Interactive Dashboard for Ontology Catalogs

Jakub Skřišovský[1], Avetis Mkrtchian[1] and Petr Křemen[1,*,†]

[1]Czech Technical University in Prague, Faculty of Electrical Engineering, Dept. of Computer Science

Abstract

We introduce a web-based tool for interactive monitoring of quality and structural properties of a set of ontologies. The tool performs custom SHACL-based data validation as well as computation of various structural metrics. These metrics are based on the OntoMetrics framework, allowing to detect the type of an OWL ontology based on its structural properties (like depth, width, sibling count, etc.). We present a prominent use case of OBO ontology catalog, allowing to dynamically index the catalog through time, detect version changes, compute structural properties and providing interactive visualizations over these data.

Keywords

OWL, ontology, OBO, dashboard

1. Introduction

Ontologies are well-accepted for sharing meaning of complex data, supporting their integration and understanding. Various communities, like OBO Foundry [1] or IOF[2]) as well as governmental organizations, like the Publication Office of the European Union[1] publish and supervise development of a collection of OWL [3] ontologies, including supervising their quality, validity and reusability.

However, these ontology catalogs focus mainly on *provenance metadata* (e.g. who is the author, when the dataset was updated) and much less on the actual *content metadata*) (e.g. how does the data schema, classes, properties, look like). Lack of such metadata makes it difficult to monitor the quality of the ontologies, as well as to understand the interrelationships between the ontologies.

In our previous work [4] we introduced a generic interactive dashboard framework over a catalog of ontologies evolving in time. In this paper we extend our previous work on the dashboard with ontology metrics, allowing to assess the depth, width, level of branching and other structural properties. The ontologies in the catalog are periodically indexed, their quality validated using SHACL and structural quality metrics computed and stored using the Data Quality Vocabulary [5]. We exemplify our approach on the use case of the OBO Foundry, which maintains a catalog of *OWL ontologies* in the domain of biology, chemistry and medicine.

Section 2 presents OBO Foundry, as well as existing tools for validating and monitoring quality of a set of ontologies. After a brief introduction of the dashboard framework we present the overall architecture as well as quality metric computation implementation in section 3. Information about the OBO Foundry use case and evaluation is in section 4 and the paper is concluded in section 5.

2. Related Work

The OBO Foundry [6] is a community that maintains a catalog of open, collaborative, logically well-formed and mutually interoperable biomedical ontologies. Since OBO Foundry is concerned with the ontology quality problem, they developed the ROBOT tool, which is a general-purpose swiss-knife tool for processing, managing and especially validating OWL ontologies. OBO Foundry came up with their own principles to enforce on the respective ontologies, including openness, common formatting, URI/Identifier spacing, versioning, specified scope, textual definitions, relations, provided

Developers Workshop, co-located with SEMANTiCS'25: International Conference on Semantic Systems, September 3–5, 2025, Vienna, Austria

skrisjak@fel.cvut.cz (J. Skřišovský); mkrtcave@fel.cvut.cz (A. Mkrtchian); petr.kremen@fel.cvut.cz (P. Křemen)

[1]https://op.europa.eu/en/web/eu-vocabularies/ontologies

documentation, documented plurality of users, commitment to collaboration, locus of authority, naming conventions, notification of changes, maintenance, term stability and responsiveness. The OBO Foundry presented its own dashboard solution based on the ROBOT tool, which tests the OBO Foundry ontology catalog with a certain periodicity and provides a report including the number of violations (quality check) and also compliance with these principles for each ontology. According to OBO Foundry, this solution should help developers more easily identify problem areas to be improved. In our previous work we already proposed generalization of the validation using SHACL rules [4]. SHACL focuses on validating individual statements (triples), ensuring conformance to predefined shapes, but does not provide a holistic evaluation of the ontology's structure or quality.

Existing works on structural metrics exist, ranging from simple computations of basic statistics (e.g. class/axiom counts), e.g. via ROBOT[7], which is widely used within the OBO Foundry community. Structural properties of an ontology have been studied e.g. in OntoMetrics[8]. It has been further evolved towards Neontometrics[9], allowing tracking and comparing metric values over time for ontologies stored in GitHub repositories.

3. Architecture

The overall architecture of the dashboard framework is depicted in 1.

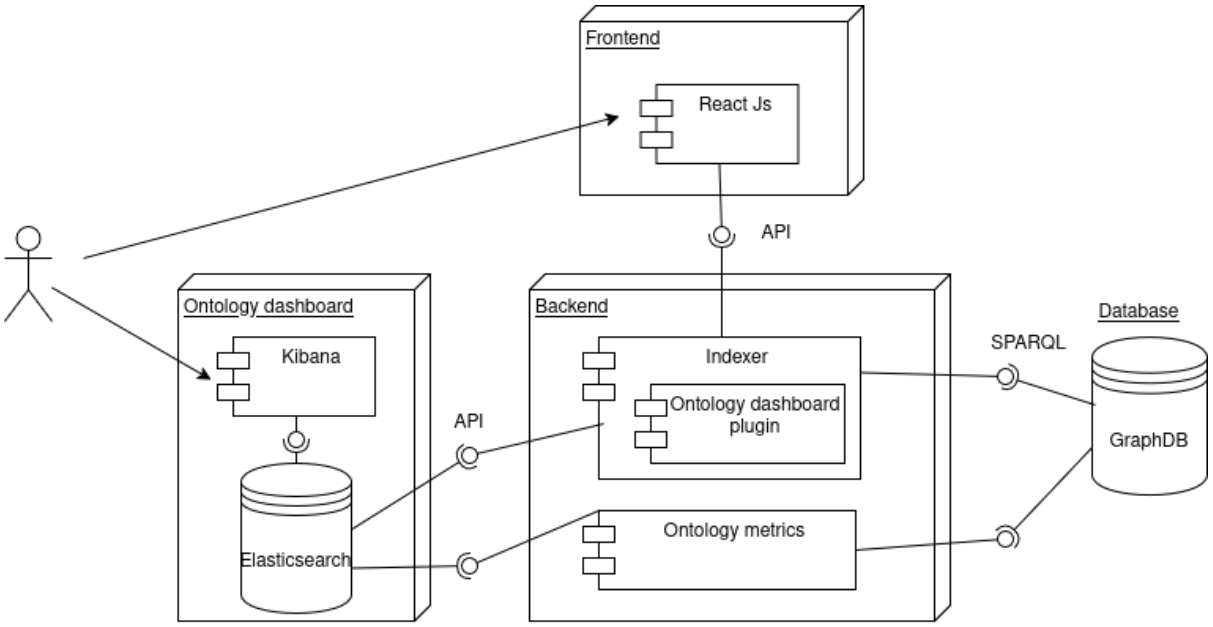

Figure 1: Component diagram

The core component is the backend indexing data into the ELK stack. The *indexer* component is responsible for validating OWL ontologies and indexing the results to Elastics. The validation is performed using SHACL (Shapes Constraint Language) as described in [4]. The indexed data are visualized using Kibana, showing different interactive dashboards for validation results and metrics. GraphDB is a triplestore where the data is stored after it has been processed by the plugin and then queried by the RDF Indexer.

Our novel contribution are structural metrics that are computed and indexed by the *Ontology metrics* component[2]. The component implements selected metrics[10] defined in OntoMetrics[8], revealing the structure of the ontologies.

[2]https://gitlab.fel.cvut.cz/skrisjak/obo-ontologies-metrics

The size and scope of an ontology can be initially assessed through the overall number of classes. Additionally, we distinguish specific types of classes, such as **leaf classes**—those without any subclasses, typically representing concrete concepts—and **tangled classes**, which inherit from more than one superclass.

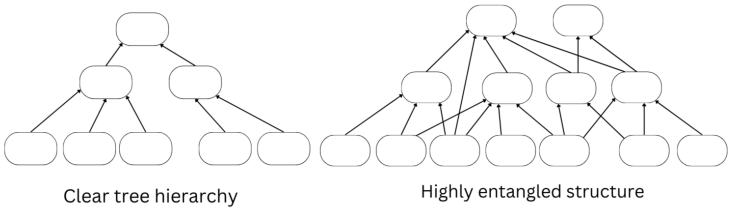

Figure 2: Ontology Complexity

Ontological hierarchies exhibit both vertical and horizontal complexity, measured by their:

- **Breadth** — the number of classes at each hierarchical level, reflecting how concepts are grouped and differentiated.
- **Depth** — the number of subclassing steps from the ontology root to leaf classes. This metric helps identify how deep the ontology goes in terms of specialization, and whether concrete concepts are well-distinguished from abstract categories.

Figure 3: Ontology Dimension

These metrics focus on the similarity of classes in terms of their structural context. Specifically, the **sibling count** captures how many classes share the same direct superclass with a given class. Meanwhile, the **sibling group size** measures the number of direct subclasses of each non-leaf class, providing insight into the ontology's overall branching pattern and the distribution of conceptual categories.

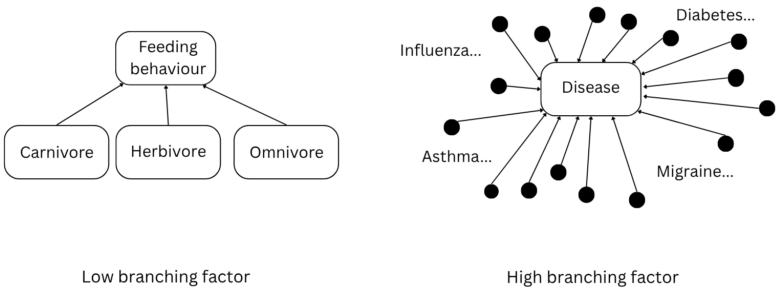

Figure 4: Ontology Branching

The representation of the metric as well as validation results is unified using a part of the DQV (Data Quality Vocabulary) We specialized DQV with a custom ontology defining specific metrics – the OQO (Ontology Quality Ontology)[3] serves as a structured vocabulary for describing a wide range of ontology quality metrics. Each metric OQO is modeled as an instance of `dqv:Metric`, with SKOS annotations such as `skos:prefLabel`, and `skos:definition`, also characterized by its `dqv:expectedDataType`. Metrics are grouped into semantic dimensions using `dqv:inDimension` property, which captures the aspect of quality the metric focuses on. All dimensions are grouped thematically under broader categories using `dqv:inCategory` property.

Each measured value is expressed as a `dqv:QualityMeasurement`. Unlike ontology validation—where each constraint violation is stored individually—ontology metrics are indexed collectively for each ontology procession.

Structural metrics can serve as feedback for developers to check if the design matches the intention of the developed ontology and reveal possible anti-patterns. They can also be used to recognize specific types of ontologies, such as glossary, taxonomy, thesauri, etc. by reaching the specific values for specific metrics. For example, if the ontology has most of the classes in one hierarchical level (max breadth / class count > 0.9) and low average depth, we would categorize it as a glossary - a simple set of terms. If it had a larger average depth (>4) and low tangled class ratio (tangled class count / class count < 0.1), we would assume it as taxonomy to categorize objects. As the average depth, ratio of tangled classes, and average sibling group size (number of subclasses for each parent class) rise, so does the complexity and richness of an ontology.

We explored possibilities of acquiring metrics by using the Apache Jena library - either by executing SPARQL queries or using the API itself. Not every ontology is materialized (reasoned before serialization), so the OWL API is included, ready to use well-known reasoners, such as Pellet, Hermit, and ELK. Large ontologies require a lot of memory and performance to process, so to prevent unnecessary loading, the ontology version extractor [4] retrieves the ontology header with the current version, and checks for existing metrics, avoiding duplicate procession.

4. Use Case

As a use case, we created dashboard for the OBO Foundry ontology catalog. OBO Foundry also offers its own dashboard, which provides basic information about ontologies such as: tests on OBO Foundry principles, violation reports, metrics, etc. The main limitation is that this dashboard does not provide the whole violation report, so that the user interested in this will have to use the ROBOT tool, which does not simplify but only complicate the process of browsing the necessary data about a ontology. Also an important limitation of the OBO Dashboard is the lack of any filtering of the data, e.g. to view violations of a specific level, or moreover to view all violations related to a particular subject in a ontology.

The dashboard shows validation results as well as computed metrics and their evolution in time. Our dashboard consists of five main sections: ”All ontologies”, ”Single ontology”, ”Specific ontologies”, ”Ontologies metrics” and ”Single ontology metrics”. Section ”All ontologies” provides general data and statistics over catalog, ”Single ontology” section shows detailed violation report and OBO metrics, and ”Specific ontologies” is used for comparing ontologies. ”Ontologies metrics” and ”Single ontology metrics” are relevant, but with our own metrics. ”Ontologies metrics” compares ontologies with a table and shows the distribution of metrics values over a catalog, such as class count 5.

[3] http://purl.org/oqo

[4] https://github.com/psiotwo/ontology-version-extractor

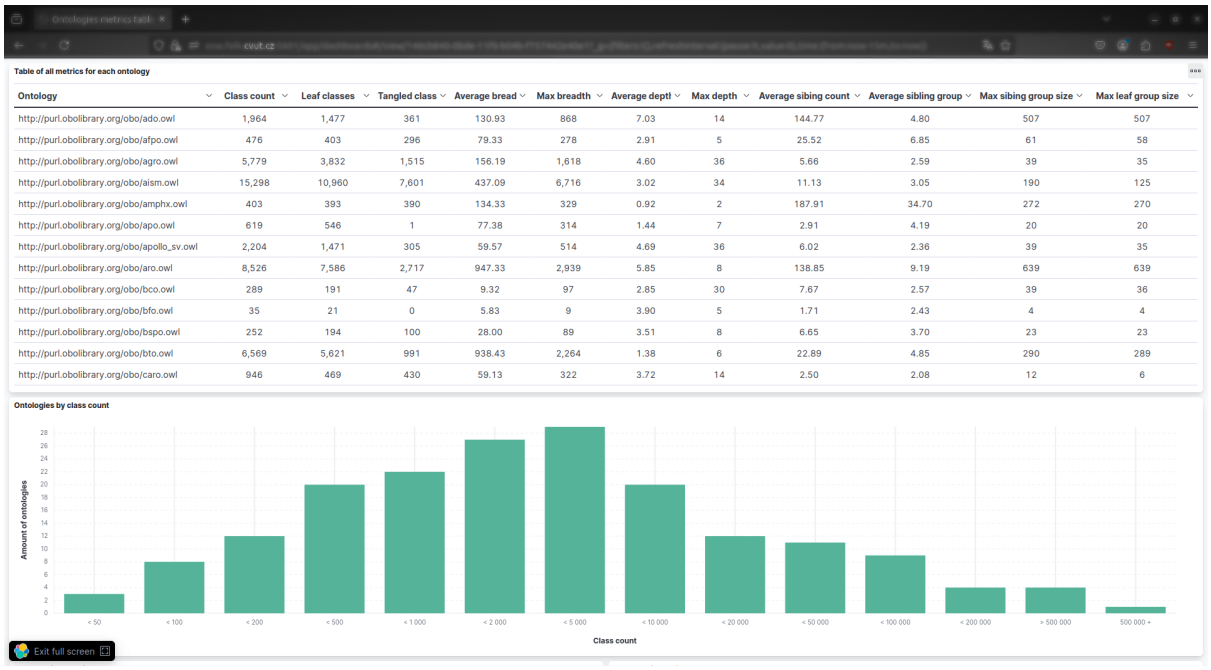

Figure 5: Ontologies metrics section

Figure 6: Ontologies metrics section

5. Conclusions

This paper focused on augmenting our dashboard solution with structural ontological metrics which is yet to be evaluated by the community. The dashboard framework provides insights for ontology curators to the current status of ontologies and their evolution in time and our previous testing with OBO users revealed the need to still improve the understandability of the Kibana user interface for non-technical people. Apart from UX improvements, we would like to also work on traceability features, including providing details on to the axioms not passing the validation, as well as computing and showing changes between ontology versions.

Declaration on Generative AI

The author(s) have not employed any Generative AI tools.

References

[1] R. Jackson, N. Matentzoglu, J. A. Overton, R. Vita, J. P. Balhoff, P. L. Buttigieg, S. Carbon, M. Courtot, A. D. Diehl, D. M. Dooley, W. D. Duncan, N. L. Harris, M. A. Haendel, S. E. Lewis, D. A. Natale, D. Osumi-Sutherland, A. Ruttenberg, L. M. Schriml, B. Smith, C. J. Stoeckert Jr., N. A. Vasilevsky, R. L. Walls, J. Zheng, C. J. Mungall, B. Peters, Obo foundry in 2021: operationalizing open data principles to evaluate ontologies, Database 2021 (2021) baab069. URL: https://doi.org/10.1093/database/baab069. doi:10.1093/database/baab069. arXiv:https://academic.oup.com/database/article-pdf/doi/10.1093/database/baab069/40854912/baab069.pdf.

[2] B. Kulvatunyou, M. Drobnjakovic, F. Ameri, C. Will, B. Smith, The industrial ontologies foundry (iof) core ontology, Formal Ontologies Meet Industry (FOMI) 2022, Tarbes, FR, 2022. URL: https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=935068.

[3] OWL 2 Web Ontology Language Document Overview, W3C Recommendation, W3C, 2009. Https://www.w3.org/TR/2009/REC-owl2-overview-20091027/.

[4] A. Mkrtchian, P. Kremen, An interactive dashboard for ontology quality monitoring, in: F. Farinelli, A. D. de Souza, E. R. Felipe (Eds.), Proceedings of the International Conference on Biomedical Ontologies 2023 together with the Workshop on Ontologies for Infectious and Immune-Mediated Disease Data Science (OIIDDS 2023) and the FAIR Ontology Harmonization and TRUST Data Interoperability Workshop (FOHTI 2023), Brasília, Brazil, August 28 - September 1, 2023, volume 3603 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2023, pp. 204–210. URL: https://ceur-ws.org/Vol-3603/workshopFOHTI.pdf.

[5] R. Albertoni, A. Isaac, Data on the Web Best Practices: Data Quality Vocabulary, W3C Note, W3C, 2016. Https://www.w3.org/TR/2016/NOTE-vocab-dqv-20161215/.

[6] B. Smith, M. Ashburner, C. Rosse, J. Bard, W. Bug, W. Ceusters, L. Goldberg, K. Eilbeck, A. Ireland, C. Mungall, N. Leontis, P. Rocca-Serra, A. Ruttenberg, S.-A. Sansone, R. Scheuermann, N. Shah, P. Whetzel, S. Lewis, The obo foundry: Coordinated evolution of ontologies to support biomedical data integration, Nature biotechnology 25 (2007) 1251–5. doi:10.1038/nbt1346.

[7] R. Jackson, J. Balhoff, E. Douglas, N. Harris, C. Mungall, O. Overton, Robot: A tool for automating ontology workflows, Database (2019). URL: https://link.springer.com/epdf/10.1186/s12859-019-3002-3?author_access_token=bB8BLjFWrdh42vR6DjT-nG_BpE1tBhCbnbw3BuzI2RPCZ2BK7EeexaCNYfT-cCz8Q_mrZomT2_svoQf12CW661Sagzw6JGF9DhJq3Q3fTPdMGFMtais7MRgx8-kDhp6uC9g2qcVh5FumTsveV22XVQ%3D%3D.

[8] M. Poppe, M. Lichtwark, Ontometrics, https://ontometrics.informatik.uni-rostock.de/ontologymetrics/index.jsp, 2016.

[9] A. Reiz, Neontometrics, http://neontometrics.informatik.uni-rostock.de/#/, 2023.

[10] J. Skřišovský, Quality checking of semantic vocabularies, CTU dspace (2025). URL: https://dspace.cvut.cz/handle/10467/122781?locale-attribute=en.