The OVAL Policy Evaluator Engine for ODRL

Paolo Pareti[1], Jaime Osvaldo Salas[1], Christopher Maidens[1] and George Konstantinidis[1]

[1]University of Southampton, Southampton, UK

Abstract

This paper presents OVAL, a novel evaluation engine for the Open Digital Rights Language (ODRL). ODRL is a W3C Recommendation that provides a flexible and interoperable way to express policies, and it is increasingly becoming the de-facto standard to express rights, obligations and usage policies in a range of fields. Despite its growing adoption, ODRL lacks a reference evaluator, as several aspects of its evaluation semantics are not yet formally defined, leading existing tools to interpret the specification differently. OVAL addresses this gap by (1) faithfully implementing a recent comprehensive mathematical formalisation of ODRL, resulting in clear and unambiguous policy evaluation; (2) providing extensive coverage of the ODRL language, including the first comprehensive support for all ODRL rule and constraint types; (3) including dedicated support for both the access control and monitoring usage scenarios (4) employing an efficient single-pass evaluation algorithm that scales to large event logs while also supporting streaming evaluation.

Keywords

ODRL, Policy Evaluation, Policy Monitoring, Access Control

1. Introduction

The Open Digital Rights Language (ODRL) is a W3C Recommendation that aims to provide a flexible and interoperable language and vocabulary to express rights and obligations. Its flexibility has made it ideal for a variety of domains, including the financial,[1] medical [1] and urban-planning [2] sectors. However, a commonly cited concern in both academia and industry is the lack of a standard evaluator for ODRL policies. While the ODRL Community Group is actively working on a formalisation of ODRL semantics,[2] this is currently still in draft form. As a result of this, each ODRL tool implements its own version of the semantics, with details not always fully specified. This is problematic for the problem of policy evaluation, as precise semantics are needed to automate evaluation.

Existing ODRL evaluators include the SolidLabs ODRL Evaluator [3] and the ODRE Framework [4]. Additional evaluators have been reviewed in [5], and a list of ODRL tools, including evaluators, is currently being maintained.[3] While ODRL semantics has not been standardised, ODRL evaluators typically focus on one of two scenarios, namely *monitoring* and *access control*. In the monitoring scenario, a policy is evaluated over a *state of the world*, that is, a log of events that have happened. In this scenario, evaluating a policy means determining whether the events in the state of the world have been compliant with the policy or not, for example by detecting if any forbidden actions have been performed. In the access control scenario, a policy is evaluated on an *access request*, to determine if the requested action can be permitted in compliance with the policy. An access request can also be accompanied by a state of the world object, that provides more context to the access request. For example, when a permission can only be exercised subject to the fulfilment of a duty, such as a payment being required before a dataset can be accessed, evidence of completion of the duty can be stored in this state of the world object.

ISWC 2026 Companion Volume, October 25–29, 2026, Bari, Italy

p.pareti@soton.ac.uk (P. Pareti); j.o.salas@soton.ac.uk (J. O. Salas); c.maidens@soton.ac.uk (C. Maidens); g.konstantinidis@soton.ac.uk (G. Konstantinidis)

0000-0002-2502-0011 (P. Pareti); 0000-0002-9353-8955 (J. O. Salas); 0000-0002-4385-7202 (C. Maidens); 0000-0002-3962-9303 (G. Konstantinidis)



© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

[1]https://www.w3.org/2026/ODRLWS/agenda.html

[2]https://w3c.github.io/odrl/formal-semantics/

[3]https://w3c.github.io/odrl/landscape/

As our main contribution, we present OVAL, an open source ODRL Evaluator backed by the first comprehensive mathematical formal semantics for ODRL 2.2 [6]. This semantics unambiguously defines the expected result of evaluating a policy over a state of the world. Our evaluator further extends this semantics in several directions, such as supporting multiple semantics in the access control scenario and ODRL reasoning (through the `includedIn` and `partOf` relations). To the best of the authors' knowledge, no existing ODRL evaluator supports the extensive range of ODRL features that OVAL supports. Moreover, OVAL provides a simple and efficient evaluation of ODRL policies over states of the world that can be used for automated policy compliance at scale. It is used by industrial partners within several consortiums, such as DataPACT [7] and UPCAST [8], and thus it is designed to be a practical and usable tool.

2. System Architecture

The OVAL evaluator implements two core evaluation functionalities: one for the monitoring and one for the access control scenarios. In the monitoring scenario an evaluation takes two inputs, an ODRL policy and a state of the world object (i.e. a log of past events), and returns a structured compliance report object. The most important information contained in the compliance report is a boolean compliance value, that specifies whether any violations to the policy have been detected in the state of the world.

In the access control scenario, the state of the world object is optional, and the policy is instead evaluated against an access request object, which contains information about the event that is being requested. Information on which semantics should be considered during the evaluation is provided as additional inputs. In this second scenario, an access request can result in one of three possible values, namely *access granted*, *access denied* and *unspecified* (in case a final decision cannot be reached).

2.1. ODRL Policy Support

ODRL policies can be seen as sets of *rules* that define how agents should behave. Rules specify the type of events they apply to by defining a number of components. The core component of a rule is the *action* being regulated. Optional components further refine the applicability of a rule, such as on actions performed by specific *parties* on specific *assets*. Additionally, *constraints* specify more fine-grained conditions over additional properties of rules or other components. These conditions are expressed as a < *left operand*, *operator*, *right operand* > triple, that define, respectively, the property to check (e.g. "resolution"), the type of comparison to perform (e.g. "less than or equal"), and the value to compare against (e.g. "150 dpi"). The meaning of a rule within a policy depends on its type, such as permission or prohibition. For example, the policy in Figure 1 defines two permissions. The first permits the action "read" on any asset by any party. The second allows Alice to print a Manual, but only at a resolution of 150dpi or lower.

```
:policy-001 odrl:permission [
    odrl:action odrl:read ;
  ], [
    odrl:assignee :Alice ;
    odrl:action odrl:print ;
    odrl:target :Manual ;
    odrl:constraint [
      odrl:leftOperand :Resolution ;
      odrl:operator odrl:lteq ;
      odrl:rightOperand "150"^^xsd:integer ;
      odrl:unit :dpi ; ] ] .
```

Figure 1: Example of an ODRL policy.

The OVAL evaluator provides comprehensive support for ODRL. At a syntax level, it accepts all standard RDF serialisations of ODRL policies. In terms of language coverage, it supports all ODRL terms

whose semantics are defined in [6], plus additional ones such as `includedIn`, `partOf` and `conflict`. A detailed list of supported features can be found in the OVAL documentation. As such, it is the first evaluator to provide functional support for all ODRL rule types, which include the basic rule types of *Permissions*, *Prohibitions* and *Obligations*, along with the complex rules of *Duties*, *Remedies* and *Consequences*. Complex rules depend on other rules for their interpretation.

OVAL supports all ODRL constraint operators, in addition to logical constraints – boolean combinations of constraints using the logical operators *and*, *or*, and *xor*. Logical constraints are fully supported, allowing OVAL to process logical constraints of arbitrary complexity. Furthermore, it supports ODRL policies with extended vocabularies and is therefore not limited to specific ODRL profiles. This means, for example, that it can process arbitrary ODRL actions and left operands, making it applicable to domain specific applications that require specialised terminology.

2.2. Alternative ODRL Semantics

Additionally to the ODRL conflict resolution strategy, the access control evaluation of OVAL offers two semantics configurations. The first configuration concerns the treatment of actions that are not regulated by the policy, that is, actions that do not directly match any permission or prohibition rule of the policy. It can take one of the following three values: *permitted-by-default*, *prohibited-by-default* and *unspecified-by-default* that specify that a request for such actions should be permitted, prohibited, or no decision should be made, respectively. Intuitively, under permitted-by-default semantics requests are granted unless explicitly prohibited by the policy, while the converse is true for prohibited-by-default. The unspecified-by-default semantics reflects the general legal principle that no determination can be made with respect to events falling outside the scope of the policy.

The second semantics configuration specifies whether proof of duty completion is mandatory or not. If it is mandatory, a request that can only be accepted on the basis of a permission subject to duties will be rejected unless it is accompanied by a state of the world object that shows records of completion of such duties. This corresponds to a strict check on duty completion. If it is not mandatory, a request is accepted regardless of duties, but the evaluation report will list any unfulfilled duties that the request requires. Intuitively, this corresponds to granting permission on the promise that the requester will fulfil the duties before performing the action.

2.3. State of the World and Access Request Objects

To enhance the applicability in practical scenarios, OVAL ingests simple tabular datasets in CSV format as state of the world objects. In order for the evaluator to correctly interpret the dataset as a state of the world object, two conditions must be met. First, each row of the dataset must represent the record of a distinct event that has taken place or, in the access control scenario, that is planned to take place. Second, each column that contains information about a feature regulated by the policy (either a component such as “asset”, or a left operand such as “resolution”) must reuse the same IRI used in the policy for that component or left operand. Intuitively, this establishes a vocabulary alignment between the column headers of the dataset and the terms used in the policy.

Besides the aforementioned two requirements, the data format of a state of the world affords ample flexibility. In a real-world scenario, event logs might contain information not relevant to a policy, or they might lack other relevant information types. The OVAL evaluator can handle both cases. Column headers not related to the policy are ignored, and when a policy requires evaluation over a feature not present in the log, that feature will be assumed to have a “null” value. This corresponds to a conservative best-effort approach when faced with missing information. Moreover, the order of the columns in the dataset does not matter. Table 1 illustrates a possible state of the world relevant to the example policy presented earlier. This state of the world contains records of four events. The first event records that the action “Print” has been performed by “Alice” on asset “Manual” at a “Resolution” of 150 dpi, with unspecified page numbers. The second event states that “Bob” has “Read” 80 “Pages” of asset “Book”, but leaves the “Resolution“ feature unspecified, as it is not relevant to this type of action.

Table 1
Example of a state of the world object in tabular form.

odrl:dateTime	odrl:Party	odrl:Action	odrl:Asset	:Resolution	:Pages
2026-10-10T10:12:00	Alice	Print	Manual	150 dpi	null
2026-10-10T10:11:00	Bob	Read	Book	null	80
2026-10-10T10:10:00	Bob	Print	Manual	100	300
2026-10-11T09:09:00	Alice	Print	Manual	300 dpi	null

An access request object is a JSON object that, intuitively, represents a potential row of data of a state of the world. As such, it contains key-value pairs, with the keys mapping to the same features as the columns of a state of the world, and the values representing the intended value in the request. If a specific date for performing the action is not specified, the evaluator assumes the requester wants to perform the action at the current time. For example, the following JSON object describes a request from Alice to print a manual at the current time.

```
{
"odrl:Party": "Alice",
"odrl:Action": "Print",
"odrl:Asset": "Manual",
}
```

Figure 2: Example of an access request JSON object.

2.4. Evaluation Engine

The OVAL evaluator engine efficiently processes large states of the world by performing a single-pass evaluation over its recorded events. After ensuring the log is ordered, it processes one row at a time from the oldest to the newest. Using this approach, it is able to detect, for example, that a duty has been correctly performed before the associated permission was exercised, or that a violated prohibition was then remedied by a subsequent remedial action. For a policy of fixed complexity, each row of the state of the world is evaluated in constant time. Based on this algorithm, OVAL also exposes a streaming evaluation functionality that is able to process states of the world streaming one event at a time.

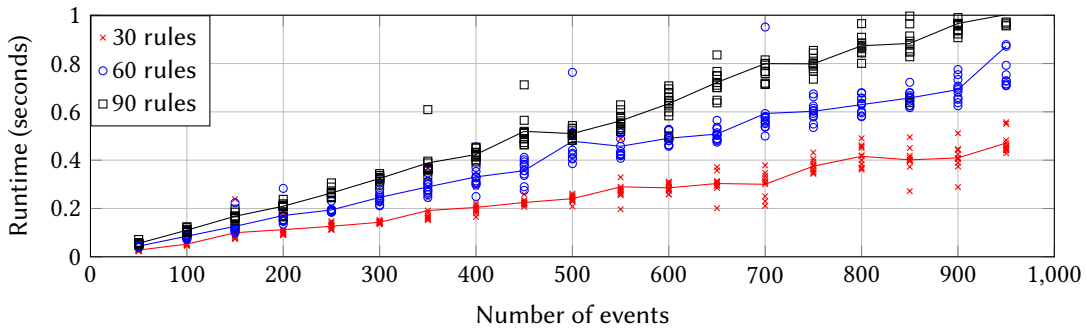


Figure 3: Runtime as a function of the number of events in the state of the world increases.

The OVAL evaluator was tested for scalability over states of the world of increasing size. Efficient processing of large states of the world is important because, while the size of a policy, or an access request, can often be assumed to remain constant, states of the world can be very large. In the monitoring scenario, for example, they can constantly increase in size. This test uses functionalities to generate synthetic policies and associated synthetic states of the world of arbitrary sizes [5]. The policy generator generates the required number of rules for a policy by selecting random action, assignee and target components, and randomly adding additional constraints, taking values from a predefined

pool of constants derived from the ODRL standard profile. Synthetic states of the world for a policy are generated by selecting a random rule in the policy, and then generating a row that matches its conditions. Figure 3 shows the result of this scalability test for random policies of 30, 60 and 90 rules respectively. This experiment shows how the evaluation time increases linearly, and how OVAL can evaluate hundreds of events for large policies (containing 90 rules) in under a second.

In the monitoring scenario the result of the evaluation that OVAL generates is a compliance report comprised of several fields. As previously mentioned, the main field is a boolean value indicating whether the state of the world is compliant. In case of non-compliance, the additional fields provide more information on how the policy was violated, allowing better diagnosis of the compliance issues. In particular, the report lists the row number of the events of the states of the world that were not found to be explicitly permitted, and the rows of those that are explicitly prohibited by the policy. Moreover, if any required obligation, duty, consequence or remedy remained unfulfilled, these will also be listed in the report.

The state of the world in Table 1 does not comply with the policy in Figure 1. While the events in the first two rows are compliant, the events in the last two are not. This is because they are not explicitly permitted, and the evaluation in the monitoring scenario uses prohibited-by-default semantics. The event in the third row is not permitted because the print action is performed by Bob, instead of Alice, while the event in the fourth row is not permitted because Alice printed at a resolution higher than the permitted limit.

In the access control scenario, the evaluation report details whether the request is accepted, rejected, or if it is left unspecified. This is complemented with additional information that explains the reason for such as a decision, and any particular conditions, such as any required duties. Considering the policy in Figure 1 under prohibited-by-default semantics, the request in Figure 2 is rejected due to the resolution feature not being specified, and thus it being impossible to determine that it would be low enough to satisfy the conditions of the permission. Since the resolution feature is unspecified, the evaluation report will contain the information that, should this feature be specified, the acceptance decision could change. Unlike a simple rejection, this can inform a requester on how to correct their request so that it may be accepted.

3. System Implementation

The OVAL evaluator has been implemented in Python as part of a toolkit of ODRL functionalities. The source code is available on GitHub[4] and a permanent version is archived on Zenodo [9]. A live version is also available for testing.[5] To ensure the correctness of the evaluation algorithm, the OVAL code contains a comprehensive test suite of policy, state of the world and access request files for all of the major ODRL features implemented.[6] OVAL can be used programmatically as a library of evaluation functions, or it can be deployed as a Docker container. When deployed as a service, it exposes its evaluation functionality through a FastAPI endpoint, and it provides an intuitive frontend interface.

The OVAL evaluator contains dedicated graphical interfaces for both the monitoring and the access control scenarios. These interfaces allow the user to upload an ODRL policy, a CSV file representing a state of the world and, in the access control interface, a JSON file containing the access request. To enhance usability, states of the world are presented in a tabular format. However, a “Raw Text” toggle over the state of the world field can be switched on and off to visualise and edit the raw text of the CSV file. This allows users to quickly test the effects of modifying the values of the state of the world, to see their effects on compliance. If the user is unsure about which column names the state of the world CSV file should contain, they can click the “Show expected column names” button under the policy input field to display the complete list of such IRIs. In the interface for the access control scenario, an access request form can also be generated dynamically after uploading a policy. This form can be used

[4] https://github.com/DIPS-Tools/odrl-Engine/
[5] https://dips.soton.ac.uk/odrl-engine/odrl-engine-dashboard/
[6] https://github.com/paolo7/ODRL-Engine/tree/main/test_cases/evaluation

to elicit policy-relevant information from the user about the request they intend to make, to allow them to easily experiment with access requests and policies. Evaluation outputs contain a clear visualisation of the compliance result, and provide additional explanation. A video showing the OVAL web interface performing a monitoring evaluation task is available on Github.[7]

4. Conclusion and Future Work

We have presented OVAL, a novel ODRL evaluator that provides comprehensive support for all ODRL rule types, and the first to implement a published mathematical formal semantics of ODRL evaluation. Designed to be both efficient and practical, it evaluates ODRL policies over event logs represented as simple CSV files and supports both monitoring and access control scenarios. Its single-pass evaluation algorithm enables scalable evaluation of large states of the world and naturally supports streaming data. Its open-source implementation includes a web interface, a comprehensive test suite, and scripts for scalability experiments with synthetic data.

We believe OVAL provides a practical foundation for ODRL policy evaluation that further ODRL research and standardisation efforts can benefit from. Future work will expand its functionalities with support for additional types of semantics and more complex types of reasoning. However, these extensions would first need to be formalised, as they are left partially unspecified in the ODRL specification in the context of evaluation.

Acknowledgments

This research was partially funded by the Horizon Europe projects DATAPACT (No. 101189771) and RAISE Suite (No. 101188337).

Declaration on Generative AI

During the preparation of this work, the authors used ChatGPT in order to: Grammar and spelling check. After using this tool, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] H. J. Pandit, B. Esteves, Enhancing data use ontology (duo) for health-data sharing by extending it with odrl and dpv, Semantic Web 15 (2024) 1473–1498.

[2] D. Petrova-Antonova, P. Pareti, P. Tomov, S. Yumusak, C. Maidens, S. Jiang, G. Konstantinidis, D. Roman, Knowledge graph-driven policy enforcement in urban dataspace, in: International Joint Conference on Knowledge Graphs, Springer, 2025, pp. 406–421.

[3] W. Slabbinck, J. Rojas Meléndez, B. Esteves, P. Colpaert, R. Verborgh, Interoperable Interpretation and Evaluation of ODRL Policies, in: European Semantic Web Conference, Springer, 2025, pp. 192–209.

[4] A. Cimmino, J. Cano-Benito, R. García-Castro, Open Digital Rights Enforcement Framework (ODRE): from descriptive to enforceable policies, 2024. URL: https://arxiv.org/abs/2409.17602. arXiv:2409.17602.

[5] J. O. Salas, P. Pareti, A. Aslam, C. Maidens, G. Konstantinidis, A formally grounded odrl evaluator: Implementation and comparison, 2026. URL: https://arxiv.org/abs/2607.15987. arXiv:2607.15987.

[6] J. O. Salas, P. Pareti, S. Yumuşak, S. Gheisari, L.-D. Ibáñez, G. Konstantinidis, Evaluation and Comparison Semantics for ODRL, in: Knowledge Graphs, Springer Nature Singapore, Singapore, 2026, pp. 252–267.

[7] https://github.com/DIPS-Tools/ODRL-Engine/blob/main/resources/Policy_Evaluation_Demo_July_2026.mp4

[7] C. Maidens, P. Pareti, D. Roman, G. Konstantinidis, Automated compliance for data and ai pipelines: The datapact project., in: RuleML+ RR (Companion), 2025.
[8] S. Gheisari, C. Maidens, S. Yumusak, J. O. Salas, P. Pareti, G. Giamouridis, M. Hu, A. Aslam, B. P. Jaysawal, L.-D. Ibáñez, G. Konstantinidis, D. Roman, Core Services for Compliant and Trustworthy Data Marketplaces: the EU UPCAST Project, in: Companion Proceedings of the 9th International Joint Conference on Rules and Reasoning — RuleML+RR-Companion 2025: Rule Challenge, Doctoral Consortium, Industry Track, and Networking Session, volume 4083, CEUR Workshop Proceedings, 2025, p. paper 70. URL: https://ceur-ws.org/Vol-4083/paper70.pdf, cEUR-WS.org, online.
[9] P. Pareti, J. O. Salas Trejo, A. Aslam, paolo7/odrl-engine: v1.2, 2026. URL: https://doi.org/10.5281/zenodo.22714952. doi:10.5281/zenodo.22714952.