A new Approach to Model Transformation using Graph Transformation System

ZHIHANG YAO[1] & THOMAS H. KOLBE[2]

1 Introduction

Nowadays, many application data models are developed and maintained based on the so-called 'Model Driven Architecture (MDA)' in the fields of software and systems engineering. In general, the MDA stipulates that the data model shall be first defined as an abstract platform-independent model (PIM) and later automatically translated into different platform-specific models (PSM) depending on the target application environments (cf. GASEVIC et al. 2006). A typical case of applying model transformation in real-world applications is the automatic derivation of a relational database schema from an object-oriented data model for realizing the efficient management of large and complex-structured data by making full use of the capabilities of the relational database management systems (RDBMS). According to the literature (cf. NG & LEARMONT 2002), the general conceptual solution is to map the source and target data models onto computer-interpretable formats, such that the model transformation process can be automatically performed by applying a set of user-definable mapping rules. When examining both model representations, it can be seen that they both can be represented by a graph-like structure and can be descriptively formalized as typed and attributed graphs. In addition, the mapping relationships between the different model entities such as classes and tables can also be expressed using directed graph edges connecting the respective graph nodes (cf. Figure 1). Therefore, the model transformation task could be abstracted to a graph transformation problem and accomplished using Graph Transformation Systems (cf. TAENTZER et al. 2005).

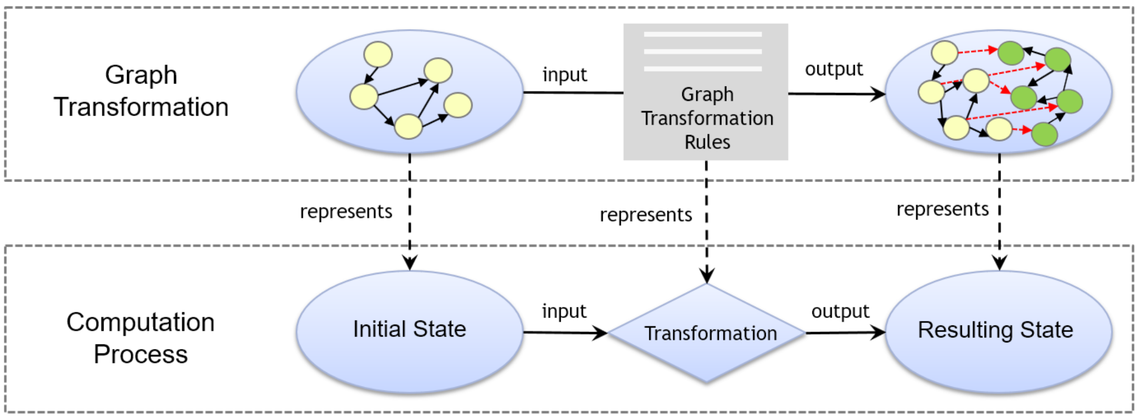

Fig. 1: Conceptual Workflow of using Graph Transformation Systems for performing a model transformation process

[1] virtualcitySYSTEMS GmbH, Marktplatz 2, D-85567 Grafing, E-Mail: zyao@virtualcitysystems.de
[2] Technische Universität München, Lehrstuhl für Geoinformatik, Arcisstr. 21, D-80333 München, E-Mail: thomas.kolbe@tum.de

2 Graph Transformation System

The fundamental concept of using graph transformation systems is an algebraic approach which employs typed and attributed graphs to represent the processing logics at an abstract level for solving specific problems ranging from software engineering up to computation simulations (cf. EHRIG et al. 2005). Basically, the initial state of the computation process shall be first formulated as a typed attributed graph called 'host graph' which is treated as the input for the graph transformation. This transformation process can be carried out by means of a set of user-defined transformation rules called 'graph transformation rules', which uses graph representations to formalize the computation logics and can be denoted as $\{r_1,\ r_2, \dots, r_n\}$. Each rule is equivalent to a match morphism denoted as $r: L \to R$ where L is called left-hand side (LHS) graph, whereas R is called right-hand side (RHS) graph. Both LHS and RHS graphs are also typed and attributed and the LHS graph is considered as a match pattern which could be algebraically isomorphic to every member of those graphs $\{G_1',\ G_2',\ \dots\ ,\ G_n'\}$ that are subsets of the host graph G_S, where $L\ \cong\ G_x'$ and $\{G_1',\ G_2',\ \dots\ ,\ G_n'\} \subseteq\ G_S$. While running a graph transformation, each of the given rules will be checked by the graph transformation system. If a match $m(L)$ of LHS graph has been found in the host graph, the respective graph transformation rule will be considered applicable and the matched subgraph in the host graph will be substituted by the corresponding RHS graph. Subsequently, the modified host graph will in turn be treated as the input for the next transformation step. The entire graph transformation process will be continued by successively processing the transformation rules until there is no further match of their LHS graphs can be found in the host graph. After completing the graph transformation process, the computation result can be yielded by interpreting the resulted host graph and can also be graphically displayed to the human operators.

3 Example: Mapping OO-Model onto Relational Model

In this section, an example of using graph transformation systems for model transformations is presented. It is realized using a JAVA-based software tool developed on top of the Open Source graph transformation system engine AGG (cf. TAENTZER 1999) and allows to automatically perform the mapping of an object-oriented model onto a relational database model. The conceptual workflow of the developed approach mainly consists of the following four steps (cf. Figure 2): In the first step, the XML schema definition (XSD) file containing the contents of the input object-oriented model is read and the parsed data model is subsequently mapped onto a graph structure. The respective classes along with their attributes and associations are represented as a set of typed attributed nodes and edges which can fully reflect the semantics of the input data model. In the second step, the mapped graph is passed to the graph transformation system where a number graph transformation rules are predefined to formulate the relevant mapping rules for performing the transformation from the object-oriented model to the desired relational database model. For this step, AGG provides an intuitive graphical editor tool which allows users to conveniently define the transformation rules by dragging and dropping the graph elements in an interactive way. After completing the graph transformation process, a set of new graph objects representing the derived relational database model as well as the mapping relationships between the source and target model objects are generated.

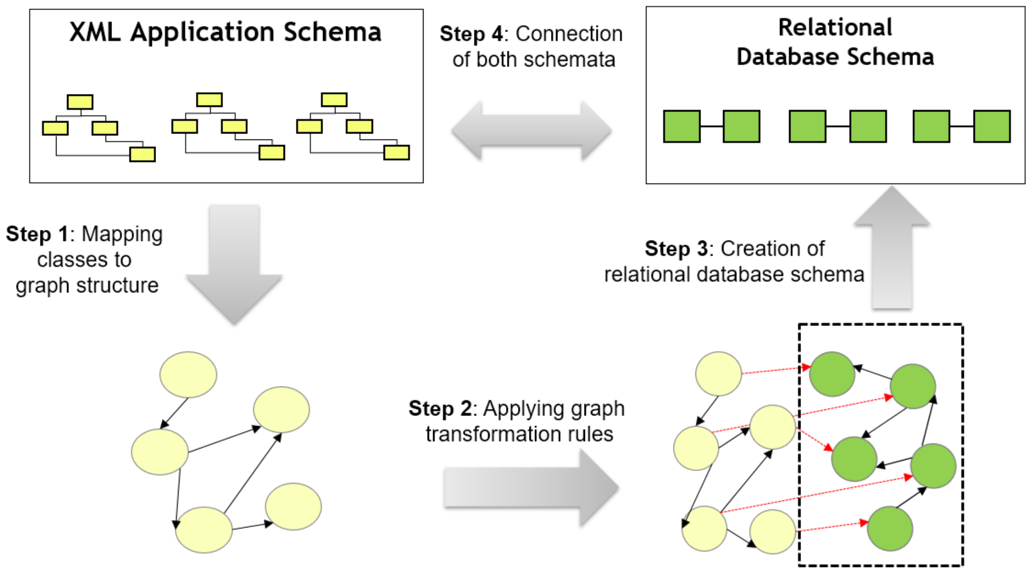
XML Application Schema
Step 4: Connection of both schemata
Relational Database Schema
Step 1: Mapping classes to graph structure
Step 3: Creation of relational database schema
Step 2: Applying graph transformation rules

Fig. 2: Workflow of deriving a relational database schema from an XML application schema using graph transformation systems

In the third step, the subgraph representing the newly derived relational database model shall be retrieved from the resulted graph by interpreting the respective graph objects for reconstructing the relational database structure. This can be done by iterating through all those nodes and edges that represent the relational database objects such as database tables, columns, sequences, indexes, and foreign key constraints etc. It is also possible to automatically generate the individual SQL statements for creating different types of database schema according to the specific database products like Oracle and PostgreSQL (cf. YAO & KOLBE 2017). In the last step, the mapping relationships between the XML application schema and the its relational database schema can also be parsed from the output graph and formally expressed using a XML-based document which allows to be easily transmitted between and analyzed in computer applications as well as explored by human users.

4 Conclusions and Outlook

This extended abstract presents a new graph-based approach for realizing model transformations from object-oriented data models to spatially-extended relational database schemas. This approach differs from the classical approaches of OO→relational model transformation, because it allows to express complex and user defined rules (as graph transformation rules), which can transform special configurations of input OO data model entities to special configurations of target relational data model structures. Also optimization rules on the input data model as well as on the resulting relational schema can be expressed using the same language. The concept is currently being employed to automatically derive relational database schemas for CityGML

Application Domain Extensions (ADE) for the Open Source 3D geodatabase 3DCityDB (www.3dcitydb.org).
A brief review of the relevant definitions for graph transformation systems was given first to show their key concepts and mechanism, which allows to facilitate the model transformation task in a systematic way. As a practical example showing how it works, the Open Source graph transformation system tool 'AGG' has been chosen for developing a model transformation tool which allows for automatically deriving a relational database schema from a given XML application schema with an object-oriented data structure. Future work may focus on improving the developed transformation tool by fine-graining or extending the existing transformation rules to make the derived database schema much more efficient in terms of query and processing performance.

5 References

EHRIG, H.; ERMEL, C.; GOLAS, U. & HERMANN, F., 2015: Graph and Model Transformation - General Framework and Applications. Springer Berlin Heidelberg, ISBN: 9783662479797.

GAAEVIC, D.; DJURIC, D.; DEVEDZIC, V. & SELIC, B., 2006: Model Driven Architecture and Ontology Development. Springer New York, Inc., ISBN: 3540321802.

NG, T. C. T. & LEARMONT, T. R., 2002: Rule-based approach to object-relational mapping strategies. U.S. Patent 6,360,223, issued March 19, 2002.

TAENTZER, G.; EHRIG, K.; GUERRA, E.; LARA, J. D.; LENGYEL, L.; LEVENDOVSZKY, T.; PRANGE, U.; VARRÓ, D. & VARRÓ-GYAPAY, S., 2005: Model transformation by graph transformation: A comparative study. Proceedings of Model Transformations in Practice Workshop, 2 -7 October 2005, Models Montego Bay, Jamaica.

TAENTZER, G., 2000: AGG: A Tool Environment for Algebraic Graph Transformation. Applications of Graph Transformations with Industrial Relevance, Lecture Notes in Computer Science, Nagl M., Schürr, A., Münch, M. (eds.), Springer Berlin Heidelberg, 481-488.

YAO, Z. & KOLBE, T.H., 2017: Dynamically Extending Spatial Databases to support CityGML Application Domain Extensions using Graph Transformations. Publikationen der Deutschen Gesellschaft für Photogrammetrie, Fernerkundung und Geoinformation e.V. (DGPF), vol 26, Kersten, T. (ed), 37. Wissenschaftlich-Technischen Jahrestagung der DGPF, 7-10 March 2017 in Würzburg, 316-331.