in 2005 [2]. It is based on cognitive science and modal logic. As opposed to UML, it precisely defines several types of entities (e.g. Kinds, Categories, Roles, etc.) that reflect our perception of reality. Moreover, various relation types and metaattributes are added dealing with part-whole relations, (in)separability and more.

This allows analysts to create more accurate and information-rich models while retaining simplicity and being understandable even by non-technical people. This greatly facilitates communication and models quality, thus lowering the software development costs.

Implementation

The solution consists of a set of transformations that gradually transform OntoUML model into a target language source code and an application providing a framework for these transformations. The approach is illustrated by transforming RefOntoUml files (used by the NEMO OntoUML editor [3]) into C# classes. This architecture allows for easy addition of support for other input and output forms, being various tools and programming languages. The application was written in TypeScript for the Node.js platform.



Transformation sequence diagram

Results

1. Formal transformation of OntoUML model into a language-independent object-oriented model.
2. Transformation of the formal object-oriented model into C# source code that generates code with checks of compliance with the UFO rules.
3. Multiplatform, extensible software application implementing these transformations.
4. A C# utility library for working with associations and member-count bounded collections

Conclusions & Applicability

The transformations formulated in this thesis can be used as a basis for an integrated design environment that could benefit from both the Model-Driven Engineering and OntoUML. As demonstrated in the thesis' Case studies part, the resulting code is fairly concise and retains most of the OntoUML semantic constraints.

Bibliography

[1] Brambilla, M.; Cabot, J. Model-driven software engineering in practice. Online-Ausg. San Rafael, Calif.: Morgan & Claypool, 2012. ISBN 978-16-084-5882-0.

[2] Guizzardi, G. Ontological foundations for structural conceptual models. Enschede, The Netherlands: Centre for Telematics and Information Technology, Telematica Instituut, 2005. ISBN 90-75176-81-3.

[3] Sales, T. P.; Amorim, V.; Brasileiro, F. OntoUML lightweight editor (OLED) [software]. 2016, [Accessed 2016-01-21]. Available from: https://github.com/nemo-ufes/ontouml-lightweight-editor