On Metadata Support for Integrating Evolving Heterogeneous Data Sources

Darja Solodovnikova(✉), Laila Niedrite, and Aivars Niedritis

Faculty of Computing, University of Latvia, Riga 1050, Latvia
{darja.solodovnikova, laila.niedrite, aivars.niedritis}@lu.lv

Abstract. With the emergence of big data technologies, the problem of structure evolution of integrated heterogeneous data sources has become extremely topical due to dynamic and diverse nature of big data. To solve the big data evolution problem, we propose an architecture that allows to store and process structured and unstructured data at different levels of detail, analyze them using OLAP capabilities and semi-automatically manage changes in requirements and data expansion. In this paper, we concentrate on the metadata essential for the operation of the proposed architecture. We propose a metadata model to describe schemata and supplementary properties of data sets extracted from sources and transformed to obtain integrated data for the analysis in a flexible way. Furthermore, the unique feature of the proposed model is that it allows to keep track of all changes that occur in the system.

Keywords: Big data · Data warehouse · Evolution · Metadata

1 Introduction

In recent years, the concept of big data has been increasingly attracting attention and interest from researchers and businesses around the world. There are different approaches to the processing and analysis of big data and one of them is to use a data warehouse and OLAP techniques. Various solutions based on Hadoop and similar frameworks allow to implement a data warehouse also to support analysis of big data.

In the context of relational data warehouses, evolution problems have been known for a long time. Evolution can be caused by changes in data sources of the data warehouse, and changes in requirements when additional information becomes necessary for decision making. In the big data world, evolution problems have become even more topical [1, 2] as big data are more dynamic, diverse and can be generated at higher speeds, but the solution to the big data evolution problems is a more challenging task for several reasons. First, there is currently no standard data warehouse architecture that should be used to support big data analysis. Second, in the context of big data, data sources are very often unstructured or semi-structured, and tracking and processing changes in such data sources is a complex task. Finally, in big data systems, the data could be generated in real time, which means that the changes could occur in real time, so they should be processed in real time too.

T. Welzer et al. (Eds.): ADBIS 2019, CCIS 1064, pp. 378–390, 2019.
https://doi.org/10.1007/978-3-030-30278-8_38

In this paper, we propose a data warehouse architecture over big data that can adapt to evolving user requirements and changes in data sources. We present the model that allows to store the metadata describing schemata of involved data sets and their changes that are essential for the operation of the proposed architecture.

The paper is structured as follows. In Sect. 2 the related work is discussed. Section 3 introduces a data warehouse architecture over big data. The main contribution of this paper is presented in Sect. 4, where the metadata model is described. Section 5 discusses evolution support in our proposal. Section 6 is devoted to the description of the case study system. We conclude with directions for future work in Sect. 7.

2 Related Work

The problem of representing metadata of heterogeneous data sources has been studied extensively in the context of data lakes, where structure and other characteristics of data are not defined a priori and are discovered during the data analysis or processing. The paper [3] describes challenges that developers of data lakes are often facing. One of such challenges is obtaining metadata about data extracted from heterogeneous sources. Along with other challenges, the authors mention data evolution when data representation changes over time or the same data are provided in different formats.

The authors of the paper [4] propose a metadata classification to support a data mining process in big data. Although the evolution is not mentioned in the classification and some types of metadata described are specific to data mining tasks, the proposed classification may be used complementary to our metadata model.

The paper [5] proposes a formal metadata model for data lakes. The model is represented as a graph based on the metadata classification proposed by [6]. The model is applied in two algorithms used to obtain a structure from unstructured data lake sources and to integrate sources of different formats.

A metadata model that represents 3 types of metadata describing data lake sources: structure metadata, metadata properties, and semantic metadata is proposed in the paper [7]. The structure metadata describe the schemata of data sources. The authors distinguish between 2 types of structures: matrices and trees. Metadata properties are descriptive information about the contents of data source files, such as file name, author, date modified, including information from the file itself, such as date, information about the experiment for which the source file contains data. Semantic metadata contain annotations of any source elements that could be expressed as URIs that point to a concept in an ontology. The authors use data unit templates to obtain metadata properties that may have different structures. We adapted the metadata model proposed in [7] for our big data warehouse architecture and extended it with metadata necessary for evolution support.

3 Big Data Warehouse Architecture

To solve the topical problem of big data evolution in the data warehousing context, we propose to design a big data management system according to the big data warehouse architecture shown in Fig. 1. The detailed description of the architecture is given in [8].

The architecture consists of a data processing pipeline (*Data Highway* in the figure). The idea and the concept of the data highway was first presented in [9]. Data in the system are gathered from various heterogeneous data sources and loaded in their original format at the first level of the data highway which may be considered as a data lake. Each next level data are obtained from the previous level data by ELT processes by means of applying transformation operations, adding structure to unstructured and semi-structured data, integrating and aggregating data. The number and contents of the levels of the data highway depend on the system requirements. The final level of the data highway is a data warehouse which stores structured aggregated information ready for OLAP operations. Since the volume of data stored in the data warehouse may be too large to provide a reasonable performance of data analysis queries, the architecture may be augmented by the cube engine component, which precomputes various dimensional combinations and aggregated measures for them.

To support big data evolution, we expanded the big data warehouse architecture with an adaptation component that is responsible for handling changes in data sources and levels of the data highway. The main idea of this component is to generate several potential adaptation options to schema of the data highway levels affected by each change in a data source of other level of the highway and to allow a developer to choose the most appropriate option that must be implemented.

One of the central components of the architecture is the metastore that incorporates six types of interconnected metadata necessary for the operation of various components of the architecture. In this paper, we concentrate on the metadata support to maintain information about big data evolution. Three types of such metadata are highlighted with a dark font color in the figure. Schematic metadata describe schemata of data sets stored in the different levels of the highway. Mapping metadata define the logic of ELT processes. Information about changes in data sources and data highway levels is accumulated in the evolution metadata. Such information may be obtained from wrappers or during the execution of ELT processes.

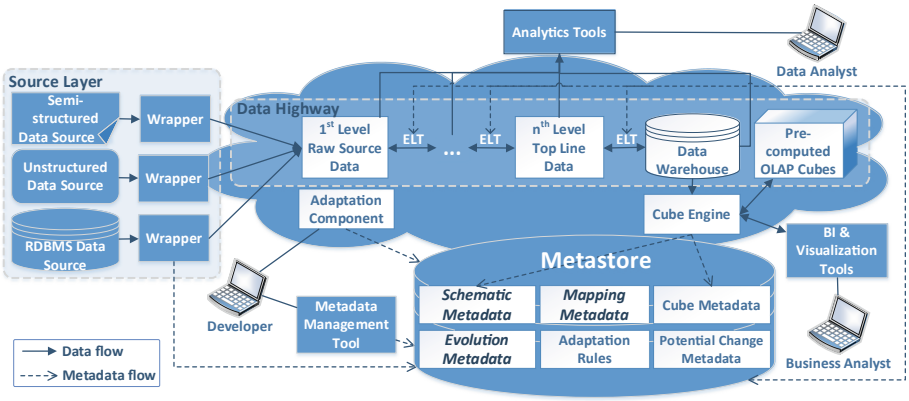

Fig. 1. Big data warehouse architecture

There are also other three types of metadata used for other purposes. Cube metadata describe schemata of precomputed cubes and are leveraged not only during the cube computation process but also for execution of queries. Adaptation rules specify adaptation options that must be implemented for different types of changes. Finally, potential change metadata accumulate proposed changes in the data warehouse schema.

4 Metadata Model

To accumulate metadata about the structure of data sources as well as data sets included at various levels of the data highway and to maintain information about changes that occur in them, we propose the use of metadata that is designed according to the conceptual model presented in Fig. 2.

4.1 Schematic Metadata and Mappings

In this section, we concentrate on the elements of the model that are used to describe schemata of data sources and data highway levels. A class *Data Set* is used to represent a collection of *Data Items* that are individual pieces of data. The *Data Set* class is split into three subclasses according to the type and format. *Structured Data Set* represents a relational database table where data items correspond to table columns. *Semi-structured Data Set* reflects files where data items are organized into a schema that is not predefined. The attribute *Type* of a data item incorporated into a such data source indicates the position of it in the schema. For example, XML documents are composed of *Elements* and their *Attributes*, JSON documents may contain *Objects* and *Arrays*. Semi-structured proprietary file formats of software tools or hardware devices may be also described by the *Semi-structured Data Set* class. *Unstructured Data Sets* include data that do not have any organization or acknowledged schema, such as text files, images, other multimedia content, etc. Usually, an unstructured data set is represented by a single data item. However, pieces of supplementary information like keywords or tags that are used to get an idea of the data set content may be available. They are represented as additional data items associated with the corresponding data set in the model.

A data set may be obtained from a *Data Source* or it may be a part of a *Data Highway Level*. We also maintain the information about the rate at which data in the data set are collected or updated by assigning one of the velocity types and frequency attribute of the *Data Set* class. If a data set is a part of a multidimensional model of a data warehouse, its role (dimension or fact table) is assigned to the attribute *Role* and data items contained in such a data set get roles of either dimension attribute or fact table measure.

There usually exist relationships between data items in the same data set or across different data sets determined by the format of these data sets. For example, elements in XML files are composed of other subelements and attributes, objects in JSON files may contain other objects as property values, foreign keys relate columns of relational tables, predicates relate subjects and objects in RDF. There may exist a link between an

unstructured data set and structured/semi-structured data. We modelled such relationships by an association class *Relationship* that connects child and parent data items and assigns the corresponding relationship type. The relationship type *Equality* is assigned if two items of different data sets contain the same data. Equality relationships help integrate separate data sets.

Our proposed architecture of the big data system implies that data sets contained in various data highway levels are obtained either from data sources in their original format or from data sets at other data highway levels by performing transformation, aggregation and integrating related data items. To maintain metadata about provenance of data sets within the data highway and make it possible to follow their lineage, an association class *Mapping* was introduced in the model. A mapping defines a way how a target data item is derived from origin data items by a transformation indicated in the attribute *Operation* of the class *Mapping*. Not only our model allows to maintain provenance metadata of data items in the highway calculated directly from source data, but it also supports such cases when previously processed data sets of the highway are further transformed to obtain new data sets at subsequent data highway levels.

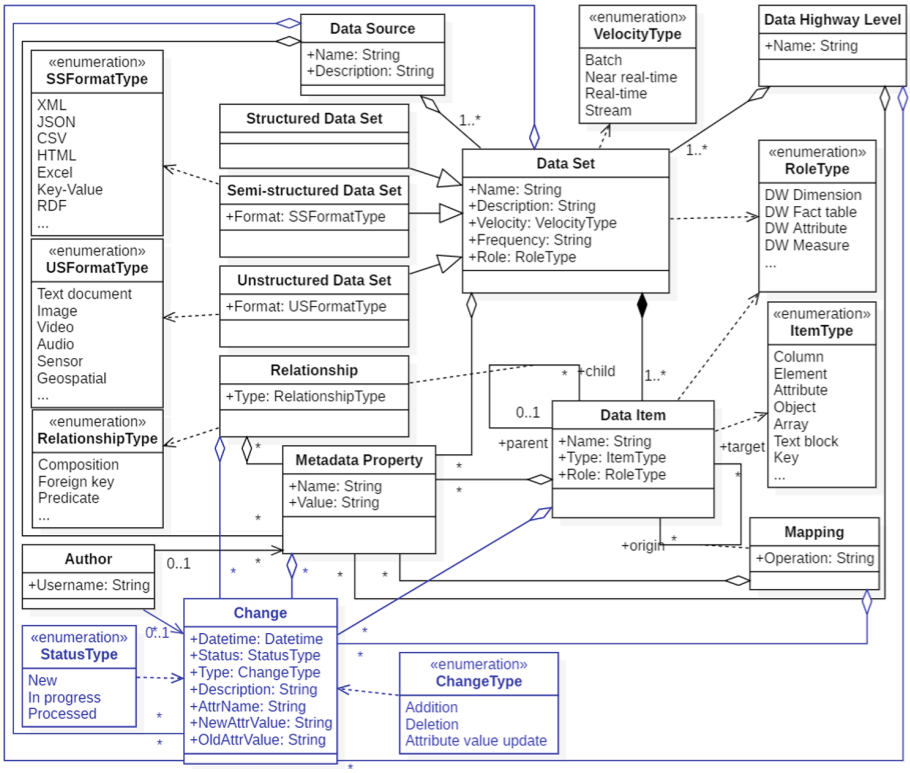

Fig. 2. Conceptual metadata model for evolution support

4.2 Metadata Properties

The classes and associations of the model described above reflect mainly the schematic metadata of the system whereas other characteristics of the data should also be represented in the model. For this purpose, we included a class *Metadata Property* that is used to store various characteristics of different elements of the model. Each metadata property is represented by a name:value pair to allow for some flexibility as metadata properties of different classes may vary considerably. Examples of metadata properties include file or table name, dates created and updated, location, file size, character set, version, check constraints for data sets; type, length, precision, scale, nullable, range for data items; mechanism used to retrieve data from a data source (for instance, API request).

Certain metadata properties may be obtained in a machine-processible form automatically, but others may be only entered manually. Furthermore, an experienced user may discover some new metadata properties of data sets obtained from a data source during data analysis or transformation and of processed data sets that might be valuable for other users of the system. In such a case, the user may augment the metadata by the discovered properties. Such conversational [3] metadata are associated with a user who recorded the property represented by the class *Author* in the model.

4.3 Evolution Metadata

In this section, we discuss classes and associations introduced in the model to store information about the evolution of both the schemata of data sources and data highway data sets and their supplementary characteristics represented as metadata properties. Examples of changes that are supported by the model are given in more detail in the next section.

Evolution is reflected in the model by a class *Change* that is connected by associations with other classes. These associations determine the element of the model that was affected by each change. In the metadata, we store the date and time when the change took place, *Type* of the change, *Status* that determines whether that change is new, processed (propagated in the system) or in progress (being currently processed). In case if evolution was caused by a change in a value of any attribute of a model element including metadata property, we record the name of the affected attribute as an attribute *AttrName* of the class *Change* and both the value before the change (attribute *OldAttrValue*) and after it (attribute *NewAttrValue*). If a change was performed manually by a known user, the corresponding *Author* is associated with such change.

5 Evolution Support

In the proposed big data architecture, data from heterogeneous data sources are integrated and transformed gradually to obtain a data warehouse level of the data highway. Various kinds of changes to the data employed in each step of this process must be recorded in the metadata model. The list of atomic changes classified according to the part of the metadata model they affect is given in Table 1. For each change, the table

describes classes that are connected with the instance of the class *Change* in the model by an association, the key attributes of the class *Change* with their value and additional metadata that must be recorded in the model.

Table 1. Supported atomic changes

Change	Associated classes	Key attribute values	Additional metadata
Schematic changes			
Addition of a data source	Data Source	Type: Addition	For each data set included in the new data source, the corresponding schematic metadata and metadata properties must be added
Unavailable data source	Data Source	Type: Deletion	
Addition of a data highway level	Data Highway Level	Type: Addition	For each data set included in the new data highway level, the corresponding schematic metadata and metadata properties must be added along with mappings that define the origin of data
Deletion of a data highway level	Data Highway Level	Type: Deletion	
Addition of a data set	Data Set	Type: Addition	For each data item included in the new data set, the corresponding schematic metadata and metadata properties must be added
Unavailable data set	Data Set	Type: Addition	
Change of data set format	Data Set Data Item Relationship	Type: Deletion Type: Addition	Such change must be recorded as two changes: deletion and addition of a data set. After that, new relationships with the type *Equality* between the corresponding data items of the removed data set and the new data set must be added to the metadata
Renamed data set	Data Set	Type: Attribute value update AttrName: Name	The attributes *OldAttrValue* and *NewAttrValue* of the class *Change* are filled with the previous and updated name of the data set
Addition of a data item	Data Item	Type: Addition	A change is associated with the new instance of the class *Data Item*

(*continued*)

Table 1. (*continued*)

Change	Associated classes	Key attribute values	Additional metadata
Renamed data item	Data Item	Type: Attribute value update AttrName: Name	The attributes *OldAttrValue* and *NewAttrValue* of the class *Change* are filled with the previous and updated name of the data item
Change of a data item type	Data Item	Type: Attribute value update AttrName: Type	The attributes *OldAttrValue* and *NewAttrValue* of the class *Change* are filled with the previous and updated type of the data item
Deletion of a data item from a data set	Data Item	Type: Deletion	
Addition of a relationship	Relationship	Type: Addition	A new instance of the association class *Relationship* connects two related data items
Deletion of a relationship	Relationship	Type: Deletion	
Addition of a new mapping	Mapping	Type: Addition	A new instance of the class *Mapping* that is being added within the change and associated with target and origin data items defines the way the target data item is obtained in the attribute *Operation*
Deletion of a mapping	Mapping	Type: Deletion	
Changes in metadata			
Addition of a metadata property	Metadata Property	Type: Addition	
Deletion of a metadata property	Metadata Property	Type: Deletion	
Update of an attribute value	A class containing an updated attribute	Type: Attribute value update	The attribute values before and after the update are recorded in the attributes *OldAttrValue* and *NewAttrValue* of the class *Change*, which is associated with the model class instance affected by the change

When elements of the model are deleted, they remain in the model, but information about deletion is maintained as instances of the class *Change* with the type *Deletion*. In case of a deletion of a relationship, such change may affect integration of heterogeneous data sets. This aspect must be considered during change propagation. If a data item or a data set is deleted from the system individually or by deletion of the data source or data highway level containing it, such change affects data sets that were populated with data from a deleted model element (determined by mappings) before the change. If there are any alternative ways how affected data sets may be obtained from other data sets described in metadata, new mappings should be defined by a change "Addition of a new mapping". When a mapping is deleted, it should be replaced by a new mapping to provide successful execution of ELT processes and maintain data lineage. If a replacement is not possible, a change of deletion of a mapping must be registered.

In case of addition of a new element to the system, a new instance of the corresponding class described in the column Associated Class of Table 1 is created in the metadata. If an element containing child elements (such as a data set, data source or data highway level) is added to the system, only one instance of the class *Change* with the type *Addition* is created and associated with the new class containing other child elements.

When a new change is discovered automatically by a source wrapper or during the execution of ELT processes or observed or generated by a user, the corresponding metadata about it must be entered in the metadata repository as an instance of the class *Change* with the status *New*. Thereafter, the adaptation component of the big data warehouse architecture must inject the change and generate possible adaptation scenarios. At this stage the status of the change must be updated to *In progress*. Finally, when a developer or administrator of the system accepts any of the adaptation scenarios proposed by the adaptation component and change propagation is complete, the status of the change must be adjusted to *Processed*.

6 Case Study

As a proof of concept, we have applied our proposed approach to the publications big data system with the purpose to validate the metadata model. The goal of the system is to integrate data about publications authored by employees and students of the University of Latvia from multiple heterogeneous sources and to provide these data for analysis in a data warehouse. The architecture of the developed system that includes data sources and data highway levels is shown in Fig. 3.

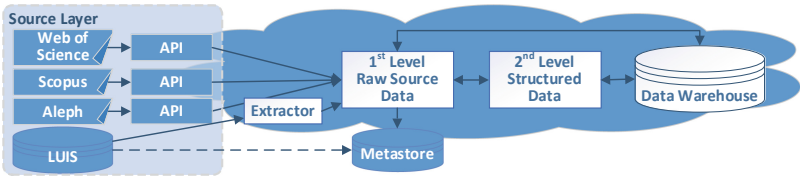

Fig. 3. Architecture of the publication data warehouse

6.1 Data Sources

We integrated data from four structured and semi-structured data sources. The sources contain complementary information about publications, therefore integration of all sources is necessary to obtain the unified view of all publications authored by employees and students of the university.

LUIS is the university data management system implemented in a relational database Oracle. Data about employees and students of the university as well as data about publications entered by LUIS users in the database are gathered and loaded into the first level of the data highway. Aleph is the library data management system. Bibliographical data about publications are obtained from it using API in XML format. All data sets gathered from Aleph have the same structure. Scopus is an indexation system and data from this source are obtained using API in XML format. We used four data set types from this source: publication bibliographical data, author data, affiliation data, and data about publication citation metrics. Web of Science (WOS) is another indexation system. We use API to gather data from WOS in XML format. One type of a data set is available to the university and it contains information about publications, which also includes limited author data (names, surnames and ResearcherID field).

6.2 Data Highway

There are three levels in the data highway of the case study system. Data from the sources are ingested and loaded into the first raw data level. We use Scoop to extract and transfer data from the relational database LUIS into Hive tables. Data from other sources are first pulled from the API and saved in Linux file system and then data are transferred into HDFS using a script with HDFS commands.

At the 2nd level of the data highway, XML files are transformed into structured Hive tables. Data at this level are not yet fully integrated. It is not necessary to transform structured data from the relational database, therefore such data are not included at the 2nd level of the data highway.

Finally, the 3rd level of the highway is a data warehouse implemented in Hive. Data from external data sources that are partially transformed at the 2nd level are integrated with LUIS data directly from the raw source data level.

For the sake of simplicity, we will discuss the fragment of the data warehouse schema represented as a dimensional fact model in Fig. 4. Aggregated data from all four sources are used to populate this star schema. The measures in the fact table contain summarized values classified across four dimensions. Category dimension with the same attribute is used to classify publications by types, such as journal article, book, article in a conference proceedings, etc. Time is a traditional dimension in data warehouses with the quarter of a year granularity in our case study. Faculty dimension contains a hierarchy from a department within a faculty to represent the organizational structure used at the university. Journal metrics dimension stores attributes that characterize a journal or conference proceedings where publications were published.

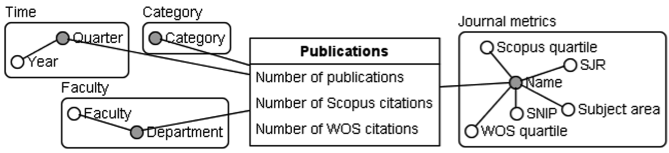

Fig. 4. Dimensional fact model of the publications data warehouse

6.3 Metadata

We implemented the metastore as a relational database Oracle in accordance with the proposed metadata model. Since we have access directly to LUIS data source, we embedded a procedure directly into the data source system that collects metadata about the structure and other metadata properties of tables used to populate the publications system. After source data in XML format are loaded from other data sources, we run a procedure that collects metadata about the structure of XML documents and other metadata properties. Both procedures are able to compare discovered metadata with the information in the metastore and register in the evolution metadata any detected differences.

We defined mappings between data items of the 3 data highway levels. Examples of mappings are given in Table 2. For the sake of clarity, we prefixed names of data items with the corresponding data sources or data highway levels and names of data sets. Question marks denote placeholders for origin data items in the order as they appear in the column *Origin Data Items*.

Table 2. Example mappings of the publication system

Target data item	Origin data items	Mapping operation	Description
Level3.Faculty. Department	Level1.LUIS_person. Affiliation	?	The department of an author of the publication is determined from the column Affiliation of the table LUIS_person
Level3. Publications. Number of Scopus citations	Level2.Scopus_publ. citedby_count	SUM(?)	The number of Scopus citations is calculated as a sum of citations of individual publications extracted from Scopus data source
Level2.Aleph. doc_number	Level1.Aleph. doc_number	?	The item doc_number is obtained from the same item from the 1st level of the data highway
Level3. Publications. Number of publications	Level1.LUIS_publ. Publ_ID Level2.Aleph. doc_number Level2.Scopus_publ. identifier Level2.WOS.uid	COUNT (?) + COUNT (?) + COUNT (?) + COUNT(?)	The number of publications is calculated as a sum of counts of identifiers from all sources. In this case relationships are defined to avoid counting the same publication from multiple sources several times

6.4 Evolution

During operation of the publication system, several changes were discovered during the comparison of the metadata present in the metastore and structure and properties of the data incoming from the data sources. The list of changes with the corresponding metadata additions is presented in Table 3.

Table 3. Changes in the publication system

Change	Change processing
Addition of a data item *citeScoreYearInfoList* to the data set *Scopus_metrics*	A new data item (XML element) was composed of several subelements that were also absent in the previously gathered data sets, however, only one change was created in the metadata for this case and assigned to the uppermost ancestor of the subelements
Deletion of a data item *IPP* from a data set *Scopus_metrics*	This changed affected the data loading process. Since it was not possible to substitute the deleted data item by another data item present in any of the data sources, a deletion of a mapping was recorded in the metadata
Addition of a data source *DSpace*	According to new analysis requirements, the system must have been supplemented by data that contain pre-prints or published full texts of papers. A new data source contained unstructured data (full text files) and metadata associated with them as tags. Files and tags were added as individual data items. The change (addition of a new data source) was associated only with the new data source
Update of a metadata property *API request* value of a data set *Scopus_metrics*	This change was discovered during the execution of the script that extracts data from the API. It had to be processed manually since the new API request could not be discovered automatically. The change was processed as a change in the value of the attribute *Value* of the class *Metadata Property*

7 Conclusions and Future Work

In this paper, we presented a system architecture to address the topical problem of evolution in heterogeneous big data sources. The architecture consists of data highway levels that contain data extracted from sources in their original format and gradually transformed and integrated to obtain a structured data warehouse data. The main contribution of this paper is the flexible metadata model that describes the structure and other characteristics of such data sources and data highway levels as well as changes in the structure not only of data sources, but also of data highway levels for the purpose to

support data expansion and evolution of user requirements for data. We defined a set of atomic changes and their representations in the model and applied the model in the case study system.

The directions for future work include full implementation of the proposed architecture. We will develop algorithms for automatic and semi-automatic change treatment. We also plan technology improvements to our case study system by incorporating Presto or Spark SQL for data processing to improve performance.

Acknowledgments. This work has been supported by the European Regional Development Fund (ERDF) project No. 1.1.1.2./VIAA/1/16/057.

References

1. Ceravolo, P., et al.: Big data semantics. J. Data Semant. **7**(2), 65–85 (2018)
2. Kaisler, S., Armour, F., Espinosa, J.A., Money, W.: Big data: issues and challenges moving forward. In: Proceedings of 46th Hawaii International Conference on System Sciences, pp. 995–1004 (2013)
3. Terrizzano, I.G., Schwarz, P.M., Roth, M., Colino, J.E.: Data wrangling: the challenging Yourney from the wild to the lake. In: Proceedings of 7th Biennial Conference on Innovative Data Systems Research (CIDR 2015), Asilomar, CA, USA (2015)
4. Bilalli, B., Abelló, A., Aluja, T., Wrembel, R.: Towards intelligent data analysis: the metadata challenge. In: Proceedings of the International Conference on Internet of Things and Big Data - Volume 1, IoTBD, pp. 331–338, Rome, Italy (2016)
5. Diamantini, C., Lo Giudice, P., Musarella, L., Potena, D., Storti, E., Ursino, D.: A new metadata model to uniformly handle heterogeneous data lake sources. In: New Trends in Databases and Information Systems, ADBIS 2018 Short Papers and Workshops, Budapest, Hungary, pp. 165–177 (2018)
6. Oram, A.: Managing the Data Lake. O'Reilly, Sebastopol (2015)
7. Quix, C., Hai, R., Vatov, I.: Metadata extraction and management in data lakes with GEMMS. Complex Syst. Inform. Model. Q. **9**, 67–83 (2016)
8. Solodovnikova, D., Niedrite, L.: Towards a data warehouse architecture for managing big data evolution. In: Proceedings of the 7th International Conference on Data Science, Technology and Applications (DATA 2018), Porto, Portugal, pp. 63–70 (2018)
9. Kimball, R., Ross, M.: The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling, 3rd edn. Wiley, Hoboken (2013)