rudof: A Rust Library for handling RDF data models and Shapes

Jose-Emilio Labra-Gayo[1], Angel Iglesias-Préstamo[1], Diego Martín-Fernández[1] and Marc-Antoine Arnaud[2,*,†]

[1]*WESO Lab, University of Oviedo, Spain*
[2]*Lum::invent, France*

Abstract

In this paper we present rudof, a Rust library for RDF and RDF Data shapes. The library can be used as a command line tool but it can also be invoked from different systems. The flexibility of Rust enables the creation of binaries in Windows, Linux and Mac, as well as offering Python bindings and WebAssembly components. The library supports ShEx and SHACL, as well as other RDF data modeling languages like DCTAP, offering conversion mechanisms between them. It can also be used to generate UML-like visualizations of those data models.

Keywords

RDF, ShEx, SHACL, Data Quality, Rust

1. Introduction

With the increasing adoption of RDF data shapes, there is a need for efficient libraries and tools which can be used to validate and process RDF. ShEx[1] was introduced in 2014 as a human-readable and concise language for RDF validation which was later adopted by Wikidata in 2019, and SHACL was accepted as a W3C recommendation in 2017[1]. Both ShEx and SHACL have been increasingly adopted to increase the quality of RDF data. At the same time, other technologies have appeared to help domain experts declare their expectations on RDF-based knowledge graphs like DCTAP[2], a tabular template that can be used to define data shapes.

The Rust programming language[3] offers some interesting features which are intended to increase safety with static typing while keeping performance competing with other low-level languages. Another advantage of Rust is the possibility of developing Python bindings that allow Python programmers to invoke Rust libraries that have better performance and even the

ISWC 2024: Posters and demos
labra@uniovi.es (J. Labra-Gayo); angel.iglesias.prestamo@gmail.com (A. Iglesias-Préstamo); diegomartinfnz@gmail.com (D. Martín-Fernández); marc-antoine.arnaud@luminvent.com (M. Arnaud)
http://labra.weso.es/ (J. Labra-Gayo); https://angelip2303.github.io/ (A. Iglesias-Préstamo); https://luminvent.com/ (M. Arnaud)
0000-0001-8907-5348 (J. Labra-Gayo); 0009-0004-0686-4341 (A. Iglesias-Préstamo); 0009-0003-6640-9474 (D. Martín-Fernández); 0009-0004-2130-3366 (M. Arnaud)

[1]https://www.w3.org/TR/shacl/
[2]https://www.dublincore.org/specifications/dctap/
[3]https://www.rust-lang.org/

compilation to Web Assembly[4], that can interoperate with Javascript code.

In this paper, we present a new library implemented in Rust, called rudof[5] which offers support for both ShEx and SHACL, conversion between different data models like DCTAP, generation of UML-like visualizations and RDF data validation. The different components of the library are published at crates.io[6], the Rust module registry, and the library publishes binary releases in Windows, Linux, and MacOS, as well as Debian packages, Docker images, and Python bindings[7].

2. rudof modules and features

The library consists of different modules which are also published as Rust crates. In order to minimize external dependencies to other libraries, we created a simple RDF trait called SRDF which offers the basic RDF functionalities required for validation (mainly accessing the neighborhood of a node). We provide two implementations of SRDF, one based on RDF files like Turtle, and another one based on SPARQL, which allows the library to validate RDF graphs obtained either from files or through SPARQL endpoints.

The library defines other crates that contain the abstract syntax tree representation of both ShEx and SHACL called *shex_ast* and *shacl_ast*, as well as their corresponding parsers and validators. There is a module called *shapes_converter* which contains converters between different data models like DCTAP to ShEx, ShEx to UML visualizations, etc. A special module is *rudof_cli*, which implements the command line tool which is later published as a binary called *rudof* in different platforms like Linux, Mac and Windows.

The library already supports the following features[8]:

- Show information about RDF data and convert between different formats like Turtle, NTriples, RDF/XML, etc.
- Support for data shapes languages like ShEx and SHACL: show information about shapes and schemas, and validate RDF data to check conformance.
- Parsing DCTAP data models and conversion to shapes schemas. As an example, Figure 1a contains a DCTAP file which can be obtained from an spreadsheet in CSV and figure 1b shows the result of converting it to ShEx.
- Generating UML visualizations of shapes data models. As an example, Figure 1c shows the UML generated from 1b.
- Generating HTML representations of those schemas, which can be useful when the schemas contain a large number of shapes. In these cases, the UML visualizations can be too big and become unusable, while representing each shape in its own web page makes it possible to browse the shapes in the schema.
- Obtaining information about the neighborhood of a node in an RDF graph (either incoming or outgoing arcs), which can be useful to create a schema or to debug the validation results.

[4] https://webassembly.org/
[5] https://rudof-project.github.io/rudof/
[6] https://crates.io/
[7] https://github.com/rudof-project/rudof/releases
[8] The project Wiki page contains instructions and how-to guides

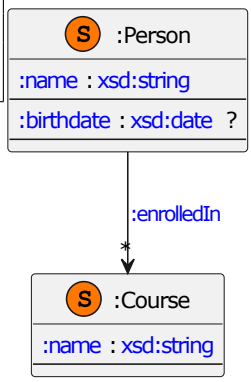
- Other conversions: we are exploring the conversion between ShEx/SHACL and SPARQL as it is a feature that can be useful to create queries based on some shapes.

ShapeId	PropertyId	Mandatory	Repeatable	valueDatatype	valueShape
Person	name	true	false	xsd:string	
	birthdate	false	false	xsd:date	
	enrolledIn	false	true		Course
Course	name	true	false	xsd:string	

(a) DCTAP example

```
prefix : <http://example.org/>
prefix xsd: <http://www.w3.org/2001/XMLSchema#>
:Person { :name       xsd:string  ;
          :birthdate  xsd:date  ? ;
          :enrolledIn @:Course  * }
:Course { :name       xsd:string  }
```

(b) ShEx obtained from DCTAP example conversion



(c) ShEx visualization

3. Performance benchmarks.

Using Rust as an implementation language can improve the performance compared to Java or Python. Some preliminary benchmarks have been conducted[9] to compare the tool against several state-of-the-art SHACL implementations. As of September 2, 2024, the tool demonstrates performance improvements in SHACL validation when compared to Apache Jena[10] and TopQuadrant[11], although it exhibits longer execution times than rdf4j[12]. In addition to these SHACL validation benchmarks, the Python bindings provided by rudof can also be used to compare performance with other Python libraries for SHACL like pySHACL [2]. In this case, measuring loading, parsing and validating time for the same graph. A summary of the performance results is available in Table 1[13].

Dataset	rudof	rdf4j	Apache Jena	TopQuadrant	pyrudof	pySHACL
10-LUBM	7.8971	1.6447	60.3583	85.7421	39,364.2842	72,227.2940

Table 1
Performance comparison of SHACL validation across state-of-the-art implementations against the rudof. Execution times are reported in milliseconds, using the LUBM dataset with the same SHACL shape across all comparisons.

[9] https://github.com/weso/shacl-validation-benchmarks
[10] https://jena.apache.org/
[11] https://github.com/TopQuadrant/shacl
[12] https://rdf4j.org/
[13] More details about performance comparisons will be available at the rudof github repository

4. Related work.

Although there are several libraries for ShEx, SHACL or DCTAP, most of them are focused on one of those technologies, and do not offer a common mechanism for conversion between shapes-based data models. The main exception could be the SHaclEX library[14] written by the first author of this paper in Scala.

In the Rust ecosystem, Sophia [3] is a toolkit for RDF and linked data which contains several traits although it does not support for shapes yet. Oxigraph[15] is an graph database that supports SPARQL written in Rust that also publishes several crates related with RDF and doesn't support shapes yet. Recently, an W3C RDF Rust Common Crates community group[16] has been created and we are planning to align the dependencies with the expected results of that group.

5. Conclusions and future work

Although rudof is still work-in-progress, we consider that it can fill a need in the RDF data shapes tools Rust ecosystem. The Rust programming language offers some advantages in terms of performance and memory safety. It also offers the possibility to generate binaries for different operating systems like Windows, Linux and Max, as well as Python bindings. We are exploring the use of the library in WebAssembly[17]. Our goal is to gradually migrate the code of our RDFShape playground [4] which was implemented in React and Scala to a new version based on Web Assembly and Rust.

References

[1] E. Prud'hommeaux, J. E. Labra Gayo, H. Solbrig, Shape Expressions: An RDF Validation and Transformation Language, in: H. Sack, A. Filipowska, J. Lehmann, S. Hellmann (Eds.), Proceedings of the 10th International Conference on Semantic Systems, SEMANTICS 2014, Leipzig, Germany, September 4-5, 2014, ACM Press, 2014, pp. 32–40. doi:10.1145/2660517.2660523.

[2] A. Sommer, N. Car, pyshacl, 2024. URL: https://doi.org/10.5281/zenodo.10958008. doi:10.5281/zenodo.10958008.

[3] P.-A. Champin, Sophia: A Linked Data and Semantic Web toolkit for Rust, in: E. Wilde, M. Amundsen (Eds.), The Web Conference 2020: Developers Track, Taipei, TW, 2020. URL: https://www2020devtrack.github.io/site/schedule.

[4] J. E. Labra Gayo, D. Fernández Álvarez, H. García-González, RDFShape: An RDF Playground Based on Shapes, in: Proceedings of the ISWC 2018 Posters and Demonstrations, Industry and Blue Sky Ideas Tracks, co-located with 17th International Semantic Web Conference, volume 2180 of *CEUR Workshop Proceedings*, 2018.

[14] https://www.weso.es/shaclex/

[15] https://github.com/oxigraph/oxigraph

[16] https://www.w3.org/community/r2c2/

[17] A prototype based on WebAssembly is available at https://uo271080.github.io/TFG_UO271080/