2010 18th IEEE International Requirements Engineering Conference

MDGore: Towards Model-Driven and Goal-Oriented Requirements Engineering

Rui Monteiro, João Araújo, Vasco Amaral, Pedro Patrício
CITI/Departamento de Informática
Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa
2829-516 Caparica, Portugal
{rm.chambel | pedropt}@gmail.pt, {va | ja}@di.fct.unl.pt

***Abstract*—Goal-Oriented Requirements Engineering (GORE) has received increasing attention over the past few years. There are several goal-oriented approaches, each one using different kinds of models. We believe that it would be useful to relate them or even transform them automatically, in order to, for example, smooth the process of migration from one model to another. In this paper we propose the definition and implementation of goal model transformations between i* and KAOS. This approach will contribute to relate the concepts of i* and KAOS models and will help, for example, a development team in making the decision on which approach to follow, according to the nature of the project.**

Keywords- Goal-Oriented Requirements Engineering, Model-Driven Development.

I. INTRODUCTION

Goal-Oriented Requirements Engineering (GORE) helps identifying, organizing and structuring requirements as well as exploring and evaluating alternative solutions to a problem [3]. There is a wide variety of Goal Modeling Languages (GML), such as i* [11] and KAOS [4]. Despite the existing work on these approaches, such variety, according to [6], neither facilitates the selection of the appropriate approach, nor makes it easier to learn the main differences of each one. Moreover, it may be desirable to convert a model into another because a new policy for requirements elicitation and analysis in an organization dictates that a migration to another goal modeling approach must be followed, preferably in an automatic way.

To achieve requirements models' transformations, Model-Driven Development (MDD) [10] techniques can be used. These techniques range from metamodeling to transformation languages and tools. The goal of MDD is to automate the process of creating new software and to facilitate their evolution in changing environments through model transformation. The final result is a more productive software development process. Typically, in MDD, a model is associated with an abstract syntax that is defined by a metamodel. In other words, a model must be constructed following certain rules, i.e., in agreement with a certain metamodel, which defines the model abstract syntax. In the same way, the metamodel is a model of a modeling language and must conform to the rules defined in the meta-metamodel. Once the metamodels and models are defined, the model transformations are used to derive a model from one or more models. That is, a model transformation takes as input a model conforming to a given metamodel and produces as output another model conforming to another given metamodel. In the context of model transformation, our work uses the graph transformation approach. A graph transformation [1] is a technique for model transformation that involves models that can be represented as graphs, which is the case of i* and KAOS models.

In this work we propose a model transformation framework, called MDGore (Model-Driven in Goal-oriented requirements engineering), to transform i* models into KAOS models using rules defined in ATL (Atlas Transformation Language) [12]. To achieve this, i* and KAOS models are specified using our tools that were developed in an Eclipse platform, LDE i* [2] and modularKAOS framework [8]. The application of the transformation rules is done at abstract syntax level, defined in the metamodel of each one of the frameworks mentioned, and uses the graph transformation technique in the MDD context, using the transformation language ATL. This approach allows relating the two goal modeling languages in order to, for example in a business or academic environment, facilitate the conversion of models when desired. Furthermore, another possible contribution would be to help the development team in making the decision on which approach to follow, according to the project nature.

II. MDGORE

MDGore is a model-to-model transformation approach between i* and KAOS models. Figure 1 gives an overview of our framework. The dashed rectangle is the focus of our approach. The source model is the i* SR (Strategic Rationale) model which is transformed into a KAOS Goal model through the application of ATL rules. An SR model is specified using the LDE i* framework and it conforms to the i* metamodel [11] implemented in this framework. A KAOS model conforms to the KAOS metamodel [5] implemented in modularKAOS. ATL rules conform to the ATL metamodel. Finally, these three metamodels were implemented using the Ecore formalism [9] and, therefore, conform to the Ecore meta-metamodel. This framework is semi-automatic, since the initial phase of the transformation process is applied with user intervention, to make decisions about the mapping of some i* elements into the

1090-705X/10 $26.00 © 2010 IEEE
DOI 10.1109/RE.2010.60

corresponding KAOS elements. We define a "decision model" so that the user can make and record their choices.

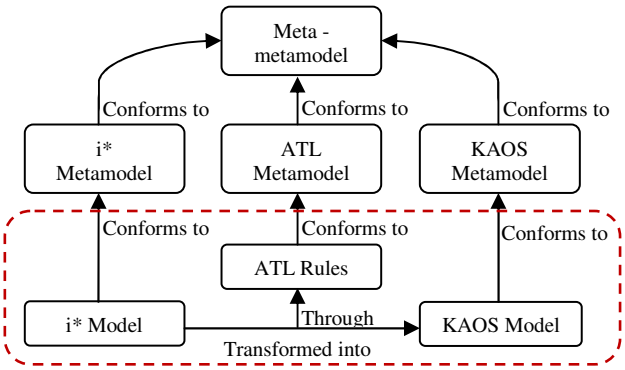

Figure 1. Process overview of the MDGore framework.

One of the problems associated with the transformations between models of different approaches is the information loss. This problem arises from the fact that the abstract syntaxes of the approaches involved in the transformations are different and as a consequence it is not always possible to transform all the elements automatically. This problem is also present when relating i* and KAOS models. To address this problem we introduce the notion of a Log model. This model keeps the information of all the elements that are not possible to transform, allowing users to complement the model generated by the transformations according to their discretion. The application of the transformation rules results in two models: one KAOS Goal model with the transformed elements and a Log model, with the elements of i* SR model, which were not able to be transformed.

Another concern that we had was to ensure traceability in the transformation process. Through attributes in the KAOS metamodel, we store the history of the transformation in the transformed KAOS model. For example, when an i* element is transformed into a KAOS element, the type of the i* element is stored into an attribute of the KAOS element. This solution enables, in future, to consider the possibility of transforming a KAOS Goal model back to the corresponding i* SR model, ensuring bidirectionality in the transformation process. Finally, our framework is flexible enough to change the transformed model, in particular by removing, adding or changing elements in the model. To implement the transformation rules, a study of relations between the elements of the two selected approaches was made, as shown in Table I.

TABLE I. Mappings of i* elements into KAOS elements

i*	KAOS
Goal	Goal
Task	Expectation
	Requirement
Softgoal	Softgoal
Resource	Entity
Actor, Position, Agent, Role	SystemAgent
	EnvironmentAgent
Decomposition	AndRefinement
	ConcernsLink
Means–End Link	AndRefinement
	OrRefinement
Is-part-of Association	Aggregation
ISA Association	Inheritance
Break, Some-, Hurt	Conflict
Make, Some+, Help	OrRefinement
Dependency	AndRefinement

There is some related work on MDD and GORE. For example, [7] proposes a framework for tracing aspects from requirements goal models to implementation and testing. It is provided mechanisms for transforming models into aspect-oriented programs. The main difference between the two approaches is that the approach in [7] uses model-to-code and MDGore uses model-to-model transformation.

III. CONCLUSIONS AND FUTURE WORK

We have proposed a semi-automatic model-to-model transformation framework, called MDGore, to relate GORE approaches. In this work, we focus on KAOS and i* approaches and our framework allows to transform i* models into KAOS Goal models. For these models, transformation rules were defined at the abstract syntax level, using a graph transformation technique (ATL). This approach allows relating the two goal modeling languages in order to enable automatic conversion between them or to help a development team in making the decision on which approach to adopt. Future work includes the generalization of the transformation process to other GORE approaches.

ACKNOWLEDGMENTS

This project is partially financed by the Portuguese FCT MCTES and BATICS (PTDC/EIA/65798/2006).

REFERENCES

[1] K. Czarnecki, S.Helsen, "Classification of model transformation approaches", 2nd OOPSLA'03 Workshop on Generative Techniques in the context of Model-Driven Architecture, USA, October 2003.

[2] A. Dias, V. Amaral, J. Araújo, "Towards a Domain Specific Language for a Goal-Oriented Approach based on KAOS", (RCIS'09), Fès, Morocco, April 2009.

[3] A. Lamsweerde, "Goal-Oriented Requirements Engineering: A Guide Tour", RE'01, Toronto, Canada, 2001.

[4] A. Lamsweerde, "Requirements Engineering: From System Goals to UML Models to Software Specefications", Willey, 2009.

[5] R. Matulevicius, P. Heymans. "Analysis of KAOS Meta-model", Technical report, University of Namur, Belgium, 2005.

[6] R. Matulevicius, P. Heymans, "Comparing Goal Modelling Languages: an Experiment", REFSQ'07, Norway, 2007.

[7] N. Nan, Y. Yu, B. González-Baixauli, N. Ernst, J. Leite, J. Mylopoulos, "Aspects across Software Life Cycle: A Goal-Driven Approach", Transactions on AOSD, Vol. VI, 2009, Springer.

[8] C. Nunes, J. Araújo, V.Amaral, Carla Silva, "A Domain Specific Language for the i* Framework", ICEIS'09, Milan, Italy, May 2009.

[9] D. Steinberg, F. Budinsky, M. Paternostro, E. Merks, "EMF Eclipse Modeling Framework, 2nd Edition", Addison-Wesley, 2008.

[10] M. Völter, T. Stahl, "Model-Driven Software Development Technology, Engineering, Management", Wiley, 2006.

[11] E. Yu, "Modelling Strategic Relationships for Process Reengineering", P.h.d. thesis, University of Toronto, Canada, 1995.

[12] The ATL User Guide: http://wiki.eclipse.org/ATL/User_Guide.