FAIR Data Based on Extensible Unifying Data Model Development

© Sergey Stupnikov © Leonid Kalinichenko
Institute of Informatics Problems, Federal Research Center "Computer Science and Control" of the Russian Academy of Sciences, Moscow, Russia
sstupnikov@ipiran.ru

Abstract. Nowadays data sources within data infrastructures are quite heterogeneous, they are represented using very different data models. Data models vary from relational one to NoSQL zoo of data models. A prerequisite for (meta)data interoperability, integration and reuse within some data infrastructure is unification of source data models and their data manipulation languages. A unifying data model (called canonical) has to be chosen for the data infrastructure. Every source data model has to be mapped into the canonical model, mapping should be formalized and verified. The paper overviews data unification techniques have been developed during recent years and discusses application of these techniques for data integration within FAIR data infrastructures.

Keywords: FAIR data principles, unifying data model, data integration.

1 Introduction

Data sources nowadays are quite heterogeneous: they are represented using very different data models. Variety of data models includes traditional relational model and its object-relational extensions, array and graph-based models, semantic models like RDF and OWL, models for semi-structured data like NoSQL, XML, JSON and so on. These models provide also very different data manipulation and query languages for accessing and modifying data.

A prerequisite for (meta)data interoperability, integration and reuse within some data infrastructure is unification of source data models and their data manipulation languages. A unifying data model (called canonical) has to be chosen for the data infrastructure. The canonical data model serves as the language for knowledge representation mentioned in FAIR I1 principle ((meta)data use a formal, accessible, shared, and broadly applicable language for knowledge representation) [1][2]. Every source data model has to be mapped into the canonical model. Mapping can be accompanied with the extension of the canonical model if required. A mapping should be formalized and verified: a formal proof that the mapping preserves semantics of data structures and data manipulation operations of the source data model should be provided.

As the core of the canonical model some concrete data model like SQL (conforming to ISO/ANSI SQL standard of 2011 or later) or RDF/RDF Schema with SPARQL query language can be used. To cover features of various source data models the canonical model has to be extensible. Examples of extensions are specific data structures (data types), compound operations or restrictions (dependencies). An extension is constructed for every source data model. Canonical model is formed as the union of the core data model and all extensions.

Data unification techniques were extensively studied at FRC CSC RAS [3]. As the core of the canonical model specific object-frame language with broad range of modeling facilities was used [4]. Approaches for mapping of different classes of source data models were developed: process models [5], semantic models [6][13], array [9] and graph-based [10] models, some other kinds of NoSQL models [8]. Techniques for verification of mappings applying a formal language based on the first order logic and set theory and supported by automatic and interactive provers were developed [11][12].

Techniques mentioned are proposed as a formal basis for (meta)data interoperability, integration and reuse within FAIR data infrastructures. Such infrastructures may combine virtual integration facilities (subject mediators) as well as data warehouses to integrate heterogeneous data sources in an interoperable way.

The rest part of the paper is structured as follows: section 2 overviews data unification techniques that have been developed during recent years and section 3 discusses application of these techniques for data integration within FAIR data infrastructures.

2 Data Model Unification

Various source data models and their data manipulation languages applied within some data infrastructure have to be unified in the frame of some canonical data model.

The main principle of the canonical model design (synthesis) for a data infrastructure is the *extensibility* of the canonical model kernel in heterogeneous environment [3], including various models used for the representation of resources of the data infrastructure. A kernel of the canonical model is fixed (for instance, SQL or RDF). A specific source data model R of the environment is said to be *unified* if it is mapped into the canonical model C [11][12]. This means a creation of such extension E of the canonical model kernel (note that

Proceedings of the XX International Conference "Data Analytics and Management in Data Intensive Domains" (DAMDID/RCDL'2018), Moscow, Russia, October 9-12, 2018

such extension can be empty) and such mapping *M* of a source model into extended canonical one that the source model *refines* the extended canonical one. Model refinement of *C* by *R* means that for any admissible specification (schema) *r* represented in R its image *M(r)* in *C* under the mapping *M* is refined by the specification *r*. Such refining mapping of models means preserving of operations and information of a source model while mapping it into the canonical one. Preserving of operations and information should be formally proven. The canonical model for the environment is synthesized as the union of extensions, constructed for all models of the environment.

The following languages and formal methods are required to support data model mapping:

- a kernel of the canonical data model;
- formal methods allowing to describe data model syntax as well as semantic mappings (transformations) of one model to another;
- formal methods supporting verification of refinement reached by the mapping.

Within studies on data unification techniques at FRC CSC RAS as a *kernel of the canonical data model* the SYNTHESIS language [4] was used. The SYNTHESIS language, as a hybrid semistructured and object-oriented data model, includes the following distinguishing features: facilities for definitions of frames, abstract data types, classes and metaclasses, functions and processes, logical formulae facilities applied for description of constraints, queries, pre- and post-conditions of functions, assertions related to processes. For extension of the canonical model kernel, metaclasses, metaframes, parameterized constructions including assertions and generic data types were applied. Data unification teqhniques developed can be adopted also for other canonical data model kernels like SQL or RDF.

For *data model's semantics formalization and refinement verification* the AMN (Abstract Machine Notation) language [14] was used. The language is supported by technology and tools for proving of refinement (B-technology) [15]. AMN is based on the first order predicate logic and Zermelo-Frenkel set theory and enables to consider state space specifications and behavior specifications in an integrated way. The system state is specified by means of state variables and invariants over these variables, system behavior is specified by means of operations defined as generalized substitutions – a sort of predicate transformers. Refinement of AMN specifications is formalized as a set of refinement proof obligations – theorems of first order logic. Generally speaking in terms of pre- and post-conditions of operations, refinement of AMN specifications means weakening pre-conditions and strengthening post-conditions of corresponding operations included in these specifications. Proof requests are generated automatically and should be proven with the help automatic and interactive theorem prover [15].

For the *formal description of model syntax and transformations* two approaches were developed and prototyped.

The first approach [11][12] is based on the metacompilation languages SDF (Syntax Definition Formalism) and ASF (Algebraic Specification Formalism). For the languages a tool support — Meta-Environment [16] — is provided based on term rewriting techniques. Data model syntax is represented using SDF in a version of extended Backus–Naur form. Data model transformations are defined as ASF language modules constituted by sets of functions. A function defines a transformation of a syntactic element of a source model into a syntactic element of the canonical model. Recursive calls of transformation functions are allowed. According to the ASF-definition the transformation program code (C language) is generated automatically by means of Meta-Environment tools. The transformation obtained is used for mapping of source model specifications into the canonical model specifications.

The second approach [17] is based on the Model-Driven Architecture (MDA) [18] proposed by Object Management Group. Data model abstract syntax neglecting any syntactic sugar is defined using *Ecore* metamodel (an implementation of OMG's Essential Meta-Object Facility) used in Eclipse Modeling Framework [19]. Concrete syntax of data models binding syntactic sugar and abstract syntax was for-malized using EMFText framework [20]. Data model transformations are defined using ATLAS Transformation Language (ATL) [21] combining declarative and imperative features. ATL transformation programs are composed of rules that define how source model elements are matched and navigated to create and initialize the elements of the target models. Type system of the ATL is very close to the type system of the OMG Object Constraint Language.

Using both approaches construction of a mapping of a source data model *R* into the canonical model *C* is divided into the following stages:

- formalization of syntax and semantics the models *R* and *C* (if the latter has not yet been defined);
- definition of reference schemas of the models *R* and *C* (if the latter has not yet been defined);
- integration of reference schemas of the model *R* and *C*;
- creation of a required extension *E* of the canonical model *C*;
- construction of a transformation of the model *R* into the extended canonical model;
- verification of refinement of the extended canonical model by the model *R*.

The Reference schema of a data model is an abstract description containing concepts related to constructs of the model and significant associations among these concepts. Both concepts and associations may be annotated by verbal definitions (looking like entries in an explanatory dictionary). Using MDA terms reference schemas are just metamodels conforming the Ecore metamodel.

Formalization of data model semantics and verification of data model refinement can be performed

in two ways.

In the first way formalization of data model semantics means a construction of transformations of source and canonical data model specifications into AMN-specifications. So for any specification of a source data model the AMN-specification expressing its semantics is generated automatically (for instance, in [11][12] the Ontology Web Language [22] is considered as a source model and its semantics in AMN is illustrated by example). Also, for any specification of the canonical data model the AMN-specification expressing its semantics is generated automatically [23]. After that refinement of a canonical data model specification by a source data model specification is reduced to refinement of their semantic AMN specifications and can be verified by the refinement theorem prover [15]. So verification of model refinement is realized over a set of source model specification samples.

In the second way semantics of a data model (source or canonical) as a whole is expressed by an AMN specification. For instance, in [9] AMN semantics for an array data model is defined, in [10] AMN semantics for a graph data model is defined. AMN semantics for the SYNTHESIS language as the canonical data model was also provided [9][10]. Data structures used in data models were represented by variables in AMN specifications, properties of data structures were represented by AMN invariants, typical operations of data models were represented by AMN operations. Generally refinement of the AMN-specification M_C corresponding to the canonical data model C by the AMN-specification M_R corresponding to a source data model R should be also proved using refinement theorem prover [15].

Partial automation of data unification techniques mentioned above was implemented within Unifying Information Models Constructor (Model Unifier in short) [11][12]. Unifier consists of the following main components:

- tool for the formal description and correctness checking of model syntax and transformations (Meta-Environment, ATL Tools);
- Atelier B [15], supporting AMN and providing facilities for proving of specification refinement;
- model manager.

Meta-Environment, ATL Tools and Atelier B are third-party products. Model manager provides a graphical interface allowing an expert to search for, view and register data models and extensions of the canonical model; to call specific components for generating templates, editing and integration of reference schemas, generating templates for translators of source models into the canonical one, translation of source models specifications into AMN or into canonical specifications, translation of canonical specifications into AMN.

Recent years data unification techniques were applied to wide range of source data models. In [5] a canonical process model has been synthesized for the environment of workflow patterns classified by W. M. P. van der Aalst. Thus the canonical process model possesses a property of completeness with respect to broad class of process models used in various Workflow Management Systems as well as the languages used for process composition of Web services.

In [11][12] the Ontology Web Language was unified with the SYNTHESIS language, in [6] OWL 2 QL was mapped into the SYNTHESIS.

In [7] application of the canonical model synthesis methods for *the value inventive data models* was discussed. The distinguishing feature of these data models is inference of new, unknown values in the process of query answering.

In [8] an approach to mapping of different types of NoSQL models into the object model of the SYNTHESIS language used as unifying data model was considered.

In [9] unification of an array-based data model used in SciDB DBMS was considered, and in [10] unification of an attributed graph data model was considered. For both models verification using AMN specifications is provided.

In [13] issues on unification of RDF with accompanying RDF Schema and SPARQL languages were discussed.

3 FAIR Data Based on Data Model Unification

The following levels of integration (from higher to lower) can be distinguished: data model integration (unification), schema matching and integration (metadata integration) and data integration proper. Usually completion of the integration on a higher level is a prerequisite for integration on a lower level. Obviously the highest level, i.e. data model unification is a prerequisite for (meta)data interoperability, integration and reuse within FAIR data infrastructures and data model unification techniques overviewed in the previous section can be considered as a formal basis for achieving FAIRness of data.

Any level of integration makes data more FAIR: integrated data are much easier to find, access and reuse and also integrated data are more interoperable than heterogeneous data stored in different data sources. The most mature level of integration is achieved within data integration systems like *subject mediators* or *data warehouses*.

Subject mediators implement *virtual integration* with user queries defined in some unified data model. Such queries are to be decomposed into sets of subqueries and these subqueries are to be transferred to heterogeneous data sources. Data sources are connected with a subject mediator via wrappers which transforms queries into source data models and also transforms query answers from source data models into unified mediator data model. Query answers are transferred by wrappers back to the mediator, combined and sent to users. One of the latest trends nowadays is construction of subject mediators over *data lakes* [24].

Data warehouses implement *materialized integration*

with all required data extracted from sources, transformed into unified warehouse data model, and stored into a warehouse.

Any kind of integration system requires unified data model. One of the important issues to be resolved for data integration within FAIR data infrastructures is the choice of the canonical model kernel. Even the choice between SQL and RDF is difficult. On the one hand SQL is supported by industrial standards, methods and technologies evolving for decades. On the other hand, RDF is W3C Recommendation supported by triplestore vendors, is strongly connected with OWL ontological framework, allows flexible evolution of data schema, provides logic inference in a native way that is very important for knowledge bases.

To integrate heterogeneous data sources in an interoperable way FAIR data infrastructures may support both mentioned kinds of data integration systems and also combined data integration systems [25] with data warehouses considered as resources to be integrated within subject mediators. For all kinds of data integration systems the data model unification techniques can provide a formal basis.

Acknowledgments. The research is partially supported by Russian Foundation for Basic Research, project 18-07-01434.

References

[1] Wilkinson, M. D. et al.: The FAIR Guiding Principles for scientific data management and stewardship. Sci. Data 3:160018 (2016). DOI: 10.1038/sdata.2016.18.

[2] Wilkinson, M. D.: Interoperability and FAIRness through a novel combination of Web technologies. PeerJ Preprints (2016). URL: https://doi.org/10.7287/peerj.preprints.2522v1

[3] Kalinichenko, L. A.: Canonical model development techniques aimed at semantic interoperability in the heterogeneous world of information modeling. Knowledge and model driven information systems engineering for networked organizations: Proc. of the CAiSE INTEROP Workshop. -- Riga: Riga Technical University, 2004. -- P. 101-116.

[4] Kalinichenko, L. A., Stupnikov, S. A., Martynov D.O.: SYNTHESIS: a Language for Canonical Information Modeling and Mediator Definition for Problem Solving in Heterogeneous Information Resource Environments. Moscow: IPI RAN, 2007. - 171 p.

[5] Kalinichenko, L.A., Stupnikov, S.A., Zemtsov, N.A.: Extensible Canonical Process Model Synthesis Applying Formal Interpretation. In: Avances in Databases and Information Systems: Proceedings of the East European Conference. LNCS 3631, pp. 183-198. Berlin-Heidelberg: Springer-Verlag (2005)

[6] Kalinichenko, L.A., Stupnikov, S.A.: OWL as Yet Another Data Model to be Integrated. In: Advances in Databases and Information Systems: Proc. II of the 15th East-European Conference, pp. 178-189. Vienna: Austrian Computer Society (2011)

[7] Kalinichenko L.A., Stupnikov S.A.: Synthesis of the Canonical Models for Database Integration Preserving Semantics of the Value Inventive Data Models. Advances in Databases and Information Systems: Proc. of the 16th East European Conference, LNCS 7503, pp. 223-239. Berlin-Heidelberg: Springer-Verlag (2012)

[8] Skvortsov N. A.: Mapping of NoSQL data models to object specifications. Proc. of the 14th Russian Conference on Digital Libraries RCDL'2012. CEUR Workshop Proceedings 934:53-62 (2012)

[9] Stupnikov, S. A.: Unification of an Array Data Model for the Integration of Heterogeneous Information Resources. In: Proc. of the 14th Russian Conference on Digital Libraries RCDL'2012. CEUR Workshop Proceedings, Vol. 934, pp. 42-52 (2012)

[10] Stupnikov, S. A.: Mapping of a Graph Data Model into an Object-Frame Canonical Information Model for the Development of Heterogeneous Information Resources Integration Systems. In: Proc. of the 15th Russian Conference on Digital Libraries RCDL'2013. CEUR Workshop Proceedings 1108:85-94 (2013)

[11] Zakharov, V. N., Kalinichenko, L. A., Sokolov, I. A., Stupnikov, S. A.: Development of Canonical Information Models for Integrated Information Systems. Informatics and Applications, 1(2):15-38 (2007)

[12] Kalinichenko, L.A., Stupnikov, S.A.: Constructing of Mappings of Heterogeneous Information Models into the Canonical Models of Integrated Information Systems. In: Advances in Databases and Information Systems: Proc. of the 12th East-European Conference, pp. 106-122. Pori: Tampere University of Technology (2008)

[13] Skvortsov N. A.: Mapping of RDF Data Model into the Canonical Model of Subject Mediators. Proc. of the 15th Russian Conference on Digital Libraries RCDL'2013. CEUR Workshop Proceedings 1108:95-101 (2013)

[14] Abrial, J.-R.: The B-Book: Assigning Programs to Meanings. Cambridge: Cambridge University Press (1996)

[15] Atelier B, the industrial tool to efficiently deploy the B Method. http://www.atelierb.eu/

[16] Van den Brand M. G. J. et al.: The ASF+SDF meta-environment: a component based language development environment // Compiler Construction 2001 / Ed. By R. Wilhelm, pp. 365–370. Springer (2001)

[17] Stupnikov, S.A., Kalinichenko, L.A.: Methods for Semi-automatic Construction of Information

Models Transformations. Proc. of the 13th East-European Conference Advances in Databases and Information Systems, workshop Model – Driven Architecture: Foundations, Practices and Implications (MDA), pp. 432-440. Riga: Riga Technical University (2009)

[18] Object Management Group Model Driven Architecture (MDA). MDA Guide rev. 2.0. OMG Document ormsc/2014-06-01 (2014)

[19] Steinberg, D., Budinsky, F., Paternostro, M., Merks, E.: EMF: Eclipse Modeling Framework, 2nd Edition. Addison-Wesley Professional (2008)

[20] EMFText Concrete Syntax Mapper. https://github.com/DevBoost/EMFText

[21] ATL - a model transformation technology. https://eclipse.org/atl/

[22] OWL Web Ontology Language Reference. W3C Recommendation. http://www.w3.org/TR/owl-ref/ (2004)

[23] Stupnikov, S. A.: A semantic transformation of the canonical information model into a formal specification langage for the refinement verification. Proc. of the 12th Russian Conference on Digital Libraries RCDL'2010, pp. 383-391. Kazan: Kazan Federal University (2010)

[24] Hai, R., Quix, C., Zhou, C.: Query Rewriting for Heterogeneous Data Lakes. In: Benczúr A., Thalheim B., Horváth T. (eds) Advances in Databases and Information Systems. ADBIS 2018. LNCS 11019:35-49. Springer (2018)

[25] Stupnikov, S.A., Vovchenko, A.E.: Combined Virtual and Materialized Environment for Integration of Large Heterogeneous Data Collections. In: 16th Russian Conference on Digital Libraries RCDL 2014 Proceedings. CEUR Workshop Proceedings 1297:201-210 (2014)