Monolith 2.0: the Semantic OBDM Knowledge Graph Platform

Lorenzo Lepore[a,b], Giacomo Ronconi[a,b], Marco Ruzzi[b], Valerio Santarelli[b]

(a) Sapienza Università di Roma
⟨lastname⟩@diag.uniroma1.it

(b) OBDA Systems
⟨lastname⟩@obdasystems.com

Abstract. In this demo we showcase the innovative features of MONOLITH 2.0, the Semantic Knowledge Graph platform, which gives access to the Ontology-based Data Management (OBDM) capabilities of the MASTRO 2.0 ontology reasoner, in particular to its enhanced SPARQL query answering and data quality checking features, provides a user-friendly environment to build SQL-based mappings of structured enterprise data to the ontology, and allows to build Virtual Knowledge Graphs from this data, or from external RDF or tabular datasets.

1 Introduction

Ontology Based Data Management (OBDM) [11] is an innovative approach to data access, integration, and governance. The fundamental idea behind OBDM is to adopt an ontology as a unified, semantically rich, and comprehensible model of enterprise data, and to enable access through the ontology to such data by means of a mapping layer, which defines the semantic correspondences between the ontology entities and the source data. OBDM has in recent years consolidated its position in the academic and enterprise world as an effective means to manage enterprise data [1, 6, 9].

Knowledge Graphs (KGs) [7] are data models that use a graph structure, built on nodes and edges, to represent enterprise data, and to highlight the relationships between the data entities. When adopting the terms of a domain ontology, the KG acquires semantics and meaning which enrich the representation of these relationships. The graph model is extremely flexible, and, like the OBDM approach, is expandable, can be applied to any real-world use case, and allows to abstract from the organization of the data in the underlying data stores.

In this demo[1] we present MONOLITH 2.0, the latest major release of the MONOLITH OBDM platform [12], and in particular we will focus on its newest features and innovations. In particular, we will take a look under the hood at the fully revamped query answering motor of the MASTRO 2.0 system, and we will introduce the enhanced SPARQL query interface and the new data quality checking environment. These features go together with MONOLITH 2.0's interactive visual inspection environment for GRAPHOL [10] ontologies and with its fully SQL-based editing environment for the

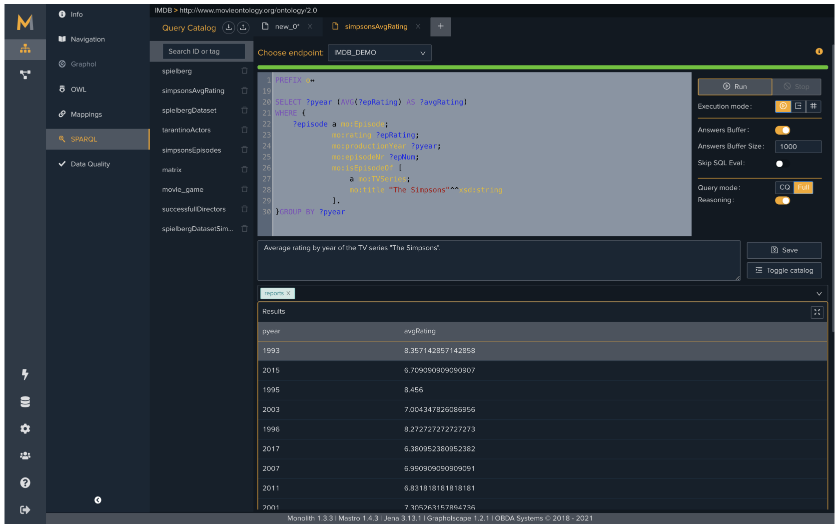

Fig. 1. The ontology SPARQL endpoint in MONOLITH 2.0

ontology mappings to provide a full-fledged environment for all OBDM-related data management services. Furthermore, MONOLITH 2.0's dedicated KG section has been upgraded to provide all the necessary functionalities to build RDF datasets from enterprise structured data through MASTRO 2.0, and to combine these datasets with external ones, which can be natively in RDF form, or can be tabular datasets, which MONOLITH 2.0 allows to transform into RDF statements through a SPARQL interface.

MONOLITH 2.0 and MASTRO 2.0 are developed by OBDA Systems, a start-up of Sapienza University of Rome.

2 Monolith 2.0's main new features and improvements

In this section we present the main novel features and improvements of MONOLITH 2.0, starting with the new version of its underlying query answering motor, MASTRO 2.0. In a nutshell, MASTRO supports data access through OWL 2, specifically the OWL 2 QL profile, ontologies by leveraging a mapping layer which is constituted by a set of views over the database and mapping assertions [11] which associate ontology elements with such views. Crucially, while also supporting the W3C standard R2RML[4] mapping syntax, MASTRO 2.0 provides a fully SQL-based proprietary mapping syntax, which greatly reduces the learning curve of mapping design in real-world scenarios, where IT experts are typically fluent in SQL, but not in R2RML. OBDM services such as query answering and data quality checking are carried out in MASTRO through a very efficient technique that reduces them, via query rewriting [5], to standard SQL query evaluation. In essence, the SPARQL user query is reformulated with respect to first the ontology and then the mappings, in such a way that a new query, which encodes this reasoning and that can be directly executed on the relational data sources, is produced.

The development of MASTRO 2.0 has been primarily focused on extending the fragment of SPARQL 1.1 [8] that was supported by earlier versions of MASTRO (which was limited to the conjunctive query fragment of SPARQL), specifically aiming for full support of SPARQL 1.1's graph patterns, aggregates, functions, and solution modifiers[2]. This was achieved by restructuring the core SPARQL-to-SQL translation process, and enhancing MASTRO 2.0's query-time optimization features to maintain very good performances with respect to MASTRO 2.0's reasoning time. Intuitively, Mastro's query answering algorithm now features a first step in which the conjunctive query fragments of the SPARQL query (called *cores*) are identified and queued for rewriting using Mastro's two step query reformulation algorithm (the ontology rewriting and then the mapping rewriting). The final SQL code is then obtained by compiling all parts of the query, with the aid of an intermediate relational algebra-based query language. In general, Mastro's query answering performances are almost entirely dependent on the underlying database, and on the complexity of the provided SPARQL query: in other words, Mastro's SPARQL-to-SQL query processing times are almost irrelevant with respect to total query answering times when compared to the DBMS query evaluation time. Moreover, parallelizing the query rewriting of each single SPARQL core allows Mastro 2.0 to process more complex SPARQL queries in almost identical times as more simple SPARQL conjunctive queries.

Furthermore, we have widened the scope of MASTRO 2.0's database management system connectors, to include not only the market leaders among traditional RDBMSs, but also connections to data stored in Apache Hadoop format through Apache Impala, and to data virtualization and federation systems, such as Denodo. For each supported DBMS, Mastro features a specifically tailored SQL dialect, in order to produce the final SQL query in compliance with the chosen system.

MONOLITH 2.0 features a new version of the SPARQL endpoint, where users can query the ontology through MASTRO 2.0. The SPARQL endpoint now allows to choose between three query execution modes: *standard* mode, which outputs the query results to the interface, and which is coupled with an answer buffer to limit the number of produced results; a *file streaming* mode which streams the results directly to a chosen output file; a *result count* mode which runs the query in background and produces the result count. These execution modes are designed to handle scenarios where the user wants to inspect a portion of the query results directly in MONOLITH 2.0's interface, or in which large volumes of data are being extracted, and streamed directly to a physical file. MONOLITH 2.0 provides different export options for both standard and file streaming execution modes, including CSV, JSON, XML, RDF, and PowerBI (.pbids) formats. The *SPARQL Query Catalog* has also been enriched with a *query tagging* system in which queries in the catalog can by easily classified and then searched for through user-defined tags.

Finally, MONOLITH 2.0 introduces a new *Data Quality* section, where users leverage MASTRO 2.0's ability to automatically identify and extract data quality rules (or *data integrity constraints*) from the OWL 2 ontology, and translate them into SPARQL queries. MASTRO 2.0 current supports the following constraints: class disjointness con-

[2] `http://www.monolith.obdasystems.com/monolith-user-manual/#Mastro-SPARQL` for a complete list of supported SPARQL 1.1 operators

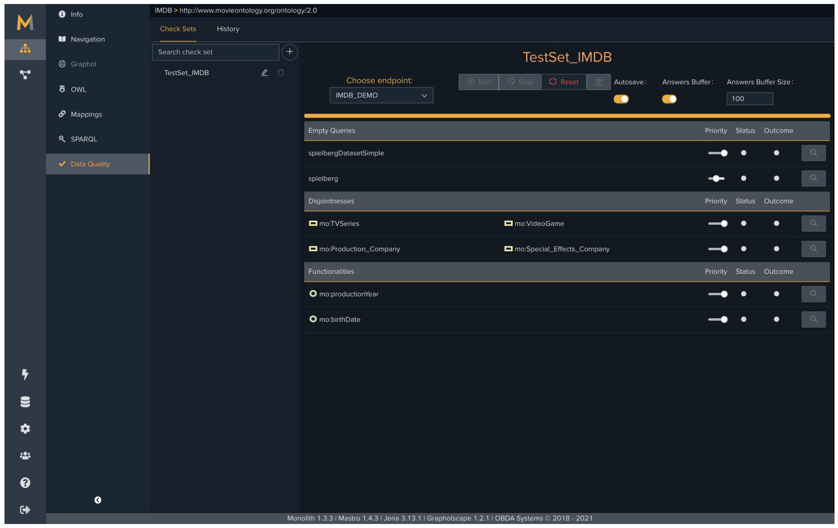

Fig. 2. The Data Quality section in MONOLITH 2.0

straints, property functionality and universal participation constraints, cardinality constraints, and participation constraints. Each such constraint type is processed by MASTRO 2.0 in order to produce a specific kind of SPARQL query in such a way as to interpret any data extracted by these queries as a violation of the ontology data integrity rule. In MONOLITH 2.0's interface users are provided a preview of such results for each constraint, along with the query plan details to understand the provenance of the violation. The Data Quality verification process essentially consists in building a set of constraints to check: the user selects one or more integrity constraints for each constraint type to schedule for verification, and then provides a priority level for each constraint. Once the execution of each constraint is complete, MONOLITH 2.0 allows to save the execution to a *history* log, which provides the results for each query and a representation of the aggregate results, based on priority and/or constraint type, through charts and graphs.

3 Application scenarios and Demo Session Overview

MONOLITH 2.0 is currently commercially distributed by OBDA Systems, and is being used in various OBDM-related projects, in particular with clients from the Italian public administration sector. The more common application scenarios are projects in which data from different business units is modelled in an ontology, therefore allowing for integrated data access and data quality verification processes, and projects in which MONOLITH 2.0 and MASTRO 2.0 are used to produce Linked Open Data (LOD) datasets.These LOD datasets can either be obtained by the conversion (or *triplification*) of structured data, typically CSV, XML, or JSON files) into RDF datasets by MASTRO 2.0, or by extracting new datasets through MONOLITH 2.0 from legacy data stores.

Participants during the demo will be able to see MONOLITH 2.0 in action on one of the specifications used in these projects, specifically the SIR (System of Integrated Registers) ontology, which is being built in a joint project between OBDA Systems and the Italian National Institute of Statistics (ISTAT). This ontology integrates information from statistical censuses regarding, among others, demographic, territorial, and public administration data. Another specification that will be featured in the demo is the Movie Ontology [2], which provides a vocabulary to semantically describe movie related concepts. During the demo, attendees will interact with MONOLITH 2.0 in the above scenarios and will be introduced to its main new features.

References

1. N. Antonioli, F. Castanò, C. Civili, S. Coletta, S. Grossi, D. Lembo, M. Lenzerini, A. Poggi, D. F. Savo, and E. Virardi. Ontology-based data access: The experience at the italian department of treasury. In *Proc. of CAISE 2013*, volume 1017 of *CEUR Workshop Proceedings*, pages 9–16, 2013.
2. A. Bouza. Mo - the movie ontology, 2010. [Online; 26. Jan. 2010].
3. D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. Tractable reasoning and efficient query answering in description logics: The *DL-Lite* family. *J. Autom. Reasoning*, 39(3):385–429, 2007.
4. S. Das, S. Sundara, and R. Cyganiak. R2RML: RDB to RDF mapping language. W3C Recommendation, World Wide Web Consortium, Sept. 2012. Available at `http://www.w3.org/TR/r2rml/`.
5. F. Di Pinto, D. Lembo, M. Lenzerini, R. Mancini, A. Poggi, R. Rosati, M. Ruzzi, and D. F. Savo. Optimizing query rewriting in ontology-based data access. In *Proc. of EDBT 2013*, pages 561–572. ACM Press, 2013.
6. M. Giese, A. Soylu, G. Vega-Gorgojo, A. Waaler, P. Haase, E. Jiménez-Ruiz, D. Lanti, M. Rezk, G. Xiao, Ö. L. Özçep, and R. Rosati. Optique: Zooming in on big data. *IEEE Computer*, 48(3):60–67, 2015.
7. J. M. Gómez-Pérez, J. Z. Pan, G. Vetere, and H. Wu. Enterprise knowledge graph: An introduction. In *Exploiting Linked Data and Knowledge Graphs in Large Organisations*, pages 1–14. Springer, 2017.
8. S. Harris and A. Seaborne. SPARQL 1.1 query language. Mar. 2013. Available at `http://www.w3.org/TR/sparql11-query`.
9. E. Kharlamov, D. Hovland, M. G. Skjæveland, D. Bilidas, E. Jiménez-Ruiz, G. Xiao, A. Soylu, D. Lanti, M. Rezk, D. Zheleznyakov, M. Giese, H. Lie, Y. E. Ioannidis, Y. Kotidis, M. Koubarakis, and A. Waaler. Ontology based data access in statoil. *J. Web Semant.*, 44:3–36, 2017.
10. D. Lembo, D. Pantaleone, V. Santarelli, and D. F. Savo. Easy OWL drawing with the graphol visual ontology language. In *Proc. of KR*, pages 573–576. AAAI Press, 2016.
11. M. Lenzerini. Managing data through the lens of an ontology. *AI Magazine*, 39(2):65–74, 2018.
12. V. Santarelli, L. Lepore, M. Namici, G. Ronconi, M. Ruzzi, and D. F. Savo. Monolith: an OBDM and knowledge graph management platform. In M. C. Suárez-Figueroa, G. Cheng, A. L. Gentile, C. Guéret, C. M. Keet, and A. Bernstein, editors, *In Proc. of ISWC 2019 Satellite Tracks, Auckland, New Zealand, October 26-30, 2019*, volume 2456 of *CEUR Workshop Proceedings*, pages 173–176. CEUR-WS.org, 2019.