ATL: a QVT-like Transformation Language

Frédéric Jouault Freddy Allilaire Jean Bézivin Ivan Kurtev Patrick Valduriez
ATLAS (INRIA and LINA), Université de Nantes
{frederic.jouault,freddy.allilaire,jean.bezivin,ivan.kurtev}@univ-nantes.fr, patrick.valduriez@inria.fr

Abstract

In the context of Model Driven Engineering (MDE), models are the main development artifacts and model transformations are among the most important operations applied to models. A number of specialized languages have been proposed in order to specify model transformations. The OMG has, for instance, adopted the QVT specification. Apart from the software engineering properties of transformation languages, the availability of high quality tool support is also of major importance for the industrial adoption and ultimate success of MDE. In this paper, we present ATL: a QVT-like model transformation language and its execution environment based on the Eclipse framework.

Categories and Subject Descriptors D.3.2 [*Language Classifications*]: specialized application languages

General Terms Languages

Keywords Model-driven engineering, model transformation, QVT, ATL

1. Introduction

Model transformations play an important role in the Model Driven Engineering (MDE) approach. Developing model transformation definitions is expected to become a common task in model driven software development. Software engineers should be supported in performing this task by mature MDE tools and techniques in the same way as they are currently supported by traditional IDEs, compilers, and debuggers in their everyday programming work. One direction for providing such support is to develop domain-specific languages designed to solve common model transformation tasks. A number of model transformation languages have thus been proposed [1, 2, 3].

In this paper, we describe ATL (ATLAS Transformation Language): a transformation language developed as a part of the AMMA (ATLAS Model Management Architecture) platform [4]. We have built development tools for ATL on top of Eclipse. These tools, along with documentation documents and examples are available in the ATL Eclipse/GMT subproject [5].

The paper is organized as follows. Section 2 presents ATL. Section 3 describes the available tools and Section 4 concludes.

OOPSLA'06 October 22–26, 2006, Portland, Oregon, USA.
ACM 1-59593-491-X/06/0010.

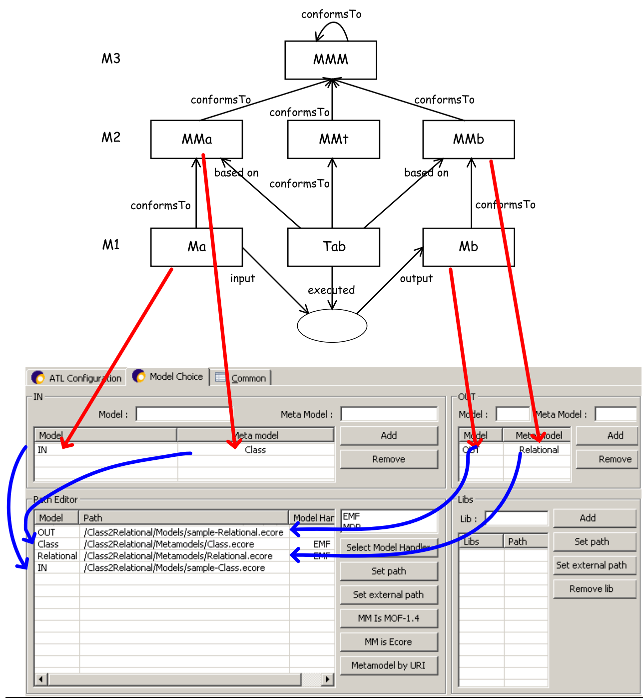

Figure 1. ATL launch configuration

2. ATL

ATL is applied in the context of the transformation pattern shown in the upper part of Figure 1. In this pattern, a source model *Ma* is transformed into a target model *Mb* according to a transformation definition *mma2mmb.atl* written in the ATL language. These three elements are models respectively conforming to the *MMa*, *MMb*, and *ATL* metamodels. All metamodels conform to the metametamodel (*MOF* in the context of OMG standards). Source and target models and metamodels may be expressed in XMI. Metamodels can also use the more convenient KM3 notation [6].

The transformation scenarios supported by QVT [2] (Query / View / Transformation) are also supported by ATL [7]. Moreover, a large part of QVT requirements are taken into account by ATL, such as compatibility with MOF [8], XMI, and the use of OCL [9] for navigation. Therefore, ATL is a QVT-like language.

ATL contains a mixture of declarative and imperative constructs. We encourage a declarative style of specifying transformations. However, developers may resort to the imperative features if necessary. A more detailed presentation of the language and examples can be found in [10] and [5]. A more detailed and formal specification of the ATL semantics can be found in [11].

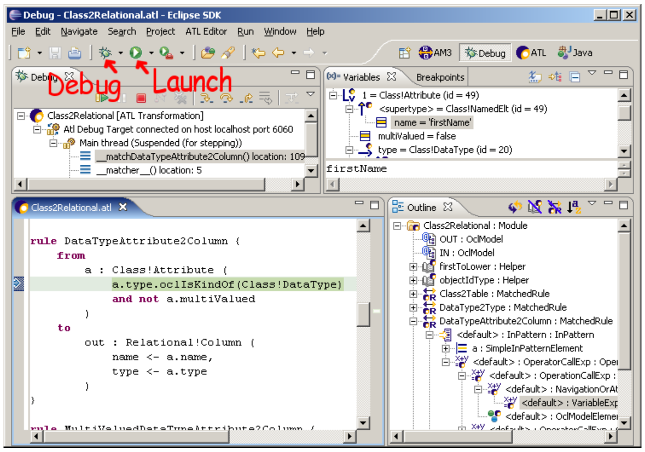

Figure 2. ATL editor and debugger screenshot

3. ATL Development Tools

ATL Development Tools (ADT) is developed under the ATL Eclipse/GMT subproject [5]. ADT is composed of the ATL transformation engine and the ATL Integrated Development Environment (IDE): an editor, a compiler and a debugger.

3.1 Engine

The ATL engine is responsible for dealing with core ATL tasks: compilation and execution. ATL transformations are compiled to programs in specialized byte-code. Byte-code is executed by the ATL Virtual Machine (VM). The VM is specialized in handling models and provides a set of instructions for model manipulation. The VM may run on top of various model management systems such as the Eclipse Modeling Framework (EMF) [12].

3.2 Editing

The ATL editor supports syntax highlighting, error reporting, and outline view (a tree-based representation of the ATL program). The bottom left-hand part of Figure 2 shows this editor. The bottom right-hand part shows the corresponding outline.

3.3 Building and Launching

The ATL compiler is automatically called for each ATL file in each ATL project during the Eclipse build process. By default, this process is triggered when a file is modified.

Executing an ATL transformation requires the declared source and target models and metamodels to be bound to actual models. Figure 1 gives a screenshot of ATL launch configuration wizard and shows the correspondences between the user interface and the operational context of the transformation. The four top arrows map models and metamodels to their declarations whereas the bottom arrows map the declarations to the corresponding files.

3.4 Debugging

ATL transformations may be debugged using the same configuration used for launching. The only difference is that we now use the *Debug* button (see Figure 2) instead of *Launch*. Transformations can be executed step-by-step or run normally. Execution stops when an error occurs or a breakpoint is reached. The current context may be analyzed using the variable view (see the top right part of Figure 2). It enables simple navigation in source and target models from the current context (rule or helper).

4. Conclusions

In this paper we presented ATL: a hybrid model transformation language developed as a part of AMMA. ATL is supported by a set of development tools built on top of the Eclipse environment: a compiler, a virtual machine, an editor, and a debugger. ATL allows both imperative and declarative approaches to be used in transformation definitions depending on the problem at hand.

ATL is currently used or evaluated by more than 100 sites, academic and industrial. There is an initial library of ATL transformations and some documentation (a user manual, an installation guide, and a starter guide) available in open source from the GMT Eclipse project. The current state of ATL tools already allows solving non-trivial problems. This is demonstrated by the increasing number of implemented examples and the interest shown by the rapidly growing ATL user community.

Acknowledgments

This work is partially supported by ModelWare, IST European project 511731 and by System@tic Paris-region, the french cluster in complex system design and management.

References

[1] Agrawal, A., Karsai, G., Kalmar, Z., Neema, S., Shi, F., Vizhanyo, A.: The Design of a Language for Model Transformations. Journal of Software and System Modeling (2005) in review.

[2] OMG: MOF 2.0 Query / Views / Transformations RFP, OMG Document ad/2002-04-10. (2002)

[3] Varró, D., Varró, G., Pataricza, A.: Designing the automatic transformation of visual languages (2002)

[4] Bézivin, J., Jouault, F., Rosenthal, P., Valduriez, P.: Modeling in the Large and Modeling in the Small. In Uwe Amann, Mehmet Aksit, A.R., ed.: Proceedings of the European MDA Workshops: Foundations and Applications, MDAFA 2003 and MDAFA 2004, LNCS 3599, Springer-Verlag GmbH (2005) 33–46

[5] ATLAS team: ATLAS Transformation Language (ATL) Home page, `http://www.eclipse.org/gmt/atl/`. (2006)

[6] Jouault, F., Bézivin, J.: KM3: a DSL for Metamodel Specification. In: Proceedings of 8th IFIP International Conference on Formal Methods for Open Object-Based Distributed Systems, LNCS 4037, Bologna, Italy (2006) 171–185

[7] Jouault, F., Kurtev, I.: On the Architectural Alignment of ATL and QVT. In: Proceedings of ACM Symposium on Applied Computing (SAC 06), model transformation track, Dijon, Bourgogne, France (2006)

[8] OMG: Meta Object Facility (MOF) 2.0 Core Specification, OMG Document formal/2006-01-01. (2006)

[9] OMG: UML OCL 2.0 Specification, OMG Document ptc/2003-10-14. (2003)

[10] Jouault, F., Kurtev, I.: Transforming Models with ATL. In Bruel, J.M., ed.: Satellite Events at the MoDELS 2005 Conference: MoDELS 2005 International Workshops OCLWS, MoDeVA, MARTES, AOM, MTiP, WiSME, MODAUI, NfC, MDD, WUsCAM, Montego Bay, Jamaica, October 2-7, 2005, Revised Selected Papers, LNCS 3844, Springer Berlin / Heidelberg (2006) 128–138

[11] Di Ruscio, D., Jouault, F., Kurtev, I., Bézivin, J., Pierantonio, A.: Extending AMMA for Supporting Dynamic Semantics Specifications of DSLs. Technical Report 06.02, LINA (2006)

[12] Budinsky, F., Steinberg, D., Ellersick, R., Merks, E., Brodsky, S.A., Grose, T.J.: Eclipse Modeling Framework. Addison Wesley (2003)