

Displaying triple provenance with extensions to the Fresnel vocabulary for semantic browsers

Lloyd Rutledge^[0000–0003–2814–9483], Pascal Mellema,
Tije Pietersma, and Stef Joosten

Open University of the Netherlands, Heerlen, The Netherlands
`Lloyd.Rutledge@ou.nl`

Abstract. We propose extensions to the Fresnel semantic browser vocabulary that displays data about triples instead of just about resources. This facilitates broadly applicable rapid prototyping of information systems that include displaying the provenance of triples. This provenance can include a triple’s role in logical conclusions, how it can be edited and its documented sources. Our contribution’s technical focus is the addition of a box for reification to Fresnel’s display model. We also propose extensions to SPARQL selectors in Fresnel for querying whole triples as context instead of single resources. We evaluate our extensions with screen displays within an illustrative scenario that applies them. Furthermore, we demonstrate our proposal’s technical feasibility with prototype software that generates these displays: a component called TransFresnel (Transparent Fresnel) in a semantic browser called RuleStyle.

Keywords: Provenance · Justification · Fresnel · Semantic browsers.

1 Introduction

Users increasingly demand from their Semantic Web interfaces not only data but also its origins. Such provenance often includes source documents, justification of conclusions, and interfaces for editing that data. This enables users, for any given unit of data they browse upon, to check if it is correct and, when needed, correct it. Of these types of provenance, presenting explanations of reasoning to end users is the primary motivating application for this work. It is also the subject of much recent research [16].

The Fresnel vocabulary for semantic browsers offers a generalized technology for designing data interfaces to Semantic Web data [11]. It takes a presentation-independent stylesheet-based approach similar to that of CSS. Thus, one Fresnel stylesheet can apply to multiple ontologies, such as one CSS stylesheet can apply to HTML from different websites. In addition, one ontology can be presented in different ways by different Fresnel stylesheets. This enables efficient creation of reusable stylesheets for browsing unfamiliar datasets in a tailored and readily adaptable style and manner.

In earlier work, we explored applying Fresnel to making rapid prototypes of basic end-user information system interfaces for data in new ontologies [14]. We implemented this in the software Fresnel Forms. Fresnel Forms automatically generates

Copyright © 2021 for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

a basic, default Fresnel stylesheet for any given ontology. In addition to generating stylesheets, this software processes Fresnel stylesheets into user interfaces that offer not just data browsing but also form-based data input. These forms also provide some basic implementation of business rules by, for example, setting logic-based constraints on what data users can enter. Therefore, Fresnel Forms shows how Fresnel can provide a rapid prototype of a basic business information system for any ontology.

While, until now, Fresnel has only been applied to displaying information about individual resources, we propose here an extension to Fresnel for specifying the display of information about triples instead of just about instances. This Fresnel extension facilitates rapid prototyping of business information systems that provide transparency for business rule logic along with the more general data. We use the Protégé ontology editor’s visual interface as an example to emulate for displaying reasoning in the Semantic Web along with its justification [7]. Validation comes in part from illustrative scenarios of this approach applied to the fictional business rule example EU-Rent [8]. This work also validates its proposals in prototype software we name TransFresnel (Transparent Fresnel), as a component in the broader Fresnel-conformant semantic browser RuleStyle.

2 Related Work

This section presents related work in the key relevant subject areas. Triple provenance display is the functionality we aim to introduce into our proposed solution. Protégé provides interface functionality that our more generalized approach should be able to emulate. EU-Rent is the often-used fictional case we validate our proposal with. Finally, Fresnel is the technological foundation for our proposed solutions. The related work this section presents includes existing ontologies that apply to our work’s semantic needs. We aim to maximize reuse of such ontologies by using them as is where possible and extending them where necessary.

2.1 Justification and other provenance

The primary technology for provenance on the Semantic Web is reification: treating a triple as a resource so it can be the subject of other triples. Its original technical specification is RDF’s `Statement` class as a domain for the properties `subject`, `predicate` and `object`. In addition, there are proposals for other arguably more efficient representations of reification than requiring four additional triples to make each triple reifiable. RSP-QL*, for example, offers both RDF and SPARQL the syntactic shortcut of encapsulating simple triple expressions in double angle brackets [5]. This work here proposes how Fresnel can recognize and display reification as encoded by a variety of ontologies for it.

The type of provenance that motivates this work is the explanation of the reasoning behind specific triples. The Semantic Web Application Platform’s (SWAP) reason ontology is a “vocabulary for proofs”². Its namespace offers the class `Inference`

² <https://www.w3.org/2000/10/swap/reason>

for inferences, the property `gives` for the triples that are inferred, and the property `evidence` for the triples causing the inference. As such, SWAP's reason ontology directly supports justification of inferred triples. We apply this ontology and these components here in our examples of how our proposal can have Fresnel present justifications of inferred triples.

The tool `Validatrr` applies these SWAP constructs in explanations it generates for SHACL constraint violations [3]. It does so by having the object of the `gives` property not be a single inferred triple, but instead by treating each violation as inferred, and thus giving the collection of inferred triples defining that violation. We propose here how Fresnel can select such inferences and display the justifications for them. `Validatrr` reifies triples with RDF containers of three resources instead of using RDF reification constructs. Fresnel SPARQL queries can query this construction of reification, and our examples in this paper do so.

Proof Markup Language (PML) represents justifications for data that Semantic Web services produce, such as inferencing [2]. PML broadened its scope to more general provenance of information and had large influence on the development of the PROV ontology for provenance [1].

In earlier work, we proposed design patterns for inferences from which reasoners generate understandable justifications [10]. Then, we used Protégé to process example code to generate illustrative scenarios. We now apply Protégé's reasoner and explanation boxes in the same manner, using the same illustrative scenario evaluation technique. More recently, we proposed a reference architecture for implementing traceability in rule-based information systems [15]. Such traceability is another form a provenance. The techniques we propose now can apply to implementing parts of that reference architecture.

2.2 Protégé

The Protégé ontology editor [7] is often used in Semantic Web research and industry. We apply Protégé screendumps here in illustrative scenarios. The focus is on Protégé's facilities for highlighting and explaining inferred triples.

Fig. 1 shows a Protégé display with a triple highlighted as inferred. Protégé shows inferred triples by giving their display a distinct style: a yellow background. In previous work, we used such Protégé displays as examples to emulate with Fresnel [6]. This current work does so as well. However, this earlier work proposed a Fresnel selector as a binary property stating whether a given triple is inferred. This current work, on the other hand, proposes extending selector queries to include contextual triples along with provenance data about them.

Fig. 1 also shows a Protégé display with an explanation for an inference. This work uses these displays to show what information an explanation offers. We then show how letting selector queries access such information helps with styling inferred triple display and showing explanation information.

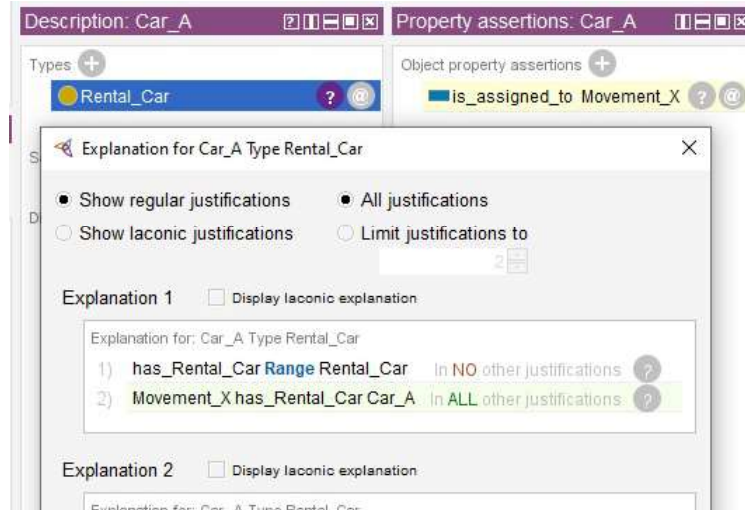


Fig. 1. Protégé display of inferred triples. One is indicated as inferred with a yellow background. An explanation box shows for another inferred triple.

2.3 EU-Rent

EU-Rent is an informative example of the business rule notation format Semantics of Business Vocabulary and Business Rules (SBVR) [8]. It provides SBVR code for the data model and business rules that define how a fictional car rental company runs. Other research has since applied EU-Rent to illustrate and evaluate other business rule formats and technologies as well. In one such work, Reynares uses the EU-Rent case as example application of a proposed mapping from SBVR to OWL-2 [12]. This section later presents our previous work on implementing business rules with Semantic Web and Fresnel technologies that uses EU-Rent in this way. We also discussed then how this previous work has built on the OWL ontology that Reynares wrote for EU-Rent. This paper here continues with EU-Rent in general, and Reynares’s OWL ontology in particular, as an example for displaying triple metadata with Fresnel stylesheets.

2.4 Fresnel

Fresnel is a Semantic Web vocabulary for defining stylesheets for how browsers display Semantic Web data to end-users [11]. Its general model is that, at each step of browsing, the interface displays all triples for the current subject’s resource in a table. The user can then click on objects for the resources displayed for a current subject to navigate to the display with that resource as the new current subject. Typically, a table for the Fresnel display of a resource shows a triple in each row with columns for the predicates and objects.

Fresnel’s box model lets stylesheet specifications apply to either the whole table, or the row for a property, or the cell for a property or object. Fig. 2 shows Fresnel’s box

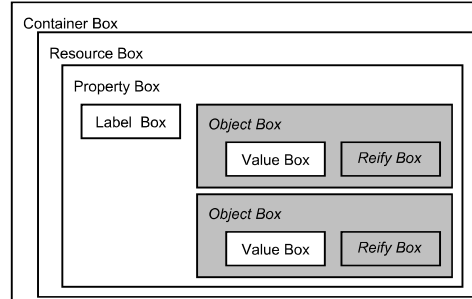


Fig. 2. The box model for Fresnel displays [11], with our extension to it in italics and gray backgrounds.

model for its browser displays [11], along with an extension to it that we propose later in this paper. The container box in this model represents an entire display at a given interaction moment on a Fresnel browser. It contains the resource box, which contains the display for all data regarding a given Semantic Web resource. Typically, this display shows the triples for which the given resource is the subject. The resource box contains property boxes, which each display a label for the property and all the objects in triples for this property and the current subject. The label box contains the label of the given property. Then the value box shows the object for a triple. Later, this paper describes the object and reify boxes that we propose as Fresnel extensions.

Each style specification, or “lens”, in Fresnel has a selector that determines which resources’ displays to apply its style to. The most general type of Fresnel selector is a SPARQL query. The URIs that a lens’s selector query returns are the resources that the browser applies the lens’s styles to.

Fresnel’s format domain properties all have selectors triggered by a single resource, be it an individual, class or property. Property format domains, on the other hand, affect the style of the cells for a given property in any table using it. If such a selector is a SPARQL query then the given style applies to the display of the current resource if the list returned contains the resource. As such, Fresnel is, up until now, resource-centric: selectors query for resources to decide how to display all triples involving that resource. We describe later how our proposed extension to Fresnel is necessarily triple-centric because the style is determined by a triple and applies to the display of that triple in the context of the broader display of all information around a resource.

2.5 Extending Fresnel beyond data browsing

Fresnel has typically focused on resource-driven browsing of existing, non-changing data. In earlier work, we explored extending Fresnel to define interfaces for form-based addition of new data by the user, implemented in software called Fresnel Forms [14]. Fresnel Forms also provides the automated generation of default stylesheets for any

given ontology. These features combined to have Fresnel Forms provide rapid prototyping of information systems by generating a browse and data input interface for any ontology.

We then zoomed in from information systems to business systems by proposing another extension to Fresnel for handling business rule violations that are triggered by data the user enters [13]. With this approach, a Fresnel stylesheet can determine whether the business system handles violations of a given rule by blocking entry of the data into the system, or by allowing it but warning the user of the violation. Thus, the block and the warning are two different styles that can apply to the same business rule. We apply a scenario from EU-Rent to illustrate and evaluate this technique.

Later work of ours has proposed extensions to Fresnel that directly specify style for presenting inferred triples, again using EU-Rent as an example [6]. Now we build upon that work by presenting a more general extension to Fresnel that enables the same styling for provenance. EU-Rent continues to apply here as well as a source for illustrative scenarios. This paper goes further by showing explanations for inferences, styling and explanations for reasoning beyond inferencing, and displaying data for triples beyond reasoning about them.

3 Method

The methodology the rest of this paper applies in its research has three components, each in each of the following two sections: specifications and validation. First, we propose new technological components for the Fresnel vocabulary in the form of their specification. These focus on a new box in the Fresnel box model that displays information about triples, as well as on how content is selected for that box and which style applies to it. This simple extension aims to reuse as much of existing Fresnel as possible and thus be minimal. Where extension to Fresnel remains nonetheless necessary, its technical form will be as consistent as possible with the rest of Fresnel.

Then we validate this extension with the design science evaluation techniques of illustrative scenarios and prototyping [9]. The scenarios implement common, existing types of reification displays. They show how one can encode Fresnel with this extension to display reification along with resource data. We present these illustrative scenarios as browser displays that demonstrate the functionality of each extension. These displays accompany the fragments of Fresnel code that generate them, which include our extensions and examples. We create data for these scenarios as instances of EU-Rent's conceptual model and business rules EU-Rent [8] as encoded in OWL by Reynares [12].

We also evaluate our proposal by developing prototype software for it: the TransFresnel component of our RuleStyle browser. TransFresnel generates the illustrative scenario displays mentioned above by processing the code we present with them. The scenarios are thus not only illustrative; our prototype implementation that generates them supports the technical feasibility of our approach.

4 Specifications

This section presents technical specifications of our proposed extension to Fresnel that account for reification triples. There are two types of specifications: new vocabulary components, and new abstract model components to which the vocabulary components apply. We use the namespace prefix `transfr:` (TRANSPARENT FResnel) for new constructs that we propose adding to Fresnel.

In Fresnel, formats encapsulate style definitions that apply to components of the box model in a display. Each format can have format domain properties. These define the domains, or patterns of displayed semantic data, to which each format applies. Fresnel has format domains for properties, classes and instances. The latter two apply to resource boxes in the box model by selecting the resources to display there. Property format domains, on the other hand, specify which properties receive the given style in a property or label box. Each current Fresnel format domain regards a single URI: that of a class, instance or property. The extensions we propose ahead in this section each addresses the need of applying style to the display of a triple and its provenance instead of a single resource.

4.1 Reify and object boxes

Fig. 2 shows the current Fresnel box model along with the extension we propose to it. This extension is the addition of two boxes to the model: the object box, and the reify box. The reify box is the core of this extension. It comes just after the value box in the Fresnel box model.

A reify box is similar to a resource box in that it displays triples with a given subject. The key difference is that, while this subject is typically a single resource for a resource box, for a reify box, the subject is itself a triple. Furthermore, while a resource box is typically an entire, independent display, a reify box is linked to a value box, joined in an object box, that is itself a cell, or item in a cell, within a larger tabular resource box.

The reify box is a place where a Fresnel stylesheet can place information from reifications for the triple associated with the adjacent value box. When using the example display of Protégé from Fig. 1 as an analogy, the reify box would be where the buttons with the '?' and '@' icons are. The object box in our extension supports the reify box by keeping it attached to the value box for the triple it describes. It also allows the value and reify boxes to be styled together as all information about the given triple.

4.2 SPARQL selectors with triples as context

The coming subsections propose new format domains for Fresnel. Since Fresnel's previous format domains all have individual resources as context, their SPARQL selectors each return a single variable binding. When this matches a resource, the resource falls in the domain and is thus displayed with the associated style.

Our new format domains' types of triggers no longer have single resources as context but instead entire triples. This can affect SPARQL selectors in two ways. One is that the SPARQL query returns not one but three variable bindings, which then must

match the three URIs in a triple. The other effect is our addition of three prebound variables that let SPARQL query with the given triple as reference context. The subsections ahead describe how each of these is used.

4.3 `transfr:valueFormatDomain` and `transfr:objectFormatDomain`

In Fresnel's current box model, the value box is what corresponds best to the display of a triple. The value that each Fresnel value box shows is the object in a triple with the content of the preceding property box as the property and the resource of the encapsulating resource box as the subject. While Fresnel provides the value box for displaying triples, it provides no format domain for it, and thus no means of tailoring the style of the value it presents based on its triple.

Here we propose the property `transfr:valueFormatDomain` as an extension for Fresnel. This specifies which triples have the presentation style of the given value box. It applies as well to the object and reify boxes we propose in this work. SPARQL selector queries that value format domains use return not one but three variables, which correspond to the three URIs of the context triple displayed in a given value box.

Our extension adds an object box that encapsulates Fresnel's value box with our extension's reify box. To be able to format both the value and object boxes together through their object box, we also propose extending Fresnel with the `transfr:objectFormatDomain` property. This works for object boxes equivalently to how `transfr:valueFormatDomain` works for value boxes.

4.4 `transfr:reifyFormatDomain` and prebound triple variables

In our proposed additions to Fresnel, the property `transfr:reifyFormatDomain` specifies the format domain for the reify box. As with our value format domain, we propose some unique characteristics of SPARQL selector queries for reify format domains. One is that it returns a single variable, instead of the three variables for the reified triple by value format domains. Furthermore, this single variable is a resource for reification information for the triple, rather than for the context to be matched. The URI bound to this single returned variable lets the browsers put a link to it in the reify box.

In order to let SPARQL selector queries in reify format domains still match context reified triples, we also propose adding three prebound variables to represent the triple. These are: `$thisSubject`, `$thisPredicate` and `$thisObject`. They are replaced by the subject, predicate and object of the triples being displayed in such SPARQL queries. In this way, the queries get not just the context of triples to apply their style to, but they can be tailored to return resources containing reifications for the triples.

4.5 `transfr:reifyStyle` and `transfr:reifyLabel`

Fresnel provides what it calls styling hooks for each box in its model. Each type of box has gets a hook as a property that assigns a format a string of CSS code for styling it. Since our extension to Fresnel adds a new box and corresponding format domain

for reifications, we add a styling hook for them as well. This extension is the property `transfr:reifyStyle`, which applies to the reify box. We also proposed adding the property `transfr:reifyLabel` for reify formats, which lets a string of text serve as the content of reify boxes styles by the formats. It is based on the existing property `fresnel:label` for property formats, which defines the property-describing text content of label boxes. In addition, this text serves as a link to the browser lens display for the reification.

5 Validation scenarios and prototype

This section evaluates our proposed technical specifications from the previous section by applying them to implement the EU-Rent case and emulate its display on Protégé. It uses EU-Rent for illustrative scenarios to which to apply the Fresnel extensions. We also present a prototype for these extensions that generates displays for these scenarios.

5.1 Style: Yellow background for inferred triples

There are several ways one could encode the inclusion of explanation triples in what Fresnel selectors query. Fig. 3 shows the display from our browser RuleStyle highlighting an inferred triple. Fig. 3 is our recreation of the Protégé display in Fig. 1 with the same Semantic Web code processed instead with RuleStyle. Fig. 4 shows how we set Protégé's style for inferred triples in Fresnel.

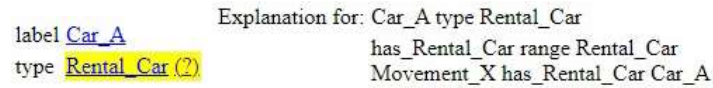


Fig. 3. Prototype displays for the style in Fig. 4 and the explanation in Fig. 6.

```
ex:infFormat rdf:type fresnel:Format ;
transfr:objectFormatDomain ""
SELECT ?thisSubject ?thisPredicate ?thisObject
WHERE { ?Inferred a reas:Inference ;
  reas:gives/rdf:first ?thisSubject ;
  reas:gives/rdf:rest/rdf:first ?thisPredicate ;
  reas:gives/rdf:rest/rdf:rest/rdf:first ?thisObject .
}""^^fresnel:sparqlSelector ;
fresnel:objectStyle "background-color:yellow"^^fresnel:stylingInstructions .
```

Fig. 4. Fresnel code with CSS code emulating Protégé's styling of inferred triples.

5.2 Links: Question mark link to justification display

Fig. 5 shows how we put a question mark as an anchor to explanations for triples in Fresnel. The tiny string with the question mark can be replaced with others for other types of reifications, such as with square brackets to represent citations.

```
ex:reifLinkFormat rdf:type fresnel:Format ;
transfr:reifyFormatDomain """
  SELECT ?Inferred WHERE { ?Inferred a reas:Inference ;
    reas:gives/rdf:first $thisSubject ;
    reas:gives/rdf:rest/rdf:first $thisPredicate ;
    reas:gives/rdf:rest/rdf:rest/rdf:first $thisObject .
  } """^^fresnel:sparqlSelector ;
transfr:reifyLabel "(?) "^^xsd:string .
```

Fig. 5. Fresnel code emulating Protégé’s links to explanation boxes.

5.3 Explanation boxes

Fig. 6 shows Fresnel code for the illustrative scenario for displaying justification triples. Here, the query applies to data using the SWAP ontology as implemented in Validatrr [3]. It selects members of the `Inference` class.

5.4 Prototype

The code for the software and the demonstrations we present here for it are available online¹. This prototype software is the partial Fresnel browser `RuleStyle`, which we started as a proof of concept and feasibility study for this paper. It generates the screen displays shown in Fig. 3. It validates the scenarios presented here by implementing them, showing that a program can have the scenarios function as shown. We have thus implemented in `RuleStyle` no more than is needed to generate the scenario screen displays from the code in this paper.

`RuleStyle` is a PHP server script that references a SPARQL endpoint to collect and display data regarding the given resource. The SPARQL endpoint we set up for our installed `RuleStyle` server is Jena Fuseki. We put the N3 scenario code in a larger N3 file with contextual code, which we then convert to XML RDF for Jena. This contextual code includes fragments of the EU-Rent ontology and our small test population for it. It also includes inferences from this population code generated by Protégé. The explanation code is typed in by hand from Protégé explanation box displays.

As of this writing, `RuleStyle` provides generalized implementation for Fig. 4’s example of the styled display of inferred triples. For the link implementation in Fig. 5,

¹ <https://github.com/LloydRutledge/RuleStyle>

RuleStyle retrieves and presents the icon in a generalized fashion. However, the pre-bound variables in the selector still need implementing. Finally, RuleStyle supports the selection of the explanation box lens in Fig. 6.

```
ex:explBox a fresnel:Lens ; fresnel:instanceLensDomain ""
  SELECT ?inference
  WHERE {?inference a reas:Inference}""^^fresnel:sparqlSelector ;
  fresnel:showProperties (
    [ fresnel:property reas:gives      ; fresnel:sublens ex:stmtLens ]
    [ fresnel:property reas:evidence ; fresnel:sublens ex:evidLens ] ) .

ex:stmtLens a fresnel:Lens ;
  fresnel:showProperties ( rei:subject rei:predicate rei:object ) .

ex:evidLens a fresnel:Lens ;
  fresnel:showProperties ( rdf:first
    [ fresnel:property rdf:rest ; fresnel:sublens ex:evidLens ] ) .
```

Fig. 6. Fresnel code for displaying an explanation box.

6 Conclusion

This paper adds triple provenance display to Fresnel by extending its box model and SPARQL selectors. The new reify box can contain links to reification information for triples displayed. SPARQL selectors can now acknowledge triples as context instead of just single resources, and return reification information links for contextual triples. This work builds upon the Master's theses of Pascal Mellema [6] and Tije Pietersma [10].

This work represents several iterations of a design science approach. Additional potential extensions to Fresnel these iterations motivate include having the reify box selectors return a property with the object, and extending `fresnel:showProperties` to include property paths, perhaps in the form of semantic paths [4]. Having reify box selectors return an additional variable binding for a property along with that for the object it links to would let the browser know what type of provenance it provides. This would enable adapting our explanation link example to present other icons for other types for provenance, such as Protégé's '@' for annotation, brackets for citations such as in Wikipedia infoboxes, and links for editing the data.

In addition, Fresnel's `fresnel:showProperties` list currently only contains single URI's. We would like to extend this to include SPARQL property paths as well. This would facilitate presenting SWAP triples because its members are in an RDF list. Property paths could locate each triple member in the list so our extended Fresnel treats is as a property of the triple and displays it as such.

Other potential future work for which this new technology lays a foundation includes generating human readable explanations. To do this, specialized lenses can select justification triple sets that match readable text templates with SPARQL queries

for given shapes. Finally, browsers such as Rule Style that implement this provenance-based extension to Fresnel can apply it in broader business information system research.

References

1. PROV-Overview. An Overview of the PROV Family of Documents. Project report (April 2013), <https://eprints.soton.ac.uk/356854/>
2. da Silva, P.P., McGuinness, D.L., Fikes, R.: A proof markup language for Semantic Web services. *Information Systems* **31**(4), 381 – 395 (2006). <https://doi.org/https://doi.org/10.1016/j.is.2005.02.003>
3. De Meester, B., Heyvaert, P., Arndt, D., Dimou, A., Verborgh, R.: RDF Graph Validation Using Rule-Based Reasoning. *Semantic Web Journal* (2020)
4. Destandau, M., Appert, C., Pietriga, E.: S-Paths: Set-based visual exploration of linked data driven by semantic paths. *Semantic Web* **12**(1), 99–116 (01 2021). <https://doi.org/10.3233/SW-200383>
5. Keskiärrkkä, R., Blomqvist, E., Lind, L., Hartig, O.: RSP-QL: Enabling Statement-Level Annotations in RDF Streams, pp. 140–155 (11 2019). https://doi.org/10.1007/978-3-030-33220-4_11
6. Mellema, P.: Extending semantic browser style specifications for Semantic Web inferencing. Master's thesis, Open University of the Netherlands, Heerlen, The Netherlands (2020)
7. Musen, M.A., Protégé, T.: The Protégé Project: A Look Back and a Look Forward. *AI matters* **1**(4), 4–12 (2015). <https://doi.org/10.1145/2757001.2757003>
8. Object Management Group: Semantics of Business Vocabulary and Business Rules (SBVR), Appendix G - EU-Rent Example (2016)
9. Peffers, K., Rothenberger, M., Tuunanen, T., Vaezi, R.: Design science research evaluation. In: *Proceedings of the 7th International Conference on Design Science Research in Information Systems: Advances in Theory and Practice*. p. 398–410. DESRIST'12, Springer-Verlag, Berlin, Heidelberg (2012)
10. Pietersma, T.: Ontologie-ontwerppatronen voor gevolgtrekkingen met automatische uitleg. Master's thesis, Open University of the Netherlands, Heerlen, The Netherlands (2019)
11. Pietriga, E., Bizer, C., Karger, D., Lee, R.: Fresnel: A browser-independent presentation vocabulary for RDF. In: *International Semantic Web Conference*. pp. 158–171. Springer (11 2006)
12. Reynares, E., Caliusco, M., Galli, M.: SBVR to OWL 2 mappings: An automatable and structural-rooted approach. *CLEI Electronic Journal* **17** (12 2014). <https://doi.org/10.19153/cleiej.17.3.2>
13. Rutledge, L., Bouwer, E., Joosten, S.: Rule style: Patterns of and extensions to data system user interface specification for business rule violations. In: Shishkov, B. (ed.) *Business Modeling and Software Design*. pp. 3–16. Springer International Publishing, Cham (2019)
14. Rutledge, L., Brenninkmeijer, T., Zwanenberg, T., van de Heijning, J., Mekker, A., Theunissen, J.N., Bos, R.: From ontology to semantic wiki – designing annotation and browse interfaces for given ontologies. In: Molli, P., Breslin, J.G., Vidal, M.E. (eds.) *Semantic Web Collaborative Spaces*. pp. 53–72. Springer International Publishing, Cham (2016)
15. Rutledge, L., Italiaander, R.: Toward a reference architecture for traceability in SBVR-based systems. In: *Proceedings of the Seventh International Workshop on Controlled Natural Language (CNL 2020/21)*. Special Interest Group on Controlled Natural Language, Amsterdam, Netherlands (Sep 2021), <https://aclanthology.org/2021.cnl-1.13>
16. Wärtl, B., Vogl, R.: Explainable artificial intelligence: The new frontier in legal informatics. *Jusletter IT* **4**, 1–10 (2018)