**D3.1.1 Review of model-to-model transformation approaches and technologies**

ModelWriter

Text & Model-Synchronized Document Engineering Platform

Project number: ITEA 2 13028

Work Package: WP3

Task: T3.1 - Review of M2M transformation approaches

Edited by:

Ferhat Erata <ferhat.erata@unitbilisim.com> (UNIT)

Moharram Challenger <moharram.challenger@unitbilisim.com> (UNIT)

Date: 07-June-2015

Document version: 1.0.0

Apart from the deliverables which are defined as public information in the Project Cooperation Agreement (PCA), unless otherwise specified by the consortium, this document will be treated as strictly confidential.

Document History

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Author(s) | Date | Remarks |
| 0.1.0 | Ferhat Erata  Moharram Challenger | 07-June-2015 | Draft |
| 1.0.0 | <name> | <date> | Initial Release |
|  |  |  |  |

Table of Contents

Contents

[Document History 2](#_Toc424549155)

[1. Introduction 4](#_Toc424549156)

[1.1. Role of the deliverable 4](#_Toc424549157)

[1.2. The List of Technical Work Packages 4](#_Toc424549158)

[1.3. Structure of the document 4](#_Toc424549159)

[1.4. Terms, abbreviations and definitions 5](#_Toc424549160)

[2. State-of-the-technology 6](#_Toc424549161)

[2.1. Preliminaries 6](#_Toc424549162)

[3. Comparison 8](#_Toc424549163)

[4. Selected approach 9](#_Toc424549164)

[5. Conclusion and way forward 10](#_Toc424549165)

[References 11](#_Toc424549166)

[Appendixes 12](#_Toc424549167)

[Appendix 1 12](#_Toc424549168)

1. Introduction
   1. Role of the deliverable

This document consists of a review of model transformation approaches in general and model to model transformation approaches in specific. Also, the document will discuss the selection of the most convenient and widely used transformation approach in the industry for inclusion into the ModelWriter tool. It may be up-dated depending on the further details and requirements we get during the project.

* 1. The List of Technical Work Packages

|  |  |
| --- | --- |
| UC Code | Requirements derived from |
| WP2 | Semantic Parsing and Generation of Documents and Documents Components |
| WP3 | Model to/from Knowledge Base (synchronization mechanism) |
| WP4 | Knowledge Base Design and Implementation |
| WP6 | Architecture, Integration and Evaluation |

* 1. Structure of the document

This document is organized as follows:

Chapter 1 introduces the document.

Chapter 2 reviews the model transformation approaches in different groups and for various uses which are available as the state-of-the-technology.

Chapter 3 compares the available approaches and compares them considering their pros and cons.

Chapter 4 discusses the approach selected for the ModelWriter tool

Annex 1 Demonstration examples of various transformation approaches

* 1. Terms, abbreviations and definitions

|  |  |
| --- | --- |
| Abbreviation | Definition |
| M2M | Model to Model Transformation |
| M2T / M2C | Model to Text or Model to Code Transformation |
| WP | Work Package |
| UC | Use Case |
| KB | Knowledge base |
|  |  |

1. State-of-the-technology

The notion of model transformation is central to model-driven development. A model transformation, which is essentially a program which operates on models, can be written in a general-purpose programming language, such as Java. However, special-purpose model transformation languages can offer advantages, such as syntax that makes it easy to refer to model elements. For writing bidirectional model transformations, which maintain consistency between two or more models, a specialist bidirectional model transformation language is particularly important, because it can help avoid the duplication that would result from writing each direction of the transformation separately [4].

* 1. Preliminaries

Model transformations and languages for them have been classified in many ways [1] [2]. Some of the more common distinctions drawn are:

Number and type of inputs and outputs:

In principle a model transformation may have many inputs and outputs of various types; the only absolute limitation is that a model transformation will take at least one model as input. However, a model transformation that did not produce any model as output would more commonly be called a model analysis or model query.

Endogenous versus exogenous:

Endogenous transformations are transformations between models expressed in the same language. Exogenous transformations are transformations between models expressed using different languages [3]. For example, in a process conforming to the OMG Model Driven Architecture, a platform-independent model might be transformed into a platform-specific model by an exogenous model transformation.

Unidirectional versus bidirectional:

A unidirectional model transformation has only one mode of execution: that is, it always takes the same type of input and produces the same type of output. Unidirectional model transformations are useful in compilation-like situations, where any output model is read-only. The relevant notion of consistency is then very simple: the input model is consistent with the model that the transformation would produce as output, only.

For a bidirectional model transformation, the same type of model can sometimes be input and other times be output. Bidirectional transformations are necessary in situations where people are working on more than one model and the models must be kept consistent. Then a change to either model might necessitate a change to the other, in order to maintain consistency between the models. Because each model can incorporate information which is not reflected in the other, there may be many models which are consistent with a given model.

* 1. M2M Transformation Languages

Ad hoc

Atom3

Extend2

Higher Order Transformations (HOTs)

ATL:

ATL (ATL Transformation Language) is a model transformation language and toolkit developed and maintained by OBEO and INRIA-AtlanMod (see [1]). It was initiated by the AtlanMod team (previously called ATLAS Group). In the field of Model-Driven Engineering (MDE), ATL provides ways to produce a set of target models from a set of source models. Released under the terms of the Eclipse Public License, ATL is an M2M (Eclipse) component, inside of the Eclipse Modeling Project (EMP).

ATL is based on the QVT which is an Object Management Group standard for performing model transformations. It can be used to do syntactic or semantic translation. ATL is built on top of a model transformation Virtual Machine.

GReAT: a transformation language available in the GME

Epsilon family (see [2]): a model management platform that provides transformation languages for model-to-model, model-to-text, update-in-place, migration and model merging transformations. (ETL)

Henshin (see [3]): a model transformation language for EMF, based on graph transformation concepts, providing state space exploration capabilities

JTL: a bidirectional model transformation language specifically designed to support non-bijective transformations and change propagation (see [4]).

Kermeta: a general purpose modelling and programming language, also able to perform transformations

Lx family (see [5]): a set of low-level transformation languages

M2M is the Eclipse implementation of the OMG QVT standard

Mia-TL: a transformation language developed by Mia-Software

MOF Model to Text Transformation Language: the OMG has defined a standard for expressing M2T transformations

MOLA (see [6]): a graphical high-level transformation language built in upon Lx.

MT: a transformation language developed at King's College, London (UK) (based on Converge PL)

QVT: the OMG has defined a standard for expressing M2M transformations, called MOF/QVT or in short QVT. QVTd (Declarative), QVTo (Operational/Procedural)

SiTra [7]: a pragmatic transformation approach based on using a standard programming language, e.g. Java, C#

Stratego/XT: a transformation language based on rewriting with programmable strategies

Tefkat: a transformation language and a model transformation engine

Tom: a language based on rewriting calculus, with pattern-matching and strategies

UML-RSDS [8]: a model transformation and MDD approach using UML and OCL

VIATRA2: a framework for transformation-based verification and validation environment

Categories:

M2M

M2T: Acceleo, JET, Xpand, MOFScript

Eclipse based

Industrial Tooling

* 1. M2M Transformation Languages

1. Comparison
2. Selected approach
3. Conclusion and way forward

References

Czarnecki; Helsen (2006). "Feature-based survey of model transformation approaches". IBM Systems Journal. doi:10.1147/sj.453.0621.

Jump up ^ Stevens, Perdita (2008). "A landscape of bidirectional model transformations". Berlin / Heidelberg: Springer. Doi:10.1007/978-3-540-88643-3\_10.

Tom Mens, Pieter Van Gorp: A Taxonomy of Model Transformation. Electr. Notes Theor. Comput. Sci. 152: 125-142 (2006)

Wikipedia- Model transformation language, available at: <https://en.wikipedia.org/wiki/Model_transformation_language>, last access: July 2015.

Appendixes

Appendix 1

Doloreptium dic temquo qui voluptate dellabo. Ut labo. Et pel maxim resed molore nit andios volorumenis eum enihiti nciasim olorepeles ea aut maximolupta vendae sundites dolecaborem ni nonseque poreri dolora plati quid ut lab iuscia volorio rporemp edisitatis sed quis aut explit, to cuptas sendae volor ad moloreium dollat lande iduci dolupta eribus etur, sintem quae videbit estiore ommosapel ea delia volesciustio quiam, sit evero blabore